REDES NEURONALES

ABSTRACT
By these days when we have to improve everything from our lives to the tools we use in order to make easier the activities, where we locate our selves to analyze the importance that implier the posibility of creating control algorithms that let emulate the characteristics of the biologic organisms. For any enginneer, is interesting to know the maximun point he can go, and to do it, use the Artificial Neural Network as one of the many tools to attain it. With Neural Networks is necessary give to system the steps for make some action. The advantage of Neural Networks is that this have the experience and that make the system more independence, “Not request to program a sistem makes many expectatives and many doubts since it’s and advantage that could become in a whole conflict between man and machine”.

INTRODUCCIÓN

Una de las razones por la cual los seres humanos y los computadores se necesitan mutuamente, es la diferencia radical que hay entre ellos. Los computadores hacen muy bien tareas que nosotros realizamos torpemente y viceversa. Por ejemplo, los computadores son lógicos y exactos; nosotros por el contrario no siempre lo somos. Los computadores pueden almacenar, sin errores, cantidades impresionantes de todo tipo de información; los humanos que intenten hacer algo similar pertenecen a los archivos de Guiness. Pero, los humanos podemos manejar, en escalas que llegan a la genealidad, la incertidumbre, la ambigüedad, la suposición e integrar en conclusiones, información proveniente de diferentes y multiples fuentes. El sistema biológico de un insecto es miles de veces más poderoso y flexible que cualquier programa de computador por extenso y poderoso que sea.

Desarrollar un sistema de inteligencia artificial con la flexibilidad, la creatividad y la capacidad de aprendizaje del sistema biológico humano, se constituyó desde hace muchos años en un reto atractivo para los científicos de muchas disciplinas. Los intentos por imitar el funcionamiento del cerebro han seguido la evolución del estado de la tecnología. Por ejemplo, al finalizar el siglo XIX se le comparó con la operación de la bomba hidráulica; durante la década de 1920 a 1930 se intento utilizar la teoría de la conmutación telefónica como punto de partida de un sistema de conocimiento similar al del cerebro. Entre 1940 y 1950 los científicos comenzaron a pensar seriamente en las redes neuronales utilizando como concepto la noción de que las neuronas del cerebro funcionan como interruptores digitales que puedan dispararse (on – off) de manera también similar al computador digital. Así nace la idea de "revolución cibernética" que maneja la analogía entre el cerebro y el computador digital y a su vez, crea la necesidad de estudiar y analizar dichos sistemas.

Las redes neuronales son un elemento importante de las denominadas tecnologías de Inteligencia Artificial (IA).

La IA es "la disciplina científica y técnica que se ocupa del estudio de las ideas que permiten ser inteligentes a los ordenadores" (definición de H. Winston). Otra posible definición sería que la IA es una rama de la computación que se encarga, entre otras cosas, de los problemas de percepción, razonamiento y aprendizaje en relación con sistemas artificiales, y que tiene como áreas de investigación a los sistemas expertos y de conocimiento, la robótica, los lenguajes naturales y las redes neuronales.

Y, a pesar de que el objetivo final de la IA, es dotar de autentica inteligencia a las máquinas, queda todavía muy lejos (e incluso hay autores que defienden que esto nunca será posible), la ciencia de la Inteligencia Artificial ha generado numerosas herramientas prácticas, entre las que se encuentran las redes neuronales.

Algunas definiciones de redes neuronales artificiales (Artificial Neural Network) dadas en el transcurso del tiempo son:

Modelos bastante simplificados de las redes de neuronas que forman el cerebro; y, al igual que este, intentan "aprender" a partir de los datos que se le suministran.
Interconexión masiva y en paralelo de elementos simples organizados jerárquicamente; basándose en la emulación por medio de hardware o software de la actividad de las neuronas del sistema nervioso biológico.
Serie de neuronas individuales (que son los elementos procesadores). Cada neurona actúa como un elemento procesador independiente, las entradas y los pesos de interconexión son procesados por una función suma (típicamente una sumatoria de pesos), el resultado de esta sumatoria es mapeado por una función de transferencia de característica no lineal (generalmente una función sigmoide). La salida de esta función no lineal es la salida de la neurona.
Modelos matemáticos especializados que pueden aplicarse en dominios muy concretos. Las redes neuronales están mostrando su utilidad en muchos problemas reales.
Múltiples capas de neuronas interconectadas con otras neuronas en la misma o en diferentes capas.
Dispositivos procesadores (algoritmos o hardware), que tratan de modelar la estructura de la corteza cerebral animal, pero en mucho menor escala. Una Red Neuronal Artificial grande puede tener cientos o miles de unidades procesadoras, mientras que el cerebro animal, tiene billones de neuronas con su respectivo incremento en magnitud debido a la interacción y al comportamiento.
" ... sistema computacional hecho por un alto número de simples elementos procesadores, pero altamente interconectados, los cuales procesan la información por la respuesta en estado dinámico de entradas externas" [1]

Las Redes Neuronales deben "aprender" cómo procesar la información de entrada antes de que ésta pueda ser utilizada en una aplicación. El proceso de entrenamiento de una Red Neuronal involucra el ajuste de los pesos de entrada en cada neurona hasta que la salida de la red se aproxima a la salida deseada. Este procedimiento involucra la creación de un archivo de entrenamiento, el cual está formado por los datos de cada nodo de entrada y la respuesta deseada para cada nodo de salida de la red. Una vez que la red está entrenada, sólo los datos de entrada son provistos a la red, la cual "recuerda" la respuesta que "aprendió" durante el entrenamiento.

HISTORIA

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 se presentaron alrededor de 1943 y fueron Warren McCulloch y Walter Pitts quienes, se basaron en tres fundamentos; conocimientos sobre fisiología básica y funcionamiento de las neuronas en el cerebro, el análisis formal de la lógica proposicional de Russell y, la teoría de la computación de Turing. En su trabajo "A logical calculus of the ideas immanent in nervous activity", propusieron un modelo constituido por neuronas artificiales, en el cual se caracterizaba por estar "encendidas o "apagadas"; el "encendido " se daba como respuesta a la estimulación

[1] *Hecht-Nielsen, inventor de una de las primeras neurocomputadoras.*

producida por una cantidad suficiente de neuronas vecinas. El estado de una neurona se veía como "equivalente", de hecho, a una proposición que propusiera sus estímulos adecuados.

Donald Hebb señaló en 1949 en su trabajo "The Organization of Behavior" que si dos neuronas que están interconectadas entre sí, se activan al mismo tiempo esto indica que existe un incremento en la fuerza sináptica. Así mismo, la forma de corrección que emplea esta regla, es incrementar la magnitud de los pesos si ambas neuronas están inactivas al mismo tiempo.

En la década de los cincuenta, Minsky comienza a construir la primera neurocomputadora (basada en modelos de redes neuronales que imitan al cerebro).

Otro pionero fue el psicólogo Frank Rosenblatt de la Universidad de Cornell. En 1959, Rosenblatt construyó una máquina neuronal simple que él llamó perceptrón. El perceptrón tenía una matriz con 400 fotoceldas que se conectaban aleatoriamente a 512 unidades tipo neurona. Cuando se presentaba un patrón a las unidades sensoras, estas enviaban una señal a un banco de neuronas que indicaba la categoría del patrón. Rosenblatt reconoció todas las letras del alfabeto con el perceptrón.

En los años cincuenta y sesenta el movimiento en redes neuronales fue liberado por Bernard Widrow y Marcial Hoff. En 1959, desarrollaron los modelos ADALINE (ADAptive LINear Elements) y MADALINE (Multiple ADALINE) . Estos modelos fueron los primeros en aplicarse a problemas reales (filtros adaptativos para eliminar ecos en las líneas telefónicos).

En 1967 Stephen Grossberg realizó una red, Avalancha, que consistía en elementos discretos con actividad que varía con el tiempo que satisface ecuaciones diferenciales continuas, para resolver actividades tales como reconocimiento continuo del habla y aprendizaje del movimiento de los brazos de un robot. Además, junto con Gail Carpenter de la Univeridad Northeastern han propuesto un modelo de red neuronal llamado ART (Adaptative Resonance.Theory). Grossberg es el director del centro para sistemas adaptativos de la Universidad de Boston

En 1969 Marvin Misnsky y Seymour Papert, publicaron un libro, Perceptróns que, además de contener un análisis matemático detallado del Perceptrón, consideraba la extensión a Perceptrónes multinivel. Las limitaciones del Perceptrón eran importantes, sobre todo su incapacidad para resolver muchos problemas interesantes y simples tales como sintetizar la función lógica XOR. Las matemáticas del libro Perceptróns eran indiscutibles y su tono dio el mensaje que los perceptrónes eran un camino sin salida lo cual hizo que se perdiera interés en el campo de las redes neuronales hasta la década de los 80, en que el estudio de nuevas arquitecturas de redes y la mayor potencia de los ordenadores permiten el diseño de redes muy eficientes en tareas en las que otros procedimientos de tipo simbólico encuentran dificultades.

En el mismo año, James Anderson desarrolló un modelo lineal, llamado Asociador lineal, que consistía en unos elementos integradores (neuronas) lineales que sumaban sus entradas. Este modelo se basa en el principio de que las conexiones entre neuronas son reforzadas cada vez que están activadas.

Otros investigadores que trabajaron durante los años 1970s fueron Teuvo Kohonen de la Universidad de Helsinki y Jim Anderson de la Universidad de Browm.

Geoff Hinton llegó en 1978 a la Universidad de California en San Diego desde Inglaterra. Hinton trabajó en inteligencia artificial tradicional en la Universidad de Cambridge y luego en la Universidad de Edinburgo.

James Anderson trajo algunas semillas de conexionismo a la UCSD en los setentas y junto con Hinton organizaron el primer encuentro neo-conexionista en el verano de 1979, al cual asistieron: David Rumelhart, McClelland, Jerry Feldman y Terry Sejnowski. Luego publicaron una serie de artículos doble redes neuronales y memorias asociativas.

En Europa y Japón, las investigaciones también continuaron. Kunihiko Fukushima desarrolló el Neocognitrón, un modelo de red neuronal para el reconocimiento de patrones visuales.

En 1982, John Hopfield presentó una red que llevaba su mismo nombre; una variación del Asociador Lineal, pero, además, mostró cómo tales redes pueden trabajar y qué pueden hacer, entre sus aplicaciones están la reconstrucción de patrones y la optimización. Se utilizan funciones de energía para entender las redes dinámicas.

Cohen y Grossberg desarrollan en el 83 el principio de la memoria direccional.

En 1985 se realizó la primera reunión anual de redes Neuronales Neural Networks for Computing.

En 1986 Rumelhart, Hunton y Williams redescubren el algoritmo de "back-propagation" (desarrollado en 1974 por Paul Werbor) para el aprendizaje de redes neuronales, entre sus aplicaciones están la síntesis de voz de texto y control de robots. Por estas fechas, y gracias a las nuevas tecnologías de fabricación de microchips, comienzan a construirse redes neuronales implementadas en silicio (mucho más rápidas que las de software).

Rumelhart junto con McClelland publicaron un libro en dos volumenes titulado Parallel Distribuited Processing: Explorations in the Microstructure of Cognition. Este libro se considera un clásico en el área de redes neuronales y se puede decir que su aparición trajo un nuevo impulso a la investigación en sistemas neuronales.

Actualmente se siguen haciendo investigaciones y trabajos acerca de las RNA. Además, el uso de redes neuronales se ha extendido bastante en el mercado de software doméstico, dejando de estar restringidas a los entornos de investigación y a las grandes empresas. De esta forma, se pueden encontrar modelos de redes neuronales en programas de reconocimiento de voz, en juegos de ordenador, programas de contabilidad, tutores, y muchos otros.

CARACTERÍSTICAS

Aprendizaje adaptativo

La capacidad de aprendizaje adaptativo es una de las características más atractivas de las 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 entrenamiento no es necesario que elaboremos modelos a priori ni necesitamos especificar funciones de distribución de probabilidad.

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

Autoorganización

Las redes neuronales usan 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 Backpropagation, creará su propia representación característica, mediante la cual puede reconocer ciertos patrones.

Tolerancia a fallos

Las redes neuronales son los primeros métodos computacionales con la capacidad inherente de tolerancia a fallos. Comparados con los sistemas computacionales tradicionales, los cuales pierden su funcionalidad en cuanto sufren un pequeño error de memoria, en las redes neuronales, si se produce un fallo en un pequeño número de neuronas, aunque el comportamiento del sistema se ve influenciado, sin embargo no sufre una caída repentina.

Hay dos aspectos distintos respecto a la tolerancia a fallos: primero, las redes pueden aprender a reconocer patrones con ruido, distorsionados o incompletos, ésta es una tolerancia a fallos respecto a los datos. Segundo, 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 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. Las redes neuronales almacenan información no localizada. Por tanto, la mayoría de las interconexiones entre los nodos de la red tendrán unos valores en función de los estímulos recibidos, y se generará un patrón de salida que represente la información almacenada.

Operación en tiempo real

Una de las mayores prioridades de la mayoría de las áreas de aplicación, es la necesidad de realizar grandes procesos con datos de forma muy rápida. Las redes neuronales se adaptan bien a esto debido a su implementación paralela. Para la mayoría de las redes pueden operar en un entorno de tiempo real, la necesidad de cambio en los pesos de las conexiones o entrenamiento es mínima. Por tanto, de todos los métodos posibles, las redes neuronales son la mejor alternativa para reconocimiento y clasificación de patrones en tiempo real.

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). Debido a que una red puede ser rápidamente entrenada, comprobada, verificada y trasladada a una implementación hardware de bajo coste, es fácil insertar redes neuronales para aplicaciones específicas dentro de sistemas existentes. De esta manera, las redes neuronales se pueden utilizar para mejorar sistemas de forma incremental y cada paso puede ser evaluado antes de acometer un desarrollo más amplio.

VENTAJAS Y DESVENTAJAS DE LAS REDES NEURONALES

Algunas ventajas de las redes neuronales frente a otros sistemas de procesamiento de información son:

Las redes neuronales pueden sintetizar algoritmos a través de un proceso de aprendizaje.
Para utilizar la tecnología neuronal no es necesario conocer los detalles matemáticos. Sólo se requiere estar familiarizado con los datos de trabajo.
La solución de problemas nolineales es uno de los fuertes de las redes neuronales.
Las redes son robustas, pueden fallar algunos elementos de procesamiento pero la red continua trabajando, esto es contrario a lo que sucede en programación tradicional.

Las desventajas de las redes neuronales son:

Las redes neuronales se deben entrenar para cada problema. Además, es necesario realizar múltiples pruebas para determinar la arquitectura adecuada. El entrenamiento es largo y puede consumir varias horas de CPU.
Debido a que las redes se entrenan en lugar de programarlas, estas requieren la definición de muchos parámetros antes de poder aplicar la metodología. Por ejemplo hay que decidir la arquitectura más apropiada, el número de capas ocultas, el número de nudos por capa, las interconexiones, la función de transformación, etc. Las técnicas estadísticas convencionales, sin embargo, sólo requieren la extracción y normalización de una muestra de datos. Es un argumento erróneo sostener que el tiempo de desarrollo para los modelos basados en una red neuronal sea más corto que el tiempo necesario para desarrollar, por ejemplo, una tabla de puntuación basada en una regresión múltiple. Los estudios donde se ha constatado un tiempo de desarrollo más corto no han tenido en cuenta la preparación de datos que requiere una red neuronal.
Las redes neuronales presentan un aspecto complejo para un observador externo que desee realizar cambios. Para adicionar nuevo conocimiento, es necesario cambiar las interacciones entre muchas unidades para que su efecto unificado sintetice este conocimiento. Para un problema de tamaño significativo es imposible hacer esto manualmente, por lo tanto una red con representación distribuida debe emplear algún esquema de aprendizaje.

APLICACIONES DE LAS REDES NEURONALES

Las Redes Neuronales deben ser aplicadas en situaciones en las que las técnicas tradicionales han fallado en dar resultados satisfactorios, o cuando una mejora en el modelado puede significar una diferencia en la eficiencia de la operación de un sistema, lo que lleva como consecuencia una mejora en la relación costo-beneficio. El sistema trabajará de mejor manera cuando presente una alta tolerancia al error, pues las Redes Neuronales son aproximadores universales.

Se debe considerar el uso de las Redes Neuronales cuando:

a. El número de variables o la diversidad de los datos sean muy grandes.
b. Las relaciones entre las variables sean vagamente entendibles.
c. La relación entre estas variables sean difíciles de describir adecuadamente mediante los métodos convencionales.
d. Las variables o capturas presenten semejanzas dentro de un conjunto de patrones, tal como sucede en aplicaciones de procesamiento de señales, de control, de reconocimiento de patrones, producción y reconocimiento del habla, en los negocios, en la medicina, etc.

Las características de las redes neuronales hacen que sus posibles aplicaciones sean muy amplias; y,

a la vez, permiten que esta tecnología se vea al alcance de cualquier programador.

Entre el gran número de aplicaciones se encuentran:

Finanzas, ventas, marketing, Manufacturación.
- Predicción de índices.
- Detección de fraudes.
- Identificación de falsificaciones.
- Interpretación de firmas.
- Predicción de la rentabilidad de acciones.
- Campañas de venta.
- Perfilamiento de clientes en función de la compra.

Tratamiento de textos y proceso de formas.
Procesamiento de formas y documentos
Reconocimiento de caracteres impresos mecánicamente.
Reconocimiento de gráficos.
Reconocimiento de caracteres escritos a mano.
Reconocimiento de escritura manual cursiva.
Control de producción en líneas de proceso.
Inspección de la calidad.
Robots automatizados y sistemas de control (visión artificial y sensores de presión, temperatura, gas, etc.)
Energía.
- Predicción consumo eléctrico.
- Distribución recursos hidráulicos para la producción eléctrica.
- Predicción consumo de gas ciudad.

Biología
- Alimentación.
- Medio Ambiente
- Desarrollo postnatal temprano de las corrientes de potasio dependientes de calcio en neuronas de hipocampo.
- Estudio de las modificaciones durante el desarrollo de las corrientes de potasio dependientes de calcio en el hipocampo de conejo
- Bases neurobiológicas de los cuadros convulsivos en la infancia, su modulación por el zinc y simulación computacional.
- Análisis de olor y aroma.

Medicina y salud.
- Aprender más acerca del cerebro y otros sistemas.
- Obtención de modelos de la retina.
- Ayuda al diagnóstico.
- Análisis de Imágenes.
- Desarrollo de medicamentos, predicción de reacciones adversas a los medicamentos.
- Distribución de recursos.
- Analizadores del habla para la ayuda de audición de sordos profundos.
- Diagnóstico y tratamiento a partir de síntomas y/o de datos analíticos (electrocardiograma, encefalograma, análisis sanguíneo, etc.)

Monitorización en cirugía.
Lectores de rayos X.
Entendimiento de la causa de los ataques epilépticos.
Ciencia e Ingeniería.
Análisis de datos y clasificación.

Evaluación de probabilidad de formaciones geológicas y petrolíferas.
Ingeniería Química.
Ingeniería Eléctrica.
Ingeniería Mecánica.
Monitorización de la condición de maquinaria
Modelos meteorológicos.
Transportes y Comunicaciones.
Optimización de rutas.
Optimización de plazas y horarios en líneas de vuelo.
Optimización en la distribución de recursos.
Militares
Análisis de prestaciones de Redes Neuronales aplicadas a la detección de señales sonar.
Clasificación de las señales de radar.
Creación de armas inteligentes.
Optimización del uso de recursos escasos.
Reconocimiento y seguimiento en el tiro al blanco.
Pentágono.

A pesar de estas aplicaciones tomadas como ejemplo, y de muchas otras que se encuentran en desarrollo, sigue sin tenerse de manera clara el concepto de las Redes Neuronales así como de su uso.

Implementación

A la hora de implementar una red neuronal como parte de un programa o sistema informático, se pueden distinguir 3 fases básicas:

Diseño: en esta fase se elige el tipo de red neuronal a usar (la arquitectura o topología), el número de neuronas que la compondrán,...
Entrenamiento: en esta fase se le presentan a la red neuronal una serie de datos de entrada y datos de salida (resultados), para que a partir de ellos pueda aprender.
Uso: se le suministran las entradas pertinentes a la red, y esta genera las salidas en función de lo que ha aprendido en la fase de entrenamiento.

FUNDAMENTOS DE LAS REDES NEURONALES

ESTRUCTURA DEL MODELO BIOLÓGICO

Durante su evolución, el ser humano desarrolló sistemas que le permiten conocer y adecuarse a los cambios en su medio ambiente (externocepción), así como identificar las modificaciones que tienen lugar en su propio organismo (internocepcion). El sistema integrador por excelencia es el nervioso, pues regula a los otros dos sistemas de comunicación, el endocrino y el inmune. Al sistema nervioso, el cual se divide en Sistema Nervioso Central (SNC) y Sistema Nervioso Periférico (SNP), pertenece todo el tejido nervioso del cuerpo. Las propiedades de irritabilidad y conductividad residen en la neurona, su unidad anatómica.

El SNC (Sistema nervioso Central) esta formado por la médula Espinal la cual cumple una función de integración; el cerebro el cual cumple funciones de coordinación y da origen a los nervios craneales que controlan acciones voluntarias.

El SNP (Sistema nervioso periférico) lo conforman millones de neuronas agrupadas en nervios que transmiten los impulsos nerviosos entre el SNC y las demás áreas del cuerpo. Los nervios del SNP son de diferentes tipos; los nervios sensitivos, que llevan sensaciones y estímulos de todo el cuerpo a los centros nerviosos y; los nervios motores, que llevan órdenes de los centros nerviosos a todo el organismo.

