Implementación de redes neuronales y lógica difusa para la clasificación de patrones obtenidos por un Sónar[3]

***Cruz Salazar, Luis A.**[1], ***Muñoz Aldana, David J.**[1], ****Contreras Montes, Juan A.**[2]

**Fundación Universitaria Tecnológico Comfenalco ** Escuela Naval Almirante Padilla "ENAP"*

dmunoz@tecnologicocomfenalco.edu.co
lcruzs@tecnocomfenalco.edu.co
epcontrerasj@ieee.org

Cartagena, Colombia

Abstract— This article describes a methodology using neural network models Adaline architecture and linguistically interpretable fuzzy systems, both algorithms were used to classify data signal rocks and metals, obtained through sonar. First, the neural network requires training match each input vector with the corresponding output vector for comparison with the desired output, and obtained, through feedback differential, an algorithm that minimizes the error. Secondly, the diffuse pattern contains overlapping of triangular sets to adjust the number of data sets for the antecedent, and *singletons* for to the consequent. In the evaluation of the rules are used instead of operators average *T-norm* and the consequents are adjust using recursive least squares. The promising aspect of this research was to achieve a good accuracy in the validation, after training of neural networks, and apply the fuzzy model without sacrificing the fuzzy system interpretability. Both methods are used making little modifications of parameters for obtain the best succes percentage, the lowest mean square error (MSE), decreasing execution time, reducing effort in the processing machine and without recourse to other artificial intelligence techniques.

Index Terms— Sonar, Neural Networks, Fuzzy Logic, Interpretability, MSE, Singleton sets.

I. INTRODUCCIÓN

Los científicos se han caracterizado por su búsqueda constante en brindar facilidades y orientado para algunos, sus estudios en las capacidades humanas como una fuente de nuevas ideas para el diseño de las nuevas máquinas, donde puedan implementarse fácilmente algoritmos para resolver multitud de problemas que antes resultaban engorrosos de resolver. Así, la inteligencia artificial es un intento por descubrir y describir aspectos de la inteligencia humana que pueden ser simulados mediante máquinas. Esta disciplina se ha desarrollado fuertemente en los últimos años teniendo aplicación en algunos campos como visión artificial, demostración de teoremas, procesamiento de información expresada mediante lenguajes humanos, etc.

Las redes neuronales al igual que la lógica difusa son más que otras formas de emular ciertas características propias de los humanos, como la capacidad de memorizar y de asociar hechos. El hombre es capaz de resolver estas situaciones acudiendo a la experiencia acumulada. Así, que una forma de aproximarse al problema consiste en la construcción de sistemas y que sean capaces de reproducir esta característica humana. Por lo tanto, la de Inteligencia Artificial (AI) en sus inicios [18], fueron utilizados en procesos mucho más sencillos que la creación de un cerebro artificial.

Es por ello que las redes neuronales ocupan un lugar importante en la solución de problemas complejos, tales como: clasificación de patrones [1], modelamiento y control [14] y procesamiento de señales [15]. La mayoría de las implementaciones de redes neuronales son simuladas en computador, subutilizando una máquina poderosa en la ejecución de una sola labor y restringiendo la movilidad del sistema.

Este documento también intentará mostrar una idea general del mundo de la lógica borrosa (en inglés, *fuzzy logic*), adentrándose en un mundo de información imprecisa que se tratará como un conjunto difuso. Estos sistemas pueden ser aplicados a problemas similares de las redes neuronales, resultando especialmente interesante en problemas no lineales o bien, no definidos.

Los algoritmos de ambos métodos se estarán aplicando para clasificar 208 patrones de un repositorio oficial[4], obtenidos por un sónar y de los cuales se establece que 111 corresponden a materiales metálicos y 97 a rocas. El conjunto de datos de entrada contiene señales obtenidas a partir de una variedad de ángulos de diferente aspecto, que abarcan desde los 90 grados para el metal hasta los 180 grados para la roca.

[1]Docente de la Fundación Universitaria Tecnológico Comfenalco y estudiante de Maestría en Ingeniería Electrónica, Cartagena, Colombia.
[2]Director del Grupo de Investigación en Control, Comunicaciones y Diseño Naval. Docente Investigador de la Escuela Naval "Almirante Padilla", Cartagena, Colombia.
[3]El Sonar o Sónar, (del inglés *SONAR*, acrónimo de *Sound Navigation And Ranging*, "navegación por sonido") es una técnica que usa la propagación del sonido bajo el agua (principalmente) para navegar, comunicarse o detectar objetos sumergidos (tomado de la RAE).

[4]http://archive.ics.uci.edu/ml/datasets/Connectionist+Bench+%28Sonar%2C+Mines+vs.+Rocks%29

978-1-4799-2471-4/13/$31.00 ©2013 IEEE

II. MÉTODOS DE IDENTIFICACIÓN

A. Estructura de Redes Neuronales Adaline

Las redes de neuronas artificiales (denominadas habitualmente como RNA[5] o en inglés como: "ANN"), tras obtener información, es capaz de aprender y de resolver problemas así como lo hace el ser humano.

