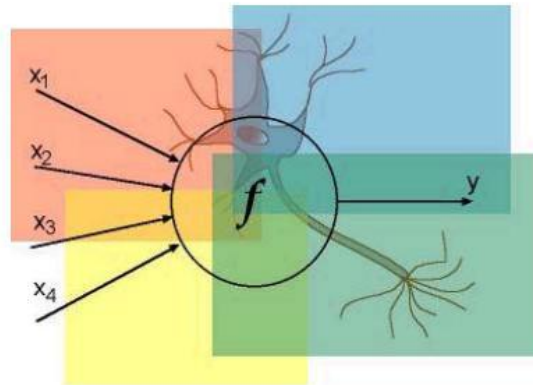


UNIVERSIDAD DE LAS CIENCIAS INFORMÁTICAS

**“TESIS EN OPCIÓN AL GRADO DE MÁSTER EN BIOINFORMÁTICA.”**

**Título:**

**“Algoritmo para reducir el tiempo de entrenamiento de un perceptrón multicapa sin afectar su eficiencia.”**



Autora:

Ing. Yuleidys Mejias Cesar

Tutor:

Dr. Ramón Carrasco Velar.

Ciudad de La Habana,

Cuba.

2010

***“TESIS EN OPCIÓN AL GRADO DE MÁSTER EN BIOINFORMÁTICA.”***

***Título:***

***“Algoritmo para reducir el tiempo de entrenamiento de un perceptrón multicapa sin afectar su eficiencia.”***

**Autora:**

**Ing. Yuleidys Mejias Cesar <sup>1</sup>**

**Tutor:**

**Dr. Ramón Carrasco Velar <sup>1</sup>**

<sup>1</sup> Universidad de las Ciencias Informáticas. Carretera a San Antonio de los Baños, Km 2½.  
Boyeros. Ciudad de La Habana. Cuba.

Autor para la correspondencia: <sup>1</sup> ymejias@uci.cu, <sup>1</sup> rcarrasco@uci.cu

Ciudad de La Habana,

Cuba.

2010

*Si me ofreciesen toda la sabiduría del mundo con la condición de guardarla para mí, sin compartirla con nadie, no la querría.*

*Lucio Anneo Séneca*

*Dedico esta tesis a mis padres y hermanos. La dedico además a mis compañeros del proyecto alasGRATO y a los del departamento de Bioinformática que siempre estuvieron apoyándome.*

*Pero por encima de todo, se la dedico a mi tutor Ramón Carrasco, por ser la mayor inspiración de esta investigación y sin el cual no hubiera podido lograr mi sueño de graduarme como máster.*

*Quiero agradecer primeramente a mi tutor, por brindar de forma desinteresada sus conocimientos y ayudarme a estar hoy aquí. A mis mamitas Mercy y María Rosa por darme las fuerzas que necesitaba y enjugar mis lágrimas cada vez que estaba triste. A mis papitos Miguel y Manza por apoyarme aún desde muy lejos. A mi maravillosa hermanita y mis hermanitos del alma por incitarme a ser el ejemplo que deben seguir.*

*Agradecer a todos aquellos que tuvieron que ver de una forma u otra con mi formación profesional, a mis amistades por estar conmigo en los buenos y malos momentos y por darme todo el apoyo que necesitaba para poder desarrollar esta investigación, principalmente a mis compañeras de cuarto y a los del proyecto alasGRATO. A Edel Moreno e Isbel Ochoa por su ayuda desinteresada, su paciencia y su amistad. A mis tesisistas José Carlos Pardo, Osniel Tortosa y Yuniel Valenzuela por contribuir fuertemente a la realización de mi maestría. A Yaikiel, Yania, Julio y Mario por su ayuda incondicional. A mis amigas más fieles Yanitza, Nina e Indy por su amor eterno. A Yudelkis, por ser mi amiga de madrugadas interminables y de momentos de desespero.*

*A mi Comandante en Jefe Fidel por darme la oportunidad de estudiar en una universidad de excelencia como esta.*

## **Resumen**

Dentro de los diferentes tipos de redes de neuronas artificiales (RNA) que existen, el perceptrón multicapa (PMC) es uno de los que mejores resultados ha aportado en los estudios de relación estructura-actividad. Se conoce que los volúmenes de datos previstos para procesar en el proyecto alasGRATO, son definitivamente grandes, por lo que se propuso evaluar algoritmos para acortar el tiempo del entrenamiento de la red sin afectar su eficiencia. Se evaluaron diferentes herramientas que trabajan con las RNA de las cuales se seleccionó el Weka para extraer el algoritmo del PMC y la Plataforma de Tareas Distribuidas Tarenal para distribuir el entrenamiento del mismo. Finalmente se desarrolló un algoritmo para el entrenamiento local y distribuido del PMC con la posibilidad de variar las funciones de transferencia. Se demostró que en dependencia de la muestra de entrenamiento, la variación de las funciones de transferencia pueden reportar resultados mucho más eficientes que los obtenidos con la función *Sigmoidea* (función de que se emplea generalmente con este tipo de red) con incremento de la G-Media entre el 4,5 y el 17%. Por otra parte, se encontró que en los entrenamientos distribuidos es posible alcanzar, eventualmente, mejores resultados que los logrados en un ambiente local.

**Palabras claves:** redes neuronales, perceptrón multicapa, funciones de transferencia, G-Media.

## **Abstract**

Among the different types of artificial neural networks (ANN) that exist, the multilayer perceptron (MLP) is generally the ones with the best results in structure-activity relationship studies. On the other hand, the structural information in any drug design project afford an extensive data volume, so that it was proposed to employ ANN algorithms to develop SAR and QSAR models without affecting the efficiency of the results. There were evaluated different tools as Weka and the Tarenal Platform for Distributed Task to distribute the training of MLP. Finally, it was developed a training algorithm for local and distributed training for the MLP with the possibility of varying the transfer functions. It was shown that in dependence on the sample, the change of transfer functions can improve results if compared with classic sigmoid function. In the experiments, the G-Media was increased in 4.5 to 17%. Moreover, it also was found that the classification results can be eventually improved with distributed training against local environment.

**Keywords:** G-Media, Multi-layer perceptron, neural networks, transfer functions.

---

Introducción .....	6
Capítulo 1: Fundamentación Teórica.....	13
1.1 El modelo biológico de la neurona. ....	13
1.2 Redes Neuronales Artificiales. ....	14
1.2.1 Breve reseña histórica. ....	16
1.2.2 La neurona artificial. ....	17
1.2.3 Definición de Red Neuronal Artificial. ....	20
1.2.4 Estructura de un Sistema Neuronal Artificial.....	21
1.2.5 Características de los Sistemas Neuronales Artificiales. ....	22
1.2.6 Ventajas de las RNA.....	23
1.2.7 Desventajas de las RNA.....	24
1.3 Clasificación de las Redes Neuronales Artificiales.....	24
1.3.1 Topología.....	24
1.3.2 Tipo de conexiones entre las neuronas. ....	25
1.3.3 Número de conexiones. ....	26
1.4 Tipos de Aprendizajes.....	26
1.4 .1 Aprendizaje supervisado.....	27
1.4 .2 Aprendizaje no supervisado.....	27
1.5 Tipos de redes neuronales artificiales. ....	27
1.5.1 Perceptrón Simple. ....	28
1.5.2 Perceptrón Multicapa. ....	28
1.6 Entrenamiento de la Red Neuronal Artificial.....	29
1.6.1 Selección de la arquitectura para el entrenamiento del perceptrón multicapa.....	29
1.7 Método de aprendizaje Backpropagation (propagación del error hacia atrás). ....	31
1.8. Aplicaciones de las Redes Neuronales Artificiales.....	34
1.9. Las Redes Neuronales aplicadas al diseño de nuevos fármacos. ....	36
1.9.1 Construcción de un modelo de predicción para la propiedad farmacocinética Excreción Urinaria en fármacos orgánicos: .....	36



---

1.9.2 Discriminación y predicción de propiedades de fármacos mediante redes neuronales:.....	36
1.9.3 Discriminación, predicción y diseño de nuevos fármacos aplicando métodos de inteligencia artificial:.....	36
1.9.4 Aplicación de métodos topológicos y de inteligencia artificial a la selección de nuevos antibacterianos:.....	37
1.9.5 Uso de una RNA para distinguir compuestos con actividad anticancerígena usando descriptores QSAR inductivos: (18) .....	38
1.10 Programas que trabajan con Redes Neuronales Artificiales. ....	38
1.10.1 Weka.....	38
1.10.2 Matlab .....	38
1.10.3 Joone (Java Object Oriented Neural Engine) .....	39
1.10.4 SNNS (Stuttgart Neural Network Simulator) .....	39
JavaNNS (Java Neural Network Simulator) .....	39
1.10.6 ALYUDA-NEURO-INTELIGENT: .....	40
1.11 Variación de las funciones de transferencia en las Redes Neuronales Artificiales...	40
1.12 Sistemas distribuidos. ....	41
1.12.1 Ventajas y desventajas de los Sistemas Distribuidos. ....	42
1.12.2 Condiciones que un problema debe cumplir para ser distribuible.....	43
1.12.3 Computación distribuida. ....	44
1.12.4 Entrenamiento distribuido de una RNA.....	46
1.13 Datos desbalanceados y medidas de evaluación de los algoritmos.....	47
1.13.1 La Matriz de Confusión .....	47
1.13.2 G-Media (g-means).....	48
1.13.3 Curvas de ROC .....	49
1.13.4 Área Bajo La Curva De ROC (AUC) .....	50
Conclusiones.....	50
CAPÍTULO II: PROGRAMAS Y PROCEDIMIENTOS.....	52
2.1 Weka (se explicó en la sección 1.10) .....	52

---

2.2 Plataforma de Tareas Distribuidas (Tarenal) (se explicó en la sección 1.12.3).....	52
2.2.1 Clases de la Plataforma Tarenal.....	52
2.3 Java como lenguaje de programación.....	53
2.4 Eclipse como herramienta IDE .....	54
2.5 El SPSS como software estadístico. ....	54
2.6 Procedimientos.....	54
2.6.1 Determinación de la arquitectura del perceptrón .....	55
2.6.2 Breve explicación sobre el entrenamiento del perceptrón multicapa. ....	55
2.7 Características de las muestras utilizadas. ....	55
2.7.1 Muestra 1: Cefalosporinas .....	56
2.7.2 Muestra 2: Inhibidores del Factor 1 del receptor esteroideogénico (SF-1). ....	56
2.7.3 Muestra 3: Reconocimiento facial.....	57
2.8 Reducción del espacio muestral.....	57
CAPÍTULO III: RESULTADOS Y DISCUSIÓN. ....	59
3.2 Resultados experimentales al ejecutar el perceptrón multicapa en una computadora personal.....	59
3.2.1 Resultados experimentales del entrenamiento del perceptrón con la muestra de cefalosporinas.....	59
3.2.2 Resultados experimentales del entrenamiento del perceptrón para la muestra SF-1.....	62
3.2.3 Resultados experimentales del entrenamiento del perceptrón para la muestra de reconocimiento facial. ....	66
Conclusiones .....	72
Recomendaciones .....	73
Referencias.....	74
Glosario de Términos.....	78

## **Introducción**

Desde la antigüedad el hombre se ha caracterizado por buscar constantemente nuevas vías para mejorar sus condiciones de vida. Sus investigaciones han estado dirigidas principalmente hacia la búsqueda de aquellas sustancias naturales que alivien sus dolores o sanen sus enfermedades. Estos estudios han permitido disponer de una gran gama de información, la cual ha dado lugar al surgimiento de diversas investigaciones y de nuevas vías para la producción de medicamentos.

En el proceso de obtención de fármacos, las moléculas candidatas deben atravesar dos fases fundamentales: descubrimiento y desarrollo. La primera fase va desde la síntesis, el aislamiento de la fuente natural, o la obtención biotecnológica y toda la fase pre-clínica, incluida la toxicología, con el fin de comprobar que el compuesto es eficiente y que puede probarse con seguridad en seres humanos. Se estima que en países con una amplia infraestructura investigativa todo este proceso dura aproximadamente 42.6 meses, o lo que es lo mismo, 3.55 años como promedio. En países con menos desarrollo podría demorar de 5 a 6 años (1). Por otra parte la fase de desarrollo comprende la de los estudios clínicos y la del registro farmacéutico, y se estima que dure entre 68.6 meses y 30.3 meses respectivamente.

Si calculamos el tiempo total que se requiere para obtener y registrar un fármaco llegamos a la conclusión de que se necesitan alrededor de 11.8 años de investigación. A esto se le debe sumar 231 millones de dólares, que representa el costo promedio por cada nuevo medicamento que salga al mercado.

Lo más alarmante es que sólo una de cada 10 000 moléculas ensayadas pasa a la fase de desarrollo, una de cada 100 000 supera los ensayos clínicos y logra registrarse y sólo 3 de cada 10 nuevos medicamentos registrados recupera su inversión inicial (1). Por tal motivo, en los últimos años las industrias farmacéuticas han reorientado sus investigaciones y prestado más atención a los métodos que permitan un diseño racional de nuevos compuestos.

La creciente sensibilidad social ante nuevos procesos patológicos como el síndrome de inmunodeficiencia adquirida (VIH-SIDA), el aumento de tasas de mortalidad por distintas enfermedades, como las cardiovasculares y el cáncer, son otros de los factores que proporcionan un ímpetu adicional en la búsqueda de nuevos métodos, que permitan acortar el tiempo de investigación-desarrollo de medicamentos.

Los primeros intentos dirigidos a incrementar la probabilidad de sintetizar un fármaco potencialmente efectivo se basaron en encontrar correlaciones entre la estructura química de una serie de compuestos y su actividad biológica, debido a la gran relación existente entre estos factores. Estos estudios permitieron identificar las características y propiedades estructurales causantes de la actividad biológica de un compuesto. De ahí surgieron las famosas siglas QSAR, acrónimo de Quantitative Structure-Activity Relationships.

En los últimos años se ha originado una auténtica revolución en el desarrollo de las técnicas QSAR, las cuales, aparejadas al acelerado desarrollo de las tecnologías de la computación y la programación, han dado paso a su vez a la posibilidad de enfrentar nuevos problemas dentro de esta nueva disciplina: el Diseño Racional de Fármacos.

En la actualidad todos los centros de investigación dedicados a la química medicinal y las compañías farmacéuticas, hacen uso de los métodos QSAR para diseñar y obtener productos más potentes, más específicos, con menos efectos colaterales y sobre todo más seguros, con un gasto mínimo de recursos. Una gran parte de sus investigaciones se han centrado en establecer estas relaciones con el empleo de técnicas de reducción de datos, métodos basados en árboles de decisión, técnicas de inteligencia artificial (IA), entre otras, con las que han logrado resultados positivos.

Por todo lo anterior es que resulta de gran interés disponer de una herramienta, que haciendo uso de estas técnicas sea capaz de predecir la actividad biológica de los posibles candidatos a fármacos, a fin de restringir el campo de búsqueda, privilegiando así las moléculas más prometedoras con lo cual aumentarían las posibilidades de éxitos y disminuirían los costos de la investigación.

Actualmente existen diferentes herramientas concebidas para el diseño racional de fármacos, la mayoría son propietarias y están generalmente orientadas a servicios. Entre ellas podemos citar a Insigth II, Accelerys, ChemFinder/BioViz Ultra 11, Apex-3D, sistema experto ADAPT (Automated Data Analysis using Pattern Recognition Toolkit), así como el OREX. Todas ellas se basan en diferentes técnicas de procesamiento de la información, en métodos químicos-cuánticos y en técnicas de Inteligencia Artificial.

En la práctica, existen aplicaciones que utilizan indistintamente las diferentes técnicas de IA. Cada una de ellas tiene sus propios seguidores y todas reúnen, virtudes y deficiencias. Sin embargo, la naturaleza de las relaciones entre la estructura química y la actividad biológica es,

matemáticamente hablando, una relación no-lineal. Para enfrentar relaciones de esta naturaleza existen diferentes enfoques. Entre ellos se destaca desde hace años, las Redes de Neuronas Artificiales (RNA). Las técnicas de aprendizaje de las RNA funcionan tanto como aprendizaje supervisado como no-supervisado. Tienen la ventaja sobre otras técnicas de IA que son capaces de extraer información estructural de una muestra dada de manera eficiente, simulando las interconexiones entre las neuronas del cerebro humano. Tienen la desventaja de que el entrenamiento de las mismas debe ser controlado para minimizar los problemas de memorización de la muestra, lo cual impediría la generalización de los resultados.

Desafortunadamente, la mayoría de estas herramientas mencionadas son de difícil acceso para nuestro país por sus altos precios, además de que no se pueden utilizar para resolver la necesidad del proyecto de disponer de dicha opción de predicción y modelación estructural de la actividad biológica en la Plataforma.

El presente trabajo tiene su origen dentro del proyecto CITMA de investigación-desarrollo que se ejecuta en la Facultad 6 de la Universidad de las Ciencias Informáticas, (UCI) denominado “Plataforma Inteligente para la Predicción de Actividad Biológica de Compuestos Orgánicos”. En el aspecto de la modelación de la actividad, las investigaciones actuales en el proyecto se están desarrollando para el establecimiento de diferentes modelos matemáticos que describan la actividad biológica de compuestos orgánicos, utilizando diferentes técnicas de IA. Para describir la estructura química se parte de dos enfoques diferentes: división de la molécula en fragmentos ponderados por el Índice del Estado Refractotopológico Total y descriptores moleculares de tipo topológico, topográfico e híbridos. Como técnicas de IA se emplean Lógica Difusa y Máquinas de Soporte Vectorial. No obstante, atendiendo a la naturaleza del problema y las capacidades de las RNA de extraer información no-lineal de un conjunto de datos, se decidió ampliar las investigaciones en este campo para incrementar las potencialidades de la Plataforma en la búsqueda de modelos eficientes de relación estructura-actividad biológica.

Al enfrentar inicialmente el estudio de las RNA, se identificaron dos interrogantes fundamentales. Una de ellas estaba relacionada con el “cómo” reducir el tiempo de entrenamiento de las redes y la capacidad de cómputo que consumen, más cuando se cuenta con grandes volúmenes de información, como es el caso del proyecto. La otra interrogante estaba asociada a las funciones de transferencia y cómo su combinación podría afectar la eficiencia de la respuesta durante el entrenamiento de la red. La función de transferencia, o de

activación, como también se le conoce, es uno de los elementos principales del funcionamiento de las RNA. La función de activación de una neurona artificial simula la respuesta de una neurona biológica ante un cierto estímulo. Se conoce que el cerebro posee un comportamiento diferente ante cada estímulo, sin embargo, en la gran mayoría de los casos las RNA son entrenadas y utilizadas haciendo uso de una única función de transferencia.

De ahí que **nuestro problema científico** sea ¿cómo reducir el tiempo de entrenamiento del perceptrón multicapa sin afectar su eficiencia? Para esto se plantea como **hipótesis de trabajo** que es posible reducir el tiempo de entrenamiento de un perceptrón multicapa sin afectar la eficiencia de la optimización. Teniendo como **objeto de estudio** las Redes Neuronales Artificiales y como **campo de acción** el perceptrón multicapa aplicado a problemas de predicción.

Por lo que se trazó como **objetivo general**: Proponer un algoritmo para disminuir el tiempo de entrenamiento del perceptrón multicapa sin afectar los resultados de la predicción.

