UN NUEVO ENFOQUE PARA LA RESOLUCION DE PROBLEMAS: LAS REDES NEURONALES

JONATAN GOMEZ PERDOMO*
FABIO GONZALEZ OSORIO*

RESUMEN

El presente artículo expone los lineamientos generales de un nuevo enfoque para la resolución de problemas mediante computador. Inspiradas en las redes neuronales biológicas, las **redes neuronales artificiales** han mostrado mayor efectividad para resolver problemas que los enfoques tradicionales no han solucionado o lo han hecho de manera parcial. En la actualidad este enfoque ha ganado un gran auge, lo cual ha generado en el país grupos de investigación que están trabajando sobre el mismo. Como parte de su trabajo en el grupo de la Universidad Nacional, los autores desarrollaron una herramienta para el diseño, entrenamiento y simulación de redes neuronales artificiales. Las características generales de esta herramienta se exponen al final.

* Ingenieros de Sistemas Universidad Nacional, Profesores Facultad de Ingeniería de Sistemas E.A.N.

I. INTRODUCCION

Tradicionalmente el enfoque utilizado para la resolución de problemas mediante el uso del computador, ha sido el enfoque algorítmico. Ello implica entender totalmente el problema en cuestión para poder construir un algoritmo que efectivamente lo pueda resolver.

Este enfoque se ha venido aplicando con éxito en diversidad de áreas, sin embargo hoy en día existen problemas cuya gran complejidad (tales como reconocimiento de texto manuscrito, reconocimiento de la voz, control de procesos, etc.) hace prácticamente imposible afrontarlos utilizando el enfoque algorítmico.

Las Redes Neuronales (RNs) surgen como un enfoque alternativo que ha mostrado ser exitoso para enfrentar dichos problemas, gracias a su robustez, flexibilidad y potencia para encontrar soluciones eficientes y eficaces.

Las RNs permiten resolver problemas que no se comprendan del todo gracias a que son capaces de aprender, lo cual les permite encontrar una solución al problema a través de ejemplos.

Supongamos que tenemos archivos en algún formato gráfico que representan fotografías de hombres y mujeres, deseamos desarrollar un programa que pueda decidir si una gráfico dado es la fotografía de un hombre o de una mujer, podemos imaginar la inmensa cantidad de factores que deberíamos tomar en consideración para poder diseñar un algoritmo que realizara dicha tarea, y además la cantidad de horas/programador que implicaría la implementación de dicho algoritmo.

Esta tarea la podríamos catalogar como 'casi imposible' de llevar a cabo mediante el enfoque algorítmico. Sin embargo la solución de dicho problema utilizando RNs, solo implicaría el diseño de una RN que tuviese como entrada la codificación en mapa de bits de la gráfica y dos salidas (una representando «hombre» y otra «mujer»), la preparación de patrones de entrenamiento para la red (archivos de ejemplo, los cuales deben estar ya clasificados en «hombre» y «mujer») y el entrenamiento de la RN utilizando dichos patrones.

Es importante notar que para llegar a esta solución no se necesita sino una comprensión global del problema, omitiendo los detalles complejos.

Como vemos las RN tienen características que las hacen sumamente atractivas para ser utilizadas en diversidad de problemas, pero estas no son las únicas características importantes que poseen.

II. CARACTERISTICAS DE LAS REDES NEURONALES

- Son capaces de manejar entradas con ruido, inexactas, probabilísticas y difusas. Para las RNs no se cumple el principio informático GIGO (Garbage In Garbage Out, basura entra basura sale), pues las RN son capaces de filtrar la información que manejan.

- Son robustas y tolerantes a fallas, la eventual falla de una o varias neuronas no implica un fallo total en la RN. Se han hecho experimentos con RN entrenadas a las que se les elimina un grupo de neuronas y conexiones, y su rendimiento no decae totalmente si no que se disminuye dependiendo del 'daño' total causado, como ocurre en los sistemas biológicos. Imaginémonos que sucedería si se le quita una o más líneas a un programa en C o PASCAL.

