UNIVERSIDAD AUTÓNOMA DEL ESTADO DE MÉXICO

CENTRO UNIVERSITARIO UAEM TEXCOCO

Análisis de la complejidad de los datos y su efecto en las redes neuronales artificiales

TESIS

QUE PARA OBTENER EL GRADO DE

MAESTRO EN CIENCIAS DE LA COMPUTACIÓN

PRESENTA:

PRIMITIVO TORIBIO LUNA

TUTOR ACADÉMICO:

DR. ROBERTO ALEJO ELEUTERIO

TUTORES ADJUNTOS:

DRA. ROSA MARÍA VALDOVINOS ROSAS
DR. RENÉ ARNULFO GARCÍA HERNÁNDEZ

TEXCOCO, ESTADO DE MÉXICO. MAYO 2011.

Universidad Autónoma del Estado de México
UAEM

COPIA

DICTÁMEN PARA AUTORIZACIÓN DE GRADO DE MAESTRÍA

Texcoco, Méx. , a 26 de Abril de 2011

TÍTULO DEL PROYECTO:

Análisis de la complejidad de los datos y su efecto en las redes neuronales artificiales

TESISTA:

Primitivo Toribio Luna

DICTAMEN: **NO. DE REVISIÓN:** 3

- ☐ **RECHAZADO**
- ☐ **SUJETO A MODIFICACIONES**
- ☐ **ACEPTADO, CONDICIONADO**
- ☑ **ACEPTADO**

OBSERVACIONES GENERALES: sin observaciones.

Aceptado para la impresión

Aceptado para la defensa de grado

TITULACION
RECIBIDO
Por
Texcoco, Méx., a 4 de 11

Tutor Académico	Tutor Adjunto	Tutor Adjunto
Dr. Roberto Alejo Eleuterio	Dra. Rosa María Valdovinos Rosas	Dr. René Arnulfo García Hernández

Agradecimientos

A mis padres y hermanos por su confianza y apoyo en todo momento.

De una manera muy especial quiero agradecer la paciencia, apoyo incondicional y amistad de mi asesor, el Dr. Roberto Alejo. A la Dra. Rosa María Valdovinos por su apoyo en todo momento y sus sabios consejos. Al Dr. Rene Arnulfo García por su tiempo y apoyo incondicional para culminar este proyecto.

A Paulina, mi novia y compañera que siempre estuvo dispuesta a ayudarme en lo que necesitara. A mis compañer@s de la maestría, principalmente a Flor del Rió y Bianca por su compañía y apoyo.

Agradezco a las instituciones que me apoyaron económicamente para la culminación de este proyecto. Al Consejo Mexiquense de Ciencia y Tecnología (COMECYT) por el financiamiento para la elaboración de la presente Tesis. A la Universidad Autónoma del Estado de México por su confianza al otorgarme una beca para cursar la Maestría. También al proyecto 2933/2010 de la UAEM al ayudarme con la publicación de los artículos derivados de esta investigación.

Índice general

Lista de Figuras

Lista de Tablas

Capítulo 1

Introducción

1.1 Redes neuronales artificiales

Las redes neuronales artificiales (RNA) gozan de gran popularidad entre teóricos y especialistas en el área de aprendizaje automático, minería de datos y reconocimiento de patrones. Una de las principales razones de este auge, es que las RNA están dotadas de algoritmos que les permiten aprender a partir de ejemplos o datos de entrada [Russell 1996], por lo que no precisan para su funcionamiento de un modelo *a priori* del problema a tratar.

Actualmente, existen numerosos modelos de RNA entre los más usados se encuentran las redes de Funciones de Base Radial (*Radial Basis Function, RBF*) y el Perceptrón Multicapa (*Multilayer Perceptron, MLP*), aunque también han tomado auge las maquinas de vectores soporte (*Support vector Machine, SVM*).

Las redes RBF y MLP se han utilizado en tareas de clasificación, aproximación de funciones, modelado y problemas de control [Ding 2004]. Ambas redes son consideradas de propagación hacia adelante (*feedforward*) y comparten varias características en común [Hutchinson 1995]. Por ejemplo, son aproximadores universales [Haykin 1999], modelos con capas no lineales [Looney 1997] o pueden ser entrenadas con métodos similares de descenso por gradiente [Schwenker 2001].

La principal diferencia entre las redes RBF y los MLP está en la función de activación de los nodos ocultos [Ding 2004]. En RBF depende de la distancia entre los vectores de entrada y los centros de la red, mientras que en MLP depende del producto del vector de entrada y el vector de pesos.

No obstante, aún se conoce poco de estos modelos, lo que se traduce en debilidades tales como la lentitud en el aprendizaje y la pobre capacidad de generalización que se observa en un número importante de aplicaciones prácticas.

Numerosos estudios se han desarrollado para mejorar el desempeño del MLP. En [Anand 1993], se propone una modificación al algoritmo backpropagation para acelerar la convergencia de la red en problemas de dos clases y en [Anand 1995] se extiende

el estudio a problemas de múltiples clases. Otras alternativas para acelerar la convergencia de la red se presentan en [Jacobs 1988] y [Fahlman 1988]. En [Lippmann 1988], [Cybenko 1989], [Russell 1996], [Funahashi 1989], se ha estudiado la capacidad de representación del MLP, en relación al número de capas ocultas, mientras que en [Pao 1989], [Kanellopoulos 1997], se ha discutido la estimación del número óptimo de nodos para las capas ocultas.

Por su parte, las redes RBF han sido estudiadas desde diferentes enfoques. Se ha propuesto la construcción de redes RBF a partir de otros modelos, como por ejemplo las máquinas de vectores soporte (Support Vector Machine, SVM) [Schölkopf 1997] o los árboles de decisión [Kubat 1998]. En [Xu 1998] las redes RBF son presentadas como casos especiales del modelo *Alternative mixture of experts (ME)* [Xu 1995] y los parámetros de la capa oculta y la de salida son obtenidos por el algoritmo *expectation-maximization (EM)* [Jordan 1994], [Jordan 1995]. Otra tendencia es la utilización de algoritmos genéticos para la determinación del número de centros, así como su localización y varianzas, o la obtención de todos los parámetros de la red [Harpham 2004]. En [Uykan 2000], se estudia la relación entre el rendimiento de la red y la forma en como se establecen los centros de la misma.

Durante mucho tiempo ha prevalecido que la selección y/o transformación de las variables o atributos (reducción de la dimensión) era suficiente para tener un modelo supervisado óptimo. No obstante, en la actualidad está tomando auge la idea de que es necesario prestar mucha más atención a la calidad de la muestra de entrenamiento [Visa 2005, Barandela 2001, Barandela 2000], entendiéndose por esto la preocupación por la representatividad de sus elementos y por el grado de confianza en sus etiquetas. Esto reviste mayor importancia cuando se trabaja con métodos como la regla del vecino mas cercano (Nearest Neighbor, NN), las redes RBF o MLP. Ocurre con estos métodos que, precisamente por apoyarse principalmente en la información suministrada por la muestra de entrenamiento, son sensibles a cualquier deficiencia en la calidad y confiabilidad de ésta.

Una de las principales razones que contribuye al bajo rendimiento de una red neuronal, es la falta de representatividad de los datos de entrenamiento. Por ejemplo, cuando existen considerables desproporciones en el número de patrones de las distintas clases [Murphey 2004](desbalance de clases), solapamiento entre clases [Alejo 2006], ruido[1] en los datos de entrenamiento, patrones atípicos[2] [Barandela 2002], muestras de entrenamiento incompletas, es decir, cuando alguna de las clases existentes no ha sido representadas suficientemente por patrones de entrenamiento [Barandela 2001].

Recientemente, el problema del desbalance de clases se ha reconocido como un problema fundamental en el aprendizaje automático y minería de datos [Zhou 2006], pues la mayoría de los métodos de aprendizaje funcionan mal con este tipo de bases de datos. Lamentablemente, en el mundo real hay un gran número de casos que tienen este pro-

[1]Datos con errores, originados en su medición o registro.

[2]Excepciones a la regla, es decir, que no encaja en un tipo o modelo.

blema [Japkowicz 2002], esto es principalmente por la naturaleza de los datos como la detección de fraudes en llamadas telefónicas [Fawcett 1997], en transacciones con tarjeta de crédito [Chan 1999], al tratar con enfermedades pues puede haber una clase que represente un enfermedad rara (supóngase que esta es la que les interesa a los expertos en el área), y por consiguiente esta pobremente representada (pequeño número de patrones). También, surge este problema porque es demasiado caro obtener datos para la clase minoritaria. En el contexto del MLP, entrenado con el algoritmo backpropagation, el problema se ha formulado como sigue: la clase mayoritaria domina el proceso de entrenamiento y los elementos de la clase menos representada o minoritaria pueden ser ignorados, así la convergencia de la clase minoritaria es muy lenta [Anand 1993], [Murphey 2004], [Bruzzone 1997a], [Bruzzone 1997b], [Lu 1998].

Por otro lado, en los últimos años las maquinas de vectores soporte (SVM) se han popularizado como un método alternativo de clasificación debido a su buen fundamento teórico y su buena capacidad de generalización [Burges 1998]. Las SVM mapean los puntos de entrada (patrones) a un espacio de características altamente dimensional para luego encontrar el hiperplano que los separe y maximice el margen entre las clases. Las SVM han sido aplicadas exitosamente a un gran número de aplicaciones yendo desde identificación de partículas, identificación de rostros y categorización de texto hasta bioinformática y medicina.

Un ventaja notable de las SVM radica en que al finalizar la fase de aprendizaje solo una pequeña parte del conjunto original (conjunto de vectores soporte) representa la tarea de clasificación dada. Sin embargo, para encontrar el hiperplano óptimo de separación las SVM necesitan resolver un problema de programación cuadrática (QP), que involucra una matriz de densidad de NxN, donde N es el numero de patrones en la ME [Platt 1999]. Como tiene una complejidad cuadrática, se requieren gran cantidad de tiempo computacional y memoria para grandes conjuntos de datos.

1.2 Objetivos de la tesis

En este trabajo, se busca una definición para complejidad de los datos que involucre aquellas características de la muestra de entrenamiento que afecten el aprendizaje de la red neuronal. También, con base en lo anterior se analizan algunas propuestas para mejorar la efectividad de las redes RBF, MLP y SVM (esta ultima solo con fines comparativos y de exploración). Para cumplir con el objetivo propuesto se desarrollaran las siguientes actividades.

1. Obtener información acerca de la naturaleza de los datos cuando se realiza el proceso de entrenamiento, es decir, cuáles son las características que afectan el comportamiento de una RNA.

2. Evaluar diversas estrategias para reducir la influencia de algunas características definidas dentro de complejidad de los datos sobre la efectividad del clasificador.

Por lo extenso de la investigación solo se aborda parte del concepto de complejidad de los datos.

En resumen, se pretende analizar el comportamiento del error que se genera en redes neuronales artificiales ante diferentes situaciones, con el objetivo de buscar estrategias que mejoren la precisión de este tipo de clasificadores. En aquellas situaciones en que esto no sea posible, se analizarán otras posibles reglas de decisión que mejoren la solución del problema. Para ello, diferentes problemas reales y bases de datos se someterán a prueba. En el capítulo 2 se da una descripción de las redes neuronales artificiales que es el enfoque del reconocimiento de patrones (RP) sobre el cual se hará la investigación, específicamente de las redes de funciones de base radial, perceptrón multicapa y maquinas de vectores soporte. En el capítulo 3 se habla sobre la complejidad de los datos y las técnicas de edición que se utilizaran para disminuir este tipo de complejidad. En el capítulo 4 se habla sobre la metodología que se uso para realizar la experimentación y las métricas que su utilizaron para medir los resultados. Los resultados de la experimentación se analizan en el capítulo 5. Por ultimo en el capítulo 6 se concluye el trabajo puntualizando las aportaciones de la presente investigación, trabajos futuros y líneas abiertas.

Capítulo 2

Redes Neuronales Artificiales

En este capitulo se describen las principales característica y aplicaciones de una red neuronal artificial, así como, el funcionamiento de los modelos MLP, RBF y SVM que son los clasificadores sobre los cuales se probaron todos los experimentos realizados en este proyecto.

2.1 Inspiración biológica

Los sistemas biológicos procesan información de forma no convencional, no requieren modelos de referencia, y se desempeñan exitosamente en presencia de incertidumbre; aprenden a realizar nuevas tareas y se adaptan con facilidad a ambientes cambiantes. Se dice que un sistema tiene la capacidad de aprender si adquiere y procesa información acerca de su desempeño y del ambiente que lo rodea, para mejorar dicho desempeño [Sánchez 2006].

La idea que impulsó el desarrollo de las Redes Neuronales Artificiales (RNA) ha sido la de imitar al sistema de procesamiento más complejo que se conoce hasta ahora, el cerebro humano. La teoría y modelado de redes neuronales está inspirada en la estructura y funcionamiento de los sistemas nerviosos, donde la neurona es el elemento fundamental.

Una neurona es una célula viva que consta de un cuerpo (soma) más o menos esférico, de 5 a 10 micras de diámetro, del que sale una rama principal, el axón, y varias ramas más cortas, llamadas dendritas. Una de las características de las neuronas es su capacidad de comunicarse. En términos generales las dendritas y el cuerpo celular reciben señales de entrada; el cuerpo celular las combina e integra y emite señales de salida (ver Fig. 2.1). El axón transmite dichas señales a los terminales axónicos, que distribuyen información a un nuevo conjunto de neuronas.

El proceso de comunicación se da a traves de unos espacios de interconexión llamados sinapsis [Russell 1996]. La comunicación entre neuronas es el resultado de la liberación de unas sustancias electro-químicas llamadas neurotransmisores (señales). Esta señal

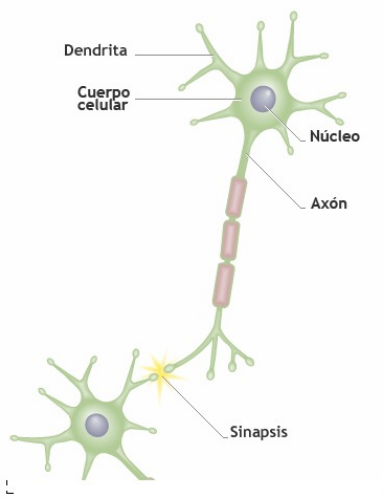


Figura 2.1: Neurona Biológica.

generada por la neurona y transportada a lo largo del axón es un impulso eléctrico, mientras que la señal que se transmite entre los terminales axónicos de una neurona y las dendritas de la otra es de origen químico [Haykin 1999]. En el caso de las neuronas receptoras su axón envía señales hacia el cerebro, por el contrario, las neuronas motoras mandan señales desde el cerebro hacia el resto del cuerpo (para información más detallada se puede consultar [Brunak 1990]).

2.1.1 Redes neuronales

Están constituidas por un gran número de neuronas, conectadas en forma masiva. Conforman el sistema nervioso y el cerebro; el cerebro humano puede contener 10^{11} neuronas y 10^{15} interconexiones [Sánchez 2006]. Las redes neuronales biológicas pueden establecerse como grupos de neuronas activas especializadas en tareas como: cálculos matemáticos, posicionamiento y memoria [Simpson 1990].

Es importante notar que aunque el tiempo de conmutación de la neurona es mucho menor que los elementos de las computadoras [Jain 1996], el cerebro humano tiene la capacidad de realizar tareas miles de veces más rápido que las actuales supercomputadoras. Esta versatilidad es utilizada por el cerebro humano a través de la información que recibe por medio de los sentidos para interpretar su entorno: reconocimiento de rostros humanos, comunicación mediante lenguaje natural, etc.

2.2 Historia de las redes neuronales artificiales

En la antigüedad Platón (427-347 a.C) y Aristóteles (384-422 a.C) dieron a conocer las primeras explicaciones teóricas sobre el cerebro y el pensamiento. Las mismas ideas sobre el proceso mental las mantuvieron Descartes (1596-1650) y los filósofos empiristas del siglo XVIII [Hilera 1995]. Sin embargo, la historia de las RNA comenzó con el científico aragonés Santiago Ramón y Cajal descubridor de la estructura neuronal del sistema nervioso [Martín 2001].

Más adelante, en 1943 el neurofisiólogo Warren McCulloch y el matemático Walter Pitts [McCulloch 1943] conciben una teoría acerca de la forma de trabajar de las neuronas artificiales. En 1949 Donald Hebb [Hebb 1949] pública el libro "The organization of behavior - a neurophysiological theory" en el que se establece la conexión entre la Fisiología y la Psicología. Propone una ley de aprendizaje que le permite explicar cualitativamente algunos ejemplos experimentales de carácter psicológico.

En 1958 Rosenblatt [Rosenblatt 1958] publica "The Perceptron: a probabilistic model for information storage & organization in the brain" donde se aborda uno de los modelos neuronales de mayor impacto en los inicios de la neurocomputación. Rosenblatt opinaba que la herramienta de análisis más apropiada era la teoría de probabilidades. Esta teoría se basa en la separabilidad estadística para caracterizar las propiedades más visibles de estas redes de interconexión ligeramente aleatoria.

En 1959 Bernard Widrow y Marcial Hoff [Widrow 1960] desarrollan el modelo ADALINE (ADAptative LINear Elements) equipado con una potente algoritmo de aprendizaje. Es la primera red neuronal aplicada a un problema real[1].

Con la publicación de "Perceptrons: An Introduction to Computational Geometry"[2] por Marvin Minsky y Seymour Papert del Instituto de Tecnología de Massachussets surgen numerosas críticas que frenaron hasta 1982 el crecimiento que estaban experimentando las investigaciones sobre RNA. A pesar de esto algunos investigadores continuaron avanzando en este campo del conocimiento. James Anderson planteó un modelo lineal llamado Asociador Lineal y en 1977 diseña una potente extensión a este modelo llamado Brain-State-in-a-Box (BSB) [Anderson 1977]. Kunihiko Fukushima [Fukushima 1979] desarrolla el Neocognitron, un modelo de red neuronal para el reconocimiento de patrones visuales. Teuvo Kohonen [Kohonen 1977] propone un modelo similar al de Anderson de forma independiente.

En 1982 coinciden numerosos eventos que hacen resurgir el interés por las RNA. John Hopfield [Hopfield 1982] presenta su trabajo sobre RNA en la Academia Nacional de Ciencias de los Estados Unidos de América[3]. En este trabajo escribe con claridad y rigor matemático una red a la que se ha dado su nombre, pero además, muestra cómo tales

[1] Fue usada comercialmente durante varias décadas como filtros adaptativos para eliminar ecos en líneas telefónicas.

[2] En este trabajo se presentan las limitaciones del Perceptrón de Rosenblatt. Principalmente la imposibilidad de este modelo de resolver problemas no linealmente separables.

[3] National Academy of Sciences of the United States of America.

RNA pueden trabajar de forma óptima. En 1987 Stephen Grossberg [Grossberg 1987] aporta importantes innovaciones con su modelo ART (Adaptative Resonance Theory) y junto a Michel Cohen elabora un importante teorema sobre la estabilidad de las RNA recurrentes en términos de una función de energía. La publicación de "PDP Books"[4] editados por David Rumelhart y James Mc Clelland supone un verdadero acontecimiento por la presentación del método de retropropagación ("back-propagation") [Rumelhart 1986].

Por otra parte, en 1982 se celebra la U.S.-Japan Joint Conference on Cooperative/Competitive Neuronal Networks, y la compañía Fujitsu comienza el desarrollo de computadoras con capacidad de aprendizaje para aplicaciones en robótica.

En 1986 el American Institute of Physics (AIP) inicia lo que ha sido la reunión anual Neural Networks for Computing. En 1987 el IEEE[5] celebra la primera conferencia internacional sobre RNA. Ese mismo año se forma la International Neural Networks Society (INNS). En 1988 surge la International Joint Conference on Neural Networks (IJCNN) como resultado de la unión entre la IEEE y la INNS.

La alternativa europea es la International Conference on Artificial Neural Networks (ICANN) surgida en septiembre de 1991. También merece una referencia aparte la reunión anual Neural Information Processing System (NIPS) celebrada en Denver (Colorado) desde 1987.

