Universidad Tecnológica de la Mixteca

Implementación y análisis comparativo de una red neuronal ART2 y una Backpropagation, aplicadas al reconocimiento de patrones en imágenes digitales.

T E S I S

QUE PARA OBTENER EL TÍTULO DE:

Ingeniero en computación

Presenta
María Luisa Cruz López.

Asesores

Dr. José Javier Báez Rojas.
M.C. Raúl Cruz Barbosa.

Huajuapan de León, Oaxaca noviembre 2001

A mis padres

Ma. Olivia López Martínez
Wilfrido Cruz Gutierrez †

Quienes con su amor y dedicación
me han dado la mejor herencia en esta vida,
mi educación.

A mis amigos y familiares

Quienes con su apoyo y ánimo me han ayudado a
superar los momentos difíciles de mi vida.

Agradecimientos:

A mis asesores:

Dr. José Javier Bâez Rojas
M.C. Raúl Cruz Barbosa.

Por su apoyo, paciencia y buenos consejos.

A mi universidad

Por haberme dado la oportunidad de lograr
uno de mis más grandes anhelos,
la conclusión de mi carrera.

A Karla y Guille
Por toda su ayuda y comprensión.

En especial al Instituto de Astrofísica Optica y Electrónica.

Por el apoyo y las facilidades brindadas para la realización de esta tesis.

Contenido

Capítulo 1: Generalidades

1.1 Introducción.

El ser humano en su afán por mejorar sus condiciones de vida ha desarrollado una serie de máquinas y herramientas para realizar diferentes tareas. Las primeras máquinas fueron hechas para ejecutar tareas mecánicas, simples y repetitivas, como son los motores mecánicos, podadoras, sierras, en fin una gran serie de aparatos eléctricos. Pero conforme la tecnología se fue desarrollando las máquinas se hicieron más especializadas y capaces de realizar tareas más complejas cada vez. Así, nacieron los autos, aviones, locomotoras y otras máquinas complejas. Luego, el hombre paso del campo de las tareas mecánicas al campo de las tareas intelectuales y surgieron las calculadoras, computadoras, robots, etc.

Así, el último reto impuesto por el hombre, es la construcción de máquinas que sean capaces de reproducir procesos con cierto grado de inteligencia. Como son el aprendizaje de nuevos conocimientos, el reconocimiento de imágenes y de voz, el desplazamiento de robots en ambientes no controlados, etc. Para poder lograr esto, primero se ha abierto una gran polémica acerca de la propia definición de inteligencia, la cual todavía no se resuelve de una manera general; por lo que algunos científicos han optado por enfocar sus esfuerzos en descubrir cómo se llevan a cabo los procesos llamados inteligentes, en vez de resolver la pregunta de ¿qué es la inteligencia?. Por lo tanto se han desarrollado varias teorías acerca de cómo el hombre realiza estos procesos llamados inteligentes. Dando origen al nacimiento de una nueva línea de investigación, la Inteligencia Artificial que a su vez se divide en varias subáreas como son los sistemas expertos, la lógica difusa, las redes neuronales, la teoría de probabilidad. Donde cada una de estas subáreas está basada en una teoría diferente sobre cómo el hombre realiza los procesos inteligentes. Por ejemplo, los sistemas expertos se basan en el uso de una base de conocimientos y un motor de inferencia. Mientras que las redes neuronales toman como modelo la forma en que funciona fisiológicamente el cerebro humano. Será precisamente esta subárea, las redes neuronales uno de los temas tratados en este trabajo, junto con el procesamiento digital de imágenes.

Las redes neuronales se aplican a la solución de diversos problemas relacionados con procesos inteligentes, por lo que es necesario definir sobre que campo se trabajará. Ya que dentro de todos los procesos inteligentes, el reconocimiento de imágenes es uno de los cuales usamos desde que somos pequeños y lo aprendemos de una forma natural durante los primeros meses de vida, llegando a ser después uno de los más utilizados durante toda la vida para la mayoría de los seres

humanos, es comprensible que sea uno de los temas de más investigación en la actualidad y será el tema sobre el cual se aplicarán las redes neuronales para el desarrollo del presente trabajo.

Para el ser humano la vista es uno de los sentidos por los que obtiene mayor cantidad de datos, "una imagen dice más que mil palabras", este es un dicho popular que encierra una gran verdad, puesto que las imágenes suelen contener una gran cantidad de información. Sin embargo, depende de las habilidades y conocimientos de la persona que observa la imagen, la cantidad de información que se puede obtener de dicha imagen, por ejemplo, un geólogo puede saber que tipos de suelo existen en una región, al ver las correspondientes fotografías áreas, debido al color y texturas presentadas en la fotografía, pero sin ir tan lejos, existen patrones que la mayoría de seres humanos aprendemos a reconocer desde una edad temprana, como son los rostros de las personas y las formas de los objetos comunes, como mesas, sillas, casas, etc. Teniendo en cuenta todo lo anterior, el reconocimiento de patrones dentro de imágenes es uno de los procesos inteligentes que más se ha tratado de reproducir por medio de las computadoras.

1.2 Planteamiento.

Las redes neuronales ya llevan más de 60 años de existencia, pero ha sido en los últimos 30 años en los que han tomado mayor fuerza, y se ha difundido tanto su aplicación como su enseñanza.

En la actualidad, tanto las redes neuronales como el reconocimiento de patrones en imágenes digitales son temas actuales de investigación por numerosos grupos científicos de distintas universidades a lo largo de todo el mundo. Algunos grupos se enfocan en solo uno de los dos temas y otros trabajan en la combinación de los dos, logrando buenos resultados en distintas aplicaciones, e incluso ya empiezan a circular varios libros que tratan sobre la utilización de las redes neuronales en el reconocimiento de patrones en imágenes.

Para poder comprender mejor un tema, siempre es recomendable contar con algunos ejemplos que nos permitan ver la aplicación que nos aclare el tema en cuestión. Como una opción se presenta este trabajo, en el cual se realiza la implementación y comparación de dos redes neuronales aplicadas al reconocimiento de patrones de imágenes digitales, con el fin de ofrecer al estudiante un ejemplo de cómo funcionan dos redes neuronales ante el mismo problema, destacando las ventajas y desventajas de cada una de ellas, poniendo así de manifiesto algunas de las principales características de las redes neuronales y un ejemplo de su aplicación al reconocimiento de patrones en imágenes digitales.

1.3 Objetivo.

Instrumentar y realizar un análisis comparativo del funcionamiento entre una red neuronal basada en la teoría de la resonancia adaptativa (ART) y una red neuronal basada en la propagación del error hacia atrás(Backpropagation), aplicadas al reconocimiento de patrones en imágenes digitales.

1.4 Universo y limitaciones del estudio.

Para lograr este objetivo, en el presente trabajo se realizó la implementación de una red neuronal, basada en la teoría de resonancia adaptativa y enfocada al reconocimiento de patrones. Esta red funciona tanto como reconocedor, como clasificador de imágenes, ya que es la propia red la que le asigna una clase a cada imagen que procesa. Alternadamente se presenta una red neuronal basada en la regla delta generalizada o propagación del error hacia atrás, la cual necesita de una etapa de aprendizaje donde el usuario determina la salida que tendrá la red ante cada tipo de entrada que se le presenta, ambas redes fueron probadas con los mismos datos, para poder hacer un análisis comparativo entre los resultados que arrojen. Resaltando al mismo tiempo algunas de las principales características de las redes neuronales. Las dos redes fueron implementadas en un programa computacional.

Las imágenes utilizadas para probar estas redes se dividen en cuatro grupos diferentes, los dos primeros grupos contienen figuras geométricas, el tercero letras, y último grupo está compuesto de rostros humanos. Esto con el fin de poder apreciar el funcionamiento que presenta las redes ante cada uno de estos tipos de imágenes. Pero en ninguno de los casos se presenta una solución definitiva al reconocimiento de imagen de este tipo, ya que el reconocimiento completo de uno solo de estos tipos de imágenes requiere de un análisis más completo y representaría otro tema de tesis.

Debido a que la red ART trabaja como un clasificador, asignando una salida para cada entrada de acuerdo al patrón recibido. La red backpropagation se programó para que produzca una salida similar a la de la ART. Esto con el fin de poder realizar la comparación en circunstancias similares. Aunque la red backpropagation se puede programar no solo como un clasificador, sino tambien como un generador de claves, o bien como un reconstructor del patrón de entrada, estos temas no serán tratados en este trabajo.

Para que el sistema sea un poco más versátil, se presenta una serie de métodos de mejora de imagen con los cuales se podrá eliminar ruido y resaltar algunas características de la imagen, para poder introducirlas en la red con mejores resultados que sí se introducen las imágenes originales. Siendo factible entonces hacer experimentos con otros grupos de imágenes. Esto permitirá mejorar así el aprendizaje y comprensión del funcionamiento de las redes neuronales y el reconocimiento de imágenes.

1.5 Organización de la tesis.

En este primer capítulo, se ha realizado una presentación general del objetivo de este trabajo, así como sus limitaciones y sus expectativas.

En el segundo capítulo, se presentan algunos de los conocimientos elementales del procesamiento de imágenes. Para tener un panorama general de esta área, también aparecen varios métodos para la extracción de algunas características de las imágenes digitales, estos métodos serán utilizados más adelante para producir las entradas de las redes.

En el capítulo tres, se explica primero la teoría general de las redes neuronales, así como algunos de sus fundamentos y características principales. Esto con el fin de otorgar las bases teóricas suficientes para comprender los modelos de redes presentados en este trabajo. Después se hace una explicación a detalle de la teoría de cada una de las redes implementadas durante el desarrollo del trabajo que son la ART1, ART2 y la Backpropagation. Al final de cada explicación se presentan los algoritmos de cada red. De la red ART2 se presentan tres versiones y se hace un pequeño análisis de las diferencias entre estas versiones.

En el capítulo cuatro, se encontrará la descripción de la implementación de cada uno de los modelos vistos a detalle en el capítulo tres. Esta descripción incluye la forma en que se programó cada algoritmo, los problemas encontrados durante este proceso y las variables necesarias para la ejecución de cada red.

El último capítulo, el número 5, contiene los resultados obtenidos con cada una de las redes implementadas, estos resultados fueron obtenidos al aplicar la misma base de datos a las redes ART2 y backpropagation y nos sirven como base para hacer la comparación del funcionamiento de las dos redes, la cual se presenta en este mismo capítulo en el apartado de conclusiones. Para finalizar se presenta una serie de sugerencias para mejorar las implementaciones aquí desarrolladas o para la realización de algunos otros trabajos relacionados con redes neuronales.

Capítulo 2: Procesamiento de imágenes

En este capítulo veremos que es el procesamiento digital de imágenes, las etapas que lo componen y cual es el objetivo de cada una de ellas, también se presenta paso a paso como una computadora llega al reconocimiento de patrones en imágenes digitales.

2.1 Principios del procesamiento de imágenes.

El Procesamiento Digital de Imágenes (PDI), son los procesos por los cuales pasa una imagen digital con el fin de mejorarla y extraer algunas de sus características. El PDI comprende: una parte física, las cámaras o instrumentos para obtener la imagen; una parte teórica, los métodos para realizar los procesos de mejora o extracción y una parte computacional, que es la programación de estos métodos. [González 1996]

En este trabajo nos enfocamos principalmente en la parte teórica y en el software, dejando de un lado la parte física.

2.1.1 Características de una imagen digital.

Una imagen digital es una función bidimensional de la intensidad de luz f(x,y), donde x e y representan las coordenadas de un punto de la imagen. Este punto es denominado elemento de la imagen, pixel o pels (abreviatura de "picture elements"), y el valor de f en un punto cualquiera (x,y) se refiere al brillo de la imagen en ese punto.[González, 1996]

La resolución de una imagen digital es el grado de detalle con que se logra capturar la imagen original [González 1996] y depende de dos factores fundamentalmente: el tamaño del muestreo y el número de colores o niveles de grises utilizados.

El muestreo de una imagen es el tamaño real que tendrá cada pixel dentro de la imagen real, entre más pequeño sea, mayor será la resolución de la imagen.

El número de niveles de gris a utilizar en una imagen dependen del número de bits que se utilizan para guardar cada pixel, por lo que existen imágenes de 2, 8, 16, 32 , 64, 128, 256, 512 y 1024 tonos de grises.

Conforme aumentan estos dos factores aumenta la nitidez de la imagen, pero también aumentan el espacio de almacenamiento y los recursos para el procesamiento [González 1996]

Para facilitar el procesamiento se optó por utilizar imágenes en tonos de grises de 20 x20 y de 256 x 256 pixeles.

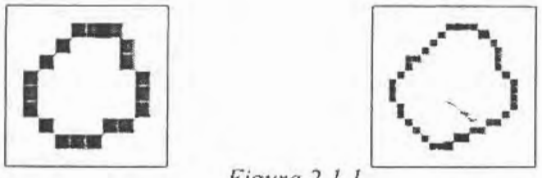


Figura 2.1.1
Imágenes con diferente resolución, la primera tiene una resolución menor que la segunda por lo que la imagen tiene menos definición.

2.1.2 Etapas.

El PDI se compone de las siguientes etapas [González 1996]: adquisición de imágenes, preprocesado, segmentación, representación y descripción y reconocimiento e interpretación.

Todas estas etapas interacturan con una base de conocimiento, como se muestra en el siguiente diagrama:

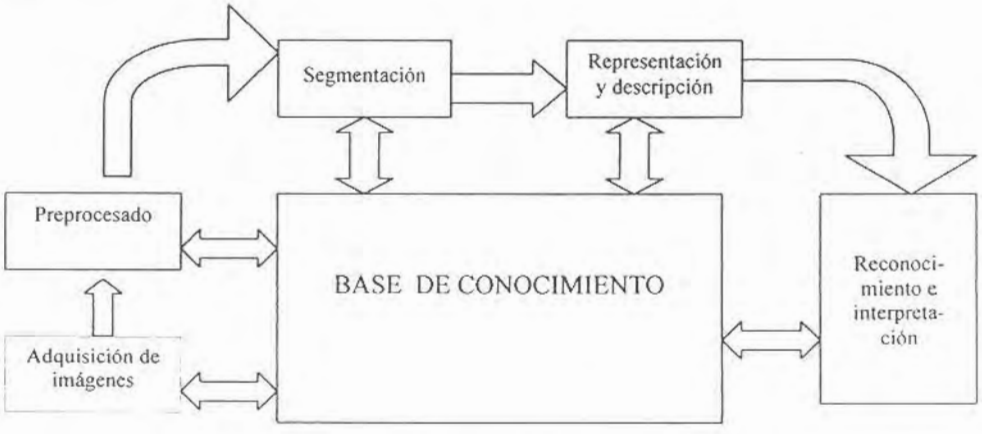


Figura 2.1.2
Diagrama de las etapas del procesamiento digital de imágenes.

La adquisición de imágenes se refiere a la obtención de una imagen digital a través de un sensor de una imagen del mundo real, o bien a la creación de una imagen digital. El preprocesamiento de la imagen es el proceso mediante el cual se mejora la calidad de la imagen

para facilitar la tarea en las siguientes etapas. La segmentación es dividir la imagen en los objetos que nos interesen. La representación, es la representación codificada de los datos obtenidos de las etapas anteriores. La descripción es la selección de los rasgos de la imagen que nos interesan. Por último, el reconocimiento y la interpretación son darle un significado a los resultados obtenidos anteriormente.

Dependiendo de la calidad, origen, y contenido de las imágenes procesadas, será necesario utilizar las diferentes etapas. Para el desarrollo de este trabajo se utilizaron sólo la segmentación, la representacion y el reconocimiento.

2.2 Métodos para el procesamiento de imágenes.

Para poder extraer las características principales de las imágenes y poder así reconocerlas, se utilizaron dos métodos de extracción. Los cuales son: los momentos invariantes y la trasformada rápida de Fourier discreta.

2.2.1 Momentos

Los momentos, dentro del análisis de imágenes se utilizan como vectores de características, atributos de textura o descriptores de formas de objetos[MICHELI 1999].

Los momentos pueden servir como descriptores de contorno o como descriptores de regiones, dependiendo de la dimensión de los datos a utilizar. Si un contorno se expresa como una función dependiente de una variable, el momento nos dará una descripción de dicho contorno. Mientras que si la función de entrada depende de dos entradas, los momentos serán descriptores de regiones dentro de la imagen.

Momentos invariantes.

A principios de los 60 Hu desarrolló 7 momentos invariantes a la traslación, rotación y escala[MICHELI 1999]. Estos momentos invariantes están definidos en función de los momentos centrales y estos a su vez en los momentos simples, así que empezaremos por ver la definición de cada uno de estos tipos de momentos.

Los momentos de una función bidimensional discreta, se definen como:

$$m_{pq} = \sum_{x}\sum_{y} x^{p} y^{q} f(x,y)$$

2.2.1.1

Que es la que se utilizará para una imagen bidimensional, donde

$f(x,y)$, es el valor de la intensidad del pixel en la posición x,y.

x indica la columna del pixel dentro de la imagen.

y indica el renglón del pixel dentro de la imagen.

$p+q$ representan el orden del momento y toman valores de 0, 1, 2, 3,...

Los momentos centrales son invariantes a la traslación, donde x'=x+α , y'=y+β y están dados para una función discreta por:[JAIN 1989]:

$$\mu_{pq} = \sum_{x}\sum_{y}(x-\bar{x})^{p}(y-\bar{y})^{q} f(x,y) \qquad 2.2.1.2$$

donde

$$\bar{x} = \frac{m_{1,0}}{m_{0,0}}$$

$$\bar{y} = \frac{m_{0,1}}{m_{0,0}}$$

Sustituyendo valores en la ecuación, se puede obtener la definición de los momentos centrales de hasta orden 3 en función de momentos simples de acuerdo a la siguiente relación[GONZALEZ 1992].

$$\mu_{00} = m_{00}$$
$$\mu_{10} = 0$$
$$\mu_{01} = 0$$
$$\mu_{20} = m_{20} - \bar{x}m_{10}$$
$$\mu_{02} = m_{02} - \bar{y}m_{01}$$
$$\mu_{11} = m_{11} - \bar{y}m_{10}$$
$$\mu_{12} = m_{12} - 2\bar{y}m_{11} - \bar{x}m_{02} + 2\bar{y}^{2}m_{10}$$
$$\mu_{21} = m_{21} - 2\bar{x}m_{11} - \bar{y}m_{20} + 2\bar{x}^{2}m_{01}$$
$$\mu_{30} = m_{30} - 3\bar{x}m_{20} + 2m_{10}\bar{x}^{2}$$
$$\mu_{03} = m_{03} - 3\bar{y}m_{02} + 2m_{01}\bar{y}^{2} \qquad 2.2.1.3$$

Los momentos normalizado son invariantes a la escala, donde x'=αx, y'=αy, y están dados por[JAIN 1989]:

$$\eta_{p,q} = \frac{\mu_{p,q}}{(\mu_{0,0})^{\gamma}} \qquad 2.2.1.4$$

y donde

$$\gamma = \frac{p+q+2}{2}$$

Utilizando estos momentos normalizados se definen los 7 momentos invariantes mencionados antes, así tenemos[HAROLICK 1993]:

$$\varphi_1 = \eta_{20} + \eta_{02}$$
$$\varphi_2 = (\eta_{20} - \eta_{02})^2 + 4\eta_{11}^2$$
$$\varphi_3 = (\eta_{30} - 3\eta_{12})^2 + (3\eta_{21} - \eta_{03})^2$$
$$\varphi_4 = (\eta_{30} + \eta_{12})^2 + (\eta_{21} + \eta_{03})^2$$
$$\varphi_5 = (\eta_{30} - 3\eta_{12})(\eta_{30} + \eta_{12})\left[(\eta_{30} + \eta_{12})^2 - 3(\eta_{21} + \eta_{03})^2\right] +$$
$$(3\eta_{21} - \eta_{03})(\eta_{21} + \eta_{03})\left[3(\eta_{30} + \eta_{12})^2 - (\eta_{21} + \eta_{03})^2\right]$$
$$\varphi_6 = (\eta_{20} - \eta_{02})\left[(\eta_{30} + \eta_{12})^2 - (\eta_{21} + \eta_{03})^2\right] +$$
$$4\eta_{11}(\eta_{30} + \eta_{12})(\eta_{21} + \eta_{03})$$
$$\varphi_7 = (3\eta_{21} - \eta_{03})(\eta_{30} + \eta_{12})\left[(\eta_{30} + \eta_{12})^2 - 3(\eta_{21} + \eta_{03})^2\right] +$$
$$(\eta_{30} - 3\eta_{12})(\eta_{21} + \eta_{03})\left[3(\eta_{30} + \eta_{12})^2 - (\eta_{21} + \eta_{03})^2\right]$$

2.2.1.5

Estos momentos también se pueden definir utilizando solo los momentos centralizados, para esto solo es necesario sustituir los momento normalizados por los centrales es decir η por μ .

2.2.2 Transformadas de la imagen.

Como ya se ha dicho antes, el procesamiento digital de una imagen se puede realizar en el dominio del espacio en el que se encuentra la imagen directamente, o bien, se puede transformar los datos de la imagen a otro espacio para facilitar su procesamiento. Como sucede con la transformada de Fourier que se utilizó en este trabajo.

Transformada de Fourier.

En el procesamiento de imágenes se utiliza esta trasformada debido a que hay operaciones que son mucho más fáciles de implementar en el dominio de la frecuencia, que en el dominio del espacio. Pero pasar al dominio de la frecuencia requiere el gasto de grandes recursos como son memoria y tiempo. Debido a lo anterior es necesario escoger con sumo cuidado el dominio en el cual se va ha trabajar.

Debido a que la función de las imágenes digitales es discreta, se utiliza la versión discreta de la transformada de Fourier. Y se escogió el algoritmo de la transformada rápida para eliminar gastos excesivos de recursos.

$$F(u,v)=\frac{1}{MN}\sum_{x=0}^{M-1}\sum_{y=0}^{N-1}f(x,y)\exp\left[-i2\pi\left(\frac{ux}{M}+\frac{vy}{N}\right)\right]$$

2.2.2.1

para

u = 0,1,2,3,...,M-1.

v = 0,1,2,3,...,N-1.

Y la antitransformada, o transformada inversa como.

$$f(x,y)=\sum_{u=0}^{M-1}\sum_{v=0}^{N-1}F(u,v)\exp\left[-i2\pi\left(\frac{ux}{M}+\frac{vy}{N}\right)\right]$$

2.2.2.2

para x = 0,1,2,3,...,M-1

y = 0,1,2,3,...,N-1.

El espectro de Fourier, la fase y el espectro de potencia son funciones discretas y dependen de dos variables, quedando las ecuaciones como siguen:

Para el espectro de Fourier.

$$|F(u,v)|=\left[R^2(u,v)+I^2(u,v)\right]^{1/2}$$

2.2.2.3

Donde R(u,v) y I(u,v) son la parte real y la parte imaginaria de F(u,v), respectivamente.

para la fase:

$$\phi(u,v) = tan^{-1}\left[\frac{I(u,v)}{R(u,v)}\right]$$

2.2.2.4

y para el espectro de potencia:

$$P(u,v) = |F(u,v)|^2 = R^2(u,v) + I^2(u,v)$$

2.2.2.5

Ahora bien, el espectro de Fourier de una imagen se presenta como una serie de arcos, donde el arco de mayor altitud es el arco centrado en la intersección de los ejes, y los demás arcos van disminuyendo su altura conforme se alejan de dicha intersección, como se muestra en la figura:

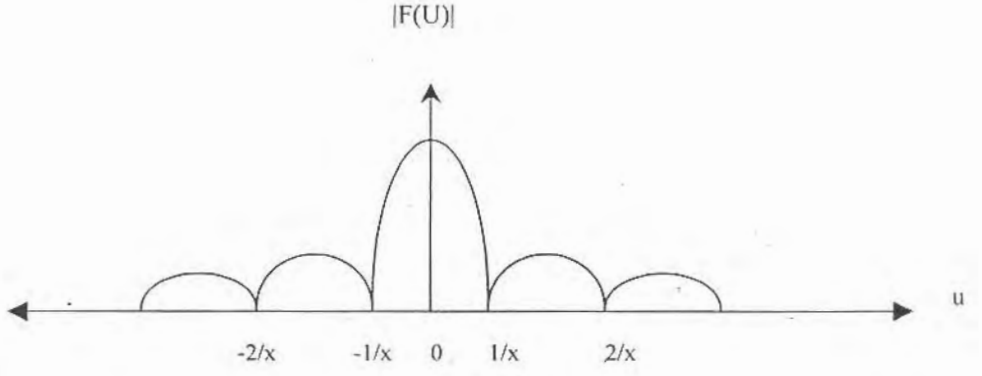


Figura 2.2.1
Espectro de Fourier [González 1996]

Donde el eje u representa la variable de frecuencia, así que se toman como frecuencias bajas las que se encuentran en el arco de mayor altura y como frecuencias altas las que se encuentran más alejadas del mismo. Esto se presenta también en la transformada bidimensional, donde las frecuencias bajas se agrupan al centro de la imagen, y desde ahí van aumentando hacia las orillas.

La transformada de Fourier posee varias propiedades que ayudan en el procesamiento de imágenes, a continuación se hace una explicación de estas propiedades.

2.2.3 Propiedades de la transformada de Fourier.

La primera propiedad que veremos es la de separabilidad, que nos dice que la transformada bidimensional discreta puede expresarse en forma separable como:

$$F(u,v) = \frac{1}{N}\sum_{x=0}^{M-1}\exp\left[\frac{-2i\pi ux}{M}\right]\sum_{y=0}^{N-1} f(x,y)\exp\left[\frac{-i2\pi vy}{N}\right]$$

2.2.3.1

para u,v = 0, 1, . . . , N-1.

La antitransformada como:

$$f(x,y) = \frac{1}{N}\sum_{u=0}^{M-1}\exp\left[\frac{i2\pi ux}{N}\right]\sum_{v=0}^{N-1} F(u,v)\exp\left[\frac{i2\pi vy}{N}\right]$$

2.2.3.2

Esta propiedad es muy útil al momento de implementar este método en la computadora, ya que nos permite obtener la transformada bidimensional en dos pasos. Obteniendo primero la transformada de Fourier de la imagen para las filas y al resultado volverle aplicar la Transformada de Fourier para las columnas. Esto se puede apreciar mejor al ver las siguientes ecuaciones.

$$F(u,v) = \frac{1}{N}\sum_{x=0}^{N-1} F(x,v)\exp\left[\frac{-2i\pi ux}{N}\right]$$

2.2.3.3

Donde

$$F(x,v) = N\left[\frac{1}{N}\sum_{y=0}^{M-1} f(x,y)\exp\left[\frac{-2i\pi vy}{N}\right]\right]$$

2.2.3.4

Lo anterior se puede representar por la siguiente figura:

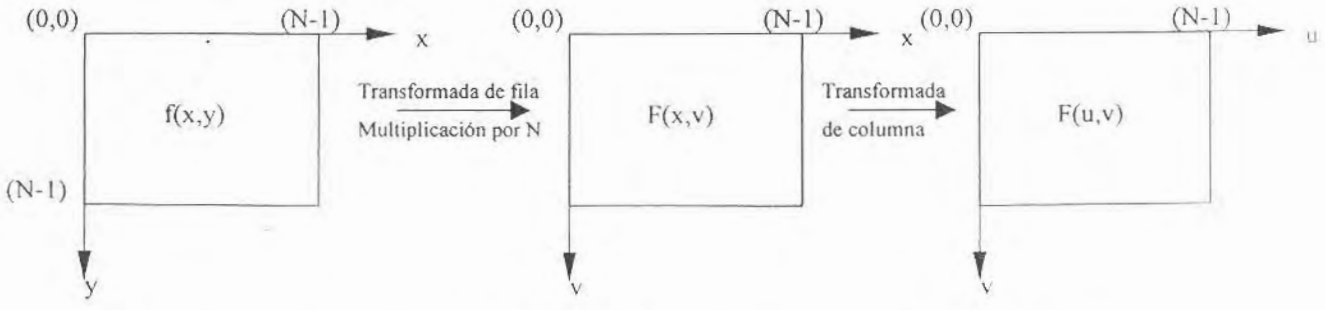


Figura 2.2.2
Esquema para obtener la transformada de Fourier en dos dimensiones [González 1996]

La siguiente propiedad a revisar es la de traslación, en la que nos dice que si multiplicamos la función F(x,y) por un exponencial dado, y se aplica la transformada de Fourier, lo que se obtiene es la misma transformada sólo que trasladada a través del plano de la frecuencia, y viceversa. Así tenemos las siguiente ecuaciones.

$$f(x,y)\exp\left[\frac{i2\pi(u_0x+v_0y)}{N}\right] \Leftrightarrow F(u-u_0,v-v_0)$$

2.2.3.5

$$f(x-x_0,y-y_0) \Leftrightarrow F(u,v)\exp\left[\frac{-i2\pi(ux_0+vy_0)}{N}\right]$$

2.2.3.6

Otra propiedad útil es la de periodicidad y simetría conjugada, la cual nos resalta que la trasformada de Fourier se repite cada determinado período N, y cada período presenta una simetría, donde la segunda parte del período es el reflejo de la primera parte.

La periodicidad se puede representar como:

$$F(u,v) = F(u+N,v) = F(u,v+N) = F(u+N,v+N)$$

2.2.3.7

Esto nos asegura que solo es necesario obtener la transformada de Fourier en un período completo para tener todos los valores de F(u,v) en el dominio de la frecuencia.

Mientras que la simetría dice que los valores de N/2+1 hasta N-1, son iguales a los valores de un rango igual a la izquierda del origen del eje de las frecuencias. Entonces tenemos que el primer período a la derecha del origen presenta dos semiperíodos opuestos en este intervalo. Por lo que para representar un período completo situado justo a la derecha del origen, es necesario trasladar la transformada de Fourier un intervalo igual a N/2, esto se puede hacer con base a la propiedad anterior de traslación.

Lo anterior nos servirá para poder centrar las frecuencias bajas en el centro de la transformada de Fourier de dos dimensiones de las imágenes.

Existen otras propiedades importantes de la transformada de Fourier, que son la de rotación, valor medio y Laplaciano, que no serán tratadas aquí puesto que no se utilizaron para el desarrollo del trabajo.

2.2.4 Transformada rápida de Fourier.

Para calcular la transformada de Fourier de un arreglo de N elementos se necesitan N^2 ecuaciones. Por lo que para una imagen N x N, primero tendríamos que sacar las transformadas de cada fila, lo que implica N^3,y repetir el proceso para las columnas, otra vez N^3 ecuaciones, lo que daría en total N^6 ecuaciones necesarias para calcular la transformada completa de la imagen, si tenemos una imagen de 20 x 20 pixeles, tendríamos que utilizar 64 000 000 operaciones, si queremos procesar imágenes más grandes esta cantidad aumentara exponencialmente y esto requeriría un gran consumo de recursos de tiempo y de memoria. Por lo anterior es conveniente utilizar el algoritmo de la transformada rápida de Fourier. Este algoritmo se basa en que la transformada de Fourier de un arreglo de N elementos es igual a la suma de la transformada de las componente pares del arreglo más la transformada de las componentes impares del arreglo por un factor. El procedimiento se puede hacer recursivo, ahorrando el calculo de muchas operaciones, reduciendo el numero de operaciones necesarias a N log N, lo que en cálculos con un N grande reduce en mucho las operaciones. La transformada rápida de Fourier también se conoce como FFT, debido a su nombre en inglés Fast Fourier Transform.

Sí representamos la transformada de Fourier como

$$F(u)=\frac{1}{N}\sum_{x=0}^{N-1} f(x)W_{2M}^{ux}$$

2.2.4.1

donde

$$W_N=\exp\left[\frac{-i2\pi}{N}\right]$$

y $N = 2^n = 2M$ donde n es un número entero positivo.

Así podemos expresar la transformada de Fourier como:

$$F(u)=\frac{1}{2M}\sum_{x=0}^{2M-1} f(x)W_{2M}^{ux}$$

2.2.4.2

$$=\frac{1}{2}\left[\frac{1}{M}\sum_{x=0}^{M-1} f(2x)W_{2M}^{u(2x)}+\frac{1}{M}\sum_{x=0}^{M-1} f(2x+1)W_{2M}^{u(2x+1)}\right]$$

2.2.4.3

De la ecuación 2.2.4.1, tenemos $W_{2M}^{2ux} = W_M^{ux}$ que por lo que la ecuación anterior la podemos sustituir por:

$$= \frac{1}{2}\left[\frac{1}{M}\sum_{x=0}^{M-1} f(2x)W_M^{ux} + \frac{1}{M}\sum_{x=0}^{M-1} f(2x+1)W_M^{ux}W_{2M}^{u}\right]$$

2.2.4.4

donde si definimos

$$F_{par}(u) = \frac{1}{M}\sum_{x=0}^{M-1} f(2x)W_M^{ux}$$

2.2.4.5

para u = 0,1,2,...,M-1

$$F_{impar}(u) = \frac{1}{M}\sum_{x=0}^{M-1} f(2x+1)W_M^{ux}$$

2.2.4.6

para u = 0,1,2,...,M-1

sustituyendo esto en la ecuación de la transformada obtenemos:

$$F(u) = \frac{1}{2}\left(F_{par}(u) + F_{impar}(u)W_{2M}^{u}\right)$$

2.2.4.7

Como se dijo antes esto se hace de forma recursiva, de modo que se pueden calcular a su vez como la suma de las transformadas de sus elementos par e impar. Esto ocasiona que los elementos se vayan procesando en un orden diferente, por ejemplo para un arreglo de 8 elementos el orden resultante será :

f(0) f(4) f(2) f(6) f(1) f(5) f(3) f(7)

Ahora bien, una manera sencilla de reordenar un arreglo siguiendo esta secuencia, es utilizando una inversión de los bits del número que especifica el lugar del elemento en el arreglo, como se muestra en la siguiente tabla. No se debe confundir la inversión de bits con el complementario binario del número, ya que no es lo mismo.

Numero de elemento	Matriz original	Inversión de bits	Matriz reordenada
0 0 0	f(0)	0 0 0	f(0)
0 0 1	f(1)	1 0 0	f(4)
0 1 0	f(2)	0 1 0	f(2)
0 1 1	f(3)	1 1 0	f(6)
1 0 0	f(4)	0 0 1	f(1)
1 0 1	f(5)	1 0 1	f(5)
1 1 0	f(6)	0 1 1	f(3)
1 1 1	f(7)	1 1 1	f(7)

Tabla 2.2.1.
Ordenamiento de los elementos de un arreglo para calcular la transformada rápida de Fourier.

Con la implementación de este método se ahorra tiempo en el uso de llamadas recursivas de funciones y en el cálculo de ecuaciones.

Ahora bien, la única restricción que existe para poder aplicar la FFT, es que N sea exponente de 2, esto asegura que cada nuevo subarreglo se pueda dividir en sus partes pares e impares. Sí no se da esta condición, al quedar un subarreglo con número impar diferente de 1 no se puede seguir dividiendo en otros subarrreglos y menos hacer el reacomodamiento por medio de la inversión de bits.

En el sistema se implemento una función que calcula la transformada de Fourier utilizando los métodos expuestos y que es propuesta en el libro de Recipes in C [PRESS 1992]. Como la transformada y la antitransformada solo se diferencian en un signo, basta poner una condición para que la misma función calcule cualquiera de las dos.

Capitulo 3: Redes Neuronales Artificiales.

En este capitulo, veremos las redes neuronales como un método para el reconocimiento de imágenes. Se hará un breve repaso sobre su fundamento biológico, sus principales componentes, ventajas y ejemplos.

3.1 Generalidades

3.1.1 Historia.

Las redes Neuronales están basadas en la estructura anatómica del cerebro humano. Las primeras teorías acerca del funcionamiento del cerebro las dieron Platón y Aristóteles, y a lo largo de la historia estas teorías se han ido transformando. Su aplicación dentro del mundo de la computación fue estudiada por primera vez por Alan Turing en 1936, y en 1943 por el neurofisiólogo Warren McCulloch, quien junto con Walter Pitts coincibió los fundamentos de la computación neuronal. A finales de los cincuentas Frank Rosenblatt desarrolló el Perceptrón que aún se utiliza en reconocimiento de patrones, también en la misma época Widrow y Hoff crearon el modelo ADALINE, que fue usada como filtro en líneas telefónicas. En los años 60's Stephen Grossberg diseñó la red Avalancha, utilizada en reconocimiento de lenguaje y movimiento de robots, pero debido a las criticas hacia este enfoque de IA las investigaciones decayeron a finales de los 60's.

Solo unos cuantos trabajos continuaron desarrollándose en la década de los 70's y principios de los 80's, pero en 1982 varios acontecimientos hicieron resurgir el interés en esta área. Primero John Hopfield presentó ante la Academia Nacional de las Ciencias un trabajo en el que muestra como pueden trabajar las redes neuronales y lo que pueden hacer. Al mismo tiempo la compañía japonesa Fujitsu empezó a desarrollar las computadoras pensantes, y se llevo a cabo la Conferencia conjunta de EUA y Japón en redes neuronales cooperativas/competitivas. Con estos eventos se dio un nuevo auge a los trabajos de redes neuronales y empezaron a surgir diferentes asociaciones y congresos a favor de las redes neuronales.

En la actualidad, cada año se presenta una gran cantidad de trabajos sobre redes neuronales, existen diversas asociaciones a favor de este tema. Se editan revistas como la Neural Networks, y también se producen diversos productos de software y hardware para el desarrollo de aplicaciones basadas en redes neuronales.

3.1.2 Definición.

No existe una definición general de las redes neuronales, para este trabajo se escogió la dada por Kohonen en 1988, que dice:

"Redes neuronales artificiales son redes interconectadas masivamente en paralelo de elementos simples (usualmente adaptativos) y con organización jerárquica, las cuales intentan interactuar con los objetos del mundo real del mismo modo que lo hace el sistema nervioso biológico" [HILERA 1995]

3.1.3 Fundamento biológico.

El funcionamiento del cerebro humano es muy complejo, por lo que solo se han tomado algunos de sus principales aspectos para el modelado de las redes neuronales artificiales.

El cerebro humano esta compuesto por células básicas llamadas neuronas, cada neurona esta compuesta por tres partes principales que son: el soma, o cuerpo de la neurona, el axón y las dendritas [ROGERS 1997]. Todas las neuronas están interconectadas entre si por medio de los dos tipos diferentes de conecciones que poseen, los axones y las denditras. Al paso de información entre las neuronas, ya sea química o eléctrica, se llama sinapsis. El conocimiento se origina cuando la información pasa a través de las diversas neuronas. Cada neurona procesa la información de acuerdo a funciones internas y genera salidas de excitación o inhibición para las neuronas vecinas, así sucesivamente hasta llegar a las ultimas neuronas y producir la salida correspondiente.

Cada neurona posee solo un axón y muchas dendritas. El axón es el encargado de conducir la salida generada por la neurona y se puede ramificar para conectarse a varias neuronas, a su vez cada neurona es capaz de recibir señales de otras neuronas al mismo tiempo por medio de diferentes dendritas. Ahora bien, las sinapsis sólo puede afectar a la siguiente neurona sí pasan de un nivel dado, conocido como umbral. Cada neurona se considera como un elemento básico de cálculo no lineal, y todas las neuronas trabajan en forma paralela.

