UNIVERSIDAD DE SEVILLA

Departamento de Lenguajes y Sistemas Informáticos

Tesis Doctoral

NUEVOS MODELOS DE REDES NEURONALES EVOLUTIVAS PARA CLASIFICACIÓN. APLICACIÓN A UNIDADES PRODUCTO Y UNIDADES SIGMOIDE

Antonio Javier Tallón Ballesteros

Sevilla, enero de 2013

UNIVERSIDAD DE SEVILLA

Departamento de Lenguajes y Sistemas Informáticos

NUEVOS MODELOS DE REDES NEURONALES EVOLUTIVAS PARA CLASIFICACIÓN. APLICACIÓN A UNIDADES PRODUCTO Y UNIDADES SIGMOIDE

MEMORIA QUE PRESENTA

Antonio Javier Tallón Ballesteros

PARA OPTAR AL GRADO DE DOCTOR EN INFORMÁTICA POR LA UNIVERSIDAD DE SEVILLA DENTRO DEL PROGRAMA DE DOCTORADO "INGENIERÍA Y TECNOLOGÍA DEL SOFTWARE", CON MENCIÓN HACIA LA EXCELENCIA MEE2011-0129

Directores

César Hervás Martínez

José C. Riquelme Santos

Enero de 2013

D. César Hervás Martínez, Catedrático de Ciencias de la Computación e Inteligencia Artificial de la Universidad de Córdoba, y

D. José C. Riquelme Santos, Catedrático de la Universidad de Sevilla del área de Lenguajes y Sistemas Informáticos

C E R T I F I C A N Q U E:

D. Antonio Javier Tallón Ballesteros ha realizado bajo su supervisión el Trabajo de Investigación titulado:

NUEVOS MODELOS DE REDES NEURONALES
EVOLUTIVAS PARA CLASIFICACIÓN.
APLICACIÓN A UNIDADES PRODUCTO
Y UNIDADES SIGMOIDE

Una vez revisado el mismo, consideran que cumple con los estándares de calidad del área y, por lo tanto, autorizan la presentación del mismo como Tesis Doctoral en la Universidad de Sevilla y estiman oportuna su presentación al tribunal que habrá de valorarla. Dicha tesis ha sido realizada dentro del Programa de Doctorado "Ingeniería y Tecnología del Software", con mención de calidad MEE2011-0129, según la Resolución de 6 de octubre de 2011, de la Secretaría General de Universidades, por la que se concede la Mención hacia la Excelencia a los programas de doctorado de las universidades españolas.

Sevilla, enero de 2013

Firmado:
D. César Hervás Martínez

Firmado:
D. José C. Riquelme Santos

Tesis Doctoral parcialmente subvencionada por:

- El Ministerio de Educación y Ciencia con el proyecto **TIN2007-68084-C02-02**

TIN2007-68084-C02-02

- El Ministerio de Ciencia e Innovación con el proyecto **TIN2011-28956-C02-02** y conjuntamente con la Unión Europea con fondos FEDER en el proyecto **TIN2011-22794**

TIN2011-28956-C02-02 y TIN2011-22794

- La Consejería de Innovación, Ciencia y Empresa de la Junta de Andalucía con los proyectos **P08-TIC-3745** y **P11-TIC-7528**

P08-TIC-3745 y P11-TIC-7528

Índice de contenidos

Índice de figuras

Índice de tablas

Agradecimientos

La realización de la Tesis Doctoral es, indudablemente, la combinación del trabajo del doctorando y de muchas más personas, entre las que se incluyen los directores, familiares, amigos y compañeros.

Me gustaría agradecer a César Hervás y a José Riquelme, los directores de esta Tesis, por haberme brindado la oportunidad de llevarla a cabo. Son numerosas las muestras de interés, conocimiento, ánimo y dedicación que he recibido. Muchas gracias por las certeras recomendaciones.

También ha sido muy importante el cariño y apoyo recibido de mis padres, Miguel y Pilar. Sin vosotros no habría sido factible. Sandra, mi esposa, a ti también tengo mucho que agradecerte por el sacrificio realizado y por haber tenido la paciencia, la comprensión y las ganas de colaborar activamente para que esta Tesis se concluya.

Gracias a mis hermanos, Miguel Ángel y Pili, que sé que estáis deseosos de ver y leer esta Tesis; a mis restantes familiares por el interés mostrado y a mi familia política, especialmente a Antonio y Chari, Mariola y Antonio, quienes siempre han estado pendientes del proceso.

No deseo olvidarme de mis amigos, quienes también se han interesado en múltiples ocasiones por el desarrollo de esta Tesis, entre los que se encuentran Javi, Juanjo, Leticia, Manuel, Juan Antonio y María José, Antonio y Mari Carmen. A partir de ahora voy a aumentar el tiempo disponible para el ocio.

También me gustaría agradecer a mis compañeros del Departamento de Lenguajes y Sistemas Informáticos de la Universidad de Sevilla, entre los que debo destacar a Octavio Martín, Mayte Gómez, Rafael Ceballos, Toñi Reina y Rafael Corchuelo, por todos los momentos compartidos tanto dentro como fuera de la Escuela, y a mis compañeros del grupo de Aprendizaje Automático de la Universidad de Sevilla y del grupo BIGS de la Universidad Pablo de Olavide por todas las colaboraciones tanto pasadas como futuras.

Finalmente, me gustaría hacer extensivo el agradecimiento a otros colegas y/o compañeros de estudio de diferentes universidades, entre los que cabe mencionar a Jaime, Jesús, Felipe, Javi, Juan Carlos, F. Palao, Salva, Julián, José Miguel, Juan, José Manuel, Paco Urendes, Manolo y Pedro Antonio.

A todos, sinceramente gracias.

Resumen

El tema de investigación en el que se circunscribe nuestra Tesis Doctoral se incluye dentro de la extracción automática de conocimiento. Existen distintas formas de realizar razonamiento inductivo, centrándonos aquí en aprendizaje supervisado. En concreto, los modelos de clasificación que manejamos son Redes Neuronales Artificiales (RNAs) y el aprendizaje de los mismos se realiza mediante Computación Evolutiva. Este tipo de redes se conocen como Redes Neuronales Artificiales Evolutivas (RNAEs).

En las RNAEs, el entrenamiento de la red neuronal se lleva a cabo mediante un algoritmo evolutivo. La razón es que, al tratarse de un método de optimización global, no posee la limitación de los métodos locales de quedarse fácilmente estancado en óptimos locales. El algoritmo evolutivo usado pertenece al paradigma de Programación Evolutiva, que se basa en emplear sólo dos operadores: replicación y mutación.

Al tratarse de modelos de RNAs, se aplican dos tipos de mutación: estructural y paramétrica para modificar tanto la estructura del modelo de red, esto es, el número de nodos en la capa intermedia, como los pesos de las conexiones. No se hace uso del operador de cruce debido al conocido problema de la permutación, que consiste en que diferentes representaciones pueden corresponder al mismo modelo de RNA. Merece ser reseñado que en todas las propuestas realizadas en la presente Tesis

Doctoral se lleva a cabo una evolución simultánea de las arquitecturas de red y de los pesos y coeficientes asociados al modelo de red. Los modelos de red que empleamos en la Tesis Doctoral están formados por tres capas: la capa de entrada, la capa intermedia o capa oculta y la capa de salida. El tipo de unidades en la capa oculta son unidades de tipo producto o unidades de tipo sigmoide.

En la Tesis Doctoral se presentan varios modelos de Redes Neuronales Evolutivas para Clasificación que podemos discernir a grandes rasgos bajo las siguientes categorías:

a) Distribución del Diseño Experimental, denominado EDD (*Experimental Design Distribution*). La idea básica es aplicar el algoritmo evolutivo de manera automática a una serie de configuraciones diferentes y evaluar qué combinación de parámetros es más idónea para cada problema. La propuesta inicial se llevó a cabo con unidades producto y recientemente ha sido extendida al uso de unidades sigmoide, dando lugar a EDDSig (*Experimental Design Distribution with Sigmoidal units*). La metodología EDD se ha empleado en muchos trabajos posteriores para poder evaluar la eficacia de las propuestas. En total, se ha empleado en 25 bases de datos, 23 procedentes del UCI [Frank y Asuncion, 2010] y 2 problemas reales, uno de Microbiología Predictiva y otro de Medio Ambiente.

b) Algoritmo Evolutivo en dos fases, cuyo acrónimo es TSEA (*Two-Stage Evolutionary Algorithm*). La metodología TSEA emplea un algoritmo evolutivo refinado con dos fases, que trabaja con dos poblaciones iniciales de individuos, cada una de ellas con individuos con topologías diferentes, que en la primera fase son entrenadas durante un número pequeño de generaciones, para posteriormente seleccionar

la mitad mejor de cada población y fusionarlas en una nueva población. A esta última se le aplica, en la segunda fase, el ciclo evolutivo completo, salvo la fase de inicialización aleatoria de la población inicial. Experimentalmente, se concluyó que para el número de generaciones de la primera fase el 10% es un valor muy apropiado, pues permite combinar de una manera muy adecuada una mayor exploración de soluciones en la primera fase y una explotación de soluciones con mayor calidad respecto a las que se obtendrían aleatoriamente sin ser sometidas a ningún tipo de evolución parcial. Los resultados obtenidos con la metodología TSEA son mejores y más eficientes, respecto a los obtenidos con EDD. Con el propósito de validar la metodología en dos fases, se ha ampliado al uso de unidades de tipo sigmoide, generando la metodología TSEASig (*Two-Stage Evolutionary Algorithm for neural networks with Sigmoidal units*). Se ha llevado a cabo una comparación entre las metodologías TSEA, EDDSig y TSEASig aplicadas a 14 bases de datos, 12 disponibles en el repositorio UCI y 2 problemas complejos del mundo real: *BTX* y *Listeria monocytogenes*. *BTX* es un problema de clasificación multiclase relativo al Medio Ambiente que considera diferentes tipos de agua contaminadas. *Listeria monocytogenes* es un problema biclase en Microbiología Predictiva.

c) Aplicación de Selección de Atributos de manera independiente a las metodologías EDD y TSEA. Las metodologías mencionadas previamente son muy útiles y efectivas, sin embargo, la idea de mejorar debe ir más allá. Por tanto, en este caso se inició una línea de investigación muy prometedora que aplica un preprocesamiento a las diferentes bases de datos o, mejor dicho, al conjunto de entrenamiento, mediante el uso de métodos de selección de atributos basados en filtros. Las nuevas metodologías se han denominado EDDFS

(*Experimental Design Distribution with Feature Selection*) y TSEAFS (*Two-Stage Evolutionary Algorithm with Feature Selection*) y han sido aplicadas a problemas de especial complejidad, tomando como referencia que la tasa de error con los clasificadores C4.5 o 1-NN esté al menos en el 20%, aproximadamente. La metodología EDDFS ha sido evaluada con 14 bases de datos del UCI y se ha experimentado con 6 filtros de manera independiente. Los resultados obtenidos han mejorado significativamente en cuanto a tasa de acierto en clasificación, respecto a los de EDD. En estos momentos el trabajo está sometido a revisión en una revista. Por otra parte, en la experimentación de TSEAFS se hace uso de 4 filtros y 19 bases de datos, 18 del UCI y un problema real de Biomedicina de Trasplantes, *Liver-transplantation*, en España. La precisión de TSEAFS supera significativamente a la de TSEA.

El dominio de aplicación de los algoritmos propuestos ha sido bastante amplio. Concretamente, se han empleado 31 bases de datos de problemas de clasificación, 28 procedentes del UCI y 3 de problemas del mundo real, del ámbito de Microbiología Predictiva, Medio Ambiente y de Trasplantes Hepáticos en Medicina.

Este documento presenta nuestros objetivos, hipótesis de partida, publicaciones, estado del arte, modelos implementados, resultados, conclusiones y referencias bibliográficas.

Capítulo 1

Introducción

1.1. PLANTEAMIENTO

En diversas áreas del conocimiento como Medicina, Biología, Psicología, Ingeniería, etc. surge el problema de establecer una relación funcional entre las diferentes variables que intervienen en el fenómeno que se está estudiando. Por tanto, la idea es obtener un modelo a partir de un problema expresado en términos de una tabla de valores numéricos de diferentes variables.

De manera general, existen dos maneras de abordarlo. La primera es de carácter deductivo. Básicamente, el método consiste en la división del sistema en subsistemas que son modelados a partir de leyes físicas y relaciones previamente aceptadas como hipótesis. La segunda aproximación es de naturaleza inductiva, puesto que realiza la estimación de los modelos a partir del análisis de los datos. Este proceso de estimación se denomina comúnmente Aprendizaje, el cual persigue obtener una representación de los datos que pueda emplearse para diferentes objetivos como predecir o clasificar.

La Computación Bioinspirada se basa en emplear una analogía con sistemas naturales y sociales para diseñar métodos heurísticos no

determinísticos de búsqueda, optimización, etc. En la actualidad, los Algoritmos Bioinspirados son uno de los campos más prometedores en el diseño de algoritmos. Dichos algoritmos modelan, por tanto, de forma aproximada un fenómeno existente en la naturaleza empleando una metáfora biológica. Vienen caracterizados por ser no determinísticos, presentar una estructura paralela implícita y ser adaptativos, esto es, utilizan realimentación con el entorno para modificar el modelo y los parámetros. Como ejemplo de estos modelos destacan los algoritmos basados en Computación Evolutiva (CE), cuyas implementaciones sustentadas en evolución de población se denominan Algoritmos Evolutivos (AEs) –asentados en los principios darwinianos de la Evolución Natural–, las Redes Neuronales Artificiales (RNAs) –centradas en la simulación del comportamiento del Sistema Nervioso–, los Algoritmos Inmunológicos –toman como referencia la simulación del comportamiento del Sistema Inmunológico– y la Optimización basada en Colonias de Hormigas, que tiene su fundamento en la simulación del comportamiento de las hormigas cuando buscan y recogen comida. Se pretende, por tanto, establecer una dualidad entre la Evolución Natural y la Evolución Artificial.

La CE está compuesta por modelos basados en poblaciones, cuyos elementos (individuos) representan soluciones a problemas [De Jong, 2006]. La simulación de este proceso en un ordenador resulta ser una técnica de optimización probabilística que con frecuencia mejora a otros métodos clásicos en problemas difíciles. Tiene múltiples aplicaciones como Optimización Estructural, Aprendizaje, Generación de Trayectorias, Control de Procesos Químicos y Planificación de Sistemas de Producción.

Las RNAs [Bishop, 1995; Haykin, 1999; Haykin, 2009; Stahl y Jordanov, 2012] constituyen una de las áreas de la Ingeniería del

Conocimiento que más se ha desarrollado en los últimos años; pueden situarse dentro de la segunda forma de abordar el problema de modelado comentado anteriormente. Sus propiedades y características han hecho de ellas una herramienta usada con frecuencia en la resolución con éxito de problemas reales de gran complejidad, pertenecientes a diferentes ramas como Reconocimiento de Patrones [Pao, 1989], Diagnóstico Médico, Modelado de Datos Financieros [Saito et al., 2000], Predicción Meteorológica [Krasnopolsky y Fox-Rabinovitz, 2006] o Veterinaria [Kuncheva, Del Rio Vilas y Rodríguez, 2007], etc.

La combinación de la CE con las RNAs ha dado lugar a las Redes Neuronales Artificiales Evolutivas (RNAEs) [Montana y Davis, 1989; Yao, 1993; Yao, 1999; García-Pedrajas, Hervás-Martínez y Muñoz-Pérez, 2003; Rocha, Cortez y Neves, 2007; Azzini y Tettamanzi, 2008]. Usualmente, para diseñar redes neuronales se emplean procedimientos de prueba y error, por ejemplo, mediante la exploración de un número diferente de nodos en la capa oculta, siguiendo una estrategia de búsqueda ciega, que usa solamente un conjunto pequeño de configuraciones posibles, lo que supone una tarea computacionalmente intensiva. Los algoritmos evolutivos son costosos en tiempo, por lo que para un problema de mediano tamaño el diseño de las RNAs mediante dichos algoritmos puede resultar prácticamente inabordable por un ordenador con buenas prestaciones.

En esta Tesis Doctoral, utilizamos dos tipos de modelos de neuronas o unidades entre los modelos de proyección: a) *el modelo aditivo*, en el cual el valor de salida es función de la suma de las entradas ponderadas por los correspondientes pesos y b) *el modelo multiplicativo*, donde la alternativa más general es la de las denominadas Unidades Producto (UP), cuya salida viene expresada en función del producto de

cada entrada elevada a un peso que pertenece al dominio de los números reales.

En esta investigación empleamos habitualmente el modelo de red neuronal basado en UP, introducido por Richard Durbin y David E. Rumelhart en 1989 [Durbin y Rumelhart, 1989]. Con menor frecuencia usamos redes basadas en Unidades Sigmoide (US). La arquitectura de red presenta tres capas: la capa de entrada, la capa intermedia –que está compuesta por unidades de tipo producto o bien sigmoide– y la capa de salida. Para el entrenamiento de la red neuronal se emplea una metaheurística de tipo global, basada en poblaciones, ampliamente extendida como son los Algoritmos Evolutivos. Nuestro ámbito de aplicación es la resolución de problemas de Clasificación biclase y multiclase.

Una vez que han sido validadas las nuevas metodologías propuestas, con el propósito de mejorar los modelos en cuanto a calidad, eficiencia y sencillez, se han empleado técnicas de Preprocesamiento o Preparación de Datos [Zhang, Zhang y Yang, 2003]. Aplicamos técnicas de Selección de Atributos (SA) mediante filtros a los conjuntos de datos de entrenamiento [Liu et al., 2010].

1.2. CONTEXTO DE INVESTIGACIÓN

La Minería de Datos (MD) es un campo interdisciplinar [Han, 2005; Tan, Steinback y Kumar, 2005; Witten, Frank y Hall, 2011] cuyo objetivo general es descubrir relaciones en los datos. Dentro de la MD encontramos dos vertientes: a) descriptiva o no supervisada como, por ejemplo, el descubrimiento de patrones que representan los datos y b) predictiva o

supervisada –es el caso que nos ocupa–, como método para pronosticar el comportamiento del modelo basado en los datos disponibles.

Los algoritmos de MD tienen tres componentes: el modelo, el criterio de preferencia o elección y el algoritmo de búsqueda. Existen dos tipologías para el modelo, según su función o según su representación. En el primer caso, podemos hablar de clasificación, regresión, clustering, generación de reglas, reglas de asociación, modelos de dependencia o análisis de secuencias. En el segundo caso, puede ser RNAs, árboles de decisión, discriminación lineal, etc. Los parámetros del modelo son determinados mediante un algoritmo de búsqueda teniendo en cuenta el criterio de elección o preferencia que hace un mejor ajuste del modelo a los datos.

Por otra parte, la MD está relacionada con otras disciplinas, entre las que se encuentra el Aprendizaje Automático [Flach, 2012], que es un área de la inteligencia artificial concebida hace aproximadamente cinco décadas con el objetivo de desarrollar métodos computacionales capaces de aprender, es decir, inducir conocimiento a partir de los datos. Los algoritmos de aprendizaje se clasifican en dos categorías: a) métodos de caja negra, tales como redes neuronales o los métodos bayesianos y b) métodos orientados al conocimiento, tales como los que generan árboles de decisión, reglas de asociación o reglas de decisión.

Las técnicas de MD son sensibles a la calidad de la información sobre la que se pretende extraer conocimiento. Cuanto mayor sea esta calidad, mayor será la de los modelos de toma de decisiones generados. Por tanto, la obtención de información útil es un factor clave. Esto da lugar a la conveniencia de llevar a cabo previamente al proceso de MD un Preprocesamiento o Preparación de Datos que incluye todas aquellas

técnicas de análisis de datos que permiten mejorar la calidad de los mismos. La importancia de la preparación de los datos se debe a varias razones: a) los datos reales pueden ser impuros debido a datos incompletos, con ruido o inconsistentes; b) la preparación de los datos puede generar un conjunto de menor tamaño que el original lo cual puede mejorar la eficiencia en MD. En este sentido, se pueden desarrollar técnicas dirigidas a seleccionar datos relevantes, eliminar anomalías o bien reducir el volumen de datos, mediante la selección de atributos, selección de instancias, discretización y c) la preparación selecciona datos de mayor calidad, los cuales pueden conducir a modelos mejores. Para ello, se emplean procedimientos para recuperar información incompleta, resolver conflictos, eliminar datos erróneos, o bien eliminar variables que estén fuertemente correladas.

El dominio del trabajo realizado en la Tesis Doctoral se encuadra dentro de la disciplina del Reconocimiento de Patrones [Bishop, 2006], que se encuentra integrada en la MD. Su aplicación en bases de datos es uno de los problemas de mayor interés en numerosas áreas de la ciencia. Uno de los objetivos de este tipo de problemas es encontrar un modelo de clasificación que, tras ser entrenado con un conjunto de datos observados relativos a un determinado fenómeno (datos de entrenamiento), sea capaz de generalizar, es decir, de predecir la clase a la que pertenecen con nuevos datos (datos de generalización o test), que no habían sido tratados durante la generación del modelo.

Por otro lado, gran parte de las metodologías que pueden ser aplicadas en MD se engloban dentro de la disciplina emergente de Inteligencia Computacional (IC). Entre las herramientas integradas en IC, tal como establece Amit Konar, se encuentran la Computación Granular, la Neurocomputación y la Computación Evolutiva [Konar, 2005].

La Computación Evolutiva interpreta la naturaleza como una inmensa máquina de resolver problemas y trata de encontrar el origen de dicha potencialidad para reutilizarla en sus propias aproximaciones [Eiben y Smith, 2008]. Algunos de sus algoritmos de búsqueda, por ejemplo, los Algoritmos Evolutivos (AE) [Bäck, 1996], están basados en el concepto de la selección natural. Existen cuatro paradigmas de investigación dentro del campo de los AEs: los Algoritmos Genéticos (AG), las Estrategias Evolutivas (EE), la Programación Evolutiva (PE) y la Programación Genética (PG). En realidad, todas estas técnicas son muy parecidas y comparten muchos aspectos que describiremos a continuación.

a) *Algoritmos Genéticos*. Los AG se caracterizan por utilizar una codificación del individuo mediante una cadena (cromosoma) con un número fijo de elementos (genes). Fueron propuestos en 1975 por John Holland [Holland, 1975] y desde entonces han tenido una gran aceptación en la comunidad científica. Su principal operador de intercambio entre los modelos de la población es el operador de cruce.

b) *Estrategias Evolutivas*. Las EE fueron desarrolladas en los años 70 del siglo XX por Ingo Rechenberg [Rechenberg, 1973] y, más adelante, por Hans-Paul Schwefel [Schwefel, 1981] como herramientas para la optimización aerodinámica. En principio, estaban basadas en una población de un solo individuo al que se le aplicaba en distintos pasos un operador de mutación. Posteriormente, las EE han dado lugar a un conjunto de métodos computacionales que trabajan con una población de individuos que pertenecen al dominio de los números reales y que, mediante los operadores de mutación y de recombinación, evolucionan para alcanzar el óptimo de la función objetivo.

c) *Programación Evolutiva*. La PE fue originalmente concebida por Lawrence J. en 1960 y, posteriormente, fue plasmada en el que se considera el libro de referencia sobre este tema, escrito por él junto con otros investigadores [Fogel, Owens y Walsh, 1966]. Se define como un método de optimización estocástica para generar autómatas de estado finito capaces de reconocer secuencias de un alfabeto. A pesar de sus muchas coincidencias con las EE, ha sido desarrollada de forma independiente durante cerca de treinta años. Un algoritmo de PE tiene como principal característica que no codifica las soluciones candidatas, sino que trabaja directamente con ellas. Por esta razón es, generalmente, más complicado plantear el operador de recombinación genética, de manera que el algoritmo resultante delega toda la potencia de búsqueda al operador de mutación.

d) *Programación Genética*. John R. Koza [Koza, 1991] propuso un modelo de AE específico, denominado PG, para obtener programas de ordenador que resuelvan problemas dados. El algoritmo utiliza árboles que simbolizan expresiones en lenguaje de programación y se aplican operadores específicos que modifican las expresiones.

Así mismo, la IC [Fulcher, 2008] es denominada a veces *Soft Computing* (SC), aunque la idea general del SC es la de ser una combinación flexible de diferentes metodologías de IC. En los casos en los que se hace uso de una única herramienta computacional para resolver un problema, SC se puede considerar como IC. Algunas de las técnicas más extendidas de SC son los AEs, descritos anteriormente; la Lógica Difusa, que se encuentra fuera del ámbito de esta Tesis y las RNAs.

Los modelos de RNAs han demostrado ser paradigmas de computación potentes [Sundararajan y Saratchandran, 1998] que pueden

tratar eficazmente problemas complejos de Clasificación y Reconocimiento. A diferencia de los algoritmos procedurales convencionales, que siguen un flujo secuencial, las computaciones neuronales se realizan a través de un gran número de nodos. El número de neuronas empleado en una aplicación concreta es generalmente proporcional a la complejidad del problema.

1.3. OBJETIVOS

Los objetivos de la Tesis Doctoral son los siguientes:

1) Realizar una revisión del estado de arte existente sobre Redes Neuronales Artificiales en general, haciendo especial hincapié en los métodos de entrenamiento.

2) Proponer nuevas metodologías de entrenamiento de modelos de redes neuronales evolutivas para resolver problemas de Clasificación binaria y multiclase que permitan mejorar la precisión.

3) Combinar el preprocesamiento de datos mediante Selección de Atributos basada en filtros con las anteriores metodologías y determinar los métodos que, en términos generales, son más apropiados.

4) Aplicar las metodologías mencionadas a problemas del mundo real de Microbiología Predictiva y de Biomedicina de Trasplantes.

1.4. HIPÓTESIS DE PARTIDA

El punto de partida de esta Tesis Doctoral ha sido un algoritmo de PE para el entrenamiento de Redes Neuronales artificiales de Unidades Producto (RNUP) para problemas de clasificación, publicado originariamente en [Hervás-Martínez, Martínez-Estudillo y Gutiérrez, 2006]. El objetivo principal de este algoritmo es el de diseñar la estructura (número de nodos y número de conexiones) y, simultáneamente, optimizar los coeficientes de los modelos de RNUP. En lo sucesivo, nos referiremos a este algoritmo como Algoritmo Evolutivo para el entrenamiento de redes neuronales de Unidades Producto (AEUP).

Las hipótesis que manejamos en la presente Tesis son las siguientes:

- Hipótesis 1: La modificación de parámetros tanto de la topología de la red como del AEUP puede mejorar la eficacia de la búsqueda.

- Hipótesis 2: Una mayor diversidad en el proceso evolutivo puede contribuir a la mejora de las soluciones con mayor eficiencia.

- Hipótesis 3: El empleo de técnicas de preprocesamiento de datos podría dar lugar a una mejora tanto en la calidad de las soluciones como en el tiempo de computación requerido para obtener dichas soluciones.

Todas estas hipótesis son las que han motivado el desarrollo de todas nuestras publicaciones que aparecen reseñadas en el siguiente capítulo. En la fase preliminar del desarrollo de las publicaciones se ha realizado una exhaustiva revisión bibliográfica, que se sintetiza en el

apartado que trata sobre el estado del arte contenido en los capítulos 2 y 3 de esta Tesis Doctoral.

1.5. PUBLICACIONES

En este apartado enumeramos las publicaciones que se han realizado en el proceso investigador de la Tesis por orden cronológico inverso agrupadas en tres bloques: revistas internacionales con factor de impacto, capítulos de libro y congresos.

1.5.1. Revistas internacionales con factor de impacto

Se han publicado dos artículos en revistas internacionales con factor de impacto:

1. Antonio J. Tallón-Ballesteros, César Hervás-Martínez, José C. Riquelme y Roberto Ruiz (2012, en prensa). Feature selection to enhance a two-stage evolutionary algorithm in product unit neural networks for complex classification problems. *Neurocomputing*, Holanda. DOI: 10.1016/j.neucom.2012.08.041. Factor de impacto: 1,580 Q2 (JCR 2011) [Tallón-Ballesteros et al., 2012].

Este artículo presenta una metodología que mejora la precisión del algoritmo evolutivo en dos fases presentado en el artículo publicado en la revista *Expert System with Applications* [Tallón-Ballesteros y Hervás-Martínez, 2011] sobre redes neuronales evolutivas de unidades producto para tareas de clasificación por medio de métodos de selección de atributos implementados como filtros. La metodología mejorada se ha denominado TSEAFS (*Two-Stage Evolutionary Algorithm with Feature Selection*, algoritmo evolutivo en dos fases con selección de atributos). Han

sido considerados cuatro filtros para probar la propuesta. El dominio de aplicación ha sido 18 bases de datos del repositorio UCI de especial complejidad, tomando como referencia que la tasa de error en clasificación esté en torno al 20% con clasificadores como C4.5 [Quinlan, 1993] o 1-NN [Cover y Hart, 1967; Aha, Kibler y Albert, 1991]. Así mismo, la nueva metodología se ha aplicado a un problema real de trasplantes hepáticos en España. El análisis estadístico de los resultados revela que nuestra propuesta actual mejora significativamente la precisión sobre el conjunto de generalización y es mucho más eficiente que nuestra metodología previa, TSEA. En total se ha experimentado con 19 bases de datos y 4 filtros, llevando a cabo una comparación con otros clasificadores basados en redes neuronales como RBF [Howlett y Jain, 2001] o MLP [Bishop, 1995].

2. Antonio J. Tallón-Ballesteros y César Hervás-Martínez (2011). A two-stage algorithm in evolutionary product unit neural networks for classification. *Expert Systems with Applications*, 38(1), 743-754, Estados Unidos. Factor de impacto: 2,203 Q1 [Tallón-Ballesteros y Hervás-Martínez, 2011].

En este artículo se propone un procedimiento para añadir mayor diversidad al comienzo del proceso evolutivo. Consiste en crear dos poblaciones iniciales con diferentes ajustes de parámetros, evolucionarlas durante un número pequeño de generaciones, seleccionar los mejores individuos de cada población en la misma proporción y combinarlos para constituir una nueva población inicial. En este punto, el bucle principal del algoritmo evolutivo se aplica a la nueva población. La propuesta introducida ha sido denominada TSEA (*Two-Stage Evolutionary Algorithm*, algoritmo evolutivo en dos etapas). Se ha llevado a cabo la experimentación con 12 bases de datos del repositorio del UCI y 2

problemas complejos del mundo real (*BTX* y *Listeria monocytogenes*). *BTX* (compuesto formado por Benzeno, Tolueno y Xyleno) es un problema de clasificación multiclase relativo al Medio Ambiente que considera diferentes tipos de agua contaminadas [Hervás et al., 2008]. *Listeria monocytogenes* [Beuchat, 1996] es un problema biclase en Microbiología Predictiva de especial interés relacionado con las industrias alimentarias debido a la aparición de este patógeno en el entorno natural y las condiciones específicas que llevan a su alto predominio en diferentes tipos de productos de alimentación. Los resultados indican que nuestra propuesta mejora considerablemente la eficiencia y significativamente su eficacia en la mayoría de las bases de datos.

1.5.2. Capítulo de libro

Se ha publicado un capítulo de libro en la serie *Studies in Computational Intelligence* de la editorial Springer:

1. Antonio J. Tallón-Ballesteros, César Hervás-Martínez y Pedro A. Gutiérrez (2013). An extended approach of a two-stage evolutionary algorithm in artificial neural networks for multiclassification tasks. En: I. Jordanov y L. C. Jain (Eds.). *Innovations in Intelligent Machines – 3: Contemporary Achievements in Intelligent Systems*. Serie: Studies in Computational Intelligence, vol. 442. Ed. Springer-Verlag Berlin Heidelberg, pp. 139-153. ISBN: 978-3-642-32176-4. [Tallón-Ballesteros, Hervás-Martínez y Gutiérrez, 2013].

Este capítulo extiende el algoritmo TSEA [Tallón-Ballesteros y Hervás-Martínez, 2011] a RNAs de tipo sigmoide. Con el citado algoritmo se realiza una evolución simultánea de arquitecturas y pesos. La experimentación se ha realizado con 14 bases de datos –12 procedentes del

repositorio del UCI y 2 de problemas complejos del mundo real, *BTX* y *Listeria monocytogenes* que han sido explicados en la segunda publicación de la categoría de revistas internacionales: [Tallón-Ballesteros y Hervás-Martínez, 2011]– que difieren en el número de instancias, atributos y clases. Los resultados obtenidos con nuestra nueva propuesta, TSEASig (*Two-Stage Evolutionary Algorithm for neural networks with Sigmoidal units*, algoritmo evolutivo en dos fases para redes neuronales con unidades sigmoide), han sido comparados frente a los de TSEA y EDDSig. EDDSig (*Experimental Design Distribution with Sigmoidal units,* distribución del diseño experimental con unidades sigmoide) es una nueva aportación que se presenta por vez primera en este artículo y que consiste en hacer uso de US en la metodología EDD en lugar de UP. Los resultados han sido contrastados con tests estadísticos y muestran que nuestra nueva propuesta mejora significativamente la precisión sobre el conjunto de generalización respecto a los obtenidos con una metodología estándar basado en una única población. Además, la nueva propuesta es mucho más eficiente que el resto de métodos desarrollados previamente por nosotros. TSEASig obtiene resultados competitivos respecto a TSEA. Si bien los de este último son ligeramente superiores, no existen diferencias significativas.

1.5.3. Congresos

Se han presentado tres trabajos en congresos, dos de los cuales son internacionales:

1. Antonio J. Tallón-Ballesteros, César Hervás-Martínez, José C. Riquelme y Roberto Ruiz (2011). Improving the accuracy of a two-stage algorithm in evolutionary product unit neural networks for classification by means of feature selection. En: J. M. Ferrández, J. R.

Álvarez Sánchez, F. de la Paz y F. J. Toledo (Eds.). *Proceedings of the 4th International work-conference on the Interplay Between Natural and Artificial Computation* (IWINAC 2011). Lecture Notes in Computer Science (LNCS) 6687 (vol. II). Ed. Springer, La Palma (Islas Canarias), Spain, pp. 381-390. ISBN: 978-3-642-21325-0 [Tallón-Ballesteros et al., 2011].

Este trabajo plantea una metodología que mejora la precisión del algoritmo evolutivo en dos fases, presentado en el artículo publicado en la revista *Expert System with Applications* [Tallón-Ballesteros y Hervás-Martínez, 2011] sobre redes neuronales evolutivas de unidades producto para tareas de clasificación por medio de selección de atributos. Han sido considerados un par de filtros para probar la propuesta. La metodología mejorada se ha denominado TSEAFS (*Two-Stage Evolutionary Algorithm with Feature Selection*, algoritmo evolutivo en dos fases con selección de atributos). La experimentación ha sido realizada en 7 bases de datos del repositorio UCI que presentan tasas de error sobre el conjunto de generalización en torno al 20% con clasificadores de referencia como C4.5 o 1-NN. El análisis estadístico de los resultados confirma que nuestra propuesta actual mejora significativamente la precisión sobre el conjunto de test y es mucho más eficiente que nuestra metodología previa, TSEA.

2. Antonio J. Tallón-Ballesteros, Pedro A. Gutiérrez-Peña y César Hervás-Martínez (2007). Distribution of the search of Evolutionary Product Unit Neural Networks for Classification. En: N. Guimaraes y P. Isaías (Eds.). *Proceedings of the IADIS Internacional Conference Applied Computing* (AC 2007). Ed. IADIS, Salamanca, Spain, pp. 266-273. ISBN: 978-972-8924-30-0 [Tallón-Ballesteros, Gutiérrez-Peña y Hervás-Martínez, 2007].

Este artículo trata el procesamiento distribuido en la búsqueda de modelos de clasificación óptima mediante redes neuronales evolutivas de unidades producto. Se presentan dos propuestas denominadas distribución del diseño experimental (EDD, *Experimental Design Distribution*) y distribución del procesamiento, respectivamente. La experimentación se ha llevado a cabo con bases de datos del repositorio del UCI [Newman et al., 1998]. En la primera propuesta analizamos el rendimiento de manera independiente para base de datos de tamaño medio (con menos de 1.000 instancias) y de tamaño grande (entre 1.000 y 5.000 instancias). En la segunda, se evalúa la distribución del procesamiento con bases de datos de tamaño medio, obteniendo la ganancia y la eficiencia.

3. Antonio J. Tallón-Ballesteros, C. Hervás-Martínez, P. A. Gutiérrez y P. Jiménez (2007). Distribución de Modelos de Redes Neuronales Evolutivas de Unidades Producto para Clasificación. En: F. Almeida Rodríguez, B. Melián Batista, J. A. Moreno Pérez y J. M. Moreno Vega (Eds.). *Actas del V Congreso Español sobre Metaheurísticas, Algoritmos Evolutivos y Bioinspirados* (MAEB 2007). Ed. Thomson, Puerto de la Cruz, Tenerife, Spain, pp. 151-158. ISBN: 978-84-690-3470-5 [Tallón-Ballesteros et al., 2007].

El tema en el que se centra este trabajo es la distribución del procesamiento asociado a la búsqueda de los mejores modelos de redes neuronales de tipo unidad producto. Se han presentado tres modelos que distribuyen ciertos parámetros del algoritmo evolutivo o de la topología de la red neuronal o bien las diferentes iteraciones que constituyen la experimentación. Los modelos han sido evaluados en varias bases de datos procedentes del repositorio UCI. Los resultados han sido comparados con los obtenidos mediante modelos de UP estándar.

1.6. PROYECTOS DE INVESTIGACIÓN

El doctorando ha participado y continúa participando en diversos proyectos de investigación. Seguidamente, se enumeran por orden cronológico inverso tanto los proyectos vigentes como los concluidos:

- 2012-2015. Participación como investigador en el Proyecto I+D "Modelos Avanzados para el Análisis Inteligente de Información. Aplicación a Datos Biomédicos y Medioambientales". Referencia: TIC-7528. Responsable: Cristina Rubio Escudero. Tipo de Proyecto: Proyectos de Excelencia de la Junta de Andalucía. Duración: 4 años. Organismo/Empresa financiador/a: Junta de Andalucía (Consejería de Innovación, Ciencia y Empresa). Número de investigadores: 7. Importe concedido: 31.435,25 €.

- 2012-2014. Participación como investigador en el Proyecto I+D "Análisis Inteligente de Información Medioambiental". Referencia: TIN2011-28956-C02-02. Responsable: José Cristóbal Riquelme Santos. Tipo de Proyecto: Plan Nacional del 2011. Fecha de inicio: 01/10/2012. Fecha de finalización: 31/12/2014. Organismo/Empresa financiador/a: Ministerio de Ciencia e Innovación. Número de investigadores: 13. Importe concedido: 47.008,50 €.

- 2007-2012. Participación como investigador en el Proyecto I+D "Heurísticas escalables para la extracción de conocimiento en grandes volúmenes de información". Referencia: TIN2007-68084-C02-02. Responsable: José Cristóbal Riquelme Santos. Tipo de Proyecto: Plan Nacional del 2007. Fecha de inicio: 01/10/2007. Fecha de finalización: 31/03/2012. Organismo/Empresa financiador/a: Ministerio de

Educación y Ciencia. Número de investigadores: 12. Importe concedido: 99.220 €.

1.7. ORGANIZACIÓN

Esta Tesis Doctoral se compone de siete capítulos. En el capítulo actual enunciamos el problema que nos planteamos resolver, el contexto de investigación, los objetivos que se persiguen alcanzar, se formulan las hipótesis de partida y enumeramos las contribuciones realizadas y los proyectos de investigación, tanto los concluidos como los vigentes.

En el capítulo 2 se describe el estado del arte relativo a Redes Neuronales Artificiales y el capítulo 3 presenta una revisión bibliográfica sobre Selección de Atributos. Los modelos implementados se detallan en el capítulo 4 y el diseño experimental se muestra en el capítulo 5.

Los resultados logrados con las diferentes propuestas, junto a una discusión conjunta de los mismos se exponen en el capítulo 6 y las conclusiones obtenidas, así como las futuras líneas de investigación figuran en el capítulo 7. Finalmente, se recoge un listado con las referencias bibliográficas que han sido citadas en la Tesis.

Capítulo 2

Redes Neuronales Artificiales

2.1. TERMINOLOGÍA

Una Red Neuronal Artificial (RNA) es un paradigma de procesamiento de información inspirado en la forma en la que los sistemas nerviosos biológicos procesan la información. El órgano más representativo de tales sistemas es el cerebro humano. La estructura del cerebro se compone de un gran número de elementos de procesamiento interconectados (neuronas) que trabajan al unísono para resolver problemas específicos. Al igual que los humanos, las RNAs aprenden a través de ejemplos. Una RNA se configura para una aplicación específica, como reconocimiento de patrones o clasificación de datos, mediante un proceso de aprendizaje. En los sistemas biológicos aprender implica ajustes en las conexiones sinápticas que existen entre neuronas [Yegnanarayana, 1999].

Existen múltiples definiciones, entre las que conviene destacar algunas. Por un lado, la que propone Simon Haykin en el libro *Neural Networks. A Comprehensive Foundation* [Haykin, 1999]. Una red neuronal es un procesador distribuido, masivamente paralelo, que tiene una tendencia natural para almacenar conocimiento experiencial y hacerlo disponible para su uso. Éste se asemeja al cerebro en dos aspectos: a) el conocimiento

es adquirido por la red a través de un proceso de aprendizaje y b) las fuerzas de conexión entre neuronas, conocidas como pesos sinápticos, se emplean para almacenar el conocimiento. Por otro lado, Jacek M. Zurada [Zurada, 1992] define una red neuronal como un sistema físico que puede adquirir, almacenar y utilizar el conocimiento de la experiencia.

Desde el punto de vista funcional, una RNA es un conjunto de unidades de procesamiento, conocidas como nodos o unidades, las cuales están conectadas entre sí. Con el propósito de entender mejor en qué consiste una neurona, representamos a continuación de manera gráfica en la figura 2.1 una neurona abstracta. La neurona tiene *n* entradas, representadas por x_i. Cada entrada transmite una señal, cuya intensidad es ponderada mediante un peso (w_i). La neurona posee una función, denominada función de activación, que realiza una correspondencia entre el nivel de señal total que recibe, expresada habitualmente como una suma ponderada, y la salida de dicha neurona; las ponderaciones corresponden al valor de cada entrada ponderado por su correspondiente peso.

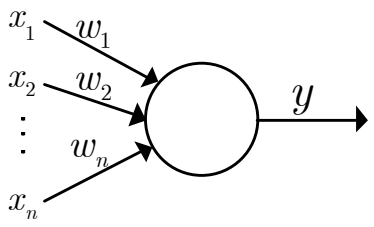


Figura 2.1. Neurona abstracta.

Matemáticamente, la salida de la neurona abstracta viene dada por:

$$y = f(w_1 x_1 + w_2 x_2 + \ldots + w_n x_n) \tag{2.1}$$

siendo f la función de activación de la neurona.

La disposición de los nodos y las conexiones de una RNA se conoce como topología [Simpson, 1990].

Dependiendo de la ubicación, los nodos o neuronas de la red pueden ser de tres tipos:

1) *Nodos de entrada*, que reciben las señales de entrada de la red, esto es, los valores de las variables independientes del modelo y forman la capa de entrada.
2) *Nodos de salida*, que envían las señales de salida al exterior, esto es, los valores de las variables dependientes del modelo y forman la capa de salida.
3) *Nodos ocultos*, que son el resto de nodos y se agrupan en una o más capas ocultas.

En sentido estricto, los nodos de computación *per se* de una RNA son los nodos ocultos y los de salida. Usualmente, las capas de una RNA se numeran mediante los números ordinales, siendo la primera capa la capa de entrada y la última, la capa de salida. En las redes de tres capas, tendremos, por tanto, una capa de cada tipo, es decir, la capa de entrada (primera capa), la capa oculta (segunda capa) y la capa de salida (tercera capa). A partir de lo anterior, se definen las capas adyacentes como aquellas capas que difieren en un número ordinal. En algunos casos, en la capa de entrada y/o en la capa oculta pueden existir nodos de sesgo, que representan un valor umbral de activación de la neurona. Por otra parte, es común representar una topología concreta indicando el número de nodos en cada capa, desde la primera hasta la última separado por dos puntos. En esta nomenclatura, para denotar la topología no es frecuente incluir los nodos de sesgo, por consiguiente, en esta Tesis no se consideran contabilizados en cada una de las capas donde existan.

Cada unidad de procesamiento se compone de un conjunto de conexiones de entrada, una función de activación (encargada de calcular la entrada total combinada de todas las conexiones), un núcleo central de proceso (encargado de aplicar la función de activación) y una salida o función de transferencia (por donde se transmite el valor de activación a otras unidades). De esta forma, los elementos que caracterizan un nodo de una RNA son:

- *Conexiones ponderadas*: juegan el papel de las conexiones sinápticas. La existencia de conexiones determina si es posible que un nodo influya sobre otro, mientras que el valor de los pesos y el signo de los mismos definen el tipo (excitatorio/inhibitorio) y la intensidad de la influencia.
- *Función de activación*: calcula el valor de base o entrada total al nodo, generalmente, como simple suma ponderada de todas las entradas recibidas, es decir, en primer lugar, se calcula para cada entrada su intensidad, habitualmente, multiplicada por el correspondiente peso y, a continuación, se suman todas las intensidades ponderadas. Representaremos con $a_j = f(\boldsymbol{x}, \boldsymbol{w})$ el valor obtenido por el nodo *j*, siendo $\boldsymbol{x}$ el vector de variables de entrada y $\boldsymbol{w}$ el vector de pesos; a_j también se conoce como valor de activación. Expresado en otros términos, la función de activación f transforma el estado actual del nodo en una señal de salida.
- *Función de salida o de transferencia*: a partir de la activación del nodo, calcula la salida del mismo. Se denota como $h(\bullet)$ y se usan diferentes tipos de funciones, desde simples funciones de umbral a funciones no lineales. El valor final de salida ϕ_j para un vector $\boldsymbol{x}$ de un nodo oculto viene dado por:

$$\phi_j(\boldsymbol{x}) = h\left(f\left(\boldsymbol{x}, \boldsymbol{w}\right)\right) \tag{2.2}$$

Existen tres tipos de conexiones entre neuronas [Fiesler, 1994]:

1) *Conexión inter-capa*. Es una conexión entre neuronas en capas adyacentes.
2) *Conexión intra-capa*. Es una conexión entre neuronas de la misma capa.
3) *Conexión supra-capa*. Es una conexión entre neuronas que no son adyacentes ni están en la misma capa. Es decir, enlazan diferentes capas obviando al menos una capa.

Desde el punto de vista de la transmisión de la información, las conexiones pueden ser asimétricas o unidireccionales y bidireccionales. En el primer caso, la conexión es transversal en un sentido y, en el segundo, la conexión actúa en dos sentidos, siendo la fuerza de la conexión (peso) la misma en ambos sentidos.

El grafo permite representar el esquema de conexiones de una RNA. La figura 2.2 ilustra la topología de una RNA abstracta. Como podemos observar, presenta 3 capas: la capa de entrada que tiene *n* nodos, la capa oculta que tiene *M* nodos y la capa de salida con 2 nodos, por tanto, su topología es n: M: 2. Por otra parte, percibimos que todas las conexiones son inter-capa, entre capas adyacentes, excepto una, que es supra-capa y enlaza el nodo s_1 de la capa de salida al nodo x_1 de la capa de entrada.

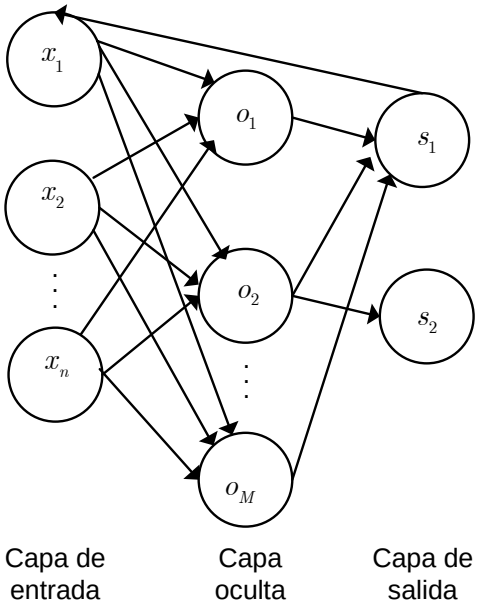


Figura 2.2. Red neuronal artificial abstracta.

2.2. TAXONOMÍA DE MODELOS

Existen muchos tipos de redes neuronales, como dan cuenta libros específicos como [Hertz, Krogh y Palmer, 1991]. A continuación, presentamos diferentes criterios que permiten establecer varias taxonomías según la transmisión de información, el tipo de función de base, la naturaleza de los algoritmos de aprendizaje, el tipo de arquitectura y el tipo de unidad.

2.2.1. Modelos de redes neuronales según la transmisión de información

Las RNAs, dependiendo de la transmisión de información o la dirección de propagación de la señal, se clasifican en:

- *Redes con conexiones hacia adelante*. También conocidas como redes *feedforward*. La señal se propaga desde la capa menos profunda (capa de entrada) a la capa más profunda (capa de salida), atravesando las

capas intermedias (capas ocultas). Por tanto, nunca se hacen transmisiones hacia capas anteriores. En la figura 2.3a) se muestra una red *feedforward* con topología 3: 3: 2. La flexibilidad de las redes *feedforward* de tres capas ha sido bien documentada y se resume por la propiedad de aproximación universal.

