Universidad
Carlos III de Madrid

TESIS DOCTORAL

Enfatizado y Diversificación en Clasificación Máquina

Autor:

Anas Ahachad

Directores:

Dr. Aníbal R. Figueiras Vidal
Dra. Lorena Álvarez Pérez

Dpto. de Teoría de la Señal y Comunicaciones

Leganés, Noviembre de 2017

TESIS DOCTORAL

Enfatizado y Diversificación en Clasificación Máquina

Autor:
Anas Ahachad

Directores:
Dr. Aníbal R. Figueiras Vidal
Dra. Lorena Álvarez Pérez

Firma del Tribunal Calificador:

Firma

Presidente:

Vocal:

Secretario:

Calificación:

Leganés, 6 de Noviembre de 2017

RESUMEN

Las excepcionales capacidades de los métodos de Boosting, especialmente del algoritmo Real AdaBoost (RAB), para resolver problemas de decisión y clasificación son universalmente conocidas. Estas buenas prestaciones provienen de la construcción progresiva de un conjunto de aprendices débiles e inestables, combinados de forma lineal, que prestan más atención a las muestras de más difícil clasificación. Sin embargo, el correspondiente énfasis que se aplica puede ser inadecuado, en particular, en casos de elevados niveles de ruido o abundante presencia de muestras fuera de margen ("outliers"). Para estos escenarios de trabajo, se han propuesto varias modificaciones del algoritmo de Boosting básico para controlar la cantidad de énfasis que se aplica, pero ninguna de estas modificaciones parece ofrecer los resultados esperados cuando se trabaja con conjuntos de datos desequilibrados, en presencia de outliers o con distribuciones de datos asimétricas.

Con esto en mente, en primer lugar, se propone en el Capítulo 2 una modificación sencilla de la función de énfasis del algoritmo RAB estándar, que no solo tiene en cuenta el error de la muestra a clasificar, sino también los errores de clasificación de las muestras más próximas a ella. A continuación, se presenta en el Capítulo 3 una generalización de la función de énfasis híbrido utilizada en versiones del algoritmo RAB que ponderan (a través de un parámetro de mezcla) las muestras según su error de clasificación y proximidad a la frontera. Esta nueva función de énfasis incluye un término constante que sirve para moderar la intensidad de énfasis, o en otras palabras, limitar la atención centrada en las muestras más próximas a la frontera o más difíciles de clasificar. Los resultados obtenidos en el Capítulo 2 y Capítulo 3 indican que estas modificaciones de las funciones de énfasis permiten alcanzar mejores prestaciones.

Posteriormente, en el Capítulo 4 se propone enfatizar los costes asociados a las muestras de entrenamiento para mejorar los resultados de clasificación de conjuntos basados en esquemas de diversificación estándar y binarización. Los resultados obtenidos en este capítulo muestran cómo las técnicas de binarización permiten que métodos de diversificación estándar (Bagging, concretamente) consigan alcanzar mejores

prestaciones; y se obtienen mejoras mucho más significativas cuando previamente se enfatizan las muestras de entrenamiento.

Esta Tesis Doctoral concluye enumerando las principales contribuciones de la misma y con una sugerencia de líneas de investigación abiertas.

ABSTRACT

The exceptional capabilities of Boosting methods, in particular of Real Adaboost (RAB) ensembles, for solving decision and classification problems are universally recognized. These capabilities come from progressively constructing unstable and weak learners that pay more attention to samples that oppose more difficulties to be correctly classified, and linearly combine them in a progressive manner. However, the corresponding emphasis can be inappropriate, in particular, when there is an intensive noise or in the presence of outliers. In this scenario, there are many proposed modifications to control the emphasis but they show limited success for imbalanced or asymmetric problems. A simple way to deal with these situations is to modify the forms of the emphasis weighting which is applied to train each new learner.

Firstly, we propose in Chapter 2 a simple modification of the well-known RAB emphasis function. The basic idea underlying this modification, which makes use of the neighborhood concept to reduce the above drawbacks, is to emphasize the samples according to their errors and those of their neighbors. Next, in Chapter 3, we propose a general form of the emphasis, which not only includes an error dependent and a proximity to the classification dependent terms, but also a constant value which serves to graduate the intensity of that mixed emphasis, limiting the increased attention which is paid to highly erroneous samples and those samples that are near to the boundary. Experimental results obtained in both Chapter 2 and Chapter 3 support the effectiveness of these forms of Boosting emphasis.

In Chapter 4, it is proposed to weight the costs associated to training examples, in order to improve the classification results of ensembles based on standard diversification and binarization techniques. Experimental results show that binarization permits that standard direct diversification methods (Bagging, in particular) achieve better results, and even more significant performance improvements are obtained when pre-emphasizing the training samples.

This Doctoral Thesis concludes enumerating its main contributions and with some suggestions of new research lines arising from this work.

AGRADECIMIENTOS

Esta Tesis Doctoral es el fruto de años de trabajo duro, de paciencia y de éxito. En este tiempo he aprendido un montón de cosas nuevas y he intentado aportar un poco. Y esto ha sido posible gracias a gente que me han acompañado, que me han dado consejos valiosos, me han animado en los momentos difíciles, y lo más importante, que han confiado en mí. Por ello, quiero transmitir mis agradecimientos a todos ellos:

En primer lugar, quiero dar las gracias a mi director de Tesis el Prof. Dr. Aníbal R. Figueiras Vidal por darme la oportunidad de formar parte de su grupo de investigación desde mi primer año de Máster. Le agradezco facilitarme todo lo necesario para desarrollar mi trabajo de Tesis dentro del departamento Teoría de la señal y comunicaciones. Le debo las gracias por su apoyo y su paciencia para guiarme durante esa aventura. Gracias por el magnífico profesor y persona que eres.

Mi agradecimiento más sincero va también dirigido a mi Co-directora de Tesis Dra. Lorena Álvarez Pérez por su dedicación, su generoso apoyo y su implicación absoluta. Gracias Lorena por todo y especialmente, por tu tiempo que me has dedicado.

A mis compañeros de laboratorios 42C01 y 42C03, que he compartido con ellos buenos momentos durante años. No puedo olvidar Ricardo, Sergio, Adil, Luis, Efrain, Juan José, Mariléa, y también los nuevos compañeros: Simón, Óscar y Alexander.

A todos los profesores del Depto. de Teoría de la Señal y Comunicaciones, gracias por su apoyo, su ayuda y su disposición, especialmente mis profesores del Máster.

A mis amigos que comparte con ellos los buenos y los duros momentos durante estos años. A Mohamad, Mohamed, Kerlos, Ahmad, Mahmoud, Dr. Yasser y Dr. Munir y sus familias por su amistad, confianza y cariño.

A mis amigos de más de media vida, Mounir, Amine, Amine, Yassine, Mohamed, Yassine, Mohamed, Youssef, Yassine, Ayoub, Zakaria, Adil, Otman, Achraf y a todos los amigos de infancia en Tánger. Muchas gracias por no dejar estar conmigo a pesar de la distancia.

A mis padres, estoy eternamente agradecido por todo el cariño que siempre me han brindado. Cuando estoy desesperado con mis problemas basta solo con recibir un llamado suyo con sus sabios consejos, para entrar en paz con mi conciencia y así dar solución a todas las dificultades que se presentan. Muchas gracias!

A toda mi familia, a mi hermano Jaber y su peque Iyad, a mi tío Mohamed por sus consejos valiosos durante mis estudios desde los primeros días en la Universidad, a mis tíos Driss, Brahim y Younes por todos los momentos inolvidables que hemos vivido allí, gracias a mis abuelos, mis tíos, mis primos y mis suegros por sus rezos y sus oraciones.

A mi amor de vida, mi esposa Khaoula, gracias por entregarme tanto cariño y amor y haberme acompañado desde la distancia, sin tu apoyo y tu paciencia habría sido más difícil. Gracias Khaoulita, nunca podré agradecerte el apoyo y amor que me has dedicado.

A todos ellos, muchas gracias.

"Quien emprende un camino buscando el conocimiento,
Allá le facilitará un camino hacía el Paraíso."

Profeta Mohammad -la paz y las bendiciones de Allá sean con él-

Índice general

Índice de figuras

Índice de tablas

Capítulo 1

Introducción

1.1. Antecedentes

Basta un instante de reflexión para aceptar que la toma de decisiones es la más frecuente actividad de nuestras vidas; no siéndonos extrañas las estimaciones. Curiosamente, sus bases probabilísticas no empezaron a sentarse hasta la Edad Moderna, y no parece necesario resaltar aquí el impulso que su posterior desarrollo ha proporcionado a ciencias (desde la física hasta la sociología, pasando por la biología y la sicología) y técnicas (sobre todo, en comunicaciones y en computación). Desde luego, es posible una visión de la toma de decisiones personales no únicamente centrada en la probabilidad; no obstante, desde las observaciones de Herbert Simon [Simon, 1957] sobre la limitación de nuestra racionalidad, no es posible que tal visión prescinda de consideraciones analíticas. Al lector curioso recomendamos (únicamente) un par de lecturas fundamentales de carácter divulgativo, que presentan posturas encontradas: la que acepta nuestro proceder [Gigerenzer, 2007] y la que advierte sobre nuestros errores [Kahneman, 2011].

En lo que se refiere a los antecedentes de la Teoría de la Probabilidad y sus muchas variantes y extensiones –variables aleatorias, procesos estocásticos, juegos de azar y estratégicos, etc.–, prescindiremos de la bibliografía por tres razones de peso.

La primera, la extrema dificultad de tener acceso a buena parte de las publicaciones de quienes la iniciaron y la pusieron en marcha. La segunda, el no menor obstáculo de entender la nomenclatura y la notación de las más antiguas. La tercera y definitiva, que las más sobresalientes de esas ideas, casi siempre de forma actualizada, se presentan en muchos libros de texto y otras publicaciones actuales.

Fueron los juegos de dados los que merecieron los primeros estudios, los de Gerolamo Cardano (1501–1576). Posteriormente –hacia finales del XVI–, llegaron las aportaciones del coloso Galileo Galilei (1564–1642), como también las hizo en el XVII Blaise Pascal (1623–1662), quien, además de notas manuscritas, se extiende sobre el tema en su correspondencia con Pierre de Fermat (1601–1665). Es Christian Huygens (1629–1695) quien publica un primer tratado, seguido por Abraham De Moivre (1667–1754). Son muy relevantes las contribuciones de Jakob (I) Bernoulli (1654-1705) en esa época. Es universalmente reconocido el carácter fundamental de la póstumamente publicada contribución de Thomas Bayes (1702–1761). También las del prodigioso Karl Friedrich Gauss (1777–1855); lamentablemente, reciben menos reconocimiento las de Pierre–Simon de Laplace (1749–1827), quien además recopiló en sus textos lo más relevante del conocimiento sobre probabilidad hasta entonces.

Ya en el siglo XX se incorpora a la probabilidad la teoría de medida de Henri–Léon Lebesgue (1875–1941), mientras que se asientan sólidamente las bases de los conocimientos actuales por bien conocidos investigadores, como el inglés Karl Pearson (1857–1936) y el estadounidense Irving Fisher (1867–1947). Irrumpe la probabilidad en la Física, sobre todo con los trabajos de Ludwig Boltzmann (1844–1906), J. Willard Gibbs (1839–1903), Werner Karl Heisenberg (1901–1976) y Erwin Schrödinger (1877–1961). En el ámbito de la Ingeniería, el tratamiento de señales (y datos) se yergue a partir de las aportaciones de Norbert Wiener (1894–1964), algunas de ellas en paralelo con las del matemático ruso Andréi Nikoláyevich Kolmogórov (1903–1987). John Von Neumann (1903–1957) construye los primeros peldaños de la Teoría de Juegos, que tanto se han estudiado en Economía.

Y así llegamos al momento de la aparición de las Máquinas de Aprendizaje.

1.2. Las Máquinas de Aprendizaje

Se acepta muy mayoritariamente que la primera Máquina de Aprendizaje, LM ("Learning Machine"), fue el clasificador binario lineal entrenado mediante la Regla del Perceptrón por Rosenblatt [Rosenblatt, 1958]. Y ello, a pesar de que son anteriores procedimientos estadísticos para tratar con observaciones como los métodos generativos paramétricos –estimando los parámetros de distribuciones que se aceptan como válidas–, el bien conocido discriminante de Fisher [Fisher, 1936], y hasta las técnicas de k vecinos más próximos (kNN), cuya base es intuitiva y que solo posteriormente se plantearon como métodos para estimar (impropiamente) densidades de probabilidad. La razón es que el Perceptrón de Rosenblatt no hacía asunciones estadísticas e incluía una regla que determinaba iterativamente los valores de los parámetros –pesos– del clasificador.

Dicha Regla del Perceptrón, de tipo hebbiano [Hebb, 1949] y para corregir errores, y que después fue el origen de toda una línea de trabajo, el Aprendizaje por Refuerzo –puede consultarse la monografía [Sutton and Barto, 1998]–, presentaba limitaciones serias: tanto que, alegando además que el Perceptrón únicamente podía construir fronteras lineales, el siempre exaltado Minsky postuló que el camino abierto carecía de salida [Minsky and Papert, 1969]; equivocadamente, como sabemos, pero con efectos devastadores, cegando dicho camino durante dos decenios.

Durante dicho intervalo de tiempo el Aprendizaje Máquina avanzó por otras rutas. En cuanto a métodos generativos, apareció el muy apreciado método no paramétrico de las ventanas de Parzen [Parzen, 1962], y algo después el algoritmo EM ("Expectation–Maximization") simplificó grandemente la parametrización de las mezclas [Dempster et al., 1977], un procedimiento generativo semi–paramétrico. También se desarrollaron los métodos basados en Reglas (Sistemas Expertos), trabajosa –y discutiblemente– elicitados por expertos humanos. Tempranas aplicaciones de las Reglas se deben a Michalski; puede consultarse [Michalski, 1980] para una presentación fundamentada. Y otras muchas aplicaciones aparecieron en la época,

incluso algunas que gozaron de elevada fama, como la de diagnóstico clínico MICYN [Shortliffe, 1976]. El transcurso del tiempo ha evidenciado las serias limitaciones de estas técnicas, pese a su cómoda apariencia para las personas.

Algo similar podría decirse de los Árboles, cuyos precedentes pueden rastrearse en [Hunt et al., 1966], muchas de cuyas elaboradas versiones, como ID3 [Quinlan, 1983], C.4.5 [Quinlan, 1993] y CART, que se puede encontrar en la buena monografía [Breiman et al., 1984], obtuvieron gran reconocimiento. Pero nos atrevemos a decir que actualmente sólo la diversificación –las Selvas Aleatorias [Breiman, 2001]– los han salvado de pasar a un discreto segundo plano.

Diferente, a nuestro juicio, es el caso de otros trabajos cuya algoritmia comparte algunos aspectos con la de Rosenblatt. Ese es el caso de los trabajos en agrupamiento y subsiguiente clasificación llevado a cabo por Kohonen [Kohonen, 1982] [Kohonen, 1995], cuya relevancia y utilidad nadie discute. Pero hemos de regresar a la línea de las máquinas tipo perceptrón si no queremos que esta exposición pierda su hilo conductor.

Entre las propuestas de Rosenblatt y la destructiva intervención de Minsky tan solo una publicación tuvo que ver con la línea que se había abierto [Nilsson, 1965]. La devastación que produjo el texto de Minsky y Papert antes citado duró hasta 1986; y ello, pese a que la posibilidad de recurrir a activaciones blandas y las primeras visiones para aplicar la regla de la cadena a un entrenamiento para minimizar un coste asociado a la salida –lo que acabó siendo el algoritmo de Retro-Propagación, BP ("Back-Propagation")–, tales como [Werbos, 1974], [Parker, 1982] y [LeCun, 1985], habían dejado expedito el camino. Pero hubo que esperar a que investigadores del proprio MIT lo dejaran explícitamente manifiesto [Rumelhart et al., 1986b] [Rumelhart et al., 1986a] para que naciesen formalmente los conocidos como Perceptrones Multicapa, MLPs por sus siglas en inglés ("Multi-Layer Perceptrons").

La arquitectura general de un MLP se muestra en la Figura 1.1. Sobre la entrada $\mathbf{x}$ se disponen sucesivas capas ocultas de unidades no lineales de tipo sigmoidal, a las que llegan las salidas de las anteriores y de valores unitarios afectadas por unos

pesos entrenables, que se suman antes de someterse a la transformación no lineal.

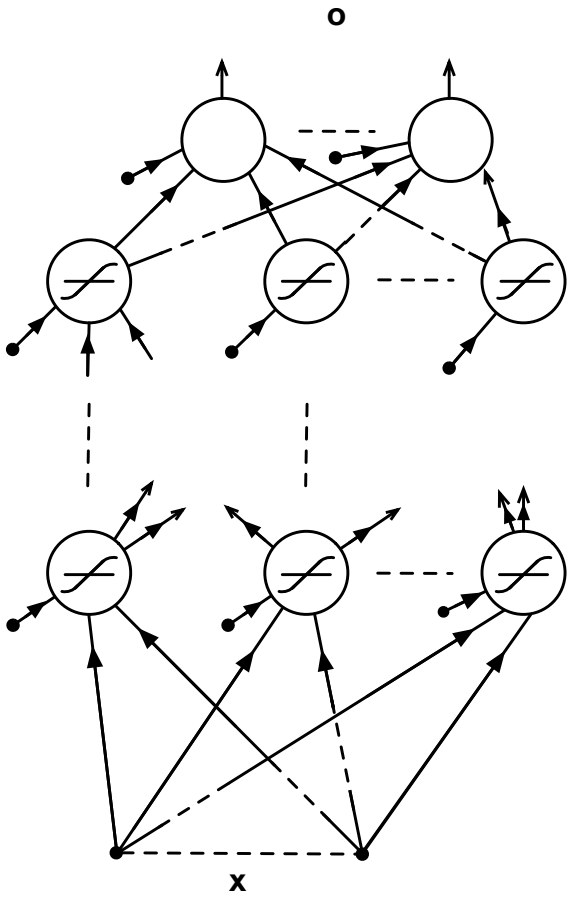


Figura 1.1: Arquitectura de un MLP.

La salida, en general un vector **o**, se obtiene a través de unidades análogas, pero lineales en el caso de estimación o regresión. Son ventajas de estas arquitecturas su condición paralelo (que ofrece la posibilidad de rápida operación) y su carácter distribuido (que implica degradación suave en caso de fallo). Pero son difíciles de dimensionar, difíciles de entrenar, y no explicativas. Por compacidad, las representaremos a lo largo de esta Tesis en la forma simplificada de la Figura 1.2, donde las flechas abiertas barradas simbolizan matrices de pesos (las flechas barradas simbolizarán vectores de pesos). Naturalmente, hay otras arquitecturas de Redes Neuronales, NNs ("Neural Networks") que compiten con los MLPs, como las de Funciones Ra-

diales de Base, RBFs ("Radial Basis Functions"), que incluyen una sola capa oculta de transformaciones radiales (como las exponenciales gaussianas), en las que se inserta una distancia entre la muestra de entrada y un "centroide" convenientemente determinado; la salida se obtiene como en un MLP, pero el entrenamiento suele ser diferente, basándose en algún método de agrupamiento: el tradicional de los MLPs es el ya citado BP, una búsqueda por gradiente convencional (naturalmente, hay variantes con bases de búsqueda más elaboradas, como gradiente conjugado o de tipo Newton). Elegir entre las diversas arquitecturas de las NNs representa una dificultad adicional, aunque hay principios generales: por ejemplo, un MLP es un aproximador global, mientras que una RBFN es un aproximador local.

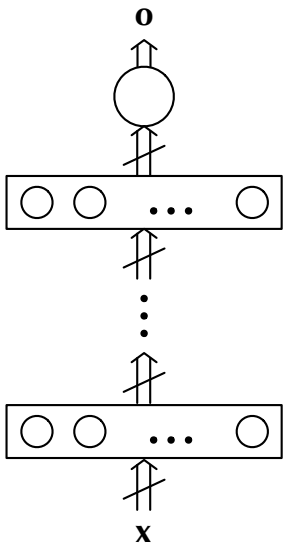


Figura 1.2: Representación esquemática de un MLP.

Pese a las dificultades y limitaciones señaladas, a partir de los trabajos de Rumelhart, Hinton y Williams que se han citado se produjo una explosión de estudios y aplicaciones de las NNs, de la que no cabe dar cuenta cierta en unas cuantas páginas. Por ello, nos limitaremos a citar unos pocos textos –seleccionados cuidadosamente– para satisfacer cualquier curiosidad, en ellos y las referencias que contienen. La selección, claro está, responde a nuestros gustos y criterios.

Dedicados casi exclusivamente a NNs están [Haykin, 1994], [Haykin, 2007], y [Bishop, 1995]. Muy recomendable es [Reed and Marks II, 1999] porque trata en detalle los aspectos prácticos que generalmente se omiten en otros textos. El manual [Arbib, 2002] facilita una primera visión de innumerables aspectos y bibliografía complementaria sobre cada uno de ellos.

Más amplio espectro ofrecen [Duda et al., 2001] y [Bishop, 2006], que incluyen otros muchos tipos de LMs, tratando todas ellas ordenada y claramente. Menos claro, pero más enciclopédico y actualizado, es [Murphy, 2012].

El desatado éxito de los MLPs –y de las NNs, en general– se debió muy probablemente a su extraordinaria capacidad expresiva, es decir, de establecer cualesquiera relaciones entre entradas y salidas. De hecho, existen demostraciones –no constructivas– de que un MLP con una sola capa oculta es un aproximador universal [Cybenko, 1989] [Hornik et al., 1989]. Pero precisamente en este aspecto radica la principal debilidad de las NNs: no se sabe cómo determinar sus parámetros no entrenables para conseguir la debida capacidad expresiva cuando ha de resolverse un determinado problema, y en muchas situaciones prácticas, el conjunto de entrenamiento (en principio, muestras etiquetadas) resulta insuficiente (de escasas muestras) para conseguir prestaciones satisfactorias. La progresiva constatación de ese punto débil disminuyó la atención a las NNs convencionales desde principios del siglo XXI.

Poco después, las dos vías para superar la limitación práctica de la capacidad expresiva –los conjuntos y las NN profundas, DNNs ("Deep Neural Networks"), que discutiremos un poco más adelante– hicieron renacer el interés por estas LMs. Entretanto, una nueva familia de LMs –con algunos aspectos comunes con las NNs y otros nacidos de nuevas perspectivas– las Máquinas de Núcleos, KMs ("Kernel Machines"), ocuparon el centro de la escena. Incluiremos aquí una muy breve revisión, puesto que no son objeto de las tareas abordadas en esta Tesis.

Las KMs más relevantes son los clasificadores conocidos como Máquinas de Vectores Soporte, SVMs ("Support Vector Machines"). Emanadas de los estudios de Vapnik [Vapnik, 1982] sobre representaciones útiles de las estructuras de los datos,

su primera formulación apareció en [Boser et al., 1992] con el empleo del primer algoritmo de Máximo Margen, MM (también referido como algoritmo SVM), cuya esencia es penalizar la proximidad de las muestras a clasificar de la frontera de decisión buscada, al tiempo que se aplica regularización y se establecen ligaduras que dan sentido al objetivo, aplicando multiplicadores de Lagrange para obtener la solución. Las fronteras no lineales se consiguen mediante la incorporación del Truco del Núcleo –véanse [Aizerman et al., 1964] [Kimeldorf and Wahba, 1971]–, que consiste en sustituir los productos escalares de muestras por un núcleo de Mercer; soslayándose así la necesidad de definir (o entrenar) transformaciones no lineales.

Han ido apareciendo muy numerosas variantes del algoritmo MM original, que ya hemos dicho que no resulta apropiado detallar aquí. Citaremos los textos que consideramos más afortunados: los más recientes de Vapnik [Vapnik, 1995], [Vapnik, 1998], el que tiene como primer autor a uno de los investigadores más activos en este ámbito [Schölkopf and Smola, 2002], y el más reciente a que ha contribuido el ya veterano Shawe–Taylor [Shawe-Taylor and Cristianini, 2004]. También es recomendable la lectura de los artículos de carácter tutorial [M ller et al., 2001] y [Shawe-Taylor and Sun, 2011].

Aunque existen modificaciones de las SVMs para aplicaciones en regresión –algunas de ellas muy elaboradas y efectivas, como [Rojo-Álvarez et al., 2004] [Rojo-Álvarez et al., 2005] y [Martínez-Ramón et al., 2006]–, los llamados Procesos Gaussianos, GPs ("Gaussian Processes"), resultan más naturales para tal tarea (y procede advertir aquí que también existen modificaciones de los GPs para clasificación, aunque la mayoría poco naturales y de prestaciones comparativamente bajas, siendo la excepción [El Jelali et al., 2009] y la subsiguiente ruta de construir blancos blandos).

Los GPs nacieron con el nombre de "kriging" en trabajos de tratamiento de señales geológicas [Cressie, 1993]. En esencia, pueden considerarse como el resultado de la aplicación del Truco del Núcleo a la formulación (lineal) del predictor transversal de Wiener: el dicho truco permite considerar espacios de observación multidimensionales y muestras tomadas irregularmente. Remitimos a la buena monografía

[Rasmussen and Williams, 2006] para ampliaciones y detalles.

1.3. Las Redes Neuronales Profundas

Las DNNs ya han sido mencionadas como una de las dos alternativas existentes para enfrentarse a los problemas de capacidad expresiva de los MLPs. La razón es que una arquitectura con varias capas ocultas parece –aunque solo sea por el número de pesos que puede incluir– una razonable alternativa a la búsqueda de un apropiado dimensionado de MLPs con una capa oculta.

La dificultad radica aquí en el propio aprendizaje: en el entrenamiento de la DNN. Es fácil comprender que los sucesivos pasos del algoritmo BP dan lugar a derivadas evanescentes, y que los problemas provocados por las parálisis –saturaciones de las activaciones sigmoidales durante el entrenamiento– acentúan esos efectos. Por ello, durante mucho tiempo no hubo avances significativos en el diseño y el empleo de esta clase de LMs.

Con dos excepciones.

Mucho antes de iniciarse la vida del algoritmo BP, se propuso –para tareas de regresión polinómica– el Método de Agrupamiento para Manejo de Datos, GMDH ("Group Method of Data Handling") [Ivakhnenko, 1968][Ivakhnenko, 1971]. Su fundamento se halla en la constatación de que combinar polinómicamente polinomios da lugar a polinomios de mayor grado, e incluso en más variables si los polinomios que se combinan las tienen distintas. De modo que Ivakhnenko propuso ajustar todos los posibles polinomios de segundo grado en dos variables como primer paso para construir su polinomio de regresión, y podar las ramas de peores prestaciones para limitar la maldición dimensional (que aquí se manifestaría como una ramificación combinatoriamente creciente); después se repite el proceso a partir de las salidas supervivientes, se detiene mediante un oportuno mecanismo de parada (resultado satisfactorio o saturación, originalmente), y se podan las ramas inútiles. La Figura 1.3 representa gráficamente este procedimiento.

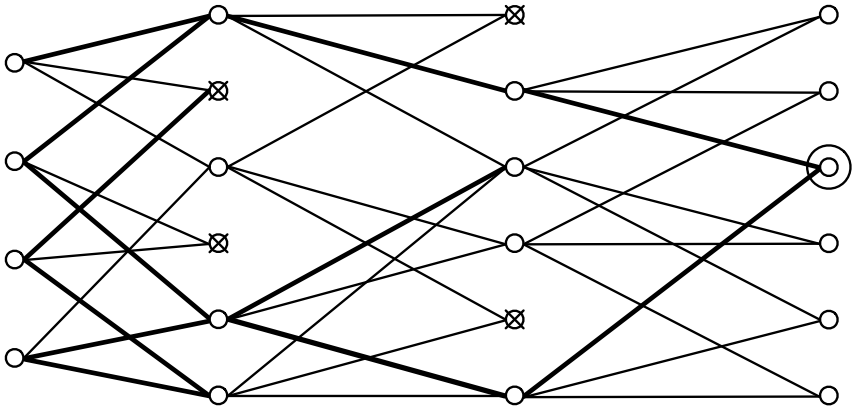


Figura 1.3: Construcción de una máquina GMDH.

El GMDH no ha conseguido un recorrido apreciable. Muy distinto es el caso del Neocognitrón [Fukushima, 1979] [Fukushima, 1980], una arquitectura tipo DNN pero convolucional –es decir, con pesos idénticos para regiones de las variables de entrada resultantes de desplazamientos con solapamiento máximo, como muestra la Figura 1.4–, que, tras cada paso, reduce la dimensión operando sobre salidas inmediatas (normalmente, tomando el máximo). El proceso se itera hasta una determinada profundidad, y se lleva a cabo en varias ramas paralelas –debido a la sensibilidad respecto a los valores de los pesos convolucionales, e incluso a su inicialización en las versiones entrenables–, que dan lugar a diversos "mapas de características", y se añade una clasificación final. Obviamente, esta clase de arquitectura será eficaz para datos espacial o temporalmente estructurados, como imágenes o habla.

Las máquinas propuestas por Fukushima estaban parametrizadas manualmente. Pero LeCun [LeCun et al., 1989] [LeCun et al., 1990] [LeCun et al., 1998] entendió, y comprobó, que la limitación de parámetros a determinar que implicaba la compartición de pesos permitía la exitosa aplicación del algoritmo BP para entrenarla. Así nacieron las muy útiles NN Convolucionales, CNNs ("Convolutional Neural Networks"), de las que diferentes versiones ostentan récords en problemas utilizados como referencia [Ciresan et al., 2012b] [Ciresan et al., 2012a] [Szegedy et al., 2014].

Mientras tanto, los intentos de entrenar DNNs generales −arquitecturas MLP con

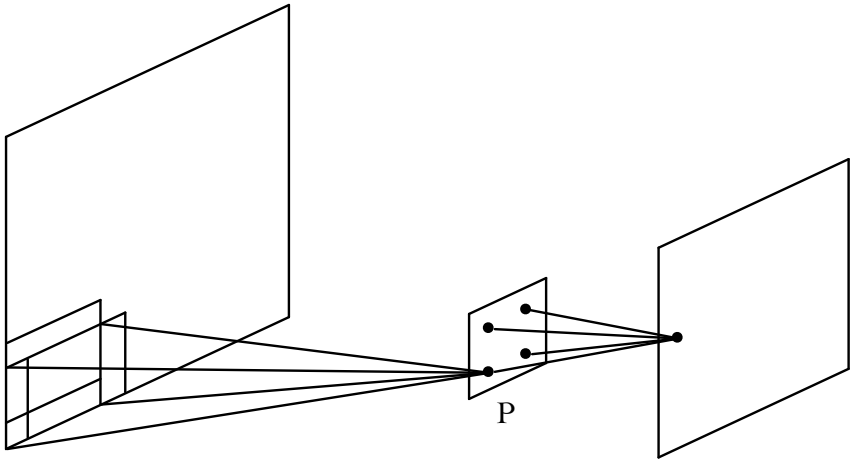


Figura 1.4: Etapa de construcción de mapas de características en una CNN (o Neocognitron). En P se funden resultados (por promedio, por máximo, ...)

varias capas ocultas− resultaban fallidos. Como se discute claramente en [Bengio, 2009], la causa de ese fracaso se halla en el desvanecimiento de las derivadas que es necesario calcular encadenadamente en el algoritmo BP: no sólo debido al creciente número de derivadas de activaciones –que podría, en principio, compensarse mediante pasos de búsqueda crecientes–, sino a los efectos de parálisis que se presentan cuando cualesquiera de esas activaciones se saturan, lo que conduce el proceso a inadecuados mínimos locales del error o coste muestral que se pretende minimizar.

Desde bien temprano [Ballard, 1987] se sugirió sortear esa dificultad mediante un primer entrenamiento dirigido a otro propósito. Si bien la Auto–Codificación, AE ("Auto–Encoding") fue el primero de los sugeridos, las versiones originales, contractivas (número de unidades en las capas ocultas menor que la dimensión de la entrada) perdían información valiosa, como se deja ver en el estudio [Rifai et al., 2011]; y, naturalmente, las expansivas conducían a la transformación identidad.