El modelo de cada neurona incluye una función no lineal a diferencia del perceptrón, donde es la función escalón. Debido a la necesidad de que sea una función continua y derivable, es la función sigmoide, donde uk es la suma total de la actividad interna en la neurona k (la señal de entrada) e yk la salida que se produce en la neurona [26,27]. En la actualidad se llevan a cabo esfuerzos encaminados a unificar conceptos para la normalización y nomenclaturas, por comités de estandarización como la *Neural Network Society* del IEEE, o los de algunos investigadores [13].

El objetivo del entrenamiento de una RNA es conseguir que una aplicación determinada, para un conjunto de entradas produzca el conjunto de salidas deseadas o mínimamente consistentes. Existe una gran variedad de algoritmos de entrenamiento hoy en día.

La implementación de la RNA que se presenta en este artículo, es una simple aplicación de la neurona con arquitectura Adaline (de ADAptative LINear Element). El Adaline del profesor Bernie Widrow y su alumno Ted Hoff [29], desarrollado en 1960 correspondía originalmente a ADAptative LInear NEuron.

La diferencia entre el Adaline y el Perceptrón [24], estándar de McCulloch-Pitts [21], es que el Perceptrón solo tiene capacidad para clasificar, ya que utiliza una función umbral sobre la suma ponderada de las entradas, a diferencia del Adaline, que es capaz de estimar una salida real. Generalmente Adaline se compone de una sola capa de n neuronas (por tanto n valores de salida) con m entradas con las siguientes características:

A) Las m entradas representan un vector x de entrada que pertenece al espacio R^m. B) Por cada neurona, existe un vector w de pesos sinápticos que indican la fuerza de conexión entre los valores de entrada y la neurona. C) En la práctica representan la ponderación de cada entrada sobre la neurona. D) Una constante θ. E) La salida y de la neurona se representa por la función de activación, que se define como la ecuación (1):

$$y = \sum_{i=1}^{n} x_i * w_i + \theta \qquad \text{ecu. (1)}$$

La selección de una RNA se realiza en función de las características del problema a resolver. La mayoría de éstos se pueden clasificar en aplicaciones de Predicción, Clasificación, Asociación, Conceptualización, Filtrado y Optimización. Los tres primeros tipos de aplicaciones requieren un entrenamiento supervisado. En otros modelos se aprecian esquemas de una red neuronal multicapa [3, 8].

B. Estructura del modelo difuso borroso

En la construcción de modelos difusos [5, 6] debe tenerse en cuenta la selección y sintonización de diferentes parámetros, como son: la forma y distribución de funciones de pertenencia de las variables de entrada, la base de reglas, los operadores lógicos empleados, la forma y distribución de los consecuentes, etc. Esta cantidad de parámetros ha dificultado desarrollar una técnica única de modelación, especialmente en la identificación borrosa a partir de datos experimentales de entrada y salida.

Wang [28, 29], realizó una de las primeras propuestas para diseño automático de sistemas borrosos a partir de los datos, Sugeno y Yasukawa en 1993 [26,27], propusieron una metodología para identificación de parámetros de modelos borrosos empleando consecuentes tipo singletons, pero requiere de muchas reglas y presenta una pobre capacidad de descripción. La técnica más adecuada para la obtención de modelos borrosos han sido los algoritmos de agrupamiento borroso, destacándose los métodos de Bezdek el "*Fuzzy C-Means*" en 1987 [4] y el de Gustafson-Kessel en 1979 [17]. Años más adelante, en el 2000, Espinosa y Vandewalle [12], propusieron una metodología para extraer reglas a partir de los datos en un marco de integridad lingüística incluyendo algoritmos de fusión para agrupar conjuntos cuyos valores modales estén a una distancia muy cercana. Sala en 1998 [25], introdujo una novedosa técnica basada en el error de inferencia para aproximar funciones empleando partición suma 1 con conjuntos triangulares [23].

En el mismo año, Paiva y Dourado [22], presentaron un modelo generado por medio del entrenamiento de una red neuro-difusa implementado en dos fases: en la primera fase, se obtiene la estructura del modelo empleando un algoritmo clustering substractivo, lo cual permite extraer las reglas a partir de datos de entrada y salida; en la segunda fase, se realiza la sintonización de los parámetros del modelo mediante una red neuronal que emplea retropropagación, pero imponiendo restricciones en el ajuste de los parámetros y en la fusión de las funciones de pertenencia con el fin de garantizar la interpretabilidad del modelo resultante.

Joo y Lee [19,20], propusieron un aproximador universal para cualquier función continua en un conjunto compacto empleando un sistema borroso jerárquico en el que las salidas de las capas previas no son usadas en las partes *IF* de la reglas difusas sino en las partes *THEN*. Chen y Saif en 2005 [5], propusieron un sistema borroso novedoso que emplea bases de reglas dinámicas que cambian con las entradas y facilita su empleo tanto para modelación como para control.