Formalmente, tal como se define en [Bishop, 2006], las RNAs con conexiones o con transmisión hacia adelante y una sola capa oculta no son más que una forma de un modelo de regresión lineal, que considera una combinación lineal de transformaciones no lineales ($\phi_j(\boldsymbol{x},\boldsymbol{\beta})$) de las variables de entrada. Esto nos lleva a la definición de un modelo de red neuronal básico, formado por una serie de transformaciones funcionales, que viene dado por la siguiente expresión:

$$\phi_j(\boldsymbol{x},\boldsymbol{\beta}) = \beta_0 + \sum_{j=1}^{M} \beta_j \phi_j(\boldsymbol{x}) \tag{2.3}$$

siendo *M* el número de transformaciones no lineales, $\boldsymbol{\beta} = (\beta_0,...,\beta_M)$ el valor de los coeficientes asociados al modelo, $\phi_j(\boldsymbol{x},\boldsymbol{\beta})$ las funciones de base y $\boldsymbol{x} = (x_1,...,x_n)$ las variables de entrada o variables independientes del problema. Este tipo de modelos son conocidos como *modelos lineales de funciones de base* (del inglés *linear basis function models*) [Bishop, 2006]. La regresión polinomial es un caso particular de estos modelos en el que existe una única variable y cada función de base toma la forma de potencia de dicha variable, $\phi_j(\boldsymbol{x},\boldsymbol{\beta}) = x^j$. Una limitación de la funciones de base polinómica es el hecho de que son funciones globales de la variable de entrada, por lo que los cambios en una región del espacio de entrada afectan a otras regiones. Esto se puede resolver dividiendo el espacio de entrada en regiones y ajustando un polinomio diferente en cada región, dando lugar a funciones spline o splines [Hastie, Tibshirani y Friedman, 2008]. Existen otras muchas posibilidades en la elección de la tipología de

las funciones de base, por ejemplo, las funciones de base Gaussianas, que dan lugar a las Redes Neuronales de Funciones de Base Radial (Redes RBF, del inglés *Radial Basis Function*) [Bishop, 1991; Lee y Hou, 2002], las funciones de base de Unidad Sigmoide, que dan lugar al Perceptrón Multicapa (MLP, del inglés *Multi-Layer Perceptron*) [Cybenko, 1989], o las Unidades Producto (UP) [Durbin y Rumelhart, 1989].

- *Redes recurrentes*. Conocidas también como redes *feedback*. En este caso, algunas conexiones van de una capa hacia otra capa previa o bien hay nodos de la misma capa conectados entre sí. Un ejemplo de una red recurrente, cuya topología es 3: 3: 2, aparece en la figura 2.3b); en este caso particular, existen conexiones desde los nodos de la capa de salida a algunos nodos de la capa de entrada.

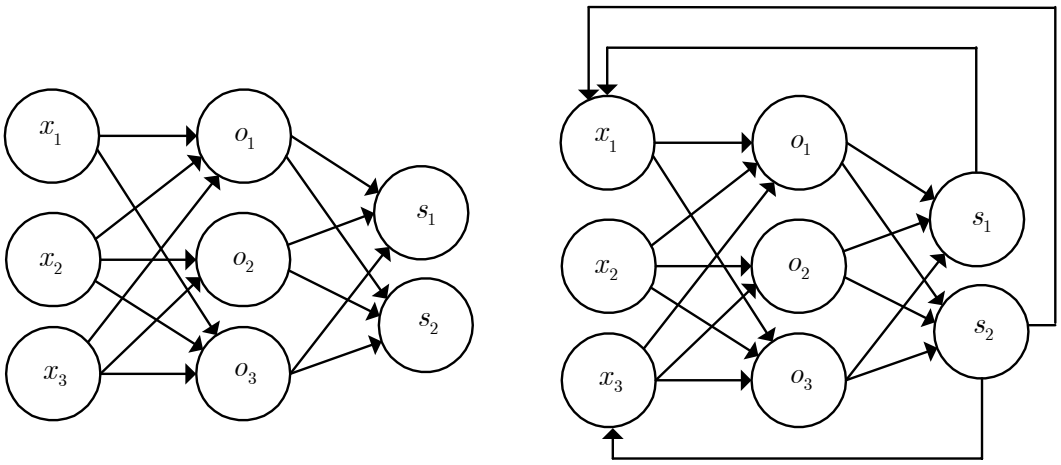


Figura 2.3. Ejemplos de Redes Neuronales:
a) Red Feedforward con topología 3: 3: 2; b) Red Recurrente con topología 3: 3: 2.

2.2.2. Modelos de redes neuronales según el tipo de función de base

Según el tipo de función de base o función de red, encontramos los siguientes tipos:

- *Basadas en función de base de tipo proyección*. Las funciones de proyección son funciones de entorno global como las US o las UP. Al ser globales, presentan dificultades en la aproximación de datos aislados, pero suelen actuar mejor en problemas donde la dimensión del espacio de entrada, esto es el número de entradas, es elevado.
- *Basadas en funciones de base de tipo kernel*. Las funciones de kernel son funciones de entorno local, como pueden ser las RBFs. Poseen una mayor capacidad para aproximar datos anómalos aislados, pero también una mayor dificultad en entornos globales o cuando el número de variables de entrada es demasiado alto.

En [Donoho y Johnstone, 1989], se demuestra la dualidad entre la aproximación basada en funciones de proyección y la basada en funciones tipo kernel. De este modo, una función se puede descomponer en dos funciones, una radial y la otra proyectiva, siendo ambas funciones mutuamente excluyentes.

2.2.3. Modelos de redes neuronales según el conocimiento sobre la variable de salida

Existen varios tipos de redes atendiendo al conocimiento que tengamos sobre la variable de salida:

- *Redes neuronales supervisadas*. Los datos que se proporcionan a dichas redes consisten en una serie de entradas y sus respectivos valores

objetivo. Dichos valores objetivo actúan como las especificaciones de un maestro para que la red en el proceso de entrenamiento, esto es, durante la obtención del modelo, sepa para cada entrada cuál es la salida esperada y poder evaluar la capacidad del modelo con los datos conocidos. Dentro de esta categoría encontramos, entre otras, las siguientes redes: red de Hopfield [Hopfield, 1982], red de Hamming, perceptrón y perceptrón multicapa.

- *Redes neuronales no supervisadas*. También conocidas como redes de autoorganización o redes autoorganizativas. En este caso, la red recibe los datos como un conjunto de entradas sin valores objetivo. Como un caso típico de aplicación de este tipo de redes, encontraríamos la separación de un conjunto de patrones en grupos haciendo uso de ciertas propiedades de similaridad o disimilaridad que permiten llevar a cabo tal división. Algunos modelos de redes que se engloban en este tipo serían los mapas de características auto-organizadas de Kohonen [Kohonen, 1984], la red de Carpenter y Grossberg.

- *Redes neuronales basadas en refuerzo*. Se puede considerar como un modelo intermedio entre los anteriores. Como información de entrada, se proporcionan un conjunto de patrones o instancias y, una vez que la red obtiene la salida, se le indica si es correcto o no, aunque no se le proporciona el valor esperado de salida. Un ejemplo de este tipo serían las redes neuronales aleatorias [Gelenbe, 1989] que habitualmente están basadas en aprendizaje por refuerzo. Una característica de dicho tipo de red es el empleo de diferentes señales que, en la terminología, se conocen como señales positivas y señales negativas. A menudo, dicho tipo de redes se emplean en sistemas de control.

De los tres modelos que acabamos de describir, los más comunes con diferencia son los dos primeros.

2.2.4. Modelos de redes neuronales según el tipo de unidad

Atendiendo al tipo de unidad o nodo, encontramos dos modelos de redes:

- *El modelo aditivo*. Es el usado con más frecuencia dentro de la teoría de RNAs. La red está compuesta de unidades aditivas (UAs); la salida de cada unidad o nodo es función de la suma de las entradas ponderadas con los correspondientes pesos más el valor de activación o sesgo. El tipo de función de activación da lugar a diferentes tipos de neuronas aditivas, entre las cuales cabe destacar las neuronas umbral o perceptrón (que utilizan una función tipo escalón), las neuronas tipo sigmoide (que utilizan funciones de activación logísticas sigmoidales, tangente hiperbólica o arcotangente) y las neuronas lineales (cuya función de activación es la función identidad, también denominada función lineal). En la figura 2.4 se expone la representación de una UA genérica y de una UA sigmoide.

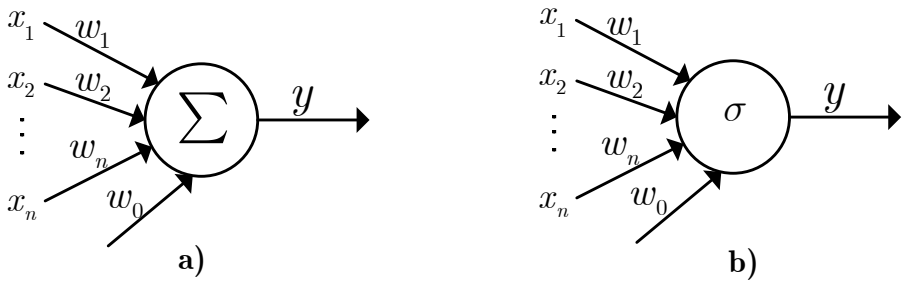


Figura 2.4. a) Unidad aditiva genérica; b) unidad aditiva sigmoide.

La expresión de salida de una UA genérica y una UA sigmoide se refleja, respectivamente, en las ecuaciones (2.4) y (2.5):

$$y = w_0 + \sum_{i=1}^{n} w_i x_i; \quad w_0, w_i \in \mathbb{R} \tag{2.4}$$

$$y = \frac{1}{1 + e^{-\left(w_0 + \sum_{i=1}^{n} w_i x_i\right)}}; \quad w_0, w_i \in \mathbb{R} \tag{2.5}$$

La motivación para usar modelos de red basados en unidades aditivas se basa en la relevancia de los primeros datos biológicos, como el modelo original propuesto por Warren S. McCulloch y Walter H. Pitts en los trabajos [McCulloch y Pitts, 1943; Pitts y McCulloch, 1947]. La investigación ha demostrado que las redes neuronales basadas en unidades aditivas (RNUA) pueden aproximar cualquier función continua hasta un arbitrario grado de precisión, siempre que las capas ocultas contengan un número suficiente de unidades ocultas [Funahashi, 1989; Hornik, Stinchcombe y White, 1990]. Sin embargo, estas redes requieren para la aproximación de funciones complejas un gran número de UAs comparando con combinaciones de las entradas de orden superior [Leerink, Horne y Jabri, 1995]. Por otra parte, la comunidad científica coincide en que sólo existe una correspondencia menor entre estos modelos de neuronas y el comportamiento de las neuronas biológicas, en concreto, la interacción de las entradas sinápticas es esencialmente no lineal [Koch, 1999].

Dentro de este modelo, las redes más extendidas, que están formadas por unidades aditivas y presentan una función de activación sigmoidal, se denominan Redes Neuronales de Unidad Sigmoide (RNUS). Dichas redes se describen en el apartado 2.4.2 y constituyen uno de los dos tipos de red que empleamos en la presente Tesis.

- *El modelo multiplicativo*. La red está constituida por unidades que multiplican sus entradas en lugar de sumarlas. De esta manera, se obtienen las redes neuronales multiplicativas. La operación de multiplicación aumenta las capacidades computacionales y de almacenamiento de las redes neuronales; dicha afirmación se justifica a partir de extensiones de RNA donde dicha operación aparece en la forma de una unidad de orden superior [Giles y Maxwell, 1987]. A las unidades multiplicativas también se les denomina conexiones multiplicativas, debido a que los valores de salida de dos o más unidades se multiplican antes de llevar a cabo la típica suma para calcular la salida global. Dicha conexión multiplicativa permite a una unidad enlazarse con otra unidad [Rumelhart y McClelland, 1986]. En el trabajo [Schmitt, 2001] aparece un estudio muy detallado del modelo multiplicativo, el cual intenta afrontar aquellos casos en los que existe una interacción entre las variables y las regiones de decisión no pueden separarse por hiperplanos.

Desde el punto de vista biológico, existen evidencias de la capacidad del sistema nervioso animal de realizar operaciones de tipo multiplicación; más exactamente las de tipo multiplicativo se pueden encontrar en todos los niveles de procesamiento de información neuronal [Koch y Poggio, 1992].

El tipo más simple de unidad multiplicativa es el monomio, que viene dado por:

$$M = x_1^{w_1} \cdot x_2^{w_2} \cdot \ldots \cdot x_n^{w_n}; \quad w_i \in \{0,1,2,\ldots\}, \quad i = 1,2,\ldots,n \tag{2.6}$$

Cada peso se conoce como grado o exponente de la variable a la que afecta. La suma de los pesos se denomina orden del monomio.

A partir de (2.6), el polinomio es una suma algebraica de monomios y viene dado por:

$$P = w_1 M_1 + w_2 M_2 + \ldots + w_n M_n; \quad w_i \in \mathbb{R}, \;\; i = 1, 2, \ldots, n \tag{2.7}$$

que se conoce con el nombre de unidad sigma-pi [Rumelhart, Hinton y McClelland, 1986].

Dentro de este modelo, la alternativa más general es la unidad producto (UP), que fue propuesta por Richard Durbin y David E. Rumelhart en 1989 [Durbin y Rumelhart, 1989].

Matemáticamente, una UP viene dada por:

$$y = x_1^{w_1} \cdot x_2^{w_2} \cdot \ldots \cdot x_n^{w_n} = \prod_{i=1}^{n} x_i^{w_i}; \quad w_i \in \mathbb{R} \tag{2.8}$$

En la figura 2.5 se muestra una unidad producto.

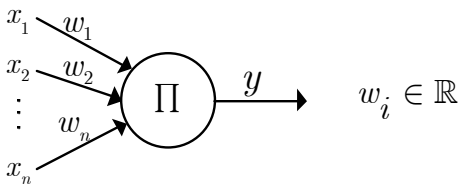


Figura 2.5. Unidad producto.

A diferencia del monomio, en la UP los exponentes no son números enteros no negativos de valor fijo, sino que son números reales, cuyo valor es variable, con la consiguiente posibilidad de ser entrenados [Durbin y Rumelhart, 1990]. Por tanto, una UP es, al menos, tan potente computacionalmente como un monomio. Además, se pueden expresar ratios entre variables y, por tanto, la división usando pesos negativos.

Explícitamente, en el trabajo original los autores recalcan que el propósito de las UP no es reemplazar a las UA sino combinarlas; lo más

atractivo, mencionan, es mezclar ambos tipos, alternando capas de UA con capas de UP. Comparando con las unidades sigma-pi, que llevan a cabo la misma tarea, las UP no aumentan el número de parámetros, debido a que sólo presentan un peso por cada entrada, como las UAs [Durbin y Rumelhart, 1989].

Otra ventaja que éstos y otros autores han explorado es la posibilidad de que los exponentes de las UP puedan ser ajustados automáticamente, por ejemplo, mediante métodos basados en el gradiente y otros métodos de aprendizaje [Durbin y Rumelhart, 1989; Leerink et al. 1995a; Leerink et al. 1995b]. Así mismo, como base para modelar redes neuronales biológicas, en [Durbin y Rumelhart, 1990] se propuso extender el modelo MLP clásico a unidades de tipo multiplicativo o producto.

Por otra parte, son frecuentes las situaciones en el modelado de datos reales, entre las que destacan leyes físicas, en las cuales la relación entre las variables responde a una estructura de tipo potencial en la que las potencias no están restringidas a los números naturales o enteros.

Kazumi Saito y Ryohei Nakano [Saito y Nakano, 1997a; Saito y Nakano, 1997b] desarrollaron el algoritmo RF5, basado en unidades producto, y experimentaron con varias leyes físicas, entre las que destaca *La tercera ley de Kepler o ley armónica* enuncia que "para cualquier planeta, el cuadrado de su período orbital (T) es directamente proporcional al cubo de la longitud del semieje mayor de su órbita elíptica (r)". Matemáticamente, viene dada por:

$$T^2 = kr^3 \tag{2.9}$$

Los resultados obtenidos con RF5 aproximan con bastante exactitud la expresión analítica de la mencionada ley.

Las redes que emplean unidades producto se denominan Redes Neuronales de Unidad Producto (RNUP). Se hace necesario recalcar que para tener una RNUP sólo tenemos que tener nodos de tipo unidad producto, aunque no es necesario que sea de manera exclusiva. Las RNUP se describen en el apartado 2.4 junto con las RNUS. Ambos tipos de redes son los empleados en esta Tesis.

Las unidades más conocidas que están compuestas de *sinapsis* multiplicativas son, probablemente, las neuronas de orden superior que son los elementos básicos de las Redes Neuronales de Orden Superior (RNOS) [Lee et al., 1986; Giles y Maxwell, 1987]; dichas redes han sido desarrolladas para mejorar la habilidad de expresar la no linealidad que poseen las redes *feedforward*; la diferencia básica radica en que en las primeras se hace uso de la operación de multiplicación en sustitución de la suma. Consideremos una unidad de orden superior, con entradas x_j, x_k, x_l,…, cuya salida y viene dada por:

$$y = f\left(\sum_j w_j x_j + \sum_{j,k(j\le k)} w_{jk} x_j x_k + \sum_{j,k,l(j\le k\le l)} w_{jkl} x_j x_k x_l + ...\right) \qquad (2.10)$$

Como podemos observar en este caso, cada entrada aparece en agrupaciones de términos formados por una, dos, etc. entradas, de tal manera que cada agrupación de una, dos, etc. entradas tiene un peso específico que viene expresado mediante el uso de uno, dos, etc. subíndices en el correspondiente peso.

Dentro de las RNOS hallamos diferentes variantes como son:

- *Redes pi-sigma*. Fueron presentadas en 1991 [Shin y Ghosh, 1991] con el propósito de incorporar las capacidades de las RNOS usando un

menor número de pesos y unidades de procesamiento. En la figura 2.6 apreciamos una red pi-sigma con una salida. La red consta de *n* entradas, una capa con *k* unidades aditivas y una capa de salida con una unidad de tipo producto.

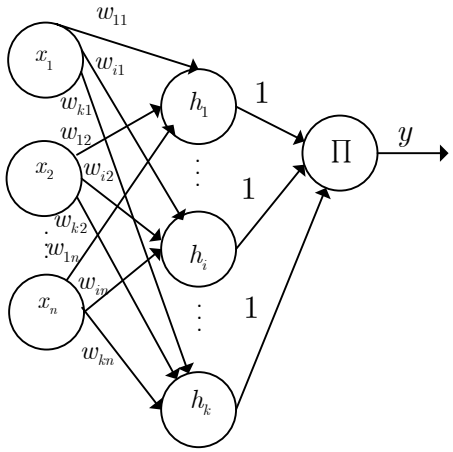


Figura 2.6. Red pi-sigma con una salida.

La salida de la unidad aditiva *i* viene dada por:

$$h_i = \sum_{k=1}^{n} w_{ik} x_k; \quad i = 1, 2, ..., k \tag{2.11}$$

La expresión del nodo de salida es la siguiente:

$$y = \sigma\left(\prod_{i=1}^{k} h_i \right) \tag{2.12}$$

donde $\sigma(.)$ denota la función de activación.

- *Redes polinomiales en cadena* (del inglés *Ridge Polynomial Networks*). Fueron desarrolladas en 1992 [Shin y Ghosh, 1992] y constituyen una generalización de las redes pi-sigma. Están estrechamente relacionadas con una técnica de regresión estadística llamada *projection pursuit regression* [Friedman y Stuetzle, 1981]. Una red polinomial en cadena usa como bloques básicos de construcción las

redes pi-sigma. Por tanto, ahora la capa oculta está compuesta por la combinación de las unidades aditivas de la capa oculta y la capa de salida de las redes sigma-pi.

2.3. Redes Neuronales Aplicadas a Problemas de Clasificación

Las redes neuronales presentan múltiples aplicaciones entre las que destacan: Regresión, Asociación, Clasificación, etc. En la presente Tesis nos centramos exclusivamente en tareas de clasificación [Anthony y Bartlett, 2009]. Por tanto, en primer lugar vamos a enunciar en qué consiste el problema de clasificación.

El objetivo de un problema de clasificación es predecir la clase de pertenencia de nuevos patrones o ejemplos de estudio, a partir del modelo obtenido por el clasificador con una serie de patrones que son proporcionados como entrada. Tom Mitchell, en su libro titulado *Machine Learning* [Mitchell, 1997], define los problemas de clasificación como aquéllos en los que la tarea es clasificar ejemplos en una de un conjunto discreto de posibles categorías.

Formalmente, los patrones de entrenamiento tienen la forma $D = (x_n, y_n)$; $n = 1, 2, ..., N$, donde $\boldsymbol{x_n} = (x_{1n}, ..., x_{kn})$ es el vector con las k variables de entrada e $y_n \in \{1, 2, ..., J\}$ es la clase del patrón n-ésimo. Las medidas x_i, $i = 1, 2, ..., k$ son tomadas en un único patrón que debe ser clasificado en una de las J clases basándonos en dichas medidas.

En la obtención de un modelo de clasificación, podemos distinguir dos fases: entrenamiento y generalización. En la fase de entrenamiento, se emplea una cantidad limitada de datos de entrenamiento y conocimiento a

priori, relativo al dominio del problema para ajustar los parámetros y/o aprender la estructura del clasificador. Durante la fase de generalización, el clasificador diseñado en la fase de entrenamiento se evalúa con nuevos datos (datos de generalización) proporcionando una decisión de clasificación para cada patrón de entrada. Es importante destacar que, para que la evaluación sea justa, los datos de generalización nunca pueden ser usados para estimar los parámetros del clasificador o para determinar su estructura.

Por tanto, como paso previo a la obtención de clasificación es imprescindible disponer de los datos del problema, que deben estar divididos en dos conjuntos:

- *Patrones de entrenamiento*. Con estos patrones se realizará el entrenamiento del clasificador que permita obtener un modelo. En el caso de redes neuronales, se suele tomar un porcentaje mayor o igual al 50% del total de patrones para realizar el proceso de entrenamiento.
- *Patrones de generalización o test*. Sirven para comprobar la bondad del modelo aprendido por el clasificador. Se utiliza el conjunto de patrones restantes para ser evaluados tras el entrenamiento. El objetivo de los patrones de test es valorar hasta qué punto el clasificador ha aprendido correctamente a generalizar, es decir, a obtener unos valores correctos en aquellas circunstancias para las que no ha sido explícitamente entrenado.

2.3.1. Medidas de evaluación de un clasificador

Para evaluar un clasificador se suele emplear una medida conocida como el porcentaje de patrones bien clasificados (CCR, del inglés *Correct Classification Rate*), es decir:

$$CCR = \frac{\sum_{n=1}^{N} I(C(\boldsymbol{x_n}) = y_n)}{N} \tag{2.13}$$

siendo *I(g)* una función que devuelve un 1 si *g* es cierto y un 0 en caso contrario, $C(\boldsymbol{x_n})$ la clase que el clasificador asigna al patrón $\boldsymbol{x_n}$ y *N* el número total de patrones. Un buen clasificador maximizaría el CCR o minimizaría el error correspondiente, es decir, $1-CCR$. Calcularemos, por un lado, el CCR de entrenamiento y, por otro, el de generalización. Como es lógico, interesa que el clasificador sea capaz de predecir lo mejor posible, por consiguiente lo ideal es obtener un CCR de generalización alto, esto es, cercano a 1.

Otra medida usada frecuentemente en problemas desbalanceados, tanto biclase como multiclase, es la sensibilidad (S), que expresa el valor de CCR de cada una de las clases. Su cálculo se realiza a partir de la matriz de confusión o tabla de contingencia. Para el caso de un problema de 3 clases, la matriz de confusión, incluyendo los valores marginales, vendría dada por:

		Clase predicha			
		C1	*C2*	*C3*	Total
	C1	*a*	*b*	*c*	$a+b+c= C1_{real}$
Clase real	*C2*	*d*	*e*	*f*	$d+e+f= C2_{real}$
	C3	*g*	*h*	*i*	$g+h+i= C3_{real}$
	Total	$a+d+g = C1_{predicha}$	$b+e+h = C2_{predicha}$	$c+f+i = C3_{predicha}$	*N*

Tabla 2.1. Matriz de confusión para un problema de 3 clases.

siendo *N* el número total de patrones, *C1*, *C2* y *C3* el valor correspondiente a las clases 1, 2 y 3, respectivamente. Los valores de las celdas se han representado por *a*, *b*, *c*, *d*, *e*, *f*, *g*, *h* e *i*. Para simplificar la explicación del cálculo, se reflejan las sumas totales por filas y por columnas.

A partir de la tabla de contingencia anterior, el valor del CCR viene dado por:

$$CCR = \frac{a+e+i}{N} \tag{2.14}$$

Los valores de la sensibilidad para cada clase se calculan mediante las siguientes expresiones:

$$S(C1) = \frac{a}{C1_{real}}; \qquad S(C2) = \frac{e}{C2_{real}}; \qquad S(C3) = \frac{i}{C3_{real}}; \tag{2.15}$$

A partir de las sensibilidades de cada una de las clases, es posible obtener la sensibilidad mínima (MS, del inglés *Minimum Sensitivity*), esto es, el valor de CCR para la clase que es peor clasificada [Fernández Caballero et al., 2010]. La MS viene dada por:

$$MS = \min(C_i); \quad i = 1, 2, ..., J \tag{2.16}$$

siendo *i* el índice de la clase y *J* el número de clases del problema.

En problemas desbalanceados, es muy importante evaluar correctamente las instancias de la clase minoritaria. Un buen clasificador debe lograr un CCR alto y clasificar correctamente tantas instancias como sea posible de la clase minoritaria. Por ejemplo, un clasificador puede reconocer todas las instancias de la clase mayoritaria y ninguna de la clase minoritaria, por consiguiente, el CCR será muy alto, mientras que MS será cero. Las medidas CCR y MS no son cooperativas en general. Al principio del proceso de aprendizaje, CCR y MS pueden ser cooperativas, pero después de un cierto tiempo son competitivas y una mejora en una tiende a implicar un decremento en la otra [Fernández Caballero et al., 2010].

Procedemos a ilustrar el cálculo de las sensibilidades con un ejemplo extraído de [Flach, 2012]:

		Clase predicha			
		C1	*C2*	*C3*	Total
	C1	*15*	*2*	*3*	*20*
Clase real	*C2*	*7*	*15*	*8*	*30*
	C3	*2*	*3*	*45*	*50*
	Total	*24*	*20*	*56*	*100*

Tabla 2.2. Ejemplo de matriz de confusión para un problema de 3 clases.

La sensibilidad de cada una de las clases de la tabla 2.2 viene dada por:

$$S(C1) = \frac{15}{20} = 0,75; \qquad S(C2) = \frac{15}{30} = 0,5; \qquad S(C3) = \frac{45}{50} = 0,9;$$

El valor de MS y de CCR correspondiente a la tabla 2.2 se muestran a continuación:

$$MS = \min(S(C1), S(C2), S(C3)) = 0,5; \quad \text{CCR} = \frac{15+15+45}{100} = 0,75$$

El valor de CCR del problema anterior es 0,75, mientras que las sensibilidades extremas serían 0,5 y 0,9, para las clases *C2* y *C3*, respectivamente. Los resultados del clasificador en cuestión indican que sólo el 50% de las instancias que realmente son de la clase *C2* son clasificadas correctamente. Por el contrario, en la clase *C3* la tasa de aciertos es del 90%. En este caso particular, la tasa de desbalanceo entre las clases *C1* y *C3* es 1:2,5.

2.3.2. Clasificadores basados en redes neuronales

En el apartado anterior, hemos explicado tres tipos de redes: supervisadas, no supervisadas y por refuerzo. Al mismo tiempo, dentro de las redes neuronales supervisadas, para el caso particular de un problema de clasificación, podemos diferenciar los siguientes clasificadores basados en redes neuronales [Lippman, 1989]:

a) *Clasificadores probabilísticos*. Asumen distribuciones de probabilidad a priori como las distribuciones gaussianas para las características de entrada. Los parámetros de dichas distribuciones son estimados por medio de entrenamiento supervisado, donde se asumen que todos los datos de entrenamiento están disponibles simultáneamente. Un ejemplo de este tipo es el modelo de mezcla gaussiana (del inglés *Gaussian Mixture Model*).

b) *Clasificadores de tipo hiperplano*. Crean regiones de decisión complejas usando nodos que forman fronteras de decisión de tipo hiperplano en el espacio delimitado por las entradas. Los nodos típicamente establecen una suma ponderada de las entradas, la cual se pasa, por ejemplo, a una no linealidad de tipo sigmoide. Otros tipos de no linealidades incluyen polinomios de grado superior de las entradas. En esta categoría encontramos el perceptrón multicapa (MLP), las redes de orden superior, etc.

c) *Clasificadores locales de tipo kernel*. Los clasificadores de tipo kernel o de campo receptivo construyen regiones de decisión complejas a partir de nodos con funciones de tipo kernel que crean campos receptivos solapados. Algunos ejemplos son: los clasificadores convencionales que estiman la función de densidad de probabilidad mediante la aproximación de Parzen, los clasificadores que emplean el método de las funciones locales, a menudo denominados clasificadores RBF.

d) *Clasificadores ejemplares*. También se denominan clasificadores basados en ejemplos. Se basan en identificar los ejemplos de entrenamiento o ejemplares que están más cerca de la entrada. En esta categoría se incluyen, entre otros, el clasificador de mapa de características (en

inglés *feature-map classifier*) [Huang y Lippman, 1988] y la cuantización del vector de aprendizaje (del inglés *Learning Vector Quantizer*).

2.4. Redes Neuronales de Unidades Producto y Redes Neuronales de Unidades Sigmoide

En este apartado describimos con más detalle los dos modelos de red que usaremos en esta Tesis: las RNUP y las RNUS.

2.4.1. Redes Neuronales de Unidades Producto

Las Redes Neuronales de Unidades Producto (RNUP) fueron introducidas por Richard Durbin y David E. Rumelhart en 1989 en el primer número, denominado *Spring*, del volumen de apertura de la revista *Neural Computation* [Durbin y Rumelhart, 1989]. Ellos sugirieron dos tipos de redes con unidades producto, consideradas en un modelo de tres capas, esto es, con una capa de entrada, una capa intermedia y una capa de salida.

En el primer tipo de red existe un grupo de unidades producto dedicadas (del inglés *group of dedicated product units*) en la capa intermedia, a los que algunos nodos de la capa de entrada están conectados y, por otra parte, todos los nodos de entrada están conectados a la capa de salida que está formada por unidades de tipo aditivo. El segundo tipo consiste en intercalar capas de unidades producto y unidades aditivas en las capas finales de la red; debemos destacar, por tanto, que, en este caso, no existen conexiones supra-capa y que la capa intermedia sólo contiene unidades producto. Además, hay una interpretación neurobiológica plausible de la configuración de red formada por unidades de tipo aditivo y de tipo producto. En esta Tesis seguimos el segundo modelo descrito.

Existen investigaciones posteriores de otros autores sobre RNUP, entre las que cabe destacar los siguientes trabajos: [Janson y Frenzel, 1993; Leerink et al. 1995a; Schmitt, 2001; Elliott, Topiwala y Browne, 2006; Martínez-Estudillo et al., 2008; Wang et al., 2008; Huang y Tong, 2009].

A continuación, se presentan las ventajas de las RNUP:

- Como consecuencia del Teorema de Stone-Weierstrass [Rudin, 1958], se ha demostrado que las RNUP son aproximadores universales [Martínez-Estudillo et al., 2006a] y, por consiguiente, pueden aproximar cualquier función con un nivel de precisión dado de la misma manera que las redes MLP [Hornik, Stinchcombe y White, 1989].
- La posibilidad de usar exponentes negativos permite expresar cocientes y ratios entre las variables [Durbin y Rumelhart, 1989].
- Está demostrado en [Durbin y Rumelhart, 1989] que la capacidad de información de una única unidad de tipo producto con patrones booleanos aleatorios es aproximadamente igual a $3N$, comparado con el valor de $2N$ [Cover, 1965], que corresponde a una unidad de tipo aditivo, siendo N el número de entradas de la unidad. En dicha demostración, la capacidad de información [Mitchison y Durbin, 1989] se mide como el valor numérico para el cual la probabilidad de almacenar todos los patrones perfectamente decrece a la mitad. Una mayor capacidad significa que las mismas funciones pueden ser implementadas por redes con menor número de unidades.
- Los exponentes del modelo son números reales. Esta característica tiene especial importancia, sobre todo, si se tiene en cuenta que son frecuentes las situaciones en el modelado de datos reales en las que la relación entre las variables responde a una estructura de tipo potencial, donde las potencias no están restringidas a los números

naturales o enteros. Como ejemplo, mencionamos el modelado de datos financieros [Saito et al., 2000].

- Permiten implementar funciones polinómicas de orden superior. Han demostrado buenos resultados en el modelado de datos en los que existen interacciones de diferentes órdenes entre las variables independientes del problema. De esta forma, cuando existen interacciones entre las variables que intervienen en un determinado fenómeno, las Redes Neuronales basadas en UPs permiten obtener modelos más simplificados que las redes MLP, en cuanto a necesitar un número inferior de funciones de base.

Como contrapartida, las redes de UP presentan un inconveniente importante, la superficie de error asociada es especialmente compleja con numerosos óptimos locales y regiones planas y, por tanto, existe una mayor probabilidad de que los algoritmos de búsqueda queden atrapados en alguno de los óptimos locales [Engelbrecht e Ismail, 1999]. La estructura potencial del modelo provoca que pequeños cambios en los exponentes tengan como consecuencia un cambio significativo en los valores de la función de activación y en la función de error [Schmitt, 2001]. Así, los algoritmos de entrenamiento de redes basados en el gradiente quedan con frecuencia, y de forma especial en este tipo de redes, atrapados en óptimos locales. Por ejemplo, está estudiado el hecho de que el clásico Algoritmo de Retropropagación (del inglés *Backpropagation Algorithm*) no funciona bien con RNUP [Janson y Frenzel, 1993].

Esta dificultad relacionada con el entrenamiento es una de las razones por las que, a pesar de las ventajas anteriormente señaladas, los modelos de Redes Neuronales basados en UPs han tenido poco desarrollo y han sido menos utilizados como herramientas en la resolución de problemas de clasificación que otros tipos de redes [Lippmann, 1989].

La arquitectura de las RNUP que consideramos en esta Tesis presenta las siguientes características:

- Una capa de entrada con un nodo para cada una de las variables de entrada.
- Una sola capa oculta con varios nodos de tipo unidad producto.
- Una única capa de salida con tantos nodos como clases menos una tenga el problema, cuyos patrones queremos clasificar; esto se debe a que se ha adoptado un enfoque probabilístico que considera la función de activación *softmax* en cada uno de dichos nodos. Esta propiedad se detalla en el apartado 2.4.3.
- Se añade sesgo a cada uno de dichos nodos de la capa de salida.
- Las únicas conexiones que hay son inter-capa, no existiendo conexiones de tipo intra-capa ni supra-capa.
- La función de transferencia de cada nodo de las capas oculta y de salida es la función lineal, con el fin de mantener la sencillez de los modelos obtenidos.

En la figura 2.7 se muestra la estructura de la red neuronal con unidades producto que utilizaremos en esta Tesis. Concretamente, la figura representa una topología n: m: 1 y corresponde a un problema de clasificación biclase, por lo que el número de nodos en la capa de salida es uno. En los casos en los que el problema es de tipo multiclase con *J* clases, tendremos *J-1* nodos en la capa de salida.

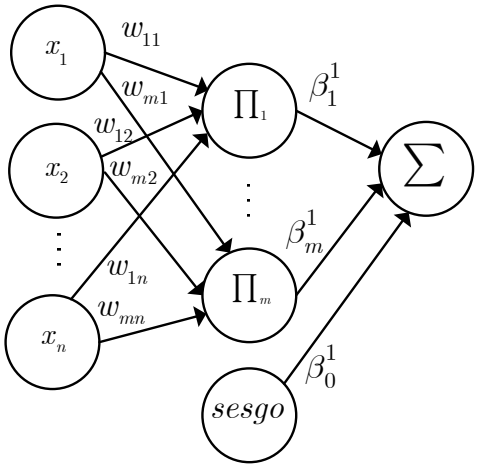


Figura 2.7. Red Neuronal con Unidades Producto y con topología n: m: 1 para un problema de clasificación binaria.

La función de computación o función de salida de los nodos de la capa oculta siguen el modelo multiplicativo de proyección y viene dada por:

$$\phi(\boldsymbol{x}, \boldsymbol{w_j}) = x_1^{w_{j1}} \cdot x_2^{w_{j2}} \cdot \ldots \cdot x_n^{w_{jn}} = \prod_{i=1}^{n} x_i^{w_{ji}}; \quad w_{ji} \in \mathbb{R} \tag{2.17}$$

donde x_1, x_2,…, x_n son variables de entrada de la red y w_{j1}, w_{j2},…, w_{jn} son los exponentes de las variables, los cuales constituyen los coeficientes del modelo.

El modelo funcional obtenido por cada uno de los nodos de la capa de salida de una RNUP es:

$$f(x_1, x_2, \ldots, x_n) = \beta_0^l + \sum_{j=1}^{m} \beta_j^l \left(\prod_{i=1}^{n} x_i^{w_{ji}} \right) \quad l = 1, 2, \ldots, J-1; \quad \beta_0^l, \beta_j^l, w_{ji} \in \mathbb{R} \tag{2.18}$$

que puede reescribirse de una forma más compacta teniendo en cuenta (2.17), con lo cual obtenemos:

$$f(x_1, x_2, \ldots, x_n) = \beta_0^l + \sum_{j=1}^{m} \beta_j^l \phi(\boldsymbol{x}, \boldsymbol{w_j}) \quad l = 1, 2, \ldots, J-1; \quad \beta_0^l, \beta_j^l \in \mathbb{R} \tag{2.19}$$

2.4.2. Redes Neuronales de Unidades Sigmoide

Las Redes Neuronales de Unidades Sigmoide (RNUS) son las más empleadas y han sido ampliamente exploradas [Chandra y Singh, 2004; Menon et al., 1996]. Una RNUS está compuesta por unidades aditivas de tipo sigmoide, que se ubican en la capa oculta.

Estas redes poseen una importante propiedad: la familia de funciones reales que representan es un subconjunto denso en el espacio de las funciones continuas definidas sobre un compacto con la convergencia uniforme. Esta propiedad de densidad significa que pueden aproximar cualquier función continua con suficiente precisión, con tal de que el número de nodos ocultos no esté acotado, y proporciona una sólida base de carácter teórico que ha sido esencial para el desarrollo del estudio y de las aplicaciones de estas redes [Cybenko, 1989; Hornik, Stinchcombe y White, 1989; Leshno et al., 1993].

La capacidad de aproximación de las RNUS ha hecho de ellas una herramienta alternativa para la resolución de problemas de modelado. Sin embargo, esta propiedad de carácter teórico presenta numerosas limitaciones en la práctica. En primer lugar, el número de nodos ocultos de la red necesarios para obtener una buena aproximación suele ser muy alto, lo que puede limitar la sencillez de los modelos obtenidos. Esto supone un grave inconveniente en muchas áreas de investigación, en las que la comprensión del modelo es tan importante como la precisión alcanzada en la aproximación.

Por otra parte, los diferentes resultados teóricos sobre la capacidad de aproximación de las redes neuronales son teoremas de existencia que no construyen la función que aproxima. La construcción de esta función

no es tarea sencilla, ya que implica determinar la estructura de la red y los pesos de las conexiones necesarios para conseguir un determinado nivel de aproximación. A este hecho hay que añadir que, con frecuencia, los algoritmos de entrenamiento que se utilizan para obtener los pesos de la red, o equivalentemente los coeficientes del modelo, tienen el riesgo de quedar atrapados en óptimos locales de una superficie de error que con frecuencia está plagada de ellos y que, además, es posible que tenga regiones planas.

Respecto a la arquitectura de las RNUS que empleamos en esta Tesis, es similar a la de las RNUP con las siguientes diferencias: a) la capa oculta se compone de varios nodos de tipo unidad aditiva sigmoide y b) cada uno de los nodos de la capa oculta presenta un valor umbral de activación o sesgo.

La estructura de la red neuronal con unidades sigmoide, que utilizamos en la Tesis, se ilustra en la figura 2.8, que representa una topología n: m: 1 y concierne a un problema de clasificación biclase, por lo que posee un único nodo en la capa de salida. Si se trata de un problema de tipo multiclase con *J* clases, existen *J-1* nodos en la capa de salida. El nodo denominado sesgo juega el mismo papel que nodo homónimo en el modelo RNUP, mientras que el nodo $sesgo_{\sigma}$ es el valor umbral de activación de los nodos ocultos.

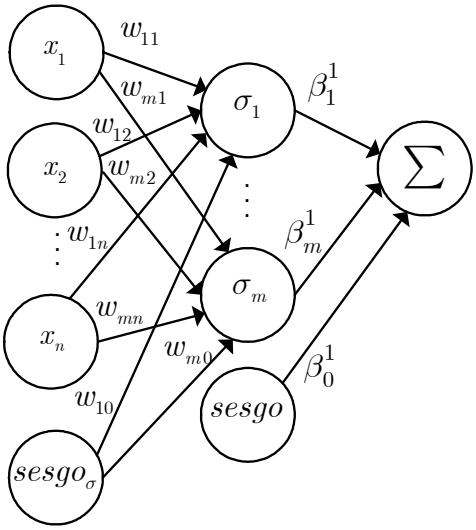


Figura 2.8. Red Neuronal con Unidades Sigmoide y con topología n: m: 1 para un problema de clasificación binaria.

La función de salida de los nodos de la capa oculta siguen un modelo aditivo de proyección y viene dada por:

$$\phi(\boldsymbol{x}, \boldsymbol{w_j}) = \frac{1}{1+e^{-(w_{j0}+w_{j1}x_1+w_{j2}x_2+\ldots+w_{jn}x_n)}} = \frac{1}{1+e^{-\left(w_{j0}+\sum_{i=1}^{n} w_{ji}x_i\right)}}; \quad w_{j0}, w_{ji} \in \mathbb{R} \tag{2.20}$$

donde x_1, x_2,…, x_n son variables de entrada de la red y w_{j0}, w_{j1}, w_{j2},…, w_{jn} son los los coeficientes del modelo, que corresponden al valor de umbral de activación del nodo o sesgo, w_{j0}, y a los factores de ponderación de las variables.

El modelo funcional obtenido por cada uno de los nodos de la capa de salida de una RNUS es:

$$f(x_1, x_2, \ldots, x_n) = \beta_0^l + \sum_{j=1}^{m} \beta_j^l \frac{1}{1+e^{-\left(w_{j0}+\sum_{i=1}^{n} w_{ji}x_i\right)}} \quad l = 1, 2, \ldots, J-1; \quad \beta_0^l, \beta_j^l, w_{j0}, w_{ji} \in \mathbb{R} \tag{2.21}$$

que puede reescribirse de una forma más compacta teniendo en cuenta (2.20), con lo cual resulta:

$$f(x_1, x_2, \ldots, x_n) = \beta_0^l + \sum_{j=1}^{m} \beta_j^l \phi(\boldsymbol{x}, \boldsymbol{w_j}) \quad l = 1, 2, \ldots, J-1; \quad \beta_0^l, \beta_j^l \in \mathbb{R} \tag{2.22}$$

2.4.3. Modelo funcional conjunto, evaluación, función de activación softmax y función de error

Si observamos, el modelo funcional obtenido en el caso de las RNUP y RNUS es similar. Por tanto, las diferencias radican en la computación de los nodos de la capa oculta y la ausencia o presencia de sesgo en los nodos de la capa oculta.

La función de transferencia de la capa de salida es lineal. Podemos representar con $\boldsymbol{\theta} = (\theta_1,...,\theta_{J-1})$ el conjunto de coeficientes asociado a la Red Neuronal (RN). Dado que tenemos *J-1* nodos en la capa de salida, tendremos *J-1* funciones de salida, que denotaremos con:

$$f_l(\boldsymbol{x},\boldsymbol{\theta_l}) = \beta_0^l + \sum_{j=1}^{m} \beta_j^l \phi_j(\boldsymbol{x},\boldsymbol{w_j}) \quad l = 1,2,...,J-1; \quad \beta_0^l, \beta_j^l \in \mathbb{R} \tag{2.23}$$

donde $\boldsymbol{\theta_l} = \left(\boldsymbol{\beta^l}, \boldsymbol{W}\right)$ es el conjunto de coeficientes para el nodo *l* de salida, formado por dos componentes, donde $\boldsymbol{\beta^l} = \left(\beta_0^l, \beta_1^l, ..., \beta_m^l\right)$ es el valor de las conexiones entre la capa oculta y la capa de salida para dicho nodo, y $\boldsymbol{W} = \left(\boldsymbol{w_1}, \boldsymbol{w_2}, ..., \boldsymbol{w_m}\right)$ es una matriz de coeficientes asociados a los nodos de la capa oculta, donde cada $\boldsymbol{w_j} = \left(w_{j0}, w_{j1}, w_{j2}, ..., w_{jn}\right)$ es un vector con el valor de las conexiones del nodo *j* de la capa oculta y $\phi_j(\boldsymbol{x},\boldsymbol{w_j})$ es la salida del nodo *j* de la capa oculta. En el caso de las RNUP, el valor w_{j0} no existe.

Adoptaremos la técnica común, consistente en utilizar un vector "1-de-J" para representar la clase del patrón. Estos vectores tienen la forma $\boldsymbol{y} = (y^{(1)}, y^{(2)}, ..., y^{(J)})$, siendo $y^{(J)}$ igual a 1 si el ejemplo pertenece a la clase *J* e igual a cero en caso contrario. Basándonos en este conjunto de entrenamiento, el objetivo es encontrar una estimación de los coeficientes de la RN, $\hat{\boldsymbol{\theta}}$, que clasifique correctamente a los patrones del conjunto de generalización. Esto equivale a obtener una función de decisión

$R:\Omega \to \{1,2,...,J\}$ para clasificar las instancias. En otras palabras, R proporciona una división de las instancias del conjunto de entrenamiento del tipo $D_1, D_2, ..., D_J$, donde D_i corresponde a la clase i. Una clasificación incorrecta se produce si R asigna a un patrón la clase i cuando en realidad pertenece a la clase j, verificándose, $i \neq j$.

El rendimiento de la red neuronal será evaluado mediante el valor de CCR, que depende de los coeficientes del modelo de RN y viene dado por:

$$CCR(\hat{\boldsymbol{\theta}}) = \frac{\sum_{n=1}^{N} I(R(\boldsymbol{x_n}) = y_n)}{N} \tag{2.24}$$

siendo *I(g)* una función que devuelve un 1 si *g* es cierto y un 0 en caso contrario, $R(\boldsymbol{x_n})$ la clase en la que el clasificador clasifica al patrón $\boldsymbol{x_n}$ y *N* el número total de patrones. Una buena red neuronal maximizaría el ratio $CCR(\hat{\boldsymbol{\theta}})$ o minimizaría el error correspondiente, es decir, $1-CCR(\hat{\boldsymbol{\theta}})$. Calculamos, por un lado, el CCR de entrenamiento (CCR_{ent}) y, por otro, el de generalización (CCR_{gen}). Como es lógico, interesa que el clasificador sea capaz de predecir lo mejor posible, con lo cual lo ideal es obtener un CCR_{gen} alto. Habitualmente, en las publicaciones, por motivos de espacio, sólo aparece un valor de CCR y es, precisamente, el de generalización; por tanto, en lo sucesivo, en caso de indicar CCR sin ningún tipo de subíndice, nos estamos refiriendo al valor de generalización que, en último término, es el más importante.

Por otra parte, a la hora de interpretar los valores de salida de los nodos de la capa de salida se sigue un enfoque de tipo probabilístico que considera la función de activación *softmax* [Bridle, 1990], que viene dada por:

$$g_l(\boldsymbol{x},\hat{\boldsymbol{\theta}}_l) = \frac{e^{f_l(\boldsymbol{x},\hat{\boldsymbol{\theta}}_l)}}{\sum_{l=1}^{J} e^{f_l(\boldsymbol{x},\hat{\boldsymbol{\theta}}_l)}} \tag{2.25}$$

siendo J el número de clases del problema, $f_l(\boldsymbol{x},\hat{\boldsymbol{\theta}}_l)$ la función de salida del nodo l para el patrón $\boldsymbol{x}$ y $g_l(\boldsymbol{x},\hat{\boldsymbol{\theta}}_l)$ la probabilidad de que el patrón $\boldsymbol{x}$ pertenezca a la clase l. La transformación *softmax* produce estimaciones positivas en todas las salidas, de tal forma que la suma de todas ellas es uno, lo que hace que las salidas puedan ser interpretadas como la probabilidad de pertenecer a la clase correspondiente. Teniendo en cuenta esta consideración, se puede comprobar que la clase predicha por la RN es la correspondiente a la neurona de la capa de salida, cuyo valor de salida es mayor. Expresado formalmente, esto quiere decir que:

$$R(\boldsymbol{x}) = \hat{l} \quad \text{donde } \hat{l} = \arg\ \max_l\ g_l(\boldsymbol{x},\hat{\boldsymbol{\theta}}_l) = \arg\ \max_l\ f_l(\boldsymbol{x},\hat{\boldsymbol{\theta}}_l) \quad l = 1,2,...,J \tag{2.26}$$

Al tratarse de probabilidades de pertenencia a una clase, está claro que no es necesario calcularlas todas, ya que la última probabilidad $g_J(\boldsymbol{x},\hat{\boldsymbol{\theta}}_J)$ se puede calcular en función del resto como $1-\sum_{l=1}^{J-1} g_l(\boldsymbol{x},\hat{\boldsymbol{\theta}}_l)$. De esta forma, podemos simplificar el modelo considerando la salida del último nodo de la capa de salida constante e igual a 0, es decir, $f_J(\boldsymbol{x},\hat{\boldsymbol{\theta}}_J) = 0$. Este nodo no será entrenado para reducir así el número de parámetros a estimar y la carga computacional del entrenamiento.