De modo que las primeras DNNs representacionales –que se entrenan en una primera fase para conseguir extraer rasgos de alto nivel representativos de las muestras, según la nomenclatura empleada en [Bengio et al., 2013]– no fueron las basadas en DAEs, sino las derivadas de las Máquinas de Boltzmann, BMs ("Boltzmann Machi-

nes"), propuestas en [Ackley et al., 1985] y estudiadas en más detalle en [Hinton and Sejnowski, 1986]. Tales BMs minimizan una "energía" ligada a las probabilidades de sus estados, mediante la aplicación de métodos MonteCarlo. En [Hinton et al., 2006] se presentaron las Máquinas de Creencias Profundas, DBNs ("Deep Belief Networks"), que posibilitaron diseños de dificultad moderada recurriendo a emplear BMs sin conexiones horizontales, las BMs Restringidas, RBMs ("Restricted Boltzmann Machines") [Smolensky, 1986], que permiten la parametrización con un sencillo muestreo de Gibbs aplicado solamente unos cuantos pasos [Carreira-Perpiñán and Hinton, 2005]; incluso un paso resulta suficiente, teniendo en cuenta que, tras el preentrenamiento, se añade una LM "llana" (no profunda) y se refina la búsqueda, orientada ya al problema a resolver. Aunque se ha trabajado mucho en esta línea, no debemos dedicar más espacio a su revisión, puesto que no guarda relación directa con los trabajos de esta Tesis; más adelante −al concluir este apartado− se proporcionarán al lector interesado referencias bibliográficas de amplias revisiones de las DNNs.

La vía de los DAEs se abrió definitivamente cuando en [Vincent et al., 2008] se recurrió a entrenar arquitecturas expansivas mediante procedimientos generativos –que habían sido propuestos por primera vez mucho tiempo atrás [Holmström and Koistinen, 1992], y que a día de hoy han dado lugar al muy interesante y amplio ámbito del Aprendizaje Ruidoso, NL ("Noisy Learning"), todavía en sus primeros pasos–, y designando como AEs Reductores de Ruido, DAEs ("Denoising Auto-Encoders") a las LMs resultantes. En [Erhan et al., 2010] se discute en detalle este tipo de DNNs.

No debemos cerrar la mención a las máquinas representacionales –las DBNs y las basadas en DAEs– sin subrayar dos aspectos: el primero, que su carácter representacional –es decir, el hecho de que proporcionen rasgos de alto nivel de las muestras de entrada– es un paso hacia el análisis de la información subyacente en las bases de datos y resulta ventajoso en algunas aplicaciones que requieren automatizar procesos. El segundo, que hay indicios de que los procesos representacionales implican un

"desenmarañamiento" de los subespacios en donde yacen las muestras, lo que supone facilitar el (posterior) tratamiento orientado a la clasificación o a la estimación.

Ha de destacarse una familia más de DNNs cuyo entrenamiento resulta viable sin medidas especiales: las llamadas Redes Apiladas Profundas, DSNs ("Deep Stacking Networks") –originalmente, se utilizó "convexas" en lugar de "apiladas"– [Deng and Yu, 2011] [Deng et al., 2012], en que se entrenan consecutivamente NNs convencionales, pero inyectando la salida de las previamente entrenadas a la entrada de la que se va a entrenar. El efecto "guía" y la profundización resultante proporcionan a estos diseños altas capacidades expresivas, y han sido aplicadas con éxito a problemas de habla y lenguaje, principalmente. Hay otros trabajos que examinan la (alta) capacidad expresiva de diversos tipos de DNNs, como [Sutskever and Hinton, 2008], [Montufar and Ay, 2011] y [Szymanski and Brendan McCane, 2014].

En literatura muy reciente han aparecido diferentes métodos que hacen factible el entrenamiento directo de DNNs: el de inicialización expuesto en [Glorot and Bengio, 2010], la optimización sin Hessianos [Martens, 2010], la introducción de parámetros adicionales para posicionamiento y escalado [Ioffe and Szegedy, 2015], ... Además de ello, muchos métodos aplicados a otras LMs se han ido incorporando a las DNNs. Es tal la velocidad de la innovación en esta disciplina que todo resumen de estado actual y tendencias se quedaría obsoleto entre la redacción de esta Tesis y su lectura; y ello se debe no sólo a la curiosidad científica, sino al fuerte impulso que están recibiendo los trabajos por parte de grandes compañías del ámbito TIC, muchas de ellas recientemente nacidas (Google, Facebook, Amazon) y otras en rápida evolución (IBM, Microsoft, Apple). Quien se interese por las DNNs tendrá necesariamente que consultar lo que está apareciendo publicado ahora mismo; una ayuda con las aplicaciones más recientes –en muchos campos y muy numerosas– la puede obtener de [Memkite, 2014]. Cumpliremos, eso sí, con lo antes prometido: revisiones recomendables (no completamente actualizadas, claro) de las DNNs son la ya citada [Bengio, 2009], y las posteriores [Deng and Yu, 2013] y [Schmidhuber, 2015].

Como única excepción a nuestra renuncia a mencionar trabajos recientes, hemos

de citar [Alvear-Sandoval and Figueiras-Vidal, 2015], [Alvear-Sandoval and Figueiras-Vidal, 2016], [Alvear-Sandoval and Figueiras-Vidal, 2018] y [Alvear-Sandoval et al., 2017] porque en estas comunicaciones y artículos se trata, en el ámbito de las DNNs –y concretamente en el de las basadas en DAEs–, de estudiar la eficacia de utilizar métodos de binarización, diversificación y pre-énfasis: de forma tal que, como expondremos más adelante, se convierte en interesante analizar si combinaciones de dichos métodos válidas para las ML profundas estudiadas son también útiles para ML convencionales –"llanas"–, lo que constituirá uno de los objetivos de esta Tesis.

1.4. Conjuntos de LMs

Produce no pequeña sorpresa que, habiendo sido simultánea la emergencia de las propuestas de crear conjuntos de LMs por vías indirectas –es decir, no recurriendo a un simple aprendizaje de los parámetros entrenables– con el desarrollo efectivo de las KMs, resultasen éstas merecedoras de más atención durante muchos años, hasta llegar casi al momento presente. Y ello, porque la experiencia demuestra que, en una abrumadora mayoría de casos, las soluciones que brindan los conjuntos de LMs son superiores a las que ofrecen las KMs. Tal vez la causa esté que se considera que las formulaciones de las KMs son analíticamente más sólidas que las que dan lugar a la mayoría de los conjuntos, o que el origen de las primeras –la dimensión VC, como se ha dicho– precedió al de los segundos –la teoría de lo Probablemente Casi Correcto, PAC ("Probably Almost Correct"), formalizada en [Valiant, 1984], que aboga por construir clasificadores fuertes agrupando clasificadores débiles, se toma como el nacimiento del concepto de conjuntos de LMs–. Lo cierto es que los antecedentes de los conjuntos son muy anteriores: aceptar como el primero el famoso Teorema del Jurado de Marie–Jean–Antoine–Nicolas de Caritat, marqués de Condorcet, en un ensayo sobre las decisiones por mayoría publicado en 1785, no puede calificarse de especulación; teniendo además en cuenta que las LMs surgieron a partir de conceptos biológicos, la sorpresa se acrecienta: mientras la optimización vía algoritmos de ese

tipo (Genéticos, Evolutivos, de Enjambres, etc.) irrumpía con fuerza, los conjuntos de LMs no conseguían una imagen similar. Para colmo, el primer artículo dedicado a su análisis [Hansen and Salamon, 1990] es precisamente un cuidado estudio estadístico.

Sea como fuese, estamos hablando de una arquitectura compuesta tal y como la mostrada en la Figura 1.5, en la que las unidades $\{U_m\}$ aprenden a resolver el problema de formas diversas, y sus salidas se agregan mediante el elemento AG para proporcionar la salida global o (que se supone unidimensional, por claridad).

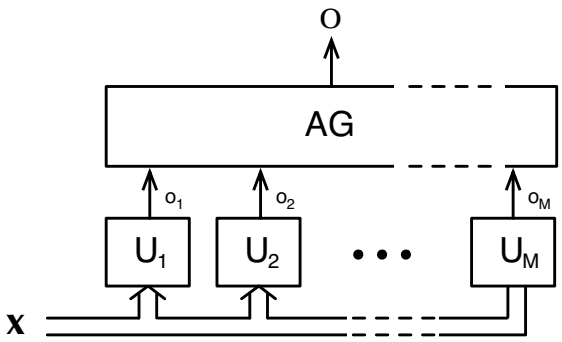


Figura 1.5: Representación esquemática general de un conjunto de LMs, las unidades $\{U_m\}$. **x** es la entrada, $\{o_m\}$ las salidas de las unidades, AG la agregación, y o la salida global.

La clave del éxito en el diseño de conjuntos de LMs es la diversificación, i.e., la no coincidencia de funcionamiento de las unidades –para que la agregación pueda sacar beneficio–. Hay dos tipos de procesos para conseguirla: forzarla en las unidades y proceder después a la agregación, dando lugar a los llamados comités, o inducirla en la construcción conjunta de unidades y agregación, que es el caso de lo que nosotros denominamos consorcios.

Los más conocidos tipos de comités vieron la luz a mediados y finales de los 1990, siguiendo el intento de apilar capas llamado "Stacking" [Wolpert, 1992], de aplicabilidad y éxito limitados. Curiosamente, aunque la diversificación puede obtenerse mediante arquitecturas o entrenamientos distintos, casi todos los trabajos recurrieron en diversificar la información que se empleaba para entrenar las diversas unidades. Así

ocurre con dos destacables aportaciones del ya fallecido Leo Breiman –investigador de sobresaliente mérito–, el "Bagging" ("Bootstrap and aggregating") [Breiman, 1996], en que las unidades se diseñan con remuestreos "Bootstrap" del conjunto de muestras disponible para ello, y el hoy conocido como "Switching" (aleatorización de las salidas, originalmente) [Breiman, 2000], en que las etiquetas que ve cada unidad en el aprendizaje se han sometido a una aleatorización. Igualmente ocurre con otra propuesta del mismo autor, las Selvas Aleatorias, RF ("Random Forests") [Breiman, 2001], árboles en que las ramificaciones se llevan a cabo probabilísticamente (con posteriores adiciones y mejoras, como las presentadas en [Archer and Kimes, 2008]). Obviamente, las unidades o aprendices han de ser inestables –sensibles a la información de entrenamiento– para que estos diseños den resultado; lo son no sólo los árboles, sino también las NNs. Y, aunque haremos uso de algunos de estos comités en los trabajos de esta tesis, "Bagging" y "Switching", lo sencillo de su concepción no hace necesario extenderse más. Sólo queda añadir que la agregación en los comités es inexcusablemente no entrenable –voto por mayoría o promedio aritmético simple–, porque la diversificación separada previa impone el empleo de muchos aprendices, de modo que no resulta efectivo intentar agregaciones entrenables (además, puede haber notable similitud entre las salidas de las unidades).

Al contrario que el de los comités, el diseño de los consorcios procede a la vez con la diversificación y la agregación, bien en modo bloque, bien secuencialmente, como vamos a ver acto seguido. De ese modo se alcanzan ventajas en las prestaciones con respecto a los conjuntos recién examinados.

Antes de centrarnos en la familia de consorcios de mayor importancia (tanto general como en esta Tesis), mencionaremos brevemente otras dos que no deben ser olvidadas. Una, las Mezclas de Expertos, MoEs ("Mixtures of Experts"), aparecidas por primera vez en [Jacobs et al., 1991] (con algún defecto formal).

La Figura 1.6 presenta (corregida) la estructura básica de dicha versión inicial (para regresión): las salidas de los expertos (aprendices), combinaciones lineales extendidas $o_{\mathrm{m}}(\mathbf{x}) = \mathbf{w}_{\mathrm{me}}^{\mathrm{T}}\mathbf{x}_{\mathrm{e}}$, se multiplican por las salidas de una puerta

de estructura análoga salvo porque concluye en una activación softmax, $g_m(\mathbf{x}) = \exp\left(\mathbf{w}_{\mathrm{gme}}^{\mathrm{T}}\mathbf{x}_{\mathrm{e}}\right) \Big/ \sum_{m'} \exp\left(\mathbf{w}_{\mathrm{gm'e}}^{\mathrm{T}}\mathbf{x}_{\mathrm{e}}\right)$, y los resultados se suman para obtener la salida global

$$\mathrm{o}(\mathbf{x}) = \sum_{\mathrm{m}} \mathrm{g_m}(\mathbf{x})\mathbf{w}_{\mathrm{me}}^{\mathrm{T}}\mathbf{x}_{\mathrm{e}} \tag{1.1}$$

que tiene una forma paralela a la del estimador mmse para un modelo de mezcla de Gaussianas

$$\widehat{\mathrm{s}}(\mathbf{x}) = \sum_{\mathrm{m}} \frac{\mathrm{p_m(x)}}{\sum\limits_{\mathrm{m'}} \lambda_{\mathrm{m}}\, \mathrm{p_m(x)}} \mathbf{w}_{\mathrm{me}}^{\mathrm{T}}\mathbf{x}_{\mathrm{e}} \tag{1.2}$$

ocupando $\mathrm{g_m}(\mathbf{x})$ el lugar del coeficiente m de la combinación convexa variable que aparece en (1.2), que sirve para focalizar el efecto de $\mathbf{w}_{\mathrm{me}}^{\mathrm{T}}\mathbf{x}_{\mathrm{e}}$ en la región apropiada del espacio de observaciones. La aproximación puede entrenarse por Máxima Verosimilitud (ML), o variantes como el algoritmo de Esperanza−Maximización, EM ("Expectation Maximization") [Dempster et al., 1977].

Las limitaciones intrínsecas de esta arquitectura se asemejan a las propias de los modelos de mezclas de Gaussianas: surgen del carácter lineal de los expertos y de la propia puerta. Para solventarlas, se han propuesto muchas generalizaciones de expertos y puertas, que crean dificultades en los procesos de aprendizaje, así como disposiciones jerárquicas [Jordan and Jacobs, 1994] [Jordan and Xu, 1995], de diseño más asequible. Como también hay otras muchas generalizaciones y propuestas de mejora, remitimos al trabajo tutorial [Yuksel et al., 2012]. Pero hemos de decir que las extensiones de las formulaciones básicas para problemas de clasificación proporcionan resultados no particularmente destacables; sin embargo, una sencilla reorganización arquitectónica permite aplicar algoritmos MM y conseguir excelentes prestaciones [Omari and Figueiras-Vidal, 2013], así como abre el camino a la aplicación de un tipo de técnicas de post-agregación [Omari and Figueiras-Vidal, 2015], si bien todo ello a costa de elevadas cargas computacional en diseño.

Otra familia interesante de consorcios, que se apoya en las ideas para forzar diversificación de las salidas de los aprendices apuntados en [Fahlman and Lebiere, 1990],

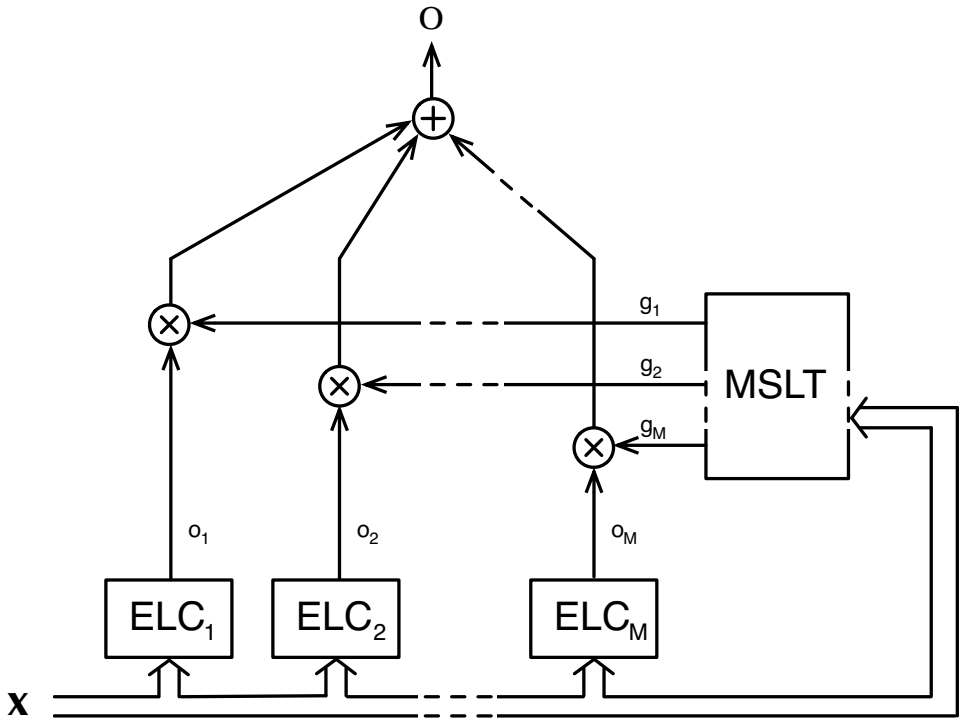


Figura 1.6: Versión original de MoE. $\{\mathrm{ELC_m}\}$ son los expertos (aprendices), combinadores lineales extendidos, con salidas $\{\mathrm{o_m}\}$, que se multiplican una a una por $\{\mathrm{g_m}\}$, las M salidas "softmax" alimentadas también por transformaciones lineales extendidas de la entrada $\mathbf{x}$.

son los conjuntos con Aprendizaje por Correlación Negativa, NCL ("Negative Correlation Learning"), introducidos en [Liu and Yao, 1999a], [Liu and Yao, 1999b]. El calificativo "interesante" se emplea aquí intencionadamente: estos consorcios, cuya construcción se fundamenta en añadir al coste a minimizar un término que penaliza la cercanía de los valores de salida de los aprendices, no ofrecen resultados completamente satisfactorios; pero la idea de diversificar las salidas sí merece una exploración más intensa que las llevadas a cabo hasta ahora.

Y llegamos ahora a los más famosos de los conjuntos de LMs: los basados en "Boosting". A ellos dedicaremos un mayor espacio, porque van a ser elemento importante en buena parte de los trabajos de esta Tesis.

Las técnicas de "Boosting" se desarrollan a partir de la teoría PAC ya menciona-

da: así, una primera versión empleaba aprendices débiles que iban siendo entrenados con una selección de muestras que dependía de las prestaciones de los aprendices previamente parametrizados [Schapire, 1990]. A ella siguieron las dos fundamentales, basadas en entrenar consecutivamente aprendices débiles con todas las muestras, pero ponderadas según la calidad de las correspondientes salidas agregadas de los aprendices previamente diseñados, tal y como se muestra en la Figura 1.7.

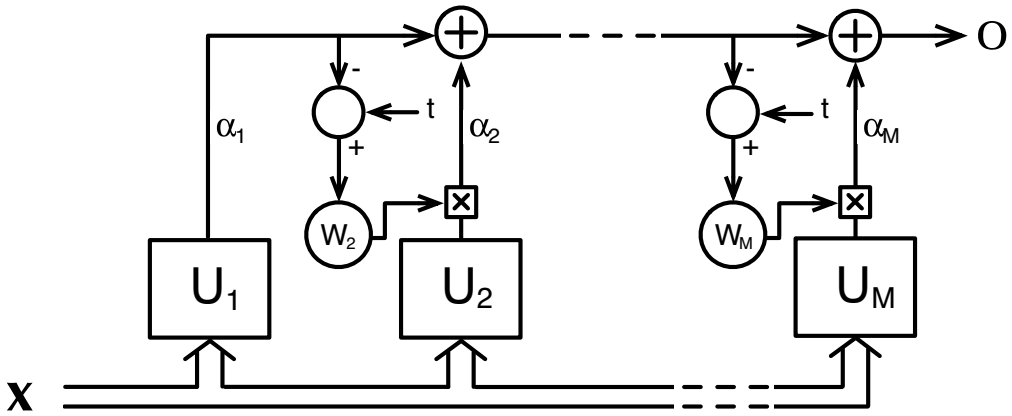


Figura 1.7: Entrenamiento tipo "Boosting". Los costes en el entrenamiento de cada aprendiz se ponderan de acuerdo con las reglas $\{W_m\}$ que se extraen de los errores de la agregación de salidas anteriores (ponderadas por los pesos $\{\alpha_m\}$) respecto a las etiquetas, t.

La primera forma de las representadas en la Figura 1.7 utilizaba aprendices con salidas duras, $[-1, 1]$; junto con un coste de tipo exponencial del margen, $\exp(-to)$, daba lugar a un algoritmo de entrenamiento secuencial cerrado, que se denominó "AdaBoost", AB ("Adaptive Boosting") [Freund and Schapire, 1995] [Freund and Schapire, 1996a] [Freund and Schapire, 1997]. Inmediatamente, los autores extendieron la formulación al caso de aprendices con salidas blandas, en $[-1, 1]$ [Freund and Schapire, 1996b], [Schapire and Singer, 1998], [Schapire and Singer, 1999], recurriendo para ello a la minimización de una cota del coste de entrenamiento: el "Real AdaBoost", RAB. Como quiera que el RAB es elemento central en buena parte de estos trabajos y para mayor comodidad del lector, resumimos su formulación en el

Apéndice A.

Tras estos inicios, y beneficiándose de su solidez, se presentó una notable cantidad de variantes y extensiones, buena parte de ellos apoyados en el brillante hallazgo de –otra vez– Leo Breiman [Breiman, 1999a] –de este artículo volveremos a hablar en breve–, quien señaló que el "Boosting" y ciertas variantes podían formularse como una búsqueda por gradiente. Entonces fue cuando se propuso el "AnyBoost" y se desarrollaron algunos casos particulares de ese algoritmo general [Mason et al., 2000b], mientras Jerome Friedman proponía algoritmos que se basan en la regresión logística [Friedman et al., 2000] [Friedman, 2001]. Más recientemente, incluso se emplearon agregaciones locales [Mayhua-López et al., 2012], o se debilitaron mediante submuestreo estratificado SVMs al objeto de utilizarlas como aprendices [Mayhua-López et al., 2014]. Pero lo que más se debe destacar aquí son los estudios de Leo Breiman –¿quién si no?– [Breiman, 1998] [Breiman, 1999a] y [Breiman, 1999b], que dejan claro, por vía experimental, que la forma particular que se emplee para enfatizar las muestras no es tan relevante como el proceso, propiamente dicho, de enfatizar y agregar, y que énfasis distintos dan lugar a resultados tan solo ligeramente mejores o peores, dependiendo del problema bajo análisis; devaluando así la conjetura de que era la presencia de una función de margen en el coste del "Boosting" la causa de las excepcionales prestaciones de esta familia de algoritmos. Por eso dio Breiman en hablar de "Arcing" ("Adaptive resampling and combining") como término para designar genéricamente a la familia completa; pese a ello, se mantiene "Boosting" para tal misión.

Una de las propiedades de los algoritmos de "Boosting" que han causado más sorpresa –y no poco contento– es su resistencia intrínseca al sobreajuste, que fue observada desde el principio en experimentos llevados a cabo por muchos investigadores, como los mismos creadores de estas ideas [Drucker et al., 1993] (basándose en versiones informales) y [Schapire et al., 1998] –en donde atribuyen alegremente esa resistencia al empleo del margen–, y también otros [Drucker et al., 1994], [LeCun et al., 1995] y [Schwenk and Bengio, 1997]. Pero otros autores encontraron casos

en que sí se producía sobreajuste; algunos de ellos son [Quinlan, 1996], [Bauer and Kohavi, 1999], [Dietterich, 1998], [Dietterich, 2000], el propio Breiman [Breiman, 1999a], [Rätsch et al., 1999], [Mason et al., 2000a] y [Rätsch et al., 2001]. La mayoría de los citados perciben que el sobreajuste aparece en algunas situaciones: cuando para resolver el problema que se considera se dispone de datos muy ruidosos; cuando hay un número apreciable de muestras fuera de margen ("outliers"); y cuando los ejemplos de entrenamiento incluyen frecuentes volteos de etiquetas. De ello concluimos nosotros que el sobreajuste –la limitación principal del "Boosting"– se debe a la presencia de un apreciable número de muestras del lado incorrecto de la frontera; por tanto, enfatizar únicamente de acuerdo con el error no es la opción más prudente –como ya se demostró en [Gómez-Verdejo et al., 2006] y [Gómez-Verdejo et al., 2008], trabajos en los cuales se proponían énfasis mixtos que incluían la proximidad a la frontera junto con el error–; aquí se originan, como pronto expondremos, líneas de trabajo que se abordan en la presente Tesis.

Como era de esperar, se han propuesto muchos y variados procedimientos para combatir la aparición de sobreajuste en "Boosting": la pura eliminación de muestras perturbadoras [Freund, 2001], la regularización [Rätsch et al., 1999], procedimientos basados en modificar los márgenes [Mason et al., 2000a], [Rätsch et al., 2001], [Rätsch and Warmuth, 2005], [Shen and Li, 2010], los énfasis mixtos [Gómez-Verdejo et al., 2006], [Gómez-Verdejo et al., 2008], la penalización de la asimetría de los datos [Sun et al., 2006] y el submuestreo [Zhang et al., 2008], por citar los más relevantes. En esta tesis se extenderá la exploración de una de esas vías: la relativa a modificaciones del enfatizado.

[Schapire and Freund, 2012] es una extensa revisión de los métodos de "Boosting".

Una naturaleza completamente distinta a la diversidad (informacional, sustancialmente) que se acaba de revisar es la proveniente de la binarización de problemas multiclase.

Cuando han de asignarse muestras a más de dos clases, las dificultades crecen (de lo que no cabe extrañarse, puesto que a las personas nos ocurre igual). Ha de

admitirse que un problema multiclase es intrínsecamente más complicado que uno binario: resulta necesario establecer un número mayor de fronteras, cuyas formas resultan poco intuitivas y difícilmente imaginables.

Por lo interior –y plausiblemente como reflejo de nuestra propia forma de proceder, aunque se difundiesen incomprensiblemente tarde desde la emergencia de las LMs para clasificación– aparecieron los métodos de binarización. Los más simples, el Uno contra Uno, OvO ("One versus One") [Hastie and Tibshirani, 1998], en que se construyen todas las posibles dicotomías de una clase contra otra, $C(C-1)/2$ si C es el número de clases, y se declara el triunfo de la clase con más victorias; y el Uno contra el Resto, OvR ("One versus Rest"), en el que cada clase se enfrenta a todas las demás, y se elige la que mayor éxito consiga en esos enfrentamientos. Ambos tienen inconvenientes: el alto número de aprendices en el OvO, el desequilibrio de los enfrentamientos en el OvR. Ni siquiera está claro cuál de los dos métodos ofrece mejores prestaciones: la respuesta parece depender del problema considerado [Allwein et al., 2000].

Sí queda claro que hay alternativas de binarización mejores que las anteriores: las derivadas de la utilización de los códigos correctores de errores (largamente empleados en comunicaciones digitales), y conocidos en ML como Códigos de Salida Correctores de Errores, ECOCs ("Error Correcting Output Codes"). A cada clase se le asigna una palabra código, y el conjunto de bits en iguales posiciones definen las dicotomías a resolver. Recuérdese que, si las palabras código presentan entre sí una distancia de Hamming D, la aparición de hasta $E[(D+1)/2]-1$ errores en una de ellas permite su correcta identificación; lo que significa hacer posible que algunos problemas binarios se resuelvan incorrectamente. Por tanto, la construcción de un ECOC ha de perseguir códigos con altas distancias de Hamming manteniendo una prudente longitud (ya que ésta indica el número de máquinas que compondrán el conjunto necesario para aplicarlo); a la vez, debe cuidarse que los bits en iguales posiciones den lugar a problemas equilibrados.

Curiosamente, una versión primitiva de ECOC apareció antes [Sejnowski and

Rosenberg, 1987] que las ideas de emplear OvO y OvR, y [Dietterich and Bakiri, 1995] es una temprana revisión de sus posibilidades. A ella y a la monografía [Rokach, 2010] remitimos al lector para informarse más en detalle, puesto que en esta tesis será suficiente recurrir a los códigos de Hadamard –recomendados en [Dietterich and Bakiri, 1995] hasta para 7 clases, posiblemente por las limitaciones computacionales de la época; aunque es cierto que su longitud crece exponencialmente con el número de clases.

Los códigos de Hadamard son de construcción elemental. Para C clases, la primera palabra código consta de $2^{C-1} - 1$ "1"s; las siguientes van alterando tiras de 2^{C-p} "0"s y "1"s, siendo p el número de la palabra. La Tabla 1.1 muestra un ejemplo sencillo, para C = 4. Puede en ella comprobarse que D = 4: se corrige hasta un error en las clasificaciones binarias.

	Pr1	Pr2	Pr3	Pr4	Pr5	Pr6	Pr7
C_1	1	1	1	1	1	1	1
C_2	0	0	0	0	1	1	1
C_3	0	0	1	1	0	0	1
C_4	0	1	0	1	0	1	0

Tabla 1.1: ECOC de Hadamard para 4 clases. C_c: clase; Prm: dicotomía.

Pondremos fin a este apartado completando la bibliografía de monografías dedicadas a los conjuntos de LMs: además de [Schapire and Freund, 2012] y [Rokach, 2010], son interesantes [Sharkey, 1999], [Kuncheva, 2004], [Zhang and Ma, 2012] y [Zhou, 2012].

1.5. Las técnicas de pre-énfasis

La razón para la pronta aparición de estas técnicas –su primera aparición, en [Hart, 1968], se dirigía a clasificación con k vecinos más próximos, k-NN ("k-Nearest

Neighbours"), antecediendo en muchos años a las LMs actuales– se halla en la comprensión de que no todos los ejemplos disponibles tienen la misma importancia para diseñar un clasificador: los más difíciles de clasificar y los más cercanos a una razonable frontera tienen más utilidad para ello que el resto. Obsérvese que hay una relación conceptual evidente con los procesos de "Boosting", y en particular con los que proponen énfasis mixto [Gómez-Verdejo et al., 2006], [Gómez-Verdejo et al., 2008]: la diferencia radica en que el pre-énfasis –que indicaremos en lo que sigue mediante PrE– se incluye en el propio diseño del clasificador, o bien hace uso de los resultados de un clasificador auxiliar o guía para el diseño del clasificador final (que, obviamente, no tiene por qué ser débil, como se requiere en "Boosting"). En todo caso, de esta discusión surgirá también la posibilidad de aplicar formas generales y flexibles de énfasis a conjuntos diseñados mediante procesos de "Boosting".

Las versiones iniciales de PrE consistían en una selección de muestras, bien de acuerdo con su proximidad a la frontera (ocasionalmente, a la frontera de la guía), como en [Sklansky and Michelotti, 1980], [Plutowski and White, 1993] y [Choi and Rockett, 2002], o según la dificultad de clasificación (dada por el error u otra forma de coste), tal como en [Munro, 1992] y [Cachin, 1994].

Pero de los experimentos de [Franco and Cannas, 2000] se pudo concluir que la importancia relativa de ambas medidas resultaba dependiente del problema que se buscaba resolver. Por ello, en [El Jelali et al., 2008a], [El Jelali et al., 2008b] y [El Jelali et al., 2009] se propuso la construcción de "blancos blandos", transformando las etiquetas ("duras") en dichos blancos mediante la consideración de ambas medidas, y siendo esos blancos más exigentes (mayores, en valor absoluto) para las muestras más problemáticas. Estas técnicas se aplicaron a MLPs y a GMMs, pero resultaron especialmente útiles en el caso de GPs, ya que permitían su diseño siguiendo el esquema natural de regresión, evitando las aproximaciones para adaptarlo a la clasificación. Otra versión de "blancos blandos" se expone en [Mora-Jiménez and Figueiras-Vidal, 2009].

Fue a partir del anterior concepto de "blancos blandos" como se propusieron for-

mas generales de PrE apoyadas en el funcionamiento de guías para probar su eficacia en la mejora de prestaciones de DLs diversificadas [Alvear-Sandoval and Figueiras-Vidal, 2016] [Alvear-Sandoval and Figueiras-Vidal, 2018]. Recurriendo a formas que incluyen un término constante en combinación convexa con la combinación convexa de medidas del error y de la proximidad a la frontera de cada muestra, los resultados fueron muy favorables.

Para evitar cargar de formulación las páginas de esta Introducción, se presentan y discuten esas formas de PrE en el Apéndice B. Pero aquí hay que resaltar que:

1) El éxito obtenido con las DLs combinando binarización, diversidad y PrE justifica investigar si en MLs convencionales se pueden obtener análogas ventajas. Es cierto que hay estudios parciales que examinan el potencial de aunar binarización y diversificación [Leisch and Hornik, 1997], [Valdovinos and Sanchez, 2006]; pero son incompletos y de resultados no concluyentes; además, no consideran el PrE.

