UNIVERSIDAD NACIONAL DE INGENIERÍA

FACULTAD DE INGENIERÍA ECONÓMICA Y CIENCIAS SOCIALES

ESCUELA PROFESIONAL DE INGENIERÍA ESTADÍSTICA

IDENTIFICACIÓN DE PUNTOS DE PESCA EN LOS VIAJES PESQUEROS DE ANCHOVETA MONITOREADOS POR SATÉLITE MEDIANTE REDES NEURONALES ARTIFICIALES

Tesis para optar por el Título Profesional de Ingeniera Estadística

Autora: **Rocío Joo**

Asesor: **Luis Navarro Huamaní**

Asesora externa: **Sophie Bertrand**

Lima - Perú

2008

A los unecos y unecas, compañeros de universidad y todos quienes siguen "optando" por hacer tesis.

A todos aquellos (y aquellas) que de una u otra forma también influyeron en que la comience… y la termine.

Caminante son tus huellas
El camino nada más;
caminante no hay camino
se hace camino al andar.
Al andar se hace camino
y al volver la vista atrás
se ve la senda que nunca
se ha de volver a pisar.

Antonio Machado

Agradecimientos

No me interesa dónde has estudiado, ni qué has estudiado, ni con quién lo has hecho.
Quiero saber qué es lo que te sostiene desde adentro cuando todo lo demás falla.

Oriah Soñador de la Montaña
Anciano Indio

Gracias al IMARPE y a las personas que trabajan en él porque sin ellos no hubiese tenido la posibilidad de desarrollar este trabajo. Gracias a Miguel Ñiquen, Director de Investigaciones en Recursos Pelágicos Neríticos y Oceánicos; a toda la Unidad de Investigaciones en Dinámica de Poblaciones, especialmente a Erich Díaz, Marilú Bouchon y Cecilia Peña; por su interés en el proyecto y por su completa disponibilidad para acogerme en su oficina y acudir en mi auxilio cuando –por mi casi nulidad de conocimiento en pesquería –les solicitaba datos, literatura y respuestas. Gracias también a Sara Purca y a David Correa, del CIMBOP, por permitirme robarles algo de su tiempo y estar siempre prestos a ayudarme con las corridas del Matlab y mis dudas.

Deseo hacer una mención aparte para Erich, mi primer contacto en IMARPE: gracias por ayudar a orientarme en la aplicación de las redes neuronales, ofreciéndome tus conocimientos por la experiencia de trabajo previo; pero sobre todo, quiero agradecerte por tu entusiasmo por la investigación a contrapelo de los obstáculos del día a día.

Gracias al IRD, por financiar mi estancia en IMARPE. Gracias Arnaud y Sophie por brindarme la oportunidad de realizar mis prácticas y mi tesis aquí, por enseñarme mucho en este poco tiempo, sobre la investigación, ecología pesquera, la vida… gracias por apostar por mi futuro y ofrecerme más oportunidades. Gracias Sophie, por soportar mis espaciados silencios seguidos de una excesiva carga de páginas esperando tu lectura y corrección. Gracias por las críticas, los consejos, las opiniones, por introducirme a este nuevo mundo de la investigación, que anhelaba pero del que poco sabía.

Merci aux deux, Sophie et Arnaud, pour de bonnes et très intéressantes conversations, pour vos luttes quotidiennes pour changer ce qui va mal… pour des « aventures » de la cuisine… enfin, merci d'être les athées les plus sympas que je connaisse jusqu'à présent.

Gracias Alexis y Jérémie, por poder hacerse un tiempo para echarnos dos manos con el Matlab, sin su ayuda hubiese sido mucho más difícil (quizá imposible) hacer aplicable y accesible lo que aquí presentamos. Gracias Alexis también por las explicaciones sobre las corrientes oceánicas. Je sais que ce jour-là tu as beaucoup souffert pour m'expliquer ceci. Désolée !

Merci beaucoup Matthieu Lengaigne : le travail que je présente ici est seulement une continuation du tien ; sans lequel le « mien » n'aurait pas été posible (alors, il n'est pas complètement mien). Merci aussi pour les orientations et le temps pour répondre mes doutes. J'espère de te connaître tôt !

Gracias también a Edith Seier por sus aportes para las discusiones sobre los indicadores de la red, gracias por tu tiempo para las placenteras y fructuosas reflexiones en estadística. Y gracias a Fico, por la revisión del abstract.

Creo que con esta tesis concluyo una etapa, lo que es muy significativo. Pero comienzo otra, por lo que es sólo el inicio (a pesar de que la vida es corta y el mañana, incierto). Haré entonces una mención breve a todas las personas que han sido parte de este proceso largo de ser universitaria no porque no sean importantes, sino porque espero que gratuitamente sigamos acompañándonos en el caminar de la vida: gracias a mi familia por su apoyo incondicional, a mi familia de UNEC en especial a mi comunidad por motivarme a ser mejor universitaria, ciudadana, hija, hermana, cristiana y profesional con sus vidas, a Daniel por ser el mejor compañero que he podido tener en este tiempo –y darte el tiempo de leer varios de mis borradores de tesis también –a mis amigos y amigas de la UNI –porque, ¿qué sería la vida universitaria sin ellos y ellas? –y a todas las otras personas que me han acompañado en mi hacer camino y me han permitido aprender del suyo.

Resumen

La sostenibilidad de los recursos marinos y de su ecosistema hace necesario un manejo responsable de las pesquerías. Conocer la distribución espacial del esfuerzo pesquero y en particular de las operaciones de pesca es indispensable para mejorar el monitoreo pesquero y el análisis de la vulnerabilidad de las especies frente a la pesca. Actualmente en la pesquería de anchoveta peruana, se recoge información del esfuerzo y capturas mediante un programa de observadores a bordo, pero esta solo representa una muestra de 2% del total de viajes pesqueros. Por otro lado, se dispone de información por cada hora (en promedio) de la posición de cada barco de la flota gracias al sistema de seguimiento satelital de las embarcaciones (VMS), aunque en estos no se señala cuándo ni dónde ocurrieron las calas. Las redes neuronales artificiales (ANN) podrían ser un método estadístico capaz de inferir esa información, entrenándose en una muestra para la cual sí conocemos las posiciones de calas (el 2% anteriormente referido), estableciendo relaciones analíticas entre las calas y ciertas características geométricas de las trayectorias observadas por el VMS y así, a partir de las últimas, identificar la posición de las operaciones de pesca. La aplicación de la red neuronal requiere un análisis previo que examine la sensibilidad de la red a variaciones en sus parámetros y bases de datos de entrenamiento, y que nos permita desarrollar criterios para definir la estructura de la red e interpretar sus resultados de manera adecuada.

La problemática descrita en el párrafo anterior, aplicada específicamente a la anchoveta (*Engraulis ringens*) es detalllada en el primer capítulo, mientras que en el segundo se hace una revisión teórica de las redes neuronales. Luego se describe el proceso de construcción y pre-tratamiento de la base de datos, y definición de la estructura de la red previa al análisis de sensibilidad. A continuación se presentan los resultados para el análisis en los que obtenemos una estimación del 100% de calas, de las cuales aproximadamente 80% están correctamente ubicadas y 20% poseen un error de ubicación. Finalmente se discuten las fortalezas y debilidades de la técnica empleada, de métodos alternativos potenciales y de las perspectivas abiertas por este trabajo.

Palabras clave:

Redes neuronales; análisis de sensibilidad; anchoveta; *Engraulis ringens*, puntos de pesca; identificación de calas; sistema de seguimiento satelital (SISESAT); parada temprana, VMS, observadores a bordo.

Abstract

Responsible management of fisheries is necessary in order to guarantee the sustainability of marine resources and their ecosystem. Knowing the spatial distribution of fishing effort and particularly the fishing set locations is key for improving fisheries monitoring and the analysis of vulnerability of the species to fishing. Today in Peruvian anchovy fisheries information on effort and catches is collected by an at-sea observer program. However, this represents only a 2% sample of the fishing trips performed by the fleet. On the other hand, hourly information on position is available for each vessel thanks to the satellite vessel monitoring system (VMS). Nonetheless, VMS does not indicate when and where fishing sets occurred. Artificial neural networks (ANN) are a statistical method that could allow inferring such information, by training the net on a sample with known fishing set locations (the 2% discussed earlier), then establishing analytical relationships between fishing sets and some geometric characteristics of the trajectories observed in VMS data. This allows us to identify a priori unknown fishing set locations. The neural network application requires a preliminary analysis that examines the sensitivity of the net to variations in its parameters and training databases, which allows us to develop criteria for defining the net structure and interpreting the results in an adequate manner.

The problematic described above is applied to Peruvian anchovy (*Engraulis ringens*) and detailed in the first chapter, while the second one undertakes a theoretical review of ANN. In the following chapter, we describe the process of obtaining and processing the database and we define the network structure on which the sensitivity analysis is performed. Later, the results of the analysis are presented: we obtain a 100% estimation of the fishing set with 80% correctly located and 20% with location uncertainty. Finally, we discuss the strengths and weaknesses of the technique, potential alternative methods and the perspectives opened by this work.

Key words:

Artificial neural networks (ANN); sensitivity analysis; Peruvian anchovy; *Engraulis ringens*; Vessel Monitoring System (VMS); fishing set locations; early-stopping.

ÍNDICE

CAPITULO I
INTRODUCCIÓN

1. Tematización o marco situacional

La pesca es actualmente la segunda fuente generadora de divisas para el Perú después de la minería, generando entre 1 000 y 1 650 millones de dólares entre el 2000 y el 2005[1] y llegando a un valor de 1 749.5 millones de dólares en el 2006[2]. La pesquería de anchoveta peruana (*Engraulis ringens*), con una extracción de casi 7 millones de toneladas de pescado, representa en promedio (entre el 2000 y el 2006) el 79% de la producción pesquera peruana total y el 13.3% de la producción pesquera mundial y se constituye como la primera pesquería monoespecífica[3] del mundo[4] .

Para quienes no conocemos mucho de la actividad, sería importante describir brevemente aspectos principales de su historia:

La pesquería de anchoveta en el Perú se inició a principios de los años 50, con unos 50 barcos bolicheros[5] de 50-60 toneladas métricas de capacidad de bodega[6], extrayendo alrededor de 32 mil toneladas en promedio en sus 5 primeros años (FAO, 2008). Rápidamente la flota comenzó a crecer en número y en tecnología (ver más detalles en Tilic, 1963; Smetherman y Smetherman, 1973). Así, en 1970, con 1 499 embarcaciones y 265 735 TM de capacidad de bodega acumulada en (IMARPE, Unidad de Dinámica de Poblaciones), la historia de esta pesca llega a su pico más alto al obtener 12 277 000 toneladas en desembarques oficiales (FAO, 2008) –aunque se estima que hubo bastante sub-declaración y que la verdadera cifra de captura fue mucho mayor. Súbitamente después de este "*auge*" sobrevino el *colapso* del stock (figura 1), manifestado en producciones pesqueras de 4 447 400 (decrecimiento del 63.4% con respecto a 1970) en 1972 y 1 513 000 toneladas (decrecimiento del 86.7% con respecto a 1970) en 1973 (FAO, 2008); este colapso coincidió con un Niño muy fuerte y largo (Zuta *et al*., 1976), y un período de cambio climático a escala decadal entre períodos fríos ('La Vieja') y cálidos ('El Viejo') (Alheit y Ñiquen, 2004; Bertrand *et al.*, 2004a), que provocaron condiciones ambientales desfavorables, las que aunadas a la potencia pesquera y la poca regulación existente, ocasionaron una gran reducción de la población de anchoveta.

La actividad se convirtió en insostenible tanto en el campo de la extracción de la materia prima, como de la producción de harina, para empresarios y empleados (Caviedes, 1975). Simultáneamente a la crisis económica,[7] sobrevino la ecológica. Al desaparecer la anchoveta en magnitud significativa, afectó a diferentes especies en cuya cadena alimenticia participa directa o indirectamente (como sus principales predadores además del hombre, las aves y lobos de mar), perturbando la sostenibilidad del ecosistema marino[8].

---

[1] http://www.produce.gob.pe/descarga/estadisticas/pesca/come/2005/valorfob.pdf

[2] http://www.apec2008.org.pe/contenidospanish/perupesca.html

[3] Pesca monoespecífica: basada en una sola especie.

[4] Datos obtenidos a partir de: FAO Servicio de Información y Estadísticas de Pesca y Acuicultura. 2008. Producción pesquera total 1950-2006. FISHSTAT Plus: Programa informático universal para series cronológicas de estadísticas pesqueras [en línea o CD-ROM]. Organización de las Naciones Unidas para la Agricultura y la Alimentación. Disponible en: http://www.fao.org/fi/statist/FISOFT/FISHPLUS.asp. De aquí en adelante le haremos referencia simplemente como FAO, 2008.

[5] Son unidades de pesca que usan redes de cerco para la extracción de la anchoveta.

[6] Capacidad de almacenamiento de lo pescado de un barco (volumen en TM – toneladas métricas).

[7] Al respecto de las repercusiones económicas de explotación y ordenación (o no ordenación) de los recursos pesqueros, la FAO señala: "Existe una interacción estrecha entre las variables social y económica y, con toda probabilidad, cualquier medida de ordenación repercutirá, por ejemplo, en la distribución de los ingresos y la riqueza, el volumen y el tipo de empleo, la asignación de los derechos de uso, y la composición y cohesión de los grupos y subgrupos de interés. Con carácter más general, las decisiones y medidas adoptadas en materia de ordenación influirán en las actitudes, positivas y negativas, de los grupos interesados. Las medidas de ordenación pesquera pueden condicionar también la contribución de la pesca en aspectos de política primordiales como la seguridad alimentaria, los ingresos netos en divisas, las subvenciones y otros costos y beneficios." (FAO, 1999)

[8] En el mismo documento (FAO, 1999), la FAO se refiere a las consideraciones ecológicas que debe tener la actividad pesquera: "Por norma general, las actividades pesqueras se centran en una o más especies de un ecosistema, pero con frecuencia también afectan a otros componentes de dicho ecosistema, por ejemplo, debido a la pesca accidental de otras especies, a los daños causados al ecosistema o a los efectos producidos en la cadena alimentaria. La ordenación de la pesca responsable debe tener en cuenta el impacto de la actividad en el ecosistema en su conjunto, incluso la biodiversidad, y el objetivo debe ser la utilización sostenible de ecosistemas y comunidades biológicas completos."

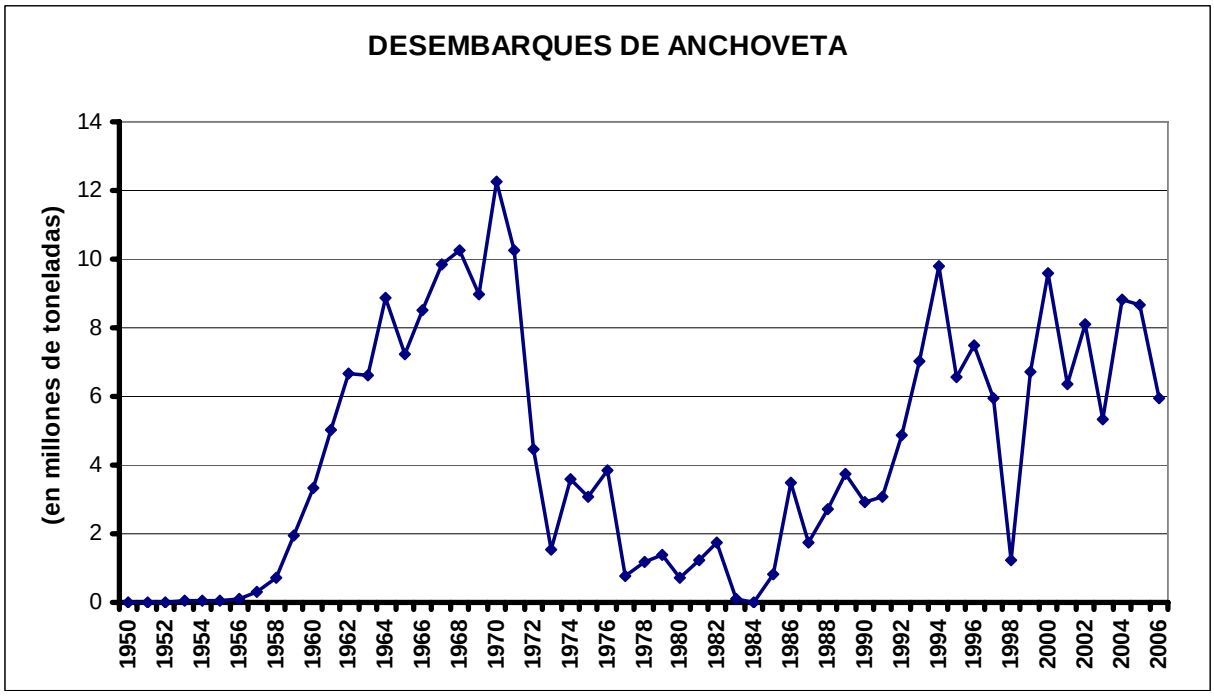


Figura 1: Desembarques de anchoveta en toneladas métricas desde el comienzo de la actividad pesquera y hasta el 2006. El colapso del stock de principios de los '70 se observa en la caída más grande en la gráfica (1972-1973). Datos: FAO, 2008.

En sus inicios, las investigaciones en ciencia pesquera eran necesarias para optimizar la eficiencia en pesca (Holden y Raitt, 1975); hoy en día esto ya no es necesario. Como claramente lo demuestra el episodio anteriormente descrito (el colapso de la anchoveta), ya no apremia la ayuda a los pesqueros para extraer más; urge por el contrario, que los científicos pesqueros realicen investigaciones que apunten a dar recomendaciones sobre un manejo pesquero adecuado que permita la sostenibilidad del recurso y del ecosistema.

En el país, el Instituto del Mar del Perú (IMARPE) es el ente encargado del monitoreo e investigación científica de la actividad pesquera y de la biología y dinámica de las poblaciones de seres vivos que habitan en nuestro mar, generando un conocimiento de este último que le permita asesorar al Estado en la toma de decisiones con respecto al uso racional de los recursos pesqueros y la conservación del ambiente marino, evitando así la sobre-explotación de los recursos hidrobiológicos (www.imarpe.gob.pe).

Para que un recurso pueda ser (o seguir siendo) explotado, debe contar con una abundancia y una población de adultos en capacidad de reproducción tales que las posibilidades de crecimiento neto de la población sean mayores que el nivel de aprovechamiento del recurso (la mortalidad causada por la pesca), salvaguardándolo de una potencial extinción. Por ello, conocer la abundancia existente, su sensibilidad a las condiciones ambientales variables, la estructura de edad y de distribución de la población, y el efecto de la pesca en la mortalidad de esta, son requerimientos de importancia extrema para poder regular el esfuerzo pesquero (controlando y regulando los métodos de pesca, el número de capturas, fechas y lugares en los que se permite la pesca, etc.). (FAO, 1999)

Para la estimación de biomasa y el manejo del stock pesquero, el IMARPE utiliza diferentes métodos como la hidroacústica, el análisis de población virtual (APV) y la captura por unidad de esfuerzo (CPUE). Los 2 primeros son explicados brevemente en apartados complementarios, y el último, por ser de mayor interés para la problemática que presentamos, será abordado a continuación:

Los modelos de captura-esfuerzo (o de CPUE) son aquellos que se basan en información de las capturas (nivel de extracción obtenido por las embarcaciones en zonas de pesca determinadas) y el esfuerzo pesquero (cantidad de trabajo realizado por una unidad pesquera en cierta unidad de tiempo) (Laurec y Le Guen, 1981).

Una primera aproximación a estos modelos plantea que:

$$B_t = CPUE_t / q \quad \ldots(1),$$

donde B_t: biomasa (abundancia expresada en peso total del recurso) en el tiempo *t*,
q: coeficiente de capturabilidad, definido como la fracción del stock capturada, en promedio, por unidad de esfuerzo.
y $CPUE_t$ es el índice de captura por unidad de esfuerzo en el tiempo *t*,

siendo $CPUE_t = C_t / E_t \quad \ldots(2),$

donde C_t: captura total en el tiempo *t*
y E_t: esfuerzo pesquero en el tiempo *t* (Fréon y Misund, 1999)

Existen diversas maneras de calcular el esfuerzo, y por lo tanto, diversas maneras de calcular la CPUE. Bouchon *et al.* (2000) describen 3 índices de captura por unidad de esfuerzo para la pesquería de anchoveta que se estiman a escalas mensual, estacional y anual: CPUE en base a captura por viaje estandarizado, CPUE en base a captura por TRB – viaje (Tonelaje de registro bruto por embarcación por viaje) y CPUE en base a captura por TRB-HVJE (Tonelaje de registro bruto por embarcación por hora de viaje).

Con un coeficiente de capturabilidad constante, este sencillo modelo (1) asume una relación directamente proporcional entre la CPUE y la biomasa; y asume, entre otras cosas, que los grupos de edad se mantienen constantes a través del tiempo; lo que haría de la CPUE, un buen índice de abundancia.

Sin embargo, Fréon y Misund (1999) exponen algunos factores importantes relacionados a variaciones (en tiempo y espacio) en el medio ambiente, la densidad de los peces, sus niveles de agregación (cardúmenes, conglomerados de cardúmenes, conglomerados de conglomerados), así como las estrategias de pesca, el esfuerzo y la capturabilidad, por los que la relación de proporcionalidad entre la CPUE y la abundancia no se cumple (ver también Arreguin-Sanchez, 1996; Harley *et al.*, 2001; Csirke, 1989; por citar algunos ejemplos en los que CPUE y biomasa pueden guardar hasta una relación inversa). Bertrand (2005) explica que para especies que forman agregaciones como la anchoveta, el tamaño de éstas últimas disminuye cuando la explotación pesquera aumenta, y los peces que quedan forman nuevas agregaciones. Al reducirse el tamaño de la población y del hábitat, el esfuerzo de búsqueda también se atenúa (por lo que la CPUE aumenta). Cuando esto sucedía, el aumento de captura era interpretado por los pescadores como un indicador de incremento en la abundancia, por lo que pescaban aun más; cuando en realidad la capturabilidad aumentaba y la abundancia disminuía. Una lección histórica es el colapso del bacalao cerca al este de Canadá a principios de los años 90, en el que la CPUE crecía mientras la abundancia real decaía. Rose y Kulka (1999) realizaron un análisis espacial de la CPUE y de las densidades y abundancia del bacalao, obteniendo diferentes relaciones entre la CPUE y la biomasa en las distintas zonas observadas, indicando probablemente diferencias espaciales en la agregación de los peces; y concluyeron que la incapacidad de reconocer los cambios en la abundancia real debido a la insensibilidad relativa de la CPUE local llevó al colapso del stock.

Gaertner y Dreyfus-Leon (2004) también apuntan a un análisis espacial del esfuerzo pesquero al señalar su importancia para entender las causas de la relación no proporcional entre el CPUE y la abundancia. Bertrand (2005) propone conocer las formas de interacción en el espacio entre peces y pescadores (Bertrand *et al.*, 2004b y 2005, realizan estudios sobre la interacción entre la distribución espacial de la anchoveta peruana y el comportamiento espacial de los pescadores) para además obtener una información espacial de la presa y el predador como partes de un mismo ecosistema. Así, una CPUE espacializada, interpretada no como índice de biomasa, sino como indicador de la vulnerabilidad del stock a la pesca, se constituiría en una herramienta muy útil para un apropiado ordenamiento pesquero[9].

[9] El ordenamiento pesquero o la ordenación pesquera comporta un conjunto amplio y complejo de tareas encaminadas a conseguir los máximos beneficios para los usuarios locales, el Estado o la región mediante la utilización sostenible de los recursos acuáticos vivos a los que tienen acceso (FAO, 1999).

ESTIMACIÓN DE LA BIOMASA MEDIANTE MÉTODOS HIDROACÚSTICOS

Este método consiste en la emisión de pulsos (ondas de sonido) desde transductores ubicados debajo del agua (debajo del casco del barco de investigación) y orientados hacia el fondo; y la recepción de los ecos devueltos de los obstáculos que las ondas encontraron en el camino (como por ejemplo, peces); a partir de estos ecos de diferentes intensidades de sonido (pues los "obstáculos" encontrados son distintos) se realizarán estimaciones por unidad de muestreo de 1 milla náutica de la densidad y número de los peces, lo cual servirá para inferir la biomasa de las diferentes especies, a través de técnicas de ecointegración y de identificación de ecoregistros (Simmonds y MacLennan, 2005).

Simmonds y MacLennan (2005) detallan algunos sesgos de la hidroacústica en la evaluación de recursos pesqueros (relacionados a los límites de cobertura/detección, el evitamiento de las embarcaciones acústicas por parte de los peces, el reconocimiento de especies y otros) que hacen que el método tienda, por lo general, a subestimar.

Por otro lado, continuamente se realizan nuevos avances tecnológicos y científicos –estos últimos apoyados fuertemente en la estadística –que van aminorando los problemas referidos (Gerlotto, 2001a y 2001b); desarrollando mayores conocimientos y herramientas que permiten identificar con mayor precisión a las especies a partir de los registros acústicos (Ballón et al., 2008), lo cual otorga un mayor grado de confiabilidad a los resultados obtenidos por hidroacústica (MacLennan y Holliday, 1998).

En el Perú y desde 1975 –aunque de manera sistemática desde 1983 -, IMARPE hace uso de las técnicas hidroacústicas para la determinación de la distribución y abundancia de recursos pelágicos (Gutiérrez, 2000a y 2000b). Actualmente, se realizan en promedio, 2 cruceros al año dirigidos específicamente a la evaluación acústica en especies pelágicas y 1 o 2 más (ejm. huevos y larvas) de los cuales se infiere también biomasa de peces..

Hoy en día, los estimados de biomasa hidroacústica constituyen la información más relevante al momento de establecer cuotas de captura. No obstante, datos obtenidos por otros métodos, como los de CPUE y análisis de población virtual complementan a los de acústica y permiten obtener un mejor panorama del estado del recurso y su vulnerabilidad a la pesca, brindándole a IMARPE mayores herramientas para un adecuado manejo del stock (M. Ñiquen, com. pers; Simmonds *et al*., 2008).

ANÁLISIS DE POBLACIÓN VIRTUAL (APV)

Una manera de estudiar el efecto de la pesca sobre los peces es a través de modelos de dinámica poblacional. Dentro de esta categoría se encuentran los métodos de APV que se basan en las estructuras demográficas por edades de las poblaciones marinas explotadas. Calcula la abundancia y la mortalidad por pesca para cada cohorte (grupo de peces de la misma edad) utilizando información previa sobre las capturas anteriores y la mortalidad natural (Laurec y Le Guen, 1981). Una ventaja para su utilización es que requiere pocos datos cuya recolección es fácil y poco costosa. No obstante, factores como el supuesto de mortalidad natural constante (o en su defecto el no considerar la variabilidad de la mortalidad natural y los factores que en ella influyen) y otros que mencionamos también para el caso de la CPUE (Fréon y Misund, 1999; y Sampson, 1988, quien muestra que la estabilidad en la estimación de los tamaños de cohortes pueden ser dramáticamente afectada por la magnitud de los errores en la mortalidad natural) son limitantes en las que debemos reparar al interpretar los resultados del APV. Teniendo todo esto en consideración, la información que el método puede brindar sobre el comportamiento de las cohortes complementada y contrastada siempre con las procedentes de hidroacústica y CPUE, es de gran utilidad por IMARPE para la evaluación de los stocks pesqueros pesqueros (Pauly et al., 1987; Ñiquen et al., 2000; Bouchon et al., 2000).

Ahora que el análisis espacial del esfuerzo pesquero y del ecosistema en general están teniendo mayor repercusión en el manejo de las pesquerías (Babcock *et al.*, 2005), nuevas tecnologías como el VMS (Vessel Monitoring System), se asoman como herramientas potentes para la obtención de información de muy buena calidad y a costos asequibles (Bertrand *et al.*, 2005; Deng *et al.*, 2005; Mills et al, 2007). El VMS, llamado en el Perú SISESAT (Sistema de Seguimiento Satelital), es un sistema de monitoreo continuo y en tiempo real de los desplazamientos de las embarcaciones. Estos se realizan a partir de equipos especiales colocados en los barcos que emiten señales satelitales de la posición de los mismos. En el Perú, este sistema está siendo utilizado desde 1999 y actualmente toda la flota industrial está equipada con transmisores satelitales que juntamente con las coordenadas de localización, indican la calidad o grado de precisión de cada una de estas. Los datos que brinda el VMS son de gran beneficio para el IMARPE y para la pesquería peruana en general; al permitir conocer el comportamiento espacial de las embarcaciones (actualmente la única información de este tipo es la que se obtiene mediante el programa Bitácoras de Pesca cuya capacidad de recojo de información es apenas del 2% de los viajes que realiza la flota entera en un día), se presentan como una herramienta valiosísima para el monitoreo de la explotación de los recursos marinos y realización de estudios en un enfoque ecosistémico de la pesquería[10] (Bertrand, 2005; Bertrand *et al.*, 2005, y 2008).

Así, Bertrand *et al.* (2005), desarrollan un algoritmo para reconocer las trayectorias de pesca (figura 2) a partir de los datos brutos de localización de las embarcaciones obtenidos por el VMS, y las comparan con la distribución espacial de la biomasa acústica de la anchoveta; utilizando luego la información de los viajes pesqueros para contrastar el comportamiento de los pescadores con los de predadores naturales o no-humanos (Bertrand *et al.*, 2007).

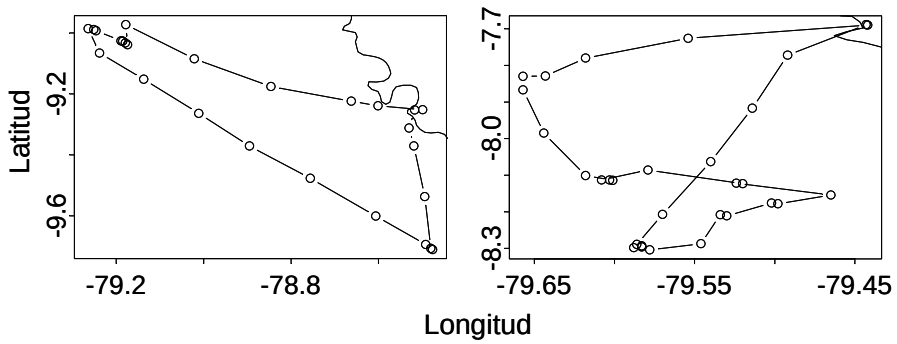


Figura 2. Ejemplos de trayectorias de pesca obtenidas con el algoritmo de reconocimiento de viajes pesqueros en base a datos de temporadas de pesca entre el 2000 y el 2002 y entre los 7ºS y 10ºS.

En base a los datos de VMS y el algoritmo de reconocimiento de viajes de pesca, el objetivo ahora es estimar las ubicaciones o coordenadas de operaciones de pesca[11].

A partir de los datos de localización (latitud, longitud) y tiempo (momento de la emisión) obtenidos del VMS se pueden derivar otros como la velocidad, el cambio de velocidad, la aceleración, el cambio de rumbo, que guardan relación con los comportamientos característicos de las embarcaciones durante una cala (operación de pesca).

Así, la velocidad es uno de los criterios más usados para identificar los momentos de pesca en la literatura al trabajar con datos de VMS, pues en ellos la velocidad es más baja que cuando el barco está en tránsito; con un rango de velocidades que varía de acuerdo al tipo de pesca y la zona (ver los diferentes criterios de velocidad de Deng *et al.*, 2005; Mills *et al.*, 2007; y Witt y Godley, 2007). Sin embargo, estas bajas velocidades no se dan sólo durante las calas, también pueden ser indicadoras

[10] Para una extensión sobre el enfoque ecosistémico y la sostenibilidad pesquera, ver FAO (1995), que implementa criterios ecosistémicos para la pesca responsable en su reglamento de conducta; Escobar (2001), quien desde la CEPAL discute el enfoque ecosistémico para la pesca en América Latina; y García *et al.* (2003), quienes desde la FAO revisan el enfoque ecosistémico desde su concepto e importancia, y discute las acciones para hacerlo operacional, evaluando los problemas que hasta el momento se encontraron.

[11] Nos referiremos a operación de pesca como el acto de disponer las redes en el mar y realizar la acción de pesca. Emplearemos la misma definición para la expresión "cala".

de garete (cuando el barco está yendo a la deriva), de búsqueda (de peces), aproximación a puerto, etc. Bertrand *et al.* (2008) realizaron una estimación utilizando únicamente el criterio de velocidad y encontraron una sobre-estimación del 182% del número de los puntos de pesca. Entonces, aunque la velocidad ayuda a descartar a los puntos en los que el barco está en tránsito, no debe ser el único criterio a utilizar.

Una forma de discriminar entre cala y garete en la pesquería peruana de anchoveta es mediante la hora. Generalmente, un barco está a la deriva en horas en las que no suele pescar, es decir, en altas horas de la noche o en las primeras horas de la mañana (figura 3).

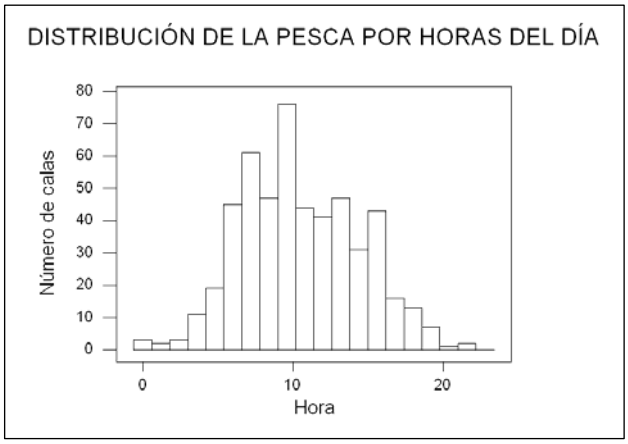
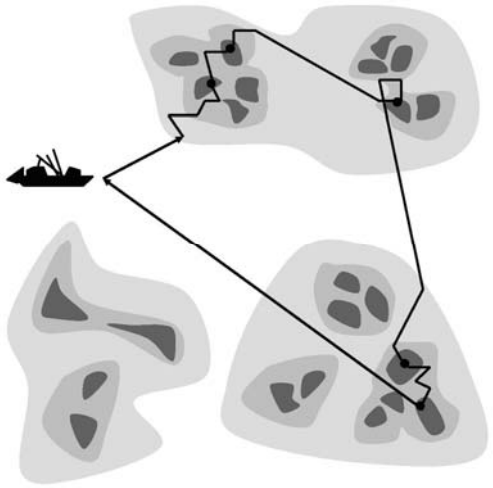


Figura 3: Distribución de la pesca por horas. Se utilizaron los datos recolectados por observadores a bordo en las temporadas de pesca 2005-2006.

Mills *et al.* (2007) señala algunos patrones de comportamiento de las embarcaciones relacionados a la operación de pesca, entre ellos los cambios de dirección (rumbo) bruscos después de haber pescado (figura 4); esto acrecentaría la probabilidad de encontrar nuevos cardúmenes cerca, pues la anchoveta no se distribuye uniformemente en el espacio, sino en la forma de agregaciones (distribución contagiosa o patchy).

Finalmente, la aceleración nos da otras pautas para la discriminación. Como cualquier depredador, cuando una embarcación ubica su presa, desacelera para prepararse a la captura. Cuando termina de pescar, vuelve a acelerar. Esto significa que la aceleración entre la ubicación anterior a la pesca y la de pesca será grande, al igual que la aceleración entre el punto de pesca y el siguiente a él.

Figura 4: Representación gráfica de la relación entre los cambios de rumbo de la embarcación y la pesca, para especies que forman agregaciones como la anchoveta. Fuente: Bertrand, 2005..



Se piensa que utilizando una técnica estadística adecuada, se pueden relacionar estas variables (variables independientes) de tal manera que brinden información suficiente para identificar los puntos de pesca (coordenadas geográficas de los datos del VMS para los que se llevaron a cabo las operaciones de pesca).

Las redes neuronales artificiales (ANN – siglas en inglés, Bishop, 1995; Jain y Mao, 1996; Hagan *et al.*, 1996; Demuth y Beale, 1998) son una técnica estadística (Warner y Misra, 1996; Tusell, 2005) que aprende, a partir de los datos, a encontrar la solución del problema (sea éste de modelamiento, predicción, clasificación, entre otros). Las ANN ofrecen ventajas y facilidades al no necesitar asumir funciones de densidad de probabilidad de las variables para la solución del problema, ser capaz de trabajar con grandes volúmenes de datos y de reconocer relaciones complejas y no-lineales en ellos. Esto ha hecho de las redes neuronales una técnica muy usada para el modelamiento, predicción y clasificación en ecología (Lek y Guégan, 2000; Park *et al.*, 2003) y en especial para la ciencia pesquera, donde las ANN han probado ser exitosas tanto para el modelamiento en el ciclo de vida de los peces, identificación (presencia/ausencia, detección/identificación, distribución, clasificación/discriminación), predicción de abundancia, etc., en diferentes especies e ubicaciones geográficas (ver Suryanarayana *et al.*, 2008, para una revisión más detallada de estos trabajos en pesquería aplicando redes neuronales).

La literatura en redes neuronales ofrece una variedad de opciones y de criterios para seleccionar la arquitectura de la red, la forma de entrenamiento, los algoritmos para ello, etc., lo cual puede ser rico pero a la vez confuso (Goethals *et al.*, 2007; Sarle, 2002). Por ello se debe hacer una profunda revisión de la literatura para tomar las mejores decisiones posibles respecto a la red, y ejecutar siempre pruebas de sensibilidad que avalen los resultados obtenidos y permitan conocer la robustez o sensibilidad de estos frente a cambios en la estructura de la red o en el proceso de entrenamiento.

Debido a las ventajas previamente mencionadas y a sus buenos resultados obtenidos en aplicaciones relacionadas a la pesca, Bertrand *et al.* (2008) proponen el uso de las ANN para el reconocimiento de los puntos de pesca de embarcaciones anchoveteras utilizando las variables descritas anteriormente, obtenidas de una muestra de datos del VMS de temporadas de pesca entre los años 2000 y 2002 en aguas costeras peruanas entre 7ºS y 10ºS. Entrenaron la red con una muestra de viajes pesqueros con posiciones de pesca conocidas (proporcionadas por observadores a bordo) para predecir la respuesta (cala/no cala) y obtuvieron un 83% de clasificación correcta, mayor que con un modelo de regresión lineal generalizado (65% de clasificación correcta).

2. Problematización o planteamiento del problema

Los problemas anteriormente descritos así como los avances realizados para un mayor conocimiento y mejor control del esfuerzo pesquero, hacen considerar seriamente a la red neuronal como una alternativa potente para la identificación de calas. Sin embargo, las ANN presentan una gama de opciones en cuanto a su estructura, entrenamiento y otras, cuyos cambios pueden variar mucho sus resultados (o puede que no sucede así). Además, la red necesita entrenarse a partir de una base de datos. Poseemos una base de datos del 2000 al 2006, pero puede que particionándola obtengamos mejores resultados. Es desde estos cuestionamientos que formulamos las preguntas de investigación:

¿Qué tan sensible será la red frente a los cambios en sus parámetros? ¿Cuáles serán sus valores óptimos?

¿Qué tan sensible será la red a los cambios de base de entrenamiento y al período de la base (bianual, trianual)? ¿Cuál será la base más indicada para entrenar la red?

¿Cuál es la propuesta, diseño y estructura final de la red neuronal más adecuada para la identificación de las calas?

3. Objetivos

En esta misma línea, los objetivos de este trabajo son:

General:

Realizar la propuesta final de una red neuronal que permita identificar puntos de pesca (o cala) de anchoveta en el mar peruano.

Específicos:

- Definir los parámetros a testear.
- Cuantificar el desempeño de la red y sus variaciones al cambiar los parámetros mediante indicadores de eficacia y eficiencia.
- Realizar el análisis de sensibilidad.
- Encontrar los valores óptimos para los parámetros de la red.
- Comparar los resultados para las diferentes bases temporales y encontrar la base más indicada para entrenar la red.

CAPÍTULO II
MARCO TEÓRICO

1. Introducción

Las redes neuronales artificiales (ANN – siglas en inglés de Artificial Neural Network) son una clase de técnica estadística no-paramétrica.

En un enfoque paramétrico se infiere de la estructura de los datos de observación un modelo analíticamente explícito. Esto otorga ventajas para la interpretación de los procesos que pueden generar los patrones observados y para la extrapolación a nuevos casos afuera de las observaciones. El modelo paramétrico trata de corresponder al "mejor"[12] modelo posible. No obstante, a veces no es posible o significa costos altos de esfuerzo y tiempo encontrar el modelo más apropiado; o, en algunas otras, la expresión analítica de este puede ser poco parsimoniosa y/o poco objetiva (en términos de capacidad de generalización).

En estos casos, los modelos no-paramétricos, que asumen muy poca información sobre la estructura del modelo, como por ejemplo, sobre las funciones de distribución de probabilidad para los parámetros, son comúnmente utilizados. Un primer problema con estos modelos es la estimación de sus parámetros, por la poca o nula información sobre su estructura que se posee. Estos parámetros deben ser estimados partiendo de los datos, mayormente mediante rutinas computacionales muy intensas. Un segundo problema está relacionado a su multidimensionalidad. Es decir, sobre el grado de dimensionalidad del modelo. Con dimensiones mayores, mayor porcentaje de variabilidad es explicada por el modelo, mas los datos pueden volverse dispersos y el diseño del modelo más difícil si la dimensión espacial de entrada (input) crece demasiado. Otro problema, considerado el más severo, es la difícil interpretación del ajuste a un modelo no paramétrico. Una aproximación posible es la incorporación de cierta estructura en los modelos no paramétricos. Lo cual es, como se verá más adelante, lo que sucede con la técnica de Redes Neuronales Artificiales, para la que se fija parte de la estructura, y para el resto, se va aprendiendo a construirlo en el proceso, a través de los datos.

Se ha dicho que las ANN son una herramienta informática, perteneciente a la inteligencia artificial. La inteligencia artificial es una ciencia e ingeniería que crea programas computacionales inteligentes - máquinas inteligentes, en general, y parte de esa inteligencia está inspirada en la inteligencia humana[13] Sabemos que sin las ventajas computacionales, la técnica sería totalmente ineficiente debido a los costos en tiempo y esfuerzo para ejecutar la red –y todo el proceso de aprendizaje que lleva dentro –y obtener los resultados de predicción o clasificación. Pero la potencia que la computación le otorga no lo hace herramienta de la informática, más bien, se la puede considerar enriquecida por esta última.

1.1. Red neuronal biológica: la inspiración

Aunque desde el punto de vista estadístico no se trata de imitar el comportamiento del cerebro humano, las redes neuronales biológicas fueron quienes inspiraron en un principio el desarrollo de las redes neuronales artificiales. Ha sido y continúa siendo objeto de estudio la capacidad del cerebro humano para aprender, adaptarse a las nuevas situaciones, extraer de su memoria la información necesaria para el momento y sobreponerse a ciertos daños locales que pueda sufrir, para tomar decisiones de diferente tipo -desde reconocimiento de imágenes simples y claras hasta cálculos muy complejos (figura 5) .

[12] Ver más adelante una discusión relacionada al uso de esta expresión.
[13] John McCarthy, Professor Emeritus of Computer Science at Stanford University, principal pionero de la inteligencia artificial. http://www-formal.stanford.edu/jmc/

EL "MEJOR" MODELO

Cuando hablamos de mejor modelo, nos referimos a que no existe "el" modelo; es decir, no existe un modelo que explique completa y perfectamente bien la realidad. Matheron (1978) decía que "no hay probabilidad en sí. No hay más que modelos probabilistas. La única pregunta que se formula realmente, en cada caso particular, es la de saber si tal modelo probabilista, en relación con tal fenómeno real, presenta o no un sentido objetivo". La probabilidad, dirá él, es de naturaleza matemática, y como tal, no es parte de la realidad. Son los modelos probabilísticos –o probabilistas, en los que se aplica la teoría de probabilidades –los que se relacionan con la realidad. A partir de esta afirmación, distingue 2 tipos de objetivos relacionados a estos modelos: científicos y prácticos. Los primeros se refieren al conocimiento científico en sí como objetivo general, mientras que para los segundos interesa dar una solución o presentar buenas alternativas de solución a un problema particular de la vida real en miras a una importante toma de decisiones.

Es esto último lo que nos interesa y es importante tenerlo siempre presente, porque muchas veces –en problemas específicos de diferentes materias –terminamos buscando "el modelo verdadero" (Klinke y Grassmann, 1998) o "el modelo correcto". Aquel que represente correctamente la realidad de tal manera que nos permita rechazar los otros y afirmar que el nuestro es "el modelo". Olvidamos que el modelo, por definición, es la representación -de manera más simple –de una realidad -por lo general muy compleja ("Un modelo nunca dice toda la verdad" – J. Simmonds, com. pers.). Y que el componente probabilístico no implica que la probabilidad en sí exista en la realidad, sino que la realidad puede ser representada por un modelo –o incluso varios modelos –que haga operativa la teoría de la probabilidad. Es decir que el modelo probabilístico es una herramienta, para la cual se debe señalar bien en qué medida y hasta qué punto puede representar la realidad y permitir responder una pregunta de investigación particular.

En ese sentido, debemos recordar que el objetivo nuestro es identificar o reconocer los puntos de cala a partir de los patrones de datos del VMS. No es definir qué variables son las que explican totalmente el problema ni precisar las relaciones que debe haber entre ellas de manera que podamos predecir una cala. Si pudiésemos lograr esto último sería muy bueno pero no es la prioridad –aunque no se puede explicar o resolver totalmente un problema, justamente por ello es probabilístico; lo que se podría aspirar a lograr, al fin y al cabo, sería una "muy buena" explicación.

Siguiendo esta línea de pensamiento, el empleo de una técnica como las redes neuronales artificiales, se centra en el objetivo del problema y no intenta buscar la "mejor" expresión analítica, sino encuentra una que ayude a resolverlo. Por supuesto, las ANN no son la panacea. Una herramienta estadística cuando busca ser aplicada es sólo eso: una herramienta; no una meta en sí. La alternativa que aquí se ofrece es la que hasta ahora brinda los mejores resultados; para la cual se han realizado pruebas de sensibilidad que permiten presentar más objetivamente sus capacidades y limitaciones, y hacer un mejor uso de ella.

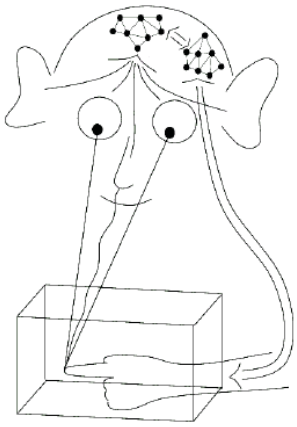


Figura 5. Sistema de control sensitivo motor. Fuente: Computacional Neuroscience. Feng (ed.). Chapman & Hall/CRC, 2004

Por ello, nos parece importante explicar a grandes rasgos el comportamiento de las neuronas del cerebro humano, que inspiraron el desarrollo de la técnica en sus principios.

El cerebro humano está formado por aproximadamente 10^{11} células nerviosas altamente interconectadas (aproximadamente 10^4 conexiones por elemento) llamadas neuronas. Cada una de ellas tiene dendritas, un cuerpo y un axón. Las dendritas son como ramificaciones de fibras nerviosas que llevan señales eléctricas hacia el cuerpo de la célula. El cuerpo suma y regula mediante umbrales estas señales entrantes (también regula las sustancias de salida). El axón es una fibra larga única que lleva las señales desde el cuerpo de la célula hacia otras neuronas o células (musculares por ejemplo). El punto de contacto entre el axón de una célula y la dendrita de otra célula es llamado sinapsis. Las sinapsis pueden ser excitatorias o inhibitorias. El arreglo (orden) de las neuronas y las fortalezas de cada sinapsis individual – determinadas por un proceso químico complejo – son los que finalmente establecen la función de la red neuronal. La figura 6 trata de ser una ayuda gráfica para comprender este aspecto.

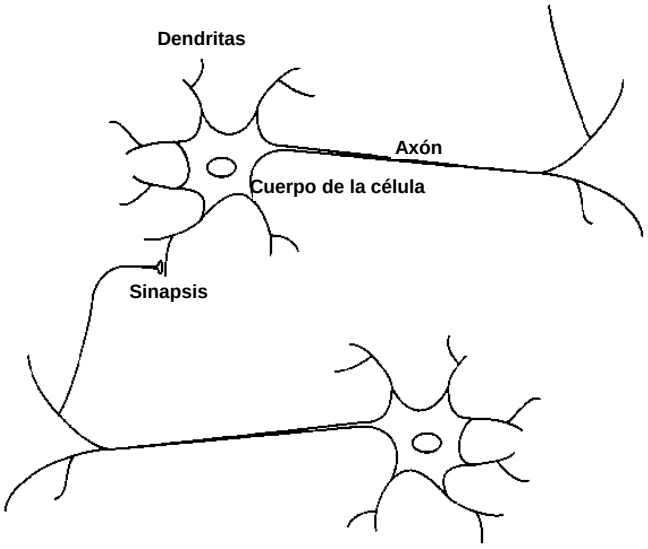


Figura 6. Dibujo esquemático simplificado de dos neuronas biológicas. Fuente: Neural Network Design – Hagan et al., Demuth y Beale, Beale, 1996

Parte de la estructura neuronal es definida en el nacimiento. Otras partes son desarrolladas durante el aprendizaje, y nuevas conexiones se establecen mientras otras se desechan. Aunque este desarrollo se advierte más en las primeras etapas de la vida, las estructuras neuronales continúan cambiando durante el resto de la existencia. Estos últimos cambios tienden a consistir principalmente en el fortalecimiento o debilitamiento de las uniones sinápticas. Por ejemplo, el proceso de aprender la cara de un nuevo amigo consiste en la alteración de varias sinapsis (Hagan *et al.*, 1996).

Las ANN tratan de simular el funcionamiento de las redes neuronales biológicas en
1) su proceso de aprendizaje: a partir de ciertos estímulos (variables de entrada) y realización de operaciones entre sí (sinapsis), mediante funciones de activación, sintetiza una o varias respuestas (variables respuestas);
y 2) dentro del proceso de aprendizaje, los pesos sinápticos o fuerza con que están interconectadas las neuronas, que van cambiando en el proceso, se utilizan para almacenar la información (Aldabas-Rubira, 2002).

1.2. Utilidad de la red neuronal artificial

Algunas razones por las que las redes neuronales aparecen como la técnica más adecuada (las de nuestro caso de estudio), son:

- En un principio, no hay información suficiente para plantear un modelo que se ajuste a los datos y pueda predecir correctamente.
- No tener la necesidad de conocer la expresión analítica del modelo, sólo es importante la predicción/clasificación.

En resumen, se recurre a las ANN para resolver problemas de clasificación o predicción de alta complejidad (Aldabas-Rubira, 2002).

2. Estructura y funcionamiento

Una red neuronal, como su nombre lo indica, es una red de neuronas interconectadas. La red neuronal artificial es una representación de una red neuronal biológica. Las neuronas son unidades constituidas por expresiones matemáticas (explicadas más adelante), y sus interconexiones se realizan por medio de funciones llamadas *funciones de activación*.

Warner y Misra (1996) describen los componentes de la red neuronal y su funcionamiento. Cada una de las unidades o neuronas son también llamadas nodos[14], que, en la figura 7 son representadas por círculos. Sus conexiones, que intentan modelar las conexiones sinápticas en el cerebro, son representadas por flechas. Cada conexión tiene un peso – llamado peso sináptico – asociado a ella, denotado por w_{ij}. Este peso sináptico es interpretado como la fuerza de la conexión entre las unidades i-ésima y j-ésima (la fuerza de la sinapsis entre la neurona i y la neurona j).

Todas las redes neuronales poseen un nivel de entrada – compuesto por las variables de entrada (inputs) – y una capa de salida (output layer). Algunas ANN tienen capas que se ubican entre las dos anteriores, llamadas capas escondidas (hidden layers). Para hablar del número de capas, hay ambigüedad en la literatura, pero generalmente el nivel de entrada no se cuenta como capa porque está compuesto por las variables de entrada que no han sido modificadas por la red. Ya que la mayoría de autores lo asumen de esa manera, aquí también contaremos el número de capas a partir de las escondidas (aún cuando en algún momento nos refiramos a la capa de entrada). Así, la red de la figura 7 tiene 2 capas: una escondida con 3 nodos y una de salida con 2; además posee 4 nodos en el nivel de entrada.