Considerando las salidas como probabilidades, otra medida del error cometido por el modelo de RN es la función de error *cross-entropy* (entropía cruzada) [Bishop, 1995], que viene dada por la siguiente expresión:

$$l(\hat{\boldsymbol{\theta}}) = -\frac{1}{N} \cdot \sum_{n=1}^{N} \sum_{l=1}^{J} (y_n^l \ln(g_l(\boldsymbol{x}_n,\hat{\boldsymbol{\theta}}_l))) \tag{2.27}$$

y en función de f_l,

$$l(\hat{\boldsymbol{\theta}}) = \frac{1}{N} \cdot \sum_{n=1}^{n} \left(-\sum_{l=1}^{J} y_n^l \, f_l(\boldsymbol{x_n}, \hat{\boldsymbol{\theta}}_l) + \ln\left(\sum_{l=1}^{J} e^{f_l(\boldsymbol{x_n}, \hat{\boldsymbol{\theta}}_l)} \right) \right) \tag{2.28}$$

siendo $D = \left\{ (\boldsymbol{x_n}, \boldsymbol{y_n}) \right\} \quad n = 1, 2, ..., N$ el conjunto de datos, y_n^l el valor esperado para la clase l del patrón n (y_n^l será igual a 1 si el patrón $\boldsymbol{x_n}$ pertenece a la clase l y a 0 en caso contrario), $f_l(\boldsymbol{x_n}, \hat{\boldsymbol{\theta}}_l)$ el valor de salida del nodo l con el patrón $\boldsymbol{x_n}$ y N el número de patrones. La ventaja del uso de la función de error $l(\hat{\boldsymbol{\theta}})$, en lugar de $1 - CCR(\hat{\boldsymbol{\theta}})$, es que es una función continua, lo que permite que el entrenamiento pueda converger siempre hacia soluciones más óptimas sin quedar estancado.

2.5. MÉTODOS DE ENTRENAMIENTO

2.5.1. Métodos basados en optimización local y global

La búsqueda de RNAs óptimas [Rocha, Cortez y Neves, 2007] es un desafío. Una red pequeña tendrá capacidades de aprendizaje limitadas, mientras que una red demasiado grande puede producir el efecto de sobreentrenamiento, el cual no es deseable dado que tiende a disminuir el rendimiento sobre el conjunto de generalización. Típicamente, el diseño de una buena Red Neuronal se suele hacer mediante procedimientos de prueba y error (ejemplo: explorando un número diferente de nodos en la capa oculta), siguiendo una estrategia de búsqueda ciega, que sólo emplea un conjunto pequeño de posibles configuraciones.

Existen dos tipos de planteamientos para el entrenamiento de una red neuronal multi-capa con conexiones hacia adelante: algoritmos de optimización local y métodos de optimización global.

Dentro del primer grupo encontramos como principal exponente los métodos del gradiente descendente. Estos algoritmos utilizan una

arquitectura fija y se calcula el valor de los coeficientes $\hat{\theta}$ del modelo de RN buscando minimizar una función de coste que es evaluada con los datos de entrenamiento. Su nombre hace alusión al hecho de que en cada paso se realiza una modificación de los pesos sinápticos en la misma dirección del gradiente de la función de error. Sin embargo, al estar basados en derivadas, las funciones de transferencia deben ser cuidadosamente escogidas, siendo necesario que sean continuas y derivables.

El ejemplo más conocido es el popular algoritmo de Retropropagación, también denominado BP (del inglés *Back-Propagation*). Fue propuesto de manera independiente por varios investigadores: a) Arthur E. Bryson y Ho Yu-Chi [Bryson y Yu-Chi, 1969], b) Paul J. Werbos [Werbos, 1974], c) David B. Parker [Parker, 1985] y d) David E. Rumerlhart, Geoffrey E. Hinton y Ronald J. Williams [Rumelhart, Hinton y Williams, 1986]. Un estudio detallado aparece en los trabajos [Hecht-Nielsen, 1989] y [Le Cun, 1988]. Se basa en la inicialización de los pesos con valores aleatorios y en el cálculo iterativo de la estimación de los pesos, realizado en dos pasadas que recorren la red en direcciones opuestas [Werbos, 1974]. El principal inconveniente de este algoritmo es que la minimización de la función de coste es un problema no lineal y, por lo tanto, es habitual que quede atrapado en mínimos locales.

Los parámetros más importantes del algoritmo básico hacen referencia, entre otras cuestiones, al criterio de finalización del algoritmo (número de épocas) [Kramer y Sangiovanni-Vincentelli, 1989], a la tasa de aprendizaje, η, o al tipo de minimización (minimización *batch* del error producido por todos los patrones en cada iteración o minimización *on-line* de cada uno de los patrones por separado). El principal inconveniente del algoritmo radica en la dificultad común a los métodos basados en

gradiente: la convergencia de la función de coste es lenta. Para evitar este problema, se propone el uso del término denominado momento, α, que suaviza el comportamiento oscilatorio del algoritmo [Silva y Almeida, 1990].

En términos generales, el algoritmo BP obtiene buenos resultados con redes de tipo aditivo, no obstante, presenta serias dificultades con redes de tipo multiplicativo. Ésa es la principal motivación que nos lleva a emplear en la presente Tesis otros tipos de algoritmos, como son los métodos de búsqueda global.

Los métodos de búsqueda global tratan de escapar de óptimos locales, explorando el espacio de búsqueda con mayor eficiencia. Generalmente, se añade algún componente aleatorio a la búsqueda, de forma que, si se encuentra un óptimo local, se salte a otro punto del espacio de búsqueda, donde pueda haber otro óptimo, posiblemente global [Torn y Zilinskas, 1987]. Entre los métodos de búsqueda global, hallamos muchos algoritmos metaheurísticos, que se definen como algoritmos aproximados de optimización y búsqueda de propósito general. En [Alba y Martí, 2006] se hace una amplia revisión de un buen número de metaheurísticas que se han empleado en el entrenamiento de redes neuronales. Dentro de los algoritmos metaheurísticos distinguimos los que se basan en trayectorias y los que se basan en poblaciones.

En el entrenamiento de redes neuronales se han empleado diferentes metaheurísticas basadas en trayectorias. Entre ellas, podemos destacar el método de enfriamiento simulado (del inglés *simulated annealing*) [Kirkpatrick, Gelatt y Vecchi, 1983; Černý, 1985], también denominado recocido simulado, que se define como un algoritmo de búsqueda basado en entornos con un criterio probabilístico de aceptación

de soluciones basado en la Termodinámica. En el proceso de búsqueda, a partir de una solución no siempre se pasa a una solución vecina mejor, sino que se permite transitar a soluciones peores con una cierta probabilidad, dependiendo de la *temperatura*. Se basa en el criterio de Metrópolis [Metropolis et al., 1953], quien modeló el proceso de enfriamiento mediante la simulación de los cambios energéticos en un sistema de partículas conforme decrece la temperatura, hasta que converge a un estado estable (congelado). Las leyes de la Termodinámica dicen que a una temperatura t la probabilidad de un cambio de energía, ΔE, se puede aproximar mediante la expresión:

$$P(\Delta E) = e^{\left(-\frac{\Delta E}{kT}\right)} \tag{2.29}$$

siendo k la constante física de Boltzmann y T la temperatura de la partícula.

En la siguiente figura se muestra el algoritmo básico de enfriamiento simulado:

```
Programa: Enfriamiento Simulado
1: iniciar el procedimiento en un punto del espacio de búsqueda
2: mientras no se cumpla la condición de terminación hacer
3:         mejorar la solución actual, dependiendo de la temperatura
4:         actualizar la temperatura
5: fin mientras
```

Figura 2.9. Pseudocódigo del Algoritmo de Enfriamiento Simulado.

Entre los trabajos en los que se ha empleado el método de enfriamiento simulado [Aarts y van Laarhoven, 1989; Otten y van Ginneken, 1989] para entrenar una RNA, podemos destacar los siguientes: el artículo de Kenneth D. Boese y A. B. Kahng [Boese y Kahng, 1993], el que se aplica a un modelo de red de tipo perceptrón, y la investigación llevada a cabo por Randall S. Sexton y otros [Sexton, Dorsey y Johnson,

1999], en la que se pone de manifiesto, desde el punto de vista experimental, que los resultados obtenidos son mejores que los que se alcanzan con el método BP.

Por otra parte, también se ha empleado la Búsqueda Tabú (BT), introducida por Fred Glover [Glover, 1986] e, independientemente, por Pierre Hansen [Hansen, 1986] con otra denominación, aunque con similares ideas. La BT es una técnica de búsqueda global por entornos, caracterizada por dos aspectos principales: a) permite escapar de óptimos locales mediante movimientos de empeoramiento y, para evitar recorridos cíclicos, emplea un esquema denominado generación de entornos tabú restringidos, y b) emplea mecanismos de reinicialización para mejorar la capacidad del algoritmo para la exploración-explotación del espacio de búsqueda: intensificación y diversificación. La BT ha sido aplicada al entrenamiento de RNA en varios trabajos [De Werra y Hertz, 1989; Sexton et al., 1998] y una extensión, denominada BT reactiva, en [Battiti y Tecchiolli, 1995].

Respecto a las metaheurísticas basadas en poblaciones, resaltamos la búsqueda dispersa y todas las técnicas de computación bioinspirada, como son los algoritmos basados en Computación Evolutiva y/o en Inteligencia de Enjambres. De los dos tipos de estrategias mencionadas, describimos en este apartado el primer tipo y el segundo, en el siguiente apartado, debido a su mayor difusión y mayor relación con la temática de la presente Tesis.

La Búsqueda Dispersa (del inglés *scatter search*), BD, fue propuesta por Fred Glover en 1977 [Glover, 1977]. La continuación en la investigación de este método se plasmó en el trabajo posterior, casi dos décadas después, del propio Glover [Glover, 1995]. A raíz de dicho trabajo,

otros autores, como Manuel Laguna y Rafael Martí, empezaron a investigar conjuntamente con Fred Glover en la temática de la BD [Glover, Laguna y Martí, 2000; Laguna y Martí, 2003]. Como una estrategia complementaria a la BD, se ha propuesto la técnica de reencadenamiento de trayectorias (del inglés *Path Relinking*).

La BD trabaja con un conjunto de soluciones, denominado conjunto de referencia, que son combinadas para crear nuevas soluciones. La idea de combinar reglas de decisión y restricciones tiene sus orígenes en la década de los 60 del siglo XX [Glover, 1963], en el contexto de métodos de planificación en problemas de asignación de tareas. El enfoque de la BD es explorar de una manera sistemática respecto al conjunto de referencia, que típicamente se compone de buenas soluciones, donde el criterio de bondad no se restringe al valor de la función objetivo sino a otros aspectos tales como diversidad, complejidad de la solución. La BD, a pesar de ser menos común que los algoritmos genéticos y la búsqueda tabú, se ha aplicado con gran éxito en la resolución de problemas de optimización difíciles. Entre los trabajos existentes que han aplicado BD en el entrenamiento de RNAs, cabe mencionar los siguientes: [Kelly, Rangaswamy y Xu, 1996] y [Larson y Newman, 2011].

2.5.2. Redes Neuronales Evolutivas

Las Redes Neuronales Artificiales Evolutivas (RNAEs) han sido un área de investigación clave en la última década del siglo XX y la primera del siglo XXI [Schaffer, Whitley y Eshelman, 1992], proporcionando una plataforma mejorada para optimizar simultáneamente el rendimiento y la arquitectura de la red (número de nodos en la capa oculta y número de conexiones). Geoffrey F. Miller, Peter M. Todd y Shailesh U. Hegde [Miller, Todd y Hegde, 1989] propusieron que la Computación Evolutiva (CE) era

una buena candidata para buscar en el espacio de arquitecturas, debido a que la función de aptitud asociada con dicho espacio presenta ruido, es compleja, no diferenciable, multimodal y deceptiva. Desde entonces, muchos métodos de Programación Evolutiva (PE) han sido desarrollados para evolucionar redes neuronales, tales como [Yao y Liu, 1997; García-Pedrajas, Hervás-Martínez y Muñoz-Pérez, 2002; Ohkura et al., 2007]. En menor número de trabajos se emplean AGs [Azzini y Tettamanzi, 2008].

Existen tres formas de abordar el entrenamiento de una RNA con un AE:

1) *Evolucionar los pesos de la RNA*. En este caso, se fija la arquitectura de la red y se desarrolla una búsqueda global en el espacio de pesos. Los pesos son tradicionalmente representados con codificación real. En comparación con los algoritmos de retropropagación (BP, *Back-Propagation*) clásicos, pueden ser más lentos, pero no requieren el cálculo del gradiente.

2) *Evolucionar la arquitectura de la RNA*. Se parte, en este caso, de una inicialización aleatoria de los pesos y, tras la ejecución del AE, se suelen realizar varias iteraciones de un algoritmo BP clásico. La representación más natural para esta tarea es la codificación binaria. De cualquier forma, la optimización de la arquitectura es un proceso difícil, dando únicamente resultados aceptables en problemas relativamente pequeños.

3) *Evolucionar simultáneamente tanto los pesos como la arquitectura de la RNA*. Esta estrategia pretende reunir las ventajas de las dos anteriores. El principal inconveniente radica en la dificultad asociada al

entrenamiento de los modelos de red. Hoy en día, es una de las técnicas más interesantes de RNAEs.

Se han realizado múltiples esfuerzos para llevar a cabo métodos de aprendizaje para Unidades Producto (UP). David J. Janson y James F. Frenzel [Janson y Frenzel, 1993] desarrollaron un algoritmo genético para evolucionar los pesos de una red, basada en UP con una arquitectura predefinida. El mayor problema de este tipo de algoritmo es cómo obtener de antemano la arquitectura óptima. Desafortunadamente, hasta el momento, el problema de diseñar una arquitectura de red neuronal casi óptima para una determinada aplicación permanece sin resolver.

Adiel Ismail y A. P. Engelbrecht aplicaron cuatro métodos diferentes de optimización para entrenar redes de tipo unidad producto [Ismail y Engelbrecht, 1999; Ismail y Engelbrecht, 2000; Ismail y Engelbrecht, 2002]: algoritmo genético estándar, método de optimización mediante cúmulo de partículas (en inglés *Particle Swarm Optimization*), basada en el problema físico del movimiento de una partícula de masa unitaria en un campo de fuerzas conservativo de dimensión *n* (LeapFrog) y un algoritmo de poda. Ajith Abraham [Abraham, 2004] propuso MLEANN (*Meta-Learning Evolutionary Artificial Neural Networks*, Metaaprendizaje de Redes Neuronales Artificiales Evolutivas), un entorno de trabajo computacional adaptivo, basado en CE para el diseño automático de RNAs óptimas. Sin embargo, dicho método presenta la limitación de que el número de nodos oscila entre 5 y 16.

Esto nos da idea de que diseñar RNAs no es una tarea sencilla como a primera vista podría parecer. En [Hervás-Martínez, Martínez-Estudillo y Gutiérrez, 2006] se presenta un algoritmo de PE para el entrenamiento de RNUP para problemas de clasificación que permite diseñar la estructura

(número de nodos y número de conexiones) y, al mismo tiempo, optimizar los coeficientes de los modelos de RNUP. Este artículo, además, revisa los trabajos previos sobre esta temática y constituye una actualización del área de RNAEs usando PE para llevar a cabo una evolución simultánea, por lo que lo hemos tomado como algoritmo base.

La diversidad es un concepto muy importante en CE. La evolución, por definición, requiere diversidad, la cual se refiere a la variación (genética) de los miembros de la población [Heitkoetter y Beasley, 2001; Amor y Rettinger, 2005]. Es una idea valiosa debido a que hace posible explorar nuevas regiones del espacio de búsqueda, que permiten evitar que ésta quede atrapada en óptimos locales [Chop y Calvert, 2005]. En realidad, una alta diversidad implica que la población cubre una parte mayor del espacio [Ursem, 2002].

Una visión común del proceso evolutivo es que la diversidad mejora el rendimiento de una población proporcionando más oportunidades para evolucionar. Una población homogénea no ofrece ninguna ventaja que permita progresar, debido a que la población entera está centrada en una porción específica del espacio de búsqueda. Por el contrario, una población diversa explorará simultáneamente una zona grande del espacio de búsqueda, proporcionando la oportunidad de situarse en soluciones diferentes, potencialmente mejores [Curran y O'Riordan, 2006].

Mantener la diversidad en las primeras generaciones es una tarea crucial en AE porque ésta puede no sólo afectar a la velocidad de convergencia, sino también a la calidad de la solución final. Una población diversa es preferible al comienzo del algoritmo y una más condensada al final de la búsqueda [Maaranen, Miettinen y Mäkelä, 2004]. Otra cuestión

asociada con la diversidad a través del proceso evolutivo es si se crea una o varias poblaciones. El tema de generar varias poblaciones ha sido tratado previamente. En [Wang, Zheng y Tang, 2002] se presenta la idea de que varias poblaciones son sometidas a operadores de mutación diferentes y, a continuación, todos los individuos son reunidos y repartidos en varias poblaciones, cada una de las cuales es sometida a otro operador de mutación. Por lo general, las poblaciones exploran diferentes áreas del espacio de búsqueda a través de distintas semillas; el número de poblaciones puede variar y no existe un valor común aceptado.

Capítulo 3

Selección de Atributos

3.1. MOTIVACIÓN

El problema de la Selección de Atributos (SA), también denominado reducción de atributos o de dimensionalidad y selección de características o de variables, ha sido ampliamente explorado tanto desde la perspectiva de Reconocimiento de Patrones como desde la de Aprendizaje Automático. Sin embargo, las publicaciones en el contexto de Reconocimiento de Patrones aparecen en los comienzos de la década de los 60 del siglo XX [Lewis, 1962; Fu y Min, 1968; Fu, Min y Li, 1970; Kittler, 1975], mientras que en el caso de Aprendizaje Automático aparecen en la década de los 90 de dicho siglo [Vafaie y De Jong, 1992; Blum y Langley, 1997].

En los años 60, el propósito de la SA era hacer, por un lado, que el número de entradas del problema fuera tal que pudiera ser procesado por las máquinas de la época, debido a la limitación del sistema de reconocimiento respecto a la ocupación en memoria de los datos del problema y, por otro, procesar los datos en el sistema en el tiempo disponible, debido a que las pocas máquinas existentes eran compartidas por muchos usuarios. A partir de los 90, con el aumento de la información disponible y la reducción del coste de obtención, la idea es ser capaces de

extraer sólo las características que son útiles y relevantes para cada problema; en esta década empieza a emerger con fuerza el área de Minería de Datos y se pone especial énfasis en el preprocesamiento de datos. Dorian Pyle afirma: "el objetivo fundamental de la preparación de datos es manipular y transformar los datos en bruto para que el contenido implícito de la información en el conjunto de datos pueda ser expuesto o más fácilmente accesible" [Pyle, 1999]. La selección de atributos forma parte de tal preparación de datos.

Para muchos problemas prácticos, el número posible de entradas de una RNA puede ser enorme o, incluso, podría haber redundancia entre diferentes entradas. Un número grande de entradas incrementa el tamaño de la RNA y, por consiguiente, requiere más datos de entrenamiento y tiempos de procesamiento mayores para alcanzar una capacidad de generalización aceptable. El preprocesamiento es, pues, necesario para reducir el número de entradas de la RNA. El problema de encontrar un conjunto de atributos casi óptimo se puede formular en términos de un problema de búsqueda. Dado un número grande de entradas potenciales, deseamos encontrar un subconjunto próximo al óptimo que tenga el menor número de entradas y que el rendimiento de la RNA usando tal subconjunto no sea peor que el obtenido usando el conjunto de entrada completo [Yao, 1999].

La motivación para aplicar técnicas de SA ha pasado de ser un asunto opcional a convertirse en un prerrequisito para la obtención del modelo. La principal razón es la naturaleza multidimensional de muchas tareas de modelado en diferentes ámbitos de aplicación. Podría pensarse que tener más atributos nos debería otorgar un mayor poder de discriminación de los datos en diferentes clases o grupos. No obstante, tal hecho puede provocar muchos problemas: a) aumento de la complejidad y

del coste computacional, b) muchos atributos redundantes o irrelevantes y c) degradación en la estimación del error de clasificación.

La SA y la eliminación o reducción de información redundante no relacionada con la tarea de clasificación a realizar no solo reducirá la complejidad del problema y mejorará la eficiencia del procesamiento sino que también simplificará significativamente el diseño del clasificador, con lo cual la sencillez del modelo será mayor. Todos estos motivos hacen que la SA sea una técnica esencial y frecuentemente usada en Aprendizaje Automático.

3.2. DEFINICIÓN Y TAXONOMÍA

La selección de atributos es el problema consistente en escoger un subconjunto de atributos pequeño que, idealmente, es necesario y suficiente para describir el concepto deseado [Liu y Motoda, 2008]. En términos matemáticos, tal como se propuso en uno de los primeros trabajos [Fu, Min y Li, 1970], se trata de seleccionar un subconjunto de atributos de menor tamaño, formado por p atributos de un conjunto dado de N atributos ($p \leq N$).

Un método de SA genera diferentes candidatos del espacio de atributos y valorarlos basándose en un criterio de evaluación para encontrar el mejor subconjunto de atributos [Dash y Liu, 1997]. Hay varias formas en las que los algoritmos de SA pueden ser agrupados: 1) según la medida o el criterio de evaluación de atributos, técnica de filtro o de *wrapper* (envoltorio), y 2) dependiendo de la manera de evaluar los atributos y, por tanto, la generación de subconjuntos (evaluación individual o de subconjuntos) [Blum y Langley, 1997].

Los modelos filtro [Liu y Setiono, 1996] se basan en características generales o propiedades intrínsecas de los datos, tales como distancia, consistencia y correlación [Dash y Liu, 1997; Hall, 2000; Dash y Liu, 2003], para evaluar y seleccionar subconjuntos de atributos sin considerar ningún algoritmo de aprendizaje. Por el contrario, los modelos *wrapper* requieren un algoritmo predeterminado de aprendizaje y emplea su rendimiento como criterio de evaluación y determinar qué atributos se seleccionan [Kohavi y John, 1997].

En selección de subconjuntos de atributos, es un hecho que dos tipos de atributos se perciben generalmente como innecesarios: atributos que son irrelevantes para el concepto deseado y atributos que son redundantes dados otros atributos. Los métodos de tipo *wrapper* seleccionan con frecuencia características que tienen una mayor precisión, sin embargo, son criticados debido al alto coste computacional y a que son específicos para el clasificador para el que han sido diseñados, por lo que presentan una baja generalidad. Para aprovechar las ventajas de los dos modelos anteriores, fue propuesto un modelo híbrido para manejar grandes conjuntos de datos [Xing, Jordan y Karp, 2001]. Además, algunos métodos, conocidos como embebidos o empotrados, emplean información interna del modelo de clasificación para realizar la SA [Guyon y Elisseeff, 2003; Saeys, Abeel y de Peer, 2008].

Según el procedimiento de generación, los métodos de SA se dividen en: ranking individual del atributo (FR, del inglés *Feature Ranking*) y selección de subconjunto de atributos (FSS, del inglés *Feature Subset Selection*) [Blum y Langley, 1997; Guyon y Elisseeff, 2003]. FR mide la relevancia de cada atributo con la clase y, posteriormente, ordena los atributos por puntuaciones decrecientes y selecciona los que están en las

primeras posiciones. Estos métodos son ampliamente usados debido a su simplicidad, escalabilidad y buen éxito empírico [Golub et al., 1999; Guyon y Elisseeff, 2003].

Mark A. Hall y Geoffrey Holmes [Hall y Holmes, 2003] propusieron un método de búsqueda de tipo ranking para comparar algoritmos de SA, pero no es eficiente en dominios de gran dimensionalidad. Por eso, FR es criticado debido a que captura sólo la relevancia de los atributos respecto al concepto objetivo, mientras que la redundancia y las interacciones básicas no son descubiertas. Adicionalmente, el número de atributos a retener es difícil de determinar y, como consecuencia, se requiere un umbral.

Por el contrario, FSS intenta encontrar un subconjunto de atributos que tengan buen rendimiento. Este método integra la métrica para cuantificar la relevancia atributo-clase y las interacciones atributo-atributo. En el trabajo [Liu y Yu, 2005], un gran número de métodos de selección son categorizados, en los cuales algoritmos diferentes tratan el problema de manera muy particular. Encontramos diferentes estrategias de búsqueda (exhaustiva, heurística y aleatoria) y la combinación con muchos tipos de medidas para obtener diferentes algoritmos. La complejidad temporal es exponencial en términos de la dimensionalidad de los datos para una búsqueda exhaustiva y cuadrática para una búsqueda heurística. La complejidad puede ser lineal respecto al número de iteraciones en una búsqueda aleatoria, aunque los experimentos muestran que para encontrar el mejor subconjunto de atributos, el número de iteraciones requerido es habitualmente al menos cuadrático respecto al número de atributos [Dash, Liu y Motoda, 2000].

En esta misma taxonomía, para manejar conjuntos de datos grandes, se ha propuesto un modelo híbrido para combinar los beneficios las técnicas FR y FSS. Este modelo desacopla el análisis de la relevancia y de la redundancia y las pruebas indican que es más efectivo que los métodos de ranking y más eficiente que los métodos de evaluación de subconjuntos en muchos conjuntos de datos de alta dimensionalidad. En este contexto, en el trabajo [Ruiz et al., 2006] se propuso un algoritmo de búsqueda híbrido. En 2004, en [Yu y Liu, 2004] fue presentado el método FCBF (del inglés *Fast Correlation-Based Filter algorithm*, algoritmo rápido de tipo filtro basado en correlación) que emplea una medida de correlación para obtener atributos relevantes y eliminar redundancia. Chris Ding y Hanchuan Peng [Ding y Peng, 2003] usaron la información mutua para la selección de genes, encontrando la relevancia máxima con mínima redundancia, mediante la resolución de una optimización simple biobjetivo.

3.3. MECANISMO TÍPICO

Manoranjan Dash y Huan Liu [Dash y Liu, 1997] identificaron cuatro pasos básicos en un método de SA típico:

1. Procedimiento de generación: permite generar el siguiente subconjunto candidato.
2. Función de evaluación: sirve para evaluar el subconjunto en consideración.
3. Criterio de parada: se emplea para decidir cuando detener el proceso.
4. Procedimiento de validación: permite verificar si el subconjunto es válido.

El procedimiento de generación es un procedimiento de búsqueda. Básicamente, genera subconjuntos de atributos para ser evaluados. Tal procedimiento puede empezar: a) sin atributos, b) con todos los atributos o c) con un conjunto de atributos aleatorio. En los dos primeros casos, las características son iterativamente añadidas o eliminadas, mientras que en el último caso los atributos o bien son iterativamente añadidos o eliminados, o bien se generan aleatoriamente a partir de entonces. Una función de evaluación mide la bondad de un subconjunto de atributos producido por algún procedimiento de generación y este valor es comparado con el mejor valor hasta el momento. Si el valor encontrado es mejor, el nuevo subconjunto reemplaza al anterior. Sin un criterio de parada adecuado, el proceso de SA podría ejecutar de una manera exhaustiva o constantemente a través del espacio de subconjuntos. Los procedimientos de generación y las funciones de evaluación pueden influir en la elección de un criterio de parada.

El criterio de parada basado en un procedimiento de generación incluye: i) si un número predefinido de atributos son seleccionados, ii) si se alcanza un número de iteraciones predefinido y iii) si se obtiene un subconjunto óptimo de acuerdo a alguna función de evaluación. El bucle continúa hasta que algún criterio de parada sea satisfecho. El mecanismo de SA se detiene, dando como salida un subconjunto de atributos seleccionado para un procedimiento de validación. Existen muchas variantes de los tres primeros pasos descritos.

3.4. DESCRIPCIÓN DE LOS FILTROS EMPLEADOS

Roberto Ruiz, José C. Riquelme y Jesús S. Aguilar-Ruiz [Ruiz, Riquelme y Aguilar-Ruiz, 2006] propusieron el método BIRS (*Best*

Incremental Ranked Subset, mejor subconjunto incremental ordenado por ranking) para obtener atributos relevantes y eliminar redundancia. En comparación con otros métodos del estado del arte, este método presenta mejor eficiencia. BIRS pertenece a una categoría híbrida donde el proceso de selección se divide en dos fases: en la primera, los atributos son evaluados individualmente; en la segunda fase, se aplica un evaluador de subconjuntos de atributos a un cierto número de atributos del ranking anterior siguiendo una estrategia de búsqueda.

BIRS puede usar como evaluador de subconjuntos CFS (del inglés *Correlation-based Feature Selection*, selección de atributos basada en correlación) [Hall, 2000] o también CNS (*CoNSistency based measure*, medida basada en consistencia) [Liu y Setiono, 1996] –que se fundamentan en conceptos de correlación y consistencia, respectivamente– en la segunda fase y la medida SOAP (*Selection Of Attributes by Projection*, selección de atributos por proyección) [Ruiz, Riquelme y Aguilar-Ruiz, 2002] o el propio evaluador de subconjuntos en la primera fase como evaluador de ranking. Para guiar la búsqueda, CFS evalúa la calidad de un subconjunto de atributos teniendo en cuenta la hipótesis que los subconjuntos de atributos buenos contienen características altamente correladas con la clase. CNS usa la medida de consistencia, que se calcula a partir de la tasa de inconsistencia, según la cual un subconjunto de atributos es inconsistente si existen al menos dos instancias con los mismos valores de sus atributos aunque con clases diferentes [Dash y Liu, 2003].

El núcleo central del algoritmo CFS es una heurística para evaluar el valor o mérito de un subconjunto de atributos. Tal heurística considera la utilidad de características individuales para predecir la etiqueta de la clase junto con el nivel de intercorrelación entre ellas. La hipótesis en la

que se basa es: buenos subconjuntos de atributos contienen características altamente correladas con la clase, aunque no correladas con las restantes clases. Esto se expresa como sigue:

$$Mérito_S = \frac{k\overline{r_{cf}}}{\sqrt{k + k(k-1)r_{ff}}} \quad (3.1)$$

donde $Mérito_S$ es la heurística del subconjunto S formado por k atributos, $\overline{r_{cf}}$ el promedio de la correlación atributo-clase y r_{ff} el promedio de la intercorrelación atributo-atributo. Para problemas de clasificación, CFS primero discretiza los atributos numéricos y luego emplea la incertidumbre simétrica para estimar el grado de asociación entre atributos discretos.

SOAP se basa en el siguiente principio: contar los cambios de etiqueta (NLC, del inglés *Number of Label Changes*) de los ejemplos proyectados sobre cada atributo. El valor NCL relaciona cada atributo con la etiqueta usada para clasificar. Este valor se calcula proyectando las instancias del conjunto de datos sobre el eje respectivo del atributo, ordenándolas según este atributo. A continuación, se recorre dicho eje desde el origen (representa el valor menor del atributo) hasta el valor mayor del atributo, contando el número de cambios de etiqueta producido.

BIRS trata con la utilidad ordenada por ranking de manera incremental para obtener una aproximación para identificar de manera explícita atributos relevantes y no tener en cuenta atributos redundantes. La idea es escoger cada atributo de una lista ordenada por ranking de uno en uno de la siguiente manera: en primer lugar, las características se ordenan según alguna medida de evaluación (SOAP, CFS o CNS) y, en segundo lugar, BIRS maneja la lista de atributos una vez, marcando el ranking desde el primer al último atributo del ranking. Los resultados de

evaluación se obtienen usando CFS o CNS con el primer atributo de la lista y se marca como seleccionado. A continuación, el resultado se obtiene con los atributos primero y segundo; el segundo será marcado como seleccionado dependiendo si la evaluación obtenida es mejor de una manera significativa. El procedimiento se repite hasta que la última característica en la lista ordenada por ranking se alcanza. Finalmente, el algoritmo devuelve el mejor subconjunto encontrado, del que se puede afirmar que no contiene atributos irrelevantes o redundantes.

El algoritmo FCBF emplea la incertidumbre simétrica (del inglés *symmetrical uncertainty*) en dos pasos. En el primer paso, genera un ranking basado en la incertidumbre simétrica entre cada atributo y la clase. El segundo paso empieza con un conjunto completo de atributos y va eliminando algunos, esto es, encuentra el mejor subconjunto mediante una estrategia de búsqueda secuencial hacia atrás, analizando si un atributo se descarta o no según la incertidumbre simétrica atributo-atributo.

El selector de atributos InfoGain tiene sus fundamentos en la ganancia de información (del inglés *Information Gain*) [Cover y Thomas, 1991], que es un típico concepto usado para evaluar la relevancia de un atributo, junto con Ranker como método de ranking. En BestFirst_CFS la conocida estrategia de búsqueda BestFirst (búsqueda primero el mejor) es combinada con la medida de evaluación de subconjuntos CFS.

En tabla 3.1 se exponen los diferentes métodos de SA que han sido empleados en la presente Tesis Doctoral.

Nombre del selector de atributos	**Método de ranking**	**Evaluación de subconjuntos**
spBI_CFS	spBI	CFS
spBI_CNS	spBI	CNS
cnBI_CNS	cnBI	CNS
InfoGain	Ranker	InfoGain
FCBF	Symmetrical Uncertainty	FCBF
BestFirst_CFS	BestFirst	CFS

Tabla 3.1. Métodos de selección de atributos empleados en la experimentación.

Por tanto, en los experimentos spBI_CFS indica que SOAP se emplea como una medida individual en la primera fase de BIRS y CFS se usa como evaluador de subconjuntos en la segunda fase. De la misma manera, cnBI_CNS denota que el evaluador CNS será usado en ambas fases del algoritmo BIRS.

Capítulo 4

Modelos implementados

4.1. Algoritmo Evolutivo Básico

En el presente capítulo se describe en primer término el algoritmo evolutivo que marca el punto de partida de la presente Tesis Doctoral. A continuación, se detallan todos los modelos implementados. El Algoritmo Evolutivo (AE) básico seguido [Hervás-Martínez, Martínez-Estudillo y Gutiérrez, 2006; Martínez-Estudillo et al., 2006b] permite diseñar simultáneamente la estructura y aprender los coeficientes de una RNUP.

La búsqueda comienza con una población inicial aleatoria que en cada iteración es modificada siguiendo un proceso evolutivo. La población es sometida a las operaciones de replicación y mutación en sus variantes paramétrica y estructural. El operador de cruce no se usa debido a sus desventajas potenciales en redes neuronales evolutivas [Angeline, Saunders y Pollack, 1994; Yao y Liu, 1997]. Con estas propiedades el algoritmo se engloba bajo el paradigma de Programación Evolutiva. En [Ohkura et al., 2007], se ha propuesto el modelo MBEANN, que tiene puntos en común con el AE mencionado, pues lleva a cabo un entrenamiento simultáneo de la estructura y los coeficientes de la red y no emplea el operador de cruce. El esquema general del AE básico para un problema de clasificación está representado en la figura 4.1.

```
Programa: Algoritmo Evolutivo Básico
Datos: Conjunto de entrenamiento
Parámetros de entrada: gen, neu
Constantes:
N ← 1000                          // Tamaño de la población
        // Índices para acceder a individuos que limitan diferentes porcentajes de la población
N10 ←0,1*N                        // 10% de la población
N90 ←0,9*N                        //  90% de la población
N9 ←0,09*N                        //  9% de la población
Salida: Mejor modelo de RNUP
1:  t ← 0
2:  P(t) ← {ind_1, …, ind_10*N}                          // Inicialización aleatoria de la población
3:  f(P(t) {ind_1, …, ind_10*N}) ← aptitud (P(t) {ind_1, …, ind_10*N})      // Cálculo aptitud individuos
4:  P(t) ← ordenar (P(t) {ind_1, …, ind_10*N})   // Ordenación individuos por aptitud: f(ind_i) > f(ind_i+1)
5:  P(t) ← P(t) {ind_1,… ind_N}                          // Conservación de los N mejores individuos
6:  while condición-de-parada no se verifique do         // Bucle principal
7:      P(t) {ind_N90+1,…ind_N} ← P(t) {ind_1, …, ind_N10}   // El 10% mejor reemplaza al 10% peor
8:      P(t+1) ← P(t) {ind_1, …, ind_N90}
9:      P(t+1) ← mp (P(t) {ind_1, …, ind_N9})            // Mutación paramétrica (10% P(t+1))
10:     P(t+1) ← me (P(t) {ind_N9+1, …, ind_N90})        // Mutación estructural (90% P(t+1))
11:     f(P(t+1) {ind_1, .. ind_N90}) ← aptitud (P(t+1) {ind_1, …, ind_N90})      // Evaluación
12:     P(t+1) ← P(t+1) {ind_1, …, ind_N90} ⋃ P(t) {ind_N90+1, …, ind_N}
13:     P(t+1) ← ordenar (P(t+1) {ind_1, …, ind_N})      // Ordenación individuos
14:     t ← t+1
15:     última_generación ← t
16: end while
17: return mejor (P(última_generación) {ind_1})
```

Figura 4.1. Pseudocódigo del AE básico.

Explicamos seguidamente los aspectos principales del AE básico:

1) *Notación, datos, parámetros de entrada, salida, constantes y variables*. Hemos indicado las palabras clave del pseudocódigo en negrita y las funciones en cursiva. Los comentarios aparecen precedidos por "//". Como datos de entrada es necesario un conjunto de entrenamiento. Los parámetros de entrada del AE son el número máximo de generaciones (*gen*) y el número de nodos máximo en la capa oculta (*neu*). Los restantes parámetros a configurar se describen a continuación. Al final de la ejecución, el AE devuelve el mejor modelo de RNUP con un máximo de *neu* nodos en la capa oculta. Las constantes que se emplean para hacer referencia a ciertos valores numéricos, que no se modifican a lo largo de la ejecución del AE, son

N, *N10*, *N90* y *N9*, que representan el tamaño de la población, los índices relativos al 10%, 90% y 9% de la población, respectivamente. Las variables usadas son el número de generación actual (*t*), la última generación (*última_generación*) y dos tablas (arrays), una con los individuos de la población que evoluciona y otra, con las aptitudes de cada individuo, denominadas respectivamente *P* y *f*. Las funciones se denominan *aptitud*, *ordenar*, *me*, *mp* y *mejor*, que, respectivamente, obtienen la aptitud, llevan a cabo la ordenación decreciente en función de la aptitud, aplican mutación estructural, mutación paramétrica y obtiene el mejor individuo.

2) *Representación de los individuos*. Respecto a la representación de los individuos, el AE trata la población como un conjunto de modelos de RNUP. Se adopta una aproximación orientada a objetos y el algoritmo trata directamente con el fenotipo de la RNA. Cada conexión se especifica con un valor binario que indica si la conexión existe y un valor real que representa su peso. Dado que el cruce no se considera, esta representación orientada a objetos no asume un orden fijo entre los nodos de la capa oculta. Todos los individuos de la población tienen el mismo número máximo de neuronas en la capa oculta. El valor concreto debe ser indicado como entrada del AE y éste afecta claramente al rendimiento y complejidad de la red neuronal.

3) *Funciones de error y de aptitud*. Tal como hemos comentado en el apartado 2.4.3, el error cometido por el modelo de red neuronal es la función de error *cross-entropy* (entropía cruzada) [Bishop, 1995], $l(\hat{\boldsymbol{\theta}})$, que viene dada por la ecuación (2.28).

Dado que el objetivo del AE es minimizar la función de error elegida, se emplea una función de aptitud que viene dada por:

$$A(\hat{\boldsymbol{\theta}}) = \frac{1}{1+l(\hat{\boldsymbol{\theta}})} \qquad 0 < A(\hat{\boldsymbol{\theta}}) \leq 1 \tag{4.1}$$

4) *Inicialización de la población*. Al comienzo del AE, $10*tam\_pob$ individuos son generados aleatoriamente (paso 2), siendo tam_pob el tamaño de la población; en todos los experimentos llevados a cabo en la presente Tesis Doctoral tam_pob toma el valor 1000. A continuación, todos los individuos son evaluados, se ordenan por aptitud decreciente y los tam_pob mejores individuos forman la población inicial (pasos 3-5).

5) *Condición de parada*. El bucle principal del AE se repite hasta que se verifique una de las dos siguientes condiciones: a) alcanzar el número máximo de generaciones especificado por el párametro *gen*, que es fijado específicamente para el problema a resolver, o b) que el mejor individuo de la población o la aptitud media de la población no mejore durante $gen-sin-mejorar$ generaciones, que en la presente Tesis Doctoral toma el valor 20, obtenido experimentalmente.

6) *Mutación paramétrica*. La mutación paramétrica modifica el valor de los coeficientes del modelo (paso 9) y consiste en un algoritmo de enfriamiento simulado. La severidad de la mutación de un individuo g de la población la establece la temperatura $T(g)$, que viene dada por:

$$T(g) = 1 - A(g) \qquad 0 \leq T(g) < 1 \tag{4.2}$$

La mutación paramétrica se lleva a cabo para cada exponente w_{ji} y cada coeficiente β_j^l del modelo con ruido gaussiano, donde la varianza depende de la temperatura:

$$w_{ji}(t+1) = w_{ji}(t) + \xi_1(t) \qquad j = 1,...,m \quad i = 1,...,n \tag{4.3}$$

$$\beta_j^l(t+1) = \beta_j^l(t) + \xi_2(t) \qquad j = 0,...,m \quad l = 1,...,J-1 \tag{4.4}$$

donde $\xi_k(t) \in N(0, \alpha_k T(g)) \quad k = 1,2$, sigue una distribución normal unidimensional con media 0 y varianza $\alpha_k(t) \cdot T(g)$, *t* y *t+1* son respectivamente la generación t-ésima y la siguiente. Merece ser destacado que la modificación de los exponentes w_{ji} debe ser más suave que la realizada sobre los β_j^l, siendo recomendable asignar valores a α_1 y α_2 que verifiquen que $\alpha_1 << \alpha_2$. Para la evolución de los parámetros α_1 y α_2 se aplica la regla de éxito 1/5, debido a que converge hacia los valores óptimos muy rápidamente [Rechenberg, 1989].

Según dicha regla, la razón de mutaciones satisfactorias debería ser siempre 1/5. De esta forma, si la razón es mayor que 1/5, la desviación típica debería incrementarse; si es menor que 1/5, debería disminuirse y, en caso contrario, debe dejarse igual. Esto es:

$$\alpha_i(t+s) = \begin{cases} (1+\lambda)\alpha_i(t), & si\ s_g > 1/5 \\ (1-\lambda)\alpha_i(t), & si\ s_g < 1/5 \\ \alpha_i(t), & si\ s_g = 1/5 \end{cases} \qquad i = 1,2 \tag{4.5}$$

siendo s_g la razón de mutaciones satisfactorias durante *s* generaciones. En toda nuestra experimentación hemos tomado $\lambda = 0{,}1$ y $s = 5$, debido a que en el diseño experimental previo se constató que eran valores suficientemente robustos.

7) *Mutación estructural*. La mutación estructural implica una modificación en la estructura del modelo (paso 10) y permite explorar diferentes regiones en el espacio de búsqueda y ayuda a mantener diversidad de modelos en la población. Hay 5 tipos diferentes de mutaciones estructurales; las 4 primeras son similares a las del

modelo GNARL [Angeline, Saunders y Pollack, 1994]: añadir nodos, eliminar nodos, añadir conexión, eliminar conexión y fusión de nodos. Todas las mutaciones comentadas se realizan secuencialmente en la misma generación en el orden especificado sobre cada individuo, con probabilidad $T(g)$, tal como se sugiere en [Angeline, Saunders y Pollack, 1994]. Si el valor de probabilidad obtenido no permite realizar mutación, se elige aleatoriamente una de las 5 mutaciones comentadas y se aplica al individuo. El número de nodos a añadir o eliminar pertenece a {1, 2}. El porcentaje de las conexiones que se añaden o se eliminan es de un 30% para las que van de la capa de entrada a la capa intermedia y de un 5% para las que van de la capa intermedia a la capa de salida. Estos valores son robustos y pequeñas variaciones no producen mejoras significativas en el rendimiento.

8) *Sumario de los parámetros*. Para finalizar la especificación del AE, se explican algunos parámetros o características. Como hemos comentado previamente, el valor de parámetros específicos como el número máximo de generaciones *(gen)* y el número máximo de neuronas en la capa oculta (*neu*) deben indicarse como valores de entrada al AE. No existen valores típicos para ellos, por lo que la dificultad radica en determinar buenos valores para el problema que estamos tratando. Además, el rendimiento del AE depende de dichos valores.

Finalmente, para concluir la explicación del AE, la tabla 4.1 describe los valores de algunos parámetros.

Parámetro / Característica	Valor
Tamaño de la población (tam_pob)	1000
$gen-sin-mejorar$	20
Intervalo para los exponentes w_{ji} y los coeficientes β_j^l	[-5, 5]
Valores iniciales de α_1 y α_2	0,5 y 1 respectivamente
Normalización de los datos de entrada	[1, 2]
Número de nodos implicados en las operaciones añadir nodo y eliminar nodo	{1, 2}

Tabla 4.1. Parámetros / Características del AE básico.

Los valores asignados a los parámetros son robustos y pequeñas variaciones no producen mejoras significativas en el rendimiento. Constituyen el resultado de los estudios experimentales [Hervás-Martínez, Martínez-Estudillo y Gutiérrez, 2006; Martínez-Estudillo et al., 2006b], realizados por diferentes miembros del grupo de investigación AYRNA (Aprendizaje Y Redes Neuronales Artificiales) de la Universidad de Córdoba, que han desarrollado el mencionado AE básico.

4.2. MODELOS IMPLEMENTADOS

A continuación, presentamos las aportaciones de la Tesis Doctoral relativas al entrenamiento de los modelos de redes neuronales de UP y US. La implementación de los modelos se ha realizado en lenguaje Java usando la librería JCLEC (*Java Class Library for Evolutionary Computation*) [Ventura et al., 2008].

4.2.1. Primer Modelo: EDD (Experimental Design Distribution)

4.2.1.1. Planteamiento

Como hemos comentado en el AE básico, el número máximo de generaciones y el número de nodos de la capa oculta son dos parámetros que afectan al rendimiento de la RNA. El número de generaciones, como es conocido por la Teoría de Computación Evolutiva, afecta muy significativamente al rendimiento de un algoritmo evolutivo. Así mismo, los modelos de redes neuronales son muy sensibles al valor del número de nodos en la capa oculta. Por otra parte, el efecto de una mutación en el peso w_{ji}, valor de la conexión entre un nodo de entrada y un nodo oculto, es mayor que el efecto en un coeficiente β_j^l, conexión entre un nodo oculto y un nodo de salida, por lo que los cambios en los exponentes w_{ji} deben ser menores que los que se aplican sobre los coeficientes β_j^l.

El parámetro α_2 actúa sobre los coeficientes de la capa de salida, controlando la diversidad de los individuos de la población; α_2 se multiplica por la temperatura de la red. Al principio de la evolución, hay una alta temperatura que permite movimientos exploratorios de una solución a otra con un valor de aptitud muy diferente, mientras que en la fase final de dicha evolución, la temperatura es baja y valores de α_2 mayores pueden lograr una mayor diversidad.

La motivación de esta propuesta se fundamenta en asumir que la modificación de los tres parámetros recién comentados pueda dar lugar a una mejora de las soluciones. Por otra parte, en el contexto de RNAs la obtención de la arquitectura óptima es un problema que sigue sin resolverse.

En la figura 4.2 se representan los pasos del AE básico. Con el modelo EDD vamos a actuar sobre diferentes acciones. En concreto, respecto a "Comprobar Condición Parada", evaluamos hasta qué punto es interesante modificar el número de generaciones. Por otra parte, el parámetro α_2 está relacionado con el paso denominado "Mutación Paramétrica de la población". La modificación en el número máximo de neuronas en la capa oculta es un aspecto relacionado con la arquitectura de los individuos de la población, por lo que se trata de una alteración sobre la fase en la que se genera la población inicial.

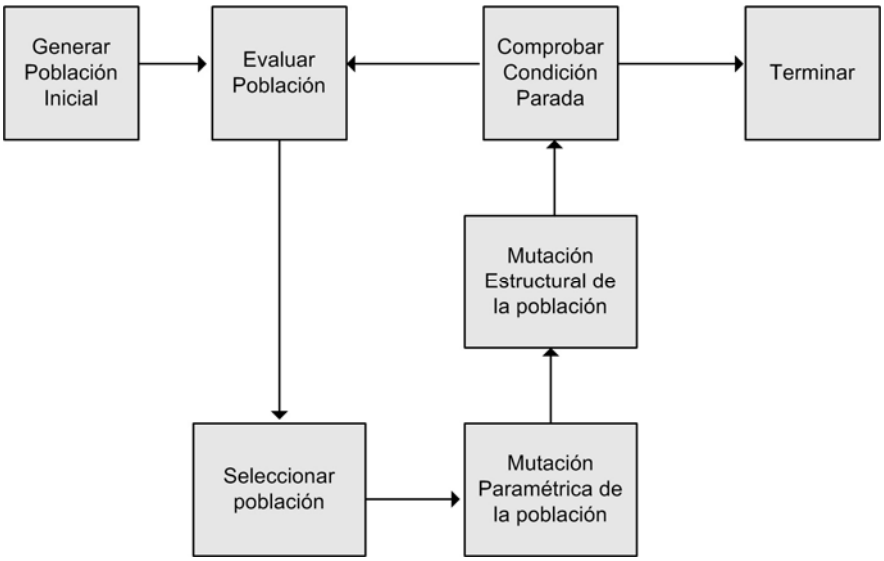


Figura 4.2. Esquema del AE básico.