2) Las formas de PrE que se consideran pueden resultar útiles como énfasis para "Boosting", y en particular para "Real AdaBoost", y siguiendo la estela los aquí repetidamente citados trabajos sobre énfasis mixto: nótese que la presencia de un término moderador –el constante– en esas formas puede contribuir a combatir posibles tendencias al sobreajuste, que, como se ha visto, propende a aparecer cuando muchos ejemplos de entrenamiento resultan (claramente) mal clasificados; al tiempo, puede evitar que sea excesiva la desatención a las muestras claramente bien clasificadas.

1.6. Orientación, objetivos y contenido de la Tesis

La orientación de esta Tesis ya se ha podido vislumbrar en las páginas que anteceden: se dirige al análisis de procedimientos de énfasis, cuyo objeto es facilitar el aprendizaje de LMs para clasificación, y, por consiguiente, obtener de ellas mejores prestaciones. Y se considerarán tanto el enfatizado en algoritmos de tipo "Boosting" –en particular, "Real AdaBoost", y con aprendices MLPs– para construir conjuntos de clasificadores, como el de máquinas monolíticas, a partir del comportamiento de clasificadores auxiliares o guías, es decir, el pre-énfasis.

En el primero de los casos, se sabe que las altas prestaciones de los conjuntos para "Boosting" se pueden ver afectadas por la aparición del fenómeno de sobreajuste, que se da cuando el problema que se está tratando corresponde a un ambiente muy ruidoso, o con alta presencia de muestras fuera de margen ("outliers"), o también con frecuentes volteos de etiqueta: es decir, en situaciones en que un número apreciable de ejemplos resultan, necesariamente, mal clasificados. Y nosotros creemos que también se producirá ese sobreajuste cuando se trate de un problema en que las poblaciones muestrales se solapen en alto grado –"difícil", en el sentido de baja separabilidad potencial–, ya que también se producirán en ellas muchas clasificaciones incorrectas, de modo que el recurso a emplear una medida del error para el enfatizado entrañará el riesgo de sobreajuste.

Se estudiarán dos formas distintas de limitar el riesgo citado.

La primera, mediante un enfatizado que incluya el error de las muestras vecinas además del error en cada ejemplo a enfatizar; de manera que si éste se encuentra en el lado indebido de la frontera, la mayoría de esas muestras vecinas –que pertenecerán, en problemas con suficiente información, a la clase correcta– contribuyan a moderar el énfasis. En este algoritmo, absolutamente original, tanto el número de las muestras vecinas a considerar como el peso relativo de sus errores en la determinación del valor del énfasis para el ejemplo correspondiente se explorarán y determinarán mediante sencillos procesos de validación cruzada, CV ("Cross Validation").

La segunda, mediante el empleo, dentro de un algoritmo del tipo "Real AdaBoost", de la estructura del énfasis que se ha visto que proporcionaba notable ventaja aplicado como pre-énfasis a DAECs y DAECs binarizados y/o diversificados, cuyas referencias aparecen en el apartado 1.5 de este mismo capítulo. Insistimos en que la combinación de medidas de error y de proximidad a la frontera –que ya se ha mostrado eficaz– seguida de la combinación del resultado con un término constante que actúe de moderador del énfasis y, por tanto, pueda evitar excesos de atención a unos ejemplos en detrimento de la que se presta a otros. Los parámetros de ambas combinaciones (convexas) también se determinarán mediante CV.

Además de estos dos estudios, se llevará a cabo un tercero, siempre dentro del ámbito del enfatizado de ejemplos de entrenamiento. El objetivo de este estudio es verificar que la combinación del PrE con la binarización y diversificación produce en el caso de aprendices "llanos" –MLPs de una capa oculta– beneficios comparables a los observados en los estudios recién aludidos cuando se recurre a DAECs. Implícitamente, los experimentos que se muestran arrojan más luz sobre un aspecto insuficientemente estudiado con anterioridad: las relaciones entre binarización y diversificación cuando se aplican a estas arquitecturas. Y no debe dejar de resaltarse que el PrE es un mecanismo de coste computacional comparativamente reducido: si produce ventaja, su empleo está, pues, más que justificado.

De acuerdo con lo anterior se suceden los siguientes capítulos de esta tesis.

En el Capítulo 2, se detalla el algoritmo de énfasis "local" que se ha descrito cualitativamente, y se diseña un buen número de experimentos –con bases de datos que han sido empleadas en trabajos análogos, y que sirven, pues, de referencia–, cuyos resultados se presentan y discuten, comparándolos con los ofrecidos por algoritmos "no locales" equivalentes a los considerados. Las conclusiones correspondientes ponen fin al capítulo.

El Capítulo 3 se dedica a examinar las prestaciones del énfasis general que se presenta en el Apéndice B en algoritmos "Real AdaBoost" con aprendices MLPs de una capa oculta y débiles –cuya dimensión oculta se determina mediante CV,

junto con los parámetros de la doble combinación convexa del énfasis. Los resultados –también para un buen número de bases de datos– se valoran en comparación con los que se obtienen con versiones reducidas del mismo tipo de énfasis, con objeto de comprobar cuál es el papel de cada término del repetido énfasis general. Y se extraen las conclusiones más relevantes de dichas comparaciones.

El estudio de las combinaciones de PrE, binarización y diversificación, en arquitecturas "llanas" se expone en el Capítulo 4. El PrE es el general utilizado para "Real AdaBoost" en el capítulo anterior; los problemas seleccionados son multiclase, pero con un número moderado de clases –se rehuye una carga computacional excesiva, dado que el propósito del estudio es verificar, y no establecer taxonomías–, lo que permite sencillos ECOCs tipo Hadamard; la diversificación se limita a "Bagging", por tratarse de una algoritmia relativamente simple pero razonablemente representativa. Los experimentos se llevan a cabo con todas las combinaciones de los tres ingredientes que merecen interés, y sus resultados se discuten comparativamente; cerrando el capítulo con las correspondientes conclusiones.

Finalmente, el último capítulo destaca las aportaciones de la tesis, resalta las principales conclusiones que se obtienen, y propone algunas direcciones, abiertas por estos trabajos, para posterior investigación.

Capítulo 2

Real AdaBoost con función de énfasis suavizada

2.1. Introducción

Tal y como se comentó en el Capítulo 1, la más sorprendente propiedad tanto de AdaBoost (AB) como de Real AdaBoost (RAB) es su notable resistencia a la nociva tendencia al sobreajuste, lo que permite obtener excelentes prestaciones en problemas de clasificación y decisión[1]; salvo en casos de elevados niveles de ruido o de abundante presencia de muestras fuera de margen ("outliers") [Dietterich, 2000]. La razón para este deterioro se debe a que la actualización exponencial de los pesos que AB o RAB asigna a cada muestra se concentra demasiado en las muestras que son las más difíciles de clasificar. En otras palabras, cuando algunas muestras son ruidosas, la función de énfasis tiende a concentrarse en ellas, ocasionando que dominen el proceso de entrenamiento.

En el Capítulo 1, se hizo un breve repaso bibliográfico de las diferentes soluciones que se han propuesto en la literatura para solventar el exceso de atención que prestan los métodos de Boosting básicos a las muestras mal clasificadas, lo que representa

[1]Se ha verificado experimentalmente que la adición de más y más clasificadores base al conjunto no deteriora la capacidad de generalización sobre las muestras de test de estos conjuntos, incluso cuando el error de entrenamiento se ha hecho nulo.

una de sus más importantes limitaciones prácticas. Lamentablemente, ninguna de las modificaciones propuestas del algoritmo de Boosting básico parecen ofrecer los resultados esperados cuando se trabaja con conjuntos de datos desequilibrados[2], en presencia de muestras fuera de margen ("outliers") o con distribuciones de datos con formas asimétricas.

Este capítulo se centra en el estudio de problemas que contienen muestras fuera de margen o "outliers" cuando se utiliza el método RAB estándar. Dado que se trata de muestras atípicas o excepcionales, una posibilidad para trabajar, en métodos de Boosting básicos con problemas de clasificación en los que aparecen dichas muestras (que no son fáciles de percibir), es utilizar una novedosa función de énfasis que no sólo tenga en cuenta el error de la muestra a clasificar, sino también los errores de clasificación de las muestras más "próximas" a ellas.

Siguiendo esta línea de razonamiento, en este capítulo se propone modificar la función de énfasis del algoritmo RAB estándar[3], utilizando una combinación convexa que sea la suma de dos términos: el primero, la importancia o atención que se le da al valor de énfasis proporcionado por RAB para la muestra a clasificar, y el segundo, la importancia que se le da al valor medio de los énfasis proporcionados por RAB para las muestras más próximas a la muestra a clasificar. La pregunta que surge entonces aquí es: ¿cuáles y cuántas son las muestras más próximas?

Los clasificadores por vecino más próximo [Cover and Hart, 1967], también conocidos como clasificadores NN de su expresión en inglés *Nearest Neighbor Classifiers*, se definen como aquellos que, a partir de un conjunto de N muestras o patrones etiquetados, que puede ser o no el conjunto completo de patrones de entrenamiento,

[2]El desequilibrio en un conjunto de datos de dos clases se refiere a la situación en la que una clase (mayoritaria) presenta una cantidad notablemente mayor de ejemplos en comparación con la otra clase (minoritaria).

[3]Nótese, por favor, que la función de énfasis indica la importancia que el clasificador debe asignar a cada muestra. En el caso del algoritmo RAB, en cada iteración o ronda, esta función se actualiza para que el siguiente clasificador preste atención a los patrones que se han clasificado erróneamente por la agregación de los anteriores.

asignan a cada muestra no etiquetada, $\mathbf{x}^{(n)}$, la etiqueta del patrón o patrones más "próximos". Si se utiliza sólo el patrón más próximo, se habla del clasificador 1-NN, y si se usan las k muestras más próximas, se habla de clasificador por k vecinos (k-NN).

El concepto de proximidad o similitud requiere de la definición de una medida sobre el espacio de atributos de las muestras. Es habitual el uso de la distancia euclídea, pero también hay otras opciones [Atkeson et al., 1997], como la distancia euclídea ponderada, la distancia de Minkowski, simple o ponderada, la distancia de Manhattan, la distancia de Camberra, etc. La selección de la medida de proximidad apropiada para un problema puede ser determinante para que el clasificador obtenga buenos resultados. En esta Tesis Doctoral se utilizará la distancia euclídea.

Generalmente, si no se especifica, se entiende que se considera sólo el patrón más próximo, dentro del conjunto de N muestras etiquetadas, para determinar la clase. Sin embargo, hay ocasiones en las que el uso de un solo vecino para la clasificación no es la mejor opción. El resultado de la utilización de un vecino con distancia euclídea da lugar a fronteras de decisión poliédricas a igual distancia entre los patrones más próximos a ambos lados de la frontera. Se pueden generar fronteras de decisión más complejas, bien variando la medida de proximidad empleada, bien utilizando k vecinos. Existen varios métodos de K vecinos, entre los cuales aquí se ha optado por usar el k-NN puro, también conocido como k-NN por votación. Es el algoritmo de vecinos más próximos que utiliza la clase más frecuente dentro de los K vecinos para determinar la clase de la muestra que se quiere etiquetar. De este modo, las fronteras se suavizan, pues una muestra que se encuentre rodeada de muestras de otra clase tenderá a ser ignorada en la clasificación.

Para completar esta revisión de los algoritmos basados en vecino más próximo, hay que decir que tienen la característica de que no construyen un modelo que resuma la información de partida. El hecho de prescindir de este modelo hace que, desde el punto de vista del rendimiento, la tarea de clasificar una muestra suponga un proceso costoso, puesto que obliga a calcular la distancia a todas las muestras conocidas. A

pesar de que existen procedimientos que evitan realizar la comparación completa de distancia [Arya and Mount, 1993], sigue siendo un factor de ineficiencia.

Con esto en mente, en la Sección 2.2 se describe matemáticamente la modificación que se propone en esta Tesis Doctoral a la función de énfasis del método RAB estándar, discutiéndose sus principales características, así como sus ventajas frente a dicho algoritmo. A continuación, en la Sección 2.3, a través de una serie de experimentos, se evaluarán las prestaciones de la modificación propuesta. Por último y para finalizar el capítulo, en la Sección 2.4, se presentarán las conclusiones de estos experimentos, discutiéndose conjuntamente las ventajas y limitaciones de la propuesta principal del mismo.

2.2. Algoritmo "K-Nearest Neighbor Real AdaBoost" (KRAB)

2.2.1. El algoritmo KRAB

En el método RAB estándar (véase Apéndice A para mayor detalle), el clasificador base generado en la ronda m-ésima, cuya salida es $o_m(\mathbf{x})$, se entrena para minimizar el error cuadrático medio enfatizado del conjunto de datos original, o en otras palabras, para que minimice la siguiente función:

$$E_m = \sum_{l=1}^{L} D_m(\mathbf{x}^{(l)}) \left[d^{(l)} - o_m(\mathbf{x}^{(l)}) \right]^2 \tag{2.1}$$

donde L es el número de muestras y $D(\cdot)$ es la función de énfasis que indica la importancia que el clasificador debe asignar a cada muestra.

En esta Tesis Doctoral se propone una mejora al algoritmo RAB estándar consistente en reemplazar la función de énfasis $D_m(\mathbf{x}^{(l)})$ utilizada en la función de coste a minimizar en (2.1) por la siguiente combinación convexa:

$$Q_m(\mathbf{x}^{(l)}) = \beta\, D_m(\mathbf{x}^{(l)}) + (1-\beta)\, \frac{\sum_{i \in V^{(l)}} D_m(\mathbf{x}^{(i)})}{K} \tag{2.2}$$

donde $\beta \in [0, 1]$ es un parámetro de mezcla y $V^{(l)}$ representa la vecindad de los K vecinos más cercanos en el algoritmo K-NN de la muestra $\mathbf{x}^{(l)}$. Por tanto, resulta evidente que β es un parámetro empleado para ponderar la atención que el conjunto presta al valor de énfasis proporcionado por el método RAB estándar para la muestra $\mathbf{x}^{(l)}$ y al valor medio de los énfasis proporcionados por RAB para las muestras vecinas o más próximas.

Tal y como puede verse en (2.2), esta nueva función define una énfasis para cada muestra que es combinación convexa del valor de énfasis RAB que corresponde a la muestra $\mathbf{x}^{(l)}$ y los valores de énfasis RAB medios que corresponderían a las K muestras más cercanas. De esta forma, la nueva función de énfasis permite controlar la atención concedida a la muestra y a sus vecinas según el valor del parámetro de mezcla β seleccionado.

Por tanto, si $\mathbf{x}^{(l)}$ es una muestra atípica ("outlier"), la mayoría de sus muestras vecinas serán clasificadas correctamente, y (2.2) reducirá el valor de énfasis aplicado a la muestra $\mathbf{x}^{(l)}$. Si $\mathbf{x}^{(l)}$ se trata de una muestra próxima al borde de la frontera de clasificación, sus muestras vecinas más próximas también estarán localizadas cerca de la frontera, y por tanto, el efecto de la ponderación será menor. Lo mismo ocurre si $\mathbf{x}^{(l)}$ es una muestra localizada en el lado correcto de la frontera de clasificación.

Tal y como se ha comentado a lo largo de este documento, aunque el empleo de criterios para asignar más o menos importancia a las muestras durante el entrenamiento de una red neuronal no es algo novedoso, ya que es posible encontrar en la literatura otros trabajos previos como las técnicas de énfasis mixto [Gómez-Verdejo et al., 2006] y [Gómez-Verdejo et al., 2008], que proponen una función de énfasis consistente en una combinación flexible de dos términos de énfasis, sobre el error cuadrático medio y sobre la proximidad a la frontera, y con las que se podrían obtener prestaciones similares, han demostrado tener una eficacia limitada cuando existen "outliers" en el conjunto de entrenamiento o asimetría en las muestras que hacen imposible una correcta clasificación de las mismas.

A esta nueva modificación del RAB le denominaremos algoritmo KRAB (del

inglés K-*Nearest Neighbor RAB*).

2.2.2. Ventajas del algoritmo KRAB

Para ilustrar de forma gráfica y así entender más claramente las ventajas del algoritmo KRAB respecto al algoritmo RAB estándar, se muestran en esta sección los resultados de clasificación obtenidos por ambos algoritmos para un problema de clasificación binario sencillo ("problema de juguete"[4]). Concretamente, como aprendices se utilizan Perceptrones Multicapa (MLPs), con tres neuronas en la capa oculta, y los parámetros de la combinación convexa utilizados en (2.2) son β =0.3 y K=3. En este problema, las muestras perteneciente a la clase C_{-1} y a la clase C_1 son datos correspondientes a dos distribuciones gaussianas circulares, de igual varianza v=0.004, y medias $[0.3, 0]$ y $[-0.3, 0]$, respectivamente. De cada clase, se han tomado 100 muestras, además de 5 "outliers" situados alrededor del centro de la primera gaussiana (distribución correspondiente a las muestras de la clase C_{-1}).

La Figura 2.1 muestra las fronteras de decisión obtenidas con el algoritmo RAB estándar, y con el algoritmo que se ha propuesto, KRAB. Además, para facilitar la comparación, se muestra también la frontera de decisión teórica (línea vertical $X_2 = 0$). Se puede observar que el algoritmo KRAB crea una frontera de decisión más similar a la teórica que el algoritmo RAB estándar, debido a que evita la "atracción" de la frontera por parte de los "outliers".

Efectos similares pueden aparecer, por ejemplo, cuando las distribuciones gaussianas no son circulares o una de ellas presenta una forma más aguda hacia el centro de la otra. Todos estos desbalanceos y asimetrías pueden aparecer en problemas de clasificación prácticos, pero son difíciles de visualizar.

A continuación, se demostrará experimentalmente que el algoritmo KRAB pro-

[4]Un problema de juguete es un problema sencillo destinado a ilustrar las características de los métodos de resolución de problemas. Éstos se pueden describir de forma exacta y concisa. Esto significa que estos problemas se pueden utilizar para comparar fácilmente el funcionamiento de diversos algoritmos.

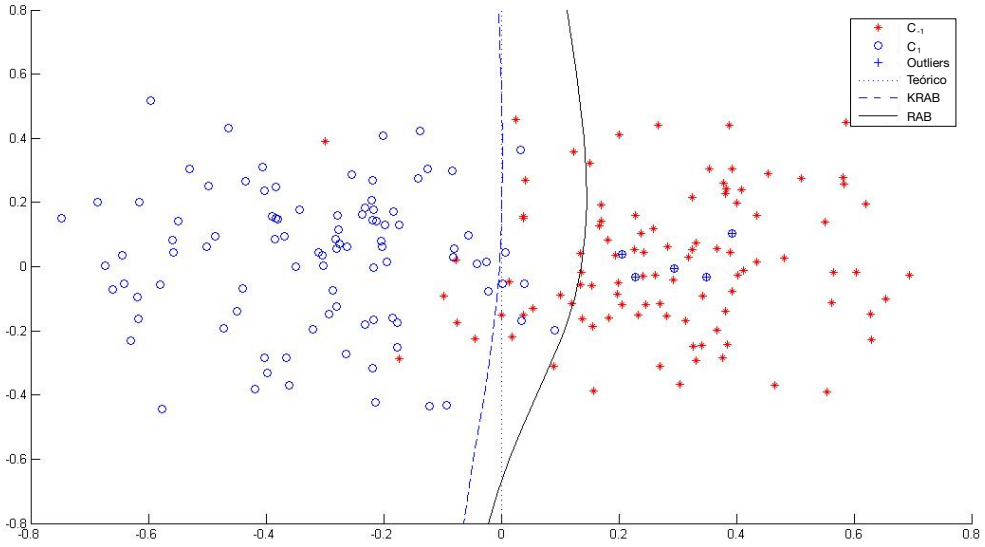


Figura 2.1: Fronteras de decisión obtenidas con el algoritmo RAB estándar y el algoritmo KRAB, para un "problema de juguete".

puesto no sólo proporciona buenos resultados para un "problema de juguete", sino también cuando se evalúa sobre un conjunto de diez bases de datos binarias reales, la mayoría de ellas pertenecientes al repositorio de la Universidad de California en Irvine (UCI), y en otros dos problemas sintéticos.

2.3. Pruebas experimentales

Las prestaciones del algoritmo propuesto, KRAB, se van a comparar con los resultados proporcionados por el algoritmo RAB estándar. Se analizarán su capacidad de generalización y la sensibilidad a los diferentes parámetros propios del diseño propuesto. Antes de ello, se detallan las bases de datos usadas y la configuración de los experimentos.

2.3.1. Bases de datos

Para la evaluación del método propuesto se han considerado doce base de datos binarias con diferentes características (tamaño, dimensión y dificultad). Dos de ellas son problemas sintéticos: Kwok [Kwok, 1999] y Ripley [Ripley, 1994]. Las restantes bases de datos son problemas reales; nueve de ellas se han obtenido del repositorio de la Universidad de California en Irvine (UCI) [Frank and Asuncion, 2010] (abalone, breast, credit, diabetes, german, hepatitis, image, ionosphere, y waveform), y la última, crabs, perteneciente a [Ripley, 1996].

En la Tabla 2.1 se resumen las principales características de cada uno de estos problemas (D: dimensión; C_1/C_{-1}: número de muestras de cada clase) para los conjuntos de entrenamiento y test. La notación o abreviatura con la que se va a denotar cada problema es con sus tres primeras letras. En el Apéndice C se incluye una descripción más detallada de cada uno de estos problemas.

2.3.2. Diseño del clasificador

Para el diseño del algoritmo KRAB, se emplean como aprendices base redes neuronales de tipo MLP con una única capa oculta. En los siguientes apartados se indican los parámetros de diseño y de entrenamiento empleados para llevar a cabo estos experimentos. Asimismo, también se explica cómo se lleva a cabo la selección de M (número de clasificadores del conjunto) durante el entrenamiento de los conjuntos.

2.3.2.1. Diseño de los clasificadores base

Tal y como se explicó en el Capítulo 1, la arquitectura de los MLPs empleados consiste en una única capa oculta, cuyo número de neuronas, denotado como H, se selecciona mediante un proceso de validación cruzada (CV) realizado de manera independiente sobre cada problema de clasificación. Concretamente, se ha llevado a cabo un proceso de CV de 5 particiones con 20 inicializaciones sobre cada partición para seleccionar el número de neuronas en la capa oculta entre valores que varían

Base de datos	**Nº. Muestras entrenamiento** C_1/C_{-1}	**Nº. Muestras test** C_1/C_{-1}	**Dimensión** (D)
Abalone (aba)	2507 1238/1269	1670 843/827	8
Breast (bre)	420 145/275	279 96/183	9
Crabs (cra)	120 59/61	80 41/39	7
Credit (cre)	414 167/247	276 140/136	15
Diabetes (dia)	468 172/296	300 96/204	8
German (ger)	700 214/486	300 86/214	20
Hepatitis (hep)	93 70/23	62 53/9	19
Image (ima)	1300 736/564	1010 584/426	18
Ionosphere (ion)	201 101/100	150 124/26	34
Kwok (kwo)	500 300/200	10200 6120/4080	2
Ripley (rip)	250 125/125	1000 500/500	2
Waveform (wav)	400 124/276	4600 1523/3077	21

Tabla 2.1: Principales características de las bases de datos empleadas en la evaluación del algoritmo propuesto.

en el intervalo [2, H_{max}] (en pasos unitarios), donde, para cada problema, H_{max} se define:

$$H_{max} = \frac{N^{o}. \text{ muestras entrenamiento}}{2 \times (D + 1)}$$

siendo D el número de atributos del problema de clasificación. De esta manera, se asegura que hay, al menos, dos muestras de entrenamiento por cada parámetro entrenable (aproximadamente). Es importante hacer notar que solo ha sido necesario extender este rango hasta 20 y 70 para las bases de datos hep y rip, respectivamente.

En cuanto a la función de coste para el entrenamiento, tal y como se ha comentado en la Sección 2.2.1 de este capítulo, los MLPs se entrenan con el objetivo de (2.1), para RAB; para el algoritmo KRAB, $D_m(\mathbf{x}^{(l)})$ se sustituye por la función de énfasis propuesta en (2.2).

Se inicializan aleatoriamente las componentes de los pesos del MLP con valores uniformemente distribuidos en el intervalo $[-1, 1]$, y se realiza su actualización mediante el algoritmo de retropropagación. Los ajustes para este algoritmo son: un paso de aprendizaje inicial de 0.01 (tanto para la capa oculta como para la de salida) que decrece linealmente hasta 0 durante las 100 épocas que dura el entrenamiento (habiéndose comprobado que éste es un número suficiente para asegurar la convergencia).

Además, se emplea un algoritmo de detención prematura del entrenamiento, utilizando el 80 % de los datos disponibles para entrenar el MLP y el 20 % restante para detener el entrenamiento, es decir, para elegir los pesos correspondientes a la época que presenta un menor error sobre esta partición, reduciendo así los efectos perniciosos del sobreajuste.

2.3.2.2. Selección del número de clasificadores base

Otro de los aspectos a considerar en el diseño es el número total de clasificadores base a emplear, es decir, el número total de rondas, M. Aunque se podría pensar en validación cruzada para seleccionar este parámetro o atender a la evolución del

error de entrenamiento, ninguna de estas soluciones es adecuada para obtener una buena generalización. Así, por ejemplo, si se emplease un proceso de validación cruzada, se reduciría el número efectivo de muestras de entrenamiento, modificando por ello la velocidad de convergencia del algoritmo y llegando a resultados de convergencia distintos de los óptimos. Si, por el contrario, se atendiese al valor del error de entrenamiento, no habría garantías de que el error de generalización presentase un comportamiento similar, ya que podría ocurrir que el error de entrenamiento dejase de reducirse o, incluso, llegase a anularse, y aún así, podría ser conveniente aumentar el número de redes base del conjunto para mejorar su capacidad de generalización [Gómez-Verdejo, 2007].

Por las razones anteriores, para el diseño de los conjuntos se ha optado por seleccionar el número final de rondas M en función de la evolución de los valores α_{m}, dado que cuando estos valores se hacen muy próximos a cero, el añadir nuevos clasificadores apenas afecta a las prestaciones del mismo. Es decir, se propone incrementar el número de redes hasta que el hecho de añadir nuevos componentes apenas tenga influencia en la clasificación. De forma analítica, el criterio empleado consiste en detener el crecimiento del conjunto cuando el valor relativo de α_{m} en las últimas $\mathrm{M_{ult}}$ rondas es menor que un cierto valor de parada, $\mathrm{C_{stop}}$ [Gómez-Verdejo et al., 2008]; es decir,

$$\frac{\sum_{\mathrm{m'}=\mathrm{M}-\mathrm{M_{ult}}+1}^{\mathrm{M}} \alpha_{\mathrm{m'}}}{\mathrm{M_{ult}} \sum_{\mathrm{m'}=1}^{\mathrm{M}} \alpha_{\mathrm{m'}}} < \mathrm{C_{stop}} \tag{2.3}$$

donde se ha fijado de forma experimental $\mathrm{M_{ult}} = 10$ y $\mathrm{C_{stop}} = 0.01$ para todos los problemas.

2.3.3. Selección de los parámetros de diseño de la mezcla

Los valores de los parámetros de la mezcla, β y K, se seleccionan mediante un proceso de validación cruzada de 5 particiones en que se han empleado 20 repeticiones

por cada partición.

El algoritmo KRAB necesita seleccionar dos parámetros: el número de vecinos más próximos K, junto con el parámetro de la mezcla, β. El parámetro K se explora en once valores linealmente espaciados en el intervalo $[1, 11]$, mientras que β entre once valores equidistantes en el intervalo $[0, 1]$ (i.e., $\beta \in \{0, 0.1, 0.2, \ldots, 1\}$). El rango de exploración para cada uno de los parámetros se ha fijado de tal manera que los valores óptimos no estén ubicados en los extremos en el momento de evaluar la evolución del error de validación.

Nótese que ambos algoritmos, RAB estándar y KRAB, requieren también la selección del número de neuronas de la capa oculta de los MLPs, H, tal y como se explicó en la subsección 2.3.2.1.

2.3.4. Resultados

En la Tabla 2.2 se muestran las tasas de error y desviaciones estándar de clasificación de test, CE, obtenidas después de promediar sobre 50 repeticiones para cada conjunto usando los valores de los parámetros seleccionados por validación cruzada. También se indica, por un lado, el tamaño medio (T) de los conjuntos, así como el número de neuronas ocultas (H) que han empleado los MLPs que forman el conjunto, y, por otro lado, el p-valor (p) proporcionado por el T-test aplicado a los errores de clasificación de los algoritmos RAB y KRAB (para mayor información sobre el T-test, recomendamos al lector ver el Apéndice D). Cabe reseñar que un valor positivo del T-test indica que la tasa de error del KRAB mejora la del RAB, y un valor negativo indica lo contrario. Además, para facilitar la comparación entre los algoritmos, se ha resaltado con negrita aquel error de clasificación que resulta ser el menor de la comparación de estos algoritmos, y los valores de p por debajo de 0.05 (lo que indica diferencia estadística entre los resultados con un nivel de certeza del 95 %).

De los análisis de los resultados de la Tabla 2.2, pueden extraerse las siguientes conclusiones:

		RAB M	KRAB H/K/β	T-test	Omni RAB H	Omni KRAB H/K/β
aba	CE (%)	19.40 ± 0.02	**18.98 ± 0.18**	**7.7597e-025**	19.32 ± 0.26	18.88 ± 0.15
	T	31.20 ± 0.40	26.64 ± 2.31			
	Param.	4	27/9/0.6		7	21/5/0.4
bre	CE (%)	2.60 ± 0.40	**2.19 ± 0.34**	**7.0811e-008**	2.29 ± 0.38	1.97 ± 0.22
	T	21.30 ± 4.20	30.32 ± 7.12			
	Param.	6	2/7/0.9		7	2/10/0.5
cra	CE (%)	2.50 ± 0	2.50 ± 0	– –	2.50 ± 0	2.50 ± 0
	T	11.10 ± 0.80	20.42 ± 1.58			
	Param.	2	2/1/0		2	2/1/0
cre	CE (%)	10.14 ± 0.74	**9.86 ± 0.81**	0.0726	9.40 ± 0.76	9.18 ± 0.52
	T	29.36 ± 6.49	21.60 ± 3.43			
	Param.	2	4/1/0.5		3	5/1/0.8
dia	CE (%)	20.61 ± 0.78	**20.39 ± 0.66**	0.2279	20.05 ± 0.44	20.00 ± 0.81
	T	33.20 ± 4.98	33.68 ± 4.41			
	Param.	2	2/3/0.8		8	11/3/0
ger	CE (%)	22.27 ± 0.71	**22.11 ± 0.80**	0.0731	21.67 ± 0.63	21.57 ± 0.55
	T	33.60 ± 5.92	18.18 ± 1.16			
	Param.	2	2/2/0		2	2/5/0.8
hep	CE (%)	8.90 ± 1.80	**6.55 ± 0.89**	**7.6872e-013**	7.26 ± 1.85	6.05 ± 1.27
	T	22.20 ± 3.90	60.06 ± 17.52			
	Param.	17	18/11/0.3		11	2/6/0.2
ima	CE (%)	2.99 ± 0.43	**2.86 ± 0.46**	0.0807	2.82 ± 0.40	2.63 ± 0.36
	T	21.16 ± 3.08	21.62 ± 2.81			
	Param.	11	11/11/0.6		15	8/8/0.5
ion	CE (%)	4.90 ± 0.90	**4.20 ± 0.90**	**1.4028e-004**	4.50 ± 1.06	4.07 ± 0.75
	T	13.40 ± 4.50	22.98 ± 4.08			
	Param.	5	2/9/0.1		2	2/9/0.2
kwo	CE (%)	11.70 ± 0.01	**11.59 ± 0.06**	**5.1686e-009**	11.70 ± 0.12	11.52 ± 0.07
	T	29.30 ± 0.10	29.88 ± 2.50			
	Param.	15	13/1/0.8		9	25/5/0.9
rip	CE (%)	9.70 ± 0.01	**9.16 ± 0.21**	**2.6148e-18**	9.00 ± 0.20	9.00 ± 0.20
	T	28.90 ± 0.90	33.12 ± 4.52			
	Param.	48	53/2/0.9		61	61/1/1
wav	CE (%)	11.65 ± 0.36	**11.31 ± 0.42**	**3.8557e-005**	11.57 ± 0.26	11.31 ± 0.25
	T	30.08 ± 6.05	15.98 ± 1.13			
	Param.	2	3/10/0		2	2/10/10

Tabla 2.2: Tasa de error de clasificación (CE) y tamaño medio de los conjuntos (T) para los algoritmos RAB y KRAB y por la aproximación "omnisciente". También se muestra el p-valor proporcionado por el T-test.

