Universidad Tecnológica Nacional – Facultad Regional Rosario
Departamento de Ingeniería Química
Grupo de Investigación Aplicada a la Ingeniería Química (GIAIQ)

Cátedra:
Informática Aplicada a la Ingeniería de Procesos – Orientación I

Tema:

Redes Neuronales: Conceptos Básicos y Aplicaciones.

Profesores:
Carlos Alberto Ruiz
Marta Susana Basualdo

Autor:
Damián Jorge Matich

Marzo de 2001

Rosario – Marzo del 2001

0 Contenidos

Contenidos *2*

Introducción *4*

1.1 Acerca de las redes neuronales. **4**

Reseña Histórica *6*

2.1 Historia de las redes neuronales. **6**

Generalidades *8*

3.1 Definiciones de una red neuronal. **8**

3.2 Ventajas que ofrecen las red neuronal. **8**

3.2.1 Aprendizaje adaptativo. 9

3.2.2 Auto-organización. 9

3.2.3 Tolerancia a fallos. 9

3.2.4 Operación en tiempo real. 10

3.2.5 Fácil inserción dentro de la tecnología existente. 10

3.3 Redes neuronales y computadoras digitales. **10**

Elementos Básicos *12*

4.1 Elementos básicos que componen una red neuronal. **12**

4.2 Función de entrada (*input function*). **12**

4.3 Función de activación (*activation function*). **14**

4.4 Función de salida (*output function*). **15**

Aprendizaje, Validación y Codificación *16*

5.1 Niveles o capas de una red neuronal. **16**

5.2 Tipos de neuronas artificiales. **16**

5.3 Técnicas de decisión. **16**

5.4 Mecanismos de aprendizaje. **18**

5.4.1 Aprendizaje supervisado. 19

5.4.1.1 Aprendizaje por corrección de error. 20

5.4.1.2 Aprendizaje por refuerzo. 20

5.4.1.3 Aprendizaje estocástico. 21

5.4.2 Aprendizaje no supervisado. 21

5.4.2.1 Aprendizaje hebbiano. 22

5.4.2.2 Aprendizaje competitivo y comparativo. 22

5.5 Elección del conjunto inicial de pesos. **22**

5.6 Detención del proceso de aprendizaje. **22**

5.7 Codificación de los datos de entrada. **23**

5.7.1 Codificación de los atributos numéricos. 23

5.7.2 Codificación de los atributos simbólicos. 24

5.7.3 Resumen de los procedimientos de codificación. 25

5.8 Validación de la red neuronal. **25**

5.9 Cuestiones a resolver al trabajar con una red neuronal..... 25

Principales Topologías..... 27

6.1 Topología de las redes neuronales. 27

6.2 Redes monocapa. 27

6.3 Redes multicapa. 27

6.4 Conexión entre neuronas. 27

6.5 Redes de propagación hacia atrás (*backpropagation*). 28

6.5.1 Ejemplo. 28

6.6 Estructura de la Red Hopfield. 29

6.7 Simulated Annealing aplicada a una Red Hopfield. 30

6.8 Asociaciones entre la información de entrada y salida. 31

6.8.1 Redes heteroasociativas. 32

6.8.2 Redes autoasociativas. 32

Aplicaciones 34

7.1 Aplicaciones de las redes neuronales. 34

7.1.1 Asociación y clasificación. 35

7.1.2 Regeneración de patrones. 36

7.1.3 Regeneración y generalización. 36

7.1.4 Optimización. 36

7.2 Casos concretos de aplicación. 36

7.2.1 Planificación del staff (cuerpo) de empleados. 37

7.2.2 Planificación de la demanda de materiales. 38

7.2.3 Puntuación para la solicitud de un crédito. 40

Software Comerciales 43

8.1 Aplicaciones del NeurOn-Line Studio a procesos de refinería y petroquímica. 43

8.2 Software que pueden ser empleados en la industria de procesos. 54

Bibliografía 55

1 Introducción

1.1 Acerca de las redes neuronales.

El hombre se ha caracterizado siempre por su búsqueda constante de nuevas vías para mejorar sus condiciones de vida. Estos esfuerzos le han servido para reducir el trabajo en aquellas operaciones en las que la fuerza juega un papel primordial. Los progresos obtenidos han permitido dirigir estos esfuerzos a otros campos, como por ejemplo, a la construcción de máquinas calculadoras que ayuden a resolver de forma automática y rápida determinadas operaciones que resultan tediosas cuando se realizan a mano.

Uno de los primeros en acometer esta empresa fue Charles Babbage, quien trató infructuosamente de construir una máquina capaz de resolver problemas matemáticos. Posteriormente otros tantos intentaron construir máquinas similares, pero no fue hasta la Segunda Guerra Mundial, cuando ya se disponía de instrumentos electrónicos, que se empezaron a recoger los primeros frutos. En 1946 se construyó la primera computadora electrónica, ENIAC. Desde entonces los desarrollos en este campo han tenido un auge espectacular.

Estas máquinas permiten implementar fácilmente algoritmos para resolver multitud de problemas que antes resultaban engorrosos de resolver. Sin embargo, se observa una limitación importante: ¿qué ocurre cuando el problema que se quiere resolver no admite un tratamiento algorítmico, como es el caso, por ejemplo, de la clasificación de objetos por rasgos comunes? Este ejemplo demuestra que la construcción de nuevas máquinas más versátiles requiere un enfoque del problema desde otro punto de vista. Los desarrollos actuales de los científicos se dirigen al estudio de las capacidades humanas como una fuente de nuevas ideas para el diseño de las nuevas máquinas. 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 son más que otra forma de emular ciertas características propias de los humanos, como la capacidad de memorizar y de asociar hechos. Si se examinan con atención aquellos problemas que no pueden expresarse a través de un algoritmo, se observará que todos ellos tienen una característica en común: la experiencia. El hombre es capaz de resolver estas situaciones acudiendo a la experiencia acumulada. Así, parece claro que una forma de aproximarse al problema consista en la construcción de sistemas que sean capaces de reproducir esta característica humana. En definitiva, las redes neuronales no son más que un modelo artificial y simplificado del cerebro humano, que es el ejemplo más perfecto del que disponemos para un sistema que es capaz de adquirir conocimiento a través de la experiencia. Una red neuronal es "un nuevo sistema para el tratamiento de la información, cuya unidad básica de procesamiento está inspirada en la célula fundamental del sistema nervioso humano: *la neurona*".

Todos los procesos del cuerpo humano se relacionan en alguna u otra forma con la (in)actividad de estas neuronas. Las mismas son un componente relativamente simple del ser humano, pero cuando millares de ellas se conectan en forma conjunta se hacen muy poderosas.

Lo que básicamente ocurre en una neurona biológica es lo siguiente: la neurona es estimulada o excitada a través de sus *entradas* (inputs) y cuando se alcanza un cierto umbral, la neurona se dispara o activa, pasando una señal hacia el *axon*. Posteriores investigaciones condujeron al descubrimiento de que estos procesos son el resultado de eventos electroquímicos.

Como ya se sabe, el pensamiento tiene lugar en el cerebro, que consta de billones de neuronas interconectadas. Así, el secreto de la "inteligencia" -sin importar como se defina- se sitúa dentro de estas neuronas interconectadas y de su interacción. También, es bien conocido que los humanos son capaces de aprender. Aprendizaje significa que aquellos problemas que inicialmente no pueden resolverse, pueden ser resueltos después de obtener más información acerca del problema. Por lo tanto, las Redes Neuronales...

- Consisten de unidades de procesamiento que intercambian datos o información.
- Se utilizan para reconocer patrones, incluyendo imágenes, manuscritos y secuencias de tiempo (por ejemplo: tendencias financieras).
- Tienen capacidad de aprender y mejorar su funcionamiento.

Una primera clasificación de los modelos de redes neuronales podría ser, atendiendo a su similitud con la realidad biológica:

1) El modelo de tipo biológico. Este comprende las redes que tratan de simular los sistemas neuronales biológicos, así como las funciones auditivas o algunas funciones básicas de la visión.

2) El modelo dirigido a aplicación. Este modelo no tiene por qué guardar similitud con los sistemas biológicos. Su arquitectura está fuertemente ligada a las necesidades de las aplicaciones para la que es diseñada.

2 Reseña Histórica

2.1 Historia de las redes neuronales.

1936 - Alan Turing. Fue el primero en estudiar el cerebro como una forma de ver el mundo de la computación. Sin embargo, los primeros teóricos que concibieron los fundamentos de la computación neuronal fueron Warren McCulloch, un neurofisiólogo, y Walter Pitts, un matemático, quienes, en 1943, lanzaron una teoría acerca de la forma de trabajar de las neuronas (Un Cálculo Lógico de la Inminente Idea de la Actividad Nerviosa - Boletín de Matemática Biofísica 5: 115-133). Ellos modelaron una red neuronal simple mediante circuitos eléctricos.

1949 - Donald Hebb. Fue el primero en explicar los procesos del aprendizaje (que es el elemento básico de la inteligencia humana) desde un punto de vista psicológico, desarrollando una regla de como el aprendizaje ocurría. Aun hoy, este es el fundamento de la mayoría de las funciones de aprendizaje que pueden hallarse en una red neuronal. Su idea fue que el aprendizaje ocurría cuando ciertos cambios en una neurona eran activados. También intentó encontrar semejanzas entre el aprendizaje y la actividad nerviosa. Los trabajos de Hebb formaron las bases de la Teoría de las Redes Neuronales.

1950 - Karl Lashley. En sus series de ensayos, encontró que la información no era almacenada en forma centralizada en el cerebro sino que era distribuida encima de él.

1956 - Congreso de Dartmouth. Este Congreso frecuentemente se menciona para indicar el nacimiento de la inteligencia artificial.

1957 - Frank Rosenblatt. Comenzó el desarrollo del Perceptron. Esta es la red neuronal más antigua; utilizándose hoy en día para aplicación como identificador de patrones. Este modelo era capaz de generalizar, es decir, después de haber aprendido una serie de patrones podía reconocer otros similares, aunque no se le hubiesen presentado en el entrenamiento. Sin embargo, tenía una serie de limitaciones, por ejemplo, su incapacidad para resolver el problema de la función OR-exclusiva y, en general, era incapaz de clasificar clases no separables linealmente.

1959 - Frank Rosenblatt: Principios de Neurodinámica. En este libro confirmó que, bajo ciertas condiciones, el aprendizaje del Perceptron convergía hacia un estado finito (Teorema de Convergencia del Perceptron).

1960 - Bernard Widroff/Marcian Hoff. Desarrollaron el modelo Adaline (ADAptative LINear Elements). Esta fue la primera red neuronal aplicada a un problema real (filtros adaptativos para eliminar ecos en las líneas telefónicas) que se ha utilizado comercialmente durante varias décadas.

1961 - Karl Steinbeck: Die Lernmatrix. Red neuronal para simples realizaciones técnicas (memoria asociativa).

1969 - Marvin Minsky/Seymour Papert. En este año casi se produjo la “muerte abrupta” de las Redes Neuronales; ya que Minsky y Papert probaron (matemáticamente) que el Perceptrons no era capaz de resolver problemas relativamente fáciles, tales como

el aprendizaje de una función no-lineal. Esto demostró que el Perceptron era muy débil, dado que las funciones no-lineales son extensamente empleadas en computación y en los problemas del mundo real.

1974 - Paul Werbos. Desarrolló la idea básica del algoritmo de aprendizaje de *propagación hacia atrás* (backpropagation); cuyo significado quedó definitivamente aclarado en 1985.

1977 - Stephen Grossberg: Teoría de Resonancia Adaptada (TRA). La Teoría de Resonancia Adaptada es una arquitectura de red que se diferencia de todas las demás previamente inventadas. La misma simula otras habilidades del cerebro: memoria a largo y corto plazo.

1985 - John Hopfield. Provocó el renacimiento de las redes neuronales con su libro: “Computación neuronal de decisiones en problemas de optimización.”

1986 - David Rumelhart/G. Hinton. Redescubrieron el algoritmo de aprendizaje de propagación hacia atrás (backpropagation).

A partir de 1986, el panorama fue alentador con respecto a las investigaciones y el desarrollo de las redes neuronales. En la actualidad, son numerosos los trabajos que se realizan y publican cada año, las aplicaciones nuevas que surgen (sobretodo en el área de control) y las empresas que lanzan al mercado productos nuevos, tanto hardware como software (sobre todo para simulación).

3 Generalidades

3.1 Definiciones de una red neuronal.

Existen numerosas formas de definir a las redes neuronales; desde las definiciones cortas y genéricas hasta las que intentan explicar más detalladamente qué son las redes neuronales. Por ejemplo:

1) Una nueva forma de computación, inspirada en modelos biológicos.

2) Un modelo matemático compuesto por un gran número de elementos procesales organizados en niveles.

3) ...un sistema de computación compuesto por un gran número de elementos simples, elementos de procesos muy interconectados, los cuales procesan información por medio de su estado dinámico como respuesta a entradas externas.

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

3.2 Ventajas que ofrecen las red neuronal.

Debido a su constitución y a sus fundamentos, las redes neuronales artificiales presentan un gran número de características semejantes a las del cerebro. Por ejemplo, son capaces de aprender de la experiencia, de generalizar de casos anteriores a nuevos casos, de abstraer características esenciales a partir de entradas que representan información irrelevante, etc. Esto hace que ofrezcan numerosas ventajas y que este tipo de tecnología se esté aplicando en múltiples áreas. Entre las ventajas se incluyen:

- Aprendizaje Adaptativo. Capacidad de aprender a realizar tareas basadas en un entrenamiento o en una experiencia inicial.
- Auto-organización. Una red neuronal puede crear su propia organización o representación de la información que recibe mediante una etapa de aprendizaje.
- Tolerancia a fallos. La destrucción parcial de una red conduce a una degradación de su estructura; sin embargo, algunas capacidades de la red se pueden retener, incluso sufriendo un gran daño.
- Operación en tiempo real. Los cómputos neuronales pueden ser realizados en paralelo; para esto se diseñan y fabrican máquinas con hardware especial para obtener esta capacidad.
- Fácil inserción dentro de la tecnología existente. Se pueden obtener chips especializados para redes neuronales que mejoran su capacidad en ciertas tareas. Ello facilitará la integración modular en los sistemas existentes.

3.2.1 Aprendizaje adaptativo.

La capacidad de aprendizaje adaptativo es una de las características más atractivas de redes neuronales. Esto es, aprenden a llevar a cabo ciertas tareas mediante un entrenamiento con ejemplos ilustrativos.

Como las redes neuronales pueden aprender a diferenciar patrones mediante ejemplos y entrenamientos, no es necesario elaborar modelos a priori ni necesidad de especificar funciones de distribución de probabilidad.

Las redes neuronales son sistemas dinámicos autoadaptativos. Son adaptables debido a la capacidad de autoajuste de los elementos procesales (neuronas) que componen el sistema. Son dinámicos, pues son capaces de estar constantemente cambiando para adaptarse a las nuevas condiciones.

En el proceso de aprendizaje, los enlaces ponderados de las neuronas se ajustan de manera que se obtengan ciertos resultados específicos. Una red neuronal no necesita un algoritmo para resolver un problema, ya que ella puede generar su propia distribución de pesos en los enlaces mediante el aprendizaje. También existen redes que continúan aprendiendo a lo largo de su vida, después de completado su período de entrenamiento.

La función del diseñador es únicamente la obtención de la arquitectura apropiada. No es problema del diseñador el cómo la red aprenderá a discriminar. Sin embargo, sí es necesario que desarrolle un buen algoritmo de aprendizaje que le proporcione a la red la capacidad de discriminar, mediante un entrenamiento con patrones.

3.2.2 Auto-organización.

Las redes neuronales emplean su capacidad de aprendizaje adaptativo para autoorganizar la información que reciben durante el aprendizaje y/o la operación. Mientras que el aprendizaje es la modificación de cada elemento procesal, la autoorganización consiste en la modificación de la red neuronal completa para llevar a cabo un objetivo específico.

Cuando las redes neuronales se usan para reconocer ciertas clases de patrones, ellas autoorganizan la información usada. Por ejemplo, la red llamada backpropagation, creará su propia representación característica, mediante la cual puede reconocer ciertos patrones.

Esta autoorganización provoca la generalización: facultad de las redes neuronales de responder apropiadamente cuando se les presentan datos o situaciones a las que no había sido expuesta anteriormente. El sistema puede generalizar la entrada para obtener una respuesta. Esta característica es muy importante cuando se tiene que solucionar problemas en los cuales la información de entrada no es muy clara; además permite que el sistema dé una solución, incluso cuando la información de entrada está especificada de forma incompleta.

3.2.3 Tolerancia a fallos.

Las redes neuronales fueron los primeros métodos computacionales con la capacidad inherente de tolerancia a fallos. Comparados con los sistemas

computacionales tradicionales, los cuales pierden su funcionalidad cuando sufren un pequeño error de memoria, en las redes neuronales, si se produce un fallo en un número no muy grande de neuronas y aunque el comportamiento del sistema se ve influenciado, no sufre una caída repentina.