2.3 Neurona artificial

La neurona es la unidad de proceso de información fundamental en una red neuronal. El modelo de neurona artificial más conocido es el de MCCulloch-Pitts y su teoría se basa en cinco suposiciones[Freeman 1991]:

1. La actividad de una neurona es un proceso de todo o nada.
2. Es preciso que un número fijo de sinapsis(> 1) sean excitadas dento de un periodo de adición latente para que se excite una neurona.
3. El único retraso significativo dentro de un sistema nervioso es el retardo sináptico.
4. La actividad de cualquier sinapsis inhibitoria impide por completo la excitación de la neurona en ese momento.
5. La estructura de la red de interconexiones no cambia con el transcurso del tiempo.

En la Figura 2.2 se ilustra el modelo de la neurona artificial de McCulloch-Pitts. De este modelo se pueden identificar cinco elementos básicos [McCulloch 1943]:

1. Conjunto de x señales de entrada, que suministran a la red la información del entorno.

[4]Parallel Distributed Processing, Vol. I and II.
[5]Institute of Electrical and Electronics Engineers.

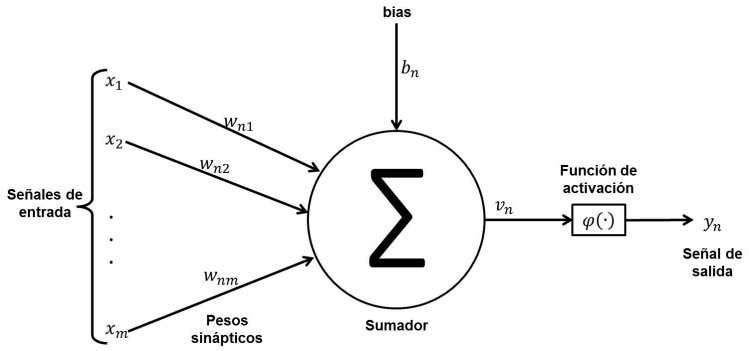


Figura 2.2: Modelo de una neurona artificial.

2. Vector de pesos sinápticos w. El peso w_{nm} está asociado a la sinapsis que conecta la entrada *m-esima* con la neurona *n-esima*.

3. Un umbral o bias (b_n), que aumenta la capacidad de procesamiento de la neurona y que eleva o reduce la entrada a la neurona, según sea su valor positivo o negativo.

4. Un sumador o integrador (Σ) que suma las señales de entrada, ponderadas con sus respectivos pesos (w_{nm}) y el bias.

5. Una función de activación $\varphi(\cdot)$ que suele limitar la amplitud de la salida de la neurona (y_n).

En términos matemáticos, es posible describir la neurona n de la Figura 2.2 por el siguiente par de ecuaciones:

$$u_n = \sum_{j=1}^{m} w_{nj} x_j \tag{2.1}$$

y

$$y_n = \varphi(u_n + b_n) \tag{2.2}$$

donde $x_1, x_2, ..., x_m$ son las señales de entrada; $w_{n1}, w_{n2}, ..., w_{nm}$ son los pesos sinápticos de la neurona n; u_n es la combinación lineal de las entradas ponderadas por los pesos sinápticos; b_n es el umbral o bias; $\varphi(\cdot)$ es la función de activación; y finalmente, y_n es la señal de salida de la neurona.

El umbral o bias es un parámetro externo de la neurona n, pero es posible considerarla como parte de las señales de entrada, de tal forma que si se combinan (2.1) y (2.2) se tiene:

$$u_n = \sum_{j=0}^{m} w_{nj} x_j \tag{2.3}$$

y

$$y_n = \varphi(v_n) \tag{2.4}$$

A v_n se le denomina potencial de activación. En la ecuación 2.3 se le a agregado una nueva sinapsis. Su entrada es $x_0 = +1$ y el peso correspondiente es $w_{n0} = b_n$.

La función de activación es la que define en última instancia la salida de la neurona. En la Tabla 2.1 se muestra una analogía entre las neuronas biológicas y la neurona artificial de McCulloch-Pitts.

Tabla 2.1: Analogía entre las neuronas biológicas y las artificiales.

Neurona Biológica	Neurona Artificial
Sinapsis	Pesos sinápticos (w)
Axón	Respuesta de la neurona y_n (función de activación $\varphi(\cdot)$)
Dendritas	Señales de entrada (x)
Soma	Sumador o integrador (Σ)

El modelo de McCulloch-Pitts asume varias suposiciones que no reflejan el verdadero comportamiento de las neuronas biológicas [Jain 1996], sin embargo, este modelo ha sido generalizado de diferentes formas. Actualmente, la función de activación puede admitir distintos modelos, entre los mas comunes podemos mencionar las siguientes:

- **Función lineal.** Se utiliza cuando no se desea acotar la salida de la neurona.

$$f(net) = c \cdot net, \tag{2.5}$$

  donde c es un valor constante.

- **Función escalón.** Adopta la forma

$$f(net) = \begin{cases} +1 & \text{si } net > 0 \\ -1 & \text{si } net < 0 \end{cases} \tag{2.6}$$

  y proporciona una salida bivaluada.

- **Función sigmoidea.** Es habitual en muchos modelos neuronales y provoca una transformación no lineal de su argumento. Una de las funciones más utilizadas es la función logística definida por:

$$f(net) = \frac{1}{1 + exp(-a \cdot net + b)}\,, a > 0\,, \tag{2.7}$$

  donde a es el parámetro de inclinación que ajusta la pendiente de la función y b el sesgo o bias.

- **Función Gaussiana.** su uso es común cuando se requiere de una transformación no lineal del espacio de entrada a otro con mayor dimensionalidad. Se puede expresar como:

$$f(net) = c \cdot exp\left(\frac{-net^2}{2\sigma^2}\right), \tag{2.8}$$

donde c es una constante y σ^2 es la varianza o amplitud de la función.

2.4 Red neuronal artificial

Una Red Neuronal Artificial (RNA) es un modelo simplificado de las redes neuronales biológicas. Tratan de extraer las excelentes capacidades del cerebro para resolver ciertos problemas complejos, como visión, reconocimientos de patrones o control moto-sensorial. Una red neuronal artificial es un procesador paralelo distribuido y masivamente interconectado que almacena conocimiento experimental [Haykin 1999].

Es el resultado de investigaciones académicas que involucran la utilización de fórmulas matemáticas para modelar operaciones del sistema nervioso. Las técnicas resultantes están empezando a tener éxito en una variedad de aplicaciones en los negocios cotidianos. Las RNA representan un provechoso acercamiento para usar las computadoras en los lugares de trabajo, usualmente son usadas para aprender patrones y relaciones de datos. Los datos pueden ser el resultado del esfuerzo de una investigación de mercado, el resultado de un proceso de producción dando variación a las condiciones de operación, o las decisiones de un prestamista dado un conjunto de aplicaciones de préstamo, entre otras aplicaciones.

Para establecer una similitud directa entre la actividad sináptica y la analogía con las redes neuronales artificiales se puede considerar que: las señales que llegan a la sinapsis son las entradas a la neurona; estas son ponderadas (atenuadas o simplificadas) a través de un parámetro, denominado peso asociado a la sinapsis correspondiente. Estas señales de entrada pueden excitar a la neurona (sinapsis con peso positivo) o inhibirla (peso negativo). El efecto es la suma de las entradas ponderadas. Si la suma es igual o mayor que el umbral[6] de la neurona, entonces la neurona se activa (da salida). Esta es una situación de todo o nada; cada neurona se activa o no. La facilidad de transmisión de señales se altera mediante la actividad del sistema nervioso. Esta habilidad de ajustar señales es un mecanismo de aprendizaje [Hilera 2000].

2.4.1 Características de una RNA

Las RNA se han convertido en una buena opción para multiples tareas del reconocimiento de patrones y otras areas de la Inteligencia Artificial, por que estas ofrecen numerosas ventajas sobre de los sistemas convencionales, algunas de las importantes son:

[6] Valor mínimo de una magnitud a partir del cual se produce un potencia de acción

- **No linealidad.** El procesador neuronal es básicamente no lineal y, por consecuencia, la red neuronal también.
- **Transformación entrada-salida.** El proceso de aprendizaje consiste básicamente en presentar a la red un ejemplo y modificar sus pesos sináptico de acuerdo con su respuesta. Aprende, por lo tanto, de una transformación entrada-salida.
- **Adaptabilidad.** La red tiene la capacidad de adaptar sus parámetros, aún en tiempo real.
- **Tolerancia a fallas.** Debido a la interconexión masiva, la falla de un procesador no altera seriamente la operación.
- **Uniformidad en el análisis y diseño.** Esto permite garantizar características precisas.

Por otra lado, una red neuronal matemáticamente puede verse como un grafo dirigido y ponderado [Hilera 2000] donde cada uno de los nodos son neuronas artificiales y los arcos o líneas que unen los nodos son denominadas conexiones sinápticas. Se dice que es dirigido porque los arcos son unidireccionales, sigue un único sentido desde una neurona presináptica a una neurona postsináptica. El grafo es ponderado, lo que significa que las conexiones tienen asociado un número real (peso) que indica la importancia de esa conexión con respecto a las otras, si dicho peso es positivo la conexión es excitadora, mientras que si es negativo la conexión es inhibidora. Lo usual es que las neuronas se agrupen en capas y cada capa recibe un nombre según la función que desempeñe, las más comunes son:

- **Capa de entrada:** las neuronas de esta capa reciben los datos para procesarlos.
- **Capas ocultas:** estas capas introducen grados de libertad adicionales a la RNA y en estas se realiza gran parte del procesamiento. El número de capas depende de la red que estemos utilizando.
- **Capas de salida:** proporciona la respuesta de la red neuronal.

Existen dos fases en toda aplicación de las RNA: la fase de aprendizaje o entrenamiento y la fase de prueba y aplicación [Hilera 2000].

- **Fase de Aprendizaje:** Una de las características de las RNA es su capacidad de aprender. Aprenden por la actualización o cambio de los pesos sinápticos que caracterizan a las conexiones en cada iteración del modelo neuronal y los pesos óptimos se obtienen optimizando (ya sea minimizando o maximizando) alguna función definida. Se pueden utilizar varios tipos de aprendizaje, los más comunes son [Barandela 2001]:

- **Aprendizaje supervisado:** Se proporciona a la RNA una serie de ejemplos consistentes en un conjunto de patrones de entrada, junto con la salida que debería dar la red. El proceso de entrenamiento consiste en el ajuste de los pesos para que la salida de la red sea lo más parecida posible a la salida deseada. Es por ello que en cada iteración se use alguna función que determina el error o el grado de acierto que está cometiendo la red.
- **Aprendizaje no supervisado o autoorganizado:** Se presenta a la red una serie de ejemplos pero no se presenta la respuesta deseada. Lo que hace la RNA, es reconocer regularidades en el conjunto de entradas, es decir, estimar una función densidad de probabilidad p(x) que describe la distribución de patrones x en un espacio de entrada Rn.
- **Aprendizaje hibrido:** Es una mezcla de los anteriores. Unas capas de la red tienen un aprendizaje supervisado y otras capas de la red tienen un aprendizaje de tipo no supervisado.

- **Fase de Prueba:** Lo parámetros de diseño de la RNA se han obtenido a partir de una muestra de aprendizaje formada de patrones que se denominan patrones de entrenamiento. Una vez calculados los pesos de la red, los valores de las neuronas o nodos de la última capa, se comparan con la salida deseada para determinar la validez del diseño. Una vez que el diseño ha sido validado correctamente y arrojado valores próximos a las salidas deseadas se procede a determinar la generalización de la red suministrándole datos o información de diferentes problemas.

2.4.2 Algunas aplicaciones de las RNA

Las aplicaciones prácticas de esta rama de la investigación científica y tecnológica son múltiples y no parecen tener fin. En el terreno de la medicina han apoyado la obtención de diagnósticos con mayor precisión y menor riesgo para los pacientes, además de permitir el análisis de las imágenes de diversos estudios clínicos, y facilitar la elaboración de medicamentos [Papik 1998]. En aplicaciones financieras se examinan diariamente los movimientos del mercado financiero y se realiza la predicción de los índices bursátiles con lo cual se cubre la función de asesoría. También hay programas que apoyan los procesos de transacciones bancarias mediante la detección de fraudes [Chan 1999] no evidentes para otros sistemas de control. En el tema de la edición de sonido un programa basado en este principio permite discriminar voces o sonidos ambientales sin detrimento de las demás señales de la grabación.

Las aplicaciones de las RNA son diversas y en diferentes campos de aplicación. A continuación se muestran algunas aplicaciones clasificadas de acuerdo a los siguientes tópicos [Kemley 1992]:

- **Visión.** Se engloban todas las aplicaciones de reconocimiento de caracteres. En [DARPA-USA 1988] se habla de una aplicación de reconocimiento de caracteres

escritos a mano que puede reconocer 2,500 caracteres del lenguaje escrito *kanji* de Japón con una precisión del 92%. *Nestor*[7] creo una aplicación para verificar si una firma era valida o falsa con un 4% de falsas alarmas comparado con un 50% de falsas alarmas humanas. También se incluye el reconocimiento de Rostros [Midorikawa 1988] y categorización de imágenes (números, letras y símbolos electrónicos) [krishman 1988]. También en identificación de productos defectuosos en la línea de ensamblaje para la fabricación de automóviles [Murphey 2004].

- **Reconocimiento de voz.** La mayoría de este tipo de aplicaciones analizan la forma de onda del sonido para después procesarlo e interpretarlo. Por ejemplo, una máquina de escribir fonética, reconocimiento de un locutor, un reconocedor de palabras [Kemley 1992]. También se han creado aplicaciones para la detección de fraudes en llamadas telefónicas [Fawcett 1997].

- **Análisis de señales.** Se incluyen aplicaciones para el análisis, reconocimiento y clasificación de señales. Son aplicaciones para analizar datos extraídos de sensores para la evaluación de amenazas, seguimiento de blancos (objetivos) mediante imágenes en 2-D del objeto [DARPA-USA 1988], reconocimiento de objetivos por medio señales de radar (*Nestor systems*) y reconocimiento del sonido de llamada de una Orca [Kemley 1992], entre otras.

- **Robótica.** Principalmente se compone de aplicaciones para la construcción de vehículos autónomos y control de trayectorias en manipuladores. Un ejemplo de vehículo autónomo, fue el robot creado en los laboratorios de Fujitsu que puede interpretar los datos del sensor y tomar decisiones simples y apropiadas. Otro ejemplo, fue el sistema creado por la Universidad de Boston llamando *Visual Tracking and Navigation* basado en dos cámaras para detectar y rastreas objetos en movimiento. Para robot manipuladores se han aplicado las redes neuronales para la corrección de posición.

- **Sistemas expertos neuronales.** Dentro de este tópico se encuentran aplicaciones para servicios médicos como el sistema de [Saito 1988] para el diagnostico de 23 diferentes enfermedades con un 67% de precisión y la detección de tumores mediante imágenes en infra-rojo. También se encuentran aplicaciones financieras, por ejemplo para el análisis de mercado[8], valoración del riesgo de los créditos, etc.

- **Computación.** Aquí se incluyen memorias asociativas, recuperación de textos, ordenamiento, interfaces inteligentes, enrutamiento de red y soportes para paginación [Kemley 1992].

- **Planeación y control de procesos.** Se compone de aplicaciones para optimizar

[7] Nestor, Inc., Richmond Square, Providence, RI 02690
[8] Hecht-Nielsen Neurocomputers, 5501 Oberlin Drive, San Diego, CA 92121

un proceso o una secuencia de procesos. Por ejemplo, sistemas de inspección automática o planificadores de tareas [DARPA-USA 1988].

2.4.3 Proceso de aprendizaje

El proceso de aprendizaje o entrenamiento de la RNA consiste en modificar los pesos de la conexión sistemáticamente para codificar las relaciones de entrada-salida [Freeman 1991]. En otras palabras, es el mecanismo por el cual los parámetros libres de la RNA son adaptados a través de un procedimiento de estimulación del ambiente en el cual esta contenida.

El tipo de aprendizaje esta determinado por la manera en la cual el cambio de los parámetros es realizado [Russell 1996]. Este proceso implica la siguiente secuencia de eventos:

- La red es estimulada por su entorno.
- La RNA sufre cambios en sus parámetros como resultado de esta estimulación.
- La RNA responde al entorno de manera diferente por los cambios ocurridos en su estructura interna.

Lo mas común es usar la arquitectura y el tipo de aprendizaje como criterios de clasificación de las RNA. Es por ello que un tipo de clasificación que se realiza sobre las redes neuronales obedece al tipo de aprendizaje utilizado por dichas redes. Así se pueden distinguir tres tipos de aprendizaje: el supervisado, no supervisado e híbrido (explicados en la sección 2.4.1).

Al conjunto prescrito de reglas definidas para solucionar el problema de adquisición de conocimiento se le llama algoritmo de aprendizaje. Existen varios algoritmos y cada uno obedece a una regla de aprendizaje especifica. Básicamente existen 4 reglas [Jain 1996]:

- Corrección de error. En el paradigma de aprendizaje supervisado se establece un valor deseado (d) como salida de la RNA para cada entrada (x). Durante el proceso de aprendizaje la salida (y) generada por la red suele no ser igual a la salida deseada. El principio básico de la corrección del error es usar el error generado por la red ($d-y$) para modificar los pesos de las conexiones gradualmente y así reducir este error.
- La regla de Boltzmann. Es una regla de aprendizaje estocástica obtenida a partir de principios de la teoría de la información y de la termodinámica. El objetivo de esta regla es ajustar los pesos de las conexiones de tal forma que el estado de las unidades visibles satisfagan una distribución de probabilidad deseada. Puede ser vista como un caso especial de corrección del error, en el cuál el error no es medido directamente como la diferencia entre la salida deseada y la actual, sino que corresponde a la correlación de las salidas de las neuronas.

- Regla Hebbiana. Es un mecanismo dependiente del tiempo y del ámbito local. Incrementa la eficiencia de una sinapsis en función de la correlación entre las actividades pre y pos-sinápticas. Si dos neuronas están simultanea y repetidamente activas, las conexiones entre ellas deben ser fortalecidas y en caso contrario deben ser debilitadas.

- Regla de aprendizaje competitivo. Suele decirse que las neuronas compiten unas con otras con el fin de llevar a cabo una tarea. Con este tipo de aprendizaje se pretende que cuando se presenta a la red cierta información de entrada, sólo una de las neuronas de salida de la red o un cierto grupo de neuronas sean activadas (hasta alcanzar su valor de respuesta máximo). Por tanto, las neuronas compiten para activarse quedando finalmente una o un grupo como neuronas vencedoras, y el resto quedan anuladas siendo forzadas a que sus valores de respuesta sean mínimos.

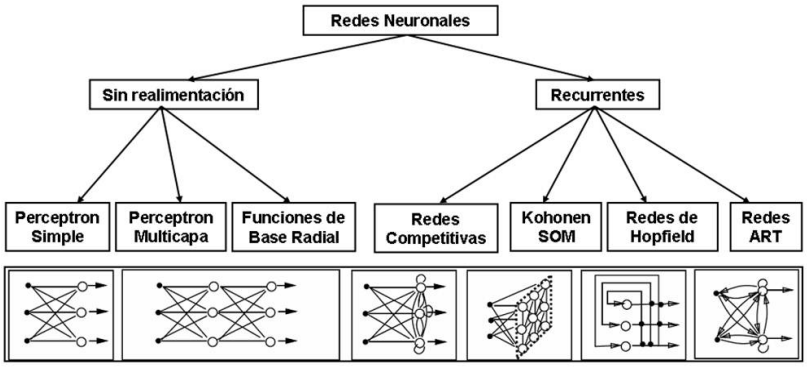


Figura 2.3: Esquema de las principales arquitecturas de RNA.

El otro criterio de organización de la RNA es mediante la arquitectura que disponen (ver Fig. 2.3). Se pueden tener dos posibilidades distintas: a) si la arquitectura de la red presenta ciclos la red se llama unidireccional (feedforward), y b) si se puede trazar un camino de una neurona a sí misma entonces este tipo de redes se denominan recurrentes o retroalimentadas [Russell 1996].