Como actualmente la mayoría de las computadoras están basadas en el modelo Von Neumann, el cual procesa la información en serie, la implementación de las redes neuronales se hace sobre circuitos electrónicos diseñados exclusivamente para este fin como pueden ser las redes de transputers o circuitos integrados específicos, chips neuronales, etc. Sin embargo también se puede hacer una simulación de redes neuronales sobre computadoras convencionales utilizando software diseñado con este fin. Aunque este último método no aprovecha todas las ventajas de las

redes, aporta beneficios como el bajo costo y la rápida implementación. También se utiliza principalmente para el entrenamiento y evaluación de las redes.

Se puede considerar que existen tres principales tipos de redes neuronales, basándose en la naturaleza de su composición[ROGERS 1997], y así tenemos:

- Redes Neuronales Biológicas, que están formadas por elementos orgánicos, y se encuentran en los cerebros de los seres vivos.
- Redes Neuronales Artificiales,(ANN, por sus siglas en ingles, Artificial Neural Network) están formadas por elementos electrónicos. Por lo que son tangibles y su costo es elevado debido al alto número de componentes que necesita para su desarrollo. Están diseñadas para un fin especifico, y no pueden ser modificadas fácilmente. Pero ofrecen una gran rapidez al momento de su funcionamiento, ya que respetan el trabajo en paralelo, la generalización del conocimiento.
- Redes Neuronales Artificiales Simuladas,(SANN, por sus siglas en ingles, Simulated Artificial Neural Network) están basadas en algoritmos y se implementan utilizando sistemas computacionales, no son tangibles, son muy económicas y poseen una gran flexibilidad. Son ideales para probar nuevas arquitecturas de redes neuronales antes de implementarlas físicamente.

Como se puede ver de los dos tipos de redes artificiales, cada una tiene sus pros y sus contras, y dependiendo del fin que le queramos dar a nuestra red y los recursos con que contemos, será la elección de fabricar una red física o sólo simulada.

En este trabajo nos centraremos en las redes neuronales artificiales simuladas, por lo que de ahora en adelante al hacer mención a las redes neuronales, se estará refiriendo específicamente este tipo de red.

3.1.4 Ventajas de las redes neuronales

Las redes neuronales ofrecen múltiples ventajas, con respecto al tratamiento de la información sobre otros métodos de la inteligencia artificial como son los sistemas expertos, esto es debido principalmente a su arquitectura, la forma de procesar los datos recibidos y de guardar los conocimientos adquiridos. Algunas de estas ventajas son[HILERA 1995]:

Aprendizaje adaptativo: Las redes neuronales aprenden por medio de ejemplos durante una fase de entrenamiento, los elementos de las redes (las neuronas y conecciones) son dinámicos y

se van autoajustando hasta aprender toda la información recibida en la fase de aprendizaje. **Autoorganización:** La red define su organización completa (la distribución de los pesos entre las conecciones) para poder responder adecuadamente a cada entrada, esto da como consecuencia la generalización. **Tolerancia a fallos:** La cuál comprende tanto reconocer patrones con ruido, distorsión o incompletos, como, que sí la red sufre un daño parcial en su arquitectura puede seguir funcionando con un grado de tolerancia, dependiendo del tamaño del daño. Esto se debe a que la información almacenada dentro de una red no esta localizada en un lugar específico sino distribuida e incluso es redundante a lo largo de toda la red. **Operación en tiempo real:** Las redes neuronales al trabajar en forma paralela logran procesar un gran volumen de información en tiempos muy pequeños. Esto solo se logra cuando se hace una implementación física de la red, cuando se hacen simulaciones en computadoras, que procesan la información en serie, se pierde en gran medida esta ventaja. **Fácil inserción dentro de la tecnología existente:** Para tareas específicas es fácil diseñar, entrenar y comprobar el funcionamiento de una red neuronal, luego esta se puede trasladar a un circuito electrónico de fácil operación y fácil inserción en otros sistemas, esto se da únicamente con tareas bien definidas y relativamente simples, ya que en tareas muy complejas es necesario la interconección de varias redes para poder resolver con éxito el problema.

3.1.5 Aplicaciones:

Con la invención de los circuitos integrados de gran escala se ha hecho más sencillo la implementación de redes neuronales en hadware, lo que ha permitido la aplicación de esta tecnología en diversas ramas de la ciencia [HILERA 1995]como son: la biología, en la obtención de modelos de la retina; las empresas, para la explotación de bases de datos, optimización de lugares y horarios en aviones, o reconocimiento de caracteres escritos, también se aplican en el campo del medio ambiente, en la previsión del tiempo, análisis de tendencias y patrones. En otra rama donde han encontrado aplicación las redes neuronales es en las finanzas para la identificación de falsificaciones, interpretación de firmas, evaluación de riesgos en los créditos etc.; en medicina se utiliza en la ayuda de sordos profundos, análisis y diagnósticos a partir de síntomas, lectores de rayos X, análisis de las causas de los ataques epilépticos; y en la milicia para la clasificación de señales de radar, creación de armas inteligentes, y reconocimiento de objetivos de blanco.

3.1.6 Tipos de redes

De acuerdo a como se interconectan cada uno de los elementos de una red y el flujo que siguen los datos a través de la misma, se puede clasificar las redes neuronales en distintos tipos. [HILERA 1995]:

En 1957 Frank Rosenblatt desarrollo el Perceptrón, que es la red más antigua, se instrumentó en hardware y se usó para el reconocimiento de caracteres, pero tenía la desventaja de no reconocer caracteres complejos. En 1960 Bernand y Widrow desarrollaron ADALINE y MADALINE, redes de fácil implementación con circuitos analógicos o VLSI, utilizadas para filtrado de señales y como ecualizador, tienen la desventaja de no poder clasificar conjuntos linealmente no separables. Entre 1974 y 1985 K. Fukushima desarrollo el Neocognitron una red para el reconocimiento de caracteres manuscritos, insensible a la traslación, rotación y cambio de escala, pero que requiere de muchos elementos de procesos, niveles y conecciones. En este mismo periodo Werbos, David Parker, y Rumelhart desarrollaron el algoritmo de aprendizaje Back propagation, que se ha convertido en uno de los más populares, ya que tiene facilidad de aprendizaje y es potente. Su desventaja es que requiere de mucho tiempo de aprendizaje y muchos ejemplos.

En 1982 John Hopfield presentó la red con su nombre Hopfield, que se puede implementar en VLSI y se utiliza en reconstrucción de patrones y optimización, pero tiene poca capacidad y estabilidad. En 1986 Carpenter y Grossberg desarrollaron la red ART (Adaptative Resonace Teory) que es una red sofisticada de poco uso, que se aplica en el reconocimiento de patrones, además es sensible a la translación, distorsión y cambio de escala.

Estos son solo algunos de los tipos de redes más conocidos, pero de cada uno de ellos existen diferentes variaciones, además de que hay más tipos de redes, como la BAM, Counter-propagation, SOM, etc., son muchas las arquitecturas de redes neuronales artificiales existentes hoy en día, y este número crece con las propuestas de versiones mejoradas, combinaciones de redes ya existentes o planteamientos de nuevas arquitecturas. En este trabajo solo se mencionan algunas de ellas para crear un contexto que facilite el entendimiento de las dos arquitecturas que se estudiaran a detalle que son la ART y la Backpropagation.

3.1.7 Componentes de las redes neuronales artificiales.

Una red neuronal se puede definir de acuerdo con tres importantes aspectos que son su topología, el conjunto de patrones a utilizar en las etapas de aprendizaje, prueba y funcionamiento, y un conjunto de parámetros que utiliza la red durante su ejecución [ROGERS 1997]

La topología de la red es la forma en que se acomodan e interconectan las neuronas dentro de la red, en ella se pueden distinguir dos aspectos principales que son: el número de capas de neuronas, y las conecciones entre las neuronas. Las neuronas se agrupan por capas, y cada capa tiene como característica que todas sus neuronas están a una misma distancia de la entrada o salida de la red. Es decir reciben señales del mismo origen, que puede ser otra capa de neuronas o la entrada misma de la red, y envían sus señales al mismo destino, que pueden ser otra capa o la salida de la red.

Las neuronas se pueden clasificar de acuerdo al tipo de capa a la que pertenecen, y a su vez las capas se pueden clasificar según la procedencia o destino de las señales que reciben y producen [HILERA 1995].

Tenemos entonces, las neuronas de entrada, que reciben las señales del mundo real o exterior a la red, es decir, son la puerta de entrada de la red, y forman la capa de entrada. Las neuronas ocultas, que no tienen conección directa con las entradas o salidas de la red, contienen la información interna de la red, y forman las capas ocultas de la red. Una red puede tener ninguna, una o más capas ocultas. Y por último están las neuronas de salida, encargadas de producir la salida generada por toda la red al procesar la información de entrada, y forman la capa de salida.

Una red siempre tendrá una capa de entrada y una de salida, pero estas dos capas pueden ser una sola capa, entonces la red utiliza las mismas neuronas como entradas y como salidas. Para la mayoría de redes estas dos capas son diferentes, y en algunos casos existen capas ocultas entre la capa de entrada y la de salida.

Las conecciones entre las neuronas pueden ser: unidireccionales, si solo poseen un sentido, o bidireccionales, si la conección actúa en dos sentidos. También se pueden clasificar en conecciones uno a uno, cuando una neurona de una capa solo esta conectada con una y solo una neurona de la otra capa. Y conecciones una a muchas, que ocurre cuando una neurona de la capa origen envía su señal a varias neuronas de la capa destino.

El conjunto de patrones a utilizar por una red, se divide a su vez en tres subconjuntos: patrones de entrenamiento, que son los que se utilizaran para el aprendizaje de la red, y se recomienda que muestren las principales características de los patrones que se quiera que reconozca la red. Los patrones de prueba, son aquellos que se utilizarán durante la fase de aprendizaje, para comprobar que la red este aprendiendo correctamente la información. Los patrones de funcionamiento, son una muestra de aquellos patrones sobre los cuales ya va ha trabajar la red y se utilizan como prueba final.

El último aspecto de una red son los parámetros, estos por lo regular son valores que se utilizan como umbrales o mediadores de señales, y su número y valores dependen del tipo de red de la cual se trate.

Así tenemos ya definidos los aspectos más importantes de una red, y basándonos en ellos podemos tener una idea clara de la estructura de cualquier tipo de red.

3.1.8 Clasificación de redes.

Las redes neuronales se pueden clasificar de muchas formas, cada clasificación toma como base alguna de sus características topológicas o de funcionamiento.

De acuerdo al número de capas con que cuente una red se pueden clasificar en:

- Red monocapa: posee una sola capa de neuronas y las conecciones se dan de forma lateral y/o autorrecurentes, a esta misma capa llegan y salen las entradas y salidas de la red, un ejemplo es la red HOPFIELD.
- Red multicapa: esta red consta de 2 o más capas y también puede presentar conecciones laterales y/o autorrecurentes.

Debido al tipo de conecciones que presente la red se puede a su vez clasificar en redes con conecciones hacia delante y redes con conecciones hacia delante y hacia atrás. Las conecciones hacia adelante se dan si la información fluye de la entrada a la salida de la red. Por el contrario las conecciones hacia atrás se dan si la información va de la salida a la entrada de la red .

- Las redes con conecciones hacia adelante se dan cuando son estas conecciones las únicas que se presentan entre una capa y otra, en este tipo de red las conecciones laterales y autorrecurentes casi no se presentan, algunos ejemplos son el Perceptrón, ADALINE y MADALINE.
- Las redes con conecciones hacia adelante y hacia atrás, como su nombre lo dice, presentan los dos tipos de conecciones. Por lo tanto existe una retroalimentación entre las capas de la red, dos ejemplos son la red ART (Adaptative Resonance Theory) y la red BAM(Bidirectional Associative Memory).

3.1.9 Funcionamiento de las redes neuronales artificiales.

Para entender el funcionamiento de las redes neuronales, es necesario conocer primero el funcionamiento de una neurona.

Básicamente, el trabajo de una neurona es modificar la información que recibe y generar una salida que sea transmitida a otras neuronas o a la salida de la red. Por tanto, el valor de una neurona varía con el tiempo, a este valor se le denomina "estado de activación"[HILERA 1995] y se puede representar como $a_i(t)$ donde i indica la neurona a la cual se hace la referencia y depende del tiempo. Existen dos estados de activación que son: en reposo y en excitación, y sus valores pueden ser valores continuos o discretos, variando dentro de un intervalo definido para cada tipo de red.

El estado de activación presente en una neurona depende del estado anterior de activación y de la suma total de las entradas de la neurona, y esta regido por una función llamada Función de activación, la cual se puede definir como sigue:

$$\mathbf{a_i(t+1) = F(a_i(t), Net_i)} \qquad (3.1.9.1)$$

donde:

- $\mathbf{a_i(t+1)}$ Es el nuevo estado de activación de la neurona i.
- $\mathbf{a_i(t)}$ Es el estado anterior de activación de la neurona i.
- $\mathbf{Net_i}$ Es la suma total de entradas a la neurona i.

Cada señal que recibe una neurona de otra neurona a través de una conección es modificada o ponderada por el peso asignado a dicha conección, este peso indica el grado de influencia que ejercerá la neurona origen en la neurona destino. A la forma en que se combinan la señal enviada a través de una conección entre neuronas y al peso asignado a esta, se le conoce como regla de propagación [HILERA 1995], y por lo regular esta función esta dada por el producto de estos dos valores.

Con base al estado de activación la neurona calcula su salida. Esta función también recibe el nombre *de Función de transferencia o de salida*,[HILERA 1995] ya que su resultado se transfiere a otras neuronas por medio de las conecciones. Esta función puede tener varias formas, las cuatro principales son:

- Función escalón. En este caso si la suma de entradas a la neurona pasa un límite, su salida será un valor máximo o activado, en caso contrario será un valor mínimo o en reposo.

- Función lineal y mixta. En este caso si la suma esta en un intervalo, su valor será igual a la suma total de las entradas, si rebasa el intervalo tendrá el valor máximo, y si no alcanza el limite inferior del intervalo, tendrá el valor mínimo.
- Sigmodal: Esta función se parece a la de escalón, solo que en lugar de tener un umbral o punto de separación para los valores de salida, de mínimo a máximo, este cambio se va dando gradualmente, por lo tanto su derivada siempre es positiva y esto la hace más apta para algunos métodos de aprendizaje en lugar de la función escalón.
- Función gaussiana. Esta función tiene la forma de la campana de Gauss, es decir se parece a una campana, donde el centro y la anchura pueden variar.

Muchas veces la función de transferencia toma el valor generado por la función de activación, por lo que se dice que es la función identidad, ya que sus valores son idénticos.

3.1.10 Aprendizaje en una red neuronal.

Un aspecto muy importante de las redes neuronales es su mecanismo de aprendizaje, que es el proceso por medio del cual la red va cambiando el peso asociado a las conecciones que existen entre las neuronas que la componen. Con el fin de aprender la información que tiene en la entrada y producir la salida correcta. Este mecánismo tiene como base la regla de aprendizaje de la red, que es la que indica cuando y como varían los pesos asignados a las conecciones entre las neuronas.

Se dice entonces que el conocimiento dentro de una red neuronal esta representado en los pesos asociados a dichas conecciones.

Con base en el mecanismo de aprendizaje se puede clasificar a las redes neuronales en redes con:

- Aprendizaje supervisado.
- Aprendizaje no supervisado.

Dependiendo si existe un agente externo a la red que controle o no este aprendizaje.

También se puede clasificar las redes de acuerdo al tiempo durante el cual la red realiza su aprendizaje:

- Si la red puede aprender al mismo tiempo que funciona normalmente se dice que posee un aprendizaje en línea (ON LINE).
- Si se requiere deshabilitar la red para que esta pueda aprender se dice que posee un aprendizaje fuera de línea (OFF LINE).

En el primer caso, la red debe ser capaz de aprender nuevos conocimientos sin olvidar los pasado. Mientras que en el segundo caso si se agrega un nuevo patrón, es necesario volver a enseñar

todos los patrones anteriores a la red, junto con el nuevo patrón, esto para evitar la perdida de la información.

3.2 Red ART

La mayoría de las redes neuronales poseen un aprendizaje OFFLINE, es decir las etapas de aprendizaje y funcionamiento se encuentran separadas, por lo que antes de ponerlas a trabajar se les tiene que entrenar con una muestra del universo de datos que se pretenden reconocer. Lo anterior funciona muy bien para los casos en que se tiene un problema específico con un número de entradas delimitado. Si queremos procesar algún patrón que no este incluido en el conjunto de aprendizaje, la red esta no podrá reconocerlo. Por eso se tendría que volver a poner la red en estado de aprendizaje para enseñarle los nuevos patrones, lo que significa perdida de tiempo.

Algunas redes neuronales permiten aprender al mismo tiempo que funcionan, un ejemplo de esto es la red ART basada en la Teoría de resonancia adaptativa con retroalimentación desarrollada por Grossberg en 1976[PANDYA 1995], de donde proviene su nombre (Adaptative Resonance Theory ART).

La red ART fue desarrollada por Grossberg y Carpenter a mediados de los 80´s [PANDYA 1995 y HILERA 1995] y otros colaboradores. Su funcionamiento consiste en clasificar patrones de entrada en diferentes clases. Donde todos los patrones similares deben pertenecer a una sola clase y las clases son definidas por la propia red. Por lo que se dice que su aprendizaje es de tipo no supervisado. La red se basa en hacer resonar los patrones de entrada con los patrones que tiene almacenados la red. Si esta resonancia o concordancia supera un nivel dado, la red clasifica la entrada como de la misma clase del patrón con que resonó. Y procede a modificar sus pesos interiores para hacer que el patrón guardado, o prototipo, sea la mezcla del patrón antiguo más la entrada actual. Si no encuentra ningún patrón con quien la entrada resuene o coincida, la red le asigna una nueva clase y toma como prototipo el patrón de entrada.

Esta red resuelve el dilema de la estabilidad y plasticidad planteado por Grossberg. El Cual dice que una red debe ser capaz de aprender nuevos patrones, plasticidad del aprendizaje, y al mismo tiempo no olvidar los patrones ya aprendidos, estabilidad del aprendizaje. El dilema se soluciona debido al uso de un parámetro de tolerancia que es el que determina cuando un patrón será incluido o no en una clase. Esto permite que los nuevos patrones que se introduzcan a la red puedan crear nuevas clases, sin modificar las clases existentes, a menos que su parecido sea muy grande. Con lo cual la degradación de la información almacenada es muy poca.

Carpenter y Grossberg diseñaron primero la red ART1 que sólo acepta valores binarios en su entrada. Luego a finales de los 80's desarrollaron la ART2 que tiene como caracteristica que acepta valores continuos a su entrada. Estos modelos se han seguido perfeccionando, dando paso al modelo de red la ART3[PANDYA 1995].

3.2.1 Arquitectura de una red ART

La red ART, es una red compuesta solo de dos capas principales de neuronas, la capa de entrada F1 y la capa de salida F2. Estas capas están unidas por medio de conecciones hacia adelante y hacia atrás. Además la capa de salida cuenta con conecciones laterales y autorrecurentes. En la red existen otras neuronas, que no forman parte propiamente dicha de estas capas, pero que ayudan al buen funcionamiento de la red, las cuales reciben el nombre de Reset y neurona de control de ganancia.

Para comprender mejor el funcionamiento de una red ART se ha dividido su funcionamiento en dos subsistemas, el de atención y el de orientación.

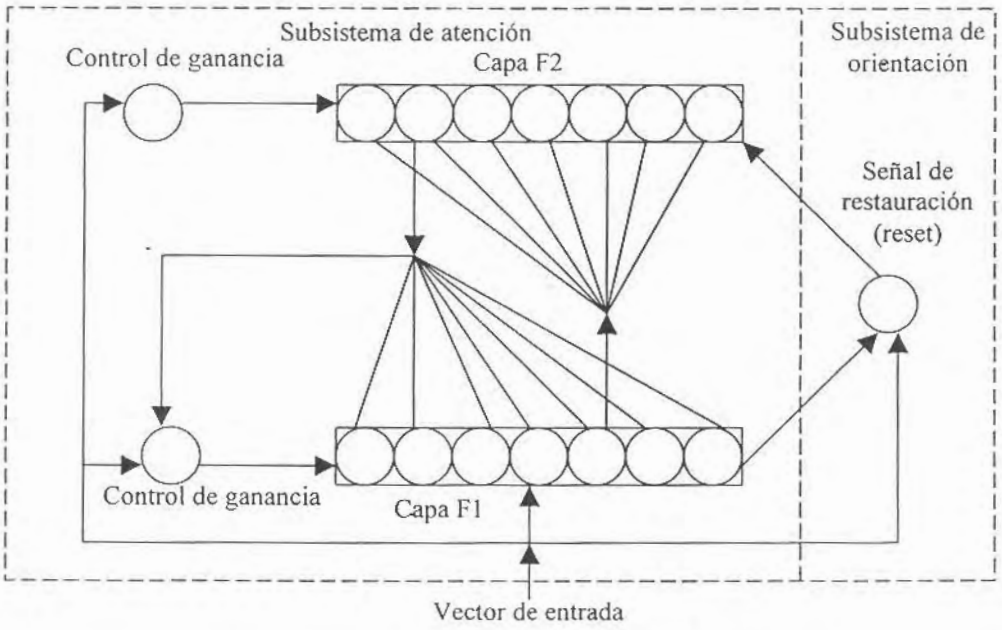


Figura 3.2.1 [FREEMAN 1993]

3.2.2 Subsistema de atención.

El subsistema de atención abarca las dos capas de neuronas, las conecciones que existen entre estas dos capas y una neurona que se conoce como control de ganancia. Su función es procesar las entradas de la red para ver si resuenan con algún patrón almacenado en la misma. Dentro de él se encuentran dos tipos de memoria, la memoria a largo plazo y la memoria a corto

plazo. La primera está dada por los pesos de las conecciones entre las capas, y se llama así por que se conserva aún cuando la red no se encuentre en funcionamiento. El otro tipo de memoria, se da en los valores de actividad que se presentan en las dos capas de neuronas (entrada y salida), y se llama así debido a que solo aparecen cuando la red esta en funcionamiento, y desaparece cuando no existe información a la entrada de la red.

3.2.3 El subsistema de orientación

El subsistema de orientación se encarga de decidir a cual clase pertenece la información presentada a la entrada de la red. Para esto utiliza el parámetro de vigilancia, que es el grado de parecido mínimo que debe haber entre el patrón de entrada y un patrón almacenado en la red, para que se puedan considerar de la misma clase. Este subsistema esta compuesto por una neurona auxiliar llamada reset.

Tanto la neurona de control de ganancia y la de reset se conectan a las neuronas de las capas de entrada y salida. Por lo tanto se obtiene que cada neurona de las capas F1 y F2 reciben al menos 3 entradas de diferentes lugares, por lo que para activarse utilizan la regla dos de tres. Dicha regla indica que la neurona será activada únicamente cuando dos de sus entradas sean diferentes de cero.

3.3 Red ART1

Como se mencionó antes, la red ART1, es la primera red desarrollada usando la teoría de resonancia adaptativa y únicamente acepta valores binarios como entrada, a continuación se explica su funcionamiento siguiendo el paso de un conjunto de entradas por toda la red.

3.3.1 Funcionamiento:

- Se presenta una entrada a la red, donde cada elemento de la entrada es encauzado a solo una de las neuronas de entradas de la red (N). Por tanto el número de elementos de la entrada debe ser igual al número de neuronas de entrada de la red.
- Cada neurona de entrada transmite el valor recibido a cada una de las neuronas de la capa de salida. Por tanto, cada neurona de la capa F2 recibe N entradas provenientes de la capa F1.

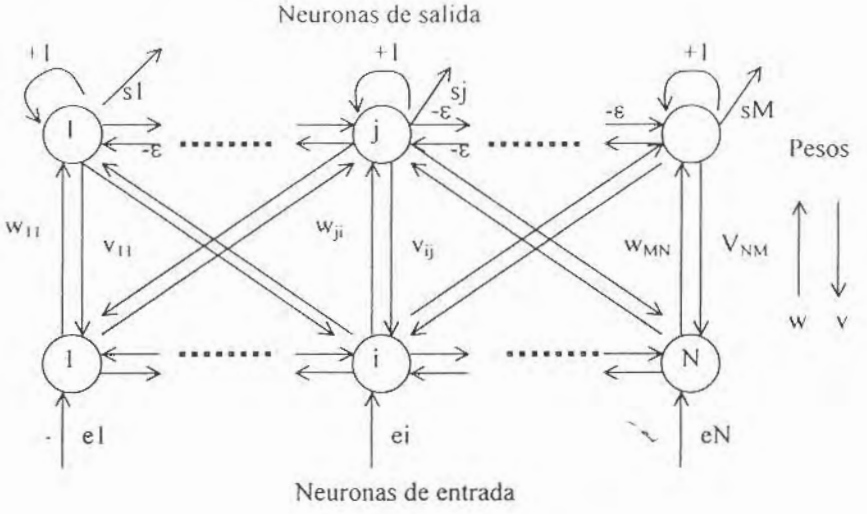


Figura 3.3.1.[HILERA 1995]
Esquema simplificado de las conecciones entre la capa F1 y F2 de la red ART2

- Las neuronas de la capa F2 entran en un proceso de competición por medio de las conecciones laterales entre ellas. Estas conecciones tiene un peso negativo fijo menor a 1/M (- ε) y como todos los pesos son iguales, la neurona ganadora será aquella para la cual la suma de sus entradas provenientes de F1 sea más grande.
- La neurona ganadora de la capa de salida o F2 es la única que tendrá una salida activada ó 1, mientras que las demás permanecerán en reposo o con salida 0. Las neuronas de la capa F2 envían su salida a las neuronas de la capa F1 por medio de las conecciones hacia atrás. Cada una de las neuronas de la capa de entrada reciben M señales de la capa F2, pero como en la capa de salida solo una neurona tiene su salida activada, solo una de estas señales tiene valor diferente de cero.
- Cada señal enviada de una capa a otra es ponderada por el peso asignado a la conección existente entre las neuronas que se envían la señal.
- Se procede a comparar los valores de entrada de la red, contra los valores recibidos en la capa F1 a través de las conecciones provenientes de la capa de salida. Para esto se utiliza la relación entre los módulos de estos valores. Como en este caso se trabaja con valores binarios 0 y 1, el módulo se calcula mediante la suma de todos los componentes de cada capa.
- Si el resultado de la relación anterior es mayor que el parámetro de vigilancia, se dice que la entrada pertenece a la clase asociada con la neurona ganadora de la capa F2. El parámetro de

vigilancia se da en un rango de 0 a 1, donde 1 representa una coincidencia exacta de todos los componentes, mientras 0 significa una desigualdad total. Por tanto entre más cercano este el parámetro a 1, la red tolerará una menor variación entre el patrón de entrada y el patrón guardado como prototipo.

- Si la relación calculada es menor que el patrón de vigilancia, significa que el patrón de entrada no pertenece a la clase representada por la neurona ganadora de la capa F2. Por lo cual se marca esta neurona como no representativa de la clase del patrón de entrada. Para lograr esto la neurona de reset, envía una señal inhibitoria a la neurona activada de F2 (neurona ganadora) para que no intervenga en una nueva competencia.
- Se hace una revisión entre las neuronas de la capa F2, para ver si todavía existe alguna neurona que no haya sido marcada como no apta. Si es así, se repite el proceso desde el paso de los datos del patrón de entrada a través de las neuronas de la capa F1. Cuidando de conservar las señales inhibitorias a las neuronas que ya han sido descartadas de la capa F2, para que no intervengan de nuevo en la competencia que se da en esta capa.
- Si todas las neuronas de la capa F2 ya fueron descartadas, la red da como salida una señal negativa al reconocimiento del patrón de entrada.
- Otra opción es que la red agregue una neurona más a la capa de salida, creé sus conecciones con la capa F1 y las neuronas de control de ganancia y reset. Y asigne a esta nueva neurona una nueva clase, donde el prototipo será el patrón de entrada.
- Una vez que se ha identificado la clase a la que pertenece el patrón de entrada se actualizan los pesos de las conecciones correspondientes a la neurona ganadora de la capa F2, por medio de una operación lógica AND entre los pesos guardados en las conecciones hacia atrás de la capa F2 y la capa F1, donde esta guardado el prototipo del patrón de la clase ganadora, y los valores de entrada de la red.

Esta red posee un aprendizaje no supervisado, ya que es ella misma la que determina si una información debe ser aprendida o descartada. A su vez este aprendizaje se puede clasificar en lento y rápido. El aprendizaje lento se da cuando el patrón de entrada pertenece a una clase ya definida dentro de la red, por lo tanto los pesos sólo sufren un ajuste de algunos de sus valores. Mientras que el aprendizaje rápido se lleva a cabo cuando el patrón de entrada no pertenece a ninguna clase definida en la red, y es necesario asociarle una neurona de la capa de salida. Por tanto los pesos correspondientes se modifican para representar el patrón de entrada como prototipo de la nueva clase.

El comportamiento anterior es la esencia del funcionamiento de la red ART. A continuación veremos las ecuaciones que representan los pasos anteriores. Para esto se ha optado por representar el conjunto de valores de entrada y los valores de activación o de salida de las capas de neuronas como vectores. Mientras que los pesos de las conecciones se representan por matrices, usando la letra w para los ascendentes y la letra v para los descendentes. Para ambos pesos se utilizan dos subíndices, el primero indica el índice de la neurona destino y el segundo él de la neurona origen.

Los pesos o factores de las conecciones hacia adelante y hacia atrás se inicializan de acuerdo a las siguientes ecuaciones:

$$w_{ji} = \frac{1}{1+N} \tag{3.3.1}$$

$$v_{ij} = 1 \tag{3.3.2}$$

El valor de 1 en los pesos v permite que para una nueva categoría, el prototipo a guardar sea igual al patrón de entrada, ya que la operación AND sólo será verdadera para todos los elementos del patrón de entrada igual a 1.

Para poder ver el funcionamiento de cada una de las capas que conforman la red es necesario explicar primero el funcionamiento de las neuronas de control de ganancia y de reset.

Empezaremos con la neurona de ganancia, esta neurona tiene el propósito de evitar que señales adelantadas de la capa F2 puedan llegar a la capa F1 antes de que llegue a ésta un patrón de entrada. Esto puede ocurrir si la red ART forma parte de una serie de redes neuronales. La neurona de ganancia recibe señales de las entradas de la red y de las neuronas de la capa F2. Utiliza un umbral de 0.5 como se puede ver en la ecuación (3.3.3). Su señal ayuda a que se cumpla la regla dos de tres para la activación de las neuronas de entrada. Si no hay un patrón a la entrada de la red, la suma de las señales, en la neurona de control de ganancia, siempre será negativa y su salida será cero. Esto deshabilitará las neuronas de entrada e impide que la red reaccione a una entrada adelantada de la capa F2.

$$S_G = \begin{cases} 1 & \sum_{i=1}^{N} e_i^{(k)} - N\sum_{j=1}^{M} s_{ns_j} \geq 0.5 \\ 0 & \sum_{i=1}^{N} e_i^{(k)} - N\sum_{j=1}^{M} s_{ns_j} < 0.5 \end{cases}$$

(3.3.3)

El nodo de reset recibe señales de las entradas de la red y de las salidas de las neuronas de la capa de entrada. Si la suma de estas señales es positiva genera una señal de reset para la neurona ganadora de la capa F2. La función de activación de esta neurona esta dada por:

$$S_R = \begin{cases} \text{"Re } set\text{"} ns_j & \sum \rho e_i^{(k)} - \sum s_{ne_i} > 0 \\ 0 & \sum \rho e_i^{(k)} - \sum s_{ne_i} \leq 0 \end{cases}$$

(3.3.4)

Donde ρ es el parámetro de vigilancia de la red, la función anterior también se puede definir como la relación entre el módulo de las salidas de las neuronas de entrada y el módulo de las entradas de la red.

$$\frac{\|s_{ne}\|}{\|e_i\|} < \rho$$

(3.3.5)

Si se cumple la condición anterior se resetea la neurona ganadora de F2.

Para las capas F1 y F2 las Funciones de activación son iguales a la función de salida o transferencia.

La capa de entradas recibe señales de entrada de tres diferentes fuentes: las señales del vector de entrada; las señales enviadas por la capa de salida a través de las conecciones hacia atrás y una señal proveniente de la neurona del control de ganancia, por lo que su entrada total es

$$net_{ne_i} = e_i^{(k)} + \sum_{j=1}^{M} v_{ij} s_{ns_j} + s_G$$

(3.3.6)

De acuerdo con la regla dos de tres, la salida de la neurona será igual al vector de entrada cuando ninguna salida de la capa F2 este activada. Debido a que entonces la neurona G esta activada. Por lo tanto la neurona de entrada recibe dos señales excitadoras, la que proviene del patrón de entrada y la que proviene del nodo de control de ganancia. La salida de esta capa será igual al resultado de una operación AND entre el vector de entrada y la sumatoria de las señales recibidas a través de las conecciones hacia atrás. Cuando existe una neurona con su salida activada en la capa F2, ya que entonces la salida de la neurona de ganancia es cero. Su salida está dada por:

$$S_{ne_i} = \begin{cases} e_i^{(k)} & s_{ns_j} = 0 \qquad \forall_j \\ e_i^{(k)} * \sum_{j=1}^{M} v_{ij} S_{ns_j} & OTRO \quad CASO \end{cases}$$

(3.3.7)

Por otro lado las neuronas de la capa de salida reciben las señales de la capa F1. Las salidas de las otras neuronas de la misma capa multiplicadas por el factor ε, y su propia salida. Como las señales de conecciones autorecurrentes y laterales tienen valores constantes no afectan en la competición, por lo que la única señal que se toma en cuenta en este caso es la suma de todas las señales recibidas de la capa F1, con lo que obtenemos la siguiente función.

$$S_{ns_j} = \begin{cases} 1 & MAX\left(\sum_{i=1}^{N} w_{ji} e_i^{(k)}\right) \\ 0 & resto \end{cases}$$

(3.3.8)

Donde MAX, representa el valor máximo obtenido del conjunto de datos.

3.3.2 Aprendizaje de la red ART1.

Una vez que se ha encontrado una neurona ganadora, se procede al ajuste de los pesos para que las características del patrón de entrada sean agregadas al prototipo de la clase guardada en la red. Los pesos de las conecciones descendentes se actualizan de acuerdo a la siguiente función:

$$v_{ij}(t+1) = v_{ij}(t) * e_i^{(k)}$$

(3.3.9)

Esto implica que el nuevo patrón prototipo de la clase será el resultado de una función AND entre el antiguo patrón y la entrada actual de la red.

Como los pesos w tiene el valor normalizado de los pesos v, su nuevo valor será:

$$w_{ji}(t+1) = \frac{v_{ij}(t)e_i^{(k)}}{\gamma + \sum_{i=1}^{N} v_{ij}(t)e_i^{(k)}}$$

(3.3.10)

Donde γ es un valor pequeño, por ejemplo 0.5, y se utiliza para evitar la división entre cero.

3.4 Red ART2.

La segunda red desarrollada con base a la teoría de la resonancia es la llamada redART2, cuya principal característica es que puede procesar valores continuos. Esto nos permite tener una mayor gama de posibles tipos de entradas.

Existen varias versiones de la red ART2, que varían en el número de capas de la red, así como en el flujo que tiene la información a través de la red. Pero todas están basadas en el principio de la resonancia adaptativa. En este trabajo se presentaran tres variaciones distintas de esta red, pero en la literatura actual sobre este tema podemos encontrar otras versiones.

La primera versión de la red se explicará a detalle, mientras que para las otras dos, sólo se explicara sus diferencias con respecto a la primera versión.

3.4.1 Primera versión de la red ART2.

Como es de esperarse, la red ART2 tiene un funcionamiento muy similar al de la red ART1, ya que ambas están basadas en los mismos principios, el de resonancia y competencia.

En esta versión, la capa F1 se ha subdividido en varias capas con el fin de poder normalizar los datos de la entrada antes de pasarlos a la capa F2. Así, el vector de entrada es procesado en forma cíclica por las subcapas de F1 hasta que sus valores se estabilicen. Luego, los valores son pasados a la capa F2, donde se lleva a cabo una competencia entre las neuronas de esta capa. Al igual que en ART1, la neurona ganadora será la única con señal activada. Esta señal se envía a las neuronas de la última subcapa de F1 a través de las conecciones hacia atrás.

En la red ART1 los pesos ascendentes w dependen de los pesos descendentes v y son diferentes. Mientras que en la ART2 los pesos w son iguales a los pesos v.

Una diferencia más se da en el subsistema de atención. Mientras que en la red ART1 éste está regulado por sólo una neurona de reset, en el modelo ART2 está representado por una capa de neuronas. Ésta capa recibe señales de dos subcapas de la capa F1. Una de estas capas contiene los valores del patrón ascendente, es decir, los valores del patrón de entrada normalizados, mientras que la otra subcapa contiene los valores del patrón descendente que es el prototipo guardado en la red, correspondiente a la neurona ganadora de F2. La capa de neuronas de reset genera una salida que es comparada con el parámetro de vigilancia de la red. Dependiendo del resultado, envía una señal de inhibición a las neuronas de la capa F2, cuando se ha determinado que el patrón de entrada no corresponde al prototipo asociado a la neurona de F2.

3.4.2 Ecuaciones de funcionamiento.

Una forma alternativa de representar el funcionamiento de las redes ART, es utilizando una ecuación general para las funciones de activación de las diferentes capas de la red. Esta ecuación toma diferentes parámetros dependiendo de la capa a la que se refiere. Es precisamente esta forma la que utilizaremos para explicar la funciones de las diferentes capas de la red ART2.

La ecuación general para las capas de la red es:

(3.4.1)

$$\varepsilon \chi_k' = -A\chi_k + (1 - B\chi_k)J_k^+ - (C + D\chi_k)J_k^-$$

Donde A, B, C y D son constantes que nos servirán para determinar el grado de influencia que tendrá cada señal en la neurona.

J_k^+ Es la señal excitadora que llega a la neurona.

J_k^- Es la señal inhibitoria que llega a la neurona

Para el modelo de red, presentado por Freeman y Skapura, se hace B y D igual a cero lo que nos da por resultado:

$$\varepsilon \chi_k' = -A\chi_k + J_k^+ - DJ_k^- \tag{3.4.2}$$

Como solo se toma en cuenta su solución asintótica que se logra cuando se hace cero el miembro izquierdo de la ecuación. Dando por resultado que la función de activación para las capas queda como:

$$\chi_k' = \frac{J_k^+}{A + DJ_k^-} \tag{3.4.3}$$

Esta ecuación se utilizara para todas las capas de la red, incluyendo las subcapas de la capa F1.

La capa F1 se subdivide en varias subcapas, las cuales están unidas entre si por medio de conecciones de un solo sentido. Manteniendo una relación uno a uno entre neuronas de dos subcapas diferentes, como se puede ver en la figura 3.4.1. Las conecciones entre las subcapas de F1 no tiene asignados un peso variante, mas bien se puede asumir que su peso es 1. Por tanto sólo sirven para el paso de la señal de una capa a otra, es la función de activación en cada capa la que va modificando el valor de las señales.