Para dar cumplimiento al objetivo general se definieron los siguientes **objetivos específicos**:

- Disponer de algoritmos para realizar el entrenamiento local y distribuido del PMC.
- Identificar el efecto de la variación de las funciones de transferencia en la calidad del entrenamiento local del perceptrón multicapa.
- Identificar el efecto de la variación de las funciones de transferencia en la calidad del entrenamiento distribuido del perceptrón multicapa.

Para dar cumplimiento a los objetivos trazados se realizaron las siguientes **tareas**:

- Análisis del estado del arte respecto al uso del Perceptrón Multicapa en el diseño de fármacos.
- Estudio de las posibles funciones de transferencia a utilizar.
- Evaluación de arquitecturas de perceptrones multicapa con diferentes topologías para escoger la más eficiente.
- Selección de los ensayos en la base de datos del proyecto, con datos cualitativos, con el fin de realizar el entrenamiento.
- Estudio del funcionamiento de los Sistemas Distribuidos.
- Desarrollar un algoritmo para la distribución del perceptrón multicapa.

- Evaluar los resultados de un Perceptrón Multicapa (MLP) con una sola función de transferencia.
- Evaluar los resultados de un Perceptrón Multicapa (MLP) variando las funciones de transferencias lineales y no-lineales.
- Entrenamiento de la red neuronal con el 80% de la muestra.
- Realización de las pruebas de entrenamiento con el 20% restante.

Después de concluido este trabajo se podrá contar con el siguiente resultado:

Un nuevo enfoque para el entrenamiento de un PMC aplicado a la clasificación, así como un algoritmo para la distribución del entrenamiento de la red que permita disminuir el tiempo de entrenamiento del perceptrón multicapa sin afectar los resultados de la predicción.

El éxito de esta propuesta permitirá agilizar la etapa de desarrollo de los estudios de modelación para la búsqueda de nuevos fármacos, ya que a partir de la descripción de la estructura de una molécula, la red neuronal eficientemente entrenada, podrá predecir, la actividad biológica asociada a una molécula nueva a partir de sus descriptores.

El presente documento está estructurado de la siguiente manera:

### **Capítulo 1:** Fundamentación Teórica.

En este capítulo se hace una introducción al tema de las Redes Neuronales Artificiales, fundamentalmente al Perceptrón Multicapa, exponiéndose los conceptos matemáticos y arquitectónicos elementales de las mismas. Se muestran además algunos sistemas automatizados reportados en la literatura que se utilizan para resolver problemas de distribución del entrenamiento y de variación de las funciones de transferencia.

### **Capítulo 2:** Programas y Procedimientos.

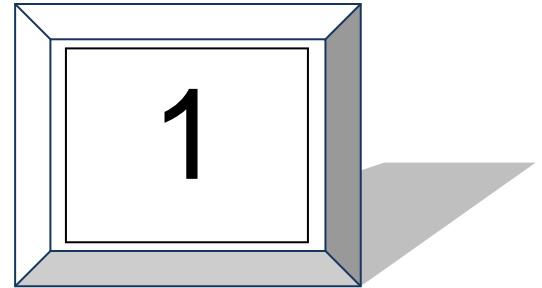
En este capítulo se describen las herramientas utilizadas en la tesis para desarrollar y evaluar el algoritmo del PMC local y distribuido, así como los pasos seguidos para desarrollar los diferentes experimentos. Finalmente se hace una breve descripción de las muestras utilizadas para realizar la investigación.

**Capítulo 3:** Resultados y Discusión.

En este capítulo se muestran los resultados experimentales obtenidos, tanto en los experimentos realizados en una computadora personal como los de la plataforma de tareas distribuidas. Además se realiza una comparación de los resultados (en ambiente local y distribuido) teniendo en cuenta el tiempo de ejecución del entrenamiento y la eficiencia de la respuesta del algoritmo.



Capítulo



# Fundamentación Teórica

**Capítulo 1: Fundamentación Teórica.**

En este capítulo se realiza un estudio de los principales aspectos de las redes neuronales que servirán para una mejor comprensión del documento. Se abordan temas tales como el funcionamiento, arquitectura y las diferentes clasificaciones de las redes neuronales artificiales. De la misma manera se describen sus ventajas, desventajas y principales aplicaciones. Finalmente se hace un estudio más profundo de las Redes Neuronales Artificiales aplicadas al diseño racional de fármacos, en el cual se hace referencia a diferentes trabajos consultados en la literatura. Se hace un estudio de algunos de los software que se utilizan para el trabajo con las RNA. Por último, se hacen algunos reportes sobre investigaciones realizadas para conocer el efecto de la variación de la función de transferencia en el funcionamiento de las RNA, y la distribución del algoritmo del PCM en aras de disminuir el tiempo de entrenamiento.

**1.1 El modelo biológico de la neurona.**

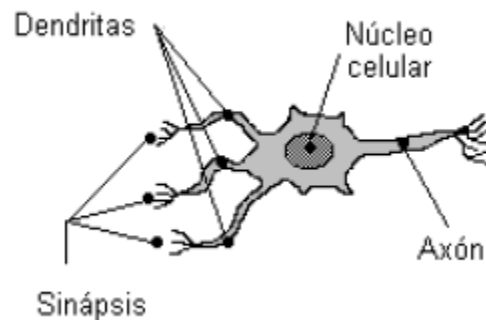
A finales del siglo XIX se pudo conocer mucho mejor el funcionamiento del cerebro gracias a los trabajos de Ramón y Cajal en España y Sherrington en Inglaterra. Ramón y Cajal trabajó en la anatomía de las neuronas; los estudios de Sherrington estuvieron enfocados a los puntos de conexión de las mismas o sinapsis. Se estima que en cada milímetro del cerebro hay cerca de 50.000 neuronas, conteniendo en total más de cien mil millones de neuronas y sinapsis en el sistema nervioso humano. (2)

La neurona es una célula viva, que se diferencia del resto de las células porque posee la capacidad de comunicarse. Es la unidad elemental del sistema nervioso y está compuesta por (3):

**Las Dendritas:** Son las conexiones de entrada de la neurona. Una prolongación del cuerpo de la célula nerviosa, que conduce el impulso nervioso hacia el cuerpo de la neurona.

**El Axón:** Es la salida de la neurona y se utiliza para enviar impulsos o señales 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. Esta unión puede ser inhibitoria o excitadora según el transmisor que las libere.

**La Sinapsis o saltos sinápticos:** Es la unión de dos neuronas. Un pequeño espacio que separa dos neuronas, y donde los impulsos nerviosos se transmiten del axón de la primera neurona a la dendrita de la segunda.



**Figura 1. Estructura Básica de una Neurona Biológica. (3)**

En sentido general, las dendritas y el cuerpo celular reciben las señales de entrada, el cuerpo celular las combina e integra y emite señales de salida en respuesta a estos estímulos. El axón transporta esas señales a los terminales axónicos, que se encargan de distribuir información a un nuevo conjunto de neuronas.

## 1.2 Redes Neuronales Artificiales.

Resulta contradictoria la idea de que las computadoras sean capaces de realizar cientos de operaciones por segundo, de resolver problemas matemáticos y científicos muy complejos, de crear y manipular grandes volúmenes de datos y que no sean capaces de entender el significado de las formas visuales o de distinguir entre distintas clases de objetos.

Esta dificultad de los sistemas de cómputo ha despertado la curiosidad para muchos investigadores, convirtiéndose en el centro de atención y estudio para muchos de ellos. Muchos de estos investigadores estudian el desarrollo de nuevos sistemas de tratamiento de la información, que permitan solucionar problemas cotidianos, tal como lo hace el cerebro humano.

El cerebro humano es, indiscutiblemente, una computadora extraordinaria capaz de interpretar a una velocidad increíble la información imprecisa suministrada por los sentidos. Logra

---

identificar un rostro en un local poco iluminado, identificar a una persona solamente por su voz o por una foto antigua. Pero lo más impresionante de todo, es que el cerebro aprende sin instrucciones explícitas de ninguna clase, a crear las representaciones internas que hacen posibles estas habilidades.

Basados precisamente en los procesos llevados a cabo por el cerebro, múltiples investigadores han desarrollado desde el año 1943, teorías sobre las Redes Neuronales Artificiales (RNA), las cuales se basan en el funcionamiento de las redes neuronales biológicas. Las RNA han sido utilizadas para aprender estrategias de solución basadas en ejemplos de comportamiento típico de patrones; estos sistemas no requieren que la tarea a ejecutar se programe, ellos generalizan y aprenden de la experiencia.

Las RNA han brindado una solución alternativa a problemas en los cuales los métodos tradicionales no han devuelto resultados muy convincentes. Se pueden encontrar resolviendo problemas de (4):

- Procesamiento de imágenes y de voz.
- Reconocimiento de patrones.
- Predicción.
- Control y optimización.
- Filtrado de señales.

Es importante conocer que los sistemas de cómputos tradicionales procesan la información en forma secuencial. Una computadora serial consiste por lo general de un solo procesador que puede manipular instrucciones y datos que se localizan en la memoria; el procesador lee, y ejecuta una a una las instrucciones en la memoria; este sistema serial es secuencial, todo sucede en una sola secuencia de operaciones. Por su parte, las RNA no ejecutan instrucciones, estas responden en paralelo a las entradas que se les presenta; el resultado no se almacena en una sola posición de memoria, sino que está distribuido por toda la red. El conocimiento de una red neuronal no se almacena en instrucciones, el poder de la red está en su topología y en los valores de las conexiones (pesos) entre las neuronas que la componen.

A pesar de todos los estudios realizados alrededor de las RNA, y de los potentes algoritmos de aprendizaje que se han desarrollado aún queda mucho por investigar. Tarde o temprano los estudios computacionales del aprendizaje de las RNA acabarán convergiendo a los métodos

---

descubiertos por la evolución, cuando eso ocurra, muchos datos empíricos concernientes al cerebro comenzarán a tomar sentido y se tornarán factibles muchas aplicaciones desconocidas de las redes neuronales.

### 1.2.1 Breve reseña histórica.

La evolución histórica de las redes neuronales artificiales se puede representar en tres períodos fundamentales:

**El período de surgimiento**, que se inicia a mediados de la década del 40 y termina a mediados de la década del 60. Es válido aclarar, que existieron trabajos anteriores que les abrieron el camino a las primeras investigaciones, tal es el caso del trabajo realizado por Karl Lashley en los años 20. En su trabajo de 1950 resume su investigación de 30 años; en ella destaca fundamentalmente que el proceso de aprendizaje no es un proceso local a una determinada área del cerebro, sino distribuido. Esta investigación fue el punto de partida de uno de sus estudiantes, D. Hebb, quien determinó una de las reglas de aprendizaje más usadas en la regla del conexionismo y que se conoce con el nombre de aprendizaje hebbiano. (5)

En 1943 McCulloch y Pitts proponen un modelo de neurona artificial que fue el que realmente marcó el inicio del primer período. Otros resultados relevantes de la primera etapa fueron el modelo Perceptron y el modelo Adaline, ambos desarrollados alrededor del año 60 y con características muy similares. El primero fue propuesto por Frank Rosenblatt y el segundo por Bernard Widrow.

**El segundo período es conocido como período oscuro** el cual se inicia en los años finales de la década del 60 y llega a inicios de los 80. En esta etapa se destaca fundamentalmente el libro publicado en 1969 por Minsky y Papert, titulado Perceptrons, donde se analizan las capacidades computacionales y limitaciones claves de este modelo. (6) En los estudios realizados pudieron comprobar que tales redes neuronales artificiales no constituían un modelo computacional de propósito general y sólo podía resolver problemas linealmente separables. Esta fue una de las principales causas que conllevó al ocaso de este período; lo que no impidió el desarrollo de la actividad investigativa, que sentó las bases para el tercer período.

En 1982 John Hopfield presentó un modelo de cálculo neuronal que se basa en la interacción de las neuronas (red de Hopfield). En su publicación "Neural Networks and Physical Systems with Emergent collective computational Abilities" (7) argumentó que hay capacidades

---

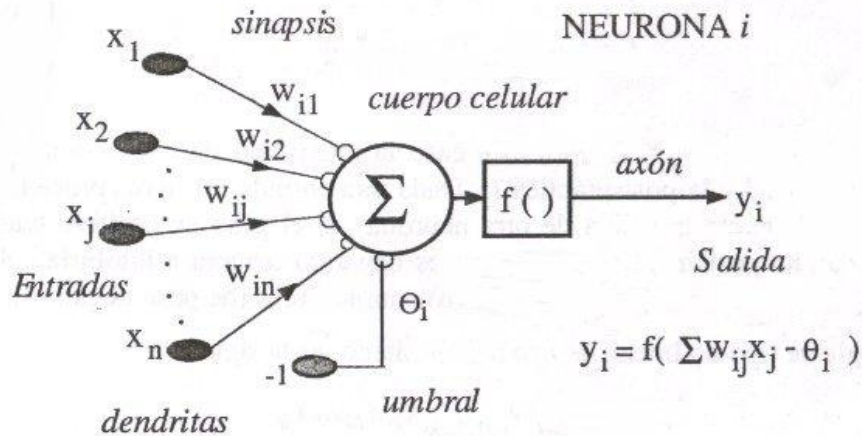
computacionales emergentes a nivel de la red que no existen a nivel de neurona. Este trabajo constituyó, sin lugar a dudas, uno de los puntos claves para el resurgimiento de las redes. En el mismo año de 1982 Kohonen publica un importante artículo sobre mapas auto-organizativos que se ordenan de acuerdo a unas reglas simples. El aprendizaje que se da en el modelo planteado no necesita de un “maestro”; se está ante un aprendizaje de tipo no supervisado.

En el año 1986 aparece un trabajo, que junto al desarrollado por Hopfield, despierta nuevamente el interés por las redes neuronales artificiales. Este trabajo fue el realizado por Rumelhart, Hinton y Williams, el cual consistía en el desarrollo de un algoritmo de retro propagación del error (backpropagation) para redes neuronales multicapa, dando una serie de ejemplos en los que se mostraba la potencia del método desarrollado (8). Estos descubrimientos marcaron el inicio del **tercer período, también conocido como el renacimiento**, el cual comenzó a mediados de los 80 y llega hasta la actualidad.

En 1987, se realizó la primera conferencia abierta sobre redes neuronales del I.E.E.E. (1700 participantes) fue realizada en San Diego. Ya para el año 1988 fue formada la Sociedad Internacional de Redes neuronales, la cual es seguida por la computación neuronal en 1989 y el *I.E.E.E. Transacción sobre Redes Neuronales* en 1990. A principios de 1987, muchas universidades anunciaron la formación de institutos de investigación y programas de educación acerca de la neuro-computación. En la actualidad, las RNA juegan un papel fundamental en el solución de diversos problemas, en todos los campos de la vida.

### 1.2.2 La neurona artificial.

La neurona artificial es la unidad básica de una RNA. Existen diferentes tipos de neuronas, pero la más común es la de tipo McCulloch-Pitts. En la siguiente figura se muestra la representación de la misma:



**Figura 2. Elementos claves de una neurona artificial de tipo McCulloch-Pitts. (9)**







Los elementos clave de una neurona artificial son los siguientes: (9)

- **Las entradas:** reciben los datos de otras neuronas. En una neurona biológica corresponderían a las dendritas.
- **Los pesos sinápticos  $w_{ij}$ :** como mismo ocurre en la neurona biológica, en la neuronal artificial se establecen conexiones entre las dendritas de una neurona y el axón de otra. En la neurona artificial, a las entradas que provienen de otras neuronas se les asigna un peso, el cual es un número que se modifica durante el entrenamiento de la red neuronal. Es en el peso donde se almacena la información que hará que la red sirva para resolver un problema u otro.
- **Una regla de propagación:** Haciendo uso de las entradas y los pesos sinápticos, se realiza algún tipo de operación definida para el modelo matemático de la red, para obtener el valor del potencial post sináptico (valor que es en función de las entradas y los pesos, el cual se utiliza en último término para realizar el procesamiento). La operación que se utiliza frecuentemente para obtener el valor del potencial post sináptico es la suma ponderada, la cual consiste en sumar todas las entradas  $x_j$  (donde  $j$  toma valores desde 1 hasta la cantidad de neuronas existentes en la capa que se esté analizando) teniendo en cuenta el peso sináptico de cada una,  $w_{ij}$  (donde  $i$  toma valor desde 1 hasta la cantidad de neuronas existentes en la capa siguiente). La otra regla de propagación más usada es la distancia euclidiana. Este es el tipo de regla que tienen redes como las RBF (Radial Basic Function) o el SOM (Self-Organizing Map).






- Una función de activación.** El valor obtenido con la regla de propagación, es utilizada para calcular la salida de la neurona. Esto se logra a partir de filtrar el valor obtenido con la regla de propagación usando una función conocida como función de activación o de transferencia. En dependencia del objetivo para el que se desee utilizar la red, se suele utilizar una función de activación u otra en ciertas neuronas de la red. Básicamente, las funciones de transferencia realizan dos tareas importantes: la primera, es que sirven para limitar el valor de la salida de una neurona, y lograr así que los resultados sean computacionalmente manejables y la segunda es que proporciona características de linealidad o no de la red.

En la siguiente tabla se muestran las funciones de activación más usuales.

**Tabla 1. Principales funciones de transferencia empleadas en el entrenamiento de redes neuronales artificiales. (10)**

Nombre (s)	Ícono	Relación Entrada (y) /Salida (x)
1. Hardlim (Limitador fuerte)		$a = 0 \quad n < 0$ $a = 1 \quad n \geq 0$
2. Hardlims (Escalón) (Limitador Fuerte Simétrico)		$a = -1 \quad n < 0$ $a = +1 \quad n \geq 0$
3. Poslin (Lineal Positiva)		$a = 0 \quad n < 0$ $a = n \quad 0 \leq n$
4. Purelin (Identidad) (Lineal)		$a = n$
5. Satlin (Lineal Saturado)		$a = 0 \quad n < 0$ $a = n \quad 0 \leq n \leq 1$ $a = 1 \quad n > 1$
6. Satlins (Lineal a tramos) (Lineal Saturado Simétrico)		$a = -1 \quad n < -1$ $a = n \quad -1 \leq n \leq 1$ $a = +1 \quad n > 1$



7. Logsig (Sigmoidea) (Sigmoidal Logarítmico)		$a = \frac{1}{1 + e^{-n}}$
8. Tansig (Tangente-Sigmoidea Hiperbólica)		$a = \frac{e^n - e^{-n}}{e^n + e^{-n}}$
9. Compet (Competitiva)		$a = 1$ Neurona con $n$ max $a = 0$ El resto de neuronas
10. Radbas (Radial Básica)		$a = e^{-n^2}$
11. Tribas (Triangular Básica)		$a = 1 -  n $ $-1 < n < 1$ $a = 0$ otro caso

De las funciones de transferencia que se muestran en la tabla 1 hay algunas destinadas a un uso específico, como por ejemplo: las funciones *radbas* y *tribas*, que se utilizan en Redes Radiales Básicas. (11)

En muchas ocasiones, la razón para aplicar una función de transferencia distinta a la función identidad, surge de la necesidad de que las neuronas produzcan una salida acotada. Desde un punto de vista de similitud con el sistema biológico, esto tiene sentido, ya que las respuestas de las neuronas biológicas están acotadas en amplitud. Además, cada neurona tiene asociado un número denominado bias o umbral, que puede verse como un número que indica a partir de qué valor del potencial post sináptico de la neurona se obtiene una salida significativa.