2.4.4 Perceptrón Multicapa

En 1969 Minsky y Papert publicaron su libro Perceptrons: An introductions to computacional geometry [Minsky 1969], el cual causó una des-aceleramiento en el desarrollo de las redes neuronales artificiales. En él, se presentaba un análisis detallado del Perceptrón, en términos de sus capacidades y limitaciones, en especial en cuanto a las restricciones que existen para las redes tipo Perceptrón; la mayor desventaja de este

tipo de redes es su incapacidad para resolver problemas de clasificación que no sean linealmente separables [Haykin 1999].

El Perceptrón Multicapa (MLP, multilayer perceptron) (ver Fig. 2.4), inicialmente desarrollado por P. Werbos (1974), permite resolver el problema de la no linealidad. Estas han sido aplicadas satisfactoriamente para resolver tareas de clasificación, predicción, aproximación de funciones y control

El MLP es una estructura interconexionista compuesta por una o más capas ocultas entre los nodos de entrada y los de salida [Lippmann 1988]. Estas capas están integradas por nodos que pueden o no estar conectados directamente a los de entrada y a los de salida. Se caracteriza fundamentalmente por que no existen conexiones entre neuronas de la misma capa ni conexiones hacia atrás. Cada capa alimenta a todas las neuronas de la capa siguiente.

En el MLP la cantidad de neuronas de entrada se corresponde con la dimensión del vector de entrada ($\mathbf{x}$) y el número de neuronas en la capa de salida con el total de clases en la ME. La cantidad de neuronas y capas ocultas no esta predefinida por lo que ha sido objeto de estudio durante muchos años [Haykin 1999].

El MLP tiene tres características distintivas:

1. El modelo de cada neurona en la red incluye una función de activación no lineal. Lo importante aquí es que la no linealidad es suave (en cualquier punto existen todas sus derivadas).

2. La red contiene una o más capas ocultas que no son parte de las entradas o las salidas de la red. Estas neuronas ocultas permiten que la red aprenda tareas complejas por la extracción progresiva de las características principales de los patrones de entrada.

3. La red presenta altos grados de conectividad, determinados por las sinapsis de la propia red.

La combinación de estas características junto con la habilidad de aprender de la experiencia a través del entrenamiento del MLP, dan como resultado un gran potencial de computación.

La Fig. 2.4 muestra un MLP con tres capas: entrada, oculta y salida. Observe, que cada neurona proporciona una salida como resultado de la combinación de todas las señales que recibe y su procesamiento por medio de la función de activación $f(\cdot)$.

Un MLP es considerado un prototipo de entrada-salida dado que a cada entrada $\mathbf{x}_n = [x_1, x_2, ..., x_i]$ se produce una salida $\mathbf{z}_n$ donde

$$\mathbf{z}_n = f(\mathbf{y}_n, \mathbf{U}), \quad \text{para} \quad \mathbf{y}_n = f(\mathbf{x}_n, \mathbf{W}), \tag{2.9}$$

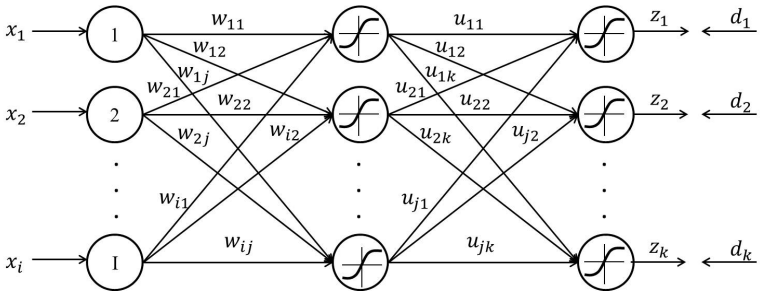


Figura 2.4: MLP de tres capas con I nodos en la capa de entrada, J neuronas ocultas y K nodos de salida. z es la salida real y d la esperada para la entrada x.

$$\mathbf{W} = \begin{pmatrix} w_{11} & w_{12} & \cdots & w_{1J} \\ w_{21} & w_{22} & \cdots & w_{2J} \\ \vdots & \vdots & \vdots & \vdots \\ w_{I1} & w_{I2} & \cdots & w_{IJ} \end{pmatrix}, \ \mathbf{U} = \begin{pmatrix} u_{11} & u_{12} & \cdots & u_{1K} \\ u_{21} & u_{22} & \cdots & u_{2K} \\ \vdots & \vdots & \vdots & \vdots \\ u_{J1} & u_{J2} & \cdots & u_{JK} \end{pmatrix},$$

y la función de activación $f(\cdot)$ es definida como

$$f(net) = 1/(1 + exp(-a \cdot net + c)), \tag{2.10}$$

o bien

$$f(net) = a \cdot tanh(b \cdot net + c), \tag{2.11}$$

donde a, b y c son constantes, y net[9] es el sumador o integrador que combina linealmente las señales de entrada de la neurona con sus respectivos pesos. Ambas funciones (Ec. 2.10, Ec. 2.11) cumplen con dos condiciones fundamentales:

- No linealidad: necesaria para superar los inconvenientes de la separabilidad lineal.
- Diferenciabilidad: imprescindible para el uso de algoritmos de entrenamiento o aprendizaje robustos.

El MLP supera muchas de las limitaciones del perceptrón de Rosenblatt pero no fue utilizada en el pasado por falta de un algoritmo de entrenamiento efectivo [Lippmann 1988]. Más adelante con la popularización del algoritmo back-propagation [Rumelhart 1986] el MLP se convirtió en una atractiva alternativa para la resolución problemas complejos.

[9]En el caso de la Fig. 2.4, la variable net puede asumir dos valores dependiendo de la capa en que se encuentre (en la oculta $net_j = \sum_{i=1}^{I} w_{ij} x_i$, o en la salida $net_k = \sum_{j=1}^{J} u_{jk} y_j$).

2.4.4.1 Arquitectura del MLP

En la estructura del MLP es fundamental la elección del número de capas ocultas y la cantidad de nodos en cada una de ellas. En [Lippmann 1988] se indica que los MLP de una capa oculta son capaces de producir regiones de decisión linealmente separables, las de dos capas pueden generar zonas convexas cerradas o abiertas, y finalmente las de tres capas poseen la capacidad de dividir el espacio de observación en áreas de diversas formas. En [Cybenko 1989] se estudia la capacidad de representación del MLP en relación al número de capas ocultas. Ahora se sabe que un MLP con una capa oculta es capaz de encontrar cualquier relación que pueda ser aproximada por una función continua, y que con dos capas ocultas cualquier relación que implique funciones discontinuas [Russell 1996]. No obstante, se ha demostrado que para la mayoría de los problemas bastará con una capa oculta [Funahashi 1989].

Otro problema relacionado a la topología del MLP es la determinación del número óptimo de nodos por capa oculta. A diferencia de la elección del número de capas ocultas, la identificación del número de neuronas ocultas necesarias para resolver un problema es aún más complicado. Al día de hoy no existe ninguna solución sistematizada que sea completamente aceptada para realizar esta labor. Sin embargo, algunos métodos se han desarrollado con este propósito. En la Tabla 2.2 se muestran algunas de estas estrategias.

La importancia de obtener el número idóneo de neuronas ocultas descansa en el hecho de que determinan la capacidad de representación de la red y la complejidad de la frontera de decisión [Duda 2001b].

En la actualidad, el proceso más común para la estimación del número de elementos en la capa oculta es realizado mediante prueba y error. En otras palabras, el investigador usando su experiencia fija el número de nodos para cada capa oculta.

2.4.4.2 Algoritmo Back-propagation

El proceso de aprendizaje o entrenamiento del MLP consiste en la estimación de sus parámetros libres (pesos de la red). El algoritmo back-propagation descrito en primer lugar por Werbos [Werbos 1974], posteriormente por Parker [Parker 1985] y finalmente por Rumelhart [Rumelhart 1986], es el método de aprendizaje más ampliamente utilizado en el MLP. Esta basado en una técnica de descenso por gradiente que utiliza la minimización del error cuadrático medio (Mean Square Error, MSE) mediante un proceso iterativo. El MSE es definido como:

$$E(V) = \frac{1}{2N} \sum_{n=1}^{N} \sum_{k=1}^{K} (d_k^n - z_k^n)^2 \,, \tag{2.12}$$

donde $\mathbf{d}^n$ es la salida esperada de la red, $V = \{\mathbf{W}, \mathbf{U}\}$ los parámetros libres de la red, y $\mathbf{z}^n(\cdot)$ la salida real. N es el total de muestras de entrenamiento y K el número de clases.

Tabla 2.2: Número óptimo (N_J) de neuronas ocultas sugerido en algunos trabajos. N_I representa la cantidad de neuronas en la capa de entrada, N_w identifica el número total de conexiones en la red, N es la cantidad de muestras de entrenamiento y N_K las neuronas correspondientes a la capa de salida.

Autor(es)	Estrategia
Hecht-Nielsen [Hecht-Nielsen 1990]	$N_J \leq N_I + 1$
Jadid y Fairbairn [Jadid 1996]	$N_J = \frac{N}{R+N_I+N_K}$ donde $R = -5$
Lachtermacher y Fuller [Lachtermacher 1995]	$\frac{0.11N}{N_I+1} \leq N_J \leq \frac{0.3N}{N_I+1}$
Masters [Masters 1993]	$N_J \approx (N_I \cdot N_K)^{1/2}$
Pao [Pao 1989]	$N_J = 2N_I$; o $N_J = N_I + 1$;
Duda et al. [Duda 2001b]	$N_{wk} \approx \frac{N}{10}$ N_{wk} conexiones por unidad oculta
Upadhaya and Eryureka [Upadhyaya 1992]	$N_w = Nlog_2(N)$; N_w esta relacionado a N_J

El método de descenso por gradiente iterativo puede ser formulado como sigue

$$V^{t+1} = V^t + \eta_i \nabla E(V^t)\,, \tag{2.13}$$

donde t es la t-*ésima* iteración y η_i la *razón de aprendizaje* o *learning rate*[10] ($0 < \eta_i \leq 1$). Al pasar de la iteración t a la $t+1$ el algoritmo aplica la corrección

$$\nabla V = V^{t+1} - V^t = \eta_i \nabla E(V^t)\,, \tag{2.14}$$

en la dirección opuesta al vector gradiente $\nabla E(V^t)$.

En términos generales, el algoritmo back-propagation para un MLP de tres capas se puede resumir como sigue:

1. Inicializar aleatoriamente con valores pequeños los pesos de la red. Generalmente con valores entre -0.5 y 0.5.

2. Aleatoriamente elegir una muestra de entrada $\mathbf{x}^{(n)}$.

[10]Ésta tiene una enorme influencia en la convergencia del método [Anand 1993, Looney 1997].

3. Propagar la señal hacia adelante a través de la red.

$$z_k^{(n)} = g(\sum_{j=1}^{J} u_{jk} h(\sum_{i=1}^{I} w_{ij} x_i^{(n)}))\,, \tag{2.15}$$

donde $z_k^{(n)}$ es la salida de la red para la entrada $x_i^{(n)}$; $g(\cdot)$ y $h(\cdot)$ representan la función de activación (Ec. 2.10).

4. Calcular δ_k^L para la capa de salida.

$$\delta_k^L = [z_k^{(n)}(1 - z_k^{(n)})](d_k^{(n)} - z_k^{(n)})\,, \tag{2.16}$$

donde L es el número de capas ocultas mas 1 (en este caso $L = 2$).

5. Calcular las deltas (δ) para las capas previas por propagación del error hacia atrás.

$$\delta_j^l = [y_j^n(1 - y_j^{(n)})] \sum_{k=1}^{K} u_{jk}^{(t)} \delta_k^L, \tag{2.17}$$

para $l = (L-1), ..., 1$, donde $t = (1, ..., T)$ es el número de iteración o repetición del algoritmo.

6. Actualizar los pesos usando

$$u_{jk}^{(t+1)} = u_{jk}^{(t)} + (\eta_i \delta_k^L y_j^{(n)})^{(t)}, \tag{2.18}$$

para la capa de salida y

$$w_{ij}^{(t+1)} = w_{ij}^{(t)} + (\eta_i \delta_j^l x_i^{(n)})^{(t)}, \tag{2.19}$$

para la capa oculta.

7. Regresar al paso 2 y repetir para la siguiente muestra hasta alcanzar el mínimo de error (fijado a priori) o hasta alcanzar el número máximo de iteraciones.

2.4.4.3 Razón de aprendizaje y Momento

En el algoritmo back-propagation la elección apropiada de la razón de aprendizaje (η) es un factor crítico en el desempeño de la red [Looney 1997], porque determina la magnitud de las actualizaciones en los pesos. Si η es demasiado pequeña la velocidad de la convergencia es excesivamente lenta y la probabilidad de quedar atrapado en un mínimo local se eleva, mientras que si η es demasiado grande conduce a inestabilidad (oscilaciones) dentro de la función de error con el peligro de pasar por encima del mínimo global [Haykin 1999].

Para ayudar a disminuir las oscilaciones en la función de error de una iteración a otra e incrementar la velocidad de convergencia, en [Rumelhart 1986] se propone la técnica llamada *momento* (μ). Cuando se calcula el valor del cambio de peso $\nabla \mathbf{v}$ se añade una fracción del cambio anterior. Este término adicional tiende a mantener los cambios de peso en la misma dirección: de aquí el término *momento* [Freeman 1991]. La adición del término momento μ da como resultado

$$\mathbf{v}^{(t+1)} = \mathbf{v}^{(t)} + \eta \nabla \mathbf{v}^{(t)} + \mu \nabla \mathbf{v}^{(t-1)}, \quad 0 < \mu < 1, \tag{2.20}$$

donde t es la t-ésima iteración y $\mathbf{v}$ el vector de pesos. Los valores más comúnmente usados para el momento son $\mu \approx 0.9$ [Duda 2001b].

2.4.5 Redes de Función de Base Radial

Las Redes Neuronales de Funciones de Base Radial (RBF, *radial basis function*) son un poderoso tipo de redes de propagación hacia adelante [Looney 1997]. Estas fueron introducidas a finales de los años 80 como solución de problemas con interpolación de funciones multivaradas [Powell 1987].

La principal diferencia entre las redes MLP y las RBF, está en la función de activación de los nodos ocultos [Uykan 2000]. Esta última, se caracteriza por tener sólo tres capas que realizan actividades diferentes. La capa de entrada relaciona a la red con el entorno, la segunda capa es oculta y aplica una transformación no lineal al espacio de entrada, y la tercera capa es la de salida, es lineal y sólo da las respuestas de la red a las estimulaciones recibidas del entorno.

Una RBF está diseñada con neuronas en la capa oculta activadas mediante funciones radiales de carácter no lineal con sus centros gravitacionales propios y en la capa de salida mediante funciones lineales (ver Fig. 2.5).

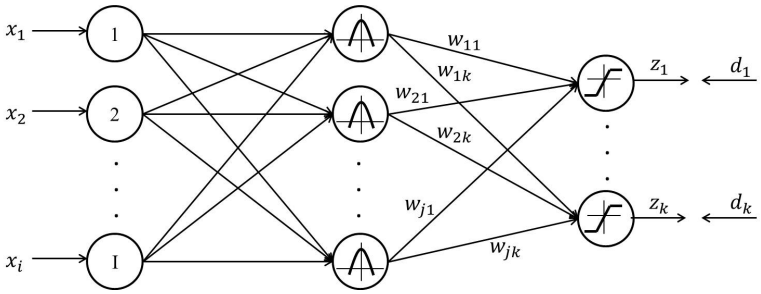


Figura 2.5: Arquitectura de RBF, con I nodos en la capa de entrada, J neuronas ocultas y K nodos de salida. z_k es la salida real y d_k la esperada para la entrada x_n, w_{jk} son los pesos de la red.

El modelo de red RBF estándar puede ser formulado como sigue

$$z_k(\mathbf{x}) = \sum_{j=1}^{J} w_{jk} h_j(\|\mathbf{x} - \mathbf{c}_j\|) + w_{0k}\,, \tag{2.21}$$

donde

$$h_j(\|\mathbf{x} - \mathbf{c}_j\|) = exp\left(-\frac{\|\mathbf{x} - \mathbf{c}_j\|^2}{2\sigma_j^2}\right)\,, \tag{2.22}$$

$\|\cdot\|$ representa la norma Euclidiana, h_j es la función activación en la capa oculta con centro en $\mathbf{c}_j$ y varianza σ_j^2. El vector de pesos (w_{jk}) establece la relación entre la capa oculta y la de salida, w_{0k} representa el valor asignado al sesgo o "bias".

2.4.6 Proceso de aprendizaje

El entrenamiento o aprendizaje de una red es el proceso por el cuál los parámetros libres de la red son adaptados a partir de las estimulaciones recibidas del ambiente, y de esta forma puedan desempeñar las tareas que se les asignen eficientemente [Jain 1996]. En el caso particular de las redes RBF, los parámetros libres que hay que adaptar al problema son: pesos $(\mathbf{w})$, centros $(\mathbf{c})$ y varianzas (σ^2) [Looney 1997].

Antes de iniciar el proceso de entrenamiento de la red se deben considerar dos aspectos básicos: a) seleccionar el modelo de RBF para la capa oculta y b) determinar el número necesario de neuronas ocultas para resolver el problema.

Algunos modelos de RBF se muestran a continuación:

1. Cuadrática: $h(x) = (x^2 + c^2)^{\frac{1}{2}}$ para algún valor $c > 0$ y $x \in \mathbb{R}$
2. Cuadrática Inversa: $h(x) = \frac{1}{(x^2 + c^2)^{\frac{1}{2}}}$ para algún valor $c > 0$ y $x \in \mathbb{R}$
3. Función Gaussiana: $h(x) = exp(-\frac{(x-c)^2}{2\sigma^2})$ para algún valor $c > 0$ y $x \in \mathbb{R}$

En la red RBF es común el uso de funciones basadas en una distribución de probabilidad normal o Gaussiana (Ec. 2.22) [Ghodsi 2003] y determinar empíricamente el número de neuronas de la capa oculta. No obstante, en los últimos años se ha incrementado el interés por automatizar este proceso [Schölkopf 1997, Xu 1998, Harpham 2004].

2.4.7 Aprendizaje Híbrido

Las capas de la red RBF realizan diferentes tareas. Por lo tanto, el entrenamiento de la red puede ser dividido en dos fases [Haykin 1999]. La primera fase (aprendizaje no supervisado) consiste en determinar el número y posiciones de los centros, así como la varianza de las RBF en la capa oculta para después calcular la distancia de los patrones a los centros calculados. La segunda fase (aprendizaje supervisado) corresponde a la estimación de los pesos de la red.

2.4.7.1 Aprendizaje no supervisado

Estudios teóricos y empíricos han mostrado que el rendimiento de las redes RBF dependen directamente de los valores usados por la capa oculta [Uykan 2000].

Una de las técnicas más sencillas para la selección de los centros consiste en elegirlos de forma aleatoria desde los datos de entrada. Este método no puede ser considerado óptimo [Lowe 1989] por que se asume que los datos de entrenamiento están dispuestos de acuerdo a la distribución real del problema.

Otro enfoque es el uso de estrategias de *clustering* para la ubicación de los centros de la red [Schwenker 2001]. Por ejemplo, el algoritmo k-means[11] [Duda 2001b], el LVQ (learning vector quantization) [Gray 1984], o los mapas auto-organizados de Kohonen (Selft Organization Maps, SOM) [Kohonen 1990].

Para fijar el valor de la desviación estándar de las RBF se suele utilizar algún valor heurístico. En [Haykin 1999] se sugiere el uso de un valor proporcional a la distancia máxima entre los centros como se expresa a continuación:

$$\sigma = \frac{d_{max}}{\sqrt{2J}}\,, \tag{2.23}$$

smo Sequential Minimal Optimization donde d_{max} es la distancia máxima entre los centros elegidos y J el número de centros. Esta fórmula asegura que la forma de las funciones no sea demasiado suave ni excesivamente pronunciada, condiciones que deben ser evitadas. No obstante, al establecer el mismo valor de σ para todas las RBF se asume que los datos están distribuidos de manera uniforme, situación que pocas veces ocurre en la práctica [Benoudjit 2003].