La novedad del procedimiento propuesto en este documento radica en la consecución de modelos borrosos interpretables con alta precisión y bajo número de parámetros, debido que no es un método híbrido que no requiere de otras técnicas de inteligencia artificial para su entrenamiento.

La técnica presentada ha sido implementada en tareas de identificación de modelos dinámicos [11, 8], de clasificación [7], y en análisis y tratamiento de imágenes [9]. La metodología empleada es presentada en tres fases: en la primera, se utiliza el error de inferencia para generar un sistema borroso interpretable y, además, detectar clases o agrupamientos posibles en los datos; en la segunda, mediante la aplicación de mínimos cuadrados, para ajuste de consecuentes; en la tercera, se emplea el método para la clasificación de minas a partir de información recibida de un sonar.

III. DESCRIPCIÓN DEL PROBLEMA

El conjunto de datos tomados del supositorio para la implementación de los algoritmos de este artículo, son una contribución a la colección de referencia de Terry Sejnowski, que

[5] En inglés y francés se utiliza el acrónimo ANN (de Artificial Neural Networks) pero para referirse a este artículo se utilizará su equivalente castellano RNA.

ahora pertenecen al Instituto Salk y la Universidad de California en San Diego. El conjunto de datos fue desarrollado en colaboración con R. Paul Gorman del *Allied-Signal Technology Aerospace Center*. Los datos fueron utilizado por Gorman y Sejnowski en su estudio de la clasificación de las señales de sónar utilizando una red neuronal [16]. Los autores formaron una red RNA para diferenciar las señales del sónar rebotadas de un cilindro de metal y las que rebotaron de una piedra más o menos cilíndrica.

El archivo "sonar.mines" contiene 111 patrones obtenidos por el rebote del sónar de señales fuera de un cilindro de metal en varios ángulos y bajo diferentes condiciones. El archivo "sonar.rocks" contiene 97 patrones obtenidos de rocas bajo condiciones similares. La señal transmitida de un sónar es un chirrido de frecuencia modulada, aumentando la frecuencia. El conjunto de datos contiene señales obtenidas a partir de una variedad de ángulos de aspecto diferente, que abarca 90 grados para el cilindro y 180 grados para la roca. Cada patrón es un conjunto de 60 números en el rango de 0,0 a 1,0. Cada número representa la energía dentro de una banda de frecuencia particular, integrada a lo largo un cierto período de tiempo. La abertura de integración para las frecuencias más altas ocurre en tiempos más largos, ya que estas frecuencias se transmiten después durante el chirrido. La etiqueta asociada a cada registro de patrones contiene la letra "CR" si el objeto es una roca (ejemplo CR036) y "CM" si es una mina o cilindro metálico (ejemplo CM078). Los números en la etiquetas están en orden creciente de ángulo de aspecto, pero estos no indican el ángulo directamente.

IV. Materiales y Métodos

Como herramientas para la investigación en el desarrollo de sistemas de control y procesamiento de señal, se han utilizado frecuentemente entornos de base matemática que facilitan la evaluación de diversas técnicas, sin obligar inicialmente al desarrollo de programas específicos, haciendo uso de paquetes matemáticos diversos incluidos en estos sistemas matemáticos de propósito general. Así, dado un sistema de desarrollo matemático de propósito general, se suelen suministrar por separado paquetes o conjuntos de utilidades específicos para tratamiento de señal, control de sistemas, trabajo con redes neuronales, lógica borrosa, etc. La mayoría de estos sistemas matemáticos tienen su origen en entornos textuales, aunque en sus versiones actuales se han adaptado a los entornos gráficos disponibles hoy en día, tanto en su interacción con el usuario, como en la presentación de los resultados de las aplicaciones desarrolladas.

El conjunto combinado de los 208 casos se dividieron en datos que son de entrenamiento y datos para la validación o prueba (clasificación indicada en los archivos originales mediante "*"), separados de esta forma: de 97 patrones de rocas 55 son de entrenamiento y 42 de prueba. Y de 111 patrones de minas 49 son de entrenamiento y 62 de prueba. De esta forma y de manera general, para este artículo, para el entrenamiento de los algoritmos se tomarán 40 datos de rocas y 40 de minas, con salidas de 1 y 0 respectivamente; de la misma forma para la validación de los algoritmos se tomaran 9 datos de rocas y 9 de minas indicando las salidas con la misma clasificación de 1´s y 0´s.

A. Implementación de la RNA Adaline

La arquitectura de Adaline utiliza un dispositivo lógico que realiza una suma lineal de las entradas y genera una función umbral para el resultado de dicha suma. Widrow y Hoff propusieron un método más eficiente computacionalmente denominado LMS (least mean square algorithm) para determinar parámetros del Adaline (1962), que se basa en el método de descenso de gradiente para minimizar la diferencia entre la salida deseada y la actual (entrenamiento supervisado). La arquitectura Madaline (Multilayer Adaline) creada también por Widrow presenta una configuración constituida por dos o más unidades Adaline [29].