A continuación revisaremos la tipología de las redes neuronales, sus tipos de aprendizaje y sus funciones de activación, para luego profundizar en el detalle de las mismas, especialmente la multi-capas, por su aplicación en el presente estudio.

[14] De aquí en adelante se utilizarán indistintamente los términos “nodo” y “neurona”, o simplemente “unidad” para mencionar a las unidades que componen la red.

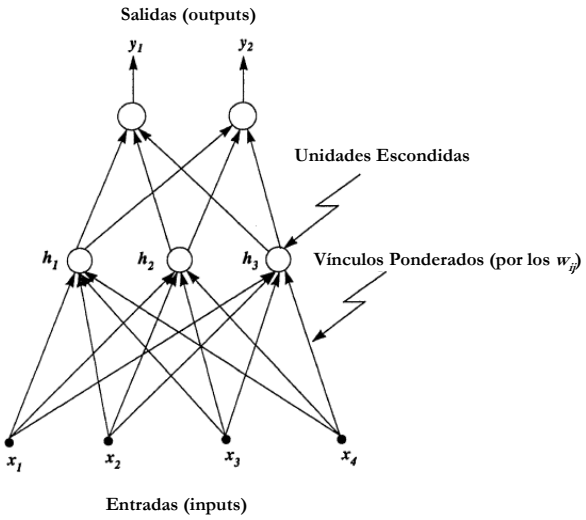


Figura 7. Representación de una red neuronal. Fuente: Computacional Neuroscience. Feng (ed.). Chapman & Hall/CRC, 2004

2.1. Tipología de las arquitecturas de las ANN

En la figura 8 se puede observar más claramente la clasificación de estas arquitecturas de redes, así como sus subdivisiones de acuerdo a características más específicas en las relaciones entre los nodos.

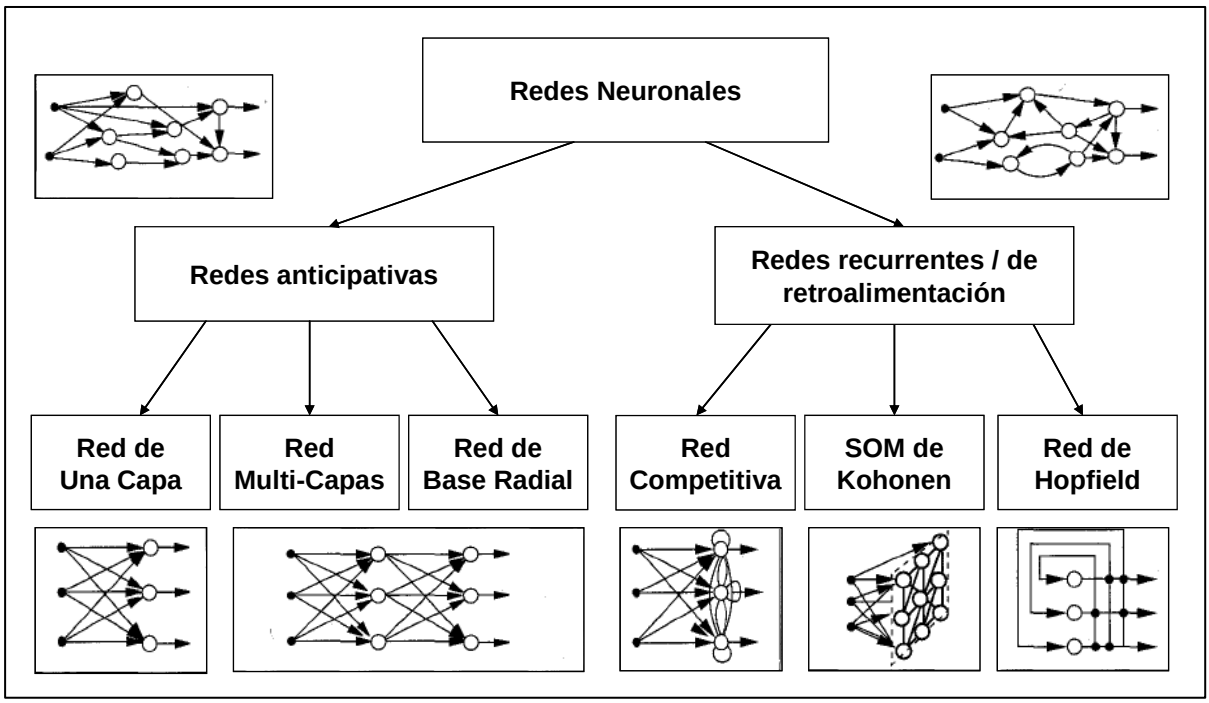


Figura 8. Una tipología de las arquitecturas de redes de feed-forward y recurrentes/feedback. Fuente: Artificial Neural Networks: a Tutorial. Jain y Mao, 1996.

2.1.1. Redes anticipativas (feed-forward networks):

Una red anticipativa (feedforward network) es una donde la conexión de los nodos es únicamente hacia delante; es decir, los nodos de una capa se conectan unidireccionalmente con los nodos de la siguiente capa. Las unidades de una misma capa no se ligan entre sí directamente. (Warner y Misra, 1996)

Dentro de las redes anticipativas tenemos:

- Redes de una sola capa
- Redes multi-capas
- Redes de funciones de base radial

entre las más conocidas.

Las redes anticipativas son más fáciles de entrenar que las recurrentes o de retroalimentación (Sarle, 2002).

2.1.2. Redes recurrentes o de retroalimentación (recurrent/feedback networks):

Son aquellas redes donde las conexiones entre los nodos puede ser bidireccional o unidireccional (en cualquier sentido); incluso las neuronas pueden conectarse a ellas mismas. (Warner y Misra, 1996)

Dentro de las redes recurrentes/de-retroalimentación se incluyen (entre otras):

- Redes competitivas
- Mapas auto-organizativos de Kohonen
- Redes Simples de Elman
- Redes de Hopfield

2.2. Paradigmas del aprendizaje (Warner y Misra, 1996; Jain y Mao, 1996)

Así como el cerebro aprende mediante el fortalecimiento o debilitamiento de las conexiones sinápticas; los pesos sinápticos de las ANN se ajustan para solucionar el problema presentado a la red. En esto se parece a los coeficientes de los modelos de regresión.

Aprendizaje o entrenamiento es el término utilizado para describir el proceso de encontrar los valores de estos pesos. Los 3 tipos de aprendizaje (paradigmas del aprendizaje) de las ANN son: aprendizaje supervisado, no-supervisado e híbrido.

a) Aprendizaje Supervisado

El aprendizaje supervisado, también llamado aprendizaje "con profesor", ocurre cuando existen valores conocidos del objetivo (target), cada uno asociado con cada entrada a la base de datos de entrenamiento. Este valor conocido es como la "respuesta correcta" para esos patrones de entrada. La salida de la red es entonces comparada con el valor del objetivo, y esta diferencia es usada para entrenar la red: alterando los pesos, de manera que permitan que la red produzca respuestas que se acerquen lo más posible a las respuestas correctas. Existen muchos algoritmos para entrenar las redes neuronales usando aprendizaje supervisado: el de retro-propagación es uno de los más comunes, y será desarrollado más adelante. Un ejemplo biológico del aprendizaje supervisado se da cuando a un niño se le enseña el alfabeto. Se le enseña una letra, y basada en su respuesta, se retroalimenta al niño de tal manera que pueda aprender la letra (que viene a ser la "respuesta correcta"). Este proceso se repite para cada letra hasta que el niño conozca el alfabeto.

b) Aprendizaje No-Supervisado

El aprendizaje no-supervisado o aprendizaje "sin profesor", es necesario cuando los datos de entrenamiento no poseen estas "respuestas correctas" asociadas a los patrones[15] de entrada. Por ello lo que hace la red es explorar la estructura subyacente a los datos y aprender a agrupar o realizar clusters de los patrones de entrada basados en características comunes, algo similar al análisis factorial o a los componentes principales. Un ejemplo clásico –sobre todo en la ciudad de Lima –es la manera de aprender a ser precavidos frente a los robos, secuestros, etc. Alguien puede estar caminando por la calle y escoger pasar o no pasar por una cuadra de acuerdo a la hora, la oscuridad,

[15] En el lenguaje de redes neuronales, el término "patrón" se refiere a lo que en estadística llamamos "observación". Así pues, un patrón de entrada está compuesto por valores particulares en cada una de las variables de entrada. En el documento, nos referiremos indistintamente a patrones y a observaciones.

la zona, etc., basándose en información externa o experiencias propias que no necesariamente son acerca de esa cuadra específica, pero que de alguna manera ayudan a formar criterios (especie de perfiles) de lugares por los cuales no creemos que sea aconsejable caminar. Un ejemplo de algoritmo para este tipo de aprendizaje es el de conglomerados por K-medias.

c) Aprendizaje Híbrido

El aprendizaje híbrido combina el aprendizaje supervisado y el no-supervisado. Aquí, una etapa del entrenamiento se realiza mediante aprendizaje supervisado mientras que otra, a través del aprendizaje no-supervisado.

2.3. Funciones de activación

Una función de activación es una función que relaciona las entradas de la red con un nodo de esta, convirtiendo a las primeras en un escalar. La función de activación es escogida por el o la diseñadora de la red. Para cada capa o nivel, esta función puede cambiar. Estas funciones están agrupadas en 4 clases: de umbral, lineal compuesta, sigmoidal y gaussiana (figura 9). Se describirán a continuación las 4 clases y algunas de las funciones que pertenecen a ellas (ver también tabla 1).

a) Funciones de Umbral:

Son aquellas que clasifican directamente a la unidad de salida. Dentro de esta categoría tenemos, por ejemplo, a la función de límite fuerte y de límite fuerte simétrica.

b) Funciones Lineales Compuestas:

Esta clase de función es llamada compuesta, porque puede estar conformada por dos o más funciones, aunque se considera también dentro de la categoría actual a las funciones simples también. Es lineal porque se compone de funciones lineales (es decir, de grado uno como máximo). Dentro de ellas están la función lineal, la lineal positiva, la lineal saturada y la lineal saturada simétrica, la base triangular, entre otras.

c) Funciones sigmoidales:

Una función sigmoidal es llamada así porque produce una curva con la forma de una “S”. En general, una función sigmoidal tiene su dominio en los números reales y es diferenciable, y sus derivadas de primer orden son no-negativas o no-positivas, teniendo un único punto de inflexión. Algunas de las funciones que pertenecen a esta categoría son: la logística, el arco tangente y la tangente hiperbólica.

d) Funciones Gaussianas:

Una función gaussiana (cuyo nombre se debe a Carl Friedrich Gauss) es una función matemática de la forma:

$$f(x) = ae^{-\frac{(x-b)^2}{2c^2}},$$

donde $a>0$, b y $c>0$ son constantes reales.

Gráficamente (figura 9d), el parámetro a es la altura del pico gaussiano, b es la posición del centro del pico y c controla el ancho del “bulto”.

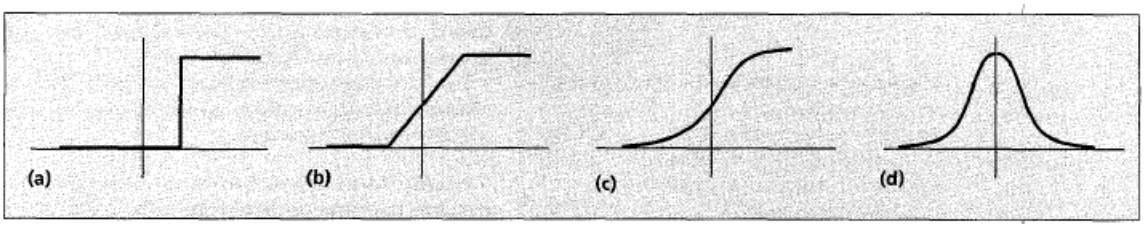


Figura 9. Representación gráfica de los 4 tipos de funciones de activación: a) de umbral; b) lineal compuesta; c) sigmoidal; y d) gaussiana. Fuente: Artificial Neural Networks: a Tutorial. Jain y Mao, 1996.

En la tabla 1, podremos ver una lista de funciones de activación, comúnmente usadas para las ANN. Siempre existe la posibilidad de que se puedan definir nuevas funciones de activación que se acomoden mejor a la realidad del problema de investigación.

2.3.1. Elección de la función de activación

La elección de la función de activación depende de nuestro objetivo detrás del uso de la red neuronal y de las características de las variables de entrada y salida. Si la aplicación de la red es para la aproximación a una función, entonces la elección de la función de activación dependerá de la función a la que se quiere aproximar la variable de salida. También depende de la clase y escala de medición de las variables de entrada y salida. Por ejemplo, funciones de activación como la logística, la tangente hiperbólica sigmoidal y la de base radial sólo se utilizan cuando sus entradas son continuas.

Si lo que se necesita es tomar decisiones de clasificación entre dos categorías, para hacerlo directamente se puede elegir como función de activación (hacia la capa de salida) una función de umbral. ¿Cuál de ellas? Solo depende de la respuesta que deseamos obtener: dicotómica 0 y 1 (Límite Fuerte) o dicotómica -1 y +1 (Límite Fuerte Simétrica).

Para objetivos con valores continuos, se aconseja utilizar una función de activación lineal, a menos que hayan otros criterios que conduzcan a una elección de función diferente (Sarle, 2002). Las funciones de activación lineales han sido usadas también para aproximar una regresión lineal (Bueche, 2005). Las funciones compuestas como la lineal positiva, saturada y saturada simétrica limitan la región del dominio (los valores de las entradas), eliminando algunas que pueden generar ruido para el modelo. Ejemplo: la utilización de filtros adaptativos lineales para la cancelación del eco (Hagan *et al.*, 1996). Nuestra elección entre ellas, depende de dónde queremos colocar los límites superior e inferior ($x<-1$; $x<0$; $x>1$; etc.). Cuando el objetivo presenta patrones con un rango limitado de valores, se pueden utilizar las funciones logísticas o tangenciales, re-escalando las salidas o los objetivos, para que ambos estén en la misma escala (Sarle, 2002).

Cuando las entradas son continuas y se tienen objetivos binarios (0 y 1), es preferible utilizar funciones de activación logísticas, cuyo resultado (el nodo de salida) puede ser interpretado como una probabilidad a posteriori (dadas las entradas) de que el patrón de salida pertenezca a una u otra categoría (Jordan, 1995). Sarle (2002) sostiene que es preferible usar sigmoidales a umbrales porque las primeras hacen que la red sea más fácil de entrenar. Además, como veremos más adelante, Bishop (1995) afirma que un nodo escondido sigmoidal (con función de activación sigmoidal) puede aproximar muy bien un nodo escondido lineal (con función de activación lineal). Si se utilizan funciones sigmoidales en la red, se sugiere que para los nodos escondidos sean tangentes o tangentes hiperbólicas pues al tener valores negativos y positivos, tienden a lograr aprendizajes más rápidos (convergencias más rápidas de la red) que con funciones logísticas (cuyos valores son no negativos) (Sarle, 2002 y 1999).

Si los valores del objetivo son positivos sin límite superior, se puede utilizar una función gaussiana, siempre y cuando sea posible transformar el objetivo y la salida de manera que tengan una forma gaussiana.

FUNCIONES DE ACTIVACIÓN

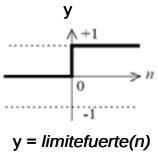
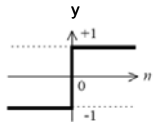
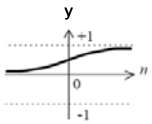
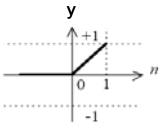
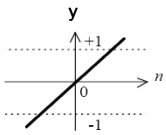
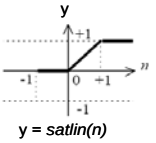
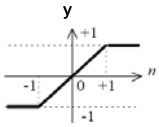
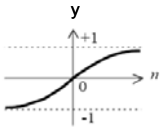
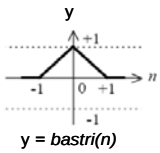
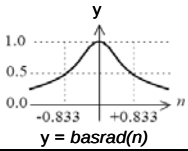
NOMBRE	RELACIÓN ENTRADA/SALIDA	ÍCONO	GRÁFICO
Límite Fuerte	y=0; n<0 y=1; n≥0		 y = *limitefuerte(n)*
Límite Fuerte Simétrica	y=-1; n<0 y=+1; n≥0		 y = *limitefuertesim(n)*
Log sigmoidal	$y = \frac{1}{1+e^{-n}}$		 y = *logsig(n)*
Lineal Positiva	y=0; n<0 y=n; n≥0		 y = *poslin(n)*
Lineal	y=n		 y = *lin(n)*
Lineal Saturada	y=0; n<0 y=n; 0≤n≤1 y=1; n>1		 y = *satlin(n)*
Lineal Saturada Simétrica	y=-1; n<0 y=n; -1≤n≤1 y=1; n>1		 y = *satlinsim(n)*
Tangente Hiperbólica Sigmoidal	$y = \frac{e^{n} - e^{-n}}{e^{n} + e^{-n}}$		 y = *tansig(n)*
Base Triangular	y=0; n<-1 ∨ n>1 y=n+1; -1≤n<0 y=1-n; 0≤n≤1		 y = *bastri(n)*
Base Radial	$y = e^{-n^2}$		 y = *basrad(n)*

Tabla 1. Funciones de Activación; donde y es la salida del nodo y n es la entrada al mismo. Fuentes: Neural Network Design – Hagan et al., 1996 y Neural Network Toolbox for use with Matlab – Demuth y Beale, 1998

2.4. ANN de un nodo (en una única capa)

Consideraremos primero una red con una sola capa con un solo nodo:

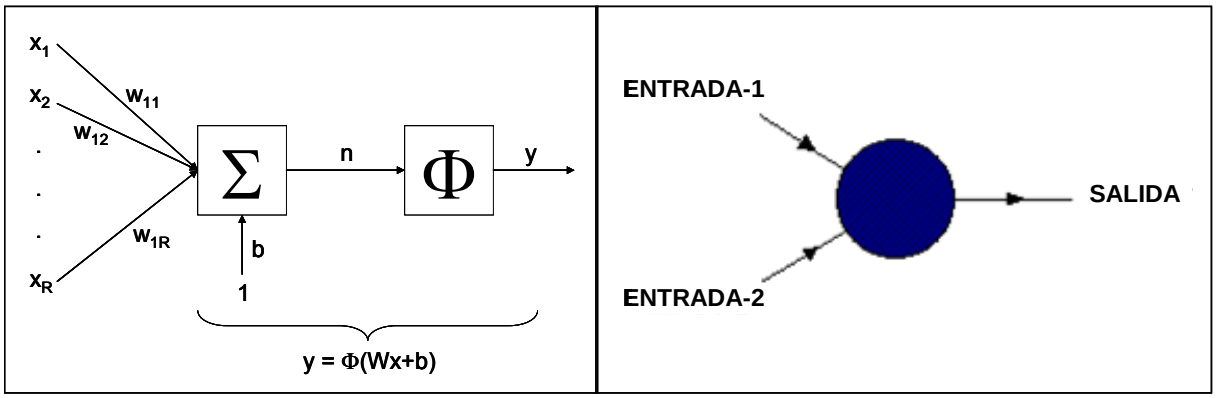


Figura 10. Representación de una red de una capa con un solo nodo. A la derecha, un diseño más simplificado en el que se observan 2 variables de entrada y sus conexiones con el nodo de salida. Fuentes: Neural Network Toolbox for use with Matlab – Demuth y Beale, 1998 y AlanTuring.net - Copeland y Proudfoot, 2000

En el panel izquierdo de la figura 10, las variables de entrada están representadas por los x_i, cuyo vector es *x*. Los pesos (w_{1j}) están representados en conjunto por la matriz *W* (el primer índice de *w*, "1", quiere decir que están relacionados al nodo 1). *b* es el sesgo[16], un escalar al que para propósitos prácticos se le considera como "peso" de la constante "1". Σ es la suma de los productos internos de los *w* y los *x*,

$$\Sigma = \sum_{j=1}^{R} w_{1j} \times x_j \quad \ldots(1),$$

donde *R* es el número de variables de entrada.

n es la suma de Σ y *b*1*, es decir,

$$n = \Sigma + b \times 1 = \sum_{j=1}^{n} (w_{1j} \times x_j) + b = Wx + b \quad \ldots(2),$$

n es la entrada[17] para el nodo.

Seguidamente, a *n* se le aplica una función de activación, representada por Φ.

La salida, para una red de una sola neurona está representada por *y*:

$$y = \Phi(n) \quad \ldots(3)$$

Si relacionamos este modelo simple al de neuronas biológicas que se discutió anteriormente, el peso w_{ij} corresponde a la sinapsis entre las células *i* y *j*, la suma y la función de activación representan al cuerpo de la neurona, y la salida *y*, a la señal que lleva el axón (Hagan *et al.*, 1996). Un peso w_{ij} positivo representa a una sinapsis excitatoria; mientras uno negativo, a una inhibitoria.

[16] En el lenguaje de redes neuronales, el término "sesgo" es análogo al de "intercepto" para el lenguaje estadístico, si la expresión analítica de una regresión lineal. (Sarle, 1996)

[17] Cuando mencionemos los términos "entrada" o "salida" nos estaremos refiriendo a "variable de entrada (input)" o "variable de salida (output)", respectivamente.

2.5. Red de una capa (SLN – single layer network)

Ahora consideremos una red con una capa de varias neuronas (por ser una sola capa, será una capa de salida).

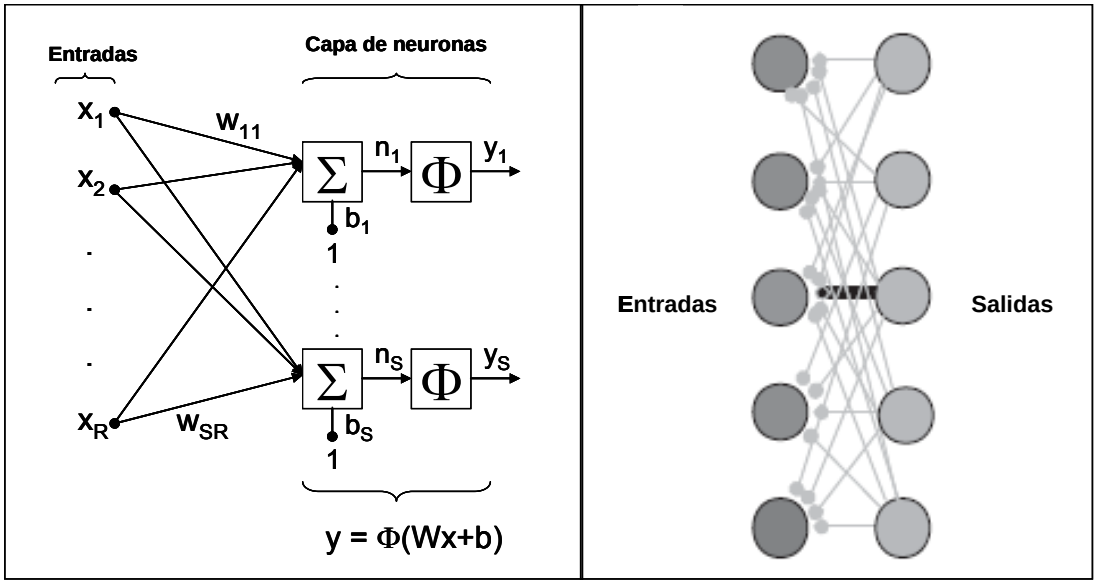


Figura 11. Representación de una red de una sola capa. A la derecha, un diseño más simplificado en el que se observan las variables de entrada y sus conexiones con la capa de neuronas. Fuentes: Neural Network Toolbox for use with Matlab – Demuth y Beale, 1998 y Computacional Neuroscience. Feng (ed.). Chapman & Hall/CRC, 2004

En el panel izquierdo de la figura 11, los pesos w_{ij} relacionan al nodo *i* con su entrada *j*; b_i es el sesgo en la entrada para el nodo *i*.

Entonces,

$$n_i = \sum_{j=1}^{R} (w_{ij} \times x_j) + b_i \quad \ldots(4),$$

donde *R* es el número de variables de entrada,

representa la entrada de la red al nodo *i*,

cuyo valor de salida es

$$y_i = \Phi(n_i) \quad \ldots(5),$$

siendo Φ la función de activación.

En forma matricial, podemos representar a *y* como:

$$y = \Phi(Wx + b) \quad \ldots(6),$$

donde

$$W = \begin{bmatrix} w_{11} & w_{12} & \cdots & w_{1R} \\ w_{21} & w_{22} & \cdots & w_{2R} \\ \vdots & \vdots & \ddots & \vdots \\ w_{S1} & w_{S2} & \cdots & w_{SR} \end{bmatrix}$$

indicándose el nodo de destino mediante el índice de fila y la variable de entrada mediante el índice de columna; siendo w_{ij} la fuerza de la conexión entre el nodo *i* y la entrada *j.*

2.5.1. Aprendizaje

Recordemos primero que el objetivo de la red es minimizar la función del error. A esta función la podemos expresar por diferencias de error al cuadrado de la siguiente forma:

$$E = \frac{1}{2}\sum_{m=1}^{p}(y^m - t^m)^2 \quad \ldots.(7)$$

Donde *E*: Función del error
y^m*:* salida (patrón de salida) *m-ésimo*,
t^m*:* objetivo (target) *m-ésimo,*
p: el número total de observaciones (patrones de entrada) de la base de datos

Esta es la forma más simple de la función del error. Utilizándola, podemos aplicar técnicas de mínimos cuadrados para el entrenamiento de la red. Una de las más comunes es usando el algoritmo de gradiente descendiente.

a) Gradiente Descendiente

El gradiente descendiente trata de encontrar los pesos que mejor ajusten las salidas a los objetivos; y es la base del algoritmo de retro-propagación.

Si añadimos una primera fila al vector *x*, cuyo elemento es "1", y consideramos a *b* como una primera columna de *W*, podemos decir que *y* es una función que depende únicamente de *x* y *W*:

$$y = y(x;W) \ldots(8)$$

Y más específicamente, que $y^m = y(x^m;W)$, es decir que la salida (o el vector de salida) para cada observación, se puede representar como una función que depende de x^m (el vector de entrada para esa observación) y *W.*

Así, la ecuación (7) se puede reescribir de la siguiente manera:

$$E = \frac{1}{2}\sum_{m=1}^{p}\left(y(x^m;W) - t^m\right)^2 \quad \ldots(9),$$

Y, habiendo agrupado a todos los parámetros (pesos y sesgos) juntos en la red a través de un único vector *W*, la función del error puede ser expresada como $E = E(W)$.

Esta función del error viene a ser por lo tanto, una función de suavización de los parámetros de peso w_{ij}, pudiendo ser minimizada por una variedad de técnicas estándar; siendo una de las más utilizadas y simples, la del gradiente descendiente.

Si la función de activación Φ es diferenciable, entonces *E* también es una función de *W* diferenciable y se puede realizar el siguiente procedimiento:

Lo que queremos es determinar el vector de pesos que minimice el error E. Esto se logra alterando los pesos en la dirección que produce el máximo descenso en la superficie del error.

La dirección de cambio se obtiene mediante el gradiente (∇). El gradiente especifica la dirección que produce el máximo incremento, por lo que el mayor descenso es el negativo de esa dirección $(-\nabla)$.

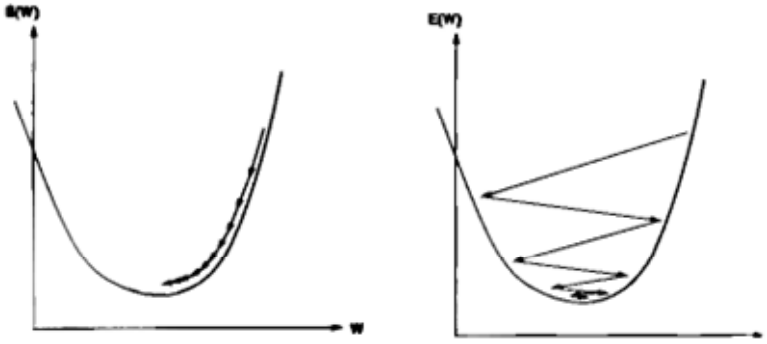
Se empieza con un valor inicial para *W* (que puede ser, por ejemplo, un valor escogido aleatoriamente) y luego se actualiza el vector de pesos moviendo una distancia pequeña en el espacio-*W* en la dirección en la que *E* decrece más rápidamente, es decir, en la dirección de $-\nabla_W E$.

Al hacer de este un proceso iterativo, generamos una secuencia de vectores de pesos $W^{(\tau)}$ cuyos componentes son calculados usando

$$w_{ij}^{(\tau+1)} = w_{ij}^{(\tau)} - \eta \left.\frac{\partial E}{\partial w_{ij}}\right|_{W^{(\tau)}} \quad \ldots(10)$$

Donde η es un número positivo pequeño llamado parámetro de *ratio de aprendizaje*, y τ es el paso temporal en el algoritmo iterativo.

Bajo condiciones apropiadas, la secuencia de vectores de pesos convergerá a un punto en el que *E* sea minimizado. La elección del valor para η puede ser crítica, dado que si es demasiado pequeño, la reducción en el error será muy lenta, mientras que, si es demasiado grande, el algoritmo puede resultar en oscilaciones divergentes (figura 12).



RATIO DE APRENDIZAJE PEQUEÑO RATIO DE APRENDIZAJE GRANDE

Figura 12. Representación gráfica de la respuesta al algoritmo de aprendizaje, buscando minimizar la función del error. A la izquierda, con un η muy pequeña, el problema es el tiempo y número de iteraciones que demorará llegar a un mínimo global de error, encontrando los valores de W apropiados para esta solución. A la derecha, con un η muy grande, las oscilaciones pueden causar problemas de divergencia en la solución. Fuente: Université de Montréal, Département d'informatique et de recherche opérationnelle, Introduction aux reseaux de neurones http://www.iro.umontreal.ca/~bengioy/ift6266/neuron/neuron.html

En general, la función del error está dada por una suma de términos, cada uno de los cuales es calculado usando sólo uno de los patrones de la base de entrenamiento, de tal manera que,

$$E(W) = \sum_{m=1}^{p} E^m(W) \quad \ldots(11),$$

donde el término E^m es calculado usando solamente el patrón *m* (la observación número m de la base de datos). En este caso podemos adaptar el vector de peso usando sólo un patrón por vez:

$$w_{ij}^{(\tau+1)} = w_{ij}^{(\tau)} - \eta \frac{\partial E^m}{\partial w_{ij}} \quad \ldots(12)$$

Y esto es repetido muchas veces, actualizando los pesos mientras se va pasando de patrón en patrón, de forma secuencial. **En particular, esta técnica usa el sistema en aplicaciones adaptativas en tiempo real, en los que continuamente se recibe más datos.** Cada patrón de datos puede ser usado una vez y luego descartado, y si el valor de η es elegido apropiadamente, el

sistema podrá realizar un “seguimiento” a cualquier cambio pequeño en las características de los datos.

En otras palabras, hay dos maneras de realizar el aprendizaje con el gradiente descendiente: actualizando los pesos después de haber utilizado todos los datos de entrada (10) o actualizando los pesos después de usar cada patrón de entrada (12).

Para implementar el gradiente descendiente, necesitamos expresiones explícitas para las derivadas de la función del error con respecto a los pesos.

Así, para redes con funciones de activación no-lineales diferenciables, como la función logística sigmoidal (Tabla 1), se puede expresar las salidas de la red en la forma

$$y_i = \Phi(n_i) \quad \ldots(13),$$

Donde, por el criterio expuesto para la expresión (8),

$$n_i = \sum_{j=0}^{R} (w_{ij} \times x_j) \quad \ldots(14),$$

siendo $w_{i0}=b$, y $x_0=1$.

Consideramos primero la forma del gradiente descendiente basada-en-patrón (dado por el algoritmo iterativo en (12)), donde la función del error para el patrón *n* es:

$$E^m = \frac{1}{2}\{y(x^m;W) - t^m\}^2 \quad \ldots(15)$$

Entonces, derivando E^m con respecto a un w_{ij} y reemplazando la expresión con (15), (13), (14), las derivadas de la función del error con respecto a los pesos, para el patrón *m* están dadas por:

$$\frac{\partial E^m}{\partial w_{ij}} = \left(y^m - t^m\right)\Phi'\left(n_i^m\right)x_j^m \quad \ldots(16)$$

Con lo que el cambio en los pesos dada la presencia de un patrón particular se puede escribir de la siguiente manera:

$$\Delta w_{ij} = -\eta\delta_i^m x_j^m \quad \ldots(17),$$

Siendo $\delta_i^m = \Phi'\left(n_i^m\right)\left(y^m - t^m\right)$ …(18)

Para la función logística sigmoidal, la derivada de la función de activación puede ser expresada de esta forma:

$$\Phi'\left(n_i\right) = \Phi\left(n_i\right)\left(1 - \Phi\left(n_i\right)\right) \quad \ldots(19)$$

Para el gradiente descendiente basado en la función del error total (sumando para todos los patrones en la base de entrenamiento), dado por el algoritmo iterativo en (10), las derivadas se obtienen calculando de manera independiente las derivadas para cada patrón y sumándolas todas luego:

$$\frac{\partial E}{\partial w_{ij}} = \sum_{m=1}^{p} \frac{\partial E^m}{\partial w_{ij}} \quad \ldots(20)$$

Y entonces

$$\Delta w_{ij} = -\eta \sum_{m=1}^{p} \delta_i^m x_j^m \quad \ldots(21)$$

Generalmente se utiliza el gradiente descendiente cuando se considera que la relación entre las variables es no lineal – y por lo tanto la función de activación es no lineal – y, cuando al tener un problema de clasificación, no se puede obtener información previa que permita establecer a priori umbrales de clasificación directa para la(s) variable(s) de salida.

b) La Función Discriminante Logística

Esta técnica de aprendizaje intenta discriminar directamente hacia las *k* categorías de las salidas (C_k), basándose en el teorema de Bayes y utilizando probabilidades a priori para la pertenencia de los patrones de entrada a cada categoría.

La variable de salida está definida por la ecuación (6):

$$y = \Phi(Wx + b),$$

Donde $\Phi(n) \equiv \dfrac{1}{1+e^{-n}}$ …(22),

Bishop (1995) muestra que el uso de la función de activación logística sigmoidal permite a las salidas del discriminante ser interpretadas como probabilidades a posteriori. Esto implica que tal discriminante provee más que una simple decisión de clasificación, y es potencialmente un resultado muy poderoso.

Los discriminantes lineales con funciones de activación logísticas han sido ampliamente usados en la literatura estadística como *discriminación logística* (Anderson, 1982). Las funciones de activación sigmoidales también juegan un rol importante en las redes multi-capas, como veremos más adelante.

En resumen, si se desea discriminar directamente hacia una de *k* categorías y se cuenta con información a priori para modelar probabilidades, es aconsejable utilizar la función discriminante logística como función de activación.

c) El Perceptrón

McCulloch y Pitts (1943) introdujeron una forma de discriminante lineal como modelo matemático simple para el comportamiento de una única neurona en un sistema nervioso biológico. Toma la forma de (6) con una función de activación escalonada

$$\Phi(n) = \begin{cases} 0 \ \ cuando \ \ n < 0 \\ 1 \ \ cuando \ \ n \geq 0 \end{cases} \quad \ldots(23)$$

En este modelo las entradas x_i representan el nivel de actividad de otras neuronas que conectan a la neurona que está siendo modelada; los pesos w_i representan las fortalezas de las sinapsis entre las neuronas; y el sesgo *b* representa el umbral para que la neurona tome el valor de "1". Aunque este modelo tiene sus orígenes en la biología, es claro que puede estar igualmente inspirado dentro del marco del reconocimiento de patrones estadístico. Rosenblatt, en 1962 llamó a estas redes *perceptrones;* mientras que, al mismo tiempo, Widrow y su estudiante Hoff, en 1960, trabajaron en redes neuronales muy similares a las que llamaron *adalines* (ADAptative LINear Element o elemento lineal adaptativo) que se diferencian de los perceptrones en que su función de activación no es de umbral sino lineal (Hagan *et al.*, 1996)..

El perceptrón es un modelo muy simple que se puede utilizar en las aplicaciones que corresponden a las funciones de umbral (ver sección 2.3.1.).

2.6. Red multicapas (MLN – multilayer network)

Un caso más general es el de varias capas de neuronas:

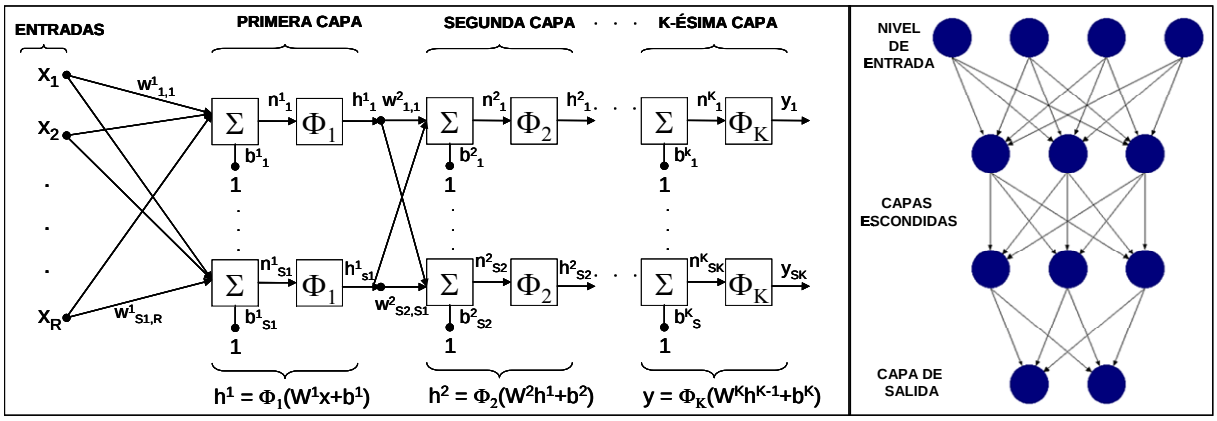


Figura 13. Representación de una red de múltiples capas. En el panel derecho, un diseño más simplificado en el que se observan los nodos, y sus conexiones con nodos de diferentes capas. Fuentes: Neural Network Toolbox for use with Matlab – Demuth y Beale, 1998 y AlanTuring.net - Copeland y Proudfoot, 2000

Podemos decir entonces, en general, que la entrada para un nodo es una suma ponderada de las salidas de los nodos que se conectan a él (desde una capa anterior, en sentido unidireccional, como lo indican las flechas de las figuras anteriores). Por ello, la entrada para un nodo de la red se expresa de la siguiente manera (Warner y Misra, 1996):

$$entrada\_al\_nodo_i = \sum_j w_{ij} \times salida\_de\_nodo\_anterior_j + b_i$$

En la figura anterior, las entradas a los nodos están representadas por n^k_s; para un nodo de la primera capa, las salidas del nodo anterior están denotadas por los x_r, para las de las capas escondidas, por los h^k_s y para las de la capa de salida, por los y_s. Los sesgos[18] son los b^k_s. *R* es el número de variables de entrada, los superíndices *k* indican el número de capa y los subíndices *s* indican el número de nodo en esa capa. Los pesos $w^k_{si,sj}$ se interpretan como la fuerza de la sinapsis que conecta la neurona *sj* de la capa *k-1* con la neurona *si* de la capa *k*.

Con ello la expresión anterior se puede reescribir así:

$$h_i = \sum_j w_{ij} \times n_j + b_i \quad \ldots(24)$$

El cálculo de cada nodo se realiza de la misma manera que para los anteriores casos: para cada nodo, de cada capa, se le aplica la función de activación a la entrada del nodo (definida en la ecuación 7).

Para una red de 2 capas, 1 escondida y 1 de salida (figura 13), la red neuronal podría ser expresada así:

[18] Técnica y computacionalmente sí es posible que una red neuronal no incluya sesgo. Sólo si los datos de entrada tienen valor inicial "0" cuando se empieza a correr la red los pesos nunca se modificarán (ver las ecuaciones que se muestran en este apartado; Will Dwinnell, http://will.dwinnell.com). En otros casos sí es factible que la red funcione. Recordemos que de todas maneras la red toma en cuenta el residual para la salida. Los sesgos son como interceptos dentro de la función de activación. Ayudan a cambiar más de prisa los valores de los pesos para llegar al óptimo (minimización de la función del error). Sarle (2002) dice que una unidad escondida o de salida va dibujando un hiperplano mientras va alimentándose de sus entradas. Los pesos determinan dónde se ubica este hiperplano en el espacio de entradas; pero sin un término de sesgo, este hiperplano se restringe a pasar por el origen del espacio definido por las entradas, cuando podría quizás ser más útil en otra ubicación. El autor muestra también algunos criterios para el uso del sesgo.

$$y_k = \Phi_2\left\{b_k + \sum_j w_{jk}\Phi_1\left(b_j + \sum_i w_{ij}x_i\right)\right\} \ldots(25),$$

donde y_k son las salidas; x_i, las entradas; w_{ij}, los pesos entre la neurona de entrada *i* y la neurona escondida *j*; w_{jk}, los pesos entre la neurona escondida *j* y la neurona de salida *k*; b_j y b_k son los sesgos para la neurona escondida *j* y para la neurona de salida *k*; y Φ_1 y Φ_2 son las funciones de activación para la capa escondida y la de salida, respectivamente. (Demuth y Beale, 1998; Warner y Misra, 1996)

Es necesario señalar que la elección de las funciones de activación para las unidades escondidas pueden ser diferentes a las de las unidades de salida, e incluso diferir en las funciones entre unidades de salida. La diferencia en las funciones se da más frecuentemente entre la de las capas escondidas y la de salida, puesto que las neuronas de ambos tipos de capa juegan diferentes roles.

Bishop (1995) realiza una extensa discusión sobre las funciones de activación de tipo umbral y sus limitaciones para redes con varias capas, generando, en muchos casos, regiones de decisión arbitrarias. Por otro lado, como veremos más adelante, cualquier límite de decisión dado puede ser aproximado arbitraria y cercanamente por una red de sólo dos capas con funciones de activación sigmoidal.

2.6.1. Unidades sigmoidales[19]

Muchas veces se utilizan funciones de activación logísticas sigmoidales, pues al obtenerse salidas en el rango <0,1>, dados por (22)

$$\Phi(n) \equiv \frac{1}{1+e^{-n}}$$

permite darle interpretaciones probabilísticas a estas salidas.

Aunque es frecuente usar la logística sigmoidal para las neuronas escondidas, hay algunas ventajas prácticas al usar una función de activación tangente hiperbólica (*tanh*) de la forma

$$\Phi(n) \equiv \tanh(n) \equiv \frac{e^n - e^{-n}}{e^n + e^{-n}} \quad \ldots(26)$$

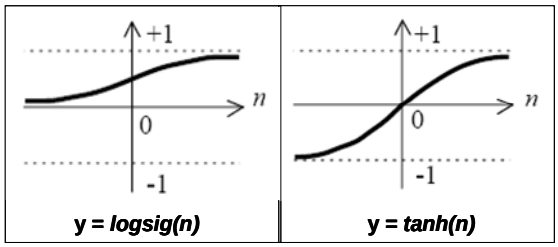


Figura 14. Representación gráfica de una función de activación logsig (izquierda) y una tanh (derecha). Fuente: Neural Network Toolbox for use with Matlab – Demuth y Beale, 1998

Si comparamos los paneles izquierdo y derecho de la figura 14, veremos que la función *tanh* difiere de la *logsig* sólo por una transformación lineal (Bishop, 1995). Por lo tanto, una red neuronal cuyas unidades escondidas usan la función de activación en (26) es equivalente a una con unidades escondidas usando (22), pero obteniendo diferentes valores para los pesos y los sesgos. Empíricamente, se ha encontrado de manera frecuente que las funciones de activación *tanh* hacen que los algoritmos de entrenamiento lleguen a una convergencia más rápida que cuando se utilizan funciones logísticas (Sarle, 2002 y 1999).

Bishop (1995) discute la capacidad de las redes de 2 capas con unidades escondidas sigmoidales para aproximar funciones continuas y discontinuas, y con ello, prueba que puede aproximar bastante

[19] Una unidad sigmoidal es aquella que proviene de una función de activación sigmoidal.

bien cualquier límite de decisión. Utiliza también la función *tanh* (que es sólo una transformación lineal de la *logsig*).

2.6.2. Error de retro-propagación

El error de retro-propagación o propagación hacia atrás (*backpropagation*) es un algoritmo de aprendizaje supervisado, con un concepto muy simple: si la red da una respuesta equivocada, los pesos son corregidos para que el error disminuya, y así las futuras respuestas de la red tengan más probabilidades de ser correctas (Lek y Guégan, 2000). El objetivo de este algoritmo, entonces, es ajustar los pesos haciendo que la función del error se minimice; por lo tanto, la función del error es la función objetivo. Este ajuste de pesos ocurre hacia atrás, capa por capa, empezando por la capa de salida y terminando con la capa de entrada (Chen y Ware, 1999). En su forma estándar, las adaptaciones de los pesos se producen por el algoritmo de gradiente descendiente, que explicaremos más adelante.

Este algoritmo iterativo, se utiliza para redes con arquitectura multicapa (MLN), y funciones de activación no-lineales diferenciables (Demuth y Beale, 1998), para que así las activaciones (entradas) para las unidades de salida se vuelvan funciones diferenciables para las variables de entrada, los pesos y sesgos. Consideraremos la función del error en (7), que es diferenciable para las salidas y por lo tanto para los pesos de la red (lo cual veremos a continuación).

Se ha demostrado que por medio de la propagación hacia atrás, una red neuronal anticipativa de 2 capas con un número suficiente de neuronas escondidas puede aproximar cualquier función continua a cualquier grado de precisión (Cybenko, 1989), lo cual hace de las MLN herramientas poderosas para el modelamiento.

Propagación hacia delante

La propagación hacia delante es el primer paso, en el que la red va recibiendo, transformando y comunicando la información recibida a través de los nodos, estableciendo conexiones entre las capas, cuyas fuerzas (pesos) varían; calculando los h_s^k, n_s^k e y_s.

Evaluación de las derivadas de la función del error

Recordando la estructura general de una MLN, para cada unidad

$$n_i = \sum_j w_{ij} \times h_j \quad \text{...(27)},$$

donde n_i es la entrada al nodo *i* y h_j es la activación del nodo *i* (la salida del nodo anterior). Recordemos que por el criterio para la expresión (8) no necesitamos explicitar en (27) la presencia del sesgo.

Luego, $h_i = \Phi(n_i)$ ...(28),

donde h_i es la activación del nodo de una siguiente capa.

Esto sucede para cada capa. En el caso de la primera capa, la activación se denota x_i; y en el caso de la última, la salida es y_i.

La función del error se puede considerar como la suma de las funciones del error para cada uno de los patrones de la base de entrenamiento:

$$E = \sum_{m=1}^{p} E^m \quad \text{...(29)},$$

donde E^m puede ser expresado como una función diferenciable de las variables de salida de la red:

$$E^m = E^m(y_1, \ldots, y_{sk}) \quad \ldots(30)$$

Y usando la regla de la cadena, tenemos:

$$\frac{\partial E^m}{\partial w_{ij}} = \delta_i^m \frac{\partial n_i^m}{\partial w_{ij}} \quad \ldots(31),$$

Siendo $\delta_i^m = \frac{\partial E^m}{\partial n_i^m}$ …(32)

Los δ son llamados comúnmente *señal del error*.

Utilizando la expresión en (27), se puede decir que

$$\frac{\partial n_i^m}{\partial w_{ij}} = h_j^m \quad \ldots(33)$$

Por lo tanto,

$$\frac{\partial E^m}{\partial w_{ij}} = \delta_i^m h_j^m \quad \ldots(34)$$

Que tiene la misma forma que la expresión (16) para las SLN.

La evaluación de los δ_k, para las unidades de salida, es directa, pues

$$\delta_k^m = \frac{\partial E^m}{\partial n_k^m} = \Phi'(n_k)\frac{\partial E^m}{\partial y_k} \quad \ldots(35)$$

Lo cual se puede calcular fácilmente.

Para las unidades escondidas, se pueden hallar los δ_k usando nuevamente la regla de la cadena:

$$\delta_k^m = \frac{\partial E^m}{\partial n_k^m} = \sum_s \frac{\partial E^m}{\partial n_s^m}\frac{\partial n_s^m}{\partial n_k^m} \quad \ldots(36),$$

que suma todas las unidades *s* a las que la unidad *k* dirige sus conexiones. Esta expresión muestra que las variaciones en n_k llegan a causar variaciones en el error sólo a través de las variaciones en las n_s.

Sustituyendo (32) en (36) y usando (27) y (28), se puede calcular la señal del error así:

$$\delta_k^m = \Phi'(n_k^m)\sum_s w_{sk}^m \delta_s^m \quad \ldots(37)$$

Aquí se ve claramente cómo el valor de δ para un nodo escondido puede ser obtenido al propagar hacia atrás los δ desde nodos en niveles mayores de la red, como se explicaba al principio.

Estando ahora en la capacidad de obtener todos los δ de la red, se pueden evaluar cada una de las derivadas del error y la derivada total también (la expresión (29)).

Esta es la propagación hacia atrás. Para que se convierta en un algoritmo de aprendizaje, se necesita escoger algún método basado en las derivadas del error para actualizar los pesos (como en el caso del gradiente descendiente para SLN).

2.6.3. Algoritmos para la optimización de los parámetros de la red

Para una mayor simplicidad, agruparemos a los parámetros peso y sesgo en un solo vector de pesos W-dimensional *w* con componentes $w_1 \ldots w_k$.

a) Gradiente Descendiente

Funciona de la misma manera que para las SLN; habiendo, igualmente, dos maneras de realizar el aprendizaje: actualizando los pesos después de haber utilizado todos los datos de entrada (aprendizaje por lote) o actualizando los pesos después de usar cada patrón de entrada (aprendizaje secuencial).

En la versión por lotes, el valor inicial es generalmente escogido aleatoriamente y se denota por $W^{(0)}$. Luego se actualiza por iteraciones el vector de pesos de manera que, en el paso τ, nos hayamos movido en una distancia corta en la dirección del mayor ratio de disminución del error –la dirección del gradiente negativo, evaluado en $W^{(\tau)}$:

$$\Delta W^{(\tau)} = -\eta \nabla E\big|_{W^{(\tau)}} \quad \ldots(38)$$

donde η el ratio de aprendizaje, ∇E es el gradiente de E en el espacio de pesos, que se reevalúa para cada paso τ.

En la versión secuencial, el gradiente de la función del error se evalúa para un patrón por vez, y los pesos son actualizados mediante

$$\Delta W^{(\tau)} = -\eta \nabla E^m\big|_{W^{(\tau)}} \quad \ldots(39)$$

siendo η el ratio de aprendizaje, y τ es el paso temporal en el algoritmo iterativo y E^m es la función del error para el patrón *m*.

Al igual que en SLN, la secuencia de vectores de pesos convergerá a un punto en el que *E* sea minimizado (figura 15). Y los criterios para la elección del valor para η son los mismos: si es demasiado pequeño, la reducción en el error será muy lenta y habrán pérdidas en tiempos computacionales; mientras que, si es demasiado grande, el algoritmo puede resultar en oscilaciones divergentes, lo cual podría resultar incluso en una ruptura del algoritmo (figura 12).

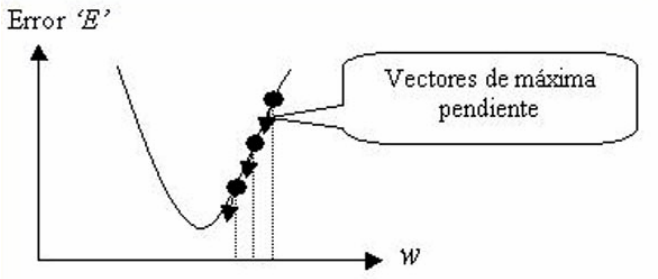


Figura 15. Gradiente: vector en el que ocurre la máxima pendiente de minimización del error. La dirección opuesta de este vector es aquella en la que se da el mayor ratio de disminución

de la función del error. Fuente: http://www.monografias.com/trabajos901/fundamentos-basicos-reconocimiento-voz/fundamentos-basicos-reconocimiento-voz.shtml

Más adelante encontraremos otros algoritmos alternativos para el aprendizaje. Si un algoritmo es muy complicado (como el de Levenberg-Marquardt) no se aconseja utilizar el aprendizaje secuencial. Si en el caso de estudio el entrenamiento secuencial fuera necesario, se debería recurrir a un algoritmo más simple (como el del gradiente). (Sarle, 2002).