Células del sistema nervioso

El sistema nervioso esta constituido por dos tipos celulares principales: las neuronas y las células de sostén. De las neuronas se puede afirmar que son las encargadas de recibir, integrar, transmitir, generar, guardar y modificar los estímulos, para hacer nuestras funciones compatibles con la vida. Estas funciones pueden ser voluntarias o involuntarias y en todas ellas hay participación de neuronas. La complejidad de sus funciones ha hecho que evolutivamente en muchas especies como la humana las neuronas sean incapaces de reproducirse luego del nacimiento. Así mismo son incapaces de nutrirse por si mismas y no poseen mecanismos de defensa muy desarrollados en comparación con otras células del cuerpo.

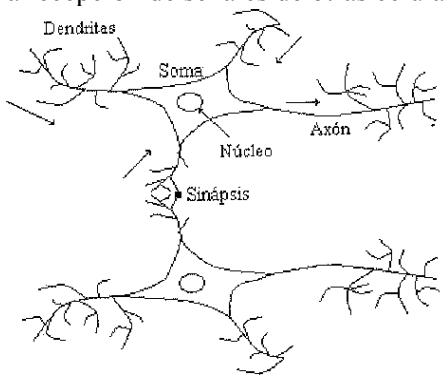
Las células de sostén del sistema nervioso suplen las deficiencias de las neuronas. Han desarrollado un mecanismo de defensa, nutrición y prestan un sustrato físico a las neuronas y por tanto al tejido nervioso. Estas células no poseen la capacidad de transmitir impulsos nerviosos, pero son importantes para la creación de comunicaciones entre las neuronas (sinapsis) y para la transmisión de impulsos nerviosos pues hacen más rápida su conducción y establecen las vías apropiadas. Cuando están localizadas en el SNC, las células de sostén se denominan Neuroglia o células de Glia. Cuando son del SNP se les llama células de Schwann o células satélite.

Neuronas

A finales del siglo XIX se logró una mayor compresión del cerebro debido a los trabajos de Ramón y Cajal en España y Charles Sherrintgton en Inglaterra , investigación y Ciencia (1991). Cajal trabajó en la anatomía de las redes neuronales y demostró que el tejido del cerebro no es una masa continua sino una red de unidades discretas denominadas neuronas . Sherrington realizó investigaciones para entender el funcionamiento de las sinapsis o puntos de conexión entre neuronas.

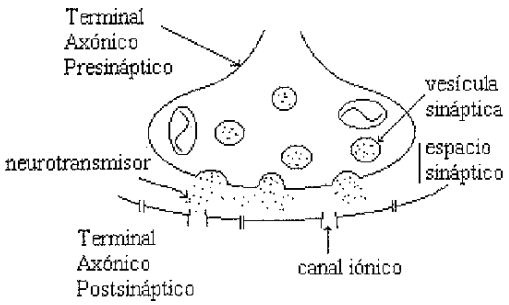
Se estima que hay 26.000 millones de neuronas en el cerebro humano y en un área de un milímetro cuadrado hay aproximadamente 50.000. el tamaño y forma de las neuronas es variable, pero todas poseen las mismas subdivisiones anatómicas.

El soma o cuerpo de la célula contiene el núcleo y es el encargado de las actividades metabólicas de toda neurona, el soma puede recibir información de otras neuronas a través de sinapsis en su superficie. Las dendritas son estructuras que parten del soma con ramificaciones, se especializan en la recepción de señales de otras células nerviosas por medio de conexiones sinápticas.



El axón permite enviar impulsos a otras células nerviosas. En algunas neuronas los impulsos se inician en la unión del axón y el soma, y luego se transmiten a lo largo del axón a otras células nerviosas. Cuando el axón está cerca de sus células destino se divide en muchas ramificaciones que forman sinapsis con el soma o axones de otras células.

La sinapsis es una conexión entre dos células nerviosas. Las sinapsis pueden ser excitativas o inhibitorias según el neurotransmisor que se libere, cada neurona recibe de 10.000 a 100.000 sinapsis y su axón realiza una cantidad similar de sinapsis. La sinapsis se clasifica según su posición en la superficie de la neurona receptora, hay tres tipos : axo-somática, axo-dendrítica, axo-axómatica. La sinapsis es química por naturaleza pero tiene efectos eléctricos laterales que se pueden medir.

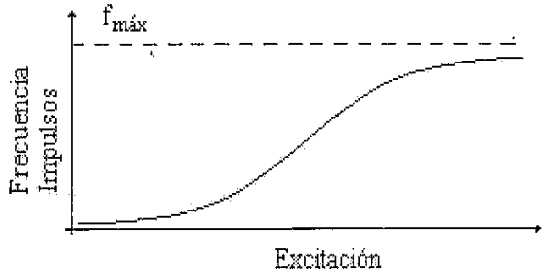


Cuando la neurona esta en reposo, su membrana externa mantiene una diferencia de potencial de -70 mv (la superficie interior es negativa respecto de la superficie exterior). En reposo, la membrana es más permeable a los iones de potasio que a los iones de sodio. Cuando se estimula la célula, la permeabilidad al sodio se incrementa, lo cual produce una entrada repentina de cargas positivas. Esta entrada produce un impulso una inversión momentánea de potencial de la membrana. El impulso se inicia en la unión del cuerpo celular y el axón, y se propaga lejos del cuerpo celular.

Cuando el impulso alcanza los terminales del axón de la neurona presináptica, este induce la

liberación de moléculas neurotransmisoras. Los transmisores se difunden y alcanzan los receptores de la membrana postsináptica.

La frecuencia de los impulsos descargados por una neurona y la intensidad de la excitación obedecen la función:



hay una frecuencia máxima o frecuencia de saturación.

De acuerdo con el número de prolongaciones emitidas, las neuronas pueden ser MULTIPOLARES (dos o más dendritas y un axón), BIPOLARES (una dendrita y un axón en polos opuestos) y SEUDOUNIPOLARES (un axón y una dendrita unidos que se dividen cerca del soma). Hay tres categorías de neuronas desde el punto de vista funcional; las neuronas sensoriales, son las que llevan los impulsos desde los receptores hacia el SNC, las neuronas motoras, son las que transmiten los impulsos desde el SNC hasta las células efectoras: en medio de estos dos tipos hay una red de interneuronas; neuronas de asociación o neuronas centrales, que constituyen la mayoría del cuerpo. Las neuronas motoras y las interneuronas son de tipo morfológico multipolar. Las neuronas sensoriales son de tipo seudounipolar.

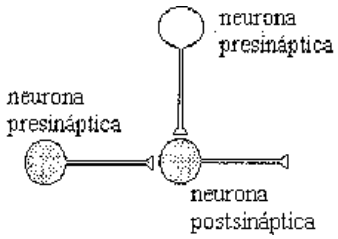
Las formas y tamaños de las neuronas son muy variadas, pero todas comparten cuatro zonas funcionales: una zona receptora o dendrítica aferente, en la que se integran los cambios de despolarización locales-excitatorios-o-inhibitorios que se inician en las uniones sinápticas; una zona en que se generan potenciales de descarga que corresponde al lugar de unión del axón con el soma; una prolongación axónica eferente que transmite los impulsos hasta las terminaciones nerviosas, última porción en la que los potenciales de despolarización hacen que se liberen sustancias neurotransmisores en la sinapsis.

En general, la comunicación entre neuronas o sinapsis se da por contacto físico en organismos muy poco evolucionados. En los mamíferos como el ser humano, la propagación en el espacio sináptico, es a través de mensajeros químicos liberados de las terminales de los axones o de las dendritas, que se encuentren próximas a otras neuronas. Cada axón o dendrita posee una prolongación terminal conocida como botón sináptico, en donde se depositan las sustancias que sirven de mensajeros, los NEUROTRANSMISORES. La recepción de la información se da por la existencia de receptores específicos para cada neurotransmisor sobre la membrana de la célula postsináptica, los cuales forman un complejo similar a la de una llave en una herradura con el neurotransmisor especifico. Los receptores generalmente se encuentran en las prolongaciones de las neuronas (axones o dendritas) aunque se pueden encontrar sobre el soma neuronal.

Aprendizaje en la neurona biológica

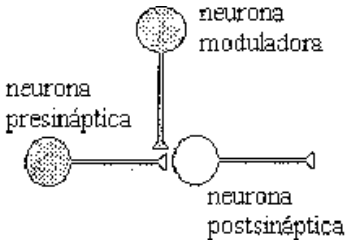
El psicólogo canadiense Donals o. Hebb propone que el aprendizaje en las neuronas ocurre por la actividad neuronal coincidente, esto se conoce como la ley de Hebb o aprendizaje Hebbiano:

“cuando el axón de la célula A excita la célula B y repentinamente toma lugar en su activación, ocurre algún proceso de crecimiento o cambio metabólico en una o ambas células tal que la eficacia de A, como una de las células que dispara a B, se incrementa”. Según la regla de aprendizaje de Hebb, la actividad coincidente en las neuronas presináptica y postsináptica es critica para fortalecer la conexión entre ellas, esto se denomina mecanismo asociativo pre-post.



Ladislav Tauv y Eric Kandel propusieron en 1963 una segunda regla de aprendizaje mientras trabajan en el Instituto Marey en París en el sistema nervioso del caracol APLYSIA. Ellos encontraron que la conexión sináptica entre dos neuronas se puede fortalecer sin actividad de la célula postsináptica.

La tercera neurona llamada neurona moduladora, incrementa la liberación del neurotransmisor de la célula presináptica



Las neuronas y las conexiones entre ellas (sinapsis) constituyen la clave para el procesamiento de la información.

FUNDAMENTOS MATEMÁTICOS DE LAS REDES NEURONALES ARTIFICIALES (RNA, ANN ó ANS)

El cerebro humano, compuesto por neuronas, tiene muchas características deseables por cualquier sistema artificial: No necesita programación; con el tiempo aprende, puede manejar información ambigua, múltiple e inconsistente, es seguro y robusto frente a las fallas, la muerte de células nerviosas no afecta de forma inmediata su desempeño.

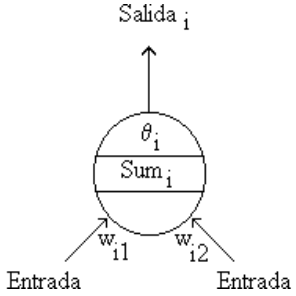
Una red neuronal artificial, "intenta ser" la representación matemática de una red neuronal biológica. Decimos que intenta ser, porque dada la complejidad todavía no resuelta del funcionamiento del cerebro, todos los trabajos hasta ahora desarrollados son muy burdos en comparación de lo que realmente es, esto es en gran parte debido a las limitantes tecnológicas actuales.

Las idea básica de las redes neuronales artificiales, consiste en simplificar la información de entrada, y tomando los elementos más relevantes de la misma, obtener una respuesta.

En el modelo matemático, las neuronas están representadas como elementos procesadores (EP), las rutas de entrada están definidas como las interconexiones, los procesos combinan las señales y la salida generada es tratada por una función de transferencia de naturaleza no lineal. La fuerza sináptica de cada conexión está representada como un peso y el cambio en la fuerza sináptica lo definimos como el aprendizaje de la red.
Elementos procesadores

Un elemento procesador (EP), es un dispositivo simple, cuyo funcionamiento trata de asemejar de una manera muy burda al funcionamiento de una neurona biológica. Aunque los detalles biológicos no pueden ser representados por un modelo de computadora o electrónico (por lo menos hasta nuestros días) la estructura de los EP la podemos tomar para simular o tratar de entender la manera en que trabaja una neurona biológica.

En los modelos de redes neuronales artificiales los EP realizan diversas funciones tales como: la evaluación de las señales de entrada, la suma de las mismas, así como la comparación de dicha suma con una valor de umbral determinado que fija el valor de la salida.



La implementación de una neurona puede ser en un microprocesador de 1 bit, en un procesador con punto flotante o por medio de un amplificador operacional. Dicha implementación depende del modelo de red seleccionado y de la aplicación en cuestión.

Uj : Denota una EP y lleva un subíndice respectivo para diferenciarla de las demás.

Cada EP puede recibir varias señales de entrada simultáneamente, pero sólo presenta una señal de salida, la cual puede ser positiva o negativa, 0 ó 1 o en general entre un pequeño rango, dicha salida depende de las señales de entrada de los pesos y del umbral para cada EP.

Pesos o sinapsis

Es una magnitud que indica la fuerza que tiene la conexión entre dos neuronas o elementos procesadores, dicha magnitud funciona como un factor determinante para definir la excitación o inhibición de la neurona. Algunas entradas pueden ser más importantes que otras, esto se debe al peso correspondiente a cada entrada en el EP. Estos pesos son la fuerza de interconexión entre los EP y normalmente son representados en términos de vectores del tipo:

$$peso_i = (peso_{i1}, \ldots, peso_{in})$$

$\zeta_{j\,i}$: Peso (Sinapsis) o valor de la información, que sale de la neurona i, y a través de un canal unidireccional donde se modifica, entra a la neurona j.
Estado de activación

a_i (t) : Estado de activación, es un valor numérico característico de cada neurona, el cual nos informa el comportamiento o estado de la unidad i en un tiempo t.

Función de transferencia

La función de transferencia es usada para convertir el valor de activación de cada EP en el valor de salida.

f_i : Función de salida o de transferencia, relaciona o convierte ai (t) con/en una señal de salida, puede ser una de cuatro tipos:

Escalón; Asociada a neuronas binarias, en redes de capacidades limitadas.
Lineal y mixta; Función más sencilla para neuronas análogas también bastantes limitadas, en cuanto a sus capacidades.
Continua (Sigmoidal); Para redes análogas, con la ventaja de que se pueden utilizar las reglas de aprendizaje definidas para la función escalón, que en muchas ocasiones utilizan derivadas, esto implica una ventaja, ya que la derivada está definida para toda la función
Gaussiana; Más adaptativas que las funciones sigmoidales, puede simplificar dos capas ocultas con función de transferencia sigmoidal en una sola.

Salida

yi (t): Es la salida de la neurona en el tiempo t, definida por la función de transferencia, de esta manera:

$$y_i(t) = f_i(a_i(t))$$

Entrada neta

Netj: Denota la entrada neta o total a la unidad de proceso, también es función del tiempo y se define como:

$$Net_j = \sum_i y_i \cdot w_{ji}$$

Función de activación

F: Función de activación determina el nuevo estado de activación a_i (t + 1)que depende de Net_j, y de a_i (t).

$$a_j(t+1) = F(Net_j, a_j(t))$$

Puede tener una de cuatro representaciones o formas posibles:

Escalón.

$$y_i(t+1) = \begin{cases} 1 si[Net_i > \theta_i] \\ y(t) si[Net_i = \theta_i] \\ 0 o -1 si[Net_i < \theta_i] \end{cases}$$

Lineal o identidad.

$$y_i(t+1) = Net_i - \theta_i^t$$

Lineal - mixta.

$$y_i(t+1) = \begin{cases} b \ldots Si[Net_i \leq b + \theta_i^t] \\ Net_i - \theta_i^t \ldots Si[b + \theta_i^t < Net_i < B + \theta_i^t] \\ B \ldots Si[Net_i \geq B + \theta_i^t] \end{cases}$$

Sigmoidal.

$$y_i(t+1) = \frac{1}{1 + e^{-(Net_i - \theta_i)}}$$

También se define un vector estado de activación, de N números reales, donde N es el número de neuronas que componen la red, y se denota como:

$$A(t) = (a_1(t), a_2(t), \ldots, a_i(t), \ldots, a_N(t))$$

Existe otro vector, que contiene la información acerca de las salidas de todas las neuronas de la red en un instante t:

$$Y(t) = (f_1(a_1(t)), f_2(a_2(t)), \ldots, f_i(a_i(t)), \ldots, f_N(a_N(t)))$$

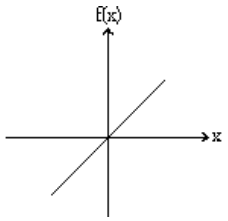
La elección de la función de activación depende de la forma en que se desee sean representados los datos de salida, por ejemplo, si se desea que las unidades de salida sean binarias, se utiliza la función de activación sigmoidal en la que su primer derivada es continua.

El funcionamiento de una RNA, está basado en la evolución dinámica de la red mediante la cual se van actualizando los estados de las neuronas y los pesos de la red hasta alcanzar un estado deseado, este proceso, se conoce como entrenamiento, y puede ser de dos maneras:

Asíncrono: Las neuronas evalúan su estado de manera continúa, e independiente.
Síncrono: También en forma continua pero simultanea.
Otras concepciones para las representaciones o formas posibles de la función de activación son:

Las funciones de activación mapean, escalan o limitan el dominio de los EP en su entrada a un rango específico a la salida del mismo. Las funciones de activación comúnmente usadas son:

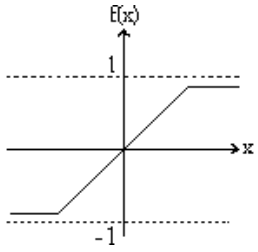
La función lineal.



$$f(x) = \alpha x$$

donde a es una constante en el dominio de los números reales la cual indica la ganancia en la actividad del EP "x".

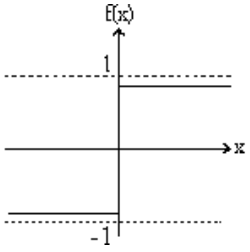
b) La función rampa.



$$f(x) = \begin{cases} +\gamma & x \geq \gamma \\ x & |x| < \gamma \\ -\gamma & x \leq -\gamma \end{cases}$$

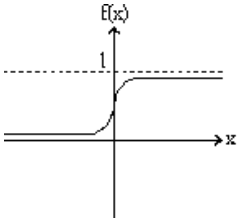
Cuando la función lineal es limitada al rango [- g , + g], se convierte en la función rampa, donde g (- g) es el valor máximo y mínimo respectivamente, que puede tomar el EP, esos valores son los que se refieren al nivel de saturación de la señal.

La función escalón.



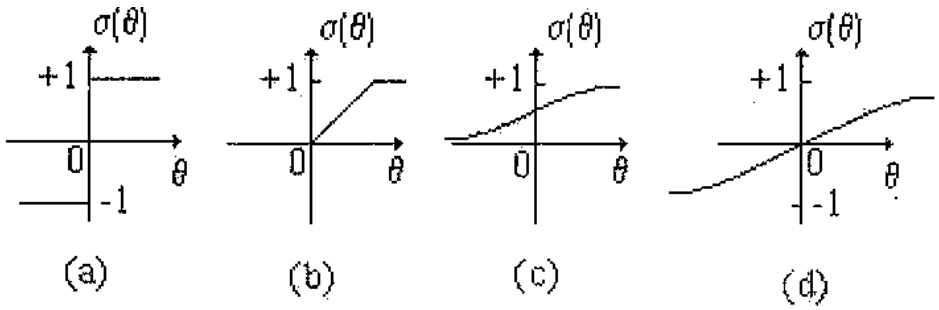
$$f(x) = \begin{cases} +\gamma & x \geq 0 \\ -\delta & x \leq \text{cualquier otro} \end{cases}$$

Si la función de activación, solamente responde a la señal de entrada, entregando + g si la entrada es positiva y entregando - g si es negativa, entonces la función de activación es llamada función escalón, donde d y g son escalares positivos.
d) La función sigmoide.



$$f(x) = (1 + e^{-x})^{-1}$$

La función de activación más comúnmente usada, es la función sigmoide, esta función con forma de S, es acotada y no decreciente, la cual provee una respuesta no lineal.



Función de activación. (a) Limitador duro ; (b) Umbral lógico ;
(c) Función logística ; (d) Tangente jiperbólica.

TOPOLOGÍA

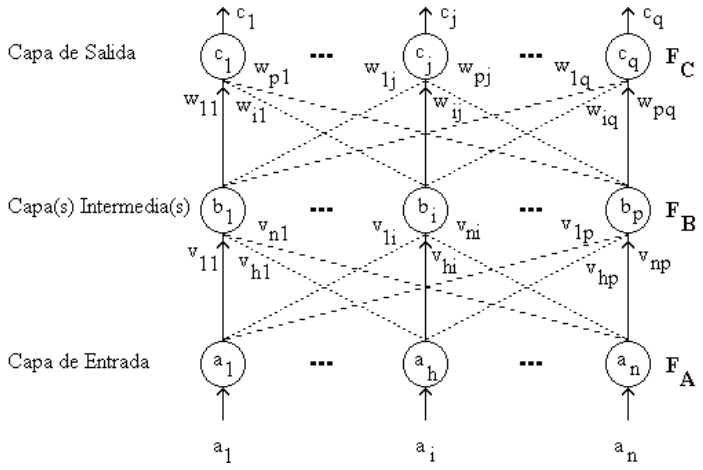
La arquitectura empleada para modelar una red neuronal, define la configuración para cada uno de los elementos básicos, en el que el paradigma de las redes neuronales artificiales está caracterizado en cómo los algoritmos de aprendizaje determinan el cambio de los pesos.

Algunos modelos de red empleados tienen una entrada extra, la cual es denominada "bias" cuyo único objetivo es lograr una convergencia más rápida de la red, aunque el empleo de este término dependerá de la aplicación.

La organización de la RNA en cuanto a su arquitectura consiste en la disposición de las neuronas en forma de capas, de manera que las neuronas de una capa están conectadas con las neuronas de la capa siguiente, a las que pueden enviar información. Se agrupan en tres tipos:

EP de la capa de entrada: Sólo reciben las señales de entrada desde fuentes externas al sistema y sin procesarlas transmiten la señal a las capas siguientes.
EP de la capa de salida: Mandan las señales que reciben hacia afuera del sistema. En sí, son las encargadas de entregar la respuesta de la red y en muchos casos del control de los dispositivos que se conecten a su salida.
EP de la capa intermedia u oculta: No tienen contacto con el exterior de la red, puede no haber niveles ocultos, son las principales encargadas del proceso de representación interna de la información.