La Fig. (1a) muestra una Adaline básica. La unidad procesadora representada por un círculo con el símbolo sumatorio implementa una función umbral. Las conexiones de cada una de las entradas tienen asociadas un valor de ponderación llamado también peso w_i.

El mecanismo de ajuste de los pesos representado en la Fig. (1b), consiste en utilizar la diferencia entre el valor de la salida y el valor esperado. La unidad procesadora actúa como un sumador y después realiza la función umbral según la ecuación (2).

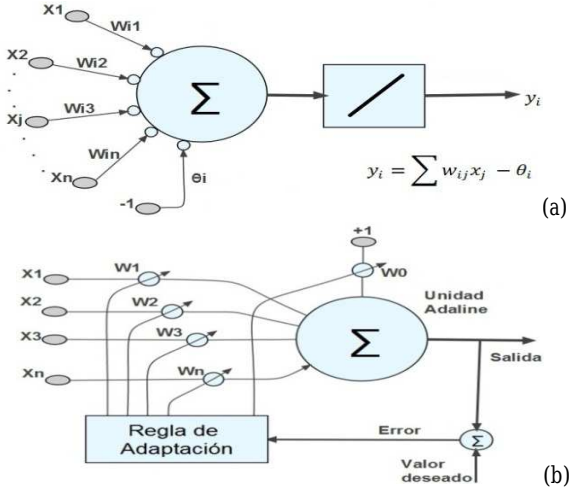

Fig. 1. Red Adaline.

$$x_j = \begin{cases} 1 \text{ si } S = \Sigma i\, x_i\, w_i \geq 0 \\ -1 \text{ si } S = \Sigma i\, x_i\, w_i < 0 \end{cases} \quad \text{ecu. (2)}$$

La salida de la unidad Adaline es ±1 a diferencia de la arquitectura del Perceptron que sólo permite los valores 0y 1. El entrenamiento se realiza presentando repetidamente una serie de parejas de entradas y salidas. El objetivo de la Adaline durante el proceso de la adaptación es producir la salida deseada como propia suya. La regla de aprendizaje en la arquitectura de la Adaline es la regla de Widrow-Hoff expresada en la ecuación (3)[32, 33].

$$\Delta w_i = \eta\, x_i\, (t * y) \quad \text{ecu. (3)}$$

Siendo η la constante de aprendizaje, x_i la salida de la unidad i, t la salida deseada y por último y la salida de la unidad Adaline. No obstante la variante de esta regla más utilizada considera el valor de la suma ponderada S en vez del valor de la salida de la unidad Adaline. A diferencia de una RNA perceptrón, a la hora de modificar los pesos durante el entrenamiento, el Adaline tiene en cuenta el grado de corrección de la salida estimada respecto a la deseada. Esto se consigue mediante la aplicación de la regla Delta, y que se define, para un patrón de entrada x^p con una salida estimada y^p y una salida deseada d^p, como $|d^p + y^p|$.

Dado que el objetivo del Adaline es poder estimar de la manera más exacta la salida (conseguir una salida exacta es prácticamente imposible en la mayoría de los casos), se busca minimizar la desviación de la red para todos los patrones de entrada, eligiendo una

medida del error global. Normalmente se utiliza el error cuadrático medio de acuerdo la ecuación (4).

$$E = \frac{1}{2}\sum_{p=1}^{m}(d^p - y^p)^2 \qquad \text{ecu. (4)}$$

La manera de reducir este error global es ir modificando los valores de los pasos al procesar cada entrada, de forma iterativa, mediante la regla del descenso del gradiente. Suponiendo que se tiene una constante de aprendizaje α de la ecuación (5).

$$\Delta_p \mathrm{w_j} = -\alpha \frac{\partial E^p}{\partial \mathrm{w_j}} \qquad \text{ecu. (5)}$$

Si se opera con la derivada, queda la ecuación (6).

$$\Delta_p \mathrm{w_j} = -\alpha(d^p - y^p) * \mathrm{x_j} \qquad \text{ecu. (6)}$$

1) Descripción del algoritmo RNA: la ecuación (6) será la expresión que se utilizará por cada entrada para modificar los pesos, y teniendo una constante de aprendizaje de α = 0.001, se crea una red lineal adaptativa mediante la función NEWLIN (PR,S,ID,LR) de donde:

- PR: matriz Rx2 de los valores máximos y mínimos para R elementos de entrada (60).
- S: número de elementos de salida (1).
- ID: vector de retardo, (default = [0]).
- LR: tasa de aprendizaje, (0.01).