### 1.2.3 Definición de Red Neuronal Artificial.

Según Kohonen, las redes neurales artificiales son redes interconectadas masivamente en paralelo compuestas de elementos simples (usualmente adaptativos) y con una organización jerárquica, las cuales intentan interactuar con los objetos del mundo real del mismo modo que lo hace el sistema nervioso biológico. (12)

No existe una definición general u oficial de red neuronal artificial. Existen diferentes conceptos, de los cuales cabe citar los siguientes: (13)

- Una red neuronal es un modelo computacional, paralelo, compuesto de unidades procesadoras adaptativas con una alta interconexión entre ellas.

- Sistemas de procesamiento de la información que hacen uso de algunos de los principios que organizan la estructura del cerebro humano.
- Modelos matemáticos desarrollados para emular el cerebro humano.
- Sistema de procesamiento de la información que tiene características de funcionamiento comunes con las redes neuronales biológicas.

Lo real es que cada autor lo define a su manera, lo que siempre utilizando como factor común el componente de simulación del comportamiento biológico, así como la distribución de las operaciones a realizar entre las neuronas.

1.2.4 Estructura de un Sistema Neuronal Artificial.

Los sistemas neuronales artificiales se basan en la estructura hardware del sistema nervioso, con el objetivo de construir sistemas de procesamiento de información paralelos, adaptativos y distribuidos, que puedan representar un comportamiento “inteligente”. (16)

A continuación se representa la estructura jerárquica de un sistema basado en Redes Neuronales Artificiales.

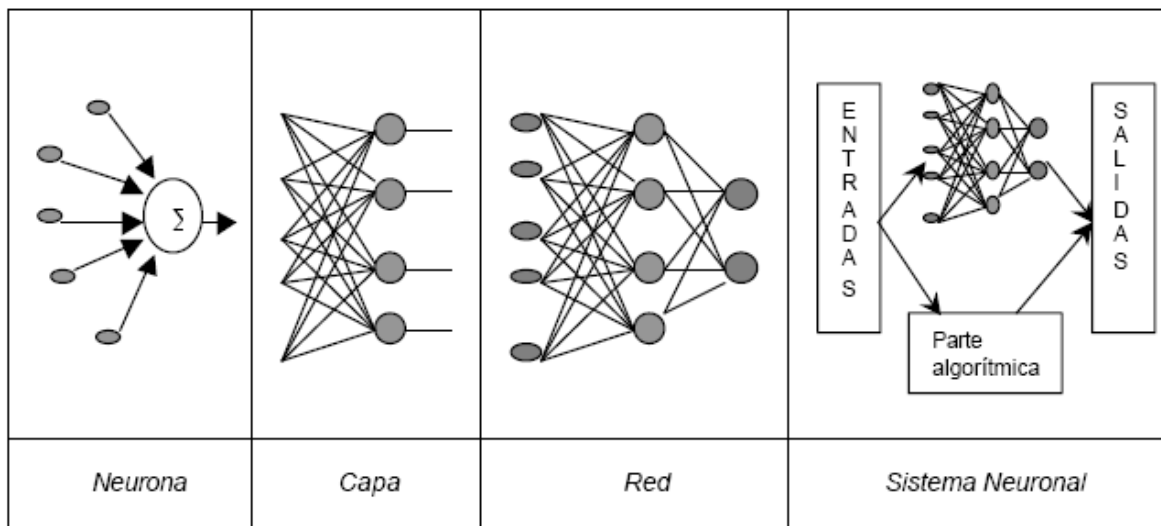


Figura 3. Estructura jerárquica de un sistema basado en Redes Neuronales Artificiales. (16)

Cada una de las neuronas realiza una función matemática. A su vez, las neuronas son agrupadas por capas, constituyendo así una red neuronal. Una red neuronal es entrenada para realizar una función específica. Una o varias redes, más las interfaces con el entorno conforman el sistema global.

Formalmente, un sistema neuronal artificial está compuesto por los siguientes elementos: (16)

- Un conjunto de procesadores elementales (neuronas artificiales).
- Una arquitectura.
- Una o varias funciones de transferencia o de activación.
- Una regla de aprendizaje.
- El entorno donde opera.

### 1.2.5 Características de los Sistemas Neuronales Artificiales.

Las RNA presentan un gran número de características similares a las del cerebro. Por ejemplo, son capaces de aprender de la experiencia, de generalizar a partir de casos anteriores a casos totalmente desconocidos, de abstraer características esenciales a partir de entradas que representan información irrelevante, entre otras.

A continuación se explican algunas de sus características más relevantes: (16)

**Memoria distribuida:** En las computadoras la información ocupa posiciones de memoria bien definidas. Sin embargo en las RNA la información se encuentra distribuida por las sinapsis, de manera de que si una de ellas es dañada no se perderá toda la información, sino una parte de ella.

**Aprendizaje adaptativo:** Las RNA son sistemas dinámicos auto-adaptativos. Esto se logra gracias a la capacidad que tienen las neuronas de auto-ajustarse. Se dice que son dinámicos debido a la capacidad que tienen de adaptarse constantemente a las nuevas condiciones que se le presentan.

Durante el proceso de aprendizaje, los enlaces ponderados de las neuronas se ajustan de manera que se obtengan los resultados especificados. La red neuronal no necesita un algoritmo para resolver los problemas, ya que durante el aprendizaje va generando su propia distribución de los pesos.

La posición del diseñador se limita a la definición de la arquitectura adecuada, así como de desarrollar un buen algoritmo de aprendizaje que le permita a la red discriminar mediante un entrenamiento con patrones.

**Auto-organización:** Las RNA utilizan su capacidad de aprendizaje adaptativo para organizar la información que reciben durante el aprendizaje. El aprendizaje consiste en la modificación de

los pesos, por su parte la auto-organización consiste en la modificación de la red neuronal completa para llevar a cabo un objetivo u otro.

Es la auto-organización la que proporciona la capacidad de generalizar, que no es más que responder adecuadamente cuando se le presentan datos nuevos, con los que no se habían entrenado la red. El sistema debe ser capaz de dar una respuesta “acertada” (con un margen de error) cuando se le presentan los nuevos datos. Esta característica permite que el sistema de una solución aún cuando la información de entrada está especificada de forma incompleta.

**Procesamiento paralelo:** Esta característica es de vital importancia ya que permite realizar las tareas más rápido que las computadoras. Las computadoras convencionales trabajan secuencialmente las instrucciones, lo cual requiere de mucho tiempo cuando la información a procesar es abundante. Por su parte las RNA ganan en tiempo, la clave está en todas las neuronas que intervienen en el proceso de análisis y procesamiento de la información, ya que estas operan en paralelo.

#### 1.2.6 Ventajas de las RNA.

A partir de las características de las RNA se pueden extraer sus ventajas. Entre ellas podemos citar: (14)

**Son sistemas distribuidos no lineales:** Una neurona es un elemento no lineal por lo que una interconexión de ellas (red neuronal) también será un dispositivo no lineal. Esta propiedad permitirá la simulación de sistemas no lineales y caóticos, lo cual le atribuye una gran ventaja frente a los sistemas clásicos lineales, ya que estos no poseen esa característica.

**Son sistemas tolerantes a fallos:** Una red neuronal, al ser un sistema distribuido, permite el fallo de algunas neuronas sin alterar significativamente la respuesta total del sistema. Esta característica las hace especialmente atractivas frente a las computadoras actuales que, por lo general, son sistemas secuenciales, de tal forma que un fallo en uno de sus componentes conlleva que el sistema total no funcione.

**Adaptabilidad:** Una red neuronal tiene la capacidad de modificar los parámetros de los que depende su funcionamiento de acuerdo con los cambios que se produzcan en su entorno de trabajo (cambios en las entradas, presencia de ruido, etc).

**Establecen relaciones no lineales entre los datos:** Las RNA son capaces de relacionar dos conjuntos de datos mediante relaciones complejas.

### 1.2.7 Desventajas de las RNA.

Existen algunos aspectos negativos de las RNA que hay que considerar al hacer uso de ellas. Entre ellos se pueden citar:

Elevada duración de los tiempos de entrenamiento.

El algoritmo de gradiente descendente puede alcanzar mínimos locales (es decir un mínimo en la función que relaciona los pesos con el error) más que un mínimo global.

El algoritmo de gradiente descendente también puede oscilar y nunca alcanzar el mínimo global.

Los valores de pesos que se aplican inicialmente y que suelen ser valores pequeños y aleatorios influyen en los resultados. (15)

### 1.3 Clasificación de las Redes Neuronales Artificiales.

Las RNA se pueden clasificar teniendo en cuenta varios aspectos tales como: la topología, número de conexiones, tipos de conexiones entre otros.

#### 1.3.1 Topología.

Desde un punto de vista topológico, las RNA pueden clasificarse en monocapas y multicapas. (40)

**Redes monocapas:** se establecen conexiones laterales entre las neuronas que pertenecen a la única capa por la que está formada la red. Las redes monocapa se utilizan generalmente en tareas relacionadas con lo que se conoce como auto-asociación; por ejemplo para regenerar informaciones de entrada que se presenta como distorsionada o incompleta. (Figura 4)

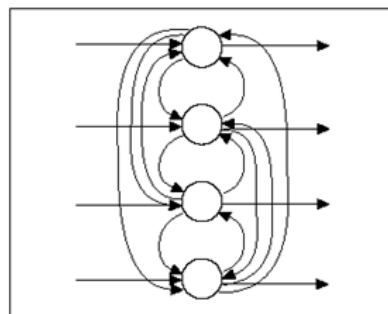
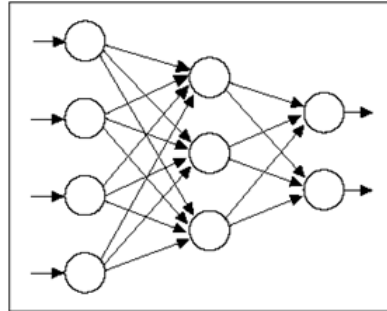


Figura 4. Red Monocapa.

**Redes Multicapas:** disponen las neuronas agrupadas en varios niveles. Debido a que este tipo de redes disponen de varias capas, las conexiones entre neuronas pueden ser del tipo feedforward (conexión hacia adelante) o del tipo feedback (conexión hacia atrás). (Figura 5)



**Figura 5. Red Multicapa.**

### 1.3.2 Tipo de conexiones entre las neuronas.

La conectividad entre las neuronas de una RNA está relacionada con la forma en que las salidas de las neuronas están canalizadas para convertirse en las entradas de otras neuronas. La señal de salida de una neurona puede ser la entrada de otra, o incluso de sí misma (conexión auto-recurrente). A continuación se muestran los tipos de conexiones reportados en la literatura: (18)

**Redes feedforward, o con conexiones hacia adelante:** Este tipo de redes contienen solo conexiones entre capas hacia adelante. Esto implica que una capa no puede tener conexiones a una que reciba la señal antes que ella. Un ejemplo de esta red se puede apreciar en la Figura 5.

**Redes feedback, o con conexiones hacia atrás:** Aparte del orden normal algunas capas están también unidas desde la salida hasta la entrada en el orden inverso en que viajan las señales de información. Este tipo de redes se diferencia de las anteriores en que sí pueden existir conexiones de capas hacia atrás y por tanto la información puede regresar a capas anteriores en la dinámica de la red.

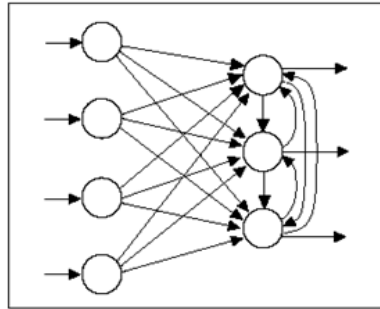


Figura 6. Red feedback.

**Redes feedlateral o con conexiones laterales:** Son conexiones entre neuronas de la misma capa, este tipo de conexión es muy común en las redes monocapa. Un ejemplo de red con este tipo de conexión se puede ver en la Figura 4.

### 1.3.3 Número de conexiones.

El número de conexiones está dado por la cantidad de conexiones que existen entre las neuronas de una capa y las de la capa siguiente. En este sentido se pueden encontrar dos grupos: las redes totalmente conectadas y las localmente conectadas.

**Redes neuronales totalmente conectadas:** Cuando la salida de una neurona en una capa  $I$  es entrada a todas las neuronas de la capa  $I+1$ . Un ejemplo de ello lo podemos ver en la figura 5). (39)

**Redes neuronales localmente conectadas:** Cuando la salida de una neurona en una capa  $I$  es entrada a un cierto número de neuronas pertenecientes a la capa  $I+1$ . (Figura 7)

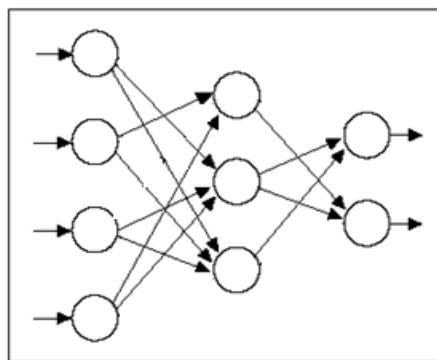


Figura 7. Red localmente conectada.

## 1.4 Tipos de Aprendizajes.

Es importante destacar que la propiedad más importante de las redes neuronales artificiales es su capacidad de aprender a partir de un conjunto de patrones de

entrenamientos, es decir, de ser capaz de encontrar un modelo que ajuste los datos de entrenamiento. En el proceso de aprendizaje, también conocido como entrenamiento de la red, se dice que la red ha “aprendido” cuando los valores de los pesos en función del tiempo se mantienen estables, o sea, cuando se logra que  $dw_{ij}/dt = 0$ . El entrenamiento de la red puede ser supervisado o no supervisado. (17)

#### 1.4 .1 Aprendizaje supervisado.

Consiste en entrenar la red a partir de un conjunto de datos o patrones de entrenamiento compuesto por patrones de entrada y salida. El objetivo del algoritmo de aprendizaje es ajustar los pesos de la red ( $w_{ij}$ ) de manera tal que la salida generada por la RNA sea lo más cercanamente posible a la verdadera salida dada una cierta entrada. Es decir, la red neuronal trata de encontrar un modelo al proceso desconocido que generó la salida. Este aprendizaje se llama supervisado pues se conoce el patrón de salida el cual hace el papel de supervisor de la red.

#### 1.4 .2 Aprendizaje no supervisado.

A diferencia del aprendizaje supervisado, en el aprendizaje no supervisado se presenta sólo un conjunto de patrones a la RNA, y el objetivo del algoritmo de aprendizaje es ajustar los pesos de la red de manera tal que la red encuentre alguna estructura o configuración presente en los datos.

Existe otro tipo de aprendizaje, que usa una fórmula híbrida, el supervisor no enseña patrones objetivos, sino que solo le dice si acierta o si falla en su respuesta ante un patrón de entrada. Este es conocido como *aprendizaje reforzado*.

### 1.5 Tipos de redes neuronales artificiales.

Existen diferentes tipos de RNA, entre ellas podemos citar:

- El Perceptrón Simple.
- La Red de Hopfield.
- El Perceptrón Multicapa.
- Red neuronal Competitiva Simple.
- Redes Neuronales Online ART1.
- Redes Neuronales competitivas ART2.



- Redes neuronales autoorganizadas: Mapas de Kohonen.

En la presente investigación se hace un estudio detallado del Perceptrón Multicapa, ya que es uno de los tipos de RNA que más popularidad ha alcanzado en la actualidad y de la que se reportan resultados satisfactorios en problemas de predicción.

### 1.5.1 Perceptrón Simple.

La red tipo perceptrón fue inventada por el psicólogo Frank Rosenblatt en el año 1957. El primer modelo de perceptrón fue desarrollado en un ambiente biológico imitando el funcionamiento del ojo humano. El fotoperceptrón era un dispositivo que respondía a señales ópticas. (10)

Se puede definir al perceptrón como una red neuronal no lineal que consta de una combinación lineal entre las entradas y los pesos sinápticos, un valor de ajuste externo (umbral o tendencia) y la función signo como función de transferencia si las entradas son bipolares  $\{-1,1\}$ , o bien la función escalón si las entradas son binarias. Inicialmente es un dispositivo de aprendizaje, en su configuración inicial no tiene la capacidad de distinguir patrones de entrada muy complejos, sin embargo mediante el proceso de aprendizaje puede adquirir esta capacidad.

El perceptrón simple tiene una serie de limitaciones muy importantes. La más importante es su incapacidad para clasificar conjuntos que no son linealmente independientes.

### 1.5.2 Perceptrón Multicapa.

Este modelo es una ampliación del perceptrón simple a la cual se le añade una serie de capas que, básicamente, hacen una transformación sobre las variables de entrada, que permiten eludir el problema anterior. Las capas presentan interconexión total.

Esto acaba con el problema del perceptrón, convirtiendo las funciones linealmente no independientes en linealmente independientes gracias a la transformación de la capa oculta. Su modo de propagación de los datos es hacia adelante (feedforward), es decir, que los patrones de entrada se presentan en la capa de entrada y se propagan hasta generar la salida. Las entradas y las salidas son continuas  $[0,1]$ .

La función de activación que generalmente se utiliza en sus neuronas es la sigmoidea.

Los perceptrones multicapa con aprendizaje backpropagation utilizan la regla Delta como forma de aprendizaje (esta regla de aprendizaje, se fundamenta en la utilización del error entre la salida real y esperada de la red para modificar los pesos). Estas redes adaptan la regla Delta

---

de tal forma, que se facilite el entrenamiento de todas las conexiones entre los distintos niveles de la red.

El perceptrón multicapa admite valores reales. Se puede decir que el perceptrón multicapa es un modelador de funciones universal.

### **1.6 Entrenamiento de la Red Neuronal Artificial.**

Se conoce que las RNA manejan dos tipos de información: la volátil y la no volátil. El primer tipo se refiere a los datos que se están usando durante el entrenamiento y varían con la dinámica de la computación de la red. Esta información se encuentra almacenada en el estado dinámico de las neuronas. En el caso de la información no volátil, esta se mantiene para recordar los patrones aprendidos y se encuentra almacenada en los pesos sinápticos. (19)

Como se vio anteriormente, el aprendizaje de las RNA, es el proceso de presentar los patrones a aprender, a la red y el cambio de los pesos de las conexiones sinápticas usando una regla de aprendizaje.

La regla de aprendizaje consiste en algoritmos basados en fórmulas matemáticas, que usando técnicas como minimización del error o la optimización de alguna "función de energía", modifican el valor de los pesos sinápticos en función de las entradas disponibles y con ello optimizan la respuesta de la red a las salidas que se desean.

El aprendizaje se basa en el entrenamiento de la red con patrones, que usualmente son llamados patrones de muestra o entrenamiento. El proceso del algoritmo consiste en que la red ejecute los patrones iterativamente, cambiando los pesos de las sinapsis, hasta que converjan a un conjunto de pesos óptimos que representen a los patrones lo suficientemente bien.

Es importante destacar que algunas redes no tienen un aprendizaje iterativo como el descrito en el párrafo anterior, de presentar los patrones una y otra vez hasta que la red sea capaz de dar resultados correctos, sino que los pesos de las sinapsis son calculados previamente a partir de los patrones, como es el caso de la red de Hopfield.

#### *1.6.1 Selección de la arquitectura para el entrenamiento del perceptrón multicapa.*

La arquitectura del perceptrón multicapa se compone de: el número de neuronas en la capa de entrada el cual depende del número de componentes del vector de entrada; cantidad de capas ocultas y número de neuronas de cada una de ellas; número de neuronas en

la capa de la salida, el cual depende del número de componentes del vector de salida o patrón objetivo.