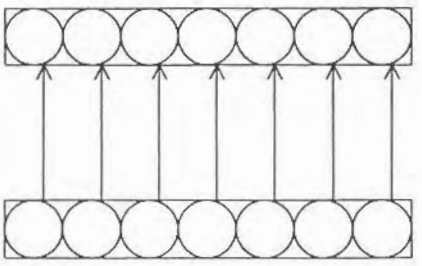


Figura 3.4.1
Relación de las conecciones, una a una, de una subcapa a otra dentro de la capa F1 de la red ART2.

La capa F1 esta compuesta por 6 subcapas, las cuales se designan utilizando las letras w, x, v, u, p y q. La capa p es la que genera las salidas que se transmitirán a la capa F2. Mientras que la capa u contiene los valores del patrón de entrada normalizados que servirán para hacer la verificación de coincidencia entre el patrón de entrada y los prototipos de patrones almacenados en la red.

El flujo de la información a través de las subcapas de F1 se lleva a cabo según el siguiente diagrama:

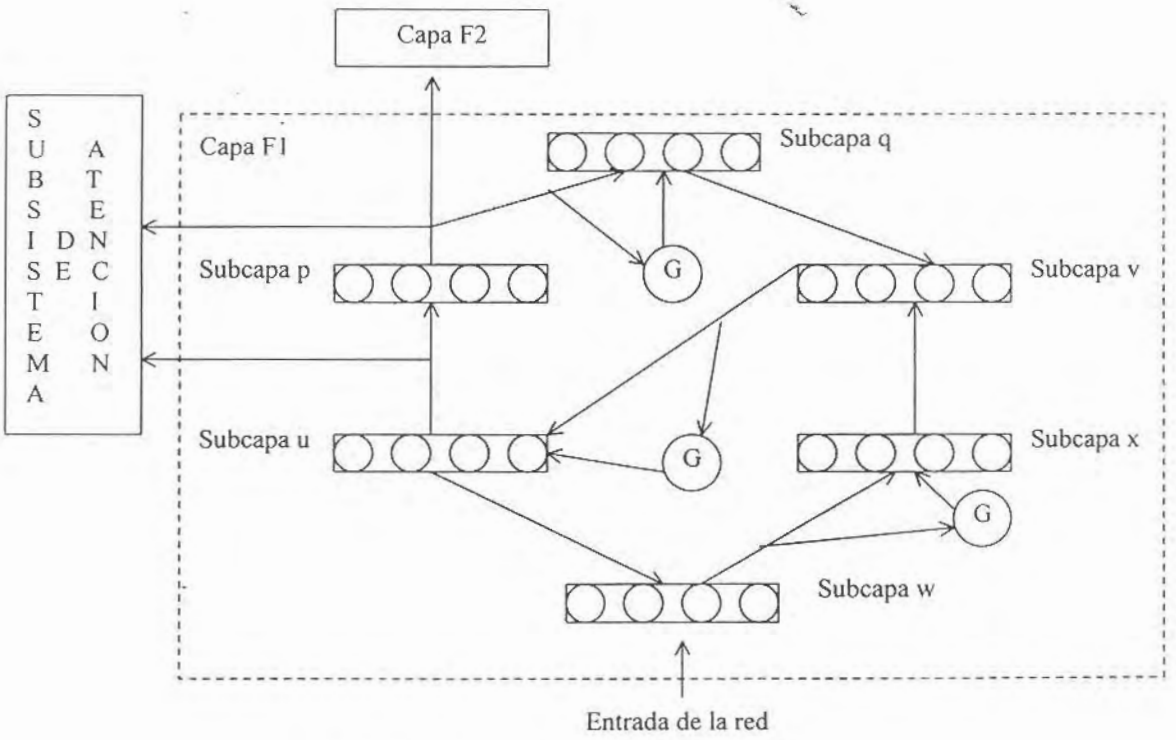


Diagrama de la capa F1 de la red ART2.
Figura 3.4.2[FREEMAN 1993]

En este diagrama los nodos o neuronas G (ganancia) son las encargadas de generar las señales inhibitorias para cada subcapa. Estas señales son iguales al modulo del vector de la capa de la cual provienen.

Para cada capa se definen la señal excitadora e inhibidora, así como los valores de las constantes A y D de acuerdo a la siguiente tabla:

CAPA	A	D	J_K^+	J_K^-
w	l	l	I_i+au_i	0
x	e	l	w_i	$\|\|w\|\|$
u	e	l	v_i	$\|\|v\|\|$
v	l	l	$f(x_i)+bf(q_i)$	0
p	l	l	$u_i+\sum_i g(y_i)z_{ij}$	0
q	e	l	p_i	$\|\|p\|\|$
r	e	l	u_i+cp_i	$\|\|u\|\|+\|\|cp\|\|$

Tabla de funciones excitadoras e inhibitorias para las subcapas de la capa F1 y la capa r de la red ART2.
Tabla 3.4.1[FREEMAN 1993]

El parámetro ε mostrado aquí, debe tener un valor positivo mucho menor que 1. Se utiliza para evitar la división por cero en las operaciones. Sin embargo se puede omitir si en la programación se crean mecanismos para evitar dicho problema. En nuestro caso se le asignará un valor de cero eliminándose de las ecuaciones[1]. Entonces las funciones de activación para cada una de las subcapas de F1 y de la capa r del subsistema de orientación quedan definidas como sigue:

$$w_i = I_i + au_i \tag{3.4.4}$$

$$x_i = \frac{w_i}{e + \|w\|} \tag{3.4.5}$$

$$v_i = f(x_i) + bf(q_i) \tag{3.4.6}$$

$$u_i = \frac{v_i}{e + \|v\|} \tag{3.4.7}$$

$$p_i = u_i + \sum g(y_j) z_{ij} \tag{3.4.8}$$

$$q_i = \frac{p_i}{e + \|p\|} \tag{3.4.9}$$

La función f(x) que se muestra en las funciones activación esta definida por

$$f(x) = \begin{cases} 0 & 0 \le x \le \theta \\ x & x > \theta \end{cases} \tag{3.4.10}$$

[1] En la programación se crearán los mecanismos necesarios para evitar la división por cero.

Donde θ es un valor positivo menor que 1. Esta función sirve como filtro contra el ruido de las señales de entrada.

Y g(x) es la función de salida de la capa F2. Toma el valor de *d* solo para la neurona cuya suma total de sus entradas sea la máxima, en esta competencia no entran aquellas neuronas que han sido desactivadas por el subsistema de orientación.

El funcionamiento de la capa F2 se realiza en forma similar a como se da en ART1. En este caso las señales que llegan a F2 tienen como origen la subcapa p de F1, por lo que la función de salida esta dada por:

$$g(y_i) = \begin{cases} d & T_j = \max_k \{T_k\} \quad \forall k \\ o & otro \end{cases} \tag{3.4.11}$$

donde

$$T_j = \sum_i p_i w_{ij} \tag{3.4.12}$$

3.4.3 Subsistema de orientación:

El subsistema de orientación esta compuesto por una subcapa de neuronas denominada r. Una vez que las señales han sido procesadas por esta capa se calcula su modulo y si la ecuación siguiente se cumple, se produce la restauración de la neurona ganadora de la capa F2.

$$\frac{\rho}{\|r\|} > 1 \tag{3.4.13}$$

3.4.4 Aprendizaje:

Los pesos v y w dentro de la arquitectura de capa ART2 tiene valores iguales, y se inician de acuerdo a la siguiente ecuación:

$$v_{ji} = w_{ij} \leq \frac{1}{(1-d)\sqrt{M}} \tag{3.4.14}$$

Una vez que se ha encontrado una resonancia entre el patrón de entrada y un prototipo almacenado en la red se procede al ajuste de los pesos, con base a la siguiente ecuación.

$$v_{Ji} = w_{iJ} = \frac{u_i}{1-d} \tag{3.4.15}$$

Antes de poner a funcionar la red ART2, es necesario determinar todas las constantes utilizadas en las diferentes funciones de la red. Para esto se han establecido las siguiente restricciones.

$$a, b > 0 \tag{3.4.16}$$

$$0 \leq d \leq 1 \tag{3.4.17}$$

$$\frac{cd}{1-d} \leq 1 \tag{3.4.18}$$

$$0 \leq \theta \leq 1 \tag{3.4.19}$$

$$0 \leq \rho \leq 1 \tag{3.4.20}$$

En resumen, se puede decir que un patrón entra a la red, y se procesa en la capa F1 hasta que se estabilicen los valores de F1. Cuando esto ocurre, se propagan los valores de u y p a la capa r, si es la primera vez que se realiza este proceso y aun no se tiene una neurona ganadora en la capa F2, se propagan los valores de p a F2; se encuentra la neurona ganadora y se envían las señales a través de las conecciones hacia atrás a la capa F1. Esto significa que se alteran los datos que entran a la subcapa p, por lo que es necesario esperar otra vez hasta que se estabilicen los pesos en esta capa. Cuando esto sucede se vuelven a propagar de la subcapa p al subsistema de orientación, es decir, a la capa r. Se hace la comparación de la relación entre el parámetro de vigilancia y el modulo de las salidas de esta capa y si la relación es mayor a 1 se marca la neurona para que ya no participe en futuras búsquedas y se repite todo el proceso. En caso de que la relación sea menor que 1 se ha encontrado la clase a la cual pertenece el patrón de entrada. Por lo que se procede a actualizar los pesos de las conecciones y a eliminar los valores de las capas F1 y F2 para el proceso del nuevo patrón.

3.4.5 Algoritmo:

En teoría, una red neuronal debe funcionar en forma paralela, es decir todas las neuronas de una capa funcionan al mismo tiempo. Recibiendo y enviando sus señales en el mismo instante. A su vez, varias capas pueden estar activadas en el mismo momento, dependiendo de sus valores de

entrada, pero en una simulación por computadora convencional, esto no es posible, por lo que se optó por procesar en forma secuencial la información. Primero se procesa toda la información correspondiente a una capa, y luego se continúa con otra capa, según el flujo de información que marque la estructura de la red. Por tanto es necesario definir un algoritmo que ponga de manifiesto la secuencia con la que se va a procesar la información.

El algoritmo de la red ART2 se puede expresar con los siguientes pasos[FREEMAN 1993]:

1. Se inicializan todas las capas de la red con cero, y un contador de ciclos con uno.
2. Se recibe el vector de entrada en la primera subcapa de F1 y se ejecuta su función de activación. Ecuación 3.4.4.
3. La salida de la capa w se envía a la capa x y se calcula su respectiva salida. Ecuación 3.4.5.
4. La salida de la capa x se envía a la capa v y se calcula su respectiva salida. Ecuación 3.4.6.
5. La salida de la capa v se envía a la capa u y también se calcula su respectiva salida. Ecuación 3.4.7.
6. Esta salida se envía a la capa p, la cual también recibe señales de la capa F2, si es que ya esta activada. Ecuación 3.4.8.
7. Para terminar el proceso en la capa F1, se envía la señal generada por p a la capa q y se calcula su salida, ecuación 3.4.9.
8. Se repiten los pasos del 2 al 7 hasta que los valores de las capas se estabilicen, es decir, no varíen. Por lo regular se pone un margen de variación muy pequeño para evitar que el proceso se alargue demasiado.
9. Se envían las señales de las capas u y p a la capa r, y se calcula su respectiva salida, de acuerdo a la siguiente ecuación.

$$r_i = \frac{u_i + cp_i}{e + \|u\| + \|cp\|} \qquad (3.4.21)$$

10. Dependiendo de este resultado se tiene las siguiente opciones:
    a) Si se cumple la condición de restauración (ecuación 3.4.13), se envía una señal de reset a la capa F2, marcando como no apta a la neurona ganadora de esa capa, y el contador de ciclos se vuelve a poner en 1.
    b) Si no se cumple la condición de restauración se tienen las siguientes opciones:

I. Si el contador de ciclos es igual a 1, se incrementa este contador y se pasa al paso 11, pues todavía no se han propagado las señales hacia la capa F2.

II. Si el contador de ciclos es mayor que 1, se pasa al paso 14; ya que existe una resonancia.

11. Se envía la señal de salida de la capa p a la capa F2. Donde cada señal es ponderada por los pesos w de las conecciones entre las dos capas y se calculan las entradas a las neuronas de esta capa. Ecuación 3.4.12.
12. Se encuentra la neurona ganadora de la capa F2 y se calcula la salida para todas las neuronas de esta capa. Ecuación 3.4.11.
13. Se repiten los pasos del 6 al 10.
14. Se actualizan los pesos ascendentes correspondientes a la neurona ganadora de la capa F2, ya que hemos obtenido una resonancia entre el patrón de entrada y el prototipo guardado en los pesos relacionados con la neurona ganadora de F2. Por lo que es necesario ahora incluir parte de la información del patrón de entrada en los pesos que representa el prototipo. Ecuación 3.4.15.
15. También se actualizan los pesos descendentes correspondientes a la neurona ganadora de la capa F2. Ecuación 3.4.15.
16. Se han terminado los proceso tanto de clasificación como de aprendizaje de la red. Así que se procede a eliminar el patrón de entrada y a quitar todas las señales de inhibición de la capa F2. Con esto la red queda preparada para recibir un nuevo patrón de entrada y empezar con el paso 1 de nuevo.

3.5 2ª. Versión de la red ART2

Se presenta ahora una segunda versión de la red ART2. Esta versión está basada en varios documentos publicados en Internet, [GAUDIANO 1989, LEHAR 1989]. Todos ellos parten de artículos de Carpenter y Grossberg.

Las principales características en que difiere esta versión de la red ART2 con la presentada anteriormente, es la creación de una capa denominada F0, la selección de las capas que envían su información al sistema de orientación o capa de reset y el orden en que se procesa la información dentro de la subcapa F1.

3.5.1 La capa F0

En la versión anterior, primero la red cae en un ciclo que procesa únicamente las subcapas de F1(pasos del 1 al 8 del algoritmo), para lograr un equilibrio en los valores de entrada. Como hasta ese momento no se ha activado la capa F2, las señales que provienen de ella no tienen efecto en la capa F1. Esto se hace con el fin de estabilizar y normalizar las entradas a la red y evitar que el ruido de las señales de entrada intervengan en el proceso de clasificación. Una vez estabilizada la capa F1, la información se propaga a la capa F2. Ésta, manda su señal de regreso a la capa F1, la cual vuelve a caer en un ciclo para volver ha estabilizar las actividades en sus subcapas. Pero como sigue recibiendo las entradas originales de la red. Se tiene que volver a eliminar el ruido proveniente de la entrada de la red. Para evitar este problema, se propone hacer una copia de la capa F1, llamada capa F0 que se encargue de la normalización y filtrado de las señales de entrada de la red. Esta capa será la que le pase la información a la capa F1, que estará conectada con la capa F2. Así se evitará que cuando regresen los valores de la capa F2 a la F1, esta última capa tenga que volver ha eliminar el ruido proveniente de la entrada de la red.

La capa F0 queda compuesta entonces por el mismo número de subcapas que la capa F1. Estas subcapas reciben el mismo nombre, sólo que se les agrega un subíndice 0 para diferenciarlas de las subcapas de F1. Sus ecuaciones están dadas por [GAUDIANO 1989]:

$$w0_i = I_i + au\,0_i \tag{3.5.1}$$

$$p0_i = u0_i \tag{3.5.2}$$

$$q0_i = \frac{p0_i}{e + \|p0\|} \tag{3.5.3}$$

$$x0_i = \frac{w0_i}{e + \|w0\|} \tag{3.5.4}$$

$$v0_i = f(x0_i) + bf\,(q0_i) \tag{3.5.5}$$

$$u0_i = \frac{v0_i}{e + \|v0\|} \tag{3.5.6}$$

La única ecuación que sufre un cambio, es la ecuación 3.5.2, referente a la subcapa p0. Debido a que la capa F0 no esta conectada con la capa F2 ya no se reciben señales de esta capa y la ecuación se simplifica. Convirtiendo a la capa p una copia de la capa u. Por tanto se podría eliminar

una subcapa, pero se prefiere dejar las ecuaciones lo más parecidas a las anteriores para evitar confusiones.

3.5.2 La capa F1

Las ecuaciones en esta capa se conservan iguales a excepción la capa w, ya que ahora no recibe las señales directas de la entrada de la red, en su lugar recibe la salida de la subcapa q0 de la capa F0, quedando la ecuación para esta subcapa como: [GAUDIANO 1989]:

$$w_i = q0_i + au_i \tag{3.5.7}$$

3.5.3 Capa reset

En esta versión las capas que envían su salida al nodo reset son la subcapa q0 de la capa F0 y la subcapa p de la capa F1. Lo que nos da la siguiente función de activación para esta capa [GAUDIANO 1989]:

$$r_i = \frac{q0_i + cp_i}{\|q0\| + c\|p\|} \tag{3.5.8}$$

Esto es lógico, ya que la capa q0 representan el patrón a la entrada de la red. Mientras que la capa p representa el prototipo guardado en las conecciones entre las capas F1 y F2. La relación entre ellas nos dará la semejanza entre los dos patrones.

3.5.4 Algoritmo de la red ART2

La última diferencia entre las dos versiones es la secuencia que sigue la información a través de la red. Esto se refleja en el algoritmo de la misma, el cual se presenta con todas las modificaciones que se requieren.

1. Se inicializan todas las capas y subcapas de la red a cero, y un contador de ciclos a uno.
2. Propagación de la capa F0:
   a) Primero se hace una copia de la capa w0
   b) Se propaga la información a través de las subcapas w0 y p0. Ecuaciones 3.5.1 y 3.5.2
   c) Se calcula los módulos de las subcapas w0 y p0

d) Se propaga la información por las capas q0, x0 y v0. Ecuaciones 3.5.3, 3.5.4 y 3.5.5

e) Se calcula el modulo de la subcapa v0.

f) Se obtiene los valores de la capa u0. Ecuación 3.5.6

g) Se incrementa el contador de ciclos.

h) Se repiten desde el paso 2a, hasta que la diferencia entre la copia de w0 y su valor actual sea muy pequeña o bien el contador de ciclos sea mayor que 100.

3. Se le asigna un valor de cero al contador de ciclos utilizado en el primer paso.
4. Propagación de la capa F1.
   a) Se hace una copia de las subcapas w y p
   b) Se propaga la información por la capa w. Ecuación 3.5.7.
   c) Se propaga la información a través de la subcapa p. Ecuación 3.4.8.
   d) Se calculan los módulos de las subcapas p y w
   e) Se propaga la información por las capas q, x y v. Ecuaciones 3.4.9,3.4.5 y 3.4.6.
   f) Se calcula el módulo de la capa v.
   g) Se calcula la salida de la capa u. Ecuación 3.4.7.
   h) Se obtiene las diferencias entre las copias de w y p y sus valores actuales, respectivamente. Se suman los dos resultados finales.
5. Si la capa F2 no tiene ninguna neurona marcada como ganadora, se envían las salidas de la capa p a F2. Ecuación 3.4.12. Se calcula su salida. Ecuación 3.4.11
6. Se actualiza la salida de la capa de reset. Ecuación 3.5.8,
7. Dependiendo del resultado de la condición de restauración. Ecuación 3.4.13, tenemos:
   a) Si se cumple la condición, se resetea la neurona ganadora la capa F2, y el contador de ciclos se vuelve cero.
   b) Si no se cumple, se hace la actualización de pesos correspondientes a las conecciones de la neurona ganadora de la capa F2, para esta actualización se siguen los siguientes pasos:
      I. Se hace una copia de los pesos anteriores.
      II. Se actualizan los pesos. Ecuación 3.4.15
      III. Se saca la diferencia entre las copias de los pesos y los pesos actuales. Se suman los resultados.
8. Se incrementa el contador de ciclos.
9. Se repiten los pasos desde el 4 al 8 hasta que las diferencias en las capas (paso 4h) y las diferencias en los pesos, paso 7-b-III sean muy pequeñas, o bien, el contador de ciclos sea mayor que 100.

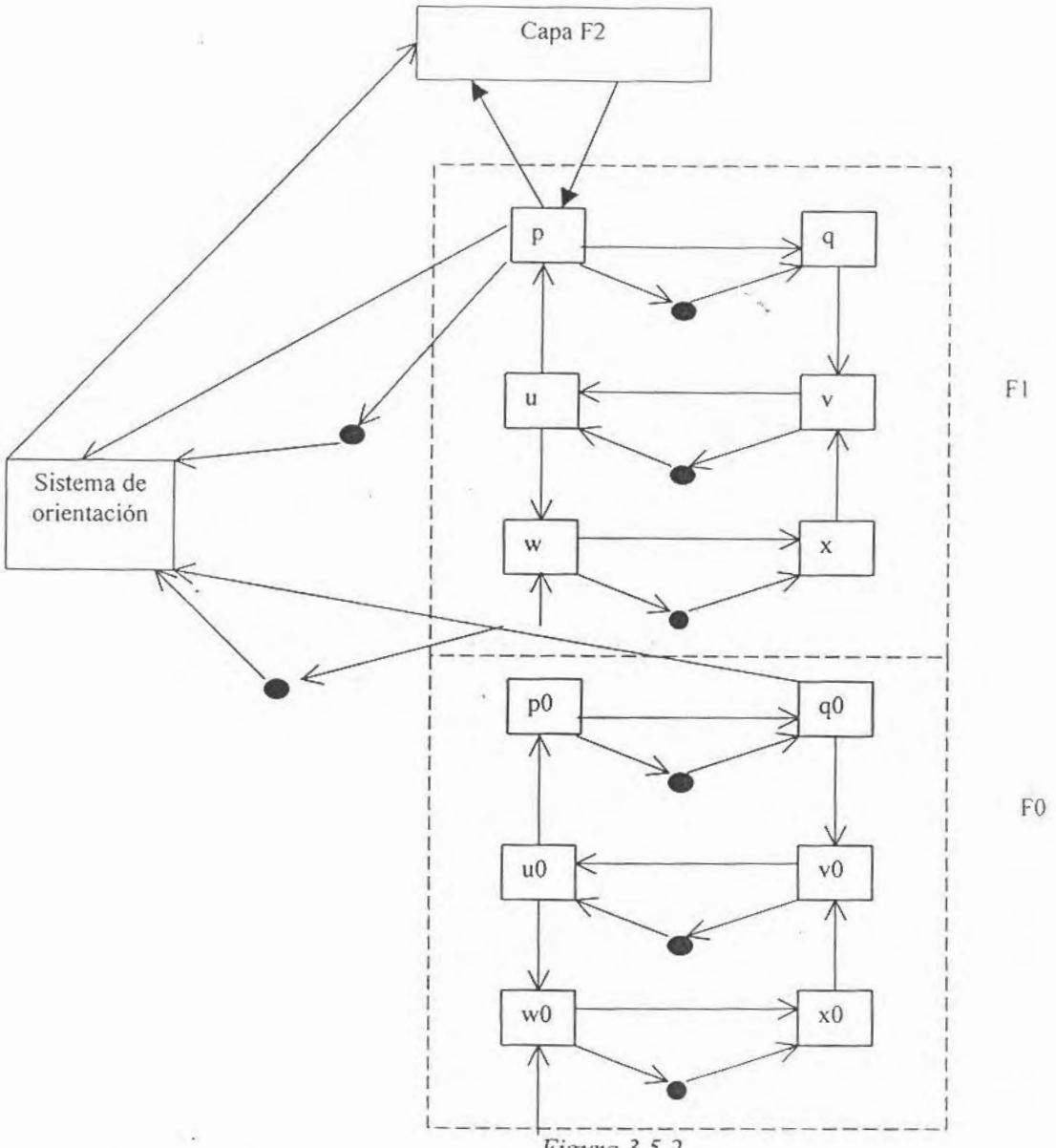


Figura 3.5.2

Diagrama de flujo de la información por las subcapas de las capas F0 y F1 de la 2da. Versión de la red ART2.

Como se puede apreciar en el algoritmo anterior, hay dos ciclos principales. Uno que estabiliza la capa F0 y con ello la entrada de la red. El otro estabiliza a las capas F1, F2, reset y los

pesos, que es en sí, el funcionamiento real de la red. Con estas mejoras nos aseguramos que la red logre un equilibrio en sus valores.

La topología de la red ART2 de esta versión es la misma que la topología de la versión anterior, en cuanto a la capa F1 y F2. La capa F0 tiene la misma topología que la capa F1. Con la única diferencia que las neuronas de la subcapa q0 están conectadas uno a uno con las neuronas de la subcapa w de la capa F1.

Esta topología puede verse como la figura 3.5.2. Sólo tome en cuenta que las flechas indican el flujo de la información y las capas están representadas por cajas cerradas.

3.6 3ª. Versión de la red ART2.

En esta sección veremos la última versión de la red ART2 utilizada en este trabajo. Al igual que con la segunda versión sólo se estudiará a detalle las diferencias que existen entre esta versión de la red y las anteriores.

Esta versión es presentada por Fausett [FAUSEETT 1994]. Sin embargo, al igual que en los casos anteriores está basada en artículos de Carpenter y Grossberg. De hecho su diferencia reside en el flujo que sigue la información por la red, en tanto que la topología como las ecuaciones de cada capa se conservan iguales a la utilizadas en la primera versión. Por lo anterior, sólo nos abocaremos a explicar el algoritmo para esta versión.

3.6.1 Algoritmo.

1. Iniciación de todas la capas con valor cero, e inicialización de los parámetros.
2. Se actualizan los valores de la capa F1. Dando dos recorridos para asegurarse que todas las subcapas obtengan valores diferentes de cero. Ecuaciones de la 3.4.4 a la 3.4.9. De acuerdo al siguiente orden de las subcapas u, w, p, x, q, y v.
3. Se propaga la información de la subcapa p de F1 a la capa F2 y se calculan las entradas de ésta..
4. Se encuentra la neurona ganadora de la capa F2. Si ya no hay neuronas disponibles termina, y como salida muestra que no se puede clasificar el patrón de entrada.
5. Verificar la condición de restauración siguiendo estos pasos:
   a) Se actualizan los valores de las capas u, p y r. Ecuaciones 3.4.7, 3.4.8, 3.4.21.
   b) De acuerdo con el resultado de la condición de restauración. Ecuación 3.313. Tenemos:

I. Si se cumple la condición, hay restauración. Se envía la señal inhibitoria a la neurona ganadora de la capa F2 y se regresa al paso 4.

II. Si no se cumple la condición. Se encontró una resonancia y se actualizan las demás subcapas de la capa F1 en el siguiente orden w, x, q y v.

6. Se actualizan los pesos correspondientes a la neurona ganadora, de acuerdo a las siguiente ecuaciones:

$$w_{iJ} = \alpha d u_i + \{1 + \alpha d (d - 1)\} w_{iJ} \tag{3.6.1}$$

$$v_{Ji} = \alpha d u_i + \{1 + \alpha d (d - 1)\} v_{Ji} \tag{3.6.2}$$

7. Actualiza todas las subcapas de la capa F1, de acuerdo al siguiente orden u, w, p, x, q, y v,
8. Repite los pasos 6 y 7 hasta que los pesos se hayan estabilizado o se cumpla un número de ciclos.

Como se puede ver este algoritmo se parece mucho al de la primera versión. Su diferencia radica en la actualización de las capas u y p después de encontrar la neurona ganadora de la capa F2, y antes de mandar sus señales a la capa de reset para checar la condición de restauración. Esto nos asegurara que realmente se este haciendo una comparación entre el patrón guardado por la red, reflejado en la capa p, y el patrón de entrada, correspondiente a la capa u. Ya que estas capas se actualizan inmediatamente después de encontrar la neurona ganadora.

En la primera versión, la actualización de las subcapas de la capa F1 después de encontrar la neurona ganadora de F2, se daba en forma general, es decir, se actualizan todas las subcapas de la capa F1 (paso 13), y se espera un equilibrio en la red (paso 8). Por lo que los valores de p pueden llegar a afectar a los valores de u. Lo que ocasiona que a la neurona de reset no lleguen los valores correctos.

3.7 Notas generales

Se han presentado ya tres versiones de la red ART2 en las cuales la información fluye de distinta forma. Pero las tres respetan el principio de resonancia y utilizan las mismas variables. Es

importante remarcar algunas características importantes para el buen funcionamiento de la red y que afectan a las tres versiones de forma similar.

El valor de θ determinará la cantidad de información que se considerará como ruido. Por tanto, si nuestras entradas de diferentes clases son muy parecidas es recomendable poner este parámetro lo más pequeño posible o asignarle un valor igual a cero. Esto con el fin de que la red tome en cuenta el mayor número de datos de la entrada.

Las variables a y b afectan a los pesos dentro de la capa F1. Si su valor es cero, la red se vuelve inestable. Pero para valores diferentes de cero la red no es muy sensible a cambios en estas variables[FAUSEETT 1994].

La variable d es la que afecta directamente a la neurona ganadora. Por lo que se recomienda que tenga un valor lo más grande posible, siempre dentro del rango permitido [0,1]. Esto con el fin de que exista una mayor diferencia entre las salidas de la neurona ganadora y las demás neuronas de la capa F2.

La variable c influye en la prueba para determinar si una neurona se reseta o no. Si se le dan valores pequeños se obtendrá una mayor eficiencia en la comparación con el parametro de vigilancia. Por lo que se recomienda un valor de 0.1. Si se aumenta este valor disminuye la sensibilidad de la red. [FAUSEETT 1994].

Aparte de las redes ART1 y ART2, existe una tercer tipo de red basada en la teoría de la resonancia adaptativa, que es la red ART3. Dicha red tiene la misma topología de la red ART2, pero utiliza ecuaciones que representan el modelo de la dinámica de los neurotransmisores químicos. En este modelo Grossberg y Carpenter ponen un mayor énfasis en incluir la funcionalidad del modelo ART en una topología que sea representativa de la estructura neuronal biológica. Una de las ventajas de la red ART3 es que revisa constantemente las entradas de la red y si estas cambian significativamente, la red se resetea por sí misma y vuelve a empezar el proceso de reconocimiento.

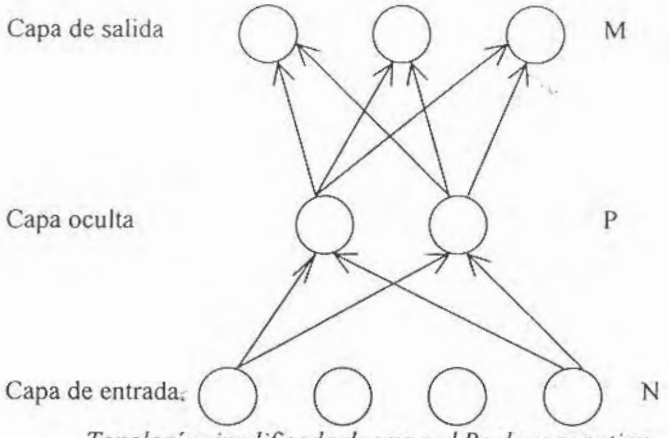
3.8 Red Backpropagation.

Toca el turno a la red backpropagation. Ésta, en realidad es una red multicapa de conecciones hacia adelante, la cual le debe su nombre al método utilizado para su aprendizaje. Este método consiste en la propagación hacia atrás del error producido por la red y se conoce como regla

delta generalizada. Durante el desarrollo de este trabajo se utilizará el nombre de red Backpropagation para esta red.

La red Backpropagation tiene la siguiente topología:

Esta compuesta por al menos tres capas, una capa de entrada, una capa de salida y al menos una capa de neuronas ocultas.



Topología simplificada de una red Backpropagation.

Figura 3.8.1

Sus conecciones son unidireccionales y cada neurona de entrada envía su señal a todas la neuronas de la capa oculta. Por lo que tienen una conección de una a muchas. A su vez cada neurona de la capa oculta envía sus señales a todas las neuronas de la capa de salida, estableciéndose también una conección de una a muchas.

El método de aprendizaje Backpropagation o regla delta generalizada consiste en un método de gradiente descendente para minimizar el error cuadrado de la red, más adelante se explicará a detalle en que consiste esto. Su naturaleza general permite que sea usado para resolver diversos problemas. Por lo que existen muchas aplicaciones que utilizan este método. Debido a que se puede utilizar para resolver cualquier problema que involucren relacionar un conjunto de datos de entrada con un conjunto de datos de salida. Esta red es de aprendizaje OFFLINE, por lo que existe una diferencia marcada entre la etapa de entrenamiento y la de funcionamiento.

El objetivo del entrenamiento de la red consiste en lograr que la red produzca las respuestas correctas a la entrada de los patrones utilizados durante su aprendizaje, memorización. Y que también produzca respuestas coherentes a entradas similares pero no iguales a las usadas durante el entrenamiento, generalización.

La fase de entrenamiento de una red considera 3 etapas:

- La propagación del patrón de entrada a través de toda la red hacia adelante.
- El calculo del error generado entre la salida obtenida y la deseada en la capa de salida y su propagación hacia atrás en la red. Es decir, de la salida a la entrada.
- El ajuste de los pesos basándose en los errores calculados en el paso anterior.

Mientras que la fase de funcionamiento de una red backpropagation , incluye únicamente al primero de los 3 pasos anteriores. Es decir, la propagación hacia adelante de los patrones de entrada, y su salida será el resultado final de la red.

La fase de entrenamiento es la más lenta puesto que se tienen que procesar todos lo patrones muestra. Mientras que la fase de funcionamiento es muy rápida, ya que solo se procesa un solo patrón. Por lo anterior se han desarrollado diversas variantes del algoritmo de aprendizaje para aumentar su velocidad de aprendizaje, como es el uso de una taza de aprendizaje o de momentos.

Una red con una sola capa puede resolver problemas muy sencillos con poca separabilidad, mientras que las redes multicapa tienen más capacidad de resolver problemas con alto grado de separabilidad. Sin embargo, con base a experimentos, se ha determinado que aunque más de una capa oculta puede ayudar a encontrar más rápido la solución de un problema en ciertas aplicaciones, por lo general, la mayoría de problemas se resuelven con solo una capa de neuronas ocultas. Con variar el número de neuronas de esta única capa se puede mejorar la rapidez de aprendizaje de la red[FAUSETT 1994].

Aunque solo se muestran las conecciones hacia adelante que se utilizan durante la primera etapa del entrenamiento. Para algunos autores existen conecciones con la dirección inversa que son las encargados de hacer la propagación del error hacia atrás durante la fase del entrenamiento. Otros asumen que la propagación hacia atrás se da por las mismas conecciones hacia delante.

3.8.1 Algoritmo:

Se empezará por estudiar el algoritmo para el aprendizaje, desglosando cada una de sus tres etapas.

- Primera etapa, cada neurona de entrada recibe un elemento del patrón de entrada y lo propaga a la capa siguiente, que es la capa oculta, en forma directa sin utilizar ninguna señal de activación. Las señales enviadas a cada una de las neuronas de la siguiente capa serán ponderadas por los pesos vij asociados a las conecciones entre las neuronas de entrada y ocultas.
- Cada neurona oculta suma todas las señales recibidas y utilizando su función de activación genera una señal de salida. Esta señal es enviada a todas las neuronas de la capa siguiente o sea la de salida y al igual que en el caso anterior son ponderadas por pesos. En este caso los pesos w_{ij}, asociados a las conecciones entre las neuronas de la capa oculta y la capa de salida.
- Ahora bien, cada una de las neuronas de la capa de salida suma todas sus entradas incluyendo el termino de corrección, le aplica su función de activación, generando así una salida que es la salida de la red al patrón de entrada.
- La segunda etapa, que es el cálculo y propagación del error, empieza en este punto. Donde cada neurona de la capa de salida compara la salida obtenida en la etapa anterior con la salida correspondiente al patrón de salida asociado al patrón de entrada. Obteniendo así un error, el cual se utilizará para calcular el factor dk, término de corrección. El cual es usado tanto para la propagación del error hacia las capas anteriores como en la actualización de los pesos asociados a las conecciones entre neuronas.
- Una vez calculados los dk para las neuronas de salida, estos se envían a las neuronas de la capa oculta, donde se calculan sus respectivos dj, términos de corrección.
- En la tercera etapa se actualizan los pesos de las conecciones de la red. Para los pesos w, que van de la capa oculta a la capa de salida, se utilizan los dk calculados en la capa de salida, así como las entradas a las neuronas de esta capa, zj. Mientras que para los pesos v que van de las neuronas de la capa de entrada a las neuronas de la capa oculta se utilizan los dj obtenidos para las neuronas ocultas y sus respectivas entradas xi.

3.8.2 Funciones de activación.

Como se ha mencionado, tanto la capa de neuronas ocultas como la capa de salida, utilizan una función de activación para generar su salida. Esta función deber ser continua, diferenciable y monoticamente no decreciente para poder aplicar el concepto de gradiente descendente.

Es preferible también que la derivada de la función se pueda expresar en términos de la función original. Esto nos facilitará mucho la implementación del algoritmo, las funciones más usadas para estos casos son:

$$f_1(x) = \frac{1}{1+e^{-x}} \qquad 3.8.1$$

$$f_1'(x) = f_1(x)[1 - f_1(x)] \qquad 3.8.2$$

conocida como función sigmoide binaria, que tiene un rango de salida (0,1)

Otra función es

$$f_2(x) = \frac{2}{1+e^{-x}} - 1 \qquad 3.8.2$$

$$f_2'(x) = \frac{1}{2}[1 + f_2(x)][1 - f_2(x)] \qquad 3.8.4$$

que es conocida como la función sigmoide bipolar y cuyo rango de salida es de [-1, 1].

La elección de la función a utilizar dependerá del problema a tratar.

3.8.3 Definiciones.

Los elementos de la red y variables que utiliza.

- El patrón de entrada esta dado por el vector.

$$X=(x_1,x_2,...,x_n)$$

3.8.5

- El patrón de salida deseada esta dado por el vector.

$$T=(t_1,t_2,...t_m)$$

3.8.6

- w_{jk} son los pesos de las conecciones entre las neuronas de la capa oculta y la capa de salida.
- v_{ij} son los pesos de las conecciones de las neuronas de la capa de entrada y la capa oculta.
- δi representa la diferencia de ajuste para los pesos w.
- δj representa la diferencia de ajuste para los pesos v.
- xi representan tanto las señales de entrada y de salida de la capa de neuronas de entrada, ya que en esta capa su salida es igual a su entrada.
- v_{0j} es el término de corrección para las neuronas ocultas.
- w_{0k} es el término de corrección para las neuronas de la capa de salida.
- La entrada en las neuronas de la capa oculta se define como

$$y_{ink} = w_{0k} + \sum_j z_j w_{jk}$$

3.8.7

- Y su salida esta dada por:

$$y_k = f(y_{ink})$$

3.8.8

- La entrada de las neuronas de salida esta dada por

$$z_{inj} = v_{0j} + \sum_i x_i v_{ij}$$

3.8.9

- Y su salida se define como

$$z_j = f(z_{inj})$$

3.8.10

- El cálculo de errores en la capa de salida

$$\delta_k = (t_k - y_k) f'(y_{ink})$$

3.8.11

- Y el ajuste para cada peso asociado a las conecciones de esta capa esta dado por

$$\Delta w_{jk} = \alpha \delta_k z_j$$

.8.12

Donde α es la taza de aprendizaje y determina que tan rápido o lento será el aprendizaje de la red.