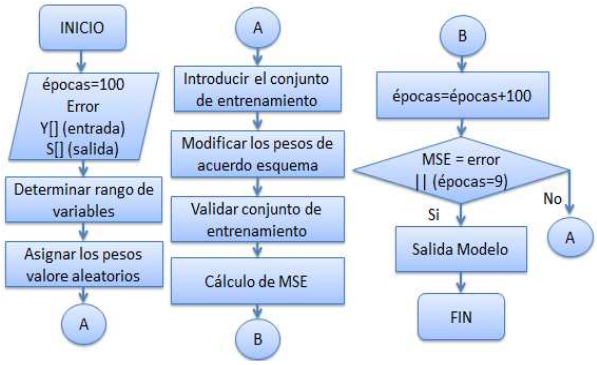

Fig. 2. Diagrama de Flujo Modelo RNA.

Se relacionarán los datos del error cuadrático medio obtenidos mediante la función MSE (*Mean squared error performance function*) y el porcentaje de aciertos de la validación para 9 patrones de rocas y metales respectivamente. Aplicando al algoritmo una búsqueda de un error de 0.1 (net.trainParam.goal = 0.1) y aumentando en centenares las épocas para la iteración (net.trainParam.epochs = de 100 a 900). El algoritmo de programación se demuestra en la Fig. (2).

B. Implementación del sistema borroso

En este parte se expondrá las diversas maneras disponibles para realizar la identificación d patrones mediante lógica borrosa. Como en el caso de redes neuronales, un sistema borroso podrá implementarse como programa ejecutable por un microprocesador convencional (o microcontrolador), o podrá realizarse en hardware específico. La gran diferencia con las redes neuronales es que un sistema borroso precisa en general de recursos de cálculo relativamente reducidos, por lo que muchas veces podrá implementarse como programa ejecutado en un sencillo microcontrolador de 8 bits.

Inicialmente se considera entornos de desarrollo de sistemas borrosos, mostrando el por qué son necesarios, así como algunos de los actualmente disponibles, que serán descritos con cierto detenimiento. Después el lector podrá ver los distintos métodos de realización hardware de sistemas borrosos, incluyendo los denominados aceleradores de procesamiento. Para el desarrollo de este algoritmo de aproximación funcional, se ha pensado en alcanzar altos niveles de precisión e interpretabilidad, la estructura del modelo se ha fundamentado en los siguientes criterios:

Funciones de Pertenencia: se emplearon funciones de pertenencia triangulares normales, ya que estas permiten la reconstrucción del valor lingüístico en el mismo valor numérico, luego de aplicar un método de *defuzzyficación* [23]. Se escogió solapamiento desde 0.04 hasta 0.005 para asegurarse que los soportes de los conjuntos difusos sean diferentes. Para el caso de las variables de salida se emplearon conjuntos difusos tipo singleton.

Distribución de las Funciones de Pertenencia: la distribución de los conjuntos difusos triangulares de las variables de entrada estará dada de forma simétrica en cada universo, buscando así que cada elemento del universo de discurso de cada variable pertenezca al menos a un conjunto difuso. Operadores: se utilizó operador tipo OWA o promedio ponderado para la combinación de los antecedentes de las reglas. Este operador permite promediar los grados de pertenencia de cada antecedente evaluado [2].

Método de Inferencia: el método esta dado por la ecuación (7)

$$f(x^{(i)}) = \frac{\sum_{j=1}^{L} y^{-1} m_j(x^{(i)})}{\sum_{j=1}^{L} m_j(x^{(i)})} \qquad \text{ecu. (7)}$$

Donde la ecuación (8),

$$m_j(x^{(i)}) = u_{A_1},\ (x_1{}^{(i)})u_{A_2}, (x_2{}^{(i)}) \ldots\ldots u_{A_n}, (x_n{}^{(i)}) \qquad \text{ecu. (8)}$$

es el grado de cumplimiento de los antecedentes de la j-ésima regla de un sistema difuso tipo Sugeno, y-j es el valor del singleton correspondiente a la regla j[26]. El denominador siempre arroja un valor igual a 1 cuando se trata de particiones suma 1, el cual es el caso del método de aproximación propuesto en este artículo.

1) Descripción del algoritmo difuso: el algoritmo diseñado para la generación de sistemas borrosos interpretables a partir de los datos, los cuales se organizan como una colección de datos experimentales de entrada y salida $\{\boldsymbol{Xk^{(i)}}, \boldsymbol{Y^{(i)}}\}$ con $\boldsymbol{i = 1\ldots.N}$; k=1…p, donde $\boldsymbol{Xk^{(i)}}$ es el vector de entrada p-dimensional $\boldsymbol{X1^{(i)}, X2^{(i)}\ldots.Xp^{(i)}}$ e $\boldsymbol{Y^{(i)}}$ es el vector unidireccional de salida.

En el algoritmo inicialmente se definen las variables de entrada y se determinan los rangos de cada variable. La partición del universo de cada variable de entrada se hace empleando funciones de pertenencia triangulares con solapamiento en 0.005 hasta 0,04, iniciando con dos funciones de pertenencia que tienen sus valores modales (vértices del triángulo) en los valores mínimo y máximo de la variable respectiva, como muestra la Fig. (3). Para no afectar la interpretabilidad del sistema difuso se ha limitado el número de

funciones de pertenencia a un máximo de 10 por cada variable [15].