Inconvenientes del Gradiente Descendiente

- Mínimo Local:

Generalmente, para la función del error, una función no lineal de los pesos, existen muchos mínimos que satisfacen

$$\nabla E = 0 \quad \text{...(40)}$$

El valor mínimo para el que la función del error es la más pequeña es llamado *mínimo global*, mientras que los otros mínimos son llamados *mínimo local* y son soluciones parciales para la red en respuesta de la data entrenada. Puede haber otros puntos que satisfacen la condición como el *máximo local* o el *punto de silla*.

Idealmente, se desea encontrar un mínimo global; no obstante, es muy difícil llegar a él y entonces lo que se espera, en la práctica, es encontrar un buen mínimo local (o un mínimo local suficientemente bueno) donde el error es pequeño.

Un problema del gradiente descendiente es que tiende a caer mucho en mínimos locales y quedarse atascado en ellos (figura 16).

Figura 16. Mínimo global y mínimos locales para la función del error en una red neuronal multi-capas. Fuente: ***http://www.monografias.com/trabajos901/fundamentos-basicos-reconocimiento-voz/fundamentos-basicos-reconocimiento-voz.shtml***

- Velocidad de Convergencia:

El algoritmo de aprendizaje utilizando gradiente descendiente puede ser muy lento, por lo cual se han desarrollado otras técnicas en su mayoría basadas en métodos numéricos que logran tiempo de convergencia menores que el gradiente descendiente.

2.6.3.1. Aprendizaje más rápido

b) Gradiente con momento

Una técnica utilizada para evitar caer atorado en un mínimo local (Lek y Guégan, 2000) y a la vez acelerar la convergencia (Klinke y Grassmann, 1998) es realizar una modificación al gradiente descendiente y añadir un término de "momento" que cambia el ancho del paso dependiendo del paso previo. Así,

$$\Delta W^{(\tau)} = -\eta\left[(1-\alpha)\nabla E\big|_{W^{(\tau)}} + \alpha\left(\Delta W^{(\tau+1)}\right)\right] \quad \text{...(41)}$$

Aunque la inclusión del momento lleva generalmente a una mejora en la eficiencia de la red en comparación con el gradiente descendiente; introduce por otro lado un segundo parámetro α cuyo valor debe ser elegido, además del parámetro del ratio de aprendizaje η.

A pesar de ello, sigue siendo más rápido que el gradiente descendiente, y continúa siendo una buena opción si necesitamos un algoritmo simple para el aprendizaje.

c) Búsqueda Lineal

En el procedimiento de búsqueda lineal, buscamos minimizar la función del error moviéndonos en alguna dirección de búsqueda en el espacio de pesos.

El método genera las iteraciones calculando los pesos en cada paso:
$w^{(\tau+1)} = w^{(\tau)} + \lambda^{(\tau)} d^{(\tau)}$ …(42)

donde
$d^{(\tau)}$ es una dirección de búsqueda en particular a través del espacio de pesos en el paso τ, y
$\lambda^{(\tau)}$ es el parámetro escogido para minimizar

$E(\lambda) = E\left(w^{(\tau)} + \lambda d^{(\tau)}\right)$ …(43), en el paso τ

El problema es que necesita mucho tiempo computacional para encontrar la dirección de búsqueda y evaluar los mínimos en aquella dirección. Algunos algoritmos de aprendizaje que resuelven parcialmente el problema son: *Sección de Búsqueda Dorada, Búsqueda de Brent, Búsqueda Híbrida Bisección-Cúbica, Búsqueda de Charalambous* y *Retro-Seguimiento*, cuyo desarrollo se puede encontrar en Hagan *et al.* (1996) o Demuth y Beale (1998).

Varios algoritmos de gradiente conjugado y métodos cuasi-Newton (que veremos a continuación) utilizan rutinas de búsqueda lineal para una convergencia más rápida. (Bishop, 1995 y Demuth y Beale, 1998).

Aproximación Cuadrática local

Las técnicas que presentaremos a continuación consideran una aproximación cuadrática local a la función del error; es decir, aproximando el error mediante expansiones de Taylor sobre algún punto $\hat{w}$ en el espacio de pesos:

$$E(w) = E(\hat{w}) + (w - \hat{w})^T b + \frac{1}{2}(w - \hat{w})^T H(w - \hat{w}) \quad \text{…(44)}$$

donde *b* es el gradiente de *E* evaluado en $\hat{w}$

$b \equiv \nabla E\big|_{\hat{w}}$ …(45)

y *H* es la matriz Hessiana definida por

$$(H)_{ij} \equiv \left.\frac{\partial E}{\partial w_i \partial w_j}\right|_{\hat{w}} \quad \text{…(46)}$$

Derivando *E* con respecto a *w* en la expresión (44), tenemos

$\nabla E = b + H(w - \hat{w})$ …(47) ,

que sería la aproximación local para el gradiente.

Para un punto *w**, que es mínimo local de la función del error, *b*=0, y la expresión (44) se reescribe como

$$E(w) = E(w*) + \frac{1}{2}(w - w*)^T H(w - w*) \quad \ldots(48)$$

y $$\nabla E = H(w - w*) \quad \ldots(49)$$

d) Método de Newton

Sabemos por (49) que el gradiente en cualquier punto *w* está dado por

$$g = H(w - w*) \quad \ldots(50)$$

donde $g \equiv \nabla E$.

Luego, despejando de la expresión, el vector de pesos *w** que corresponde al mínimo en la función del error, satisface la siguiente ecuación:

$$w* = w - H^{-1} g \quad \ldots(51)$$

Donde el vector $-H^{-1}g$ es conocido como la *dirección de Newton* o el *paso de Newton*, que apunta directamente hacia el mínimo de la función del error.

Como la aproximación cuadrática usada no es exacta, se debe aplicar (51) iterativamente, reevaluando a la matriz Hessiana en cada paso.

- Dificultades:

- La evaluación de la matriz Hessiana, al estar constituida por derivadas de segundo orden, es computacionalmente demandante; y lo es más aún la Hessiana invertida.
- El paso de Newton puede dirigirse a un máximo o punto de silla en lugar de a un mínimo, si la matriz Hessiana no es definida positiva.

Este último problema puede resolverse adicionándole a la matriz *H* un múltiplo de la matriz identidad, siendo la nueva matriz

$$H + \lambda I \quad \ldots(52)$$

y así la expresión (51) se transforma en

$$w* = w - (H + \lambda I)^{-1} g \quad \ldots(53)$$

Hay métodos llamados *métodos cuasi-Newton* que generan secuencias de matrices que aproximan a las inversas de la Hessiana, como el BFGS (Broyden-Fletcher-Goldfarb-Shanno) pero que tienen mayores requerimientos de memoria. Otros como el OSS reducen el requerimiento de memoria para almacenamiento pero pierden en precisión al aproximar la inversa de la matriz Hessiana al asumir, para efectos prácticos, que la matriz Hessiana del paso anterior es la matriz identidad. (Bishop, 1995; Hagan *et al.*, 1996; Demuth y Beale, 1998).

Los métodos cuasi-Newton necesitan almacenar y actualizar una matriz del tamaño $W \times W$, se aconseja utilizar estos métodos para redes con un número moderado de pesos, pues así la matriz no será tan grande y esto no repercutirá significativamente en los requerimientos de memoria para el aprendizaje. (Bishop, 1995 y Sarle, 2002)

e) Gradiente Conjugado

El método del gradiente conjugado es una técnica que presupone, primero, que el gradiente de la función puede ser calculado. Utiliza las direcciones conjugadas[20] en lugar del gradiente local para ir bajando en la función, algo por lo general produce convergencias más rápidas que las direcciones negativas del gradiente.
Entre los algoritmos de aprendizaje de este tipo se encuentran: *Actualización de Fletcher-Reeves, Actualización de Polak-Ribiére, Reinicio de Powell-Beale* y *Gradiente Conjugado Escalado* (ver Demuth y Beale, 1998).

El gradiente descendiente es el método más simple, mas su convergencia es lenta; y los métodos de Newton que vimos son rápidos pero requieren suficiente memoria para el cálculo de la matriz Hessiana. El gradiente conjugado se presenta como un método intermedio entre las 2 formas anteriores: no requiere el cálculo de segundas derivadas pero sí utiliza la aproximación cuadrática al error (con lo cual se gana en convergencia). Por ello, al requerir menos memoria, esta técnica es apropiada cuando se tiene un número muy grande de pesos (algunos miles, por ejemplo).

f) Levenberg-Marquardt

Aproxima la matriz Hessiana (de derivadas de segundo orden) mediante la matriz Jacobiana, una matriz con derivadas de primer orden:

$$H = J^T J \quad \text{...(54)}$$

Esto le otorga una mayor velocidad para los cálculos y por lo tanto, para la convergencia; y no tiene grandes requerimientos de memoria como otros métodos.

La matriz Jacobiana *J* está compuesta por elementos definidos de la siguiente manera:

$$\left(J\right)_{mi} \equiv \frac{\partial \varepsilon^m}{\partial w_i} \quad \text{...(55),}$$

donde $\varepsilon^m = y^m - t^m$ ...(56)

Además, desarrollando ∇E , podemos expresar el gradiente como

$$g = J^T \varepsilon \quad \text{...(57)}$$ (Hagan *et al.*, 1996)

Por lo tanto, (53) puede ser expresada como

$$w^* = w - \left(J^T J + \mu I\right)^{-1} J^T \varepsilon \quad \text{...(58)}$$

Aquí el parámetro μ gobierna el tamaño del paso. Lo que se hace un algoritmo de aprendizaje computacional en este caso es iniciar con un valor arbitrario para μ . Con ese valor de μ , si el paso fue exitoso (se redujo la función del error), el nuevo vector de pesos es retenido, y el valor de μ es reducido por algún factor multiplicador. En contraste, si el error se incrementa, μ es dilatado por otro factor multiplicador, el antiguo vector de pesos se mantiene y con el nuevo μ , un nuevo vector de pesos es calculado. Este procedimiento se repite iterativamente, hasta converger en un punto en el que *E* sea minimizado. De esta manera, la función objetivo (función del error) siempre se reduce en cada iteración del algoritmo.

---

20 Se dice que dos vectores no nulos u y v son conjugados (con respecto a una matriz) si $u^T A v = 0$; en nuestro caso es $d^T A d = 0$

Es importante mencionar que Levenberg-Marquardt es un algoritmo diseñado específicamente para minimizar funciones del error que son sumas de cuadrados de otras funciones no lineales (como la suma de cuadrados para el error y el error cuadrático medio). (Bishop, 1995 y Hagan *et al.*, 1996).

Este algoritmo converge en menos iteraciones que los métodos anteriores, pero requiere mayor cálculo por iteración que cualquier otro algoritmo, dado que necesita inversión de matrices. Sin embargo y a pesar de todos estos cálculos, Levenberg-Marquardt parece ser el algoritmo más rápido para el entrenamiento de la red para un número pequeño de pesos[21] (Hagan *et al.*, 1996).

En cuanto a los óptimos locales y globales, Sarle (2002) afirma que frecuentemente en la práctica (aunque no ofrece explicación teórica al respecto) este algoritmo encuentra un mejor óptimo (para una variedad de problemas) que los otros métodos.

En resumen y tomando como criterio principal la rapidez para la convergencia, se aconseja utilizar Levenberg-Marquardt cuando la red no es muy grande (número pequeño de pesos), los métodos cuasi-Newton cuando el número de pesos es moderado, y los de gradiente conjugado cuando la arquitectura de la red es enorme (ver Demuth y Beale, (1998) para un mayor desarrollo del tema).

2.6.4. Sobre-entrenamiento y sobre-ajuste de la red

Un problema frecuente en el proceso de entrenamiento de la red es el sobre-ajuste. En una red sobre-ajustada la red no aprende las características generales inherentes presentes en la base de entrenamiento, mas aprende perfectamente más y cada vez más detalles particulares de esta base de datos específica. Por lo tanto, la red pierde su capacidad de generalizar. Esto sucede cuando la red es sobre-entrenada. Es decir, entrenada más de la cuenta, ya sea en tiempo o en complejidad (si la red posee más nodos o más capas de los que necesita, se estará sobre-entrenando).

Desde el punto de vista estadístico, podemos considerarlo como un problema de intercambio sesgo-varianza, para el cual el error de generalización se descompone en la suma del sesgo al cuadrado más la varianza.

Pues

$$MSE = \frac{1}{p}\left[\sum_{j=1}^{p}(y^j - t^j)^2\right] \quad \text{...(59}^{22}\text{)}$$

donde p es el número de observaciones de la base de datos,
y^j es el patrón *j-ésimo* de salida de la red, y
t^j es el objetivo *j-ésimo*

entonces

$$E\{MSE\} = E\left\{\frac{1}{p}\sum_{j=1}^{p}(y^j - t^j)^2\right\} = \frac{1}{p}\sum_{j=1}^{p}E\{(y^j - t^j)^2\} \quad \text{...(60)}$$

y descomponiendo el término en el numerador, finalmente tenemos:

$$E\{(y^j - t^j)^2\} = Var\{ruido\} + sesgo^2 + Var\{y^j\} \quad \text{...(61) (Vijayakumar, notas de clase)}$$

---

[21] La restricción para el número de pesos en la red aquí es la misma que para el método de Newton. Dado que necesita suficiente memoria para los cálculos de la matriz inversa (56), solo con un número pequeño de pesos (hasta unos pocos cientos) la demanda de memoria no se convertirá en una desventaja importante del algoritmo.

22 La función del error aquí es diferente a la definida en la expresión (7). Como se señala en el anexo, se puede elegir una función para el error diferente, según sea el caso. Por otro lado, el MSE es igual a (7) si se le multiplica por n/2. Por lo tanto el desarrollo de las técnicas y algoritmos de aprendizaje no deben variar seriamente en sus expresiones matemáticas si se toma como función del error la de (59).

Nótese que la varianza del ruido no puede ser minimizada, pues es independiente de la red neuronal. Así, para minimizar la función del error, necesitamos minimizar tanto el sesgo como la varianza. Sin embargo, hacer esto no es trivial. Tener un modelo demasiado simple o inflexible, poseerá un gran sesgo y una varianza pequeña (panel izquierdo de la figura 17). El caso extremo sería rechazar toda la data de entrada y predecir de alguna manera la salida (quizás solo una constante), lo cual definitivamente minimizaría la varianza de las predicciones: estas serían siempre las mismas, y la varianza, cero; pero el sesgo de nuestro valor estimado (que representa qué tanto nos hemos distanciado del valor verdadero) sería tremendamente grande. Por otro lado, tener un modelo muy flexible en relación a una base de datos en particular asumirá una gran varianza (panel derecho de la figura 17). Un caso extremo aquí es el de aquel modelo que predice *y=t* para cada punto de la data. En una situación como esta el sesgo se desaparece totalmente, pero el término de varianza se hace igual a la varianza del ruido, lo que puede ser significativo. Como podemos ver, el sesgo y la varianza son cantidades complementarias; y la mejor generalización se obtiene cuando se logra un equilibrio entre la suavización que se necesita hacer para que el modelo tenga varianza pequeña y la fidelidad a los datos para que el sesgo sea pequeño. A esto se le conoce como intercambio sesgo-varianza o dilema sesgo-varianza (para una discusión más amplia sobre el tema, ver German *et al.*, 1992).

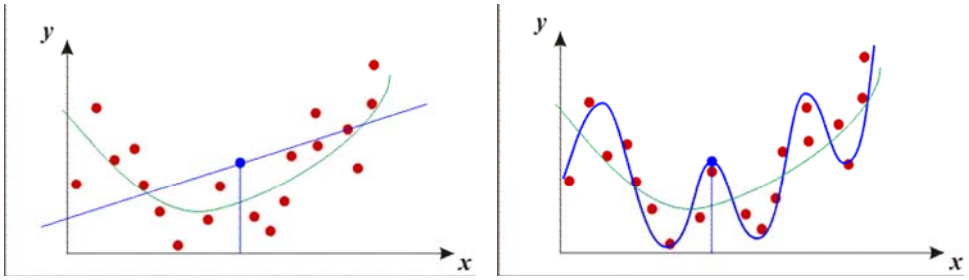


Figura 17. A la izquierda, una función demasiado simple para aproximar a la función verdadera tiene un gran sesgo aunque varianza pequeña (sub-ajuste). A la derecha, una función que pasa por cada uno de los puntos de la data no tiene sesgo, mas sí una alta varianza (sobre-ajuste). Fuente: http://www.aiaccess.net/English/Glossaries/GlosMod/e_gm_bias_variance.htm

En general, resolver el dilema, encontrar un intercambio sesgo-varianza óptimo es difícil, pero algunas técnicas como *parada temprana*, *regularización bayesiana* y *validación cruzada* pueden ayudar en este sentido.

a) Parada Temprana (Early-Stopping)

Es un método de regularización que mejora la generalización (figura 19). El primer paso en esta técnica es dividir la base de datos en tres partes.

1) La de **entrenamiento**, con la cual se realiza el proceso de aprendizaje estándar. Se calcula el gradiente del error y se actualizan los pesos y sesgos, utilizando alguno de los algoritmos de la sección anterior.

2) La de **validación**, es para validar la generalización del entrenamiento que se está haciendo con la primera parte de los datos: los patrones de entrada de esta parte de la base se ingresan a la red y, utilizando los valores de los pesos y sesgos actualizados para la data de entrenamiento, se hallan las salidas de la red; con lo cual se calcula la función del error. El error en la validación es monitoreado durante el proceso de entrenamiento pues, aunque es normal que este decrezca durante la primera parte del entrenamiento, cuando la red empieza a sobre-ajustarse a la data, el error para la validación comienza a elevarse (figura 18). En el momento en que el error de validación aumenta para un número específico de iteraciones (establecidas por el/la diseñadora de la red), el entrenamiento para y nos quedamos con los pesos y sesgos para los que el error de validación fue mínimo.

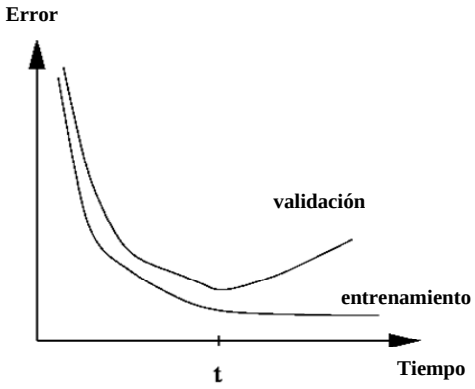


Figura 18. Mientras va pasando el tiempo de aprendizaje, llega un punto en el que la red empieza a aprender de características muy específicas de la data de entrenamiento, y se pierde generalización, lo cual se puede observar cuando en cierto momento el error en la data de validación empieza a crecer. Fuente: http://www.willamette.edu/~gorr/classes/cs449/figs/earlystop.gif

3) La base del **test** no es usada durante el entrenamiento, pero sirve para probar al final con esta data, al haber fijado ya los pesos y sesgos del modelo, con qué exactitud predice la red entrenada. Esto es sumamente útil para la comparación de diferentes modelos.

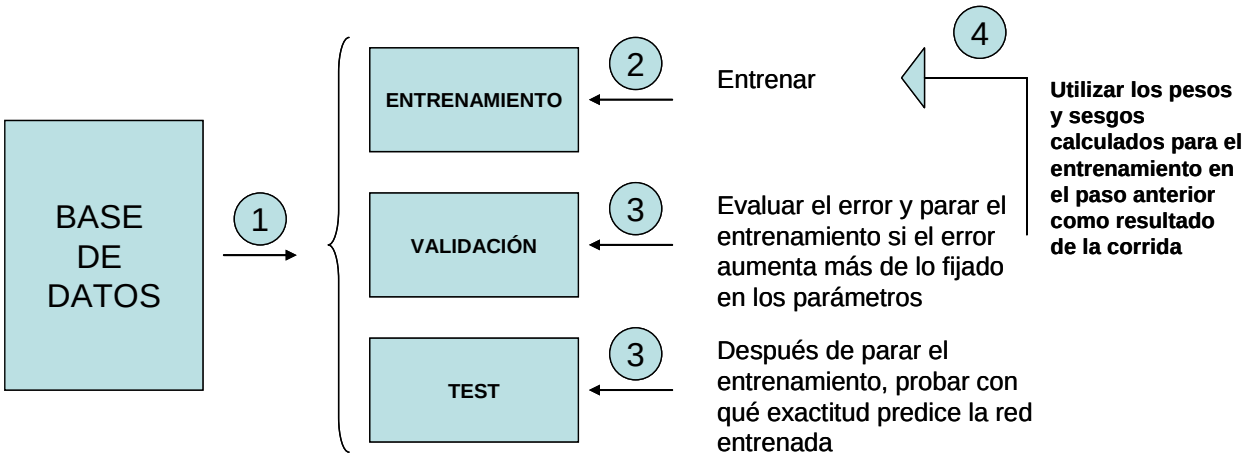


Figura 19. Esquema del método de parada temprana.

Es importante señalar que aquí no se calcula un error de generalización, sino se toma el de validación como aproximación al primero. Por ello la decisión de parar no se da apenas el error de validación aumenta, sino cuando el aumento se da en un número fijado de iteraciones consecutivas. La idea detrás de este criterio de parada es que, al incrementarse el error de validación no sólo una vez sino durante varias iteraciones consecutivas, se asume que esta es señal inequívoca del comienzo de un sobre-ajuste cada vez mayor; independientemente de cuán grandes hayan sido estos incrementos.

b) Regularización Bayesiana (Bayesian Regularization)

La aproximación por regularización bayesiana se realiza modificando la función objetivo:

$$E = \frac{1}{p}\sum_{m=1}^{p}\left(\varepsilon^m\right)^2 \quad \text{...(62),} \quad \text{(recordando que } \varepsilon^m = y^m - t^m\text{)}$$

por

$$E^* = \beta E + \alpha E_W \quad \text{...(63)}$$

para mejorar la capacidad de generalización del modelo; donde la función objetivo es expandida añadiéndole un término E_W, que es la suma de cuadrados de los pesos de la red, y α y β son parámetros que deben ser optimizados durante el aprendizaje.

En este método se asume que los pesos y sesgos son variables aleatorias que siguen distribuciones normales y los parámetros α y β están relacionados a las varianzas desconocidas asociadas a estas distribuciones (ver MacKay, 1995 para una mayor profundización sobre el tema).

Entonces, se trata aquí de minimizar esta nueva función del error (63), pudiendo utilizar cualquiera de los algoritmos de aprendizaje presentados anteriormente u otros.

Las ventajas de la regularización bayesiana son:

- Al tener la distribución de probabilidad de la salida, se puede medir la incertidumbre de nuestras predicciones.
- Otorga criterios para la arquitectura de la red (ejm.: número de capas escondidas y de unidades escondidas para cada capa) (Ripley, 1995 y Medeiros, 2001 presentan algunos métodos bayesianos para elección de parámetros, sugiriendo algoritmos diferentes para el entrenamiento de las redes).
- Permite adaptar a la red ciertas características de los datos (ejm., la suavización de la función, el grado en que las diferentes entradas son relevantes).

Estas dependen de qué tan acertados estemos al escoger las distribuciones de probabilidad a priori, usualmente definidas mediante una aproximación jerárquica.

Si no poseemos información a priori para definir las distribuciones, se pueden realizar pruebas de sensibilidad para redes que adopten diferentes distribuciones de los pesos. Otra alternativa es usar la Parada Temprana, una técnica de generalización más rápida y que demanda del usuario dos decisiones importantes: la proporción de tamaños para las bases de entrenamiento, validación y test, y el número de iteraciones para las que el error de la validación puede aumentar antes de que se pare de entrenar la red (porque muestra una tendencia creciente). (Sarle, 2002). Doang y Liong (2004) discuten la capacidad de generalización de ambos métodos obteniendo mejores resultados con la regularización bayesiana, pero señalando que cuando la base de datos es grande y relativamente limpia, el enfoque de generalización no juega un rol crucial en el entrenamiento de la red.

Por otro lado, Chan *et al.* (2003) propone realizar primero un entrenamiento con el criterio de Parada Temprana y sobre esa base pre-entrenada, entrenar con regularización bayesiana. Lo llaman **Método de Regularización Bayesiana de Parada Temprana** o Early-Stopping Bayesian Regularization Method (EBR). Ellos obtienen mejores resultados con este método que utilizando la regularización bayesiana estándar. No obstante, este criterio híbrido vuelve más complejo el entrenamiento y no presenta soluciones a los problemas de elección de parámetros de la Parada Temprana ni de distribuciones para la Regularización Bayesiana.

Independientemente del método escogido para la generalización de la red, la multicapas es el tipo de red más utilizada. Podemos encontrar aplicaciones en ecología marina en Chen y Ware (1999), Dreyfus-León (1999), Drumm *et al.* (1999), Scardi (2000), Baruah *et al.* (2001), Olden y Jackson (2001), Gaertner y Dreyfus-León (2004), Megrey *et al.* (2005) y Goethals *et al.* (2007), por citar algunos.

2.7. Red de base radial

La red de función de base radial (RBF – siglas en inglés) es una red anticipativa de dos capas que utiliza las funciones de base radial como funciones de activación (Figura 20).

Las entradas x_i se conectan con la capa escondida a través estas funciones de base radial mediante una medida de distancia. Así tenemos por ejemplo a la función gaussiana

$$h_j = \exp\left\{\frac{-\left\|x_i - U_j\right\|}{2\sigma_j^2}\right\} \quad \ldots(64)$$

donde $\|x_i - U_j\|$ es la distancia (medida por alguna norma) entre los puntos x_i y U_j en el espacio, siendo U_j el centro de la función en el nodo *j*. σ_j es una medida de dispersión de la función en el nodo *j*.

Las salidas de la red están dadas por

$$y_i = \sum_{j=1}^{L} h_j w_{ij} \quad ...(65)$$

donde los w_{ij} son los pesos para la capa de salida y *L* es el número de nodos en la capa escondida.

El aprendizaje aquí es del tipo híbrido, pues la capa de salida es entrenada por un método supervisado y la capa escondida, por uno no supervisado.

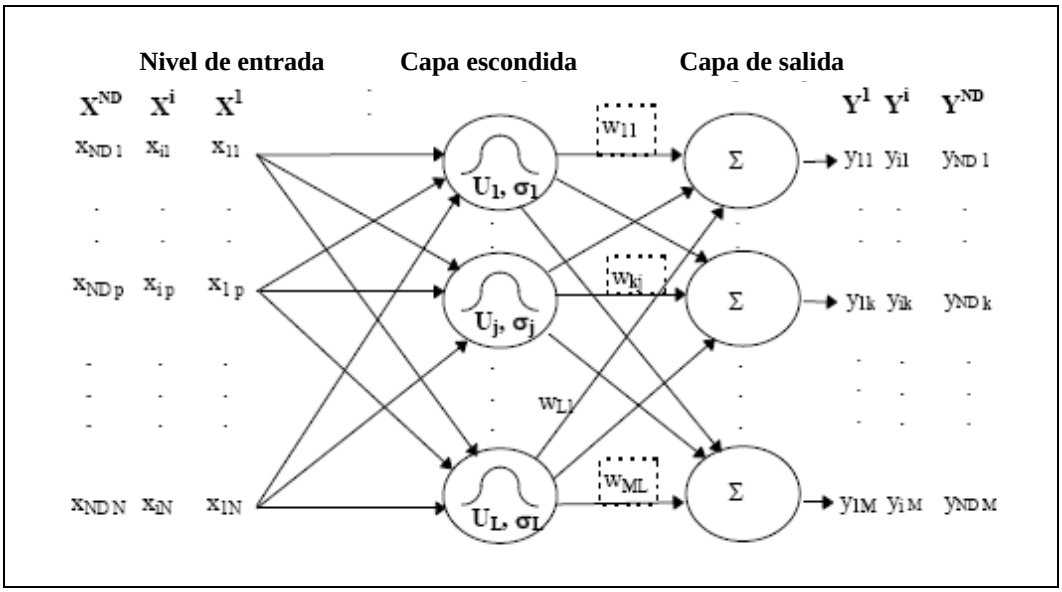


Figura 20. Diagrama Esquemático de una Red de Función Base Radial. El primer subíndice de las variables x e y representan al número de observación o patrón. Fuente: Jayawardena et al., 1998.

Las redes de función de base radial, por lo general, tienden a aprender mucho más rápido que las MLN. La desventaja es que la primera necesita muchas más neuronas que las últimas. (Demuth y Beale, 1998).

Otra desventaja es que las RBF consideran a cada variable de entrada igualmente importante como entrada a los nodos escondidos. Pero esto puede solucionarse penalizando los parámetros del proceso de estimación (ver Lendasse *et al.*, 2003 y su aplicación al precio de opciones).

Por otro lado, Venkatesan y Anitha (2006) manifiestan que las redes de base radial son sensibles a la dimensionalidad de la base de datos y presenta grandes dificultades si esta es muy grande. Por ello, sólo cuando la base de datos para entrenamiento es limitada, es aconsejable usar redes de función de base radial en lugar de las MLN (Goethals *et al.*, 2007).

2.8. Redes competitivas

Son redes auto-organizativas con arquitecturas recurrentes, como se aprecia en la figura 21. Tiene una única capa: la capa de salida. La entrada al nodo se expresa de la siguiente manera:

$$n_i = \|ndist(x_i, w_{i.})\| + b_i \quad ...(66)$$

Donde ndist es la distancia negativa entre el vector de entrada (x_i) y el que representa a los pesos de las entradas (w_i). (Demuth y Beale, 1998)

Estas redes aprenden haciendo "competir" a los nodos entre ellos:
El nodo que tenga su vector de pesos más cerca al vector de entradas es declarado **ganador** – sucederá cuando la distancia negativa entre la entrada y los pesos será la menos negativa , su salida será considerada "1" y únicamente los pesos de este nodo ganador serán ajustados mediante algún algoritmo de entrenamiento. Mientras que, para el resto de nodos, su salida será "0" y sus vectores de pesos mantendrán los valores iniciales. Este proceso es luego repetido para cada patrón de entradas, una y otra vez, para un número (usualmente grande) de ciclos.

Aquí, la función de activación es la función competitiva, cuya relación entrada/salida es la siguiente:

$$y = \begin{cases} 1, & si \;\; n = \max(n) \\ 0 & o.c. \end{cases} \quad \ldots(67)$$

(Hagan *et al.*, 1996)

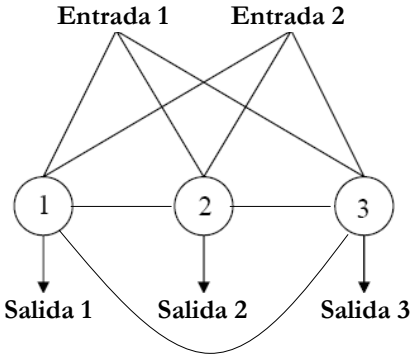


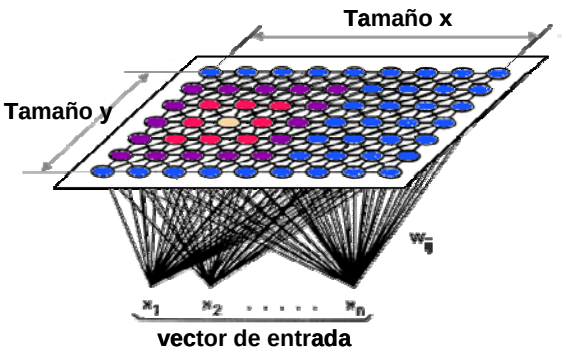
Figura 21. Representación de una red competitiva de 3 nodos.

Este tipo de red aprende de manera no supervisada y se utiliza para realizar conglomerados y reducir la dimensionalidad de la data, aunque es más usada la red de Kohonen –que presentaremos a continuación. Como ejemplo de aplicación, podemos encontrar el trabajo de Puntonet *et al.* (2001) en el que clasifican (separan) fuentes de las que provienen señales de comportamiento estocástico utilizando una red estocástica competitiva.

2.9. Mapas auto-organizativos de Kohonen (SOM)

Consiste, al igual que las redes competitivas, en 2 capas, una de entrada y una de salida. La capa de salida posee un mapeo auto-organizativo, que consiste en una red bi-dimensional de neuronas ordenadas en un cuadrado generalmente, u otra forma geométrica, haciendo una grilla. Cada neurona está conectada a su vecino más próximo en la grilla. (Figura 22)

Figura 22. Mapas Auto-Organizativos de Kohonen. Fuente: www.sis.pitt.edu/~ssyn/som/som.html. School of Information Sciences. University of Pittsburgh



El funcionamiento del aprendizaje no-supervisado en estos mapas es similar al de las redes competitivas. Los nodos siguen compitiendo entre sí por poseer la menor distancia entre los pesos y

las entradas, y el ganador tiene el "derecho" de ajustar sus pesos. Pero esta vez, no es solo el ganador, sino su vecindad, la que obtiene ese derecho. El tamaño de la vecindad del ganador va decreciendo linealmente después de cada nueva "competencia", hasta que incluya únicamente al ganador. La cantidad en la que a los nodos de la vecindad se les permite ajustar sus pesos también se va reduciendo linealmente durante el período de aprendizaje. (Lek y Guégan, 2000)

Cuando la red ha sido entrenada totalmente, los nodos con menores normas (66) se van acercando a las entradas que ocasionan estos valores, lo cual hace que se produzca un "mapa ordenado" de entradas, un mapa semántico[23] donde los patrones similares son mapeados juntos (conglomerados) y los disímiles, disjuntos; reduciendo así, la dimensionalidad de la data.

Kohonen escribió acerca de su creación[24]: "El mapeo auto-organizativo es una herramienta efectiva para la visualización de data de altas dimensiones. Convierte las relaciones estadísticas no-lineales complejas entre los ítems de data de altas dimensiones en relaciones geométricas simples de bajas dimensiones. Como comprime la información preservando a la vez las relaciones topológicas y métricas más importantes de los patrones de datos, también puede ser pensado para producir algún tipo de abstracción". Park *et al.* (2003) realiza una aplicación de SOM y MLN en ecología marina que puede ayudar a visualizar mejor la utilidad de las redes de Kohonen para el modelamiento no-lineal.

2.10. Redes simples recurrentes de Elman (SRN)

La Red de Elman tiene un conjunto extra de unidades de entrada, llamadas *unidades de contexto***.** Estas neuronas no reciben entradas desde fuera de la red, mas sí de la capa escondida de la red en una relación uno-a-uno (figura 23). Básicamente, las unidades de contexto contienen una copia del estado interno de la red en el paso previo. Luego, las unidades de contexto alimentan a la capa escondida tal y como lo hacen las otras unidades de entrada, y así la red es capaz de establecer una función que no sólo depende de las entradas actuales, sino además del estado interno de la red (que es determinado por las entradas previas o del tiempo previo).

Esta conexión recurrente permite a la red tanto detectar como generar patrones con variación espacio-temporal, lo cual la hace muy útil en áreas como las de procesamiento y predicción de señales donde el tiempo juega un rol predominante (Demuth y Beale, 1998).

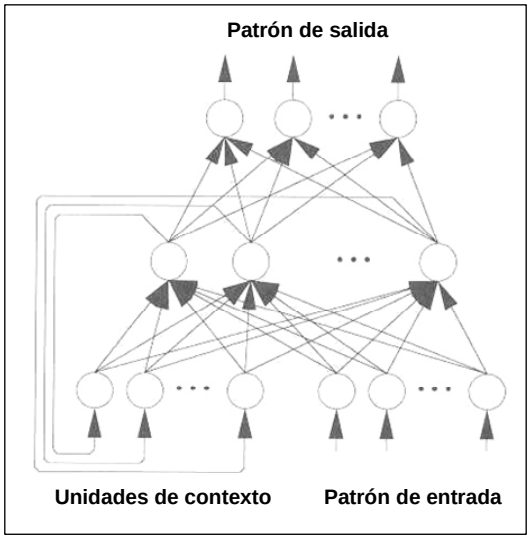


Figura 23. Red de Elman. Nótese que difiere de las redes de 2 capas convencionales, ya que la primera capa tiene una conexión recurrente. El rezago en esta conexión guarda valores del paso en el tiempo previo, los cuales pueden ser usados en el paso del tiempo actual. (Demuth y Beale, 1998). Fuente: www.math.tau.ac.il/~nin/Courses/NC05/misc.ppt School of Mathematical Sciences. Israel Mathematical Union

Además, las redes recurrentes de Elman se han mostrado eficientes en el manejo de fenómenos físicos con inercia temporal (ver Ciarlini *et al.*, 2005 con un caso de estudio de estimación de parámetros ambientales de un teatro romano).

[23] El " mapa semántico " es una representación visual de un concepto particular; es decir, es una estructuración de la información en categorías, representada gráficamente. (http://biblioteca-digital.ucentral.cl/documentos/libros/lintegrado2/mapa%20semantico.html)
[24] Tomado de la página tutorial de SOM de Sue Yeon Syn, de la Universidad de Pittsburg. http://www.sis.pitt.edu/~ssyn/som/som.html

Sin embargo, para aproximar funciones complejas, necesitarán muchas neuronas para ajustar los patrones de datos (Demuth y Beale, 1998).

2.11. Redes de Hopfield

Una Red Hopfield (figura 24) es una red neuronal recurrente que tiene patrones de conexión sináptica en los que subyace una función Lyapunov para las dinámicas de las actividades. Empezando en cualquier estado inicial, el estado del sistema evoluciona a un estado final que es mínimo (local) de la función Lyapunov.

Hay 2 formas populares del modelo:

Redes binarias con tiempo discreto, actualizadas una a la vez:

$$V_j(t+1) = \begin{cases} 1, & si \quad \Sigma_k \mathrm{T}_{jk} V_k(t) + I_j > 0 \\ 0, & otro \quad caso \end{cases} \quad ...(68)$$

Y neuronas clasificadas con tiempo continuo:

$$\frac{\partial x_j}{\partial t} = -{x_j}/{\tau} + \Sigma_k T_{jk} g(x_k) + I_j \quad ...(69)$$

donde V_j denota la actividad para la j-ésima neurona, x_j es la media del potencial interno de la neurona, I_j es la entrada directa a la neurona, T_{jk} es la fuerza de entrada sináptica (peso) desde la neurona k hacia la neurona j y g es una función monótona que convierte el potencial interno en una salida de la neurona (función de activación para la salida).

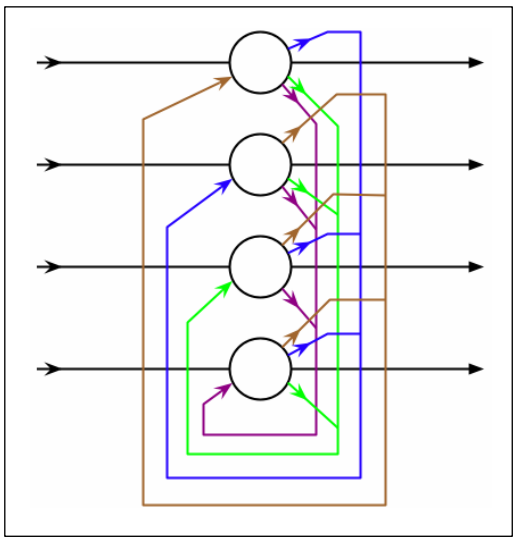


Figura 24. Red de Hopfield con 4 nodos. En este modelo, las neuronas están conectadas entre sí. Aquí se presenta primero el patrón o la observación de entrada para luego calcular la salida de cada neurona, y así hasta llegar a un resultado estable. Incluso existe la posibilidad de que el patrón de salida nunca llegue a un estado estable. En ese caso, se concluye que la red no reconoció el patrón. Fuente: Wikipedia

Puede ser un recurso alternativo cuando, por restricciones de costos computacionales, no busco necesariamente "la mejor" solución, sino una "buena" solución que esté "muy cerca" de ser la mejor (solución "estable": donde la probabilidad calculada de que sea la mejor solución es muy alta). (De Campos, 2006)

No obstante, existe el riesgo de diseñar puntos estables espúreos que lleven a respuestas incorrectas. Si es posible, es mejor optar por técnicas de aprendizaje supervisado como las que utilizan la retro-propagación. (Demuth y Beale, 1998).

2.12. Síntesis

Como hemos visto, aunque sabemos que en general las redes neuronales son útiles para problemas de reconocimiento de patrones, aproximación de funciones, clasificación y agrupación, cada clase de red tiene características propias que la hace más o menos adecuada que otra para resolver un tipo de problema en específico. La tabla 2 presenta un resumen de las redes presentadas anteriormente y sus aplicaciones.

Presentamos también algunos de los algoritmos usados para los tipos de redes neuronales artificiales descritas anteriormente (Tabla 3).

3. Relación con otras técnicas (de la estadística clásica)

3.1. Modelos lineales generalizados (GLM) (Dobson, 2002)

Un GLM posee 3 componentes:

- Un componente aleatorio de la variable respuesta *y*, con media μ y varianza σ^2.
- Un componente sistemático que relaciona la variable independiente x_i a un predictor lineal:

$$\eta = \sum_{i=0}^{N} \beta_i x_i \quad \ldots(70)$$

- Una función cadena o link $g(.)$ que relaciona la media con el predictor lineal

$$\eta = g(\mu) \quad \ldots(71)$$

Warner y Misra (1996) comparan el GLM con las redes neuronales. Señalan que, en general, se puede aproximar a un GLM con un modelo equivalente a una red SLN (de una sola capa), que fue descrita en la sección 2.4. Los coeficientes β_i corresponden a los pesos w_{1i}, las variables independientes son como las variables de entrada (hay *N-1* variables independientes), y la dependiente es la variable de salida. En este caso, la función de activación sería una igual a la inversa de la función cadena ($\Phi = g^{-1}$) y la función del error, igual a la devianza (que es la función que el GLM tiene como objetivo reducir); y se podría utilizar la retro-propagación para encontrar los β_i adecuados.

Es más, si el componente aleatorio tiene una función de distribución normal con media cero y varianza σ^2, y definimos a la función cadena como la función identidad, el modelo lineal generalizado se reduce a un **modelo de regresión lineal múltiple**:

$$y^m = \sum_{i=0}^{N} \beta_i x_i + \varepsilon^m \quad \ldots(72),$$

donde

$$\varepsilon^{(m)} \sim N(0, \sigma^2)$$

el cual es equivalente a una SLN con función de activación igual a la función identidad.

RED	CLASIFICACIÓN DE LA RED	APRENDIZAJE	PARA QUÉ SIRVE	CARACTERÍSTICAS	ALGUNOS CASOS DE ESTUDIO PUBLICADOS
Red de una capa (SLN)	Anticipativa	Supervisado	Para realizar estimaciones en las variables de salida.	Posee una arquitectura más simple que las redes alternativas. Podemos utilizarla cuando creemos que la relación entre las variables de entrada y de salida es directa (no hay necesidad de una capa escondida). Si en realidad no hay una relación directa entre las variables de entrada y salida, la eficiencia y la capación de generalización de la red será mala.	
Red Multicapas (MLN)	Anticipativa	Supervisado	Para realizar estimaciones en las variables de salida.	Es la más usada (hay mucha literatura en cuanto a sus aplicaciones) Es una extensión de la SLN. El número de capas escondidas y la(s) función(es) de activación son a criterio del diseñador. Presenta mejores desempeños que otras redes cuando la base de datos de entrenamiento es muy grande.	Chen y Ware (1999), Dreyfus-León (1999), Drumm et al. (1999), Scardi (2000), Baruah et al. (2001), Olden y Jackson (2001), Gaertner y Dreyfus-León (2004), Megrey et al. (2005) y Goethals et al. (2007)
Red de Función de Base Radial (RBFN)	Anticipativa	No supervisado y supervisado	Para realizar estimaciones en las variables de salida.	Su arquitectura presenta una sola capa escondida y la función de activación es siempre del tipo base radial. Tiene una mayor rapidez de convergencia que la MLN, por lo general. Se aconseja utilizarla cuando la base de datos para el entrenamiento es limitada.	Lendasse et al. (2003), Venkatesan y Anitha (2006), Sahin (1997), Jayawardena (1997)
Red Simple Recurrente de Elman (SRN)	Recurrente / de retroalimentación	Supervisado	Para realizar estimaciones en las variables de salida.	Utiliza una capa escondida, que retroalimenta al primer nivel. Puede detectar y/o generar patrones tomando en cuenta la variación espacio-temporal. Es recomendable si se necesita usar la red en tiempo real. Sus resultados no son tan confiables porque para el entrenamiento aproxima el error del gradiente (no lo calcula). Necesita más unidades escondidas que otros tipos de redes para optener buenos resultados en el aprendizaje.	Ciarlini et al. (2005), Rodriguez et al. (1999)
Red Competitiva	Recurrente / de retroalimentación	No supervisado	Para realizar conglomerados y reducir la dimensionalidad de la data	Es preferible utilizarla cuando se desea agrupar los datos sólo tomando en cuenta la distribución de las variables de entrada.	Puntonet et al. (2001)
Mapas Auto-Organizativos de Kohonen (SOM)	Recurrente / de retroalimentación	No supervisado	Para realizar conglomerados y reducir la dimensionalidad de la data	Se aconseja usarla cuando, además de la distribución de las variables, es importante la topología de la matriz de datos para el agrupamiento (clusters).	Park et al. (2003), Guerrero (2001),
Red de Hopfield	Recurrente / de retroalimentación	Supervisado	Para encontrar un estado estable, un conjunto de puntos de equilibrio.	Se presenta como recurso alternativo cuando, por restricciones de costos computacionales, no busco necesariamente "la mejor" solución, sino una "buena" solución que esté "muy cerca" de ser la mejor (solución "estable": donde la probabilidad calculada de que sea la mejor solución es muy alta). Existe el riesgo de encontrar puntos estables espúreos que lleven a respuestas incorrectas	He y Sýkora (2006), Ritthipravat y Nakayama - 2002

Tabla 2. Diversas clases de redes neuronales y sus características.

RED	TÉCNICA/ALGORITMO DE APRENDIZAJE
Red de una capa (SLN)	Regla de aprendizaje de perceptrón (Hagan et al., 1996; y Demuth y Beale, 1998) Gradiente descendiente (Bishop, 1995) Algoritmo del promedio mínimo al cuadrado (LMS) o algoritmo de aprendizaje Widroff-Hoff (Hagan et al., 1996; y Demuth y Beale, 1998).
Red Multicapas (MLN)	Gradiente descendiente (Bishop, 1995; Jayawardena et al., 1997; Chen y Ware, 1999; y Olden y Jackson, 2001) Gradiente descendiente con momento (Bishop, 1995; Hagan et al., 1996; Demuth y Beale, 1998; y Lek y Guégan, 2000) Método de Newton (Bishop, 1995; y Hagan et al., 1996) Métodos cuasi-Newton (Bishop, 1995; y Demuth y Beale, 1998) Gradiente conjugado (Bishop, 1995; Hagan et al., 1996; Demuth y Beale, 1998; Chan et al., 2003; Doan y Liong, 2004; y Venkatesan y Anitha, 2006) Levenberg-Marquardt (Bishop, 1995; Hagan et al., 1996; Foresee y Hagan, 1997; y Demuth y Beale, 1998)
Red de Función de Base Radial (RBFN)	Métodos de aprendizaje por cuantización vectorial (Vector Quantization) (Lendasse et al., 2003). Es un método híbrido que comprende aprendizaje no-supervisado -como los algoritmos de conglomerados por k-medias -y aprendizaje supervisado -como algoritmos similares al de retro-propagación (ver Jayawardena et al., 1997 y Venkatesan y Anitha, 2006).
Red Simple Recurrente de Elman (SRN)	Algoritmos basados en la retro-propagación, como el gradiente descendiente con momento secuencial (Demuth y Beale, 1998) y el de retro-propagación-a-través-del-tiempo (BPTT - Rodriguez et al., 1999).
Red Competitiva	Regla de aprendizaje de Kohonen (Demuth y Beale, 1998) Regla de aprendizaje por el sesgo (Demuth y Beale, 1998)
Mapas Auto-Organizativos de Kohonen (SOM)	Regla de aprendizaje de Kohonen (Demuth y Beale, 1998; y Park et al., 2003)
Red de Hopfield	Aprendizaje supervisado hebbiano (Hagan et al., 1996) Algoritmo de escalamiento-de-montaña (Hill climbing algorithm - He y Sýkora, 2006)

Tabla 3. Técnicas y algoritmos de aprendizaje para las redes neuronales artificiales.

Las diferencias más marcadas entre ambas propuestas es que en los GLM la forma funcional (función cadena) es previamente impuesta a la data. Cuando no se tiene idea de la relación funcional entre las variables dependientes e independientes, se puede utilizar un modelo GAM, un método de suavización multidimensional que permite tener ideas previas sobre una posible relación funcional. Otra alternativa es emplear las redes neuronales, pues aunque al empezar a entrenar una red neuronal se debe especificar a priori la función o funciones de activación, existen redes –como la anticipativa de 2 capas con funciones de activación sigmoidales –que pueden aproximar con precisión cualquier función continua (Bishop, 1995), con lo que no se necesita conocer la o las expresiones analíticas antes del aprendizaje.

3.2. Modelos aditivos generalizados (GAM) (Hastie y Tibshirani, 1984; 1995a y 1995b)

Son métodos no paramétricos. A diferencia de los GLM, los GAM sí permiten establecer relaciones no-lineales entre predictores y la variable predicha.

Aquí

$$\eta = \sum_{i=0}^{N} \Phi_i x_i \quad \ldots(73),$$

donde los Φ_i son funciones no-lineales que serán estimadas a partir de la data mediante un suavizador bivariado (ver Acuña, 2004 para una exposición detallada de la suavización en regresión no paramétrica).

La forma del modelo es, entonces:

$$y = g\left(\sum_{i=1}^{N} \Phi_i \left(x_i \right) + w_0 \right) \quad \ldots(74)$$

donde $g(.)$ es la función que finalmente relaciona (conecta) las variables independientes con la dependiente. Esta función es similar a la función de activación para la capa de salida en una ANN.

3.3. Regresión de búsqueda de proyección (PPR – projection pursuit regression) (Klinke y Grassmann, 1998)

Es también un modelo de regresión no paramétrica. Para una sola variable de salida, puede ser escrito de esta forma:

$$y = \sum_{i=1}^{N} w_i \Phi_i \left(u_i^T x \right) + w_0 \quad \ldots(75)$$

donde los parámetros u_i son vectores unitarios direccionales y Φ_i, funciones de activación no lineales. Se le llama regresión de búsqueda de proyección porque el vector *x* es proyectado en los vectores direccionales para obtener las proyecciones $u_i^T x$, y mediante técnicas de optimización se buscan "buenos" vectores direccionales (así como los "buenos" w_i y Φ_i) que minimicen la función del error.

Podemos ver que este modelo es similar a una ANN anticipativa de dos capas, donde la segunda capa tiene una función identidad como función de activación. Una diferencia importante es que en el PPR cada "unidad escondida" puede tener una función de activación diferente, a pesar de encontrarse todas en la misma capa. Estas funciones son determinadas a partir de la data como parte del proceso de aprendizaje. Esto le otorga mayor libertad al modelo, pero le agrega complejidad. Aquí, además, se entrena la red de PPR un nodo (escondido) a la vez, mientras que en el ANN se entrenan todos simultáneamente.

Klinke y Grassmann (1998) sugieren que la popularidad de las ANN frente a las PPR se debe a una mayor facilidad para el entendimiento y para la programación, de las primeras sobre las segundas.

Otras técnicas estadísticas:

Otras técnicas comparables son los árboles de decisión (CART), cuando el problema es de clasificación; y la Regresión multivariada adaptativa usando splines (MARS), cuando el problema es de predicción.

Conclusión

En el campo de la ecología marina, se han realizado algunos trabajos de aplicación comparando las redes neuronales artificiales y otras técnicas estadísticas como regresión lineal múltiple (Brey *et al.*, 1996, Chen y Ware, 1999; Olden y Jackson, 2001; Baruch *et al.*, 2001; y Megrey *et al.*,2005), regresión logística (Olden y Jackson, 2001; y Baruch *et al.*, 2001), regresión no lineal y modelos GAM (Megrey *et al.*, 2005). En estos trabajos, las redes neuronales han resultado técnicas apropiadas para el modelamiento de dichos casos. También podemos encontrar en Suryanarayana (2008) una revisión de las aplicaciones de las redes neuronales en pesquería, refiriéndose a las áreas y objetivos específicos que abordaron los autores de las investigaciones conteniendo dichas aplicaciones, varios de ellos comparando las ANN con otras técnicas estadísticas y no estadísticas.