En [Saha 1989] se sugiere una σ distinta para cada RBF y se propone el uso de la estrategia heurística *nearest neighbour* (Ec. 2.24). El valor de σ_j se obtiene de multiplicar la distancia entre el centro c_j y su vecino más próximo c_p por un valor constante de solapamiento (r).

$$\sigma_j = r \cdot min(\|c_j - c_p\|)\,. \tag{2.24}$$

smo Sequential Minimal Optimization Se han propuesto otras alternativas heurísticas para la ubicación de σ, por ejemplo, el *p-nearest neighbour* [Moody 1989]. En este procedimiento σ es establecida como el promedio de las distancias del centro c_j y sus p vecinos más próximos (Ec. 2.25).

$$\sigma_j = \frac{1}{p}\left(\sum_{i=1}^{p} \|c_j - c_i\|^2\right)^{\frac{1}{2}}\,. \tag{2.25}$$

[11] Es el algoritmo más frecuentemente empleado con este propósito.

2.4.7.2 Aprendizaje Supervisado

Al obtenerse los parámetros libres de la capa oculta ($\mathbf{c}$ y σ) solo resta estimar los valores de los pesos ($\mathbf{W}$) de la red. Esta tarea se puede realizar a través de la aplicación de algún método de optimización lineal [Jain 1996].

Suponiendo que $f(\mathbf{x}_n) = d_n$ es la salida deseada para la entrada $\mathbf{x}_n$ $(n = 1, ..., N)$. La Ec. 2.21 se puede generalizar mediante notación matricial de la siguiente manera:

$$\mathbf{HW} = \mathbf{D} \tag{2.26}$$

en otras palabras,

$$\begin{pmatrix} h(\cdot)_{11} & h(\cdot)_{12} & \cdots & h(\cdot)_{1J} \\ h(\cdot)_{21} & h(\cdot)_{22} & \cdots & h(\cdot)_{2J} \\ \vdots & \vdots & \vdots & \vdots \\ h(\cdot)_{N1} & h(\cdot)_{N2} & \cdots & h(\cdot)_{NJ} \end{pmatrix} \begin{pmatrix} w_{11} & \cdots & w_{1K} \\ w_{21} & \cdots & w_{2K} \\ \vdots & \vdots & \vdots \\ w_{J1} & \cdots & w_{JK} \end{pmatrix} = \begin{pmatrix} d_{11} & \cdots & d_{1K} \\ d_{21} & \cdots & d_{2K} \\ \vdots & \vdots & \vdots \\ d_{N1} & \cdots & d_{NK} \end{pmatrix},$$

donde $\mathbf{H} = (\mathbf{h_n}(\mathbf{x_n}))$, J el número de centros y K el número de clases en la ME o unidades en la capa de salida (ver Fig. 2.5). smo Sequential Minimal Optimization A partir de la Ec. 2.26 se puede calcular $\mathbf{W}$ como sigue

$$\mathbf{W} = \mathbf{H}^{+}\mathbf{D} = (\mathbf{H^T H})^{-1}\mathbf{H^T D}\,, \tag{2.27}$$

donde $\mathbf{H}^{+}$ es la matriz *pseudo-inversa* de $\mathbf{H}$. En [Golub 1996] se estudian algunos algoritmos eficientes para el calculo de la matriz *pseudo-inversa*.

Las redes RBF al igual que el MLP pueden ser entrenadas por métodos similares de descenso por gradiente [Ding 2004]. Así, los parámetros $U = \{w, c, \sigma\}$ de la red RBF son obtenidos simultáneamente. A continuación, se presenta una breve descripción del algoritmo back-propagation en el contexto de las redes RBF.smo Sequential Minimal Optimization

Considérese el error cometido por la RNA (Ec. 2.12), donde la salida real es:

$$z_k(\mathbf{x}_n) = \sum_{j=1}^{J} w_{jk} h_j(\|\mathbf{x} - \mathbf{c}_j\|) + w_{0k}\,. \tag{2.28}$$

smo Sequential Minimal Optimization Por lo tanto, nos interesa encontrar una solución que minimice el MSE (Ec. 2.12). Esta situación se puede formular como un problema de optimización sin restricciones. La condición necesaria para el valor óptimo es que $\nabla E(U^*) = 0$ y que $E(U^*) \leq E(U)$. smo Sequential Minimal Optimization Una de las estrategias más populares para la minimización del MSE es el método de descenso por gradiente descrito en la sección 2.4.4.2. La clave en este método es encontrar los ∇ apropiados para actualizar en cada iteración los parámetros $U = \{w, c, \sigma\}$.

Al descomponer la Ec. 2.12 se tiene:

$$E(U) = E^{(1)} + E^{(2)} + \cdots + E^{(N)}\,, \tag{2.29}$$

donde el error parcial es

$$E^{(n)} = \sum_{k=1}^{K} \frac{1}{2}(d_k - z_k)^2 \, , smoSequentialMinimalOptimization \tag{2.30}$$

y así el problema es simplificado, y a partir de la Ec. 2.30 se pueden obtener las siguientes reglas de actualización para la red RBF.

$$\nabla w_{jk} = -\frac{\partial E}{\partial w_{jk}} = -\sum_{n=1}^{N} h_j(\|\mathbf{x}^n - \mathbf{c}_j\|)(d_k^n - f_k^n) \, , \tag{2.31}$$

$$\nabla c_{ji} = -\frac{\partial E}{\partial c_{ji}} = -\frac{1}{\sigma_j^2} \sum_{n=1}^{N} \left[\sum_{k=1}^{K} (d_k^n - f_k^n) w_{jk} \right] h_j(\|\mathbf{x}^n - \mathbf{c}_j\|)(x^n - c_{ji}) \, , \tag{2.32}$$

$$\nabla \sigma_j = -\frac{\partial E}{\partial \sigma_j} = -\frac{1}{\sigma_j^3} \sum_{n=1}^{N} \sum_{k=1}^{K} (d_k^n - f_k^n) w_{jk} h_j(\|\mathbf{x}^n - \mathbf{c}_j\|) \|\mathbf{x}^n - \mathbf{c}_j\|^2 \, . \tag{2.33}$$

Los parámetros libres (U) de la red también pueden ser obtenidos por algún otro método de optimización no lineal, como por ejemplo el método de *Quasi-Newton* [Lowe 1989] o el de gradiente conjugado [Wettschereck 1992].

2.4.8 Máquinas de vectores soporte

Las máquinas de vectores soporte (Support Vector Machine, SVM), son mecanismos de aprendizaje supervisado que se han popularizados en los últimos años en tareas de clasificación y regresión. Sus orígenes están la teoría de del aprendizaje estadístico desarrollada por Vladimir Vaqnik y sus colegas de los laboratorios Bell AT&T en 1995 [Shawe-Taylor 2000].

Las SVM están basadas en el principio de Minimización de Riesgo Estructural de una forma especial y como concecuencia proporcionan una buena capacidad de generalización independientemente de la distribución de los datos [Marques de Sa 2001]. Este principio consiste básicamente en realizar un mapeo no lineal del espacio de entrada a un espacio de características de mayor dimensión.

- En tareas de clasificación lo que se busca es construir un hiperplano que separe de manera óptima el espacio de características, es decir, un hiperplano que maximice la distancia entre éste y las clases contenidas en el espacio de características.

- En tareas de regresión su función es desarrollar una regresión lineal en el espacio de características.

Este trabajo se centra en el estudio de las SVM como mecanismos de clasificación y no se profundiza en los conceptos de regresión relacionados a las SVM.

Las SVM originalmente fueron desarrolladas para trabajar con problemas de clasificación de dos clases, donde lo que se busca es encontrar un hiperplano óptimo que separe con un margen de separación máximo las clases representadas en el espacio de características. Al día de hoy se ha extendido su uso a problemas de múltiples clases.

Una característica fundamental de las SVM es que su solución esta basada únicamente en un subconjunto de los datos de entrenamiento que sirven para establecer el margen de separación entre clases. A estos datos se les conoce como **vectores soporte** (ver Fig. 2.6).

En esta sección se describen brevemente los principio básicos del funcionamiento de las SVM. Primeramente se muestra su modo de operación cuando los datos de entrada son linealmente separables y posteriormente se describen las modificaciones necesarias para que puedan operar en problemas no linealmente separables.

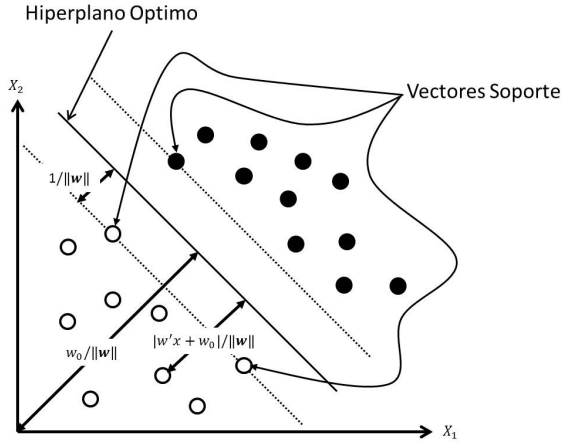


Figura 2.6: Representación de un hiperplano óptimo de separación para una máquina de vectores soporte.

2.4.8.1 Problemas linealmente separables

Dado un conjunto de patrones linealmente separables $(\mathbf{X}_i, y_i)_{1 \leq i \leq N}$, donde $\mathbf{X}_i \in R^d$, $y_i \in \{-1, 1\}$ es la etiqueta de clase a la que pertenece $\mathbf{x}_i$. La forma general de la función de clasificación lineal es dada por:

$$g(\mathbf{x}) = \mathbf{w} \cdot \mathbf{x} + b, \tag{2.34}$$

la cual corresponde al hiperplano de separación $\mathbf{w} \cdot \mathbf{x} + b = 0$. Si se normaliza $g(\mathbf{x})$ para satisfacer $\|g(\mathbf{x})\| \geq 1$ para todos los valores de $\mathbf{x}_i$, entonces la distancia desde los patrones al hiperplano es $1/\|\mathbf{w}\|$.

El *hiperplano de separación óptimo* (HSO) es aquel donde la distancia de los puntos cercanos (patrones) a él es máxima, y también la distancia de los puntos cercanos al HSO es aproximadamente $1/\|\mathbf{w}\|$ (véase la Fig. 2.6). Para encontrar el HSO se requiere minimizar $\|\mathbf{w}\|$, así que la función objetivo se puede plantear como:

$$\min\phi(\mathbf{w}) = \frac{1}{2}\|\mathbf{w}\|^2 = (\|\mathbf{w}\| \cdot \|\mathbf{w}\|) \tag{2.35}$$

sujeto a:

$$y_i(\mathbf{w} \cdot \mathbf{x} + b) \geq 1, \quad i = 1, ..., N. \tag{2.36}$$

Si se asocian N no negativos multiplicadores de Lagrange $(\alpha, ..., \alpha_N)$ con la restricción 2.36, se puede construir un HSO, siempre y cuando se resuelva el problema de programación cuadrático planteado.

La solución de $\mathbf{w}$ tiene una expansión $\mathbf{w} = \sum \alpha_i y_i \mathbf{x}$ en términos de un subconjunto de patrones de entrenamiento llamados *vectores soporte*, los cuales se encuentran en el margen del HSO. La función de clasificación para este problema puede ser escrita como:

$$f(\mathbf{x}) = sing(\sum_{i=1}^{N} \alpha_i y_i \mathbf{x}_i \cdot \mathbf{x} + b). \tag{2.37}$$

2.4.8.2 Problemas no linealmente separables

Cuando los datos o patrones no son linealmente separables, se introducen variables de holgura y un factor de penalización de tal manera que la función objetivo puede ser modificada como:

$$\phi(\mathbf{w}) = \frac{1}{2}(\mathbf{w} \cdot \mathbf{w}) + C(\sum_{i=1}^{N} \epsilon_i) \tag{2.38}$$

Por otro lado, los datos de entrada son mapeados a un espacio de características no lineal de alta dimensionalidad, en el cual el HSO es construido. Así, el producto punto puede ser representado por $k(\mathbf{x}, \mathbf{y}) \equiv \phi(\mathbf{x}) \cdot \phi(\mathbf{y}))$, siempre y cuando el kernel k satisface las condiciones de Mercer [Haykin 1999]. Finalmente se obtiene la función de clasificación:

$$f(\mathbf{x}) = sing(\sum_{i=1}^{N} \alpha_i y_i k(\mathbf{x}_i, \mathbf{x}) + b). \tag{2.39}$$

Las SVM pueden ser analizadas teóricamente usando conceptos de la teoría del aprendizaje estadístico, lo que les da ventajas cuando son aplicadas a problemas donde el conjunto de patrones es limitado y se tiene una alta dimencionalidad. Consecuentemente, se ha observado que en algunas aplicaciones son clasificadores altamente efectivos [Tong 2001].

2.4.8.3 Optimización minima secuencial

Para encontrar el hiperplano óptimo de separación las SVM necesitan resolver un problema de programación cuadrática (QP), que involucra una matriz de densidad de NxN, donde N es el numero de patrones en la ME [Platt 1999]. Como tiene una complejidad cuadrática, se requieren gran cantidad de tiempo computacional y memoria para grandes conjuntos de datos. Para resolver este problema se han diseñado estrategias como LibSVM, Simple Support Vector Machine (SSVM) y Optimización Mínima Secuencial (SMO, *Sequential Minimal Optimization*), siendo este ultimo el más utilizado.

El algoritmo de Optimización Mínima Secuencial [Platt 1998] es obtenido a partir de la idea del método de descomposición a su extremo, al optimizar un subconjunto mínimo de únicamente dos puntos en cada iteración. El poder de esta técnica reside en el hecho de que el problema de optimización para dos puntos admite una solución analítica, eliminando la necesidad de usar un optimizador de programación cuadrática iterativo como parte del algoritmo [Joachims 1999].

El requisito de que la condición $\Sigma_{i=1}^{N} \alpha_i y_i = 0$ obliga en todo momento a que el número de multiplicadores que puede ser optimizado en cada paso es 2. Cada vez que un multiplicador es actualizado, por lo menos otro multiplicador necesita ser ajustado con el propósito de mantener la condición verdadera. En cada paso, SMO elige dos elementos α_i y α_j para optimizarlos, encuentra el valor óptimo de esos dos parámetros, dado que los demás se encuentran fijos y actualiza el vector α. La elección de los dos puntos es determinada por una heurística, mientras que la optimización de los dos multiplicadores es realizada analíticamente.

Experimentalmente, el desempeño de SMO es muy bueno para SVM con entradas escasas, así como para SVM no lineales. Esto es debido a que el tiempo de computación del kernel puede ser reducido, mejorando directamente su desempeño. A pesar de que necesita más iteraciones para converger, cada iteración usa solo pocas operaciones, por lo tanto converge muy rápido. Además del tiempo de convergencia, otra característica del algoritmo radica en que este no necesita almacenar la matriz del kernel en la memoria, ya que no se involucran operaciones matriciales. El algoritmo SMO se desempeña bien para problemas grandes, porque éste escala bien con el tamaño del conjunto de entrenamiento. Los autores de SMO aseguran que es un fuerte candidato para llegar a ser el algoritmo de entrenamiento estándar de SVM [Platt 1998].

Capítulo 3

Complejidad de los datos

Algunas de las RNA pertenecen al conjunto de modelos con aprendizaje supervisado, por lo que necesitan de un conjunto de datos con el cual se entrenara la red y esta aprenderá de ellos, a este conjunto se le llama comúnmente muestra de entrenamiento. Aunque, estas redes son muy utilizadas en muchas tareas de aprendizaje automático, reconocimiento de patrones se ha observado lentitud en su aprendizaje y una pobre capacidad de generalización en un numero importante de aplicaciones practicas [Visa 2005, Barandela 2000]. Estos dos problemas: incrementar la rapidez del procedimiento de entrenamiento del clasificador y mejorar la precisión en el resultado de sus decisiones, han motivado un esfuerzo importante en investigación de criterios más adecuados para definir parámetros y algoritmos vinculados al proceso de aprendizaje [Atiya 1997, Lee 1997].

También, surgió el interés por examinar la calidad de la muestra de entrenamiento y la validez de sus elementos, varios investigadores han coincidido en que la precisión de estos clasificadores depende de la calidad de los datos [Barandela 2001, Barandela 2000]. Por ejemplo, Barandela en [Barandela 2001, Barandela 2002] dice que el conjunto de datos para entrenar la red contiene patrones atípicos y ruidosos que la afectan negativamente, Visa en [Visa 2005] tratan varios factores como lo es el desbalance (el problema más grave en redes neuronales [Anand 1993, Murphey 2004, Bruzzone 1997b]), el tamaño de la muestra de entrenamiento y el solapamiento de clases.

3.1 La complejidad de los datos

De manera general, se definirá a la complejidad de los datos como todos aquellos factores o características de la muestra de entrenamiento (ME) que reducen su calidad y que por consiguiente disminuyen la precisión del clasificador. Estos factores se define como sigue:

- **Ruido.** Son patrones con errores, originados en su medición o registro (mal etiquetados, es decir, cuando no pertenecen a la clase donde fueron colocados). En

[Barandela 2001], se habla del ruido (Ver Fig. 3.1.a) como situaciones imperfectamente supervisadas. También en [Murphey 2004] se habla sobre el ruido y propone algunas métricas para medirlo.

- **Patrones Atípicos.** Son excepciones a la regla, es decir, que aunque han sido identificados correctamente estos son muy diferentes al resto de los patrones de la misma clase (en su tratamiento comúnmente son confundidos como ruido). En [Barandela 2001], se habla sobre el efecto negativo sobre la presencia de patrones atípicos (Ver Fig. 3.1.b).

- **Solapamiento de clases.** Es cuando dos o más clases se encuentran interceptadas entre sí compartiendo elementos en común. En [Prati 2004, Visa 2003, García 2006], se define que es el solapamiento (Ver Fig. 3.1.c), sus efectos sobre el clasificador y algunas soluciones a este problema (algunas basadas en el clasificador del vecino mas cercano).

- **Desbalance de clases .** Sucede cuando una clase tiene más patrones que las otras, es decir, cuando una clase es demasiado pequeña respecto a las demás (Ver Fig. 3.1.d). Dentro del desbalance de clases lo que se estudia es el problema de las **clases poco representadas**, porque no es lo mismo tener un conjunto de datos con 5 patrones en una clase 1 y 95 patrones en la clase 2 que tener 50 patrones en la clase 1 y 950 patrones en la clase 2 ya que en el primer caso la clase 1 no tiene la suficiente información para representarla [Visa 2005]. El desbalance de clases es considerado el problema más grave y por lo mismo el más estudiado por muchos investigadores del área como[Alejo 2007, Alejo 2008, He 2009, Visa 2005] por citar algunos.

- **Tamaño de la ME.** La cantidad de patrones es directamente proporcional al tiempo que tarda en aprender una RNA de todo el conjunto de estos, debido a esto, entre más grande (Ver Fig. 3.1.f) sea la ME más tiempo consumirá la red en el proceso de aprendizaje, agregando la carga computacional asociado a este proceso. Algunos autores [Lu 1998] no solo asocian el tamaño a la cantidad de patrones sino también a la cantidad de atributos de cada patrón. También, cuando el tamaño de la ME es pequeña (Ver Fig. 3.1.e) se vuelve un problema, ya que, tenemos el problema de la clases poco representadas. Según [Foody 1995], por cada atributo debe haber 30 patrones en cada clase.

- **Distribución de los datos.** Se refiere a como están ubicados los datos (patrones) en el espacio de características. La Fig. 3.1.g y 3.1.h muestran una distribución donde las clases son linealmente separables, la primera con una frontera ancha y clases compactas y la segunda con una frontera estrecha y extendida. En las Fig. 3.1.i y 3.1.j se observa una distribución que no es linealmente separable y por consiguiente representan un reto mayor para el clasificador.

