Universidad Carlos III de Madrid

TESIS DOCTORAL

Diversidad en Aprendizaje Profundo por Auto-codificación

Autor:

Ricardo Fernando Alvear Sandoval

Director:

Dr. Aníbal R. Figueiras Vidal

Dpto. de Teoría de la Señal y Comunicaciones

Leganés, julio de 2017

TESIS DOCTORAL

Diversidad en Aprendizaje Profundo por Auto-codificación

Autor:
Ricardo Fernando Alvear Sandoval

Director:
Dr. Aníbal R. Figuiras Vidal

Firma del Tribunal Calificador:

Firma

Presidente:

Vocal:

Secretario:

Calificación:

Leganés, de julio de 2017

AGRADECIMIENTOS

En primer lugar agradezco a Dios por haberme concedido la oportunidad para desarrollarme profesionalmente y por estar presente en todas las etapas de mi vida.

Al Prof. Dr. Aníbal Figueiras Vidal por permitirme formar parte del grupo de investigación Gamma Learning+, en cuyo seno se ha llevado a cabo la presente Tesis Doctoral, cuyas aportaciones han sido fundamentales para que se plasmase en documentos científicos el trabajo realizado durante estos casi 7 años.

A todo el Departamento de Teoría de la Señal y Comunicaciones de la Universidad Carlos III de Madrid, por las herramientas facilitadas y el conocimiento transferido. Y también al Prof. Dr. Monson Hayes por su contribución en las publicaciones.

En el proyecto CASI-CAM-CM (S2013/ICE-2845) de la Comunidad de Madrid se han insertado los trabajos de esta Tesis.

A mis padres José y María por su respaldo desde la distancia: con su sacrificio me han permitido desenvolverme en un país distinto y no notar su ausencia. A mis hermanos: María de Lourdes y Juan Francisco; mis cuñados: Luis y Verónica; y a mis queridísimos sobrinos: Dilan, Erick, Nicole, Fernanda, Francys, David, Jessica, Mateo y Keyla, a quienes auguro mucho éxito y pido que comprendan que con esfuerzo y sacrificio se pueden alcanzar todas las metas que se propongan.

A Dani, un pilar fundamental durante esta época y a quien debo agradecer enormemente su apoyo, y también el de su familia que ahora considero mía: Luisa, Gregorio, Marisa, Yolanda, Pablo, Antonio, Nuria, David e Ismael.

A Jorge y Carlos, por su ayuda para adaptarme a una nueva cultura sin notar la ausencia familiar. A Neila por su cariño y consejos para progresar en el ámbito profesional y que con ese empuje me permitiese formar parte de Telefónica.

Finalmente, a mis amigos que han compartido muchas experiencias en España: Mariléa, Inés, Francis, Iris, Jessica, Darwin, Edgar, Elsa, Karen, Alicia, Lucrecia, Isabel, Gonzalo, Anita, Jhonny, Ingrid, Fernando, Jhonny L., Adil, Anas, Juan José, Efraín, Carlos, Luis, Emilio, Lorena. Y de Ecuador, a quienes me han dado mucho ánimo: Verónica, Nancy, Fanny, Marco, Diana, Leandro, Mireya, Andrea, Irene, Sheny y demás amigos que, aunque no les nombre, los tengo muy presentes.

¡Gracias a todos!

"*Nuestra mayor debilidad reside en rendirnos.*
La forma más segura de tener éxito
es intentarlo una vez más"

Thomas A. Edison
1847-1931

RESUMEN

El diseño de aprendices profundos generales se ha mantenido como reto durante décadas. En el siglo actual se está produciendo la aparición de varios nuevos –y eficaces– procedimientos para ello. Esos procedimientos incluyen los métodos representacionales, que merecen especial atención porque no solo permiten construir máquinas potentes, sino que también extraen relevantes rasgos de alto nivel de las observaciones. Los auto-codificadores expansivos reductores de ruido son (elementos de) una de las familias de máquinas representacionales profundas.

Por otra parte, los conjuntos son una alternativa sólidamente establecida para conseguir soluciones con altas prestaciones para problemas empíricos –basados en muestras– de inferencia. Se valen de la introducción de diversidad en un grupo de aprendices. Obviamente, este es un principio que también puede aplicarse a redes neuronales profundas; pero, sorprendentemente, hay muy pocos estudios que exploran esta posibilidad.

En esta disertación doctoral se investiga si las técnicas convencionales de diversificación –incluyendo la binarización en el caso de bases de datos multiclase– permiten mejorar las prestaciones de clasificadores basados en auto-codificadores expansivos con reducción de ruido. Se usan tanto "Bagging" como "Switching", junto con esquemas de binarización uno-contra-uno y de códigos de salida correctores de errores, sobre dos tipos básicos de arquitecturas: T, que tiene una unidad de auto-codificación común, y G, que también diversifica ese elemento representacional. Los resultados experimentales confirman que –si se incluye la binarización– la combinación de diversidad y profundidad conduce a mejores prestaciones, especialmente con las arquitecturas T.

Para completar la exploración sobre posibles mejoras, se analiza también la aplicación de formas flexibles de pre-énfasis. Tales formas proporcionan por sí solas mejoras de prestaciones, pero las mejoras son muy importantes cuando el pre-énfasis se combina con la diversificación, en especial si se emplean diferentes parámetros de pre-énfasis a diferentes dicotomías en los problemas multiclase. Una distorsión elástica convencional permite alcanzar resultados récord.

Estos resultados no son tan solo relevantes "per se", sino que abren una vía de prometedoras líneas de investigación, las cuales se exponen en el capítulo final de esta tesis.

ABSTRACT

Designing general deep learners has remained as a challenge along decades. The present century sees the emergence of several new effective procedures for it. Among them, representational methods merit particular attention, because they not only serve to build powerful machines, but also extract relevant high-level features of the observations. Expansive denoising auto-encoders are (elements of) one of such representational deep machine families.

On the other hand, ensembles are a well established alternative to get high performance solutions for empirical –sample based– inference problems. They are principled on introducing diversity in a number of different learners. Obviously, this is a principle which can also be applied to deep neural networks, but, surprisingly, there are very few studies exploring this possibility.

In this doctoral dissertation, we investigate if conventional diversification techniques –including binarization for multiclass databases– further improve the performance of expansive denoising auto-encoder based classifiers. Both "Bagging" and "Switching" are used, as well as one-versus-one and error-correcting-output-code binarization schemes, with two basic types of architectures: T, which has a common auto-encoding unit, and G, which also diversifies that representational element. The experimental results confirm that –if binarization is included– combining diversity and depth offers significant performance advantages, specially with T architectures.

To complete the exploration on improving denoising auto-encoding based classifiers, the application of flexible enough pre-emphasis functions is also analyzed. Using this kind of pre-emphasis provides performance advantages by itself, but the advantages are very important when pre-emphasis is combined with diversification, specially if different emphasis parameters are applied to different dichotomies in multiclass problems. A conventional elastic distortion allows record results.

These results are not only relevant by themselves, but they open a series of promising research avenues, that are presented in the final chapter of this thesis.

Índice general

Índice de figuras

Índice de tablas

Capítulo 1

Introducción

1.1. Los orígenes del Aprendizaje Máquina

Debe considerarse que el Aprendizaje Máquina (ML, "Machine Learning") nació con la propuesta de la Regla del Perceptrón por Frank Rosenblatt para entrenar discriminantes lineales con activaciones duras [Rosenblatt, 1958], [Rosenblatt, 1962], al amparo de la visión, hebbiana –que expande conceptos de Santiago Ramón y Cajal– de activación de las neuronas [Hebb, 1949]; y ello, a pesar de que el trabajo de Ronald Fisher sea anterior [Fisher, 1936], porque introdujo un verdadero algoritmo de aprendizaje y no una formulación aproximada.

Las dificultades de extender el aprendizaje a arquitecturas multicapa (MLPs, "Multi-Layer Perceptrons"), que llevaron a Bernard Widrow a reencaminar sus esfuerzos [Widrow and Lehr, 1990] hacia problemas de filtrado [Widrow and Hoff, 1960] –lo que dio lugar a la emergencia de los sistemas adaptativos–, provocaron también la desacertada reacción –una más– de Marvin Minsky [Minsky and Papert, 1969]. Su furioso ataque eliminó casi cualquier interés por los MLPs durante muchos años. Por el contrario, las Máquinas de Aprendizaje (LMs, "Learning Machines") generativas (de las que no nos ocuparemos aquí en detalle, dada la orientación de esta Tesis) se desarrollaron rápidamente a partir

de la aplicación de las ventanas de Emanuel Parzen [Parzen, 1962] a regresión [Nadaraya, 1964], [Watson, 1964], y otras líneas de trabajo en ML avanzaron con éxito, como es el caso del agrupamiento máquina, sobre todo con las contribuciones de Teuvo Kohonen en Mapas Auto-Organizados [Kohonen, 1982], [Kohonen, 1989].

La posibilidad de recurrir a activaciones blandas –que se desarrolló independientemente en Estadística como regresiones logística y probit; véase [Bishop, 2006], por ejemplo– y a la regla de la cadena para permitir un entrenamiento de los MLPs basado en la minimización de la versión muestral de un coste, fue desarrollándose silenciosamente [Werbos, 1974], [Werbos, 1994], [Parker, 1982], [Parker, 1985], [LeCun, 1985], hasta que, en 1986, David Rumelhart y sus colaboradores [Rumelhart et al., 1986a], [Rumelhart et al., 1986b] la formalizaron completamente. El impacto de ese avance fue –y sigue siendo– extraordinario, tanto en aportaciones teóricas o técnicas cuanto en aplicaciones; lo que queda evidenciado por los muchos libros de texto dedicados a las Redes Neuronales (NNs, "Neural Networks") –denominación que incluye otras LMs también relevantes, como las Redes de Funciones Radiales de Base (RBFNs, "Radial Basis Function Networks") y las Máquinas de Boltzmann (BM, "Boltzmann Machines"), entre las que los MLPs conservan un lugar preeminente– desde poco después. Entre muchas decenas de esos textos, destacaremos aquí [Pao, 1990], [Hecht-Nielsen, 1990], [Hertz et al., 1991], [Haykin, 1994], [Zurada, 1994] y [Bishop, 1995] de entre los pioneros, así como los más avanzados –que también tratan otros tipos de LMs– [Ripley, 1996], [Duda et al., 2001] –que incluye numerosos detalles históricos–, [Hastie et al., 2001] y el ya citado [Bishop, 2006], junto con los extensos manuales [Maren et al., 1990], [Arbib, 2002] y el muy práctico [Reed and Marks II, 1999].

Con el paso del tiempo se manifestaron las verdaderas limitaciones de los MLPs, adicionales a un carácter sustancialmente estático y a la necesidad de reentrenarlos ante cualesquiera cambios en la información disponible. Si bien George Cybenko [Cybenko, 1989] y Kurt Hornik y sus colegas [Hornik et al., 1989] demostraron que un MLP con una sola capa oculta es un aproximador universal –de modo no cons-

tructivo; es decir, sin establecer el número de unidades de dicha capa oculta–, la disponibilidad de un número limitado de ejemplos de entrenamiento –en principio, pares de entrada y sus etiquetas, a las que se han de aproximar las correspondientes salidas– constituye un serio impedimento para obtener provecho de esa teóricamente ilimitada capacidad expresiva.

Posiblemente por ello irrumpieron arrolladoramente en los 1990 las llamadas Máquinas de Núcleos (KM, "Kernel Machines"), en sus formas esenciales de Procesos Gaussianos (GP, "Gaussian Processes") para regresión y Máquinas de Vectores Soporte (SVM, "Support Vector Machines") para clasificación –aunque existen modificaciones de ambas familias para utilizarlas en el ámbito natural de la otra.

Pueden considerarse dichas familias como el resultado de la conveniente aplicación del conocido como "truco del núcleo" –también conocido como Teorema del Representante– [Aizerman et al., 1965], [Kimeldorf and Wahba, 1971], en que se introduce un núcleo de Mercer –función definida positiva general– en la elaboración de planteamientos que suponen transformaciones no lineales de los datos: transformaciones que el "truco" hace innecesario especificar. Con el "truco", se pueden obtener los GPs como una fácil extensión del filtro de Wiener (Wiener-Hoff, Wiener-Kolmogorov) [Wiener, 1949][1], [Kolmogorov, 1939] a observaciones multidimensionales e irregularmente espaciadas, aunque pueden también presentarse desde otras perspectivas: véase [Rasmussen and Williams, 2006].

Las SVMs parten del concepto de dimensión de Vapnik-Chervonenkis, largamente trabajado por el primero [Vapnik, 1982], [Vapnik, 1995] y [Vapnik, 1998]. En 1992, una combinación práctica del resultante principio de Máximo Margen (MM, "Maximal Margin") con el "truco del núcleo" condujo a su primera versión reducible a programación cuadrática [Boser et al., 1992]. A partir de ese momento se produjo una explosión de contribuciones proponiendo mejoras y variantes, de la que aquí no procede dar cuenta detallada porque el carácter local de estas LMs las inhabilitan –en principio– para construir Aprendices Profundos (DLs,

[1] Editado muchos años después de haber sido escrito.

"Deep Learners"), que constituyen uno de los ingredientes principales de los estudios y experimentos de esta Tesis. No obstante, y por mor de completitud, haremos referencia a algunos tutoriales y algunos textos selecionados: entre los primeros, [Burges, 1998], [Müller et al., 2001] y [Shawe-Taylor and Sun, 2011]; y de los segundos, [Schölkopf et al., 1999], [Herbrich, 2001], [Schölkopf and Smola, 2002], [Shawe-Taylor and Cristianini, 2004] y [Wang, 2005].

El mérito de las KMs, y en particular de las SVMs, radica en que con sólidos fundamentos les permitieron acaparar buena parte del interés de la comunidad activa en ML pese a que al tiempo se desarrollaba una de las soluciones a la anteriormente señalada limitación en el aprendizaje de los MLPs y otras LMs anteriores: los conjuntos de LMs, de los que nos ocuparemos acto seguido. Sin embargo, debe advertirse de que el principio de MM puede ser extendido a los MLPs con cierta facilidad [Lázaro-Teja et al., 2016]. Además, el explosivo desarrollo de los DLs ha reducido mucho la atención a las KMs.

1.2. Los conjuntos de Máquinas de Aprendizaje

Aunque la binarización de problemas multiclase es de pleno derecho un procedimiento de construcción de conjuntos de LMs, tiene una naturaleza diferente de la que caracteriza a los que ahora vamos a presentar; por ello, pospondremos su revisión a un apartado propio –que sigue a éste.

El germen del desarrollo de los conjuntos de LMs se encuentra en el trabajo de Lee Valiant [Valiant, 1982], que propone la Teoría de lo Probablemente Casi Correcto (PAC, "Probably Almost Correct"), en la que se apoya la línea que construye clasificadores fuertes (de altas prestaciones) agregando clasificadores débiles (poco mejores que la toma de decisiones al azar), cuya manifestación más relevante son los conjuntos construidos mediante "Boosting", técnica que goza de merecidísima fama y que revisaremos brevemente con este mismo apartado. Y debe señalarse [Hansen and Salamon, 1990] como el primer análisis matemático de las agregaciones

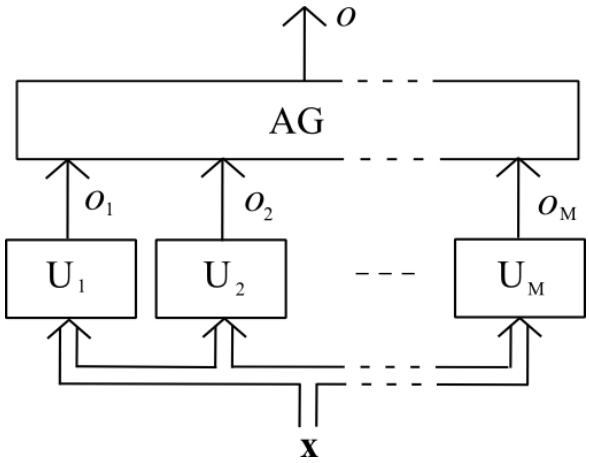

Figura 1.1: Arquitectura genérica (en operación) de un ME. **x**: entrada; U_m: unidades; o_m: salidas de las unidades; AG: agregación; o: salida del ME.

de LMs. Véase que, como se ha anticipado, la aparición de los conjuntos de LMs es simultánea con la de las KMs.

Aunque la nomenclatura no está uniformizada –y puede decirse que resulta confusa– emplearemos aquí las siglas MEs ("Machine Ensembles") para referirnos a cualquier forma de agrupación de LMs.

Si hubiese que señalar un principio común para la construcción de las muy numerosas y variadas familias de MEs, sería obligado mencionar la diversidad en sus componentes –unidades o aprendices–, que es la que permite obtener ventaja mediante la agregación de sus salidas. Aprendices diversos (diversificados) son los que abordan un problema en condiciones suficientemente distintas como para que las soluciones que proporcionan no sean idénticas, sino diferentes: la apropiada combinación de esas soluciones supone una especie de "promediado", que da lugar a una solución final mejor que la de cualquier unidad por separado, e incluso que cualquier LM monolítica. Así que es posible decir que estamos tratando con arquitecturas como la que se muestra en la Figura 1.1, en el momento de la operación (se supone una única salida).

Como quiera que en nuestros trabajos la diversidad tiene un muy relevante papel,

procede incluir aquí un rápido repaso de los MEs. Naturalmente, con la concisión debida: quienes deseen mayor detalle pueden recurrir a los muchos textos monográficos dedicados al ámbito, de los que destacamos [Sharkey, 1999], [Kuncheva, 2004], [Rokach, 2010], [Zhang and Ma, 2012], [Zhou, 2012] y [Schapire and Freund, 2012].

Puede decirse que las diferentes familias de MEs surgen de dos troncos principales: los comités, en los que se entrenan inicialmente las unidades, introduciendo diversificación mediante procedimientos específicos para ello, y, tras lo anterior, se lleva a cabo la agregación (típicamente mediante esquemas sencillos, casi siempre no entrenables, como la votación con decisión mayoritaria o el promediado aritmético directo); y los consorcios, en los que el entrenamiento se aplica juntamente a aprendices y agregación –por lo que, en general, ofrecen mejores prestaciones.

La Figura 1.2 simboliza las posibilidades de diversificación abiertas para el diseño de comités: modificar los ejemplos disponibles para entrenar (bien las observaciones $\mathbf{x}^{(n)}$, bien los blancos $t^{(n)}$), es decir, la información de que dispone cada aprendiz, o modifico la arquitectura de los aprendices. En la práctica, esta segunda opción se ha empleado en muy pocas ocasiones, mereciendo mención las Selvas Aleatorias (RFs, "Random Forests"), en que se diversifican los árboles constituyentes mediante ramificaciones probabilísticas, acompañándola de otras diversificaciones –típicamente reducciones a subespacios de las observaciones [Ho, 1998], un método de eficiencia limitada por sí solo–. Las referencias principales sobre RFs son [Breiman, 2001] y [Archer and Kimes, 2008].

Los métodos fundamentales de diversificación mediante la modificación de los ejemplos son dos, ambos debidos a la agudeza del reconocido investigador Leo Breiman (fallecido, para tristeza de todos, hace pocos años): "Bagging" (Bootstrap aggregating) [Breiman, 1996] –que tiene también una versión probabilística, "Wagging" [Bauer and Kohavi, 1999]–, en el que cada aprendiz recibe una versión remuestreada mediante "Booststrap" de los ejemplos, y "Switching" (originalmente, "aleatorización de las salidas") [Breiman, 2000], en el que los blancos se modifican aleatoriamente. Como quiera que "Bagging" y "Switching" se utilizarán como méto-

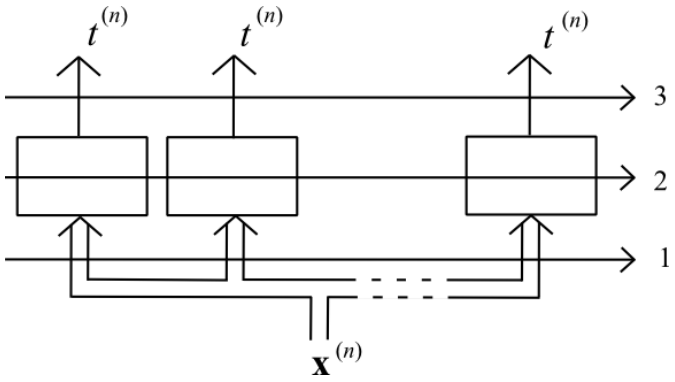

Figura 1.2: Diversificación en comités: 1, observaciones; 2, arquitecturas; 3, blancos (cabría también: 4, agregación).

dos de diversificación en esta Tesis, se describirán en el Capítulo 2. No obstante, debe señalarse desde el principio que estos métodos requieren que los aprendices del comité sean inestables, es decir, sensibles a cambios en el conjunto de entrenamiento –y los MLPs, desde luego, lo son–, pero no necesariamente débiles.

Debe incluirse también entre los comités el método de apilamiento o amontonamiento, "Stacking" [Wolpert, 1992], en el que los aprendices se parametrizan con subconjuntos disjuntos del conjunto de ejemplos para el entrenamiento, y se añade una capa de agregación que se entrena evitando repeticiones en la presentación de ejemplos. Aunque el principio aplicado resulta atractivo, e incluso se han propuesto modificaciones y extensiones del mismo, no ha proporcionado resultados especialmente brillantes.

Cerraremos la concisa exposición de los comités con un par de observaciones. La primera, que resulta sorprendente la escasísima atención dedicada a la diversificación de arquitecturas. Si bien algunos autores consideran como tales los métodos "Drop-Out" [Srivastava et al., 2014] y "Drop-Connect" [Wan et al., 2013] –que son presentados en el contexto de los DLs,– en que se decide probabilísticamente si una activación o un peso de un MLP se actualiza o no con cada ejemplo, creemos más acertado considerarlos como novedosos –y valiosos– métodos de regularización: otra

cosa sería recurrir a los muy variados métodos de poda existentes [Reed, 1993] para crear aprendices diversificados. La segunda observación es otra expresión de sorpresa, esta vez, por el escasísimo esfuerzo dedicado a aplicar diversidad a la etapa de agregación (y decimos escasísimo admitiendo que el "Stacking" podría considerarse como tal).

Pasamos ahora a una somera presentación de los consorcios, aunque dedicaremos cierto espacio al claramente más importante, "Boosting", no porque hagamos posterior uso de él, sino por su mucho interés intrínseco.

En lo referente a los consorcios de LMs, también nos encontramos con una amplísima variedad de métodos para diseñarlos. Aquí discutiremos las tres familias que consideramos –y se consideran en general– más importantes: "Boosting", Mezclas de Expertos (MoEs, "Mixtures of Experts") y Aprendizaje por Correlación Negativa (NCL, "Negative Correlation Learning"). Por su particularísimo valor, empezaremos por el "Boosting".

Las raíces del "Boosting" son las que más profundamente se hunden en la Teoría PAC: de hecho, la primera versión de estos métodos –un proceso de filtrado en etapas, que pronto se demostró inferior a otros con idéntico propósito– apareció en un trabajo, [Schapire, 1990], cuyo título recurre expresamente a la denominación "aprendizaje débil": y aprovecharemos el momento para advertir de que el carácter débil de los aprendices es un requisito para estos procedimientos.

Los métodos de "Boosting" parten de entrenar secuencialmente aprendices débiles con versiones de muestras de entrenamiento enfatizadas (es decir, ponderadas en la versión muestral del coste a minimizar) de acuerdo con la dificultad que hayan ofrecido para ser bien clasificadas hasta el momento en que se diseña cada aprendiz; al tiempo, las salidas se van agregando linealmente para minimizar un coste del tipo exponencial de una función de margen: $\exp(-to)$, siendo $o = \sum_n \alpha_m o_m$ y $\{o_m\}$ las salidas de los aprendices. Es decir, procede como se simboliza en la Figura 1.3.

La consideración de aprendices de salidas binarias ± 1 posibilita una formulación cerrada de todo el proceso, dando lugar al AdaBoost (AB, "Adap-

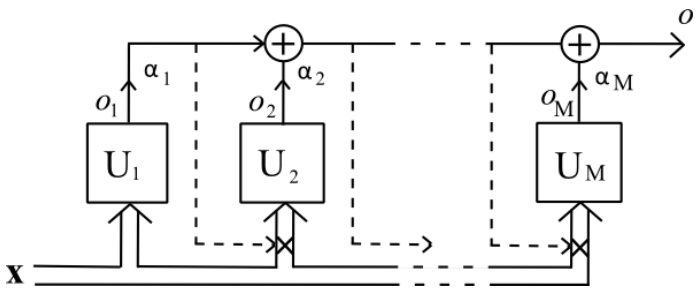

Figura 1.3: Proceso iterativo para construir conjuntos por "Boosting". Las líneas discontinuas indican la aplicación de los énfasis (×).

tive Boosting") [Freund and Schapire, 1995], [Freund and Schapire, 1996a], [Freund and Schapire, 1997]. Si se posibilita que los aprendices tengan salidas en $(-1, 1)$, se hace necesario minimizar una cota del coste para conseguir una formulación cerrada, teniéndose el Real AdaBoost (RAB) [Freund and Schapire, 1996b], [Schapire and Singer, 1998], [Schapire and Singer, 1999]. Y ya desde el principio surgieron un sinnúmero de modificaciones y variantes que no es posible ni siquiera citar en un espacio prudencial –por lo que se remite otra vez a [Schapire and Freund, 2012]–. A cambio, sí concederemos espacio a la visión de Breiman: para diseñar conjuntos por "Boosting", lo importante es los principios –proceder a entrenar aprendices débiles con ejemplos convenientemente enfatizados–, y no las versiones concretas. Ésta es la visión que puede denominarse del "Arcing" ("Adaptive resampling and combining) [Breiman, 1998], [Breiman, 1999a] y [Breiman, 1999b], que nos parece una visión fundamental, y que de hecho abrió el camino a aún más modificaciones y variantes, el relajar los requisitos para llevarles a cabo. A guisa de ejemplo, por ello fue posible proponer un alto número de mecanismos correctores para las escasas ocasiones en que el "Boosting" presenta problemas de sobreajuste (una de las más destacables características de los métodos básicos de "Boosting" es su resistencia al sobreajuste a medida que se incorporan nuevos aprendices): además de la simple y directa eliminación de muestras [Freund, 2001], se han ido aplicando regularización [Rätsch et al., 1999], márgenes blandos [Rätsch et al., 2001], mini-

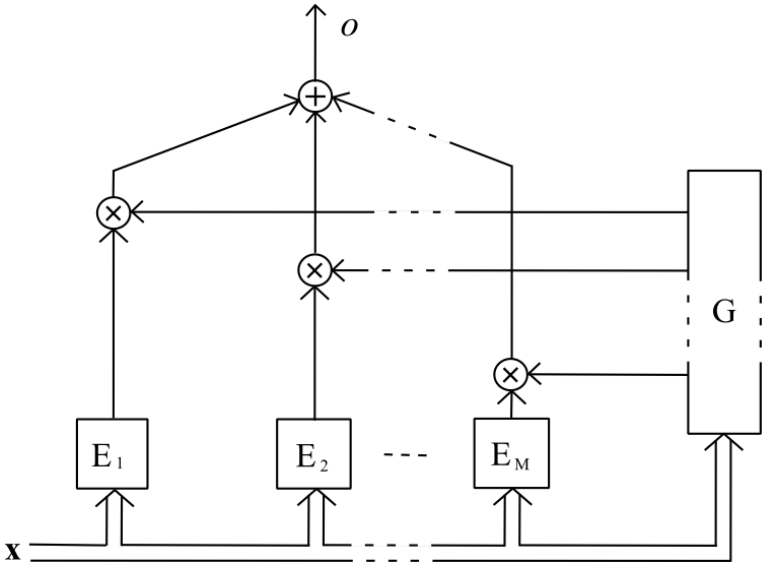

Figura 1.4: Mezcla de Expertos. E_m: expertos; G: puerta (soft-max); $\mathbf{x}$: entrada; o: salida final.

mización del margen [Rätsch and Warmuth, 2005], penalización de la asimetría de los datos [Sun et al., 2006], énfasis mixtos [Gómez-Verdejo et al., 2006], [Gómez-Verdejo et al., 2008] –aún cabrían otras versiones–, submuestreo [Zhang et al., 2008], optimización de margen [Shen and Li, 2010] y alisado por proximidad [Ahachad et al., 2014]. También el muestreo estratificado se ha encontrado efectivo para debilitar SVMs [Mayhua-López et al., 2015].

Otra familia de consorcios que es obligado mencionar es las MoEs. Su forma general se representa en la Figura 1.4. Como puede verse, una puerta o árbitro realiza la tarea de agregación combinando apropiadamente –la salida del árbitro es softmax– los resultados de las unidades –aquí llamadas expertos– para constituir la salida final: una mezcla de (las salidas de) los expertos, por tanto.

La versión original de los MoEs [Jacobs et al., 1991] se propuso para regresión, y recurría a aprendices lineales y puerta lineal softmax, con lo que era sencillo un entrenamiento por Máxima Verosimilitud (ML, "Maximum Likelihood"), directo o recurriendo a Promedio-Maximización (EM, "Expectation-Maximization",

[Dempster et al., 1977]).

La arquitectura original ofrecía una capacidad expresiva limitada. Por ello, se propusieron dos líneas para aumentarla: las agrupaciones jerárquicas de MoEs [Jordan and Jacobs, 1994], [Jordan and Xu, 1995] y el aumento de la capacidad expresiva de los expertos [Olteanu and Rynkiewicz, 2008] o de la puerta [Weigend et al., 1995], [Zeevi et al., 1997] y [Principe, 2001]. Pero los intentos de extender estos diseños a funciones de clasificación no alcanzaron el éxito esperado [Jordan and Jacobs, 1992]; como se deduce del resultado de reestructuraciones que permitan entrenamiento MM [Omari and Figueiras-Vidal, 2013], casi con certeza a causa de aplicar ML.

La dificultad de los MoEs para tratar con problemas de clasificación y la alta complejidad de los diseños de [Omari and Figueiras-Vidal, 2013] no hacen aconsejable recurrir a esta forma de diversificación en los primeros intentos de introducir ésta en DLs, tal y como se persigue en esta Tesis. Pero no ha de descartarse su futuro empleo en tal sentido: por eso, ha de citarse el artículo tutorial [Yuksel et al., 2012], que presenta otras muchas variantes de los métodos y algoritmos básicos.

Concluiremos este apartado con un apartado resumen de los consorcios constuidos mediante NCL, que se introdujeron en [Liu and Yao, 1999a], [Liu and Yao, 1999b], y pueden considerarse extensiones del clásico aprendizaje por cascada de correlaciones [Fahlman and Lebiere, 1990].

En lo esencial, el NCL propone minimizar la versión muestral de

$$\sum_{m=1}^{M}[C(t,o_m)+\lambda(o-o_m)\sum_{m'\neq m}(o-o_{m'})] \tag{1.1}$$

siendo $C(\cdot)$ una función de coste, m, el índice de las máquinas aprendices, o_m sus salidas y o la salida total; λ es un parámetro que ajusta la influencia del segundo sumando, que penaliza la correlación de las salidas con referencia o.

Aún existiendo muchas variantes, el NCL también resulta poco efectivo para diseñar consorcios de clasificadores. Se incluye aquí no solo por razones de completitud, sino porque no resulta improbable que esa inefectividad se corrija pronto mediante

la combinación con otros procedimientos: al fin y al cabo, el NCL puede considerarse como diversificación de las salidas, y en esa línea queda mucho por hacer.

1.3. Binarización

Tradicionalmente, los MLPs diseñados para resolver problemas multiclase proporcionan tantas salidas como clases mediante una activación softmax (también conocida como activación de Potts), tal como

$$o_c(\mathbf{x}) = \frac{\exp(\mathbf{w}_{ce}^T \mathbf{z}_e)}{\sum_{c'} \exp(\mathbf{w}_{c'e}^T \mathbf{z}_e)} \tag{1.2}$$

donde o_c indica la salida asociada a la clase C, $1 \leq c \leq C$ (C es el número de clases), $\mathbf{z}_e$ es el vector extendido de salidas de la última capa oculta, y $\mathbf{w}_{ce}$ es el vector extendido de los pesos asociado a la clase c. De este modo, todas las salidas son no negativas y suman 1, y además la aplicación de una adecuada función de coste tipo divergencia de Bregman [Bregman, 1967], [Cid-Sueiro et al., 1999] permite que $o_c(\mathbf{x})$ sea un (razonable) estimador de la probabilidad a posteriori de que la muestra $\mathbf{x}$ pertenezca a la clase c, $o_c(\mathbf{x}) = \hat{Pr}(c|\mathbf{x})$. Con ello, queda fundamentado el criterio de asignar $\mathbf{x}$ a la clase cuya salida es la mayor de entre las C existentes.

Pese a la consistencia de lo expuesto en el párrafo anterior, en la práctica el entrenamiento de MLPs monolíticos para resolver problemas multiclase presenta mayor dificultad y proporciona resultados de calidad limitada; lo que parece natural, dado que una multiclasificación es intrínsecamente más compleja que una clasificación binaria: piénsese en las fronteras necesarias para la primera.

En virtud de esas dificultad y limitación, desde ha largo tiempo se ha buscado transformar los problemas multiclase en un conjunto de dicotomías –o problemas binarios– de cuyas soluciones se puede deducir la clase a que hay que atribuir cada muestra. Designaremos estos métodos, en su conjunto, como procedimientos de binarización.

El más inmediato de esos procedimientos se conoce como Uno contra Uno (OvO,

"One versus One") [Hastie and Tibshirani, 1998], y consiste en crear $C(C-1)/2$ dicotomías de la forma clase c contra clases c'. En situación ideal, la clase correcta sería seleccionada siempre, y bastaría observarlo. En la práctica no será así, y tendrá que recurrirse a seleccionar la clase con mayor número de victorias (o menor número de derrotas). Es fácil apreciar que un error de un clasificador binario no siempre puede ser corregido, lo que limita la calidad de las binarizaciones OvO. Ello puede –al menos en parte– deberse a que no se utiliza más que una parte de los ejemplos de entrenamiento para diseñar cada clasificador binario –los correspondientes a las dos clases implicadas–. Además, $C(C-1)/2$ puede resultar un número prohibitivo de máquinas cuando C es relativamente alto (para $C = 10$ se tendrán 45 máquinas OvO).

Ya desde aquí resulta apropiado resaltar que la binarización es una forma de diversificación (de naturaleza singular, eso sí): sustituye una máquina monolítica por un conjunto de máquinas que se enfrentan a problemas diversos (relacionados con el multiclase original), y eso proporciona normalmente las ventajas esperables de la aplicación de la diversidad; incluso para procedimientos OvO en muchas aplicaciones reales.

Una alternativa bien conocida al OvO es la binarización Uno contra el Resto (OvR, "One versus Rest")[2]. Las dicotomías consisten en enfrentar cada clase contra todas las demás –con lo que siempre se hace uso de todos los ejemplos, y el número de LMs necesario se reduce a C–. La decisión final se toma eligiendo la clase que ofrece mayor salida en su test contra las demás. Se percibe que este criterio se hace necesario porque no resulta fácil que una clase se imponga al conjunto de todas las otras; lo que en cierta medida se debe a que los problemas binarios OvR son desequilibrados (R tiene más muestras que O –al menos en la mayoría de los casos–, y eso propicia soluciones en que se decide siempre a favor de R).

Aunque no hay conclusiones acerca de si OvO es preferible a OvR o al revés [Allwein et al., 2000] –creemos firmemente que la respuesta depende del problema–,

[2]Preferimos OvR a la menos apropiada Uno contra Todos (OvA, "One versus All")

en (algunos de) nuestros experimentos recurriremos al primero de estos procedimientos, ya que las prestaciones pretendidas y el desequilibrio resultante del número de clases considerado ($C = 10$) suponen que el riesgo de emplear OvR sea inaceptablemente alto.

La tercera –y más elaborada– forma de dicotomizar problemas multiclase es la que recurre al diseño y aplicación de los denominados Códigos de Salida Correctores de Errores (ECOCs, "Error Correcting Output Codes"). Dado que su fundamento es ajeno al ML –se encuentra en los Códigos Correctores de Errores (ECCs, "Error Correcting Codes") a los que se recurre en transmisión digital–, les dedicaremos un prudente espacio.

En los canales de comunicación, el omnipresente ruido y la distorsión hacen inevitable que se produzcan errores incluso cuando se transmite del modo más resistente a estas perturbaciones, en forma binaria. Recuérdese que Claude Shannon demostró –de forma no constructiva– que un canal de comunicación ruidoso limita la cantidad de información recuperable que puede atravesarlo, su capacidad [Shannon, 1948a], [Shannon, 1948b]. Por tanto, se producen irremediablemente errores en esos bits transmitidos: controlar dichos errores –detectarlos y corregirlos– es una necesidad en comunicaciones. Es revelador que la distancia de Hamming apareciese en este ámbito muy poco después [Hamming, 1950] de que viese la luz el trabajo fundamental de Shannon.

La distancia de Hamming permite entender muy fácilmente los principios de los ECCs: si los bits informativos se transmiten con los códigos 000 y 111, un error en cualquiera de ellos puede corregirse, ya que su aparición hace que la distancia de Hamming –número de bits diferentes– entre el código recibido y el original sea 1, mientras que es 2 con respecto al código alternativo. Eso es así porque la distancia de Hamming entre los códigos transmitidos es 3, lo que permite corregir un error: obviamente, no más. El empleo constructivo y las extensiones de esta visión han originado los avances en ECCs.

En el caso de los ECOCs, cabe considerar cada clase como un símbolo, y atribuirle

	P_1	P_2	P_3	P_4	P_5	P_6	P_7
C_1	1	1	1	1	1	1	1
C_2	1	1	1	0	0	0	0
C_3	1	0	0	1	1	0	0
C_4	0	1	0	1	0	1	0

Tabla 1.1: Un ECOC para un problema de cuatro clases. C_c: clase; P_l: dicotomía.

una palabra código de forma que las distancias de Hamming entre todas las palabras sea alta. Los bits en la posición l-ésima de esas palabras definen una dicotomía: las clases asociadas a 1s vs. las clases asociadas a 0s. Esas dicotomías se atacan con clasificadores binarios y, si el número de errores que se produzcan en su solución es menor que la (menor) distancia de Hamming, la decisión por síndrome –la clase más cercana al código de las soluciones– es la correcta.

Paradójicamente, el empleo de esta concepción, aunque de forma rudimentaria, antecedió al de las formulaciones OvO y OvR: se propuso en [Sejnowski and Rosenberg, 1987]. Fue el excelente trabajo de Thomas Dietterich y Ghulum Bakiri [Dietterich and Bakiri, 1995] el que asentó las bases del diseño de ECOCs. En él se recomienda el método exhaustivo de generación –basado en las funciones de Rademacher– que hemos usado en la Tabla 1.1 (la recomendación indica que se emplee para problemas de hasta 6 clases: opinamos que esa restricción corresponde a tiempos de mayores limitaciones de potencia computacional). Es inmediato comprobar que la distancia de Hamming entre las palabras código es 3: se puede corregir un error en la solución de alguno de los problemas. También se presenta un código "ad hoc" de longitud 15 y distancia de Hamming mínima 7 para el reconocimiento de dígitos manuscritos al que recurriremos más adelante para experimentar con los diseños máquina que propondremos en la presente Tesis.

Advierten Dietterich y Bakiri no sólo de la importancia de obtener una distancia de Hamming alta, sino de la conveniencia de que las dicotomías resultantes sean

también diversas, ya que no sería extraño que las soluciones a problemas binarios semejantes diesen lugar a muchos errores comunes. A ello cabe añadir que es aconsejable que los problemas binarios resulten equilibrados, o bien a que se preste atención al desequilibrio en su solución. Como se ve, condiciones que no aparecen en los ECCs.

Aunque no han sido pocos los esfuerzos dedicados a establecer satisfactorios métodos de diseño de ECOCs, desde la selección de códigos generados aleatoriamente [Berger, 1999] hasta aproximaciones formales como la que considera las matrices de Hadamard [Zhang et al., 2003], pasando por algoritmos subóptimos para su construcción [Pimenta and Gama, 2005] –un buen resumen se incluye en el capítulo 6 de [Rokach, 2010]–, estamos firmemente convencidos de que aún no se han explotado suficientemente todas las posibilidades que ofrece la teoría de los ECCs para la construcción de ECOCs. Eso es por lo que vamos a incluir aquí una bibliografía seleccionada de ECCs, aún a riesgo de que alguien lo considere inoportuno: si otro obtiene provecho de su consulta, será mayor la satisfacción que el gasto.

Sería imperdonable no citar a los clásicos, en donde se aprecian las visiones de partida: [Abramson, 1963], [Forney, 1966], [Berlekamp, 1968], [Peterson and Weldon Jr., 1972], [McEliece, 1977] y [Hamming, 1986]. Entre los posteriores, elegimos [Adámek, 1991], [Honary and Markarian, 1997], [Heegard and Wicker, 1999] y [Lin and Costello Jr., 2004]. Y, para extinguir toda duda sobre la relación entre la Teoría de la Información y el ML, bastará traer a colación [MacKay, 2003].

1.4. Aprendizaje Profundo

Las arquitecturas profundas –en lo sustancial, MLPs con varias capas ocultas– constituyen la segunda posibilidad de conseguir alta capacidad expresiva en MLs: las sucesivas transformaciones entrenables hacen aparecer un elevado número de pesos. Pero conviene advertir desde el primer momento de que, a pesar de que la formulación del algoritmo BP produce la falsa impresión de que basta con iterarlo para conseguir

una Red Neuronal Profunda (DNN, "Deep Neural Network") –que es la denominación que aquí adoptaremos, por considerarla mayoritaria, para designar genéricamente estas arquitecturas–, su desarrollo resultó al menos tan dificultoso como el del citado algoritmo: cuando se aplica a estructuras de varias capas ocultas, las derivadas del coste respecto a los pesos se desvanecen a medida que se retrocede hacia la entrada –posiblemente debido a la proliferación de casos de parálisis, o salidas de unidades intermedias en la zona de saturación de sus activaciones, lo que tiene relación con los procesos de inicialización de pesos necesarios para el primer paso del algoritmo BP–, y, además, la inadecuada selección de pasos de adaptación diferentes para cada capa –un proceso para nada trivial– propicia la caída en mínimos locales durante la búsqueda del coste muestral mínimo, como se discute claramente en [Bengio, 2009].