Teniendo en cuenta que tanto el vector de entrada como el de salida están definidos por el problema a resolver no se hace difícil la determinación de la cantidad de neuronas en esas capas. Respecto al criterio a tener en cuenta para la selección de las neuronas de las capas ocultas surge una interrogante, este número en general debe ser lo suficientemente grande como para que se forme una región compleja que pueda resolver el problema, sin embargo no debe ser muy grande pues la estimación de los pesos puede ser no confiable para el conjunto de los patrones de entrada disponibles.

La mayor eficiencia del perceptrón multicapa en cuanto a su arquitectura se logra principalmente con una mayor experiencia su diseñador, aunque también existen varios criterios para realizar esta selección.

Para determinar la cantidad de capas ocultas:

**Sin capa oculta:** Sólo capaz de representar funciones linealmente separables o decisiones.

**Una capa oculta:** Pueden aproximar cualquier función continua desde menos infinito a más infinito.

**Dos capas ocultas:** Puede representar todas las funciones matemáticas conocidas hasta el momento.

**Tres capas ocultas o más:** No se ha encontrado justificación matemática para su uso. (35)

Decidir la cantidad de capas ocultas es una pequeña parte del problema, también se tiene que determinar la cantidad de neuronas por capa oculta. Esta selección es muy importante en la arquitectura final de la red neuronal, porque a pesar de no interactuar directamente con el ambiente externo tienen gran influencia en la salida final de la red. Ambos números, la cantidad de capas y la cantidad de neuronas en esas capas deben ser cuidadosamente escogidos.

Para determinar la cantidad adecuada de neuronas en las capas ocultas existen varios criterios:

1. La cantidad de neuronas ocultas debe estar entre la cantidad de neuronas de entrada y de la cantidad de neuronas de salida.
2. La cantidad de neuronas ocultas debe ser las dos terceras partes de la cantidad de neuronas de entrada más la cantidad de neuronas de salida.

3. La cantidad de neuronas ocultas debe ser menor que dos veces la cantidad de neuronas de entrada.

Esas tres reglas se pueden tomar como punto de partida para considerar este importante aspecto de la arquitectura de las redes neuronales artificiales. (35)

### **1.7 Método de aprendizaje Backpropagation (propagación del error hacia atrás).**

El método de aprendizaje backpropagation tiene como objetivo ajustar los pesos de la RNA para reducir el error cuadrático medio de las salidas. Este método pertenece a la categoría de supervisado, ya que requiere conocer las salidas correctas para cada ejemplo de entrada. En este método no es necesario dar información sobre las salidas de las unidades intermedias. El procedimiento de aprendizaje con backpropagation que se describe es utilizado para redes con una capa de unidades de entrada, cualquier cantidad de capas intermedias y una capa de unidades de salida. No existen conexiones entre unidades de una misma capa o de unidades situadas en capas posteriores a neuronas situadas en capas anteriores; pero sí pueden existir conexiones entre unidades situadas en la capa  $i$  hasta las situadas en la capa  $j$ , siempre que  $j > i$ . Existen pesos  $w_{ij}$  asociados a las conexiones. (20)

El procedimiento de aprendizaje ejecuta dos pasadas a través de la red. Durante la pasada hacia adelante se aplica un patrón de entrada a la red con sus pesos actuales (inicialmente pesos aleatorios pequeños, preferentemente en el rango de -0.1 a 0.1). Las salidas de todas las neuronas en cada nivel se calculan comenzando a partir de la capa de entrada y trabajando adelante en dirección a la capa de salida. La salida real de la red se compara con la salida deseada y se calcula el error cometido. Seguidamente se ejecuta una pasada hacia atrás en la cual la derivada del error se propaga hacia atrás a través de la red y todos los pesos son ajustados en proporción a su responsabilidad en el error de la salida. Esto se repite para todos los patrones ejemplos disponibles hasta que la red produzca la salida deseada o se cumpla una condición de parada especificada que puede ser número de iteraciones o el error relativo máximo deseado.

En este proceso juega un papel fundamental el momentum y la velocidad de aprendizaje. Sin momentum una red puede estancarse en un valle o mínimo local. El valor del momentum oscila generalmente entre 0 y 1. La velocidad de aprendizaje representa los saltos en el algoritmo de disminución del gradiente, también puede constituir un problema, ya que si el valor el muy

pequeño, puede ser muy lento el proceso y si es muy grande puede hacer que la red presente inestabilidad en el proceso de búsqueda o de aprendizaje. (38)

El algoritmo de aprendizaje procede de la siguiente manera:

Se presenta un vector de entrada a la red. Luego los estados de las unidades de cada capa son determinados aplicando las ecuaciones (1) y (2) a las conexiones provenientes de las capas anteriores. La entrada total  $X_j$  a la unidad  $U_j$  es una función lineal de las salidas  $Y_i$  de las unidades que están conectadas a  $U_j$  y de los pesos  $W_{ij}$  sobre estas conexiones (el valor de  $j$  va desde 1 hasta la cantidad de neuronas en la capa que se esté analizando y el valor de  $i$  va desde 1 hasta la cantidad de neuronas de la capa siguiente a la que se esté trabajando). Para cada unidad  $U_j$  existe una entrada extra con valor 1 y peso  $W_{0j}$  que se trata como el resto.

$$X_j = \sum_i y_i * w_{ij}$$

**Ecuación 1 (20)**

La salida de cada unidad de procesamiento,  $Y_j$ , se calcula usando una función no lineal de su entrada total.

$$Y_j = \frac{1}{1 + e^{-X_j}}$$

**Ecuación 2 (20)**

Es importante aclarar que no es de estricto uso las funciones dadas en las ecuaciones (1) y (2). Se puede utilizar cualquier función con derivada acotada (diferenciable), esto se debe al criterio empleado para propagar los errores, el cual se basa en el uso del gradiente. El uso de una función lineal para combinar las entradas a una unidad antes de aplicar la no lineal para calcular su salida (activación) simplifica grandemente el procedimiento de aprendizaje.

Haciendo uso de las ecuaciones (1) y (2) el algoritmo de aprendizaje backpropagation funciona de la siguiente manera:

Se inicializan todos los pesos  $W_{ij}$  con valores aleatorios pequeños (preferentemente se seleccionan números aleatorios en el rango de -0.1 a 0.1).

Se presenta una entrada de la clase  $m$  y se especifica la salida deseada.

Se calculan las salidas reales de todas las neuronas usando los valores actuales de los  $W_{ij}$ . La salida de la neurona  $U_j$ , denotada por  $Y_j$ , es una función no lineal de sus entradas totales:

$$Y_j = \frac{1}{1 + e^{-\sum_i y_i * w_{ij}}}$$

**Ecuación 3 (20)**

Se calcula el error  $E_j$  para todas las unidades. Si  $D_j$  es el valor de la salida deseada y  $Y_j$  es el valor de salida real para la  $j$ -ésima unidad, los errores se calculan por:

a) Para las unidades de salida

$$E_j = (Y_j - D_j) * Y_j * (1 - Y_j)$$

**Ecuación 4**

b) Para los nodos en las capas ocultas

$$E_j = Y_j * (1 - Y_j) * \sum_k z_k * w_{kj}$$

**Ecuación 5**

donde  $k$  recorre todos las neuronas conectadas a la neurona  $j$  en la capa siguiente a la que se encuentra esta y  $Z_k$  representa los errores calculados para las salidas de estas neuronas (que fueron calculadas usando las ecuaciones (4) ó (5) en dependencia de si la capa en la que se encuentran las unidades indicadas por  $k$  es de salida u oculta respectivamente).

Ajustar los pesos usando la ecuación

$$W_{ij(n+1)} = W_{ij(n)} + t * E_j * Y_i + h * (W_{ij(n)} - W_{ij(n-1)})$$

**Ecuación 6**

donde:

$(n+1)$ ,  $(n)$  y  $(n-1)$  indican el nuevo peso, el presente y el anterior respectivamente

$t$  es la velocidad de aprendizaje

$h$  es el momentum.

Presentar otra entrada e ir al paso 2.

Este procedimiento se repite hasta que los pesos y el error se estabilicen. Algunos investigadores iteran hasta que el error se acerca a 0; sin embargo, si el número de ejemplos

de entrenamiento excede la cantidad de pesos de la red, como se recomienda en diversos estudios, puede no ser posible hacer 0 el error. (20)

Este algoritmo es un procedimiento iterativo descendente del gradiente en el espacio de los pesos, el cual minimiza el error total entre las salidas deseadas y las reales de todas las unidades de procesamiento de la red.

Este algoritmo tiene varias deficiencias, entre las que podemos mencionar: es un procedimiento lento, temporalmente inestable, olvida patrones aprendidos previamente, y tiende a caer en mínimos locales. La principal objeción es que no es biológicamente posible; no hay evidencias de que las sinapsis puedan ser usadas en dirección inversa.

El defecto más obvio de este algoritmo es que la superficie de error puede contener un mínimo local de modo que el gradiente descendente no garantiza encontrar un mínimo global. La experiencia con muchos trabajos ha mostrado que la red muy raramente cae en un mínimo local que sea significativamente peor que el mínimo global. Este comportamiento indeseable se ha encontrado en redes que tienen justamente las conexiones suficientes para resolver la tarea, por lo cual añadiendo más conexiones se crean dimensiones extras en el espacio de los pesos que ofrecen caminos para evitar estos mínimos locales malos.

### 1.8. Aplicaciones de las Redes Neuronales Artificiales.

Las aplicaciones de las redes neuronales se podrían dividir según el campo del conocimiento donde se aplican. A continuación se verán algunos ejemplos. (13)

**Medicina.** Las aplicaciones en medicina se agrupan en problemas de diagnóstico médico. Es uno de los campos con más futuro y desafortunadamente, uno de los menos desarrollados. Algunas de las aplicaciones en este campo son:

*Detección de tumores cancerígenos.* Una red neuronal entrenada localiza y clasifica en imágenes médicas la posible existencia de tumores cancerígenos.

*Predicción del nivel de ciclosporina.* La ciclosporina es un fármaco usado habitualmente para evitar la reacción de rechazo en transplantes de riñón, corazón, pulmón e hígado. Predecir la concentración de este fármaco a corto plazo ayudaría a la optimización de la dosis siguiente.

*Compresión de señales electrocardiográficas.* Uno de los temas más activos actualmente en el campo de la ingeniería biomédica es la telemedicina. Esta disciplina consiste en el desarrollo de algoritmos que permitan el diagnóstico de una determinada enfermedad sin que el paciente se

tenga que desplazarse al centro médico. Las diferentes señales que necesita el médico se transmiten vía telefónica. Para aumentar la eficacia de esta transmisión se podría pensar en la compresión de la señal que consiste en aplicar diferentes algoritmos para reducir su tamaño. Uno de los métodos de compresión es con redes neuronales.

**Procesado de la señal.** En este campo las redes neuronales han encontrado un gran hueco de tal forma que ya existe una sociedad internacional sobre la aplicación de redes neuronales en problemas de procesamiento de la señal. Algunos problemas de clasificación donde se aplican las redes neuronales serían:

*Reconocimiento de patrones en imágenes.* Esta aplicación evidencia la capacidad de las redes neuronales ya que se trata de una tarea relativamente sencilla para un ser humano pero tremendamente costosa de implementar en un sistema artificial.

*Reconocimiento de voz.* Esta aplicación, de gran importancia de cara a la implementación de sistemas controlados por la voz, ha encontrado en las redes neuronales un camino para su desarrollo.

**Economía.** En esta disciplina las redes neuronales son directamente aplicables frente a otros métodos por sus características de no linealidad. Algunas de estas aplicaciones son:

*Concesión de créditos.* En esta aplicación las redes neuronales en virtud de determinados marcadores económicos de la persona que pide el préstamo decide su viabilidad o no.

*Detección de posibles fraudes en tarjetas de crédito.* Las redes neuronales pueden ser usadas como elementos discriminativos para conceder o no una determinada cantidad de dinero en un cajero automático.

*Determinación de la posibilidad de quiebra de un banco.* En esta aplicación la red neuronal determina el riesgo de quiebra de un banco en virtud de determinados parámetros económicos.

**Medio Ambiente.** Que vivimos en un ambiente dinámico y no lineal nadie lo puede negar; cualquier método aplicado a este campo necesariamente debe tener en cuenta estos hechos. Tenemos, pues, otro campo importante de aplicación de las redes neuronales. Algunas aplicaciones de éstas son:

*Predicción de niveles tóxicos de ozono en zonas urbanas y rurales.* Este gas nos protege de la radiación ultravioleta del sol, sin embargo, un exceso de este gas puede conducir a problemas. Una predicción de su concentración en la atmósfera a corto plazo (uno o dos días) podría



conducir a la aplicación de medidas para evitar posibles incrementos indeseados en la concentración de este gas.

### **1.9. Las Redes Neuronales aplicadas al diseño de nuevos fármacos.**

Como bien queda reflejado en la sección anterior, las RNA han sido aplicadas en diferentes ramas de la vida, demostrando su capacidad de resolver problemas complejos, fundamentalmente aquellos en que la relación entre sus datos es no lineal. A continuación se muestran brevemente algunos de los ejemplos que evidencian la aplicación de esta técnica en el diseño racional de fármacos:

#### *1.9.1 Construcción de un modelo de predicción para la propiedad farmacocinética Excreción Urinaria en fármacos orgánicos:*

En este trabajo se construye un modelo con los momentos espectrales de la matriz de adyacencia entre aristas ponderadas del grafo que representa a la estructura molecular, para predecir la propiedad farmacocinética excreción urinaria en fármacos orgánicos, utilizando una red neuronal artificial. El mejor modelo de red neuronal escogido, es capaz de predecir la propiedad con un error cuadrático medio de 5,85 y 9,88 para los conjuntos de entrenamiento y prueba, respectivamente. (21)

#### *1.9.2 Discriminación y predicción de propiedades de fármacos mediante redes neuronales:*

En este trabajo se estudia el problema de la discriminación y predicción de las propiedades farmacológicas de ciertos compuestos moleculares a partir de su topología ("structure-activity methods") utilizando perceptrones multicapa. Se concentran en el diseño de las redes neuronales mediante un barrido por multitud de configuraciones distintas. Entre ellas eligen aquellas que puedan formar parte de un comité de redes para intentar incrementar a tasa de aciertos en la predicción. Tras realizar pruebas con distintas estrategias internas en los comités, comprueban que las predicciones de las redes presentan una fuerte correlación, por lo que las mejoras obtenidas frente a redes individuales son poco significativas. (22)

#### *1.9.3 Discriminación, predicción y diseño de nuevos fármacos aplicando métodos de inteligencia artificial:*

En este trabajo se introducen nuevos índices que basados en la topología molecular permiten una adecuada representación de la estructura química y una importante mejora de la capacidad discriminante; predictiva y de diseño, respecto a otros índices topológicos. La aplicación de un

método de inteligencia artificial, como son las redes neuronales, permite un adecuado tratamiento de los datos y la superación de algunas de las limitaciones de los métodos clásicos de correlación y discriminación. El empleo de nuevos índices topológicos y su tratamiento mediante redes neuronales, conduce a obtener para el grupo farmacológico de los antibacterianos, unos excelentes resultados en la discriminación de actividad, predicción de la potencia y diseño de nuevas estructuras con potencial actividad farmacológica. Los resultados obtenidos confirman la capacidad de esta nueva herramienta en el diseño de nuevas moléculas con potencial actividad farmacológica. (23)

#### *1.9.4 Aplicación de métodos topológicos y de inteligencia artificial a la selección de nuevos antibacterianos:*

Este trabajo constituye una aplicación de la topología molecular, que tiene por objetivo la obtención de nuevos compuestos con actividad antibacteriana aplicando un modelo basado en redes neuronales artificiales combinado con análisis lineal discriminante, para realizar la selección molecular, empleando como descriptores un conjunto de Índices Topológico-Estructurales desarrollados en una investigación realizada por los autores. A partir de las descripción en formato SMILES de las moléculas se obtuvieron sus correspondientes índices topológico-estructurales que posteriormente fueron tratados matemáticamente mediante análisis lineal discriminante y redes neuronales artificiales, obteniendo una función discriminante y una arquitectura de la red neuronal que era capaz de discriminar de entre un conjunto de compuestos que mostraron teórica actividad antibacteriana. Estos compuestos fueron sometidos en primer lugar a ensayos de susceptibilidad antimicrobiana frente a cuatro cepas de microorganismos para poner de manifiesto su actividad y la capacidad predictiva del método propuesto, posteriormente los compuestos que se mostraron activos en estos ensayos fueron sometidos a ensayos de toxicidad aguda en animales de experimentación con objeto de determinar la Dosis Letal 50 de los mismos. De los compuestos seleccionados como teóricos antibacterianos, cuatro de ellos confirmaron dicha actividad, lo cual representó un resultado satisfactorio. (24)

### 1.9.5 Uso de una RNA para distinguir compuestos con actividad anticancerígena usando descriptores QSAR inductivos: (18)

En este trabajo se utilizan 30 descriptores inductivos, se utiliza una muestra de 380 compuestos y se trabaja con una RNA feed-forward con el algoritmo de entrenamiento back-propagation. Haciendo un estudio con estos datos logran un 84,28% de precisión para la correcta separación de los compuestos con y sin actividad contra el cáncer. El modelo QSAR elaborado basándose en el enfoque de redes neuronales artificiales fue ampliamente validado y por ello se le asignó carácter de confianza contra el cáncer a una serie de medicamentos contra el cáncer de prueba de la literatura. (25)

## 1.10 Programas que trabajan con Redes Neuronales Artificiales.

Existen varios programas concebidos para el trabajo con RNA, la gran mayoría de ellos ofrecen múltiples funcionalidades, pero lamentablemente no se pueden integrar a la plataforma que da origen a esta investigación (la plataforma alasGRATO) como se necesita. Además, algunos de los software son propietarios y la gran mayoría no brindan el código implementado para dar solución a los problemas. La diferencia entre los distintos productos software radica en aspectos tales como el tipo y el número de arquitecturas de red que soporta, velocidad de procesamiento, interfaz gráfica, exportación de código C para el desarrollo automático de aplicaciones, etc. A continuación se muestran algunos de los software que fueron estudiados y sus características representativas:

### 1.10.1 Weka.

Weka (Waikato Environment for Knowledge Analysis) es un software de código abierto publicado bajo la licencia GNU (General Public License). Posee una colección de algoritmos de aprendizaje automático para la extracción de datos. Los algoritmos pueden ser aplicados directamente a un conjunto de datos o llamar desde su propio código Java. Weka contiene herramientas para los datos de pre-procesamiento, clasificación, regresión, clustering, reglas de asociación, y la visualización.

### 1.10.2 Matlab

MATLAB en abreviatura MATrix LABoratory (laboratorio de matrices), es un software matemático que ofrece un entorno de desarrollo integrado (IDE) con un lenguaje de programación propio (lenguaje M). Este software es multiplataforma y propietario, entre sus

prestaciones básicas se hallan: la manipulación de matrices, la representación de datos y funciones, la implementación de algoritmos, la creación de interfaces de usuario (GUI) y la comunicación con programas en otros lenguajes y con otros dispositivos hardware.

Esta herramienta brinda una serie de funciones ya implementadas para el proceso de creación, entrenamiento y simulación de una red neuronal, específicamente el Perceptrón Multicapa, además un programa escrito en MATLAB admite la mayoría de las estructuras de programación tales como las estructuras (if, for, while).

