UNIVERSIDAD POLITÉCNICA DE VALENCIA

DEPARTAMENTO DE SISTEMAS INFORMÁTICOS Y COMPUTACIÓN

TESIS DOCTORAL

ARQUITECTURA DE BÚSQUEDA BASADA EN TÉCNICAS SOFT COMPUTING PARA LA RESOLUCIÓN DE PROBLEMAS COMBINATORIOS EN DIFERENTES DOMINIOS DE APLICACIÓN

Autor: Soledad Valero Cubas
Directores: Dr. Vicent J. Botti Navarro
Dra. Estefanía Argente Villaplana

PROGRAMA DE DOCTORADO DE RECONOCIMIENTO DE FORMAS E INTELIGENCIA ARTIFICIAL

VALENCIA
FEBRERO 2010

Fecha: Febrero 2010

Autor: **Soledad Valero Cubas**

Directores: **Dr. Vicent J. Botti Navarro**
Dra. Estefanía Argente Villaplana

Título: **Arquitectura de Búsqueda Basada en Técnicas Soft Computing para la Resolución de Problemas Combinatorios en Diferentes Dominios de Aplicación**

Departamento: **Sistemas Informáticos y Computación**

Universidad: **Universidad Politécnica de Valencia**

Grado: **Doctor** Mes: **Febrero** Año: **2010**

Firma del Autor

Índice general

Índice de figuras

Índice de tablas

Resumen

En los problemas de optimización combinatoria se estudian colecciones finitas de objetos que satisfacen unos criterios específicos y se persigue determinar si cierto objeto "óptimo" existe. En la mayoría de las ocasiones, a pesar de que el dominio de búsqueda es finito, éste puede ser de dimensiones exponenciales. En la actualidad es posible solucionar un gran número de problemas combinatorios presentes en la vida real empleando técnicas basadas en programación entera. Sin embargo, en numerosas ocasiones no es posible resolverlos de forma exacta debido a la gran dificultad que presentan algunos problemas de optimización combinatoria y sólo es posible encontrar soluciones cercanas al óptimo. Para estas ocasiones, los esfuerzos de investigación se han centrado en la aplicación de técnicas meta-heurísticas. En este último caso se enmarca el presente trabajo, es decir, en la resolución de problemas combinatorios complejos, de grandes dimensiones, donde explorar todas las posibilidades a fin de encontrar el óptimo es inabordable, ya sea por motivos económicos (probar cada combinación sea caro) o por motivos computacionales (temporalmente sea intratable).

En concreto, en esta tesis se propone una arquitectura de búsqueda independiente del dominio de aplicación y capaz de abordar problemas combinatorios de grandes dimensiones, de los que se disponga de poca información de partida. Esta arquitectura está basada en técnicas *Soft Computing*, pues combina un algoritmo genético basado en codificación real con modelos basados en redes neuronales, concretamente en perceptrones multicapa. Así, el algoritmo genético emplea, en los casos en los que sea necesario, modelos aproximados de las funciones de aptitud mediante perceptrones diseñados para tal fin. El sistema obtenido ofrece la flexibilidad y versatilidad requeridas para poder adaptarse a los requisitos propios de cada problema combinatorio a tratar, sea cual sea su dominio.

A fin de determinar las técnicas más adecuadas para la arquitecta resultado del presente trabajo, se revisaron las principales técnicas *Soft Computing* actuales. Como resultado de este trabajo pudo constatarse que estas técnicas ofrecen soluciones a bajo coste, robustas y flexibles. Además, cuando actúan de forma combinada potencian sus virtudes minimizando sus desventajas.

Además, la arquitectura de búsqueda *Soft Computing* fruto de la presente tesis fue aplicada a la resolución de problemas combinatorios de interés, tanto en el área de la Catálisis Combinatoria como en el dominio de los Sistemas de Recomendación. Así pues, en un primer paso se estudiaron los requisitos y necesidades de los problemas a resolver dentro del ámbito de ambos dominios. En un segundo paso, la técnica propuesta fue utilizada en el ámbito de la Catálisis Combinatoria tanto para optimizar las condiciones de distintas reacciones, como para determinar las composiciones idóneas de determinados catalizadores para reacciones de naturaleza y complejidad diferentes. Asimismo, la arquitectura de búsqueda planteada fue aplicada en el ámbito de los Sistemas de Recomendación, concretamente sobre un dominio de entretenimiento: la valoración de películas. Para ello se empleó el conjunto de datos *MovieLens*, utilizado habitualmente como *benchmark* en este ámbito. Así, la arquitectura fue utilizada para determinar los perfiles de las preferencias de ciertos usuarios a partir de la información disponible sobre ellos o a partir de la información disponible para otros usuarios similares a ellos.

Finalmente, la arquitectura de búsqueda desarrollada ha sido empleada en la obtención del paquete de aplicaciones o herramientas *SoftCombi*, que permite el diseño inteligente de experimentos en el ámbito de la Catálisis Combinatoria.

Resum

En els problemes d'optimització combinatòria s'estudien col·leccions finites d'objectes que satisfan uns criteris específics i es perseguix determinar si un cert objecte "óptim" existix. En la majoria de les ocasions, a pesar que el domini de recerca és finit, este pot ser de dimensions exponencials. En l'actualitat és possible solucionar un gran nombre de problemes combinatoris en la vida real emprant tècniques basades en programació sencera. No obstant això, en nombroses ocasions no és possible resoldre'ls de forma exacta a causa de la gran dificultat que presenten alguns problemes d'optimització combinatòria i només és possible trobar solucions pròximes a l'òptim. Per a estes ocasions, els esforços d'investigació s'han centrat en l'aplicació de tècniques meta-heurístiques. En aquest últim cas s'emmarca el present treball, és a dir, en la resolució de problemes combinatoris complexos, de grans dimensions, on explorar totes les possibilitats a fi de trobar l'òptim és inabordable, ja siga per motius econòmics (provar cada combinació siga car) o per motius computacionals (temporalment siga intractable).

En concret, en esta tesi es proposa una arquitectura de recerca independent del domini d'aplicació i capaç d'abordar problemes combinatoris de grans dimensions, dels que es dispose poca informació de partida. Esta arquitectura està basada en tècniques Soft Computing, perquè combina un algoritme genètic basat en codificació real amb models basats en xarxes neuronals, concretament en perceptrons multicapa. Així, l'algoritme genétic utilitza, en els casos en què siga necessari, models aproximats de les funcions d'aptitud per mitjà de perceptrons dissenyats per a tal fi. El sistema obtingut oferix la flexibilitat i versatilitat requerides per a poder adaptar-se als requisits propis de cada problema combinatori a tractar, siga quin siga el seu domini.

A fi de determinar les tècniques més adequades per a l'arquitectura resultat del present

treball, es van revisar les principals tècniques Soft Computing actuals. Com resultat d'este treball va poder constatar-se que estes tècniques oferixen solucions a baix cost, robustes i flexibles. A més, quan actuen de forma combinada potencien les seues virtuts minimitzant els seus desavantatges. A més, l'arquitectura de recerca *Soft Computing* fruit de la present tesi es va aplicar a la resolució de problemes combinatoris d'interés, tant en l'àrea de la Catàlisi Combinatòria com en el domini dels sistemes de recomanació. Així, en un primer pas es van estudiar els requisits i necessitats dels problemes a resoldre dins de l'àmbit d'ambdós dominis. En un segon pas, la tècnica proposta va ser utilitzada en l'àmbit de la Catàlisi tant per a optimitzar les condicions de distintes reaccions, com per a determinar les composicions idònies de catalitzadors indicats per a reaccions de naturalesa i complexitat diferents. Així mateix, l'arquitectura de recerca plantejada va ser aplicada en l'àmbit dels Sistemes de Recomanació, concretament a un domini d'entreteniment: la valoració de pel·lícules. Per això es va emprar el conjunt de dades *MovieLens*, utilitzat habitualment com *benchmark* en este àmbit. Així, l'arquitectura es va utilitzar per a determinar els perfils de les preferències de certs usuaris a partir de la informació disponible sobre ells o a partir de la informació disponible per a altres usuaris semblants a ells.

Finalment, l'arquitectura de recerca desenvolupada ha sigut emprada en l'obtenció del conjunt d'aplicacions o ferramentes *SoftCombi*, que permet el disseny intel·ligent d'experiments en l'àmbit de la Catàlisi Combinatòria.

Abstract

In combinatorial optimization problems, finite collections of objects that meet specific criteria are studied in order to determine whether there is any optimal object. In most cases, although the search domain is finite, it can be of exponential size. Nowadays, it is possible to solve many combinatorial problems presented in real life using techniques based on integer programming. However, combinatorial problems are extremely difficult in many cases and only near-optimal solutions are possible. For these occasions, research efforts have focused on the application of meta-heuristic techniques. This is the case of this work, which is focused on solving complex high-dimensional combinatorial problems. In this kind of problems, exploring all possibilities to find the optimum is intractable, either for economic reasons (i.e. testing each combination is very expensive) or for computational reasons (i.e. being temporarilly intractable).

Specifically, this thesis proposes a domain-independent search architecture, which is able to tackle large combinatorial problems, even in situations where there is little starting data. This architecture is based on Soft Computing techniques, combining a genetic algorithm based on real coding with artificial neural network-based models (multilayer perceptrons). Thus, the genetic algorithm makes use of these perceptrons for fitness evaluations, when necessary. The obtained system offers the required flexibility and versatility to be able to tackle with whatever combinatorial problem.

Actual Soft Computing techniques have been reviewed in order to select the most appropriate to the pursued architecture. As a result, it was found that these techniques offer low cost, robust and flexible solutions. Furthermore, when acting in combination, they enhance their strengths while minimizing their disadvantages.

Furthermore, the developed Soft Computing architecture was applied to solve combi-

natorial problems of interest, both in the area of Combinatorial Catalysis and in the domain of Recommender Systems. Firstly, requirements and needs of the problems to be solved within the scope of both domains were considered. Secondly, the proposed technique was used in the field of catalysis in order to optimize the conditions for different reactions, as well as to determine the best catalyst compositions suitable for reactions of different nature and complexity. Also, the proposed search architecture was applied to an entertainment domain in the field of Recommender Systems: the evaluation of films. For this study, the *MovieLens* dataset was employed, which is a well-known benchmark in this field. Thus, the architecture was used to determine the preferences of certain users from the information available about them or from the information available about others users with similar preferences.

Finally, the proposed architecture has been employed as a framework to obtain the *SoftCombi* package, whose tools help in the intelligent design of experiments in the field of Combinatorial Catalysis.

Agradecimientos

En primer lugar, quisiera agradecer a mis directores todo el tiempo y dedicación que me han prestado durante estos años. A D. Vicente Botti le debo toda mi labor investigadora y docente. Sin sus consejos, apoyo y ayuda, nada de lo logrado durante estos últimos años habría sido posible. Debo agradecerle el interés y confianza en mi persona que siempre me ha demostrado. Gracias también por ese don que hace que uno se sienta lleno de tranquilidad, tras comentarle cualquier problema, puesto que siempre sabe qué necesitas oír.

De igual modo, debo agradecer a Dña. Estefanía Argente el gran esfuerzo realizado en revisar con detalle cada uno de los párrafos escritos en esta tesis. Sin su paciencia y buen hacer, el resultado final habría sido, sin duda, menos legible y comprensible. Asimismo, debo agradecerle todos sus consejos, ideas y aportaciones. Sin su inestimable labor, esta tesis no hubiera visto la luz. Gracias también por todos los buenos ratos pasados, puesto que además de mi directora, ha sabido ser siempre mi amiga.

Me faltan las palabras para agradecer a Jorge, mi compañero de viaje, mi mejor amigo y guía, todo el cariño, ánimo y comprensión que me brinda cada día. Gracias por ayudarme a seguir creyendo en mí, incluso en los peores momentos. Gracias también por entender y soportar todo el tiempo "libre" que me roba este trabajo. Además, en el último tramo de este largo viaje llegó a nuestra vida Víctor, a quien debo agradecerle su eterna sonrisa que ilumina mis días. Gracias por no enfadarte porque mamá se encerrara en el despacho cuando querías jugar a los "poches" o a la "pota". Tu personita hace que cada día me levantase con la energía suficiente para acabar cualquier trabajo.

También debo dar las gracias a mis padres, cuyo esfuerzo posibilitó que iniciara este camino.

Muchas gracias a los doctores D. José M. Serra y D. Avelino Corma por prestarme su tiempo y explicarme todas mis dudas acerca de la catálisis combinatoria.

Gracias a todos mis compañeros del grupo de investigación del GTI-IA, del que me siento una privilegiada por pertenecer. Su buen humor y compañerismo hacen de nuestro lugar de trabajo un ambiente tan agradable que las horas en su compañía (y son muchas), pasan sin darte cuenta. En especial quiero agradecer a Luis (Burdi para los amigos) toda la ayuda prestada, que ha sido mucha durante estos años. Gracias por estar siempre ahí cuando se te necesita, ha sido un privilegio conocerte y trabajar a tu lado. Espero que sigas ofreciéndonos buenos ratos con tu guitarra, pues sé que es en esa vertiente de tu vida donde más disfrutas. Gracias también a Martí, por soportarme durante estos años como compañera de mesa. Gracias por esperar con paciencia mis respuestas, puesto que a veces mi retraso puede desesperar. Gracias Nancy, por estar siempre dispuesta a echar una mano. De igual modo, gracias a Natalia y a Miguel, por la ayuda prestada y por sufrirme como compañera de laboratorio, tarea ardua donde las haya. Gracias a Javi, por alegrarme cada día con su "guapa" y dedicarme parte de su tiempo cuando me ha hecho falta.

Finalmente, no puedo olvidarme de la vieja guardia, cuyos consejos y ayuda es inestimable, especialmente: gracias Carlos, por tener siempre tu puerta abierta; gracias Andrés, por tus consejos y ánimos durante estos años; gracias a Miguel y a Emilio, por ayudarme a *desestresarme* con la raqueta cada semana, antes de que mis "rodillas" me lo impidieran; gracias Vicente Julián, ha sido un placer trabajar contigo; gracias a Agus y Ana, por estar ahí cada día; gracias a Miguel y Adriana, por saber ofrecer sus mentes privilegiadas cuando se necesitan; gracias a Óscar, por su alegría y ayuda; gracias a Eli, por ser un encanto desde el primer día; gracias a Eva, por traerme de la mano a este maravilloso grupo de investigación; gracias a todos y cada uno de vosotros.

Capítulo

1

Introducción

1.1. Motivación

En los problemas de optimización combinatoria se estudian colecciones finitas de objetos que satisfacen unos criterios específicos y se persigue determinar si cierto objeto "óptimo" existe. En la mayoría de las ocasiones, a pesar de que el dominio de búsqueda es finito, éste puede ser de dimensiones exponenciales. Numerosos problemas pueden modelarse como problemas de optimización combinatoria, por ejemplo, los problemas abordados por la programación lineal, donde una función lineal $f(X)$ de n variables y el conjunto de soluciones posibles puede representarse por un conjunto de desigualdades lineales. También constituyen problemas combinatorios aquellos que son abordados por la programación entera, donde se imponen restricciones de integridad adicionales sobre algunos subconjuntos de las variables de decisión de un problema de programación lineal. De hecho, la mayoría de problemas de optimización combinatoria pueden plantearse como problemas de programación entera [Pardalos and Resende, 2002].

La optimización combinatoria es un área muy activa dentro de la investigación en optimización aplicada, donde se combinan técnicas procedentes de la teoría combinatoria, de la programación lineal y no lineal y de la teoría de algoritmos y estructuras de datos. Los avances recientes en la optimización combinatoria, que incluyen la combi-

natoria poliédrica, los métodos de planos de corte [Gomory, 1963], ramificación y poda [Nemhauser, 1966], ramificación y corte[Grötschel and Holland, 1991][Padberg and Rinaldi, 1991], búsquedas locales [Aarts and Lenstra, 1997] y meta-heurísticas[Pardalos and Resende, 2002], junto con los avances en la tecnología computacional disponible, han hecho posible aplicar la optimización combinatoria a un amplio rango de problemas. Así, estas técnicas se aplican para resolver problemas en dominios tan dispares como transportes y logística, agricultura, producción, telecomunicaciones, energía, arquitectura, industria farmacéutica y química, biología molecular, etc.

En la actualidad es posible solucionar un gran número de problemas combinatorios presentes en la vida real empleando las técnicas citadas basadas en programación entera [Pardalos and Resende, 2002], que han permitido la aparición de potentes sistemas comerciales de programación entera, como por ejemplo CPLEX [1] y Xpress [2]. Sin embargo, en numerosas ocasiones no es posible resolverlos de forma exacta debido a la gran dificultad que presentan algunos problemas de optimización combinatoria y sólo es posible encontrar soluciones cercanas al óptimo. Para estas ocasiones, los esfuerzos de investigación se han centrado en la aplicación de técnicas meta-heurísticas como, por ejemplo, la búsqueda tabú [Glover and Laguna, 1997], el temple simulado (*simmulated annealing*) [Kirkpatrick et al., 1983], GRASP[Feo and Resende, 1995], así como técnicas Soft Computing [Zadeh, 1994; Kecman, 2001], por ejemplo los algoritmos genéticos [Holland, 1992], que han permitido encontrar soluciones de buena calidad, en tiempos computacionales razonables, a problemas combinatorios de todo tipo [Pardalos and Resende, 2002].

En este último caso se enmarca el presente trabajo, es decir, en la resolución de problemas combinatorios complejos, de grandes dimensiones, donde explorar todas las posibilidades a fin de encontrar el óptimo es inabordable, ya sea por motivos económicos (probar cada combinación sea caro) o por motivos computacionales (temporalmente sea intratable). Así, en este trabajo se propondrá tanto una arquitectura de búsqueda que sea capaz de tratar problemas de naturaleza combinatoria en dominios diferentes, como la implementación de una herramienta que siga esta arquitectura y que facilite al usuario definir y resolver problemas de esta índole. Se persigue obtener soluciones a bajo coste, manteniendo la robustez y flexibilidad necesarias para abordar complejos problemas combinatorios. Así pues, la arquitectura se basará en técnicas Soft Computing, obteniendo una arquitectura híbrida que sea capaz de explorar grandes dimensiones de

[1] http://www.dashoptimization.com/
[2] http://www.ilog.com/products/cplex/

soluciones posibles, así como aproximar y predecir la calidad de las mismas.

La motivación inicial de encontrar un marco de trabajo de esta naturaleza vino marcada por la estrecha colaboración mantenida entre nuestro grupo de investigación de Tecnología Informática - Inteligencia Artificial (GTI-IA) [3] y el Instituto de Tecnología Química (ITQ) [4] del CSIC , dentro de un proyecto europeo ESPRIT (COMBICAT) en el ámbito de la Catálisis Combinatoria, que perseguía la obtención de nuevas técnicas que acelerasen la búsqueda de nuevos materiales catalíticos. La búsqueda de nuevos catalizadores implica el estudio de múltiples variables, por ejemplo seleccionar los elementos químicos y cantidades de los mismos que formarán parte del catalizador, elegir los métodos de preparación necesarios, establecer las condiciones de reacción a las que ofrecen mejor rendimiento, etc. En definitiva, el número de variables y la naturaleza de las mismas puede variar en gran medida según sea el ámbito para el cuál se busca un nuevo catalizador y qué propiedades se desean del mismo (respetuoso con el medio ambiente, económico, rápido).

Otras fuentes de motivación para realizar este proyecto fueron, por un lado, la participación del GTI-IA en un proyecto nacional en el que se pretende la gestión inteligente de actividades de ocio (GV06/315). Por otro, la colaboración mantenida con una empresa dedicada a la recomendación de música a través de la red y el interés de la misma en obtener nuevos mecanismos que le permitan recomendar música que sea del agrado de sus clientes, especialmente para el caso en el que no se disponga de mucha información de partida sobre los gustos de sus clientes. Así, se pretende aplicar la arquitectura de búsqueda propuesta dentro del ámbito de los Sistemas de Recomendación para la obtención de las preferencias de los usuarios, concretamente a la determinación de cuáles son los atributos más importantes para los usuarios de entre aquellos que caracterizan a un determinado producto o servicio, teniendo en cuenta un marco de trabajo en el que exista una escasez de información que impida aplicar otras técnicas.

Los dominios de aplicación planteados pueden parecer muy diferentes, puesto que el número y naturaleza de las variables bajo estudio varia en gran medida según sea el problema a tratar. Sin embargo, en ambos casos se atacan problemas combinatorios complejos, en los que se pretende conocer el conjunto de variables óptimo, así como determinar el valor que deben tomar estas variables para maximizar una función objetivo concreta.

[3] http://gti-ia.dsic.upv.es/
[4] http://itq.webs.upv.es/

1.2. Objetivos

El objetivo principal de este trabajo es establecer una arquitectura de búsqueda independiente del dominio de aplicación y capaz de abordar problemas combinatorios de grandes dimensiones, de los que se disponga de poca información de partida. Por tanto, el sistema a obtener deberá ser lo suficientemente flexible para poder adaptarse a las necesidades y requisitos propios del dominio de cada problema combinatorio a tratar. Para ello, se estudiará la aplicación de técnicas *Soft Computing* como herramientas principales de la arquitectura. Así pues, para la consecución de este objetivo, será necesario alcanzar una serie de objetivos complementarios:

- Revisión de las principales técnicas Soft Computing, analizando su aplicabilidad a problemas combinatorios de grandes dimensiones, determinando cuáles son las técnicas más adecuadas para el propósito de este trabajo.
- Análisis de las características de los problemas combinatorios planteados en el área de la Catálisis Combinatoria, a fin de evaluar la aplicabilidad de la arquitectura de búsqueda propuesta a este dominio.
- Análisis de las características, carencias y requisitos a contemplar dentro del área de los Sistemas de Recomendación, estableciendo en qué ámbito de este área podría ser de ayuda una arquitectura como la que se pretende obtener.
- Definición de una propuesta de arquitectura de búsqueda, basada en aquellas técnicas *Soft Computing* que se consideren más apropiadas.
- Evaluación de la arquitectura de búsqueda propuesta, validando su flexibilidad e independencia del dominio al ser aplicada a la resolución de problemas de diversa índole. Para ello se plantearán problemas combinatorios de interés, tanto dentro del área de la Catálisis Combinatoria como de los Sistemas de Recomendación.
- Desarrollo de un prototipo de aplicación que emplee la arquitectura de búsqueda propuesta y que pueda ser empleado fácilmente por cualquier usuario para resolver problemas combinatorios diversos.

1.3. Estructura del trabajo

La memoria del trabajo realizado ha sido dividida en dos partes. La primera parte aborda el estudio del arte requerido para la elaboración de este trabajo, abarcando los

tres siguientes capítulos. La segunda parte abarca la propuesta realizada en este trabajo y comprende los tres últimos capítulos.

Así, la primera parte comienza con el capítulo 2, donde se describen los fundamentos de las técnicas *Soft Computing*, estudiando las principales características de algunas de las técnicas más empleadas en los sistemas actuales de este tipo. El capítulo 3 aborda el área de la Catálisis Combinatoria, describiendo a grandes rasgos en qué consiste un catalizador, así como explicando los problemas principales contemplados en este dominio. Para finalizar la primera parte, el capítulo 4 analiza las principales características de los Sistemas de Recomendación destinados a usuarios y empresas dentro del comercio y la industria del entretenimiento, mostrando los requisitos de diseño e implementación que plantean. También se presentan en este capítulo los algoritmos más utilizados en los sistemas de recomendación empleados hasta la fecha.

La segunda parte comienza con el capítulo 5, donde se explica la propuesta de arquitectura de búsqueda planteada en el presente trabajo, describiendo cada una de las etapas de las que consta. El capítulo 6 relata los problemas que han sido planteados para ser resueltos con la arquitectura propuesta dentro de los dominios de la Catálisis Combinatoria y los Sistemas de Recomendación, así como los resultados obtenidos en su resolución, a fin de evaluar tanto cada una de las etapas de la propuesta como el marco de trabajo al completo. Las conclusiones y aportaciones realizadas se describen en el capítulo 7. Finalmente, en el anexo A, se describe el paquete de aplicaciones *SoftCombi*, desarrollado a partir del resultado del trabajo de investigación presentado en este trabajo.

Parte I

Estado del Arte

Capítulo

2

Técnicas Soft Computing

La Inteligencia Computacional o *Soft Computing*, en su término anglosajón, hace referencia a la utilización de una serie de metodologías que pueden trabajar tolerando cierto nivel de imprecisión, incertidumbre e información parcialmente cierta, siendo capaces de obtener soluciones con bajo coste, manteniendo la robustez y flexibilidad necesarias [Zadeh, 1994]. Así, bajo este término podemos encontrar combinaciones de la lógica difusa, computación neuronal, máquinas de soporte vectorial, algoritmos genéticos, computación evolutiva, aprendizaje automático, razonamiento probabilístico, etc [Kecman, 2001]. Es de reseñar que la característica más atrayente del *Soft Computing* consiste en facilitar la utilización combinada de las técnicas anteriormente citadas, obteniendo sistemas inteligentes híbridos.

A pesar de que las técnicas *Soft Computing* se encuentran en pleno desarrollo, muchas instituciones de investigación, industrias y firmas comerciales han comenzado a aplicar estas novedosas herramientas a problemas complejos de diversa índole [Kecman, 2001]. Las aplicaciones más importantes incluyen:

- Reconocimiento o clasificación de patrones, ya sean visuales, de sonido, táctil,etc.
- Pronóstico de series temporales, por ejemplo financieras, predicción meteorológi-

ca, etc.

- Diagnósticos aplicados a la medicina, a la ingeniería, etc.
- Robótica, en el área del control, la navegación, la coordinación o el reconocimiento de objetos.
- Control de procesos, como por ejemplo el control de plantas químicas, plantas de electricidad, vehículos o misiles.
- Optimización de problemas combinatorios, tales como la planificación de recursos, rutas, etc.
- Tratamiento de señales, reconocimiento del habla y de palabras.
- Visión artificial, aplicada a la inspección de calidad en la fabricación, reconocimiento facial, etc.
- Pronóstico financiero (porcentajes de interés, índices de stock, divisas).
- Servicios financieros y de ventas (predicción de ventas, clasificación de clientes, etc).

En ciertas áreas de aplicación, las técnicas *Soft Computing* y en concreto las redes neuronales, los modelos basados en la lógica difusa y las máquinas de soporte vectorial han superado a los métodos estadísticos convencionales. Sin embargo, en otras áreas como la robótica o los servicios financieros, siguen siendo una alternativa prometedora, sin estar totalmente empleadas en aplicaciones del mundo real.

Seguidamente se describirán los fundamentos de los algoritmos genéticos, las redes neuronales, la lógica difusa y las máquinas de soporte vectorial, ya que son algunas de las técnicas más empleadas en los sistemas *Soft Computing* actuales.

2.1. Algoritmos Genéticos

Los *Algoritmos Genéticos* (AG) son técnicas adaptativas empleadas tanto para la solución de problemas de búsqueda como de optimización. Están basados en los procesos genéticos llevados a cabo por los organismos biológicos. Los AG fueron desarrollados en la década de los setenta por John Holland y sus estudiantes de la Universidad de Michigan [Holland, 1992]. Su objetivo era simular el proceso adaptativo de los sistemas naturales y desarrollar sistemas artificiales que capturaran las características de los sistemas naturales. Principalmente son cuatro las características que distinguen los AG de otras técnicas de búsqueda y optimización [Goldberg, 1989]:

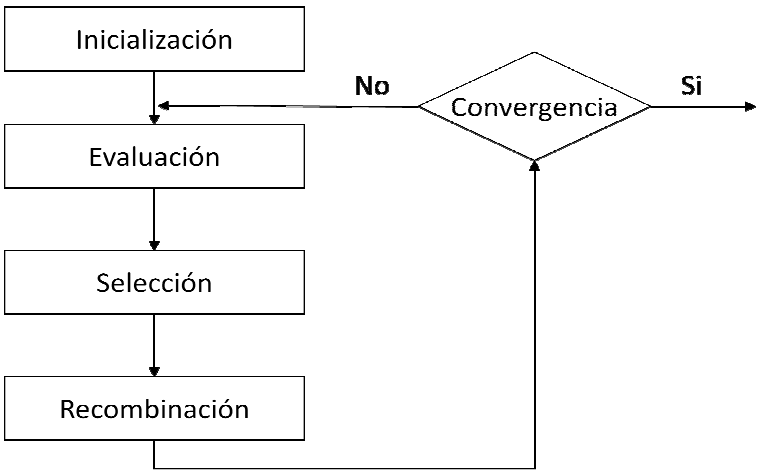

Figura 2.1: Esquema de actuación de un algoritmo genético simple

- No trabajan directamente sobre los parámetros a optimizar, sino sobre una codificación de los mismos.
- No buscan un único punto, sino que investigan sobre conjuntos o poblaciones de posibles soluciones o puntos.
- Emplean directamente la función objetivo, no su derivada o otro tipo de conocimiento auxiliar.
- Utilizan reglas de transición probabilísticas, no deterministas.

Los AG se han empleado con gran éxito tanto en problemas de búsqueda como en problemas de optimización [Oduguwa et al., 2005] en campos muy diversos. Esto es debido a diversas razones. En primer lugar, a pesar de resultar computacionalmente simples, son enormemente eficaces y eficientes [Goldberg, 1989]. Además, no están sujetos a restricciones sobre espacio de búsqueda (continuidad, derivable, etc.). Son capaces de explotar la información disponible sobre un espacio de búsqueda inicial desconocido, a fin de conducir la búsqueda hacia subespacios útiles. Esta característica es especialmente útil cuando deben enfrentarse a espacios de búsqueda amplios, complejos y poco conocidos, donde las herramientas clásicas de búsqueda (enumerativas, heurísticas, etc.) son inapropiadas, ofreciendo una aproximación válida a los problemas que requieren técnicas de búsqueda efectivas y eficientes.

La forma canónica de un AG codifica cada solución candidata a un problema dado como una cadena binaria o numérica (de enteros o reales), denominada cromosoma. Los AG simulan la evolución genética (cromosomas) de una población de individuos

empleando operadores de recombinación (cruce y mutación) y selección (de progenitores o de pertenencia a la siguiente generación)(Fig. 2.1). En un AG básico, el operador de cruce intercambia información genética entre dos progenitores, a fin de obtener hijos mejores. Por contra, el operador de mutación altera de forma aleatoria el valor de alguna de las características codificadas en el cromosoma, a fin de preservar la diversidad genética de la población y evitar la convergencia a máximos locales. Cada individuo es evaluado asignándole una determinada puntuación de aptitud o calidad alcanzada (*fitness*) que se calcula de forma proporcional al valor obtenido por el individuo al evaluar la función objetivo. La siguiente generación se forma con aquellos individuos que posean mejor puntuación por medio de los operadores de selección.

Desgraciadamente en numerosas ocasiones en los problemas que se presentan en el mundo real es difícil poder calcular la aptitud de los individuos, puesto que no se conoce una función específica para ello o calcularla es demasiado costoso. Por ejemplo, en el campo de la Catálisis (capítulo 3), es necesario realizar numerosos experimentos a fin de conocer los resultados catalíticos de las muestras a estudiar (estableciendo así su calidad). Estos experimentos resultan tremendamente costosos, tanto temporal como económicamente. Por todo ello, en numerosas ocasiones es necesario obtener modelos aproximados que permitan estimar la aptitud de los individuos propuestos. Para obtener estos modelos pueden emplearse numerosas técnicas, como por ejemplo métodos polinómicos, redes neuronales, máquinas de soporte vectorial, etc. [Jin, 2005].

Existen dos aspectos a tener en cuenta cuando se emplean modelos en la evaluación de la aptitud de los individuos. Primero, debe asegurarse que el algoritmo genético converge a un óptimo global o a un óptimo cercano al global. Segundo, que el coste computacional se reduzca todo lo posible. Hay que tener en cuenta que es muy difícil construir un modelo aproximado que sea globalmente correcto debido a que generalmente se parte de un número limitado de muestras de entrenamiento, con una distribución poco homogénea y un gran número de dimensiones.

Una consecuencia negativa de emplear modelos aproximados para la evaluación de la aptitud de los individuos propuestos por los AG consiste en que es más probable que los AG converjan hacia óptimos falsos. Por ello, en muchos casos es aconsejable que el modelo aproximado sea empleado conjuntamente con la función original de evaluación, a fin de supervisar la evolución del algoritmo. Así, la función original puede emplearse para evaluar la calidad de todos o una selección de los individuos de cierta generación, denominándose generaciones o individuos de control. La inclusión en las siguientes

generaciones de estos individuos de control, cuya calidad no ha sido estimada sino obtenida mediante la función original, ayuda a que el AG converja correctamente. Los mecanismos de control pueden ejercerse de diferentes formas, por ejemplo puede seguirse una política basada en las generaciones, aplicándose la función original cada cierto número de generaciones a todos los individuos. También puede llevarse a cabo el control en cada generación, obteniendo una selección de individuos controlados, etc.

A continuación se detallarán aspectos relativos a la codificación y a los operadores más comúnmente empleados por los AG, a fin de ofrecer una visión más específica de las bondades y características que ofrecen los AG al ser empleados en tareas de búsqueda y optimización.

2.1.1. Codificación

Los AG no trabajan directamente con los parámetros o variables a optimizar, sino que trabajan con una codificación de los mismos. La forma en la que se realiza dicha codificación condiciona en gran medida la elección de los operadores a emplear, así como el rendimiento final del AG. La codificación binaria, basada en cadenas fijas de unos y ceros, ha sido muy utilizada desde los inicios de los AG, puesto que es sencilla de aplicar y es posible obtener implementaciones muy simples y eficaces de los operadores de mutación y cruce. Además, diversos resultados teóricos demuestran que la utilización de alfabetos pequeños puede ser más efectiva que la utilización de alfabetos grandes [Goldberg, 1991].

Sin embargo, las bondades de los AG no residen en la utilización de cadenas binarias [Herrera et al., 1998], por este motivo la tendencia actual es la de emplear aquellas representaciones no binarias que resulten más adecuadas para el problema a tratar. En este sentido, una de las posibilidades más convenientes para la resolución de problemas cuyas variables pertenezcan a espacios de búsqueda continuos es la utilización de la codificación real, donde cada cromosoma está formado por un vector de números reales, donde cada gen representa a cada una de las variables a optimizar. Así pues, los AG que emplean este tipo de codificación han ofrecido excelentes resultados en la resolución de diversos problemas de optimización en dominios continuos [Eshelman and Schaffer, 1993][Ortiz et al., 2001][Herrera et al., 2002].

De forma similar, otras propuestas emplean AG que codifican el problema a tratar mediante una codificación basada en objetos, donde se establece el conjunto de com-

ponentes que pueden formar parte de un cromosoma, así como las reglas que rigen la forma en la que éstos pueden combinarse. En este caso, suele ser necesario que los operadores a emplear por el AG (cruce, mutación, etc.) estén específicamente diseñados para tratar con los objetos especificados en la codificación. Por ejemplo, mediante esta aproximación Karl Sims aplicó algoritmos genéticos para la creación de criaturas virtuales de bloques que se movían e interaccionaban en entornos tridimensionales virtuales [Sims, 1994b][Sims, 1994a]. Así, la morfología de las criaturas se optimizaba para que fueran capaces de realizar una tarea concreta, como andar, nadar, saltar, etc. Cada cromosoma codificaba las instrucciones necesarias para su montaje, por ejemplo, indicando qué piezas se empleaban y dónde se unían unas con otras. Otro ejemplo de este tipo de codificación nos lo ofrece el trabajo realizado por Anantha Sundaram et al., en el que se empleaba un AG para el diseño de aditivos para combustibles [Sundaram et al., 2001]. En este caso, los cromosomas estaban formados por tres tipos de objetos diferentes (cabeza, conexión y cola). Cada tipo de objeto contenía información que lo describía, así como también indicaba qué tipo de objetos y en qué lugar podían unirse a él. Mediante este sistema eran capaces de representar cualquier tipo de aditivo posible, aportando mucha información relativa a su estructura.

2.1.2. Operadores de Selección

A lo largo del proceso seguido por un AG, en numerosas ocasiones es necesario seleccionar individuos de entre los propuestos por el algoritmo, ya sea para reproducirse o recombinarse, o para formar parte de la siguiente generación propuesta por un AG. Así, en la mayoría de AG, a partir de una generación de partida se seleccionan algunos de sus individuos, aplicando sobre ellos los operadores de recombinación (cruce, mutación,etc.), obteniendo así nuevos individuos. Una vez obtenidos, es necesario volver a aplicar algún mecanismo de selección que dé lugar a la siguiente generación de individuos propuestos por el AG, de entre el conjunto formado por los individuos de la generación de partida y aquellos recién obtenidos (en el caso en el que los descendientes no reemplacen directamente a sus progenitores).

Esta selección puede realizarse de diferentes formas y atendiendo a criterios diversos, según cuál sea la estrategia de optimización seguida por el AG que se ajuste mejor a sus propósitos. Normalmente el operador de selección es empleado para mejorar la calidad media de una población, para ello asigna a los individuos de más calidad (en términos de la función de aptitud) una mayor probabilidad de ser seleccionados para

aplicar sobre ellos los operadores de recombinación y dar lugar a la siguiente generación. Por tanto, a través del operador de selección se focaliza la exploración sobre regiones prometedoras del espacio de búsqueda.

En otras ocasiones, como por ejemplo en el caso en el que se quiera que la población permanezca constante y mantenga un buen nivel de diversidad, será necesario ajustar el tamaño de la población tras la aplicación de los operadores de cruce y mutación, seleccionando de entre los progenitores e hijos aquellos individuos que posean un buen balance entre su aptitud y diversidad.

Por tanto, la pérdida de diversidad genética en la población dependerá en gran medida del mecanismo de selección empleado. Normalmente, los peores individuos son reemplazados por los mejores, luego su material genético desaparece tras realizar diferentes fases de selección. En otras ocasiones, todos los individuos son susceptibles de ser reemplazados, aunque con diferente probabilidad según sea su aptitud, dando lugar a poblaciones más diversas. Finalmente, existen aproximaciones en las que la cantidad de población no permanece constante, creciendo de una generación a otra, sin existir pérdida de material genético.

Por otra parte, algunos mecanismos de selección no permiten que un cromosoma sea seleccionado en más de una ocasión, mientras que en la mayoría de casos sí se permite.

En definitiva, existen numerosos mecanismos y algoritmos de selección, así como formas de asignar probabilidades de selección a cada uno de los individuos de la población. Seguidamente se citarán los esquemas de selección más comunes, pudiendo consultar para más información los siguientes trabajos [Goldberg and Deb, 1991][Bäck et al., 1991][Blickle and Thiele, 1995][Herrera et al., 1998].

Selección proporcional. En este caso, la probabilidad p_s de que un individuo o cromosoma C_i sea seleccionado se calcula como:

$$p_s(C_i) = \frac{f(Ci)}{\sum_{j=1}^{N} f(C_j)} \tag{2.1}$$

donde f es la función de aptitud o fitness empleada para conocer la calidad de cada cromosoma de la población estudiada [Holland, 1992][Goldberg, 1989].

Distintos mecanismos de selección emplean esta modalidad de asignar la probabilidad, por ejemplo el sistema Monte Carlo o de ruleta (*roulette wheel selection)* [Goldberg and Deb, 1991] y la selección estocástica universal [Baker, 1987].

Cromosoma	Aptitud
A	20
B	17
C	15
D	11
E	10
F	7
G	5
H	3
I	1

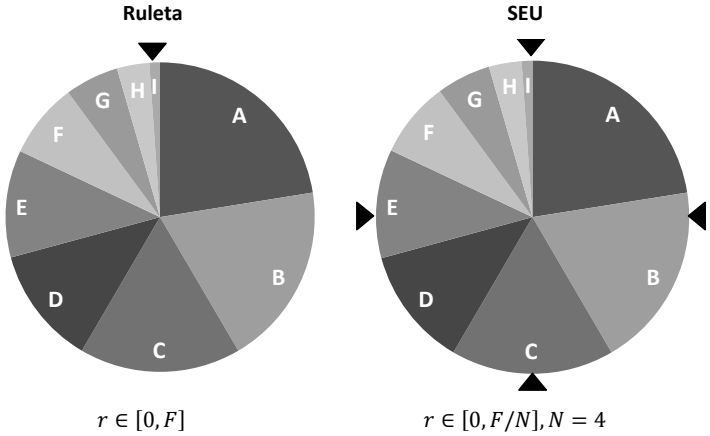

Figura 2.2: Selección Proporcional. Métodos de ruleta y selección estocástica universal (SEU)

En el sistema de ruleta un cromosoma tiene mayor probabilidad de ser elegido cuanto mayor sea su aptitud. A cada individuo de la población se le asigna una porción de la ruleta de forma proporcional a su aptitud, eligiendo los N individuos a seleccionar obteniendo N posiciones dentro de la ruleta de forma aleatoria ($r \in [0, F]$), es decir, como si se jugara a la ruleta N veces.

El sistema seguido por la selección estocástica universal es similar al de ruleta, pero difiere en que se seleccionan los N cromosomas a la vez, con un único valor aleatorio ($r \in [0, F/N]$) que indica la primera posición dentro de la ruleta y el resto se obtienen incrementando a la posición de partida un intervalo igual a F/N, $N - 1$ veces.

Selección por Ranking. Este tipo de selección fue sugerida inicialmente por Baker [Baker, 1985], permitiendo controlar de forma efectiva la probabilidad con la que los mejores individuos son seleccionados comparados con la probabilidad media de seleccionar al resto de individuos (*selection pressure*).

En cada generación, los individuos o cromosomas de la población son ordenados de acuerdo a su aptitud, asignándoles una puntuación o *ranking* a cada uno de ellos. Así, al peor individuo se le asigna una puntuación de 1, mientras que el mejor recibe una puntuación de N, siendo N el tamaño de la población. A partir de este ranking, se asigna a cada individuo i una probabilidad de selección mediante la siguiente función:

$$p_i = \frac{1}{N}\left(min + \frac{(max - min)\,(i-1)}{N-1}\right) \tag{2.2}$$

donde $max + min = 2$ and $1 \leq max \leq 2$. Por tanto la probabilidad de selección del individuo con peor aptitud es $p_1 = \frac{min}{N}$, mientras que la del mejor individuo es $p_N = \frac{max}{N}$. Los individuos finalmente seleccionados son aquellos resultantes de aplicar un muestreo sobre la población aplicando las probabilidades de selección así calculadas.

Una variante de este tipo de selección es la propuesta de Patrick H. Winston en la que también se tiene en consideración la diversidad de cada uno de los individuos, dando lugar al método denominado *Rank-space*, que persigue mantener un equilibrio entre la calidad de los individuos seleccionados y su diversidad, a fin de evitar la pérdida excesiva de material genético [Winston, 1992].

En este caso, el ranking o puntuación que recibe un individuo es el resultado de la combinación de la puntuación recibida de acuerdo a la aptitud que posea y de la puntuación recibida atendiendo a su diversidad (por ejemplo, simplemente sumando ambos). A partir de esta puntuación combinada, se calculan las probabilidades de selección de los individuos de forma análoga al método anterior.

en la figura 2.3 se muestra un ejemplo comparativo de los métodos de ranking original y de su variante *Rank-space*.

Selección por torneo. Este método consiste en formar subconjuntos de cromosomas de tamaño fijo de forma aleatoria a partir de la generación, seleccionando algunos de ellos [Brindle, 1981][Goldberg and Deb, 1991][Chakraborty and Chakraborty, 1997]. Por ejemplo, pueden tomarse los cromosomas de cinco en cinco, seleccionando sólo uno de ellos en cada ocasión. Para realizar esta selección, dentro de cada subconjunto se realiza un «torneo», donde los cromosomas son comparados entre sí, considerando como ganadores a aquellos k cromosomas que tengan mejor

Cromosoma	Aptitud	Diversidad	Ranking	Rank-Space
A	20	3	9	**9**
B	17	4	8	8
C	15	4	7	7
D	11	3	6	6
E	10	3	5	5
F	7	1	4	3
G	5	5	3	4
H	3	2	2	1
I	1	4	1	2

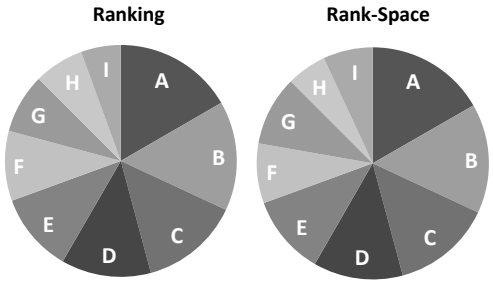

Figura 2.3: Selección por Ranking. Métodos de Ranking y Rank-Space, tomando para el cálculo de la probabilidad de selección: $min = 0.5$ y $max = 1.5$.

aptitud (que formarán parte del grupo de individuos seleccionados).

Mediante este tipo de selección es posible ajustar de forma sencilla la probabilidad con la que los mejores individuos son seleccionados comparados con la probabilidad media de seleccionar al resto de individuos (*selection pressure*).

2.1.3. Operadores de Mutación

Los operadores de mutación tienen un carácter eminentemente explorador, cuya finalidad es la de preservar la diversidad genética de la población, puesto que permiten la aparición de nuevo material genético. Así, mediante la mutación se permite explorar nuevas áreas del espacio de búsqueda, que no hayan aparecido ni en la generación inicial ni hayan sido el fruto del cruce de sus individuos. Por tanto asegura que la probabilidad de alcanzar cualquier punto dentro del espacio de búsqueda no es nula. Además, mediante la mutación se impide que un AG converja prematuramente hacia máximos locales.

Básicamente, un operador de mutación altera el contenido genético de un cromosoma de forma aleatoria. La forma en la que esta alteración se lleva a cabo depende en gran medida del tipo de codificación empleada, así como de su grado de incidencia. Generalmente el operador es aplicado siguiendo una probabilidad p_m dada, indicando además el número de genes que debe alterar cuando es aplicado. Una p_m alta introduce demasiado ruido dentro del sistema de búsqueda, impidiendo la convergencia de un AG. Por tanto, debe alcanzarse un equilibrio entre la necesidad de mantener la diversidad dentro de la población y el grado de convergencia alcanzado por el AG, indicando para ello una p_m baja.

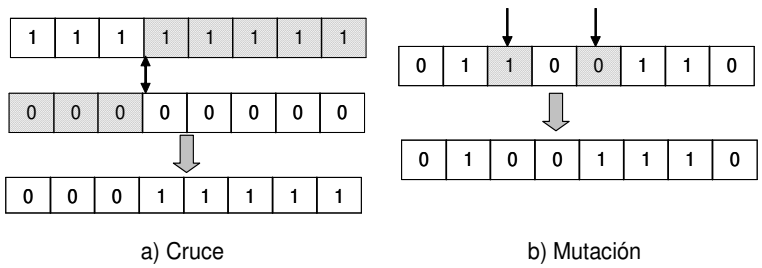

Figura 2.4: Ejemplo de aplicación de operadores de cruce y mutación básicos sobre cromosomas binarios

En el caso de emplear codificación binaria, se selecciona de forma aleatoria el gen o los genes a modificar y se intercambia su valor, es decir, si actualmente el gen tiene un valor de "0", se le asigna un "1" y viceversa [Holland, 1992][Goldberg, 1989]. En la figura 2.4-b puede apreciarse un ejemplo de este tipo de operador.

En el caso de emplear codificación real existen numerosos métodos para aplicar este operador. Seguidamente citaremos alguno de ellos, teniendo en cuenta que $C = (c_1, c_2, \ldots, c_i, \ldots, c_n)$ es un cromosoma, $c_i \in [a_i, b_i]$ el gen seleccionado para ser mutado y c_i' el gen i mutado.

Mutación aleatoria. En este caso, a c_i' se le asigna un valor r aleatorio uniforme dentro del dominio del gen $[a_i, b_i]$ [Michalewicz, 1992].

Mutación no uniforme. Al aplicar este método, el nuevo valor para el gen mutado viene dado por la ecuación 2.3, donde t indica la generación en la que se está, g_{max} el número máximo de generaciones y τ puede tomar un valor de cero o uno aleatoriamente [Michalewicz, 1992].

$$c_i' = \begin{cases} c_i + \Delta(t, b_i - c_i) \; si \; \tau = 0 \\ c_i - \Delta(t, c_i - a_i) \; si \; \tau = 1 \end{cases} \tag{2.3}$$

$$\Delta(t, y) = y(1 - r^{(1 - \frac{1}{gmax})^b}) \tag{2.4}$$

En la ecuación 2.4, r representa a un número aleatorio dentro del intervalo $[0, 1]$ y b es un parámetro elegido por el usuario, que determina el grado de dependencia sobre el número de iteraciones. El valor devuelto por esta función se encuentra en el rango $[0, y]$ de tal forma que la probabilidad de devolver un valor cercano a cero se incrementa conforme avanza el algoritmo. El tamaño del intervalo para cada gen dentro de la generación puede decrecer con el paso de las generaciones. Esta característica permite que este operador realice una búsqueda uniforme dentro del espacio inicial cuando t es pequeño, transformándose en una búsqueda más localizada conforme pasan las generaciones.

Real Number Creep. Este método fue presentado por [Davis, 1991]. Cuando se optimiza una función continua que presenta diversos máximos y mínimos locales y, en un momento dado, se obtiene un cromosoma que está situado en un buen máximo local, puede ser interesante generar nuevos cromosomas cercanos a éste, a fin de alcanzar el punto situado en el pico de ese máximo. Para llevar a cabo esta

tarea, se modifica el valor de los genes del cromosoma incrementándolos o decrementándolos en una pequeña cantidad obtenida de forma aleatoria. Es el usuario el encargado de determinar cual es el máximo desplazamiento que se permite.

Mutación de Mühlebein. En este caso, Mühlebein [Mühlenbein and Schlierkamp-Voosen, 1993] propuso obtener el nuevo valor para cada gen de la siguiente forma:

$$c_i' = c_i \pm rang_i \cdot \gamma \tag{2.5}$$

$$\gamma = \sum_{k=0}^{15} \alpha_k 2^{-k} \tag{2.6}$$

donde $rang_i$ define el rango de mutación y suele establecerse en $0.1 \cdot (b_i - a_i)$. El signo positivo o negativo se elige con una probabilidad de 0.5 y $\alpha_i \in \{0, 1\}$ se obtiene aleatoriamente con una probabilidad $p(\alpha_i = 1) = \frac{1}{16}$. Por tanto, mediante este operador se obtienen valores dentro del intervalo $[c_i - rang_i, c_i + rang_i]$, con una alta probabilidad de generar valores cercanos a c_i. En concreto, es posible obtener valores con una proximidad mínima de $rang_i \cdot 2^{-15}$.

2.1.4. Operadores de Cruce

Los operadores de cruce tienen un carácter explotador, cuya finalidad es intercambiar material genético entre cromosomas seleccionados para ello. La filosofía seguida es tratar de mejorar la población a partir del intercambio genético entre candidatos prometedores, con la idea de que su descendencia obtenga mejores resultados por poseer combinaciones de genes provenientes de buenos cromosomas. Normalmente no todos los cromosomas de una generación son cruzados, sino que se aplica una probabilidad de cruce p_c para seleccionar, mediante alguno de los mecanismos presentados anteriormente (sección 2.1.2), los cromosomas que actuarán como progenitores.

La manera en la que se aplica este operador depende de la elección realizada respecto a la representación o codificación del problema (sección 2.1.1). Así pues, para el caso de emplear codificación binaria existen diversas posibilidades para llevarlo acabo [Herrera et al., 1998], por lo que a continuación se explicarán algunas de ellas.

Cruce simple. Fue el primer operador de cruce pospuesto por John Holland y sus estudiantes [Holland, 1992]. Consiste en seleccionar a dos cromosomas como progenitores, $C_1 = (c_1^1, c_2^1, \ldots, c_n^1)$ y $C_2 = (c_1^2, c_2^2, \ldots, c_n^2)$ y elegir aleatoriamente una

posición i a partir de la cual se intercambiará el material genético de los progenitores, teniendo como resultado dos descendientes $H_1 = (c_1^1, \ldots, c_i^1, c_{i+1}^2, \ldots, c_n^2)$ y $H_2 = (c_1^2, \ldots, c_i^2, c_{i+1}^1, \ldots, c_n^1)$. Un ejemplo de este tipo de operador puede observarse en la figura 2.4-a.

Cruce en k-puntos. Consiste en una generalización del cruce simple, en donde se establecen de forma aleatoria k puntos de intercambio de información genética en lugar de un único punto [Eshelman et al., 1989]. Por ejemplo para $k = 2$, se determinarían aleatoriamente dos posiciones $i, j \in [1, n]$ intercambiando los segmentos de material genético de los progenitores C_1 y C_2 que determinan, dando lugar a dos hijos $H_1 = (c_1^1, \ldots, c_i^1, c_{i+1}^2, \ldots, c_j^2, c_{j+1}^1, \ldots c_n^1)$ y $H_2 = (c_1^2, \ldots, c_i^2, c_{i+1}^1, \ldots, c_j^1, c_{j+1}^2, \ldots c_n^2)$.

Cruce uniforme. En este caso, el valor de cada gen i de los descendientes se obtiene seleccionando de forma aleatoria uniforme de entre los valores de los genes i de los dos progenitores [Sywerda, 1989].

Si el tipo de representación elegida para el problema se basa en codificación real, también podemos seleccionar la forma de aplicar este operador de entre las numerosas posibilidades ya planteadas hasta el momento. A modo de ejemplo, citaremos algunas de ellas, donde se asume que se han seleccionado como progenitores (siguiendo alguno de los criterios citados anteriormente sección 2.1.2) a dos cromosomas $C_1 = (c_1^1, \ldots, c_n^1)$ y $C_2 = (c_1^2, \ldots, c_n^2)$, que tras aplicar el operador de cruce ofrecerán un número diferente de descendientes dependiendo del método seguido. Lo que estos métodos de cruce tienen en común es que explotan o interpolan los intervalos de los genes que determinan los progenitores en lugar de intercambiar conjuntos de símbolos, como en el caso de la codificación binaria o basadas en representaciones mediante caracteres o similares.

Cruce plano o de sobrero de copa. A partir de los genes de los progenitores C_1 y C_2, se genera un descendiente $H = (h_1, \ldots, h_i, \ldots, h_n)$, donde h_i se obtiene seleccionando de forma aleatoria uniforme un valor dentro del intervalo $[c_i^1, c_i^2]$, construido a partir del valor de los genes i de los progenitores [Radcliffe, 1991].

Cruce Simple. Este operador [Wright, 1991][Michalewicz, 1992] actúa de forma análoga al primer operador de cruce propuesto por Holland para el caso de emplear codificación binaria o basada en cadenas, citado anteriormente. Así pues, se selecciona de forma aleatoria una posición $i \in \{1, 2, \ldots, n-1\}$ y se generan dos nuevos

cromosomas, H_1 y H_2, intercambiando el material genético de los progenitores a partir de esa posición:

$$H_1 = (c_1^1, c_2^1, \ldots, c_i^1, c_{i+1}^2, \ldots, c_n^2)$$
$$H_2 = (c_1^2, c_2^2, \ldots, c_i^2, c_{i+1}^1, \ldots, c_n^1)$$

Cruce Aritmético. En este caso se obtienen dos descendientes $H_k = (h_1^k, \ldots, h_i^k, \ldots, h_n^k)$ $k = 1, 2$, donde $h_i^1 = \lambda c_i^1 + (1 - \lambda)c_i^2$ y $h_i^2 = \lambda c_i^2 + (1 - \lambda)c_i^1$. Si se establece un λ constante se obtiene el cruce aritmético uniforme, mientras que si se establece un λ que varíe conforme avancen las generaciones obtenidas, se establece el cruce aritmético no-uniforme [Michalewicz, 1992].

Cruce BLX-α. Aplicando este operador, se obtiene un descendiente $H = (h_1, \ldots, h_i, \ldots, h_n)$, donde a cada gen h_i de forma aleatoria se le asigna un valor dentro del intervalo $[c_{min} - I \cdot \alpha, c_{max} + I \cdot \alpha]$, donde $c_{max} = max(c_i^1, c_i^2)$, $c_{min} = min(c_i^1, c_i^2)$ e $I = c_{max} - c_{min}$. Al emplear este operador con un $\alpha = 0.0$ se obtienen los mismos resultados que con el cruce plano o de sombrero de copa [Eshelman and Schaffer, 1993].

Cruce Lineal. Este operador de cruce [Wright, 1991] calcula tres descendientes $H_k = (h_1^k, \ldots, h_i^k, \ldots, h_n^k)$ $k = 1, 2, 3$, de los cuales los dos más prometedores son seleccionados para formar parte de la siguiente generación y sustituir a los padres, mientras que el restante se descarta. Los tres descendientes se calculan de la siguiente forma:

$$h_i^1 = \frac{1}{2}c_i^1 + \frac{1}{2}c_i^2$$
$$h_i^2 = \frac{3}{2}c_i^1 - \frac{1}{2}c_i^2$$
$$h_i^3 = -\frac{1}{2}c_i^1 + \frac{3}{2}c_i^2$$

Cruce Discreto. Mediante este procedimiento se obtiene un descendiente $H = (h_1, \ldots, h_i, \ldots, h_n)$, donde a cada gen h_i se le asigna aleatoriamente un valor del conjunto $\{c_i^1, c_i^2\}$ [Mühlenbein and Schlierkamp-Voosen, 1993].

Cruce de línea extendida . En este caso también se obtiene un único descendiente $H = (h_1, \ldots, h_i, \ldots, h_n)$, donde $h_i = c_i^1 + \alpha(c_i^2 - c_i^1)$, siendo α un valor aleatorio

dentro del intervalo $[-0.25, 1.25]$ [Mühlenbein and Schlierkamp-Voosen, 1993].

Cruce extendido intermedio. A partir de los dos progenitores se determina un descendiente $H = (h_1, \ldots, h_i, \ldots, h_n)$, cuyos genes $h_i = c_i^1 + \alpha_i(c_i^2 - c_i^1)$, siendo α_i un valor aleatorio dentro del intervalo $[-0.25, 1.25]$ [Mühlenbein and Schlierkamp-Voosen, 1993]

Cruce heurístico de Wright. En este tipo de cruce, también se obtiene un descendiente $H = (h_1, \ldots, h_i, \ldots, h_n)$. Para calcularlo, se toma el progenitor con mejor aptitud, supongamos que sea C_i^1 y se calcula el gen i como $h_i = r \cdot (c_i^1 - c_i^2 + c_i^1)$, donde r es un número aleatorio perteneciente a $[0, 1]$[Wright, 1990].

Cruce lineal BGA. En este caso también se parte del progenitor con mejor aptitud, supongamos de nuevo que se trata de C_i^1. Para calcular el descendiente $H = (h_1, \ldots, h_i, \ldots, h_n)$, se obtiene el gen h_i como:

$$h_i = c_i^1 \pm rang_i \gamma \Lambda$$
$$\Lambda = \frac{c_i^2 - c_i^1}{\|C_1 - C_2\|}$$

El signo "$-$" se elige con una probabilidad de 0.9. Normalmente $rang_i$ es $0.5 \cdot (b_i - a - i)$ y $\gamma = \sum_{k=0}^{15} \alpha_k 2^{-k}$, donde $\alpha_i \in \{0, 1\}$ se genera aleatoriamente con la probabilidad $p(\alpha_i = 1) = \frac{1}{16}$. Este operador de cruce está basado en la mutación de Mühlebein (sección 2.1.3) [Schlierkamp-Voosen, 1994].

Finalmente, para problemas en los que se emplee una codificación específica, será necesario establecer el mecanismo de cruce, teniendo en cuenta la filosofía de este operador, es decir, el intercambio genético entre los mejores cromosomas o individuos de la población.

Una vez detallados los fundamentos de los algoritmos genéticos, se comentarán a continuación las características más relevantes de otra técnica ampliamente utilizada dentro de los sistemas soft computing: las redes neuronales artificiales.

2.2. Redes Neuronales Artificiales

Las Redes Neuronales Artificiales (RNA)[Bishop, 2006; Bishop, 1996; Ripley, 1996] son herramientas no lineales que ofrecen un alta eficiencia, siendo capaces de establecer relaciones entre datos de entrada y de salida, sin ningún tipo de conocimiento previo acerca de la posible correlación existente entre las variables involucradas en el sistema. Por tanto, siguen un modelo conceptual de caja negra, donde el proceso es totalmente desconocido, no existe conocimiento previo, pero existen datos de partida, ya sean medidas, observaciones, muestras, etc.

Las RNA están inspiradas en las redes neuronales biológicas, consistiendo en un determinado número de neuronas artificiales, que están interconectadas a través de pesos sinápticos formando la red. La unidad básica de cómputo de una RNA es la neurona o nodo, compuesta de:

- un conjunto de entradas, $x_j(t)$, cada una de las cuales está caracterizada por medio de un peso sináptico w_{jk}, que representa la intensidad de las interacciones entre una neurona k y cada una de las neuronas j de la capa anterior.
- una regla de propagación (ecuación 2.7), que determina la entrada efectiva de la neurona k a partir de todas las entradas individuales a la misma.
- una función de activación F_k, que determina la salida y_k (ecuación 2.8) de la neurona k por medio de su nivel de excitación.
- una salida externa adicional, llamada polarización o bias (b_k), que incrementa o disminuye la excitación umbral de una neurona.

Las funciones de activación más comunes son las funciones sigmoidales, tales como las logísticas (ecuación 2.9) y tangenciales (ecuación 2.10), que representan un equilibrio entre comportamientos lineales y no lineales.

Las fórmulas asociadas a una neurona son las siguientes:

$$S_k = \sum_{j=1}^{n} w_{jk} * x_j + b_k \tag{2.7}$$

$$y_k = F_k(S_k) \tag{2.8}$$

$$F_k(S_k) = 1/(1 + e^{-S_k}) \tag{2.9}$$

$$F_k(S_k) = (e^{S_k} - e^{-S_k})/(e^{S_k} + e^{-S_k}) \tag{2.10}$$

Las RNA poseen dos características que hacen que su utilización sea muy interesante en un amplio número de problemas, puesto que son capaces de dar respuestas rápidamente y además, pueden ofrecer resultados satisfactorios para muestras o patrones totalmente desconocidos por las redes. Las RNA necesitan aprender acerca del problema bajo estudio, denominándose a este periodo de aprendizaje como proceso de entrenamiento. Durante éste, se proporciona a las RNA un conjunto de muestras o individuos que pertenezcan al dominio del problema. Seguidamente, las RNA establecen las correlaciones matemáticas necesarias entre sus neuronas y las muestras proporcionadas, siendo normalmente necesario un gran número de muestras y tiempo de cómputo [Ripley, 1996].

Existen dos maneras de llevar a cabo el proceso de entrenamiento: de forma supervisada o no supervisada. Las técnicas que siguen un modelo supervisado proporcionan a las RNA datos de entrenamiento que contienen tanto la información de entrada como la respuesta o salida esperada para los mismos. Por ejemplo, en el caso de un problema de reconocimiento de caracteres, se proporciona tanto el patrón de la imagen que representa al carácter (la imagen de una X) como el carácter representado en sí (X). A lo largo del proceso de entrenamiento, la RNA ajusta el peso de sus neuronas de forma que se minimice el error entre la salida ofrecida por la red y el valor de salida deseado. Por otra parte, las técnicas que emplean un modelo no supervisado proporcionan sólo datos con información de entrada, sin información sobre las salidas esperadas para los mismos. Por tanto, la RNA debe clasificar las entradas y las salidas basándose en su similitud con otras entradas.

Las redes neuronales han demostrado ser también una buena herramienta para la aproximación de funciones [Jin, 2005]. De hecho, son una buena opción cuando hay que enfrentarse a problemas de los que se dispone de poca información, pero cuyo dominio tiene múltiples dimensiones. Para estos fines han sido ampliamente utilizados tanto los perceptrones multicapa como las redes basadas en funciones radiales.

Seguidamente se explicarán estas dos técnicas, así como los mapas auto-organizativos, al ser un buen ejemplo de la utilización del entrenamiento no supervisado.

2.2.1. Perceptrón Multicapa

Una de las RNA más conocidas y empleadas usando aprendizaje supervisado es el Perceptrón Multicapa, creado originalmente por Frank Rosenblatt [Rosenblatt, 1962]. Éste ha sido utilizado de forma exitosa en problemas de clasificación y predicción. En un perceptrón multicapa, las neuronas se agrupan en capas o niveles, donde cada entrada a una neurona está compuesta por las salidas de las capas anteriores, excepto para las neuronas de la capa de entrada, cuya entrada corresponde a los datos de partida del problema a resolver. Así, las características del problema determinan el número de neuronas de la capa de entrada y la de salida. Sin embargo, el problema no ofrece ninguna información que ayude a establecer el resto de la topología de la red. Por ello, es necesario llevar a cabo estudios previos a fin de determinar el número necesario de capas ocultas o intermedias y el número de neuronas en cada capa que resulte más adecuado para resolver el problema. La ecuación siguiente (2.11) muestra un ejemplo de un perceptrón con una capa de entrada, una capa oculta y una capa de salida, siendo D el número de nodos de la capa de entrada, M el número de nodos de la capa oculta, K el número de salidas y $h(\cdot)$ y $\sigma(\cdot)$ las funciones de activación empleadas. Los parámetros bias se representan añadiendo una entrada (x_0) y un nodo (z_0) adicionales por cada capa oculta.

$$y_k(x, w) = \sigma(\sum_{j=0}^{M} w_{kj}^{(2)} h(\sum_{i=1}^{D} w_{ji}^{(1)} x_i)) \tag{2.11}$$

La figura 2.5 nos muestra la representación gráfica de la ecuación (2.11). Los nodos de la red representan las variables de entrada (x_i), ocultas (Z_m) y de salida (Y_k). Los pesos se representan mediante las conexiones entre los diferentes nodos. Los parámetros bias se corresponden con las variables añadidas X_0 y Z_0. El sentido de las flechas indica la dirección del flujo de información durante el proceso de entrenamiento.