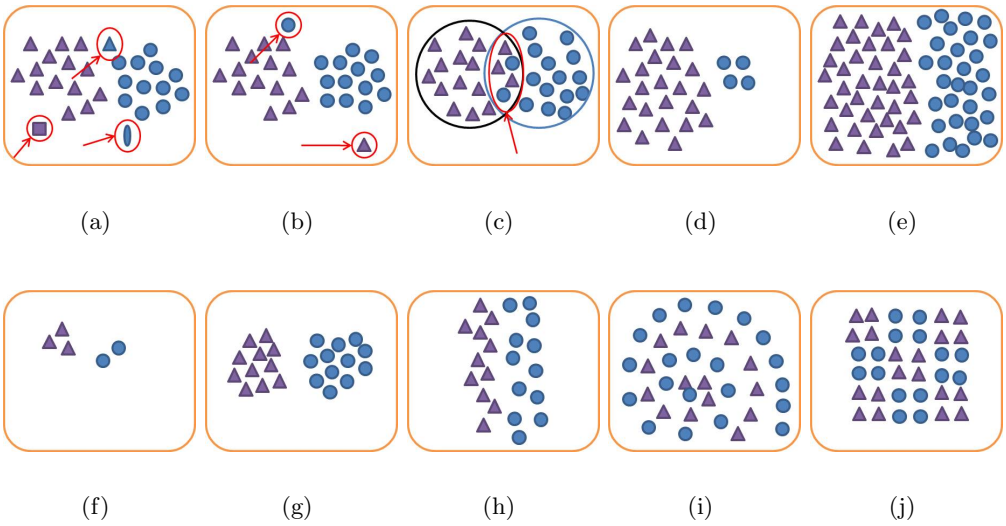


Figura 3.1: (a) Ruido, (b) Patrones atípicos, (c) Solapamiento de clases, (d)Desbalance de clases (e) Tamaño grande de la ME, (f)Tamaño pequeño de la ME, (g) ME linealmente separable con una frontera ancha y clases compactas, (h) ME linealmente separable con una frontera estrecha y clases extendidas, (i) ME con clases no linalmente separables, (j) Clases fuertemente entrelazadas siguiendo el diseño de un tablero de ajedrez.

3.2 Técnicas para reducir la complejidad de los datos

Para reducir o eliminar alguno de los factores mencionados en la sección 3.1 se han desarrollado una serie de técnicas y otras han sido extraídas del contexto de la regla del vecino más cercano [Haibo 2009, Visa 2005, Barandela 2000, Sánchez 2001], a continuación se describen las técnicas que su usaron en este proyecto. .

3.2.1 Técnicas de edición

En [Gopalakrishnan 1995, Barandela 2000] se habla del uso de técnicas basadas en la regla del vecino más cercano, para la eliminación de ruido en la ME, mejorando el rendimiento del clasificador. En la mayoría de las investigaciones se tratan en conjunto al solapamiento, ruido y patrones atípicos (englobaremos estos tres factores como limpieza de la ME). Los algoritmos más utilizados para limpiar la ME son los de Edición que consisten en descartar patrones que se encuentren en la región correspondiente a una clase distinta a la suya, es decir, patrones cuya probabilidad de pertenencia a su clase se vea superada por la probabilidad de pertenencia a alguna otra clase.

3.2.1.1 Edición de Wilson

La Edición de Wilson (NNE) [Wilson 1972], constituye la primera propuesta formal con el objetivo de reducir el conjunto de entrenamiento para la regla NN mediante la eliminación de prototipos erróneamente etiquetados. La idea de este método radica en que si un prototipo es erróneamente clasificado usando la regla k-NN es eliminado del conjunto de entrenamiento (TS), para este fin, se utilizarán todos los prototipos del conjunto de entrenamiento para determinar los k-vecinos más próximos (excepto el prototipo que se esta considerando en cada momento), es decir, el método de estimación del error empleado corresponderá al leaving-one- out.

Algoritmo 3.1: Algoritmo de edición de Wilson.

- Inicialización: $S \leftarrow$ ME
- Para cada elemento $\mathbf{x}_i \in$ ME
  - Buscar $K-$NN a $\mathbf{x}_i$ ($\mathbf{x}_j^k \in$ ME, $i \neq j$ y $\mathbf{x}_j^k = \min\|(\mathbf{x}_i, \mathbf{x}_j^k)\|$ donde $k = 1, ..., K$)
  - Si la mayoría de los $K-$NN son de distinta clase a $\mathbf{x}_i$, el elemento es eliminado de S
- Finalización: ME $\leftarrow S$

3.2.1.2 Vecindad del centroide más cercano

Una variante del algoritmo de la edición de Wilson es la vecindad del centroide mas cercano (*Nearest Centroid Neighborhood*, NCNE) [Sánchez 2001]. Esta técnica se basa en usar la regla *k-NCN* (regla de los k-Vecinos de Centroides más Cercanos) en lugar de la regla *k-NN* como clasificador central de la técnica, debido a que la misma obtiene mejores resultados, sobre todo cuando el conjunto de entrenamiento es relativamente pequeño. Es importante destacar que este algoritmo presenta el inconveniente de tener un costo computacional superior a la variante original lo que limita su utilización en determinados problemas reales [Sánchez 2001].

Sea $X = x_1, x_2, ..., x_n$ un conjunto de patrones, y sea p un cierto punto al que queremos encontrar sus k-vecinos de centroide más próximos, con este fin seguiremos el siguiente procedimiento iterativo, en el que el primer vecino del punto p corresponde a su vecino más próximo, mientras que los sucesivos vecinos se tomarán de manera que minimicen la distancia entre p y el centroide de todos los vecinos seleccionados hasta el momento. Así, si calculamos el k-ésimo vecino a partir de los $k-1$ vecinos previamente elegidos por el principio de centroide más próximo conseguiremos cumplir con los criterios de distancia y simetría. Todos los vecinos seleccionados se situarán

alrededor del punto p, es decir, de alguna forma se consigue que dicho punto quede envuelto por sus k-vecinos. Esta técnica se puede formalizar de la siguiente manera:

$$\delta_{k-NCN}(x) = \varpi_i \Leftrightarrow d(x, P_i) = min_{i=1,2,...,M} d_k(x, P_j), \tag{3.1}$$

Esta expresión significa, que la clase asignada a la muestra x corresponderá a la clase más votada entre los k-vecinos de centroide más cercano. El algoritmo se puede expresar de la siguiente manera:

Algoritmo 3.2: Algoritmo de la vecindad del centroide mas cercano.

- Inicialización: $S \leftarrow$ ME
- Para cada elemento $\mathbf{x}_i \in$ ME
  - Buscar los k vecinos de centroide más cercano de x_i en $X - x_i$
  - Si la mayoría de los $K-$NCN son de distinta clase a $\mathbf{x}_i$, el elemento es eliminado de S
- Finalización: ME $\leftarrow S$

3.2.1.3 Grafo de Gabriel

La técnica GGE (Grafo de Gabriel) [Sánchez 1997] establece que para un conjunto V de n puntos (refiriéndose a un vector) $V = \{p_1, p_2, ..., p_n\}$, dos puntos p_i y p_j son vecinos de Gabriel si:

$$dist^2(p_i.p_j) < dist^2(p_i, p_k) + dist^2(p_k, p_j) \qquad \forall k \neq i, j\,, \tag{3.2}$$

Uniendo todos los vecinos de Gabriel en pares mediante una arista se obtiene el grafo de Gabriel. En un sentido geométrico, los dos puntos p_i y p_j son vecinos de Gabriel, si el circulo con diámetro igual a la distancia entre p_i y p_j no contiene ningún otro punto $p_k \in V$.

3.2.1.4 Grafo de la vecindad relativa

La técnica Grafo de Vecindad Relativa (RNGE) [Sánchez 1997], basa su funcionamiento en que si dos punto x, y son relativamente cercanos sí el área definida por la intersección del circulo con centro x y radio xy, y el circulo con centro en y con el mismo radio no contiene otro punto. Formalmente, el grafo de la vecindad relativa de un conjunto de puntos S tiene una arista entre x e y si:

$$dist(x, y) \leq max[dist(x, z), dist(y, z)] \qquad \forall z \in S, z \neq x, y\,, \tag{3.3}$$

La desventaja de estos métodos es que no se tiene control sobre los patrones que se eliminan de la ME, y su uso en algunos casos aumenta el índice de error en proceso de aprendizaje [Haibo 2009]. Implícitamente se mejora la distribución de los datos porque aumenta la separabilidad entre clases al reducir el solapamiento, así como se disminuye el tamaño de la ME por la eliminación de patrones solapados, ruidosos y atípicos.

3.2.2 Técnicas basadas en muestreo y matrices de costos

Particularmente, el problema más grave y al cual la mayoría de los investigadores se han enfocado es el desbalance de clases [Visa 2005, Anand 1993, Murphey 2004, Lu 1998], y es que en la mayoría de los problemas reales los datos estas desequilibrados, por ejemplo, en los fraudes por vía telefónica (son mucho menos que las llamadas telefónicas normales) o al tratar con enfermedades como el SIDA donde hay menos personas contagiadas en comparación con las sanas. Se han propuesto varios técnicas para corregir el problema del desbalance en los datos, entre los más conocidos son los métodos de muestreo y los basados en costos.

Cuando se habla de corregir este problema por muestreo, se hace referencia principalmente al *over-sampling* y *under-sampling* [Haibo 2009]. La función principal del *over-sampling* es añadir a la clase minoritaria copias de los patrones de esta misma clase hasta equilibrar las clases de manera aleatoria, aunque Haibo et al. en [Haibo 2009] dice que la desventaja de usar este método es que se puede llegar a un sobre-ajuste [Mease 2007]. De forma contraria el *under-sampling*, elimina patrones de manera aleatoria de la clase mayoritaria hasta balancear la ME. Además se ha popularizado *SMOTE*, que es una variación del *over-sampling*, la diferencia consiste en el hecho de que ya no se duplican patrones aleatoriamente, si no se crean patrones sintéticos mediante la interpolación. Algunos autores, rechazan estas técnicas porque el *under-sampling* involucra pérdida de información (por descartar datos potencialmente útiles), y el *over-sampling* incrementa el tamaño de ME sin ningún aumento de información, por lo que algunos expertos sugieren que las investigaciones se enfoquen a los algoritmos de aprendizaje [Visa 2005].

Otro enfoque para combatir el problema del desbalance de clases, es el costo-sensitivo (*cost-sensitive*) [Zhou 2006]. Este se basa en la afirmación de que el precio de cometer un error de clasificación debe ser distinto para cada clase. En este método se implementa una matriz de costos para C clases (los valores dependen del problema que se esté tratando), donde $Cost[i,j](i,j \in [1...C])$ denota el costo de una clasificación errónea de un patrón de clase *i-ésima* en la clase *j-ésima*, por ejemplo *Cost[1, 2]=1* (ver Tabla 3.1). Además, durante los últimos años se ha popularizado el paradigma de aprendizaje en conjuntos [Valdovinos 2006],es decir, se entrenan varios clasificadores y se combinan sus predicciones. En este paradigma, cada clasificador es un aprendiz, entonces cada uno de estos votan por una clase y la clase que recibió más votos es la que se retorna.

Tabla 3.1: Ejemplo de una Matriz de Costo (*Cost[i, j]*) para una ME con tres clases.

		j		
		1	2	3
	1	0	1	8
i	2	1	0	9
	3	1	1	0

3.3 Criterios para medir la complejidad de los datos

El conjunto de datos o ME se puede estudiar desde el punto de vista geométrico (llamada Complejidad Geométrica), tal que, cada patrón es un punto de n-dimensiones en un espacio real $\Re_n$ y cada punto es etiquetado a una clase. se han investigado y propuesto algunas medidas para analizar el comportamiento y describir el desempeño de clasificador, varias de ellas se han surgido del análisis a la regla del vecino mas cercano y se han aplicado a RNA. Una métrica muy común, es la tasa de error del clasificador, sin embargo, la meta de muchos investigadores es encontrar otras medidas menos dependientes de clasificador para poderlas aplicar a otros. Además, estas métricas podrían dar indicios del origen de los errores y así mejorar el diseño del clasificador o depurar eficientemente la ME.

La complejidad geométrica (complejidad de los datos) agrupa múltiples factores que no pueden ser descritos por una sola métrica, por lo que, estas miden algunos de los factores en forma individual (la mayoría fueron diseñadas para ME de 2 clases). Estas métricas se pueden dividir en tres categorías y a continuación se describen [Mitra 2006].

3.3.1 Métricas del solapamiento sobre los valores de los atributos de las diferentes clases

Estas métricas se enfocan sobre la efectividad del atributo de un patrón en separar clases, examinan el rango y la propagación de los patrones dentro de cada clase y revisan el solapamiento entre las diferentes clases.

3.3.1.1 Indice Discriminante de Fisher (F1)

El índice discriminante de Fisher (F1) analiza que tan separadas están dos clases de acuerdo a un atributo dado, en otras palabras, calcula la efectividad de un atributo para separar las clases correspondientes. F1 para dos clases, es definido como:

$$f = \frac{(\mu_1 - \mu_2)^2}{\sigma_1^2 + \sigma_2^2}\,, \tag{3.4}$$

donde μ_1, μ_2, σ_1^2 y σ_2^2 son las medias y las varianzas de las dos clases respectivamente. Se calcula f para cada atributo y se toma el máximo para la métrica F1.

3.3.1.2 Volumen de la región de solapamiento (F2)

Esta métrica busca para cada atributo f_i el máximo ($max(f_i, c_j)$) y mínimo ($min(f_i, c_j)$) valor para cada clase c_j, después calcula el tamaño de la región de solapamiento normalizado por el tamaño de la región total donde están distribuidos todos los valores de ambas clases. F2 se define como:

$$F2 = prod_i^d \frac{MINMAX_i - MAXMIN_i}{MAXMAX_i - MINMIN_i}, \tag{3.5}$$

donde $i = 1, ..., d$ para un problema de d-dimensiones y

$MINMAX_i = MIN(max(f_i, c_1), max(f_i, c_2))$
$MAXMIN_i = MAX(min(f_i, c_1), min(f_i, c_2))$
$MAXMAX_i = MAX(max(f_i, c_1), max(f_i, c_2))$
$MINMIN_i = MIN(min(f_i, c_1), min(f_i, c_2))$

3.3.1.3 Máxima eficiencia de cada atributo (F3)

Mide la eficiencia de cada atributo en forma individual que describe cuánto cada característica contribuye a la separación de dos clases. La eficiencia de cada característica es definida como la fracción de todos los puntos restantes separables por esa región. Para representar la contribución de la mayoría de las características útiles, se usa la máxima eficiencia de característica como la métrica F3.

3.3.2 Métricas para la separación de clases

Estas métricas evalúan que tanto, dos clases son separables examinando la existencia y forma de la frontera de clase. La contribución de las dimensiones de características en forma individual están combinadas y resumidas en una sola puntuación, usualmente una medida de distancia, en vez de evaluarlo separadamente.

3.3.2.1 Suma minimizada de la separación del error por programación lineal(L1)

Los clasificadores lineales se pueden obtener por una formulación en programación lineal propuesta por Smith [Smith 1968]. El método minimiza la suma de la distancias entre los puntos de error para el hiperplano que loa separa (substrayendo un margen constante):

$$\text{minimize } a^t t$$

$$\text{sujeto a } Z^t w + t \geq b \text{ y } t \geq 0$$

Donde a, b son vectores de constantes arbitrarias (de preferencia 1), w es un vector de pesos a determinarse, t es un vector de error y Z es una matriz donde cada columna z es definido sobre el vector de entrada x (ampliada por la adición de una dimensión con el valor constante 1) y su clase c (con el valor c_1 y c_2) como sigue:

$$z = +x \ \textit{if}\, c = c_1$$

$$z = -x \ \textit{if}\, c = c_2$$

El valor de la función objetivo en esta formulación es usada como medida L1. Esta medida tiene un cero cuando el problema es linealmente separable, su valor puede ser afectado en gran medida por valores atípicos que surgen del lado equivocado del hiper-plano óptimo. La medida es normalizada por el número de puntos en el problema y también por la longitud de la diagonal del hiper-rectángulo que encierra todos los puntos de entrenamiento en el espacio de característica.

3.3.2.2 Índice de error del clasificador lineal por programación lineal (L2)

Métrica del índice de error del clasificador lineal definido por L1, medido con el conjunto de entrenamiento. Con un conjunto pequeño de entrenamiento puede ser severamente desestimado por el índice de error verdadero.

3.3.2.3 Fracción de puntos en la frontera de clase (N1)

Este método construye un árbol (sin tomar en cuenta la clase) de expansión mínima sobre el conjunto de datos entero, y cuenta el número de puntos incidentes sobre la frontera entre las dos clases. La fracción de esos puntos sobre todos los puntos en el conjunto de datos es utilizado como una medida.

3.3.2.4 Índice de la distancia promedio intra/inter clase (N2)

Primero se calcula la distancia euclidiana de cada punto con su vecino más próximo dentro de la clase y también para su vecino más próximo fuera de la clase, se calcula el promedio de las distancias de los vecinos más próximos dentro de cada clase (intercalase) y el promedio (sobre todos los puntos) de las distancias de los vecinos más próximos fuera de la clase. El índice sobre los dos promedios es utilizado como medida N2.

3.3.2.5 Índice de error del clasificador 1-NN (N3)

Ésta es simplemente la tasa de errores de un clasificador del vecino próximo medido con el conjunto de entrenamiento. El índice de error es estimado por el método *leave-one-out*.

3.3.3 Métricas de la geometría, topología y densidad de *manifolds*

Estas medidas dan caracterizaciones indirectas de la separabilidad de clases. Estas asumen que una clase están hechos un solo o múltiples *manifolds* que forman el soporte de la distribución de probabilidad de una clase dada. La forma, la posición y la interconexión de los *manifolds*. dan indicios de que tan bien están separadas dos clases.

3.3.3.1 No linealidad del clasificador lineal por programación lineal (L3)

Hoekstra y Duin [Hoekstra 1996] propusieron una medida para la no linealidad de un clasificador con relación a un conjunto de datos dado. Dado un conjunto de entrenamiento, la primera parte de método crea un conjunto de prueba por interpolación lineal (con coeficientes aleatorios) entre pares de puntos aleatorios de la misma clase. Luego el índice de error del clasificador (entrenada por el conjunto dado de entrenamiento) es medido sobre el conjunto de prueba.

3.3.3.2 No linealidad para el clasificador 1-NN (N4)

Ésta es la medida de no linealidad, definida para L3 y calculado para un clasificador del vecino más próximo.

3.3.3.3 La fracción de puntos asociados al subconjunto retenido (T1)

Esta medida se originó de un trabajo en describir formas de los monifolds de clase usando la noción de subconjuntos de adherencia en la pre-topología. Simplemente, cuenta el número de pelotas necesitadas cubrir cada clase, dónde cada pelota está centrada en un punto de entrenamiento y crece hasta alcanzar el máximo tamaño sin que esta toque otra clase. Las pelotas redundantes que recaen completamente sobre el interior de otras pelotas son eliminadas. Normalizamos la cuenta por el número total de puntos.

3.3.3.4 Promedio del número de puntos por dimensión (T2)

Éste es un índice simple del número de puntos en el conjunto de datos sobre el número de dimensiones características. Esta medida se incluye sobre todo para las conexiones con estudios previos sobre el tamaño de las muestras.

Capítulo 4

Metodología

En este trabajo se estudia la complejidad de los datos y su efecto en las redes neuronales MLP y RBF, aunque por motivos exploratorios y de comparación también se analizo su efecto sobre el clasificador SVM. El problema se abordo de la siguiente manera:

1. Se realizó una investigación sobre que características de las muestras de entrenamiento afectan la precisión del clasificador, con el objetivo de formular una definición de **complejidad de los datos**.

2. Se retomaron algunas técnicas (las cuatro técnicas mencionadas en la sección 3.2.1) aplicadas al clasificador de la regla del vecino más cercano para aplicarlas a las RNA así como algunas métricas de complejidad de la sección 3.3, (solo F1, F2, F3, N2 y N3) con el objetivo de disminuir la complejidad de los datos. Cabe destacar que en este trabajo solo se abordaron el tamaño, ruido, patrones atípicos y solapamiento de clases, dejando para investigaciones posteriores los demás factores.