Cualquiera sea el problema de investigación, al escoger la técnica estadística a emplear, debemos tener en cuenta cuál(es) de ellas cumple(n) con el objetivo de estudio, el tipo de datos que manejamos (escalas de medición y tipos de variables), relación entre las variables

(dependencia/independencia), supuestos que se deben cumplir para cada técnica y finalmente, la eficiencia de esta (en simpleza, demanda de recursos, rapidez, etc.) en la ejecución del objetivo fundamental del estudio.

Las redes neuronales artificiales, se presentan como herramientas estadísticas muy útiles para diferentes tipos de aplicaciones (clasificación de patrones, clusters, aproximación de funciones, predicción, optimización, memorización asociativa, control de procesos; Jain y Mao, 1996), ofrecen bastante libertad para el tipo de datos de entrada y salida, y en muchos casos, no se necesita conocer la función de densidad de probabilidad de las variables ni la expresión analítica del modelo. Asimismo, los avances computacionales le brindan la velocidad que hoy en día se necesita para la obtención de resultados. Por estas razones, las ANN se perfilan como una técnica que, si se utiliza con el conocimiento suficiente y los criterios adecuados, puede ser aplicada para la solución de problemas de complejidades diversas.

CAPÍTULO III
METODOLOGÍA

1. Tipo de estudio

Los antecedentes y el marco teórico nos brindan criterios para la estructura de la red mas no hipótesis para la investigación. Por ello, la actual investigación responde a un análisis exploratorio para la cual no se tienen hipótesis previas.

En cuanto al manejo de las variables, el estudio es observacional.
Por la época de obtención de datos: retrospectivo.
De acuerdo a la evolución del fenómeno en estudio: transversal.
Y respecto a si hay comparación entre poblaciones: descriptivo.

2. Definición de la población de estudio

2.1. Características generales

2.1.1. Criterios de inclusión

Se utilizaron datos de embarcaciones anchoveteras recolectados por dos fuentes: el VMS y las bitácoras de observadores a bordo (de una muestra de barcos), que correspondan a los mismos viajes, entre 1999 y el 2006. El nivel de precisión “G” en las emisiones de VMS es requerido.

2.1.2. Criterios de exclusión

Datos del VMS con niveles de precisión inferiores.

2.1.3. Criterios de eliminación

Se eliminaron las primeras y últimas observaciones de cada viaje, puesto que al calcular valores de diferencia de velocidad con respecto a la observación anterior (para la primera observación) y con respecto a la siguiente observación (para la última), en esos casos resultaba “0”, lo que podía alterar el aprendizaje de la red.

2.2. Ubicación espacio-temporal

Datos de embarcaciones anchoveteras recolectados por VMS (con nivel “G”) y bitácoras, que correspondan a los mismos viajes, en todo el mar peruano, entre 1999 y el 2006.

3. Muestra

No se realizó un muestreo estadístico. Los datos de bitácoras pertenecen a una muestra de barcos y constituyen aproximadamente el 2% del total de embarcaciones, lo que se debe principalmente a las restricciones económicas. La metodología de IMARPE para la designación de observadores a las embarcaciones es de naturaleza determinística. Todos los viajes (trayectorias entre salida de puerto y retorno a puerto) registrados en bitácoras y los datos de VMS que correspondían a esos viajes fueron utilizados para formar las bases de entrenamiento.

4. Definición de variables y escalas de medición

La base de datos armada con datos de VMS y bitácoras, está compuesta por las siguientes variables:

4.1. Variables de primer orden

ID: Identificador de la observación. Es una variable categórica.

Código_del_barco: Es el código que representa a una embarcación particular. Variable categórica.

Código_de_viaje: Es el código con el que se representa a un viaje pesquero en particular. Un viaje pesquero comprende desde el momento en el que la embarcación parte del puerto, hasta que vuelve a un puerto. Es una variable categórica.

Tiempo: Es el tiempo en el que se emitió la señal del VMS. Fue separada en las siguientes variables:

Día: es el día en el que se transmitió la señal.

Mes: es el mes en el que se transmitió la señal.

Año: año en el que se transmitió la señal.

Hora: es el tiempo (momento) en horas en el que se transmitió la señal. Es una variable continua y circular.

La localización geográfica se expresa mediante dos variables:

Longitud: en grados sexagesimales. Variable continua y circular.

Latitud: en grados sexagesimales. Variable continua y circular.

Cala: variable indicadora. 1: cala; 0: no cala.

4.2. Variables de segundo orden (calculadas)

Velocidad: es la velocidad en la que se encuentra navegando la embarcación.

Sea x_i la ubicación actual de la embarcación con componentes (lat_i, lon_i); y x_{i-1} la ubicación inmediata anterior emitida por el VMS, con componentes (lat_{i-1}, lon_{i-1}).

La distancia entre ambos puntos será la distancia ortodrómica, que es la distancia más corta entre dos puntos de una esfera (la Tierra).

$$d_i = R \times a\cos\left[sen\left(lat_{i-1}\frac{\pi}{180}\right) sen\left(lat_i \frac{\pi}{180}\right) + \cos\left(lat_{i-1}\frac{\pi}{180}\right)\cos\left(lat_i \frac{\pi}{180}\right)\cos\left(lon_i \frac{\pi}{180} - lon_{i-1}\frac{\pi}{180}\right)\right]$$

…(76)

donde R es el radio promedio de la tierra (6 378 140 metros).

La velocidad será entonces:

$$v_i = \frac{d_i}{\Delta t_i} \quad \ldots(77),$$

donde Δt_i representa a la diferencia de tiempos entre la observación actual y la anterior.

4.3. Variables de tercer orden (calculadas)

Cambio_de_velocidad_1 (Δv_{-1}): es la diferencia entre la velocidad actual y la de la observación inmediata anterior.

$$\Delta v_{-1}{}^{i} = v_i - v_{i-1} \quad \ldots(78)$$

Cambio_de_velocidad_2 (Δv_{+1}): es la diferencia entre la velocidad de la observación inmediata posterior y la actual.

$$\Delta v_{+1}{}^{i} = v_{i+1} - v_i \quad \ldots(79)$$

*En los antecedentes se hizo referencia a la aceleración. Esta es igual al cambio de velocidad dividido entre el tiempo. Por lo tanto, lo expresado en la introducción sobre la relación entre aceleración y cala (desaceleración antes de la captura y aceleración después), se aplica también para los cambios de velocidades y la cala (diferencia de velocidades negativa antes de la cala y positiva después). Como nuestro interés central es la discriminación entre puntos de cala y no cala, más que el tiempo que se demora en pasar de un estado al otro, utilizaremos para este estudio los cambios de velocidades en lugar de la aceleración.

Diferencia_rumbo (|Δθ|): es la diferencia entre el rumbo (dirección en la que va la embarcación en grados sexagesimales) actual y el de la observación inmediata anterior.

El rumbo es calculado a partir de las coordenadas de los puntos (variables latitud, y longitud) y la distancia ortodrómica entre ellos.

$$\theta_i^* = \frac{180}{\pi} \times a\cos\left[\frac{sen\left(lat_i \frac{\pi}{180}\right) - sen\left(lat_{i-1} \frac{\pi}{180}\right)\cos(d_i)}{\cos\left(lat_{i-1} \frac{\pi}{180}\right) sen(d_i)}\right] \quad \ldots(80)$$

$$\Delta lon_i = lon_{i-1} - lon_i \quad \ldots(81)$$

$$\theta_i = \begin{cases} \theta_i = \theta_i^*, & \Delta lon_i \le 0 \\ \theta_i = 360 - \theta_i^*, & \Delta lon_i > 0 \end{cases} \quad \ldots(82)$$

5. Proceso de obtención de datos

Para la identificación de calas mediante la red neuronal, como se verá más adelante, se realizan 2 etapas:
La primera consiste en entrenar una red neuronal sobre una muestra de viajes de pesca para los cuales tenemos una información completa: a la vez las trayectorias observadas por satélites (VMS, que documenta toda la flota) y las posiciones de las operaciones de pesca (colectadas por

observadores a bordo de algunas embarcaciones, bitácoras). La flota muestreada a bordo representa menos del 2% del total de embarcaciones, por lo que para el 98% restante se necesita inferir los puntos de pesca. Por ello, la segunda etapa usará la red entrenada para estimar solamente a partir de los datos de trayectorias (VMS) los puntos probables de las operaciones de pesca de toda la flota pesquera.

5.1. Las bitácoras (registros de observación directa)

Las bitácoras de pesca son herramientas utilizadas a nivel mundial para cuantificar las capturas y el esfuerzo pesquero requerido para ellas.

En el Perú, "Bitácoras de Pesca" es un programa de monitoreo de pesca mediante observadores a bordo de las embarcaciones anchoveteras principalmente, aunque también se realiza para otras especies marinas. Constituye una importante herramienta de apoyo en la evaluación de la anchoveta y a las decisiones propias del manejo de su pesquería, complementaria a las fuentes tradicionales de información como el seguimiento en puertos de los desembarques y la evaluación por acústica durante cruceros científicos dedicados.

Se empieza a aplicar este programa en el país en 1974 (con el nombre de "Partes de pesca"; Csirke, 1990). Su uso surgió a raíz del colapso de la pesquería de anchoveta en 1972-1973, sintiéndose la necesidad de realizar un seguimiento más detallado de la actividad pesquera en las embarcaciones industriales, y así, tener indicadores más confiables sobre el esfuerzo pesquero, el número de operaciones de pesca por viaje y la duración de los mismos. En aquel entonces las fichas de bitácoras eran llenadas por los capitanes de todas las embarcaciones.

A principios de los años 1980 el programa fue descontinuado (Csirke, 1990) hasta su re-apertura en 1996. A partir de esa fecha los requerimientos de información se han ampliado y diversificado, siendo estos de carácter pesquero, biológico, físico, oceanográfico, etc. (Bouchon, 1998). Por ello las fichas (figura 26) también han sufrido modificaciones que responden a estos requerimientos (la última modificación se realizó a finales del 2007). Además, las fichas ya no son llenadas por los capitanes sino por observadores a bordo, personal capacitado y calificado del IMARPE, que realizan el monitoreo a las embarcaciones muestreadas por todo el tiempo que dura el período de pesca en el año. Para cada temporada de pesca, son aproximadamente 25 los observadores, cada uno a bordo de un barco diferente, de un total de 1200 embarcaciones. La, cuota es aparentemente pequeña - depende mucho de las restricciones presupuestarias y de personal –mas como cifra absoluta (25) indica un esfuerzo importante del IMARPE. Asimismo,, es necesario resaltar que el monitoreo que realiza IMARPE de la pesquería es único en el mundo, por ser de carácter continuo e intenso, por la calidad del programa bitácoras y por la accesibilidad a la información casi en tiempo real.

El aporte de las bitácoras para el presente estudio fue la información de los momentos y posiciones de cala durante los viajes[25].

5.2. El VMS

El sistema de seguimiento satelital de los barcos (Vessel Monitoring System o VMS en inglés) es un sistema satelital de monitoreo de los desplazamientos de las embarcaciones pesqueras. Inicialmente implementado por razones de control (pesca prohibida a dentro de las 5 millas costeras) y de seguridad (identificación de la posición de naves en dificultad), el gobierno peruano decretó la obligatoriedad del uso del VMS por parte de todas las embarcaciones anchoveteras de pesca tipo cerco desde el 2000 (Decreto Supremo N° 001-2000-PE). No obstante, algunas empresas adquirieron el servicio en los últimos meses de 1999, por lo que se cuenta con información de algunos barcos desde esas fechas.

[25] Un viaje de pesca es una sucesión de puntos en mar. Es decir, comprende desde el momento de partida de un puerto hasta el momento de llegada a un puerto.

En el Perú se trabaja mayormente con el sistema CLS-Argos, y el proceso de obtención y transmisión de información es el siguiente (figura 27):

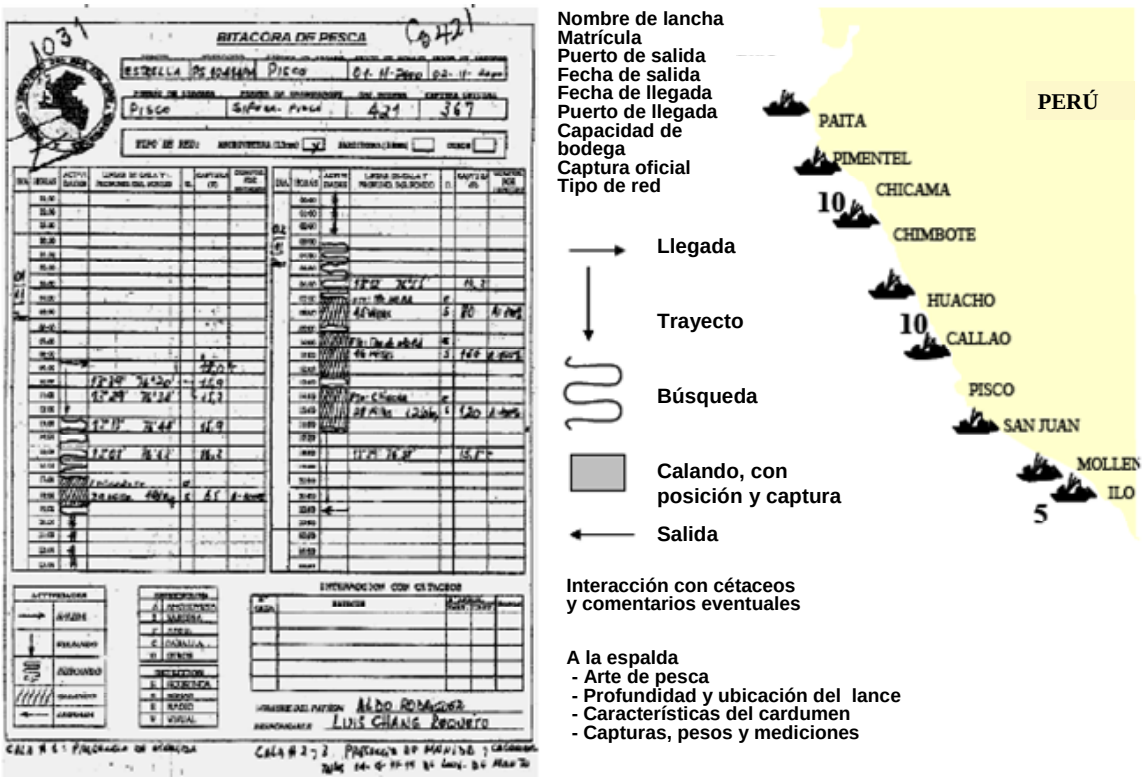


Figura 25. Ejemplo del formulario de bitácora para las embarcaciones pesqueras. Presentamos esta y no la última versión porque los datos que utilizamos en el presente trabajo provienen de este tipo de bitácora. Fuentes: IMARPE y Bertrand, 2005.

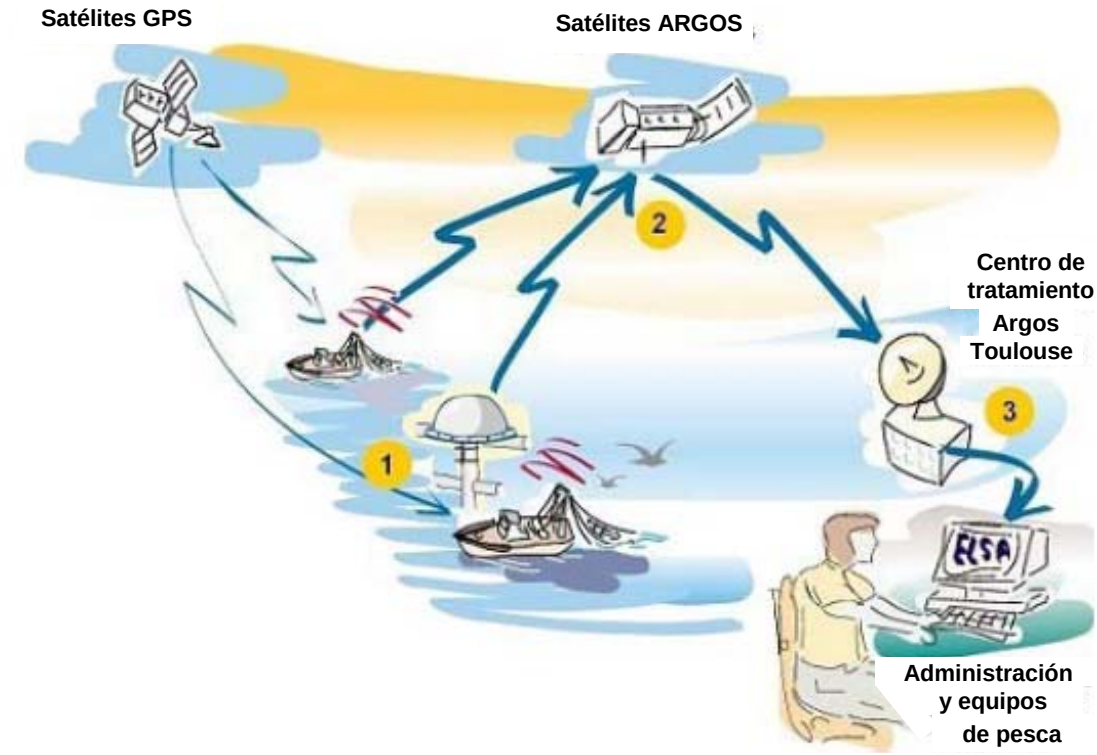


Figura 26. Recolección, procesamiento y distribución de la información por medio del VMS. Fuente: CLS Argos, http://www.argonet-vms.com/welcome_fr.html

Se introduce el equipo (baliza Argos/GPS, figura 28) a cada una de las embarcaciones. Las balizas son las que (1) recuperan la señal de localización por triangulación[26] desde la constelación de satélites GPS o por efecto Doppler desde los satélites Argos[27] y (2) envían esta información a los satélites de la constelación Argos que a su vez las transmiten a los centros de procesamientos terrestres, quienes se encargan de procesar los datos (darles el formato adecuado y efectuar los controles de calidad) y enviarlos a los usuarios (empresas pesqueras y administradores). (Argonet, http://www.argonet-vms.com/welcome_fr.html; Bertrand, 2005).

Figura 27. Baliza Argos. Fuente: Argonet

Clases de emisión:

El VMS transmite, con cierta frecuencia, las coordenadas (latitud, longitud en grados, minutos y segundos) de la posición de la embarcación, así como el tiempo en el que se captaron las señales (fecha, hora, minutos y segundos). La señal envía la posición junto con una clasificación que describe el proceso de localización (de la posición) y la precisión asociada a él:

CLASE	PROCESO DE LOCALIZACIÓN	PRECISIÓN (metros)
G	Estimada por el método de triangulación, con la ayuda de un receptor GPS fijado sobre la baliza.	100
0	Obtenida por el sistema Argos mediante el efecto Dopler.	> 1000
1		1000
2		350
3		150
A		no estimable
B		no estimable
Z	Suspendido	----

Tabla 4. Procesos de localización y medidas de precisión para las diferentes clases de señales VMS.

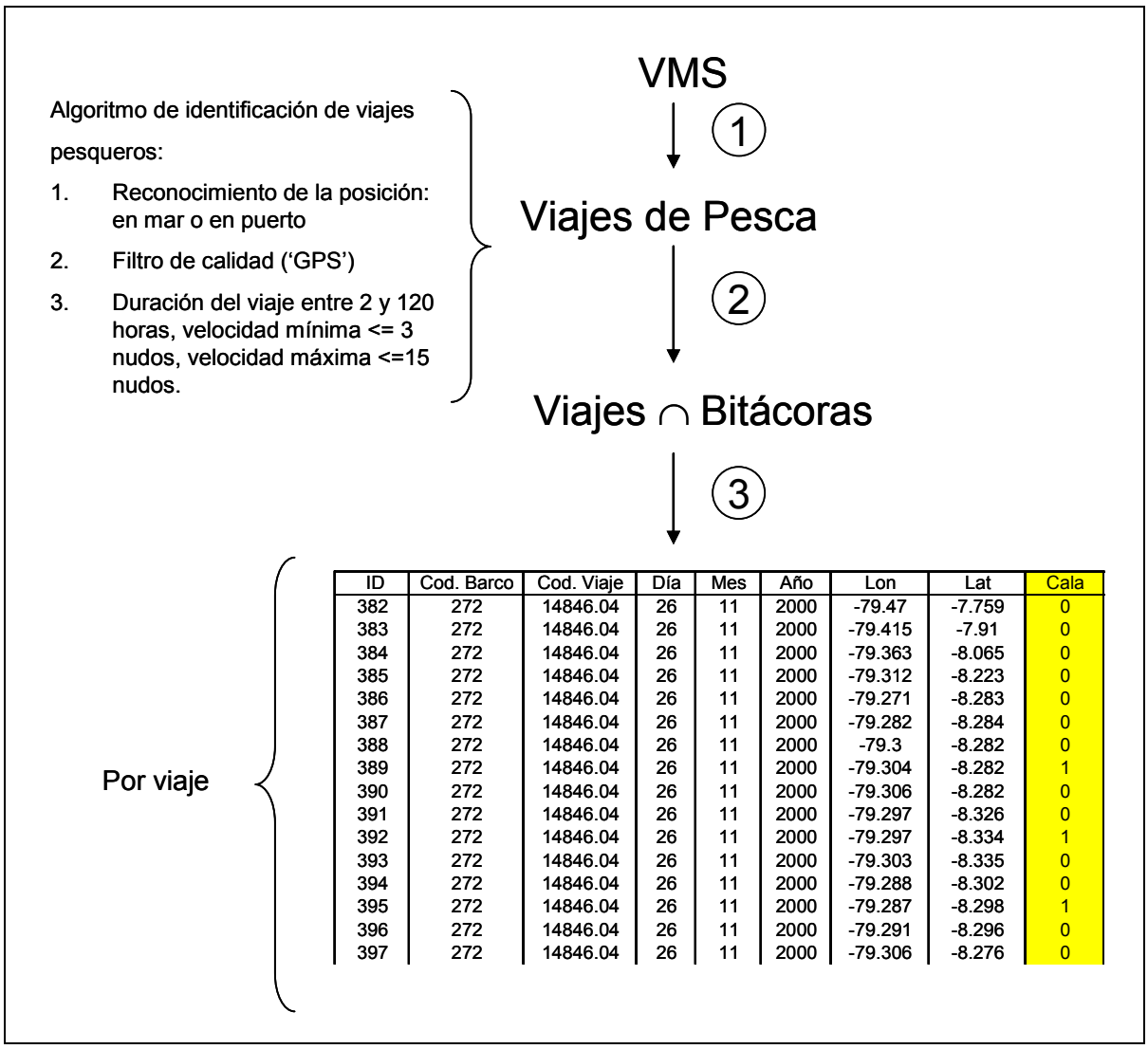
5.3. Pre-tratamiento de la base de datos

Los datos de posicionamiento obtenidos por VMS contienen observaciones correspondientes a embarcaciones anchoveteras tanto en mar como en puerto, pescando o no. Por ello la base original

[26] El principio de triangulación se aplica de la siguiente manera: 1) Un primer dato de algún satélite ubica al receptor en algún punto de la superficie de una esfera de radio igual a la distancia obtenida entre los dos dispositivos. 2) Luego se mide la distancia entre un segundo satélite y el receptor. Esto da como resultado que la ubicación real no es únicamente en la primera esfera, ni tampoco en la segunda, sino, que realmente el receptor estará en algún punto del círculo en el cual se intersectan las dos esferas. 3) Posteriormente se obtiene un tercer dato de otro satélite. Esto reduce los posibles puntos de ubicación del receptor a dos (los lugares geométricos posibles están en la intersección de las tres esferas). (Salazarweb.com)

[27] Las posiciones calculadas por la constelación Argos están basadas en el efecto Doppler, es decir la variación de frecuencia del 'ruido' según un objeto se aleja o se acerca. Sucede porque hay menos satélites ARGOS que satélites GPS y es similar a una triangulación; desafortunadamente el error cometido por efecto Doppler es mayor al obtenido por triangulación.

de datos debe ser pre-tratada (“limpiada”) antes de su utilización (Bertrand *et al.* 2005 y 2007) (figura 29).



ID	Cod. Barco	Cod. Viaje	Día	Mes	Año	Lon	Lat	Cala
382	272	14846.04	26	11	2000	-79.47	-7.759	0
383	272	14846.04	26	11	2000	-79.415	-7.91	0
384	272	14846.04	26	11	2000	-79.363	-8.065	0
385	272	14846.04	26	11	2000	-79.312	-8.223	0
386	272	14846.04	26	11	2000	-79.271	-8.283	0
387	272	14846.04	26	11	2000	-79.282	-8.284	0
388	272	14846.04	26	11	2000	-79.3	-8.282	0
389	272	14846.04	26	11	2000	-79.304	-8.282	1
390	272	14846.04	26	11	2000	-79.306	-8.282	0
391	272	14846.04	26	11	2000	-79.297	-8.326	0
392	272	14846.04	26	11	2000	-79.297	-8.334	1
393	272	14846.04	26	11	2000	-79.303	-8.335	0
394	272	14846.04	26	11	2000	-79.288	-8.302	0
395	272	14846.04	26	11	2000	-79.287	-8.298	1
396	272	14846.04	26	11	2000	-79.291	-8.296	0
397	272	14846.04	26	11	2000	-79.306	-8.276	0

Figura 28. Esquema del pre-tratamiento de los datos

Ya que lo que se necesitaba era trabajar solo con los viajes pesqueros, pues el objetivo del estudio es la identificación de calas, se comenzó por reconocer si la posición de la observación era en puerto o en mar, procediendo a eliminar de la base de datos aquellas posiciones en puerto. Para ello, se comparaba la posición del barco con la posición de todos los puertos. La posición se reconocía «en mar » si se encontraba a más de 2 millas náuticas de todos los puertos. A la inversa, la posición se reconocía « en puerto » si estaba dentro de un radio de 2 millas náuticas de alguno de los puertos (figura 30).

Un segundo filtro fue el de calidad, pues se decidió trabajar solo con los datos cuyas posiciones fueron estimadas con la mayor precisión posible (clase “G); y para los cuales la frecuencia entre 2 emisiones consecutivas era menor a 2 horas. Finalmente, de entre aquellos datos, se seleccionaron aquellos que pertenecían a viajes de pesca. La suposición que se hizo fue que los viajes de pesca debían cumplir las siguientes condiciones: durar entre 2 y 120 horas (5 dias), tener una velocidad mínima observada menor o igual a 3 nudos (velocidad máxima probable durante una cala) y una velocidad máxima menor o igual a 15 nudos.

Los viajes que no cumplían estos requisitos fueron eliminados de la base de datos (figura 29 – número 1).

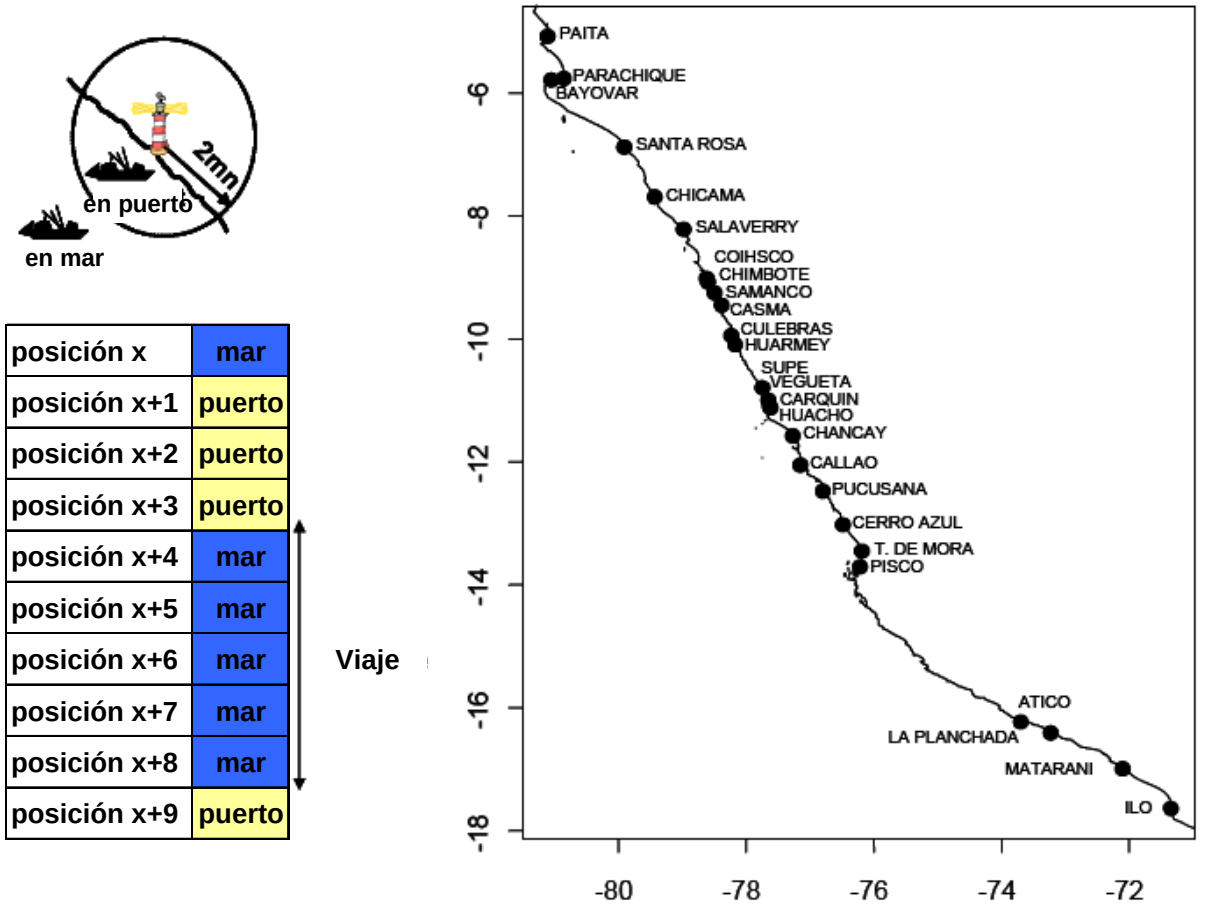


Figura 29. Reconocimiento de los viajes de pesca, utilizando los datos del VMS. Aquellas posiciones de barcos que se encontraban a un radio mayor a 2 mn de alguno de los puertos, fueron identificadas como "en mar". Un viaje es una sucesión de posiciones « en mar » (a la izquierda). Fuente: Bertrand, 2005.

Si bien el VMS aporta información sobre la posición de todas las embarcaciones en cada momento, y somos capaces de reconocer, mediante el algoritmo anteriormente descrito, los viajes de pesca de cada una de estas embarcaciones (Bertrand *et al.* 2005 y 2007), todo esto es insuficiente para reconocer los puntos de pesca. La información de las posiciones de las calas se obtiene a través de las bitácoras. Estas han sido tomadas únicamente de una muestra de embarcaciones; y la información que poseemos del VMS es de la totalidad de viajes pesqueros. Por esta razón era necesario cruzar la información de ambas bases de datos en 2 tiempos. En el primero, a nivel de viajes, fue para quedarnos solo con los viajes descritos por las bitácoras, obteniendo de esta manera, una base constituida por una muestra de viajes VMS (figura 29 – número 2). En el segundo, a nivel de observaciones dentro de los viajes, se hizo con el de conocer la ocurrencia o no ocurrencia de cala para cada una de los observaciones (emisiones) dentro de la base de datos. Para ello se comparó cada observación VMS con las de las bitácoras. Cuando una observación VMS coincidía en tiempo y espacio (posición) con un registro de cala en bitácora, esta información era registrada en una nueva columna de la base de datos VMS colocando en ella el valor "1" para dicha observación. Para los casos restantes, se colocó "0" (figura 29 – número 3).

Con todo esto, obtuvimos nuestra base de datos para el entrenamiento de la red.

Es importante mencionar que, durante el período 2000-2006 (que incluye también los últimos meses de 1999), las frecuencias de las emisiones han ido variando. En los últimos años la frecuencia de emisión es mayor que la de los primeros años; esta variabilidad puede afectar la calidad de los resultados de la red dependiendo la base temporal que se tome (puede haber diferencias en la eficiencia para una base completa (2000-2006) en comparación con las de otros períodos interanuales).

Por lo anterior, y en la búsqueda del mejor desempeño de la red, es importante no sólo analizar la sensibilidad de esta con respecto a los parámetros de entrada, sino además encontrar la mejor base de datos (de período anual, bianual, etc.) para la identificación de calas.

6. Procesamiento, análisis estadístico e interpretación de la información

Se utiliza una red neuronal artificial con las características siguientes:

6.1. Variables de entrada para la red

Hora, velocidad, cambio_de_velocidad_1, cambio_de_velocidad_2 y diferencia_rumbo. Bertrand et al. (2008) realizaron pruebas con otras variables como duración y distancia quedándose con las variables inicialmente enunciadas por dar mejores resultados y no presentar problemas de multicolinealidad.

6.2. Variable objetivo de la red

El objetivo es la variable cala, una categórica con valores “0” y “1”. El nodo de salida de la red (variable de salida) será comparado con la variable objetivo. Si la red está prediciendo (identificando) bien, los valores de ambas variables serán cercanos.

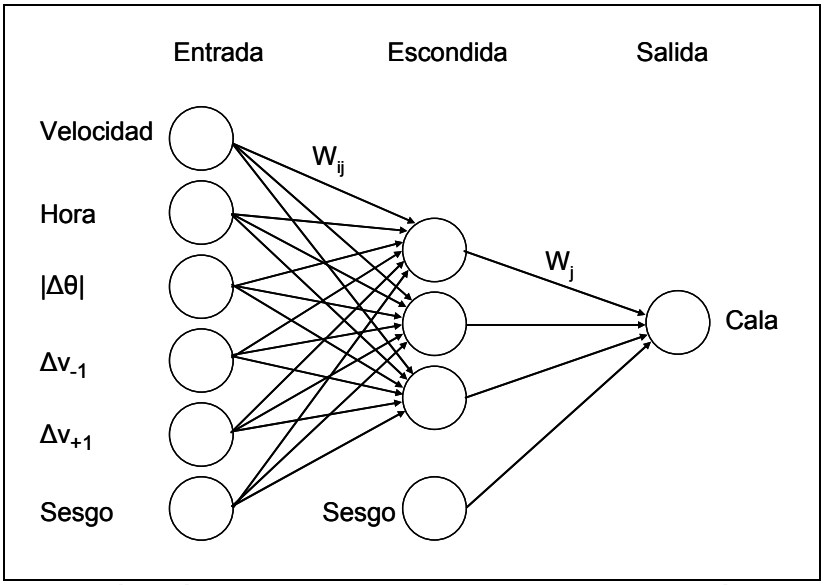


Figura 30. Representación gráfica de la red neuronal para la identificación de calas. Fuente: Bertrand et al., 2008.

6.3. Arquitectura de la red

La arquitectura es la de una red multi-capas, con una sola capa escondida cuya función de activación es la tangente hiperbólica sigmoidal. Para la capa de salida, la función de activación escogida fue la logística sigmoidal. La función del error considerada fue el Error Cuadrático Promedio (MSE).

Se decidió utilizar una red anticipativa multi-capas por diferentes razones: 1) estudios anteriores en pesca se han realizado con este tipo de red (Chen y Ware, 1999; Dreyfus-León, 1999; Gaertner y Dreyfus-León, 2004), 2) una red anticipativa es más parsimoniosa que una recurrente, pues sus conexiones son únicamente unidireccionales.

Como hemos señalado en la sección 2.6.1. del marco teórico, una red multi-capas con una capa escondida y función sigmoidal, puede aproximar con gran precisión cualquier función (recordemos que la tangente hiperbólica se diferencia de la logística sigmoidal por una transformación lineal). Una función de activación *tanh* para la capa escondida por lo general conduce a una convergencia más rápida que la *logsig*.

Otra alternativa podría ser utilizar una función de umbral para el nodo de salida, puesto que nuestra variable objetivo es una indicadora. El problema es que no tenemos información anterior para poder establecer el valor de umbral antes de entrenar la red. Por ello la mejor alternativa sigue siendo la logística sigmoidal, que, de rango <0,1>, permite darle interpretaciones probabilísticas a los patrones de salida (como una probabilidad de que sea cala).

El MSE es la función del error más comúnmente utilizada, función que no necesita asumir alguna distribución para el objetivo.

El entrenamiento de la red se realizó de manera supervisada con el algoritmo de retro-propagación Levenberg-Marquardt, cuyo tiempo de operación es muy reducido en comparación a otras, al aproximar la matriz hessiana mediante la jacobiana y así, evitando el cálculo de derivadas de segundo orden (ver la sección de Levenberg-Marquadt. en el capítulo II).

Para mejorar la generalización y disminuir el sobre-ajuste, se utiliza el criterio de parada temprana. Este criterio presenta ventajas sobre las otras formas de generalización en cuanto a una menor demanda de trabajo computacional. Además de eso, para aplicar la regularización bayesiana necesitaríamos 1) tener información a priori de los datos, que nos permita conocer o suponer una distribución de probabilidad conjunta o 2) tomar por supuesto una función de verosimilitud para una distribución normal (lo que usualmente hacen los softwares). En nuestro caso, no se cumplen ninguna de las condiciones mencionadas. Además, Bertrand et al. (2008), para una base de datos más pequeña espacial y temporalmente –pero que está contenida dentro de la nuestra -, utilizaron la regularización bayesiana y sus resultados no superaron a los de la red con parada temprana.

Aunque el algoritmo Levenberg-Marquardt reduce el riesgo de atascarse en un mínimo local (a comparación de Gradiente descendiente), esto puede seguir ocurriendo en el entrenamiento. Por ello se aplicó un filtro a aquellas corridas en las que ocurría un mínimo local no aceptable.

El entrenamiento de la red se realizó en el software Matlab versión 7.5., con ayuda del toolbox Neural Net (ver anexo 1)[28].

6.4. Parámetros a los que se les realizó las pruebas de sensibilidad

Si bien se ha definido esta parte de la estructura de la red, hay otras que se desconocen a priori y que se concretaron después del entrenamiento:

6.4.1. Umbral

Sabemos que nuestra variable objetivo es una indicadora (0=”no cala” y 1=”cala”) y la de salida es una continua que viene como resultado de una *logsig* con valores entre 0 y 1. Cuando el patrón de salida es cercano a “0”, se considera como “no cala” y cuando es cercano a “1” se le considera “cala”. Una primera aproximación –usual cuando el investigador no posee criterio alguno ni información previa –es tomar un umbral igual a 0.5, lo que le da a la salida igual probabilidad de ser considerado “cala” o “no cala”. Sin embargo, tener probabilidades iguales puede llevarnos a cometer errores

[28] Se tomó como base el algoritmo de entrenamiento de Bertrand et al. (2008); apoyándonos en la ayuda de Matlab, así como en García et al. (2001) y los códigos de Xiong (2006) para la programación de las pruebas de sensibilidad de la red.

quizás demasiado grandes. Fue entonces necesario hallar un umbral que permitiera discriminar los patrones de salida en las 2 categorías de la variable objetivo, sin cometer errores de clasificación grandes.

6.4.2. Número de neuronas

Es el número de nodos o neuronas de la capa escondida de la red. No hay una regla o criterio para determinar el número de nodos a utilizar en la capa escondida. Tener muy pocas neuronas (un número insuficiente de neuronas) puede dar un elevado error en el entrenamiento y en la generalización, al haber sub-ajuste y residuales grandes. Por el contrario, tener demasiadas neuronas (un número mayor al necesario) puede dar un elevado error en la generalización, al haber sobre-ajuste y gran variabilidad en el MSE (Sarle, 2002).

Un mayor número de neuronas aumenta la complejidad de la red y por lo tanto también el tiempo de operación. Un requisito para la posterior aplicabilidad de la red, es la parsimonia y rapidez de operación. Por ello se testeó la red prefiriéndose elegir aquella con la menor cantidad de neuronas que dé los mejores resultados.

6.4.3. Número de bucles

Para la parada temprana, como se explicó anteriormente, se divide la base de datos en 3 partes: base para el entrenamiento, para la validación y para el test. El criterio por el que se optó (para hacer la división) fue la elección aleatoria de las observaciones que conformarían cada una de las partes. Al ser una elección aleatoria, se decidió correr la red varias veces para que el factor de "elección aleatoria de la base de datos" no sea determinante en los resultados de la red. Así, en cada una de esas corridas o bucles se elige nuevamente de manera aleatoria las observaciones que corresponderán a cada una de las 3 partes y la red neuronal vuelve a realizar todo su proceso de aprendizaje.

De este modo, se quiso hallar el número óptimo de bucles, es decir un número de veces suficientemente grande de tal manera que la red no sea muy sensible a la manera en que los patrones de datos han sido ordenados en los 3 grupos. Por otro lado, se tomó en cuenta también que un número de bucles muy grande conduce a una menor rapidez para el entrenamiento.

Así, el número de bucles es el número de veces que se correrá la red.

6.4.4. Tamaño de las bases

No hay un criterio para determinar el tamaño de las bases para el entrenamiento, la validación y el test. Muchas veces depende de los datos, del problema en cuestión y del criterio del investigador (Sarle, 1995). Es necesario que la base para el entrenamiento sea suficientemente grande como para tener suficiente datos con los cuales entrenar la red (y ser así, una base de datos representativa del problema), pero una base para entrenamiento demasiado grande llevará a tener para el test demasiado pequeña, lo que podría ocasionar varianzas grandes en las predicciones, alargando el tiempo de convergencia de la red. (El anexo 1 muestra la programación realizada para efectuar esta prueba de sensibilidad).

6.4.5. MSE máximo (MSEmax)

El MSE máximo (Error Cuadrático Promedio máximo) es el máximo valor que puede tomar el MSE (de entrenamiento) por cada bucle de la red. Si el valor del MSE en la corrida sobrepasa el máximo especificado (el parámetro MSE máximo), se vuelve a correr el algoritmo de la red sin que el anterior quede registrado.

Es importante regular este parámetro, porque aunque a mayor MSE máximo, el tiempo de entrenamiento es menor, valores demasiado grandes podrían llevar a problemas de mínimo local y no

salir de allí. Es decir, la función del error puede quedar en un mínimo local y afectar con ello en la calidad de los resultados. Sin embargo, tener un valor demasiado pequeño del MSEmax puede hacer que todo el proceso demore bastante en correr al tratar de cumplir la restricción, y en el peor de los casos, no permitir pasar de bucle en ningún momento, por lo cual la red nunca logra aprender y entonces jamás llega a una solución.

Por otro lado, si bien el MSE máximo reduce significativamente el riesgo de que la base de entrenamiento caiga en un mínimo local (que lleva a que suceda lo mismo con la del test, ver la sección sobre mínimo local a continuación), para tener la certeza de que en la base de test no sucede lo mismo, se añadió al algoritmo un filtro para el que si el número de calas identificadas por la red es cero (para la base de test), esa corrida no quede registrada y se entrene la red nuevamente (ver las sintaxis en el anexo 1). Esto impide que suceda en el entrenamiento, mas no en la predicción.

6.5. Indicadores de desempeño de la red

Para determinar los valores adecuados de los parámetros de la red anteriormente descritos (umbral, número de neuronas, número de bucles, tamaño de las bases y MSE máximo) se realizaron pruebas de sensibilidad para diferentes valores de estos parámetros.

Los criterios de elección, además de los empíricos ya señalados, fueron los indicadores de eficiencia de la red.

Antes de describirlos, es necesario presentar algunas definiciones:

Calas observadas (CO): Son las calas que han sido registradas en bitácoras por los observadores a bordo y que, por lo tanto, aparecen registradas bajo el valor “1” en la variable objetivo. Entonces, el total de calas observadas es el total observaciones que tengan “1” en la variable objetivo.

Calas identificadas (CI): Son las observaciones o emisiones que han sido identificadas como calas por la red, después del proceso de entrenamiento de esta. Más adelante se explicará el papel del umbral para la identificación de calas por la red.

Calas verdaderas (CV): Son las observaciones que han sido correctamente identificadas como calas por la red. Es decir, en la variable objetivo toman el valor 1 (porque fueron registradas como calas en las bitácoras) y, a su vez, la red las ha identificado como “calas”.

Calas fantasmas o falsas (CF): Son las observaciones que han sido erróneamente identificadas como calas por la red; aquellas para las que la variable objetivo toma el valor 0 (lo que significa “no cala”); sin embargo, la red las ha identificado como “calas”.

Se debe tomar en cuenta que, al realizarse varios bucles de la red, seleccionando aleatoriamente las bases de entrenamiento, validación y test para cada una de ellas, el número de calas para cada base en cada bucle, es diferente.

Los indicadores de desempeño se obtienen tanto para las bases de entrenamiento, validación y test:

6.5.1. Indicadores de eficacia

a) Porcentaje de calas verdaderas (% CV):

Es el total de calas verdaderas reconocidas por la red entre el total de calas observadas en todos los bucles.

MÍNIMO LOCAL

El objetivo del aprendizaje de la red es siempre la minimización de su función del error. Por defecto, la red siempre buscará lograr un MSE=0 (considerando MSE como la función del error, lo cual es cierto en nuestro caso). Pero este no es el único criterio. En la parada temprana, la red deja de entrenar cuando el error de validación está desarrollando una tendencia creciente, lo que representa una alerta a la capacidad de generalización de la red y un riesgo de sobre-ajuste. La red reconoce esta tendencia cuando el MSE de la validación va creciendo durante un número de iteraciones especificado (max_fail) antes de correr la red.

En ese camino de aprendizaje y entrenamiento de la red, se ha ido disminuyendo el error (MSE); pero puede darse el caso que cuando se llega a max_fail aún el MSE es muy grande. Por lo tanto, la "distancia" entre los patrones de salida y los de objetivo será grande también. Si eso ocurre para la base de entrenamiento, será aún peor para la base del test, porque si ni siquiera está funcionando bien para la propia data entrenada, menos para una ajena al entrenamiento. Eso llevará entonces a problemas de reconocimiento (identificación) de calas, incluyendo aquellos casos en los que la red no llega a identificar cala alguna.

Este, por ejemplo, es un caso de mínimo local. La red se queda en una solución que no es la óptima. A un mínimo local se puede caer de diferentes maneras y muy frecuentemente; lo que realmente importa es que, si se ha caído en un mínimo local, que éste no se aleje mucho del mínimo global. En el caso planteado, donde el número de calas identificadas por la red es cero, este "no reconocimiento" de calas distiende mucho de un mínimo global teórico en el que el número de calas identificadas por la red es igual al número de calas observadas.

Una salida para este problema la da el parámetro MSE_max, que restringe el MSE del bucle hasta cierto valor especificado. Cuando el MSE de esa corrida alcanza valores mayores al MSE_max –lo que significa que el error es demasiado grande -, el bucle se vuelve a correr, y la red se vuelve a entrenar. Esto ocurre hasta que el MSE del bucle cumpla los requerimientos del parámetro. Recordemos que en cada bucle la red vuelve a realizar todo el proceso de entrenamiento y aprendizaje, por lo que MSE de la etapa de entrenamiento cambiará. Tener un MSE bajo en el entrenamiento es lo que dará posibilidades de tenerlo así para el test, lo cual permitirá cumplir la restricción y pasar de bucle.

Que el MSE (la función del error) se haga suficientemente pequeño implica entonces que la variable salida se aproxima a los valores de la variable objetivo, y hay una buena identificación de calas, para cada una de las bases (entrenamiento, validación y test), salvando las distancias porcentuales.

Por todo esto, se debe tener cuidado en escoger el valor de MSE_max, ya que un valor muy grande será casi como no tener la restricción, y se caerá nuevamente en mínimos locales. Sin embargo, tener un valor demasiado pequeño del MSE_max puede hacer que todo el proceso demore bastante (porque le costará muchos bucles para pasar la restricción), y en el peor de los casos, no permitir pasar de bucle en ningún momento, por lo cual la red nunca logra aprender y entonces nunca llega a una solución.

Al encontrarnos con estos problemas, colocamos además en el algoritmo, una restricción que había que volver a entrenar cuando en un bucle la red no reconocía calas. de calas en el entrenamiento, y ciertamente Esto anulaba la posibilidad de no-identificación reducía la probabilidad de obtenerla en el test –al mejorar el entrenamiento –pero no resolvía el problema del todo.

# bucle	1	2	8	9	10	11	33	34	35	39	40
MSE_test	0.054381	0.018854	0.020417	0.082115	0.022068	0.021707	0.025113	0.085489	0.025661	0.022406	0.08324
# epochs	44	80	36	9	43	39	51	6	31	43	4
MSE_tr	0.056368	0.016358	0.019226	0.092308	0.021405	0.021773	0.018311	0.093433	0.017647	0.019762	0.098311
CO	76	82	85	73	80	89	84	76	87	80	74
CV	0	74	63	0	62	77	57	0	58	61	0
CF	0	14	9	0	14	12	13	0	12	12	0
ratio	NaN	5.2857	7	NaN	4.4286	6.4167	4.3846	NaN	4.8333	5.0833	NaN

Tabla. Caso particular de mínimos locales (MSE 0.04 para la base de entrenamiento 2000-2001). Las columnas sombreadas corresponden a iteraciones en las que ocurren nulas identificaciones de calas, que coinciden con valores muy altos del MSE del entrenamiento y del test.

Indica el porcentaje de calas verdaderas que han sido reconocidas por la red. Se le divide entre el total de calas observadas porque se quiere comparar con el número de calas que realmente sucedieron.

$$\%CV = \frac{\sum_{i=1}^{n} cv_i}{\sum_{i=1}^{n} co_i} * 100 \quad \ldots(83)$$

donde cv_i: número de calas verdaderas en el bucle *i*;
co_i: número de calas observadas para el bucle *i*;
n: número de bucles

b) Porcentaje de calas fantasmas (% CF):

Es el total de calas falsas de las reconocidas por la red entre el total de calas observadas en todos los bucles.
Indica el porcentaje de calas fantasmas que han sido reconocidas por la red. Se le divide entre el total de calas observadas porque se quiere comparar con el número de calas que realmente sucedieron.

$$\%CF = \frac{\sum_{i=1}^{n} cf_i}{\sum_{i=1}^{n} co_i} * 100 \quad \ldots(84)$$

donde cf_i: número de calas fantasmas en el bucle *i*;
co_i: número de calas observadas para el bucle *i*;
n: número de bucles

c) Ratio CV/CF (RATIO):

Es un ratio entre el total de calas verdaderas y el total de calas falsas identificadas por la red.
Nos da una razón de cuántas calas verdaderas está reconociendo la red por cala fantasma que identifica.

Es el promedio de los ratios entre calas verdaderas y falsas de cada bucle. Es el número de calas verdaderas que identificamos por cada cala falsa que obtenemos.

$$CV/CF = \frac{\sum_{i=1}^{n} cv_i}{\sum_{i=1}^{n} cf_i} \quad \ldots(85)$$

donde: cv_i: número de calas verdaderas en el bucle *i*;
cf_i: número de calas fantasmas en el bucle *i*;
n: número de bucles

6.5.2. Indicadores de eficiencia

a) MSE:

La función del error que se utiliza es el error cuadrático promedio (MSE - Mean Squared Error). Se utiliza como función de eficiencia de la red.

$$MSE = \frac{\sum_{i=1}^{n}\sum_{j=1}^{p}(y_i^j - t_i^j)^2}{np} \quad \ldots(86)$$

donde: *n*: número de bucles;

p: número total de observaciones de la base para la que se está calculando el indicador (de entrenamiento, validación o test);

y_i^j **:** salida *j* dentro del bucle *i*;

t_i^j : objetivo *j* dentro del bucle *i*;

b) Desviación estándar del MSE (STD(MSE)):

Es la desviación estándar de los MSE calculados para cada bucle. Es una medida de variabilidad en los valores de la función del error (MSE) entre los bucles.

6.5. Predicción (identificación)

Realizados ya los tests de sensibilidad y obtenidos los valores para los parámetros, se entrena la red con aquellos valores. Una vez entrenada la red, se procede a la predicción.

La predicción se realiza para aquellos datos obtenidos a partir del VMS que cuenta con información de las variables de entrada a la red, pero de la que no sabemos para cuáles patrones sucedieron calas. La variable de salida será nuevamente una continua entre 0 y 1, para la cual, utilizando el umbral obtenido en las pruebas de sensibilidad, se clasifican las observaciones en “calas” y “no calas” (ver anexo 1, en la parte de predicción).