- En todos los casos, el algoritmo KRAB ofrece mejores resultados que el RAB convencional, salvo en cra, en que ambos algoritmos empatan. Cra es una base de datos "curiosa", ya que la tasa de clasificación correcta para el conjunto de test de muchos clasificadores para este problema es siempre CE = 2.50 %.

- Se puede observar claramente que para siete de las doce base de datos, el algoritmo KRAB ofrece los mejores resultados, superando con una diferencia significativa ($p < 0.05$) al RAB estándar, sin incrementar el coste computacional de operación, salvo para las bases de datos aba y hep.

- Para ocho de los doce problemas de clasificación, los diseños con el algoritmo KRAB requieren un menor número (promedio) de clasificadores base. Incluso más, para la mitad de los problemas, se requiere un número de neuronas ocultas para los MLPs igual o menor con el algoritmo KRAB que con el algoritmo RAB convencional.

Aparte de lo anterior, si se comparan los resultados del CV KRAB con la aproximación "omnisciente" para evaluar la idoneidad de seleccionar los tres parámetros de diseño del algoritmo KRAB mediante CV (H, K y β), se puede concluir que el proceso de CV es tan solo moderadamente útil. Nótese que, aunque en la mayoría de los casos se ha seleccionado valores próximos a los óptimos, una selección más fina en los problemas de clasificación bre, cre, ger y hep proporcionaría mejoras todavía más significativas.

En definitiva, a la vista de los resultados de la Tabla 2.2 y analizando los resultados del T-test, se puede concluir que, en general, la idea de utilizar una función de énfasis que sea combinación convexa de la atención concedida al valor de énfasis RAB de la muestra a clasificar y los valores de énfasis RAB medios de sus muestras vecinas, sirve para mejorar las prestaciones de los diseños resultantes. Obviamente, esto no es gratis, ya que el entrenamiento es más costoso y, especialmente, el costoso proceso de CV aumenta en gran medida el esfuerzo computacional de diseño (son

tres parámetros de diseño). Éste es el precio a pagar para mejorar los excelentes resultados que ya de por sí proporciona el algoritmo RAB estándar.

2.4. Conclusiones

En este capítulo se han analizado las prestaciones de una novedosa función de énfasis, para el método RAB estándar, basada no sólo usar información del error de la muestra a clasificar, sino también de los errores de clasificación de sus muestras más próximas.

Esta idea surge de la finalidad de limitar la influencia de muestras atípicas y aisladas, también denominadas "outliers". Una manera sencilla de limitar su influencia es implementando una función de énfasis que sea una combinación convexa del valor de la función de énfasis del algoritmo RAB estándar para la muestra a clasificar y el promedio de los valores de énfasis RAB de sus muestras vecinas. De esta forma, la nueva función de énfasis permite controlar la atención concedida a la muestra y a sus vecinas según el valor del parámetro de mezcla β seleccionado, reduciendo así la influencia de posibles muestras atípicas o "outliers". En esta Tesis Doctoral, a este nuevo algoritmo se le ha denominado KRAB (del inglés K-*Nearest Neighbor* RAB).

Se ha demostrado experimentalmente que esta nueva función de énfasis suavizada proporciona mejores resultados que el algoritmo RAB estándar en un número considerable de problemas de clasificación ampliamente utilizados para la evaluación de máquinas de clasificación.

Capítulo 3

Conjuntos de clasificadores diseñados por Boosting con intensidad de énfasis controlada

3.1. Introducción

En este capítulo se considerará una generalización de la función de énfasis híbrido utilizada en versiones del algoritmo Real AdaBoost (RAB) propuestas en trabajos previos [Gómez-Verdejo et al., 2006, Gómez-Verdejo et al., 2008] que, como ya se ha discutido, ponderan (a través de un parámetro de mezcla) las muestras según su error de clasificación y proximidad a la frontera. En esta nueva función de énfasis se incluye un término constante que sirve para moderar la intensidad del énfasis, o en otras palabras, limitar la atención centrada en las muestras próximas a la frontera o más difíciles de clasificar. Este tipo de énfasis se ha probado con éxito en clasificadores profundos usando máquinas auxiliares [Alvear-Sandoval et al., 2017] [Alvear-Sandoval and Figueiras-Vidal, 2018], permitiendo obtener una mejora de las prestaciones mucho mayor que la que se alcanza con formas más simples de la función de énfasis.

Es importante resaltar desde el principio de este capítulo que no existe ninguna garantía teórica de obtener ventaja en todas las situaciones prácticas cuando se utiliza esta nueva función de énfasis: un exceso de énfasis en las muestras que están

próximas a la frontera puede crear dificultades aún peores que el sobreajuste, y la necesidad de establecer empíricamente los valores de los parámetros de énfasis puede llevar a diseños subóptimos.

Este capítulo se estructura como sigue. En la Sección 3.2 se presentará y justificará la función de énfasis propuesta. Cabe destacar aquí que, en los experimentos llevados a cabo, únicamente se han considerado problemas binarios, aunque la formulación es fácilmente extensible a situaciones multiclase. En la Sección 3.3 se mostrarán y discutirán los resultados obtenidos en comparación con aquellos alcanzados con conjuntos de clasificadores diseñados por RAB y los obtenidos con una versión no moderada de la función de énfasis propuesta. Finalmente, en la Sección 3.4 se presentarán las conclusiones de estos experimentos, discutiéndose las ventajas y limitaciones de la función de énfasis que se propone.

3.2. Función de énfasis propuesta

De acuerdo con lo mencionado en la Introducción, la función de énfasis que se considerará en este capítulo es:

$$p_m\big(\mathbf{x}^{(n)}\big) = \alpha + (1-\alpha)\left[\frac{\beta\big(t^{(n)} - o_{m-1}^{(n)}\big)^2}{4} + (1-\beta)\big(1 - o_{m-1}^{(n)\,2}\big)\right] \tag{3.1}$$

donde:

- p_m es el peso para la muestra $\big\{\mathbf{x}^{(n)}, t^{(n)}\big\}$ (vector de observación y su correspondiente etiqueta, ± 1) para el clasificador base o aprendiz m;
- $o_{m-1}^{(n)}$ representa la salida agregada de los $m-1$ clasificadores base previos para la muestra $\mathbf{x}^{(n)}$ (la agregación se lleva a cabo según la forma estándar del algoritmo RAB);
- α, β son los dos parámetros de la combinación convexa ($0 \leq \alpha, \beta \leq 1$) que sirven, por un lado, para determinar la importancia concedida al error cuadrático

de la muestra $\left(t^{(n)} - o_{m-1}^{(n)}\right)^2$ y la proximidad a la frontera de clasificación, $1 - o_{m-1}^{(n)\,2}$, y por otro lado, para moderar la intensidad del énfasis. Cabe resaltar aquí que $\alpha = 0$ simplificará el énfasis general a una combinación convexa del error y la cercanía a la frontera de la muestra, sirviendo, en este caso, el parámetro β para balancear la importancia concedida a ambos términos de énfasis. Nótese que esta situación coincide con la función de énfasis propuesta en [Gómez-Verdejo et al., 2006, Gómez-Verdejo et al., 2008], pero haciendo uso de medidas analíticas alternativas para el cálculo del error y proximidad a la frontera. Para el caso particular de $\alpha = 0$ y $\beta = 1$, se tiene la forma de coste del error cuadrático de un algoritmo RAB puro, cuya notación en este capítulo es ARA (del inglés *Alternative Real Adaboost*). Por último, mencionar que los valores de α y β se seleccionan mediante un proceso de validación cruzada (CV) como se describirá más adelante.

Por claridad, insistimos: la función de énfasis (3.1) está formada por tres componentes. El primero es el término constante α: cuando este parámetro toma valores muy altos, la intensidad del énfasis se reduce, lo que puede ser beneficioso cuando se resuelven ciertos problemas. Los otros dos términos, que se combinan con α de una manera convexa, consideran, por una parte, el error de las muestras, medido en su forma cuadrática clásica, $\left(t^{(n)} - o^{(n)}\right)^2$, y por otro lado, la proximidad de las mismas a la frontera, $1 - o^{(n)\,2}$. Esta descomposición permite conceder más importancia a las muestras cuya salida es próxima a cero en la máquina auxiliar, es decir, aquellas que están próximas a la frontera de decisión. Como se ilustra en (3.1), se utiliza también una combinación convexa para estos dos términos, en la que el parámetro de mezcla ajustable β permite controlar el compromiso entre ambos.

Con respecto a la máquina de clasificación auxiliar, o guía, que proporciona los valores de salida $o^{(n)}$ que se utilizan en la expresión (3.1), existen evidencias que muestran la ventaja de utilizar máquinas relativamente potentes que ofrezcan salidas no muy diferentes de aquellas que se obtendrían con el diseño enfatizado. Según ello, una posible y apropiada selección de máquina auxiliar sería la salida del conjunto

parcial cuando se entrena cada clasificador base. La salida de este conjunto parcial sería suficientemente potente en los pasos finales del proceso de construcción del conjunto, resultando evidente su similitud.

Para concluir este apartado, cabe mencionar que podrían usarse diferentes medidas de error y de proximidad a la frontera en (3.1), obteniéndose mejores o peores resultados dependiendo de la base de datos bajo estudio. A este respecto, recomendamos la referencia bibliográfica [Breiman, 1998] para defender el objetivo de esta Tesis Doctoral: comprobar si moderar el énfasis con $\alpha \neq 0$ puede ser beneficioso o no, y no explorar el uso de usar otras medidas de error o de distancia a la frontera en diferentes problemas de clasificación. A este respecto, nótese que la medida de error y distancia a la frontera elegidas hacen que la forma (3.1) sea computacionalmente eficiente.

3.3. Pruebas experimentales

3.3.1. Bases de datos

Para la evaluación de la función de énfasis propuesta en este capítulo se han considerado las mismas doce bases de datos que se utilizaron en los experimentos del Capítulo 2. Remitimos al lector al Apéndice C para informarse más en detalle sobre estos problemas de clasificación.

3.3.2. Diseño de los clasificadores base

Como clasificadores base o aprendices, se han utilizando Perceptrones Multicapa (MLPs) de una sola capa oculta, ya que son máquinas inestables, y esto los hace sensibles a diferencias en la función de énfasis. Los máquinas se entrenan con el algoritmo de retropropagación, utilizando como función de coste el error cuadrático entre la salida deseada y la salida de la red. Para ello, se inicializan aleatoriamente las componentes de los pesos del MLP con valores uniformemente distribuidos en

el intervalo $[-0.2, 0.2]$, y se realiza su actualización mediante el algoritmo de retropropagación. La tasa o paso de aprendizaje se fija en 0.01, habiéndose comprobado experimentalmente que este valor es adecuado para asegurar la convergencia.

Junto con el algoritmo de retropropagación, se emplea un algoritmo de detención prematura del entrenamiento, utilizando el 80 % de los datos disponibles para entrenar el MLP y el 20 % restante para detener el entrenamiento, es decir, para elegir los pesos correspondientes a la época que presenta un menor error sobre esta partición, reduciendo así los efectos perniciosos del sobreajuste.

El número de neuronas de la capa oculta de los MLPs, H, se establece mediante una CV de 5 repeticiones del conjunto de entrenamiento original (80 % para entrenamiento y 20 % para validación); también los parámetros libres de los diseños se seleccionan con la misma CV (α y β explorados de 0 a 1 en pasos de 0.1). Se realizan 20 inicializaciones (repeticiones) independientes del MLP por cada partición de la CV.

3.3.3. Resultados

Las prestaciones de la función de énfasis propuesta se presentan para cinco tipos de conjuntos diseñados vía "boosting":

- El método que se propone, y que se indica como CRA (del inglés *Controlled RAB*), donde α y β se establecen mediante un proceso de CV.
- Dos algoritmos que no incluyen el término constante para moderar la intensidad del énfasis, es decir, que tienen $\alpha = 0$:
  - βRA: incluye tanto el término de error de las muestras como el de proximidad de las mismas a la frontera –repetimos que este esquema de énfasis es análogo al propuesto en [Gómez-Verdejo et al., 2006, Gómez-Verdejo et al., 2008].
  - ARA: corresponde a la situación particular en que $\alpha = 0$ y $\beta = 1$.

- Y, finalmente, para completar, dos métodos que sí incluyen el término constante que permite moderar la intensidad del énfasis, α; pero en cada caso solo se incluye uno de los dos términos de la función de énfasis híbrido:
  - α1RA ($\beta = 1$): método que incluye el error cuadrático de las muestras.
  - α0RA ($\beta = 0$): método que incluye la proximidad de las muestras a la frontera.

En la Tabla 3.1 se muestran las tasas de error de clasificación de test $\pm$ desviaciones estándar, obtenidas después de promediar 50 repeticiones para cada conjunto usando los valores de los parámetros seleccionados por CV en cada caso. También se indica, entre paréntesis, el tamaño medio de los conjuntos (M). Además, para facilitar la comparación entre algoritmos y las diferente variantes, el símbolo $\bullet$ indica si existe una diferencia estadística significativa, para los métodos CRA y βRA con respecto al método estándar ARA según el test de Wilcoxon (véase el apéndice D) con un nivel de significación del 95 %.

Analizando los resultados de la Tabla 3.1 pueden extraerse las siguientes conclusiones:

- CRA vs. ARA: En ocho bases de datos (bre, dia, ger, ima, ion, kwo, rip y wav), el algoritmo CRA ofrece mejores resultados que el ARA, y no es peor que el ARA para las otras cuatro bases de datos restantes (aba, cra, cre y hep), de acuerdo con los test de Wilcoxon llevados a cabo con un nivel de confianza del 95 %.

  Debe señalarse que los resultados de los test de Wilcoxon son solo indicativos, ya que los experimentos no son estrictamente independientes. Se puede comprobar fácilmente que, en nuestros casos, estos resultados son muy similares a los obtenidos por la simple regla general de aceptar que las prestaciones son estadísticamente diferentes cuando la diferencia entre los valores medios de error es mayor que el valor medio de las desviaciones estándar, regla general que se aplicará en esta Tesis Doctoral para otras comparaciones menos relevantes.

	CRA (M) (H, α, β)	βRA ($\alpha = 0$) (M) (H, β)	ARA ($\alpha = 0$, $\beta = 1$) (M) (H)	α1RA ($\beta = 1$) (M) (H, α)	α0RA ($\beta = 0$) (M) (H, α)
aba	19.2 ± 0.2 (18.4 ± 0.5) (5/0.4/0.1)	19.3 ± 0.3 (24.6 ± 3.3) (5/0.1)	19.2 ± 0.4 (28.7 ± 3.8) (8)	19.3 ± 0.4 (19.6 ± 1.2) (5/0.2)	19.3 ± 0.2 (19.2 ± 0.8) (5/0.3)
bre	2.1 ± 0.2 • (16.9 ± 3.4) (2/0.1/0.1)	2.1 ± 0.4 • (29.0 ± 9.4) (6/0.5)	2.5 ± 0.4 (20.1 ± 6.2) (2)	2.1 ± 0.2 (13.6 ± 0.5) (5/0.5)	2.1 ± 0.2 (14.0 ± 0.3) (3/0.6)
cra	2.5 ± 0 (98.0 ± 3.8) (2/0.1/0.1)	2.5 ± 0 (98.9 ± 3.7) (2/0.1)	2.5 ± 0 (89.0 ± 7.4) (2)	2.5 ± 0 (92.8 ± 4.8) (2/0.1)	2.5 ± 0 (99.2 ± 3.2) (2/0.1)
cre	7.4 ± 1.3 (18.1 ± 1.5) (2/0/0.9)	7.4 ± 1.3 (18.1 ± 1.5) (2/0.9)	7.4 ± 1.3 (18.0 ± 1.8) (2)	8.7 ± 0.8 (15.9 ± 0.7) (2/0.1)	10.8 ± 0.8 (15.2 ± 0.6) (3/0.5)
dia	22.2 ± 0.8 • (18.7 ± 0.5) (2/0.9/0.3)	22.5 ± 0.9 • (19.9 ± 0.8) (2/0.1)	25.8 ± 1.4 (35.9 ± 7.9) (6)	22.3 ± 0.8 (18.6 ± 0.6) (2/0.9)	22.1 ± 0.8 (18.7 ± 0.6) (2/0.4)
ger	22.2 ± 1.1 • (17.5 ± 0.9) (3/0.8/0.2)	22.8 ± 1.3 • (20.5 ± 1.9) (2/0.1)	25.9 ± 1.5 (56.9 ± 12.3) (6)	22.4 ± 0.9 (18.6 ± 1.1) (3/0.7)	22.6 ± 1.0 (17.7 ± 0.7) (2/0.8)
hep	6.9 ± 0.8 (19.7 ± 2.1) (9/0.9/0.6)	7.2 ± 1.2 (22.2 ± 3.9) (18/1)	7.2 ± 1.2 (22.2 ± 3.9) (18)	7.1 ± 0.9 (23.0 ± 2.6) (18/0.1)	7.0 ± 0.9 (25.9 ± 3.8) (18/0.8)
ima	3.0 ± 0.4 • (60.9 ± 9.0) (15/0.2/0.2)	3.1 ± 0.3 • (64.0 ± 9.0) (15/0.1)	4.1 ± 0.6 (32.5 ± 3.4) (15)	3.3 ± 0.5 (16.7 ± 0.6) (15/1)	3.1 ± 0.4 (54.1 ± 10.3) (15/0.4)
ion	3.9 ± 0.7 • (17.1 ± 1.5) (2/0.6/0)	4.3 ± 0.9 (18.2 ± 1.4) (2/0.3)	4.3 ± 0.8 (29.8 ± 6.2) (7)	4.2 ± 0.8 (18.0 ± 1.7) (7/1)	3.9 ± 0.7 (17.1 ± 1.5) (2/0.6)
kwo	11.6 ± 0.1• (22.6 ± 4.5) (15/0/0.5)	11.6 ± 0.1 • (22.6 ± 4.5) (15/0.5)	11.8 ± 0.2 (18.7 ± 1.5) (6)	11.7 ± 0.1 (16.5 ± 0.5) (13/0.9)	11.7 ± 0.1 (16.9 ± 0.7) (13/0.7)
rip	9.0 ± 0.2 • (31.6 ± 5.3) (14/0.3/0.1)	9.1 ± 0.2 • (37.7 ± 6.3) (15/0.1)	9.3 ± 0.3 (37.1 ± 6.4) (43)	9.6 ± 0.3 (22.7 ± 3.2) (48/0.1)	9.0 ± 0.2 (35.5 ± 5.8) (13/0.1)
wav	11.3 ± 0.3 • (32.2 ± 10.0) (4/0.2/0.1)	11.6 ± 0.5 (17.2 ± 3.2) (2/0.2)	11.5 ± 0.3 (40.1 ± 6.3) (6)	11.3 ± 0.3 (16.0 ± 1.2) (3/0.7)	11.3 ± 0.4 (24.4 ± 6.0) (3/0.3)

Tabla 3.1: Tasa de error de clasificación (± desviación estándar) sobre datos de test para los cinco algoritmos de boosting para aba, bre, cra, cre, dia, ger, hep, ima, ion, kwo, rip y wav, y parámetros de diseño de cada método (H, α, β) seleccionados por CV. También se muestra el tamaño medio de los conjuntos (M). El símbolo • indica si existe una diferencia estadística significativa, para CRA y βRA, con respecto al método estándar ARA según el test Wilcoxon con un nivel de significancia del 95 %.

Los resultados anteriores permiten concluir que el método CRA proporciona claras ventajas con respecto al método tradicional ARA.

- βRA vs. ARA: βRA no mejora los resultados de ARA para dos bases de datos que muestran claras ventajas cuando se utiliza CRA (ion y wav), de acuerdo con los mismos test de Wilcoxon, pero sí que mejora los resultados obtenidos con ARA para las seis bases de datos restantes (bre, dia, ger, ima, kwo y rip). Por tanto, βRA puede considerarse mejor algoritmo que el ARA convencional, pero no tan bueno como lo es el CRA. Cabe resaltar que esto ocurre incluso usando un mecanismo de CV sencillo, un hecho que afecta más al algoritmo CRA que al βRA, porque el primero incluye dos parámetros no entrenables (α y β) mientras el segundo solo uno (β).

  También resulta inmediato observar en los resultados anteriores que tanto el término constante como el término de proximidad a la frontera son útiles para mejorar las prestaciones de los algoritmos en un número suficiente de casos, aunque la mayoría de las ventajas se deben principalmente al segundo término.

- α1RA y α0RA: El restringir β a cada uno de sus dos valores extremos ($\beta = 0$ ó $\beta = 1$), consigue los efectos que se esperaban. Cuando se utiliza α1RA ($\beta = 1$) –una versión moderada de ARA– las cosas se vuelven claramente peores con respecto a CRA para las bases de datos cre y rip. Se puede observar que no hay una degradación significativa para las bases de datos que utilizaban un valor grande de α en CRA (dia, ger y hep). Por otro lado, cuando se utiliza α0RA ($\beta = 0$) –combinación de un término de control y un término que indica la proximidad a la frontera–, se produce más degradación para cre, pero no para rip. Esto indica que una función de énfasis moderado según la proximidad de las muestras a la frontera de clasificación puede ocasionalmente proporcionar mejores prestaciones que la función de énfasis tradicional que considera solamente el error cuadrático de las muestras.

Con objeto de apreciar más claramente las diferencias de los resultados obtenidos

	RAB	ARA
aba (M) (H)	19.4 ± 0.02 (31.2 ± 0.4) (4)	19.2 ± 0.4 (28.7 ± 3.8) (8)
bre (M) (H)	2.6 ± 0.4 (21.3 ± 4.2) (6)	2.5 ± 0.4 (20.1 ± 6.2) (2)
cra (M) (H)	2.5 ± 0 (11.1 ± 0.8) (2)	2.5 ± 0 (89.0 ± 7.4) (2)
cre (M) (H)	10.1 ± 0.7 (29.4 ± 6.5) (2)	7.4 ± 1.3 (18.0 ± 1.8) (2)
dia (M) (H)	20.6 ± 0.8 (33.2 ± 5.0) (2)	25.8 ± 1.4 (35.9 ± 7.9) (6)
ger (M) (H)	22.3 ± 0.7 (33.6 ± 5.9) (2)	25.9 ± 1.5 (56.9 ± 12.3) (6)
hep (M) (H)	8.9 ± 1.8 (22.2 ± 3.9) (17)	7.2 ± 1.2 (22.2 ± 3.9) (18)
ima (M) (H)	3.0 ± 0.4 (21.2 ± 3.1) (11)	4.1 ± 0.6 (32.5 ± 3.4) (15)
ion (M) (H)	4.9 ± 0.9 (13.4 ± 4.5) (5)	4.3 ± 0.8 (29.8 ± 6.2) (7)
kwo (M) (H)	11.7 ± 0.01 (29.3 ± 0.1) (15)	11.8 ± 0.2 (18.7 ± 1.5) (6)
rip (M) (H)	9.7 ± 0.01 (28.9 ± 0.9) (48)	9.3 ± 0.3 (37.1 ± 6.4) (43)
wav (M) (H)	11.7 ± 0.4 (30.1 ± 6.1) (2)	11.5 ± 0.3 (40.1 ± 6.3) (6)

Tabla 3.2: Tasa de error de clasificación (± desviación estándar) sobre datos de test para los métodos ARA y RAB convencional para las bases de datos de los experimentos, y parámetros de diseño de cada método (H, número de neuronas ocultas), seleccionados por CV. También se muestra el tamaño medio de los conjuntos (M).

al aplicar los distintos métodos de énfasis, la Tabla 3.2 muestra una comparación de las prestaciones del método ARA y RAB convencional (los resultados de este último han sido tomados de la Tabla 2.2 del Capítulo 2). A simple vista, parece que el método "ganador" es dependiente del problema, es decir, para los problemas cre, hep, ion y rip, los mejores resultados se obtienen con ARA, mientras que para dia, ger e ima se obtienen con RAB.

Comparando los resultados obtenidos con CRA con los del mejor método propuesto en el Capítulo 2, KRAB (remitimos al lector a la Tabla 2.2 para informarse más en detalle), se puede observar que hay claras diferencias para el problema cre (ventaja para CRA) y para ger, hep, e ima (ventaja de KRAB) (insistimos en la gran carga computacional que demanda el diseño de conjuntos basados en el método

KRAB). Estas diferencias pueden ser atribuidas a las diferentes medidas de proximidad y error que se utilizan en ambas familias de métodos. En este sentido, es importante mencionar que las técnicas DW (del inglés *Dynamic Weighting*) permiten seleccionar los términos de énfasis híbrido y proximidad a la frontera para cada aprendiz maximizando el valor límite, es decir, el valor que proporciona el mayor margen generalizado –ver [Gómez-Verdejo et al., 2008]–, y esto es una ventaja implícita, y también elimina la necesidad de utilizar CV para seleccionar el parámetro de la combinación.

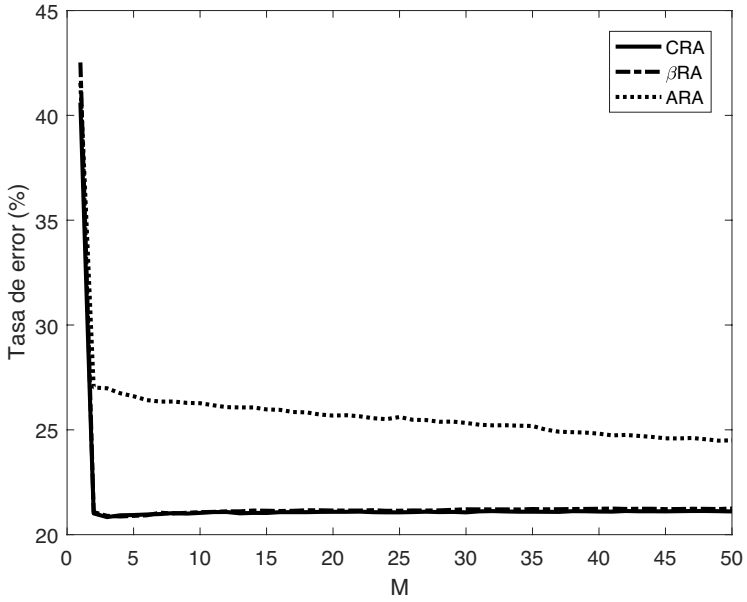


Figura 3.1: Evolución de la tasa de error media de entrenamiento (en %) con el número de clasificadores base (M) para el algoritmo CRA (α =0.9, β =0.3) (línea continua), βRA (β =0.1) (línea punto/raya) y ARA (línea punteada) en el problema dia.

Con el fin de comprobar si esta simplificación de los procesos de CV tiene alguna influencia en los resultados, es interesante visualizar la convergencia de los algoritmos que se han propuesto en este capítulo (CRA, βRA y ARA). Las Figuras 3.1 y 3.2 muestran la evolución de la tasa media de error (en %) con el número de clasificadores base (M) para el problema dia con los algoritmos CRA, βRA y ARA, seleccionando

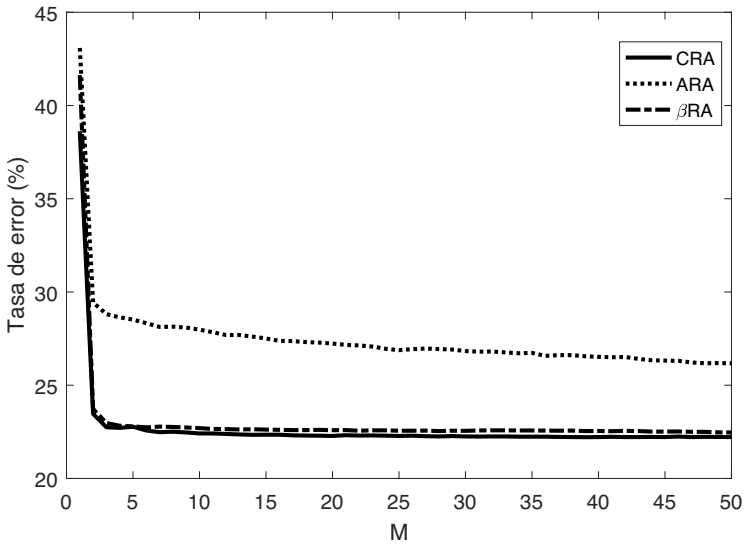


Figura 3.2: Evolución de la tasa de error media de test (en %) con el número de clasificadores base (M) para el algoritmo CRA ($\alpha = 0.9$, $\beta = 0.3$) (línea continua), βRA ($\beta = 0{,}1$) (línea punto/raya) y ARA (línea punteada) en el problema dia.

sus parámetros de diseño por CV: CRA con $\alpha = 0.9$, $\beta = 0.3$, βRA con $\beta = 0.1$. Nótese que tanto el método CRA como el método βRA alcanzan (aproximadamente) sus mejores prestaciones para el conjunto de entrenamiento con 2 aprendices, y mantienen estas prestaciones cuando el número de aprendices aumenta. Esto provocará problemas de parada, y en consecuencia, dificultades a la hora de seleccionar sus parámetros por CV. De hecho, un diseño omnisciente de CRA (es decir, seleccionando sus parámetros de diseño según los resultados obtenidos sobre el conjunto de test) lleva a obtener $\alpha = 0.3$, $\beta = 0.6$ con el mismo valor de H ($H = 2$), y con un número similar de aprendices (19.3 ± 0.5)... pero con mejores prestaciones: 19.1 ± 0.4 %. Es importante dejar claro que el diseño omnisciente no es válido, pero sirve para concluir si el proceso de CV es efectivo o no. Por tanto, es obvio que mejores métodos de parada así como mejores técnicas de CV permitirán, a costa de un no despreciable esfuerzo computacional, obtener diseños claramente más competitivos en cuanto a prestaciones. Sin embargo, en este capítulo se ha comprobado experi-

mentalmente, sobre un razonable conjunto de problemas de referencia, que el método CRA es un muy buen algoritmo de boosting que requiere un esfuerzo computacional relativamente moderado para su diseño. En otras palabras, introducir un parámetro de control de énfasis tal como α es una herramienta eficiente y eficaz para la construcción de conjuntos vía "boosting".

3.4. Conclusiones

En este capítulo se han analizado las prestaciones de una nueva función de énfasis para construir conjuntos por "boosting", la cual no solo se encuentra compuesta de dos factores, uno relacionado con el error cuadrático de las muestras y otro asociado con la proximidad de las mismas a la frontera, sino que incluye también un término constante que sirve para controlar cuánto énfasis se aplica.

Los experimentos llevados a cabo sobre doce bases de datos de diferentes características evidencian las buenas prestaciones que se obtienen incluyendo este término constante en la función de énfasis. Las ventajas conseguidas únicamente requieren un sencillo y computacionalmente razonable proceso de validación cruzada.

Las líneas de trabajo que siguen abiertas y que sería interesante abordar en un futuro próximo serían analizar la sensibilidad de las prestaciones con respecto a α y β y, posteriormente, establecer reglas complementarias para mejorar estos diseños.

Capítulo 4

Diseño de conjuntos de clasificadores binarizados pre-enfatizados

4.1. Introducción

Una alternativa diferente para la construcción de conjuntos de clasificadores consiste en descomponer, a través de técnicas de binarización, un problema original de múltiples clases en un conjunto de diferentes clasificadores que se entrenan como problemas de dos clases, que son más fáciles de discriminar.

La idea de combinar métodos de binarización con métodos de creación de conjuntos de máquinas de aprendizaje comentados en el Capítulo 1 (comités o consorcios), para resolver problemas multiclase no es algo novedoso. En [Leisch and Hornik, 1997], se estudiaron las prestaciones que presentaban la combinación de técnicas de diversificación convencionales −Bagging, en particular−, con técnicas de binarización sobre varios conjuntos de datos. Los resultados obtenidos no fueron muy satisfactorios, ya que eran muy similares a los obtenidos cuando únicamente se aplicaba Bagging. En [Zor et al., 2011], se evaluaron Bagging y conjuntos binarizados sobre nueve bases de datos utilizando un esquema de sesgo-varianza y comparando los resultados alcanzados con los que se obtendrían con un único clasificador basado en redes neuronales. Se comprobó que los resultados obtenidos superaban las prestaciones ofrecidas por

Bagging y técnicas de binarización en algunos casos, mientras que en otros proporcionaban prestaciones similares.

En este capítulo, damos un paso más y exploramos un procedimiento adicional que podría aplicarse junto con la combinación de técnicas de binarización y métodos de diversificación estándar, con el objetivo de mejorar aún más las prestaciones del conjunto de clasificadores: el pre-énfasis, o simplemente énfasis, que hemos presentado en el capítulo anterior.

