Revista Iberoamericana de Automática e Informática Industrial

ISSN: 1697-7912. Vol. 5, Núm. 1, Enero 2008, pp. 115-123 www. revista-riai.org

ALGORITMO DE ENTRENAMIENTO ÓPTIMO PARA DISEÑAR UNA MEMORIA ASOCIATIVA DE DIAGNÓSTICO DE FALLAS

José A. Ruz Hernández * **Edgar N. Sánchez** **
Dionisio A. Suárez ***

* *Universidad Autónoma del Carmen, Av. 56 No. 4, Col. Aviación, C.P. 24180, Cd. del Carmen, Campeche, MEXICO, e-mail: jruz@pampano.unacar.mx*
** *CINVESTAV, Unidad Guadalajara, Apartado Postal 31-430, Plaza la Luna, C.P. 45091, Guadalajara, Jalisco, MEXICO, e-mail: sanchez@gdl.cinvestav.mx*
*** *Instituto de Investigaciones Eléctricas, Calle Reforma No. 113, Col. Palmira, C.P. 62490, Cuernavaca, Morelos, MEXICO, e-mail: suarez@iie.org.mx*

Resumen: En este artículo, los autores presentan un nuevo enfoque de síntesis para entrenar memorias asociativas implementadas con redes neuronales recurrentes. Los pesos de la red recurrente se determinan como la solución óptima de la combinación lineal de vectores soporte. El algoritmo de entrenamiento propuesto maximiza el margen entre los patrones de entrenamiento y la superficie de decisión. El problema de diseño considera: 1) la obtención de los pesos por medio del algoritmo de hiperplano óptimo utilizado para máquinas de vector soporte y 2) la obtención de las condiciones para reducir el número total de memorias espurias. El nuevo algoritmo desarrollado se utiliza para diseñar una memoria asociativa que diagnostique fallas en centrales termoeléctricas.

Palabras clave: Memoria asociativa, red neuronal recurrente, máquinas de vector soporte, hiperplano óptimo, detección y diagnóstico de fallas, central termoeléctrica.

1. INTRODUCCIÓN

Las redes neuronales recurrentes de una sola capa completamente conectadas son una clase especial de sistemas dinámicos no lineales que pueden presentar puntos de equilibrio asintóticamente estables (memorias estables), así como puntos de equilibrio inestables. La implementación de memorias asociativas por medio de redes neuronales recurrentes se presenta en (Liu and Lu, 1997). El objetivo de la memoria asociativa consiste en almacenar un conjunto de patrones deseados como vectores de memoria estable de tal forma que puedan recuperarse cuando el patrón de entrada contiene suficiente información acerca del patrón almacenado. En la práctica, los patrones de memoria deseados se representan comúnmente por vectores bipolares o vectores binarios.

Existen diversos métodos disponibles en la literatura que resuelven el problema de síntesis de redes neuronales recurrentes para el diseño de memorias asociativas. Entre ellos podemos mencionar al método del producto externo, la regla de aprendizaje por proyección y el método de estructura de valores propios (Liu and Michel, 1994). El método del producto externo requiere que los patrones deseados sean mutuamente ortogonales para que se almacenen en la red. La regla de aprendizaje por proyección no requiere que los patrones que se desean almacenar en la red sean ortogonales; sin embargo, este método no garantiza que un punto de equilibrio que corresponde a una memoria deseada sea asintóticamente estable. El método de estructura de valores propios es más efectivo ya que garantiza el almacenamiento de un conjunto de patrones bipolares como memorias estables que no requieren ser mutuamente ortogonales y que corresponden a puntos de equilibrio asintóticamente estable de una red neuronal. Así mismo, el método de estructura de valores propios se generaliza al caso de síntesis de redes neuronales recurrentes con restricciones predeterminadas en la estructura de interconexión (Liu and Michel, 1996). Estos métodos de síntesis utilizan un conjunto de desigualdades lineales para el diseño de redes neuronales recurrentes. En los métodos de diseño discutidos en (Hassoun, 1993) y (Hassoun and Youssef, 1990), un conjunto de desigualdades lineales se resuelven con el método de Hop-Kashyap utilizando el criterio del error cuadrático medio (Li *et al.*, 1988). Una contribución importante a la síntesis de redes neuronales recurrentes para memorias asociativas se propone en (Liu and Lu, 1997); el enfoque propuesto está basado en el algoritmo de entrenamiento tipo perceptrón. Este tipo de entrenamiento resulta convergente sin requerir restricciones no hay restricciones en la matriz de conexión de la red neuronal recurrente. Así mismo, la contribución establece las propiedades de la red neuronal recurrente con restricciones en los elementos de la diagonal de la matriz de conexión para reducir el número total de memorias espurias. Sin embargo, este algoritmo no garantiza que el margen entre los patrones de entrenamiento y la superficie de decisión sea máximo.

El diseño de memorias asociativas basado en nuevas redes neuronales atractoras que reducen el número de memorias espurias se describe en (Chartier and Proulx, 2006). Además, el diseño de memorias asociativas tipo Hopfield asimétricas con estabilidad de Hamming de alto orden se propone en (Lee and Chuang, 2006). Este último enfoque incrementa el tamaño de la región de atracción para cada vector característico pero incrementa ligeramente el número de memorias espurias.

Debido a su alta capacidad de generalización, las máquinas de vector soporte (MVS) han cobrado importancia para el reconocimiento de patrones, máquinas de aprendizaje, redes neuronales y otras aplicaciones. El aprendizaje con MVS conduce a un problema de programación cuadrática, que se resuelve con varias técnicas (Boser *et al.*, 1992) y (Cortes and Vapnik, 1995).

Este artículo propone un nuevo algoritmo de entrenamiento óptimo para diseñar memorias asociativas implementadas con redes neuronales recurrentes. El problema de diseño considera: 1) la obtención de los pesos mediante el algoritmo de hiperplano óptimo utilizado para MVS y 2) la obtención de las condiciones necesarias para reducir el número total de memorias espurias. El algoritmo utiliza el enfoque de síntesis basado en las MVS para determinar los pesos de la red neuronal recurrente mediante una solución que se expresa como la combinación lineal de vectores soporte. Además, a diferencia de los algoritmos existentes maximiza el margen entre los patrones de entrenamiento y la superficie de decisión. El algoritmo propuesto se utiliza en un esquema basado en redes neuronales recurrentes para la detección y diagnóstico de fallas en centrales termoeléctricas.