La evolución de diferentes poblaciones de redes con diferentes parámetros permite observar el avance en paralelo del entrenamiento de los modelos. Si el problema pertenece a un dominio de aplicación, en el que el tiempo de decisión no está totalmente determinado, un entrenamiento distribuido es muy útil. La ocurrencia de algún determinado evento puede obligar a dar una respuesta con carácter inmediato; en este caso, nos quedamos con aquellos modelos que tienen un mejor rendimiento hasta dicho momento de decisión. Esta formulación

se adapta perfectamente al paradigma de programación Maestro/Esclavo, que puede usar para su ejecución máquinas geográficamente distribuidas considerando el avance en el área de Computación Distribuida. Mediante una estrategia secuencial no sería posible entrenar al mismo tiempo tantos modelos de red neuronal, debido a que hasta que no terminara el entrenamiento con la primera población de redes no podría comenzar el entrenamiento de la segunda población.

Existen diversas estrategias de paralelización de las metaheurísticas según la fuente de paralelismo usada. La metodología propuesta en esta Tesis puede considerarse como un ejemplo de paralelismo que se obtiene de múltiples exploraciones concurrentes del espacio de soluciones [Crainic y Toulouse, 2003]. En esta estrategia cada hilo o proceso concurrente puede ejecutar el mismo o diferente método heurístico; en nuestro caso, todos los procesos ejecutan la misma heurística, un AE, aunque con diferentes ajustes de parámetros. Por otra parte, entre los diferentes esquemas que existen, EDD se adapta al denominado método de búsqueda independiente, que permite llevar a cabo una buena exploración del espacio de soluciones. Como es lógico pensar, a mayor número de procesos o hilos independientes, la exploración es más exhaustiva. En nuestro caso, para cada problema se ejecutan como máximo 8 procesos en paralelo.

Dentro del campo de los AG, se encuentran los denominados AG paralelos de grano grueso, según los cuales una población se divide en varias subpoblaciones de individuos que evolucionan en paralelo. EDD, sin embargo, se diferencia en que no tenemos varias subpoblaciones, sino poblaciones completas. Este hecho, por un lado, limita nuestro modelo, pues requiere más tiempo de computación, pero, por otra parte, lo hace mejor explorador, debido a que en cada población tenemos un mayor

número de individuos y, por tanto, podría suponerse que más información genética.

Algunos investigadores han trabajado en la temática de la distribución de parámetros de algunas metaheurísticas, especialmente de AG, en problemas de optimización [Herrera y Lozano, 2000]. En dicho dominio, se utilizan estrategias que hacen uso de multitud de variantes de operadores de cruce y mutación sobre diferentes subpoblaciones, debido a que las mayores posibilidades de evolución se encuentran en el intercambio de material genético entre individuos, así como pequeñas alteraciones de los genes (mutaciones). Sin embargo, en el área de RNAE no conocemos que existan trabajos sobre esta temática.

4.2.1.2. Descripción

El primer modelo implementado se denomina EDD (del inglés *Experimental Design Distribution*, Distribución del Diseño Experimental). Se distribuyen ciertos parámetros asociados a la topología del modelo de RNUP o bien al AE a lo largo de todos los elementos de procesamiento del sistema de computación. Los parámetros que se distribuyen son el número máximo de neuronas en la capa oculta (*neu*), el número máximo de generaciones (*gen*) y el valor del parámetro α_2, que afecta a la distribución que se emplea para modificar los coeficientes β_j^l en la mutación paramétrica. Hemos elegido estos tres parámetros porque experimentalmente mostraban que pequeños cambios determinaban rendimientos significativamente diferentes. Se han propuesto dos variantes: una que distribuye tres parámetros y otra que distribuye dos parámetros.

Existe una configuración base que se modifica en cada uno de los nodos de cómputo. Por tanto, cada uno de éstos tiene una misión específica y siempre ajusta los mismos parámetros. Los cambios se han definido de manera relativa, por lo que dependen de la configuración base. Desde el proceso maestro se lanzan los procesos esclavos que se encargan de realizar tales modificaciones en los restantes nodos y ejecutar la nueva configuración resultante. La figura 4.3 ilustra la jerarquía de procesos del modelo EDD. La configuración 1 (Config. 1) corresponde a la configuración base. Como infraestructura de computación, hemos empleado para esta primera propuesta un cluster formado por 8 nodos de computación, lo que explica que se hayan establecido 8 configuraciones. Lo ideal es llevar a cabo la experimentación asociada a esta primera propuesta en un sistema distribuido haciendo uso de tantos procesadores como configuraciones se deseen ejecutar. En caso de que por algún motivo no se desee hacer uso de tal sistema, es posible ejecutar las configuraciones de una manera secuencial en cualquier ordenador.

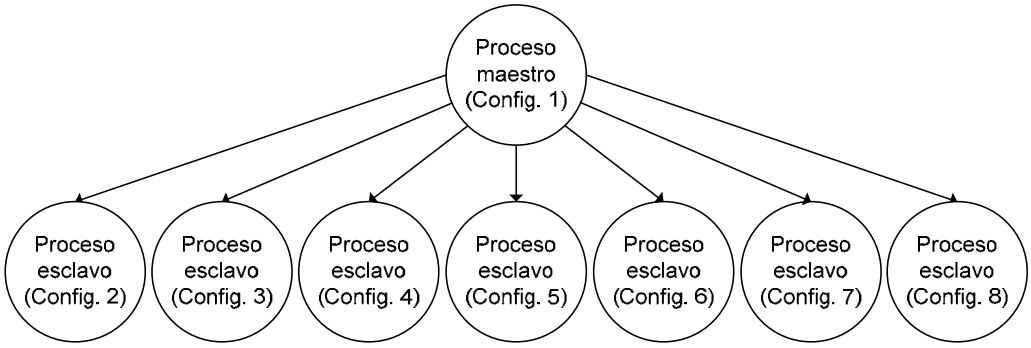


Figura 4.3. Jerarquía de procesos del modelo EDD.

En las tablas 4.2 y 4.3 se muestran las configuraciones de EDD asociadas a las variantes que distribuyen 3 y 2 parámetros, respectivamente. En las celdas de dichas tablas aparecen los valores que toma el parámetro indicado en la correspondiente columna; el valor concreto del parámetro *neu*, *gen* y α_2 se ha representado de manera

genérica con **neu, gen** y α_2, afectado en algunos casos por una operación aritmética elemental, que, como se observa, aparecen en negrita, indicando que el valor concreto depende o bien del conjunto de datos a entrenar (**neu** y **gen**) o del número de configuración (α_2).

Habitualmente, para cada una de las configuraciones se realizan 30 repeticiones con diferentes semillas. Se ha implementado un submodelo derivado de EDD, que permite dividir las repeticiones entre varios nodos de cómputo; el número de nodos fue establecido en 1, 2, 4 u 8 pensando en realizar la experimentación en un cluster de 8 nodos de cálculo.

Configuración	**Número máximo de neuronas (*neu*)**	**Número máximo de generaciones (*gen*)**	α_2
1	$\mathbf{neu}$	$\mathbf{gen}$	α_2
2	$\mathbf{neu}+1$	$\mathbf{gen}$	α_2
3	$\mathbf{neu}$	$\mathbf{gen}$	$1,5*\alpha_2$
4	$\mathbf{neu}+1$	$\mathbf{gen}$	$1,5*\alpha_2$
5	$\mathbf{neu}$	$0,8*\mathbf{gen}$	α_2
6	$\mathbf{neu}+1$	$0,8*\mathbf{gen}$	α_2
7	$\mathbf{neu}$	$0,8*\mathbf{gen}$	$1,5*\alpha_2$
8	$\mathbf{neu}+1$	$0,8*\mathbf{gen}$	$1,5*\alpha_2$

Tabla 4.2. Configuraciones de EDD con 3 parámetros.

Configuración	**Número máximo de neuronas (*neu*)**	**Número máximo de generaciones (*gen*)**
1	$\mathbf{neu}$	$\mathbf{gen}$
2	$\mathbf{neu}+1$	$\mathbf{gen}$
3	$\mathbf{neu}+2$	$\mathbf{gen}$
4	$\mathbf{neu}+3$	$\mathbf{gen}$

5	**neu**	$0,8 * \mathbf{gen}$
6	$\mathbf{neu} + 1$	$0,8 * \mathbf{gen}$
7	$\mathbf{neu} + 2$	$0,8 * \mathbf{gen}$
8	$\mathbf{neu} + 3$	$0,8 * \mathbf{gen}$

Tabla 4.3. Configuraciones de EDD con 2 parámetros.

Inicialmente, el modelo EDD fue propuesto para trabajar con RNUP. Con posterioridad, fue extendido el modelo a RNUS. El nuevo modelo fue denominado EDDSig (del inglés *Experimental Design Distribution with Sigmoidal units*, Distribución del Diseño Experimental con Unidades Sigmoide).

4.2.2. Segundo Modelo: TSEA (Two-Stage Evolutionary Algorithm)

4.2.2.1. Planteamiento

El segundo modelo propuesto diversifica la arquitectura de la RNA al comienzo de la evolución. Para ello, crea dos poblaciones con diferentes características (máximo número de nodos en la capa oculta), les aplica el AE básico con los mismos valores de los parámetros durante un número pequeño de generaciones, selecciona la mitad mejor de cada población y las combina para crear una nueva población. En la segunda fase, aplica el bucle principal del AE básico durante el ciclo evolutivo completo.

De esta manera, la población tiene mayor diversidad debido a las diferentes topologías. El breve entrenamiento inicial permite que los individuos creados aleatoriamente puedan explorar posibles zonas prometedoras en diferentes direcciones, debido a que la topología es diferente. A continuación, individuos con diferentes topologías coexisten y los más adecuados permanecerán en la población.

Respecto a los trabajos relacionados con este modelo, no hemos encontrado ninguno que se le aproxime. Juan R. Rabuñal y sus colaboradores, [Rabuñal et al., 2005] contemplan una metaheurística diferente a los AEs como son los AG, que usan dos poblaciones para entrenar los modelos de RNAs. La idea de una de las poblaciones es almacenar los individuos que siguen el proceso evolutivo y la otra guarda los individuos elite.

4.2.2.2. Descripción

Este modelo ha sido denominado TSEA (del inglés *Two-Stage Evolutionary Algorithm*, Algoritmo Evolutivo en Dos Fases). En primer lugar, el usuario define un número máximo de nodos en la capa oculta, *neu*. La primera fase consiste en crear dos poblaciones con neu y neu+1 nodos, respectivamente, en la capa oculta que son inicializadas siguiendo el mismo mecanismo descrito en los pasos 3-5 del AE básico. A continuación, cada una de dichas poblaciones, de tamaño tam_pob evoluciona durante $0{,}1 * gen$ generaciones. Seguidamente, se seleccionan los $tam\_pob/2$ mejores individuos de cada población y se fusionan en una nueva población. En la segunda fase, esta nueva población es evolucionada durante un ciclo evolutivo completo (pasos 6-17 del AE básico). Los parámetros son dos: el número de generaciones (*gen*) y el valor base del número de nodos en la capa oculta (*neu*). De manera opcional, se puede establecer un valor diferente para el parámetro α_2 que, por defecto, toma el valor 1. El pseudocódigo de TSEA aparece en la figura 4.4.

```
Programa: Algoritmo Evolutivo en dos etapas
Datos: Conjunto de entrenamiento
Parámetros de entrada: gen, neu
Constantes:
N ← 1000                        // Tamaño de la población
        // Índices para acceder a individuos que limitan diferentes porcentajes de la población
N10 ←0,1*N                      // 10% de la población
N90 ←0,9*N                      //  90% de la población
N9 ←0,09*N                      //  9% de la población
Salida: Mejor modelo de RNUP
1:                                  // Primera etapa
2:  t ← 0
3:                                  // Población P1
4:  P1(t) ← {ind1, …, ind10*N}          // Los individuos de P1 tienen neu nodos en la capa oculta
5:  f1 (P1(t) {ind1, …, ind10*N}) ← aptitud (P1(t) {ind1, …, ind10*N})      // Cálculo aptitud individuos
6:  P1(t) ← P1(t) {ind1, …, ind10*N}        // Ordenación de individuos por aptitudes decrecientes
7:  P1(t) ← P1(t) {ind1,… indN}             // Conservación de los N mejores individuos
8:                                  // Población P2
9:  P2(t) ← {ind1, …, ind10*N}          // Los individuos de P2 tienen neu+1 nodos en la capa oculta
10: f2 (P2(t) {ind1, …, ind10*N}) ← aptitud (P2(t) {ind1, …, ind10*N})      // Cálculo aptitud individuos
11: P2(t) ← P2(t) {ind1, …, ind10*N}        // Ordenación de individuos por aptitudes decrecientes
12: P2(t) ← P2(t) {ind1,… indN}             // Conservación de los N mejores individuos
13:                             // Evolución de las poblaciones P1 y P2 durante 0,1*gen generaciones
14: for each Pi
15:   t ← 0
16:    while t < 0,1*gen no se verifique do
17:       Pi(t) {indN90+1,…indN} ← Pi(t) {ind1, …, indN10}     // El 10% mejor reemplaza al 10% peor
18:       Pi(t+1) ← Pi(t) {ind1, …, indN90}
19:       Pi(t+1) ← mp (Pi(t+1) {ind1, …, indN9})          // Mutación paramétrica (10% Pi (t+1))
20:       Pi(t+1) ← me (Pi(t+1) {indN9+1, …, indN90})      // Mutación estructural (90% Pi (t+1))
21:       fi(Pi(t+1) {ind1, .. indN90}) ← aptitud (Pi(t+1) {ind1, …, indN90})          // Evaluación
22:       Pi(t+1) ← Pi(t+1) {ind1, …, indN90} ⋃ Pi(t) {indN90+1, …, indN}
23:       Pi(t+1) ← Pi(t+1){ind1, …, indN}                          // Ordenación de individuos
24:       t ← t + 1
25:    end while
26: end for
27: P(t) ← P1{ind1, …, indN/2} ∪ P2{ind1, …, indN/2}     // Los individuos de P tienen [neu, neu+1]
28:                                                  // nodos en la capa oculta
29:  P(t) ← P(t) {ind1, …, indN}                 // Ordenación de individuos por aptitud: indi > indi+1
30:                                 // Segunda etapa
31:                                 // Parámetros de entrada: gen, neu+1
32:  t ← 0
33:  while condición-de-parada no se verifique do          // Bucle principal
34:       P(t) {indN90+1,…indN} ← P(t) {ind1, …, indN10}     // El 10% mejor reemplaza al 10% peor
35:       P(t+1) ← P(t) {ind1, …, indN90}
36:       P(t+1) ← mp (P(t+1) {ind1, …, indN9})          // Mutación paramétrica (10% P(t+1))
37:       P(t+1) ← me (P(t+1) {indN9+1, …, indN90})      // Mutación estructural (90% P(t+1))
38:       f(P(t+1) {ind1, .. indN90}) ← aptitud (P(t+1) {ind1, …, indN90})          // Evaluación
39:       P(t+1) ← P(t+1) {ind1, …, indN90} ∪ P(t) {indN90+1, …, indN}
40:       P(t+1) ← P(t+1) {ind1, …, indN}                           // Ordenación de individuos
41:       t ← t+1
42:       última_generación ← t
43:  end while
44:  return mejor (P(última_generación) {ind1})
```

Figura 4.4. Pseudocódigo del algoritmo TSEA para un problema de clasificación.

En la figura 4.5, se presenta el esquema del modelo TSEA.

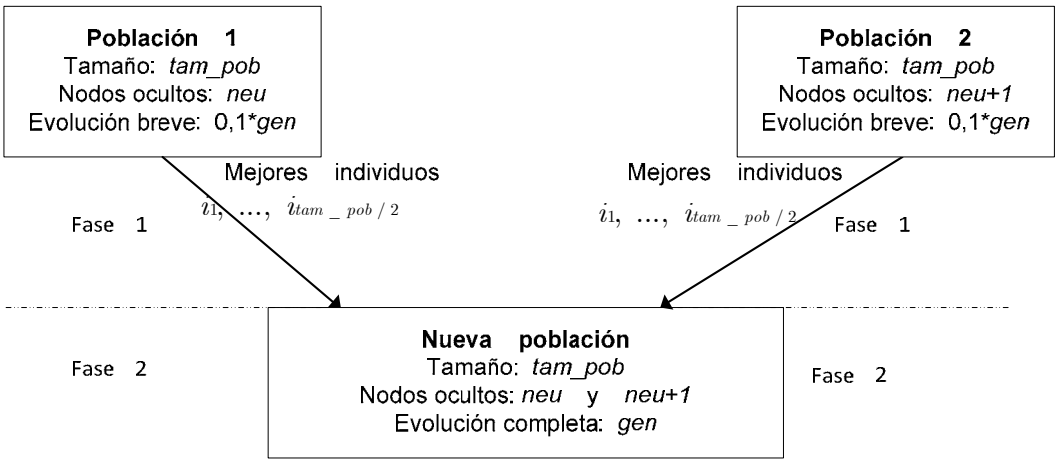


Figura 4.5. Esquema del modelo TSEA.

En el modelo TSEA se han definido dos configuraciones considerando diferentes valores para el parámetro α_2, que aparecen en la tabla 4.4. En EDD, tanto en la versión con 2 parámetros como en la versión de 3, las dos primeras configuraciones sólo difieren en una unidad en el número de nodos en la capa oculta. Los valores de los parámetros que dependen del conjunto de datos se han representado en negrita en la tabla 4.4. Ahora, en TSEA la primera configuración trata con individuos cuyas topologías difieren en un nodo en la capa oculta. En cierto modo, un experimento en TSEA es equivalente a dos de los realizados con EDD.

Configuración	**Número máximo de neuronas en cada población (*neu*)**	**Tamaño de cada población**	**Número máximo de generaciones en cada población (*gen*) (Fase 1)**	α_2
1*	**neu** y **neu**+1	1000	0,1***gen**	$\boldsymbol{\alpha_2}$
2*	**neu** y **neu**+1	1000	0,1***gen**	1,5*$\boldsymbol{\alpha_2}$

Tabla 4.4. Configuraciones de TSEA.

El modelo TSEA ha sido formulado para ser empleado con RNUP. Recientemente, ha sido ampliado para considerar RNUS. La nueva propuesta ha sido denominada TSEASig (*Two-Stage Evolutionary Algorithm for neural networks with Sigmoidal units*, algoritmo evolutivo en dos fases para redes neuronales con unidades sigmoide). Se llevó a cabo una comparación frente a TSEA y EDDSig.

4.2.3. Tercer Modelo: EDDFS (Experimental Design Distribution with Feature Selection)

4.2.3.1. Planteamiento

El modelo EDD es muy útil debido a que permite entrenar de una manera automática modelos de RNUP, empleando en paralelo diferentes configuraciones de parámetros. Si se dispone de un conocimiento previo del problema, es posible que sepamos que con un determinado número de nodos y generaciones la tasa de acierto está cercana a un determinado valor. Para estos casos, proponemos como vía de mejora aplicar selección de atributos y tratar de mantener o reducir el número de nodos y/o generaciones.

4.2.3.2. Descripción

El tercer modelo implementado se designa EDDFS (del inglés *Experimental Design Distribution with Feature Selection*, Distribución del Diseño Experimental con Selección de Atributos).

EDDFS combina métodos de selección de atributos implementados mediante filtros, de manera individual, con la metodología EDD. Dado el conjunto de entrenamiento, aplicamos de manera independiente varios filtros y obtenemos para cada uno de ellos un conjunto de atributos

reducido. Del conjunto de entrenamiento y generalización original, nos quedamos para todas las instancias con los valores de los atributos seleccionados y el de la etiqueta que representa la clase. A continuación, sobre ambos ficheros se emplea la metodología EDD, como hemos comentado en el apartado 4.2.1.

En el modelo EDDFS hemos definido 4 configuraciones que aparecen en la tabla 4.5. Los parámetros son el número de nodos en la capa oculta, el número de generaciones y el valor de α_2, denominados *neu'*, *gen'* y α_2, respectivamente. Se resaltan en negrita los valores de los parámetros que dependen del conjunto de datos, sin tener en cuenta las operaciones elementales adicionales posteriores, suma o multiplicación, que se requieren en algunas configuraciones.

Configuración	**Número máximo de neuronas (*neu'*)**	**Número máximo de generaciones (*gen'*)**	α_2
1′	**neu'**	**gen'**	$\boldsymbol{\alpha_2}$
2′	**neu'** $+1$	**gen'**	$\boldsymbol{\alpha_2}$
3′	**neu'**	**gen'**	$1,5 * \boldsymbol{\alpha_2}$
4′	**neu'** $+1$	**gen'**	$1,5 * \boldsymbol{\alpha_2}$

Tabla 4.5. Configuraciones de EDDFS.

Para este modelo hemos empleado los filtros que aparecen enumerados en la tabla 4.6. Usamos el símbolo "-" para expresar que no se ha aplicado ningún tipo de selector de atributos.

Nombre del selector de atributos	**Método de ranking**	**Evaluación de subconjuntos**
-	Ninguno	Ninguno
spBI_CFS	spBI	CFS
spBI_CNS	spBI	CNS

cnBI_CNS	cnBI	CNS
InfoGain	Ranker	InfoGain
FCBF	Symmetrical Uncertainty	FCBF
BestFirst_CFS	BestFirst	CFS

Tabla 4.6. Métodos de selección de atributos empleados en EDDFS.

4.2.4. Cuarto Modelo: TSEAFS (Two-Stage Evolutionary Algorithm with Feature Selection)

4.2.4.1. Planteamiento

El modelo TSEA es muy eficaz y eficiente y permite reducir de una manera considerable el número de pruebas a realizar para lograr un buen rendimiento, combinando dos poblaciones de individuos con diferentes características. Sin embargo, en este cuarto modelo se lleva a cabo un preprocesamiento del conjunto de datos de entrenamiento con el fin de mejorar la eficacia y la sencillez de los modelos obtenidos. Para el preprocesamiento se han empleado técnicas de selección de atributos implementadas como filtros.

4.2.4.2. Descripción

El cuarto modelo implementado se denomina TSEAFS (del inglés *Two-Stage Evolutionary Algorithm with Feature Selection*, Algoritmo Evolutivo en Dos Fases con Selección de Atributos). La nueva metodología combina TSEA y una fase de preprocesamiento. En primer lugar, varios selectores de atributos implementados mediante filtros se aplican de manera independiente al conjunto de entrenamiento de un problema para obtener una lista de atributos que se empleará posteriormente, tanto en la fase de entrenamiento como en la fase de generalización, para obtener el modelo de RNUP reducido y evaluarlo, respectivamente. De este modo, se

generan dos conjuntos reducidos de datos, conjunto de entrenamiento y de generalización reducidos, donde sólo se incluyen las características más relevantes. Es importante destacar que la selección de atributos sólo se realiza sobre el conjunto de entrenamiento; los conjuntos de datos reducidos se emplean como entrada para el algoritmo TSEA. La figura 4.6 ilustra el esquema de la metodología propuesta. TSEAFS presenta dos etapas: i) selección de atributos y ii) clasificación por medio de TSEA.

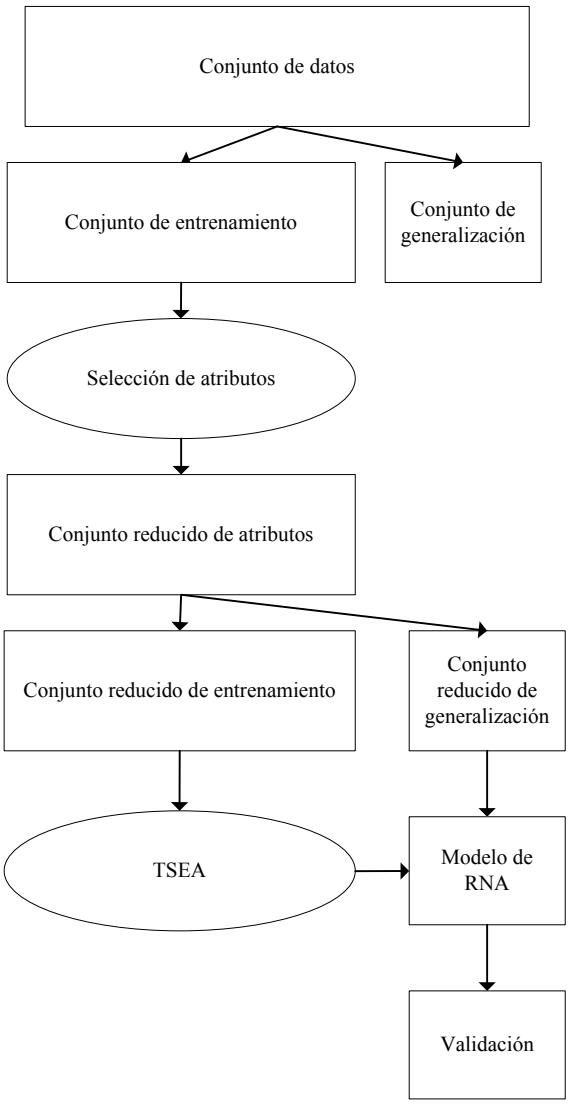


Figura 4.6. Esquema del modelo TSEAFS.

El número de configuraciones empleado es 2, como se aprecia en la tabla 4.7. Los parámetros a definir son: a) el número de neuronas en la capa oculta teniendo en cuenta que el número de entradas, en la gran mayoría de los casos, es inferior al del problema original y b) el número máximo de generaciones, que habitualmente será menor al de los modelos obtenidos con TSEA, debido a que ahora el modelo a obtener es más simple. Estos parámetros se representan por *neu#* y *gen#*. Como en anteriores ocasiones, los valores dependientes del problema a resolver se muestran en negrita.

Configuración	**Número máximo de neuronas en cada población (*neu#*)**	**Tamaño de cada población**	**Número máximo de generaciones en cada población (Fase 1)**	α_2
1*#	**neu#** y **neu#**+1	1000	0,1***gen#**	α_2
2*#	**neu#** y **neu#**+1	1000	0,1***gen#**	1,5*α_2

Tabla 4.7. Configuraciones de TSEAFS.

En la tabla 4.8, se exponen los diferentes filtros empleados en la etapa de selección de atributos. El símbolo "-" indica que no se ha aplicado ningún tipo de filtro.

Nombre del selector de atributos	**Método de ranking**	**Evaluación de subconjuntos**
-	Ninguno	Ninguno
spBI_CFS	spBI	CFS
cnBI_CNS	cnBI	CNS
FCBF	Symmetrical Uncertainty	FCBF
BestFirst_CFS	BestFirst	CFS
sp significa SOAP, BI es BIRS y cn simboliza CNS		

Tabla 4.8. Métodos de selección de atributos empleados en TSEAFS.

Capítulo 5

Diseño experimental

5.1. CONJUNTOS DE DATOS

En diferentes momentos de la experimentación de la presente Tesis hemos empleado variados conjuntos de datos. La mayoría de ellos están disponibles públicamente en el conocido repositorio internacional UCI [Frank y Asuncion, 2010] y otros relativos a problemas complejos del mundo real. En concreto, se han usado los siguientes 31 conjuntos de datos, tal y como refleja la tabla 5.1: *Appendicitis*, Statlog (*Australian* credit approval), *Balance*, *Breast* Cancer, Breast Tissue (*Breast-t*), Breast *Cancer* Wisconsin, *Cardiotocography*, Statlog (*Heart*), Heart disease Cleveland (*Heart-c*), *HeartY*, *Hepatitis*, *Horse* colic, Thyroid disease (allhypo, *Hypothyroid*), *Ionos* (Ionosphere), *Labor* Relations, *Led24*, *Liver* disorders, *Lymphography*, Thyroid disease (*Newthyroid*), *Parkinsons*, *Pima* Indians diabetes, Steel *Plates* Faults, Molecular Biology (*Promoter* Gene Sequences), *SPECTF*, *Vowel*, *Waveform* database generator (version 2), Wine Quality (*Winequality-red*), *Yeast*, *BTX*, *Listeria* monocytogenes y *Liver-transplantation*.

Conjuntos de datos	**Total patrones**	**Patrones entrenamiento**	**Patrones generalización**	**Atributos**	**Entradas**	**Clases**
Appendicitis	106	80	26	7	7	2
Australian	690	517	173	14	51	2

Balance	625	469	156	4	4	3
Breast	286	215	71	9	15	2
Breast-t	106	81	25	9	9	6
Cancer	699	525	174	10	9	2
Cardiotocography	2126	1594	532	23	31	3
Heart	270	202	68	13	13	2
Heart-c	303	227	76	13	26	2
HeartY	270	202	68	13	13	2
Hepatitis	155	117	38	19	19	2
Horse	368	276	92	27	83	2
Hypothyroid	3772	2829	943	29	29	4
Ionos	351	263	88	34	34	2
Labor	57	43	14	16	29	2
Led24	3200	200	3000	24	24	10
Liver	345	259	86	6	6	2
Lymphography	148	111	37	18	38	4
Newthyroid	215	161	54	5	5	3
Parkinsons	195	146	49	23	22	2
Pima	768	576	192	8	8	2
Plates	1941	1457	484	27	27	7
Promoter	106	80	26	58	114	2
SPECTF	267	80	187	44	44	2
Vowel	990	528	462	12	11	11
Waveform	5000	3750	1250	40	40	3
Winequality-red	1599	1196	403	11	11	6
Yeast	1484	1112	372	8	8	10
BTX	63	42	21	3	3	7
Listeria	539	305	234	4	4	2
Liver-transplantation	615	462	153	39	53	2

Tabla 5.1. Resumen de los 31 conjuntos de datos empleados en diferentes momentos de la experimentación.

Los 3 últimos conjuntos de datos aludidos corresponden a problemas complejos del mundo real. *BTX* es un problema medioambiental de clasificación multiclase de diferentes tipos de agua [Hervás et al., 2008]. Los datos de este problema se obtuvieron usando un conjunto de 63 muestras de agua combinadas con niveles individuales de Benceno, Tolueno o Xileno, así como con mezclas binarias o ternarias de ellos con concentraciones entre 5 y 30 μg/l, las cuales constituyen un conjunto de datos global de 7 clases diferentes de agua contaminada. El número de patrones de cada clase es el mismo.

Listeria monocytogenes es un problema biclase perteneciente al área de Microbiología Predictiva. Ha sido una cuestión concerniente a las industrias alimentarias debido a su aparición en el entorno natural [Beuchat, 1996; Fenlon, Wilson y Donachie, 1996] y a las condiciones específicas de crecimiento del patógeno que llevan a su alta prevalencia en diferentes tipos de alimentos. La motivación que impulsó esta investigación fue el problema de listeriosis [Tienungoon et al., 2000]; se propusieron diferentes estrategias para limitar los niveles contaminantes en el tipo de consumo a menos de 100 CFU/g (CFU, del inglés *Colony Forming Unit*; es un valor que indica el grado de contaminación microbiológica de un ambiente) [European Commission, 1999]. Se llevó a cabo un diseño factorial fraccionado para descubrir los niveles de crecimiento de *Listeria monocytogenes*. Los datos fueron obtenidos [Valero et al., 2007] en concentraciones de ácidos cítrico y ascórbico entre 0% y 0,4% (w/v) en intervalos de 0,05%, a temperaturas de 4º, 7º, 10º, 15º y 30ºC y niveles de pH de 4,5, 5, 5,5 y 6. Este conjunto de datos fue dividido de tal manera que 305 condiciones que cubren el dominio extremo del modelo fueron elegidas para entrenamiento y 234 condiciones fueron seleccionadas dentro del rango del modelo para probar su capacidad de

generalización. Entre las diferentes condiciones probadas, hay 240 casos de no crecimiento y 299 casos de crecimiento del microorganismo.

El trasplante hepático (del inglés *liver-transplantation*) está fuertemente limitado por la disponibilidad de donantes adecuados de hígado. El desequilibrio entre la demanda, receptores, y los donantes está desafortunadamente acompañado del terrible escenario de muertos en las listas de espera. Se han realizado múltiples esfuerzos para donantes con criterios expandidos y para priorizar los receptores de las listas de espera. La aceptación de donante e injerto –considerando la escasez de órganos y los criterios expandidos-, la priorización de candidatos –incluyendo factor de mortalidad en las listas de espera- y la política de asignación –combinando los principios de justicia, eficacia y eficiencia- representan un complejo escenario que no es fácil modelar. Más de 100 variables pueden ser consideradas en una decisión clínica concreta para aceptar donante y órgano, lograr su asignación y encontrar el mejor emparejamiento del par donante-receptor. Los datos corresponden a pacientes de diferentes unidades de trasplante hepático de España y presentan una importante dificultad en la distribución de los datos debido al importante desbalanceo existente entre las instancias de las dos clases.

Por otra parte, en relación a los 31 conjuntos de datos, hay que resaltar que el tamaño de los conjuntos oscila entre 57 y 5.000 instancias. El número de atributos depende del problema y varía entre 3 y 58, mientras que el número de entradas está comprendido entre 3 y 114 atributos. El número de clases va desde 2 hasta 11. La columna Entradas representa el número de nodos de la capa de entrada en el modelo de RN. Obviamente, cuando se aplica selección de atributos el número de entradas, en la mayoría de los casos, es inferior; los valores concretos de estos casos se presentan en el siguiente capítulo. En los modelos con preprocesamiento

de datos hemos experimentado con conjuntos de datos con una tasa de error sobre el conjunto de generalización en torno al 20% con clasificadores C4.5 o 1-NN.

5.2. DISEÑO EXPERIMENTAL

Dado que estamos empleando redes neuronales, todas las variables nominales se han convertido en binarias. Como consecuencia, en algunas ocasiones el número de entradas es mayor que el número de atributos. Así mismo, los valores perdidos han sido reemplazados, en el caso de variables nominales, por la moda y, en el caso de variables continuas, por la media, teniendo en cuenta el conjunto de datos completo.

El diseño experimental emplea la técnica de validación cruzada llamada *hold-out* que consiste en dividir los datos en dos conjuntos: un conjunto de entrenamiento y otro de generalización. El primero se emplea para entrenar la red neuronal y el segundo permite evaluar el proceso de entrenamiento y medir la capacidad de generalización de la red neuronal. En nuestro caso, el tamaño del conjunto de entrenamiento es $3n/4$ y $n/4$ para el conjunto de generalización, siendo n el número de patrones del problema. Estos porcentajes son similares a los empleados en [Prechelt, 1994]. Más concretamente, hemos empleado un *hold-out* estratificado en el que ambos conjuntos están estratificados [Kohavi, 1995], de tal manera que la distribución de clases de las instancias en cada conjunto es aproximadamente la misma que en el conjunto completo de datos original. Estas proporciones no se cumplen en algunos conjuntos de datos, ya que los datos están organizados específicamente debido al dominio del problema como en los problemas del mundo real *BTX* [Hervás et al., 2008] y *Listeria* [Valero et al., 2007].

Capítulo 6

Resultados

6.1. INTRODUCCIÓN

En este capítulo se presentan los resultados obtenidos en el conjunto de generalización con los diferentes modelos. Como medidas de evaluación se emplea la precisión (o CCR) y, ocasionalmente, la Mínima Sensibilidad (MS). Se han realizado múltiples comparaciones con otros clasificadores, implementados en la versión 3.7.4 de la herramienta WEKA (*Waikato Environment for Knowledge Analysis*) [Hall et al., 2009], que ha sido desarrollada por un grupo de investigadores de la Universidad de Waikato, ubicada en Hamilton, Nueva Zelanda. El número de repeticiones para los algoritmos no deterministas ha sido 30.

6.2. PRIMER MODELO: EDD

6.2.1. Resultados preliminares

En este primer modelo, que hemos denominado EDD, se presentaron dos posibilidades que llevaban a cabo la distribución de dos o tres parámetros. Se experimentó con 8 configuraciones en ambos casos, que aparecen en las tablas 4.2 y 4.3, respectivamente. Los valores de los parámetros en cada configuración están definidos de manera relativa

respecto a los valores de la configuración base. En las tablas 6.1 y 6.2 se detallan los parámetros de la configuración base para los modelos que distribuyen 3 y 2 parámetros, que son específicos para cada conjunto de datos.

Conjunto de datos	**Número máximo de neuronas (*neu*)**	**Número máximo de generaciones (*gen*)**	α_2
Balance	5	150	1
Cancer	2	100	1
Pima	3	120	1

Tabla 6.1. Valores de los parámetros de las configuraciones base de EDD con 3 parámetros.

Conjunto de datos	**Número máximo de neuronas (*neu*)**	**Número máximo de generaciones (*gen*)**
Hypothyroid	3	500
Waveform	3	500

Tabla 6.2. Valores de los parámetros de las configuraciones base de EDD con 2 parámetros.

Los parámetros fueron determinados a partir de la información publicada en trabajos relacionados con esta Tesis [Hervás-Martínez, Martínez-Estudillo y Gutiérrez, 2006; Martínez-Estudillo et al., 2006b], junto con un diseño experimental previo sobre el conjunto de entrenamiento, aplicando una validación cruzada de tipo *5-fold* con 5 repeticiones. Se tomaron diferentes grupos de valores para los parámetros en función del número de instancias del conjunto de datos. Aquéllos con menos de 1000 instancias se consideran problemas de tamaño pequeño y los que tienen más de 1000 instancias problemas de tamaño grande.

En el primer caso, los valores para el número máximo de neuronas en la capa oculta y el número de generaciones son {2, 3, 4 y 5} y {100, 120 y 150}, respectivamente. En el segundo caso, esto es, en los problemas de tamaño grande, los valores fueron {2, 3, 4, 5, 6, 7 y 8} para el número máximo de neuronas en la capa oculta, mientras que para el número de generaciones se definieron valores medianos, grandes y muy grandes: {300, 500, 1000}. En concreto, para *Hypothyroid* y *Waveform*, con 5 y 6 neuronas en ambos casos, aparecía el conocido problema de sobreentrenamiento, por lo que no fue necesario probar valores superiores.

Las tablas 6.3 y 6.4 reflejan los resultados medios obtenidos en la fase de generalización considerando 30 iteraciones ejecutadas con el modelo EDD y distribuyendo 3 y 2 parámetros, respectivamente. Se muestra para cada conjunto de datos y configuración la topología, los valores medios y las desviaciones típicas del CCR_{gen} y del número de conexiones de los respectivos modelos. Los mejores valores medios de CCR_{gen} obtenidos para cada conjunto de datos, junto con su respectivo número de conexiones, se han representado en negrita.

Conjunto de datos	**Configuración**	**Topología**	**CCR_{gen}± D. Típica**	**Número_Conexiones± D. Típica**
Balance	1	4: 5: 2	95,30±1,47	22,87±2,52
	2	4: 6: 2	95,15±0,98	25,23±2,22
	3	4: 5: 2	95,04±1,41	23,37±3,06
	4	4: 6: 2	**95,62±1,16**	**26,67±3,29**
	5	4: 5: 2	94,55±1,76	22,20±3,19
	6	4: 6: 2	94,70±1,63	24,73±3,05
	7	4: 5: 2	94,47±1,39	22,63±2,33
	8	4: 6: 2	94,55±1,32	25,33±2,17
Cancer	1	9: 2: 1	98,49±0,61	12,23±1,50

	2	9: 3: 1	98,97±0,38	15,80±1,73
	3	9: 2: 1	98,51±0,49	11,97±1,79
	4	9: 3: 1	98,72±0,60	16,23±2,49
	5	9: 2: 1	98,56±0,56	10,83±1,23
	6	9: 3: 1	98,81±0,54	15,90±2,31
	7	9: 2: 1	98,37±0,66	11,40±1,57
	8	9: 3: 1	**98,97±0,38**	**15,80±2,02**
Pima	1	8: 3: 1	77,33±2,36	13,53±2,39
	2	8: 4: 1	**78,61±1,88**	**18,60±3,04**
	3	8: 3: 1	76,96±1,67	13,73±1,91
	4	8: 4: 1	77,69±1,79	17,37±2,20
	5	8: 3: 1	77,15±1,70	13,27±2,46
	6	8: 4: 1	77,22±2,24	17,00±2,91
	7	8: 3: 1	76,75±1,96	13,47±1,76
	8	8: 4: 1	77,03±1,76	18,60±3,04

Tabla 6.3. Resultados preliminares de EDD con 3 parámetros.

Conjunto de datos	**Configuración**	**Topología**	**CCR_{gen}± D. Típica**	**Número_Conexiones± D. Típica**
Hypothyroid	1	29:3:4	95,27±0,77	31,13±6,38
	2	29:4:4	95,32±0,58	39,63±8,78
	3	29:5:4	**95,57±0,50**	**45,13±8,25**
	4	29:6:4	95,43±0,55	54,89±8,16
	5	29:3:4	95,17±0,29	30,29±6,40
	6	29:4:4	95,04±0,22	42,50±8,24
	7	29:5:4	95,04±0,26	49,67±9,39
	8	29:6:4	95,29±0,16	51,71±5,25
Waveform	1	40:3:2	82,44±1,59	37,30±5,19
	2	40:4:2	83,59±1,04	52,67±12,62
	3	40:5:2	84,35±1,53	57,71±18,61
	4	40:6:2	**85,11±1,48**	**70,13±13,02**
	5	40:3:2	82,45±0,84	35,17±5,85
	6	40:4:2	83,53±1,18	48,86±11,67

7	40:5:2	83,57±1,69	58,50±19,21
8	40:6:2	84,49±0,81	72,29±12,42

Tabla 6.4. Resultados preliminares de EDD con 2 parámetros.

Los resultados preliminares nos permitieron concluir que es muy interesante distribuir el número de nodos y el valor de α_2. Sin embargo, en lo que atañe al tercer parámetro distribuido, se observa que, al realizar el entrenamiento durante 0,8*gen generaciones, los resultados eran peores que con un entrenamiento de mayor duración. Por dicho motivo, en las sucesivas pruebas haciendo uso de EDD se optó por emplear las cuatro primeras configuraciones de EDD con 3 parámetros, que realizan una evolución mayor. Expresado en otros términos, se adaptó la versión inicial de la primera variante de EDD para llevar a cabo la distribución de 2 de los 3 parámetros de la primera versión, con lo que, dado que cada uno de dichos parámetros podía tomar dos valores diferentes, el número total de configuraciones es 4.

En el siguiente subapartado llevamos a cabo una comparación de los resultados de EDD, que acabamos de comentar, con los de otros clasificadores de la literatura.

6.2.1.1. Comparación con otros clasificadores

Los resultados preliminares fueron comparados con el método *Logistic Model Trees* (LMT) [Landwehr, Hall y Frank, 2005]. En dicho trabajo se hacía una exhaustiva experimentación con otros métodos muy potentes como MLogistic y C4.5 y se ponía de manifiesto la superioridad de LMT. A continuación, mostramos una comparación incluyendo otros métodos, entre los que destacan SVM [Vapnik, 1995], PART [Frank y Witten, 1998], MLP [Bishop, 1995] con un método de aprendizaje de Retropropagación (conocido como BP, del inglés *Back-Propagation*) y RBF

[Howlett y Jain, 2001]. Se han utilizado las implementaciones de WEKA de estos clasificadores.

Respecto a los parámetros, en BP se emplearon los siguientes: $\eta = 0,3$ para la tasa de aprendizaje (*learning rate*) y $\alpha = 0,2$ para el momento (*momentum*). Este último es un parámetro que suaviza el comportamiento oscilatorio. Los restantes algoritmos han sido ejecutados usando los valores por defecto de los parámetros, ya que presentan un comportamiento muy robusto y han sido los propuestos por los diferentes investigadores que han presentado y/o codificado los algoritmos. En referencia a EDD, se incluye únicamente el valor medio de la mejor de las configuraciones recién expuestas.

De los métodos estocásticos, como es el caso de MLP, RBF y EDD, se ha promediado el resultado de las 30 repeticiones realizadas con diferentes semillas. En la tabla 6.5 se exponen los resultados obtenidos con cada clasificador para cada conjunto de datos, así como el valor medio del rendimiento global de cada método. Los mejores resultados figuran destacados en negrita.

Conjunto de datos	**Clasificador**								
	C4.5	**1-NN**	**SVM**	**Mlogistic**	**LMT**	**PART**	**MLP**	**RBF**	**EDD**
Balance	83,33	77,56	88,46	90,38	93,59	85,26	93,78	88,27	**95,62**
Cancer	97,13	97,13	98,28	97,70	96,55	97,13	97,81	97,20	**98,97**
Hypothyroid	**99,15**	90,99	93,85	96,92	**99,15**	98,83	94,39	92,83	95,57
Pima	74,48	73,96	78,13	**81,77**	76,56	74,48	75,94	77,34	78,61
Waveform	74,80	68,96	86,24	85,84	86,64	76,88	84,85	**87,29**	85,11
Promedio	85,78	81,72	88,99	90,52	90,50	86,52	89,35	88,59	**96,72**

Tabla 6.5. Comparación de los resultados preliminares obtenidos con EDD frente a otros clasificadores.

Como podemos observar, en 2 de las 5 bases de datos el mejor resultado se logra con EDD, mientras que MLogistic y RBF lo consiguen en otro conjunto de datos. Finalmente, en *Hypothyroid* LMT y C4.5 presentan el mismo rendimiento. En media, EDD obtiene el mejor resultado con un CCR_{gen} de 96,72%, seguido de MLogistic y LMT con 90,52% y 90,50%, respectivamente. En las figuras 6.1, 6.2, 6.3, 6.4 y 6.5 mostramos para cada conjunto de datos los diagramas de cajas correspondientes a los resultados de la tabla 6.5.

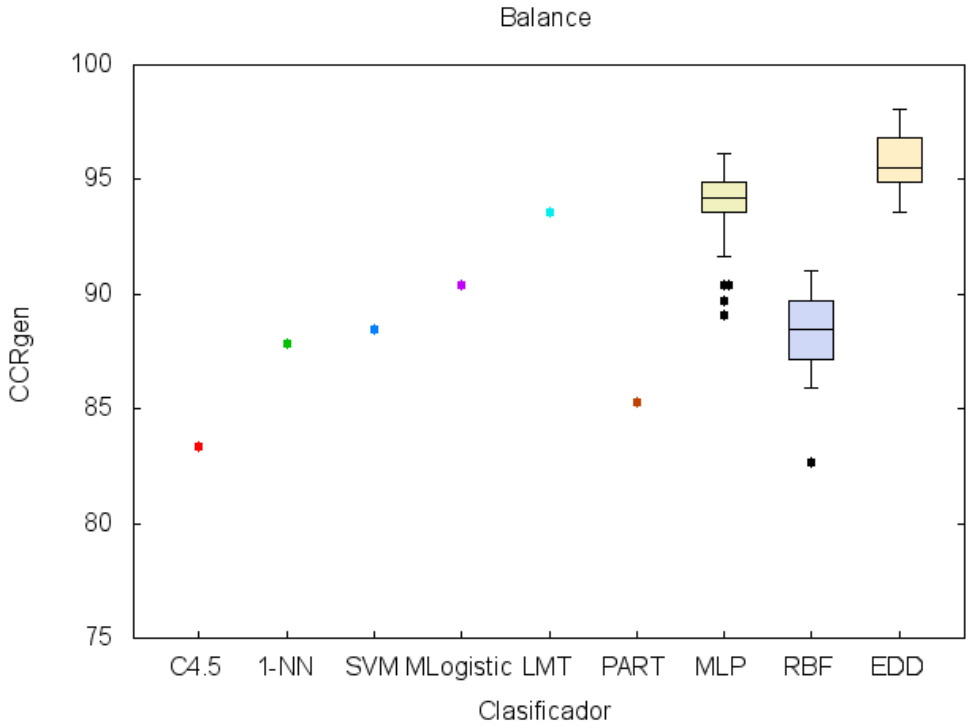


Figura 6.1. Diagrama de cajas comparativo de EDD con el conjunto de datos *Balance*.

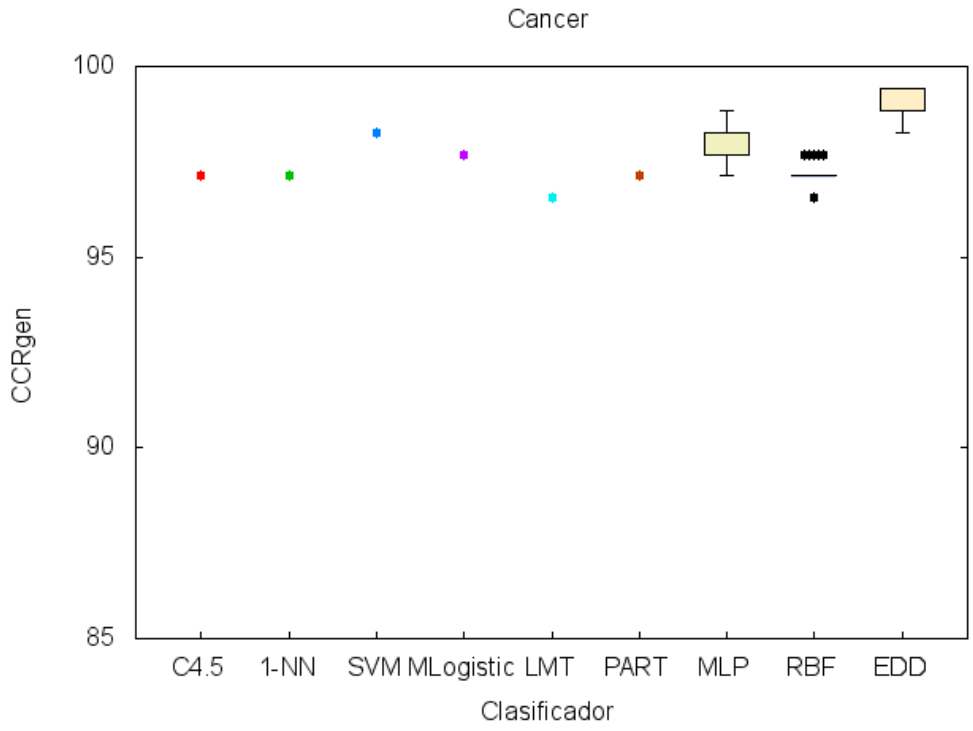


Figura 6.2. Diagrama de cajas comparativo de EDD con el conjunto de datos *Cancer*.