- Son flexibles, lo cual les permite adaptarse fácilmente a nuevos ambientes, gracias a su característica de aprender por ejemplos. Contrario a lo que sucede con el enfoque algorítmico el cual exige un nuevo diseño si las características del problema cambian.

Que son capaces de hacer las Redes Neuronales ?

Las RNs tienen una clara inspiración biológica, y son una analogía de las Redes Neuronales Biológicas presentes en el hombre y muchos animales. Por la tanto las RNs son buenas haciendo lo mismo para lo que son buenos los sistemas biológicos, el reconocimiento de patrones y la síntesis funcional, esto quiere decir que no debemos esperar usar una RN para llevar la contabilidad de una empresas o para invertir una matriz, problemas que el enfoque algorítmico ha podido resolver con éxito.

El reconocimiento de patrones, implica la clasificación de información según ciertas características, aplicaciones típicas son el ejemplo de las fotografías de hombres y mujeres que vimos anteriormente, la diferenciación de sonidos muy similares, el reconocimiento de escritura manuscrita, etc..

La síntesis funcional, consiste en establecer relaciones entre varias entradas continuas y una o más salidas continuas, ejemplos de aplicaciones son: estimar la demanda de un producto, filtrar el ruido a una señal, controlar una planta, etc.

Que es una Red Neuronal?

Ya hemos visto las características y lo que es capaz de hacer una RN, pero, que es en definitiva una RN?.

Desde el punto de vista informático, una RN es un sistema procesador de información que acepta entradas, las procesa y da salidas, el cual está compuesto de simples unidades procesadoras llamadas **Neuronas**, interconectadas entre si por **Conexiones**.

Una definición más formal cataloga una RN como un grafo dirigido cuyos nodos son llamados **Neuronas** y cuyos arcos son llamados **Conexiones**, cada neurona posee una función de transferencia que la caracteriza, y cada conexión posee un peso.

Estas definiciones pueden parecer complejas, pero cuando veamos como funcionan las RNs lo entenderemos mejor.

III. PRINCIPIOS BASICOS DE LAS REDES NEURONALES

Como habíamos dicho anteriormente las RNs tienen una inspiración biológica, de modo que para entender como funcionan debemos entender como funciona una neurona biológica.

En la figura 1 se observa una neurona biológica, las **dendritas** se encuentran conectadas al **cuerpo de la célula** o soma, donde el **núcleo** esta localizado. Extendiéndose desde el cuerpo de la célula se encuentra una larga fibra llamada el **axón**, el cual eventualmente ramifica en hilos cuyas terminaciones constituyen las **sinapsis** a otras neuronas. Los receptores de estas sinapsis en otras neuronas pueden ser las dendritas o el cuerpo de la célula en si mismo[1].

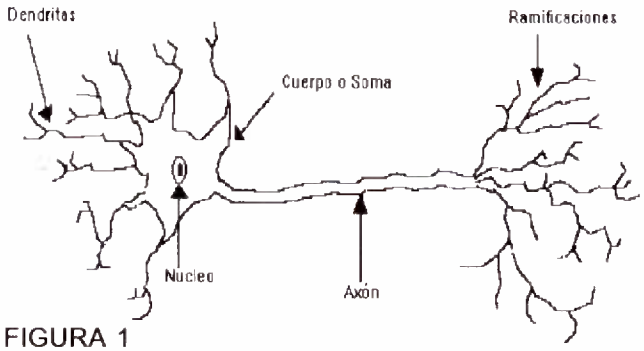


FIGURA 1

La transmisión de una señal desde una neurona a otra en una sinapsis es un complejo proceso químico. El efecto es aumentar o disminuir el potencial eléctrico dentro del cuerpo de la neurona receptora. Si este potencial alcanza un umbral, un pulso o señal es enviado por el axón hacia otras neuronas.