2. PRELIMINARES

Esta sección introduce preliminares útiles acerca de memorias asociativas implementadas con redes neuronales recurrentes y el problema de síntesis correspondiente. Los conceptos fundamentales de MVS y el problema de encontrar el hiperplano óptimo también se describen.

La red neuronal recurrente considerada está dada por

$$\begin{aligned} \dot{x} &= -Ax + Tsat(x) + I \\ y &= sat(x) \end{aligned} \tag{1}$$

donde $x \in R^n$ es el vector de estado, $\dot{x}$ denota la derivada de x con respecto al tiempo t, $y \in D^n = \{x \in R^n : -1 \leq x_i \leq 1, i = 1, ..., n\}$ es el vector de salida, $A = diag\,[a_1, ..., a_n]$ con $a_i > 0$ para $i = 1, ..., n, T \in R^{nxn}$ es la matriz de conexión con elementos $T_{i,j} \in R$, $I = [I_1, ..., I_n]^T$ es un vector de polarización y $sat(x) = sat\,[sat(x_1), ..., sat(x_n)]^T$ representa la función de activación, donde

$$sat(x_i) = \begin{cases} 1, x_i > 1 \\ x_i, -1 \leq x_i \leq 1 \\ -1, x_i < -1 \end{cases}$$

Aquí se supone que los estados iniciales de (1) satisfacen $|x_i(0)| \leq 1$ para $i = 1, ..., n$. El sistema (1) es una variante del modelo de Hopfield con la función de activación $sat(\cdot)$.

2.1 Problema de síntesis

Los siguientes resultados tomados de (Liu and Lu, 1997)se incluyen en esta sección. Un vector se llama vector de memoria estable o simplemente memoria del sistema (1) si $\alpha = sat(\beta)$ y si β es un punto de equilibrio asintóticamente estable de (1). En el siguiente lema, B^n se define como un conjunto n-dimensional de vectores bipolares $B^n = \{x \in R^n : x_i = -1$ o $1, i = 1, 2, \ldots, n\}$. Para $\alpha = [\alpha_1, \alpha_2, \ldots, \alpha_n]^T \in B^n$ se define $C(\alpha) = \{x \in R^n : x_i\alpha_i > 1, i = 1, 2, \ldots, n\}$

Lema 2.1 Si $\alpha \in B^n$ y si

$$\beta = A^{-1}(T\alpha + I) \in C(\alpha) \tag{2}$$

entonces (α, β) es un par de vectores de memoria estable y un punto de equilibrio asintóticamente estable de (1).

La prueba de este lema se incluye en (Liu and Lu, 1997). El siguiente problema de síntesis concierne al diseño de (1) para memorias asociativas.

Problema de Síntesis: Dados m vectores α^1,α^2,...,α^m en el conjunto n-dimensional de vectores bipolares, B^n, seleccione $\{A, T, I\}$ de tal modo que:

1. α^1, α^2,..., α^m sean vectores de memoria estable del sistema (1);

2. El número total de vectores de memoria espurios (i.e., vectores de memoria que no se desean) sea lo más pequeño posible, y el dominio (o región) de atracción de cada vector de memoria deseado sea lo más grande posible.

El punto 1 del problema de síntesis se garantiza mediante la selección de $\{A, T, I\}$ de manera que cada patrón satisface la condición (2) del Lemma 2.1. El punto 2 se asegura parcialmente mediante restricciones en los elementos de la diagonal de la matriz de conexión T. Para resolver el problema de síntesis, se requiere determinar $\{A, T, I\}$ de

$$A^{-1}(T\alpha^k + I) \in C(\alpha^k) \tag{3}$$

para $k = 1, 2, ..., m$. La condición (3) puede escribirse de manera equivalente como

$$\begin{aligned} T_i\alpha^k + I_i &> a_i \text{ si } \alpha_i^k = 1 \\ T_i\alpha^k + I_i &< -a_i \text{ si } \alpha_i^k = -1 \end{aligned} \tag{4}$$

para $k = 1, 2, ..., m$ y para $i = 1, 2, ..., n$; donde T_i representa la i-ésima fila de T, I_i denota el i-ésimo elemento de I, y α_i^k es la i-ésima entrada de α^k.

2.2 Máquinas de vector soporte y el algoritmo del hiperplano óptimo

La máquina de vector soporte es un mecanismo de aprendizaje para problemas de clasificación entre dos clases. La máquina implementa la siguiente idea: los vectores de entrada se transforman, por medio de funciones no lineales, a un espacio caracterísitico de alta dimensión. En este espacio característico, se construye una superficie de decisión lineal. Las propiedades especiales de la superficie de decisión aseguran una gran capacidad de generalización (Cortes and Vapnik, 1995). El problema consiste en determinar un hiperplano de separación que generalize bien (ver Figura 1). El problema fue resuelto en 1965, para el caso de clases separables (Vapnik, 1982). Un hiperplano óptimo se define como una función lineal con margen máximo entre los vectores de las dos clases. Para construir el hiperplano óptimo de separación solamente se toman en cuenta una pequeña cantidad de datos de entrenamiento (llamados vectores soporte) que determinan el margen máximo. Si los vectores de entrenamiento se separan sin errores por un hiperplano óptimo, entonces el valor esperado de la probabilidad de que ocurra un error en un conjunto de prueba está acotado por el radio entre el valor esperado del numero de vectores soporte y el número de vectores de entrenamiento:

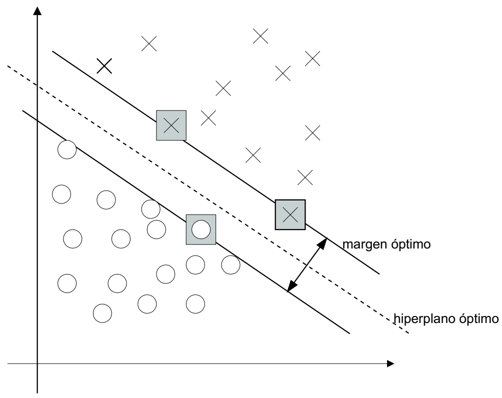

Figura 1. Hiperplano óptimo.

$$E[pr(error)] \leq \frac{E[\text{No. de vectores soporte}]}{\text{No. de vectores de entrenamiento}} \tag{5}$$

Sea

$$W_0 \cdot z + b_0 = 0 \tag{6}$$

el hiperplano óptimo en el espacio característico, donde $W_0 \cdot z$ es el producto interno entre los pesos y el vector z en este espacio. Los pesos W_0 para el hiperplano óptimo en el espacio caracterísitico pueden escribirse como la combinación lineal de vectores soporte

$$W_0 = \sum_{\text{vectores soporte}} \lambda_i z_i \tag{7}$$

La función de decisión lineal $I(z)$, en el espacio característico es de la forma

$$I(z) = sign\left(\sum \lambda_i z_i \cdot z + b_0\right) \tag{8}$$

donde $z_i \cdot z$ es el producto interno entre los vectores soporte z_i y el vector z en el espacio característico.