Uno de los problemas de los perceptrones multicapa es el peligro de obtener un modelo sobre-entrenado, es decir, un modelo que se ajuste demasiado a la información con la que ha sido entrenado, pero que es incapaz de ofrecer buenas respuestas para individuos de los que no posee información previa. Por ello, es necesario establecer ciertos meca-

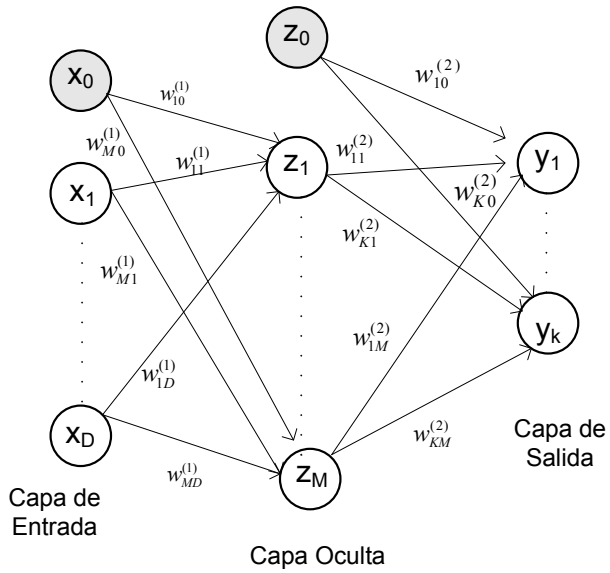

Figura 2.5: Perceptron Multicapa, con una capa de entrada, una oculta y una de salida

nismos que controlen el rendimiento ofrecido por el modelo, a fin de detectar cuándo es necesario volver a entrenar la red o cuándo dejar de hacerlo.

2.2.2. Redes Neuronales de Funciones Base Radiales

Este tipo de redes emplean tanto aprendizaje no supervisado (en una primera fase) como supervisado (en una segunda y definitiva fase). Difieren de los perceptrones en la forma en la que se activa una neurona, puesto que en lugar de calcular la función no lineal del producto escalar de un vector de entrada y un vector de pesos, en su caso la activación se define mediante la *distancia* entre un vector de entrada y un vector prototipo. Los métodos empleados en entrenar Redes de Funciones Base Radiales (RFBR) son más rápidos que los empleados en entrenar perceptrones multicapa y consisten en dos fases. En una primera fase se determinan los parámetros que gobiernan las funciones base mediante rápidos algoritmos no supervisados. En la segunda fase de entrenamiento se obtienen los pesos finales de la capa, para lo que se necesita resolver un problema lineal, empleando algoritmos supervisados.

Los métodos de funciones base radiales tienen sus orígenes en técnicas para obtener una interpolación exacta de un conjunto de puntos de información en un espacio multidimensional [Powell, 1987][Bishop, 1996]. Para solucionar el problema de la interpolación exacta se necesita que cada vector de entrada (x) se corresponda exactamente con

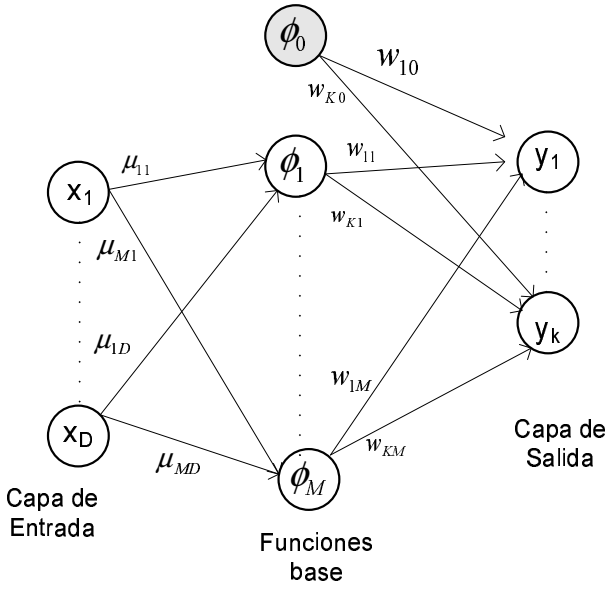

Figura 2.6: Red Neuronal de funciones base radial

su correspondiente vector objetivo (t).

Introduciendo una serie de modificaciones al procedimiento de interpolación exacta se obtiene el modelo de redes neuronales de funciones base radiales [Broomhead and Lowe, 1988][Moody and Darken, 1989][Bishop, 1996]. Éstas proporcionan una función de interpolación suave donde el número de funciones de base radial viene determinado por la complejidad del mapa a representar, en lugar del tamaño del conjunto de datos de que dispone. La ecuación 2.12 define un modelo de RFBR, donde x representa al vector de entrada, ϕ_j son las funciones base , w_{kj} los pesos y w_{k0} las bias. Asimismo, la figura 2.6 muestra la representación gráfica de la RFBR definida por la citada ecuación.

$$y_k(x) = \sum_{j=1}^{M} W_{kj}\phi_j(x) + w_{k0} \tag{2.12}$$

$$\phi_j(x) = exp\left(-\frac{\|x-\mu_j\|^2}{2\sigma_j^2}\right) \tag{2.13}$$

La ecuación 2.13 nos muestra una función base Gaussiana, donde x es el vector de entrada de d dimensiones con elementos x_i, μ_j es el vector que determina el centro de la función base ϕ_j , cuyos elementos son μ_{ji} y σ_j es el parámetro de amplitud.

2.2.3. Mapas auto-organizativos

Los mapas auto-organizativos (*Self-Organizing Maps,SOM*) fueron propuestos en 1982 [Kohonen, 1998][Kohonen, 2001] y constituyen un ejemplo de redes neuronales que emplean algoritmos de aprendizaje no supervisados. En su forma básica producen un grafo de similitud de la información de entrada. Convierten las relaciones estadísticas no lineales existentes en grandes volúmenes de información, en simples relaciones geométricas de sus puntos imagen sobre una visualización de pequeñas dimensiones denominada *mapa*, que generalmente es una cuadrícula regular de dos dimensiones. Así, los SOM compactan la información mientras preservan las relaciones topológicas y/o métricas más importantes entre los elementos principales, obteniendo cierto grado de abstracción de la información de partida.

Como la mayoría de las redes neuronales, los SOM actúan en dos fases: entrenamiento y *mapeo*. A lo largo de la fase de entrenamiento, la red construye el mapa empleando para ello ejemplos como entrada. El objetivo de la fase de entrenamiento es conseguir que partes diferentes de la red respondan de forman similar ante ciertos patrones de entrada. Cuando la red está entrenada, puede clasificar cualquier vector de entrada automáticamente en la fase de mapeo.

Las características de visualización y abstracción que aportan los SOM hacen de ellos una herramienta muy útil en tareas complejas como análisis de procesos, percepción automática, control y comunicaciones.

Tras esta breve revisión acerca de algunos de los tipos de redes neuronales existentes, se expondrán otras dos técnicas destacables dentro del área soft computing: los sistemas de Lógica Difusa y las Máquinas de Soporte vectorial.

2.3. Sistemas de Lógica Difusa

Los Sistemas de Lógica Difusa (SLD), junto con las redes neuronales, constituyen las principales herramientas de modelado de las técnicas de *Soft Computing*. En [Kecman, 2001] la lógica difusa se define brevemente como: *"una herramienta para transformar el conocimiento estructurado humano a algoritmos ejecutables"*. El campo de la lógica difusa es muy amplio y cubre muchos conceptos matemáticos y lógicos.

El concepto de la lógica difusa (*Fuzzy Logic*) se aplica con diferentes sentidos. Así, la lógica difusa puede considerarse como un sistema lógico que persigue modelar el razonamiento humano cuando éste es aproximado en lugar de exacto. Sin embargo, en un sentido más amplio, la lógica difusa puede entenderse como un conjunto difuso de teoría de clases cuyos límites no están definidos o son difusos o borrosos. Los métodos basados en la lógica difusa pueden utilizarse para diseñar sistemas inteligentes sobre la base del conocimiento expresado en un lenguaje común. En la actualidad casi no existen áreas de actividad humana en las que la lógica difusa no haya sido empleada. La principal razón de que la lógica difusa sea tan versátil es que permite tanto el procesado de información expresada de forma numérica como de información expresada de manera simbólica. Frecuentemente, los sistemas diseñados y desarrollados empleando lógica difusa han resultado ser más eficientes que aquellos basados en aproximaciones convencionales [Kecman, 2001].

En el modelado mediante lógica difusa se sigue un modelo conceptual de caja blanca, donde se tiene cierto conocimiento sobre la solución, siendo el conocimiento mismo lo que se modela. La lógica difusa puede considerarse una herramienta muy eficaz para transformar el conocimiento estructurado humano en útiles algoritmos. Al igual que en el proceso de razonamiento e inferencia humanos, cada afirmación, cada medida u observación, posee cierto grado de certeza. Este grado se expresa a través de funciones de pertenencia que miden el grado de pertenencia de ciertas entradas a los subconjuntos difusos dados.

2.4. Máquinas de Soporte Vectorial

Las máquinas de Soporte Vectorial , conocidas en su termino anglosajón como *Support Vector Machines* (SVM), comenzaron a desarrollarse a finales de los años sesenta, teniendo como origen la Teoría de Aprendizaje Estadística desarrollada por V. Vapnik y A.Y. Chervonenkis [Vapnik et al., 1997]. Estos métodos de aprendizaje constituyen una buena herramienta de clasificación y reconocimiento de patrones, así como de estimación de funciones en problemas de regresión. Además, también pueden emplearse para entrenar modelos polinómicos, tales como las redes neuronales, los modelos difusos o las funciones de base radial [Kecman, 2001].

Estas técnicas de aprendizaje aparecieron en el marco de la minimización del riesgo estructural (*structural risk minimization, SRM*), en oposición a la minimización del

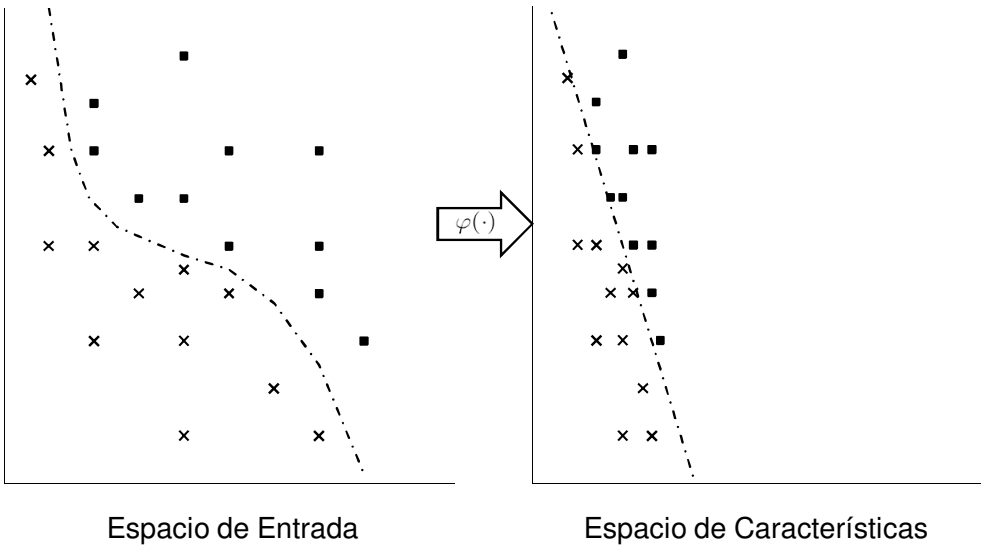

Figura 2.7: Aplicación de una función de mapeo $\varphi(\cdot)$ para transformar las muestras a fin de que sean separables mediante un hiperplano

riesgo empírico (*error risk minimization, ERM*) comúnmente empleado en métodos de aprendizaje estadísticos así como en el entrenamiento de redes neuronales. Así pues, siguiendo el principio SRM, se persigue minimizar un límite superior de la generación del error, en lugar de minimizar el error predicho para los datos de entrenamiento, tal y como se realiza siguiendo el principio ERM. Por tanto, gracias a seguir una estrategia basada en SRM, las SVM ofrecen buenos resultados al enfrentarse a nuevos datos, ofreciendo modelos capaces de generalizar de forma adecuada.

Las SVMs [Duda et al., 2001][Cristianini and Shawe-Taylor, 2000], a diferencia de las redes neuronales, abstraen el problema desde un espacio de atributos a un espacio de patrones de características con mayor dimensión, a fin de que puedan ser separadas por un hiperplano. Así, mediante una función no lineal de mapeo apropiada $\varphi(\cdot)$, que aumente la dimensión de forma adecuada, es posible separar muestras que pertenezcan a dos categorías diferentes mediante un hiperplano. Por tanto, la idea es transformar los datos que se encuentran en una dimensión en la que no pueden ser clasificados de forma lineal a un estado de dimensión diferente, donde puedan ser clasificados haciendo uso del método de clasificación lineal. Es decir, llevar el problema de clasificación desde un espacio en el que se encuentran los patrones de entrada a otro espacio en que se utilizan sus características (figura 2.7).

Así pues, a partir de la función $\varphi(\cdot)$, se asume que es posible transformar cada muestra o patrón $\mathbf{x}_k$ a $\mathbf{y}_k = \varphi(\mathbf{x}_k)$, donde para cada una de las n muestras, $k = 1, 2, \ldots, n$,

se permite que $z_k = \pm 1$, dependiendo de que la muestra k pertenezca a la categoría ω_1 o ω_2. De esta forma, la ecuación 2.14 nos ofrece un discriminante lineal del espacio aumentado $\mathbf{y}$, donde tanto el vector de pesos como el vector de muestras transformadas están aumentados ($a_0 = w_0$ y $y_0 = 1$, respectivamente).

$$g(\mathbf{y}) = \mathbf{a}^t\mathbf{y} \tag{2.14}$$

Por tanto, un hiperplano separador asegura:

$$z_k g(\mathbf{y}_k) \geq 1, \;\; k = 1, \ldots, n \tag{2.15}$$

El objetivo del proceso de entrenamiento de una SVM es encontrar el hiperplano separador que posea el mayor margen, a fin de que sea capaz de ofrecer buenas generalizaciones a la hora de clasificar. La distancia de un hiperplano a cualquier muestra $\mathbf{y}$ es $|g(\mathbf{y})| \,/\, \|\mathbf{a}\|$, y asumiendo que existe un margen b positivo, tenemos:

$$\frac{z_k g(\mathbf{y}_k)}{\|\mathbf{a}\|} \geq b, \;\; k = 1, \ldots, n \tag{2.16}$$

Así pues, se persigue encontrar el vector de pesos $\mathbf{a}$ que maximice b. El vector solución puede ser escalado de forma arbitraria y aún así preservar el hiperplano, por lo que para asegurar la unicidad se impone la restricción $b\,\|\mathbf{a}\| = 1$. Por tanto, el objetivo es encontrar una solución a las ecuaciones 2.14 y 2.15, que además minimice $\|\mathbf{a}\|^2$.

Se denominan *Vectores de Soporte* a las transformaciones o patrones de las muestras de entrenamiento para los que aplicando la ecuación 2.15 se obtiene la igualdad, es decir, son aquellas transformaciones que resultan equidistantes al hiperplano separador. Los *Vectores de Soporte* son las muestras de entrenamiento que definen el hiperplano separador óptimo, siendo los patrones más difíciles de clasificar (figura 2.8).

Un sencillo método para entrenar una SVM se basa en seleccionar el patrón peor clasificado, de forma que al final del período de entrenamiento éste se convierta en uno de los vectores de soporte. Así, durante la mayor parte del proceso de entrenamiento, ese patrón se clasifica erróneamente, encontrándose en el lado incorrecto y muy alejado del límite de decisión del momento. En la práctica, este sencillo método es inabordable para la mayoría de problemas, puesto que el esfuerzo computacional asociado a encontrar el patrón que ha sido peor clasificado entre todo el conjunto de muestras de entrenamiento puede resultar excesivo para cada actualización.

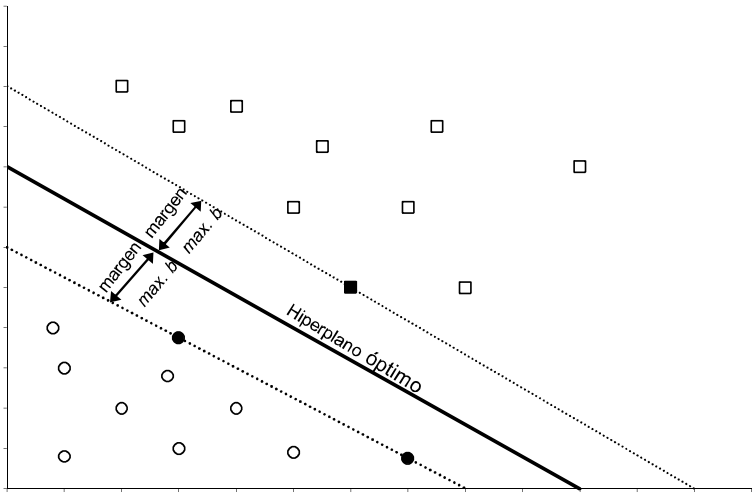

Figura 2.8: Hiperplano separador encontrado durante el proceso de entrenamiento de una SVM. Los vectores de soporte son aquellos patrones más cercanos al hiperplano, a una distancia b, representados mediante puntos sombreados.

Con respecto al error cometido por una SVM, se puede establecer una cota del error de generalización cometido para n patrones o muestras de entrenamiento, teniendo en cuenta el número total de vectores de soporte (N_s):

$$\epsilon[error] \leq \frac{\epsilon[N_s]}{n} \tag{2.17}$$

Este límite del error cometido es independiente de la dimensión del espacio de los vectores de soporte, determinado por $\varphi(\cdot)$. Así pues, uno de los beneficios de las SVM es que la complejidad final del modelo se caracteriza por el número de vectores de soporte empleados en lugar de la dimensión del espacio transformado. Como resultado de esta característica, las SVM suelen tener menos problemas de sobre-entrenamiento que otras técnicas.

2.5. Conclusiones

Las técnicas *Soft Computing* nos ofrecen soluciones de bajo coste, robustas y flexibles, a cambio de cierta imprecisión e incertidumbre. Las herramientas empleadas en este tipo de sistemas poseen ciertas ventajas e inconveniente cuando son empleadas por separado, por lo que su combinación posibilita potenciar sus virtudes minimizando sus desventajas. En la tabla 2.1 se muestra de forma resumida las principales ventajas e

Algoritmos Genéticos	VENTAJAS
	- No necesitan conocimientos específicos sobre el problema que intentan resolver. - Son adaptativos, capaces de solucionar problemas que varíen en el tiempo. - Permiten explorar rápidamente dominios de grandes dimensiones. - Son computacionalmente simples. - Operan de forma simultánea con varias soluciones, en lugar de trabajar de forma secuencial como las técnicas tradicionales.
	INCONVENIENTES
	- Su eficiencia depende en gran medida de la definición de la función objetivo o aptitud con la que se mide la calidad de cada posible solución. - Pueden converger prematuramente. - Pueden tardar mucho en converger, o no converger en absoluto, dependiendo en cierta medida de los parámetros que se utilicen (tamaño de la población, número de generaciones, etc.) - Una codificación errónea puede dar lugar a obtener soluciones para problemas diferentes a los planteados.
Redes Neuronales	VENTAJAS
	- Aprenden a partir de la información que se les proporciona, mimetizando la habilidad de aprendizaje humano. - Pueden aproximar cualquier función multivariable no lineal. - Pueden ofrecer aproximaciones con pocos ciclos de entrenamiento. - No necesitan un profundo conocimiento del proceso o del problema bajo estudio. - Son robustos ante la presencia de datos con ruido. - Una misma red neuronal puede cubrir amplias y diferentes clases de tareas.
	INCONVENIENTES
	- Necesitan largos periodos de entrenamiento o aprendizaje para ofrecer resultados precisos. - No descubren las relaciones internas de las variables físicas y no incrementan el conocimiento acerca del problema. - No existen guías claras acerca del proceso a seguir para determinar qué estructura o tipo de RNA es la apropiada para un determinado problema. - Son propensas a malas generalizaciones
Modelos de Lógica Difusa	VENTAJAS
	- Herramientas eficientes para transformar el conocimiento estructurado humano en algoritmos útiles. - Pueden aproximar cualquier función multivariable no lineal. - Son aplicables cuando los modelos matemáticos son desconocidos o imposibles de obtener. - Operan de forma satisfactoria cuando existe información imprecisa. - Son apropiados para sistemas de toma de decisiones genéricos. - Permiten procesar información numérica y simbólica. - Son apropiados para sistemas de toma de decisiones genéricos.
	INCONVENIENTES
	- Requieren de expertos humanos que conozcan la solución al problema y sean capaces de plasmar su conocimiento de forma jerárquica. - Los expertos pueden tener demasiado conocimiento sobre el dominio del problema o no querer ofrecerlo. - El número de reglas se incrementa de forma exponencial con el incremento del número de entradas y el número de subconjuntos difusos por cada variable de entrada. - El proceso de aprendizaje es más complejo que con redes neuronales.
Máquinas de Soporte Vectorial	VENTAJAS
	- Ausencia de mínimos locales. - Ofrecen buenas generalizaciones. - Su actuación es controlable por medio de optimizar el margen o el número de vectores de soporte empleados. - Estos modelos son menos propensos al sobre-entrenamiento. - Son apropiadas para problemas de clasificación y de estimación de funciones.
	INCONVENIENTES
	- Es necesario seleccionar funciones de *mapping* $\varphi(\cdot)$ adecuadas. - Presentan grandes necesidades de memoria y una alta complejidad de cómputo al enfrentarse a problemas de grandes dimensiones. - Largos procesos de entrenamiento ante problemas de grandes dimensiones - No incrementan el conocimiento acerca del problema

Cuadro 2.1: Ventajas e Inconvenientes de técnicas empleadas en sistemas *Soft Computing*

inconvenientes que ofrecen cada una de las técnicas *Soft Computing* estudiadas en este capítulo.

Los algoritmos genéticos tienen la ventaja de ofrecer un mecanismo adaptativo de resolución de problemas, de forma que aunque el problema cambie, éste se pueda seguir resolviendo. Esta propiedad llevada al límite nos sugiere que esta técnica puede aplicarse a cualquier problema. Sin embargo, para su correcto funcionamiento, los AG requieren que se pueda calcular una función de objetivo o aptitud que guíe la evolución de las generaciones a fin de obtener las soluciones esperadas. Desgraciadamente, no en todos los casos es posible conocer o determinar una función de aptitud adecuada.

Por otra parte, las RNA permiten aproximar todo tipo de funciones no lineales sin necesidad de tener un conocimiento previo del problema a resolver. Son capaces de ofrecer buenas aproximaciones con pocos datos de partida, aunque requieren de un gran volumen de información y largos períodos de aprendizaje si se requieren modelos precisos.

Por tanto, la combinación de algoritmos genéticos con redes neuronales puede producir sistemas de búsqueda efectivos para problemas combinatorios en dominios diversos. Los AG permiten tratar cualquier tipo de problema de grandes dimensiones, mientras que las RNA son un excelente mecanismo de aproximación de funciones. Así pues, mediante esta combinación sería posible aplicar los AG incluso cuando la definición de una función objetivo de forma directa sea demasiado costosa o no sea posible, como en el caso de los dominios que motivaron este estudio. En concreto, en el problema de la catálisis combinatoria, calcular la función objetivo es un proceso con un coste financiero muy elevado. Requiere procesar los posibles nuevos catalizadores y probarlos a fin de determinar sus resultados catalíticos, los cuales condicionan el valor de la función objetivo. En el siguiente capítulo se explicará con detalle éste y otros aspectos del dominio de la catálisis combinatoria.

Capítulo

3

Catálisis Combinatoria

3.1. Introducción

Los catalizadores sólidos se utilizan en la producción de una amplia gama de productos químicos, plásticos y combustibles. La industria química está continuamente buscando nuevos catalizadores eficientes, puesto que los beneficios económicos y operativos que comportan los procesos catalíticos son enormes [Serra, 2004]:

- Requieren menores inversiones.
- Tienen menores costes de operación y mantenimiento.
- Producen productos de mayor pureza y menos productos secundarios o residuos.
- Emplean como reactivos productos de menor coste.
- Son intrínsecamente más seguros.
- Son más respetuosos con el medio ambiente.

Sin embargo, el descubrimiento de nuevos catalizadores heterogéneos consiste en un arduo e impredecible proceso de prueba y error, que requiere largos procesos incompatibles con la creciente dinamización de los mercados y la necesidad de las empresas productoras de fabricar más rápido, más barato y de forma más eficiente [Trimm, 1980].

Tradicionalmente los catalizadores se desarrollan a través de un largo y laborioso proceso que incluye: preparación, caracterización y estudio catalítico de un elevado número de materiales, ya que las variables a considerar hacen que la dimensión del espacio de búsqueda sea enorme. Por ello, la cantidad de tiempo necesaria para obtener un nuevo catalizador eficiente es a menudo de meses o años. Actualmente este largo período de tiempo es inaceptable para la industria, que requiere de procesos de búsqueda más competitivos para responder a la competencia de otros países con mano de obra más barata y con menos restricciones medioambientales.

Por otra parte, cuando se sigue un proceso de búsqueda tradicional, el análisis y procesado de los resultados experimentales de la caracterización y prueba catalítica se lleva a cabo por los investigadores, quienes aplicando su experiencia previa en la materia, emplean su conocimiento e intuición tanto al diseño de nuevas librerías de catalizadores como al análisis de la información resultante, a fin de establecer las relaciones existentes entre los diferentes resultados experimentales. Sin embargo, la interpretación de los resultados puede ser realmente compleja para un humano debido al gran número de variables involucradas y la cantidad de información que se genera. Así pues, es muy probable que el investigador no sea capaz de extraer toda la información sensible que se encuentra latente en los resultados obtenidos.

La tendencia actual consiste es considerar el proceso químico de forma global, tratando de optimizar tanto la composición del catalizador (elementos químicos y porcentaje de los mismos) como las condiciones de reacción (temperatura, presión, reactivos utilizados) de forma simultánea, determinando la combinación que haga mejorar el rendimiento del catalizador. Una misma composición puede hacer enormemente eficaz a un catalizador en unas determinadas condiciones de reacción, mientras que para otras condiciones diferentes su comportamiento sea nefasto.

A fin de abordar el reto de agilizar los procesos de búsqueda en el dominio de los catalizadores heterogéneos se están adoptando técnicas aceleradas de experimentación [1] siguiendo una metodología denominada *Catálisis Combinatoria*[2]([Senkan, 1998; Senkan et al., 1999; Senkan, 2001; Derouane, 2002]), que permiten sintetizar y ensayar un gran número de nuevos materiales de una manera rápida y en paralelo, mediante la aplicación

[1] Los términos anglosajones ampliamente utilizados para estas técnicas son: *high-throughput experimentation* (HTE) y *high-througput screening* (HTS)

[2] Aunque en muchos casos se utiliza de forma indistinta *high-throughput* y combinatoria, en realidad son dos conceptos distintos. HT se refiere a la experimentación intensiva en paralelo, mientras que combinatoria se refiere fundamentalmente a un proceso de diseño de experimentos no basados en conocimiento previo.

de nuevas tecnologías como la automatización, la robótica y la electrónica. Así pues, la investigación de nuevos catalizadores ha conducido hacia una progresiva "*industrialización*" del descubrimiento y desarrollo, haciendo posible obtener librerías de cientos de materiales en un sólo día bajo condiciones de experimentación muy similares a las existentes a escala industrial.

En definitiva, el logro principal de la metodología de trabajo establecida por la catálisis combinatoria es la integración de los siguientes tres elementos: rápida síntesis de bibliotecas, test catalítico de alta capacidad y gestión de información a gran escala. Así se permite la ejecución inteligente y la organización de múltiples experimentos en paralelo. La integración inicial de la síntesis de grandes bibliotecas de materiales y el estudio catalítico en paralelo se está convirtiendo poco a poco en una realidad [Senkan, 2001; Kuntz et al., 199; Corma et al., 2002a; Serra et al., 2003d; Serra et al., 2003e]. Sin embargo, el desarrollo del software requerido para organizar y analizar los datos experimentales, que permitan la retroalimentación del sistema, está aún en vías de desarrollo.

Uno de los retos a los que se enfrenta la catálisis combinatoria es cómo diseñar los experimentos a fin de explorar el mayor número de soluciones posibles, pero reduciendo el coste al mínimo. De hecho, la gran cantidad de materiales a estudiar requiere de un coste financiero tan alto, que es necesario adoptar algún tipo de técnicas que permitan reducir de forma selectiva el número de experimentos a realizar. Así pues, algunas de las técnicas más utilizadas hasta la fecha para este propósito son:

- Procedimientos estadísticos, por ejemplo el diseño factorial [Montgomery, 2001].
- Procedimientos de optimización determinísticos como el *simplex*, la búsqueda holográfica [Végvári et al., 2003] o el *split & pool* [Sun et al., 2002] [Aramendía et al., 2002].
- Procedimientos estocásticos como, por ejemplo, los algoritmos genéticos. Las técnicas estocásticas son de gran utilidad en la optimización de problemas multidimensionales. En concreto, los algoritmos genéticos son particularmente útiles para el descubrimiento de nuevos catalizadores heterogéneos [Kirsten and Maier, 2004].

Otro reto a afrontar por la catálisis combinatoria es la interpretación de la ingente cantidad de información que se produce al emplear técnicas aceleradas de experimentación. El gran número de variables implicadas y la aplicación de algoritmos de optimización

complejos al diseño de experimentos hacen que la interpretación de la información generada sea difícil para los humanos. Así pues, sería de gran ayuda emplear herramientas software que permitan tanto reducir el número de muestras, como ayudar a entender a los investigadores las relaciones latentes en la información obtenida de cada experimento. Por ello, recientemente se han realizado numerosos esfuerzos en el desarrollo y optimización de técnicas de inteligencia artificial aplicadas a la catálisis combinatoria, a fin de permitir la extracción de información y conocimiento de la información generada en la experimentación acelerada, persiguiendo establecer relaciones y modelos entre las variables de entrada y las de salida. En concreto, las técnicas más utilizadas en este sentido son la minería de datos y las técnicas *soft computing*, principalmente redes neuronales y algoritmos genéticos.

3.2. Inteligencia Artificial aplicada a la Catálisis Combinatoria

Las técnicas de minería de datos, aplicadas a la Catálisis Combinatoria, permiten extraer información sensible de datos con múltiples dimensiones. Por tanto, analizan en profundidad la información a fin de obtener el conocimiento latente en ella de forma sistemática, estableciendo relaciones y modelos multi-factores entre las variables de entrada (composición del catalizador, condiciones de preparación y de reacción), las variables de salida (caracterización del catalizador y rendimiento del catalizador) y algunos parámetros teóricos relacionados con los componentes del catalizador. El conocimiento obtenido mediante estas técnicas puede emplearse posteriormente para el diseño de nuevos conjuntos de materiales, de forma que sean probados de manera más racional e inteligente [Corma et al., 2002a; Klanner et al., 2004; Omata et al., 2004]. Existen diversos trabajos que confirman la utilidad de la minería de datos en este sentido [Wang et al., 2001; Yamada et al., 2001; Rajan et al., 2001; Gedeck and Willett, 2001; Corma et al., 2002a; Weaver, 2004]. Así pues, se han aplicado numerosas técnicas de minería de datos al ámbito de la catálisis combinatoria, como por ejemplo modelos de particionado (*clustering*), modelos de regresión no lineal, modelos estadísticos, reglas de asociación, árboles de decisión, reglas de inducción, etc.

Una aproximación que va más allá de la optimización es la combinación de algoritmos capaces de trabajar en múltiples dimensiones con modelos predictivos [Serra et al., 2003c; Klanner et al., 2003; Gilardoni et al., 2003]. Esto permite aplicar el conocimiento extraído de la experimentación previa al diseño de nuevos conjuntos de catalizadores, a fin de que sean estudiados en los siguientes ciclos de optimización.

Las Redes Neuronales (RNA) han sido aplicadas con éxito en el campo de la catálisis, en concreto para modelar y diseñar catalizadores sólidos. Dentro del marco de la catálisis combinatoria existen dos formas de aplicar las RNA:

- En la obtención de modelos de la composición de los catalizadores, donde se establecen las correlaciones existentes entre las variables de composición y síntesis con el rendimiento final del catalizador.
- En el establecimiento de modelos cinéticos que correlacionan las condiciones de reacción con el rendimiento del catalizador.

Las primeras aplicaciones de RNA en el ámbito de la catálisis combinatoria versan sobre el diseño de catalizadores sólidos [Hattori and Kito, 1995] para diferentes reacciones de interés, como por ejemplo el diseño de catalizadores para la amoxidación del propileno [Hou et al., 1997], catalizadores para el acoplamiento oxidativo del metano [Huang et al., 2001] y el análisis y predicción de los resultados de composición de NOx con zeolitas [Sasaki et al., 1995].

También se han aplicado RNA combinadas con estrategias evolutivas en el diseño de catalizadores para la amoxidación del propano [Cundari et al., 2001] o para el descubrimiento de nuevos materiales para la reacción deshidrogenación oxidativa del etano (ODHE) [Corma et al., 2002a], que permite el análisis y la predicción de resultados catalíticos obtenidos mediante técnicas combinatorias. Además, se ha empleado RNA en el modelado de sistemas cristalinos multifase en la síntesis de zeolitas [Moliner et al., 2005]. Con respecto a los modelos cinéticos, existen diversas aplicaciones en las que las RNA se aplican a datos cinéticos experimentales a fin de obtener rápidamente modelos para diversas series de catalizadores y condiciones de reacción y así determinar rápidamente el rendimiento catalítico óptimo para cada material [Bulsari, 1995; Biniwale et al., 2002; Alaradi and Rohani, 2002; Serra et al., 2003b].

Los Algoritmos Genéticos (AG), se han empleado también con éxito en el desarrollo de nuevos catalizadores para la isomerización de la gasolina [Corma et al., 2002b], para la oxidación del monóxido de carbono [Pereira et al., 2005] y la deshidrogenación oxidativa del propano [Wolf et al., 2000].

Existen también otras aproximaciones basadas en técnicas soft computing que se han aplicado con éxito al campo de la catálisis. Por ejemplo, en [Ghosh et al., 2000] y [Sundaram et al., 2001] se aplican técnicas evolutivas al problema del diseño de aditivos para los combustibles. En concreto, se predice el rendimiento de los aditivos mediante

RNA, mientras que un AG, específicamente diseñado para este problema, se emplea para encontrar las estructuras de los aditivos que resulten más convenientes. Otro interesante trabajo es el realizado por [Nandi et al., 2002; Nandi et al., 2004], donde se emplean RNA y AG para la optimización de las condiciones operativas del reactor en la hidroxilación de benceno catalizada por la zeolita titanio silicalita (TS-1). Específicamente, se persigue maximizar la productividad para una determinada reacción catalítica heterogénea dada una composición de catalizador concreta. En el trabajo de [Omata et al., 2003] se optimiza el perfil de temperatura de un reactor de temperatura gradiente para la síntesis de dimetil éter empleando un algoritmo genético binario simple, asistido por una RNA que simula la actividad catalítica. En la búsqueda de nuevos catalizadores sólidos presentada por [Rodemerck et al., 2004], se emplean RNA entrenadas para simular experimentos virtuales a fin de optimizar los parámetros de un AG.

3.3. Conclusiones

En los últimos años, la metodología de la Catálisis Combinatoria ha permitido realizar grandes avances en el estudio y síntesis de nuevos catalizadores. A ello ha contribuido en gran medida la aplicación de técnicas de inteligencia artificial en este ámbito, en especial diversas técnicas *Soft Computing* como los AG y las RNA.

Los algoritmos de optimización iterativos actuales, especialmente los algoritmos genéticos, aplicados al descubrimiento y optimización de catalizadores heterogéneos tienen dos problemas fundamentales. Por una parte, la convergencia de la optimización es lenta, requiriéndose numerosos ciclos experimentales para alcanzar el criterio de convergencia requerido. Además, el rendimiento catalítico del óptimo encontrado puede no satisfacer el criterio de calidad requerido para poder escalar los resultados a otros problemas (*scale-up*), especialmente en optimizaciones sobre espacios de búsqueda de grandes dimensiones como en el caso de formulaciones catalíticas que comprenden colecciones de 30 elementos o variables. Además, numerosas muestras deben ser probadas experimentalmente a lo largo del proceso de optimización, incluso cuando se pueda decir a priori que el rendimiento catalítico esperado vaya a ser suficiente o pobre.

Por otra parte, las propuestas actuales basadas en *Soft Computing* aplicadas al campo de los catalizadores ofrecen diferentes estrategias de optimización para resolver problemas específicos. En concreto, suele tratarse de soluciones a medida que o bien se

centran en optimizar la composición de los materiales, o bien en las condiciones en las que un determinado catalizador ofrece mejores rendimientos. Así pues, no contemplan la posibilidad de estudiar ambos aspectos a la vez, a pesar que el rendimiento de un determinado catalizador depende tanto de su composición como de las condiciones bajo las que se realiza la reacción. En este sentido, las aproximaciones actuales que emplean AG no aplican una codificación de los problemas lo suficientemente general y flexible que sea capaz de plasmar diferentes tipos de problemas dentro de la catálisis combinatoria. Una codificación apropiada debería permitir:

- Definir una formulación catalítica compleja que comprenda diferentes componentes, tales como soportes, promotores de fase activa, potenciadores, etc.
- Establecer reglas o restricciones asociadas (químicas/termodinámicas u orientadas a la aplicación final).
- Indicar otro tipo de variables como los procedimientos de preparación, condiciones de prueba catalítica, etc.

Por tanto, se hecha en falta una herramienta que permita tratar con un amplio rango de problemas de optimización dentro del campo de la catálisis, permitiendo que los investigadores adapten la estrategia de optimización a seguir según sean las características particulares del problema a abordar, así como los recursos disponibles para realizar la investigación (no es lo mismo disponer de reactores capaces de estudiar conjuntos de 25 catalizadores a la vez, que disponer de reactores de capacidades industriales). Además, la citada herramienta debería proponer una arquitectura de búsqueda capaz de incrementar la convergencia ofrecida por las soluciones actuales, sin disminuir la capacidad exploradora que se requiere para abordar los espacios de búsqueda que definen los problemas dentro de la catálisis combinatoria. Asimismo, sería necesario que permitiera plasmar al investigador las características y restricciones particulares del problema catalítico a tratar de forma sencilla y clara.

Para conseguir la herramienta requerida podría emplearse una herramienta que siguiera una arquitectura de búsqueda que combinara algoritmos genéticos (como herramientas de optimización) y redes neuronales (para aproximar modelos multivariable). Esta combinación consistiría en que en cada iteración experimental, el AG obtuviera nuevas generaciones, haciendo uso de las RNA como modelos aproximados para los cálculos de ideonidad (*fitness*) de cada una de las propuestas, así como para reducir el número de experimentos a realizar, llevando a cabo una selección previa de las propuestas que deban ser probadas experimentalmente. Asimismo, los operadores del AG deberían ser

capaces de tratar con una codificación flexible, así como respetar las reglas y restricciones específicas del problema a tratar.

Capítulo

4

Sistemas de Recomendación

Debido al crecimiento de tiendas y artículos disponibles en Internet, existen multitud de posibilidades para elegir un determinado servicio o producto, y además la calidad de éstas varía considerablemente. Por tanto, evaluar las diferentes alternativas existentes a fin de discernir entre ellas resulta un proceso demasiado costoso para el usuario, a pesar de que tenga muy bien definido qué es lo que quiere buscar. Por tanto, en respuesta a las posibilidades y necesidades humanas derivadas del empleo de las nuevas tecnologías Web, han emergido una serie de aplicaciones, denominadas comúnmente *Sistemas de Recomendación*, que tratan de mediar, apoyar o automatizar el día a día de los procesos de intercambio de recomendaciones.

Cuando las personas debemos realizar una elección sobre un producto o servicio del que no tenemos conocimiento previo de su calidad o prestaciones, solemos recurrir a la experiencia o consejos de otros. Así, buscamos personas conocidas familiarizadas con los ítems que perseguimos para que nos recomienden uno de entre las diferentes posibilidades.

Por tanto, los sistemas de recomendación tratan de emular estos mecanismos, siendo

programas que crean modelos de las preferencias de los usuarios o de las tareas que realizan, con el fin de ayudarles a obtener ítems (noticias, webs, discos, libros, viajes, etc.) que el usuario pueda encontrar útiles. En muchas ocasiones, los usuarios realizan de forma explícita sus peticiones (a través de formularios, etc.), si bien existen sistemas que se anticipan a las necesidades de sus clientes, ofreciéndoles una asistencia proactiva [Chesñevar et al., 2006]. En definitiva, nos encontramos ante sistemas de aprendizaje automático, que pueden dedicar una fase durante la cual aprenden y construyen un modelo del comportamiento del usuario y otra en la que aplican dicho modelo para ofrecer recomendaciones en tiempo real [Schafer et al., 2001].

También existen aproximaciones que emplean técnicas más laxas de aprendizaje, que interactúan con el usuario construyendo, actualizando y realizando sus recomendaciones en una única fase. Podemos incluir aquí a muchas de las aplicaciones de recomendación presentes en las redes sociales que se encuentran en Internet. Estas redes están creciendo vertiginosamente en los últimos años, gracias a la posibilidad de conectar a los usuarios con amigos y conocidos por todo el mundo. Así pues, intervenir en una red social empieza por hallar en ella a otros usuarios con quienes compartir intereses, preocupaciones o necesidades comunes. En redes sociales como Facebook [1] o MySpace [2], es posible encontrar pequeñas aplicaciones que te conectan con las opiniones de otros usuarios que conoces sobre libros, películas, canciones, etc. Generalmente, estas aplicaciones se limitan a ofrecer un listado de recomendaciones ordenado por la valoración calculada en base a las opiniones sobre los diferentes ítems de los usuarios conocidos por el usuario.

Pueden encontrarse similitudes entre los sistemas de recomendación y los sistemas de marketing tradicional o los sistemas de ayuda en la toma de decisiones en las cadenas de suministros [Schafer et al., 2001]. La gran diferencia es que los sistemas de recomendación interaccionan directamente con el usuario, centrándose en la búsqueda de aquellos productos o servicios que puedan interesarle. Por contra, los sistemas de marketing ofrecen diferentes métodos o técnicas para que los vendedores animen a los usuarios de distintos segmentos a comprar los artículos que proponen. Asimismo, los sistemas de ayuda en la toma de decisiones ofrecen herramientas para que las compañías decidan sobre el número de productos a producir, cuál debe ser el stock a mantener o a qué tiendas enviarlos, empleando para ello, por ejemplo, la predicción de ventas para un producto en una región, etc.

[1] http://es.facebook.com/
[2] http://www.myspace.com/

A continuación se detallará qué características comprende un acto de recomendación, así como los requisitos de diseño e implementación que poseen los Sistemas de Recomendación, haciendo hincapié en los problemas a abordar cuando se requiere obtener las preferencias de los usuarios. También se citarán los algoritmos más utilizados en los sistemas de recomendación empleados hasta la fecha.

4.1. Modelo de recomendación

Los ejemplos de recomendaciones en nuestra vida cotidiana son numerosos, desde la lectura de la crítica en la cartelera de una revista para decidir qué película ver en el cine, pasando por los consejos de un amigo sobre el último disco de un grupo musical, hasta ese nuevo café de la esquina que siempre está lleno. Tanta popularidad te hace pensar que debe tener algo especial, por lo que acudes a probar su café.

Estos ejemplos pueden servirnos de apoyo para tener más claro el concepto de *recomendación*. Una persona se enfrenta a una *decisión*, cuyo propósito es elegir entre una gran variedad de alternativas. Típicamente el conjunto de posibilidades es tan grande que la persona no conoce ni siquiera todas las alternativas existentes [Munro, 1999]. Si la persona carece del conocimiento suficiente para realizar la elección, seguramente buscará que se le recomiende. Por tanto, podemos considerar que una recomendación es un acto comunicativo [Terveen and Hill, 2001].

Una recomendación puede dirigirse directamente a ciertos individuos, o por contra ser difundida entre todos aquellos que puedan estar interesados. Para la persona que la recibe, una recomendación es un recurso que le ayuda a seleccionar un producto de entre todos los que se le ofrecen. Las recomendaciones ofrecidas por un individuo se basan tanto en sus experiencias como en las de las personas a las que dirige su recomendación. Así, un individuo nunca recomendará a una persona fanática de las novelas policíacas, a la que no le guste la narrativa fantástica, el último libro que haya leído de ese género, aunque le haya parecido el mejor escrito hasta el momento. Por contra, buscará en su memoria para recomendarle un libro de serie negra que le haya parecido nuevo.

Existen formas muy diferentes de recomendar. Así los sistemas de recomendación creados hasta la actualidad varían desde la manera en la que las sugerencias son entregadas a los usuarios, la experiencia con la que se construyen o la manera en la que se presentan [Schafer et al., 2001]. Con respecto a este último punto, nos podemos encontrar con sugerencias, predicciones y valoraciones. Seguidamente se describe cada una de ellas.

Sugerencias: los sistemas pueden recomendar un único producto o servicio, por ejemplo sugiriendo que si le gustó cierto ítem, otro similar también le interesaría. El peligro de recomendar un único ítem reside en que el usuario lo descarte al poseerlo, desperdiciando todo el trabajo y esfuerzo realizado para calcular esa sugerencia. Por ello, muchos sistemas prefieren recomendar un conjunto de ítems para cada contexto. Así, se presentan listas ordenadas según el éxito esperado para cada una de las sugerencias, o bien listas desordenadas de forma aleatoria. Una lista ordenada ofrece más información al usuario, sin embargo es posible que el usuario descarte toda la lista porque no le guste el primer ítem y considere que está mal construida. En definitiva, emplear listas de sugerencias resulta efectivo para recomendar en comunidades en las que interesa atraer a nuevos consumidores, así como para que consumidores habituales conozcan nuevos productos que puedan interesarles.

Predicciones: algunos sistemas ofrecen predicciones de las valoraciones que daría el cliente a un producto o servicio determinado, siendo ésta una forma indirecta de recomendarlos. Las valoraciones pueden corresponder a un cálculo basado únicamente en las preferencias del usuario, o bien tomar como referencia la comunidad a la que éste pertenece.

Valoraciones: en aquellos sistemas en los que las comunidades de usuarios sean pequeñas o en las que los usuarios sean bien conocidos, es posible plantear una herramienta de ayuda para decidirse por un determinado ítem a base de conocer la valoración realizada por otros usuarios. Este tipo de herramienta puede ayudar a los sitios web a aumentar su credibilidad y crear un mayor grado de sentimiento de comunidad. Las valoraciones suelen ser algo más que una simple puntuación, viniendo acompañadas de un texto explicativo de la valoración realizada. Asimismo, que el usuario conozca las preferencias o perfil de aquellos que han valorado los ítems puede serle extremadamente útil a la hora de formarse un opinión acerca de un producto o servicio determinado. Algunos sistemas han planteado ir un paso más allá, permitiendo puntuar las opiniones (*meta rating*), de forma que también se clasifican, ofreciendo al usuario aquellas opciones consideradas como más significativas. Esto resulta útil cuando las comunidades no sean pequeñas y por tanto, existan múltiples valoraciones para cada ítem, siendo muy tedioso el recorrer todas y cada una de las opiniones vertidas sobre él. Así, el usuario puede acceder de forma directa a las opiniones más valoradas, sin necesidad de visitar cada una de las opiniones vertidas para un ítem determinado.

Una vez que el sistema es capaz de recomendar, debe decidir cómo hacer llegar estas recomendaciones a los usuarios. Podemos distinguir principalmente tres formas diferentes de hacerlo [Schafer et al., 2001]:

Técnicas *Push*: permiten llegar al usuario aun cuando éste no esté en contacto directo con el sistema de recomendación. Las tecnologías actuales permiten enviar mensajes a todo tipo de dispositivos móviles, empleando por ejemplo el correo electrónico, los mensajes SMS o MMS, etc. Los servicios de notificación pueden ayudar a las empresas a enviar contenidos personalizados a cada usuario, quien puede apreciarlos como un valor añadido de la empresa, siempre y cuando sean enviados bajo su petición. Desde el lado de la empresa, este tipo de servicios permiten consolidar el número de clientes, puesto que las recomendaciones y ofertas especiales incitan a volver a visitar sus páginas web.

Técnicas *Pull*: el usuario es quien controla cuándo recibir o visualizar recomendaciones, ya sea a través del efecto de su navegación por el sitio web o bien realizando peticiones a través de formularios, etc. Un ejemplo de esta técnica consiste en avisar al consumidor de que tiene disponible un listado de recomendaciones, pero no se le muestra su contenido hasta que el usuario lo solicita de forma expresa. De la misma forma, el usuario podría pedir valoraciones sobre un producto específico, solicitar que se encuentre un regalo que encaje con unas características determinadas o bien pedir recomendaciones sobre una categoría en especial.

Técnicas Pasivas: en este caso las recomendaciones se presentan como parte natural de las páginas Web que componen el portal. Por ejemplo, empleando la parte inferior de la pantalla para mostrar recomendaciones sobre productos relacionados de alguna manera con el producto que el usuario está visualizando en el momento actual; mostrar recomendaciones sobre productos relacionados con el tema del artículo que se está visitando; o bien relacionadas con el sentido de la navegación realizada. Este tipo de técnicas tiene la ventaja de llegar al consumidor cuando está receptivo sobre el tema. Sin embargo, esta forma subliminal puede verse como mera publicidad, perdiendo el valor añadido de una recomendación.

Tal y como se ha citado con anterioridad, las recomendaciones pueden realizarse teniendo en consideración a la persona o comunidad a quién está dirigida, o por contra recomendar productos o servicios atendiendo a otros criterios. Así, se distinguen los siguientes grados de personalización en los sistemas de recomendación [Schafer et al.,

2001]:

Sin personalización: cuando las aplicaciones de recomendación ofrecen recomendaciones idénticas para cada usuario. Sus sugerencias surgen de una mera selección manual o se basan en resúmenes estadísticos u otros datos globales. Un ejemplo de este tipo es la recomendación de la lista de productos más vendidos de una tienda.

Personalización efímera: cuando los sistemas emplean información sobre el comportamiento actual de los usuarios para personalizar sus recomendaciones. Esto es considerado como un primer nivel de personalización, puesto que proporcionan recomendaciones de acuerdo a la navegación y selección de productos realizada por los clientes. Las técnicas empleadas para construir sus recomendaciones no requieren información directa sobre las preferencias de los usuarios, basándose en gran medida en las características y similitudes entre productos.

Personalización persistente: es el nivel alcanzado por todos aquellos sistemas que ofrecen recomendaciones diferentes para cada uno de los clientes que acceden a sus servicios, aunque éstos visiten las mismas secciones o ítems. Para ofrecer un buen servicio, requieren que los usuarios mantengan siempre la misma identidad ante el sistema, así como proporcionar o capturar sus preferencias a largo o corto plazo.

4.2. Preferencias de los usuarios

En general, los sistemas de información personalizados necesitan conocer los intereses de sus usuarios para llegar a realizar sus funciones de forma efectiva [Crabtree and Soltysiak, 1998]. En concreto, esto es esencial para los sistemas de recomendación automáticos, puesto que sin conocer las preferencias de la gente en un determinado dominio, serían incapaces de ofrecer recomendaciones a sus usuarios [Terveen and Hill, 2001]. En los comienzos de este tipo de sistemas, las preferencias se obtenían directamente de los usuarios, siendo éstos los encargados de proporcionarlas mediante diferentes vías (formularios, añadiendo un conjunto de palabras clave, etc.). Posteriormente se comenzó a adaptar y desarrollar nuevas técnicas que permitieran capturar esas preferencias de forma transparente al usuario [Crabtree and Soltysiak, 1998].

Cuando se trabaja con preferencias, se requiere prestar atención a los siguientes aspectos:

Preferencias a emplear: es necesario determinar qué tipo de preferencias serán más interesantes para el sistema entre las diferentes posibilidades de las que se disponga. Por ejemplo, se puede elegir entre aquellas que conciernen únicamente al actual usuario, aquellas correspondientes a los anteriores usuarios del sistema, las preferencias de personas que participan en foros de discusión u otros contextos, etc. La elección viene marcada por diversos factores, como el nivel de personalización que se quiera obtener, la disponibilidad de la información, si se desea modelar las preferencias del usuario a corto o largo plazo, etc.

Obtención de las preferencias: las preferencias de un usuario se obtienen de forma implícita o explícita. Así, o se solicita al usuario que las introduzca directamente, o por contra, éstas son capturadas por el sistema sin que el usuario se percate de ello. Los sistemas que realizan minería de datos social (*data mining*) son un ejemplo de obtención de perfiles de usuario de forma implícita. Así, estos sistemas obtienen las preferencias de los usuarios explorando y analizando toda la información que es almacenada de forma implícita a lo largo de las actividades realizadas por los usuarios normalmente, sin necesitar que el usuario realice ninguna actividad extra [Terveen and Hill, 2001]. Incluso en ocasiones estas técnicas permiten extraer las preferencias de algunos contextos donde los participantes no son conscientes de que su información vaya a emplearse para esa tarea. Así pues, estos sistemas tratan de aplicar técnicas para recopilar y combinar toda la información útil que obtienen, así como diferentes mecanismos de visualización para presentar sus resultados [Amento et al., 2003]. Existen numerosos trabajos que siguen esta línea de trabajo, por ejemplo en [Crabtree and Soltysiak, 1998] se aplican técnicas para capturar los intereses de los usuarios analizando los documentos que escriben y leen. Otro ejemplo es el ofrecido por [Soltysiak and Crabtree, 1998], donde se emplea el *clustering* para la generación automática de perfiles que caractericen los intereses del usuario, tratando que la interacción con el mismo sea mínima. Para ello, analizan aquellos documentos que el usuario lee o escribe, como por ejemplo sus e-mails o las páginas web que visita, para generar vectores que los describan. Éstos son usados como base para agrupar los diferentes documentos, con la idea de que uno de estos grupos puede representar a uno de los intereses del usuario, siempre y cuando contenga un número significativo de archivos.

Incentivos para ofrecer opiniones: los sistemas de recomendación trabajan en entornos en los que los usuarios actúan adoptando diferentes roles, donde frecuentemente son pocos los que ofrecen opiniones y valoraciones en comparación con las personas interesadas en emplear esa información cuando la necesitan. Por otra parte, la gente suele ofrecer sus opiniones sobre los diferentes productos o ítems por muy diversos motivos y no todos ellos lícitos. Se puede confiar totalmente en la opinión vertida sobre un ítem en particular por parte de un usuario desinteresado, sin embargo, es posible que se ofrezcan valoraciones malintencionadas sobre los ítems. Por tanto, es interesante establecer recompensas por realizar valoraciones verídicas, a fin de incrementar la veracidad del sistema.

Representación de las preferencias: el modo en que se representan las preferencias determina las posibles técnicas a emplear, tanto para su recuperación como su posterior utilización. Por ejemplo, una forma sencilla de representar los intereses consiste en emplear una palabra clave por cada preferencia. Sin embargo, esta técnica no es lo suficientemente potente para capturar los diferentes matices que conforman una preferencia concreta. El siguiente paso natural es emplear varias palabras clave, para tratar de concretar más la definición de un interés concreto, dando lugar a descriptores de las mismas. Pero esta aproximación no soluciona el problema de cómo definir cuándo un descriptor representa adecuadamente o no una preferencia concreta. Existen otras aproximaciones donde se combina la utilización de palabras clave, pesos, probabilidades, grafos ponderados, etc [Crabtree and Soltysiak, 1998].

4.3. Importancia de la ontología empleada

Para que se pueda compartir y reutilizar el conocimiento formalmente representado entre diferentes entidades, es útil definir un vocabulario común mediante el cual se intercambie la información. De este modo, una ontología es una descripción formal y explícita de los conceptos que describen sus características y atributos, así como de las posibles restricciones a los valores que tomen. Es decir, es una definición de las clases, relaciones, funciones, roles y demás objetos presentes en un dominio. En definitiva, una ontología es una especificación concreta de un concepto [Gruber, 1993].

Así pues, las clases las constituyen conceptos que representan cualquier cosa sobre la que se habla, o bien, son descripciones de una tarea, función, acción, estrategia o proce-

so de razonamiento. Asimismo, las relaciones representan un tipo de interacción entre conceptos de un dominio. Las funciones constituyen un caso especial de relaciones en las cuales el enésimo elemento de la relación es único para los elementos precedentes. Además, pueden aparecer axiomas que modelen aquellas sentencias que sean siempre verdaderas dentro del dominio. Éstos se emplean con diferentes propósitos, por ejemplo para definir el significado de los componentes de una ontología, estableciendo restricciones complejas sobre los valores de los atributos. Por último, las instancias se utilizan para representar a elementos o individuos específicos.

Algunas de las razones que conducen a la necesidad de desarrollar una ontología son:

- Compartir cómo está estructurada la información que se desea intercambiar entre personas o entidades [Musen, 1992; Gruber, 1993].
- Permitir la reutilización del conocimiento del dominio.
- Hacer explícitas las suposiciones sobre un dominio.
- Separar el conocimiento del dominio del conocimiento operacional [McGuinness and Wright, 1998].
- Analizar el conocimiento del dominio. Una vez se dispone de una especificación declarativa de los términos del dominio, es posible realizar su análisis. El análisis formal de los términos es extremadamente valioso cuando se pretende tanto reutilizar ontologías existentes como cuando se intenta extenderlas [McGuiness et al., 2003].

La preferencia del usuario sobre los productos puede verse influenciada por los atributos disponibles para describir los artículos, que varían enormemente según sea su naturaleza (textiles, libros, discos, etc.), contemplando desde campos numéricos como el precio hasta campos descriptivos como la marca [Guan et al., 2005]. Por tanto, la ontología empleada para describir los artículos juega un importante papel en el descubrimiento de las preferencias del usuario, puesto que constituye un punto de partida para el mismo.

Por ejemplo, pueden emplearse ontologías basadas en el producto, donde se categorizan y se clasifican (mapean) los productos heterogéneos de forma taxonómica [Guan et al., 2005].

En las propuestas actuales uno de los mayores problemas es el tratamiento de los atributos no cuantificables que describen a un producto. La mayoría de la propuestas son demasiado generales y sólo tratan de entender las preferencias del usuario a través de la generalización y el estereotipado en lugar de aprender las preferencias específicas

de los usuarios. Por otra parte, en numerosas ocasiones, los sistemas de recomendación propuestos sólo son capaces de tratar con un conjunto predefinido de atributos, que guían tanto el diseño del sistema como su implementación. Así pues, si cambia la descripción del producto, el sistema no es capaz de trabajar con los nuevos atributos.

Existen técnicas que tratan de mejorar los sistemas de filtrado y recomendación de productos basados en la detección de atributos genéricos de los productos, que no han sido previamente identificados y que sin embargo son diferentes ante los usuarios. Por tanto, persiguen entender la preferencia genérica de un usuario, a fin de mejorar las recomendaciones ofrecidas al cliente. Por ejemplo, Guan et al. [Guan et al., 2004; Guan et al., 2005] proponen un sistema capaz de recomendar una lista de productos acordes con las preferencias del agente. Es capaz de detectar tanto atributos no cuantificables como cuantificables, a través de un sistema de realimentación en el que se requiere de la intervención del usuario. Por tanto, detecta características relevantes que no habían sido consideradas en la ontología del sistema, a través de un campo descriptivo del producto. Así, el sistema es capaz de adaptarse ante cambios en la descripción de los productos o nuevas incorporaciones de los mismos. Para determinar cuándo los atributos detectados son relevantes para el usuario y en qué grado, el sistema emplea un algoritmo genético con codificación binaria.

4.4. Escasez de Información

En general, uno de los retos a los que se enfrentan los sistemas de recomendación es afrontar la escasez de información en alguna de sus etapas (*Sparsity problem*). La mayoría de algoritmos de recomendación necesitan conocer las valoraciones de los usuarios sobre los diferentes ítems, así como sus preferencias. Sin embargo, los usuarios son reticentes en invertir su tiempo en realizar estas valoraciones o en describir detalladamente sus preferencias. Debido al modelo de caja negra seguido por la mayoría de técnicas de recomendación, el usuario es incapaz de entender la importancia de esos pasos, siendo para él una mera pérdida de tiempo.

Por tanto, en la mayoría de ocasiones, deben tenerse en cuenta estrategias que permitan completar los modelos necesarios, como por ejemplo la monitorización de la actividad cotidiana de los usuarios, lo cual conlleva otra clase de inconvenientes. Así, en el caso de la monitorización, a pesar de minimizarse el esfuerzo necesario por parte del usuario, se genera un gran volumen de información que puede dar lugar a descriptores adulterados

acerca de los intereses de los usuarios [Chesñevar et al., 2006].

4.5. Escalabilidad y rendimiento

Dentro de los sistemas de recomendación, la escalabilidad está relacionada tanto con el gran tamaño de los problemas a resolver como con los requisitos de tiempo real necesarios para solucionarlos [Schafer et al., 2001]. Dentro de las principales medidas de rendimiento nos encontramos con: el tiempo de espera máximo, el número de peticiones de recomendaciones atendidas de forma simultánea, el número de usuarios, la cantidad de productos o servicios sobre los que recomendar y, finalmente, el número de valoraciones realizadas que gestionar.

Para abordar los problemas de escalabilidad que presentan los sistemas de recomendación, pueden adaptarse muchas técnicas empleadas en minería de datos, como la reducción de dimensiones y el paralelismo. Sin embargo, estas técnicas deben ser modificadas para trabajar con los requisitos de simultaneidad y tiempo de espera exigidos [Schafer et al., 2001]. Un gran reto para adaptar estas técnicas es, como se ha mencionado con anterioridad, la enorme dispersión de las valoraciones a las que suelen tener que enfrentarse los sistemas de recomendación.

4.6. Algoritmos de Recomendación

Existen principalmente dos tendencias diferentes a la hora de afrontar el problema de la recomendación. Por una parte, los algoritmos basados en la premisa de que las preferencias de los usuarios tienden a permanecer inalteradas con el paso del tiempo, sin evolucionar de forma drástica [Chesñevar et al., 2006]. Dentro de este caso podemos citar como ejemplo los métodos basados en el contenido y el filtrado por colaboración ítem a ítem (*Item-to Item Collaborative Filtering*), los cuales tratan de encontrar ítems similares a los ya obtenidos o valorados por los usuarios en el pasado [Sarwar et al., 2001].

Por otra parte, existe un conjunto de algoritmos que se basan en la correlación de las preferencias de los usuarios [Chesñevar et al., 2006]. Así, muchos de los algoritmos de recomendación parten de un conjunto inicial de consumidores que compran y valoran ítems similares a los comprados y valorados por el usuario. Una vez se determina el

conjunto de partida, éste se emplea para añadir ítems a la lista de recomendación del usuario procedentes de las listas de recomendación de los consumidores pertenecientes al citado conjunto, eliminando aquellos productos que el usuario ya haya comprado o valorado [Resnick et al., 1994]. El filtrado por colaboración (*Collaborative Filtering*) [Herlocker et al., 2000; Good et al., 1999] y los modelos basados en clusters (*Cluster Models*) son dos de las versiones más populares de este tipo de algoritmos.

Finamente, existen otras aproximaciones híbridas, en las que se emplean combinaciones de diferentes técnicas de recomendación de ambas tendencias, tratando de obtener sistemas que recojan las ventajas de diferentes algoritmos de recomendación disminuyendo sus inconvenientes.

Seguidamente se detallarán los algoritmos de recomendación más empleados, tanto para los sistemas que conciben las preferencias inalterables, como para aquellos que establecen correlaciones entre preferencias, así como para los sistemas híbridos que combinan más de una técnica para ofrecer sus recomendaciones.

4.6.1. Métodos basados en el contenido

Los sistemas basados en el contenido o en la búsqueda emplean únicamente las preferencias del usuario procurando recomendarle ítems similares a los ítems que le gustaron en el pasado [Terveen and Hill, 2001; McGuinness and Wright, 1998; Guan et al., 2005; Maes, 1994]. Tratan el problema de la recomendación como una búsqueda de ítems relacionados, intentando encontrar los ítems que tengan características comunes a los ítems ya comprados y valorados por el cliente (mismo autor, mismo director, etc.). La aproximación basada en el contenido tiene sus orígenes en los algoritmos de recuperación y de filtrado de la información, empleando muchas de sus técnicas.

Los sistemas de recuperación de información (*Information Retrieval*) [Good et al., 1999] permiten que el usuario realice consultas para seleccionar ítems que se encuentren en su área de interés. Para ítems no textuales, estos sistemas emplean palabras clave, categorizaciones, etc. Sin embargo, estos sistemas no utilizan ningún mecanismo que capture y utilice las preferencias de los usuarios. Por tanto, básicamente consisten en la búsqueda, procesamiento e indexado del contenido y las consultas realizadas por los usuarios. La búsqueda suele consistir en acceder a páginas y servicios Web, así como a sistemas de archivos donde poder encontrar la información.