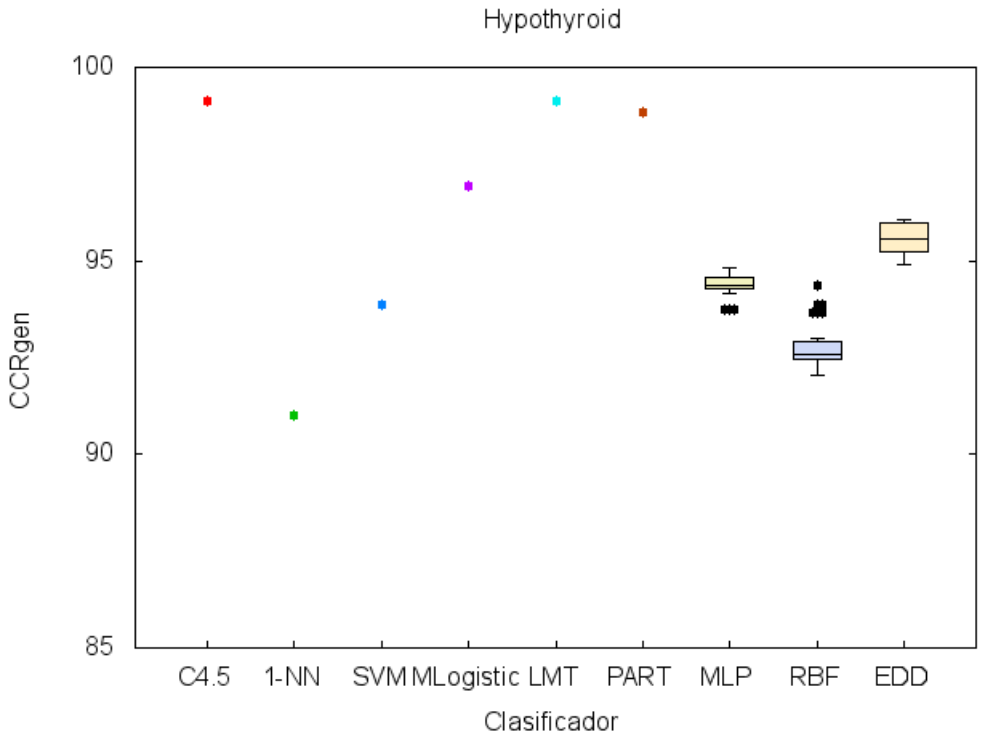


Figura 6.3. Diagrama de cajas comparativo de EDD con el conjunto de datos *Hypothyroid*.

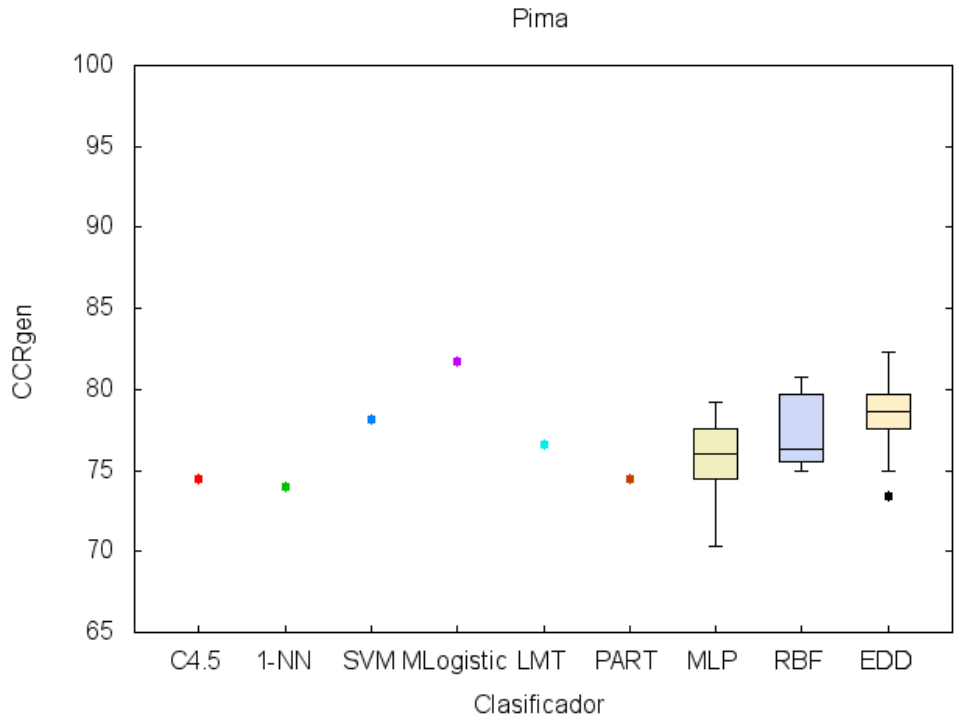


Figura 6.4. Diagrama de cajas comparativo de EDD con el conjunto de datos *Pima*.

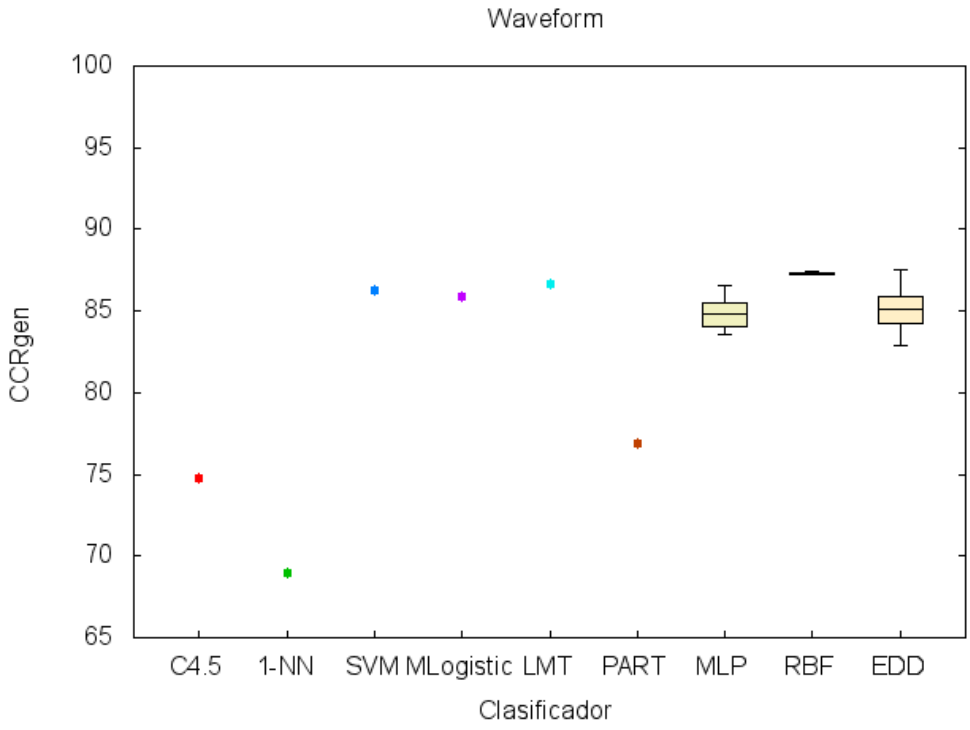


Figura 6.5. Diagrama de cajas comparativo de EDD con el conjunto de datos *Waveform*.

6.2.2. Resultados ampliados

A partir de las conclusiones obtenidas de los resultados preliminares del modelo EDD, se aumentó el número de conjuntos de datos con el fin de poder evaluar si los parámetros distribuidos entre las diferentes configuraciones eran apropiados. Como hemos comentado, el número de configuraciones se fijó en 4 a raíz de la experimentación preliminar. Los valores de los parámetros de la configuración base empleados en esta experimentación ampliada con EDD aparecen en la tabla 6.6.

Conjunto de datos	**Número máximo de neuronas (*neu*)**	**Número máximo de generaciones (*gen*)**	α_2
Australian	4	100	1
Balance	5	150	1
Breast	9	500	1
Breast-t	5	300	1
Cancer	2	100	1
Cardiotocography	6	300	1
Heart	6	500	1
Heart-c	3	300	1
Hepatitis	3	100	1
Horse	4	300	1
Hypothyroid	3	500	1
Ionos	4	500	1
Labor	6	300	1
Liver	4	300	1
Lymphography	6	500	1
Newthyroid	3	300	1
Parkinsons	6	500	1
Pima	3	120	1
Plates	6	500	1

Promoter	11	500	1
Waveform	3	500	1
Winequality-red	6	300	1
Yeast	11	500	1
BTX	5	500	1
Listeria	4	300	1

Tabla 6.6. Valores de los parámetros de las configuraciones base de EDD empleados en la experimentación ampliada.

Para la determinación de los parámetros en las bases de datos nuevas respecto a los resultados preliminares, se llevó a cabo un diseño experimental previo similar al comentado en el apartado 6.2.1. Los valores considerados para los diferentes tipos de problemas aparecen en la tabla 6.7, que presentan un mayor rango comparando con los mencionados anteriormente. El conjunto de datos *Promoter*, debido a su elevado número de entradas, 114, fue sometido también al diseño experimental correspondiente a los problemas de tamaño grande.

Tipo de problema según el tamaño	**Número máximo de neuronas (*neu*)**	**Número máximo de generaciones (*gen*)**
Pequeño	{2, 3, 4, 5, 6 y 7}	{100, 300 y 500}
Grande	{2, 3, 4, 5, 6, 7, 8, 9, 10, 11 y 12}	{300, 500 y 1000}

Tabla 6.7. Valores de los parámetros definidos en el diseño experimental previo de EDD.

Los resultados ampliados que se han obtenido con EDD se exponen en la tabla 6.8. Se indica para cada conjunto de datos la topología de la configuración base junto con los resultados medios de CCR_{gen} ejecutando 30 iteraciones de cada configuración.

Conjunto de datos	Topología Configuración Base	CCR_{gen}±D. Típica			
		Configuración			
		1	2	3	4
Australian	51: 4: 1	87,63±1,49	87,32±1,66	87,70±1,45	87,63±1,54
Balance	4: 5: 2	95,30±1,47	95,15±0,98	95,04±1,41	95,62±1,16
Breast	15: 9: 1	63,85±3,81	63,00±3,24	64,27±3,89	63,43±3,80
Breast-t	9: 5: 5	52,80±6,42	54,00±8,83	54,53±8,03	53,46±10,37
Cancer	9: 2: 1	98,49±0,61	98,97±0,38	98,51±0,49	98,72±0,60
Cardiotocography	31: 6: 2	80,93±2,76	79,94±3,70	81,25±3,17	81,04±3,79
Heart	13: 6: 1	75,93±2,40	75,83±3,27	76,23±2,48	76,03±3,50
Heart-c	26: 3: 1	82,18±2,36	82,36±2,98	82,32±3,08	81,97±3,05
Hepatitis	19: 3: 1	84,47±4,49	85,52±4,67	84,47±4,55	84,29±5,33
Horse	83: 4: 1	86,41±2,38	85,18±2,57	85,72±2,43	85,57±3,60
Hypothyroid	29: 3: 3	95,27±0,77	95,32±0,58	94,96±0,62	95,16±0,32
Ionos	34: 4: 1	92,31±2,23	92,50±2,28	91,85±2,48	93,06±1,87
Labor	29: 6: 1	84,04±11,04	83,57±10,29	84,28±10,67	82,85±13,98
Liver	6: 4: 1	73,87±2,21	73,56±1,97	72,20±3,22	73,64±3,23
Lymphography	38: 6: 3	75,49±6,98	76,12±6,95	76,48±6,70	76,85±7,56
Newthyroid	5: 3: 2	94,44±0,68	94,75±0,98	94,81±0,89	94,75±1,38
Parkinsons	22: 6: 1	79,66±5,01	78,16±4,77	78,98±4,05	79,32±4,76
Pima	8: 3: 1	77,33±2,36	78,61±1,88	76,96±1,67	77,69±1,79
Plates	27: 6: 6	52,61±5,09	51,18±5,35	52,01±3,37	52,82±4,03
Promoter	114: 11: 1	59,74±9,30	58,21±9,67	60,51±10,00	55,51±10,03
Waveform	40: 3: 2	81,43±2,10	82,78±0,64	82,05±1,64	84,32±1,73
Winequality-red	11: 6: 5	61,03±1,30	60,80±1,25	61,16±1,20	60,98±1,42
Yeast	8: 11: 9	59,18±1,17	59,62±1,27	58,50±1,74	59,18±1,83
BTX	3: 5: 6	79,04±6,32	77,61±7,09	78,73±5,26	77,14±4,90
Listeria	4: 4: 1	87,20±1,71	87,45±1,14	86,66±1,82	86,98±1,72

Tabla 6.8. Resultados ampliados de EDD.

6.2.2.1. Comparación con otros clasificadores

A continuación, en la tabla 6.9 comparamos para cada conjunto de datos el resultado medio obtenido con la mejor de las 4 configuraciones de EDD con otros clasificadores. Los mejores resultados de cada conjunto de datos se resaltan en negrita.

Conjunto de datos	**Clasificador**						
	C4.5	**1-NN**	**SVM**	**PART**	**MLP**	**RBF**	**EDD**
Australian	86,71	82,66	**88,44**	84,97	84,10	75,84	87,70
Balance	83,33	77,56	88,46	85,26	93,78	88,27	**95,62**
Breast	**70,42**	64,79	64,79	69,01	60,80	68,78	64,27
Breast-t	52,00	60,00	52,00	44,00	**63,20**	61,20	54,53
Cancer	97,13	97,13	98,28	97,13	97,81	97,20	**98,97**
Cardiotocography	82,71	76,32	**83,65**	82,52	80,75	81,80	81,25
Heart	70,59	73,53	76,47	73,53	74,85	**78,53**	76,23
Heart-c	75,00	76,32	82,89	80,26	84,82	**86,75**	82,36
Hepatitis	84,21	86,84	**89,47**	81,58	84,73	89,30	85,52
Horse	88,04	86,96	88,04	85,87	**88,51**	80,47	86,41
Hypothyroid	**99,15**	90,99	93,85	98,83	94,39	92,83	95,32
Ionos	92,05	90,91	88,64	**95,45**	89,12	92,46	93,06
Labor	85,71	71,43	78,57	**85,71**	69,52	71,67	84,28
Liver	68,60	61,63	58,14	61,63	65,65	57,17	**73,87**
Lymphography	75,68	83,78	**91,89**	75,68	86,58	70,99	76,85
Newthyroid	96,30	94,44	88,89	92,59	97,08	**98,27**	94,81
Parkinsons	71,43	77,55	75,51	75,51	77,62	70,27	**79,66**
Pima	74,48	73,96	78,13	74,48	75,94	77,34	**78,61**
Plates	39,05	49,17	57,02	46,69	53,50	**59,94**	52,82
Promoter	69,23	65,38	**88,46**	53,85	86,03	79,36	60,51
Waveform	74,80	68,96	86,24	76,88	84,85	**87,29**	84,32
Winequality-red	53,85	49,88	59,55	51,36	56,35	57,11	**61,16**
Yeast	54,84	48,39	55,91	56,72	**59,94**	58,31	59,62
BTX	80,95	76,19	61,90	80,95	54,12	**80,95**	79,04

Listeria	85,93	83,70	80,74	86,67	84,49	83,70	**87,45**
Promedio	86,32	83,18	84,59	85,82	84,30	79,77	**87,58**

Tabla 6.9. Comparación de resultados obtenidos con la experimentación ampliada de EDD frente a otros clasificadores.

Considerando el rendimiento global de cada clasificador, EDD alcanza el mejor resultado (87,58%), seguido por C4.5 y PART. Si analizamos los resultados de cada conjunto de datos de forma individual, EDD obtiene el mejor resultado en 7 conjuntos de datos, RBF presenta el mejor rendimiento en 6 bases de datos, secundado por SVM que gana en 5 ocasiones. Le siguen el clasificador MLP con 3 y, finalmente, con 2 se sitúan C4.5 y PART.

6.2.3. Resultados del modelo extendido

El modelo EDD fue ampliado para considerar como nodos en la capa oculta unidades de tipo sigmoide, lo que originó el modelo denominado EDDSig. En EDDSig se han considerado sólo 2 configuraciones, debido a que la intención es emplear 2 valores posibles de un mismo parámetro para poder ser comparado con una configuración de TSEASig. Como es lógico pensar, el número de nodos en la capa oculta debe influir notablemente, por lo que fue el parámetro que se decidió distribuir. En la tabla 6.10 se describen las configuraciones de EDDSig, siendo la primera de ellas, como en EDD, la configuración base. Las celdas cuyo valor aparece en negrita, indican que el valor concreto es específico para cada conjunto de datos. El valor de α_2 no depende de la configuración, sino de los parámetros de AE básico, considerando en este caso unidades sigmoide en la capa oculta.

Configuración	Número máximo de neuronas (*neu*)	Número máximo de generaciones (*gen*)	α_2
1S	neu	gen	α_2
2S	neu + 1	gen	α_2

Tabla 6.10. Configuraciones de EDDSig.

La tabla 6.11 especifica los valores de los parámetros de AE básico aplicado a unidades sigmoide. La principal diferencia respecto a las unidades producto es la normalización de los datos de entrada en el intervalo [0,1, 0,9], debido a que con unidades sigmoide, valores próximos a 0 no son un problema, dado que no modifican tan considerablemente los órdenes de magnitud de las variables de entrada, que ahora son ponderadas y no elevadas a exponentes.

Parámetro / Característica	Valor
Tamaño de la población (tam_pob)	1000
$gen-\sin-mejorar$	20
Intervalo para los exponentes w_{ji}	[-5, 5]
Intervalo para los coeficientes β_j^l	[-10, 10]
Valores iniciales de α_1 y α_2	0,5 para ambos
Normalización de los datos de entrada	[0,1, 0,9]
Intervalo de nodos en las operaciones añadir nodo y eliminar nodo	{1, 2}

Tabla 6.11. Parámetros / Características del AE básico aplicado a unidades sigmoide.

Los valores de los parámetros de la configuración base de EDDSig, 1S, que son asignados específicamente para cada conjunto de datos, se ilustran en la tabla 6.12. Para la determinación de los valores de los parámetros se realizó un estudio experimental previo sobre el conjunto de entrenamiento aplicando un *5-fold* con 5 repeticiones con algunas

combinaciones de parámetros. En concreto, se definieron los posibles valores para el número de neuronas, tomando como referencia el valor empleado en EDD, el anterior y el siguiente. Para el número de generaciones se establecieron los valores {100, 300, 500 y 1000}. A efectos prácticos, en la gran mayoría de los problemas, los modelos de red obtenidos con una evolución de 500 generaciones sobreentrenan, no siendo necesario, por tanto, experimentar con 1000 generaciones. Al conjunto de datos *HeartY* se le aplicó el diseño experimental previo comentado en el apartado 6.2.2, debido a que en EDD no se empleó el citado problema.

Conjunto de datos	**Número máximo de neuronas (*neu*)**	**Número máximo de generaciones (*gen*)**
Australian	4	100
Balance	4	300
Cancer	2	100
Heart-c	3	300
HeartY	4	100
Hepatitis	3	100
Horse	4	300
Hypothyroid	3	500
Newthyroid	3	300
Pima	3	100
Waveform	3	500
Yeast	11	1000
BTX	5	500
Listeria	4	300

Tabla 6.12. Valores de los parámetros de las configuraciones base de EDDSig.

Los resultados obtenidos con el modelo EDDSig se detallan en la tabla 6.13. Para cada conjunto de datos y configuración, aparecen los valores medios y las desviaciones típicas del CCR_{gen} junto con la topología

asociada. Se destacan en negrita los mejores valores de CCR_{gen} obtenidos para cada conjunto de datos.

Conjunto de datos	**Configuración**			
	1S		**2S**	
	CCR_{gen}±D. Típica	**Topología**	**CCR_{gen}±D. Típica**	**Topología**
Australian	87,12±1,45	51: 4: 1	**87,46±1,18**	51: 5: 1
Balance	94,14±1,87	4: 4: 2	**95,08±1,27**	4: 5: 2
Cancer	98,39±0,74	9: 2: 1	**98,69±0,54**	9: 3: 1
Heart-c	82,89±2,46	26: 3: 1	**83,02±2,59**	26: 4: 1
HeartY	**83,28±2,92**	13: 4: 1	83,18±2,61	13: 5: 1
Hepatitis	**86,22±4,77**	19: 3: 1	85,78±4,12	19: 4: 1
Horse	85,21±2,98	83: 4: 1	**85,50±2,69**	83: 5: 1
Hypothyroid	94,69±0,39	29: 3: 3	**94,94±0,35**	29: 4: 3
Newthyroid	94,07±0,75	5: 3: 2	**94,25±1,01**	5: 4: 2
Pima	77,62±1,62	8: 3: 1	**78,14±1,60**	8: 4: 1
Waveform	85,35±1,45	40: 3: 2	**85,96±1,22**	40: 4: 2
Yeast	58,96±1,27	8: 11: 9	**59,24±1,2**	8: 12: 9
BTX	72,85±7,41	3: 5: 6	**73,01±4,89**	3: 6: 6
Listeria	85,06±1,13	4: 4: 1	**85,43±0,97**	4: 5: 1

Tabla 6.13. Resultados de EDDSig.

Seguidamente, en la figura 6.6 mostramos una comparación de EDDSig con EDD. Para hacer justa tal comparación sólo se han considerado los mejores resultados de las dos primeras configuraciones de EDD. El valor medio de CCR_{gen} alcanzado por EDD es ligeramente superior al de EDDSig. De manera individual, EDDSig es superior en *Waveform*, *Heart-c* y *Hepatitis*. La mayor mejora de EDDSig respecto a los resultados de EDD se produce en *Waveform*, con un aumento de CCR_{gen} de más de 3 puntos; en *Heart-c* y *Hepatitis* los incrementos están cercanos a 3/4

de punto. Por el contrario, las diferencias mayores a favor de EDD se hallan en los casos de *BTX* y *Listeria*, con algo más de 6 y 2 puntos, respectivamente. A partir de lo anterior, concluimos que la diferencia obtenida sobre *BTX* por EDDSig desequilibra la balanza a su favor.

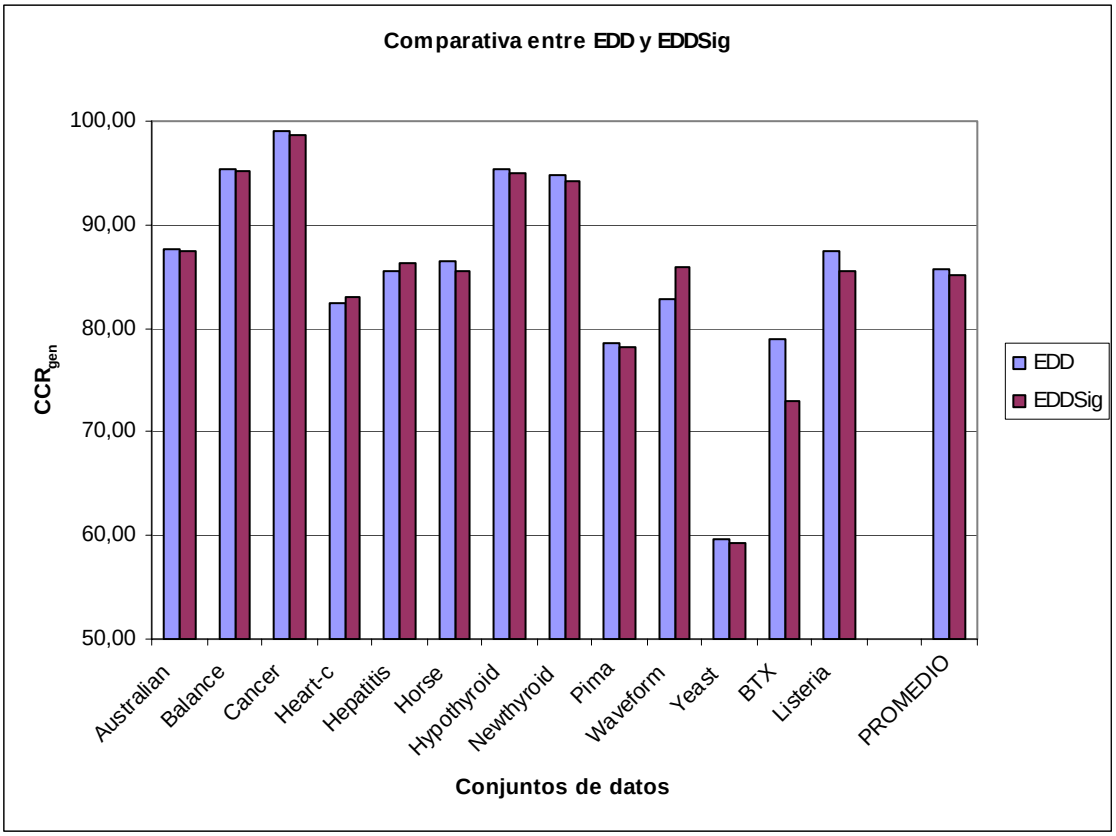


Figura 6.6. Comparativa entre EDD y EDDSig.

En el apartado 6.3.2.1 mostramos una visión global de los resultados obtenidos por EDDSig, junto con los resultados de otros clasificadores y el método TSEASig, cuyos resultados se presentan en el apartado 6.3.2.

6.3. SEGUNDO MODELO: TSEA

En el modelo TSEA se han definido 2 configuraciones, cuya única diferencia radica en el valor del parámetro α_2. Ambas configuraciones, por consiguiente, presentan una topología común. La tabla 6.14 indica los valores de los parámetros de la configuración base para cada conjunto de datos, que han sido definidos tomando como referencia los valores de los parámetros de EDD. Dado que el número máximo de neuronas de las dos poblaciones de TSEA es diferente, se han considerado el valor base de EDD y el valor siguiente. El propósito de TSEA es evaluar si con una configuración podemos obtener resultados mejores que con 2 configuraciones de EDD. Para los conjuntos de datos nuevos que no se han usado en los modelos anteriores, hemos seguido un diseño experimental como el especificado en el apartado 6.2.2.

Conjunto de datos	**Número máximo de neuronas en cada población (*neu* y *neu+1*)**	**Número máximo de generaciones en cada población (0,1**gen*) (Fase 1)**	α_2
Appendicitis	4 y 5	30	1
Australian	4 y 5	10	1
Balance	5 y 6	15	1
Breast	9 y 10	50	1
Breast-t	5 y 6	30	1
Cancer	2 y 3	10	1
Cardiotocography	6 y 7	30	1
Heart	6 y 7	50	1
Heart-c	3 y 4	30	1
HeartY	4 y 5	10	1
Hepatitis	3 y 4	30	1
Horse	4 y 5	30	1
Hypothyroid	3 y 4	50	1

Ionos	4 y 5	50	1
Labor	6 y 7	30	1
Led24	8 y 9	50	1
Liver	4 y 5	30	1
Lymphography	6 y 7	50	1
Newthyroid	3 y 4	30	1
Parkinsons	6 y 7	30	1
Pima	4 y 5	15	1
Plates	6 y 7	50	1
Promoter	11 y 12	50	1
SPECTF	6 y 7	50	1
Vowel	6 y 7	100	1
Waveform	3 y 4	50	1
Winequality-red	6 y 7	30	1
Yeast	11 y 12	100	1
BTX	5 y 6	50	1
Listeria	4 y 5	30	1

Tabla 6.14. Valores de los parámetros de las configuraciones base de TSEA.

La tabla 6.15 muestra, para cada conjunto de datos, la topología común así como los resultados medios obtenidos sobre el conjunto de generalización junto con la desviación típica para cada configuración. El mejor resultado de cada base de datos aparece destacado en negrita.

Conjunto de datos	**Topología común**	**CCR_{gen}±D. Típica**	
		Configuración	
		1*	**2***
Appendicitis	7: [4, 5]: 1	**81,66±4,24**	80,51±2,84
Australian	51: [4, 5]: 1	88,11±1,56	**88,68±1,10**
Balance	4: [5, 6]: 2	**96,20±1,06**	95,62±1,12
Breast	15: [9, 10]: 1	**65,96±2,89**	62,76±3,08
Breast-t	9: [5, 6]: 5	54,53±7,89	**55,33±9,16**
Cancer	9: [2, 3]: 1	98,74±0,61	**98,98±0,54**

Cardiotocography	31: [6, 7]: 2	**81,69±3,56**	81,55±2,90
Heart	13: [6, 7]: 1	76,62±2,33	**77,45±3,09**
Heart-c	26: [3, 4]: 1	**83,68±2,57**	83,64±2,33
HeartY	13: [4, 5]: 1	**84,01±3,05**	82,97±2,61
Hepatitis	19: [3, 4]: 1	82,10±4,44	**87,01±3,78**
Horse	83: [4, 5]: 1	85,50±2,97	**86,59±2,38**
Hypothyroid	29: [3, 4]: 3	**95,37±0,40**	94,94±0,25
Ionos	34: [4, 5]: 1	91,51±1,88	**93,22±1,92**
Labor	29: [6, 7]: 1	85,24±8,78	**86,90±5,96**
Led24	24: [8, 9]: 9	50,29±6,59	**51,03±5,58**
Liver	6: [4, 5]: 1	**74,61±2,00**	72,36±1,51
Lymphography	38: [6, 7]: 3	**79,37±4,73**	78,73±4,79
Newthyroid	5: [3, 4]: 2	**94,88±0,93**	94,19±1,92
Parkinsons	22: [6, 7]: 1	73,94±2,43	78,09±3,51
Pima	8: [4, 5]: 1	78,38±1,59	**79,21±1,53**
Plates	27: [6, 7]: 6	50,74±4,24	**51,46±3,03**
Promoter	114: [11, 12]: 1	65,76±8,99	**68,20±9,52**
SPECTF	44: [6, 7]: 1	60,17±4,15	**61,56±4,97**
Vowel	11: [6, 7]: 10	45,04±2,93	**47,18±4,03**
Waveform	40: [3, 4]: 2	**84,46±0,92**	82,01±1,48
Winequality-red	11: [6, 7]: 5	60,95±1,58	**61,11±1,02**
Yeast	8: [11, 12]: 9	60,05±1,21	**60,16±1,10**
BTX	3: [5, 6]: 6	79,68±7,39	**81,11±6,55**
Listeria	4: [4, 5]: 1	86,54±1,67	**87,68±1,06**

Tabla 6.15. Resultados de TSEA.

6.3.1. Comparación con otros clasificadores

La tabla 6.16 expone una comparación entre el valor medio logrado por la mejor configuración de TSEA frente a otros clasificadores, C4.5, 1-NN, SVM, PART, MLP y RBF, usando las implementaciones de WEKA con los mismos valores de los parámetros y número de repeticiones que en las comparaciones previas. El mejor resultado de cada fila,

correspondiente a una base de datos aplicando cada clasificador o bien al promedio global de los clasificadores con todos los conjuntos de datos, figura resaltado en negrita.

Conjunto de datos	**Clasificador**						
	C4.5	**1-NN**	**SVM**	**PART**	**MLP**	**RBF**	**TSEA**
Appendicitis	73,08	69,23	**84,62**	73,08	76,92	74,67	81,66
Australian	**86,71**	82,66	88,44	84,97	84,10	75,84	**88,68**
Balance	83,33	77,56	88,46	85,26	93,78	88,27	**96,20**
Breast	**70,42**	64,79	64,79	69,01	60,80	68,78	65,96
Breast-t	52,00	60,00	52,00	44,00	**63,20**	61,20	55,33
Cancer	97,13	97,13	98,28	97,13	97,81	97,20	**98,98**
Cardiotocography	82,71	76,32	**83,65**	82,52	80,75	81,80	81,69
Heart	70,59	73,53	76,47	73,53	74,85	**78,53**	77,45
Heart-c	75,00	76,32	82,89	80,26	84,82	**86,75**	83,68
HeartY	86,76	79,41	**86,76**	83,82	84,29	83,79	84,01
Hepatitis	84,21	86,84	**89,47**	81,58	84,73	87,01	87,01
Horse	88,04	86,96	88,04	85,87	**88,51**	80,47	86,59
Hypothyroid	**99,15**	90,99	93,85	98,83	94,39	92,83	95,37
Ionos	92,05	90,91	88,64	**95,45**	89,12	92,46	93,22
Labor	85,71	71,43	78,57	85,71	69,52	71,67	**86,90**
Led24	**65,67**	39,43	58,97	55,80	57,48	55,14	51,03
Liver	68,60	61,63	58,14	61,63	65,65	57,17	**74,61**
Lymphography	75,68	83,78	**91,89**	75,68	86,58	70,99	79,37
Newthyroid	96,30	94,44	88,89	92,59	97,08	**98,27**	94,88
Parkinsons	71,43	77,55	75,51	75,51	77,62	70,27	**78,09**
Pima	74,48	73,96	78,13	74,48	75,94	77,34	**79,21**
Plates	39,05	49,17	57,02	46,69	53,50	**59,94**	51,46
Promoter	69,23	65,38	**88,46**	53,85	86,03	79,36	68,20
SPECTF	67,91	61,50	72,19	70,59	71,28	**76,19**	61,56
Vowel	39,39	**48,48**	45,45	38,53	45,87	47,25	47,18
Waveform	74,80	68,96	86,24	76,88	84,85	**87,29**	84,46
Winequality-red	53,85	49,88	59,55	51,36	56,35	57,11	**61,11**
Yeast	54,84	48,39	55,91	56,72	59,94	58,31	**60,16**
BTX	80,95	76,19	61,90	80,95	54,12	80,95	**81,11**

Listeria	85,93	83,70	80,74	86,67	84,49	83,70	**87,68**
Promedio	74,83	72,22	76,80	73,97	76,15	76,09	**77,39**

Tabla 6.16. Comparación de resultados obtenidos con TSEA frente a otros clasificadores.

TSEA obtiene el mejor valor medio (77,39%) considerando el resultado global de cada clasificador con todas las bases de datos, seguido de SVM (76,80%) y MLP (76,09%). Analizando cada conjunto de datos de manera individual, TSEA logra el mejor resultado en 11 de ellos. Le siguen los clasificadores SVM y RBF con 6 victorias cada uno. C4.5 es el 4º mejor clasificador con un resultado superior a los demás algoritmos en 3 ocasiones.

El clasificador TSEA obtiene los mejores resultados con importantes diferencias frente a los demás y, en especial, con los clasificadores que presentan el rendimiento más próximo en *Liver* -el 74,61% supera en más de 4 puntos a C4.5- y en *Balance*, que alcanza el 96,20% frente al 93,78% de MLP. SVM logra resultados destacados en *Lymphography* -con más de 5 puntos de ventaja sobre MLP-, *Appendicitis* y *Promoter*. Respecto al rendimiento de RBF, destaca el resultado de *SPECTF* y *Plates*. MLP consigue en *Breast-t* un resultado excelente y resultados aceptables en los restantes conjuntos de datos que lo convierten, en media, en el tercer mejor clasificador con la batería de pruebas formada por los 30 conjuntos de datos de la tabla 6.16.

6.3.2. Comparación con EDD

En este apartado analizamos conjuntamente los resultados de TSEA y EDD para evaluar el rendimiento, así como el número medio de evaluaciones requerido por cada uno.

La tabla 6.17 expone el valor medio del CCR_{gen} de la mejor configuración de EDD y TSEA, obtenido en 30 iteraciones para cada conjunto de datos y el promedio de todos ellos. Para cada conjunto de datos, el mejor valor aparece destacado en negrita. Desde el punto de vista de la generalización, el rendimiento de TSEA es superior a EDD. La media del primero es mayor que la del segundo en casi un punto. TSEA tiene un mejor comportamiento en 22 de los 25 conjuntos de datos. En *Promoter* aventaja en más de 7,5 puntos al resultado obtenido por EDD. Mejoras de algo más de 2,5 puntos se producen en *Lymphography* y *Labor*. TSEA funciona mejor con el problema real *BTX*. Por el contrario, de los 3 conjuntos de datos donde EDD es mejor, la mayor diferencia se sitúa en el caso de *Parkinsons* con 1,57 puntos de diferencia.

Conjunto de datos	**Clasificador**	
	EDD	**TSEA**
Australian	87,70	**88,68**
Balance	95,62	**96,20**
Breast	64,27	**65,96**
Breast-t	54,53	**55,33**
Cancer	98,97	**98,98**
Cardiotocography	81,25	**81,69**
Heart	76,23	**77,45**
Heart-c	82,36	**83,68**
Hepatitis	85,52	**87,01**
Horse	86,41	**86,59**
Hypothyroid	95,32	**95,37**
Ionos	93,06	**93,22**
Labor	84,28	**86,90**
Liver	73,87	**74,61**
Lymphography	76,85	**79,37**
Newthyroid	94,81	**94,88**

Parkinsons	**79,66**	78,09
Pima	78,61	**79,21**
Plates	**52,82**	51,46
Promoter	60,51	68,20
Waveform	84,32	84,46
Winequality-red	**61,16**	61,11
Yeast	59,62	**60,16**
BTX	79,04	**81,11**
Listeria	87,45	**87,68**
Promedio	78,97	79,90

Tabla 6.17. Resultados obtenidos con TSEA y EDD.

Pasamos ahora a obtener el número de evaluaciones por iteración de TSEA y EDD. El modelo TSEA presenta dos fases: a) en la primera de ellas tenemos dos tipos de individuos, cada uno en una población, que evolucionan durante 0,1*gen generaciones y b) en la segunda fase una única población que sigue el proceso evolutivo completo, salvo la fase de inicialización. Teniendo en cuenta el pseudocódigo reflejado en la figura 4.4 el número medio de evaluaciones de una iteración de TSEA con una configuración viene dado por:

$$evaluciones(TSEA) = (tam\_pob * 10 + 0,9 * tam\_pob * 0,1 * gen) * 2 + 0,9 * tam\_pob * gen \quad (6.1)$$

El modelo EDD sólo consta de una única fase en la que tenemos individuos que tienen el mismo número máximo de neuronas en la capa oculta. El número medio de evaluaciones por iteración, considerando el proceso evolutivo completo, representado en la figura 4.1, y usando una configuración de EDD, es el siguiente:

$$evaluaciones(EDD) = tam\_pob * 10 + 0,9 * tam\_pob * gen \quad (6.2)$$

La experimentación de una configuración de TSEA se puede considerar equivalente a 2 configuraciones de EDD. En la segunda fase de

TSEA, en la misma población tenemos individuos un poco evolucionados con *neu* y *neu+1* nodos en la capa oculta sin necesidad de duplicar el tamaño de la población. La tabla 6.18 muestra los valores numéricos medios del número de evaluaciones por iteración para cada conjunto de datos y el promedio, referentes a una configuración de TSEA y a sus 2 configuraciones equivalentes de EDD, junto con la carga computacional requerida por TSEA comparando con EDD.

El número de generaciones que se han empleado en los cálculos son los que se han especificado en las secciones relativas a los resultados de TSEA y EDD y corresponden a los valores con los que se han obtenido los resultados de CCR_{gen} de la tabla 6.17. La carga computacional de TSEA frente a EDD oscila, como norma general, entre 0,61 y 0,64. Encontramos alguna excepción, como el caso *Hepatitis* con un valor de 1,72, debido a que el número de generaciones con TSEA es 300, frente a 100 de EDD. Para la batería de pruebas completa, el promedio es de 0,66. El porcentaje de la mejora de la eficiencia de TSEA frente a EDD viene dado por:

$$\%Mejora\_Eficiencia(TSEA) = (1 - Carga\_computacional(TSEA)) * 100 \qquad (6.3)$$

A partir de la ecuación (6.3), se obtiene que el porcentaje de la mejora de la eficiencia de TSEA en promedio es del 34%.

Conjunto de datos	EDD	TSEA	Carga_computacional (TSEA)
Australian	200000	128000	0,64
Balance	290000	182000	0,63
Breast	920000	560000	0,61
Breast-t	560000	344000	0,61
Cancer	200000	128000	0,64
Cardiotocography	560000	344000	0,61
Heart	920000	560000	0,61

Heart-c	560000	344000	0,61
Hepatitis	200000	344000	1,72
Horse	560000	344000	0,61
Hypothyroid	920000	560000	0,61
Ionos	920000	560000	0,61
Labor	560000	344000	0,61
Liver	560000	344000	0,61
Lymphography	920000	560000	0,61
Newthyroid	560000	344000	0,61
Parkinsons	920000	344000	0,37
Pima	236000	182000	0,77
Plates	920000	560000	0,61
Promoter	920000	560000	0,61
Waveform	920000	560000	0,61
Winequality-red	560000	344000	0,61
Yeast	1820000	1100000	0,60
BTX	920000	560000	0,61
Listeria	560000	344000	0,61
Promedio	687440	421760	0,66

Tabla 6.18. Número medio de evaluaciones por iteración con EDD y TSEA y carga computacional de TSEA.

Otra ventaja adicional de TSEA, además de requerir aproximadamente un 34% menos de evoluciones por iteración, radica en los resultados de CCR_{gen} son superiores. Por tanto, TSEA obtiene resultados más precisos y es un 34% más eficiente.

6.3.3. Resultados del modelo extendido

Una extensión del modelo TSEA fue propuesta, para permitir emplear unidades de tipo sigmoide en lugar de tipo producto en la capa oculta. Al nuevo modelo le otorgamos el nombre de TSEASig. Para ser coherentes con la propuesta de EDDSig, que sólo consideraba 2

configuraciones, en TSEASig hemos definido una única configuración, con la intención de compararla con las 2 configuraciones de aquel modelo. La tabla 6.19 especifica la configuración de TSEASig. Como es habitual, las celdas cuyo valor aparece en negrita indican que el valor concreto es específico para cada conjunto de datos. Por el contrario, el valor concreto de α_2 depende de los parámetros del AE básico aplicado a unidades sigmoide, que han sido descritos en la tabla 6.11, a partir de los cuales se definen los parámetros de TSEASig. Para el modelo TSEASig, el parámetro α_2 toma el valor 0,5.

Configuración	**Número máximo de neuronas en cada población (*neu*)**	**Tamaño de cada población**	**Número máximo de generaciones en cada población (*gen*) (Fase 1)**	α_2
1S*	**neu** y **neu**+1	1000	0,1***gen**	α_2

Tabla 6.19. Configuración de TSEASig.

Los valores concretos de los parámetros de la configuración de TSEASig para cada conjunto de datos se indican en la tabla 6.20. Hemos considerado los valores de los parámetros de EDDSig, tomando para el número máximo de neuronas el valor siguiente al definido anteriormente como valor adicional, debido a que en TSEASig el número máximo de nodos es un intervalo formado por dos valores consecutivos.

Conjunto de datos	**Número máximo de neuronas en cada población (*neu* y *neu+1*)**	**Número máximo de generaciones en cada población (0,1**gen*) (Fase 1)**
Australian	4 y 5	10
Balance	4 y 5	30
Cancer	2 y 3	10
Heart-c	3 y 4	30
HeartY	4 y 5	10
Hepatitis	3 y 4	10
Horse	4 y 5	30

Hypothyroid	3 y 4	50
Newthyroid	3 y 4	30
Pima	3 y 4	10
Waveform	3 y 4	50
Yeast	11 y 12	100
BTX	5 y 6	50
Listeria	4 y 5	30

Tabla 6.20. Valores de los parámetros de la configuración TSEASig.

Los resultados obtenidos con el modelo TSEASig aparecen reflejados en la tabla 6.21. Para cada conjunto de datos se muestra la topología y el resultado medio junto con la desviación típica, obtenidos sobre el conjunto de generalización.

Conjunto de datos	**Configuración 1S***	
	Topología	**CCR_{gen}±D. Típica**
Australian	51: [4, 5]: 1	86,55±1,43
Balance	4: [4, 5]: 2	93,93±2,21
Cancer	9: [2, 3]: 1	98,31±0,69
Heart-c	26: [3, 4]: 1	83,85±2,65
HeartY	13: [4, 5]: 1	83,87±2,66
Hepatitis	19: [3, 4]: 1	86,75±3,41
Horse	83: [4, 5]: 1	88,04±2,19
Hypothyroid	29: [3, 4]: 3	95,15±0,18
Newthyroid	5: [3, 4]: 2	94,38±0,59
Pima	8: [3, 4]: 1	78,76±1,39
Waveform	40: [3, 4]: 2	86,58±1,18
Yeast	8: [11, 12]: 9	60,33±0,61
BTX	3: [5, 6]: 6	76,34±5,65
Listeria	4: [4, 5]: 1	85,75±0,66

Tabla 6.21. Resultados de TSEASig.

En la figura 6.7 mostramos una comparación gráfica entre las metodologías TSEA y TSEASig, en la que se han considerado las

configuraciones 1* y 1S* de cada una de ellas, para que exista igualdad. La diferencia entre ambas metodologías radica en el tipo de unidad que emplean en la capa oculta y la similitud se encuentra en la utilización de un AE en dos fases para el entrenamiento de los parámetros de la red neuronal.

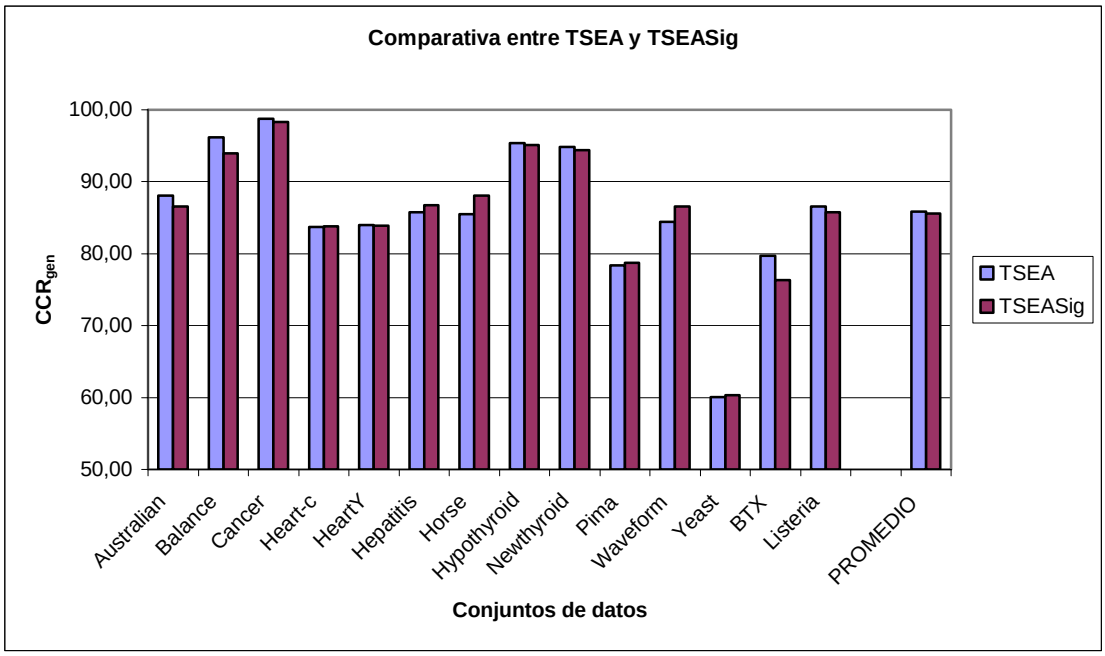


Figura 6.7. Comparativa entre TSEASig y TSEA.

Como podemos observar, el promedio de CCR_{gen} de TSEA es ligeramente superior al de TSEASig. Considerando los conjuntos de datos de manera individual, cabe destacar que, por ejemplo, en *Horse*, *Waveform*, *Hepatitis* y *Pima*, TSEASig es mejor. Por otra parte, TSEA es superior en los dos problemas reales, *BTX* y *Listeria*, y en otros del UCI como *Australian* y *Balance*. En el caso de conjuntos de datos con un número de entradas superior a 40, no hay un claro dominio de ninguna de las dos metodologías. En el típico problema difícil de clasificación de *Pima*, TSEASig mejora ligeramente el CCR_{gen} sobre TSEA.

6.3.3.1. Comparación con otros clasificadores

En la tabla 6.22 se expone una comparativa de TSEASig frente a otros clasificadores, incluyendo el método EDDSig presentado anteriormente. Para cada conjunto de datos se encuentra destacado en negrita el mejor valor obtenido. No hemos incluido EDD, debido a que TSEA, que es su alternativa mejorada, es superior, como hemos comentado en el apartado 6.3.2. Los algoritmos implementados en WEKA, como son C4.5, 1-NN, SVM, PART, MLP y RBF, han sido ejecutados con los mismos parámetros y repeticiones que hemos usado en toda la experimentación y han sido detallados anteriormente.