En conjunto, todos los EP presentan el siguiente esquema :



Cuya nomenclatura es la siguiente:

FA, FB, FC Capa de Entrada, Intermedia y de Salida respectivamente
a1 ... an Señales de entrada
c1 ... cq Señales de salida
a1 ... an Elementos procesadores de la capa de entrada
b1 ... bp Elementos procesadores de la capa intermedia
c1 ... cq Elementos procesadores de la capa de salida
vnp Indica los pesos asociados entre las neuronas n, p correspondientes a las capas de entrada e intermedia
wpq Indica los pesos asociados entre las neuronas p, q correspondientes a las capas de entrada e intermedia

El número de capas intermedias y el número de neuronas de cada capa dependerá del tipo de aplicación al que se vaya a destinar la red neuronal.

Las conexiones, son las uniones entre una o más neuronas, las cuales indican la relación existente entre ambas y tiene que ver con la definición de los pesos.

El esquema de interconexión es lo que define a la arquitectura de una red neuronal artificial, es el que indica como se propagará la señal desde un EP a otro o hacia sí mismo. Dichas interconexiones son unidireccionales y tienen un peso asociado para cada conexión, estos pesos forman la memoria de la red.
Las conexiones excitatorias, usualmente señales positivas o cercanas a 1, aumentan el valor de entrada al EP, mientras que las conexiones inhibitorias decrementan el valor.

Existen tres diferentes esquemas de interconexión entre los EP en una red neuronal

Conexiones entre EP de la misma capa.
Conexiones entre EP de diferente capa.
Conexiones recurrentes que conectan a un EP consigo mismo.

Existe una clasificación para los tipos de conexiones existentes entre las distintas neuronas y capas que conforman la red:

Hacia delante o feedforward , si la información fluye en una dirección.
Hacia atrás o feedback, que pueden ser recurrentes o autorrecurrentes.
Laterales.

En cuanto al número de capas, las RNA se pueden clasificar en:

Redes Monocapa: Con conexiones laterales o autorecurrentes, e implementadas para su utilización.
Redes Multicapa: Cada capa de una RNA se distingue por que esta conformada por el conjunto de neuronas que comparten el origen de las señales de entrada y el destino de las señales de salida, de acuerdo al tipo de conexión tendremos entonces redes multicapa:
Feedfoward: Generalmente, salvo algunas excepciones, las señales se propagan hacia delante, sin conexiones hacia atrás, ni laterales, ni autorrecurrentes. Ampliamente utilizadas en reconocimiento o clasificación de patrones.
Feedforward/feedback: En general, suelen ser bicapa, estructura particularmente adecuada y generalmente utilizada en lo que se conoce como heteroasociación. Dentro de este grupo de RNA encontraremos algunas con conexiones laterales, y otras con conexiones autorrecurrentes. También existe un tipo particular de red denominada NEOCOGNITRON topológicamente organizada en capas bidimensionales, lo que permite eliminar las variaciones geométricas o distorsiones en la información de entrada de la red.

MECANISMOS DE APRENDIZAJE

Es el proceso mediante el cual una red neuronal modifica sus pesos (destruye, modifica o crea conexiones) en respuesta a una información de entrada. El valor de cero para una conexión en la red, implica que esa conexión no existe. El proceso de aprendizaje finaliza, cuando los valores de los pesos dejan de cambiar, entonces decimos que la red ha aprendido.

Los criterios que definen las técnicas mediante las cuales se han de modificar los pesos de las conexiones de la red son propios de cada red, y definen lo que se conoce como regla de aprendizaje. Podemos encontrar redes que aprenden durante su funcionamiento, estos es aprendizaje ON LINE, y redes que aprenden desconectadas, lo que se conoce como aprendizaje OFF LINE.

De acuerdo al mecanismo de aprendizaje utilizado se clasifica la regla de aprendizaje, y por tanto la red de esta manera se puede distinguir entre aprendizaje supervisado y no supervisado.

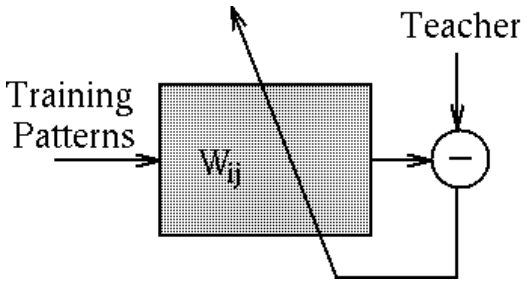
El tipo de aprendizaje se debe a que nosotros, dependiendo de la arquitectura de red empleada, podemos considerar un proceso que force a una red a entregar una respuesta dada ante una entrada específica. Una respuesta particular puede o no ser especificada.

Supervisado
En este caso, el proceso de aprendizaje se realiza mediante un entrenamiento controlado por un agente externo (supervisor), quien modifica los pesos de las conexiones, mientras la salida del conjunto difiera de la deseada para una entrada dada, buscando aproximarse a la misma. Han sido los modelos de redes más desarrolladas desde inicios de estos diseños.

Dado un nuevo patrón de entrenamiento, por ejemplo, (m+1)-ésimo, los pesos serán adaptados de la siguiente forma:

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

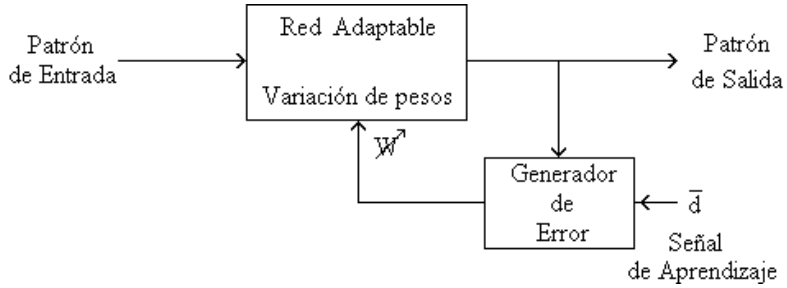
Se puede ver un diagrama esquemático de un sistema de entrenamiento supervisado en la siguiente figura:



Corrección de error
Consiste en variar los pesos de las conexiones de la red en función del error cometido a la salida. Se definen entonces, unas reglas y algoritmos matemáticos para tal fin, los cuales, resultaran más ventajosos en la medida que nos brinden mayor información, acerca del error global a la salida de la red, y se discriminen los errores cometido por cada unidad de proceso individualmente.
Es generalmente empleado en redes de propagación hacia adelante, en las que los patrones de entrenamiento están compuestos de dos partes fundamentales, un vector de entrada y un vector de salida, asociando la entrada con su correspondiente salida en cada elemento procesador denominado nodo.

La manera general en que trabaja este tipo de aprendizaje es la siguiente:

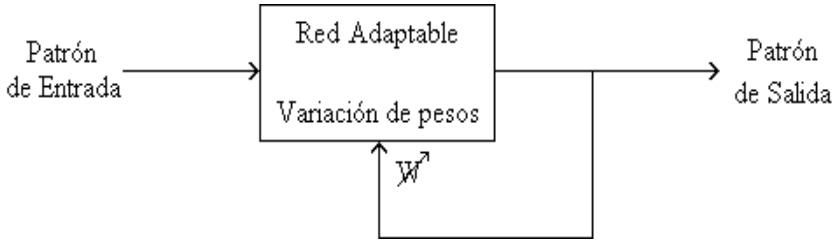
Un vector de entrada es presentado en la capa de entrada, junto con un conjunto de respuestas deseadas para ese vector $\bar{\mathbf{d}}$. Al realizar una iteración, se genera el error o discrepancia entre la respuesta deseada y la obtenida para cada nodo en la capa de salida, dicho valor de error es utilizado para actualizar el vector de pesos y, se realimenta para producir una nueva salida. Los pares de vectores de entrenamiento se van modificando hasta que el porcentaje de cambio de los pesos esté cercano a 0 indicando en realidad que el error total está próximo a cero. No es necesario llegar hasta un valor de cero en el error; dependiendo de la aplicación, puede estar dentro de un rango dado. Este tipo de aprendizaje es utilizado en arquitecturas de redes neuronales artificiales que necesitan ser entrenadas fuera de línea, es decir, que se necesitan entrenar antes de poder aplicarlas.



Aprendizaje por refuerzo
Más lento que el de corrección de error, no se pretende exactitud en la salida del sistema, de modo que la función del supervisor se reduce a excitar o inhibir la actividad neuronal, mediante una señal de refuerzo que indica si la señal obtenida a la salida de la red se ajusta a la deseada (éxito = 1, fracaso = -1), y en función de ello se ajustan los pesos en base a un mecanismo de probabilidades.

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. Comúnmente se hace analogías con el estado energético de un sólido, si la energía del sistema es menor después de un cambio, se acepta el cambio, en caso contrario la aceptación del cambio queda condicionada a una distribución de probabilidades determinada y preestablecida. Se utiliza un procedimiento conocido como simulated annealing consistente, en utilizar ruido para escapar de los errores locales, facilitando la búsqueda del error global. Este ruido va siendo decrecido durante el proceso de aprendizaje. La combinación de la asignación probabilística, con el simulated annealing, es lo que se conoce como aprendizaje estocástico.

No supervisado o autosupervisado
La red no requiere influencia externa para ajustar los pesos de las conexiones. La utilidad de estas redes, se reduce a encontrar las características, regularidades, correlaciones o categorías que se pueden establecer entre los datos que se presenten a su entrada. La interpretación de la salida de una red de este tipo, depende de su estructura, y del algoritmo de aprendizaje empleado. La red aprende a adaptarse basada en las experiencias recogidas de los patrones de entrenamiento anteriores, sin el beneficio de un maestro.. El siguiente es un esquema típico de un sistema "No Supervisado":



Hebbiano
Utilizado cuando se pretende medir la familiaridad, o extraer características de los datos de entrada, se basa en el siguiente postulado formulado por Donald O. Hebb en 1949: Cuando un axón de una celda A está suficientemente cerca como para conseguir excitar una celda B, y repetida o persistentemente toma parte en su activación, algún proceso de crecimiento o cambio metabólico tiene lugar en una o ambas celdas, de tal forma que la eficiencia de A, cuando la celda a activar es B aumenta. Resumiendo, el ajuste de los pesos se lleva a cabo de acuerdo con la correlación de los valores de las salidas de las dos neuronas conectadas.

Competitivo y cooperativo
El trabajo de la red se orienta hacia la clusterización o clasificación de los datos de entrada, se busca que el número de neuronas que se activen para una determinada entrada, sea menor que el número de neuronas de dicha entrada. De esta manera las neuronas compiten por activarse, quedando sólo una, o una por cada cierto grupo como vencedora, dejando al resto forzadas a sus valores de respuesta mínimos.

Algoritmos de aprendizaje

Algoritmos formados por un conjunto de reglas que permiten a la red neuronal aprender (a partir de los datos que se le suministran), mediante la modificación de los pesos sinápticos de las conexiones entre las neuronas (el umbral de cada neurona se modificará como si fuera un peso sináptico más). Generalmente los datos que se usan para entrenar la red se le suministran de manera aleatoria y secuencial.

Manejo de la información de entrada y salida

Forma de asociación IN/OUT
La información que aprende la red se almacena en forma distribuida en los pesos asociados a las conexiones entre neuronas. Desde este punto de vista se comporta como un cierto tipo de memoria, que al recibir una determinada información de entrada, entrega una respuesta asociada.

Heteroasociativas
Las redes de este tipo, deben asociar informaciones de entrada, con distintas 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. En algunos casos el objetivo de este tipo de redes, es computar una función general de su entrada. En otros casos el objetivo es realizar una clasificación, relacionando (mapeando) un gran número de informaciones de entrada con un pequeño número de informaciones de salida, que representan los conjuntos en los que se pueden clasificar los datos de entrada.

Autoasociativas
Este tipo de redes asocia una información de entrada con el ejemplar más parecido de los almacenados (conocidos) por la red, y pueden implementarse con una sola capa de neuronas. Son comúnmente utilizadas en tareas de filtrado de información para la reconstrucción de datos, eliminando distorsiones o ruido. También se utilizan para explorar relaciones entre informaciones similares, para facilitar la búsqueda por contenido en bases de datos y para resolver problemas de optimización.

Representación de la información IN / OUT
Los datos manipulados por la RNA pueden ser de naturaleza binaria (digital o discreta), o análoga (continua). A partir de esta consideración se puede clasificar las RNA como:

Redes Continuas: Tanto la información de entrada como la de salida son de naturaleza analógica.
Redes Híbridas: Con información de entrada analógica, e información de salida binaria.
Redes Discretas: Donde ambas, la información de entrada y de salida, son binarias.

Funciones separables linealmente
Los problemas de clasificación de patrones que se pueden resolver con una línea recta o con un hiperplano se denominan problemas separables linealmente.

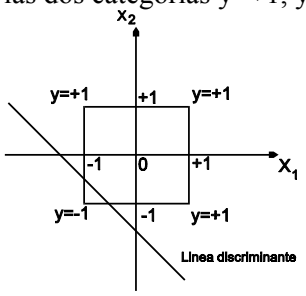
El combinador lineal (neurona sin función de activación) permite resolver problemas de clasificación separables linealmente. La función de activación limitador duro produce dos categorías con salidas y=+1 o y=-1, esto es, divide el espacio de patrones de entrada en dos áreas.

Ejemplo. Función lógica OR

X	X_2	Y
-1	-1	-1
-1	+1	+1
+1	-1	+1
+1	+1	+1

Se gráfica la función (4 patrones son 4 puntos en el espacio (x_1-x_2). Una línea recta puede separar

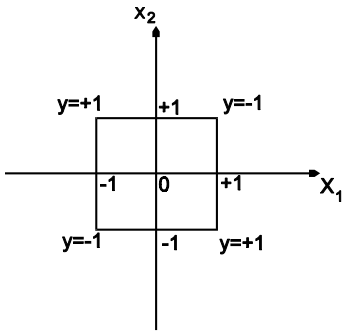
las dos categorías y=+1, y=-1.



Se concluye que la función lógica OR es separable linealmente y por lo tanto una neurona resuelve el problema.

Ejemplo. Función XOR

X	X_2	Y
-1	-1	-1
-1	+1	+1
+1	-1	+1
+1	+1	+1



Una línea recta NO puede separar las dos clases y=+1, y=-1.

Se concluye que la función lógica XOR no es separable linealmente y por lo tanto una neurona no resuelve el problema.

En el libro Perceptrons (1969), Minsky y Papert, mostraron que las redes neuronales de una capa sólo pueden resolver problemas separables linealmente, es decir, estas redes no pueden sintetizar funciones tipo XOR. Esta limitación de las redes neuronales de una capa causó que la investigación en redes neuronales entrara en un período de hibernación científica de aproximadamente 20 años.

ALGORITMOS DE ENTRENAMIENTO PARA UNA NEURONA

Hay dos grandes categorías de algoritmos de entrenamiento para una neurona o una red neuronal. Los algoritmos supervisados requieren de patrones entrada-salida deseada, esto es, la red necesita un profesor que le muestre las respuestas correctas. La segunda categoría se denomina algoritmos no supervisados y la red neuronal establece su propia organización de los datos de entrada.

Los algoritmos de aprendizaje supervisado se pueden dividir en dos tipos; corrección de error y gradiente.

Corrección de error
Altera los pesos de la neurona después de la presentación de un patrón para corregir el error de la salida.

Gradiente
Modifica los pesos de la neurona después de la presentación de un patrón para minimizar el error cuadrático medio sobre todos los patrones. Esto se logra modificando los pesos en dirección opuesta al gradiente de la función de error.

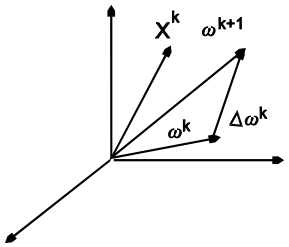
ALGORITMOS DE CORRECCION DE ERROR

Se tienen dos algoritmos:

Alfa – Least Square (α-LMS)
Regla del perceptron

(α-LMS)

Esta regla se utiliza para la neurona sin función de activación y modifica el vector de pesos en la dirección del patrón de entrada.



El vector de pesos actual ω^k se modifica en dirección del patrón de entrada x^k para producir un nuevo vector de pesos ω^{k+1}. En lenguaje matemático,

$$\omega^{k+1} = \omega^k + \alpha . e^k . \frac{x^k}{\left| x^k \right|^2} .$$

K=1,2,…, n (número de iteraciones)

El vector x^k se normaliza al dividirlo por su norma al cuadrado. La ganancia α determina la importancia de cada nuevo vector x^k, α también se conoce como rata de aceleración o rata de aprendizaje. El error e^k entre la salida deseada y la salida de la neurona indica la magnitud. El error e^k entre la salida deseada y la salida actual de la neurona indica la magnitud y dirección de la corrección.

Calculando el error en la iteracción k-ésima $e^k = d^k - (\omega^k)^T .x^k$
este error es para la el combinador lineal, esto es, no hay función de activación en la neurona. Perturbando el error, $\Delta e^k = -(x^k)^T .\Delta\omega^k$
pero según la formula de entrenamiento

$$\Delta\omega^k = \omega^{k+1} - \omega^k = \alpha.e^k.\frac{x^k}{\left|x^k\right|^2}$$

y reemplazando $\Delta e^k = -\alpha.e^k$ el error se reduce proporcionalmente al factor de aceleración, entonces el error decae exponencialmente durante el entrenamiento y el parámetro α determina la rata de decrecimiento.

Se debe recordar que: $(x^k)^T .x^k = \left|x^k\right|^2$

La velocidad de convergencia del algoritmo α-LMS depende de la correlación existente entre loa Patrones de entrada, a mayor correlación menor velocidad. Cuando los patrones están fuertemente correlacionados, el combinador lineal maneja información redundante y su aprendizaje es lento.

Regla del perceptrón

Modelo desarrollado en 1958 por Rosemblatt; ésta red fue el primer modelo Red Neuronal Artificial, capaz de reconocer patrones sencillos y/o clasificar dicho patrón en dos clases (A o B).A pesar de que se estudiaron varias configuraciones, la única con una regla efectiva de aprendizaje en aquel tiempo fue la de una sola capa.

En la actualidad, el perceptron está formada por dos capas:

Capa de Entrada. Compuesta por una o varias neuronas lineales; esto es, no existen conexiones laterales ni autorrecurrentes, las cuales reciben el patrón de entrada.
Capa de Salida del perceptrón. Conformada por una única neurona, la cual realiza la suma ponderada de las entradas y resta el Umbral para obtener una salida.

La salida se pasa a una función de Transferencia en este caso la función Escalón o función umbral.

Si el patrón pertenece a la clase A responderá con +1, si por el contrario el patrón pertenece a la clase B responderá con -1. Luego la salida dependerá de la suma neta de las entradas s^k y del valor de umbral q.

$$Y^K = \sigma(S^K)$$

Una vez obtenida una salida total o salida de la red, podemos establecer el criterio de modificación de los pesos a través de la siguiente ecuación:

Error = Salida deseada— Salida obtenida $e^k = d^k - y^k$

con el cual procedemos modificar los pesos de la red: $\omega^{k+1} = \omega^k + \alpha.e^k .x^k$

En esta ecuación α es un factor de "ganancia" denominado parámetro de aprendizaje, que se encuentra en el intervalo [0.0 , 1.0] cuya misión es dar rapidez y/o estabilidad a la red en su proceso de aprendizaje.

Sin embargo, cuando se requiere hacer una clasificación donde se presenten varias clases, el Perceptrón (monocapa) se ve imposibilitado; un ejemplo de esta limitación es la función EX-OR, debido a su no- linealidad.

Regla del perceptrón modificada

Una versión mejorada de la regla del perceptrón es:

$$\omega^{k+1} = \omega^k + \alpha.e^k.x^k + \mu.(\omega^k - \omega^{k-1})$$

El término adicional se denomina momentum, es una memoria o inercia y permite que los cambios en el vector de pesos ω sean suaves porque incluyen información sobre el cambio anterior

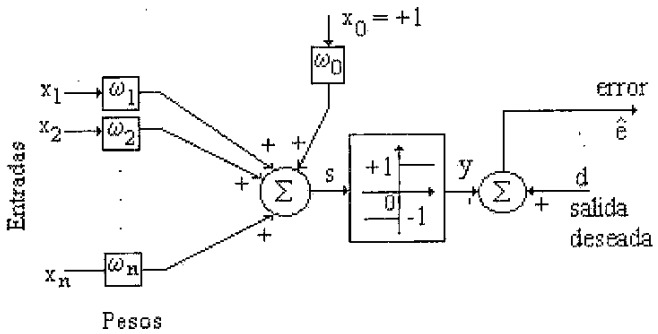
Ejemplo. Entrenar una neurona con función de activación limitador duro para aprender la función lógica OR. Utilizar el algoritmo de corrección de error α-LMS y tomar.

$$\left[\omega^1\right]^T = [0.2\ 0.1\ 0.1] \qquad \alpha = 0.1$$

Solución: La tabla de la función lógica OR con valores bipolares es:

X	X_2	Y
-1	-1	-1
-1	+1	+1
+1	-1	+1
+1	+1	+1

Se utiliza valores bipolares {-1,1} en lugar de {0,1} para acelerar el aprendizaje.



La ecuación para actualizar los pesos es,

$$\omega^{k+1} = \omega^k + \alpha.e^k.\frac{x^k}{\left|x^k\right|^2}$$

donde

ω^k : vector de pesos de la iteración k

x^k : patrón de entrada en la iteración k.

La suma en la neurona para la iteracción k-ésima es

$$s^k = (1)(\omega_0^k) + (x_1^k)(\omega_1^k) + (x_2^k).(\omega_2^k)$$

si la suma s^k es mayor o igual a cero entonces la salida es positiva $y^k = +1$ en caso contrario la salida es $y^k = -1$

Iteración 1, k=1

Para las entradas $x_1^1 = -1, x_2^1 = -1$, la salida deseada es $y_d^1 = -1$. Reemplazando en la suma :

$s^1 = 0.0$ por lo tanto $y^1 = +1$

La función de activación pasa por el origen.