Un siguiente paso lo constituyen los sistemas de filtrado de información (*Information*

Filtering) [Good et al., 1999], que buscan en el contenido semántico y sintáctico de los ítems para determinar cuáles pueden ser interesantes para el usuario de acuerdo al perfil que se tiene de él. Los sistemas más simples obtienen el perfil preguntando directamente al usuario, mientras que los más avanzados construyen el perfil del usuario aprendiendo sobre sus preferencias. Estas técnicas no requieren de otros usuarios en el sistema, permitiendo usuarios aislados.

Los sistemas basados en el contenido emplean algoritmos que aprenden las preferencias de los usuarios y filtran, de los ítems posibles, aquellos que más se asemejan a éstas. Así, las preferencias se aprenden de forma explícita a través de la interacción directa del usuario, por ejemplo, indicándole que valore como buenos o malos una serie de ítems. También suelen emplear técnicas implícitas, por ejemplo fijándose en qué enlaces son visitados por los usuarios entre los ofrecidos y cuánto tiempo dedican a su visita.

Generalmente las preferencias son representadas mediante perfiles de los intereses del usuario en un determinado dominio. También pueden emplearse palabras clave ponderadas mediante diferentes pesos. Tanto para el aprendizaje como para la representación de las preferencias es adecuado emplear técnicas basadas en el aprendizaje automático y de recuperación de información [Terveen and Hill, 2001].

Si el usuario tiene pocas compras o valoraciones, los algoritmos de búsqueda son escalables y ofrecen un buen rendimiento. Sin embargo, para los usuarios con miles de compras, resulta inviable realizar todas las consultas necesarias para tener en cuenta todos los productos [Maes, 1994]. Por ello, el algoritmo debe utilizar un subconjunto o compendio de la información, lo que conlleva una reducción en la calidad de las recomendaciones, o bien realizar parte del cómputo con anterioridad.

De todas formas, las recomendaciones ofrecidas pueden ser demasiado generales o demasiado concretas. Este comportamiento es poco adecuado cuando lo que se persigue es que los sistemas de recomendación constituyan una herramienta adecuada para que el consumidor encuentre y descubra nuevos productos interesantes para él. Normalmente es imposible satisfacer este objetivo a través de un sistema que recomienda los ítems más populares de un mismo autor o que pertenezcan a la misma área temática, ya que las recomendaciones se restringen sobre conjuntos limitados de productos.

Por otra parte, al basarse en muchas de las técnicas de recuperación de información, estos sistemas no pueden tratar con todo tipo de ítems. Existen algunas características que no pueden recuperarse de la descripción de los productos y que sin embargo influyen enormemente en la decisión de los usuarios. Por ejemplo, para el caso de evaluar páginas

Web, la valoración de su apariencia, su estética o su facilidad de uso, no puede extraerse a través de un mero análisis de su contenido.

Dentro de esta categoría se enmarca el sistema *Handy Broker* presentado por [Guan et al., 2002] para la búsqueda y recomendación de productos en aplicaciones de comercio electrónico sobre dispositivos móviles, empleando un sistema adaptativo para capturar las preferencias del usuario. Para adaptarse a las variaciones en las preferencias del usuario a lo largo del tiempo emplea tanto la realimentación dada por el usuario como sus propias observaciones. La aplicación busca a través de la red los productos que se adecuan a las preferencias aprendidas del usuario. La descripción de las preferencias del usuario se realiza mediante funciones que valoran a cada producto. Los parámetros de las funciones son ajustados inicialmente mediante heurísticas, evolucionando a lo largo del tiempo a través de un algoritmo genético, para que caractericen en todo momento las preferencias del usuario para un tipo de productos en particular. Así, cada producto es valorado por estas funciones, empleando para ello sólo los atributos que sean cuantificables, por ejemplo su precio.

En el *MIT Media Lab*, Shearin et al. [Shearin and Lieberman, 2001] desarrollaron una herramienta destinada a ayudar al usuario en la búsqueda de apartamentos de alquiler que se adecuaran a sus necesidades. En concreto, el agente *Apt Decision* emplea técnicas de aprendizaje por refuerzo a fin de construir un perfil de las preferencias del usuario, que puede ser guardado y empleado en búsquedas futuras. En el proceso seguido para obtener el perfil, cada característica de un apartamento tiene un peso base. Los pesos de cada una de las características cambian cuando el usuario las selecciona o elimina el conjunto de características que persigue. Por otra parte, algunas características se añaden o eliminan de forma automática, siguiendo un proceso que se basa en la comparación de aquellas características presentes en algunos apartamentos que selecciona el usuario y que no aparecen en su perfil actual. Además, el sistema mantiene un historial de los cambios realizados, siendo capaz en cualquier momento de restablecer un estado anterior.

Otro ejemplo de sistema de recomendación que se basa en el contenido es *TopicShop* [Amento et al., 2003] cuyo principal objetivo es proporcionar al usuario un mecanismo para encontrar y evaluar sitios web que coincidan con sus intereses. Así, *TopicShop* analiza la estructura y contenido de las páginas web a fin de proveer la información requerida (buscando qué páginas web son más mencionadas por otras personas), proporcionando además una interfaz que permite explorar y tratar la información obtenida.

Así pues, este sistema trata de encontrar aquellos sitios recomendados por la mayoría de los usuarios, otorgándoles una valoración que recoge la opinión de éstos.

4.6.2. Filtrado por colaboración basada en los ítems

Los sistemas de recomendación basados en el filtrado por colaboración ítem a ítem (*Item-to-item Collaborative Filtering)* buscan productos similares a los ya comprados y valorados por el cliente [Sarwar et al., 2001]. Normalmente, los sistemas combinan estos productos para formar una lista de recomendaciones [Linden et al., 2003]. La utilización de este tipo de aplicaciones de recomendación posibilita que las empresas respondan a los intereses actuales de sus clientes, permitiendo las asociaciones naturales entre diferentes productos para guiar a los consumidores en la compra correcta [Schafer et al., 2001]. Para establecer la combinación más similar para un determinado ítem, se construye de forma off-line una "tabla de ítems similares", encontrando ítems que los consumidores tienden a comprar de forma conjunta. La similitud entre dos ítems se determina, por ejemplo, a partir del coseno del ángulo entre dos vectores [Sarwar et al., 2001], donde cada vector corresponde con un ítem (función 4.1). Así, el vector representativo de cada producto tendrá M dimensiones correspondientes a los clientes que lo han comprado. Por tanto, esta aproximación puede verse como una variación de los métodos de recomendación basados en el contenido (sección 4.6.1) [Burke, 2002].

$$similarity(\vec{A}, \vec{B}) = \cos(\vec{A}, \vec{B}) = \frac{\vec{A} \bullet \vec{B}}{\|\vec{A}\| * \|\vec{B}\|} \tag{4.1}$$

La computación off-line de la tabla de similitudes tiene un coste computacional elevado [Linden et al., 2003], con $O(N^2M)$ en el peor caso, siendo N el número de productos del catálogo y M los clientes que han comprado. Sin embargo, en la práctica se encuentra cercano al $O(NM)$, ya que para muchos consumidores el número de compras realizadas es bajo.

Dada una tabla de similitudes, el algoritmo encuentra ítems similares para cada una de las compras y valoraciones realizadas por los usuarios, agrega estos ítems y después recomienda los más populares o correlacionados. Este cómputo se realiza muy rápidamente y sólo depende del número de ítems que el usuario compró o valoró.

Resumiendo, estos sistemas permiten trabajar con grandes bases de datos, tanto de productos como de clientes. Son capaces de ofrecer rápidamente recomendaciones en

línea, puesto que gran parte del cálculo se realiza con anterioridad. Este tipo de sistemas reaccionan de forma inmediata ante cambios en la información disponible de un usuario. Además, las recomendaciones ofrecidas a los clientes son independientes del número de compras o valoraciones realizadas por ellos. Como las relaciones entre los ítems son relativamente estáticas, los algoritmos basados en los ítems pueden proporcionar la misma calidad que los basados en los usuarios, pero con menos cálculo en línea [Sarwar et al., 2001].

El sistema de recomendación de URLs ofrecido por [Chalmers et al., 1998] es un ejemplo de este tipo de sistemas, considerando las URLs como los ítems a recomendar. En concreto este sistema usa la secuencia de URLs seguida durante una sesión de navegación como unidad de partida para realizar sus recomendaciones. Calculan las similitudes entre las diferentes secuencias de URLs almacenadas para realizar sus recomendaciones, por ejemplo, para recomendar páginas que han sido visitadas por otros usuarios que hayan accedido a páginas parecidas a las visitadas por el usuario.

Un exitoso ejemplo es el sistema de personalización de la tienda *on-line* para cada cliente realizado por Amazon.com[3], el famoso sitio de compras de libros, discos, electrónica, etc. Así, el aspecto del portal cambia según sean los intereses del usuario. Además, amazon.com emplea las recomendaciones como una herramienta de marketing para campañas publicitarias a través del e-mail. El algoritmo de filtrado colaborativo "ítem a ítem"desarrollado por esta compañía es capaz de tratar con la ingente cantidad de información de la que disponen, produciendo recomendaciones de gran calidad en tiempo real [Linden et al., 2003]. Los clientes de Amazon.com, a través del enlace "Tus recomendaciones", acceden a una sección en la que pueden filtrar sus recomendaciones tanto por línea de producto como por área temática, puntuar los productos recomendados, valorar los productos previamente comprados, así como conocer por qué le han sido recomendados los diferentes productos de la lista.

Otro ejemplo de sistema de recomendación *on-line* que sigue este algoritmo es el proyecto Pandora[4], que a partir de un artista o canción dada, busca canciones similares y se las propone al usuario, creando una radio on-line personalizada. El usuario puede valorar favorable o desfavorablemente cada canción propuesta, así como conocer por qué le ha sido recomendada. Conforme el usuario valora más canciones, el sistema refuerza el aprendizaje de las preferencias del usuario, ofreciéndole mejores recomendaciones. Para realizar sus recomendaciones, un equipo de más de cincuenta analistas musicales

[3] www.amazon.com
[4] http://www.pandora.com/

analizan las canciones a fin de obtener todas las características que resulten relevantes. Todo este trabajo de clasificación comenzó en el año 2000, dentro del proyecto Music Genome[5], del que forman parte músicos y amantes de las tecnologías musicales, liderados por Tim Westergren.

4.6.3. Filtrado por colaboración basada en los usuarios

Los sistemas de filtrado por colaboración (*Collaborative Filtering*)[Good et al., 1999; Herlocker et al., 2000] predicen la afinidad por algunos bienes o información, conectando los intereses almacenados de la persona con los intereses almacenados de aquellas personas que se esperan sean semejantes a ella. Así, estos sistemas construyen una base de datos para encontrar usuarios con opiniones similares. Realizan predicciones sobre la opinión de un usuario respecto a un ítem determinado a través de la combinación de las opiniones de otros individuos de los que se espera tengan formas de pensar parecidas.

Las decisiones del filtrado proceden de los propios usuarios. Cada usuario de un sistema de este tipo valora aquellos ítems sobre los que tiene conocimiento para establecer su perfil de intereses. Recientemente algunos sistemas han comenzado a inferir las preferencias del usuario a través de sus acciones en lugar de requerir que el usuario puntúe de forma explícita cada ítem [Terveen and Hill, 2001; Linden et al., 2003].

Por tanto, el sistema conecta al usuario con aquellas personas con las que comparte preferencias, intereses o gustos. Las valoraciones de esas personas son empleadas para realizar las recomendaciones a los usuarios.

Uno de los puntos fuertes de los sistemas de filtrado por colaboración es la sencillez del patrón conceptual de operación, análogo al sistema de recomendación humano del boca a boca, denominado "modelo conceptual de caja blanca" (*White Box conceptual Model*). En concreto, estos sistemas funcionan siguiendo tres sencillos pasos: primero el usuario introduce el perfil de sus valoraciones; seguidamente el sistema localiza a las personas con perfiles similares (sus vecinos); y finalmente las valoraciones de los vecinos se combinan para dar lugar a las recomendaciones.

Así pues, el algoritmo genera recomendaciones basadas en aquellos consumidores más similares al usuario. La similitud se mide de diferentes formas [Sarwar et al., 2000; Linden et al., 2003], si bien las más común es emplear el coseno del ángulo entre dos vectores A y B (función 4.1), que representen a dos consumidores (cuyas componentes

[5]http://www.pandora.com/mgp.shtml

corresponden a los ítems comprados por ellos).

Asimismo, los sistemas seleccionan las recomendaciones a partir de los clientes similares mediante diversas vías, siendo común puntuar cada ítem de acuerdo al número de clientes que lo compraron, alcanzando mayor puntuación cuanto más exitosa haya sido su venta.

El filtrado por colaboración constituye una parte importante de un sistemas de recomendación, permitiendo descubrir nuevos ítems de interés para el usuario, siempre y cuando éstos hayan sido del agrado de otros usuarios similares a él. Además, permiten realizar buenas recomendaciones en el caso en el que se desconozcan o estén ocultos los atributos de mayor interés para el usuario. Para que un sistema de este tipo funcione correctamente, es imprescindible que muchos usuarios evalúen cada ítem, ya que los nuevos ítems no pueden ser recomendados antes de que algunos usuarios se hayan tomado su tiempo en evaluarlos. Por tanto, si el número de usuarios del sistema es relativamente inferior al volumen de información presente en el sistema, existe el peligro de que no se tenga una cantidad suficiente de valoraciones, siendo éstas muy dispersas, disminuyendo la colección de ítems que puedan ser recomendados [Terveen and Hill, 2001; Konstan et al., 1997].

Por otra parte, un sistema de filtrado por colaboración se centra en sugerir aquello que los usuarios similares al destinatario han sugerido. No se centra en conocer qué es lo que realmente quiere el usuario actual [Konstan et al., 1997]. Por tanto, sería muy interesante combinar los resultados ofrecidos por un sistema de filtrado por colaboración con los resultados ofrecidos en un sistema de recomendación basado únicamente en el conocimiento de las preferencias del usuario.

La utilización del filtrado por colaboración para la obtención de recomendaciones tiene un coste computacional elevado [Guan et al., 2005]. Así, es de $O(MN)$ en el peor caso (siendo M el número de consumidores y N el número de ítems presentes en el catálogo de productos), puesto que se examinan M consumidores y hasta un máximo de N productos para cada uno de ellos. Sin embargo, como muchos de los clientes compran muy pocos productos, en comparación al enorme tamaño de los catálogos, el coste medio de examinar un cliente tiende a ser $O(M)$. Pero como hay algunos clientes que compran o realizan valoraciones sobre un porcentaje significante de productos del catálogo, se requiere un coste de $O(N)$. Por tanto, el coste computacional del algoritmo es aproximadamente $O(M + N)$. A pesar de ello, para grandes conjuntos de datos (por ejemplo en el caso de portales con millones de usuarios) el algoritmo se enfrenta a

grandes problemas de rendimiento y escalabilidad.

A fin de solucionar estos problemas surgieron los sistemas basados en algoritmos de *clustering*[Ungar and Foster, 1998], que pueden considerarse como un caso especial de sistemas colaborativos, donde las recomendaciones también se basan en encontrar consumidores con gustos parecidos a los del usuario, pero que se diferencian en el modo en el que se determinan esos usuarios similares. Los algoritmos de *clustering* construyen conjuntos de clientes similares, de forma que se centran en las compras y valoraciones realizadas por éstos para realizar las oportunas recomendaciones a un usuario considerado similar al citado conjunto. Para encontrar estas particiones de clientes similares, los algoritmos de *clustering* dividen la base de datos de los clientes en diversas particiones, tratando el problema como si fuera un problema de clasificación [Linden et al., 2003].

Típicamente las particiones se crean empleando algoritmos no supervisados, aunque algunas aplicaciones determinan las particiones de forma manual. Empleando una métrica de similitud, un algoritmo de *clustering* agrupa a los clientes más similares a fin de construir los denominados *clusters* o particiones. Debido a que resulta impracticable realizar un *clustering* de forma optimizada sobre grandes bases de datos, muchas aplicaciones emplean algoritmos voraces basados en la obtención de generaciones de *clusters* [Bradley et al., 1998]. Normalmente estos algoritmos comienzan con un conjunto inicial de particiones, que generalmente contienen un único cliente seleccionado de forma aleatoria. Después van asignando los clientes restantes a la partición a la que sea más similar, permitiendo la creación de nuevas particiones o la unificación de las existentes. Para bases de datos verdaderamente grandes y especialmente aquellas con un gran número de dimensiones, se hace imprescindible aplicar el muestreo o reducir el número de dimensiones estudiadas.

Una vez el algoritmo consigue determinar los segmentos adecuados, calcula la similitud del usuario a los vectores que representan cada uno de los segmentos, clasificando al usuario en el segmento más parecido a él. Algunos algoritmos clasifican al usuario en más de un segmento, describiendo el grado de pertenencia a cada uno de ellos [Ungar and Foster, 1998].

Los algoritmos de *clustering* obtienen mejor rendimiento y son más escalables on-line que los algoritmos tradicionales de filtrado por colaboración, ya que comparan al usuario con un conjunto de particiones en lugar de con toda la base de datos de consumidores [Breese et al., 1998]. Además, la parte pesada del cálculo, es decir, la obtención de

los diferentes *clusters*, se realiza off-line. Sin embargo, el inconveniente de esta técnica consiste en que las recomendaciones sugeridas no son demasiado certeras, al estar poco personalizadas [Schafer et al., 2001]. Esto puede ser debido a que los algoritmos de *clustering* agrupan a un gran número de clientes en un mismo *cluster* y a pesar de que a un usuario se le asigne ese *cluster*, no significa que todos y cada uno de los clientes que lo conforman sean totalmente similares a él. Por tanto, las recomendaciones que surgen del conjunto de estos clientes son menos relevantes que las producidas por un algoritmo basado únicamente en el filtrado por colaboración. Siempre puede mejorarse la calidad de las particiones obteniendo particiones de grado más fino.

Por otra parte, los sistemas de filtrado por colaboración resultan convenientes en entornos de entretenimiento [Resnick et al., 1994; Hill et al., 1995; Balabanović and Shoham, 1997; Terveen et al., 1997], si bien no son adecuados en entornos con contenidos de alto riesgo. Debido al proceso seguido para obtener las recomendaciones, no existe la certeza de que éstas sean correctas y, además, el usuario no dispone de sistemas para determinar cuándo creer en una recomendación o cuándo dudar de ella.

En el mismo sentido, recientes estudios determinan que los sistemas de recomendación que emplean modelos de colaboración basados en usuarios son sensibles a recibir ataques consistentes en la incorporación de información falsa [Williams et al., 2006; O'Mahony et al., 2004; O'Mahony et al., 2005; O'Mahony et al., 2006; Lam and Riedl, 2004]. Para ello, los atacantes necesitan tener cierto acceso al conocimiento del dominio, por ejemplo el rango empleado en la valoración de los productos, qué productos son actualmente los más populares, cuáles gustan o disgustan más, etc. Desgraciadamente, en la mayoría de los sistemas actuales es posible acceder a este tipo de información de forma sencilla. Existen principalmente dos tipos de ataques, denominados comúnmente como *"Push"* y *"Nuke"*, cuyos objetivos son, respectivamente, promocionar o degradar los ítems objeto de ataque, por medio de modificar las predicciones o puntuaciones obtenidas para ellos. Para esto, los atacantes asumen un número de identidades dentro del sistema a atacar, creando un perfil de usuario (denominado perfil de ataque) para cada una de ellas. De esta manera, la información atacante se introduce en el sistema de la misma forma que lo hace la información proveniente de usuarios reales. Por todo ello, cada vez se está dedicando más esfuerzo a estudiar la manera de combatir este tipo de ataques, así como de detectar otras posibles vías de engaño [Burke et al., 2006; Williams et al., 2006; O'Mahony et al., 2005]. Por ejemplo, una posible solución es tratar de detectar qué perfiles coinciden con un modelo de ataque, a fin de no ser considerados por los algoritmos que computan las recomendaciones.

GroupLens es un ejemplo de sistema de recomendación basado en el filtrado colaborativo aplicado al dominio de los grupos de noticias [Resnick et al., 1994][Konstan et al., 1997]. Los usuarios, a través de una sencilla puntuación, indican si la noticia es de su agrado o no. Las recomendaciones consisten en la predicción de la valoración que daría el usuario sobre un determinado artículo tras haberlo leído.

El sistema experimental PHOAKS (*People Helping One Another Know Stuff*) es un ejemplo de sistema de recomendación que sigue un modelo colaborativo [Terveen et al., 1997]. PHOAKS consiste en una arquitectura no dependiente del dominio para filtrar información de mensajes electrónicos y un conjunto de técnicas para generar y administrar interfaces dinámicas basadas en la web. En concreto, ha sido utilizado para reconocer y procesar recomendaciones sobre recursos Web y mensajes FAQ. Una característica que lo distingue de otros sistemas es la utilización de roles, asumiendo que los roles de los que ofrecen recomendaciones y aquellos que las reciben son diferentes y se encuentran especializados. PHOAKS se centra en reutilizar las recomendaciones presentes en las conversaciones on-line existentes, a fin de evitar trabajo a aquellos que recomiendan y a los usuarios finales.

4.6.4. Métodos Híbridos

Los sistemas de recomendación híbridos combinan dos o más técnicas de recomendación con el fin de obtener un mejor rendimiento y reducir los inconvenientes que presentan las técnicas al ser empleadas por separado. En su mayoría, los sistemas que utilizan esta estrategia combinan un sistema de filtrado por colaboración con alguna otra técnica, a fin de solventar sus deficiencias ante la escasez y dispersión de la información o para obtener recomendaciones más personalizadas. Además, dependiendo de la forma en la que se realice la combinación, es posible obtener diferentes resultados aunque se empleen las mismas técnicas. Seguidamente se describen algunas de las fórmulas más utilizadas a partir de la descripción realizada por [Burke, 2002].

Ajustable: Un sistema híbrido ajustable (*Weighted*) es aquel donde la valoración obtenida por un ítem recomendado por el sistema se computa a partir de todas las técnicas de recomendación del sistema, ajustando el peso que se le da a la respuesta de cada una de ellas en la puntuación final. Mediante este mecanismo se refinan los resultados variando el peso específico dado a cada una de las técnicas, por ejemplo teniendo en cuenta la respuesta del cliente ante las recomendaciones

recibidas. Un ejemplo de este tipo de combinación es el sistema de recomendación de noticias on-line *P-Tango* [Claypool et al., 1999], basado en la combinación de filtros colaborativos y métodos basados en el contenido.

Intercambiable: En este tipo de sistemas híbridos, conocidos por su nomenclatura inglesa *Switching*, existe un determinado criterio para elegir entre las diversas técnicas de recomendación que estén presentes en el sistema, a fin de seleccionar la más adecuada en cada caso a la hora de ofrecer una recomendación. Por tanto, introducen una complejidad adicional al ser necesario determinar el criterio de selección a emplear. Sin embargo, mediante este sistema se evitan los inconvenientes y potencian las ventajas de las diferentes técnicas, por ejemplo eludiendo emplearlas cuando las condiciones no son idóneas para ellas. Por ejemplo, el sistema *Daily Learner*, que permite acceder a las noticias de forma personalizada, emplea diferentes técnicas según se pretenda recomendar un ítem siguiendo preferencias a corto o a largo plazo [Billsus and Pazzani, 2000]. En concreto, en la construcción del perfil de usuario a corto plazo se emplean solamente las últimas valoraciones realizadas, así como las observaciones más recientes de su comportamiento. El algoritmo del vecino más próximo es empleado tanto para detectar las noticias parecidas en las que el usuario está actualmente interesado, como para descartar las noticias que ya conoce. Con respecto al modelo a largo plazo, el sistema pretende capturar las preferencias generales del usuario empleando para ello un algoritmo de aprendizaje probabilístico, en concreto un clasificador bayesiano Naïve [Duda and Hart., 1973], para calcular la probabilidad de que las noticias resulten de interés para el usuario, considerando que contienen una serie de características que las definen.

Mezcla: Son sistemas en los que se muestra conjuntamente una combinación de las recomendaciones obtenidas a partir de diferentes métodos. Las posibles inconsistencias deben resolverse empleando alguna regla de arbitraje, por lo que se añade un nivel de complejidad al tener que decidir qué reglas emplear. Los sistemas híbridos mezclados (*mixed*) se emplean cuando se pretende obtener una lista de recomendaciones sobre los ítems más adecuados para un determinado caso. Los autores [Smyth and Cotter, 2000] ofrecen un claro ejemplo con su sistema *PTV*, el cual realiza recomendaciones sobre la programación televisiva combinando estrategias de recomendación basadas en el contenido y técnicas colaborativas.

Combinación de características: En este caso se emplea un sistema de recomen-

dación basado en el contenido aplicado sobre un conjunto de datos ampliado con información obtenida mediante filtrado por colaboración, considerándola como si fuera una característica más asociada a cada ejemplo (*Feature Combination*). Estos sistemas híbridos permiten considerar los datos obtenidos de forma colaborativa sin centrarse exclusivamente en ellos, reduciendo la sensibilidad del sistema ante el número de usuarios que hayan valorado un ítem, etc. Además, permite que el sistema tenga información sobre la inherente similitud entre ítems que de otra forma permanecería oculta ante el sistema colaborativo. Un ejemplo de este tipo es el sistema *Ripper* de [Basu et al., 1998], empleado en la recomendación de películas.

En cascada: en un sistema híbrido en cascada se aplican por pasos las diferentes técnicas de recomendación que lo compongan. Así, se aplica una primera técnica para producir un primer conjunto de candidatos, que serán refinados con la aplicación de una segunda técnica, etc. El objetivo principal de la segunda técnica consiste en discriminar de forma efectiva aquellos ítems de los que no se haya obtenido un resultado claramente positivo o negativo. Puede establecerse alguna condición de parada, de forma que se considere que el conjunto de candidatos es lo suficientemente refinado para dejar de aplicar otros métodos de recomendación sobre él. El sistema de recomendación de restaurantes *EntreeC* [Burke, 2002] es un ejemplo de sistema híbrido en cascada que combina métodos basados en el conocimiento con técnicas colaborativas.

Aumento de Características: En este caso el sistema híbrido aplica una determinada técnica para producir una valoración o clasificación para un ítem y esa información es usada dentro del proceso de cómputo empleado por la siguiente técnica de recomendación (*Feature Augmentation*). Se diferencian de los sistemas híbridos basados en la combinación de características en que en esos sistemas los datos que se combinan proceden de diferentes fuentes.

Por otra parte, tanto en el caso de los sistemas híbridos en cascada como en los de aumento de características, se emplea una secuencia de técnicas de recomendación. No obstante, existen diferencias entre ambos métodos de combinación. Así, en los sistemas con aumento de características, las características usadas en el cómputo de la segunda técnica de recomendación empleada incluyen las salidas del primer algoritmo. Sin embargo, al emplear un método en cascada, el segundo algoritmo de recomendación no usa ninguna salida ofrecida por la primera técnica empleada para obtener valoraciones, sino que los resultados de ambos se priorizan de alguna

forma.

Un ejemplos de sistema híbrido de este tipo es el sistema *LIBRA* [Mooney and Roy, 2000] que realiza recomendaciones de libros basadas en el contenido sobre la información que el sistema de Amazon.com genera empleando sus sistemas colaborativos [Linden et al., 2003] (autores relacionados, títulos relacionados, etc.). Otro ejemplo es la mejora que realizó [Sarwar et al., 1998] sobre el sistema de filtrado de noticias *GroupLens* [Konstan et al., 1997].

Meta-nivel: en este tipo de sistemas híbridos, el modelo obtenido por una técnica es empleada como entrada por la siguiente (*Meta-level).* Por ejemplo, en un sistema híbrido que combine mediante esta fórmula un algoritmo basado en el contenido y otro del tipo colaborativo, el modelo obtenido por la primera técnica consiste en una representación comprimida de los intereses del usuario. De esta forma, la siguiente fase colaborativa opera sobre esta densa representación de la información más fácilmente que sobre la información típica de valoraciones que se dispone de partida. Uno de los primeros sistemas híbridos de este tipo fue el sistema multi-agente adaptativo de recomendación de páginas web *FAB* [Balabanović and Shoham, 1997; Balabanović, 1997]. La arquitectura de este sistema consta de tres elementos principales: agentes recopiladores (*collection agents*), que se encargan de encontrar páginas de un determinado tema; agentes seleccionadores (*selection agents*), que obtienen páginas para un usuario concreto; y un *router* central que actúa como comunicador entre los agentes y los usuarios. Cada agente mantiene un perfil, basado en las palabras contenidas en las páginas Web que valora. El perfil de un agente recopilador representa el tema en el que se ha especializado, mientras que para el caso del agente seleccionador, el perfil representa los intereses de un usuario en concreto. Las páginas encontradas por un agente recolector se envían al *router* central, quien las reenvía a todos aquellos usuarios cuyos perfiles coincidan en cierta medida, de forma que estas páginas puedan ser de su interés. Los perfiles de los agentes se actualizan a través de las valoraciones que realizan los usuarios acerca de las recomendaciones que reciben. De manera adicional, todas aquellas páginas que reciben una alta puntuación son recomendadas a todas las personas que tengan un perfil similar al usuario.

4.7. Conclusiones

A lo largo de este capítulo se han presentado los requisitos de diseño e implementación que poseen los Sistemas de Recomendación, haciendo hincapié en los problemas a abordar cuando se requiere obtener las preferencias de los usuarios. Asimismo, se han comentado los algoritmos más utilizados en los sistemas de recomendación empleados hasta la fecha. En la tabla 4.1 se muestra un resumen de las principales ventajas e inconvenientes de los mismos.

Uno de los principales inconvenientes de los algoritmos basados en el contenido y de los algoritmos de filtrado por colaboración ítem a ítem, es que no son capaces de ofrecer recomendaciones novedosas a sus usuarios. Es decir, no pueden proponer nuevos productos o servicios que sean totalmente diferentes a los ya valorados o comprados por el usuario y que puedan resultarle interesantes. Este aspecto sí es abordado por los sistemas de filtrado por colaboración basada en los usuarios y por los sistemas basados en algoritmos de *clustering*.

Otro problema común de los algoritmos presentados consiste en afrontar la escasez y dispersión de la información en alguna de sus etapas, sobre todo cuando entra un usuario nuevo en el sistema. Los sistemas que emplean el filtrado por colaboración ítem a ítem son menos sensibles a este problema, puesto que a partir de una única valoración son capaces de ofrecer nuevas recomendaciones. Sin embargo, para que ofrezcan recomendaciones realmente útiles para el usuario, requieren que éste valore o compre numerosos artículos. Por tanto, los sistemas de recomendación requieren de técnicas que les permitan ofrecer recomendaciones aceptables para el usuario en todo momento, aunque no se disponga de un gran volumen de información.

Así pues, con el fin de minimizar los inconvenientes y maximizar las ventajas que ofrecen los algoritmos presentados, han aparecido soluciones híbridas donde se combinan dos o más técnicas de recomendación. Los resultados y recomendaciones obtenidas dependen en gran medida de la forma en la que se realiza esta combinación, puesto que es posible obtener diferentes resultados empleando las mismas técnicas combinadas de forma distinta.

Por otra parte, muchos de los algoritmos presentados son utilizados en los Sistemas de Recomendación empleados por empresas dedicadas al comercio electrónico para sugerir productos a sus consumidores que se adecúen a sus preferencias, a fin de incrementar su volumen de ventas. Sin embargo, estos algoritmos no dotan de mecanismos para

Basados en el Contenido. Aprenden las preferencias de los usuarios y recomiendan los ítems que más se asemejen a éstas.	VENTAJAS
	- Escalables y con buen rendimiento cuando el usuario tiene pocas valoraciones. - No requieren de otros usuarios en el sistema permitiendo usuarios aislados.
	INCONVENIENTES
	- Inviables cuando el usuario tiene miles de valoraciones. Emplean subconjuntos de la información disponible, reduciendo la calidad de las recomendaciones. - Las recomendaciones ofrecidas pueden ser demasiado generales o demasiado concretas. - No son adecuados para que el usuario descubra productos diferentes y atractivos para él. - No toman en cuenta las características de los productos que no pueden recuperarse de su descripción.
Filtrado por colaboración ítem a ítem. Buscan productos similares a los ya valorados por el cliente.	VENTAJAS
	- Permiten que las empresas respondan a los intereses actuales de sus usuarios. - Permiten trabajar con grandes bases de datos ya que gran parte del cómputo se realiza off-line. - Ofrecen recomendaciones rápidas en línea a los usuarios. - Pueden recomendar ítems a usuarios con pocas valoraciones.
	INCONVENIENTES
	- No son adecuados para que el usuario descubra productos diferentes y atractivos para él. - No contemplan que las preferencias de los usuarios pueden variar. - No son adecuados para sistemas a emplear por usuarios que deseen atacar diferentes bases de datos (búsquedas globales en Internet).
Filtrado por colaboración basada en los usuarios. Sus recomendaciones se basan en las valoraciones de consumidores con gustos similares al usuario.	VENTAJAS
	- Son adecuados para que el usuario descubra productos diferentes y atractivos para él. - Son capaces de recomendar ítems aunque se desconozca o estén ocultos los atributos que resultan relevantes para el usuario.
	INCONVENIENTES
	- Requieren que muchos usuarios evalúen cada ítem para que pueda ser recomendado a terceros. - No ofrecen buenos resultados si disponen de valoraciones muy dispersas. - Recomendaciones poco personalizadas. No se centra en conocer qué es lo que realmente quiere el usuario actual. - Ofrece problemas de rendimiento y escalabilidad en el caso de enfrentarse a grandes volúmenes de información. - No son adecuados en entornos de alto riesgo - Sensibles a recibir ataques consistentes en la incorporación de información falsa
Basados en algoritmos de clustering. Sus recomendaciones se basan en las valoraciones de conjuntos de consumidores considerados similares al usuario.	VENTAJAS
	- Son adecuados para que el usuario descubra productos diferentes y atractivos para él. - Son capaces de recomendar ítems aunque se desconozca o estén ocultos los atributos que resultan relevantes para el usuario. - Obtienen mejor rendimiento y son más escalables on-line que los algoritmos tradicionales de filtrado por colaboración - La parte pesada de su cálculo se realiza off-line
	INCONVENIENTES
	- Recomendaciones poco personalizadas. Menos relevantes que las obtenidas por un sistema basado únicamente en el filtrado por colaboración. - En base de datos enormemente grandes deben reducir el número de dimensiones estudiadas - No son adecuados en entornos de alto riesgo - Sensibles a recibir ataques consistentes en la incorporación de información falsa

Cuadro 4.1: Ventajas e inconvenientes de los principales algoritmos de recomendación

determinar el perfil de las características de los productos más deseadas o valoradas por los usuarios, de forma que las empresas fueran capaces de diseñar y ofertar productos y servicios más atractivos para sus clientes, asegurándose así su venta. Por tanto, sería muy interesante para las empresas el disponer de una herramienta capaz de abordar este problema de combinatoria, en el que se persigue conocer la combinación de atributos de un producto o servicio óptimos para un cliente, incluso cuando no se disponga de mucha información sobre sus gustos. Por tanto, para determinar esa combinación ideal podría aplicarse la técnica de búsqueda basada en técnicas *Soft Computing* que se presentará en este trabajo, capaz de abordar problemas combinatorios de grandes dimensiones.

Parte II

Propuesta

Capítulo

5

Arquitectura propuesta

El marco de trabajo presentado en este capítulo ha sido diseñado para ser aplicado a la búsqueda y optimización de problemas de grandes dimensiones en dominios diversos. Está basado en técnicas *Soft Computing*, combinando algoritmos genéticos y redes neuronales, siendo fruto de la evolución de las soluciones planteadas anteriormente en nuestros trabajos previos [Corma et al., 2002a; Valero et al., 2004b; Corma et al., 2005; Serra, 2004; Valero et al., 2004a].

Las redes neuronales (RNA) y en concreto los perceptrones multicapa, son empleados como modelos aproximados de las funciones de aptitud o *fitness* utilizadas por el algoritmo genético (AG). El AG es el encargado de encontrar la solución óptima al problema planteado mediante la exploración de un conjunto de soluciones posibles de forma simultánea. LA decisión de emplear perceptrones multicapa está basada en diferentes aspectos como su sencillez, su capacidad de dar respuestas de forma rápida, su capacidad de ofrecer buenas aproximaciones aunque se disponga de poca información de partida y el dominio del problema tenga múltiples dimensiones, así como que es necesario aportar ningún tipo de conocimiento previo acerca de la posible correlación

existente entre las variables involucradas en el sistema.

La arquitectura propuesta comienza con la obtención de una generación de partida idónea, generada de forma aleatoria. Esta primera generación es evaluada con la función original para obtener el *fitness* de cada uno de los individuos que la forman (por ejemplo, en el caso de un problema de búsqueda de nuevos catalizadores, este paso consistiría en probar las muestras propuestas directamente en el reactor). Después, la generación es utilizada para construir un modelo adecuado de la función de *fitness* mediante una RNA. En la siguiente etapa, el AG emplea la RNA obtenida para aproximar la función de *fitness* necesaria para calcular la aptitud de las siguientes generaciones que propone (sin necesidad de emplear la función original, con el beneficio económico y temporal que esto puede suponer). En concreto, el AG desarrollado emplea el valor de *fitness* de cada individuo en su operador de cruce, tanto para seleccionar a los individuos que actuarán como progenitores como para cruzar a cada individuo (este operador se trata con mayor detalle en la sección 5.4).

En la figura 5.1 pueden apreciarse con mayor detalle las etapas que conforman la técnica *Soft Computing* propuesta:(i) configuración, (ii) re-entrenamiento de la RNA, (iii) modelado de la función de *fitness*, (iv) aplicación de los operadores del AG, (v) evaluación simulada y (vi) evaluación de control. Las etapas comprendidas entre la ii y vi se repiten hasta que se alcanza el criterio de convergencia establecido. Sin embargo, el número de pasos y acciones a realizar en cada etapa dependen de la configuración de los parámetros de actuación de la arquitectura propuestas. Por ejemplo, es posible obtener varias generaciones antes de obtener una generación de control (pasando para ello por los pasos v y vi), es decir, una generación en la que el *fitness* de sus individuos sea calculado por medio de la función original.

Seguidamente se explicará de forma detallada los aspectos más relevantes de cada uno de los pasos o etapas que conforman la arquitectura de búsqueda.

5.1. Configuración

El problema a estudiar debe ser codificado apropiadamente en la fase de configuración. Después, se establece el comportamiento de la aproximación *Soft Computing* fijando el valor de sus parámetros de configuración. Seguidamente se obtiene la generación de partida para el AG y finalmente, se obtiene una RNA que aproxime adecuadamente la función de aptitud o *fitness* a emplear.

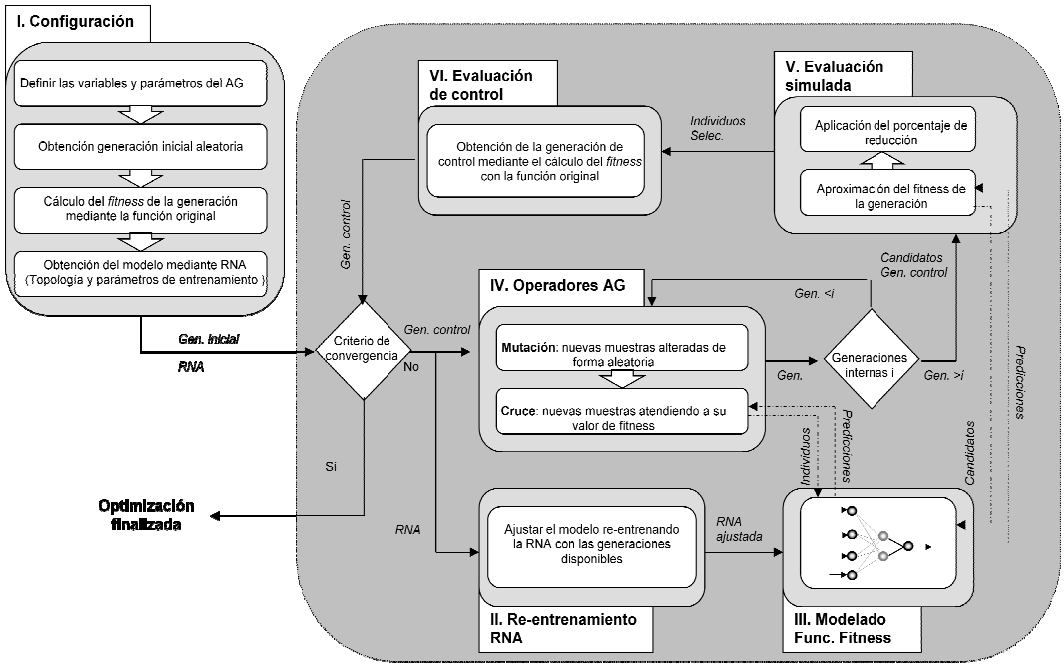

Figura 5.1: Etapas de la arquitectura de búsqueda *Soft Computing* propuesta

5.1.1. Codificación del problema

Los AG son unos algoritmos muy potentes siempre y cuando la codificación seleccionada refleje adecuadamente el problema a optimizar, puesto que en caso contrario el AG puede no converger o converger hacia falsos óptimos, que solucionen un problema distinto del planteado. Por tanto, la correcta codificación del problema es crucial para el buen funcionamiento de la técnica. En la arquitectura propuesta se ha elegido emplear codificación real [Goldberg, 1991] en lugar de la codificación binaria clásica, fundamentalmente por dos aspectos. En primer lugar, porque estudios previos muestran que los algoritmos genéticos que emplean genes codificados mediante números reales ofrecen mejor rendimiento que los codificados de forma binaria [Eshelman and Schaffer, 1993]. En segundo lugar, porque la codificación numérica es menos artificiosa que la binaria, y puede emplearse de forma más natural por los usuarios finales de la arquitectura (por ejemplo, ingenieros químicos).

El sistema de codificación propuesto (figura 5.2) representa cada posible solución como un cromosoma jerarquizado, en el que sus genes o elementos se encuentran representados en dos clases de agrupaciones: condiciones y compuestos. Un mismo cromosoma puede contener más de una condición y compuesto a la vez, o estar formado únicamente

por una condición o por un compuesto. Mediante las condiciones se expresan aquellas variables que sean independientes entre sí, mientras que a través de un compuesto se define a un conjunto de variables que representen porcentajes de un todo (los valores que tomen deben sumar cien). Cada condición está divida en uno o más tipos, y éstos a su vez en uno o más subtipos que agrupan los elementos. De forma similar, los compuestos están definidos por una o más secciones, y éstas a su vez están formadas por una o más subsecciones que agrupan a los elementos que finalmente forman el compuesto. A través de los diferentes niveles de agrupación propuestos es posible establecer una serie de guías y restricciones que dirigen el proceso de optimización. Así pues, se establecen los valores máximos y mínimos que pueden alcanzar la suma de elementos que contienen, así como el número máximo de elementos o subniveles que pueden estar seleccionados y tomar valor a la vez en cada cromosoma. De este modo se hace posible atacar problemas combinatorios en los que no sólo deba establecerse el valor de las variables a estudiar, sino también el conjunto adecuado de las mismas. Además, para cada variable bajo estudio (representada por medio de los elementos) puede establecerse el máximo y mínimo valor que pueden adoptar, así como la cantidad en la que su valor puede incrementarse o decrementarse.

Esta codificación permite abordar todo tipo de problemas combinatorios pertenecientes a diferentes dominios. Por ejemplo, dentro de la catálisis combinatoria sería posible atacar problemas que tratasen de optimizar la composición idónea de un catalizador, las condiciones óptimas para su actuación, las mejores condiciones de preparación o todos estos aspectos a la vez. Así pues, la formulación de un catalizador podría representarse como un cromosoma que posea un compuesto, cuyos elementos describan los elementos químicos presentes en el mismo y que se agrupen por medio de las secciones y subsecciones según sea la función que realicen (promotores, soportes, etc). A través de las secciones y subsecciones sería posible establecer la compatibilidad entre los diferentes elementos.

La figura 5.3 muestra un ejemplo de codificación para un problema de búsqueda de la formulación idónea de un catalizador basado en oro. Puede observarse cómo la codificación establece que una muestra concreta puede estar formada por oro, dos tipos de promotores y un soporte. Para el primer promotor, se permite elegir entre una de sus dos subsecciones (metales o metales nobles), y del mismo modo, elegir sólo un elemento cada vez. De esta forma es posible agrupar los elementos de manera que sea posible evitar combinaciones de los mismos no deseadas (ya sea por incompatibilidad o por razones económicas, etc.). Del mismo modo, la codificación nos ofrece dos posibles promotores

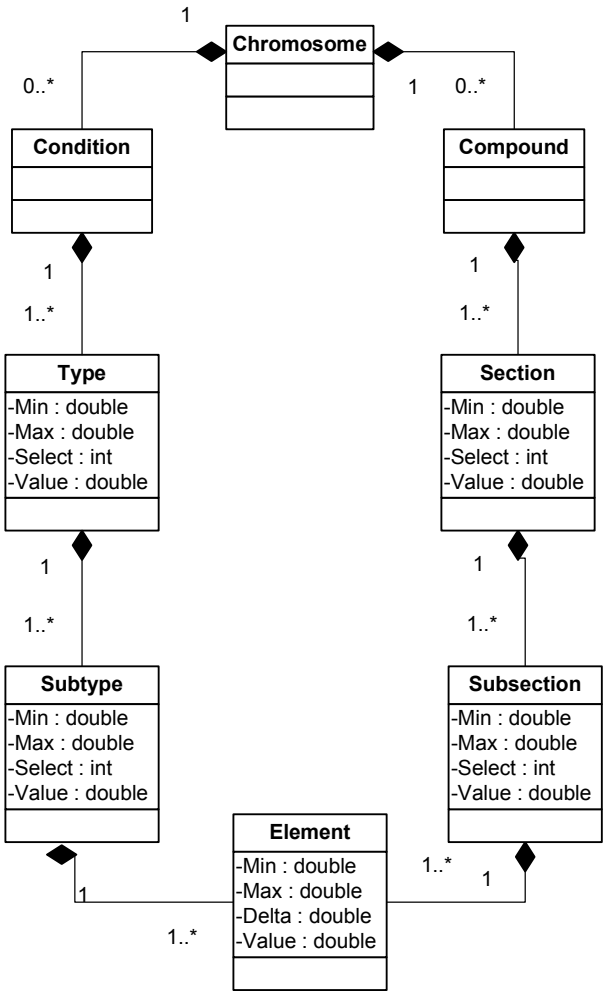

Figura 5.2: Esquema de codificación de un cromosoma

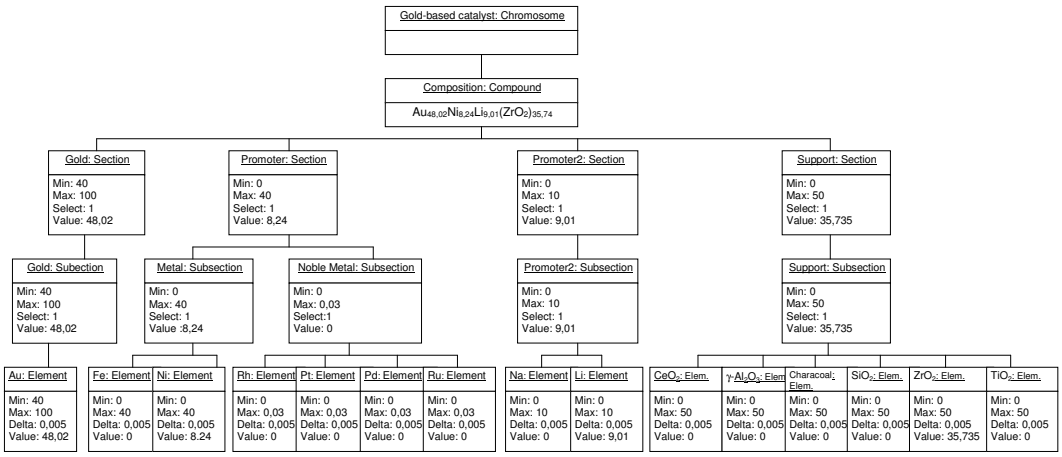

Figura 5.3: Ejemplo de codificación para una formulación general de un catalizador basado en oro

del segundo tipo, siendo posible elegir solamente uno en cada ocasión. Igualmente, sólo puede elegirse un elemento de entre los seis posibles como soporte. Además, como el valor mínimo establecido para las secciones correspondientes a los promotores y soporte es cero, es posible que un catalizador propuesto no contenga siempre todos los tipos de elementos.

5.1.2. Configuración de los parámetros Soft Computing

La arquitectura propuesta permite configurar la manera en la que se lleva a cabo el proceso de optimización a través de seis parámetros diferentes, a fin de que el usuario establezca la forma de trabajo que más se ajuste a sus intereses. Seguidamente se comentarán en qué medida afecta cada parámetro al comportamiento de la herramienta *Soft Computing* propuesta.

Generaciones Internas. Mediante este parámetro se fija el número de generaciones propuestas por el AG (generaciones internas) antes de llevar a cabo la evaluación simulada y obtener la generación de control definitiva (que será evaluada por la función original). Para obtener una generación interna, el AG emplea únicamente la aproximación de la función de aptitud que le proporciona la RNA para calcular el *fitness* de cada individuo cuando le es necesario (con el beneficio económico y temporal que comporta). Sin embargo, debe tenerse en cuenta que la frecuencia en la que debe obtenerse una generación de control depende en gran medida de la precisión del modelo aproximado empleado. Si el modelo de RNA obtenido

no es demasiado preciso, será conveniente fijar este parámetro con valores bajos, puesto que el error introducido en el sistema por el modelo puede hacer que el AG converja hacia falsos óptimos si se generan muchas generaciones sin realizar un ajuste del valor de la calidad alcanzada por las generaciones. Por ello, puede establecerse si se desea que el valor de este parámetro sea fijo o por contra, se incremente de forma lineal o exponencial. Además, es posible modificar el valor de este parámetro durante el proceso de optimización. Por tanto, la arquitectura permite seguir un control adaptativo [Jin, 2005], de forma que obtenga más generaciones internas conforme su modelo de aproximación sea más preciso.

Población virtual. Este factor indica el número de individuos propuestos por el AG en cada generación. Cuanto mayor sea el tamaño de la población, más rápidamente convergerá (en términos del número de generaciones necesarias). Asimismo, si el tamaño de la población no es suficientemente grande, la convergencia hacia el máximo global es menos probable. Por otra parte, cuanto mayor sea el tamaño de la población, mayor coste computacional requerirá la técnica de búsqueda propuesta. Por tanto es necesario establecer mediante este parámetro el equilibrio necesario entre rapidez de convergencia y esfuerzo computacional a realizar.

Ratio de reducción. Mediante este parámetro se establece el porcentaje de reducción a realizar sobre la población virtual para obtener los individuos que formarán parte de la generación de control (ecuación 5.1) .

$$Ratio\ de\ Reducción\,\% = 1 - \frac{Tamaño\ de\ Población\ de\ Control}{Tamaño\ de\ la\ Población\ Virtual} * 100 \qquad (5.1)$$

Si el ratio de reducción se establece en 0 %, la evaluación simulada no se lleva a cabo y todos los individuos propuestos por el AG son evaluados con la función de aptitud original. Por contra, si se establecen valores muy grandes para este parámetro, es posible que el número final de individuos de control sea insuficiente para un correcto funcionamiento del sistema de optimización. En este caso, el error introducido por la RNA al aproximar la función de aptitud puede interferir demasiado en el comportamiento del AG, ya que muchas de las aproximaciones del *fitness* de los individuos nunca serían contrastadas con los valores de control obtenidos en el proceso de evaluación.

Probabilidad de Mutación. El operador de mutación tiene un carácter fundamen-

talmente explorador que busca nuevas soluciones y previene que el AG converja de forma prematura, asegurando la diversidad de la población (sección 2.1.3. Sin embargo, la mutación puede interferir en el rendimiento y evitar que el AG converja adecuadamente. Por esta razón, son deseables valores bajos para estos parámetros, en concreto dentro del rango [5 %..20 %].

Número de genes a mutar. Mediante este parámetro se selecciona el número de elementos que deben ser modificados cuando se selecciona un individuo para ser mutado. El operador de mutación tendrá más impacto e incorporará mayor diversidad en la población con valores altos de este parámetro.

Parámetro $\alpha \in [0, 1]$. Mediante este valor se establece el tamaño del intervalo de confianza usado por el operador de cruce, afectando al rendimiento de este operador. El operador de cruce propuesto tiene carácter explorador y explotador. Mediante un α de 0.5 se establece un equilibrio entre sus capacidades exploradoras y explotadoras, mientras que valores altos de α representan un incremento en su aptitud exploradora y viceversa.

Ratio de progenitores. El número final de individuos a seleccionar como progenitores se establece por medio de este parámetro. Los progenitores (los mejores individuos) se emplean para calcular los intervalos de confianza usados por el operador de cruce propuesto.

5.1.3. Obtención de la Generación Inicial

Como punto de partida del algoritmo de optimización propuesto se obtiene un conjunto inicial de individuos a través de un proceso que garantice la diversidad de la población inicial. Este proceso consiste en la creación de diversas generaciones aleatorias, llevando a cabo un estudio estadístico de la generación a fin de elegir la más diversa. La diversidad de la generación inicial asegura que el proceso de optimización parta de una población que abarque la máxima información posible de todo el espacio de búsqueda. La generación inicial es evaluada por la función de aptitud original, con lo que se calcula la aptitud o *fitness* de cada muestra y la calidad alcanzada por la generación.

5.1.4. Obtención del modelo de RNA

La calidad del modelo de aproximación debe mejorarse todo lo posible, teniendo en cuenta la limitación de información disponible para obtener el modelo y las grandes dimensiones del espacio de entrada. Por tanto, se analizan diferentes factores involucrados en el comportamiento ofrecido por la RNA, como la topología de la RNA, los algoritmos de entrenamiento y las funciones de activación. Diferentes experiencias que siguen este procedimiento metodológico, así como los resultados obtenidos pueden consultarse en los siguientes estudios previos [Serra et al., 2003a; Valero et al., 2004b; Moliner et al., 2005; Corma et al., 2002a].

Para llevar a cabo los experimentos citados, se emplea la generación inicial junto con los valores de *fitness* calculados para cada muestra, que la RNA a obtener debe ser capaz de aproximar. Como el número de muestras de partida suele ser muy reducido, se utiliza la validación cruzada en la construcción del conjunto de datos de entrenamiento, tal y como se explica en [Bishop, 1996]. Así, el conjunto de entrenamiento se divide de forma aleatoria en diferentes subconjuntos de muestras de entrenamiento (80 %) y test(20 %). Por tanto, cada experimento se lleva a cabo con diferentes combinaciones de conjuntos de entrenamiento y test, teniendo en consideración los resultados medios de las mismas.

Para obtener un modelo óptimo se aplica un método incremental usando aprendizaje supervisado, probando diferentes topologías de redes neuronales basadas en el perceptrón multicapa. Comenzando con una sola capa oculta y unas pocas neuronas, la topología se modifica incrementando el número de neuronas (sin superar en número al doble de los datos de entrada) y probando con una y dos capas ocultas respectivamente.

Además, se llevan a cabo diferentes experimentos con aquellos algoritmos de entrenamiento que se espera que ofrezcan un mejor comportamiento con los perceptrones multicapa de acuerdo a la literatura [Bishop, 1996]. En concreto, las redes neuronales se entrenan con los algoritmos *backpropagation* y *backpropagation momentum*, probando con diferentes parámetros de entrenamiento (factor de aprendizaje η y *momentum* μ). Se han seleccionado estos algoritmos de aprendizaje puesto que la literatura indica que se comportan bien para la mayoría de problemas al entrenar a perceptrones multicapa [Ripley, 1996][Duda et al., 2001].

Finalmente, se estudia qué funciones de activación se comportan mejor en los nodos ocultos, si las sigmoidales logísticas o las sigmoidales tangenciales [Bishop, 1996].

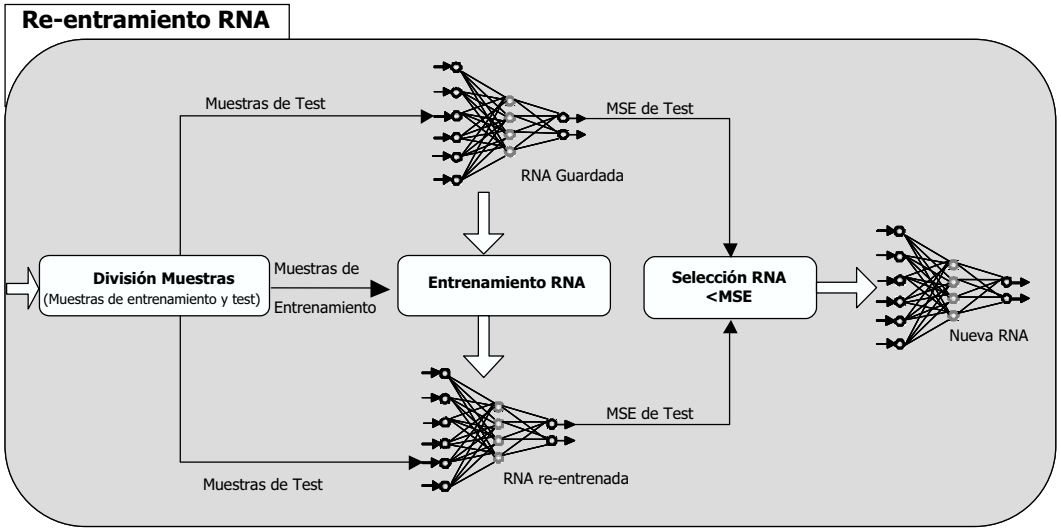

Figura 5.4: Pasos a realizar durante la etapa de re-entrenamiento de la RNA

5.2. Re-entrenamiento de la RNA

Para actualizar el modelo a fin de mejorar su precisión, se emplea la última generación de control obtenida, cuya calidad ha sido calculada por medio de la función de *fitness* original. Los datos disponibles se dividen en dos conjuntos distintos de entrenamiento y test. El conjunto de entrenamiento se emplea para actualizar la RNA y obtener un nuevo modelo. Con el conjunto de muestras de test se realiza una prueba con el nuevo modelo obtenido y la anterior RNA. Se comparan los resultados de predicción de ambos y se toma como actual modelo aquella que ofrezca mejores resultados (figura 5.4).

Mediante este procedimiento se regula la complejidad del modelo, procurando evitar el sobre-entrenamiento que puede provocar una bajada en la calidad de las predicciones ofrecidas [Jin, 2005]. Como los datos de entrenamiento proceden del proceso evolutivo de optimización, con el paso del tiempo la aptitud o *fitness* de los individuos se va incrementando, a la vez que la diversidad de la población va disminuyendo, puesto que el AG va focalizándose en aquellas regiones del espacio de búsqueda que resulten más prometedoras. Se estudiaron otras formas de realizar esta etapa, seleccionando la presentada como mejor opción [Serra et al., 2007].

5.3. Modelado de la función de fitness

Se emplea el modelo de RNA obtenido en la fase de re-entrenamiento cuando se quiere obtener los valores de *fitness* de cada individuo, así como obtener la calidad media de las generaciones propuestas sin necesidad de emplear la función de aptitud original. En concreto, la aproximación de la función de aptitud se requiere tanto durante la etapa de aplicación de los operadores del AG como durante la etapa de la evaluación simulada. Así pues, durante la actuación del operador de cruce, se obtiene la aproximación de la calidad de los individuos a cruzar. Del mismo modo, para aplicar el factor de reducción durante la etapa de evaluación simulada también se calcula la aptitud aproximada de todos los candidatos a formar parte de la generación de control.

5.4. Aplicación de los operadores del algoritmo genético

Los individuos de la siguiente generación son diseñados por medio de la aplicación de los operadores del algoritmo genético sobre la generación actual, teniendo en cuenta los valores de *fitness* obtenidos por sus individuos. En concreto, el algoritmo genético diseñado utiliza dos operadores: mutación y cruce. Seguidamente se detallará la forma en la que ambos operadores actúan a fin de obtener una nueva generación de soluciones posibles.

Mutación

El operador de mutación modifica los genes o elementos de los individuos con nuevos valores obtenidos de forma aleatoria dentro del rango permitido para éstos (sección 2.1.3). El operador actúa sobre un mismo cromosoma tantas veces como se le indique a través del parámetro *"Número de genes a mutar"*, realizando su labor de dos formas distintas, siendo el azar quien determina de qué manera lo hace en cada ocasión:

a) Modificando únicamente el valor de un elemento. El operador determina de forma aleatoria qué elemento o gen debe ser mutado de entre aquellos que forman parte del cromosoma. Una vez seleccionado, se modifica su valor asignándole un nuevo valor aleatorio, respetando el dominio indicado para ese elemento. En la figura 5.6 puede observarse un ejemplo de aplicación de esta variante del operador de mutación.

b) Modificando la selección actual dentro de los diferentes niveles de agrupación existentes en el cromosoma. En un primer paso, el operador selecciona qué elemento o gen debe ser sustituido, eligiéndolo de forma aleatoria de entre los subniveles de agrupación presentes en el cromosoma (subtipos y subsecciones). En un segundo paso, selecciona un nuevo elemento a formar parte del cromosoma, de entre los disponibles del subnivel seleccionado, de forma que se cumplan las restricciones establecidas en los diferentes niveles de agrupación con respecto al número de elementos a seleccionar en cada caso. Finalmente, se le asigna un nuevo valor al gen, respetando el dominio indicado para él. Mediante esta forma de actuar se pretende garantizar que todas las posibilidades puedan llegar a ser exploradas. En la figura 5.5 se muestra un ejemplo de mutación por cambio de selección sobre un cromosoma que representa posibles soluciones para un problema de optimización de la composición de catalizadores basados en oro.

Por tanto, el operador de mutación tiene un carácter eminentemente explorador, que propone nuevas soluciones y evita que el sistema converja rápidamente hacia máximos locales y se produzca la pérdida de diversidad genética. Las muestras modificadas por el operador de mutación no son modificadas posteriormente por el operador de cruce.

Cruce

El operador de cruce propuesto por [Ortiz et al., 2001] ha sido adaptado a nuestros intereses, teniendo en cuenta las restricciones definidas en la codificación desarrollada. Este operador está basado en intervalos de confianza construidos a partir de los mejores individuos o cromosomas de la generación, denominados progenitores. Se considera que un individuo es mejor que otro si obtiene mejor puntuación al aplicarle la función de aptitud o *fitness* establecida para el problema a resolver, ofreciendo por tanto una solución de mayor calidad.

El carácter del operador de cruce utilizado es tanto explorador como explotador. Así, su capacidad de interpolación o explotación viene determinada por el hecho de que el individuo a ser cruzado pertenezca al intervalo de confianza construido a partir de los progenitores. Por contra, su capacidad extrapoladora o exploradora viene marcada por el hecho de que el individuo no pertenezca al citado intervalo.