Conjunto de datos	**Clasificador**								
	C4.5	**1-NN**	**SVM**	**PART**	**MLP**	**RBF**	**TSEA**	**TSEASig**	**EDDSig**
Australian	86,71	82,66	88,44	84,97	84,10	75,84	**88,68**	86,55	87,46
Balance	83,33	77,56	88,46	85,26	93,78	88,27	**96,20**	93,93	95,08
Cancer	97,13	97,13	98,28	97,13	97,81	97,20	**98,98**	98,31	98,69
Heart-c	75,00	76,32	82,89	80,26	84,82	**86,75**	83,68	83,85	83,02
HeartY	**86,76**	79,41	**86,76**	83,82	84,29	83,79	84,01	83,87	83,28
Hepatitis	84,21	86,84	**89,47**	81,58	84,73	89,30	85,79	86,75	86,22
Horse	88,04	86,96	88,04	85,87	**88,51**	80,47	86,59	88,04	85,50
Hypothyroid	**99,15**	90,99	93,85	98,83	94,39	92,83	95,37	95,15	94,94
Newthyroid	96,30	94,44	88,89	92,59	97,08	**98,27**	94,88	94,38	94,25
Pima	74,48	73,96	78,13	74,48	75,94	77,34	**79,21**	78,76	78,14
Waveform	74,80	68,96	86,24	76,88	84,85	**87,29**	84,46	86,58	85,96
Yeast	54,84	48,39	55,91	56,72	59,94	58,31	60,16	**60,33**	59,24
BTX	80,95	76,19	61,90	80,95	54,12	80,95	**81,11**	76,34	73,01
Listeria	85,93	83,70	80,74	86,67	84,49	83,70	**87,68**	85,75	85,43
Promedio	83,40	80,25	83,43	83,29	83,49	84,31	86,20	85,61	85,02

Tabla 6.22. Comparación de los resultados obtenidos con TSEASig frente a otros clasificadores.

El mejor promedio de CCR_{gen} se obtiene con TSEA (86,20%), seguido de TSEASig (85,61%) y EDDSig (85,02%). A continuación, se sitúa RBF con un 84,31%, secundado por C4.5 (83,40%). TSEA, como se puede observar, obtiene el mejor resultado en 5 ocasiones, seguido de RBF con 3; tanto C4.5 como SVM tienen mejor comportamiento 2 veces y, finalmente, MLP y TSEASig, una vez. En el problema real de *Listeria* el método TSEA consigue el mejor resultado, con una mejora de casi 1 punto sobre el clasificador que obtiene el valor más cercano, PART. TSEASig supera ligeramente a TSEA en el problema *Yeast*, mientras que en el caso de *Waveform* se acerca al mejor valor, logrado por RBF.

6.3.3.2. Comparación con TSEA y EDDSig

TSEASig presenta como particularidad que emplea unidades de tipo sigmoide. Para evaluar la utilidad de la propuesta, lo comparamos con los modelos más parecidos como son EDDSig y TSEA. EDDSig también hace uso de unidades sigmoide, aunque utiliza un AE para el entrenamiento que sólo tiene una población. TSEA emplea un AE con dos fases, contando la primera de ellas con dos poblaciones.

Siguiendo las recomendaciones apuntadas por Janez Demšar [Demšar, 2006], realizamos tests estadísticos no paramétricos. En nuestro caso, permiten determinar la significación estadística de las diferencias en ranking observadas para cada filtro con todos los conjuntos de datos. Existen dos métodos: test de Friedman [Friedman, 1937] y test de Iman-Davenport [Iman y Davenport, 1980].

El primer test es equivalente al modelo de ANOVA (análisis de varianza, del inglés *ANalysis Of VAriance*) con medidas repetidas [Miller, 1981] y se basa en el estadístico χ_F^2 de Friedman. La hipótesis nula afirma

que todos los algoritmos tienen un comportamiento similar en media de rango, por lo que un rechazo de ésta implica la existencia de diferencias significativas.

El último test [Nisbet, Elder y Miner, 2009] es una variante del primero y se basa en F_F que es un estadístico mejor, derivado de χ_F^2, y no es conservador. F_F sigue una distribución F de Fisher con $(k-1)$ y $(k-1)*(N-1)$ grados de libertad con k algoritmos y N conjuntos de datos, que denotaremos con $F\left((k-1),\ (k-1)*(N-1)\right)$.

En esta Tesis aplicamos el test de Iman-Davenport. Si la hipótesis nula es rechazada, podemos proceder con un test *post-hoc* que, en este caso, es el test de Nemenyi [Nemenyi, 1963], que compara diferentes métodos entre sí. El rendimiento de 2 clasificadores es significativamente diferente si los rankings medios correspondientes difieren en al menos la Diferencia Crítica (DC) [Miller, 1996]. Ésta puede ser calculada a partir de los valores críticos propuestos en la literatura, k y N. Los niveles de significación considerados han sido 0,05 para el test de Iman-Davenport y 0,05 y 0,10 para el test *post-hoc*.

Para determinar si hay diferencias significativas entre los tres modelos, en primer lugar, aplicamos el test de Iman-Davenport. Debido a que en EDDSig tenemos 2 configuraciones, en cierta manera equivalentes a la configuración de TSEASig, consideramos el mejor valor de los 2 valores medios de ambas configuraciones. En TSEA, tenemos dos configuraciones, 1* y 2*, aunque para hacer justa la comparación sólo usamos una de ellas, en nuestro caso 1*. En consecuencia, comparamos el valor de TSEASig, el mejor valor de las configuraciones 1S y 2S de EDDSig y el valor de la configuración 1* de TSEA.

Los rankings medios de las metodologías EDDSig, TSEASig y TSEA son 2,64, 1,78 y 1,57, respectivamente. Según los resultados del test de Iman-Davenport, dado que el estadístico $F_F = 6,16$ es mayor que el valor crítico $F(2,\ 26) = 3,37$ a un nivel de significación $\alpha = 0,05$, se rechaza la hipótesis nula. Por tanto, procedemos con un test *post-hoc* de Nemenyi. Si los rankings medios entre 2 clasificadores difieren en un valor igual o superior a la DC, existen diferencias significativas.

La tabla 6.23 expone los resultados del test de Nemenyi e indica para mayor claridad la diferencia de ranking entre cada par diferente y el nivel de diferencia significativa detectado. En una única fila, se muestra la DC para los diferentes niveles de significación. Un análisis basado en los resultados del test de Nemenyi nos permite afirmar lo siguiente: existen diferencias significativas entre TSEASig y EDDSig a favor del primero al nivel $\alpha = 0,10$. En otras palabras, respecto a las RNAs con US, la propuesta basada en el algoritmo evolutivo en dos fases, TSEASig supera la formulación basada en la distribución del diseño experimental, EDDSig. Entre TSEA y TSEASig no hay diferencias significativas. Esto indica que la nueva estrategia es competitiva respecto a TSEA. Podemos concluir que la metodología basada en el AE en dos fases es apropiada tanto para US como para UP. Finalmente, TSEA es significativamente mejor que EDDSig al nivel $\alpha = 0,05$.

	EDDSig	TSEASig	TSEA
EDDSig		$0,86^{o}$	$1,07^{*}$
TSEASig			$0,21$
		$DC_{(\alpha\ =\ 0,05)} = 0,89;\ DC_{(\alpha\ =\ 0,10)} = 0,78$	

Cada celda rellena contiene la diferencia de ranking entre los métodos en la fila y la columna. También se especifica si el segundo método supera al primero a un nivel de significación de 0,05 (*) o 0,10 (^{o}).

Tabla 6.23. Comparaciones por pares de EDDSig, TSEASig y TSEA por medio de un test de Nemenyi.

Comparando los diferentes modelos comentados hasta el momento, podemos concluir que TSEA es cuantitativamente superior a TSEASig y EDDSig, sin embargo, estadísticamente hablando las diferencias son sólo significativas frente a EDDSig.

Un análisis de coste computacional permite completar la comparación entre TSEA, EDDSig y TSEASig. Los experimentos han sido ejecutados en un ordenador de sobremesa con un procesador Intel Core 2 Quad a 2,4GHz y una memoria física RAM de 2GB. La tasa de aceleración del método *i* con respecto al método *j*, viene dada por la ecuación (6.4).

$$Tasa\_Aceleración(i, j) = \frac{tiempo(j)}{tiempo(i)} \quad (6.4)$$

La tabla 6.24 presenta los resultados de tiempo relativos a coste computacional por iteración, medido en segundos (s). La primera columna especifica el nombre del conjunto de datos. Desde la segunda columna a la quinta se refleja el tiempo consumido en una iteración con cada configuración de las diferentes metodologías. Las dos últimas columnas ilustran las tasas de aceleración de TSEASig frente a EDDSig y a TSEA. La última fila contiene el promedio de los valores de la columna. Dado que 2 configuraciones de EDDSig son equivalentes a una de TSEASig, la tasa de aceleración representada por *Tasa_Aceleración(TSEASig, EDDSig)* se calcula como la suma de los tiempos de las configuraciones 1S y 2S dividido por el tiempo de la configuración 1S*.

Observando la tabla 6.24, podemos concluir que, respecto a las US, TSEASig es 1,66 veces más rápido que EDDSig. En la comparación entre TSEASig y TSEA, el primero es 1,37 veces más rapido que el segundo. Los tiempos empíricos informan que la propuesta TSEASig es mucho más eficiente que la metodología previa, TSEA. Las medidas de eficiencia se

basan en el tiempo de computación de una iteración de la fase de entrenamiento de toda la población durante la ejecución del AE. La velocidad depende del propio proceso evolutivo, esto es, el recorrido en el espacio de búsqueda, y del número de operaciones matemáticas que están implicadas en el entrenamiento de los diferentes modelos.

Conjunto de datos	**Coste Computacional (s)**				**Tasa_Aceleración**	
	Metodologías				(TSEASig, EDDSig)	(TSEASig, TSEA)
	TSEA	EDDSig		TSEASig		
	1*	1S	2S	1S*		
Australian	303	126	155	191	1,47	1,59
Balance	287	317	370	382	1,80	0,75
Cancer	98	40	51	61	1,49	1,61
Heart	207	114	133	134	1,84	1,54
HeartY	62	34	37	51	1,39	1,22
Hepatitis	36	19	22	25	1,64	1,44
Horse	817	304	378	344	1,98	2,38
Hypothyroid	6503	4525	6090	6694	1,59	0,97
Newthyroid	122	83	101	116	1,59	1,05
Pima	105	63	73	88	1,55	1,19
Waveform	9213	5217	6412	7155	1,63	1,29
Yeast	49320	21983	30349	28538	1,83	1,73
BTX	256	158	174	190	1,75	1,35
Listeria	231	168	198	216	1,69	1,07
Promedio	4825,71	2367,93	3181,64	3156,07	1,66	1,37

Tabla 6.24. Coste computacional y tasas de aceleración de TSEA, EDDSig y TSEASig.

La ventaja de TSEASig frente a la propuesta en la que se basa, TSEA, radica en el hecho de que es más eficiente computacionalmente, obteniendo una tasa de aceleración media para todos los conjuntos de

datos de 1,37. Por ejemplo, para el conjunto de datos *Horse*, TSEASig requiere menos de la mitad del tiempo empleado por TSEA para llevar a cabo el entrenamiento. Respecto a la precisión, TSEASig es significativamente superior a EDDSig y, frente a TSEA, es algo inferior, aunque es competitiva, dado que no existen diferencias significativas.

6.4. TERCER MODELO: EDDFS

El modelo EDDFS usa 6 métodos de selección de atributos implementados como filtros que han sido aplicados de manera independiente sobre el conjunto de entrenamiento de cada conjunto de datos. Para la experimentación, se ha llevado a cabo una selección sobre el repositorio del UCI de los conjuntos de datos con una tasa de error en el conjunto de generalización de al menos el 20%, aproximadamente, con clasificadores de referencia como C4.5 o 1-NN.

En la tabla 6.25 se muestra el número de atributos tanto del conjunto de entrenamiento original (columna FØ) como del conjunto de entrenamiento reducido, obtenido aplicando cada uno de los filtros, junto con el porcentaje de reducción del número de entradas para cada uno de los mencionados filtros siguiendo la ecuación (6.5). En el filtro InfoGain, hemos realizado un diseño experimental previo para determinar el valor del umbral del método de ranking *Ranker* aplicando un *5-fold* con 5 repeticiones sobre el conjunto de entrenamiento. Se experimentó con los valores 0,01 y 0,05, optando por el primer valor, dado que el rendimiento de los clasificadores es superior. Para referirnos a cada uno de los filtros, empleamos la denominación *Fi* siendo *i* el índice del filtro, cuyo significado aparece desglosado en el pie de la tabla. En la última fila se

indican los promedios de cada una de las columnas. El porcentaje de reducción del número de entradas se define de la siguiente manera:

$$Reducción\_Entradas(\%) = \left(1 - \frac{Entradas(F_i)}{Entradas(F\varnothing)}\right) * 100 \qquad i = 1,\ldots,6 \qquad (6.5)$$

siendo *i* el índice del filtro y Entradas(*j*) representa el número de entradas de un conjunto de datos datos con el filtro *j*. La reducción media de dimensionalidad oscila entre un poco más de un quinto y algo menos de la mitad del número de entradas de partida.

Conjunto de datos	**Entradas**							**Reducción_Entradas (%)**					
	FØ	**F1**	**F2**	**F3**	**F4**	**F5**	**F6**	**F1**	**F2**	**F3**	**F4**	**F5**	**F6**
Breast	15	4	2	2	5	3	4	73,33	86,67	86,67	66,67	80,00	73,33
Breast-t	9	6	5	6	8	4	6	33,33	44,44	33,33	11,11	55,56	33,33
Cardiotocogr.	31	9	10	21	20	8	7	70,97	67,74	32,26	35,48	74,19	77,42
Heart	13	7	8	9	9	6	7	46,15	38,46	30,77	30,77	53,85	46,15
Hepatitis	19	10	11	5	12	6	10	47,37	42,11	73,68	36,84	68,42	47,37
Labor	29	7	5	5	11	8	8	75,86	82,76	82,76	62,07	72,41	72,41
Lymphogr.	38	11	9	9	15	8	12	71,05	76,32	76,32	60,53	78,95	68,42
Parkinsons	22	5	7	6	21	4	6	77,27	68,18	72,73	4,55	81,82	72,73
Pima	8	3	4	5	6	4	4	62,50	50,00	37,50	25,00	50,00	50,00
Plates	27	16	19	21	27	6	10	40,74	29,63	22,22	0,00	77,78	62,96
Promoter	114	7	8	7	13	11	10	93,86	92,98	93,86	88,60	90,35	91,23
Waveform	40	14	15	15	19	5	14	65,00	62,50	62,50	52,50	87,50	65,00
Winequal.-red	11	5	8	8	8	4	4	54,55	27,27	27,27	27,27	63,64	63,64
Yeast	8	5	7	7	7	6	7	37,50	12,50	12,50	12,50	25,00	12,50
Promedio	27,43	7,79	8,43	9,00	12,93	5,93	7,79	60,68	55,83	53,17	36,71	68,53	59,75

F1 es spBI_CFS; F2 es spBI_CNS; F3 es cnBI_CNS;
F4 es InfoGain; F5 es FCBF; F6 es BestFirst_CFS

Tabla 6.25. Número de entradas y porcentaje de reducción del modelo EDDFS.

En el modelo EDDFS se han definido 4 configuraciones que se describen en la tabla 4.5. Los valores de los parámetros de la configuración base de EDDFS figuran en la tabla 6.26 junto con los de EDD para observar más fácilmente las diferencias. Para la determinación de los parámetros de EDDFS, hemos usado un diseño experimental previo sobre el conjunto de entrenamiento de tipo *5-fold* restringiendo los valores superiores a los establecidos para EDD, ya que es lógico pensar que con un menor número de entradas en el conjunto de datos, el número máximo de nodos en la capa oculta no debe incrementarse y la evolución necesaria debe ser igual o menor.

Conjunto de datos	**EDD**			**EDDFS**		
	Número máximo de neuronas (*neu*)	**Número máximo de generaciones (*gen*)**	α_2	**Número máximo de neuronas (*neu'*)**	**Número máximo de generaciones (*gen'*)**	α_2
Breast	9	500	1	7	500	1
Breast-t	5	300	1	5	150	1
Cardiotocography	6	300	1	5	150	1
Heart	6	500	1	4	25	1
Hepatitis	3	100	1	3	100	1
Labor	6	300	1	5	300	1
Lymphography	6	500	1	6	100	1
Parkinsons	6	500	1	3	500	1
Pima	3	120	1	3	120	1
Plates	6	500	1	5	500	1
Promoter	11	500	1	5	300	1
Waveform	3	500	1	3	500	1
Winequality-red	6	300	1	4	300	1
Yeast	11	500	1	11	500	1

Tabla 6.26. Valores de los parámetros de las configuraciones base de EDD y EDDFS.

Los resultados de EDDFS, junto con los de EDD, figuran en la tabla 6.27. Para cada conjunto de datos se muestran los resultados tanto sin como con selección de atributos para cada una de las 4 configuraciones tanto de EDD como de EDDFS. Los mejores resultados se indican en negrita tanto sin aplicar selección de atributos como aplicando cada uno de los diferentes filtros. El mejor resultado global de EDDFS para cada conjunto de datos considerando todos los filtros aparece sombreado.

Conjunto de datos	**Filtro**	**Topología config. base**	**CCR_{gen}±D. Típica**			
			Configuración			
			1/1′	**2/2′**	**3/3′**	**4/4′**
Breast	-	15: 9: 1	63,85±3,81	63,00±3,24	**64,27±3,89**	63,43±3,80
	spBI_CFS	4: 7: 1	70,84±1,92	**70,93±1,59**	70,18±1,77	70,00±1,92
	spBI_CNS	2: 7: 1	**69,20±0,48**	69,10±0,35	69,06±0,25	69,06±0,25
	cnBI_CNS	2: 7: 1	**69,20±0,48**	69,10±0,35	69,06±0,25	69,06±0,25
	InfoGain	5: 7: 1	68,49±1,83	**68,73±2,96**	68,26±3,02	67,88±2,67
	FCBF	3: 7: 1	68,26±1,03	68,17±1,20	68,54±0,77	**68,64±0,90**
	BestFirst_CFS	4: 7: 1	70,84±1,92	**70,93±1,59**	70,18±1,77	70,00±1,92
Breast-t	-	9: 5: 5	52,80±6,42	54,00±8,83	**54,53±8,03**	53,46±10,37
	spBI_CFS	6: 5: 5	**56,00±9,15**	54,66±7,52	53,33±8,15	55,33±7,20
	spBI_CNS	5: 5: 5	54,93±7,64	54,26±7,90	**59,86±4,98**	57,73±8,44
	cnBI_CNS	6: 5: 5	**61,06±8,19**	57,46±8,88	58,13±8,64	58,80±7,80
	InfoGain	8: 5: 5	55,73±7,49	55,60±7,07	**57,20±9,28**	54,26±9,43
	FCBF	4: 5: 5	55,20±6,51	56,00±5,66	**57,87±6,79**	57,33±6,91
	BestFirst_CFS	6: 5: 5	54,93±7,27	**57,07±7,50**	54,27±8,51	55,73±7,20
Cardiotocography	-	31: 6: 2	80,93±2,76	79,94±3,70	**81,25±3,17**	81,04±3,79
	spBI_CFS	9: 5: 2	**85,45±1,61**	84,16±2,27	84,34±2,31	83,93±2,85
	spBI_CNS	10: 5: 2	**83,17±2,42**	82,78±3,63	82,30±3,51	81,04±3,35
	cnBI_CNS	21: 5: 2	**77,27±1,41**	76,44±2,14	76,89±1,73	76,52±2,40
	InfoGain	20: 5: 2	81,37±2,76	80,35±3,08	81,14±3,20	**81,71±2,62**
	FCBF	8: 5: 2	81,09±2,36	**81,81±1,87**	81,22±2,37	81,62±2,47

	BestFirst_CFS	7: 5: 2	81,57±1,86	**81,79±2,34**	81,30±1,91	81,35±1,80
Heart	-	13: 6: 1	75,93±2,40	75,83±3,27	**76,23±2,48**	76,03±3,50
	spBI_CFS	7: 4: 1	76,23±1,86	76,47±2,12	75,93±2,33	**77,50±2,01**
	spBI_CNS	8: 4: 1	76,08±2,50	75,98±2,30	**76,47±2,01**	75,59±2,37
	cnBI_CNS	9: 4: 1	77,40±2,10	76,76±2,09	77,89±2,49	**77,99±1,79**
	InfoGain	9: 4: 1	77,40±2,10	76,76±2,09	77,89±2,49	**77,99±1,79**
	FCBF	6: 4: 1	76,08±3,27	76,32±2,68	75,58±3,10	**76,32±3,04**
	BestFirst_CFS	7: 4: 1	76,23±1,86	76,47±2,12	75,93±2,33	**77,50±2,01**
Hepatitis	-	19: 3: 1	84,47±4,49	**85,52±4,67**	84,47±4,55	84,29±5,33
	spBI_CFS	10: 3: 1	88,77±2,49	88,77±2,93	88,95±2,80	**89,91±2,59**
	spBI_CNS	11: 3: 1	89,04±2,40	89,30±2,49	**89,56±2,13**	89,47±2,85
	cnBI_CNS	5: 3: 1	86,67±1,53	**86,67±1,82**	86,40±1,40	86,32±1,45
	InfoGain	12: 3: 1	87,01±3,01	87,63±2,20	87,89±2,72	**88,07±2,74**
	FCBF	6: 3: 1	88,42±2,81	**88,68±2,68**	88,24±2,82	88,24±2,26
	BestFirst_CFS	10: 3: 1	88,77±2,49	88,77±2,93	88,95±2,80	**89,91±2,59**
Labor	-	29: 6: 1	84,04±11,04	83,57±10,29	**84,28±10,67**	82,85±13,98
	spBI_CFS	7: 5: 1	**92,85±4,96**	92,61±5,13	90,95±4,93	91,42±4,74
	spBI_CNS	5: 5: 1	**91,19±3,07**	90,95±3,21	89,76±4,05	89,76±3,60
	cnBI_CNS	5: 5: 1	**91,19±3,07**	90,95±3,21	89,76±4,05	89,76±3,60
	InfoGain	11: 5: 1	92,61±5,13	**95,00±5,97**	92,61±5,77	**95,00±5,01**
	FCBF	8: 5: 1	91,42±3,93	91,42±4,35	91,66±4,22	**92,38±4,56**
	BestFirst_CFS	8: 5: 1	95,71±4,44	93,57±5,42	**96,19±4,08**	94,29±4,75
Lymphography	-	38: 6: 3	75,49±6,98	76,12±6,95	76,48±6,70	**76,85±7,56**
	spBI_CFS	11: 6: 3	**78,28±5,91**	77,47±4,83	78,01±6,05	75,04±6,49
	spBI_CNS	9: 6: 3	77,02±4,90	76,93±5,15	**77,20±4,79**	75,31±5,48
	cnBI_CNS	9: 6: 3	76,93±4,74	**79,18±4,60**	78,46±4,83	**79,18±4,97**
	InfoGain	15: 6: 3	78,64±5,46	78,79±5,38	78,19±4,48	**79,27±5,37**
	FCBF	8: 6: 3	**80,00±4,79**	79,91±4,64	79,64±4,95	77,48±3,71
	BestFirst_CFS	12: 6: 3	78,46±6,08	81,08±4,76	**81,17±5,56**	79,54±4,69
Parkinsons	-	22: 6: 1	**79,66±5,01**	78,16±4,77	78,98±4,05	79,32±4,76
	spBI_CFS	5: 3: 1	79,66±2,37	79,32±2,32	**79,93±2,15**	79,05±2,83
	spBI_CNS	7: 3: 1	80,27±4,23	**81,84±4,20**	80,61±3,07	80,14±3,90
	cnBI_CNS	6: 3: 1	78,03±1,16	**80,07±2,82**	78,78±1,98	79,66±3,24
	InfoGain	21: 3: 1	80,20±4,45	79,65±3,14	**80,34±3,65**	78,50±3,24

	FCBF	4: 3: 1	81,42±1,54	**82,17±2,33**	81,83±1,88	80,95±3,14
	BestFirst_CFS	6: 3: 1	80,88±1,96	**82,24±2,21**	81,49±1,92	81,29±2,73
Pima	-	8: 3: 1	77,33±2,36	**78,61±1,88**	76,96±1,67	77,69±1,79
	spBI_CFS	3: 3: 1	79,54±0,90	79,49±0,79	79,60±0,87	**79,89±0,92**
	spBI_CNS	4: 3: 1	**75,48±1,42**	75,19±1,26	75,00±1,17	75,31±1,51
	cnBI_CNS	5: 3: 1	78,42±1,09	**78,76±1,13**	78,73±1,06	78,71±1,47
	InfoGain	6: 3: 1	78,94±1,34	79,09±1,40	79,02±1,50	**79,35±1,48**
	FCBF	4: 3: 1	80,73±0,87	**80,80±1,14**	80,69±1,00	80,78±0,82
	BestFirst_CFS	4: 3: 1	80,53±0,87	80,65±1,18	80,65±1,14	**80,83±0,96**
Plates	-	27: 6: 6	52,61±5,09	51,18±5,35	52,01±3,37	**52,82±4,03**
	spBI_CFS	16: 5: 6	**55,25±3,66**	52,36±2,94	50,83±5,95	54,14±3,89
	spBI_CNS	19: 5: 6	54,51±4,92	55,07±2,81	54,24±2,32	**57,30±4,88**
	cnBI_CNS	21: 5: 6	53,08±4,96	54,91±4,44	54,82±5,62	**55,06±4,12**
	InfoGain	27: 5: 6	52,61±5,09	51,18±5,35	52,01±3,37	**52,82±4,03**
	FCBF	6: 5: 6	52,21±4,40	53,25±4,22	52,96±4,40	**53,28±3,72**
	BestFirst_CFS	10: 5: 6	49,94±3,72	49,35±5,14	52,00±6,12	**52,74±5,91**
Promoter	-	114: 11: 1	59,74±9,30	58,21±9,67	**60,51±10,00**	55,51±10,03
	spBI_CFS	7: 5: 1	84,48±3,97	**84,62±3,78**	83,20±3,97	82,94±4,12
	spBI_CNS	8: 5: 1	67,43±5,77	67,05±5,11	66,15±5,56	**67,94±5,74**
	cnBI_CNS	7: 5: 1	75,89±4,39	75,51±4,45	76,41±3,74	**76,53±4,55**
	InfoGain	13: 5: 1	77,06±6,61	**77,43±5,41**	75,25±6,36	76,79±6,26
	FCBF	11: 5: 1	80,64±6,50	79,69±5,71	78,46±7,46	**81,69±5,59**
	BestFirst_CFS	10: 5: 1	77,69±5,84	76,15±6,50	77,05±5,93	**77,95±6,69**
Waveform	-	40: 3: 2	81,43±2,10	82,78±0,64	82,05±1,64	**84,32±1,73**
	spBI_CFS	14: 3: 2	84,97±1,13	**86,54±0,48**	84,92±0,98	86,30±0,95
	spBI_CNS	15: 3: 2	85,39±1,41	85,78±0,74	85,20±1,14	**86,37±0,84**
	cnBI_CNS	15: 3: 2	84,87±0,93	**86,75±0,57**	85,55±1,21	85,66±0,80
	InfoGain	19: 3: 2	85,58±1,15	**86,35±0,95**	85,22±1,30	85,99±1,20
	FCBF	5: 3: 2	79,95±0,61	**80,06±0,54**	80,00±0,49	80,01±0,53
	BestFirst_CFS	14: 3: 2	84,97±1,13	**86,54±0,48**	84,92±0,98	86,30±0,95
Winequality-red	-	11: 6: 5	61,03±1,30	60,80±1,25	**61,16±1,20**	60,98±1,42
	spBI_CFS	5: 4: 5	**61,67±1,10**	61,49±0,99	61,21±1,11	61,15±1,30
	spBI_CNS	8: 4: 5	61,70±1,06	61,54±1,10	**61,75±1,01**	61,66±1,05
	cnBI_CNS	8: 4: 5	61,70±1,06	61,54±1,10	**61,75±1,01**	61,66±1,05

	InfoGain	8: 4: 5	61,70±1,06	61,54±1,10	**61,75±1,01**	61,66±1,05
	FCBF	4: 4: 5	61,39±0,88	61,51±0,95	61,29±1,04	**61,67±0,84**
	BestFirst_CFS	4: 4: 5	**61,67±1,10**	61,49±0,99	61,21±1,11	61,15±1,30
Yeast	-	8: 11: 9	59,18±1,17	**59,62±1,27**	58,50±1,74	59,18±1,83
	spBI_CFS	5: 11: 9	**59,82±1,22**	59,53±1,36	58,95±1,14	59,72±1,46
	spBI_CNS	7: 11: 9	60,10±1,41	**60,36±1,16**	59,93±1,85	60,20±1,46
	cnBI_CNS	7: 11: 9	60,10±1,41	**60,36±1,16**	59,93±1,85	60,20±1,46
	InfoGain	7: 11: 9	60,10±1,41	**60,36±1,16**	59,93±1,85	60,20±1,46
	FCBF	6: 11: 9	**58,23±1,27**	58,19±1,41	57,91±1,02	58,05±1,22
	BestFirst_CFS	7: 11: 9	60,10±1,41	**60,36±1,16**	59,93±1,85	60,20±1,46

Tabla 6.27. Resultados de EDD y EDDFS.

Desde el punto de vista cuantitativo, se observa que, en términos generales, el modelo EDD mejora los resultados al aplicar selección de atributos. Seguidamente, comparamos los modelos EDD y EDDFS por medio de tests estadísticos no paramétricos.

6.4.1. Análisis estadístico

A continuación, detallamos el análisis estadístico no paramétrico que hemos seguido. Para determinar si hay diferencias significativas aplicamos un test de Iman-Davenport, que compara los rankings medios de los algoritmos, donde un ranking bajo indica un buen rendimiento del algoritmo y un valor alto un mal rendimiento. En las siguientes tablas de este subapartado, empleamos FØ para especificar que no se aplica ningún filtro. Los rankings medios de todos los métodos, sin (FØ) y con selección de atributos (spBI_CFS, spBI_CNS, cnBI_CNS, InfoGain, FCBF y BestFirst_CFS) se reflejan en la tabla 6.28. Los resultados del test de Iman-Davenport se presentan en la tabla 6.29, según los cuales, dado que el estadístico F_F es mayor que el valor crítico, se rechaza la hipótesis nula.

Filtro	Ranking medio
FØ	6,61
spBI_CFS	3,29
spBI_CNS	3,75
cnBI_CNS	3,71
InfoGain	3,75
FCBF	4,07
BestFirst_CFS	2,82

Tabla 6.28. Rankings medios de EDD y EDDFS.

Estadísticos		Valor crítico para $\alpha = 0,05$
χ_F^2	F_F	$F(6,\ 78)$
26,60	6,07	2,22

Tabla 6.29. Estadísticos y valor crítico del test de Iman-Davenport de los modelos EDD y EDDFS.

Se aplica, por tanto, un test post-hoc de Bonferroni-Dunn, que compara una serie de métodos con un método de control, determinando si las diferencias de ranking difieren el menos en la DC [Miller, 1996]. Ésta se calcula a partir de los valores críticos del test de Bonferroni-Dunn, el número de algoritmos y el número de conjuntos de datos. En nuestro caso, realizamos una comparación de los métodos que emplean selección de atributos frente al método de control (FØ) que no usa selección de atributos. La tabla 6.30 exhibe los resultados del test de Bonferroni-Dunn, donde la diferencia de ranking, la DC (para $\alpha = 0.05$ y $\alpha = 0.10$) y el nivel de diferencia significativa detectado han sido indicados para mayor claridad.

FØ frente a	Diferencia de ranking (Método de control – método comparado)	Nivel de significación para el método comparado
spBI_CFS	3,32	*
spBI_CNS	2,86	*
cnBI_CNS	2,90	*
InfoGain	2,86	*

FCBF	2,54	*
BestFirst_CFS	3,79	*
	$DC_{(\alpha = 0,05)} = 2,15;\ DC_{(\alpha = 0,10)} = 1,95$	

*: Diferencia estadísticamente significativa con $\alpha = 0,05$

Tabla 6.30. Valores de diferencias críticas y diferencias de ranking de EDD y EDDFS por medio de un test de Bonferroni-Dunn (FØ es el método de control).

Del análisis de los resultados del test de Bonferroni-Dunn podemos concluir que existen diferencias significativas entre el modelo EDD aplicando cada uno de los filtros considerados y EDD sin aplicar selección de atributos. Los tests estadísticos ponen de manifiesto que el rendimiento de las RNUP mejora significativamente preprocesando el conjunto de datos con cualquiera de los filtros empleados en el modelo EDDFS. No obstante, el comportamiento de BestFirst_CFS es ligeramente superior a los demás filtros.

6.4.2. Comparación con otros clasificadores

La tabla 6.31 presenta una comparativa de los resultados obtenidos con la mejor de las 4 configuraciones de EDDFS usando los diferentes filtros considerados frente a otros clasificadores tanto sin aplicar como aplicando selección de atributos. Para cada filtro, el mejor promedio obtenido se indica en negrita y el segundo mejor, en cursiva. Como podemos observar, el modelo EDDFS es superior a los otros clasificadores aplicando selección de atributos. Es destacable el hecho de que SVM sin aplicar selección atributos tiene una precisión de 74,12%, que es ligeramente inferior frente al mejor resultado obtenido por EDDFS con el filtro spBI_CFS (75,27%). La selección de atributos en el caso de SVM aporta mejoras en algunos conjuntos de datos.

Los tres filtros que obtienen los mejores resultados medios con EDDFS son, por este orden, spBI_CFS, FCBF y BestFirst_CFS; es reseñable que entre dichos filtros no hay más de 0,3 puntos de diferencia en cuanto a promedio global de CCR_{gen}. El algoritmo C4.5 presenta un excelente comportamiento con BestFirst_CFS y logra un aumento en media de 3,68 puntos. Con el clasificador MLP, los filtros spBI_CFS e InfoGain son los más adecuados, mejorando ligeramente el rendimiento en promedio. En PART los filtros más apropiados son BestFirst_CFS y spBI_CFS, que permiten incrementar en media casi 3,30 y 2,44 puntos, respectivamente.

Conjunto de datos	**Filtro**	**Clasificador**					
		C4.5	**1-NN**	**SVM**	**PART**	**MLP**	**EDDFS**
Breast	-	**70,42**	64,79	64,79	69,01	60,80	64,27
	spBI_CFS	69,01	70,42	66,20	**71,83**	69,01	70,93
	spBI_CNS	69,01	**70,42**	64,79	69,01	69,01	69,20
	cnBI_CNS	69,01	**70,42**	64,79	69,01	69,01	69,20
	InfoGain	64,79	**70,42**	64,79	70,42	69,30	68,73
	FCBF	69,01	**70,42**	64,79	69,01	69,53	68,64
	BestFirst_CFS	69,01	70,42	66,20	**71,83**	69,01	70,93
Breast-t	-	52,00	60,00	52,00	44,00	**63,20**	54,53
	spBI_CFS	56,00	52,00	60,00	44,00	**65,33**	56,00
	spBI_CNS	52,00	48,00	60,00	52,00	**66,40**	59,86
	cnBI_CNS	52,00	52,00	64,00	52,00	**67,20**	61,06
	InfoGain	56,00	56,00	60,00	48,00	**64,67**	57,20
	FCBF	48,00	48,00	56,00	48,00	**65,60**	57,87
	BestFirst_CFS	**68,00**	56,00	60,00	56,00	65,47	57,07
Cardiotocography	-	82,71	76,32	**83,65**	82,52	80,75	81,25
	spBI_CFS	77,07	81,77	81,20	82,52	81,94	**85,45**
	spBI_CNS	84,21	79,70	79,14	**85,15**	80,91	83,17
	cnBI_CNS	75,19	63,91	75,19	75,00	68,29	**77,27**
	InfoGain	83,08	76,88	**84,21**	81,58	81,87	81,71
	FCBF	77,82	81,20	81,20	77,26	80,13	**81,81**
	BestFirst_CFS	78,38	80,45	81,39	81,20	80,86	**81,79**

Heart	-	70,59	73,53	**76,47**	73,53	74,85	76,23
	spBI_CFS	73,53	73,53	76,47	**77,94**	72,50	77,50
	spBI_CNS	73,53	75,00	**76,47**	75,00	73,09	**76,47**
	cnBI_CNS	72,06	75,00	76,47	75,00	74,85	**77,99**
	InfoGain	72,06	75,00	76,47	75,00	74,85	**77,99**
	FCBF	73,53	70,59	**77,94**	75,00	74,90	76,32
	BestFirst_CFS	73,53	73,53	76,47	**77,94**	72,50	77,50
Hepatitis	-	84,21	86,84	**89,47**	81,58	84,73	85,52
	spBI_CFS	84,21	89,47	86,84	84,21	87,28	**89,91**
	spBI_CNS	89,47	**92,11**	89,47	86,84	86,84	89,56
	cnBI_CNS	**89,47**	84,21	**89,47**	84,21	84,21	86,67
	InfoGain	**89,47**	86,84	86,84	84,21	87,89	88,07
	FCBF	**89,47**	84,21	**89,47**	86,84	87,72	88,68
	BestFirst_CFS	84,21	89,47	86,84	84,21	87,28	**89,91**
Labor	-	85,71	71,43	78,57	**85,71**	69,52	84,28
	spBI_CFS	85,71	71,43	78,57	85,71	64,29	**92,85**
	spBI_CNS	85,71	64,28	78,57	78,57	78,57	**91,19**
	cnBI_CNS	85,71	64,28	78,57	78,57	78,57	**91,19**
	InfoGain	85,71	78,57	71,43	85,71	71,43	**95,00**
	FCBF	85,71	78,57	71,43	78,57	71,43	**92,38**
	BestFirst_CFS	85,71	64,29	71,43	85,71	57,62	**96,19**
Lymphography	-	75,68	83,78	**91,89**	75,68	86,58	76,85
	spBI_CFS	**88,29**	78,38	83,78	70,27	73,24	78,28
	spBI_CNS	75,68	70,27	**78,38**	64,86	71,89	77,20
	cnBI_CNS	75,68	70,27	78,38	64,86	71,89	**79,18**
	InfoGain	81,08	81,08	**86,49**	64,86	83,42	79,27
	FCBF	**81,08**	75,68	**81,08**	70,27	74,50	80,00
	BestFirst_CFS	81,08	81,08	81,08	64,86	80,45	**81,17**
Parkinsons	-	71,43	77,55	75,51	75,51	77,62	**79,66**
	spBI_CFS	75,51	79,59	75,51	77,55	**81,56**	79,93
	spBI_CNS	75,51	81,63	75,51	75,51	75,71	**81,84**
	cnBI_CNS	79,59	79,59	75,51	**81,63**	75,65	80,07
	InfoGain	71,43	**85,71**	75,51	75,51	79,05	80,34
	FCBF	81,63	73,47	79,59	77,55	**84,83**	82,17

	BestFirst_CFS	73,47	81,63	75,51	79,59	**83,13**	82,24
Pima	-	74,48	73,96	78,13	74,48	75,94	**78,61**
	spBI_CFS	76,04	74,48	77,60	76,04	78,18	**79,89**
	spBI_CNS	69,79	71,88	73,96	72,92	74,25	**75,48**
	cnBI_CNS	74,48	67,19	78,65	74,48	76,89	**78,76**
	InfoGain	74,48	69,27	77,08	74,48	76,13	**79,35**
	FCBF	76,04	67,71	79,17	76,04	79,01	**80,80**
	BestFirst_CFS	76,04	67,71	79,17	76,04	78,73	**80,83**
Plates	-	39,05	49,17	**57,02**	46,69	53,50	52,82
	spBI_CFS	40,50	51,24	51,03	46,90	**56,71**	55,25
	spBI_CNS	39,67	51,24	**57,44**	46,90	55,14	57,30
	cnBI_CNS	38,22	50,62	55,17	44,63	**55,24**	55,06
	InfoGain	39,05	49,17	**57,02**	46,69	53,50	52,82
	FCBF	44,63	43,18	45,04	49,79	52,85	**53,28**
	BestFirst_CFS	54,75	47,31	51,65	51,65	**57,33**	52,74
Promoter	-	69,23	65,38	**88,46**	53,85	86,03	60,51
	spBI_CFS	73,08	57,69	**84,62**	80,77	84,49	**84,62**
	spBI_CNS	76,92	61,54	76,92	**80,77**	65,38	67,94
	cnBI_CNS	80,77	57,69	**84,62**	76,92	75,64	76,53
	InfoGain	73,08	69,23	76,92	**80,77**	72,95	77,43
	FCBF	73,08	76,92	73,08	80,77	78,21	**81,69**
	BestFirst_CFS	73,08	69,23	73,08	**80,77**	76,28	77,95
Waveform	-	74,80	68,96	**86,24**	76,88	84,85	84,32
	spBI_CFS	74,40	75,36	**86,88**	77,04	83,21	86,54
	spBI_CNS	74,88	74,88	**87,12**	79,92	83,41	86,37
	cnBI_CNS	74,40	76,64	**87,12**	79,68	86,27	86,75
	InfoGain	76,32	73,76	**87,12**	76,64	82,96	86,35
	FCBF	74,72	69,12	78,80	74,00	77,57	**80,06**
	BestFirst_CFS	74,40	75,36	**86,88**	77,04	83,21	86,54
Winequality-red	-	53,85	49,88	59,55	51,36	56,35	**61,16**
	spBI_CFS	50,87	48,88	59,80	52,11	59,36	**61,67**
	spBI_CNS	50,12	49,63	58,81	52,85	57,04	**61,75**
	cnBI_CNS	50,12	49,63	58,81	52,85	57,04	**61,75**
	InfoGain	50,87	48,88	59,80	52,11	59,36	**61,75**

	FCBF	51,36	50,37	59,31	49,13	59,64	**61,67**
	BestFirst_CFS	50,87	48,88	59,80	52,11	59,36	**61,67**
Yeast	-	54,84	48,39	55,91	56,72	**59,94**	59,62
	spBI_CFS	53,49	48,92	54,03	54,84	**60,20**	59,82
	spBI_CNS	54,03	49,46	54,84	54,30	60,20	**60,36**
	cnBI_CNS	54,03	49,46	54,84	54,30	60,20	**60,36**
	InfoGain	54,03	49,46	54,84	54,30	60,20	**60,36**
	FCBF	52,69	48,12	51,61	52,96	**58,96**	58,23
	BestFirst_CFS	54,03	49,46	54,84	54,30	60,20	**60,36**
Promedio	-	68,50	67,86	74,12	67,68	72,48	71,40
	spBI_CFS	69,84	68,08	*73,04*	70,12	72,66	**75,27**
	spBI_CNS	69,32	67,15	*72,24*	69,62	71,28	**73,36**
	cnBI_CNS	69,34	65,07	*72,97*	68,80	71,50	**73,94**
	InfoGain	69,39	69,31	*72,75*	69,31	72,68	**74,25**
	FCBF	69,91	66,97	70,61	68,94	*72,49*	**74,97**
	BestFirst_CFS	71,18	68,20	71,74	70,95	*72,25*	**74,91**

Tabla 6.31. Comparación de los resultados obtenidos con EDDFS frente a otros clasificadores.

En términos generales, los filtros considerados permiten alcanzar en algunos conjuntos de datos importantes mejoras, que detallamos respecto al mejor resultado de partida con el conjunto de datos original, esto es, sin aplicar selección de atributos. En *Labor*, la selección de atributos permite mejorar más de 10 puntos, pasando del 85,71% con PART al 96,19% con EDDFS+BestFirst_CFS. Para el problema *Parkinsons*, el incremento es de más de 6 puntos, desde el 79,66% con EDDFS hasta el 85,71% con 1-NN+InfoGain. En *Breast-t*, el aumento está próximo a 5 puntos, puesto que partiendo del 63,20% con MLP se alcanza el 68,00% con C4.5+BestFirst_CFS. Encontramos mejoras de algo más de 2 puntos en *Hepatitis* y *Pima*; en el primero de ellos, desde el 89,47% de acierto con SVM, logramos un 92,11% con 1-NN+spBI_CNS, y, en el segundo conjunto

de datos, desde el 78,61% con EDDFS hasta el 80,83% con EDDFS+BestFirst_CFS.

Lo anteriormente comentado pone de manifiesto el teorema denominado *No-Free Lunch* [Wolpert, 2001], propuesto por David H. Wolpert, en el sentido de que en este contexto, como en cualquier otro ámbito de metaheurísticas, no es posible obtener un par formado por clasificador y filtro que alcance los mejores resultados para cualquier problema.

6.5. CUARTO MODELO: TSEAFS

Como hemos mencionado al describir el modelo TSEAFS, han sido aplicados 4 métodos de selección de atributos implementados como filtros, de manera independiente a cada conjunto de datos. Para la experimentación, se han seleccionado conjuntos de datos del UCI con una tasa de error sobre el conjunto de generalización de al menos el 20%, aproximadamente, con clasificadores de referencia como C4.5 o 1-NN.

La tabla 6.32 ilustra para cada conjunto de datos el número de atributos del conjunto de entrenamiento original (véase la columna etiquetada como FØ, en alusión a que ningún filtro ha sido aplicado) y aquellos que han sido obtenidos con los diferentes filtros (véase las columnas denominadas spBI_CFS, cnBI_CNS, FCBF y BF_CFS que es una abreviatura de BestFirst_CFS como se indica en el pie de la tabla) junto con el porcentaje de reducción del número de entradas del modelo de red para cada filtro en comparación con el conjunto de datos original, calculado siguiendo la ecuación (6.5), presentada en la sección 6.4. La última fila muestra el promedio del número de atributos y el porcentaje de reducción.

En todos los casos, los filtros decrementan satisfactoriamente la dimensionalidad de los datos, seleccionando en media menos de un tercio de los atributos originales.

Conjunto de datos	**Entradas**					**Reducción_Entradas (%)**			
	FØ	**spBI_CFS**	**cnBI_CNS**	**FCBF**	**BF_CFS**	**spBI_CFS**	**cnBI_CNS**	**FCBF**	**BF_CFS**
Appendicitis	7	4	2	2	5	42,86	71,43	71,43	28,57
Breast	15	4	2	3	4	73,33	86,67	80,00	73,33
Breast-t	9	6	6	4	6	33,33	33,33	55,56	33,33
Cardiotocogr.	31	9	21	8	7	70,97	32,26	74,19	77,42
Heart	13	7	9	6	7	46,15	30,77	53,85	46,15
Hepatitis	19	10	5	6	10	47,37	73,68	68,42	47,37
Labor	29	7	5	8	8	75,86	82,76	72,41	72,41
Led24	24	6	6	6	6	75,00	75,00	75,00	75,00
Lymphogr.	38	11	9	8	12	71,05	76,32	78,95	68,42
Parkinsons	22	5	6	4	6	77,27	72,73	81,82	72,73
Pima	8	3	5	4	4	62,50	37,50	50,00	50,00
Plates	27	16	21	6	10	40,74	22,22	77,78	62,96
Promoter	114	7	7	11	10	93,86	93,86	90,35	91,23
SPECTF	44	12	9	6	12	72,73	79,55	86,36	72,73
Vowel	11	3	9	7	3	72,73	18,18	36,36	72,73
Waveform	40	14	15	5	14	65,00	62,50	87,50	65,00
Winequal.-red	11	5	8	4	4	54,55	27,27	63,64	63,64
Yeast	8	5	7	6	7	37,50	12,50	25,00	12,50
Promedio	26,11	7,44	8,44	5,78	7,50	61,82	54,92	68,26	60,31

BF_CFS es BestFirst_CFS

Tabla 6.32. Número de entradas y porcentaje de reducción aplicando el modelo TSEAFS.

Los detalles relativos a las propiedades y los resultados experimentales del conjunto de datos *Liver-transplantation* aparecen en la

sección 6.4.2, puesto que se ha realizado un preprocesamiento específico por el problema que presenta la distribución de sus datos.

En el modelo TSEAFS se han definido 2 configuraciones, cuya única diferencia reside en el valor del parámetro α_2, que presentan una topología común. En TSEA también hay 2 configuraciones. La tabla 6.33 señala los valores de los parámetros de la primera configuración, denominada configuración base, de los modelos TSEA y TSEAFS para cada conjunto de datos, lo cual nos permite tener una visión conjunta de las diferencias. Los valores de los parámetros de TSEAFS, que son específicos para cada conjunto de datos, han sido determinados usando un diseño experimental de tipo *5-fold* sobre el conjunto de entrenamiento, explicado anteriormente, aunque tomando como cota superior los valores definidos en TSEA.