El resto del capítulo se organiza de la siguiente manera: en la Sección 4.2 se presenta una breve revisión de los conjuntos creados a partir de técnicas de binarización en problemas multiclase. La Sección 4.3 define matemáticamente la función de énfasis utilizada en los experimentos. La Sección 4.4 describe las pruebas experimentales llevadas a cabo: conjuntos de datos seleccionados y arquitecturas, que elegimos basadas en MLPs, diseño de los conjuntos de clasificadores propuestos, método de binarización a aplicar, así como el tipo de máquina auxiliar que se utiliza para determinar cuánto enfatizar previamente cada muestra. Una breve discusión de los resultados obtenidos se incluye también en esta sección. Finalmente, en la Sección 4.5 se listan las conclusiones más importantes, discutiendo las ventajas y limitaciones de los métodos propuestos; así como se indican algunas líneas de investigación que quedan abiertas.

4.2. Binarización

Existen tres métodos bien conocidos para transformar un problema multiclase en un conjunto de problemas de clasificación binarios equivalente: 1) uno contra uno, OvO (del inglés *One vs. One*); 2) uno contra todos, OvR (del inglés *One vs. Rest*); y 3) códigos de corrección de errores, más comúnmente conocidos como ECOC (del inglés *Error Correcting Output Codes*) [Rokach, 2010].

Los códigos de corrección de errores se utilizan en la transmisión de datos, siendo objeto de investigación y nuevas implementaciones desde que nació la teoría de la

codificación. Estas técnicas se basan en añadir redundancia por parte del emisor para identificar y corregir posibles errores ocurridos en la transmisión. Así se consigue el envío de mensajes válidos a través de canales reales, donde se pueden producir errores durante la transmisión.

En el contexto de conjuntos, el uso de ECOC consiste en crear clasificadores base, o aprendices, y entrenarlos acorde a la información obtenida de una matriz de código preestablecida. Como se ha podido comprobar en trabajos experimentales previos, el uso de técnicas basadas en ECOC ofrece mejoras en comparación con los métodos de clasificación OvO u OvR [Rokach, 2010, Dietterich and Bakiri, 1995]. Esta es básicamente la razón por la que en esta Tesis Doctoral se ha optado por estudiar la aplicación de métodos ECOC para binarizar conjuntos de clasificadores.

A continuación, presentamos un breve resumen de las técnicas ECOC.

4.2.1. Códigos de corrección de errores (ECOC)

Dado un conjunto de C clases, la idea básica de los ECOC consiste en el diseño de un código por cada clase, comúnmente conocido como palabra código; estos códigos se almacenan en una matriz M $\in \{0,1\}^{C\times P}$ por filas, siendo C el número de clases y P la longitud del código (número de clasificadores binarios). A modo de ejemplo, la Tabla 4.1 muestra una matriz ECOC para un problema de clasificación de cuatro clases.

En el ejemplo que se muestra en la tabla anterior, a cada clase del problema se le asigna una cadena de bits única de longitud 7. Tal y como se ha comentado, esta cadena de bits se conoce como palabra código. A modo ilustrativo, la clase 2 (C_2) tiene asociada la palabra código 0011001, mientras que la clase 3 (C_3) tendría asociada la palabra código 0101010. Durante el proceso de entrenamiento, se diseñan tantos clasificadores binarios como columnas tenga la matriz mostrada en la Tabla 4.1, y se entrena cada uno de ellos de forma independiente. A este respecto, acorde con la primera columna de la matriz, se construye un clasificador binario para separar la clase 0 (C_0) de las clases 1, 2 y 3 (C_1, C_2 y C_3). Es decir, las clases se

Clase	Problema						
	P_0	P_1	P_2	P_3	P_4	P_5	P_6
C_0	1	1	1	1	1	1	1
C_1	0	0	0	0	1	1	1
C_2	0	0	1	1	0	0	1
C_3	0	1	0	1	0	1	0

Tabla 4.1: ECOC obtenido con un método exhaustivo para un problema de clasificación de cuatro clases. C_0 a C_3 son las clases y P_0 a P_6 representan los problemas binarios. La palabra código (fila de la matriz) que está más próxima al vector que se obtiene de las salidas de las unidades indica la clase que debe ser seleccionada.

re-etiquetan como pertenecientes a una de estas dos clases. Por tanto, se entrenarán siete clasificadores binarios. Para clasificar una nueva muestra $\mathbf{x}^{(n)}$, se evalúan los siete clasificadores binarios, y se obtiene una cadena de bits. Finalmente, la muestra se clasifica calculando la distancia o similitud entre la cadena de siete bits obtenida y la palabra código de cada clase, usando como métrica la distancia de Hamming.

4.3. Función de pre-énfasis

En el Capítulo 3 se describió la función de pre-énfasis propuesta en esta Tesis Doctoral para problemas de clasificación binarios. Extender dicha expresión a problemas multiclase no resulta inmediato, ya que el término $\left[1 - o_{m-1}^{(n)2}\right]$ no tiene una equivalencia directa en problemas de más de dos clases. Sin embargo, si consideramos formas discriminativas, las cuales son eficaces para entrenamiento multiclase, es posible reemplazar el término antes mencionado por $\left[1 - |o_{\text{ac}}^{(n)} - o_{\text{ac}'}^{(n)}|\right]$, o formas similares, donde $o_{\text{ac}}^{(n)}$ es la salida *softmax* del clasificador o máquina auxiliar para la clase verdadera, y $o_{\text{ac}'}^{(n)}$ es la salida de entre las unidades restantes cuyo valor está más próximo a $o_{\text{ac}}^{(n)}$.

Con esto, la función de pre-énfasis utilizada en problemas multiclase es la que se

muestra a continuación:

$$p\big(\mathbf{x}^{(n)}\big) = \alpha + (1-\alpha)\left[\beta\big(1-o_{\mathrm{ac}}^{(n)}\big)^2 + (1-\beta)\big(1-|o_{\mathrm{ac}}^{(n)} - o_{\mathrm{ac}'}^{(n)}|\big)\right] \tag{4.1}$$

Obviamente, hay muchas otras formas que podrían considerarse en la definición de $p\big(\mathbf{x}^{(n)}\big)$. Por ejemplo, el valor absoluto en la expresión $1 - |o_{\mathrm{ac}}^{(n)} - o_{\mathrm{ac}'}^{(n)}|$ podría sustituirse por el valor cuadrático, o $o_{\mathrm{ac}'}^{(n)}$ podría ser reemplazado por el valor de salida máximo correspondiente a las clases erróneas; y así, podrían explorarse más cambios. Sin embargo, en los experimentos llevados a cabo se ha comprobado que las diferencias obtenidas en los resultados con estas modificaciones son poco relevantes.

4.4. Pruebas experimentales

4.4.1. Bases de datos

Para la evaluación de los métodos propuestos, se han seleccionado un número reducido de bases de datos multiclase que se utilizan frecuentemente como problemas de referencia en este tipo de experimentos. La primera de ellas es un problema sintético: Firm-Teacher Clave-Direction Classification [Vurkac, 2011]; y las otras tres son problemas reales: Satimage [Giannakopoulos et al., 1999], Splice [Noordewier et al., 1991] y Vehicle [Siebert, 1987]. Todas ellas se han obtenido del repositorio de la Universidad de California en Irvine [Frank and Asuncion, 2010].

En la Tabla 4.2 se resumen las principales características de cada uno de estos problemas (dimensión, número de clases, así como el número de muestras para el conjunto de entrenamiento y test). Cuando las bases de datos no tienen un conjunto de entrenamiento y test predefinidos, como sucede con fir y veh, se crea de forma aleatoria una partición entrenamiento/test con 70 % de los datos para entrenamiento y 30 % para test, manteniendo la proporción relativa de cada clase en cada uno de los subconjuntos. En el Apéndice C se incluye una descripción más detallada de cada uno de estos problemas.

Finalmente, ha de comentarse que, tal y como sucedía con las bases de datos empleadas en el Capítulo 2, la notación o abreviatura con la que se va a denotar cada problema es sus tres primeras letras.

Base de datos	**Nº. muestras entrenamiento**	**Nº. muestras test**	**Dimensión**	**Nº. clases**
Firm-Teacher Clave-Direction (fir)	10800*	–	16	4
Satimage (sat)	4435	2000	36	6
Splice (spl)	3190*	–	60	3
Vehicle (veh)	946*	–	18	4

*: El número total de muestras de las bases de datos sin conjuntos de entrenamiento y test separados (fir, spl y veh), se muestra en la columna correspondiente al nº. de muestras de entrenamiento. Para estas bases de datos, se ha realizado una partición aleatoria con 70-30 % de datos para entrenamiento y test, respectivamente, manteniendo la proporción relativa de cada clase en cada subconjunto.

Tabla 4.2: Principales características de las bases de datos multiclase empleadas en la evaluación de los métodos propuestos.

4.4.2. Diseño de los clasificadores base

Las arquitecturas objeto de estudio en este capítulo, así como el diseño de un simple clasificador, hacen uso de MLPs de una sola capa oculta como clasificadores o aprendices base, ya que se trata de máquinas inestables que presentan buenas prestaciones. Los clasificadores se entrenan para minimizar el error cuadrático medio entre la salida deseada y la salida real del clasificador. Para ello, se inicializan aleatoriamente las componentes de los pesos del MLP con valores uniformemente distribuidos en el intervalo $[-0.2, 0.2]$, y se realiza su actualización mediante el algoritmo de retropropagación. La tasa de aprendizaje se fija en 0.01 (se ha comprobado experimentalmente que este valor es suficiente para asegurar la convergencia). Junto con el algoritmo de retropropagación, se emplea un algoritmo de detención prematura del entrenamiento, utilizando el 80 % de los datos disponibles para entrenar el MLP y

el 20 % restante para la detención, es decir, para elegir los pesos correspondientes a la época que presenta un menor error sobre esta partición, reduciendo así los efectos perniciosos del sobreajuste.

Como ya se mencionó en el Capítulo 2, los MLPs explorados en esta Tesis Doctoral consisten en una capa de entrada, una capa oculta y una capa de salida. En este tipo de arquitecturas, el número de neuronas en la capa de entrada, así como el número de neuronas en la capa de salida, están determinados por el tipo de problema a resolver. Concretamente, el número de neuronas en la capa de entrada coincide con el número de características o atributos extraídos de las muestras (es decir, es congruente con la dimensión de cada una de las bases de datos que se muestra en la Tabla 4.2), mientras que el número de neuronas de salida coincide con el número de clases a separar (nótese que en un problema de clasificación binaria sería suficiente con tener una única neurona de salida). El número de neuronas en la capa oculta (H) no está predeterminado. Con el objeto de conseguir un comportamiento óptimo del conjunto de clasificadores, debería fijarse un valor de H diferente para cada tipo de experimento. Sin embargo, a pesar de que cada conjunto pueda necesitar un valor diferente de H, para llevar a cabo una comparación equitativa entre las arquitecturas propuestas, se ha optado por fijar el mismo valor de H para todas las tasas de remuestreo de Bagging exploradas en los experimentos. Este valor de H se ha establecido mediante un proceso de validación cruzada de 5 particiones repetido 20 veces (20 veces “5-fold CV”).

No sucede lo mismo cuando se binariza el problema, es decir, cuando se generan diferentes clasificadores que se entrenan para problemas de dos clases. En este caso, para cada problema de clasificación dicotómica debe establecerse un valor de H diferente, que se obtiene mediante un 20 veces “5-fold”. Evidentemente, este tipo de conjuntos presentan un mayor coste computacional en operación.

4.4.3. Binarización

Según [Dietterich and Bakiri, 1995], para problemas de clasificación donde el número de clases es mayor o igual que 3 y menor o igual que 7[1], es decir, $3 \leq C \leq 7$, el procedimiento general hace uso de la siguiente expresión para calcular el número total P de clasificadores a construir:

$$P = 2^{(C-1)} - 1 \tag{4.2}$$

donde C es el número de clases del problema. Para formar cada palabra código se utiliza el método exhaustivo, el cual aplica el siguiente procedimiento:

- La fila 1 es todo unos.
- La fila 2 consiste de $2^{(C-2)}$ ceros, seguidos de $2^{(C-2)} - 1$ unos.
- La fila 3 consiste de $2^{(C-3)}$ ceros, seguidos de $2^{(C-3)}$ unos, $2^{(C-3)}$ ceros y finalmente $2^{(C-3)} - 1$ unos.
- La siguiente fila i se alternan $2^{(C-i)}$ ceros y unos, salvo el último bloque de unos que tiene longitud $2^{(C-i)} - 1$.
- La última fila es intercalada con unos y ceros.

4.4.4. Diversificación convencional

En la serie de experimentos llevados a cabo con Bagging, cada conjunto de entrenamiento se obtiene a partir de un remuestreo con reemplazo (*bootstrap*) del conjunto de entrenamiento original (es decir, puede haber muestras que se repitan) hasta llegar a un determinado porcentaje del número de muestras de partida. Una vez entrenados los clasificadores base, la salida de la arquitectura conjunta se obtiene bien promediando, bien por mayoría de las salidas de todos los clasificadores.

[1]Para un número de clases mayor que 7, se tendría un elevado coste computacional.

Las tasas de remuestreo que se han explorado en este trabajo han sido B = 60, 80, 100, 120 y 140 %. Asimismo, se han examinado diferentes tamaños del conjunto de clasificadores, concretamente, M = 11, 21, 31, 41, 51, 101, y 201. Como se verá más adelante, estos valores sirven para asegurar que los algoritmos propuestos alcanzan sus máximas prestaciones, ya que los resultados obtenidos se saturan a partir de un determinado número de clasificadores base.

Se estudiarán aquí dos modos diferentes de diversificación:

1. Construir conjuntos de clasificadores a partir de clasificadores base diversificados;

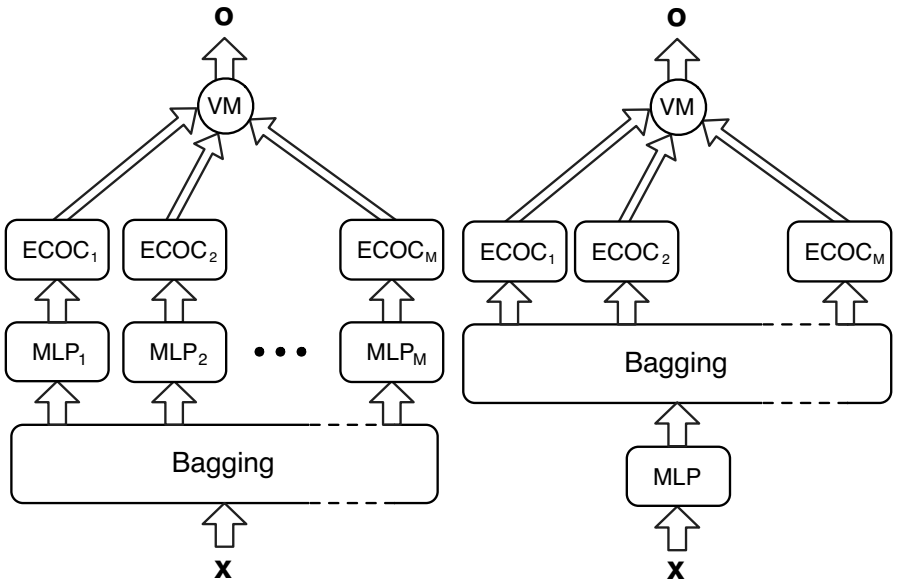
2. Aplicar diversificación en la binarización, después de obtener los valores de salida de un único MLP, que se entrena con el conjunto de entrenamiento original (sin diversificación).

De ahora en adelante, estas formas de diversificar conjuntos de clasificadores van a denotarse como MLP-O (del inglés *Overall*) y MLP-T (por su arquitectura en forma de T), respectivamente. Con esto en mente, cuando se aplica la técnica de Bagging a problemas multiclase, se obtienen dos arquitecturas:

- MLP-BINARIZADO-O: como se muestra en la Figura 4.1(a), en este caso, se aplica la técnica de Bagging a los datos de entrada para construir conjuntos de M clasificadores base, y la clase final se decide por voto por mayoría de las salidas que se obtienen de los conjuntos binarizados por ECOC que siguen a cada MLP diversificado.

- MLP-BINARIZADO-T: en este caso, como se ilustra en la Figura 4.1(b), existe un único clasificador base (un MLP), y se aplica Bagging a sus valores de salida para construir M conjuntos ECOC diversificados, cuyas salidas se agregan; la decisión final se obtiene, también, utilizando voto por mayoría.

Evidentemente, las arquitecturas en O necesitan un mayor esfuerzo computacional para su diseño, ya que es necesario entrenar M MLPs. Con objeto de reducir este



(a) (b)

Figura 4.1: Ilustración de las dos formas de combinar las técnicas de binarización y diversificación estudiadas para resolver problemas multiclase: arquitectura MLP-BINARIZADO-O (Figura 4.1(a)) y arquitectura MLP-BINARIZADO-T (Figura 4.1(b)). MLP representa un único Perceptrón Multicapa; ECOC_n son los conjuntos ECOC diversificados. VM: voto por mayoría; **x**: entrada; **o**: decisión de la clase de salida.

coste computacional, se hace uso de una simplificación frecuentemente usada en este tipo de experimentos: se diseñan y se entrenan, utilizando Bagging, más clasificadores base que los necesarios (en este caso, concretamente, se han diseñado y entrenado 450 máquinas[2]) para cada valor de la tasa de remuestreo (B), y se seleccionan, de entre todos ellos, M de forma aleatoria. Los resultados mostrados son los obtenidos después de promediar sobre 10 repeticiones. Obviamente, esta simplificación hará que se reduzca ligeramente la ventaja obtenida por usar diversificación; pero, como ya se ha comentado, el objetivo aquí es comprobar si realmente la diversificación de clasificadores puede mejorar los resultados de un clasificador simple.

4.4.5. Pre-énfasis

A la hora de entrenar un único clasificador, a cada muestra se le aplica el peso asignado por la función de énfasis descrita en (4.1). Sin embargo, si se observan las arquitecturas planteadas en la sección anterior (MLP-BINARIZADO-O y MLP-BINARIZADO-T), el lector podría preguntarse cuál sería la forma apropiada de aplicar a las muestras los pesos de la función de pre-énfasis. En el primer escenario, estructura MLP-BINARIZADO-O, parece evidente que los pesos se aplican a las muestras de entrada de cada MLP diversificado. En el segundo caso, no es tan sencillo saber cómo aplicar los pesos asignados. Para esta arquitectura, se han considerado dos posibles opciones en los experimentos: 1) Aplicar los pesos únicamente a las muestras de entrada del clasificador basado en el MLP, ó 2) Enfatizar las muestras de entrada de cada clasificador binario que surge después de aplicar la técnica de Bagging a los valores de salida del clasificador basado en el MLP. En una serie de experimentos preliminares realizados, se comprobó que se obtenían ligeras ventajas cuando se enfatizaban las muestras de entrada de cada clasificador binario. En consecuencia, en la parte experimental de este capítulo, para la arquitectura MLP-BINARIZADO-T se ha optado por enfatizar las muestras de entrada de los problemas

[2]Nótese que el número máximo de clasificadores explorado en los experimentos es M = 201.

de clasificación binarios.

Para determinar los valores de los parámetros de mezcla de la función de énfasis descrita en (4.1), α y β, se ha llevado a cabo un proceso de 5 particiones del conjunto de entrenamiento original y los valores explorados son de 0 a 1 en pasos de 0.1 para las cuatro bases de datos bajo estudio. Se han realizado 10 inicializaciones (repeticiones) independientes del MLP por cada partición de la CV.

Finalmente, cabe mencionar que se utiliza un MLP con 50 neuronas en la capa oculta como máquina auxiliar para determinar la distribución de pesos o de énfasis sobre las muestras. En la parte experimental de este capítulo se ha comprobado, para las cuatro bases de datos, sobre un razonable conjunto de validación, que este número de neuronas es adecuado para obtener un clasificador claramente competitivo en cuanto a prestaciones.

4.4.6. Resultados

Las Tablas 4.3, 4.4, 4.5 y 4.6 muestran las tasas de error de clasificación en test (además de las desviaciones estándar) después de promediar sobre 10 selecciones diferentes de clasificadores base agregados cuando se utiliza la técnica de Bagging ó 10 realizaciones distintas de los experimentos en conjuntos ECOC, para las bases de datos fir, sat, spl y veh, respectivamente. Para facilitar la comparación entre las distintas alternativas, se resalta en negrita el mejor resultado (el que presenta el menor error de clasificación), incluso aunque las diferencias estadísticas no sean significativas. Adicionalmente, se muestran también, para los mismos problemas de clasificación, los resultados conseguidos con un único clasificador MLP, así como los alcanzados con un conjunto de clasificadores base cuando solamente se utiliza la técnica de Bagging para agregarlos (indicado como "MLP Bagging" en las tablas) o cuando únicamente se emplean ECOC para reducir un problema multiclase a un conjunto de clasificadores binarios (señalado como "MLP ECOC" en las tablas). El objetivo de incluir estos resultados es poder apreciar, de forma clara, cuáles son las mejoras (en términos de reducción del error de clasificación) aportadas por la

combinación de las distintas alternativas de diversificar problemas de clasificación.

Como se ha mencionado en la subsección 4.4.4, aunque en la serie de experimentos llevados a cabo se han explorado diferentes tamaños de los conjuntos de entrenamiento usando Bagging (concretamente, se han obtenido resultados con tasas de remuestreo de B = 60, 80, 100, 120 y 140 %), así como distintos tamaños del conjunto de clasificadores (M = 11, 21, 31, 41, 51, 101, y 201), las Tablas 4.3, 4.4, 4.5 y 4.6 ilustran los resultados que suponen una reducción notable del error con respecto al MLP estándar, indicando los valores del parámetro de remuestreo y de M.

En una primer observación de las tablas, es fácil comprobar que se manifiestan efectos de saturación en las prestaciones de los diseños cuando el valor de M o el de B aumentan. Esto significa que no es necesario realizar más experimentos ampliando los márgenes de exploración de estos parámetros. Además, para las cuatro bases de datos bajo estudio, comparando las alternativas propuestas con un MLP estándar o cuando se utiliza únicamente Bagging o ECOC, se comprueba que ambas alternativas, en muchos casos, superan las prestaciones de dicho MLP, lo que ilustra la eficacia de estos métodos a la hora de explotar las ventajas potenciales de combinar las diferentes formas de diversificación estudiadas en este capítulo. Como simple ejemplo ilustrativo, en el problema veh, con la arquitectura MLP-BINARIZADO-T, se consigue una reducción del error de aproximadamente 30 % en comparación con un MLP estándar.

Obviamente, es preferible emplear los diseños con la arquitectura en forma de T, ya que requieren menor esfuerzo computacional durante la fase de diseño y de operación. Nótese que no existen diferencias relevantes entre los resultados obtenidos cuando se utilizan, por separado, las técnicas de diversificación exploradas (Bagging o ECOC).

Analizando más en detalle los resultados de las Tablas 4.3, 4.4, 4.5 y 4.6, pueden extraerse las siguientes conclusiones:

- Para la base de datos **fir**: aplicar directamente Bagging o generar diferentes clasificadores binarios en un contexto de múltiples clases (método ECOC) con-

MLP	14.66 ± 0.18		
MLP-ECOC	12.03 ± 0.13		
MLP bagging	B	B	B
M	100 %	120 %	140 %
11	13.09 ± 0.13	12.87 ± 0.11	12.75 ± 0.15
21	12.94 ± 0.14	12.73 ± 0.10	**12.66 ± 0.14**
31	12.97 ± 0.12	12.83 ± 0.09	12.71 ± 0.12
MLP-BINARIZED-O bagging	B	B	B
M	100 %	120 %	140 %
11	12.75 ± 0.13	12.04 ± 0.11	12.13 ± 0.10
21	12.64 ± 0.11	**11,70 ± 0,10**	12.01 ± 0.11
31	12.58 ± 0.09	11.83 ± 0.09	11.97 ± 0.09
MLP-BINARIZED-T bagging	B	B	B
M	100 %	120 %	140 %
21	12.47 ± 0.07	11.65 ± 0.08	11.97 ± 0.08
31	12.35 ± 0.07	**11.53 ± 0.07**	11.86 ± 0.07
51	12.33 ± 0.07	11.60 ± 0.07	11.75 ± 0.08

Tabla 4.3: Tasa de error de clasificación (± desviación estándar) sobre datos de test de la base de datos **fir**, para los métodos de diversificación convencionales Bagging y ECOC (utilizando como clasificadores base MLPs), así como para las diferentes formas de combinarlos estudiadas en esta Tesis Doctoral.

MLP	10.28 ± 0.98		
MLP-ECOC	9.44 ± 0.40		
MLP bagging	B	B	B
M	100 %	120 %	140 %
11	10.58 ± 0.84	10.04 ± 0.63	10.14 ± 0.60
21	10.39 ± 0.71	**9.88 ± 0.55**	10.19 ± 0.62
31	10.43 ± 0.69	9.94 ± 0.57	10.21 ± 0.55
MLP-BINARIZED-O bagging	B	B	B
M	100 %	120 %	140 %
11	10.34 ± 0.44	**9.24 ± 0.38**	9.35 ± 0.39
21	10.23 ± 0.37	9.31 ± 0.42	9.29 ± 0.37
31	10.19 ± 0.35	9.29 ± 0.36	9.33 ± 0.36
MLP-BINARIZED-T bagging	B	B	
M	120 %	140 %	
11	9.31 ± 0.39	9.23 ± 0.34	
21	9.29 ± 0.36	**9.18 ± 0.29**	
31	9.25 ± 0.31	9.22 ± 0.28	

Tabla 4.4: Tasa de error de clasificación (± desviación estándar) sobre datos de test de la base de datos **sat**, para los métodos de diversificación convencionales Bagging y ECOC (utilizando como clasificadores base MLPs), así como para las diferentes formas de combinarlos estudiadas en esta Tesis Doctoral.

MLP	26.11 ± 1.20		
MLP-ECOC	24.42 ± 1.72		
MLP bagging	B	B	B
M	100 %	120 %	140 %
51	21.73 ± 0.58	22.52 ± 0.31	23.28 ± 0.22
121	**21.61 ± 0.22**	22.52 ± 0.38	23.25 ± 0.26
151	21.65 ± 0.24	22.49 ± 0.38	23.27 ± 0.34
MLP-BINARIZED-O bagging	B	B	B
M	100 %	120 %	140 %
121	20.63 ± 0.51	19.95 ± 0.44	19.76 ± 0.45
161	20.62 ± 0.42	20.09 ± 0.41	**19.62 ± 0.39**
201	20.57 ± 0.37	20.03 ± 0.41	19.76 ± 0.39
MLP-BINARIZED-T bagging	B	B	B
M	100 %	120 %	140 %
151	20.59 ± 0.45	20.02 ± 0.47	19.77 ± 0.34
171	20.65 ± 0.42	20.03 ± 0.37	**19.71 ± 0.37**
191	20.56 ± 0.35	19.98 ± 0.38	19.82 ± 0.32

Tabla 4.5: Tasa de error de clasificación (± desviación estándar) sobre los datos de test de la base de datos **spl**, para los métodos de diversificación convencionales Bagging y ECOC (utilizando como clasificadores base MLPs), así como para las diferentes formas de combinarlos estudiadas en esta Tesis Doctoral.

MLP	18.91 ± 3.79		
MLP-ECOC	14.20 ± 3.40		
MLP bagging	B		B
M	120 %		140 %
51	14.93 ± 3.41		14.73 ± 3.16
101	14.80 ± 3.35		**14.69 ± 3.11**
201	14.89 ± 3.27		14.75 ± 3.08
MLP-BINARIZED-O bagging	B	B	B
M	100 %	120 %	140 %
31	14.72 ± 3.24	13.59 ± 3.33	13.82 ± 3.20
51	14.68 ± 3.19	**13.51 ± 3.29**	13.85 ± 3.21
101	14.51 ± 3.20	13.63 ± 3.26	13.80 ± 3.18
MLP-BINARIZED-T bagging	B	B	B
M	100 %	120 %	140 %
31	14.87 ± 3.29	13.19 ± 3.30	13.42 ± 3.25
51	14.78 ± 3.19	**13.04 ± 3.25**	13.18 ± 3.14
101	14.80 ± 3.15	13.08 ± 3.22	13.16 ± 3.11

Tabla 4.6: Tasa de error de clasificación (± desviación estándar) sobre los datos de test de la base de datos **veh**, para los métodos de diversificación convencionales Bagging y ECOC (utilizando como clasificadores base MLPs), así como para las diferentes formas de combinarlos estudiadas en esta Tesis Doctoral.

sigue mejorar, en los dos casos, la tasa de error de clasificación de un clasificador MLP estándar. Puede observarse que simplemente usando el método ECOC, se logran reducciones de tasas de error de hasta el 18 %, mientras que, en el mejor escenario, la técnica de Bagging consigue una reducción del 14 %. Sin embargo, combinar los métodos Bagging y ECOC ofrece resultados ligeramente mejores. Como puede observarse en la Tabla 4.3, en la arquitectura en forma de T, para M = 31 y B = 120 %, se consigue reducir la tasa de error hasta el 22 %.

- Para la base de datos **sat**: aplicar, por separado, los métodos de Bagging y ECOC consigue mejorar la tasa de error, siendo el conjunto de clasificadores construido según el procedimiento de ECOC el que obtiene una mayor reducción de la misma ($\approx 8\,\%$) con respecto al MLP estándar, tal y como ocurre con la base de datos anterior. Debemos indicar, no obstante, que para esta base de datos, cuando se combinan ambos métodos, no se obtienen prestaciones de clasificación superiores a las alternativas de diversificación convencionales consideradas por separado cuando se emplean tasas de remuestreo (B) pequeñas. Sin embargo, a medida que el valor de B aumenta, se consigue reducir de forma notable la tasa de error, obteniéndose la máxima ventaja, nuevamente, con la arquitectura en forma de T, para M = 21 y B = 140 % ($\approx 11\,\%$).

- Para la base de datos **spl**: para esta base de datos, aplicar directamente la técnica de Bagging permite lograr una reducción del error de clasificación de aproximadamente 17 %, mientras que, como era de esperar, la combinación de binarización y diversidad, utilizando la arquitectura en forma de T, consigue reducir todavía más la tasa de error, siendo ésta aproximadamente del 25 %.

- Para la base de datos **veh**: por un lado, simplemente mediante la técnica de ECOC se consigue una considerable reducción de la tasa de error ($\approx 25\,\%$), que supera, en todos los casos, a las ventajas que el Bagging puede ofrecer (tasas de error en torno al 21-22 %). Por otro lado, combinando ambos métodos, se ve cómo disminuye el error de clasificación, llegándose a resultados mode-

radamente buenos, especialmente cuando los valores de M y B son elevados. Concretamente, para la arquitectura en forma de T y con M= 51 clasificadores y una tasa de remuestreo de B= 120 %, se consigue una reducción del error de aproximadamente 31 %.

Con respecto a esta última base de datos, debemos tener en cuenta que, cuando se emplea un número reducido de muestras, los resultados que se obtiene son más inestables y las diferencias son estadísticamente menos significativas.

Base de datos	MLP		MLP-BINARIZADO-T bagging	
	No PrE	PrE (α, β)	No PrE	PrE (α, β)
Fir	14.66 ± 0.18	11.87 ± 0.05 (0.3, 0.5)	11.53 ± 0.07	10.34 ± 0.03 (0.4, 0.6)
Sat	10.28 ± 0.98	9.26 ± 0.31 (0.4, 0.3)	9.18 ± 0.29	8.13 ± 0.17 (0.4, 0.5)
Spl	26.11 ± 1.20	22.09 ± 0.14 (0.4, 0.3)	19.71 ± 0.37	17.87 ± 0.16 (0.2, 0.4)
Veh	18.91 ± 3.79	14.05 ± 2.32 (0.6, 0.3)	13.04 ± 3.25	11.59 ± 1.67 (0.5, 0.4)

Tabla 4.7: Comparación de la tasa de error de clasificación (± desviación estándar) sobre los datos de test, para las bases de datos **fir**, **sat**, **spl** y **veh**, cuando se aplica el método de énfasis.

Aunque los diseños propuestos presentan prestaciones excelentes, se ha decidido dar un paso más en los experimentos y aplicar la función de énfasis presentada en (4.1) a los diseños que consiguen los mejores resultados para las bases de datos bajo estudio. Los resultados obtenidos se muestran en la Tabla 4.7. Para facilitar la comparación, se incluyen también los resultados conseguidos cuando no se aplica la

función de énfasis a estos diseños[3], así como las prestaciones de un clasificador MLP estándar en el que se aplica la distribución de pesos o de énfasis sobre las muestras (señalado como "MLP PrE" en la tabla), y también, cuando no se enfatizan las muestras (indicado simplemente como "MLP" en la tabla)[3]. Los valores de α y β usados en cada diseño se han seleccionado por CV, y se indican entre paréntesis.