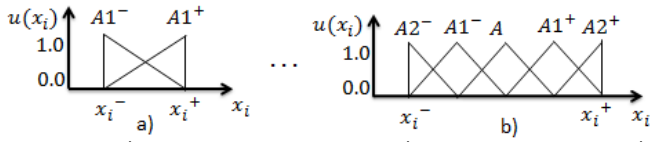

Fig. 3. Partición triangular suma 1. a) Partición inicial, n =2; b) Partición con n = 5

Por cada función de pertenencia habrá un consecuente tipo singleton; es decir, el número máximo de reglas generadas estará dado por la multiplicación entre número de variables de entrada y el número pertenencia, lo cual es significativamente menor al máximo de reglas generadas en los sistemas difusos convencionales. Para el cálculo de los consecuentes tipos singleton, se procede de la siguiente manera:

La ecuación (7) se plantea como se muestra en la ecuación (9)

$$f(x^{(i)}) = \sum_{j=1}^{L} w_j(x^{(i)}) y^{-1} \qquad \text{ecu. (9)}$$

Donde

$$w_j\left(x^{(i)}\right) = \frac{m_j(x^{(i)})}{\sum_{j=1}^{L} m_j(x^{(i)})} = w_j^i \qquad \text{ecu. (9)}$$

La ecuación (9) puede expresarse en forma matricial como $\boldsymbol{Y = W\theta + E}$, donde Y representa los valores de salida reales, $\boldsymbol{W\theta}$ representa la salida del modelo borroso, siendo $\boldsymbol{W}$ la matriz de grados de pertenencia obtenida de la ecuación (9) y $\boldsymbol{\theta}$ el vector de consecuentes, y $\boldsymbol{E}$ es el error de aproximación que debe ser minimizado. El resultado se presenta mediante la matriz de la ecuación (10).

$$\underbrace{\begin{bmatrix} y^1 \\ y^2 \\ \vdots \\ y^L \end{bmatrix}}_{Y} = \underbrace{\begin{bmatrix} w_1^1 & w_2^1 & \vdots & w_L^1 \\ w_1^2 & w_2^2 & \dots & w_L^2 \\ \vdots & \vdots & \ddots & \vdots \\ w_1^n & w_2^n & \dots & w_L^n \end{bmatrix}}_{W} \underbrace{\begin{bmatrix} y^{-1} \\ y^{-2} \\ \vdots \\ y^{-L} \end{bmatrix}}_{\theta} + \underbrace{\begin{bmatrix} e_1 \\ e_2 \\ \vdots \\ e_n \end{bmatrix}}_{E} \qquad \text{ecu. (10)}$$

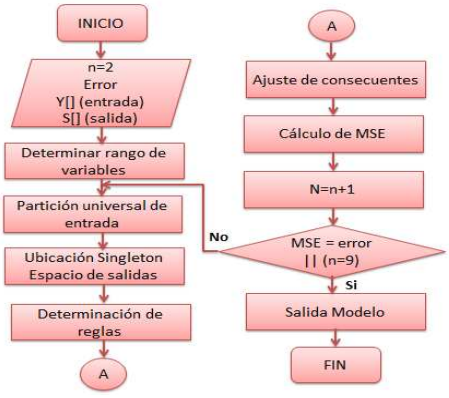

Fig. 4. Diagrama de Flujo Modelo Difuso.

El vector de consecuentes θ es calculado empleando mínimos cuadrados recursivos [8]. Por cada iteración en el proceso de entrenamiento, se compara la salida del modelo, dada por la ecuación (9), con la salida real y se calcula una métrica del error MSE (error cuadrático medio).

El algoritmo se detiene cuando se ha alcanzado una métrica de error menor a la requerida por el usuario o cuando el número de conjuntos borrosos por variable de entrada es mayor a 9. De otra manera se debe incrementar en 1 el número n de conjuntos de la variable de entrada y se procede con otra iteración. En la Fig. (4) se describe paso a paso el algoritmo a través de un diagrama de flujo.

V. Resultados

Para el entrenamiento se tomaron los primeros 40 datos y para la validación los siguientes 9 datos por cada materia respectivamente, sacados del archivo **RocasVsMetales_Entrena.txt**.

Del entrenamiento se obtuvo ambos modelos que se aproximan a esa función, además se validaron con los mismos patrones obteniendo los valores de error cuadrático medio MSE en el entrenamiento y en la validación para ambos modelos. La tabla (1a) muestra las iteraciones para el modelo RNA Adaline al cual se le fueron aumentando las épocas de iteración de 100 a 900. La tabla (1b) muestra los resultados de los diferentes tipos de iteraciones para el modelo borroso aumentando sus conjuntos en cada modelo.

Las mejores aproximaciones y el mayor porcentaje de aciertos se obtuvieron en las 900 épocas de iteración de la RNA Adaline y en los 10 conjuntos del modelo difuso respectivamente.