Históricamente hablando, la primera arquitectura profunda no se corresponde con lo que estrictamente se podría llamar una DNN: fue el Método de Agrupamiento para Manejo de Datos (GMDH, "Group Method of Data Handling") en su versión constructiva capa a capa [Ivakhnenko, 1968], [Ivakhnenko, 1971]. En esencia, se trata de construir polinomios de alto orden en muchas variables a base de iterar construcciones de polinomios de orden dos en dos variables, descartando los resultados menos eficaces para evitar la explosión dimensional. Ha habido mejoras y extensiones en esta dirección [Onwubolu, 2015], pero las vías alternativas las apartaron del centro de la escena.

Pocos años después, Kunihiko Fukushima propuso el Neocognitrón [Fukushima, 1979], [Fukushima, 1980], que sí puede considerarse una arquitectura profunda tipo DNN, pero que inicialmente carecía de un algoritmo de entrenamiento y había que ajustar artesanalmente. Se trata de un apilamiento de estructuras de dos capas, la primera de las cuales aplica una misma transformación lineal –pesos compartidos– de longitud (o área) limitada a segmentos (o subáreas) solapadas del mismo tamaño definidas sobre las variables de entrada (casi siempre secuencias temporales, como el habla, o imágenes), seguida de una capa de promediado –actualmente se prefiere la obtención del máximo local [Ranzato et al., 2007],

[Scherer et al., 2010].

El relativamente anterior reducido número de pesos diferentes por capa convolucional permitió a Yann LeCun aplicar su versión del entrenamiento por BP [LeCun, 1985] a estas arquitecturas, con una salida para clasificación, dando lugar a las Redes Neuronales Convolucionales Profundas (DCNNs, "Deep Convolutional Neural Networks"; o simplemente CNNs) [LeCun et al., 1989], [LeCun et al., 1990] y [LeCun et al., 1998]. Conviene destacar que la base de dígitos manuscritos MNIST (extraída de códigos postales en correspondencia estadounidense), clásica como banco de pruebas para evaluar clasificadores, aparece por primera vez en las dos referencias iniciales anteriores; y que LeCun tuvo éxito al conseguir que el US Post Office implementase una versión de sus diseños para reconocer automáticamente dichos códigos.

No resulta necesario argumentar que las DCNNs son arquitecturas "ad hoc" para problemas de clasificación en que las instancias a clasificar presentan una estructura adecuada para un tratamiento convolucional, como ocurre con las imágenes y con las series temporales localmente estacionarias, como las señales de voz. En estos casos, consecutivas mejoras han permitido obtener excelentes resultados: así ocurre con la versión diversificada según sub-imágenes de [Cireşan et al., 2011], que obtiene para MNIST una tasa de error de clasificación del 0.27 %, y, sobre todo con la versión diversificada según distorsiones [Cireşan et al., 2012a], denominada CNN Multi-Columna (MC-CNN, "Multi-Column CNN"), que la rebaja a 0.23 %. Estos diseños han sido aplicados con éxito a otros muchos problemas, como la clasificación de señales de tráfico [Cireşan et al., 2012b].

Lo expresado en el párrafo precedente indica que, independientemente del éxito de las DCNNs, subsistían serias dificultades en el desarrollo de las DNNs. Una posibilidad alternativa empezó a abrirse paso a partir de que se sugiriese en [Ballard, 1987]: recurrir a una profundización no supervisada para, tras ello, añadir una etapa de clasificación supervisada y, en su caso, refinar el entrenamiento. La citada propuesta inicial propuso recurrir a la Auto-Codificación (AE, "Auto-Encoding"); volveremos

sobre ella, pero ahora hemos de pasar a la primera de estas aproximaciones que alcanzó éxito: la basada en las máquinas que minimizan una función de energía, cuya genuina representación ostentan las Máquinas de Boltzmann (BM, "Boltzmann Machines") [Ackley et al., 1985], [Hinton and Sejnowski, 1986].

En forma resumida, una BM contiene nodos observables y nodos ocultos, y se le asocia una probabilidad de estados proporcional a una exponencial de menos una función de energía, que –de acuerdo con la mecánica de Boltzmann– se minimiza para determinar el estado de la BM, fijando los valores observables (a las entradas y a las salidas deseadas). El proceso requiere, en general, la aplicación de muestreo Monte Carlo. La posibilidad de apoyarse en estas ideas para construir DNNs se cimentó en varios pilares. El primero, recurrir a las BMs Restringidas (RBMs, "Restricted Boltzmann Machines") [Smolensky, 1986], sin conexiones entre las unidades de cada capa, para construir las Redes Profundas de Creencias (DBNs, "Deep Belief Networks") [Hinton et al., 2006]. La restricción arquitectónica de las RBMs permite un entrenamiento simplificado con pocos pasos de un muestreo de Gibbs, el llamado Aprendizaje por Divergencia de Contraste (CDL, "Contrastive Divergence Learning") [Carreira-Perpiñán and Hinton, 2005]; incluso con un paso resulta efectiva para entrenar capa a capa un DBN. La publicación en "Science" de estos resultados [Hinton and Salakhutdinov, 2006] proporcionó un fortísimo impulso a la I+D en DNNs.

El lector interesado sufrirá ante la ausencia de formulación para presentar las DBNs: lo cierto es que nos llevaría demasiado lejos del enfoque de esta Tesis. A cambio, mencionaremos que el precitado artículo tutorial [Bengio, 2009] contiene una presentación bastante detallada de todo lo relativo al ámbito de las DBNs.

Volvemos ahora a la utilización de los AEs.

La Auto-Codificación es un tema bien conocido en tratamiento lineal de datos y señales: hablamos de Componentes Principales (e incluso de Descomposición en Valores Singulares). Lo que se pretende al construir AE Profundos (DAEs, "Deep Auto-Encoders") es extraer progresivamente rasgos de niveles crecientes que sean

adecuados para una más sencilla resolución del problema bajo análisis; es decir, se espera un efecto de "desenmarañamiento" de las muestras en el espacio original –que, cualitativamente, parece que se produce [Bengio et al., 2013].

Los intentos de diseñar AEs no lineales –y, concretamente, DAEs– encontraron dificultades desde un primer momento: en [Bourlard and Kamp, 1988] se reconoce que un MLP con una capa oculta proporciona una representación que aproxima la de Componentes Principales si la dimensión de la entrada se mantiene; y ello, a pesar de que incluye activaciones no lineales. Solamente restan, pues, dos vías para salir del atolladero: aplicar transformaciones contractivas o expansivas, o bien forzar soluciones dispersas.

Las transformaciones contractivas fueron las primeras en explorarse, dado que las expansivas, utilizadas directamente tienden a aproximar la transformación identidad. Un interesante estudio sobre esta primera clase de transformaciones es [Rifai et al., 2011]. Implican un riesgo: la destrucción de información valiosa por efecto de la compresión. En [Ranzato et al., 2008] se formaliza una interesante posibilidad de dispersión (si bien para DBM). Pero los DAEs que han conseguido mayor éxito son los expansivos con reducción de ruido [Vincent et al., 2008], [Erhan et al., 2010]. Como quiera que, por razones que se exponen más adelante, hemos decidido emplear este tipo de DNNs en nuestros experimentos, expondremos ahora brevemente sus principios.

La propuesta de los DAE expansivos con reducción de ruido se asienta en procedimientos tiempo ha conocidos en ML [Holmstrom and Koistinen, 1992], [Grandvalet et al., 1997], que hoy, bajo el nombre colectivo de Aprendizaje Ruidoso (NL, "Noisy Learning"), se han ampliado mucho y concitan gran atención. En lo esencial, consisten en utilizar un modelo generativo para producir muestras adicionales de cada clase, y añadir éstas, con su correspondiente etiqueta, a las disponibles para el proceso de entrenamiento.

Como se representa esquemáticamente en la Figura 1.5, construiremos este tipo de DAEs mediante un entrenamiento capa a capa: se aplica a la entrada una muestra

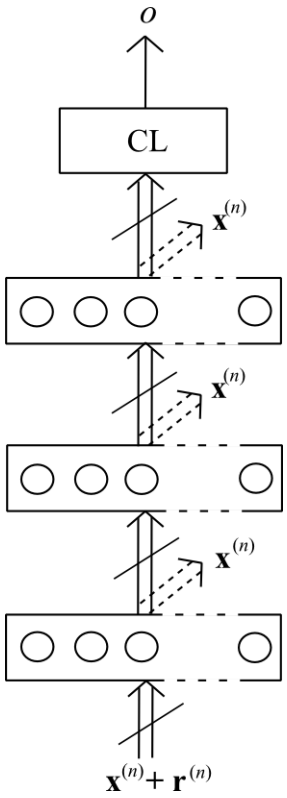

Figura 1.5: Clasificador DAE expansivo con reducción de ruido. Las capas se entrenan una a una con ejemplos que incluyen muestras ruidosas $\mathbf{x}^{(n)} + \mathbf{r}^{(n)}$, pero requiriendo salidas limpias, $\mathbf{x}^{(n)}$. El clasificador final se incluye para obtener la etiqueta $t^{(n)}$, realizándose para ello un refinado, o entrenamiento global.

ruidosa $\mathbf{x}^{(n)} + \mathbf{r}^{(n)}$ y se demanda a la salida la correspondiente muestra limpia $\mathbf{x}^{(n)}$; congelando cada capa después de ser entrenada y procediendo con la siguiente. La máquina se corona con un clasificador más o menos convencional, procediéndose a un refinado o entrenamiento global como etapa última.

Las DNNs que recurren a un primer entrenamiento no supervisado y después aplican supervisión –señaladamente las DBNs y las basadas en DAEs que acabamos de exponer– reciben la apelación de representacionales, y los algoritmos se incluyen en el Aprendizaje Representacional (RL, "Representation Learning"). El ya men-

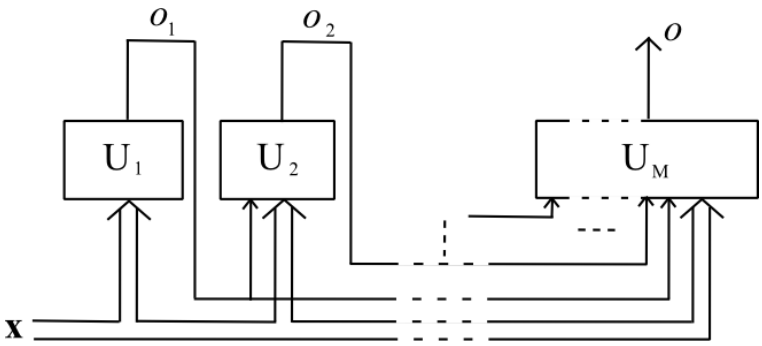

Figura 1.6: DSN: se entrenan secuencialmente unidades cuyas salidas se aplican como entrada a todas las unidades siguientes.

cionado artículo tutorial [Bengio, 2009] expone magistralmente esta familia RL, y destaca tanto su función de desenmarañamiento como el hecho de que los rasgos que se obtienen en las capas ocultas son importantes caracterizaciones de las observaciones con progresivos niveles de abstracción. Todo ello es muy importante: primero, porque facilita grandemente el diseño de DNNs eficaces para problemas cualesquiera; segundo, porque esos rasgos ocultos pueden ser relevantes para el análisis y la comprensión de las dificultades del problema que se pretende resolver; por ello, resultan útiles para mejorar funciones como el etiquetado automático, por ejemplo [Najafabadi et al., 2015]. Estas razones, que recordaremos en el Capítulo 2, nos han llevado a seleccionar las DNNs basadas en DAEs –y concretamente en DAEs expansivos reductores de ruido– para nuestros experimentos.

Hasta ahora sólo hemos hecho referencia a una familia de DNNs directamente entrenables, las DCNNs, que no presentan mayores dificultades dada su compartición de pesos. Otra familia de DCNNs directamente entrenable es la formada por las Redes Apiladas Profundas (DSNs, "Deep Stacking Networks") [Deng and Yu, 2011], [Deng et al., 2012]. En este caso, el entrenamiento directo es posible gracias a que la construcción es secuencial.

Como se puede ver en la Figura 1.6, para construir una DSN se entrenan secuencialmente unidades cuyas salidas se colocan como partes de la entrada en todas las

unidades subsiguientes. De este modo, esos entrenamientos resultan asequibles –y además, cada unidad puede dedicar más atención a aquellos ejemplos que continúan siendo problemáticos–, y, al tiempo, se gana en profundidad: se puede decir que cada unidad constituye una capa adicional. Se cree que, con esa profundización, se obtiene ventaja expresiva.

Recientemente, estudios de las limitaciones del algoritmo de BP y variaciones del mismo han posibilitado el entrenamiento directo del DNNs tipo MLPs: por ejemplo, dedicando el debido cuidado a la inicialización de los pesos en las diversas capas [Glorot and Bengio, 2010] o evitando el uso directo del Hessiano en el entrenamiento [Martens, 2010]. No podemos dedicar aquí espacio a estos avances, ni a otras concepciones del DL, como son el uso de NNs Recursivas (RNN, "Recurrent Neural Networks") [Martens and Sutskever, 2011] o de elementos latentes [Salakhutdinov et al., 2011], [Girshick et al., 2011]. En su lugar, proponemos que se acuda a los tutoriales [Bengio, 2009], [Deng and Yu, 2013] y [Schmidhuber, 2014].

Las aplicaciones con éxito del DL son numerosísimas y se extienden sobre muy diversos ámbitos: además de las ya entrevistas de reconocimiento de objetos e imágenes y de habla, en lenguaje natural, reconocimiento de rostros, detección de emociones, análisis de sentimientos, recuperación de información, ..., y otros en Audio y Sonido, Biología, Bioinformática, Educación, Energía, Fabricación, Finanzas, Robótica, Salud, Transporte ... Obviamente, incluso un selectísimo resumen de referencias desbordaría los límites de una Tesis Doctoral: en [DeepLearningUniversity, 2014] se puede saciar casi cualquier curiosidad concreta.

Concluiremos este apartado con algunas líneas dedicadas a la capacidad expresiva de las DNNs: aunque puede intuirse que tiene límites más amplios que la de los MLPs "llanos" –de los que ya se ha indicado que la tienen potencialmente infinita–, análisis dedicados a ello arrojan luz sobre sus específicos puntos fuertes. Naturalmente, el foco ha de ponerse en el caso de las DNNs "estrechas": son aproximadores universales [Sutskever and Hinton, 2008]. Mejoras sobre el estudio anterior aparecen en [Montufar and Ay, 2011]. Finalmente, [Szymanski and McCane, 2014] resaltan que

las DNNs son efectivas para codificar la periodicidad: nótese que este hecho explica su espectacular ventaja en algunas aplicaciones. [Håstad and Goldmann, 1991] y [Delalleau and Bengio, 2011] discuten cómo la adición de capas facilita el establecimiento de correspondencias entrada-salida.

1.5. Orientación, objetivos y contenido de la Tesis

En esta Tesis se propone y lleva a cabo un primer estudio sistemático de las posibilidades de obtener (aún) mejores prestaciones en LMs para clasificación mediante oportunas combinaciones de profundidad y diversificación.

La frase que antecede no implica en modo alguno que no existan precedentes de esta orientación. Ya hemos citado trabajos del equipo de Jürgen Schmidhuber en IDSIA (Univ. Lugano) en los que se diversifican DCNNs mediante recortes y distorsiones de imágenes [Cireşan et al., 2011], [Cireşan et al., 2012a]. Sobre esos esquemas se han aplicado agregaciones entrenables –pero sencillas– para mejorar las prestaciones [Frazão and Alexandre, 2014]. El flujo óptico obtenido de fotogramas consecutivos de vídeo es utilizado en [Nina et al., 2014] para construir conjuntos CNN. De manera similar, los múltiples estados trifonémicos se utilizan como fuente de diversidad para la tarea de reconocimiento del habla en [Zhao et al., 2014], utilizando aprendices que incluyen unidades del modelo oculto de Markov. Los autores de [Shao et al., 2014] utilizan la diversidad espectral [Xia et al., 2010] para entrenar algunas DNNs cuyas salidas se agregan, mientras que para el entrenamiento parcial de los aprendices se utilizan muestras distorsionadas en la diversificación [Simpson, 2015]. Ya hemos expuesto nuestra opinión de que las DSNs [Deng and Yu, 2011], [Deng et al., 2012] pueden considerarse resultado de la diversificación que supone añadir a las variables observadas las salidas de las unidades anteriores en la entrada de cada nuevo aprendiz. Por el contrario, y oponiéndonos a algunos autores, consideramos que "Drop-Out" [Srivastava et al., 2014] y "Drop-Connect" [Wan et al., 2013] no son tanto mecanismos de diversificación como de regularización: aunque admitimos que podrían

utilizarse –de distinta manera– para lo primero, y aunque la estructura CNN con "Drop-Connect" y diversificación final de [Wan et al., 2013] ostenta el récord de prestaciones para el clásico problema de reconocimiento de dígitos manuscritos MNIST [LeCun et al., 1989], [Vincent et al., 2010], que se tratará experimentalmente en esta Tesis.

Un primer estudio de diversificación de las CNNs aparece en [Lee et al., 2015], donde se destaca que el carácter inestable de estas DNNs permiten diversificarlas simplemente a través de diferentes inicializaciones, siendo la única excepción GoogLeNet [Szegedy et al., 2014], que lo combina con "Bagging". Esta inestabilidad crea mucha dificultad para aplicar técnicas de diversificación convencional, y así se comprueba en los resultados de [Lee et al., 2015].

Además, con sistemático no queremos decir exhaustivo: estudiar todas las posibles combinaciones de diversidad y profundidad –o una parte significativa de ellas– excede lo asequible para un trabajo de tesis; y aún para el trabajo a corto o medio plazo de un equipo de I+D. Pero sí examinaremos cómo aplicar métodos de diversificación convenientemente seleccionados a un tipo de DNNs también elegido con criterio: será éste las DNNs basadas en DAEs –y concretamente en los expansivos con reducción de ruido–, y las razones son tanto las anteriormente citadas ventajas para los DNNs por RL –amplio espectro de potenciales aplicaciones y provisión de información valiosa en los nodos ocultos– como por la inclusión del NL en el tipo de DAEs seleccionado, ya que este aprendizaje se encuentra en intensa evolución y son de esperar valiosos progresos. En cuanto a la diversificación, además de la binarización, emplearemos "Bagging" y "Switching": son sencillos –aunque suele pagarse con muchos aprendices– y por tanto su éxito podrá tomarse como señal para examinar otras técnicas; y no exigen unidades débiles. Como aproximación al "Boosting" fácilmente implantable se estudiará la aplicación del pre-énfasis (véase el Capítulo 4 para una presentación de su concepto, historia e interés), diseñando para esa implantación una forma analítica lo suficientemente general como para que puedan apreciarse el papel de las diversas componentes que se han aplicado. Procederemos paso a paso,

primero con las combinaciones inmediatas de diversidad y profundidad, después con el pre-énfasis, y finalmente con pre-énfasis y diversidad sobre profundidad.

Nótese que, al contrario de lo que ocurre con casi todos los trabajos que hemos considerado precedentes de éste, no se va a hacer uso de la opción de proponer modos de diversificar específicos para los problemas bajo análisis –que también seleccionaremos cuidadosamente, como se verá en el Capítulo 2–, porque ello comprometería la generalización de los resultados que se obtuviesen.

Así, pues, se cifran los objetivos de la Tesis en determinar si la binarización, la diversificación mediante "Bagging" y "Switching", y el pre-enfatizado, aplicados de diversos modos, son eficaces para mejorar las prestaciones de las DNNs basadas en DAEs expansivos y reductores de ruido: estudio de por sí relevante, pero que, además, dados los ingredientes que se manejan, permitirá extraer conclusiones con un esperable alto grado de generalidad.

De acuerdo con lo dicho se estructura el contenido de esta Tesis.

En el Capítulo 2, "Elementos básicos", se revisarán brevemente:

- los DAEs expansivos y reductores de ruido específicos a emplear en los experimentos;
- los métodos de binarización elegidos;
- las diversificaciones mediante "Bagging" y "Switching";
- las formas de pre-énfasis a aplicar y la razón para elegirlas (aunque el detalle se dejará para el Capítulo 4);
- las bases de datos con las que se trabajará en los experimentos

de forma razonada y concisa.

En el Capítulo 3, "Aprendizaje Diverso y Profundo (D2L)", ("Diverse and Deep Learning"), se presentarán los esquemas básicos de combinaciones D2L, se realizarán los correspondientes diseños para los problemas bajo análisis, se describirán

los experimentos y sus resultados, y se discutirán éstos y extraerán unas primeras conclusiones.

"Una posibilidad adicional: Pre-Énfasis" es el título del Capítulo 4. Se comenzará por justificar la utilidad de los mecanismos de pre-énfasis, al tiempo que se revisa su ya larga historia en el campo de ML. Inmediatamente, se presentarán y justificarán las formas analíticas elegidas para aplicar pre-énfasis, resaltando su generalidad y flexibilidad y la conveniencia de éstas. Tras ello, y como en el capítulo anterior, vendrán experimentos, resultados, discusión de los mismos, y primeras conclusiones.

El Capítulo 5 es aquél en que se combina lo estudiado en los dos anteriores en las formas que sugieren sus resultados como más apropiadas: un pre-énfasis global de las mejores máquinas D2L del Capítulo 2, y pre-énfasis binarios separados para dicotomías ECOC, seguidas de DAEs y diversificación "Switching". Además, y para explorar los límites de los diseños que estamos analizando, se aplicará al mejor de los obtenidos hasta aquí un método particular de lo que denominaremos en esta Tesis Aumento de Ejemplos (de "Data Augmentation")([Tabik et al., 2017],[Nielsen, 2015]): la conocida como Distorsión Elástica ("Elastic Distortion") ([O'Neill, 2006]). Con ello obtendremos un nuevo récord en el clásico problema de clasificación de dígitos binarios MNIST ([LeCun et al., 1998], [Vincent et al., 2010]). Aunque el propósito de de nuestro estudio no es conseguir prestaciones récord, sino evidenciar que combinar diversificación y profundidad produce relevantes mejoras, se trata de un apoyo adicional a nuestra suposición. De nuevo se exponen experimentos y sus resultados, que se discuten acto seguido, y se cierra el capítulo con las conclusiones pertinentes al mismo.

"Conclusiones y oportunidades" sirve de título al capítulo final. Se resumen en él las conclusiones concretas obtenidas, así como se estiman las posibilidades de generalizarlas, y se sugieren las que se consideran las vías más atractivas para comprobar si esas estimaciones son ciertas. Además, se proponen otras más amplias avenidas para extender estos trabajos, tanto en tipos de problemas como en LM complementarias, y también en ingredientes D2L; y además se resulta el interés de combinar todo ello

con Aprendizaje Dinámico ("Dynamic Learning") y Aprendizaje Distribuido ("Distributed Learning") para avanzar al muy necesario D4L, el verdadero y final Gran Aprendizaje ("Big Learning").

Se citan también en este último capítulo las publicaciones científico-técnicas realizadas a partir de estos trabajos.

Capítulo 2

Elementos Básicos

Tal y como se ha adelantado en el Capítulo 1, se expondrán aquí las DNNs que se manejarán en los trabajos de la Tesis, los métodos de binarización y diversificación clásica que se aplicarán, las formas de preénfasis a utilizar y, por último, las bases de datos sobre las que se llevarán a cabo las tareas experimentales. Así lo hacemos para evitar repeticiones y conseguir mayor claridad en los capítulos que tratan de las propuestas concretas propias de la Tesis y su evaluación.

2.1. DNNs: clasificadores SDAE3

Por las razones ya expuestas (generalidad, carácter representacional y potenciales ventajas implicadas en la interpretación y aplicaciones, posibilidades de expandir y mejorar los procesos de NL incluidos en ellas, claridad en la atribución de mejoras a sus causas reales), hemos elegido las DNNs tipo DAE expansivo y ruidoso para nuestros estudios y experimentos. Concretamente, las conocidas como clasificadores SDAE3 ("Stacked Denoising Autoencoders - 3 layers") [Vincent et al., 2008], [Vincent et al., 2010], un diseño que ha acreditado excelentes prestaciones en varias aplicaciones relevantes. Utilizaremos aquí una versión propia.

Los parámetros que definen la arquitectura concreta del SDAE3 a emplear en

nuestros experimentos (correspondientes a las bases de datos que se han seleccionado: véase la sección 2.5) son las siguientes [Vincent et al., 2010]:

- número de capas de auto-codificación: 3;
- número de activaciones por capa: 1000 (dichas activaciones son de la forma tanh);
- aprendizaje por gradiente con pasos 0.01 para la primera capa y 0.02 para las dos siguientes;
- el número de épocas de entrenamiento se limita a 40 (que resulta suficiente para la convergencia en todos los casos);
- la arquitectura anterior se corona con un clasificador MLP con una capa oculta de 1000 unidades (también tanh) y salida soft-max (Potts) para 10 clases;
- paso de entrenamiento para el refinado: 0.02.

La función de coste a minimizar es la entropía simétrica muestral

$$C(\{\mathbf{t}^{(n)}, \mathbf{o}^{(n)}\}) = -\sum [\mathbf{t}^{(n)T} \ln \mathbf{o}^{(n)} + (\mathbf{1} - \mathbf{t}^{(n)})^T \ln (\mathbf{1} - \mathbf{o}^{(n)})] \tag{2.1}$$

Dejamos intencionadamente libre el nivel de ruido aditivo a aplicar a los ejemplos de entrenamiento. Hay para ello una razón fundamental: en los experimentos mostrados en [Vincent et al., 2010] para las mismas bases de datos que consideraremos aquí, ese nivel de ruido es el 10 % del de señal en todos los casos, y se obtienen muy buenos resultados: pero hay indicios de que la forma del ruido seleccionado es "ad hoc" –dependiente del problema–, e insistimos en que nuestro estudio quiere que las ventajas de las propuestas que en él se hacen no puedan ponerse en tela de juicio por restringir –o especializar– los diseños que se usen; además, no se desea ocluir ninguna posibilidad de incorporar mejoras en la mecánica del NL.

El "software" de partida se ha tomado de [HintonWeb, 2015].

2.2. Métodos de binarización

Consideraremos dos de las posibles: el directo OvO, y el ECOC para 10 clases propuesto en [Dietterich and Bakiri, 1995], específicamente concebido para reconocimiento de dígitos manuscritos: tiene en cuenta los rasgos definitorios de tales dígitos[1].

Seleccionando OvO eludimos el problema de desequilibrio que produce OvR; desequilibrio que puede ser intenso para bases de datos de 10 clases equilibradas entre sí como las que nos servirán para los experimentos. Diez clases son todavía suficientemente pocas para que no se produzca la explosión de unidades que puede provocar OvO para un número de clases C relativamente alto: ese número de unidades es $C(C-1)/2$, y con $C = 10$ resulta ser 45: nada alarmante en procesos de diversificación. Podremos así comprobar si la acreditada ventaja de los métodos ECOC (razonablemente elegidos) cuando se trabaja con máquinas "llanas" continúan manifestándose en caso de emplear DNNs.

En cuanto al antes mencionado código ECOC, la Tabla 2.1 lo presenta completo. Ya se ha indicado que su concepción tiene en cuenta los rasgos definitorios de los dígitos manuscritos: una explicación de cómo se incluye en la referencia que lo presenta, [Dietterich and Bakiri, 1995]. Resaltaremos, eso sí, que la mínima distancia de Hamming entre las palabras-código que representan a las clases es 7; de ese modo, se corrigen en la agregación hasta 3 errores en las soluciones de las dicotomías, una potencia correctora muy apreciable para un código tan compacto.

Puede observarse que entre los problemas binarios definidos por este ECOC los hay desequilibrados (supuestas clases equilibradas): cuatro de modo no preocupante (P_3, P_{11}, P_{12} y P_{13}), ya que el desequilibrio es 70 % - 30 %; pero P_{14} muestra un 90 % - 10 %. Sería posible incluir mecanismos de compensación de desequilibrio, o al menos hacerlo en aquellos casos en que se comprobase que el desequilibrio está produciendo una merma en las prestaciones; pero esa actuación no estaría directamente relacionada con el propósito y los objetivos de la Tesis, cifrados en explorar la posible ventaja

[1] Lo que no quiere decir que se haya definido para las bases de datos que vamos a considerar.

	Problema														
Clase	P_0	P_1	P_2	P_3	P_4	P_5	P_6	P_7	P_8	P_9	P_{10}	P_{11}	P_{12}	P_{13}	P_{14}
C_0	1	1	0	0	0	0	1	0	1	0	0	1	1	0	1
C_1	0	0	1	1	1	1	0	1	0	1	1	0	0	1	0
C_2	1	0	0	1	0	0	0	1	1	1	1	0	1	0	1
C_3	0	0	1	1	0	1	1	1	0	0	0	0	1	0	1
C_4	1	1	1	0	1	0	1	1	0	0	1	0	0	0	1
C_5	0	1	0	0	1	1	0	1	1	1	0	0	0	0	1
C_6	1	0	1	1	1	0	0	0	0	1	0	1	0	0	1
C_7	0	0	0	1	1	1	1	0	1	0	1	1	0	0	1
C_8	1	1	0	1	0	1	1	0	0	1	0	0	0	1	1
C_9	0	1	1	1	0	0	0	0	1	0	1	0	0	1	1

Tabla 2.1: El ECOC de [Dietterich and Bakiri, 1995] para clasificación de dígitos manuscritos. C_0 a C_9 son las clases, P_0 a P_{14} las dicotomías. El código más cercano en distancia de Hamming al vector de resultados para las dicotomías indica la clase de la muestra bajo análisis

de combinar diversidad y profundidad. Consecuentemente, no nos detendremos en ello.

No cerraremos este apartado sin unas observaciones generales. La primera, que a día de hoy es difícil sostener, como se hizo a mediados de los 1990, que el empleo de códigos exhaustivos para siete clases o más impondría una carga computacional –por el número de dicotomías implicado– inaceptable: la potencia de cómputo disponible ha crecido órdenes de magnitud, y además el procesamiento paralelo resulta muy accesible. La segunda, que tampoco resultaría complicado extender el ECOC de la Tabla 2.1 para incrementar la distancia mínima de Hamming (incluso sin recurrir a técnicas ECCs muy elaboradas), y, con ello, mejorar resultados. Si no se ha llevado a cabo ninguna de las dos cosas anteriores es porque requerirían un esfuerzo que no

contribuiría significativamente a los ya repetidos propósitos y objetivos de la Tesis. Y una tercera observación: no resulta insensato sostener que, incluso en problemas con (relativamente) elevados números de clases (digamos más de 20, que supondría ECOCs tradicionales con códigos de clase muy largos), podrían diseñarse –en función del problema– agrupaciones de clases –para proceder secuencialmente (primero, decidir sobre los grupos; después, sobre las clases)– con suficiente tino como para que la degradación que toda simplificación estadística de este tipo provoca fuese tolerable, permitiendo obtener ventaja respecto a un ataque multiclase.

2.3. Diversificaciones vía ejemplos

"Bagging" [Breiman, 1996] y "Switching" [Breiman, 2000] serán los dos mecanismos de los que haremos uso. Son extremadamente sencillos, de entrenamiento sin dificultades, y no requieren aprendices débiles, sino simplemente inestables: como las DNNs –y los SDAE3, en particular– lo son por su estructura, no queda cerrada la posibilidad de aplicarlos directamente a ellas; la otra es, como se discutirá más adelante, emplear la diversificación sólo en los clasificadores finales, en el caso de arquitecturas representacionales. Bien es cierto que ambos métodos requieren un muy notable número de aprendices –véase, por ejemplo, el completo estudio [Martínez-Muñoz et al., 2008]–, pero insistimos en que, a día de hoy, dicho inconveniente es soportable. Disponen además ambos mecanismos de parámetros de diseño –el tamaño del remuestreo y la tasa de conmutación de etiquetas– que se pueden modificar de modo directo y que, junto con el número de aprendices, N, a incluir en los conjuntos, brindan suficiente flexibilidad para que un proceso de validación seleccione modelos efectivos.

Los conjuntos construidos mediante "Bagging" entrenan cada aprendiz con una colección de ejemplos obtenidos mediante un remuestreo "Bootstrap" –es decir, selección al azar con reemplazamiento–. Las ventajas del "Booststrap" desde el punto de vista estadístico –reducción de varianza, en pocas palabras– son bien conocidas y

no entretendremos al lector reiterándolos aquí. Sí ha de destacarse que el remuestreo puede realizarse para dar lugar a un número de ejemplos menor, igual o mayor que el número original disponible (en el conjunto de entrenamiento), y que la buena elección de ese número –el tamaño del remuestreo– es dependiente del problema. Por eso en la Tesis se explorarán sistemáticamente esos tamaños de remuestreo, que se indicarán con la letra B, seleccionando el más apropiado en cada caso mediante validación. Concretamente, se considerarán para B los valores 60, 80, 100, 120 y 140 %.

En el caso del "Switching", el parámetro intrínseco seleccionable es la tasa de cambio de etiqueta (de volteo, en problemas binarios), que indicaremos por S. Naturalmente, esa tasa ha de mantenerse por debajo del 50 %, ya que, caso contrario, se estaría "invirtiendo" el problema. Debe resaltarse que esta inyección de ruido (discreto) en las etiquetas tiene efectos cualitativamente iguales al "Bootstrap": en el contexto que nos ocupa, se producen diferencias en las fronteras de separación entre clases que trazan los diversos aprendices, y una forma adecuada de promediado (p. ej., aritmético directo) o análoga (p. ej., votación mayoritaria; que es la que emplearemos aquí) tiende a reducir la dispersión y a proporcionar mejores resultados que los ofrecidos por una LM monolítica.

En el caso del "Switching", se emplearán para S los valores 10, 20, 30 y 40 %.

En cuanto al número de aprendices, N, se barrerán 25, 50 y 100, inicialmente; después, 25, 51, 101, 121, para evitar empates.

2.4. Pre-énfasis y formas seleccionadas para aplicarlo

Aunque por razón de la debida coherencia expositiva demoraremos hasta el Capítulo 4 una presentación completa de las técnicas de pre-énfasis, su origen y evolución y los motivos de la selección de las formas analíticas que en esta Tesis se emplean, a efectos de completar la presentación del escenario de los trabajos que se hace en el presente capítulo incluiremos ahora un breve resumen de principios y explicitaremos las formulaciones seleccionadas, con un breve comentario.

En todo proceso de clasificación se desea cometer tan pocos errores como sea posible –sin caer en sobreajustar las fronteras a los ejemplos de entrenamento, desde luego– y, por tanto, convendría prestar atención a los ejemplos que resulten mal clasificados mediante algún mecanismo razonable. Este es el principio que alumbró la aparición del "Boosting"; que, como ya hemos justificado, consiste en una gran familia con cantidad de algoritmos que no podemos aplicar con toda generalidad cuando tratamos con DNNs, que son LMs fuertes. Pero mucho antes ya se propusieron métodos de diseño que concedían particular atención a los ejemplos "difíciles" –véase la breve reseña histórica que hemos incluido en el Capítulo 4–. Pronto se observó que no sólo las muestras difíciles de clasificar merecían atención, sino también aquellas que se encontraban en las inmediaciones de fronteras de decisión razonables, fueron éstas resultado de un proceso previo o simplemente previstas. De las formas más elementales de conceder atención –selección de muestras– se puso pronto a formas graduales: ponderación de muestras en el coste muestral a minimizar según su importancia para una clasificación, es decir, tomando en cuenta su proximidad a una frontera de referencia o su error de clasificación, y también combinaciones de ambas características. La experiencia demostró que la importancia relativa de cada una de ellas depende del problema que se esté resolviendo, y así nacieron los más elaborados métodos de pre-énfasis, que combinaban de forma ajustable la influencia de ambos aspectos en la ponderación de los ejemplos, típicamente en virtud de los resultados obtenidos de un clasificador auxiliar debidamente fiable.

Desafortunada y sorprendentemente, la inmensa mayoría de esas formas de énfasis –que denominamos intencionadamente de este modo general para no excluir versiones utilizadas en "Bosting"– carecían de un término que ejerciese la función de moderar o controlar la intensidad de dicho énfasis; es decir, que evitase que todas las muestras fuesen ponderadas tan sólo de acuerdo con combinaciones de medidas de error y de proximidad a la frontera. Y es obvio que una ponderación no directamente proporcional a esa combinación puede resultar mejor en algunos problemas.

De acuerdo con esa razón, proponemos en esta Tesis formas analíticas para el

pre-énfasis que incluyen no sólo (una combinación convexa de) medidas de error y proximidad, sino que combinamos convexamente lo anterior con un término constante (recurrir a combinaciones convexas tiene como propósito minimizar el número de parámetros a explorar).

Bajo dicho principio, cabe proponer muchas diferentes medidas de error y de proximidad a la frontera. No nos esforzaremos por elegirlos de acuerdo con los problemas que vamos a abordar en nuestros experimentos, porque, una vez más, preferimos proceder de modo que no se comprometa el valor general de las conclusiones que se deriven de los resultados: recurriremos a medidas simples y directas, con sentido en cualesquiera situaciones que pudiesen aparecer. Pero, antes de presentar (las dos versiones de) la forma elegida, debemos advertir que otras medidas seleccionadas según el problema pueden –y deben– conducir a mejores prestaciones; y muy en particular, debemos señalar que el empleo de medidas paramétricas, acompañado de la necesaria exploración y validación de los valores de los correspondientes parámetros, es una línea de trabajo de no poca importancia, recomendable si se puede justificar el incremento de coste computacional implicado en la fase de diseño.

La forma analítica que aquí elegimos de la ponderación p en pre-énfasis para problemas binarios es la siguiente:

$$p(\mathbf{x}^{(n)}) = \alpha + (1-\alpha)[\beta(t^{(n)} - o_a^{(n)})^2 + (1-\beta)(1 - o_a^{(n)2})] \tag{2.2}$$

donde $p(\mathbf{x}^{(n)})$ es el factor de ponderación a aplicar al ejemplo de entrenamiento $\mathbf{x}^{(n)}$, $0 \leq \alpha, \beta \leq 1$ los parámetros de combinación convexa, $t^{(n)}$ la etiqueta de $\mathbf{x}^{(n)}$, y $o_a^{(n)}$ la salida de un clasificador auxiliar, o guía, cuando se aplica $\mathbf{x}^{(n)}$ a su entrada.

Nótese que α representa el término de moderación del énfasis: es común a todas las muestras. Evidentemente, $\alpha = 0$ da lugar a un énfasis combinado sólo en función de las medidas de error, $(t^{(n)} - o_a^{(n)})^2$, y de proximidad a la frontera, $1 - o_a^{(n)2}$. Comprobaremos en los experimentos que incluir α es más que recomendable: los diseños validados que iremos obteniendo adoptan valores de α decididamente no

nulos, y los que se obtienen con la restricción $\alpha = 0$ son netamente inferiores. Por otro lado, anular β limita el énfasis variable al efecto de la medida de proximidad a la frontera, y $\beta = 1$ lo limita al de la medida de error: también veremos que ambas opciones limitan grandemente la calidad de los diseños obtenidos.

Para problemas multiclase se propone:

$$p(\mathbf{x}^{(n)}) = \alpha + (1-\alpha)\{\beta(1-o_{ac}^{(n)})^2 + (1-\beta)[1-|o_{ac}^{(n)} - o_{ac'}^{(n)}|]\} \tag{2.3}$$

siendo $o_{ac}^{(n)}$ la salida del clasificador auxiliar para la clase correcta, c, y $o_{ac'}^{(n)}$ la salida de esa guía más próxima a $o_{ac}^{(n)}$ de entre los que corresponden a clases incorrectas, c'. Recuérdese que $0 \leq o_{ac}^{(n)}, o_{ac'}^{(n)} \leq 1$. Nótese la fuerte analogía que existe entre (2.3) y (2.2); la única diferencia reseñable es la expresión de la proximidad a la frontera: en problemas multiclase, conviene que sea a la frontera más cercana y, aunque hay opciones distintas a la elegida –p. ej., incluir $\text{máx}_{c' \neq c}\{o_{ac'}^{(n)}\}$ en lugar de la salida más cercana–, nuestra experiencia avala que (2.3) es una buena elección.

No creemos preciso repetir la discusión para α, β, que hemos expuesto en el caso binario.

Obsérvese que se emplea el error cuadrático y la proximidad lineal: no es conceptual ni cualitativamente relevante, aunque otras normas pueden permitir mejores resultados en algunos casos.

Queremos subrayar, antes de pasar al apartado siguiente, que el pre-énfasis supone una carga computacional adicional a la precisa para el diseño directo sólo en entrenamiento, debido a la necesidad de una máquina auxiliar y al proceso de exploración y selección por validación de α y β –como se ve, un incremento cómodamente soportable–, pero no en operación: la LM diseñada para pre-énfasis no se distingue de una diseñada directamente salvo por los valores de los parámetros estructurales. Conforme a ello, toda mejora apreciable de prestaciones será un beneficio barato.

2.5. Aumento de Ejemplos

Los métodos de aumento de datos pre-procesan ejemplos de entrenamiento para crear nuevas muestras que tengan características similares a las de los originales. Permiten aumentar el número de muestras de entrenamiento añadiendo versiones aumentadas con etiquetas correspondientes a las muestras originales de las que se han obtenido. Si son cuidadosamente seleccionados y diseñados, estos métodos son eficaces para aumentar el rendimiento de las máquinas de clasificación.

