PARADIGMAS 2 (1) 141-164 (2010)
Revista del Departamento Académico de Ingeniería.
Universidad Ricardo Palma. ISSN: 2220-2056

Entrenamiento de Redes Neuronales para el Reconocimiento de Frutas

Hugo Froilán Vega Huerta,[1] Augusto Cortez Vásquez

Resumen

En la revista RISI - Volumen 6 - número 2 - 2009 II (Revista de Investigación de la facultad de Ingeniería de Sistemas e Informática de la UNMSM) presentamos el artículo Reconocimiento de patrones Mediante Redes Neuronales Artificiales, que resolvía el reconocimiento de los caracteres numéricos 3, 4, 5 con una red neuronal pequeña de 42 neuronas de entrada, 3 de salida, y 100 neuronas en la capa oculta.
El presente es artículo, basado en el artículo mencionado, presenta en entrenamiento de redes neuronales para el reconocimiento de frutas. Con el propósito hacer un trabajo sencillo e ilustrativo solamente trabajaremos con dos tipos de frutas (plátanos y naranjas) y dos modelos diferentes por cada tipo de fruta.

Palabras Claves

Redes Neuronales Artificiales, Reconocimiento de Frutas, Reconocimiento de Patrones

Abstract

In the magazine, RISI - Volume 6 - Number 2 - 2009 II (Journal of Research of the Faculty of Engineering and Computer Systems of San Marcos University) we presented the article Pattern Recognition Using Artificial Neural Networks, which solved the recognition of numeric characters 3, 4, 5 with a small neuronal network of 42 input neurons, 3 output neurons, and 100 neurons in the hidden layer.
This article is based on the aforementioned article and presented in training neural networks for the recognition of fruit. In order to make a simple and illustrative work, we only work with two kinds of fruit (bananas and oranges) and two different models for each type of fruit.
The consideration of "factors" - "goal factors" as influential to the Academic Performance, are some of the "variables" to consider in order to explain the level of learning acquired by a student, learning is expressed as a product rating of the evaluation process.

Key words

Artificial Neural Networks, Fruit Recognition, Pattern Recognition

1 Licenciado en Computación, Magister, Docente Universitario, U.R.P.

Introducción.

Existe en el hombre un deseo profundo de poder reproducir la habilidad cognoscitiva por medios artificiales. La fascinación que la inteligencia como materia de estudio ha suscitado al género humano, puede verse reflejada en la aparición de una rama íntegra del estudio científico llamada "Inteligencia Artificial" de la cual una de las múltiples ramas es "redes neuronales".

Una de las aplicaciones de las redes neuronales es el mecanismo de aprendizaje y reconociendo de patrones.

Una persona puede apreciar la foto de un familiar y reconocer inmediatamente de quien se trata, pero para ello es obvio que en algún lugar de su cerebro, previamente, debe tener almacenado dicha imagen vinculada con la identificación de la persona.

De modo similar, mediante las redes neuronales artificiales podemos realizar reconocimientos de imágenes, pero para ello previamente debemos entrenarla para que pueda almacenar dentro de sus estructuras a la información previa para poder realizar dichos reconocimientos. Justamente éste será el trabajo que demostraremos en el presente artículo.

Fundamentación Teórica

Los conceptos teóricos que presentaremos corresponden en su totalidad a los que enunciamos en el artículo Reconocimiento de patrones Mediante Redes Neuronales Artificiales de la revista RISI - Volumen 6 - Número 2 - 2009 (Revista de Investigación de la facultad de Ingeniería de Sistemas e Informática de la UNMSM). Lo presentamos por que es necesario que los lectores tengan un marco teórico básico para entender nuestro planteamiento.

Definición de Redes Neuronales Artificiales (RNA) [1]

Son modelos Matemáticos construidos inspirados en el funcionamiento de las Redes Neuronales biológicas (Sistema Nervioso), por consiguiente, las unidades de procesamiento fundamental de una RNA, serán las Neuronas Artificiales.

Una Red Neuronal Artificial la definiremos como un conjunto de unidades de procesamiento llamados Neuronas, células o nodos, interconectados entre sí por varias ligaduras de comunicación directa llamadas conexiones, con la finalidad de recibir señales de entrada, procesarlas y emitir señales de salida. Cada conexión está asociada a un peso, que representan la información utilizada por las neuronas para resolver un problema.

A continuación presentamos la representación gráfica de una RNA.

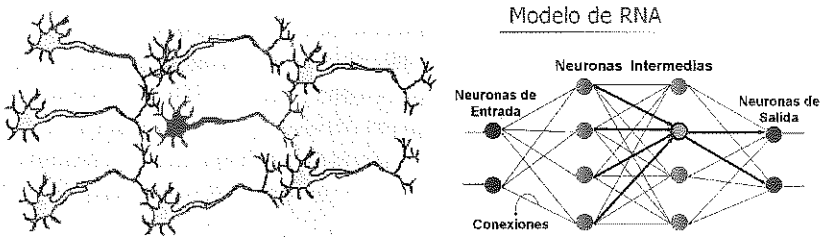


Figura 1: Representación Comparativa de una Red Neuronal Natural y de una RNA

Cada uno de estos procesadores o neuronas suman las entradas ponderadas y su resultado pasa a través de una función no lineal conocida como función de activación. Las entradas a una neurona pueden provenir de fuentes externas o de otras neuronas en la red. Así mismo la salida de una neurona es enviada a otras neuronas o al entorno. El conocimiento en una red neuronal está distribuido a lo largo de todo el sistema, debido a esto, se utilizan muchas interconexiones para obtener la solución de un problema en particular

Características de las Redes Neuronales Artificiales [2]

Las características de las RNA son, como apreciaremos muy semejantes a las de las Redes Neuronales Biológicas, entre las principales podemos mencionar las siguientes:

- Aprenden a través de ejemplos
- Inferencia estadística
- Adaptabilidad
- Dilema plasticidades y estabilidad
- Capacidades de generalización
- Tolerante a fallas.
- Rápida implantación

Principales Aplicaciones de las Redes Neuronales Artificiales [3]

Las principales aplicaciones de las RNA son:

- Clasificación de Patrones
- Agrupamiento Categorización
- Aproximación de funciones
- Memoria direccionable por contenido

Funcionamiento de una Neurona Artificial [2]

De igual modo que las neuronas biológicas, una neurona artificial, tiene varias entradas y una sola salida, la misma que a su vez se puede aplicar a muchos otras neuronas. En la siguiente figura presentamos el esquema de una Neurona Artificial.