Para construir el intervalo de confianza se calculan tres nuevos individuos[1] formados por

[1]CILL= Confidence Interval Lower Limit; CIUL=Confidence Interval Upper Limit; CIM= Confi-

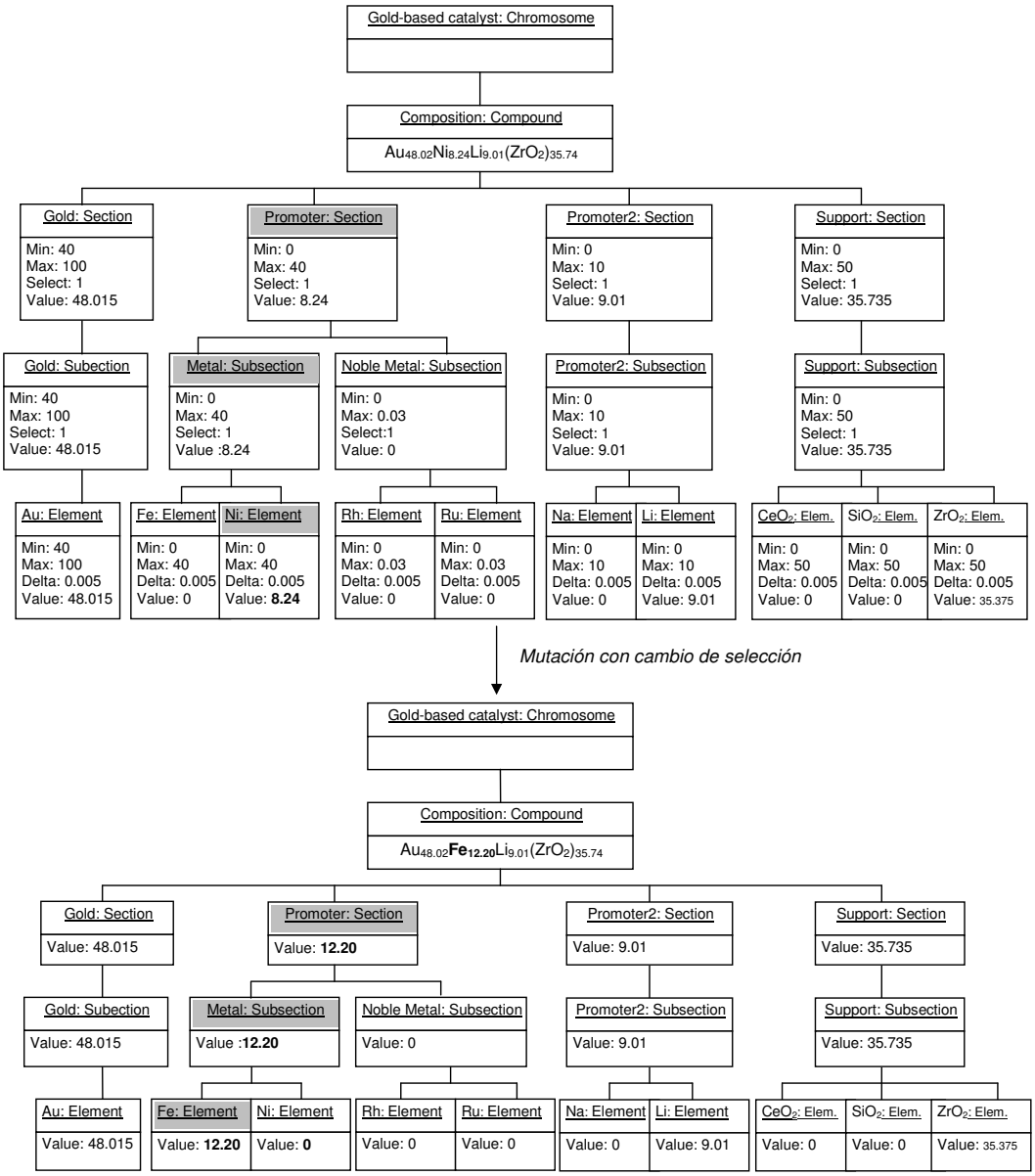

Figura 5.5: Ejemplos de la operación de mutación por cambio de selección

los límites inferiores (CILL), los límites superiores (CIUL) y los valores medios (CIM) de los intervalos de confianza para cada elemento o gen de los progenitores seleccionados, considerando que estos genes son variables aleatorias independientes cuyos valores se distribuyen siguiendo una distribución normal $N(\mu, \sigma_i^2)$). La ecuación 5.2 nos indica su definición para cada elemento o gen, siendo k el número de progenitores, $\bar{\beta}_i$ la media de cada gen, $\bar{S}_{\beta_i}$ la cuasidesviación típica, $t_{k-1}(\alpha/2)$ el valor de la τ de *Student* con $k-1$ grados de libertad y $1-\alpha$ el coeficiente de confianza del intervalo.

$$CILL_i = \bar{\beta}_i - t_{k-1}(\alpha/2)\frac{\bar{S}_{\beta_i}}{\sqrt{k}};\ CIUL_i = \bar{\beta}_i + t_{k-1}(\alpha/2)\frac{\bar{S}_{\beta_i}}{\sqrt{k}};\ CIM_i = \bar{\beta}_i; \tag{5.2}$$

Por tanto, los individuos CILL y CIUL dividen el dominio de cada elemento en tres subintervalos I_1, I_2 y I_3, con Min_i y Max_i como los límites inferiores y superiores del dominio D_i.

$$\begin{aligned} D_i &\equiv I_1 \bigcup I_2 \bigcup I_3; \\ I_1 \equiv [Min_i, CILL];\ I_2 &\equiv]CILL, CIUL[;\ I_3 \equiv [CULL, Max_i] \end{aligned} \tag{5.3}$$

El intervalo de confianza I_2 (el intervalo de explotación) se construye a partir de los mejores individuos de la población bajo la hipótesis que éstos están distribuidos siguiendo una distribución τ de *Student* y que existe una probabilidad $1-\alpha$ de que sus elementos pertenezcan a ese intervalo. Por tanto, α es la probabilidad de que el valor tomado por los elementos (genes) pertenezcan a los intervalos I_1 ó I_3 (intervalos de exploración). Así, un α de 0.5 representa que están balanceados el carácter explorador y explotador del operador, mientras que valores superiores representan un incremento de la función exploradora y viceversa.

Cuando un individuo es cruzado, los elementos de un nuevo individuo se obtienen a partir de los originales, siguiendo las reglas de cruce establecidas por [Ortiz et al., 2001], teniendo en cuenta el intervalo de confianza al que pertenece, el valor de *fitness* aproximado de los tres individuos calculados para obtener el intervalo de confianza y la aptitud del individuo a cruzar (aproximada para las generaciones internas y concreta para las generaciones de control), así como las restricciones establecidas a través de la codificación del cromosoma. Por tanto, dado un individuo a ser cruzado β^f se obtiene

dence Interval Mean

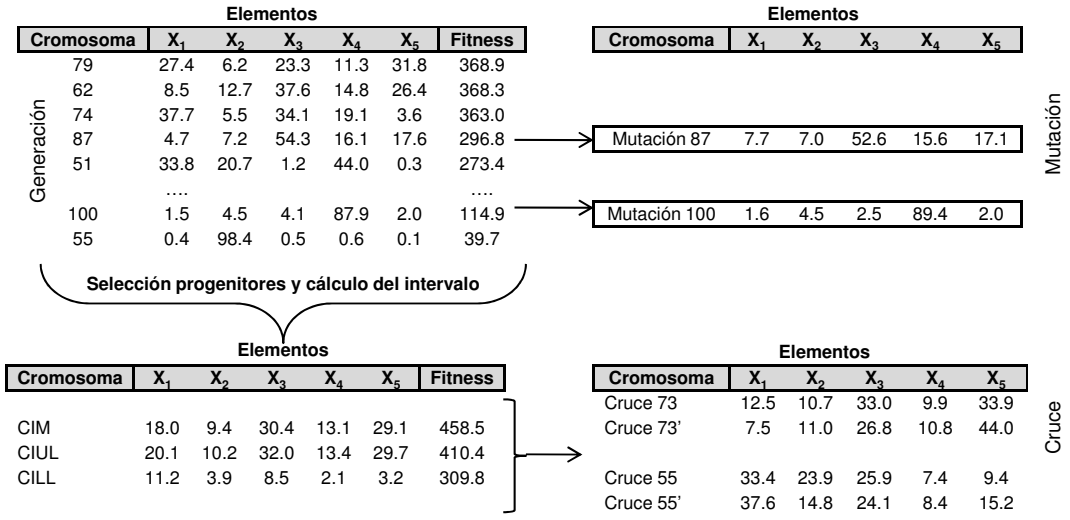

Figura 5.6: Ejemplos de las operaciones de mutación y cruce

un único hijo β^s atendiendo a las siguientes reglas, donde r es un número aleatorio entre $[0, 1]$:

- Si $\beta_i^f \in I_1$ entonces, si la aptitud de β^f es mayor que la de CILL entonces $\beta_i^s = r(\beta_i^f - CILL_i) + \beta_i^f$, sino $\beta_i^s = r(CILL_i - \beta_i^f) + CILL_i$.
- Si $\beta_i^f \in I_2$ entonces, si la aptitud de β^f es mayor que la de CIM entonces $\beta_i^s = r(\beta_i^f - CIMi) + \beta_i^f$, sino $\beta_i^s = r(CIM_i - \beta_i^f) + CIM_i$.
- Si $\beta_i^f \in I_3$ entonces, si la aptitud de β^f es mayor que la de CIUL entonces $\beta_i^s = r(\beta_i^f - CIUL_i) + \beta_i^f$, sino $\beta_i^s = r(CIUL_i - \beta_i^f) + CIUL_i$.

Así, los genes o elementos del hijo generados toman valores próximos al progenitor más prometedor en cada caso, sufriendo grandes cambios si β^f está muy alejado de los mejores individuos de la generación. Por otra parte, a fin de cumplir con las restricciones indicadas en la codificación del problema, el valor final tomado por cada elemento o gen siempre debe cumplir con el dominio que se le haya marcado. Del mismo modo, a fin de cumplir con las restricciones acerca del número de elementos y subsecciones que pueden formar parte de cada cromosoma, cuando un individuo es cruzado, sólo se tienen en cuenta aquellos elementos o genes que están seleccionados, dando lugar a hijos con los mismos genes seleccionados, aunque con valores diferentes para los mismos. La figura 5.6 muestra un ejemplo ilustrativo de la acción de este operador de cruce.

5.5. Evaluación de simulación

Por medio de esta etapa se permite que el algoritmo genético trabaje con poblaciones mayores que aquellas que son finalmente evaluadas por la función de aptitud original. De esta forma, el AG puede trabajar con poblaciones numerosas, favoreciendo su convergencia, sin que ello implique un incremento en el esfuerzo experimental y económico que suponga aplicar la citada función de aptitud sobre los individuos.

La etapa de simulación se lleva a cabo en dos pasos. En el primero, se obtiene el valor aproximado de la aptitud de todos los individuos de la generación (candidatos) por medio del modelo de la RNA disponible. En el segundo paso, se obtiene una generación de control aplicando el ratio o porcentaje de reducción establecido sobre los candidatos, empleando un sistema de selección en el que se tiene mayor probabilidad de ser elegido cuanto mayor sea la aptitud del individuo. En concreto se aplica el método *Roulette Wheel Selection*, comentado en la sección 2.1.2.

5.6. Evaluación de control

En problemas complejos de grandes dimensiones en los que se dispone de poca información, es muy difícil obtener una función de aproximación global perfecta de la función de aptitud o *fitness* original. Por esta razón, el modelo aproximado (un perceptrón multicapa en nuestro caso) debe ser empleado de forma conjunta con la función original. En esta etapa, la generación de control obtenida en la etapa anterior se evalúa mediante la función original, obteniendo los valores de *fitness* de control que serán empleados tanto para evaluar si se ha alcanzado el criterio de convergencia requerido como para continuar con la siguiente iteración del sistema de optimización en caso de no ser alcanzado dicho criterio.

Por ejemplo, para un problema de optimización de la composición de un catalizador, en esta etapa se sintetizarían y probarían en el reactor las diferentes composiciones propuestas por los individuos de la generación de control, a fin de obtener sus resultados catalíticos y calcular el valor de la función objetivo de cada individuo de control.

5.7. Conclusiones

En este capítulo se ha propuesto una arquitectura de búsqueda basada en técnicas *Soft Computing* que permite ser aplicada a problemas combinatorios de diferente índole. En concreto, esta arquitectura combina AG y RNA definiendo un total de seis etapas (figura 5.1): (i) configuración, (ii) re-entrenamiento de la RNA, (iii) modelado de la función de aptitud o *fitness*, (iv) aplicación de los operadores del AG, (v) evaluación simulada y (vi) evaluación de control . La arquitectura ofrece un sistema de codificación de los problemas a tratar que le permite abordar cualquier tipo de problema combinatorio. Además, permite la adecuada configuración de sus etapas para adaptarse a las particularidades concretas del problema a tratar, a fin de ofrecer en todo momento el rendimiento adecuado.

Capítulo

6

Evaluación y Resultados

La arquitectura de búsqueda basada en técnicas *Soft Computing* planteada será aplicada a dos dominios completamente diferentes, con el fin de evaluar la viabilidad de la misma, así como su independencia del dominio al que se aplique. En concreto, se han elegido dos problemas combinatorios muy diferentes: la búsqueda de nuevos catalizadores y condiciones de reacción, dentro del campo de la Catálisis Combinatoria; y la búsqueda de las características de los productos o servicios más deseadas o valoradas por los usuarios, dentro del ámbito de la obtención de preferencias de los usuarios en los Sistemas de Recomendación.

El dominio de la Catálisis Combinatoria será empleado en primer lugar para determinar si cada una de las etapas planteadas en la arquitectura son factibles y si el orden establecido para ellas es correcto. Se ha elegido este dominio debido a la gran cantidad de información disponible y a la necesidad de resolver ciertos problemas de carácter industrial. En concreto se comenzará por validar tanto si los perceptrones multicapa son adecuados como modelos aproximados de la función de aptitud requerida por el algoritmo genético, como la forma planteada para obtener y ajustar estos modelos (etapas I,II y III, figura 5.1). Posteriormente se estudiará la convergencia del algoritmo genético propuesto, considerando el caso en el que se empleen funciones de aptitud aproximadas (etapas III y IV figura 5.1). Finalmente, se estudiarán todas las etapas propuestas de forma conjunta, a fin de determinar las prestaciones que ofrecen, empleando para ello

diferentes problemas dentro de este dominio.

Por otra parte, el dominio de los Sistemas de Recomendación se usará para validar que la técnica propuesta puede emplearse en diferentes tipos de dominios. Así pues, la arquitectura se utilizará para determinar las preferencias de los usuarios cuando no se dispone de un gran volumen de información. Para ello se empleará información procedente de *benchmarks* bien conocidos dentro de este ámbito.

6.1. Aplicación a la Catálisis Combinatoria

Mediante la aplicación al dominio de la Catálisis Combinatoria se tratará de evaluar cada una de las etapas definidas en la arquitectura de búsqueda propuesta en el capítulo 5. Así, la arquitectura de búsqueda será aplicada tanto a la búsqueda de nuevos catalizadores como a la determinación de las condiciones óptimas de reacción para un catalizador específico, a fin de que su rendimiento sea el óptimo.

En primer lugar se determinará si es posible emplear redes neuronales y, en concreto, perceptrones multicapa como modelos de aproximación de los resultados catalíticos de reacciones químicas de interés industrial, disponiendo de un número limitado de muestras ofrecidas a lo largo de la optimización realizada por un algoritmo genético. Seguidamente se pretenderá probar la hipótesis de que un perceptrón entrenado para ofrecer predicciones para una determinada reacción, es capaz de ofrecer predicciones para otra reacción de comportamiento similar, empleando para ello un número reducido de muestras y ciclos de re-entrenamiento. De este modo, sería posible establecer una biblioteca de modelos reutilizables en búsquedas posteriores sobre materiales nuevos, acelerando enormemente el proceso de obtención de modelos aproximados para estas reacciones. A continuación, se evaluará el rendimiento ofrecido por el algoritmo genético combinado con las redes neuronales, así como el resto de etapas propuestas en la arquitectura. Finalmente, la arquitectura propuesta será aplicada a diferentes problemas de interés industrial a fin de probar su utilidad en el ámbito de la catálisis combinatoria.

6.1.1. Redes Neuronales modelando resultados catalíticos

En este primer estudio se evaluará la capacidad de las redes neuronales y, en concreto, de los perceptrones multicapa como modelos de aproximación de los resultados catalíticos de reacciones químicas de interés industrial, cuando la información disponible es

limitada y proviene del proceso de optimización seguido por un algoritmo genético. Por tanto, se realizará una primera evaluación de parte de los pasos planteados en las etapas I, II y III de la arquitectura propuesta (capítulo 5). Así pues, se probará si las redes neuronales son adecuadas como modelos aproximados de la función de aptitud requerida por el algoritmo genético diseñado, así como del procedimiento planteado en la arquitectura para obtener y ajustar estos modelos.

Para problemas de modelado similares al planteado se han empleado los distintos tipos de redes neuronales existentes, como los mapas auto-organizativos, las funciones bases radiales o los perceptrones multicapa. En la arquitectura de búsqueda objeto de estudio se propone emplear perceptrones multicapa como modelos de aproximación por distintos aspectos, como su sencillez, su capacidad de ofrecer respuestas de forma rápida, su habilidad para ofrecer buenas aproximaciones aunque se disponga de poca información de partida y el dominio del problema tenga múltiples dimensiones, así como la falta de necesidad de aportar algún tipo de conocimiento previo acerca de la posible correlación existente entre las variables involucradas en el sistema. Asimismo, dentro de la catálisis combinatoria, los perceptrones multicapa han sido utilizados previamente con éxito asistiendo en tareas de diseño [Huang et al., 2001][Sasaki et al., 1995].

Para este estudio se emplearán datos experimentales de un estudio previo [Buyevskaya et al., 2001][Wolf et al., 2000][Wolf et al., 2002], realizado en el marco del proyecto europeo COMBICAT. Este estudio se realizó sobre la reacción de deshidrogenación oxidativa del ethano (conocida por sus siglas anglosajonas *ODHE*), donde se aplicó un sencillo algoritmo genético para descubrir nuevos catalizadores a partir de trece elementos diferentes (Ga, Cu, P, Mn, Mo, W, Sn, Cr, Co, Zr, Ca, La, Au). El AG partía de una generación inicial de 64 elementos, que contenían combinaciones aleatorias de tres a cuatro elementos. Las siguientes generaciones se obtuvieron aplicando los operadores del AG atendiendo a la función de aptitud o *fitness* calculada a partir de los resultados catalíticos obtenidos empíricamente de las generaciones anteriores. En concreto, se tenían en consideración seis resultados catalíticos: la conversión de etano (C_2H_6) y oxígeno (O_2), el rendimiento de etileno (C_2H_4) y la selectividad de óxido de carbono (CO), dióxido de carbono (CO_2) y etileno (C_2H_4). Para más detalles sobre la estrategia evolutiva empleada y los resultados obtenidos puede consultarse [Wolf et al., 2000][Wolf et al., 2002].

Por tanto, para poder validar la utilidad de emplear redes neuronales para aproximar la función de aptitud o *fitness* necesaria para aplicar un AG, se obtendrá un modelo que

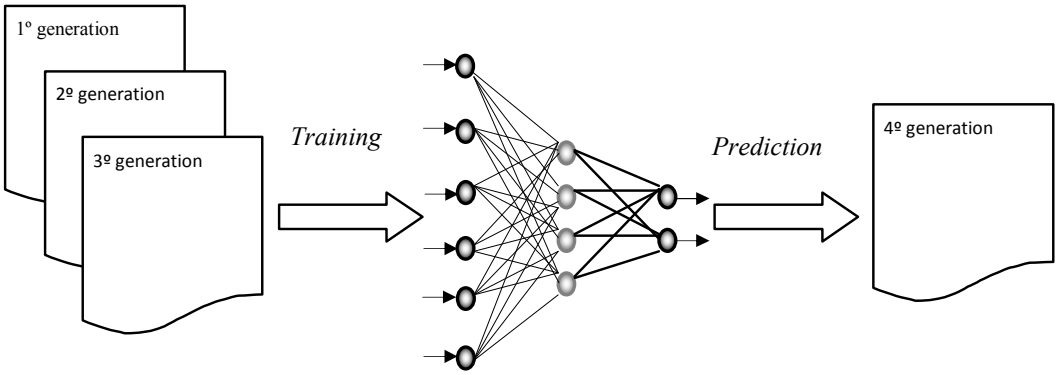

Figura 6.1: RNA entrenada con muestras de generaciones previas empleada para predecir los resultados catalíticos de la generación actual

sea capaz de obtener predicciones para los seis resultados catalíticos citados, teniendo como datos de entrada los porcentajes molares de cada uno de los trece elementos posibles en la fase activa de cada catalizador. Para realizar este estudio se tendrán en cuenta un número limitado de muestras, correspondientes a la primera generación del genético, a fin de simular las condiciones reales en las que ambas técnicas deban trabajar en combinación.

Una vez se obtenga un perceptrón adecuado y entrenado con la totalidad de muestras de la primera generación, éste se empleará para predecir los resultados de la siguiente generación. Después, la red será re-entrenada con los datos de la primera y segunda generación, utilizándose a continuación para predecir los resultados catalíticos de la tercera generación y así sucesivamente (Figura 6.1).

A fin de medir la precisión de las predicciones ofrecidas por los perceptrones se empleará como métrica el error cuadrático medio y el error absoluto obtenidos, tanto en el proceso a seguir para seleccionar el perceptrón, algoritmo de entrenamiento, etc., más adecuados, como para la simulación de su actuación combinada con el genético a realizar posteriormente.

Metodología

A fin de obtener el modelo requerido, se seguirá el proceso descrito en la sección 5.1.4 de la propuesta. Por tanto, al no disponer de una regla general que nos indique qué combinación de topología de red y algoritmo de entrenamiento es el más indicado para cada problema, el proceso a seguir para la construcción del perceptrón multi-capa con-

siste en probar con diferentes topologías de perceptrón multicapa, partiendo de una sola capa oculta con unas pocas neuronas e ir incrementando su número, así como el número de capas internas. Además, es interesante probar cada una de ellas con diferentes algoritmos de aprendizaje. En concreto, se ha estudiado el comportamiento de las RNA con los algoritmos de aprendizaje de retro-propagación del error (*Backpropagation*) y retro-propagación del error con momento (*Backpropagation with momentum*) con diferentes valores para sus parámetros (factor de aprendizaje α y momento μ). Se han seleccionado estos algoritmos de aprendizaje puesto que la literatura indica que se comportan bien para la mayoría de problemas al entrenar a perceptrones multicapa [Ripley, 1996][Duda et al., 2001]. Todas las pruebas se han llevado a cabo empleando el simulador de redes neuronales SNNS, desarrollado por el Instituto de Sistemas de Alto Rendimiento Paralelos y Distribuidos de la Universidad de Stuttgart [Zell et al., 1995].

Para realizar este estudio, es necesario partir de un conjunto de muestras sobre las que realizar el entrenamiento y posterior test, a fin de seleccionar como modelo a aquella red que ofrezca mejores resultados. En el problema planteado como caso de estudio se dispone de los resultados catalíticos de un total de 329 catalizadores actuando sobre la reacción ODHE. Como datos de entrada se dispone de los porcentajes molares para cada elemento perteneciente a la fase activa de cada catalizador, mientras que como datos de salida se tienen las siguientes conversiones X [%], rendimientos Y [%] y selectividades S [%]: $X(Etano), X(O_2), S(CO_2), S(CO)$ e $Y(Etano)$.

Como punto de inicio para el estudio planteado, a fin de simular las condiciones de partida en las que se trabajaría en un proceso de búsqueda llevado a cabo con la arquitectura propuesta, se ha tomado un subconjunto de las muestras disponibles, formado por 50 catalizadores, que representaría a los individuos de la generación aleatoria inicial. Siguiendo la metodología descrita en la propuesta (sección 5.1.4), se ha aplicado la validación cruzada, dividiendo el conjunto de entrenamiento en diferentes subconjuntos de 40 muestras de entrenamiento (80 %) y 10 muestras de test (20 %). Por tanto, cada experimento se ha llevado a cabo con diferentes combinaciones de conjuntos de entrenamiento y test, teniendo en consideración los resultados medios obtenidos en las diferentes pruebas.

Resultados

En las tablas mostradas en las figuras 6.2 y 6.3 se ofrece un resumen de las pruebas realizadas sobre las diferentes topologías y algoritmos de entrenamiento. En concre-

Training data (40 samples)

					Learning process: Backpropagation						Backpropagation Momentum					
				α:	0.2	0.5	0.8	0.2	0.2	0.2	0.5	0.5	0.5	0.8	0.8	0.8
				μ:				0.2	0.5	0.8	0.2	0.5	0.8	0.2	0.5	0.8
Neural network topology																
input layer	1st hidden layer	2nd hidden layer	output layer	weights												
					Mean square error											
13	8		6	152	98.43	59.20	48.54	88.30	65.65	48.25	54.16	43.80	27.85	44.99	34.43	26.80
13	20		6	380	104.81	64.92	44.23	99.51	71.66	34.89	54.54	33.82	11.20	35.63	21.87	8.09
13	40		6	760	101.14	63.27	38.07	96.06	70.24	26.86	50.29	27.08	11.40	27.62	14.25	9.01
13	8	6	6	188	227.88	125.87	115.83	144.71	131.26	123.74	122.10	117.97	117.04	114.58	112.20	113.03
13	26	12	6	722	87.56	38.87	22.00	78.13	51.64	15.64	28.60	15.74	12.13	15.32	10.72	8.10
13	40	20	6	1440	74.59	33.01	22.52	62.55	39.90	14.52	25.31	16.29	7.94	16.12	8.81	8.64
					Mean absolute error											
13	8		6	152	7.14	5.48	4.88	6.82	5.90	4.79	5.17	4.58	3.57	4.64	4.08	3.43
13	20		6	380	7.23	5.80	4.64	7.02	6.16	4.07	5.25	4.00	2.45	4.19	3.30	2.03
13	40		6	760	7.03	5.78	4.42	6.82	6.03	3.78	5.12	3.79	2.49	3.89	2.79	2.16
13	8	6	6	188	10.43	7.89	7.68	8.33	7.98	7.85	7.82	7.70	7.71	7.65	7.60	7.66
13	26	12	6	722	6.76	4.43	3.34	6.29	5.07	2.87	3.75	2.90	2.59	2.83	2.35	2.08
13	40	20	6	1440	6.24	4.18	3.69	5.61	4.47	2.90	3.80	3.05	1.99	3.03	2.07	2.04

Figura 6.2: Resultados del proceso de entrenamiento

Testing data (10 samples)

					Learning process: Backpropagation						Backpropagation Momentum					
				α:	0.2	0.5	0.8	0.2	0.2	0.2	0.5	0.5	0.5	0.8	0.8	0.8
				μ:				0.2	0.5	0.8	0.2	0.5	0.8	0.2	0.5	0.8
Neural network topology																
input layer	1st hidden layer	2nd hidden layer	output layer	weights												
					Mean square error											
13	8		6	152	177.2	252.9	224.0	184.0	224.7	237.1	255.4	216.1	218.7	216.1	241.3	180.2
13	20		6	380	178.9	208.1	225.2	183.7	188.3	249.6	226.8	233.5	297.2	230.8	260.8	310.3
13	40		6	760	188.0	219.1	260.1	193.8	202.7	285.3	237.6	286.0	302.1	287.5	301.3	313.2
13	8	6	6	188	238.9	189.2	198.8	163.3	185.3	216.3	198.9	211.6	213.0	205.8	209.2	213.9
13	26	12	6	722	154.4	211.2	334.6	153.4	172.5	328.2	304.2	321.7	387.9	316.3	330.1	401.0
13	40	20	6	1440	184.8	235.7	300.2	213.4	234.4	361.8	281.8	334.6	328.3	344.6	308.7	338.4
					Mean absolute error											
13	8		6	152	8.71	10.53	10.46	9.00	10.02	10.61	10.79	10.38	10.03	10.32	10.61	8.51
13	20		6	380	8.75	9.77	9.90	8.86	9.45	10.37	9.87	10.23	11.75	10.16	11.20	11.82
13	40		6	760	9.05	10.07	10.81	9.22	9.80	11.47	10.32	11.46	11.50	11.39	11.36	12.05
13	8	6	6	188	11.31	9.05	9.60	8.39	8.89	9.63	9.31	9.80	9.76	9.73	9.92	9.87
13	26	12	6	722	8.48	10.08	11.13	8.53	9.48	11.34	11.19	11.05	13.07	10.98	11.91	13.56
13	40	20	6	1440	9.40	10.91	11.13	10.26	11.04	12.09	11.33	11.93	12.22	11.92	11.28	12.73

Figura 6.3: Resultados del proceso de test

to la tabla de la figura 6.2 muestra la media de los errores absolutos y cuadráticos medios obtenidos en el proceso de entrenamiento (sobre 40 muestras) por las diferentes RNA empleando retro-propagación del error (*backpropagation*) con y sin momento (*momentum*). De forma similar, la tabla mostrada en la figura 6.3 ofrece los errores cuadráticos medios y absolutos obtenidos por esas RNA en el proceso de test, es decir, al ser empleadas para predecir los resultados catalíticos de las 10 muestras restantes.

A la vista de los resultados obtenidos, las RNA que se comportan mejor son aquellas que poseen un mayor número de neuronas, ofreciendo errores menores tanto en el proceso de entrenamiento como en las fases de test. Además, para este caso, se obtienen mejores resultados en la fase de test con las RNA entrenadas con retro-propagación del error sin momento. Sin embargo, las diferencias entre los diferentes resultados no son demasiado significantes, por lo que la selección de la mejor combinación de topología de RNA y algoritmo de entrenamiento se ha realizado atendiendo además a cómo las predicciones

se asemejan a las tendencias seguidas por las muestras experimentales.

Atendiendo a los criterios anteriormente mencionados, se ha seleccionado la red compuesta por 13 neuronas en su capa de entrada (porcentaje molar de los trece elementos posibles en la fase activa del catalizador), 26 neuronas en la primera capa oculta, 12 neuronas en la segunda capa oculta y 6 neuronas en la capa de salida (correspondientes a las predicciones sobre las conversiones, rendimientos y selectividades citados anteriormente). En la figura 6.4 pueden observarse los resultados de predicción ofrecidos por la red seleccionada para los 10 catalizadores empleados en la fase de test. En concreto se muestran los valores experimentales de rendimiento de etileno y conversión del oxígeno obtenidos para cada catalizador, los resultados predichos por la red para esos valores y el área de predicción establecida para la red.

El área de predicción establecida determina la región en la que la RNA tiene mayor probabilidad de ofrecer buenas predicciones de los resultados catalíticos de los diferentes catalizadores. El límite inferior (LI) y el límite superior (LS) del intervalo han sido calculados a partir de la media del error absoluto cometido por la red durante su fase de entrenamiento (ϵ) y la predicción realizada para cada muestra i, tal y como puede apreciarse en la siguiente ecuación.

$$LI(i) = X(i) - \epsilon;\ LS(i) = X(i) + \epsilon \tag{6.1}$$

Tal y como ha podido observarse, el modelo establecido por RNA seleccionada es capaz de ajustarse a las pautas experimentales, a pesar del reducido número de muestras empleadas para entrenarla con respecto a la amplitud del espacio combinatorio de posibles catalizadores existente. Por tanto, es posible emplear un perceptrón multicapa para modelar resultados catalíticos, a fin de utilizarlo como aproximación a una función de aptitud a emplear posteriormente por un AG. A pesar de no disponer de gran información de partida y aunque el modelo no sea capaz de ofrecer resultados precisos, éste sí sigue las tendencias de forma adecuada. De este modo, para un excelente catalizador, la RNA predecirá buenos resultados catalíticos y para un pésimo catalizador, indicará que sus resultados no serán buenos. Así pues, un AG que disponga de este tipo de información podrá realizar adecuadamente su labor.

Por último, a fin de probar la idoneidad de los pasos establecidos en el paso II de la arquitectura propuesta (sección 5.2), donde se pretende ajustar el modelo obtenido a medida de que se disponga de más muestras de control (muestras evaluadas con la

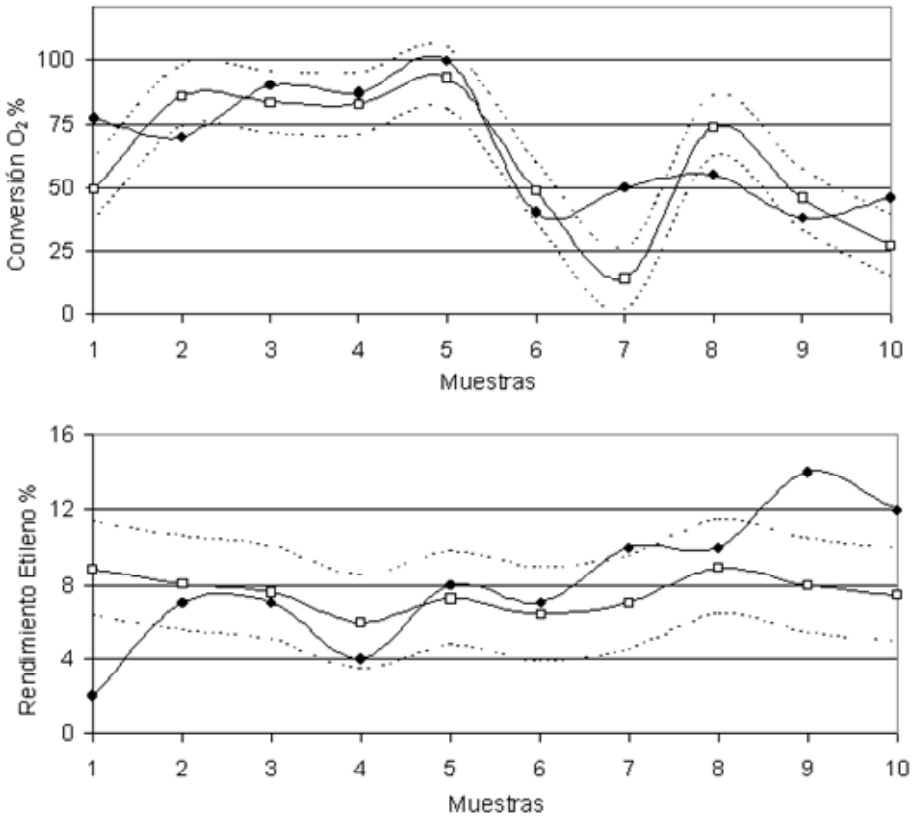

Figura 6.4: Resultados experimentales (•) y los ofrecidos por la red neuronal seleccionada (□) para el rendimiento del etano y la conversión del O_2. Las lineas punteadas muestran la región de predicción establecida.

	Error Absoluto						
Predicciones	$X(Etano)$	$X(O_2)$	$S(Etileno)$	$S(CO_2)$	$S(CO)$	$Y(Etileno)$	Media
4 Generación	8.54	22.52	16.48	20.85	8.07	2.51	13.16
5 Generación	6.47	21.14	16.30	23.78	12.28	2.46	13.74
6 Generación	3.47	17.78	9.43	14.35	5.44	1.71	8.70
7 Generación	4.58	21.15	14.60	16.45	8.31	2.52	11.27

Cuadro 6.1: Media de los rerrores absolutos cometidos por la RNA al predecir los resultados catalíticos de las diferentes generaciones ofrecidas por el AG

función de aptitud original, consistente en la obtención de resultados experimentales en este caso), la RNA obtenida fue re-entrenada con los datos de las generaciones predecesoras de aquella para la que la RNA ofrece su predicciones. Así en la figura 6.5 pueden apreciarse los resultados experimentales y aproximados por la red para la conversión del oxígeno y del etano para cada generación, cuando el modelo ha sido entrenado con muestras procedentes de las anteriores generaciones. Asimismo, en la tabla 6.1 pueden observarse la media de los errores absolutos cometidos al predecir cada generación.

Puede apreciarse que el modelo ajusta mejor las tendencias experimentales conforme aumenta el número de generaciones empleadas en su entrenamiento. Asimismo, a pesar de que las predicciones realizadas por el modelo no son excesivamente precisas, son suficientemente buenas para realizar un proceso de selección previo a la experimentación real (*pre-screening*), donde se descarten aquellos catalizadores que se prevé no ofrecerán buenos resultados. De forma similar, un AG que emplee este modelo como aproximación de su función objetivo será capaz de determinar adecuadamente que catalizadores deben emplearse como progenitores y cuáles deben descartarse. Finalmente, los valores predichos son comparados entre sí, siendo el error cometido en su predicción similar (tanto en exceso como en defecto), y además el error experimental cometido posteriormente en numerosas ocasiones puede llegar a ser parecido.

Por otra parte, aunque el modelo obtenido mediante un perceptrón multicapa difícilmente puede llegar a ofrecer buenas predicciones sobre las singularidades (al no seguir la tendencia general), sí es capaz de indicar qué zonas del espacio de búsqueda pueden ofrecer actividades prometedoras.

Finalmente, a la vista de los resultados obtenidos, las RNA y en concreto los perceptrones multicapa resultan adecuados para ser empleados de forma combinada con un AG dentro del ámbito de la catálisis combinatoria. Así, el AG se encargaría de buscar qué combinación de elementos y concentraciones obtendrían mejor rendimien-

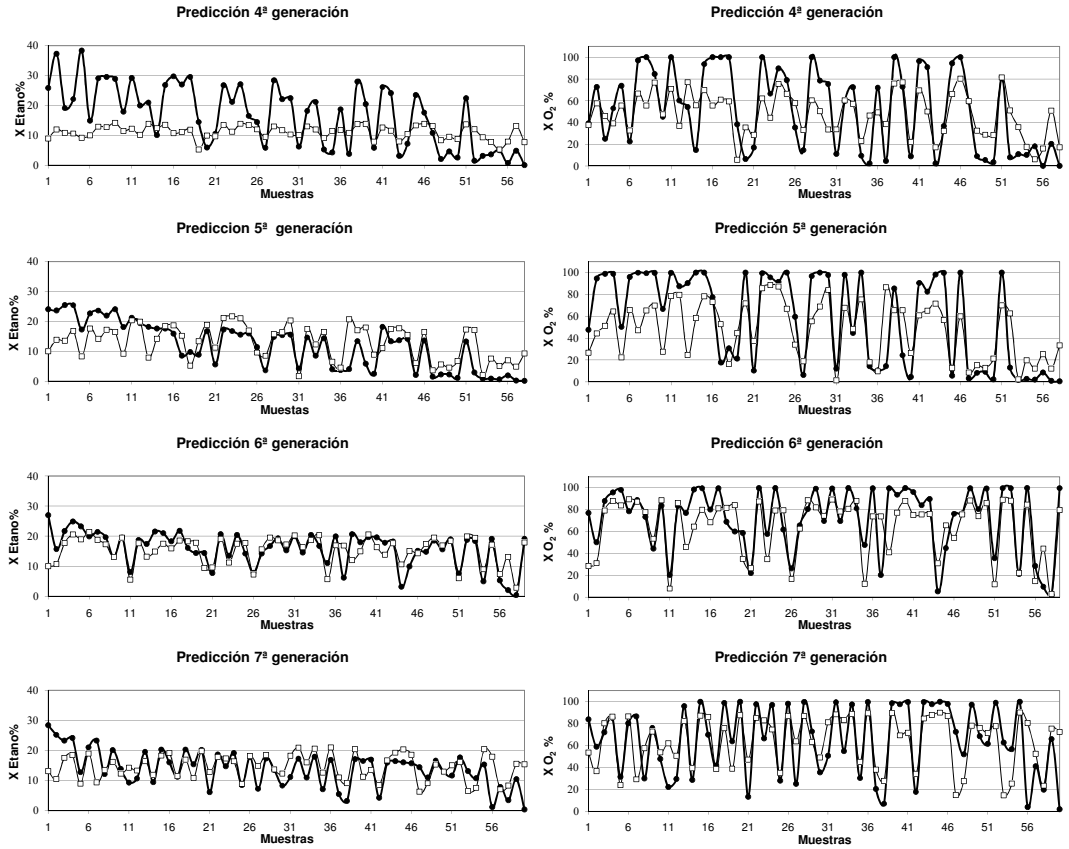

Figura 6.5: Evolución de las predicciones ofrecidas por el modelo para una generación, habiendo sido entrenado con muestras procedentes de las generaciones anteriores. Resultados experimentales (•) y aproximados (□) por el modelo para la conversión del etano y del O_2.

to catalítico. A lo largo de su proceso de búsqueda, el AG se concentraría en ciertas regiones prometedoras (regiones de explotación) y por tanto el modelo de la RNA se iría ajustando mejor a las tendencias ofrecidas en éstas regiones. Por otra parte, el AG también dispone de operadores encaminados a no perder en demasía la diversidad de las generaciones, a fin de mantener la posibilidad de explorar la mayor cantidad posible de regiones del espacio de búsqueda (regiones de exploración). En este caso, el modelo de la RNA ofrecerá predicciones menos precisas para las muestras pertenecientes a éstas regiones, si bien será capaz de indicar su idoneidad gracias al conocimiento adquirido a partir de las generaciones previas (de menor a mayor grado de focalización).

6.1.2. Obtención de bibliotecas de modelos

La hipótesis que se pretende probar consiste en considerar que los perceptrones multicapa son capaces de modelar el comportamiento catalítico y además, que un perceptrón entrenado para ofrecer predicciones para una determinada reacción es capaz de ofrecer predicciones para otra reacción de comportamiento similar, empleando para ello un número reducido de muestras y ciclos de re-entrenamiento. De este modo, sería posible establecer una biblioteca de modelos reutilizables en búsquedas posteriores sobre materiales nuevos, acelerando enormemente el proceso de obtención de modelos aproximados para estas reacciones.

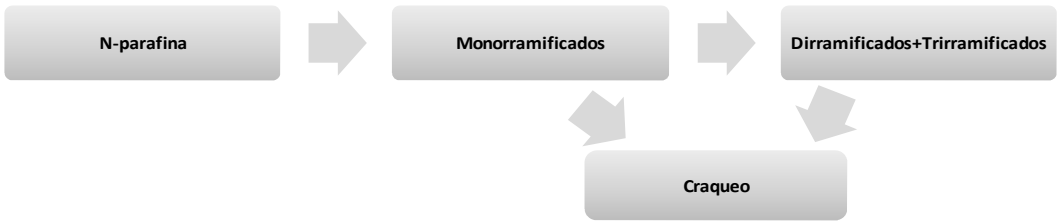

Figura 6.6: Esquema de isomerización seguido por las parafinas n-C_8, n-C_7, n-C_6 y n-C_5

Para este fin se emplearán los resultados experimentales de las reacciones de isomerización de cuatro n-parafinas diferentes n-octano (n-C_8), n-heptano (n-C_7), n-hexano (n-C_6) y n-pentano (n-C_5). Este es un proceso que ha sido ampliamente estudiado por el Instituto de Tecnología Química (ITQ)[1] del CSIC [Chica et al., 2001][Sastre et al., 2000]. Además, estas reacciones siguen un esquema de reacción similar, sobre todo en el caso de $C_7 - C_8$ (figura 6.6).

Además de datos experimentales, están disponibles diversas ecuaciones cinéticas que permiten simular las reacciones. Estas ecuaciones fueron obtenidas previamente [Chica, 2002] mediante experimentación sobre un reactor prototipo, obteniéndose los valores necesarios para el cálculo de los parámetros que describen la cinética del proceso. Por tanto, para el estudio a realizar están disponibles un gran número de muestras, tanto experimentales (300) como simuladas (34000) a través de los modelos cinéticos disponibles.

Las reacciones de isomerización son empleadas para aumentar el octanaje de las gasolinas, a la vez que se elimina la utilización de plomo como aditivo. Mediante el proceso de isomerización se reemplazan las parafinas por sus isómeros ramificados. Estos procesos suelen hacer uso de catalizadores bifuncionales, es decir, catalizadores formados por una

[1] http://itq.webs.upv.es/

función ácida, responsable de la isomerización y craqueo de hidrocarburo, y una función metálica, responsable de los procesos de hidrogenación / deshidrogenación. La reacción de isomerización es un proceso que implica la transformación de una parafina lineal en su análoga ramificada con un mismo número de carbonos. En un primer paso el alcano es isomerizado a un conjunto de sus isómeros con ramificaciones simples. En otra etapa posterior, estos isómeros son de nuevo isomerizados, dando lugar a dirramificados y así consecutivamente se va aumentado el grado de ramificación de los isomeros. Junto a la reacción de isomerización siempre aparecen, en mayor o menor grado, reacciones de rotura de cadena carbonada (denominadas de craqueo), que disminuyen el rendimiento hacia productos de interés. Por tanto, para aumentar el beneficio económico, se persigue disminuir al máximo la presencia de estas reacciones de craqueo, buscando para ello las condiciones más convenientes.

Metodologia

Se pretenden obtener bibliotecas de perceptrones multicapa capaces de modelar las diferentes reacciones de n-parafinas presentadas, a fin de ofrecer predicciones sobre los resultados de la reacción, en términos del factor de conversión de la n-parafina lineal en n-parafina ramificada, así como el rendimiento de monorramificados y dirramificados, para un catalizador determinado. En la tabla 6.2 pueden observarse los valores mínimos, máximos y medios de los resultados catalíticos de las diferentes reacciones para las muestras que se disponen para realizar este estudio (300 muestras por reacción). Asimismo, los parámetros de entrada se corresponden con las cuatro condiciones de reacción: la presión parcial de la n-parafina (de 0.5 a 2 bares), la presión parcial del hidrógeno (6 a 18 bares), la temperatura (de 503 a 543K) y el tiempo de contacto con el catalizador (de 0.16 a 18h).

Así pues, en primer lugar se realizará un estudio sobre qué topología (número de capas ocultas y neuronas por capa) y función de activación (logística o tangencial) son los más apropiados para que los perceptrones multicapa aproximen adecuadamente las reacciones de isomerización de n-parafinas, tomando como punto de partida la reacción n-octano. Como algoritmo de entrenamiento se ha seleccionado el algoritmo de aprendizaje de retro-propagación del error, por ser el que mejores resultados ofrecía en el estudio presentado anteriormente (sección 6.1.1). En la tabla (6.3) se muestra de forma detallada todas las redes y funciones de activación estudiadas. Además, se ha comparará como se comportan las diferentes configuraciones al ser entrenadas con conjuntos

Reacción	Máximo	Mínimo	Media (μ)	Desviación Estándar τ)
Conversión				
n-Octano	100	12.49	91.48	17.91
n-Heptano	100	4.50	86.72	20.79
n-Hexano	100	5.41	83.14	23.56
n-Pentano	74.89	1.97	54.70	19.86
Rendimiento Monorramificados				
n-Octano	53.37	0	11.42	14.01
n-Heptano	50.38	0	16.17	15.50
n-Hexano	83.86	0.12	53.82	25.96
n-Pentano	0.66	0	0.33	0.21
Rendimiento Dirramificados				
n-Octano	68.18	0	41.77	20.71
n-Heptano	71.36	0.09	45.97	20.28
n-Hexano	83.86	0.12	53.82	25.96
n-Pentano	0.66	0	0.33	0.21

Cuadro 6.2: Distribución de los resultados catalíticos para las reacciones n-octano, n-heptano, n-hexano, n-pentano empleados para obtener los modelos basados en perceptrones multicapa.

de muestras de diferente tamaño, a fin de seleccionar aquella combinación que ofrezca buenos resultados no sólo al estar entrenada con numerosas muestras, sino también al emplear conjuntos pequeños de muestras de entrenamiento (puesto que al emplear un AG se tienen pocas muestras en las primeras etapas). El conjunto de muestras de test está formado por 300 muestras.

En segundo lugar, la red que mejores resultados ofrezca en el apartado anterior será entrenada mediante el algoritmo de retro-propagación del error con y sin momento, empleando diferentes parámetros de entrenamiento y funciones de activación (logística y tangencial), a fin de determinar la influencia en los resultados ofrecidos por la topología seleccionada el algoritmo de entrenamiento y la función de activación utilizada.

Una vez se obtenga una topología adecuada para construir modelos válidos para el proceso cinético de las diferentes reacciones, se evaluará la viabilidad de obtener bibliotecas de modelos de forma rápida. en este sentido, se analizará la posibilidad de obtener modelos del comportamiento catalítico de una reacción n-parafina (por ejemplo n-heptano), empleando para ello un perceptrón entrenado para otra reacción n-parafina (por ejemplo n-octano), tras re-entrenarlo con un número limitado de muestras. Además, se estudiará la influencia de emplear redes más o menos informadas (entrenadas con un mayor o menor número de muestras de la reacción original), así como el número de

Entrada	1^a Capa	2^{da} Capa	Salida	Func. Activación	Muestras Entrenamiento
4	6	0	3	Logistica	10
	8	3		Tangencial	20
	10	5			30
	20	6			50
	40	10			80
	100				100
					170
					850
					1700
					4250
					17000

Cuadro 6.3: Topologías, funciones de activación y conjuntos de muestras de entrenamiento empleadas en el estudio realizado para la obtención de una topología de perceptrón adecuada para reacciones de n-parafinas.

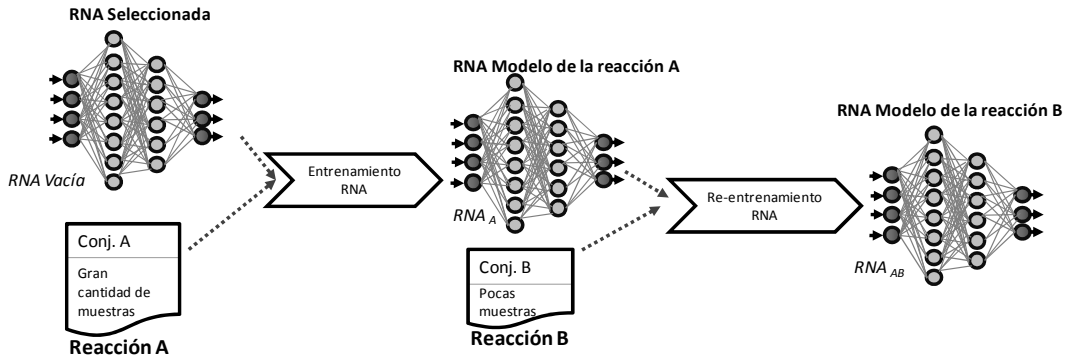

Figura 6.7: Esquema seguido en el proceso de obtención de bibliotecas de modelos mediante re-entrenamiento. En primer lugar se obtiene una RNA que modele una *reacción A*, de la que se posee mucho conocimiento acerca de su proceso cinético y por tanto se dispone de un gran número de muestras. Seguidamente, esa RNA es re-entrenada con unas pocas muestras de la *reacción B*, de la que se dispone de poco conocimiento, obteniendo así un nuevo modelo para la *reacción B*.

muestras de re-entrenamiento mínimo para obtener resultados aceptables para poder emplear la RNA como modelo aproximado de una función de aptitud (10,20,30,50 y 85 muestras).

El esquema a seguir (ver Fig. 6.7) comienza partiendo de una primera *reacción A*, de la que se posee mucha información (*Conj.A*), permitiendo obtener un buen modelo (RNA_A). Después esta RNA_A es re-entrenada con unas pocas muestras (*Conj.B*) de una *reacción B*, de la que no se dispone de mucha información pero se conoce que sigue un esquema cinético similar a A. Así, mediante este proceso de re-entrenamiento se obtiene un nuevo modelo para la reacción B (RNA_{AB}) sin necesidad del coste experimental asociado a obtener un gran número de muestras de la reacción B.

En concreto, se tendrán en consideración cuatro tamaños diferentes para los conjuntos de muestras de entrenamiento para la reacción A: $ConjA_i = \{0, 85, 850, 17000\}$. Así, a partir de estos conjuntos de entrenamiento se obtendrán cuatro modelos para la reacción A, con diferente grado de conocimiento: $RNA_{A(0)}$ ($ConjA = 0$ muestras), $RNA_{A(85)}$ ($ConjA = 85$ muestras), etc.

Con respecto a los conjuntos de muestras empleados para re-entrenar los diferentes modelos RNA_{Ai}, se tendrán en consideración seis tamaños diferentes: $ConjB_j = \{0, 10, 20, 30, 50, 85\}$. De esta manera, se obtendrán un total de 24 modelos diferentes para la reacción B ($NN_{AB(i,j)}$), con i y j indicando la cantidad de muestras empleadas para entrenar y re-entenar respectivamente. Así pues, el número total de modelos basados en RNA obtenidos en este estudio será de 288, teniendo en cuenta los diferentes conjuntos de entrenamiento y re-entrenamiento tenidos en consideración para las cuatro reacciones n-parafinas contempladas.

Los diferentes modelos $RNA_{AB(i,j)}$ serán testados mediante un conjunto de 300 muestras procedentes de la reacción B ($test_B$).

El proceso descrito anteriormente (Fig. 6.7) será aplicado a las cuatro reacciones n-parafinas citadas con antelación (n-C_8, n-C_7, n-C_6 y n-C_5), seleccionando en cada ocasión una de ellas como reacción A y otra de ellas como reacción B.

Para todos los estudios planteados anteriormente, las métricas a emplear para comparar los resultados ofrecidos por los diferentes perceptrones serán tanto el error cuadrático medio como el error absoluto cometido al realizar sus predicciones.

Resultados

En las figuras 6.8 y 6.9 se muestra el error absoluto y cuadrático cometido por aquellas topologías de red que han ofrecido mejores resultados dentro del estudio realizado (tabla 6.3) con diferentes conjuntos de muestras de entrenamiento, empleando como algoritmo de entrenamiento el de retro-progación del error (*backpropagation*). A la vista de los resultados, se ha optado por seleccionar un perceptrón multicapa (figura 6.10) con la siguiente topología (4-8-6-3): cuatro nodos en su capa de entrada (correspondientes a las condiciones de reacción), ocho nodos en su primera capa oculta, seis nodos en su segunda capa oculta y tres nodos en su capa de salida (correspondientes a los resultados esperados para la reacción). En sus nodos ocultos es utilizada una función de activación sigmoidal tangencial. Esta topología ofrece los mejores resultados para

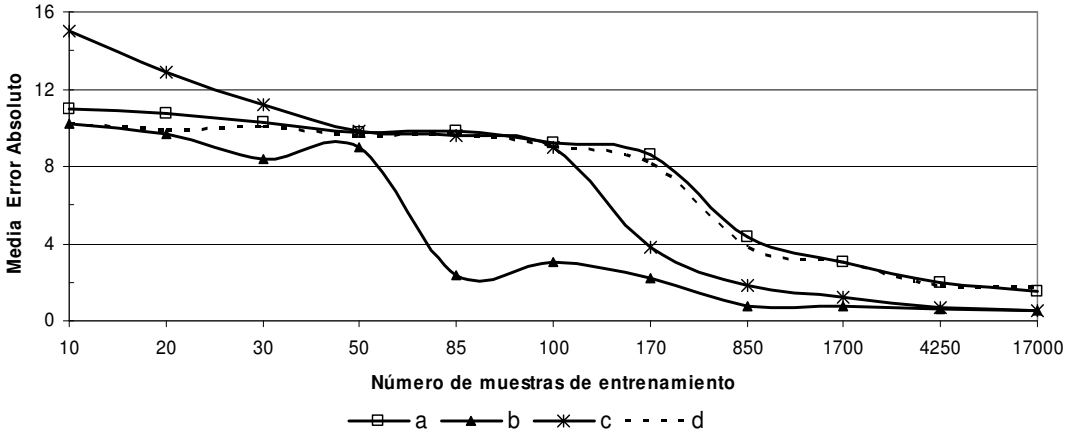

Figura 6.8: Media del error absoluto cometido por las mejores RNA al modelar los resultados catalíticos de conversión para la reacción n-octano de 300 muestras de test: (a) 4-10-3; (b)4-8-6-3 con función sigmoidal tangencial en sus nodos ocultos; (c) 4-8-6-3 con función sigmoidal logística en los nodos ocultos; (d) 4-8-3

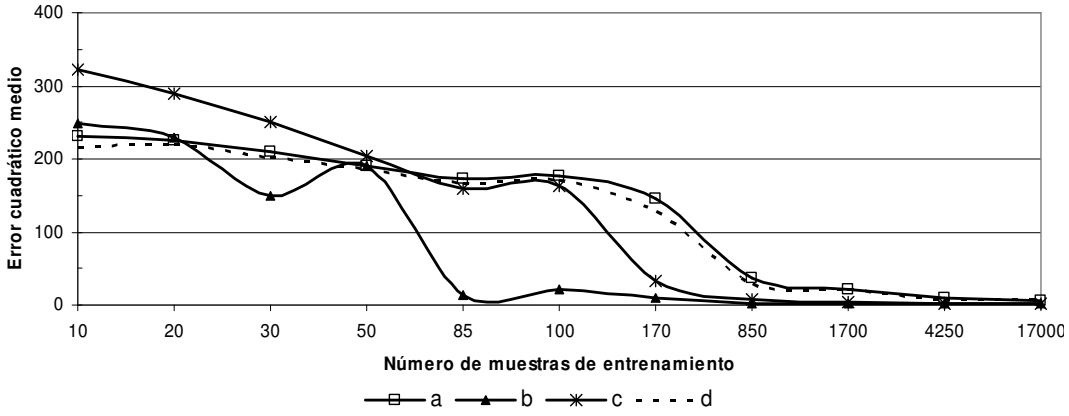

Figura 6.9: Media del error cuadrático medio cometido por las mejores RNA al modelar los resultados catalíticos de conversión para la reacción n-octano de 300 muestras de test: (a) 4-10-3; (b) 4-8-6-3 con función sigmoidal tangencial en sus nodos ocultos; (c) 4-8-6-3 con función sigmoidal logística en los nodos ocultos; (d) 4-8-3

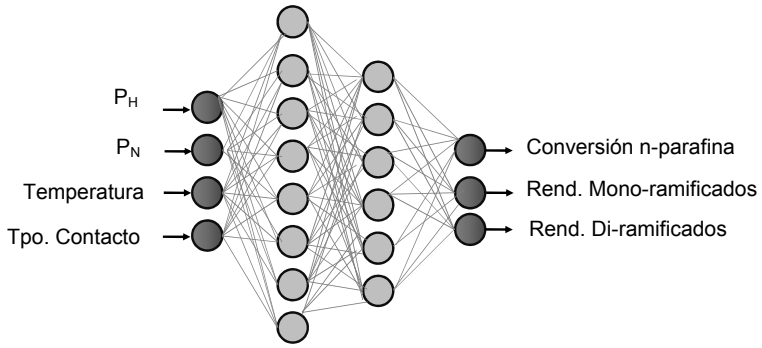

Figura 6.10: Topología de red seleccionada como mejor modelo para las reacciones n-parafinas, con cuatro entrenadas, ocho nodos en su primera capa oculta, seis nodos en su segunda capa oculta y tres en la capa de salida (RNA 4-8-6-3).

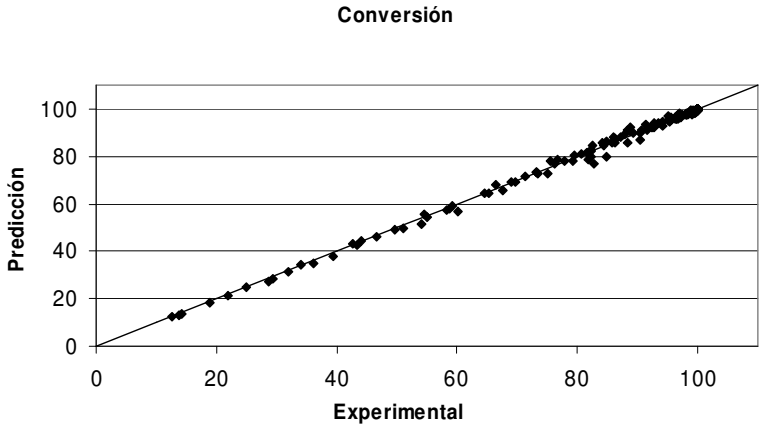

Figura 6.11: Predicciones de conversión obtenidas para la reacción n-octano mediante la RNA 4-8-6-3 empleando la función de activación tangencial en sus nodos ocultos.

todos los conjuntos de muestras de entrenamiento con los que se ha probado, siendo necesarias sólo 85 muestras para ofrecer resultados válidos para el propósito planteado, es decir, para actuar como un modelo aproximado de una función de aptitud.

Así pues, en las figuras 6.11, 6.12 y 6.13 se muestra la comparativa entre los valores experimentales y predichos o estimados por la red 4-8-6-3 (entrenada con 850 muestras) para un conjunto de 300 muestras de test. En concreto se muestra las predicciones para los diferentes resultados de la reacción n-octano que estima: conversión, rendimiento a mono-ramificados y rendimiento a di-ramificados.

Con respecto al estudio del impacto del algoritmo de entrenamiento y funciones de activación empleados, en las figuras 6.14 y 6.15 se muestran los resultados de cinco de las diferentes pruebas realizadas, correspondientes a aquellas que se han comportado mejor.

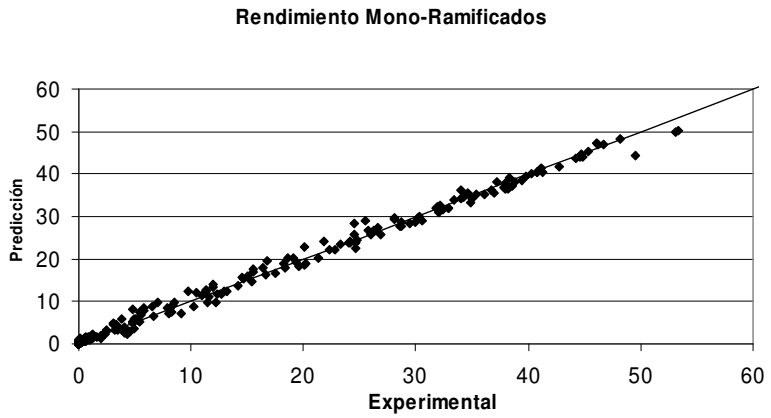

Figura 6.12: Predicciones de rendimiento a mono-ramificados obtenidas para la reacción n-octano mediante la RNA 4-8-6-3 empleando la función de activación tangencial en sus nodos ocultos.

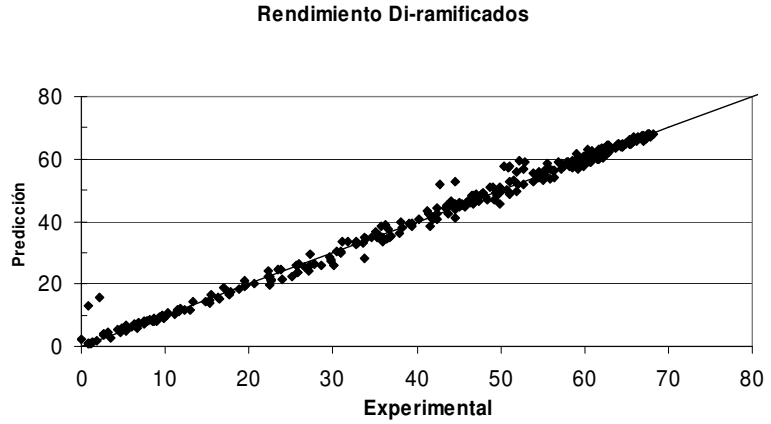

Figura 6.13: Predicciones de rendimiento a di-ramificados obtenidas para la reacción n-octano mediante la RNA 4-8-6-3 empleando la función de activación tangencial en sus nodos ocultos.

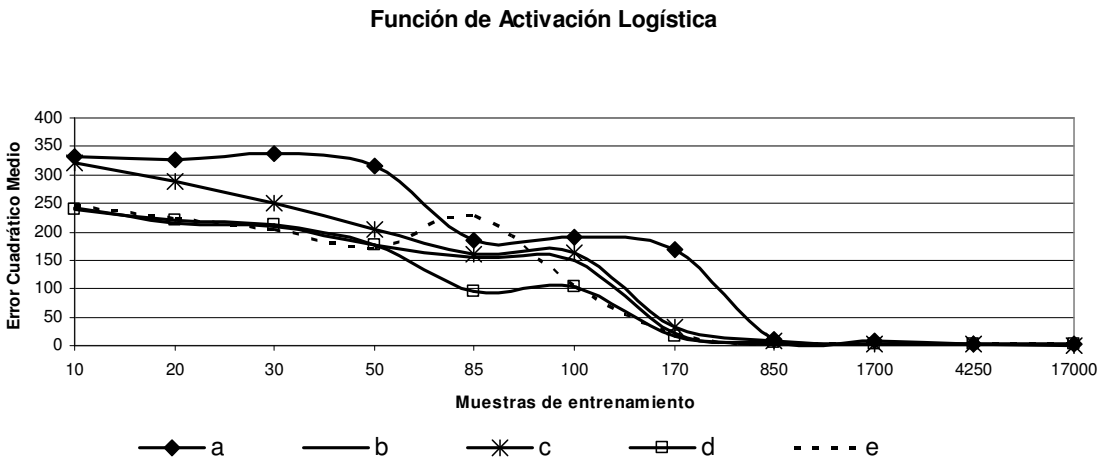

Figura 6.14: Error cuadrático medio cometido en la predicción de la conversión para la reacción n-C_8 con la RNA 4-8-6-3 empleando la función de activación logística en sus nodos ocultos y diferentes algoritmos de entrenamiento: (a) Retro-propagación del error ($\mu = 0.2$); (b) Retro-propagación del error ($\mu = 0.8$); (c) Retro-propagación del error ($\mu = 0.5$); (d) Retro-propagación del error con momento ($\mu = 0.5, \alpha = 0.8$); (e) Retro-propagación del error con momento ($\mu = 0.5, \alpha = 0.5$).

A la vista de los resultados, puede apreciarse que no existen grandes diferencias con respecto al algoritmo de entrenamiento utilizado cuando se dispone de un gran número de muestras de entrenamiento. Sin embargo, cuando se dispone de pocas muestras de entrenamiento (85 o menos), se obtienen mejores resultados empleando el algoritmo de retro-propagación del error con momento al utilizar funciones de activación logísticas, mientras que los resultados son mejores con el algoritmo de retro-propagación al usar funciones de activación tangenciales.

Por otra parte, los resultados sugieren que es más conveniente el empleo de funciones de activación tangenciales, puesto que reduce el número de muestras de entrenamiento necesarias para ofrecer resultados aceptables. Así, las redes que emplean funciones tangenciales aproximan los resultados de la reacción n-octano de forma aceptable siendo entrenadas con tan sólo 85 muestras, mientras que aquellas que emplean funciones logísticas requieren de un mínimo de 170 muestras para ofrecer resultados parecidos.

Por tanto, para modelar las diferentes reacciones de n-parafinas propuestas se empleará la RNA 4-8-6-3 empleando una función de activación tangencial en sus nodos ocultos y utilizando el algoritmo de entrenamiento de retro-propagación del error.