Puede observarse que los diseños MLP-BINARIZADO-T, cuando se enfatizan previamente las muestras, permiten obtener sistemáticamente mejores prestaciones. En los casos de las bases de datos fir y spl, se consigue una reducción del error de clasificación aproximadamente del 30 % y 32 %, respectivamente, en comparación con el MLP estándar. Además, en el caso de veh, se logra que la tasa de error decaiga hasta un 40 % con respecto al clasificador MLP.

Aparte de lo anterior, cabe mencionar que el uso del énfasis, en técnicas de diversificación convencionales y conjuntos binarizados por ECOC, así como en las diferentes formas de combinación propuestas, consigue mayores ventajas en todas las bases de datos bajo estudio. También, se puede observar que los valores de α y β varían entre 0.2 y 0.6 para todos los problemas de clasificación, lo que indica la importancia de aplicar la función de énfasis propuesta.

4.5. Conclusiones

En este capítulo se han analizado las prestaciones que ofrece la combinación de las técnicas de binarización con técnicas de diversificación convencional −Bagging, concretamente−, y además, se ha corroborado experimentalmente cómo ponderando las muestras de entrenamiento según una función de énfasis convenientemente diseñada se consigue mejorar aún más las prestaciones de los diseños propuestos.

Se han planteado dos diseños que permiten realizar la agrupación de clasificadores basados en MLPs: arquitectura en forma de O, correspondiente a una diversificación completa de todos los MLPs, y arquitectura en forma de T, en la que la diversidad

[3]Información extraída de las Tablas 4.3, 4.4, 4.5 y 4.6.

se aplica después de entrenar un único clasificador basado en un MLP estándar.

En primer lugar, para todas las bases de datos bajo estudio, se ha comprobado experimentalmente cómo cualquiera de los esquemas de combinación propuestos en este capítulo consigue mejorar los resultados que se obtendrían con un único MLP estándar. Concretamente, se consiguen reducciones de error entre un mínimo de 11 % para la base de datos sat y un máximo de 31 % para la base de datos veh. Asimismo, se ha verificado que estos esquemas de combinación superan la tasa de error presentada por el mejor conjunto de MLPs cuando se aplican, separadamente, Bagging o técnicas ECOC, en todos los casos, lo que evidencia la utilidad de estos diseños. Si además comparamos entre sí las dos arquitecturas propuestas, queda claro cuál es mejor: el diseño en forma de T es el que presenta menor tasa de error de clasificación en la mayoría de los casos, existiendo diferencia estadística significativa entre los resultados obtenidos. Además, este tipo de arquitectura presenta una clara ventaja de orden práctico, en comparación con el diseño en O, ya que supone una reducción importante en el esfuerzo computacional durante su diseño (nótese que sólo es necesario entrenar un único MLP, y no un conjunto de ellos como sucede en el diseño en forma de O). Es importante mencionar también que en los experimentos llevados a cabo se han observado claros efectos de saturación con respecto a la tasa de remuestreo (B) y número de máquinas del conjunto (M) en las prestaciones de cualquiera de los dos diseños, lo que significa que la selección de los parámetros de diversificación es un problema de validación sencillo.

A continuación, se han explorado las ventajas de aplicar una función de énfasis lo suficientemente flexible al diseño que mejores resultados proporcionaba, es decir, la arquitectura en forma de T, y se han obtenido aún mejores prestaciones. Para la base de datos veh, se ha conseguido reducir el error de un 31 %, que era lo que se obtenía con el diseño MLP-BINARIZADO-T, hasta un 40 %, simplemente aplicando pre-énfasis. Esto evidencia la utilidad de enfatizar las muestras de conjuntos de clasificadores basados en MLPs.

Convendría, como ampliación de estudio, considerar otros tipos de problemas,

otros clasificadores y otras fuentes de diversidad, lo que permitiría apreciar las ventajas potenciales que la combinación de técnicas de binarización y diversidad, incluyendo una apropiada y eficiente distribución de pesos o de énfasis sobre las muestras, puede proporcionar en general. Finalmente, debe destacarse además que combinar este tipo de diseños con técnicas adicionales que sirvan para aumentar las prestaciones de los clasificadores basados en MLPs sería una excelente forma de conseguir mejorar los resultados obtenidos aún más, cuando el problema reclame un diseño de altas prestaciones.

Capítulo 5

Conclusiones: aportaciones y posteriores líneas y ámbitos de trabajo

Breve será la exposición de las conclusiones de esta Tesis Doctoral, porque ya se han discutido las aportaciones propias de cada capítulo (2, 3 y 4) al final del mismo, así como unas extensiones inmediatas. Algún espacio más se dedicará a las futuras líneas de trabajo que surgen de combinar lo expuesto por separado, así como a lo que en realidad constituyen posibles ámbitos de investigación y no nuevas líneas.

Una aclaración inexcusable antes de pasar a la arriba aludida exposición: la presentación de los trabajos se ha dividido en tres capítulos:

- Aplicación de métodos de proximidad (k-NN) en la concepción de un nuevo algoritmo de Boosting, para reducir los indeseables efectos que en los procedimientos clásicos de dicha familia de técnicas de construcción de conjuntos tienen las muestras fuera de margen, en el Capítulo 2;

- Modificación de la generalización del algoritmo de Boosting propuesto en [Gómez-Verdejo et al., 2006, Gómez-Verdejo et al., 2008] para prestar atención también a las muestras próximas a la (deseada) frontera de clasificación, incluyendo en su énfasis un término constante que permite regular la ponderación, en el Capítulo 3;

– Combinación de una técnica de diversificación clásica –Bagging– con otra mediante binarización –concretamente, ECOC–, y de ambas con un pre-énfasis de las muestras obtenido mediante un clasificador auxiliar, en el Capítulo 4.

La razón para proceder así no radica en que se trate de asuntos aislados, sino en que se desea la mayor claridad posible; pero la conexión entre los contenidos de los tres capítulos es clara e intensa: siempre se trata con conjuntos, la proximidad temática de los Capítulos 2 y 3 es obvia –siempre Boosting–, y en el cuarto capítulo se emplea, como sustituto de los énfasis del Boosting –que no tiene encaje directo, ya que la diversificación se deriva de otras fuentes–, un pre-énfasis guiado por un clasificador auxiliar: su equivalente conceptual en ese contexto.

Las principales aportaciones de la Tesis son:

* Demostrar que la adición de mecanismos de detección implícita de muestras fuera de margen –concretamente, la asociación k-NN– resulta eficaz para reducir el negativo efecto que dichas muestras tiene en el diseño de conjuntos mediante técnicas de Boosting (si bien con un sensible incremento de la carga computacional de diseño);

* Verificar que la adición de un término constante para controlar la intensidad del énfasis que se aplica en la construcción de conjuntos mediante Boosting, es decir, para limitar la atención a las muestras erróneas y a las próximas a la frontera, permite diseños con prestaciones mejoradas, y ello, con un moderado incremento del esfuerzo computacional para el diseño, nacido de la necesidad de extender la validación cruzada a un parámetro más, pero acotado entre 0 y 1;

* Comprobar que la combinación de diversidad convencional y binarización produce mejoras de prestaciones; y que, si a ello se añade la aplicación de un pre-énfasis a las muestras de entrenamiento según lo que indique un clasificador auxiliar, las mejoras son aún mayores.

Quienquiera que trabaje concibiendo o aplicando Máquinas de Aprendizaje apreciará la relevancia de tales aportaciones; muy en particular, si lo hace en el ámbito de diversificación y diseño de conjuntos de máquinas.

En cuanto a las extensiones inmediatas de los trabajos expuestos, repetiremos lo dicho al final de los capítulos 3 y 4:

* Analizar la sensibilidad del énfasis general propuesto para Boosting con respecto a sus parámetros, α y β; y posibilidad de añadir reglas complementarias o considerar variantes potencialmente más flexibles. Como ejemplo: cabe combinar este énfasis con la variante k-NN del Capítulo 2 –aunque la carga computacional de diseño crecerá de tal modo que sólo resultará razonable para problemas de alto valor.

* Aplicar combinaciones de otros tipos de diversificación convencional o/y otros tipos de binarización, con otros tipos de máquinas base cuando proceda, junto con mecanismos de pre-énfasis, a los problemas considerados aquí, otros similares, y otros de diferentes características, a los efectos de determinar la utilidad general de la triple combinación y dónde se hallan sus límites.

Queda claro también que pueden concebirse y evaluarse formas alternativas al KRAB (Capítulo 2).

Adicionalmente, han de señalarse aquí como posibles ampliaciones:

* Buscar otras formas de pre-énfasis que puedan incrementar los efectos favorables de su aplicación: por ejemplo, mediante la incorporación de umbrales que permitan reducir la carga computacional de diseño y, al tiempo, una adicional mejora de prestaciones por efecto de los "márgenes" que implicaría esa incorporación. También sería de interés estudiar si versiones que distinguiesen la parametrización según la clase de las muestras de entrenamiento podrían ser útiles para abordar problemas desequilibrados (nótese que el diseño final es independiente de los valores de dichos parámetros).

* Muy en particular, sería importante extender los estudios de diversificación/binarización y pre-énfasis –con las variantes que se apreciasen como útiles en los análisis anteriormente propuestos– a DNNs (Redes Neuronales Profundas, *Deep Neural Networks*). Si bien ya se ha concluido un importante estudio en este ámbito, la Tesis Doctoral [Alvear-Sandoval, 2017], ésta considera el caso de DNNs representacionales –concretamente, las basadas en auto-codificación ruidosa–, y es más que aconsejable determinar si sus (positivas) conclusiones pueden extenderse a otros tipos, tales como las convolucionales o las que permitan un entrenamiento directo; todo ello, además de comparar las alternativas de diversificación/binarización y pre-énfasis que vayan demostrando ventaja.

* Por último, un ámbito completo que aquí simplemente ha asomado: un método generativo, k-NN, se ha manifestado útil para mejorar uno discriminativo –Boosting–; lo que demuestra que no hay que limitarse al aprendizaje ruidoso en lo que se refiere a la combinación de técnicas generativas, o de modelo, con técnicas discriminativas, a orientadas a problema, en el diseño de Máquinas de Aprendizaje. Nos llevaría muy lejos, fuera del contexto de la Tesis, apuntar las muchas posibilidades que se abren: por ello, nos limitamos a aseverar que combinar ambas familias de métodos puede hacerse de muy diversas maneras, y promete abrir caminos para reducir los inconvenientes de cada una de ellas (principalmente, robustez y altas prestaciones, respectivamente).

Las publicaciones de resultados de estos trabajos de Tesis Doctoral realizadas hasta ahora son:

Ahachad, A., Omari, A. and Figueiras-Vidal, A. R. (2013). Smoothed Emphasis for Boosting Ensembles. In Proceedings of the 12th International Work-Conference on Artificial Neural Networks (IWANN'13), Ignacio Rojas, Gonzalo Joya, and Joan

Gabestany (Eds.), Vol. Part I. Puerto de la Cruz, Tenerife, Spain, June 12-14, 2013, 367-375.

Ahachad, A., Omari, A. and Figueiras-Vidal, A. R. (2015). Neighborhood Guided Smoothed Emphasis for Real Adaboost Ensembles. *Neural Processing Letters*, 42: 155–165.

Ahachad, A., Álvarez-Pérez, L. and Figueiras-Vidal, A. R. (2017). Boosting ensembles with controlled emphasis intensity. *Pattern Recognition Letters*, 88: 1–5.

Álvarez-Pérez, L., Ahachad, A. and Figueiras-Vidal, A. R. (2017). Pre-emphasizing Binarized Ensembles to Improve Classification Performance. In Proceedings of the 14th International Work-Conference on Artificial Neural Networks (IWANN'17), Ignacio Rojas, Gonzalo Joya, and Joan Gabestany (Eds.), Vol. Part I. Cadiz, Spain, June 14-16, 2017, 339-350.

En estos momentos se considera la preparación y envío de un artículo de revista que complete lo que se expone en la última comunicación al IWANN.

Parte I

Apéndices

Apéndice A

El algoritmo "Real AdaBoost"

Este algoritmo construye de forma secuencial los aprendices $\{o_m(\mathbf{x})\}$ y los combina linealmente

$$f_m(\mathbf{x}) = \sum_{m'=1}^{m} \alpha_{m'} o_{m'}(\mathbf{x}) \tag{A.1}$$

minimizando una cota superior del coste muestral exponencial del margen

$$\begin{aligned} C_m &= \sum_{n=1}^{N} \exp\left(-t^{(n)} \sum_{m'=1}^{m} \alpha_{m'} o_{m'}^{(n)}\right) = \\ &= \sum_{n=1}^{N} C_{m-1}^{(n)} \exp\left(-\alpha_m t^{(n)} o_m^{(n)}\right) \end{aligned} \tag{A.2}$$

que se maneja escalada

$$C'_m = \sum_{n=1}^{N} p_{m-1}^{(n)} \exp\left(-\alpha_m t^{(n)} o_m^{(n)}\right) \tag{A.3}$$

siendo

$$p_{m-1}^{(n)} = C_{m-1}^{(n)} \Big/ \sum_{n'=1}^{N} C_{m-1}^{(n')} \tag{A.4}$$

$\left(\text{así, } \sum_{n=1}^{N} p_{m-1}^{(n)} = 1\right)$.

La cota superior a minimizar se extrae de la desigualdad

$$\mathrm{g(u)} = \left[\frac{1+\mathrm{u}}{2}\exp(-\alpha) + \frac{1-\mathrm{u}}{2}\exp(\alpha)\right] - \exp(-\alpha \mathrm{u}) \geq 0$$

si $\mathrm{u} \in [-1, 1]$: para validarla, basta comprobar que $\mathrm{g(u)}$ es convexa ($\partial^2 \mathrm{g}/\partial \mathrm{u}^2 < 0$) y $\mathrm{g}(\pm 1) = 0$.

Así, pues, el proceso de diseño consiste en:

$$\{\alpha_\mathrm{m}, \mathrm{o_m}\} = \arg \min_{\{\alpha'_\mathrm{m}, \mathrm{o'_m}\}} \sum_{\mathrm{n}=1}^{\mathrm{N}} \mathrm{p}_{\mathrm{m}-1}^{(\mathrm{n})} \left[\frac{1+\mathrm{t}^{(\mathrm{n})}\mathrm{o}_\mathrm{m}^{\prime(\mathrm{n})}}{2}\exp(-\alpha'_\mathrm{m}) + \frac{1-\mathrm{t}^{(\mathrm{n})}\mathrm{o}_\mathrm{m}^{\prime(\mathrm{n})}}{2}\exp(\alpha'_\mathrm{m})\right] \tag{A.5}$$

Considerando sólo términos dependientes de o'_m, ha de minimizarse

$$\frac{1}{2}\left[\exp(-\alpha'_\mathrm{m}) - \exp(\alpha'_\mathrm{m})\right] \sum_{\mathrm{n}=1}^{\mathrm{N}} \mathrm{p}_{\mathrm{m}-1}^{(\mathrm{n})} \mathrm{t}^{(\mathrm{n})} \mathrm{o}_\mathrm{m}^{\prime(\mathrm{n})} \tag{A.6}$$

lo que (admitiendo, naturalmente, que $\alpha'_\mathrm{m} > 0$) equivale a

$$\mathrm{o_m} = \arg \max_{\mathrm{o}'_1} \sum_{\mathrm{n}=1}^{\mathrm{N}} \mathrm{p}_{\mathrm{m}-1}^{(\mathrm{n})} \mathrm{t}^{(\mathrm{n})} \mathrm{o}_\mathrm{m}^{\prime(\mathrm{n})} \tag{A.7}$$

lo que equivale a entrenar el aprendiz m para maximizar el margen de la población enfatizada.

En la práctica, para evitar problemas de parálisis ($|\mathrm{o_m}|$ no queda acotada si se pretende (A.6)), se emplean otros modos (convencionales) de entrenamiento del aprendiz.

Determinado $\mathrm{o_m}$, queda

$$\begin{aligned}
\alpha_\mathrm{m} = \arg \min_{\alpha'_\mathrm{m}} \Big[& \exp(-\alpha'_\mathrm{m}) \sum_{\mathrm{n}=1}^{\mathrm{N}} \mathrm{p}_{\mathrm{m}-1}^{(\mathrm{n})} \left(1 + \mathrm{t}^{(\mathrm{n})} \mathrm{o}_\mathrm{m}^{(\mathrm{n})}\right) + \\
& + \exp(\alpha'_\mathrm{m}) \sum_{\mathrm{n}=1}^{\mathrm{N}} \mathrm{p}_{\mathrm{m}-1}^{(\mathrm{n})} \left(1 - \mathrm{t}^{(\mathrm{n})} \mathrm{o}_\mathrm{m}^{(\mathrm{n})}\right) \Big] = \\
& = \arg \min_{\alpha'_\mathrm{m}} \Big[\exp(-\alpha'_\mathrm{m})(1+\gamma_\mathrm{m}) + \exp(\alpha'_\mathrm{m})(1-\gamma_\mathrm{m}) \Big]
\end{aligned} \tag{A.8}$$

siendo

$$\gamma_{\mathrm{m}} = \sum_{\mathrm{m}} \mathrm{p}_{\mathrm{m}-1}^{(\mathrm{n})} \mathrm{t}^{(\mathrm{n})} \mathrm{o}_{\mathrm{m}}^{(\mathrm{n})} \tag{A.9}$$

el parámetro de separación o corte: una medida, ponderada según el énfasis, de la correlación de las salidas del aprendiz y las etiquetas.

Anulando la derivada de la función a minimizar en (A.8), se obtiene de inmediato

$$\alpha_{\mathrm{m}} = \frac{1}{2} \ln \frac{1+\gamma_{\mathrm{m}}}{1-\gamma_{\mathrm{m}}} \tag{A.10}$$

Finalmente, observando que, según lo anterior, $\mathrm{p}_{\mathrm{m}}^{(\mathrm{n})}$ se puede obtener de $\mathrm{p}_{\mathrm{m}-1}^{(\mathrm{n})}$ en el modo

$$\mathrm{p}_{\mathrm{m}}^{(\mathrm{n})} = \mathrm{p}_{\mathrm{m}-1}^{(\mathrm{n})} \exp\left(-\alpha_{\mathrm{m}} \mathrm{t}^{(\mathrm{n})} \mathrm{o}_{\mathrm{m}}^{(\mathrm{n})}\right) \Big/ \sum_{\mathrm{n}'=1}^{\mathrm{N}} \mathrm{p}_{\mathrm{m}-1}^{(\mathrm{n}')} \exp\left(-\alpha'_{\mathrm{m}} \mathrm{t}^{(\mathrm{n}')} \mathrm{o}_{\mathrm{m}}^{(\mathrm{n}')}\right) \tag{A.11}$$

se tiene el algoritmo “Real AdaBoost” como se presenta en la página siguiente. La condición de parada habitual es por saturación de prestaciones.

Real AdaBoost

1. Inicializar $\{p_0^{(n)}\}= 1/N$

2. Para $m = 1, \ldots, M$ (condición de parada)

   2.1 o_m: minimización ponderada por $\left\{p_{m-1}^{(n)}\right\}$ de un coste

   2.2 a) $\gamma_m = \sum_n p_{m-1}^{(n)} t^{(n)} o_m^{(n)}$

   b) $\alpha_m = \frac{1}{2} \ln \frac{1+\gamma_m}{1-\gamma_m}$

   2.3 $p_m^{(n)} = p_{m-1}^{(n)} \exp\left(-\alpha_m t^{(n)} o_m^{(n)}\right) \Big/ \sum_{n'=1}^{N} p_{m-1}^{(n')} \exp\left(-\alpha_m t_m^{(n')} o_m^{(n')}\right)$

3. $o(\mathbf{x}) = \widehat{t}(\mathbf{x}) = \text{sgn}\left[f_M(\mathbf{x})\right] = \text{sgn}\left[\sum_{m=1}^{M} \alpha_m o_m(\mathbf{x})\right]$

Apéndice B

Énfasis generalizado

Expondremos el énfasis generalizado que empleamos en la Tesis y que se toma de [Alvear-Sandoval and Figueiras-Vidal, 2018], [Alvear-Sandoval et al., 2017] y [Alvear-Sandoval and Figueiras-Vidal, 2016], para el contexto de pre-énfasis a partir de los resultados de una máquina auxiliar o guía, que se indicará mediante el subíndice a. Su exportación al "Real AdaBoost" (binario) requiere únicamente considerar como guía la agregación de aprendices hasta el que se va a entrenar, y la supresión de la expresión que calcula secuencialmente las ponderaciones.

Versión binaria

Se ha argumentado en el Capítulo 1 que la elección de las medidas de error y de proximidad en la frontera tiene importancia secundaria, con efectos dependientes del problema. De modo que, por sencillez, se selecciona como peso para la muestra $\mathbf{x}^{(\mathrm{n})}$ (que es el que pondera la correspondiente componente del coste a minimizar en el subsiguiente entrenamiento)

$$\mathrm{p}(\mathbf{x}^{(\mathrm{n})}) = \alpha + (1-\alpha)\left[\beta\left(t^{(\mathrm{n})} - \mathrm{o}_{\mathrm{a}}^{(\mathrm{n})}\right)^2 + (1-\beta)\left(1 - \mathrm{o}_{\mathrm{a}}^{(\mathrm{n})\,2}\right)\right] \tag{B.1}$$

con evidentes significados de las variables, y siendo α, β, $0 \leq \alpha, \beta \leq 1$, los coefi-

cientes de combinación convexa.

Resulta claro que α es la componente constante que sirve para moderar el énfasis que se aplica, $(1-\alpha)\beta\left(\mathrm{t}^{(\mathrm{n})}-\mathrm{o}_{\mathrm{a}}^{(\mathrm{n})}\right)^2$ la componente proporcional al error (cuadrático), y $(1-\alpha)(1-\beta)\left(1-\mathrm{o}_{\mathrm{a}}^{(\mathrm{n})\,2}\right)$ la que depende de la proximidad a la frontera. Así que α es el parámetro que determina el nivel del énfasis –menor a mayor α– y β el que establece el reparto de atención entre el error –mayor a mayor β– y la proximidad a la frontera.

La flexibilidad de (B.1) se reduce fijando valores para α o/y β; en particular, se tienen

- para $\alpha = 1$: diseño no enfatizado
- para $\alpha = 0$, β por validación: énfasis directo completo
- para $\alpha = 0$, $\beta = 1$: énfasis directo por error
- para $\alpha = 0$, $\beta = 0$: énfasis directo por proximidad
- para α por validación, $\beta = 1$: énfasis por error, moderado
- para α por validación, $\beta = 0$: énfasis por proximidad, moderado

Cuando α y β resulten distintos de 0 y 1, la generalidad de la forma (B.1) supone ventaja en el proceso de diseño. Determinar la importancia de esa ventaja es una cuestión cuantitativa, que requerirá comparar las prestaciones con los valores obtenidos por validación con las correspondientes formas restringidas que se acaban de listar. Debe desde ahora resaltarse que, al ser todas esas formas casos particulares de la general, ha de esperarse que lo más desfavorable será que la forma general adopte valores de los parámetros iguales a los de alguna forma reducida: si esto no sucede, ha de atribuirse a un defectuoso procedimiento de validación (que convendrá mejorar).

Versión multiclase

Suponiendo que, como es habitual, la guía incluya una activación de Potts (o análoga), que proporciona salidas no negativas y de suma unitaria, se recurrirá al énfasis

$$\mathrm{p}(\mathbf{x}^{(\mathrm{n})}) = \alpha + (1-\alpha)\left\{\beta\left(1-\mathrm{o}_{\mathrm{ac}}^{(\mathrm{n})}\right)^2 + (1-\beta)\left[1-|\mathrm{o}_{\mathrm{ac}}^{(\mathrm{n})} - \mathrm{o}_{\mathrm{cc}'}^{(\mathrm{n})}|\right]\right\} \tag{B.2}$$

donde α, β, $0 \leq \alpha, \beta \leq 1$, juegan el mismo papel que en el caso binario y $\mathrm{o}_{\mathrm{ac}}^{(\mathrm{n})}$ es la salida de la guía correspondiente a la clase c indicada por la etiqueta,

- con lo que $\left(1-\mathrm{o}_{\mathrm{ac}}^{(\mathrm{n})}\right)^2$ tiene sentido de término de error cuadrático–, y $\mathrm{o}_{\mathrm{ac}'}^{(\mathrm{n})}$ es la salida distinta de la anterior que se halla más próxima a $\mathrm{o}_{\mathrm{ac}}^{(\mathrm{n})}$,

- de modo que el factor de $1-\beta$ es una medida de la proximidad a la frontera más cercana; emplear un valor absoluto en lugar de elevar al cuadrado es simplemente por comodidad: sus efectos serán, como se ha alegado, menores.

Es cierto que puede pensarse en factores de $1-\beta$ con aspecto distinto al del incluido en (B.2) –por ejemplo, incluyendo $\max\limits_{\mathrm{c}'}\{\mathrm{o}_{\mathrm{ac}'}^{(\mathrm{n})}\}$–, pero el sentido de $1-|\mathrm{o}_{\mathrm{ac}}^{(\mathrm{n})} - \mathrm{o}_{\mathrm{cc}'}^{(\mathrm{n})}|$ es claro, y su empleo conduce a buenos resultados.

La discusión de las formas restringidas es paralela a la de la versión binaria.

Apéndice C

Descripción de las bases de datos

En este apéndice se comentan las características de las bases de datos que se han utilizado para evaluar las prestaciones de los diferentes algoritmos de clasificación propuestos en esta Tesis Doctoral. Los conjuntos de datos han sido seleccionados de modo que provean suficiente diversidad en cuanto a número de muestras, atributos y características y complejidad, razón por la cual se consideran tanto problemas reales como sintéticos.

Se han elegido doce base de datos, de las cuales trece pertenecen al repositorio de la Universidad de California en Irvine (UCI) [Frank and Asuncion, 2010], dos son problemas sintéticos [Kwok, 1999] y [Ripley, 1994], y una última, perteneciente a [Ripley, 1996]. A continuación, se describen en más detalle estos problemas de clasificación:

- *Abalone*: este problema consiste en estimar la edad de un tipo de caracol marino ("abalone"[1]) a partir de ocho atributos, como son su sexo y medidas físicas sobre su concha (longitud, diámetro, altura, peso, etc.). Los datos son reales y fueron obtenidos por los laboratorios de Investigación de la Marina de Tasma-

[1]En Castellano, "abulón", caracol marino de California de concha grande, gruesa, auriculada y muy nacarada.

nia, en Australia.

El problema se ha transformado en un problema de clasificación binaria, tal y como se indica [Ruiz and Lopez-de Teruel, 2001], para determinar si la edad del molusco es superior o inferior a 10 años.

- *Breast*: "Winconsin Breast Cancer" es un problema que consiste en determinar el carácter benigno o maligno de un tumor a partir de un conjunto de atributos medidos en los núcleos de las células cancerígenas. Las tasas de error mínimas registradas empíricamente están alrededor de 2.1 %, según se registran en [Frank and Asuncion, 2010].

- *Diabetes*: la base de datos "Pima Indian Diabetes", cuenta con información real de mujeres indígenas proveniente de la tribu Pima de Arizona, de 21 años o más de edad. A los indios Pima se les conoce por estar genéticamente predispuestos a padecer diabetes. La base de datos está compuesta por 768 muestras, con un conjunto de 8 variables o atributos (número de embarazos, concentración plasmática de glucosa a las dos horas de una prueba de tolerancia a la glucosa oral, presión diastólica de la sangre, espesor de la piel del tríceps, cantidad de insulina en dos horas, índice de masa corporal, antecedentes familiares de padecer diabetes y, la edad del paciente). La etiqueta es el diagnóstico de la condición del individuo (de acuerdo con los criterios de la Organización Mundial de la Salud).

- *Credit*: esta base de datos binaria contiene información de solicitudes de tarjetas de crédito. Todos los nombres de los atributos han sido cambiados por símbolos sin significado para proteger la confidencialidad de los datos. La base de datos contiene 690 muestras y cada muestra está caracterizada por 14 atributos. La clase indica si se ha aceptado la solicitud o no.

  Esta base de datos, que se ha utilizado en [Quinlan, 1987] y [Quinlan, 1993], resulta interesante desde el punto de vista de que se trata de una buena mez-

cla de atributos –continuos y nominales (con números de pequeños valores y grandes valores).

- *Firm-Teacher Clave-Direction*: esta base de datos de 4 clases contiene un total de 10800 muestras, en la que cada muestra se caracteriza por 16 valores binarios y la salida es un número de 4 bits[2]. Los 16 bits de entrada representan vectores de punto de ataque con valores binarios. 1 indica la presencia sustancial (0, ausencia) de un comienzo o empiece (nota de inicio) en una determinada ventana de tiempo durante un compás de 4/4 (no limitada a percusión, por tanto, son vectores de inicio sin duración), donde una nota dura una dieciseisava parte del tiempo. Cada vector tiene 16 posiciones en los que puede existir o no un comienzo. Las cuatro clases de salida son (de izquierda a derecha): neutral, clave inversa, clave delantera e incoherente.

- *German*: base de datos pública de créditos alemanes. Contiene un total de 1000 muestras, en la que cada muestra está formada por 20 variables (7 numéricas y 13 categóricas). El objetivo de este problema es categorizar a una persona que tiene una tarjeta de crédito, en dos categorías, según exista o no el riesgo de impago.

- *Hepatitis*: base de datos cuyo objetivo es predecir la presencia o ausencia de hepatitis en pacientes que han sido sometidos a diferentes análisis clínicos. Contiene un total de 155 muestras, en la que cada muestra está formada por 19 atributos, y la clase es decidir si el enfermo está vivo o muerto (el número de pacientes que han fallecido es 32).

- *Ionosfera*: base de datos que contiene ejemplos de clasificación de retornos a los radares desde la ionosfera. El objetivo es determinar si la señal radar que se

[2]Codificación "one four-bit one hot encoding": hay tantos número de bits como número de clases, y para cada clase, uno y sólo un bit se selecciona. A modo de ejemplo, en un problema de 3 clases, la clase 1 se codificaría como 001, clase 2 como 010 y, finalmente, la clase 3 como 100.

refleja en la ionosfera indica o no la presencia allí de algún tipo de estructura. Los retornos de tipo "Good" son los que muestran evidencias de algún tipo de estructura en la ionosfera. Este problema contiene 351 muestras y cada muestra está caracterizada por 34 atributos.

Las tasas de error mínimas registradas empíricamente están alrededor de 1.8 %, según se indica en [Frank and Asuncion, 2010].

- *Image*: esta base de datos representa un problema de clasificación binaria basado en la segmentación de imágenes. Las muestras se eligieron de forma aleatoria a partir de una base de datos de siete imágenes tomadas al aire libre. El número de muestras es 2310, donde cada muestra queda caracterizada por 18 atributos.

- *Biología molecular ("splice-junction gene sequences")*: las zonas de empalme o unión (en inglés, "splice-junction") son puntos en una secuencia de ADN en los que el ADN superfluo se elimina durante el proceso de creación de proteínas en grandes organismos. El problema planteado en este conjunto de datos es reconocer, dada una secuencia de ADN, los límites entre los exones[3] y los intrones[4]. Concretamente, este problema de clasificación consiste en reconocer las siguientes 3 clases: límite exión-intrón (también conocidos como EI), límite intrón-exión (conocidos como IE) y ninguno de los anteriores (esta última clase representa la mitad de las muestras de la base de datos).

  La base de datos contiene un total de 3190 secuencias de ADN (muestras), donde cada secuencia queda caracterizada por un total de 60 valores, empezando en -30 y finalizando en $+30$, y cada atributo toma uno de los siguientes valores: {a, g, t, c}.

- *Satimage*: esta base de datos de 6 clases contiene un total de 6635 muestras

[3]Un exón es la región de un gen que no se separa durante el proceso de corte y empalme, y por tanto, se mantiene en el ARN mensaje maduro.

[4]Un intrón es una región del ADN que forma parte de la transcripción primaria de ARN, pero a diferencia de los exones, son eliminados del transcrito maduro, previamente a su traducción.

obtenidas de imágenes por satélite (divididas en 4435 para entrenamiento y 2000 para test). Cada muestra, es una imagen diferente, y se caracteriza por 36 atributos (4 bandas espectrales $\times$ 9 píxeles) más el nombre de la clase. Los atributos son numéricos, en el rango de 0 a 255 (8 bits). El nombre de la clase tiene un código, como se muestra en la Tabla C.1.