El aumento de datos se ha utilizado a lo largo de dos décadas [LeCun et al., 1998], y sus formas han evolucionado significativamente. El segundo mejor error de clasificación para MNIST, 0.23 %, se obtuvo por medio de un conjunto CNN con aumento de datos como fuente de diversificación [Cireşan et al., 2012a]. Existen varios mecanismos de aumento de datos que han demostrado eficacia en la mejora de prestaciones de los clasificadores de imágenes, tales como traslaciones aleatorias, rotaciones aleatorias, centrado y deformaciones elásticas. Para una revisión concisa, recomendamos al lector [Tabik et al., 2017],[Nielsen, 2015]. Dada la ventaja que proporciona el aumento de datos y que, con el pre-enfatizado no hay un aumento de la carga computacional de la operación, también incluimos una forma de la misma, la versión más frecuente de deformación elástica [O'Neill, 2006]. Este será el último paso de nuestros experimentos, después de combinar pre-énfasis y diversidad.

Los aspectos básicos de la deformación elástica que emplearemos son los siguientes. En primer lugar, hay una traslación de píxeles, cuyos valores horizontales y verticales se obtienen multiplicando los elementos de dos matrices del tamaño de la imagen con pequeños valores aleatorios por un factor de escala Δ que debe seleccionarse apropiadamente. Las traslaciones se limitan hasta los bordes de la imagen, y los valores que llegan a la misma posición se promedian. Después de ello, se realiza un filtrado gaussiano normalizado con el objetivo de suavizar los resultados. El parámetro σ del filtro también debe seleccionarse con cuidado. Nuestra selección de Δ, σ, se discutirá en la sección dedicada a los experimentos.

2.6. Bases de datos para los experimentos

Entre las bases de datos con las que experimentar hay que incluir una multiclase que reúna condiciones importantes:

- representar un problema real de reconocida relevancia y no pequeño grado de dificultad, que haya servido repetidamente como banco de pruebas y aún siga sirviendo para ello;
- constar de un número de clases suficiente para que se puedan apreciar claramente las (posibles) diferencias entre diseños softmax y binarizados;
- ser abordable mediante DNNs "ad hoc" –DCNNs– con importante ventaja (aunque, según se ha indicado ya, dichas arquitecturas no vayan a ser incluidas en los diseños de esta Tesis);
- tener récords reconocidos con dichos diseños "ad hoc" que puedan considerarse el nivel del estado de arte científico-técnico actual, así como prestaciones públicamente conocidas con diseños DNN convencionales: en concreto, con los aquí seleccionados (que, en realidad, también lo han sido por haberse utilizado para el problema);
- no dar lugar, por sus dimensiones y volumen, a excesos en la carga computacional requerida por la combinación de diversidad y profundidad, que excederá sensiblemente la que precisa cada una de ellas por separado;
- en lo posible, contener explícitamente no sólo un conjunto de test, sino uno de validación, con objeto también de reducir esfuerzo computacional; en este caso, el inducido por los necesarios procesos de exploración y selección por validación de los parámetros de diseño no entrenables.

La ya clásica base de datos MNIST, de dígitos manuscritos, presentada en [LeCun et al., 1989], [LeCun et al., 1990], cumple con todas las condiciones que se

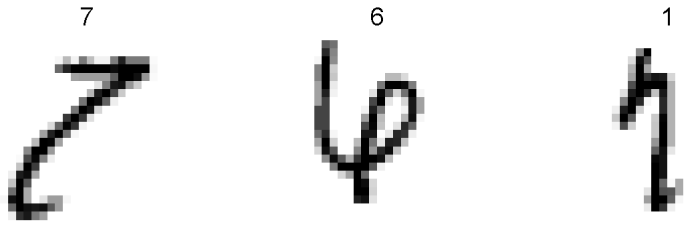

Figura 2.1: Tres dígitos de test de la base de datos MNIST y sus correspondientes etiquetas

han enumerado. El récord de prestaciones, tasa de error 0.21 %, se ha conseguido con la arquitectura DCNN con "Drop-Connect" y diversificada en la última etapa de [Wan et al., 2013]. En [Cireşan et al., 2012b], un sistema de DCNNs diversificada por Aumento de Ejemplos, la CNN Multi-Columna (MC-CNN), proporciona un 0.23 %. Estos valores se encuentran en el nivel que obtiene una persona cualificada en la tarea. La tasa de error récord obtenida con una DNN representacional se encuentra en el 0.81 % [Bengio et al., 2013]. Entre 0.81 y 0.21 % hay un margen apreciable, y una sensible reducción de la tasa de error demostraría el éxito de los métodos que se propondrán en esta Tesis. Es conveniente indicar también que la versión del SDAE3 diseñada para MNIST en [Vincent et al., 2010] ofrece una tasa de error (promedio de 10 inicializaciones $\pm$ desviación típica) de 1.28 ± 0.22 % (ruido aditivo de nivel 25 %, plausiblemente "ad hoc").

Para evitar que el lector tenga la sensación de que clasificar los dígitos manuscritos de MNIST no presenta mayores dificultades, mostramos en la Figura 2.1 tres de sus dígitos con sus correspondientes etiquetas: las dudas asaltan a cualquier observador.

Las características principales de la base MNIST son:

- número de variables de entrada: 784 (de un enrejillado 28×28 aplicado a los dígitos originales)
- variables de entrada quasi-continuas, con 256 niveles de gris

- números de ejemplos en los conjuntos de entrenamiento / validación / test: 50000 / 10000 / 10000, respectivamente.

Tras elegir el problema MNIST en primera instancia, queda claro que resultaría muy ilustrativo considerar LMs de clasificación con diferentes grados de aprendizaje, a fin de apreciar cómo influye la fortaleza o debilidad de los DAE3 básicos en las combinaciones con diversidad que se proponen y evalúan en la Tesis. Afortunadamente, ya existe una versión de MNIST, MNIST-BASIC [Vincent et al., 2010] en la que la información disponible para el diseño se reduce grandemente. Manteniendo las dos primeras características de MNIST (dimensión de la entrada y niveles de sus variables), las restantes son:

- números de ejemplos en los conjuntos de entrenamiento / validación / test: 10000 / 2000 / 50000, respectivamente.

Como es comprensible, la reducción del número de ejemplos de entrenamiento y validación debilita cualquier LM con respecto a la diseñada con MNIST. Así, SDAE3 ofrece como prestación un error de $2.84 \pm 0.15\,\%$ (10 inicializaciones y ruido aditivo al $10\,\%$, posiblemente "ad hoc"). Se aprecia que el diseño se ha debilitado con la reducción de tamaño de los conjuntos de entrenamiento y validación.

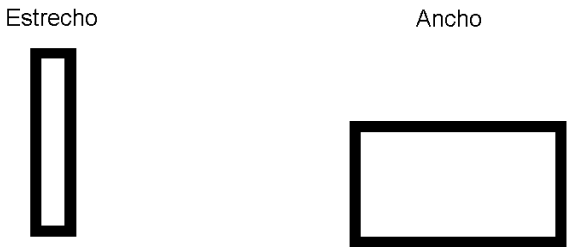

Figura 2.2: Rectángulo "estrecho" y rectángulo "ancho" de la base de datos RECTANGLES.

Para completar el abanico de casos experimentales, sería apropiado un problema binario, con la intención de apreciar resultados con y sin binarización. Se propone uno en [Vincent et al., 2010] que nos permite la ventaja de mantener la dimensión de las entradas de los anteriores, 784 (con lo que se pueden mantener las dimensiones de los correspondientes DAE3, evitando influencias externas en las comparaciones): RECTANGLES, en que se trata de distinguir rectángulos horizontales de rectángulos verticales (o estrechos de anchos) con un valor binario contrario al del fondo de la imagen (0/1, blanco/negro, son intercambiables). La Figura 2.2 muestra dos ejemplos de esta base de datos.

Dado que este problema binario es intrínsecamente más sencillo que los anteriores y puramente sintético, se establecen como características complementarios a la dimensión (784) y los niveles (2):

- números de ejemplos en los conjuntos de entrenamiento / validación / test: 10000 / 2000 / 50000

(idénticos a los de MNIST-BASIC). El SDAE3 ofrece para RECTANGLES un error de $1.99 \pm 0.12\,\%$ (10 inicializaciones y ruido aditivo del $10\,\%$).

Capítulo 3

Aprendizaje Diverso y Profundo (D2L)

Este capítulo, como se ha anticipado en la Sección 1.5, se destina a explorar la posibilidad de obtener ventajas combinando Diversificación y Aprendizaje Profundo, por tratarse de las dos opciones existentes para evitar las limitaciones prácticas que aparecen cuando se pretende diseñar un MLP de alta capacidad expresiva –dificultades que emanan de la disponibilidad de un número acotado de ejemplos de entrenamiento.

3.1. Introducción y recordatorio

Para evitar que los capítulos de alto contenido experimental (éste y los dos siguientes) resulten de lectura difícil por incluir demasiado detalle, se ha dedicado una buena parte del Capítulo 2 a describir los elementos que se utilizan en los correspondientes experimentos; de modo que aquí sólo recordaremos, por posibilitar una lectura independiente, lo esencial de dichos elementos (pero no entrando en razones para su elección y omitiendo la bibliografía: para eso se ruega al lector que revise dicho capítulo):

- los clasificadores básicos utilizados son los SDAE3, que consisten en 3 capas de auto-codificación diseñadas una tras otra, y un clasificador final tipo MLP

llano (concretamente, con una capa oculta);

- se recurre tanto a la diversificación mediante binarización, y en particular a los métodos OvO y ECOC (como se verá, porque se hace precisa para dar eficacia a la segunda diversificación en problemas multiclase) cuanto a la diversificación de información tipo comité: "Bagging" y "Switching" son las técnicas que se utilizarán;

- las bases de datos seleccionadas para llevar a cabo los experimentos son la clásica MNIST –por esa condición y porque existen para ella numerosos resultados experimentales con los que se puede comparar–, MNIST-B[1] –que permitirá determinar si el grado de aprendizaje que consigue la estructura profunda básica (DAE3), que dependerá del número de ejemplos disponibles, marca diferencias en la ventaja que se puede obtener diversificando–, y, a los efectos de considerar también un problema binario, RECT[2], cuyas otras características son similares a las de MNIST y MNIST-B.

Tras el recordatorio, pasamos a describir las formas de combinación de profundidad y diversidad que consideraremos, y acto seguido a los experimentos y su discusión, para establecer finalmente las conclusiones de lo que en este capítulo se expone.

3.2. Diseños considerados

Se van a utilizar dos arquitecturas básicas. La primera, el método Global (G), donde N DAE3s (aprendices) se entrenan con N conjuntos de datos diferentes generados mediante "Bagging" (no puede aplicarse el "Switching" ya que no tiene sentido la diversidad en las etiquetas en un DAE3). La salida de cada aprendiz pasa a la etapa de Binarización (OvO o ECOC) para determinar la salida de cada uno de ellos. Por último, se pasa a la etapa de agregación final ("Final Aggregation", FA),

[1]Forma abreviada para llamar a la base de datos MNIST-BASIC.
[2]Forma abreviada para llamar a la base de datos RECTANGLES.

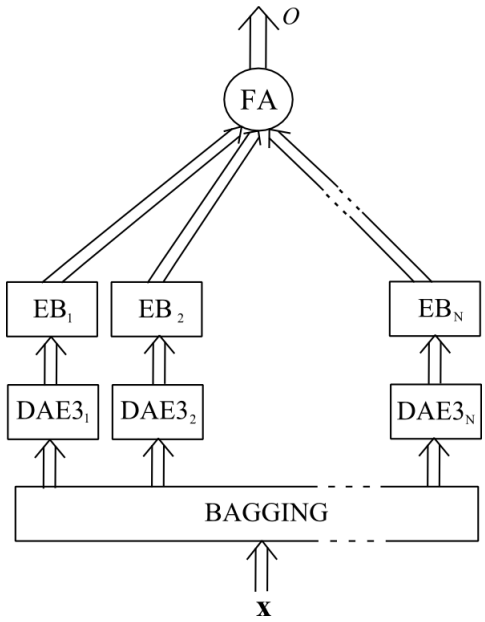

Figura 3.1: Arquitectura G para un problema multiclase. DAE3$_n$: auto-codificadores de 3 capas, EB$_n$: conjuntos de binarización, FA: agregación final.

que consiste en el recuento de votos, más votación por mayoría en el caso de OvO o selección según distancia de Hamming para el caso de ECOCs. Esta arquitectura se muestra en la Figura 3.1.

La razón para recurrir a la binarización y no sólo a la diversificación mediante "Bagging" o "Switching" radica en que una serie de ensayos preliminares permitieron concluir que la segunda sin la primera no resulta efectiva en problemas multiclase (tales como MNIST y MNIST-B).

Para la operación, basta pasar la muestra por los N auto-codificadores y agregar sus salidas tal y como se hace en el entrenamiento.

El segundo diseño que presentamos es la arquitectura de forma T; en este caso, se generan un conjunto de N máquinas a partir de la salida de un único auto-codificador, es decir, se expande formando una T; de ahí su nombre. Esta técnica requiere menos esfuerzo computacional debido a que se diseña o evalúa solamente un auto-codificador, a diferencia del método G, en el cual tenemos N auto-codificadores.

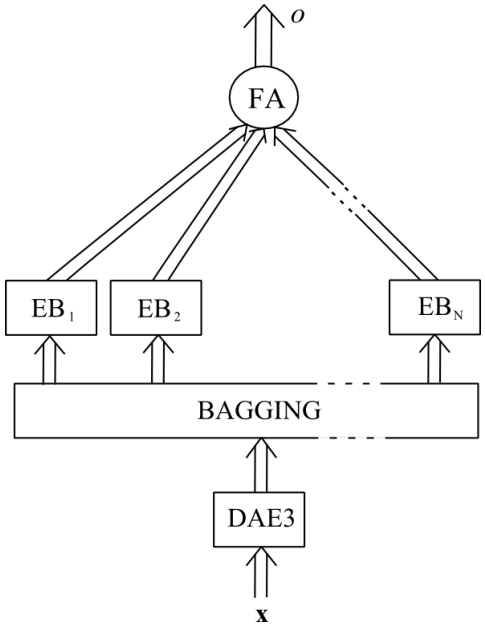

Figura 3.2: Arquitectura TB para un problema multiclase con "Bagging". DAE3: auto-codificador de 3 capas, EB_n: conjuntos de binarización, FA: agregación final.

También este caso nos merece más atención debido a que se pueden aplicar ambos métodos de diversidad, "Bagging" y "Switching". La etapa final es igual a la del método G. La arquitectura T se muestra en la Figura 3.2 para el caso con diversidad "Bagging".

Es necesario indicar que en estos dos diseños no se aplica el refinado.

Dado que se requieren N conjuntos de datos para entrenar las dos arquitecturas propuestas, se procede a generar M conjuntos, siendo $M > N$, y elegir aleatoriamente los N necesarios. Cabe indicar que el proceso de selección y generación de los conjuntos no se ha optimizado, debido a que aquí nuestro objetivo es comprobar si la diversidad puede mejorar las prestaciones del DL. Se podría, pues, mejorar estos diseños.

Se van a emplear "Bagging" y "Switching" (binario), para los que se han de seleccionar dos parámetros en cada caso. Para "Bagging", es necesario determinar el tamaño adecuado de los conjuntos de entrenamiento "Bootstrap", B, así como el

número de conjuntos N. Para el caso del "Switching", se debe determinar la tasa de volteo de etiquetas, S, y también el número de conjuntos N.

Tal y como se mencionó en el capítulo anterior, se va a elegir el procedimiento de binarización OvO, y adicionalmente se considerará el uso del ECOC de 10 clases descrito en la sección 2.2 y que, para su rápida visualización, se vuelve a mostrar en la Tabla 3.1.

	Problema														
Clase	P_0	P_1	P_2	P_3	P_4	P_5	P_6	P_7	P_8	P_9	P_{10}	P_{11}	P_{12}	P_{13}	P_{14}
C_0	1	1	0	0	0	0	1	0	1	0	0	1	1	0	1
C_1	0	0	1	1	1	1	0	1	0	1	1	0	0	1	0
C_2	1	0	0	1	0	0	0	1	1	1	1	0	1	0	1
C_3	0	0	1	1	0	1	1	1	0	0	0	0	1	0	1
C_4	1	1	1	0	1	0	1	1	0	0	1	0	0	0	1
C_5	0	1	0	0	1	1	0	1	1	1	0	0	0	0	1
C_6	1	0	1	1	1	0	0	0	0	1	0	1	0	0	1
C_7	0	0	0	1	1	1	1	0	1	0	1	1	0	0	1
C_8	1	1	0	1	0	1	1	0	0	1	0	0	0	1	1
C_9	0	1	1	1	0	0	0	0	1	0	1	0	0	1	1

Tabla 3.1: El ECOC de [Dietterich and Bakiri, 1995] para clasificación de dígitos manuscritos. C_0 a C_9 son las clases, P_0 a P_{14} las dicotomías. El código más cercano en distancia de Hamming al vector de resultados para las dicotomías indica la clase de la muestra bajo análisis

3.3. Experimentos y resultados

3.3.1. Binarización OvO

Los parámetros no entrenables, N y B para "Bagging" y N y S para "Switching", se seleccionarán por el procedimiento de validación: con la combinación que proporcione el mejor resultado para el conjunto de validación se estimará el error en el conjunto de test.

Se exploran B=60, 80, 100 y 120 % del tamaño original de los datos de entrenamiento, S=10, 20, 30 y 40 %, y N=25, 50 y 100. Utilizando diez repeticiones con inicializaciones diferentes, se calcula el promedio y la desviación típica de los errores de clasificación y se elige el menor promedio; en caso de empate, la combinación a elegir es la que proporcione menor desviación típica.

En prácticamente todos los casos la combinación que proporciona mejores prestaciones es la de N=100 y B=120 % para el caso de "Bagging"[3] y para el caso del "Switching" es N=100 y S=40 %. Como estos valores son los máximos utilizados en la exploración, se podría pensar que valores mayores podrían mejorar los resultados. La Figura 3.3 muestra el comportamiento de la superficie de error para el caso de la base de datos MNIST usando la arquitectura T con "Switching" y binarización OvO. Se ve que existe una tendencia a la saturación en la superficie de error para ambos parámetros, lo que nos garantiza que los valores elegidos son los adecuados; también se puede observar este comportamiento en la Tabla 3.2. Análogamente ocurre en los demás casos analizados, y se representan en tablas y figuras contenidas en el Anexo A.

Nótese, además, el claro paralelismo existente entre las dos curvas, que debe servir para eliminar dudas sobre la solidez de la validación realizada.

Las tasas de error ± desviación típica para la máquina SDAE3 (en porcentajes) son 1.58 ± 0.06, 3.42 ± 0.10, y 2.40 ± 0.13 para MNIST, MNIST-B, y RECT,

[3]En la arquitectura T la mejor combinación es N=100 y B=100 %; las diferencias son inapreciables.

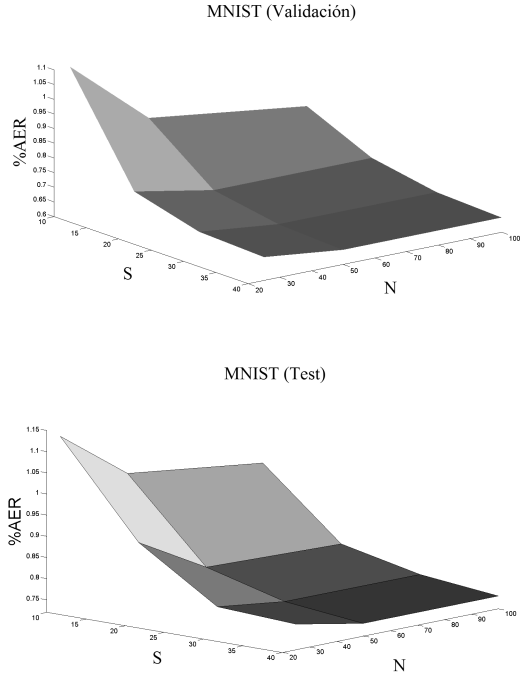

Figura 3.3: Porcentaje de tasa de error promedio (% AER) para los conjuntos de validación y test versus N y B para la base de datos MNIST usando la arquitectura TS OvO.

respectivamente.

Si se aplica únicamente OvO, la mejora es modesta: 1.40 ± 0.06 para MNIST, y 2.60 ± 0.08 para MNIST-B.

Sin embargo la combinación de binarización OvO y "Bagging" o "Switching" ofrece mejoras sustanciales. Como se muestra en la Tabla 3.3, con la arquitectura T con "Bagging" (TB) la tasa de error obtenida para MNIST es 0.77 ± 0.00, y con "Switching" (TS) es 0.75 ± 0.01. Con la arquitectura G y "Bagging", la tasa de error es algo mayor, pero aún mejor que la del SDAE3. Se puede observar que con las otras bases de datos también se presenta este comportamiento. En el caso de RECT, problema binario, no tiene sentido la binarización.

Las tasas de error se encuentran entre el 46 % y el 54 % de las del SDAE3.

	N \ S	10 %	20 %	30 %	40 %
Validación	25	1.10 ± 0.02	0.75 ± 0.01	0.69 ± 0.02	0.68 ± 0.03
	50	0.87 ± 0.01	0.70 ± 0.01	0.66 ± 0.00	0.65 ± 0.01
	100	0.80 ± 0.01	0.70 ± 0.00	0.66 ± 0.00	$\mathbf{0.65 \pm 0.00}$
Test	25	1.13 ± 0.02	0.91 ± 0.00	0.79 ± 0.03	0.78 ± 0.03
	50	1.01 ± 0.02	0.82 ± 0.01	0.77 ± 0.02	0.75 ± 0.03
	100	0.97 ± 0.01	0.81 ± 0.01	0.77 ± 0.00	$\boldsymbol{0.75 \pm 0.01}$

Tabla 3.2: Tasa de error promedio ± desviación típica (%) en TS con binarización OvO para los conjuntos de validación y test de MNIST. En cursiva aparecen los valores elegidos por validación y en negrita los mejores valores.

	MNIST	MNIST-B	RECT
SDAE3	1.58 ± 0.06	3.42 ± 0.10	2.40 ± 0.13
GB	0.86 ± 0.01	1.76 ± 0.04	1.20 ± 0.04
TB	0.77 ± 0.00	1.68 ± 0.04	1.19 ± 0.01
TS	0.75 ± 0.01	1.67 ± 0.04	1.10 ± 0.02

Tabla 3.3: Tasa de error de test (en porcentaje) para un clasificador DAE único (SDAE3), y para las arquitecturas G y T con "Bagging" (GB y TB) o "Switching" (TS) con binarización OvO.

Comparando las arquitecturas, la forma T presenta un error 10 % menor que la forma G, y en cuanto a "Bagging" y "Switching", hay muy pequeña diferencia con la estructura T. También hay que indicar que la estructura T es computacionalmente menos costosa que la estructura G. La arquitectura TS resulta ser la mejor, aunque la diferencia con TB es mínima en las tres bases de datos.

Recuérdese que la simple binarización OvO es modestamente efectiva en la diver-

sificación de los clasificadores SDAE3; las mejoras son un tanto mayores en MNIST-B que en MNIST. Como la tarea de reconocimiento es la misma, se concluye que la diversificación (por binarización) es más efectiva cuando se utilizan DNNs más débiles.

3.3.2. Binarización ECOC

Es sabido que, en la mayoría de los casos, los ECOCs son mejores que el OvO para incrementar las prestaciones de un clasificador convencional. Se considerará, por sencillez, el mejor de los casos anteriores (TS), para aplicar el ECOC en lugar del OvO. El "Switching", en este caso, se aplica por separado a cada problema binario del ECOC (no en común, tal como se ha hecho en OvO).

El proceso se iniciará como en la binarización OvO, es decir, se determinarán los parámetros no entrenables, N y S, utilizando el conjunto de validación. Se eligen N=21, 51, 101 y 121 y tasas de "Switching" S=10, 20, 30 y 40 %. Los valores que proporcionan las mejores prestaciones se obtienen con N=101 y S=30 %. Al igual que en el caso de OvO, la saturación se presenta cuando N=101 para todos los casos de S. A diferencia del caso OvO, se percibe un mínimo, alrededor de S=30 %, y se ve empeoramiento con tasas mayores de volteado de etiquetas. El porcentaje de error obtenido en test para esta combinación de parámetros y arquitectura T (TS) es 0.36 ± 0.02 %, que claramente es mucho mejor que OvO y, además, está cerca del récord con DCNN con "Drop-Connect", 0.21 %.

Para la base de datos MNIST-B se obtiene una tasa de error de 0.75 ± 0.01 %, obviamente mucho mejor que la del OvO.

Todo lo anterior se puede comprobar en la Tabla 3.4 y Figura 3.4. Esas tabla y figura para MNIST-B se anexan en el Apéndice A.

En resumen: la arquitectura que proporciona mejores prestaciones es la de forma T. Y, específicamente, la combinación TS con binarización ECOC, que además requiere menos esfuerzo computacional, tanto en la fase de diseño como en la de operación, en comparación con la arquitectura G.

	N \ S	10 %	20 %	30 %	40 %
Validación	25	0.98 ± 0.06	0.66 ± 0.06	0.40 ± 0.04	0.55 ± 0.05
	51	0.87 ± 0.00	0.55 ± 0.02	0.32 ± 0.01	0.52 ± 0.04
	101	0.81 ± 0.02	0.52 ± 0.02	$\mathbf{0.30 \pm 0.02}$	0.52 ± 0.04
	121	0.81 ± 0.02	0.52 ± 0.02	$\mathbf{0.30 \pm 0.02}$	0.52 ± 0.04
Test	25	0.91 ± 0.07	0.67 ± 0.06	0.45 ± 0.03	0.52 ± 0.03
	51	0.86 ± 0.03	0.58 ± 0.04	0.38 ± 0.02	0.48 ± 0.03
	101	0.84 ± 0.03	0.55 ± 0.04	$\boldsymbol{0.36 \pm 0.02}$	0.48 ± 0.02
	121	0.84 ± 0.04	0.55 ± 0.03	$\mathbf{0.36 \pm 0.03}$	0.48 ± 0.02

Tabla 3.4: Tasa de error promedio ± desviación típica (%) en TS con binarización ECOC para los conjuntos de validación y test de MNIST. En cursiva aparecen los valores elegidos por validación y en negrita los mejores valores.

3.3.3. Coste computacional

Dado que los experimentos se han ejecutado en una granja de cómputo[4], herramienta propia del departamento, los tiempos de ejecución no indicarán la carga computacional.

Para clasificar una muestra, dado que el número de pesos sinápticos es extremadamente grande en comparación con el número de neuronas que implementan las funciones de activación no lineal, las multiplicaciones a realizar son órdenes de magnitud más que las transformaciones no lineales. Por lo tanto, el número total de multiplicaciones se utilizará como representativo de la complejidad computacional.

Para los problemas que estamos estudiando, el vector de entrada al SDAE3 es de

[4]1024 cores, 4 TBytes de memoria RAM (1 GByte por core), velocidad sostenida de 15 Tflops, capacidad de disco 40TBytes centralizados + 40TBytes distribuidos (HDFS), sistema operativo Linux (Gentoo).

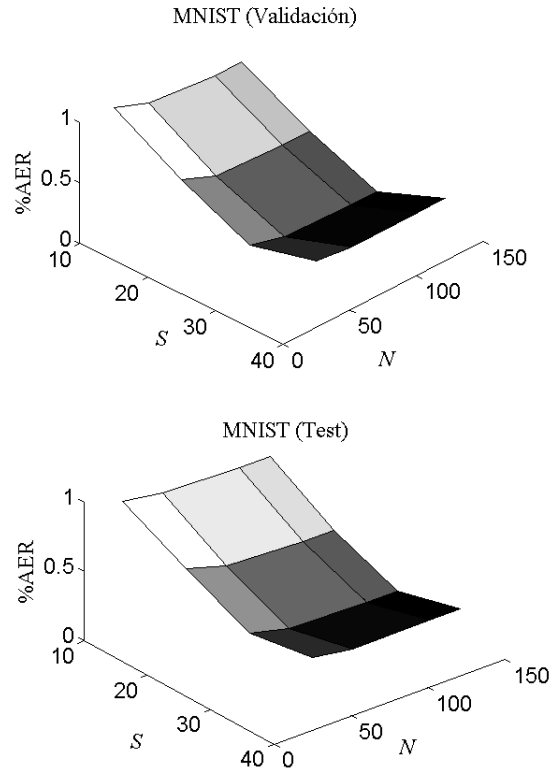

Figura 3.4: Porcentaje de tasa de error promedio (% AER) para los conjuntos de validación y test versus N y B para la base de datos MNIST usando la arquitectura TS ECOC.

dimensión 784, y alimenta tres capas ocultas cada una de 1000 neuronas, coronadas con un MLP de una capa oculta de 1000 neuronas y de 10 salidas; el número de multiplicaciones por los pesos de esta máquina es:

$$M_0 = (784)(1000) + 3(1000)(1000) + (1000)(10) \approx 4 \times 10^6$$

Esta es una estimación de la carga computacional para la operación –clasificación de una nueva muestra.

En esta discusión, se va a considerar el mejor de los diseños, la arquitectura TS con binarización ECOC (obviamente, para bases de datos multiclase). Esta máquina

requiere $\sim 3 \times 10^6$ multiplicaciones para el DAE. Las quince máquinas binarias que nacen de los ECOCs tienen N=101 clasificadores con 1000 unidades ocultas, luego suman

$$15 \times 101[(1000)(1000) + (1000)(10)] \approx 1.5 \times 10^9$$

multiplicaciones adicionales. De modo que el coste computacional en operación sube tres órdenes de magnitud respecto al de un SDAE3. Sin embargo, es importante indicar que la parte T de nuestro diseño puede ser directamente paralelizada; de este modo se reducirían, significativamente las exigencias computacionales directas.

Ahora corresponde determinar el incremento del coste computacional para el entrenamiento de la arquitectura ECOC TS respecto al clasificador SDAE3, considerando los problemas MNIST y MNIST-B. Calculamos el número de operaciones para obtener los costes de las derivadas de las salidas con respecto a los pesos, y añadimos el factor 2 porque hay que multiplicar por el paso de aprendizaje. Resaltamos que las derivadas de las unidades de las salidas aparecen en varios sitios y esto significa una multiplicación adicional cuando se usan las activaciones tangente hiperbólica: se van a ignorar estas multiplicaciones debido a que el número de unidades es mucho menor que el número de pesos.

Sea ℓ el nivel de cada capa de pesos, $\ell = 1, ..., L$. Cada capa de pesos llega a N_ℓ unidades, la dimensión de la entrada es D. El cálculo del cada gradiente incluye dos factores. El primero es la derivada del coste de salida con respecto a las salidas de las N_ℓ unidades, a partir de las mismas derivadas en la capa superior, lo cual necesita $2N_{\ell+1}N_\ell$ multiplicaciones, con excepción de la capa de salida, donde no se necesitan multiplicaciones. Después, cada una de las N_ℓ derivadas se debe multiplicar por las derivadas de las salidas con respecto a los pesos de entrada, estos son aproximadamente $N_{\ell-1}$. Esto necesita 2 productos más (excepto para los pesos de las entradas constantes). Se tiene un total de aproximadamente $2(N_{\ell+1}N_\ell + N_\ell N_{\ell-1})$ multiplicaciones para $\ell \neq L$ (utilizando $N_0 = D$), para todas las capas excepto la final, que requiere $2N_L N_{L-1}$ productos.

De acuerdo con lo expuesto, al entrenar el primer auto-codificador del clasificador SDAE3, se necesitan aproximadamente $2[(784)(1000)+(1000)(784)+(784)(1000)] \approx 4.7 \times 10^6$ multiplicaciones por paso de entrenamiento, mientras que para el segundo y tercer auto-codificador este número se convierte en $2[(1000)(1000)+(1000)(784)+(784)(1000)] \approx 5.1 \times 10^6$. Es decir, alrededor de 1.5×10^7 multiplicaciones para entrenar el SDAE3 (pero la convergencia suele ser rápida). El clasificador final requiere de $2[(1000)(1000) + (1000)(10) + (10)(1000)] \approx 2 \times 10^6$ multiplicaciones por paso de entrenamiento. Finalmente, el refinado necesita alrededor de $2\{(784)(1000) + (1000)(1000) + 3[(1000)(1000) + (1000)(1000)] + (10)(1000)\} \approx 1.5 \times 10^7$ productos por paso de entrenamiento, pero el tiempo de entrenamiento es típicamente 10 % del que se aplica en los niveles de auto-codificación. Así, 1.5×10^7 es un indicador razonable para el esfuerzo computacional en entrenamiento.

En el caso de la arquitectura TS con ECOC, para la parte común del DAE se necesitan el mismo esfuerzo de entrenamiento que el indicado anteriormente. El número de multiplicaciones para entrenar cada clasificador final binario es de aproximadamente $2[(1000)(1000)+(1000)(1)+(1)(1000)] \approx 2 \times 10^6$, pero se tiene 15 problemas binarios y se debe diseñar 4 veces (por validación cruzada la tasa de volteado de etiquetas) $25 + 51 + 101 + 121 \approx 3 \times 10^2$ clasificadores. Esto da un número total de multiplicaciones de aproximadamente $15 \times 4 \times 300 \times 2 \times 10^6 \approx 3.6 \times 10^{10}$. Aquí no hay refinado. De modo que, para entrenar el mejor de nuestros diseños de la arquitectura ECOC TS para MNIST y MNIST-B se requiere alrededor de tres órdenes de magnitud más que el del entrenamiento del clasificador SDAE3, al igual que en operación.

Este coste adicional que se paga tiene su justificación en conseguir las ventajas que un SDAE3 diversificado brinda.

3.4. Conclusiones

Con el propósito de determinar si la combinación de profundidad y diversidad resulta ventajosa respecto a un empleo separado, y tras verificar que combinarlos directamente (auto-codificación profunda con "Bagging", concretamente) es negativo en problemas multiclase, se ha evaluado en este capítulo la combinación de binarización (OvO y ECOC) con DAE y dos diversificaciones de tipo comité –"Bagging" y "Switching"– para tres bases de datos: una, MNIST, clásica como banco de pruebas; las otras, MNIST-B y RECT, para apreciar los efectos de un aprendizaje más somero y las diferencias que se presentan entre situaciones multiclase y binarias, respectivamente.

Tal evaluación se ha llevado a cabo sobre las dos arquitecturas de combinación posibles: una general (G), en que el primer paso es la diversificación (necesariamente "Bagging"), y otra en que se diversifica (tanto mediante, "Bagging" cuanto utilizando "Switching") después de aplicar el DAE, que denominamos arquitectura T; siempre incluyendo binarización.

Los resultados muestran ventaja en todos los casos; pero:

- las arquitecturas T permiten mayores ventajas (lo que demuestra que los DAE cumplen la función de desenmarañamiento que se les atribuye);
- el "Switching" es ligeramente mejor que el "Bagging";
- la binarización ECOC resulta más efectiva que la OvO (computacionalmente y en prestaciones);
- las ventajas son mayores para problemas multiclase y, en éstos, para situaciones en que el aprendizaje está limitado (por un relativamente pequeño número de ejemplos de entrenamiento disponibles).

Además, debe indicarse que no hay dificultades de selección de parámetros no entrenables, ya que se presenta una saturación con el tamaño y el indicador de

diversificación de las tasas de error que se obtienen y, además, hay un importante paralelismo de los comportamientos de los conjuntos de validación y de test.

La ventaja de prestaciones obtenida es muy apreciable: así, para MNIST, la mejor arquitectura (T con ECOC y "Switching") rebaja la tasa de error de un SDAE3 de 1.58 % a 0.36 %: este último valor es muy inferior al récord previo utilizando DAEs, que era del 0.81 %, y se acerca notablemente al récord absoluto, 0.21 %.

Lógicamente, estas ventajas de prestaciones se obtienen a cambio de algo: concretamente, de un aumento de carga computacional con respecto al SDAE3 básico de unos 4 órdenes de magnitud en el diseño y unos 3 en la operación (clasificación de muestras nuevas), en el caso (representativo) del mejor diseño para MNIST. Nada que deba alarmar, dada la amplia ventaja que se consigue; tampoco nada inesperado, dado que se pretende ganar capacidad expresiva, y eso, obviamente, requiere más parámetros entrenables y, en lógica consecuencia, más carga computacional.

Ha de indicarse que no hay razón para dudar que el empleo de binarización y diversificación sea útil con otras DNNs –aunque probablemente haya que prestar atención a sus diferentes características; p. ej., las CNNs son muy sensibles a la inicialización, y habrá que considerarlo al diversificar–. Sin embargo, sí ha de admitirse que la binarización "sensu stricto" sólo es posible para un número moderado de clases (digamos una veintena), y que a partir de ese número tendría que recurrirse a métodos aproximados, que deberían elegirse cuidadosamente según el problema para evitar pérdidas innecesarias.

La calidad de los diseños que se han introducido avala el paso siguiente en esta investigación: examinar si los "trucos" habituales para mejorar las prestaciones de los clasificadores máquina convencionales tienen también éxito en este contexto. Así se procederá con el pre-énfasis en los dos capítulos que siguen (el cuarto, aplicándolo directamente a SDAE3s; el quinto, combinándolo todo). Pero ya desde aquí puede afirmarse que la combinación "Diverse and Deep Learning" (D2L) es una dirección de investigación muy prometedora, y que la inclusión de Aprendizaje Dinámico y Aprendizaje Distribuido desembocará en un D4L que será clave para conseguir lo

verdaderamente importante: el “Big Learning”.

Capítulo 4

Una posibilidad adicional: Pre-Énfasis

4.1. Noción de pre-énfasis

Como se anticipó en el apartado 2.4, la razón para pre-enfatizar ejemplos de entrenamiento –que consiste en ponderar adecuadamente los costes muestrales a minimizar– se debe al razonable deseo de prestar más atención a las muestras difíciles de clasificar y a las próximas a una razonable frontera de clasificación, en la confianza de que así se pueda mejorar las prestaciones de las correspondientes LMs. Reiteramos también que se recurre a las técnicas de pre-énfasis ante la imposibilidad de aplicar directamente "Boosting" a las DNNs, ya que éstas son LMs fuertes. Se indicó también en el Capítulo 2 que, por razones de coherencia estructural, se demoraba la revisión histórica de la aparición y evolución de estas técnicas hasta aquí; cosa a la que procedemos acto seguido, después de lo cual recordaremos las formas de ponderación que se van a utilizar, completando su discusión.

4.2. Evolución histórica

Basándose en la evidencia de que todos los métodos de entrenamiento de LMs son, en realidad, aproximaciones a la minimización de un coste medio, como en

la tasa de error –piénsese en que minimizar el error cuadrático muestral se propone ante la imposibilidad de una minimización efectiva de dicha tasa de error–, [Hart, 1968] fue la primera propuesta en el sentido de otorgar diferente atención a diferentes muestras, recurriendo a una versión extrema del pre-énfasis, la selección de muestras de entrenamiento de entre las disponibles para tal fin, utilizando una máquina tradicional (no "entrenable" de modo distinto a memorizar los ejemplos): la de k vecinos más próximos (k-NN, "k-Nearest Neighbors"). A partir de ese momento, se propusieron diversos métodos de selección de dos tipos básicos: uno, los que procedían de acuerdo con la proximidad a la frontera (o a una razonable frontera previamente establecida) de las muestras, como son [Sklansky and Michelotti, 1980], [Plutowski and White, 1993] y [Choi and Rockett, 2002]; otro, los que llevaban a cabo la selección teniendo en cuenta la dificultad de clasificar la muestra –una medida del error, o, en general del coste–, que es el caso de [Munro, 1992] y [Cachin, 1994]. Nótese cómo los títulos de estas últimas contribuciones dejan entrever lo razonable de estos procedimientos desde el punto de vista del aprendizaje. Alternativamente, se emplearon técnicas de selección para reducir el número de centroides en diseños de RBFs [Lyhyaoui et al., 1999], lo que constituye una alternativa a los diseños basados en Máximo Margen (o SVMs).

Los experimentos recogidos en [Franco and Cannas, 2000] llevaron a la conclusión de que la importancia relativa de las muestras de entrenamiento difíciles y de las próximas a la frontera depende del problema bajo estudio. En esta contribución se apoyaron las exitosas propuestas [Gómez-Verdejo et al., 2006] y [Gómez-Verdejo et al., 2008] de modificar los énfasis típicos de los conjuntos por "Boosting", incluyendo en esos énfasis medidas de proximidad a la frontera y ajustando la influencia relativa de éstas y las del error. También los trabajos [El Jelali et al., 2008b], [El Jelali et al., 2008a] y [El Jelali et al., 2009], en los que se construyen blancos "blandos" para clasificación combinando las etiquetas teóricas con las salidas de un clasificador previo mediante ponderaciones que tienen en cuenta proximidad y error, lo que produce mejoras en clasificación mediante MLPs

[El Jelali et al., 2008b] y Modelos de Mezclas Gaussianas ("Gaussian Mixture Models", GMMs) [El Jelali et al., 2008a], pero es particularmente útil el caso de GPs, ya que estas LMs, por su propia concepción como esquemas de regresión, no pueden emplearse sin modificaciones para clasificación: la introducción de blancos "blandos" lo posibilita, y mejora sensiblemente los resultados en relación a las versiones GP modificadas tradicionales. No debemos dejar de citar [Reed et al., 1995] y [Gorse et al., 1997] como precedentes en la construcción de blancos "blandos", así como [Mora-Jiménez and Figueiras-Vidal, 2009].