Hay dos aspectos distintos respecto a la tolerancia a fallos:

a) Las redes pueden aprender a reconocer patrones con ruido, distorsionados o incompletos. Esta es una tolerancia a fallos respecto a los datos.

b) Las redes pueden seguir realizando su función (con cierta degradación) aunque se destruya parte de la red.

La razón por la que las redes neuronales son tolerantes a los fallos es que tienen su información distribuida en las conexiones entre neuronas, existiendo cierto grado de redundancia en este tipo de almacenamiento. La mayoría de los ordenadores algorítmicos y sistemas de recuperación de datos almacenan cada pieza de información en un espacio único, localizado y direccionable. En cambio, las redes neuronales almacenan información no localizada. Por lo tanto, la mayoría de las interconexiones entre los nodos de la red tendrán sus valores en función de los estímulos recibidos, y se generará un patrón de salida que represente la información almacenada.

3.2.4 Operación en tiempo real.

Una de las mayores prioridades, casi en la totalidad de las áreas de aplicación, es la necesidad de realizar procesos con datos de forma muy rápida. Las redes neuronales se adaptan bien a esto debido a su implementación paralela. Para que la mayoría de las redes puedan operar en un entorno de tiempo real, la necesidad de cambio en los pesos de las conexiones o entrenamiento es mínimo.

3.2.5 Fácil inserción dentro de la tecnología existente.

Una red individual puede ser entrenada para desarrollar una única y bien definida tarea (tareas complejas, que hagan múltiples selecciones de patrones, requerirán sistemas de redes interconectadas). Con las herramientas computacionales existentes (no del tipo PC), una red puede ser rápidamente entrenada, comprobada, verificada y trasladada a una implementación hardware de bajo coste. Por lo tanto, no se presentan dificultades para la inserción de redes neuronales en aplicaciones específicas, por ejemplo de control, dentro de los sistemas existentes. De esta manera, las redes neuronales se pueden utilizar para mejorar sistemas en forma incremental y cada paso puede ser evaluado antes de acometer un desarrollo más amplio.

3.3 Redes neuronales y computadoras digitales.

Para entender el potencial de la computación neuronal, sería necesario hacer una breve distinción entre sistemas de computación neuronales y digitales: *los sistemas neurológicos no aplican principios de circuitos lógicos o digitales.*

Un sistema de computación digital debe ser síncrono o asíncrono. Si fuera asíncrono, la duración de los impulsos neuronales debería ser variable para mantener uno de los valores binarios por periodos de tiempo indefinido, lo cual no es el caso. Si el

principio fuera síncrono, se necesitaría un reloj global o maestro con el cual los pulsos estén sincronizados. Éste tampoco es el caso. Las neuronas no pueden ser circuitos de umbral lógico, porque hay miles de entradas variables en la mayoría de las neuronas y el umbral es variable con el tiempo, siendo afectado por la estimulación, atenuación, etc. La precisión y estabilidad de tales circuitos no es suficiente para definir ninguna función booleana. Los procesos colectivos que son importantes en computación neuronal no pueden implementarse por computación digital. Por todo ello, el cerebro debe ser un computador analógico.

Ni las neuronas ni las sinapsis son elementos de *memoria biestable*. Todos los hechos fisiológicos hablan a favor de las acciones de las neuronas como integradores analógicos, y la eficiencia de la sinapsis cambia de forma gradual, lo cual no es característico de sistemas biestables.

Los circuitos del cerebro no implementan *computación recursiva* y por lo tanto no son *algorítmicos*. Debido a los problemas de estabilidad, los circuitos neuronales no son suficientemente estables para definiciones recursivas de funciones como en computación digital. Un algoritmo, por definición, define una función recursiva.

4 Elementos Básicos

4.1 Elementos básicos que componen una red neuronal.

A continuación se puede ver, en la *Figura 4.1*, un esquema de una red neuronal:

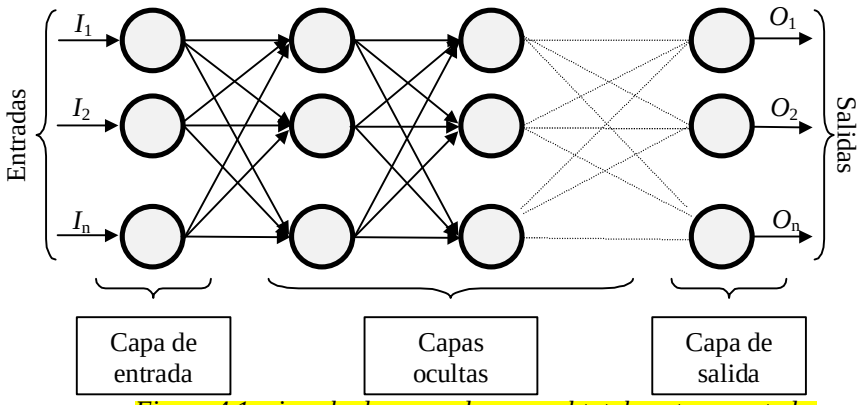

Figura 4.1: ejemplo de una red neuronal totalmente conectada.

La misma está constituida por neuronas interconectadas y arregladas en tres capas (esto último puede variar). Los datos ingresan por medio de la "*capa de entrada*", pasan a través de la "*capa oculta*" y salen por la "*capa de salida*". Cabe mencionar que la capa oculta puede estar constituida por varias capas.

Antes de comenzar el estudio sobre las redes neuronales, se debe aprender algo sobre las neuronas y de cómo ellas son utilizadas por una red neuronal. En la *Figura 4.2* se compara una neurona biológica con una neurona artificial. En la misma se pueden observar las similitudes entre ambas (tienen entradas, utilizan pesos y generan salidas).

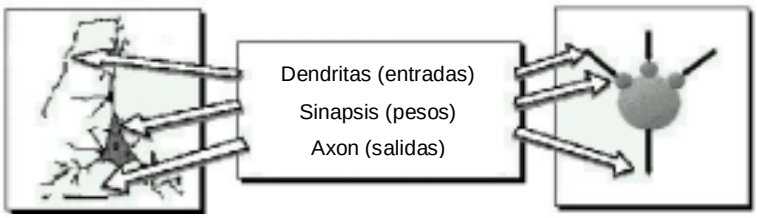

Figura 4.2: comparación entre una neurona biológica (izquierda) y una artificial (derecha).

Mientras una neurona es muy pequeña en sí misma, cuando se combinan cientos, miles o millones de ellas pueden resolver problemas muy complejos. Por ejemplo el cerebro humano se compone de billones de tales neuronas.

4.2 Función de entrada (*input function*).

La neurona trata a muchos valores de entrada como si fueran uno solo; esto recibe el nombre de *entrada global*. Por lo tanto, ahora nos enfrentamos al problema de cómo se pueden combinar estas simples entradas (in_{i1}, in_{i2}, ...) dentro de la entrada

global, gin_i. Esto se logra a través de la *función de entrada*, la cual se calcula a partir del *vector entrada*. La función de entrada puede describirse como sigue:

$$input_i = (in_{i1} \bullet w_{i1})^* (in_{i2} \bullet w_{i2})^* \ldots (in_{in} \bullet w_{in})$$

donde: * representa al operador apropiado (por ejemplo: máximo, sumatoria, productoria, etc.), n al número de entradas a la neurona N_i y w_i al peso.

Los valores de entrada se multiplican por los pesos anteriormente ingresados a la neurona. Por consiguiente, los pesos que generalmente no están restringidos cambian la medida de influencia que tienen los valores de entrada. Es decir, que permiten que un gran valor de entrada tenga solamente una pequeña influencia, si estos son lo suficientemente pequeños.

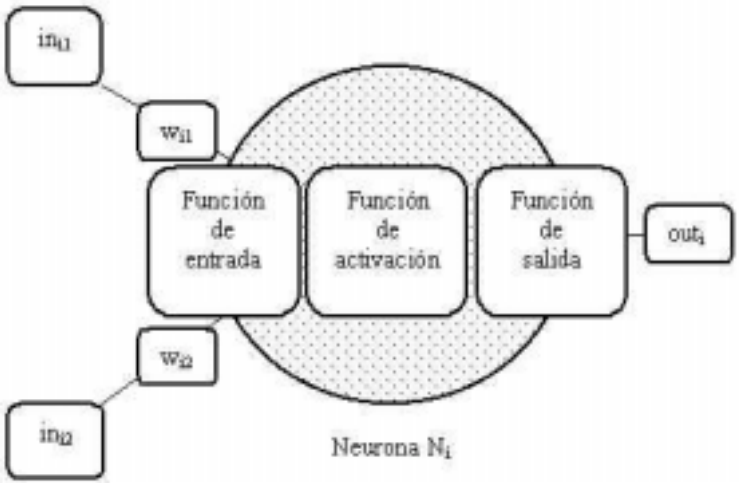

Figura 4.3: ejemplo de una neurona con 2 entradas y 1 salida.

La nomenclatura utilizada en la *Figura 4.3* es la siguiente: in_{i1} = entrada número 1 a la neurona N_i; w_{i1} = peso correspondiente a in_{i1}; in_{i2} = entrada número 2 a la neurona N_i; w_{i2} = peso correspondiente a in_{i2}; y out_i = salida de la neurona N_i. El conjunto de todas las n entradas $in_i = (in_{i1}, in_{i2}, ..., in_{in})$ es comúnmente llamado "*vector entrada*".

Algunas de las funciones de entrada más comúnmente utilizadas y conocidas son:

1) *Sumatoria de las entradas pesadas:* es la suma de todos los valores de entrada a la neurona, multiplicados por sus correspondientes pesos.

$$\sum_j \left(n_{ij}\, w_{ij}\right), \quad \text{con } j = 1, 2, ..., n$$

2) *Productoria de las entradas pesadas:* es el producto de todos los valores de entrada a la neurona, multiplicados por sus correspondientes pesos.

$$\prod_j \left(n_{ij}\, w_{ij}\right), \quad \text{con } j = 1, 2, ..., n$$

3) *Máximo de las entradas pesadas:* solamente toma en consideración el valor de entrada más fuerte, previamente multiplicado por su peso correspondiente.

$$\operatorname{Max}_j \left(n_{ij}\, w_{ij}\right) \quad \text{con } j = 1, 2, ..., n$$

4.3 Función de activación (*activation function*).

Una neurona biológica puede estar activa (excitada) o inactiva (no excitada); es decir, que tiene un "*estado de activación*". Las neuronas artificiales también tienen diferentes estados de activación; algunas de ellas solamente dos, al igual que las biológicas, pero otras pueden tomar cualquier valor dentro de un conjunto determinado.

La *función activación* calcula el estado de actividad de una neurona; transformando la entrada global (menos el umbral, Θ_i) en un valor (estado) de activación, cuyo rango normalmente va de (0 a 1) o de (–1 a 1). Esto es así, porque una neurona puede estar totalmente inactiva (0 o –1) o activa (1).

La función activación, es una función de la entrada global (gin_i) menos el umbral (Θ_i). Las funciones de activación más comúnmente utilizadas se detallan a continuación:

1) *Función lineal:*

$$f(x) = \begin{cases} -1 & x \le -1/a \\ a * x & -1/a < x < 1/a \\ 1 & x \ge 1/a \end{cases}$$

con $x = gin_i - \Theta_i$, y $a > 0$.

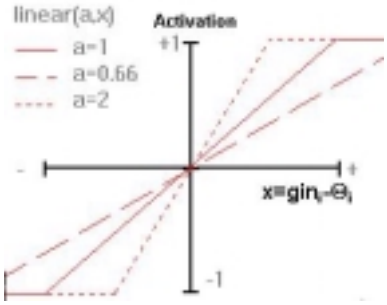

Figura 4.4: función de activación lineal.

Los valores de salida obtenidos por medio de esta función de activación serán: $a \cdot (gin_i - \Theta_i)$, cuando el argumento de $(gin_i - \Theta_i)$ esté comprendido dentro del rango (-1/a, 1/a). Por encima o por debajo de esta zona se fija la salida en 1 o –1, respectivamente. Cuando a = 1 (siendo que la misma afecta la pendiente de la gráfica), la salida es igual a la entrada.

2) *Función sigmoidea:*

$$f(x) = \frac{1}{1 + e^{-g\,x}}, \text{ con } x = gin_i - \Theta_i.$$

Los valores de salida que proporciona esta función están comprendidos dentro de un rango que va de 0 a 1. Al modificar el valor de *g* se ve afectada la pendiente de la función de activación.

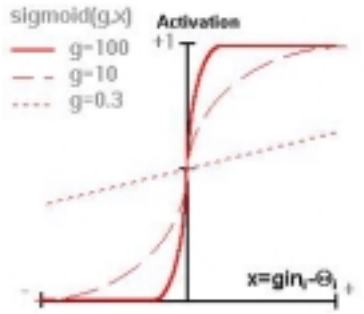

Figura 4.5: función de activación sigmoidea

3) *Función tangente hiperbólica:*

$$f(x) = \frac{e^{g\,x} - e^{-g\,x}}{e^{g\,x} + e^{-g\,x}}, \text{ con } x = gin_i - \Theta_i.$$

Los valores de salida de la función tangente hiperbólica están comprendidos dentro de un rango que va de -1 a 1. Al modificar el valor de *g* se ve afectada la pendiente de la función de activación.

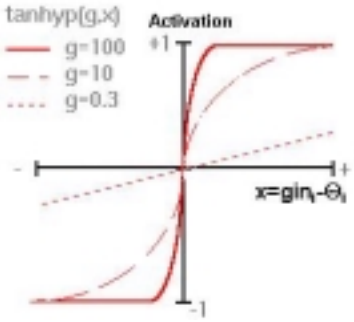

Figura 4.6: función de activación tangente hiperbólica.

Para explicar porque se utilizan estas funciones de activación se suele emplear la analogía a la aceleración de un automóvil. Cuando un auto inicia su movimiento necesita una potencia elevada para comenzar a acelerar. Pero al ir tomando velocidad, este demanda un menor incremento de dicha potencia para mantener la aceleración. Al llegar a altas velocidades, nuevamente un amplio incremento en la potencia es necesario para obtener una pequeña ganancia de velocidad. En resumen, en ambos extremos del rango de aceleración de un automóvil se demanda una mayor potencia para la aceleración que en la mitad de dicho rango.

4.4 Función de salida (*output function*).

El último componente que una neurona necesita es la *función de salida*. El valor resultante de esta función es la salida de la neurona *i* (out_i); por ende, la función de salida determina que valor se transfiere a las neuronas vinculadas. Si la función de activación está por debajo de un umbral determinado, ninguna salida se pasa a la neurona subsiguiente. Normalmente, no cualquier valor es permitido como una entrada para una neurona, por lo tanto, los valores de salida están comprendidos en el rango [0, 1] o [-1, 1]. También pueden ser binarios {0, 1} o {-1, 1}.

Dos de las funciones de salida más comunes son:

- Ninguna: este es el tipo de función más sencillo, tal que la salida es la misma que la entrada. Es también llamada *función identidad*.
- Binaria: $\begin{cases} 1 & \text{si } act_i \geq \xi_i \\ 0 & \text{de lo contrario} \end{cases}$, donde ξ_i es el umbral.

5 Aprendizaje, Validación y Codificación

5.1 Niveles o capas de una red neuronal.

La distribución de neuronas dentro de la red se realiza formando niveles o capas, con un número determinado de dichas neuronas en cada una de ellas. A partir de su situación dentro de la red, se pueden distinguir tres tipos de capas:

- *De entrada*: es la capa que recibe directamente la información proveniente de las fuentes externas de la red.

- *Ocultas*: son internas a la red y no tienen contacto directo con el entorno exterior. El número de niveles ocultos puede estar entre cero y un número elevado. Las neuronas de las capas ocultas pueden estar interconectadas de distintas maneras, lo que determina, junto con su número, las distintas topologías de redes neuronales.

- *De salidas*: transfieren información de la red hacia el exterior.

En la *Figura 4.1* se puede ver el ejemplo de la estructura de una posible red multicapa, en la que cada nodo o neurona únicamente está conectada con neuronas de un nivel superior. Notar que hay más conexiones que neuronas en sí; en este sentido, se dice que una red es totalmente conectada si todas las salidas desde un nivel llegan a todos y cada uno de los nodos del nivel siguiente.

5.2 Tipos de neuronas artificiales.

Las neuronas artificiales se pueden clasificar de acuerdo a los valores que pueden tomar. Por ahora es suficiente distinguir entre dos tipos principales:

a- Neuronas binarias.

b- Neuronas reales.

Las *neuronas binarias* solamente pueden tomar valores dentro del intervalo {0, 1} o {-1, 1}, mientras que las *neuronas reales* pueden hacerlo dentro del rango [0, 1] o [-1, 1]. Los peso normalmente no están restringidos a un cierto intervalo, aunque para aplicaciones específicas puede ser esto necesario.