Un modelo de neurona binaria fue propuesto por McCulloch y Pitts[2] en 1943, el cual es el que básicamente se utiliza hoy en día, con ciertas modificaciones, en la construcción de RN artificiales.

Este modelo de neurona computa una suma ponderada de sus entradas desde otras unidades, y da como salida un uno (1) o un cero (0) dependiendo de si la suma ponderada está por encima o por debajo de un valor llamado el umbral.

Las ecuaciones que gobiernan el funcionamiento de dicho modelo de neurona son las siguientes:

$$n_i = £(\sum_j w_{ij} n_j (t) - u_i)$$

$$f(x) = \{1 \text{ si } x \geq 0,\ 0 \text{ de otra manera}\}$$

Donde:

- n_i : Salida de la neurona i.
- w_{ij} : Peso de la conexión entre las neuronas j e i.
- u_i : Umbral de la neurona i.
- f(x) : Función de activación.

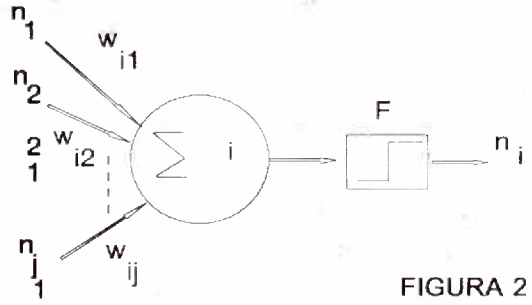


FIGURA 2
Neurona Artificial

Este es apenas uno de los tantos modelos de neurona artificial. Sin embargo, ilustra el funcionamiento básico de otros modelos, los cuales se diferencian en el tipo de función de activación (f(x)) que utilizan.

Aunque simple, este modelo de neurona artificial es un poderoso elemento procesador. Se ha demostrado que un conjunto de tales neuronas organizadas en una

[1] Axón: Filamento grueso que se desprende del cuerpo de la neurona.
Dendritas: Filamentos que se desprenden del cuerpo de la célula nerviosa.
Sinápsis: Punto donde se encuentran los colaterales de una neurona y las dendritas de otra neurona. En este punto se realiza el proceso quimico de transmisión de una señal de la primer neurona a la otra.

[2] McCULLOCH, Warren S. y PITTS, Walter. A logical calculus of the ideas immanent in nervous activity. Bulletyn of Mathematical Biophysics, 5. 1943. pp 115-133.

red (una red neuronal), es capaz de computación universal[3], es decir, que puede hacer todo lo que hace un computador digital, dada una disposición apropiada de las conexiones y con un peso apropiado para las mismas.

El punto principal en el campo de las RN , es cómo definir su arquitectura o topología y cómo asignarle peso a las conexiones, de modo que realice la tarea que se desea. Esta es la función de los algoritmos de aprendizaje o entrenamiento, los cuales a partir de ejemplos cambian la disposición de las neuronas, las conexiones y sus respectivos pesos, de tal forma que la red aprenda dichos ejemplos.

Los ejemplos (también llamados patrones) consisten en conjuntos de valores que describen un estado posible de la RN. En ciertos tipos de RN, estos valores pueden determinar una relación entre dos subconjuntos, denominados entradas y salidas, correspondientes a los valores que deben tomar las neuronas de entrada y salida respectivamente. Esto implica que a una RN a la que se le presente una entrada correspondiente a un ejemplo que ya aprendió, debe dar la salida que indica dicho ejemplo.

Gracias a su poder de generalización, una RN es capaz de responder de forma acertada frente a una entrada que nunca le fue enseñada.

En la actualidad, existen gran cantidad de algoritmos de entrenamiento o aprendizaje, pero el más ampliamente utilizado es el Back Propagation[4] o algoritmo de propagación inversa, el cual ha sido aplicado con gran éxito a diverso número de problemas.