- El ajuste para cada termino de corrección correspondiente a cada neurona de la capa de salida esta dado por:

$$\Delta w_{0k} = \alpha \delta_k$$

3.8.13

- Cada δ_k generado por cada neurona de salida es enviado a cada una de las neuronas de la capa anterior, o sea la capa oculta, por lo que cada neurona de esta última capa recibe m términos δ_k, y el error recibido por las neuronas de la capa oculta esta dado por:

$$\delta_{inj} = \sum_{k=1}^{m} \delta_k w_{jk}$$

3.8.14

- El error en cada neurona de la capa oculta se determina con base al resultado de la ecuación 3.8.14, y esta dado por:

$$\delta_j = \delta_{inj} f'(z_{inj})$$

3.8.15

- El ajuste de los pesos v esta dado por

$$\Delta v_{ij} = \alpha \delta_j x_i$$

3.8.16

- Por último la variación para el término de corrección de cada neurona oculta se calcula de la siguiente forma

$$\Delta v_{0j} = \alpha \delta_j$$

3.8.17

Con esta descripción de componentes se finaliza la explicación de todas las topologias y algoritmos usados para las diferentes redes neuronales. Para la red Backpropagation no se usará ningún mecanismo para acelerar su convergencia. Por tanto la versión presentada aquí es de las más sencillas que puede haber de este tipo de red.

Capítulo 4: Implementación del sistema

En este capítulo veremos las formas en como se implementaron cada una de las redes antes expuestas, algunos de los problemas encontrados en este proceso y las variables que se necesitan para su buen funcionamiento. Al igual que se expuso en la teoría, se verá cada red por separado, primero se analizara la red ART1, las tres versiones de la red ART2 y por último la red Backpropagation.

4.1 Esquema general.

Para realizar la simulación de las redes se requiere de un sistema de computo además de un lenguaje de programación. Ya que está enfocado al reconocimiento de patrones en imágenes digitales es necesario elegir un medio para la adquisición y desplegado de las imágenes, así como el formato que éstas tendrán.

Empezaremos con el lenguaje de programación. La implementación de las redes neuronales requiere del manejo de una gran cantidad de memoria, así como de una interfaz gráfica para el despliegue de las imágenes. Por tanto, se decidió utilizar el lenguaje de programación Visual C++, ya que permite el manejo de memoria de acuerdo a nuestras necesidades y brinda una interfaz completamente gráfica para el manejo de nuestras imágenes. Puesto que Visual C++ trabaja sobre la plataforma de Windows se necesita un equipo que lo soporte. Se utilizó uno con las siguientes características: CPU MMX a 500 Mhz, 32 Mbytes de memoria RAM, disco duro de 4 Gbyte, monitor SVGA. Y los siguientes programas: Windows 95 y Visual C++ V. 6.

Para poder realizar la comparación en las condiciones mas similares, ambas redes se implementaron dentro del mismo sistema y se permite su funcionamiento en forma alternada entre el proceso de una entrada y otra. Primero veremos una descripción general del sistema completo y después nos enfocaremos a la implementación a detalle de las redes neuronales.

El sistema tiene una ventana principal, donde se despliegan las imágenes y un menú principal que da acceso a todas las funciones con que cuenta. Las opciones del menú principal son: archivo, red ART1, red ART2, red Backpropagation, utilerías, filtros y Ayuda.

La primera opción nos permite abrir los archivos de las imágenes. La segunda, tercera y cuarta nos dan las opciones para poder ejecutar las correspondientes redes. La quinta ofrece una serie de métodos de procesamiento digital de imágenes para realizar el preprocesado de la imagen y la última nos ofrece información sobre el sistema. Para una explicación más detallada de la operación del sistema consulte el manual de usuario en el apéndice A. Por el momento nos abocaremos en la implementación de las redes neuronales artificiales simuladas. Todas las redes se

implementaron como clases. Se crearon tres clases principales ART1, ART2 y la Back. Una para cada tipo de red. Para accesar a una red se declara un objeto de su clase correspondiente.

4.2 *Implementación de la ART1.*

Para implementar esta red se usaron punteros para el manejo de memoria dinámica, ya que no se sabe con anterioridad la cantidad exacta de memoria que se va ha utilizar. Después se enumeran y describen cada una de las funciones que se definieron para esta clase y por último se ve en forma general como se enlazan todas estas funciones, es decir, el funcionamiento real de la red.

4.2.1 Variables.

Se utilizaron diferentes punteros para controlar la memoria de los siguientes datos: las entradas de la red, las salidas de las neuronas de entrada de la red, entrada y salida para las neuronas de salida de la red, pesos w, v, nodos de ganancia y de reset. También se ocuparon variables para los siguientes datos: Neurona ganadora, número de entradas, número de salidas, parámetro ρ y parámetro comp.

Esto debido a que se permite al usuario determinar el número de entradas de la red, pero el número de salidas solo puede estar entre 1 y 10. Esto con el fin de conservar este número como límite para todas las redes y los resultados sean fáciles de desplegar.

Los números de entrada y salida guardan la dimensión de las dos capas de la red, la variable neurona ganadora guarda el índice de la neurona vencedora de la capa de salida, la variable ρ guarda el parámetro de vigilancia y por último la variable comp guarda la semejanza obtenida entre el patrón de entrada y el prototipo guardado en la red.

4.2.2 Funciones.

Dentro de la clase se definieron las siguientes funciones:

- *Asigna_memoria:* Asignar la memoria dinámica de acuerdo al tamaño elegido para la red.
- *Libera_memoria:* Libera toda la memoria asignada dinámicamente.
- *Asigna_tam_capas:* Asigna el tamaño de la red a los número de entrada y salida. *Asigna_parametros*: Recibe y guarda el parámetro de vigilancia a utilizar en la red.
- *Iniciar_pesos:* Incializa los pesos ascendentes y descendentes.

- *Guardar_pesos:* Guarda los pesos actuales en archivos ".dat", para poder recuperarlos después. Los pesos ascendentes w se guardan en el archivo a1pesosw.dat y los pesos descendentes v en el archivo a1pesosv.dat.
- *Lee_pesos.* Lee los pesos guardados en los archivos ".dat" y los asigna a sus respectivos arreglos.
- *Procesar_imagen*: Toma como entrada la imagen que este desplegada en ese momento en la pantalla y procesa la información a través de la red.

4.2.3 Funcionamiento.

Puesto que la red ART1 solo recibe valores binarios 0 ó 1, las imágenes se someten a una separación de pixeles con base a un umbral. Se analizaron las imágenes del grupo uno y dos, y se encontró que son imágenes con 256 colores. Aunque solo utilicen dos, el blanco y el negro. Convirtiéndolas a tonos de grises y obteniendo su histograma se observó que solo se presentan dos picos en los valores 0 y 255. Por lo cual se puede determinar que si tomamos un umbral a la mitad de estos valores separaremos correctamente los pixeles obscuros y claros. Así todos los pixeles con valores menores a 128 se tomaron como valor 0 y los mayores a 128 como valor 1, y estas serán las entradas a la red.

En este caso todo el proceso de la red esta incluido en una sola función, esto se debe a que el algoritmo no es muy largo ni complejo.

En el menú principal, bajo el título RedART1, hay dos opciones, una de inicialización y otra de procesar, la primera pide las dimensiones de la red, le asigna la memoria correspondiente, inicializa sus pesos y los guarda en unos archivos ".dat", guarda los parámetros definidos para la red y por último libera la memoria utilizada. La segunda opción lee los parámetros de la red, asigna la memoria suficiente, lee los pesos guardados, los cuales pueden ser resultado de una inicialización o del proceso anterior de una o varias imágenes, procesa la imagen actual, despliega el resultado, guarda los pesos modificados y los parámetros de la red y libera la memoria usada.

La respuesta de la red será el número de una clase. Así que, si en la red la neurona ganadora de la capa de salida es la 1, el resultado será que la imagen desplegada pertenece a la clase 1, mientras que si la neurona ganadora es la 5, el resultado será que la imagen pertenece a la clase 5, por esto se dice que la red funciona como un clasificador, asignando a cada patrón de entrada una clase.

Durante la programación no se utilizaron ni el nodo g ni la regla 2 de tres, esto se debe a que se trata de una red neuronal artificial simulada, que trabaja en forma serial. Por lo cual se anula la posibilidad de que se presente información en la capa de entrada procedente de la capa de salida antes de que llegue una entrada real a la red y el nodo de ganancia se puede eliminar, pero solo en caso de simulación y cuidando como se procesa la información, ya que en una implementación física es indispensable que se tomen en cuenta ambos.

4.3 Implementación de la red ART2.

La red ART2 se implemento de manera similar a la red ART1, definiendo una clase para la red con sus respectivas variables y funciones. Pero ya que tanto la topología como el algoritmo de funcionamiento de esta red es un poco más complejo que la anterior, se optó por dividirla en pequeñas estructuras que nos facilitaran su comprensión al momento de trabajar con ellas. Debido a que la topología de las tres versiones de la red varía muy poco se utilizo la misma definición de clase para las tres versiones, cambiando solo el uso de algunas funciones de una versión a otra.

4.3.1 Variables.

Como primer paso se definieron las estructuras especiales para cada una de las capas F0, F1 y F2, es decir para las capas de entrada y salida. Cada una de las cuales contiene los punteros necesarios para manejar la memoria que utilizan sus datos. Así la capa F0 tiene un puntero para cada una de sus capas w0, x0, v0, u0, p0, q0 y dos punteros para una copia de w0 y p0. La capa F1 tiene un puntero para cada una de sus capas w, x, v, u, p y q. También contiene punteros a los siguientes datos: las salidas de la capa, los pesos ascendentes y una copia de los pesos ascendentes. Por último la capa F2 tiene punteros a: las entradas y salidas de la red, un arreglo para la restauración, los pesos descendentes y una copia de dichos pesos.

Estas estructura nos permitirán diferenciar más fácilmente entre los valores de una capa y otra. En la clase se declararon un elemento de cada estructura, y se les denominó F0, F1, F2, respectivamente, para facilitar aún más la referenciación de las subcapas.

A cada puntero de estos elementos se le asigna un arreglo de la misma dimensión del número de neuronas que corresponde a la capa a la que pertenece ya sea N para las capas F0 y F1, y M para la capa F2. Excepto los punteros a los pesos, que se les asigna una matriz de N x M para los pesos ascendentes y M x N, para los pesos descendentes. Estas matrices representan las conecciones de una capa hacia otra.

Aparte de las tres capas declaradas, es necesario declarar un puntero a la capa r, encargada del sistema de orientación, y un puntero a un arreglo auxiliar para las entradas originales de la red .

Por último, también se declaran variables para cada parámetro de la red, como son las variables: a, b, c, d y el parámetro de similitud ρ. También se declaran otras variables para los módulos de las subcapas w, v, p, u, q y r, un contador de ciclos, una variable para marcar la neurona ganadora de la capa F2 y variables para las diferencias en los cambios de valores de las capas w, q y los pesos.

4.3.2 Funciones.

Una vez vistos los datos declarados para esta clase, veremos las funciones implementadas para su funcionamiento, estas funciones se pueden dividir en dos grupos, uno incluye las funciones que utilizan indistintamente las tres versiones de la red, y el otro lo constituyen las funciones cuyo uso depende de la versión de la red de que se trate. Primero veremos las funciones comunes, y después, de acuerdo a cada versión, se explicaran las demás funciones.

- *Asigna_memoria*: Se encarga de la asignación de la memoria dinámica a todos los punteros declarados en la clase.
- *Libera_memoria:* Se encarga de liberar toda la memoria asignada dinámicamente. *Asigna_tam_capas*: Asigna las dimensión de las capas F0, F1 y F2.
- *Asigna_variables:* Asigna nuevos valores a las variables a, b, c, d, θ y ρ.
- *Encuentra_ganadora:* Encuentra la neurona ganadora de la capa F2
- *ResetF2:* marca la neurona ganadora de la capa F2 como no apta, es decir, la reseta.
- *Calcula_mod2:* hace el cálculo del modulo de un arreglo o una capa.
- *Fun_sigma:* Hace la propagación de la salida de la capa F2 hacia la capa F1.
- *Diferencia:* Calcula la suma de las diferencias entre los elementos de dos arreglos.
- *Fun_f:* aplica la función f , ecuación 3.4.10 para eliminar el ruido en la señal de entrada de la red.
- *Guarda_pesos:* Vacía los pesos actuales de la red en archivos ".dat", los pesos ascedentes w se guardan en el archivo a2pesosw.dat, mientras que los pesos descendentes se guardan en el archivo a2pesosv.dat, ambos archivos en formato binario.
- *Lee_pesos:* Se encarga de cargar los pesos guardados en los archivos ".dat" en los bloques de memoria asignados a los punteros de los pesos.
- *Inicializa:* Inicializa todas las capas de la red, los pesos descendentes a cero, los pesos ascendentes con la ecuación 3.3.14, y todas las variables con los valores por defecto.

- *Ini_arreglo:* Inicializa las capas de la red a cero, sin afectar pesos o variables.
- *Actualizar:* Actualiza los pesos cuando se ha encontrado una resonancia, solo actualiza los pesos de la neurona ganadora.
- *Hay_1_libre:* Busca en la capa F2 si todavía queda alguna neurona que no haya sido marcada como no apta para el patrón de entrada vigente.
- *Coincide* : Hace la comparación entre el modulo de la capa r y el parámetro de similitud ρ, para saber si el patrón de entrada coincide con el prototipo guardado en la red.

Como se puede apreciar todas estas funciones realizan operaciones comunes a las tres versiones de la red, como son la iniciación de valores, manejo de memoria y algunas funciones auxiliares, ahora se detallaran las funciones de propagación para cada versión de la red, que son principalmente las que varían.

4.4 Implementación de la 1era. Versión.

Se presentan a continuación las funciones que son exclusivas de la primera versión de la red ART2, cada una de ellas explicadas brevemente, pero la función que lleva el papel principal en el funcionamiento de la red y que refleja de forma mas exacta el funcionamiento del algoritmo es desglosada en sus principales pasos para poder entenderla mejor.

4.4.1 Funciones.

- *Prop_wxvu:* Propaga la información a través de las subcapas w ,x, v, y u de la capa F1 calculando sus respectivos módulos.
- *Prop_pq:* Propaga la información por las subcapas p y q de la capa F1.
- *Prop_r_a:* Calcula los valores para la capa r utilizando las subcapas p y u de la capa F1.
- *Prop_F2:* Calcula las entradas de las neuronas de la capa F2, propagando la salida de la subcapa p de la capa F1 a través de los pesos ascendentes.
- *Estabilizada:* Verifica si la capa w ha cambiado su valor respecto a la ultima iteración, para ver si la capa F1 se encuentra estabilizada.
- *Propagar1*: Esta función refleja todo el algoritmo de propagación de la red ART2, como se planteó para esta primera versión, llama a las funciones anteriores como sigue:

```
Inicializa el contador de ciclos a 0
Repite
        Repite
                Propaga las subcapas w, x, u y v de F1
                Etiqueta 1
                Propaga las subcapas p y q de F1
        Mientras la red no este estabilizada
        Propaga los valores a la capa r
        Si el patrón de entrada no es lo suficientemente parecido al prototipo
                Resetea la neurona ganadora de F2
                Pone el contador de ciclos a 1
        Caso contrario
                Si el contador esta a 1
                        Aumenta el contador
                        Propaga a F2
                        Encuentra la neurona ganadora
                        Regresa a la etiqueta 1
                Caso contrario
                        Actualiza los pesos
                        Devuelve como resultado la neurona ganadora de F2
                Fin de condición
        Fin de condición.
Mientras queden neuronas no marcadas como no aptas en F2
```

4.5 Implementación de la segunda versión de la red ART2.

Se presentan ahora las funciones correspondientes a la segunda versión de la red ART2. A diferencia de la versión anterior, las funciones en este caso procesan la información de toda una capa de la red, en vez de solo procesar la información de una o algunas subcapas de la red. Esto clarifica un poco más el procesamiento del patrón de entrada.

4.5.1 Funciones.

- *Prop_F0:* Esta función se encarga de todo el funcionamiento de la capa F0 y la utilizan tanto la segunda como la tercera versión de la red. Se encarga de hacer pasar el patrón de entrada por todas las subcapas de la capa F0 según las ecuaciones 3.4.1 a la 3.4.6, hasta que esta capa se estabilice, es decir hasta que las capas w y p varíen en cantidades muy pequeñas de un ciclo a otro.
- *Prop_F1_a:* Esta función se encarga de procesar la información a través de la capa F1, y toma su entrada de la subcapa q0 de la capa F0. También se encarga de llamar a

prop_F2 y prop_r, es decir realiza el resto del algoritmo, incluyendo la actualización de pesos.

Repite
Hace copia de p y w
Calcula valores de la capas w, p, x, q, v, u.
Calcula la variación en las capas p y w, comparando estas con sus copias.
Si no exista neurona ganadora
Propaga la información a la capa F2
Encuentra la neurona ganadora en F2
Fin de condición
Calcula los valores de la capa r y su modulo.
Si no se cumple la relación de similitud
Resetea la neurona ganadora
El contador de ciclos vuelve a cero.
Caso contrario
Actualiza pesos
Fin de condición
Aumenta el contador de ciclos
Hasta que la capa F1 se estabilice.

- *Propagar2* : esta función se encarga de llamar a prop_F0 y prop_F1_a y será la encargada de regresar el valor de la neurona ganadora al menú principal.

4.6 Implementación de la tercera versión de la red ART2.

Al igual que la anterior versión utiliza la función de prop_F0, para procesar la información en la capa F0, y otra función diferente para la capa F1, esto debido a que el flujo de información varía sólo un poco de una versión a otra.

4.6.1 Funciones.

- *Prop_F1:* Esta función es muy parecida a la prop_F1_a, pero incluye un bloque de procesamiento de las subcapas de F1, que es el que encierra la diferencia esencial entre la segunda y la tercera versión de la red, quedando de la siguiente forma:

Repite
Hace copia de p y w
Calcula valores de la capas w, p, x, q, v, u.
Calcula la variación en las capas p y w, comparando estas con sus copias.
Mientras no exista neurona ganadora
Propaga la información a la capa F2
Encuentra la neurona ganadora en F2
Actualiza la información de las capas p, u y r.

Sí no se cumple la relación de similitud
Resetea la neurona ganadora
El contador de ciclos vuelve a cero.
En caso contrario
Hace copia de p y w
Actualiza las capas w, x, q, y v
Fin de condición
Fin de ciclo
Sí existe neurona ganadora
Actualiza pesos
Guarda pesos
Fin de condición
Aumenta el contador de ciclos
Hasta que la capa F1 se estabilice.

Los renglones que aparecen en letras negras son los pasos que cambian respecto a la función prop_F1_a. Implican la actualización de las capas que intervienen en el sistema de orientación después de la propagación de la información a la capa F2, y la forma en que se define el ciclo en que están inmersos estos pasos.

- *Propagar:* es la función encargada de enlazar las funciones de prop_F0 y prop_F1 y de regresar la neurona ganadora final al menú principal.

En resumen la clase de la red ART2 tiene 3 funciones principales que se encargan de enlazar a las demás funciones, para procesar la información dependiendo de la versión de la red que se requiera usar, estas funciones son propagar1, propagar2 y propagar.

Sin embargo cabe hacer la aclaración de que estas funciones solo se encargan del flujo de la información del patrón de entrada dentro de la red. Mientras que las funciones de manejo de memoria y asignación de variables se acceden desde otros módulos.

Para aclarar lo anterior se dará una breve explicación de cómo operan las funciones que responden directamente a los eventos del menú principal.

La función encargada de incializar la red realiza los siguientes pasos:

Inicio
Pide el tipo de entrada que va a procesar la red.(histograma, centro de TF, Etc.)
Sí el tipo de entrada es el centro de la Transformada de Fourier
Pide el tamaño del centro de la transformada de Fourier.
Fin de condición
De acuerdo al tipo de entrada determina el máximo de neuronas de entrada para la red
Pide las dimensiones de la red.
Pide el parámetro teta.
Asigna tamaño de las capas de la red.
Asigna memoria a la red

Inicializa arreglos y pesos en la red
Guarda los pesos en archivos ".dat"
Guarda los parámetros de la red en archivos ".dat"
Libera memoria.
Fin de función

de

Por lo anterior podemos ver que esta inicialización es independiente de la versión a utilizar y consiste en determinar el tipo de entrada de la red, sus dimensiones y el parámetro θ. Todos ellos serán constantes a lo largo de todo el funcionamiento de la red, hasta que esta se vuelva a incializar en forma total.

Por otra parte, la función encargada de procesar las entradas en la red sigue los siguientes pasos:

Lee los parámetros de la red ART2
Pide el valor de las variables a, b, c, d.
Asigna los valores de a, b, c, y d a la red ART
Pide el parámetro de similitud a utilizar
Asigna el parámetro de similitud a la red ART
Asigna la memoria de la red
Inicializa los arreglos de la red
Lee los pesos guardados en los archivos ".dat"
Sí el tipo de entrada es:
1. *Calcula la transformada de Fourier*
   Asigna el centro de la TF a las entradas de la red
2. *Asigna la imagen original a las entradas de la red.*
3. *Calcula el histograma de la imagen*
   Asigna el resultado a la entrada de la red.
4. *Calcula los momentos de la imagen*
   Asigna el resultado a la entrada de la red

Fin de elección
Pregunta la versión de la red a utilizar
Si la elección es:
1. *llama a la función propagar1*
2. *llama a la función propagar2*
3. *llama a la función propagar*

Fin de elección
Guarda los pesos actuales en los archivos ".dat"
Guarda los parámetros de la red en los archivos ".dat"
Muestra el resultado de la red
Libera la memoria utilizada
Fin de función

Analizando los pasos anteriores se puede apreciar que la asignación de variables, memoria, cálculo de las entrada de la red y despliegue de resultados se mantiene igual para las tres versiones.

Otra opción posible para la implementación de las tres versiones, es crear una clase base que agrupe las funciones comunes entre ellas, y luego derivar las tres diferentes versiones redefiniendo en cada una de ellas las funciones que se diferencian. Pero debido a que cada versión

se plantea como una mejora de la anterior, se decidió dejarlas en una misma clase, con el fin de eliminar las funciones no óptimas al final del trabajo, y contar con solo una clase para la red ART2.

4.7 . Implementación de la red de Propagación del error hacia atrás o "Backpropagation".

La programación de la red Backpropagation se hizo de manera muy similar a la de la red ART2. Primero se definieron las estructuras para cada uno de los tres tipos de capas que posee la red. Ninguna de estas tres estructuras es igual a alguna de las que se usaron para la red ART, esto debido a que manejan otros datos. En segundo lugar se definió la clase de la red, incluyendo en ella elementos de los tres tipos de capas más algunas otras variables para el manejo de los demás datos. También se definieron una serie de funciones auxiliares para la red. En el caso de la red Backpropagation su aprendizaje es OFFLINE, de acuerdo con la definición dada en el capítulo 3. Sin embargo en algunos libros consideran que la red tiene un aprendizaje "ONLINE" cuando los cambios en los pesos se hacen cada vez que se termina de procesar un patrón de entrada. Mientras que se considera un aprendizaje OFFLINE cuando los pesos se alteran despues de haber procesado todos los patrones de entrada, es decir cada ciclo. Aunque en realidad los dos tipos de aprendizaje son OFFLINE, porque se realizan en la fase de aprendizaje y no en la de funcionamiento, se utilizara esta convención de nombres. Pero sólo en el caso de la red Backpropagtion. Se implementaron ambos aprendizajes y dado que sólo es necesario modificar un par de funciones, se optó por definir éstas en la misma clase. Modificando solo el llamado de la función dependiendo del tipo de aprendizaje a utilizar.

4.7.1 Variables

Empezaremos por ver la definición de las estructuras de las capas. La capa de entrada sólo contiene un puntero a las entradas de esta capa. La capa oculta tiene un puntero para cada uno de los siguiente arreglos de datos: entradas, salidas, error recibido, error, termino de error, ajuste de error, los pesos ascendentes y ajuste de pesos. Por otro lado, la capa de salida tiene un puntero para los arreglos de entrada, salida obtenida, salida deseada, error, término de error, ajuste de error, pesos ascendentes y ajuste de pesos.

En este caso, los pesos están asociados a la capa destino, en lugar de la capa origen como ocurría en la red ART2, esto con el fin de facilitar la propagación del error hacia atrás.

Dentro de la definición de la clase se agregan otras variables para el manejo de otros datos como son: tres enteros para las dimensiones de cada una de las capas, Num_M para la capa de entrada, Num_P para la capa oculta y Num_N para la capa de salida por último una variable para la tasa de aprendizaje alfa.

4.7.2 Funciones

De igual forma que ocurrió en la implementación de la red ART2, existen algunas funciones que se crearon para el manejo de memoria, asignación de variables e inicialización de valores y que no tienen que ver directamente con el procesamiento de la información, estas son las funciones que se detallaran primero, y después se pasará a explicar aquellas funciones que realizan el procesamiento de la información.

- *Asigna_memoria:* Reserva la memoria necesaria para el funcionamiento de la red.
- *Libera_memoria:* Se encarga de liberar toda la memoria asignada dinámicamente en la red.
- *Inicializa_ajustes:* Inicializa a cero los arreglos de los ajustes que se aplican tanto a los pesos como a los términos de error.
- *Carga_pat_sal:* Guarda los valores del patrón de salida deseada en la capa de salida.
- *Guarda_pesos:* Guarda los pesos actuales en archivos ".dat". Los pesos v en el archivo "bcpesosv.dat", mientras que los pesos w, en el archivo "bcpesosw.dat".
- *Lee_pesos:* Lee los pesos guardados en los archivos ":dat" y los asigna a las matrices de los pesos,
- *Inicializa_pesos*: Realiza la iniciación de los pesos v y w. De forma aleatoria en el rango [-0.5 a 0.5].

Ahora veremos las funciones que se encargan del procesamiento y flujo de la información a través de la red:

- *Propaga_oculta:* Se encarga de propagar los valores de la capa de entrada a través de los pesos v, hacia la capa oculta. Calcula la entrada y salida para cada neurona de esta capa, aplicando la función f. Ecuación 3.8.1.
- *Propaga_salida:* Se encarga de propagar los valores de la capa oculta a través de los pesos w, hacia la capa de salida, calculando la entrada para cada neurona de esta capa, también calcula las salidas obtenidas de la red aplicando la función f, ecuación 3.8.1, a la entrada de cada neurona.

- *Cal_eror_salida:* Calcula el termino de error para cada neurona de salida. Ecuación 3.8.11.
- *Cal_eror_reb_ocul:* Calcula el error recibido en cada neurona de la capa oculta, sumando todos los errores que le llegan provenientes de la capa de salida.
- *Cal_error_oculta:* Calcula el termino de error para cada neurona de la capa oculta usando la ecuación 3.8.14 al error recibido en cada neurona de dicha capa.
- *Actualiza_pesos*: Esta función se encarga de actualizar los pesos y los términos de error de la red utilizando directamente los errores de cada capa y la tasa de aprendizaje, es decir, se encarga del aprendizaje ONLINE.
- *Error_general:* Esta función calcula y retorna el valor del error medio para un patrón de entrada dado.
- *Calcula_ajuste*: Calcula el ajuste de los pesos y los términos de error para un patrón dado y los acumula en los arreglos de ajuste.
- *Aplica_ajuste:* Actualiza los pesos y los términos de error aplicando el ajuste acumulado en los arreglos de ajuste. Es decir, en la función encargada de aplicar el aprendizaje OFFLINE.
- *Aprender:* Se encarga de llamar a las demás funciones para completar un ciclo de la fase de entrenamiento cuando se utiliza el aprendizaje ONLINE. De acuerdo a la siguiente secuencia:
    - *Propaga_oculta*
    - *Propaga_salida*
    - *Cal_error_salidad*
    - *Cal_error_reb_ocul*
    - *Cal_error_oculta*
    - *Actualiza_pesos*
    - *Cal_error_general*

- *Aprendedos:* esta función trabaja de manera similar que la función aprender, solamente que se utiliza en el aprendizaje OFFLINE, por lo que no llama a la función de actualiza_pesos.
- *Clasifica:* esta función se encarga de enlazar otras funciones para realizar el funcionamiento de la red en la fase de funcionamiento. Determina la salida de la red encontrando la neurona con salida más cercana a uno en la capa de salida, realizando los siguientes pasos:

- *Propaga_oculta*
- *Propaga_salida*
- *Encuentra la neurona con más alto valor de salida en la capa de salida.*
- *Retorna la neurona seleccionada.*

Este funcionamiento se debe a que se trata se igualar lo más posible el funcionamiento de esta red con la red ART2, la cual da como resultado final una neurona ganadora. Por lo que en esta red se pretende hacer lo mismo, pero en otros casos se podría tomar como salida la combinación de todos los valores obtenidos en la capa de salida y darles su respectiva interpretación.

Todas las funciones antes descritas están implementadas dentro de la red. Pero debido a que en la fase de entrenamiento se necesita el acceso a varios patrones de entrada, se crearon una serie de funciones que no pertenecen a la clase, sin embargo, ayudan al funcionamiento de la red sobre todo en la fase de entrenamiento. Dichas funciones están agrupadas en los módulos que responden directamente a eventos del menú principal.

Antes de ver estas funciones, es necesario dar una breve explicación de cómo recibe sus entradas la red backpropagation durante la fase de entrenamiento Durante esta fase la red requiere tener ya un conjunto definido de patrones de entrada, los cuales no pueden modificarse a mitad del entrenamiento o cambiarse una vez que la red ya esta en funcionamiento, si llegase a ocurrir algo así, es necesario repetir la fase de entrenamiento con el nuevo conjunto de datos modificados. Por todo esto es necesario crear un mecanismo que nos permita definir el conjunto de datos que servirán como entrada a la red backpropagation durante la fase de entrenamiento. Se optó por crear un archivo para definir un conjunto de datos de entrenamiento. Este archivo a su vez contiene los nombres de tres archivos. Uno de los cuales guarda los parámetros principales de la red como son las dimensiones de las capas y la tasa de aprendizaje. Los otros dos archivos se encargan de guardar los patrones de entrada y salida deseada de la red por separado. Los datos se deben ir agregando a estos archivos por pareja, es decir, al agregar un patrón de entrada se debe agregar al mismo tiempo el patrón de salida deseada. Al finalizar de agregar todos los datos que serán necesarios para el aprendizaje de la red, se puede seleccionar la opción de aprendizaje de la red. Entonces el sistema se encargara de leer todos los patrones pertenecientes a un conjunto de datos e introducirlos en la red en la secuencia correcta. Para poder diferenciar entre los archivos de un conjunto de datos y otro se utiliza un número de control, el cual va aumentado su valor conforme se van creando nuevos conjuntos de datos.

Para poder definir estos conjuntos de datos se crearon dos opciones principales, una que permite crear un nuevo conjunto de datos y otra que permite agregar datos a un conjunto de datos creado con anterioridad. Cada una de estas opciones esta ligada a su respectiva función, las cuales son OnBcCreaCd y OnBcAgreCd. Mientras que las funciones que se encargan del aprendizaje son OnBcAprCd y OnBcAprCddos, una para el aprendizaje ONLINE y otra para el aprendizaje OFFLINE. Por último, también se necesita una función que se encargue de la etapa de funcionamiento de la red, esta función es OnRbClass. El pseudocódigo de cada una de estas funciones se encuentra en el apéndice C.

Capítulo 5: Resultados

En este capítulo se presentan los resultados obtenidos al procesar los distintos grupos de imágenes en las diferentes redes neuronales artificiales implementadas. Primero se presentan los resultados en la red ART1, los cuales nos ayudaran a destacar las características más importantes de la red ART. En este caso no se realizará una comparación directa con los resultados de otras redes. Ya que por trabajar sólo con valores discretos el poder de la red disminuye en gran medida respecto con las otras dos redes, la red ART2 y la backpropagtion.

Después se presentan los resultados al procesar un mismo grupo de imágenes con las tres distintas versionès de la red ART2. Esto nos permitirá distinguir la diferencia en el desempeño de estas tres versiones. Al mismo tiempo que nos brindará la oportunidad de destacar algunas de las características que difieren a la red ART2 de la red ART1.

En tercer lugar se presentan los resultados obtenidos al procesar un grupo de imágenes usando la red backpropagation al variar su modo de aprendizaje.

Por último aparecen los resultados de procesar el mismo conjunto de imágenes con las redes ART2 y backpropagtion, lo que nos permitirá darnos cuenta de las diferencias que se presentan entre estas dos redes, permitiéndonos así, hacer una comparación entre ellas.

5.1 Red ART1

Debido a que la red ART1 solo acepta valores binarios, se usó sólo el conjunto de imágenes de figuras geométricas.

Se utilizaron dos subconjuntos de este tipo de figuras, ambos sólo cuentan con cuatro formas principales que son: el círculo, el cuadrado, el rectángulo y el triángulo. De cada una de ellas existen 7 figuras distintas. En el primer grupo, cada figura sufre una degradación muy pequeña en los bordes respecto a la figura original, que sería la número 1. Mientras, que en el segundo grupo no sólo hay pequeñas degradaciones en los bordes, sino que también existe un desplazamiento de la figura respecto al centro de la imagen total. Esto nos permitirá observar, de forma más directa, como se comporta la red ante diversos grados de variación en los datos.

Los nombres de las imágenes se repiten de un grupo a otro, pero no así el contenido de las imágenes, que varía de un grupo a otro. Ya que los datos se procesan por grupos no hay riesgo de confundir las imágenes de un grupo con otro.

5.1.1 Asignación de clases.

Debido a que la red va asignando las clases, conforme procesa datos que no coincidan con los que ya tiene almacenados, se optó por procesar primero una imagen de cada forma. Esto con el fin de que la red pueda asignarle a cada forma una clase diferente. Ya que sí procesamos primero varias imágenes de la misma forma y estas no cumplen el mínimo de similitud establecido para la red, se corre el riesgo que ésta le asigne a cada una de las imágenes una clase diferente. Así cuando se procesen las imágenes de otra forma geométrica ya no quedará ninguna clase disponible para asignarle.

El orden del procesamiento de las imágenes, para ambos grupos, queda entonces de la siguiente forma[1]:

1. circulo1.bmp
2. cuadro1.bmp
3. rectangulo1.bmp
4. triangulo1.bmp
5. circulo2.bmp
6. cuadro2.bmp
7. rectangulo2.bmp
8. triangulo2.bmp
9. circulo3.bmp
10. cuadro3.bmp
11. rectangulo3.bmp
12. triangulo3.bmp
13. circulo4.bmp
14. cuadro4.bmp
15. rectangulo4.bmp
16. triangulo4.bmp
17. circulo5.bmp
18. cuadro5.bmp
19. rectangulo5.bmp
20. triangulo5.bmp
21. circulo6.bmp
22. cuadro6.bmp
23. rectangulo6.bmp
24. triangulo6.bmp
25. circulo7.bmp
26. cuadro7.bmp
27. rectangulo7.bmp
28. triangulo7.bmp

El funcionamiento de la red ART1 esta ligado directamente con el parámetro de similitud escogido, por lo que se realizó el procesamiento de las imágenes con distintos valores para dicho parámetro. Se escogieron los siguientes valores: 0.25, 0.50, 0.75, 0.80, 0.85, 0.90, 0.95, 0.97, 0.99 y 1, para poder determinar como se comporta la red cuando se le pide diferentes grados de exactitud en sus comparaciones.

[1] Los siguientes son nombres de archivos en DOS, por lo que no llevan acento.

5.1.2 Resultados del grupo uno

Para este caso, se utilizó como entrada la imagen completa, es decir los 400 pixeles. Como salida se definieron 4 neuronas, por lo que sólo existirán 4 clases. Las cuales se asignaron en el mismo orden en que se presentan las primeras figuras. La clase uno para el círculo, la clase dos para el cuadrado, la clase tres para el rectángulo y por último la clase cuatro al triángulo.

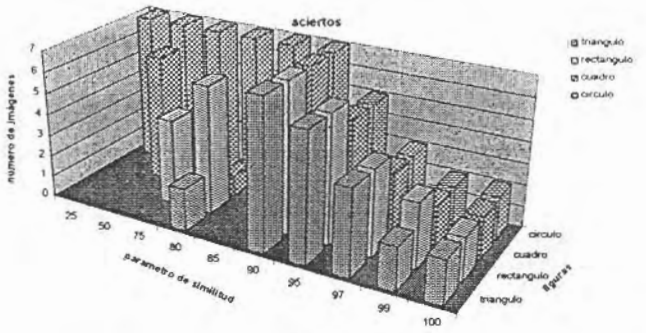
Los resultados obtenidos se muestran en la tabla 1 del apéndice D. Para $\rho = 0.25$ la red considera de la misma clase a varias de la primeras imágenes procesadas, puesto que no se tienen que parecer mas que en un cuarto del total de sus pixeles. Esto ocasiona que el prototipo asignado a dicha clase se modifica radicalmente cada vez que se procesa una nueva imagen. Guardando sólo los pixeles que son comunes a todas las imágenes clasificadas en la misma clase.

Para ρ igual a 0.75 tenemos que la red ya es capaz de diferenciar las dos primeras formas geométricas, el círculo y el cuadrado. Aunque todavía presenta problemas con el rectángulo y el triángulo. Esto se debe a que sí sobreponemos estas figuras al cuadrado, encontraremos que tienen muchos pixeles en común, esto es lo que ocasiona que la red no los pueda diferenciar y les asigne la misma clase. Además, esto también ocasionará una erosión en los prototipos de estas clases. Que se ve reflejado al paso del tiempo cuando la red ya no diferencia las formas geométricas del círculo y del cuadrado, asignándoles diferentes clases a la misma forma.

Para ρ igual a 0.90, podemos observar que la red ha logrado clasificar con éxito todas las figuras, lo que nos indica que este es el valor mínimo correcto para el buen funcionamiento de la red cuando se procesan este tipo de imágenes. Puesto que aquí no existe confusiones entre las formas de las imágenes no hay riesgo de que los patrones prototipo de la red sufran degradaciones. Sin embargo, debido a que las imágenes de la misma forma geométrica varían. existe todavía un grado de degradación para los prototipos.

Para ρ igual a 0.99, la red logra clasificar las primeras imágenes que procesa de forma correcta, ya que son las que determinan la clase, y algunas otras que no varían mucho. Pero debido al ligero cambio del prototipo de las clases, cuando procesa las imágenes de índice 3, éstas ya no logran tener la similitud requerida y no se realiza su correcta clasificación.

Ahora, una forma diferente de ver estos resultados es por medio de gráficas. Observando la figura 5.1 obtenemos las siguientes conclusiones.



Gráfica 5.1
Gráfica de aciertos de la red ART1 con imágenes de figuras geométricas.

Esta gráfica puede resultar un poco engañosa, ya que a primera vista parece que la red clasifica bien la figura del círculo utilizando tanto parámetros bajos como medios de ρ. Pero en realidad lo que ocurre es que el círculo es la primera figura que se presenta a la red, por lo cual es a la que le asigna la clase 1, y como hemos visto, para parámetros bajos de ρ la red clasifica a la mayoría de las figuras en la primeras clases, lo que hace coincidir la clase real del círculo con la clase obtenida por la red para la mayoría de los casos.

Para la figura del cuadrado, tenemos que el comportamiento de la red es un poco más realista. Para valores de ρ bajos y la red clasifica a la mayoría de figuras en la primera clase, que no es la clase correcta de la figura, por tanto se dice que la red falla. Sin embargo existe una excepción, que se da cuando ρ tiene valores de 0.50, aquí aparentemente la red obtiene un gran número de aciertos al clasificar correctamente la figura, pero si revisamos la tabla D.1 encontraremos que la red sólo está separando las imágenes en dos clases, por lo que no se puede decir que su funcionamiento sea bueno.