CAPÍTULO IV
RESULTADOS

Introducción

Se realizaron las pruebas de sensibilidad para los parámetros de umbral, número de nodos en la capa escondida, número de bucles, tamaño de las bases y MSE máximo; todo esto para la base completa (2000-2006) y remuestreada (con un criterio de frecuencia de emisión mayor a 0.5 horas), luego con bases bianuales (2000-2001 y 2005-2006) y trianuales (2000-2002 y 2004-2006)

1. Duración de las emisiones

Como se mencionó anteriormente, las frecuencias de las emisiones del VMS han ido variando a través del tiempo; la duración entre 2 emisiones es menor ahora que en los primeros años de uso del VMS (figura 32). Recordemos que variables como velocidad (por ende, cambios de velocidades) y diferencia de rumbo, no son extraídas directamente por las emisiones del VMS, sino obtenidas indirectamente a través de los datos de localización y hora. La red neuronal aprende a identificar calas en base a esos datos. Si para diferentes bases de tiempo la duración entre emisiones cambia significativamente, de igual manera variará el cálculo de los datos de entrada; lo que hará que la red aprenda, para diferentes períodos de tiempo, modos diferentes de reconocer calas. Eso complicaría su trabajo de entrenamiento para la base completa (u otras a posteriori que abarquen muchos años con duración entre emisiones muy distintas) y nos demandaría buscar períodos para la base de entrenamiento en los que la variabilidad de la duración de emisiones no sea alta ni difiera mucho de la aquella de la base de datos para la predicción.

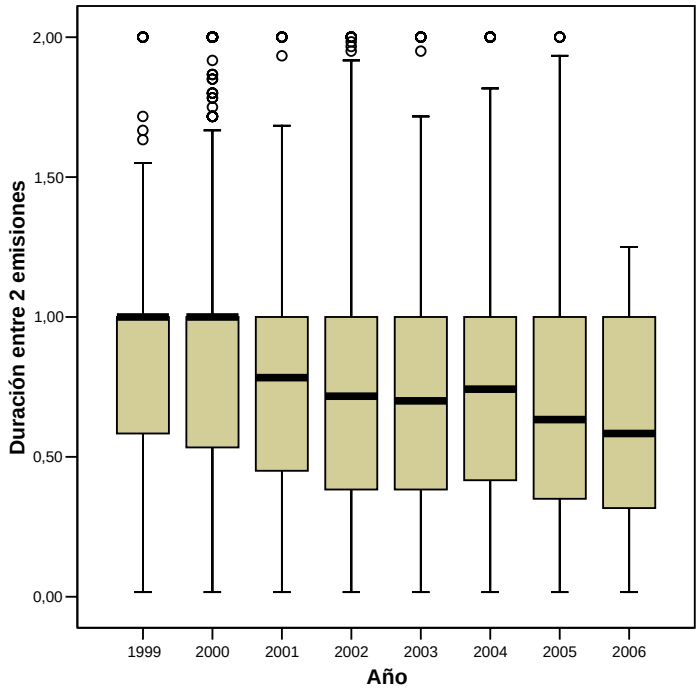


Figura 31. Diagrama de cajas de la duración entre emisiones por año entre 1999 y 2006

Por esta razón, la base de datos con la que se efectuó la prueba fue limpiada y remuestreada de tal manera que la frecuencia entre dos emisiones seguidas sea mayor a 0.5 horas. Esto consiguió que la mediana del promedio de frecuencias de emisión por viaje para cada uno de los años sea 1 hora (figura 33).

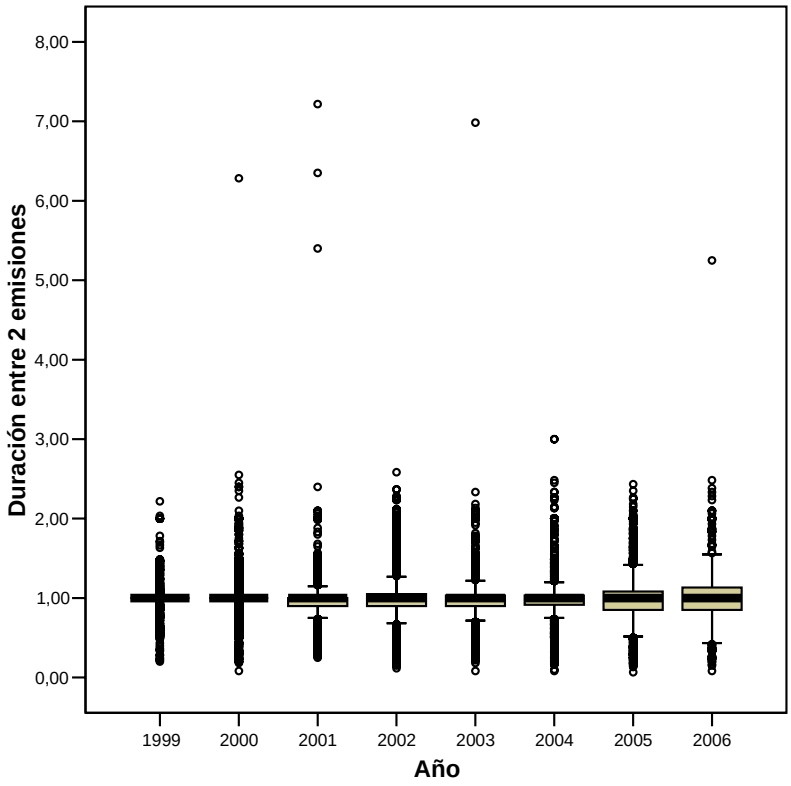


Figura 32. Diagrama de cajas de la duración promedia entre emisiones por año para la base de datos remuestreada. La mediana de las frecuencias de emisiones es 1 hora.

2. Pruebas de sensibilidad para la base 2000-2006 remuestreada

DATOS DE LA BASE TOTAL 2000-2006	
Total de observaciones	14967
Total de calas	1905
Valores iniciales:	
Umbral	0.5
Bucles	50
Nodos	4
Tam. Entrenamiento	66.70%
Tam. Validación	16.70%
Tam. Test	16.70%
MSE máximo	0.05

Tabla 5. Se muestra el número de observaciones y calas; así como los valores iniciales de los parámetros (umbral, bucles, nodos, tamaño de la base para el entrenamiento, tamaño de la base para la validación, tamaño de la base para el test y MSE máximo) antes de testear la sensibilidad de la red.

Los valores iniciales para las pruebas de sensibilidad (tabla 5), a excepción del último, fueron los utilizados para la red de Bertrand *et al.* (2008) para una base de entrenamiento de período trianual 2000-2002. Ellos utilizaron un MSE máximo de 0.04, que tuvimos que incrementar a 0.05 por la no convergencia de la red a valores de MSE menores a 0.04 para cada uno de los bucles (el incremento del período de entrenamiento puede haber incrementado la variabilidad en el modelo).

Cuando, mediante los tests de sensibilidad, obtengamos el "valor óptimo" de un parámetro[29], éste será el nuevo valor inicial de tal parámetro para las siguientes pruebas de sensibilidad de dicha base.

[29] Cuando hablamos de valor óptimo, nos referimos a "mejor" o "más adecuado", en el sentido de la discusión sobre "el mejor modelo" presentada en el capítulo 2.

2.1. Umbral:

Se decidió elegir aquel umbral para el que el número total de calas identificadas por la red (sean estas calas verdaderas o fantasmas) sea igual al total de calas observadas (Bertrand *et al.*, 2008). Este criterio evita que la red identifique demasiadas calas (en relación al total de observadas). Interesa para los objetivos de esta investigación, no sobrestimar el número de calas totales; y como segunda prioridad, tener una correcta identificación de calas (un muy alto porcentaje de calas verdaderas y un muy bajo porcentaje de calas fantasmas). Además, al tener una cantidad de calas identificadas igual al 100% de calas observadas, las calas falsas adquieren otra dimensión: **no serían totalmente falsas (pues realmente ocurrieron), solo que nos estamos equivocando en su ubicación**.

Se testeó la red variando el umbral desde 0.50 hasta 1 incrementándolo en 0.05 para cada ensayo (anexo 2.1.). La intersección o momento en el que el número de calas identificadas se hace igual al de observadas, se encontraba entre 0.5 y 0.55, por lo que se realizaron nuevos test para el umbral entre 0.5 y 0.55, esta vez con incrementos de 0.01, encontrando la intersección en 0.52 (figura 34).

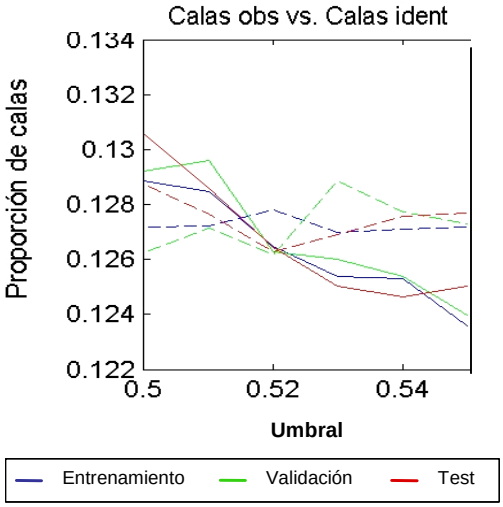


Figura 33. Total de calas observadas vs. Total de calas. Se muestra la variación del umbral entre 0.5 y 0.55. Las líneas continuas representan a la proporción de calas observadas y las líneas discontinuas, a la proporción de calas identificadas.

Como acabamos de explicar, si escogiéramos un umbral menor al de la intersección (0.52 en este caso), la red sobrestimaría el número de calas (anexo 2.1.). Esto implicaría una elevación del porcentaje de calas verdaderas (en relación al porcentaje de CV para nuestro umbral) pero una elevación también en el porcentaje de calas falsas (anexo 2.2.). Lo contrario sucede cuando se elige un umbral mayor al de la intersección: se subestima el número de calas total y por lo tanto, disminuyen los porcentajes de CV y CF (anexo 2.2.).

2.2. Bucles:

Se testearon redes que empleaban desde 15 hasta 85 bucles para el entrenamiento. Las redes que utilizaron 55 o más bucles mostraron más estabilidad en sus resultados que las que usaron menos bucles (anexo 2.3.). Esto sugiere una sensibilidad de las redes hacia la manera en que se han ordenado los datos, cuando tienen pocos bucles; efecto que es contrarrestado con un aumento en su número. Por ello, escogimos 60 bucles como nuevo valor del parámetro.

2.3. Nodos:

Inicialmente, el propósito era testear la red variando su arquitectura desde 1 hasta 8 neuronas en la capa escondida (aunque 8 implica tener una red muy compleja, fue para chequear variaciones en los indicadores). No se pudo realizar el entrenamiento de la red con 1 sólo nodo escondido, porque la red

no lograba converger: no cumplía la restricción MSE máximo ni para el primer bucle. Para alcanzar la convergencia, el MSE máximo debía ser menos restrictivo, elevándolo de 0.05 a 0.65.

Cuando es necesario incrementar el MSE máximo para lograr la convergencia, significa que el modelo que la red va armando en el proceso de aprendizaje está teniendo errores muy grandes. Lo que nos quiere decir, en primer lugar, que hay una relación entre los parámetros MSE máximo y número de nodos –no son independientes -, y en segundo lugar, que en este caso en concreto, una arquitectura conteniendo solo una neurona en la capa escondida no es eficiente.

Se testeó entonces la red usando desde 2 a 8 nodos sin presentar las complicaciones anteriores. El porcentaje de calas fantasmas fue el indicador más sensible a los cambios en la arquitectura de la red, disminuyendo al aumentar el número de unidades escondidas hasta 4 (figura 35 – base de test). A partir de ahí, al añadir más neuronas a la arquitectura de la red, comienza un comportamiento de subidas y bajadas en el porcentaje de CF. Por otro lado, la red aprende más rápido con 4 neuronas, aunque las diferencias en tiempo con las otras redes no es mucha (anexo 2.5.). Así, tanto en resultados como en velocidad, la red neuronal con 4 nodos muestra un mejor desempeño que las otras.

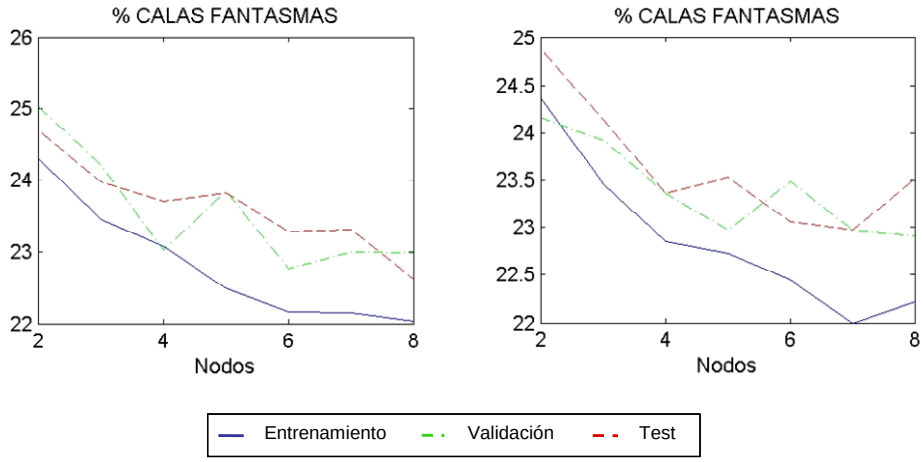


Figura 34. Porcentaje de calas fantasmas, testeando el número de nodos en la capa escondida desde 2 hasta 8 (2000-2006). Aquí se muestran los resultados de 2 réplicas de la prueba. (Los resultados para los demás indicadores se presentan en el anexo 2.4.)

2.4. Tamaño de las bases (para el entrenamiento, la validación y el test):

Se deseaba encontrar un equilibrio de tal manera que la base para el entrenamiento fuera lo bastante grande como para que la red pueda aprender; y que a la vez, la base para la validación fuera suficientemente grande para que la red aprenda a generalizar y la del test, también de un tamaño adecuado como para que esta generalización se pudiera ver efectiva.

El análisis se realizó variando simultáneamente los tamaños (proporciones) de las bases para el test y la validación –y consecuentemente de la base para el entrenamiento, cuyo tamaño es el complemento. Se probó variando el tamaño para el test desde 5% hasta 50% y el tamaño para la validación, desde 5% hasta 40% del tamaño total de la base de datos.

Tamaño para entrenamiento (%) = 100 – tamaño para test (%) – tamaño para validación (%)

En el anexo 2.6. se muestran todas las combinaciones de porcentajes de tamaños para las que se obtuvieron los indicadores de desempeño.

La elección del tamaño de las bases se hizo en base a los resultados (indicadores) de desempeño de la red aplicada a la base del test, pues es la única porción de la base de datos que no participó en la etapa de entrenamiento. Considerando aquellos tamaños de base que proporcionaban CV mayores al 75%, ratios CV/CF altos (mayores a 3.2), MSE y su correspondiente desviación estándar pequeños (anexo 2.7.), obtuvimos no un tamaño óptimo para cada partición de la base de datos, sino una "región de eficiencia", en la que el tamaño de la base de test puede estar entre el 15% y el 25%, y el de la base de entrenamiento, entre el 60% y el 80% del total de la base. Una "región de eficiencia" en lugar de un "tamaño óptimo", es reflejo de la alta variabilidad de los resultados, que impide visualizar valores puntuales de los tamaños de las particiones para los que la red refleje un mejor desempeño de manera evidente. Los tamaños que se encuentran dentro de la región de eficiencia pueden interpretarse entonces (por los resultados obtenidos) como zonas en la que la probabilidad de sacar buenos resultados de identificación con la red es mayor.

Para la siguiente prueba de sensibilidad (MSE máximo), se dividió la base de datos en 75% para el entrenamiento, 10% para la validación y 15% para el test.

2.5. MSE máximo

Se evaluó la sensibilidad de la red a los valores de MSE máximo, desde 0.045 hasta 0.07, incrementándolo en 0.001 en cada ensayo. Con valores menores a 0.045 la red demoró mucho en converger y en el peor de los casos (y la mayoría de ellos) no llegaba a converger a la soluciones. Mientras más grande es el valor del parámetro, la red se hace menos restrictiva y por lo tanto, el tiempo de aprendizaje disminuye. No obstante, recordemos que una menor restricción en la red puede llevar a mínimos locales en los que la identificación de calas es muy mala (incluso suceder que en algunos bucles no se identifique cala alguna). En el anexo 2.8., se pueden detectar cambios bruscos en los valores de los indicadores para MSE máximos mayores a 0.05. Esto debido a la variabilidad en los resultados que puede generar un MSE máximo demasiado alto (muy poco restrictivo).

2.6. Conclusión

Así, después de efectuar las pruebas de sensibilidad respectivas, los nuevos valores de los parámetros son:

Umbral	0.52
Bucles	60
Nodos	4
Tam. Entrenamiento	60% - 80%
Tam. Test	15% - 25%
Tam. Validación	complemento
MSE máximo	0.05

Tabla 6. Valores de los parámetros de la red luego de los tests de sensibilidad (base 2000-2006).

Y el desempeño de la red con ellos:

CV	77% - 78%
CF	23% - 24%
RATIO	3.1 - 3.2

Tabla 7. Indicadores de eficacia para la red después de realizar las pruebas de sensibilidad y establecer los valores de los parámetros (base 2000-2006).

3. Entrenando bases bianuales

A través de los años, es posible que la manera de actuar de los pescadores –cuya información se refleja de cierta manera en las variables de entrada a la red –haya cambiado (influidos por una parte, por la reducción de días de pesca en estos años), así como la duración entre las emisiones de VMS – con efectos en los cálculos de las variables de segundo y tercer orden, como se explicó anteriormente. La variabilidad creada por estos factores se debe manifestar la base total. En cambio, una base bianual coge un intervalo de tiempo más pequeño y por lo tanto, más homogéneo (los cambios aquí no serán tan grandes), por lo que se espera que con ella se puedan obtener mejores indicadores de desempeño (aunque al hacer esto, se pierde capacidad de generalización, y la robustez de los indicadores es menor). Por ello se testeó la red con bases bianuales y se analizaron los resultados. Las bases bianuales no fueron remuestreadas por criterios de frecuencia de emisión, puesto que la variación entre estas fue a nivel de años, es decir, año a año la emisión de señales de VMS fue cada vez más frecuente. Por ello, tomando 2 años consecutivos el efecto de esto no sería tan determinante como para bases de períodos más largos (figura 32).

Por restricciones de tiempo, se realizaron las pruebas para 2 conjuntos de bases bianuales: 2000-2001 y 2005-2006 (cuyos datos iniciales se muestran en la tabla 8). El análisis fue comparativo pues se deseaba analizar qué tanto diferían los resultados entre ambas bases y con ello determinar si el factor “base de datos” es un factor importante para el entrenamiento.

DATOS DE LA BASE TOTAL 2000-2001	
Total de observaciones	4443
Total de calas	420
DATOS DE LA BASE TOTAL 2005-2006	
Total de observaciones	5900
Total de calas	511
Valores iniciales:	
Bucles	50
Nodos	4
Tam. Entrenamiento	60%
Tam. Validación	20%
Tam. Test	20%
MSE máximo	0.02

Tabla 8. Se muestra el número de observaciones y calas; así como los valores iniciales de los parámetros (umbral, bucles, nodos, tamaño de la base para el entrenamiento, tamaño de la base para la validación, tamaño de la base para el test y MSE máximo) para ambos períodos de tiempo antes de testear la sensibilidad de la red.

3.1. Umbral:

Se testeó la red variando el umbral desde 0.50 hasta 0.8, incrementándolo en 0.01 para cada ensayo. Este rango de valores fue elegido por ensayo-error, como ocurrió para la base completa 2000-2006. Esta vez, para la base 2000-2001 se había probado primero con un rango más pequeño de 0.5 a 0.65 (anexo 3.1.) para el que no se llegó a la intersección entre calas observadas e identificadas, por lo que se testeó se amplió el rango hasta un umbral de 0.8, el mismo que se utilizó por consiguiente para las pruebas de la base 2005-2006. La intersección se encontró en 0.72 para la base 2000-2001 (figura 36 - izquierda).y en 0.66 para la base 2005-2006 (figura 36 - derecha). Ambas figuras muestran cómo el número de calas identificadas por la red va disminuyendo (reduciéndose la sobrestimación) hasta aproximarse al de calas observadas (intersección) luego de lo cual continúa disminuyendo alejándose esta vez del número de CO (aumentando la subestimación). Así como para la base 2000-2006, los indicadores de desempeño de la red varían significativamente al cambiar el umbral (anexos 3.2. y 3.3.). Por eso, sea cual sea el criterio por el cual se escoja el umbral, es importante saber hasta qué porcentaje de calas verdaderas estamos dispuestos a disminuir o calas falsas a aumentar.

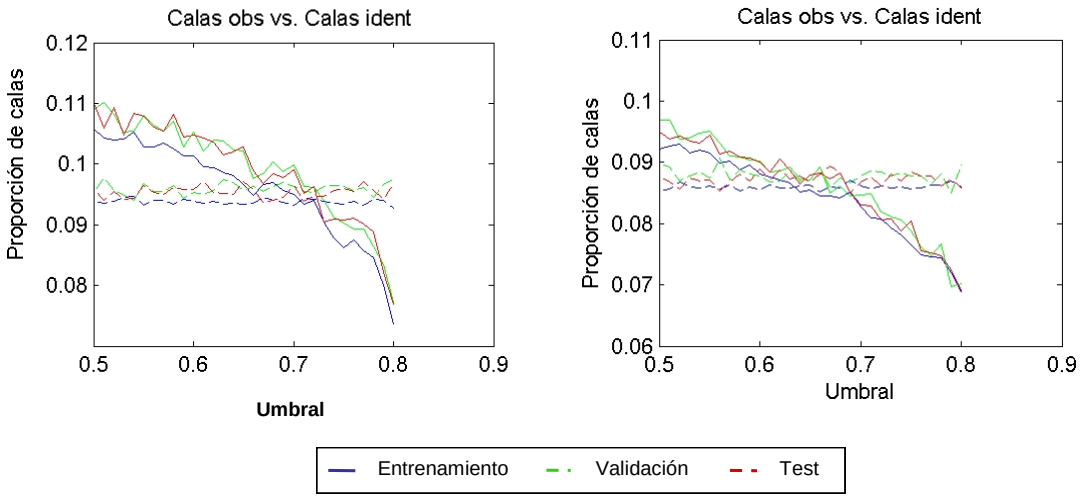


Figura 35. Total de calas observadas vs. Total de calas identificadas. A la izquierda, para la base 2000-2001 y a la derecha, para la base 2005-2006. Se varió el umbral desde 0.5 hasta 0.8 incrementando el umbral en 0.01 para cada ensayo.

3.2. Bucles:

Nuevamente se hicieron pruebas para las redes que usando entre 15 y 85 bucles para el entrenamiento. Se halló una mayor estabilidad en las redes desde 50 y 35-40 bucles, para las bases 2000-2001 y 2005-2006, respectivamente (ver %CV y %CF del anexo 3.4.); pues a partir de dicho número de bucles, los indicadores iban convergiendo (aunque, como se aprecia en los anexos 3.5. y 3.6., el número de bucles no causa grandes variaciones en los indicadores de desempeño de la red).

3.3. Nodos:

Se testeó la red neuronal con arquitecturas de 2 a 8 nodos (escondidos) para la base 2000-2001, y de 3 a 8 nodos para la base 2005-2006; esto por los problemas de convergencia –detallados en el análisis de la base completa –que presentaron las redes con 1 nodo (base 2000-2001 y 2005-2006) y 2 nodos (base 2005-2006).

En la primera base (anexo 3.7. – izquierda), el porcentaje de calas fantasmas disminuye hasta cuando se utilizan 3 nodos; en 4 vuelve a aumentar. En la segunda (anexo 3.7. – derecha), el %CF aumenta en la primera variación (de 3 a 4 nodos) Exactamente lo contrario sucede con el ratio. Se mencionó en la sección de métodos que el criterio de elección sería aquella red con el menor número de neuronas que dé los mejores resultados. En ambos casos, ese número es 3, porque al aumentar una unidad extra empiezan a aumentar las calas fantasmas y disminuir el ratio CV/CF.

Aunque la convergencia más rápida no se da para las redes de 3 neuronas, la diferencia en tiempo no representa costos significativamente grandes para su aplicación. Es notoria la diferencia entre las velocidades de convergencia para ambas bases (anexo 3.8.). Una causa posible sería la gran variabilidad dentro de la última, lo que no le permitiría converger rápidamente a soluciones que minimicen la función del error hasta el máximo impuesto: 0.02.

3.4. Tamaño de las bases:

Nuevamente se probó variando el tamaño para el test desde 5% hasta 50% y el tamaño para la validación, desde 5% hasta 40% del tamaño total de la base de datos (anexo 2.6.). Las pruebas de sensibilidad para este parámetro utilizaron un MSEmax=0.03, para la base 2005-2006, por

restricciones de tiempo (con MSEmax=0.02 el test demoraba varios días, y para MSEmax=0.03, era cuestión de horas); mientras que para todas las otras pruebas para ambas bases temporales (exceptuando la del MSE máximo) fueron realizadas con un MSEmax=0.02.

El desempeño de la red en el momento del test fue diferente entre las 2 bases de tiempo. Así, los criterios para la elección de la región de eficiencia fueron distintos también:

Para la base 2000-2001 (anexo 3.11. - izquierda), se eligieron tamaños de base para los cuales CV>79% y CF<21%; mientras que para la base 2005-2006 (anexo 3.11. - derecha), se debía cumplir que CV>77% y CF<22%. En cuanto al ratio, como en general para ambas bases los resultados se mostraron homogéneos, se descartaron aquellos tamaños cuyos resultados mostraban mucha variabilidad. Además, en las dos bases también, se excluyeron aquellos tamaños para los que el MSE era mayor o igual a 0.03, y su desviación estándar, mayor o igual a $7*10^{-3}$.

Con las restricciones impuestas, se estableció una "región de eficiencia" de la red para la base 2000-2001 para valores de la base del test a partir del 20%, de entrenamiento, a partir de 35% y de la validación, del 10%. Mientras que para la base 2005-2006, la "región de eficiencia" está comprendida entre el 10% y 30% de la base para el test y entre 45% y 75% para el entrenamiento. Para ambas bases temporales, los valores iniciales se ubican dentro de la región. En el siguiente test de sensibilidad, los valores iniciales del parámetro se mantendrán iguales.

3.5. MSE máximo:

Se evaluó la sensibilidad de la red a los valores de MSE máximo, desde 0.02 hasta 0.07, incrementándolos en razón de 0.005.

El porcentaje de calas verdaderas disminuyó cuando el MSE máximo se incrementó (este comportamiento sucedió hasta MSEmax=0.03 para la base 2000-2001 (-4.5%), y hasta MSEmax=0.05 para la base 2005-2006 (-5.5%)). A su vez, el porcentaje de calas fantasmas disminuyó cuando MSEmax cambió de 0.02 hasta 0.03 (-3%), solo para la base 2000-2001; pues para la base 2005-2006 comenzó una tendencia creciente en las CF (+1,5%). El desempeño de la red fue, en general, mejor cuando esta era más restrictiva (MSEmax=0.02). Es importante notar también que para valores muy grandes de MSEmax, la variabilidad de la función del error (cuantificada a través del std(MSE)) aumentaba. (anexo 3.12.); por ello, aunque los resultados en los indicadores no varían mucho al cambiar el valor del parámetro (anexos 3.13. y 3.14.), es preferible quedarnos con el menor, 0.02.

3.6. Conclusión:

De esta manera, después de efectuar las pruebas de sensibilidad respectivas, los nuevos valores de los parámetros son:

Base 2000 - 2001		**Base 2005 - 2006**	
Umbral	0.72	Umbral	0.66
Bucles	50	Bucles	40
Nodos	3	Nodos	3
Tam. Entrenamiento	>35%	Tam. Entrenamiento	45% - 75%
Tam. Test	>20%	Tam. Test	10% - 30%
Tam. Validación	>10%	Tam. Validación	complemento
MSE máximo	0.02	MSE máximo	0.02

Tabla 9. Valores de los parámetros de la red luego de los tests de sensibilidad. A la izquierda, para la base 2000-2001. A la derecha, para la base 2005-2006.

Y el desempeño de la red con ellos:

Base 2000 - 2001		Base 2005 - 2006	
CV	82%	CV	81%
CF	20%	CF	18%
RATIO	4.4 - 4.9	RATIO	4.3 - 4.5

Tabla 10. Indicadores de eficacia para la red después de realizar las pruebas de sensibilidad y establecer los valores de los parámetros. A la izquierda, para la base 2000-2001. A la derecha, para la base 2005-2006.

4. Entrenando bases trianuales

Si bien las bases bianuales retornan mejores indicadores que la base total entrenada,

INDICADORES	BASE	
	TOTAL	BIANUALES
CV	77% - 78%	81% - 82%
CF	23% - 24%	18% - 20%
RATIO	3.1 - 3.2	4.3 - 4.9

Tabla 11. Comparación de los indicadores de eficacia de la base total y las bianuales.

habíamos postulado que esto ocurriría pues al ser bases más pequeñas (en la dimensión temporal), son más homogéneas y por lo tanto con ellas la red aprende mejor –con mejores indicadores de eficiencia como resultado -. Por otro lado, el costo de utilizar bases más chicas implica una cierta pérdida de robustez. Por ello, se realizaron pruebas con las bases trianuales 2000-2002 y 2004-2006 para testear la eficiencia de la red aprendiendo de estas. Se esperaba además que los indicadores para estas redes tengan valores intermedios entre los de la base total y las bianuales.

DATOS DE LA BASE TOTAL 2000-2002	
Total de observaciones	9738
Total de calas	860
DATOS DE LA BASE TOTAL 2004-2006	
Total de observaciones	9624
Total de calas	809
Valores iniciales:	
Bucles	50
Nodos	3
Tam. Entrenamiento	60%
Tam. Validación	20%
Tam. Test	20%
MSE máximo	0.03

Tabla 12. Se muestra el número de observaciones y calas; así como los valores iniciales de los parámetros (umbral, bucles, nodos, tamaño de la base para el entrenamiento, tamaño de la base para la validación, tamaño de la base para el test y MSE máximo) para ambos períodos de tiempo antes de testear la sensibilidad de la red.

4.1. Umbral:

Al igual que para los períodos de tiempo anteriores, se testeó la red variando el umbral desde 0.50 hasta 0.8, incrementándolo en 0.01 para cada ensayo. La intersección se encontró en 0.67 para la base 2000-2002 (anexo 4.1. - izquierda).y en 0.6 para la base 2004-2006 (anexo 4.1. - derecha). Los indicadores trianuales de la red también varían grandemente como ocurrió con las bases de tiempo bianuales (anexos 4.2. y 4.3.).

4.2. Bucles:

Así como ocurrió en las bases de tiempo testeadas anteriormente, se hicieron pruebas para las redes que usando entre 15 y 85 bucles para el entrenamiento; y se observó que para ambas bases, los indicadores de desempeño se estabilizan a partir de 40 bucles (anexo 4.4.).

4.3. Nodos:

Para ambas bases temporales se realizaron pruebas de sensibilidad para redes usando entre 2 y 8 nodos.

Los resultados para el porcentaje de calas fantasmas, el ratio CV/CF y el MSE, para ambas bases de tiempo, presentan mejoras (disminución, aumento y disminución, respectivamente), cuando se aumenta el número de nodos hasta 4 (anexo 4.7.). En cuanto a los tiempos de entrenamiento, no hay grandes diferencias entre las arquitecturas testeadas (máximo 4 minutos –anexo 4.8.). Es decir, para las dos bases trianuales, una red con 4 nodos presenta mejores desempeños y se muestra más eficiente que redes con arquitecturas más simples (parsimoniosas).

4.4. Tamaño de las bases:

Los valores para los tamaños de base fueron los mismos que para las bases de tiempo anteriores. Aquí también el desempeño de la red fue diferente entre las 2 bases de tiempo y por ello los criterios para la elección de la región de eficiencia fueron distintos:

Para la base 2000-2002 se eligieron tamaños de base para los cuales CV>79% y se descartaron aquellos con ratios menores a 3.9; mientras que para la base 2004-2006 se incluía a aquellos para los que CV>78% y se eliminaron aquellos con CF>22% y RATIO>=3.6. Además, para ambas bases, se excluyeron aquellos tamaños para los que el MSE era mayor a 0.026, y su desviación estándar, mayor o igual a $4*10^{-3}$. (anexo 4.11.)

Así, la región de eficiencia para la base 2000-2002 comprende tamaños para el test entre 10% y 45%, y para el entrenamiento, a partir del 45% de toda la base; y para la base 2004-2006, tamaños del test entre 15% y 35% y del entrenamiento, a partir del 35% de la base de datos. En este caso, al igual que en los anteriores, los valores iniciales de tamaños con los que hemos testeado los demás parámetros se encuentran dentro de la región.

4.5. MSE máximo:

Se evaluó la sensibilidad de la red a los valores de MSE máximo, desde 0.03 hasta 0.07, incrementándolos en razón de 0.01 (anexo 4.12.). Al aumentar el valor del parámetro (y hacer la red menos restrictiva), crecía también el valor del MSE (indicador de eficiencia y función del error). El aumento del MSE es contraproducente, pues con MSE más grandes la variabilidad en las respuestas de la red es mayor (ver también la columna STD del anexo 4.14) y hay mayores riesgos de caer en mínimos locales de sub-ajuste en la identificación de calas. En razón a ese riesgo y dado que la red no toma mucho tiempo en aprender, decidimos quedarnos con el valor inicial, 0.03.

4.6. Conclusiones:

Así, después de efectuar las pruebas de sensibilidad respectivas, los nuevos valores de los parámetros son:

Base 2000 - 2002		Base 2004 - 2006	
Umbral	0.67	Umbral	0.6
Bucles	40	Bucles	40
Nodos	4	Nodos	4
Tam. Entrenamiento	>=45%	Tam. Entrenamiento	>=35%
Tam. Test	10% - 45%	Tam. Test	15% - 35%
Tam. Validación	complemento	Tam. Validación	complemento
MSE máximo	0.03	MSE máximo	0.03

Tabla 13. Valores de los parámetros de la red luego de los tests de sensibilidad. A la izquierda, para la base 2000-2002. A la derecha, para la base 2004-2006.

Y el desempeño de la red con ellos:

Base 2000 - 2002		Base 2004 - 2006	
CV	81%	CV	80%
CF	19%	CF	20%
RATIO	4.2 - 4.5	RATIO	3.7 - 4

Tabla 14. Indicadores de eficacia para la red después de realizar las pruebas de sensibilidad y establecer los valores de los parámetros. A la izquierda, para la base 2000-2001. A la derecha, para la base 2005-2006.

La diferencia en el desempeño entre las redes entrenadas con bases bianuales y trianuales es muy pequeña. Un poco más grande es la diferencia entre éstas y la base total (Tabla 15).

INDICADORES	BASE		
	TOTAL	BIANUALES	TRIANUALES
CV	77% - 78%	81% - 82%	80% - 81%
CF	23% - 24%	18% - 20%	19% - 20%
RATIO	3.1 - 3.2	4.3 - 4.9	3.7 - 4-5

Tabla 15. Comparación de los indicadores de eficacia de la base total, las bianuales y las trianuales.

CAPÍTULO V
DISCUSIÓN

1. Importancia del análisis de sensibilidad

Como se mencionó en la introducción, para una técnica como las redes neuronales que ofrece tantas posibilidades para regular su arquitectura, entrenamiento, algoritmo, etc., era necesario un análisis de sensibilidad que nos permitiera definir adecuadamente los parámetros de la red, para cuya elección no era suficiente basarse en la teoría de las ANN ni del problema pesquero en estudio.

Descubrimientos hechos a partir del análisis. Antes de las pruebas de sensibilidad, se creía que algunos parámetros a los que la red sería más sensible podían ser el número de nodos o el tamaño de las particiones de la base, pero a raíz del análisis se obtuvieron algunos resultados inesperados:

- **La red se mostró muy sensible a las variaciones en el umbral.** Esto quiere decir que del criterio con el que se elija el umbral (número de calas identificadas igual al de observadas, maximizar las calas verdaderas, minimizar las falsas, etc.) dependerá bastante los valores de los indicadores de eficacia de la red. También significa que, si queremos maximizar las calas verdaderas o minimizar las falsas, habrá que establecer restricciones para estos criterios. Porque pedir un máximo de CV significa reconocer muchísimos puntos como calas, lo sean o no, para así tener mayor probabilidad de que entre todos los puntos identificados como calas, estén contenidos todos los que verdaderamente son calas. Lo contrario sucede cuando pedimos minimizar el número de calas falsas. Aquí, poner un umbral muy alto de tal manera que casi ningún punto (quizás ninguno) sea reconocido como cala, otorga una menor probabilidad de que dentro del pequeño conjunto de puntos identificados como calas se encuentre uno que en la realidad no es cala. Así, las restricciones o límites que se deban colocar estarán probablemente relacionadas al número total de calas identificadas –que no sobrepase por mucho el de calas observadas, en el caso de maximización de CV, o que por el contrario, no descarte en la identificación muchas calas observadas, al minimizar CF –y quizás también analizar bien la eficiencia de la red (relacionada a la cuantificación del error). Asimismo, si se elige algún otro criterio para la elección del umbral, se deberá analizar las consecuencias que traerá su aplicación y tomar algunas consideraciones al respecto que regulen el desempeño de la red.
- **Es necesario que el MSE máximo sea el mínimo posible.** El MSE máximo es un parámetro opcional de la red; es decir, la red puede aprender y predecir sin la necesidad de este parámetro. Se había colocado en un principio para evitar que el error (MSE, en la base del test) sea demasiado grande, y cuando sucedía se volvía a entrenar la red. Pero un error (MSE) grande para la base del test ocurría como consecuencia de un MSE grande para la base de entrenamiento (discutiremos más adelante porqué no siempre la función del error del entrenamiento llega a su mínimo posible al hacer una "parada temprana"). Entonces, un MSE máximo pequeño produce una red restrictiva para el error, permite un mejor entrenamiento y minimiza el riesgo de caer en un mínimo local, en el que la red se queda en una solución que no es óptima (por ejemplo y como ha pasado en varios casos, sin poder reconocer ninguna cala). Por ello es necesario establecer un MSE_máx con el valor mínimo posible. Posible, porque valores demasiado pequeños podrían provocar una red tan restrictiva que demore un tiempo excesivo en cumplir con la condición (muchas horas o incluso días) o en el peor de los casos, una red que nunca pueda cumplir su propia condición durante el entrenamiento (y entonces jamás llegar a una solución).
- **La cantidad de neuronas escondidas y de bucles tienen poca influencia en el desempeño de la red.** Aunque sí se presentan variaciones en el desempeño de la red en respuesta a los cambios en los valores de ambos parámetros (más en el caso de los nodos –presentados en las tablas de los anexos 3.9., 3.10., 4.9. y 4.10.; que en los bucles – anexos 3.5., 3.6., 4.5. y 4.6.), éstas son pequeñas, más aún en comparación a aquellas ocasionadas por los cambios en el umbral (anexos 3.2., 3.3., 4.2. y 4.3.).
- **Identificación de una región de eficacia para los tamaños de las particiones de la base de datos.** Previamente al análisis, lo que se esperaba era que las pruebas de sensibilidad nos dieran suficientes recursos para elegir valores puntuales de los tamaños de las bases para entrenamiento, validación y test. Pero no pudimos encontrar las "combinaciones mágicas" que nos brindaran los mejores resultados de la red. En su lugar, obtuvimos regiones

o combinaciones de tamaños para las cuales tanto el desempeño como la eficiencia de la red eran buenas.

- **Pocas diferencias entre las bases bianuales y trianuales.** Después de realizado el análisis de sensibilidad y escogido los "mejores" parámetros para cada base de tiempo, las diferencias en porcentajes de calas verdaderas y falsas entre ambos tipos de bases son de 1 o 2 puntos porcentuales, indicando quizás que la variabilidad contenida en una base trianual es casi igual a la contenida en una base bianual (aunque se esperaba que hubiera más variabilidad en la primera que en la segunda).

2. Resultados de los indicadores de eficacia

Actualmente, IMARPE utiliza a la velocidad como único criterio para la identificación de calas, con lo cual se sobrestima considerablemente las calas (+182%; Bertrand *et al.*, 2008).

La red neuronal identifica correctamente un 80% de las calas –en promedio, entre las diferentes bases temporales. Y al haber sido entrenada con un criterio (umbral) para el que no se deben sobrestimar ni subestimar el número de calas, el otro 20%, definido como calas falsas, serían entonces calas mal ubicadas. De este modo, las redes neuronales artificiales se convierten en el método que mejores resultados otorgan en cuando a la identificación de calas; recomendándose su uso como insumo para la construcción de un CPUE espacializado u otros indicadores que contribuyan al monitoreo continuo de las operaciones de pesca que realiza IMARPE.

En el anexo 5, podemos observar la ubicación espacial de las calas identificadas por la red neuronal a partir de datos de viajes pesqueros entre los 7ºS y 10ºS, recolectados entre el 17 y el 30 de noviembre y luego entre el 6 y el 15 de diciembre del 2007, para 393 barcos. Las 5338 calas identificadas –considerando que no todos los barcos realizaron viajes de pesca todos los días en los que se recogieron los datos –es un número razonable para una zona altamente pesquera como son Chimbote y sus alrededores.

3. Sobre los indicadores elegidos para el análisis de sensibilidad

3.1. Indicadores de eficacia:

En problemas de clasificación binaria, se suele emplear los indicadores de sensibilidad y especificidad para evaluar el desempeño del modelo o la técnica. Para este caso, se definiría sensibilidad o fracción de verdaderos positivos, como la probabilidad de identificar como cala a una verdadera cala; y especificidad o fracción de verdaderos negativos, como la probabilidad de clasificar como no cala a un patrón que verdaderamente no es cala.

Como en nuestro caso el criterio para el umbral es CO=CI (CO: calas observadas; CI: calas identificadas), la fórmula para calcular el %CV (83) se parece mucho a la de sensibilidad; sin embargo no ocurre lo mismo con el %CF y la especificidad. Mientras que la especificidad es la fracción de verdaderos negativos, el %CF mide el porcentaje de falsos positivos; pues como lo que investiga IMARPE es el número y locaciones de las calas, nuestro interés es el porcentaje de equivocaciones de la red al identificar calas, y no cuántas "no calas" fueron identificadas correctamente.

Luego un tercer indicador, matemáticamente igual a la división entre los otros dos indicadores, señala la cantidad de calas correctamente identificadas por cada cala falsa.

3.2. Indicadores de eficiencia:

Los indicadores de eficiencia tienen como fin evaluar la productividad (en tiempo, orden, veracidad y precisión en la estimación, etc.). En este sentido, el indicador que usamos es el MSE, que mide el error que produce la red al estimar los valores de la salida (de la variable cala: continuos, entre 0 y 1) con respecto a los observados desde la base de datos (de la variable objetivo: dicotómica, de ceros y unos). Por un lado, es lógico pensar en él como indicador de eficiencia pues el MSE es la función del error más utilizada en redes neuronales (Bishop, 1995; Warner y Misra, 1996; Goethals *et al.*, 2007) y en este caso, es la función del error de la base (aunque el Matlab permite ingresar medidas del error propias, las alternativas que presenta en principio son el MSE y algunas derivadas de este). Pero por otro lado, el MSE está pensado como una función que cuantifica el error para variables continuas (su misma fórmula nos lo hace intuir). El problema entonces es encontrar una función para el error apropiada para medir diferencias entre los valores de una variable continua y otra categórica. Hasta el momento no hemos encontrado literatura al respecto.

Una alternativa que ahora presentamos, elaborada con la colaboración de Edith Seier (PhD, Mathematics Department of the East Tennessee State University), es la concepción de una "Distancia a la Decisión Correcta".

Para definirla, partimos recordando la manera en que la red decide si el valor de salida es cala o no: la discriminación la hace mediante el uso del umbral. Si el patrón de salida es menor a éste, se considera "no cala", de lo contrario, es "cala". Pues bien, algunas veces la red identifica correctamente y otras veces no. Para aquellos patrones en que la red identifica correctamente, al error que comete lo cuantificaremos como "0". Y para aquellos en los que la red se equivoca, diremos que el error que comete es la diferencia entre el patrón de salida y el umbral. Es decir,

$$\begin{cases} decisión \quad correcta: & d_j = 0; \\ decisión\, equivocada: & d_j = y_j - u; \end{cases}$$

donde d_j es el error o la distancia *j*-ésima a la decisión correcta,
y_j es el patron *j*-ésimo de salida de la red y
u es el parámetro umbral

Luego, la nueva función del error, la distancia a la decisión correcta (DDC), sería:

$$DDC = \frac{\sum_{j=1}^{p} \left| d_j \right|}{p} \quad \ldots(89),$$

donde d_j es la distancia *j*-ésima y
p es el número de patrones en la base de datos

El DDC sí nos permitiría cuantificar la distancia entre la variable de salida y el valor de decisión (umbral), y al ser calculada en base a una variable continua (*d*), es válido pensar en la desviación estándar del DDC (std(DDC)) como otro indicador (válido) de eficiencia.

Estos dos indicadores de eficiencia (DDC y std(DDC)) serían buenas alternativas frente a los problemas que presentaban el MSE y, por lo tanto, std(MSE).

4. Inconvenientes de la parada temprana

El criterio de parada temprana es un criterio de generalización rápido, simple y fácil de aplicar (Sarle, 2002). Sin embargo, ofrece algunas desventajas:

Primero, es difícil saber qué tamaños deben tener las bases para el entrenamiento, la validación y el test. En respuesta a este problema, para nuestro caso, hemos encontrado "regiones de eficiencia", que proporcionan pistas para la elección de los tamaños.

Y segundo, el criterio de parada es la aparición de una tendencia creciente en la función del error para la validación. Es decir, cuando para un número específico de iteraciones en error de la validación continúe en aumento, la red parará. Esto presenta dos inconvenientes: el primero, que no siempre en el momento de la detención la red ha aprendido lo suficiente de la data de entrenamiento, pues se le da prioridad a la validación en pro de la generalización (por ello su nombre es "parada temprana"); y el segundo, que no hay criterios muy concluyentes para elegir el número de iteraciones para las que el aumento continuo del error de validación nos indique un riesgo de sobre-ajuste y por lo que interrupción del entrenamiento sea necesaria. Para las pruebas de sensibilidad y el entrenamiento de la red hemos usado un valor de 20, cuando el valor automático de la red es 5. Este cambio fue hecho arbitrariamente, aumentando el valor para darle un mayor tiempo de aprendizaje a la red y asegurarnos de que la tendencia creciente del error en la validación sea realmente un indicador de riesgo de sobre-ajuste (puesto que la teoría señala que es normal que la función del error presente indistintamente subidas y bajadas durante el tiempo de entrenamiento). No obstante, Sarle (2002) sugiere que una aproximación más segura sería desarrollar un algoritmo que entrene hasta la convergencia (de la base para el entrenamiento), luego vuelva sobre ese entrenamiento y revise en qué iteración la red experimentó el error de validación más bajo, y así para la predicción, utilice la red entrenada solo hasta ese punto. Prechelt (1997) utiliza este error de validación más bajo para construir un indicador de pérdida de generalización y fijar una condición de "parada temprana" con respecto a éste. También propone un segundo criterio haciendo un ratio entre éste y un indicador de "progreso" (del entrenamiento) de manera que se tome en cuenta no sólo la pérdida de generalización analizada desde la validación sino también las mejoras en el aprendizaje que va teniendo la red. Él propone escoger uno de estos criterios dependiendo de las prioridades que se tengan: tiempo, robustez, eficiencia, eficacia, etc. En resumen, no hay una respuesta definitiva concerniente a la manera de realizar la parada temprana, mas sí diversos criterios para la elección.

5. ¿Qué base temporal utilizar y cuándo pensar en cambiar de base de entrenamiento?

La pequeña diferencia entre los resultados de las bases bianuales y trianuales y la necesidad de identificación de calas de bases de datos de fechas más recientes (predicción), hace aconsejable el uso de la base bianual 2005-2006 para el entrenamiento, por ser la más actual y a cuyas características más se parecerán las de las nuevas bases. Pero, ¿cuándo sería un momento adecuado para cambiar de base de entrenamiento por una más actual? Dos criterios parecen los más adecuados:

- **Cuando la frecuencia de emisión en el VMS cambie.** Como se explicó anteriormente, la variación en la duración entre 2 emisiones afecta también a las variables de entrada de la red (por sus formas de cálculo). Si en un futuro la frecuencia de emisión es menor, las relaciones entre las variables de entrada que lleven a una identificación de calas serán diferentes a las de la base 2005-2006 (con la que se entrenaría). Por lo tanto, se debería volver a entrenar con una base temporal que tenga características similares a aquella para la que se realizará la predicción.

- **Cuando la composición de la flota cambie.** La flota pesquera está compuesta por 3 tipos de embarcaciones: 1) las "vikingas" (5% del total de la flota), con una capacidad de bodega inferior a 100 m^3 y un pobre equipamiento electrónico; 2) las "tradicionales" (77%), con capacidades de bodega entre 100 y 400 m^3, y mayor equipamiento tecnológico que las "vikingas" y 3) las embarcaciones equipadas con sistemas de refrigeración de agua de mar (RSW), (18%), con capacidades de bodega superiores a 400 m^3 (Díaz, 2005). Hay características de pesca, como el tiempo que demoran en calar, por ejemplo, que pueden influir en las variables de entrada de la red, de tal manera que para un tipo de nave las relaciones entre las variables de entrada para la identificación de una cala sean diferentes

que para los otros tipos de naves; es decir, que existan aprendizajes distintos o redes distintas para cada tipo de embarcación.

Desde el 2003, las "vikingas" estás obligadas a portar transmisores satelitales y utilizar el VMS. Quizás la inclusión de los datos del VMS pertenecientes a las "vikingas" ha proporcionado mayor variabilidad a la base de datos –y por lo tanto al entrenamiento de la red –desde el año de su incorporación. No hemos testeado la sensibilidad de la red a la composición de la flota, pero creemos que hacerlo podría proporcionarnos mayores conocimientos sobre su efecto en el aprendizaje de la red y por consiguiente, en su desempeño (traducido acaso en errores de predicción). Si su efecto es importante, también se debe tomar en cuenta este componente al considerar la base de entrenamiento.

6. ¿Por qué no utilizar técnicas estadísticas clásicas como la regresión logística o el análisis discriminante?

El análisis discriminante y la regresión logística son las técnicas más comunes para la clasificación de patrones en categorías, como "cala" o "no cala".

El primero, en la mayoría de paquetes estadísticos y en la mayoría de los casos, se trabaja bajo el supuesto de una función de densidad normal multivariada para las variables discriminatorias. Ésta no es una condición estrictamente necesaria, pero sí se necesita conocer la función de densidad conjunta de dichas variables. Esto no sucede en nuestro caso.

Además, se pensó –desde un principio –en una red neuronal porque el objetivo era de naturaleza predictiva y no explicativa. Es decir, se quería reconocer a través de los patrones de datos cuáles pertenecían a o indicaban una cala, mas no era importante saber qué combinaciones de variables –ni qué forma de combinarlas –producían una buena predicción de cala. No era ineluctable conocer ni definir un modelo, sino la identificación misma, por lo que la regresión logística, como modelo de probabilidad definido –y por lo tanto explicativo y predictivo -, era innecesaria.

Sin embargo, Bertrand *et al.* (2008) utilizaron una red neuronal sin neuronas en la capa escondida y con función de activación logística para la base 2000-2002 entre 7ºS y 10ºS, comparando sus resultados con los de una red de más neuronas. Esta red es muy similar a una regresión logística, diferenciándose ambas en los algoritmos utilizados para el entrenamiento en la primera y la estimación de parámetros en la segunda. Los resultados de la red sin capa escondida tuvieron un 65% de buena clasificación. Además, para efectos de la actual discusión, se aplicó un modelo de regresión logística en el SPSS 12.0 utilizando los métodos de selección de variables: introducir, selección hacia adelante (estadístico de Wald) y eliminación hacia atrás (estadístico de Wald); y se encontró que aquel no cumplía con el supuesto de homocedasticidad ni de distribución binomial del error, para ninguna de las bases temporales; y rechazaba la hipótesis del ajuste del modelo mediante el test de Hosmer-Lemeshow para las bases 2004-2006 y 2000-2006 –Hosmer *et al.*, 1997 señalan algunas debilidades de dicho test, especialmente cuando la muestra es pequeña, lo cual no sucede en nuestro caso; aún así debemos interpretar sus resultados con cautela. Además resultó en subestimaciones de las calas: en ningún caso superaron el 50% de identificación correcta de calas, y el porcentaje de calas identificadas (entre verdaderas y falsas) estaba entre un 60% y 65% del total de calas observadas (anexo 6). Es conveniente señalar que en la regresión logística, si bien existe la posibilidad de indicar cuál es el umbral del modelo para la clasificación –aunque por defecto se suele utilizar 0.5 –no hay forma de indicar un criterio para el umbral como "identificar un número de calas igual al número real", lo que sí se puede hacer con el algoritmo de la red neuronal.