3. Se análizaron las técnicas de edición para modificarlas y mejorar sus desempeño en la reducción de la complejidad de los datos. La técnica original propone que si la mayoría de vecinos de un dado patrón pertenecen a otra clase este patrón se elimina, ahora en este trabajo se modifico de tal manera que si por lo menos un vecino pertenece a otra clase el patrón examinado se elimina de la ME.

4.1 Descripción de los datos

Con la finalidad de validar las nuevas propuestas presentadas en este trabajo con otras investigaciones relacionadas, los resultados aquí reportados corresponden a experimentos realizados sobre conjuntos de datos que se emplean con gran frecuencia en publicaciones de diversas áreas del reconocimiento de patrones. Estos experimentos fueron desarrollados con conjuntos de datos extraídos del UCI Database Repository

[Newman 1998] y el proyecto Elena [Guérin-Dugué 1995]. La descripción de las bases de datos son las siguientes:

- **Australian**. Esta base de datos hace referencia a aplicaciones con tarjetas de crédito Todos los nombres de los atributos y valores se han sido cambiados por símbolos sin sentido para proteger la confidencialidad de los datos.

  Este conjunto de datos es interesante porque hay una buena mezcla de los atributos - continuos, nominales con un número reducido de valores, y nominales con un número mayor de valores. También hay algunos valores que faltan. En este trabajo los datos faltantes han sido tratados según las técnicas presentadas en [Han 2006].

- **Balance**. Este conjunto de datos se genero a partir de los resultados experimentales de un modelo psicológico. Cada ejemplo es clasificado como el extremo de una balanza (derecha, izquierda) o un punto de balance. Los atributos son el peso a la izquierda, la distancia a la izquierda, el peso de la derecha, y la distancia derecha. La forma correcta de encontrar la clase es el mayor de (distancia-izquierda * peso-izquierda) y (distancia-derecha * peso-derecha). Si son iguales, es equilibrada. Cada clase esta compuesta de 49 (equilibrada), 288 (izquierda) y 288 (derecha) muestras.

- **Cayo**. Este conjunto de datos representa un cayo de una región del golfo de México. Esta distribuida en 11 clases y un total de 6020 muestras etiquetadas y cuatro características. Las clases están distribuidas de la siguiente forma: Nubes (838 muestras), Sombras (293 muestras), Bosque (624 muestras), Yanal (322 muestras), Caminos (133 muestras), Arenas (369 muestras), Mangle (324 muestras), Pantanos (722 muestras), MarNorte(789 muestras), MarSurCercano (833 muestras) y MarSurLejano (772 muestras).

- **Pima Indian Diabetes**. En este conjunto de datos se debe identificar si las pacientes son diabeticos o no de acuerdo a los criterios establecidos por la Organización Mundial de la Salud. Esta base de datos esta distribuida en dos clases con 500 (no diabetica) y 268 (diabetica) muestras respectivamente. Cada muestra esta representada por un vector de 8 dimensiones.

- **Ecoli6**. Fue obtenida a partir de Ecoli [Newman 1998]. Esta última es una base de datos biológica creada por el *Institue of Molecular and Cellular Biolgy* [1] de la universidad de Osaka, Japón. Esta distribuida en 8 clases. Para este trabajo y por consideraciones prácticas se eliminaron las clases 7 y 8 que corresponden a las clases *imL* (inner membrane lipoprotein) y *imS* (inner membrane, cleavable signal sequence) dado que cada clase sólo disponen de dos muestras, y este hecho dificulta la aplicación eficiente del método de validación cruzada.

[1] Información detallada sobre la distribución de los datos, precisión de clasificación y características específicas de la base de datos Ecoli se puede encontrar en [Nakai 1991] y [Horton].

- **Feltwell**. Hace referencia a una sección (de 250x350 píxeles) de una imagen de percepción remota de una zona reservada para la agricultura cercana a la villa de Feltwell (Reino Unido) [Serpico 1993]. Para cada píxel se seleccionaron 15 características en función de un criterio heurístico y un análisis de correlación. Se seleccionaron 10944 píxeles que pertenecen a cinco clases relacionadas a la agricultura: remolacha de azúcar (3531 píxeles), hojarasca (2441 píxeles), suelo (896 píxeles), patatas (2295 píxeles) y zanahorias (1781 píxeles). Para más detalle véase [Roli 1996].

- **German Credit**. Esta base de datos sirve para identificar potenciales clientes con bajo o alto riesgo crediticio en función de su solvencia económica. Incluye características como la edad, el estado de la cuenta de ahorro del cliente, tipo de automóvil, empleo, entre otras. En [Newman 1998] están disponibles dos conjuntos de datos de German Credit. Uno con datos mezclados (categóricos y numéricos) y otro con datos numéricos únicamente. En este trabajo se hace referencia a la base de datos numérica que cuenta con 700 clientes buenos y 300 clientes malos identificados por 24 características.

- **Glass**. Glass es una base de datos con información sobre tipos de cristales. Su estudio está motivado por la investigación criminológica, pruebas de cristales en el lugar del crimen. Consta de 214 ejemplos, 10 atributos (clase incluida), 7 clases. Los atributos indican, en este orden: Índice de refración, cantidad de Na, Mg, Al, Si, K, Ca, Ba, Fe. Las clases representan tipos de cristal: Ventana de edificio procesada o no procesada, ventana de vehículo procesada o no procesada, recipiente, vajilla y lámparas.

- **Heart**. El conjunto de datos describe el diagnóstico cardíaco de un *Single Proton Emission Computed Tomography* (SPECT) sobre imágenes procesadas tomográficamente. Cada uno de los pacientes se clasifican en dos categorías: normales y anormales. La base de datos contiene 5 conjuntos de imágenes SPECT (pacientes), fueron procesadas para extraer características que representen las imágenes SPECT originales. Como resultado, 44 características fueron creadas para cada paciente. El algoritmo CLIP3 se utilizó para generar las reglas de clasificación de estos patrones. El algoritmo CLIP3 genera reglas con una precisión de 77.0% (en comparación con los diagnósticos cardiológicos).

  SPECTF es un buen conjunto de datos para probar algoritmos de aprendizaje automático, tiene 267 casos que se descritos por 45 atributos.

- **Liver**. Este conjunto de datos es conocido como BUPA Liver Disorders. Esta relacionado a problemas de hígado ocasionados por el consumo excesivo de alcohol. Fue generada por BUPA Medical Research Company. Incluye 345 muestras representadas por seis atributos y dos clases (con 200 y 145 muestras). Cada ejemplo corresponde a una muestra tomada a un hombre soltero.

- **Phoneme**. El conjunto de datos de Phoneme proviene del proyecto ELENA [Guérin-Dugué 1995]. Contiene vocales procedentes de 1809 silabas aisladas. Cinco atributos caracterizan a cada vocal. El objetivo de esta base de datos es distinguir entre vocales de la clase nasal y la clase oral (3818 y 1586 muestras, respectivamente).

- **Satismage**. Fue extraída de [Newman 1998] y corresponde a una imagen de percepción remota. Cada una de las 6435 muestras en esta base de datos está compuesto por cuatro bandas espectrales de la misma escena. El vector de características corresponde a una región cuadrada de 3x3 píxeles. Por lo tanto, las 36 características representan a cada uno de los nueve píxeles en cada una de las cuatro imágenes espectrales. Esta dividida en 6 clases con 1533, 703, 1358, 626, 707, y 1508 muestras respectivamente.

- **Sonar**. Se trata de discernir entre señales de sonar rebotadas en rocas (97) y de las rebotadas de cilindros metálicos (111). Ambas obtenidas desde distintos ángulos y condiciones. Cada muestra es un vector de señales de 60 características en el rango de [0, 1] y representan la energía para una banda de frecuencia integrada durante un determinado lapso de tiempo.

En las Tablas 4.1 y 4.2 se resumen las características más relevantes de los conjuntos de datos de dos y de múltiples clases respectivamente.

Tabla 4.1: Bases de Datos de Dos Clases.

Base de Datos	Clases	Atributos	ME	TST
Australian	2	42	552	138
Diabetes	2	8	614	154
German	2	24	800	200
Heart	2	25	217	55
Liver	2	6	276	69
Phoneme	2	5	4323	1081
Sonar	2	60	166	40

A cada conjunto de datos se le aplicó la técnica *k-fold-cross-validation*[Kuncheva 2004] con un valor de $k = 5$ en un principio y $k = 10$ para los experimentos posteriores, obteniéndose k muestras de entrenamiento (ME o TS) y k muestras de evaluación (TST). El tamaño de la ME fue de un 80% y 90% según el valor de k y el tamaño de la TST fue del 20% y 10%, respectivamente. Los resultados mostrados en este trabajo corresponden al promedio obtenido de las k particiones.

Después de aplicar el *k-fold-cross-validation* al conjunto de datos, a cada partición se le aplicaron las técnicas descritas en la sección 3.2.1, cabe señalar que solo se aplicaron

Tabla 4.2: Bases de Datos de Múltiples Clases.

Base de Datos	Clases	Atributos	ME	TST
Balance	3	4	500	125
Cayo	11	4	4815	1204
Ecoli6	6	7	265	67
Fetwell	5	15	8755	2189
Glass	6	9	171	43
Satismage	6	36	5148	1287

a la ME mientras que la TST quedo intacta. Con base a esto, se obtuvieron cuatro versiones adicionales al ME original con las cuales se entrenaron los clasificadores y se evaluó su desempeño mediante la muestra TST. Con el objetivo de estudiar si las técnicas aplicadas eran eficientes se creo una muestra adicional basada en la eliminación aleatoria de patrones a partir de ME original, es decir, primero se analizo que técnica eliminaba más patrones y se calculo en número de patrones eliminados, entonces, para crear la ME a la llamaremos USR (under sampling random) se elimino de la ME original esa la misma cantidad de patrones pero de forma aleatoria.

4.2 Configuración de los clasificadores

Los experimentos se realizaron mediante la herramienta Weka [Ian 2005], que es una colección de algoritmos orientados a la extracción de conocimiento desde bases de datos con grandes cantidades de información. La configuración de los clasificadores (configuración por defecto) fue la siguiente:

- Para el MLP el numero de capas ocultas se calculo a partir de la formula $a = (atributos + clases)/2$, la razón de aprendizaje es 0.3, el *momento* fue de 0.2, el número de repeticiones fue de 500. El algoritmo de aprendizaje fue el *backpropagation*.

- La red RBF utilizó el algoritmo de agrupamiento k-means para el calculo de los clusters. El número de clusters que genero el k-means fue de 2.

- Se utilizo el clasificador SVM con la implementación del algoritmo de optimización mínima secuencial de John Platt [Platt 1999] el cual recibe el nombre de SMO (ver sección 2.4.8.3).

4.3 Criterios de evaluación de resultados

Generalmente, el desempeño de los clasificadores es cuantificado a partir del cálculo del error de clasificación. El error de clasificación consiste en identificar el número de errores cometidos por el clasificador C. Considérese un conjunto disponible de datos etiquetados ($\mathbf{X}$) para estimar el error de clasificación, entonces la forma más natural de realizar esta tarea es:

$$Error(C) = \frac{N_{error}}{N_x}\,, \tag{4.1}$$

donde N_{error} es el número de identificaciones erróneas cometidas por C y N_x la cantidad de muestras en $\mathbf{X}$. La Precisión en la Clasificación (PC) se obtiene de la siguiente forma:

$$PC(C) = 1 - Error(C)\,, \tag{4.2}$$

4.3.1 Matriz de Confusión

Es común evaluar la precisión del clasificador en forma de matriz de error, comúnmente denominada matriz de confusión [Russell 1999]. La estructura de la matriz es tal que las columnas representan las clases reales mientras que las filas representan las clases predichas, en ella estas contenidas las asignaciones correctas (elementos de la diagonal, donde $i = j$) y las incorrectas (elementos fuera de la diagonal, donde $i \neq j$).

Tabla 4.3: Matriz de confusión. K es el número de clases y N el total de elementos.

	Clases Reales				
Clases predichas	1	2	...	K	total (N_{i+})
1	N_{11}	N_{12}	...	N_{1K}	N_{1+}
2	N_{21}	N_{22}	...	N_{2K}	N_{2+}
.	.	.		.	.
.	.	.		.	.
.	.	.		.	.
K	N_{K1}	N_{K2}	...	N_{KK}	N_{K+}
total (N_{+j})	N_{+1}	N_{+2}	...	N_{+K}	N

Dentro de la matriz de confusión están contenidos los errores por omisión y por comisión. Los errores por omisión son aquellos elementos que perteneciendo a una clase x no aparecen en ella por estar incorrectamente incluidos en otra. Los errores por comisión lo forman los elementos que no pertenecen a una clase y pero que aparecen en ella. A partir,de esta matriz podemos obtener varios datos importantes, de los cuales destaca la **Precisión Global y por Clase** que son los criterios que se utilizaron en

este trabajo para evaluar el desempeño del clasificador, a continuación se describen brevemente:

- La precisión global (PG) se obtiene a partir de la siguiente formula

$$PG = \frac{\sum_{i=1}^{K} N_{ij}}{N} \, i = j, \tag{4.3}$$

  donde N es el total de elementos. Obsérvese que esta ecuación es equivalente a la Ec. 4.2

- La precisión por clase (PXC) para la clase j se obtiene mediante

$$PXC_j = \frac{N_{jj}}{Ni+}\,, \tag{4.4}$$

En [Ariza 1996] se establecen las condiciones necesarias para la construcción de la matriz de confusión:

1. Las clases que se establezcan deben ser independientes, mutuamente excluyentes y exhaustivas.
2. Deben usarse métodos de muestreo que excluyan auto-correlación.
3. Conviene el uso de métodos estratificados para asegurar la presencia de clases extrañas o minoritarias.
4. Para comprobar la bondad de un proceso de clasificación supervisado no se deben usar los elementos de entrenamiento del clasificador.

4.3.2 Evaluación del producto final

Para construir y evaluar la capacidad de generalización del clasificador se han presentado diversas propuestas [Duda 2001a] basadas en la separación de los datos para construir y estimar la probabilidad de error o acierto del producto final. En ellas, el conjunto de datos X juega un papel fundamental a la hora de cuantificar la efectividad del clasificador. Algunas de las principales alternativas se resumen en [Kuncheva 2004]. A continuación se describe la validación cruzada, la cual se utilizó en este trabajo.

Consiste en dividir a X en k subconjuntos disjuntos. Donde $k-1$ subconjuntos serán utilizados como datos de entrenamiento (ME) y el resto como datos de evaluación (TST). Esto se realiza de tan manera que cada unos de los subconjuntos k actué como TST una sola vez y el resto sea utilizado como ME. El trabajo termina hasta que cada subconjunto haya participado como muestra de evaluación una sola vez. En este caso se utilizará la variante $k - fold - crossvalidation$ con un valor de $k = 5$ y $k = 10$.

Capítulo 5

Resultados (Publicaciones)

En este capítulo se muestran los resultados de la investigación realizada. Los resultados se resumen en cuatro artículos enviados y aceptados en congreso o revista. A continuación se incluyen los artículos en su versión final y se da una descripción general sobre cada artículo.

5.1 Complejidad de los datos en las redes neuronales artificiales: estado de la cuestión

El primer artículo que se titula "Complejidad de los datos en las Redes Neuronales Artificiales: Estado de la Cuestión" fue enviado y aceptado en el **Séptimo Congreso Internacional de Cómputo en Optimización y Software** (CICOS2009)[1], organizado por la Universidad Autónoma del Estado de Morelos[2].

En el articulo se propone la definición de complejidad de los datos planteada en este trabajo, así como, las principales aportaciones que se han hecho para mitigar algunas de las características incluidas en complejidad de los datos. Este artículo fue presentado como ponencia y fue publicado en el libro del congreso. A continuación se incluye el articulo completo.

[1]Para mas información visitar `http://campusv.uaem.mx/cicos/imagenes/memorias/7mocicos2009/7moCongreso.html`, ultima consulta 23/05/2011

[2]artículo en linea, `http://campusv.uaem.mx/cicos/imagenes/memorias/7mocicos2009/Articulos/24%20%20Complejidad%20de%20los%20Datos%20en%20Redes.pdf`

Complejidad de los datos en las Redes Neuronales Artificiales: Estado de la Cuestión

P. Toribio[1], B. G. Rodriguez[1], R. Alejo[2]

[1] Centro Universitario UAEM Atlacomulco, Universidad Autónoma del Estado de México, Km. 60 Carretera Toluca-Atlacomulco (México).
[2] Dept Llenguatges I Sistemes Informátics, Universitat Jaume I, Av. Sos Baynat S/N, 12071 Castelló de la Plana (Spain).

Resumen. En la actualidad, las redes neuronales artificiales son ampliamente utilizadas para tareas de reconocimiento de patrones, minería de datos y aprendizaje automático. Lamentablemente, su aplicación a problemas reales todavía está restringida por algunas debilidades como lentitud en el proceso de aprendizaje y la pobre capacidad de generalizar que han mostrado en múltiples aplicaciones prácticas.

En este trabajo se estudian las principales características de los datos de entrenamiento que tienen un efecto negativo en el proceso de aprendizaje de la red neuronal, por ejemplo: el ruido, el solapamiento de clases, desbalance de clases, principalmente. Actualmente, la complejidad de los datos ha atraído la atención de numerosos investigadores. Se ha observado que la capacidad de generalización y la convergencia de la red dependen en gran medida del nivel de complejidad en los datos de entrenamiento

Palabras clave: Redes Neuronales Artificiales, costo-sensitivo, over-sampling, under-sampling, Wilson.

1 Introducción.

Hoy en día el área de reconocimiento de patrones, aprendizaje automático y minería de datos han despertado el interés de muchos investigadores, ya que, se han convertido en herramientas para automatizar procesos en diferentes áreas como la medicina, la industria, la biología, etc. Por ejemplo, la minería de datos ha surgido como alternativa para la extracción del conocimiento de grandes cantidades de datos con la finalidad de tomar decisiones o comprender algún fenómeno social o de la naturaleza.

Actualmente, las Redes Neuronales Artificiales (RNA) son muy utilizadas como clasificadores en tareas de minería de datos, aprendizaje automático y reconocimiento

M.A Cruz-Chávez, J.C Zavala Díaz(Eds):CICos2009, ISBN:978-607-00-1970-8, pp. 229 - 235, 2009.

de patrones. El Perceptron Multicapa (PM) ha sido utilizado en la interpretación de imágenes de lectura remota [1] y como aproximador universal, principalmente. Sin embargo, aún se conoce poco de estos modelos, lo que trae como consecuencia, lentitud en el proceso de aprendizaje y pobre capacidad de generalización. Varios investigadores han coincidido en que la precisión de estos clasificadores depende de la calidad de los datos [2, 3, 4, 16], porque las RNA fueron diseñadas para aprender mediante bases de datos con clases balanceadas y también con datos que puedan ser linealmente separables, o por lo menos que para este tipo de clasificador sea fácil transformar el espacio no linealmente separable a uno linealmente separable, por lo tanto todos aquellos factores o características del conjunto de datos que impiden que una RNA aprenda correctamente reciben el nombre de complejidad de los datos [2, 3, 5, 6, 7].

Cada investigador define de forma diferente o engloba diferentes factores en complejidad de los datos, por ejemplo, Barandela en [3, 4] dice que el conjunto de datos para entrenar la red contiene patrones atípicos y ruidosos que la afectan negativamente, Visa *et al.* en [2] tratan varios factores como lo es el desbalance (el problema más grave en redes neuronales) [5, 6, 7, 8], el tamaño de la muestra de entrenamiento y el solapamiento de clases (ver Fig. 1).

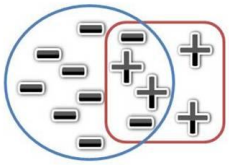


Fig. 1. Solapamiento de clases

2 Complejidad de los datos.

Como describimos en la sección anterior la complejidad de los datos son aquellos factores que disminuyen la calidad de la muestra de entrenamiento (ME) y que por consiguiente disminuyen la precisión del clasificador. Estos factores se pueden definir como sigue:

- **Ruido**: Son datos con errores, originados en su medición o registro (mal etiquetados, es decir, cuando no pertenecen a la clase donde fueron colocados).
- **Patrones Atípicos**: Son excepciones a la regla, es decir, que aunque han sido identificados correctamente estos son muy diferentes al resto de los patrones de la misma clase.
- **Solapamiento**: es cuando dos o más clases se encuentran interceptadas entre sí compartiendo elementos en común (ver Fig. 1).