4.3. Formas propuestas

Los trabajos anteriores permiten concluir que es razonable dar cabida a términos que midan el error –o coste– y términos que midan la proximidad a la frontera –ambas magnitudes según los resultados proporcionados por un clasificador auxiliar– en la ponderación implicada por el pre-énfasis; además, debe añadirse un término que pese de igual modo todas las muestras, para no sobrepasar los niveles de atención a muestras "relevantes" que sean más adecuados para cada problema considerado. Todo ello, de manera ajustable: lo que se logra con una doble combinación convexa, cuyos dos parámetros pueden determinarse mediante CV. Eso es lo propuesto y evaluado en [Alvear-Sandoval et al., 2016], aplicándolo a SDAE3s, con formas convencionales para las medidas del error y de la proximidad a la frontera –se reitera que las formas más adecuadas serán dependientes del problema, pero que no deben producir grandes cambios en los resultados–, que presentamos en el Capítulo 2 y que, para comodidad de quien lea estas líneas, reproduciremos aquí:

- para el caso binario, un factor de ponderación para la muestra de entrenamiento $\mathbf{x}^{(n)}$

$$p(\mathbf{x}^{(n)}) = \alpha + (1-\alpha)[\beta(t^{(n)} - o_a^{(n)})^2 + (1-\beta)(1 - o_a^{(n)2})] \tag{4.1}$$

  donde $t^{(n)}$ es la etiqueta para $\mathbf{x}^{(n)}$ y $o_a^{(n)}$ la salida del clasificador auxiliar ante dicha muestra;

- para el caso multiclase (se suponen salidas soft-max),

$$p(\mathbf{x}^{(n)}) = \alpha + (1-\alpha)\{\beta(1-o_{ac}^{(n)})^2 + (1-\beta)[1-|o_{ac}^{(n)} - o_{ac'}^{(n)}|]\} \tag{4.2}$$

  donde $o_{ac}^{(n)}$ es la salida del clasificador guía para la clase c de $\mathbf{x}^{(n)}$, y $o_{ac'}^{(n)}$ la salida más próxima a $o_{ac}^{(n)}$ de entre las restantes para la misma muestra;

siendo α, β, $0 \leq \alpha, \beta \leq 1$, los parámetros de combinación convexa lineal, que, como ya se ha señalado, podrán determinarse mediante validación (hay conjuntos de validación preestablecidos para los problemas que se van a tratar).

Obviamente, $(t^{(n)} - o_a^{(n)})^2$ y $(1 - o_{ac}^{(n)})^2$ son los términos de error, mientras que $1 - o_a^{(n)2}$ y $1 - |o_{ac}^{(n)} - o_{ac'}^{(n)}|$ miden la proximidad a la frontera más cercana (existen alternativas que no se explorarán en esta Tesis).

α es el parámetro que limita el efecto del énfasis, y β el que marca el reparto de atención entre el error y la proximidad a la frontera. Los casos particulares que merecen ser considerados son:

$\alpha = 1$: diseño sin énfasis

$\alpha = 0$, β por validación: énfasis directo completo

$\alpha = 0$, $\beta = 1$: énfasis directo por error

$\alpha = 0$, $\beta = 0$: énfasis directo por proximidad

α por validación, $\beta = 1$: énfasis por error, moderado

α por validación, $\beta = 0$: énfasis por proximidad, moderado

Como se verá, estas formas restringidas proporcionan resultados subóptimos.

4.4. Experimentos y sus resultados

4.4.1. Pre-enfatizado

Al tratarse de un enfatizado sobre las muestras, dada las características de la arquitectura del SDAE3 –formado por un DAE con un clasificador final–, se puede realizar la ponderación de dos formas: la primera, a la entrada del SDAE3, a la que identificaremos como Pre-Énfasis Completo; y la segunda, a la salida del DAE entrenado –es decir, que la ponderación se hará a la entrada del clasificador final–, a la que llamaremos Pre-Énfasis Final. En nuestro trabajo se han examinado las dos opciones indicadas, para poder comprobar si la ventaja es menor cuando se aplica a estructuras de una sola capa, como es el clasificador final del SDAE3.

4.4.2. Clasificadores auxiliares

Un elemento importante para realizar el pre-enfatizado es el clasificador auxiliar –o guía– cuyas salidas se emplean para el cálculo del énfasis $p(\mathbf{x})$. En el caso de MLs convencionales, se sabe que la elección de una guía adecuada proporciona buenas prestaciones. Se comprobará si lo mismo ocurre cuando se trabaja con DNNs. Para ello se van a emplear dos tipos de guías:

- un perceptrón multicapa, MLP, de una capa oculta de 1000 neuronas, y
- un SDAE3, con la misma estructura que el que se va a pre-enfatizar.

Los errores que se han conseguido con estas máquinas guía se muestran en la Tabla 4.1. Se ve que el SDAE3 ofrece mejores prestaciones que el MLP, por lo que cabría esperar que el comportamiento en el diseño final tenga la misma tendencia, esto es, que la guía SDAE3 proporcione los mejores resultados.

4.4.3. Parámetros de exploración para el pre-enfatizado

Como se ha mencionado en la sección 4.3, en las ecuaciones (4.1) y (4.2), aparecen los parámetros no entrenables, α y β, que regulan la intervención de los diversos términos del pre-énfasis. Se utilizará el conjunto de validación para seleccionar los valores de dichos parámetros, el rango de búsqueda estará en el conjunto $[0, 1]$, y fijamos intervalos de 0.1, esto es, $0, 0.1, 0.2, ..., 1$, de modo que existen 121 pares de parámetros a considerar para cada base de datos.

		Bases de Datos		
Pre-énfasis	Guía	MNIST	MNIST-B	RECT
No (MLP)	-	2.66 ± 0.10	4.44 ± 0.23	7.20 ± 0.15
No (SDAE3)	-	1.58 ± 0.06	3.42 ± 0.10	2.40 ± 0.13
Completa	MLP	$0.40 \pm 0.04^*$ $(0.3, 0.6)$	0.82 ± 0.01 $(0.3, 0.5)$	0.92 ± 0.10 $(0.4, 0.4)$
Completa	SDAE3	$0.37 \pm 0.01^*$ $(0.4, 0.5)$	0.72 ± 0.01 $(0.3, 0.5)$	0.87 ± 0.04 $(0.4, 0.3)$
Final	MLP	0.57 ± 0.00 $(0.4, 0.6)$	0.91 ± 0.03 $(0.3, 0.5)$	1.26 ± 0.04 $(0.6, 0.3)$
Final	SDAE3	$0.67 \pm 0.05^*$ $(0.4, 0.4)$	0.83 ± 0.02 $(0.3, 0.5)$	$1.31 \pm 0.02^*$ $(0.4, 0.3)$

Tabla 4.1: Error de test en porcentaje más/menos desviación típica para las tres bases de datos. Los valores de α, β obtenidos por validación están entre paréntesis. El asterisco (*) indica resultado subóptimo (como se detalla en el texto).

4.4.4. Resultados

Los resultados que se han obtenido para las bases de datos que estamos estudiando se muestran en la Tabla 4.1. Se han tabulado para las dos formas de enfatizado que se

indicaron previamente, Completa y Final, y para las dos guías que se han empleado en cada caso. Se incluyen los resultados para dichas guías.

A primera vista se puede decir que existen claras mejoras de las tasas de error cuando se aplica el pre-enfatizado, como se esperaba. También de la tabla se extrae que el reconocimiento es mejor cuando:

- se utiliza el énfasis Completo, y
- la máquina guía es mejor.

4.5. Discusión de los resultados

Los mejores resultados se obtienen para el caso de guía SDAE3 con énfasis Completo. Las tasas de error obtenidas son, aproximadamente, 25 %, 21 %, y 36 % menores que sin enfatizar, para MNIST, MNIST-B, y RECT, respectivamente. La técnica propuesta de enfatizado, por tanto, mejora las prestaciones del este tipo de máquinas.

Otro de los resultados relevantes es que cuando el aprendizaje de la guía SDAE3 es menor, la mejora de la máquina final es mayor: se evidencia en el caso de MNIST-B con respecto al MNIST, y se debe a que MNIST-B tiene menor número de muestras de entrenamiento y, por tanto, no se llega a muy altas prestaciones. El pre-enfatizado en los problemas binarios es menos efectivo: obsérvese el caso de RECT.

Los tres términos que se derivan de las combinaciones de los parámetros α y β, término común, término de error y término de proximidad, juegan siempre un papel relevante: α y β, nunca son 0 ó 1. Los resultados obtenidos para versiones restringidas del énfasis (Completo y guía SDAE3) son:

MNIST:

$\alpha = 1$: 1.58 %

$\alpha = 0$, $\beta = 0.5$ por validación: 0.55 %

$\alpha = 0$, $\beta = 1$: 0.79 %

$\alpha = 0$, $\beta = 0$: 0.68 %

$\alpha = 0.2$ por validación, $\beta = 1$: 0.72 %

$\alpha = 0.2$ por validación, $\beta = 0$: 0.54 %

Es importante indicar que todos estos valores son significativamente peores que el 0.37 % obtenido con los valores validados de $\alpha = 0.4$ y $\beta = 0.5$.

MNIST-B:

$\alpha = 1$: 3.42 %

$\alpha = 0$, $\beta = 0.5$ por validación: 0.95 %

$\alpha = 0$, $\beta = 1$: 2.46 %

$\alpha = 0$, $\beta = 0$: 3.09 %

$\alpha = 0.3$ por validación, $\beta = 1$: 2.24 %

$\alpha = 0.3$ por validación, $\beta = 0$: 2.60 %

Otra vez, todos estos valores son significativamente peores que el 0.72 % obtenido con los valores validados de $\alpha = 0.3$ y $\beta = 0.5$.

RECT:

$\alpha = 1$: 2.40 %

$\alpha = 0$, $\beta = 0.3$ por validación: 1.42 %

$\alpha = 0$, $\beta = 1$: 1.88 %

$\alpha = 0$, $\beta = 0$: 1.72 %

$\alpha = 0.4$ por validación, $\beta = 1$: 1.37 %

$\alpha = 0.4$ por validación, $\beta = 0$: 1.15 %

Por tercera vez, todos estos valores son significativamente peores que el 0.87 % obtenido con los valores validados de $\alpha = 0.4$ y $\beta = 0.3$.

4.5.1. Validación

La combinación de α y β se selecciona según el mejor resultado sobre el conjunto de validación; y resulta ser óptimo –también da el mejor resultado en test– salvo en los casos marcados con asterisco en la Tabla 4.1. Para la base de datos MNIST con énfasis Completo, cuando se usa la guía MLP, el menor error de validación lo ofrece la combinación $\alpha = 0.3$ y $\beta = 0.6$; pero, aunque la diferencia sea muy pequeña, en test el menor error, 0.39 ± 0.01, lo da la combinación $\alpha = 0.4$ y $\beta = 0.6$. De igual manera ocurre cuando la guía es un SDAE3, con un error en test de 0.36 ± 0.00, pero con $\alpha = 0.3$ y $\beta = 0.5$. La diferencia es despreciable y cae dentro del margen de la desviación típica.

Para los casos de énfasis Final con guía SDAE3, las diferencias son apreciables respecto al valor óptimo en test. En MNIST con $\alpha = 0.3$ y $\beta = 0.4$, se tiene un error en test de 0.52 ± 0.01, y para RECT con $\alpha = 0.5$ y $\beta = 0.3$, de 1.08 ± 0.03. Ambos valores son mejores que los elegidos por validación, pero aún son peores que los de énfasis Completo: por ello, estas diferencias no se consideran relevantes.

Así, se puede decir que los diseños elegidos por validación son satisfactorios. Esto es debido a las variaciones lentas y la coicidencia de las regiones óptimas promedio de las superficies de validación y test, tal como representamos en la Figura 4.1 y las Tablas 4.2 y 4.3 para el caso de MNIST. El resto de casos se ilustran en el Apéndice B.

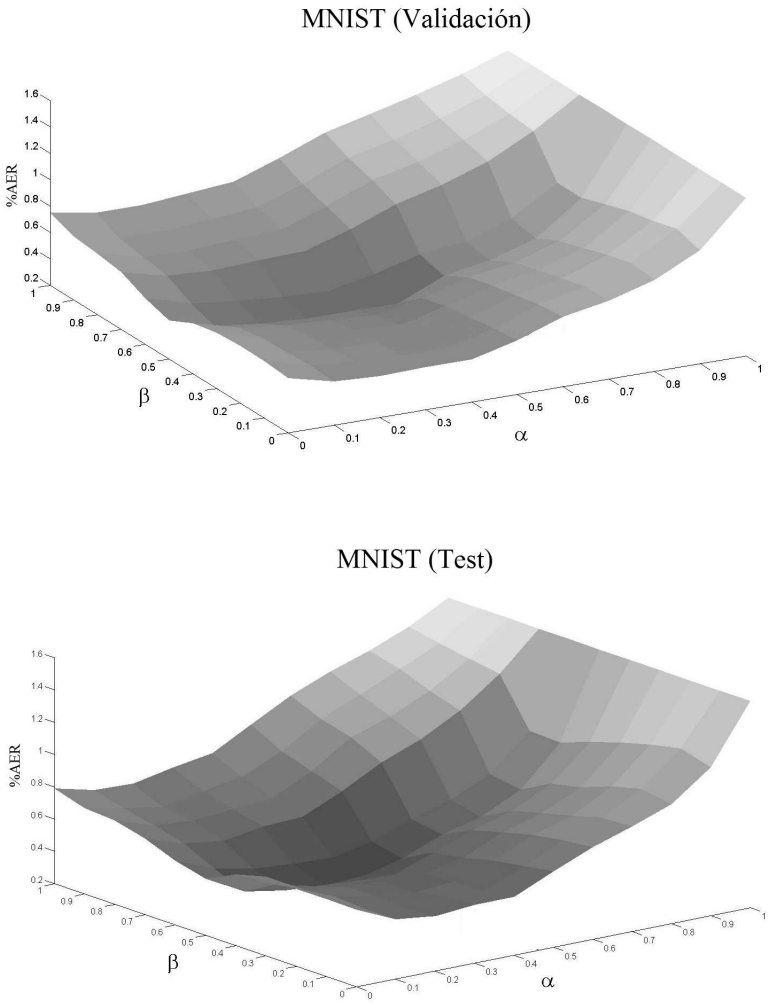

Figura 4.1: Porcentaje de tasa de error promedio (Average Error Rate, AER) para los conjuntos de validación y test versus α, β, del problema MNIST, con guía SDAE3, y énfasis Completo.

β \ α	0	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9	1
0	0.61 ± 0.01	0.53 ± 0.01	0.51 ± 0.02	0.52 ± 0.02	0.52 ± 0.01	0.61 ± 0.01	0.73 ± 0.01	0.79 ± 0.01	0.89 ± 0.01	1.06 ± 0.03	1.49 ± 0.05
0.1	0.63 ± 0.01	0.54 ± 0.02	0.51 ± 0.01	0.50 ± 0.01	0.50 ± 0.01	0.59 ± 0.04	0.71 ± 0.00	0.79 ± 0.02	0.91 ± 0.02	1.11 ± 0.01	1.49 ± 0.05
0.2	0.63 ± 0.01	0.53 ± 0.01	0.49 ± 0.01	0.49 ± 0.02	0.48 ± 0.01	0.56 ± 0.02	0.67 ± 0.00	0.74 ± 0.04	0.86 ± 0.01	1.06 ± 0.00	1.49 ± 0.05
0.3	0.62 ± 0.01	0.52 ± 0.01	0.48 ± 0.01	0.47 ± 0.01	0.45 ± 0.01	0.53 ± 0.01	0.63 ± 0.00	0.69 ± 0.03	0.80 ± 0.00	1.02 ± 0.01	1.49 ± 0.05
0.4	0.58 ± 0.01	0.49 ± 0.01	0.45 ± 0.01	0.43 ± 0.02	0.41 ± 0.01	0.48 ± 0.02	0.58 ± 0.02	0.63 ± 0.01	0.74 ± 0.01	0.96 ± 0.01	1.49 ± 0.05
0.5	0.49 ± 0.03	0.39 ± 0.00	0.35 ± 0.00	$\mathbf{0.33 \pm 0.01}$	$\mathbf{0.33 \pm 0.01}$	$\mathbf{0.33 \pm 0.01}$	0.48 ± 0.01	0.54 ± 0.01	0.65 ± 0.02	0.90 ± 0.01	1.49 ± 0.05
0.6	0.55 ± 0.03	0.45 ± 0.01	0.41 ± 0.01	0.38 ± 0.00	0.37 ± 0.01	0.44 ± 0.02	0.53 ± 0.01	0.57 ± 0.00	0.69 ± 0.01	0.90 ± 0.03	1.49 ± 0.05
0.7	0.63 ± 0.01	0.55 ± 0.04	0.53 ± 0.02	0.55 ± 0.01	0.56 ± 0.01	0.67 ± 0.01	0.80 ± 0.01	0.89 ± 0.03	1.01 ± 0.02	1.18 ± 0.03	1.49 ± 0.05
0.8	0.66 ± 0.03	0.58 ± 0.01	0.57 ± 0.01	0.59 ± 0.01	0.61 ± 0.01	0.73 ± 0.00	0.87 ± 0.01	0.95 ± 0.02	1.06 ± 0.02	1.20 ± 0.02	1.49 ± 0.05
0.9	0.68 ± 0.01	0.61 ± 0.05	0.61 ± 0.02	0.64 ± 0.00	0.67 ± 0.00	0.79 ± 0.01	0.93 ± 0.02	1.01 ± 0.03	1.11 ± 0.01	1.23 ± 0.01	1.49 ± 0.05
1	0.75 ± 0.00	0.69 ± 0.01	0.69 ± 0.01	0.72 ± 0.01	0.75 ± 0.01	0.87 ± 0.01	1.01 ± 0.01	1.09 ± 0.01	1.18 ± 0.03	1.28 ± 0.02	1.49 ± 0.05

Tabla 4.2: Resultados (porcentaje de error $\pm$ desviación típica) de validación para los valores de los distintos parámetros α y β del diseño con énfasis Completo y guía SDAE3, para MNIST. En negrita aparecen los mejores valores.

β \ α	0	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9	1
0	0.68 ± 0.02	0.57 ± 0.02	0.54 ± 0.02	0.56 ± 0.04	0.56 ± 0.02	0.67 ± 0.05	0.77 ± 0.06	0.85 ± 0.02	0.94 ± 0.03	1.12 ± 0.04	1.58 ± 0.06
0.1	0.68 ± 0.03	0.57 ± 0.01	0.53 ± 0.01	0.55 ± 0.02	0.53 ± 0.00	0.64 ± 0.02	0.75 ± 0.02	0.84 ± 0.01	0.97 ± 0.01	1.17 ± 0.02	1.58 ± 0.06
0.2	0.68 ± 0.02	0.58 ± 0.01	0.54 ± 0.00	0.51 ± 0.02	0.53 ± 0.00	0.61 ± 0.03	0.72 ± 0.02	0.79 ± 0.01	0.92 ± 0.01	1.13 ± 0.02	1.58 ± 0.06
0.3	0.69 ± 0.01	0.56 ± 0.01	0.53 ± 0.00	0.51 ± 0.00	0.51 ± 0.00	0.56 ± 0.00	0.67 ± 0.00	0.74 ± 0.01	0.85 ± 0.02	1.08 ± 0.03	1.58 ± 0.06
0.4	0.64 ± 0.01	0.53 ± 0.02	0.47 ± 0.03	0.45 ± 0.00	0.44 ± 0.01	0.53 ± 0.02	0.62 ± 0.02	0.69 ± 0.00	0.81 ± 0.01	1.02 ± 0.00	1.58 ± 0.06
0.5	0.55 ± 0.00	0.42 ± 0.01	0.38 ± 0.02	***0.36 ± 0.00***	*0.37 ± 0.01*	*0.43 ± 0.02*	0.53 ± 0.01	0.59 ± 0.01	0.71 ± 0.02	0.95 ± 0.00	1.58 ± 0.06
0.6	0.60 ± 0.00	0.50 ± 0.02	0.45 ± 0.02	0.43 ± 0.02	0.42 ± 0.01	0.47 ± 0.00	0.56 ± 0.01	0.63 ± 0.01	0.74 ± 0.02	0.96 ± 0.00	1.58 ± 0.06
0.7	0.67 ± 0.01	0.59 ± 0.03	0.56 ± 0.00	0.60 ± 0.01	0.61 ± 0.02	0.72 ± 0.00	0.85 ± 0.01	0.94 ± 0.01	1.07 ± 0.02	1.25 ± 0.00	1.58 ± 0.06
0.8	0.72 ± 0.00	0.64 ± 0.02	0.62 ± 0.00	0.64 ± 0.02	0.66 ± 0.03	0.78 ± 0.03	0.94 ± 0.01	1.02 ± 0.03	1.13 ± 0.03	1.28 ± 0.02	1.58 ± 0.06
0.9	0.73 ± 0.03	0.68 ± 0.02	0.64 ± 0.03	0.69 ± 0.03	0.73 ± 0.02	0.84 ± 0.04	0.99 ± 0.02	1.09 ± 0.02	1.20 ± 0.04	1.31 ± 0.03	1.58 ± 0.06
1	0.79 ± 0.01	0.73 ± 0.00	0.72 ± 0.03	0.77 ± 0.00	0.81 ± 0.01	0.94 ± 0.04	1.07 ± 0.02	1.17 ± 0.02	1.25 ± 0.06	1.35 ± 0.05	1.58 ± 0.06

Tabla 4.3: Resultados (porcentaje de error ± desviación típica) de test para los valores de los distintos parámetros α y β del diseño con énfasis Completo y guía SDAE3, para MNIST. En cursiva aparecen los valores elegidos por validación y en negrita los mejores valores.

4.5.2. Costes computacionales

Es obvio que la exploración de los valores de los parámetros no entrenables α, β implica repetir el proceso de entrenamiento del SDAE3 tantas veces cuantos pares de valores se consideren: aquí, $11 \times 11 = 121$; digamos dos órdenes de magnitud. También resulta evidente que, tras el entrenamiento, la máquina resultante es indistinguible de un SDAE3: de modo que el coste computacional de operación no se incrementa en absoluto por incluir el pre-énfasis; un hecho que debe resaltarse por lo que significa de beneficio a costa únicamente de un modesto esfuerzo en el diseño del clasificador máquina, pero sin gravar la operación.

4.6. Conclusiones

Hemos comprobado en este capítulo que las sencillas técnicas de pre-énfasis –ponderar los costes de las muestras de acuerdo con la importancia que cada una tenga para la resolución del problema de clasificación que se esté considerando–, ya acreditadas para máquinas llanas, son también efectivas para mejorar las prestaciones de clasificadores profundos: en concreto, de SDAE3s para las bases de datos MNIST, MNIST-B y RECT –lo que puede considerarse representativo de lo que ocurre en general.

Tal mejora es no sólo significativa, sino importante si se elige una función de pre-énfasis suficientemente general y flexible: como las versiones (multiclase y binaria) de doble combinación convexa que aquí hemos presentado y aplicado; versiones que incluyen un término común –o moderador: no implica mayor peso para ninguna de las muestras– y dos términos dependientes de las muestras: uno, según su error (en la clasificación guía o auxiliar); otro, según su proximidad a la frontera. Y se ha comprobado además que la ausencia de cualquiera de dichos tres componentes implica un decremento en la efectividad del pre-énfasis –aunque siga proporcionando mejora.

Todo ello, sin dificultades para la selección de los dos parámetros no entrenables

contenidos en la expresión de la función de énfasis: hay paralelismo entre los resultados de validación y test, y las superficies correspondientes son razonablemente planas. Tampoco hay sobrecoste computacional en la operación; y el incremento de la carga computacional de diseño es modesto (unos 2 órdenes de magnitud); de modo que el pre-énfasis no sólo es efectivo, sino eficaz.

La selección de la guía es relevante, pero no crítica: a mejor guía, más ventaja; pero las diferencias no son drásticas.

Visto esto, procederemos en el siguiente capítulo a estudiar la inclusión de pre-énfasis en diseños diversificados y profundos (D2L).

Capítulo 5

Pre-énfasis y D2L

En este capítulo se va a comprobar si combinando la binarización y la diversidad con el pre-enfatizado mejoran las prestaciones del clasificador SDAE3.

Se evaluarán dos formas de combinación:

- pre-enfatizando el mejor diseño de la sección 3.2, arquitectura TS ECOC (un SDAE3 con binarización ECOC y "Switching" para cada dicotomía)
- comenzando con binarización ECOC más SDAE3 pre-enfatizado más "Switching" por cada rama de dicotomías.

El objetivo principal es verificar si las arquitecturas SDAE3, que son representacionales, mantienen sus ventajas en proveer información valiosa del problema en las unidades profundas, a la vez que posibilitan la obtención de mejores prestaciones.

Para permitir cómodas comparaciones, en la Tabla 5.1 se presenta un resumen de resultados de los capítulos anteriores.

		VAER (±SD)	TAER (±SD)
MNIST	SDAE3	1.49 ± 0.05	1.58 ± 0.06
	PrE SDAE3 ($\alpha = 0.4, \beta = 0.6$)	0.33 ± 0.01	0.37 ± 0.01
	DAE+ECOC+SW	0.30 ± 0.02	0.36 ± 0.02
MNIST-B	SDAE3	2.65 ± 0.15	3.42 ± 0.10
	PrE SDAE3 ($\alpha = 0.3, \beta = 0.5$)	0.74 ± 0.00	0.72 ± 0.01
	DAE+ECOC+SW	0.71 ± 0.03	0.75 ± 0.01
RECT	SDAE3	2.54 ± 0.22	2.40 ± 0.13
	PrE SDAE3 ($\alpha = 0.4, \beta = 0.3$)	1.02 ± 0.09	0.87 ± 0.04
	DAE+SW	1.07 ± 0.05	1.10 ± 0.02

Tabla 5.1: Resultados para las arquitecturas: aprendiz profundo básico (SDAE3), su mejor forma de pre-enfatizado (PrE+SDAE3) y su mejor binarización (ECOC para los problemas multiclase) más diversificación "Switching" (DAE+ECOC+SW). AER(±SD): % Tasa de error promedio ± desviación típica; V: validación; T: test.

5.1. Arquitecturas

5.1.1. SDAE3 pre-enfatizado más binarización a la salida y diversificación

Este caso corresponde a aplicar pre-énfasis a la mejor arquitectura de la sección 3.2. La Figura 5.1 muestra la arquitectura completa, que denominaremos PrE+DAE3+ECOC+SW (Pre-enfatizado de DAE3 más ECOC más "SWitching").

No se ha modificado ningún parámetro no entrenable de la máquina original (sin pre-énfasis), esto es, el número de aprendices para cada problema binario es $N = 101$,

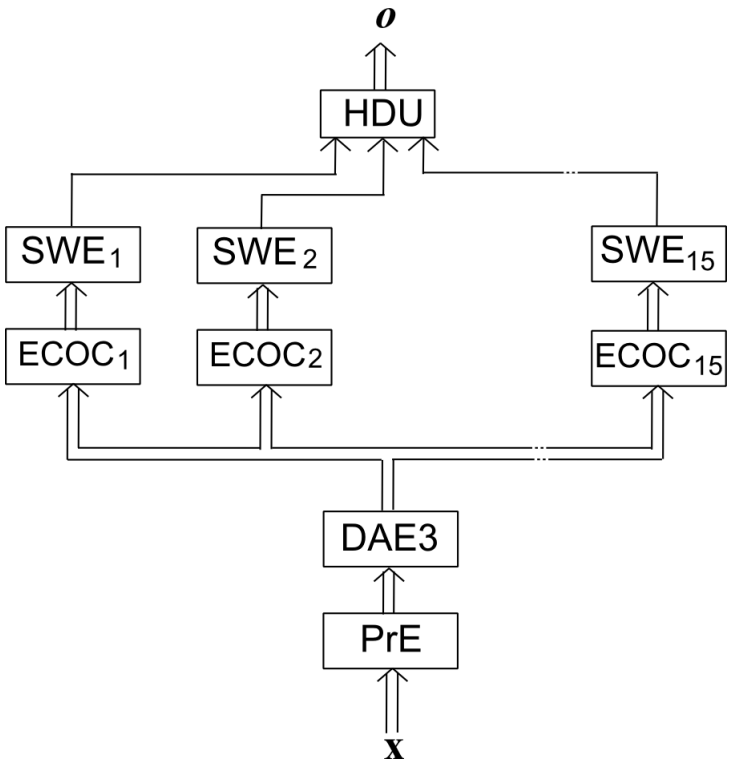

Figura 5.1: Arquitectura PrE+DAE3+ECOC+SW para un problema multiclase. PrE: unidad de pre-enfatizado. DAE3: auto-codificador de 3 capas. ECOC_m: elementos del problema de codificación. SWE_m: conjunto de máquinas "Switching", incluyendo el voto por mayoria. HDU: procesador final de distancia de Hamming, selecciona la clase correspondiente ($\boldsymbol{o}$) de la muestra de entrada $\mathbf{x}$.

la tasa de volteado de etiquetas es $S = 30\,\%$ (para MNIST y MNIST-B) y $S = 40\,\%$ (para RECT; caso sin ECOC). La razón de mantener estos parámetros es evitar el alto coste computacional adicional de diseño, que al incluir además la búsqueda de los parámetros de pre-enfatizado excedería un límite razonable. Sin embargo, es evidente que una validación conjunta podría conducir a mayores mejoras.

5.1.2. Binarización más pre-enfatizado por separado de conjuntos de máquinas con salidas diversificadas

En diseños anteriores se ha comprobado que se obtienen mejores resultados binarizando y después diversificando: no está claro que ocurra de igual manera cuando se aplica pre-enfatizado, porque cada problema binario es más flexible y, por tanto, pre-enfatizándolos por separado podría ser beneficioso.

La Figura 5.2, muestra la segunda estructura, a la que denominaremos ECOC+PrE+DAE3+SW (ECOC más Pre-enfatizado de DAE3s con conjunto de máquinas "SWitching" como clasificador final).

Para este diseño se mantienen el tamaño del conjunto de máquinas "Switching", $N = 101$, pero la tasa de volteo de etiquetas de la máquina auxiliar para el pre-enfatizado se obtiene por validación (valores de S=10, 20, 30 y 40 %), juntamente con los parámetros de pre-enfatizado; esos valores validados se adoptan en el diseño final.

La función de énfasis empleada es la misma que se ha mencionado en capítulos anteriores, considerando el caso multiclase y su equivalente para el caso binario. Se exploran los parámetros de pre-enfatizado α y β dentro del rango $[0, 1]$ en pasos de 0.1. La máquina auxiliar o guía es la versión sin enfatizar de la máquina bajo análisis. Se aplican 10 inicializaciones independientes para calcular los resultados finales.

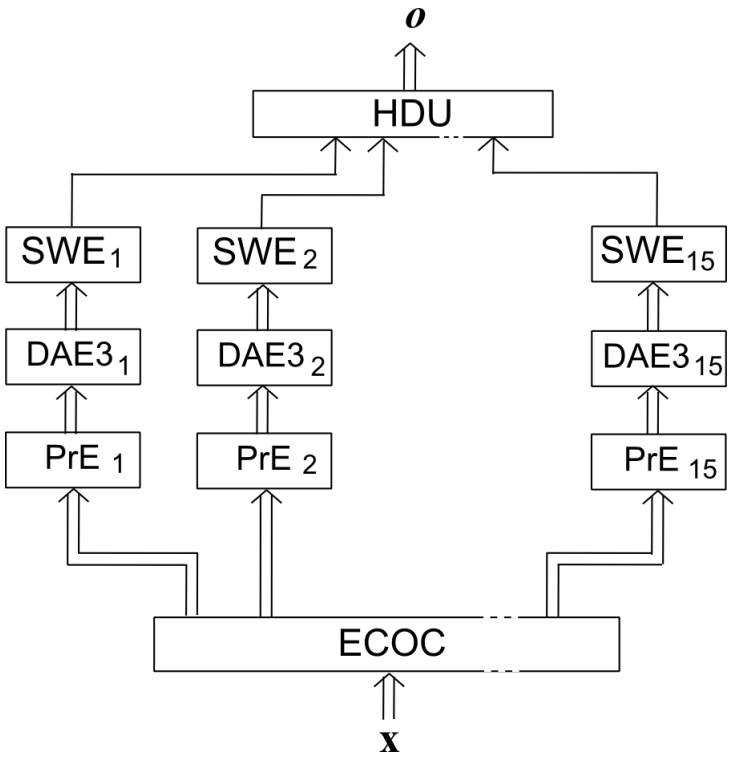

Figura 5.2: Arquitectura ECOC+PrE+DAE3+SW para un problema multiclase. Arquitectura con pre-énfasis separado. PrE$_m$: pre-enfatizado para cada problema binario a partir del ECOC. DAE3$_m$ y SWE$_m$: indica el correspondiente DAE3 y su conjunto de máquinas "Switching", incluyendo el voto por mayoria. HDU: procesador final de distancia de Hamming.

5.2. Resultados

5.2.1. Resultados para PrE+DAE3+ECOC+SW

La Tabla 5.2 muestra los resultados experimentales para la arquitectura PrE+DAE3+ECOC+SW. Se ve claramente que el pre-enfatizado supera en prestaciones a la mejor máquina diversificada –MNIST (TS ECOC): 0.36 ± 0.02, MNIST-B(TS ECOC): 0.75 ± 0.01, RECT (TS): 1.10 ± 0.02.

La menor mejora se da para el caso de RECT, que consigue un escaso beneficio

	VAER (±SD)	TAER (±SD)
MNIST ($\alpha = 0.3, \beta = 0.4$)	0.28 ± 0.06	0.30 ± 0.01
MNIST-B ($\alpha = 0.2, \beta = 0.6$)	0.63 ± 0.04	0.62 ± 0.01
RECT ($\alpha = 0.4, \beta = 0.3$)	0.81 ± 0.02	0.76 ± 0.03

Tabla 5.2: Resultados para las versiones pre-enfatizadas de la mejor arquitectura binarizada (para multiclase) y diversificada (por "Switching") (PrE+DAE3+ECOC+SW). AER(±SD): % tasa de error promedio (± desviación típica); V: validación; T: test.

en comparación con el caso de diversificación "Switching".

Como siempre, se debe destacar el papel que juegan los valores de α y β, ya que con valores que simplifiquen las ecuaciones del pre-enfatizado los resultados empeoran; por ejemplo, en el caso de MNIST, se obtienen:

- con énfasis directo por error ($\alpha = 0$, $\beta = 1$): 0.33 ± 0.03
- con énfasis directo por proximidad ($\alpha = 0$, $\beta = 0$): 0.32 ± 0.01

Las diferencias no son muy apreciables debido a que estamos comparando con una arquitectura de muy alta capacidad expresiva.

Estos resultados podrían mejorar si se aplica una búsqueda adicional más fina de los parámetros no entrenables α y β; en este caso no lo haremos, dejando esta posibilidad para el siguiente modelo.

5.2.2. Resultados para ECOC+PrE+DAE3+SW

La Tabla 5.3 muestra los valores validados de los parámetros α y β para cada problema binario para MNIST.

Una vez más, se manifiesta que el empleo de la ecuación de pre-enfatizado juega un rol importante. Todas las componentes de la ecuación están presentes en cada una de las formas óptimas para las 15 dicotomías. Es decir, que los valores de α_n y β_n son distintas de 0 y 1, y ligeramente diferentes para algunos de los problemas binarios.

	α_n	β_n	$S(\%)$	TAER
P_1	0.2	0.3	20	0.25
P_2	0.2	0.3	10	0.25
P_3	0.3	0.3	30	0.40
P_4	0.4	0.3	40	0.16
P_5	0.3	0.4	20	0.22
P_6	0.2	0.4	30	0.26
P_7	0.2	0.4	30	0.20
P_8	0.4	0.6	30	0.15
P_9	0.6	0.4	30	0.18
P_{10}	0.5	0.6	20	0.22
P_{11}	0.5	0.6	20	0.30
P_{12}	0.7	0.4	10	0.26
P_{13}	0.7	0.4	10	0.45
P_{14}	0.7	0.4	10	0.32
P_{15}	0.3	0.2	40	0.35

Tabla 5.3: Parámetros no entrenables y prestaciones (Tasa de error promedio de test(%), TAER) para cada una de las 15 máquinas que conforman el ECOC+PrE+DAE3+SW para el problema MNIST.

En la Tabla 5.4 se muestran los resultados experimentales de clasificación promediando 10 inicializaciones independientes, con los parámetros de α y β indicados en la tabla anterior.

	VAER (±SD)	TAER (±SD)
MNIST	0.24 ± 0.12	0.26 ± 0.04
MNIST-B	0.61 ± 0.02	0.55 ± 0.04

Tabla 5.4: Resultados para la arquitectura ECOC+PrE+DAE3+SW. V/TAER (±SD): validación/test; tasa de error promedio(± desviación típica) %.

Se puede ver que estos resultados son objetivamente buenos, mejores que los de las arquitecturas presentadas anteriormente. La tasa de error de test para la base de datos MNIST es de 0.26 %, prácticamente junto al récord absoluto, 0.21 %.

Forzando los valores de α_n y β_n a valores que simplifican el énfasis, vemos que para el caso de MNIST los resultados empeoran:

- clasificador auxiliar ($\alpha_n = 1$): 0.36 ± 0.02
- énfasis directo completo ($\alpha_n = 0$, $\beta_n = 0.4$) : 0.34 ± 0.02
- énfasis directo por error ($\alpha_n = 0$, $\beta_n = 1$) : 0.30 ± 0.03
- énfasis directo por proximidad ($\alpha_n = 0$, $\beta_n = 0$) : 0.28 ± 0.03

El proceso de validación para seleccionar los parámetros no entrenables no ofrece ninguna dificultad; incluso abre una posibilidad interesante, esto es, aplicar una exploración más fina de los parámetros α y β. Para ello, se realiza una búsqueda de los valores de estos parámetros con pasos más cortos, 0.02, alrededor de los valores encontrados en la validación anterior. Las tasas de error calculadas con estos nuevos parámetros se indican en la Tabla 5.5. Como se puede ver, son ligeramente inferiores. Nos hemos acercado aún más al récord.

	VAER (±SD)	TAER (±SD)
MNIST	0.22 ± 0.08	0.24 ± 0.08
MNIST-B	0.57 ± 0.07	0.52 ± 0.06

Tabla 5.5: Resultados para la arquitectura ECOC+PrE+DAE3+SW con exploración fina de α y β. V/TAER (±SD): validación/test; tasa de error promedio (± desviación típica) %.

5.3. Inclusión de Distorsión Elástica

El Aumento de Ejemplos ha sido un complemento tradicional para los algoritmos de clasificación de dígitos manuscritos [LeCun et al., 1998]. Proporciona buenas prestaciones [Tabik et al., 2017], en particular cuando se aplica en la construcción de conjuntos [Cireşan et al., 2012a]. Por tanto, añadir esto a los diseños anteriormente citados es una posibilidad interesante. Presentamos un resumen del procedimiento que se ha aplicado.

Se trabaja con diversas distorsiones, seleccionando cinco combinaciones de parámetros que producen resultados visualmente aceptables: $\{\sigma, \Delta\} = \{3, 30\}, \{4, 20\}, \{4, 30\}, \{5, 10\}$, y $\{5, 20\}$, donde el parámetro de escalado se aplica a los desplazamientos horizontal y vertical de las matrices que están formadas por valores aleatorios de una distribución uniforme entre $[-0.1, 0.1]$. Se explora la tasa apropiada de generación (de ejemplos distorsionados respecto al tamaño original de datos): $100, 200, 300$, y $400\,\%$ del número de muestras de entrenamiento, y se encuentra que $300\,\%$ es un porcentaje adecuado para la mayoría de los diseños. A la vez que se incluyen nuevas muestras distorsionadas, se explora el nivel de ruido alrededor del que previamente fue elegido $10\,\%$, y se selecciona el $7\,\%$ como nuevo valor apropiado para su varianza.

Se mantiene el resto de parámetros no entrenables en sus previos valores, excepto α, β, que se validarán después, se obtiene los % de tasas de error ± desviación

	VAER (±SD)	TAER (±SD)
MNIST	0.19 ± 0.06	0.19 ± 0.01
MNIST-B	0.52 ± 0.05	0.50 ± 0.03

Tabla 5.6: Resultados para la arquitectura ED ECOC+PrE+DAE3+SW: Diseño ECOC+PrE+DAE3+SW cuando se entrena con Aumento de Ejemplos utilizando Distorsión Elástica (ED), proceso explicado en el texto. V/TAER (±SD): validación/test; tasa de error promedio (± desviación típica) %.

típica de la Tabla 5.6 cuando se añaden las muestras con distorsiones elásticas para el entrenamiento de la máquina ECOC+PrE+DAE3+SW. Se promedia sobre 10 inicializaciones independientes.

Puede apreciarse mejoras significativas en las prestaciones, y esto sustenta la intuición de que la combinación de la distorsión elástica con el diseño de pre-enfatizado más binarización y diversificación convencional, es efectiva. Hay que resaltar que las prestaciones para MNIST constituye un nuevo récord absoluto, aun cuando las CNN son máquinas más adaptadas a la tarea de reconocimiento de dígitos manuscritos. Este hecho permite conjeturar que combinar diversidad con otros mecanismos de mejora de distinta naturaleza podría proporcionar ventajas al trabajar con DNN.

Hay que destacar que tanto el pre-enfatizado como las distorsiones elásticas incrementan el esfuerzo de diseño, pero no modifican el tamaño de la máquina diseñada y, consecuentemente, no cambian la carga computacional en operación.

5.4. Ejemplos de dígitos erróneos

En esta sección se presentan, en la Figura 5.3, las diferentes muestras que han generado error en la clasificación en test con la última de las máquinas diseñadas, ECOC+PrE+DAE3+SW más distorsión elástica, problema MNIST, para la última inicialización (los casos son semejantes para las otras nueve).

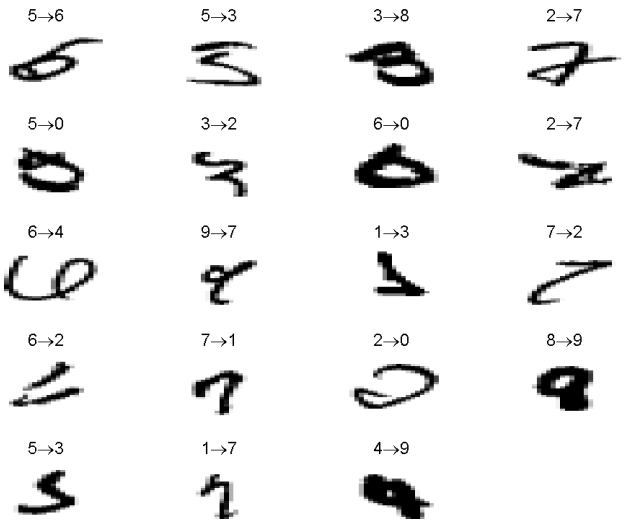

Figura 5.3: Dígitos erróneos de MNIST clasificados con ECOC+PrE+DAE3+SW y distorsión elástica. Sobre cada dígito se encuentran: a la izquierda el dígito correcto y a la derecha la salida de la máquina.

