Helo G., José E.; Sell S., Carlos. *Reconocimiento de dígitos utilizando redes neurales y lógica difusa.* **Tecnología en marcha.** Vol. 12, no. especial.

# RECONOCIMIENTO DE DIGITOS UTILIZANDO REDES NEURALES Y LOGICA DIFUSA

José E. Helo G.*
Carlos Sell S.**

*Las teorías de redes neurales y de lógica difusa se utilizan ampliamente para el reconocimiento de patrones. En particular se utilizan para el reconocimiento de formas tales como los dígitos numéricos.*

*Se propone un esquema mixto que utiliza una red neural y la teoría de la lógica difusa. La red neural de retropropagación establece un primer reconocimiento de los dígitos y posteriormente un programa que utiliza lógica difusa se encarga de evaluar los resultados de la red neural.*

*Se presenta además un proceso de construcción incremental, que se puede utilizar para construir programas de reconocimiento de patrones más complejos.*

## INTRODUCCION

La tecnología de las redes neurales se ha utilizado con bastante éxito para el reconocimiento de patrones. El nivel de reconocimiento de una red neural depende totalmente del proceso de entrenamiento al cual haya sido sometida. Muchas veces, para mejorar el nivel de reconocimiento de una red, es necesario utilizar archivos o registros computacionales de gran tamaño, y proceder con una nueva sesión de entrenamiento[3,6,7].

Para mejorar la calidad del reconocimiento en una red se propone unificar la teoría de las redes neurales (RN) con la teoría de la lógica difusa (LD). De esta manera, la red proporcionará los resultados en su forma numérica convencional y un programa que utiliza lógica difusa se encargará de interpretarlos. La red actúa como un primer filtro en el proceso de reconocimiento y el programa que utiliza lógica difusa proporcionará una mejor aproximación al proceso.

Para lograr un adecuado reconocimiento, se utiliza un bloque básico, que permite un proceso de construcción incremental. De esta forma, se puede mejorar un bloque sin tener que preocuparse por el comportamiento de los bloques restantes.

Finalmente se pueden construir programas para el reconocimiento de patrones más complejos utilizando las mismas unidades elementales, y el mismo principio de conexión entre las unidades elementales.

## DESCRIPCION DE LA RED NEURAL UTILIZADA

Para la construcción del reconocedor de patrones se utilizará la teoría clásica de la lógica difusa[9], y de las redes de retropropagación.

Se utiliza una red neural de propagación hacia adelante, que utiliza el paradigma de aprendizaje de retropropagación.

* Departamento de Computación, Instituto Tecnológico de Costa Rica.
** Coordinador Carrera de Informática, Sede del Atlántico, Universidad de Costa Rica.

Para calcular la función de salida de las neuronas se utiliza una función sigmoide monotónica con centro en "0".

Para el entrenamiento de la red, se utilizó un archivo de ejemplos. Los ejemplos consistían en patrones de dígitos digitalizados en una matriz de 8 x 8. La red fue entrenada de manera supervisada utilizando este archivo.

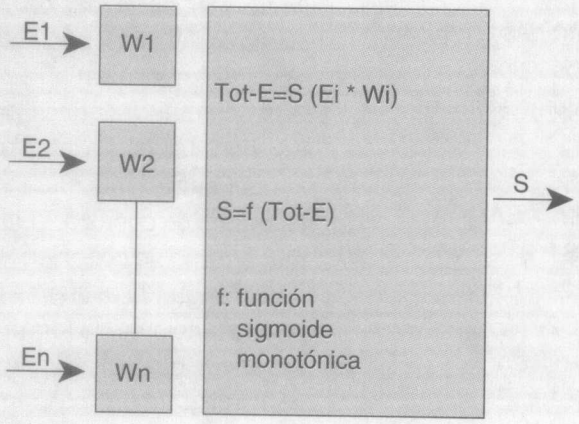
## INTERPRETACION DE LA INFORMACION EN UNA RED NEURAL DE RETROPROPAGACION

### La neurona artificial

La neurona artificial es el elemento donde se almacena la información en una red.