Con respecto al método propuesto para obtener rápidamente bibliotecas de modelos de los procesos catalíticos, en la figura 6.16 muestra una representación gráfica del error medio (normalizado como $error/2(\sigma)$) cometido por los diferentes modelos obtenidos al

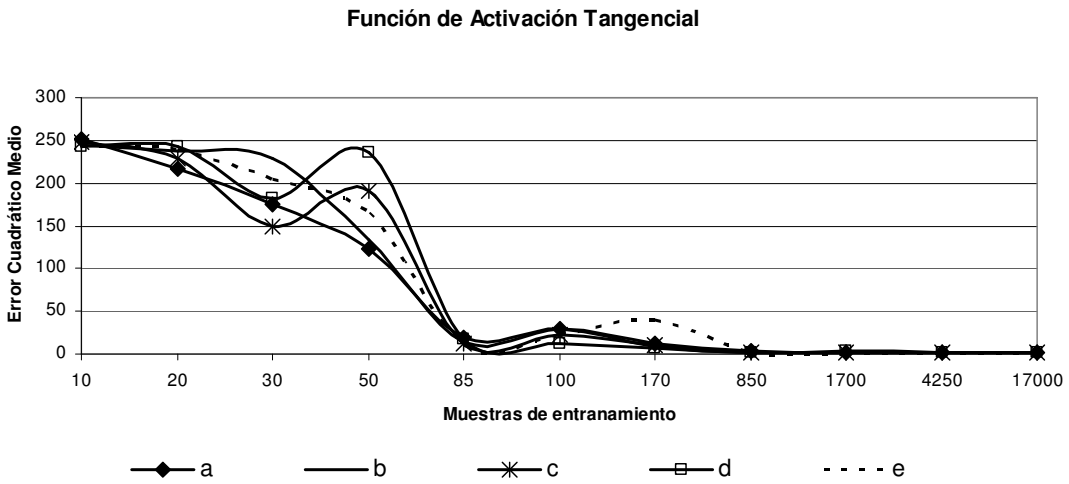

Figura 6.15: Error cuadrático medio cometido en la predicción de la conversión para la reacción n-C_8 con la RNA 4-8-6-3 empleando la función de activación tangencial en sus nodos ocultos y diferentes algoritmos de entrenamiento: (a) Retro-propagación del error ($\mu = 0.2$); (b) Retro-propagación del error ($\mu = 0.8$); (c) Retro-propagación del error ($\mu = 0.5$); (d) Retro-propagación del error con momento ($\mu = 0.5, \alpha = 0.8$);(e) Retro-propagación del error con momento ($\mu = 0.5, \alpha = 0.5$).

predecir los tres resultados catalíticos bajo estudio, es decir, la conversión del hidrocaburo, el rendimiento del isómero a mono-ramificados y el rendimiento a di-ramificados. Las buenas predicciones vienen representadas por las áreas más oscuras (que presentan errores normalizados menores a 0.1), mientras que las malas predicciones aparecen en las áreas más claras. Por ejemplo, empleando como *reacción A* a la reacción de n-octano, se aprecia que se obtienen buenos resultados tomando como *reacción B* a las reacciones de n-heptano, n-hexano y n-pentano (áreas oscuras de las primeras filas de la figura 6.16), tomando tan sólo 20 o 30 muestras de re-entrenamiento procedentes de las mismas. Sin embargo, al no realizar el proceso de re-entrenamiento, se obtienen predicciones de baja calidad (áreas claras en las primeras filas).

En general, puede apreciarse que los modelos entrenados inicialmente con un gran número de muestras de una *reacción A* y re-entrenados posteriormente con unas pocas muestras de una *reacción B*, ofrecen mejores predicciones sobre los rendimientos y conversiones de la *reacción B* que un modelo que sólo posea conocimiento de esas pocas muestras (esté entrenado sólo con ellas). Concretamente, este comportamiento puede apreciarse comparando los resultados de $RNA_{C_8C_7(850,20)}$ y $RNA_{C_7(85)}$, o comparando las figuras 6.17d y 6.18.

Las mejores predicciones aparecen en mayor medida en los modelos RNA_{AB} dónde se ha seleccionado como *reacción A* a una n-parafina con un mayor número de hidrocarburos

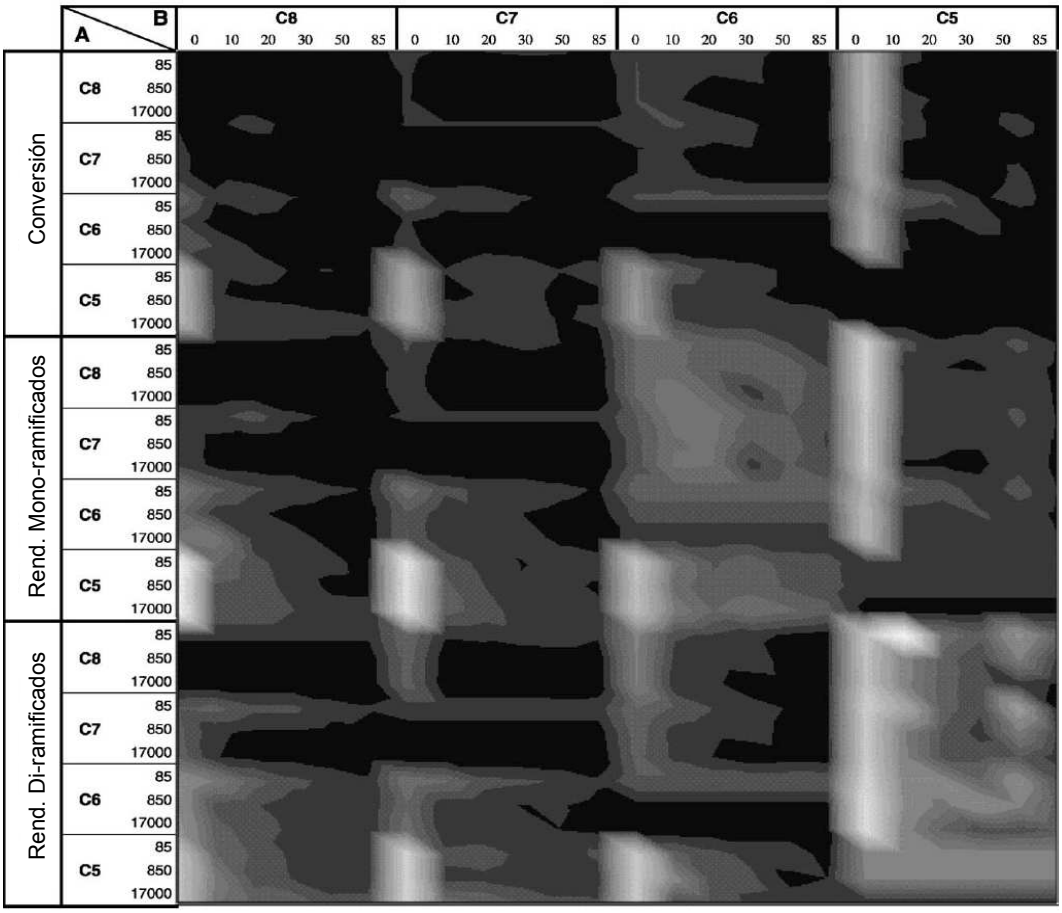

Figura 6.16: Representación gráfica de los errores de predicción medios normalizados ($error/2(\sigma)$) para la conversión y rendimientos para mono y di-ramificados de todas las pruebas realizadas. Las zonas negras representan predicciones con errores normalizados menores a 0.1, mientras que las áreas blancas indican malas predicciones que han alcanzado el error normalizado máximo de 1.

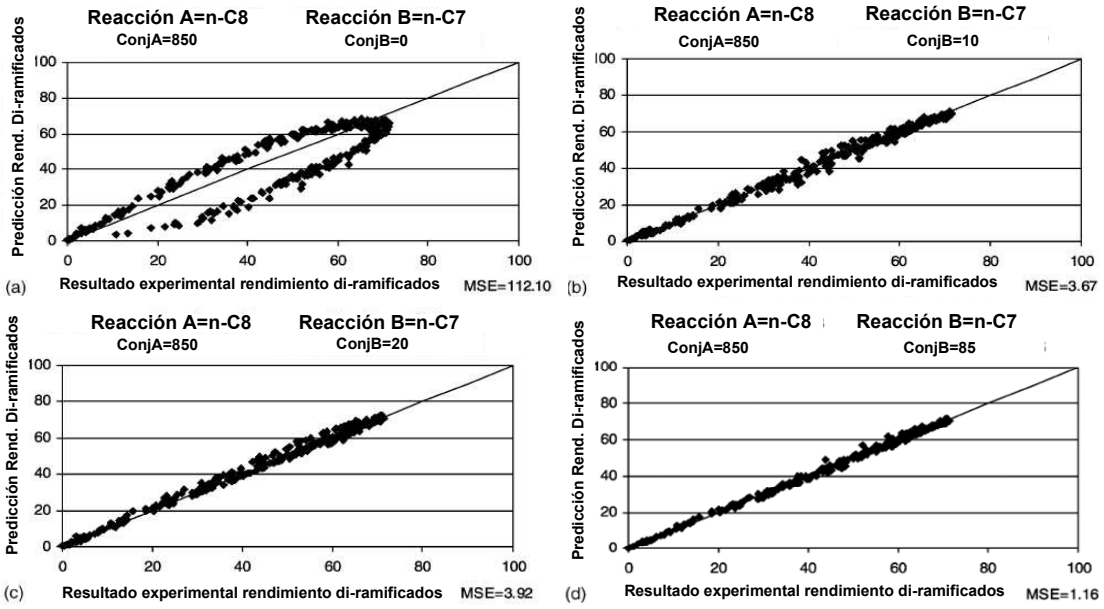

Figura 6.17: Influencia del número de muestras de los ConjB (re-entrenamiento) en los resultados ofrecidos por los modelos para predecir el rendimiento a di-ramificados de 300 muestras de test de la reacción n-heptano (reacción B). Para obtener el modelo se parte de una RNA entrenada con 850 muestras de la reacción n-octano (reacción A) y entrenada con diferentes conjuntos de muestras de la reacción n-heptano: (a) ninguna, sin realizar re-entrenamiento; (b) 10 muestras; (c) 20 muestras; (d) 85 muestras. También se indica el error cuadrático medio cometido en cada test.

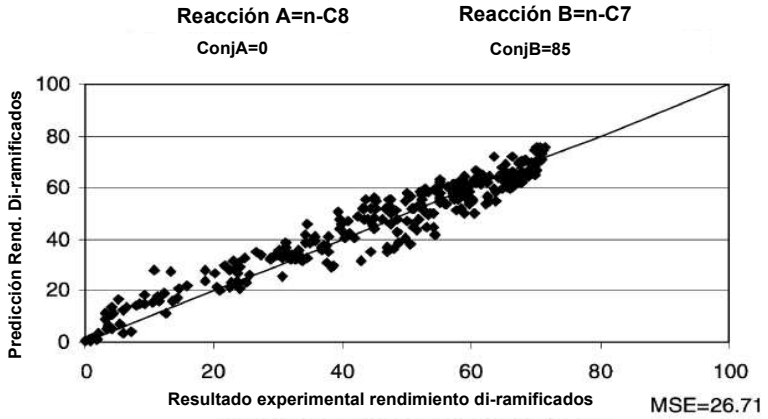

Figura 6.18: Predicciones del rendimiento a di-ramificados de 300 muestras de la reacción n-heptano (reacción B) empleando una red neuronal sin conocimiento previo (ConjA=0) y entrenada sólo con 85 muestras de n-heptano (ConjB). También se indica el error cuadrático medio alcanzado.

que la *reacción B* (por ejemplo, $A = C_8$ y $B = C_7$). Esto puede explicarse, desde un punto de vista químico, teniendo en consideración que una n-parafina con mayor número de carbonos implica un mayor número de procesos reactivos en el proceso de isomerización y por tanto el conocimiento adquirido por la RNA es mayor. Por tanto, estos modelos tienen una mayor capacidad de aprender sobre reacciones similares con un proceso reactivo más sencillo. Por contra, cuando los esquemas reactivos son muy diferentes, se obtienen errores medios más elevados (áreas claras de la figura 6.16).

Asimismo, en la figura 6.16 puede apreciarse que se aproximan mejor los valores de la conversión alcanzada que los valores de rendimiento a mono y di-ramificados, alcanzando resultados aceptables con pocas muestras de re-entrenamiento (20 o incluso 10 muestras). Sin embargo, para los valores de rendimiento es necesario un mayor número de muestras para obtener buenos resultados, tanto para el conjunto de entrenamiento (ConjA) como de re-entrenamiento (ConjB).

A la vista de los resultados obtenidos, la metodología propuesta para obtener bibliotecas de modelos para problemas similares resulta efectiva. Así, en la figura 6.17 pueden compararse los resultados ofrecidos por modelos entrenados para una reacción diferente a la que se pretende predecir, empleando el proceso de re-entranamiento o sin utilizarlo. En concreto en la figura 6.17a, se muestran las predicciones del rendimiento a di-ramificados alcanzado para 300 muestras de n-heptano ofrecidas por una red entrenada únicamente con 850 muestras de la reacción n-octano ($RNA_{C_8C_7(850,0)}$). Puede apreciarse que las predicciones no son muy acertadas, puesto que sólo posee conocimiento de una reacción diferente a la que se pretende aproximar. Sin embargo, cuando este modelo $RNA_{C_8C_7(850,0)}$ es re-entrenado con 10 muestras de la reacción n-heptano ($RNA_{C_8C_7(850,10)}$), se produce una mejora considerable en las predicciones ofrecidas (Fig. 6.17b), reduciéndose de forma notable el error cuadrático medio cometido. Del mismo modo, cuando emplean más muestras en el proceso de re-entrenamiento, se obtienen mejores modelos con predicciones de buena calidad y errores cuadráticos medios más bajos, tal y como puede apreciarse al emplear 20 (Fig.6.17c) y 85 (Fig.6.17d) muestras de la reacción n-heptano.

Por otra parte, la figura 6.18 muestra las predicciones ofrecidas por un modelo construido a partir de 85 muestras de la reacción n-heptano ($RNA_{C_7(85)}$) para el mismo conjunto de muestras de test anterior . Puede observarse que los resultados ofrecidos por este modelo, que sólo posee conocimiento de la reacción n-heptano, son peores que aquellos ofrecidos por los modelos obtenidos mediante el proceso de re-entrenamiento

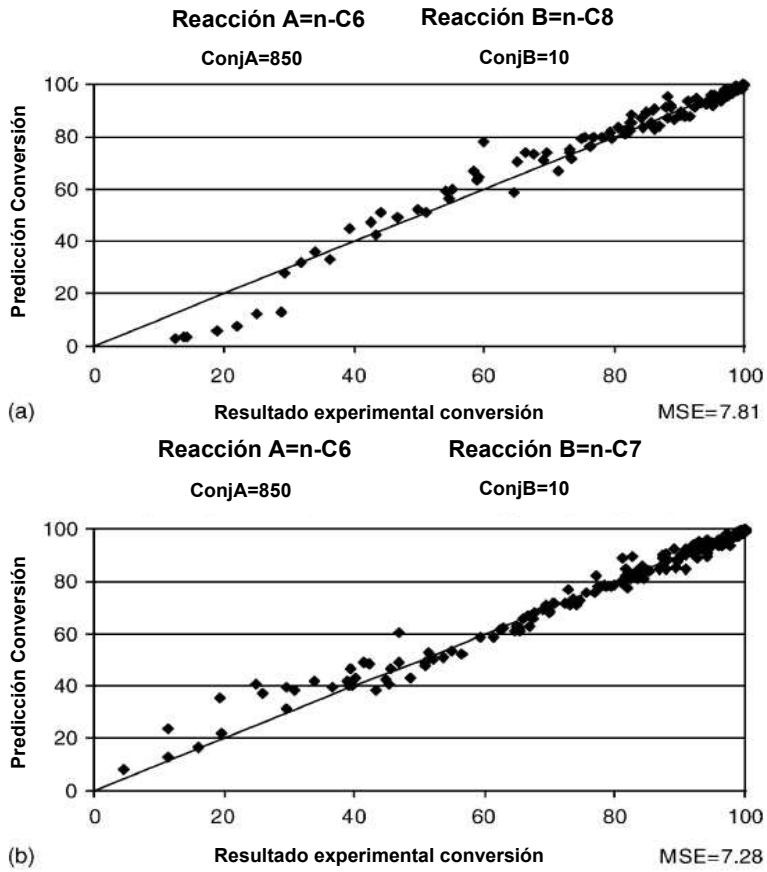

Figura 6.19: Predicciones de la conversión alcanzada para 300 muestras de test empleando una RNA entrenada con 850 muestras de n-hexano (reacción A): (a) re-entrenada con 10 muestras de n-octano (reacción B); (b) re-entrenada con 10 muestras de n-heptano (reacción B). También puede observarse el error cuadrático medio alcanzado.

descrito anteriormente, empleando un menor o igual número de muestras procedentes de la reacción n-heptano (Fig.6.17b-d). Por tanto, puede ser más útil emplear como punto de partida un modelo que posea un conocimiento previo sobre el esquema reactivo y los parámetros cinéticos de una reacción previa similar, que utilizar una RNA vacía.

Con respecto al número de muestras requeridas para los procesos de entrenamiento y re-entrenamiento propuestos, se obtienen buenos resultados cuando se emplean conjuntos de entrenamiento de 850 muestras, mientras que sólo son necesarias 10 o 20 muestras de re-entrenamiento para obtener resultados aceptables. Además, se observa que cuando se emplean modelos de partida (RNA_A) más informados (entrenados con 850 o 17000 muestras de la *reacción A*), los resultados obtenidos tras re-entrenar con muestras de la *reacción B* son mejores.

En la figura 6.19 se pueden apreciar los resultados de las predicciones de conversión realizados sobre 300 muestras de la reacción n-octano (Fig. 6.19a) y n-heptano (Fig. 6.19b). En este caso, para obtener el modelo se partió de una red entrenada con 850 muestras de la reacción n-hexano ($RNA_{C_6(850)}$) y re-entrenada posteriormente con 10 muestras de la reacción n-octano ($RNA_{C_6C_8(850,10)}$) y 10 muestras de la reacción n-heptano respectivamente ($RNA_{C_6C_7(850,10)}$). Puede observase que la calidad de las predicciones para ambos casos es similar.

Así pues, a la vista de los diferentes resultados, puede apreciarse que cuánto más similares son las *reacciones A y B* (atendiendo a como son sus esquemas reactivos), más sencillo es el proceso de obtención de modelos precisos, requiriéndose menos muestras de re-entrenamiento.

Finalmente, se ha estudiado el impacto que pudiera tener el error experimental cometido al obtener las muestras de re-entrenamiento. Así, en la figura 6.20 se muestran las predicciones para la conversión, el rendimiento a mono y a di-ramificados para 300 muestras de n-heptano, ofrecidas por modelos inicialmente entrenados con 850 muestras de la reacción n-octano y re-entrenado con 10 muestras procedentes de la reacción n-heptano ($RNA_{C_8C_7(850,10)}$), considerando que estas muestras tienen cierto grado de error experimental (Fig.6.20 a) o no lo tienen (Fig.6.20b).

6.1.3. Convergencia del Algoritmo Genético al emplear modelos aproximados de las funciones de aptitud

En este caso se pretende evaluar si un algoritmo genético converge adecuadamente al emplear perceptrones multicapa como modelos aproximados de las funciones de aptitud o *fitness*. Para ello se emplearán los modelos obtenidos para las diferentes reacciones n-parafinas obtenidas en el estudio presentado anteriormente, así como las funciones cinéticas disponibles para dichas reacciones. Además, para evaluar la ideonidad de esta técnica en el ámbito de la catálisis combinatoria, el tamaño de la población a emplear por el algoritmo genético se fijará en treinta y dos muestras, puesto que ésta es la capacidad del reactor con el que se realizó el estudio original.

En primer lugar se empleará la combinación de un algoritmo genético y una red neuronal en el estudio de cómo se comporta el AG al variar sus diferentes parámetros (Tabla 6.4), tomando como caso de estudio la optimización de las condiciones de reacción de la isomerización del n-octano.

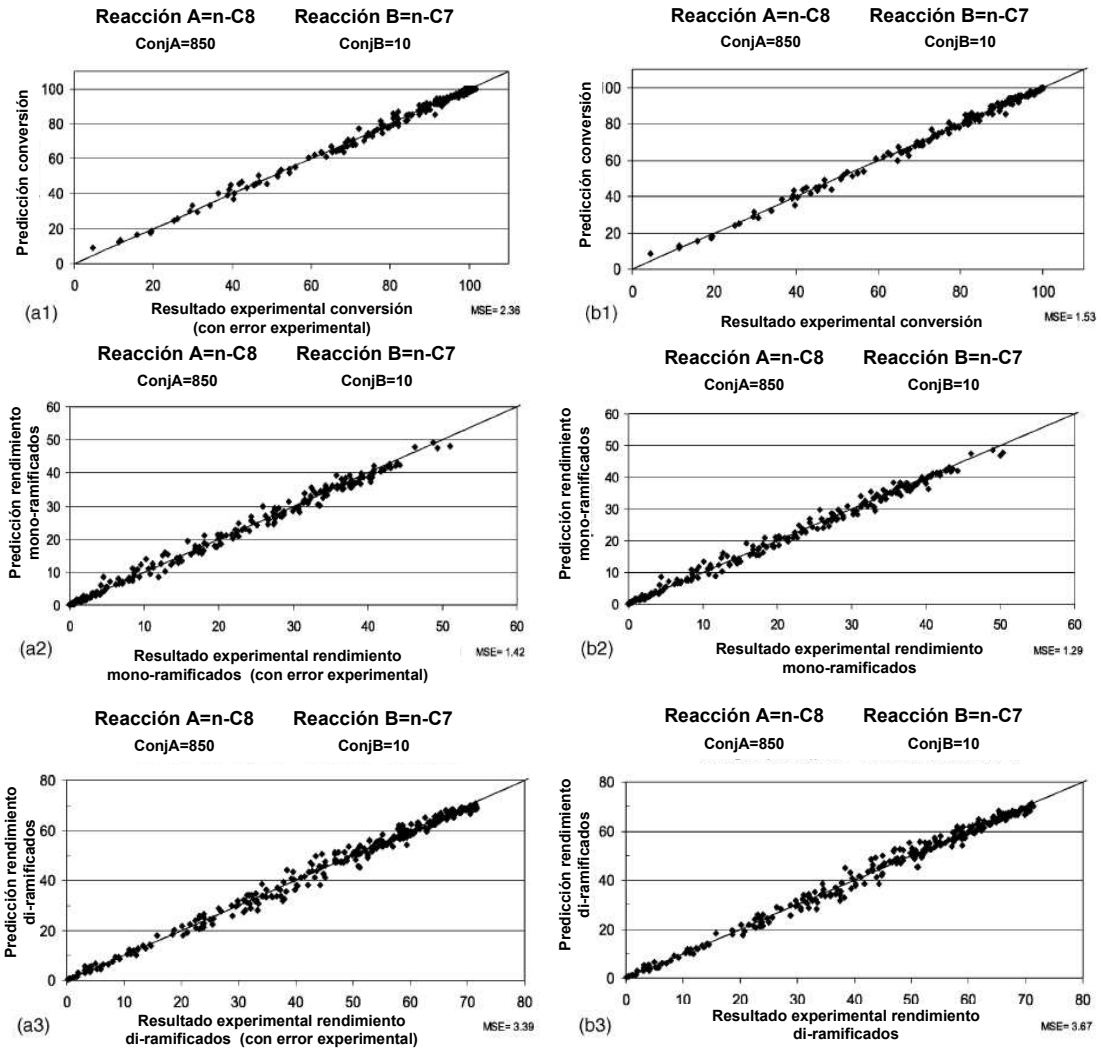

Figura 6.20: Predicciones de la conversión, rendimiento a mono y di-ramificados empleando una RNA entrenada con 850 muestras de n-octano (reacción A) y re-entrenada con 10 muestras de n-hexano (reacción B):(a) se emula un error experimental ($\sigma = 1$); (b) sin error experimental.

En segundo lugar se comprobará cuál es el impacto de la generación inicial en el rendimiento del AG, a fin de establecer qué criterios deben cumplir, empleando para ello la misma reacción de isomerización.

Finalmente, seleccionando los valores de los parámetros que hayan resultado más eficaces y aplicando los criterios necesarios para la obtención de una adecuada generación inicial, se empleará nuevamente el AG combinado con un perceptrón multicapa en la optimización de las condiciones de reacción para las diferentes n-parafinas estudiadas (n-octano, n-heptano, n-hexano y n-pentano). Los perceptrones a emplear serán aquellos que se hayan obtenido en el estudio presentado anteriormente (subsección 6.1.2). Además se compararán los resultados ofrecidos por el algoritmo genético al ser combinado con una red neuronal entrenada con pocas muestras (sólo 85), con respecto a utilizar una red entrenada con un número muy superior de muestras (850). De este modo se comprobará si es posible emplear las redes neuronales como modelos aproximados al realizar una experimentación real dentro del ámbito de la catálisis combinatoria, pues sólo se tendrían las muestras de la generación inicial como punto de partida para entrenar la red (considerando que no se dispusiera de un perceptrón que modelara una reacción similar).

Metodología

El algoritmo genético a emplear es una versión simplificada y preliminar del planteado en la sección 5, diseñado exclusivamente para optimizar las condiciones de reacción para las diferentes n-parafinas estudiadas. Está basado en codificación real, donde una posible solución se codifica mediante cuatro genes correspondientes al valor que toman cada una de las cuatro condiciones de reacción estudiadas: la presión parcial de la n-parafina (de 0.5 a 2 bares), la presión parcial del hidrógeno (6 a 18 bares), la temperatura (de 503 a 543K) y el tiempo de contacto con el catalizador (de 0.16 a 18h).

Operador	Parámetro	Valores				
Mutación	Probabilidad	5 %	10 %	15 %	20 %	
	Número de genes	1	2	3	4	
Cruce	Probabilidad	20 %	40 %	60 %	80 %	100 %
	Progenitores	20 %	30 %	40 %		
	α	0.3	0.5	0.7		
Ajuste Población	Selec. Criterio Calidad	10 %	20 %	30 %	40 %	

Cuadro 6.4: Parámetros del AG y valores que se estudiarán para determinar la mejor combinación

El operador de cruce empleado es el indicado en el apartado 5.4, pero sin contemplar más restricciones que los valores máximos y mínimos establecidos para cada uno de las condiciones. Para determinar el conjunto de individuos a seleccionar como progenitores se emplea el método proporcional de selección de ruleta, que tal y como se explicó en el apartado 2.1.2, asigna a cada individuo una probabilidad de ser seleccionado que es directamente proporcional a la aptitud de dicho individuo.

Con respecto al operador de mutación, se emplea el indicado en el apartado 5.4.a, es decir, se modifica aleatoriamente el valor del gen o genes seleccionados para ser mutados.

La función de aptitud (F) será calculada a partir de las predicciones ofrecidas por las RNA utilizadas para modelar las reacciones n-parafinas, en términos del factor de conversión estimado de la n-parafina lineal en n-parafina ramificada (X) y del rendimiento a dirramificados esperado (Y_{di}), tal y como indica la ecuación 6.2.

$$F = \frac{Y_{di}}{X} * 100 \tag{6.2}$$

Concretamente, en la figura 6.21 se muestra la forma en la que se combinará el AG y el modelo basado en una RNA para realizar la optimización planteada (condiciones de reacción para las diferentes n-parafinas), con el objetivo de estudiar la convergencia del AG cuando se emplea una RNA para aproximar el valor de la función de aptitud. Se parte de una generación aleatoria inicial, cuya aptitud es calculada a partir de las predicciones de conversión y rendimiento ofrecidas por la RNA que modele a la reacción n-parafina bajo estudio. Posteriormente se aplican los operadores de mutación y cruce del AG con la probabilidad establecida para cada uno de ellos, generando nuevos individuos que son añadidos a la población de la generación actual. A continuación, se estima la aptitud de los individuos de esta nueva generación ampliada, empleando para ello las predicciones de la RNA. Finalmente, se ajusta el número de individuos de la población al tamaño indicado, a través del método de selección *rank-space* (sección 2.1.2), que tiene en consideración tanto la aptitud de los individuos como su diversidad. Sin embargo, se establece un porcentaje de individuos (los mejores) que sólo son elegidos atendiendo a su calidad.

Para medir la convergencia del algoritmo genético al ser combinado con la RNA, se tendrá en cuenta la media de la calidad alcanzada por cada generación, entendiéndose por calidad el valor de fitness alcanzado por cada individuo, así como los máximos ab-

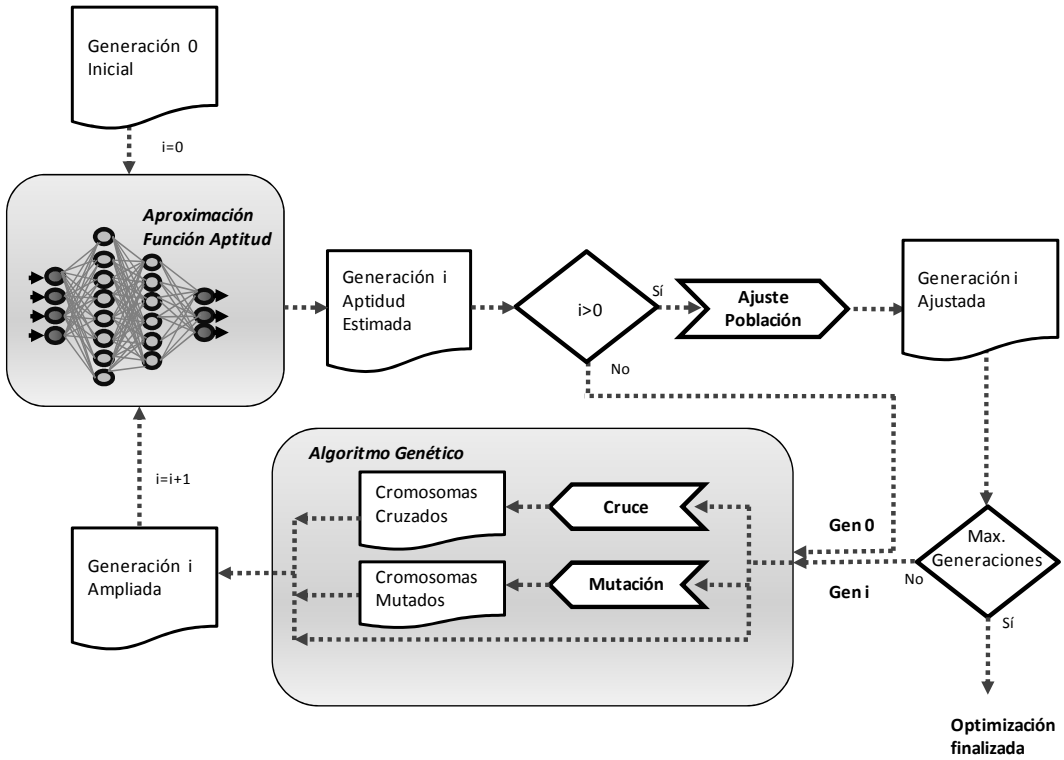

Figura 6.21: Combinación de un AG y una RNA para la optimización de las condiciones de reacción de los procesos de isomerización de n-parafinas.

solutos alcanzados. Como en la aplicación del algoritmo genético existe un componente aleatorio, cada experimento a realizar será repetido en varias ocasiones.

En definitiva, empleando el proceso de optimización planteado (Figura 6.21), van a estudiarse diversos aspectos a fin de validar el correcto comportamiento de un AG al emplear una RNA en el cálculo de su función de aptitud:

Convergencia del AG al variar sus diferentes parámetros. Para realizar este estudio, se aplicará la técnica de optimización descrita anteriormente, variando ciertos parámetros relacionados con el AG y el ajuste de la población (Tabla 6.4), a fin de observar su comportamiento en los distintos casos y establecer un conjunto de valores óptimo para dichos parámetros. Además, cada prueba será repetida cuatro veces, a fin de mitigar los efectos que la componente aleatoria introducida por los operadores del AG pudiera tener en el resultado de cada combinación de parámetros. En estas pruebas se empleará como modelo de la función de aptitud la RNA más informada que se obtuvo en el estudio anterior (sección 6.1.2) para la reacción n-octano (entrenada con 17000 muestras). Concretamente, los parámetros serán establecidos mediante tres pasos:

a) *Obtención de los mejores parámetros relacionados con el operador de mutación.* Para establecer qué probabilidad de mutación (PM) y qué número de genes a mutar en cada ocasión son los más adecuados de entre los valores estudiados (Tabla 6.4), se fijarán el resto de parámetros del AG, tratando de establecer un equilibrio entre el carácter explorador y explotador del operador de cruce. En concreto, los valores de dichos parámetros serán: probabilidad de cruce=60 %; α=0.5; individuos seleccionados como progenitores=30 %; individuos que pasan a formar parte de la siguiente generación atendiendo sólo a criterios de calidad= 20 %.

b) *Determinación de los valores óptimos para algunos de los parámetros relacionados con el operador de cruce.* En concreto se estudiarán de forma conjunta las diferentes combinaciones de probabilidades de cruce (PC) y porcentaje de progenitores (Pr) indicadas en la tabla 6.4. Con respecto a los parámetros relacionados con el operador de mutación, se tomará aquella PM y el número de genes a mutar que se establezcan como mejores en el paso anterior. El resto de parámetros se fijarán al mismo valor que en el punto anterior: α=0.5; individuos que pasan a formar parte de la siguiente generación atendiendo sólo a criterios de calidad= 20 %.

c) *Establecimiento del resto de parámetros.* Para finalizar con la elección de los parámetros más apropiados para la técnica de optimización planteada, se estudiarán las combinaciones de valores indicadas en la tabla 6.4 para el parámetro α y el porcentaje adecuado de individuos que pasan a formar parte de la siguiente generación atendiendo sólo a criterios de calidad (SC). El resto de parámetros se establecerán tomando en cuenta las mejores combinaciones fijadas a través de las pruebas realizadas en los dos pasos anteriores.

Impacto de la generación inicial en el rendimiento del AG. Con respecto al impacto que tiene la configuración de la generación inicial en la convergencia de la técnica de optimización propuesta, se ha planteado una experiencia en la que se comparará la convergencia cuando el punto de partida es una generación inicial cuya calidad media es pésima con respecto a tomar una generación cuya calidad inicial es aceptable. Para ello, también se aplicará la combinación de AG y RNA planteada en esta sección (Figura 6.21), tomando como caso de estudio la optimización de las condiciones de reacción de la isomerización del n-octano. Asimismo, en estas pruebas también se empleará un perceptrón entrenado con 17000 muestras de la reacción n-octano, obtenido anteriormente (sección 6.1.2).

El procedimiento a seguir será generar 100 generaciones de forma aleatoria. Cada una de ellas será evaluada calculando su calidad, empleando el modelo de RNA citado anteriormente y aplicando la fórmula 6.2, que establece la aptitud de cada individuo atendiendo al rendimiento y conversión que alcanza. Seguidamente se seleccionarán aquellas dos generaciones que obtengan la peor y la mejor calidad media.

Estas dos generaciones se tomarán como punto de partida para la realización de dos procesos de optimización sobre la reacción de n-octano, siguiendo el esquema planteado (Figura 6.21), fijando el número de generaciones a obtener en diez. Además, para evitar la distorsión en los resultados que pudiera introducir la componente aleatoria de los operadores del AG, cada optimización será repetida a su vez en diez ocasiones, tomando los resultados medios obtenidos para cada generación inicial bajo estudio.

AG+RNA optimizando diferentes n-parafinas. El último estudio que planteamos en ese apartado consiste aplicar el proceso de optimización (Fig. 6.21) en la búsqueda de las condiciones de reacción óptimas para los procesos de isomerización de las diferentes n-parafinas estudiadas (n-C_8, n-C_7, n-C_6 y n-C_5).

Para ello se emplearán los resultados obtenidos en los apartados anteriores, donde se habrán determinado los mejores parámetros para el AG a emplear, así como el criterio más adecuado para seleccionar la generación aleatoria inicial punto de partida del proceso de optimización.

En este caso, el AG se combinará con dos modelos diferentes para cada una de las n-parafinas (obtenidos en el estudio realizado en la sección 6.1.2), entrenados respectivamente con 85 y 850 muestras de cada proceso reactivo (8 modelos en total). Éste es un número de muestras muy inferior al modelo empleado para obtener los parámetros de AG (entrenado con 17000). De este modo, podremos validar la posibilidad de emplear RNAs como modelos de aptitud desde las primeras fases de los procesos de búsqueda, cuando no se dispone de mucha información.

En definitiva, el proceso de optimización se aplicará combinando el AG con ocho RNAs que modelan los resultados catalíticos (dos para cada reacción) a partir de los cuales se calcula la función de aptitud, siendo 15 el número de generaciones obtenidas en cada ocasión. Además, como el proceso de optimización planteado tiene una fuerte componente aleatoria, cada prueba se repetirá en cinco ocasiones, a fin de obtener los resultados medios de las mismas.

Resultados

A continuación se mostrarán los resultados obtenidos en los estudios planteados en el punto anterior con el objetivo de validar el correcto comportamiento de un AG al emplear una RNA en la aproximación del cálculo de su función de aptitud.

Convergencia del AG al variar sus diferentes parámetros. En este apartado se detallarán los resultados obtenidos en los tres pasos establecidos para parametrizar adecuadamente la combinación de AG y RNA propuesta (Figura 6.21).

a) *Obtención de los mejores parámetros relacionados con el operador de mutación.* En la figura 6.22 se muestran los resultados de las pruebas planteadas para determinar qué parámetros relacionados con el operador de mutación son los más adecuados, es decir, establecer la probabilidad de mutación (PM) y el número de genes a mutar. En concreto, se muestra la calidad media alcanzada en cada generación, tomando como calidad la media de la aptitud de los individuos de la población (32 muestras).

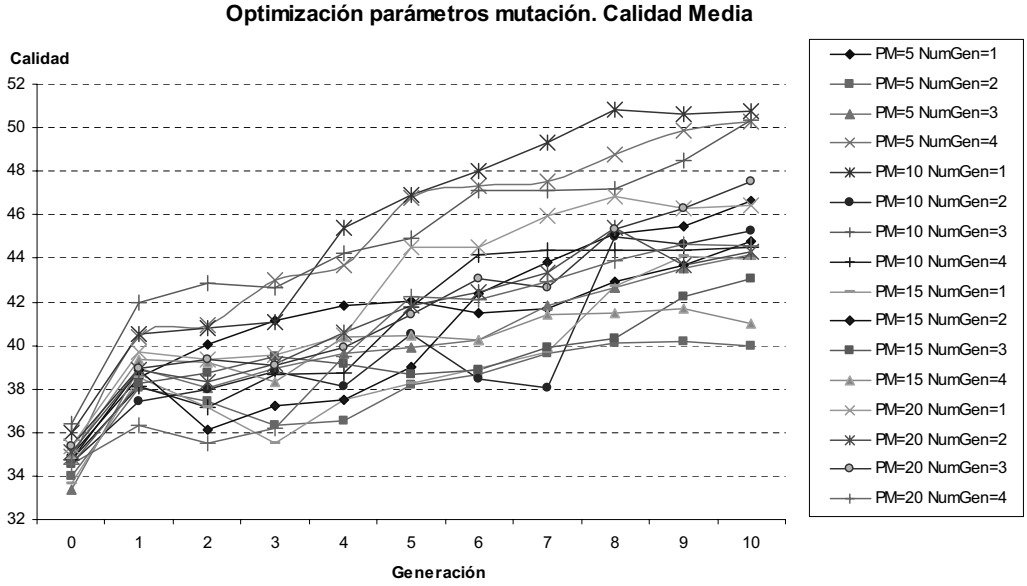

Figura 6.22: Evolución de la calidad media a lo largo de las diferentes generaciones durante el estudio realizado para determinar la mejor probabilidad de mutación (PM) y el número de genes a mutar (NumGen) más adecuados.

Tal y como puede observarse, los mejores resultados se obtuvieron con las combinaciones: (i)PM=10 %, 1 gen a mutar; (ii) PM=5 %, 4 genes a mutar; (iii) PM=10 %, 3 genes a mutar. Se ha seleccionado la primera combinación puesto que es la que finalmente ha ofrecido los mejores resultados y además, mantiene una mayor regularidad en la mejora de la calidad obtenida generación tras generación.

Puede apreciarse que en todos los casos en los que el AG se comporta mejor se introduce una gran componente aleatoria en el proceso, seguramente dedibo a la necesidad de incorporar nuevo material genético en la búsqueda, puesto que la población de cada generación ha tenido que establecerse en un valor muy bajo (sólo 32 individuos).

b) *Determinación de los valores óptimos para algunos de los parámetros relacionados con el operador de cruce.* En la figura 6.23 se muestran los resultados del estudio realizado con las diferentes combinaciones de probabilidades de cruce (PC) y porcentaje de progenitores (Pr) probadas. En concreto, se muestran las calidades medias por generación alcanzadas (Fig. 6.23.a), así como la media de los valores máximos de aptitud o calidad alcanzados para las cuatro combinaciones que ofrecen los mejores resultados (Fig. 6.23.b).

Los valores que toman los parámetros estudiados para las mejores combina-

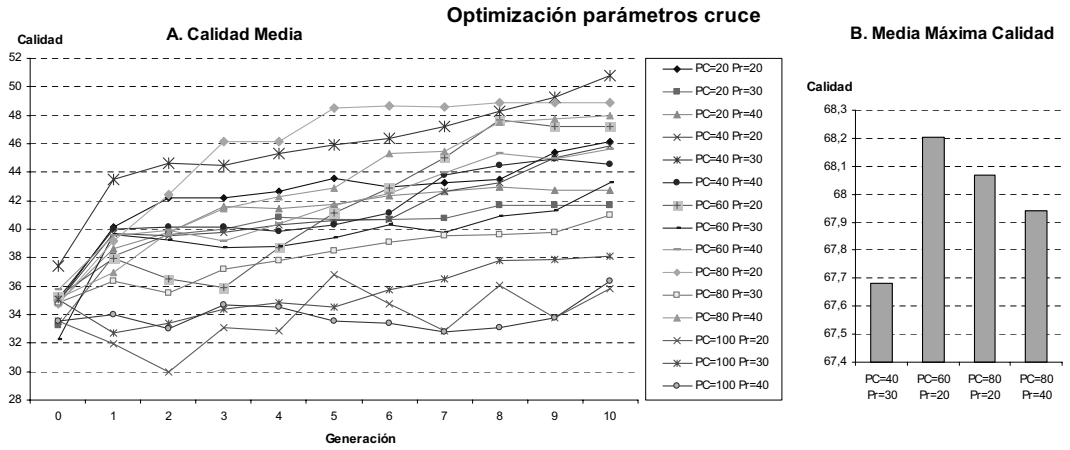

Figura 6.23: Resultados obtenidos en el estudio realizado para determinar la mejor combinación de probabilidad de cruce (PC) y porcentaje de progenitores (Pr), mostrando: (a) calidad media obtenida para cada generación; (b) media de la calidad máxima obtenida en cada prueba.

ciones son: (i) PC=40 %, Pr=30 %; (ii) PC=80 %, Pr=20 %; (iii) PC=80 %, Pr=40 %; (iv) CP=60 %, Pr=20 %. A la vista de los resultados obtenidos, se ha optado por seleccionar la segunda combinación, puesto que ofrece las mejores calidades medias en la mayoría de las generaciones, manteniendo una evolución estable y alcanzando el segundo mejor resultado de aptitud máxima.

c) *Establecimiento del resto de parámetros.* Los resultados de las pruebas llevadas a cabo para establecer un α y un porcentaje adecuado de individuos que pasan a formar parte de la siguiente generación atendiendo sólo a criterios de calidad (SC) pueden observarse en la figura 6.24. En particular, se muestran las calidades medias por generación alcanzadas (Fig.6.4.a), así como la media de los valores máximos de aptitud o calidad alcanzados para las cuatro combinaciones que ofrecen los mejores resultados (Fig.6.4.b).

   Los valores que han ofrecido mejores resultados han sido: (i) α=0.5 , SC=40 %; (ii) α=0.7 , SC=20 %; (iii) α=0.7 , SC=10 %; (iv) α=0.7, SC=40 %. La segunda combinación obtiene una de las mejores tendencias atendiendo a la calidad media alcanzada por generación y además obtiene los individuos con mejor aptitud (mejor calidad máxima).

Atendiendo a los resultados obtenidos, la técnica de optimización planteada en este apartado (Fig 6.21) se parametrizará para el resto de estudios a realizar de la siguiente forma: MP=10 %, 1 gen a mutar, PC=80 %,α=0.7, Pr=40 %, Pr=20 % y SC=20 %.

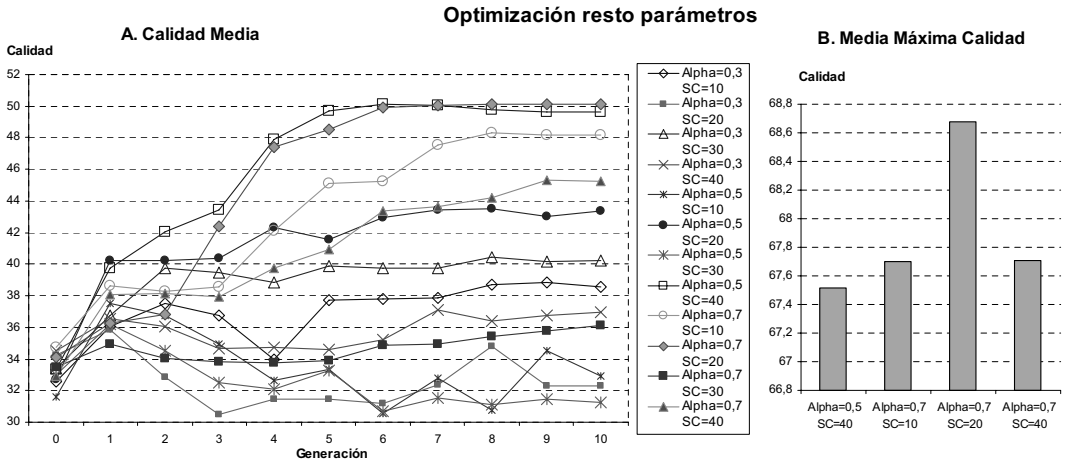

Figura 6.24: Resultados obtenidos en el estudio realizado para determinar la mejor combinación del valor del parámetro α y porcentaje de individuos que se seleccionan para formar parte de la siguiente generación por su calidad (SC), mostrando: (a) calidad media obtenida para cada generación; (b) media de la calidad máxima obtenida en cada prueba.

Impacto de la generación inicial en el rendimiento del AG. En la tabla 6.5 se muestran las calidades medias por generación alcanzadas para los dos procesos de optimización planteados (cada uno de ellos repetido en diez ocasiones), donde un proceso toma como punto de inicio una generación inicial con una calidad media baja y el otro parte de una generación inicial aleatoria con una calidad media aceptable. Puede observarse que la tendencia en la convergencia media es peor cuando se parte de una generación inicial con una baja calidad, sin embargo en ambos casos la técnica consigue converger a máximos similares.

Por tanto, cuando se disponga de un modelo bien informado capaz de aproximar adecuadamente la función de aptitud bajo estudio, sería interesante tener en cuenta tanto la diversidad como la calidad media en la selección de la generación aleatoria inicial.

AG+RNA optimizando diferentes n-parafinas. Tal y como se planteó en este estudio, la técnica de optimización que combina un AG con RNA (Fig 6.21) ha sido parametrizada con los mejores valores obtenidos en el estudio anterior, es decir: MP=10 %, 1 gen a mutar, PC=80 %,α=0.7, Pr=40 %, Pr=20 % y SC=20 %.

Seguidamente se mostrarán los resultados de aplicar el citado proceso de optimización a la obtención de las condiciones de reacción óptimas para las diferentes n-parafinas planteadas. Así, en la tabla 6.6 pueden apreciarse los resultados obtenidos en la 15ªgeneración, mostrando la calidad media en esa generación y

	Generación	Peor Gen.0	Mejor Gen.0
Calidad media			
	0	28.90	40.56
	1	33.33	38.89
	2	30.40	39.49
	3	27.61	39.82
	4	33.38	41.91
	5	32.76	45.09
	6	35.55	46.23
	7	36.03	47.18
	8	34.20	47.50
	9	31.50	47.50
	10	31.49	47.68
Media máximos calidad			
		67.03	68.44

Cuadro 6.5: Evolución de la calidad media obtenida por la combinación del AG y la RNA cuando la generación inicial (generación 0) tiene una baja calidad contra la calidad obtenida cuando la generación de partida es de muy buena calidad.

la media de la calidad máxima alcanzada en la prueba (entendiendo que cuanto mayor es la aptitud de un individuo, mejor es su calidad), para todas las combinaciones de AG y RNA estudiadas. Se aprecia que en todos los casos la combinación de AG y RNA converge a máximos aceptables.

Por otra parte, se aprecia que las aptitudes predichas por los modelos entrenados con un número menor de muestras son algo mejores que aquellas cuya función de aptitud (ecuación 6.2) se aproxima con un modelo más informado. Esto es debido a que las RNA entrenadas con sólo 85 muestras tienen mayor margen de error en sus predicciones, sobre todo en la predicción del resultado del rendimiento a di-ramificados (Y_{di}), para el que suele indicar un rendimiento algo superior al real. En la tabla 6.6 también pueden compararse las medias de los errores absolutos cometidos en las predicciones de las conversiones (X) y rendimientos (Y_{di}) para los dos modelos empleados.

Sin embargo, a pesar del diferente grado de precisión de los modelos empleados, en ambos casos los conjuntos de individuos mejores y peores son prácticamente idénticos, tomando valores muy similares para sus genes (condiciones de reacción a optimizar). Así, en la tabla 6.7 se muestra la aptitud de los cinco mejores y peores individuos obtenidos en la última generación calculada para la reacción n-octano empleando una RNA entrenada con 85 muestras. Esta 15^{a}generación ha sido también evaluada empleando como modelo la RNA entrenada con 850

	RNA-85				RNA-850			
Reacción	Calidad Media	Calidad Máxima	X	Y_{di}	Calidad Media	Calidad Máxima	X	Y_{di}
C8	59.95	74.13	0.050	0.081	60.68	68.55	0.013	0.029
C7	62.05	77.48	0.075	0.102	59.87	70.80	0.019	0.036
C6	64.30	87.79	0.142	0.188	65.32	82.85	0.025	0.047
C5	14.18	20.81	0.050	0.413	09.02	16.01	0.024	0.400

Cuadro 6.6: Se muestran los resultados obtenidos en la generación 15 para cada combinación de RNA y AG probadas (calidades medias y máximas), así como la media de los errores absolutos cometidos en la predicción de los resultados catalíticos utilizados en el cálculo de la función de aptitud (X y Y_{di}). Para cada reacción n-parafina estudiada, el AG se ha combinado con dos modelos diferentes, uno entrenado con 85 muestras (RNA-85) y otro con 850 (RNA-850).

Mejores Individuos				Peores Individuos			
RNA-85		RNA-850		RNA-85		RNA-850	
Individuo	Calidad	Posición	Calidad	Individuo	Calidad	Posición	Calidad
1	70.62	1	67.87	28	51.62	28	45.11
2	69.95	2	68.33	29	36.96	29	35.64
3	69.70	4	67.33	30	36.12	30	33.31
4	69.66	5	67.11	31	32.07	31	28.51
5	69.40	6	66.73	32	12.24	32	13.63

Cuadro 6.7: Comparativa de las calidades alcanzadas por los cinco mejores y peores individuos de la última generación obtenidos para la reacción de n-octano combinando el AG con un modelo entrenado con 85 muestras (RNA-85) con los valores de calidad y posición al predecir esa misma generación con una red entrenada con 850 muestras (RNA-850).

muestras, a fin de obtener su aptitud y ordenar los diferentes individuos según la nueva aproximación de su calidad, para estimar la posición que alcanzarían en la generación utilizando un modelo más informado. Puede apreciarse que en ambos casos los individuos obtienen casi la misma posición o *ranking*.

Por tanto, es viable emplear un AG combinado con una RNA entrenada sólo con 85 muestras para aproximar el valor de la función de aptitud, puesto que a pesar de que introduce cierto error en la calidad esperada, sí predice adecuadamente las tendencias, determinando correctamente los peores y mejores individuos de una generación.

6.1.4. Evaluación de las etapas propuestas en la arquitectura

Para evaluar las distintas fases que conforman la arquitectura propuesta, así como la efectividad de las mismas, se emplearán funciones hipotéticas que modelen el comportamiento de reacciones catalíticas.

En primer lugar, a fin de evaluar el rendimiento ofrecido por el algoritmo genético propuesto combinado con una red neuronal (sección 5), se empleará una función hipotética previamente utilizada en [Wolf et al., 2000] con otra técnica de optimización evolutiva, a fin de comparar los resultados obtenidos. Para este caso se empleará la etapa de actuación del algoritmo combinado con la red neuronal de forma independiente del resto de la arquitectura propuesta. Así, en la sección 6.1.4.1 se aplicarán las etapas I, III, IV y VI de la arquitectura para la optimización de la citada función hipotética.

La función hipotética a emplear (ecuación 6.3) describe la dependencia de la composición del catalizador con respecto al rendimiento catalítico final en la reacción de deshidrogenación oxidativa del propano (ODHP), reemplazando al proceso experimental. Las variables del catalizador consideradas en este modelo son el contenido ($x_i \in [0, 1]$) de cinco elementos diferentes: V, Mg, Mo, Mn y Fe. La función objetivo (función 6.3) a maximizar es el rendimiento del propileno (Y %), cuyo máximo está cercano al 7.5 %.

$$Y = S \cdot X \tag{6.3}$$

$$S = [66 \cdot x_V \cdot x_{Mg} \cdot (1 - x_V - x_{Mg}) + 2 \cdot x_{Mo} - 0.1 \cdot (x_{Mn} + x_{Fe})] \tag{6.4}$$

$$X = [66 \cdot x_V \cdot x_{Mg} \cdot (1 - x_V - x_{Mg}) - 0.1 \cdot x_{Mo} + 1.5 \cdot (x_{Mn} + x_{Fe})] \tag{6.5}$$

$$\sum x_i = 1, x_i \geq 0 \tag{6.6}$$

En segundo lugar, se estudiará la validez del resto de etapas propuestas (configuración, re-entrenamiento de la red neuronal, evaluación de simulación, etc), empleando para ello otra función hipotética (función 6.7) que modela el comportamiento de un catalizador formado por múltiples componentes.

$$Y(x_1, x_2, x_3, x_4, x_5) = z_i(x_1, x_2) + z_j(x_2, x_3) \cdot z_k(x_3, x_4, x_5) \tag{6.7}$$

donde:

$$\sum x_i = 100, x_i \geq 0$$

$$z_i(x_1, x_2) = 0.6 \cdot g(100 \cdot x_1 - 35, 100 \cdot x_2 - 35) +$$

$$0.75 \cdot g(100 \cdot x_1 - 10, 100 \cdot x_2 - 10) + 1 \cdot g(100 \cdot x_1 - 35, 100 \cdot x_2 - 10)$$

$$z_j(x_2, x_3) = 0.4 \cdot g(100 \cdot x_2 - 10, 100 \cdot x_3 - 30)$$

$$z_k(x_3, x_4, x_5) = 5 + 25 \cdot (1 - (1 + (x_3 - 0.3)^2 +$$

$$(x_4 - 0.15)^2 + (x_5 - 0.1)^2)^{0.5})$$

$$g(u, v) = 100 - (u^2 + v^2)^{0.5} +$$

$$50 \cdot (sin(1 \cdot (u^2 + v^2)^{0.5}))/((u^2 + v^2 + 0.001)^{0.5})$$

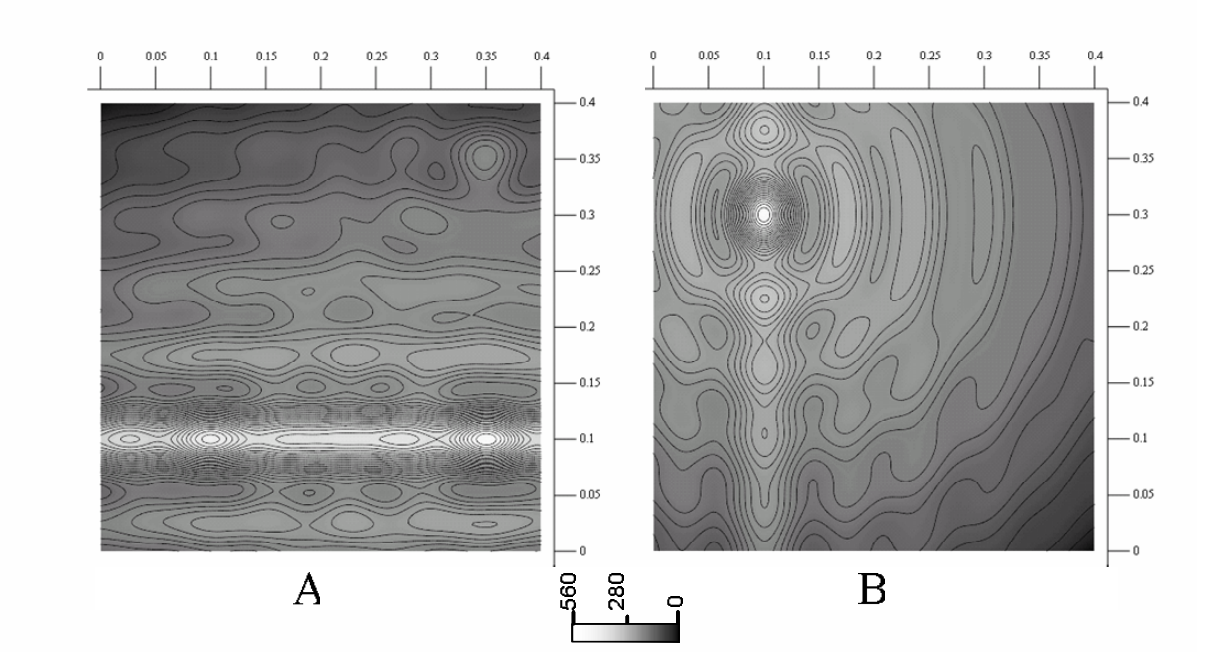

Figura 6.25: Representación de la función hipotética en los planos que contienen los máximos absolutos de la función: A) Variando x_1 y x_2 con $(x_3, x_4, x_5) = (0.3, 0.15, 0.1)$; B) Variando x_2 y x_3 con $(x_1, x_4, x_5) = (0.11, 0.15, 0.1)$

La figura 6.25 muestra una representación de esta función hipotética, donde puede apreciarse que tiene una topología compleja con diferentes máximos locales. La función presenta tres áreas de alta actividad con cierta periodicidad. Este tipo de comportamiento es común para los catalizadores heterogéneos, cuando se varía su composición y condiciones de síntesis. Los valores máximos (cercanos a 550) se encuentran en las áreas más claras de la figura.

En concreto, en la sección 6.1.4.2 se optimizará la función 6.7 empleando todas las etapas propuestas en la arquitectura (I,II,III, IV y VI), a excepción de la etapa de evaluación simulada (V). Seguidamente, en la sección 6.1.4.3 se empleará parte de los datos obtenidos en el estudio anterior para evaluar el método propuesto para la optimización del modelo basado en la RNA, que comprende las tareas realizadas en la etapa II. A continuación, en la sección 6.1.4.4 se optimizará nuevamente la función 6.7, empleando todas las etapas descritas en la arquitectura (I-VI), haciendo hincapié en la evaluación de los beneficios que pueda reportar la etapa de evaluación simulada (V). Finalmente, en la sección 6.1.4.5 se hará uso de la arquitectura *Soft Computing* propuesta haciendo uso de las *generaciones internas*, que permiten ejecutar el AG (etapa IV) en un número determinado de ocasiones por cada ciclo de optimización, proponiendo así varias generaciones que sólo son evaluadas de forma simulada (etapa V) por cada generación de control obtenida en cada ciclo de optimización.

6.1.4.1. Combinación de los operadores del AG con la RNA (Etapas I, III, IV y VI).

Para validar la convergencia obtenida mediante el AG propuesto combinado con una RNA, se empleará esta combinación de forma separada al resto de la arquitectura para optimizar la función 6.3, comparando los resultados que se obtengan con aquellos conseguidos por otra técnica evolutiva empleada anteriormente para resolver el mismo problema [Wolf et al., 2000]. Por tanto, no se realizará un re-entrenamiento de la red neuronal (etapa II), no se producirán generaciones internas y la población estudiada por el genético (población virtual) será del mismo tamaño que la propuesta finalmente en cada ciclo de optimización (sin realizar la etapa V).

Metodología

En primer lugar, se obtendrá un modelo basado en una RNA (tareas comprendidas en la etapa I) de la función 6.3, que ofrecerá aproximaciones del rendimiento a propileno esperado para cada composición de catalizador (cantidades de V, Mg, Mo, Mn y Fe) estudiada por el AG. Seguidamente, se realizará un breve estudio para obtener los valores más eficientes de los diferentes parámetros que definen la actuación de la técnica propuesta en las etapas bajo estudio. Finalmente, se empleará el AG combinado con el modelo basado en la RNA obtenida para hallar la composición del catalizador que

Topología RNA						Algoritmos de Entrenamiento			Conjuntos de muestras		
Nombre	Ent.	1^a	2^{da}	Sal.	Pesos	Nombre	Parám.			Muestras	
							η	μ	Nombre	Entrena.	Test
5_2_2	5	2	-	2	14		0.2	-	Set1	25	10
5_4_2	5	4	-	2	28	Backpropagation	0.5	-	Set2	50	20
5_6_2	5	6	-	2	42		0.8	-	Set3	100	25
5_8_2	5	8	-	2	56				Set4	200	40
5_10_2	5	10	-	2	70		0.2	0.2			
5_12_2	5	12	-	2	84		0.2	0.5			
5_10_4_2	5	10	4	2	98		0.2	0.8			
5_4_3_2	5	4	3	2	38	Backpropagation	0.5	0.2			
5_3_2_2	5	3	2	2	25	wih	0.5	0.5			
						momentum	0.5	0.8			
							0.8	0.2			
							0.8	0.5			
							0.8	0.8			

Cuadro 6.8: Topologías de RNA, algoritmos de entrenamiento y conjuntos de muestras a estudiar

ofrezca mejores resultados, aplicando las etapas III, IV y VI de la arquitectura. A continuación se detallarán las acciones a realizar en cada uno de los estudios citados.

Obtención del modelo de RNA (Etapa I). Siguiendo los pasos descritos en el apartado 5.1.4, se procederá a estudiar diferentes combinaciones de topologías y algoritmos de entrenamiento, tal y como se muestra en la tabla 6.8. Además, se tendrán en consideración conjuntos de muestras de entrenamiento y test de diferente tamaño. Sin embargo, para elegir la topología más adecuada, sólo se observarán los resultados obtenidos por las redes entrenadas con aquellos conjuntos formados por un menor número de muestras (Set1 y Set2). Se ha tomado esta decisión puesto que normalmente en la etapa I de la arquitectura de búsqueda sólo están disponibles las muestras de la generación aleatoria inicial para obtener un modelo adecuado.

Finalmente, se procederá a realizar un estudio de la sensibilidad del modelo obtenido ante muestras que presenten errores experimentales. Para ello, el modelo será entrenado con 50 muestras con errores experimentales emulados (mediante una distribución Gaussiana normalizada), empleando diferentes desviaciones estándar (1, 2, 15 y 20). En cada caso, el modelo aproximará 20 muestras con errores emulados del mismo tipo.

Configuración de los parámetros *Soft Computing* (Etapa I). Para establecer qué parámetros de la arquitectura de búsqueda relacionados con la etapa IV (sección 5.1.2) son los más adecuados para optimizar la función 6.3, se realizarán diversas pruebas empleando diferentes valores para los mismos. Para cada combinación se calcularán 25 generaciones, repitiendo todo el proceso 10 veces, to-

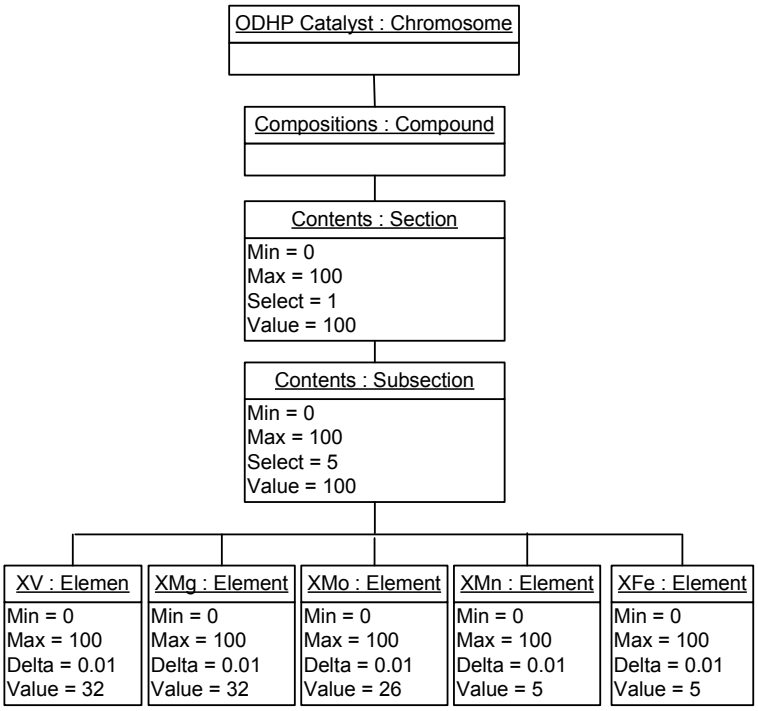

Figura 6.26: Codificación del problema de optimización de un catalizador para la reacción ODHP

mando en consideración los valores medios obtenidos. Como punto de partida, se tomará una generación aleatoria que posea la diversidad necesaria, siguiendo para ello los pasos descritos en el punto 5.1.3. El resto de parámetros se establecerá a: Generaciones internas=0; Ratio de reducción=0 %.