Las características de los dígitos que han sido clasificados de forma errónea corresponden a formas bastante confusas –incluso para el ojo humano–. Quizá utilizando técnicas de distorsión, por ejemplo ensanchamiento o encogimiento del dígito, las características fundamentales resultasen más visibles y podrían clasificarse correctamente.

5.5. Conclusiones

En este capítulo se han propuesto dos arquitecturas que combinan el preenfatizado con la binarización y la diversificación tradicional en máquinas profundas basadas en auto-codificadores, en particular los SDAE3. La evaluación de las prestaciones sobre las bases de datos MNIST, MNIST-B y RECT, ampliamente utilizadas

como bancos de pruebas, muestra mejoras apreciables respecto a las versiones que incluyen únicamente binarización y diversificación o solamente pre-enfatizado. Las mejoras son muy importantes con respecto a la máquina básica SDAE3, y el resultado de las últimas arquitecturas presentadas en este apartado, ECOC+PrE+DAE3+SW, prové un récord para la base de datos MNIST para diseños de clasificación basados en auto-codificadores, 0.24 %, que es muy cercano al récord absoluto de 0.21 % obtenido mediante el diseño DCNN con "Drop-Connect". Y, añadiendo Distorsión Elástica al mejor diseño previo, se baja la tasa de error para MNIST a 0.19 %, que mejora dicho récord absoluto.

Se puede concluir que la combinación del pre-enfatizado con binarización y diversificación es una opción efectiva para mejorar las prestaciones de los clasificadores basados en auto-codificadores, cuyas ventajas debidas a su carácter representacional son bien conocidas.

Queda mucho trabajo por hacer para extender esta investigación. Una de las vías por las que se puede seguir es la búsqueda de buenos mecanismos para generación de muestras –por ejemplo, la adecuada distorsión de las imágenes de entrenamiento– para resaltar las características fundamentales de las muestras. Otra de las grandes vías sería recurrir a otras máquinas profundas.

Capítulo 6

Conclusiones y oportunidades

Se ha orientado esta Tesis, como se declara en su resumen y resulta evidente en su desarrollo, a investigar si la diversificación de redes neuronales profundas produce beneficio en sus prestaciones; una posibilidad que, sorprendentemente, ha despertado poco –a lo sumo, moderado– interés entre quienes se dedican al Aprendizaje Máquina.

Naturalmente, se ha hecho necesario delimitar el ámbito en que se han llevado a cabo los correspondientes estudios y experimentos:

- las máquinas profundas consideradas han sido los clasificadores basados en auto-codificación (profunda) expansiva con reducción de ruido; y ello, porque esta familia no responde a necesidades "ad hoc" –como sí hacen las arquitecturas convolucionales– y, además, tiene carácter representacional, lo que puede suponer importantes ventajas en algunas aplicaciones;

- los modos de diversificación aplicados han sido uno-contra-uno y mediante códigos de salida correctores de errores, a efectos de binarizar problemas multiclase, y los clásicos "Bagging" y "Switching" para la diversificación de la información aplicada para el entrenamiento de los aprendices; lo segundo, por razones de

sencillez y suficiencia a efectos de llevar a cabo las comprobaciones esenciales para el fin perseguido;

- además, y ante las dificultades de incluir el más acreditado método de diversificación, "Boosting" –imposible directamente, dado que las redes profundas no son aprendices débiles; requiriendo una notable carga de cálculo adicional a la ya elevada en este contexto si se restringe al clasificador final–, se ha estudiado también la posibilidad de recurrir a técnicas de pre-énfasis –conceptualmente paralelas a las de "Boosting"–; proponiendo una forma de ponderación general y flexible, que se ha evaluado tanto aisladamente como en combinación con la diversificación;

- a los efectos de disponer de referencias experimentales amplias y acreditadas, los experimentos se han llevado a cabo con tres bases de datos muy utilizadas y de distintas características –MNIST, MNIST-BASIC y RECTANGLES–, que además consisten en datos cuya estructura es propicia para otro tipo de máquinas profundas, las convolucionales: con lo que las comparaciones no solo evitan sesgos a favor de los diseños que en esta Tesis se proponen, sino que son intrínsecamente desfavorables para estos diseños.

Cierto es que de los resultados obtenidos no puede concluirse que cualquier diversificación es beneficiosa para cualquier red neuronal profunda en la resolución de cualquier problema –y ya se han publicado trabajos en que se comprueba que no es así bajo determinadas condiciones–, pero, aparte de recordar el teorema "No Free Lunch", se puede decir que la selección de métodos y problemas posibilita generalizaciones con un elevado grado de verosimilitud.

Se pasa ahora a resumir y comentar los resultados expuestos en la Tesis.

6.1. Aportaciones de la Tesis

Por claridad, expondremos y comentaremos los resultados en el mismo orden en que se han presentado en los Capítulos 3, 4 y 5.

a) La diversificación mediante "Bagging" o "Switching" no produce beneficios (de hecho, produce pérdidas) en la resolución con clasificadores basados en auto-codificación de problemas multiclase si no se incluye binarización.

b) La binarización abre camino a la mejora de prestaciones mediante diversificación, en particular si esta se aplica en la etapa de clasificación final, arquitecturas T; lo que demuestra que los auto-codificadores expansivos con reducción de ruido cumplen perfectamente con el papel que se espera que jueguen (además, posiblemente llevan a cabo su hipotética función de desenmarañamiento). Las mejoras obtenidas son realmente significativas (la tasa de error baja a menos de la mitad con uno-contra-uno, y a una cuarta parte con códigos), sobre todo con "Switching" aplicado a arquitecturas T; mejores para binarización por código que para binarización uno-contra-uno (como cabía esperar: nótese que esto supone una ganancia en compacidad).

   En cuanto a la influencia de las características de las bases de datos, no aparecen diferencias sustanciales según ellas en los niveles de mejora; sí leves, a favor de los problemas multiclase y de situaciones de entrenamiento menos intensivo (MNIST-BASIC frente a MNIST).

c) Ha de destacarse que los procesos de validación, para los problemas considerados, se ven favorecidos por la aparición de efectos de saturación con los parámetros y con el tamaño de la diversificación. Aunque esto no admita una

generalización incondicional, sí cabe esperar que aplicar procedimientos de validación más elaborados sirva para hacer frente a situaciones menos favorables.

d) No podía dudarse de que las significativas ventajas que la diversificación (binarización más informacional) proporciona ha de pagarse de algún modo: el esfuerzo computacional de diseño y de operación se incrementa muy sensiblemente con respecto a las versiones no diversificadas; por ejemplo, para MNIST con código y "Switching", unos tres órdenes de magnitud cada uno. Sin embargo, ha de afirmarse que tales cargas de cálculo no son inasumibles –al menos para problemas de suficiente valor– y que, con el incremento de la potencia de cálculo de los ordenadores (en general), la importancia de este obstáculo es rápidamente decreciente.

e) La directa aplicación de las formas de pre-énfasis que en esta Tesis se proponen –una doble combinación convexa de ponderación uniforme, ponderación dependiente del error, y ponderación dependiente de la proximidad a la frontera– produce mejoras en las prestaciones de los clasificadores estudiados que, en cierta manera, sorprenden: reducciones de la tasa de error de tres cuartas partes, "grosso modo". Es necesario resaltar que la forma general que se propone para la ponderación, según revelan los experimentos, es clave para tanta mejora: si se eliminan términos, las prestaciones se degradan perceptiblemente. Así ocurre incluso cuando se prescinde del término constante, cuya función moderadora resulta muy provechosa.

   Otros aspectos –como la ventaja de emplear mejores guías– eran previamente conocidos.

f) Dada la forma de doble combinación convexa elegida para el pre-énfasis, resulta

computacionalmente, muy asequible su parametrización: se trata de explorar dos parámetros en el margen $[0, 1]$, lo que, para pasos de 0.1, por ejemplo, supone que la carga de diseño se incrementa únicamente en unos dos órdenes de magnitud: muy aceptable, a la vista del sustancial beneficio que se consigue. Además, tiene mucha importancia el hecho de que la máquina diseñada mediante pre-énfasis requiere en operación una carga computacional exactamente igual a la que necesita la versión diseñada directamente: no parece preciso insistir en la trascendencia de esta igualdad.

Tampoco aparecen dificultades de validación en el caso del pre-énfasis.

g) Combinar diversificación y pre-énfasis conduce a adicionales mejoras de prestaciones. Así se ha verificado experimentalmente en esta Tesis para dos casos: pre-enfatizando el mejor diseño obtenido mediante diversificación (con esa misma máquina como guía), y pre-enfatizando por separado máquinas binarias análogas correspondientes a la codificación que se ha venido aplicando. Los resultados son espectacularmente buenos, sobre todo en el segundo caso –lo que es atribuible a poder diversificar el pre-énfasis–, llegando con una exploración fina (pasos 0.02) de los parámetros de pre-énfasis a una tasa de error del 0.24 % para la base de datos MNIST: el récord absoluto (que se obtuvo mediante las más adaptadas al problema arquitecturas convolucionales), 0.21 %, es casi equivalente a lo aquí conseguido con una arquitectura genérica y sin diversificación por distorsión.

h) Por último, añadiendo al mejor diseño obtenido según lo anterior una Distorsión Elástica, se supera para el MNIST el récord recién mencionado, dejándolo en una tasa de error de tan sólo 0.19 %.

Esta larga lista de resultados significativos nos permite asegurar aquí que la opción de combinar diversificación y profundidad (lo que puede simbolizar por D2L, "Diverse and Deep Learning") no es sólo prometedora, sino –con las debidas cautelas y, en su caso, aplicando "trucos" apropiados– acreditadamente eficaz para obtener prestaciones en el "estado del arte" para problemas de clasificación.

Aunque ya se ha referenciado alguna, repetimos aquí la lista de publicaciones –aparecidas o remitidas– que presentan los resultados de la Tesis:

- la comunicación [Alvear-Sandoval and Figueiras-Vidal, 2015] incluye resultados preliminares en diversificación;

- la comunicación [Alvear-Sandoval and Figueiras-Vidal, 2016] presenta resultados iniciales de la aplicación del pre-énfasis que hemos propuesto;

- el artículo [Alvear-Sandoval et al., 2016] completa los experimentos, los resultados y la discusión relativos al pre-énfasis; equivale al Capítulo 4;

- finalmente, el artículo [Alvear-Sandoval and Figueiras-Vidal, 2018] se dedica a la combinación de pre-énfasis y diversificación, además de a la introducción de Distorsión Elástica, con un contenido análogo al del Capítulo 5.

La presentación de esta lista no debe interpretarse como que los trabajos se hayan cerrado; lo veremos en el apartado que sigue.

6.2. Líneas para futuros trabajos

Distinguiremos tres tipos de trabajos: los inmediatamente nacidos de los aquí presentados (A), los que suponen ampliaciones dentro del ámbito del aprendizaje profundo (B), y en tercer lugar, los que extienden a otros campos del aprendizaje máquina direcciones abiertas en esta Tesis (C).

Para evitar una excesiva extensión debida a largas enumeraciones, no concretaremos posibilidades cuando estas sean muy numerosas –aunque ocasionalmente se mencionen ejemplos–, salvo en aquellos casos en que haya constancia de que se ha iniciado su exploración.

A. Líneas inmediatas

A1. Comprobación del grado en que se pueden generalizar las conclusiones sobre los diseños propuestos mediante su aplicación a otras bases de datos de referencia y a problemas prácticos; prestando particular atención a las posibles técnicas de binarización para situaciones de clases numerosas.

A2. Comprobación de la ventaja de las funciones de pre-énfasis propuestas con otros tipos de máquinas. Miembros del grupo de investigación en cuyo seno se han llevado a cabo los trabajos que aquí se incluyen ya han abordado su aplicación en “Boosting”.

A3. Examen de la validez de las conclusiones relativas a la aplicación conjunta de binarización y diversificación informativa y de éstas con pre-énfasis para arquitecturas llanas: para éstas no hay estudios sistemáticos en tal sentido. Se han iniciado ya trabajos en esta línea en el grupo de investigación en el que se ha desarrollado la Tesis.

A4. Verificación de la validez de las conclusiones que se han expuesto para otros métodos de auto-codificación profunda. En particular, se acaba de iniciar una colaboración con investigadores de la Universidad de Valencia para examinar los resultados con auto-codificadores construidos con Máquinas de Aprendizaje Extremo.

A5. Valoración de las posibles mejoras derivadas de la inclusión de otros trucos y otros modos de diversificación. Particularmente atractiva parece la vía de incluir mecanismos de diversidad en los pasos de agregación, que también puede considerarse separadamente, por sí misma.

B. Ampliaciones en aprendizaje profundo

B1. Aplicación de la binarización y diversificación informativa o/y el pre-énfasis a otras familias de máquinas profundas; en particular, a las de diseño directo, y también a las convolucionales, cuya "inestabilidad" con la inicialización de los pesos supone una dificultad añadida que muy probablemente requerirá mayor elaboración de los métodos aquí explicados (pero, a la vez, ofrecen la oportunidad de conseguir prestaciones récord en problemas de referencia). Se han dado pasos preliminares en esta dirección.

B2. Lo mismo que se ha indicado en A5 (otros trucos de diversificaciones) puede estudiarse para otras familias de redes neuronales profundas.

C. Extensiones generales

C1. La posibilidad de abordar problemas singulares (desequilibrados, sensibles al coste, con costes dependientes de las observaciones, multitarea, de clasificación ordinal, etc.) con los diseños aquí propuestos (u otros análogos) merece particular atención. Y ya se han establecido colaboraciones para trabajar en ello; concretamente, en problemas de imputación de valores perdidos con investigadores de la Universidad Politécnica de Cartagena, y en clasificación ordinal, con el grupo de trabajo establecido en las Universidades de Córdoba y Loyola de Andalucía.

C2. Prever la aparición de demandas de aplicación real de estos diseños (o análogos) en situaciones "Big Data", iniciando los pasos para su eficiente implementación algorítmica usando los recursos apropiados, es asunto que no se debe olvidar.

C3. Y, finalmente: un planteamiento más a largo plazo, dado que es muy ambicioso, es combinar el aquí denominado D2L ("Diverse and Deep Learning") con otras dos clases de aprendizaje de la máxima importancia: el aprendizaje dinámico ("Dynamic Learning") y el aprendizaje distribuido ("Distributed Learning"); la convergencia de todo ello, D4L, constituirá lo que debe denominarse "Big Learning"; un concepto, a nuestro juicio, más importante aún que el de "Big Data".

Apéndice A

Tablas y gráficas para binarización y diversidad

A.1. Binarización OvO para GB

Las siguientes tablas muestran las tasas de error promedio ± desviación típica (%) para los conjuntos de validación y test, obtenidos con el diseño de arquitectura G y diversidad "Bagging" para MNIST, MNIST-B y RECT. Se tabulan de acuerdo al porcentaje de submuestreo, B, y el número de conjuntos "Bagging", N.

	N \ B	60 %	80 %	100 %	120 %
Validación	25	2.50 ± 0.04	1.73 ± 0.20	1.65 ± 0.09	1.65 ± 0.08
	50	1.65 ± 0.10	1.07 ± 0.08	1.02 ± 0.10	1.00 ± 0.10
	100	1.27 ± 0.05	0.82 ± 0.04	0.70 ± 0.02	$\mathbf{0.70 \pm 0.01}$
Test	25	2.61 ± 0.08	1.81 ± 0.21	1.72 ± 0.12	1.72 ± 0.12
	50	1.73 ± 0.12	1.11 ± 0.14	1.04 ± 0.13	1.02 ± 0.12
	100	1.31 ± 0.01	0.96 ± 0.00	$\mathbf{0.85 \pm 0.01}$	$\boldsymbol{0.86 \pm 0.01}$

Tabla A.1: Tasa de error promedio ± desviación típica (%) en GB con binarización OvO para los conjuntos de validación y test de MNIST. En cursiva aparecen los valores elegidos por validación y en negrita los mejores valores.

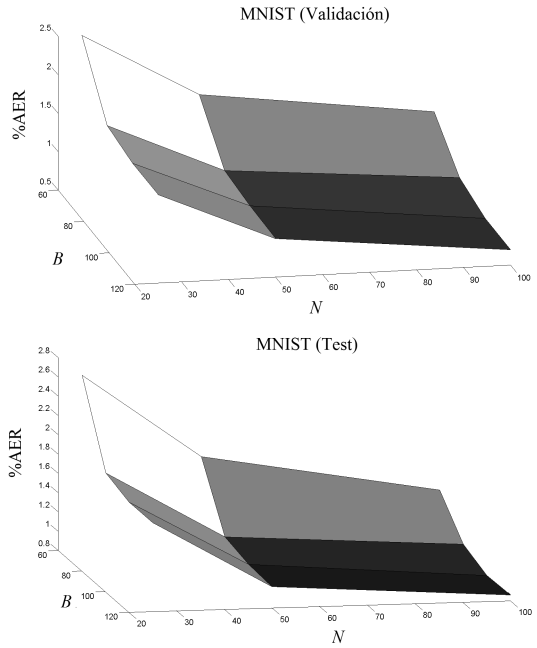

Figura A.1: Porcentaje de tasa de error promedio (% AER) para los conjuntos de validación y test versus N y B para la base de datos MNIST usando la arquitectura GB OvO.

	N \ B	60 %	80 %	100 %	120 %
Validación	25	2.09 ± 0.02	1.87 ± 0.09	1.80 ± 0.04	1.76 ± 0.06
	50	1.95 ± 0.06	1.78 ± 0.04	1.73 ± 0.03	1.72 ± 0.07
	100	1.85 ± 0.05	1.72 ± 0.03	1.70 ± 0.04	$\mathbf{1.70 \pm 0.03}$
Test	25	2.18 ± 0.04	1.91 ± 0.09	1.89 ± 0.03	1.87 ± 0.06
	50	1.99 ± 0.09	1.80 ± 0.06	1.78 ± 0.04	1.78 ± 0.04
	100	1.91 ± 0.08	1.78 ± 0.05	$\mathbf{1.76 \pm 0.04}$	$\boldsymbol{1.76 \pm 0.04}$

Tabla A.2: Tasa de error promedio ± desviación típica (%) en GB con binarización OvO para los conjuntos de validación y test de MNIST-B. En cursiva aparecen los valores elegidos por validación y en negrita los mejores valores.

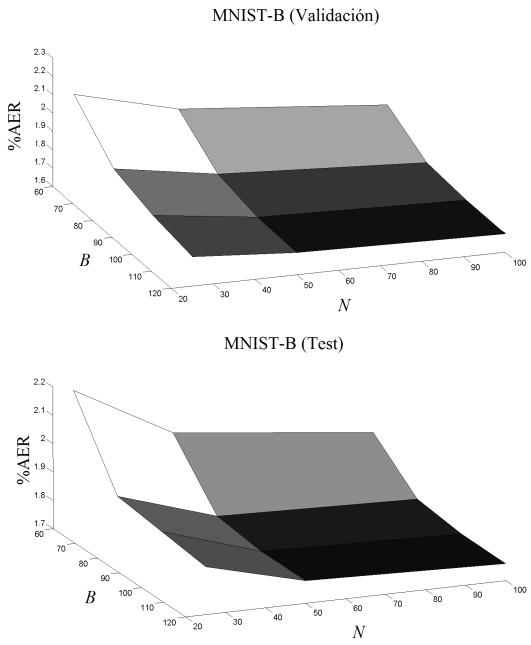

Figura A.2: Porcentaje de tasa de error promedio (% AER) para los conjuntos de validación y test versus N y B para la base de datos MNIST-B usando la arquitectura GB OvO.

	N \ B	60 %	80 %	100 %	120 %
Validación	25	1.53 ± 0.08	1.50 ± 0.02	1.45 ± 0.03	1.45 ± 0.03
	50	1.36 ± 0.01	1.30 ± 0.03	1.26 ± 0.02	1.25 ± 0.04
	100	1.30 ± 0.02	1.20 ± 0.03	1.17 ± 0.03	$\mathbf{1.17 \pm 0.02}$
Test	25	1.58 ± 0.06	1.53 ± 0.02	1.50 ± 0.04	1.48 ± 0.03
	50	1.40 ± 0.01	1.33 ± 0.04	1.29 ± 0.01	1.27 ± 0.03
	100	1.35 ± 0.03	1.26 ± 0.03	1.21 ± 0.03	$\boldsymbol{1.20 \pm 0.04}$

Tabla A.3: Tasa de error promedio ± desviación típica (%) en GB para los conjuntos de validación y test de RECT. En cursiva aparecen los valores elegidos por validación y en negrita los mejores valores.

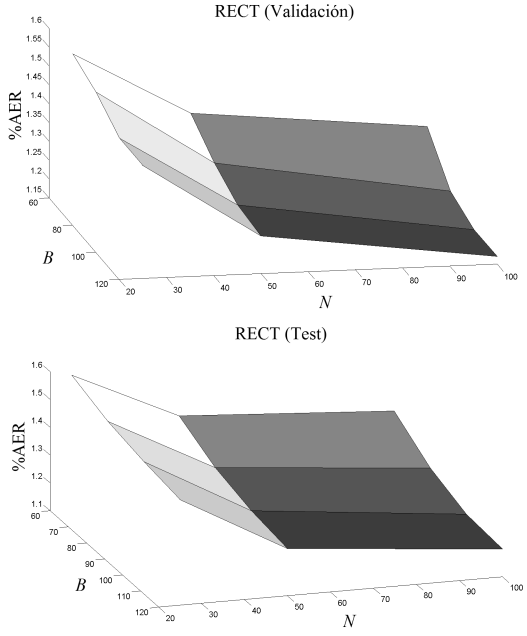

Figura A.3: Porcentaje de tasa de error promedio (% AER) para los conjuntos de validación y test versus N y B para la base de datos RECT usando la arquitectura GB.

A.2. Binarización OvO para TB

Las siguientes tablas muestran las tasas de error promedio para los conjuntos de validación y test, obtenidos con el diseño de arquitectura T y diversidad "Bagging" para MNIST, MNIST-B y RECT. Se tabulan de acuerdo al porcentaje de submuestreo, B, y número de conjuntos "Bagging", N.

	N \ B	60 %	80 %	100 %	120 %
Validación	25	1.25 ± 0.01	0.80 ± 0.02	0.69 ± 0.01	0.69 ± 0.00
	50	1.22 ± 0.01	0.77 ± 0.01	0.66 ± 0.00	0.66 ± 0.00
	100	1.19 ± 0.00	0.69 ± 0.01	$\mathbf{0.65 \pm 0.00}$	$\mathbf{0.65 \pm 0.00}$
Test	25	1.30 ± 0.00	0.81 ± 0.00	0.80 ± 0.00	0.81 ± 0.00
	50	1.27 ± 0.00	0.87 ± 0.00	0.78 ± 0.00	0.78 ± 0.00
	100	1.26 ± 0.00	0.80 ± 0.00	$\boldsymbol{0.77 \pm 0.00}$	$\boldsymbol{0.77 \pm 0.00}$

Tabla A.4: Tasa de error promedio ± desviación típica (%) en TB con binarización OvO para los conjuntos de validación y test de MNIST. En cursiva aparecen los valores elegidos por validación y en negrita los mejores valores.

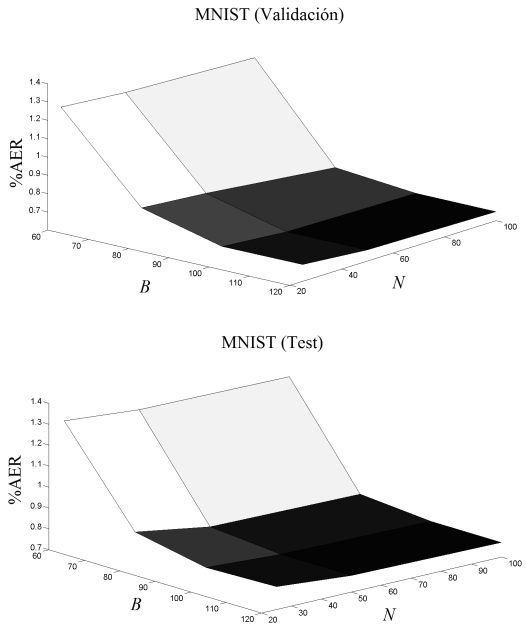

Figura A.4: Porcentaje de tasa de error promedio (% AER) para los conjuntos de validación y test versus N y B para la base de datos MNIST usando la arquitectura TB OvO.

	N \ B	60 %	80 %	100 %	120 %
Validación	25	1.25 ± 0.01	0.80 ± 0.02	0.69 ± 0.01	0.69 ± 0.00
	50	1.22 ± 0.01	0.77 ± 0.01	0.66 ± 0.00	0.66 ± 0.00
	100	1.19 ± 0.00	0.69 ± 0.01	$\mathbf{0.65 \pm 0.00}$	$\mathbf{0.65 \pm 0.00}$
Test	25	2.07 ± 0.12	1.92 ± 0.01	1.74 ± 0.05	1.72 ± 0.07
	50	1.91 ± 0.08	1.80 ± 0.04	1.69 ± 0.07	1.69 ± 0.08
	100	1.86 ± 0.07	1.78 ± 0.03	$\textit{1.69} \pm \textit{0.05}$	$\boldsymbol{\textit{1.68} \pm \textit{0.04}}$

Tabla A.5: Tasa de error promedio ± desviación típica (%) en TB con binarización OvO para los conjuntos de validación y test de MNIST-B. En cursiva aparecen los valores elegidos por validación y en negrita los mejores valores.

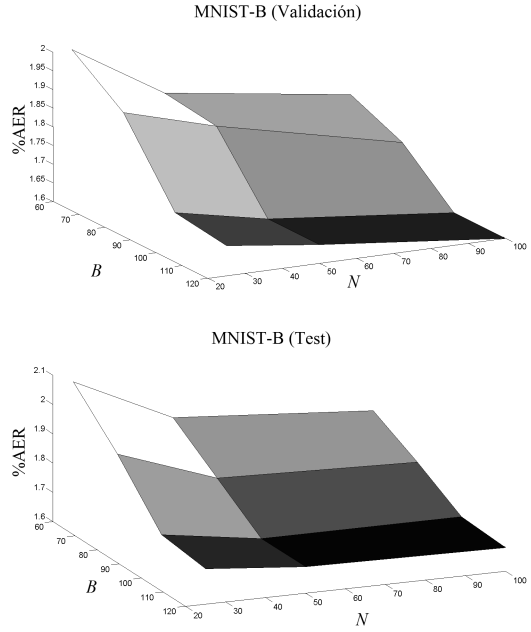

Figura A.5: Porcentaje de tasa de error promedio (% AER) para los conjuntos de validación y test versus N y B para la base de datos MNIST-B usando la arquitectura TB OvO.

	B / N	60 %	80 %	100 %	120 %
Validación	25	1.45 ± 0.01	1.41 ± 0.04	1.35 ± 0.04	1.34 ± 0.06
	50	1.30 ± 0.02	1.22 ± 0.05	1.20 ± 0.01	1.20 ± 0.01
	100	1.25 ± 0.02	1.20 ± 0.00	$\mathbf{1.16 \pm 0.01}$	$\mathbf{1.16 \pm 0.01}$
Test	25	1.50 ± 0.00	1.43 ± 0.02	1.39 ± 0.02	1.36 ± 0.08
	50	1.37 ± 0.04	1.28 ± 0.06	1.24 ± 0.01	1.23 ± 0.01
	100	1.34 ± 0.04	1.24 ± 0.00	$\mathit{1.19 \pm 0.02}$	$\mathbf{\mathit{1.19 \pm 0.01}}$

Tabla A.6: Tasa de error promedio ± desviación típica (%) en TB para los conjuntos de validación y test de RECT. En cursiva aparecen los valores elegidos por validación y en negrita los mejores valores.

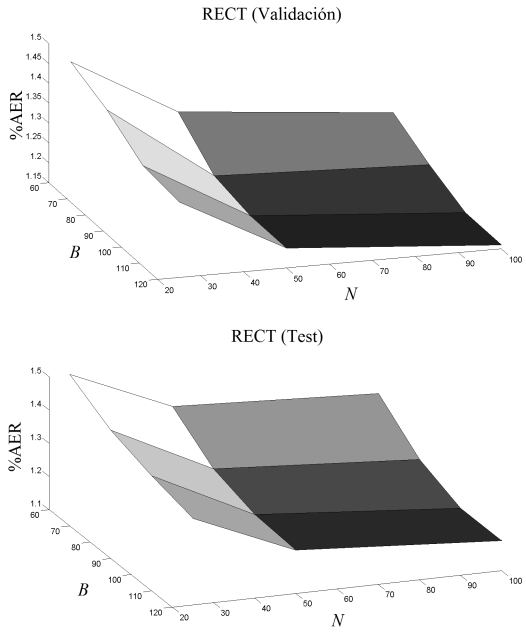

Figura A.6: Porcentaje de tasa de error promedio (% AER) para los conjuntos de validación y test versus N y B para la base de datos RECT usando la arquitectura TB.

A.3. Binarización OvO para TS

A continuación se muestran las tasas de error promedio ± desviación típica (%) de validación y test para MNIST-B y RECT, para los valores correspondiente de N y S, para el caso de binarización OvO.

	N \ S	10 %	20 %	30 %	40 %
Validación	25	2.00 ± 0.07	1.87 ± 0.01	1.70 ± 0.05	1.66 ± 0.05
	50	1.82 ± 0.02	1.73 ± 0.02	1.65 ± 0.04	1.65 ± 0.04
	100	1.77 ± 0.03	1.72 ± 0.02	1.60 ± 0.03	$\mathbf{1.60 \pm 0.02}$
Test	25	2.06 ± 0.09	1.90 ± 0.08	1.72 ± 0.04	1.70 ± 0.05
	50	1.89 ± 0.05	1.79 ± 0.01	1.67 ± 0.03	1.67 ± 0.04
	100	1.84 ± 0.06	1.77 ± 0.04	$\mathbf{1.67 \pm 0.01}$	$\boldsymbol{1.67 \pm 0.01}$

Tabla A.7: Tasa de error promedio ± desviación típica (%) en TS con binarización OvO para los conjuntos de validación y test de MNIST-B. En cursiva aparecen los valores elegidos por validación y en negrita los mejores valores.

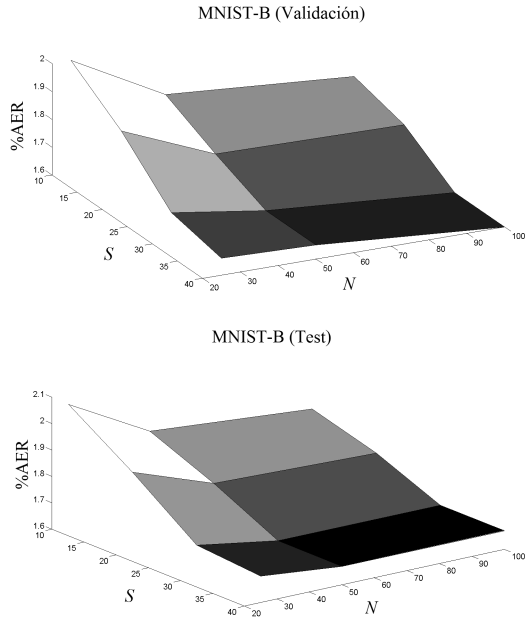

Figura A.7: Porcentaje de tasa de error promedio (% AER) para los conjuntos de validación y test versus N y B para la base de datos MNIST-B usando la arquitectura TS OvO.

	N \ S	10 %	20 %	30 %	40 %
Validación	25	1.35 ± 0.02	1.30 ± 0.03	1.19 ± 0.02	1.15 ± 0.03
	50	1.30 ± 0.07	1.20 ± 0.02	1.14 ± 0.01	1.12 ± 0.02
	100	1.27 ± 0.02	1.18 ± 0.02	1.12 ± 0.02	$\mathbf{1.08 \pm 0.01}$
Test	25	1.41 ± 0.03	1.31 ± 0.02	1.23 ± 0.04	1.22 ± 0.05
	50	1.32 ± 0.06	1.22 ± 0.02	1.16 ± 0.02	1.12 ± 0.03
	100	1.30 ± 0.02	1.21 ± 0.02	1.15 ± 0.03	$\boldsymbol{1.10 \pm 0.02}$

Tabla A.8: Tasa de error promedio ± desviación típica (%) en TS para los conjuntos de validación y test de RECT. En cursiva aparecen los valores elegidos por validación y en negrita los mejores valores.

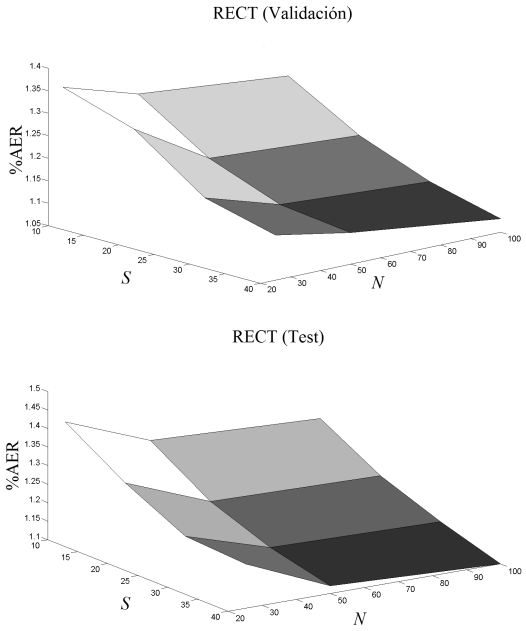

Figura A.8: Porcentaje de tasa de error promedio (% AER) para los conjuntos de validación y test versus N y B para la base de datos RECT usando la arquitectura TS.

A.4. Binarización ECOC para TS

Para la binarización ECOC se presentan las tasas de error promedio ± desviación típica (%) de validación y test para el caso de MNIST-B. Así se puede ver el paralelismo que existe entre los valores de validación y test.

	N \ S	10 %	20 %	30 %	40 %
Validación	25	1.75 ± 0.06	1.47 ± 0.02	0.84 ± 0.01	1.22 ± 0.04
	51	1.66 ± 0.04	1.30 ± 0.03	0.74 ± 0.01	1.16 ± 0.05
	101	1.60 ± 0.05	1.21 ± 0.00	0.71 ± 0.03	1.15 ± 0.03
	121	1.60 ± 0.05	1.21 ± 0.00	$\mathbf{0.71 \pm 0.02}$	1.15 ± 0.02
Test	25	1.80 ± 0.06	1.50 ± 0.02	0.86 ± 0.02	1.20 ± 0.04
	51	1.72 ± 0.04	1.32 ± 0.04	0.78 ± 0.02	1.14 ± 0.05
	101	1.68 ± 0.02	1.23 ± 0.01	0.75 ± 0.01	1.12 ± 0.02
	121	1.68 ± 0.02	1.23 ± 0.01	$\boldsymbol{0.75 \pm 0.01}$	1.12 ± 0.02

Tabla A.9: Tasa de error promedio ± desviación típica (%) en TS con binarización ECOC para los conjuntos de validación y test de MNIST-B. En cursiva aparecen los valores elegidos por validación y en negrita los mejores valores.

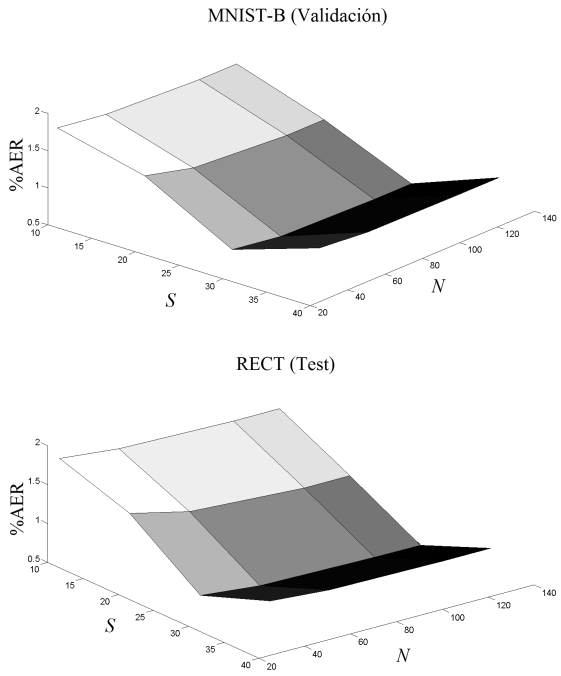

Figura A.9: Porcentaje de tasa de error promedio (% AER) para los conjuntos de validación y test versus N y B para la base de datos MNIST-B usando la arquitectura TS ECOC.

Apéndice B

Tablas y gráficas de errores para α, β

A continuación se muestran tablas y gráficas para los resultados de clasificación en función de los α y β utilizados en las ecuaciones de pre-enfatizado:

$$p(\mathbf{x}^{(n)}) = \alpha + (1-\alpha)\{\beta(1-o_{ac}^{(n)})^2 + (1-\beta)[1-|o_{ac}^{(n)} - o_{ac'}^{(n)}|]\} \tag{B.1}$$

$$p(\mathbf{x}^{(n)}) = \alpha + (1-\alpha)[\beta(t^{(n)} - o_{a}^{(n)})^2 + (1-\beta)(1-o_{a}^{(n)2})] \tag{B.2}$$

La ecuación (B.1) se utiliza para los problemas multiclase: MNIST y MNIST-B. La ecuación (B.2) se utiliza para la base de datos binaria RECT.

B.1. Guía MLP

B.1.1. Énfasis Completo guía MLP

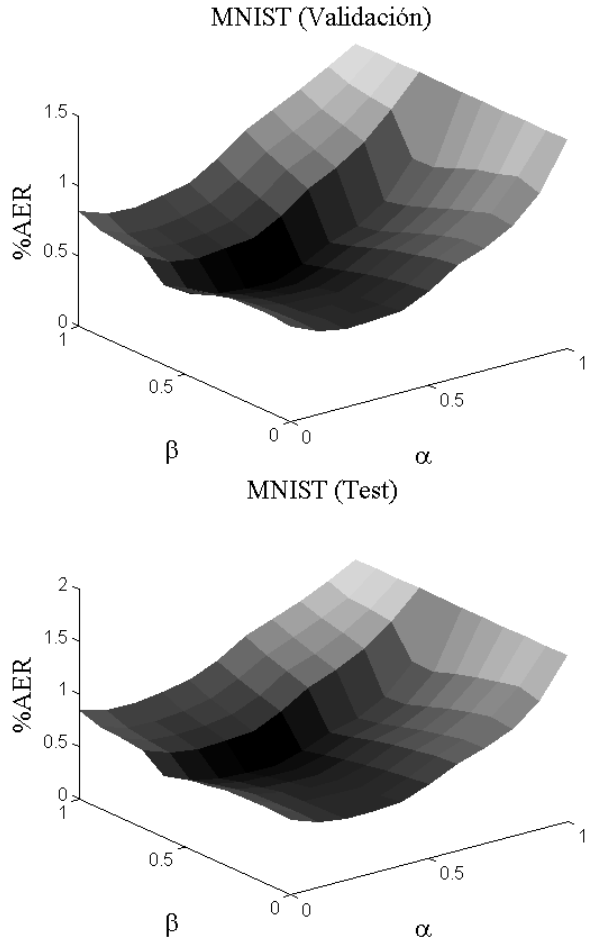

Figura B.1: Porcentaje de tasa de error promedio (Average Error Rate, AER) para los conjuntos de validación y test versus α, β, del problema MNIST, con guía MLP y énfasis Completo.

β \ α	0	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9	1
0	0.68 ± 0.02	0.59 ± 0.02	0.56 ± 0.02	0.57 ± 0.02	0.58 ± 0.02	0.67 ± 0.02	0.80 ± 0.02	0.87 ± 0.00	0.98 ± 0.03	1.16 ± 0.02	1.49 ± 0.05
0.1	0.70 ± 0.02	0.60 ± 0.03	0.56 ± 0.01	0.55 ± 0.01	0.56 ± 0.02	0.65 ± 0.03	0.78 ± 0.02	0.87 ± 0.02	1.00 ± 0.03	1.21 ± 0.02	1.49 ± 0.05
0.2	0.69 ± 0.01	0.59 ± 0.01	0.54 ± 0.03	0.54 ± 0.01	0.54 ± 0.02	0.62 ± 0.03	0.74 ± 0.04	0.82 ± 0.02	0.95 ± 0.02	1.16 ± 0.02	1.49 ± 0.05
0.3	0.69 ± 0.01	0.58 ± 0.01	0.53 ± 0.00	0.52 ± 0.01	0.51 ± 0.03	0.59 ± 0.04	0.70 ± 0.01	0.77 ± 0.03	0.89 ± 0.02	1.12 ± 0.03	1.49 ± 0.05
0.4	0.65 ± 0.03	0.55 ± 0.01	0.50 ± 0.00	0.48 ± 0.04	0.47 ± 0.01	0.54 ± 0.06	0.65 ± 0.01	0.71 ± 0.01	0.83 ± 0.04	1.06 ± 0.01	1.49 ± 0.05
0.5	0.62 ± 0.02	0.51 ± 0.00	0.46 ± 0.02	0.43 ± 0.02	0.43 ± 0.01	0.50 ± 0.02	0.60 ± 0.04	0.65 ± 0.03	0.78 ± 0.01	1.00 ± 0.00	1.49 ± 0.05
0.6	0.56 ± 0.02	0.45 ± 0.03	0.40 ± 0.02	$\mathbf{0.38 \pm 0.01}$	$\mathbf{0.38 \pm 0.01}$	0.45 ± 0.03	0.55 ± 0.03	0.62 ± 0.02	0.74 ± 0.05	1.00 ± 0.02	1.49 ± 0.05
0.7	0.70 ± 0.02	0.61 ± 0.03	0.58 ± 0.04	0.60 ± 0.01	0.62 ± 0.00	0.73 ± 0.03	0.87 ± 0.00	0.97 ± 0.00	1.10 ± 0.03	1.28 ± 0.01	1.49 ± 0.05
0.8	0.73 ± 0.01	0.64 ± 0.02	0.62 ± 0.03	0.64 ± 0.01	0.67 ± 0.01	0.79 ± 0.01	0.94 ± 0.03	1.03 ± 0.00	1.15 ± 0.03	1.30 ± 0.01	1.49 ± 0.05
0.9	0.75 ± 0.01	0.67 ± 0.01	0.66 ± 0.00	0.69 ± 0.00	0.73 ± 0.00	0.85 ± 0.01	1.00 ± 0.02	1.09 ± 0.01	1.20 ± 0.04	1.33 ± 0.01	1.49 ± 0.05
1	0.82 ± 0.00	0.75 ± 0.01	0.74 ± 0.03	0.77 ± 0.01	0.81 ± 0.00	0.93 ± 0.01	1.08 ± 0.01	1.17 ± 0.02	1.27 ± 0.02	1.38 ± 0.01	1.49 ± 0.05

Tabla B.1: Resultados (porcentaje de error $\pm$ desviación típica) de validación para los valores de los distintos parámetros α y β del diseño con énfasis Completo y guía MLP para MNIST. En negrita el mejor valor.