2.3 Algoritmo del hiperplano óptimo

Esta sección se toma de (Cortes and Vapnik, 1995). El conjunto de entrenamiento

$$(y_1, X_1), ..., (y_l, X_l), \quad y_i \in \{-1, 1\} \tag{9}$$

se dice que es linealmente separable si existe un vector W y un escalar b tal que las desigualdades

$$\begin{aligned} W \cdot X_i + b &\geq 1 \text{ si } y_i = 1 \\ W \cdot X_i + b &\leq -1 \text{ si } y_i = -1 \end{aligned} \tag{10}$$

son válidas para todos los elementos del conjunto de entrenamiento (9), (Sanchez and Alanis, 2006) y (Haykin, 1999). Las desigualdades (10) se rescriben como

$$y_i(W \cdot X_i + b) \geq 1 \, , i = 1, ..., l. \tag{11}$$

El hiperplano óptimo

$$(W_0 \cdot X + b_0) = 0 \tag{12}$$

es el único que separa los datos de entrenamiento con un margen máximo: determina la dirección $W/|W|$, donde la distancia entre las proyecciones de los vectores de entrenamiento de dos clases diferentes es máxima. Esta distancia $\rho(W, b)$ está dada por

$$\rho(W,b) = \min_{X:y=1} \frac{X \cdot W}{|W|} - \max_{X:y=-1} \frac{X \cdot W}{|W|} \tag{13}$$

El hiperplano óptimo (W_0, b_0) es el argumento que maximiza la distancia (13). De acuerdo con (13) y (11) se tiene que

$$\rho(W_0, b_0) = \frac{2}{|W_0|} = \frac{2}{\sqrt{W_0 \cdot W_0}} \tag{14}$$

Esto significa que el hiperplano óptimo es el único que minimiza $W \cdot W$ bajo las restricciones (11). Los vectores X, para los cuales $y_i(W \cdot X_i + b) = 1$, son los vectores soporte. Para resolver el problema de optimización asociado, es posible construir el Lagrangiano

$$L(W,b,\Lambda) = \frac{1}{2} W \cdot W -$$

$$\sum_{i=1}^{l} \lambda_i [y_i(X_i \cdot W + b) - 1] \tag{15}$$

donde $\Lambda^T = [\lambda_1, \lambda_2, ..., \lambda_l]$ es el vector de multiplicadores de Lagrange no negativos (11). Se sabe que la solución para este problema de optimización se determina por el punto silla del Lagrangiano en el espacio dimensional $2l + 1$ de W, Λ, y b, donde el mínimo se toma con respecto a los parámetros W y b, y el máximo se toma con respecto a los multiplicadores de Lagrange Λ, (Cortes and Vapnik, 1995). En el punto mínimo, (con respecto a W y b), se obtiene

$$\frac{\partial L(W,b,\Lambda)}{\partial W} |_{W=W_0} \equiv (W_0 - \sum_{i=1}^{l} \lambda_i y_i X_i) = 0 \tag{16}$$

$$\frac{\partial L(W,b,\Lambda)}{\partial b} |_{b=b_0} = \sum_{\lambda_i} \lambda_i y_i = 0 \tag{17}$$

De la igualdad (16) se obtiene

$$W_0 = \sum_{i=1}^{l} \lambda_i y_i X_i \tag{18}$$

que expresa que la solución para el hiperplano óptimo puede escribirse como la combinación lineal de vectores de entrenamiento. Nótese que solamente los vectores de entrenamiento X_i con $\lambda_i > 0$ tienen una contribución efectiva a la suma (18).

Sustituyendo (17) y (18) en (15) resulta

$$F(\Lambda) = \sum_{i=1}^{l} \lambda_i - \frac{1}{2} W_0 \cdot W_0 \tag{19}$$

$$F(\Lambda) = \sum_{i=1}^{l} \lambda_i - \frac{1}{2} \sum_{i=1}^{l} \sum_{j=1}^{l} \lambda_i \lambda_j y_i y_j X_i \cdot X_j \tag{20}$$

En notación vectorial, la ecuación anterior puede escribirse como

$$F(\Lambda) = \Lambda^T P - \frac{1}{2} \Lambda^T Q \Lambda \tag{21}$$

donde P es un vector unitario l-dimensional, y Q es una matriz simétrica de $l \times l$ con elementos

$$Q_{ij} = y_i y_j X_i \cdot X_j \tag{22}$$

Para encontrar el punto silla deseado se requiere calcular el máximo de (20) bajo las siguientes restricciones

$$\Lambda^T Y = 0 \tag{23}$$

$$\Lambda \geqslant 0 \tag{24}$$

De acuerdo con el Teorema de Kuhn-Tucker (Kuhn and Tucker, 1961), en el punto silla con W_0, b_0, Λ_0, cualquier multiplicador de Lagrange y sus correspondientes restricciones están relacionadas por la igualdad

$$\lambda_i [y_i(X_i \cdot W_0 + b_0)] - 1 = 0, \;\; i = 1, 2, ..., l. \tag{25}$$

De esta desigualdad se sigue que los valores $\lambda_i \neq 0$ se alcanzan solamente cuando

$$y_i(X_i \cdot W_0 + b_0) - 1 = 0 \tag{26}$$

En otras palabras, $\lambda_i \neq 0$ se presentan para los casos donde la desigualdad (11) satisface la condición de igualdad. Los vectores X_i para los cuales

$$y_i(X_i \cdot W_0 + b_0) = 1 \tag{27}$$

son los vectores soporte. De (18), W_0 puede expandirse en los vectores soporte. Basado en la ecuación (17) y (18) para la solución óptima, la relación entre el valor máximo $F(\Lambda_0)$ y la distancia de separación ρ_0 es como sigue:

$$\begin{aligned} W_0 \cdot W_0 &= \sum_{i=1}^{l} \lambda_i^0 y_i X_i \cdot W_0 \\ &= \sum_{i=1}^{l} \lambda_i^0 (1 - y_i b_0) = \sum_{i=1}^{l} \lambda_i^0 \end{aligned} \tag{28}$$