ÉPOCAS	MSE ENTRENAMIENTO	MSE VALIDACIÓN	% ACIERTO
100	0,2031	0,2354	55,5556
200	0,1869	0,218	61,1111
300	0,1775	0,2118	61,1111
400	0,1706	0,2098	61,1111
500	0,165	0,2098	66,6667
600	0,1603	0,2108	66,6667
700	0,1562	0,2124	66,6667
800	0,1526	0,2143	72,2222
900	0,1494	0,2165	72,2222

CONJUNTO	MSE ENTRENAMIENTO	MSE VALIDACIÓN	% ACIERTO
2	0.0916	0,2958	42,5532
3	0,0309	0,335	59,5745
4	0,0187	0,3384	61,7021
5	0,0087	0,3481	57,4468
6	0,0037	0,3128	63,8298
7	7,9046e-004	0,1969	77,7778
8	6,2962e-004	0,1898	77,7778
9	4,9268e-004	0,1545	83,3333
10	3,3893e-004	0,1441	83,3333

Tabla 1. MSE y porcentaje de aciertos. a) Modelo RNA Adaline b) Modelo difuso borroso

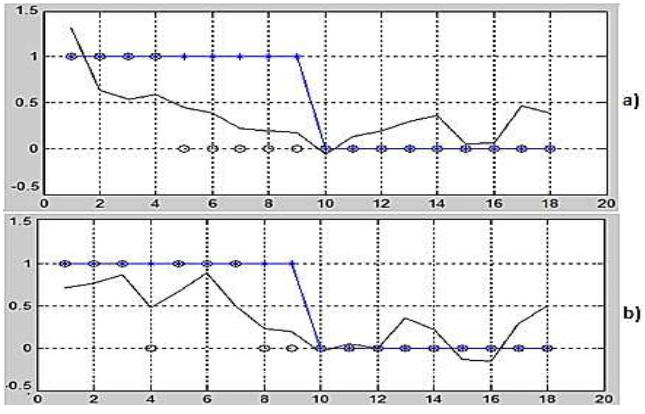

Fig. 5. Mejor aproximación de la Funciones. a) Modelos RNA. b) Modelo difuso borroso.

Las gráficas de la Fig. (5a y 5b) muestran las mejores aproximaciones de los algoritmos hacia el conjunto de los 18 patrones de validación.

La validación de los algoritmos se relaciona en las tablas (2a y 2b); estas tablas establecen los resultados de aproximación para los dos tipos de materiales (metal y roca) con la misma cantidad de pruebas en ambos algoritmos. El tiempo de duración en arrojar las respuestas al entrenamiento y validación fue un factor importante ya que la RNA duró un máximo de 12 segundos en relación a los 19 segundos del modelo borroso.

PATRÓN	TIPO	RED RNA	MODELO DIFUSO
*CR302	ROCA	ROCA	ROCA
*CR306	ROCA	ROCA	ROCA
*CR307	ROCA	ROCA	ROCA
*CR312	ROCA	ROCA	METAL
*CR314	ROCA	METAL	ROCA
*CR316	ROCA	METAL	ROCA
*CR317	ROCA	METAL	ROCA
*CR318	ROCA	METAL	ROCA
*CR321	ROCA	METAL	METAL
% ACIERTO		**44,4**	**77,8**

PATRÓN	TIPO	RED RNA	MODELO DIFUSO
*CM505	METAL	METAL	METAL
*CM507	METAL	METAL	METAL
*CM512	METAL	METAL	METAL
*CM514	METAL	METAL	METAL
*CM516	METAL	METAL	METAL
*CM521	METAL	METAL	METAL
*CM522	METAL	METAL	METAL
*CM542	METAL	METAL	METAL
*CM544	METAL	METAL	METAL
% ACIERTO		**100,0**	**100,0**

Tabla 2. Aciertos a los patrones. a) Tipo Rocas b) Tipo Metales

REFERENCIAS

[1] Arias, E. y Torres, H. *Sistemas inteligentes en un chip utilizando FPGAs: aplicaciones a la visión por computadora*. Cholula, México: X Congreso Internacional de Electrónica, Comunicaciones y Computadoras, pp. 37-41, 2000.

[2] Babuska, R. *Fuzzy Modeling and Identification*, PhD. dissertation. Delft University of Technology, Delft, The Netherlands, 1996.

[3] Baratta, Bo, Caviglia, Diotalevi y Valle. *Microelectronic Implementation of ANN*. 5th Electronic Devices and Systems Intl Conf.. Brno, Rep. Checa, 1998.

[4] Bezdek, J. C., *Pattern recognition with Fuzzy Objective Function Algorithms*. Ed. Plenum Press, 1987.

[5] Chen, W., and Saif, M. *A Novel Fuzzy System with Dynamic Rule Base, IEEE Trans. Fuzzy Systems*, Vol.13, (5), pp.569-582, 2005.