β \ α	0	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9	1
0	0.71 ± 0.01	0.62 ± 0.01	0.58 ± 0.00	0.59 ± 0.01	0.60 ± 0.02	0.70 ± 0.05	0.83 ± 0.01	0.91 ± 0.00	1.02 ± 0.01	1.21 ± 0.02	1.58 ± 0.06
0.1	0.73 ± 0.01	0.62 ± 0.01	0.58 ± 0.00	0.58 ± 0.02	0.58 ± 0.02	0.68 ± 0.00	0.81 ± 0.01	0.90 ± 0.01	1.04 ± 0.02	1.26 ± 0.01	1.58 ± 0.06
0.2	0.72 ± 0.00	0.62 ± 0.02	0.57 ± 0.03	0.56 ± 0.02	0.56 ± 0.00	0.65 ± 0.00	0.77 ± 0.02	0.85 ± 0.01	0.99 ± 0.01	1.21 ± 0.03	1.58 ± 0.06
0.3	0.72 ± 0.02	0.61 ± 0.03	0.56 ± 0.02	0.54 ± 0.03	0.54 ± 0.01	0.61 ± 0.03	0.73 ± 0.02	0.80 ± 0.03	0.93 ± 0.01	1.16 ± 0.01	1.58 ± 0.06
0.4	0.68 ± 0.01	0.57 ± 0.00	0.52 ± 0.01	0.50 ± 0.00	0.49 ± 0.01	0.57 ± 0.02	0.68 ± 0.01	0.74 ± 0.02	0.87 ± 0.00	1.11 ± 0.01	1.58 ± 0.06
0.5	0.65 ± 0.01	0.53 ± 0.01	0.48 ± 0.01	0.46 ± 0.01	0.45 ± 0.04	0.52 ± 0.01	0.62 ± 0.00	0.68 ± 0.01	0.79 ± 0.00	1.04 ± 0.04	1.58 ± 0.06
0.6	0.58 ± 0.01	0.47 ± 0.01	0.42 ± 0.04	*0.40 ± 0.04*	***0.39 ± 0.01***	0.47 ± 0.02	0.58 ± 0.03	0.64 ± 0.04	0.78 ± 0.02	1.04 ± 0.02	1.58 ± 0.06
0.7	0.73 ± 0.02	0.64 ± 0.01	0.61 ± 0.04	0.62 ± 0.04	0.64 ± 0.01	0.76 ± 0.01	0.91 ± 0.02	1.01 ± 0.02	1.15 ± 0.03	1.33 ± 0.02	1.58 ± 0.06
0.8	0.76 ± 0.02	0.67 ± 0.02	0.65 ± 0.03	0.67 ± 0.01	0.70 ± 0.01	0.82 ± 0.01	0.98 ± 0.01	1.07 ± 0.02	1.20 ± 0.02	1.36 ± 0.02	1.58 ± 0.06
0.9	0.78 ± 0.04	0.70 ± 0.04	0.69 ± 0.05	0.72 ± 0.02	0.76 ± 0.00	0.88 ± 0.03	1.04 ± 0.00	1.14 ± 0.03	1.25 ± 0.04	1.38 ± 0.03	1.58 ± 0.06
1	0.85 ± 0.06	0.78 ± 0.05	0.77 ± 0.06	0.81 ± 0.06	0.85 ± 0.03	0.97 ± 0.04	1.13 ± 0.03	1.22 ± 0.01	1.32 ± 0.02	1.44 ± 0.01	1.58 ± 0.06

Tabla B.2: Resultados (porcentaje de error ± desviación típica) de test para los valores de los distintos parámetros α y β del diseño con énfasis Completo y guía MLP para MNIST. En cursiva, los valores elegidos por validación, y en negrita el mejor valor en test.

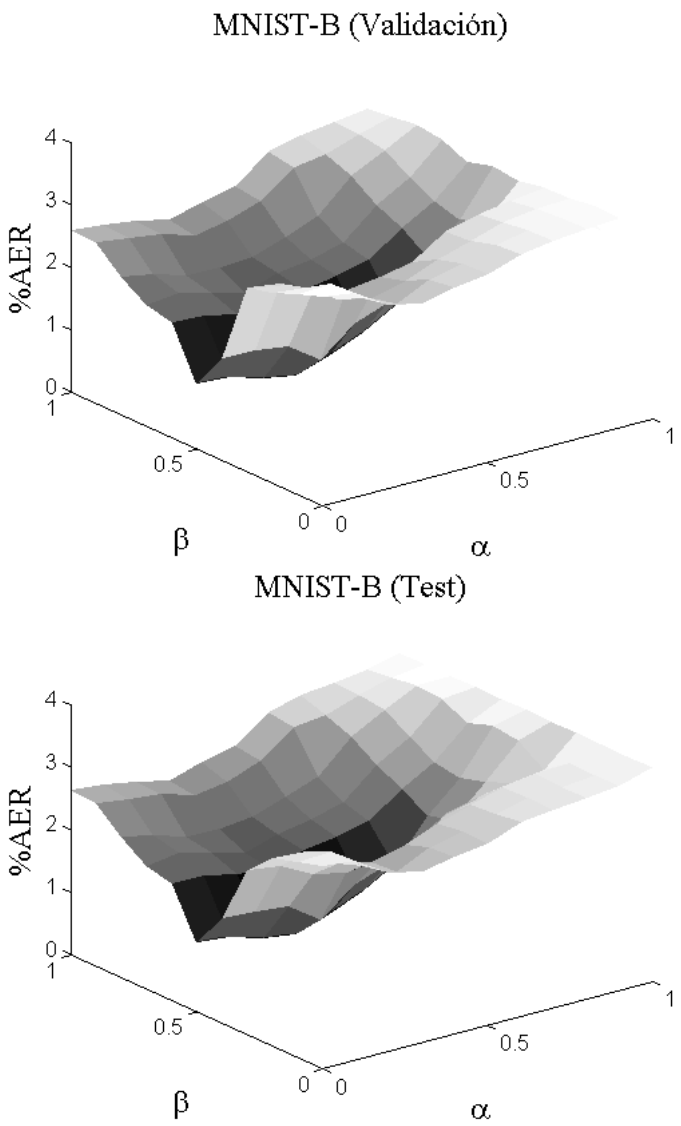

Figura B.2: Porcentaje de tasa de error promedio (Average Error Rate, AER) para los conjuntos de validación y test versus α, β, del problema MNIST-B, con guía MLP y énfasis Completo.

β \ α	0	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9	1
0	3.35 ± 0.02	3.28 ± 0.01	2.95 ± 0.02	2.83 ± 0.03	2.94 ± 0.02	3.01 ± 0.02	3.20 ± 0.01	3.27 ± 0.02	3.32 ± 0.02	3.36 ± 0.02	2.65 ± 0.15
0.1	3.22 ± 0.02	3.25 ± 0.02	2.87 ± 0.01	2.64 ± 0.02	2.88 ± 0.02	3.00 ± 0.01	3.15 ± 0.01	3.26 ± 0.05	3.30 ± 0.01	3.35 ± 0.02	2.65 ± 0.15
0.2	3.04 ± 0.02	3.00 ± 0.01	2.67 ± 0.01	2.55 ± 0.01	2.69 ± 0.01	2.89 ± 0.01	3.04 ± 0.02	3.19 ± 0.03	3.27 ± 0.01	3.30 ± 0.03	2.65 ± 0.15
0.3	2.95 ± 0.02	2.88 ± 0.00	2.60 ± 0.02	2.48 ± 0.01	2.57 ± 0.01	2.66 ± 0.02	3.00 ± 0.03	3.15 ± 0.03	3.22 ± 0.02	3.29 ± 0.02	2.65 ± 0.15
0.4	1.64 ± 0.02	1.62 ± 0.02	1.55 ± 0.01	1.20 ± 0.02	1.62 ± 0.03	1.80 ± 0.00	1.99 ± 0.01	2.22 ± 0.01	2.70 ± 0.01	3.22 ± 0.03	2.65 ± 0.15
0.5	1.05 ± 0.02	1.00 ± 0.03	0.84 ± 0.01	$\mathbf{0.75 \pm 0.02}$	0.94 ± 0.02	1.15 ± 0.01	1.44 ± 0.01	2.09 ± 0.04	2.55 ± 0.03	3.26 ± 0.01	2.65 ± 0.15
0.6	1.84 ± 0.02	1.82 ± 0.03	1.70 ± 0.01	1.68 ± 0.02	1.80 ± 0.02	1.89 ± 0.02	2.00 ± 0.02	2.38 ± 0.01	2.67 ± 0.01	3.15 ± 0.01	2.65 ± 0.15
0.7	1.94 ± 0.01	2.00 ± 0.02	1.86 ± 0.01	1.72 ± 0.02	1.89 ± 0.02	1.97 ± 0.02	2.11 ± 0.01	2.57 ± 0.01	2.88 ± 0.01	3.34 ± 0.01	2.65 ± 0.15
0.8	2.21 ± 0.01	2.10 ± 0.02	1.94 ± 0.02	1.93 ± 0.02	1.97 ± 0.01	2.15 ± 0.01	2.34 ± 0.03	2.85 ± 0.01	3.12 ± 0.01	3.38 ± 0.01	2.65 ± 0.15
0.9	2.57 ± 0.01	2.50 ± 0.02	2.43 ± 0.02	2.27 ± 0.02	2.35 ± 0.00	2.55 ± 0.01	2.94 ± 0.04	3.00 ± 0.02	3.18 ± 0.01	3.33 ± 0.01	2.65 ± 0.15
1	2.60 ± 0.00	2.54 ± 0.00	2.45 ± 0.01	2.33 ± 0.03	2.48 ± 0.00	2.60 ± 0.00	3.00 ± 0.02	3.13 ± 0.02	3.20 ± 0.00	3.27 ± 0.01	2.65 ± 0.15

Tabla B.3: Resultados (porcentaje de error $\pm$ desviación típica) de validación para los valores de los distintos parámetros α y β del diseño con énfasis Completo y guía MLP para MNIST-B. En negrita el mejor valor.

β \ α	0	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9	1
0	3.25 ± 0.03	3.18 ± 0.02	2.85 ± 0.01	2.73 ± 0.02	2.84 ± 0.00	2.91 ± 0.01	3.10 ± 0.01	3.17 ± 0.00	3.22 ± 0.01	3.26 ± 0.02	3.42 ± 0.10
0.1	3.13 ± 0.03	3.16 ± 0.02	2.78 ± 0.03	2.55 ± 0.01	2.79 ± 0.02	2.91 ± 0.00	3.06 ± 0.00	3.17 ± 0.00	3.21 ± 0.00	3.26 ± 0.02	3.42 ± 0.10
0.2	3.01 ± 0.01	2.97 ± 0.02	2.64 ± 0.02	2.52 ± 0.02	2.66 ± 0.01	2.86 ± 0.01	3.01 ± 0.01	3.16 ± 0.01	3.24 ± 0.01	3.27 ± 0.02	3.42 ± 0.10
0.3	2.63 ± 0.01	2.56 ± 0.00	2.28 ± 0.03	2.16 ± 0.02	2.25 ± 0.02	2.34 ± 0.00	2.68 ± 0.02	2.83 ± 0.00	2.90 ± 0.02	2.97 ± 0.01	3.42 ± 0.10
0.4	1.67 ± 0.02	1.65 ± 0.01	1.58 ± 0.02	1.23 ± 0.01	1.65 ± 0.02	1.83 ± 0.00	2.02 ± 0.00	2.25 ± 0.02	2.73 ± 0.01	3.25 ± 0.00	3.42 ± 0.10
0.5	1.11 ± 0.00	1.06 ± 0.01	0.90 ± 0.01	***0.82 ± 0.01***	1.00 ± 0.01	1.21 ± 0.00	1.50 ± 0.01	2.15 ± 0.02	2.61 ± 0.02	3.32 ± 0.02	3.42 ± 0.10
0.6	1.87 ± 0.02	1.85 ± 0.01	1.73 ± 0.01	1.71 ± 0.01	1.83 ± 0.01	1.92 ± 0.00	2.03 ± 0.02	2.41 ± 0.02	2.70 ± 0.01	3.18 ± 0.01	3.42 ± 0.10
0.7	1.97 ± 0.02	2.03 ± 0.01	1.89 ± 0.02	1.75 ± 0.01	1.92 ± 0.00	2.00 ± 0.01	2.14 ± 0.01	2.60 ± 0.00	2.91 ± 0.01	3.37 ± 0.00	3.42 ± 0.10
0.8	2.24 ± 0.03	2.13 ± 0.00	1.97 ± 0.01	1.96 ± 0.00	2.00 ± 0.02	2.18 ± 0.00	2.37 ± 0.02	2.88 ± 0.02	3.15 ± 0.00	3.41 ± 0.01	3.42 ± 0.10
0.9	2.60 ± 0.04	2.53 ± 0.00	2.46 ± 0.02	2.30 ± 0.01	2.38 ± 0.01	2.58 ± 0.01	2.97 ± 0.02	3.03 ± 0.00	3.21 ± 0.01	3.36 ± 0.01	3.42 ± 0.10
1	2.63 ± 0.03	2.57 ± 0.01	2.48 ± 0.00	2.36 ± 0.00	2.51 ± 0.02	2.63 ± 0.02	3.03 ± 0.00	3.16 ± 0.00	3.23 ± 0.01	3.30 ± 0.01	3.42 ± 0.10

Tabla B.4: Resultados (porcentaje de error ± desviación típica) de test para los valores de los distintos parámetros α y β del diseño con énfasis Completo y guía MLP para MNIST-B. En cursiva, los valores elegidos por validación, y en negrita el mejor valor en test.

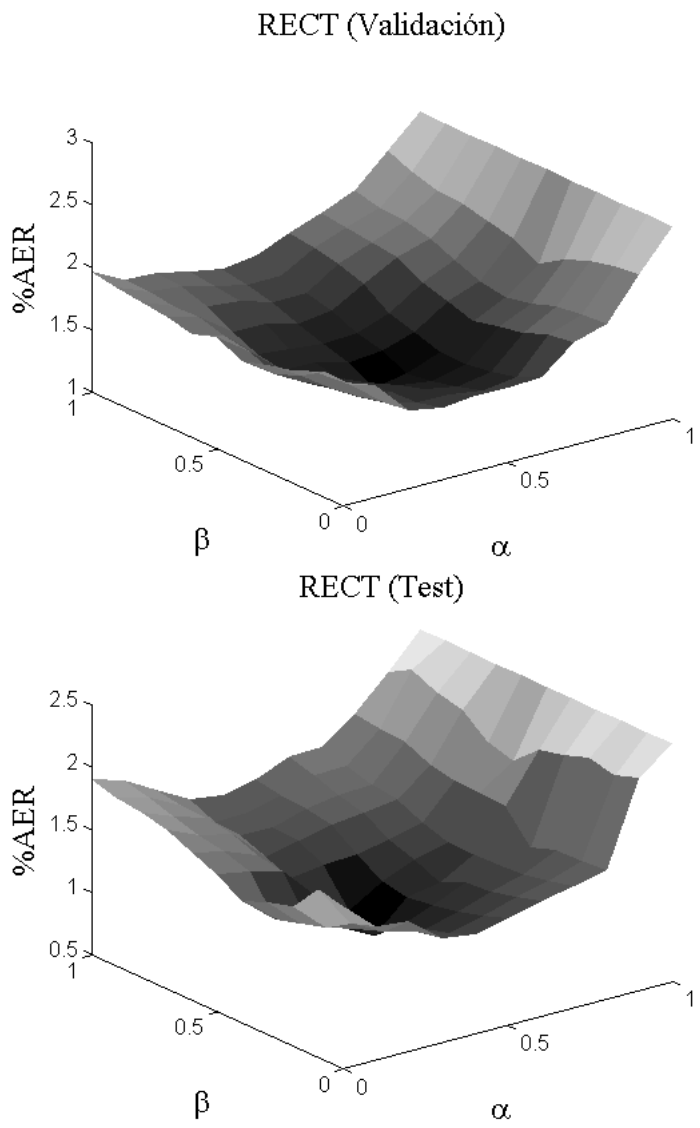

Figura B.3: Porcentaje de tasa de error promedio (Average Error Rate, AER) para los conjuntos de validación y test versus α, β, del problema RECT, con guía MLP y énfasis Completo.

β \ α	0	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9	1
0	1.99 ± 0.12	1.90 ± 0.10	1.63 ± 0.08	1.58 ± 0.05	1.60 ± 0.10	1.60 ± 0.08	1.61 ± 0.07	1.82 ± 0.10	1.90 ± 0.14	2.22 ± 0.10	2.54 ± 0.22
0.1	1.97 ± 0.08	1.81 ± 0.07	1.60 ± 0.11	1.53 ± 0.08	1.52 ± 0.12	1.48 ± 0.12	1.52 ± 0.06	1.79 ± 0.12	1.88 ± 0.16	2.20 ± 0.08	2.54 ± 0.22
0.2	1.90 ± 0.11	1.82 ± 0.09	1.55 ± 0.07	1.52 ± 0.12	1.47 ± 0.10	1.42 ± 0.10	1.47 ± 0.10	1.71 ± 0.08	1.86 ± 0.09	2.15 ± 0.12	2.54 ± 0.22
0.3	1.85 ± 0.04	1.70 ± 0.06	1.52 ± 0.05	1.50 ± 0.09	1.41 ± 0.08	1.38 ± 0.09	1.44 ± 0.08	1.67 ± 0.08	1.78 ± 0.12	2.07 ± 0.07	2.54 ± 0.22
0.4	1.80 ± 0.07	1.62 ± 0.08	1.47 ± 0.10	1.45 ± 0.07	$\mathbf{1.30 \pm 0.07}$	1.35 ± 0.05	1.45 ± 0.07	1.62 ± 0.06	1.75 ± 0.08	1.94 ± 0.08	2.54 ± 0.22
0.5	1.90 ± 0.10	1.71 ± 0.08	1.49 ± 0.03	1.48 ± 0.10	1.41 ± 0.11	1.44 ± 0.06	1.52 ± 0.06	1.69 ± 0.10	1.78 ± 0.08	2.06 ± 0.04	2.54 ± 0.22
0.6	1.83 ± 0.14	1.75 ± 0.11	1.52 ± 0.10	1.50 ± 0.09	1.47 ± 0.14	1.49 ± 0.07	1.58 ± 0.11	1.75 ± 0.15	1.86 ± 0.12	2.10 ± 0.04	2.54 ± 0.22
0.7	1.87 ± 0.09	1.80 ± 0.10	1.62 ± 0.07	1.61 ± 0.10	1.54 ± 0.13	1.58 ± 0.08	1.72 ± 0.12	1.86 ± 0.12	1.94 ± 0.12	2.15 ± 0.10	2.54 ± 0.22
0.8	1.90 ± 0.13	1.80 ± 0.12	1.73 ± 0.09	1.70 ± 0.14	1.62 ± 0.15	1.62 ± 0.12	1.79 ± 0.10	1.89 ± 0.16	1.99 ± 0.10	2.17 ± 0.08	2.54 ± 0.22
0.9	1.93 ± 0.10	1.82 ± 0.12	1.80 ± 0.07	1.73 ± 0.10	1.67 ± 0.12	1.67 ± 0.15	1.83 ± 0.08	1.92 ± 0.13	2.01 ± 0.06	2.22 ± 0.12	2.54 ± 0.22
1	1.96 ± 0.09	1.83 ± 0.06	1.81 ± 0.08	1.75 ± 0.12	1.70 ± 0.09	1.74 ± 0.10	1.85 ± 0.06	1.94 ± 0.16	2.05 ± 0.12	2.31 ± 0.10	2.54 ± 0.22

Tabla B.5: Resultados (porcentaje de error $\pm$ desviación típica) de validación para los valores de los distintos parámetros α y β del diseño con énfasis Completo y guía MLP para RECT. En negrita el mejor valor.

β \ α	0	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9	1
0	1.80 ± 0.12	1.57 ± 0.09	1.54 ± 0.10	1.33 ± 0.15	1.30 ± 0.05	1.35 ± 0.07	1.42 ± 0.07	1.47 ± 0.10	1.53 ± 0.04	2.20 ± 0.10	2.40 ± 0.13
0.1	1.85 ± 0.15	1.50 ± 0.10	1.37 ± 0.10	1.29 ± 0.04	1.21 ± 0.04	1.28 ± 0.09	1.36 ± 0.09	1.42 ± 0.06	1.51 ± 0.05	2.13 ± 0.12	2.40 ± 0.13
0.2	1.62 ± 0.13	1.38 ± 0.10	1.30 ± 0.09	1.22 ± 0.10	1.15 ± 0.08	1.22 ± 0.07	1.31 ± 0.06	1.38 ± 0.13	1.50 ± 0.09	2.17 ± 0.12	2.40 ± 0.13
0.3	1.56 ± 0.08	1.32 ± 0.12	1.24 ± 0.13	1.12 ± 0.06	1.08 ± 0.11	1.19 ± 0.09	1.30 ± 0.10	1.37 ± 0.11	1.43 ± 0.10	2.10 ± 0.09	2.40 ± 0.13
0.4	1.48 ± 0.06	1.26 ± 0.09	1.15 ± 0.12	1.05 ± 0.08	***0.92 ± 0.10***	1.16 ± 0.10	1.21 ± 0.10	1.36 ± 0.10	1.67 ± 0.12	2.05 ± 0.07	2.40 ± 0.13
0.5	1.60 ± 0.10	1.55 ± 0.08	1.48 ± 0.10	1.22 ± 0.09	1.03 ± 0.09	1.20 ± 0.15	1.30 ± 0.12	1.37 ± 0.09	1.69 ± 0.07	1.88 ± 0.06	2.40 ± 0.13
0.6	1.68 ± 0.13	1.62 ± 0.10	1.52 ± 0.07	1.27 ± 0.10	1.30 ± 0.08	1.32 ± 0.12	1.37 ± 0.09	1.43 ± 0.07	1.70 ± 0.09	1.94 ± 0.12	2.40 ± 0.13
0.7	1.77 ± 0.10	1.66 ± 0.11	1.60 ± 0.06	1.35 ± 0.07	1.40 ± 0.09	1.46 ± 0.14	1.49 ± 0.09	1.52 ± 0.12	1.73 ± 0.12	2.10 ± 0.13	2.40 ± 0.13
0.8	1.82 ± 0.14	1.71 ± 0.08	1.63 ± 0.15	1.42 ± 0.09	1.42 ± 0.07	1.50 ± 0.16	1.53 ± 0.10	1.55 ± 0.15	1.82 ± 0.10	2.15 ± 0.10	2.40 ± 0.13
0.9	1.84 ± 0.12	1.77 ± 0.15	1.64 ± 0.03	1.44 ± 0.11	1.45 ± 0.10	1.51 ± 0.10	1.60 ± 0.06	1.61 ± 0.12	1.84 ± 0.10	2.24 ± 0.07	2.40 ± 0.13
1	1.90 ± 0.11	1.82 ± 0.07	1.68 ± 0.07	1.53 ± 0.13	1.49 ± 0.12	1.55 ± 0.09	1.66 ± 0.08	1.67 ± 0.09	1.86 ± 0.09	2.13 ± 0.09	2.40 ± 0.13

Tabla B.6: Resultados (porcentaje de error ± desviación típica) de test para los valores de los distintos parámetros α y β del diseño con énfasis Completo y guía MLP para RECT. En cursiva, los valores elegidos por validación, y en negrita el mejor valor en test.

B.1.2. Énfasis Final guía MLP

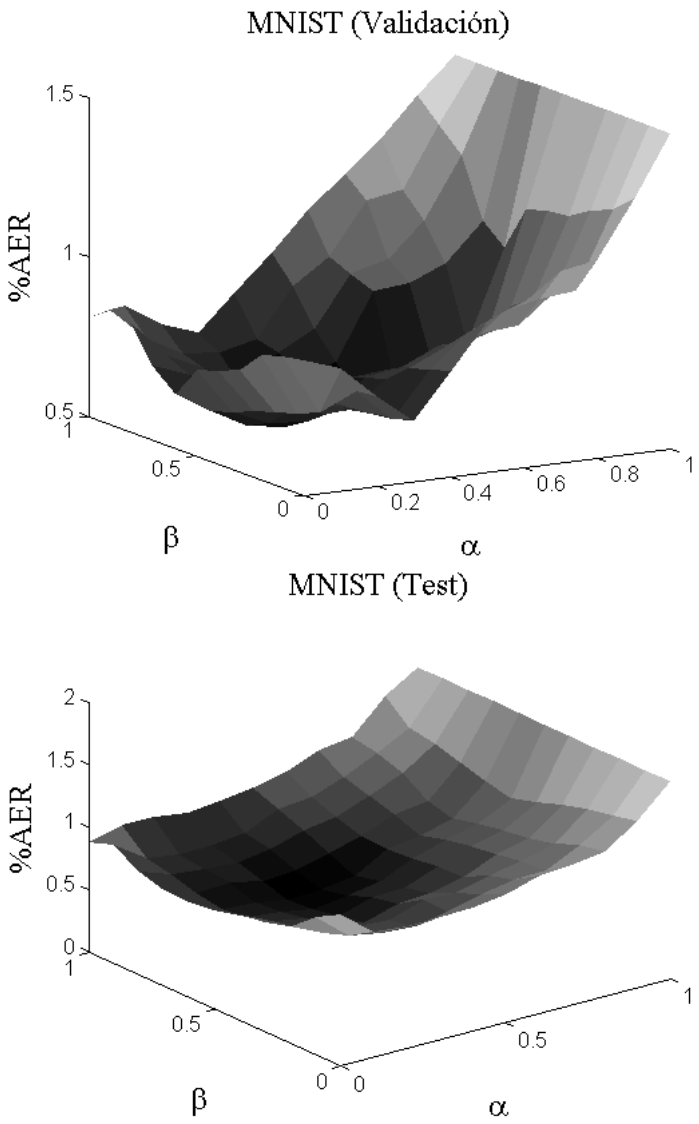

Figura B.4: Porcentaje de tasa de error promedio (Average Error Rate, AER) para los conjuntos de validación y test versus α, β, del problema MNIST, con guía MLP y énfasis Final.

β \ α	0	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9	1
0	0.92 ± 0.02	0.88 ± 0.02	0.78 ± 0.02	0.69 ± 0.01	0.83 ± 0.03	0.97 ± 0.04	1.02 ± 0.06	1.10 ± 0.03	1.12 ± 0.04	1.30 ± 0.03	1.49 ± 0.05
0.1	0.91 ± 0.03	0.72 ± 0.05	0.77 ± 0.02	0.67 ± 0.02	0.81 ± 0.02	0.91 ± 0.03	0.96 ± 0.05	0.99 ± 0.02	1.00 ± 0.04	1.24 ± 0.02	1.49 ± 0.05
0.2	0.89 ± 0.02	0.69 ± 0.01	0.71 ± 0.03	0.66 ± 0.01	0.79 ± 0.02	0.84 ± 0.03	0.87 ± 0.04	0.88 ± 0.02	0.97 ± 0.05	1.21 ± 0.04	1.49 ± 0.05
0.3	0.82 ± 0.03	0.67 ± 0.01	0.69 ± 0.05	0.65 ± 0.01	0.73 ± 0.03	0.75 ± 0.04	0.81 ± 0.01	0.84 ± 0.03	0.92 ± 0.03	1.17 ± 0.03	1.49 ± 0.05
0.4	0.79 ± 0.05	0.65 ± 0.06	0.64 ± 0.02	0.61 ± 0.04	0.71 ± 0.04	0.73 ± 0.06	0.78 ± 0.02	0.79 ± 0.01	0.88 ± 0.04	1.16 ± 0.02	1.49 ± 0.05
0.5	0.77 ± 0.02	0.62 ± 0.03	0.57 ± 0.02	0.56 ± 0.02	0.68 ± 0.01	0.69 ± 0.02	0.69 ± 0.04	0.72 ± 0.03	0.79 ± 0.05	1.14 ± 0.03	1.49 ± 0.05
0.6	0.67 ± 0.02	0.60 ± 0.01	0.54 ± 0.02	$\mathbf{0.52 \pm 0.01}$	0.55 ± 0.02	0.60 ± 0.03	0.62 ± 0.04	0.68 ± 0.02	0.71 ± 0.04	1.00 ± 0.02	1.49 ± 0.05
0.7	0.73 ± 0.02	0.69 ± 0.04	0.62 ± 0.04	0.55 ± 0.05	0.63 ± 0.07	0.67 ± 0.03	0.88 ± 0.00	0.89 ± 0.00	0.94 ± 0.02	1.06 ± 0.01	1.49 ± 0.05
0.8	0.84 ± 0.04	0.72 ± 0.06	0.67 ± 0.06	0.60 ± 0.01	0.69 ± 0.01	0.75 ± 0.06	0.91 ± 0.03	0.94 ± 0.00	1.05 ± 0.03	1.24 ± 0.04	1.49 ± 0.05
0.9	0.86 ± 0.02	0.75 ± 0.03	0.69 ± 0.05	0.68 ± 0.00	0.71 ± 0.00	0.86 ± 0.03	0.94 ± 0.02	1.10 ± 0.01	1.12 ± 0.04	1.31 ± 0.04	1.49 ± 0.05
1	0.81 ± 0.04	0.83 ± 0.02	0.76 ± 0.03	0.72 ± 0.01	0.83 ± 0.00	0.94 ± 0.01	1.06 ± 0.01	1.16 ± 0.02	1.26 ± 0.02	1.38 ± 0.01	1.49 ± 0.05

Tabla B.7: Resultados (porcentaje de error $\pm$ desviación típica) de validación para los valores de los distintos parámetros α y β del diseño con énfasis Final y guía MLP para MNIST. En negrita el mejor valor.

β \ α	0	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9	1
0	1.21 ± 0.05	1.01 ± 0.06	0.98 ± 0.04	0.98 ± 0.02	0.99 ± 0.04	1.02 ± 0.05	1.10 ± 0.05	1.12 ± 0.04	1.16 ± 0.02	1.33 ± 0.01	1.58 ± 0.06
0.1	1.10 ± 0.06	0.88 ± 0.05	0.83 ± 0.06	0.82 ± 0.02	0.86 ± 0.04	0.93 ± 0.06	0.98 ± 0.04	0.99 ± 0.04	1.12 ± 0.04	1.28 ± 0.02	1.58 ± 0.06
0.2	0.94 ± 0.02	0.81 ± 0.02	0.79 ± 0.07	0.78 ± 0.01	0.87 ± 0.02	0.85 ± 0.04	0.92 ± 0.06	0.97 ± 0.02	1.04 ± 0.01	1.24 ± 0.05	1.58 ± 0.06
0.3	0.88 ± 0.03	0.76 ± 0.04	0.71 ± 0.03	0.67 ± 0.03	0.74 ± 0.05	0.76 ± 0.04	0.82 ± 0.04	0.93 ± 0.04	0.99 ± 0.04	1.19 ± 0.03	1.58 ± 0.06
0.4	0.85 ± 0.05	0.72 ± 0.05	0.68 ± 0.04	0.63 ± 0.04	0.71 ± 0.03	0.71 ± 0.07	0.81 ± 0.05	0.90 ± 0.02	0.96 ± 0.02	1.12 ± 0.03	1.58 ± 0.06
0.5	0.79 ± 0.04	0.67 ± 0.05	0.64 ± 0.06	0.59 ± 0.00	0.63 ± 0.03	0.67 ± 0.05	0.79 ± 0.02	0.87 ± 0.04	0.92 ± 0.03	1.06 ± 0.05	1.58 ± 0.06
0.6	0.76 ± 0.04	0.64 ± 0.09	0.62 ± 0.08	***0.57 ± 0.00***	0.59 ± 0.02	0.63 ± 0.06	0.77 ± 0.03	0.81 ± 0.04	0.87 ± 0.02	1.10 ± 0.03	1.58 ± 0.06
0.7	0.77 ± 0.09	0.68 ± 0.06	0.69 ± 0.04	0.63 ± 0.06	0.65 ± 0.04	0.72 ± 0.07	0.78 ± 0.03	0.88 ± 0.01	0.92 ± 0.04	1.16 ± 0.02	1.58 ± 0.06
0.8	0.82 ± 0.08	0.73 ± 0.03	0.71 ± 0.04	0.69 ± 0.05	0.73 ± 0.01	0.81 ± 0.04	0.84 ± 0.01	0.94 ± 0.02	1.06 ± 0.00	1.22 ± 0.02	1.58 ± 0.06
0.9	0.95 ± 0.04	0.77 ± 0.04	0.73 ± 0.05	0.74 ± 0.03	0.76 ± 0.06	0.89 ± 0.05	0.91 ± 0.02	1.00 ± 0.01	1.12 ± 0.04	1.26 ± 0.03	1.58 ± 0.06
1	0.88 ± 0.10	0.95 ± 0.05	0.94 ± 0.05	0.90 ± 0.06	0.93 ± 0.05	0.97 ± 0.05	1.02 ± 0.04	1.12 ± 0.01	1.16 ± 0.04	1.42 ± 0.02	1.58 ± 0.06

Tabla B.8: Resultados (porcentaje de error $\pm$ desviación típica) de test para los valores de los distintos parámetros α y β del diseño con énfasis Final y guía MLP para MNIST. En cursiva, los valores elegidos por validación, y en negrita el mejor valor en test.

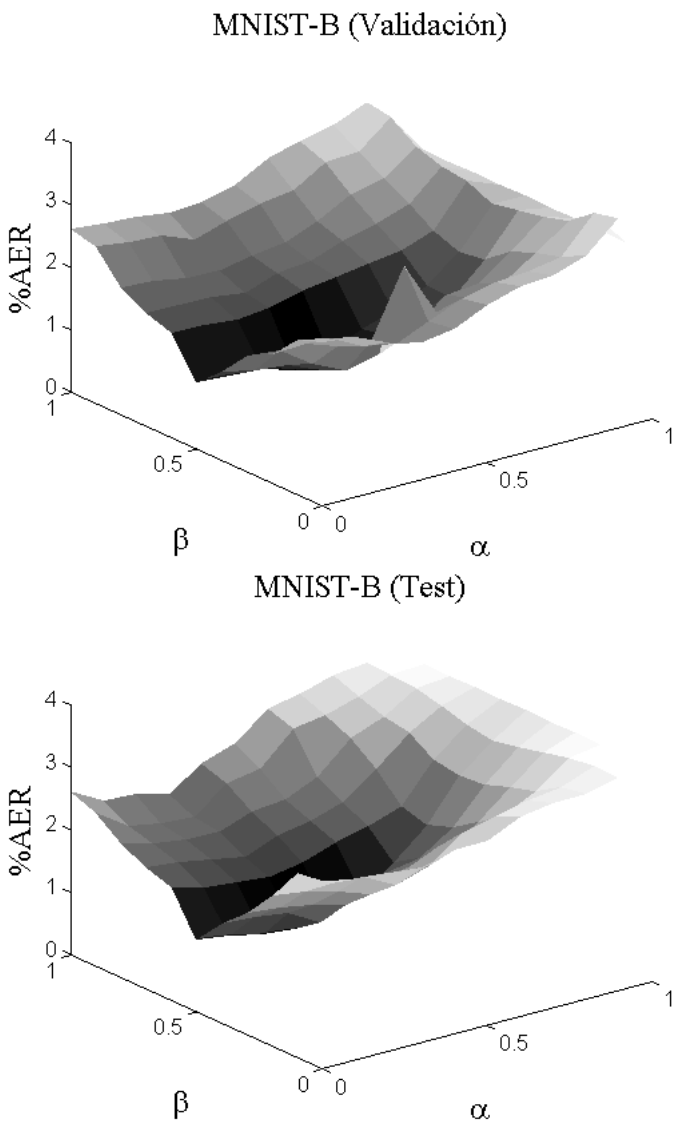

Figura B.5: Porcentaje de tasa de error promedio (Average Error Rate, AER) para los conjuntos de validación y test versus α, β, del problema MNIST-B, con guía MLP y énfasis Final.

β \ α	0	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9	1
0	2.62 ± 0.12	2.56 ± 0.07	2.35 ± 0.06	2.20 ± 0.06	2.28 ± 0.08	2.51 ± 0.06	2.67 ± 0.10	2.72 ± 0.12	2.90 ± 0.10	3.36 ± 0.06	2.65 ± 0.15
0.1	2.49 ± 0.08	2.37 ± 0.06	2.10 ± 0.05	2.00 ± 0.05	2.21 ± 0.06	2.44 ± 0.08	2.55 ± 0.09	2.60 ± 0.12	2.68 ± 0.07	3.25 ± 0.08	2.65 ± 0.15
0.2	2.07 ± 0.06	2.10 ± 0.09	1.92 ± 0.04	1.86 ± 0.05	2.94 ± 0.09	2.05 ± 0.07	2.14 ± 0.12	2.21 ± 0.15	2.56 ± 0.09	2.94 ± 0.06	2.65 ± 0.15
0.3	1.86 ± 0.10	1.80 ± 0.08	1.45 ± 0.06	1.20 ± 0.08	1.32 ± 0.10	1.55 ± 0.05	1.98 ± 0.06	2.04 ± 0.09	2.41 ± 0.10	2.82 ± 0.05	2.65 ± 0.15
0.4	1.41 ± 0.13	1.31 ± 0.05	1.24 ± 0.06	1.01 ± 0.09	1.10 ± 0.12	1.32 ± 0.09	1.47 ± 0.10	1.62 ± 0.10	2.10 ± 0.09	2.75 ± 0.04	2.65 ± 0.15
0.5	1.07 ± 0.08	1.04 ± 0.08	0.99 ± 0.06	$\mathbf{0.84 \pm 0.05}$	1.00 ± 0.03	1.21 ± 0.10	1.37 ± 0.06	1.64 ± 0.06	2.04 ± 0.11	2.66 ± 0.04	2.65 ± 0.15
0.6	1.67 ± 0.10	1.66 ± 0.03	1.60 ± 0.07	1.65 ± 0.03	1.80 ± 0.07	1.86 ± 0.12	1.94 ± 0.05	2.02 ± 0.10	2.20 ± 0.07	2.86 ± 0.08	2.65 ± 0.15
0.7	1.82 ± 0.10	1.80 ± 0.05	1.79 ± 0.09	1.72 ± 0.08	1.89 ± 0.10	1.94 ± 0.06	2.12 ± 0.06	2.27 ± 0.09	2.45 ± 0.03	2.91 ± 0.06	2.65 ± 0.15
0.8	2.04 ± 0.12	2.00 ± 0.06	1.90 ± 0.06	1.89 ± 0.06	1.98 ± 0.10	2.07 ± 0.10	2.26 ± 0.08	2.56 ± 0.07	2.66 ± 0.08	3.03 ± 0.09	2.65 ± 0.15
0.9	2.52 ± 0.06	2.44 ± 0.03	2.40 ± 0.08	2.21 ± 0.09	2.35 ± 0.11	2.50 ± 0.08	2.57 ± 0.06	2.89 ± 0.09	2.92 ± 0.06	3.33 ± 0.04	2.65 ± 0.15
1	2.61 ± 0.09	2.55 ± 0.10	2.52 ± 0.10	2.44 ± 0.06	2.48 ± 0.08	2.62 ± 0.09	2.90 ± 0.06	3.03 ± 0.06	3.15 ± 0.05	3.37 ± 0.05	2.65 ± 0.15

Tabla B.9: Resultados (porcentaje de error $\pm$ desviación típica) de validación para los valores de los distintos parámetros α y β del diseño con énfasis Final y guía MLP para MNIST-B. En negrita el mejor valor.

β \ α	0	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9	1
0	3.00 ± 0.04	2.90 ± 0.04	2.85 ± 0.09	2.80 ± 0.05	2.93 ± 0.02	2.94 ± 0.06	3.20 ± 0.10	3.26 ± 0.08	3.28 ± 0.06	3.40 ± 0.06	3.42 ± 0.10
0.1	2.95 ± 0.04	2.64 ± 0.05	2.60 ± 0.06	2.35 ± 0.06	2.49 ± 0.04	2.89 ± 0.06	3.00 ± 0.04	3.12 ± 0.07	3.20 ± 0.08	3.40 ± 0.05	3.42 ± 0.10
0.2	2.40 ± 0.08	2.32 ± 0.04	2.20 ± 0.07	2.00 ± 0.06	2.25 ± 0.00	2.47 ± 0.01	2.50 ± 0.01	2.68 ± 0.06	2.95 ± 0.06	3.35 ± 0.05	3.42 ± 0.10
0.3	1.95 ± 0.09	1.89 ± 0.05	1.80 ± 0.08	1.66 ± 0.05	1.94 ± 0.10	2.00 ± 0.06	2.08 ± 0.03	2.42 ± 0.06	2.84 ± 0.10	3.26 ± 0.05	3.42 ± 0.10
0.4	1.60 ± 0.04	1.56 ± 0.06	1.49 ± 0.06	1.20 ± 0.06	1.60 ± 0.10	1.71 ± 0.10	1.80 ± 0.08	2.15 ± 0.10	2.70 ± 0.08	3.15 ± 0.04	3.42 ± 0.10
0.5	1.15 ± 0.09	1.07 ± 0.07	0.96 ± 0.06	***0.91 ± 0.03***	0.99 ± 0.06	1.20 ± 0.08	1.54 ± 0.06	2.00 ± 0.08	2.65 ± 0.12	3.08 ± 0.06	3.42 ± 0.10
0.6	1.80 ± 0.07	1.66 ± 0.04	1.60 ± 0.04	1.56 ± 0.04	1.65 ± 0.08	1.82 ± 0.09	1.94 ± 0.08	2.12 ± 0.06	2.73 ± 0.06	3.16 ± 0.08	3.42 ± 0.10
0.7	1.94 ± 0.04	1.89 ± 0.03	1.79 ± 0.08	1.70 ± 0.05	1.89 ± 0.09	1.99 ± 0.06	2.15 ± 0.08	2.54 ± 0.09	2.97 ± 0.07	3.26 ± 0.10	3.42 ± 0.10
0.8	2.15 ± 0.07	2.00 ± 0.10	1.93 ± 0.10	1.92 ± 0.03	2.10 ± 0.10	2.15 ± 0.10	2.36 ± 0.10	2.85 ± 0.10	3.05 ± 0.09	3.37 ± 0.10	3.42 ± 0.10
0.9	2.45 ± 0.04	2.10 ± 0.12	2.01 ± 0.10	1.96 ± 0.10	2.35 ± 0.03	2.46 ± 0.12	2.95 ± 0.12	3.03 ± 0.09	3.20 ± 0.08	3.40 ± 0.06	3.42 ± 0.10
1	2.62 ± 0.05	2.32 ± 0.10	2.29 ± 0.10	2.15 ± 0.06	2.50 ± 0.10	2.67 ± 0.09	3.10 ± 0.09	3.18 ± 0.10	3.35 ± 0.08	3.41 ± 0.04	3.42 ± 0.10

Tabla B.10: Resultados (porcentaje de error ± desviación típica) de test para los valores de los distintos parámetros α y β del diseño con énfasis Final y guía MLP para MNIST-B. En cursiva, los valores elegidos por validación, y en negrita el mejor valor en test.