La figura 6.26 describe la codificación empleada por la técnica *Soft Computing* propuesta para optimizar la función hipotética 6.3 (que describe el comportamiento de los catalizadores dependiendo de su composición para la reacción ODHP), en la que pueden observarse los rangos permitidos para cada variable (cantidades de los distintos elementos estudiados).

En primer lugar estableceremos los parámetros relacionados con el operador de mutación, es decir, la probabilidad de mutación (PM) y el número de genes a mutar (Genes), estudiando las posibilidades indicadas en la tabla 6.9. El resto de parámetros se mantendrá a los siguientes valores: $\alpha = 0.5$, población=50 y progenitores=20 %.

En segundo lugar se variarán los dos parámetros relacionados con el operador de cruce: el tamaño del intervalo de confianza (α) y el porcentaje de individuos a seleccionar como progenitores (*progenitores*). Pueden observarse los valores a

Op. Mutación		Op. Cruce		
PM	Genes	α	Progenitores	Población
5	1	0.3	10 %	25
10	2	0.5	20 %	50
15	3	0.7	30 %	75
20	4	0.8	40 %	100
	5	0.9		

Cuadro 6.9: Valores a estudiar para los diferentes parámetros de la arquitectura de búsqueda requeridos en la etapa IV

estudiar en la tabla 6.9. Los parámetros de mutación se fijarán con los mejores valores obtenidos en la prueba anterior, tomando nuevamente una población de 50 individuos.

En último lugar se estudiará el tamaño de población adecuado (tabla 6.9), teniendo en consideración que se pretende minimizar el número de muestras a estudiar en cada generación.

Optimización de la composición de un catalizador (Etapas III, IV y VI). A fin de optimizar la composición de un catalizador a aplicar en la reacción de deshidrogenación oxidativa del propano (ODHP), cuyo proceso reactivo es simulado mediante la función hipotética 6.3 explicada anteriormente, se aplicará la combinación del AG y el modelo ofrecido por la RNA indicada en la arquitectura de búsqueda *Soft Computing* propuesta (etapas III y IV). Los parámetros relacionados con las etapas III y IV a emplear serán aquellos que se establezcan como mejores en los estudios antes planteados. El resto de parámetros de la arquitectura se fijará a: Generaciones internas=0; Ratio de reducción=0 %.

Resultados

Seguidamente se describirán los resultados obtenidos en los tres estudios planteados en este apartado.

Obtención del modelo de RNA (Etapa I). Tras probar las diferentes topologías y algoritmos de entrenamiento planteadas (tabla 6.8), los mejores resultados se han conseguido al emplear el algoritmo de aprendizaje de retro-propagación del error con momento (*Backpropgation with momentum*). Además, las mejores combinaciones de parámetros para este algoritmo han sido: $\eta = 0.8$, $\mu = 0.8$; $\eta = 0.8$,

Set1	Set2
5_10_2	5_10_2
5_6_2	5_6_2
5_12_2	5_12_2
5_10_4_2	
5_8_2	

Cuadro 6.10: Topologías de RNA con mejor comportamiento para los conjuntos de muestras *Set1* y *Set2*.

	Parámetros Entrenamiento		MAE		MSE	
RNA	η	μ	S	X	S	X
5_10_2	0.5	0.8	0.070	0.096	0.009	0.017
5_6_2	0.5	0.8	0.076	0.102	0.012	0.025
5_10_2	0.8	0.8	0.076	0.081	0.011	0.011
5_12_2	0.5	0.8	0.079	0.092	0.011	0.017

Cuadro 6.11: Errores absolutos (MAE) y cuadráticos medios (MSE) obtenidos por las RNA con mejores resultados al modelar la función hipotética 6.3. En todos los casos el algoritmo de entrenamiento empleado es el de retro-propagación del error con momento. El conjunto de muestras para el entrenamiento y test es el *set2*, formado por 70 muestras.

$\mu = 0.5$ y $\eta = 0.5$, $\mu = 0.8$.

Con respecto al estudio de las topologías, en la tabla 6.10 se muestran aquellas que han dado mejores resultados para los conjuntos de muestras *Set1* (formado por 35 muestras) y *Set2* (formado por 70 muestras). Asimismo, en la tabla 6.11 se detallan los errores absolutos (MAE) y cuadráticos medios (MSE) ofrecidos por los tres modelos que se comportaban favorablemente en ambos conjuntos, siendo entrenados con el algoritmo de retro-propagación del error con momento y empleando el conjunto *Set2* de muestras. A la vista de los resultados, la topología seleccionada como modelo para ser empleada posteriormente por la arquitectura de búsqueda posee cinco nodos en su capa de entrada, una única capa oculta con diez nodos y dos nodos en su capa de salida (5_10_2), empleando el algoritmo de aprendizaje de retro-propagación del error con momento (parametrizado con $\eta = 0.8$ y $\mu = 0.8$). En la figura 6.27 se muestran las aproximaciones ofrecidas por este modelo al ser entrenado y probado con los diferentes conjuntos de muestras disponibles. Puede observarse que las aproximaciones ofrecidas por la RNA están muy cercanas a los valores reales de conversión y selectividad, incluso cuando el modelo está entrenado con muy pocas muestras.

En la figura 6.28 pueden apreciarse los resultados obtenidos en el estudio realizado

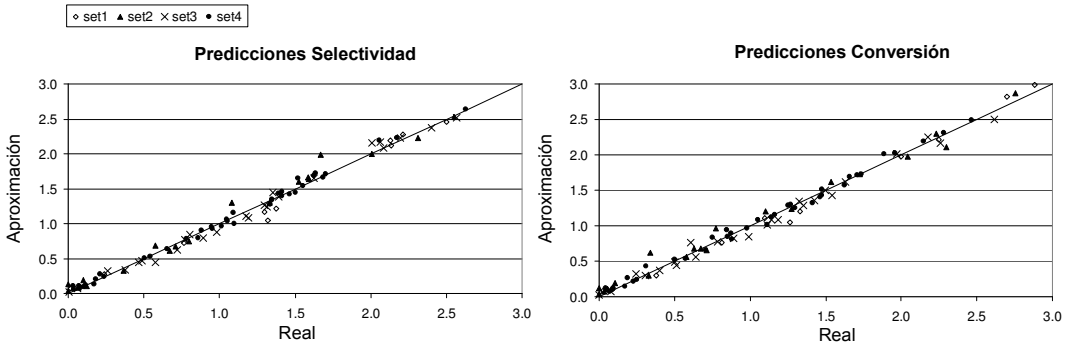

Figura 6.27: Aproximaciones de la conversión y selectividad realizadas por el modelo 5_10_2 entrenado y probado con conjuntos de muestras de diferente tamaño (*Set1, Set2, Set3 y Set4*), empleando como algoritmo de aprendizaje el de retro-propagación del error con momento (parametrizado con $\eta = 0.8$ y $\mu = 0.8$)

para determinar la sensibilidad del modelo ante los posibles errores experimentales. Tal y como puede observarse, la RNA ofrece aproximaciones de buena calidad, incluso al ser entrenada con muestras que presentan errores experimentales.

Configuración de los parámetros *Soft Computing* (Etapa I). En la figura 6.29 pueden observarse los resultados obtenidos en el estudio realizado para determinar los mejores parámetros relacionados con el operador de mutación. Se muestran las aptitudes medias para las generaciones 5, 10 y 15 (es decir, la media de la aptitud o *fitness* obtenido por los 50 individuos estudiados en cada una de esas generaciones). La aptitud media de partida es de 1.03, necesitando pocas generaciones para que el algoritmo converja hacia el óptimo de la función (cercano a 7.5). La combinación de parámetros que ha ofrecido mejores resultados de entre los estudiados (tabla 6.9) ha sido una probabilidad de mutación (PM) de 5 % y 1 gen mutado en cada ocasión (*genes*).

Con respecto a los parámetros relacionados con el operador de cruce, en la figura 6.30 se muestran los resultados obtenidos para las diferentes combinaciones de parámetros estudiadas (tabla 6.9). De forma similar al caso anterior, puede apreciarse la aptitud media obtenida en las generaciones 5, 10 y 15. En este caso, las diferencias entre las diferentes combinaciones no son excesivamente grandes, pero se ha seleccionado la combinación de un tamaño del intervalo de confianza de 0.9 (α) y un 10 % de individuos seleccionados como progenitores (*progenitores*). Esta combinación es la que consigue una mejor media al finalizar la prueba, así como los mejores resultados en la quinta generación, proporcionando una rápida convergencia.

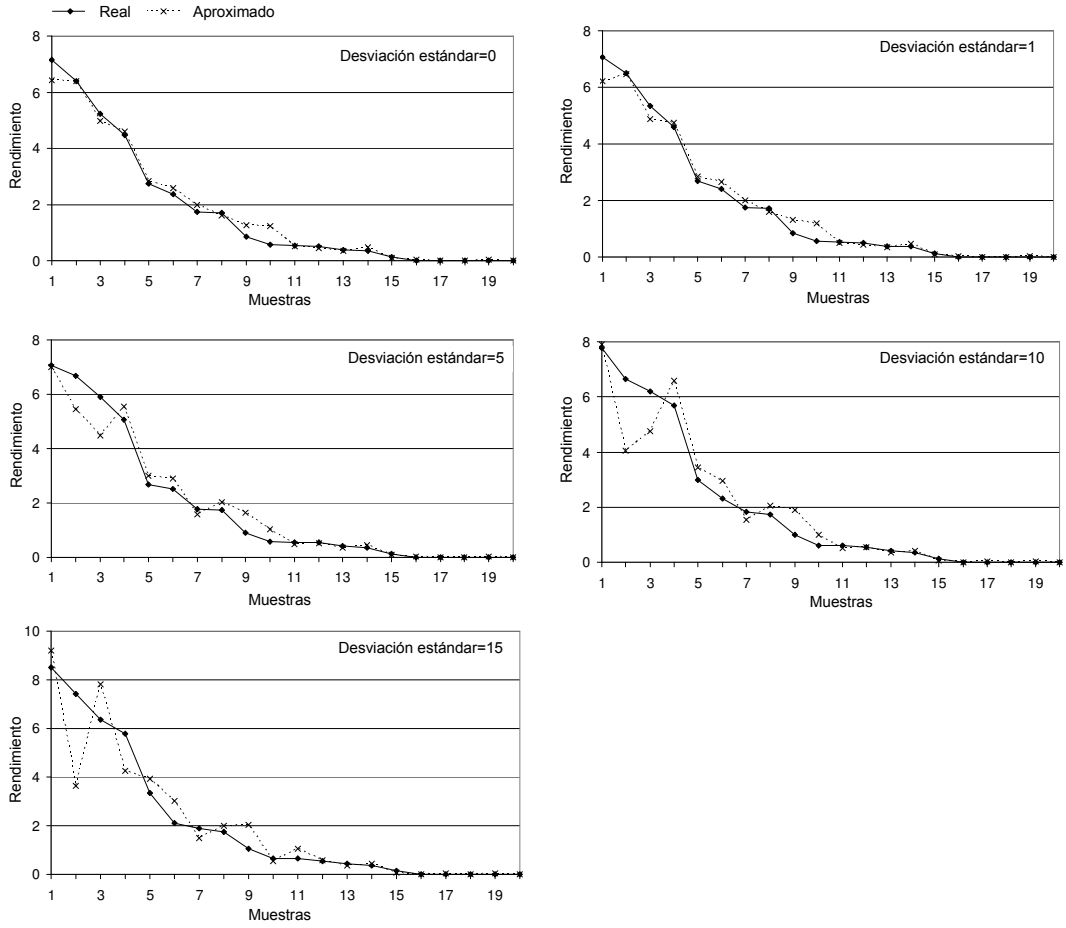

Figura 6.28: Aproximaciones obtenidas mediante el modelo 5_10_2 del rendimiento ($Y = S \cdot X$) para muestras con error experimental emulado. La RNA se ha entrenado con 50 muestras con errores emulados empleando diferentes desviaciones estándar (1, 5, 10 y 15) y probada con 20 muestras con errores emulados del mismo tipo. También se muestra a modo comparativo los resultados de la aproximación cuando no existe error experimental (desviación=0)

La influencia en la convergencia de la técnica *Soft Computing* al variar el tamaño de la población puede apreciarse en la figura 6.31, fijando el resto de parámetros a los valores obtenidos con anterioridad: PM=5, Genes=1, $\alpha = 0.9$ y progenitores=10 %. Tal y como puede apreciarse, no existen grandes diferencias entre los resultados medios obtenidos, ya que en todos los casos el algoritmo converge rápida y adecuadamente (alcanzando la estabilidad a partir de la decimoquinta generación). Por tanto, con tan sólo 25 muestras podrían obtenerse resultados adecuados.

Optimización de la composición de un catalizador (Etapas III, IV y VI). En la figura 6.32 se observan las medias y máximos alcanzados en promedio para las

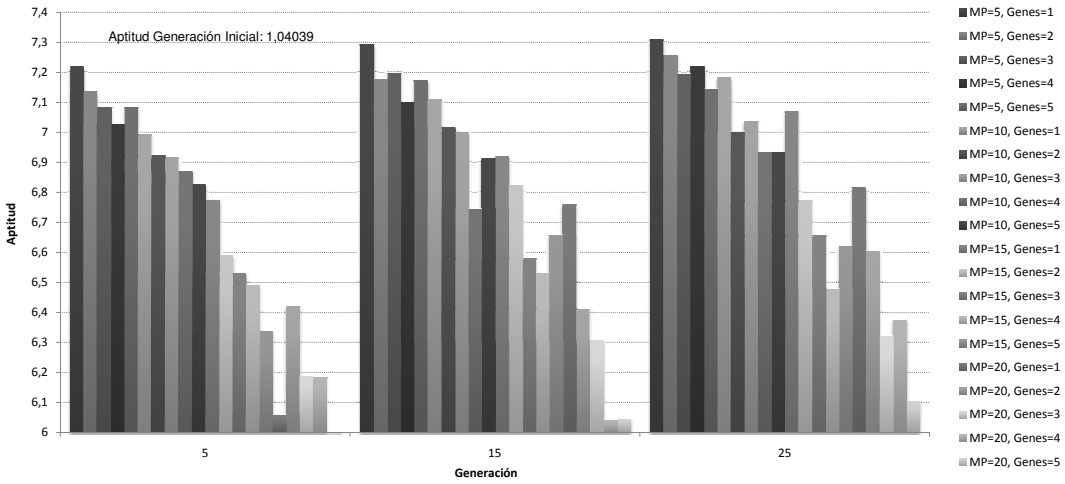

Figura 6.29: Evolución de la aptitud media de las generaciones para cada combinación de parámetros del operador de mutación estudiados. Resto de parámetros: *alpha* = 0.5, población=50 y progenitores=20 %.

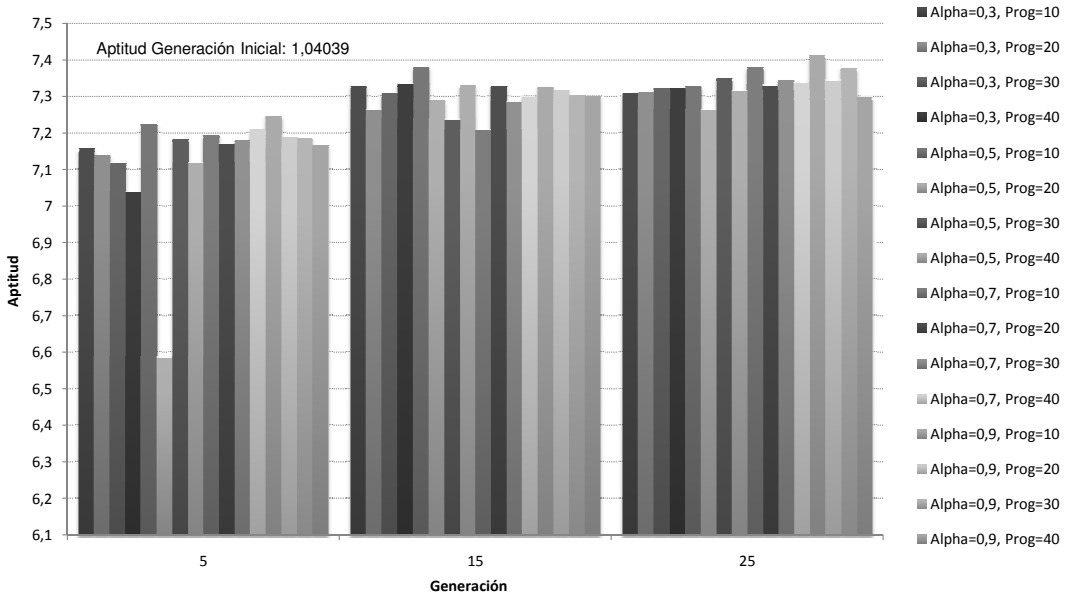

Figura 6.30: Evolución de la aptitud media de las generaciones para cada combinación de parámetros del operador de cruce estudiados. Resto de parámetros: PM=5, Genes=1 y población=50.

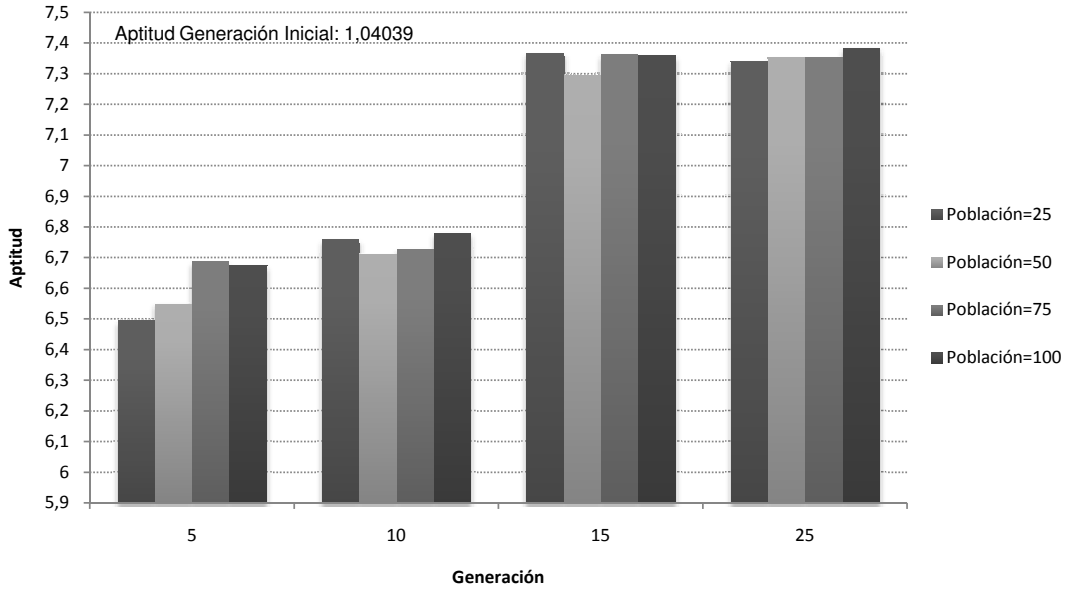

Figura 6.31: Evolución de la aptitud media de las generaciones al variar el tamaño de la población estudiada. Resto de parámetros: PM=5, Genes=1,$\alpha = 0.9$ y progenitores=10 %.

diez ejecuciones del algoritmo realizadas al aplicar la combinación del AG y el modelo ofrecido por la RNA indicada en la arquitectura de búsqueda *Soft Computing* propuesta (etapas III y IV) al optimizar la composición del catalizador para la reacción ODHP (función 6.3). Se han empleado los parámetros obtenidos en las experiencias mostradas anteriormente: PM=5, Genes=1, $alpha = 0.9$, progenitores=10 % y población virtual=25 %; así como los establecidos para no ejecutar el resto de etapas: ratio de reducción=0 % y generaciones internas=0.

La técnica empleada en todo este estudio ofrece mejores resultados que la técnica evolutiva empleada anteriormente para optimizar la misma función 6.3, presentada en [Wolf et al., 2000]. En dicho trabajo se indica que la máxima calidad obtenida fue de 7.5 %, necesitándose 6 generaciones de 100 individuos cada una para obtener este resultado. Tal y como puede apreciarse en la figura 6.32, en nuestro caso sólo es necesario obtener una generación de 25 muestras para igualar ese resultado. Además, la aptitud media de todas las muestras estudiadas se sitúa en torno al máximo (7.558) a partir de la cuarta generación.

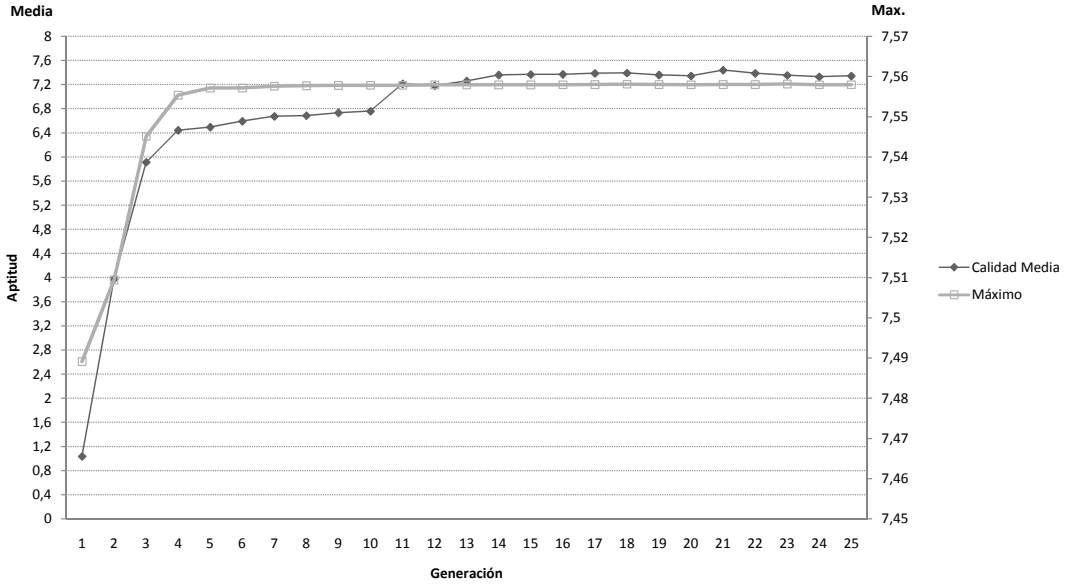

Figura 6.32: Evolución de la aptitud media de las generaciones al optimizar la composición para un catalizador para la reacción de deshidrogenación oxidativa del propano. Parámetros: PM=5, Genes=1,$\alpha = 0.9$, progenitores=10 %, población virtual=25, ratio de reducción=0 %, generaciones internas=0.

6.1.4.2. Optimización de la composición de un catalizador sin emplear evaluación simulada (Etapas I, II, III,IV y VI).

En este apartado se estudiará la validez de las etapas propuestas en la arquitectura, a excepción de la etapa V, que permite trabajar al AG con una población virtual mayor de la que finalmente se evalúa con la función original (población de control de la etapa VI). Para realizar este estudio se empleará la función hipotética 6.7, que modela el comportamiento de un catalizador formado por múltiples componentes. Por tanto, se utilizará la arquitectura de búsqueda propuesta para encontrar las cantidades adecuadas para cada uno de los cinco elementos presentes en el catalizador hipotético planteado.

Metodología

En primer lugar se obtendrá una RNA capaz de modelar adecuadamente la función 6.7, siguiendo para ello los pasos descritos en el apartado 5.1.4. Seguidamente, la arquitectura *Soft Computing* se parametrizará con aquellos valores que han dado mejores resultados en el estudio realizado sobre la función hipotética 6.3: PM=5, Genes=1, $\alpha = 0.9$, progenitores=10 %; así como los establecidos para no ejecutar el resto de pasos posibles de la arquitectura: ratio de reducción=0 % y generaciones internas=0. El

	Parámetros entrenamiento		Rendimiento esperado	
RNA	η	μ	MAE	MSE
5_4_3_1	0.8	0.8	13.969	0.044
5_12_1	0.8	0.5	13.992	0.044
5_8_1	0.5	0.8	14.112	0.045
5_6_1	0.8	0.8	14.179	0.045

Cuadro 6.12: Errores absolutos (MAE) y cuadráticos medios (MSE) obtenidos por las RNA con mejores resultados al modelar la función hipotética 6.7. En todos los casos el algoritmo de entrenamiento empleado es el de retro-propagación del error con momento. Se han utilizado 70 muestras para el entrenamiento (70 %) y test (30 %).

único parámetro sobre el que se realizará un estudio será el número de individuos a obtener en cada generación (*población*).

Para realizar el estudio de la influencia del tamaño de la población en la convergencia de la arquitectura propuesta al optimizar la composición del catalizador para la reacción descrita por la función 6.7, se realizará una batería de pruebas empleando diferentes tamaños de población dentro del rango utilizado normalmente en experimentos empíricos realizados para la obtención de nuevos catalizadores: 15, 35, 55, 75 y 95 individuos. Cada experiencia realizada con una determinada combinación de parámetros será repetida en diez ocasiones, tomando siempre los valores promedios obtenidos, a fin de evitar el efecto que la componente aleatoria presente en la arquitectura pudiera tener en los resultados.

Para cada uno de los tamaños se obtendrá una generación aleatoria inicial, siguiendo los pasos descritos en 5.1.3. Pero además se forzará a que la calidad media de cada una de ellas (media de la aptitud de los individuos que la forman) sea similar, de forma que no existan perturbaciones en los resultados finales a causa de la calidad del punto de partida del proceso de búsqueda.

Resultados

En la tabla 6.12 se recogen parte de los resultados del estudio realizado (combinando diferentes topologías y algoritmos de entrenamiento) para seleccionar una RNA adecuada que modele la función hipotética 6.7, mostrando los errores absolutos medios (MAE) y cuadráticos medios (MSE) obtenidos por los modelos que ofrecían las mejores aproximaciones. En concreto, los mejores resultados los ofrece el modelo formado por una red neuronal con cinco nodos de entrada, cuatro nodos en su primera capa ocul-

ta, tres nodos en la segunda capa oculta y un nodo de salida (5_4_3_1). El algoritmo de entrenamiento seleccionado ha sido el de retro-propagación del error con momento (parametrizado con $\eta = 0.8$ y $\mu = 0.8$).

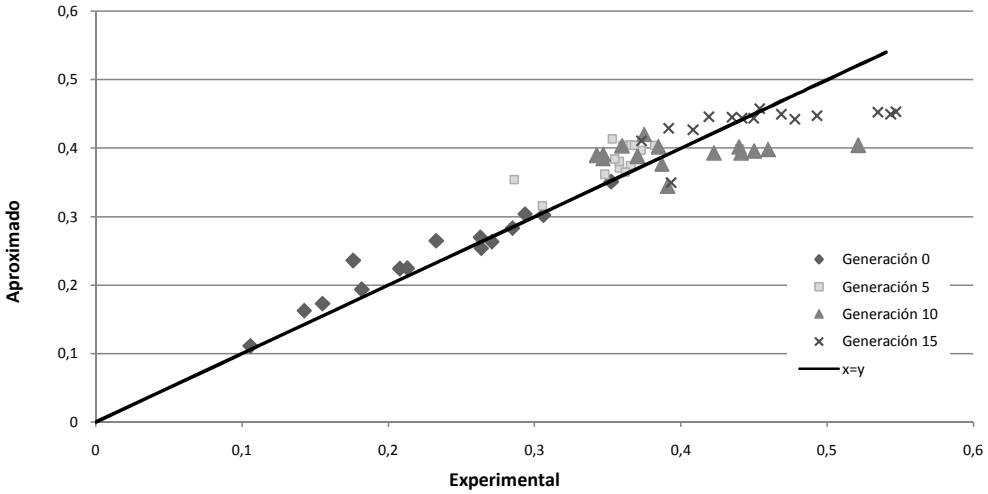

Figura 6.33: Aproximaciones realizadas por la red en la etapa de re-entrenamiento vs. valores experimentales, a lo largo de una experiencia en la que la arquitectura *Soft Computing* evalúa 75 muestras por generación. El modelo se re-entrena con 60 muestras y el test se realiza con 15.

Asimismo, la figura 6.33 muestra las predicciones ofrecidas por ese modelo en una experiencia en la que se estudiaban 75 muestras por generación (60 muestras se emplean para re-entrenar el modelo y 15 para su test). Concretamente la figura muestra las aproximaciones realizadas para las 15 muestras empleadas para el test en la fase de re-entrenamiento (etapa II), tras la evaluación (etapa VI) de las generaciones 0, 5, 10 y 15. Como puede apreciarse, el modelo ofrecido por la RNA se adapta de forma progresiva a las sucesivas generaciones propuestas por la arquitectura de búsqueda.

Por otra parte, la figura 6.34 muestra los resultados obtenidos al optimizar la función 6.7 mediante la arquitectura propuesta, empleando diferentes tamaños de población (con generaciones de control del mismo tamaño que las propuestas por el AG). En concreto se muestran los valores medios y máximos de la aptitud obtenida por los individuos de cada una de las generaciones propuestas por la arquitectura de búsqueda. Tal y como puede apreciarse, cuanto mayor es el tamaño de la población, más rápida es la convergencia ofrecida por la técnica. Además, la arquitectura es capaz de obtener valores muy próximos al máximo empleando poblaciones a partir de 35 individuos. Sin embargo, cuando se emplean poblaciones menores (por ejemplo de 15 muestras), la arquitectura no es capaz de converger hacia el máximo global, quedándose estancada en máximos locales.

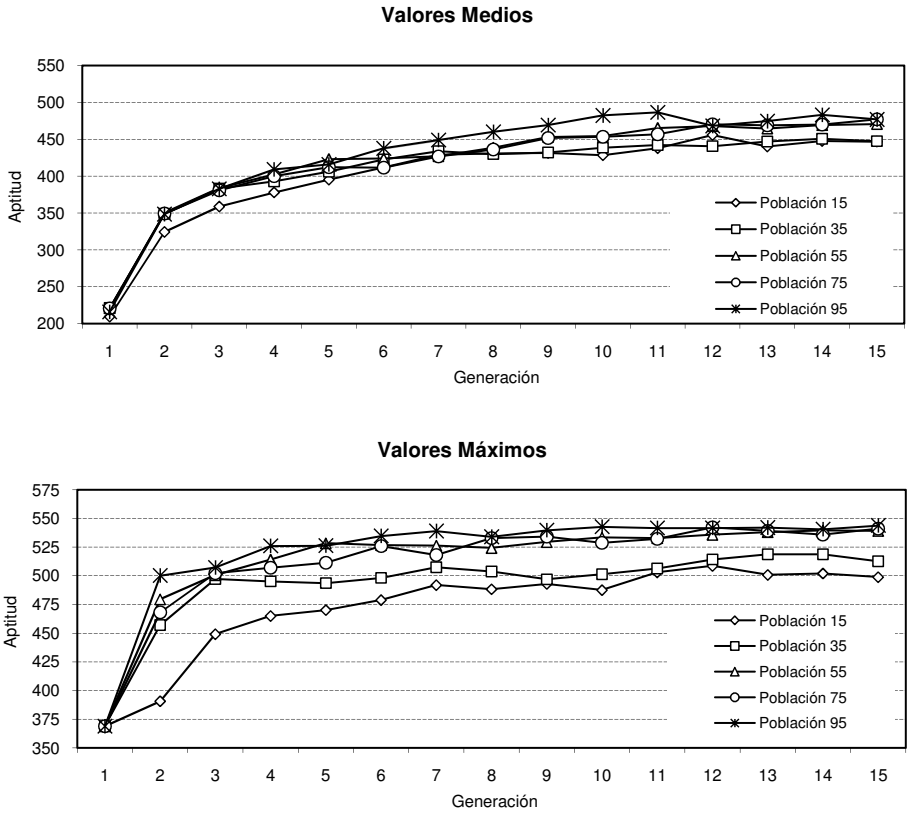

Figura 6.34: Influencia del tamaño de la población. Valores medios y máximos de la aptitud obtenida por los individuos en cada una de las generaciones propuestas por la arquitectura, empleando diferentes tamaños de población (población virtual=población control).

6.1.4.3. Evaluación de la etapa de re-entrenamiento propuesta (Etapa II)

Empleando parte de los datos obtenidos en el apartado anterior, se evaluará el método descrito en el apartado 5.2 de la propuesta para actualizar el modelo de RNA conforme se obtienen más muestras de control. Por tanto se realizará un estudio de los pasos a realizar en la etapa II, sin realizar ningún proceso de optimización.

Es de reseñar que conforme aumentan los ciclos de optimización completados, los individuos propuestos para su evaluación (generación de control) van incrementando su aptitud, pero se disminuye la diversidad de la generación, puesto que la arquitectura de búsqueda va focalizándose en aquellas áreas que resulten más prometedoras. Por tanto, el método buscado para actualizar la RNA debe prevenir un sobre-entrenamiento de la misma, que le impida aproximar de forma adecuada a aquellos individuos que pertenezcan a áreas de búsqueda no demasiado visitadas.

Metodología

Se estudiarán tres métodos diferentes para realizar la actualización del modelo basado en una RNA:

- *Entrenamiento completo.* En cada ciclo de optimización se parte de una RNA sin ningún tipo de conocimiento previo (red vacía). Seguidamente se toman todas las muestras de control disponibles (pertenecientes a todos los ciclos de optimización o generaciones propuestas hasta el momento), creando a partir de ellas dos conjuntos de muestras: uno destinado a entrenar el modelo (80 % de las muestras totales) y otro para su test (20 %). Se entrena la RNA vacía con las muestras destinadas a ello, realizando después una prueba o test con el resto de muestras. Finalmente se actualiza el modelo mediante la nueva RNA obtenida.

- *Re-entrenamiento.* En cada ciclo de optimización se parte de la RNA empleada actualmente como modelo. Se toman las muestras de control obtenidas en el último ciclo de optimización (última generación de control propuesta), creando a partir de ellas dos conjuntos de muestras: uno destinado a re-entrenar el modelo (80 % de las muestas) y otro para su test (20 %). Se re-entrena la actual RNA con las muestras destinadas a ello, realizando después una prueba o test del mismo con el resto de muestras. Finalmente se actualiza el modelo mediante la nueva RNA obtenida.

- *Re-entrenamiento mejorado.* En cada ciclo de optimización se parte de la RNA empleada actualmente como modelo. Se toman las muestras de control obtenidas en el último ciclo de optimización (última generación de control propuesta), creando a partir de ellas dos conjuntos de muestras: uno destinado a re-entrenar el modelo (80 %) y otro para su test (20 %). Se re-entrena la actual RNA con las muestras destinadas a ello, aplicando después el test. Seguidamente, se compara el error cuadrático medio obtenido en el proceso de test por este nuevo modelo, con aquel obtenido por la RNA utilizada hasta el momento al predecir el mismo conjunto de muestras de test. Si la RNA re-entrenada ofrece mejores resultados, reemplaza a la anterior. En caso contrario, el modelo empleado actualmente (sin re-entrenamiento) se mantiene para el siguiente ciclo de optimización.

Los datos a emplear en este estudio pertenecen a la optimización realizada en el apartado anterior para la función 6.7 con una población de control de 55 individuos, tomando

datos de las primeras diez generaciones disponibles. Para cada generación de control disponible se obtendrán tres nuevos modelos, utilizando los tres métodos citados anteriormente. Finalmente, se comparará el error relativo medio cometido por los modelos obtenidos con cada uno de los métodos descritos.

En todos los casos se tendrá como punto de partida una red sin conocimiento previo con cinco nodos de entrada, cuatro nodos en su primera capa oculta, tres nodos en la segunda capa oculta y un nodo de salida (5_4_3_1). Además el algoritmo de entrenamiento a utilizar será el de retro-propagación del error con momento (parametrizado con $\eta = 0.8$ y $\mu = 0.8$). Esta combinación es la que ofreció mejores resultados en el estudio realizado en el apartado anterior para la función 6.7.

Resultados

En la figura 6.35 pueden apreciarse los errores relativos cometidos por las RNA obtenidas mediante los tres métodos estudiados: *entrenamiento completo*, *re-entrenamiento* y *re-entrenamiento mejorado*. En concreto se muestra la media del error relativo cometido por el modelo obtenido empleando cada uno de los métodos al predecir las muestras destinadas al test en cada generación de control.

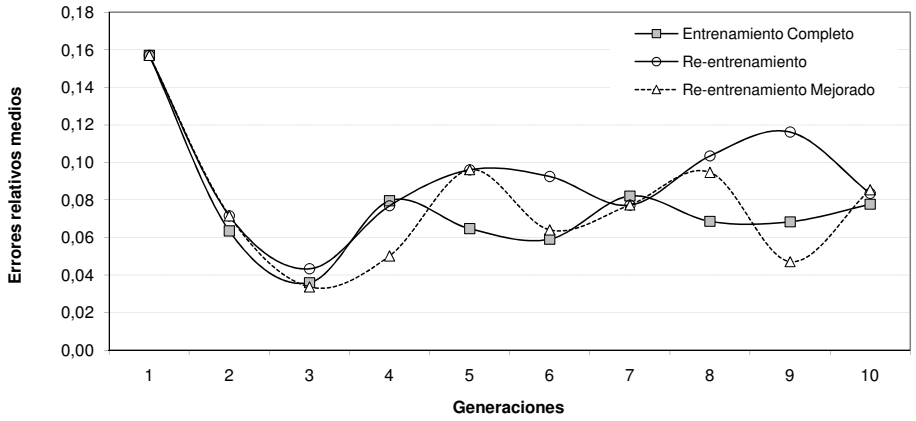

Figura 6.35: Influencia del método de actualización del modelo basado en una RNA en la precisión final del mismo

Puede apreciarse que se obtienen resultados similares al utilizar el método de *entrenamiento completo* y el de *re-entrenamiento mejorado*. El error relativo medio cometido para todas las generaciones en el caso del *entrenamiento completo* es de 0.0756, mientras que el cometido al utilizar *re-entrenamiento mejorado* es de 0.0777. Se ha decidido adoptar este último método para actualizar adecuadamente los modelos por dos moti-

vos. El primer motivo consiste en que conforme aumenta el número de generaciones, se dispone de un mayor número de muestras y es más costoso obtener un nuevo modelo que se entrene partiendo de cero con todas las muestras disponibles, que actualizar el actual modelo utilizando sólo las muestras disponibles para la última generación. El segundo motivo es que mediante el método de *re-entrenamiento mejorado* comprobamos si el modelo re-entrenado obtiene mejores resultados que la RNA de partida, evitando así el sobre-entrenamiento.

6.1.4.4. Optimización de la composición de un catalizador empleando evaluación simulada (Etapas I-VI).

En este apartado se planteará el estudio a realizar para evaluar la utilidad de la etapa V de evaluación simulada (apartado 5.5), que permite trabajar al genético con una población de mayor tamaño (*población virtual*) que la que finalmente es propuesta por la arquitectura tras un ciclo de optimización (*generación de control*), siendo evaluada por la función de aptitud original en la etapa VI (apartado 5.6). Así pues, se planteará un proceso de optimización sobre la función hipotética 6.7, que describe el comportamiento de un catalizador según varía su composición, a fin de determinar la composición que ofrece los mejores resultados catalíticos. En esta optimización se emplearán todas las etapas planteadas en la arquitectura, pero sin emplear generaciones internas (que posibilitan la repetición de la etapa IV en un mismo ciclo de optimización).

Metodología

La arquitectura *Soft Computing* propuesta será parametrizada con los valores que mejores resultados ofrecieron en los estudios realizados con anterioridad: PM=5, Genes=1, $\alpha = 0.9$, progenitores=10 % y generaciones internas=0. Se estudiarán diversas combinaciones para los dos parámetros restantes (población virtual y ratio de reducción), relacionados con la población empleada por el genético y el tamaño de la generación de control que finalmente es propuesta para su evaluación en la etapa VI en un ciclo de optimización. Los diferentes valores que tomarán ambos parámetros se describen en la tabla 6.13, tomando aquellas combinaciones que dan lugar a poblaciones de control de 35, 45 o 55 muestras.

Para cada uno de los tamaños de población virtual a estudiar se obtendrá una generación aleatoria inicial, siguiendo los pasos descritos en 5.1.3. Pero además se forzará a

Población Control	Población Virtual	Ratio Reducción
35	35	0 %
35	44	20 %
35	50	30 %
35	58	40 %
45	45	0 %
45	56	20 %
45	64	30 %
45	75	40 %
55	55	0 %
55	69	20 %
55	79	30 %
55	92	40 %

Cuadro 6.13: Valores estudiados para los parámetros relacionados con la etapa de evaluación simulada de la arquitectura *Soft Computing* propuesta (etapa V)

que la calidad media de cada una de ellas (media de la aptitud de los individuos que la forman) sea similar, de forma que no existan perturbaciones en los resultados finales a causa de la calidad del punto de partida del proceso de búsqueda.

Resumiendo, la arquitectura de búsqueda se parámetrizará con cada una de las combinaciones de valores propuesta, realizando 15 ciclos de optimización (por tanto, obteniendo 15 generaciones de control). Además, cada prueba se repetirá 10 veces, a fin de considerar los resultados medios.

Resultados

En la figura 6.36 pueden apreciarse las aptitudes máximas obtenidas al emplear la arquitectura de búsqueda *Soft Computing* parametrizada con diferentes tamaños de poblaciones virtuales y distintos ratios de reducción (20 %, 30 % y 40 %), para obtener generaciones de control de 35, 45 y 55 muestras. Tal y como puede apreciarse, la utilización de la evaluación simulada (etapa V) tiene efectos posistivos para generaciones de control pequeñas, mientras que conforme aumenta el tamaño de la generación de control, este efecto disminuye.

En concreto, se observa una importante mejora en la rapidez de la convergencia cuando la generación de control posee 35 individuos, que puede explicarse por el hecho de que los operadores del AG (etapa IV) se aplican directamente sobre la población virtual.

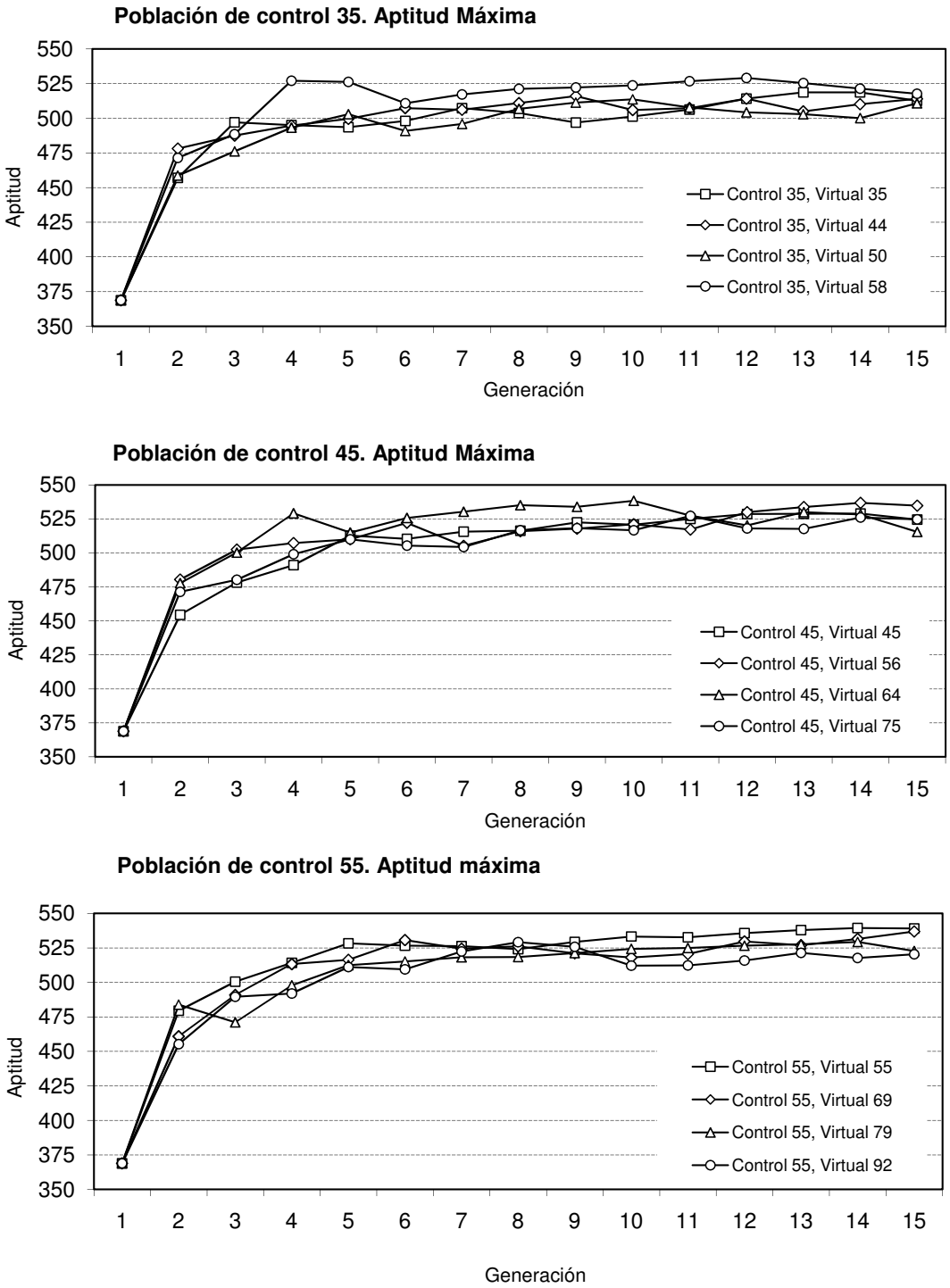

Figura 6.36: Efecto de emplear evaluación simulada. Se muestran las aptitudes máximas obtenidas al utilizar distintos tamaños de poblaciones virtuales sobre los que se aplican ratios de reducción del 20 %, 30 % y 40 %, a fin de obtener generaciones de control de 35, 45 y 55 muestras.

Por tanto, el espacio explorado por el AG aumenta al utilizar una población virtual mayor que la generación de control propuesta en cada ciclo de optimización, mejorando la convergencia hacia áreas prometodoras del espacio de búsqueda. Sin embargo, se observa que no es conveniente emplear ratios de reducción muy grandes (utilizando poblaciones virtuales muy superiores a las de control). En estos casos, el número de individuos cuya aptitud se calcula siempre de forma aproximada (sin contrastarse con la función de aptitud original) es muy elevado, introduciendo demasiado ruido en el sistema de optimización que puede traducirse en un empeoramiento de la convergencia ofrecida.

Por otra parte, en las figuras 6.37 y 6.38 se aprecian los beneficios de emplear evaluación simulada (etapa V) en relación al coste experimental requerido (el número de individuos de control evaluados en la etapa VI).

En concreto, la figura 6.37 muestra los máximos conseguidos por las diferentes estrategias de optimización (en líneas), así como el número de muestras o individuos de control (coste experimental) necesitados en cada caso (en barras), para dos tamaños de población virtuales diferentes (cercanos a 55 y 75 muestras). Cada estrategia de optimización emplea diferentes porcentajes de reducción ($\leq 40\,\%$). Puede apreciarse que la convergencia ofrecida por las distintas estrategias es similar, aunque el coste experimental sea diferente. En ambas pruebas los operadores del AG (etapa IV) trabajan con tamaños de poblaciones virtuales parecidos, por lo que el espacio explorado por ciclo de optimización es similar, diferenciándose en el tamaño de las generaciones que finalmente son propuestas para su evaluación de control (fase VI).

Por ejemplo, la estrategia de optimización que emplea una población virtual de 75 individuos, sin realizar evaluación simulada (ratio de reducción=0 %), consigue una composición de catalizador que ofrece un rendimiento máximo de 509.93 en la quinta generación o ciclo de optimización, requiriendo 375 muestras evaluadas de forma experimental (75 muestras de control por ciclo). Sin embargo, la estrategia que emplea una población virtual de 75 muestras con un ratio de reducción del 40 % obtiene un catalizador con un rendimiento máximo de 509.93 en la quinta generación, pero necesitando sólo 225 muestras de control (45 muestras en cada generación de control). Por tanto, empleando evaluación simulada pueden conseguirse resultados similares, reduciendo considerablemente el número de individuos de control que deben evaluarse con la función de aptitud original (muestras experimentales en este tipo de problemas).

De forma similar, en la figura 6.38 se muestra la evolución de la aptitud o calidad

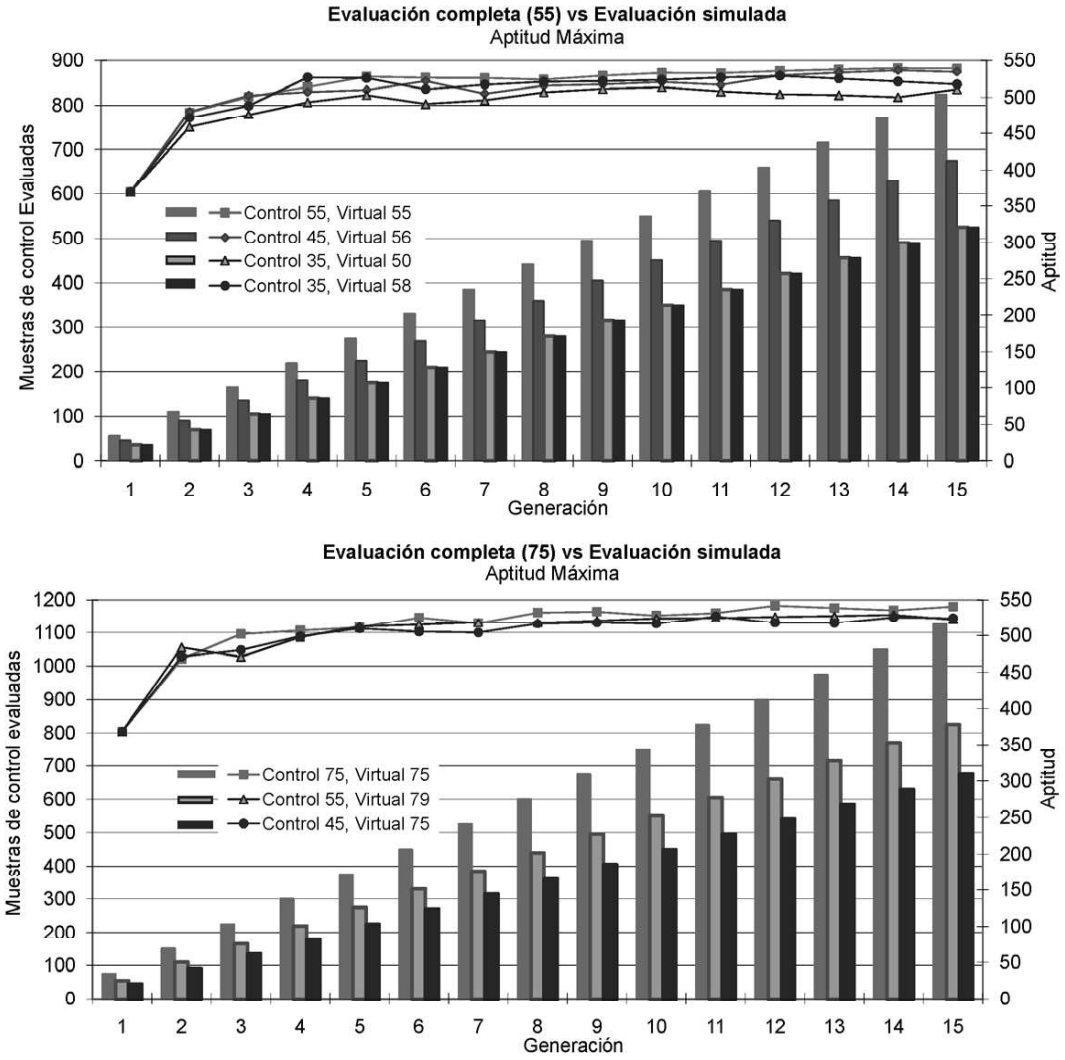

Figura 6.37: Máximos alcanzados y número de muestras de control evaluadas por ciclo de optimización realizado, utilizando poblaciones virtuales alrededor de 55 y 75 muestras. El número acumulado de muestras de control evaluadas con la función original aparece en barras (eje de ordenadas primario). La aptitud máxima alcanzada en cada generación de control se muestra en las líneas (eje de ordenadas secundario).

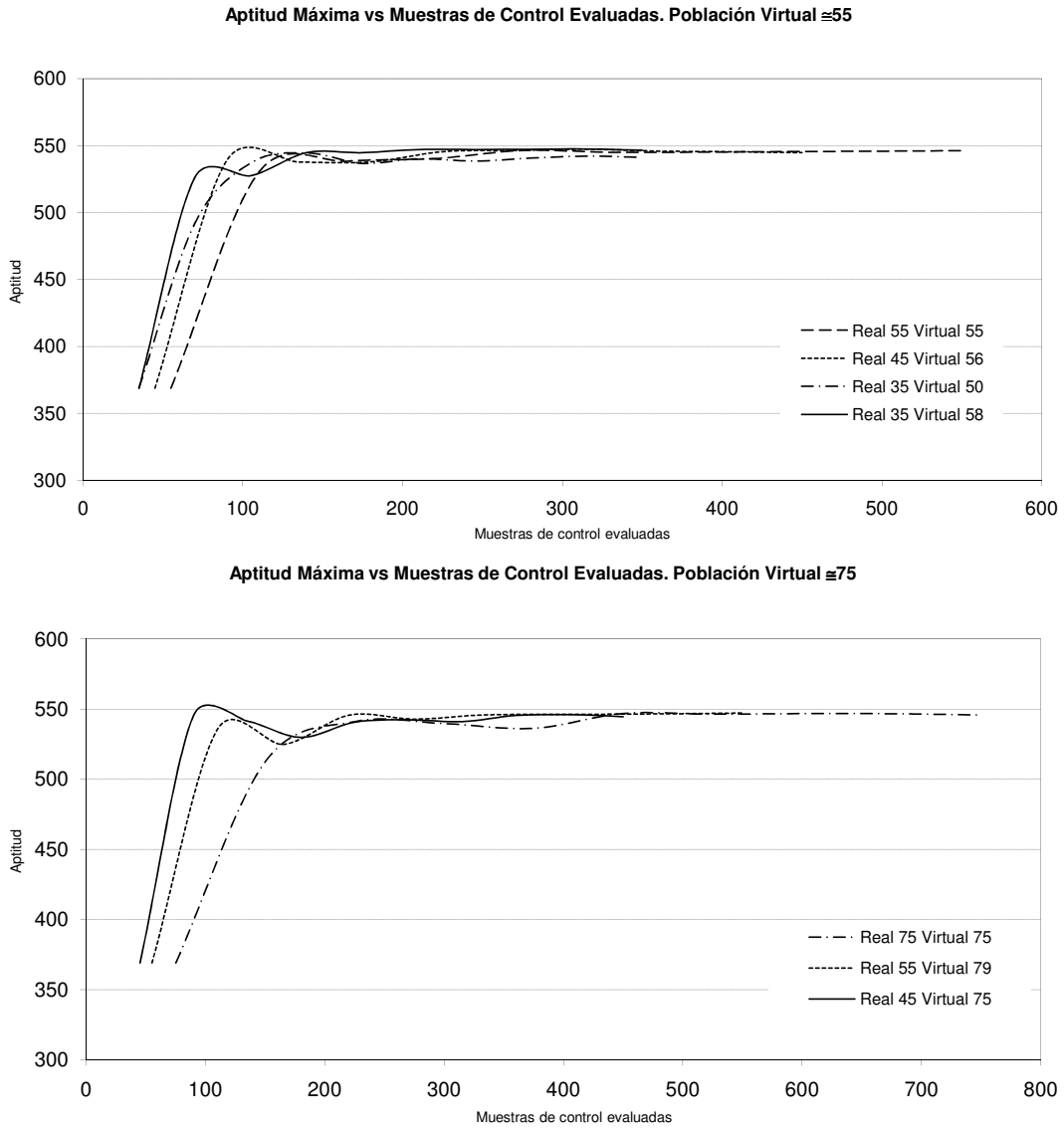

Figura 6.38: Correlación entre los máximos obtenidos y el número de muestras de control evaluadas (coste experimental).

obtenida en relación al número de individuos o muestras de control evaluadas de forma acumulada a lo largo de diez ciclos de optimización. Puede apreciarse que la arquitectura converge más rápidamente al emplear evaluación simulada (etapa V). Sirvan como ejemplo los resultados ofrecidos cuando se emplea una estrategia con una población virtual de 75 individuos y un ratio de reducción del 40 %, dando lugar a una población de control de 45 muestras.

6.1.4.5. Optimización de la composición de un catalizador empleando generaciones internas (Etapas I-VI).

En los apartados anteriores se ha evaluado la utilidad y validez de cada una de las etapas planteadas en la arquitectura propuesta (figura 5.1). Sin embargo, no se ha estudiado la posibilidad de aumentar el número de individuos explorados por el AG por medio de la utilización de las denominadas *generaciones internas* (sección 5.1.2), que permiten aplicar los operadores del AG (etapa IV) en más de una ocasión antes de proponer una generación de control en un mismo ciclo de optimización. Por tanto el número de puntos explorados dentro del dominio de búsqueda aumenta, sin incrementar el número de individuos que son evaluados con la función de aptitud original. Además, tal y como se planteó en la descripción de la arquitectura (capítulo 5), es posible combinar la evaluación simulada con la utilización de generaciones internas, incrementando por tanto el número de individuos que son evaluados exclusivamente empleando la RNA que modele la función de aptitud requerida.

Por todo ello, se planteará un proceso de optimización en el que se utilice enteramente la arquitectura de búsqueda planteada (etapas I-VI), incluyendo el cálculo de algunas *generaciones internas*. La optimización se realizará nuevamente sobre la función hipotética 6.7, a fin de comparar los resultados de convergencia obtenidos en este estudio con los obtenidos anteriormente (secciones 6.1.4.2 y 6.1.4.4). En concreto se comparan los resultados de emplear o no evaluación simulada (etapa V) , con la de combinar ambas estrategias al aplicar generaciones internas (repitiendo la etapa IV).

Metodología

A fin de poder comparar adecuadamente los resultados conseguidos al emplear *generaciones internas* con aquellos obtenidos en los estudios anteriores (secciones 6.1.4.2 y 6.1.4.4), se utilizará el mismo valor usado en esos estudios para los parámetros re-

PM	Genes	α	Progenitores	Población Control	Población Virtual	Ratio Reducción	Generaciones Internas
5	1	0.9	10 %	35	35	0 %	2
5	1	0.9	10 %	35	50	30 %	2
5	1	0.9	10 %	45	45	0 %	2
5	1	0.9	10 %	45	64	30 %	2

Cuadro 6.14: Parámetros de la arquitectura de búsqueda *Soft Computing* utilizados en las pruebas planteadas para evaluar la utilidad de la obtención de *generaciones internas*

lacionados con los operadores del AG: PM=5, Genes=1, $\alpha = 0.9$, Progenitores=10 %. Además, se plantearán diferentes pruebas, variando el tamaño de la población virtual y el ratio de reducción. En concreto se estudiarán diferentes combinaciones de poblaciones virtuales que aplicando un ratio de reducción del 0 % ó 30 % den como resultado poblaciones de control de 35 o 45 individuos. Finalmente, se establecerá en 2 el número de generaciones internas a obtener, incrementando su número si los resultados obtenidos son positivos. La tabla 6.14 recoje de forma resumida los valores que tomarán los diferentes parámetros para las cuatro pruebas planteadas inicialmente. Cada proceso de optimización será repetido en diez ocasiones con cada combinación de parámetros, a fin de considerar los resultados medios conseguidos.

De forma similar, como punto de partida se seleccionarán las mismas generaciones iniciales empleadas en los anteriores estudios (secciones 6.1.4.2 y 6.1.4.4), que garantizaban una buena diversidad y una calidad media similar. Al emplear los mismos puntos de partida, se posibilita que la comparativa entre los resultados obtenidos al utilizar las diferentes estrategias sea más precisa.

Resultados

En la tabla 6.15 se indica el número de muestras de control evaluadas, así como las muestras exploradas tras finalizar diez ciclos de optimización para las diferentes pruebas realizadas. Además se indica el porcentaje de reducción realizado sobre las muestras exploradas por el AG (*Población Virtual*) al obtener cada población de control. Tal y como puede observarse, la utilización de estrategias de optimización que utilicen generaciones internas permiten que el AG explore un mayor número de muestras dentro del espacio de búsqueda, sin incrementar el coste asociado a la evaluación final de las muestras de control en cada ciclo de optimización.

Por otra parte, en las figuras 6.39 y 6.40 se muestra la calidad máxima obtenida por la arquitectura al aplicar las diferentes estrategias de optimización planteadas (mostrando la media de las diez optimizaciones realizadas por estrategia).

En concreto, en la figura 6.39 se pueden comparar los resultados de las diferentes estrategias cuando se obtienen generaciones de control de 35 individuos (que son evaluadas finalmente por la función original). Puede apreciarse que las dos pruebas en las que se empleaba la estrategia de optimización que plantea la obtención de generaciones internas ofrecen peores resultados que el resto. Además, se observa que para el caso en el que combina la evaluación simulada con el cálculo de generaciones internas, a partir de la quinta generación la arquitectura de búsqueda es incapaz de mejorar los resultados de aptitud máximos, desplazándose a áreas menos prometedoras, sin ofrecer una buena progresión hacia la convergencia final.

De forma similar, la figura 6.40 muestra la comparativa de los resultados aportados por las distintas estrategias cuando se obtienen generaciones de control de 45 individuos. Del mismo modo que en el caso anterior, las estrategias que obtienen generaciones internas presentan peores resultados que las otras dos estrategias contempladas. Además, la evolución de la convergencia ofrecida es similar al caso anterior, ofreciendo incluso peores resultados.

Observando de forma conjunta los resultados para las ocho estrategias evaluadas, puede apreciarse que se obtienen peores resultados cuando el porcentaje de reducción obtenido por la estrategia supera el 30 % (tabla 6.15). Por tanto, en esos casos resulta excesivo para el sistema el número de muestras cuya aptitud nunca es contrastada con la función original, de forma que es posible que para muchas de ellas la aptitud esperada sea muy superior a la real. El ruido introducido por éstas hace que la arquitectura se desplace hacia áreas que a priori se muestran más prometedoras de lo que resultan en realidad, como se observa al contrastar la aptitud esperada para los individuos de la generación de control (calculada a partir de las aproximaciones de la RNA) con aquella devuelta por la función de aptitud original.

Por contra, se obtienen mejores resultados cuando se aplica algún tipo de reducción menor al 30 % que cuando el AG sólo trabaja con poblaciones virtuales de igual tamaño que las de control. En estos casos, las muestras cuya aptitud sólo se conoce de forma aproximada (mediante la RNA), pertenecen en su mayoría a zonas poco prometedoras, por lo que el ruido introducido no afecta a la convergencia de la arquitectura de búsqueda.

Estrategia de Optimización			Número de muestras			
Población de Control	Población Virtual	Generaciones Internas	Muestras Control	Muestras Exploradas	Porcentaje Reducción	Aptitud Máxima
35	35	0	385	385	0,00 %	507.48
35	50	0	385	550	30,00 %	513.64
35	35	2	385	735	47,62 %	481.91
35	50	2	385	1050	63,33 %	480.89
45	45	0	495	495	0,00 %	526.06
45	64	0	495	704	29,69 %	538.38
45	45	2	495	945	47,62 %	483.81
45	64	2	495	1344	63,17 %	478.38

Cuadro 6.15: Comparativa de las muestras de control empleadas con respecto a las muestras exploradas tras finalizar diez ciclos de optimización en las diferentes pruebas realizadas. Asimismo se indica el porcentaje de reducción realizado sobre las muestras exploradas por el AG (*Población Virtual*) para obtener cada población de control. En la última columna se muestra la media de la aptitud máxima alcanzada por cada estrategia (para las diez pruebas realizadas por cada una de ellas)

Por tanto, el empleo de generaciones internas debe utilizarse con precaución, siendo recomendable para casos en los que el modelo de aproximación utilizado ofrezca resultados muy precisos. En estos casos, es menos probable que individuos mediocres sean etiquetados como prometedores.

6.1.5. Aplicación a reacciones de interés industrial

La arquitectura de búsqueda propuesta (sección 5) será aplicada a un problema con implicaciones industriales como es la optimización de catalizadores Ti-Silicatos para la epoxidación de oleofinas. Concretamente se estudiarán las composiciones iniciales del gel de materiales mesoporosos que contienen titanio, a fin de optimizar la actividad y selectividad catalítica en la epoxidación del ciclohexeno con tert-butyl hidroperoxido (TBHP) como oxidante. Un catalizador de este tipo puede emplearse en la epoxidación de otras oleofinas, especialmente el propileno. Los propilenos son materias primas empleadas en todo tipo de artículos de consumo, como por ejemplo plásticos o medicinas [Taramasso et al., 1983].