5.3 Técnicas de decisión.

En general, el proceso de decisión puede ser caracterizado como se muestra en el diagrama de la *Figura 5.1*:

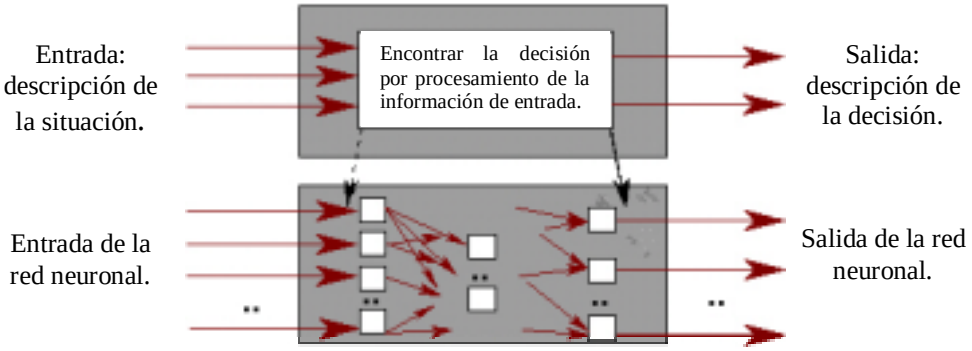

Figura 5.1: esquema del proceso de decisión.

Lo mismo ocurre cuando se utiliza una red neuronal como un sistema de sostén de decisiones. La salida de la red neuronal es directa o indirectamente la solución al problema o la decisión a tomar. Por ejemplo: si uno desea determinar si un cliente, el cual está preguntando por la solicitud de un crédito, es un buen candidato (lo que significa que es confiable, o sea, que puede devolver el crédito solicitado), se podrían tomar en consideración variables o atributos tales como “antecedentes de créditos, pasivo, garantías y ganancias”. Las variables o atributos pertenecientes a un cliente específico son las entradas al proceso de decisión. El resultado de tal proceso podría ser una decisión similar a la siguiente: “cliente bueno” o “cliente malo”.

Los atributos expuestos en la *Tabla 5.1* serán tomados en cuenta para el caso del puntaje para un crédito:

Abreviación	**Nombre**	**Valores**
A1	Historia de créditos	Mala Desconocida Buena
A2	Pasivo	Alto Bajo
A3	Garantía	Ninguna Adecuada
A4	Ganancia	1 (baja) 2 (adecuada) 3 (alta)

Tabla 5.1

A menudo las reglas que muestran la lógica (o falta de lógica) sobre las que se basan las decisiones no son tan obvias. Por lo tanto, una persona que no conoce mucho acerca del sujeto que solicita el préstamo, no puede tomar una decisión correcta. Notar que aquí los valores son más bien simbólicos que numéricos. Las redes neuronales reales necesitan entradas numéricas, pero por el momento se utilizarán entradas simbólicas para facilitar el entendimiento.

En la *Tabla 5.2* se presenta un conjunto de datos de anteriores situaciones de decisión, cada una de ellas caracterizada por diferentes valores de las variables de entrada, con su respectiva decisión (0 es un cliente bueno, 1 uno promedio y 2 uno malo).

Nº	A1	A2	A3	A4	Clase
01	malo	alto	ninguno	1	2
02	conocido	alto	ninguno	2	2
03	conocido	bajo	ninguno	2	1
04	conocido	bajo	ninguno	1	2
05	conocido	bajo	ninguno	3	0
06	conocido	bajo	adecuado	3	0
07	malo	bajo	ninguno	1	2
08	malo	bajo	adecuado	3	1
09	bueno	bajo	ninguno	3	0
10	bueno	alto	adecuado	3	0
11	bueno	alto	ninguno	1	2
12	bueno	alto	ninguno	2	1
13	bueno	alto	ninguno	3	0
14	bueno	alto	ninguno	2	2

Tabla 5.2

Dentro de esta base de datos yace el procedimiento de decisión, o en otras palabras, la *regla de decisión* que conforma las bases para el problema de decisión: “puntaje para un crédito”.

La pregunta a contestar ahora es: ¿puede encontrarse una estructura o regla de decisión en este conjunto de datos? La respuesta es *si*, las redes neuronales pueden hallar una regla de decisión a través de un conjunto de datos como el presentado en la *Tabla 5.2*.

5.4 Mecanismos de aprendizaje.

Se ha visto que los datos de entrada se procesan a través de la red neuronal con el propósito de lograr una salida. También se dijo que las redes neuronales extraen generalizaciones desde un conjunto determinado de ejemplos anteriores de tales problemas de decisión. Una red neuronal debe aprender a calcular la salida correcta para cada constelación (arreglo o vector) de entrada en el conjunto de ejemplos. Este proceso de aprendizaje se denomina: *proceso de entrenamiento o acondicionamiento*. El conjunto de datos (o conjunto de ejemplos) sobre el cual este proceso se basa es, por ende, llamado: *conjunto de datos de entrenamiento.*

Si la topología de la red y las diferentes funciones de cada neurona (entrada, activación y salida) no pueden cambiar durante el aprendizaje, mientras que los pesos sobre cada una de las conexiones si pueden hacerlo; el aprendizaje de una red neuronal significa: *adaptación de los pesos*.

En otras palabras el aprendizaje es el proceso por el cual una red neuronal modifica sus pesos en respuesta a una información de entrada. Los cambios que se producen durante el mismo se reducen a la destrucción, modificación y creación de conexiones entre las neuronas. En los sistemas biológicos existe una continua destrucción y creación de conexiones entre las neuronas. En los modelos de redes neuronales artificiales, la creación de una nueva conexión implica que el peso de la misma pasa a tener un valor distinto de cero. De la misma manera, una conexión se destruye cuando su peso pasa a ser cero.

Durante el proceso de aprendizaje, los pesos de las conexiones de la red sufren modificaciones, por lo tanto, se puede afirmar que este proceso ha terminado (la red ha aprendido) cuando los valores de los pesos permanecen estables ($dw_{ij}/dt = 0$).

Un aspecto importante respecto al aprendizaje de las redes neuronales es el conocer cómo se modifican los valores de los pesos, es decir, cuáles son los criterios que se siguen para cambiar el valor asignado a las conexiones cuando se pretende que la red aprenda una nueva información.

Hay dos métodos de aprendizaje importantes que pueden distinguirse:

a- Aprendizaje supervisado.

b- Aprendizaje no supervisado.

Otro criterio que se puede utilizar para diferenciar las reglas de aprendizaje se basa en considerar si la red puede aprender durante su funcionamiento habitual o si el aprendizaje supone la desconexión de la red, es decir, su inhabilitación hasta que el proceso termine. En el primer caso, se trataría de un aprendizaje *on line*, mientras que el segundo es lo que se conoce como *off line*.

Cuando el aprendizaje es off line, se distingue entre una *fase de aprendizaje o entrenamiento* y una *fase de operación o funcionamiento*, existiendo un conjunto de datos de entrenamiento y un conjunto de datos de test o prueba, que serán utilizados en la correspondiente fase. Además, los pesos de las conexiones permanecen fijos después que termina la etapa de entrenamiento de la red. Debido precisamente a su carácter estático, estos sistemas no presentan problemas de estabilidad en su funcionamiento.

Una generalización de la fórmula o regla para decir los cambios en los pesos es la siguiente:

$$Peso\ Nuevo = Peso\ Viejo + Cambio\ de\ Peso$$

Matemáticamente esto es:

$$w_{ij}(t+1) = w_{ij}(t) + \Delta w_{ij}(t)$$

donde t hace referencia a la etapa de aprendizaje, $w_{ij}(t+1)$ al peso nuevo y $w_{ij}(t)$ al peso viejo.

5.4.1 Aprendizaje supervisado.

El aprendizaje supervisado se caracteriza porque el proceso de aprendizaje se realiza mediante un entrenamiento controlado por un agente externo (supervisor, maestro) que determina la respuesta que debería generar la red a partir de una entrada determinada. El supervisor controla la salida de la red y en caso de que ésta no coincida con la deseada, se procederá a modificar los pesos de las conexiones, con el fin de conseguir que la salida obtenida se aproxime a la deseada.

En este tipo de aprendizaje se suelen considerar, a su vez, tres formas de llevarlo a cabo, que dan lugar a los siguientes aprendizajes supervisados:

1) Aprendizaje por corrección de error.

2) Aprendizaje por refuerzo.

3) Aprendizaje estocástico.

5.4.1.1 Aprendizaje por corrección de error.

Consiste en ajustar los pesos de las conexiones de la red en función de la diferencia entre los valores deseados y los obtenidos a la salida de la red, es decir, en función del error cometido en la salida.

Un ejemplo de este tipo de algoritmos lo constituye la *regla de aprendizaje del Perceptron*, utilizada en el entrenamiento de la red del mismo nombre que desarrolló Rosenblatt en 1958 [Rosenblatt 58]. Esta es una regla muy simple, para cada neurona en la *capa de salida* se le calcula la desviación a la salida objetivo como el error, δ. El cual luego se utiliza para cambiar los pesos sobre la conexión de la neurona precedente. El cambio de los pesos por medio de la regla de aprendizaje del Perceptron se realiza según la siguiente regla:

$$\Delta w_{ij} = \sigma * out_j * (a_{qi} - out_i);$$

donde: a_{qi} es la salida deseada/objetivo de la neurona de salida N_i, $\delta_i = (a_{qi} - out_i)$ la desviación objetivo de la neurona N_i y σ el aprendizaje.

La salida de la neurona N_j (out_j) se utiliza, porque este valor influye en la entrada global y, por ende, en la activación y luego en la salida de la neurona N_i. Esto es semejante a un "efecto en cadena". Ver *Figura 5.2*

Figura 5.2: influencia de la salida de la neurona N_j en la entrada de la neurona N_i.

Otro algoritmo muy conocido y que pertenece a esta clasificación es la *regla de aprendizaje Delta* o regla del mínimo error cuadrado (LMS Error: Least Mean Squared Error), que también utiliza la desviación a la salida objetivo, pero toma en consideración a todas las neuronas predecesoras que tiene la neurona de salida. Esto permite cuantificar el error global cometido en cualquier momento durante el proceso de entrenamiento de la red, lo cual es importante, ya que cuanto más información se tenga sobre el error cometido, más rápido se puede aprender. Luego el error calculado (δ) es igualmente repartido entre las conexiones de las neuronas predecesoras.

Por último se debe mencionar la *regla de aprendizaje de propagación hacia atrás o de backpropagation*, también conocido como regla LMS multicapa, la cual es una generalización de la regla de aprendizaje Delta. Esta es la primer regla de aprendizaje que permitió realizar cambios sobre los pesos en las conexiones de la capa oculta.

5.4.1.2 Aprendizaje por refuerzo.

Se trata de un aprendizaje supervisado, más lento que el anterior, que se basa en la idea de no disponer de un ejemplo completo del comportamiento deseado, es decir, de no indicar durante el entrenamiento exactamente la salida que se desea que proporcione la red ante una determinada entrada.

En el aprendizaje por refuerzo la función del supervisor se reduce a indicar mediante una señal de refuerzo si la salida obtenida en la red se ajusta a la deseada (éxito = +1 o fracaso = -1), y en función de ello se ajustan los pesos basándose en un

mecanismo de probabilidades. Se podría decir que en este tipo de aprendizaje la función del supervisor se asemeja más a la de un crítico (que opina sobre la respuesta de la red) que a la de un maestro (que indica a la red la respuesta concreta que debe generar), como ocurría en el caso de supervisión por corrección del error.

5.4.1.3 Aprendizaje estocástico.

Consiste básicamente en realizar cambios aleatorios en los valores de los pesos de las conexiones de la red y evaluar su efecto a partir del objetivo deseado y de distribuciones de probabilidad.

En el aprendizaje estocástico se suele hacer una analogía en términos termodinámicos, asociando a la red neuronal con un sólido físico que tiene cierto estado energético. En el caso de la red, la energía de la misma representaría el grado de estabilidad de la red, de tal forma que el estado de mínima energía correspondería a una situación en la que los pesos de las conexiones consiguen que su funcionamiento sea el que más se ajusta al objetivo deseado.

Según lo anterior, el aprendizaje consistiría en realizar un cambio aleatorio de los valores de los pesos y determinar la energía de la red (habitualmente la función energía es una función de Liapunov). Si la energía es menor después del cambio, es decir, si el comportamiento de la red se acerca al deseado, se acepta el cambio; si, por el contrario, la energía no es menor, se aceptaría el cambio en función de una determinada y preestablecida distribución de probabilidades.

5.4.2 Aprendizaje no supervisado.

Las redes con aprendizaje no supervisado (también conocido como autosupervisado) no requieren influencia externa para ajustar los pesos de las conexiones entre sus neuronas. La red no recibe ninguna información por parte del entorno que le indique si la salida generada en respuesta a una determinada entrada es o no correcta.

Estas redes deben encontrar las características, regularidades, correlaciones o categorías que se puedan establecer entre los datos que se presenten en su entrada. Existen varias posibilidades en cuanto a la interpretación de la salida de estas redes, que dependen de su estructura y del algoritmo de aprendizaje empleado.

En algunos casos, la salida representa el grado de familiaridad o similitud entre la información que se le está presentando en la entrada y las informaciones que se le han mostrado hasta entonces (en el pasado). En otro caso, podría realizar una clusterización (clustering) o establecimiento de categorías, indicando la red a la salida a qué categoría pertenece la información presentada a la entrada, siendo la propia red quien debe encontrar las categorías apropiadas a partir de las correlaciones entre las informaciones presentadas.

En cuanto a los algoritmos de aprendizaje no supervisado, en general se suelen considerar dos tipos, que dan lugar a los siguientes aprendizajes:

1) Aprendizaje hebbiano.
2) Aprendizaje competitivo y comparativo.

5.4.2.1 Aprendizaje hebbiano.

Esta regla de aprendizaje es la base de muchas otras, la cual pretende medir la familiaridad o extraer características de los datos de entrada. El fundamento es una suposición bastante simple: si dos neuronas N_i y N_j toman el mismo estado simultáneamente (ambas activas o ambas inactivas), el peso de la conexión entre ambas se incrementa.

Las entradas y salidas permitidas a la neurona son: {-1, 1} o {0, 1} (neuronas binarias). Esto puede explicarse porque la regla de aprendizaje de Hebb se originó a partir de la neurona biológica clásica, que solamente puede tener dos estados: activa o inactiva.

5.4.2.2 Aprendizaje competitivo y comparativo.

Se orienta a la clusterización o clasificación de los datos de entrada. Como característica principal del aprendizaje competitivo se puede decir que, si un patrón nuevo se determina que pertenece a una clase reconocida previamente, entonces la inclusión de este nuevo patrón a esta clase matizará la representación de la misma. Si el patrón de entrada se determinó que no pertenece a ninguna de las clases reconocidas anteriormente, entonces la estructura y los pesos de la red neuronal serán ajustados para reconocer la nueva clase.

5.5 Elección del conjunto inicial de pesos.

Antes de comenzar el proceso de entrenamiento se debe determinar un estado inicial, lo que significa: escoger un conjunto inicial de pesos para las diversas conexiones entre las neuronas de la red neuronal. Esto puede realizarse por varios criterios; por ejemplo uno de ellos es otorgar un peso aleatorio a cada conexión, encontrándose los mismos dentro de un cierto intervalo. Generalmente un intervalo del tipo [-*n*, *n*], donde *n* es un número natural positivo.

Cabe mencionar que durante el transcurso del entrenamiento los pesos no se encuentran restringidos a dicho intervalo.

5.6 Detención del proceso de aprendizaje.

Para determinar cuándo se detendrá el proceso de aprendizaje, es necesario establecer una *condición de detención*.

Normalmente el entrenamiento se detiene cuando el cálculo del error cuadrado sobre todos los ejemplos de entrenamiento ha alcanzado un mínimo o cuando para cada uno de los ejemplos dados, el error observado está por debajo de un determinado umbral. Ya que para controlar este proceso, la mayor parte de las herramientas de las redes neuronales muestran estos errores utilizando gráficos especiales; los cuales no son utilizados para el aprendizaje, si no que solamente para dar un indicio del proceso en sí mismo.

Otra condición de detención del aprendizaje puede ser cuando un cierto número de ciclos y/o pasos de entrenamiento hayan sido completamente corridos.

Luego de alcanzarse la condición de detención, los pesos no se volverán a cambiar. Entonces podemos decir que la transformación de los datos de entrada a los de salida está resuelta. Esto se puede interpretar como una *función f* oculta en el conjunto de la red neuronal. Esta función es exactamente la "instrucción" de cómo la salida será calculada a partir de una constelación (vector) de entrada.

El orden en que los ejemplos de entrenamiento se presentan a la red neuronal es otro tema importante. En general se ha observado que en la mayoría de los casos es beneficioso realizarlo en forma aleatoria.