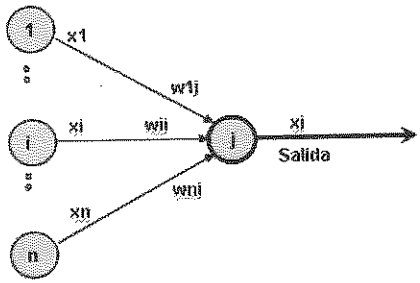


Figura 2: Esquema de una Neurona Artificial

Si comparamos las neuronas artificiales del presente gráfico con las neuronas biológicas, entonces (x1, x2, ...xi,...xn) vendrían a ser los impulsos de entrada para la neurona j que provienen de las neuronas (1,..i,...n). Los pesos (wx1j, ...,wij,....wnj) representarían el potencial excitador o inhibidor de las conexiones sinápticas respectivas.

Cada elemento de procesamiento obtiene un valor de entrada neto basándose en todas las conexiones de entrada. La forma típica es calcular el valor de entrada neto sumando los valores de entrada, ponderados (multiplicados) con sus pesos correspondientes. La entrada neta del j-ésimo nodo se escribe

$$\text{Neto j} = \sum \text{S xij} * \text{wij}$$

Clasificación de las Redes Neuronales Artificiales según sus Conexiones [2]

Las neuronas de una RNA se organizan en niveles (conocidos también como camada, capas o estratos). Dependiendo de las conexiones que unen las capas, las RNA se pueden clasificar en recurrentes (Feedback) y no recurrentes (Feedforward)

- No Recurrentes (Feedforward)
- Recurrentes (Feedback)

Principales Modelos de Redes Neuronales Artificiales [1] [2]

Veremos solamente el modelo Perceptrón y el modelo Retropropagación

Modelo Perceptrón

El primer modelo de red neuronal artificial fue desarrollado por Rosenblatt en 1958.
Despertó un enorme interés en los años 60, debido a su capacidad para aprender a reconocer patrones sencillos: un Perceptrón, formado por varias neuronas lineales para recibir las entradas a la red y una neurona de salida, es capaz de decidir cuándo una entrada presentada a la red pertenece a una de las dos clases que es capaz de reconocer. A continuación podemos apreciar la arquitectura del Perceptrón.

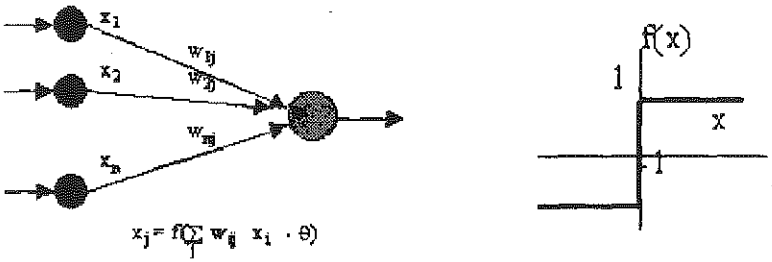


Figura 3: Arquitectura del Perceptrón

La única neurona de salida del Perceptrón realiza la suma ponderada de las entradas, resta el umbral y pasa el resultado a una función de trasferencia de tipo escalón. La regla de decisión es responder +1 si el patrón presentado pertenece a la clase A y con –1 si el patrón pertenece a la clase B. La salida dependerá de la entrada neta (suma de las entradas Xi ponderadas) y del valor umbral

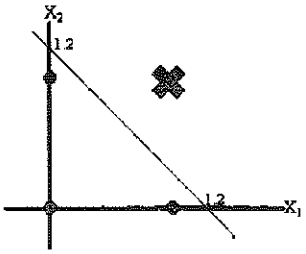


Figura 4: Un problema linealmente separable: la función lógica OR

Este modelo sólo es capaz de discriminar patrones muy sencillos, linealmente separables. La separabilidad lineal limita a las redes con sólo dos capas a la resolución de problemas en los cuáles el conjunto de puntos (correspondientes a los valores de entrada) sean separables geométricamente. En el caso de dos entradas, la separación se lleva a cabo mediante una línea recta. Para tres entradas, la separación se realiza mediante un plano en el espacio tridimensional, y así sucesivamente hasta el caso de N entradas, en el cuál el espacio N-dimensional es dividido en un hiperplano.

El caso de la función XOR es un problema que no puede resolver el perceptrón ya que no es linealmente separable. Sin embargo es posible resolver correctamente este problema usando una red de perceptrones (multiperceptrón).

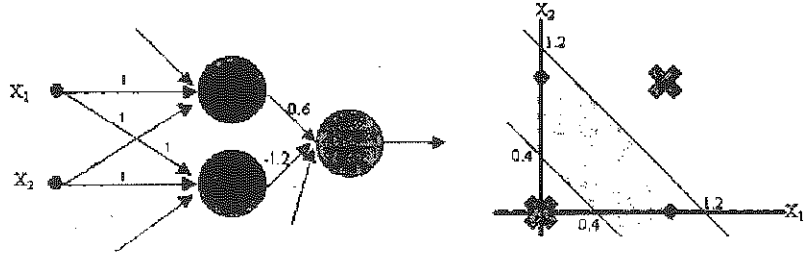


Figura 5: Solución del XOR con una red Multiperceptrón

Modelo Retropropagación [2]

Rumelhart (1986), formalizó un método para que una red neuronal! aprendiera una asociación que existe entre sus patrones de entrada y las clases correspondientes. Este método es conocido como backpropagation, propagación del error hacia atrás o retropropagación, y está basado en la regla de aprendizaje que es posible aplicar solo a modelos de redes multicapa. Una característica importante de este algoritmo es la representación interna del conocimiento que es capaz de organizar en la capa o capas intermedias, para conseguir cualquier correspondencia entre la entrada y la salida de la red.