La Figura 1 muestra el esquema general de una neurona artificial.

*FIGURA 1: La neurona artificial.*




Sean E1, E2, ... , En las entradas a la neurona "N". Cada una de estas entradas se encuentra en el intervalo [0,1]. La función de salida de la neurona se calcula mediante la fórmula:

$$Tot\text{-}E = S\ Ei * Wi$$
$$S = f(Tot\text{-}E)$$

En esta fórmula "f" es una función sigmoide monotónica, cuyos valores de salida se encuentran en el intervalo [0,1]. De esta manera la salida de una neurona es siempre un valor entre [0,1].

El proceso de cómputo que realiza la neurona nos permite interpretar su salida como una "función de pertenencia de lógica difusa". Dado un conjunto de variables involucradas E1, E2, ..., En la neurona asocia a este grupo de variables el valor "S". Luego para el grupo de variables se puede establecer el conjunto difuso dado por:

$$[E1,E2,...,En\ ;\ u(E1,E2,...,En)] \text{ donde } u(E1,E2,...,En) = S.$$

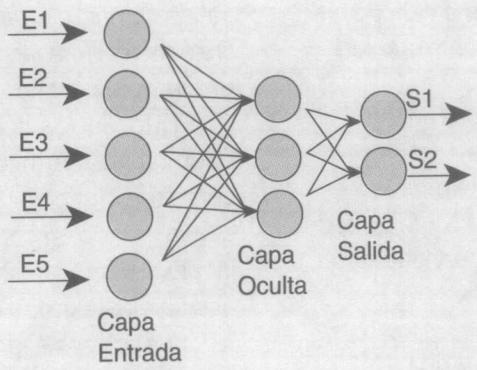
La neurona provee información adicional con respecto a cómo se construye la función de pertenencia. Para cada variable Ei, su respectivo peso Wi, indica la importancia relativa de esta variable en el cálculo de "u(E1,E2,...,En)". Por ejemplo cuando Wi es igual a 0, la variable Ei no tiene importancia en el cálculo de "u". Si por el contrario, el valor de Wi es un valor positivo alto, esto significa que la variable Ei tiene una alta importancia en el cálculo de "u".

### La red de neuronas

Una vez que se ha interpretado el papel de una neurona como una función de pertenencia difusa, es fácil realizar una extensión de esta idea para toda la red neural.

Una red de retropropagación no es sino una colección de neuronas, ordenadas por capas, donde existe conectividad total entre cualesquiera dos capas sucesivas. La primera capa, llamada *capa de entrada* recibe la información del medio externo y propaga sus resultados hacia las capas internas, llamadas *capas ocultas.* La última capa llamada *capa de salida*, es la que produce los resultados de la red neural. La Figura 2 muestra la arquitectura típica de una red de retropropagación.

*FIGURA 2. Red Neural de Retropropagación.*




En el ejemplo se presenta una red con 5 neuronas de entrada y 2 neuronas de salida. El número de neuronas de entrada y de salida puede ser de cualquier tamaño, dependiendo de la aplicación en la cual se trabaja.

Si se tiene una red con "n" neuronas de entrada y "m" neuronas de salida, se podría interpretar que cada neurona de salida representa una "función de pertenencia difusa", para el conjunto de variables de entrada.

Si una red posee las entradas E1, E2, ... , En y tiene como salidas los valores S1, S2, ... , Sm existen "m" conjuntos difusos constituidos por:

[ E1,E2,...,En ; u1(E1,E2,...,En) ]
con u1(E1,E2,...,En) = S1
[ E1,E2,...,En ; u2(E1,E2,...,En) ]
con u2(E1,E2,...,En) = S2
hasta llegar al conjunto difuso:
[ E1,E2,...,En ; um(E1,E2,...,En) ]
donde u1(E1,E2,...,En) = Sm

Los "m" conjuntos difusos se pueden describir mediante la fórmula:

[E1,E2,...,En ; ui(E1,E2,...,En)]
donde u1(E1,E2,...,En) = Si
con i:=1,...,m.