El error está dado por $e^1 = y_d^1 - y^1 = (-1) - (+1) = -2$, el nuevo vector de pesos está dado por

$$\omega^2 = \omega_1 + \alpha.e^1.\frac{x^1}{\left|x^1\right|^2}$$

reemplazando los valores numéricos

$$\begin{bmatrix} \omega_0^2 \\ \omega_1^2 \\ \omega_2^2 \end{bmatrix} = \begin{vmatrix} 0.2 \\ 0.1 \\ 0.1 \end{vmatrix} + (0.1).(-2).\frac{1}{\sqrt{(1.0)^2 + (-1.0)^2 + (-1.0)^2}}.\begin{bmatrix} 1.0 \\ -1.0 \\ -1.0 \end{bmatrix}$$

realizando operaciones

$$\begin{bmatrix} \omega_0^2 \\ \omega_1^2 \\ \omega_2^2 \end{bmatrix} = \begin{vmatrix} 0.2 \\ 0.1 \\ 0.1 \end{vmatrix} + \frac{(0.1).(-2)}{3.0}.\begin{bmatrix} 1.0 \\ -1.0 \\ -1.0 \end{bmatrix}$$

se tiene

$$\begin{bmatrix} \omega_0^2 \\ \omega_1^2 \\ \omega_2^2 \end{bmatrix} = \begin{vmatrix} 0.2 \\ 0.1 \\ 0.1 \end{vmatrix} - 0.0667. \begin{bmatrix} 1.0 \\ -1.0 \\ -1.0 \end{bmatrix}$$

finalmente,

$$\begin{bmatrix} \omega_0^2 \\ \omega_1^2 \\ \omega_2^2 \end{bmatrix} = \begin{vmatrix} 0.1333 \\ 0.1667 \\ 0.1667 \end{vmatrix}$$

Iteración 2, k=2

Para las entradas $x_1^2 = -1, x_2^2 = +1$, la salida deseada es $y_d^2 = +1$. Reemplazando en la suma $s^2 = 0.1333$ por lo tanto $y^2 = +1$

El error está dado por $e^2 = y_d^2 - y^2 = (+1) - (+1) = 0$ el nuevo vector de pesos es igual al anterior

$$\omega^3 = \omega^2$$

Iteración 3, k=3

Para las entradas $x_1^3 = +1, x_2^3 = -1$, la salida deseada es $y_d^3 = +1$. Reemplazando en la suma $s^3 = 0.1333$ por lo tanto $y^3 = +1$

El error esta dado por

$e^3 = y_d^3 - y^3 = (+1) - (+1) = 0$,

el nuevo vector de pesos es igual al anterior porque el error es cero $\omega^4 = \omega^3$

Iteración 4, k=4

Para las entradas $x_1^4 = +1, x_2^4 = +1$, la salida deseada es $y_d^4 = +1$. Reemplazando en la suma

$s^4 = 0.4667$ por lo tanto $y^4 = +1$

El error esta dado por

$e^4 = y_d^4 - y^4 = (+1) - (+1) = 0$

el nuevo vector de pesos es igual al anterior porque el error es cero para este patrón de entrada

$$\omega^5 = \omega^4 = \begin{bmatrix} 0.1333 \\ 0.1667 \\ 0.1667 \end{bmatrix}$$

en este punto se ha finalizado el recorrido de la tabla lógica de la función OR. Es necesario revisar de nuevo si el primer patón se satisface para el vector de pesos ω^5

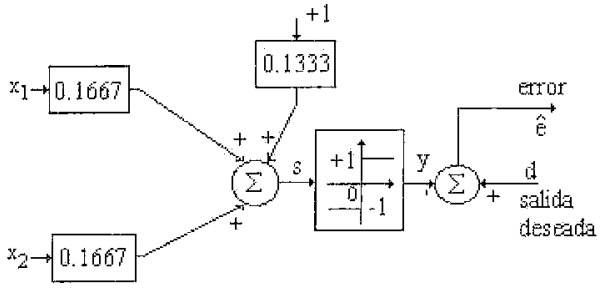
Iteración 5, k=5

Para las entradas $x_1^5 = -1, x_2^5 = -1$, la salida deseada es $y_d^5 = -1$. Reemplazando en la suma $s^5 = -0.2001$ por lo tanto $y^5 = -1$

El error esta dado por

$$e^5 = y_d^5 - y^5 = (-1) - (-1) = 0$$

Como el error es cero significa que el vector de pesos asociado a la neurona satisface la tabla de la función lógica OR.



ALGORITMOS BASADOS EN GRADIENTE

Hay dos algoritmos:
(α-LMS)
Propagación inversa

Un algoritmo basado en gradiente toma un vector de pesos inicial, calcula la función de error y su gradiente, y luego obtiene un nuevo vector de pesos modificando el vector de pesos inicial en dirección opuesta al gradiente de la función error, por lo general la función error es cuadratico. El proceso se repite hasta que el error se encuentra en el límite establecido.

La regla para actualizar pesos en estos algoritmos esta dada por,

$$\omega^{k+1} = \omega^k + \alpha.(-\nabla^k),$$

donde

α : rata de aprendizaje

∇^k : gradiente de la función de error respecto de ω^k

(α-LMS)

Es posible obtener el algoritmo de entrenamiento para un combinador lineal (neurona sin función de activación), utilizando el gradiente de la función de error.

Para el patrón de entrada x^k el error cuadratico es,

$$(e^k)^2 = \left\{d^k - (\omega^k)^T . x^k\right\}^2$$

El gradiente esta dado por,

$$\nabla^k = \frac{\partial (e^k)^2}{\partial \omega^k} = \left[\frac{\partial (e^k)^2}{\partial \omega_0^k} \frac{\partial (e^k)^2}{\partial \omega_1^k} \cdots \frac{\partial (e^k)^2}{\partial \omega_n^k}\right]$$

El nuevo vector de pesos será

$$\omega^{k+1} = \omega^k - \alpha . \frac{\partial (e^k)^2}{\partial \omega^k}$$

derivando el error en se tiene,

$$\omega^{k+1} = \omega^k - 2.\alpha.e^k . \frac{\partial e^k}{\partial \omega^k}$$

reemplazando el error en la derivada

$$\omega^{k+1} = \omega^k - 2.\alpha.e^k . \frac{\partial (d^k - (\omega^k)^T . x^k)}{\partial \omega^k}$$

finalmente,

$$\omega^{k+1} = \omega^k - 2.\alpha.e^k . x^k$$

En esta ecuación el error es lineal, se utiliza para actualizar los pesos de una neurona sin función de activación.

Propagación inversa

Las redes neuronales trabajando bajo el algoritmo de retropropagación del error, han sido independientemente desarrolladas por un basto número de investigadores de muy diversas disciplinas. La primera aproximación del gradiente descendente en el entrenamiento de una red neuronal artificial multicapa fue desarrollado por un matemático llamado Amari en 1967 con su obra "A theory of adaptive pattern classifiers", quien introdujo una capa intermedia de elementos procesadores para realizar una clasificación no lineal. Las aproximaciones de Amari, iban por el

camino correcto, pero no acabó la descripción de cómo formar un mapeo multicapa.

Bryson y Ho, en 1969 desarrollaron un algoritmo muy similar al de retropropagación del error para un control no lineal adaptable.

Werbos, en 1974, desarrolló de manera independiente el algoritmo de retropropagación del error y varias variantes, llamándolo algoritmo de propagación dinámica hacia adelante, todo esto mientras desarrollaba su tesis doctoral en estática.

Parker, en 1982, rediseñó el algoritmo de retropropagación del error, llamándolo algoritmo de aprendizaje lógico y dió los primeros pasos para patentar su trabajo mientras se graduaba en la Universidad de Stanford.

En 1986 Rumelhart, Hinton y Williams explotaron el potencial del algoritmo de retropropagación del error, despertando el interés de la comunidad científica.

Son muchas las aplicaciones que resultan difíciles de realizar porque hay muchos problemas cuya solución no es adecuada mediante procesos secuenciales. Por ejemplo, las aplicaciones que deben realizar complejas traducciones de datos que no poseen una función de correspondencia predefinida que describa el proceso de traducción o aquellas que deben proporcionar una mejor aproximación como salida cuando se les presentan unos datos de entrada ruidosos. Las Redes Neuronales se emplean debido a su capacidad para adaptarse o aprender, generalizar u organizar datos cuyas operaciones están basadas en procesos paralelos.

Un algoritmo que ha resultado útil para la solución de estos y muchos otros problemas que requieren el reconocimiento de tramas complejas y la realización de funciones de correspondencia no triviales es el algoritmo de retropropagación del error, descrito formalmente por Werbos y posteriormente por Parker, Rumelhart y McClelland.

Una red trabajando bajo el algoritmo de Retropropagación del error, está diseñada para que funcione como una red multicapa con propagación hacia adelante, empleando un aprendizaje supervisado. Una red de Retropropagación es entrenada con una serie de entradas y sus correpondientes salidas, motivo por el cual, se dijo que el algoritmo de retropropagación del error trabaja bajo un modo de aprendizaje supervisado.

Una red de Retropropagación funciona de la manera siguiente:

Durante la sesión de entrenamiento, los pesos de la red convergen en un punto en el espacio de pesos de la red, en el que el problema es conocido por la red; cuando la red ha alcanzado este punto, puede dar la salida correcta para cada entrada dada. Sin embargo, debido a la naturaleza de la red, no siempre se puede aprender el problema exactamente. La salida producida por la red cuando se presenta una entrada, puede ser una aproximación de la salida correcta, motivo por el cual es difícil decir cuando la red ha aprendido el problema en cuestión.

Ejemplo. Considere los precios de una acción dados

Trimestre	Valor Acción $
1	0.5
2	0.4

3	0.7
4	0.8

Entrenar una neurona con función de activación tanh para sintetizar una autoregresión de segundo orden,

$$p^k = g(p^{k-1}.p^{k-2})$$

tomar el vector de pesos inicial y la rata de aprendizaje como

$$\left[\omega^1\right]^T = \left[0.1\ \ 0.1\ \ 0.1\right] \qquad \alpha = 0.1$$

Solución. Se construye una nueva tabla que contiene los patrones que se utilizan para entrenar la neurona.

La autoregresión de orden dos indica que la red calcula el precio del trimestre actual p^k tomando como entradas los precios de los trimestres anteriores p^{k-1}, p^{k-2}.

X_1	X_2	y_d
0.5	0.4	0.7
0.4	0.7	0.8

La formula para actualizar los pesos es,

$$\omega^{k+1} = \omega^k + 2\alpha.e^k.\left\{1-(y^k)^2\right\}x^k$$

La suma s para la iteración k es

$$s^k = (1)(\omega_0^k) + (x_1^k).(\omega_1^k) + (x_2^k).(\omega_2^k)$$

Iteración 1. K=1

Reemplazando el patrón de entrada $x_1^1 = 0.5, X_2^1 = 0.4$ en la suma

$$s^1 = (1)(0.1) + (0.5)(0.1) + (0.4)(0.1) = 0.19$$

la salida de la neurona es

$$y^1 = tanh(s^1) = 0.1877$$

El error no lineal se obtiene restando la salida actual de la salida deseada,

$$e^1 = y_d^1 - y^1 = 0.7 - 0.1877 = 0.5123$$

actualizando el vector de pesos,

$$\omega^2 = \omega^1 + 2.\alpha.e^1.\left\{1 - (y^1)^2\right\}.x^1$$

reemplazando los valores numéricos,

$$\begin{bmatrix} \omega_0^2 \\ \omega_1^2 \\ \omega_2^2 \end{bmatrix} = \begin{bmatrix} 0.1 \\ 0.1 \\ 0.1 \end{bmatrix} + (2)(0.1).(0.5123).\left[1 - 0.81877^2\right].\begin{bmatrix} 1.0 \\ 0.5 \\ 0.4 \end{bmatrix}$$

realizando las operaciones indicadas

$$\begin{bmatrix} \omega_0^2 \\ \omega_1^2 \\ \omega_2^2 \end{bmatrix} = \begin{bmatrix} 0.1988 \\ 0.1494 \\ 0.1395 \end{bmatrix}$$

Iteración 2. K=2

Reemplazando el patrón de entrada $x_1^2 = 0.4, X_2^2 = 0.7$ en la suma

$$s^2 = (1)(0.1988) + (0.4)(0.1494) + (0.7)(0.1395) = 0.3562$$

la salida de la neurona es

$$y^2 = tanh(s^2) = 0.3419$$

El error no lineal se obtiene restando la salida actual de la salida deseada,

$$e^2 = y_d^2 - y^2 = 0.8 - 0.3419 = 0.4581$$

actualizando el vector de pesos,

$$\omega^3 = \omega^2 + 2.\alpha.e^2.\left\{1 - (y^2)^2\right\}.x^2$$

reemplazando los valores numéricos,

$$\begin{bmatrix} \omega_0^3 \\ \omega_1^3 \\ \omega_2^3 \end{bmatrix} = \begin{bmatrix} 0.1988 \\ 0.1494 \\ 0.1395 \end{bmatrix} + (2)(0.1).(0.4581).\left[1 - 0.3419^2\right].\begin{bmatrix} 1.0 \\ 0.4 \\ 0.7 \end{bmatrix}$$

realizando las operaciones indicadas

$$\begin{bmatrix} \omega_0^3 \\ \omega_1^3 \\ \omega_2^3 \end{bmatrix} = \begin{bmatrix} 0.2797 \\ 0.1818 \\ 0.1961 \end{bmatrix}$$

Iteración 3. K=3

Se inicia el recorrido de la tabla de nuevo, reemplazando el patrón de entrada $x_1^3 = 0.5, X_2^3 = 0.4$ en la suma

$$s^3 = (1)(0.2797) + (0.5)(0.1818) + (0.4)(0.1961) = 0.4490$$

la salida de la neurona es $y^3 = tanh(s^3) = 0.4211$

El error no lineal se obtiene restando la salida actual de la salida deseada,
$e^3 = y_d^3 - y^3 = 0.7 - 0.4211 = 0.2789$

actualizando el vector de pesos, $\omega^4 = \omega^3 + 2.\alpha.e^3.\left\{1 - (y^3)^2\right\}x^3$

reemplazando los valores numéricos,

$$\begin{bmatrix} \omega_0^4 \\ \omega_1^4 \\ \omega_2^4 \end{bmatrix} = \begin{bmatrix} 0.2797 \\ 0.1818 \\ 0.2145 \end{bmatrix} + (2)(0.1).(0.2789).\left[1 - 0.4211^2\right]\begin{bmatrix} 1.0 \\ 0.5 \\ 0.4 \end{bmatrix}$$

realizando las operaciones indicadas

$$\begin{bmatrix} \omega_0^4 \\ \omega_1^4 \\ \omega_2^4 \end{bmatrix} = \begin{bmatrix} 0.3256 \\ 0.2047 \\ 0.2145 \end{bmatrix}$$

Iteración 4. K=4

Reemplazando el patrón de entrada $x_1^4 = 0.4, X_2^4 = 0.7$ en la suma

$$s^4 = (1)(0.3198) + (0.4)(0.2018) + (0.7)(0.2121) = 0.5490$$
la salida de la neurona es

$$y^4 = tanh(s^4) = 0.4998$$

El error no lineal se obtiene restando la salida actual de la salida deseada,

$$e^4 = y_d^4 - y^4 = 0.8 - 0.4998 = 0.3002$$

actualizando el vector de pesos,

$$\omega^5 = \omega^4 + 2.\alpha.e^4.\left\{1 - (y^4)^2\right\}.x^4$$

reemplazando los valores numéricos,

$$\begin{bmatrix} \omega_0^5 \\ \omega_1^5 \\ \omega_2^5 \end{bmatrix} = \begin{bmatrix} 0.3256 \\ 0.2047 \\ 0.2145 \end{bmatrix} + (2)(0.1).(0.3002).\left[1 - 0.4998^2\right]\begin{bmatrix} 1.0 \\ 0.4 \\ 0.7 \end{bmatrix}$$

realizando las operaciones indicadas

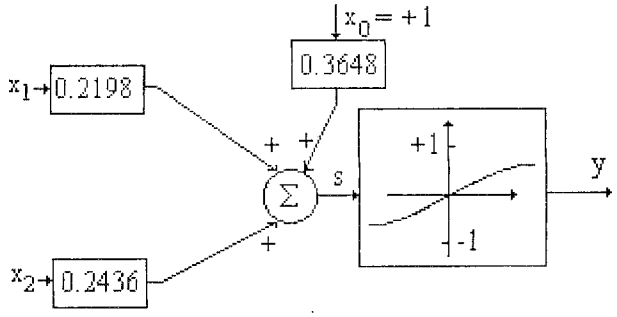
$$\begin{bmatrix} \omega_0^5 \\ \omega_1^5 \\ \omega_2^5 \end{bmatrix} = \begin{bmatrix} 0.3648 \\ 0.2198 \\ 0.02436 \end{bmatrix}$$

El procedimiento se repite hasta que se cumpla lo siguiente:

El error máximo e_m sobre todos los patrones es menor a un error dado ε.
El error medio cuadratico e_{rms} sobre todos los patrones es menor a un ε, el error medio cuadratico se conoce como error RMS y esta dado por,

$$e_{rms} = \sqrt{\frac{\sum_{l=1}^{NP}(y_d^l - y^l)^2}{NP}}$$

donde NP es el número de patrones.



Error producido por la red con los pesos obtenidos en la iteracción 4

X_1	X_2	y_d	y	e	$(e)^2$

0.5	0.4	0.7	0.5170	0.1830	0.0335
0.4	0.7	0.8	0.5534	0.2466	0.0608

$$\sum (e)^2 = 0.0943$$

El máximo error sobre todos los patrones es e_m=0.2466
El error cuadrático sobre los patrones se obtiene con la ecuación antes mencionada, e_{rms}=0.2171.

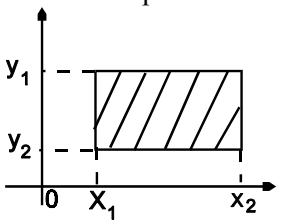
Los dos errores son grandes por lo tanto se debe continuar con el entrenamiento, usualmente ε=0.01.

REDES NEURONALES ESTÁTICAS

La tecnología neuronal trata de reproducir el proceso de solución de problemas del cerebro. Así como los humanos aplican el conocimiento ganado con la experiencia a nuevos problemas o situaciones, una red neuronal toma como ejemplos problemas resueltos para construir un sistema que toma decisiones y realiza clasificaciones.

Los problemas adecuados para solución neuronal son aquellos que no tienen solución computacional precisa o requieren algoritmos muy extensos como en el caso de reconocimiento de imágenes.

Si se va a realizar un control de calidad sobre las piezas terminadas en una factoría, se utiliza una prueba no destructiva que separa las piezas buenas de las piezas malas; inicialmente, hay dos criterios de prueba:

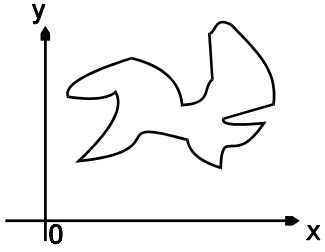


x: Tamaño del defecto.
y: Localización.

Las piezas son buenas si se encuentran en el área sombreada.

$x_1 \le x \le x_2$
$y_1 \le x \le y_2$

Sin embargo, el caso anterior no es real y la relación entre x e y puede ser más compleja como:



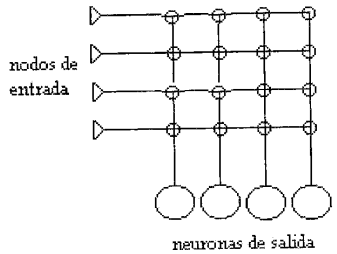
La forma tradicional para resolver el problema consiste en construir una base de datos con las coordenadas (x,y) de producto bueno y malo.

Los sistemas basados en redes neuronales aprenden relaciones complejas entre muchas variables. Al entrenar la red con ejemplos de entradas-salidas, esta determina la superficie en ele espacio multidimensional. En lugar de codificar muchas reglas o algoritmos, la red neuronal aprende el borde del área sombreada.

RED TIPO PERCEPTRÓN

El Perceptrón fué propuesto por Rosenblatt en 1959 en su obra "Principles of Neurodynamics".Los perceptrones, Rosemblatt probó un teorema sobre el aprendizaje del perceptrón y dado esto, en los 60´s los Perceptrones crearon un gran interés en la comunidad científica sobre las Redes Neuronales.

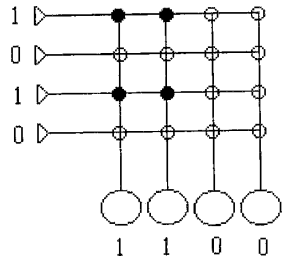
Es perceptrón es el tipo de red neuronal más simple. Tienen una capa de conexiones modificables entre sus nodos de entrada y las neuronas de salida, el conocimiento se almacena en el patrón de conexiones.



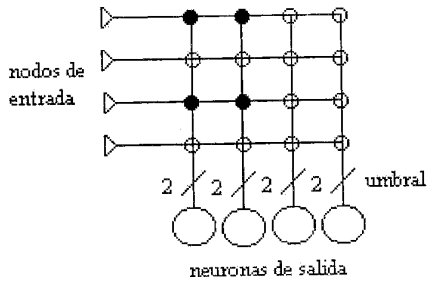
Para un patrón binario de entrada se asocia un patrón binario de salida. Cada neurona suma sus entradas y si el resultado es mayor o igual al umbral (al iniciar el aprendizaje es cero) se activa y toma el valor de uno, en caso contrario la salida permanece en cero.

Ejemplo. Asociar el patrón de entrada 1010 con el patrón de salida 1100.