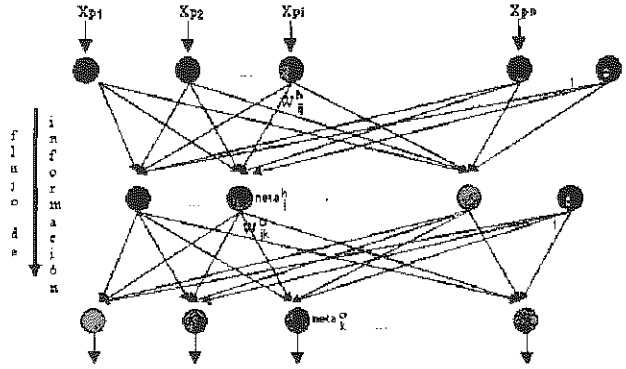


Figura 6: RNA Modelo Retropropagación

Un punto importante en la red de retropropagación es su capacidad de auto adaptar los pesos de las neuronas de las capas intermedias para aprender la relación que existe entre un conjunto de patrones dados como ejemplo y sus salidas correspondientes. Y después utilizar esa misma relación a nuevos vectores de entrada con ruido o incompletos, dando una salida activa si la nueva entrada es parecida a las presentadas durante el aprendizaje.

El Aprendizaje en las Redes Neuronales Artificiales [1] [2]

El problema principal al trabajar con RNA es programar el aprendizaje: ¿cómo escoger los pesos en las conexiones para que la red realice una tarea específica?

En algunos casos, es posible encontrar algunos problemas cuyos pesos se tienen de manera a priori y esta información es considerada para el diseño de la red, pero estos problemas son más bien raros ya que en la mayoría de los casos se debe enseñar a la red a ejecutar los cálculos por ajustes iterativos de los pesos wij. Esto puede tomar dos caminos.

Aprendizaje No Supervisado (Sin Maestro)

Cuando el fin del aprendizaje no es definir en términos específicos de ejemplos correctos, debido a que la información disponible solo está en correlación de datos o señales de entrada y solo se requiere, que en base a esta entrada, la red forme categorías de estas correlaciones, y produzca una señal de salida correspondiente a cada categoría de entrada. Se caracteriza por que la salida no requiere ser contrastada con algo específico ya conocido (maestro).

Hay dos formas de llevar a cabo el Aprendizaje No Supervisado:

- Aprendizaje Hebbiano.
- Aprendizaje Competitivo y Cooperativo.

Aprendizaje Supervisado (con Maestro)

En este modo, el aprendizaje se logra en base a la comparación directa de la salida de la red con la respuesta correcta (maestro) ya conocida. Se basa en el reforzamiento del aprendizaje, donde la retroalimentación se realiza en base a la diferencia de salida real con la salida esperada.

- A continuación presentamos algunos tipos de aprendizajes supervisados
- Aprendizaje Supervisado por Corrección de Error.
- Aprendizaje Supervisado por Refuerzo.
- Aprendizaje Supervisado Estocástico.

En el presente artículo solamente analizaremos en detalle el caso del **Aprendizaje Supervisado por Corrección de Error.**

Aprendizaje por Corrección de Error

El entrenamiento consiste en presentar al sistema un conjunto de pares de datos, representando la entrada y la salida deseada para dicha entrada. Este conjunto recibe el nombre de conjunto de entrenamiento. El objetivo es tratar de minimizar el error entre la Salida Deseada y la actual.

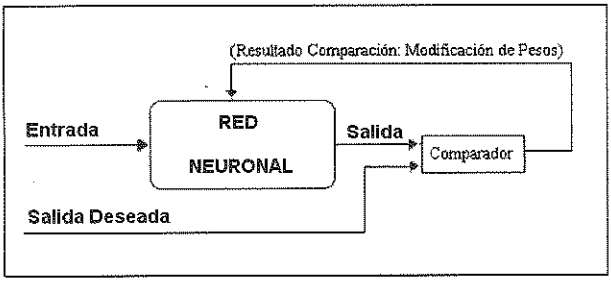


Figura 7: Aprendizaje por Corrección de Error

Algoritmos de Aprendizaje por Corrección del error

Los pesos se ajustan en función de la diferencia entre los valores deseados y los obtenidos en la salida.

$$\Delta w_{ij} = \alpha\, x_i\, (d_j - x_j)$$

donde: Δw_{ij} Variación en el peso de la conexión entre el i-ésimo nodo y el j-ésimo.

α Umbral en el aprendizaje que regula velocidad y precisión ($0<\alpha \leq 1$)

x_i Salida del i-ésimo nodo.

x_j Salida del j-ésimo nodo.

d_j Valor de salida deseado del j-ésima unidad de procesamiento .

Regla de Mínimo Error Cuadrado.

Widrow y Hoff definieron una función que permitía cuantificar el error global cometido en cualquier momento durante el proceso de entrenamiento, agilizando este proceso

$$\text{Error}_{\text{global}} = \frac{1}{2p} \sum_{k=1}^{p} \sum_{i=1}^{n} \left(x_i^{(k)} - d_i^{(k)}\right)^2$$

n = número de nodos de salida
p = número de tramas de entrenamiento

Error cometido en el aprendizaje de la k-ésima trama.

$$\frac{1}{2p}\sum_{i=1}^{n}\left(x_i^{(k)} - d_i^{(k)}\right)^2$$

Se trata de modificar los pesos para que las conexiones de la red minimicen esta función de error, se puede hacer de manera proporcionada a la variación relativa del error.

$$\Delta w_{ij} = k\,(\eth Error_{global}) / \eth w_{ij}$$

Metodología: Entrenamiento de Redes Neuronales para el Reconicimiento de Frutas

Planteamientos Previos

Al margen de la estructura interna de una RNA, para trabajar en el reconocimiento de patrones debemos preocuparnos primeramente de establecer el número de neuronas en la capa de entrada y el número de neuronas en la capa de salida.

Por lo tanto los Datos de Entrada estarán en relación biunívoca con las Neuronas de Entrada y los Datos de Salida con las Neuronas de Salida, ver figura 8

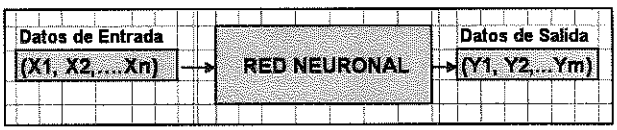


Figura 8: Dependencia Funcional de los Datos de Entrada y Salida

Estableciéndose entre ellos una dependencia funcional que la podemos llamar F donde
[Y1, Y2, ... Ym] = F ([X1, X2, ... Xn])

Representar las Frutas como Datos de Entrada en un Vector Lineal Binario

Los datos de entrada corresponden a los patrones o imágenes de las frutas que se desea reconocer y los datos de salida corresponden a la identificación del patrón o fruta reconocida, en la siguiente figura presentamos algunos ejemplos..