- **Desbalance de clases**: sucede cuando una clase tiene más patrones que las otras, es decir, cuando una clase es demasiado pequeña respecto a las demás.
- **Tamaño de ME**: La cantidad de patrones es directamente proporcional al tiempo que tarda en aprender una RNA todo el conjunto de estos, debido a esto, entre más grande sea la muestra más tiempo consumirá la red en aprenderlos, agregando la carga computacional asociado a este proceso. Algunos autores [8] no solo asocian el tamaño a la cantidad de patrones si no también a la cantidad de atributos de cada patrón.
- **Distribución de la los datos**: la Fig. 2 muestra dos casos en donde se observa cómo están distribuidos los datos. En la Fig. 2a se ve una distribución que para una red neuronal seria la óptima. Aquí tomaremos a la linealidad dentro de la distribución de los datos, pues la primera depende de la segunda.

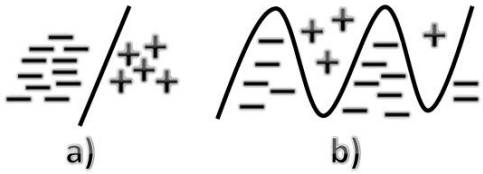


Fig. 2. Distribución de los datos, a) caso óptimo, b) caso no óptimo.

3 Métodos para tratar la complejidad de los datos

Particularmente, el problema más grave y al cual la mayoría de los investigadores se han enfocado es el desbalance de clases [2, 5, 6, 8, 10], y es que en la mayoría de los problemas reales los datos estas desequilibrados, por ejemplo, en los fraudes por vía telefónica (son mucho menos que las llamadas telefónicas normales) o al tratar con enfermedades como el SIDA donde hay menos personas contagiadas en comparación con las sanas. Se han propuesto varios métodos para corregir el problema del desbalance en los datos, entre los más conocidos son los métodos de muestreo [2, 10], basados en costo-sensitivo [11], los cuales se describen en la sección 3.1.

Para tratar el solapamiento de clases, patrones atípicos y ruido se utilizan métodos como la edición de Wilson, basados en la técnica del vecino más cercano (*nearest neighbor rule*, *NN*) y sus variantes, por ejemplo el *1-NN*, *k-NN* y *CNN*. También se han utilizados técnicas basadas en el criterio de vecindad como el Grafo de Gabriel (*GG*, *Gabriel Graph*) y Grafo de la vecindad relativa (*RNG*, *Relative Neighbourhood Graph*) [12], las cuales se presentan en un panorama más amplio en la sección 3.2. Cabe señalar que tanto estos métodos como los de muestreo para combatir el desbalance modifican la distribución de los datos, así como el tamaño de la ME, de esto podemos plantearnos la siguiente pregunta, ¿qué tan conveniente es modificar el tamaño y la distribución de los datos?

3.1 Métodos de muestreo (re-balancear la ME) y Costo-Sensitivo.

Cuando hablamos de corregir el desbalance de datos por muestreo, nos referimos principalmente al *over-sampling*, *under-sampling* [2, 10]. La función principal del *over-sampling* es añadir a la clase minoritaria copias de los patrones de esta misma clase hasta equilibrar las clases de manera aleatoria, aunque Haibo *et al.* en [10] nos dicen que la desventaja de usar este método es que se puede llegar a un sobre-ajuste [13]. De forma contraria el *under-sampling*, elimina patrones de manera aleatoria de la clase mayoritaria hasta balancear la ME. Además se ha popularizado *SMOTE*, que es una variación del *over-sampling*, la diferencia consiste en el hecho de que ya no se duplican patrones aleatoriamente, si no se crean patrones sintéticos mediante la interpolación [2]. Algunos autores, rechazan estos métodos porque el *under-sampling* involucra pérdida de información (por descartar datos potencialmente útiles), y el *over-sampling* incrementa el tamaño de ME sin ningún aumento de información, por lo que algunos expertos sugieren que las investigaciones se enfoquen a los algoritmos de aprendizaje [2].

Otro enfoque para combatir el desbalance de clases, es el costo-sensitivo (*cost-sensitive*) [11]. Este se basa en la afirmación de que el precio de cometer un error de clasificación debe ser distinto para cada clase [14]. En este método se implementa una matriz de costos para C clases (los valores dependen del problema que se esté tratando), donde $Cost[i,j](i,j,\epsilon\,[1\ldots C])$ denota el costo de una clasificación errónea de un patrón de clase *i*-ésima en la clase *j*-ésima, por ejemplo $Cost[1,2]=1$ (ver Tabla 1). Además, durante los últimos años se ha popularizado el paradigma de aprendizaje en conjuntos [11, 18], es decir, se entrenan varios clasificadores y se combinan sus predicciones. En este paradigma, cada clasificador es un aprendiz, entonces cada uno de estos votan por una clase y la clase que recibió más votos es la que se retorna.

Tabla 1. Ejemplo de una Matriz de Costo *(Cost[i, j])* para una ME con tres clases.

		j		
		1	2	3
i	1	0	1	8
	2	1	0	9
	3	1	1	0

3.2 Métodos para eliminar el solapamiento, ruido, patrones atípicos y solapamiento de clases.

En [15,16] se habla del uso de técnicas basadas en la regla del vecino más cercano, para la eliminación de ruido en la ME, mejorando el rendimiento del clasificador. En la mayoría de las investigaciones se tratan en conjunto al solapamiento, ruido y patrones atípicos (englobaremos estos tres factores como limpieza de la ME). Los algoritmos más utilizados para limpiar la ME son los de Edición que consisten en

descartar prototipos (patrones) que se encuentren en la región correspondiente a una clase distinta a la suya, es decir, prototipos cuya probabilidad de pertenencia a su clase se vea superada por la probabilidad de pertenencia a alguna otra clase. La Edición de Wilson [17], propone eliminar los elementos atípicos de la muestra de entrenamiento mediante la aplicación del procedimiento *k-NN* (*k* vecinos más cercanos), dado un patrón de prueba de la ME se buscan sus *k*-vecinos más cercanos y si la mayoría estos no pertenecen a la misma clase, el patrón de prueba se elimina. La Edición de WilsonCN [18] es una modificación al algoritmo de Wilson. Este método se basa en usar, en vez de la regla *k-NN*, la regla *k-NCN* (regla de los *k*-Vecinos de Centroides más Cercanos) como clasificador central del método, debido a que la misma obtiene mejores resultados, sobre todo cuando el conjunto de entrenamiento es relativamente pequeño. Es importante destacar que este algoritmo presenta el inconveniente de tener un costo computacional superior a la variante original lo que limita su utilización en determinados problemas reales [18].

Además de métodos basados en la regla del vecino más cercano y su variantes los métodos basados en el criterio de vecindad han dado buenos resultados para la limpieza de la ME, aunque su impacto sobre la precisión del clasificador no ha alcanzado lo esperado. A continuación se describe con más detalle el funcionamiento de estos métodos.

La técnica *GG* (Grafo de Gabriel) [18] dice que para un conjunto *V* de *n* puntos (refiriéndose a un vector) $V = \{p_1, p_2, \ldots, p_n\}$, dos puntos p_i y p_j son vecinos de Gabriel si:

$$\text{dist}^2(p_i, p_j) < \text{dist}^2(p_i, p_k) + \text{dist}^2(p_k, p_j) \qquad \forall k \neq i, j \qquad (\mathbf{1})$$

Uniendo todos los vecinos de Gabriel en pares mediante una arista se obtiene el grafo de Gabriel. En un sentido geométrico, los dos puntos p_i y p_j son vecinos de Gabriel, si el circulo con diámetro igual a la distancia entre p_i y p_j no contiene ningún otro punto $p_k \in V$.

La técnica Grafo de Vecindad Relativa (*RNG*) [18], basa su funcionamiento en que si dos punto *x, y* son relativamente cercanos sí el área definida por la intersección del circulo con centro *x* y radio *xy*, y el circulo con centro en *y* con el mismo radio no contiene otro punto. Formalmente, el grado de la vecindad relativa de un conjunto de puntos *S* tiene una arista entre *x* e *y* si:

$$\text{dist}(x, y) \leq \max[\text{dist}(x, z), \text{dist}(y, z)] \quad \forall z \in S, \; z \neq x, y \qquad (\mathbf{4})$$

La desventaja de estos métodos es que no se tiene control sobre los patrones que se eliminan de la ME, y su uso en algunos casos aumenta el índice de error en proceso de aprendizaje [2].

4 Conclusiones

En este artículo concluimos que la complejidad de los datos son todos aquellos factores que afectan la calidad de la muestra de entrenamiento y por consiguiente disminuyen la presión de clasificación de la RNA, estos factores son el ruido, patrones atípicos, solapamiento de clases, desbalance de clases, tamaño de la ME y la distribución de los datos.

Hemos discutido sobre el problema llamado complejidad de los datos y sus efectos en el entrenamiento de una red neuronal. También, describimos las principales soluciones utilizadas para tratar este problema, lo que nos ayuda a comprender la magnitud de este problema, así como la dirección que deben tomar las investigaciones futuras respecto a este.

Algunas investigaciones futuras que podemos abordar como parte de este artículo es la búsqueda de métricas que nos ayuden a medir los factores englobados en la complejidad de los datos, así como la modificación de las técnicas anteriores para obtener mejores resultados, como por ejemplo, hasta ahora varios de estos métodos mencionados se han aplicado en el espacio de entrada, pero ¿qué pasaría si se aplicaran en el espacio oculto de la red neuronal?, que es donde en realidad trabaja la red, ¿se mejoraría el desempeño de la red?.

Bibliografía

1. Foody, G. M., *Using prior knowledge in artificial neural network with a minimal training set*, Int. Journal of Remote Sensing, 16, 2, 301-312, (1995).

2. Visa, S., Ralescu, A., *Issues in Mining Imbalanced Data Sets - A Review Paper*, Proceedings of the Sixteen Midwest Artificial Intelligence and Cognitive Science Conference, 67-73, (2005).

3. Barandela, R., Gasca, E., Alejo R., *Corrección de la muestra para el aprendizaje del Perceptron Multicapa*, Revista Iberoamericana de Inteligencia Artificial No. 13, 2{9, (2001).

4. Barandela, R., Gasca, E., Alejo R., *Correcting the training data. Combinatorial Optimization*, -Dordrecht- 13, 1-4-2,(2002).

5. Anand, R., Mehrotra, K., Mohan, C., Ranka, S., *An improved algorithm for neural network classification of imbalanced training sets*, IEEE Transactions on Neural Networks, 4, (1993).

6. Murphey, Y., Guo, H., Feldkamp, L., *Neural learning from imbalanced data*, Applied Intelligence, 21, (2004).

7. Bruzzone, L., Serpico, S., *Classification of imbalanced remote-sensing data by neural networks*, Pattern Recognition Letter, 18, (1997).

8. Lu, Y., Guo, H., Feldkamp, L., *Robust neural learning from unbalanced data examples*, In: Proc. of IEEE International Joint Conference on Neural networks, (1998).

9. Sanchez, J. S., Mollineda, R. A., Sotoca, J. M., *An analisis of how training data complexity affects the nearest neighbor classifiers*, Springer-Verlag London, (2007).

10. Haibo He, Garcia, Edwardo A., *Learning from Imbalanced Data*, IEEE Transactions on knowledge and Data Engineering, Vol. 21, No. 9, (2009).

11. Zhi-Hua, Xu-Ying Liu, *Training Cost-Sensitive Neural Networks with Methods Addressing the Class Imbalance Problem*, IEEE Transactions on knowledge and Data Engineering, Vol. 18, No. 1, (2006).

12. Sánchez, Garreta, José Salvador, *Training Aprendizaje y clasificación basados en criterios de vecindad: Métodos alternativos y análisis comparativo*, Tesis Doctoral, Universitat Jaume I, (1998).

13. D. Mease, A. J., Wyner, and A. Buja, *Boosted Classification Trees and Class Probability/Quantile Estimation*, J. Machine Learning Research, Vol. 8, (2007).

14. Kukar, M., Kononenko, I., *Cost-sensitive learning with neural networks*, In: 13th European Conference on Artificial Intelligence, (1998).

15. Gopalakrishnan, M., V. Sridhar y H. KrishNamurthy. *Some application of clustering in the desing of neural networks*, Patter Recognition Letters, 16, 59-65, 1995.

16. Barandela, R y E. Gasca. *Decontamination of training samples for supervised pattern recognition methods*. En: Advances in Pattern Recognition, F. Ferri et al. (eds), Lecture Notes in Computer Science, 1876, Springer, 2000.

17. Wilson, D.L.: *Asymptotic properties of nearest neighbor rules using edited data sets*. IEEE Trans. on Systems, Man and Cybernetics 2 408-421, 1972.

18. Sánchez,J.S., F,Pla and F.J.Ferri, *Using the nearest centroid neighbourhood concept for editing purpose*, In Proc. VII Simposium Nacional de Reconocimiento de formas y Análisis de Imágenes 1, 175-180, 1997.

18. T. G. Dietterich, *Ensamble Learning*, The Handbook of Brain Theory and Neural Networks, second ed., M.A. Arbib, ed., Cambridge, Mass: MIT Press, 2002.

5.2 Optimización del entrenamiento de redes neuronales artificiales

Este artículo cuyo nombre es "Training Optimization for Artificial Neural Networks", fue aceptado en la revista **CIENCIA ergo sum**[3], publicada por el Universidad Autónoma del Estado de México.

Este muestra los primeros resultados de la aplicación de las técnicas de edición (Edición de Wilson, el grafo de Gabriel y el de la vecindad relativa) con el objetivo de disminuir tamaño y el costo computacional asociado al proceso de aprendizaje y también incrementar la convergencia del clasificador. Además, se entreno el SVM con la finalidad de comparar sus resultados con los del MLP y RBF. A continuación se incluye el articulo completo.

[3]artículo en linea, `http://redalyc.uaemex.mx/pdf/104/10415212010.pdf`

Toribio Luna, Primitivo; Alejo Eleuterio, Roberto; Valdovinos Rosas, Rosa María; Rodríguez Méndez, Benjamín Gonzalo

Training Optimization for Artificial Neural Networks

Ciencia Ergo Sum, vol. 17, núm. 3, noviembre-febrero, 2011, pp. 313-317
Universidad Autónoma del Estado de México
México

Ciencia Ergo Sum
ISSN (Versión impresa): 1405-0269
ciencia.ergosum@yahoo.com.mx
Universidad Autónoma del Estado de México
México

Training Optimization for Artificial Neural Networks

Primitivo Toribio Luna*, Roberto Alejo Eleuterio**, Rosa María Valdovinos Rosas***,
Benjamín Gonzalo Rodríguez Méndez*

Recepción: 11 de diciembre de 2009
Aceptación: 29 de abril de 2010

*** Innovación y estrategias tecnológicas Centro Universitario Atlacomulco, Universidad Autónoma del Estado de México, Atlacomulco, México.**
**** Universidad Jaume I, Castelló de la Plana, España.**
***** Centro Universitario Valle de Chalco, Universidad Autónoma del Estado de México, Valle de Chalco, México.**
Correo electrónico:guffyprimo@hotmail.com, ralejoll@hotmail.com, li_rmvr@hotmail.com y benrome@hotmail.com .
Acknowledgements:
This work has been partially supported by the Spanish Ministry of Science and Education under project CSD2007-00018, UAEMCA-114 and SBI112 from the Mexican SEP, FE28/2009 (103.5/08/3016) of the Mexican PROMEP and the 2703/2008U from the UAEM project.

Optimizaciòn del entrenamiento para Redes Neuronales Artificiales

Resumen. Debido a la habilidad para modelar problemas complejos, actualmente las Redes Neuronales Artificiales (NN) son muy populares en Reconocimiento de Patrones, Minería de Datos y Aprendizaje Automático. No obstante, el elevado costo computacional asociado a la fase en entrenamiento, cuando grandes bases de datos son utilizados, es su principal desventaja. Con la intención de disminuir el costo computacional e incrementar la convergencia de la NN, el presente trabajo analiza la conveniencia de realizar pre-procesamiento a los conjuntos de datos. De forma específica, se evalúan los métodos de grafo de vecindad relativa (RNG), grafo de Gabriel (GG) y el método basado en los vecinos envolventes k-NCN. Los resultados experimentales muestran la factibilidad y las múltiples ventajas de esas metodologías para solventar los problemas descritos previamente.

Palabras clave: redes neuronales artificiales, perceptrón multicapa, redes de función de base radial, máquinas de vectores soporte, pre-procesado de datos.

Abstract. Nowadays, with the capacity to model complex problems, the artificial Neural Networks (NN) are very popular in the areas of Pattern Recognition, Data Mining and Machine Learning. Nevertheless, the high computational cost of the learning phase when big data bases are used is their main disadvantage. This work analyzes the advantages of using pre-processing in data sets In order to diminish the computer cost and improve the NN convergence. Specifically the Relative Neighbor Graph (RNG), Gabriel's Graph (GG) and k-NCN methods were evaluated. The experimental results prove the feasibility and the multiple advantages of these methodologies to solve the described problems.

Key words: artificial neural networks, multilayer perceptron, radial basis function neural network, support vector machine, preprocessing data.

Introduction

The artificial Neuronal Networks (NN) has become popular in many tasks of Machine Learning, Pattern Recognition and Data Mining. For example, Multilayer Perceptron (MP) has been applied in remote sensing, prediction and approach of functions and control. The Radial Basic Function NN offered an alternative to the treatment formal of the NN allows to realize constructive procedures, that is to say, to determine the optimal structure of a neuronal network for a specific application (Barandela and Gasca, 2000) and are used in applications such as function approach, interpolation with noise and classification tasks. On the other hand, the Support Vector Machines (SMV) can be defined as a static network based on kernels, which realizes a linear classification on transformed vectors in a superior dimension space. That is to say, they are separated by means of a hyperplane in the transformed space. The MVS has been applied successfully in several problems related to Pattern Recognition, recognition of the writing and the natural language processing (Vaptnik, 1995). Unfortunately, in these models, the computational cost associated to the learning phase, depends of the Training Set (TS) size.

In this study, we analyze and explore several methods in order to modify and reduce the TS size, eliminating atypical or noisy training samples and correcting possible erroneous identifications of those training samples. These proposals have the goal of decrease the computer cost and to accelerate the learning process. The proposal is taken from the experience obtained with another non-parametric rule, such as the Nearest Neighbor Rule (Lippmann, 1988).

In the experimentation, we employed 14 real-problem databases with different classes number, 6 with two-classes and 6 of multi-class problem. The paper is organized in the following way: Section 2 and 3 give the theoretical background of the neural network and the nearest neighbor rule. The Section 4 shows the preprocessing algorithms here used as solution strategy and, the Section 5 provides the experimental results. Finally, the main conclusions of this work and possible lines for future research are commented in Section 6.

1. Neural Networks

Nowadays, the Multilayer Perceptron (MP) NN with backpropagation of the error is one of the most popular models for classification purposes (Haykin, 1999), (Jain et al., 1996). This model had among other aspects: the capacity for organizing the representation of the knowledge in the hidden layers and their high power of generalization. Typical architecture has three sequential layers: input, hidden and output layer (Cristianini and Shawe-Taylor 2000), (Dasarathy, 1995), (Sherstinsky and Rosalind, 1994). Such that, a MP with one layer can build a linear hyperplane, a MP with two layers can build convex hyperplane, and a MP with three layers can build any hyperplane.

By simplicity of their architecture and the training method, the Radial Basis Function (RBF) NN, is an attractive alternative for MP. The RBF NN is designed with one hidden layer; the neurons are activated by means of non linear radial functions (Gaussian), and in the output layer use linear functions (Sánchez and Alanís, 2006). In this way, the output RBF, is influenced by a nonlinear transformation produced in the hidden layer through the radial function and a linear one in the output layer through the linear continuous function.

Differences between these NN are following:

a) The RBF has a single hidden layer and the MP could had more than one.

b) The activation function of the hidden nodes is different, in the RBF, it depends of the distance between the input vectors and the centoids in the hidden layer, whereas in the MP depends on the product of the input vector and the weights vector.

c) Generally, the hidden layer nodes and the output layer nodes of MP have the same neuronal model while in the RBF it is different.