Debido a que al rectángulo se le asigna la tercera clase, los aciertos presentados para esta figura, cuando ρ es pequeño, ocurren sólo cuando la red ya ha podido dividir las imágenes al menos en tres clases. Pero esto no significa que la red este funcionando correctamente con dichos valores. Sin embargo, si comparamos esta gráfica con las anteriores veremos que su comportamiento a

partir de $\rho = 0.90$ es muy parecido. Lo que significa que con dichos valores la red de comporta de una forma semejante para estas figuras. Por ende los resultados se vuelven más confiables.

Se puede observar que para valores pequeños de ρ casi no se tiene aciertos, esto se debe, a que con estos valores la red sólo divide a las imágenes en 1 o 2 clases, Ya que al triángulo se le asigno la clase 4 no se presentan coincidencias entre la clase obtenida por la red y la clase real de la figura. Sin embargo, para valores de ρ iguales o mayores a 0.90 encontramos un comportamiento muy similar al presentado en las otras figuras, así podemos decir que la red se comporta de forma muy similar para las cuatro figuras a partir de $\rho = 0.90$.

Observando la gráfica 5.1 y la tabla D1 podemos obtener las siguientes conclusiones:

- La red se comporta de una manera muy aleatoria cuando ρ esta entre 0.25 y 0.85.
- Se obtiene un máximo de eficacia en la red cuando $\rho = 0.90$.
- La red se comporta de forma similar para las cuatro figuras, con valores de ρ entre 0.95 y 1.
- Para valores de ρ mayores a 0.90 la red deja de confundir las figuras de una clase con otra, pero aumenta el número de imágenes que no puede clasificar.

Existen todavía más conclusiones que se pueden obtener del análisis de las gráficas y las tablas, pero por el momento estas son las más importantes a considerar para este trabajo.

5.1.3 Resultados del grupo dos.

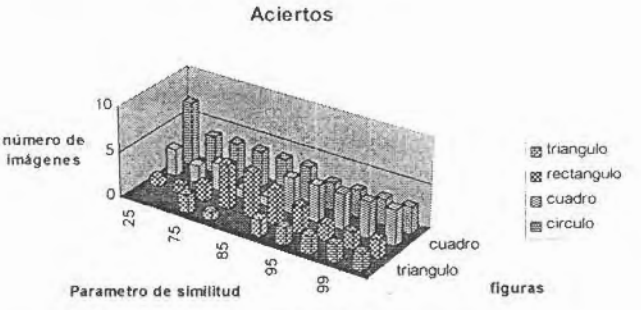
Para el segundo grupo de imágenes, se volvió a repetir el mismo procedimiento, los resultados están en la tabla D.2.

Al analizar dicha tabla, podemos apreciar que ninguno de los valores de ρ hace que la red clasifique de forma correcta todas las imágenes de este grupo. Además, a partir del $\rho = 75$ empiezan a aparecer muchas imágenes que la red no puede clasificar y que se toman como errores en el momento de hacer las gráficas.

Al igual que con el grupo anterior de imágenes, podemos ver que para valores muy pequeños de ρ, por ejemplo $\rho = 0.25$, la red clasifica a la mayoría de las imágenes dentro de la

misma clase. También podemos apreciar que sigue revolviendo imágenes de diferentes formas geométricas dentro de una misma clase, hasta que ρ = 0.90 En este valor, si bien la red no logra clasificar a todas las imágenes en forma correcta, ya no revuelve imágenes de distintos tipos en una clase. A partir de aquí, la red también se vuelve a comportar de una manera uniforme para las cuatro figuras geométricas.

La gráfica de estos datos es:



Gráfica 5.2

Gráfica concentrada de los aciertos de la red ART1 con imágenes de figuras geométricas.

En la gráfica notamos que se repite el mismo patrón que en el otro conjunto de imágenes. Es decir, la gráfica resulta engañosa para la figura del círculo, debido a que con valores bajos de ρ la red funciona bien. Pero esto se debe a que como es la primera imagen, la red le asigna la clase 1 y dicha clase también se la asigna a la mayoría de imágenes. Por lo que la coincidencia entre la clase real y la obtenida se eleva. Aunque esto no signifique que la red funcione adecuadamente.

Para la figura del cuadrado, todavía podemos observar que para valores de ρ menores a 90, la red se comporta de forma muy diferente, mientras que, arriba de dicho valores, se conservan el número de aciertos en imágenes del cuadrado. Si se observa la tabla D2, podemos ver que son las figuras de los cuadrados con índice del 1 al 4, las que resultan bien clasificadas en este rango, y las otras siempre son clasificadas en otra clase o no se pueden clasificar.

Para el rectángulo los mejores desempeños existen cuando ρ es igual a 0.80 y 0.85, decreciendo para valores más altos hasta llegar a 2 aciertos, que son los que se conservan. Entonces podemos decir que al igual que en el caso del conjunto anterior de imágenes, la red se comporta de manera más uniforme si ρ tiene valores mayores o iguales a 0.90.

Por último, para el triángulo se muestra claramente que la red tiende a comportarse en forma muy similar, que con las otras tres figuras cuando ρ sea mayor o igual a 0.90. Mientras, que su comportamiento debajo de estos niveles es muy diferente de una figura a otra.

Lo anterior se debe a que las imágenes del segundo grupo difieren más entre si, por lo que la cantidad de pixeles que cambia de una imagen a otra supera el margen de error.

En resumen, podemos ver que para valores menores a $\rho = 0.90$ la red se comporta de forma desigual para cada figura, y para valores de 0.90 o mayores, su comportamiento se hace uniforme, aunque no llega nunca a tener el cien por ciento de aciertos para ninguna figura. Lo anterior nos indica que la red no puede diferenciar una imagen de otra, lo cual se puede comprobar al revisar las tablas D1 y D2. Donde se puede ver que la red primero solo utiliza una clase, es decir cataloga a todas las imágenes como iguales, y conforme ρ aumenta se incrementa el número de clases que diferencia la red. En ambos casos cuando la red utiliza $\rho = 0.90$ se obtiene un máximo de aciertos, lo que nos indica que con dicho valor la red puede distinguir una figura de otra, siempre y cuando las variaciones de las imágenes no sean muy grandes. Para valores de ρ más grandes los aciertos disminuyen, esto se debe a que la red ya no clasifica muchas imágenes dentro de sus respectivas clases, debido a que la similitud entre ellas y el prototipo guardado en la red es menor al requerido por ρ. Pero tampoco las clasifica dentro de otras clases, es decir, la confusión de clases disminuye e incluso casi llega a desaparecer. Presentándose únicamente en el caso del círculo en ambos grupos, en el segundo en las figuras 4, 5, 6, y 7 y en el primer grupo en la figura 7. En ambos casos se clasificó al círculo como perteneciente a la clase 2 que es la asignada al cuadrado. Por ende sí observamos las figuras utilizadas veremos que el círculo cabe dentro del cuadrado y la cantidad de pixeles que tienen en común es muy grande.

En el primer grupo, debido a que las degradaciones del cuadrado son en la orilla, es lógico pensar que el prototipo almacenado en la red que corresponde a esta clase se va degenerando en las orillas. Lo que lleva a que se parezca cada vez más a un círculo, por lo cual la red llega a confundir estas dos imágenes. Mientras, que en el grupo dos, sucede algo similar, aunque en dicho caso las figuras se desplazan por las orillas de la imagen. Así, los pixeles que se conservan en los prototipos son los del centro de la figura, los cuales no cambian de una imagen a otra. Lo anterior ocasiona que el círculo de la figura 7, que está centrado en la imagen coincida más fácilmente con el prototipo del cuadrado.

Como conclusión final de estos casos, podemos decir que la red ART1 funciona mejor con valores de ρ cercanos a la unidad. Pero si estos son demasiados altos se puede perder también buen porcentaje de la eficacia de la red. Por lo que para un conjunto de entradas hay que encontrar el valor de ρ que nos proporcione los mejores resultados evitando la confusión de clases y al mismo tiempo tolerando pequeñas variaciones entre las imágenes de una misma clase. Sin embargo, como

se ha visto pueden presentarse casos en los cuales no se logre el cien por ciento de efectividad. Debido a que las entradas de una misma clase varían mucho entre si y al mismo tiempo se parecen más a las entradas de otras clases. En estos casos, lo que se puede hacer es cambiar el contenido de las entradas, de manera tal qué reflejen de una forma más fidedigna las características comunes de una clase.

En los casos anteriores se utilizaron la imágenes originales como entrada, sin ningún procesamiento anterior, esto con el fin de poder observar de una forma más clara el comportamiento de la red. Pero para obtener un mejor funcionamiento es necesario preprocesar las imágenes y extraer de ellas las características que diferencian una clase de otra y tomar dichas características como las entradas de la red. Esto mejoraría en gran medida el funcionamiento de la redes.

5.2 Resultados de la red ART2

Ahora se presentan los resultados obtenidos con la red ART2. Debido a que esta red si es capaz de procesar entradas de valores reales, se amplía la gama de entradas que utiliza esta red. Para este trabajo se utilizaron tres tipos principales de entrada, el primer tipo, es el valor real de los pixeles de la imagen utilizando los mismos grupos que se procesaron con la red ART1. Esto con el fin de comparar los resultados y ver la efectividad de la red ART2. El segundo tipo de entrada son los momentos de una imagen. Donde se utiliza una base de letras blancas sobre fondo negro., El tercer y último tipo de entrada serán los valores del centro de la transformada de Fourier de una imagen, donde se utiliza una base de imágenes de rostros humanos. Los tres tipos de entradas nos permitirán apreciar un poco mejor el potencial de las redes en el reconocimiento de diversos patrones dentro de una imagen.

Basándonos en los resultados de la red ART1, podemos ver que la red trabaja mejor cuando el valor de ρ se acerca a la unidad. Por tanto, para el primer tipo de entrada utilizaremos los valores de 0.25, 0.50, 0.80, 0.90, 0.95, 0.97, 0.99 y 1.00. Los dos primeros valores solo se utilizan para confirmar el funcionamiento de la red con valores de ρ bajos. Los demás valores son para detectar cual es el valor con el cual se obtiene el mejor funcionamiento de la red.

5.2.1 Resultados del grupo dos.

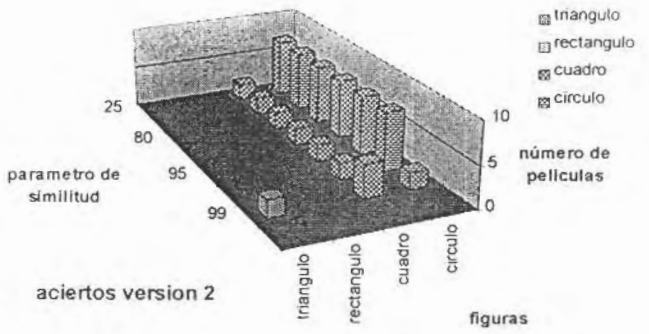
Utilizando el segundo grupo de imágenes y la primera versión de la red ART2 obtenemos la tabla de resultados D3.

Observando dicha tabla podemos ver que la red no logra distinguir la imagen de una forma geométrica de otra forma, incluso para valores de ρ igual a 1, la red sigue clasificando a todas las imágenes como pertenecientes a la misma clase.

Utilizando la segunda versión, obtenemos la tabla D4.En donde podemos observar que la red ya es capaz de separar las imágenes en dos clases, para los valores de ρ menores a 0.99, y en 4 clases para ρ = 0.99. Aunque no asigna correctamente las clases y confunde a la mayoría de las imágenes, clasificando como una sola clase a imágenes con diferentes figuras geométricas. En algunos casos si se logra la correcta clasificación en varias imágenes, pero esto se debe a coincidencias más que nada. Ya que la red sólo clasifica las imágenes dentro de una segunda clase después de haber procesado varias imágenes y no porque distinga una forma de otra. Sin embargo, el hecho de que ya pueda separar las imágenes en distintas clases indica que esta versión es mucho más sensible que la anterior, pero su funcionamiento no se puede considerar todavía como bueno.

Usando la tercera versión obtenemos la tabla D5.En donde podemos ver que la red ya tiene un mejor funcionamiento con valores de ρ mayores a 0.95. Mientras, que con valores menores a éste la red sigue clasificando a todas las figuras como pertenecientes a una misma clase. Con ρ = 0.95 la red ya es capaz de diferenciar las formas de círculo, cuadro y rectángulo, sin embargo nunca se logra el cien por ciento de efectividad. Esto se debe a la gran variación de posición de las figuras dentro de las imágenes, ya que las imágenes en que la red falla son aquellas en las que las figuras están desplazadas respecto a la imagen 1 de esa figura. Cuando se aumenta el parámetro de vigilancia a 0.97 ó 0.99, podemos observar que la red deja de clasificar varias imágenes, sin embargo no confunde las clases. Por último, cuando ρ es igual a 1 la red sólo clasifica la primera imagen de cada clase que se le presenta, y las demás imágenes, como no son exactamente iguales a las primeras, ya no puede clasificarlas.

Para la versión dos tenemos:



Gráfica 5.3
Gráfica concentrada de los aciertos de la red ART2 (2da. Versión).

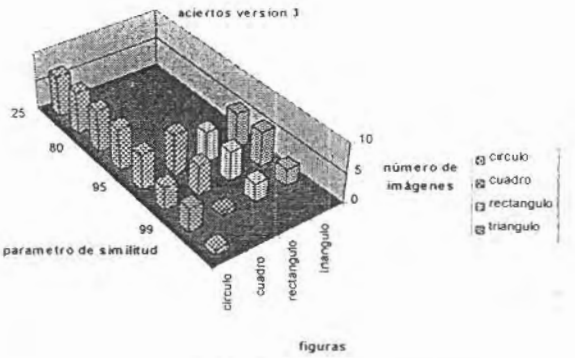
Como el círculo es la primera figura que se le presenta a la red, podemos ver que la red siempre la reconoce para valores de ρ menor o igual a 0.97, pero en realidad para los valores de ρ de 0.25 a 0.90 es una coincidencia, ya que la red clasifica como clase uno a todas la imágenes y esta clase coincide con la del círculo

Para el cuadrado ocurre algo muy similar que al círculo, la red separa las imágenes en dos clases para ρ menor o igual a 0.95, y como la case dos coincide con la clase del cuadro, aparentemente la red si reconoce esta figura en dos imágenes. Pero en realidad esta clase dos también incluirá imágenes de otras figuras. Por tanto no se debe tomar como un correcto resultado de la red.

En el caso del rectángulo la red nunca logra clasificarlo correctamente por lo que en la gráfica no tiene ningún acierto. En el caso del triángulo solo se logra su correcta clasificación para dos imágenes cuando ρ es igual a 0.99. Pero si se observa la tabla D4 veremos que en la clase 4, que es la clase correspondiente al triángulo, la red también incluye otras figuras. Por lo que tampoco se debe interpretar como un correcto funcionamiento de la red, si no simplemente es una coincidencia.

En la gráfica podemos observar que la red nunca logra aciertos al mismo tiempo en las cuatro formas, lo que indica que en realidad nunca logra una clasificación correcta de las imágenes. Entonces, deducimos que su funcionamiento no es correcto, ya que los aciertos presentados son debidos más que nada a coincidencias.

Para la versión tres se obtiene la siguiente gráfica.



Gráfica 5.4
Gráfica concentrada de los aciertos de la red ART2(versión 3)

Para valores de ρ igual o menores a 0.90, se descartan los datos, porque ahí la red clasifica todas las imágenes iguales. Sin embargo si observamos la tabla D5 vemos que para valores de ρ igual o mayor a 0.95 la red ya no confunde las imágenes. Por lo que se pueden tomar estos resultados como correctos. Entonces podemos ver que la red logra su mejor funcionamiento cuando ρ es igual a 0.95 y disminuye conforme aumenta este valor, llegando al mínimo de un acierto cuando es igual a 1. También podemos ver que en estos valores no logra clasificar correctamente todas la imágenes del círculo y falta una. Por tanto la red no logra una efectividad del cien por cien.

Para el cuadrado, la red si logra su funcionamiento óptimo con ρ = 0.95 y su efectividad decrece si se aumenta el valor de ρ, lo que se parece a los resultados obtenidos con el círculo.

Para el rectángulo, observamos que la red también obtiene un buen funcionamiento para los valores de ρ = 0.95 y 0.97. Mientras, que si se aumenta este valor el número de aciertos disminuye, llegando a cero aciertos cuando ρ es igual a 1. Sí se observa la tabla D5 de resultados descubrimos que esto se debe a que la red no clasifica la figura del cuadro y por tanto recorre el número de clase para las demás figuras, lo que ocasiona que ninguna de ellas resulte bien clasificada.

La gráfica para el triángulo se comporta de la misma forma que la gráfica del rectángulo, por lo que se puede decir que la red reacciona de igual forma para estas dos figuras..

Comparando los resultados obtenidos para todas las figuras, podemos observar que la red tiene un buen funcionamiento general para ρ igual a 0.95. Puesto que tiene un buen número de aciertos para todas la figuras. Con este valor es con el que se obtiene los mejores resultados. Mientras que los aciertos disminuyen conforme el valor de ρ aumenta, de forma similar para todas

las figuras. Aunque el cuadro es el más sensible ya que es el único que alcanza su óptimo, pero también es en el que más rápido decrece el número de aciertos al acercarse ρ a 1.

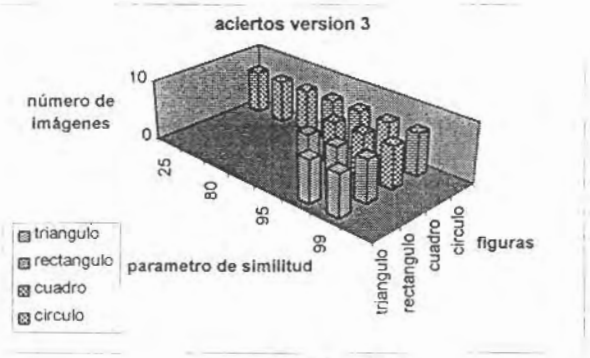
5.2.2 Resultados del grupo uno.

Para el primer grupo de imágenes se volvió a repetir el proceso para cada versión, y los resultados son:

Para la primera versión, la red clasifica todas las imágenes como pertenecientes a la clase1, aun cuando ρ es igual a 1, es decir se presentan los mismos resultados que con el grupo anterior de imágenes, lo que confirma que esta versión de la red tiene un rendimiento muy bajo.

Para la segunda versión, la red vuelve a clasificar todas las imágenes como pertenecientes a la clase 1, para todos los valores de ρ, excepto para el valor de 1, donde la red no logra clasificar ninguna imagen.

Para la versión tres, la red arroja los resultados de la tabla D6. Donde se puede ver que la red logra un funcionamiento óptimo para los valores de ρ igual a 0.97 y 0.99, y disminuye para ρ igual a 1. En este último, la red sólo clasifica una imagen de cada figura, pero en una clase diferente a la correcta. Mientras que para valores menores o iguales a 0.90 clasifica todas las imágenes como de la clase 1.



Gráfica 5.5.
Gráfica concentrada de aciertos de la red ART2 (versión 3) para imágenes de figuras geométricas.

Para el círculo la red aparenta funcionar para todos los valores de ρ a excepción de 1, pero como ya vimos antes, en el caso del círculo esto es engañoso por que los resultados de los valores de ρ igual o menores a 0.90 son más coincidencias que el funcionamiento correcto de la red.

Para las figuras del cuadro y del rectángulo, la red contienen la misma cantidad de aciertos en los mismos valores de ρ. Observamos que la red tiene un óptimo desempeño para los valores de ρ igual a 0.95, 0.97 y 0.99, mientras que para los otros valores su desempeño es nulo.

Para la figura del triángulo, observamos que el funcionamiento en ρ igual a 0.95 es nulo, y si observamos la tabla de resultados D6 veremos que se debe a que la red confunde el triángulo con el rectángulo cuando ρ tiene dicho valor, asignándole la misma clase que la del rectángulo. Por lo cual no se presenta ningún acierto para las imágenes de los triángulo. Mientras que con valores de ρ igual a 0.95 y 0.97 se logra la clasificación correcta de las siete imágenes de triángulo.

En la gráfica podemos ver que la red tiene un funcionamiento similar para las cuatro figuras cuando ρ vale 0.97 y 0.99, además de que logra el cien por ciento de aciertos. Lo anterior nos indica que la red si está clasificando correctamente las imágenes y su funcionamiento es óptimo en dichos valores.

Resumiendo, de las tres versiones presentadas en este trabajo hemos visto que la que mejor resultados produce es la versión tres. Ya que usándola sobre los dos grupos de imágenes se logra un buen desempeño de la red al momento de clasificar las imágenes, obteniendo los mejores resultados cuando ρ varia entre los valores de 0.95 y 0.99.

Un factor determinante en el funcionamiento de la red es la semejanza entre las entradas de una misma clase. Tomando en cuenta que en el segundo grupo de imágenes la diferencia entre una y otra imagen, de la misma figura, es mayor que la que se presenta en el primer grupo. Podemos ver que la sensibilidad de la segunda versión es baja. Porque para el primer grupo de imágenes la red no logra diferenciar una imagen de otra, clasificándolas todas en la misma clase. Mientras que para el segundo grupo si existe una separación de las imágenes en diferentes clases, aunque estas no sean las correctas. Lo que implica que la versión dos necesita una variación grande en sus entradas para lograr diferenciar una imagen de otra.

Por otro lado, al igual que sucedía con la red ART1, si las diferencias de entradas de una misma clase son muy grandes, nunca se conseguirá una clasificación correcta del cien por ciento de todas las figura. Como sucede con las imágenes del segundo grupo, donde nunca se logra clasificar todas las imágenes correctamente. Mientras que para el primer grupo, como las imágenes varían menos, sí se logra en cien por ciento de aciertos.

Con la versión uno, nunca se logra la separación de clases, en ninguno de los dos grupos, por lo que se puede decir que su funcionamiento no es el adecuado.

Por todo lo anterior, se tomará como referencia los resultados arrojados únicamente por la versión tres de la red. Ya que son los únicos que presentan resultados correctos en la clasificación.

Comparando los resultados de la versión 3 de la red ART2, con los resultados de la ART1, vemos que ambas redes obtienen clasificaciones correctas para valores de ρ entre 0.90 y 0.99. Pero mientas que en la red ART1, se utiliza varias clases para valores de ρ menores a 0.90, en la ART2 se utiliza sólo una clase para estos mismos valores ρ.

Como punto final, cabe hacer notar que la red ART funciona mejor cuando su parámetro de semejanza se acerca a 1, pero sin tomar dicho valor. Además de que entre menor sean las diferencias entre imágenes de la misma clase, mejores resultados se obtienen.

5.3 Resultados de la red Backpropagation.

Como el objetivo del presente trabajo es realizar una comparación entre la red ART2 y la red backpropagation utilizadas en el reconocimiento de patrones dentro de imágenes digitales, es necesario procesar las mismas imágenes y tipos de entradas con la red backpropagation conservando las condiciones lo más parecidas posibles.

Para empezar, se procesarán las mismas imágenes del primer grupo de imágenes de figuras geométricas, ya que como vimos en la red ART2, en este grupo si se logra una efectividad del cien por cien. Los resultados los utilizaremos para ver algunas de las principales características de la red backpropagation.

Como vimos en el capitulo tres, la red backpropagación no utiliza tantos parámetros como la red ART2. Sus únicas variantes son: el número de neuronas ocultas, la tasa de aprendizaje utilizada por la red y el tipo de aprendizaje utilizado. De acuerdo con lo anterior se procesó el mismo grupo de imágenes cambiando alguno de estos elementos para ver como varía el funcionamiento de la red al modificar estos.

Como se explicó anteriormente, la red backpropagation tiene dos fases, una de aprendizaje y otra de funcionamiento, y sus resultados se evalúan de forma diferente.En la fase de aprendizaje nos interesa cuantos ciclos se tarda la red en aprender un conjunto de entrada y el número total de aciertos que se logra en dicho conjunto. Mientras que en la etapa de funcionamiento lo único que importa son los aciertos logrados al clasificar las diferentes imágenes.

Esta red utiliza dos conjunto de imágenes, uno de aprendizaje y otro de prueba o funcionamiento. Por lo que fue es necesario dividir el grupo de imágenes a utilizar en dos subgrupos, el primero contiene las tres primeras imágenes de cada figura y servirá para el

aprendizaje de la red y el segundo subgrupo es en si todo el grupo y sirve para la fase de funcionamiento. Se toman todas las imágenes para comprobar que la red realmente aprendió las imágenes utilizadas durante el aprendizaje y para probar como funciona ante imágenes nuevas.

Para la fase de entrenamiento es necesario determinar la salida de la red para cada entrada, así como el orden en que las imágenes serán procesadas, quedando de la siguiente forma:

Imagen	salida	Imagen	salida
Circulo1.bmp	1000	Rectangulo2.bmp	0010
Cuadro1.bmp	0100	Triangulo2.bmp	0001
Rectangulo1.bmp	0010	Circulo3.bmp	1000
Triangulo1.bmp	0001	Cuadro3.bmp	0100
Circulo2.bmp	1000	Rectangulo3.bmp	0010
Cuadro2.bmp	0100	Triangulo3.bmp	0001

Tabla 5.1

Tabla de entradas y salidas de la red Backpropagation.

Así se define la entrada de la red como los valores de los pixeles de la imagen original, lo cual nos da un total de 400 entradas y 4 salidas. Representando cada neurona de salida como un dígito.

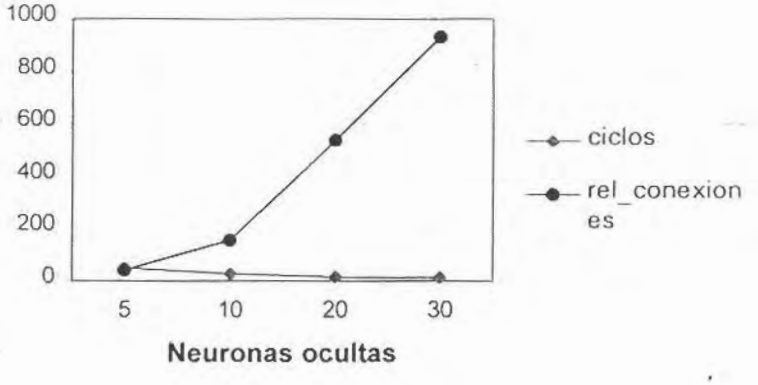
Primero se verá el funcionamiento variando en número de neuronas ocultas y el tipo de aprendizaje utilizado. Para el aprendizaje "ON LINE" se obtuvieron los siguientes valores. Donde se utilizaron una tasa de aprendizaje de 0.5, y un error mínimo de 0.02:

Neurona Oculta	ciclos	aciertos	Conecciones	Relación Conecciones/ciclos
5	48	12	2020	42,08333333
10	26	12	4040	155,3846154
20	15	12	8080	538,6666667
30	13	12	12120	932,3076923
50	10	12	20200	2020
70	8	12	28280	3535
90	7	12	36360	5194,285714
110	5	12	44440	8888
130	5	12	52520	10504
150	5	12	60600	12120
200	5	12	80800	16160

Tabla 5.2.

La primera columna indica el número de neuronas usadas, la segunda los ciclos hechos durante la fase de entrenamiento, la tercera los aciertos obtenidos del grupo de datos de entrenamiento durante dicha fase, la cuarta las conecciones utilizadas por la red en cada caso, que son el número de neuronas de entrada por ocultas, más número de neuronas ocultas por las de salida, y la ultima columna indica el resultado de dividir las conecciones de la red entre los ciclos ocupados

Como se puede ver, al aumentar el número de neuronas de la capa oculta disminuye el número de ciclos utilizados por la red para el aprendizaje. Pero también aumenta el número de interconecciones que son necesarias entre las neuronas de la red, por tanto, la relación de conecciones entre ciclos aumenta al incrementar las neuronas ocultas. Si graficamos estos resultados obtenemos:



Gráfica 5.6.

Muestra la tendencia de los ciclos ocupados por la red en la fase de entrenamiento y la relación del número de conecciones de la red entre el número de ciclos ocupados.

- Donde podemos observar que el número de interconecciones o pesos de la red aumenta en forma exponencial rápidamente conforme aumenta el número de neuronas ocultas. Al mismo tiempo el número de ciclos utilizados en el aprendizaje decrece, pero en forma más lenta. Lo anterior nos lleva a que la reducción de cada ciclo represente el aumento de miles de conecciones en la red. Este aumento es más grande conforme el número de ciclos se hace más pequeño. Al observar la gráfica vemos que cuando el número de neuronas ocultas es cinco se logra un buen equilibrio entre el número de conecciones de la red y el número de ciclos utilizados. Sí tomamos un número de neuronas mayor, los ciclos se reducen muy poco mientras que el número de pesos aumenta muchísimo, lo que consittuye un mayor gasto de recursos por parte de la red. En dicho caso, se utilizaría más memoria y tiempo de procesamiento. Debemos tener presente que esto es sólo valido para este ejemplo, ya que sí se cambia el tipo de entradas o su número, el número de neuronas ocultas óptimo variaría.

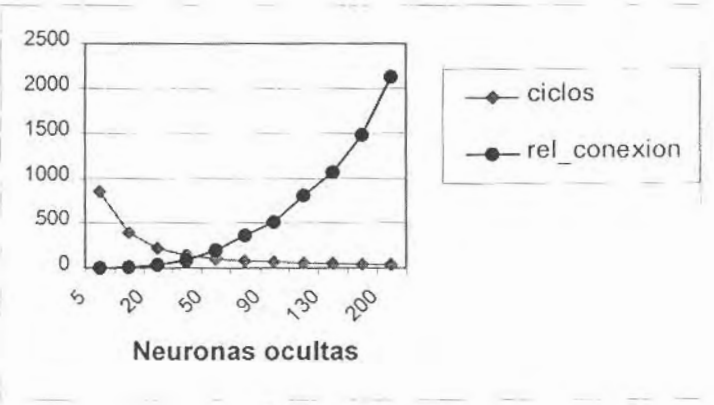
Utilizando el mismo grupo de imágenes con el segundo aprendizaje "OFF LINE", obtenemos los resultados de la tabla 5.3:

oculta	ciclos	aciertos	conecciones	Relación entre conecciones / ciclos
5	848	12	2020	2,38207547
10	387	12	4040	10,4392765
20	220	12	8080	36,7272727
30	141	12	12120	85,9574468
50	101	12	20200	200
70	79	12	28280	357,974684
90	71	12	36360	512,112676
110	55	12	44440	808
130	49	12	52520	1071,83673
150	41	12	60600	1478,04878
200	38	12	80800	2126,31579

Tabla 5.3.

La primera columna indica el número de neuronas usadas, la segunda los ciclos hechos durante la fase de entrenamiento, la tercera los aciertos obtenidos del grupo de datos de entrenamiento durante dicha fase, la cuarta las conecciones utilizadas por la red en cada caso, que son el número de neuronas de entrada por ocultas, más número de neuronas ocultas por las de salida, y la ultima columna indica el resultado de dividir las conecciones de la red entre los ciclos ocupados, la tasa de aprendizaje es igual a 0.5 y el error mínimo de 0.02, la red se detiene si llega los 1000 ciclos..

En los cuales, existe un gran aumento en el número de ciclos utilizados para el aprendizaje, aunque el comportamiento es muy similar al anterior. Ya que mientras se aumenta el número de neuronas disminuye el número de ciclos utilizados, pero debido a que el número de ciclos utilizados es mucho mayor que con el aprendizaje ONLINE, la relación de ciclos entre conecciones crece más lentamente.



Gráfica 5.7
Muestra la tendencia de los ciclos ocupados por la red en la fase de entrenamiento y la relación del número de conecciones de la red entre el número de ciclos ocupados.

Al revisar la gráfica veremos que su comportamiento es muy similar al presentado por la gráfica del aprendizaje "ON LINE", pero menos pronunciado. Lo que nos permite ver un mayor rango del número de neuronas ocultas. En este caso el equilibrio entre el número de ciclos y conecciones utilizadas por la red existe cuando se utiliza 50 neuronas ocultas.

Podemos concluir entonces, que con el aprendizaje ONLINE se utilizan un menor número de ciclos que con el aprendizaje OFFLINE, al tener el mismo número de neuronas ocultas. Esta diferencia es bastante significativa ya que de un óptimo de 5 neuronas ocultas se pasa a uno de 50, lo cual implica un aumento del 10 veces el número de conecciones en la red. También es necesario tener presente que los pesos de esta red se inician al azar, dentro de un rango dado. Por lo cual al repetir el proceso podemos encontrar una variación en el número de ciclos utilizados para cada aprendizaje. Dicha variación es muy grande cuando se utilizan muchos ciclos. Así si se utilizan aproximadamente 300 ciclos para un aprendizaje, la diferencia del número de ciclos usados de una prueba a otra prueba puede variar por cien o más ciclos. En cambio sí sólo se utilizan 50 ciclos las variaciones serán de 5 o 10 ciclos. En el caso del aprendizaje ONLINE, donde se utilizan 9 a 5 ciclos, la variación es de 1 o 2 ciclos de una prueba a otra.

En cuanto el desempeño de la red en la fase de funcionamiento, se probó con el conjunto completo de imágenes del grupo uno, y se encontró que se logra la clasificación correcta de todas las figuras, para todos los distintos valores del número de neuronas ocultas. Esto era de esperarse, ya que en ninguno de los casos anteriores la red llegó a los 1000 ciclos. El cual es el límite de ciclos en los cuales la red considera que todavía hay solución. Siempre se obtuvo los 12 aciertos durante el aprendizaje, lo que indica que la red logra separar correctamente las distintas clases de figuras. Como ya vimos en el caso de la red ART2, las figuras del grupo uno no varían demasiado entre ellas, por lo que la red Backpropagation puede clasificar correctamente aun las imágenes que no se le habían presentado antes, obteniendo así el cien por ciento de aciertos.

Ahora veremos el funcionamiento de esta red ante el grupo dos de figuras geométricas, para ver que tanto afecta la variación de las imágenes en el funcionamiento de la red.

Para esto se utilizó la misma lista de nombres de imágenes elaborada para el ejemplo anterior, y los mismos valores para el número de neuronas de la capa oculta, a excepción del 200. También se utilizaron los dos diferentes tipos de aprendizaje.

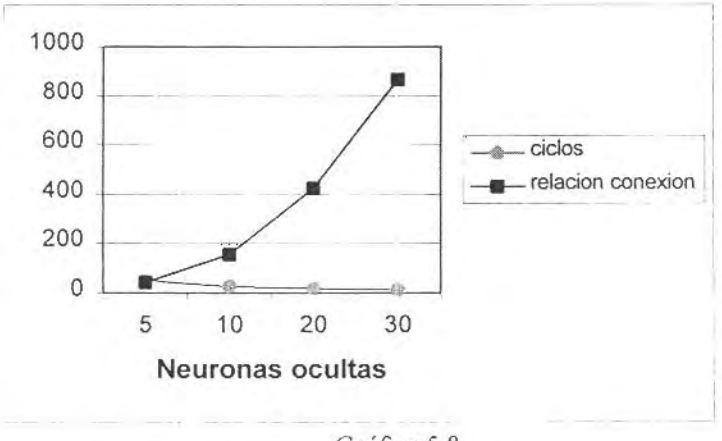
Para el aprendizaje ONLINE obtuvimos los siguientes resultados:

Neuronas ocultas	Ciclos	Aciertos	Conecciones	Relación Con/ciclos
5	48	12	2020	42,083333
10	26	12	4040	155,38462
20	19	12	8080	425,26316
30	14	12	12120	865,71429
50	11	12	20200	1836,3636
70	9	12	28280	3142,2222
90	8	12	36360	4545
110	6	12	44440	7406,6667
130	5	12	52520	10504
150	6	12	60600	10100

Tabla 5.4.

La primera columna indica el número de neuronas usadas, la segunda los ciclos hechos durante la fase de entrenamiento, la tercera los aciertos obtenidos del grupo de datos de entrenamiento durante dicha fase, la cuarta las conecciones utilizadas por la red en cada caso, que son el número de neuronas de entrada por ocultas, más número de neuronas ocultas por las de salida, y la ultima columna indica el resultado de dividir las conecciones de la red entre los ciclos ocupados.

Los cuales se parecen mucho a los obtenidos anteriormente al utilizar el mismo tipo de aprendizaje y diferente grupo de imágenes. Al graficar, vemos que el número de neuronas ocultas con mejor rendimiento, porque no realiza demasiados ciclos y tampoco la red necesita de muchas interconecciones, vuelve a quedar alrededor de 5 neuronas.



Gráfica 5.8
Muestra la tendencia de los ciclos ocupados por la red en la fase de entrenamiento y la relación del número de conecciones de la red entre el número de ciclos ocupados.

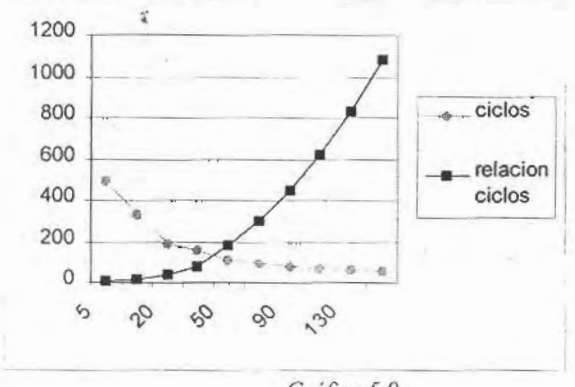
Para el aprendizaje OFFLINE, los resultados tienen la misma tendencia, aunque el número de ciclos aumenta considerablemente como se puede ver en la siguiente tabla:

Neuronas ocultas	ciclos	aciertos	Conecciones	Relación Con/ciclos
5	492	12	2020	4,1056911
10	331	12	4040	12,205438
20	188	12	8080	42,978723
30	157	12	12120	77,197452
50	111	12	20200	181,98198
70	94	12	28280	300,85106
90	82	12	36360	443,41463
110	72	12	44440	617,22222
130	63	12	52520	833,65079
150	56	12	60600	1082,1429

Gráfica 5.5

La primera columna indica el número de neuronas usadas, la segunda los ciclos hechos durante la fase de entrenamiento, la tercera los aciertos obtenidos del grupo de datos de entrenamiento durante dicha fase, la cuarta las conecciones utilizadas por la red en cada caso, que son el número de neuronas de entrada por ocultas, más número de neuronas ocultas por las de salida, y la ultima columna indica el resultado de dividir las conecciones de la red entre los ciclos ocupados

Al graficar los datos se reafirma, que para este caso, el número de neuronas ocultas con mejores resultados esta entre 30 y 50, como paso al procesar las imágenes del grupo uno.