El aprendizaje es Hebbiano, se presenta un patrón de activación en los nodos de entrada y, simultáneamente, el patrón deseado en las neuronas de salida. Si un par entrada-salida está activo entonces la conexón se fortalece, en otras palabras, las conexiones que tengan entrada uno y salida uno se marcan en negro.



Al incrementar el número de asociaciones almacenadas, ocurren activaciones no deseadas en las neuronas de salida. Para una operación correcta de la red se debe incrementar el umbral en las neuronas. Esto es, cada neurona se activa sólo si la suma de las actividades excede el umbral dado.



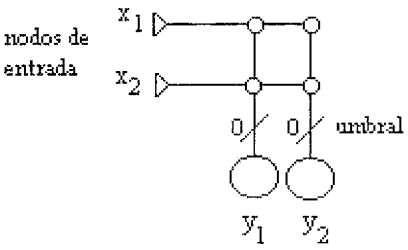
La red tipo perceptrón fue también explorada por David Willshaw de la Universidad de Edinburgo a principios de los ochentas. Esta re es muy limitada, puede almacenar unos pocos patrones y realizar algunas asociaciones simples.

Ejemplo. Propagar las conexiones de una red tipo perceptrón para sintetizar la siguiente tabla:

X_1	X_2	Y_1	Y_2
0	0	0	0
0	1	0	1
1	0	0	1
1	1	1	1

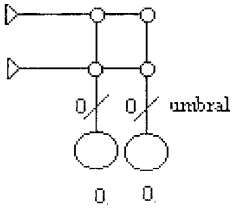
Los patrones son separables linealmente; el umbral inicial para las neuronas es cero, una conexión activa (negro) vale uno y la conexión no-activa (blanco) vale cero.

La red, tiene dos nodos de entrada y dos neuronas de salida.



Se entrena la red perceptrón utilizando la ley de Hebb, esto es, si la entrada a una neurona está activa y su salida también, la conexión se fortalece (se incrementa en uno). La función de activación de cada neurona opera cuando la suma de las señales de entrada a una neurona s, es mayor o igual al umbral y entonces, la salida y de esa neurona es uno.

Patrón 1

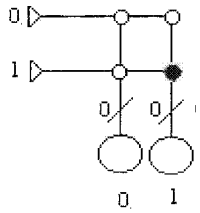


Las sumas en cada neurona y sus salidas después de la función de activación son:

S_1=(0).(0)+(0).(0)=0, y_1=0
S_2=(0).(0)+(0).(0)=0, y_2=0,

No es necesario activar conexiones pues se satisface el patrón entrada-salida.

Patrón 2

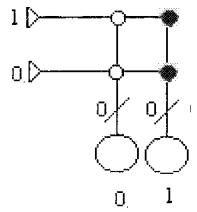


Las sumas en cada neurona y sus salidas después de la función de activación son

S_1=(0).(0)+(0).(0)=0, y_1=0
S_2=(0).(0)+(1).(1)=1, y_2=1,

Se activa una conexión pues hay una entrada y salida en 1.

Patrón 3

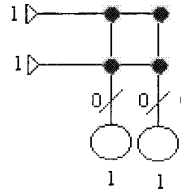


Las sumas en cada neurona y sus salidas después de la función de activación son,

S_1=(0).(0)+(0).(0)=0, y_1=0
S_2=(1).(1)+(0).(1)=1, y_2=1,

Se activa una conexión pues hay una entrada y salida en 1.

Patrón 4



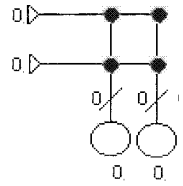
Las sumas en cada neurona y sus salidas después de la función de activación son,

$S_1=(1).(1)+(1).(1)=2$, $y_1=1$
$S_2=(1).(1)+(1).(1)=2$, $y_2=1$,

Se activan dos conexiones en la red pues las dos entradas están en uno y las dos salidas se encuentran en el mismo estado.

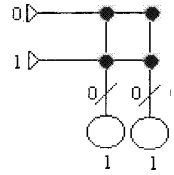
Después de finalizar el primer barrido por el conjunto de patrones , vamos a revisar la tabla desde el comienzo para ajustar los umbrales, si es necesario y así evitar salidas erróneas de la red.

Patrón 1

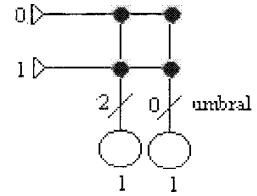


Se cumple

Patrón 2

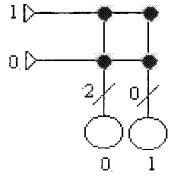


El patrón deseado para las entradas mostradas es, según la tabla $y_1=0$, $y_2=1$. Se ve que la red produce una salida errónea por lo tanto se debe modificar el umbral de la primera neurona como se muestra a continuación.



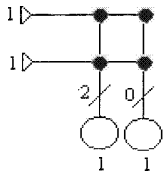
Con este ajuste el patrón dos se cumple.

Patrón 3



Se cumple

Patrón 4



Se cumple y por lo tanto, esto finaliza el entrenamiento de la red.

La red tipo perceptrón podría ser el primer tipo de red a utilizar en un minirobot para que aprenda por sí mismo, por ejemplo, a navegar un espacio asociando entradas (lecturas de los sensores) a las salidas (acciones), por recompensa y castigo. Un esquema de aprendizaje puede ser el siguiente, cuando la asociación es correcta se incrementa las conexiones en uno, en caso contrario se disminuyen en uno.

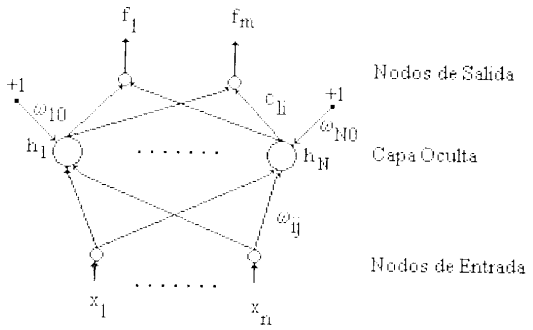
RED MULTICAPA

Una red neuronal es una estructura paralela que procesa información en forma distribuida y está constituida por elementos de procesamiento, llamados neuronas, interconectados con canales unidireccionales.

Cada elemento de procesamiento tiene una conexión de salida, la cual se ramifica en tantas conexiones colaterales como se requiera (cada una lleva la misma señal, la salida del elemento procesador).

La red multicapa es de estructura n-N-m. Hay n nodos de entrada, N neuronas ocultas y m neuronas de salida. Los nodos de entrada reciben los valores a procesar x_j (j= 1,…,n), las neuronas ocultas reciben las entradas xj ponderadas por los pesos w_{ij} y producen las salidas h_i (i=1,…,N), finalmente los nodos de salida hacen la suma ponderada de las salidas de las neuronas ocultas.

La red neuronal multicapa puede resolver problemas de clasificación de patrones no separables linealmente como la función lógica XOR.



La salida de las neuronas ocultas esta dada por

$$h_i = \sigma(s_i) = \sigma\left(\sum_{j=1}^{n} \omega_{ij} . x_j + \omega_{i0}\right)$$

donde

i=1,…,N

ω_{ij} peso que une la neurona i con la entrada j.

ω_{i0} peso asociado a la entrada umbral x_0=1

Las salidas de la red son:

$$f_1 = \sum_{i=1}^{N} c_{li} . h_i$$

donde,
l=1,…m,
c_{li}: peso que une la salida l con la neurona i

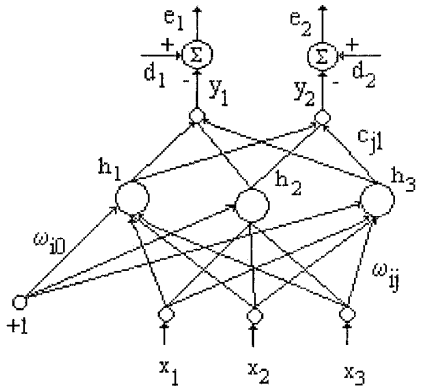
La red no tiene entradas umbral ni pesos umbral en las salidas cuando se trata de nodos. Es posible tener neuronas en la capa de salida, en este caso:

$$f_l = \sigma\left(\sum_{i=1}^{N} c_{li} . h_i + c_{l0} \right)$$

la suma ponderada pasa por la función de activación y existe un peso umbral por neurona de salida.

Entrenamiento con el algoritmo de propagación inversa

Para entrenar una red neuronal se asume que las neuronas tienen una función de activación continua y se utiliza el gradiente de la función de error para actualizar el vector de pesos.



Si se considera una red multicapa 3-3-2 se utiliza la ley general del aprendizaje basada en gradiente,
$\omega_{k+1} = \omega_k + \alpha.(-\nabla_k)$

Las entradas a la red son x_1, x_2, x_3, y las correspondientes salidas deseadas son d_1, d_2, por patrón entradas – salidas es el vector [x_1, x_2, x_3, d_1, d_2]. Las salidas de la red son y_1, y_2, por lo tanto los errores entre salidas deseadas y salidas actuales son

$e_1 = d_1 - y_{1,} e_2 = d_2 - y_2 .$

Durante la presentación de un patrón, el error cuadrático de las salidas esta dado por,

$$e^2 = (d_1 - y_1)^2 + (d_2 - y_2)^2$$

donde

$$y_1 = c_{11}.h_{1+} c_{12}.h_{2+} c_{13}.h_3$$

$$y_2 = c_{21}.h_{1+} c_{22}.h_{2+} c23.h_3$$

El error es cero si los errores para cada salida son cero. Calculando el gradiente del error cuadrático respecto de los pesos de salida de la red neuronal,

$$\frac{\partial e^2}{\partial c_{li}} = -2.e_l.h_i$$

entonces la regla para actualizar los pesos de salida es,

$$c_{li}^{k+1} = c_{li}^{k} + 2.\alpha.e_l.h_i$$

donde
l=1,..,m
i=1,...,N

La salida de las neuronas ocultas esta dada por,

$$h_1 = \sigma(s_1) = \sigma(\omega_{11}.x_1 + \omega_{12}.x_2 + \omega_{13}.x_3 + \omega_{10})$$

$$h_2 = \sigma(s_2) = \sigma(\omega_{21}.x_1 + \omega_{22}.x_2 + \omega_{23}.x_3 + \omega_{20})$$

$$h_3 = \sigma(s_3) = \sigma(\omega_{31}.x_1 + \omega_{32}.x_2 + \omega_{33}.x_3 + \omega_{30})$$

El gradiente del error cuadrático respecto de los pesos de la capa oculta de la red neuronal es:

$$\frac{\partial e^2}{\partial \omega_{ij}} = -2.\delta_i.\sigma'(s_i).x_j$$

Donde δ_i es un error dado por:

$$\delta_i = \sum_{l=1}^{m} c_{li} e_l = \sum_{l=1}^{m} c_{li}.(d_l - y_l)$$

i=1,2,…,N

Entonces la regla para actualizar los pesos de entrada es: $\omega_{ij}^{k+1} = \omega_{ij}^{k} + 2.\alpha.\delta_i.\sigma'(s_i).x_j$

Donde, i=1,…,N, j=0,…,n

Y la derivada de la función de activación es: $\sigma'(\theta) = \frac{d\sigma(\theta)}{d\theta}$

Los pesos de salida se actualizan utilizando los errores e_l. Los pesos de entrada se actualizan utilizando los errores δ_i, estos son los errores de la red ponderados. El algoritmo se denomina de propagación inversa porque los errores se propagan en sentido inverso a las entradas para actualizar los pesos de la red, los errores e_1 y e_2 fluyen en sentido contrario a las entradas x_1, x_2, x_3.

Este algoritmo fue desarrollado independientemente por David Rumelhart, David Parker y Yann Le Cun.

Durante el entrenamiento de la red las unidades ocultas codifican las regularidades subyacente. Cuando se mira localmente, el procedimiento sintoniza los pesos, pero el efecto global es el aprendizaje estructural, creando términos que expresan regularidades.

El algoritmo de propagación inversa es criticado por su lentitud incluso para problemas pequeños. El problema propio del gradiente, esto es, trampas en los mínimos locales no es tan importante como la lentitud del algoritmo. En la práctica un método basado en gradiente es lento porque trabaja con información puntual (un patrón a la vez) y no un panorama claro de la superficie de error. En espacios de alta dimensión la superficie de error contiene pendientes suaves pero altamente curvadas, un paso pequeño (α pequeño) tarda bastante en bajar por la pendiente y un paso grande (α grande) produce oscilaciones.

Momentum en el algoritmo de propagación inversa

Algunos investigadores como D.E. Rumelhart, G.E. Hinton y R. J. Williams propusieron una modificación al algoritmo de propagación inversa. La idea es adicionar un "inercia" a las fórmulas de actualización para tener en cuenta los cambios realizados durante la presentación del patrón anterior. Las fórmulas modificadas con el término adicional de momentum son,

$$c_{li}^{k+1} = c_{li}^{k} + 2.\alpha.e_l.h_i + \mu.(c_{li}^{k} - c_{li}^{k-1})$$

$$\omega_{ij}^{k+1} = \omega_{ij}^{k} + 2.\alpha.\delta_i.\sigma'(s_i).x_j + \mu.(\omega_{ij}^{k} - \omega_{ij}^{k-1})$$

$$i = 1,...,N;\ j = 0,...,n;\ l = 1,...,m$$

La entrada cero es la que afecta el peso de umbral. El término de momentum filtra las variaciones de alta frecuencia de la superficie de error en el espacio de pesos, la constante μ es el factor que determina la importancia del momentum este trata de hacer el cambio del vector de pesos en la misma dirección anterior, evitando cambios erráticos.

Tasa de aprendizaje dinámica

Por medio de simulaciones se ha logrado establecer que una rata de aprendizaje dinámica puede acelerar la convergencia de la red. Carter (1987, Silva y Almeida (1991). La idea es que α cambie según el error de aprendizaje, α debe ser grande para un error grande, α debe ser pequeño para un error pequeño.

Algunas expresiones para hacer la rata de aprendizaje dinámica

$$\alpha = \alpha_0 . e$$
$$\alpha = \alpha_0 . \csc(\pi . e)$$
$$\alpha = \alpha_0 . [2 + \cos(2 . \pi . e)]$$
$$\alpha = \alpha_0 . \left[1 + tan(\pi . e - \frac{\pi}{2})\right]$$
$$\alpha = 3 . \alpha_0 . \csc(\pi . e)$$

donde α_0 : rata de aprendizaje inicial

Para un entrenamiento rápido de la red se recomienda seleccionar los pesos iniciales al azar (valores pequeños) y escoger una función de activación adecuada para los datos.

Generalización de las redes neuronales

Una de las características más sobresalientes de las redes neuronales es su habilidad para reconocer o clasificar patrones nunca antes presentados a la red. Aún es tema de investigación, cómo el tamaño o la estructura de la red afectan esta cualidad.

Para mejorar la capacidad de generalización de una red neuronal, se entrena una red con un número grande de neuronas ocultas (igual al número de patrones de entrenamiento, una neurona oculta por patrón). Cuando el error empieza a disminuir se eliminan las neuronas (esto se conoce como podar la res) y se entrena de nuevo la red a partir de los valores de pesos anteriores. Un número pequeño de neuronas ocultas extraen características generales lo cual implica una buena generalización. Por el contrario cuando hay demasiadas neuronas en la capa oculta, la red memoriza los patrones y no sus características. En el otro extremo, si el número de neuronas ocultas es muy pequeño, la red no aprende los patrones.

Determinar el número de neuronas ocultas para un problema dado involucra experiencia e intuición. Las neuronas ocultas son detectores de características, en otras palabras, cada neurona oculta se activa cuando un patrón particular de entradas está presente.

Algoritmo Chemotaxis

El algoritmo de chemotaxis permite entrenar redes neuronales de cualquier tipo sin utilizar el gradiente de la función error.

El algoritmo ha sido formulado considerando el movimiento de una bacteria en un medio donde hay gradiente de solución alimenticia. La bacteria se mueve inicialmente al azar hasta detectar un aumento en la concentración de la solución, luego continua en esa dirección.

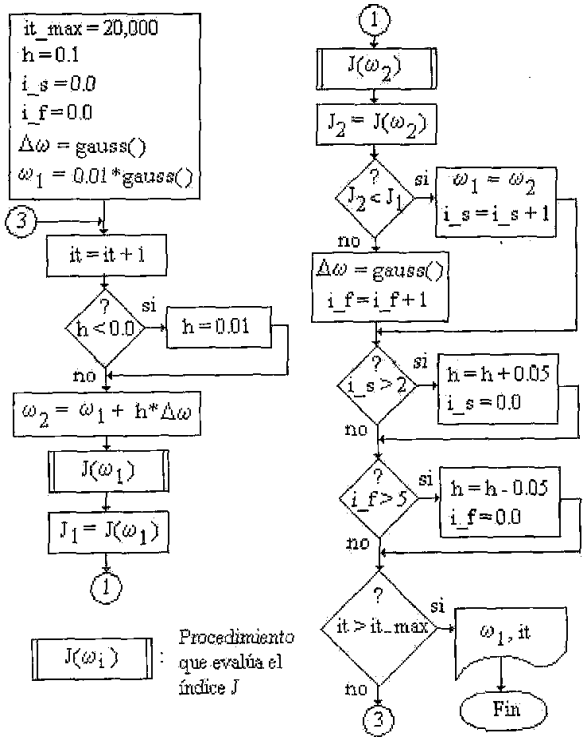
Este algoritmo toma pasos al azar (vector de pesos) con distribución Gaussiana, cuando el paso es exitoso (la función de error se reduce en un problema de minimización), el algoritmo continua en esa dirección hasta que la función de error J no muestra más cambios. Una función de error puede ser la distancia entre una respuesta deseada y la respuesta actual para todos los patrones,

$$J=\sum_{k=1}^{NP}(d_{1k}-y_{1k})^2+(d_{2k}-y_{2k})^2+\ldots+(d_{mk}-y_{mk})^2$$

Para una red con m salidas y NP patrones de entrenamiento:

d_{ik}: salida deseada i para el patrón k.
y_{ik}: salida actual i para el patrón k.

También es posible con este algoritmo penalizarla función de error para permitir ciertos valores del vector de pesos.



Las variables que aparecen en el algoritmo son:

it_max: máximo número de iteraciones.
it: contador de iteraciones.
h: rata de aprendizaje.
i_s: contador de pasos exitosos.
i_f: contador de pasos errados.
$\Delta\varpi$: perturbación del vector de pesos.
ϖ_1: viejo vector de pesos.
ϖ_2: nuevo vector de pesos .
Gauss(): generador de números aleatorios con distribución gaussiana (0,1).
J_i: índice (función de error correspondiente al vector de pesos ϖ_1.

Algunas características del algoritmo de chemotaxis son:
No requiere el cálculo del gradiente de la función de error con respecto a los pesos.

La función de error se puede penalizar para satisfacer otras condiciones, por ejemplo, pesos limitados para garantizar la estabilidad en lazos de control.
La programación del algoritmo es muy sencilla.
Se puede utilizar para redes estáticas (redes multicapa) o para redes dinámicas (redes recurrentes).

Aplicaciones de las redes neuronales multicapa

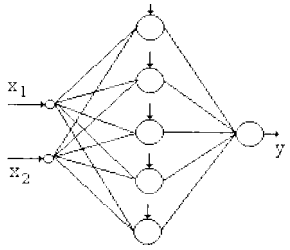
El primer paso para utilizar una red consiste en organizar los patrones como entrada(s) – salida(s) deseada(s), en ocasiones es necesario filtrar los datos para acelerar el aprendizaje de la red.

Las redes multicapa se pueden usar básicamente en dos áreas: clasificación de patrones y síntesis de funciones lineales.

Función lógica XOR

X_1	X_2	Y
-1	-1	-1
-1	+1	+1
+1	-1	+1
+1	+1	-1

Red neuronal 2-5-1, con un error cuadratico sobre los patrones menor al 1%. La función de activación de las neuronas ocultas es tanh.



Los pesos son

ω_{10}=4.6449 ω_{20}=-3.6780 ω_{30}=-0.2215
ω_{11}=-2.9748 ω_{21}=3.7414 ω_{31}=4.5761
ω_{12}=0.2032 ω_{22}=-4.7533 ω_{32}=-1.7894

ω_{40}=2.6933 ω_{50}=-2.0001
ω_{41}=1.8923 ω_{21}=1.2154
ω_{42}=3.7912 ω_{22}=3.2576

c_{10}=1.4127 c_{11}=-0.6015
c_{12}=4.8919 c_{13}=-3.0623
c_{14}=4.3013 c_{15}=0.2352

Las salidas de las neuronas ocultas son,

$$h_1 = tanh(4.6449 - 2.9748x_1 + 0.2032x_2)$$
$$h_2 = tahnh(-3.6780 + 3.7414x_1 - 4.7533x_2)$$
$$h3 = tanh(-0.2215 + 4.5761x_1 - 1.7894x_2)$$
$$h_4 = tanh(2.6933 + 1.8923x_1 + 3.7912x_2)$$
$$h_5 = tanh(-2.0010 + 1.2154x_1 + 3.2576x_2)$$

la salida de la red es,

$$y = tanh(1.4127 - 0.6015h_1 + 4.8919h_2 - 3.0623h_3 + 4.3013h_4 + 0.2352h_5)$$

Estudio de Crédito

Una empresa de financiamiento comercial recibe diariamente solicitudes de crédito por parte de personas naturales. El proceso normal de cada solicitud es la remisión de los formularios, tanto del solicitante, como de sus codeudores, a un abogado especializado en el estudio de los mismos. El abogado genera un concepto de aprobación o rechazo de la solicitud.