Matlab posee un grupo muy amplio de funciones de transferencia, brindando la posibilidad de seleccionar las deseadas para cada caso de entrenamiento de una red neuronal.

### *1.10.3 Joone (Java Object Oriented Neural Engine)*

Joone es un framework en Java para construir y ejecutar aplicaciones de inteligencia artificial basadas en redes neuronales artificiales. Es open source (código abierto) y está cubierto por la Licencia Pública GNU. Las aplicaciones realizadas con Joone pueden ser construidas en una máquina local, entrenadas en un ambiente distribuido y ejecutarse en cualquier dispositivo.

Joone consiste de una arquitectura modular basada en componentes vinculados que pueden ser extendidos para construir nuevos algoritmos de aprendizaje y nuevas arquitecturas de redes neuronales.

Todos los componentes poseen algunas características básicas específicas, como persistencia, multihilos (multithreading), serialización y parametrización. Estas características garantizan escalabilidad y expansibilidad. (26)

### *1.10.4 SNNS (Stuttgart Neural Network Simulator)*

SNNS es un software para simular Redes Neuronales Artificiales en estaciones de trabajo Unix. Fue desarrollado en el Instituto para Sistemas paralelos y distribuidos de alto desempeño de la Universidad de Stuttgart. Este software posee dos componentes fundamentales: un simulador de Kernel escrito en C y una interfaz de usuario. (27)

### *JavaNNS (Java Neural Network Simulator)*

1.10.5 JavaNNS es el sucesor de SNNS. Se basa en su núcleo de computación, con un nuevo desarrollo, cómoda interfaz de usuario gráfica escrita en Java. Por lo tanto la compatibilidad con SNNS se logra, mientras que la plataforma logra mayor independencia. JavaNNS está

---

disponible gratuitamente y se puede descargar. En la actualidad, JavaNNS está disponible para Windows NT / Windows 2000, Solaris und RedHat Linux. (28)

#### 1.10.6 ALYUDA-NEURO-INTELIGENT:

Software de Redes Neuronales para Expertos, diseñado para soporte inteligente en la aplicación de redes neuronales para resolver problemas de predicción, clasificación, y de aproximación del " mundo real". Utiliza características inteligentes para pre-procesar hojas de datos, encontrar la arquitectura eficiente para el tipo de problema, analizar performance y aplicar la red neuronal a los nuevos datos. Los expertos pueden crear y probar sus soluciones de manera mucho más rápida, incrementando su productividad y mejorando resultados. (29)

Este software, al igual que el ALYUDA-FORECASTER XL y el ALYUDA-FORECASTER, son software propietarios que no se pueden probar ya que para utilizarlos hay que comprarlos. Otros de los productos comerciales son: ANSim (DOS), ExpertNet (DOS, Windows), Neuralesk (Windows), Neuralworks Pro II/Plus (DOS, OS/2, UNIX, VMS).

### 1.11 Variación de las funciones de transferencia en las Redes Neuronales Artificiales.

La gran mayoría de los software y de los estudios relacionados con las RNA emplean una única función de transferencia durante el entrenamiento de la misma. Por otra parte, han surgido algunas investigaciones entorno a este tema, ya que, partiendo de la idea de que las redes simulan el funcionamiento del cerebro humano, y que las funciones de transferencia denotan la salida de la red, existe la posibilidad de que la variación de las mismas puedan proporcionar mejoras en el funcionamiento de las redes.

A continuación se muestran algunas de las investigaciones que giran alrededor de este tema y de las conclusiones a las que han podido arribar.

El primer ejemplo hace referencia a un estudio iniciado en el año 2005 en el departamento de Neuroinformática y Robótica Cognitiva, en la Universidad Técnica de Ilmenau, Alemania.

La simulación de funciones de transferencia fue aplicada en una conferencia sobre neuroinformática, para demostrar el efecto del uso de diferentes funciones de transferencia en el comportamiento entrada-salida de las neuronas artificiales. En el mismo contexto, la simulación se utilizó como un ejercicio que los estudiantes tuvieran que resolver individualmente mediante el uso de la simulación prevista en un servidor de instituto. El ejercicio se realizó en tres repeticiones del mismo curso, uno de cada año, más de 100 estudiantes cada año. Las

---

experiencias indicaron que este escenario de aplicación para la simulación, como la mayoría de las soluciones, eran correctas pero no uniformes. (31)

*Red para la estimación de la densidad constructiva basada en diferentes funciones de transferencia:* En este trabajo se plantea que las redes que utilizan diferentes funciones de transferencias abren nuevas posibilidades para resolver problemas de difícil clasificación y el descubrimiento de una representación sencilla de los datos, pero también presentan nuevos desafíos teóricos. Encontrar el modelo con la complejidad óptima puede ser bastante difícil, aunque hay varios algoritmos basados en la teoría de la información que podrían ser de utilidad. Varios de los experimentos realizados hasta ahora indican que el algoritmo constructivo utilizado por FMS da respuestas con una gran varianza, encontrando soluciones con precisión similar usando diferentes combinaciones de funciones.

Hasta ahora, la red FSM ha sido probada con algunos tipos de funciones de transferencia en una red. Una posibilidad que vale la pena explorar es utilizar las familias de funciones de transferencia que son parametrizadas para proporcionar bordes de decisión flexibles. (32)

### **1.12 Sistemas distribuidos.**

La idea fundamental de un Sistema Distribuido (SD) es que constituye una combinación de computadoras y sistemas de transmisión de mensajes bajo un solo punto de vista lógico a través del cual los elementos de cómputo resuelven tareas en forma colaborativa. El sistema es capaz de procesar información debido a sus características esenciales:

El sistema consiste de una cantidad de computadoras cada una de las cuales tienen su propio almacenamiento, dispositivos periféricos y potencia computacional.

Todas las computadoras están adecuadamente interconectadas.

Una variedad de componentes que incluyen tanto plataformas de cómputo como las redes de interconexión que transportan mensajes entre ellas unificadas en un solo ambiente de procesamiento.

Por medio del sistema operativo adecuado, las computadoras mantienen su capacidad de procesamiento de tareas localmente, mientras constituyen elementos colaborativos de procesamiento en el ambiente distribuido.

### 1.12.1 Ventajas y desventajas de los Sistemas Distribuidos.

#### Ventajas:

Procesadores más poderosos y a menos costos.

Desarrollo de Estaciones con más capacidades

Las estaciones satisfacen las necesidades de los usuarios.

#### Uso de nuevas interfaces:

Avances en la Tecnología de Comunicaciones.

Disponibilidad de elementos de Comunicación.

#### Compartición de Recursos:

Dispositivos (Hardware).

Programas (Software).

#### Eficiencia y Flexibilidad:

Respuesta Rápida.

Ejecución Concurrente de procesos (En varias computadoras).

Empleo de técnicas de procesamiento distribuido.

#### Disponibilidad y Confiabilidad:

Sistema poco propenso a fallas (Si un componente falla no afecta la disponibilidad del sistema).

Mayores servicios que elevan la funcionalidad (Monitoreo, Telecontrol, Correo Eléctrico, etc.).

#### Crecimiento Modular:

Es inherente al crecimiento.

Inclusión rápida de nuevos recursos.

Los recursos actuales no se afectan.

Desventajas:

Requerimientos de mayores controles de procesamiento.

Velocidad de propagación de información muy lenta en algunos casos.

Servicios de recopilación de datos y servicios con posibilidades de fallas.

Mayores controles de acceso y proceso.

Administración más compleja.

Otro problema potencial tiene que ver con las redes de comunicaciones, ya que se deben considerar problemas debidos a pérdidas de mensajes, saturación en el tráfico y expansión.

### *1.12.2 Condiciones que un problema debe cumplir para ser distribuible.*

Para lograr la distribución de un problema que requiera grandes prestaciones de cómputo hay que tener en cuenta una serie de características que deben tener todo problema para encontrarle una buena solución distribuida. La característica más importante que debe tener es la posibilidad de dividir el problema en unidades más pequeñas que puedan ser procesadas de forma individual. En este sentido existen dos formas de dividir el problema:

**Particionado de los datos o Descomposición del Dominio:** En este particionado, el centro de atención es la partición de los datos, de ser posible estos son separados en pequeñas partes de aproximadamente el mismo tamaño. Luego, se dividen los cálculos a ser realizados, asociándolos con los datos sobre los cuales actúan.

**Particionado funcional o Descomposición Funcional:** La descomposición funcional es una estrategia diferente para enfrentar los problemas. El punto principal en este esquema es el cómputo a realizar en lugar de los datos manipulados. Si uno tiene éxito dividiendo las funciones del programa en grupos disjuntos, luego puede proceder a estudiar los requerimientos de datos de estos grupos funcionales. Si se da el caso de que la partición de la data resultante es también disjunta se dice que la partición funcional es completa. En este caso se procede al revés que en el caso anterior. Aquí se divide primero el cómputo y luego se mira la posibilidad de descomponer los datos.



---

### 1.12.3 Computación distribuida.

Existen varios modelos fundamentales que se usan para la creación de sistemas de cómputo distribuido, los cuales se clasifican en diferentes categorías: modelo de minicomputadora, modelo de Workstation, modelo Workstation-server, modelo de pool de procesadores e híbrido.

#### *Modelo de minicomputadora.*

Es simplemente la extensión del modelo de equipo centralizado. Este esquema consiste de varias minicomputadoras conectadas por una red de comunicación. Cada minicomputadora tiene su propio grupo de usuarios quienes pueden tener acceso a otros recursos no presente en su sistema a través de la red de datos. Este modelo se usa cuando existe necesidad de compartir recursos de diferentes tipos a través de acceso remoto.

#### *Modelo de Workstation.*

Consiste en varias estaciones de trabajo interconectadas por una red, cada usuario se autentica en su computadora de inicio y luego desde ahí puede enviar trabajos para ejecución. Cada estación tiene su propio disco duro, y su propio sistema de archivos. La implementación mostrada tiene la desventaja de que se puede desperdiciar tiempo de procesamiento si una o más computadoras no están haciendo uso de su CPU. Si el sistema determina que la estación del usuario no posee la suficiente capacidad de proceso para una tarea, transfiere el trabajo de forma automática hacia el equipo que tiene menos actividad en ese momento y por último se regresa la salida del trabajo a la estación del usuario.

#### *Modelo Workstation-Server.*

Posee varias estaciones de trabajo (algunas de las cuales pueden ser estaciones sin disco duro) comunicadas mediante red. Ahora que existe la potencialidad de tener estaciones sin disco, el sistema de archivos a usar por estas computadoras puede ser el de una computadora completa o alguno compartido de las diferentes computadoras del sistema. Algunas otras computadoras se pueden usar para proveer otros tipos de servicios, como base de datos, impresión, etc. Estas máquinas cumplen ahora nuevas tareas especializadas para proveer y administrar acceso a los recursos, en la jerga común se les llama servidores (servers).

---

*Modelo de pool de procesadores.*

Este se basa en el hecho de que los usuarios en promedio no requieren capacidad de procesamiento durante un buen rato, pero existen instantes en los que la actividad y los programas que ejecutan demandan potencia de trabajo en alto grado. A diferencia del modelo anterior en el que cada persona tiene su servidor asignado, en éste se dispone de un conjunto (pool) de servidores que son compartidos y asignados conforme a demanda. Cada procesador en el pool tiene su propia memoria y ejecuta un programa de sistema o de aplicación que le permite participar en el ambiente de cómputo distribuido.

Comparado con el modelo de Workstation-Server, el de Pool de Servers permite una mejor utilización del poder de cómputo disponible en el ambiente distribuido, dado que dicho poder computacional está disponible para todos, a diferencia de Workstation-Server, donde varios equipos pueden estar desocupados en algún momento pero su capacidad no se puede asignar a otros. Por otra parte, esta metodología provee gran flexibilidad ya que el sistema se puede expandir agregando procesadores que operarán como servidores adicionales para soportar una carga extra de trabajo originada por incremento en el número de usuarios o por la implantación de nuevos servicios.

*Modelo híbrido.*

El modelo híbrido permite combinar las mejores características de Workstation-Server y Pool, esencialmente agregando a la red de estaciones de trabajo un Pool de servidores que pueden ser asignados dinámicamente para trabajos que son muy extensos para máquinas individuales o que necesitan varios equipos concurrentemente para una ejecución adecuada. Esta variante tiene la ventaja de garantizar el nivel de respuesta a trabajos interactivos dado que permite la ejecución en la misma computadora de la persona que lo solicita. Por otra parte, su principal desventaja estriba en que el costo de implantación se eleva puesto que requiere mayor número de componentes para su construcción.

*Plataforma de Tareas Distribuidas (Tarenal).*

La Plataforma de Tareas Distribuidas surgió como alternativa a los costosos clústeres y supercomputadores. Su objetivo principal es brindar una herramienta, que funcione como una “supercomputadora virtual”, capaz de aunar y coordinar los esfuerzos entre los recursos disponibles y utilizar de esta forma, el poder computacional que ofrecen las computadoras

personales generalmente de bajas prestaciones pero interconectados mediante una red local lo suficientemente rápida para entre todos lograr obtener resultados.

Ofrece una alternativa de cómputo que aglutina en un solo conglomerado un conjunto de estaciones de trabajo. El sistema de cómputo desarrollado ha sido desplegado y utilizado en la Facultad de Bioinformática de la Universidad de las Ciencias Informáticas y en centros del Polo Científico del Oeste de La Habana. La Plataforma de Tareas Distribuidas está basada en el modelo Workstation – Server.

#### 1.12.4 Entrenamiento distribuido de una RNA

El entrenamiento de una RNA, teniendo en cuenta el tamaño de la muestra y la arquitectura de la red, pueden requerir largo tiempo en una sola computadora, impidiendo un uso interactivo de esta técnica para la extracción de información.

Por ejemplo, la RNA empleada en la búsqueda de Higgs boson utilizando los datos almacenados por el detector DELPHI en el acelerador LEP del CERN, requiere de varias horas y hasta de varios días para completar el proceso de formación con casi un millón de eventos simulados para una arquitectura simple de dos capas ocultas con cerca de 250 pesos de los entrenados. Esto evita que el usuario final pueda trabajar en un modo interactivo real mientras busca un análisis óptimo.

Resultados previos han demostrado que un algoritmo de RNA puede ser adaptado para que corra en un ambiente distribuido y de esta forma poder reducir el tiempo de entrenamiento de la red. (30)

No han sido muchas las investigaciones realizadas al respecto. A continuación se muestra uno de los trabajos más recientes que se enfocan en la reducción del tiempo de entrenamiento de las RNA a partir de su distribución:

#### **Implementación del entrenamiento interactivo de una Red Neuronal Artificial.**

Este trabajo fue realizado en el año 2004, en el Instituto de Física de Cantabria, España. Se implementó el entrenamiento de una RNA de forma distribuida usando MPICH-G2 (una grid), y se desplegó en el banco de pruebas del proyecto CrossGrid Europea. El balanceo de carga, incluidas las técnicas de adaptación, se utilizó para hacer frente a la configuración heterogénea de los recursos informáticos. Los primeros resultados obtenidos mostraron la viabilidad de este enfoque, y la oportunidad de un marco de calidad de servicio. Para dar un ejemplo, una

reducción en el tiempo de entrenamiento de 20 minutos utilizando un único nodo local fue reducido a menos de 3 minutos con 10 nodos distribuidos por toda España, Polonia y Portugal. (33)

### 1.13 Datos desbalanceados y medidas de evaluación de los algoritmos.

Teniendo en cuenta que una gran parte de los problemas presentes en la bioinformática pueden tener una naturaleza desbalanceada, se decidió realizar un estudio sobre este tema y por consiguiente analizar las posibles medidas de evaluación de los algoritmos al trabajar con datos con estas características. (34)

Los datos desbalanceados, específicamente los desbalanceados por clases, son conjuntos de datos cuya característica principal es que hay mucha diferencia entre la cantidad de casos de cada clase. Cada uno de los datos usados para trabajar pertenecen a una categoría, y el hecho de que existan muchos más casos de una que de otra puede dar lugar a problemas al intentar hallar métodos de aprendizaje.

La técnica de inteligencia artificial que en este trabajo se aborda, necesita aprender con tantos ejemplos sean posibles de cada clase, para luego poder hacer una buena generalización. Cuando la muestra para el entrenamiento no es suficiente o no está balanceada, puede traer problemas de memorización y por lo tanto la técnica dejaría de ser eficiente.

Se conocen diversas métricas que permiten evaluar cómo de buena ha sido una clasificación. Algunas de ellas, que se usan normalmente para evaluar clasificaciones genéricas, no son válidas para conjuntos desbalanceados, pues darían resultados irreales. A continuación se describen las métricas usadas y propuestas en la literatura de conjuntos desbalanceados.

#### 1.13.1 La Matriz de Confusión

Una Matriz de Confusión muestra la cantidad de individuos bien o mal clasificados por un método dado, según la clase a la que pertenezcan. La Tabla 2 muestra un ejemplo.

	Hipótesis	
	Negativos	Positivos
Clase actual		
Negativos	VN	FP
Positivos	FN	VP

Tabla 2. Matriz de confusión (34)

Donde **VN** representa el número de verdaderos negativos, o sea, los patrones negativos que han sido clasificados como negativos. La variable **VP** es el número de verdaderos positivos, o lo que es lo mismo, los patrones positivos que han sido clasificados como positivos. Por otra parte la variable **FP** representa el número de falsos positivos, que no es más que los patrones negativos que han sido clasificados como positivos y **FN** es el número de falsos negativos o lo que es lo mismo, los patrones positivos que han sido clasificados como negativos.

La precisión del método clasificador (Ecuación 7) es la división del número de aciertos por el número total de patrones:

$$precisión = \frac{VP + VN}{VP + VN + FP + FN}$$

Ecuación 7

En la Ecuación 7 los valores de precisión se encuentran en el intervalo [0,1], ya que a medida que los valores no clasificados correctamente (FP y FN) tienden a cero, la precisión aumenta de valor, siendo 1 el caso en que se tenga una clasificación perfecta (FP=0 y FN=0).

### 1.13.2 G-Media (g-means)

La G-Media se conoce como la media geométrica de las precisiones de cada caso para tener un indicador de medida del grado de precisión alcanzado por un método de clasificación. Su fórmula se puede ver en la Ecuación 8.

$$g - media = \sqrt{prec^+ \cdot prec^-}$$

Ecuación 8

donde los valores de  $prec^+$  y  $prec^-$  son los obtenidos por la Ecuación 9, correspondientes a la precisión en la clasificación de los elementos positivos y la de los negativos respectivamente.

$$prec^+ = \frac{VP}{VP + FN} \quad prec^- = \frac{VN}{VN + FP}$$

Ecuación 9

Se conoce además que el índice G-Media es una buena medida de la precisión alcanzada por un algoritmo sobre un conjunto desbalanceado, además de que es el más utilizado en este tipo de problemas.

### 1.13.3 Curvas de ROC

El análisis de la curva de ROC (Receiver Operating Characteristic) proporciona herramientas para distinguir clasificadores que son óptimos en alguna de las clases con respecto a clasificadores que no son tan óptimos, dependiendo del valor de algunos de sus parámetros.