De igual forma, cualquier subsección de la red se podría interpretar como una "función de pertenencia parcial". De esta manera, al unir todas las funciones de pertenencia parciales se obtendría la "función de pertenencia final".

Al unir dos redes diferentes, se podría interpretar el proceso como un conjunto difuso al cual se le asocia una nueva función de pertenencia difusa. Se construirían entonces conjuntos difusos de segundo grado.

## POSIBLES MECANISMOS DE CONEXION ENTRE UNA RED NEURAL Y LOGICA DIFUSA

Como se ha dicho, se puede interpretar una red neural cualquiera como una función de pertenencia asociada a un conjunto difuso. Por lo tanto resulta natural considerar dos posibles mecanismos de conexión entre las redes y la lógica difusa.

Primero un mecanismo de conexión, donde una red propaga su información a un programa que utiliza lógica difusa. El programa de lógica difusa se encargaría de interpretar las salidas de la red como conjuntos difusos, pudiendo transformar estos valores en variables lingüísticas. Posteriormente se encargaría de tomar una decisión con la información construida.

La Figura 3 muestra este posible esquema de conexión. Llamaremos a este mecanismo de conexión RN->LD.

Un segundo esquema de integración podría funcionar de forma inversa. Un programa que utiliza lógica difusa podría construir para un conjunto de variables lingüísticas v1, v2,...,vk con sus respectivas funciones de pertenencia p1,p2,...,pk

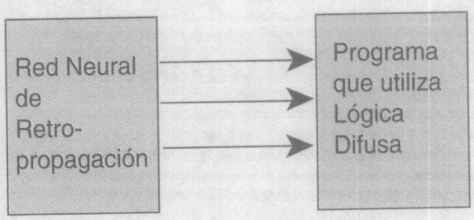



*FIGURA 3. Un esquema de conexión red neural, programa de lógica difusa (RN->LD).*

donde pi es elemento de [0,1]. De tal manera que los valores p1,p2,...,pk sirvan como valores de entrada a una red neural. La red neural utilizaría estos valores como entradas y construiría la respectiva función de salida.

La Figura 4 muestra este posible esquema de conexión entre un programa con LD y una RN al que denominaremos LD->RN.

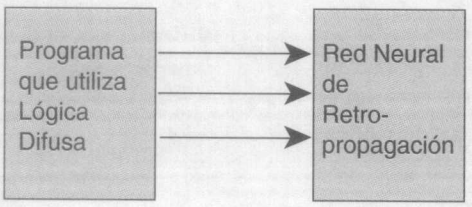



*FIGURA 4. Mecanismo de conexión entre un programa que utiliza lógica difusa y una red neural (LD->RN).*

## DESARROLLO DE UN RECONOCEDOR DE DIGITOS

A continuación se presenta un programa para el reconocimiento de dígitos. Se entenderá por un dígito un elemento del conjunto [0,1,2,...,9].

Para la construcción del reconocedor de dígitos se utilizará un enfoque incremental. Se construirá primero un bloque de construcción al que llamaremos "unidad de reconocimiento parcial" (URP). Para la construcción del sistema total se utilizarán varios de estos bloques unidos de forma apropiada.

Las redes neurales presentan dos problemas bien conocidos. Primero el problema de saturación de información[1,2,4] esto significa que una red puede reconocer o almacenar únicamente una cantidad limitada de información. Entre mayor sea el número de patrones que se desea reconocer, mayor debe ser el tamaño de la red que se debe utilizar. El segundo problema es el tiempo que se necesita para entrenar una red[3,6,7]. Para que una red pueda reconocer los patrones deseados, debe ser sometida a un proceso de entrenamiento. Para el proceso de entrenamiento se utiliza un archivo de "ejemplos". Entre mayor sea el número de patrones, mayor número de ejemplos debe contener el archivo. Es claro que un archivo de ejemplos de mayor tamaño representa mayor tiempo en el proceso de aprendizaje de la red.