Gráfica 5.9

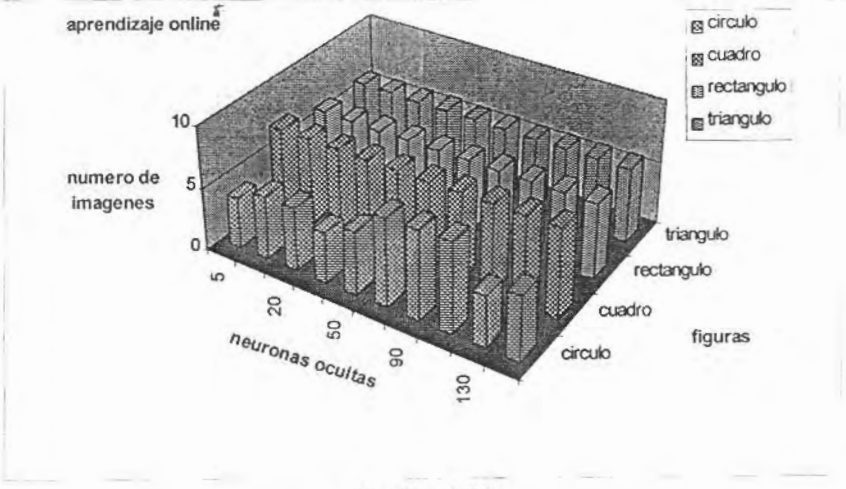
Muestra la tendencia de los ciclos ocupados por la red en la fase de entrenamiento y la relación del número de conecciones de la red entre el número de ciclos ocupados.

Cabe resaltar, que igual que en el caso anterior, los ciclos que se presentan para cada número de neuronas ocultas pueden variar si se repite el proceso de aprendizaje. Esto debido a que la inicialización de pesos se hace al azar. Lo que significa que siempre se empieza a buscar la solución desde un punto diferente en la superficie de error, y entre mayor sea el número de ciclos mayor puede ser la variación de dichos ciclos al repetir el aprendizaje.

En cuanto a la fase de funcionamiento los resultados si varían, obteniendo los valores de la tabla D.7 con el aprendizaje ONLINE

Como se puede ver la red no comete ningún error en las primeras tres imágenes de cada figura, que fueron las que se utilizaron para el entrenamiento. Sino que las equivocaciones aparecen en las demás imágenes, que no se habían presentado antes a la red. Pero estas equivocaciones no son muchas y aparecen en las imágenes del círculo 4,5 y 6, del triángulo4 y del rectangulo6, que son algunas de la imágenes en las cuales la red ART2 con su versión 3 y ρ igual a 0.95 o 0.97 tampoco pudo reconocerlas. Por lo que podemos decir que en estas imágenes existe una gran variación en sus pixeles respecto a las demás imágenes con la misma figura. Sin embargo se puede apreciar que los errores no varían mucho al utilizar un número de neuronas ocultas u otro, y la diferencia se puede deber en parte a la variación en la inicialización de pesos de la red. Ya que esta durante su aprendizaje siempre logra su equilibrio y el total de aciertos del conjunto de datos de aprendizaje. Otro detalle es que la red siempre le asignara una clase a la imagen, aun cuando el resultado este mal. A diferencia de la ART2, la cual si la imagen no resuena con un mínimo de similitud con alguna clase, indica que no puede clasificar la imagen.

Graficando los aciertos obtenidos por la red con el segundo grupo de imágenes y utilizando el aprendizaje ONLINE, obtuvimos lo siguiente:



Gráfica 5.10

Gráfica de aciertos obtenidos por la bakpropagation utilizando el aprendizaje ONLINE, el número máximo de imágenes de cada figura es siete.

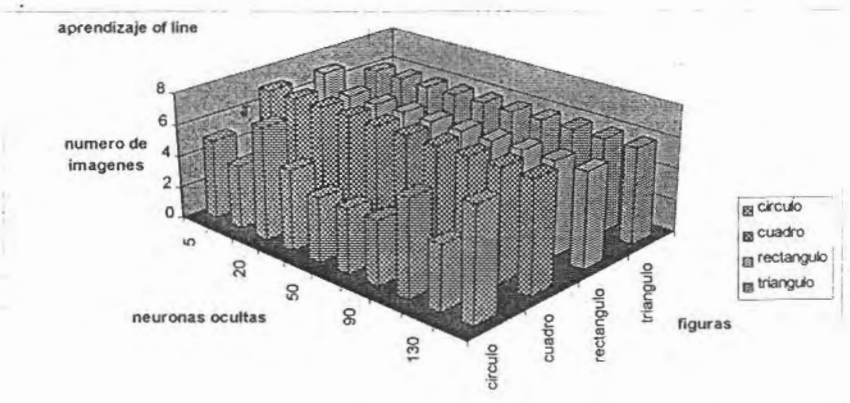
Vemos que es en la figura del círculo donde existe la mayor variación de aciertos. Es decir en esta figura la red se equivoca más seguido, pero también es donde podemos lograr el mayor número de aciertos cuando se utilizan alrededor de 90 neuronas ocultas, mientras que la figura del cuadro tiene siempre el cien por ciento de aciertos. Por otro lado el rectángulo y el triángulo nunca llegan a tener todas sus imágenes como aciertos, pero se conservan siempre con 6 imágenes bien

clasificadas, lo que indica que la red solo falla en una imagen de ambas figuras. Si observamos la tabla D7 veremos que las imagen que no se logran clasificar son el triangulo4 y el rectangulo6.

Aquí observamos una de las diferencias con la red ART2, donde siempre el círculo por ser la primera figura presentada a la red obtenía la mayoría de casos el cien por ciento de aciertos. Mientras que ahora todas las imágenes se tratan de igual forma, debido a que en el aprendizaje todas las imágenes se procesan en un ciclo que se repite varias veces.

Para el aprendizaje OFFLINE, en la fase de funcionamiento se obtuvieron resultados de la tabla D8. Al observar dicha tabla encontramos, que la red no se equivoca en las primeras tres imágenes de todas las figuras, y si llega a cometer algunos errores en las imágenes que no se le habían presentado antes, coincidiendo estas imágenes con las del aprendizaje anterior. Por lo cual podemos decir que estos errores se deben a la diferencia de estas imágenes con el resto de las imágenes de la misma figura.

Si graficamos los aciertos de la red obtenidos para este aprendizaje, encontramos:



Gráfica 5.11

Gráfica de aciertos obtenidos por la bakpropagation utilizando el aprendizaje ONLINE, el número máximo de imágenes de cada figura es siete.

Donde podemos observar, que es en la figura del círculo donde se vuelve a notar un gran número de errores. Pero en este caso se logra un mayor acierto cuando el número de neuronas ocultas es 20, 110 y 150. En tanto las demás figuras se mantienen igual, el cuadro con cerro errores, y el rectángulo y el triángulo con un error en cada ocasión. Excepto cuando se utilizan 5 neuronas ocultas, entonces la red logra clasificar bien las siete figuras del rectángulo. Si revisamos la tabla veremos que las imágenes que no se clasifican bien, vuelven a ser el triangulo4 y el rectangulo6, coincidiendo con los resultados arrojados por la red con el aprendizaje ONLINE.

En resumen, podemos decir que al usar el aprendizaje ONLINE obtenemos una gran reducción del número de ciclos utilizados por la red, en comparación con los ciclos utilizados en el aprendizaje OFFLINE, todo esto en la fase de entrenamiento. Mientras que en la fase de funcionamiento no existe una gran diferencia entre los resultados arrojados al utilizar un aprendizaje u otro. Si bien la clasificación de las imágenes con círculo varía de un caso a otro esta variación no es muy marcada. Sólo cambia los números de neuronas con los que se logra los mayores aciertos, pero estos números no siguen un patrón, por lo cual no podemos poner una norma a seguir o designar un número de neuronas que nos de los mejores resultados en la etapa de funcionamiento. Es bueno recalcar que todos los resultados anteriores son válidos para el caso específico de 400 entradas con los valores de los pixeles reales de las imágenes de los grupos 1 y 2, y utilizando cuatro neuronas de salidas y las salidas definidas para este caso.

Comparando los resultados de la red Backpropagation con los de la red ART2, para el caso de estos dos ejemplos podemos obtener las siguientes conclusiones:

- La red ART2 logra su mejor funcionamiento con el uso de la versión 3 presentada en este trabajo, y con valores de ρ iguales a 0.95 y 0.97.
- La red bakpropagation logra su mejor desempeño en la fase de aprendizaje cuando el número de neuronas ocultas oscila entre 30 y 50 para el aprendizaje OFFLINE y 5 para el aprendizaje ONLINE.

Los siguientes puntos son tomando el mejor desempeño de cada red.

- Ambas redes logran clasificar correctamente todo el grupo de imágenes del grupo uno.
- Ambas redes no clasifican correctamente las mismas figuras del grupo dos.
- La red ART2, no asigna clases equivocadas a una imagen, sino que no la clasifica cuando no encuentra un prototipo que resuene con ella.
- La red backpropagation le asigna una clase a todas las imágenes, aún cuando dicha clase esté equivocada.
- La red backpropagation utiliza de 3 a 6 veces más conecciones que la red ART2.
- Con el grupo uno de imágenes la red backpropagation logra siempre el cien por ciento de aciertos sin importar el número de neuronas ocultas. Mientras que la red ART2 solo lo logra con valores de ρ iguales a 0.95 y 0.97.
- En algunos casos como el de 5 y 10 neuronas ocultas con aprendizaje ONLINE, la red backpropagación logra menos errores para el grupo dos de imágenes (2 errores), que la red ART2 con ρ igual a 0.95 (5 errores).

Éstas, más las diferencias inherentes a cada una de las estructuras nos permiten ver que para casos en que la base de datos del problema no este bien definida y sea cambiante es mejor utilizar la red ART2. Ya que si encontramos el valor de ρ más apropiado para nuestras entradas, esta red funcionara bastante bien logrando un gran porcentaje de aciertos en sus clasificaciones, al mismo tiempo que permite agregar nuevas imágenes a la base de datos. Pero hay que tener cuidado con las entradas que le apliquemos, ya que podemos degenerar los prototipos almacenados en la red perdiendo nuestra información. En cambio, si nuestro problema esta bien definido y tenemos una base de imágenes bien determinada para aprender, la cual no sufrirá variaciones con el tiempo. Es mejor utilizar la red backpropagation ya que disminuiremos un poco el margen de error al momento de clasificar las imágenes, y la información de la red no corre el riesgo de ser alterada. Pero sí queremos incluir una nueva clase en nuestro sistema será necesario repetir el proceso de aprendizaje completo.

5.4 Resultados sobre imágenes de letras.

Hasta ahora se han presentado como funcionan las dos redes cuando las entradas son los valores reales de los pixeles de una imagen. Aunque las redes obtienen resultados satisfactorios en la clasificación de imágenes muy parecidas, como las del grupo uno. Las redes empiezan a fallar si las imágenes sufren cambios, como las del grupo dos. Para disminuir estas fallas una opción es utilizar valores que representen las características principales de las clases de las imágenes en lugar de los valores de sus pixeles. Para lograr esto utilizaremos el procesamiento digital de imágenes, que como ya vimos permite extraer de una imagen alguna característica especial.

El tipo de característica de una imagen que seleccionamos como entrada dependerá del tipo de imágenes que se quiera procesar. En este trabajo además de los dos primeros grupos de imágenes de figuras geométricas, se procesaron un grupo de imágenes con caracteres alfabéticos y otro con rostros humanos. Por lo que se eligieron diferentes características para cada uno de los dos últimos grupos. Para uno momentos y el otro la intensidad del espectro de la transformada de Fourier.

En esta sección se presentan los resultados obtenidos al procesar las imágenes de los caracteres alfabéticos que forman el grupo tres. Primero veremos los resultados al utilizar la red ART2 y después los de usar la red Backpropagation. Para este grupo de imágenes se escogió como característica de extracción los momentos de la imagen, ya que como se vio en el capitulo dos, los momentos de una imagen describen la forma de un objeto dentro de una imagen. Las letras se diferencian precisamente por su forma, si revisamos las formulas de los momentos (ec. 2.2.10 –

2.2.16) veremos que dependen de una potencia del valor de cada pixel de la imagen, por lo cual se opto que los fondos de estas imágenes fueran negros, pixel = 0, para que contribuyeran lo menos posible al valor del momento. Mientras que el objeto en la imagen, en este caso la letra, es de color blanco (pixel = 255), para ejercer la máxima influencia en los valores finales de los momentos.

Como entradas de las redes se utilizaron los momentos invariantes, puesto que no se ven afectados por la rotación, traslación o escala de los objetos dentro de una imagen. Esto nos permitirá reconocer diferentes tamaños de letras, Así como si las letras que hayan sufrido algún cambio de posición u orientación dentro de la imagen.

Al observar los resultados obtenidos de calcular los momentos de las imágenes de este grupo observamos que los valores del primer momento son apenas un poco mayores que una o dos unidades, el segundo y tercer momento tiene valores en el orden de las milésimas, el cuarto momento en el orden de micras, el quinto en nanos, el sexto de nuevo en el orden de micras y el séptimo con factor de 10^{-10}.Por lo que si se procesan al mismo tiempo los siete valores, el primer momento es tan grande respecto a los otros que será el único valor significativo, es decir los valores de los otros seis momentos se tornarían insignificantes. Lo anterior ocasionaría la perdida de la información que contienen. Por todo esto se decidió multiplicar a los momentos por diferentes factores para compensar su diferencia, quedando de la siguiente forma:

Momento	factor
1	1
2	1000
3	1000
4	100000
5	10^9
6	10^6
7	10^{10}

Tabla 5.6

La primera columna indica el número del momento invariante, la segunda indica el valor por el cual se multiplica el momento para igualar el orden de los momentos.

El resultado de las multiplicaciones anteriores serán las entradas de las redes, lo cual nos asegura que cada momento sea tomado en cuenta.

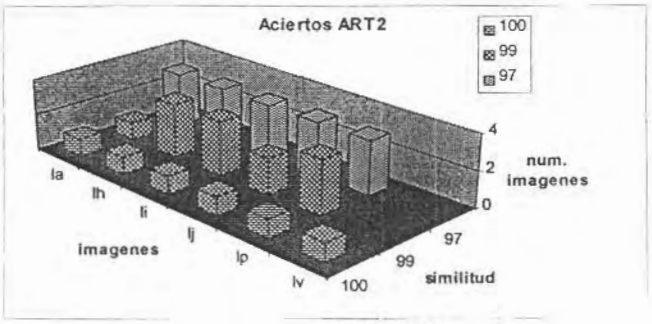
El grupo tres consta de 42 imágenes, 7 imágenes son de la misma letra, las letras usadas son la A, H, I J, P y V, las tres primeras imágenes de cada letra varían solo en el tamaño de la letra, la cuarta y quinta en la posición de la letra y la sexta y séptima en la rotación de la letra dentro de la imagen.

Primero se hizo una prueba utilizando únicamente las tres primeras figuras de cada imagen, es decir, aquellas donde solo varía el tamaño de la letra. Para la red ART2 se realizó la prueba con valores de ρ igual a 0.95, 0.97 , 0.99 y 1.00. Se tomó θ igual a cero, ya que los valores de los momentos pueden llegar a ser pequeños, con esto se asegura minimizar las perdidas de información. A las demás variables se les asignó los siguientes valores A= 10, B = 8, C= 0.10 y D = 0.90. 7 neuronas de entrada y 6 de salida, obteniéndose los resultados de la tabla D9.

En la tabla D9 se puede ver que la red sí llega a confundir la imágenes de distintas clases, clasificándolas mal, por lo que ya no es valido lo que pasaba con el grupo de imágenes geométricas, donde la red sí lograba separar claramente las imágenes en sus diferentes clases.

Si observamos nunca se logra el cien por ciento de aciertos ni aun con 0.99 ya que la red no logra clasificar correctamente la letra v que es la correspondiente a la clase seis. Sólo al utilizar ρ igual a 1, sí se logra la separación correcta de las primeras 6 imágenes, cada una asignada a una clase distinta, pero las demás imágenes no se clasifican por no ser idénticas a la primera.

Si graficamos los resultados anteriores obtenemos:



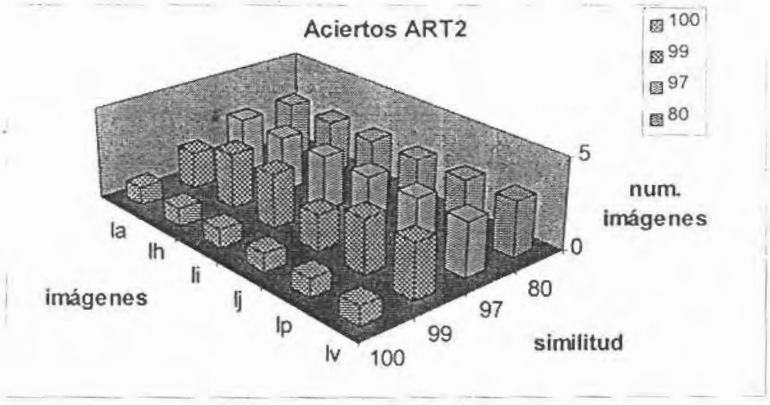
Gráfica 5.12

Gráfica de los aciertos obtenidos por cada tipo de imagen utilizando la red ART2, y con diferentes valores de ρ.

Donde observamos con mayor claridad que las imágenes donde más falla la red son las de la letra v. Mientras que para las demás letras se obtiene el total de aciertos cuando ρ es igual a 0.97, es decir con este valor la red trabaja mejor. Pero si después de procesar las imágenes con ρ igual a 1, se vuelven a procesar las mimas imágenes pero bajando el parámetro de similitud, se obtienen los resultados de la tabla D10.

En dicha tabla, podemos observar que la red ya logra un cien por ciento de efectividad, y es capaz de reconocer todas la imágenes en sus clases correctas cuando ρ es más bajo que 0.99. Esto se debe a que los prototipos de las imágenes ya están almacenados en la red, y al entrar cualquier imagen ésta resuena con su prototipo en la primera interacción, Si el parámetro de similitud no es muy alto, se toleran algunas diferencias entre la imagen entrante y el prototipo, por lo cual la red le asigna la misma clase.

Si graficamos ahora estos resultados encontraremos:



Gráfica 5.13

Gráfica de los aciertos obtenidos por cada tipo de imagen utilizando la red ART2, y con diferentes valores de ρ, tomando la primera iteración con ρ igual a 1, y los demás sin reinicializar valores de los pesos de la red.

Aquí se comprueba que con los valores de ρ igual a 0.97 y 0.95 se logra clasificar todas la imágenes de todas la figuras correctamente. A diferencia de lo que pasaba en la gráfica anterior, las imágenes de las letras V si son clasificadas correctamente. Al mismo tiempo se logra una mejora en los resultados de ρ igual a 0.99 ya que solamente falla en algunas imágenes de las letras A y J.

Por otro lado, para procesar las mismas imágenes con la red backpropagation necesitamos dividir nuestro conjunto de imágenes en uno de aprendizaje y otro de prueba. El primero estará constituido por las dos primeras imágenes de cada letra y el segundo por las tres primeras imágenes

de cada letra, es decir, por el mismo grupo utilizado para la red ART2. Para la fase de entrenamiento se asigna a cada imagen una clase quedando de la siguiente forma:

Imagen	salida	Imagen	Salida
la1	100000	la2	100000
lh1	010000	lh2	010000
li1	001000	Li2	001000
lj1	000100	lj2	000100
lp1	000010	lp2	000010
lv1	000001	lv2	000001

Tabla 5.7

La primera y tercer columna indican el nombre de la imagen, y la segunda y cuarta la clase asignada para el aprendizaje en la red Backpropagation.

Para el aprendizaje se hicieron varias pruebas con los dos tipos de aprendizaje. Con el fin de ver con cual aprendizaje funciona mejor con este tipo de entrada, se utilizaron los valores de 5, 7, 8 ,10, y 12 para las neuronas ocultas, utilizando una tasa de aprendizaje de 0.5 y un error mínimo de 0.02. Durante el período de aprendizaje se obtuvieron los siguiente resultados:

	Aprendizaje ONLINE		Aprendizaje OFFLINE	
oculta	ciclos	Aciertos	ciclos	aciertos
5	1000	8	1000	0
7	1000	8	1000	6
8	1000	8	1000	7
10	1000	8	1000	7
12	1000	8	1000	7

Tabla 5.8

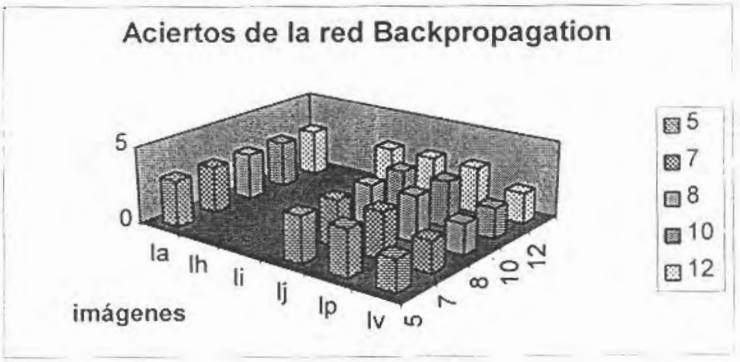
Números de ciclos y aciertos utilizados por la red Backpropagation durante la fase de entrenamiento de dicha red al variar el número de neuronas oculta. Valores para el grupo compuesto por las dos primeras imágenes de cada letra.

De donde podemos ver que en ningún caso se logró el total de aciertos que serían 14, ya que este es el número de imágenes utilizadas para el aprendizaje. Sino que la red llega primero a los mil ciclos y termina el aprendizaje con un número bajo de aciertos. Por otro lado, para el aprendizaje ONLINE, no importa el número de neuronas ocultas, ya que siempre se obtiene el mismo número de aciertos. Mientras que para el aprendizaje OFFLINE, al variar el número de neuronas varía el

número de aciertos, pero llega a un valor en el que se mantiene estático, 8, por lo que ya no es conveniente seguir aumentando el número de neuronas para este aprendizaje.

Estos resultados pueden mejorar en gran medida si se altera la topología de la red aumentando una capa oculta de neuronas y variando la combinación de neuronas en las capas ocultas, o bien modificando el algoritmo de propagación de error para acelerar y mejorar la búsqueda del mínimo absoluto de la función de error. Sin embargo, este proceso es de prueba de error y cambia para cada conjunto de aprendizaje, por lo cual ahora solo utilizaremos esta versión sencilla de esta red.

Para la fase de funcionamiento se probó la red para cada caso de entrenamiento. Con el fin de ver si mejora dicho funcionamiento con un número dado de neuronas, y para el aprendizaje ONLINE se obtuvieron los resultados de la tabla D11, en donde podemos ver que la red nunca logra el cien por ciento de efectividad. Incluso llega a equivocarse en imágenes utilizadas dentro de la fase de aprendizaje. Esto se debe a que nunca llega a tener todos los aciertos en dicha fase, lo que ocasiona que la red no pueda clasificar correctamente estas figuras. Sin embargo, la red si logra reconocer algunas de la imágenes de las letras A, J, y P, que no se le habían presentado antes. Si graficamos los aciertos de estos resultados obtenemos:



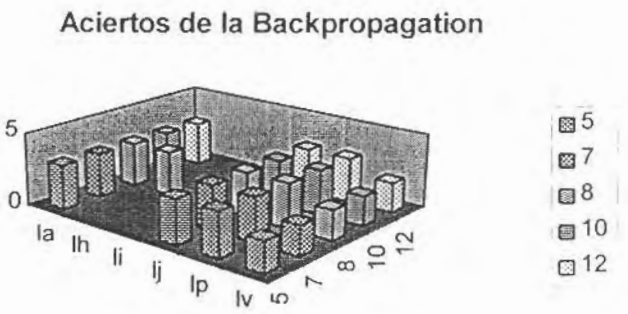
Gráfica 5.14

Gráfica de los aciertos obtenidos por cada tipo de imagen utilizando la red Backpropagation, y con diferentes valores para el número de neuronas ocultas.

Donde vemos que la red falla en la imágenes de las letras H e I. Aunque para el caso de 12 neuronas ocultas si logra reconocer las imágenes de I, mientras que para las demás imágenes la red se comporta de manera muy similar de un valor a otro de neuronas ocultas.

Utilizando el aprendizaje OFFLINE, se obtuvieron los datos de la tabla D12. En donde vemos que la red tampoco logra un funcionamiento del cien por ciento. Sin embargo, si se logra clasificar correctamente algunas de las figuras que no se habían presentado antes a la red.

Si graficamos los aciertos obtendremos



Gráfica 5.15

Gráfica de los aciertos obtenidos por cada tipo de imagen utilizando la red Backpropagation, y con diferentes valores para el número de neuronas ocultas.

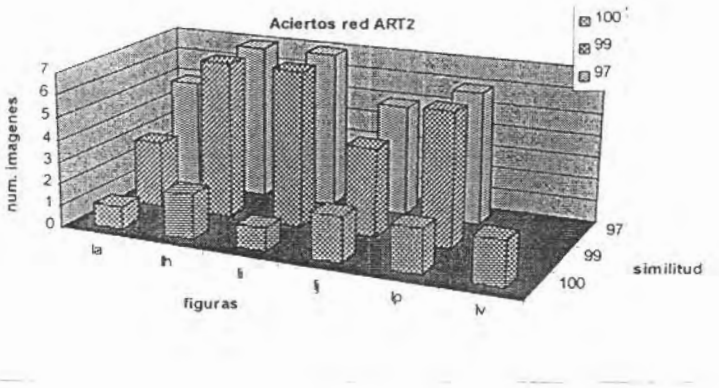
La cual se parece mucho a la gráfica que se obtuvo en el caso anterior. Volvemos a notar que donde más falla la red es en las figuras de la letra H, e I, a excepción del caso en que se utilizan 8 neuronas ocultas donde se logra la clasificación de todas las imágenes de la letra H. Además, el hecho de que las imágenes de la A, J y P obtengan en todos sus casos tres aciertos, indica que la red si está reconociendo las imágenes que no se utilizaron en el entrenamiento y que tienen un cambio en el tamaño de la imagen.

Una vez que hemos visto como se comportan las redes ante imágenes que sufren solo una variación en el tamaño del objeto, es bueno ver como se comportan si la variación se da en la posición del objeto y en su orientación, aumentando 4 imágenes a este grupo de prueba.

Primero veremos los resultados obtenidos con la red ART2, con la cual se utilizaron valores de ρ igual a 0.95, 0.97, 0.99 y 1.00, siete neuronas de entrada y 6 de salida. A las variables se les asigna los valores de A= 10, B= 8, C = 0.10 y D= 0.90 obteniendo los resultados de la tabla D13. Donde podemos observar que la red no logra identificar correctamente todas las letras, pero en el caso de la J y la P logra clasificar casi todas la imágenes en la clase correcta, es decir, no se ve afectada por la escala, traslación y rotación del objeto dentro de la imagen. Las únicas imágenes de

estas letras que no logra clasificar son las séptimas, que son aquellas donde la letra sufre una rotación de 90 grados.

Si graficamos los aciertos por imágenes de cada letra obtenemos:



Gráfica 5.16

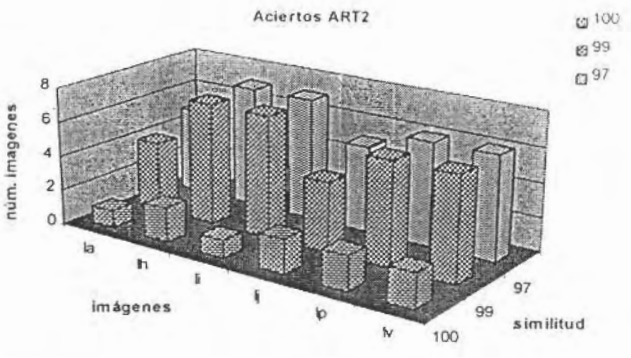
Gráfica de los aciertos obtenidos por cada tipo de imagen utilizando la red ART2, y con diferentes valores de ρ.

Donde es más visible que la red obtiene sus mejores resultados en la clasificación de las letras H e I. Mientras que con las imágenes de la letra V, no obtiene ningún acierto hasta que ρ es igual a 1.00. En las demás letras sus aciertos se deben principalmente a las primeras imágenes procesadas, donde solo varía el tamaño, y las fallas ocurren en las últimas imágenes, donde las letras aparecen rotadas

Al igual que se hizo con el primer subgrupo de letras, se hizo la prueba de procesar las imágenes con ρ igual a 1.00, para que la red separe correctamente cada clase, y después se vuelven procesar las imágenes bajando el valor de ρ con lo cual se obtuvieron los resultados de la tabla D14.

En dicha tabla podemos ver que el funcionamiento de la red mejora considerablemente, sobre todo para las primeras imágenes de las figuras, aunque en las séptimas imágenes todavía presenta algunas fallas. También es notorio, que aun cuando ρ es igual a 1, no sólo se clasifican las primeras imágenes de cada letra, sino que también algunas otras imágenes logran clasificarse. En dichas imágenes las letras son del mismo tamaño que en la primera imagen y solo sufren de una traslación respecto a la primera.

Si graficamos los aciertos respecto a cada letra tenemos:



Gráfica 5.17

Gráfica de los aciertos obtenidos por cada tipo de imagen utilizando la red ART2, y con diferentes valores de ρ, tomando la primera iteración con ρ igual a 1, y los demás sin reinicializar valores de los pesos de la red.

Donde podemos ver con mayor claridad que la red mejora sus resultados, sobre todo para las imágenes de la letra V. Aunque la red sólo logra tener todos los aciertos con las letras H e I, mientras que con las demás tiene un mínimo de tres aciertos. Al observar la gráfica vemos que son las primeras imágenes donde se llevan acabo los aciertos y en las séptimas donde ocurren la mayor cantidad de fallas.

Al utilizar todo el grupo de imágenes vemos que ya no se logra el cien por ciento de aciertos como cuando sólo se utilizan las tres primeras imágenes del grupo. Pero aun así vemos que la red es capaz de reconocer algunas imágenes que hayan sufrido una traslación o una rotación muy grande como son el caso de las letras H e I, las cuales reconoce aun rotadas por 90 grados.

En este trabajo se utilizó los valores de los momentos obtenidos directamente de la imagen. Pero también con base en estos momentos se puede hacer transformaciones a las imágenes para que se parezcan más entre una misma clase. Por ejemplo, con los dos primeros momentos se puede obtener el centro de masa de un objeto y moverlo hacia el centro de la imagen. Con esto se eliminarían en gran parte las diferencias que existen por traslación del objeto a través de la imagen. Existen otros datos que se pueden obtener de los momentos, sin embargo, como el presente trabajo solo nos interesa ver como actúan las redes ART2 y backpropagation aplicadas al reconocimiento de patrones, no nos enfocaremos a crear todo el método para el reconocimiento correcto de todas las letras.

Ahora veremos los resultados obtenidos al utilizar el mismo grupo de imágenes con la red backpropagation. Para esto necesitamos definir otra vez el conjunto de imágenes de entrenamiento, aunque se puede utilizar el mismo. Pero para que la red tenga una gama más amplia del tipo de imágenes que pueden ser clasificadas dentro de una clase, se optó por escoger las primeras cuatro imágenes de cada figura asignándoles la clase correspondiente a la letra que representa, aumentando así a 28 el número de imágenes en esta fase. También se hicieron varias pruebas variando los valores de las neuronas ocultas, y durante la fase de entrenamiento se encontraron los siguientes resultados:

Neurona ocultas	Aprendizaje ONLINE		Aprendizaje OFFLINE	
	ciclos	aciertos	ciclos	aciertos
5	1000	16	1000	4
7	1000	16	1000	2
8	1000	16	1000	12
10	1000	16	1000	15
12	1000	16	1000	15

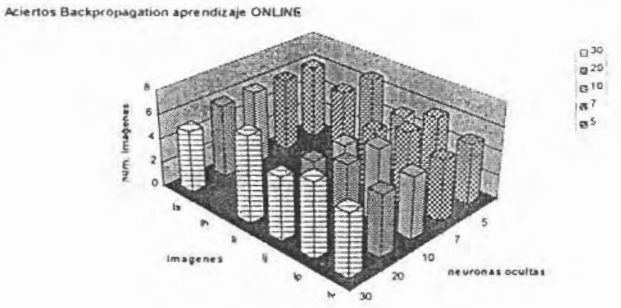
Tabla 5.9

Números de ciclos y aciertos utilizados por la red Backpropagation durante la fase de entrenamiento de dicha red al variar el número de neuronas ocultas. Valores para el grupo compuesto por las cuatro primeras imágenes de cada letra.

Donde se puede ver que otra vez la red no logra el total de aciertos en ningún caso, por lo que siempre se detiene a los 1000 ciclos. En el caso del aprendizaje ONLINE el número de aciertos se vuelve a mantener constante sin importar la variación de las neuronas ocultas. Mientras que para el aprendizaje OFFLINE, el número de aciertos se incrementa al aumentar el número de neuronas, pero después de 20 neuronas se mantiene constante, por lo que ya no tiene caso hacer más pruebas aumentando este número de neuronas.

Para la fase de funcionamiento se utilizó el grupo de imágenes tres completo. Obteniendo los resultados de la tabla D15 para el aprendizaje ONLINE. Donde observamos que cuando se utilizan 5, 7 y 30 neuronas ocultas, la red clasifica bien todas las imágenes que se utilizaron para el entrenamiento a excepción de las imágenes de la letra H. Mientras, en el grupo de imágenes que no había procesado con anterioridad los errores se presentan en todas la letras con diferente frecuencia y en diferente momento.

Graficando los aciertos obtenidos por cada tipo de letra tenemos:



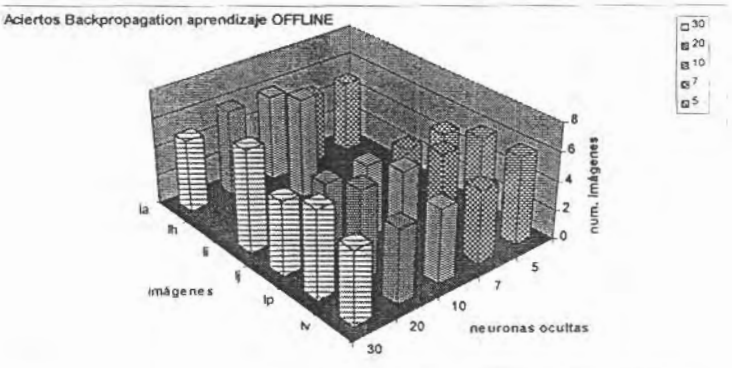
Gráfica 5.18

Gráfica de los aciertos obtenidos por cada tipo de imagen utilizando la red Backpropagation, y con diferentes valores para el número de neuronas ocultas

Donde podemos ver que ahora la mayoría de fallas se dan en las imágenes de la letra H, no importando el número de neuronas ocultas usadas. Siguiendo, en número de fallas, las imágenes de la letra I, aunque aquí sí varían los aciertos dependiendo del número de neuronas ocultas usadas en la red. Siendo los mejores valores cuando se utilizan 5, 7 y 30 neuronas ocultas. Los aciertos en las imágenes de otras letras casi no varían al cambiar el número de neuronas ocultas.

Para el aprendizaje OFFLINE se obtuvieron los resultados de la tabla D16, donde podemos observar que tampoco en ningún caso se logra la clasificación correcta de todas las imágenes. Pero ahora los resultados presentan una variación según va cambiando el número de neuronas ocultas, sin embargo al igual que ocurrió con el aprendizaje ONLINE, la mayoría de las fallas ocurren en las últimas imágenes de las figuras.

Graficando los aciertos por cada letra obtenemos:



Gráfica 5.19

Gráfica de los aciertos obtenidos por cada tipo de imagen utilizando la red Backpropagation, y con diferentes valores para el número de neuronas ocultas

Donde se vuelve a apreciar que es en la clasificación de las imágenes de las letras H e I donde se presentan el mayor número de fallas. Sin embargo, en este caso cuando el número de neuronas ocultas es igual a 10 si se logra la clasificación correcta de todas las imágenes de la letra H, y cuando el número de neuronas ocultas es igual a 30 se logra la clasificación correcta de todas la imágenes de la letra I. Pero en ninguno de los casos se presentan aciertos al mismo tiempo en las imágenes de estas dos letras, en cambio para el resto de imágenes los valores se conservan casi constantes sin depender del número de neuronas ocultas, y la letra que sufre mayor variación es la A.

Sí comparamos los resultados obtenidos por la red ART2 contra los obtenidos por la backpropagation, nos damos cuenta que la red ART2 logra un mejor funcionamiento. Puesto aunque no se haga la primera interacción con ρ igual a 1.00, se logran clasificar correctamente la mayoría de la imágenes a excepción de las de la letra V que es la última en presentarse. En cambio con la red backpropagation aparte de que también existen letras cuyas imágenes no se clasifican correctamente como la H o la I, ninguna de las demás letras logra su máximo de aciertos. Regresando a la red ART2, son precisamente la H y la I, las que logran este máximo, es decir la red puede reconocerlas aun rotadas o trasladadas. Por otro lado si en la red ART2 se procesan las imágenes por primera vez con ρ igual a 1, los resultados mejoran para la letra V. Por lo cual el número de aciertos de la red se incrementa y se logra tener aciertos para todas las letras al mismo tiempo, cosa que no ocurre con la red Backpropagation, donde siempre existe una letra que no tiene ningún acierto.

5.5 Resultados obtenidos con imágenes de rostros humanos.

El último grupo de imágenes, está compuesto por rostros humanos de diez diferentes personas, y de cada una de ellas hay tres imágenes, haciendo un total de 30 imágenes. Pero cinco personas del grupo usan lentes. Aprovechando esta característica se tomaron otras tres imágenes de cada una de ellas sin lentes. Dando un total de 45 imágenes para este grupo. Para facilitar el manejo de las imágenes, a cada persona se le asigno un número n, y se utilizo la letra a para la primera imagen de cada persona, la b para la segunda y la c para la tercera, formándose así tres grupos principales a, b, c, de acuerdo a las letras de las imágenes que contiene cada grupo. Existe un cuarto grupo compuesto con las imágenes que no tienen lentes, a las que se les agrego un '_1', para diferenciarlas.

Los rostros son imágenes muy complicadas, puesto que están compuestos de varias partes ojos, boca nariz, etc., las cuales tienen una forma estándar y varian solo un poco de una persona a otra. Además, de que como son volúmenes, sus características pueden variar desde el ángulo que se tome la imagen. Debido a esto se estableció que para reducir esta variación lo más posible, todas la imágenes se tomaron con el rostro de frente, sobre un fondo blanco y a una distancia similar, para evitar la variación del tamaño. Aun así entre una imagen y otra de la misma persona los pixeles cambian de valores y lugar de ubicación, por lo cual no basta con comparar dos imágenes pixel a pixel, sino que se le tiene que aplicar un procesamiento para extraer sus principales características.

Estas imágenes son de 256 por 256 pixeles, por lo cual no es muy recomendable utilizar los valores reales de la imagen como entradas de la red. Ya que tendríamos 65536 neuronas de entrada, y al multiplicarlas por las neuronas de salida, en el caso de la red ART2, o por las neuronas ocultas, en el caso de la red Backpropagation, obtendríamos un número muy alto para las conecciones necesarias para las redes. Lo que implica un mayor gasto de recursos tanto de memoria como de tiempo para el procesamiento de la información. Recordando lo que vimos en el capitulo dos, sabemos que el espectro de la transformada de Fourier tiene la ventaja de concentrar el máximo de información en el centro [González 1996], lo cual nos puede ayudar a reducir el número de entradas a la red sin perder mucha información de la imagen.