Sustituyendo esta igualdad en la expresión (19) para $F(\Lambda_0)$ se obtiene

$$F(\Lambda_0) = \sum_{i=1}^{l} \lambda_i^0 - \frac{1}{2} W_0 \cdot W_0 = \frac{1}{2} W_0 \cdot W_0 \tag{29}$$

Tomando en cuenta la expresión (13), resulta

$$F(\Lambda_0) = \frac{2}{\rho_0^2} \tag{30}$$

donde ρ_0 es el margen para el hiperplano óptimo. Si para algún Λ^* y constante grande W^0 la desigualdad

$$F(\Lambda^*) > W^0 \tag{31}$$

es válida, entonces todos los hiperplanos que separan los datos de entrenamiento tienen un margen

$$\rho < \sqrt{\frac{2}{W^0}} \tag{32}$$

Si el conjunto de entrenamiento no puede separarse por un hiperplano, el margen entre los patrones de las dos clases llega a ser arbitrariamente pequeño, resultando en un valor del funcional $F(\Lambda)$ arbitrariamente largo.

3. NUEVO ALGORITMO DE ENTRENAMIENTO ÓPTIMO

Esta sección contiene nuestra contribución principal. Considerando la determinación del hiperplano óptimo explicado anteriormente y el problema de síntesis presentado en los preliminares, proponemos el siguiente algoritmo innovativo para entrenar memorias asociativas implementadas con (1).

Algoritmo de Síntesis 3.1: Dados m patrones de entrenamiento α^k, $k = 1, 2, \ldots, m$ que se sabe pertenecen a la clase C_1 (correspondiente a Z = 1) o C_2 (correspondiente a Z = -1), el vector de pesos W^i puede determinarse por medio del siguiente algoritmo.

1. Inicie por resolver el problema de programación cuadrática dado por

$$F(\Lambda^i) = (\Lambda^i)^T P - \frac{1}{2}(\Lambda^i)^T Q \Lambda^i \qquad (33)$$

bajo las restricciones

$$(\Lambda^i)^T Y^i = 0, Y^i = [\alpha_i^1, \alpha_i^2, ..., \alpha_i^m]$$

$$\Lambda^i \geq 0$$

para obtener $(\Lambda^i)^T$ $=[\lambda_i^1, \lambda_i^2, \ldots, \lambda_i^m]$. 2. Calcule el vector de pesos

$$\begin{aligned} W^i &= \sum_{k=1}^{m} \lambda_i^k \alpha_i^k \alpha^{-k} \\ &= [w_1^i, w_2^i, \ldots, w_n^i, w_{n+1}^i] \end{aligned} \qquad (34)$$

para $i = 1, 2, \ldots, n$, tal que

$$\begin{aligned} W^i \alpha^{-k} + b_i &\geqslant 1, \text{ if } \alpha_i^k = 1 \\ W^i \alpha^{-k} + b_i &\leq -1, \text{ if } \alpha_i^k = -1 \end{aligned} \qquad (35)$$

y para $k = 1, 2, \ldots, m$ donde

$$\alpha^{-k} = \begin{bmatrix} \alpha^k \\ ... \\ 1 \end{bmatrix}$$

3. Seleccione $A = diag[a_1, a_2, \ldots, a_n]$ con $a_i > 0$. Para $i, j = 1, 2, \ldots, n$ seleccione $T_{ij} = w_j^i$ si $i \neq j$, con $\mu_i > 1$, $T_{ij} = w_j^i + a_i\mu_i - 1$ si $i = j$ e $I_i = w_{n+1}^i + b_i$.

Para este algoritmo, se requiere seleccionar $\mu_i > 1$ y A puede escogerse como la matriz identidad. El siguiente resultado establece la validez del algoritmo de síntesis propuesto.

Teorema 3.1:

1. Las matrices A, T e I obtenidas en el Algoritmo de síntesis 3.1 satisface las condiciones (3),(4), y la ecuación (1); por lo tanto, A, T e I satisfacen el punto 1 del problema de síntesis.

2. El Algoritmo de Síntesis 3.1 siempre converge.

Prueba:

1. Considerando la definición de α^{-k} y la selección para T_{ij} e I_i en el Algoritmo de Síntesis 3.1 con $a_i > 0$ y $\mu_i > 1$, puede verse que para i=1,2,...,n y k=1,2,...,m, $W^i\alpha^{-k}+b_i \geq 1$, (cuando α_i^k=1) implica

$$\begin{aligned} &T_{i1}\alpha_1^k + T_{i2}\alpha_2^k + \ldots + T_{ii}\alpha_i^k + \ldots \\ &+T_{in}\alpha_n^k + I_i \geq a_i\mu_i\alpha_i^k > a_i \end{aligned} \qquad (36)$$

y $W^i\alpha^{-k} + b_i \leq -1$, (cuando $\alpha_i^k = -1$) implica

$$\begin{aligned} &T_{i1}\alpha_1^k + T_{i2}\alpha_2^k + \ldots + T_{ii}\alpha_i^k + \ldots \\ &+T_{in}\alpha_n^k + I_i \leq a_i\mu_i\alpha_i^k < -a_i \end{aligned} \qquad (37)$$

2. Considerando el i-ésimo entrenamiento óptimo en el Algoritmo de Síntesis 3.1. Los patrones de entrenamiento α^{-k} en el algoritmo se etiquetan de acuerdo con el i-ésimo elemento α_i^k de α^{-k} para $k = 1, 2, \ldots, m$. Puesto que cada $\alpha^k \in B^n$, está claro que el presente caso es linealmente separable y que ésto es cierto para cada $i = 1, 2, \ldots, n$.