IV. APLICACIONES DE LAS REDES NEURONALES

Los campos en los que las redes neuronales han sido aplicadas con éxito son innumerables y van desde la sintetización de voz, hasta la predicción de valores de acciones en la bolsa. Debido al espacio reducido solo vamos a nombrar unas cuantas aplicaciones que se han realizado.

El proyecto NetTalk consiguió entrenar una red para pronunciar texto en inglés, la RN fue entrenada con 1024 palabras y sus correspondientes fonemas. Después de entrenada fue probada con palabras que no estaban en el conjunto de entrenamiento y obtuvo un precisión del 78%, produciendo un vocalización bastante inteligible similar a la de un niño que está aprendiendo a hablar. Dañando la RN adicionando ruido aleatorio a los pesos de las conexiones, o removiendo algunas unidades, se encontró que degradaba su desempeño continuamente (no catastróficamente, como se esperaría en un computador digital).

Una RN fue desarrollada para reconocer códigos postales (ZIP) manuscritos del correo de los EE.UU. (U.S. Mail). Cerca de 10.000 dígitos fueron utilizados para entrenar y probar el sistema, estos dígitos fueron localizados en los sobres y segmentados en dígitos por otro sistema. La RN se entrenó utilizando BackPropagation, se utilizaron 7300 dígitos para el entrenamiento y 2000 dígitos de prueba, dando un porcentaje de error del 1% en el conjunto de entrenamiento y del 5% en el conjunto de prueba.

Otras áreas de aplicación son: visión por computador, control de procesos industriales, predicción de variables económicas, interpretación de encefalogramas, compresión de datos, identificación de estructuras químicas, conducción de automática de un automóvil, problemas de optimización, memorias asociativas, simulación de procesos, reconocimiento de la voz, procesamiento de señales[5].

V. PROYECTO UN_NEURO

Teniendo en cuenta la importancia de las RNs , en la Universidad Nacional de Colombia, en los departamentos de Matemáticas e Ingeniería (Sistemas y Eléctrica), se viene trabajando sobre las bases teóricas y sobre sus múltiples aplicaciones.

Parte de este trabajo fue el desarrolló de UN_Neuro[6], una herramienta amigable para el diseño, entrenamiento y simulación de RNs de manera gráfica. UN_Neuro es de utilidad para personas que investigan, para quienes desarrollan aplicaciones prácticas, o aun para quienes solo quieren explorar esta nueva tecnología.

UN_Neuro es una aplicación Windows, la cual se divide en tres módulos independientes: UN_NeuroDiseñador, UN_NeuroEntrenador y UN_NeuroSimulador, cuyo nexo es un archivo con un formato común para la representación de la RN.

[3] ARBIB, Michael A.. Cerebros, máquinas y matemáticas. 2a Ed. Madrid. Alianza Editorial S.A., 1982. pp 23-29.

[4] RUMELHART, D.E.. Et. al. Learning internal representations by error propagation. IN: Parallel Distributed Processing. Vols. I y II. U.S.A. MIT press, 1986. pp 318-361.

[5] HERTZ, John , KROGH, Anders y PALMER, Richard. Introduction to the theory of neural computation. U.S.A.. Addison-Wessley, 1991. 324 p.

[6] J. Gómez, y F. González. Prototipo de herramienta para el diseño, entrenamiento y simulación de redes neuronales, Tesis de grado Ingeniería de Sistemas, Universidad Nacional de Colombia. 1993.

UN_NeuroDiseñador

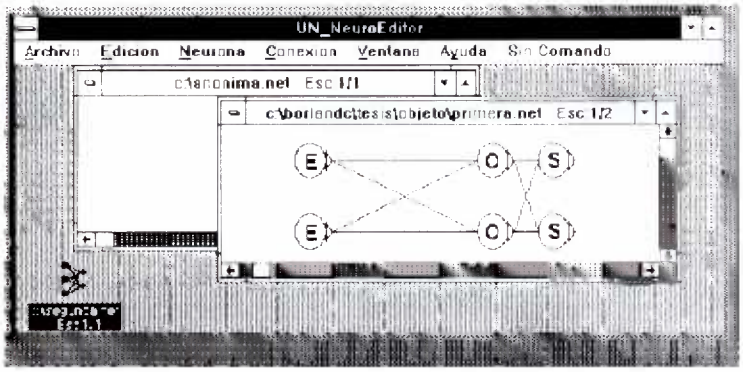
El módulo de diseño (el cual se puede observar en la figura 3), le brinda al usuario la posibilidad de:

- Crear una red neuronal con la estructura deseada, mediante la adición y eliminación de neuronas y conexiones.
- Consultar y modificar los atributos de las neuronas y de las conexiones.
- Almacenar la red neuronal diseñada en un archivo.
- Recuperar una red neuronal almacenada en un archivo.
- Editar varias redes neuronales a la vez.
- Manipular conjuntos de neuronas y conexiones (bloques), mediante funciones de copiar, mover, cortar y pegar.

Este módulo permite:

- Cargar una red neuronal para ser entrenada validando que su estructura sea correcta.
- Especificar el archivo de patrones con el cual se entrenará la red neuronal, validando que su estructura sea consistente con la de la red neuronal.
- Especificar un archivo de patrones diferente al de entrenamiento, con el que se calcula el error de generalización.
- Cambiar los parámetros del algoritmo de aprendizaje.
- Iniciar el proceso de entrenamiento y pararlo en cualquier momento.
- Guardar una red entrenada, o parcialmente entrenada, en un archivo de disco.
- Mostrar la evolución del entrenamiento mediante gráficas de los errores.

FIGURA 3



UN_NeuroEntrenador

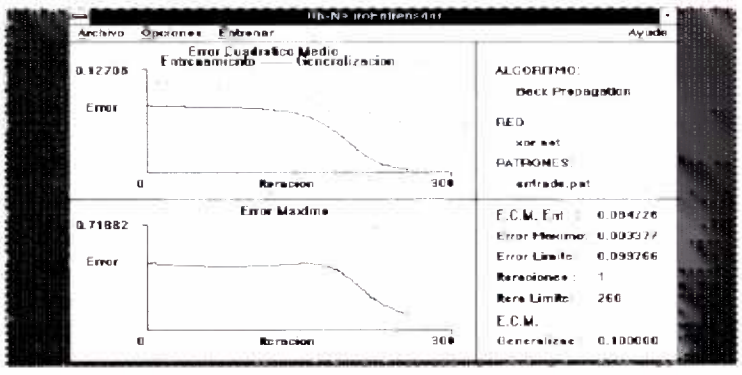
El módulo de entrenamiento (figura 4), le brinda al usuario la posibilidad de entrenar una red neuronal almacenada en un archivo de disco, con un conjunto de patrones también almacenados en disco. El algoritmo de entrenamiento utilizado por este módulo, es totalmente independiente de la aplicación, pues se encuentra en una librería de encadenamiento dinámico. Esto permite implementar nuevos algoritmos de entrenamiento sin modificar la aplicación.

UN_NeuroSimulador

En el módulo de simulación (figura 5), se puede observar el comportamiento de una red neuronal de manera gráfica, mediante las siguientes facilidades:

- Cargar una red neuronal, almacenada en un archivo de disco, para ser simulada.
- Cambiar y consultar valores de las salidas de las neuronas.