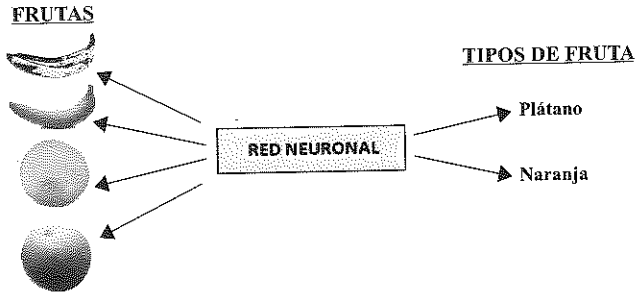


Figura 9: Ejemplos de la Dependencia Funcional de los Datos de Entrada y Salida

Para trabajar en el reconocimiento de patrones debemos encontrar una única estructura de datos que nos permita representar todos los elementos que deseamos reconocer. Para lograrlo, en nuestro caso, hemos seguido los siguientes pasos

Seleccionar la muestra de entrenamiento

Con la finalidad de presentar una explicación sencilla solamente trabajaremos con plátanos y naranjas y tomaremos dos modelos por cada una de ellas.
Por lo tanto, para el plátano elegimos dos imágenes mostradas en la figura 10.

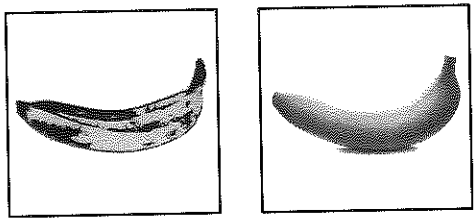
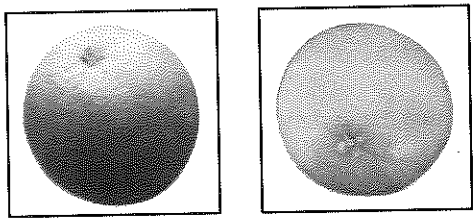


Figura 10: Dos modelos de plátanos

Para la naranja, igualmente elegimos dos imágenes mostradas en la figura 11.



Binarización de las imágenes

Como nuestro objetivo es trabajar solamente con formas y no con clores, eliminamos los colores y la escala de grises y nos quedamos solamente con los colores negro y blanco procurando que se mantengan las formas de nuestras muestras o PATRONES, tal como se muestra en la figura 12.

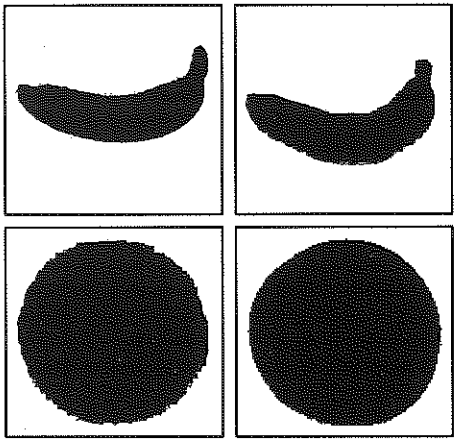


Figura 12: Imágenes binarizadas de las frutas elegidas.

Representar los Patrones en una Matriz

Luego a cada imagen le superponemos una malla (matriz) de cuadrículas lo suficientemente finas tal como se aprecia en la figura 13.

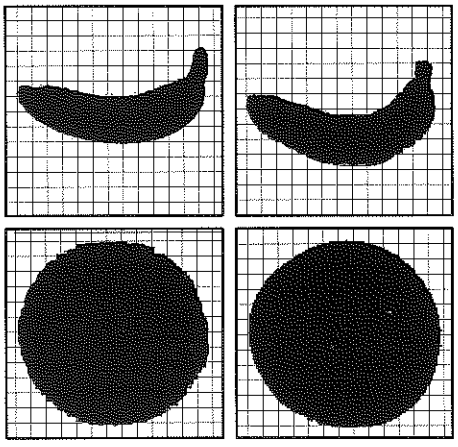


Figura 13: Imágenes binarizadas con Matrices de cuadrículas superpuestas

Por cierto cuanto mayor sea el número de celdas de la matriz superpuesta, la figura que se va a representar será más nítida sobre todo en los bordes. En este caso, si utilizamos por ejemplo una matriz de 20 x 20, la primera imagen del plátano quedaría de la siguiente manera.

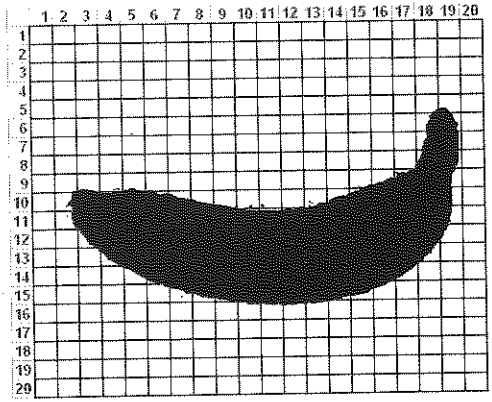


Figura 14: Imágenes binarizadas dentro de una Matriz de 20 x 20

Por cierto cuanto mayor sea el número de celdas de la matriz superpuesta, la figura que se va a representar será más nítida sobre todo en los bordes. En este caso, si utilizamos por ejemplo una matriz de 20 x 20, la primera imagen del plátano quedaría de la siguiente manera.

A cada imagen la asociaremos con una estructura matemática de modo que a cada celda vacía (color blanco) se le asignará un cero y a cada celda llena (color negro) se le asignará un uno, como se aprecia en la figura 15.

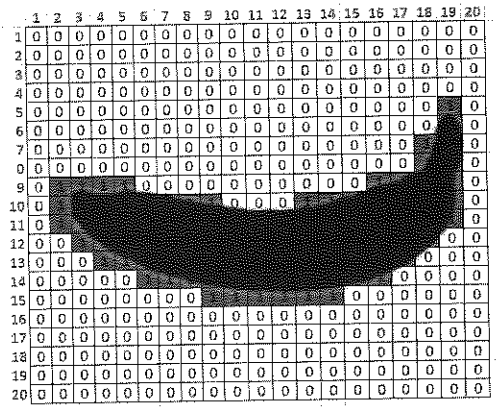


Figura 15: Matriz de 20 x 20 con valores binarios (0,1) para un modelo de fruta.

Por lo tanto la matriz asociada a la imagen elegida quedaría tal como se muestra en la figura 16.