Observación 3.1: Si se desea que el Algoritmo de Síntesis 3.1 resulte en un sistema de la forma (1) con I=b_i, la ecuación (35) debe modificarse en el Algoritmo de Síntesis 3.1 como sigue

$$\begin{aligned} W^i\alpha^k + b_i &\geq 1, \text{ si } \alpha_i^k = 1 \\ W^i\alpha^k + b_i &\leq -1, \text{ si } \alpha_i^k = -1 \end{aligned} \qquad (38)$$

donde

$$W^i = \sum_{k=1}^{m} \lambda_i^k \alpha_i^k \alpha^k = [w_1^i, w_2^i, \ldots, w_n^i] \qquad (39)$$

Seleccione A y T como en el Algoritmo de Síntesis 3.1, y seleccione I=b_i. El Teorema 3.1 es aún válido para este caso. Es justo mencionar que el algoritmo de síntesis propuesto es válido solamente para entradas bipolares a la memoria asociativa.

4. DISEÑO CON RESTRICCIONES ÓPTIMAS

Del Algoritmo de Síntesis 3.1 y del Teorema 3.1, puede verse que cuando se utiliza este algoritmo no hay restricciones en cuanto al número de vectores bipolares que pueden almacenarse como vectores de memoria estable en el sistema (1). Esto implica que la capacidad de almacenamiento (máximo número de patrones deseados permitidos) puede ser muy grande. Sin embargo, cuando el número de patrones a almacenar es muy grande, la matriz de conexión T debe cumplir con ciertas restricciones para reducir el número total de memorias espurias. En general, elementos positivos grandes en la diagonal de la matriz de conexión T resultan en un número grande de memorias espurias. Las regiones de atracción de las memorias deseadas se reducen conforme el número de memorias espurias se incrementa. Uno de los objetivos del enfoque de síntesis propuesto consiste en diseñar redes neuronales con restricciones en los elementos de la diagonal de la matriz de conexión T de tal modo que la cantidad de memorias espurias se reduzca y que las regiones de atracción de las memorias deseadas incrementen su tamaño.

Al igual que en (Liu and Lu, 1997), las restricciones en los elementos de la diagonal de la matriz de conexión T dadas por $T_{ii} = a_i$ para $i = 1, 2, \ldots, n$ serán denominadas aquí como *restricciones óptimas*. Las redes neuronales que satisfacen dichas restricciones, usualmente tienen menos memorias espurias que aquellas redes que no las satisfacen (Liu and Lu, 1997). Con estas restricciones:

1. La red neuronal recurrente tendrá solamente vectores de memoria estable.

2. Cada esquina del hipercubo que está en la vecindad inmediata de una memoria estable no puede llegar a ser una memoria estable y es muy probable que esté en el dominio de atracción de este vector de memoria estable.

La siguiente observación y el Teorema consecuente constituyen también aportaciones en el presente artículo.

Observación 4.1: Del Algoritmo de Síntesis 3.1, se sabe que $T_{ii} = w_i^i + a_i\mu_i - 1$ donde μ_i se selecciona de tal modo que $\mu_i > 1$. Cuando el peso w_i^i obtenido del Algoritmo de Síntesis 3.1 satisface que $w_i^i < 1$, se puede seleccionar $a_i > 0$, $T_{ii} = a_i$, y $\mu_i = (1 - w_i^i)/a_i + 1 > 1$, o bien se puede seleccionar $a_i > 0$ y $T_{ii} < a_i$ tal que $\mu_i = (T_{ii} - w_i^i)/a_i + 1 > 1$. Es fácil probar que el diseño de una red neuronal con el i-ésimo elemento diagonal de la matriz de conexión T satisfaciendo $T_{ii} \leq a_i$ donde $a_i > 0$ es el i-ésimo elemento de la diagonal de la matriz A, se alcanza si $w_i^i < 1$ se obtiene del Algoritmo de Síntesis 3.1.

Teorema 4.1: El diseño de una red neuronal con el i-ésimo elemento de la diagonal de T satisfaciendo $T_{ii} \leq a_i$,donde $a_i > 0$ es el i-ésimo elemento de la diagonal de la matriz A, se alcanza si los patrones deseados α^1, α^2,..., α^m con la i-ésima entrada eliminada son linealmente separables, donde las dos clases se determinan de acuerdo con el i-ésimo elemento de los patrones deseados.

Prueba: Con base en el algoritmo de hiperplano óptimo, es posible obtener los pesos W^i utilizando el Algoritmo de Síntesis 3.1, tal que (38) se satisface para $k = 1, 2, \ldots, m$ y para $i = 1, 2, \ldots, n$. Suponiendo que los patrones deseados α^1,α^2,...,α^m con la i-ésima entrada eliminada son linealmente separables para algún i, $1 \leq i \leq n$, donde las dos clases se determinan de acuerdo con el i-ésimo elemento. Con este fin, el Algoritmo de Síntesis 3.1 se aplica para obtener W^i sin utilizar el i-ésimo elemento de los patrones deseados (patrones de entrenamiento). La desigualdad (38) implica en este caso que si $\alpha_i^k = 1$, entonces

$$w_1^i\alpha_1^k + \ldots + w_{i-1}^i\alpha_{i-1}^k + w_{i+1}^i\alpha_{i+1}^k + \ldots + w_{n+1}^i + b_i \geq 1 \qquad (40)$$

y que si $\alpha_i^k = -1$, entonces

$$w_1^i\alpha_1^k + \ldots + w_{i-1}^i\alpha_{i-1}^k + w_{i+1}^i\alpha_{i+1}^k + \ldots + w_{n+1}^i + b_i \leq -1 \qquad (41)$$

para $k = 1, 2, \ldots, m$. Las ecuaciones (40) y (41) pueden rescribirse como

$$w_1^i\alpha_1^k + \ldots + w_{i-1}^i\alpha_{i-1}^k + w_{i+1}^i\alpha_{i+1}^k + \ldots + w_{n+1}^i + b_i - 1 \geq 0 \qquad (42)$$

$$w_1^i\alpha_1^k + \ldots + w_{i-1}^i\alpha_{i-1}^k + w_{i+1}^i\alpha_{i+1}^k + \ldots + w_{n+1}^i + b_i + 1 \leq 0 \qquad (43)$$

para $k = 1, 2, \ldots, m$. Seleccione ϵ^i tal que

$$0 < \epsilon^i < \min_{1 \leq k \leq m} \{| \sum_{j=1, j\neq i}^{n} w_j^i\alpha_j^k + w_{n+1}^i + b_i|\} \qquad (44)$$