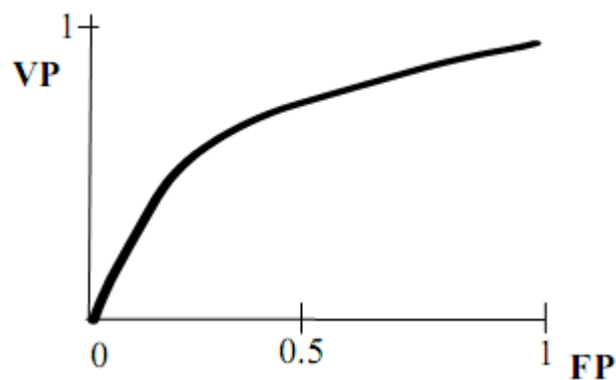
Las curvas de ROC para dos clases están basadas en una representación visual entre dos parámetros: la sensibilidad y la especificidad, que se muestran en la Ecuación 10:

$$\text{sensibilidad} = \text{prec}^+ \quad \text{especificidad} = \text{prec}^-$$

**Ecuación 10**

Donde  $\text{prec}^+$  y  $\text{prec}^-$  son los valores calculados en la Ecuación 9. La representación gráfica corresponde a colocar el valor  $1-\text{especificidad}$  en el eje de las X y la *sensibilidad* en el eje de las Y. Esto corresponde también a colocar la tasa de verdaderos positivos (VP) en el eje Y y la tasa de falsos positivos (FP) en el eje X. Esto proporciona un punto para cada clasificador. Así, se obtiene una curva de puntos. En la Figura 8 se muestra un ejemplo de curva de ROC.

Así, el punto (0,0) representa el clasificador que clasifica todos los patrones como negativos, mientras que el punto (0,1) representa el clasificador que clasifica correctamente todos los patrones.



**Figura 8. Ejemplo de Curva de ROC**

---

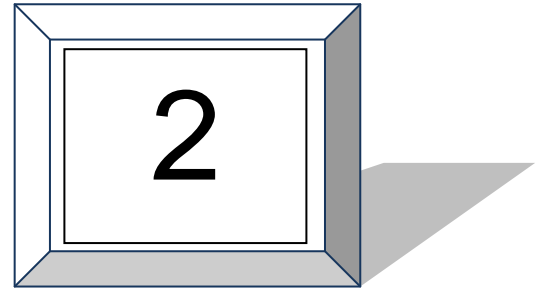
#### 1.13.4 Área Bajo La Curva De ROC (AUC)

La mayor exactitud de un clasificador se traduce en un desplazamiento "hacia arriba y a la izquierda" de la curva ROC. Basándose en esto, Bradley (34) sugirió que el área bajo la curva ROC (AUC: Area Under Curve) se puede emplear como un índice conveniente de la exactitud global de la prueba. Cuanto mayor es el área, mejor es el clasificador, y su valor máximo es 1, ya que es el área de un rectángulo de 1 por 1.

#### **Conclusiones.**

En este capítulo se realizó una breve introducción a las Redes Neuronales Artificiales presentando sus principales características, ventajas, arquitectura, así como sus clasificaciones teniendo en cuenta diferentes aspectos. Se hizo énfasis en el perceptrón multicapa como el tipo de RNA seleccionada para el estudio a desarrollar. Teniendo en cuenta los diferentes criterios consultados para la selección de la arquitectura del perceptrón durante el entrenamiento, se decidió utilizar una sola capa oculta y utilizar el segundo criterio de la selección de la cantidad de neuronas en la capa oculta. Asimismo se mostraron algunos de los software que trabajan con las RNA, de los cuales se escogió el Weka para extraer el algoritmo del perceptrón multicapa. Se mostraron además algunas de las investigaciones que se han desarrollado entorno a la variación de las funciones de transferencia y a la distribución de algoritmos para agilizar la etapa de entrenamiento de las RNA. Se presentó también información referente a los Sistemas Distribuidos en cuanto a su concepto, características básicas, funciones principales y modelos de computación distribuida, específicamente el modelo Workstation–Server que es el presente en la Plataforma de Tareas Distribuidas, la cual se escogió para distribuir el entrenamiento del perceptrón multicapa. Se hizo un breve estudio sobre las muestras desbalanceadas así como los métodos que se utilizan para conocer la precisión de los algoritmos que trabajen con muestras de este tipo. Se decidió utilizar la G-Media como medida de evaluación del algoritmo utilizado.

Capítulo



Programas y Procedimientos



---

## CAPÍTULO II: PROGRAMAS Y PROCEDIMIENTOS.

En este capítulo se hace referencia a las herramientas seleccionadas para resolver el problema en cuestión y se explican sus principales características. Se hace referencia al Weka que es el programa del que se extrae el algoritmo del perceptrón multicapa, así como se hace alusión a la plataforma de cálculo distribuido Tarenal, que es la herramienta con la que se distribuyen los datos para realizar el entrenamiento. Además se explican otras herramientas y los principales procedimientos realizados para la selección de la arquitectura y realizar el entrenamiento de la red.

### 2.1 Weka (se explicó en la sección 1.10)

En el capítulo anterior se hizo una breve descripción de esta herramienta. Solo resta agregar que al algoritmo que se propone en la misma se le hicieron algunas modificaciones para que se ajustara mejor al problema en cuestión. En el algoritmo propuesto se brinda la opción de modificar las funciones de transferencias a utilizar durante el entrenamiento, y de esta manera poder conocer el efecto que puede ocasionar esta variación en la eficiencia de la respuesta de la red.

### 2.2 Plataforma de Tareas Distribuidas (Tarenal) (se explicó en la sección 1.12.3).

En el capítulo anterior se hizo alusión a la plataforma Tarenal. A continuación se expondrán las clases fundamentales que fueron utilizadas para resolver el problema en cuestión.

#### 2.2.1 Clases de la Plataforma Tarenal.

La clase *DataManager* pertenece al servidor y su propósito es generar todas las unidades de trabajo que serán enviadas a los clientes, procesar los resultados, supervisar y ajustar el tamaño de las unidades, brindar información del estado de la ejecución del problema en un momento determinado y además se encarga de cerrar todos los recursos utilizados una vez concluido el trabajo distribuido.

La clase padre *DataManager* consta de una serie de métodos que deben ser redefinidos por el desarrollador para que el problema sea aceptado por el sistema. Una vez ejecutado el mismo se le asigna un directorio que podrá usar durante el cómputo.

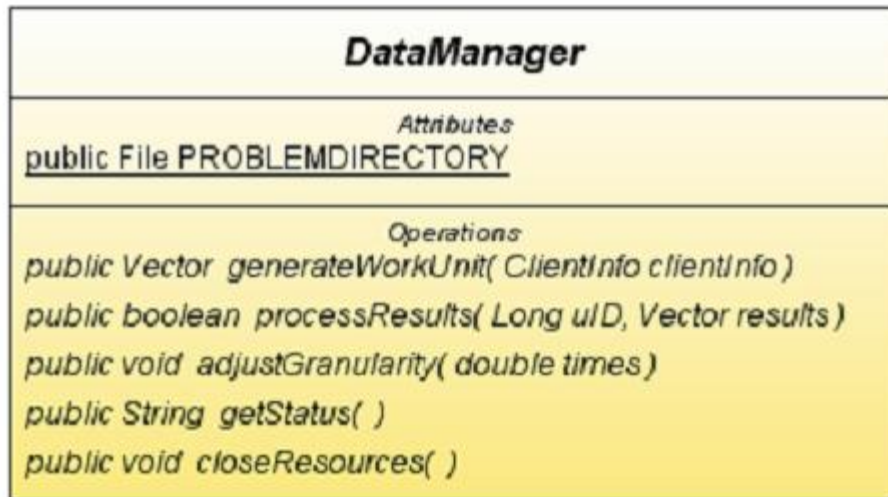


Figura 9. UML de la clase DataManager del Tarenal.

La clase *Task* se ejecuta en el cliente y se encargará de procesar todas las unidades de trabajo generadas por el método *generateWorkUnit()* de la clase *DataManager* del servidor.

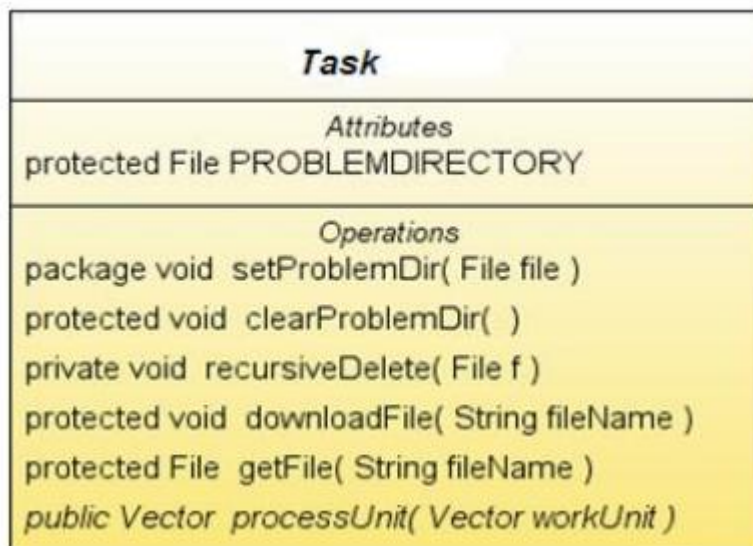


Figura 10. UML de la clase Task del Tarenal.

### 2.3 Java como lenguaje de programación.

Con el uso de este lenguaje de programación se hicieron los cambios necesarios a los algoritmos presentes en el Weka y en la plataforma Tarenal para ajustarlos al problema presente en esta investigación. Este es un lenguaje potente, orientado a objeto pues trabaja sus datos como objetos e interfaces a estos objetos. Cuenta con grandes capacidades de

interconexión TCP/IP por lo cual se puede acceder a la información disponible en red con mucha facilidad y seguridad. Una de sus características fundamentales es que es multiplataforma (Windows, Unix y Mac) debido a que todas sus ejecuciones y compilaciones son ejecutadas sobre una máquina virtual, por lo que no depende de la arquitectura de la máquina donde se ejecute.

#### **2.4 Eclipse como herramienta IDE**

Eclipse fue la herramienta IDE utilizada en el trabajo. Es un entorno de desarrollo integrado distribuido, de código abierto y multiplataforma. Al igual que el lenguaje de programación java puede ejecutarse en sistemas operativos con características diversas. La arquitectura basada en plug-ins permite integrar varios lenguajes de programación e introducir otras aplicaciones complementarias.

#### **2.5 El SPSS como software estadístico.**

Statistical Package for the Social Sciences (SPSS), es un software estadístico informático muy usado en las ciencias sociales y las empresas de investigación de mercado.

El sistema de módulos de SPSS provee toda una serie de capacidades adicionales a las existentes en el sistema base. El módulo utilizado en la investigación fue el **Pruebas no Paramétricas**, específicamente el de dos muestras relacionadas, el cual permite realizar distintas pruebas estadísticas especializadas en distribuciones no normales.

El SPSS fue empleado con el objetivo de comparar los resultados de los 2 algoritmos utilizados, o sea, el algoritmo para el entrenamiento local de un perceptrón multicapa y el algoritmo distribuido, y de esta forma conocer si eran significativas las diferencias entre las respuestas dadas por dichos algoritmos. Para ello se realizaron las pruebas no paramétricas utilizando la medida de evaluación (G-Media) de los algoritmos para realizar las comparaciones.

#### **2.6 Procedimientos.**

Se realizaron diferentes estudios sobre el funcionamiento y rendimiento del perceptrón multicapa tanto en ambientes distribuidos como locales. Todas las pruebas o experimentos que se llevaron a cabo fueron realizados en computadores personales con un procesador Pentium 4 con 1 Gb de RAM.

### 2.6.1 Determinación de la arquitectura del perceptrón

Para establecer la arquitectura a utilizar durante el entrenamiento del perceptrón se tuvieron en cuenta la cantidad de neuronas en las capas de entrada y oculta, las cuales dependen de la cantidad de descriptores de la muestra escogida. En todos los casos, la cantidad de neuronas en la capa de salida es 1. Se fijaron en 0.003 y 0.001 los valores de velocidad de aprendizaje y de momentum respectivamente. Como condición de parada se empleó la cantidad de iteraciones.

### 2.6.2 Breve explicación sobre el entrenamiento del perceptrón multicapa.

Luego de entrada los valores con los que se va a entrenar, y de definir la arquitectura de la red, la velocidad de aprendizaje y el momentum, además de la cantidad de patrones de entrada con que se va a realizar el entrenamiento, la aplicación genera aleatoriamente los pesos sinápticos.

Cada patrón de entrada se hace pasar a través de la estructura activando cada neurona y generando salidas en estas, dichas salidas son multiplicadas por los pesos sinápticos y constituyen la entrada de las neuronas de la capa siguiente, así sucesivamente hasta llegar a la capa de salida donde el resultado final es comparado con el resultado esperado generando un error el cual es propagado por toda la red (backpropagation) hasta llegar al origen, corrigiendo los valores sinápticos. Así sucede con cada patrón hasta que todos hayan pasado a través de la red, esto constituye una iteración o epoch.

Como se expuso anteriormente, el criterio para establecer la condición de parada para el entrenamiento es la *cantidad de iteraciones o epochs*, mediante el cual el investigador define la cantidad de iteraciones que considere necesaria para obtener una red neuronal lo suficientemente entrenada para procesar sus datos de manera satisfactoria.

## 2.7 Características de las muestras utilizadas.

Para realizar las pruebas necesarias se utilizaron 3 muestras. Dos de ellas de naturaleza bioinformática y que se utilizaron fundamentalmente para analizar el efecto que podría ocasionar la variación de las funciones de transferencia en la eficiencia de la respuesta de la red. Para realizar el entrenamiento estas muestras fueron separadas en 2 grupos, el 80 % para realizar el entrenamiento del perceptrón y el 20 % restante para realizar la prueba de la red entrenada. Las muestras se relacionan a continuación:

### 2.7.1 Muestra 1: Cefalosporinas

Las cefalosporinas son compuestos antibacteriales pertenecientes a la familia de los  $\beta$ -lactámicos que poseen una gran actividad antibacterial. Estos antibióticos pueden utilizarse para el tratamiento de niños con fallos renales o hepáticos. (1)

La muestra utilizada contiene 179 descriptores y 104 instancias (o patrones de entrenamiento). Se conoce que todos los compuestos de la muestra son activos, es por ello que la investigación va enfocada a determinar los compuestos más activos y los menos activos. Para ello se tuvo que adaptar la variable dependiente (actividad biológica) para que pudiera ser tratada como un problema de clasificación. El procedimiento seguido fue el de determinar el promedio de los valores de la actividad biológica (AB) de cada compuesto y considerar como muy activos aquellos cuya AB fuera mayor o igual que el promedio, y como poco activos los que no cumplieran con este criterio. De esta manera la muestra quedó dividida en 52 elementos muy activos y 52 poco activos.

### 2.7.2 Muestra 2: Inhibidores del Factor 1 del receptor esteroideogénico (SF-1).

La segunda muestra utilizada fue la 599, correspondiente a Inhibidores del Factor 1 del receptor nuclear esteroideogénico, (SF-1). Este es un ensayo de dosis-respuesta basado en células para la inhibición del receptor huérfano A relacionado a RAR (RORA). En él se observa relación entre la concentración y la respuesta biológica. El receptor SF-1 pertenece a la clase de receptores nucleares huérfanos que han sido poco investigados al nivel farmacológico (celular) que se reporta. Él se expresa en las glándulas adrenal, pituitaria, testículos y ovarios, y regula la producción de la hormona esteroidea a diferentes niveles. También incluye la expresión directa de la enzima P-450, principal involucrada en la síntesis de la hormona esteroidea.

Se plantea que el diseño apropiado de antagonistas del SF-1 puede brindar compuestos con utilidad terapéutica en el tratamiento del cáncer de próstata metastásico a través de la supresión, tanto de la síntesis de testosterona gonadal como de andrógeno adrenal. Otro beneficio potencial de estos resultados puede ser la identificación de ligandos del SF-1 que pudieran convertirse en una nueva clase de pequeñas moléculas reguladoras del metabolismo energético y la obesidad.

La muestra utilizada contiene 311 descriptores, 315 instancias (o patrones de entrenamiento) y la actividad biológica a predecir va a estar dividida en 175 elementos activos y 140 no activos.

### 2.7.3 Muestra 3: Reconocimiento facial.

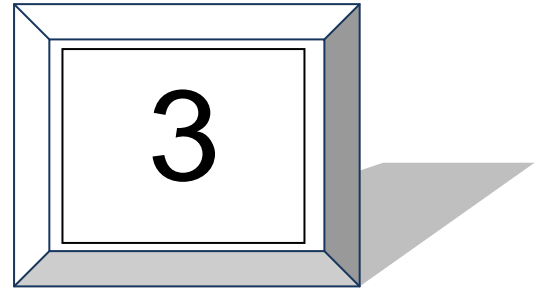
La tercera muestra utilizada es de reconocimiento facial. Está destinada a la identificación automática de una persona en una imagen digital, mediante la comparación de determinadas características faciales a partir de una imagen digital o un fotograma de un video. A pesar de no ser de la rama de bioinformática se decidió trabajar con esta muestra por la cantidad de patrones que contiene. Dentro del proyecto alasGRATO se prevé en un período muy cercano el empleo de muestras similares en tamaño, por lo que, desde el punto de vista del tratamiento de muestras grandes con redes neuronales, se justifica su empleo como preparación de condiciones para ese procesamiento.

La muestra contiene 70 mil patrones de entrenamientos con 376 descriptores, cuenta con 63175 en la clase 1 y 6825 en la clase 2.

### **2.8 Reducción del espacio muestral.**

Para realizar la reducción del espacio muestral se emplearon como métodos de búsqueda el Enfriamiento Simulado, Algoritmo Genético y un Algoritmo Híbrido (compuesto por el Greddy y Algoritmo Genético). En el caso del Enfriamiento Simulado se varió el factor alfa (factor que da la probabilidad de tomar o no una solución de la muestra) desde 0.7 hasta 0.9. Por otra parte, en los restantes algoritmos la probabilidad de cruzamiento de varió desde 0.6 hasta 0.9, con incremento de 0.1 y se mantuvo constante la probabilidad de mutación en 0.01. Se tuvieron en cuenta las medidas de evaluación CFS (selección basada en correlación de características) y Consistencia. La combinación de los métodos de búsqueda con las medidas de evaluación permitió obtener varias muestras, las cuales fueron utilizadas para seleccionar las de entrenamiento.

Capítulo



# Resultados y Discusión

---

## CAPÍTULO III: RESULTADOS Y DISCUSIÓN.

En el presente capítulo se muestran los resultados experimentales obtenidos, tanto en las pruebas realizadas en una computadora personal como en la plataforma de tareas distribuidas Tarenal. Se realiza una comparación de los resultados obtenidos teniendo en cuenta la calidad de los modelos, medido por la G-Media, para de esta forma conocer qué combinación de funciones de transferencia reportan mejores resultados para las muestras analizadas y si existen diferencias significativas entre los resultados obtenidos en un ambiente local con respecto a los distribuidos.

### **3.2 Resultados experimentales al ejecutar el perceptrón multicapa en una computadora personal.**

Se analizaron y compararon los resultados de varias ejecuciones del perceptrón multicapa en un procesador, variando algunos parámetros como la cantidad de patrones de entrada (variables independientes, descriptores moleculares), la arquitectura de la red, y las funciones de transferencia a utilizar. Para realizar la comparación de los resultados se tuvo en cuenta el tiempo de entrenamiento de cada ejecución y la medida de evaluación del algoritmo (G-Media).