El bloque de construcción o "unidad de reconocimiento parcial" (URP) se encuentra constituido por una red neural que propaga sus resultados hacia un programa que utiliza lógica difusa. De esta manera, se debe entrenar de forma aislada cada red neural dentro de cada URP.

Al utilizar un enfoque incremental en el cual se utilizan varias URP se disminuye el efecto de los problemas antes citados.

El sistema de reconocimiento de dígitos total se construye al interconectar apropiadamente cada una de estas unidades. Cada URP reconoce solamente un subconjunto de los dígitos, pero al agrupar las unidades parciales es posible reconocer cualquier miembro del conjunto total de dígitos.

### Construcción de la Unidad de Reconocimiento Parcial de dígitos (URP)

Para la construcción de la URP se usa un enfoque del tipo RN->LD. Cada URP consta de una red neural, que ha sido previamente entrenada. La red transmite su información a un programa que utiliza lógica difusa. Este programa se encarga de interpretar los resultados de la red.

Cada URP, debe reconocer un subconjunto de los dígitos. Por lo tanto la red que forma parte de la URP debe entrenarse con un archivo de ejemplos que concuerde con los dígitos por reconocer.

A continuación se describe el proceso para la construcción de una URP que se encarga de reconocer el subconjunto de los dígitos [1,2,3].

Se utilizó un archivo de entrenamiento con patrones de los dígitos "1", "2" y "3".

Para la creación de archivo de entrenamiento se le solicitó a un grupo de voluntarios que escribieran en una cuadrícula de 4 cm de largo por 4 cm de ancho cada uno de estos dígitos varias veces.

Posteriormente se llevó a cabo un proceso de digitalización para transformar cada uno de esos dígitos en una matriz de ceros y unos. La matriz o imagen digitalizada posee una dimensión de 8 filas por 8 columnas. Se utilizó un proceso de digitalización tradicional: cada cero de la matriz corresponde a un lugar donde "no hay ningún trazo" en la imagen original, cada uno de la matriz corresponde a un lugar donde "hay un trazo" en la imagen original.

Una vez obtenida la imagen digitalizada, se utiliza un algoritmo de acomodamiento. Este algoritmo busca posicionar la imagen hacia la derecha y hacia abajo en la matriz. El algoritmo de acomodamiento es relativamente sencillo y rápido. Consiste en borrar la columna más a la derecha mientras esta conste solo de ceros; posteriormente se borra la fila más hacia abajo mientras esta se encuentre llena de ceros.

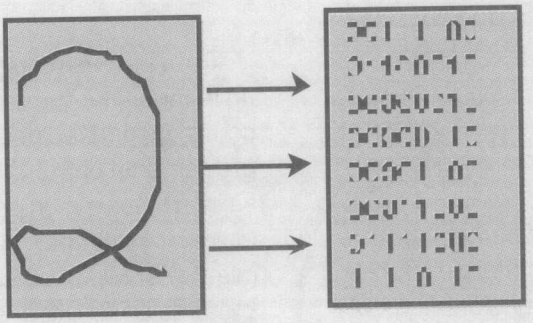


*FIGURA 5. Proceso de digitalización.*

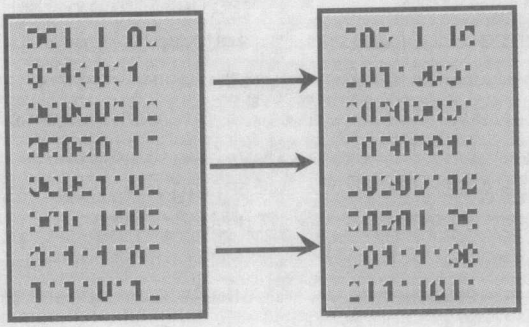


*FIGURA 6. Algoritmo de acomodamiento.*

Las Figuras 5 y 6 explican respectivamente el proceso de digitalización y el algoritmo de acomodamiento. En ellas se presenta el proceso para convertir el patrón de un dígito "2", en una imagen digitalizada. Primeramente se convierte el dígito en una matriz de ceros y unos. Posteriormente la matriz se "acomoda" de manera que la imagen digitalizada se localice lo más hacia abajo y hacia la derecha posible.