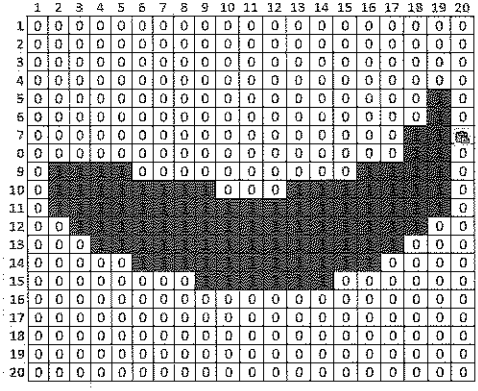


Figura 16: Matriz que representa la fruta elegida.

Finalmente, la figura 17 muestra la fruta elegida con la matriz binaria asociada que la representa, compuesta solamente por los valores 0s y 1s.

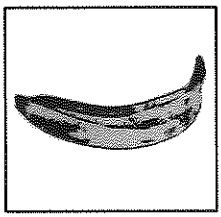
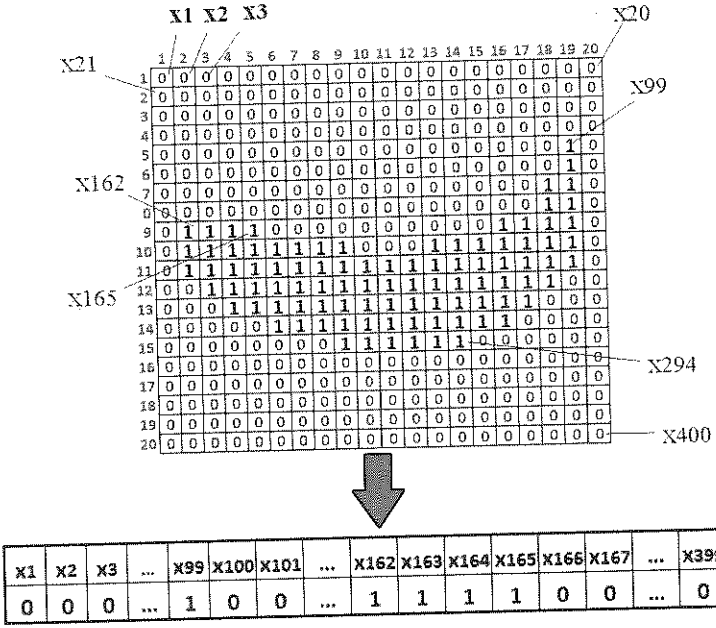


Figura 17: Matriz binaria de 20 x 20 que representa la fruta elegida.

Convertir la Matriz en un Vector Lineal

Como se ha mencionado, los Datos de Entrada deben estar representados por un vector lineal, por esa razón en nuestro caso pasaremos la matriz binaria de 20 x 20 la vamos a transformar a un vector lineal X de tamaño 400 (20x20), cuyos elementos serán, x1, x2, ... x400, según se precisan en la figura 18.



X1	X2	X3	...	X99	X100	X101	...	X162	X163	X164	X165	X166	X167	...	X399	X400
0	0	0	...	1	0	0	...	1	1	1	1	0	0	...	0	0

Figura 18: Matriz de 20 x 20 convertida en un Vector lineal de tamaño 400

Teniendo en cuenta dicha conversión, en la misma figura x9 se pueden apreciar los 400 valores del vector X, que representan el plátano elegido.

Por lo tanto los datos que representan a la fruta estarán constituidos por un vector binario X de tamaño 400 que se puede apreciar en la figura 19. Donde:

Xi = 0 para las celdas blancas y
Xi = 1 para las celdas que contienen la imagen

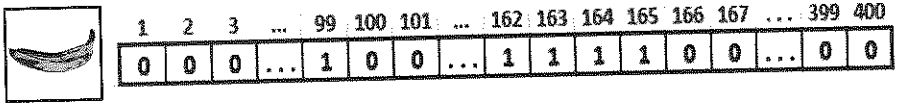


Figura 19: Vector lineal de tamaño 400 que representa la fruta elegida

Representar los Tipos de Fruta como Datos de Salida en un Vector lineal binario

Los datos de salida, para cada fruta, estarán representado por un vector Y, cuya dimensión será equivalente al número de tipos de fruta que queremos reconocer.

En nuestro caso se desean reconocer plátanos y naranjas, entonces la dimensión de nuestro vector de salida será 2, Y = [y1, y2]

Donde:

y1 = 1 si la fruta representada es un plátano, en caso contrario será 0
y2 = 1 si la fruta representada es una naranja, en caso contrario será 0

Por lo tanto, si solamente se trata de reconocer plátanos y naranjas, las posibles salidas serán las mostradas en la figura 20:

Frutas	Y=[y1,y2]
Plátano	[1,0]
Naranja	[0,1]

Figura 20: Vector de salida para el reconocimiento de frutas

Finalmente, en la figura 21 presentamos un ejemplo completo de las estructuras de datos para representar los datos de entrada y salida para el reconocimiento de una fruta.

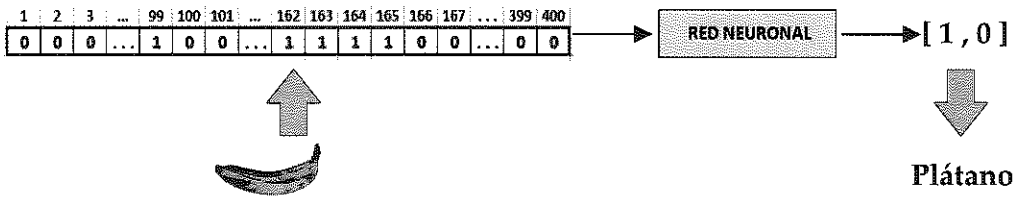


Figura 21: Ejemplo de datos de entrada y salida para el reconocimiento de una fruta

Planteamiento de la Arquitectura de una RNA para el Reconocimiento de Frutas

De acuerdo a lo indicado en los numerales 3.2 y 3.3, el modelo funcional de la RNA para el reconocimientos de frutas (plátanos y naranjas) los Datos de Entrada estarán representados por el vector X=[x1, x2, ... X400] y los Datos de Salida estarán representados por el vector Y= [y1, y2], tal como se puede apreciar en la figura 22.