Conjunto de datos	**TSEA**			**TSEAFS**		
	Configuración 1*			**Configuración 1*#**		
	Número máximo de neuronas en cada población (*neu* y *neu+1*)	**Número máximo de generaciones en cada población (0,1**gen*) (Fase 1)**	α_2	**Número máximo de neuronas en cada población (*neu#* y *neu#+1*)**	**Número máximo de generaciones en cada población (0,1**gen#*) (Fase 1)**	α_2
Appendicitis	4 y 5	30	1	4 y 5	10	1
Breast	9 y 10	50	1	9 y 10	30	1
Breast-t	5 y 6	30	1	5 y 6	15	1
Cardiotocography	6 y 7	30	1	5 y 6	15	1
Heart	6 y 7	50	1	4 y 5	2	1
Hepatitis	3 y 4	30	1	3 y 4	30	1
Labor	6 y 7	30	1	4 y 5	30	1
Led24	8 y 9	50	1	8 y 9	50	1

Lymphography	6 y 7	50	1	6 y 7	10	1
Parkinsons	6 y 7	30	1	6 y 7	30	1
Pima	4 y 5	15	1	4 y 5	15	1
Plates	6 y 7	50	1	6 y 7	50	1
Promoter	11 y 12	50	1	6 y 7	30	1
SPECTF	6 y 7	50	1	6 y 7	30	1
Vowel	6 y 7	100	1	6 y 7	100	1
Waveform	3 y 4	50	1	3 y 4	50	1
Winequality-red	6 y 7	30	1	4 y 5	30	1
Yeast	11 y 12	100	1	11 y 12	100	1

Tabla 6.33. Valores de los parámetros de las configuraciones base de TSEA y TSEAFS.

En la tabla 6.34 se expresa para cada conjunto de datos las topologías y los resultados asociados sobre el conjunto de generalización de TSEAFS, junto con los de TSEA para obtener una visión global de las mejoras obtenidas por la aplicación de selección de atributos sobre los diferentes conjuntos de datos. Para cada conjunto de datos se muestra en su primera fila el resultado de TSEA, esto es, usando el conjunto de datos original, seguido por los de TSEAFS en las siguientes filas, empleando cada uno de los filtros indicados en la primera columna de dicha fila. Los mejores resultados se indican en negrita tanto sin aplicar selección de atributos como aplicando cada uno de los diferentes filtros. El mejor resultado global para cada conjunto de datos considerando todos los filtros aparece sombreado.

Conjunto de datos	**Filtro**	**Topología**	**CCR_{gen}±D. Típica**	
			Configuración	
			1* / 1*#	**2* / 2*#**
Appendicitis	-	7: [4, 5]: 1	**81,66±4,24**	80,51±2,84
	spBI_CFS	4: [4, 5]: 1	**82,82±2,19**	81,02±2,65
	cnBI_CNS	2: [4, 5]: 1	**80,89±0,70**	80,76±0,00

	FCBF	2: [4, 5]: 1	**80,38±2,73**	79,61±3,21
	BestFirst_CFS	5: [4, 5]: 1	**81,79±2,00**	81,66±2,97
Breast	-	15: [9, 10]: 1	**65,96±2,89**	62,76±3,08
	spBI_CFS	4: [9, 10]: 1	**69,85±1,50**	68,21±1,08
	cnBI_CNS	2: [9, 10]: 1	**69,01±0,00**	**69,01±0,00**
	FCBF	3: [9, 10]: 1	68,92±0,73	**69,10±0,36**
	BestFirst_CFS	4: [9, 10]: 1	**69,01±0,00**	**69,01±0,00**
Breast-t	-	9: [5, 6]: 5	54,53±7,89	**55,33±9,16**
	spBI_CFS	6: [5, 6]: 5	**54,40±6,77**	48,93±8,83
	cnBI_CNS	6: [5, 6]: 5	55,73±8,72	**57,87±6,87**
	FCBF	4: [5, 6]: 5	**60,93±4,77**	56,53±7,03
	BestFirst_CFS	6: [5, 6]: 5	**59,47±8,90**	56,00±6,96
Cardiotocography	-	31: [6, 7]: 2	**81,69±3,56**	81,55±2,90
	spBI_CFS	9: [5, 6]: 2	**85,26±2,27**	84,88±2,11
	cnBI_CNS	21: [5, 6]: 2	71,20±2,55	**76,71±1,04**
	FCBF	8: [5, 6]: 2	**81,55±1,69**	81,12±1,80
	BestFirst_CFS	7: [5, 6]: 2	81,58±2,48	**82,38±2,42**
Heart	-	13: [6, 7]: 1	76,62±2,33	**77,45±3,09**
	spBI_CFS	7: [4, 5]: 1	77,45±2,16	**77,69±2,28**
	cnBI_CNS	9: [4, 5]: 1	**78,57±1,99**	77,79±1,60
	FCBF	6: [4, 5]: 1	75,24±2,70	**75,34±2,80**
	BestFirst_CFS	7: [4, 5]: 1	77,45±2,16	**77,69±2,28**
Hepatitis	-	19: [3, 4]: 1	**85,79±4,51**	83,68±3,87
	spBI_CFS	10: [3, 4]: 1	**90,78±1,79**	89,29±1,53
	cnBI_CNS	5: [3, 4]: 1	86,14±1,81	**87,45±1,49**
	FCBF	6: [3, 4]: 1	85,00±1,56	**91,05±2,55**
	BestFirst_CFS	10: [3, 4]: 1	**90,78±1,79**	89,29±1,53
Labor	-	29: [6, 7]: 1	85,24±8,78	**86,90±5,96**
	spBI_CFS	7: [5, 6]: 1	93,09±4,39	**96,19±4,08**
	cnBI_CNS	5: [5, 6]: 1	87,62±4,16	**88,33±4,39**
	FCBF	8: [5, 6]: 1	**90,95±5,28**	90,48±5,73
	BestFirst_CFS	8: [5, 6]: 1	89,76±6,41	**89,76±5,20**
Led24	-	24: [8, 9]: 9	50,29±6,59	**51,03±5,58**
	spBI_CFS	6: [8, 9]: 9	67,26±1,46	**68,30±0,57**

	cnBI_CNS	6: [8, 9]: 9	67,26±1,46	**68,30±0,57**
	FCBF	6: [8, 9]: 9	67,26±1,46	**68,30±0,57**
	BestFirst_CFS	6: [8, 9]: 9	67,26±1,46	**68,30±0,57**
Lymphography	-	38: [6, 7]: 3	**79,37±4,73**	78,73±4,79
	spBI_CFS	11: [6, 7]: 3	**79,09±5,71**	78,55±4,42
	cnBI_CNS	9: [6, 7]: 3	80,18±3,27	**80,36±4,54**
	FCBF	8: [6, 7]: 3	79,18±5,17	**80,61±3,12**
	BestFirst_CFS	12: [6, 7]: 3	78,19±3,88	**80,90±5,71**
Parkinsons	-	22: [6, 7]: 1	73,94±2,43	**78,09±3,51**
	spBI_CFS	5: [6, 7]: 1	78,36±2,86	**78,77±1,66**
	cnBI_CNS	6: [6, 7]: 1	**80,13±2,26**	80,06±3,73
	FCBF	4: [6, 7]: 1	82,52±2,92	**82,79±2,50**
	BestFirst_CFS	6: [6, 7]: 1	**79,25±2,15**	76,05±3,47
Pima	-	8: [4, 5]: 1	78,38±1,59	**79,21±1,53**
	spBI_CFS	3: [4, 5]: 1	79,35±1,09	**79,72±1,08**
	cnBI_CNS	5: [4, 5]: 1	78,52±0,80	**78,54±1,37**
	FCBF	4: [4, 5]: 1	78,42±1,35	**79,53±0,98**
	BestFirst_CFS	4: [4, 5]: 1	78,42±1,35	**79,53±0,98**
Plates	-	27: [6, 7]: 6	50,74±4,24	**51,46±3,03**
	spBI_CFS	16: [6, 7]: 6	**53,81±3,99**	53,38±4,17
	cnBI_CNS	21: [6, 7]: 6	**56,93±2,43**	54,40±4,95
	FCBF	6: [6, 7]: 6	**52,61±5,09**	51,18±5,35
	BestFirst_CFS	10: [6, 7]: 6	50,87±4,75	**51,87±3,06**
Promoter	-	114: [11, 12]: 1	65,76±8,99	**68,20±9,52**
	spBI_CFS	7: [6, 7]: 1	83,84±3,83	**85,64±4,03**
	cnBI_CNS	7: [6, 7]: 1	**80,00±2,74**	76,30±4,10
	FCBF	11: [6, 7]: 1	73,66±6,77	**75,12±4,48**
	BestFirst_CFS	10: [6, 7]: 1	**74,74±5,11**	73,97±3,73
SPECTF	-	44: [6, 7]: 1	60,17±4,15	**61,56±4,97**
	spBI_CFS	12: [6, 7]: 1	73,20±2,18	**73,85±2,71**
	cnBI_CNS	9: [6, 7]: 1	**72,07±1,16**	71,64±1,56
	FCBF	6: [6, 7]: 1	**73,99±1,30**	70,60±1,84
	BestFirst_CFS	12: [6, 7]: 1	72,35±1,69	**73,76±1,02**
Vowel	-	11: [6, 7]: 10	45,04±2,93	**47,18±4,03**

	spBI_CFS	3: [6, 7]: 10	48,07±3,11	**54,31±2,29**
	cnBI_CNS	9: [6, 7]: 10	**47,65±5,01**	46,80±4,27
	FCBF	7: [6, 7]: 10	48,12±3,40	**49,45±2,54**
	BestFirst_CFS	3: [6, 7]: 10	48,07±3,11	**54,31±2,29**
Waveform	-	40: [3, 4]: 2	**84,46±0,92**	82,01±1,48
	spBI_CFS	14: [3, 4]: 2	86,35±0,85	**86,89±0,71**
	cnBI_CNS	15: [3, 4]: 2	**86,02±2,16**	85,67±0,96
	FCBF	5: [3, 4]: 2	79,96±0,47	**80,67±0,37**
	BestFirst_CFS	14: [3, 4]: 2	86,35±0,85	**86,89±0,71**
Winequality-red	-	11: [6, 7]: 5	60,95±1,58	**61,11±1,02**
	spBI_CFS	5: [4, 5]: 5	**61,63±1,09**	61,25±1,62
	cnBI_CNS	8: [4, 5]: 5	**61,47±0,95**	60,87±1,29
	FCBF	4: [4, 5]: 5	**61,65±0,95**	60,95±0,91
	BestFirst_CFS	4: [4, 5]: 5	**61,63±1,09**	61,25±1,62
Yeast	-	8: [11, 12]: 9	60,05±1,21	**60,16±1,10**
	spBI_CFS	5: [11, 12]: 9	59,25±1,44	**60,06±1,09**
	cnBI_CNS	7: [11, 12]: 9	**60,78±1,29**	59,43±1,29
	FCBF	6: [11, 12]: 9	**60,78±1,29**	59,43±1,29
	BestFirst_CFS	7: [11, 12]: 9	**58,29±1,18**	57,91±1,32

Tabla 6.34. Resultados de TSEA y TSEAFS.

Del análisis de los resultados, se puede concluir desde un punto de vista puramente descriptivo que el modelo TSEAFS obtiene mejores resultados que TSEA para todos los conjuntos de datos. En la mayoría de los casos, se produce una clara reducción de la desviación típica con TSEAFS, lo que expresa que los modelos de redes obtenidos son más homogéneos comparados con los que se obtienen con TSEA.

6.5.1. Análisis estadístico

Seguidamente, se presenta un análisis estadístico de comparaciones múltiples TSEA y TSEAFS usando tests estadísticos no paramétricos. En primer lugar, aplicamos un test de Iman-Davenport. Si la hipótesis nula es

rechazada, podemos proceder con un test *post-hoc* que, en nuestro caso, es el test de Bonferroni-Dunn [Dunn y Clark, 1974]. Éste compara diferentes métodos frente a un método de control, determinando si los rankings medios difieren en al menos la DC. Los niveles de significación considerados han sido 0,05 para el test de Iman-Davenport y 0,05 y 0,10 para los métodos *post-hoc*.

Los rankings medios de todos los métodos sin (FØ) y con selección de atributos (spBI_CFS, cnBI_CNS, FCBF y BestFirst_CFS), teniendo en cuenta el mejor promedio entre las dos configuraciones consideradas, son 4,33, 2,28, 3,14, 2,72 y 2,53, respectivamente. Según los resultados del test de Iman-Davenport, dado que el estadístico $F_F = 6,03$ es mayor que el valor crítico $F(4,\ 68) = 2,51$ para $\alpha = 0,05$, la hipótesis nula es rechazada.

Por tanto, aplicamos un test *post-hoc* de Bonferroni-Dunn, que compara una serie de métodos con un método de control, determinando si los rankings medios difieren en al menos la DC. En nuestro caso, hacemos una comparación de los métodos, que emplean selección de atributos (spBI_CFS, cnBI_CNS, FCBF y BestFirst_CFS), frente al método de control (FØ), que no usa selección de atributos. La tabla 6.35 refleja los resultados del test de Bonferroni-Dunn, indicando la diferencia de ranking, la DC (con $\alpha = 0,05$ y $\alpha = 0,10$) y el nivel de diferencia significativa detectado.

A continuación, se analizan los resultados de dicho test, que nos permiten afirmar lo siguiente. Hay diferencias significativas entre TSEA aplicando cada uno de los métodos de selección de atributos considerado y sin aplicar tal selección. Los tests estadísticos señalan que el rendimiento de las RNUP mejora significativamente preprocesando el conjunto de datos con cualquiera de los filtros considerados. Sin embargo, spBI_CFS,

FCBF y BestFirst_CFS son mejores respecto al nivel de significación estadística.

FØ frente a	Diferencia de ranking (Método de control – método comparado)	Nivel de significación para el método comparado
spBI_CFS	2,05	*
cnBI_CNS	1,19	°
FCBF	1,61	*
BestFirst_CFS	1,80	*
	$DC_{(\alpha = 0,05)} = 1,32;\ DC_{(\alpha = 0,10)} = 1,18$	

*: Diferencia estadísticamente significativa con $\alpha = 0,05$

°: Diferencia estadísticamente significativa con $\alpha = 0,10$

Tabla 6.35. Valores de diferencias críticas y diferencias de ranking de los modelos TSEA y TSEAFS por medio de un test de Bonferroni-Dunn (FØ es el método de control).

6.5.2. Comparación con otros clasificadores

En la tabla 6.36 se muestra una comparativa de los resultados de la mejor de las configuraciones de TSEAFS aplicando los diferentes filtros considerados frente a otros clasificadores tanto sin selección de atributos como con ella. Para cada filtro, el mejor promedio obtenido aparece en negrita y el segundo mejor en cursiva.

Conjunto de datos	Filtro	Clasificador						
		C4.5	**1-NN**	**SVM**	**PART**	**MLP**	**RBF**	**TSEAFS**
Appendicitis	-	73,08	69,23	84,62	73,08	76,92	74,67	81,66
	spBI_CFS	80,77	69,23	76,92	80,77	78,85	80,00	82,82
	cnBI_CNS	76,92	57,69	76,92	76,92	77,95	77,05	80,89
	FCBF	80,77	80,77	80,77	80,77	80,77	74,49	80,38
	BestFirst_CFS	80,77	65,38	76,92	80,77	79,23	79,36	81,79
Breast	-	70,42	64,79	64,79	69,01	60,80	68,78	65,96
	spBI_CFS	69,01	70,42	66,20	71,83	69,01	67,46	69,85

	cnBI_CNS	69,01	70,42	64,79	69,01	69,01	69,01	69,01
	FCBF	69,01	70,42	64,79	69,01	69,53	67,65	69,10
	BestFirst_CFS	69,01	70,42	66,20	71,83	69,01	67,46	69,01
Breast-t	-	52,00	60,00	52,00	44,00	63,20	61,20	55,33
	spBI_CFS	56,00	52,00	60,00	44,00	65,33	58,67	54,40
	cnBI_CNS	52,00	52,00	64,00	52,00	67,20	61,20	57,87
	FCBF	48,00	48,00	56,00	48,00	65,60	60,40	60,93
	BestFirst_CFS	68,00	56,00	60,00	56,00	65,47	60,67	59,47
Cardiotocography	-	82,71	76,32	83,65	82,52	80,75	81,80	81,69
	spBI_CFS	77,07	81,77	81,20	82,52	81,94	83,40	85,26
	cnBI_CNS	75,19	63,91	75,19	75,00	68,29	65,91	76,71
	FCBF	77,82	81,20	81,20	77,26	80,13	80,50	81,55
	BestFirst_CFS	78,38	80,45	81,39	81,20	80,86	84,12	82,38
Heart	-	70,59	73,53	76,47	73,53	74,85	78,53	77,45
	spBI_CFS	73,53	73,53	76,47	77,94	72,50	78,24	77,69
	cnBI_CNS	72,06	75,00	76,47	75,00	74,85	77,60	78,57
	FCBF	73,53	70,59	77,94	75,00	74,90	76,37	75,34
	BestFirst_CFS	73,53	73,53	76,47	77,94	72,50	78,53	77,69
Hepatitis	-	84,21	86,84	89,47	81,58	84,73	89,30	87,01
	spBI_CFS	84,21	89,47	86,84	84,21	87,28	89,30	90,78
	cnBI_CNS	89,47	84,21	89,47	84,21	84,21	88,42	87,45
	FCBF	89,47	84,21	89,47	86,84	87,72	90,79	91,05
	BestFirst_CFS	84,21	89,47	86,84	84,21	87,28	89,30	90,78
Labor	-	85,71	71,43	78,57	85,71	69,52	71,67	86,90
	spBI_CFS	85,71	71,43	78,57	85,71	64,29	71,43	96,19
	cnBI_CNS	85,71	64,28	78,57	78,57	78,57	64,29	88,33
	FCBF	85,71	78,57	71,43	78,57	71,43	64,29	90,95
	BestFirst_CFS	85,71	64,29	71,43	85,71	57,62	71,43	89,76
Led24	-	65,67	39,43	58,97	55,80	57,48	55,14	51,03
	spBI_CFS	68,10	67,90	67,93	68,50	68,44	67,42	68,30
	cnBI_CNS	68,10	67,90	67,93	68,50	68,44	67,42	68,30
	FCBF	68,10	67,90	67,93	68,50	68,44	67,42	68,30
	BestFirst_CFS	68,10	67,90	67,93	68,50	68,44	67,42	68,30
Lymphography	-	75,68	83,78	91,89	75,68	86,58	70,99	79,37
	spBI_CFS	88,29	78,38	83,78	70,27	73,24	68,92	79,09

	cnBI_CNS	75,68	70,27	78,38	64,86	71,89	75,77	80,36
	FCBF	81,08	75,68	81,08	70,27	74,50	69,64	80,61
	BestFirst_CFS	81,08	81,08	81,08	64,86	80,45	69,16	80,90
Parkinsons	-	71,43	77,55	75,51	75,51	77,62	70,27	78,09
	spBI_CFS	75,51	79,59	75,51	77,55	81,56	77,75	78,77
	cnBI_CNS	79,59	79,59	75,51	81,63	75,65	73,47	80,13
	FCBF	81,63	73,47	79,59	77,55	84,83	80,27	82,79
	BestFirst_CFS	73,47	81,63	75,51	79,59	83,13	77,55	79,25
Pima	-	74,48	73,96	78,13	74,48	75,94	77,34	79,21
	spBI_CFS	76,04	74,48	77,60	76,04	78,18	79,17	79,72
	cnBI_CNS	74,48	67,19	78,65	74,48	76,89	75,64	78,54
	FCBF	76,04	67,71	79,17	76,04	79,01	80,28	79,53
	BestFirst_CFS	76,04	67,71	79,17	76,04	78,73	80,28	79,53
Plates	-	39,05	49,17	57,02	46,69	53,50	59,94	51,46
	spBI_CFS	40,50	51,24	51,03	46,90	56,71	64,08	53,81
	cnBI_CNS	38,22	50,62	55,17	44,63	55,24	62,17	56,93
	FCBF	44,63	43,18	45,04	49,79	52,85	55,88	51,87
	BestFirst_CFS	54,75	47,31	51,65	51,65	57,33	59,88	48,84
Promoter	-	69,23	65,38	88,46	53,85	86,03	79,36	68,20
	spBI_CFS	73,08	57,69	84,62	80,77	84,49	83,46	85,64
	cnBI_CNS	80,77	57,69	84,62	76,92	75,64	85,00	80,00
	FCBF	73,08	76,92	73,08	80,77	78,21	79,74	75,12
	BestFirst_CFS	73,08	69,23	73,08	80,77	76,28	80,00	74,74
SPECTF	-	67,91	61,50	72,19	70,59	71,28	76,19	61,56
	spBI_CFS	66,84	59,36	72,19	72,19	73,67	76,24	73,85
	cnBI_CNS	65,78	60,96	70,05	65,78	70,02	74,60	72,07
	FCBF	67,91	59,36	65,24	64,71	69,57	74,58	73,99
	BestFirst_CFS	66,84	57,75	73,26	70,05	72,26	74,63	73,76
Vowel	-	39,39	48,48	45,45	38,53	45,87	47,25	47,18
	spBI_CFS	45,24	46,54	54,33	44,59	52,79	43,12	54,31
	cnBI_CNS	38,53	51,52	48,48	40,04	52,05	44,73	47,65
	FCBF	41,56	46,97	41,34	36,58	44,97	46,95	49,45
	BestFirst_CFS	45,24	46,54	54,33	44,59	52,79	43,12	54,31
Waveform	-	74,80	68,96	86,24	76,88	84,85	87,29	84,46
	spBI_CFS	74,40	75,36	86,88	77,04	83,21	82,24	86,89

	cnBI_CNS	74,40	76,64	87,12	79,68	86,27	82,22	86,02
	FCBF	74,72	69,12	78,80	74,00	77,57	76,88	80,67
	BestFirst_CFS	74,40	75,36	86,88	77,04	83,21	82,24	86,89
Winequality-red	-	53,85	49,88	59,55	51,36	56,35	57,11	61,11
	spBI_CFS	50,87	48,88	59,80	52,11	59,36	59,00	61,63
	cnBI_CNS	50,12	49,63	58,81	52,85	57,04	59,19	61,47
	FCBF	51,36	50,37	59,31	49,13	59,64	59,17	61,65
	BestFirst_CFS	50,87	48,88	59,80	52,11	59,36	59,00	61,63
Yeast	-	54,84	48,39	55,91	56,72	59,94	58,31	60,16
	spBI_CFS	53,49	48,92	54,03	54,84	60,20	58,48	60,06
	cnBI_CNS	54,03	49,46	54,84	54,30	60,20	58,91	60,78
	FCBF	52,69	48,12	51,61	52,96	58,96	58,78	58,29
	BestFirst_CFS	54,03	49,46	54,84	54,30	60,20	58,91	60,78
Promedio	-	66,95	64,92	72,16	65,86	70,34	70,29	69,88
	spBI_CFS	68,81	66,46	*71,88*	69,32	71,73	71,58	**74,39**
	cnBI_CNS	67,78	63,83	*71,39*	67,47	70,52	70,14	**72,84**
	FCBF	68,73	66,25	69,10	67,54	*71,03*	70,23	**72,87**
	BestFirst_CFS	69,86	66,25	70,93	69,84	*71,34*	71,28	**73,32**

Tabla 6.36. Comparación de los resultados obtenidos con TSEAFS frente a otros clasificadores.

La aplicación de la selección de atributos permite que el método TSEAFS obtenga mejores resultados que los restantes clasificadores. En dicho clasificador, los 2 mejores filtros son, en este orden, spBI_CFS y BestFirst_CFS. Los filtros spBI_CFS y BestFirst_CFS logran el mejor rendimiento en MLP y RBF. En SVM, el filtro spBI_CFS en algunos conjuntos de datos mejora el rendimiento, aunque en términos globales no lo incrementa; no obstante, permite simplificar el número de variables que intervienen en el modelo. En C4.5 y PART, el filtro más adeacuado es BestFirst_CFS, que hace que el resultado medio entre un clasificador y otro sea prácticamente indistinguible. En 1-NN el mejor rendimiento se obtiene

con spBI_CFS, aunque las diferencias son mínimas frente a otros filtros como FCBF y BestFirst_CFS.

En relación a los restantes filtros que tienen un rendimiento inferior a spBI_CFS y BestFirst_CFS, cabe indicar que el filtro FCBF, salvo con TSEAFS, no logra mejorar el rendimiento medio del clasificador más de 2 puntos, considerando los conjuntos de datos originales. Por otra parte, el filtro basado en consistencia, cnBI_CNS, sólo consigue mejoras destacables en TSEAFS; en PART, el rendimiento es similar al que se alcanza con FCBF. En los métodos PART, MLP, RBF y TSEAFS, los filtros cnBI_CNS y FCBF presentan un comportamiento medio similar. La figura 6.8 muestra los resultados de la tabla 6.36 mediante un gráfico de barras con los resultados medios globales de cada clasificador y filtro, agrupados por clasificador. El símbolo "-" indica que no se ha aplicado ningún selector de atributos.

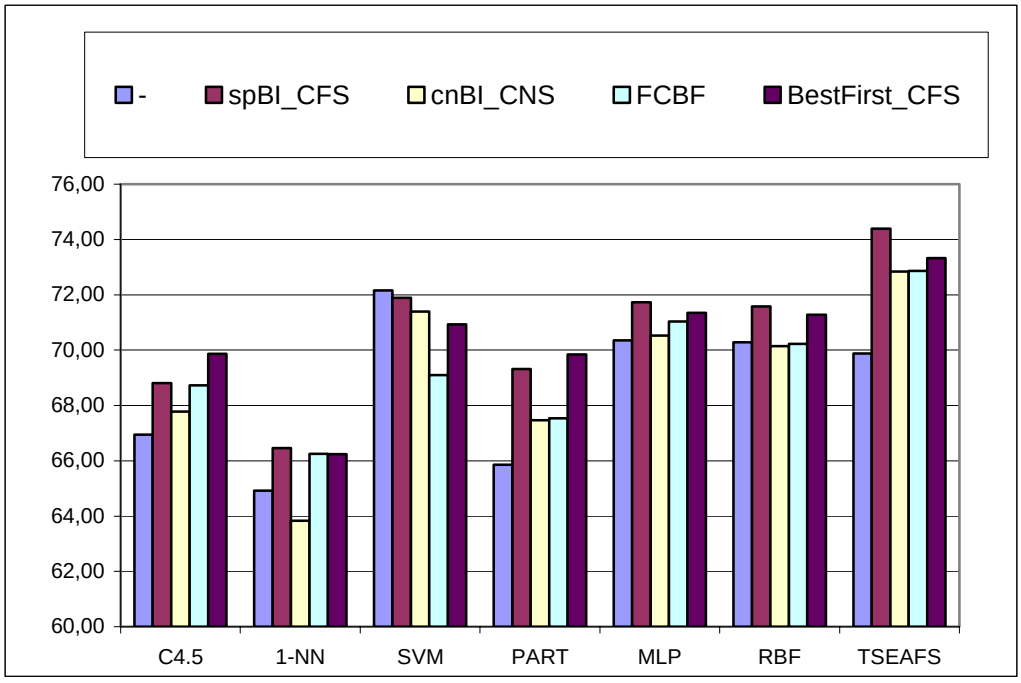


Figura 6.8. Gráfica de resultados globales obtenidos por TSEAFS y otros clasificadores.

6.5.3. Aplicación a Liver-transplantation

El problema *Liver-transplantation* pertenece al área de Biomedicina de Trasplantes. Más concretamente, se trata de un caso de trasplante hepático, relativo a clasificación binaria. Se ha llevado a cabo un análisis retrospectivo de varios centros sanitarios de 11 unidades de trasplante hepático en España, entre ellos, el Hospital Universitario Reina Sofía de Córdoba y el Hospital Universitario Virgen del Rocío de Sevilla, incluyendo todos los trasplantes consecutivos de hígado realizados entre el 1 de enero de 2007 y el 31 de diciembre de 2008. Se incluyeron todos los receptores de 18 años o más.

Las características del receptor y donante fueron recopiladas en el momento del trasplante. Para cada par donante-receptor se dispone de 19 características del receptor, 20 del donante y 3 factores de la operación. La variable objetivo para la clasificación es la supervivencia o no durante los tres meses siguientes a su implante. Un total de 1.031 trasplantes de hígado fueron incluidos inicialmente. El período de seguimiento fue realizado sobre 1.003 trasplantes de hígado; 28 casos fueron excluidos debido a la ausencia de datos de supervivencia del injerto.

En nuestro caso particular, el modelo de aceptación consiste una red neuronal basada en UP para predecir la probabilidad de supervivencia o no. El análisis de los datos detectó inconsistencia en algunas instancias, que se eliminaron. El conjunto de datos resultante contiene 615 instancias, las cuales están desbalanceadas con un ratio 1:8. El número de atributos es 39, que se convierten en 53 entradas, debido a la transformación de las variables nominales. El diseño experimental seguido fue un *hold-out* estratificado, con 462 y 153 instancias en los conjuntos de entrenamiento y generalización, respectivamente.

Como consecuencia del desbalanceo, este conjunto de datos puede generar modelos distorsionados para muchos algoritmos de aprendizaje para los cuales a) el impacto de algunos factores puede estar oculto y b) la precisión de predicción puede ser engañosa. Esto se debe al hecho de que la mayoría de algoritmos de minería de datos asumen conjuntos de datos balanceados. Cuando se trabaja con conjuntos de datos desbalanceados hay dos alternativas: i) técnicas de muestreo o balanceo, esto es, algoritmos de sobremuestreo con el propósito de balancear la distribución de clases aumentando la clase minoritaria o algoritmos de submuestreo, que eliminan instancias de la clase mayoritaria, o ii) aplicar algoritmos que sean robustos a este problema.

Para medir el rendimiento del clasificador, en este problema también hemos considerado la mínima sensibilidad (del inglés *Minimum Sensitivity*, MS) del conjunto de generalización (MS_{gen}) o, en su caso, el subconjunto. Es muy importante evaluar correctamente las instancias de la clase minoritaria, esto es, la clase de no supervivencia. Un buen clasificador debe lograr un CCR alto y clasificar correctamente tantas instancias como sea posible de la clase minoritaria. Por ejemplo, un clasificador puede reconocer todas las instancias de la clase mayoritaria y ninguna de la clase minoritaria, por consiguiente, el CCR será muy alto.

En primer lugar, para intentar balancear el conjunto de datos *Liver-transplantation*, hemos empleado SMOTE (del inglés *Synthetic Minority Over-sampling TEchnique*) [Chawla et al., 2002] sobre el conjunto de entrenamiento. A continuación, la selección de atributos se aplica sobre el nuevo conjunto de entrenamiento y, una vez que hemos obtenido la lista de atributos seleccionados aplicando cada filtro, usamos sólo estas características para obtener el correspondiente conjunto reducido de

generalización a partir del conjunto original de generalización. El número de atributos de *Liver-transplantation* sin y con selección de atributos, así como el porcentaje de reducción de entradas con los diferentes filtros aparecen en la tabla 6.37.

Conjunto de datos: Liver-transplantation	**Filtro**				
	-	**spBI_CFS**	**cnBI_CNS**	**FCBF**	**BestFirst_CFS**
Entradas	53	13	11	7	12
Reducción_Entradas (%)		75,47	79,25	86,79	77,36

Tabla 6.37. Número de entradas sin y con selección de atributos y porcentaje de reducción de entradas para el problema *Liver-transplantation* con TSEA y TSEAFS.

En este problema hemos experimentado con la primera configuración de TSEA y de TSEAFS. La tabla 6.38 muestra los valores de los parámetros.

TSEA			**TSEAFS**		
Configuración 1*			**Configuración 1*#**		
Número máximo de neuronas en cada población (*neu* y *neu+1*)	**Número máximo de generaciones en cada población (0,1**gen*) (Fase 1)**	α_2	**Número máximo de neuronas en cada población (*neu#* y *neu#+1*)**	**Número máximo de generaciones en cada población (0,1**gen#*) (Fase 1)**	α_2
6 y 7	50	1	6 y 7	30	1

Tabla 6.38. Valores de los parámetros de las configuraciones de TSEA y TSEAFS en el problema *Liver-transplantation*.

Los resultados del problema *Liver-transplantation* se exponen en la tabla 6.39.

Clasificador	Métrica	Filtro				
		-	spBI_CFS	cnBI_CNS	FCBF	BestFirst_CFS
C4.5	CCR_{gen}	89,54	89,54	89,54	89,54	89,54
	MS_{gen}	0,00	0,00	0,00	0,00	0,00
1-NN	CCR_{gen}	73,78	85,62	82,35	32,06	86,92
	MS_{gen}	0,00	18,75	**31,25**	68,75	25,00
SVM	CCR_{gen}	89,54	89,54	89,54	89,54	89,54
	MS_{gen}	0,00	0,00	0,00	0,00	0,00
PART	CCR_{gen}	80,39	87,58	88,89	89,54	84,96
	MS_{gen}	0,00	0,00	12,50	0,00	6,25
MLP	CCR_{gen}	83,70	85,86	87,67	89,54	87,32
	MS_{gen}	4,16	4,38	10,83	0,00	4,38
RBF	CCR_{gen}	89,54	88,89	89,39	89,52	89,37
	MS_{gen}	0,00	0,00	1,25	0,00	0,80
TSEAFS	CCR_{gen}	88,69	87,27	**89,71**	89,54	89,25
	MS_{gen}	2,91	9,37	11,46	9,58	9,75

Tabla 6.39. Resultados de TSEAFS y otros clasificadores en el problema *Liver-transplantation* sin y con selección de atributos.

TSEAFS obtiene el mejor CCR_{gen} con el filtro cnBI_CNS (89,71%) y su MS_{gen} es 11,46%, lo que significa que se incrementa el número de instancias de la clase minoritaria bien clasificadas. Con ese mismo filtro, el clasificador 1-NN obtiene el mejor resultado de MS_{gen}, 31,25%, sin embargo, su CCR_{gen} es 82,35%. Entre ambos clasificadores es preferible TSEAFS, dado que tiene un mayor CCR_{gen} y la mejora en MS_{gen} como consecuencia de la aplicación del filtro no compromete el CCR_{gen} y, además, es un clasificador mucho más robusto para el problema en cuestión, pues se observa un comportamiento mucho más estable al aplicar los restantes filtros. En 1-NN la selección de atributos permite mejorar el CCR_{gen} y clasificar correctamente alguna instancia de la clase minoritaria. No obstante, las diferencias entre CCR_{gen} y MS_{gen} entre los diversos filtros son muy grandes y se observa que, partiendo de un CCR_{gen}

de 73,78%, es posible obtener resultados tan dispares como CCR_{gen}=32,06% y MS_{gen}=68,75%, aplicando FCBF cuando precisamente con ese filtro los otros clasificadores tienen un CCR_{gen} por encima del 89%.

MLP tiene un buen rendimiento con el filtro cnBI_CNS, logrando una MS_{gen} de 10,83% y un valor CCR_{gen} de 87,67%. Esto pone de manifiesto que los clasificadores basados en redes neuronales de tipo global, como las RNUS y las RNUP, son muy adecuados para este problema real. Algunos clasificadores, como C4.5 y SVM, no son capaces de asignar correctamente ninguna instancia de la clase minoritaria lo que se traduce en una MS_{gen}=0%, tanto aplicando como sin aplicar filtros. La selección de atributos ayuda al modelo TSEAFS a clasificar correctamente un mayor número de instancias de la clase minoritaria, manteniendo el valor de CCR_{gen} en un nivel por encima del 89% en 3 de los 4 filtros.

Capítulo 7

Conclusiones

Se han alcanzado los objetivos propuestos en esta Tesis. Hemos llevado a cabo una revisión del estado del arte sobre Redes Neuronales Artificiales y sus métodos de entrenamiento. En el área de Inteligencia Computacional, las redes neuronales se utilizan con mucha frecuencia para obtener modelos y poder realizar predicciones del comportamiento de un determinado sistema, conocidos unos datos observados que se emplean para el entrenamiento del modelo, que es evaluado con datos de generalización o desconocidos. Para el entrenamiento de las redes neuronales, puede resultar muy útil guiar el proceso de búsqueda mediante algoritmos de Computación Evolutiva.

De acuerdo al segundo objetivo, se han propuesto diferentes metodologías de entrenamiento de modelos de RNAEs para tareas de clasificación. Nuestras aportaciones son cuatro: EDD, TSEA, EDDFS y TSEAFS. Se ha propuesto un primer modelo denominado EDD, que trabaja con unidades producto, del que podemos afirmar que los parámetros más relevantes son el número de nodos de la capa oculta (parámetro *neu*) y el valor que actúa sobre los coeficientes de la capa de salida del modelo de red neuronal (parámetro α_2). En la ampliación del modelo a unidades sigmoide, EDDSig, se pone de manifiesto que el

número de nodos de tipo unidad sigmoide en la capa oculta influye decisivamente. En general, las RNUP son algo más precisas que las RNUS. Analizando individualmente los conjuntos de datos, existen diferencias puntuales entre EDD y EDDSig. En comparación con otros clasificadores, el rendimiento global de EDD es superior y obtiene el mejor resultado en un mayor número de conjuntos de datos.

El segundo modelo desarrollado para unidades producto, TSEA, es un refinamiento del primer modelo, siendo un 34%, en media, más eficiente y significativamente más preciso. Al comparar TSEA con otros clasificadores, el rendimiento medio de éste es superior. De manera individual, obtiene el mejor resultado en mayor número de ocasiones, con una distancia sobre el segundo mejor clasificador de casi el doble. De la comparación entre TSEA y su ampliación a unidades sigmoide, TSEASig, se desprende que no hay un claro dominio en cuanto a precisión media de ninguno de los dos modelos, si bien existen diferencias en precisiones individuales en los conjuntos de datos. En la comparación por pares entre TSEA, EDDSig y TSEASig, la precisión global de TSEASig es significativamente superior a EDDSig, mientras que cuantitativamente la precisión de TSEA es superior a la de TSEASig. Con respecto a la eficiencia, TSEASig es más rápido que TSEA y considerablemente más rápido que EDDSig. La comparación múltiple con otros clasificadores revela que TSEA y TSEASig, en este orden, consiguen la mejor precisión.

Siguiendo lo indicado en el tercer objetivo, hemos añadido preprocesamiento de datos mediante SA, basada en filtros, en los modelos anteriores, EDD y TSEA, y hemos determinado los filtros más apropiados. Los modelos resultantes son EDDFS y TSEAFS. En EDDFS los tres filtros con el mejor ranking en precisión son BestFirst_CFS, spBI_CFS y cnBI_CNS, seguido exaequo por spBI_CNS e InfoGain. Tanto los filtros

basados en correlación, a excepción de FCBF, como en consistencia demuestran ser muy útiles en el preprocesamiento mediante SA con diferencias significativas según los tests estadísticos no paramétricos. La reducción de entradas oscila entre un 36% y un 69%, lo que implica una reducción del tiempo de computación, logrando modelos de clasificación más sencillos.

En el cuarto modelo, TSEAFS, la precisión de las RNUP mejora significativamente con cualquiera de los cuatro filtros considerados. No obstante, según los tests estadísticos no paramétricos, los filtros con el mejor ranking son, por este orden, spBI_CFS, BestFirst_CFS y FCBF y todos presentan un nivel superior de diferencia significativa. El número de entradas de los modelos de RNUP con SA se reduce más del 50%, con lo cual la disminución de la complejidad computacional y la mayor sencillez es obvia. TSEAFS supera a los restantes clasificadores con y sin SA.

Hemos aplicado las metodologías propuestas a dos problemas del mundo real atendiendo al cuarto objetivo. En el problema *Listeria*, del área de Microbiología Predictiva, en cuanto a precisión, TSEA es superior a TSEASig, EDD y EDDSig, mientras EDD es superior a EDDSig y TSEASig es superior a EDDSig. En el problema de Biomedicina de Trasplantes, *Liver-transplantation* (trasplantes hepáticos), en el que hemos aplicado TSEA y TSEAFS, con TSEA la Mínima Sensibilidad es mayor que cero, lo que se traduce en que algunas instancias de la clase minoritaria son clasificadas correctamente. Esta situación también se produce únicamente con el clasificador MLP, mientras que con los restantes clasificadores no existe ningún acierto. TSEA tiene una precisión ligeramente inferior al mejor resultado obtenido con otros clasificadores. Entre TSEA y TSEAFS, la precisión de este último es superior con tres filtros y la Mínima

Sensibilidad aumenta considerablemente con cualquiera de los cuatro filtros.

En síntesis, las conclusiones globales más relevantes son las siguientes:

1) El número de neuronas en la capa oculta y α_2 influyen en la precisión de las RNUP mediante EDD.
2) El uso de dos poblaciones, cada una con individuos con un número diferente de neuronas en la capa oculta, mejora la diversidad del proceso evolutivo y contribuye a obtener mejores soluciones de forma más eficiente.
3) El AE en dos fases, basado en el concepto de diversidad, es más eficaz y eficiente que el AE básico tanto en RNUS como en RNUP, esto es, TSEA y TSEASig son preferibles a EDD y EDDSig, respectivamente.
4) Las RNUP presentan una mayor precisión que las RNUS, mientras que las RNUS son más eficientes.
5) La SA combinada con RNUP permite mejorar la eficiencia y la eficacia de los dos primeros modelos (EDD y TSEA).
6) En el modelo EDDFS, los filtros significativamente más precisos son los basados en correlación, a excepción de FCBF, los basados en consistencia e InfoGain.
7) En el modelo TSEAFS los filtros más precisos con un nivel de significación superior son los basados en correlación.
8) La técnica de rebalanceo SMOTE y la aplicación de TSEAFS han permitido mejorar la Mínima Sensibilidad, manteniendo un buen nivel de precisión del problema real de Biomedicina de Trasplantes, *Liver-transplantation*.

En cuanto a futuras líneas de investigación abiertas a raíz de esta Tesis Doctoral figuran, entre otras, las siguientes: emplear otros tipos de unidades en la capa oculta y evaluar los diferentes modelos propuestos; extender los modelos que emplean SA a unidades de tipo sigmoide; ampliar la experimentación, considerando más problemas del mundo real tan actuales y relevantes como los de tipo medioambiental y biomédico y, por último, usar el conocimiento disponible en cuanto a la topología adecuada para los diferentes conjuntos de datos con los distintos modelos y tratar de optimizar alguno de los restantes parámetros.

Bibliografía

A continuación, se recoge la bibliografía que se ha consultado en la realización de esta Tesis Doctoral. Aparece el nombre de la referencia y una descripción en la que se detallan los autores, título y algunos datos de interés adicionales.

[Aarts y van Laarhoven, 1989]. Aarts, E. H. L. y van Laarhoven, P. J. M. (1989). Simulated annealing: An introduction. *Statistica Neerlandica*, 43(1), 31-52.

[Abraham, 2004]. Abraham, A. (2004). Metalearning evolutionary artificial neural networks. *Neurocomputing*, 56, 1-38.

[Aha, Kibler y Albert, 1991]. Aha, D., Kibler, D. y Albert, M. K. (1991). Instance-based learning algorithms. *Machine Learning*, 6, 37-66.

[Alba y Martí, 2006]. Alba, E. y Martí, R. (2006). *Metaheuristic Procedures for Training Neural Networks*. Ed. Springer, USA.

[Amor y Rettinger, 2005]. Amor, H. B. y Rettinger, A. (2005). Intelligent Exploration for Genetic Algorithms: Using Self-Organizing Maps in Evolutionary Computation. En: *Proceedings of the 2005 conference on genetic and evolutionary computation* (GECCO 2005). Ed. ACM, Washington, USA, pp. 1531-1538.

[Angeline, Saunders y Pollack, 1994]. Angeline, P. J., Saunders, G. M. y Pollack, J. P. (1994). An Evolutionary Algorithm that Construct Recurrent Neural Networks. *IEEE Transactions on Neural Networks*, 5(1), 54-65.

[Anthony y Bartlett, 2009]. Anthony, M. y Bartlett, P. L. (2009). *Neural Network Learning: Theoretical Foundations*. Ed. Cambridge University Press, New York, USA.

[Azzini y Tettamanzi, 2008]. Azzini, A. y Tettamanzi, A. G. B. (2008). A new genetic approach for neural network design. En: *Engineering Evolutionary Intelligent Systems*. Serie: Studies in Computational Intelligence, vol. 82. Ed. Springer, pp. 289-323.

[Bäck, 1996]. Bäck, T. (1996). *Evolutionary Algorithms in Theory and Practice: Evolution Strategies, Evolutionary Programming, Genetic Algorithms*. Ed. Oxford University Press, New York, USA.

[Battiti y Tecchiolli, 1995]. Battiti, R. y Tecchiolli, G. (1995). Training Neural Nets with the Reactive Tabu Search. *IEEE Transactions on Neural Networks*, 6(5), 1185-1200.

[Beuchat, 1996]. Beuchat, L. R. (1996). Listeria monocytogenes: incidence on vegetables. *Food Control*, 7(4-5), 223-228.

[Bishop, 1991]. Bishop, C. M. (1991). Improving the generalization properties of radial basis function neural networks. *Neural Computation*, 3(4), 579-581.

[Bishop, 1995]. Bishop, M. (1995). *Neural Networks for Pattern Recognition*. Ed. Oxford University Press, New York, USA.

[Bishop, 2006]. Bishop, C. M. (2006). *Pattern Recognition and Machine Learning*. Ed. Springer, Singapore.

[Blum y Langley, 1997]. Blum, A. L. y Langley, P. (1997). Selection of relevant features and examples in machine learning. *Artificial Intelligence*, 97(1-2), 245-271.

[Boese y Kahng, 1993]. Boese, K. D. y Kahng, A. B. (1993). Simulated annealing of neural networks: The cooling strategy reconsidered. En: *Proceedings of the IEEE International Symposium on Circuits and Systems* (ISCAS '93) (vol. 4), pp. 2572-2575.

[Bridle, 1990]. Bridle, J. S. (1990). Probabilistic Interpretation of Feedforward Classification Network Outputs, with Relationships to Statistical Pattern Recognition. En: F. Fogleman Soulie y J. Herault (Eds.). *Neurocomputing: Algorithms, Architectures and Applications*. Ed. Springer-Verlag, Berlin, Germany, pp. 227-236.

[Bryson y Yu-Chi, 1969]. Bryson, A. E. y Yu-Chi, H. (1969). *Applied optimal control: optimization, estimation, and control*. Ed. Blaisdell Publishing Company, Waltham, MA, USA.

[Černý, 1985]. Černý, V. (1985). Thermodynamical approach to the traveling salesman problem: An efficient simulation algorithm. *Journal of Optimization Theory and Applications*, 45(1), 41-51.

[Chandra y Singh, 2004]. Chandra, P. y Singh, Y. (2004). Feedforward Sigmoidal Networks-Equicontinuity and Fault-Tolerance Properties. *IEEE Transactions on Neural Networks*, 15(6), 1350-1366.

[Chawla et al., 2002]. Chawla, N. V, Bowyer, K. W., Hall, L. O. y Kegelmeyer, W. P. (2002). Smote: Synthetic minority over-sampling technique. *Journal of Artificial Intelligence Research*, 16(1), 321-357.

[Chop y Calvert, 2005]. Chop, N. E. y Calvert, D. (2005). The chopper genetic algorithm: A variable population genetic algorithm. En: *Proceedings of the Artificial Neural Networks In Engineering conference* (ANNIE 2005). *Computational Intelligence and Systems Engineering*. Ed. Asme Press, St. Louis, MI, USA.

[Cover, 1965]. Cover, T. (1965). Geometrical and statistical properties of systems of linear inequalities with applications to patterns recognition. *IEEE Trans. Electron. Comput.*, 14, 326-334.

[Cover y Hart, 1967]. Cover, T. y Hart, P. (1967). Nearest neighbor pattern classification. *IEEE Transactions on Information Theory*, 13(1), 21-27.

[Cover y Thomas, 1991]. Cover, T. y Thomas, J. (1991). *Elements of Information Theory*. Ed. Wiley, New York, USA.

[Crainic y Toulouse, 2003]. Crainic, T. G. y Toulouse, M. (2003). Parallel Strategies for Meta-Heuristics. En: F. Glover y G. Kochenberger (Eds.). *Handbook of Metaheuristics*. International Series in Operations Research & Management Science, vol. 57, pp. 475-513.

[Curran y O'Riordan, 2006]. Curran, D. y O'Riordan, C. (2006). Increasing Population Diversity Through Cultural Learning. *Adaptive Behavior*, 14(4), 315-338.

[Cybenko, 1989]. Cybenko, G. (1989). Approximation by Superposition of a Sigmoidal Function. *Mathematics of Control, Signals and Systems*, 2, 303-314.

[Dash y Liu, 2003]. Dash, M. y Liu, H. (2003). Consistency-based search in feature selection. *Artificial Intelligence*, 151(1-2), 155-176.

[Dash y Liu, 1997]. Dash, M. y Liu, H. (1997). Feature selection for classification. *Intelligent Data Analysis*, 1(1-4), 131-156.

[Dash, Liu y Motoda, 2000]. Dash, M., Liu, H. y Motoda, H. (2000). Consistency based feature selection. En: *Proceedings of Pacific-Asia Conference on Knowledge Discovery and Data Mining*, pp. 98-109.

[De Jong, 2006]. De Jong, K. A. (2006). *Evolutionary computation: a unified approach*. Ed. The MIT Press, Cambridge, MA, USA.

[De Werra y Hertz, 1989]. De Werra, D. y Hertz, A. (1989). Tabu Search Techniques: A Tutorial and an Application to Neural Networks. *OR Spektrum*, 11(3), 131-141.

[Demšar, 2006]. Demšar, J. (2006). Statistical comparisons of classifiers over multiple data sets. *Journal of Machine Learning Research*, 7, 1-30.

[Ding y Peng, 2003]. Ding, C. y Peng, H. (2003). Minimum redundancy feature selection from microarray gene expression data. *IEEE Computer Society Bioinformatics*, 523-529.

[Donoho y Johnstone, 1989]. Donoho, D. L. y Johnstone, I. M. (1989). Projection-based approximation and a duality with kernel methods. *The Annals of Statistics*, 17(1), 58-106.