Al sacar la transformada de Fourier de una imagen, obtendremos un espectro en dos dimensiones, por tanto la mayor información de él se encuentra en el centro del cuadro que forma dicho espectro. Dependiendo del tamaño del cuadro que se tome en cuenta será la cantidad de información de la imagen que conservemos. Este es el principio del filtro ideal, donde se hacen cero los elementos de la transformada de Fourier que no estén incluidos en un círculo centrado respecto a dicha transformada. Al realizar la antitransformada recuperamos la imagen original con un menor o mayor degradamiento, dependiendo del tamaño del círculo tomado. Si el círculo es muy pequeño sólo recuperaremos una imagen muy borrosa con apenas unos rasgos de la imagen original, y si tomamos un círculo que comprenda casi toda la transformada de Fourier recuperaremos la imagen original con muy poca degradación. En este caso se optó por cambiar el círculo por un cuadro, para hacer más simple el cálculo de entradas de la red, así como la selección de valores de la TF. Haciendo diferentes pruebas se llegó a la determinación de que con un cuadro de 30 por 30 pixeles se logra una recuperación aceptable de la imagen, lo suficiente para diferenciar un rostro de otro.

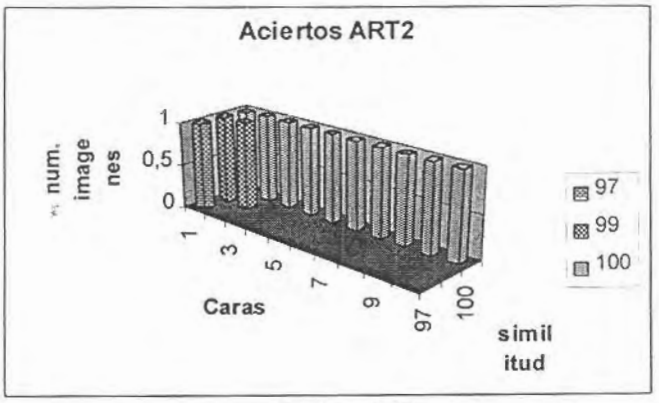
Primero se presentan los resultados obtenidos al utilizar la red ART2, para lo cual se asignaron los siguientes valores a las variables A= 900, B=800, C = 0.10 y D = 0.90, θ igual a

cero. Se obtuvó que para valores de ρ menores a 0.97 la red clasifica todas la imágenes como pertenecientes a la misma clase, 1, y es hasta que ρ toma un valor de 0.99, cuando la red empieza a distinguir dos clases obteniéndose los resultados de la tabla D17.

Observamos que la red revuelve las imágenes en dos clases, pero no logra separarlas adecuadamente. Como vimos antes, la red mejora su funcionamiento si la primera vez que procesa las imágenes se utiliza ρ igual a 1.00. Por tanto se repitió este procedimiento, obteniendo los resultados de la tabla D18.

En dicha tabla se puede observar que la red logra separar adecuadamente todas las caras, del grupo a, mientras que para los otros tres grupos no pudo clasificar las imágenes.

Si graficamos los resultados obtenidos para el grupo a, con los valores de ρ igual a 0.97, 0.99 y 1.00 obtenemos:



Gráfica 5.20
Números de ciclos y aciertos utilizados por la red ART2 al variar el parámetro ρ.

Donde es evidente la mejora de resultados, aunque solo sea del grupo a. Porque para los demás grupo con estos valores de ρ, se tiene que la red se equivoca de clase, o bien no clasifica la imagen.

Si después, se vuelven a procesar todas las imágenes utilizando ρ igual a 0.99 se encuentra que la red logra clasificar correctamente todas la imágenes de los grupos a, b, c, y el de sin lentes sin ningún error.

Para la red Backpropagation se creó un grupo de imágenes de aprendizaje, el cual contiene las imágenes de los grupos a y b, y a cada imagen se le asignó su clase correspondiente, para

conservar la uniformidad se utilizo una taza de aprendizaje de 0.5 y un error mínimo de 0.02. Se procesaron las imágenes con los dos tipos de aprendizaje obteniendo lo siguiente

Neurona ocultas	Aprendizaje ONLINE		Aprendizaje OFFLINE	
	ciclos	aciertos	ciclos	Aciertos
10	1000	4	1000	0
30	1000	11	1000	0
50	1000	13	1000	4

Tabla 5.10

Números de ciclos y aciertos utilizados por la red Backpropagation durante la fase de entrenamiento de dicha red al variar el número de neuronas ocultas

Donde podemos ver que la red no logra un aprendizaje total del grupo de imágenes de entrenamiento, y termina el proceso al llegar a los 1000 ciclos. En este caso vemos que se obtienen mejores resultados con el aprendizaje ONLINE, donde el número de aciertos es mucho mayor que en el aprendizaje OFFLINE. Se hicieron varias pruebas con cada aprendizaje y se encontró que el número de aciertos varía cada vez que se ejecuta el aprendizaje ONLINE. Por ejemplo en el caso de 30 neuronas ocultas se obtuvieron valores de 8, 11 y 16. Sin embargo no hay que perder de vista que ésta es la estructura más simple de la red Backpropagation, y que sí variamos el número de capas ocultas, así como las neuronas que hay en ellas, los resultados pueden mejorar mucho. También es muy posible que se este cayendo en mínimos locales lo cual se puede solucionar modificando el algoritmo de aprendizaje insertando algún mecanismo que detecten cuando la red ya no baja el error pero éste todavía sea muy grande, que es lo que se llama un mínimo local, entonces obliga a la red a dar un brinco fuera de esa región para seguir buscando otro mínimo más pequeño.

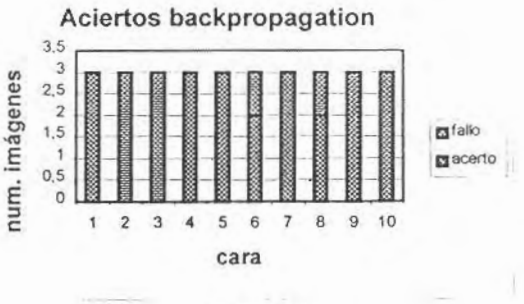
Después de cada aprendizaje se probó el funcionamiento de la red ante las imágenes del grupo a, que es uno de los que componen el grupo de entrenamiento. Esto nos da el parámetro de que tan bien trabaja la red. Se obtuvieron los resultados de la tabla D19 para el aprendizaje ONLINE:

Donde podemos ver que se obtuvieron los mejores resultados cuando el número de neuronas es de 30. Pero como se dijo antes el número de aciertos en la fase de aprendizaje varia si se repite el proceso. También se encontró que los resultados de la fase de funcionamiento varían dependiendo del número de aciertos de la fase de aprendizaje. A menor número de aciertos en el aprendizaje, mayor número de errores en el funcionamiento.

Para el aprendizaje OFFLINE se encontró que el mejor resultado se da cuando el número de neurona es igual a 50, por tanto se procesaron todas las imágenes utilizando 30 neuronas ocultas

para el aprendizaje ONLINE, y 50 para el OFFLINE, y se obtuvieron los resultados de la tabla D21.

Donde se puede ver que el mayor número de aciertos se da en el caso del aprendizaje OFFLINE, aunque la diferencia no es mucha, ya que sólo algunas imágenes son las que nos se logran clasificar correctamente. SI se grafica el número de aciertos en los rostros para el aprendizaje ONLINE tenemos:

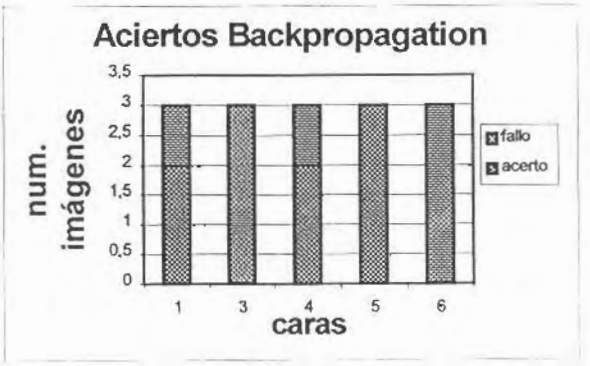


Gráfica 5.21

Aciertos de la red Backpropagation utilizando el aprendizaje ONLINE con el grupo de rostros humanos.

Donde se ve que la red no logra clasificar correctamente los rostros 2 y 3, y tiene una falla en los rostros 6 y 8. En estos dos últimos la falla ocurre al procesar la imagen del grupo c, lo que indica que la red si puede clasificar correctamente las imágenes que ya había procesado con anterioridad. Mientras que para los rostros 2 y 3, nunca logra hacer su correcta clasificación.

Para el grupo de imágenes con la variación de lentes obtenemos:

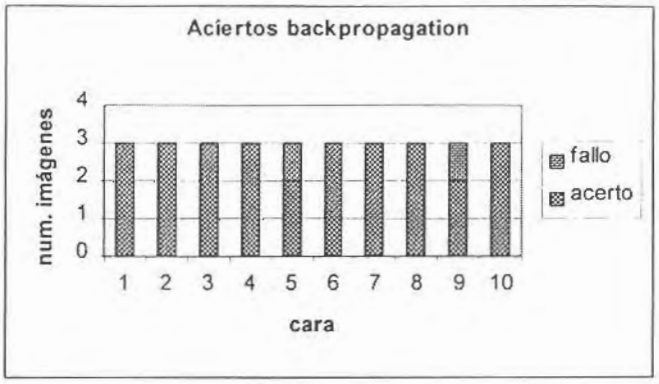


Gráfica 5.22

Aciertos de la red Backpropagation utilizando el aprendizaje ONLINE con el grupo de rostros humanos con la variación de lentes.

Donde se puede ver que existe una concordancia de resultados, ya que en el rostro 3 no se logra ningun acierto como sucedió para los grupos a, b, y c. Tampoco en el rostro 6 se logra aciertos y éste es un rostro que presento fallas con el grupo c. Por último los rostros 1 y 4 fallan al clasificar una imagen, pero si tomamos en cuenta que en esta imágenes hay una variación por los lentes, se puede tomar este resultado como bueno.

Para el aprendizaje OFFLINE, obtenemos lo siguiente para las imágenes de los grupos a, b, y c:

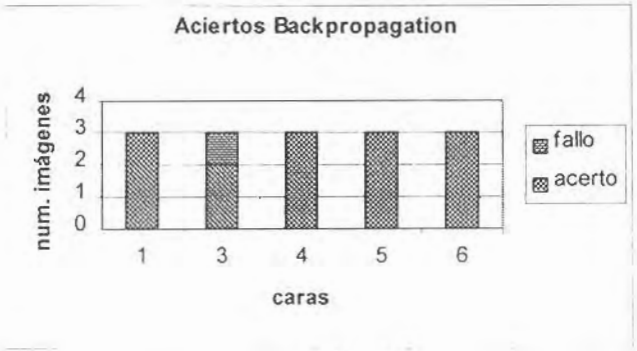


Gráfica 5.23

Aciertos de la red Backpropagation utilizando el aprendizaje OFFLINE con el grupo de rostros humanos.

Donde podemos ver que sólo en los rostros 5 y 9 es donde se comete un error, revisando la tabla D21 encontramos que las fallas fueron en el grupo c, el cual no se incluyó en el aprendizaje.

Y para el grupo de imágenes con la variación de los lentes obtenemos:



Gráfica 5.24

Aciertos de la red Backpropagation utilizando el aprendizaje OFFLINE con el grupo de rostros humanos con la variación de lentes.

Donde el error aparece en el rostro número tres, y ya que ninguna de estas imágenes fue incluida en la fase de aprendizaje, se toma que todas estas imágenes son nuevas para la red. Por tanto se considera que se tiene un alto grado de eficiencia, mejorando los resultados presentados por el aprendizaje ONLINE, para el mismo grupo.

5.6 Conclusiones.

A lo largo de este capítulo hemos visto como se comportan las dos redes ante distintos tipos de entradas para el reconocimiento de diferentes grupos de imágenes. Para el caso de las figuras geométricas se encontró que la red Backpropagation arroja mejores resultados. Pero conforme la complejidad de las entradas aumenta, como sucede en el caso de los momentos y el centro de la transformada de Fourier, es la red ART2 la que da mejores resultados. Pero la diferencia sólo es de un par de imágenes.

También se observó que la red ART2 siempre arroja los mismos resultados cuando los valores de sus variables se mantienen constantes. Mientras que el funcionamiento de la backpropagation va variando ya que sus pesos se inician al azar, lo que implica que sí queremos dejar una red funcionando para un problema dado, tenemos que hacer varias pruebas hasta encontrar los valores del número de neuronas ocultas que nos den el mayor número de aciertos en la fase de aprendizaje, sin gastar muchos recursos. Pero una vez encontrado dicho número, si los valores de las entradas presentan una gran diversidad, es necesario hacer varias pruebas para lograr el mejor funcionamiento de la red, esto es debido a que la superficie de error se vuelve más compleja y la red puede caer en mínimos locales más fácilmente. Debido a que se inician los valores de los pesos al azar siempre caeremos en distintos lugares de dicha superficie, los resultados mejoraran o empeorarán según el lugar de la superficie donde se empiece la búsqueda del mínimo. Si la red contiene mecanismos para la aceleración de la búsqueda del error mínimo y para salir de mínimos locales, sólo será necesario encontrar el número óptimo de neuronas ocultas. Ya que no importará donde comience la red a buscar el mínimo los mecanismos señalados asegurarán que encontrará un mínimo que proporcione los resultados deseados.

En general se obtienen mejores resultados con la red ART2 usando menos tiempo para las pruebas, ya que no depende del azar. Pero la información corre el riesgo de degradarse si no se tiene cuidado al introducir nuevas imágenes cuando el ρ esté en valores más bajos que la mínima similitud requerida entre imágenes de la misma clase. Ya que entonces la red clasificará mal las

imágenes y por lo tanto los prototipos guardados serán alterados con imágenes de clases diferentes. Lo anterior ocasionará que cada prototipo ya no contenga sólo características de una clase, sino que sea una mezcla de varias clases. Esto propiciará que la red se equivoque cada vez más en la clasificación de imágenes, y por lo tanto se degraden más los prototipos. Formándose un círculo vicioso que va mermando la información contenida en la red. Es por ello que en la red ART2 hay que tener sumo cuidado con el parámetro ρ.

En oposición, la red Backpropagation requiere más experimentos durante su fase de aprendizaje para encontrar la solución a un problema dado. Pero una vez logrado esto, la red conservará la información sin alterarla durante su proceso de funcionamiento. Sin embargo, en esta red es necesario tener muy bien definido el conjunto de imágenes que la red tiene que clasificar, ya que nuestro conjunto de imágenes de aprendizaje debe de representar lo más fidedignamente posible todas las variaciones de las imágenes a procesar. Para lo cual se puede optar por introducir varias veces la misma imagen aplicándole distintos tipos de cambios que se puedan presentar en la etapa de funcionamiento de la red como son la rotación, traslación, degradación, ruido, etc.. También es necesario hacer modificaciones en la topología de la red para mejorar tanto el tiempo como la eficiencia del aprendizaje, ya que hay problemas en los cuales será necesario agregar una o más capas ocultas de neuronas para que la red pueda encontrar más rápidamente el mínimo total de la función. Por otro lado, también se pueden agregar distintos mecanismos al algoritmo de la red Backpropagation para acelerar el encuentro del mínimo de la función de error, uno de los mecanismo más usados es el uso de momentos, el cual va revisando la dirección de variación del error y orienta los cambios siempre al mínimo. Pero todas estas modificaciones tiene como punto de partida la superficie de error sobre la que se esté trabajando, y como dicha superficie depende de las imágenes del conjunto de aprendizaje, va a variar según el problema a resolver. Ya que cada tipo de imagen nos dará diferentes tipos de superficies de error, entre más simples sean las entradas menos variaciones le tendremos que hacer al algoritmo, esto se ve reflejado en los resultados de esta red con el grupo de imágenes geométricas.

En el caso del procesado de imágenes de figuras geométricas se vió que la red funciona muy bien a pesar de estar utilizando su topología más simple. En cambio cuando se utiliza para el reconocimiento de letras ya no logra converger completamente durante la fase de aprendizaje, lo que ocasiona que no se obtenga el mismo número de aciertos que de imágenes utilizadas en esta etapa. Esto indica que la red cayó en un mínimo local o está variando sobre un mínimo sin llegar a él. Provocando que la red no pueda clasificar correctamente todas la imágenes durante su funcionamiento.

En resumen, la red ART2 tiene un buen funcionamiento y su topología no se ve afectada por el problema a solucionar, ya que únicamente se pueden cambiar sus variables para mejorar o empeorar su funcionamiento. Pero tiene siempre el riesgo de sufrir una degradación en su información. Mientras que la red que utiliza el algoritmo backpropagation, requiere de un trabajo más intenso durante su aprendizaje. Esto debido a la modificación de su topología de acuerdo al problema a solucionar, pero no corre el riego de perder ninguna porción de su información durante su funcionamiento, es decir, es muy estable en dicha etapa. Sin embargo es necesario contar con una gran base de datos para asegurar su óptimo funcionamiento. Por lo que podemos concluir que cada red posee ventajas en diferentes aspectos. Por tanto, depende de las características del problema a solucionar, el tipo de red que sea mejor utilizar.

En los casos en que las diferencias entre las distintas clases de imágenes sean muy marcadas las dos redes funcionan bien, y en la red Backpropagation no será necesario una gran cantidad de pruebas durante la fase de aprendizaje.

En el caso de que la diferencia entre las distintas clase de imágenes sea poco marcada, debemos analizar, si contamos con una buena base de ejemplos de todas la imágenes a procesar. Y sí estas imágenes son las únicas que se van a utilizar o la base irá cambiando y creciendo a lo largo del tiempo, en el primer caso se utilizaría una red Backpropagation, y en el segundo una red ART2.

5.7 *Trabajo futuro.*

En el presente trabajo sólo se mostró como funcionan las redes ART2 y backpropagation ante determinadas entradas para el reconocimiento de patrones en imágenes digitales, pero en ningún caso se llegó a su solución óptima. Para cada uno de los casos presentados se pueden introducir diversas mejoras que lograrían hacer que las redes mejoren su funcionamiento y poder así aplicarlas en problemas donde las imágenes no sean tan controladas. Pero en dicho caso es necesario enfocarse en un solo problema, por ejemplo el reconocimiento de caracteres, donde se pueden hacer más procesos a las imágenes antes de calcular sus momentos, con el fin de que la variación de un una imagen a otra sea mínima.

Respecto a las redes, se pueden tomar diferentes estrategias para mejorar su funcionamiento, para la red ART2, se pueden probar más combinaciones de valores en las variables A, B, C y D, mientras que en la red Backpropagation se puede agregar mecanismos para aumentar su convergencia. Pero esto dependerá de su aplicación. Hasta ahora no se ha encontrado la solución optima para el reconocimiento de patrones dentro de imágenes digitales, sino se han ido solucionando casos muy específicos donde se han obtenido resultados satisfactorios, como son el

reconocimiento de caracteres manuscritos, piezas mecánicas, componentes electrónicos, etc. Pero existen todavía una infinidad de problemas donde es necesario reconocer algún tipo de patrón en un conjunto de imágenes, por lo que el uso de redes neuronales aplicadas en este campo puede ser una buena opción. No solo las dos redes presentadas en este trabajo sirven para el reconocimiento de imágenes, sino que existe un gran número de redes neuronales artificiales con otras caracteristicas que se pueden utilizar, y no sólo para resolver el problema del reconocimiento de imágenes digitales, sino también se pueden ocupar para solucionar problemas de otra índole como son: el reconocimiento de voz, aprendizaje de procesos etc.

Apéndices

Apéndice A

Conjunto 1 de imágenes


Circulo1


Cuadro1


Rectangulo1


Triangulo1


Circulo2


Cuadro2


Rectangulo2


Triangulo2


Circulo3


Cuadro3


Rectangulo3


Triangulo3


Circulo4


Cuadro4


Rectangulo4


Triangulo4


Circulo5


Cuadro5


Rectangulo5


Triangulo5


Circulo6


Cuadro6


Rectangulo6


Triangulo6

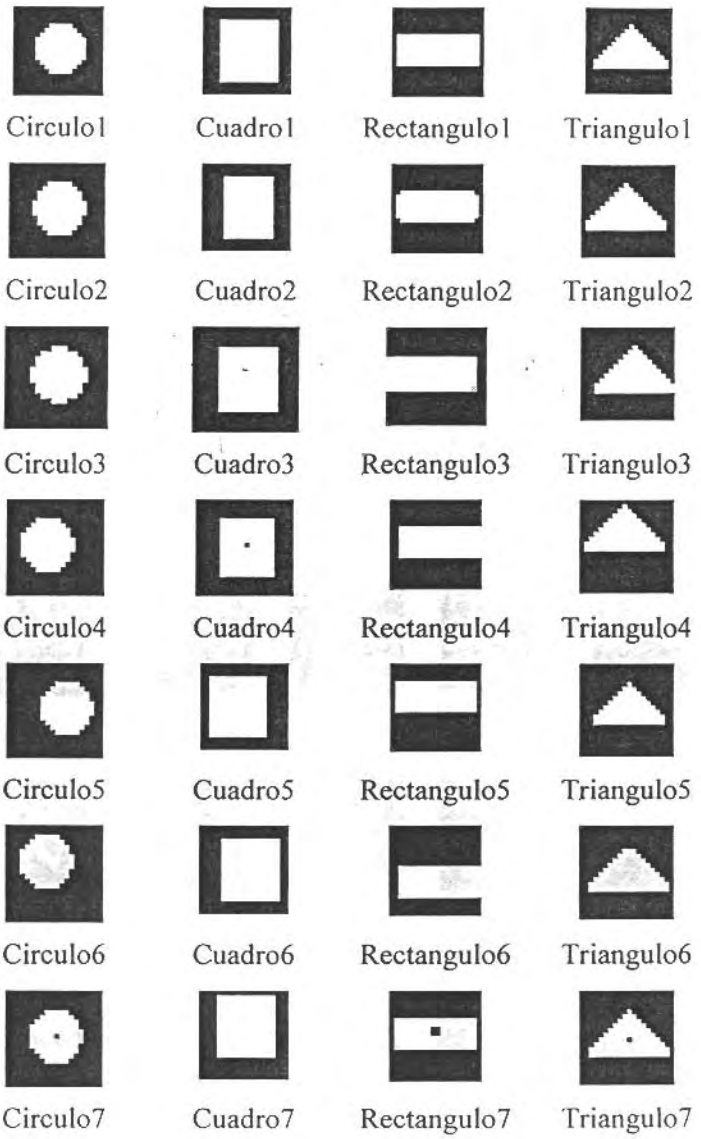

Circulo7


Cuadro7


Rectangulo7


Triangulo7

Conjunto 2 de imágenes



Grupo 3 de imágenes

La1.bmp Lh1.bmp Li1.bmp Lj1.bmp Lp1.bmp Lv1.bmp

La2.bmp Lh2.bmp Li2.bmp Lj2.bmp Lp2.bmp Lv2.bmp

La3.bmp Lh3.bmp Li3.bmp Lj3.bmp Lp3.bmp Lv3.bmp

La4.bmp Lh4.bmp Li4.bmp Lj4.bmp Lp4.bmp Lv4.bmp

La5.bmp Lh5.bmp Li5.bmp Lj5.bmp Lp5.bmp Lv5.bmp

La6.bmp Lh6.bmp Li6.bmp Lj6.bmp Lp6.bmp Lv6.bmp

La7.bmp Lh7.bmp Li7.bmp Lj7.bmp Lp7.bmp Lv7.bmp

Conjunto 4 de imágenes


Cara1a.bmp


Cara1b.bmp


Cara1c.bmp


Cara2a.bmp


Cara2b.bmp


Cara2c.bmp


Cara3a.bmp


Cara3b.bmp


Cara3c.bmp


Cara4a.bmp


Cara4b.bmp


Cara4c.bmp


Cara5a.bmp


Cara5b.bmp


Cara5c.bmp


Cara6a.bmp


Cara6b.bmp


Cara6c.bmp


Cara7a.bmp


Cara7b.bmp


Cara7c.bmp

Cara8a.bmp

Cara8b.bmp

Cara8c.bmp

Cara9a.bmp

Cara9b.bmp

Cara9c.bmp

Cara10a.bmp

Cara11b.bmp

Cara10c.bmp

Subconjunto de rostros sin de lentes.

Cara1_1a.bmp

Cara1_1b.bmp

Cara1_1c.bmp

Cara3_1a.bmp

Cara3_1b.bmp

Cara3_1c.bmp

Cara4_1a.bmp

Cara4_1b.bmp

Cara4_1c.bmp

Cara5_1a.bmp

Cara5_1b.bmp

Cara5_1c.bmp

Cara6_1a.bmp

Cara6_1b.bmp

Cara6_1c.bmp

Apéndice B

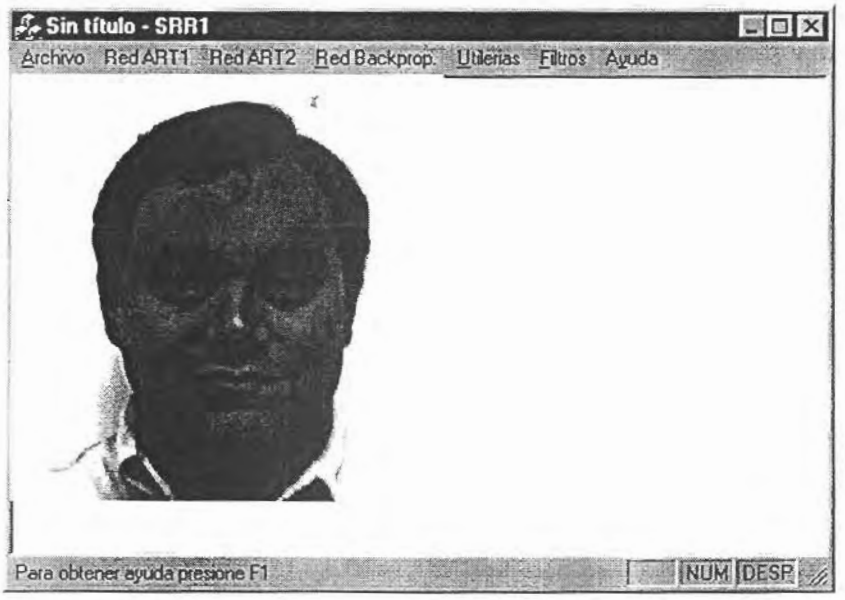
Manual de usuario

La aplicación de simulación de redes contiene la implementación de cada una de las redes descritas en este trabajo, además de contar con varios procesos de procesamiento de imágenes que sirven como utilerías para hacer el preprocesado de las imágenes.

La aplicación funciona con base a una ventana principal donde se despliega la imagen a procesar por una de las redes implementadas. Excepto cuando el procesamiento se hace por grupos, entonces las únicas imágenes desplegadas son la primera y la última. En dicha ventana existe también un menú principal, el cual contiene tanto los llamados a las funciones de cada red como los llamados a los procesamientos de imágenes digitales.

Las opciones que contiene el menú principal son: archivo, Red ART1, Red ART2, Red Backpropagation, utilerías, filtros y ayuda.

Bajo cada uno de estos rubros existen diversas opciones, las cuales se explicaran en detalle a continuación.



Archivo

Este submenú contiene las funciones de abrir y guardar, además de la opción de salida del sistema. Las dos primeras funciones, como su nombre lo indican, sirven para abrir y guardar imágenes bmp, que es el formato utilizado el sistema, solo se pueden abrir archivos *.bmp y los archivos también serán guardados en este formato.

Abrir, al seleccionar esta opción se desplegará un cuadro de diálogo para pedir el nombre del archivo a abrir. Hay que tomar en cuenta que el sistema toma como defecto el directorio de la última imagen leída, y para poder ejecutar cualquiera de las otras opciones del sistema, que no sea salir, se debe de haber abierto alguna imagen. Todas las imágenes abiertas serán desplegadas en una escala de grises de 0 a 255 tonos. Sí la imagen está a colores se transforma a tonos de grises usando el modelo de color YIQ, y los valores representados en la pantalla son los valores con los cuales trabajará la aplicación.

Guardar: esta opción nos permite guardar la imagen que tenemos desplegada en la pantalla en ese momento, no importa si es una imagen que se acaba de abrir o es el resultado de aplicarle uno o varios métodos de procesamiento de imágenes que se encuentran bajo el rubro de utilerías. El único resultado que si se despliega en la pantalla pero no se puede guardar es la transformada de Fourier.

Red ART1

Este submenú contiene todas las funciones relacionadas con el funcionamiento de la red ART1, y esta compuesto de dos opciones: iniciación y procesar.

Iniciación al escoger esta opción aparecerá un cuadro de diálogo, en donde aparecen el número neuronas de entrada y el de salida. Como la red ART1 solo puede recibir valores binarios se estableció por defecto que solo procese las imágenes completas. Debido a lo anterior el número de neuronas de entradas es el resultado de la multiplicación de las dimensiones de la imagen desplegada en la pantalla. Mientras que el número de neuronas ocultas puede variar sólo de 1 a 10. Esto es con el fin de facilitar el despliegue de resultados para todas la redes analizadas en este trabajo.

Tanto el número de neuronas de entrada como el número de neuronas de salida se pueden variar utilizando los botones que se encuentran a la izquierda de cada valor, sin embargo ambos tienen un máximo y un mínimo. El mínimo para los dos es el cero y el máximo, para las neuronas de entrada es el número total de pixeles que existen la imagen, y para las neuronas de salida es 10. Una vez establecido el número de neuronas de cada capa se elige el botón aceptar, lo que hará que la aplicación inicie los valores de los pesos de la red ART1, guardándolos en los archivos a1pesosv.dat y a1pesosw.dat, en formato binario.

Procesar ésta opción contiene a su vez dos apartados:

Procesa imagen completa, se encarga de vaciar a la entrada de la red los datos de la imagen desplegada en la pantalla y procesa la información. Para esto pide el parámetro de similitud que se

va ha usar durante el proceso, el cual puede ir de 0 a 100. Una vez terminado el proceso despliega un mensaje informando la clase que le fue asignada a la imagen.

Procesa conjunto, esta opción procesa los datos de varias imágenes y crea un archivo con los resultados obtenidos. Para esto pide el nombre de un archivo, el cual debe estar escrito en formato de sólo texto (*.txt) y contener los nombres de los archivos, con todo y extensión de las imágenes a procesar. Éste archivo debe estar en el mismo directorio que los archivos de las imágenes a procesar. El archivo con los resultados aparecerá en el mismo directorio bajo el nombre de "res1.txt", conteniendo el parámetro de similitud usado, el nombre de la imagen procesada, la clase obtenida y la similitud encontrada con el prototipo.

Red ART2

Este submenú contiene todas las funciones relacionadas con el funcionamiento de la red ART2, y está compuesto de dos opciones iniciación y procesar.

Iniciación. Al escoger esta opción aparecerá un cuadro de diálogo que permite escoger el tipo de entrada que tendrá la red, con las siguientes opciones:

- *Imagen completa*, la entrada son los valores reales de los pixeles de la imagen.
- *Centro de la transformada de Fourier* de la imagen, como su nombre lo indica la entrada de la red será los valores del centro del espectro de la transformada de Fourier de la imagen.
- *Momentos*, las entradas de la red serán los momentos invariantes de la imagen.
- *Histograma*, las entradas de la red serán el histograma de la imagen.

Seleccione el círculo a la izquierda de la opción deseada.

Si se elige la opción del centro de la transformada de Fourier, aparecerá otro cuadro de diálogo. En él que se debe especificar las dimensiones del centro que se tomará en cuenta, estas dimensiones tienen como máximo el tamaño de la transformada de Fourier de la imagen.

Después aparecerá un cuadro de diálogo, en donde se puede determinar el número neuronas de entrada de la red, el cual tendrá un mínimo de 0 y un máximo que varía según el tipo de entrada escogido con anterioridad. Quedando como sigue:

- Imagen completa – total de pixeles en la imagen.
- Centro de la transformada de Fourier- dimensión del cuadro escogido.
- Momentos – 7 entradas
- Histograma – 256 entradas.

En esta misma ventana se pide el número de salidas de la red, el cual sólo puede variar de 1 a 10. Luego aparece un cuadro que pide el valor de teta, el valor que aparece en el recuadro será dividido entre mil y es el que será usado para eliminar el ruido de la red.

Por último aparece un cuadro con los letreros de las tres versiones de la red, elija la versión que desee utilizar.

Una vez establecidos todos estos parámetros el sistema inicializa los valores de los pesos de la red ART2. Guardándolos en los archivos a2pesosv.dat y a2pesosw.dat, en formato binario.

Procesar. Esta opción contiene a su vez dos apartados:

Procesar imagen, se encarga de procesar los datos de la imagen desplegada en ese momento en la pantalla. Para esto vuelve a pedir el tipo de entrada de la red. Asegúrese que sea el mismo que se dio durante la iniciación. Después despliega un cuadro donde aparecen las variables A, B, C, y D, con valores predeterminados, si quiere puede modificarlos de acuerdo al funcionamiento que desee de la red. (Ver sección 3.7). Inmediatamente aparecerá el cuadro que pide el parámetro de similitud a utilizar en el proceso de clasificación. Por último aparece el cuadro de diálogo para elegir la versión de la red a utilizar. Se procesa la información y aparecerá un cuadro que muestra con que neuronas resonó la entrada, así como la similitud que tuvo con cada prototipo, oprima el botón aceptar y se despliega un letrero con la case obtenida para la imagen procesada.

Procesa conjunto esta opción procesa los datos de varias imágenes y crea un archivo con los resultados obtenidos. Para esto pide el nombre de un archivo, el cual debe estar escrito en formato de sólo texto (*.txt) y contener los nombres de los archivos de las imágenes a procesar, con todo y extensión. Éste archivo debe estar en el mismo directorio que los archivos de las imágenes a procesar. El archivo con los resultados aparecerá en el mismo directorio bajo el nombre de "res2.txt", conteniendo el parámetro de similitud usado, el nombre de la imagen procesada, la clase obtenida y la similitud encontrada con el prototipo. Al escoger esta opción la red pedira los mismos datos que en la opción de procesar, sólo que ahora los utilizará para todas la imágenes que aparezcan en la lista del archivo "*.txt".

Red Backprop.

Este submenu contiene las opciones de:

- *Conjunto de datos.*
- *Aprender*
- *Procesar*
- *Procesar lista*

El primer submenú es el encargado de manejar los conjuntos de datos de aprendizaje para esta red, y contiene las siguientes opciones:

Crear conjunto de datos: Permite crear un nuevo conjunto de datos de aprendizaje para la red, al seleccionarla despliega un cuadro de diálogo pidiendo el nombre del archivo que manejará el conjunto de datos, estos archivos deben tener terminación ". cd". Escriba un nombre y oprima aceptar.

Aparecerá el cuadro para elegir el tipo de entrada de la red (ver Iniciación ART2), luego otro cuadro de diálogo que la permite determinar el número de neuronas por cada capa de la red. Los número de neuronas de entrada y salida están limitados de acuerdo a la entrada escogida (ver Inciación ART2), y el número de neuronas de la capa oculta tendrá como máximo el número de neuronas de salida por el número de neuronas de entrada. Elija el valor que le convenga según el problema a solucionar, empiece con valores de 1 cifra. Recuerde que entre más neuronas ocupe, el número de conecciones en la red aumentará exponencialmente. Por último aparece un cuadro que permite determinar la tasa de aprendizaje a utilizar, va de 0 a 100. Todos estos datos se guardaran en el archivo ".cd".

Agregar datos a conjunto, esta opción permite agregar los datos de la imagen desplegada en la pantalla a un conjunto de datos. Primero pide el nombre del archivo .cd, al cual se van agregar los datos. Después pide la salida que se le asignará a dicha entrada, para esto despliega un cuadro con las clases de salida posibles, las cuales son el mismo número de neuronas de salida. Elija una clase y oprima el botón de aceptar. Una vez que se hayan guardado los datos de entrada aparecerá un letrero de "Proceso terminado", sus datos ya han sido agregados. Para agregar otro dato repita el procedimiento.

Agregar datos por lista, Funciona de manera muy similar a la anterior, sólo que agrega los datos de varias imágenes al mismo tiempo, por lo que pide el nombre de un archivo ".txt", el cual debe contener los nombres de las imágenes cuyos datos quiere agregar al conjunto de aprendizaje. Después de cada nombre debe seguir la salida deseada para esa imagen, esta salida debe estar compuesta del mismo número de ceros que el número de neuronas de salida, y se debe cambiar por un 1 solo el cero que corresponde a la neurona o clase que queremos que se le asigne a la imagen.

El sistema pedirá los mismos datos que en la opción anterior a excepción de la salida deseada y al finalizar de cargar todos los datos aparecerá un letrero de aviso.

Cambiar número de neuronas ocultas, esta opción permite cambiar el número de neuronas de la capa oculta para hacer diversas pruebas. Aaparece un cuadro de diálogo que sólo permite cambiar este número.

Aprender, este submenu contiene dos opciones:

Aprender conjunto de datos, el cual se encarga hacer que la red se aprenda un conjunto de datos utilizando el aprendizaje ONLINE, para esto pide el nombre del conjunto de datos, y al final despliega el número de aciertos y ciclos que se obtuvieron durante el aprendizaje.

Aprender conjunto de datos2, el cual se encarga hacer que la red se aprenda un conjunto de datos utilizando el aprendizaje OFFLINE, para esto pide el nombre del conjunto de datos, y al final despliega el número de aciertos y ciclos obtenidos durante el aprendizaje.

Clasifica esta opción permite poner en funcionamiento la red Backpropagation. Clasifica la imagen desplegada en la pantalla, antes de correr esta opción se tuvo que haber hecho el aprendizaje de algún conjunto de datos. La opción pide el conjunto de datos utilizado durante el aprendizaje para usar los mismos parámetros durante la clasificación. Como resultado despliega la clase asignada a la imagen.

Clasifica lista, esta opción permite clasificar una lista de imágenes, para lo cual pide un archivo ".txt", con los nombres de las imágenes solamente, y como resultado crea el archivo "resbc.txt", que contiene el nombre de la imagen y la clase asignada a ella.

Utilerías.

Este submenu contiene las siguientes opciones:

Tansformada de Fourier Calcula y despliega la transformada de Fourier de la imagen desplegada en la pantalla

Antitransformada Calcula la antitransformada de Fourier de una transformada hecha con anterioridad. Ocupa dos archivos binarios que se crean durante la ejecución de la transformada de Fourier. Por tanto sólo se puede ejecutar después de haber utilizado la opción anterior.

Filtro Ideal. No es filtro ideal tal cual, sino que pide el tamaño de un cuadro, hace cero el resto de la transformada de Fourier y le saca la antitransformada. Esta opción sirve para ver que tanto de la imagen se reconstruye utilizando diferentes valores para el centro de la transformada de Fourier en las entradas de la red.

Umbral. Convierte una imagen a blanco y negro utilizando el umbral que le demos. Los pixeles mayores al umbral se convertirán a blanco y los menores a negro.

Unificar fondo. Todos los pixeles con valor menor a 125 los vuelve blanco, sólo sirve para imágenes de tonos de gris sobre fondos casi blancos.

Multiplicar imagen. Multiplica los valores de los pixeles de una imagen por una constante. Pero no cambia la apariencia de la imagen, ya que para desplegarla tiene que escalar los valores al rango permitido.

Momentos. Calcula los momentos de la imagen desplegada en la pantalla y los guarda en el archivo "momentos.txt".

Rotar. Pide un ángulo y rota el contenido de la imagen de acuerdo al valor dado.

Agrandar cuadro. Agranda una imagen poniéndole un fondo negro, para que al rotarla no se pierda ninguna parte de la imagen original.

Aclarado. Distribuye los valores de los pixeles de la imagen, en toda la escala de grises.