Por otro lado, el umbral hallado para las diferentes bases temporales a través de la ANN, ha sido siempre mayor a 0.5, por lo que la red ha sido más estricta a la hora de reconocer una cala. Si estos umbrales se hubieran aplicado en la regresión logística, habría sido más difícil identificar calas, por lo que los porcentajes de calas identificadas y de identificación correcta habrían mermado también.

CAPÍTULO VI
CONCLUSIONES

La red neuronal más adecuada es aquella que utiliza una base de entrenamiento bianual 2005-2006, pues es el período más cercano a las nuevas bases de predicción que surgirán más adelante. Con 40 bucles, 3 nodos en la capa escondida, y un umbral para cuyo valor el número de calas identificadas sea igual al de observadas esta red obtendrá aproximadamente un 81% de identificaciones de calas correctas, convirtiéndose ahora en el mejor método de predicción disponible para ello, al brindar mejores resultados que otras técnicas –actualmente se utiliza en IMARPE sólo el criterio de velocidad –y cumplir eficientemente con los propósitos de identificación. Para llegar a esta afirmación, ha sido necesario conocer las fortalezas y limitaciones de la red comprendidas igualmente en estadística (eficacia y potencia frente a otras técnicas, medida de variabilidad en los resultados, sensibilidad frente a ciertas asunciones como los valores de los parámetros, a la base de datos a entrenar, etc.) como en ciencia y coyuntura pesqueras (en torno a las variables que se toman en cuenta para la red, la pertinencia de hacer un análisis por zonas o para todo el mar peruano, apreciaciones juiciosas para la elección y cambio de base temporal para el entrenamiento, etc.). Como es de esperarse, la coyuntura pesquera cambiará y los conocimientos en pesquería y estadística se actualizarán, de manera que es necesario estar alertas para decidir cuándo estos cambios deberán implicar también un cambio en la estructura de la red o incluso una opción por una técnica mejor que las redes neuronales artificiales.

Por otro lado, es importante destacar que la posibilidad de identificar puntos de cala a partir de datos del VMS le otorga a esta herramienta un valor agregado, pues no sólo proporciona datos para el monitoreo de los barcos de acuerdo a su localización, sino que a través de los datos de coordenadas y del momento de la emisión es posible calcular otras variables como hora, velocidad, cambio de velocidad y cambio de rumbo que relacionadas en la red neuronal, nos pueden dar información confiable sobre la ubicación de las calas.

Se ha desarrollado también una interfaz gráfica en el Matlab para que el uso de la red neuronal sea más sencillo y por lo tanto factible de utilizar para el monitoreo diario. La mayoría de parámetros de la red permanecen fijos, y solo el valor del umbral y el MSE cambian de acuerdo a criterios explicitados al programa. El o la usuario solo debe escoger la base de entrenamiento bianual a utilizar (probablemente la más próxima en tiempo a la de predicción) y cargar la base de datos para la predicción. La interfaz identifica los puntos de cala, los guarda en una matriz y los dibuja en el mapa (anexo 7).

Con ello, la comparación de la información de la distribución espacial de las calas con aquella del recurso mismo (el método más utilizado para su estimación es la hidroacústica) será posible y nos permitirá obtener medidas de vulnerabilidad de la especie frente a la actividad pesquera. Hoy hemos utilizado las ANN para la identificación de calas de anchoveta pero eso no impide –al contrario, invita –a hacerlo también para otras especies de interés. De igual manera, la relación de estas con la distribución espacial de aves y/u otros predadores nos dará acceso a una mayor comprensión del rol –y efectos –de la pesquería en el ecosistema marino.

Mientras más herramientas tecnológicas y científicas poseamos –siendo conscientes sus alcances y limitaciones –mayores horizontes tendremos para realizar un diagnóstico adecuado de la pesquería y del estado del ambiente marino. La información amplia pero al mismo tiempo veraz y precisa, permitirá a IMARPE cumplir con su finalidad de asesorar al Gobierno Peruano en las decisiones sobre políticas públicas relacionadas al ordenamiento pesquero, que tomen en consideración tanto las implicancias económicas, sociales y ecológicas a corto, mediano y largo plazo. Si bien las decisiones últimas dependen de las autoridades gobernantes, es el papel de los científicos y de un ente investigador como IMARPE generar el conocimiento que otorgue las facultades para tomar este tipo de decisiones de la mejor manera posible.

BIBLIOGRAFÍA

Acuña, E. (2004). *Análisis de regresión*. Departamento de Matemáticas, Universidad de Puerto Rico. 181-205.

Aldabas-Rubira, E. (2002). Introducción al reconocimiento de patrones mediante redes neuronales. *IX Jornades de Conferències d'Enginyeria Electrònica del Campus de Terrassa, Terrassa, España, del 9 al 16 de Diciembre del 2002.*

Alheit, J. and Ñiquen, M. (2004). Regime shifts in the Humboldt Current ecosystem. *Progress in Oceanography* 60: 201-222.

Arreguín-Sánchez, F. (1996). Catchability: a key parameter for fish stock assesment. *Reviews in Fish Biology and Fisheries* 6: 221-242.

Babcock, E. A., Pikitch, E. K., K., M. M., Apostolaki, P. and Santora, C. (2005). A perspective on the use of spatialized indicators for ecosystem-based fishery management through spatial zoning. *ICES Journal of Marine Science* 62: 469-476.

Ballón M., Lebourges-Dhaussy A., Gutierrez M., Peraltilla S. and Bertrand A. 2008. Acoustic study of spatiotemporal distribution of zooplankton biomass off Peru provides new insights into Humboldt Current system productivity. *Symposium on the Ecosystem Approach with Fisheries Acoustics and Complementary Technologies*, *Bergen, Noruega, del 16 al 20 de Junio del 2008.*

Baruah, P., Tamura, M. and Oki, K. (2001). Neural network modeling of lake surface chlorophyll and sediment content from Landsat TM imagery. *22nd Asian Conference on Remote Sensing. Singapore, del 5 al 9 de Noviembre del 2001.*

Bertrand, A., Segura, M., Gutiérrez, M. and VÁsquez, L. (2004a). From small-scale habitat loopholes to decadal cycles: a habitat-based hypothesis explaining fluctuation in pelagic fish populations off Peru. *Fish and Fisheries* 5: 296-316.

Bertrand, S. (2005). *Analyse comparée des dynamiques spatiales des poissons et des pêcheurs: mouvements et distributions dans la pêcherie d'anchois (Engraulis ringens) du Pérou*, Ecole Nationale Supérieure Agronomique de Rennes. Docteur de l'ENSAR. Mention : Halieutique.

Bertrand, S., Bertrand, A., Guevara-Carrasco, R. and Gerlotto, F. (2007). Scale-invariant movements of fishermen: the same foraging strategy as natural predators. *Ecological Applications* 17(2): 331–337.

Bertrand, S., Burgos, J. M., Gerlotto, F. and Atiquipa, J. (2005). Lévy trajectories of peruvian purse-seiners as an indicator of the spatial distribution of anchovy (*Engraulis ringens*). *ICES Journal of Marine Science* 62(3): 477-482.

Bertrand, S., Diaz, E. and Lengaigne, M. (2008). Patterns in the spatial distribution of peruvian anchovy (*Engraulis ringens*) revealed by spatially explicit data. *Progress in Oceanography* 00(00): 00-00.

Bertrand, S., Díaz, E. and Ñiquen, M. (2004b). Interactions between fish and fisher's spatial distribution and behaviour: an empirical study of the anchovy (*Engraulis ringens*) Fishery of Peru. *ICES Journal of Marine Science* 61(7): 1127 -1136.

Bishop, C. M. (1995). *Neural networks for pattern recognition*, Oxford University Press.482 pp.

Bouchon, M., Cahuín, S., Diaz, E. and Ñiquen, M. (2000). Captura y esfuerzo pesquero de la pesquería de anchoveta peruana (*Engraulis ringens*). *Boletín. Instituto del Mar del Perú*. Lima, Perú. 19**:** 109 - 116.

Bouchon, M., Ñiquen, M., Arias-Schreiber, M. and Bello, R. (1998). Informe progresivo: manual de operaciones del proyecto bitácoras de pesca. Instituto del Mar del Perú.

Brey, T., Jarre-Teichmann, A. and Borlich, O. (1996). Artificial neural networks versus multiple linear regression: predicting P/B ratios from empirical data. *Marine Ecology Progress Series* 140: 251-256.

Bueche, D. (2005). Knowledge based systems - Exercises. http://www.openopal.org/members/dirk_bueche/teaching/wbs/content/2005.04.13/wbs_exercise_2005.04.13.pdf

Campos Barroso, D. M. (2007). Noções de indicadores de desempenho. Programa de Qualidade na Gestão Pública do Estado do Pará. www.pqg.pa.gov.br/docs/indicadores.pps

Caviedes, C. N. (1975). El Nino 1972: Its climatic, ecological, human, and economic implications. *Geographical Review* 65(4): 493-509.

Chan, Z. S. H., Ngan, H. W. and Rad, A. B. (2003). Improving bayesian regularization of ANN via pre-training with early-stopping. *Neural Processing Letters* 18(1): 29–34.

Chen, D. G. and Ware, D. M. (1999). A neural network model for forecasting fish stock recruitment. *Journal Canadian des sciences halieutiques et aquatiques* 56(12): 2385-2396.

Ciarlini, P., Maniscalco, U. and Regoliosi, G. (2005). Validation of soft sensors for monitoring ambient parameters. *VII Conference on Advanced Mathematical and Computational Tools in*

metrology, Caparica, Portugal, del 27 al 29 de Junio del 2005.
CPPS (2001). Informes nacionales sobre ecuaciones de fuerza de blanco calculadas o tomadas de otras áreas y aplicadas a evaluaciones hidroacústicas de recursos pesqueros. *Segundo Taller Regional sobre Metodologías de Evaluación Hidroacústica de Recursos Pesqueros, del 4 al 7 de Diciembre del 2000.* Comisión Permanente del Pacífico Sur. Callao, Perú**:** 28-30.
Csirke, J. (1989). Changes in the catchability coefficient in the Peruvian anchoveta (*Engraulis ringens*) fishery. Paper presentado en la conferencia "The Peruvian upwelling ecosystem: dynamics and interactions", IMARPE, Callao, Peru, 1989.
Csirke, J. (1990). El uso de datos de esfuerzo y captura por unidad de esfuerzo en la investigación de recursos pelágicos en el Instituto del Mar del Peru (IMARPE). *Informe de la misión de asesoramiento realizada por la CEE a través del Programa Regional de Cooperación Técnica para la Pesca (CEE-PEC).* Lima, Perú.
Cybenko, G. (1989). Approximation by superpositions of a sigmoidal function. *Mathematics of Control, Signals and Systems* 2: 303-314.
De Campos, A. M. (2006). Soluções para o problema do caixeiro viajante usando uma rede neuronal (Hopfield Net). http://to-campos.planetaclix.pt/ind.htm.
Demuth, H. and Beale, M. (1998, 1998). *Neural network toolbox for use with Matlab*. 742 pp. From http://www.mathworks.com
Deng, R., Dichmont, C., Milton, D., Haywood, M., Vance, D., Hall, N. and Die, D. (2005). Can vessel monitoring system data also be used to study trawling intensity and population depletion? The example of Australia's Northern Prawn fishery. *Canadian Journal of Fisheries and Aquatic Sciences* 62(3): 611-622.
Diaz, E. (2005). *Eficiencia de la flota de cerco en función a la distribución espacial de la anchoveta peruana (Engraulis ringens) entre 1998 y el 2001.* Lima, Perú, Universidad Nacional Mayor de San Marcos. Bachiller en Biología con Mención en Biología Pesquera
Doan, S. D. and Liong, S. Y. (2004). Generalization for multilayer neural network bayesian regularization or early stopping. *The 2nd APHW Conference (APHW2004)* Suntec International Convention and Exhibition Center, Singapore, del 5 al 8 de Julio del 2004.
Dobson, A. (2002). *An introduction to generalized linear models*, Chapman & Hall/CRC.
Dreyfus-León, M. (1999). Individual-based modelling of fishermen search behaviour with neural networks and reinforcement learning. *Ecological Modelling* 120(2-3): 287–297.
Drumm, D., Purvis, M. and Zhou, Q. (1999). Spatial ecology and artificial neural networks: modeling the habitat preference of the sea cucumber (*Holothuria leucospilota*) on Rarotonga, Cook Islands. *The 11th Annual Colloquium of the Spatial Information Research Centre (SIRC 99).* Dunedin, New Zealand, del 13 al 15 de Diciembre de 1999.
Escobar, J. (2001). *El aporte del enfoque ecosistémico a la sostenibilidad pesquera.* Santiago de Chile, División de Recursos Naturales e Infraestructura, CEPAL. 57 pp.
FAO, Departamento de Pesca (1995). *Código de conducta para la pesca responsable.* Roma, FAO.
FAO, Departamento de Pesca. (1999). FAO Orientaciones técnicas para la pesca responsable. *La Ordenación Pesquera*. Roma, FAO.
Feng, J., Ed. (2004). *Computacional neuroscience: a comprehensive approach.* Mathematical Biology and Medicine Series, Chapman & Hall/CRC. 640 pp.
Fréon, P. and Misund, O. A. (1999). *Dynamics of pelagic fish distribution and behaviour: effects on fisheries and stock assessment.* Oxford. 348 pp.
Gaertner, D. and Dreyfus-Leon, M. (2004). Analysis of non-linear relationships between catch per unit effort and abundance in a tuna purse-seine fishery simulated with artificial neural networks. *ICES Journal of Marine Science* 61(5): 812-820.
García de Jalón, J., Rodríguez, J. I. and Brazález, A. (2001). *Aprenda Matlab como si estuviera en primero*, Escuela Técnica Superior de Ingenieros Industriales. Universidad Politécnica de Madrid. 113 pp.
Garcia, S. M., Zerbi, A., Aliaume, C., Do Chi, T. and Lasserre, G. (2003). The ecosystem approach to fisheries. Issues, terminology, principles, institutional foundations, implementation and outlook. *FAO Fisheries Technical Paper*, Rome, FAO.
Gerlotto, F. (2001a). Geoestadística: aplicación y proyecciones en hidroacústica pesquera. *Segundo Taller Regional sobre Metodologías de Evaluación Hidroacústica de Recursos Pesqueros*, Callao, Perú, del 4 al 7 de Diciembre del 2000: 16-21.
Gerlotto, F. (2001b). Problemas en la aplicación de métodos acústicos y posibles soluciones. *Segundo Taller Regional sobre Metodologías de Evaluación Hidroacústica de Recursos Pesqueros*, Callao, Perú, del 4 al 7 de Diciembre del 2000: 68-71.
German, S., Bienenstock, E. and Doursat, R. (1992). Neural networks and the bias/variance dilemma.

Neural Computation 4(1): 1-58.
Goethals, P., Dedecker, A., Gabriels, W., Lek, S. and De Pauw, N. (2007). Applications of artificial neural networks predicting macroinvertebrates in freshwaters. *Aquatic Ecology* 41: 491–508.
Guerrero, V. and de Moya, F. (2001). Reduction of the dimension of a document space using the fuzzified output of a Kohonen network. *Journal of The American Society for Information Science and Technology* 52(14): 1234-1241.
Gutiérrez, M. (2000a). Estimados de biomasa hidroacústica de los cuatro principales recursos pelágicos en el mar peruano durante 1983-2000. *Boletín Instituto del Mar del Perú*. Callao, Perú. 19 (1-2)**:** 139-156.
Gutiérrez, M. (2000b). Variaciones estacionales en la distribución y biomasa de anchoveta entre 1983 y 2000. *Boletín Instituto del Mar del Perú*. Callao, Perú. 19 (1-2)**:** 157-177.
Hagan, M. T., Demuth, H. B. and Beale, M. (1996). *Neural network design*, PWS Publishing Company. 734 pp.
Harley, S. J., Myers, R. A., and Dunn, A. (2001). Is catch-per-unit- effort proportionnal to abundance? *Canadian Journal of Fisheries and Aquatic Sciences* 58: 1760-1772.
Hastie, T. and Tibshirani, R. (1986). Generalized additive models. *Statistical Science* 1(3): 297-310.
Hastie, T. and Tibshirani, R. (1995a). Generalized additive models. *Encyclopaedia for Biostatistics*. 10 pp.
Hastie, T. and Tibshirani, R. (1995b). Generalized Additive Models. *Encyclopaedia of Statistical Sciences*. 14 pp.
He, H. and Sýkora, O. (2006). A Hopfield neural network model for the outerplanar drawing problem. *IAENG International Journal of Computer Science* 34(4): 42-47.
Holden, M. J. and Raitt, D. F. S. (1975). Manual de ciencia pesquera. Parte 2: Métodos para investigar los recursos y su aplicación. Roma, FAO.
Hosmer, D. W., Hosmer, T., Le Cessie, S. and Lemeshow, S. (1997). A comparison of goodness-of-fit tests for the logistic regression model. S*tatistics in Medicine* 16: 965-980.
Jain, A. and Mao, J. (1996). Artificial neural networks: a tutorial. *Computer* 29(3): 31 - 44.
Jayawardena, A. W., Fernando, D. A. K. and Zhou, M. C. (1997). Comparison of multilayer perceptron and radial basis function networks as tools for flood forecasting. *International conference on destructive water*, Anaheim CA, del 24 al 28 de Junio de 1996.
Jordan, M. (1995). Why the logistic function? A tutorial discussion on probabilities and neural networks. Computational Cognitive Science Technical Report. Massachusetts, *Massachusetts Institute of Technology*.
Klinke, S. and Grassmann, J. (1998). *Projection pursuit regression and neural networks*. Humboldt-University of Berlin. 47 pp. http://edoc.hu-berlin.de/series/sfb-373-papers/1998-17/PDF/17.pdf
Laurec, A. and Le Guen, J.-C. (1981). *Dynamique des populations marines exploitées. Tome I Concepts et Modèles.* Rapports scientifiques et techniques nº 45, Centre national pour l'exploitation des oceans. 118 pp.
Lek, S. and Guégan, J. F. (2000). *Artificial neuronal networks: application to ecology and evolution.* Berlin, Springer-Verlag. 262 pp.
Lendasse, A., Lee, J., De Bodt, E., Wertz, V. and Verleysen, M. (2003). Approximation by radial basis function networks: application to option pricing. *Connectionist Approaches in Economics and Management Sciences*. M. C. e. C. Lesage. Louvain-la-Neuve, Kluwer academic publishers**:** 203-214.
Mackay, D. J. C. (1995). Probable networks and plausible predictions - a review of practical bayesian methods for supervised neural networks. *Network* 6(3): 469-505.
MacLennan, D. N. and Holliday, D. V. (1998). Acústica pesquera y del plancton: pasado, presente y futuro. *Informe Instituto del Mar del Perú* 133**:** 10-14.
Matheron, G. (1978). *Estimer et choisir.* Les Cahiers Du Centre De Morphologie Mathématique De Fontainebleau. 175 pp.
Medeiros, M. C. (2001). Statistical methods for modelling neural networks. *Intelligent Systems* 9: 227-235.
Megrey, B., Lee, Y. and Macklin, S. A. (2005). Comparative analysis of statistical tools to identify recruitment-environment relationships and forecast recruitment strength. *ICES Journal of Marine Science* 62: pp. 1256-1269.
Mills, C. M., Townsend, S. E., Jennings, S., Eastwood, P. D. and Houghton, C. A. (2007). Estimating high resolution trawl fishing effort from satellite-based vessel monitoring system data. *ICES Journal of Marine Science* 64: 248 - 255.
Ñiquen, M., Bouchon, M., Cahuín, S. and Diaz, E. (2000). Trabajo presentado al taller internacional anchoveta peruana: Pesquería de anchoveta en la costa peruana. *Boletín IMARPE*. Lima,

Perú. 19**:** 117 - 124.
Olden, J. and Jackson, D. (2001). Fish–habitat relationships in lakes: Gaining predictive and explanatory insight by using artificial neural networks. *Transactions of the American Fisheries Society* 130: 878–897.
Park, Y., Céréghino, R., Compin, A. and Lek, S. (2003). Applications of artificial neural networks for patterning and predicting aquatic insect species richness in running waters. *Ecological Modelling* 160: 265-280.
Pauly, D., Palomares, M. L. and Gayanilo, F. C. (1987). VPA Estimates of the monthly population length composition, recruitment, mortality, biomass and related statistics of peruvian anchoveta, 1953 to 1981. *The Peruvian Anchoveta and Its Upwelling Ecosystem: Three Decades of Change.* D. Pauly and I. Tsukayama. *ICLARM Studies and Reviews* 15: 142-166.
Prechelt, L. (1998). Early stopping - but when? *Neural networks: tricks of the trade*. Heidelberg, Springer Verlag**:** 55-69.
Puntonet, C. G., Mansour, A., Alvarez, M. R., Prieto, B. and Rojas, I. (2001). A stochastic and competitive network for the separation of sources. *ESANN'2001 - European Symposium on Artificial Neural Networks*, Bruges (Belgium), del 25 al 27 de Abril del 2001, D-Facto public.
Ripley, B. D. (1995). Choosing network complexity. *Probabilistic Reasoning and Bayesian Belief Networks*. A. W. A. Gammerman**:** 97-108.
Ritthipravat, P. and Nakayama, K. (2002). Obstacle avoidance by using modified hopfield neural network. *International Conferencia on Artificial Intelligence, IC-AI'02*, Las Vegas, Nevada, USA, del 24 al 26 de Junio del 2002, CSREA Press.
Rodriguez, P., Wiles, J. and Elman, J. L. (1999). A recurrent neural network that learns to count. *Connection Science* 11(1): 5-40.
Rose, G. A. and Kulka, D. W. (1999). Hyperaggregation of fish and fisheries: how catch-per-unit-effort increased as the Northern Cod (*Gadus morhua*) declined. *Canadian Journal of Fisheries and Aquatic Sciences* 56(S1): 118–127.
Sahin, F. (1997). *A radial basis function approach to a color image classification problem in a real time industrial application.* Electrical Engineering. Virginia, Virginia Tech. Master of Science. http://scholar.lib.vt.edu/theses/public/etd-6197-223641/etd-title.html
Sampson, D. B. (1988). The stability of virtual population analysis cohort size estimates. *Journal du Conseil International pour l'Exploration de la Mer* 44: 135-142.
Sarle, W. (1995). Stopped training and other remedies for overfitting, SAS Institute Inc. ftp://ftp.sas.com/pub/neural/inter95.ps.Z
Sarle, W. (2002). AI FAQ/Neural nets. http://www.faqs.org/faqs/ai-faq/neural-nets/.
Sarle, W. S. (1996). Neural network and statistical jargon. ftp://ftp.sas.com/pub/neural/jargon
Sarle, W. S. (1999). Ill-Conditioning in Neural Networks. ftp://ftp.sas.com/pub/neural/illcond/illcond.html
Scardi, M. (2000). Neural Network Models of Phytoplankton Primary Production. *Artificial Neuronal Networks: Application to Ecology and Evolution*, Springer**:** pp. 115-129.
Simmonds J., Gutierrez M., Chipolini A., Gerlotto F. and Bertrand A. 2008. Acoustic surveys used for in year fisheries management of Peruvian anchoveta. *Symposium on the Ecosystem Approach with Fisheries Acoustics and Complementary Technologies, Bergen, Norway, del 16 al 20 de Junio del 2008* (Poster)
Simmonds J. y MacLennan D. (2005, 1st edition in 1991) *Fisheries acoustics: Theory and practise*. Blackwell. 437 pp.
Smetherman, B. B. and Smetherman, R. M. (1973). Peruvian fisheries: conservation and development. *Economic Development and Cultural Change* 21(2): 338-351.
Suryanarayana, I., Braibanti, A., Sambasiva Rao, R., Anantha Ramam, V., Sudarsan, D. and Nageswara Rao, G. (2008). Neural networks in fisheries research. *Fisheries Research* 92: 115-139.
Tilic, I. (1963). Información estadística sobre embarcaciones utilizadas en la pesca industrial en el Perú 1953-1962. *Informe del Instituto de Investigación de los Recursos Humanos.* Lima, Perú. 8**:** 36-65.
Tusell, F. (2005). *Análisis multivariante*. Notas de clase. 167 pp. http://www.et.bs.ehu.es/~etptupaf/nuevo/ficheros/estad4/multi.pdf
Venkatesan, P. and Anitha, S. (2006). Application of a radial basis function neural network for diagnosis of Diabetes Mellitus. *Current Science* 91(9): 1195 - 1199.
Vijayakumar, S. The bias-variance tradeoff, School of Informatics, Univ. of Edinburgh. http://www.inf.ed.ac.uk/teaching/courses/mlsc/Notes/Lecture4/BiasVariance.pdf
Warner, B. and Misra, M. (1996). Understanding neural networks as statistical tools. *The American*

Statiscian 50(4): 284-293.
Witt, M. J., and B. J. Godley. 2007. A Step Towards Seascape Scale Conservation: Using Vessel Monitoring Systems (VMS) to Map Fishing Activity. *PLoS* 10:5pp.
Xiong, J., Rayner, S., Luo, K., Li, Y. and Chen, S. (2006). Genome wide prediction of protein function via a generic knowledge discovery approach based on evidence integration. *BMC Bioinformatics* 7.
Zuta, S., Enfield, D., Valdivia, J. and Lagos, P. (1976). Aspectos físicos del fenómeno "El Niño" 1972-1973. *FAO. Informes de pesca.* 185**:** 3 - 61.

ANEXOS

Anexo 1:

Sintaxis en Matlab

1.1. Sintaxis de entrenamiento y predicción de la red

Es el programa para entrenar la red y predecir en una nueva base de datos. Es la última versión y la que se propone para el uso de la red neuronal. Los comandos están en color negro y sus respectivas explicaciones, en azul.

```
function
resultado=training7(dia,mes,ano,unaBaseEntrenamiento,AnoInicio,AnoFinal,unaBasePrediccion)
```

El nombre de la función es training7. Sus parámetros de entrada son:

- Día, mes y ano: es la fecha de ejecución de la interfaz de predicción.
- unaBaseEntrenamiento: es el nombre del archivo de la base de entrenamiento
- AnoInicio y AnoFinal: son los años para los que se entrenará la base
- unaBasePrediccion: es el nombre del archivo de la base de predicción.

Resultado es la variable de salida de la función.

```
warning off;
```

hace que las advertencias del Matlab no se muestren en la ventana de comandos.

```
inicio = AnoInicio;
final = AnoFinal;
d=dia;
m=mes;
a=ano;
BaseEntrenamiento = unaBaseEntrenamiento;
BasePrediccion = unaBasePrediccion;

ok=0;
```

es el valor inicial de ok

Especifico una ruta para buscar archivos que tengan como nombre indicadores… y los años de entrenamiento. Si algún archivo tiene ese nombre, ok tomará el valor de 1.

```
ruta = 'C:\ROCÍO\trabajo\gui\ArchivosSalida';
listArchivos = dir(ruta);
nombre=strcat('indicadores_',num2str(inicio),'_',num2str(final),'.txt');
for i=1:numel(listArchivos)
      if (strcmp(listArchivos(i).name,nombre)==1)
ok=1;
      end
end
```

Si ok=0, quiere decir que no existe base de entrenamiento para ese período de tiempo y por lo tanto habrá que entrenar una.

```
if ok==0
   entrenamiento(BaseEntrenamiento,inicio,final)
```

(la función entrenamiento se explicará a continuación)

```
end
```

Luego se hace la predicción

```
prediccion(d,m,a,inicio,final,BasePrediccion)
```

después se explicará también la función predicción

```
resultado=0;
```

porque en verdad no nos interesa enseñar ningún resultado en la ventana de comandos; los resultados que nos interesaban los hemos guardado en archivos (como se verá más adelante).

Entrenamiento

Asumiendo que se entrenarán redes bianuales el programa fija los parámetros número de nodos, de bucles y tamaño de la base, encuentra el valor del umbral bajo el criterio de número de calas observadas igual al de calas identificadas (fijando el MSE máximo en 0.03), y prueba con dos valores de MSE máximo: 0.02 y 0.03, quedándose con el que le proporciona mayor porcentaje de calas verdaderas. La base de datos contiene datos de muchos años, y lo que hará el usuario es seleccionar los años de cuyos datos se entrenará las bases (recordemos que son bases bianuales).

function resultado=entrenamiento(unaBaseEntrenamiento,AnoInicio,AnoFinal)
El nombre de la función es entrenamiento. Sus parámetros de entrada son:

- unaBaseEntrenamiento: es el nombre del archivo de la base de entrenamiento
- AnoInicio y AnoFinal: son los años para los que se entrenará la base

Resultado es la variable de salida de la función.

warning off;

Lee el nombre del archivo de la base de entrenamiento:
BaseEntrenamiento = unaBaseEntrenamiento;

tiempo=cputime; (servirá para controlar cuánto demora el entrenamiento. Lo que hace es tomar el tiempo de trabajo del Matlab hasta antes este instante).

Luego, los parámetros de la red cuyos valores son resultado de los test de sensibilidad:

num_input=[10,11,12,13,14]; (esto es porque en la base de datos, las columnas que contienen a las variables de entrada son las columnas 10, 11, 12, 13 y 14)

nb_hidden=3;
Es el número de neuronas en la capa escondida.

nb_loop=40;
dlmwrite('./ArchivosSalida/nb_loop.txt',nb_loop,'\t');
Es el número de bucles y lo guarda en el archivo nb_loop, en la carpeta ArchivosSalida. Luego servirá para la predicción.
Con dlmwrite guardo el archivo que aparece en segundo orden dentro del paréntesis, con el nombre que se estipuló en primer orden, y con formato texto

kcross=5;
El kcross es usado para dividir la base de datos en las 3 partes. Como los resultados de sensibilidad han sido buenos para algunas proporciones de tamaño donde las bases para el test y la validación tienen los mismos tamaños, este algoritmo simplificado usa dicho resultado; en el cual el tamaño para el test y para la validación se calculan así: 1/kcross, mientras que para el entrenamiento, es el tamaño total de la base menos test y validación.

thresholdmin=50;
nthreshold=1;
Es el umbral con el que empezará a probar. nthreshold representa la variación en el valor del umbral cuando la red aún no encuentra un óptimo. El óptimo se dará cuando CO (calas observadas) sea mayor o igual a CI (calas identificadas). El umbral sirve para discriminar 'calas' de 'no calas' en la variable de salida.

MSE_max=0.03;
Es el MSE máximo aceptado para entrenar una red (reduciendo el riesgo de mínimo local)

Para importar la base se llama al archivo con load.

Importamos la base de entrenamiento desde la carpeta BaseEntrenamiento:
base=load(strcat('./BaseEntrenamiento/',BaseEntrenamiento));
base=base';

Lee las entradas de AnoInicio y AnoFinal y los convierte a numéricos:
```
inicio = str2double(AnoInicio);
final = str2double(AnoFinal);
```

Reconoce el año de inicio y el de fin para la base de entrenamiento
```
partida=find(base(6,:)==inicio);
llegada=find(base(6,:)==final);
ini=partida(1);
fin=llegada(end);
```
y selecciona solo los datos que pertenecen a ese rango de años
```
tab_orig=base(:,ini:fin);
```

Copiamos la matriz de la base original a una matriz llamada tab_mod, pues será una matriz que modificaremos.
```
tab_mod=tab_orig;
```

Para que la red no tenga problemas al interpretar las variables circulares de hora y cambio de rumbo (por ejemplo, directamente no es capaz de reconocer que una distancia entre 23:00 h y 02:00 h es mínima, lo que igual sucede con los grados sexagesimales), debe realizárseles una transformación.

```
tab_mod(10,:)=cos(tab_mod(10,:)*pi/12.);
```
Es la transformación realizada a la variable hora. Se escogió así porque como las calas se realizan entre la mañana y la tarde, en horas más próximas a la medianoche tendrá valores cercanos a 0 y como al mediodía, valores próximos a -1.
De ese modo con la variable transformada la red estará en mayores condiciones de discriminar.

```
tab_mod(14,:)=cos(tab_mod(14,:)*pi/180.);
```
Es la transformación realizada a la variable cambio de rumbo. Esta vez no se escogió la transformación por la distribución de frecuencias de la variable, sino por el criterio de que un mayor cambio de rumbo es un indicador de pesca y un menor cambio, de no pesca.
Entonces esta transformación coloca a los cambios mayores valores cercanos a 0 y a los cambios menores, valores cercanos a 1.

Para evitar problemas de escala, se decidió estandarizar las variables de manera que tengan media 0 y varianza 1, aproximadamente (estandarizar no es lo mismo que transformar a una normal estándar).
```
tab_mod(10:14,:)= prestd(tab_mod(10:14,:));
```

Con length contamos el número de observaciones (patrones) en la base de datos:
```
nb_data_mod=length(tab_mod);
```

Para dividir la base de datos en las 3 partes, calculamos primero el tamaño al que equivale la proporción de kcross:
```
size_seg=round(nb_data_mod/kcross);
```

```
threshold=thresholdmin;
j=1;
```
j será nuestro contador para los thresholds;

Ahora creamos matrices vacías (con elementos NaN) que luego utilizaremos para almacenar resultados. Es necesario crearlas porque tenerlas de antemano ahorra memoria en comparación a estar creándose y cambiando de dimensión bucle a bucle.
```
test_perf=NaN*ones(4,nb_loop);
train_perf=NaN*ones(4,nb_loop);
```

```
ttotal=0;
ttotnet=0;
```
son los valores iniciales de CO(ttotal) y CI(ttotnet);

```
while ttotal<=ttotnet
```

esto nos dice que los comandos dentro de éste while se correrán mientras que CI no sobrepase a CO;

ii=1;
ii será nuestro contador para el número de bucles
iii=1;
iii será nuestro contador para el número de veces en que se obtengan resultados de 0 calas identificadas para la base del test (y entonces los resultados de generalización sean pésimos). Este número de veces será iii-1 (porque nuestro valor inicial es 1).

while ii <= nb_loop
esto nos dice que los comandos dentro de éste while se correrán siempre y cuando no sobrepasemos el número de bucles definido como parámetro

tab_mod=tab_mod(:,randperm(nb_data_mod));
con randperm se está haciendo una permutación aleatoria de las columnas de la data, es decir, de los patrones, para luego dividirlas en las 3 bases.

target=tab_mod(9,:);
Guarda el vector objetivo que se encuentra en la columna 9 de la matriz tab_mod en un vector llamado target.

input=tab_mod(num_input,:);
Guarda la matriz de entrada (que contiene únicamente patrones de entrada para la red) que está formada por las columnas indicadas en num_input de la matriz tab_mod, en un vector llamado input.

Generando la base para el test;

Recordemos que el tamaño de la base de test es size_seg.
La base para el test tomará la primera parte de la base de datos total. Es decir, desde la observación 1 hasta la observación size_seg.

target_test=target(1,1:size_seg);
guarda en un vector target_test la primera parte del vector target.

input_test=input(:,1:size_seg);
guarda en una matriz input_test la primera parte de la matriz input.

t.P=input_test;
t.T=target_test;
ahora crea la estructura t constituida por el arreglo t.P (que contiene a la matriz input_test) y el arreglo t.T (que contiene al vector target_test).

Generando la base para la validación:

Recordemos que aquí también el tamaño de la base es size_seg.
La base para la validación tomará la segunda parte de la base de datos total. Es decir, desde la observación size_seg+1 hasta la observación 2*size_seg.

target_valid=target(1,size_seg+1:2*size_seg);
input_valid=input(:,size_seg+1:2*size_seg);
v.P=input_valid;
v.T=target_valid;
de la misma manera que para la base del test, crea una estructura con el vector de objetivo y la matriz de entrada para la validación.

Generando la base para el entrenamiento:

La base para la validación tomará la tercera y última parte de la base de datos total. Es decir, desde la observación 2*size_seg+1 hasta observación nb_data_mod.

target_train=target(1,2*size_seg+1:nb_data_mod);
input_train=input(:,2*size_seg+1:nb_data_mod);
Aquí crea un vector target_train con la tercera parte del vector target, y una matriz input_train con la tercera parte de la matriz input.

A continuación, se propone la red:

net=newff([minmax(input)],[nb_hidden,1],{'tansig','logsig'},'trainlm');
newff es un comando para crear una nueva red, del tipo MLN o SLN. Los parámetros que aquí se colocan son
minmax(input): la red pide como parámetros los valores mínimo y máximo de la entrada (con ello, el rango de entrada) por ello la función minmax halla esos 2 valores.
[nb_hidden,1]: indica que tendremos 3 nodos en una única capa escondida (1 denota el número de capas escondidas).
{'tansig','logsig'}: tansig o tangente hiperbólica sigmoidal es la función de activación para la capa escondida, y logsig o logística sigmoidal es la función de activación para la capa de salida.
'trainlm': es el algoritmo de aprendizaje que se utilizará para entrenar la red (trainlm es para Levenberg-Marquardt).

net.trainParam.show=NaN;
Normalmente, en la ventana de comandos se van mostrando resultados del entrenamiento cada cierto número de iteraciones. Este número es especificado por net.trainParam.show, que en nuestro caso es NaN, lo que significa que no queremos que nos muestre los resultados durante el proceso de aprendizaje.

net.trainParam.epochs=1000;
Número máximo de iteraciones que realizará la red durante el entrenamiento.

net.trainParam.max_fail=20;
Número máximo de veces en las que el MSE de la validación puede aumentar. Después de esto, la red se detiene. Es el parámetro del criterio de Parada Temprana.

Inicialización y entrenamiento de la red:

net=init(net);
Init «resetea» la red y los pesos y sesgos vuelven a sus valores iniciales (que por defecto son arreglos vacíos). Es para que, si una red anterior ha quedado almacenada, no se utilicen los últimos valores de los pesos (de la red anterior) para comenzar a entrenar esta red.

[net,tr]=train(net,input_train,target_train,[],[],v);
Aquí la red empieza a entrenar y por ello debemos indicar los elementos para este entrenamiento:
La red definida anteriormente con el nombre net.
La matriz de entradas para el entrenamiento: input_train.
El vector de objetivo para el entrenamiento: target_train.
[], [] están vacías porque nuestro aprendizaje es tipo lote y no secuencial.
Los arreglos de entrada y objetivo para la validación: definidos dentro de la estructura v.

Archivando la red:

name=strcat('./RedesEntrenadas/net1_',num2str(ii),'_',num2str(inicio),'_',num2str(final),'.mat');
save(name, 'net');
Se guarda la red con el número de bucle en el que va y los años entre los cuales se está efectuando el entrenamiento, en la carpeta RedesEntrenadas.

thres=threshold/100;
Esto porque el threshold está multiplicado por 100 (por complicaciones anteriores sobre números enteros se tuvo que hacer esto)

Ahora que se ha entrenado y guardado la red, con ella se realizarán simulaciones para las bases de entrenamiento y test, y se calcularán sus indicadores de eficiencia.

Simulación en la base para el entrenamiento y cálculos de eficiencia

```
a=sim(net,input_train);
```

Hallamos el total de calas observadas

```
indtt=find(target_train>thres);
total=length(indtt);
train_perf(3,ii)=total;
```

Después, el total de calas verdaderas:

```
indtrue=find(a>thres & target_train>thres);
```

primero se identifican aquellos patrones que cumplan 2 condiciones: que tengan salidas cuyos valores son mayores al umbral (es decir que para la red son calas), y que al mismo tiempo sus objetivos sean 1,indicando cala.

```
true=length(indtrue);
```

luego se cuenta el número de patrones en el que estas condiciones se cumplen

```
train_perf(1,ii)=true;
```

y se almacena

Para el total de calas fantasmas se realiza el mismo procedimiento:

```
indfalse=find(a>thres & target_train<=thres);
```

primero se identifican aquellos patrones que cumplan 2 condiciones: que tengan salidas cuyos valores son mayores al umbral (es decir, patrones para los que la red identificó cala), y que al mismo tiempo sus objetivos sean 0, indicando no cala.

```
false=length(indfalse);
```

luego se cuenta el número de patrones en el que estas condiciones se cumplen

```
train_perf(2,ii)=false;
```

y se almacena

```
train_perf(4,ii)=true+false;
```

se almacena también el CI

Aquí actúa el parámetro MSE máximo: en este bucle el mse de entrenamiento debe ser menor al MSE_max. Es la primera restricción.

```
if ((mse(target_train-a) < MSE_max))
disp(' ')
disp(['Number of trained networks: ' num2str(ii)]) ;
   disp(' ')
disp(['MSE of the trained network: ' num2str(mse(target_train-a))]) ;
```

Simulación en la base para el test y cálculos de eficiencia

```
a=sim(net,input_test);
```

Hallamos el total de calas observadas

```
indtt=find(target_test>thres);
total=length(indtt);
```

Después, el total de calas verdaderas

```
indtrue=find(a>thres & target_test>thres);
true=length(indtrue);
```

```
Luego, el total de calas fantasmas
indfalse=find(a>thres & target_test<=thres);
false=length(indfalse);

Segunda restricción para la red: verificamos si en este bucle, la red ha identificado calas. Si no lo ha hecho, éste bucle no quedará registrado y se correrá nuevamente. Pero si el número de calas verdaderas y falsas es diferente de cero, sí se registrará el bucle.

if true~=0 && false~=0

test_perf(3,ii)=total;
test_perf(1,ii)=true;
test_perf(2,ii)=false;
test_perf(4,ii)=true+false;
aquí se almacenan los indicadores de eficiencia del bucle para el test

para pasar de bucle (cumpliendo las 2 restricciones anteriores):
ii=ii+1;

else
si es que no se cumplió la segunda restricción, se registran los indicadores de esta corrida en la que no se identificaron calas, en una matriz llamada no_perf
   no_perf(1,iii)=iii;
   no_perf(2,iii)=true;
   no_perf(3,iii)=false;
   no_perf(4,iii)=total;
   no_perf(5,iii)=mse(target_test-a);
   a=sim(net,input_train);
   no_perf(6,iii)=mse(target_train-a);
   no_perf(7,iii)=thres;
   iii=iii+1;
end
(finaliza la condicional de la segunda restricción)

end
(finaliza la condicional de la primera restricción)

end
terminan de ejecutarse los bucles

Cálculo de los indicadores de eficiencia globales para los nb_loop bucles

meantr=mean(train_perf,2);
meante=mean(test_perf,2);
CVte1=meante(1)/meante(3)*100;
CFte1=meante(2)/meante(3)*100;

actualización de los valores de CO y CI, que utilizan el promedio de estos de entre todos los bucles para este valor de umbral específico
ttotal=meantr(3);
ttotnet=meantr(4);

threshold=threshold+nthreshold;
j=j+1;
para pasar el bucle de umbral;

end;

name=strcat('./ArchivosSalida/umbral_',num2str(inicio),'_',num2str(final),'.txt');
```

```
dlmwrite(name,thres,'\t');
```

Solo si deseamos mostrar los resultados de la red en la pantalla, utilizamos los comandos aquí debajo:

```
disp(' ')
disp(['Mean performances of the net: ']) ;
disp(['% identified true calas: ' num2str(CVte1)]) ;
disp(['% identified phantom calas: ' num2str(CFte1)]) ;

tiempo1=cputime-tiempo;
```

cputime nos da ahora cuánto tiempo en total ha pasado desde que se inició el trabajo en Matlab hasta ahora al restarle el tiempo que se tomó antes de que empiece el algoritmo, la diferencia nos dará el tiempo que se toma en ejecutar todo el programa

Acá se guarda el tiempo que demora en correr la net:

```
dlmwrite('./ArchivosSalida/time1.txt',tiempo1,'\t')
```

Acá se guardan los resultados malos que puedan haber salido: cuando en un bucle no hay ninguna cala identificada. Si no pasa esto, no se creará el archivo

```
if iii~=1
    dlmwrite('./ArchivosSalida/bad_perf.txt',no_perf,'\t')
end
```

AHORA PROBANDO CON UN VALOR MENOR DE MSE
Hacemos el mismo procedimiento.

```
tiempo=cputime;

MSE_max=0.02;
```

Es el único parámetro que cambiará.

Nuevamente creamos matrices vacías (con elementos NaN) que luego utilizaremos para almacenar resultados.

```
test_perf=NaN*ones(3,nb_loop);
train_perf=NaN*ones(3,nb_loop);

ii=1;
iii=1;

while ii <= nb_loop

tab_mod=tab_mod(:,randperm(nb_data_mod));
target=tab_mod(9,:);

input=tab_mod(num_input,:);

target_test=target(1,1:size_seg);
input_test=input(:,1:size_seg);
t.P=input_test;
t.T=target_test;

target_valid=target(1,size_seg+1:2*size_seg);
input_valid=input(:,size_seg+1:2*size_seg);
v.P=input_valid;
v.T=target_valid;

target_train=target(1,2*size_seg+1:nb_data_mod);
input_train=input(:,2*size_seg+1:nb_data_mod);
```

```
net=newff([minmax(input)],[nb_hidden,1],{'tansig','logsig'},'trainlm');

net.trainParam.show=NaN;
net.trainParam.epochs=1000;
net.trainParam.max_fail=20;
net=init(net);
 [net,tr]=train(net,input_train,target_train,[],[],v);

name=strcat('./RedesEntrenadas/net2_',num2str(ii),'_',num2str(inicio),'_',num2str(final),'.mat');
save(name, 'net')

a=sim(net,input_train);

indtt=find(target_train>thres);
total=length(indtt);
train_perf(3,ii)=total;
indtrue=find(a>thres & target_train>thres);
true=length(indtrue);
train_perf(1,ii)=true;
indfalse=find(a>thres & target_train<=thres);
false=length(indfalse);
train_perf(2,ii)=false;

if ((mse(target_train-a) < MSE_max))
disp(' ')
disp(['Number of trained networks: ' num2str(ii)]) ;
   disp(' ')
disp(['MSE of the trained network: ' num2str(mse(target_train-a))]) ;

a=sim(net,input_test);

indtt=find(target_test>thres);
total=length(indtt);
indtrue=find(a>thres & target_test>thres);
true=length(indtrue);
indfalse=find(a>thres & target_test<=thres);
false=length(indfalse);

if true~=0 && false~=0

test_perf(3,ii)=total;
test_perf(1,ii)=true;
test_perf(2,ii)=false;

ii=ii+1;

else
   no_perf(1,iii)=iii;
   no_perf(2,iii)=true;
   no_perf(3,iii)=false;
   no_perf(4,iii)=total;
   no_perf(5,iii)=mse(target_test-a);
    a=sim(net,input_train);
   no_perf(6,iii)=mse(target_train-a);
   iii=iii+1;
end

end
```

```
end

meante=mean(test_perf,2);
CVte2=meante(1)/meante(3)*100;
CFte2=meante(2)/meante(3)*100;

disp(' ')
disp(['Mean performances of the net: ']) ;
disp(['% identified true calas: ' num2str(CVte2)]) ;
disp(['% identified phantom calas: ' num2str(CFte2)]) ;

tiempo2=cputime-tiempo;

dlmwrite('./ArchivosSalida/time2.txt',tiempo2,'\t')

if iii~=1
    dlmwrite('./ArchivosSalida/no_perf2','\t');
end

if CVte1>CVte2
   C=[CVte1 CFte1];
   red=1;
else
   C=[CVte2 CFte2];
   red=2;
end
name=strcat('./ArchivosSalida/indicadores_',num2str(inicio),'_',num2str(final),'.txt');
dlmwrite(name,C,'\t');
name=strcat('./ArchivosSalida/red_',num2str(inicio),'_',num2str(final),'.txt');
dlmwrite(name,red,'\t');
```

Se guardan los valores de %CV y %CF de la red (con el umbral elegido) que tenga mayor CV. Ese es el criterio para elegir entre los valores de MSE_max. Ese par de CV y CF “elegidos” se guardan en el archivo indicadores. El archivo red también se guarda. Guardará el número 2 si se eligió la red con MSEmax=0.02 o 1 si se escogió la red con MSEmax=0.03.

```
resultado=0;
```

Predicción (identificación) de calas con la red entrenada

Para la predicción se utilizará la red entrenada, por lo cual se debe asegurar que la base de datos para la predicción tenga el mismo formato que la entrenada, es decir, que el orden y formato de las variables sea el mismo (id, código del barco, código del viaje, día, mes, año, longitud, latitud, cala, hora, velocidad, cambio de velocidad 1, cambio de velocidad 2 y diferencia de rumbo). En el lugar que correspondía a la variable cala, al no disponer de esa información, se rellena con números cualquiera, pues luego de la predicción se colocará sobre esa variable los resultados predichos.

Recordemos que el número de variables de entrada es 5 (nb_input=5) y que están ubicadas en las columnas 10, 11, 12, 13 y 14 (num_input=[10,11,12,13,14]); y que si el número de bucles en el entrenamiento fue nb_loop, ahora se mantiene.

```
Function resultado=prediccion(dia,mes,ano,AnoInicio,AnoFinal,unaBasePrediccion)
```

El nombre de la función es predicción. Sus parámetros de entrada son:

- Día, mes y ano: es la fecha de ejecución de la interfaz de predicción.
- AnoInicio y AnoFinal: son los años para los que se entrenará la base.
- unaBasePrediccion: es el nombre del archivo de la base de predicción.