5.7 Codificación de los datos de entrada.

Si se observa nuevamente el ejemplo del problema: puntaje para un crédito (apartado 5.4), se ve que no hay ningún valor numérico en la base de datos. Por lo tanto la pregunta es ¿cómo puede entonces una red neuronal calcular una salida? La respuesta es sencilla; los datos tienen que ser *codificados*, o sea, deben hallarse valores apropiados para representar las características simbólicas (alto, bajo, adecuado, etc.).

Se distinguen dos tipo de variables a ser codificadas:

1) Variables o atributos numéricos (frecuentemente llamadas continuas).
2) Variables o atributos simbólicos (frecuentemente llamados discretos).

Un *atributo numérico* es aquel que puede tomar cualquier valor dentro de un cierto intervalo [*a*, *b*]; donde *a* puede ser -∞ (menos infinito) y *b*, ∞ (infinito). Por ejemplo el peso puede medirse en libras; entonces cualquier valor entre [0, ∞) está permitido. Ahora si los pesos son dados por un cierto número de términos, semejantes a: alto o bajo; entonces el atributo se denomina *simbólico*. Por lo tanto, dividiendo el intervalo [a, b] de una variable numérica dentro de subintervalos, podemos confeccionar un atributo continuo pseudodiscreto.

A continuación, en los apartados 5.7.1 y 5.7.2, serán descriptos en detalle dos procesos de codificación; asumiendo que todas las entradas se transforman dentro del intervalo [0, 1] o {0, 1} (la extensión a [-1, 1] o {-1, 1} es fácil).

5.7.1 Codificación de los atributos numéricos.

Los datos son codificados dentro de un intervalo, [0.0 + buffer de baja, 1.0 – buffer de alta], por medio de una *función lineal*. Los buffer (amortiguadores) son necesarios, especialmente cuando se trabaja con series de tiempo, porque a menudo puede observarse que una variable numérica cae por debajo del valor mínimo presenciado hasta el momento, o por encima del máximo. Por medio de esta manera de codificación se conduce a un conjunto de valores por encima de 0.0 y por debajo de 1.0, cuando se utiliza un salto de 0.0 a 1.0

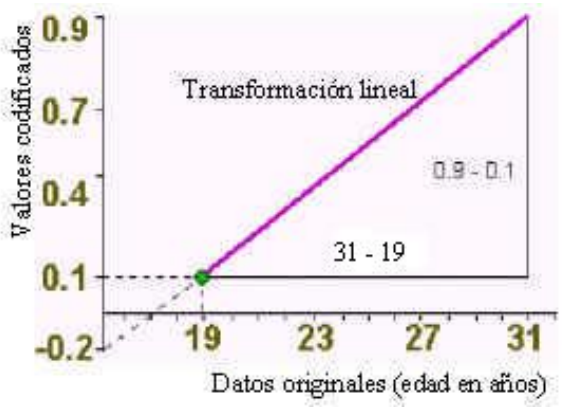

Figura 5.3: transformación de la edad al intervalo [0.1, 0.9]

Por ejemplo, para la *Figura 5.3*, se debe encontrar la ecuación que describa la función de transformación; a la cual llamaremos "*t*" y se escribe como sigue:

t: datos originales (x) $\rightarrow$ datos codificados (x_{nuevos});

$$t(x) = x_{\text{nuevos}} = a * x + b;$$

donde: a = pendiente y, b = ordenada al origen. De esta manera, para el ejemplo de la *Figura 5.3* se tiene que $a = \frac{0.9 - 0.1}{31 - 19}$ y $b = 0.1 - a * 19$.

De forma genérica:

$$a = \frac{\text{máx. en el intervalo de codificación - mín. en el intervalo de codificación}}{\text{máx. de los datos originales - mín de los datos originales}}$$

$$b = (\text{mín. en los datos codificados}) - a * (\text{mín en los datos originales})$$

5.7.2 Codificación de los atributos simbólicos.

Cada atributo simbólico que se codifica, se adjunta a una neurona en la capa de entrada. Si hay n valores simbólicos, n neuronas serán necesarias, cada una de ellas con un conjunto de entradas permitido: {0, 1} (o {-1, 1}). Por este motivo, se utilizan neuronas binarias.

Codificación de la edad		
Viejo	Medio	Joven
1	0	0
0	1	0
0	0	1

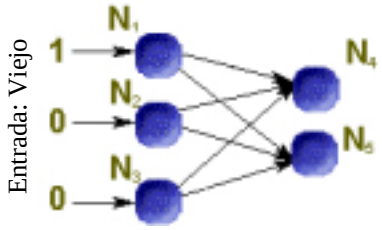

Figura 5.4: transformación de la edad en tres atributos simbólicos.

Observando la *Figura 5.4*; la edad sólo puede tomar un valor en el ejemplo dado. En consecuencia, si se tiene el valor de entrada viejo, solamente la neurona estática para viejo recibe una entrada de 1 (en el ejemplo: N_1), mientras que todas las demás tendrán una entrada igual a 0.

Por supuesto, también es posible codificar atributos simbólicos utilizando sólo una neurona "real" (recordar que los valores permitidos para este tipo de neuronas están comprendidos en el rango [0, 1] o [1, 1]). Para el ejemplo de la edad, viejo puede codificarse como 0.333, edad media como 0.666 y joven como 1; siendo el intervalo permitido de [0, 1]. Tal procedimiento solamente tiene sentido si hay un orden en los valores que las variables de entrada (o atributos) pueden tomar.

La mayor desventaja de una codificación binaria es que puede conducir a una gran capa de entrada.

5.7.3 Resumen de los procedimientos de codificación.

Nombre	Valores	Procedimiento de codificación	# neuronas
Variables numéricas	Numéricos	Función de transformación lineal	1
Variables simbólicas ▪ sin orden	*n* valores simbólicos	Cada valor simbólico se corresponde con una neurona de entrada binaria.	*n*
▪ con orden		Cada valor simbólico se codifica como un segmento del intervalo de codificación.	1
Variables pseudodiscretas	Numéricos, pero dividido dentro de *T* subintervalos	Cada subintervalo corresponde a una neurona binaria.	*T*

5.8 Validación de la red neuronal.

Después del proceso de entrenamiento los pesos de las conexiones en la red neuronal quedan fijos. Como paso siguiente se debe comprobar si la red neuronal puede resolver nuevos problemas, del tipo general, para los que ha sido entrenada. Por lo tanto, con el propósito de validar la red neuronal se requiere de otro conjunto de datos, denominado *conjunto de validación o testeo*.

Cada ejemplo del conjunto de evaluación contiene los valores de las variables de entrada, con su correspondiente solución tomada; pero ahora esta solución no se le es otorgada a la red neuronal. Luego se compara la solución calculada para cada ejemplo de validación con la solución conocida.

El nuevo ejemplo utilizado para la validación se identifica como E_u y su correspondiente salida correcta como A_u (u indica incógnita, en inglés). Ahora el problema es que hay que decidir cuando la salida de la red neuronal ha de considerarse como correcta.

5.9 Cuestiones a resolver al trabajar con una red neuronal.

Muchos problemas aparecen cuando se trabaja con redes neuronales. Primeramente se debe analizar el dominio del problema y decidir a que clase pertenece. Luego debe decidirse si una red neuronal es adecuada para resolver dicho problema. Esto es lo que se llama: *etapa preliminar*. Concluida esta etapa, las siguientes preguntas han de responderse:

a- Origen de los datos.

- ¿Qué datos son de importancia para la situación del problema definido?

- ¿Qué variables son relevantes?
- ¿De dónde pueden obtenerse los datos?

b- Preparación y codificación de los datos.

- ¿Cómo preparar y codificar los datos?

c- Topología de la red (dependiendo parcialmente del ítem b-).

- ¿Qué tipo de red debe escogerse?
- ¿Cuántas capas ocultas y con cuántas neuronas son necesarias?
- ¿Cuántas neuronas en la capa de salida (según la codificación escogida)?
- ¿Qué tipos de neuronas deben escogerse?
- ¿Qué regla de aprendizaje escoger?

d- Decisiones concernientes al proceso de aprendizaje.

- ¿Cuántos ciclos de aprendizaje?
- ¿Qué inicialización para los pesos?

6 Principales Topologías

6.1 Topología de las redes neuronales.

La topología o arquitectura de una red neuronal consiste en la organización y disposición de las neuronas en la misma, formando capas o agrupaciones de neuronas más o menos alejadas de la entrada y salida de dicha red. En este sentido, los parámetros fundamentales de la red son: el número de capas, el número de neuronas por capa, el grado de conectividad y el tipo de conexiones entre neuronas.

6.2 Redes monocapa.

En las redes monocapa, se establecen conexiones entre las neuronas que pertenecen a la única capa que constituye la red. Las redes monocapas se utilizan generalmente en tareas relacionadas con lo que se conoce como autoasociación (regenerar información de entrada que se presenta a la red de forma incompleta o distorsionada).

6.3 Redes multicapa.

Las redes multicapas son aquellas que disponen de un conjunto de neuronas agrupadas en varios (2, 3, etc.) niveles o capas. En estos casos, una forma para distinguir la capa a la que pertenece una neurona, consistiría en fijarse en el origen de las señales que recibe a la entrada y el destino de la señal de salida. Normalmente, todas las neuronas de una capa reciben señales de entrada desde otra capa anterior (la cual está más cerca a la entrada de la red), y envían señales de salida a una capa posterior (que está más cerca a la salida de la red). A estas conexiones se las denomina *conexiones hacia adelante o feedforward*.

Sin embargo, en un gran número de estas redes también existe la posibilidad de conectar la salida de las neuronas de capas posteriores a la entrada de capas anteriores; a estas conexiones se las denomina *conexiones hacia atrás o feedback.*

Estas dos posibilidades permiten distinguir entre dos tipos de redes con múltiples capas: las redes con conexiones hacia adelante o *redes feedforward*, y las redes que disponen de conexiones tanto hacia adelante como hacia atrás o *redes feedforward/feedback*.

6.4 Conexión entre neuronas.

La conectividad entre los nodos de una red neuronal está relacionada con la forma en que las salidas de las neuronas están canalizadas para convertirse en entradas de otras neuronas. La señal de salida de un nodo puede ser una entrada de otro elemento de proceso, o incluso ser una entrada de sí mismo (*conexión autorrecurrente*).

Cuando ninguna salida de las neuronas es entrada de neuronas del mismo nivel o de niveles precedentes, la red se describe como de *conexión hacia delante* (ver *Figura*

4.1). Cuando las salidas pueden ser conectadas como entradas de neuronas de niveles previos o del mismo nivel, incluyéndose ellas mismas, la red es de *conexión hacia atrás*.

Las redes de propagación hacia atrás que tienen lazos cerrados son llamadas: *sistemas recurrentes*.

6.5 Redes de propagación hacia atrás (*backpropagation*).

El nombre de backpropagation resulta de la forma en que el error es propagado hacia atrás a través de la red neuronal, en otras palabras el error se propaga hacia atrás desde la capa de salida. Esto permite que los pesos sobre las conexiones de las neuronas ubicadas en las capas ocultas cambien durante el entrenamiento.

El cambio de los pesos en las conexiones de las neuronas además de influir sobre la entrada global, influye en la activación y por consiguiente en la salida de una neurona. Por lo tanto, es de gran utilidad considerar las variaciones de la función activación al modificarse el valor de los pesos. Esto se llama *sensibilidad* de la función activación, de acuerdo al cambio en los pesos.

6.5.1 Ejemplo.

Una temperatura de 20ºC provoca que el tiempo de operación de una máquina sea de 90 segundos y un incremento de dicha temperatura hasta los 30ºC causa un tiempo de operación de 100 segundos. ¿Cómo influyó el incremento de la temperatura en el tiempo de trabajo de la máquina? Por supuesto, hizo más lenta la operación. Pero, ¿por cuánto?

El cociente, $\frac{\text{variación de tiempo}}{\text{variación de temperatura}}$, muestra la reacción o sensibilidad del tiempo de trabajo conforme a los cambios suscitados en la temperatura. En consecuencia, para nuestro ejemplo tenemos que:

$$\frac{100-90}{30-20}\left[\text{segundos}/^{o}\text{C}\right]=\frac{\Delta\text{tiempo}}{\Delta\text{temperatura}}=\frac{10}{10}\left[\text{segundos}/^{o}\text{C}\right]=1\left[\text{segundos}/^{o}\text{C}\right]$$

Esto significa que se produce un incremento de tiempo de aproximadamente 1 segundo, cuando la temperatura se eleva 1ºC.

Ahora si se supone que en lugar del tiempo se tiene la activación de una neurona y en lugar de la temperatura, la entrada global. Dado que la mayoría de las funciones de activación son no lineales (por ejemplo la función sigmoidea o la tangente hiperbólica), se tiene que calcular la derivada de la función con respecto al peso; por consiguiente, la entrada global cambia. Esta derivada se utiliza para cambiar los pesos durante el proceso de aprendizaje.

Para cada una de las neuronas en la capa de salida, la desviación objetivo (la cual es: la salida objetivo menos la salida real) es multiplicada por la derivada de la función activación. Utilizando la derivada se logra una “sintonización fina” de los pesos cuando la salida real esta cerca de la salida deseada. Al mirar la *Figura 6.1*, la misma muestra una constelación donde la salida real para una neurona es 0.95 y la deseada es de 1.0

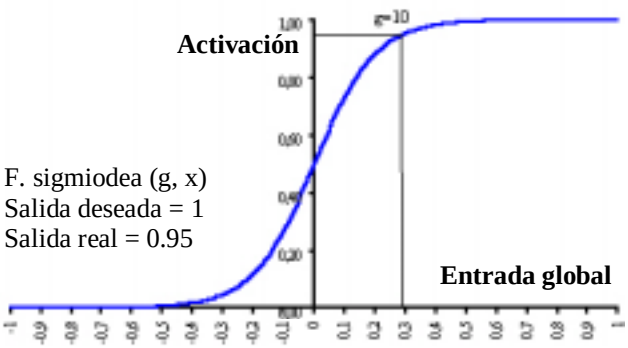

Figura 6.1: desviación a la salida objetivo.

Dado que la derivada de la función activación es relativamente baja en esta región (cuando la función activación está próxima a 1), el producto "*derivada por desviación objetivo*", igual al error, no se torna muy grande. Esto es lo que se llama *sintonía fina de los pesos*. De esta manera, la diferencia en el cómputo del error para las neuronas de salida al utilizar el algoritmo de backpropagation, en lugar de las reglas simples de aprendizaje del Perceptron, es justamente el factor *derivada de la función activación.*

6.6 Estructura de la Red Hopfield.

La Red Hopfield es recurrente y completamente interconectada. Funciona como una memoria asociativa no lineal, que puede almacenar internamente patrones presentados de forma incompleta o con ruido. De esta forma puede ser usada como una herramienta de optimización; también se han utilizado en aplicaciones de segmentación y restauración de imágenes y optimización combinatoria.

La Red Hopfield consta de un número de neuronas simétrica e íntegramente conectadas, como ya se mencionó anteriormente. Esto significa que si existe una conexión desde la neurona N_i a la neurona N_j, también existe la conexión desde N_j a N_i; ambas exhibendo el mismo peso ($w_{ij} = w_{ji}$). Vale aclarar que la conexión de una neurona con sí misma no está permitida.

El conjunto permitido de valores de entrada y salida es {0, 1} (o en alguna oportunidad {-1, 1}); o sea, es un conjunto binario. De esta manera todas las neuronas en una Red Hopfield son binarias, tomando solamente uno de los dos estados posibles: activo (1) o inactivo (-1 o 0).

Las Redes Hopfield se emplean para reconocer patrones. Después que el aprendizaje haya llegado a su fin, la red neuronal debe ser capaz de dar una salida correcta para cada patrón de entrada dado, aun cuando este sea ruidoso.

La clave del aprendizaje Hopfield es que si un patrón que tiene que ser aprendido se conoce, los pesos sobre cada conexión de la red neuronal pueden ser calculados. En esta circunstancia, solamente el estado de las neuronas cambia durante el proceso de aprendizaje. Este cálculo garantiza que cada patrón aprendido corresponda a un mínimo de la función energía.

Es importante entender que para este tipo de redes la definición de aprendizaje es diferente al dado anteriormente, donde aprendizaje significaba simplemente la adaptación de los pesos. En una Red Hopfield los pesos se pueden calcular y se

mantienen fijos durante el aprendizaje de los patrones. Solamente cambia el estado de las neuronas.

Para calcular el peso de una conexión cualquiera, w_{ij} (y por simetría para la conexión w_{ji}), en una Red Hopfield se utiliza la siguiente ecuación:

$$w_{ij} = \sum_{q=1}^{Q} (2 * e_{qi} - 1) * (2 * e_{qj} - 1), \quad i <> j$$

siendo Q el número de patrones y e_{qi} la entrada a la neurona N_i.