Apéndice C

Funciones de la red Backpropagation

OnBcCreaCd:

- Abre el archivo ctr_num.dat, el cual guarda el número de control, incrementa este numero.
- Pide el nombre del conjunto de datos
- Crea los nombres de los archivos de datos que se definen para ese conjunto de datos y los guarda en el archivo .cd, los nombre de los archivos son creados agregando el numero de control, con 4 dígitos (ejemplo 0001), a las sílabas Pr, Pe o Ps, dependiendo si es el archivo de parámetros, patrones de entrada o de salida deseada, más la extensión ".dat".
- Cierra el archivo de conjunto de datos.
- Pide el tipo de entrada que aceptara la red.
- Si el tipo de entrada es el centro de la transformada de Fourier, pide las dimensiones del centro a tomar.
- Determina el máximo de neuronas que puede tener la red de acuerdo a su entrada.
- Pide las dimensiones de la red.
- Pide la tasa de aprendizaje de la red.
- Crea el archivo correspondiente a los parámetros de la red y guarda en el:
    - Las dimensiones de la red
    - El tipo de entrada que aceptara
    - Un contador, igual a 0 ,que llevará la cuenta de patrones en el conjunto de datos.
    - La tasa de aprendizaje
    - Y en su caso las dimensiones del cuadro central de la transformada de Fourier.
- Cierra el archivo de parámetros
- Finaliza la función.

OnBcAgreCd:

- Pide el nombre del archivo de conjunto de datos al cual se le agregarán los datos.
- Abre dicho archivo y obtiene los nombres de los archivos de parámetros, patrones de entrada y salida.
- Abre el archivo de parámetros y lee
    - Las dimensiones de la red
    - El tipo de entrada
    - El contador de patrones en el conjunto de datos.
    - Y en su caso las dimensiones del centro de la trasformada de Fourier.
- Cierra el archivo de parámetros
- Abre el archivo Pe y Ps correspondientes al conjunto de datos y ubica sus respectivos punteros al final de cada archivo.
- Pide el patrón de salida para la entrada que se esta guardando.
- Dependiendo del tipo de entrada, calcula los valores que serán el patrón de entrada de la red y los guarda en el archivo Pe.

- Guarda el patrón de salida en el archivo Ps.
- Cierra los archivos Pe y Ps.
- Aumenta el contador de entradas.
- Abre el archivo de parámetros Pr.
- Guarda el contador modificado.
- Cierra el archivo de parámetros.

- Termina la función.

OnBcAprCd:

- Pide el nombre del archivo de conjunto de datos a utilizar durante el aprendizaje.
- Abre dicho archivo y obtiene los nombres de los archivos de parámetros, patrones de entrada y salida.
- Cierra el archivo de conjunto de datos
- Abre el archivo de parámetros y lee:
  - Las dimensiones de la red
  - El tipo de entrada
  - El contador del numero de patrones en el conjunto de datos
  - La tasa de aprendizaje
  - Y en su caso las dimensiones del cuadro central de la transformada de Fourier.
- Cierra el archivo de parámetros
- Abre los archivos Pe y Ps y posiciona sus punteros al principio de cada archivo.
- Define un elemento de la clase Cred_Back.
- Asigna a la red Back las dimensiones, y la tasa de aprendizaje.
- Asigna la memoria necesaria para la red.
- Inicializa los pesos de la red.
- Inicializa un contador de ciclos.
- Repite

  Mueve los punteros de los archivos Pe y Ps al inicio de cada archivo.
  Inicializa un contador, para los patrones bien clasificados
  Si es aprendizaje OFFLINE
  Inicializa los arreglos que llevan la cuenta de los ajustes de los pesos
  Fin de condición
  Lee los patrones de entrada y salida y los asigna a la red Back.
  Si es aprendizaje ONLINE
  Llama a la función aprender
  Fin de condición
  Si es aprendizaje OFFLINE
  Llama a al función aprendedos
  Llama a la función calcula_ajuste
  Fin de condición.
  Si el error medio del patrón procesado es menor que 0.02
  Aumenta el contador de patrones bien clasificados
  Fin de condición.
  Mientras el contador de ciclos sea menor que 1000 y el contador de patrones bien clasificados sea menor al numero de patrones en el conjunto de datos.
- Vacía los pesos actuales en los archivos ".dat".

- Libera la memoria utilizada por la red.
- Cierra los archivos Pe y Ps.

- Despliega los resultados del aprendizaje, el numero de ciclos utilizados y los aciertos obtenidos.

OnRbClas:

- Pide el nombre del archivo de conjunto de datos que se uso para el aprendizaje.
- Abre dicho archivo y obtiene el nombre del archivo de parámetros.
- Cierra el archivo de conjunto de datos
- Abre el archivo de parámetros y lee:
  - Las dimensiones de la red
  - El tipo de entrada
  - El contador del numero de patrones en el conjunto de datos
  - La tasa de aprendizaje
  - Y en su caso las dimensiones del cuadro central de la transformada de Fourier.
- Cierra el archivo de parámetros
- Define un elemento de la clase Cred_Back.
- Asigna a la red Back las dimensiones, y la tasa de aprendizaje.
- Asigna la memoria necesaria para la red.
- Lee los pesos guardados en los archivos ".dat"
- Dependiendo del tipo de entrada, calcula los valores que serán el patrón de entrada de la red y los asigna a esta.
- Llama a la función Clasifica y obtiene el numero de clase.
- Despliega el resultados
- Libera la memoria ocupada por la red
- Fin de función.

Apéndice D

Contenido de las tablas

La columna 1 indica la imagen utilizada, la 2 la clase en la que debe de ser clasificada la imagen, y las demás columnas indican la clase que le asigno la red a cada imagen usando cada uno de los distintos valores ρ y el – indica que la red no pudo clasificar la imagen.

Tabla 1

Nombre	clase real	Parámetro de similitud									
		25	50	75	80	85	90	95	97	99	100
circulo 1	1	1	1	1	1	1	1	1	1	1	1
cuadro 1	2	1	1	2	2	2	2	2	2	2	2
rectangulo 1	3	1	1	2	2	2	3	3	3	3	3
triangulo 1	4	1	1	2	2	3	4	4	4	4	4
circulo 2	1	1	1	1	1	1	1	1	1	1	1
cuadro 2	2	1	2	3	3	4	2	2	2	2	2
rectangulo 2	3	1	1	2	3	2	3	3	3	3	3
triangulo 2	4	1	1	2	4	3	4	4	4	-	-
circulo 3	1	1	1	1	1	1	1	1	1	2	2
cuadro 3	2	1	2	3	-	4	2	2	2	-	-
rectangulo 3	3	1	1	3	3	2	3	3	3	-	-
triangulo 3	4	1	1	2	2	3	4	4	4	-	-
circulo 4	1	1	1	1	1	1	1	1	2	-	-
cuadro 4	2	1	2	4	-	4	2	2	-	-	-
rectangulo 4	3	1	1	3	3	2	3	3	3	-	-
triangulo 4	4	1	1	2	2	3	4	4	-	-	-
circulo 5	1	1	1	1	1	1	1	1	2	-	-
cuadro 5	2	1	2	4	-	4	2	2	-	-	-
rectangulo 5	3	1	1	3	3	4	3	3	-	3	-
triangulo 5	4	1	1	2	2	3	4	4	4	-	-
circulo 6	1	1	1	1	1	1	1	2	2	-	-
cuadro 6	2	1	2	4	-	-	2	-	-	-	-
rectangulo 6	3	1	1	2	3	2	3	3	-	-	-
triangulo 6	4	1	1	2	2	3	4	4	-	4	4
circulo 7	1	1	1	1	1	1	1	2	-	-	-
cuadro 7	2	1	2	4	-	-	2	-	-	-	-
rectangulo 7	3	1	1	3	3	-	3	-	-	-	-
triangulo 7	4	1	1	2	4	3	4	-	-	-	-

Tabla 2

Nombre	clase real	Parámetro de similitud									
		25	50	75	80	85	90	95	97	99	100
circulo 1	1	1	1	1	1	1	1	1	1	1	1
cuadro 1	2	1	1	2	2	2	2	2	2	2	2
rectangulo 1	3	1	1	3	3	3	3	3	3	3	3
triangulo 1	4	1	1	1	2	2	4	4	4	4	4
circulo 2	1	1	1	1	1	1	1	1	1	1	1
cuadro 2	2	1	2	2	4	4	2	2	2	2	2
rectangulo 2	3	1	1	3	3	3	3	3	3	3	3
triangulo 2	4	1	1	3	2	-	-	-	-	-	-
circulo 3	1	1	1	1	1	1	1	1	1	1	1
cuadro 3	2	1	2	2	4	4	2	2	2	2	2
rectangulo 3	3	1	2	4	3	3	3	-	-	-	-
triangulo 3	4	1	1	4	4	-	-	-	-	-	-
circulo 4	1	1	2	2	-	4	-	-	-	-	-
cuadro 4	2	1	3	-	-	-	2	2	2	2	2
rectangulo 4	3	1	3	-	3	3	-	-	-	-	-
triangulo 4	4	1	3	-	-	-	-	-	-	-	-
circulo 5	1	1	4	-	-	-	-	-	-	-	-
cuadro 5	2	2	-	-	-	-	-	-	-	-	-
rectangulo 5	3	2	4	-	-	-	-	-	-	-	-
triangulo 5	4	1	1	4	2	2	4	4	4	4	4
circulo 6	1	1	2	-	-	-	-	-	-	-	-
cuadro 6	2	2	-	-	-	-	-	-	-	-	-
rectangulo 6	3	3	-	-	-	-	-	-	-	-	-
triangulo 6	4	1	1	-	-	-	-	-	-	-	-
circulo 7	1	1	1	1	1	1	1	2	2	2	2
cuadro 7	2	2	-	-	-	-	-	-	-	-	-
rectangulo 7	3	2	-	-	3	3	3	3	-	-	-
triangulo 7	4	1	1	-	-	-	-	-	-	-	-

Tabla 3

Nombre	Clase	25	50	80	90	95	97	99	100
circulo 1	1	1	1	1	1	1	1	1	1
cuadro 1	2	1	1	1	1	1	1	1	1
rectangulo 1	3	1	1	1	1	1	1	1	1
triangulo 1	4	1	1	1	1	1	1	1	1
circulo 2	1	1	1	1	1	1	1	1	1
cuadro 2	2	1	1	1	1	1	1	1	1
rectangulo 2	3	1	1	1	1	1	1	1	1
triangulo 2	4	1	1	1	1	1	1	1	1
circulo 3	1	1	1	1	1	1	1	1	1
cuadro 3	2	1	1	1	1	1	1	1	1
rectangulo 3	3	1	1	1	1	1	1	1	1
triangulo 3	4	1	1	1	1	1	1	1	1
circulo 4	1	1	1	1	1	1	1	1	1
cuadro 4	2	1	1	1	1	1	1	1	1
rectangulo 4	3	1	1	1	1	1	1	1	1
triangulo 4	4	1	1	1	1	1	1	1	1
circulo 5	1	1	1	1	1	1	1	1	1
cuadro 5	2	1	1	1	1	1	1	1	1
rectangulo 5	3	1	1	1	1	1	1	1	1
triangulo 5	4	1	1	1	1	1	1	1	1
circulo 6	1	1	1	1	1	1	1	1	1
cuadro 6	2	1	1	1	1	1	1	1	1
rectangulo 6	3	1	1	1	1	1	1	1	1
triangulo 6	4	1	1	1	1	1	1	1	1
circulo 7	1	1	1	1	1	1	1	1	1
cuadro 7	2	1	1	1	1	1	1	1	1
rectangulo 7	3	1	1	1	1	1	1	1	1
triangulo 7	4	1	1	1	1	1	1	1	1

Tabla 4

nombre	clase	25	50	80	90	95	97	99	100
circulo 1	1	1	1	1	1	1	1	1	-
cuadro 1	2	1	1	1	1	1	1	1	-
rectangulo 1	3	1	1	1	1	1	1	1	-
triangulo 1	4	1	1	1	1	1	1	1	-
circulo 2	1	1	1	1	1	1	1	1	-
cuadro 2	2	1	1	1	1	1	1	1	-
rectangulo 2	3	1	1	1	1	1	1	-	-
triangulo 2	4	1	1	1	1	1	1	2	-
circulo 3	1	1	1	1	1	1	1	2	-
cuadro 3	2	1	1	1	1	1	1	2	--
rectangulo 3	3	1	1	1	1	1	1	4	-
triangulo 3	4	1	1	1	1	1	1	2	-
circulo 4	1	1	1	1	1	1	1	2	-
cuadro 4	2	1	1	1	1	1	1	2	-
rectangulo 4	3	1	1	1	1	1	1	4	-
triangulo 4	4	1	1	1	1	1	1	4	-
circulo 5	1	1	1	1	1	1	1	2	-
cuadro 5	2	2	2	2	2	2	2	2	-
rectangulo 5	3	1	1	1	1	1	1	4	-
triangulo 5	4	1	1	1	1	1	1	3	-
circulo 6	1	1	1	1	1	1	1	4	-
cuadro 6	2	2	2	2	2	2	2	2	-
rectangulo 6	3	2	2	2	2	2	2	-	-
triangulo 6	4	2	2	2	2	2	2	-	-
circulo 7	1	1	1	1	1	1	1	4	-
cuadro 7	2	1	1	1	1	1	1	4	-
rectangulo 7	3	2	2	2	2	2	2	-	-
triangulo 7	4	2	2	2	2	2	2	4	-

Tabla 5

nombre	Clase	25	50	80	90	95	97	99	100
circulo 1	1	1	1	1	1	1	1	1	1
cuadro 1	2	1	1	1	1	2	2	2	-
rectangulo 1	3	1	1	1	1	3	3	3	2
triangulo 1	4	1	1	1	1	4	4	4	3
circulo 2	1	1	1	1	1	1	1	1	-
cuadro 2	2	1	1	1	1	2	2	-	4
rectangulo 2	3	1	1	1	1	3	3	3	-
triangulo 2	4	1	1	1	1	4	4	-	-
circulo 3	1	1	1	1	1	1	1	1	-
cuadro 3	2	1	1	1	1	2	2	-	-
rectangulo 3	3	1	1	1	1	3	3	-	-
triangulo 3	4	1	1	1	1	4	4	-	-
circulo 4	1	1	1	1	1	1	-	-	-
cuadro 4	2	1	1	1	1	2	2	-	-
rectangulo 4	3	1	1	1	1	3	3	-	-
triangulo 4	4	1	1	1	1	-	-	-	-
circulo 5	1	1	1	1	1	1	-	-	-
cuadro 5	2	1	1	1	1	2	-	-	-
rectangulo 5	3	1	1	1	1	-	-	-	-
triangulo 5	4	1	1	1	1	4	4	-	-
circulo 6	1	1	1	1	1	-	-	-	-
cuadro 6	2	1	1	1	1	2	2	-	-
rectangulo 6	3	1	1	1	1	-	-	-	-
triangulo 6	4	1	1	1	1	4	4	4	-
circulo 7	1	1	1	1	1	1	1	1	-
cuadro 7	2	1	1	1	1	2	-	-	-
rectangulo 7	3	1	1	1	1	3	3	3	-
triangulo 7	4	1	1	1	1	4	4	4	-

Tabla 6

Nombre	clase	25	50	80	90	95	97	99	100
circulo 1	1	1	1	1	1	1	1	1	-
cuadro 1	2	1	1	1	1	2	2	2	1
rectangulo 1	3	1	1	1	1	3	3	3	2
triangulo 1	4	1	1	1	1	3	4	4	3
circulo 2	1	1	1	1	1	1	1	1	4
cuadro 2	2	1	1	1	1	2	2	2	-
rectangulo 2	3	1	1	1	1	3	3	3	-
triangulo 2	4	1	1	1	1	3	4	4	-
circulo 3	1	1	1	1	1	1	1	1	-
cuadro 3	2	1	1	1	1	2	2	2	-
rectangulo 3	3	1	1	1	1	3	3	3	-
triangulo 3	4	1	1	1	1	3	4	4	-
circulo 4	1	1	1	1	1	1	1	1	-
cuadro 4	2	1	1	1	1	2	2	2	-
rectangulo 4	3	1	1	1	1	3	3	3	-
triangulo 4	4	1	1	1	1	3	4	4	-
circulo 5	1	1	1	1	1	1	1	1	-
cuadro 5	2	1	1	1	1	2	2	2	-
rectangulo 5	3	1	1	1	1	3	3	3	-
triangulo 5	4	1	1	1	1	3	4	4	-
circulo 6	1	1	1	1	1	1	1	1	-
cuadro 6	2	1	1	1	1	2	2	2	-
rectangulo 6	3	1	1	1	1	3	3	3	-
triangulo 6	4	1	1	1	1	3	4	4	-
circulo 7	1	1	1	1	1	1	1	1	-
cuadro 7	2	1	1	1	1	2	2	2	-
rectangulo 7	3	1	1	1	1	3	3	3	-
triangulo 7	4	1	1	1	1	3	4	4	-

La columna 1 indica la imagen utilizada, la columna 2 indica la clase en la que debe de ser clasificada la imagen, y las demás columnas indican la clase asignada de acuerdo al número de neuronas ocultas usadas.

Tabla 7

nombre	clase	Neuronas ocultas									
		5	10	20	30	50	70	90	110	130	150
circulo 1	1	1	1	1	1	1	1	1	1	1	1
cuadro 1	2	2	2	2	2	2	2	2	2	2	2
rectangulo 1	3	3	3	3	3	3	3	3	3	3	3
triangulo 1	4	4	4	4	4	4	4	4	4	4	4
circulo 2	1	1	1	1	1	1	1	1	1	1	1
cuadro 2	2	2	2	2	2	2	2	2	2	2	2
rectangulo 2	3	3	3	3	3	3	3	3	3	3	3
triangulo 2	4	4	4	4	4	4	4	4	4	4	4
circulo 3	1	1	1	1	1	1	1	1	1	1	1
cuadro 3	2	2	2	2	2	2	2	2	2	2	2
rectangulo 3	3	3	3	3	3	3	3	3	3	3	3
triangulo 3	4	4	4	4	4	4	4	4	4	4	4
circulo 4	1	**2**	**4**	**2**	**2**	**2**	1	1	1	**2**	**2**
cuadro 4	2	2	2	2	2	2	2	2	2	2	2
rectangulo 4	3	3	3	3	3	3	3	3	3	3	3
triangulo 4	4	**3**	**3**	**3**	**3**	**1**	**3**	**1**	**3**	**3**	**3**
circulo 5	1	**2**	**4**	**2**	**2**	1	1	1	1	**2**	1
cuadro 5	2	2	2	2	2	2	2	2	2	2	2
rectangulo 5	3	3	3	3	3	3	3	3	3	3	3
triangulo 5	4	4	4	4	4	4	4	4	4	4	4
circulo 6	1	**2**	1	1	**2**	**2**	1	1	1	**2**	**2**
cuadro 6	2	2	2	2	2	2	2	2	2	2	2
rectangulo 6	3	**4**	**4**	**4**	**4**	**4**	**4**	**4**	**4**	**4**	**4**
triangulo 6	4	4	4	4	4	4	4	4	4	4	4
circulo 7	1	1	1	1	1	1	1	1	1	1	1
cuadro 7	2	2	2	2	2	2	2	2	2	2	2
rectangulo 7	3	3	3	3	3	3	3	3	3	3	3
triangulo 7	4	4	4	4	4	4	4	4	4	4	4

Tabla 8

nombre	clase	Neuronas ocultas									
		5	10	20	30	50	70	90	110	130	150
circulo 1	1	1	1	1	1	1	1	1	1	1	1
cuadro 1	2	2	2	2	2	2	2	2	2	2	2
rectangulo 1	3	3	3	3	3	3	3	3	3	3	3
triangulo 1	4	4	4	4	4	4	4	4	4	4	4
circulo 2	1	1	1	1	1	1	1	1	1	1	1
cuadro 2	2	2	2	2	2	2	2	2	2	2	2
rectangulo 2	3	3	3	3	3	3	3	3	3	3	3
triangulo 2	4	4	4	4	4	4	4	4	4	4	4
circulo 3	1	1	1	1	1	1	1	1	1	1	1
cuadro 3	2	2	2	2	2	2	2	2	2	2	2
rectangulo 3	3	3	3	3	3	3	3	3	3	3	3
triangulo 3	4	4	4	4	4	4	4	4	4	4	4
circulo 4	1	3	3	1	2	2	2	2	1	2	1
cuadro 4	2	2	2	2	2	2	2	2	2	2	2
rectangulo 4	3	3	3	3	3	3	3	3	3	3	3
triangulo 4	4	3	3	3	1	3	3	3	3	3	1
circulo 5	1	1	3	1	1	2	2	2	3	2	1
cuadro 5	2	2	2	2	2	2	2	2	2	2	2
rectangulo 5	3	3	3	3	3	3	3	3	3	3	3
triangulo 5	4	4	4	4	4	4	4	4	4	4	4
circulo 6	1	3	3	1	2	2	2	2	1	3	1
cuadro 6	2	2	2	2	2	2	2	2	2	2	2
rectangulo 6	3	3	4	4	4	4	4	4	4	4	4
triangulo 6	4	4	4	4	4	4	4	4	4	4	4
circulo 7	1	1	1	1	1	1	1	1	1	1	1
cuadro 7	2	2	2	2	2	2	2	2	2	2	2
rectangulo 7	3	3	3	3	3	3	3	3	3	3	3
triangulo 7	4	4	4	4	4	4	4	4	4	4	4

La columna 1 indica la imagen utilizada, la 2 la clase en la que debe de ser clasificada la imagen, y las demás columnas indican la clase que le asigno la red a cada imagen usando cada uno de los distintos valores ρ y el – indica que la red no pudo clasificar la imagen.

Tabla 9

		Similitud		
Nombre	clase	97	99	100
Ia1	1	1	1	1
Ih1	2	2	2	2
Ii1	3	3	3	3
Ij1	4	4	4	4
Ip1	5	5	5	5
Iv1	6	2	2	6
Ia2	1	1	6	-
Ih2	2	2	2	-
Ii2	3	3	3	-
Ij2	4	4	4	-
Ip2	5	5	5	-
Iv2	6	2	2	-
Ia3	1	1	6	-
Ih3	2	2	2	-
Ii3	3	3	3	-
Ij3	4	4	-	-
Ip3	5	5	5	-
Iv3	6	2	2	-

Tabla 10

nombre	clase	100	99	97	80
Ia1	1	1	1	1	1
Ih1	2	2	2	2	2
Ii1	3	3	3	3	3
Ij1	4	4	4	4	4
Ip1	5	5	5	5	5
Iv1	6	6	6	6	6
Ia2	1	-	-	1	1
Ih2	2	-	2	2	2
Ii2	3	-	3	3	3
Ij2	4	-	4	4	4
Ip2	5	-	5	5	5
Iv2	6	-	6	6	6
Ia3	1	-	1	1	1
Ih3	2	-	2	2	2
Ii3	3	-	3	3	3
Ij3	4	-	-	4	4
Ip3	5	-	5	5	5
Iv3	6	-	6	6	6

La primera columna indica el nombre de la imagen procesada, la segunda indica la clase que le corresponde a esta imagen, y las siguientes columnas indican la clase asignada por la red a la imagen dependiendo del valor de neuronas ocultas utilizadas.

Tabla 11

		Neuronas ocultas				
nombre	Clase	5	7	8	10	12
Ia 1	1	1	1	1	1	1
Ih 1	2	1	1	5	5	5
Ii 1	3	1	5	5	5	3
Ij 1	4	4	4	4	4	4
Ip 1	5	5	5	5	5	5
Iv 1	6	6	6	6	6	6
Ia 2	1	1	1	1	1	1
Ih 2	2	1	1	5	5	5
Ii 2	3	1	5	5	5	3
Ij 2	4	4	4	4	4	4
Ip 2	5	5	5	5	5	5
Iv 2	6	6	6	6	6	6
Ia 3	1	1	1	1	1	1
Ih 3	2	1	5	5	5	5
Ii 3	3	1	5	5	5	3
Ij 3	4	4	4	4	4	4
Ip 3	5	5	5	5	5	5
Iv 3	6	1	1	1	1	1

Tabla 12

		Neuronas ocultas				
nombre	Clase	5	7	8	10	11
Ia 1	1	1	1	1	1	1
Ih 1	2	4	6	2	5	1
Ii 1	3	4	6	2	5	1
Ij 1	4	4	4	4	4	4
Ip 1	5	5	5	5	5	5
Iv 1	6	6	6	6	6	6
Ia 2	1	1	1	1	1	1
Ih 2	2	4	6	2	5	1
Ii 2	3	4	6	2	5	1
Ij 2	4	4	4	4	4	4
Ip 2	5	5	5	5	5	5
Iv 2	6	6	6	6	6	6
Ia 3	1	1	1	1	1	1
Ih 3	2	4	6	2	5	1
Ii 3	3	4	6	2	5	1
Ij 3	4	4	4	4	4	4
Ip 3	5	5	5	5	5	5
Iv 3	6	1	1	1	1	1

La columna 1 indica la imagen utilizada, la 2 la clase en la que debe de ser clasificada la imagen, y las demás columnas indican la clase que le asigno la red a cada imagen usando cada uno de los distintos valores ρ y el – indica que la red no pudo clasificar la imagen.

Tabla 13.

		similitud		
nombre	Clase	97	99	100
Ia 1	1	1	1	1
Ih 1	2	2	2	2
Ii 1	3	3	3	3
Ij 1	4	4	4	4
Ip 1	5	5	5	5
Iv 1	6	2	2	6
Ia 2	1	1	6	-
Ih 2	2	2	2	-
Ii 2	3	3	3	-
Ij 2	4	4	4	-
Ip 2	5	5	5	-
Iv 2	6	2	2	-
Ia 3	1	1	6	-
Ih 3	2	2	2	-
Ii 3	3	3	3	-
Ij 3	4	4	-	-
Ip 3	5	5	5	-
Iv 3	6	2	2	-
Ia 4	1	1	1	-
Ih 4	2	2	2	2
Ii 4	3	3	3	-
Ij 4	4	4	4	4
Ip 4	5	5	5	5
Iv 4	6	2	2	-
Ia 5	1	1	1	-
Ih 5	2	2	2	-
Ii 5	3	3	3	-
Ij 5	4	4	4	-
Ip 5	5	5	5	-
Iv 5	6	2	2	6
Ia 6	1	2	-	-
Ih 6	2	2	2	-
Ii 6	3	3	3	-
Ij 6	4	5	5	-
Ip 6	5	5	5	-
Iv 6	6	2	2	-
Ia 7	1	5	5	-
Ih 7	2	2	2	-
Ii 7	3	3	3	-
Ij 7	4	5	5	-
Ip 7	5	4	4	-
Iv 7	6	5	5	-

Tabla 14

		similitud		
nombre	clase	100	99	95
Ia 1	1	1	1	1
Ih 1	2	2	2	2
Ii 1	3	3	3	3
Ij 1	4	4	4	4
Ip 1	5	5	5	5
Iv 1	6	6	6	6
Ia 2	1	-	-	1
Ih 2	2	-	2	2
Ii 2	3	-	3	3
Ij 2	4	-	4	4
Ip 2	5	-	5	5
Iv 2	6	-	6	6
Ia 3	1	-	1	1
Ih 3	2	-	2	2
Ii 3	3	-	3	3
Ij 3	4	-	-	4
Ip 3	5	-	5	5
Iv 3	6	-	6	6
Ia 4	1	-	1	1
Ih 4	2	2	2	2
Ii 4	3	-	3	3
Ij 4	4	4	4	4
Ip 4	5	5	5	5
Iv 4	6	-	6	6
Ia 5	1	-	1	1
Ih 5	2	-	2	2
Ii 5	3	-	3	3
Ij 5	4	-	4	4
Ip 5	5	-	5	5
Iv 5	6	6	6	6
Ia 6	1	-	-	6
Ih 6	2	-	2	2
Ii 6	3	-	3	3
Ij 6	4	-	5	5
Ip 6	5	-	5	5
Iv 6	6	-	6	6
Ia 7	1	-	5	5
Ih 7	2	-	2	2
Ii 7	3	-	3	3
Ij 7	4	-	5	5
Ip 7	5	-	4	4
Iv 7	6	-	5	5

La primera columna indica el nombre de la imagen procesada, la segunda indica la clase que le corresponde a esta imagen, y las siguientes columnas indican la clase asignada por la red a la imagen dependiendo del valor de neuronas ocultas utilizadas.

Tabla 15

		Neuronas ocultas				
nombre	clase	5	7	10	20	30
la 1	1	1	1	1	1	1
lh 1	2	3	6	6	4	6
li 1	3	3	3	6	4	3
lj 1	4	4	4	4	4	4
lp 1	5	5	5	5	5	5
lv 1	6	6	6	6	6	6
la 2	1	1	1	1	1	1
lh 2	2	3	3	6	4	6
li 2	3	3	3	6	4	3
lj 2	4	4	4	4	4	4
lp 2	5	5	5	5	5	5
lv 2	6	6	6	6	6	6
la 3	1	1	1	1	1	1
lh 3	2	3	3	6	4	6
li 3	3	3	3	6	4	3
lj 3	4	4	4	4	4	4
lp 3	5	5	5	5	5	5
lv 3	6	6	6	6	6	6
la 4	1	1	1	1	1	1
lh 4	2	3	6	6	4	6
li 4	3	3	3	6	4	3
lj 4	4	4	4	4	4	4
lp 4	5	5	5	5	5	5
lv 4	6	6	6	6	6	6
la 5	1	1	1	1	1	1
lh 5	2	3	6	6	4	6
li 5	3	3	3	6	4	3
lj 5	4	4	4	4	4	4
lp 5	5	5	5	5	5	5
lv 5	6	6	6	6	6	6
la 6	1	4	4	4	4	4
lh 6	2	3	6	6	4	6
li 6	3	3	3	6	4	3
lj 6	4	5	5	5	5	5
lp 6	5	5	5	5	5	5
lv 6	6	4	4	4	4	4
la 7	1	1	1	1	1	5
lh 7	2	3	6	6	4	6
li 7	3	3	3	6	4	3
lj 7	4	5	5	5	5	5
lp 7	5	4	4	4	4	4
lv 7	6	1	1	1	1	1

La columna 1 indica la imagen utilizada, la 2 la clase en la que debe de ser clasificada la imagen, y las demás columnas indican la clase que le asigno la red a cada imagen usando cada uno de los distintos valores de similitud. – indica que la red no pudo clasificar la imagen.

Tabla 16

		Similitud				
nombre	clase	5	7	10	20	30
la 1	1	1	1	1	1	1
lh 1	2	5	5	2	5	3
li 1	3	5	5	2	5	3
lj 1	4	4	4	4	4	4
lp 1	5	5	5	5	5	5
lv 1	6	6	6	6	6	6
la 2	1	1	1	1	1	1
lh 2	2	5	5	2	5	3
li 2	3	5	5	2	5	3
lj 2	4	4	4	4	4	4
lp 2	5	5	5	5	5	5
lv 2	6	6	6	6	6	6
la 3	1	5	1	1	1	1
lh 3	2	5	5	2	5	3
li 3	3	5	5	2	5	3
lj 3	4	4	4	4	4	4
lp 3	5	5	5	5	5	5
lv 3	6	6	6	6	6	6
la 4	1	1	1	1	1	1
lh 4	2	5	5	2	5	3
li 4	3	5	5	2	5	3
lj 4	4	4	4	4	4	4
lp 4	5	5	5	5	5	5
lv 4	6	6	6	6	6	6
la 5	1	1	1	1	1	1
lh 5	2	5	5	2	5	3
li 5	3	5	5	2	5	3
lj 5	4	4	4	4	4	4
lp 5	5	5	5	5	5	5
lv 5	6	6	6	6	6	6
la 6	1	6	4	4	4	4
lh 6	2	5	5	2	5	3
li 6	3	5	5	2	5	3
lj 6	4	5	5	5	5	5
lp 6	5	5	5	5	5	5
lv 6	6	6	4	4	4	4
la 7	1	1	5	1	1	5
lh 7	2	5	5	2	5	3
li 7	3	5	5	2	5	3
lj 7	4	5	5	5	5	5
lp 7	5	4	4	4	4	4
lv 7	6	1	1	1	1	1

La primera columna indica el nombre de la imagen, la segunda indica la clase correcta de la imagen y la tercera la clase asignada por la red

Tabla 17

Nombre	clase	Resultado
cara1a	1	1
cara2a	2	2
cara3a	3	2
cara4a	4	2
cara5a	5	2
cara6a	6	2
cara7a	7	1
cara8a	8	2
cara9a	9	2
cara10a	10	2

Tabla 18

Nombre	clase	Resultado
cara1a	1	1
cara2a	2	2
cara3a	3	3
cara4a	4	4
cara5a	5	5
cara6a	6	6
cara7a	7	7
cara8a	8	8
cara9a	9	9
cara10a	10	10

La primera columna indica el nombre de la imagen procesada, la segunda columna indica la clase a la que pertenece la imagen, y la últimas tres columnas indican las clases obtenidas al variar el número de neuronas ocultas entre 10, 30 y 50

Tabla 19.

Nombre	Clase	Neuronas ocultas		
		10	30	50
cara1a	1	1	1	1
cara2a	2	3	2	2
cara3a	3	3	6	6
cara4a	4	3	4	6
cara5a	5	5	5	5
cara6a	6	3	6	6
cara7a	7	3	7	7
cara8a	8	3	8	5
cara9a	9	3	9	9
cara10a	10	3	10	10

Tabla 20

Nombre	clase	Neuronas ocultas		
		10	30	50
cara1a	1	7	1	1
cara2a	2	7	2	2
cara3a	3	7	3	3
cara4a	4	4	4	4
cara5a	5	8	5	5
cara6a	6	7	6	6
cara7a	7	7	7	7
cara8a	8	8	8	8
cara9a	9	9	3	9
cara10a	10	4	6	10

La primera columna indica el nombre de la imagen procesada, la segunda columna indica la clase a la que pertenece la imagen, y la cuarta contiene las clases obtenidas al utilizar el aprendizaje ONLINE utilizando 30 neuronas ocultas, y la quinta columna contiene la clase obtenida al utilizar el aprendizaje OFFLINE utilizando 50 neuronas ocultas.

Tabla 21

nombre	clase	ONLINE	OFFLINE
cara 1 a	1	1	1
cara 2 a	2	8	2
cara 3 a	3	6	3
cara 4 a	4	4	4
cara 5 a	5	5	5
cara 6 a	6	6	6
cara 7 a	7	7	7
cara 8 a	8	8	8
cara 9 a	9	9	9
cara 10 a	10	10	10
cara 1 b	1	1	1
cara 2 b	2	8	2
cara 3 b	3	9	3
cara 4 b	4	4	4
cara 5 b	5	5	5
cara 6 b	6	6	6
cara 7 b	7	7	7
cara 8 b	8	8	8
cara 9 b	9	9	9
cara 10 b	10	10	10
cara 1 c	1	1	1
cara 2 c	2	8	2
cara 3 c	3	6	3
cara 4 c	4	4	4
cara 5 c	5	5	8
cara 6 c	6	9	6
cara 7 c	7	7	7
cara 8 c	8	5	8
cara 9 c	9	9	10
cara 10 c	10	10	10
caral 1 a	1	7	1
caral 1 b	1	1	1
caral 1 c	1	1	1
caral 3 a	3	10	3
caral 3 b	3	8	3
caral 3 c	3	8	8
caral 4 a	4	4	4
caral 4 b	4	1	4
caral 4 c	4	4	4
caral 5 a	5	5	5
caral 5 b	5	5	5
caral 5 c	5	5	5
caral 6 a	6	9	6
caral 6 b	6	9	6
caral 6 c	6	9	6

Bibliografía

[ANIL 1989] **Anil K. Jain,** FUNDAMENTALS OF DIGITAL IMAGE PROCESSING Prentice Hall, 1989 EUA, p 381 pp. 569

[BEALE 1992] **Beale R., Jackson T.,** NEURAL COMPUTING, AN INTRODUCTION, Institute of Physics Publishing, 2da. Reimpresión, 1992, Gran Bretaña, 240pp.

[FAUSETT 1994] **Fausett, Laurene V.,** FUNDAMENTAL OF NEURAL NETWORKS, Prentice Hall, 1994 EUA, 461 pp.

[FREEMAN 1993] **Freeman James A., Skapura David M.,** REDES NEURONALES ALGORITMOS, APLICACIONES Y TECNICAS DE PROGRAMACION. Adison Wesley/Diaz de Santos, 1993,E.U.A., 431pp.

[GONZALEZ 1996] **Gónzalez Rafael C., Woods Richard E.,** TRATAMIENTO DIGITAL DE IMÁGENES, Addison-Wesley/Diaz de Santos, 1996, E.U.A.773.

[HARALICK 1993] **Haralick Robert M., Shapiro Linda G.,** COMPUTER AND ROBOT VISION, vol. II, Addison-Wesley, 1993,E.U.A.,

[HILERA 1995] **Hilera José R., Martínez Víctor J.,** REDES NEURONALES ARTIFICIALES, FUDAMENTOS, MODELOS Y APLICACIONES, Addison-Wesley Iberoamericana, rama, 1995, E.U.A. 390pp.

[JAIN 1995] **Jain Ramesh, Kasturi Rangachar, Schunck Brian G.,** MACHINE VISION, McGraw-Hill international editions, 1995, Singapure.

[KENNETH 1996] **Kenneth R. Castleman,** DIGITAL IMAGE PROCESSING, Prentice Hall, 1996 EUA, pp.667

[PANDYA 1996] **Pandya Abhijit S., Macy Robert B.,** PATTERN RECOGNITION WITH NEURAL NETWORKS IN C++, CRC-Press, IEEE-Press, 1996 EUA., 410 pp.

[PRESS 1992] **Press W.H., Jeukoisk S.A., Vetterling W.T., Flannery, B.P.,** NUMERICAL RECIPES IN C; THE ART OF SCIENTIFIC COMPUTING, Cambridge University Press, 1992 , EUA, pp.748.

[ROGERS 1997] **Rogers Joey,** OBJECT ORIENTED NEURAL NETWORKS IN C++; Academic Press, 1997, E.U.A. 380pp.

Referencias de Internet

Gaudiano Paolo, CMU ARTIFICIAL INTELLIGENCE REPOSITERY,
http://www.cs.cmu.edu/afs/cs.cmu.edu/project/ai-repository/ai/areas/neural/systems/art/,Gaudiano.tgz, 19 DE MARZO DEL 2001

Lehar Steve, CMU ARTIFICIAL INTELLIGENCE REPOSITERY,
http://www.cs.cmu.edu/afs/cs.cmu.edu/project/ai-repository/ai/areas/neural/systems/art/,Art2.tgz, 19 DE MARZO DEL 2001

Referencias de Internet

Gaudiano Paolo, CMU ARTIFICIAL INTELLIGENCE REPOSITERY,
http://www.cs.cmu.edu/afs/cs.cmu.edu/project/ai-repository/ai/areas/neural/systems/art/,Gaudiano.tgz, 19 DE MARZO DEL 2001

Lehar Steve, CMU ARTIFICIAL INTELLIGENCE REPOSITERY,
http://www.cs.cmu.edu/afs/cs.cmu.edu/project/ai-repository/ai/areas/neural/systems/art/,Art2.tgz, 19 DE MARZO DEL 2001