Resultado es la variable de salida de la función.

```
warning off;
inicio = AnoInicio;
final = AnoFinal;

fecha=strcat(num2str(dia),'-',num2str(mes),'-',num2str(ano));
se unen los 3 datos de fecha en un solo array

BasePrediccion = unaBasePrediccion;

Abrimos el archivo con la base de datos para la predicción (aquella que tiene variables de entrada mas no se sabe para qué patrones hubo cala); luego creamos una matriz en la que se almacenan todos los datos de la base de datos original:

data = load(strcat('./BasePrediccion/',BasePrediccion));
tab_orig=data';

Se calcula el número de observaciones:
nb_data_orig=length(tab_orig);

Se crea un matriz (que a continuación modificaremos) con la data de la matriz anterior:
tab_mod=tab_orig;

Como se hizo para la base de datos entrenada, primero se transforman las variables circulares y luego se estandariza toda la data:
tab_mod(10,:)=cos(tab_mod(10,:)*pi/12.);
tab_mod(14,:)=cos(tab_mod(14,:)*pi/180.);
[tab_mod(10:14,:),meaninp,stdinp] = prestd(tab_mod(10:14,:));

Se crea una matriz conteniendo los patrones de entrada:
num_input=[10,11,12,13,14];
predict_set=tab_mod(num_input,:);

Predicción:

Se descarga primero el número de bucles que se utilizó para el entrenamiento:
load('./ArchivosSalida/nb_loop.txt');

Ahora, para cada uno de los nb_loop bucles, se realizará la predicción de la variable de salida (una continua de 0 a 1)

name=strcat('./ArchivosSalida/red_',num2str(inicio),'_',num2str(final),'.txt');
red=load(name);

for ii=1:nb_loop

Dentro de cada bucle:

Se llama al archivo donde se guardó la red entrenada que corresponde a ese bucle:

filename = ['./RedesEntrenadas/net' num2str(red) '_' num2str(ii) '_' num2str(inicio) '_' num2str(final) '.mat'];
load(filename)

pred(ii,:)=sim(net,predict_set);
con esa red se simulan resultados para los patrones de entrada, lo cual dará como resultado una variable de salida: ésta es la predicción

end
(terminan de ejecutarse los bucles)
```

```
Después de haber realizado la simulación en cada uno de los bucles, se calcula el promedio para las respuestas de cada uno de los patrones
pred_mean=mean(pred,1);

j=1;

Se descarga el archivo con el valor del umbral para hacer las comparaciones e identificar para cada uno de los patrones, si hubo o no cala:

name=strcat('ArchivosSalida/umbral_',num2str(inicio),'_',num2str(final),'.txt');
thres=load(name);

for i=1:nb_data_orig
   if pred_mean(i)>thres; (si el promedio de las salidas para ese patrón es mayor al umbral)
      forecast(j,:)=[data(i,2) data(i,4:8)]; (creando una matriz que me da para cada cala: el código de barco, fecha (día, mes, año), longitud, latitud)
      j=j+1;
     (entonces hubo cala)
   end
end

calas=j-1;
dlmwrite('./ArchivosSalida/calas.txt',calas,'\t');
aquí guardamos el número de calas predichas

y la información de las calas (matriz forecast) se guarda en un archivo llamado forecast, en formato texto
name=strcat('./ArchivosSalida/forecast_',num2str(inicio),'_',num2str(final),'_',fecha,'.txt');
dlmwrite(name,forecast,'\t');

resultado=0;
```

1.2. Sintaxis de la prueba de sensibilidad para el número de nodos

Es una de las sintaxis que se crearon para la prueba de sensibilidad. En este caso, el parámetro a testear es el número de nodos.

```
clear
tic
disp(' ')
disp('PROGRAM TRAINING NEURAL NETWORK: CALA IDENTIFICATION') ;
disp(' ')

fecha=102007;
nb_input=5;
num_input=[10,11,12,13,14];
nb_min=2;
nb_max=8;
nb_loop=50;
kcross=5;
t_net='lm_es';
threshold=0.66;
MSE_max=0.03;
filename='tribase02.txt'

disp(' ')
```

```
disp(['Date: ' num2str(fecha)]) ;
disp(' ')
disp(['Number of input variables: ' num2str(nb_input)]) ;
disp(' ')
disp(['Size of the validation and test sets: 1/' num2str(kcross)]) ;
disp(' ')
disp(['Number of trained networks for this loop: ' num2str(nb_loop)]) ;
disp('')
disp(['Maximum MSE accepted for a trained network:' num2str(MSE_max)]) ;
disp(' ')
disp(['Threshold used for calas identification: ' num2str(threshold)]) ;
disp('')

fid=fopen(filename,'rt');
tab_orig=fscanf(fid,'%g %g %g %g %g %g %g %g %g %g %g %g %g %g %g',[14,inf]);
nb_data_orig=length(tab_orig);
tab_mod=tab_orig;
tab_mod(10,:)=cos(tab_mod(10,:)*pi/12.);
tab_mod(14,:)=cos(tab_mod(14,:)*pi/180.);
tab_mod(10:14,:) = prestd(tab_mod(10:14,:));
nb_data_mod=length(tab_mod);
size_seg=round(nb_data_mod/kcross);

nb_hidden=nb_min;
j=1;
 nb=nb_max-nb_min+1;
   n=NaN*ones(1,nb);
   b=NaN*ones(1,nb);
   test_perf=NaN*ones(8,nb_loop);
test_resu=NaN*ones(2,nb_loop,size_seg);
valid_perf=NaN*ones(8,nb_loop);
valid_resu=NaN*ones(2,nb_loop,size_seg);
train_perf=NaN*ones(8,nb_loop);
train_resu=NaN*ones(2,nb_loop,nb_data_mod-2*size_seg);
neural_perf=NaN*ones(2,nb_loop);
perf_tr=NaN*ones(8,nb);
perf_va=NaN*ones(8,nb);
perf_te=NaN*ones(8,nb);

while nb_hidden<=nb_max
disp(' ')
    disp(['Number of neurons in the hidden layer: ' num2str(nb_hidden)]) ;
disp(' ')
n(j)=nb_hidden;
t=cputime;
   ii=1;
   iii=1;
while ii <= nb_loop
    tab_mod=tab_mod(:,randperm(nb_data_mod));
target=tab_mod(9,:);
input=tab_mod(num_input,:);

target_test=target(1,1:size_seg);
input_test=input(:,1:size_seg);

target_valid=target(1,size_seg+1:2*size_seg);
input_valid=input(:,size_seg+1:2*size_seg);
v.P=input_valid;
v.T=target_valid;
```

```
target_train=target(1,2*size_seg+1:nb_data_mod);
input_train=input(:,2*size_seg+1:nb_data_mod);

net=newff([minmax(input)],[nb_hidden,1],{'tansig','logsig'},'trainlm');
net.trainParam.show=NaN;
net.trainParam.epochs=1000;
net.trainParam.max_fail=20;

net=init(net);
[net,tr]=train(net,input_train,target_train,[],[],v);

a=sim(net,input_train);
train_resu(:,ii,:)=[target_train;a];
train_perf(3,ii)=mse(target_train-a);
cor=corrcoef(target_train,a);
train_perf(4,ii)=cor(2,1);
indtt=find(target_train>threshold);
total=length(indtt);
train_perf(5,ii)=total;
indtrue=find(a>threshold & target_train>threshold);
true=length(indtrue);
train_perf(6,ii)=true;
train_perf(1,ii)=true/total*100;
indfalse=find(a>threshold & target_train<=threshold);
false=length(indfalse);
train_perf(7,ii)=false;
train_perf(2,ii)=false/total*100;
train_perf(8,ii)=true/false;

if ((train_perf(3,ii) < MSE_max))
disp(' ')
disp(['Number of trained networks: ' num2str(ii)]) ;
   disp(' ')
disp(['MSE of the trained network: ' num2str(train_perf(3,ii))]) ;
   disp(' ')

dim=length(tr.epoch);
neural_perf(:,ii)=[tr.epoch(dim);tr.perf(dim)];

a=sim(net,input_test);
indtt=find(target_test>threshold);
total=length(indtt);
indtrue=find(a>threshold & target_test>threshold);
true=length(indtrue);
indfalse=find(a>threshold & target_test<=threshold);
false=length(indfalse);
cor=corrcoef(target_test,a);

if true~=0 && false~=0

test_resu(:,ii,:)=[target_test;a];
test_perf(3,ii)=mse(target_test-a);
test_perf(4,ii)=cor(2,1);
test_perf(5,ii)=total;
test_perf(6,ii)=true;
test_perf(1,ii)=true/total*100;
test_perf(7,ii)=false;
test_perf(2,ii)=false/total*100;
```

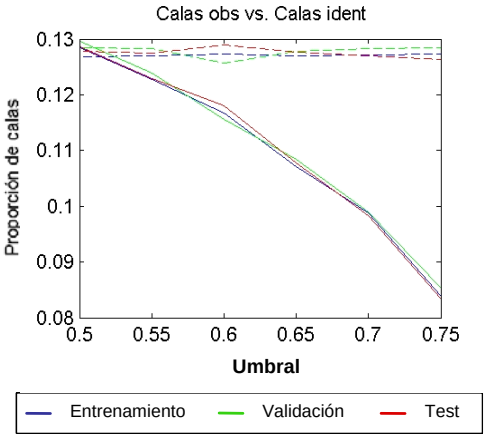
```
test_perf(8,ii)=true/false;

a=sim(net,input_valid);
valid_resu(:,ii,:)=[target_valid;a];
valid_perf(3,ii)=mse(target_valid-a);
cor=corrcoef(target_valid,a);
valid_perf(4,ii)=cor(2,1);
indtt=find(target_valid>threshold);
total=length(indtt);
valid_perf(5,ii)=total;
indtrue=find(a>threshold & target_valid>threshold);
true=length(indtrue);
valid_perf(6,ii)=true;
valid_perf(1,ii)=true/total*100;
indfalse=find(a>threshold & target_valid<=threshold);
false=length(indfalse);
valid_perf(7,ii)=false;
valid_perf(2,ii)=false/total*100;
valid_perf(8,ii)=true/false;

   ii=ii+1;

else

   no_perf(1,iii)=nb_hidden;
   no_perf(2,iii)=iii;
   no_perf(3,iii)=true;
   no_perf(4,iii)=false;
   no_perf(5,iii)=total;
   no_perf(6,iii)=mse(target_test-a);
   no_perf(7,iii)=cor(2,1);
   a=sim(net,input_train);
   no_perf(8,iii)=mse(target_train-a);
   iii=iii+1;

end
 end
 end

 meantr=mean(train_perf,2);
stdtr=std(train_perf,0,2);
CVtr=meantr(6)/meantr(5)*100;
CFtr=meantr(7)/meantr(5)*100;
Rtr=meantr(6)/meantr(7);
meanva=mean(valid_perf,2);
stdva=std(valid_perf,0,2);
CVva=meanva(6)/meanva(5)*100;
CFva=meanva(7)/meanva(5)*100;
Rva=meanva(6)/meanva(7);
meante=mean(test_perf,2);
stdte=std(test_perf,0,2);
CVte=meante(6)/meante(5)*100;
CFte=meante(7)/meante(5)*100;
Rte=meante(6)/meante(7);

perf_tr(:,j)=[meantr(3); stdtr(3); CVtr; CFtr; Rtr; meantr(1:2); meantr(8)];
perf_va(:,j)=[meanva(3); stdva(3); CVva; CFva; Rva; meanva(1:2); meanva(8)];
perf_te(:,j)=[meante(3); stdte(3); CVte; CFte; Rte; meante(1:2); meante(8)];
nb_hidden=nb_hidden+1;
```

```
b(j)=cputime-t;
j=j+1;

end;
nperf=[n; perf_tr; perf_va; perf_te];

figure('Name','nodesI','NumberTitle','off')
subplot(2,2,1), plot(n,perf_tr(3,:),'-',n,perf_va(3,:),':',n,perf_te(3,:),'--');
xlabel('Nodos');
title('% CALAS VERDADERAS');
subplot(2,2,2), plot(n,perf_tr(4,:),'-',n,perf_va(4,:),':',n,perf_te(4,:),'--');
xlabel('Nodos');
title('% CALAS FANTASMAS');
subplot(2,2,3), plot(n,perf_tr(1,:),'-',n,perf_va(1,:),':',n,perf_te(1,:),'--');
xlabel('Nodos');
title('MSE');
subplot(2,2,4), plot(n,perf_tr(2,:),'-',n,perf_va(2,:),':',n,perf_te(2,:),'--');
xlabel('Nodos');
title('STD(MSE)');
print -djpeg100 nodesI

figure('Name','newnodesI','NumberTitle','off')
subplot(2,2,1),
plot(n,perf_tr(5,:),'-',n,perf_va(5,:),':',n,perf_te(5,:),'--');
title('RATIO CV/CF');
print -djpeg100 newnodesI

name=strcat('performanceI_nodes_',num2str(nb_min),'-',num2str(nb_max),filename);
dlmwrite(name,nperf,'\t');
name=strcat('timeI_nodes_',num2str(nb_min),'-',num2str(nb_max),filename);
dlmwrite(name,b,'\t');
if iii~=1
    name=strcat('badI_nodes_',num2str(nb_min),'-',num2str(nb_max),filename);
    dlmwrite(name,no_perf,'\t');
end

toc
```

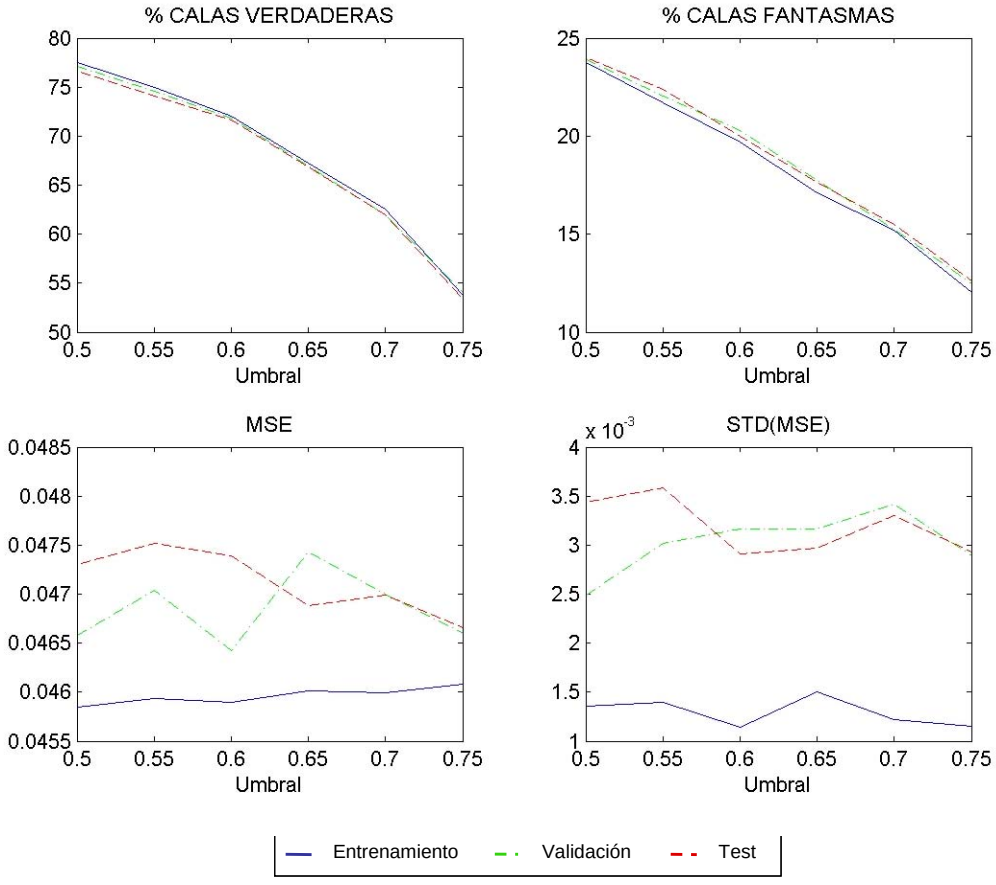
Anexo 2:

Resultados del análisis de sensibilidad – base 2000-2006

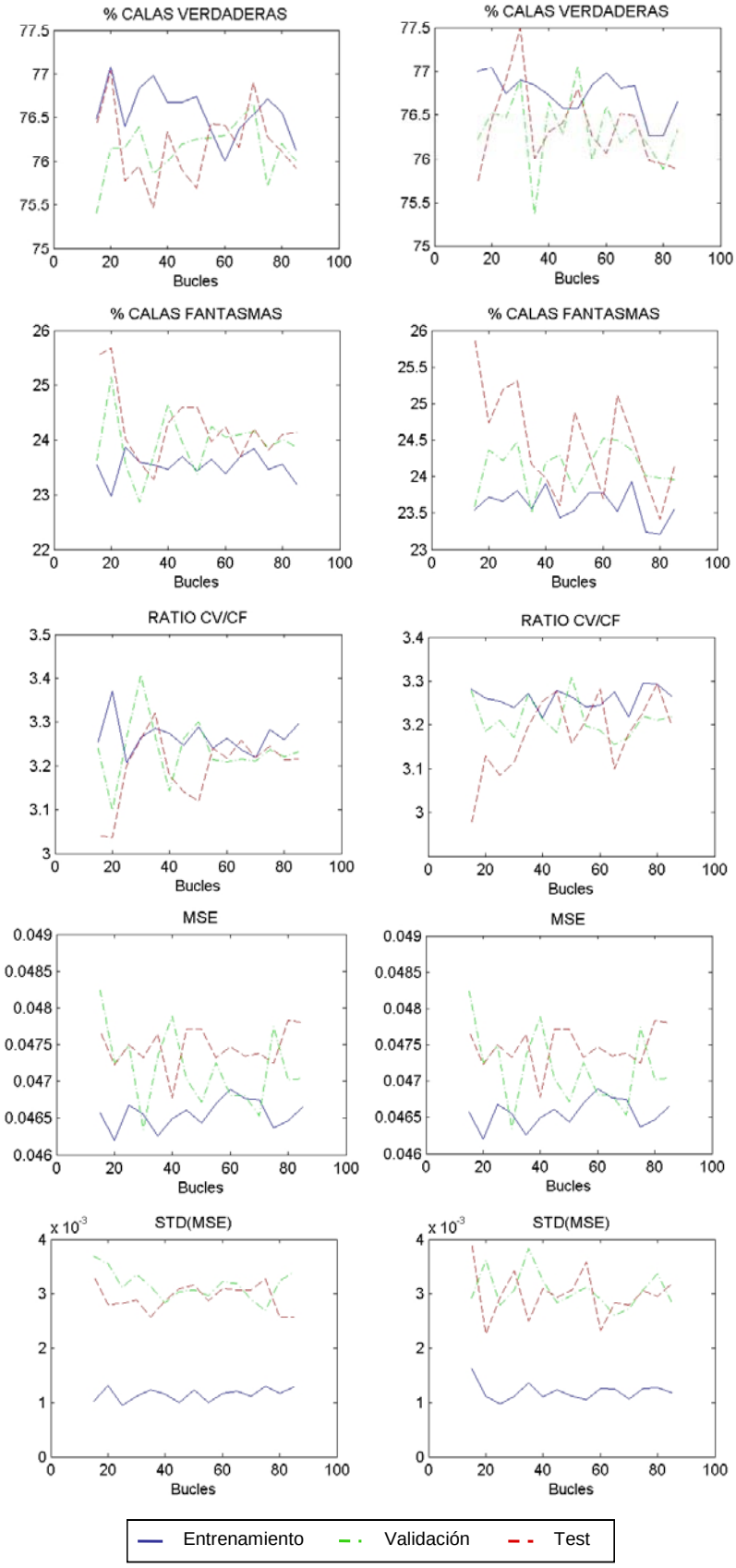
2.1. Total de calas observadas vs. Total de calas identificadas (2000-2006). Se varió el umbral desde 0.5 hasta 0.75 incrementando el umbral en 0.05 para cada ensayo. Las líneas continuas representan a la proporción de calas observadas y las líneas discontinuas, a la proporción de calas identificadas.



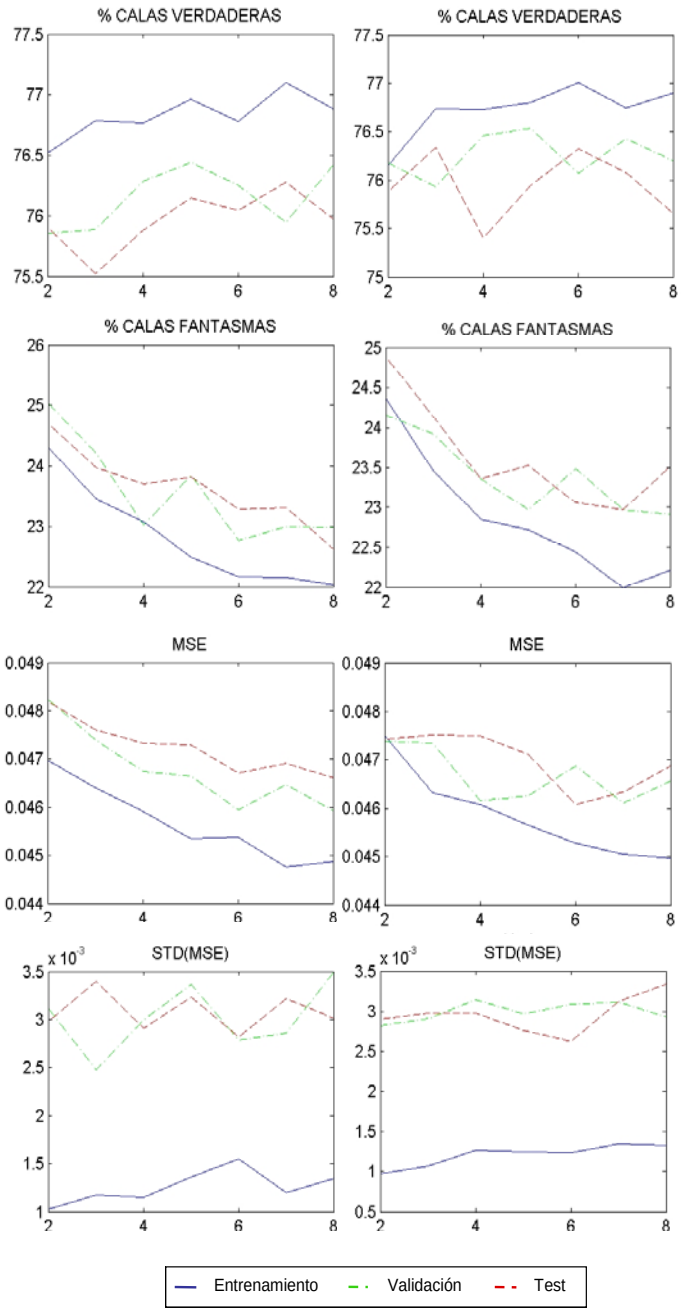
2.2. Indicadores de desempeño de la red testeando el umbral desde 0.5 hasta 0.75 (base 2000-2006).



2.3. Indicadores de desempeño de la red, testeando los bucles desde 15 hasta 85 (2000-2006). Aquí se muestran 2 réplicas de la realización de las corridas de la red.



2.4. Indicadores de desempeño de la red, testeando el número de nodos en la capa escondida desde 2 hasta 8 (2000-2006). El eje horizontal es el número de nodos escondidos. Aquí se muestran los resultados de 2 réplicas de la prueba.



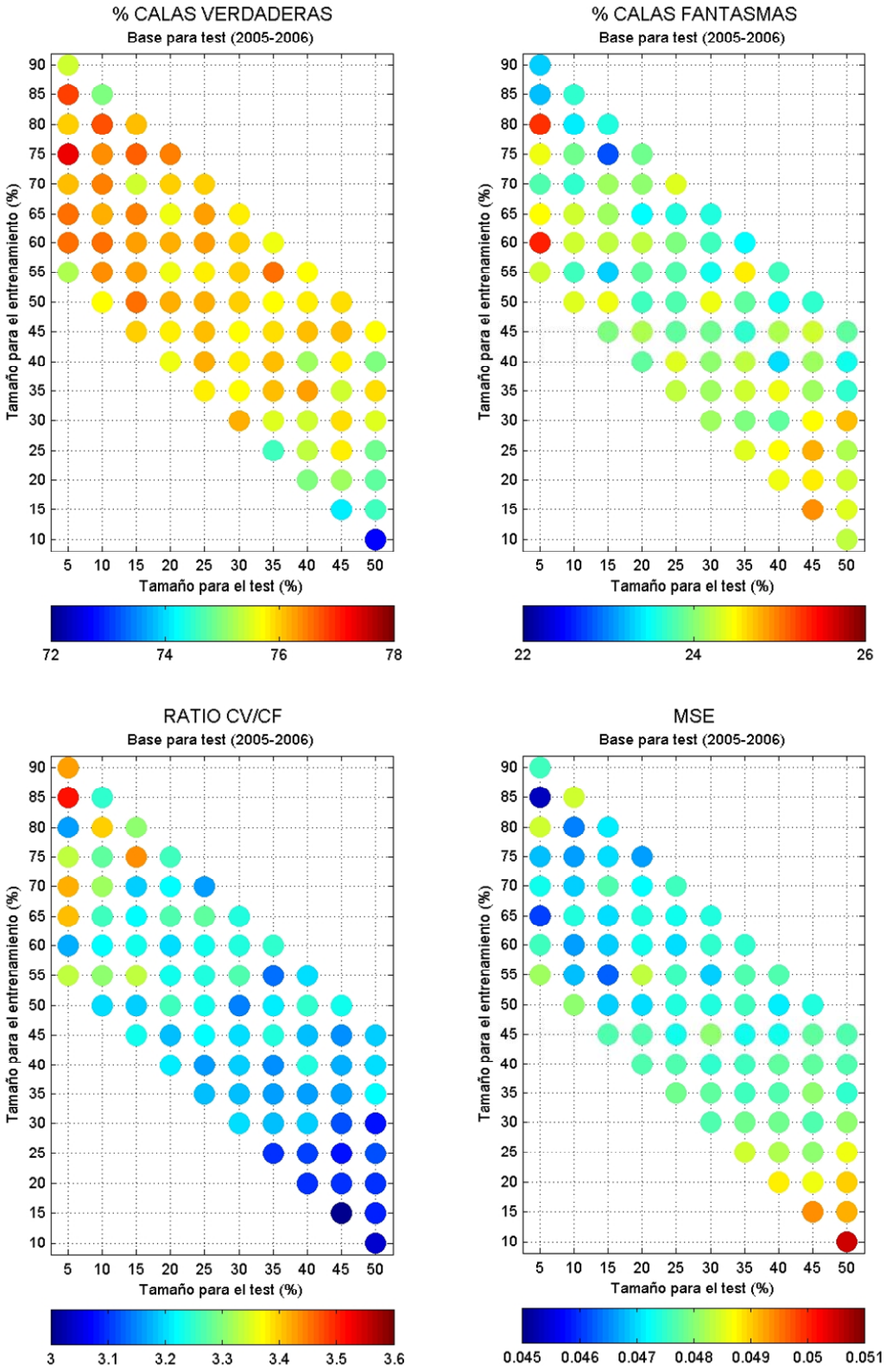
2.5. Tiempo que tarda la red (2000-2006) para mostrar resultados para diferentes número de nodos en la capa escondida.

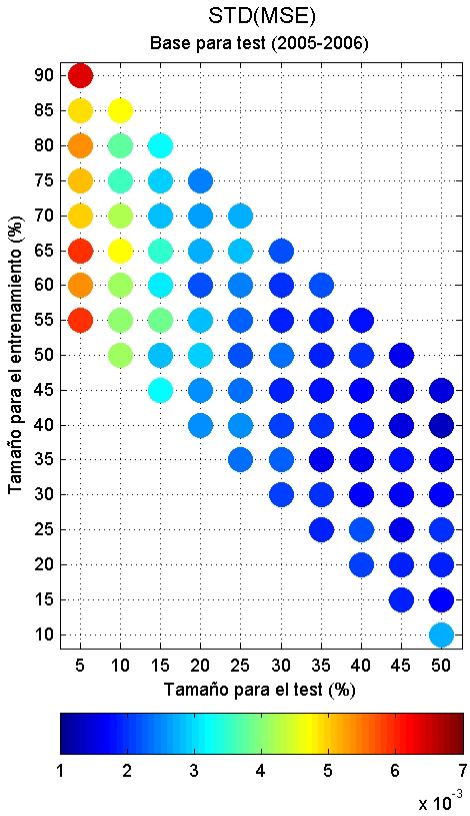
neuronas	tiempo (m)
4	14.7
5	16.8
6	16.0

2.6. 80 combinaciones para las proporciones de tamaño de las bases (respecto al total de la base de datos 2000-2006), variando desde 5 a 50% para el test y de 5 a 40% para la validación.

TEST	VALIDACIÓN	ENTRENAMIENTO
5	5	90
5	10	85
5	15	80
5	20	75
5	25	70
5	30	65
5	35	60
5	40	55
10	5	85
10	10	80
10	15	75
10	20	70
10	25	65
10	30	60
10	35	55
10	40	50
15	5	80
15	10	75
15	15	70
15	20	65
15	25	60
15	30	55
15	35	50
15	40	45
20	5	75
20	10	70
20	15	65
20	20	60
20	25	55
20	30	50
20	35	45
20	40	40
25	5	70
25	10	65
25	15	60
25	20	55
25	25	50
25	30	45
25	35	40
25	40	35
30	5	65
30	10	60
30	15	55
30	20	50
30	25	45
30	30	40
30	35	35
30	40	30
35	5	60
35	10	55
35	15	50
35	20	45
35	25	40
35	30	35
35	35	30
35	40	25
40	5	55
40	10	50
40	15	45
40	20	40
40	25	35
40	30	30
40	35	25
40	40	20
45	5	50
45	10	45
45	15	40
45	20	35
45	25	30
45	30	25
45	35	20
45	40	15
50	5	45
50	10	40
50	15	35
50	20	30
50	25	25
50	30	20
50	35	15
50	40	10

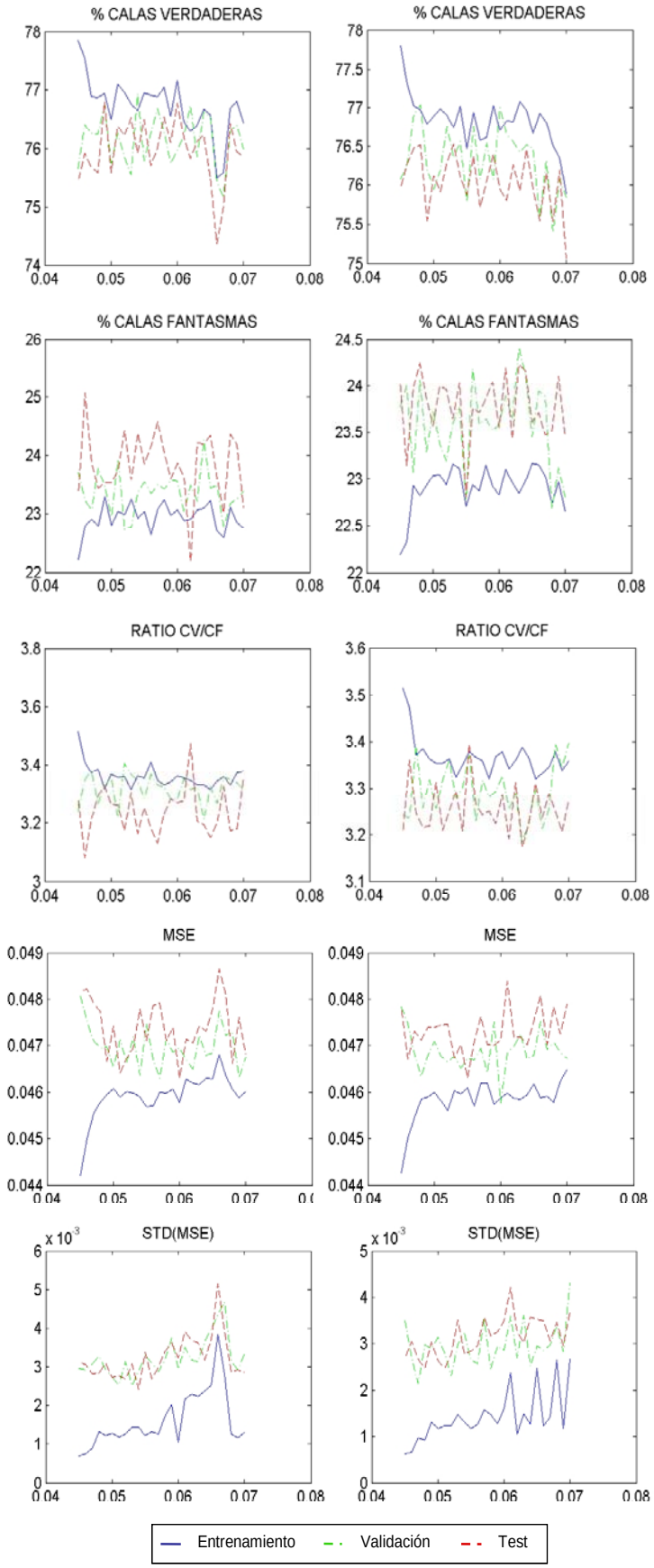
2.7. Indicadores de eficacia y eficiencia de la red (testeando el tamaño de las bases) para la base del test. El eje horizontal corresponde al tamaño de la base para el test y el eje vertical, al tamaño de la base para el entrenamiento. Ambos expresados en porcentaje respecto a la base de datos total 2000-2006. La tercera dimensión, representada por los colores, corresponde al indicador que se señala en el título de cada gráfica.




STD(MSE)
Base para test (2005-2006)
Tamaño para el entrenamiento (%)
90
85
80
75
70
65
60
55
50
45
40
35
30
25
20
15
10
5 10 15 20 25 30 35 40 45 50
Tamaño para el test (%)
1 2 3 4 5 6 7
x 10^-3

2.8. Indicadores de desempeño de la red, testeando el MSE máximo entre 0.045 y 0.07 (2000-2006). El eje horizontal es el MSE máximo. Aquí se muestran los resultados de 2 réplicas de la prueba.

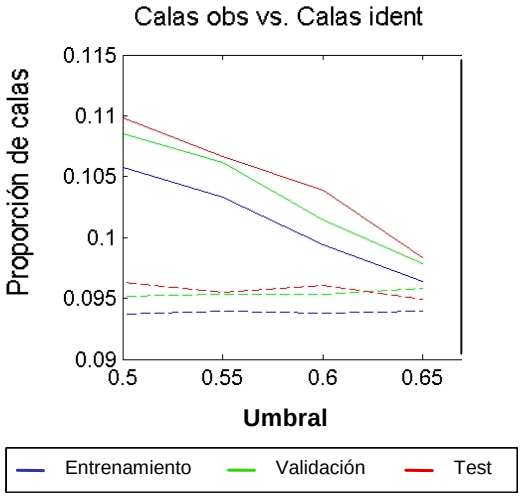
.



Anexo 3:

Resultados del análisis de sensibilidad – bases bianuales

3.1. Total de calas observadas vs. Total de calas identificadas (2000-2001). Se varió el umbral desde 0.5 hasta 0.65 incrementando el umbral en 0.05 para cada ensayo. Las líneas continuas representan a la proporción de calas observadas y las líneas discontinuas, a la proporción de calas identificadas. No se encontró punto de intersección.



3.2. Valores de los indicadores para el valor inicial de umbral 0.5 (izquierda) y porcentaje de variación de los indicadores de redes de umbrales diferentes con respecto a una red con umbral 0.5 (derecha) (2000-2001).

CV	90.18
CF	25.07
RATIO	3.94
MSE	0.026
STD(MSE)	0.005

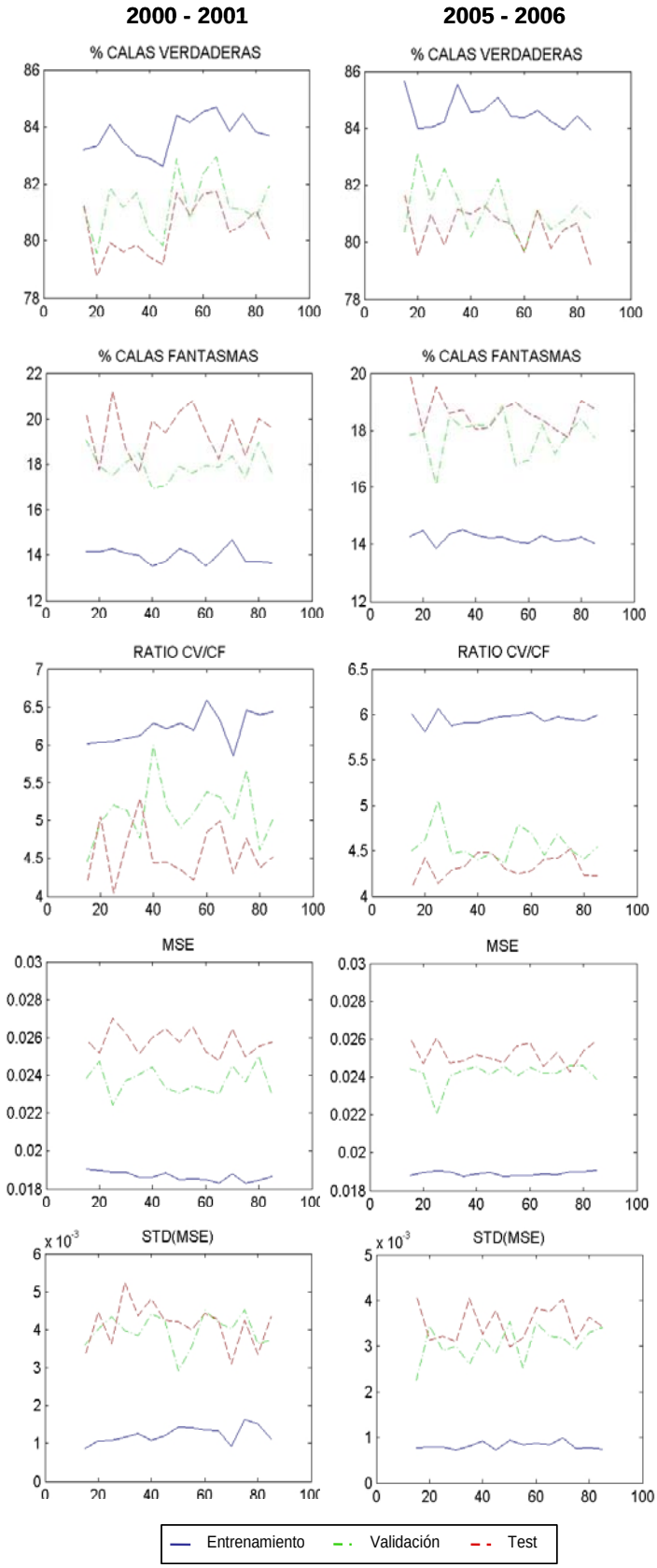
	CV	CF	RATIO	MSE	STD(MSE)
0.55	-1.50	-7.66	3.87	-2.949	-21.7356
0.6	-3.04	-11.49	6.66	-4.274	-17.7573
0.65	-5.48	-17.32	18.39	-3.573	-9.7765
0.7	-8.08	-19.40	10.39	-0.946	-25.2020
0.71	-9.14	-21.67	16.64	-4.700	-9.1457
0.75	-14.53	-30.02	21.21	-1.704	-20.8368
0.8	-20.45	-40.66	49.44	-4.179	-8.7728

3.3. Valores de los indicadores para el valor inicial de umbral 0.5 (izquierda) y porcentaje de variación de los indicadores de redes de umbrales diferentes con respecto a una red con umbral 0.5 (derecha) (2005-2006).

CV	85.95
CF	22.79
RATIO	3.77
MSE	0.026
STD(MSE)	0.0044

	CV	CF	RATIO	MSE	STD(MSE)
0.55	-0.19	-1.14	0.96	-0.942	-17.2677
0.6	-4.26	-16.93	15.25	-1.433	-5.5793
0.65	-4.74	-15.07	12.15	-1.199	-20.6653
0.66	-5.00	-19.08	17.40	-2.893	-23.4412
0.7	-8.93	-24.04	19.90	-3.847	-12.4065
0.75	-13.43	-26.99	18.59	0.004	-14.8933
0.8	-22.43	-39.96	29.21	-7.784	-30.6630

3.4. Indicadores de desempeño de la red (testeando los bucles desde 15 hasta 85). El eje horizontal es el número de bucles. A la izquierda, las gráficas de las pruebas realizadas para la base 2000-2001. A la derecha, para la base 2005-2006.



3.5. Valores de los indicadores para el valor inicial de umbral 0.72 entrenada utilizando 50 bucles (izquierda) y porcentaje de variación de los indicadores (calculados para redes con diferentes números de bucles) con respecto a una red con 50 bucles (derecha) (2000-2001).

CV	82.21
CF	19.65
RATIO	4.87
MSE	0.025
STD(MSE)	0.004

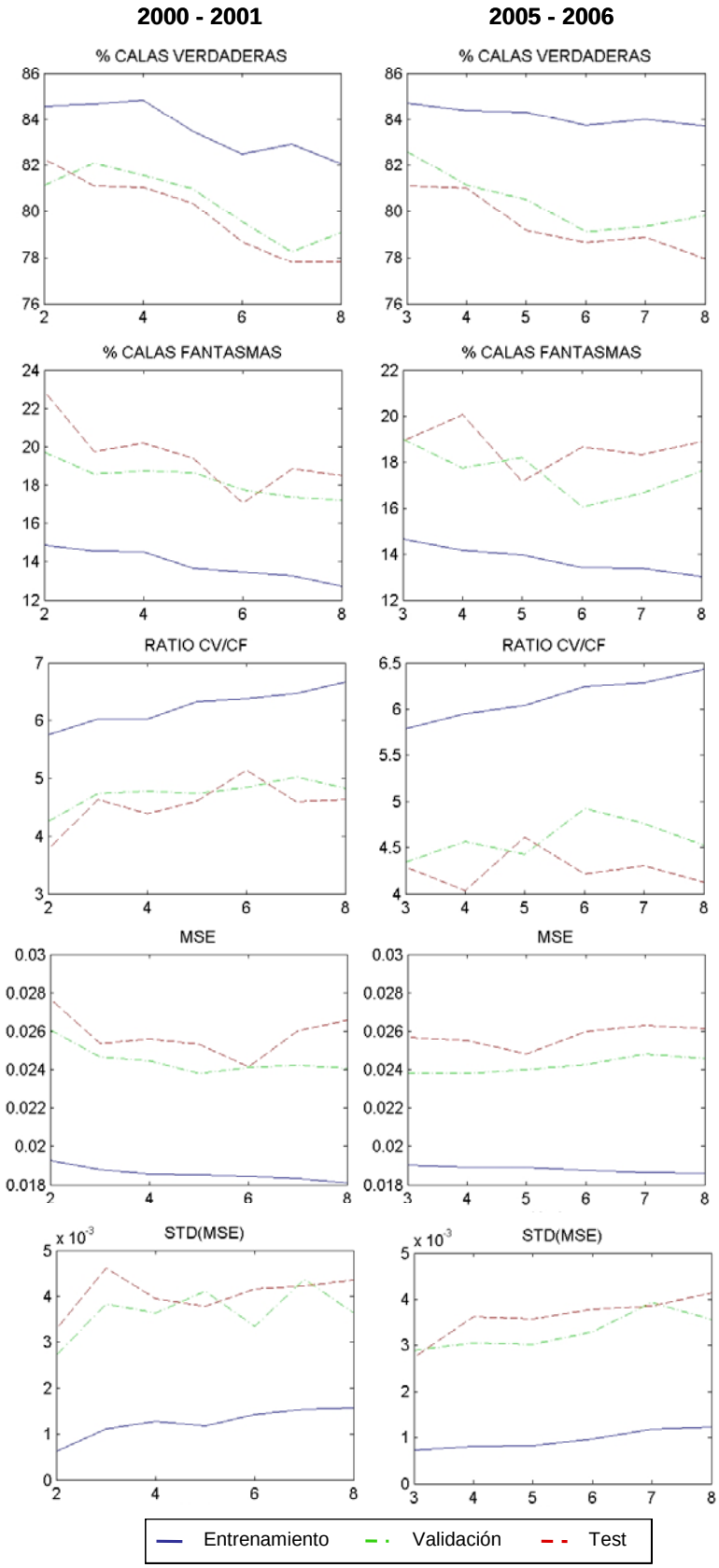
BUCLES	CV	CF	RATIO	MSE	STD(MSE)
20	-4.21	-9.75	3.98	0.793	19.4031
25	-2.79	8.10	-17.03	8.302	-3.1795
30	-3.17	-4.93	-2.85	5.077	40.1219
35	-2.89	-10.23	8.75	0.761	17.0286
40	-3.40	1.41	-8.92	4.147	28.9951
45	-3.70	-1.15	-8.75	5.966	13.9052
50	-0.65	3.41	-10.82	3.185	12.3783
55	-1.58	5.99	-13.67	6.487	7.0088
60	-0.70	-1.08	-0.48	1.162	18.1249
65	-0.61	-7.10	2.38	-0.673	14.2823
70	-2.36	1.86	-11.82	6.107	-17.0446
75	-1.98	-6.45	-2.20	0.044	13.8731
80	-1.38	1.88	-10.41	2.204	-10.3006

3.6. Valores de los indicadores para el valor inicial de umbral 0.66 entrenada utilizando 50 bucles (izquierda) y porcentaje de variación de los indicadores (calculados para redes con diferentes números de bucles) con respecto a una red con 50 bucles (derecha) (2005-2006).

CV	80.90
CF	18.43
RATIO	4.39
MSE	0.025
STD(MSE)	0.0025

BUCLES	CV	CF	RATIO	MSE	STD(MSE)
20	-3.86	-7.04	3.42	-0.775	38.8823
25	0.42	10.79	-9.35	4.620	69.5591
30	-0.96	5.33	-5.97	5.058	55.1089
35	0.36	3.23	-2.78	2.004	8.2095
40	-0.39	3.14	-3.41	1.326	39.2124
45	-0.20	2.91	-3.02	0.195	23.6815
50	-1.01	2.43	-3.36	2.109	26.1298
55	-0.01	9.77	-8.91	5.561	58.0877
60	-1.19	2.85	-3.92	2.474	13.3027
65	-0.27	-1.43	1.18	0.041	28.7354
70	-0.33	2.86	-3.11	1.314	54.9792
75	0.61	1.66	-1.04	-0.544	57.4747
80	-2.34	-4.62	2.38	-0.268	35.9546

3.7. Indicadores de desempeño de la red. A la izquierda, las gráficas de las pruebas realizadas para la base 2000-2001 (testeando el número de nodos desde 2 hasta 8). A la derecha, para la base 2005-2006 (testeando el número de nodos desde 3 hasta 8). El eje horizontal es del número de nodos escondidos.



3.8. Tiempo que tarda la red para mostrar resultados para diferentes números de nodos en la capa escondida. A la izquierda, para la base 2000-2001 y a la derecha, para la base 2005-2006.

BASE 2000-2001

neuronas	tiempo (m)
2	11.74
3	6.54
4	5.06
5	4.42
6	2.22
7	2.25
8	2.54

BASE 2005-2006

neuronas	tiempo (m)
2	----
3	38.80
4	17.94
5	16.63
6	14.69
7	14.86
8	14.82

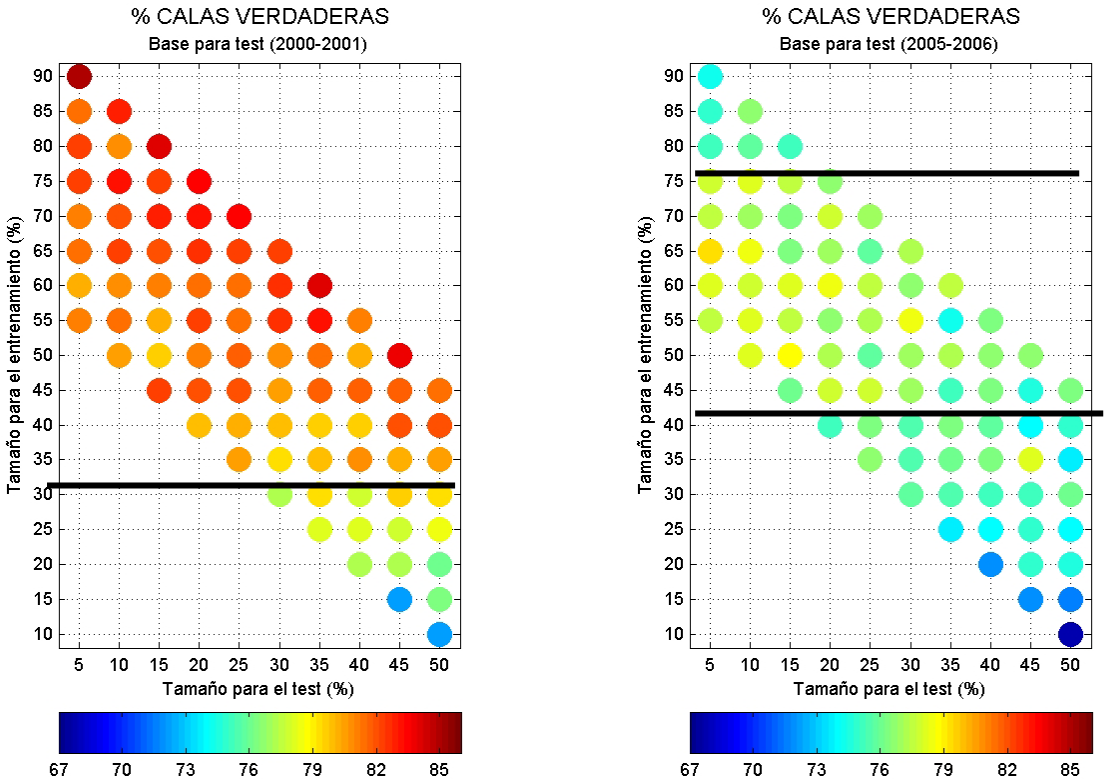
3.9. Porcentaje de variación de los indicadores (de redes con diferentes números de nodos) con respecto a una red con umbral 0.72, entrenada utilizando 4 nodos en la capa escondida (2000-2001).

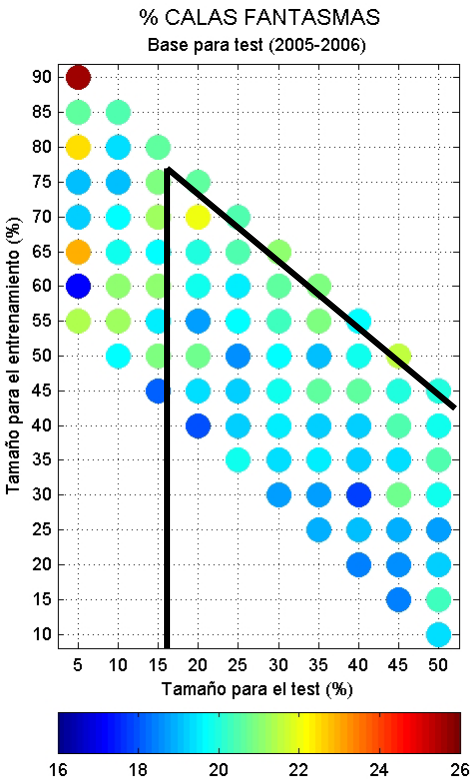
NODOS	CV	CF	RATIO	MSE	STD(MSE)
2	-0.34	9.32	-17.82	9.088	-15.2984
3	-1.56	-2.05	-7.55	3.109	2.1714
4	-1.41	0.84	-6.88	1.066	15.5712
5	-3.75	-3.16	-5.97	6.375	0.0107
6	-3.89	-4.87	-7.09	1.130	-3.6929
7	-4.34	-2.71	-8.41	6.046	-3.7624
8	-5.83	-5.49	-3.92	3.630	10.7338

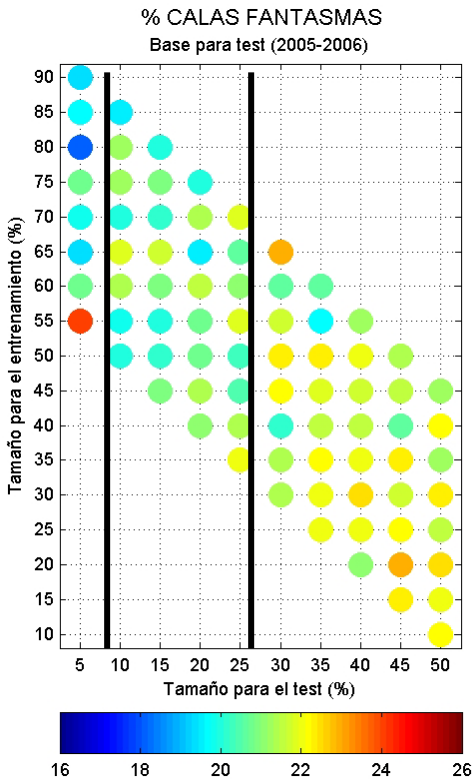
3.10. Porcentaje de variación de los indicadores (de redes con diferentes números de nodos) con respecto a una red con umbral 0.66, entrenada utilizando 4 nodos en la capa escondida (2005-2006).

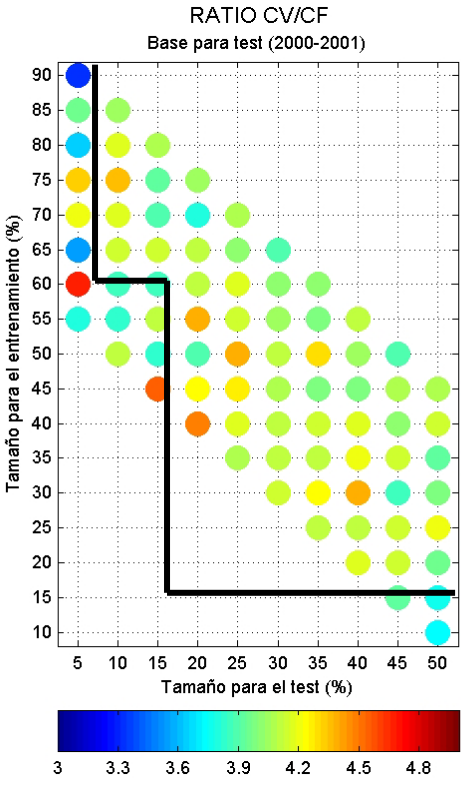
NODOS	CV	CF	RATIO	MSE	STD(MSE)
3	0.24	2.55	-2.25	4.206	7.3528
4	0.11	8.76	-7.95	3.578	42.5646
5	-2.12	-6.94	5.19	0.641	40.5801
6	-2.77	1.16	-3.88	5.444	48.8878
7	-2.48	-0.56	-1.92	6.762	51.4776
8	-3.61	2.48	-5.95	6.133	63.1533

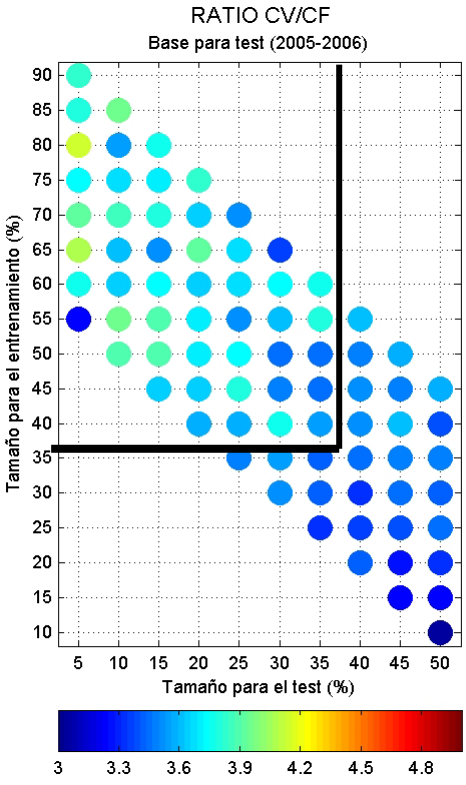
3.11. Indicadores de desempeño de la red (testeando el tamaño de las bases) para la base de test. El eje horizontal corresponde al tamaño de la base para el test y el eje vertical, al tamaño de la base para el entrenamiento. Ambos expresados en porcentaje respecto a la base de datos correspondiente. La tercera dimensión, representada por los colores, corresponde al indicador que se señala en el título de cada gráfica. A la izquierda, resultados para las pruebas con la base 2000-2001 y a la derecha, con la base 2005-2006. Las líneas internas en las gráficas dividiéndolas en regiones, señalan justamente la inclusión o exclusión de ciertas zonas en la "región de eficiencia".