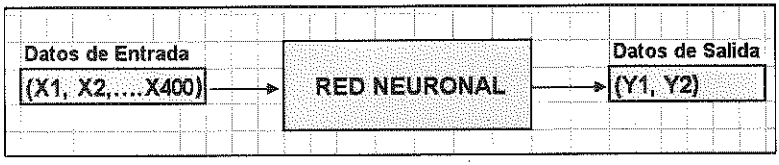


Figura 22: Representación de los Datos de Entrada y Salida

Por lo tanto, para definir la arquitectura de la RNA, debemos considerar que para recibir los 400 valores del vector de entrada, la RNA deberá contar con el mismo número de neuronas de entrada, por esta razón se establece que la RNA contará con 400 neuronas en la capa de entrada. De manera similar, como el vector de salida es de tamaño 2, se establece que la RNA también contará con 2 neuronas en la capa de salida. Ver figura 23.

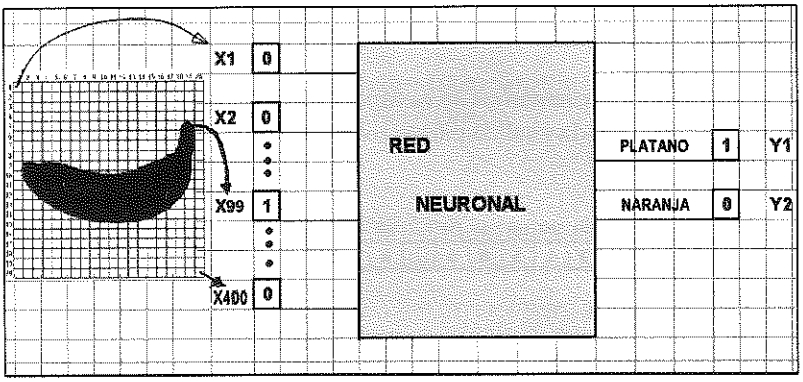


Figura 23: Arquitectura de la RNA para el reconocimiento de Frutas

El número de neuronas en la capa oculta serán asignadas en las pruebas de entrenamiento, y quedará establecida con la cantidad de neuronas que mejor entrenamiento se consiga.

Proceso de Entrenamiento de una RNA para el Reconocimiento de Frutas

Preparar la Base de Conocimientos para el Entrenamiento de la RN

La base de conocimientos (experiencia) llamados también DATOS DE ENTRENAMIENTO para la RNA contiene en forma de una tabla tanto los Datos de Entrada (vector X) como los datos de Salida (vector Y) para cada uno de los patrones (frutas) que se desean reconocer.
Como se ha mencionado, nuestro objetivo es reconocer plátanos y naranjas, para ello, con la finalidad de resumir el procedimiento, solamente hemos tomado dos muestras de cada fruta, las mismas que se presentan en la tabla de la figura 24.

Fruta	Tipo de Fruta
	Platano
	Platano
	Naranja
	Naranja

Figura 24: Modelos de Frutas para el Entrenamiento de una RNA

Con esta información, de acuerdo a lo explicado en el numeral 3.2 generaremos los 400 valores binarios de vector X para cada modelo de fruta, usando la malla de cuadriculas, ver figura 25.

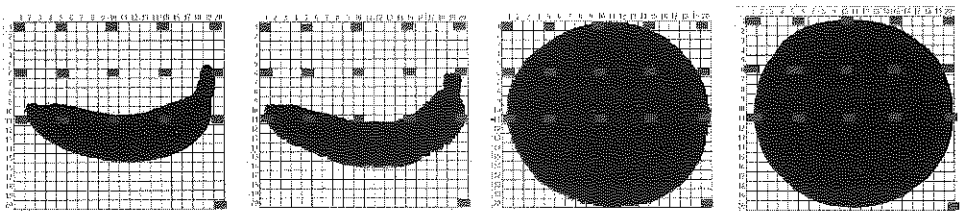


Figura 25: Modelos de Frutas para generar los Datos Entrada para el Entrenamiento de una RNA

Como son 400 valores por cada fruta, por cuestiones de espacio solo mostraremos los valores de las celdas marcadas de color rojo.
En la figura 26 podemos apreciar, la BASE DE CONOCIMIENTO (tabla) que contiene los datos de entrada X y los datos de salida Y, según los patrones (frutas) que se desean reconocer.

BASE DE CONOCIMIENTOS o DATOS DE ENTRENAMIENTO

	Vector de Entrada X de longitud 400																				Vector de Salida Y		
Fruta	X1	X5	X10	X15	X20	X101	X105	X110	X115	X120	X201	X205	X210	X215	X220	X400	Y1	Y2	Tipo
	0	0	0	0	0	0	0	0	0	0	0	1	1	1	0	0	1	0	Platano
	0	0	0	0	0	0	0	0	0	0	1	1	0	1	0	0	1	0	Platano
	0	0	1	0	0	0	1	1	1	0	1	1	1	1	1	0	0	1	Naranja
	0	0	1	0	0	0	1	1	1	0	0	1	1	1	0	0	0	1	Naranja

Figura 26: Base de Conocimientos o Datos de Entrenamiento para el Reconocimiento de los Frutas

Se puede apreciar claramente que en cada línea de la BASE DE CONOCIMIENTO, el vector X representa la imagen de cada fruta, mientras que el vector Y identifica el tipo de fruta.

Finalmente la BASE DE CONOCIMIENTO la guardamos en un archivo plano tipo texto (digamos BASE_CONOCI.TXT) para poderlo usar con el software de entrenamiento.

Entrenamiento de la Red Neuronal

Para entrenar una RNA se debe usar un software, de entre los muchos que existen nosotros usaremos el NEURONTRAINPATTERN.M que está desarrollado en MATLAB (los lectores, si desean pueden solicitar el programa fuente al correo del autor y se les enviará completamente gratis)

A continuación mostraremos con un ejemplo el proceso de entrenamiento usando el software mencionado.

a)Primer Entrenamiento

En el entorno del MATLAB ejecutar el programa NEURONTRAINPATTERN

Introducir nombre de archivo con data: BASE_CONOCI.TXT ==> Base de entrenamiento
Resultados de la lectura del archivo con data
Número de entradas: 400 ==> 400 Neuronas de Entrada
Número de salidas: 3 ==> 2 Neuronas de Salida + 1 de Neurona adicional de trabajo
Número de paquetes de datos entrada-salida: 5 ==> 4 (uno por cada modelo de fruta) + 1 de trabajo
Seleccionar de la Ventana Nueva: Serán generados automáticamente (SINAPSIS)
Introducir neuronas en capa intermedia: 100
Seleccionar de la Ventana nueva: Considera Neurona bias: SI
Introducir ratio de aprendizaje: .05
Introducir momento: 0
Introducir ratio de aprendizaje de exponente a: 0
Introducir ratio de aprendizaje del centro c: 0
Introducir el valor máximo del error (%): 10
Introducir el máximo etapas de aprendizaje: 200
erreltotal = 1.9644
...
erreltotal = 0.7141 ==> 0.7141 * 100 = 71.41 % de margen de error de la RNA
Introducir nombre del archivo donde se guardará información de la red: ENTRENAMIENTO1.prn

El software nos proporciona la imagen mostrada en la figura 27, donde se puede apreciar el seguimiento del proceso de aprendizaje de la RNA.