La RN consiste en una red que utiliza el algoritmo de aprendizaje de retropropagación clásico, tal como el expresado en [3] y [8].

La red neural posee 64 neuronas de entrada, 30 neuronas en su capa oculta y 3 neuronas de salida. Las 64 neuronas de entrada corresponden a cada elemento de las 64 entradas de la matriz producida por el proceso de digitalización y acomodamiento. Las 3 neuronas de saiida corresponden a cada uno de los 3 dígitos que se deben reconocer. De esta manera un valor cercano a "1" en la neurona de salida 1 (S1=1), y valores cercanos a cero en las neuronas de salida 2 y 3 (S2=0, S3=0) representarían que la red ha reconocido que el patrón de entrada corresponde en gran medida al patrón del dígito "2".

Una vez realizado el entrenamiento de la red, se dice que la red está lista para ser ejecutada. En el proceso de reconocimiento se utiliza únicamente la red neural entrenada.

La RN, que ya ha sido entrenada, recibe la información de entrada, procesa esta información y propaga sus resultados hacia un programa que utiliza LD. El programa que utiliza LD convertirá la información numérica de la red en una variable lingüística.

Como ya se ha dicho, la información de salida de la red puede interpretarse como una función de pertenencia difusa. Por lo tanto, el convertir los valores numéricos de salida de la RN en una variable lingüística es un proceso bastante natural.

La variable lingüística empleada refleja el grado de exactitud con el cual la RN ha logrado reconocer el patrón de entrada. Se utiliza una variable lingüística que puede tomar uno de varios valores difusos. Los valores que puede tomar la variable lingüística son:

- Muy Alta seguridad.
- Alta seguridad.
- Regular seguridad.
- Poca seguridad.
- Muy Poca seguridad.

Por ejemplo, si el patrón de salida corresponde al reconocimiento del dígito "2", el resultado de la variable lingüística sería uno de los siguientes enunciados:

- Muy Alta seguridad que es un "2".
- Alta seguridad que es un "2".
- Regular seguridad que es un "2".
- Baja seguridad que es un "2".
- Muy Baja seguridad que es un "2".

**Construcción de la variable lingüística a partir de los resultados de la red**

Para convertir los resultados numéricos de la red en una variable lingüística se utiliza la técnica de Zadeh[10, 11 y 12]. El valor que tome la variable lingüística dependerá básicamente de la diferencia absoluta que exista entre las salidas de la RN. Entre mayor sea esta diferencia, mayor certeza o "seguridad" de que el patrón ha sido reconocido correctamente. Entre menor sea la diferencia, menor certeza o "seguridad" de que el patrón ha sido reconocido correctamente.

Para convertir los resultados de la red en una variable lingüística, se utiliza una función creciente cóncava hacia abajo. Esto representa que conforme aumenta la diferencia de los resultados así aumenta el grado de la variable lingüística que se debe asignar.

La Figura 7 muestra la relación entre la diferencia de los resultados y el grado de la variable lingüística asignada.

A continuación se esboza el algoritmo que se utiliza para convertir los resultados de la RN a la respectiva variable lingüística.

Sean S1, S2,...,Sm las salidas de la RN. Se debe recordar que Si representa el valor de pertenencia que asoció la RN al patrón "i"; así el subíndice representa el dígito respectivo. Para calcular la variable lingüística (VL) se deben hacer los siguientes cálculos.

Sx = máx ({S1,S2,...,Sm}).