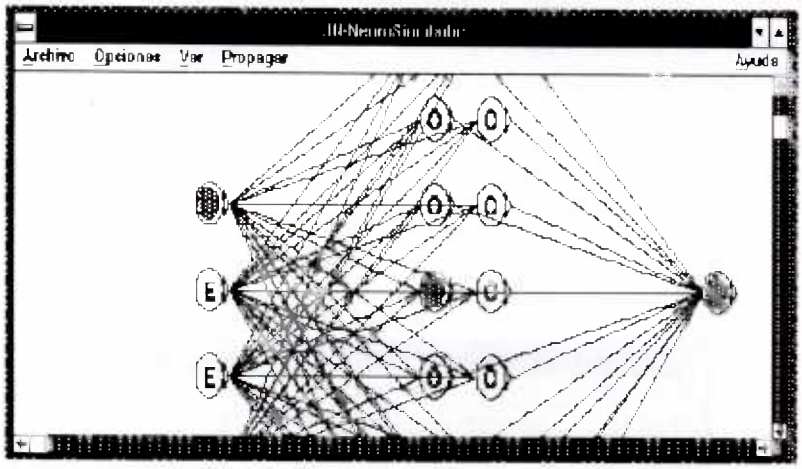
FIGURA 4



- Propagar dichos valores a través de la red neuronal.
- Observar gráficamente el comportamiento de la red neuronal.
- Simular la red neuronal utilizando un archivo con valores de entrada, y generando un archivo con los valores de salida (sólo para redes multicapa).
- Generar código en C++ que simule la red neuronal.

- Las RNs forman parte de una nueva tendencia en la computación, la Computación con Inspiración Biológica, para muchos esta tendencia se vislumbra como la posible siguiente generación de computadores, los recientes avances (Algoritmos Genéticos, Computación Evolutiva, Bio-chips) parecen confirmar esta creencia.

FIGURA 5



La herramienta posee un sistema de ayuda en línea, que utiliza hipertexto, para guiar al usuario en su utilización.

También cabe la posibilidad de implementar nuevos módulos para optimizar redes neuronales ya entrenadas, o para métodos de entrenamiento que no se adapten al módulo entrenamiento existente. Para esto, el desarrollador sólo debe conocer el formato del archivo utilizado para almacenar una red neuronal.

UN_Neuro funciona bajo ambiente MS_WINDOWS, lo cual le da una interfaz gráfica bastante amigable, con un Front-End estándar a muchas aplicaciones, además hace portable la aplicación a gran cantidad de plataformas.

UN_Neuro está desarrollado en Borland C++, utilizando una metodología de diseño y programación orientada a objetos. Con una estructura abierta y flexible que permite la fácil adición de nuevos tipos de RNs y de nuevos algoritmos de entrenamiento.

VI. CONCLUSIONES

- Las RNs han mostrado ser una excelente alternativa para la solución de innumerables problemas, sin embargo, por ser una tecnología relativamente nueva nos se ha explotado todo su potencial. Valdría la pena impulsar el desarrollo de la investigación sobre esta nueva tecnología en el país.

- UN_Neuro es una aplicación gráfica, fácil de utilizar, que proporciona herramientas para el trabajo con RNs. El objetivo de su desarrollo fue netamente académico, por lo tanto, esta disponible sin ningún costo, para el desarrollo de aplicaciones de carácter académico y para el apoyo de labores didácticas.

BIBLIOGRAFIA

E. Gelenbe (1992). Guest Editor's Introduction to the Special Issue on Neural Network Software and Systems, IEEE Transactions on SOFTWARE ENGINEERING, vol.18, pp.549-550.

J. Gómez, y F. González (1993). Prototipo de herramienta para el diseño, entrenamiento y simulación de redes neuronales. Tesis de grado Ingeniería de Sistemas. Universidad Nacional de Colombia.

J. Hertz, A. Krogh y R.G.Palmer (1991). Introduction to the theory of neural computation. Addison-Wesley publishing Company.

McCULLOCH, Warren S. y PITTS, Walter (1943). A logical calculus of the ideas immanent in nervous activity. Bulletyn of Mathematical Biophysics, 5. pp 115-133.

E. Mesrobian, y J. Skrzipek (1992). A Software Environment for Studying Computational Neural Systems , IEEE Transactions on SOFTWARE ENGINEERING, vol.18, pp.575-589.

D.E. Rumelhart, G.E. Hinton, y R.J.Williams (1986). Learning Internal Representations by Error Propagation. Parallel Distributed Processing, vol.1, cap.8, The MIT Press.