Clase numérica	Descripción de la clase	Número de muestras	
		Entrenamiento	Test
1	Tierra roja	1072	461
2	Cosecha de algodón	479	224
3	Tierra gris	961	397
4	Tierra gris húmeda	415	211
5	Tierra con vegetación	470	237
6*	Mezcla de clases	–	–
7	Cada terreno gris húmedo	1038	470

*: No hay muestras de la clase 6 en la base de datos

Tabla C.1: Conjunto de clases de la base de datos *Satimage*.

- *Vehicle*: el objetivo de esta base de datos es clasificar una silueta dada como uno de los 4 tipos de vehículos considerados en el problema (un autobús de dos plantas, furgoneta Chevrolet, Saab 900 y Opel Manta 400). Las imágenes se tomaron con una cámara apuntando hacia abajo con un ángulo de elevación fijo (34,2° sobre la horizontal). Los vehículos se situaron sobre un fondo claro y se pintaron de negro mate para minimizar reflejos. Todas las imágenes tenían la misma resolución (128×128), fueron tomadas con la misma cámara, en escala de 64 tonos de gris. Las imágenes fueron filtradas para producir siluetas binarias de los vehículos, hechas negativo, limpiadas y almacenadas. Se realizaron más fotos rotando los vehículos, almacenándose al final un total de 2 juegos de 60

fotos por vehículo, cubriendo los 360° de rotación posibles. Después se realizaron dos tandas más de 2 juegos de 60 fotos, salvo furgoneta que tuvo 2 juegos de 46, a dos elevaciones diferentes (37,5° y 30,8°).

En total esta base de datos contiene un total de 846 siluetas (muestra) y cada muestra se caracteriza por 18 atributos.

- *Waveform*: esta base de datos contiene datos de tres ondas distintas, en el que cada clase es generada a partir de la combinación de dos de las tres ondas "base" triangulares. Contiene un total de 5000 muestras y cada muestra está formada por 40 atributos. Es importante resaltar que a cada atributo o característica se le añade ruido gaussiano de media cero y varianza unidad [Breiman et al., 1984].

  Para transformar el problema de clasificación de tres clases en binario, se han agrupado dos clases en una con el objetivo de distinguir si el dato procede o no de la combinación de dos de las ondas "base".

Las bases de datos restantes, que proceden de otras fuentes, son:

- *Crabs*: esta base de datos describe cinco mediciones morfológicas de doscientos cangrejos de ambos sexos y dos colores (cincuenta cangrejos de cada sexo y color)[5], pertenecientes a la variedad Leptograpsus Variegatus. El objetivo es clasificar los cangrejos según su sexo. Esta base de datos se ha utilizado en [Ripley, 1996].

- Kwok: base de datos sintética creada por [Kwok, 1999] para problemas de clasificación binaria. Los datos corresponden a cinco gaussianas bidimensionales esféricas: dos gaussianas para generar las muestras de una clase y tres para la clase contraria. El conjunto de entrenamiento está formado por 500 patrones, mientras que el de test contiene 10200 muestras.

  La tasa de error de Bayes sobre el conjunto de test es del 11.3 %.

[5]Las mediciones se han realizado en Fremantle (ciudad del oeste de Australia)

- *Ripley*: es una base de datos binaria propuesta en [Ripley, 1994]. Los datos de cada clase proceden de distribuciones formadas por la mezcla de dos gaussianas de igual matriz de covarianza y probabilidad a priori.

  La tasa de error de Bayes sobre el conjunto de test es del 8 %.

Apéndice D

Test de diferencias estadísticas

En estadística, un resultado o efecto es "estadísticamente significativo" cuando es improbable que haya sido debido al azar[1]. El nivel de significación de una prueba estadística es un concepto estadístico asociado a la verificación de una hipótesis. En pocas palabras, se define como la probabilidad de tomar la decisión de rechazar la hipótesis nula cuando ésta es verdadera. La decisión se toma a menudo utilizando el valor p (o p-valor): si el valor p es inferior al nivel de significación, entonces la hipótesis nula es rechazada. Cuanto menor sea el valor p, más significativo será el resultado. En otros términos, el nivel de significación de un contraste de hipótesis es una probabilidad p tal que la probabilidad de tomar la decisión de rechazar la hipótesis nula –cuando ésta es verdadera– no es mayor que p.

En esta Tesis Doctoral, para poder comparar los errores de clasificación entre dos algoritmos resulta interesante conocer en qué casos sus poblaciones de muestras son estadísticamente diferentes, para lo que se ha aplicado el T-test; y calculado el p-valor como indicador de diferencia estadística entre los resultados [Hill and Lewicki, 2006], tal y como se explicará en los siguientes apartados. Asimismo, en algunos

[1]Una "diferencia estadísticamente significativa" solamente quiere decir que hay evidencias estadísticas de que hay una diferencia; no significa que la diferencia sea grande, importante o radicalmente diferente.

experimentos, se ha aplicado un test no paramétrico alternativo al T-test conocido como test de la suma de rangos de Wilcoxon, equivalente al test U de Mann-Whitney o simplemente U-test.

En rigor, el uso de los test de diferencias estadísticas antes mencionados requiere que las realizaciones sean independientes. En los casos en los que se va aplicar aquí, se emplean los mismos datos de entrenamiento, y aunque se inicializan de forma distinta los pesos de los MLPs e, incluso, las particiones de los datos de entrenamiento son aleatorias, no se puede garantizar esta independencia. Por tanto, los resultados que se obtienen son sólo indicativos.

T-test

Este test proporciona un valor, t, que resulta de calcular la diferencia entre las medias de las dos distribuciones que se comparan normalizadas por sus desviaciones estándares; dado que se desconoce el tipo de distribución que siguen los grupos de datos a comprar, el T-test considera que sus medias siguen una distribución gaussiana de media $\{\mathrm{m_i}\}_{\mathrm{i}=1,2}$ (dada por la media muestral) y desviación estándar $\{\sigma_\mathrm{i}/\sqrt{\mathrm{N_i}}\}_{\mathrm{i}=1,2}$, donde las desviaciones estándares $\{\sigma_\mathrm{i}\}_{\mathrm{i}=1,2}$ siguen una distribución Chi-cuadrado y los valores $\{\mathrm{N_i}\}_{\mathrm{i}=1,2}$ representan al número de valores en cada grupo de datos. De este modo, el valor de t viene dado por la siguiente expresión:

$$\mathrm{t} = \frac{\mathrm{m}_1 - \mathrm{m}_2}{\sqrt{\sigma_1^2/\mathrm{N}_1 + \sigma_2^2/\mathrm{N}_2}} \tag{D.1}$$

y, tal y como se puede demostrar, se distribuye según una t de Student con $\mathrm{N}_1+\mathrm{N}_2-1$ grados de libertad.

En el caso de los experimentos que se presentan en esta Tesis Doctoral, los valores m_1 y m_2 se corresponden con los valores de CE (clasificación de error) de los algoritmos que se comparan y, como cada uno de estos valores se ha promediado sobre 50 iteraciones, el número de grados de libertad de la distribución t de Student será 99 ($\mathrm{N}_1 = 50$ y $\mathrm{N}_2 = 50$).

Entonces, y de acuerdo con las tablas de la distribución t de Student, se puede afirmar con un 95 % de fiabilidad que los valores medios de las dos distribuciones son estadísticamente diferentes si $|t| > 1{,}66$.

Para extraer conclusiones con un nivel de certeza distinto del 95 %, podrían emplearse los valores límite correspondientes, algunos de los cuales se encuentran recogidos en la Tabla D.1.

Certeza (%)	75	80	85	90	**95**	97.5	99	99.5	99.9	99.95
t_{limit}	0.677	0.845	1.042	1.290	**1.660**	1.984	2.364	2.626	3.174	3.390

Tabla D.1: Valores límite del parámetro t del T-test para distintos niveles de certeza.

El p-valor

El p-valor o nivel de significación observado, es el área de la cola de una distribución normal estándar (o varias colas si el test es bilateral) definida a partir del estadístico de contraste. Este p-valor se interpreta como una medida de la evidencia estadística que las poblaciones aportan a favor de la hipótesis alternativa (existe diferencia significativa) o en contra de la hipótesis nula (no existe diferencia significativa). Cuando el p-valor es pequeño, se considera que hay una fuerte evidencia a favor de la hipótesis alternativa, es decir, la hipótesis nula es rechazada, siendo tal resultado denominado "estadísticamente significativo". Por tanto, cuanto menor sea el p-valor, más fuerte será la evidencia de que un hecho no se debe a una mera coincidencia (al azar). En la práctica, se suele adoptar el criterio de aceptar la hipótesis nula cuando el p-valor es mayor que el nivel de significación, que se prefija normalmente a 0.05 para un nivel de certeza del 95 %.

En esta Tesis Doctoral, el p-valor es el resultado de aplicar el T-test antes mencionado. Por tanto, cuando este valor sea inferior a 0.05, se podrá afirmar que los resultados son "estadísticamente significativos" con un nivel de certeza del 95 %.

Test de la suma de rangos de Wilcoxon

A veces, las hipótesis necesarias para poder aplicar un test parámetrico (normalización de las distribuciones) no se verifican y es estrictamente necesario realizar el test que se presenta en esta sección. Un caso muy claro de no normalidad es cuando los datos pertenecen a una escala ordinal.

El test de la suma de rangos de Wilcoxon, equivalente al test U de Mann-Whitney (también conocido como U-test) se utiliza cuando las muestras son variables ordinales o continuas (tienen un orden) pero no se puede asumir normalidad entre ellas. Requiere de homogeneidad de varianzas (homocedasticidad), y a diferencia del T-test, éste compara medianas y no medias como hace el T-test.

Supongamos que se desea comparar dos grupos de datos X_1 e X_2 de tamaño N_1 y N_2, respectivamente:

$$X_1 = \{x_1^1, x_2^1, \ldots, x_{N_1}^1\}$$
$$X_2 = \{x_1^2, x_2^2, \ldots, x_{N_2}^2\}$$

Un modo intuitivo de proceder consiste en ordenar en valor absoluto las observaciones obtenidas, independientemente de su distribución de origen, de menor a mayor valor y asignar rangos a los datos así ordenados. De esta forma, a la observación con un valor absoluto más pequeño se le asigna rango 1, a la siguiente rango 2, y así sucesivamente. En caso de empates (si dos o más observaciones coinciden en valor) se le asignará a cada una de estas observaciones el promedio de los rangos que les serían asignados si no hubiese empate.

Nótese que en esta Tesis Doctoral, los grupos de datos de X_1 y X_2 se corresponden con los valores de CE (clasificación de error) de los algoritmos que se desean comparar, mientras que N_1 y N_2 son el número de valores en cada grupo de datos.

Imaginemos, a modo de ejemplo, que se dispone de las dos siguientes distribuciones:

$$X_1 = \{1.1,\ 3.4,\ 4.3,\ 2.1,\ 7.0,\ 2.4\}$$

$$X_2 = \{7.0,\ 8.0,\ 3.0,\ 5.0,\ 6.2,\ 4.4\}$$

En este caso, el valor más pequeño es "1.1" y se le asigna rango "1", mientras que el más grande es "8.0", y por tanto, se le asigna rango "12". El valor "7.0" está repetido dos veces y en lugar de asignarles "10" y "11", se les asigna "10.5" a cada uno.

Consideraremos como estadístico de contraste para el test de la suma de rangos de Wilcoxon, la suma de los rangos en cada una de las distribuciones de datos:

$$t = R_1 = \sum_{j=1}^{N_1} R(x_j^1)$$

ó

$$t = R_2 = \sum_{j=1}^{N_2} R(x_j^2)$$

La distribución de probabilidad de los estadísticos anteriores ha sido tabulada para tamaños de grupos de datos pequeños ($N_1 \leq 15$ y $N_2 \leq 15$). Para tamaños muestrales mayores como es nuestro caso ($N_1 > 15$ y $N_2 > 15$), es adecuado utilizar la aproximación normal, obteniendo a partir de t la variable:

$$z = \frac{t - m_{\mathrm{t}}}{\sigma_{\mathrm{t}}}$$

donde m_{t} y σ_{t} son la media y desviación estándar, respectivamente, de t si la hipótesis nula es cierta, y vienen dadas por las siguientes expresiones:

$$m_{\mathrm{t}} = \frac{1}{2} N_1 (N_1 + N_2 + 1)$$

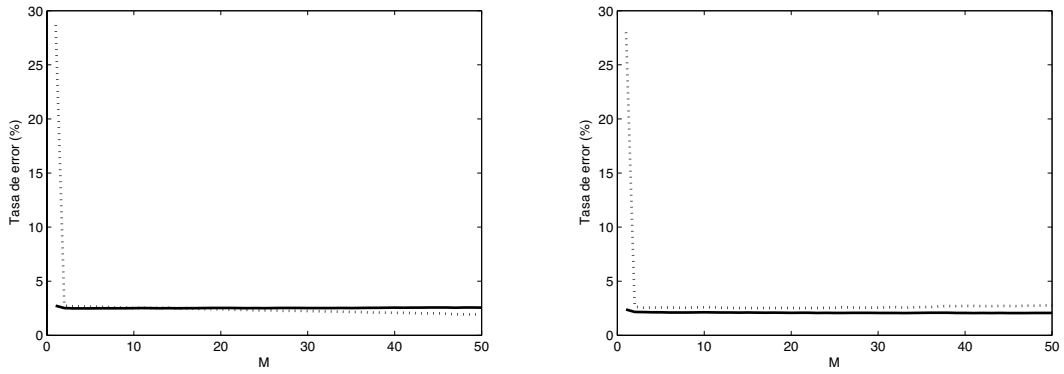
$$\sigma_{\mathrm{t}}^2 = \frac{N_1 N_2 (N_1 + N_2 + 1)}{12}$$

Una vez obtenido el valor de z, éste se debe referir a las tablas de la distribución normal para obtener el valor de significación asociado.

Apéndice E

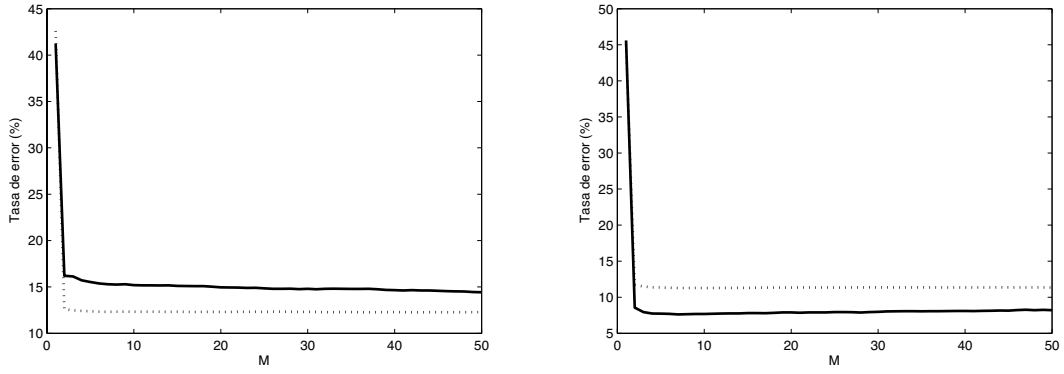
Gráficas comparativas entre el algoritmo CRA y ARA

Las Figuras E.1, E.2, E.3, E.4, E.5, E.6, E.7, E.8 y E.9 muestran, el error medio de clasificación del conjunto de entrenamiento y test (en %), para los algoritmos CRA (*Controlled Real AdaBoost*) y ARA (*Alternative Real AdaBoost*) en función del número de aprendices, para las bases de datos Bre, Cre, Ger, Hep, Ima, Ion, Kwo, Rip y Wav, respectivamente. Los valores se obtienen promediando sobre 50 repeticiones. Nótese que no se ha utilizado ningún mecanismo de parada.



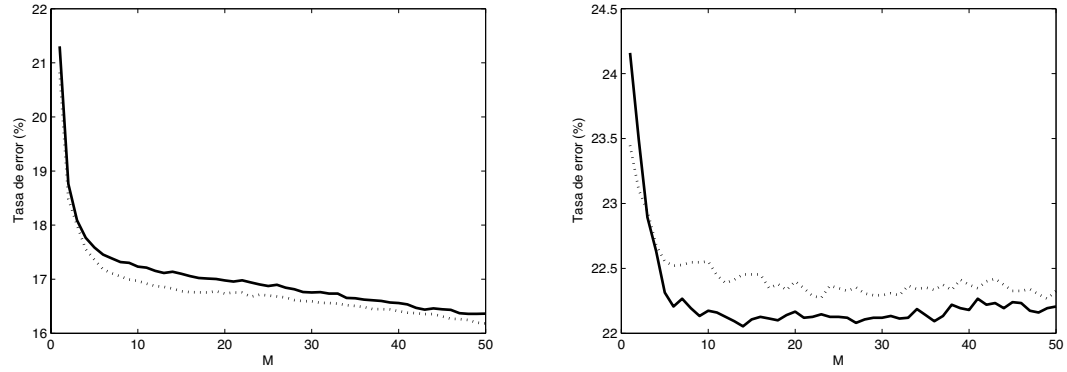
(a) (b)

Figura E.1: Evolución de la tasa de error media de entrenamiento (Figura (a)) y de test (Figura (b)), con el número de clasificadores base para el algoritmo CRA (α =0.1, β =0.1) (línea continua) y para el algoritmo ARA (línea discontinua) para el problema Bre, promediadas sobre 50 repeticiones.



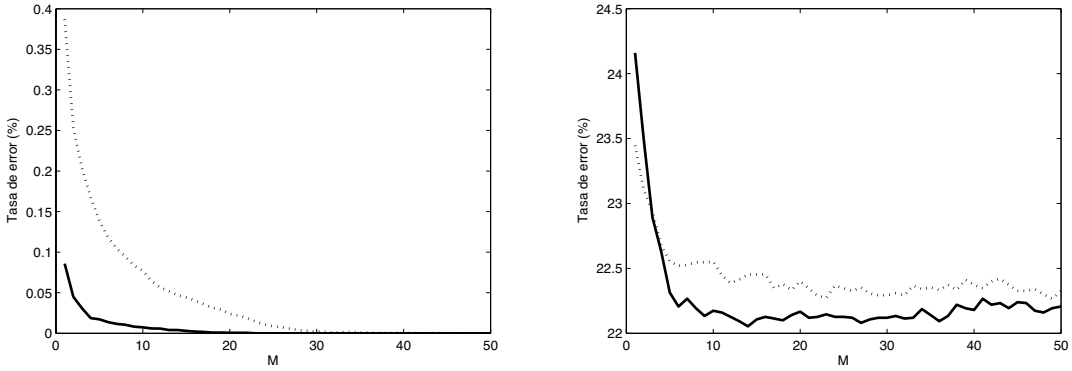
(a) (b)

Figura E.2: Evolución de la tasa de error media de entrenamiento (Figura (a)) y de test (Figura (b)), con el número de clasificadores base para el algoritmo CRA (α =0, β =0.9) (línea continua) y para el algoritmo ARA (línea discontinua) para el problema Cre, promediadas sobre 50 repeticiones.



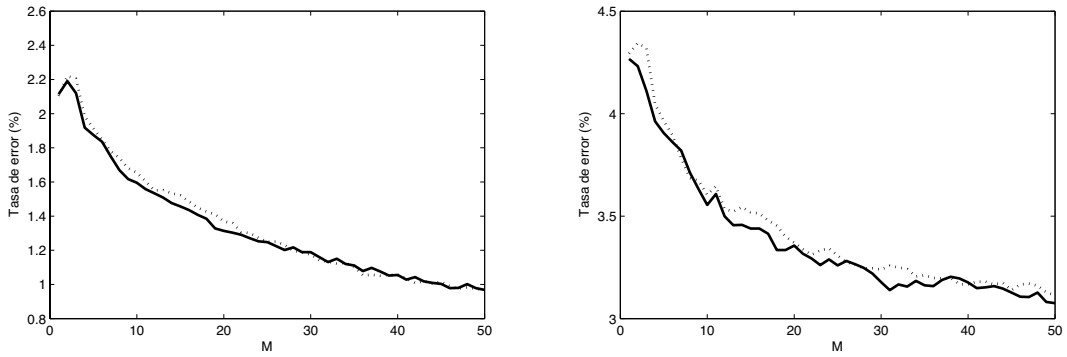
(a) (b)

Figura E.3: Evolución de la tasa de error media de entrenamiento (Figura (a)) y de test (Figura (b)), con el número de clasificadores base para el algoritmo CRA (α =0.8, β =0.2) (línea continua) y para el algoritmo ARA (línea discontinua) para el problema Ger, promediadas sobre 50 repeticiones.



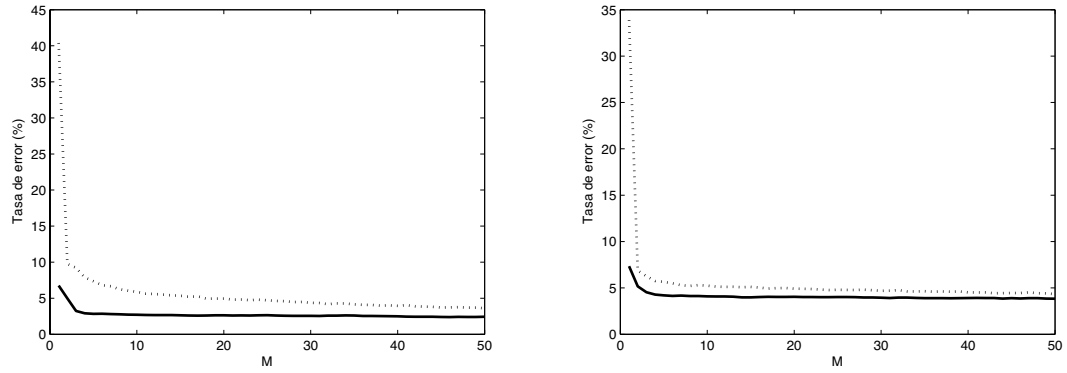
(a) (b)

Figura E.4: Evolución de la tasa de error media de entrenamiento (Figura (a)) y de test (Figura (b)), con el número de clasificadores base para el algoritmo CRA (α =0.9, β =0.6) (línea continua) y para el algoritmo ARA (línea discontinua) para el problema Hep, promediadas sobre 50 repeticiones.



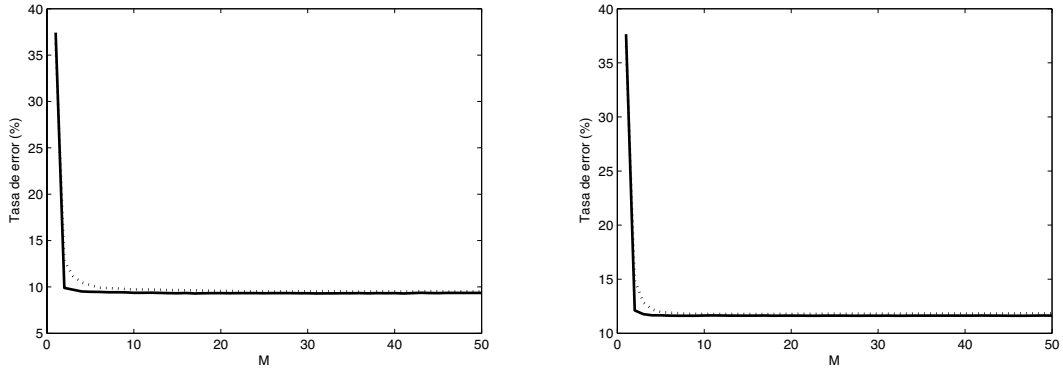
(a) (b)

Figura E.5: Evolución de la tasa de error media de entrenamiento (Figura (a)) y de test (Figura (b)), con el número de clasificadores base para el algoritmo CRA (α =0.2, β =0.2) (línea continua) y para el algoritmo ARA (línea discontinua) para el problema Ima, promediadas sobre 50 repeticiones.



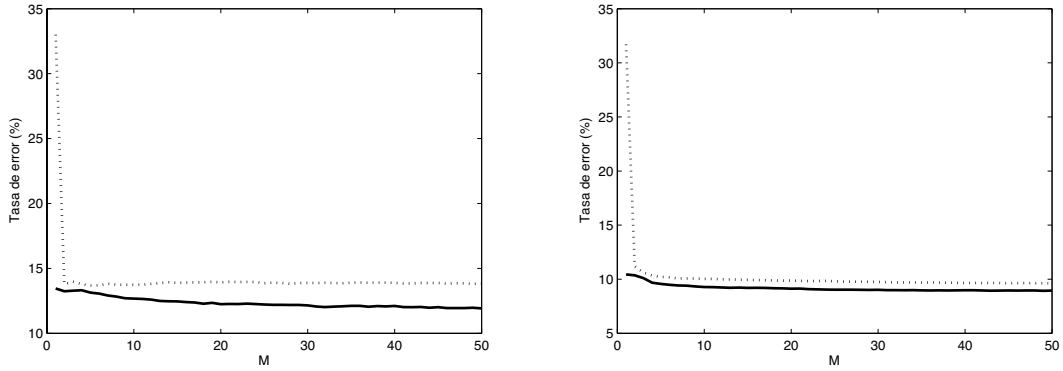
(a) (b)

Figura E.6: Evolución de la tasa de error media de entrenamiento (Figura (a)) y de test (Figura (b)), con el número de clasificadores base para el algoritmo CRA (α =0.6, β =0) (línea continua) y para el algoritmo ARA (línea discontinua) para el problema Ion, promediadas sobre 50 repeticiones.



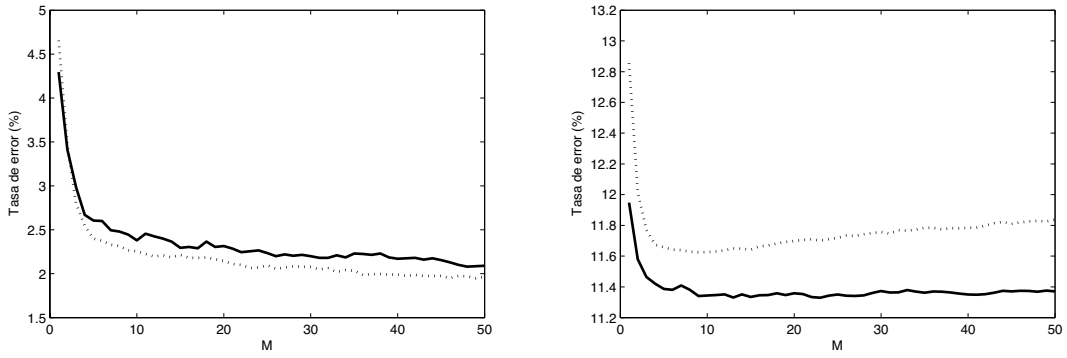
(a) (b)

Figura E.7: Evolución de la tasa de error media de entrenamiento (Figura (a)) y de test (Figura (b)), con el número de clasificadores base para el algoritmo CRA (α =0, β =0.5) (línea continua) y para el algoritmo ARA (línea discontinua) para el problema Kwo, promediadas sobre 50 repeticiones.



(a) (b)

Figura E.8: Evolución de la tasa de error media de entrenamiento (Figura (a)) y de test (Figura (b)), con el número de clasificadores base para el algoritmo CRA (α =0.3, β =0.1) (línea continua) y para el algoritmo ARA (línea discontinua) para el problema Rip, promediadas sobre 50 repeticiones.



(a) (b)

Figura E.9: Evolución de la tasa de error media de entrenamiento (Figura (a)) y de test (Figura (b)), con el número de clasificadores base para el algoritmo CRA (α =0.2, β =0.1) (línea continua) y para el algoritmo ARA (línea discontinua) para el problema Wav, promediadas sobre 50 repeticiones.

Bibliografía

[Ackley et al., 1985] Ackley, D. H., Hinton, G. E., and Sejnowski, T. J. (1985). A learning algorithm for Boltzmann machines. *Cognitive Science*, 9:147–169.

[Aizerman et al., 1964] Aizerman, M. A., Braverman, È. M., and Rozonoèr, L. I. (1964). The probability problem of pattern recognition learning and the method of potential functions. *Automation and Remote Control*, 25(9):1175–1190.

[Allwein et al., 2000] Allwein, E. L., Schapire, R. E., and Singer, Y. (2000). Reducing multiclass to binary: A unifying approach for margin classifiers. In *Proc. 17th Intl. Conf. Machine Learning*, pages 9–16, San Mateo, CA. Morgan Kaufmann.

[Alvear-Sandoval, 2017] Alvear-Sandoval, R. F. (2017). *Diversidad en Aprendizaje Profundo por Auto-codificación*. PhD thesis, UC3M.

[Alvear-Sandoval and Figueiras-Vidal, 2015] Alvear-Sandoval, R. F. and Figueiras-Vidal, A. R. (2015). Does diversity improve deep learning? In *Proc. European Conf. Signal Processing (EUSIPCO 2015)*, pages 2541–2545, Nice, France.

[Alvear-Sandoval and Figueiras-Vidal, 2016] Alvear-Sandoval, R. F. and Figueiras-Vidal, A. R. (2016). An experiment in pre-emphasizing diversified deep neural classifiers. In *Proceedings of European Symposium on Artificial Neural Networks*, volume 23, pages 527–532, Bruges, Bélgica.

[Alvear-Sandoval and Figueiras-Vidal, 2018] Alvear-Sandoval, R. F. and Figueiras-Vidal, A. R. (2018). On building ensembles of stacked denoising auto-encoding classifiers and their further improvement. *Information Fusion*, 39:41–52.

[Alvear-Sandoval et al., 2017] Alvear-Sandoval, R. F., Hayes, M. H., and Figueiras-Vidal, A. R. (2017). Improving denoising stacked auto-encoding classifiers by emphasizing training samples. Submitted to Neurocomputing.

[Arbib, 2002] Arbib, M. A., editor (2002). *The Handbook of Brain Theory and Neural Networks*. MIT Press, Cambridge, MA, USA, 2nd edition.

[Archer and Kimes, 2008] Archer, K. J. and Kimes, R. V. (2008). Empirical characterization of random forest variable importance measures. *Comp. Statistics and Data Analysis*, 52(4):2249–2260.

[Arya and Mount, 1993] Arya, S. and Mount, D. M. (1993). Approximate nearest neighbor queries in fixed dimensions. In *Proc. 4th Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 271–280, Philadelphia, PA, USA. Society for Industrial and Applied Mathematics.

[Atkeson et al., 1997] Atkeson, C. G., Moore, A. W., and Schaal, S. (1997). Locally weighted learning. *Artif. Intelligent Review.*, 11(1-5):11–73.

[Ballard, 1987] Ballard, D. H. (1987). Modular learning in neural networks. In *Proc. 6th Nat. Conf. Artificial Intelligence. Seattle, WA, July 1987.*, pages 279–284.

[Bauer and Kohavi, 1999] Bauer, E. and Kohavi, R. (1999). An empirical comparison of voting classification algorithms: Bagging, Boosting, and variants. *Machine Learning*, 36(1-2):105–139.

[Bengio, 2009] Bengio, Y. (2009). Learning deep architectures for AI. *Foundations and Trends in Machine Learning*, 2(1):1–127.

[Bengio et al., 2013] Bengio, Y., Courville, A., and Vincent, P. (2013). Representation learning: A review and new perspectives. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 35(8):1798–1828.

[Bishop, 1995] Bishop, C. M. (1995). *Neural Networks for Pattern Recognition*. Oxford Univ. Press, Oxford, UK.

[Bishop, 2006] Bishop, C. M. (2006). *Pattern Recognition and Machine Learning*. Springer, New York, NY.

[Boser et al., 1992] Boser, B. E., Guyon, I. M., and Vapnik, V. N. (1992). A training algorithm for optimal margin classifiers. In Haussler, D., editor, *Proc. 5th Annual Workshop on Computational Learning Theory*, pages 144–152, New York, NY. ACM Press.

[Breiman, 1996] Breiman, L. (1996). Bagging predictors. *Machine Learning*, 24(2):123–140.

[Breiman, 1998] Breiman, L. (1998). Arcing classifiers. *Annals of Statistics*, 26(3):801–823.

[Breiman, 1999a] Breiman, L. (1999a). Prediction games and arcing algorithms. *Neural Computation*, 11(7):1493–1517.

[Breiman, 1999b] Breiman, L. (1999b). Combining predictors. In Sharkey, A. J. C., editor, *Combining Artificial Neural Nets. Ensemble and Modular Multi–Net Systems*, pages 31–50, London. Springer-Verlag.