Sy = máx ({S1,S2,...,Sm} - {Sx} )

Dif = (Sx) - (Sy)

La variable "Dif" establece el grado de reconocimiento obtenido por la RN. Este grado de reconocimiento debe convertirse a una variable lingüística. Para ello se usa un conjunto de reglas construidas de forma aproximada a partir de la función que relaciona "magnitud de la diferencia" con "grado de seguridad".

Si Dif está en [0,40, 1,00[ entonces VL = Muy alta seguridad que es un dígito "x".

Si Dif está en [0,20, 0,40[ entonces VL = Alta seguridad que es un dígito "x".

Si Dif está en [0,10, 0,20[ entonces VL = Regular seguridad que es un dígito "x".

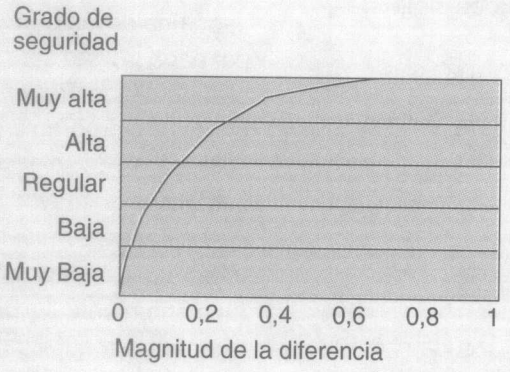



*FIGURA 7: Relación entre los resultados de la RN y el grado de la variable lingüística.*

Si Dif está en [0,05, 0,10[ entonces VL = Poca seguridad que es un dígito "x".

Si Dif está en [0,00, 0,05[ entonces VL = Muy Poca seguridad que es un dígito "x".

### Ejemplo del funcionamiento de la URP

A continuación se presenta un ejemplo del funcionamiento de la URP que reconoce los dígitos "1", "2" y "3".

La Figura 8 muestra el proceso de reconocimiento de un dígito "2" mediante una URP que reconoce los dígitos "1", "2", y "3". Dicha URP se designará como una URP(1,2,3).

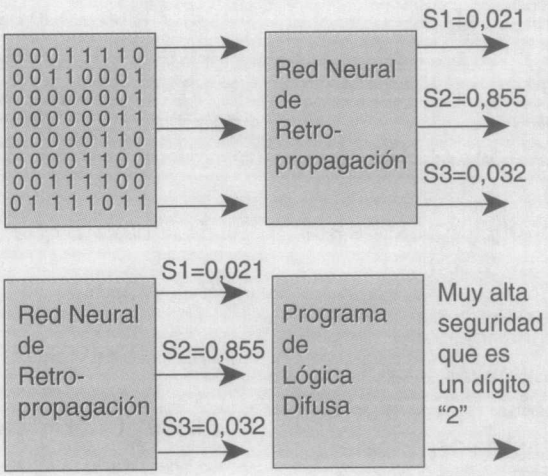



*FIGURA 8. Proceso de reconocimiento de un dígito "2" por una URP(1,2,3).*

El patrón digitalizado de un dígito "2" presentado previamente en la Figura 6 fue utilizado como información de entrada a la URP(1,2,3) (URP que reconoce dígitos "1", "2" y "3").

La RN que se encuentra en la URP produce en sus neuronas de salida los valores:

S1 = 0,021
S2 = 0,855
S3 = 0,032

El programa que utiliza LD, recibe estos valores de la RN. De acuerdo con el algoritmo presentado en la sección anterior realiza los siguientes cálculos:

Sx = máx ( {S1, S2, S3} )
Sx = máx ( {0,021, 0,855, 0,032} )
Sx = S2 (nótese que x = "2")
Sy = máx ( {S1, S3} )
Sy = máx ( {0,021, 0,032} )
Sy = S3
Dif = S2 - S3
Dif = 0,855 - 0,032
Dif = 0,823

Luego la variable lingüística producida por el conjunto de reglas es:

"Muy Alta seguridad que es un dígito "2"