Después de un estudio de la información solicitada en los formularios de cierta firma se definen las siguientes entradas binarias (SI,NO) a la red,

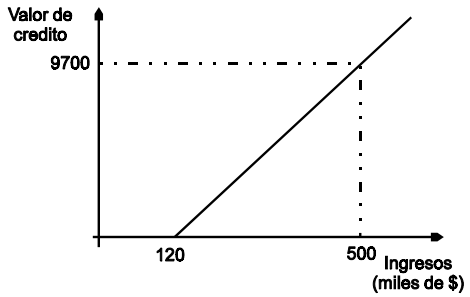
Ingresos,

Cuentas corrientes,
Cuentas de ahorros,
Vehículos,
Papeles de vehículos,
Finca raíz,
Papeles finca raíz,
Hipotecas,
Referencias comerciales,
Referencias familiares,
Existencia de codeudor.
Las salidas de la red son,

Aprobación del crédito,
Verificar ingresos,
Verificar cuentas bancarias,
Verificar documentos propiedades.

En esta aplicación se tomó la información de 36 formularios ya procesados por el abogado. Para transferir el valor de los ingresos a un simple SI o NO se utilizó una curva empírica.

El entrenamiento de la red se realizó con 12 patrones de los 36 disponibles. La estructura de la red fue 11-30-4, aprendió después de 500 ciclos con un error del 0.1%



La generalización de la red se probó con los 24 formularios restantes. La red acertó 18 casos y en 6 su resultado es distinto del concepto del abogado.

Clasificación de Electromiogramas

El control de prótesis (piernas artificiales) se puede lograr utilizando una señal microeléctrica medida por electrodos superficiales. Los sistemas tradicionales para reconocimiento de patrones consumen mucho tiempo lo cual es inaceptable en la práctica. Es posible entrenar una red neuronal para clasificar señales mioeléctricas en categorías que pueda entender la prótesis, no se necesita suponer distribución probabilística alguna y la red maneja información difusa.

La clasificación de electromiogramas (EMGs) involucra tres pasos:

Amplificar la señal muscular, filtrarla en la banda de 20 Hz – 2 KHz para eliminar el ruido, convertir la señal análoga a digital (A/D) y muestrearla a 500 Hz.
Ajustar un modelo autoregresivo a la señal muestreada, y tomar el parámetro a_1 y la potencia p de la autoregresión.
Utilizar una red para la clasificación usando a_1 y p.
La red utilizada en este problema tiene dos entradas, cuatro neuronas en la capa oculta y cuatro neuronas de salidas. Después del entrenamiento la red discrimina cuatro regiones con los parámetros a_1 y p.

Identificación de fallas en sistemas de potencia

La interpretación de alarmas en un centro de control es algo común en las centrales de generación, el operador necesita información en lugar de alarmas.

Muchos de los enfoques tradicionales requieren una definición precisa del sistema de potencia y de las reglas a usar en el análisis.

Una red neuronal aprende lo cual contrasta con los sistemas expertos basados en reglas, las redes generan sus propias reglas a partir de unos ejemplos. La habilidad de las redes neuronales para aprender y construir estructuras únicas para un problema particular, sin necesidad de reglas explícitas y un esfuerzo humano intenso, hace que estos modelos sean útiles en aplicaciones de reconocimiento de patrones donde las reglas son muy complejas para definirlas y mantenerlas.

En el caso de la interpretación de alarmas en un sistema de potencia, donde las configuraciones, patrones de alarmas y problemas de sistema cambian de planta en planta, esta habilidad para aprender abre la posibilidad de construir un procesador de alarmas genérico. Un procesador genérico de alarmas debe tener la habilidad para aprender de ejemplos sin definir reglas, funcionar

con entradas ruidosas y poseer adaptabilidad con el mínimo esfuerzo para distintas plantas.

La interpretación de alarmas involucra que:

Un sistema de potencia en un estado de falla activa cierto patrón de alarmas, si se reconoce el patrón se identifica el problema.
Muchas situaciones de alarma involucran el movimiento de relevos, por lo tanto, las alarmas que aparecen por acción de los relevos se pueden ver como un problema específico en el sistema.
El patrón de alarmas puede aparecer con ruido por problemas en los equipos, mala calibración de relevos e instrumentos. En otras palabras, el patrón esperado de alarmas no siempre es el mismo.
El procesador de alarmas actúa sobre el mismo conjunto de datos disponible al operador, por lo tanto, no se espera que supere al operador en su habilidad para diagnosticar el problema, pero se espera que haga el trabajo en una fracción del tiempo consumido por este.

Para producir un sistema de interpretación de alarmas en sistemas de potencia con una re neuronal se siguen los pasos:

Crear una lista de contingencias que considere el mayor número de casos.
Para cada contingencia, identificar a partir de la configuración de los relevos e protección y de los estudio de flujo de carga, las alarmas esperadas.
Entrenar la red.
La red está lista para interpretar alarmas reconociendo los patrones de entradas con o sin ruido.

Es importante mencionar que no es posible considerar todas las contingencias durante el entrenamiento. Sin embargo, la red neuronal se puede actualizar a medida que llegan nuevos casos.

El interpretador se instaló en una subestación de distribución, con la ayuda de los ingenieros de la subestación se generó una lista de problemas del sistema y sus alarmas asociadas. El conjunto ‘problemas del sistema/alarmas’ son los patrones de entrenamiento.

Una vez entrenada la red evaluó en la práctica, lo que se observó fue:

El interpretador funcionó correctamente cuando no hubo ruido.
Cuando hubo ruido el interpretador identificó el problema en algunos casos y en otros señaló el problema más probable.
El interpretador reconoce la falla en un segundo (usando un PC).

Detección de explosivos en aeropuertos

En esta aplicación se utilizó una red neuronal como clasificador de patrones para discriminar entre maletas con o sin explosivos.

El sistema usa activación térmica por neutrones para detectar la presencia de explosivos. En esta técnica, una maleta sobre el transportador pasa a través de una fuente de neutrones y un arreglo de detectores. Los neutrones de la fuente penetran en el equipaje y son absorbidos por todos los materiales presentes. Según el elemento químico se emiten diferentes rayos gama después de absorber los neutrones. Estos rayos gama son de alta energía, cruzan el equipaje y son tomados por el arreglo de detectores. Los detectores registran el número de rayos gama observados para cada energía. El número de rayos gama de una energía característica depende del elemento químico presente, su localización, el número de neutrones emitidos por la fuente y la probabilidad de que el elemento capture un neutrón térmico y emita un rayo gama. Como esta probabilidad es una

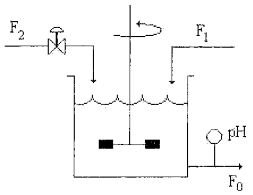
constante para cada elemento químico y el número de neutrones así como el número de rayos gama característicos se miden entonces la cantidad de cada elemento y su localización pueden determinarse.

Los explosivos comerciales y los militares tienen varias características que los diferencian del restante equipaje, una de estas características es la alta densidad de nitrógeno.

Para entrenar la red neuronal, se creó una base de datos con 20.000 maletas normales y 20.000 simuladas con explosivos. Cada detector produce información sobre la cantidad y localización de los elementos químicos en la maleta, entre estos el nitrógeno. Se utilizó una red de propagación inversa, los 200 nodos de entrada reciben un grupo de datos provenientes del detector que se han filtrado y normalizado en el intervalo (-0.5,+0.5). La capa oculta aprende a codificar relaciones que no son explícitas en la entrada. La salida produce valores en una escala suave entre (-0.5,+0.5), un valor de +0.5 significa que la maleta es peligrosa y –0.5 significa que la maleta puede pasar la inspección. Los resultados muestran que una red de dos capas ocultas es suficiente.

Modelo de un sistema químico

Se analizó la dinámica del pH en un tanque reactor, el tanque tiene dos entradas, una contiene hidróxido de sodio y la otra ácido acético.



Para la identificación de la planta se adicionó a la corriente F_2 una señal binaria seudoaleatoria de amplitud igual al 2% del valor de régimen. La entrada (F_2+ruido) y la correspondiente salida pH se almacenaron para entrenar la red.

Las entradas a la red se construyen con una ventana móvil de valores de F_2. La ventana incluye valores actuales y pasados de (pH,F_2) así como valores futuros de F_2. La salida de la red es la predicción de los valores de Ph.

Contenido armónico

Armónico es una magnitud senoidal de frecuencia múltiplo de una frecuencia fundamental. Se entrenó una red neuronal multicapa para reconocer el contenido armónico (3º y 5º) de una señal dada. La fórmula para el contenido armónico es,

$$ca = \frac{V_m(n)}{V_m(1)} \times 100\%$$

Donde n es el orden del armónico.

Para entrenar la red neuronal se crean patrones con una señal de voltaje muestreada,

$\zeta(i) = \zeta_1(i) + \zeta_3(i) + \zeta_5(i)$,

donde,
$\zeta_1(i) = V_m(1).\text{sen}(120.\pi.i/1000)$,
$\zeta_3(i) = V_m(3).\text{sen}(360.\pi.i/1000+\varpi_3)$,
$\zeta_5(i) = V_m(5).\text{sen}(360.\pi.i/1000+\varpi_5)$,

estas funciones provienen de la discretización de las señales,

$\zeta_1(t) = V_m(1).\text{sen}(2.\pi.f_1.t/1000+\varpi_1)$,
$\zeta_3(t) = V_m(3).\text{sen}(2.\pi.f_3.t/1000+\varpi_3)$,
$\zeta_5(t) = V_m(5).\text{sen}(2.\pi.f_5.t/1000+\varpi_5)$,

con frecuencia de muestreo 1000Hz y valores numéricos.

$V_m(1) = 1$, $\varpi_1=0$, $f_1=60$Hz, $f_3=180$Hz, $f_5= 300$Hz.

Se toman 35 muestras por señal $\zeta(i)$,este segmento contiene dos ciclos de 60Hz, seis ciclos de 180Hz y diez ciclos de 300Hz. Cada $\zeta(i)$ tiene distintos parámetros $V_m(3)$, $V_m(5)$, ϖ_3, ϖ_5 y sus correspondientes contenidos armónicos (3º y 5º).

La arquitectura de la red es 35-N-2, 35 neuronas de entrada correspondientes a las 35 muestras de la señal $\zeta(i)$, N neuronas ocultas y 2 neuronas de salida (una para cada contenido armónico). Un patrón para la red, durante el entrenamiento, consiste de 35 muestras de un $\zeta(i)$ y los correspondientes contenidos armónicos calculados con la fórmula (salidas deseadas).

Predicciones financieras

Una de las características de las redes neuronales es la aproximación de funciones nolineales, por lo tanto es factible utilizar una red para predecir el precio de las acciones en la bolsa de valores.

En el mercado de valores se conoce la hipótesis de los mercados eficientes que implica un comportamiento del precio de las acciones tipo random walk (la mejor predicción para el precio de mañana es el precio de hoy). La hipótesis del mercado eficiente afirma que es posible realizar predicciones de los precios de acciones tomando como base la información pública disponible como la historia de los precios y el volumen. Esto se puede explicar de la siguiente manera, cuando hay expectativa de un aumento de precio, los que participan en el mercado compran acciones elevando el precio al nivel esperado terminando con la oportunidad de beneficio que existía minutos antes.

Las redes neuronales pueden predecir el precio de las acciones si se utiliza la información del mercado yo además la información interna de la compañía, la red puede encontrar una relación nolineal entre las variables seleccionadas.

Procesamiento de imágenes

En la actualidad existen sistemas para adquisición de imágenes en tiempo real. Las aplicaciones militares requieren el procesamiento de esta información para identificar objetivos ubicados en distintos ángulos y posiciones.

En los Sandia National Laboratories científicos han desarrollado un sistema basado en redes neuronales para reconocer objetivos en distintas perspectivas, esta investigación busca identificar tanques entre edificaciones, vehículos y personal civil.

Durante las pruebas del sistema se ha logrado identificar los tanque en un 91% de los casos y en un tiempo máximo de 4s.

Verificación de firmas.

Una forma común de robo consiste en falsificar la firma de una persona para obtener dinero, el ojo humano no puede detectar los detalles finos de una falsificación. En el año de 1993 el fraude con tarjetas de crédito tuvo un costo de £ 50 millones en el reino unido. La compañía AEA Technology desarrolló un programa (red neuronal y reglas) que trabaja con formas capturadas electrónicamente, el sistema compara forma de la firma y la velocidad de escritura. Este sistema se encuentra en estudio experimental.

Monitoreo de cuidados intensivos

En el hospital St James en Leeds se ha estado trabajando durante los últimos años en el Neural Network Alarm Monitor (NNAM). El NNAM reemplazará monitores convencionales encontrados en las salas de cuidados intensivos, este dispositivo se ha refinado en un periodo de 5 años e incorpora un algoritmo especializado de entrenamiento y una red neuronal.

Un paciente en cuidados intensivos se monitorea por respiración, ritmo cardiaco, presión sanguínea, oxígeno y otras funciones vitales. En un monitor tradicional, las lecturas de los sensores se envían a la unidad que se encuentra en la pared , sobre la cabeza del paciente, y allí se presentan las funciones vitales con un procesamiento muy básico de estas. Este procesamiento se realiza con alarmas límite las cuales se activan si las funciones vitales se salen de los límites establecidos, los límites son ajustados manualmente por el personal médico.

El problema fundamental con el sistema tradicional es la falta de discriminación y una elevada rata de falsas alarmas. Cada función vital se trata separadamente, no se integran los datos y como consecuencia la rata de falsas alarmas crece con el número de funciones vitales monitoreadas.

El NNAM opera de manera opuesta al monitor tradicional. Con el incremento de funciones vitales monitoreadas aumenta su capacidad para discriminar y por lo tanto la rata de falsas alarmas disminuye. Este nuevo monitor integra las distintas entradas y produce una salida entre cero y uno la cual se interpreta como el grado de alarma. Si la salida excede un límite establecido por el personal médico se activa una alarma o un sintetizador de sonido.

Análisis de clientes

La compañía Central Research Laboratories ha desarrollado un sistema neural para ayudar a la compañía Radio Rentals en la selección de clientes.

El sistema neuronal se diseñó para mostrar la probabilidad de terminación de contratos por parte de los clientes. Este método ha mejorado su exactitud en un 30% comparado con métodos tradicionales como regresión lineal.

Para entrenar la red neuronal se utilizaron los registros de los clientes, el tipo de equipos, los estilos

de vida y la edad de las unidades rentadas.

IBM aplica el computo neuronal

La compañía IBM en Escocia ha desarrollado un sistema neuronal de inspección para detectar imperfecciones muy pequeñas para el ojo humano en circuitos impresos. Este sistema desarrollado por un grupo en el reino unido se introducirá en Japón y Estados Unidos.

La calidad de la soldadura es crítica en el ensamble de circuitos impresos para la industria de computadores. La tarjeta principal de un computador personal puede tener hasta 4000 puntos de soldadura y cuando se producen 5000 tarjetas diarias el control de calidad es un problema importante.

Análisis de gases

Cualquier compañía que produce emisiones de gases está sujeta a regulaciones ambientales, es vital por lo tanto que el contenido de los gases sea monitoreado continuamente para anticipar y prevenir el escape de compuestos tóxicos.

Los equipos tradicionales para monitoreo de gases son altamente especializados pues sólo pueden detectar un compuesto. Las compañías británicas Servomex plc y ERA technology desarrollan un equipo, basado en redes neuronales, capaz de analizar múltiples compuestos en un gas.

El equipo utiliza el espectro de absorción infra-rojo porque permite detectar mezclas de compuestos químicos en gases eficientemente y a bajo costo. El problema con el uso del espectro infra-rojo es la relación nolineal entre el espectro de absorción y las concentraciones de los compuestos, además siempre existe ruido en las mediciones.

Para entrenar la red neuronal se tomaron medidas de varias combinaciones conocidas de compuestos en un gas y sus respectivos espectros de absorción en el infra-rojo. Después del entrenamiento la red produce las concentraciones de los compuestos presentes en el gas de escape utilizando como entrada el espectro infra-rojo. La red es del tipo multicapa y fue entrenada con el algoritmo de propagación inversa, la presentación de la red es un programa el cual se almacena en una memoria y hace parte del equipo analizador de gases de escape.

Análisis de sonido

La industria de alimentos utiliza bombas para transportar productos líquidos entre equipos e proceso. Aunque el proceso es robusto, cuando ocurren bloqueos el producto que pasa por la bomba se puede dañar.

La Universidad de Brunel y Neural Technologies Limited han desarrollado un sistema para detección temprana de fallas en bombas utilizando una red neuronal. El sistema contiene una red neuronal 256-10-1 y un micrófono montado cerca de la bomba, la salida de la red activa una luz de alarma cuando se detecta un posible bloqueo. El sonido del micrófono se procesa antes de entregarlo a la red neuronal, la señal análoga del sonido se convierte a digital y luego se calcula el nivel de sonido en 256 bandas de frecuencia (espectro de frecuencia).

El computo neuronal en finanzas

Varias organizaciones financieras han encontrado donde aplicar el computo neuronal efectivamente.

El fondo de pensiones Deere & Company ha estado manejando un portafolio de US $100 millones desde diciembre de 1993 usando redes neuronales.

El fondo supervisa 1000 acciones en los Estados Unidos semanalmente, para cada una de las acciones hay una red neuronal que modela el futuro de la acción como función de 40 factores técnicos y produce un estimado del cambio de precio semanal. La compañía selecciona luego un portafolios de las 100 mejores acciones y distribuye los dineros del fondo proporcionalmente.

El programa de Deere & Compañía está soportado por 4 personas, empezó negocios en diciembre de 1993 y sus ganancias son del 30% lo cual es un buen margen según el estándar financiero.

Predicción de demanda

Britvic, una de las industrias líder de bebidas suaves en el reino unido, ha adoptado el computo neuronal para mirar el futuro del mercado de bebidas suaves.

Predecir volúmenes de ventas es importante en cualquier sector, una producción en exceso o un excesivo almacenamiento resulta en desperdicios de espacio mientras lo opuesto puede llevar a la pérdida de ventas o la demanda puede ser satisfecha por un producto rival.

El modelaje neuronal permite usar todos los datos disponibles e identificar relaciones sutiles entre las variables. Algunas variables consideradas por el grupo de mercadeo de Britvic que afectan la demanda son el nivel de empleo, publicidad, y el clima.

El sistema neuronal ha sido capaz de investigar el efecto de estos factores en las ventas y predecir el futuro del mercado de bebidas suaves con mayor exactitud que los métodod tradicionales.

Diagnóstico del cáncer de seno

El cáncer de seno es la mayor causa de muerte entre mujeres en el grupo de edad 35 a 55 años. Hay 15.000 muertes por año en el reino unido con 26.000 nuevos casos diagnosticados cada año.

El diagnóstico temprano mejora las posibilidades del tratamiento, y la revisión rutinaria se ha institucionalizado en muchos países incluyendo Norteamérica y el reino unido. La mamografía es la única técnica factible para examinar grandes números de mujeres. En la actualidad, cada mamograma (imagen en rayos x del seno) es analizado independientemente por dos expertos. Con la política actual de examen rutinario se producirán 3 millones de mamogramas para ser analizados cada año en el reino unido. Por lo tanto hay necesidad de un análisis automatizado que señale áreas de interés al radiólogo.

Se entrenó una red neuronal para aprender la condición de normalidad usando gran cantidad de mamogramas sin evidencia de estructuras tipo masa. La idea es probar novedad contra normalidad para identificar masas extrañas no vistas anteriormente.

REDES NEURONALES DINÁMICAS

La red neuronal multicapa o red tipo perceptrón es la más utilizada debido a su sencilla arquitectura y la disponibilidad de algoritmos de entrenamiento como propagación inversa y chemotaxis. Las redes multicapa son redes estáticas, esto es, producen un mapa de datos a datos por ello se utilizan

como aproximadores universales de funciones. Cuando se quiere sintetizar un comportamiento dinámico (algo que evoluciona en el tiempo) es necesario alimentar la red con muestras pasadas de las variables involucradas.

En sistemas de control en tiempo discreto o en predicción de series de tiempo se utilizan redes multicapa para producir modelos nolineales autoregresivos. En estas aplicaciones las redes multicapa se alimentan con muestras de la salida y la entrada retrasadas en el tiempo para crear un efecto dinámico. Las redes recurrentes tipo Hopfield (DNN-H: Dynamic Neural Network–Hopfield) se utilizan para aproximar la respuesta entrada-salida de sistemas dinámicos en tiempo continuo.

Las redes neuronales dinámicas tienen algunas ventajas cuando se comparan con las redes multicapa; por ejemplo:

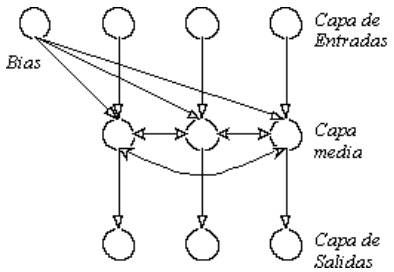
Las redes recurrentes para sistemas de una entrada-una salida requieren sólo el valor actual de la entrada en oposición a las redes multicapa que necesitan la entrada actual y la salida actual junto con varias muestras anteriores.
Las redes neuronales recurrentes obedecen ecuaciones diferenciales que se pueden convertir a ecuaciones de diferencias con cualquier periodo de muestreo.

REDES DE HOPFIELD

Jhon Hopfield, gracias al trabajo sobre neurofisiología en invertebrados, desarrolló un tipo de red neuronal autoasociativa.

Estas redes son bastante parecidas al modelo del Perceptrón, pero presentan una característica adicional, las neuronas en la capa media, presentan conexiones de salida hacia otras neuronas de la capa media.

Además, cada neurona de la capa de entradas esta conectada con una neurona de la capa media, y cada neurona de la capa media emite una sola conexión hacia la capa de salidas. Y estas conexiones (capa de entradas - capa media, y capa media -capa de salidas) no implican cálculo de pesos sinápticos ni de valores umbral.