[Breiman, 2000] Breiman, L. (2000). Randomizing outputs to increase prediction accuracy. *Machine Learning*, 40(3):229–242.

[Breiman, 2001] Breiman, L. (2001). Random forests. *Machine Learning*, 45(1):5–32.

[Breiman et al., 1984] Breiman, L., Friedman, J. H., Olshen, R. A., and Stone, C. J. (1984). *Classification and Regression Trees*. Wadsworth, Belmont, CA. Desde 1993 este libro ha sido publicado por Chapman & Hall, New York, NY.

[Cachin, 1994] Cachin, C. (1994). Pedagogical pattern selection strategies. *Neural Networks*, 7(1):175–181.

[Carreira-Perpiñán and Hinton, 2005] Carreira-Perpiñán, M. A. and Hinton, G. E. (2005). On contrastive divergence learning. In Cowell, R. G. and Ghahramani, Z., editors, *Proc. 10th Intl. Workshop on Artificial Intelligence and Statistics*, pages 59–66. Soc. for Artificial Intelligence and Statistics.

[Choi and Rockett, 2002] Choi, S. and Rockett, P. (2002). The training of neural classifiers with condensed datasets. *IEEE Trans. Systems, Man, and Cybernetics, Part B*, 32(2):202–206.

[Ciresan et al., 2012a] Ciresan, D. C., Meier, U., Masci, J., and Schmidhuber, J. r. (2012a). Multi-column deep neural network for traffic sign classification. *Neural Networks*, 32:333–338.

[Ciresan et al., 2012b] Ciresan, D. C., Meier, U., and Schmidhuber, J. (2012b). Multi-column deep neural networks for image classification. In *Proc. 25th IEEE Conf. on Computer Vision and Pattern Recognition*, pages 3642–3649, New York, NY. IEEE Press.

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

[Cressie, 1993] Cressie, N. (1993). *Statistics for Spatial Data*. J. Wiley & Sons, New York, NY.

[Cybenko, 1989] Cybenko, G. (1989). Approximation by superpositions of a sigmoidal function. *Math. Control, Signals, and Systems*, 2(4):303–314.

[Dempster et al., 1977] Dempster, A. P., Laird, N. M., and Rubin, D. B. (1977). Maximum Likelihood from incomplete data via the EM algorithm (with discussion). *J. Royal Statistical Soc., Series B*, 39:1–38.

[Deng and Yu, 2011] Deng, L. and Yu, D. (2011). Deep convex net: A scalable architecture for speech pattern classification. In *Proc. Interspeech 2011*, pages 2285–2288, Florence, Italy.

[Deng and Yu, 2013] Deng, L. and Yu, D. (2013). Deep learning: Methods and applications. *Foundations and Trends in Signal Processing*, 7(3–4):197–387.

[Deng et al., 2012] Deng, L., Yu, D., and Platt, J. (2012). Scalable stacking and learning for building deep architectures. In *Proc. Intl. Conf. Acoustics, Speech, and Signal Processing*, pages 2133–2136, New York, NY. IEEE Press.

[Dietterich, 1998] Dietterich, T. G. (1998). Approximated statistical tests for comparing supervised classification learning algorithms. *Neural Computation*, 10(7):1895–1924.

[Dietterich, 2000] Dietterich, T. G. (2000). An experimental comparison of three methods for constructing ensembles of decision trees: Bagging, boosting, and randomization. *Machine Learning*, 40(2):139–157.

[Dietterich and Bakiri, 1995] Dietterich, T. G. and Bakiri, G. (1995). Solving multiclass learning problems via error-correcting output codes. *J. Artificial Intelligence Res.*, 2(1):263–286.

[Drucker et al., 1994] Drucker, H., Cortes, C., Jackel, L. D., LeCun, Y., and Vapnik, V. (1994). Boosting and other ensemble methods. *Neural Computation*, 6(6):1289–1301.

[Drucker et al., 1993] Drucker, H., Schapire, R. E., and Simard, P. Y. (1993). Boosting performance in neural networks. *Intl. J. Pattern Recognition and Artificial Intelligence*, 7(5):705–719.

[Duda et al., 2001] Duda, R. O., Hart, P. E., and Stork, D. G. (2001). *Pattern Classification*. John Wiley & Sons, New York, NY, 2nd edition.

[El Jelali et al., 2008a] El Jelali, S., Lyhyaoui, A., and Figueiras-Vidal, A. b. R. (2008a). An emphasized target smoothing procedure to improve MLP classifiers performance. In *Proc. 16th European Symp. Artificial Neural Networks*, pages 499–504. Bruges (Belgium).

[El Jelali et al., 2008b] El Jelali, S., Lyhyaoui, A., and Figueiras-Vidal, A. b. R. (2008b). Applying emphasized soft targets for Gaussian mixture model based classification. In *Proc. of the Intl. Multiconf. Computer Sci. and Information Technology, 3rd. Intl. Symp. Advances in Artificial Intelligence and Applications*, volume 3, pages 131–136. Wisla (Poland).

[El Jelali et al., 2009] El Jelali, S., Lyhyaoui, A., and Figueiras-Vidal, A. R. (2009). Designing model based classifiers by emphasizing soft targets. *Fundamenta Informaticae*, 96(4):419–433.

[Erhan et al., 2010] Erhan, D., Bengio, Y., Courville, A., Manzagol, P.-A., Vincent, P., and Bengio, S. (2010). Why does unsupervised pre-training help deep learning? *J. Machine Learning Res.*, 11:625–660.

[Fahlman and Lebiere, 1990] Fahlman, S. E. and Lebiere, C. (1990). The cascade-correlation learning architecture. In Touretzky, D. S., editor, *Advances in Neural Information Processing Systems 2*, pages 524–532, San Mateo, CA. Morgan Kaufmann Publishers Inc.

[Fisher, 1936] Fisher, R. A. (1936). The use of multiple measurements in taxonomic problems. *Annals of Eugenics*, 7(2):179–188.

[Franco and Cannas, 2000] Franco, L. and Cannas, S. A. (2000). Generalization and selection of examples in feedforward neural networks. *Neural Computation*, 12(10):2405–2426.

[Frank and Asuncion, 2010] Frank, A. and Asuncion, A. (2010). UCI machine learning repository. University of California, Irvine, School of Information and Computer Sciences.

[Freund, 2001] Freund, Y. (2001). An adaptive version of the boost by majority algorithm. *Machine Learning*, 43(3):293–318.

[Freund and Schapire, 1995] Freund, Y. and Schapire, R. E. (1995). A decision-theoretic generalization of on-line learning and an application to Boosting. In *Proc. 2nd European Conf. Computational Learning Theory*, pages 23–37. Springer-Verlag Berlin Heidelberg.

[Freund and Schapire, 1996a] Freund, Y. and Schapire, R. E. (1996a). Experiments with a new Boosting algorithm. In Saitta, L., editor, *Proc. 13th Intl. Conf. Machine Learning*, pages 148–156, San Francisco, CA. Morgan Kaufmann Publishers Inc.

[Freund and Schapire, 1996b] Freund, Y. and Schapire, R. E. (1996b). Game theory, on-line prediction and Boosting. In *Proc. 9th Annual Conf. Computational Learning Theory*, pages 325–332. Desenzano di Garda, Italy.

[Freund and Schapire, 1997] Freund, Y. and Schapire, R. E. (1997). A decision-theoretic generalization of on-line learning and an application to Boosting. *J. Comput. and Syst. Sci.*, 55(1):119–139.

[Friedman et al., 2000] Friedman, J., Hastie, T., and Tibshirani, R. (2000). Additive logistic regression: A statistical view of Boosting. *Annals of Statistics*, 28(2):337–374.

[Friedman, 2001] Friedman, J. H. (2001). Greedy function approximation: A gradient boosting machine. *Annals of Statistics*, 29(5):1189–1232.

[Fukushima, 1979] Fukushima, K. (1979). Neural network model for a mechanism of pattern recognition unnaffected by shift in position – Neocognitron. *Trans. Inst. Electr. Comp. Eng. (Japan)*, J62-A(10):658–665.

[Fukushima, 1980] Fukushima, K. (1980). Neocognitron: A self-organizing neural network model for a mechanism of pattern recognition unaffected by shift in position. *Biological Cybernetics*, 36:193–202.

[Giannakopoulos et al., 1999] Giannakopoulos, X., Karhunen, J., and Oja, E. (1999). An experimental comparison of neural algorithms for independent component analysis and blind separation. *International Journal of Neural Systems*, 9:99–114.

[Gigerenzer, 2007] Gigerenzer, G. (2007). *Gut Feelings. The Intelligence of the Unconscious*. Viking, New York, NY.

[Glorot and Bengio, 2010] Glorot, X. and Bengio, Y. (2010). Understanding the difficulty of training deep feedforward neural networks. In *Proc. 30th Intl. Conf. Artificial Intelligence and Statistics, AISTATS*, volume 9, pages 249–256.

[Gómez-Verdejo, 2007] Gómez-Verdejo, V. (2007). *Nuevos Criterios de Ayuda para Conjuntos de Decisores Cooperativos*. PhD thesis, Universidad Carlos III de Madrid.

[Gómez-Verdejo et al., 2008] Gómez-Verdejo, V., Arenas-García, J., and Figueiras-Vidal, A. R. (2008). A dynamically adjusted mixed emphasis method for building boosting ensembles. *IEEE Trans. Neural Networks*, 19(1):3–17.

[Gómez-Verdejo et al., 2006] Gómez-Verdejo, V., Ortega-Moral, M., Arenas-García, J., and Figueiras-Vidal, A. R. (2006). Boosting by weighting critical and erroneous samples. *Neurocomputing*, 69(7-9):679–685.

[Hansen and Salamon, 1990] Hansen, L. K. and Salamon, P. (1990). Neural network ensembles. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 12(10):993–1001.

[Hart, 1968] Hart, P. E. (1968). The condensed nearest neighbor rule. *IEEE Trans. Information Theory*, 14(3):515–516.

[Hastie and Tibshirani, 1998] Hastie, T. and Tibshirani, R. (1998). Classification by pairwise coupling. *Annals of Statistics*, 26(2):451–471.

[Haykin, 1994] Haykin, S. (1994). *Neural Networks: A Comprehensive Foundation*. MacMillan Publishing Company, New York, NY.

[Haykin, 2007] Haykin, S. (2007). *Neural Networks: A Comprehensive Foundation (3rd ed.)*. Prentice-Hall, Inc., Upper Saddle River, NJ.

[Hebb, 1949] Hebb, D. O. (1949). *The Organization of Behavior*. Wiley, New York, NY.

[Hill and Lewicki, 2006] Hill, T. and Lewicki, P. (2006). *Statistics: Methods and applications: A comprehensive reference for science, industry, and data mining*. StatSoft; [United Kingdom] : [StatSoft Ltd.], Tulsa, Okla.

[Hinton et al., 2006] Hinton, G. E., Osindero, S., and Teh, Y.-W. (2006). A fast learning algorithm for deep belief nets. *Neural Computation*, 18(7):1527–1554.

[Hinton and Sejnowski, 1986] Hinton, G. E. and Sejnowski, T. J. (1986). Learning and relearning in Boltzmann machines. In Rumelhart, D. E., McClelland, J. L., and The PDP Research Group, editors, *Parallel Distributed Processing: Explorations in the Microstructure of Cognition, vol. 1: Foundations*, pages 282–317. MIT Press, Cambridge, MA.

[Holmström and Koistinen, 1992] Holmström, L. and Koistinen, P. (1992). Using additive noise in back-propagation training. *IEEE Trans. Neural Networks*, 3(1):24–38.

[Hornik et al., 1989] Hornik, K., Stinchcombe, M., and White, H. (1989). Multilayer feedforward networks are universal approximators. *Neural Networks*, 2(5):359–366.

[Hunt et al., 1966] Hunt, E. B., Marin, J., and Stone, P. J. (1966). *Experiments in Induction*. Academic Press, New York, NY.

[Ioffe and Szegedy, 2015] Ioffe, S. and Szegedy, C. (2015). Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *Proc. 32nd Intl. Conf. Machine Learning*, (JMLR W&CP 37), pages 448–456, Lille, France.

[Ivakhnenko, 1968] Ivakhnenko, A. G. (1968). The Group Method of Data Handling – A rival of the method of stochastic approximation. *Soviet Automatic Control*, 13(3):43–55.

[Ivakhnenko, 1971] Ivakhnenko, A. G. (1971). Polynomial theory of complex systems. *IEEE Trans. Systems, Man, and Cybernetics*, 1(4):364–378.

[Jacobs et al., 1991] Jacobs, R. A., Jordan, M. I., Nowlan, S. J., and Hinton, G. E. (1991). Adaptive mixtures of local experts. *Neural Computation*, 3(1):79–87.

[Jordan and Jacobs, 1994] Jordan, M. I. and Jacobs, R. A. (1994). Hierarchical Mixtures of Experts and the EM algorithm. *Neural Computation*, 6(2):181–214.

[Jordan and Xu, 1995] Jordan, M. I. and Xu, L. (1995). Convergence results for the EM approach to Mixtures of Experts architectures. *Neural Networks*, 8(9):1409 – 1431.

[Kahneman, 2011] Kahneman, D. (2011). *Thinking, Fast and Slow.* New York : Farrar, Straus and Giroux, New York, NY.

[Kimeldorf and Wahba, 1971] Kimeldorf, G. S. and Wahba, G. (1971). Some results on Tchebycheffian spline functions. *J. Math. Anal. Applications*, 33(1):82–95.

[Kohonen, 1982] Kohonen, T. (1982). Self-organized formation of topologically correct feature maps. *Biological Cybernetics*, 43(1):59–69.

[Kohonen, 1995] Kohonen, T., editor (1995). *Self-Organizing Maps.* Springer-Verlag New York, Inc., Secaucus, NJ, USA, 1st edition.

[Kuncheva, 2004] Kuncheva, L. I. (2004). *Combining Pattern Classifiers: Methods and Algorithms.* Wiley, Hoboken, NJ.

[Kwok, 1999] Kwok, J. T.-Y. (1999). Moderating the outputs of support vector machine classifiers. *IEEE Trans. Neural Networks*, 10(5):1018–1031.

[LeCun, 1985] LeCun, Y. (1985). Une procédure d'apprentissage pour réseau à seuil asymétrique. In *Proc. Cognitiva'85*, pages 599–604, Paris, France.

[LeCun et al., 1989] LeCun, Y., Boser, B., Denker, J. S., Henderson, D., Howard, R. E., Hubbard, W., and Jackel, L. D. (1989). Backpropagation applied to handwritten zip code recognition. *Neural Computation*, 1(4):541–551.

[LeCun et al., 1990] LeCun, Y., Boser, B., Denker, J. S., Henderson, D., Howard, R. E., Hubbard, W., and Jackel, L. D. (1990). Handwritten digit recognition with a back-propagation network. In Touretzky, D. S., editor, *Advances in Neural Information Processing Systems 2*, pages 396–404, San Mateo, CA. Morgan Kaufmann Publishers Inc.

[LeCun et al., 1998] LeCun, Y., Bottou, L., Bengio, Y., and Haffner, P. (1998). Gradient-based learning applied to document recognition. *Proc. IEEE*, 86(11):2278–2324.

[LeCun et al., 1995] LeCun, Y., Jackel, L. D., Bottou, L., Cortes, C., Denker, J. S., Drucker, H., Guyon, I., Müller, V. A., Säckinger, E., Simard, P., et al. (1995). Learning algorithms for classification: A comparison on handwritten digit recognition. *Neural networks*, 261:261–276.

[Leisch and Hornik, 1997] Leisch, F. and Hornik, K. (1997). Combining neural network voting classifiers and error correcting output codes. In Frollo, I. and Placková, A., editors, *Proc. MEASUREMENT 97*, pages 266–269.

[Liu and Yao, 1999a] Liu, Y. and Yao, X. (1999a). Ensemble learning via negative correlation. *Neural Networks*, 12(10):1399–1404.

[Liu and Yao, 1999b] Liu, Y. and Yao, X. (1999b). Simultaneous training of negatively correlated neural networks in an ensemble. *IEEE Trans. Systems, Man, and Cybernetics, Part B–Cybernetics*, 29(6):716–725.

[M ller et al., 2001] M ller, K.-R., Mika, S., R tsch, G., Tsuda, K., and Sch lkopf, B. (2001). An introduction to kernel-based learning algorithms. *IEEE Trans. Neural Networks*, 12(2):181–202.

[Martens, 2010] Martens, J. (2010). Deep learning via Hessian-free optimization. In *Proc. 27th Intl. Conf. Machine Learning*, pages 735–742, Haifa, Israel.

[Martínez-Ramón et al., 2006] Martínez-Ramón, M., Rojo-Álvarez, J. L., Camps-Valls, G., Muñoz-Marí, J., Navia-Vázquez, Á., Soria-Olivas, E., and Figueiras-Vidal, A. R. (2006). Support Vector Machines for nonlinear kernel ARMA system identification. *IEEE Trans. Neural Networks*, 17(6):1617–1622.

[Mason et al., 2000a] Mason, L., Bartlett, P. L., and Baxter, J. (2000a). Improved generalization through explicit optimization of margins. *Machine Learning*, 38(3):243–255.

[Mason et al., 2000b] Mason, L., Baxter, J., Barlett, P., and Frean, M. (2000b). Functional gradient techniques for combining hypotheses. In Smola, A. J., Barlett, P., Schölkopf, B., and Schuurmans, D., editors, *Advances in Large Margin Classifiers*. MIT Press, Cambridge, MA.

[Mayhua-López et al., 2012] Mayhua-López, E., Gómez-Verdejo, V., and Figueiras-Vidal, A. R. (2012). Real Adaboost with gate controlled fusion. *IEEE Trans. Neural Networks and Learning Systems*, 23(12):2003–2009.

[Mayhua-López et al., 2014] Mayhua-López, E., Gómez-Verdejo, V., and Figueiras-Vidal, A. R. (2014). Boosting ensembles with subsampled LPSVM learners. *Information Fusion*, 25:67–71.

[Memkite, 2014] Memkite (2014). *Deep Learning University: An Annotated Deep Learning Bibliography*. `http://memkite.com/deep-learning-bibliography/`.

[Michalski, 1980] Michalski, R. S. (1980). Pattern recognition as rule-guided inductive inference. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 2:349–361.

[Minsky and Papert, 1969] Minsky, M. and Papert, S. (1969). *Perceptrons*. MIT Press, Cambridge, MA.

[Montufar and Ay, 2011] Montufar, G. and Ay, N. (2011). Refinements of universal approximation results for Deep Belief Networks and Restricted Boltzmann Machines. *Neural Computation*, 23(5):1306–1319.

[Mora-Jiménez and Figueiras-Vidal, 2009] Mora-Jiménez, I. and Figueiras-Vidal, A. R. (2009). Improving performance of neural classifiers via selective reduction of target levels. *Neurocomputing*, 72(13-15):3020–3027.

[Munro, 1992] Munro, P. W. (1992). Repeat until bored: A pattern selection strategy. In Moody, J. E., Hanson, S. J., and Lippmann, R., editors, *Advances in Neural Information Processing Systems 4*, pages 1001–1008, San Mateo, CA. Morgan Kaufmann.

[Murphy, 2012] Murphy, K. P. (2012). *Machine Learning: A Probabilistic Perspective*. MIT Press, Cambridge, MA.

[Nilsson, 1965] Nilsson, N. J. (1965). *Learning machines: Foundations of Trainable Pattern Classifying Systems*. McGraw-Hill.

[Noordewier et al., 1991] Noordewier, M. O., Towell, G. G., and Shavlik, J. W. (1991). Training knowledge-based neural networks to recognize genes in DNA sequences. *Advances in Neural Information Processing Systems*, 3:530–536.

[Omari and Figueiras-Vidal, 2013] Omari, A. and Figueiras-Vidal, A. R. (2013). Feature combiners with gate-generated weights for classification. *IEEE Trans. Neural Networks and Learning Systems*, 24(1):158–163.

[Omari and Figueiras-Vidal, 2015] Omari, A. and Figueiras-Vidal, A. R. (2015). Post-aggregation of classifier ensembles. *Information Fusion*, 26:96–102.

[Parker, 1982] Parker, D. B. (1982). Learning Logic. Invention Report S81-64, File 1, Office of Technology Licensing, Stanford Univ., Standford, CA.

[Parzen, 1962] Parzen, E. (1962). On estimation of a probability density function and mode. *Annals of Mathematical Statistics*, 33(3):1065–1076.

[Plutowski and White, 1993] Plutowski, M. and White, H. (1993). Selecting concise training sets from clean data. *IEEE Trans. Neural Networks*, 4(2):305–318.

[Quinlan, 1987] Quinlan, J. (1987). Simplifying decision trees. *Intl. J. Man-Machine Studies*, 27(3):221 – 234.

[Quinlan, 1983] Quinlan, J. R. (1983). Learning efficient classification procedures and their application to chess end games. In Michalski, R. S., Carbonell, J. G., and Mitchell, T. M., editors, *Machine Learning: An Artificial Intelligence Approach*, pages 463–482, San Francisco, CA. Morgan Kaufmann.

[Quinlan, 1993] Quinlan, J. R. (1993). *C4.5: Programs for Machine Learning*. Morgan Kaufmann Publishers Inc., San Francisco, CA.

[Quinlan, 1996] Quinlan, J. R. (1996). Boosting first-order learning. In Arikawa, S. and Sharma, A., editors, *Proc. 7th Intl. Workshop on Algorithmic Learning Theory*, pages 143–155, Berlin. Springer. (LNAI 1160).

[Rasmussen and Williams, 2006] Rasmussen, C. E. and Williams, C. K. I. (2006). *Gaussian Processes for Machine Learning*. MIT Press, Cambridge, MA.

[Rätsch et al., 1999] Rätsch, G., Onoda, T., and Müller, K. R. (1999). Regularizing AdaBoost. In Kearns, M. J., Solla, S. A., and Cohn, D. A., editors, *Advances in Neural Information Processing Systems 11*, pages 564–570. MIT Press, Cambridge, MA.

[Rätsch et al., 2001] Rätsch, G., Onoda, T., and Müller, K.-R. (2001). Soft margins for AdaBoost. *Machine Learning*, 42(3):287–320.

[Rätsch and Warmuth, 2005] Rätsch, G. and Warmuth, M. K. (2005). Efficient margin maximizing with Boosting. *J. Machine Learning Research*, 6:2131–2152.

[Reed and Marks II, 1999] Reed, R. D. and Marks II, R. J. (1999). *Neural Smithing: Supervised Learning in Feedforward Artificial Neural Networks*. MIT Press, Cambridge, MA.

[Rifai et al., 2011] Rifai, S., Vincent, P., Muller, X., Glorot, X., and Bengio, Y. (2011). Contractive auto-encoders: Explicit invariance during feature extraction. In *Proc. 28th Intl. Conf. Machine Learning*, pages 833–840, Bellevue, WA.

[Ripley, 1994] Ripley, B. D. (1994). Neural networks and related methods for classification. *J. Royal Statistical Society*, 56(3):409–456.

[Ripley, 1996] Ripley, B. D. (1996). *Pattern Recognition and Neural Networks*. Cambridge University Press, Cambridge, UK.

[Rojo-Álvarez et al., 2005] Rojo-Álvarez, J. L., Camps-Valls, G., Martínez-Ramón, M., Soria-Olivas, E., Navia-Vázquez, Á., and Figueiras-Vidal, A. R. (2005). Support Vector Machines framework for linear signal processing. *Signal Processing*, 85(2):2316–2326.

[Rojo-Álvarez et al., 2004] Rojo-Álvarez, J. L., Martínez-Ramón, M., de Prado-Cumplido, M., Artés-Rodríguez, A., and Figueiras-Vidal, A. R. (2004). Support vector method for robust ARMA system identification. *IEEE Trans. Signal Proc.*, 52(1):155–164.

[Rokach, 2010] Rokach, L. (2010). *Pattern Classification Using Ensemble Methods*. World Scientific, Singapore.

[Rosenblatt, 1958] Rosenblatt, F. (1958). The perceptron: A probabilistic model for information storage and organization in the brain. *Psychological Review*, 65(6):386–408.

[Ruiz and Lopez-de Teruel, 2001] Ruiz, A. and Lopez-de Teruel, P. E. (2001). Nonlinear kernel-based statistical pattern analysis. *IEEE Trans. Neural Networks*, 12(1):16–32.

[Rumelhart et al., 1986a] Rumelhart, D. E., Hinton, G. E., and Williams, R. J. (1986a). Learning internal representations by error propagation. In Rumelhart, D. E., McClelland, J. L., and The PDP Res. Group, editors, *Parallel Distributed Processing: Explorations in the Microstructure of Cognition, Volume 1: Foundations*, pages 318–362. MIT Press, Cambridge, MA.

[Rumelhart et al., 1986b] Rumelhart, D. E., Hinton, G. E., and Williams, R. J. (1986b). Learning representations by back-propagating errors. *Nature*, 323(9):533–536.

[Schapire, 1990] Schapire, R. E. (1990). The strength of weak learnability. *Machine Learning*, 5(2):197–227.

[Schapire and Freund, 2012] Schapire, R. E. and Freund, Y. (2012). *Boosting: Foundations and Algorithms*. MIT Press, Cambridge, MA.

[Schapire et al., 1998] Schapire, R. E., Freund, Y., Bartlett, P., and Lee, W. S. (1998). Boosting the margin: A new explanation for the effectiveness of voting methods. *Annals of Statistics*, 26(5):1651–1686.

[Schapire and Singer, 1998] Schapire, R. E. and Singer, Y. (1998). Improved boosting algorithms using confidence-rated predictions. In *Proc. 11th Annual Conf. Computational Learning Theory*, pages 80–91, New York, NY. ACM Press.

[Schapire and Singer, 1999] Schapire, R. E. and Singer, Y. (1999). Improved boosting algorithms using confidence-rated predictions. *Maching Learning*, 37(3):297–336.

[Schmidhuber, 2015] Schmidhuber, J. (2015). Deep learning in neural networks: An overview. *Neural Networks*, 61:85–117. Published online 2014; based on TR arXiv:1404.7828 [cs.NE].

[Schölkopf and Smola, 2002] Schölkopf, B. and Smola, A. J. (2002). *Learning with Kernels: Support Vector Machines, Regularization, Optimization, and Beyond*. MIT Press, Cambridge, MA, USA.

[Schwenk and Bengio, 1997] Schwenk, H. and Bengio, Y. (1997). Adaboosting neural networks. In Gerstner, W., Germond, A., Hasler, M., and Nicoud, J.-D., editors, *Proc. 7th Intl. Conf. Artificial Neural Networks*, (ICANN'97), pages 967–972, Berling. Springer. (LNCS 1327).

[Sejnowski and Rosenberg, 1987] Sejnowski, T. J. and Rosenberg, C. R. (1987). Parallel networks that learn to pronounce English text. *Complex Systems*, 1:145–168.

[Sharkey, 1999] Sharkey, A. J. C., editor (1999). *Combining Artificial Neural Nets: Ensemble and Modular Multi-Net Systems*. Springer-Verlag New York, Inc., Secaucus, NJ, USA, 1st edition.

[Shawe-Taylor and Cristianini, 2004] Shawe-Taylor, J. and Cristianini, N. (2004). *Kernel Methods for Pattern Analysis*. Cambridge Univ. Press, Cambridge, UK.

[Shawe-Taylor and Sun, 2011] Shawe-Taylor, J. and Sun, S. (2011). A review of optimization methodologies in Support Vector Machines. *Neurocomputing*, 74(17):3609–3618.

[Shen and Li, 2010] Shen, C. and Li, H. (2010). Boosting through optimization of margin distributions. *IEEE Trans. Neural Networks*, 21(4):659–666.

[Shortliffe, 1976] Shortliffe, E. H., editor (1976). *Computer-Based Medical Consultations: MICYN*. Elsevier–North Holland, New York, NY.

[Siebert, 1987] Siebert, J. P. (1987). Vehicle recognition using rule based methods. Invention Report TIRM-87-018, Turing Institute Research Memorandum. Glasgow (Scotland).

[Simon, 1957] Simon, H. A. (1957). *Models of Man*. Wiley, New York, NY.

[Sklansky and Michelotti, 1980] Sklansky, J. and Michelotti, L. (1980). Locally trained piecewise linear classifiers. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 2(2):101–111.

[Smolensky, 1986] Smolensky, P. (1986). Information processing in dynamical systems: Foundations of harmony Theory. In Rumelhart, D. E., McClelland, J. L., and The PDP Research Group, editors, *Parallel Distributed Processing: Explorations in the Microstructure of Cognition, vol. 1: Foundations*, chapter 6, pages 194–281. MIT Press, Cambridge, MA.

[Sun et al., 2006] Sun, Y., Todorovic, S., and Li, J. (2006). Reducing the overfitting of Adaboost by controlling its data distribution skewness. *Pattern Recognition and Artificial Intelligence*, 20(7):1093–1116.

[Sutskever and Hinton, 2008] Sutskever, I. and Hinton, G. E. (2008). Deep, narrow sigmoid belief networks are universal approximators. *Neural Computation*, 20(11):2629–2636.

[Sutton and Barto, 1998] Sutton, R. S. and Barto, A. G. (1998). *Reinforcement Learning*. MIT Press, Cambridge, MA, 1st edition.

[Szegedy et al., 2014] Szegedy, C., Liu, W., Jia, Y., Sermanet, P., Reed, S., Anguelov, D., Erhan, D., Vanhoucke, V., and Rabinovich, A. (2014). Going Deeper with Convolutions. *arXiv*. 1409.4842v1 [cs.CV].

[Szymanski and Brendan McCane, 2014] Szymanski, L. and Brendan McCane, B. (2014). Deep networks are effective encoders of periodicity. *IEEE Trans. Neural Networks and Learning System*, 25(10):1816–1827.

[Valdovinos and Sanchez, 2006] Valdovinos, R. M. and Sanchez, J. S. (2006). Ensembles of multilayer perceptron and modular neural networks for fast and accurate learning. In *Proc. 5th Mexican Intl. Conf. Artificial Intelligence*, pages 229–236, Washington, DC. IEEE Computer Society.

[Valiant, 1984] Valiant, L. G. (1984). A theory of the learnable. *Commun. ACM*, 27(11):1134–1142.

[Vapnik, 1982] Vapnik, V. (1982). *Estimation of Dependences Based on Empirical Data*. Springer-Verlag, Secaucus, NJ.

[Vapnik, 1995] Vapnik, V. (1995). *The Nature of Statistical Learning Theory*. Springer-Verlag New York, Inc., New York, NY.

[Vapnik, 1998] Vapnik, V. (1998). *Statistical Learning Theory*. Wiley-Interscience, New York, NY.

[Vincent et al., 2008] Vincent, P., Larochelle, H., Bengio, Y., and Manzagol, P.-A. (2008). Extracting and composing robust features with denoising autoencoders. In *Proc. 25th Intl. Conf. Machine Learning*, pages 1096–1103, New York, NY. ACM Press.

[Vurkac, 2011] Vurkac, M. (2011). Clave-direction analysis: A new arena for educational and creative of music technology. *Journal Music Technology Education*, 4:27–46.

[Werbos, 1974] Werbos, P. J. (1974). *Beyond Regression: New Tools for Prediction and Analysis in the Behavioral Sciences*. PhD thesis, Harvard Univ., Cambridge, MA.

[Wolpert, 1992] Wolpert, D. H. (1992). Stacked generalization. *Neural Networks*, 5(2):241–259.

[Yuksel et al., 2012] Yuksel, S. E., Wilson, J. N., and Gader, P. D. (2012). Twenty years of Mixture of Experts. *IEEE Trans. Neural Networks and Learning Systems*, 23(8):1177–1193.

[Zhang and Ma, 2012] Zhang, C. and Ma, Y. (2012). *Ensemble Machine Learning: Methods and Applications*. Springer, New York, NY.

[Zhang et al., 2008] Zhang, C.-X., Zhang, J.-S., and Zhang, G.-Y. (2008). An efficient modified boosting method for solving classification problems. *Comput. Applied Mathematics*, 214(2):381–392.

[Zhou, 2012] Zhou, Z.-H. (2012). *Ensemble Methods: Foundations and Algorithms*. Chapman & Hall/CRC, Boca Raton, FL, 1st edition.

[Zor et al., 2011] Zor, C., Windeatt, T., and Yanikoglu, B. A. (2011). Bias-variance analysis of ECOC and bagging using neural nets. In *Ensembles in Machine Learning Applications*, pages 59–73.