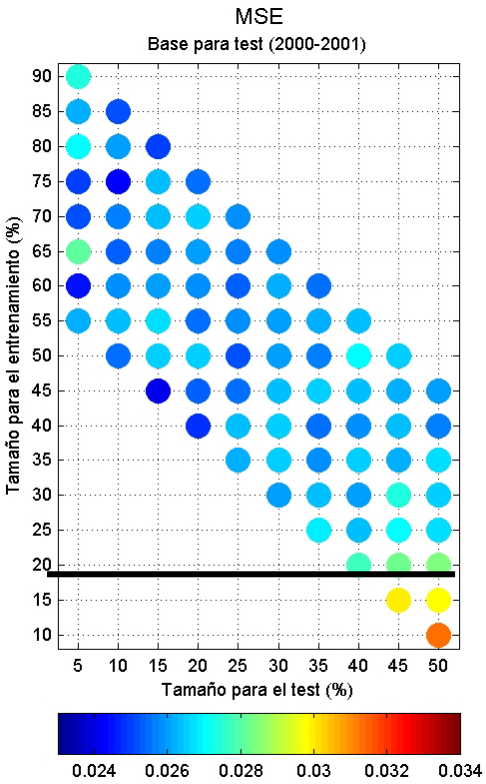


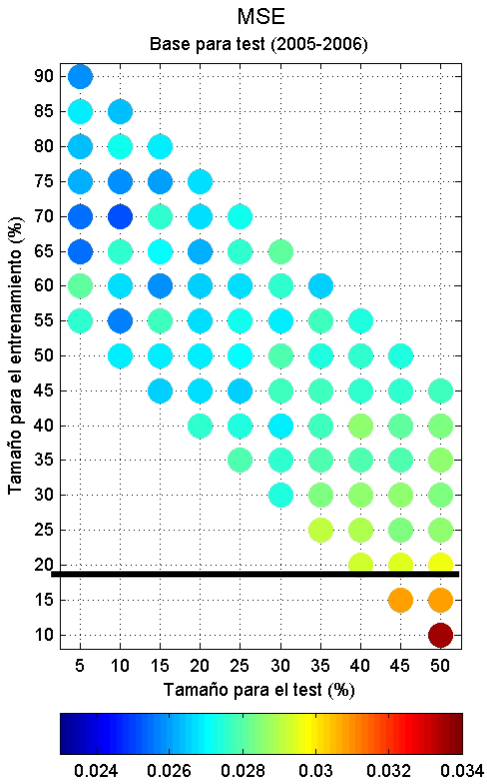

% CALAS FANTASMAS
Base para test (2005-2006)
Tamaño para el entrenamiento (%)
Tamaño para el test (%)
90 85 80 75 70 65 60 55 50 45 40 35 30 25 20 15 10
5 10 15 20 25 30 35 40 45 50
16 18 20 22 24 26

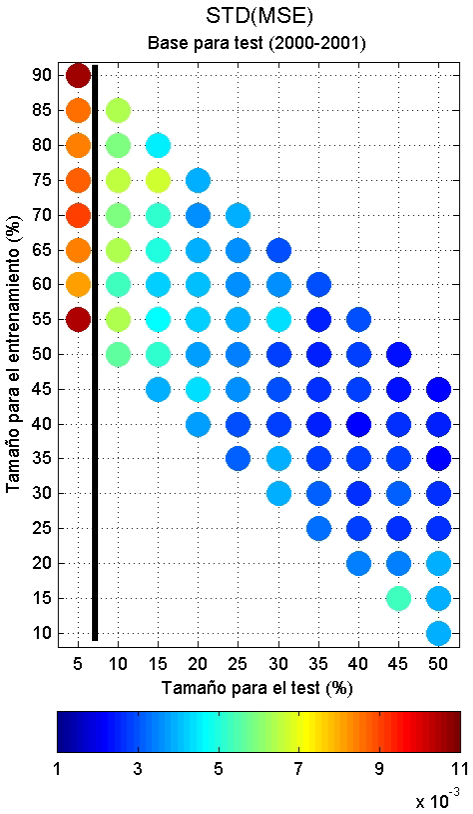

% CALAS FANTASMAS
Base para test (2005-2006)
Tamaño para el entrenamiento (%)
Tamaño para el test (%)
90 85 80 75 70 65 60 55 50 45 40 35 30 25 20 15 10
5 10 15 20 25 30 35 40 45 50
16 18 20 22 24 26

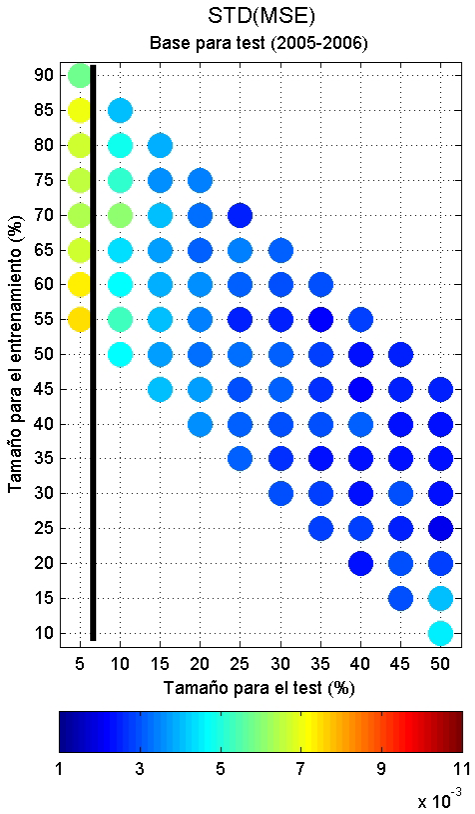

RATIO CV/CF
Base para test (2000-2001)
Tamaño para el entrenamiento (%)
Tamaño para el test (%)
90 85 80 75 70 65 60 55 50 45 40 35 30 25 20 15 10
5 10 15 20 25 30 35 40 45 50
3 3.3 3.6 3.9 4.2 4.5 4.8


RATIO CV/CF
Base para test (2005-2006)
Tamaño para el entrenamiento (%)
Tamaño para el test (%)
90 85 80 75 70 65 60 55 50 45 40 35 30 25 20 15 10
5 10 15 20 25 30 35 40 45 50
3 3.3 3.6 3.9 4.2 4.5 4.8

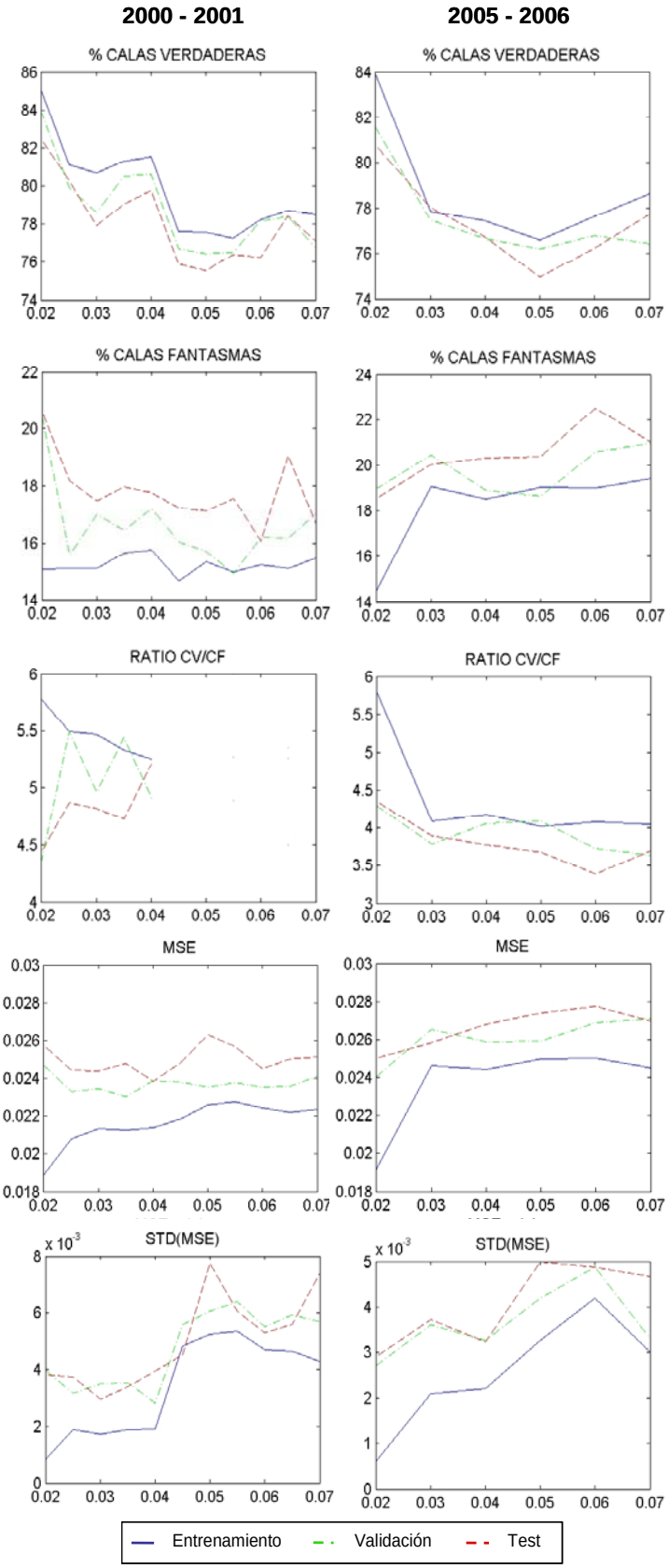

MSE
Base para test (2000-2001)
Tamaño para el entrenamiento (%)
Tamaño para el test (%)
0.024 0.026 0.028 0.03 0.032 0.034


MSE
Base para test (2005-2006)
Tamaño para el entrenamiento (%)
Tamaño para el test (%)
0.024 0.026 0.028 0.03 0.032 0.034


STD(MSE)
Base para test (2000-2001)
Tamaño para el entrenamiento (%)
Tamaño para el test (%)
1 3 5 7 9 11
x 10^-3


STD(MSE)
Base para test (2005-2006)
Tamaño para el entrenamiento (%)
Tamaño para el test (%)
1 3 5 7 9 11
x 10^-3

3.12. Indicadores de desempeño de la red (testeando el MSE máximo entre 0.02 y 0.07). A la izquierda, las gráficas de las pruebas realizadas para la base 2000-2001. A la derecha, para la base 2005-2006. En el eje horizontal están los valores del MSE máximo.



3.13. Porcentaje de variación de los indicadores (de redes con diferentes valores de MSE máximo) con respecto a una red con umbral 0.72, entrenada con una restricción de un máximo de 0.02 para el MSE_max de la base de entrenamiento para cada bucle (2000-2001).

MSE	CV	CF	RATIO	MSE	STD(MSE)
0.02	-2.53	-0.54	-1.15	3.057	26.6472
0.03	-3.34	-12.19	2.57	-5.145	-13.6138
0.04	-4.26	-9.13	0.74	-2.092	21.9676
0.05	-4.54	-8.36	-3.59	3.422	65.9215
0.06	-7.36	-18.06	-----	1.587	78.4255
0.07	-4.63	-5.63	-4.50	3.770	56.0595

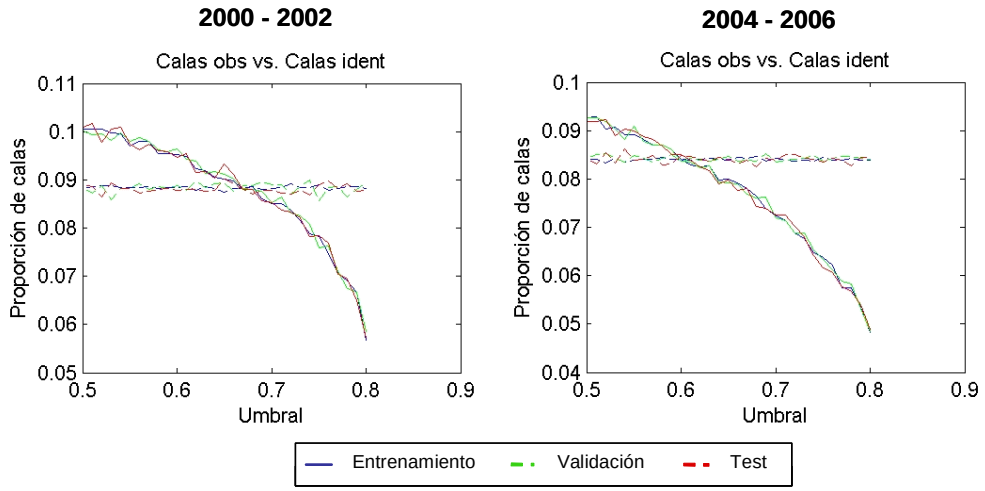
3.14. Porcentaje de variación de los indicadores (de redes con diferentes valores de MSE máximo) con respecto a una red con umbral 0.66, entrenada con una restricción de un máximo de 0.02 para el MSE_max de la base de entrenamiento para cada bucle (2005-2006).

MSE	CV	CF	RATIO	MSE	STD(MSE)
0.02	-0.19	10.96	-10.05	4.308	34.0289
0.03	-3.40	9.57	-11.83	7.435	29.1401
0.04	-3.30	13.21	-14.58	7.013	75.5600
0.05	-2.40	19.91	-18.61	7.167	29.2109
0.06	-7.17	11.15	-16.48	13.122	105.4783
0.07	-10.66	13.35	-21.18	16.116	174.8919

Anexo 4:

Resultados del análisis de sensibilidad – bases trianuales

4.1. Total de calas observadas vs. Total de calas identificadas. Se varió el umbral desde 0.5 hasta 0.8 incrementando el umbral en 0.01 para cada ensayo. Las líneas continuas representan a la proporción de calas observadas y las líneas discontinuas, a la proporción de calas identificadas. A la izquierda, las gráficas de las pruebas realizadas para la base 2000-2002. A la derecha, para la base 2004-2006.



4.2. Valores de los indicadores para el valor inicial de umbral 0.5 (izquierda) y porcentaje de variación de los indicadores (de redes con diferentes umbrales) con respecto a una red con umbral 0.5 (derecha) (2000-2002).

CV	89.74
CF	25.47
RATIO	3.52
MSE	0.024
STD(MSE)	0.003

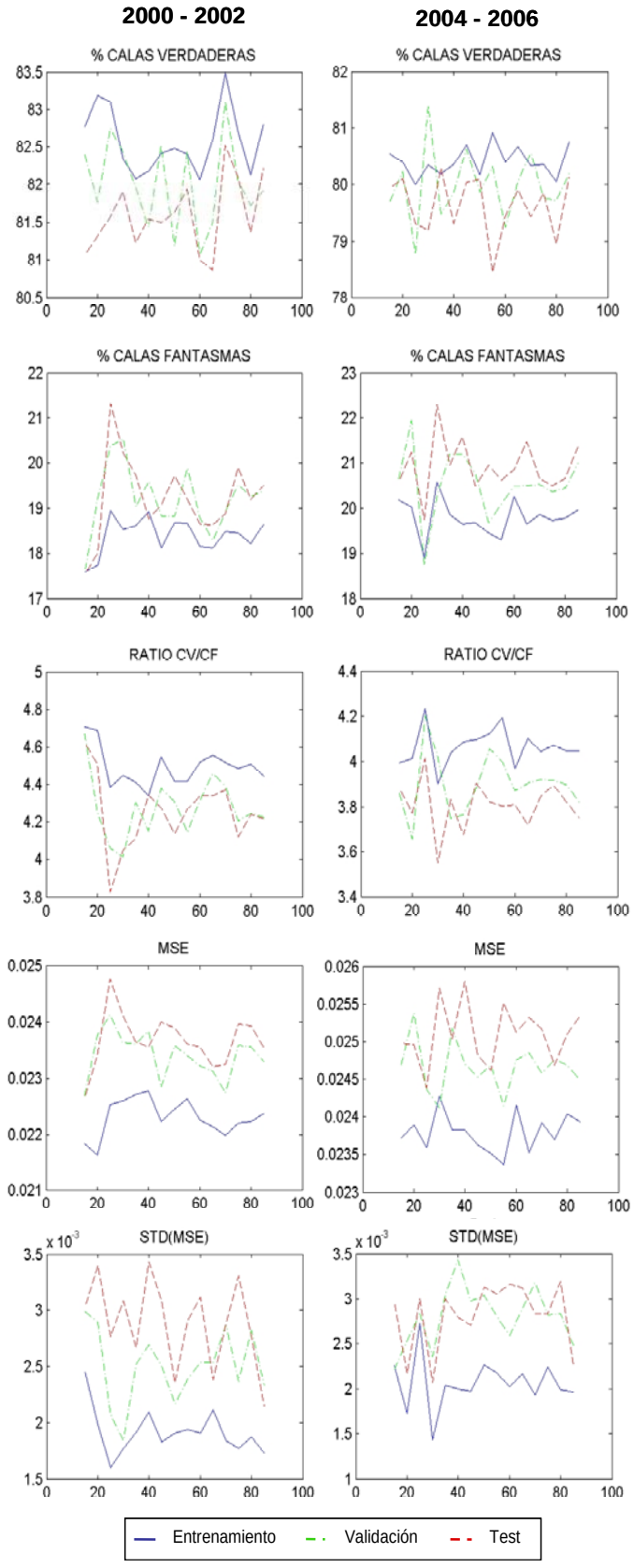
UMBRAL	CV	CF	RATIO	MSE	STD(MSE)
0.55	-2.61	-8.97	6.98	0.70	-6.67
0.6	-4.87	-12.46	8.67	2.26	-20.91
0.65	-6.85	-21.44	18.57	1.90	-10.89
0.67	-9.28	-26.80	23.94	-1.93	-14.89
0.7	-12.33	-28.38	22.41	0.97	-5.57
0.75	-18.86	-39.29	33.65	-0.20	-25.61
0.8	-40.02	-54.86	32.90	2.87	-23.28

4.3. Valores de los indicadores para el valor inicial de umbral 0.5 (izquierda) y porcentaje de variación de los indicadores (de redes con diferentes umbrales) con respecto a una red con umbral 0.5 (derecha) (2004-2006).

CV	85.23
CF	25.27
RATIO	3.37
MSE	0.025
STD(MSE)	0.003

UMBRAL	CV	CF	RATIO	MSE	STD(MSE)
0.55	-3.19	-1.93	-1.28	4.18	26.61
0.6	-6.74	-17.75	13.39	2.87	4.75
0.65	-11.01	-24.94	18.56	3.58	6.40
0.7	-16.86	-37.43	32.88	1.84	10.86
0.75	-29.16	-47.24	34.28	3.85	12.66
0.8	-43.18	-62.52	51.62	2.47	6.90

4.4. Indicadores de eficiencia de la red (testeando los bucles desde 15 hasta 85). A la izquierda, las gráficas de las pruebas realizadas para la base 2000-2002. A la derecha, para la base 2004-2006. El eje horizontal corresponde al número de bucles.



4.5. Valores de los indicadores para una red entrenada con un umbral de 0.67, utilizando 50 bucles (izquierda) y porcentaje de variación de los indicadores (para redes con diferentes números de bucles) en comparación a una red con umbral 0.67 y 50 bucles (derecha) (2000-2002).

CV	81.50
CF	19.35
RATIO	4.21
MSE	0.024
STD(MSE)	0.003

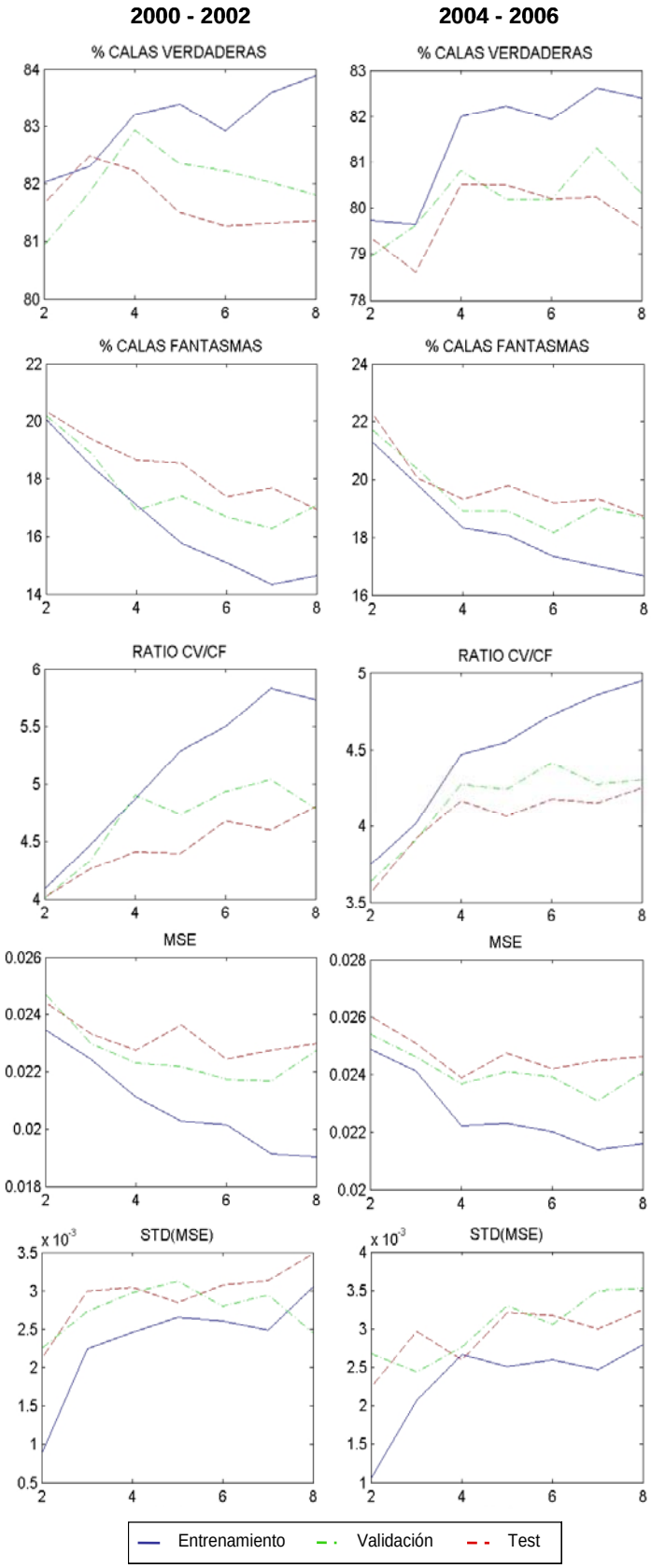
BUCLES	CV	CF	RATIO	MSE	STD(MSE)
20	-0.22	-6.86	7.12	-2.22	21.66
25	0.10	10.17	-9.14	3.39	-0.88
30	0.49	4.49	-3.83	0.70	10.46
35	-0.34	1.96	-2.25	-1.27	-4.51
40	0.06	-3.03	3.19	-1.63	22.89
45	0.00	-1.47	1.49	0.19	10.47
50	0.17	1.96	-1.76	-0.21	-15.72
55	0.53	-0.82	1.36	-1.45	4.12
60	-0.60	-3.61	3.12	-1.64	11.61
65	-0.79	-3.74	3.06	-3.10	-14.77
70	1.25	-2.45	3.80	-2.95	2.96
75	0.73	2.88	-2.09	0.05	18.39
80	-0.17	-0.86	0.70	-0.05	-0.96

4.6. Valores de los indicadores para una red entrenada con un umbral de 0.6, utilizando 50 bucles (izquierda) y porcentaje de variación de los indicadores (para redes con diferentes números de bucles) en comparación a una red con umbral 0.6 y 50 bucles (derecha) (2004-2006).

CV	80.12
CF	20.20
RATIO	3.97
MSE	0.025
STD(MSE)	0.002

BUCLES	CV	CF	RATIO	MSE	STD(MSE)
20	-1.54	4.06	-5.37	1.74	5.64
25	-1.44	4.86	-6.01	0.55	26.67
30	-1.62	2.10	-3.64	6.35	-0.62
35	0.38	2.97	-2.51	-0.72	25.21
40	-2.83	5.76	-8.12	4.20	25.89
45	-0.66	2.93	-3.49	2.02	5.40
50	-0.72	1.22	-1.91	0.07	-13.05
55	-1.21	6.77	-7.47	3.93	20.80
60	-0.37	7.42	-7.25	2.04	17.39
65	-0.52	7.61	-7.55	1.50	28.21
70	0.45	8.09	-7.07	0.39	-2.47
75	-0.02	4.93	-4.71	0.05	16.72
80	-0.48	1.20	-1.65	0.20	26.69

4.7. Indicadores de eficiencia de la red (testeando el número de nodos desde 2 hasta 8). A la izquierda, las gráficas de las pruebas realizadas para la base 2000-2002. A la derecha, para la base 2004-2006. El eje horizontal corresponde al número de nodos escondidos.



4.8. Tiempo que tarda la red para mostrar resultados para diferentes números de nodos en la capa escondida. A la izquierda, para la base 2000-2002 y a la derecha, para la base 2004-2006.

BASE 2000-2002

neuronas	tiempo (m)
2	1.85
3	2.43
4	2.87
5	3.52
6	3.86
7	4.93
8	5.07

BASE 2004-2006

neuronas	tiempo (m)
2	3.95
3	1.97
4	2.69
5	3.77
6	4.46
7	4.47
8	5.11

4.9. Porcentaje de variación de los indicadores (de redes con diferentes números de nodos) con respecto a una red con umbral 0.67, entrenada utilizando 4 nodos en la capa escondida (2000-2002).

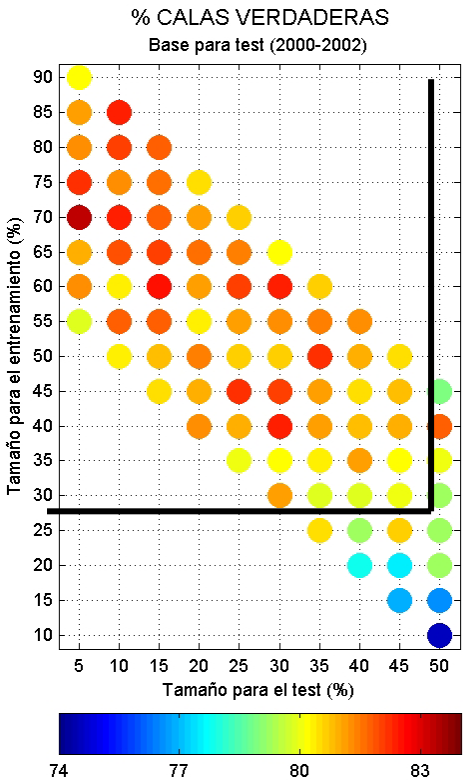
NODOS	CV	CF	RATIO	MSE	STD(MSE)
2	0,18	5,16	-4,73	1,98	-24,05
3	1,23	0,20	1,03	-2,65	7,44
4	0,88	-3,68	4,73	-5,02	8,91
5	0,00	-4,14	4,32	-1,27	1,88
6	-0,29	-10,22	11,06	-6,27	10,17
7	-0,24	-8,58	9,13	-4,96	12,20
8	-0,19	-12,51	14,09	-4,00	24,78

4.10. Porcentaje de variación de los indicadores (de redes con diferentes números de nodos) con respecto a una red con umbral 0.6, entrenada utilizando 4 nodos en la capa escondida (2004-2006).

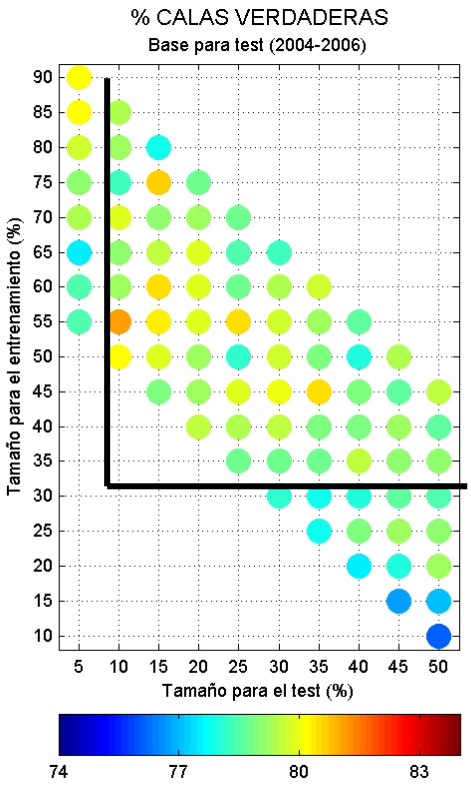
NODOS	CV	CF	RATIO	MSE	STD(MSE)
2	-0.94	10.49	-10.34	5.39	-8.66
3	-1.91	-0.67	-1.25	1.57	20.68
4	0.48	-4.36	5.07	-3.29	5.70
5	0.48	-1.97	2.50	0.14	30.63
6	0.10	-4.92	5.28	-2.08	29.03
7	0.15	-4.30	4.65	-0.86	21.94
8	-0.70	-7.26	7.07	-0.32	32.37

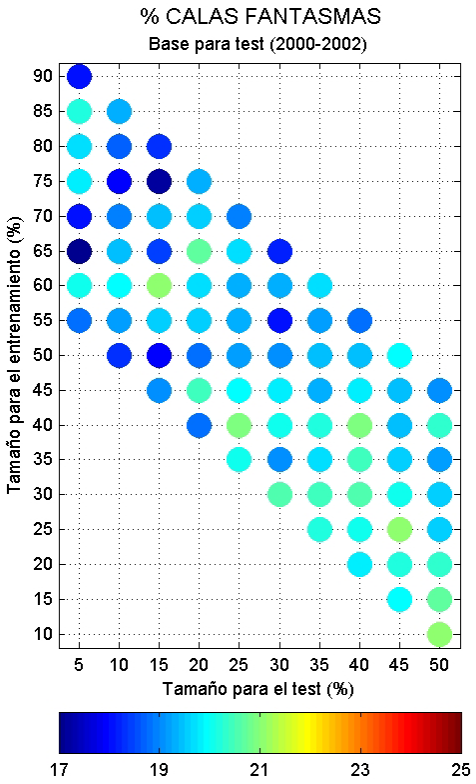
4.11. Indicadores de eficiencia de la red (testeando el tamaño de las bases) para la base de test. El eje horizontal corresponde al tamaño de la base para el test y el eje vertical, al tamaño de la base para el entrenamiento. Ambos expresados en porcentaje respecto a la base de datos correspondiente. La tercera dimensión, representada por los colores, corresponde al indicador que se señala en el título de cada gráfica. A la izquierda, resultados para las pruebas con la base 2000-2002 y a la derecha, con la base 2004-2006. Las líneas internas en las gráficas dividiéndolas en regiones, señalan justamente la inclusión o exclusión de ciertas zonas en la "región de eficiencia".

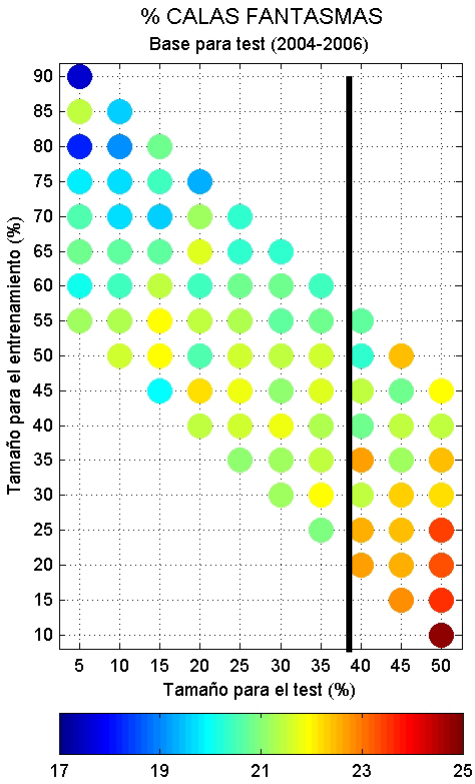
2000 - 2002

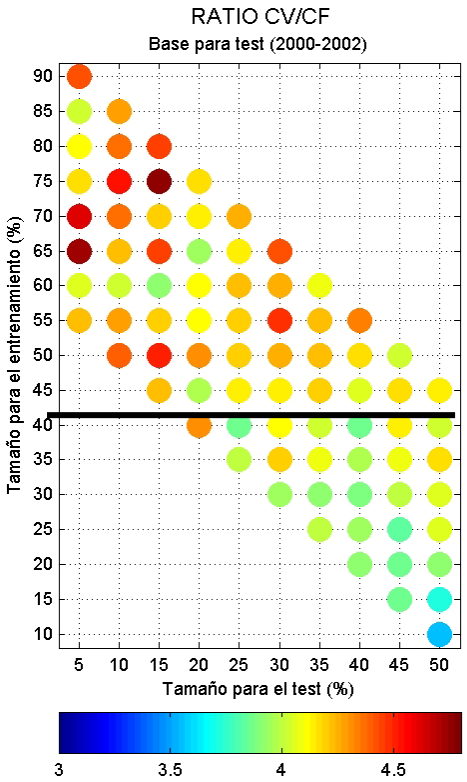


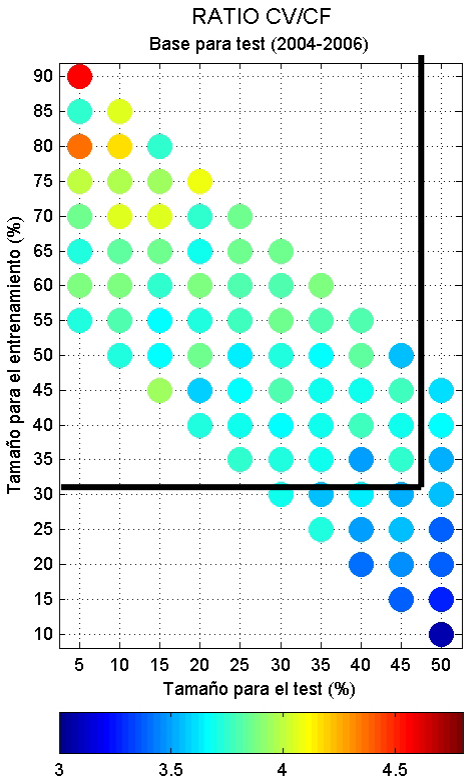
2004 - 2006

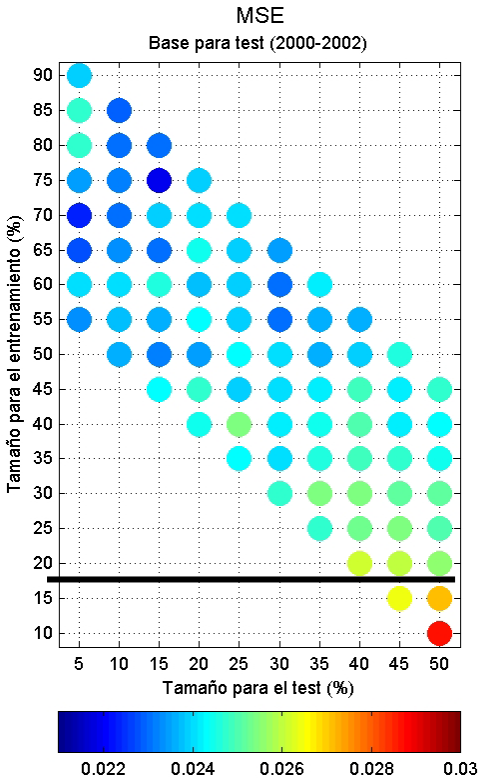


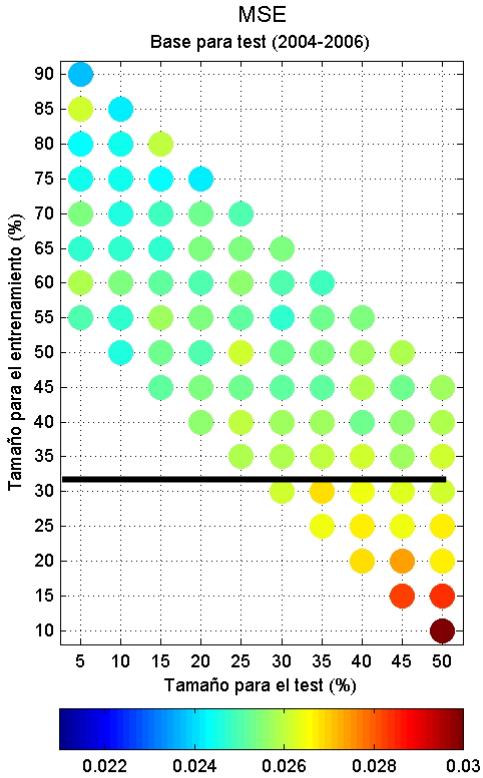

% CALAS FANTASMAS
Base para test (2000-2002)
Tamaño para el entrenamiento (%)
Tamaño para el test (%)
90 85 80 75 70 65 60 55 50 45 40 35 30 25 20 15 10
5 10 15 20 25 30 35 40 45 50
17 19 21 23 25

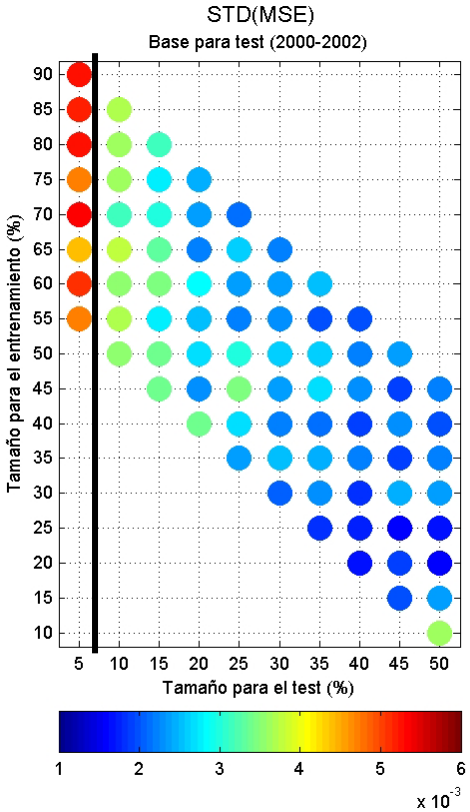

% CALAS FANTASMAS
Base para test (2004-2006)
Tamaño para el entrenamiento (%)
Tamaño para el test (%)
90 85 80 75 70 65 60 55 50 45 40 35 30 25 20 15 10
5 10 15 20 25 30 35 40 45 50
17 19 21 23 25

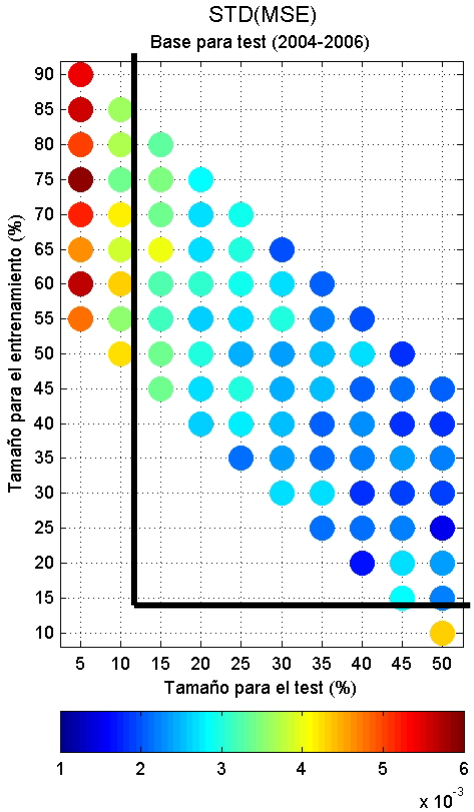

RATIO CV/CF
Base para test (2000-2002)
Tamaño para el entrenamiento (%)
Tamaño para el test (%)
90 85 80 75 70 65 60 55 50 45 40 35 30 25 20 15 10
5 10 15 20 25 30 35 40 45 50
3 3.5 4 4.5


RATIO CV/CF
Base para test (2004-2006)
Tamaño para el entrenamiento (%)
Tamaño para el test (%)
90 85 80 75 70 65 60 55 50 45 40 35 30 25 20 15 10
5 10 15 20 25 30 35 40 45 50
3 3.5 4 4.5

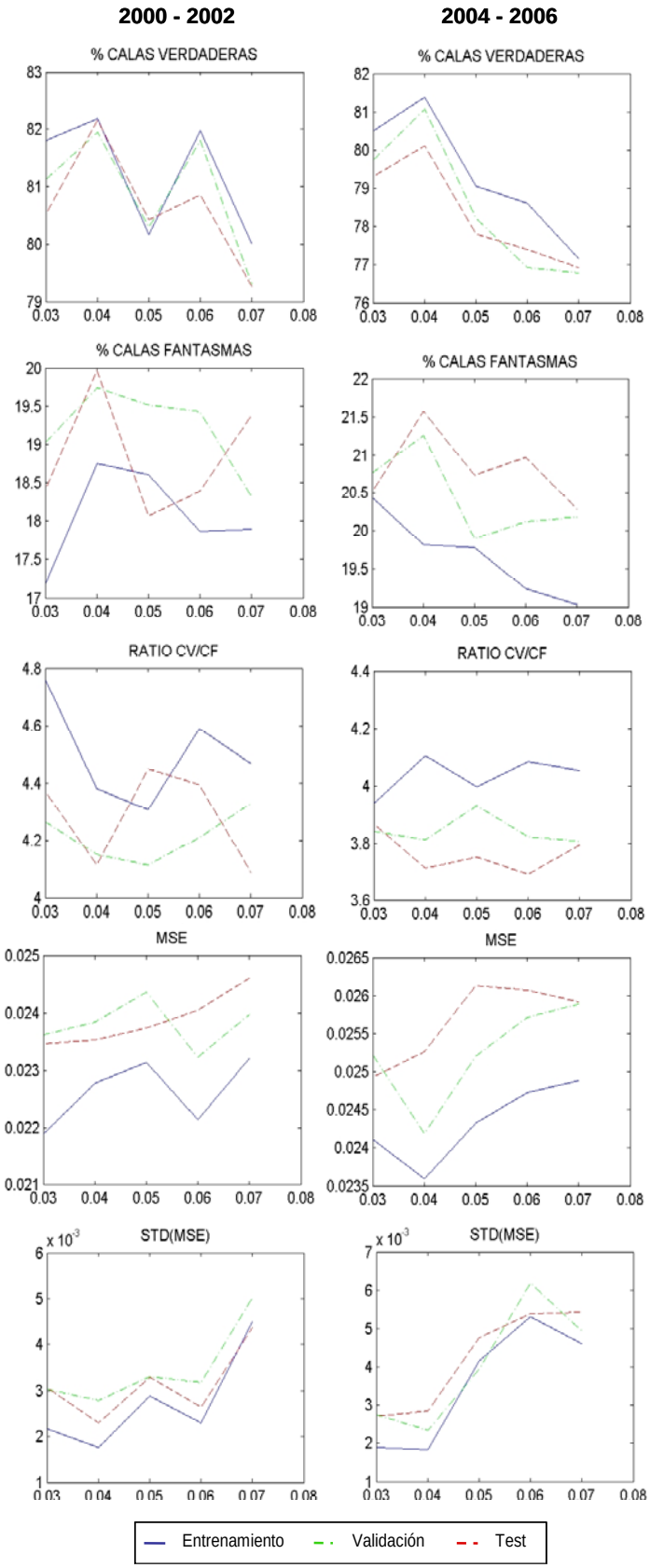

MSE
Base para test (2000-2002)
Tamaño para el entrenamiento (%)
Tamaño para el test (%)
0.022
0.024
0.026
0.028
0.03


MSE
Base para test (2004-2006)
Tamaño para el entrenamiento (%)
Tamaño para el test (%)
0.022
0.024
0.026
0.028
0.03


STD(MSE)
Base para test (2000-2002)
Tamaño para el entrenamiento (%)
Tamaño para el test (%)
1
2
3
4
5
6
x 10^-3


STD(MSE)
Base para test (2004-2006)
Tamaño para el entrenamiento (%)
Tamaño para el test (%)
1
2
3
4
5
6
x 10^-3

4.12. Indicadores de eficiencia de la red (testeando el MSE máximo entre 0.03 y 0.07). A la izquierda, las gráficas de las pruebas realizadas para la base 2000-2002. A la derecha, para la base 2004-2006. El eje horizontal corresponde al MSE máximo.



4.13. Porcentaje de variación de los indicadores (de redes con diferentes valores de MSE máximo) con respecto a una red con umbral 0.67, entrenada con una restricción de un máximo de 0.03 para el MSE_max de la base de entrenamiento para cada bucle (2000-2002).

MSE	CV	CF	RATIO	MSE	STD(MSE)
0.03	-1.20	-4.86	3.85	-2.00	9.65
0.04	0.82	3.18	-2.29	-1.76	-17.90
0.05	-1.31	-6.61	5.68	-0.87	17.77
0.06	-0.78	-4.95	4.39	0.45	-5.55
0.07	-2.74	0.11	-2.85	2.77	56.39

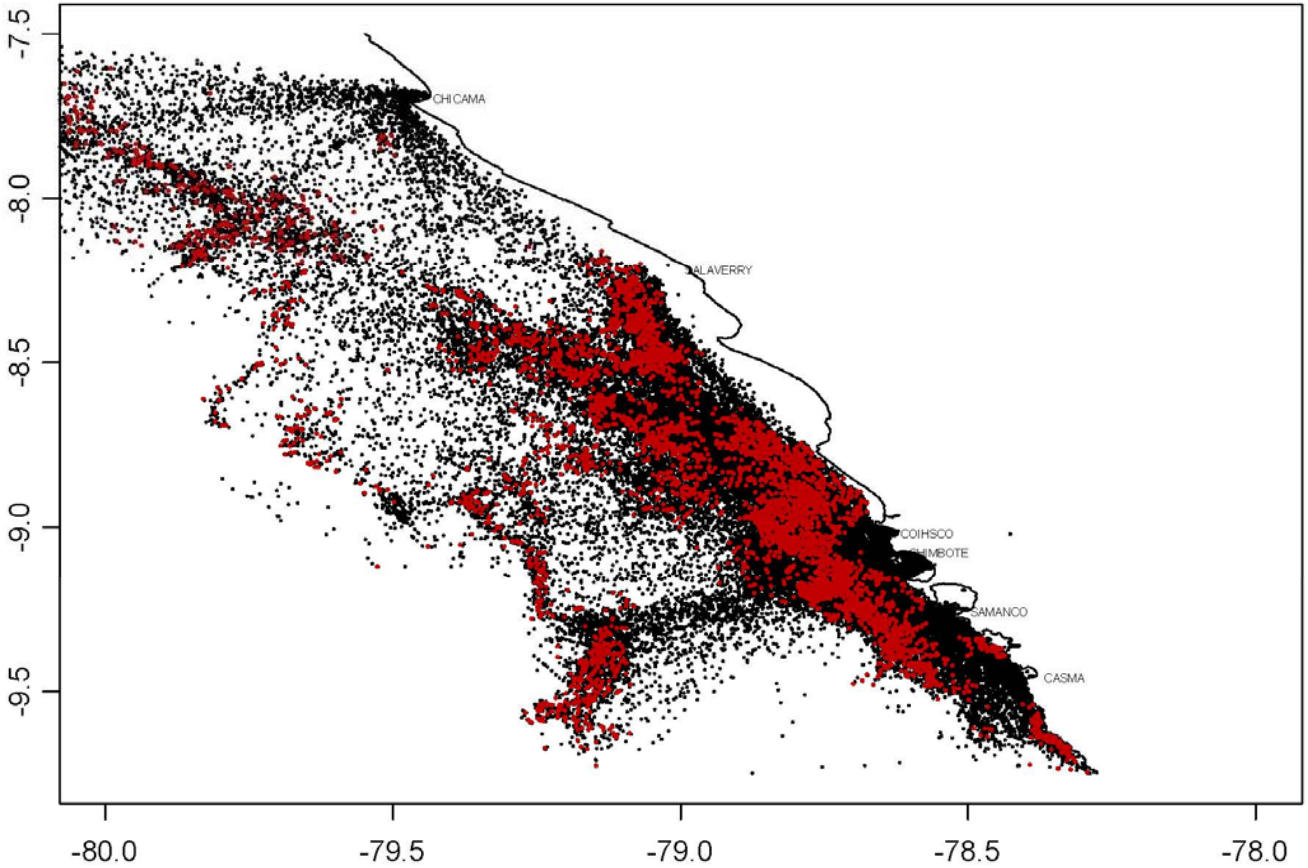
4.14. Porcentaje de variación de los indicadores (de redes con diferentes valores de MSE máximo) con respecto a una red con umbral 0.6, entrenada con una restricción de un máximo de 0.03 para el MSE_max de la base de entrenamiento para cada bucle (2004-2006).

MSE	CV	CF	RATIO	MSE	STD(MSE)
0.03	-1.01	1.55	-2.52	0.88	10.25
0.04	-0.02	6.84	-6.42	2.22	15.99
0.05	-2.88	2.67	-5.40	5.74	93.92
0.06	-3.40	3.82	-6.95	5.52	119.24
0.07	-3.98	0.44	-4.39	4.86	120.55

Anexo 5:

Identificación de calas alrededor de las islas Guañape

5338 calas identificadas por la red neuronal a partir de datos de viajes pesqueros entre los 7ºS y 10ºS, recolectados entre el 17 y el 30 de noviembre y luego entre el 6 y el 15 de diciembre del 2007, para 393 barcos. Los puntos negros son ubicaciones de los barcos obtenidos por el VMS y los rojos, los puntos de pesca.



Anexo 6:

Resultados de la regresión logística

6.1. Test de bondad de ajuste Hosmer and Lemeshow para cada base temporal por método de introducción de variables (p-valor)

MÉTODO DE INTRODUCCIÓN DE VARIABLES	BASE				
	2000-2001	2000-2002	2005-2006	2004-2006	2000-2006
INTRODUCIR	0.411	0.256	0.330	0.049	0.000
SELECCIÓN HACIA DELANTE (WALD)	0.116	0.256	0.478	0.028	0.000
ELIMINACIÓN HACIA ATRÁS (WALD)	0.437	0.256	0.478	0.028	0.000

6.2. Porcentaje de clasificación correcta de calas para cada base temporal por método de introducción de variables

MÉTODO DE INTRODUCCIÓN DE VARIABLES	BASE				
	2000-2001	2000-2002	2005-2006	2004-2006	2000-2006
INTRODUCIR	45.5%	42.8%	40.7%	40.3%	40.5%
SELECCIÓN HACIA DELANTE (WALD)	44.5%	42.8%	40.7%	40.4%	40.1%
ELIMINACIÓN HACIA ATRÁS (WALD)	46.0%	42.8%	40.7%	40.4%	40.1%

Anexo 7:

Interfaz gráfica en Matlab