El hecho de que todas las neuronas de la capa media se encuentren interconectadas, hace que en esta capa se dé un feedback o retroalimentación entre sus neuronas, de forma que al activarse una neurona de esta capa hace que las otras neuronas cambien su estado de activación, que a la vez harán cambiar el suyo propio. Esta nueva característica de las redes de Hopfield hace que funcionen de manera diferente al Perceptrón. Así, en el Perceptrón todas las neuronas de una misma capa transmiten el patrón de activación inmediatamente hacia las neuronas de la siguiente capa. (El patrón de activación sería un vector formado por los valores de las neuronas de una capa; ejemplo:

(0, 1, 0, 1, 1)). En una red de Hopfield las neuronas de la capa de entradas si transmiten inmediatamente su patrón de activación hacia las neuronas de la capa media (y además lo hacen sin variación debida a pesos sinápticos,...), pero las neuronas de la capa media no transmitirán ningún patrón de activación hasta que hayan llegado a un estado de equilibrio, en el cual el patrón de activación de la capa media se mantiene estable.

Funcionamiento de una red de Hopfield

Ejemplo. Red de Hopfield sencilla formada por 3 neuronas en la capa de entradas, 3 en la capa media y otras 3 en la capa de salidas.

Cuando se introduce un dato en la capa de entrada, este es transmitido sin variación hacia la capa media.

Una vez en la capa media, las neuronas que la forman modificaran su estado en función del estado de activación de las otras neuronas de la capa. Pero para esto, es preciso que las neuronas de la capa media no dejen de estar activas después de transmitir su estado a las otras neuronas (como sí sucede en el modelo del Perceptrón).

Además, las neuronas de esta capa media actualizan su estado de manera aleatoria: la neurona a la que le toca actualizarse es elegida al azar. De esta manera primero se actualizará una neurona elegida al azar, luego otra y así sucesivamente.
Todo el proceso de actualizaciones en la capa media se dará hasta que se alcance un equilibrio. Cuando se llega al equilibrio, los estados de las neuronas de la capa media ya no se modifican; se mantienen estables y, es entonces cuando la capa media transmite su patrón de activación a la capa de salidas, que lo recibe sin modificación alguna.

Ejemplo.

Los pesos sinápticos desde la neurona Bias hasta las neuronas de la capa media (neuronas 4, 5 y 6) valen:

W_{40}	W_{50}	W_{60}
1	0	0

Los pesos sinápticos entre las neuronas de la capa 5 valen:

$W_{45} = W_{54}$	$W_{46} = W_{64}$	$W_{56} = W_{65}$
-2	-4	3

Al simular lo que sucede al introducir el dato (1, 0, 1) en la capa de entradas, se tiene:

Bias (0)	Capa de entradas	Capa de salidas	Capa media			Neurona que se actualiza	$\sum W_{ij}X_j$	Nuevo estado de la neurona
			4	5	6			
1	**1, 0, 1**		1	0	1	*4*	-3	**0**
1			**0**	0	1	*4*	-3	**0**
1			0	0	1	*6*	0	**0**
1			0	0	**0**	*4*	1	**1**
1			**1**	0	0	*5*	-2	**0**
1			1	0	0	*6*	-1	**0**
1			1	0	0	*4*	1	**1**
		1, 0, 0	1	0	0			

Se observa que en este caso el equilibrio se alcanza cuando la capa media llega al valor (1, 0, 0); y no es hasta que se alcanza el equilibrio que el patrón de activación no es transmitido a la capa de salidas.

Estados de Equilibrio

Para hallar todos los posibles estados de equilibrio, es necesario analizar todas las posibles combinaciones de activación/inactivación que pueden presentar las neuronas de la capa media. Así, habrá que analizar 8 casos (2 posibles estados cada una de las tres neuronas: 23 = 8), suponiendo que en cada caso puede actualizarse de manera aleatoria cada una de las tres neuronas (probabilidad 1/3).

	Neurona que se actualiza	*Estado original de la capa media*	$W_{j0} + \sum W_{ij}X_j$	*Nuevo estado de la capa media*

Caso 1		0 0 0		
	1		1	1 0 0
	2		0	0 0 0
	3		0	0 0 0
Caso 2		0 0 1		
	1		-3	0 0 1
	2		3	0 1 1
	3		0	0 0 0
Caso 3		0 1 0		
	1		-1	0 1 0
	2		0	0 0 0
	3		3	0 1 1
Caso 4		**0 1 1**		
	1		-5	**0 1 1**
	2		3	**0 1 1**
	3		3	**0 1 1**
Caso 5		**1 0 0**		
	1		1	**1 0 0**
	2		-2	**1 0 0**
	3		-4	**1 0 0**
Caso 6		1 0 1		
	1		-3	0 0 1
	2		1	1 1 1
	3		-4	1 0 0

Caso 7		1 1 0		
	1		-1	0 1 0
	2		-2	1 0 0
	3		-1	1 1 0
Caso 8		1 1 1		
	1		-5	0 1 1
	2		1	1 1 1
	3		-1	1 1 0

Y se llega a la conclusión de que existen 2 estados de equilibrio: (0, 1, 1) y (1, 0, 0).

Cuando la capa media alcanza estos estados de equilibrio, nunca sale de ellos (ninguna actualización posterior de las neuronas de la capa media hace variar el patrón de activación). Así, la red del ejemplo siempre acaba en alguno de estos dos puntos de equilibrio.

Función de energía

Hopfield demostró que se podía definir una función de energía para cada posible patrón de activación o estado de la red ($T_i = -W_{i0}$):

$$E = -\frac{1}{2}\sum_i \sum_{j \neq i} W_{ij} X_i X_j + \sum_i X_i T_i$$

También demostró que el patrón de activación de la capa media evoluciona siempre hacia un estado de energía igual o inferior al estado de energía inicial; pero para ello han de cumplirse las siguientes condiciones:

El peso de la conexión entre una neurona i y otra j es el mismo que el peso de la conexión entre la neurona j y la neurona i:

$$W_{ij} = W_{ji}$$

No hay conexión de una neurona sobre si misma, o lo que es igual:

$$W_{ii} = 0$$

Además, si se cumplen las condiciones anteriores, también se cumple que la capa media es capaz de llegar a un equilibrio en el patrón de activación.

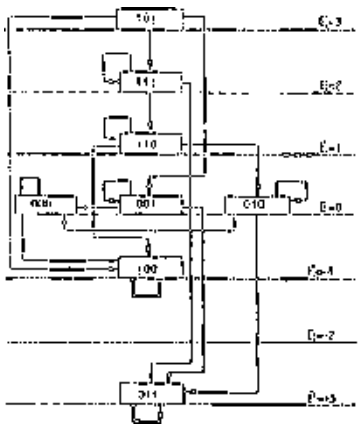
Y la función de energía para la red de Hopfield anterior sería:

$$E = -\frac{1}{2}(W_{45}X_4X_5 + W_{54}X_4X_5 + W_{46}X_4X_6 + W_{64}X_4X_6 + W_{56}X_5X_6 + W_{65}X_5X_6) + (X_4T_4 + X_5T_5 + X_6T_6)$$

Que considerando las equivalencias de pesos sinápticos quedaría:

$$E = -(W_{45}X_4X_5 + W_{46}X_4X_6 + W_{56}X_5X_6) + (X_4T_4 + X_5T_5 + X_6T_6)$$

Gracias a la función de energía se pueden definir los caminos que seguirá la red desde cada estado de activación:



Además, se observa que cada punto de equilibrio presenta áreas de atracción.

Estas áreas de atracción son el conjunto de estados de activación que con una probabilidad alta (mayor que 1/2) hacen que la capa media tienda a alcanzar uno de los puntos de equilibrio.
Así, calculando la probabilidad de llegar al punto de equilibrio (1, 0, 0) desde los 8 diferentes estados de activación, se obtienen las siguientes probabilidades:

Estado Inicial	1 0 0	0 0 0	1 1 0	1 0 1	0 0 1	0 1 1	1 1 1	0 1 0
Probabilidad	1	1	3/4	15/24	1/2	0	3/8	1/2

(0, 0, 0), (1, 1, 0) y (1, 0, 1) forman el área de atracción del punto de equilibrio (1, 0, 0), ya que si partimos de ellos la probabilidad de llegar a este punto de equilibrio es alta.

Interés práctico de las redes de Hopfield

Las redes de Hopfield pueden usarse como un modelo sencillo para explicar como ocurren las asociaciones entre ideas (o recuerdos) en las redes de neuronas del cerebro.

Así, una idea parcial sería un estado de activación que formaría parte del área de atracción de una idea más general, la cual actuaría como punto de equilibrio del área de atracción. De forma que al introducir la idea parcial en la red, se puede llegar a alcanzar la idea general (el equilibrio).

A su vez, debido a que las áreas de atracción indican sólo una probabilidad (generalmente diferente

de 1), este modelo permite explicar también la incertidumbre que se produce en las asociaciones; una idea parcial, a pesar de tener alta probabilidad de desembocar en la idea general, puede desembocar también en otras ideas diferentes (que actúen como puntos de equilibrio).

Una posible aplicación informática de las redes de Hopfield es el desarrollo de memorias direccionadas por contenido; los elementos de la memoria no estarían ordenados según índices numéricos, sino según parte de su contenido.

Así, en las memorias actuales cada conjunto de datos presenta asociada una dirección numérica (dirección de memoria), de manera que es necesario usar esta dirección para poder recuperar los datos asociados a ella. Mientras que las memorias basadas en redes de Hopfield permitirían que introduciendo datos parciales (que formen parte de un área de atracción) la memoria devolviera el conjunto de datos completo (equilibrio para el área de atracción).

OTRAS REDES NEURONALES

Además de las redes estáticas (perceptrón y multicapa) y de las redes dinámicas (DNN-H) existen otros tipos de redes con distintos algoritmos de aprendizaje y aplicaciones específicas.

RED DE KOHONEN

En pocas semanas un bebe aprende a asociar conjuntos de estímulos visuales con objetos y formas. En redes neuronales, el término auto-organización se refiere a la habilidad de algunas redes para aprender sin conocer la respuesta correcta para un patrón de entrada.

El algoritmo de propagación inversa es un algoritmo supervisado, esto significa que la red debe recibir como patrones entradas-salidas deseadas. La red de Kohonen es una red neuronal que extrae características de las entradas son conocer las salidas deseadas, es una red con aprendizaje no supervisado.

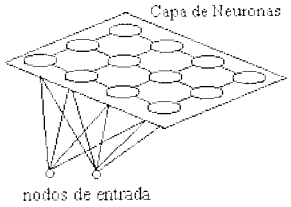
Kohonen es un profesor de la Universidad de Helsinki y ha trabajado en redes neuronales por muchos años incluso durante el periodo de hibernación científica de estos modelos. Kohonen ha trabajado con conceptos de memoria asociativa y con modelos de actividad neurobiológica.

Los neurobiólogos han demostrado que en el cerebro existen áreas donde se realizan funciones específicas como el centro del habla, el control de movimiento, el centro de visión. Cada área corresponde a un grupo de neuronas donde existe una intensa actividad local. Este hecho biológico ha llevado a la idea que el cerebro utiliza una proyección espacial para modelar estructuras de datos. La red de Kohonen utiliza esta idea para almacenar datos de tal manera que se preserven las relaciones espaciales existentes en los datos de una manera clara.

La mayor parte de la corteza cerebral esta organizada en un plano bidimensional de neuronal interconectadas pero puede manejar conceptos en espacios de mayor dimensión.

Un conjunto de datos se puede comprimir, por ejemplo, al codificar una imagen digital para reducir la necesidad de almacenamiento. La comprensión es la proyección de una entrada multidimensional a un espacio de salida de menor dimensión. Cuando trabajamos con datos los valores numéricos son la visión algebráica y la relación entre estos es la visión topológica. Una red de Kohonen es un mapa que preserva la topología de una representación multidimensional dentro de un arreglo

bidimensional de neuronas, la red proyecta señales similares a posiciones similares de neuronas.



En la red de Kohonen se nota que las neuronas no se encuentran originalmente organizadas en capas (nodos de entrada, neuronas ocultas, nodos de salida) como la red del tipo perceptrón. En el modelo de Kohonen las neuronas están organizadas en un plano y las entradas se conectan a cada neurona de la red. Existen conexiones laterales entre neuronas vecinas y no hay neuronas o nodos de salida, cada neurona del plano es una neurona de salida.

Algoritmo de entrenamiento

El algoritmo de aprendizaje para una red de Kohonen organiza las neuronas en vecindades locales que actúan como clasificadores de características de los datos de entrada. El mapa de neuronas se organiza de manera autónoma mediante un proceso cíclico que compara los patrones de entrada con vectores correspondientes a cada neurona. El algoritmo no requiere información sobre salidas deseadas.

El aprendizaje es no supervisado de tipo competitivo, las neuronas compiten por activarse y sólo una se activa cuando aparece un patrón en las entradas. Los pesos de las conexiones se actualizan según la neurona vencedora.

Al final del aprendizaje las neuronas forman grupos correspondientes a las características de los patrones.

El algoritmo es el siguiente:

1. Inicialización.

1.1 Escoger valores iniciales para los pesos asociados con cada neurona W_i^k (iteración cero, k=0), se recomienda que los W_i^0 sean distinto para i = 1,…, N, donde N es el número de neuronas. Una alternativa común es asignar a los vectores de pesos iniciales valores pequeños al azar o valores de algunos patrones de entrada. El vector de pesos asociado con la neurona i tiene componentes.

$$w_i^k = \left[w_{i1}^k \quad w_{i2}^k \quad \ldots \quad w_{in}^k \right]^T$$

n, es el número de entradas a la red.

1.2 Definir la vecindad de la neurona ganadora, un radio ρ permite seleccionar las neuronas cercanas al patrón de entrada dado en esa iteración.

2. Presentar el vector de entrada. Tomar un patrón x del conjunto de entrada con cierta probabilidad.

$$x_i^k = \left[x_1^k \quad x_2^k \quad \ldots \quad x_n^k \right]^T$$

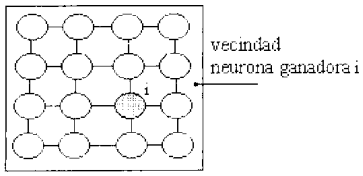
3. El aprendizaje es competitivo por lo tanto se determina la neurona con vector de pesos más cercano al vector de entrada. La distancia de Euclidiana d_i entre el vector de entrada y el vector de pesos de cada neurona es:

$$d_i = \left| x^k - w^k \right|, \qquad i = 1,2,...,N.$$

4. Actualizar los pesos de todas las neuronas.

$$w_i^{k+1} = \begin{cases} w_i^k + \mu_k . \left(x_k - w_i^k \right), i \in \mathbf{B}\rho \\ w_i^k, i \notin \mathbf{B}\rho \end{cases}$$

Bρ es la vecindad de la neurona ganadora.



Las neuronas cercanas a la neurona ganadora son aquellas que cumplen con la ecuación:

$$d_j \leq d_i + \rho$$

Donde,
j = índice de neuronas vecinas a la neurona ganadora.
i = índice de la neurona ganadora.
ρ= radio de la vecindad.

La rata de aprendizaje μ_k es dinámica y disminuye con cada iteración, algunas ratas de aprendizaje son:

$$\mu_k = \frac{1}{k}, \quad \mu_k = \mu_0 . \left(1 - \frac{k}{It} \right)$$

k = Número de iteración.
μ_k= 0.2
It= Máximo número de iteraciones.

5. Regresar al paso 2. si k ≤ It.

Resumiendo la operación de la red de Kohonen, cada neurona recibe el patrón de entrada y calcula la distancia con respecto a su vector de pesos. Después de calcular la distancia para cada neurona, estas compiten por el privilegio de aprender. La neurona con menor distancia es la ganadora, se activa tomando valor de uno y las demás permanecen en cero. Sólo la neurona ganadora y sus vecinos actualizan sus pesos.

Al final de un entrenamiento exitoso los vectores de pesos se distribuyen aproximadamente en el mismo patrón que los vectores de entrada. Los vectores de pesos están modelando la distribución de los patrones de entrada, esto se logra sin decirle a la red la distribución.

En la red de Kohonen no estamos interesados en la magnitud de los vectores de entrada sino en su orientación espacial, dos vectores son similares si apuntan en la misma dirección, sin importar sus magnitudes. La única forma para comparar la orientación de dos vectores, con una distancia Euclidiana, es normalizando dichos vectores. Normalizar un vector es reducir su magnitud a uno, esto se logra dividiendo cada componente del vector por su magnitud. Para un conjunto de vectores en el espacio Euclidiano esto significa que cada vector conservará su orientación pero tendrá una longitud constante.

En lenguaje matemático dado el vector $x_{\mathrm{i}} = \begin{bmatrix} x_1 & x_2 & \dots & x_{\mathrm{n}} \end{bmatrix}^T$, la magnitud está dada por:

$$|x| = \sqrt{x_1^2 + x_2^2 + \dots + x_n^2}$$

Y el vector normalizado es:

$$x = \left[\frac{x_1}{|x|} \quad \frac{x_2}{|x|} \quad \dots \quad \frac{x_n}{|x|} \right]^T$$

La magnitud de este vector es uno.

Los vectores inician normalizados y al entrenar la red cambia la normalización. Si tomamos pasos pequeños (pequeño,0.2) el ajuste de los pesos con la red de Kohonen es aproximadamente la rotación del vector de pesos y su magnitud es cercana a uno.

Aplicaciones

Calidad del aceite de oliva

El aceite de oliva tiene muy buen precio en el mercado y esto hace que muchos traten de adulterarlo para aumentar sus ganancias.

Se tomó una muestra de 572 aceites de oliva de 9 regiones distintas de Italia, para cada aceite se midieron 8 ácidos.

Se utilizó una red de Kohonen para clasificar 572 objetos con 8 características en 9 categorías. Algunas arquitecturas utilizadas fueron 8 – 10x10, 8 – 15x15, 8 – 17x17, 8 – 20x20. Después de entrenar la red, se puede verificar para un nuevo aceite si la región de donde proviene concuerda con su calidad y así controlar los aceites adulterados.

MEMORIAS ASOCIATIVAS

Las asociaciones son algo que nos ocurre a diario. Recordamos nombres cuando vemos rostros, alguien nos parece familiar porque trabajó o estudió con nosotros, vemos algo y nos trae imágenes, sonidos, y olores de inmediato. Formamos enlaces entre gente, eventos y lugares entre formas y objetos y conceptos, y esta habilidad nos permite construir nuestra propia representación del mundo. Las entradas a nuestros sentidos activan una cascada de asociaciones, cada una llamando la siguiente, un olor puede evocar el recuerdo de una tarde soleada en el campo lleno de flores, un parque puede evocar recuerdos de nuestra infancia. Es claro que la memoria humana funciona de manera asociativa, y en términos generales podemos decir que una memoria asociativa es un

sistema en el cual una entrada evoca una respuesta asociada.

La memoria asociativa es familiar para nosotros pues corresponde a la forma en que nuestra propia memoria funciona. Sin embargo, la memoria de un computador no funciona de esa manera. En un computador estándar cada trozo de información se almacena en una posición de memoria, y se lee cuando la dirección de la posición es conocida. Este almacenamiento local de información necesita alguna forma de decodificación para recuperar la información. En un computador la información se almacena en posiciones de memoria una tras otra, cada una en una dirección distinta.

La memoria asociativa funciona asociando una respuesta a una entrada particular; cuando presentamos una entrada, obtenemos la salida requerida. Una posible solución a este problema sería construir una lista de parejas entrada-salida, y cuando se presente una entrada el sistema busca en la lista la salida correspondiente, este enfoque es complicado y lento pues hay que recorrer la lista cada vez que se aparece una entrada. En lugar de este método, se puede considerar cada pareja entrada-salida como dos patrones y podemos asociarlos por una transformación del primero en el segundo. La memoria almacenaría solamente las transformaciones requeridas y no una lista de patrones entrada-salida. En otras palabras, consideramos la entrada y la salida como vectores, y los asociamos mediante una matriz que transforma el vector de entrada en el vector de salida.

En las memorias asociativas, las señales que corresponden a la entrada producen un conjunto de señales en la memoria que son la salida. Esto implica que la señal de entrada contiene toda la información necesaria para accesar el patrón almacenado, sin decodificación alguna. La idea de accesar la memoria con el contenido del patrón de entrada genera el nombre de memoria direccionable por contenido.

Es una memoria de computador el acceso a la información es inútil si no se conoce la dirección completa. En la memoria asociativa, es posible recuperar completamente la información con un patrón de entrada incompleto. Esta tolerancia a ruido en la entrada hace que las memorias asociativas tengan aplicación en el reconocimiento de patrones (imágenes y sonido).

Como no hay una correspondencia directa entre la entrada y la respuesta, no hay una localización de memoria que defina específicamente la salida, toda la matriz está involucrada. Una memoria con esta representación no localizada se denomina memoria distribuida. En una memoria distribuida, cada componente de la memoria tiene traza de los todos elementos almacenados, y sólo cuando se mira globalmente estas componentes de memoria forman un todo coherente. Debido al almacenamiento no-localizado de la transformación, en las memorias asociativas no hay partes críticas y toda la memoria es resistente al daño. Esto no es cierto en las memorias convencionales. Las memorias asociativas toleran fallas, en la matriz de la memoria o en los patrones de entrada, esto se debe al hecho que es todo el patrón el que importa y unos pocos errores se pueden ignorar.