Generalmente es aconsejable trabajar con esta ecuación cuando los patrones que se han de aprender no son muy semejantes unos a otros, y si el número de ceros y unos son similares para todos los patrones. Con respecto al número de ceros y unos, el *umbral* de cada neurona puede utilizarse para regular esto, distinguiéndose así dos casos posibles:

a- Si hay más 0s que 1s el umbral tiene que disminuirse, porque que las neuronas tienen una probabilidad más alta para hacerse inactivas que para hacerse activas.

b- Si hay mas 1s que 0s el umbral tiene que incrementarse, porque las neuronas tienen una probabilidad más alta para hacerse activas que para hacerse inactivas.

6.7 Simulated Annealing aplicada a una Red Hopfield.

En muchos problemas, la tarea no es justamente encontrar cualquier mínimo local, sino la de encontrar el óptimo global. Lo que significa que para una entrada determinada se debe encontrar una salida que resulte en un mínimo de la función energía. Utilizando una Red Hopfield, se encuentra que un mínimo yace cerca del vector de entrada dado, porque la energía decrece paso a paso. El cual puede ser un mínimo local.

En una Red Hopfield todos los mínimos locales son un estado estable. Un problema similar se origina en termodinámica durante el proceso de cristalización. Durante un enfriamiento lento, el cristal crece con una estructura casi perfecta, ya que cada átomo tiene bastante tiempo para saltar a otra posición dentro de la cuadrícula, de tal forma que la energía total del cristal decrezca. Para realizar dicho salto se necesita energía, es decir, que si el cristal tiene la energía suficiente (si su temperatura es aun bastante alta), todos los átomos disponen de una chance para cambiar su posición. Pero para permitir que esto ocurra la energía de un átomo tiene que incrementarse por un corto tiempo, de lo contrario el átomo descansaría en su vieja posición.

Tener una chance se puede interpretar como *"hay una probabilidad"*. Esta probabilidad depende de la activación que un átomo muestra a una determinada temperatura y tiempo del sistema.

Utilizando esta técnica donde el cristal comienza a una temperatura elevada y que luego decrece paso a paso, se les da a los átomos una posibilidad de cambiar sus estados independientemente de la activación, por medio de un incremento en la energía de los mismos de un paso a otro. Cuando la temperatura se reduce, la cuadrícula vibra menos, y el sistema (la cuadrícula) alcanza un estado estable; haciéndose gradualmente más dificultoso para un átomo encontrar la energía para saltar a otra posición. Esta es la idea de Simulated Annealing, que luego se aplica a la Red Hopfield cuando se intenta encontrar un óptimo global.

A grandes rasgos se describe que una Simulated Annealing trabaja de esta manera:

a- Escoger cualquier neurona.

b- Calcular $d = gin_i - \Theta_i$.

c- Calcular $P_i = \frac{1}{1+e^{-d/T}}$ (probabilidad)

d- Generar un número aleatorio r, con $1 \geq r \geq 0$

e- Si ($P_i \geq r$)

el conjunto out_i a 1

de otra manera

el conjunto out_i a 0

f- Disminuye T. Volver al paso a-.

El algoritmo se detiene cuando se alcanza algún criterio de detención; por ejemplo si la temperatura llega a su límite inferior o si el número de ciclos alcanza su límite superior.

La premisa fundamental de este método es que el problema de optimización puede formularse como una función energética. Por lo tanto hallar el óptimo global implica encontrar el mínimo de dicha función energética. La misma tiene un aspecto genérico de esta forma:

$$E = -0.5 * \sum_i \sum_j (w_{ij} * out_i * out_j) + \sum_i (\Theta_i * out_i)$$

Se han logrado exitosas aplicaciones de Simulated Annealing, principalmente concernientes a los problemas de optimización combinatoria, semejantes al problema del *viajante*. El cual no puede resolverse fácilmente por métodos estadísticos o analíticos. Por ejemplo, imagine que un viajante tiene que visitar 19 ciudades. ¿Qué ruta suministra el camino más corto entre todas las ciudades, o sea, en qué orden deberán visitarse todas las ciudades?

Por último una diferenciación muy importante es que en una Simulated Annealing la energía puede disminuir en un paso y crecer en el otro, pero en una Red Hopfield la energía solamente puede disminuir paso a paso.

6.8 Asociaciones entre la información de entrada y salida.

Ya se sabe que las redes neuronales son sistemas que almacenan cierta información *aprendida*. Esta información se registra de forma distribuida en los pesos asociados a las conexiones entre neuronas. Por tanto, puede imaginarse una red como cierto tipo de memoria que almacena datos de forma estable, datos que se grabarán en dicha memoria como consecuencia del aprendizaje de la red y que podrán ser leídos a la salida como respuesta a cierta información de entrada, comportándose entonces la red como lo que habitualmente se conoce por memoria asociativa: cuando se aplica un estímulo (dato de entrada) la red responde con una salida asociada a dicha información de entrada.

Existen dos formas primarias de realizar esta asociación entre entradas/salidas que se corresponden con la naturaleza de la información almacenada en la red. Una primera sería la denominada *heteroasociación*, que se refiere al caso en el que la red aprende parejas de datos [(A_1,B_1), (A_2, B_2),... (A_N,B_N)], de tal forma que cuando se presente cierta información de entrada A_i, deberá responder generando la correspondiente salida asociada B_i. La segunda se conoce como *autoasociación,* donde la red *aprende* ciertas informaciones A_1, A_2, ..., A_N; de tal forma que cuando se le presenta una información de entrada realizará una autocorrelación, respondiendo con uno de los datos almacenados, el más parecido al de entrada.

Estos dos mecanismos de asociación dan lugar a dos tipos de redes neuronales: las redes heteroasociativas y las autoasociativas. Una *red heteroasociativa* podría considerarse como aquella que computa cierta función, que en la mayoría de los casos no podría expresarse analíticamente, entre un conjunto de entradas y un conjunto de salidas, correspondiendo a cada posible entrada una determinada salida. Por otra parte, una *red autoasociativa* es una red cuya principal misión es reconstruir una determinada información de entrada que se presente incompleta o distorsionada (le asocia el dato almacenado más parecido).

En realidad estos dos tipos de modelos de redes no son diferentes en principio, porque una red heteroasiciativa puede siempre ser reducida a una asociativa mediante la concatenación de una información de entrada y su salida (respuesta) asociada, para obtener la información de entrada de la red autoasociativa equivalente. También puede conseguirse que una red autoasociativa se comporte como heteroasociativa, simplemente presentando, como entrada parcial de la autoasociativa, la información de entrada para la heteroasociativa y haciendo que la red complete la información para producir lo que sería la salida de la red heteroasociativa equivalente.

6.8.1 Redes heteroasociativas.

Las redes heteroasociativas, al asociar informaciones de entrada con diferentes informaciones de salida, precisan al menos de dos capas, una para captar y retener la información de entrada y otra para mantener la salida con la información asociada. Si esto no fuese así, se perdería la información inicial al obtenerse el dato asociado {3}, lo cual no debe ocurrir, ya que en el proceso de obtención de la salida se puede necesitar acceder varias veces a esta información que, por tanto, deberá permanecer en la capa de entrada.

En cuanto a su conectividad, pueden ser del tipo con conexión hacia adelante (*o feedforward*) o con conexión hacia atrás (*feddforward/feedback*), o bien con conexiones laterales.

6.8.2 Redes autoasociativas.

Una red autoasociativa asocia una información de entrada con el ejemplar más parecido de los almacenados *conocidos* por la red. Estos tipos de redes pueden implementarse con una sola capa de neuronas. Esta capa comenzará reteniendo la información inicial a la entrada, y terminará representando la información autoasociada. Si se quiere mantener la información de entrada y salida, se deberían añadir capas adicionales, sin embargo, la funcionalidad de la red puede conseguirse en una sola capa.

En cuanto a su conectividad, existen de conexiones laterales y, en algunos casos, conexiones autorrecurrentes.

7 Aplicaciones

7.1 Aplicaciones de las redes neuronales.

Las redes neuronales pueden utilizarse en un gran número y variedad de aplicaciones, tanto comerciales como militares.

Se pueden desarrollar redes neuronales en un periodo de tiempo razonable, con la capacidad de realizar tareas concretas mejor que otras tecnologías. Cuando se implementan mediante hardware (redes neuronales en chips VLSI), presentan una alta tolerancia a fallos del sistema y proporcionan un alto grado de paralelismo en el procesamiento de datos. Esto posibilita la inserción de redes neuronales de bajo coste en sistemas existentes y recientemente desarrollados.

Hay muchos tipos diferentes de redes neuronales; cada uno de los cuales tiene una aplicación particular más apropiada. Algunas aplicaciones comerciales son:

- Biología:
  - Aprender más acerca del cerebro y otros sistemas.
  - Obtención de modelos de la retina.
- Empresa:
  - Evaluación de probabilidad de formaciones geológicas y petrolíferas.
  - Identificación de candidatos para posiciones específicas.
  - Explotación de bases de datos.
  - Optimización de plazas y horarios en líneas de vuelo.
  - Optimización del flujo del tránsito controlando convenientemente la temporización de los semáforos.
  - Reconocimiento de caracteres escritos.
  - Modelado de sistemas para automatización y control.
- Medio ambiente:
  - Analizar tendencias y patrones.
  - Previsión del tiempo.
- Finanzas:
  - Previsión de la evolución de los precios.
  - Valoración del riesgo de los créditos.
  - Identificación de falsificaciones.
  - Interpretación de firmas.

- Manufacturación:
  - Robots automatizados y sistemas de control (visión artificial y sensores de presión, temperatura, gas, etc.).
  - Control de producción en líneas de procesos.
  - Inspección de la calidad.
- Medicina:
  - Analizadores del habla para ayudar en la audición de sordos profundos.
  - Diagnóstico y tratamiento a partir de síntomas y/o de datos analíticos (electrocardiograma, encefalogramas, análisis sanguíneo, etc.).
  - Monitorización en cirugías.
  - Predicción de reacciones adversas en los medicamentos.
  - Entendimiento de la causa de los ataques cardíacos.
- Militares:
  - Clasificación de las señales de radar.
  - Creación de armas inteligentes.
  - Optimización del uso de recursos escasos.
  - Reconocimiento y seguimiento en el tiro al blanco.

La mayoría de estas aplicaciones consisten en realizar un reconocimiento de patrones, como ser: buscar un patrón en una serie de ejemplos, clasificar patrones, completar una señal a partir de valores parciales o reconstruir el patrón correcto partiendo de uno distorsionado. Sin embargo, está creciendo el uso de redes neuronales en distintos tipos de sistemas de control.

Desde el punto de vista de los casos de aplicación, la ventaja de las redes neuronales reside en el procesado paralelo, adaptativo y no lineal.

El dominio de aplicación de las redes neuronales también se lo puede clasificar de la siguiente forma: asociación y clasificación, regeneración de patrones, regresión y generalización, y optimización.

7.1.1 Asociación y clasificación.

En esta aplicación, los patrones de entrada estáticos o señales temporales deben ser clasificadas o reconocidas. Idealmente, un clasificador debería ser entrenado para que cuando se le presente una versión distorsionada ligeramente del patrón, pueda ser reconocida correctamente sin problemas. De la misma forma, la red debería presentar cierta inmunidad contra el ruido, esto es, debería ser capaz de recuperar una señal "limpia" de ambientes o canales ruidosos. Esto es fundamental en las aplicaciones holográficas, asociativas o regenerativas.

- Asociación: de especial interés son las dos clases de asociación: autoasociación y heteroasociación. Como ya se mencionó en el apartado 6.8, el problema de la autoasociación es recuperar un patrón enteramente, dada una información parcial del patrón deseado. La heteroasociación es recuperar un conjunto de patrones *B*, dado un patrón de ese conjunto. Los pesos en las redes asociativas son a

menudo predeterminados basados en la regla de Hebb. Normalmente, la autocorrelación del conjunto de patrones almacenado determina los pesos en las redes autoasociativas. Por otro lado, la correlación cruzada de muchas parejas de patrones se usa para determinar los pesos de la red de heteroasociación.

- Clasificación no Supervisada: para esta aplicación, los pesos sinápticos de la red son entrenados por la regla de aprendizaje no supervisado, esto es, la red adapta los pesos y verifica el resultado basándose únicamente en los patrones de entrada.

- Clasificación Supervisada: esta clasificación adopta algunas formas del criterio de interpolación o aproximación. En muchas aplicaciones de clasificación, por ejemplo, reconocimiento de voz, los datos de entrenamiento consisten de pares de patrones de entrada y salida. En este caso, es conveniente adoptar las redes Supervisadas, como las bien conocidas y estudiadas redes de retropropagación. Este tipo de redes son apropiadas para las aplicaciones que tienen una gran cantidad de clases con límites de separación complejos.

7.1.2 Regeneración de patrones.

En muchos problemas de clasificación, una cuestión a solucionar es la recuperación de información, esto es, recuperar el patrón original dada solamente una información parcial. Hay dos clases de problemas: temporales y estáticos. El uso apropiado de la información contextual es la llave para tener éxito en el reconocimiento.

7.1.3 Regeneración y generalización.

El objetivo de la generalización es dar una respuesta correcta a la salida para un estímulo de entrada que no ha sido entrenado con anterioridad. El sistema debe inducir la característica saliente del estímulo a la entrada y detectar la regularidad. Tal habilidad para el descubrimiento de esa regularidad es crítica en muchas aplicaciones. Esto hace que el sistema funcione eficazmente en todo el espacio, incluso cuando ha sido entrenado por un conjunto limitado de ejemplos.

7.1.4 Optimización.

Las Redes Neuronales son herramientas interesantes para la optimización de aplicaciones, que normalmente implican la búsqueda del mínimo absoluto de una función de energía. Para algunas aplicaciones, la función de energía es fácilmente deducible; pero en otras, sin embargo, se obtiene de ciertos criterios de coste y limitaciones especiales.

7.2 Casos concretos de aplicación.

A continuación se detallan los siguientes casos concretos de aplicación de redes neuronales:

- Planificación del staff de empleados.
- Planificación de la demanda de materiales.
- Puntuación para la solicitud de un crédito.

7.2.1 Planificación del staff (cuerpo) de empleados.

Hoy más que nunca, las empresas están sujetas a la presión de los elevados costos. Esto puede verse en diferentes sectores corporativos, tales como la planificación del staff de empleados. Desde el punto de vista de las empresas, un empleado que falla al ejecutar la mayor parte de las tareas asignadas, evidencia una baja productividad. Por el otro lado, esta situación es frustrante para el empleado. Ambos efectos causan costos, los cuales podrían evitarse realizando antes una prueba de aptitud. Estos problemas no solamente son originados por los empleados nuevos, sino también por aquellos que son reubicados dentro de la misma empresa.

En este proyecto de investigación se examinó hasta donde la predicción de aptitudes puede llevarse a cabo por una red neuronal, cuya topología suministre una tarea satisfactoria y así lograr una predicción más exitosa.

Base de datos y codificación:

La base de datos inicial contenía información resultante de una investigación que realizaron por medio de un cuestionario. Las respuestas obtenidas a través del mismo las utilizaron para acumular información acerca de las cualidades específicas y habilidades técnicas de cada individuo del personal indagado. Para cada pregunta, les fue posible categorizar la respuesta en un intervalo que va de 1 a 5; constituyendo así la entrada que presentaron a la red neuronal. Al entrevistado, posteriormente, lo examinaron en el orden de obtener una cifra representativa de sus aptitudes. De esta manera el conjunto de datos de entrenamiento quedó formado de la siguiente forma:

- Respuesta obtenidas a través del cuestionario = datos de entrada.
- Cifra representativa de la aptitud de la persona encuestada = salida deseada.

El primer problema que se les presentó fue cómo codificar los datos obtenidos, decidiendo transformarlos dentro del intervalo [0.1, 1.0].

Cómo codificar la salida objetivo fue la próxima pregunta que consideraron. Normalmente la compañía sólo quiere conocer si una persona ejecutará bien o mal la tarea determinada, o si su desempeño será muy bueno, bueno, promedio, malo o muy malo. Consecuentemente, (*a*) asignaron la salida dada dentro de varias clases y (*b*) transformaron las cifras representativas dentro del intervalo [0, 1], utilizando en parte una función lineal.

Algoritmo de aprendizaje:

Ensayaron diferentes algoritmos de aprendizaje, de los cuales dos fueron escogidos como los más apropiados: Propagación Rápida (Quickpropagation) y Propagación Elástica (Resilient Propagation).

- Quickpropagation: es una modificación del algoritmo estándar de backpropagation. A diferencia de este, la adaptación de los pesos no es solamente influenciada por la sensibilidad actual, sino también por la inclusión del error previo calculado.

- Resilient Propagation: es otra modificación del algoritmo estándar de backpropagation. En oposición a este, la adaptación de los pesos es influenciada por el signo de la sensibilidad actual y antecesora, y no por su cantidad.