Sea $w_i^i - 1 = -\epsilon^i$. Entonces de (42) a (44), es evidente que $\alpha_i^k = 1$ implica

$$\sum_{j=1}^{n} w_j^i\alpha_j^k + w_{n+1}^i + b_i - 1$$

$$= \sum_{j=1, j\neq i}^{n} w_j^i\alpha_j^k + w_{n+1}^i + b_i + w_i^i\alpha_i^k - 1$$

$$= | \sum_{j=1, j\neq i}^{n} w_j^i\alpha_j^k + w_{n+1}^i + b_i| - \epsilon^i \geq 0 \qquad (45)$$

y que $\alpha_i^k = -1$ implica

$$\sum_{j=1}^{n} w_j^i\alpha_j^k + w_{n+1}^i + b_i + 1$$

$$= \sum_{j=1, j\neq i}^{n} w_j^i\alpha_j^k + w_{n+1}^i + b_i + w_i^i\alpha_i^k + 1$$

$$= -| \sum_{j=1, j\neq i}^{n} w_j^i\alpha_j^k + w_{n+1}^i + b_i| + \epsilon^i \leq 0 \qquad (46)$$

Por lo tanto, siempre se obtiene W^i con $w_i^i < 1$. Esto implica que el i-ésimo elemento de la diagonal de la matriz de conexión T satisface $T_{ii} \leq a_i$, donde $a_i > 0$.

5. APLICACIÓN AL DIAGNÓSTICO DE FALLAS EN CENTRALES TERMOELÉCTRICAS

En esta sección solamente se describen los resultados de la clasificación de dos de las fallas que comúnmente ocurren en centrales termoeléctricas. Los resultados para más fallas y bajo diferentes escenarios pueden consultarse en (Ruz-Hernandez *et al.*, 2007) y (Ruz-Hernandez, 2006), donde se describe la implementación de un esquema neuronal para detección y diagnóstico de fallas (ver Figura 2).

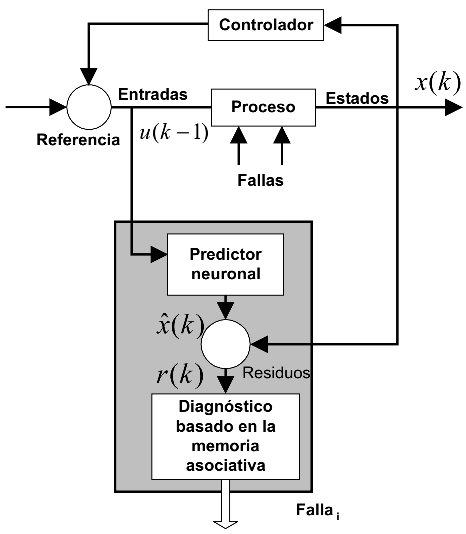

Figura 2. Esquema neuronal para detección y diagnóstico de fallas.

El esquema consta de dos componentes. La primera componente se basa en la comparación entre las mediciones provenientes de la planta y los valores estimados por el predictor neuronal. El predictor se basa en modelos neuronales que se entrenan con datos saludables (sin presencia de fallas) del proceso. Las diferencias entre estos dos valores se conocen como residuos y constituyen un buen indicador para detectar fallas. Los residuos se calculan como

$$r_j(k) = x_j(k) - \hat{x}_j(k), j = 1, \ldots, 5 \quad (47)$$

donde $x_i(k)$ denota las mediciones de la planta y $\hat{x}_i(k)$ corresponde a las predicciones. En ausencia de fallas, los residuos son solamente ruido y perturbaciones. Cuando una falla ocurre, los residuos se desvían de cero en trayectorias diferentes.

El predictor se diseña utilizando cinco redes neuronales previamente entrenadas con el algoritmo de aprendizaje de Levenberg-Marquardt. Cada red neuronal es del tipo perceptrón multicapa recurrente con funciones de activación de tangente hiperbólica en las neuronas de la capa oculta y una sola neurona de salida con función de activación lineal. Los modelos neuronales que corresponden a cada red se obtienen utilizando la herramienta NNSYSID (por su nombre en inglés "Neural Networks Based System Identification Toolbox"), que trabaja en plataforma de MATLAB (Norgaard, 2000). Los modelos tienen ocho variables de entrada y una variable de salida con estructura NNARX (por su nombre en inglés "Neural Network Autoregressive with External Input"):

$$\hat{x}_j(k) = F_j[W_j, x_j(k-1), ..., x_j(k-n_a), u_1(k-1), ..., u_1(k-n_b), ..., u_8(k-1), ..., u_8(k-n_b)] \quad (48)$$

donde $j = 1, \cdots, 5$ y las variables de entrada son:

$u_1(.)$ = Flujo de combustible (%).

$u_2(.)$ = Flujo de aire(%).

$u_3(.)$ = Flujo de agua de condensado (L/min).

$u_4(.)$ = Flujo de agua de atemperación (Kg/s).

$u_5(.)$ = Flujo de agua de alimentación (T/H).

$u_6(.)$ = Flujo de agua de repuesto al condensador (L/s).

$u_7(.)$ = Flujo de vapor (L/min).

$u_8(.)$ = Ángulo de inclinación de quemadores (Grados).

y las variables de salida son

$x_1(.)$ = Carga de la unidad (MW).

$x_2(.)$ = Presión del hogar (Pa).

$x_3(.)$ = Nivel del domo (m).

$x_4(.)$ = Temperatura de vapor recalentado (^{o}K).

$x_5(.)$ = Temperatura de vapor sobrecalentado(^{o}K).

donde W_j representa los pesos de cada modelo neuronal.

Los retardos (n_a y n_b)de cada modelo neuronal se determinan utilizando el criterio basado en los coeficientes de Lipschitz (Norgaard, 2000). Los modelos neuronales se entrenaron con 3000, 3000, 2000, 4000 and 4000 muestras experimentales, respectivamente y se validaron con datos frescos. Los errores de predicción son cercanos al 1 %.