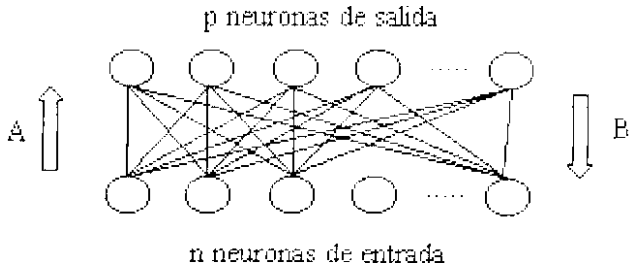
Hay dos tipos de memoria asociativa según la naturaleza de los patrones entrada-salida a asociar. Se puede asociar un patrón consigo mismo haciendo que el patrón de entrada y el patrón respuesta sean los mismos, entonces cuando se presenta un patrón incompleto a la entrada resulta en un llamado del patrón completo. Una asociación de esta naturaleza se denomina autoasociativa. Si el patrón de entrada se asocia con patrón de salida distinto, entonces la presentación de esta entrada causará que el patrón correspondiente aparezca en la salida, esta memoria se denomina heteroasociativa.

Memoria Asociativa BAM

Una memoria asociativa bidireccional (BAM) es una red neuronal de dos capas que se realimentan

entre si. Un patrón de entrada A es una cadena binaria (serie de 0s y 1s) o una cadena bipolar (serie de –1s o 1s), estas cadenas pueden ser la representación de una imagen bidimensional.

Cada neurona de la capa A está totalmente conectada a cada neurona de la capa B, no hay conexiones internas entre las neuronas de una capa. La información pasa desde la capa de neuronas A a la capa de neuronas B a través de la matriz de conexión M, luego la información regresa desde la capa B a la capa A a través de la matriz M traspuesta (M^T). Cuando las neuronas de la BAM se activan, la red evoluciona hacia un estado estable después de una reverberación entre las dos capas de neuronas.



En lenguaje matemático en la operación de la BAM, primero se presenta el patrón A y este se propaga a través de la matriz M para obtener el patrón B, $A \rightarrow M \rightarrow B$, luego el patrón B regresa a través de la matriz M^T, $B \rightarrow M^T \rightarrow A'$, el nuevo patrón A' se propaga a través de M, $A' \rightarrow M \rightarrow B'$.

el proceso se repite hasta que la red alcanza un punto de equilibrio dado por los patrones (A_i,B_i). La oscilación de un péndulo se puede asociar con el ir y venir de los patrones en una BAM.

Se puede demostrar que una BAM almacena los patrones (A_i,B_i), en los puntos mínimos de una superficie de energía. Cuando se presenta una entrada esta tiende a llamar el patrón en el mínimo de energía más cercano, la energía de una asociación (A_i,B_i) se define como $-A_i.M.B_i$.

Una memoria asociativa almacena gran cantidad de patrones complejos y puede clasificar nuevos patrones con respecto a los patrones almacenados de manera rápida. Cuando el número de patrones almacenados aumenta, la exactitud de la clasificación decrece. En un computador convencional el tiempo de búsqueda depende del número de patrones almacenados, por ello fallan en la búsqueda de patrones en tiempo real.

Aprendizaje en la memoria BAM

En una memoria asociativa toda la información del aprendizaje está contenida en la matriz de conexiones M. La BAM aprende un conjunto de asociaciones $\{(A_1,B_1), (A_2,B_2),\ldots, (A_m,B_m)\}$ sumando matrices de correlación, este es un ejemplo de aprendizaje con la ley de Hebb. El límite para el número de asociaciones que se pueden almacenar es menor a min(n,p), donde n es el número de neuronas en la capa A y p es el número de neuronas de la capa B.

En general, X_i e Y_i denotan la versión bipolar de los vectores A_i y B_i, un patrón binario A_i se convierte en bipolar con la expresión: $X_i=2.A_j-I$, donde I es un vector de componentes uno. El desempeño de la memoria asociativa es mejor con matrices y vectores bipolares.

Para entrenar la memoria asociativa se convierte cada par binario (A_i,B_i) en un par bipolar (X_i,Y_i) y luego se obtienen una matriz de correlación bipolar para cada par,

$X_i^T.Y_i$,

Luego se suman estas matrices de correlación para todos los pares que se desea almacenar en la memoria,

$$M = \sum_{i=1}^{m} X_i^T.Y_i$$

Es posible borrar una asociación (A_i,B_i) de M sumando $-X_i.Y_i$ a M.

Un caso especial de la memoria asociativa bidireccional ocurre cuando n = p y todos los casos $A_i = B_i$ (i = 1,…,m). Entonces $M = M^T$ y la memoria asociativa bidireccional se convierte en una memoria asociativa unidireccional o autoasociativa que almacena patrones A_i en mínimos de energía.

Ejemplo. Hallar la matriz M de una memoria BAM para almacenar las asociaciones binarias,

A_1=(1,0,1,0,1,0); B_1=(1,1,0,0)
A_2=(1,1,1,0,0,0); B_2=(1,0,1,0)

No se excede la capacidad de almacenamiento porque el número de patrones obedece, 2<min(6,4)

Solución. Se convierten los pares binarios a pares bipolares

X_1=(1,-1,1,-1,1,-1); Y_1=(1,1,-1,-1)
X_2=(1,1,1,-1,-1,-1); Y_2=(1,-1,1,-1)

Calculamos las matrices de correlación para cada patrón,

$$X_1^T.Y_1 = \begin{bmatrix} 1 & 1 & -1 & -1 \\ -1 & -1 & 1 & 1 \\ 1 & 1 & -1 & -1 \\ -1 & -1 & 1 & 1 \\ 1 & 1 & -1 & -1 \\ -1 & -1 & 1 & 1 \end{bmatrix}; \qquad X_2^T.Y_2 = \begin{bmatrix} 1 & -1 & 1 & -1 \\ 1 & -1 & 1 & -1 \\ 1 & -1 & 1 & -1 \\ -1 & 1 & -1 & 1 \\ -1 & 1 & -1 & 1 \\ -1 & 1 & -1 & 1 \end{bmatrix};$$

y se suman para obtener la matriz M de la memoria. El elemento m_{ij} es la intensidad de la conexión entre las neuronas a_i y b_j, teniendo en cuenta que $m_{ij}=m_{ji}$, esto es, M es simétrica. La sinapsis es excitadora si $m_{ij}>0$ e inhibitoria si $m_{ij}<0$

$$M = \begin{bmatrix} 2 & 0 & 0 & -2 \\ 0 & -2 & 2 & 0 \\ 2 & 0 & 0 & -2 \\ -2 & 0 & 0 & 2 \\ 0 & 2 & -2 & 0 \\ -2 & 0 & 0 & 2 \end{bmatrix}$$

La energía de la asociación es $E(A_1,B_1)=E(A_2,B_2)=-6$

El llamado de la asociación ocurre cuando un patrón A_i se presenta a la capa de neuronas A. Las n neuronas a_i en A se activan o desactivan según el patrón recibido y cada una entrega la salida a través de p caminos. El valor sinaptico m_{ij} multiplica la salida $a_i.m_{ij}$ y se activan o desactivan según la suma ponderada de entrada. Luego las neuronas b_j envían esta señal por las n conexiones a las neuronas a_i en A. Esto significa que B una la matriz M^T para envíar información, mientras A utiliza M, el proceso se repite hasta obtener convergencia, es decir, el patrón en la capa A y en la capa B es constante.

Algoritmo de entrenamiento y recuperación

Entrenamiento.
Inicialización. Para todo i,j borrar M(i,j), A(i), B(i)
Leer los vectores A() y B() correspondiente a las asociaciones que se almacenarán. Las entradas tomarán sólo dos valores {0,1}
Aprender la asociación:
3.1 Construir x(i) para A(i) donde X(i)=-1 si A(i)=0 y X(i)=1 si A(j)=1.
Construir Y(i) para B(i) donde Y(i)=-1 si B(i)=0 e Y(i)=1 si B(i)=1.
Para todo i,j, construya M(i,j)=M(i,j) + X(i).Y(j)
4. Si hay otra asociación para aprender, regresar a 2.
Recuperación

5. Leer de nuevo A() y B()
6. Ejecutar la iteracción $A \rightarrow B$ de la red:
6.1 El nuevo B(j) =1 si la suma de A(i), M(i,j) para todo i es mayor que cero (umbral).
6.2 El nuevo B(j)=0 si la suma de A(i), M(i,j) para todo i es cero
6.3 El nuevo B(j) no se modifica si la suma de A(i), M(i,j) para todo i es cero.
7. Ejecutar la iteración $B \rightarrow A$ de la red.
7.1 El nuevo A(i) =1 si la suma de B(j), M(i,j) para todo j es mayor que cero (umbral).
7.2 El nuevo A(i)=0 si la suma de B(j), M(i,j) para todo j es cero
7.3 El nuevo A(i) no se modifica si la suma de A(i), M(i,j) para todo j es cero.
8. Repetir los pasos 6 y 7 hasta que no cambie A() y B()
9. Presentar los resultados.

RED NEURONAL DE FUNCIONES RADIALES

El diseño de redes multicapa y su entrenamiento con el algoritmo de propagación inversa o chemotaxis se puede ver como la aplicación de un método de optimización. En la red neuronal de funciones radiales el diseño de la red se ve como un problema de ajuste o aproximación de una

superficie en un espacio de gran dimensión. Según este enfoque, el aprendizaje es equivalente a encontrar una superficie en un espacio multidimensional que proporcione el mejor ajuste a los datos de entrenamiento. Igualmente, la generalización de la red es equivalente al uso de la superficie multidimensional para interpolar los datos de prueba. En el contexto de una red neuronal, las unidades ocultas son un conjunto de funciones que forman una base arbitraria para los patrones de entrada cuando se expanden en el espacio de las unidades ocultas, estas funciones se denominan funciones radiales.

La construcción de una red de funciones radiales involucra tres capas distintas, la capa de entrada tiene los nodos fuente, la segunda capa es una capa oculta de alta dimensión con funciones radiales en las neuronas y la capa de salida entrega la respuesta de la red. La transformación del espacio de entrada al espacio de las unidades ocultas es nolineal, y la transformación del espacio de las unidades ocultas al espacio de salida es lineal.

HARDWARE EN LAS REDES NEURONALES ARTIFICIALES

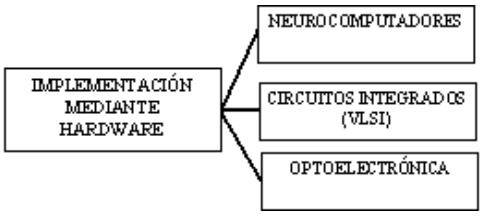
VISIÓN GENERAL

Cuando se habla de computación neuronal, normalmente se tiene en cuenta en mente las funciones sensoriales y motoras, así como algún tipo de proceso interno llamado pensamiento. Todas estas funciones son interdependientes entre sí, pero puede ser posible conceptualizar algunas de ellas de una forma idealizada.

Con respecto a las máquinas, no se trata de que compitan con los seres humanos, sino de que realicen ciertas tareas de rango intelectual, lo que supone un principio básico de la inteligencia artificial.

En cuanto a las diferentes formas de implementación se tienen:

Desarrollos de paquetes software.
Implementación hardware (Neurocomputadores, tecnología microelectrónica y tecnología electroóptica).



La diferencia entre circuitos integrados específicos y los neurocomputadores es el menor número de neuronas de los chips neuronales, a consecuencia del hecho de que dentro del propio chip se incluyen todas las interconexiones, y por tanto su velocidad resulta varios órdenes de magnitud superior que para los neurocomputadores.

Aunque la tecnología microelectrónica es actualmente la más adecuada para la realización de redes neuronales, existen problemas como la dificultad de obtener el alto grado de interconexionado propio de estas redes, o el problema de la entrada/salida masiva de datos, condicionada por el número de pines; además conseguir sinapsis con pesos variables, necesarias si se quiere retener información.

En cuanto a la tecnología optoelectrónica podría ser apropiada, utilizando la luz como medio de transporte de la información, permitiendo la transmisión masiva de datos.

REALIZACIÓN

Simulación de la red sobre un ordenador convencional mediante un software específico. Es la forma más simple. La mayor desventaja radica en el hecho de que se intentan simular redes con un alto grado de paralelismo sobre máquinas que ejecutan secuencialmente las operaciones.

Realización de redes orientadas a la ejecución de procesos con un alto grado de paralelismo, tales como redes de transputer, arquitecturas sistólicas, etc. El objetivo de tales redes es acelerar la simulación de la red neuronal permitiendo una respuesta en tiempo real; pero subsiste el hecho de que el comportamiento real de la red sigue siendo simulado por una estructura ajena a la estructura implícita de la red neuronal.

Realización mediante uno o varios circuitos integrados específicos. Se trata de construir un elemento o conjunto de elementos que se comporte lo más similarmente posible a como lo haría una red neuronal. Son los llamados chips neuronales. Estos funcionan a alta velocidad permitiendo el proceso en tiempo real, pero a costa de una pérdida notable de flexibilidad.

NEUROCOMPUTADORES

Definición

Un neurocomputador es básicamente un conjunto de procesadores conectados con cierta regularidad que operan concurrentemente.

Objetivo

Acelerar la simulación de la red neuronal, permitiendo una respuesta en tiempo real.

Procedimiento

El usuario define la red neuronal que quiere simular, y mapea dicha red sobre una serie de procesadores, de manera que cada procesador simula, en tiempos diferentes, distintas neuronas de la red. Cada procesador, en cada paso, recupera de su memoria local los pesos w_i y los estados w_j de las neuronas conectadas a la neurona que en aquel momento está simulando, realiza la suma de los productos y transmite el nuevo estado al resto de los elementos de proceso a través del bus de comunicaciones.

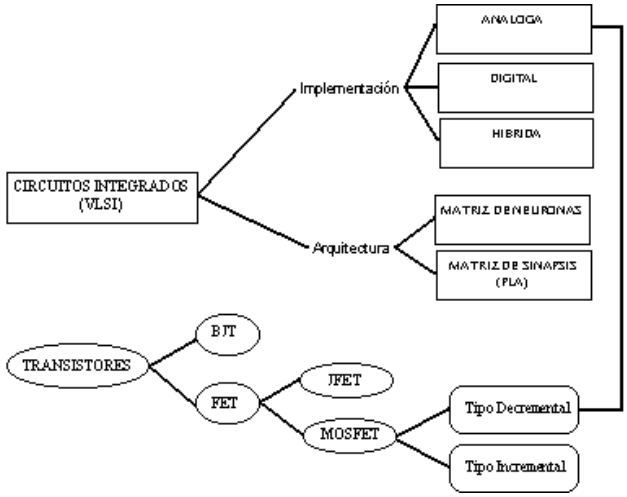
Clasificación

Neurocomputadores de propósito general.
Placas coprocesadoras.
Matrices de procesadores paralelos.

Neurocomputadores de propósito específico; diseñados para implementar un modelo específico de red neuronal.

TECNOLOGÍA VLSI

La tecnología VLSI de acuerdo al esquema, se enmarca en el siguiente contexto:



La ventaja de la implementación con circuitos integrados C.I. consiste en tener todas las interconexiones internamente, lo que incrementa la velocidad de procesamiento; por ello se busca tener el máximo número de componentes dentro del chip neuronal por lo que se hace una "integración a muy alta escala": VLSI.

En la VLSI se deben tener en cuenta parámetros de precisión, velocidad, consumo, inmunidad al ruido, memoria y capacidad de aprendizaje. Para ello, se analiza el tipo de implementación (si es análogo, digital o híbrido) y la arquitectura (matriz de neuronas o matriz de sinapsis PLA). Se trata la implementación análoga VLSI, que corresponde a la tecnología CMOS cuyos dispositivos son los MOSFET de tipo decremental, los cuales cuentan con una serie de ventajas apropiadas para la emulación de las redes neuronales biológicas.

Esta tecnología microelectrónica presenta la limitación del número de pines del chip para manejar los datos masivos de entrada y de salida. Una solución que se da es la tecnología optoelectrónica.

MOSFET Tipo Decremental

Los transistores son dispositivos electrónicos generalmente de tres pines, y que pueden cumplir con dos funciones, la de amplificación y la de conmutación.

Según el principio de funcionamiento se clasifican en bipolares, BJT (Transistores de Unión Bipolar) o en unipolares, FET (Transistores de Efecto de Campo) ; según la construcción de los FET´s estos se dividen en JFET (J = unión) y en MOSFET (MOS = Metal - Óxido - Semiconductor) ; en cuanto a los MOSFET, estos pueden ser de tipo decremental o de tipo incremental de acuerdo a la forma como operan.

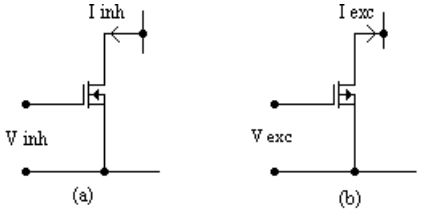
Cuando en una pastilla de silicio disponemos los MOSFET decremental en su tipo N y su tipo P se obtiene el arreglo MOS complementario también denominado CMOS, que corresponde a la técnica análoga de la VLSI.

Las ventajas obtenidas son las siguientes:

Impedancia alta de entrada debida principalmente a la capa de dieléctrico.
Alta velocidad de conmutación, de unos 20 nS.
Bajo nivel de energía de operación, entre 10-12 y 10-16 W.
Mayor inmunidad al ruido.
Ventana de salida más grande.
Estabilidad térmica más alta.

Principios

La emulación de una sinapsis se implementa mediante un arreglo CMOS; ambos transistores, canal N y canal P son usados como modelos de sinapsis inhibitoria y excitatoria, respectivamente . Se define la excitación como el aporte de cargas positivas al punto de unión (nodo), y la inhibición como el drenaje de cargas positivas desde el nodo.



Modelo para sinápis inhibitoria (a) y excitatoria (b)

En ambos casos los pesos sinápticos son simulados como una transconductancia. Para obtener la suma de las señales de entrada a una neurona (Net), se tiene en cuenta la ley de corrientes de Kirchoff, mediante la cual la suma algebraica de las corrientes de varios drenajes es igual a la corriente de entrada a la neurona. El almacenamiento de los pesos sinápticos se emula con la diferencia de voltajes entre dos compuertas (gates). Los cálculos propios de una red neuronal artificial son llevados a cabo mediante circuitos como los analizados en el laboratorio.

Otra consideración es respecto a la emulación de la función de activación; para ello tratamos al transistor BJT cuya curva de respuesta emula la función rampa. Una amplificador operacional emula la función paso. Un transitor FET o un par diferencial emula la función sigmoide también llamada tangente hiperbólica.

CONCLUSIONES

Aún con todas las investigaciones realizadas y con la tecnología existente, sigue sin entenderse el funcionamiento del cerebro, el cual es altamente complejo, por lo cual, es muy aventurado y pretencioso decir que una red neuronal artificial es un modelo representativo del cerebro. Se trata más bien de ir entendiendo y tratando de reproducir el funcionamiento de sus elementos fundamentales es decir, las neuronas.

Las redes neuronales son una tecnología computacional emergente que puede utilizarse en un gran número y variedad de aplicaciones; se pueden desarrollar redes neuronales en un período de tiempo

razonable y pueden realizar tareas concretas mejor que otras tecnologías convencionales, incluyendo los sistemas expertos.

El deseo que ha impulsado a través de los años a cientos de investigadores ha sido el crear una máquina a su imagen y semejanza, pero es muy pronto para llegar a una conclusión de esa índole, falta mucho camino por recorrer, lo único que se ha logrado con los avances que se han tenido hasta nuestros días, es que el sistema sea "menos tonto". En sí, estos sistemas no son capaces de hacer nada si nosotros no le indicamos detalladamente la acción a realizar y que si no se llega a contemplar una acción, el sistema podía llegar a fallar.

Los estudios empíricos muestran la eficiencia de las redes neuronales con datos basados en lógica difusa, patrones o rasgos ocultos para la mayoría de las técnicas incluida la capacidad humana, datos que exhiben no linealidad e información en la que se ha detectado caos.

No es posible considerar a una técnica de aprendizaje mejor que otra, ya que, dependiendo del área de aplicación y el resultado deseado, las conclusiones son distintas.

La gran mayoría de aplicaciones de las redes neuronales artificiales tienen en común la búsqueda y reconocimiento de patrones, para clasificarlos, completar una señal a partir de valores parciales o reconstruir el patrón correcto partiendo de uno distorsionado.

La ventaja de las redes neuronales es que ofrecen un método relativamente fácil para modelar las muestras complejas de datos donde se desconoce el formato. Por ello, son aplicadas efectivamente para el análisis de encuestas o solicitudes donde no todos los datos han sido proporcionados y; en estos casos, no compite con otras técnicas sino.

Se argumenta que una red neuronal imita el funcionamiento del cerebro humano; aunque es cierto que la metodología de las redes neuronales se basó, originalmente, en estudios sobre la operativa del cerebro para entender mejor los procesos de aprendizaje, no existen más similitudes. Por ejemplo, cuando un analista de crédito recomienda una decisión, no suele utilizar fórmulas matemáticas complejas para llegar a esta conclusión, como en el caso de las redes neuronales.

Lo único que se puede decir, es que en la actualidad, el control de cualquier proceso mediante un sistema inteligente es un reto para cualquier investigador; aunque, su implementación sea una incertidumbre para el futuro social y, tal vez sea un riesgo que algún día tengamos que correr.

BIBLIOGRAFIA

DELGADO Alberto. Inteligencia Artificial y Minirobots. Ecoe Ediciones. Santafé de Bogotá: 1998
BROWN D. & ROTHERY P. 1993. Models in Biology. Ed. Springer-Verlag. Berlín.
ALONSO G., BECERRIL J.L. 1993. Introducción a la inteligencia artificial. Ed. Multimedia Ediciones S.A. Barcelona.
CAUDAUDILLl Maureen. "Neural Network Primer: Part I". AI Expert, Feb. 1989.
Keller Paul. "Products That Use Neural Networks". Pacific Northwest National Laboratory. 1996.
Http://www.emsl.pnl.gov:2080/docs/cie/neural/products/
Http://www.alife.santafe.edu/
Http ://www.geocities.com/SiliconValley/Vista/7941/apren_c.htm