Topología de la red:

Evaluaron diferentes topologías de redes, las cuales no serán detalladas. La pregunta fue: (*a*) ¿cuántas capas ocultas son necesarias?, (*b*) ¿cuántas neuronas en cada una de ellas? La primera prueba que hicieron mostró que para este propósito la red debía contener 2 capas ocultas, con la primera formada por tantas neuronas como la capa de entrada y la segunda por un número menor que la primera (exactamente la mitad como mucho).

Resultados obtenidos a partir de los ensayos:

El primer resultado que consiguieron al intentar predecir la cifra representativa correcta fue relativamente mala. Asumieron que esto fue causado por el hecho de que el número de neuronas de entrada en proporción al número de ejemplos dados en el conjunto de datos de entrenamiento fue elevado. La pequeña base de datos, conforme con la gran capa de entrada, fue suficiente para realizar una tosca predicción, pero no para dar la correcta cifra representativa.

Lo mencionado en el párrafo anterior hizo que enfocaran toda la atención en reducir el número de neuronas de entradas en forma apropiada. También examinaron la red con la cual se logró el mejor resultado, en función de conseguir indicadores de las entradas que demostraran ser importantes y cúales no. Entonces, reduciendo el número de neuronas de entrada y formando nuevas redes, consiguieron un resultado bastante bueno para la predicción de las clases y aún para la predicción de la cifra representativa correcta.

En otra serie de test, examinaron los resultados que podrían favorecer a un mejoramiento por agrupación de las neuronas de entrada para las preguntas interdependientes. Cada grupo, que representaba una habilidad especial, fue conectado exactamente a una neurona en la primer capa oculta. La razón para esto fue que haciendo ciertas conexiones se reduce beneficiosamente el espacio de búsqueda, si y solo si, las conexiones representan la estructura correcta, pero puede reducir el espacio de búsqueda inapropiadamente por prohibición de otras conexiones.

7.2.2 Planificación de la demanda de materiales.

La tarea de planificar la demanda de materiales es justamente predecir en forma segura la necesidad de los mismos, o más exactamente, de los factores de consumo. Esto involucra tener una correcta información de los volúmenes aproximados de producción, así como de los factores de tiempo.

Principalmente para resolver esta tarea pueden distinguirse los siguientes procedimientos:

- Programación orientada (program-oriented).
- Consumo orientado (consumption-oriented).

En el método de programación orientada la predicción de la cantidad demandada se basa sobre los resultados del planeamiento de producción. Mientras que el método de consumo orientado toma en cuenta el consumo observado en períodos anteriores y sobre la base de dicha información intenta predecir las futuras demandas.

Como primera aproximación utilizaron una red neuronal por predicción del consumo orientado; centralizando la investigación en la determinación de las

habilidades de las redes para producir salidas correctas cuando analizan un dato real. El proyecto lo realizaron conjuntamente con la compañía que lo solicitó, la cual se dedica a la fabricación de bicicletas y por ende, necesita de tubos para la construcción de las mismas; cuyo consumo debían predecir.

Base de datos y codificación:

En la *Figura 7.1* se muestran las series de tiempo que utilizaron para la planificación de la demanda de materiales.

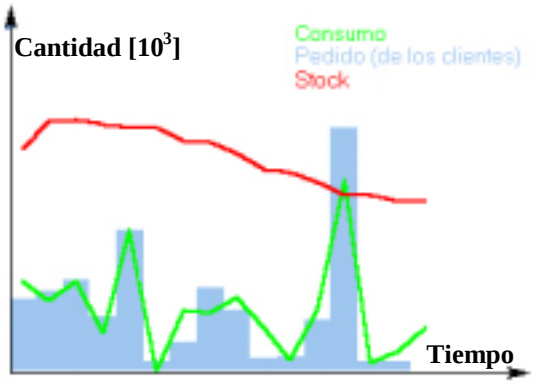

Figura 7.1: series de tiempo empleadas en la planificación de la demanda de materiales.

Cada serie de tiempo incluye datos desde 1992 a 1994 (semanalmente). Las series, pedido y stock, fueron codificados dentro del intervalo [0.0, 0.9], para lo cual utilizaron una función de transformación lineal. Mientras que al consumo lo codificaron en dos pasos, porque esta serie de tiempo varía, en partes, de manera muy brusca. Para el primer paso emplearon la fórmula:

$$z_{new} = \frac{x_{old} - m_x}{std(x)}$$

siendo m_x el promedio de las series de tiempo y $std(x)$ la desviación estándar de las series de tiempo. Y para el segundo aplicaron una transformación lineal a la nueva serie de tiempos z.

Topología de la red:

Sobre el ensayo de diferentes redes neuronales observaron que, sin importar la predicción que se haga (mensualmente, semanalmente), más de dos capas ocultas desfavorecía el resultado obtenido.

Primero experimentaron solamente con la serie de tiempo del consumo, y con una red neuronal constituida por nueve neuronas de entrada y una neurona más para la estación; empleando 80 patrones para el entrenamientos y 53 para la validación. La exactitud que lograron luego de varias corridas de la red neuronal, empleando diferentes topologías, se muestran en la *Tabla 7.1* (error permitido: 20%, ciclos de aprendizaje: 10000, algoritmos de aprendizaje: Quickpropagation o Resilient Propagation).

Topología	Exactitud
10-10-36-1	30.0%
10-20-36-1	10.0%
10-27-10-1	10.0%

Tabla 7.1

Como se puede observar, dando solo el consumo como una entrada no es suficiente para predecir la demanda.

Luego, utilizaron todas las series de tiempo descriptas en la *Figura 7.1* como entradas a la red neuronal y además, asignaron tres neuronas de entrada para cada serie de tiempo, lo que resultó en una capa de entrada de nueve neuronas; logrando para las diferentes topologías una exactitud como la que exhibe en la *Tabla 5.1*:

Topología	Error permitido	Exactitud
9-27-1	10 % 20 %	86.36 % 89.39 %
9-9-18-1	10 % 20 %	87.88 % 90.15 %
9-18-9-1	10 % 20 %	66.67 % 70.45 %

Tabla 7.2

Los resultados citados no parecen ser suficientemente buenos para la aplicación de una red neuronal real en el campo del planeamiento de la demanda de materiales; no obstante, está claro que las técnicas de las redes neuronales son prometedoras para la tarea. Actualmente se están ensayando otros escenarios diferentes para mejorar las predicciones así obtenidas.

7.2.3 Puntuación para la solicitud de un crédito.

La puntuación para un crédito representa una tarea de gran riesgo para las instituciones crediticias. Estas instituciones tienen un fuerte interés en evitar tales situaciones, rechazando los candidatos que parecen ser un riesgo malo. Ya que un candidato rechazado, que de hecho era un buen riesgo crediticio, no impone costos reales; mientras que, un candidato que es incapaz de restituir el pago del crédito extendido puede causar pérdidas sustanciales (recordar el caso Schneider en Alemania).

Diferentes métodos son aplicados en este campo. Muy a menudo la decisión tomada, aprobación o rechazo, se basa sobre los siguientes factores: carácter, capacidad y capital (créditos triple C); siendo esta solamente humana y naturalmente subjetiva.

Otro método es la utilización de un sistema de puntuación numérico. El mismo utiliza rasgos comunes que resultan ser importantes para la evaluación de los candidatos para un crédito. La importancia específica de cada simple característica está expresada por pesos. Cada candidato al crédito se pondera con una cierta cifra indicativa (suma integral de los rasgos) y si se sitúa por encima de un umbral determinado se considera como una persona digna de crédito.

Debido a que ambos métodos tienen aspectos negativos -subjetivamente por un lado y una simple dependencia lineal entre característica por el otro-, los esfuerzos apuntan a la aplicación de técnicas de inteligencia artificial, tales como Sistemas Expertos (XPS) y Redes Neuronales, en el campo de la puntuación para la solicitud de

un crédito. Por lo tanto la tarea a ser consumada por medio de una red neuronal es tratar de predecir una correcta clasificación de los clientes.

Escenario de los datos:

Para recaudar información acerca de los candidatos aprobados y desaprobados recurrieron a diferentes instituciones crediticias, pero solamente unas pocas de ellas cooperaron; ya que la mayoría alegaba que los datos sobre los aspirantes a los créditos era un asunto interno y por consiguiente, reciben un trato confidencial.

Puesto que la información de la que disponían era insuficiente, utilizaron un conjunto de datos públicos; el notorio conjunto de datos australiano (J. R. Quinlan), caracterizado como sigue en la *Tabla 7.3*:

Nº de casos	690
Clases	2 (buena/mala)
Distribución de clases	307 (buenas): clase 2 383 (malas): clase 1
Nº de atributos: -continuos -binarios -ordenados	14 6 4 4

Tabla 7.3

Codificación:

Codificaron los atributos continuos simplemente por una transformación lineal dentro del intervalo [0.1, 0.9], utilizando para esto solamente una neurona (real). Los atributos binarios también fueron codificados por medio de una neurona (binaria). Pensaron en la utilización de dos neuronas binarias, pero los ensayos no mostraron ninguna diferencia entre ambos tipos de codificación. En consecuencia utilizaron solamente una neurona para los atributos binarios.

A los atributos ordenados los trataron como sigue; ya que dependiendo de los diferentes valores que los mismos podían tomar, el número de neuronas utilizadas en la capa de entrada variaba. Codificaron cada posible valor en una neurona; por ejemplo si había tres valores posibles, se necesitaba de tres neuronas. El procedimiento de codificación fue como el mostrado en la *Tabla 7.4*:

Atributo	**Neurona de entrada**
valor 1	1 0 0
valor 2	0 1 0
valor 3	0 0 1

Tabla 7.4

La capa de salida estaba formada por una simple neurona binaria que daba una clasificación bueno/malo.

Utilizando el procedimiento de codificación de la *Tabla 7.4*, al menos 32 neuronas les fueron necesarias en la capa de entrada. Por supuesto otras formas de codificación de los atributos pueden ser posibles.

Resultados:

Para el entrenamiento y validación, dividieron el conjunto de datos de la *Tabla 7.3* en dos subconjuntos. El número de ejemplos en cada subconjunto, se puede

ver en la *Tabla 7.5*, en donde intentaron mantener una distribución similar de candidatos buenos y malos.

Subconjunto	Nº de ejemplos	Distribución
Entrenamiento	480	214 (buenos) 266 (malos)
Testeo	210	93 (buenos) 117 (malos)

Tabla 7.5

El mejor resultado que obtuvieron, es el presentado en la *Tabla 7.6* (Standard Backpropagation, tasa de aprendizaje: 0.2):

Arquitectura de la red (una capa oculta)	Épocas	Exactitud
32-40-1	66	80.6 %
32-32-1	100	80.4 %
32-16-1	100	82.5 %

Tabla 7.6

Cuando emplearon dos capas ocultas el resultado no mejoró, a pesar de tomar más tiempo de CPU. También modificaron el grado de aprendizaje, fijándolo en 0.5 y 0.7 e iniciando diferentes series de tiempo, pero los resultados tampoco mejoraron.

8 Software Comerciales

8.1 Aplicaciones del NeurOn-Line Studio a procesos de refinería y petroquímica.

Esta es una traducción textual del artículo publicado en la Webpage de Soteica S.R.L.: *NOL Studio Applications – Ruiz y Sonnet,* presentado en la Gensym User's Society (GUS) Meeting Barselona, Spain, April 2000.

Carlos A. Ruiz[1,2] , Ariel Sonnet[2]

[1]Soteica S.R.L.
Alvarez Thomas 796, 3 C, 1427 Buenos Aires, Argentina,
Tel.: +54-11-4555-5703, ext. 218, Fax: +54-11-4551-0751, e-mail: carlos@soteica.com.ar

[2]Grupo de Investigación Aplicada a la Ingeniería Química (GIAIQ),
Universidad Tecnológica Nacional, Facultad Regional Rosario, Zeballos 1341, 2000 Rosario, Argentina

Sumario:

Los procesos industriales han instalado extensamente, durante el transcurso de los últimos años, bases de datos históricas en tiempo real con gran capacidad de almacenaje. Los especialistas en Tecnologías de Información (*Information Technologies, IT*) hacen referencia a la metodología general para obtener información valiosa desde una gran base de datos como "*data mining*" (minería de datos). Una de tales metodologías son las Redes Neuronales (*Neuronal Networks, NNs*). El artículo describe la aplicación de un paquete de programas de NNs comercial (*NeurOn-Line Studio, Gensym Corporation, Cambridge, Massachusetts, USA*) a varios problemas de refinerías y petroquímicas. Se muestra como el moderno software es capaz de manejar apropiadamente la selección de la estructura de la NNs y la apropiada metodología de entrenamiento (es decir, minimizando la función objetivo adecuada, generalmente el problema del menor cuadrado). Se dan ejemplos sobre la utilización de NNs como analizadores virtuales, optimizadores y para la reducción de modelos.

Introducción:

Los procesos industriales han instalado extensamente, durante los últimos tiempos, bases de datos históricas en tiempo real con gran capacidad de almacenaje. En las refinerías e industrias petroquímicas tales bases de datos están recopilando datos del proceso en tiempo real desde los Sistemas de Control Distribuidos (*Distributed Control Systems, DCS*), a una típica frecuencia de muestreo de 1 minuto.

Generalmente, las bases de datos están disponibles en línea por muchos años e incluyen también los resultados de los análisis rutinarios de laboratorio. En una refinería típica, es normal recolectar miles de variables (*tags*), incluyendo variables de proceso y set-point, salidas de válvula y modo (por ejemplo: automático, manual, local, remoto, etc.) de los controladores. Estos historiadores en tiempo real están produciendo bases de datos muy grandes donde, aunque no siempre fácil de encontrar, puede extraerse una

muy rica información relacionada al proceso. Debido a técnicas especiales de compresión, tales bases de datos tienen una tremenda capacidad de almacenamiento. Como un ejemplo, si 40,000 tags del DCS son muestreados cada minuto (un modesto número de una refinería de petróleo de tamaño medio), 57.6 millones de puntos de datos se recogen cada día, totalizando alrededor de 21 billones de puntos de datos anualmente. No es inusual disponer on-line de varios años coleccionados, accesibles fácilmente utilizando herramientas de una PC estándar, tal como una hoja de cálculo *Excel*.

El "data mining" (también conocido como *Knowledge Discovery in Databases – KDD*) ha sido definido como "la extracción no trivial de información implícita, desconocida previamente, y potencialmente útil desde los datos" [Frawley et al., 1999]. Haciendo uso de máquinas de aprendizaje, estadística y técnicas de visualización para descubrir y presentar información en una forma en la cual es fácilmente comprensible por el ser humano. La metáfora de la mina es realmente potente: hay vetas de material rico (esto es, datos útiles desde el punto de vista de la economía, seguridad y operación del proceso), oculto en una inmensa cantidad de datos crudos almacenados, muchos de los cuales se pueden considerar como escoria. El esfuerzo para extraer el material precioso de la escoria se basa en varias técnicas matemáticas y de IT que ayudan en la tarea. Una de tales metodologías son las Redes Neuronales (NNs), las cuales se describen resumidamente más abajo, pero pueden rápidamente ser definidas como un artefacto matemático que necesita ser alimentado (esto es, recibir entradas) con datos históricos para ser entrenado en orden a predecir una o más variables (esto es, generar salidas).

Pero las NNs, para explotar correctamente las vetas del material rico que se hallan en las grandes bases de datos, no necesita solamente ser alimentada con el dato apropiado de entrada y salida, sino también cumplimentar los siguientes 2 requisitos importantes:

- La NN debe tener una estructura capaz de representar adecuadamente el problema.
- La NN debe ser entrenada con un algoritmo robusto y confiable, apto para converger a una solución aceptable.

No sorprendentemente, muchos de los artículos disponibles en este campo no abundan en las aplicaciones industriales de las NNs, pero si sobre los detalles matemáticos y características especiales de cada tipo de topología de NN (esto es: tipo de red, función de transferencia, número de capas) y/o algoritmo de entrenamiento (esto es: técnicas de minimización, temas de convergencia, detalles de programación, etc.). Muchas veces puede gastarse mucho esfuerzo luchando con la mejor topología de NN y perderse muchas horas de CPU intentando obtener un ajuste y convergencia razonable.

No fue hasta hace poco que los paquetes comerciales fueron capaces de prestar atención a los dos requisitos mencionados anteriormente. Este artículo presenta la experiencia recogida con la aplicación de uno de tales paquetes comerciales: NeurOn-Line Studio [Gensym, 1999], una herramienta poderosa para entrenar e implementar on-line las soluciones basadas en NNs.

Redes Neuronales: descripción de la terminología usual y del software.

Muy buenas descripciones teóricas y prácticas de la tecnología de las NNs pueden encontrarse en varias publicaciones, pero la colección de artículos editados por

Leonides (1998) puede mencionarse como una buena fuente de información sobre la teoría y práctica de las mismas. Esta sección solamente pretende dar una breve introducción a las capacidades del software NeurOn-Line Studio.