#### *3.2.1 Resultados experimentales del entrenamiento del perceptrón con la muestra de cefalosporinas.*

En el caso de las cefalosporinas se obtuvieron varias muestras de entrenamiento a partir de los diferentes métodos utilizados para la reducción del espacio muestral, que brindaron diferentes conjuntos de variables para el establecimiento de los modelos. A continuación se analizarán los resultados del estudio realizado a la muestra obtenida al realizar la reducción del espacio muestral a 47 descriptores.

Se contó con una muestra balanceada de 104 patrones de entrenamiento, de ellos se utilizó el 80% (42 muy activos y 41 poco activos) para realizar el entrenamiento y los restantes (10 poco activos y 11 muy activos) para desarrollar la validación. Se mantuvieron constantes los siguientes parámetros:

Cantidad de neuronas en la capa de entrada: 47

Cantidad de neuronas en la capa oculta: 32

Cantidad de neuronas en la capa de salida: 1



Velocidad de aprendizaje: 0.003

Momentum: 0.001

Algoritmo de Búsqueda: Algoritmo Genético (AG) con 0.6 de probabilidad de cruzamiento y CFS como Medida de Evaluación.<sup>1</sup>

Como variables del experimento se utilizaron: cantidad de iteraciones y las funciones de transferencia utilizadas en la capa oculta y en la capa de salida. En la tabla 3 se muestran los resultados obtenidos.

**Tabla 3 Resultados experimentales de la muestra de cefalosporinas**

		Funciones de transferencia Sigmoidea– Sigmoidea (a)		Funciones de transferencia Tangente Sigmoidea- Sigmoidea (b)		Funciones de transferencia Lineal – Sigmoidea (c)	
No	Cant de Iterac.	G-Media	Tiempo entrenam.	G-Media	Tiempo entrenam.	G-Media	Tiempo entrenam.
1	1000	0.7637	(0:0:9)	0.8606	(0:0:11)	0.9045	(0:0:9)
2	1500	0.8090	(0:0:14)	0.9045	(0:0:16)	0.9128	(0:0:14)
3	2000	<b>0.8606</b>	(0:0:20)	<b>0.9534</b>	(0:0:22)	<b>0.9534</b>	(0:0:19)
4	3000	<b>0.8606</b>	(0:0:30)	<b>0.9534</b>	(0:0:34)	0.8606	(0:0:28)

		Funciones de transferencia Sigmoidea-Tangente Sigmoidea (d)		Funciones de transferencia Sigmoidea –Lineal (e)		Funciones de transferencia Tangente Sigmoidea - Tangente Sigmoidea (f)	
No	Cant de Iterac.	G-Media	Tiempo entrenam.	G-Media	Tiempo entrenam.	G-Media	Tiempo entrenam.
1	1000	0.8204	(0:0:10)	0.8204	(0:0:10)	0.8606	(0:0:11)
2	1500	0.8606	(0:0:15)	<b>0.9045</b>	(0:0:15)	0.8606	(0:0:16)
3	2000	<b>0.9045</b>	(0:0:20)	<b>0.9045</b>	(0:0:21)	<b>0.9045</b>	(0:0:22)
4	3000	<b>0.9045</b>	(0:0:30)	<b>0.9045</b>	(0:0:30)	<b>0.9045</b>	(0:0:34)

<sup>1</sup> Yaikiel Hernández Díaz, Comunicación personal. Resultados de la Tesis de Maestría en Bioinformática, UCI, 2009

Clase actual	Hipótesis	
	Negativos	Positivos
Negativos	VN= 10	FP= 1
Positivos	FN= 0	VP=10

Tabla 4. Matriz de confusión del ensayo 3-b

Clase actual	Hipótesis	
	Negativos	Positivos
Negativos	VN= 7	FP= 2
Positivos	FN= 3	VP=9

Tabla 5. Matriz de confusión del ensayo 1-a

Como se puede apreciar en la tabla 3, el aumento del número de las iteraciones es directamente proporcional al aumento del tiempo (lo cual es bastante predecible), no siendo así con el valor de la G-Media, ya que en algunos casos este valor puede aumentar (aumenta la eficiencia del algoritmo, la red aprende más), puede disminuir (la red no aprende sino que memoriza los datos y nos es capaz de realizar generalizaciones) o simplemente mantenerse constante.

De estos resultados se puede inferir que los mejores resultados de la red para la muestra analizada se obtienen con las combinaciones de funciones *Tangente Sigmoidea* en la capa oculta y *Sigmoidea* en la capa de salida y la combinación *Lineal – Sigmoidea*, para las que se obtuvo un valor de G-Media igual a 0.9534. Se puede apreciar en la matriz de confusión asociada al ensayo 3-b (tabla 4) que solo se clasificó incorrectamente un valor del conjunto de validación.

Es importante destacar también el resultado obtenido con la combinación de funciones *Lineal - Sigmoidea*, ya que con solo 1500 iteraciones se obtuvo el segundo mejor valor de G-Media de las pruebas realizadas, con un valor de 0.9128.

Otro aspecto interesante es que, en muchos casos no es necesario realizar múltiples iteraciones de la red, ya que el mejor valor de la G-Media se logra con un menor número de iteraciones, como es el caso del ensayo 2-e.

Cuando se analizó el ensayo 1-a, para el cual se reporta el peor valor de G-Media de las pruebas realizadas, se pudo apreciar de que el resultado es alarmante, ya que de 21 compuestos, 5 fueron clasificados incorrectamente.

### 3.2.2 Resultados experimentales del entrenamiento del perceptrón para la muestra SF-1.

Inicialmente se hizo un agrupamiento (clusterización) de la muestra debido a la diversidad estructural que poseía, para ello se utilizó uno de los algoritmos jerárquicos del Weka (el Ward) y de esta forma conocer la cantidad de clúster en los que se podía dividir la muestra. Posteriormente se utilizó el algoritmo simple KMeans para realizar la clusterización. De los 4 clusters resultantes se seleccionaron aleatoriamente los clusters 0 y 3 para realizar las pruebas. A estos dos clusters se les aplicaron los algoritmos de búsqueda y las medidas de evaluación explicadas en el epígrafe 2.8 para realizar la reducción del espacio muestral y se seleccionaron dos de las muestras resultantes, una de ella con 43 descriptores (perteneciente al Clúster 3) y otra con 20 (perteneciente al Clúster 0).

En el caso del clúster 3 resultaron 96 patrones de entrenamiento, se utilizó el 80% (32 inactivos y 44 activos) para realizar el entrenamiento y los restantes (10 activos y 10 inactivos) para desarrollar la validación. Como se puede apreciar, la muestra de entrenamiento está ligeramente desbalanceada.

Para la realización de los experimentos se mantuvieron constantes los siguientes parámetros:

Cantidad de neuronas en la capa de entrada: 43

Cantidad de neuronas en la capa oculta: 29

Cantidad de neuronas en la capa de salida: 1

Velocidad de aprendizaje: 0.003

Momentum: 0.001

Algoritmo de Búsqueda: Enfriamiento Simulado con un valor de alfa igual a 0.7 y Consistencia como Medida de Evaluación.

Como variables del experimento se utilizaron: cantidad de iteraciones y las funciones de transferencia utilizadas en la capa oculta y en la capa de salida. En la tabla 6 se muestran los resultados obtenidos.

**Tabla 6. Resultados experimentales de la muestra SF-1 con 43 descriptores (Clúster 3)**

No	Cant de Iterac.	Funciones de transferencia Sigmoidea–Sigmoidea (a)		Funciones de transferencia TangenteSigmoidea- Sigmoidea (b)		Funciones de transferencia Lineal – Sigmoidea (c)	
		G-Media	Tiempo entrenam.	G-Media	Tiempo entrenam.	G-Media	Tiempo entrenam.
1	1000	0.6038	(0:0:7)	0.6629	(0:0:8)	<b>0.7905</b>	(0:0:7)
2	1500	0.6038	(0:0:11)	0.6172	(0:0:12)	<b>0.7905</b>	(0:0:10)
3	2000	<b>0.6172</b>	(0:0:15)	<b>0.7703</b>	(0:0:16)	<b>0.7905</b>	(0:0:14)
4	3000	0.6038	(0:0:22)	0.6038	(0:0:25)	<b>0.7905</b>	(0:0:21)

No	Cant de Iterac.	Funciones de transf. Sigmoidea- TangenteSigmoidea (d)		Funciones de transferencia Sigmoidea –Lineal (e)		Funciones de transferencia Lineal – Lineal (f)	
		G-Media	Tiempo entrenam.	G-Media	Tiempo entrenam.	G-Media	Tiempo entrenam.
1	1000	0.5504	(0:0:7)	<b>0.6038</b>	(0:0:7)	<b>0.7669</b>	(0:0:7)
2	1500	<b>0.5547</b>	(0:0:11)	0.5000	(0:0:11)	<b>0.7669</b>	(0:0:11)
3	2000	<b>0.5547</b>	(0:0:15)	0.5504	(0:0:14)	0.6629	(0:0:15)
4	3000	<b>0.5547</b>	(0:0:22)	0.5547	(0:0:22)	0.6513	(0:0:22)

Clase actual	Hipótesis	
	Negativos	Positivos
Negativos	VN= 4	FP= 0
Positivos	FN= 6	VP= 10

Clase actual	Hipótesis	
	Negativos	Positivos
Negativos	VN= 4	FP= 4
Positivos	FN= 6	VP= 6

Tabla 7. Matriz de confusión del ensayo 1-c

Tabla 8. Matriz de confusión del ensayo 2-e

Al analizar los resultados obtenidos con la muestra SF-1 se puede apreciar cómo los mejores resultados de la G-Media se obtienen colocando la función *Lineal* en la capa oculta y la

*Sigmoidea* en la capa de salida. El mejor valor de la G-Media obtenido fue de 0.7905 en el ensayo 1-c, ya que se clasificaron correctamente todos los valores activos y el 40 % de los valores inactivos (Ver tabla 7). El peor resultado obtenido fue en el ensayo 2-e con un 50 % de precisión de respuesta (Ver tabla 8).

También es válido destacar que en muchos casos no es necesario realizar múltiples iteraciones de la red, ya que el mejor valor de la G-Media se logra con un menor número de iteraciones como es el caso del ensayo 1-f; en otros casos, a medida que aumenta el número de iteraciones, la red no aprende más, el valor de la G-Media no varía, como es el caso de los ensayos asociados a la combinación de funciones *Lineal- Sigmoidea*.

A continuación se verán los resultados obtenidos al realizar el entrenamiento con el Clúster 0. En este caso se obtuvieron 56 patrones de entrenamiento, de ellos 38 activos y 19 inactivos. Se utilizaron 45 patrones para realizar el entrenamiento y el 20 % restante se utilizó en la validación.

Para la realización de los experimentos se mantuvieron constantes los siguientes parámetros:

Cantidad de neuronas en la capa de entrada: 20

Cantidad de neuronas en la capa oculta: 14

Cantidad de neuronas en la capa de salida: 1

Velocidad de aprendizaje: 0.03

Momentum: 0.05

Algoritmo de Búsqueda: Enfriamiento Simulado con un valor de alfa igual a 0.7 y Consistencia como Medida de Evaluación.

**Tabla 9. Resultados experimentales de la muestra SF-1 con 20 descriptores (Clúster 0)**

No	Cant de Iterac.	Funciones de transferencia Sigmoidea– Sigmoidea (a)		Funciones de transferencia TangenteSigmoidea– Sigmoidea (b)		Funciones de transferencia Lineal – Sigmoidea (c)	
		G-Media	Tiempo entrenam.	G-Media	Tiempo entrenam.	G-Media	Tiempo entrenam.
1	1000	<b>0.8164</b>	(0:0:1)	0.7745	(0:0:1)	0.7745	(0:0:1)
2	1500	0.7745	(0:0:1)	0.7745	(0:0:2)	0.7745	(0:0:1)
3	2000	0.7745	(0:0:2)	0.7745	(0:0:3)	<b>0.8164</b>	(0:0:2)
4	3000	0.7745	(0:0:3)	<b>0.8164</b>	(0:0:4)	<b>0.8164</b>	(0:0:3)

No	Cant de Iterac.	Funciones de transf. Sigmoidea– TangenteSigmoidea (d)		Funciones de transferencia Sigmoidea –Lineal (e)		Funciones de transf. TangenteSigmoidea– TangenteSigmoidea (f)	
		G-Media	Tiempo entrenam.	G-Media	Tiempo entrenam.	G-Media	Tiempo entrenam.
1	1000	<b>0.6454</b>	(0:0:1)	<b>0.8660</b>	(0:0:1)	0.7745	(0:0:1)
2	1500	0.5270	(0:0:1)	0.7745	(0:0:1)	<b>0.8164</b>	(0:0:2)
3	2000	0.5270	(0:0:2)	0.7745	(0:0:2)	<b>0.8164</b>	(0:0:2)
4	3000	0.5270	(0:0:3)	0.7745	(0:0:3)	<b>0.8164</b>	(0:0:4)

Clase actual	Hipótesis	
	Negativos	Positivos
Negativos	VN= 3	FP= 0
Positivos	FN= 2	VP= 6

Clase actual	Hipótesis	
	Negativos	Positivos
Negativos	VN=1	FP= 1
Positivos	FN= 4	VP= 5

Tabla 10. Matriz de confusión del ensayo 1-e

Tabla 11. Matriz de confusión del ensayo 2-d

En este caso el mejor valor de G-Media es de 0.8660 y se obtiene con la combinación de funciones Sigmoidea – Lineal. Se puede apreciar en la tabla 10 cómo se logra con esta combinación de funciones y parámetros clasificar el 100 % de los valores activos y el 60 % de los valores inactivos. El peor resultado se le atribuye a la combinación de Sigmoidea – Tangente Sigmoidea para el cual se obtuvo un valor de G-Media de 0.5270; en este caso se clasifican correctamente casi todos los valores activos, no siendo así con los valores inactivos. Esta diferencia se le atribuye al desbalance presente en la muestra (2 elementos activos por cada 1 inactivo).

Como se puede apreciar en la tabla 9 no son muy significativas las diferencias de los resultados obtenidos entre el Clúster 0 (tabla 9) y el Clúster 3 (tabla 6). Es válido destacar, que en este caso, los mejores resultados se obtienen al realizar el entrenamiento de la red con un número reducido de descriptores (Clúster 0), para el cual se obtiene un valor de G-Media de 0.8660 mientras que con el Clúster 3, utilizando 43 descriptores, el mayor valor de G-Media que se obtiene es de 0.7905. De la misma manera ocurre con el valor más bajo de G-Media. En el caso del Clúster 0 el menor valor obtenido es de 0.5270 y para el Clúster 3 el que se obtiene es de 0.5000. Nuevamente se confirma que para estos casos los mejores resultados se obtienen con el Clúster 0 que es el que cuenta con menos descriptores. Es válido destacar también que la mejoría de los resultados obtenidos con el Clúster 0 también se pueden deber al aumento de los valores de momentum y learning rate.

### *3.2.3 Resultados experimentales del entrenamiento del perceptrón para la muestra de reconocimiento facial.*

De los 70 000 patrones de entrenamiento se utilizaron 60 000 (5933 de la clase 2 y 54067 de la clase 1) para realizar el entrenamiento y los 10 000 restantes (892 de la clase 2 y 9108 de la clase 1) para desarrollar la validación. Como se puede apreciar, la muestra está altamente desbalanceada lo cual influyó considerablemente en la calidad de los resultados obtenidos.

Para realizar las pruebas se mantuvieron constantes los siguientes parámetros:

Cantidad de neuronas en la capa de entrada: 20

Cantidad de neuronas en la capa oculta: 14

Cantidad de neuronas en la capa de salida: 1

Velocidad de aprendizaje: 0.003

Momentum: 0.001

Como variables del experimento se utilizaron: cantidad de iteraciones y las funciones de transferencia utilizadas en la capa oculta y en la capa de salida. En la tabla 12 se muestran los resultados obtenidos.

**Tabla 12. Resultado experimental de la muestra de reconocimiento facial para el cálculo local y distribuido empleando la combinación de funciones de transferencia Sigmoidea-Lineal.**

Entrenamiento Local				Entrenamiento Distribuido				
No	Cant de Iterac.	G-Media	Tiempo entrenam.	No	Cant de Iterac.	Cant. de PC	G-Media	Tiempo entrenam.
1	1000	0.7177	(0:25:31)	5	4000	2	0.7042	(1:15:00)
2	2000	0.7230	(0:48:52)	6	4000	4	<b>0.7514</b>	(0:39:00)
3	4000	<b>0.7241</b>	(1:44:32)	7	5000	2	0.7378	(1:27:00)
4	5000	0.7230	(2:13:53)	8	5000	4	0.7469	(0:53:20)

	Hipótesis	
Clase actual	Negativos	Positivos
Negativos	VN= 69	FP= 42
Positivos	FN= 905	VP= 8984

**Tabla 13. Matriz de confusión del ensayo 6**

En la tabla 12 se puede apreciar cómo el mejor resultado obtenido para la muestra analizada se logra en un entorno distribuido, con un valor de G-Media de 0.7514. Es importante destacar que este buen resultado se debe a que solo el 0.46 % de los valores de la clase 1 fueron mal clasificados, sin embargo más del 50 % de los valores de la clase 2 fueron mal clasificados, pero al no representar un número representativo en la muestra, no influyó considerablemente en el valor de la G-Media. Lo explicado se puede apreciar con mayor claridad en la tabla 13. Esta diferencia de resultados se debe al gran desbalance que presenta la muestra (aproximadamente hay 9 elementos de la clase 1 por cada 1 elemento de la clase 2). Al tener



esta situación, la red tiende a “aprender” mejor la clase mayoritaria y a “equivocarse” fácilmente a la hora de clasificar los de la clase minoritaria.

También se puede apreciar en la tabla 12 que en los entrenamientos locales, al aumentar el número de iteraciones el valor de la G-Media se mantiene estable, oscilando entre 0.71 y 0.72.

Para conocer si era eficiente la distribución del algoritmo para entrenamiento local se hicieron pruebas estadísticas a los ensayos 3, 5 y 6 y al 4, 7 y 8 respectivamente.

Al realizar las pruebas a los ensayos 3, 5 y 6 se hizo la comparación de los valores de la G-Media obtenidos, haciendo uso del software estadístico SPSS, el cual dio como resultado del test de Wilcoxon un valor 0.665, con lo cual se demuestra que no es significativa la diferencia de las respuestas dadas por los dos algoritmos (local y distribuido), o lo que es lo mismo, que se pueden utilizar los dos indistintamente, pues la calidad de los modelos no se afecta de manera estadísticamente apreciable. Se considera significativa la diferencia cuando el valor reflejado en el test es menor que 0,05. (37) No obstante, se puede apreciar que el mejor resultado se obtiene al distribuir la muestra en 4 computadoras con un valor de G-Media de 0.7514, que supera el obtenido en el ensayo local correspondiente. Además, se obtiene con una reducción del tiempo de más del 50 % (de 104 a 39 minutos).

En las pruebas realizadas a los ensayos 4, 7 y 8 se realizó el mismo procedimiento. En estos casos, al hacer la comparación de los valores de la G-Media, el valor obtenido en el test fue de 0.180, el cual revela que tampoco se encuentra diferencia significativa entre las respuestas dadas por los algoritmos. En esta ocasión, los valores de la G-Media obtenidos en un ambiente distribuido también superan el del entrenamiento local. Nuevamente el mejor valor se obtiene al distribuir el entrenamiento en 4 PC, para el cual se reduce considerablemente el tiempo de entrenamiento de 134 a 50 minutos.