Así pues, se pretenderá optimizar el valor que deben tomar cuatro variables de síntesis (que marcan la composición del gel): pH del gel, surfactante I (CTMA) contenido en el gel, surfactante II (TMA) contenido en el gel y el contenido de titanio. El resto de

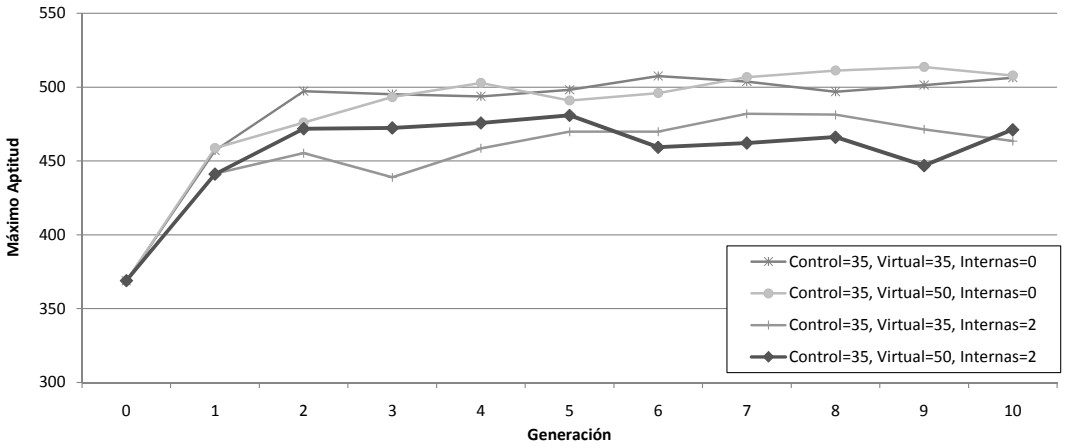

Figura 6.39: Comparativa de los resultados máximos medios obtenidos al utilizar diferentes estrategias de búsqueda en las que se evalúa una población de control de 35 individuos, variando la población virtual explorada por el AG y el número de generaciones internas calculadas antes de proponer una generación de control.

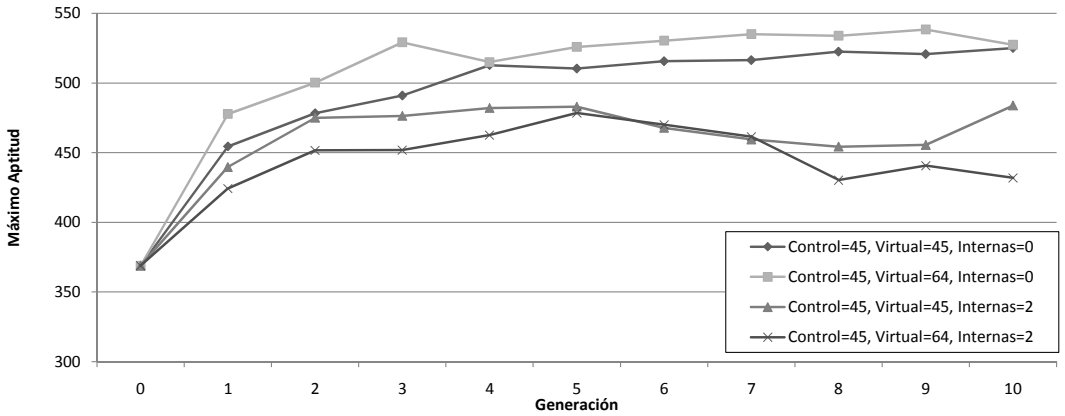

Figura 6.40: Comparativa de los resultados máximos medios conseguidos al utilizar diferentes estrategias de búsqueda en las que se obtiene una población de control de 45 individuos, variando la población virtual explorada por el AG y el número de generaciones internas calculadas antes de proponer una generación de control.

parámetros experimentales estarán fijados a los valores establecidos en estudios previos. Para calcular experimentalmente el *fitness* de cada una de las muestras propuestas por la arquitectura *Soft Computing* se medirá el rendimiento epoxido obtenido a una temperatura de 333K tras cinco horas de tiempo de reacción. Además, en cada generación se estudiarán 37 muestras, puesto que ésta es la capacidad del reactor a emplear.

Metodología

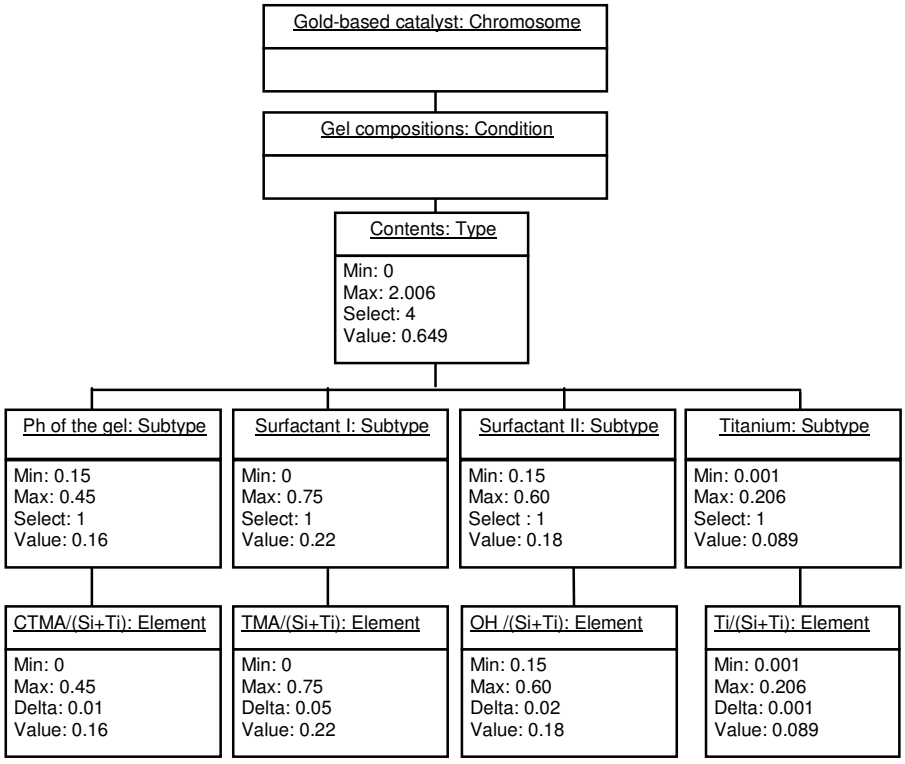

Figura 6.41: Codificación del problema de optimización del catalizador Ti-silicato

La figura 6.41 muestra la codificación utilizada por la técnica *Soft Computing* para optimizar este problema, en el que pueden observarse los rangos en los que cada variable de síntesis será estudiada.

Como punto de partida se calculará una generación inicial, de la que se obtendrán sus resultados catalíticos de forma empírica. Posteriormente, esta generación inicial se utilizará para obtener un perceptrón capaz de aproximar el comportamiento de esta reacción, siguiendo los pasos descritos en la sección 5.1.4. Seguidamente, se empleará el resto de la técnica de optimización *Soft Computing* propuesta para optimizar el valor

de las variables bajo estudio, obteniendo las generaciones que sean necesarias hasta alcanzar un resultado óptimo.

Concretamente, la arquitectura de búsqueda se parametrizará como sigue: generaciones internas=1; población virtual=37; ratio de reducción=0 %; probabilidad de mutación=10 %; número de genes a mutar=1; α=0.7; ratio de progenitores=20 %. Estos parámetros ofrecieron buenos resultados en los estudios realizados en el apartado anterior.

Resultados

El perceptrón multicapa obtenido para modelar los resultados catalíticos posee cuatro nodos en su capa de entrada, dos nodos en su primera capa oculta, un nodo en la segunda capa oculta y dos nodos en su capa de salida. El algoritmo de entrenamiento elegido ha sido el de retro-propagación del error con momento (*Backpropragation with momentum*), con un factor de aprendizaje=0.8 y un término momento=0.5.

El proceso de optimización ha necesitado sólo de tres generaciones, con 37 muestras cada una para alcanzar resultados óptimos. Cada nueva generación mejora considerablemente la actividad y selectividad de su predecesora, así en la figura 6.42 puede observarse dicha mejora, mostrando los resultados del rendimiento obtenido en la epoxidación del ciclohexeno para todas las muestras obtenidas. Además, el mejor catalizador encontrado (perteneciente a la tercera generación) muestra una mejora del 15 % en el rendimiento obtenido (rendimiento epóxido) con respecto al mejor catalizador de la misma naturaleza publicado hasta el momento [Corma et al., 1998], tal y como puede apreciarse en la figura 6.43.

A lo largo de la sección 6.1 se han presentado los diferentes estudios realizados para validar y evaluar la utilidad de cada una de las etapas planteadas en la arquitectura de búsqueda *Soft Computing* propuesta. Para ello se ha empleado esta arquitectura en diferentes problemas de optimización dentro del ámbito de la catálisis combinatoria. Así pues, la técnica propuesta se ha utilizado tanto para optimizar condiciones de distintas reacciones, como para determinar las composiciones idóneas de catalizadores indicados para reacciones de naturaleza y complejidad diferentes. En todos los casos, el

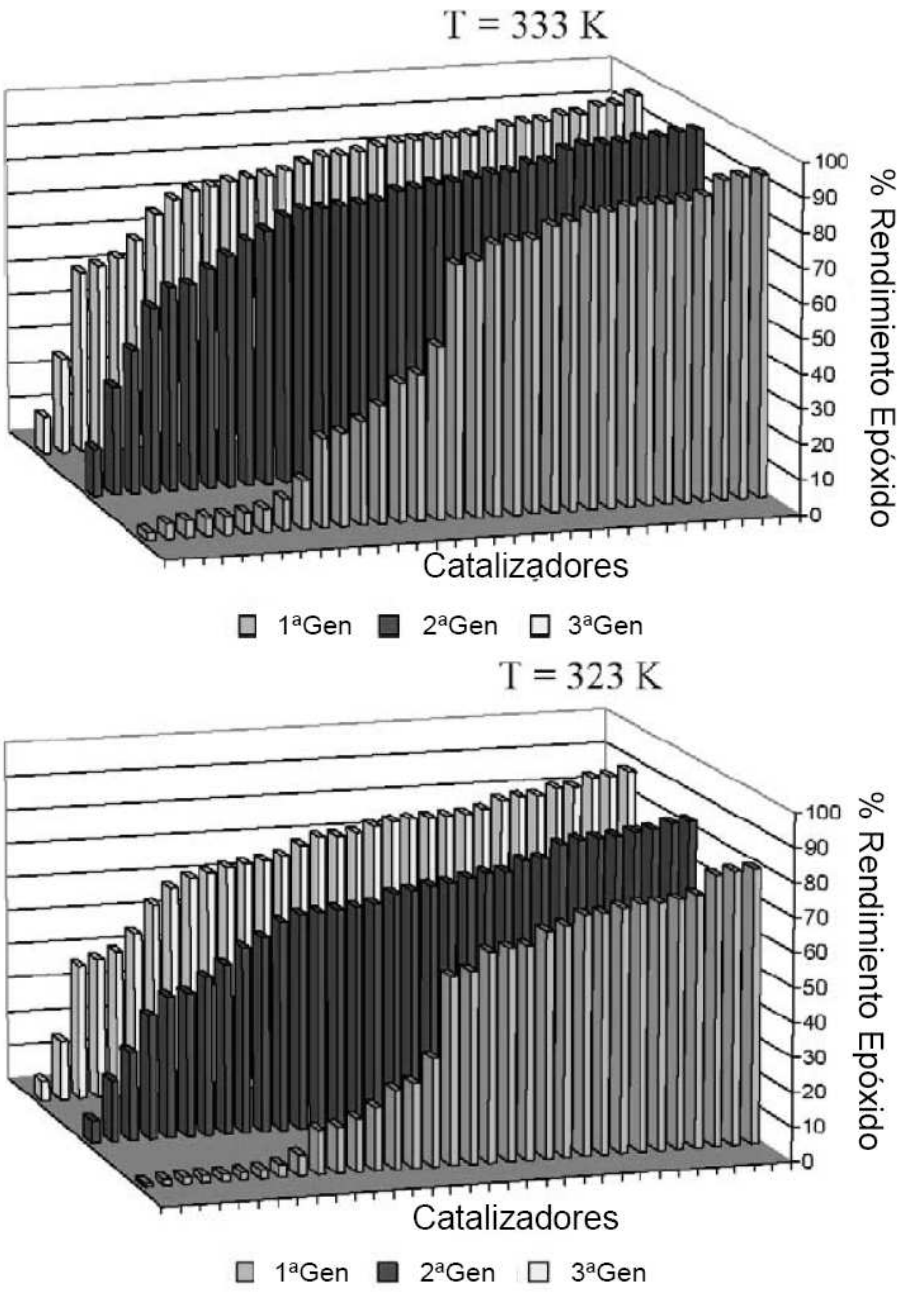

Figura 6.42: Evolución de la aptitud de las muestras del catalizador Ti-silicato para la epoxidación de oleofinas (5 h; 785 mg de cyclohexano, 215 mg de TBHP y 5 mg de catalizador).

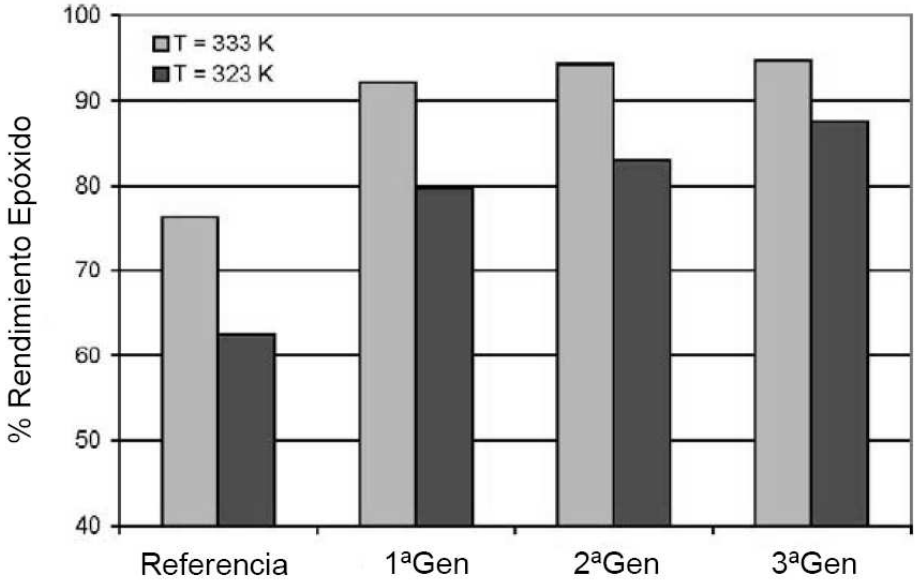

Figura 6.43: Resultados de epoxidación para los mejores catalizadores encontrados en cada generación a dos temperaturas de test diferentes, comparados con los obtenidos por el mejor catalizador conocido

número de individuos que finalmente deben ser evaluados en un reactor (catalizadores a sintetizar o condiciones diferentes a probar), ha sido muy inferior al que se hubiera requerido empleando una técnica de búsqueda tradicional.

Además, se ha mostrado la posibilidad de obtener bibliotecas de modelos basados en RNAs para diferentes reacciones catalíticas, capaces de aproximar adecuadamente los resultados catalíticos de reacciones similares para los que no fueron entrenados inicialmente, requiriendo de un número muy reducido de muestras.

Una vez la utilidad de las etapas planteadas ha quedado contrastada, es necesario probar que la arquitectura es capaz de resolver problemas pertenecientes a dominios diferentes de la catálisis combinatoria. Por este motivo, la arquitectura de búsqueda *Soft Computing* planteada será empleada en el ámbito de los Sistemas de Recomendación, recogiendo los resultados obtenidos en la siguiente sección 6.2.

6.2. Aplicación a los Sistemas de Recomendación

Hoy en día, Internet se ha convertido en un centro comercial mundial, en el que los consumidores pueden buscar todo tipo de servicios y productos, de diversas calidades y prestaciones. Así, los consumidores deben elegir entre esta enorme variedad aquellos productos que mejor se ajusten a sus preferencias y expectativas. Sin embargo, en la práctica esta tarea puede resultar extremadamente compleja debido al gran número de sitios y volumen de información existentes. Por ello, en la actualidad los Sistemas de Recomendación (explicados en el capítulo 4) son empleados principalmente por usuarios finales que desean que se les ayude a elegir qué película ver, qué nuevo disco comprar, qué libro leer, etc. Estos sistemas tratan de mediar, apoyar o automatizar el día a día de los procesos de intercambio de recomendaciones.

Por otra parte, los Sistemas de Recomendación también son empleados por las empresas dedicadas al comercio electrónico para sugerir productos a sus consumidores y para facilitarles aquella información que pueda serles útil para decidir qué productos se adaptan mejor a sus posibilidades, asegurándose así su venta [Schafer et al., 2001]. Para ello, como se explicó en el capítulo 4, puede emplearse información sobre aquellos productos que son más vendidos, basarse en datos demográficos de los clientes, históricos de venta individuales o colectivos, etc.

En este sentido, sería muy interesante para las empresas el poder definir un perfil de las características de los productos a ofertar más deseadas o valoradas por los usuarios, de forma que los productos sean los más atractivos para ellos dentro del abanico de posibilidades existentes y, por tanto, se vendan con facilidad.

Así pues, a partir de un producto o servicio caracterizado por una serie de atributos, interesaría conocer cuáles de sus atributos son los más valorados, imprescindibles o superfluos a la hora de elegir dicho producto por parte del usuario final. Por tanto, se trataría de aprender cuáles son, a grandes rasgos, sus preferencias, a fin de sugerirle o incluso producir productos más acordes a las tendencias actuales. Este problema puede verse como un problema de combinatoria, en el que se persigue conocer qué combinación de pesos de importancia para cada característica hace que un producto resulte ganador.

Además, debe tenerse en consideración el problema de la falta de información para el caso de usuarios nuevos dentro de un sistema, de los que no se tienen datos históricos de los que extraer sus preferencias. Existen aproximaciones previas que emplean diversas técnicas *Soft Computing* para aprender o determinar las preferencias de los usuarios,

pero que sin embargo no tienen en cuenta los problemas que aparecen en momentos iniciales debido a la escasez de información. Por ejemplo, en [Guan et al., 2002] se capturan las preferencias de los consumidores a partir de sus respuestas. Para ello, primero se requiere que los consumidores indiquen qué producto consideran mejor de una lista que se les proporciona. A partir de sus respuestas, el sistema va ajustando los pesos asignados a los diferentes atributos de los ítems estudiados. Otro ejemplo de aplicación es el presentado por [Shibata et al., 2002], donde agentes de recomendación de información personalizada seleccionan y recomiendan contenidos a sus usuarios. Después, estos agentes aprenden a partir de la reacción ofrecida por los usuarios al contenido propuesto. Las preferencias de los usuarios se definen como un conjunto de pesos sobre los atributos estudiados, que indican cómo de importante es el atributo para el usuario. Estos pesos son ajustados mediante aprendizaje por refuerzo.

Básicamente el tipo de problemas planteado consiste en la búsqueda de la combinación de parámetros óptimos para un usuario, en los que puede existir escasez de información. Por tanto, este problema puede abordarse mediante la arquitectura de búsqueda basada en técnicas *Soft Computing* propuesta en este trabajo, comentada en el capítulo 5.

Para probar la aplicabilidad de la técnica propuesta a este dominio, se empleará el conjunto de datos *MovieLens* [Herlocker et al., 1999], utilizado habitualmente como *benchmark* en este ámbito. La base de datos *MovieLens* fue recopilada durante un proyecto de investigación llevado a cabo por el grupo de investigación *GroupLens* de la Universidad de Minesota. En concreto, la información se recopiló a través de la página web del sistema *MovieLens* [2], durante un período de siete meses desde el 19 de septiembre de 1997 hasta el 22 de abril de 1998. Recoge información de unas 10.000 valoraciones realizadas por 943 usuarios sobre 1.682 películas. Como mínimo, por cada usuario se tienen 20 valoraciones para otras tantas películas diferentes. Las valoraciones están basadas en una escala de cinco estrellas, donde una película calificada como excelente recibe cinco estrellas y viceversa. También están disponibles ciertos datos demográficos sobre los usuarios como su edad, género, ocupación, código postal, etc. Las películas están caracterizadas por título, fecha de estreno, así como su género. En concreto, a cada película se le puede asignar hasta 18 géneros diferentes de forma simultánea (acción, aventura, drama, desconocido, etc.). A fin de facilitar su posterior utilización, toda la información recogida en la base de datos *MoviLens* será normalizada, de forma que cada uno de los campos que contiene pertenecerá al intervalo $[0, 1]$. Así, por ejemplo, las valoraciones realizadas por los usuarios irán desde el valor cero

[2] http://movielens.umn.edu

para aquellas películas que originalmente tuvieran una estrella, hasta el valor uno para aquellas películas valoradas como excelentes (cinco estrellas), en incrementos de 0.25.

Así pues, el problema combinatorio que se pretende resolver aplicando la técnica propuesta consiste en determinar las preferencias de un usuario o de un conjunto de ellos sobre los atributos que definen las diferentes películas. En concreto, se trabajará con los atributos del género y fecha de lanzamiento de cada película. A cada uno de estos atributos se le asignará un peso, y por tanto, el problema combinatorio a resolver consistirá en determinar el conjunto de valores óptimo de estos pesos para que reflejen las preferencias de los usuarios. Los pesos obtenidos se emplearán para predecir las valoraciones que realizarían los usuarios sobre un conjunto de test de películas a fin de determinar si realmente reflejan sus preferencias.

En un primer paso, se configurarán adecuadamente los parámetros de la arquitectura planteando un estudio en el que se aplicará la arquitectura de búsqueda propuesta sin emplear modelos de la función de fitness (sin realizar las etapas II, III y V), utilizando directamente la información disponible en la base de datos *Movilens*. En concreto, se utilizarán datos de un usuario del que se dispone de un volumen aceptable de información, a fin de establecer sus preferencias. Seguidamente, se aplicará la arquitectura adecuadamente configurada para determinar las preferencias de grupos de usuarios, empleando sus preferencias para predecir las valoraciones que realizarían usuarios similares a los pertenecientes a los grupos. Nuevamente, la función de aptitud será calculada utilizando la información disponible en la base de datos, sin emplear modelos aproximados. Finalmente, se aplicará la arquitectura por completo, verificando la aplicabilidad de la técnica en sistemas en los que exista un gran volumen de información, por lo que sea necesario la utilización de modelos aproximados de las funciones de aptitud.

6.2.1. Configuración de los parámetros de la arquitectura determinando las preferencias de un usuario.

Para configurar adecuadamente los parámetros de la arquitectura, se ha seleccionado un usuario (identificador 276) del que se dispone de un gran volumen de información (518 valoraciones). Así pues, se utilizarán diferentes combinaciones de valores para los parámetros de la arquitectura para obtener los pesos que definan adecuadamente las preferencias del usuario seleccionado. Para determinar qué combinación de parámetros ofrece mejores resultados, se emplearán las preferencias obtenidas en cada caso para estimar qué valoración daría el usuario a diferentes películas. Estas estimaciones se

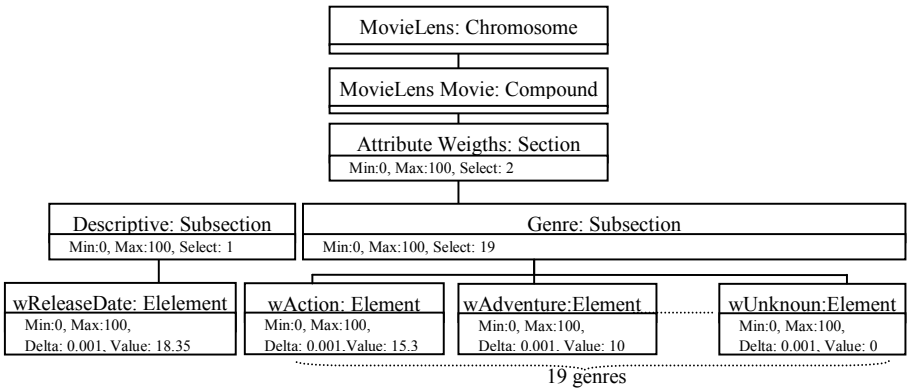

Figura 6.44: Codificación del problema MoviLens. Define el conjunto de pesos sobre los atributos de las películas que describen las preferencias de los usuarios.

compararán con las valoraciones realizada por dichos usuario y se medirá su precisión aplicando diferentes métricas de naturaleza estadística. En concreto se empleará la media del error absoluto (MAE), previamente utilizada para medir el comportamiento de sistemas de recomendación en diversos estudios, como por ejemplo en [Sarwar et al., 1998] y en [Shardnand and Maes, 1995]. También se tendrá en consideración el error cuadrático medio (MSE), usado con este propósito en [Sarwar et al., 1998].

Metodología

Tal y como se ha mencionado previamente, se pretende determinar qué atributos son decisivos para un usuario concreto o un conjunto de ellos a la hora de valorar una película. Para cada una de las películas, se estudiarán 20 atributos diferentes, referidos a características descriptivas de las películas (fecha de lanzamiento), así como al género de las mismas. La figura 6.44 presenta la codificación del problema, donde un cromosoma o posible solución está formado por un único compuesto con una sección. La sección se ha dividido en dos subsecciones, una para agrupar los atributos descriptivos y otra para los relacionados con el género. Finalmente, los elementos o genes a estudiar por el algoritmo genético son los pesos asignados a cada una de las características estudiadas. Por tanto, las preferencias de los usuarios se reflejarán con un conjunto de pesos w_i, donde $\sum_1^{20} W_i = 100$.

Las soluciones propuestas por la arquitectura de búsqueda se utilizarán para estimar las valoraciones que los usuarios realizarían sobre ciertas películas, midiendo la exactitud de las estimaciones realizadas y asignando a cada posible solución una aptitud proporcional

a esa precisión. De este modo, el fitness asignado a cada conjunto de pesos propuestos para los diferentes atributos está relacionado con cómo éstos reflejan las preferencias de los usuarios. Así, la función de fitness o aptitud (F) empleada por el algoritmo genético para este problema se define como:

$$F = \frac{1}{MAE_{Est}} \tag{6.8}$$

$$MAE_{Est} = \frac{\sum_{j=1}^{M} |P_j - R_j|}{M} \tag{6.9}$$

donde M indica el número de valoraciones realizadas por el usuario; P_j representa la estimación o predicción de la valoración del usuario para una película; R_j es la valoración realizada por el usuario para dicha película. Asimismo, la función 6.10 describe la forma en la que se calcula P_j:

$$P_j = \sum_{i=1}^{N} \frac{w_i}{100} q_i \tag{6.10}$$

donde N es el número de atributos estudiados (20 en este estudio), w_i es el peso asignado al atributo i de la película; q_i es el valor del atributo i de la película a valorar y $\sum_{i=1}^{N} w_i = 100$. El valor computado para P_j es discretizado mediante redondeo de forma que se obtiene uno de los cinco valores permitidos para cada valoración: $\{0, 0.25, 0.5, 0.75, 1\}$.

La información disponible (518 valoraciones) del usuario seleccionado para este estudio (id. 276) se ha dividido en datos de entrenamiento (90 %) y test (10 %), teniendo en cuenta para ello el momento temporal en el que se realizó la puntuación. Así, las valoraciones antiguas se incluyen en el corpus de entrenamiento, mientras que las más recientes se emplearán en las fases de test. De esta manera se simula la interacción que un usuario tiene en cualquier Sistema de Recomendación. En concreto, los datos de entrenamiento serán empleados para el cálculo de la función de aptitud (etapa VI) empleada por los operadores del algoritmo genético (etapa IV), descrita por la función 6.8 antes presentada.

En este primer estudio se configurarán los parámetros relacionados con los operadores de cruce y mutación, así como con el tamaño de población explorado por el algoritmo genético (etapa I, sección 5.1.2). En primer lugar se establecerán los mejores valores para el α y ratio de progenitores a emplear por el operador de cruce, así como el

Población Virtual	valor α	Ratio de progenitores
100	0.6	10 %
200	0.7	20 %
300	0.8	30 %
400	0.9	
500		

Cuadro 6.16: Valores utilizados en el estudio realizado para encontrar la mejor combinación para los parámetros relacionados con el operador de cruce y el tamaño de la población explorada.

Prob. Mutación	Genes a mutar
5 %	1
10 %	2
15 %	3

Cuadro 6.17: Valores estudiados en la búsqueda de la mejor combinación para los parámetros relacionados con el operador de mutación.

tamaño de la población virtual a emplear, estudiando todas las posibles combinaciones de los valores indicados en la tabla 6.16. El resto de parámetros se mantendrán fijos, empleando una probabilidad de mutación del 0 % (por lo que no es necesario establecer el número de genes a mutar), un ratio de reducción del 0 % y una generación interna por cada ciclo de optimización. En segundo lugar, se estudiará qué combinación de valores para los parámetros de mutación es más prometedora, estudiando los valores indicados en la tabla 6.17. Para este estudio se tomará la combinación de valores que haya ofrecido mejores resultados en el caso anterior para los parámetros α, ratio de progenitores y tamaño de la población. El resto de parámetros se mantendrán fijos con los mismos valores que en el caso anterior. Finalmente, en ambos estudios para todas las pruebas se empleará como criterio de convergencia la realización de 25 ciclos de optimización, es decir, se propondrán 25 generaciones diferentes.

Para decidir qué combinación de parámetros es más prometedora, se tendrá en cuenta la media del error absoluto (MAE) cometida en las estimaciones realizadas para el usuario en la fase de test, empleando para ello la combinación de pesos (cromosoma) que haya obtenido una mejor aptitud en cada prueba (utilizando la función 6.10 para computar la estimación). También se calculará el error cuadrático medio (MSE) al hacer estas estimaciones, así como el número de valoraciones estimadas correctamente. Se entenderá que una valoración estimada (P_j) para una película ha sido correcta cuando ésta indique que la película será del agrado del usuario y la valoración existente (R_j) para esa película en el corpus de datos de test así lo indique. Del mismo modo,

también se entenderá como un acierto cuando la estimación de la valoración de una película indique que ésta no será del agrado del usuario y la valoración almacenada en la base de datos lo corrobore. La función 6.11 resume la forma en la que se clasifican las valoraciones estimadas en aciertos o fallos.

$$RE = \begin{cases} (P_j \geq 0.5) \wedge (R_j \geq 0.5) & Acierto \\ (P_j < 0.5) \wedge (R_j < 0.5) & Acierto \\ Otro\ caso & Fallo \end{cases} \tag{6.11}$$

Resultados

En la tabla 6.18 se recogen los diez mejores resultados obtenidos en el estudio realizado para configurar los parámetros relacionados con los operadores de cruce, así como con el tamaño de población explorado por el algoritmo genético. La combinación de parámetros seleccionada fue: población virtual= 500, α=0.7 y un 10 % de individuos seleccionados como progenitores. Los pesos obtenidos empleando esta combinación ofrecieron el mayor grado de acierto en las estimaciones de las valoraciones de las diferentes películas presentes en el corpus de test, con un menor error absoluto y cuadrático.

Población Virtual	Valor α	Progenitores	MAE	MSE	Aciertos
500	0.7	10 %	0.2885	0.1250	69.23 %
500	0.6	10 %	0.3029	0.1358	63.46 %
100	0.6	10 %	0.3413	0.1550	50.00 %
400	0.7	30 %	0.3413	0.1550	50.00 %
400	0.9	30 %	0.3413	0.1550	50.00 %
500	0.9	30 %	0.3413	0.1550	50.00 %
400	0.6	20 %	0.3461	0.1538	48.08 %
400	0.6	30 %	0.3461	0.1538	48.08 %
400	0.8	30 %	0.3413	0.1502	48.08 %
400	0.9	10 %	0.3461	0.1635	44.23 %

Cuadro 6.18: Mejores resultados obtenidos en la fase de test durante la optimización de los parámetros de cruce y población.

Los resultados obtenidos en la fase de test al configurar los parámetros relacionados con el operador de mutación pueden apreciarse en la tabla 6.19. Asimismo, en la gráfica mostrada en la figura 6.45 puede observarse la evolución en la aptitud media obtenida para todos los cromosomas (combinaciones de pesos) propuestos por el AG en cada generación durante la fase de entrenamiento, para las cinco pruebas que posteriormente obtuvieron los mejores resultados en la fase de test. Para este estudio se

emplearon los parámetros determinados previamente para el operador de cruce y la población (*población* = 500, $\alpha = 0.7$ y *progenitores* = 10 %). Tal y como puede apreciarse en la tabla 6.19, los mejores resultados se obtuvieron empleando una probabilidad de mutación del 15 % y seleccionando 2 genes para ser mutados en cada ocasión. La combinación de pesos (cromosoma) con mejor aptitud, obtenida empleando estos parámetros, alcanzó una tasa de aciertos del 78.85 % en las estimaciones realizadas para las películas contenidas en el corpus de test.

Resumiendo, tras este estudio la arquitectura de búsqueda fue parametrizada de la siguiente forma: α value = 0.7, ratio de progenitores = 10, probabilidad de mutación=15 %, 2 genes seleccionados para ser mutados, población virtual = 500, ratio de reducción=0 % y una generación interna por cada ciclo de optimización. Empleando estos parámetros, la arquitectura de búsqueda propuso un conjunto de pesos para los diferentes atributos de las películas, que definían las preferencias del usuario 276 y obtenían un porcentaje de aciertos en las estimaciones realizadas del 78.85 % (Figura 6.46). Además, los pesos obtenidos indicaban que los atributos de las películas más valorados por el usuario 276 eran la fecha de lanzamiento de la película (19.32 % de importancia) y los géneros de acción (23.51 %), comedia (19.93 %) y drama (20.73 %). Asismismo, los atributos que resultan prescindibles para el usuario (al alcanzar un valor del 0 % de importancia) son los géneros de animación, musical, misterio, romance, ciencia ficción, guerra, del oeste y desconocido o no catalogable.

Prob. Mutación	Genes a mutar	MAE	MSE	Aciertos
15 %	2	0,2644	0,0925	78, 85 %
15 %	3	0,2645	0,0974	76, 92 %
15 %	1	0,2836	0,1166	69, 23 %
5 %	3	0,2836	0,1166	69, 23 %
10 %	3	0,2788	0,1179	69, 23 %
10 %	2	0,2885	0,1202	69, 23 %
10 %	1	0,2885	0,1250	69, 23 %
5 %	1	0,2885	0,1250	69, 23 %
5 %	2	0,3029	0,1358	63, 46 %

Cuadro 6.19: Mejores resultados obtenidos en la fase de test durante la optimización de los parámetros de mutación.

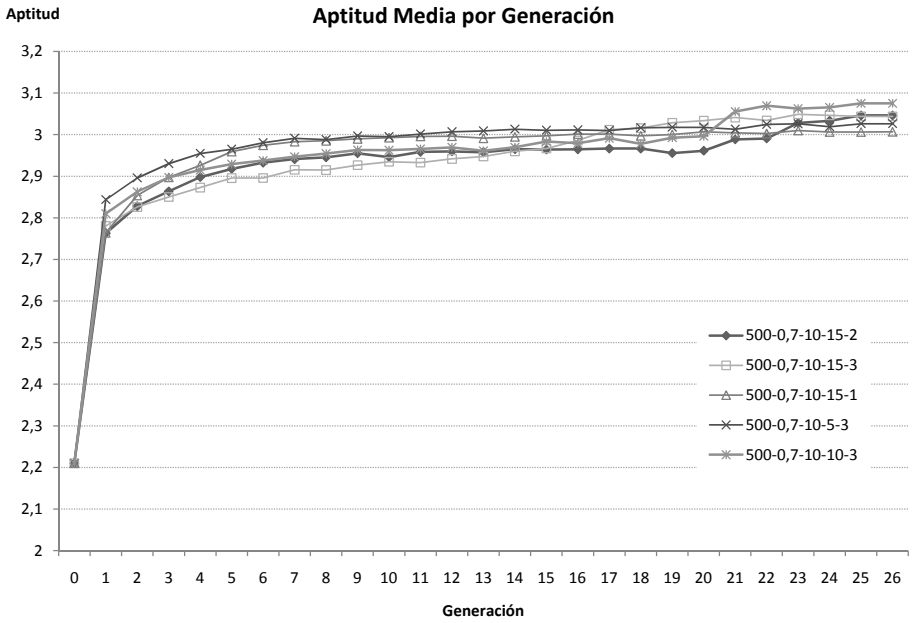

Figura 6.45: Evolución de la aptitud media alcanzada en cada generación propuesta por el AG en la fase de entrenamiento realizada durante la optimización de los parámetros del AG. En concreto se muestran las tendencias obtenidas por las cinco combinaciones de parámetros con mejores resultados. En todos los casos, los parámetros relacionados con el operador de cruce y población estaban establecidos a:*población* = 500, $\alpha = 0.7$ y *progenitores* = 10 %. En el caso de los parámetros relacionados con la mutación, se muestran los resultados de las siguientes combinaciones: PM=15 %, 2 genes; PM=15 %, 3 genes;PM=15 %, 1 gen;PM=5 %, 3 genes; PM=10 %, 3 genes.

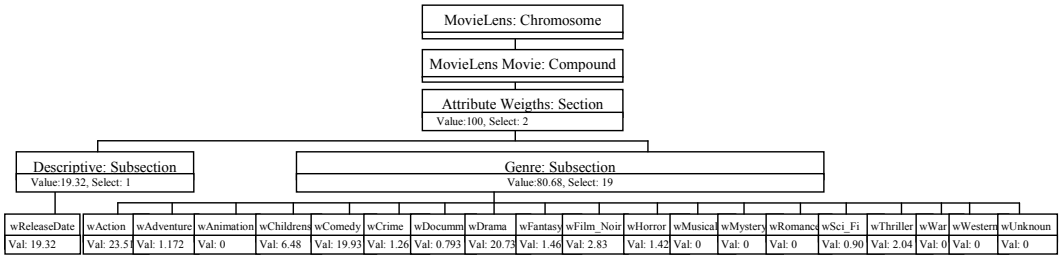

Figura 6.46: Conjunto de pesos para los diferentes atributos de las películas (cromosoma) que define las preferencias del usuario 276 del sistema *MoviLens*. Este cromosoma fue obtenido aplicando la arquitectura de búsqueda parametrizada con: α value = 0.7, ratio de progenitores = 10, probabilidad de mutación=15 %, 2 genes seleccionados para ser mutados, población virtual = 500, ratio de reducción=0 % y una generación interna por cada ciclo de optimización.

6.2.2. Determinación de las preferencias de grupos de usuarios similares identificados mediante correlación de Pearson

En la sección 6.2.1 se ha mostrado cómo la arquitectura de búsqueda era capaz de determinar las preferencias de un usuario acerca del género y fecha de lanzamiento de las películas, utilizando para ello el histórico de valoraciones realizadas por él, almacenadas en el sistema *MoviLens*. En la presente sección se mostrará cómo la arquitectura de búsqueda puede aplicarse a la captura de las preferencias de grupos de usuarios. El objetivo perseguido es poder emplear estas preferencias para predecir si ciertas películas serían o no del agrado de usuarios nuevos, de los que no se posee suficiente información para aprender sus preferencias directamente. Tal y como se comentó previamente en el capítulo 4, este es un problema muy común en los sistemas de recomendación actuales.

Metodología

La arquitectura de búsqueda será parametrizada mediante la combinación de valores que resultó ganadora en el estudio planteado en la anterior sección 6.2.1 : α value = 0.7; ratio de progenitores = 10; probabilidad de mutación=15 %; 2 genes seleccionados para ser mutados; población virtual = 500; ratio de reducción=0 % y una generación interna por cada ciclo de optimización.

Nuevamente el problema a resolver es la asignación del peso de importancia asociado a cada uno de los atributos que definen a una película, coincidiendo con el planteado en la sección previa 6.2.1. Por ello se empleará el mismo sistema de codificación (Figura 6.44). Además, el AG empleará la misma función de *fitness* (Función 6.8) para asignar la correspondiente aptitud a cada conjunto de pesos propuesto (cromosoma). Igualmente, el procedimiento seguido para estimar las valoraciones que realizarían los usuarios a partir del conjunto de pesos determinado también será el mismo (Función 6.10).

Para simular el escenario en el que un usuario nuevo entra en un sistema de recomendación, se seleccionará a un usuario, que denominaremos de referencia, a partir del cual calcularemos el grupo de usuarios considerados similares a él presentes en el sistema. Para seleccionar el grupo adecuado de usuarios similares se utilizará como medida de similitud la correlación de *Pearson* (Función 6.12) y un esquema de vecindad basado en la selección de los k usuarios más próximos [Sarwar et al., 2000].

$$Corr_{ab} = \frac{\sum_i (r_{ai} - \bar{r_a})(r_{bi} - \bar{r_b})}{\sqrt{\sum_i (r_{ai} - \bar{r_a})^2 \sum_i (r_{bi} - \bar{r_b})^2}} \tag{6.12}$$

donde r_{xi} representa la valoración del usuario x sobre el ítem i y $\bar{r_x}$ es la media de las valoraciones realizadas por el usuario x.

En concreto se tendrán en consideración dos usuarios de referencia diferentes. El primero será el usuario con el identificador 276, del que se poseen 518 valoraciones. El segundo fue seleccionado aleatoriamente de entre todos los usuarios disponibles, siendo el usuario con el identificador 710 y sólo 86 valoraciones.

Tal y como se ha mencionado previamente, para cada uno de ellos se obtendrá el grupo o conjunto de usuarios similares a partir de las primeras valoraciones que se tienen de ellos. Por tanto, el grupo de usuarios similares estará formado por aquellos usuarios que hayan valorado de forma parecida las mismas películas que el usuario de referencia. En concreto se realizarán diferentes experimentos con un distinto número de valoraciones de partida y un distinto número de usuarios seleccionados para formar parte de los grupos de usuarios similares. Así, se tendrán en consideración 3 y 5 valoraciones iniciales (M) de los usuarios de referencia, así como 10, 15 o 20 usuarios similares (k). Por tanto, se calcularán un total de 6 conjuntos distintos de usuarios similares para cada usuario de referencia.

La arquitectura de búsqueda se aplicará para cada uno de los grupos de usuarios similares calculados, a fin de obtener el conjunto de pesos de importancia que mejor reflejen las preferencias de esos usuarios (es decir, aquel conjunto con mejor aptitud o *fitness*) acerca de los diferentes atributos que describen las películas. Para ello, de forma similar a lo realizado en la sección anterior, todas las valoraciones disponibles dentro del sistema para esos usuarios serán dividas en dos corpus de datos de entrenamiento (90 %) y test (10 %), teniendo en cuenta para ello el momento temporal en el que se realizó la puntuación. Así, las valoraciones antiguas se incluyen en el corpus de entrenamiento, mientras que las más recientes se emplearán en las fases de test. En concreto, los datos de entrenamiento serán empleados para el cálculo de la función de aptitud (etapa VI) empleada por los operadores del algoritmo genético (etapa IV), descrita por la función 6.8 antes presentada.

Un vez se establezcan las preferencias de los distintos grupos de usuarios similares, éstas se emplearán en primer lugar para estimar las valoraciones que harían estos usuarios sobre las películas presentes en el corpus de test (usando nuevamente la función 6.10). Posteriormente, las preferencias aprendidas serán aplicadas a la estimación de

las valoraciones que realizarían los usuarios de referencia para las últimas 20 películas valoradas por éstos, a fin de contrastar los resultados. Además, las preferencias establecidas también se emplearán para estimar si ciertas películas (siempre las últimas 20 valoradas) son del agrado o no de otros usuarios que también puedan considerarse similares a los usuarios de referencia, pero que sin embargo no fueron seleccionados para formar parte de los grupos antes mencionados.

En todos los casos se calculará tanto la media del error absoluto (MAE), como del error cuadrático medio (MSE) cometido en las estimaciones realizadas. Asimismo, también se computarán el número de valoraciones estimadas correctamente, de forma similar a la empleada en la sección anterior (función 6.11), pero extendida a todos los usuarios seleccionados para formar parte de los grupos o conjuntos. Tal y como describe la función 6.13, una valoración estimada (P_{ij}) para un usuario i perteneciente a un grupo de usuarios similares será considerada correcta cuando ésta indique que la película j será del agrado del usuario y la valoración realizada por ese usuario almacenada en el corpus de datos de test para esa película (R_{ij}) lo corrobore. También se considera un acierto cuando la estimación indica que la película no será del agrado del usuario y la valoración realizada por el usuario así lo indique. En cualquier otro caso, la estimación realizada se considerará un fallo.

$$REG = \begin{cases} (P_{ij} \geq 0.5) \wedge (R_{ij} \geq 0.5) & Acierto \\ (P_{ij} < 0.5) \wedge (R_{ij} < 0.5) & Acierto \\ Otro\ caso & Fallo \end{cases} \tag{6.13}$$

Resultados

Una vez calculados los distintos conjuntos de usuarios similares a los seleccionados como de referencia (Tabla 6.20), pudo constatarse que para el caso del usuario de referencia 710, los conjuntos formados tomando 3 valoraciones iniciales tenían los mismos miembros que aquellos formados a partir de 5 valoraciones. Por tanto, para este usuario sólo se han empleado 3 conjuntos o grupos de usuarios para aprender sus preferencias (tomando los 10, 15 o 20 individuos más parecidos).

En la tabla 6.21 se recogen los resultados obtenidos en la fase de test realizada en el proceso de captura de las preferencias de los distintos grupos de usuarios estudiados. En concreto, se pueden observar los errores absolutos (MAE), cuadráticos (MSE), así como la tasa de aciertos conseguida al estimar las valoraciones que realizarán los

	Grupos Usuario 276		Grupo Usuario 710
NºOrden	3M	5M	3M/5M
1	42	42	405
2	903	943	399
3	943	754	194
4	533	115	224
5	754	614	222
6	525	580	719
7	891	500	575
8	363	264	618
9	833	342	346
10	614	382	627
11	101	917	62
12	500	498	885
13	251	1	326
14	264	425	766
15	342	432	316
16	919	394	886
17	382	188	167
18	1	906	715
19	54	53	533
20	314	895	429

Cuadro 6.20: Identificadores de los 20 primeros usuarios similares a los de referencia que forman parte de los grupos obtenidos teniendo en cuenta las 3 ó 5 primeras valoraciones (M) realizadas por los usuarios de referencia, ordenados de mayor a menor similitud.

Grupo	MAE	MSE	Aciertos %
276-3M-10k	0.3064	0.1320	59.40
276-3M-15k	0.2908	0.1278	59.90
276-3M-20k	0.2970	0.1313	55.40
276-5M-10k	0.2950	0.1380	63.96
276-5M-15k	0.3063	0.1398	59.34
276-5M-20k	0.2930	0.1275	65.16
710-3M/5M-10k	0.2315	0.0924	61.09
710-3M/5M-15k	0.2455	0.1048	51.80
710-3M/5M-20k	0.2567	0.1112	57.68

Cuadro 6.21: Errores absolutos (MAE), cuadráticos (MSE) y tasas de aciertos obtenidas para cada unos de los grupos de usuarios similares estudiados (userId-xM-xxk) al estimar las valoraciones que estos usuarios realizarían sobre las películas presentes en los corpus de test.

usuarios pertenecientes a cada conjunto sobre las películas presentes en los corpus de test (10 % del total disponible en el sistema). Para ello, para cada grupo se seleccionó el conjunto de pesos más prometedor para los diferentes atributos estudiados (cromosoma con mejor aptitud), empleando este conjunto para realizar las estimaciones (mediante al función 6.10). Cada grupo se ha denominado empleando en primer término el usuario de referencia utilizado, seguidamente el número de valoraciones de partida consideradas (realizadas por el citado usuario) y finalmente el número de usuarios seleccionados para formar parte del grupo (userId-xM-xxk).

Posteriormente, las preferencias aprendidas para los grupos de usuarios se emplearon para estimar las valoraciones de las 20 últimas películas puntuadas por los dos usuarios de referencia utilizados. Así, en la tabla 6.22 pueden apreciarse los errores absolutos (MAE), cuadráticos (MSE), así como la tasa de aciertos obtenida en cada caso.

En concreto, en la tabla 6.22 puede apreciarse que los mejores resultados en la estimación de las valoraciones para el usuario de referencia 276 se obtuvieron mediante las preferencias aprendidas para el grupo de usuarios formado considerando 5 valoraciones de partida y seleccionando a los 20 usuarios más cercanos (276-5M-20k). En este caso, el conjunto de pesos que definían las preferencias de este grupo indicaba que los atributos esenciales para ellos eran la fecha de lanzamiento de la película (20 % de importancia) y los géneros de acción (18.46 %), comedia (18.52 %) y drama (27.65 %). Asimismo, los atributos que no tienen ninguna influencia (0 %) en las preferencias de estos usuarios son los géneros de animación, infantil, crimen, documental, fantasia, negro, musical, misterio, guerra y del oeste.

Con respecto al usuario de referencia 710, la tabla 6.22 muestra que los resultados de

Grupo	Usu. Ref.	MAE	MSE	Aciertos %
276-3M-10k	276	0.2625	0.0844	75.00
276-3M-15k	276	0.2550	0.0750	75.00
276-3M-20k	276	0.2125	0.0719	75.00
276-5M-10k	276	0.2875	0.1094	65.00
276-5M-15k	276	0.2250	0.0750	75.00
276-5M-20k	276	0.2125	0.0719	80.00
710-3M/5M-10k	710	0.2750	0.1063	70.00
710-3M/5M-15k	710	0.2875	0.1156	70.00
710-3M/5M-20k	710	0.2750	0.1063	70.00

Cuadro 6.22: Errores absolutos (MAE), cuadráticos (MSE) y tasas de aciertos obtenidas al emplear las preferencias aprendidas para cada unos de los grupos de usuarios estudiados (userId-xM-xxk) al estimar las valoraciones que los usuarios de referencia realizarían sobre las 20 últimas películas puntuadas por ellos.

estimación son muy similares para todos los grupos de usuarios estudiados. Por ejemplo, podemos seleccionar las preferencias aprendidas del grupo de 20 usuarios similares formado a partir de las 5 o 3 primeras valoraciones realizadas por el usuario 710 (710-3M/5M-20k). En este caso, el conjunto de pesos obtenido para los diferentes atributos de las películas indican que a estos usuarios les interesan la fecha de lanzamiento de las películas (19.83 %), así como los géneros de acción (25.42 %) y drama (30.46 %). Por otra parte, los atributos que carecen de interés para estos usuarios corresponden a los géneros del oeste (0 %), no catalogable o desconocido (0 %), horror (0.072 %) y guerra (0.29 %).

Finalmente, los mismos conjuntos de pesos que establecían las preferencias de los diferentes grupos de usuarios similares considerados fueron posteriormente utilizados para estimar las valoraciones que realizarían otros usuarios que también fueran similares a los usuarios de referencia, pero no formaran parte de los diferentes grupos estudiados (en concreto, los cinco siguientes a los k seleccionados en cada caso). Así, para cada uno de estos usuarios similares se estimaron sus puntuaciones sobre las últimas 20 películas valoradas por ellos en el sistema. En las tablas 6.23, 6.24 y 6.25 se recogen los resultados obtenidos, mostrando nuevamente los errores absolutos (MAE), cuadráticos (MSE), así como la tasa de aciertos obtenida en cada prueba. Para el caso de los usuarios parecidos al usuario de referencia 276, en la mayoría de las ocasiones se alcanzan unas tasas de acierto aceptables (superiores o iguales al 75 %), obteniendo en alguna ocasión tasas del 90 % y 100 %. Los resultados obtenidos para los usuarios parecidos al usuario de referencia 710 son algo inferiores, obteniendo tasas de acierto iguales o superiores al 60 %, consiguiendo como máximo un 80 % de aciertos.

Grupos construidos a partir de 3 valoraciones del usuario 276				
Grupo	Usu. Sim.	MAE	MSE	Aciertos %
276-3M-10k	101	0.1875	0.0594	65
276-3M-10k	251	0.3500	0.1687	80
276-3M-10k	264	0.475	0.2812	55
276-3M-10k	342	0.3125	0.1594	75
276-3M-10k	500	0.1500	0.0375	75
276-3M-15k	1	0.3625	0.2031	75
276-3M-15k	54	0.2000	0.0750	90
276-3M-15k	314	0.2625	0.1156	75
276-3M-15k	382	0.2375	0.0906	75
276-3M-15k	919	0.1875	0.0656	80
276-3M-20k	58	0.3250	0.1312	40
276-3M-20k	157	0.3250	0.1437	75
276-3M-20k	276	0.2125	0.0719	75
276-3M-20k	403	0.3750	0.1812	75
276-3M-20k	864	0.3375	0.1531	65

Cuadro 6.23: Errores absolutos (MAE), cuadráticos (MSE) y tasas de aciertos obtenidas al emplear las preferencias aprendidas para cada unos de los grupos de usuarios similares construidos a partir de 3 valoraciones del usuario 276 al estimar las valoraciones que usuarios parecidos a él realizarían sobre las 20 últimas películas puntuadas por ellos.

Grupos construidos a partir de 5 valoraciones del usuario 276				
Grupo	Usu. Sim.	MAE	MSE	Aciertos %
276-5M-10k	1	0.4375	0.2469	50
276-5M-10k	382	0.2750	0.1187	60
276-5M-10k	425	0.3250	0.1500	35
276-5M-10k	498	0.3000	0.1500	60
276-5M-10k	917	0.3625	0.1719	65
276-5M-15k	53	0.3375	0.1531	75
276-5M-15k	188	0.3125	0.1594	60
276-5M-15k	276	0.2250	0.0750	75
276-5M-15k	394	0.3875	0.1781	30
276-5M-15k	906	0.2250	0.0688	100
276-5M-20k	5	0.3750	0.1875	40
276-5M-20k	58	0.3250	0.1250	40
276-5M-20k	236	0.3000	0.1188	50
276-5M-20k	879	0.2875	0.0969	90
276-5M-20k	895	0.2375	0.1031	90

Cuadro 6.24: Errores absolutos (MAE), cuadráticos (MSE) y tasas de aciertos obtenidas al emplear las preferencias aprendidas para cada unos de los grupos de usuarios similares construidos a partir de 5 valoraciones del usuario 276 al estimar las valoraciones que usuarios parecidos a él realizarían sobre las 20 últimas películas puntuadas por ellos.

Grupos construidos a partir de 3 ó 5 valoraciones del usuario 710				
Grupo	Usu. Sim.	MAE	MSE	Aciertos %
710-3M/5M-10k	62	0.3125	0.1281	80
710-3M/5M-10k	316	0.3375	0.1594	60
710-3M/5M-10k	326	0.3500	0.1750	50
710-3M/5M-10k	766	0.2375	0.1094	60
710-3M/5M-10k	885	0.2750	0.1187	55
710-3M/5M-15k	167	0.4125	0.2156	35
710-3M/5M-15k	429	0.2875	0.1344	50
710-3M/5M-15k	886	0.2625	0.1094	60
710-3M/5M-15k	533	0.2500	0.0937	75
710-3M/5M-15k	715	0.2750	0.1062	45
710-3M/5M-20k	11	0.2500	0.0937	60
710-3M/5M-20k	57	0.4000	0.2032	55
710-3M/5M-20k	236	0.3125	0.1219	50
710-3M/5M-20k	479	0.3500	0.1812	70
710-3M/5M-20k	492	0.2250	0.0937	70

Cuadro 6.25: Errores absolutos (MAE), cuadráticos (MSE) y tasas de aciertos obtenidas al emplear las preferencias aprendidas para cada uno de los grupos de usuarios similares construidos a partir de valoraciones del usuario 710 al estimar las valoraciones que usuarios parecidos a él realizarían sobre las 20 últimas películas puntuadas por ellos.

6.2.3. Determinación de las preferencias de grupos de usuarios similares empleando modelos aproximados de la función de aptitud

A continuación se presentará el estudio realizado para validar la arquitectura de búsqueda propuesta para los casos en que sea necesario disponer de modelos aproximados para obtener la función de aptitud dentro del ámbito de los sistemas de recomendación, por ejemplo, debido a la presencia de un gran volumen de información que retarde en exceso el cálculo de esta función.

En concreto, se empleará el histórico de valoraciones disponibles en la base de datos *Movilens* realizadas por un grupo de usuarios para obtener un modelo que sea capaz de aproximar futuras valoraciones. Este modelo será utilizado posteriormente por la arquitectura de búsqueda, para aprender las preferencias de uno de los grupos de usuarios calculados en la sección previa 6.2.2.

Nodos 1ªcapa	5, 10, 15,..., 30, 35, 40
Nodos 2ªcapa	0, 5, 10,..., 30, 35, 40
Algoritmos Entrenamiento	Backpropagation Backpropagation Momentum
Parámetro α	0.2, 0.3,.., 0.8, 0.9
Parámetro τ	0.2, 0.3,.., 0.8, 0.9

Cuadro 6.26: Valores estudiados para los diferentes aspectos considerados en la selección de una topología de perceptrón y función de entrenamiento adecuados

Metodología

Obtención del modelo aproximado de la función de aptitud. Se seguirá el proceso descrito en la sección 5.1.4 de la propuesta para obtener un modelo que aproxime futuras valoraciones a realizar por usuarios similares. Así pues, se probarán con diferentes combinaciones de topologías de perceptrones multicapa y funciones de entrenamiento, parametrizadas con distintos valores. La tabla 6.26 recoge los posibles valores para cada uno de los aspectos a estudiar: nodos en la primera capa oculta, nodos presentes en la segunda capa oculta, algoritmos de entrenamiento y posibles valores para los parámetros (factor de aprendizaje α y momento μ) empleados por éstos. En todos los casos se emplearán funciones de activación tangenciales para los nodos presentes en las capas ocultas, pues los resultados obtenidos en los estudios realizados previamente (sección 6.1.2) sugerían que era más conveniente el empleo de este tipo de funciones de activación, puesto que reducían el número de muestras de entrenamiento necesarias para ofrecer resultados aceptables. Además se utilizarán 20 nodos en la capa de entrada, correspondientes al número de características estudiadas que definen a las películas dentro la base de datos *MoviLens*. Asimismo, el modelo a obtener tendrá una única salida, correspondiente a la aproximación de la valoración de una determinada película.

Por otra parte, los datos a emplear para entrenar (90 %) y probar (10 %) los diferentes modelos se tomarán de las valoraciones recogidas en la base de datos *MovieLens* para uno de los grupo de usuarios similares al usuario de referencia 276 calculados en el apartado anterior (6.2.2). En concreto se tomará el grupo formado por los 20 usuarios más parecidos al usuario 276 al considerar las 5 primeras valoraciones realizadas por él (276-5M-20k), utilizando como medida de similitud la correlación de *Pearson* (Función 6.12) y un esquema de vecindad basado en la selección de los k usuarios más próximos [Sarwar et al., 2000].

Determinación de las preferencias. La arquitectura de búsqueda será aplicada para definir el peso de importancia asociado a cada uno de los atributos que definen a una película, a fin de construir un perfil de las preferencias del grupo de usuarios 276-5M-20k, considerados similares al usuario de referencia 276 (también empleado para obtener el modelo aproximado en el punto anterior). En esta ocasión, la arquitectura utilizará en su operador de cruce el modelo obtenido en el punto anterior, a fin de calcular un valor aproximado de la función de aptitud. Por tanto, en este apartado se emplearán todas las fases descritas en la propuesta (capítulo 5). La arquitectura será parametrizada con los valores que mejores resultados ofrecieron en el estudio de configuración presentado en la sección 6.2.1, añadiendo en esta ocasión cierto grado de reducción: α value = 0.7; ratio de progenitores = 10; probabilidad de mutación=15 %; 2 genes seleccionados para ser mutados; población virtual = 500; ratio de reducción=30 % y una generación interna por cada ciclo de optimización.

Nuevamente la codificación a emplear será la mostrada en la Figura 6.44, usada en los estudios planteados en las secciones anteriores 6.2.1 y 6.2.2. Asimismo, el procedimiento seguido para estimar las valoraciones que realizarían los usuarios empleando el perfil de pesos de importancia establecidos seguirá siendo el mismo que el utilizado previamente, descrito por la función 6.10. Sin embargo, en esta ocasión el AG emplea un calculo aproximado de la función de aptitud, tal y como muestra la función 6.14. En esta función M indica el número de valoraciones realizadas por el usuario; P_j representa la estimación o predicción de la valoración de un usuario el grupo para una película; Rx_j es la valoración que el modelo obtenido estima que realizaría ese usuario para dicha película. La forma en la que se calcula P_j viene descrita por la función 6.10, presentada y utilizada en los estudios anteriores. Por tanto, en esta función se utilizan las estimaciones que ofrece el modelo aprendido en el punto anterior, en lugar del histórico de valoraciones presentes en la base de datos *MovieLens*.

$$F = \frac{1}{MAE_{Aprox}} \tag{6.14}$$

$$MAE_{Aprox} = \frac{\sum_{j=1}^{M} |P_j - Rx_j|}{M} \tag{6.15}$$

La información disponible en la base de datos *MovieLens* de los usuarios que

forman el grupo 276-5M-20k será dividida en un corpus de entrenamiento (90 %) y otro de test (10 %), de forma que la información más reciente se emplee en la fase de test. Durante la fase de test, el conjunto de pesos de importancia más prometedor para los diferentes atributos estudiados (cromosoma con mejor aptitud), será empleado para estimar las valoraciones que harían los usuarios del grupo 276-5M-20k sobre las películas presentes en el corpus de test (usando nuevamente la función 6.10). Seguidamente, el perfil de las preferencias seleccionado será aplicado a la estimación de las valoraciones que realizaría el usuario de referencia 276 para las últimas 20 películas valoradas por éstos. Además, las preferencias establecidas también se utilizarán para estimar si ciertas películas (siempre las últimas 20 valoradas) son del agrado o no de otros usuarios que también se consieren similares al usuario 276, pero que no fueron seleccionados para formar parte del grupo 276-5M-20k.

En todos los casos, las métricas a emplear serán la media del error absoluto (MAE) y el error cuadrático medio (MSE) cometido en las estimaciones realizadas. Asimismo, también se computarán el número de valoraciones estimadas correctamente, mediante la misma función utilizada en la sección anterior (Función 6.13).

Resultados

Obtención del modelo aproximado de la función de aptitud. En la tabla 6.27 se muestra una selección con los mejores resultados obtenidos para las diferentes topologías estudiadas siendo entrenadas con el algoritmo de retro-propagación del error con (Backpropagation Momentum) y sin momento (Backpropagation).

Tal y como puede apreciarse, el error cuadrático medio alcanzado en la fase de test es muy similar en todos los casos mostrados. Sin embargo, los mejores resultados se obtuvieron con la topología de perceptrón formada por 20 nodos en su capa de entrada, 5 nodos en la primera capa oculta, 10 nodos en la segunda capa oculta y un sólo nodo de salida, al ser entrenada con el algoritmo de retro-propagación del error con momento, parametrizado con un factor de aprendizaje de 0.6 y un momento de 0.3.

Determinación de las preferencias. En la tabla 6.28 pueden observarse los resultados obtenidos durante la fase de test, tras la obtención por parte de la arquitectura de búsqueda de un conjunto de pesos que definen las preferencias del grupo de

Backpropagation			Backpropagation Momentum			
Topología	α	MSE	Topología	α	μ	MSE
20_5_0_1	0.2	0.07694100	20_5_15_1	0.2	0.2	0.07482306
20_5_30_1	0.3	0.07523431	20_30_0_1	0.3	0.8	0.07488376
20_20_5_1	0.4	0.07512635	20_10_15_1	0.4	0.3	0.07483873
20_5_40_1	0.5	0.07537580	20_15_10_1	0.5	0.2	0.07521048
20_35_0_1	0.6	0.07566319	**20_5_10_1**	**0.6**	**0.3**	**0.07441431**
20_25_20_1	0.7	0.07535036	20_25_5_1	0.7	0.2	0.07697986
20_5_5_1	0.8	0.07513793	20_10_0_1	0.8	0.4	0.07561837

Cuadro 6.27: Selección de los mejores resultados obtenidos durante el estudio realizado para la obtención de un modelo capaz de ofrecer aproximaciones de valoraciones. Se muestra el error cuadrático medio (MSE) obtenido en cada caso.

usuarios 276-5M-20k, empleando para ello valores aproximados de la función de aptitud o fitness. En concreto se muestran los errores absolutos (MAE), cuadráticos (MSE) y tasas de aciertos obtenidas al emplear el mejor conjunto de pesos encontrado (cromosoma con mejor aptitud) para estimar las valoraciones que los usuarios pertenecientes al grupo 276-5M-20k realizarían sobre las películas presentes en el corpus de test. Además, se muestran los resultados conseguidos al aplicar las mismas preferencias para aproximar las valoraciones de 20 últimas películas puntuadas por el usuario de referencia 276 y otros cinco usuarios similares a él.

Tal y como puede apreciarse en la tabla 6.28, los resultados son inferiores a los obtenidos en el estudio anterior (tablas 6.21, 6.22 y 6.24). En concreto, al emplear valores aproximados de la función de aptitud se obtiene una tasa de aciertos del 55.66 % al predecir las valoraciones que el grupo 276-5M-20k realizaría sobre las películas presentes en el corpus de test, mientras que al emplear los valores exactos de la función de aptitud se obtenía un 65.16 %. De forma similar, al aplicar las preferencias aprendidas empleando valores aproximados de la función de fitness para estimar las valoraciones que realizaría el usuario 276, se obtiene una tasa de aciertos del 65 %, mientras que en los estudios anteriores esa tasa de aciertos era del 80 %.