El NeurOn-Line Studio puede utilizarse off-line u on-line, siendo una herramienta para el análisis de procesos. Típicamente la fuente de datos es un historiador de datos u otro archivo de datos. Empleando poderosas herramientas de visualización, es posible analizar un amplio conjunto de datos desordenados de hasta 100,000 registros y más de 100 variables. El NeurOn-Line Studio provee una guía paso a paso a través del proceso de preprocesamiento de datos, configuración del modelo, entrenamiento, validación y puesta en línea. Para maximizar la productividad, muchas técnicas de decisión, tales como selección de las entradas relevantes, tiempos de retraso y arquitectura de la red, están automatizadas o convenientemente asistidas.

Una vez que un modelo se ha construido, es posible utilizar el NeurOn-Line Studio para descubrir formas más ventajosas para correr el proceso a través de la simulación y de la optimización. Sobre la base de una función objetivo, que expresa rentabilidad en términos de variables de proceso predichas y medidas, el NeurOn-Line Studio aplica el modelo de red neuronal para determinar las condiciones de operación óptima, dentro de las restricciones del caso.

Es posible implementar los modelos predictivos y las capacidades de optimización de NeurOn-Line Studio como controles ActiveX en el entorno de Windows NT, 2000 y 98. Los mismos pueden correrse en contenedores adecuados, que incluyen Visual Basic y aplicaciones C++, aplicaciones MS Office, y otros que siguen la norma COM, tales como los provistos por la mayoría de los proveedores de DCSs e historiadores de datos.

Los modelos del NeurOn-Line Studio pueden también fácilmente integrarse dentro del ambiente de las aplicaciones del sistema experto G2. Esta integración resulta estratégica, sobre todo por la colección de rutinas de conectividad del G2, orientación a objetos y su habilidad para representar reglas expertas en lenguaje natural estructurado. Empleando los modelos del NeurOn-Line Studio en esta forma aumentan la capacidad del G2 para el manejo inteligente de operaciones de proceso.

Aplicación de Redes Neuronales a la industria de procesos.

Como se ha mencionado, las Redes Neuronales permiten al ingeniero crear modelos para procesos utilizando datos históricos del mismo proceso. Los modelos pronostican cómo el proceso responderá a los cambios de entradas y diferentes condiciones de trabajo. Las condiciones de operación óptima, sujeta a restricciones, pueden también determinarse con las NNs correctamente formuladas. Los modelos identificados empleando NNs pueden utilizarse en estudios de proceso off-line o ser instalados on-line para suministrar una detección precoz de los problemas de proceso y determinar los set-point que continuamente optimicen el proceso para maximizar las ganancias.

Las NNs traen a la vida los datos históricos, revelando los factores más importantes que afectan la calidad y el rendimiento de los productos. Este conocimiento puede a menudo detectar mejoras sin ninguna inversión de capital. Las áreas generales de utilización potencial de las NNs son las siguientes:

Control de Calidad, Sensores Inferenciales y Reducción de Modelos. En la economía globalizada de hoy en día, el gerenciamiento de la calidad en tiempo real es una aplicación de vital importancia, pero los ensayos de calidad raramente están disponibles sin retardos y usualmente son onerosos. Los modelos basados en redes neuronales proporcionan medidas "virtuales" en tiempo real, permitiendo acciones de control rápidas para mantener la calidad en el objetivo deseado. Los modelos pueden ser obtenidos no sólo a partir de los datos de planta y laboratorio sino de datos generados con corridas de modelos de simulación rigurosos (desarrollados, por ejemplo, en HYSYS). Este último procedimiento se conoce como "reducción de modelos".

Optimización de Procesos. El valor de la optimización basada en modelos está bien probado pero, en general, los modelos analíticos de un proceso pueden ser muy difíciles de obtener. Al emplear redes neuronales en conjunto con su capacidad de optimización en línea y en tiempo real, puede ser posible obtener el mayor potencial económico de un proceso.

Mantenimiento Predictivo y Seguridad. Los modelos basados en redes neuronales pueden ser empleados para monitorear la performance de máquinas y equipos. Con ellos se pueden detectar tempranamente corrimientos o errores en los modelos operativos o sensores, permitiendo a los ingenieros corregir los problemas antes que devengan en incidentes mayores. Se puede mejorar, en consecuencia, la disponibilidad de plantas y equipos. El monitoreo continuo del contenido de emisiones (*CEM, Continuous Emissions Monitoring*) de NOx, CO_2, SO_2 en los gases de escape de hornos y calderas es una aplicación típica en esta área.

Validación de Sensores. La deriva progresiva y/o falla abrupta de las señales de sensores son la fuente principal de paradas de planta no planeadas y producción de productos fuera de especificación. Con los modelos basados en redes neuronales es posible seguir los valores de los sensores y generar alarmas cuando las medidas provenientes de los sensores físicos no están de acuerdo con los valores inferidos para los mismos. El valor inferido puede ser empleado también como línea de base en los casos en que el instrumento es recalibrado o reparado.

Predicción y Estimación. El futuro puede ser predicho dentro de la precisión que dan los modelos basados en comportamientos. Las redes neuronales pueden aprender los modelos óptimos, adaptados continuamente con el empleo de los últimos datos medidos. Los ingenieros pueden emplear estas predicciones para estimar la demanda de mercados de corto plazo, predecir estados futuros del proceso o aún condiciones meteorológicas que afecten a las emisiones e impacten sobre la vecindad de la planta.

De algunas de las muchas aplicaciones experimentadas actualmente con el software: "NeurOn-Line Studio", los ejemplos presentados en este artículo están resumidos en la *Tabla 1*:

Ejemplo	Área de aplicación de las NNs.	Descripción del potencial tecnológico de las NNs.	Modelos del NeurOn-Line.
Predicción del punto final de la curva de destilación de las naftas de un FCCU.	Control de Calidad y Sensores Inferenciales	Los equipos actuales para la determinación en línea del punto final de destilación de las naftas son costosos, poco confiables y tienen severos problemas de mantenimiento. La aplicación de las NNs como Sensores Inferenciales es una opción atractiva.	Modelo Inferencial con tiempo muerto aplicado a las entradas.
Predicción de la calidad y reducción de modelos de una columna Splitter de propano/propileno.	Sensores Inferenciales, Reducción de Modelos y Optimización de Procesos.	Existen modelos de simulación detallados y rigurosos para la columna (desarrollado con el simulador HYSYS), pero el operador quiere tener una estimación rápida de las calidades de los productos de la columna y conocer cual condición de operación debe fijarse para obtener las especificaciones objetivo.	Modelo Inferencial, Modelo de Optimización.
Estimación on-line del punto de weathering del LPG.	Control de Calidad y Sensores Inferenciales.	La determinación del weathering point se realiza una vez por turno en planta y una vez al día por el Laboratorio. Consiste de una metodología empírica para la cual no hay ningún equipamiento de análisis en línea disponible. El weathering point es una especificación importante para el LPG consumido en las viviendas.	Modelo Inferencial con tiempo muerto aplicado a las entradas.

Tabla 1: ejemplos de aplicaciones de NNs

Predicción del punto final de las naftas FCCU.

Con el objetivo de ejecutar una prueba del concepto para la aplicación del NeurOn-Line Studio como un sensor virtual, los datos fueron colectados desde el historiador de datos de la planta *PI* (OSI Soft) de la Refinería de La Plata de Repsol-YPF (Ensenada, Pcia. de Buenos Aires, Argentina). El objetivo fue inferir el 90% del Punto de Destilación de la nafta de la Unidad de Cracking Catalítico Fluidizado (*Fluidized Catalytic Cracking Unit – FCCU*). Este caso fue descripto en detalle por Ruiz (1999) pero una breve descripción se da a continuación.

Los datos fueron colectados utilizando la interfase *Excel DataLink* del historiador PI, colectando inicialmente 63 variables operativas, durante el período de junio-octubre de 1998. El volumen total de información recogida fue de 80 Mbytes aproximadamente. Después del filtrado de los datos, inspección para identificar los períodos de operación inestables o mal funcionamiento de los sensores, un período de 8 días fue seleccionado como el conjunto de entrenamiento. Un conjunto de validación independiente también fue extraído, para utilizarse sobre la validación del modelo finalmente entrenado.

El conjunto de datos de entrenamiento fue inspeccionado a fondo para detectar los períodos malos. La *Figura 1* refleja el período cuando el caudal de alimentación a la FCCU se cambió de 170 a 180 m^3/h. El NeurOn-Line Studio permite, directamente desde el gráfico, embanderar los datos para ser previamente preprocesados al entrenamiento de la NN.

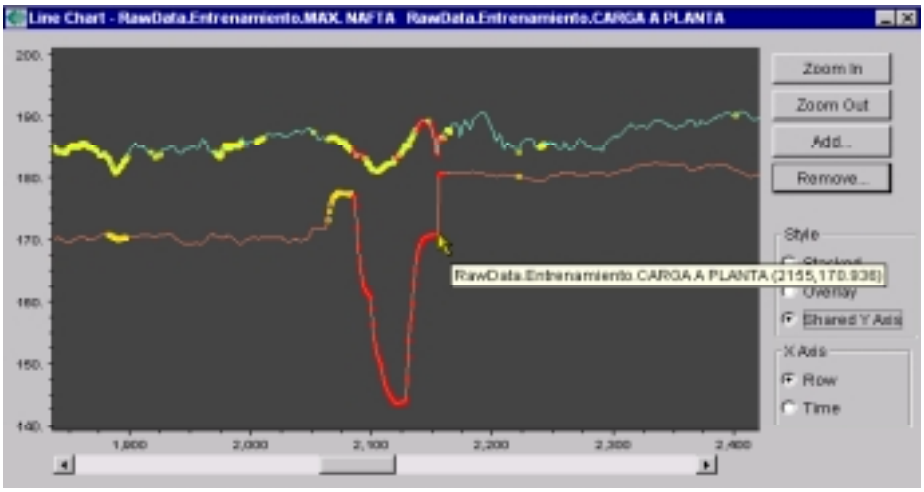

Figura 1: cambio en el flujo de alimentación a la FCCU (marcados en rojo) de 170 a 180 m^3/h.

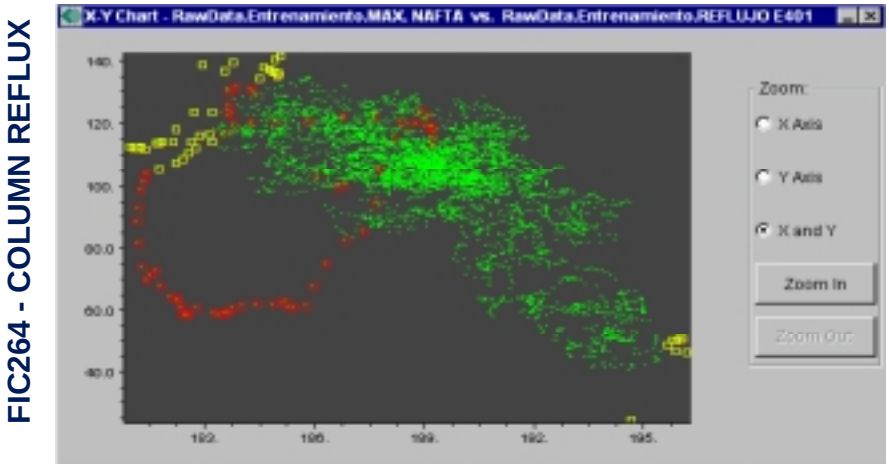

Figura 2: gráfico X-Y mostrando la relación entre el caudal del reflujo de la columna principal y la variable a ser inferida (Punto del 90% de la nafta).

Figura 3: predicción del punto final para la nafta de la FCCU (línea amarilla) vs. el actual (línea verde) durante el período de validación.

En la *Figura 2* un gráfico X-Y muestra la relación del caudal de reflujo con el punto final de la nafta. En el mismo, el período inestable correspondiente al caudal de alimentación de la planta se marcó también en rojo. Los otros puntos de operación marcados en amarillo son una porción también eliminada del período de entrenamiento porque una válvula de alivio de presión se abrió causando una perturbación en la columna principal.

Se entrenaron dos modelos, el primero de ellos utilizando como entradas 29 variables de proceso con diferentes retrasos. Algunas de ellas se emplearon con el valor instantáneo y otras retrasadas en un rango de 30 minutos a 1 hora. El segundo modelo se entrenó empleando solamente 18 variables (modelo reducido), todas ellas retrasadas 1 hora. Ambos modelos exhibieron buenas predicciones del punto de destilación 90% de la nafta. Se prefirió el modelo reducido porque la menor cantidad de variables involucradas lo hacen más robusto con respecto la falla de los sensores. La *Figura 3* muestra el valor predicho y el actual, obtenido a partir del modelo reducido, para el período de validación.

Predicción de la calidad y reducción del modelo en una columna Splitter de propano/propileno.

Un modelo riguroso, basado en los principios fundamentales, de una columna de destilación empleada para separar una mezcla de propano y propileno se desarrolló utilizando el simulador HYSYS. El diagrama de flujo del proceso se exhibe en la *Figura 4*. Con la utilidad del caso de estudio de HYSYS (*Case Study*), se generó una grilla de alrededor de 900 puntos (esto es, 900 corridas), como se muestra en la *Figura 5*. Las variables de operación que se toman en consideración para el entrenamiento de la NN se presentan en la *Figura 6*. Para predecir el propileno producto y la composición de fondo del propano, se emplearon una mezcla de las variables manipuladas (por ejemplo, el caudal de reflujo a la columna y el calor en el reboiler), las variables externas (como el caudal de alimentación y la composición) y las variables de estado (como la temperatura de la alimentación).

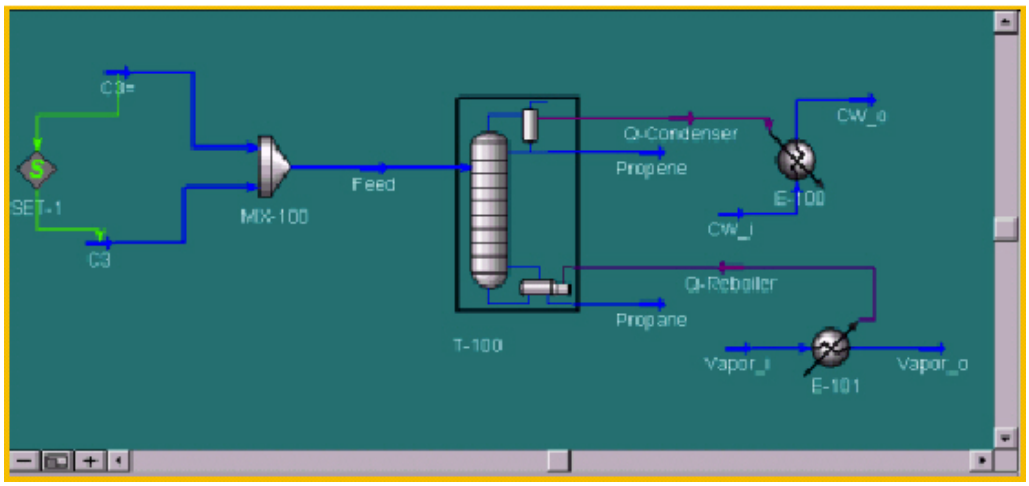

Figura 4: diagrama de flujo del proceso obtenido a través del simulador HYSYS de la columna de separación C3/C3=.

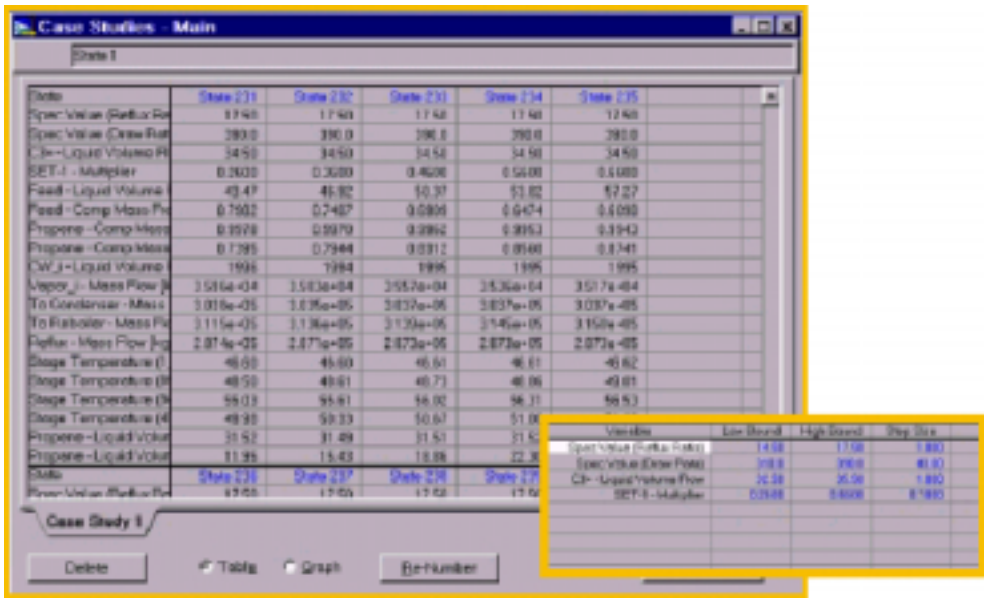

Figura 5: facilidad del caso de estudio HYSYS empleado para generar la grilla de entrenamiento.