Para generar los residuos, el predictor basado en redes neuronales trabaja en paralelo con la planta, de manera que la ecuación (48) se modifica como sigue:

$$\hat{x}_j(k) = F_j[W_j, \hat{x}_(k-1), ..., \hat{x}_j(k-n_a), u_1(k-1), ..., u_1(k-n_b), ..., u_8(k-1), ..., u_8(k-n_b)] \quad (49)$$

Los modelos neuronales operando en paralelo con la planta se probaron en simulación para efectuar hasta 6700 predicciones con datos que incluyen cambios de carga de la unidad y variaciones en el nivel del domo. Las pruebas de validación para todos los modelos pueden consultarse en (Ruz-Hernandez, 2006).

El esquema desarrollado, se aplica fuera de línea a dos fallas conocidas como *rotura de tubos de pared de agua* y *rotura de tubos en el sobrecalentador*, a las que denominaremos *falla* 1 y *falla* 2, respectivamente. Las bases de datos se adquieren con un simulador de plena escala para un 75 de carga inicial (225 MW), 15 % de severidad de la falla, 2 minutos de retardo y 8 minutos de tiempo total de simulación. En ambos casos, los residuos son cercanos a cero durante 120 segundos (2 minutos). Después de este intervalo, los residuos se desvían de cero en trayectorias diferentes. Los residuos obtenidos para la *falla* 1 se muestran en la Figura 3. En la segunda componente, los residuos

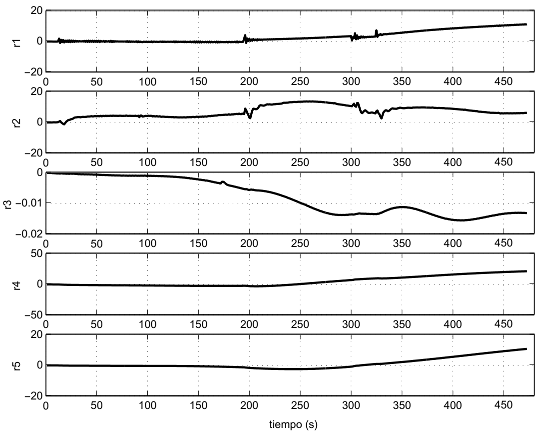

Figura 3. Residuos de la *falla* 1.

de cada falla se codifican como vectores bipolares empleando umbrales de detección (τ_i) con la finalidad de analizarlos y obtener los patrones de falla a almacenar en la memoria asociativa. Los umbrales de detección se muestran en la Tabla 1.

Tabla 1. Umbrales de detección.

i	1	2	3	4	5
τ_i	20 MW	50 Pa	0.01 m	30 K	30 K

Este proceso proporciona un conjunto de patrones de falla $[s_1, s_2, s_3, s_4, s_5]^T$, donde:

$$s_i = \begin{cases} -1 \text{ si } |r_i| & < \tau_i \\ 1 \text{ si } |r_i| & \geq \tau_i \end{cases} \quad i = 1, 2, \ldots, 5 \qquad (50)$$

Los patrones de falla se utilizan para entrenar por medio de nuestro algoritmo óptimo propuesto a la memoria asociativa, la cual se usa para diagnosticar fallas (ver Figura 4). El nuevo algoritmo de síntesis se implementa utilizando MATLAB para entrenar la red neuronal recurrente de tal forma que los patrones de falla $\alpha^1 = [-1, -1, 1, -1, -1]^T$ y $\alpha^2 = [1, 1, 1, 1, 1]^T$ se almacenen como vectores de memoria estable. El número de neuronas es $n = 5$ y el de patrones de falla es $m = 2$. La matriz $LM = [\Lambda^1, \Lambda^2, \ldots, \Lambda^n]$ que contiene los vectores con los multiplicadores de Lagrange, la matriz de pesos $WM = [W^1, W^2, \ldots, W^{n+1}]$ y la matriz de vectores de polarización $BV = [b_1, b_2, \ldots, b_n]$ se obtienen como en (51), (52) y (53). A se escoge como la matriz identidad de 5 x 5, en tanto que T e I se obtienen como en (54) y (55).

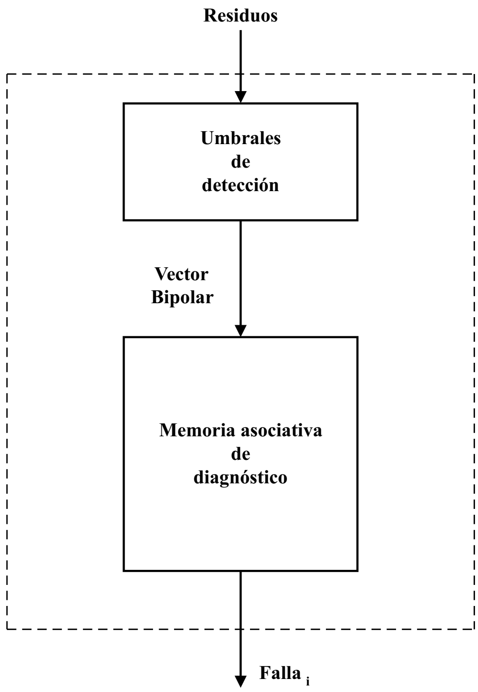

Figura 4. Esquema neuronal de clasificación de fallas.

$$LM = \begin{bmatrix} 0.125 & 0.125 & 0.00 & 0.125 & 0.125 \\ 0.125 & 0.125 & 0.00 & 0.125 & 0.125 \end{bmatrix} \qquad (51)$$

$$WM = \begin{bmatrix} 0.25 & 0.25 & 0.0 & 0.25 & 0.25 & 0.0 \\ 0.25 & 0.25 & 0.0 & 0.25 & 0.25 & 0.0 \\ 0.00 & 0.00 & 0.0 & 0.00 & 0.00 & 0.0 \\ 0.25 & 0.25 & 0.0 & 0.25 & 0.25 & 0.0 \\ 0.25 & 0.25 & 0.0 & 0.25 & 0.25 & 0.0 \end{bmatrix} \qquad (52)$$

$$BV = \begin{bmatrix} 0.00 & 0.00 & -0.19 & 0.00 & 0.00 \end{bmatrix}^T \qquad (53)$$

$$T = \begin{bmatrix} 1.00 & 0.25 & 0.00 & 0.25 & 0.25 \\ 0.25 & 1.00 & 0.00 & 0.25 & 0.25 \\ 0.00 & 0.00 & 1.00 & 0.00 & 0.00 \\ 0.25 & 0.25 & 0.00 & 1.00 & 0.25 \\ 0.25 & 0.25 & 0.00 & 0.25 & 1.00 \end{bmatrix} \qquad (54)$$

$$I = \begin{bmatrix} 0.00 & 0.00 & -0.19 & 0.00 & 0.00 \end{bmatrix}^T \qquad (55)$$

La memoria asociativa se evalúa con las matrices A, T e I obtenidas, para diferentes patrones iniciales. La Figura 5 ilustra la evolución de los estados de la red neuronal recurrente cuando el patrón de falla α^1 se utiliza como vector de entrada a la memoria. La memoria asociativa converge y el vector de salida $y = sat(x)$ es $y = [-1, -1, 1, -1, -1]^T$, el cual corresponde al patrón de la *falla* 1.