[Dunn y Clark, 1974]. Dunn, O. J. y Clark, V. (1974). *Applied Statistics: Analysis of Variance and Regression*. Ed. Wiley, New York, USA.

[Durbin y Rumelhart, 1989]. Durbin, R. y Rumelhart, D. E. (1989). Product Units: A computationally powerful and biologically plausible extension to backpropagation networks. *Neural Computation*, 1(1), 133-142.

[Durbin y Rumelhart, 1990]. Durbin, R. y Rumelhart, D. E. (1990). Product units with trainable exponents and multi-layer networks. En: F. Fogleman Soulie y J. Herault (Eds.). *Neurocomputing: Algorithms, Architectures and Applications*. Ed. Springer-Verlag, Berlin, Germany, pp. 15-26.

[Eiben y Smith, 2008]. Eiben, A. E. y Smith, J. E. (2008). *Introduction to Evolutionary Computing*. Ed. Springer-Verlag Berlin Heidelberg, Berlin, Germany.

[Elliott, Topiwala y Browne, 2006]. Elliott, P. T., Topiwala, D. y Browne, W. N. (2006). Training reformulated product units in hybrid neural networks. En: *Proceedings of 2006 International Joint Conference on Neural Networks* (IJCNN 2006), pp. 5051-5058.

[Engelbrecht e Ismail, 1999]. Engelbrecht, A. P. e Ismail, A. (1999). Training product unit neural networks. *Stability and Control: Theory and Applications*, 2(1-2), 59-74.

[European Commission, 1999]. European Commission (1999). *Opinion of the Scientific Committee on Veterinary Measures relating to Public Health on Listeria Monocytogenes*. Health & Consumer Protection Directorate-General.

[Fenlon, Wilson y Donachie, 1996]. Fenlon, D. R., Wilson, J. y Donachie, W. (1996). The incidence and level of Listeria monocytogenes contamination of food sources at primary production and initial processing. *Journal of Applied Microbiology*, 81(6), 641-650.

[Fernández Caballero et al., 2010]. Fernández Caballero, J. C., Martínez, F. J., Hervás, C. y Gutiérrez, P. A. (2010). Sensitivity Versus Accuracy in Multiclass Problems Using Memetic Pareto Evolutionary Neural Networks. *IEEE Transactions on Neural Networks*, 21(5), 750-770.

[Fiesler, 1994]. Fiesler, E. (1994). Neural network classification and formalization. *Computer Standards & Interfaces*, 16, 231-239.

[Flach, 2012]. Flach, P. (2012). *Machine Learning: The Art and Science of Algorithms that Make Sense of Data*. Ed. Cambridge University Press, United Kingdom.

[Fogel, Owens y Walsh, 1966]. Fogel, L. J., Owens, A.J. y Walsh, M. J. (1966). *Artificial Intelligence through Simulated Evolution*. Ed. John Wiley & Sons, New York, USA.

[Frank y Asuncion, 2010]. Frank, A. y Asuncion, A. (2010). *UCI Machine Learning Repository* [http://archive.ics.uci.edu/ml]. Irvine, CA: University of California, School of Information and Computer Science.

[Frank y Witten, 1998]. Frank, E. y Witten I. H. (1998). Generating accurate rule sets without global optimization Generating accurate rule sets without global optimization. En: *Proceedings of the Fifteenth International Conference on Machine Learning* (ICML 1998). Ed. Morgan Kaufmann, Madison, Wisconsin, USA, pp. 144-151.

[Friedman, 1937]. Friedman, M. (1937). The use of ranks to avoid the assumption of normality implicit in the analysis of variance. *Journal of the American Statistical Association*, 32(200), 675-701.

[Friedman y Stuetzle, 1981]. Friedman, J. y Stuetzle, W. (1981). Projection pursuit regression. *Journal of the American Statistical Association*, 76(376), 817-823.

[Fu y Min, 1968]. Fu, K. S. y Min, P. J. (1968). *On feature selection in multiclass pattern recognition*. Ed. School of Electrical Engineering, Purdue University, Lafayette, Indiana.

[Fu, Min y Li, 1970]. Fu, K. S., Min, P. J., Li, T. J. (1970). Feature selection in pattern recognition. *IEEE Transactions on Systems Science and Cybernetics*, 6(1), 33-39.

[Fulcher, 2008]. Fulcher, J. (2008). Computational Intelligence: An Introduction. En: J. Fulcher y L. C. Jain (Eds.). *Computational Intelligence: A Compendium*. Serie: Studies in Computational Intelligence, vol. 115, pp. 3-78.

[Funahashi, 1989]. Funahashi, K.-I. (1989). On the Approximate Realization of Continuous Mappings by Neural Networks. *Neural Networks*, 2(3), 183-192.

[García-Pedrajas, Hervás-Martínez y Muñoz-Pérez, 2003]. García-Pedrajas, N., Hervás-Martínez, C. y Muñoz-Pérez, J. (2003). COVNET: A cooperative coevolution model of evolving artificial neural networks. *IEEE Transactions on Neural Networks*, 14(3), 575-596.

[García-Pedrajas, Hervás-Martínez y Muñoz-Pérez, 2002]. García-Pedrajas, N., Hervás-Martínez, C. y Muñoz-Pérez, J. (2002). Multiobjetive cooperative coevolution of artificial neural networks. *Neural Networks*, 15(10), 1255-1274.

[Gelenbe, 1989]. Gelenbe, E. (1989). Random neural networks with positive and negative signals and product form solution. *Neural Computation*, 1(4), 502-510.

[Giles y Maxwell, 1987]. Giles, C.L. y Maxwell, T. (1987). Learning, invariance, and generalization in high-order neural networks. *Applied Optics*, 26(23), 4972-4978.

[Glover, 1986]. Glover, F. (1986). Future Paths for Integer Programming and Links to Artificial Intelligence. *Computer Operations Research*, 13(5), 533-549.

[Glover, 1977]. Glover, F. (1977). Heuristics for integer programming using surrogate constraints. *Decision Sciences*, 8(1), 156-166.

[Glover, 1963]. Glover, F. (1963). Parametric Combinations of Local Job Shop Rules. En: *ONR Research Memorandum*, 117, GSIA, Carnegie Mellon University, Pittsburgh, PA, USA.

[Glover, 1995]. Glover, F. (1995). Scatter search and star-paths: beyond the genetic metaphor. *OR Spektrum*, 17(2-3), 125-137.

[Glover, Laguna y Martí, 2000]. Glover, F., Laguna, M., Martí, R. (2000). Fundamentals of scatter search and path relinking. *Control and Cybernetics*, 39, 653-684.

[Golub et al., 1999]. Golub, T., Slonim, D., Tamayo, P., Huard, C., Gaasenbeek, M., Mesirov, J., Coller, H., Loh, M., Downing, J., Caligiuri, M., Bloomfield, C. y Lander, E. (1999). Molecular classification of cancer: Class discovery and class prediction by gene expression monitoring. *Science*, 286, 531-537.

[Guyon y Elisseeff, 2003]. Guyon, I. y Elisseeff, A. (2003). An introduction to variable and feature selection. *Journal Machine Learning Research*, 3, 1157-1182.

[Hall, 2000]. Hall, M. A. (2000). Correlation-based Feature Selection for Discrete and Numeric Class Machine Learning. En: *Proceedings of the 17th Int. Conf. on Machine Learning* (ICML 2000). Ed. Morgan Kaufmann, San Francisco, CA, USA, pp. 359-366.

[Hall et al., 2009]. Hall, M., Frank, E., Holmes, G., Pfahringer, B., Reutemann, P. y Witten, I. H. (2009). The WEKA data mining software: an update. *ACM SIGKDD Explorations Newsletter*, 11(1), 10-18.

[Hall y Holmes, 2003]. Hall, M. y Holmes, G. (2003). Benchmarking attribute selection techniques for discrete class data mining. *IEEE Transactions on Knowledge and Data Eng.*, 15(3), 1437-1447.

[Han, 2005]. Han, J. (2005). *Data Mining: Concepts and Techniques*. Ed. Morgan Kaufmann Publishers, San Francisco, CA, USA.

[Hansen, 1986]. Hansen, P. (1986). The Steepest Ascent Mildest Descent heuristic for combinatorial programming. En: *Proceedings of Congress on Numerical Methods in Combinatorial Optimization*. Capri, Italy.

[Hastie, Tibshirani y Friedman, 2008]. Hastie, T., Tibshirani, R. y Friedman, J. H. (2008). *The Elements of Statistical Learning*. Ed. Springer, Standford, CA, USA.

[Haykin, 1999]. Haykin, S. (1999). *Neural networks: A Comprehensive Foundation*. Ed. Prentice-Hall, New Jersey, USA.

[Haykin, 2009]. Haykin, S. O. (2009). *Neural Networks and Learning Machines*. Ed. Prentice Hall, Upper Saddle River, New Jersey, USA.

[Hecht-Nielsen, 1989]. Hecht-Nielsen, R. (1989). Theory of the Backpropagation Neural Network. En: *Proceedings of International Joint Conference on Neural Networks* (IJCNN 1989) (vol. 1). Ed. HNC Inc., San Diego, CA, USA, pp. 593-605.

[Heitkoetter y Beasley, 2001]. Heitkoetter, J. y Beasley, D. (2001). *The Hitch-Hiker's Guide to Evolutionary Computation*. En: FAQ in comp.ai.genetic, issue 9.1.

[Herrera y Lozano, 2000]. Herrera, F. y Lozano, M. (2000). Gradual Distributed Real-Coded Genetic Algorithms. *IEEE Transactions on Evolutionary Computation*, 4(1), 43-63.

[Hertz, Krogh y Palmer, 1991]. Hertz, J., Krogh, A. y Palmer, R.G. (1991). *Introduction to the theory of neural computation*. Ed. Addison-Wesley, Reading, MA, USA.

[Hervás et al., 2008]. Hervás, C., Silva, M., Gutiérrez, P. A. y Serrano, A. (2008). Multilogistic regression by evolutionary neural network as a classification tool to discriminate highly overlapping signals: Qualitative investigation of volatile organic compounds in polluted waters by using headspace-mass spectrometric analysis. *Chemometrics and Intelligent Laboratory Systems*, 92, 179-185.

[Hervás-Martínez, Martínez-Estudillo y Gutiérrez, 2006]. Hervás-Martínez, C., Martínez-Estudillo, F. J. y Gutiérrez, P. A. (2006). Classification by means of

Evolutionary Product-Unit Neural Networks. En: *Proceedings of the International Joint Conference on Neural Networks* (IJCNN 2006), pp. 2834-2842.

[Holland, 1975]. Holland, J. (1975). *Adaptation in natural and artificial systems*. Technical report, University of Michigan.

[Hopfield, 1982]. Hopfield, J. J. (1982). Neural networks and physical systems with emergent collective computational abilities. En: *Proceedings of National Academy of Sciences*, 79, pp. 2554-2558.

[Hornik, Stinchcombe y White, 1989]. Hornik, K., Stinchcombe, M. y White, H. (1989). Multilayer Feedforward Networks are Universal Approximators. *Neural Networks*, 2(5), 359-366.

[Hornik, Stinchcombe y White, 1990]. Hornik, K., Stinchcombe, M. y White, H. (1990). Universal approximation of an unknown mapping and its derivatives using multilayer feedforward networks. *Neural Networks*, 3(5), 551-560.

[Howlett y Jain, 2001]. Howlett, R. J. y Jain, L. C. (2001). *Radial Basis Function Networks 1: Recent Developments in Theory and Applications*. Ed. Springer, Heidelberg.

[Huang y Lippmann, 1988]. Huang, W. Y. y Lippmann, R. P. (1988). Neural net and traditional classifiers. En: *Proceedings of Neural Information Processing Systems* (NIPS 1987). Ed. American Institue of Physics, Denver, Colorado, USA, pp. 387-396.

[Huang y Tong, 2009]. Huang, R. y Tong, S. (2009). Evolving product unit neural networks with particle swarm optimization. En: *Proceedings of 2009 Fifth International Conference on Image and Graphics* (ICIG '09). Ed. IEEE Computer Society, Washington, DC, USA, pp. 624-628.

[Iman y Davenport, 1980]. Iman, R. L. y Davenport, J. M. (1980). Approximations of the critical region of the Friedman statistic. *Communications in Statistics*, A9(6), 571-595.

[Ismail y Engelbrecht, 2000]. Ismail, A. y Engelbrecht, A. P. (2000). Global optimization algorithms for training product unit neural networks. En: *Proceedings of the IEEE-INNS-ENNS International Joint Conference on Neural Networks* (IJCNN 2000). *Neural computing: new challenges and perspectives for the new millennium* (vol. 1). Ed. IEEE, Como, Italy, pp. 132-137.

[Ismail y Engelbrecht, 2002]. Ismail, A. y Engelbrecht, A. P. (2002). Pruning product unit neural networks. En: *Proceedings of the 2002 International Joint Conference on Neural Networks* (IJCNN 2002), pp. 257-262.

[Ismail y Engelbrecht, 1999]. Ismail, A. y Engelbrecht, A. P. (1999). Training product units in feedforward neural networks using particle swarm optimization. En: *Proceedings of the 16th International Conference on Artificial Intelligence*, pp. 36-40.

[Janson y Frenzel, 1993]. Janson, D. J. y Frenzel, J. F. (1993). Training product unit neural networks with genetic algorithms. *IEEE Expert*, 8(5), 26-33.

[Kelly, Rangaswamy y Xu, 1996]. Kelly, J. P., Rangaswamy, B. y Xu, J. (1996). A scatter-search-based learning algorithm for neural network training. *Journal of Heuristics*, 2 (2), 129-146.

[Kirkpatrick, Gelatt y Vecchi, 1983]. Kirkpatrick, S., Gelatt, C. D. y Vecchi, M. P. (1983). Optimization by Simulated Annealing. *Science*, 220, 671-680.

[Kittler, 1975]. Kittler, J. (1975). Mathematical methods of feature selection in pattern recognition. *International journal of man-machine studies*, 7(5), 609-638.

[Koch, 1999]. Koch, C. (1999). *Biophysics of Computation: Information Processing in Single Neurons*. Ed. Oxford University Press, New York, USA.

[Koch y Poggio, 1992]. Koch, C. y Poggio, T. (1992). Multiplying with synapses and neurons. En: T. McKenna, J. Davis y S. Zornetzer (Eds.). *Single neuron computation*. Ed. Academic Press, Boston, USA, pp. 315-345.

[Kohavi, 1995]. Kohavi, R. (1995). A Study of Cross-Validation and Bootstrap for Accuracy Estimation and Model Selection. En: *Proceedings of the fourteenth International Joint Conference on Artificial Intelligence* (IJCAI 1995) (vol. 2). Ed. Morgan Kaufman, Montreal, Quebec, Canada, pp. 1137-1145.

[Kohavi y John, 1997]. Kohavi, R. y John, G. (1997). Wrappers for feature subset selection. *Artificial Intelligence*, 1-2, 273-324.

[Kohonen, 1984]. Kohonen, T. (1984). *Self-Organization and Associative Memory*. Ed. Springer-Verlag Berlin, Germany.

[Konar, 2005]. Konar, A. (2005). *Computational Intelligence. Principles, Techniques and Applications*. Ed. Springer-Verlag New York, Inc., Secaucus, New Jersey, USA.

[Koza, 1991]. Koza, J. R. (1991). Evolving a computer program to generate random numbers using the genetic programming paradigm. En: *Proceedings of the 4th International Conference on Genetic Algorithms* (ICGA 1991), pp. 37-44.

[Kramer y Sangiovanni-Vincentelli, 1989]. Kramer, A. H. y Sangiovanni-Vincentelli, A. (1989). Efficient parallel learning algorithms for neural networks. En: David S. Touretzky (Ed.). *Proceedings of Advances in neural information processing systems* (NIPS 1988). Ed. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, pp. 40-48.

[Krasnopolsky y Fox-Rabinovitz, 2006]. Krasnopolsky, V. M. y Fox-Rabinovitz, M. S. (2006). Complex hybrid models combining deterministic and machine learning components for numerical climate modeling and weather prediction. *Neural Networks*, 19(2), 122-134. Special issue "Earth Sciences and Environmental Applications of Computational Intelligence".

[Kuncheva, Del Rio Vilas y Rodríguez, 2007]. Kuncheva, L. I., Del Rio Vilas, V. J. y Rodríguez J. J. (2007). Diagnosing scrapie in sheep: A classification experiment. *Computers in Biology and Medicine*, 37(8), 1194-1202.

[Laguna y Martí, 2003]. Laguna, M. y Martí, R. (2003). *Scatter Search. Methodology and Implementations in C*. Ed. Kluwer, Cambridge.

[Landwehr, Hall y Frank, 2005]. Landwehr, N., Hall, M. y Frank, E. (2005). Logistic Model Trees. *Machine Learning*, 59(1-2), 161-205.

[Larson y Newman, 2011]. Larson, J. y Newman, F. (2011). An implementation of scatter search to train neural networks for brain lesion recognition. *Involve, a Journal of Mathematics*, 4(3), 203-211.

[Le Cun, 1988]. Le Cun, Y. (1988). A Theoretical Framework for Back-Propagation. En: D.Touretzky, G. Hinton y T. Sejnowsky (Eds.). *Proceedings of the 1988 Connectionist Models Summer School*. Ed. Morgan Kaufmann, pp. 21-28.

[Lee et al., 1986]. Lee, Y. C., Doolen, G., Chen, H. H., Sun, G. Z., Maxwell, T., Lee, H. y Giles, C. L. (1986). Machine learning using a higher order correlation network. *Physica D*, 22, 276-306.

[Lee y Hou, 2002]. Lee, S.-H. y Hou, C.-L. (2002). An art-based construction of RBF networks. *IEEE Transactions on Neural Networks*, 13(6), 1308-1321.

[Leerink et al., 1995a]. Leerink, L. R., Giles, C. L., Horne, B. G. y Jabri, M. A. (1995). Learning with products units. En: *Proceedings of Advances in neural information processing systems* (NIPS 1995). Ed. MIT Press, Cambridge, MA, USA, pp. 537-544.

[Leerink et al., 1995b]. Leerink, L. R., Giles, C. L., Horne, B.G. y Jabri, M. A. (1995). *Product unit learning*. Technical Report CS-TR-3503 and UMIACS-TR-95-80. University of Maryland, College Park, MD 20742.

[Leshno et al., 1993]. Leshno, M., Lin, V., Pinkus, A. y Shocken, S. (1993). Multilayer feed-forward networks with a nonpolynomical activation function can approximate any function. *Neural Networks*, 6, 861-867.

[Lewis, 1962]. Lewis, P. M. (1962). The characteristic selection problem in recognition systems. *IRE Transactions on Information Theory*, 8(2), 171-178.

[Lippmann, 1989]. Lippmann, R. P. (1989). Pattern classification using neural networks. *IEEE Communications Magazine*, 11, 47-64.

[Liu et al., 2010]. Liu, H., Motoda, H., Setiono, R. y Zhao, Z. (2010). Feature Selection: An Ever Evolving Frontier in Data Mining. En: JMLR: Workshop and Conference Proceedings. The Fourth Workshop on Feature Selection in Data Mining (FSDM 2010) (vol. 10), Hyderabad, India, pp. 4-13.

[Liu y Motoda, 2008]. Liu, H. y Motoda, H. (2008). *Computational Methods of Feature Selection*. Ed. Chapman & Hall/CRC, USA.

[Liu y Setiono, 1996]. Liu, H. y Setiono, R. (1996). A Probabilistic Approach to Feature Selection - A Filter Solution. En: *Proceedings of the 13th Int. Conf. on Machine Learning* (ICML 1996). Ed. Morgan Kaufmann, pp. 319-327.

[Liu y Yu, 2005]. Liu, H. y Yu, L. (2005). Toward integrating feature selection algorithms for classification and clustering. *IEEE Transactions on Knowledge and Data Engineering*, 17(4), 491-502.

[Maaranen, Miettinen y Mäkelä, 2004]. Maaranen, H., Miettinen, K. y Mäkelä, M. M. (2004). Quasi-random initial population for genetic algorithms. *Computers and Mathematics with Applications*, 47, 1885-1895.

[Martínez-Estudillo et al., 2006a]. Martínez-Estudillo, A., Martínez-Estudillo, F. J., Hervás-Martínez, C. y García-Pedrajas, N. (2006). Evolutionary Product Unit based Neural Networks for Regression. *Neural Networks*, 19, 477-486.

[Martínez-Estudillo et al., 2008]. Martínez-Estudillo, F. J., Hervás-Martínez, C., Gutiérrez, P. A. y Martínez-Estudillo, A.C. (2008). Evolutionary product-unit neural networks classifiers. *Neurocomputing*, 72(1-3), 548–561.

[Martínez-Estudillo et al., 2006b]. Martínez-Estudillo, F. J., Hervás-Martínez, C., Gutiérrez-Peña, P. A., Martínez-Estudillo, A. C. y Ventura, S. (2006). Evolutionary Product-Unit Neural Networks for Classification. En: *Proceedings of the 7th international conference on Intelligent Data Engineering and Automated Learning* (IDEAL 2006). Lecture Notes in Computer Science (LNCS) (vol. 4224). Ed. Springer, Burgos, Spain, pp. 1320-1328.

[McCulloch y Pitts, 1943]. McCulloch, W. W. y Pitts, W. (1943). A Logical Calculus of the Ideas Inminent in Nervous Activity. *Bulletin of Mathematical Biophysics*, 5, 115-133.

[Menon et al., 1996]. Menon, A., Mehrotra, K., Mohan, C. K. y Ranka, S. (1996). Characterization of a Class of Sigmoid Functions with Applications to Neural Networks. *Neural Networks*, 9(5), 819-835.

[Metropolis et al., 1953]. Metropolis, N., Rosenbluth, A. W., Rosenbluth, M., Teller, A. H. y Teller, E. (1953). Equation of State Calculations by Fast Computing Machines. *J. Chem. Phys*., 21, 1087-1092.

[Miller, 1996]. Miller, R. G. (1996). *Beyond ANOVA, Basics of App. Statistics*. Ed. Chapman & Hall, London, UK.

[Miller, 1981]. Miller, R. G. (1981). *Simultaneous Statistical Inference*. Ed. Wiley, New York, USA.

[Miller, Todd y Hegde, 1989]. Miller, G. F., Todd P. M. y Hegde, S. U. (1989). Designing neural networks using genetic algorithms. En: *Proceedings of the third International Conference Genetic Algorithms and Their Applications* (ICGA 1989). Ed. Morgan Kaufman, George Mason University, Fairfax, Virginia, USA, pp. 379-384.

[Mitchell, 1997]. Mitchell, T. M. (1997). *Machine learning*. Ed. McGraw-Hill, New York, USA.

[Mitchison y Durbin, 1989]. Mitchison, G. J. y Durbin, R. M. (1989). Bounds on the learning capacity of some multilayer networks. *Biological Cybernetics*, 60, 345-356.

[Montana y Davis, 1989]. Montana, D. y Davis, L. (1989). Training feedforward neural networks using genetic algorithms. En: *Proceedings of the 11th International Joint Conference on Artificial Intelligence* (IJCAI 1989), pp. 762-767.

[Nemenyi, 1963]. Nemenyi, P. B. (1963). *Distribution-free multiple comparisons*. Tesis Doctoral. Princeton University, USA.

[Newman et al., 1998]. Newman, D. J., Hettich, S., Blake, C. L. y Merz, C. J. (1998). *UCI Repository of machine learning databases*. 1998. Irvine, CA: University of California, Department of Information and Computer Science.

[Nisbet, Elder y Miner, 2009]. Nisbet, R., Elder, J. F. y Miner, G. (2009). *Handbook of Statistical Analysis and Data Mining Applications*. Ed. Academic Press, Canada.

[Ohkura et al., 2007]. Ohkura, K., Yasuda, T., Kawamatsu, Y., Matsumura, Y. y Ueda, K. (2007). MBEANN: Mutation-Based Evolving Artificial Neural Networks. En: *Advances in Artificial Life. Proceedings of the 9th European Conference on Artificial Life* (ECAL 2007). Lecture Notes in Computer Science (LNCS) (vol. 4648). Ed. Springer, Lisbon, Portugal, pp. 936-945.

[Otten y van Ginneken, 1989]. Otten, R. H. J. M. y van Ginneken, L. P. P. P. (1989). *The annealing algorithm*. Ed. Kluwer, Boston, MA, USA.

[Pao, 1989]. Pao, Y. H. (1989). *Adaptative Pattern Recognition and Neural Networks*. Ed. Addison-Wesley Publishing Co., Reading, MA, USA.

[Parker, 1985]. Parker, D. B. (1985). *Learning Logic*. Technical Report TR-47. MIT Center for Research in Computational Economics and Management Science, Cambridge, MA, USA.

[Pitts y McCulloch, 1947]. Pitts, W. y McCulloch, W. W. (1947). How we know universals. *Bulletin of Mathematical Biophysics*, 9, 127-147.

[Prechelt, 1994]. Prechelt, L. (1994). *Proben1—A set of neural network benchmark problems and benchmarking rules*. Technical Report 21/94. Fakultat für Informatik, Univ. Karlsruhe, Karlsruhe, Germany.

[Pyle, 1999]. Pyle, D. (1999). *Data Preparation for Data Mining*. Ed. Morgan Kaufmann Publishers, San Francisco, CA, USA.

[Quinlan, 1993]. Quinlan, J. (1993). *C4.5: Programs for Machine Learning*. Ed. Morgan Kaufmann, San Francisco, CA, USA.

[Rabuñal et al., 2005]. Rabuñal, J. R., Dorado, J., Gestal, M. y Pedreira, N. (2005). Diversity and Multimodal Search with a Hybrid Two-Population GA: An Application to ANN Development. En: *Computational Intelligence and Bioinspired Systems. Proceedings of the 8th International Work-Conference on Artificial Neural Networks* (IWANN 2005). Ed. Springer, Vilanova i la Geltrú, Barcelona, Spain. Lecture Notes in Computer Science (LNCS) (vol. 3512), pp. 199-215.

[Rechenberg, 1989]. Rechenberg, I. (1989). Evolution strategy: Nature's Way of Optimization. En: H. W. Bergmann (Ed.). *Optimization: Methods and Applications, Possibilities and Limitations*. Serie: Lecture Notes in Engineering. Ed. Springer, Bonn, pp. 106-26.

[Rechenberg, 1973]. Rechenberg, I. (1973). *Evolutionsstrategie: Optimierung technischer Systeme nach Prinzipien der biologischen Evolution*. Ed. Frommann-Holzboog, Stuttgart-Bad Canstatt.

[Rocha, Cortez y Neves, 2007]. Rocha, M., Cortez, P. y Neves, J. (2007). Evolution of neural networks for classification and regression. *Neurocomputing*, 70(16-18), 2809-2816.

[Rudin, 1958]. Rudin, W. (1958). *Principles of Mathematical Analysis*. Ed. McGraw-Hill, New York, Toronto, London.

[Ruiz, Riquelme y Aguilar-Ruiz, 2006]. Ruiz, R., Riquelme, J. C. y Aguilar-Ruiz, J. S. (2006). Incremental wrapper-based gene selection from microarray data for cancer classification. *Pattern Recognition*, 39(12), 2383-2392.

[Ruiz, Riquelme y Aguilar-Ruiz, 2002]. Ruiz, R., Riquelme, J. C. y Aguilar-Ruiz, J. S. (2002). Projection-based measure for efficient feature selection. *Journal of Intelligent and Fuzzy Systems*, 12(3-4), 175-183.

[Rumelhart, Hinton y McClelland, 1986]. Rumelhart, D. E., Hinton, G. E. y McClelland, J. L. (1986). A general framework for parallel distributed processing. En: D. E Rumelhart, J. L.McClelland y PDP Research Group (Eds.). *Parallel distributed processing: Explorations in the microstructure of cognition*. Ed. MIT Press, Cambridge, MA, USA, pp. 45-76.

[Rumelhart, Hinton y Williams, 1986]. Rumelhart, D. E., Hinton, G. E. y Williams, R. J. (1986). Learning Internal Representations by Error Propagation. En: D. E

Rumelhart, J. L.McClelland y PDP Research Group (Eds.). *Parallel distributed processing: Explorations in the microstructure of cognition*. Ed. MIT Press, Cambridge, MA, USA, pp. 318-362.

[Rumelhart y McClelland, 1986]. Rumelhart, D. E. y McClelland, J. L. (1986). *Parallel distributed processing: Explorations in the microstructure of cognition*. Ed. MIT Press, Cambridge, MA, USA.

[Saeys, Abeel y de Peer, 2008]. Saeys, Y., Abeel, T. y de Peer, Y. V. (2008). Robust feature selection using ensemble feature selection techniques. En: *Proceedings of the ECML/PKDD (2)*, pp. 313-325.

[Saito et al., 2000]. Saito, K., Ueda, N., Katagiri, S., Fukai, Y., Fujimaru, H. y Fujinawa, M. (2000). Law discovery from financial data using neural networks. En: *Proceedings of IEEE/IAFE/INFORMS Conference on Computational Intelligence for Financial Engineering* (CIFEr 00), pp. 209-212.

[Saito y Nakano, 1997a]. Saito, K. y Nakano, R. (1997). Law discovery using neural networks. En: *Proceedings of the 15th International Joint Conference on Artificial Intelligence* (IJCAI 97), pp. 1078-1083.

[Saito y Nakano, 1997b]. Saito, K. y Nakano, R. (1997). Numeric law discovery using neural networks. En: *Proceedings of the 4th International Conference on Neural Information Processing* (ICONIP 97), pp. 843-846.

[Schaffer, Whitley y Eshelman, 1992]. Schaffer, J. D., Whitley, D. y Eshelman, L. J. (1992). Combinations of Genetic Algorithms and Neural Networks: A Survey of the State of the Art. En: *Proceedings of the Int. Workshop on Combinations of Genetic Algorithms and Neural Networks* (COGANN 92). Ed. IEEE Society press, Los Alamitos, CA, USA, pp. 1-37.

[Schmitt, 2001]. Schmitt, M. (2001). On the Complexity of Computing and Learning with Multiplicative Neural Networks. *Neural Computation*, 14, 241-301.

[Schwefel, 1981]. Schwefel, H. S. (1981). *Numerical Optimization of Computer Models*. Ed. John Wiley & Sons, Inc., New York, USA.

[Sexton et al., 1998]. Sexton, R., Allidae, B., Dorsey, R. y Johnson, J. (1998). Global optimization for artificial neural networks: A tabu search application. *European Journal of Operational Research*, 106(2-3), 570-584.

[Sexton, Dorsey y Johnson, 1999]. Sexton, R., Dorsey, R. y Johnson, J. (1999). Optimization of neural networks: A comparative analysis of the genetic algorithm and simulated annealing. *European Journal of Operational Research*, 114(3), 589-601.

[Shin y Ghosh, 1992]. Shin, Y. and Ghosh, J. (1992). Approximation of Multivariate Functions Using Ridge Polynomial Networks. En: *Proceedings of the International Joint Conference on Neural Networks* (IJCNN 1992) (vol. 2), pp. 380-385.

[Shin y Ghosh, 1991]. Shin, Y. y Ghosh, J. (1991). The Pi-sigma Network: An Efficient Higher-Order Neural Network for Pattern Classification and Function Approximation. En: *Proceedings of the International Joint Conference on Neural Networks* (IJCNN 1992), pp. 13-18.

[Silva y Almeida, 1990]. Silva, F. M. y Almeida, L. B. (1990). Acceleration techniques for the backpropagation algorithm. En: L.B. Almeida and C.J. Wellekens (Eds.). *Proceedings of the 1990 EURASIP Workshop on Neural Networks.* Lecture Notes in Computer Science (LNCS) (vol. 412). Ed. Springer, pp. 110-119.

[Simpson, 1990]. Simpson, P. K. (1990). *Artificial neural systems: Fundations, paradigms, applications and implementations*. Ed. Pergamon Press, New York, USA.

[Stahl y Jordanov, 2012]. Stahl, F. y Jordanov, I. (2012). An overview of the use of neural networks for data mining tasks. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, 2(3), 193-208.

[Sundararajan y Saratchandran, 1998]. Sundararajan, N. y Saratchandran, P. (1998). *Parallel Architectures for Artificial Neural Networks. Paradigms and Implementations*. Ed. IEEE Computer Society, USA.

[Tallón-Ballesteros et al., 2007]. Tallón-Ballesteros, A. J., Hervás-Martínez, C., Gutiérrez P. A. y Jiménez, P. (2007). Distribución de Modelos de Redes Neuronales Evolutivas de Unidades Producto para Clasificación. En: *Actas del V Congreso Español sobre Metaheurísticas, Algoritmos Evolutivos y Bioinspirados* (MAEB 2007), Puerto de la Cruz, Tenerife, Spain, pp. 151-158.

[Tallón-Ballesteros et al., 2012]. Tallón-Ballesteros, A. J., Hervás-Martínez, C., Riquelme, J. C. y Ruiz, R. (2012, en prensa). Feature selection to enhance a two-stage evolutionary algorithm in product unit neural networks for complex classification problems. *Neurocomputing*. DOI: 10.1016/j.neucom.2012.08.041.

[Tallón-Ballesteros et al., 2011]. Tallón-Ballesteros, A. J., Hervás-Martínez, C., Riquelme, J. C. y Ruiz, R. (2011). Improving the accuracy of a two-stage algorithm in evolutionary product unit neural networks for classification by means of feature selection. En: J. M. Ferrández, J. R. Álvarez Sánchez, F. de la Paz y F. J. Toledo (Eds.). *Proceedings of the 4th International work-conference on the Interplay Between Natural and Artificial Computation* (IWINAC 2011). Lecture Notes in Computer Science (LNCS) 6687 (vol. II). Ed. Springer, La Palma (Islas Canarias), Spain, pp. 381-390.

[Tallón-Ballesteros, Gutiérrez-Peña y Hervás-Martínez, 2007]. Tallón-Ballesteros, A. J., Gutiérrez-Peña, P. A. y Hervás-Martínez, C. (2007). Distribution of the search of Evolutionary Product Unit Neural Networks for Classification. En: *Proceedings of the IADIS Internacional Conference Applied Computing* (AC 2007). Ed. IADIS, Salamanca, Spain, pp. 266-273.

[Tallón-Ballesteros y Hervás-Martínez, 2011]. Tallón-Ballesteros, A. J. y Hervás-Martínez, C. (2011). A two-stage algorithm in evolutionary product unit neural networks for classification. *Expert Systems with Applications*, 38(1), 743-754.

[Tallón-Ballesteros, Hervás-Martínez y Gutiérrez, 2013]. Tallón-Ballesteros, A. J., Hervás-Martínez, C. y Gutiérrez, P. A. (2013). An extended approach of a two-stage evolutionary algorithm in artificial neural networks for multiclassification tasks. En: I. Jordanov y L. C. Jain (Eds.). *Innovations in Intelligent Machines – 3: Contemporary Achievements in Intelligent Systems*. Serie: Studies in Computational Intelligence, vol. 442. Ed. Springer-Verlag Berlin Heidelberg, Germany, pp. 139-153.

[Tan, Steinbach y Kumar, 2005]. Tan, P. N., Steinbach, M y Kumar, V. (2005). *Introduction to Data Mining*. Ed. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA.

[Tienungoon et al., 2000]. Tienungoon, S., Ratkowsky, D. A., McMeekin, T. A. y Ross, T. (2000). Growth limits of Listeria monocytogenes as a function of temperature, pH, NaCl, and lactic acid. *Applied and Environmental Microbiology*, 66(11), 4979-4987.

[Torn y Zilinskas, 1987]. Torn, A. y Zilinskas, A. (1987). *Global Optimization*. Ed. Springer-Verlag, New York, USA.

[Ursem, 2002]. Ursem, R. K. (2002). Diversity-Guided Evolutionary Algorithms. En: *Proceedings of Parallel Problem Solving from Nature VII* (PPSN-2002). Ed. Springer Verlag, pp. 462-471.

[Vafaie y De Jong, 1992]. Vafaie, H. y De Jong, K. (1992). Genetic algorithms as a tool for feature selection in machine learning. En: *Proceedings of the 1992 International Conference on Tools with Artificial Intelligence* (TAI 92), pp. 200-203.

[Valero et al., 2007]. Valero, A., Hervás, C., García-Gimeno, R. M. y Zurera, G. (2007). Product unit neural network models for predicting the growth limits of Listeria monocytogenes. *Food Microbiology*, 24(5), 452-464.

[Vapnik, 1995]. Vapnik, V. (1995). *The Nature of Statistical Learning Theory*. Ed. Springer-Verlag, New York, USA.

[Ventura et al., 2008]. Ventura, S., Romero, C., Zafra, A., Delgado, J.A. y Hervás, C. (2008). JCLEC: A Java Framework for Evolutionary Computation. *Soft Computing*, 12(4), 381-392.

[Wang et al., 2008]. Wang, X.-H., Qin, Z., Heng, X.-C. y Liu, Y. (2008). Research on Structure Learning of Product Unit Neural Networks by Particle Swarm Optimization. *Information Technology Journal*, 7(4), 639-646.

[Wang, Zheng y Tang, 2002]. Wang, L., Zheng, D. Z. y Tang, F. (2002). An improved evolutionary programming for optimization. En: *Proceedings of the 4th World Congress on Intelligent Control and Automation* (vol. 3). Ed. IEEE, Shangai, China, pp. 1769-1773.

[Werbos, 1974]. Werbos, P. J. (1974). *Beyond Regression: New Tools for Prediction and Analysis in the Behavioural Sciences*. Tesis Doctoral. Harvard University, Boston, USA.

[Witten, Frank y Hall, 2011]. Witten, I. H., Frank, E. y Hall, M. A. (2011). *Data Mining: Practical Machine Learning Tools and Techniques*. Ed. Morgan Kaufmann, USA.

[Wolpert, 2001]. Wolpert, D. H. (2001). The supervised learning no-free-lunch theorems. En: *Proceedings of the 6th Online World Conference on Soft Computing in Industrial Applications* (WSC6).

[Xing, Jordan y Karp, 2001]. Xing, E., Jordan, M. y Karp, R. (2001). Feature selection for high-dimensional genomic microarray data. En: *Proceedings of the 18th International Conference on Machine Learning* (ICML 2001). Ed. Morgan Kaufmann, San Francisco, CA, USA, pp. 601-608.

[Yao, 1993]. Yao, X. (1993). A review of evolutionary artificial neural networks. *International Journal of Intelligent Systems*, 8(4), 539-568.

[Yao, 1999]. Yao, X. (1999). Evolving artificial neural networks. En: *Proceedings of the IEEE*, 87(9), pp. 1423-1447.

[Yao y Liu, 1997]. Yao, X. y Liu, Y. (1997). A new evolutionary system for evolving artificial neural networks. *IEEE Transactions on Neural Networks*, 8(3), 694-713.

[Yegnanarayana, 1999]. Yegnanarayana, B. (1999). *Artificial Neural Networks*. Ed. Prentice Hall of India, New Delhi, India.

[Yu y Liu, 2004]. Yu, L. y Liu, H. (2004). Efficient feature selection via analysis of relevance and redundancy. *Journal of machine learning research*, 5, 1205-24.

[Zhang, Zhang y Yang, 2003]. Zhang, S., Zhang C. y Yang, Q. (2003). Data preparation for data mining. *Applied Artificial Intelligence*, 17(5-6), 375-381.

[Zurada, 1992]. Zurada, J. M. (1992). *Introduction to Artificial Neural Systems*. Ed. West Publishing Company, Saint Paul, Minnesota, USA.

Anexo

Curriculum vitae

Seguidamente, se exponen los aspectos más relevantes del curriculum vitae del doctorando en cuanto a publicaciones y participación en proyectos de investigación de ámbito estatal y autonómico asociados a la Tesis Doctoral.

A.1. PUBLICACIONES

A.1.1. Artículo en la revista *Expert Systems with Applications*

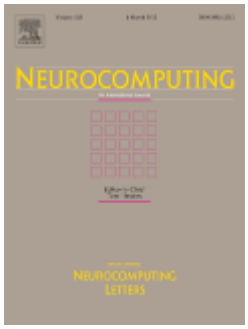
- Datos identificativos
  - Título: A two-stage algorithm in evolutionary product unit neural networks for classification
  - Autores: Antonio J. Tallón-Ballesteros y César Hervás-Martínez
  - Posición del doctorando: 1ª
  - Tipo: artículo de revista internacional (incluida en el JCR)
  - Título de la Revista: Expert Systems with Applications
  - Estado: publicado
  - Año de publicación: 2011

- Volumen: 38
- Número: 1
- Páginas: 743-754
- ISSN: 0957-4174

- Indicios de calidad
  - Factor de impacto: 2,203 Q1
  - Categorías (posición/total). Cuartil:
    - Computer Science, Artificial Intelligence (22/111). Q1
    - Engineering, Electrical & Electronic (41/244). Q1
    - Operations Research & Management Science (5/77). Q1
- URL: http://dx.doi.org/10.1016/j.eswa.2010.07.028

A.1.2. Artículo en la revista *Neurocomputing*

- Datos identificativos
  - Título: Feature selection to enhance a two-stage evolutionary algorithm in product unit neural networks for complex classification problems
  - Autores: Antonio J. Tallón-Ballesteros, César Hervás-Martínez, José C. Riquelme y Roberto Ruiz
  - Posición del doctorando: 1ª
  - Tipo: artículo de revista internacional (incluida en el JCR)
  - Título de la Revista: Neurocomputing
  - Estado actual: en prensa desde 23/10/2012
  - DOI: 10.1016/j.neucom.2012.08.041
  - ISSN: 0925-2312



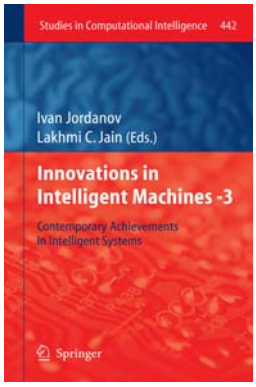
- Indicios de calidad
  - Factor de impacto: 1,580 Q2 (JCR 2011)

- Categoría (posición/total). Cuartil: Computer Science, Artificial Intelligence (39/111). Q2

- URL: http://dx.doi.org/10.1016/j.neucom.2012.08.041

A.1.3. Capítulo del libro *Innovations in Intelligent Machines – 3: Contemporary Achievements in Intelligent Systems*

- Datos identificativos
  - Título: An extended approach of a two-stage evolutionary algorithm in artificial neural networks for multiclassification tasks
  - Autores: Antonio J. Tallón-Ballesteros, César Hervás-Martínez y Pedro A. Gutiérrez
  - Posición del doctorando: 1ª
  - Tipo: capítulo de libro
  - Libro: Innovations in Intelligent Machines – 3: Contemporary Achievements in Intelligent Systems
  - Serie: Studies in Computational Intelligence
  - Editorial: Springer-Verlag Berlin Heidelberg
  - Editores: I. Jordanov y L. C. Jain
  - Estado actual: publicado
  - Año de publicación: 2013
  - Volumen: 442
  - Páginas: 139-153
  - ISBN: 978-3-642-32176-4



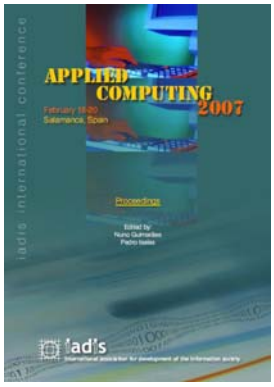
- Indicios de calidad
  - Número de volúmenes en la serie: 455 (desde 2005 hasta la actualidad)

- La serie *Studies in Computational Intelligence* (SCI) publica nuevos desarrollos y avances en las variadas áreas de Inteligencia Computacional.

- URL: http://link.springer.com/chapter/10.1007/978-3-642-32177-1_9

A.1.4. Contribución en actas del congreso Applied Computing (AC 2007)

- Datos identificativos
  - Título: Distribution of the search of Evolutionary Product Unit Neural Networks for Classification
  - Autores: Antonio J. Tallón-Ballesteros, Pedro A. Gutiérrez-Peña y César Hervás-Martínez
  - Posición del doctorando: 1ª
  - Tipo: contribución en actas de congreso internacional
  - Título del Congreso: IADIS Internacional Conference Applied Computing (AC 2007)
  - Editorial: IADIS
  - Editores: N. Guimaraes y P. Isaías
  - Estado actual: publicado
  - Año de publicación: 2007
  - Páginas: 266-273
  - ISBN: 978-972-8924-30-0



- Indicios de calidad
  - Tasa de aceptación: inferior al 22% (50/231)
  - Número de países de origen diferentes: 35
  - Número de edición: 5ª
  - Actas disponibles en Internet con acceso abierto: sí
- URL: http://www.iadisportal.org/digital-library/distribution-of-the-search-of-evolutionary-product-unit-neural-networks-for-classification

A.1.5. Contribución en actas del congreso Interplay Between Natural and Artificial Computation (IWINAC 2011)

- Datos identificativos
  - Título: Improving the accuracy of a two-stage algorithm in evolutionary product unit neural networks for classification by means of feature selection
  - Autores: Antonio J. Tallón-Ballesteros, César Hervás-Martínez, José C. Riquelme y Roberto Ruiz
  - Posición del doctorando: 1ª
  - Tipo: contribución en actas de congreso internacional
  - Título del Congreso: 4th International work-conference on the Interplay Between Natural and Artificial Computation (IWINAC 2011)

  - Editorial: Springer
  - Serie: Lecture Notes in Computer Science (LNCS)
  - Editores: J. M. Ferrández, J. R. Álvarez Sánchez, F. de la Paz y F. J. Toledo
  - Estado actual: publicado
  - Año de publicación: 2011
  - Volumen: 6687
  - Páginas: 381-390
  - ISBN: 978-3-642-21325-0
- Indicios de calidad
  - Serie Lecture Notes in Computer Science (LNCS)
  - Número de edición: 4ª
- URL: http://dx.doi.org/10.1007/978-3-642-21326-7_41

A.1.6. Contribución en actas del congreso Metaheurísticas, Algoritmos Evolutivos y Bioinspirados (MAEB 2007)

- Datos identificativos
  - Título: Distribución de Modelos de Redes Neuronales Evolutivas de Unidades Producto para Clasificación
  - Autores: Antonio J. Tallón-Ballesteros, C. Hervás-Martínez, P. A. Gutiérrez y P. Jiménez
  - Posición del doctorando: 1ª
  - Tipo: contribución en actas de congreso internacional
  - Título del Congreso: V Congreso Español sobre Metaheurísticas, Algoritmos Evolutivos y Bioinspirados (MAEB 2007)
  - Editorial: Thomson
  - Editores: F. Almeida Rodríguez, B. Melián Batista, J. A. Moreno Pérez y J. M. Moreno Vega

  - Estado actual: publicado
  - Año de publicación: 2007
  - Páginas: 151-158
  - ISBN: 978-84-690-3470-5
- Indicios de calidad
  - Número de edición: 5ª
  - Número de ediciones celebradas hasta 2012: 8
  - El congreso MAEB es un foro de encuentro, discusión y transferencia de conocimiento entre investigadores en el campo de las metaheurísticas y los algoritmos bioinspirados, cuyo fin es presentar e intercambiar experiencias y resultados.
  - Idiomas permitidos para las contribuciones: inglés o español

A.2. PROYECTOS DE INVESTIGACIÓN

A.2.1. Participación en el Proyecto de Excelencia "Modelos Avanzados para el Análisis Inteligente de Información. Aplicación a Datos Biomédicos y Medioambientales"

- Título del proyecto: "Modelos Avanzados para el Análisis Inteligente de Información. Aplicación a Datos Biomédicos y Medioambientales"
- Tipo de proyecto: Proyecto de Excelencia de la Junta de Andalucía I+D
- Referencia: TIC-7528
- Tipo de responsabilidad del doctorando: investigador
- Duración: 4 años
- Período: 2012-2015
- Organismo/Empresa financiador/a: Consejería de Innovación, Ciencia y Empresa de la Junta de Andalucía
- Entidades participantes: Universidad de Sevilla
- Investigador responsable: Cristina Rubio Escudero
- Número de investigadores: 7
- Importe concedido: 31.435,25 €

A.2.2. Participación en el Proyecto Nacional "Análisis Inteligente de Información Medioambiental"

- Título del proyecto: "Análisis Inteligente de Información Medioambiental"
- Tipo de proyecto: Plan Nacional del 2011. Proyecto I+D
- Referencia: TIN2011-28956-C02-02
- Tipo de responsabilidad del doctorando: investigador
- Fecha de inicio: 01/10/2012
- Fecha de finalización: 31/12/2014

- Organismo/Empresa financiador/a: Ministerio de Ciencia e Innovación
- Entidades participantes: Universidad de Sevilla
- Investigador responsable: José C. Riquelme Santos
- Número de investigadores: 13
- Importe concedido: 47.008,50 €

A.2.3. Participación en el Proyecto Nacional "Heurísticas escalables para la extracción de conocimiento en grandes volúmenes de información"

- Título del proyecto: "Heurísticas escalables para la extracción de conocimiento en grandes volúmenes de información"
- Tipo de proyecto: Plan Nacional del 2007. Proyecto I+D
- Referencia: TIN2007-68084-C02-02
- Tipo de responsabilidad del doctorando: investigador
- Fecha de inicio: 01/10/2007
- Fecha de finalización: 31/03/2012
- Organismo/Empresa financiador/a: Ministerio de Educación y Ciencia
- Entidades participantes: Universidad de Sevilla
- Investigador responsable: José C. Riquelme Santos
- Número de investigadores: 12
- Importe concedido: 99.220 €