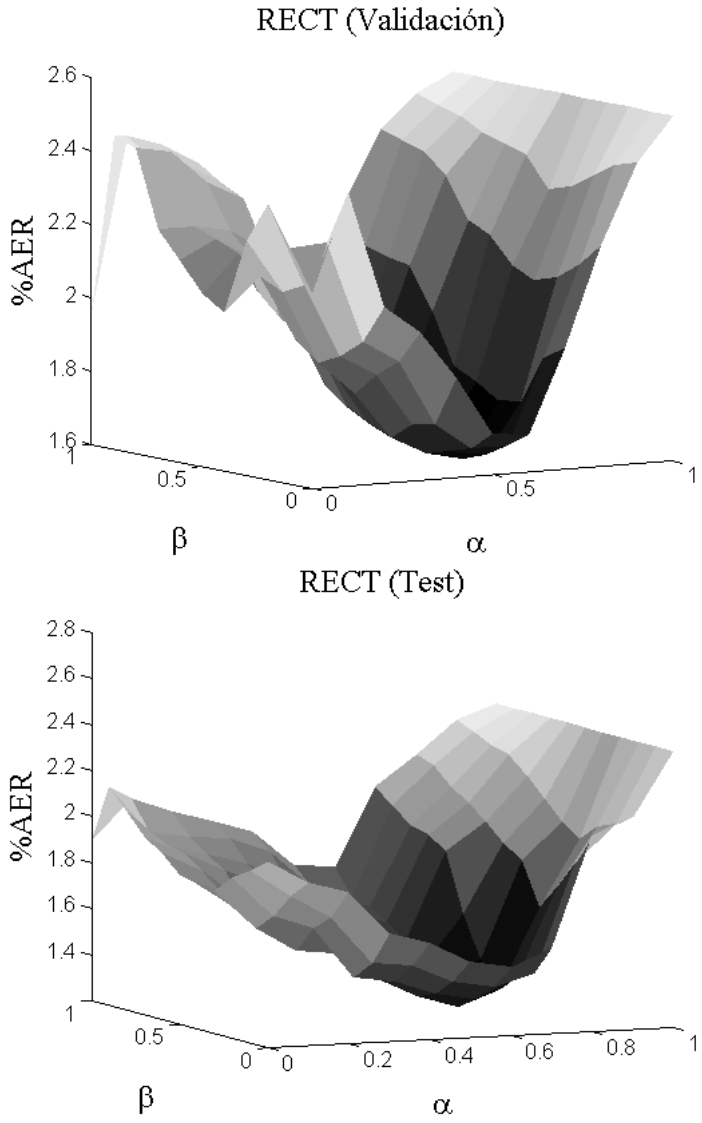

Figura B.6: Porcentaje de tasa de error promedio (Average Error Rate, AER) para los conjuntos de validación y test versus α, β, del problema RECT, con guía MLP y énfasis Final.

β \ α	0	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9	1
0	2.13 ± 0.12	2.40 ± 0.12	2.08 ± 0.10	2.00 ± 0.15	1.87 ± 0.08	1.71 ± 0.07	1.70 ± 0.10	1.93 ± 0.11	2.20 ± 0.15	2.42 ± 0.10	2.54 ± 0.22
0.1	2.24 ± 0.17	2.25 ± 0.15	1.97 ± 0.10	1.91 ± 0.09	1.73 ± 0.10	1.66 ± 0.09	1.66 ± 0.12	1.89 ± 0.15	2.14 ± 0.14	2.40 ± 0.10	2.54 ± 0.22
0.2	2.35 ± 0.10	2.12 ± 0.10	1.92 ± 0.12	1.86 ± 0.06	1.72 ± 0.09	1.64 ± 0.05	1.62 ± 0.08	1.76 ± 0.12	2.09 ± 0.13	2.36 ± 0.15	2.54 ± 0.22
0.3	2.24 ± 0.15	2.09 ± 0.12	1.89 ± 0.10	1.80 ± 0.08	1.67 ± 0.07	1.62 ± 0.07	$\mathbf{1.60 \pm 0.06}$	1.75 ± 0.05	2.07 ± 0.14	2.32 ± 0.14	2.54 ± 0.22
0.4	2.03 ± 0.12	2.12 ± 0.10	1.94 ± 0.09	1.85 ± 0.12	1.70 ± 0.10	1.68 ± 0.06	1.65 ± 0.06	1.79 ± 0.06	2.10 ± 0.08	2.30 ± 0.12	2.54 ± 0.22
0.5	2.07 ± 0.09	2.17 ± 0.12	2.05 ± 0.12	1.94 ± 0.15	1.73 ± 0.12	1.71 ± 0.06	1.68 ± 0.03	1.82 ± 0.04	2.17 ± 0.09	2.38 ± 0.09	2.54 ± 0.22
0.6	2.15 ± 0.10	2.20 ± 0.08	2.14 ± 0.15	1.99 ± 0.10	1.78 ± 0.08	1.73 ± 0.06	1.71 ± 0.10	1.96 ± 0.04	2.20 ± 0.09	2.40 ± 0.10	2.54 ± 0.22
0.7	2.22 ± 0.15	2.22 ± 0.12	2.20 ± 0.18	2.14 ± 0.12	1.90 ± 0.10	1.86 ± 0.04	1.80 ± 0.09	2.07 ± 0.05	2.31 ± 0.08	2.42 ± 0.10	2.54 ± 0.22
0.8	2.43 ± 0.13	2.41 ± 0.09	2.32 ± 0.12	2.27 ± 0.06	1.94 ± 0.06	1.88 ± 0.02	1.82 ± 0.04	2.08 ± 0.09	2.36 ± 0.12	2.45 ± 0.10	2.54 ± 0.22
0.9	2.45 ± 0.10	2.39 ± 0.07	2.36 ± 0.16	2.30 ± 0.20	2.00 ± 0.18	1.92 ± 0.08	1.89 ± 0.07	2.10 ± 0.10	2.38 ± 0.08	2.47 ± 0.08	2.54 ± 0.22
1	1.96 ± 0.10	2.43 ± 0.08	2.40 ± 0.12	2.34 ± 0.21	2.10 ± 0.13	2.00 ± 0.10	1.93 ± 0.07	2.21 ± 0.10	2.40 ± 0.10	2.49 ± 0.09	2.54 ± 0.22

Tabla B.11: Resultados (porcentaje de error $\pm$ desviación típica) de validación para los valores de los distintos parámetros α y β del diseño con énfasis Final y guía MLP para RECT. En negrita el mejor valor.

β \ α	0	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9	1
0	2.01 ± 0.08	1.91 ± 0.10	1.86 ± 0.06	1.67 ± 0.07	1.62 ± 0.06	1.53 ± 0.06	1.50 ± 0.06	1.57 ± 0.10	2.05 ± 0.10	2.12 ± 0.08	2.40 ± 0.13
0.1	1.95 ± 0.06	1.83 ± 0.10	1.73 ± 0.08	1.52 ± 0.06	1.50 ± 0.10	1.42 ± 0.05	1.39 ± 0.04	1.44 ± 0.10	1.91 ± 0.09	2.07 ± 0.06	2.40 ± 0.13
0.2	1.90 ± 0.07	1.72 ± 0.04	1.66 ± 0.06	1.44 ± 0.06	1.41 ± 0.07	1.35 ± 0.04	1.32 ± 0.05	1.40 ± 0.08	1.79 ± 0.10	2.04 ± 0.10	2.40 ± 0.13
0.3	1.82 ± 0.07	1.65 ± 0.06	1.54 ± 0.10	1.52 ± 0.12	1.40 ± 0.06	1.32 ± 0.05	***1.26 ± 0.04***	1.33 ± 0.06	1.68 ± 0.08	1.97 ± 0.09	2.40 ± 0.13
0.4	1.83 ± 0.04	1.71 ± 0.06	1.57 ± 0.10	1.57 ± 0.08	1.52 ± 0.10	1.38 ± 0.05	1.33 ± 0.04	1.44 ± 0.05	1.86 ± 0.08	2.02 ± 0.10	2.40 ± 0.13
0.5	1.85 ± 0.10	1.77 ± 0.08	1.64 ± 0.08	1.60 ± 0.06	1.68 ± 0.12	1.44 ± 0.06	1.40 ± 0.04	1.62 ± 0.06	1.92 ± 0.10	2.10 ± 0.08	2.40 ± 0.13
0.6	1.92 ± 0.06	1.84 ± 0.06	1.70 ± 0.06	1.66 ± 0.10	1.60 ± 0.08	1.52 ± 0.07	1.49 ± 0.08	1.87 ± 0.05	1.99 ± 0.06	2.18 ± 0.07	2.40 ± 0.13
0.7	1.97 ± 0.05	1.89 ± 0.05	1.81 ± 0.04	1.74 ± 0.09	1.62 ± 0.09	1.58 ± 0.06	1.55 ± 0.08	1.92 ± 0.10	2.05 ± 0.08	2.25 ± 0.09	2.40 ± 0.13
0.8	2.07 ± 0.08	1.91 ± 0.05	1.86 ± 0.07	1.81 ± 0.07	1.71 ± 0.08	1.61 ± 0.08	1.60 ± 0.06	1.94 ± 0.09	2.10 ± 0.09	2.27 ± 0.10	2.40 ± 0.13
0.9	2.15 ± 0.10	1.94 ± 0.05	1.91 ± 0.06	1.90 ± 0.06	1.83 ± 0.09	1.67 ± 0.08	1.62 ± 0.09	2.01 ± 0.08	2.17 ± 0.10	2.31 ± 0.10	2.40 ± 0.13
1	1.90 ± 0.15	2.07 ± 0.07	2.00 ± 0.08	1.95 ± 0.08	1.90 ± 0.10	1.74 ± 0.06	1.73 ± 0.09	2.07 ± 0.10	2.20 ± 0.12	2.34 ± 0.12	2.40 ± 0.13

Tabla B.12: Resultados (porcentaje de error ± desviación típica) de test para los valores de los distintos parámetros α y β del diseño con énfasis Final y guía MLP para RECT. En cursiva, los valores elegidos por validación, y en negrita el mejor valor en test.

B.2. Guía SDAE3

B.2.1. Énfasis Completo guía SDAE3

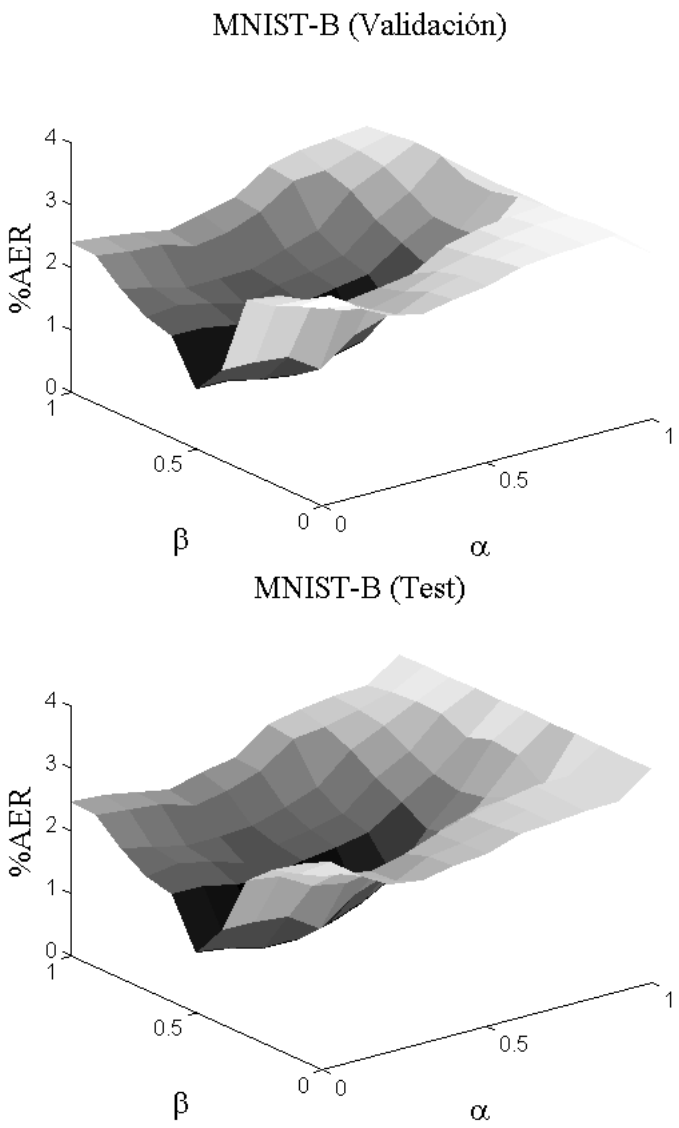

Figura B.7: Porcentaje de tasa de error promedio (Average Error Rate, AER) para los conjuntos de validación y test versus α, β, del problema MNIST-B, con guía SDAE3 y énfasis Completo.

β \ α	0	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9	1
0	3.15 ± 0.01	3.08 ± 0.01	2.77 ± 0.01	2.66 ± 0.01	2.75 ± 0.00	2.81 ± 0.01	2.98 ± 0.04	3.03 ± 0.01	3.07 ± 0.01	3.09 ± 0.04	2.65 ± 0.15
0.1	3.02 ± 0.00	3.05 ± 0.03	2.69 ± 0.04	2.47 ± 0.01	2.69 ± 0.00	2.80 ± 0.03	2.93 ± 0.02	3.02 ± 0.01	3.05 ± 0.02	3.08 ± 0.01	2.65 ± 0.15
0.2	2.84 ± 0.00	2.80 ± 0.01	2.49 ± 0.04	2.38 ± 0.02	2.50 ± 0.01	2.69 ± 0.00	2.82 ± 0.01	2.95 ± 0.02	3.02 ± 0.01	3.03 ± 0.02	2.65 ± 0.15
0.3	2.75 ± 0.03	2.68 ± 0.01	2.42 ± 0.02	2.31 ± 0.01	2.38 ± 0.01	2.46 ± 0.02	2.78 ± 0.00	2.91 ± 0.01	2.97 ± 0.01	3.02 ± 0.01	2.65 ± 0.15
0.4	1.44 ± 0.02	1.42 ± 0.00	1.37 ± 0.01	1.03 ± 0.03	1.43 ± 0.02	1.60 ± 0.02	1.77 ± 0.03	1.98 ± 0.01	2.45 ± 0.01	2.95 ± 0.02	2.65 ± 0.15
0.5	0.95 ± 0.01	0.94 ± 0.00	0.82 ± 0.01	$\mathbf{0.74 \pm 0.00}$	0.89 ± 0.02	1.00 ± 0.01	1.42 ± 0.02	1.97 ± 0.01	2.40 ± 0.01	2.86 ± 0.01	2.65 ± 0.15
0.6	1.64 ± 0.02	1.62 ± 0.00	1.52 ± 0.00	1.51 ± 0.02	1.61 ± 0.02	1.69 ± 0.00	1.78 ± 0.02	2.14 ± 0.01	2.42 ± 0.01	2.88 ± 0.01	2.65 ± 0.15
0.7	1.74 ± 0.02	1.80 ± 0.01	1.68 ± 0.02	1.55 ± 0.04	1.70 ± 0.03	1.77 ± 0.00	1.89 ± 0.03	2.33 ± 0.02	2.63 ± 0.01	3.07 ± 0.02	2.65 ± 0.15
0.8	2.01 ± 0.02	1.90 ± 0.01	1.76 ± 0.00	1.76 ± 0.02	1.78 ± 0.02	1.95 ± 0.00	2.12 ± 0.04	2.61 ± 0.02	2.87 ± 0.01	3.11 ± 0.02	2.65 ± 0.15
0.9	2.37 ± 0.01	2.30 ± 0.02	2.25 ± 0.00	2.10 ± 0.02	2.16 ± 0.01	2.37 ± 0.01	2.72 ± 0.03	2.76 ± 0.01	2.93 ± 0.00	3.06 ± 0.00	2.65 ± 0.15
1	2.40 ± 0.01	2.34 ± 0.02	2.27 ± 0.01	2.16 ± 0.02	2.29 ± 0.01	2.42 ± 0.01	2.78 ± 0.03	2.89 ± 0.00	2.95 ± 0.00	3.00 ± 0.01	2.65 ± 0.15

Tabla B.13: Resultados (porcentaje de error $\pm$ desviación típica) de validación para los valores de los distintos parámetros α y β del diseño con énfasis Completo y guía SDAE3 para MNIST-B. En negrita el mejor valor.

β \ α	0	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9	1
0	3.09 ± 0.01	3.02 ± 0.01	2.69 ± 0.02	2.60 ± 0.02	2.71 ± 0.01	2.76 ± 0.01	2.94 ± 0.00	2.97 ± 0.03	3.02 ± 0.01	3.03 ± 0.04	3.42 ± 0.10
0.1	2.96 ± 0.02	3.01 ± 0.02	2.62 ± 0.00	2.42 ± 0.03	2.66 ± 0.02	2.74 ± 0.01	2.89 ± 0.02	2.99 ± 0.03	3.02 ± 0.01	3.03 ± 0.02	3.42 ± 0.10
0.2	2.84 ± 0.02	2.81 ± 0.02	2.50 ± 0.00	2.39 ± 0.01	2.53 ± 0.02	2.71 ± 0.01	2.84 ± 0.00	2.97 ± 0.00	3.05 ± 0.01	3.06 ± 0.03	3.42 ± 0.10
0.3	2.47 ± 0.03	2.41 ± 0.02	2.12 ± 0.03	2.01 ± 0.01	2.10 ± 0.03	2.18 ± 0.01	2.50 ± 0.02	2.65 ± 0.03	2.69 ± 0.01	2.74 ± 0.01	3.42 ± 0.10
0.4	1.50 ± 0.02	1.50 ± 0.00	1.44 ± 0.00	1.10 ± 0.01	1.50 ± 0.03	1.67 ± 0.03	1.84 ± 0.03	2.05 ± 0.01	2.51 ± 0.00	3.04 ± 0.00	3.42 ± 0.10
0.5	0.95 ± 0.03	0.89 ± 0.00	0.75 ± 0.02	***0.72 ± 0.01***	0.85 ± 0.02	1.05 ± 0.02	1.34 ± 0.03	1.97 ± 0.01	2.42 ± 0.00	3.10 ± 0.00	3.42 ± 0.10
0.6	1.71 ± 0.02	1.71 ± 0.00	1.58 ± 0.01	1.58 ± 0.00	1.70 ± 0.00	1.76 ± 0.00	1.85 ± 0.02	2.22 ± 0.01	2.51 ± 0.03	2.95 ± 0.01	3.42 ± 0.10
0.7	1.82 ± 0.02	1.88 ± 0.00	1.76 ± 0.01	1.60 ± 0.00	1.77 ± 0.02	1.85 ± 0.00	1.97 ± 0.02	2.41 ± 0.01	2.70 ± 0.02	3.15 ± 0.01	3.42 ± 0.10
0.8	2.08 ± 0.00	1.96 ± 0.03	1.83 ± 0.01	1.84 ± 0.00	1.87 ± 0.00	2.03 ± 0.00	2.21 ± 0.00	2.68 ± 0.01	2.93 ± 0.03	3.17 ± 0.01	3.42 ± 0.10
0.9	2.43 ± 0.03	2.38 ± 0.02	2.31 ± 0.03	2.15 ± 0.02	2.25 ± 0.00	2.44 ± 0.03	2.80 ± 0.04	2.85 ± 0.01	2.99 ± 0.04	3.13 ± 0.02	3.42 ± 0.10
1	2.46 ± 0.03	2.42 ± 0.02	2.32 ± 0.01	2.24 ± 0.02	2.38 ± 0.01	2.47 ± 0.02	2.87 ± 0.02	2.96 ± 0.01	3.03 ± 0.01	3.06 ± 0.03	3.42 ± 0.10

Tabla B.14: Resultados (porcentaje de error ± desviación típica) de test para los valores de los distintos parámetros α y β del diseño con énfasis Completo y guía SDAE3 para MNIST-B. En cursiva, los valores elegidos por validación, y en negrita el mejor valor en test.

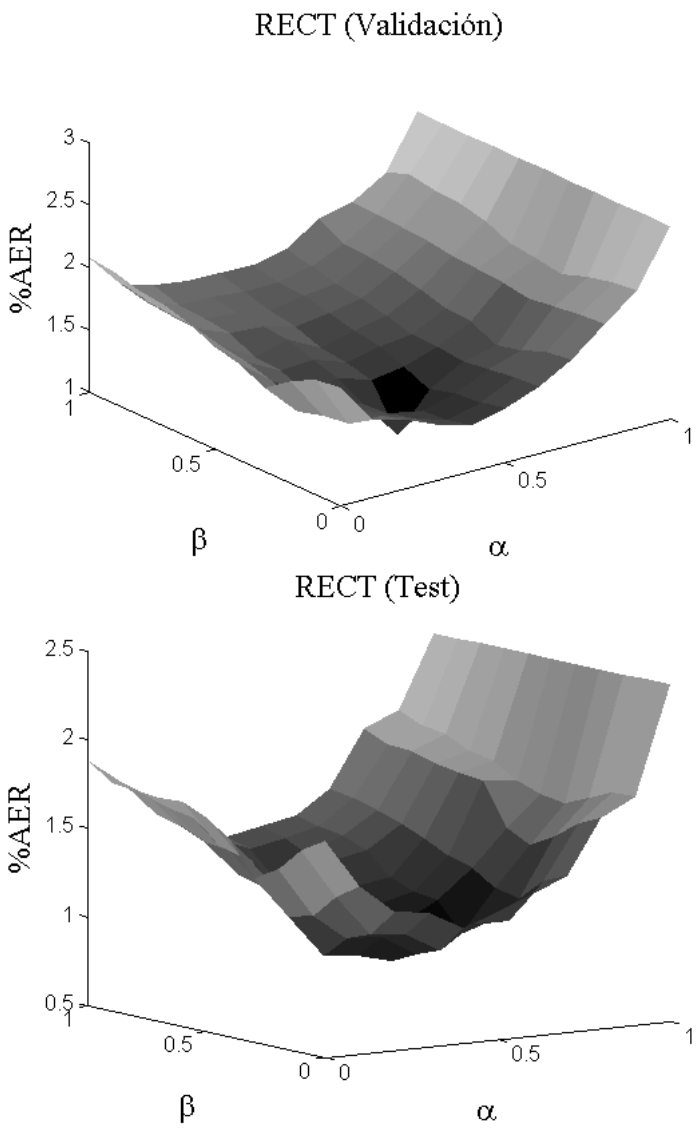

Figura B.8: Porcentaje de tasa de error promedio (Average Error Rate, AER) para los conjuntos de validación y test versus α, β, del problema RECT, con guía SDAE3 y énfasis Completo.

β \ α	0	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9	1
0	1.95 ± 0.08	1.62 ± 0.10	1.55 ± 0.12	1.44 ± 0.08	1.37 ± 0.10	1.44 ± 0.08	1.54 ± 0.06	1.68 ± 0.14	1.90 ± 0.12	2.10 ± 0.06	2.54 ± 0.22
0.1	1.92 ± 0.10	1.50 ± 0.12	1.49 ± 0.09	1.40 ± 0.09	1.32 ± 0.06	1.39 ± 0.10	1.52 ± 0.12	1.65 ± 0.12	1.88 ± 0.10	2.07 ± 0.12	2.54 ± 0.22
0.2	1.82 ± 0.12	1.47 ± 0.09	1.45 ± 0.09	1.35 ± 0.07	1.30 ± 0.08	1.37 ± 0.12	1.49 ± 0.09	1.57 ± 0.08	1.85 ± 0.12	2.02 ± 0.11	2.54 ± 0.22
0.3	1.67 ± 0.10	1.41 ± 0.08	1.38 ± 0.07	1.33 ± 0.09	$\mathbf{1.02 \pm 0.09}$	1.31 ± 0.12	1.47 ± 0.06	1.52 ± 0.07	1.83 ± 0.09	1.95 ± 0.13	2.54 ± 0.22
0.4	1.77 ± 0.10	1.56 ± 0.09	1.41 ± 0.11	1.45 ± 0.10	1.36 ± 0.10	1.40 ± 0.07	1.49 ± 0.13	1.62 ± 0.08	1.87 ± 0.07	1.99 ± 0.05	2.54 ± 0.22
0.5	1.82 ± 0.07	1.62 ± 0.10	1.49 ± 0.07	1.50 ± 0.08	1.41 ± 0.09	1.47 ± 0.06	1.50 ± 0.12	1.64 ± 0.12	1.88 ± 0.06	2.01 ± 0.06	2.54 ± 0.22
0.6	1.91 ± 0.06	1.66 ± 0.07	1.50 ± 0.08	1.57 ± 0.12	1.44 ± 0.10	1.51 ± 0.12	1.55 ± 0.10	1.70 ± 0.10	1.90 ± 0.12	2.10 ± 0.09	2.54 ± 0.22
0.7	1.97 ± 0.09	1.71 ± 0.08	1.61 ± 0.09	1.59 ± 0.10	1.52 ± 0.10	1.55 ± 0.08	1.60 ± 0.15	1.75 ± 0.09	1.92 ± 0.10	2.15 ± 0.08	2.54 ± 0.22
0.8	2.00 ± 0.10	1.72 ± 0.10	1.63 ± 0.12	1.60 ± 0.09	1.55 ± 0.07	1.60 ± 0.07	1.63 ± 0.12	1.77 ± 0.11	1.94 ± 0.07	2.17 ± 0.12	2.54 ± 0.22
0.9	2.05 ± 0.15	1.74 ± 0.12	1.65 ± 0.13	1.64 ± 0.10	1.61 ± 0.06	1.62 ± 0.09	1.68 ± 0.10	1.79 ± 0.09	1.95 ± 0.10	2.20 ± 0.15	2.54 ± 0.22
1	2.08 ± 0.09	1.79 ± 0.07	1.72 ± 0.10	1.68 ± 0.13	1.67 ± 0.09	1.67 ± 0.12	1.72 ± 0.10	1.90 ± 0.13	1.96 ± 0.12	2.12 ± 0.10	2.54 ± 0.22

Tabla B.15: Resultados (porcentaje de error $\pm$ desviación típica) de validación para los valores de los distintos parámetros α y β del diseño con énfasis Completo y guía SDAE3 para RECT. En negrita el mejor valor.

β \ α	0	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9	1
0	1.72 ± 0.07	1.44 ± 0.12	1.28 ± 0.09	1.25 ± 0.10	1.15 ± 0.08	1.22 ± 0.07	1.30 ± 0.09	1.38 ± 0.10	1.69 ± 0.06	1.79 ± 0.06	2.40 ± 0.13
0.1	1.64 ± 0.08	1.35 ± 0.09	1.12 ± 0.08	1.10 ± 0.08	0.99 ± 0.06	1.12 ± 0.12	1.12 ± 0.10	1.32 ± 0.08	1.63 ± 0.05	1.78 ± 0.06	2.40 ± 0.13
0.2	1.50 ± 0.06	1.22 ± 0.12	1.07 ± 0.06	1.06 ± 0.12	0.94 ± 0.09	1.01 ± 0.09	1.07 ± 0.09	1.20 ± 0.08	1.52 ± 0.07	1.72 ± 0.10	2.40 ± 0.13
0.3	1.42 ± 0.10	1.18 ± 0.09	0.94 ± 0.07	0.92 ± 0.05	***0.87 ± 0.04***	0.94 ± 0.06	0.99 ± 0.10	1.23 ± 0.10	1.45 ± 0.06	1.66 ± 0.05	2.40 ± 0.13
0.4	1.61 ± 0.10	1.39 ± 0.10	1.11 ± 0.12	1.10 ± 0.07	0.96 ± 0.05	1.04 ± 0.03	1.15 ± 0.08	1.31 ± 0.09	1.52 ± 0.10	1.69 ± 0.06	2.40 ± 0.13
0.5	1.70 ± 0.02	1.46 ± 0.07	1.21 ± 0.10	1.05 ± 0.06	0.98 ± 0.07	1.12 ± 0.08	1.17 ± 0.12	1.35 ± 0.10	1.75 ± 0.10	1.73 ± 0.09	2.40 ± 0.13
0.6	1.76 ± 0.08	1.50 ± 0.09	1.40 ± 0.05	1.17 ± 0.09	1.05 ± 0.12	1.19 ± 0.10	1.22 ± 0.09	1.42 ± 0.04	1.77 ± 0.12	1.82 ± 0.08	2.40 ± 0.13
0.7	1.76 ± 0.05	1.62 ± 0.06	1.42 ± 0.06	1.20 ± 0.07	1.18 ± 0.08	1.30 ± 0.12	1.34 ± 0.10	1.47 ± 0.13	1.79 ± 0.09	1.84 ± 0.06	2.40 ± 0.13
0.8	1.77 ± 0.07	1.65 ± 0.06	1.44 ± 0.09	1.23 ± 0.10	1.20 ± 0.13	1.31 ± 0.13	1.40 ± 0.10	1.52 ± 0.12	1.83 ± 0.07	1.91 ± 0.04	2.40 ± 0.13
0.9	1.78 ± 0.06	1.70 ± 0.10	1.50 ± 0.11	1.41 ± 0.12	1.33 ± 0.10	1.37 ± 0.09	1.42 ± 0.09	1.56 ± 0.10	1.84 ± 0.06	1.93 ± 0.08	2.40 ± 0.13
1	1.88 ± 0.10	1.73 ± 0.09	1.53 ± 0.10	1.50 ± 0.12	1.37 ± 0.07	1.41 ± 0.09	1.45 ± 0.08	1.58 ± 0.09	1.90 ± 0.08	1.99 ± 0.10	2.40 ± 0.13

Tabla B.16: Resultados (porcentaje de error ± desviación típica) de test para los valores de los distintos parámetros α y β del diseño con énfasis Completo y guía SDAE3 para RECT. En cursiva, los valores elegidos por validación, y en negrita el mejor valor en test.

B.2.2. Énfasis Final guía SDAE3

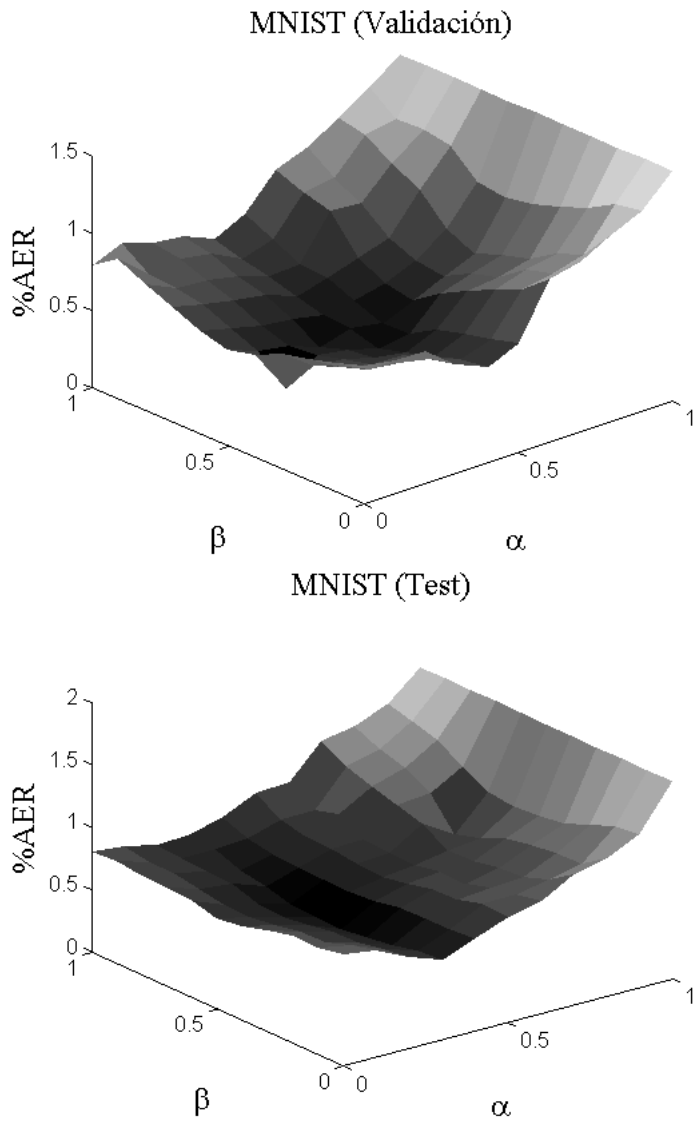

Figura B.9: Porcentaje de tasa de error promedio (Average Error Rate, AER) para los conjuntos de validación y test versus α, β, del problema MNIST, con guía SDAE3 y énfasis Final.

β \ α	0	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9	1
0	0.90 ± 0.04	0.86 ± 0.02	0.86 ± 0.02	0.70 ± 0.03	0.62 ± 0.02	0.71 ± 0.03	1.06 ± 0.03	1.10 ± 0.04	1.21 ± 0.02	1.30 ± 0.05	1.49 ± 0.05
0.1	0.84 ± 0.04	0.73 ± 0.05	0.70 ± 0.02	0.66 ± 0.06	0.60 ± 0.02	0.66 ± 0.04	0.91 ± 0.05	0.95 ± 0.03	1.00 ± 0.02	1.27 ± 0.01	1.49 ± 0.05
0.2	0.79 ± 0.01	0.67 ± 0.05	0.62 ± 0.02	0.61 ± 0.08	0.54 ± 0.04	0.62 ± 0.04	0.84 ± 0.06	0.90 ± 0.04	0.94 ± 0.04	1.16 ± 0.02	1.49 ± 0.05
0.3	0.75 ± 0.02	0.60 ± 0.03	0.58 ± 0.03	0.52 ± 0.06	0.49 ± 0.03	0.57 ± 0.03	0.75 ± 0.04	0.77 ± 0.05	0.83 ± 0.03	1.10 ± 0.02	1.49 ± 0.05
0.4	0.66 ± 0.04	0.38 ± 0.02	0.53 ± 0.06	0.50 ± 0.04	**0.44 ± 0.05**	0.47 ± 0.03	0.66 ± 0.06	0.72 ± 0.03	0.78 ± 0.02	1.04 ± 0.01	1.49 ± 0.05
0.5	0.62 ± 0.04	0.55 ± 0.04	0.51 ± 0.04	**0.44 ± 0.06**	0.49 ± 0.03	0.46 ± 0.03	0.54 ± 0.04	0.63 ± 0.04	0.71 ± 0.04	1.01 ± 0.02	1.49 ± 0.05
0.6	0.67 ± 0.03	0.62 ± 0.03	0.57 ± 0.05	0.54 ± 0.05	0.50 ± 0.03	0.53 ± 0.02	0.58 ± 0.05	0.66 ± 0.03	0.80 ± 0.03	1.03 ± 0.04	1.49 ± 0.05
0.7	0.75 ± 0.02	0.70 ± 0.04	0.66 ± 0.05	0.62 ± 0.02	0.55 ± 0.05	0.60 ± 0.04	0.67 ± 0.03	0.73 ± 0.01	0.97 ± 0.03	1.19 ± 0.04	1.49 ± 0.05
0.8	0.82 ± 0.03	0.73 ± 0.03	0.71 ± 0.05	0.67 ± 0.02	0.60 ± 0.06	0.61 ± 0.02	0.89 ± 0.01	0.88 ± 0.04	1.07 ± 0.04	1.22 ± 0.03	1.49 ± 0.05
0.9	0.91 ± 0.02	0.74 ± 0.06	0.77 ± 0.02	0.74 ± 0.03	0.64 ± 0.04	0.70 ± 0.02	0.94 ± 0.04	0.97 ± 0.01	1.18 ± 0.03	1.21 ± 0.01	1.49 ± 0.05
1	0.79 ± 0.08	0.87 ± 0.05	0.80 ± 0.05	0.78 ± 0.04	0.69 ± 0.02	0.79 ± 0.04	1.01 ± 0.00	1.09 ± 0.05	1.21 ± 0.04	1.35 ± 0.03	1.49 ± 0.05

Tabla B.17: Resultados (porcentaje de error ± desviación típica) de validación para los valores de los distintos parámetros α y β del diseño con énfasis Final y guía SDAE3 para MNIST. En negrita el mejor valor.

β \ α	0	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9	1
0	0.89 ± 0.04	0.86 ± 0.04	0.73 ± 0.02	0.64 ± 0.04	0.74 ± 0.02	0.84 ± 0.04	0.90 ± 0.06	1.04 ± 0.02	1.10 ± 0.02	1.22 ± 0.04	1.58 ± 0.06
0.1	0.86 ± 0.06	0.81 ± 0.04	0.70 ± 0.03	0.62 ± 0.02	0.72 ± 0.03	0.86 ± 0.02	0.86 ± 0.05	0.91 ± 0.02	0.97 ± 0.04	1.20 ± 0.03	1.58 ± 0.06
0.2	0.86 ± 0.02	0.74 ± 0.02	0.67 ± 0.05	0.58 ± 0.04	0.72 ± 0.03	0.80 ± 0.03	0.82 ± 0.05	0.84 ± 0.03	0.96 ± 0.04	1.12 ± 0.02	1.58 ± 0.06
0.3	0.81 ± 0.04	0.73 ± 0.03	0.66 ± 0.05	0.55 ± 0.02	0.71 ± 0.04	0.79 ± 0.04	0.77 ± 0.06	0.82 ± 0.03	0.94 ± 0.03	1.04 ± 0.05	1.58 ± 0.06
0.4	0.75 ± 0.02	0.69 ± 0.04	0.62 ± 0.03	$\mathbf{0.52 \pm 0.02}$	*0.67 ± 0.05*	0.74 ± 0.02	0.77 ± 0.02	0.79 ± 0.01	0.90 ± 0.03	1.01 ± 0.04	1.58 ± 0.06
0.5	0.73 ± 0.03	0.67 ± 0.05	0.60 ± 0.05	***0.52 ± 0.01***	0.66 ± 0.02	0.71 ± 0.02	0.73 ± 0.05	0.78 ± 0.03	0.81 ± 0.02	0.99 ± 0.06	1.58 ± 0.06
0.6	0.77 ± 0.03	0.67 ± 0.02	0.61 ± 0.04	0.54 ± 0.02	0.66 ± 0.04	0.72 ± 0.05	0.74 ± 0.07	0.90 ± 0.01	0.75 ± 0.03	1.02 ± 0.07	1.58 ± 0.06
0.7	0.79 ± 0.04	0.71 ± 0.03	0.68 ± 0.05	0.59 ± 0.02	0.67 ± 0.02	0.75 ± 0.04	0.75 ± 0.06	0.96 ± 0.08	0.99 ± 0.02	1.13 ± 0.06	1.58 ± 0.06
0.8	0.79 ± 0.02	0.74 ± 0.03	0.68 ± 0.04	0.63 ± 0.02	0.68 ± 0.03	0.75 ± 0.07	0.86 ± 0.05	1.02 ± 0.01	1.10 ± 0.05	1.25 ± 0.05	1.58 ± 0.06
0.9	0.82 ± 0.04	0.74 ± 0.04	0.70 ± 0.03	0.70 ± 0.03	0.72 ± 0.04	0.80 ± 0.04	0.80 ± 0.04	1.09 ± 0.04	1.16 ± 0.05	1.31 ± 0.03	1.58 ± 0.06
1	0.80 ± 0.03	0.77 ± 0.02	0.72 ± 0.04	0.73 ± 0.04	0.80 ± 0.03	0.92 ± 0.06	0.94 ± 0.02	1.20 ± 0.03	1.25 ± 0.04	1.36 ± 0.05	1.58 ± 0.06

Tabla B.18: Resultados (porcentaje de error $\pm$ desviación típica) de test para los valores de los distintos parámetros α y β del diseño con énfasis Final y guía SDAE3 para MNIST. En cursiva, los valores elegidos por validación, y en negrita el mejor valor en test.