6. CONCLUSIONES

En este artículo, se consideró el diseño de memorias asociativas implementadas con redes neuronales recurrentes para diagnosticar fallas en centrales termoeléctricas. Se propuso un nuevo enfoque de síntesis basado en el algorimo de hiperplano óptimo para MVS. El algoritmo de síntesis propuesto siempre converge cuando no hay restricciones en la matriz de conexión de la red recurrente (Teorema 3.1). Se establecieron los resultados que conciernen al diseño

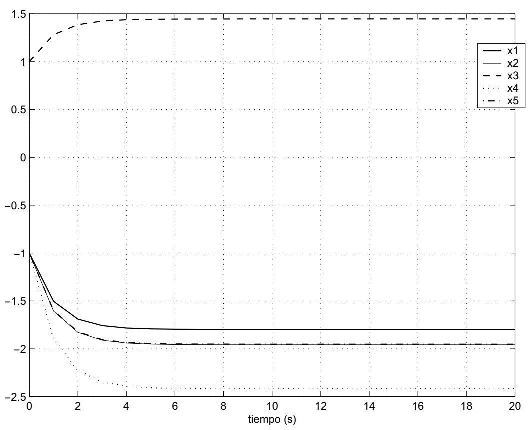

Figura 5. Evolución de los estados, *falla* 1.

con restricciones en los elementos de la diagonal de la matriz de conexión (Teorema 4.1) para reducir el número total de memorias espurias. Los resultados obtenidos nos motivan a continuar investigando para proponer nuevos enfoques de síntesis basados en algoritmos utilizados para MVS como el entrenamiento con margen suave, que también puede emplearse en el diseño de esta clase de memorias asociativas cuando los patrones de entrenamiento no son todos linealmente separables. Así mismo, se requiere considerar otros ejemplos de aplicación para efectuar un análisis comparativo con otros algoritmos en cuanto a su capacidad de almacenamiento en relación con la determinación de memorias espurias.

AGRADECIMIENTOS

Los autores agradecen al CONACYT, México su apoyo para el proyecto 39866Y y al IIE, México por permitirnos usar su simulador de centrales termoeléctricas. El primer autor agradece a la UNACAR, México y al Promep, México su apoyo a través de los proyectos PR/62/2006 y P/-CA-7 2006 05, respectivamente .

REFERENCIAS

Boser, B. E., E. M. Gullon and V. N. Vapnik (1992). A training algorithm for optimal margin classifiers. *Proceedings of the Fifth Annual Workshop of Computational Learning Theory* **5**, 144–152.

Chartier, S. and R. Proulx (2006). Ndram: Nonlinear dynamic recurrent associative memory for learning bipolar and nonbipolar correlated patterns. *IEEE Transactions on Neural Networks* **16**, 1393–1400.

Cortes, C. and V. N. Vapnik (1995). Support vector networks. *Machine Learning* **20**, 273–297.

Hassoun, M. H. (1993). *Associative Neural Memories: Theory and Implementation*. Oxford Univ. Press. Oxford, U. K.

Hassoun, M. H. and A. M. Youssef (1990). Asociative neural memory capacity and dynamics. *Proceedings of International Joint Conference on Neural Networks* **1**, 763–769.

Haykin, S. (1999). *Neural Networks, A comprehensive foundation*. Prentice Hall. New Yersey,USA.

Kuhn, H. W. and A. W. Tucker (1961). Nonlinear programming. *Proceedings of the Second Berkeley Symposium on Mathematical, Statistics and Probability* pp. 481–492.

Lee, D. L. and T. C. Chuang (2006). Designing asymetric hopfield-type associative memory with higher order hamming stability. *IEEE Transactions. on Neural Networks* **16**, 1464–1476.

Li, J. H., A. N. Michel and W. Porod (1988). Qualitative analysis and synthesis of a class of neural networks. *IEEE Transactions on Circuits and Systems* **CAS-35**, 976–986.

Liu, D. and A. N. Michel (1994). *Dynamical Systems with Saturations Nonlinearities: Analysis and Design*. Springer Verlag. New York, USA.

Liu, D. and A. N. Michel (1996). Robustness analysis and design of a class of neural networks with sparse interconnecting structure. *Neurocomputing* **12**, 59–76.

Liu, D. and Z. Lu (1997). A new synthesis spproach for feedback neural netwoks based on the perceptron training algorithm. *IEEE Transactions on Neural Networks* **8**, 1468–1482.

Norgaard, M. (2000). *Neural Networks Based System Identification Toolbox for Use with MATLAB, Technical Report 00-E-891*. Deparment of Automation, Technical University of Denmark. New York, USA.

Ruz-Hernandez, J. A. (2006). *Development and Aplication of a Neural Network-based Scheme for Fault Diagnosis in Fossil Electric Power Plants, Ph. D. Thesis*. Centro de Investigacion y Estudios Avanzados del Instituto Politecnico Nacional (CINVESTAV-IPN). Guadalajara Campus (in Spanish).

Ruz-Hernandez, J. A., E. N. Sanchez and D. A. Suarez (2007). *Optimal Training for Associative Memories: Application to Fault Diagnosis in Fossil Electric Power Plants*. Book Chapter of Hybrid Intelligent Systems, Analysis and Design, Edited by O. Castillo et al., International Series in Fuzzyness Studies and Soft Computing, Vol. 208, pp. 326-359, Springer-Verlag-Heidelberg. Germany.

Sanchez, E. N. and A. Y. Alanis (2006). *Neural Networks, Foundations and Applications to Automatic Control*. Pearson Education. Madrid,España (in Spanish).

Vapnik, V. N. (1982). *Estimation of Dependences Based in Empirical Data, Addendum I*. Springer Verlag. New York, USA.