[6] Contreras, J. "Generating Singleton Fuzzy Models from Data with Interpretable Partition". Advanced Materials Research. Vol. 629, pp. 784-791. 2012.

[7] Contreras J., Acuña O. *Generating Dynamic Fuzzy Models for Prediction Problems*. 28th North American Fuzzy Information Processing Society Annual Conference, Cincinnati, Ohio, USA, ISBN: 978-1-4244-4575-2, 2009.

[8] Contreras, J., Misa, R., Paz, J. "*Obtención de modelos borrosos interpretables de procesos dinámicos*" Revista Iberoamericana de automática e Informática Industrial-RIAI. Vol.5, pp. 70-77, 2008.

[9] Contreras J., Muñoz D. J., Cuadrado W., Díaz V., Delgado G., Archbold G. *Automatic Ship Hull Inspection using fuzzy logic*. Applied Imagery Pattern Recognition AIPR 2012. Washington, USA, 2012.

[10] Díez J. L., Navarro J. L., Sala A. *Algoritmos de Agrupamiento en la Identificación de Modelos Borrosos*. RIAI: Revista Iberoamericana de Automática e Informática Industrial, vol. 1, pp. 32-41, 2004.

[11] Efthymiou, W. *An Asynchronous, Iterative Implementation of the Original Booth Multiplication Algorithm*. 10th IEEE International Symposium on Asynchronous Circuits and Systems (ASYNC'04) pp. 207-215, 2004.

[12] Espinosa, J., Vandewalle, J., Wertz, V. *Fuzzy Logic, Identification and Predictive Control*. Springer, USA, 2005.

[13] Fiesler, E. *Neural network clasification and fomalization*. En: *Computer Standards and Interfaces*, vol. 16, special issue on Neural Networks Standards, J. Fulcher (edt.) Elseiver, 1994.

[14] Fontalba, González, Sandoval. *Realización de una Red Neuronal en una FPGA para Detección de Velocidad*. España: XI Congreso de Diseño de Circuitos Integrados, Sitges, 1996.

[15] Fu, L. Neural Networks in Computer Intelligence. S.d. McGraw-Hill, 1994.

[16] Gorman, R. P., and Sejnowski, T. J. "*Analysis of Hidden Units in a Layered Network Trained to Classify Sonar Targets*" *in Neural Networks*, Vol. 1, pp. 75-89, 1988.

[17] Gustafson, E. E., Kessel, W. C. *Fuzzy Clustering with a Fuzzy Covariance Matrix*. In Proceedings of the IEEE CDC, pp. 503 – 516, 1979.

[18] Hèrault, J., Jutten, C. *La memoria de las redes neuromiméticas. Mundo científico*, 150, 14, pp. 884-891, 1994.

[19] Joo, M.G., and Lee, J.S. *Universal Approximation by Hierarchical Fuzzy Systems with Constraints on the Fuzzy Rules*. Fuzzy Sets and Systems, vol. 130(2), pp 175-188, 2002.

[20] Joo, M.G., and Lee, J.S.. *A Class of Hierarchical Fuzzy Systems with Constraints on the Fuzzy Rules*. IEEE Trans. Fuzzy Systems, **vol. 13**(2), pp.194-203, 2005.

[21] McCulloch, W. S., Pitts, W. *A logical calculus of the ideas immanent in nervios activity*. Bulletin of Mathematical Biophysics, 5, pp. 115-133, 1943.

[22] Paiva, R. P., Dourado, A. *Interpretability and Learning in Neuro-Fuzzy Systems, Fuzzy Sets and System*. vol.147(1), pp. 17-38, 2004.

[23] Pedrycz, W. (1994) *Why Triangular Membership Functions, IEEE Trans. Fuzzy Sets and System*. vol. 64(1), pp.21-30, 1994.

[24] Perceptrón simple y Adaline, Redes de Neuronas Artificiales, UC3M, RAI, 2012.

[25] Sala, A. *Validación y Aproximación Funcional en Sistemas de Control Basados en Lógica Borrosa*. Universidad Politécnica de Valencia. Tesis Doctoral, España, 1998.

[26] Sugeno, M., Yasukawa, T. *A Fuzzy Logic Based Approach to Qualitative Modeling*. Transactions on Fuzzy Systems, vol. 1(1), pp. 7-31, 1993.

[27] Wang, L-X, Mendel, J.M. *Generating Fuzzy Rules by Learning from Examples*. IEEE Transactions on Systems Man and Cybernetics, vol. 22(6), pp. 1414-1427, 1992.

[28] Wang, L-X, Langari, R. *Building Sugeno- Type Models Using Fuzzy* [] *Discretization and Orthogonal Parameter Estimation Techniques*. IEEE Transactions on Fuzzy ystems, **vol. 3**(4), pp. 454-458, 1995.

[29] Widrow, B., Hoff, M. E. *Adaptative switching circuits*. 1960 IRE WESCON Convention Record, 4, pp. 96-104, 1960.