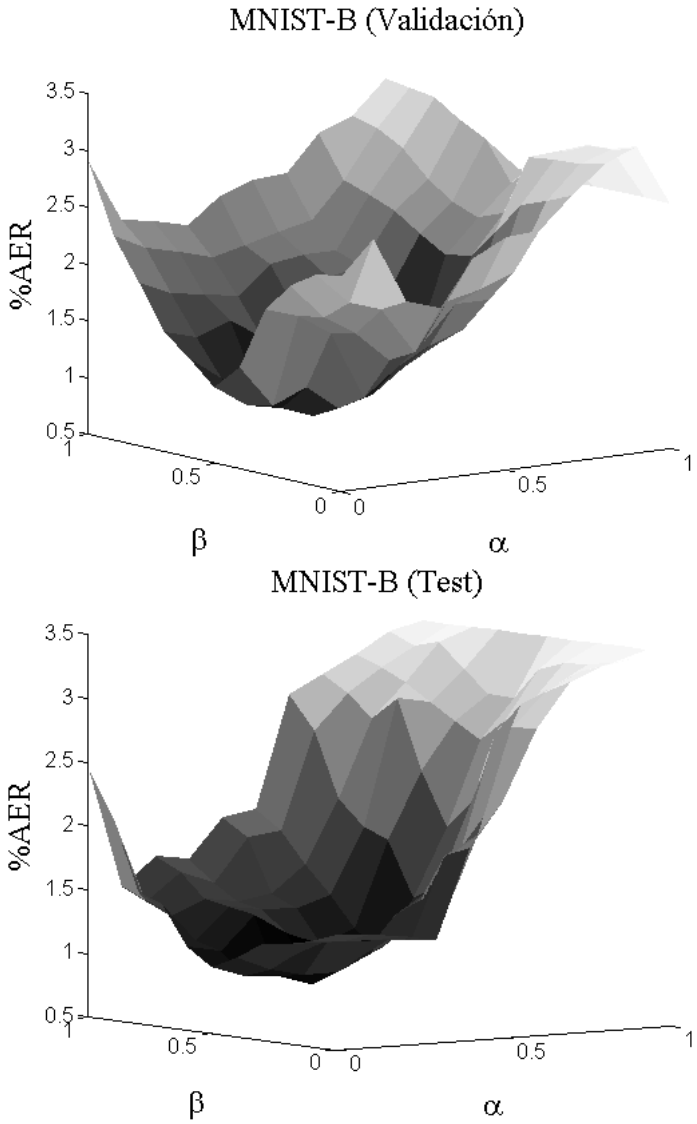

Figura B.10: Porcentaje de tasa de error promedio (Average Error Rate, AER) para los conjuntos de validación y test versus α, β, del problema MNIST-B, con guía SDAE3 y énfasis Final.

β \ α	0	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9	1
0	2.40 ± 0.10	2.69 ± 0.08	2.10 ± 0.10	2.00 ± 0.06	2.40 ± 0.05	2.55 ± 0.10	2.57 ± 0.08	2.84 ± 0.06	3.08 ± 0.12	3.19 ± 0.16	2.65 ± 0.15
0.1	2.36 ± 0.10	2.07 ± 0.07	1.98 ± 0.08	1.56 ± 0.08	1.97 ± 0.09	2.04 ± 0.10	2.15 ± 0.08	2.74 ± 0.12	3.03 ± 0.15	3.12 ± 0.12	2.65 ± 0.15
0.2	2.22 ± 0.08	1.90 ± 0.09	1.72 ± 0.07	1.44 ± 0.10	1.56 ± 0.06	1.86 ± 0.08	1.86 ± 0.06	2.65 ± 0.10	3.01 ± 0.13	3.10 ± 0.12	2.65 ± 0.15
0.3	1.95 ± 0.07	1.86 ± 0.08	1.66 ± 0.06	1.06 ± 0.09	1.29 ± 0.06	1.44 ± 0.06	1.55 ± 0.10	2.37 ± 0.10	2.99 ± 0.08	3.00 ± 0.10	2.65 ± 0.15
0.4	1.40 ± 0.05	1.02 ± 0.06	1.14 ± 0.07	0.92 ± 0.05	1.00 ± 0.07	1.20 ± 0.08	1.36 ± 0.07	2.10 ± 0.10	2.44 ± 0.06	2.86 ± 0.10	2.65 ± 0.15
0.5	1.17 ± 0.06	0.97 ± 0.05	0.91 ± 0.05	$\mathbf{0.80 \pm 0.02}$	0.94 ± 0.05	1.07 ± 0.06	1.20 ± 0.06	2.06 ± 0.10	2.35 ± 0.07	2.62 ± 0.08	2.65 ± 0.15
0.6	1.37 ± 0.08	1.15 ± 0.08	1.06 ± 0.06	1.00 ± 0.10	1.35 ± 0.04	1.66 ± 0.08	1.70 ± 0.07	2.20 ± 0.10	2.44 ± 0.09	2.96 ± 0.12	2.65 ± 0.15
0.7	1.55 ± 0.04	1.47 ± 0.06	1.58 ± 0.04	1.39 ± 0.09	1.60 ± 0.04	1.84 ± 0.09	1.91 ± 0.08	2.35 ± 0.09	2.60 ± 0.10	3.07 ± 0.13	2.65 ± 0.15
0.8	1.94 ± 0.04	1.80 ± 0.06	1.75 ± 0.04	1.70 ± 0.08	1.94 ± 0.05	1.99 ± 0.07	1.99 ± 0.10	2.44 ± 0.08	2.82 ± 0.10	3.22 ± 0.14	2.65 ± 0.15
0.9	2.30 ± 0.06	2.15 ± 0.08	2.08 ± 0.04	1.99 ± 0.09	2.06 ± 0.06	2.14 ± 0.04	2.20 ± 0.08	2.52 ± 0.10	2.94 ± 0.15	3.25 ± 0.14	2.65 ± 0.15
1	2.91 ± 0.07	2.35 ± 0.10	2.30 ± 0.08	2.23 ± 0.06	2.44 ± 0.06	2.55 ± 0.06	2.57 ± 0.09	2.88 ± 0.18	3.00 ± 0.15	3.28 ± 0.10	2.65 ± 0.15

Tabla B.19: Resultados (porcentaje de error $\pm$ desviación típica) de validación para los valores de los distintos parámetros α y β del diseño con énfasis Final y guía SDAE3 para MNIST-B. En negrita el mejor valor.

β \ α	0	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9	1
0	1.40 ± 0.09	1.38 ± 0.10	1.35 ± 0.08	1.30 ± 0.06	2.00 ± 0.08	2.35 ± 0.06	2.90 ± 0.08	3.25 ± 0.09	3.36 ± 0.12	3.40 ± 0.06	3.42 ± 0.10
0.1	1.32 ± 0.08	1.30 ± 0.08	1.29 ± 0.04	1.27 ± 0.04	1.92 ± 0.05	2.06 ± 0.06	2.88 ± 0.04	3.14 ± 0.08	3.28 ± 0.10	3.33 ± 0.04	3.42 ± 0.10
0.2	1.27 ± 0.06	1.26 ± 0.09	1.26 ± 0.04	1.25 ± 0.04	1.54 ± 0.04	1.58 ± 0.08	2.20 ± 0.03	3.02 ± 0.08	3.15 ± 0.10	3.30 ± 0.10	3.42 ± 0.10
0.3	1.25 ± 0.10	1.14 ± 0.06	1.24 ± 0.03	1.10 ± 0.06	1.40 ± 0.02	1.57 ± 0.06	1.90 ± 0.06	2.77 ± 0.06	2.99 ± 0.10	3.28 ± 0.14	3.42 ± 0.10
0.4	1.12 ± 0.12	1.00 ± 0.07	0.94 ± 0.02	0.95 ± 0.08	1.07 ± 0.06	1.23 ± 0.08	1.56 ± 0.02	2.20 ± 0.06	2.82 ± 0.10	3.20 ± 0.08	3.42 ± 0.10
0.5	1.02 ± 0.06	0.93 ± 0.08	0.91 ± 0.02	***0.83 ± 0.02***	0.96 ± 0.06	1.06 ± 0.08	1.45 ± 0.04	1.99 ± 0.04	2.65 ± 0.08	2.96 ± 0.06	3.42 ± 0.10
0.6	1.15 ± 0.07	1.10 ± 0.10	1.36 ± 0.02	1.20 ± 0.04	1.10 ± 0.04	1.20 ± 0.03	1.88 ± 0.03	2.40 ± 0.04	2.77 ± 0.10	2.96 ± 0.10	3.42 ± 0.10
0.7	1.46 ± 0.05	1.16 ± 0.05	1.40 ± 0.02	1.36 ± 0.04	1.35 ± 0.08	1.44 ± 0.05	1.96 ± 0.06	2.95 ± 0.02	2.89 ± 0.08	3.14 ± 0.10	3.42 ± 0.10
0.8	1.54 ± 0.05	1.32 ± 0.06	1.44 ± 0.04	1.40 ± 0.06	1.44 ± 0.06	1.66 ± 0.09	2.15 ± 0.06	2.76 ± 0.04	3.03 ± 0.12	3.32 ± 0.12	3.42 ± 0.10
0.9	2.10 ± 0.02	1.41 ± 0.04	1.62 ± 0.03	1.52 ± 0.05	1.86 ± 0.10	1.89 ± 0.06	2.66 ± 0.08	2.94 ± 0.03	3.12 ± 0.10	3.27 ± 0.12	3.42 ± 0.10
1	2.45 ± 0.06	1.50 ± 0.04	1.73 ± 0.06	1.68 ± 0.06	1.99 ± 0.08	2.04 ± 0.10	2.92 ± 0.12	3.06 ± 0.10	3.20 ± 0.06	3.36 ± 0.14	3.42 ± 0.10

Tabla B.20: Resultados (porcentaje de error ± desviación típica) de test para los valores de los distintos parámetros α y β del diseño con énfasis Final y guía SDAE3 para MNIST-B. En cursiva, los valores elegidos por validación, y en negrita el mejor valor en test.

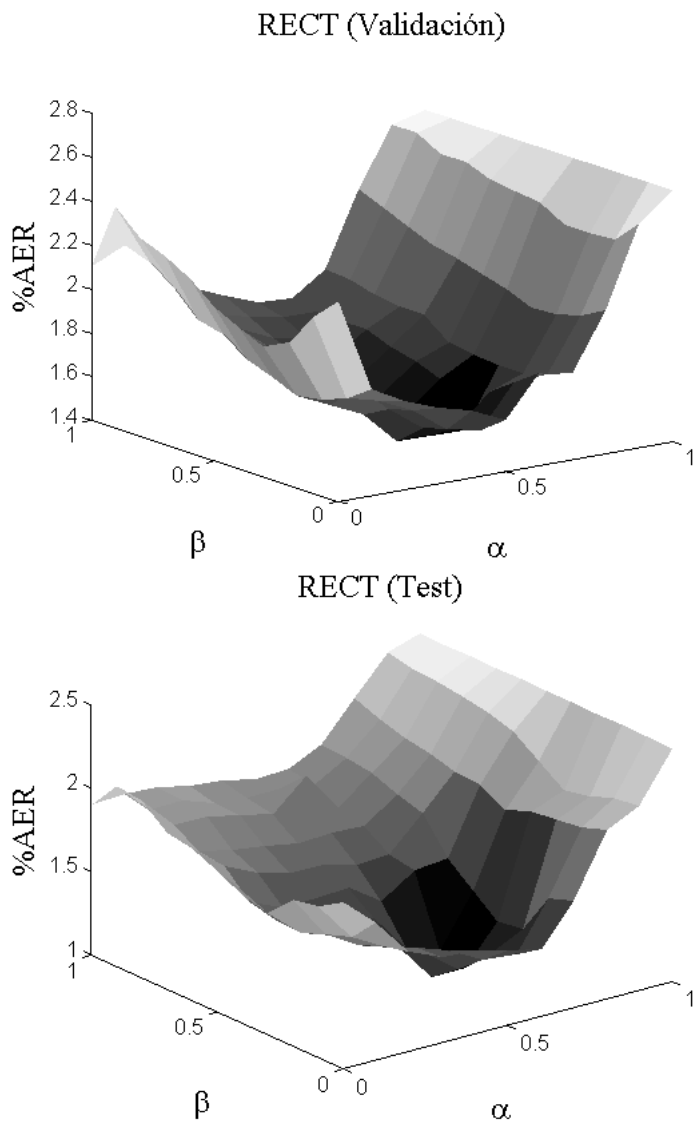

Figura B.11: Porcentaje de tasa de error promedio (Average Error Rate, AER) para los conjuntos de validación y test versus α, β, del problema RECT, con guía SDAE3 y énfasis Final.

β \ α	0	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9	1
0	2.31 ± 0.12	1.88 ± 0.08	1.80 ± 0.08	1.74 ± 0.10	1.70 ± 0.03	1.63 ± 0.09	1.82 ± 0.10	1.79 ± 0.15	2.05 ± 0.16	2.42 ± 0.20	2.54 ± 0.22
0.1	2.20 ± 0.15	1.80 ± 0.05	1.76 ± 0.03	1.69 ± 0.06	1.58 ± 0.03	1.55 ± 0.04	1.74 ± 0.08	1.75 ± 0.13	1.99 ± 0.15	2.31 ± 0.22	2.54 ± 0.22
0.2	2.05 ± 0.10	1.76 ± 0.04	1.80 ± 0.05	1.66 ± 0.06	1.55 ± 0.04	1.52 ± 0.05	1.66 ± 0.06	1.72 ± 0.12	1.96 ± 0.10	2.30 ± 0.20	2.54 ± 0.22
0.3	1.99 ± 0.12	1.74 ± 0.06	1.67 ± 0.08	1.60 ± 0.04	$\mathbf{1.45 \pm 0.02}$	$\mathbf{1.45 \pm 0.03}$	$\mathbf{1.45 \pm 0.05}$	1.72 ± 0.12	1.95 ± 0.11	2.30 ± 0.15	2.54 ± 0.22
0.4	2.04 ± 0.08	1.79 ± 0.04	1.72 ± 0.04	1.65 ± 0.04	1.56 ± 0.04	1.52 ± 0.04	1.65 ± 0.03	1.73 ± 0.10	2.00 ± 0.10	2.37 ± 0.12	2.54 ± 0.22
0.5	2.10 ± 0.09	1.83 ± 0.06	1.77 ± 0.08	1.67 ± 0.06	1.59 ± 0.04	1.57 ± 0.06	1.69 ± 0.08	1.75 ± 0.09	2.03 ± 0.08	2.38 ± 0.09	2.54 ± 0.22
0.6	2.19 ± 0.08	1.91 ± 0.06	1.81 ± 0.06	1.72 ± 0.06	1.63 ± 0.06	1.60 ± 0.08	1.72 ± 0.10	1.84 ± 0.06	2.07 ± 0.06	2.40 ± 0.06	2.54 ± 0.22
0.7	2.26 ± 0.08	1.94 ± 0.08	1.87 ± 0.05	1.77 ± 0.08	1.70 ± 0.06	1.66 ± 0.09	1.76 ± 0.12	1.84 ± 0.08	2.09 ± 0.10	2.44 ± 0.16	2.54 ± 0.22
0.8	2.30 ± 0.07	2.05 ± 0.08	1.94 ± 0.03	1.81 ± 0.08	1.74 ± 0.10	1.71 ± 0.07	1.76 ± 0.10	1.86 ± 0.10	2.14 ± 0.12	2.44 ± 0.08	2.54 ± 0.22
0.9	2.41 ± 0.10	2.13 ± 0.10	1.99 ± 0.06	1.84 ± 0.04	1.80 ± 0.12	1.77 ± 0.08	1.77 ± 0.10	1.89 ± 0.09	2.19 ± 0.06	2.50 ± 0.15	2.54 ± 0.22
1	2.10 ± 0.18	2.17 ± 0.12	2.05 ± 0.09	1.92 ± 0.06	1.86 ± 0.12	1.80 ± 0.10	1.79 ± 0.08	1.90 ± 0.10	2.24 ± 0.16	2.50 ± 0.21	2.54 ± 0.22

Tabla B.21: Resultados (porcentaje de error $\pm$ desviación típica) de validación para los valores de los distintos parámetros α y β del diseño con énfasis Final y guía SDAE3 para RECT. En negrita el mejor valor.

β \ α	0	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9	1
0	2.00 ± 0.12	1.83 ± 0.09	1.60 ± 0.08	1.51 ± 0.05	1.50 ± 0.06	1.45 ± 0.04	1.40 ± 0.04	1.64 ± 0.12	2.01 ± 0.15	2.10 ± 0.08	2.40 ± 0.13
0.1	1.91 ± 0.10	1.67 ± 0.06	1.56 ± 0.08	1.50 ± 0.08	1.43 ± 0.10	1.32 ± 0.08	1.30 ± 0.05	1.59 ± 0.10	1.96 ± 0.10	2.08 ± 0.08	2.40 ± 0.13
0.2	1.88 ± 0.10	1.62 ± 0.08	1.50 ± 0.07	1.44 ± 0.06	1.35 ± 0.08	1.20 ± 0.06	1.24 ± 0.06	1.33 ± 0.08	1.94 ± 0.08	2.02 ± 0.06	2.40 ± 0.13
0.3	1.73 ± 0.08	1.56 ± 0.06	1.49 ± 0.06	1.37 ± 0.04	*1.31 ± 0.02*	***1.08 ± 0.03***	***1.08 ± 0.04***	1.31 ± 0.09	1.92 ± 0.10	1.99 ± 0.06	2.40 ± 0.13
0.4	1.80 ± 0.06	1.57 ± 0.06	1.44 ± 0.07	1.44 ± 0.04	1.42 ± 0.04	1.19 ± 0.04	1.12 ± 0.05	1.43 ± 0.10	1.86 ± 0.10	2.10 ± 0.12	2.40 ± 0.13
0.5	1.85 ± 0.06	1.60 ± 0.08	1.52 ± 0.06	1.52 ± 0.05	1.53 ± 0.05	1.40 ± 0.04	1.53 ± 0.05	1.55 ± 0.02	1.94 ± 0.12	2.21 ± 0.14	2.40 ± 0.13
0.6	1.91 ± 0.06	1.75 ± 0.06	1.66 ± 0.08	1.61 ± 0.06	1.60 ± 0.08	1.55 ± 0.04	1.56 ± 0.04	1.76 ± 0.06	1.95 ± 0.15	2.25 ± 0.15	2.40 ± 0.13
0.7	1.94 ± 0.04	1.77 ± 0.08	1.73 ± 0.06	1.70 ± 0.08	1.66 ± 0.06	1.67 ± 0.06	1.66 ± 0.06	1.80 ± 0.12	1.99 ± 0.12	2.28 ± 0.15	2.40 ± 0.13
0.8	2.06 ± 0.06	1.86 ± 0.04	1.86 ± 0.06	1.81 ± 0.04	1.77 ± 0.10	1.95 ± 0.04	1.70 ± 0.08	1.86 ± 0.10	2.02 ± 0.10	2.30 ± 0.12	2.40 ± 0.13
0.9	2.08 ± 0.08	1.88 ± 0.04	1.89 ± 0.06	1.82 ± 0.04	1.80 ± 0.10	1.80 ± 0.04	1.77 ± 0.04	1.88 ± 0.12	2.06 ± 0.10	2.31 ± 0.10	2.40 ± 0.13
1	1.90 ± 0.10	1.91 ± 0.06	1.90 ± 0.08	1.85 ± 0.06	1.83 ± 0.07	1.79 ± 0.08	1.80 ± 0.06	1.89 ± 0.06	2.12 ± 0.06	2.34 ± 0.06	2.40 ± 0.13

Tabla B.22: Resultados (porcentaje de error $\pm$ desviación típica) de test para los valores de los distintos parámetros α y β del diseño con énfasis Final y guía SDAE3 para RECT. En cursiva, los valores elegidos por validación, y en negrita el mejor valor en test.

Bibliografía

[Abramson, 1963] Abramson, N. (1963). *Information Theory and Coding*. New York, NY: McGraw Hill.

[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.

[Adámek, 1991] Adámek, J. (1991). *Foundations of Coding: Theory and Applications of Error-Correcting Codes with an Introduction to Cryptography and Information Theory*. New York, NY: Wiley.

[Ahachad et al., 2014] Ahachad, A., Omari, A., and Figueiras-Vidal, A. R. (2014). Neighborhood guided smoothed emphasis for real adaboost ensembles. *Neural Proc. Letters*, 42:155–165.

[Aizerman et al., 1965] Aizerman, M. A., Braverman, E. M., and Rozonoer, L. I. (1965). The probablility problem of pattern recognition and the method of potential functions. *Automation and Remote Control*, 25:1175–1190.

[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 and Figueiras-Vidal, 2015] Alvear-Sandoval, R. F. and Figueiras-Vidal, A. R. (2015). Does diversity improve deep learning? In *Proc. 23rd European Signal Processing Conference*, pages 2541–2545, Nice (France). IEEE Press.

[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 *Proc. ESANN2016*, pages 527–532, Bruges (Belgium).

[Alvear-Sandoval et al., 2016] Alvear-Sandoval, R. F., Hayes, M. H., and Figueiras-Vidal, A. R. (2016). Improving denoising stacked auto-encoding classifiers by pre-emphasizing training samples. Remitido a Neurocomputing.

[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.

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

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

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

[Bauer and Kohavi, 1999] Bauer, E. and Kohavi, R. (1999). An empirical comparison of voting classification algorithms: Bagging, boosting, and variants. *Machine Learning*, 35:1–38.

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

[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*, 35:1798–1828.

[Berger, 1999] Berger, A. (1999). Error-correcting output coding for text classification. In *IJCAI'99: Workshop on Machine Learning for Information Filtering*.

[Berlekamp, 1968] Berlekamp, E. R. (1968). *Algebraic Coding Theory*. New York, NY: McGraw Hill.

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

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

[Boser et al., 1992] Boser, B. E., Guyon, I. M., and Vapnik, V. (1992). A training algorithm for optimal margin classifiers. In Haussler, D., editor, *Proc. of the 5th Workshop in Computational Learning Theory*, pages 144–152, San Mateo, CA: ACM Press.

[Bourlard and Kamp, 1988] Bourlard, H. and Kamp, Y. (1988). Auto-association by multilayer perceptrons and singular value decomposition. *Biol. Cybern.*, 59:291–294.

[Bregman, 1967] Bregman, L. M. (1967). The relaxation method of finding the common point of convex sets and its application to the solution of problems in convex programming. *USSR Computational Mathematics and Mathematical Physics*, 7:200–217.

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

[Breiman, 1998] Breiman, L. (1998). Arcing classifiers. *Annals of Statistics*, 26:801–849.

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

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

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

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

[Burges, 1998] Burges, C. J. C. (1998). A tutorial on support vector machines for pattern recognition. *Knowledge Discovery and Data Mining*, 2:121–167.

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

[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 33–40. Barbados: Soc. for Art. Intelligence and Statistics.

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

[Cid-Sueiro et al., 1999] Cid-Sueiro, J., Arribas, J. I., Urbán-Munoz, S., and Figueiras-Vidal, A. R. (1999). Cost functions to estimate a posteriori probabilities in multi-class problems. *IEEE Trans. Neural Networks*, 10:645–656.

[Cireşan et al., 2011] Cireşan, D. C., Meier, U., Gambardella, L. M., and Schmidhuber, J. (2011). Convolutional neural network committees for handwritten character classification. In *Proc. 11th Intl. Conf. on Document Analysis and Recognition*, pages 1135–1139. New York, NY: IEEE Press.

[Cireşan et al., 2012a] Cireşan, D. C., Meier, U., and Schmidhuber, J. (2012a). Multi-column deep neural networks for image classification. In *Proc. Conf. on*

Computer Vision and Pattern Recognition, pages 3642–3649. New York, NY: IEEE Press.

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

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

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

[Delalleau and Bengio, 2011] Delalleau, O. and Bengio, Y. (2011). Shallow vs. deep sum-product networks. In Shawe-Taylor, J., Zemel, R. S., Bartlett, P. L., Pereira, F., and Weinberger, K. Q., editors, *Advances in Neural Information Proc. Sys. 24*, pages 666–674. Cambridge, MA: MIT Press.

[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*, 7:197–387.

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

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

[El Jelali et al., 2008a] El Jelali, S., Lyhyaoui, A., and Figueiras-Vidal, A. R. (2008a). Applying emphasized soft targets for Gaussian mixture model based classification. In *Proc. 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., 2008b] El Jelali, S., Lyhyaoui, A., and Figueiras-Vidal, A. R. (2008b). 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., 2009] El Jelali, S., Lyhyaoui, A., and Figueiras-Vidal, A. R. (2009). Designing Model Based Classifiers by Emphasizing Soft Targets. *Fundamenta Informaticae*, 96:419–433.

[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 Proc. Sys. 2*, pages 524–532. San Mateo, CA: Morgan Kaufmann.

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

[Forney, 1966] Forney, G. D. (1966). *Concatenated Codes.* Cambridge, MA: MIT Press.

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

[Frazão and Alexandre, 2014] Frazão, X. and Alexandre, L. A. (2014). Weighted convolutional neural network ensemble. In *Progress in Pattern Recognition, Image Analysis, Computer Vision, and Applications*, pages 674–681. Zug, Switzerland: Springer.

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

[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. Berlin: Springer.

[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.

[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: ACM Press.

[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. Computer and Sys. Sci.*, 55:119–139.

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

[Girshick et al., 2011] Girshick, R. B., Felzenszwalb, P. F., and McAllester, D. A. (2011). Object detection with grammar models. In Shawe-Taylor, J., Zemel, R. S., Bartlett, P. L., Pereira, F., and Weinberger, K. Q., editors, *Advances in Neural Information Proc. Sys. 24*, pages 442–450. Cambridge, MA: MIT Press.

[Glorot and Bengio, 2010] Glorot, X. and Bengio, Y. (2010). Understanding the difficulty of training deep feedforward neural networks. *AI/Statistics*, 9:249–256.

[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 Tras. Neural Networks*, 19:3–17.

[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*, 69:679–685.

[Gorse et al., 1997] Gorse, D., Shepherd, A. J., and Taylor, J. G. (1997). The new ERA in supervised learning. *Neural Networks*, 10:343–352.

[Grandvalet et al., 1997] Grandvalet, Y., Canu, S., and Boucheron, S. (1997). Noise injection: Theoretical prospects. *Neural Computation*, 9:1093–1108.

[Hamming, 1950] Hamming, R. W. (1950). Error detecting and error correcting codes. *Bell System Technical J.*, 29:147–160.

[Hamming, 1986] Hamming, R. W. (1986). *Coding and Information Theory*. Englewood Cliffs, NJ: Prentice-Hall.

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

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

[Håstad and Goldmann, 1991] Håstad, J. and Goldmann, M. (1991). On the power of small-depth threshold circuits. *Computational Complexity*, 1:113–129.

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

[Hastie et al., 2001] Hastie, T., Tibshirani, R., and Friedman, J. (2001). *The Elements of Statical Learning*. New York, NY: Springer.

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

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

[Hecht-Nielsen, 1990] Hecht-Nielsen, R. (1990). *Neurocomputing*. Reading, MA: Addison-Wesley.

[Heegard and Wicker, 1999] Heegard, C. and Wicker, S. (1999). *Turbo Coding*. Amsterdam: Kluwer.

[Herbrich, 2001] Herbrich, R. (2001). *Learning Kernel Classifiers: Theory and Algorithms*. Cambridge, MA: MIT Press.

[Hertz et al., 1991] Hertz, J., Krogh, A., and Palmer, R. G. (1991). *Introduction to the Theory of Neural Computation*. Redwood City, CA: Addison-Wesley.

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

[Hinton and Salakhutdinov, 2006] Hinton, G. E. and Salakhutdinov, R. R. (2006). Reducing the dimensionality of data with neural networks. *Science*, 313(5786):504–507.

[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 Res. Group, editors, *Parallel Distributed Processing: Explorations in the Microstructure of Cognition, vol. 1: Foundations*, pages 282–317. Cambridge, MA: MIT Press.

[HintonWeb, 2015] HintonWeb (2015). http://www.cs.toronto.edu/~hinton/MatlabForSciencePaper.html

[Ho, 1998] Ho, T. K. (1998). The random subspace method for constructing decision forests. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 20:832–844.

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

[Honary and Markarian, 1997] Honary, B. and Markarian, G. (1997). *Trellis Decoding of Block Codes: A Practical Approach*. New York, NY: Kluwer.

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

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

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

[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*, 3:79–87.

[Jordan and Jacobs, 1992] Jordan, M. I. and Jacobs, R. A. (1992). Hierarchical mixtures of experts and the EM algorithm. Technical Report 9203, MIT Comput. Cognitive Sci. Group, MIT, Cambridge, MA.

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

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

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

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

[Kohonen, 1989] Kohonen, T. (1989). *Self-organization and associative memory (3rd ed.)*. Berlin: Springer.

[Kolmogorov, 1939] Kolmogorov, A. N. (1939). Sur l´interpolation et extrapolation des suites stationnaires. *Comptes Rendues Acad. Sci*, 208:2043–2045.

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

[Lázaro-Teja et al., 2016] Lázaro-Teja, M., Hayes, M. H., and Figueiras-Vidal, A. R. (2016). Bayessian binary classification via Parzen windows: An introduction. To be submitted to *IEEE Trans. Pattern Analysis and Machine Intelligence*.

[LeCun, 1985] LeCun, Y. (1985). Une procedure d´apprentissage pour réseau à seuil assymétrique. In *Proc. of 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*, 1:541–551.

[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 Tourezky, D. S., editor, *Advances in Neural Information Proc. Sys. 2*, pages 396–404. San Mateo, CA: Morgan Kaufmann.

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

[Lee et al., 2015] Lee, S., Purushwalkam, S., Cogswell, M., Crandall, D., and Batra, D. (2015). Why M heads are better than one: Training a diverse ensemble of deep networks. *arXiv:1511.06314v1 [cs.CV]*

[Lin and Costello Jr., 2004] Lin, S. and Costello Jr., D. J. (2004). *Error Control Coding: Fundamentals and Applications* (2nd. ed.). Englewood Cliffs, NJ: Prentice-Hall.

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

[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*, 29:716–725.

[Lyhyaoui et al., 1999] Lyhyaoui, A., Martínez, M., Mora-Jiménez, I., Vázquez-Castro, M., Sancho-Gómez, J.-L., and Figueiras-Vidal, A. R. (1999). Sample selec-

tion via clustering to construct support vector-like classifiers. *IEEE Trans. Neural Networks*, 10:1474–1481.

[MacKay, 2003] MacKay, D. J. C. (2003). *Information Theory, Inference and Learning Algorithms*. Cambridge, UK: Cambridge Univ. Press.

[Maren et al., 1990] Maren, A. J., Harston, C. T., and Pap, R. M., editors (1990). *Handbook of Neural Computing Applications*. San Diego, CA: Academic.

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

[Martens and Sutskever, 2011] Martens, J. and Sutskever, I. (2011). Learning recurrent neural networks with Hessian-free optimization. In *Proc. 28th Intl. Conf. Machine Learning*, pages 1033–1040. Bellevue, WA.

[Martínez-Muñoz et al., 2008] Martínez-Muñoz, G., Sánchez-Martínez, A., Hernández-Lobato, D., and Suárez, A. (2008). Class-switching neural network ensembles. *Neurocomputing*, 71:2521–2528.

[Mayhua-López et al., 2015] Mayhua-López, E., Gómez-Verdejo, V., and Figueiras-Vidal, A. R. (2015). A new boosting design of support vector machine classifiers. *Information Fusion*, 25:63–71.

[McEliece, 1977] McEliece, R. (1977). *The Theory of Information and Coding*. Reading, MA: Addison-Wesley.

[Minsky and Papert, 1969] Minsky, M. L. and Papert, S. E. (1969). *Perceptrons: An Introduction to Computational Geometry*. Cambridge, MA: MIT Press.

[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*, 23:1306–1319.

[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*, 72:3020–3027.

[Müller et al., 2001] Müller, K.-R., Mika, S., Rätsch, G., Tsuda, K., and Schölkopf, B. (2001). An introduction to kernel-based learning algorithms. *IEEE Trans. Neural Networks*, 12:181–201.

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

[Nadaraya, 1964] Nadaraya, E. A. (1964). On estimating regression. *Theory of Probability & Its Applications*, 9:141–142.

[Najafabadi et al., 2015] Najafabadi, M. M., Villanustre, F., Khoshgoftaar, T. M., Seliya, N., Wald, R., and Muharemagic, E. (2015). Deep learning applications and challenges in big data analytics. *J. Big Data*, 2:1–21.

[Nielsen, 2015] Nielsen, M. A. (2015). *Neural Networks and Deep Learning*. Determination Press.

[Nina et al., 2014] Nina, O., Rubiano, C., and Shah, M. (2014). Action recognition using ensemble of deep convolutional neural networks. Tech. Rep., Center R. Computer Vision, Univ. Central Florida, Gainesville, FL.

[O'Neill, 2006] O'Neill. (2006). Standard Reference Data Program NIST. http://www.codeproject.com/kb/library/NeuralNetRecognition.aspx

[Olteanu and Rynkiewicz, 2008] Olteanu, M. and Rynkiewicz, J. (2008). Estimating the number of components in a mixture of multilayer perceptrons. *Neurocomputing*, 71:1321–1329.

[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*, 24:158–163.

[Onwubolu, 2015] Onwubolu, G. C., editor (2015). *GMDH - Methodology and Implementations in C*. World Scientific, Singapore.

[Pao, 1990] Pao, Y. (1990). *Adaptive Pattern Recognition and Neural Networks*. Reading, MA: Addison-Wesley.

[Parker, 1982] Parker, D. B. (1982). Learning logic. Technical Report 581-64, F1, Stanford Univ. Office of Technology Licensing, Stanford, CA.

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

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

[Peterson and Weldon Jr., 1972] Peterson, W. W. and Weldon Jr., E. J. (1972). *Error-Correcting Codes* (2nd. ed.). Cambridge, MA: MIT Press.

[Pimenta and Gama, 2005] Pimenta, E. and Gama, J. (2005). A study on error correcting output codes. In *Proc. 2005 Portuguese Conf. Artificial Intelligence*, pages 218–223. New York, NY: IEEE Comp. Soc. Press.

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

[Principe, 2001] Principe, J. C. (2001). Modeling, segmentation, and classification of nonlinear nonstationary time series. In Sandberg, I. W., Lo, J. T., Francourt, C. L., Principe, J. C., C, K. J., and Haykin, S., editors, *Nonlinear Dynamical Systems: Feedforward Neural Network Perceptrons*, pages 103–209. New York, NY: Wiley.

[Ranzato et al., 2008] Ranzato, M. A., Boureau, Y. L., and LeCun, Y. (2008). Sparse feature learning for deep belief networks. In Platt, J., Koller, D., Singer, Y., and S, R., editors, *Advances in Neural Information Proc. Sys. 20*, pages 1185–1192. Cambridge, MA: MIT Press.

[Ranzato et al., 2007] Ranzato, M. A., Huang, F. J., Boureau, Y. L., and LeCun, Y. (2007). Unsupervised learning of invariant feature hierarchies with applications to object recognition. In *Proc. 2007 Computer Vision and Pattern Recognition Conf.*, pages 1–8. New York, NY: IEEE Press.

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

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

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

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

[Reed, 1993] Reed, R. (1993). Pruning algorithms–A survey. *IEEE Trans. Neural Networks*, 4:740–747.

[Reed et al., 1995] Reed, R., Oh, S., and Marks, R. J. I. (1995). Similarities of error regularization, sigmoid gain scaling, target smoothing, and training with jitter. *IEEE Trans. Neural Networks*, 6:529–538.

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

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

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

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

[Rosenblatt, 1962] Rosenblatt, F. (1962). *Principles of neurodynamics*. Washington D. C.: Spartan.

[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 Research Group, editors, *Parallel Distributed Processing, Explorations in the Microstructure of Cognition. Vol. I. Foundations*, pages 318–362. Cambridge, MA: MIT Press.

[Rumelhart et al., 1986b] Rumelhart, D. E., Hinton, G. E., and Williams, R. J. (1986b). Learning representations by back-propagating errors. *Nature*, 323(99):533–538.

[Salakhutdinov et al., 2011] Salakhutdinov, R. R., Tenenbaum, J. B., and Torralba, A. (2011). Learning to learn with compound hd models. In Shawe-Taylor, J., Zemel, R. S., Bartlett, P. L., Pereira, F., and Weinberger, K. Q., editors, *Advances in Neural Information Proc. Sys. 24*, pages 2061–2069. Cambridge, MA: MIT Press.

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

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

[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. *Machine Learning*, 37:297–336.

[Scherer et al., 2010] Scherer, D., Müller, A., and Behnke, S. (2010). Evaluation of pooling operations in convolutional architectures for object recognition. In *Proc. 2010 Intl. Conf. Artificial Neural Networks*, pages 92–101. Thessaloniki (Greece): Springer.

[Schmidhuber, 2014] Schmidhuber, J. (2014). Deep learning in neural networks: An overview. Technical Report IDSIA-03-14, Univ. Lugano, 2014
arXiv:1404.7828v4[cs.NE]

[Schölkopf et al., 1999] Schölkopf, B., Burges, C. J. C., and Smola, A. J., editors (1999). *Advances in Kernel Methods: Support Vector Learning.* Cambridge, MA: MIT Press.

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

[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.

[Shannon, 1948a] Shannon, C. E. (1948a). A mathematical theory of communication (I). *Bell Systems Technical J.*, 27:379–423.

[Shannon, 1948b] Shannon, C. E. (1948b). A mathematical theory of communication (II). *Bell Systems Technical J.*, 27:623–655.

[Shao et al., 2014] Shao, L., Wu, D., and Li, X. (2014). Learning deep and wide: A spectral method for learning deep networks. *IEEE Trans. Neural Networks and Learning Sys.*, 25:2303–2308.

[Sharkey, 1999] Sharkey, A. J. C., editor (1999). *Combining Artificial Neural Nets: Ensemble and Modular Multi-Net Systems*. London, UK: Springer.

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

[Shawe-Taylor and Sun, 2011] Shawe-Taylor, J. and Sun, S. (2011). A review of optimization methodologies in support vector machines. *Neurocomputing*, 74:3609–3618.

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

[Simpson, 2015] Simpson, A. J. (2015). Instant learning: Parallel deep neural networks and convolutional bootstrapping. *arXiv preprint 1505.05972*

[Sklansky and Michelotti, 1980] Sklansky, J. and Michelotti, L. (1980). Locally trained piecewise linear classifiers. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 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 Res. Group, editors, *Parallel Distributed Processing: Explorations in the Microstructure of Cognition, vol. 1: Foundations*, pages 194–281. Cambridge, MA: MIT Press.

[Srivastava et al., 2014] Srivastava, N., Hinton, G. E., Krizhevsky, A., Sutskever, I., and Salakhutdinov, R. (2014). Dropout: A simple way to prevent neural networks from overfitting. *J. Machine Learning Res.*, 15:1929–1958.

[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*, 20:1093–1116.

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

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

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

[Tabik et al., 2017] Tabik, S., Peralta, D., and Herrera-Poyatos, F. (2017). A snapshot of image pre-processing for convolutional neural networks: case study of MNIST. *Intl. J. Comput. Intell. Sys.*, 10:555–568.

[Valiant, 1982] Valiant, L. (1982). A theory of the learnable. *Communications of the ACM*, 27:1134–1142.

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

[Vapnik, 1995] Vapnik, V. (1995). *The Nature of Statical Learning Theory*. New York, NY: Springer.

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

[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.

[Vincent et al., 2010] Vincent, P., Larochelle, H., Lajoie, I., Bengio, Y., and Manzagol, P.-A. (2010). Stacked denoising autoencoders: Learning useful representations in a deep network with a local denoising criterion. *J. Machine Learning Res.*, 11:3371–3408.

[Wan et al., 2013] Wan, L., Zeiler, M., Zhang, S., LeCun, Y., and Fergus, R. (2013). Regularization of neural networks using dropconnect. In *Proc. 30th Intl. Conf. Machine Learning*, pages 1058–1066. Atlanta, GA: JMLR W&CP28.

[Wang, 2005] Wang, L. (2005). *Support Vector Machines: Theory and Applications*. New York, NY: Springer.

[Watson, 1964] Watson, G. S. (1964). Smooth regression analysis. *Sankhyā: The Indian Journal of Statistics, Series A*, 26:359–372.

[Weigend et al., 1995] Weigend, A. S., Mangeas, M., and Srivastava, A. N. (1995). Nonlinear gated experts for time series: Discovering regimes and avoiding overfitting. *Intl. J. Neural Sys.*, 6:373–399.

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

[Werbos, 1994] Werbos, P. J. (1994). *The Roots of Backpropagation: From Ordered Derivatives to Neural Networks and Political Forecasting*. New York, NY: Wiley.

[Widrow and Hoff, 1960] Widrow, B. and Hoff, M. E. (1960). Adaptive switching circuits. In *1960 IRE WESCON Conv. Record*, pages 96–104. Los Angeles, CA.

[Widrow and Lehr, 1990] Widrow, B. and Lehr, M. A. (1990). 30 years of adaptive neural networks: Perceptron, Madaline, and Backpropagation. *Proc. IEEE*, 78:1415–1442.

[Wiener, 1949] Wiener, N. (1949). *Extrapolation, Interpolation, and Smoothing of Stationary Time Series with Engineering Applications*. Cambridge, MA: MIT Press.

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

[Xia et al., 2010] Xia, T., Tao, D., Mei, T., and Zhang, Y. (2010). Multiview spectral embedding. *IEEE Trans. Systems, Man, and Cybernetics, Part B: Cybernetics*, 40:1438–1446.

[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 Sys.*, 23:1177–1193.

[Zeevi et al., 1997] Zeevi, A. J., Meir, R., and Adler, R. J. (1997). Time series prediction using mixtures of experts. In Mozer, M., Jordan, M. I., and Petsche, T., editors, *Advances in Neural Information Proc. Sys. 9*, pages 309–318. Cambridge, MA: MIT Press.

[Zhang et al., 2003] Zhang, A., Wu, Z.-L., Li, C.-H., and Fang, K.-T. (2003). On Hadamard-type output coding in multiclass learning. In *Proc. IDEAL 2003*, pages 397–404, LNCS 2690. Berlin, Heidelberg: Springer.

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

[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*, 214:381–392.

[Zhao et al., 2014] Zhao, T., Zhao, Y., and Chen, X. (2014). Building an ensemble of CD-DNN-HMM acoustic model using random forests of phonetic decision trees. In *Proc. IEEE 9th Int. Symp. on Chinese Spoken Language Processing*, pages 98–102. Singapore.

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

[Zurada, 1994] Zurada, J. M. (1994). *Introduction to Artificial Neural Systems*. St. Paul, MN: West.