### Integración de las URP para la construcción del programa reconocedor de dígitos

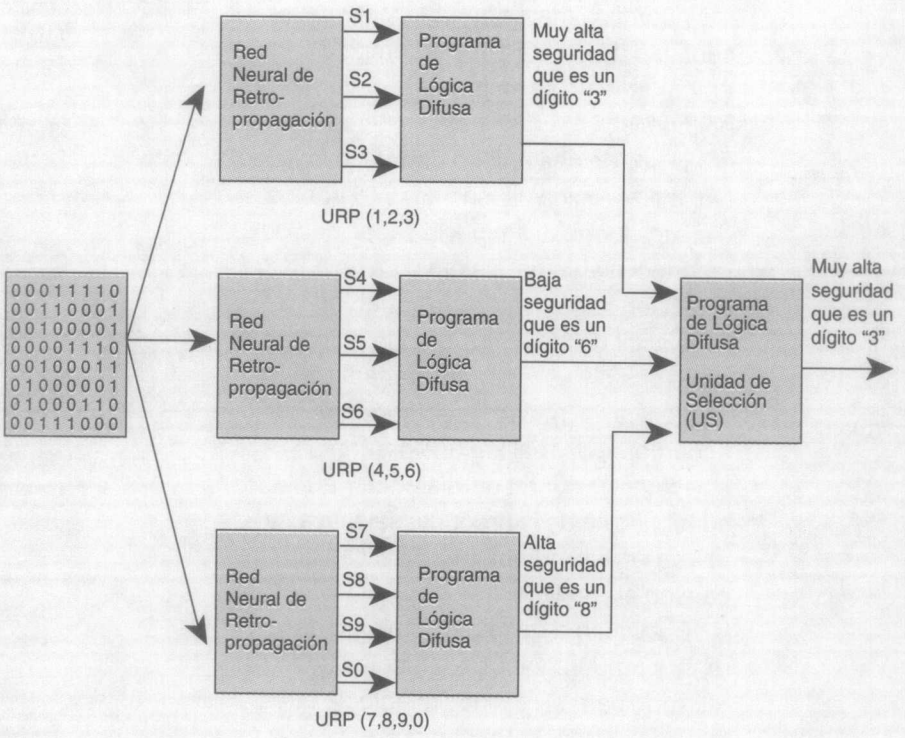
Se debe construir un conjunto de URP de tal manera que al unir todos estos bloques se pueda reconocer cualquier dígito. Ya se ha estudiado el problema para construir RN de reconocimiento de dígitos tomando subconjuntos de acuerdo con diversos criterios[2,6,7]. Para la construcción de nuestro modelo se sigue un proceso de agrupamiento natural de los dígitos.

Se deben construir entonces tres unidades de URP. Una para reconocer el conjunto [1,2,3], otra para reconocer el conjunto [4,5,6] y otra para reconocer el conjunto [7,8,9,0]. A cada una de estas unidades se les llamará respectivamente URP(1,2,3); URP(4,5,6); y URP(7,8,9,0).

El patrón de entrada se propagará a cada una de las URP y la salida producida por cada unidad será entonces una variable lingüística. Para unir las unidades se utiliza otro programa de Lógica Difusa que se encarga de seleccionar la variable lingüística de mayor "grado" de seguridad.

La Figura 9 muestra el proceso de integración de las distintas URP y el funcionamiento del programa que identifica la variable lingüística de mayor grado de seguridad.

*FIGURA 9. Proceso de Integración de las URP.*




En este ejemplo se muestra cómo un patrón ambiguo puede ser evaluado por las distintas URP. De esta forma la URP(1,2,3) lo identifica con "Muy Alta seguridad como un dígito 2"; mientras que la URP(4,5,6) lo identifica con "Baja seguridad como un dígito 6"; y la URP(7,8,9,0) lo identifica con "Alta seguridad como un dígito 8". El último programa que utiliza lógica difusa, llamado "unidad de selección" (US) se encarga de elegir la variable lingüística con mayor grado de seguridad.