Por tanto, la introducción del modelo aproximado merma la calidad del conjunto de pesos de importancia obtenido, siendo menos preciso al reflejar las preferencias de los usuarios. Sin embargo, pueden existir ocasiones en la que emplear modelos que aproximen la función de aptitud sea necesario pese a los inconvenientes, por ejemplo debido a que su cálculo exacto requiera de un tiempo de computación excesivo. Cabe reseñar que en este experimento la aptitud calculada para un 30 %

de la población estudiada no ha sido contrastada con la función de fitness original, debido al porcentaje de reducción establecido. Tal y como pudo apreciarse en los estudios planteados en las secciones 6.1.4.4 y 6.1.4.5, la introducción de demasiado ruido en el sistema de optimización puede traducirse en un empeoramiento de la convergencia ofrecida. En un escenario en el que el modelo pudiera ajustarse adecudamente mediante re-entrenamiento con nueva información, este problema podría ser mitigado.

Por otra parte, en la tabla 6.29 se presenta una comparativa de los conjuntos de pesos con mejor aptitud obtenidos al aplicar la arquitectura Soft Computing tanto al emplear valores aproximados de la función de aptitud como al no emplearlos (cromosoma obtenido en el estudio presentado en la sección anterior 6.2.2) para la determinación de las preferencias del grupo de usuarios 276-5M-20k. Como puede apreciarse, ambos conjuntos coinciden en indicar que los atributos más valorados por los usuarios presentes en el grupo 276-5M-20k son la fecha de lanzamiento de la película y los géneros de acción y drama, con porcentajes de importancia superiores al 15 %. Sin embargo, las preferencias aprendidas empleando el valor preciso de la función de aptitud indican que existe una cuarta característica muy apreciada por los usuarios, la pertenencia de la película al género de la comedia.

De forma similar, en ambos perfile se establece que ciertos atributos carecen de interés (0 % de importancia) para los usuarios, en concreto los atríbutos correspondientes a los géneros de documental, fantasia, musical, misterio, guerra y del oeste. No obstante, existen divergencias sobre otros atributos, que en un caso se determinan como prescindibles mientras que para el otro perfil tienen cierto grado de importancia. Por ejemplo, el atributo correspondiente al género de animación no es de interés según el perfil de preferencias aprendido empleando el valor exacto de la función de aptitud, mientras que para el otro caso alcanza un 6.53 % de importancia. Por tanto, pese a la imprecisión introducida al emplear un modelo aproximado de la función de aptitud, la arquitectura de búsqueda consigue obtener un perfil que define las preferencias del grupo de usuarios de forma similar a como lo hace cuando emplea valores exactos para la misma.

A lo largo la sección 6.2 se han presentado los diferentes estudios realizados para eva-

Corpus de test	MAE	MSE	Aciertos %
276-5M-20k	0.3167	0.1521	55.66 %
276	0.2875	0.1094	65.00 %
5	0.3250	0.1687	65.00 %
58	0.3625	0.1719	35.00 %
236	0.3000	0.1562	55.00 %
879	0.2625	0.1281	65.00 %
895	0.2875	0.1531	50.00 %

Cuadro 6.28: Errores absolutos (MAE), cuadráticos (MSE) y tasas de aciertos obtenidas al emplear las preferencias aprendidas para el grupo de usuarios 276-5M-20k para estimar las valoraciones que estos usuarios realizarían sobre las películas presentes en el corpus de test. Además, también se muestran los resultados obtenidos al utilizar esas preferencias para estimar las valoraciones de las 20 últimas películas puntuadas por el usuario de referencia 276 y otros cinco usuarios similares a él.

Peso	Con Aprox.	Sin Aprox.
wRelease_date	**16.05 %**	**20.00 %**
wunknoun	0.00 %	2.65 %
wAccion	**24.00 %**	**18.46 %**
wAdventure	0.00 %	0.91 %
wAnimation	6.54 %	0.00 %
wChildrens	0.89 %	0.00 %
wComedy	0.27 %	**18.52 %**
wCrime	0.63 %	0.00 %
wDocummentary	0.00 %	0.00 %
wDrama	**51.19 %**	**27.65 %**
wFantasy	0.00 %	0.00 %
wFilm_Noir	0.19 %	0.00 %
wHorror	0.03 %	4.89 %
wMusical	0.00 %	0.00 %
wMystery	0.00 %	0.00 %
wRomance	0.22 %	2.08 %
wSci_Fi	0.00 %	1.18 %
wThriller	0.00 %	3.66 %
wWar	0.00 %	0.00 %
wWestern	0.00 %	0.00 %

Cuadro 6.29: Comparativa de las preferencias aprendidas para el grupo 276-5M-20k al emplear valores aproximados de la función de aptitud y al no emplearlos. En ambos casos se presentan los mejores conjuntos de valores (cromosmas) tomados por los diferentes pesos de la importancia asociada a cada uno de los atributos que definen a una película.

luar la utilidad de la arquitectura de búsqueda *Soft Computing* propuesta en la captura de las preferencias de los usuarios dentro del ámbito de los Sistemas de Recomendación, determinando qué atributos o características son más y menos valorados por los usuarios.

En concreto, la arquitectura ha sido utilizada para determinar perfiles de las preferencias de ciertos usuarios a partir de la información disponible sobre ellos o a partir de la información disponible para otros usuarios similares a ellos. En este segundo caso, se emplearon un número reducido de valoraciones de partida realizadas por los usuarios de referencia, a partir de la cuales se realizó el proceso de selección de los usuarios similares presentes en el sistema. Además, los perfiles de preferencias aprendidos han sido empleados para predecir las valoraciones que los usuarios realizarían sobre ciertas películas, obteniendo buenos porcentajes de acierto. Asimismo, las preferencias aprendidas inicialmente para un usuario de referencia concreto, fueron posteriormente empleadas para predecir las puntuaciones que realizarían otros usuarios considerados parecidos a él, obteniendo también resultados aceptables. Por todo ello, mediante esta aproximación sería posible determinar un primer perfil de las preferencias de los nuevos usuarios presentes en un sistema, para poder recomendarle nuevos productos o servicios de forma adecuada, puesto que la arquitectura permite trabajar con poca información de partida.

Capítulo

7

Conclusiones

A lo largo de este capítulo se presentarán las principales conclusiones de este trabajo, las líneas futuras de investigación a seguir, así como las publicaciones que han sido fruto del trabajo de investigación realizado.

7.1. Aportaciones

Acorde con los objetivos de este trabajo y como resultado del trabajo de investigación realizado, la principal aportación de esta tesis ha sido la propuesta de una arquitectura de búsqueda independiente del dominio de aplicación y capaz de abordar problemas combinatorios de grandes dimensiones, en los que se disponga de poca información de partida. Esta arquitectura está basada en técnicas Soft Computing, pues combina un algoritmo genético basado en codificación real con modelos basados en redes neuronales, concretamente en perceptrones multicapa. Así, el algoritmo genético emplea, en los casos en los que sea necesario, modelos aproximados de las funciones de aptitud mediante perceptrones diseñados para tal fin. El sistema obtenido ofrece la flexibilidad y versatilidad requeridas para poder adaptarse a los requisitos propios de cada problema combinatorio a tratar, sea cual sea su dominio.

La arquitectura de búsqueda fruto de la presente tesis consta de un total de seis eta-

pas: (i) configuración, (ii) re-entrenamiento de la RNA, (iii) modelado de la función de aptitud o *fitness*, (iv) aplicación de los operadores del AG, (v) evaluación simulada y (vi) evaluación de control. Además, la arquitectura ofrece un sistema de codificación de los problemas a tratar que le permite abordar cualquier clase de problema combinatorio. Asimismo, permite la adecuada configuración de sus etapas para adaptarse a las particularidades concretas del problema a tratar, ejecutando en cada caso las etapas requeridas, a fin de ofrecer en todo momento el rendimiento adecuado.

Por otra parte, a fin de determinar las técnicas más adecuadas para la arquitecta resultado del presente trabajo, se realizó una revisión de las principales técnicas Soft Computing actuales. Como resultado de este trabajo pudo constatarse que estas técnicas ofrecen soluciones a bajo coste, robustas y flexibles, a cambio de cierta imprecisión e incertidumbre. Además, permiten trabajar con cierto grado de inexactitud en la información de partida, lo que las hace muy interesantes en dominios en los que exista cierto grado de error experimental. También se constató que las herramientas empleadas en este tipo de técnicas presentan ciertas ventajas e inconvenientes cuando son empleadas por separado, pero que su combinación posibilita potenciar sus virtudes minimizando sus desventajas. En este sentido, se determinó que un sistema de búsqueda efectivo para problemas combinatorios de diversa índole podría estar constituido por la combinación de algoritmos genéticos con redes neuronales.

Los algoritmos genéticos ofrecen un mecanismo adaptativo de resolución de problemas, de forma que aunque se produzcan cambios en el problema original, son capaces de seguir resolviéndolo. Permiten trabajar con problemas de grandes dimensiones, sin embargo para su correcto funcionamiento, los algoritmos genéticos requieren que se pueda calcular una función objetivo o de aptitud que guíe la evolución de las generaciones a fin de obtener las soluciones esperadas. Desgraciadamente, no en todos los casos es posible conocer o determinar una función de aptitud adecuada, ya sea porque su cálculo de forma directa sea muy costoso o por su complejidad. Sin embargo, las redes neuronales son un excelente mecanismo de aproximación de funciones no lineales sin necesidad de tener un conocimiento previo del problema a resolver. Son capaces de ofrecer buenas aproximaciones con pocos datos de partida, aunque requieren de un gran volumen de información y largos períodos de aprendizaje si se necesitan modelos precisos. Por todo ello, mediante la combinación de algoritmos genéticos con redes neuronales sería posible abordar problemas combinatorios de grandes dimensiones, aunque la obtención directa de la función objetivo sea difícil o costosa.

Los motivos principales que desencadenaron la obtención de una arquitectura de búsqueda capaz de abordar problemas de grandes dimensiones fueron las necesidades de resolver ciertos problemas de búsqueda planteados dentro de dos dominios muy diferentes: la Catálisis Combinatoria y los Sistemas de Recomendación. Por ello, dentro del trabajo realizado durante la presente tesis se realizó un estudio de los requisitos y necesidades de los problemas a resolver dentro del ámbito de ambos dominios.

Así pues, en relación al resultado del análisis de los requisitos y necesidades de los problemas a resolver dentro del ámbito de la Catálisis Combinatoria, el estudio estableció que dentro de este dominio eran necesarias nuevas herramientas que posibilitaran tratar con un amplio rango de problemas. Además, estas herramientas deberían permitir que los investigadores adaptasen la estrategia de optimización a seguir según fueran las características particulares del problema a abordar, así como los recursos disponibles para realizar la investigación. Las nuevas aplicaciones de búsqueda deberían incrementar la convergencia ofrecida por las soluciones actuales, sin disminuir la capacidad exploradora necesaria para abordar los grandes espacios de búsqueda que definen los problemas a resolver. Asimismo, sería necesario que permitieran plasmar al investigador las características y restricciones particulares del problema catalítico a tratar de forma sencilla y clara. Por tanto, la especificación de los problemas debería permitir, entre otros aspectos: definir una formulación catalítica compleja que comprenda diferentes componentes; establecer reglas o restricciones asociadas; indicar otro tipo de variables como los procedimientos de preparación, condiciones de prueba catalítica, etc.

Con respecto al estudio de las características y requisitos a contemplar dentro del dominio de los Sistemas de Recomendación, la revisión realizada hacía hincapié en los problemas a abordar cuando se requiere obtener las preferencias de los usuarios y revisaba las principales técnicas empleadas. Fruto del análisis de las técnicas empleadas se determinó que uno de los principales inconvenientes de los algoritmos basados en el contenido y de los algoritmos de filtrado por colaboración ítem a ítem, es que no son capaces de ofrecer recomendaciones novedosas a sus usuarios. Es decir, no pueden proponer nuevos productos o servicios que sean totalmente diferentes a los ya valorados o comprados por el usuario y que puedan resultarle interesantes. Este aspecto sí es abordado por los sistemas de filtrado por colaboración basada en los usuarios y por los sistemas basados en algoritmos de *clustering*.

Otro problema común de los algoritmos presentados consiste en afrontar la escasez y dispersión de la información en alguna de sus etapas, sobre todo cuando entra un usua-

rio nuevo en el sistema. Los sistemas que emplean el filtrado por colaboración ítem a ítem son menos sensibles a este problema, puesto que a partir de una única valoración son capaces de ofrecer nuevas recomendaciones. Sin embargo, para que ofrezcan recomendaciones realmente útiles para el usuario, requieren que éste valore o compre numerosos artículos. Por tanto, los sistemas de recomendación requieren de técnicas que les permitan ofrecer recomendaciones aceptables para el usuario en todo momento, aunque no se disponga de un gran volumen de información.

Además se constató que muchos de los algoritmos presentados son utilizados en los Sistemas de Recomendación de empresas dedicadas al comercio electrónico para sugerir productos a sus consumidores que se adecúen a sus preferencias, a fin de incrementar su volumen de ventas. Sin embargo, se observó que estos algoritmos tienen problemas para determinar el perfil de las características de los productos más deseadas o valoradas por los usuarios, sobre todo en los casos en los no se dispone de un gran volumen de información del que inferir sus gustos. Por tanto, las empresas necesitan disponer de nuevas herramientas capaces de abordar este tipo de problemas combinatorios, en el que se persigue conocer la combinación de atributos de un producto o servicio óptimos para un cliente, incluso cuando se disponga de poca información de partida.

Para evaluar la efectividad y validez de la arquitectura de búsqueda *Soft Computing* fruto de la presente tesis, ésta fue aplicada a la resolución de problemas combinatorios de interés, tanto en el área de la Catálisis Combinatoria como en el dominio de los Sistemas de Recomendación. Concretamente, dentro del dominio de la Catálisis, la técnica propuesta fue utilizada tanto para optimizar condiciones de distintas reacciones, como para determinar las composiciones idóneas de catalizadores indicados para reacciones de naturaleza y complejidad diferentes. En todos los casos, el número de individuos que finalmente fueron evaluados en un reactor (catalizadores a sintetizar o condiciones diferentes a probar), era muy inferior al que se hubiera requerido empleando una técnica de búsqueda tradicional. Además, se demostró que es posible obtener bibliotecas de modelos basados en redes neuronales artificiales para diferentes reacciones catalíticas, capaces de aproximar adecuadamente los resultados catalíticos de reacciones similares para los que no fueron entrenados inicialmente, requiriendo de un número muy reducido de muestras de las nuevas reacciones a aproximar.

Con respecto a la aplicación de la arquitectura de búsqueda planteada en el ámbito de los Sistemas de Recomendación, el estudio se centró sobre un dominio de entretenimiento: la valoración de películas. Para ello se empleó el conjunto de datos *MovieLens*

[Herlocker et al., 1999], utilizado habitualmente como *benchmark* en este ámbito. Así, la arquitectura fue utilizada para determinar los perfiles de las preferencias de ciertos usuarios a partir de la información disponible sobre ellos o a partir de la información disponible para otros usuarios similares a ellos. En este segundo caso, se emplearon un número reducido de valoraciones de partida realizadas por los usuarios de referencia, a partir de la cuales se realizó el proceso de selección de los usuarios similares presentes en el sistema. Además, las preferencias aprendidas fueron empleadas posteriormente para predecir las valoraciones que los usuarios realizarían sobre ciertas películas, obteniendo buenos porcentajes de acierto. Asimismo, las preferencias aprendidas inicialmente para un usuario de referencia concreto, fueron posteriormente empleadas para predecir las puntuaciones que realizarían otros usuarios considerados parecidos a él, obteniendo también resultados aceptables. Por todo ello, se determinó que mediante esta aproximación sería posible establecer un primer perfil de las preferencias de los nuevos usuarios presentes en un sistema, para poder recomendarle nuevos productos o servicios de forma adecuada, puesto que la arquitectura permite trabajar con poca información de partida.

Finalmente, la arquitectura propuesta ha sido empleada como marco de trabajo en la obtención del paquete de aplicaciones o herramientas *SoftCombi* (descrita en el apéndice A), que permite el diseño inteligente de experimentos en el ámbito de la Catálisis Combinatoria. Este paquete consta de dos herramientas principales. La primera de ellas permite diseñar y ejecutar todo tipo de experimentos dentro del ámbito de la catálisis: la búsqueda de nuevos compuestos, el establecimiento de condiciones óptimas de reacción, la configuración adecuada de los modos de preparación, etc. Mediante la segunda herramienta se ofrece una aplicación sencilla que permite obtener modelos de las funciones de aptitud requeridas en cada caso. Concretamente se obtienen perceptrones multicapa siguiendo los pasos descritos en la arquitectura para analizar diversas combinaciones de topologías y algoritmos de entrenamiento.

7.2. Lineas Futuras de Investigación

Una de las futuras líneas de investigación versa sobre la mejora de la arquitectura en las ocasiones en las que los modelos empleados para aproximar las funciones de aptitud, basados en perceptrones multicapa, no sean lo suficientemente precisos, de forma que la introducción de un alto grado de imprecisión produzca el empeoramiento de la convergencia ofrecida. Para ello, se podría dotar a la arquitectura de otras técnicas también

capaces de aproximar funciones, de forma que se pudiera seleccionar el tipo de modelo más adecuado en cada ocasión. Unas firmes candidatas para esta labor son las Máquinas de Soporte Vectorial (MSV). Tal y como se mostró en el análisis de las diferentes técnicas Soft Computing actuales, las MSV ofrecen buenas generalizaciones con ausencia de mínimos locales y suelen tener menos problemas de sobre-entrenamiento que otras técnicas. Sin embargo, necesitan largos procesos de entrenamiento ante problemas de grandes dimensiones, con una alta complejidad de cómputo y grandes necesidades de memoria.

Una segunda línea de investigación será la aplicación de la arquitectura propuesta en un nuevo problema de interés industrial: la optimización del funcionamiento de reactores catalíticos de membrana de alta temperatura. Mediante el uso de la herramienta de optimización desarrollada se pretende encontrar las condiciones de operación más adecuadas para maximizar el rendimiento de etileno en la reacción de deshidrogenación oxidativa de etano (ODHE). La configuración del reactor incluye: (i) una membrana conductora mixta de electrones e iones oxígeno en forma de disco plano; (ii) una cámara en la que se alimenta continuamente oxígeno diluido en nitrogeno y argon; y (iii) una cámara en la se alimenta continuamente una corriente de etano diluido en argon. La reacción tiene lugar en la superficie de la membrana donde confluyen el etano gaseoso con el oxígeno permeando a través de la membrana, mientras que la temperatura de operación está en el rango entre 750 y 900^{o}C. La dificultad de esta optimización radica en el acoplamiento de procesos muy complejos tales como: (i) condiciones hidrodínámicas en ambas cámaras que definen el intercambio de materia entre la corriente de gas y la superficie de la membrana; (ii) la reacción catalítica de ODHE, especialmente las reacciones secundarias de degradación del etileno producido; y (iii) los fenómenos de transporte y separación del oxígeno en estado sólido a través de la membrana cerámica. En resumen, las condiciones de operación a optimizar son: (i) la temperatura nominal del reactor; (ii) el caudal gaseoso en la cámara de reacción; (iii) la concentración de hidrocarburo en la cámara de reacción; (iv) el caudal gaseoso en la cámara del oxígeno; y (v) la concentración de oxígeno en dicha cámara.

7.3. Publicaciones

A continuación se presentan las publicaciones resultantes del trabajo de investigación realizado durante la realización de la tesis, que comprenden artículos en revistas internacionales, así como comunicaciones en congresos internacionales y nacionales (ordenados

según el año de publicación).

- *Capturing user's preferences using a genetic algorithm. Determining Essential and Dispensable Item Attributes.* S.Valero, E. Argente, V. Botti *Proceedings ICAART 2010*, Vol. I, pp. 599-602. (2010)

- *DoE Framework for Catalyst Development based on Soft Computing* S.Valero, E. Argente, V. Botti, J.M. Serra, M. Moliner and A. Corma *Computers & Chemical Engineering*, Vol. 33, No. 1 pp. 225-238. (2009)

- *Soft Computing Techniques Applied to Combinatorial Catalysis: A New Approach for the Discovery and Optimization of Catalytic Materials.* J.M. Serra, A. Corma, S. Valero, E. Argente and V. Botti.*QSAR & Combinatorial Science* , No. 1, pp. 11-26. (2007)

- *Titano-Silicatos Mesoporosos como Catalizadores de Expoxidación: Optimización y Estudio de la Relación Sístenis-Caracterización-Reactividad Mediante Técnicas Combinatorias.* P.Serna, J.M. Serra, M. Moliner, A. Corma, S. Valero, E. Argente and V.Botti. *SECAT'05 - Catálisis y Materiales Mesoestructurados. Reunión de la Sociedad Española de Catálisis*, pp. 119-120. (2005)

- *Optimization of Olefin Epoxidation Catalysist applying High-Throughput and Genetic Algorithms assisted by Artificial Neural Networks (Softcomputing Techniques).* A. Corma, J.M. Serra, P. Serna, S. Valero, E. Argente and V. Botti. *Journal of Catalysis*, Vol. 225, pp. 513-524. (2005)

- *Application of Artificial Neural Networks to High-Throughput Synthesis of Zeolites.* M. Moliner, J.M. Serra, A. Corma, E. Argente, S. Valero and V. Botti. *Microporous and Mesoporous Materials. Zeolits, Clarys, Carbons and Related Materials*, Vol. 78, pp. 73-81. (2005)

- *A Soft Computing Technique applied to Industrial Catalysis.* S. Valero, E. Argente, J.M. Serra, P. Serna, V. Botti and A. Corma. *Proceedings ECAI-PAIS 2004*, Vol. 1, pp. 765-769. (2004)

- *SoftComputing Techniques Applied to Catalytic Reactions.* S. Valero, E. Argente, V. Botti, J.M. Serra and A. Corma. *Lecture Notes in Artificial Intelligence*, Vol. 3040, pp. 550-559. (2004)

- *Softcomputing Techniques Applied to Catalytic Reactions*. S. Valero, E. Argente, V. Botti, J.M. Serra and A. Corma. *CAEPIA-TTIA'2003*, Vol. I, pp. 213-222. (2003)

- *Neural Networks for Modelling of Kinetic Reaction Data Applicable to Catalyst Scale Up and Process Control and Optimisation in the Frame of Combinatorial Catalysis*. A. Corma, J.M. Serra, E. Argente, S. Valero and V. Botti. *Applied Catalysis A: General*, Vol. 254, pp. 133-145. (2003)

- *Development of New Materials by Combinatorial Techniques*. J.M. Serra, A. Corma, E. Argente, S. Valero and V. Botti. *ICEE 2003*, pp. 21-25. (2003)

- *Application of Artificial Neural Networks to Combinatorial Catalysis: Modeling and Predicting ODHE Catalysts*. A. Corma, J.M. Serra, E. Argente, V. Botti and S. Valero. *ChemPhysChem, a European Journal of Chemical Physics and Physical Chemistry*, Vol. 3, No. 11, pp. 939 - 945. (2002)

Apéndice

A

Paquete SoftCombi

Como resultado del trabajo de investigación presentado en esta tesis, se han obtenido una serie de herramientas basadas en técnicas Soft Computing que facilitan el diseño de experimentos en el ámbito de la Catálisis Combinatoria.

A.1. Herramienta para el diseño inteligente de experimentos en el ámbito de la Catálisis Combinatoria

La aplicación desarrollada (Figura A.1) permite diseñar experimentos dentro de la Catálisis Combinatoria, ayudando de manera eficaz a plantear y llevar a cabo nuevas pruebas encaminadas a descubrir nuevos materiales catalíticos, sobre todo en los casos en los que es necesario explorar espacios de soluciones de grandes dimensiones. Permite la optimización eficiente en estos espacios, puesto que entre otros aspectos, utiliza los puntos ya explorados para reducir el número de muestras de baja calidad que finalmente han de ser sintetizadas y probadas experimentalmente.

Así pues, la herramienta software implementada permite un diseño inteligente de los experimentos a realizar puesto que, en primer lugar, requiere de un número reducido de muestras, permitiendo trabajar con poblaciones acordes con el rango utilizado en

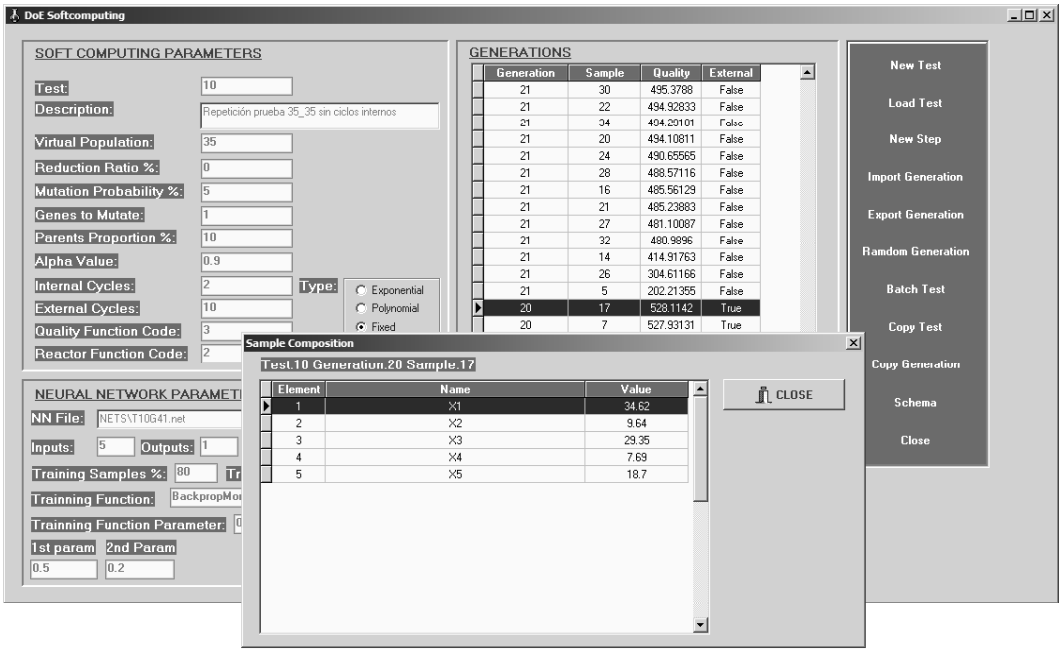

Figura A.1: Herramienta para el diseño inteligente de experimentos en el ámbito de la Catálisis Combinatoria basada en técnicas Soft Computing

las técnicas empleadas normalmente para la caracterización y prueba de los materiales. En segundo lugar, la rápida convergencia exhibida por la herramienta desarrollada permite identificar áreas de alta calidad dentro del espacio de búsqueda, requiriendo para ello calcular pocas generaciones de posibles soluciones. Por tanto, su aplicación en experiencias en las que se empleen técnicas aceleradas de experimentación puede producir una significante reducción del coste temporal y financiero requerido siguiendo las técnicas convencionales.

Otra importante contribución es que la herramienta permite definir optimizaciones (codificación) en las que se considere de forma simultánea formulaciones de catalíticas complejas y diferentes condiciones de síntesis o prueba. Además, es posible definir reglas a cumplir entre las diferentes variables estudiadas. Asimismo, la herramienta permite llevar a cabo estudios en los que no sólo se optimice la composición de las muestras, si no que también permite realizar un análisis de los elementos que componen las muestras (identificando a los elementos químicos que aparecen con mayor frecuencia en los mejores catalizadores, con qué porcentaje de composición, etc.)

Finalmente, aunque la herramienta fue inicialmente diseñada para ser empleada en el campo de la Catálisis, la codificación desarrollada es lo suficientemente general como

para permitir que la herramienta sea aplicada con facilidad en otras áreas como la ciencia de los materiales o el descubrimiento de fármacos.

A.1.1. Principales Funcionalidades

La herramienta presenta una panel de botones desde la que acceder a las funcionalidades que ofrece, mediante la pulsación del botón correspondiente (Figura A.1). Entre otras, las principales funcionalidades ofertadas son:

- **Nuevo experimento.** A través del botón *New Test* el usuario puede especificar un nuevo experimento a realizar, indicando el valor de los diferentes parámetros que condicionan el proceso de optimización a seguir (descritos en la sección 5.1.2). Asimismo, permite indicar la definición o codificación del problema a optimizar mediante una descripción xml que sigue el esquema de codificación propuesto en la sección 5.1.1, que puede editarse con cualquier tipo de procesador de textos. El usuario dispone de una serie de ejemplos de codificaciones de problemas tipo, que le facilitan la labor de descripción de un nuevo experimento y le sirven de ejemplo a la hora de describir las variables a estudiar y las reglas que deben cumplirse. Entre otros aspectos, la codificación le permite:
  - Definir una formulación catalítica compleja que comprenda diferentes componentes, tales como soportes, promotores de fase activa, potenciadores, etc.
  - Establecer reglas o restricciones asociadas (químicas/termodinámicas u orientadas a la aplicación final).
  - Indicar otro tipo de variables como los procedimientos de preparación, condiciones de prueba catalítica, etc.

  Además, en esta misma opción el usuario puede indicar el modelo basado en un perceptrón multicapa que será empleado para aproximar los valores de la función de aptitud utilizada, en el que caso en el que se desee emplear esta característica.

- **Cargar un experimento**. El usuario puede recuperar en cualquier momento un experimento ya en curso, a fin de modificar alguno de los parámetros de la arquitectura, operar con él o analizar sus resultados. Para ello sólo debe seleccionar el botón *Load Test*.

- **Obtener una generación aleatoria.** La aplicación permite obtener una generación inicial para un experimento ya especificado, indicando el número de

generaciones aleatorias que se desea calcular, a partir de las cuales propondrá la de mayor diversidad como punto de inicio para la nueva experimentación. Para ello el usuario ha de pulsar el botón *Random Generation*

- **Realizar un paso de optimización.** Tras seleccionar esta opción mediante el botón *New Step*, la herramienta realiza un paso de optimización siguiendo las directrices marcadas cuando se especificó el experimento cargado actualmente (los diferentes parámetros de la arquitectura Soft Computing desarrollada).

- **Importar y exportar generaciones.** Todos los resultados obtenidos (condiciones de reacción propuestas, composiciones de los catalizadores, calidad esperada, etc.) pueden ser exportados a ficheros de texto. Asimismo, es posible incorporar nueva información o modificaciones a los datos almacenados mediante la importación de datos contenidos en ficheros de texto convenientemente formateados.

- **Almacenamiento de la información.** La herramienta permite almacenar la información en cualquier tipo de base de datos a través de una sencilla conexión ODBC.

A.2. Herramienta para la obtención de modelos de las funciones de aptitud

Mediante esta segunda herramienta (Figura A.2) se facilita la labor de obtención de los modelos de las funciones de aptitud basados en perceptrones multicapa, los cuáles son empleados por la herramienta de diseño en su proceso de optimización. Para ello, la herramienta sigue las directrices indicadas en el la sección 5.1.4 de esta tesis para la determinación de la red neuronal adecuada a cada problema. Así, investiga entre diferentes combinaciones de topologías y funciones de entrenamiento a fin de proponer el modelo más adecuado a cada problema. La aplicación emplea internamente el simulador de redes neuronales SNNS, desarrollado por el Instituto de Sistemas de Alto Rendimiento Paralelos y Distribuidos de la Universidad de Stuttgart [Zell et al., 1995].

El usuario debe indicar el número de nodos de entrada y de salida, así como el número de nodos máximo de las capas intermedias, en base a los nodos de entrada. Si el usuario no especifica otro valor, la aplicación estudiará por defecto hasta el doble de nodos situados en la capa de entrada.

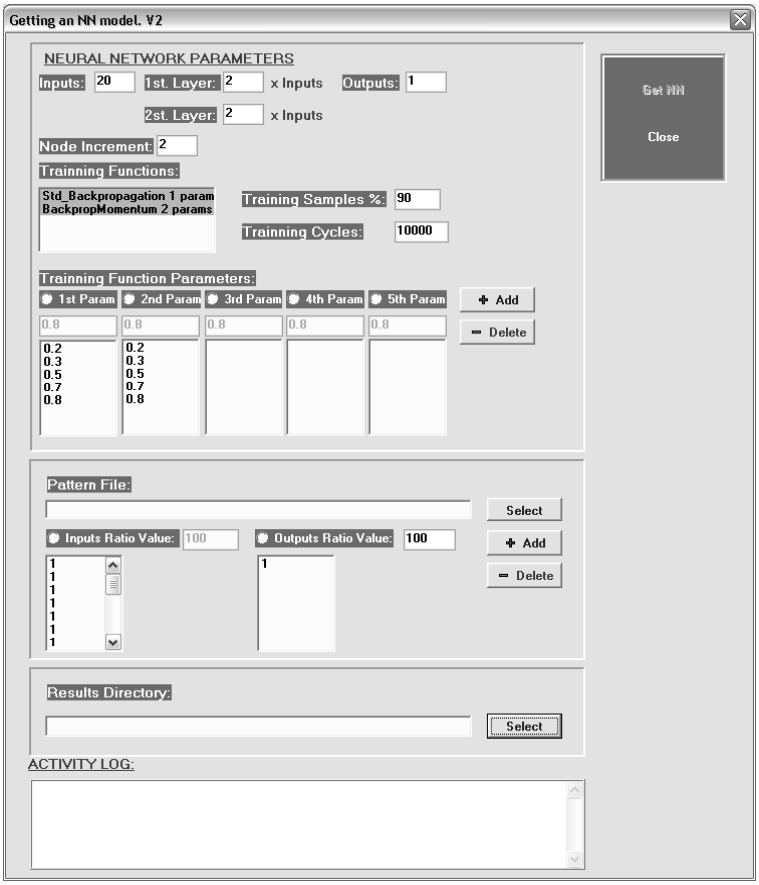

Figura A.2: Herramienta para la obtención de modelos basados en perceptrones multicapa

De forma similar, el usuario puede indicar las funciones de entrenamiento que desea estudiar, así como el valor de sus parámetros que desee probar. Sin embargo, la aplicación propone de partida estudiar los dos algoritmos de entrenamiento que mejores resultados ofrecen según la literatura [Ripley, 1996][Duda et al., 2001], así como un amplio rango de valores para sus parámetros.

Por último, el usuario necesita indicar la ruta del fichero en la que se encuentran los datos a emplear en el estudio (en un fichero de texto con el formato empleado por la aplicación de diseño). De forma similar, el usuario debe indicar una ruta de directorio donde se almacenarán todos los resultados.

Cuando la aplicación finaliza, informa al usuario de la topología seleccionada, el algoritmo de entrenamiento apropiado y valores adecuados para sus parámetros. Además, también informa del error cuadrático medio estimado para las aproximaciones a realizar por el modelo.

Bibliografía

Aarts, E. and Lenstra, J., editors (1997). *Local Search in Combinatorial Optimization*. John Wiley & Sons, Chichester, England.

Alaradi, A. and Rohani, S. (2002). Identification and control of a riser-type FCC unit using neural networks. *Computers & Chemical Engineering*, 26:401–421.

Amento, B., Terveen, L., Hill, W., Hix, D., and Schulman, R. (2003). Experiments in social data mining: The topicshop system. *ACM Transactions on Computer-Human Interaction*, 10(1):54–85.

Aramendía, M., Borau, V., Jiménez, C., Marinas, J., Romero, F., and Urbano, F. (2002). An approach to the construction of indexed libraries for the combinatorial selection of heterogeneous catalysts. *Journal of Catalysis*, 209(2):413–416.

Bäck, T., Hoffmeister, F., and Schwefel, H.-P. (1991). Extended selection mechanisms in genetic algorithms. In Belew, R. and Booker, L., editors, *Proc. of the Fourth Int. Conf. on Genetic Algorithms.*, pages 92–99, San Mateo. Morgan Kaufmmann.

Baker, J. E. (1985). Adaptive selection methods for genetic algorithms. In *Proceedings of the 1st International Conference on Genetic Algorithms*, pages 101–111, Hillsdale, NJ, USA. L. Erlbaum Associates Inc.

Baker, J. E. (1987). Reducing bias and inefficiency in the selection algorithm. In *Proceedings of the Second International Conference on Genetic Algorithms and Their Application*, pages 14–21, Hillsdale, NJ, USA. L. Erlbaum Associates Inc.

Balabanović, M. (1997). An adaptive web page recommendation service. In *AGENTS*

'97: Proceedings of the first international conference on Autonomous Agents, pages 378–385, New York, NY, USA. ACM Press.

Balabanović, M. and Shoham, Y. (1997). Fab: content-based, collaborative recommendation. *Communications ACM*, 40(3):66–72.

Basu, C., Hirsh, H., and Cohen, W. (1998). Recommendation as classification: using social and content-based information in recommendation. In *AAAI '98/IAAI '98: Proceedings of the fifteenth national/tenth conference on Artificial intelligence/Innovative applications of artificial intelligence*, pages 714–720, Madison, Wisconsin,USA. American Association for Artificial Intelligence.

Billsus, D. and Pazzani, M. J. (2000). User modeling for adaptive news access. *User Modeling and User-Adapted Interaction*, 10(2-3):147–180.

Biniwale, R., Labhsetwar, N., Kumar, R., and Hasan, M. (2002). Catalytic converter modelling: Artificial neural networks for perovskite-based catalyst. *Society of Automotive Engineers, Technical Papers*, 1676:49–54.

Bishop, C. (1996). *Neural Networks for Pattern Recognition*. Oxford Clarendon Press, Oxford.

Bishop, C. M. (2006). *Pattern Recognition and Machine Learning*. Springer Science+Bussiness Media, LLC.

Blickle, T. and Thiele, L. (1995). A comparison of selection schemes used in genetic algorithms. Technical report, Gloriastrasse 35, CH-8092 Zurich: Swiss Federal Institute of Technology (ETH) Zurich, Computer Engineering and Communications Networks Lab (TIK.

Bradley, P., Fayyad, U., and Reina, C. (1998). Scaling clustering algorithms to large databases. In *Proceedings Fourth International Conference on Knowledge Discovery and Data Mining (KDD98) Agrawal, P. Stolorz and G. Piatetsky-Shapiro (eds.)*, pages 9–15, Menlo Park, CA. AAAI Press.

Breese, J., Heckerman, D., and Kadie, C. (1998). Empirical analysis of predictive algorithms for collaborative filtering. In *In Proceedings of the Fourteenth Annual Conference on Uncertainty in Artificial Intelligence*, pages 43–52.

Brindle, A. (1981). *Genetic algorithms for function optimization*. PhD thesis, University of Alberta, Department of Computer Science, Edmonton.

Broomhead, D. and Lowe, D. (1988). Multivariable functional interpolation and adaptative networks. *Complex Systems*, 2:312–355.

Bulsari, A. (1995). *Neural Networks for Chemical Engineers.* Elsevier, Amsterdam.

Burke, R. (2002). Hybrid recommender systems: Survey and experiments. *User Modeling and User-Adapted Interaction*, 12(4):331–370.

Burke, R., Mobasher, B., Williams, C., and Bhaumik, R. (2006). Detecting profile injection attacks in collaborative recommender systems. In *8th IEEE International Conference on E-Commerce Technology and the 3th IEEE International Conference on Enterprise Computing, E-Commerce and E-Services (CEC/EEE'06)*, pages 23–30, Palo Alto, CA, USA.

Buyevskaya, O., Brückner, A., Kondratenko, E., Wolf, D., and Baerns, M. (2001). Fundamental and combinatorial approaches in the search for and optimisation of catalytic materials for the oxidative dehydrogenation of propane to propene. *Catalysis Today*, 67(4):369–378.

Chakraborty, M. and Chakraborty, U. K. (1997). An analysis of linear ranking and binary tournament selection in genetic algorithms. In *Proceedings of the International Conference on Information, Communications and Signal Processing ICICS '97*, pages 407–411, Singapore. IEEE.

Chalmers, M., Rodden, K., and Brodbeck, D. (1998). The order of things: activity-centred information access. *Computer Networks and ISDN Systems. Proceedings of the Seventh International World Wide Web Conference*, 30(1-7):359–367.

Chesñevar, C., Maguitman, A., and Simari, G. (2006). Argument-based critics and recommenders: A qualitative perspective on user support systems. *Data & Knowledge Engineering*, 59:293–319.

Chica, A. (2002). *Desarrollo de catalizadores bifuncionales para procesos de hidroisomerización de diferentes fracciones del petróleo.* PhD thesis, ITQ. Universidad Politécnica de Valencia, Valencia, España.

Chica, A., Corma, A., and Miquel, P. (2001). Isomerisation of c5-c7 n-alkanes on unidirectional large pore zeolites: activity, selectivity and adsorption features. *Catalysis Today*, 65:101–110.

Claypool, M., Gokhale, A., Miranda, T., Murnikov, P., Netes, D., and Sartin, M. (1999). Combining content-based and collaborative filters in an online newspaper. In *In Proceedings of ACM SIGIR Workshop on Recommender Systems.*

Corma, A., Domine, M., Gaona, J., Jorda, J., Navarro, M., Rey, F., Perez-Pariente, J., Tsuji, J., McCulloch, B., and Nemeth, L. (1998). Strategies to improve the epoxidation activity and selectivity of Ti-MCM-41. *Chem. Comm.*, 2211.

Corma, A., Serra, J., Argente, E., Botti, V., and Valero, S. (2002a). Application of artificial neural networks to combinatorial catalysis: Modelling and prediction of ODHE catalysts. *ChemPhysChem*, 3(11):939–945.

Corma, A., Serra, J., and Chica, A. (2002b). Application of genetic algorithms to the development and optimisation of light paraffin isomerisation catalysts. In Derouane, E., Parmon, V., Lemos, F., and Ramôa Ribeiro, F., editors, *Principles and Methods for Accelerated Catalyst Design and Testing*, pages 153–172, Dordrecht. Kluwer Academic Press.

Corma, A., Serra, J., Serna, P., Argente, E., Botti, V., and Valero, S. (2005). Optimization of olefin epoxidation catalysist applying high-throughput and genetic algorithms assisted by artificial neural networks (softcomputing techniques). *Journal of Catalysis*, 225:513–524.

Crabtree, I. and Soltysiak, S. (1998). Identifying and tracking changing interests. *International Journal on Digital Libraries*, 2(1):38 – 53.

Cristianini, N. and Shawe-Taylor, J. (2000). *An Introduction to Support Vector Machines and other kernel-based learning methods.* Cambridge University Press, UK.

Cundari, T., Deng, J., and Zhao, Y. (2001). Design of a propane ammoxidation catalyst using artificial neural networks and genetic algorithms. *Industrial & Engineering Chemistry Research*, 40(23):5475–5480.

Davis, L. (1991). *Handbook of Genetic Algorithms.* Van Nostrand Reinhold, New York.

Derouane, E. (2002). *Principles and Methods for Accelerated Catalyst Design & Testing.* Kluwer Academic Publisher.

Duda, R. and Hart., P. (1973). *Pattern Classification and Scene Analysis.* John Wiley & Sons.

Duda, R., Hart, P., and Stork, D. (2001). *Pattern Classification. 2nd Ed.* Jonh Wiley & Sons.

Eshelman, L. J., Caruana, R. A., and Schaffer, J. D. (1989). Biases in the crossover landscape. In *Proceedings of the third international conference on Genetic algorithms*, pages 10–19, San Francisco, CA, USA. Morgan Kaufmann Publishers Inc.

Eshelman, L. J. and Schaffer, J. (1993). Real-coded genetic algorithms and interval-schemata. In Whitley, L. D., editor, *Foundations of Genetic Algorithms 2*, pages 187–202, San Mateo,CA. Morgan Kaufmann Publishers.

Feo, T. and Resende, M. (1995). Greddy randomised adaptative search procedures. *Journal of Global Optimization*, 6:109–133.

Gedeck, P. and Willett, P. (2001). Visual and computational analysis of structure activity relationships in high-throughput screening data. *Current Opinion in Chemical Biology*, 5:389–395.

Ghosh, P., Sundaram, A., Venkatasubramanian, V., and Caruthers, J. M. (2000). Integrated product engineering: a hybrid evolutionary framework. *Computers & Chemical Engineering*, 24(2-7):685–691.

Gilardoni, F., Graham, A., McKay, B., and Brown, B. (2003). Rational design, an alternative to the combinatorial explosion. In *Proceeding of 225th ACS National Meeting*, New Orleans.

Glover, F. and Laguna, M. (1997). *Tabu Search.* Kluwer Academic Publishers.

Goldberg, D. (1989). *Genetic Algorithms in Search, Optimization, and Machine Learning.* Addison-Wesley.

Goldberg, D. (1991). Real-coded genetic algorithms, virtual alphabets, and blocking. *Complex Systems*, 5:139–157.

Goldberg, D. E. and Deb, K. (1991). *A comparative analisys of selection schemes used in genetic algorithms*, pages 69–93. Morgan Kaufmann, San Mateo.

Gomory, R. E. (1963). An algorithm for integer solutions to linear programs. In Graves, R. and P.Wolfe, editors, *Recent Advances Mathematical Programming*, pages 269–302, New York. McGraw-Hill.

Good, N., Schafer, J., Konstan, J., Borchers, A., Sarwar, B., Herlocker, J., and Riedl, J. (1999). Combining collaborative filtering with personal agents for better recommendations. In *In Proceedings of the Sixteenth National Conference on Artificial Intelligence*, pages 439–446.

Grötschel, M. and Holland, O. (1991). Solution of large-scale symmetric travelling salesman problems. *Mathematical Programming.*, 51(2):141–202.

Gruber, T. (1993). A translation approach to portable ontologies specifications. *Journal on Knowledge Acquisition*, 5(2):199–220.

Guan, S., Chan, T., and Zhu, F. (2005). Evolutionary intelligent agents for e-commerce: Generic preference detection with feature analysis. *Electronic Commerce Research and Applications*, 4(4):377–394.

Guan, S., Ngoo, C., and Zhu, F. (2002). Handy broker: an intelligent product-brokering agent for m-commerce applications with user preference tracking. *Electronic Commerce Research and Applications*, 1(Issues 3-4):314–330.

Guan, S., Tan, P., and Chan, T. K. (2004). Intelligent product brokering for e-commerce: an incremental approach to unaccounted attribute detection. *Electronic Commerce Research and Applications*, 3(3):232–252.

Hattori, T. and Kito, S. (1995). Neural network as a tool for catalyst development. *Catalysis Today*, 23:347–355.

Herlocker, J., Konstan, J., and Riedl, J. (2000). Explaining collaborative filtering recommendations. In *CSCW '00: Proceedings of the 2000 ACM conference on Computer supported cooperative work*, pages 241–250, New York, NY, USA. ACM Press.

Herlocker, J. L., Konstan, J. A., Borchers, A., and Riedl, J. (1999). An algorithmic framework for performing collaborative filtering. In *SIGIR '99: Proceedings of the 22nd annual international ACM SIGIR conference on Research and development in information retrieval*, pages 230–237, New York, NY, USA. ACM.

Herrera, F., Lozano, M., Pé, E., Sánchez, A., and Villar, P. (2002). Multiple crossover per couple with selection of the two best offspring: An experimental study with the blx-α crossover operator for real-coded genetic algorithms. *IBERAMIA 2002, LNAI, Springer Verlang Berlin Heidelberg*, 2527:392–401.

Herrera, F., Lozano, M., and Verdegay, J. (1998). Tackling real-coded genetic algorithms. Operators and tools for behavioural analysis. *Artificial Intelligence Review*, 12:265–319.

Hill, W., Stead, L., Rosenstein, M., and Furnas, G. (1995). Recommending and evaluating choices in a virtual community of use. In *CHI '95: Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 194–201, New York, NY, USA. ACM Press/Addison-Wesley Publishing Co.

Holland, J. H. (1992). *Adaptation in natural and artificial systems*. MIT Press, Cambridge, MA, USA.

Hou, Z., Dai, Q., Wu, X., and Chen, G. (1997). Artificial neural network aided design of catalyst for propane ammoxidation. *Applied Catalysis A: General*, 161:183–190.

Huang, K., Chen, F., and Lu, D. W. (2001). Artificial neural network-aided design of a multi-component catalyst for methane oxidative coupling. *Applied Catalysis A: General*, 219:61–68.

Jin, Y. (2005). A comprehensive survey of fitness approximation in evolutionary computation. *Soft Computing-A Fusion of Foundations, Methodologies and Applications*, 9(1):3–12.

Kecman, V. (2001). *Learning and Soft Computing: Support Vector Machines, Neural Networks, and Fuzzy Logic Models*. MIT Press, Cambridge, MA, USA.

Kirkpatrick, S., Gelatt, C. D., and Vecchi, M. P. (1983). Optimization by simulated annealing. *Science*, 220(4598):671–680.

Kirsten, G. and Maier, W. (2004). Strategies for the discovery of new catalysts with combinatorial chemistry. *Applied Surface Science*, 223:87–101.

Klanner, C., Farruseng, D., Baumes, L., Mirodatos, C., and Schüth, F. (2003). How to design diverse libraries of solid catalysts? *QSAR and Combinatorial Science*, 22(7):729–736.

Klanner, C., Farruseng, D., Baumes, L., Mirodatos, C., and Schüth, F. (2004). The development of descriptors for solids: Teaching catalytic intuition to a computer. *Angewandte Chemie International Edition*, 43(40):5347–5349.

Kohonen, T. (1998). The self-organizing map. *Neurocomputing*, 21(1-3):1–6.

Kohonen, T. (2001). *Self-Organizing Maps*. Springer, 3 ed. edition.

Konstan, J., Miller, B. N., Maltz, D., Herlocker, J., Gordon, L., and Riedl, J. (1997). Grouplens: applying collaborative filtering to usenet news. *Commun. ACM*, 40(3):77–87.

Kuntz, K. W., Snapper, M. L., and Hoveyda, A. H. (199). Combinatorial catalyst discovery. *Current Opinion in Chemical Biology*, 3(3):313–319.

Lam, S. and Riedl, J. (2004). Shilling recommender systems for fun and profit. In *In Proc. the 13th international conference on World Wide Web*, pages 393–402, New York, USA. ACM.

Linden, G., Smith, B., and York, J. (2003). Amazon.com recommendations: item-to-item collaborative filtering. *IEEE Internet Computing*, 7(1):76–80.

Maes, P. (1994). Agents that reduce work and information overload. *Commun. ACM*, 37(7):30–40.

McGuiness, D., Fikes, R., Rice, J., and Wilder, S. (2003). An environment for merging and testing large ontologies. In *Proc. of the 7th International Conference on Principles of Knowledge Representation and Reasoning (KR2000)*, pages 483–493, Breckenridge, Colorado, USA.

McGuinness, D. and Wright, J. (1998). Conceptual modelling for configuration: A description logic-based approach. *Artif. Intell. Eng. Des. Anal. Manuf.*, 12(4):333–344.

Michalewicz, Z. (1992). *Genetic algorithms+data structures=evolution programs*. Springer-Verlang, New York.

Moliner, M., Serra, J., Corma, A., Argente, E., Valero, S., and Botti, V. (2005). Application of artificial neural networks to high-throughput synthesis of zeolites. *Microporous and Mesoporous Materials. Zeolits, Clays*, 78:73–81.

Montgomery, D. (2001). *Design and analysis of experiments*. John Wiley, 5th edition.

Moody, J. and Darken, J. (1989). Fast learning in networks of locally-tuned processing units. *Neural Computation*, 1(2):281–294.

Mooney, R. J. and Roy, L. (2000). Content-based book recommending using learning for text categorization. In *DL '00: Proceedings of the fifth ACM conference on Digital libraries*, pages 19–204, New York, NY, USA. ACM Press.

Mühlenbein, H. and Schlierkamp-Voosen, D. (1993). Predictive models for the breeder genetic algorithm I. Continuous parameter optimization. *Evolutionary Computation*, 1(1):25–49.

Munro, A. J. (1999). *Personal and Social Navigation of Information Space*. Springer-Verlag, London, UK.

Musen, M. (1992). Dimensions of knowledge sharing and reuse. *Computers and Biomedical Research*, 25:435–467.

Nandi, S., Badhe, Y., Lonari, J., Sridevi, U., Rao, B. S., Tambe, S. S., and Kulkarni, B. D. (2004). Hybrid process modeling and optimization strategies integrating neural networks/support vector regression and genetic algorithms: study of benzene isopropylation on hbeta catalyst. *Chemical Engineering Journal*, 97(2-3):115–129.

Nandi, S., Mukherjee, P., Tambe, S., Kumar, R., and Kulkarni, B. (2002). Reaction modeling and optimization using neural networks and genetic algorithms: Case study involving TS-1-catalyzed hydroxylation of benzene. *Industrial & Engineering Chemistry Research*, 41(9):2159–2169.

Nemhauser, G. (1966). *Introduction to Dynamic Programming*. John Wiley & Sons Inc, New York.

Oduguwa, V., Tiwary, A., and Roy, R. (2005). Evolutionary computing in manufacturing industry: and overview of recent applications. *Applied Soft Computing*, 5(3):281–299.

O'Mahony, M., Hurley, N., and Silvestre, G. (2004). Collaborative recommendation: A robutness analysis. *ACM Transactions on Internet Technology*, 4(4):377–377.

O'Mahony, M., Hurley, N., and Silvestre, G. (2005). Recommender systems: Attack types and strategies. In *In Proc. 20th National Conference on Artificial Intelligence*, pages 334–339, Pittsburgh, Pennsylvania,USA. AAAI.

O'Mahony, M., Hurley, N., and Silvestre, G. (2006). Attaking recommender systems: The cost of promotion. In *ECAI06 Workshop on Recommender systems*, pages 24–28, Riva de Garda, Italy. IOS Press.

Omata, K., Ozaki, T., Umegaki, T., Watanabe, Y., Nukui, N., and Yamada, M. (2003). Optimization of the temperature profile of a temperature gradient reactor for DME synthesis using a simple genetic algorithm assisted by a neural network. *Energy & Fuels*, 17(4):836–841.

Omata, K., Watanabe, Y., Hashimoto, M., Umegaki, T., and Yamada, M. (2004). Simultaneous optimization of preparation conditions and composition of methanol synthesis catalyst by all-encompassing calculation on an artificial neural network. *Industrial and engineering chemistry research*, 43(13):3282–3288.

Ortiz, D., Hervas, C., and Muñoz, J. (2001). Genetic algorithm with crossover based on confidence interval as an alternative to traditional nonlinear regression methods. In *9th European Symposium On Artificial Neural Networks,ESANN'2001*, pages 193–198, Bruges,Belgium.

Padberg, M. and Rinaldi, G. (1991). A branch-and-cut algorithm for the resolution of large-scale symmetric traveling salesman problems. *SIAM Rev.*, 33(1):60–100.

Pardalos, P. M. and Resende, M. G., editors (2002). *Handbook of Applied Optimization*. Oxford University Press, Inc., New York, USA.

Pereira, S., Clerc, F., Farrusseng, D., Van der Waal, J., Marchmeyer, T., and Mirodatos, C. (2005). Effect of the genetic algorithm parameters on the optimisation of heterogeneous catalysts. *QSAR & Combinatorial Science*, 24:45–57.

Powell, M. (1987). Radial basis functions for multivariable interpolation: a review. In Mason, J. and Cox, M., editors, *Algorithms for Aproximation*, pages 143–167, Oxford. Clarendon Press.

Radcliffe, N. J. (1991). Equivalence class analysis of genetic algorithms. *Complex Systems*, 5:183–205.

Rajan, K., Zaki, M., and Bennet, K. (2001). Searching techniques for structure-property relationships in materials. *Abstracts 221st National Meeting of the American Chemical Society*.

Resnick, P., Iacovou, N., Suchak, M., Bergstrom, P., and Riedl, J. (1994). Grouplens: an open architecture for collaborative filtering of netnews. In *CSCW '94: Proceedings of the 1994 ACM conference on Computer supported cooperative work*, pages 175–186, New York, NY, USA. ACM Press.

Ripley, B. (1996). *Pattern Recognition and Neural Networks*. Cambridge University Press, Cambridge.

Rodemerck, U., Baerns, M., Holena, M., and Wolf, D. (2004). Application of a genetic algorithm and a neural network for the discovery and optimization of new solid catalytic materials. *Applied Surface Science*, 223(1-3):168–174.

Rosenblatt, F. (1962). *Principles of Neurodynamics: Perceptrons and the Theory of Brain Mechanisms*. Spartan Books.

Sarwar, B., Karypis, G., Konstan, J., and Reidl, J. (2001). Item-based collaborative filtering recommendation algorithms. In *WWW '01: Proceedings of the 10th international conference on World Wide Web*, pages 285–295, New York, NY, USA. ACM Press.

Sarwar, B., Karypis, G., Konstan, J., and Riedl, J. (2000). Analysis of recommendation algorithms for e-commerce. In *EC '00: Proceedings of the 2nd ACM conference on Electronic commerce*, pages 158–167, New York, NY, USA. ACM Press.

Sarwar, B. M., Konstan, J. A., Borchers, A., Herlocker, J., Miller, B., and Riedl, J. (1998). Using filtering agents to improve prediction quality in the grouplens research collaborative filtering system. In *CSCW '98: Proceedings of the 1998 ACM conference on Computer supported cooperative work*, pages 345–354, New York, NY, USA. ACM Press.

Sasaki, M., Hamada, H., Kintaichi, Y., and Ito, T. (1995). Application of a neural network to the analysis of catalytic reactions analysis of NO decomposition over Cu/ZSM-5 zeolite. *Applied Catalysis A: General*, 132:261–270.

Sastre, G., Chica, A., and Corma, A. (2000). On the mechanisme of alkane isomerisation (isodewaxing) with unidirectional 10-member ring zeolites. A molecular dynamics and catalytic study. *Journal of Catalysis*, 195:227–236.

Schafer, J. B., Konstan, J., and J.Riedl (2001). E-commerce recommendation applications. *Data Mining and Knowledge Discovery, Springer Netherlands*, 5:115–153.

Schlierkamp-Voosen, D. (1994). Strategy adaptation by competition. In *Proc. Second Eurpean Congress on Intelligent Techniques and Soft Computing*, pages 1270–1274.

Senkan, S. (1998). High-throughput screening of solid-state catalyst libraries. *Nature*, 394(6691):350–353.

Senkan, S. (2001). Combinatorial heterogeneous catalysis - a new path in an old field. *Angewandte Chemie International Edition*, 40(2):312–329.

Senkan, S., Ozturk, S., Krantz, K., and Zengin, V. Onal, D. (1999). Discovery and optimization of heterogeneous catalysts using combinatorial chemistry. *Abstracts of Papers of the American Chemical Society*, 217:U621–U621.

Serra, J. (2004). *Catálisis Combinatoria: desarrollo de nuevas técnicas y aplicaciones de interés*. PhD thesis, Departamento de Química, Instituto de Tecnología Química,UPV-CSIC, Valencia, Spain.

Serra, J., Corma, A., Argente, E., Valero, S., and Botti, V. (2003a). Neural networks for modelling of kinetic reaction data applicable to catalyst scale up and process control and optimization in the frame of combinatorial catalysis. *Applied Catalysis A: General*, 254(1):133–145.

Serra, J., Corma, A., Chica, A., Argente, E., and Botti, V. (2003b). Can artificial neural networks help the experimentation in catalysis? *Catalysis Today*, 81(3):393–403.

Serra, J., Corma, A., Farruseng, D., Baumes, L., Mirodatos, C., Flego, C., and Perego, C. (2003c). Styrene from toluene by combinatorial catalysis. *Catalysis Today*, 81(3):425–436.

Serra, J., Corma, A., Valero, S., Argente, E., and Botti, V. (2007). Soft computing techniques applied to combinatorial catalysis: A new approach for the discovery and optimization of catalytic materials. *QSAR & Combinatorial Science*, pages 11–26.

Serra, J. M., Chica, A., and Corma, A. (2003d). Development of a low temperature light paraffin isomerization catalysts with improved resistance to water and sulphur by combinatorial methods. *Applied Catalysis A: General*, 239(1-2):35–42.

Serra, J. M., Corma, A., Farrusseng, D., Baumes, L., Mirodatos, C., Flego, C., and Perego, C. (2003e). Styrene from toluene by combinatorial catalysis. *Catalysis Today*, 81(3):425–436.

Shardnand, U. and Maes, P. (1995). Social information filetering: Algorithms for automating "word of mouth". In *Proceedings of ACM CHI'95 Conference on Human Factors in Computing Systems*, pages 210–217.

Shearin, S. and Lieberman, H. (2001). Intelligent profiling by example. In *IUI '01: Proceedings of the 6th international conference on Intelligent user interfaces*, pages 145–151, New York, NY, USA. ACM Press.

Shibata, H., Hoshiai, T., Kubota, M., and Teramoto, M. (6-7 Nov. 2002). Agent technology recommending personalized information and its evaluation. In *2nd International Workshop on Autonomous Decentralized System, 2002*, pages 176–183.

Sims, K. (1994a). Evolving 3D morphology and behaviour by competition. *Artificial Life IV Proceeding, MIT Press*, pages 28–39.

Sims, K. (1994b). Evolving virtual creatures. In *Computer Graphics, SIGGRAPH'94 Proceedings*, pages 15–222.

Smyth, B. and Cotter, P. (2000). A personalised TV listings service for the digital TV age. *Knowledge-Based Systems*, 13(2-3):53–59.

Soltysiak, S. J. and Crabtree, I. B. (1998). Automatic learning of user profiles — Towards the personalisation of agent services. *BT Technology Journal*, 16(3):110–117.

Sun, Y., Chan, B., Ramnarayanan, R., Leventry, W., Mallouk, T. E., Bare, S. R., and Willis, R. (2002). Split-pool method for synthesis of solid-state material combinatorial libraries. *Journal of Combinatorial Chemistry*, 4:569–575.

Sundaram, A., Ghosh, P., Caruthers, J. M., and Venkatasubramanian, V. (2001). Design of fuel additives using neural networks and evolutionary algorithms. *AIChE Journal*, 47(6):1387–1406.

Sywerda, G. (1989). Uniform crossover in genetic algorithms. In *Proceedings of the third international conference on Genetic algorithms*, pages 2–9, San Francisco, CA, USA. Morgan Kaufmann Publishers Inc.

Taramasso, M., Perego, G., and Notari, B. (1983). Us patent 4410501.

Terveen, L. and Hill, W. (2001). *In Carroll, J. (Ed.), HCI in the New Millennium*, chapter Human Computer Collaboration in Recommender Systems, pages 223–242. Addison Wesley.

Terveen, L., Hill, W., Amento, B., McDonald, D., and Creter, J. (1997). Phoaks: a system for sharing recommendations. *Commun. ACM*, 40(3):59–62.

Trimm, D. (1980). Design of industrial catalyst. *Elsevier*.

Ungar, L. and Foster, D. (1998). Clustering methods for collaborative filtering. In *Proceedings of the Workshop on Recommendation Systems*. AAAI Press, Menlo Park California.

Valero, S., Argente, E., Botti, V., Serra, J., and Corma, A. (2004a). A soft computing technique applied to industrial catalysis. In López de Mántaras, R. and Saitta, L., editors, *ECAI2004*, pages 765–769. ECCAI, IOS Press.

Valero, S., Argente, E., Botti, V., Serra, J., and Corma, A. (2004b). Soft computing techniques applied to catalytic reactions. *Lecture Notes in Artificial Intelligence*, 3040:550–559.

Vapnik, V., Golowich, S., and Smola, A. (1997). Support vector method for function approximation, regression estimation, and signal processing. In Mozer, M., Jordan, M., and Petsche, T., editors, *Advances in Neural Information Processing Systems 9 - Proceedings of the 1996 Conference*, volume 9, pages 281–287, Five Cambridge Center, Cambridge, MA 02142. MIT Press.

Végvári, L., Tomposb, A., Gobölös, S., and Margitfalvi, J. (2003). Holographic research strategy for catalyst library design: Description of a new powerful optimisation method. *Catalysis Today*, 81(3):517–527.

Wang, K., Wang, L., Yuan, Q., Luo, S., Yao, J., Yuan, S., Zheng, C., and Brandt, J. (2001). Construction of a generic reaction knowledge base by reaction data mining. *Journal of Molecular Graphics and Modelling*, 19(5):427–433.

Weaver, D. (2004). Applying data mining techniques to library design, lead generation and lead optimization. *Current Opinion in Chemical Biology*, 8(3):264–270.

Williams, C., Mobasher, B., Burke, R., Sandvig, J., and Bhaumik, R. (2006). Detection of ofuscated attacks in collaborative recommender systems. In *ECAI06 Workshop on Recommender systems*, pages 19–23, Riva de Garda, Italy. IOS Press.

Winston, P. (1992). *Artificial Intelligence, Third Edition*. Addison-Wesley, USA.

Wolf, D., Buyevskaya, O., and Baerns, M. (2000). An evolutionary approach in the combinatorial selection and optimization of catalytic materials. *Applied Catalysis A: General*, 200:63–77.

Wolf, D., Buyevskaya, O., and Baerns, M. (2002). European patent application 1174186.

Wright, A. H. (1990). Genetic algorithms for real parameter optimization. In *Foundations of Genetic Algorithms. First Workshop on the Foundations of Genetic Algorithms and Classifier Systems.*, pages 205–218, Los Altos, CA. Morgan Kaufmann.

Wright, A. H. (1991). Genetic algorithms for real parameter optimization. In *Foundations of Genetic Algorithms*, pages 205–218. Morgan Kaufmann.

Yamada, Y., Kobayashi, T., and Mizuno, N. (2001). Recent advances on parallel synthesis and utilization of artificial intelligence for combinatorial chemistry of solid catalysts. *Catalysts & Catalysis*, 43(5):310–315.

Zadeh, L. (1994). Fuzzy logic, neural networks and soft computing. *Communications of the ACM*, 37(3):77–84.

Zell, A., Mamier, G., and Vogt, M. (1995). *SNNS-Stuttgart Neural Network Simulator. User Manual, Version 4.1.* University of Stuttgart, Stuttgart.