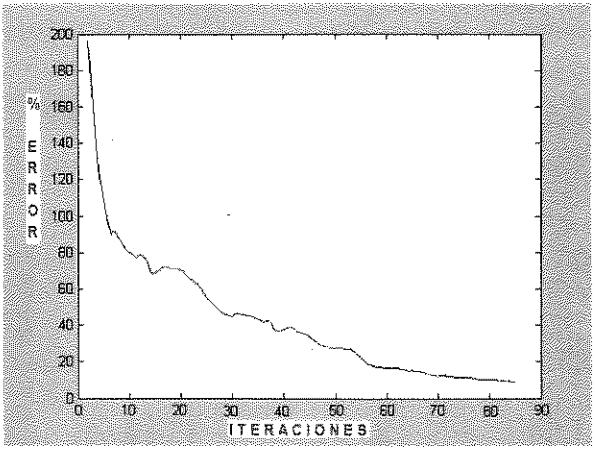


Figura 27: Curva del Aprendizaje de la RNA

Como podemos apreciar en la gráfica, el proceso de entrenamiento no ha sido satisfactorio ya que se deseaba alcanzar un margen de error del 10 % en un total máximo de 200 iteraciones, pero luego de las 200 iteraciones solo ha llegado al 71.41% del margen de error.
En dicha imagen, también se puede apreciar que aproximadamente en iteración 65 llegó al 40% de error, aunque con un aprendizaje bastante irregular; en las iteraciones subsiguientes, en lugar de seguir disminuyendo dicho margen de error, empezó a crecer, también de modo irregular, por lo tanto, este entrenamiento se considera como no exitoso.

a)Entrenamiento Óptimo
De manera similar realizamos un total de 17 pruebas, buscando obtener el entrenamiento más óptimo, cambiando los parámetros de entrada requeridos por el software, así conseguimos un entrenamiento muy adecuado que lo detallamos a continuación.

En el entorno del MATLAB ejecutar el programa NEURONTRAINPATTERN

Introducir nombre de archivo con data: BASE_CONOCI.TXT ==> Base de entrenamiento
Resultados de la lectura del archivo con data
Número de entradas: 400 ==> 400 Neuronas de Entrada
Número de salidas: 3 ==> 2 Neuronas de Salida + 1 de Neurona adicional de trabajo
Número de paquetes de datos entrada-salida: 5 ==> 4 (uno por cada modelo de fruta) + 1 de trabajo
Seleccionar de la Ventana Nueva: Serán generados automáticamente (SINAPSIS)
Introducir neuronas en capa intermedia: 500
Seleccionar de la Ventana nueva: Considera Neurona bias: SI
Introducir ratio de aprendizaje: 0.02
Introducir momento: 0
Introducir ratio de aprendizaje de exponente a: 0
Introducir ratio de aprendizaje del centro c: 0

Introducir el valor máximo del error (%): 10
Introducir el máximo etapas de aprendizaje: 200
erreltotal= 1.9644
...
erreltotal= 0.0988 ==> 0.0988 * 100 = 9.88 % de margen de error de la RNA
Introducir nombre del archivo donde se guardará información de la red: ENTRENAMIENTO1.prn

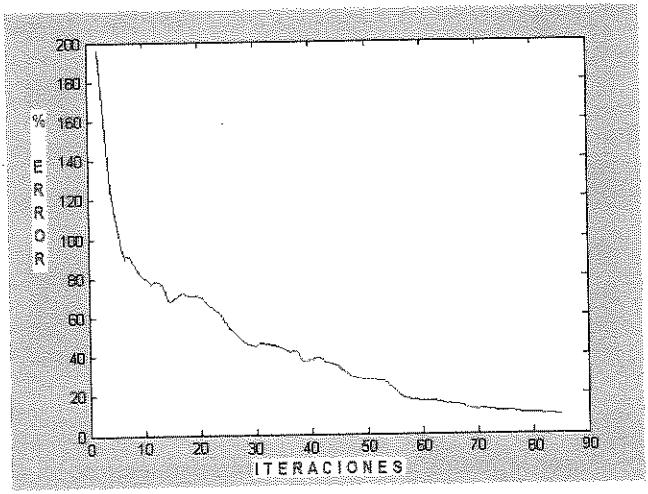


Figura 28: Curva del Aprendizaje de la RNA

Como podemos apreciar en la figura 28, esta vez hemos alcanzado el 10 % de margen de error en tan solo 86 iteraciones. Es decir hemos logrado nuestro objetivo y podemos decir que ahora tenemos una RNA bien entrenada para el reconocimiento de frutas (plátanos y naranjas).

Debemos tener presente que el entrenamiento concluye guardando en un archivo tipo texto todos los pesos sinápticos de la red neuronal, nosotros, como podemos apreciarlo en el último proceso de entrenamiento, lo hemos guardado en el archivo ENTRENAMIENTO1.prn. Este archivo será utilizado más adelante, cundo se realice el PROCESO DE RECONOCIMIENTO de una fruta.

Proceso de Reconocimiento de una Fruta utilizando una RNA Entrenada

Elegir las frutas para ser reconocidas por la RNA entrenada

Ahora, la idea es presentarle a la RNA entrenada imágenes nuevas de frutas y pedirle que reconozca que fruta es.

En nuestro caso utilizaremos una nueva imagen de un plátano, ver figura 29.

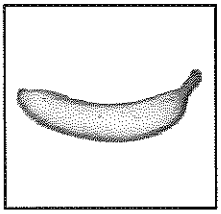


Figura 29: Nueva Imagen para ser reconocida por la RNA entrenada

Aunque obviamente, la imagen en muy parecida a los modelos anteriores, utilizadas para el entrenamiento, de hecho que tiene ciertas diferencias, las mismas que fácilmente se pueden percibir apreciándolas conjuntamente, ver figura 30.