## CONCLUSIONES

Al haberse interpretado la información contenida en las neuronas artificiales y las redes neurales como funciones de pertenencia difusa, la integración entre las RN y la LD resulta bastante natural.

Los dos posibles mecanismos de integración RN->LD y LD->RN son válidos y coherentes con la teoría que sustenta ambas tecnologías.

Se utilizó el mecanismo RN->LD para la construcción de un bloque básico llamado Unidad de Reconocimiento Parcial (URP). Utilizando las URP y mediante un enfoque incremental se construye un sistema capaz de reconocer patrones con forma de dígitos.

El enfoque incremental a partir de bloques de construcción evita dos problemas básicos en el proceso de diseño de las redes neurales.

Primero, se evita la llamada *saturación de la información*, puesto que una red solo puede reconocer un número finito de patrones, el enfoque incremental permite que cada subred neural pueda reconocer eficientemente un subconjunto del conjunto total de patrones que se deben reconocer.

Segundo, puesto que la red debe reconocer solo un subconjunto del conjunto total de patrones, el archivo de entrenamiento estará constituido únicamente por ejemplos relacionados con el subconjunto de patrones que debe reconocer la subred. Esto permite que el tiempo de entrenamiento de la red sea relativamente pequeño. Como es sabido, entre mayor sea el archivo de entrenamiento, se requiere una

red neural de mayor tamaño (con mayor cantidad de neuronas) y el tiempo de entrenamiento se incrementa de forma no polinomial.

El uso de una unidad de selección (US) que utiliza lógica difusa para integrar los distintos bloques amplía la capacidad de reconocimiento del sistema. La unidad de selección debe escoger la variable lingüística de mayor grado. Sin embargo, si no es posible escoger una variable lingüística, la unidad de selección puede encargarse de realizar una pregunta selectiva al usuario del sistema. Para la construcción de la pregunta selectiva se utilizarían las variables lingüísticas que causan el conflicto.

El esquema de reconocimiento de dígitos presentado en el artículo puede ser utilizado para el reconocimiento de otros tipos de patrones. En particular el reconocimiento de letras, o patrones de diagnóstico.

Sin embargo, la escogencia de los subconjuntos para cada una de las unidades de reconocimiento (URP), así como las variables lingüísticas por utilizar deben de ser construidas con base en las características particulares de la aplicación.

## LITERATURA CONSULTADA


1. Abu-Mostafa, Yaser S. Information Capacity of the Hopfield Model, *IEEE Transactions.* No. 4, 1985.
2. Hecht-Nielsen,Robert.Neurocomputer Applications", *IEEE Asilomar & Signals*, 1988.
3. Jones, William P.; Hoskins, Josiah. Back-Propagation: a generalized delta learning rule, *BYTE*, October, 1987.
4. Parker, D.B. Comparison of Algorithms for Neuron Like Cells. *AI-Expert*, 1990.
5. Stanley, Jeannette. *Introduction to Neural Networks.* California Scientific Software, California, 1989.
6. Wasserman, Philip D. ; Schwartz, Tom. Neural Networks: Part 1; *IEEE Expert*, 1988.
7. Wasserman, Philip D. ; Schwartz, Tom. Neural Networks: Part 2; *IEEE Expert,* 1988.
8. Wasserman, Philip D. *Neural Networks: Theory and Practice. V*an Nostrand Reinhold, New York, 1989.
9. Zadeh, L.A. Fuzzy Sets, *Information and Control.* Vol. 8, 1965.
10. Zadeh, L.A. The Concept of a Linguistic Variable and its Application to Approximate Reasoning-I.", *Information Sciences.* Vol. 8, No. 1, 1975.
11. Zadeh, L.A. The Concept of a Linguistic Variable and its Application to Approximate Reasoning-II. *Information Sciences.* Vol. 8, No. 2, 1975.
12. Zadeh, L.A. The Concept of a Linguistic Variable and its Application to Approximate Reasoning-III, *Information Sciences.* Vol. 8, No. 3, 1975.