Como la grilla de datos generada para entrenar el modelo está libre de ruidos, la NN entrenada muestra una muy buena concordancia. En la *Figura 7* puede mostrarse cuan buena es la predicción de la composición de producto de tope (propileno), ya sea en el gráfico de línea o en el X-Y, donde la composición predicha y al actual se comparan casi exactamente.

Tag	Name	Description	Units
T-100	Tower_RR	Spec Value (Reflux Ratio)	adim.
T-100	Propene_Draw_Rate	Spec Value (Draw Rate)	kgmole/h
C3=	C3=_Vol_Rate	Liquid Volume Flow	m3/h
SET-1	SET-1_Ratio	Multiplier	adim.
Propene	**Propene_Mass_Frac_C3=**	**Comp Mass Frac (Propene)**	**adim.**
Propane	**Propane_Mass_Frac_C3**	**Comp Mass Frac (Propane)**	**adim.**
CW_i	Condenser_CW_Rate	Liquid Volume Flow	m3/h
Vapor_i	Reboiler_Steam_Rate	Mass Flow	kg/h
To Condenser	To_Condenser_Rate	Mass Flow	kg/h
To Reboiler	To_Reboiler_Rate	Mass Flow	kg/h
Reflux	Reflux_Rate	Mass Flow	kg/h
T-100	Tower_Stage_1_Temp	Stage Temperature (1__Rectifier)	C
T-100	Tower_Stage_89_Temp	Stage Temperature (89__Rectifier)	C
T-100	Tower_Stage_94_Temp	Stage Temperature (94__Stripper)	C
T-100	Tower_Stage_47_Temp	Stage Temperature (47__Stripper)	C
Propene	Propene_Volume_Flow	Liquid Volume Flow	m3/h
Propane	Propane_Volume_Flow	Liquid Volume Flow	m3/h
Feed	Feed_Vol_Rate	Liquid Volume Flow	m3/h
Feed	Feed_Mass_Frac_C3=	Comp Mass Frac (Propene)	adim.

Figura 6: lista de las variables de HYSYS utilizadas para entrenar la NN.

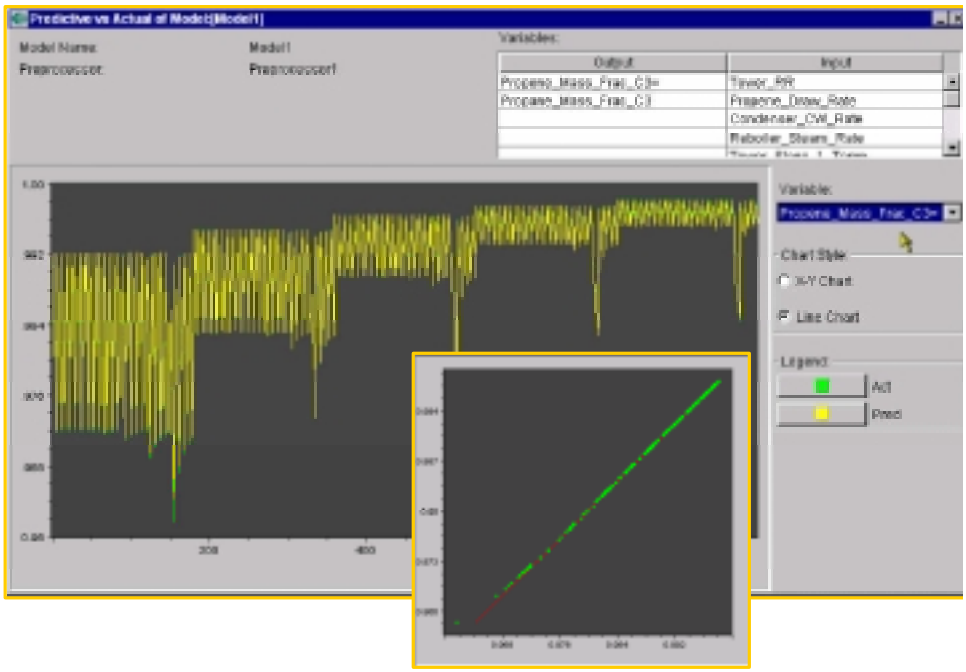

Figura 7: valores del modelo de HYSYS vs valores predichos por el NeurOn-Line Studio para la composición de tope del propileno (ambos gráficos, de línea y X-Y).

Adicionalmente al modelo reducido de la NN, se creó un modelo de optimización. En este caso, el objetivo es obtener una cierta especificación de las composiciones de tope y fondo, manipulando el operario las variables, tomándose en consideración las perturbaciones externas y satisfaciendo todas las restricciones, incluyendo las variables de estado. En la *Figura 8*, se presenta la estructura del modelo de optimización. Las variables manipuladas, en este caso, son el caudal del reflujo de la columna tower y el caudal de producto, como se muestra en la *Figura 9*.

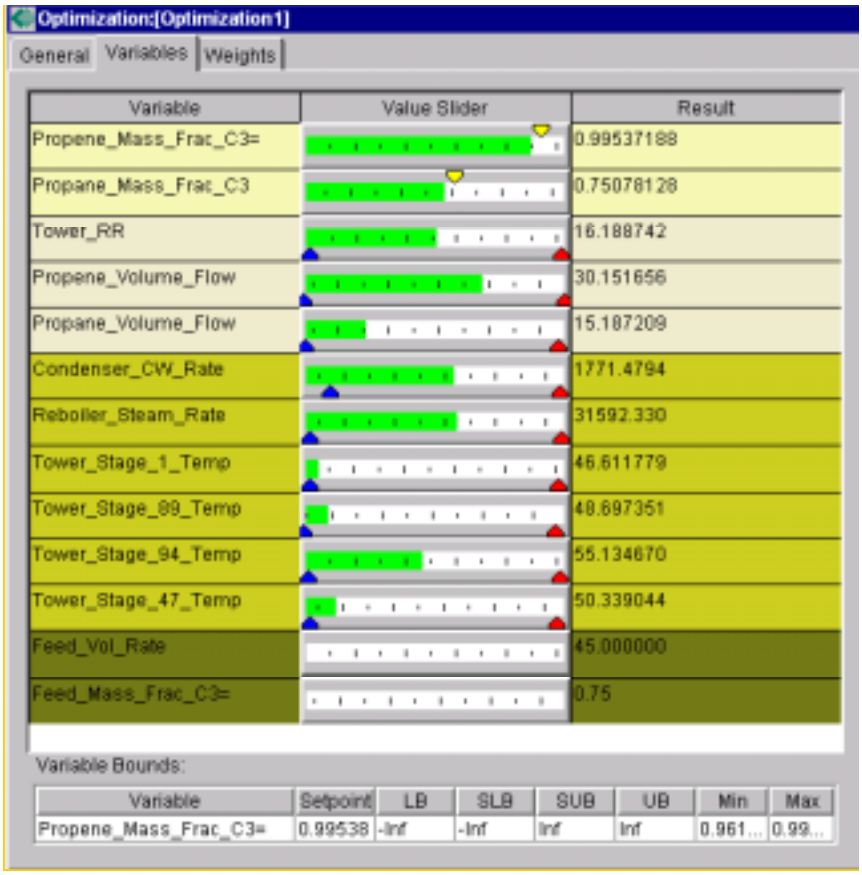

Figura 8: red de optimización para el separador C3.

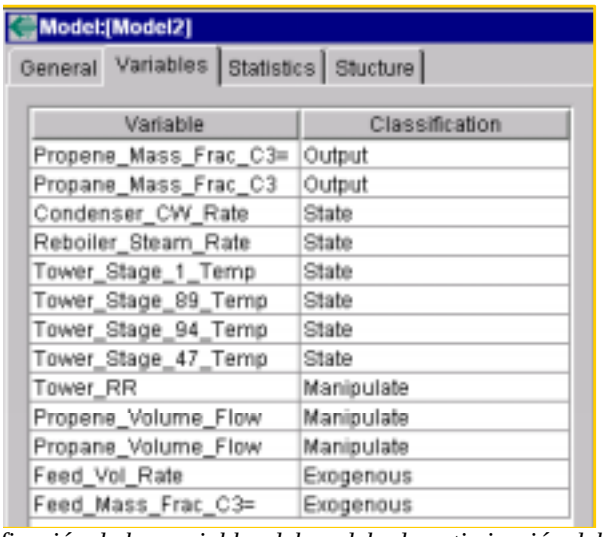

Figura 9: clasificación de las variables del modelo de optimización del separador C3.

Estimación on-line del punto de weathering del LPG.

Para controlar las especificaciones del producto propano, el operador hace determinaciones on-site del weathering point de ambos productos: propano y butano de la columna depropanizadora de la FCCU II, de la Refinería de Luján de Cuyo de Repsol-YPF. Estos análisis manuales se ejecutan en el campo, con intervalos de aproximadamente 8 horas y reportados en una hoja de cálculo Excel. La planta tiene instalado también un cromatógrafo on-line sobre ambos productos de tope y fondo.

Se desarrolló una NN para predecir el weathering point del propano, basada en las condiciones de operación de la columna y cromatógrafos on-line. Los datos de operación se recogieron desde el historiador de datos *PI* (período mayo-junio 1999) y combinados con los datos de análisis generados por el operador de planta. El weathering point es una función directa (pero no sencilla) de la composición de las corrientes de productos. El primer modelo fundamental falló al predecir el weathering point correcto porque el análisis de rutina es una evolución ni isotérmica ni adiabática, pero sí politrópica. La metodología de la red neuronal es capaz de considerar el error metodológico del laboratorio.

Una vez más, se prepararon dos series de datos, una para el entrenamiento y otra para la validación de la NN entrenada. En la *Figura 10* se exhibe el weathering point del 95% propano de para ambas series de datos.

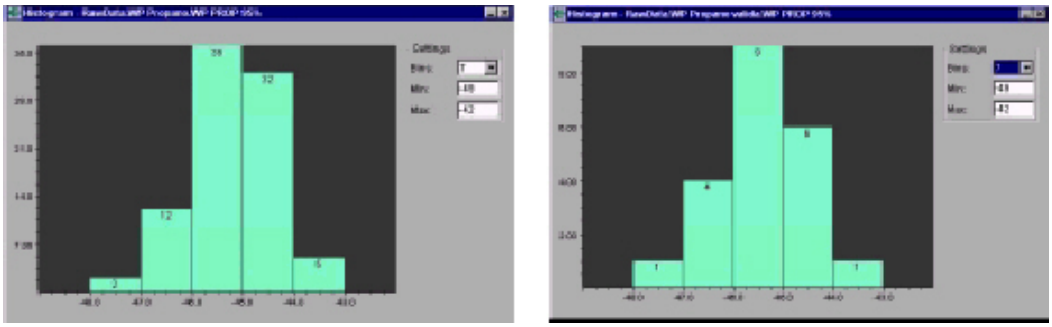

Figura 10: histograma del weathering point para ambos conjuntos de datos, entrenamiento (izquierda) y validación (derecha).

La NN entrenada fue capaz de predecir muy bien la determinación del weathering point de la planta. En la *Figura 11* puede exhibirse la predicción continua a través de un período de una semana. En el mismo gráfico están superpuestas las determinaciones hechas en el lugar.

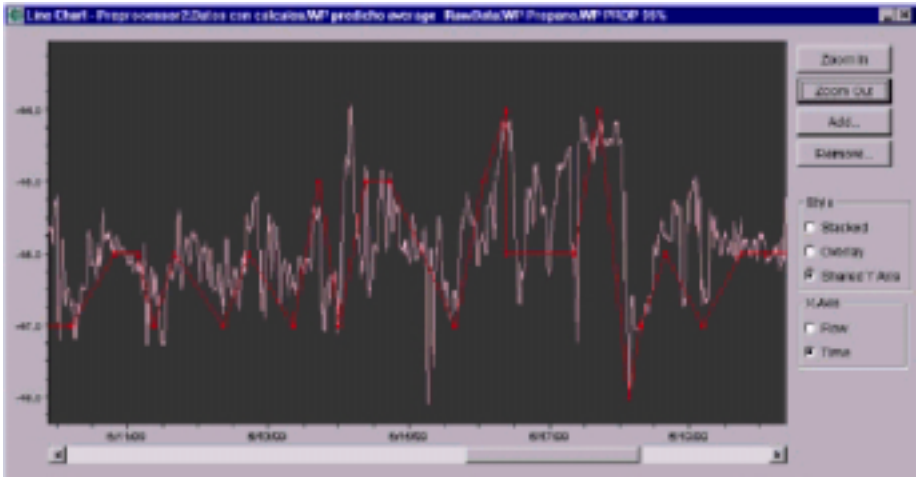

Figura 11: predicción del weathering point del propano (rosa) vs el valor actual (rojo).

Conclusiones:

La tecnología de la NN ha demostrado ser una forma muy poderosa de obtener correlaciones buenas y confiables para ejecutar la aplicación de analizadores virtuales y para instalar en línea herramientas de optimización de proceso. El software, NeurOn-

Line Studio, para desarrollar y poner en línea tal tecnología y cubrir las necesidades del personal de ingeniería, de tal manera que ningún profesionalismo se necesite poseer sobre las NNs, pero sí sobre el proceso y las relaciones causa-efecto.

Reconocimientos:

Queremos agradecer a las siguientes personas quienes suministraron los datos y participaron en el desarrollo de algunos de los ejemplos presentados:

- Sr. Carlos Lago, Repsol YPF, Centro de Tecnología Aplicada,
- Sr. Leonardo Rivas, Repsol YPF, Refinería Luján de Cuyo.

Referencias:

- Frawley W., Piatetsky-Shapiro G., Matheus, C. (1992), "Knowledge Discovery in Databases: An Overview". AI Magazine, Fall 1992, 213-228.
- Leonides, C. T. (1998), "Neural Network Systems Techniques and Applications", Volumes 1 to 7, Academic Press, San Diego, California, USA.
- Gensym Corporation (1999), “NeurOn-Line Studio User’s Guide”, Cambridge, Massachusetts.
- Ruiz C. (1999), “Predicting the 90% Distillation Point of a Fluidized Catalytic Cracking Unit Naphtha. An Application of NOL Studio”, Gensym’s Users Society Conference (GUS ’99), Orlando, Florida, USA.

8.2 Software que pueden ser empleados en la industria de procesos.

En la *Tabla 8.1* se detalla una lista con el nombre de varios software comerciales de redes neuronales:

Software	Compañía	Webpage
NeurOn-Line	Gensym Corporation 125 Cambridge Park Drive. Cambridge, Massachussetts 02140 *Representante en Argentina.* Soteica S.R.L.: Av. Alvarez Thomas 796 - 3º C, (1427) Buenos Aires, Argentina.	http://www.gensym.com http://www.soteica.com.ar
HNeT (Holographic/Quantum Neuronal Technology).	AND Corporation (212) 279-3833 Nueva York, (416) 920-8260 Toronto	http://www.andcorporation.com
NeuroMonitor Pro.	ERA Technology Cleeve Road, Leatherhead, Surrey, KT22 7SA UK.	http://www.era.co.uk
NeuroShell Predictor.	Ward Systems Group, Inc. Executive Park West, 5 Hillcrest Dr. Federick , MD 21703	http://www.warsystems.com http://www.neuroshell.com
NeuroDynamX®	DynaMind®	http://neurodynamx.com
NeuroSolutions	NeuroDimension Inc.	http://www.nd.com
Neural OptiMatch ™	Neural®	http://www.neural.com

Tabla 8.1

9 Bibliografía

- Neuronal Networks: Basics and Applications, by R. Lackes and D. Mack, in collaboration with J. Ziola and K. Ahern. CBT (Computer Based Training) Springer, Verlag Berlin Heidelberg 1998.

- Proyecto Final de la Carrera Ingeniería Electrónica, Facultad de Ciencias Exactas, Ingeniería y Agrimensura – Universidad Nacional Rosario, por Daniel Giardina. Directora del proyecto: Marta Basualdo. Año: 1995. Argentina.

- NOL Studio Applications – Ruiz y Sonnet. (www.soteica.com.ar)

- Tutorial de Redes Neuronales. Universidad Politécnica de Madrid, España (www.gc.ssr.upm.es/inves/neuronal/ann2/anntuto.htm).

- Introducción a las redes neuronales artificiales, por Alfredo Catalina Gallego (www.gui.uva.es).

- Control de procesos mediante redes neuronales, por Cristian F. Garrido Cisterna (http:\\melquiades.dic.udec.cl/~cgarrido).