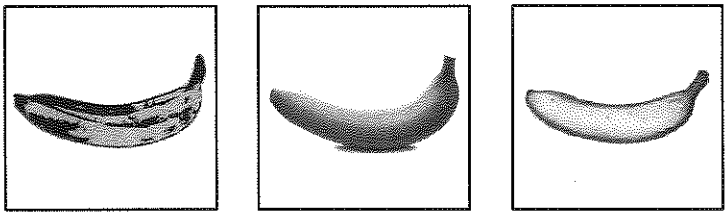


Figura 30: Comparación de la Nueva Imagen con las anteriores

Preparar los datos de Entada para el Reconocimiento de Frutas

Como se explicó en el numeral 3.2, primeramente generamos la matriz binarizada, tal como se aprecia en la figura 31.

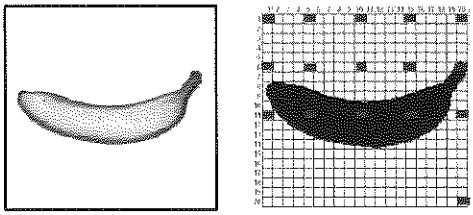


Figura 31: Nueva Imagen binarizada para generar Datos de Reconocimiento para la RNA

Luego generamos los 400 valores del vector de entrada X. Los valores del vector Y serán [0,0] ya que no debemos informarle a la RNA que fruta es, ya que la idea es, justamente, que la RNA al finalizar el proceso de reconocimiento identifique el tipo de fruta que ha reconocido, a continuación, en la figura 32 podemos apreciar los valores de los vectores X e Y.

DATOS DE ENTRADA PARA EL RECONOCIMIENTO DE FRUTAS																								
	Vector de Entrada X de longitud 400																				Vector de Salida Y		
Fruta	X1	X5	X10	X15	X20	X100	X105	X110	X115	X120	X201	X205	X210	X215	X220	X400	Y1	Y2	Tipo
	0	0	0	0	0	0	0	0	0	1	0	1	1	1	0	0	0	0	??

Figura 32: Datos para el Reconocimiento de Frutas

Finalmente los datos para el reconocimiento lo guardamos en un archivo plano tipo texto (digamos BASE_RECONOCI.TXT) para poderlo usar con el software NEURONTRAINPATTERN.

Reconocimiento de la fruta mediante la RNA entrenada.

El proceso de reconocimiento es muy similar al entrenamiento, a continuación lo describimos pasp a paso:

En el entorno del MATLAB ejecutar el programa NEURONTRAINPATTERN

Introducir nombre de archivo con data: BASE_RECONOCI.TXT ==> Datos para Reconocimiento
Resultados de la lectura del archivo con data
Número de entradas: 400 ==> 400 Neuronas de Entrada
Número de salidas: 3 ==> 2 Neuronas de Salida + 1 de Neurona adicional de trabajo
Número de paquetes de datos entrada-salida: 5 ==> 4 (uno por cada modelo de fruta) + 1 de trabajo
Seleccionar de la Ventana nueva: Se leerán desde un archivo (SINAPSIS)
Introducir nombre de archivo con información de la red: ENTRENAMIENTO1.prn ==> Archivo que fue generado y almacenado al final del proceso de entrenamiento
Introducir neuronas en capa intermedia: 100
Neurona bias en capa de entrada (1:SI) (0:NO) : 1
Introducir ratio de aprendizaje: .0
Introducir momento: 0
Introducir ratio de aprendizaje de exponente a: 0
Introducir ratio de aprendizaje del centro c: 0
Introducir el valor máximo del error (%): 5
Introducir el máximo etapas de aprendizaje: 1
Introducir nombre del archivo donde se guardará información de la red: uu

Interpretación de la Matriz de Salida

Al concluir el proceso de reconocimiento, el software tiene preparado una matriz de salida que la obtendremos e interpretaremos de la siguiente manera:

>>output
0.87 0.021

Estos son los valores para de salida del vector Y que pasados al Excel se apreciaría como se

Y1	Y2
0.87	0.021

Y1	Y2
1	0

Figura 33: Vector de Salida Y con Valores Reales y Valores Redondeados

Estos valores proporcionados por el software al concluir el proceso de reconocimiento se interpretan de la siguiente manera:

- Y1=0.87 quiere decir que los datos de entrada, ingresados en el proceso de reconocimiento, los mismos que fueron generados a partir de la nueva imagen han sido reconocidos en un 87 % como el primer patrón Y1 que se trata precisamente de un PLATANO.

- Así mismo se puede decir que dichos datos también han sido reconocidos en un 2.1 % como el segundo patrón Y2; es decir que nuestra última imagen del plátano se parece en un 2.1 % a una NARANJA.

- Si redondeamos los valores de salidas y hacemos un análisis más práctico diremos que la nueva imagen analizada ha sido reconocida como un PLATANO y no ha sido reconocida cono una NARANJA.

CONCLUSIONES.

·Como resultado de este trabajo podemos concluir que el uso de las Redes Neuronales es una gran alternativa para la solución de muchos problemas.

·Usando como base los resultados mostrados podemos resolver problemas más complejos como por ejemplo, reconocimiento de los patrones de la conducta humana, reconocimiento de enfermedades cardiacas, reconocimiento de productos de calidad etc.

·Respecto al nuestro trabajo anterior, correspondiente al reconocimiento de caracteres numéricos, hemos dado un paso más adelante pero sabemos que cada avance es un nuevo punto de inicio.

REFERENCIAS BIBLIOGRÁFICAS.

BISHOP C, 2006. Neural Networks for Pattern Recognition. New York. Oxford University Press. 504 p.

CORTEZ VÁSQUEZ, 1999. Matemáticas discretas para ciencias de la computación. UNMSM Lima-Perú

DOWLA F, ROGERS L, 1996. Solving problems in environmental engineering and geosciences with artificial neural networks. Cambridge. MIT Press. 310 p.

HILERA J, MARTINEZ V, 1995. Redes neuronales artificiales: fundamentos, modelos y aplicaciones. Madrid: Addison-Wesley Iberoamericana. RA-MA. 390 p.

SÁNCHEZ E, ALANIS A, 2006. Redes neuronales: conceptos fundamentales y aplicaciones a control automático. Madrid. Prentice-Hall. 210 p.