**Tabla 14. Resultado experimental de la muestra de reconocimiento facial para el cálculo local y distribuido empleando la combinación de funciones de transferencia Sigmoidea-TangenteSigmoidea.**

Entrenamiento Local				Entrenamiento Distribuido				
No	Cant de Iterac.	G-Media	Tiempo entrenam.	No	Cant de Iterac.	Cant. de pc	G-Media	Tiempo entrenam.
1	1000	0.6965	(0:27:36)	5	4000	2	0.6407	(1:15:00)
2	2000	<b>0.7151</b>	(0:53:3)	6	4000	4	0.6725	(0:45:00)
3	4000	0.6775	(1:45:20)	7	5000	2	<b>0.6776</b>	(1:30:23)
4	5000	0.6685	(2:16:34)	8	5000	4	0.6431	(0:55:00)

Al analizar los resultados mostrados en la tabla 14 se puede apreciar que el valor de la G-Media en un entorno local se mantiene relativamente estable, inicialmente aumenta su valor y luego disminuye en cada iteración realizada, lo que demuestra que en este caso no es conveniente aumentar el número de iteraciones, pues la red no aprende más.

Como se puede apreciar en la tabla 14, el mejor valor de la G-Media que se obtiene es de *0.7151* y se logra en un entorno local.

Para conocer la eficiencia de la distribución del algoritmo para entrenamiento local se hicieron pruebas a los ensayos 3, 5 y 6 y al 4, 7 y 8.

En las pruebas realizadas a los ensayos 3, 5 y 6, al analizar el resultado reportado por el SPSS cuando se compararon los valores de la G-Media del algoritmo local y el distribuido, se obtuvo un valor de 0.180, el cual revela que no es significativa la diferencia entre las respuestas dadas por los algoritmos. No obstante, no se recomienda hacer la distribución en 2 PC, sino en 4, ya que como se aprecia en el ensayo 5 la calidad de la respuesta para 2 PC es peor que la obtenida por el algoritmo en un entorno local, no siendo así en el caso de la distribución en 4 PC.

Al realizar las pruebas a los ensayos 4, 7 y 8 el valor obtenido con el test estadístico fue 0.655, el cual revela que tampoco hay diferencias significativas entre los resultados obtenidos. A pesar de ello no se recomienda hacer la distribución en más de 2 computadoras, ya que como se

puede apreciar en el ensayo 8 de la tabla 14, el valor de la G-Media disminuyó al aumentar el número de iteraciones.

Si comparamos los resultados obtenidos en las tablas 12 y 14 se puede apreciar cómo para esta muestra los mejores resultados se logran con la combinación de funciones Sigmoidea-Lineal.

**Tabla 15. Resultado experimental de la muestra de reconocimiento facial para el cálculo local y distribuido empleando la combinación de funciones de transferencia Sigmoidea-Sigmoidea.**

Entrenamiento Local				Entrenamiento Distribuido				
No	Cant de Iterac.	G-Media	Tiempo entrenam.	No	Cant de Iterac.	Cant. de pc	G-Media	Tiempo entrenam.
1	1000	0.6946	(0:25:46)	5	4000	2	0.6423	(1:10:00)
2	2000	<b>0.6998</b>	(1:9:42)	6	4000	4	0.5895	(0:44:21)
3	4000	0.6969	(1:51:34)	7	5000	2	<b>0.6734</b>	(1:31:00)
4	5000	0.6950	(2:20:27)	8	5000	4	0.5923	(0:57:35)

Los resultados de la tabla 15 muestran que a medida que aumenta el número de iteraciones en los entrenamientos realizados en un entorno local, no se reportan mejorías en el valor de la G-Media. El mejor resultado obtenido se logra con 2000 iteraciones, con un valor de G-Media de 0.6998. Este valor de la G-Media se obtiene fundamentalmente por la buena predicción de los valores de la clase 1.

Con la aplicación del test de Wilcoxon a los resultados de los ensayos 3, 5 y 6, se obtuvo un valor de 0.180, el cual revela que no es significativa la diferencia entre los valores obtenidos en un ambiente local y el distribuido. Los resultados alcanzados sugieren que, para esta combinación de funciones, no es aconsejable hacer la distribución en más de 2 computadoras, ya que como se puede apreciar en la tabla 15, el valor de la G-Media disminuyó (ensayo 6). Además, debe valorarse cuidadosamente la distribución en 2 PC, ya que aunque el valor de la G-Media obtenido no difiere considerablemente con el obtenido en un entorno local, la pérdida de dos unidades de precisión, puede resultar relevante para el resultado que se desea obtener.

Un resultado similar se obtiene al realizar las pruebas a los ensayos 4, 7 y 8 en que el valor obtenido con el test de Wilcoxon fue también de 0.180. A pesar de ello, tampoco se

recomienda hacer la distribución en más de 2 computadoras, ya que como se puede apreciar, el valor de la G-Media disminuyó (ensayo 8).

Resulta interesante la diferencia entre los resultados obtenidos al aplicar diferentes combinaciones de funciones de transferencia a esta muestra. Solo con variar las combinaciones durante el entrenamiento de la red, se mejoraron los resultados en muchos casos con lo que se demostró nuevamente el peso fundamental que tienen las funciones y sus combinaciones en la eficiencia de la respuesta de la red. En la actualidad no se cuenta con una respuesta definitiva de la razón por la cual algunas combinaciones de funciones reportan mejores resultados que otras, lo que será objeto de estudio en futuras investigaciones.

## Conclusiones

- Se demostró que la variación de las funciones de transferencia en el perceptrón multicapa reporta resultados diferentes, medibles por el valor de la G-Media. La sustitución de la combinación clásica *Sigmoidea-Sigmoidea* por la combinación de funciones *Sigmoidea-Lineal* o *Lineal -Sigmoidea* mejora los resultados en el orden del 4,5 al 17%.
- Se desarrollaron dos algoritmos que permiten realizar el entrenamiento del perceptrón multicapa en ambiente local y distribuido respectivamente, variando las funciones de transferencia para realizar el entrenamiento. Se disminuyó el tiempo de entrenamiento aproximadamente entre un 30-60% al entrenar la red en un ambiente distribuido. El test estadístico aplicado sugirió que la calidad de los resultados en ambiente distribuido no se afecta de manera apreciable en comparación con los resultados en un entorno local.
- Se encontró que en algunos casos los resultados del valor de la G-Media en los ensayos distribuidos pueden ser mejores que el valor obtenido en el entrenamiento local correspondiente al número de iteraciones.

**Recomendaciones**

Realizar una correcta selección de variables antes de llevar a cabo el entrenamiento del perceptrón.

Ampliar el estudio a otras muestras y otras funciones de transferencia lineales y no lineales.

## Referencias

1. **J.C. Escalona, R. Carrasco, J. A. Padrón.** *Introducción al diseño de Fármacos.* <http://revistas.mes.edu.cu/elibro/libros/tecnologia/9789591606471.pdf/view>
2. **Simanca, Pedro.** Portal Universidad de Antioquia. [En línea] <http://ingenieria.udea.edu.co/investigacion/mecatronica/mectronics/redes.htm>.
3. **Muzachiodi, ASS Silvia M. Aranguren-ASS Silvia L.** *Redes Neuronales y Algoritmos Genéticos.* <http://www.fcceco.uner.edu.ar/extinv/publicdocentantiguas/sarangur/pdf/redsneuronals.pdf>
4. **Fernando Tanco, Grupo de Inteligencia Artificial (GIA).** UNIVERSIDAD TECNOLOGICA NACIONAL FACULTAD REGIONAL BUENOS AIRES. [En línea] <http://www.secyt.frba.utn.edu.ar/gia/RNA.pdf>.
5. Universidad Politécnica de Catalunya. *Sistema neuronal vs sistema electrónico.* [En línea] <http://upcommons.upc.edu/pfc/bitstream/2099.1/6277/8/6.%20SISTEMA%20NEURONAL%20VS%20SISTEMA%20ELECTR%C3%93NICO.pdf>
6. **Moreno, Juan José Montaña y Pol, Dr. Alfonso Palmer.** *TESIS DOCTORAL Redes Neuronales Artificiales aplicadas al Análisis de Datos.* PALMA DE MALLORCA : s.n., 2002. [http://www.tesisenxarxa.net/TESIS\\_UIB/AVAILABLE/TDX-0713104-100204//tjimm1de1.pdf](http://www.tesisenxarxa.net/TESIS_UIB/AVAILABLE/TDX-0713104-100204//tjimm1de1.pdf)
7. *Neural Networks and Physical Systems with Emergent Collective Computational Abilities.* **Hopfield, J. J.** Proc. NatL Acad. Sci. USA : s.n., April 1982, Vols. 79. <http://cns.upf.edu/jclub/hopfield82.pdf>
8. *Learning representations by back-propagating errors.* **David E. Rumelhart\*, Geoffrey E. Hinton† & Ronald J. Williams\*.** Institute for Cognitive Science, C-015, University of California, San Diego, La Jolla, California 92093, USA : s.n., 1986. <http://www.nature.com/nature/journal/v323/n6088/abs/323533a0.html>
9. **Burgos, Francisco José Palacios.** Herramientas en GNU/Linux para estudiantes universitarios. *Redes Neuronales con GNU/Linux.* [En línea] [http://www.ibiblio.org/pub/linux/docs/LuCaS/Presentaciones/200304curso-glisa/redes\\_neuronales/curso-glisa-redes\\_neuronales-html/index.html](http://www.ibiblio.org/pub/linux/docs/LuCaS/Presentaciones/200304curso-glisa/redes_neuronales/curso-glisa-redes_neuronales-html/index.html).
10. **B, Ing María I. Acosta; M, Ing Camilo A. Zuluaga y I., Ing Harold Salazar.** Tutorial de Redes Neuronales . [En línea] <http://ohm.utp.edu.co:16080/neuronales/>.
11. Ayuda del Matlab. [En línea] [Citado el: 15 de enero de 2009.]

12. **Casamayor, Carlos Carrascosa e Inglada, Vicente J. Julián.** *Tema 3: Areas de la IA:Ejemplos de Investigación Actual(III) Redes Neuronales.*

[http://personales.upv.es/ccarrasc/extdoc/Tema-3\\_2\\_redes\\_neuronales.pdf](http://personales.upv.es/ccarrasc/extdoc/Tema-3_2_redes_neuronales.pdf)

13. **Soria, Emilio y Blanco, Antonio.** Portal de ACTA en Internet . [En línea] [http://www.acta.es/articulos\\_mf/19023.PDF](http://www.acta.es/articulos_mf/19023.PDF).

14. Heaton, Jeff. Heaton Research. Introduction to Neural Networks for Java, 2nd Edition. [En línea] 2005. [Citado el: 23 de abril de 2009.]

<http://www.heatonresearch.com/book/programming-neural-networks-java-2.html>. 1604390085.

15. TELEDET. *CLASIFICACION DE REDES NEURONALES ARTIFICIALES.* [En línea] <http://www.teledet.com.uy/tutorial-imagenes-satelitales/redes-neuronales-artificiales-1.htm>.

16. **Espinosa, María del Rosario Villanueva.** Las Redes Neuronales Artificiales y su importancia como herramienta en la toma de decisiones. [En línea] [http://sisbib.unmsm.edu.pe/bibvirtualdata/tesis/Basic/Villanueva\\_EM/enPDF/Cap2.PDF](http://sisbib.unmsm.edu.pe/bibvirtualdata/tesis/Basic/Villanueva_EM/enPDF/Cap2.PDF).

17. **Espinosa, María del Rosario Villanueva.** Las Redes Neuronales Artificiales y su importancia como herramienta en la toma de decisiones. [En línea] [http://sisbib.unmsm.edu.pe/bibvirtualdata/tesis/Basic/Villanueva\\_EM/enPDF/Cap3.PDF](http://sisbib.unmsm.edu.pe/bibvirtualdata/tesis/Basic/Villanueva_EM/enPDF/Cap3.PDF).

18. **Espinosa, María del Rosario Villanueva.** Las Redes Neuronales Artificiales y su importancia como herramienta en la toma de decisiones. [En línea] [http://sisbib.unmsm.edu.pe/bibvirtualdata/tesis/Basic/Villanueva\\_EM/enPDF/Cap4.PDF](http://sisbib.unmsm.edu.pe/bibvirtualdata/tesis/Basic/Villanueva_EM/enPDF/Cap4.PDF)

19. **Ballesteros, Alfonso.** Neural Networks Framework. [En línea] <http://www.redes-neuronales.netfirms.com/tutorial-redes-neuronales/clasificacion-de-redes-neuronales-respecto-al-aprendizaje.htm>.

20. **Pérez, Dr. Rafael Bello.** *CURSO INTRODUCTORIO A LAS REDES NEURONALES ARTIFICIALES.* Departamento de Ciencia de la Computación, Universidad Central de Las Villas : s.n., 1993.

21. *Construcción de un modelo de predicción para la propiedad farmacocinética Excreción Urinaria en fármacos orgánicos.* **Guevara, Lic. Carlos, Bello, Dr Rafael y González, Lic Humberto.** Villa Clara, Cuba : s.n.

22. **Juan Lucas Domínguez Rubio, María José Castro Bleda, Wladimiro Díaz Villanueva.** Discriminación y predicción de propiedades de fármacos mediante redes neuronales. [En línea] 2003.

23. **Oltra, José Jaen.** Discriminación, predicción y diseño de nuevos fármacos aplicando métodos de inteligencia artificial. [En línea] 1999.



24. **Soler, Miguel Murcia.** Aplicación de métodos topológicos y de inteligencia artificial a la selección de nuevos antibacterianos . [En línea]
25. **Jaiswa, Kunal.** Jaiswa Kunal ; Naik Kumar Pradeep . *Bioinformation*. [En línea] 2008. <http://www.ncbi.nlm.nih.gov/pmc/articles/PMC2561164/>.
26. **Autores, Colectivo de.** SourceForge. [En línea] <http://sourceforge.net/projects/joone/>.
27. **Tübingen, Colectivo de autores de University of.** JavaNNS:Java Neural Network Simulator. [En línea] [http://www.ra.cs.uni-tuebingen.de/software/JavaNNS/welcome\\_e.html](http://www.ra.cs.uni-tuebingen.de/software/JavaNNS/welcome_e.html).
28. **Stuttgart, Colectivo de autores de la University of.** Stuttgart Neural Network Simulator. [En línea] <http://www.ra.cs.uni-tuebingen.de/SNNS/>.
29. ALYUDA. [En línea] <http://www.alyuda.com/spanish/neural-network-software.htm>.
30. **O. Ponce, J. Cuevas, A. Fuentes, Marco J. , R. Marco, C. Martíñez-Rivero, R. Menéndez y Rodríguez, D.** Training of Neural Networks: Interactive Possibilities in a Distributed Framework. s.l. : Springer Berlin / Heidelberg, 2002.
31. **Klaus Debes, Alexander Koenig, Horst-Michael Gross.** Transfer Functions in Artificial Neural Networks. *Journal of New Media in Neural and Cognitive Science and Education*. April / July 2009.
32. **Adamczak, Włodzisław Duch and Rafał.** Max Planck Institute for Astrophysics. *Constructive density estimation network based on several different separable transfer functions*. [En línea] <http://www.mpa-garching.mpg.de/~opmolsrv/MolPhysGroup/GHFDiercksen/RecentPublications/p-222.pdf>.
33. **D. Rodríguez, J. Gomes, J. Marco, R. Marco, and C. Martínez-Rivero.** SpringerLink . *MPICH-G2 Implementation of an Interactive Artificial Neural Network Training*. [En línea] 2004. <http://www.springerlink.com/content/ev25xubw3a8wvr8d/fulltext.pdf?page=1>.
34. **Vicenç Soler Ruiz,** LÓGICA DIFUSA APLICADA A CONJUNTOS IMBALANCEADOS: APLICACIÓN A LA DETECCIÓN DEL SÍNDROME DE DOWN, 2007
35. jeffheaton. Heatonresearch. [En línea] [Citado el: 16 de febrero de 2009.] <http://www.heatonresearch.com/online/introduction-neural-networks-java-edition-2/chapter-5>
36. NCBI. *National Center for Biotechnology Information* . [En línea] U.S. National Library of Medicine. <http://www.ncbi.nlm.nih.gov/pubmed/14594453>.
37. Departamento de Sociología IV. *Introducción al análisis de datos*. [En línea] [http://www.ucm.es/info/socivmyt/paginas/D\\_departamento/materiales/analisis\\_datosyMultivariable/19nparam\\_SPSS.pdf](http://www.ucm.es/info/socivmyt/paginas/D_departamento/materiales/analisis_datosyMultivariable/19nparam_SPSS.pdf).

38. Méndez, R. (2004). Aplicación de las Redes Neuronales Artificiales en la predicción de la curva des destilación ASTM D86 en gasolinas automotrices. *Revista de Ingeniería UC* , 30.
39. Penedo, M. F. (s.f.). Tema 1. Conceptos Básicos.  
<http://www.varpa.org/~mgpenedo/cursos/scx/scx.html>
40. Takeyas, M. B. (2003). Redes Neuronales Artificiales.  
[http://www.itnuevolaredo.edu.mx/maestros/sis\\_com/takeyas/Apuntes/Inteligencia%20Artificial/Apuntes/tareas\\_alumnos/RNA/Redes%20Neuronales.pdf](http://www.itnuevolaredo.edu.mx/maestros/sis_com/takeyas/Apuntes/Inteligencia%20Artificial/Apuntes/tareas_alumnos/RNA/Redes%20Neuronales.pdf)

## **Glosario de Términos**

**Moléculas:** Una molécula es una partícula formada por un conjunto de átomos ligados por enlaces covalentes.

**Átomos:** Es la entidad química más pequeña, el mismo está compuesto de protones y neutrones.

**Descriptor:** Número que describe la estructura química o una propiedad de la molécula o fragmento de esta.

**Índices:** Contienen información relacionada con la forma molecular, el grado de ramificación, tamaño molecular y la flexibilidad estructural.

**Índice del Estado Refractotopológico Total:** Se define como la suma de los valores del Índice del Estado Refractotopológico de cada átomo del fragmento considerado en una molécula dada.

**Índice Topológico:** Número que se calcula generalmente a partir de la matriz de adyacencia o de distancia de los elementos de un grafo molecular.

**Plug-in:** Es una aplicación informática que interactúa con otra aplicación para aportarle una función o utilidad específica, generalmente muy específica, como por ejemplo servir como driver en una aplicación, para hacer así funcionar un dispositivo en otro programa.

**Ponderar:** Determinar el peso de algo. Atribuir un peso a un elemento de un conjunto con el fin de obtener la media ponderada.

**Compuestos orgánicos:** Los compuestos o moléculas orgánicas son los compuestos químicos basados en Carbono, Hidrógeno y Oxígeno, y muchas veces con Nitrógeno, Azufre, Fósforo, Boro, Halógenos.

**Índice Topológico:** Número que se calcula generalmente a partir de la matriz de adyacencia o de distancia de los elementos de un grafo molecular.

**CITMA:** Es el organismo encargado de dirigir, ejecutar y controlar la política del Estado y del Gobierno en materia de Ciencia, Tecnología, Medio Ambiente y uso de la energía nuclear, propiciando la integración coherente de estas para contribuir al desarrollo sostenible del país.

**GNU:** es un acrónimo recursivo que significa GNU No es Unix (GNU is Not Unix).

**IEEE:** es la asociación más grande del mundo profesional de la promoción de innovación y excelencia tecnológica en beneficio de la humanidad. Sus siglas en español significan Instituto de Ingenieros Electricistas y Electrónicos.