On the other hand, the Support Vectors Machines (SVM) base their operation on the data space transformation to other of higher dimension space, through of a kernel function, thus, this function finds the hyperplane that maximizes the margin of separation in the pattern classification of different classes (Cristianini and Shawe-Taylor, 2000).

This method has been successful, due to not to suffer the local minimums such as in the MP, the model only depends of data with more information called support vectors. The main advantages of the SVM are:

a) Excellent generalization capacity, because of the structured risk which could be dimmished (Vaptnik, 1995).

b) SVM adjusts few parameters, the model only depends of data with greater information.

c) The parameters estimation is realized through the function optimization with convex cost, which avoids the existence of a local minimum.

d) The solution of the SVM is to spar itself, this is, the majority of the variables are zero in the solution. Thus, the final model can be written like a combination of a very small number of entrance vectors, called support vectors.

2. The Nearest Neighbor Rule

The Nearest Neighbor Rule (NNR), is very popular in Patter Recognition, it bases its operation in considering the nearest patterns, like which they have the greater probability of belonging to a same class (Dasarathy, 1995). NNR has the following characteristics:

a) It is a supervised method, which needs a Training set, which assumes that it was composed by patterns perfectly identified and that represent all the interest classes in the problem.

b) It is a non-parametric method, that is to say, it does not dependent of probabilistic model for the data.

c) It suffers of a considerable computer cost, due to maintain the ME in memory and by examining each training sample in order to do the classification process.

3. Preprocessing algorithms

The NN are very much sensible to any deficiency in the quality and trustworthiness of the data set. In (John, 1997) suggests the cleaning data in order improve the precision levels in the classification process. On the other hand (Guha et al., 1998) defines a procedure to clean TS by means of an algorithm hierarchic non-supervised. A similar line is defended by (Go-

palakrishnan et al., 1995) in order to eliminate patterns that motivate slowness in the learning of the MP. On the other hand, (Barandela and Gasca, 2000) demonstrates the benefits to use a methodology based on the NNR to work with samples imperfectly supervised, producing a cleaning adapted of the TS and contributing to the yield of the classifier algorithm.

In this work several techniques based on the NNR and on its variant k-NNR were evaluated, for the pre-processing of the training data, such as the Gabriel's Graph and the Relative Neighbor Graph.

The Gabriel's Graph GG is used for editing the TS (Sánchez et al., 1997). In its operations use a proximity condition between two vertices. In the circumference formed by those two points, there must not be one other point inside. If that condition was true, then the edge would belong to the graph. For a certain V set of n points, where V = p1, p2,…, pn, two points p_i and p_j are Gabriel's neighbor if:

$$dist^2(p_i; p_j) < dist^2(p_i; p_k) + dist^2(p_k; p_j)\ k \neq i; j \qquad (1)$$

Joining all the Gabriel's neighbors in pairs, by means of an edge, the Gabriel graph is obtained. In a geometric sense, both points p_i and p_j are Gabriel neighbor, if the circle with equal diameter to the distance between p_i and p_j do not contain other point $p_k \in V$. The algorithm can be defined as follows:

1. For each pair of points $(p_i; p_j)$; $i; j = 1, 2, ..., n$; where $i < j$

2. If $dist^2(p_i; p_j) > dist^2(p_i; p_k) + dist^2(p_k; p_j)$, then, p_i, p_j are not neighbor of Gabriel, and, go to step 1.

3. Else p_i, p_j are marked as Gabriel neighbors.

The k-Nearest Centroid Neighbor rule (k-NCN) (Sánchez et al., 1997), is a modification to the Wilson's Editing. Nevertheless, this algorithm has higher computational cost than the Wilson's Editing, for that their use is limited to small TS. Is $X = x_i, x_j, ..., x_n$ a training set, and is p a certain point to which we want to find its nearest centroid k-neighbor.

Now, the first neighbor of p corresponds to his nearest neighbor, whereas the successive neighbors will be chosen if they diminish the distance between p and the centroid of all neighbors selected until this moment. This rule can be formalized as follows:

$$\delta_{k\text{-}NCN}(x) = \varpi \leftrightarrow d(x, P_i) = min_i\, d_k(x, P_j) \qquad (2)$$

where, $min_i\, d_k(x, P_j)$ is the distance between x and their k'th neighbor chosen by the k-NCN method. In this way, the class assigned to x will be the most voted between the k neighbors of the nearest centroid.

Finally, in this work we use the Relative Neighbor Graph (RNG) (Sánchez et al., 1997) method. In the RNG, an edge belongs to graph if the end points are relative neighbors, that is to say, when there are an intersection of two circumferences, the edges of the arcs and the radio of distance between them, the geometric figure which is formed in the intersection (its physical form is a moon, that is why this drawing is also known as moon) does not contain inside any other point. That is to say, the RNG of a certain points set S has an arc between x and y if:

$$dist(x, y) \leq [dist(x, z); dist(y, z)]\ z \in S, z \neq x, y \qquad (3)$$

4. Experimental results

The experiments were carried out on 12 real data sets taken from the UCI Machine Learning Database Repository (http://archive.ics.uci.edu/ml/). A brief summary is given in the Table 1. For each database, we have estimated the overall accuracy by 5–fold cross–validation: each data set was divided into five equal parts, using four folds as the training set and the remaining block as independent test set.

The experiments have been performed using the Weka Toolkit (Witten, I. and Frank, E, 2005) with the learning algorithms described in Section 2, that is, MLP, SVM and RBF. Each classifier has been applied to the original training set and also to sets that have been preprocessed by the methods RNG, k-NCN and GG. These editing approaches has decrease the computer cost of an Nearest Neighbor Rule classifier (Sánchez et al., 1997).

Accordingly, this paper addresses the problem of selecting prototypes in order to decrease the computer cost of an ANN classifier. The Table 2 reports the percentage of size reduction yielded by the different editing.

With the elimination of atypical/noise test patterns, and the patterns overlapped, we reduce the computational cost

Table 1. Data sets.

Data set	*Number classes*	*Number features*	*Samples training*	*Samples test*
Australian	2	42	552	138
Diabetes	2	8	614	154
German	2	24	800	200
Liver	2	6	276	69
Phoneme	2	5	4323	1081
Sonar	2	60	166	42
Balance	3	4	500	125
Cayo	11	4	4815	1204
Ecoli6	6	7	265	67
Fetwell	5	15	8755	2189
Glass	6	9	171	43
Satismage	6	36	5148	1287

and the learning time in a NN. From the Table 2 we can note that the two-classes TS size was reduced approximately in a 24% and in the multi-class until a 20% in average. Also we can observe that the k-NCN technique was the one where more elements eliminate on two classes data bases, whereas in the Multi-class data sets was better the GG method.

Table 2. Training samples eliminated.

Data set	RNG	k-NCN	GG
Australian	24.49	34.31	27.43
Diabetes	22.69	32.13	23.18
German	23.47	33.72	26.42
Liver	19.86	39.84	31.09
Phoneme	7.83	9.84	12.18
Sonar	17.79	18.51	34.02
Balance	6.43	8.67	8.22
Cayo	14.23	7.77	8.52
Ecoli6	1.46	1.30	7.37
Fetwell	25.35	28.50	39.60
Glass	7.22	9.49	18.12
Satismage	61.27	60.81	71.90

Table 3. MP classification results.

Data set	Original TS	RNG	k-NCN	GG
Australian	82.46	82.75	84.05	83.04
Diabetes	75.00	75.65	75.65	75.39
German	70.30	73.90	72.50	71.20
Liver	70.13	64.05	68.11	73.04
Phoneme	80.95	81.23	80.92	81.73
Sonar	86.51	86.52	85.57	76.91
Balance	90.72	89.12	89.60	90.24
Cayo	86.67	86.74	86.49	86.42
Ecoli6	87.04	88.79	87.37	87.67
Fetwell	95.27	95.64	95.53	94.84
Glass	68.26	64.50	67.82	62.63
Satismage	89.55	89.85	89.38	87.19

Table 4. RBF clasification results.

Data set	Original	RNG	k-NCN	GG
Australian	82.17	83.33	83.47	81.01
Diabetes	72.91	74.35	72.78	75.26
German	74.50	72.60	72.30	71.70
Liver	65.21	69.79	61.44	65.21
Phoneme	78.57	77.53	77.97	78.18
Sonar	74.51	74.97	76.42	77.88
Balance	85.92	81.60	86.56	84.48
Cayo	87.22	87.88	87.08	87.97
Ecoli6	85.53	89.07	86.43	86.76
Fetwell	90.33	91.63	90.19	90.46
Glass	67.30	66.38	68.25	66.36
Satismage	84.21	86.01	85.96	85.57

Table 5. SVM clasification results.

Data set	Original	RNG	k-NCN	GG
Australian	84.63	84.34	84.92	84.63
Diabetes	76.95	76.82	74.95	76.49
German	75.40	73.60	74.00	71.00
Liver	58.55	57.97	58.26	57.97
Phoneme	77.36	77.47	77.15	77.59
Sonar	77.27	81.77	79.31	78.37
Balance	88.00	87.84	87.68	86.56
Cayo	66.98	66.58	65.72	68.96
Ecoli6	84.93	87.02	84.64	86.25
Fetwell	91.02	91.20	91.25	90.99
Glass	59.76	55.11	49.56	55.59
Satismage	86.71	86.68	87.77	84.97

5.1 Classification results

In the experimental results showing here, the classification index for the NN trained on TS without preprocessing has also been included as the reference values. On the other hand, the preprocessing methods were firstly applied on the data sets and after that, each NN was trained with these preprocessed data sets.

To evaluate the performance of learning NN, we use the standard evaluation measure in pattern recognition, the Overall Accuracy is:

$$Ac = 1 - \frac{Ne}{Nx} \tag{4}$$

where *Ne* is the mistakes number and *Nx* is the number of training samples.

Table 3, Table 4 and Table 5, show the results obtained with several preprocessing methods (RNG, k-NCN and GG), using MP, RBF and SVM. The results for each original training set (i.e., without preprocessing) has also been included as a baseline. The values marked with bold typeface indicate the best results.

From these results, some initial comments can be drawn. Firstly, for the majority data sets there exist al least one preprocessing method whose classification accuracy is higher than that obtained when using the original TS (without preprocessing). Also, we can observe that, after applying the preprocessing algorithms with methods described previously, the RNG method was the best one maintaining the overall accuracy as much with the MLP as well with the RBN models.

On the other hand, when comparing the overall predictive accuracies of the NN models, we found that the MLP generally has a favorable behavior on the 85% of the data set used and, in opposite way, the accuracy obtained with the SVM can differ from one problem to another (depending of each particular data set), outperforming de original TS only in 67% of the cases. Nevertheless, when the classifier behavior decrease, it is not significant, for example: Liver with MLP (using RNG), Glass with the SVM or k-NCN (using RGN and GG respectively). On the other hand, is possible to observe that with the edition process, the SVM behavior is affected more than the MLP and RBF models, especially when GG strategy is applied.

The Glass data set constitutes a very particular case, in which NN behavior is strongly affected. In all cases, the results obtained without preprocessing always is higher than the overall accuracies obtained with any NN model. A similar

situation is observed with German data set when a RBF and SVM are used. This situation could be produced for another complexity data inside of the data set, such as, imbalance problem or if some data cover up other data.

6. Concluding Remarks

In the NN paradigm, the high computer cost associated to the learning process is a serious problem. This cost is related directly with the training data set size. In this work, we proposed to reduce the computational cost, by reducing the training data size when a NN is used. Specifically, we use the MLP, RBF and SVM models. For that, three strategies well known in the nearest neighbor rule context for reducing the training set improving the classifier behavior were used.

The experimental results shown that, in general, the methods used reduce the data bases size at least 20%. This reduction could be translated in an important computational cost diminution associated to the learning process of the network. On the other hand, was possible to observe that in the majority of the data bases the classifier behavior was stayed or increase, few cases shown lost in the classifier effectiveness.

Finally, the strategies proposed in this work can be useful when the computational cost associated to the NN learning is expected (without losing effectiveness in the classification). Nevertheless, the main impact of this proposal is not only the reduction of the computational cost, but the incorporation of a criterion based on the space neighborhood of the data, for identify those samples which give few information to the learning process of the NN.

Future work is primarily addressed to deepening in this study, not only by the importance of the subject, but by its relation with other areas such as the medicine, astronomy or the economy. At the moment, these disciplines are helped by the pattern recognition techniques, especially by the artificial neuronal networks.

Specifically, we have contemplated to deepen in the analysis of data complexity subject (dimensionality, overlapping, representativeness, and probabilistic density), due to the observed results; it suggests that some of these aspects are responsible of the low behavior of the classifier. As second line of investigation, is the idea to generate methods based on the Wilson editing (Wilson, 1972), can work in the hidden space of the neural network. This idea could generate great expectations. In order to obtain this one would be to use a dissimilarity measurement in the transformation space of the training sample and not in the entrance space, such as commonly happens with the Wilson editing and its variants.

ergo

References

Barandela, R. & E. Gasca (2000). "Decontamination of training samples for supervised pattern recognition methods", *Lecture Notes in Computer Science*. Springer. 1876.

Cristianini, N. & J. Shawe-Taylor (2000). *An Introduction to Support Vector Machines, Cambridge University Press*, Cambridge, UK.

Dasarathy, B. V. (1994). "Minimal consistent set (MCS) identification for optimal nearest neighbor decision system design", *Transactions on Systems Man and Cybernetics*. 24(3).

Gopalakrishnan, M.; V. Sridhar & H. KrishNamurthy (1995). *Some application of clustering in the desing of neural networks*. Patter Recognition Letters, 16.

Haykin, S. (1999). *Neural Networks, A Comprehensive Foundation*. 2nd ed. Pretince Hall, New Jersey.

Guha, S; R. Rastogi & K. Shim (1998). *CURE: An efficient clustering algorithm for large databases*. ACM SIGMOD International Conference on Management of Data, Seattle, Washington.

Jain, A.; J. Mao & K. Mohiuddin (1996). "Artificial neural networks: A tutorial". *Computer* 29(3).

John, G. H. (1997). *Enhancements to the Data Mining Process*. Ph. D. Thesis, Stanford University.

Lippmann, R. (1988). "An introduction to computing with neural nets", in: *Artificial neural networks: theoretical concepts*. IEEE Computer Society Press, Los Alamitos, CA, USA.

Sánchez, E. N. & Alanís A. Y. (2006). *Redes Neuronales. Conceptos Fundamentales y aplicaciones a control automático*. Pearson-PrenticeHall.

Sánchez J. S.; F. Pla & F. J. Ferri (1997). "Using the nearest centroid neighbourhood concept for editing purpose", in *Proceedings of VII Simposium Nacional de Reconocimiento de formas y Análisis de Imágenes 1*.

Sherstinsky A.; Rosalind P. (1996). "On The Efficiency Of The Orthogonal Least Squares Training Method For Radial Function Networks", *IEEE Transactions on Neural Networks*. 7(1).

Vaptnik V.N. (1995). *The nature of Statical Learning Theory*. Wiley. New York.

Wilson D. L. (1972). *Asymptotic properties of nearest neighbor rules using edited data sets*. IEEE Transaction on Systems, Man and Cybernetics. 2.

Witten, I. and Frank, E. (2005). *Data Mining: Practical Machine Learning Tools and Techniques*, Second Edition (Morgan Kaufmann Series in Data Management Systems). Morgan Kaufmann Publishers Inc., San Francisco, CA, USA

5.3 Edición basada en la regla del vecino más cercano para mejorar la precisión de una red neuronal artificial

Parte de la investigación se publico en el articulo "Edited Nearest Neighbor rule for improving neural networks classifications" en el **7th International Simposium on Neural Networks (ISNN 2010)**, celebrado en Shanghai, China en Junio de 2010.

En esta publicación las técnicas de edición se proponen como una alternativa para mejorar la precisión de una red neuronal artificial. En este articulo se observan los beneficios de eliminar el ruido y patrones atípicos mediante las técnicas de edición. A continuación se incluye el articulo completo.

Liqing Zhang
Bao-Liang Lu
James Kwok (Eds.)

Advances in Neural Networks – ISNN 2010

LNCS 6063

7th International Symposium
on Neural Networks, ISNN 2010
Shanghai, China, June 2010, Proceedings, Part I

1 Part I

Springer

Volume Editors

Liqing Zhang
Bao-Liang Lu
Department of Computer Science and Engineering
Shanghai Jiao Tong University
800, Dongchuan Road
Shanghai 200240, China
E-mail: {zhang-lq; blu}@cs.sjtu.edu.cn

James Kwok
Department of Computer Science and Engineering
The Hong Kong University of Science and Technology
Clear Water Bay, Kowloon, Hong Kong, China
E-mail: jamesk@cse.ust.hk

Library of Congress Control Number: 2010927009

CR Subject Classification (1998): I.4, F.1, I.2, I.5, H.3, J.3

LNCS Sublibrary: SL 1 – Theoretical Computer Science and General Issues

ISSN 0302-9743
ISBN-10 3-642-13277-4 Springer Berlin Heidelberg New York
ISBN-13 978-3-642-13277-3 Springer Berlin Heidelberg New York

Typesetting: Camera-ready by author, data conversion by Scientific Publishing Services, Chennai, India
Printed on acid-free paper 06/3180

Learning and Inference

Edited Nearest Neighbor Rule for Improving Neural Networks Classifications

R. Alejo[1], J.M. Sotoca[1], R.M. Valdovinos[2], and P. Toribio[3]

[1] Institute of New Imaging Technologies
Dept. Llenguatges i Sistemes Informátics, Universitat Jaume I
Av. Sos Baynat s/n, 12071 Castelló de la Plana (Spain)
[2] Centro Universitario UAEM Valle de Chalco, Universidad Autónoma del Estado de México
Hermenegildo Galena No.3, Col. Ma. Isabel, 56615 Valle de Chalco (Mexico)
[3] Centro Universitario UAEM Atlacomulco, Universidad Autónoma del Estado de México
Km. 60 Carretera Toluca - Atlacomulco, 50450 Atlacomulco (Mexico)

Abstract. The quality and size of the training data sets is a critical stage on the ability of the artificial neural networks to generalize the characteristics of the training examples. Several approaches are focused to form training data sets by identification of border examples or core examples with the aim to improve the accuracy of network classification and generalization. However, a refinement of data sets by the elimination of outliers examples may increase the accuracy too. In this paper, we analyze the use of different editing schemes based on nearest neighbor rule on the most popular neural networks architectures.

Keywords: neural networks; editing techniques; reduction training set; accuracy.

1 Introduction

Artificial neural networks (ANN) are computational models that have become a popular tool used in remote sensing data analysis, the computer-aided medical diagnosis and the identification of microbiological taxa. However, it is well know that in supervised classification the maximum accuracy depends on the quality of the training data set. In this sense, several approaches in the area of neural networks are towards to training data selection with the aim to improve the performance of the network in terms of speed, computational requirements and classification accuracy.

Most research on this topic has traditionally focused to obtain reduced training data sets (TS) by the identification of two kinds of samples: *i)* core training patterns and *ii)* border training patterns [1, 2]. However, the outliers or noise may introduce a false decision boundary. Consequently, the training data extracted may also be subject to refinement intelligent procedures.

An outlier has traditionally been defined as a prototype that does not follow the same model as the rest of the data [3]. In this context, some approaches to allow remove outliers from the original training set and also cleaning possibles overlapping among classes. This strategy has generality been referred as to editing [4].

The general idea behind almost any editing procedure consists of estimating the true classification of prototypes in the TS to retain only those which are correctly labeled.

L. Zhang, J. Kwok, and B.-L. Lu (Eds.): ISNN 2010, Part I, LNCS 6063, pp. 303–310, 2010.

The first proposal to select a representative subset of prototypes for a further nearest neighbour classification corresponds to Wilson editing algorithm [5], in which a k-NN classifier is used to retain in the TS only good samples (that is, training samples that are correctly classified by the k-NN rule).

In the present work, we study the use of several editing schemes proposed in the literature and their effects on the classification performance of three supervised artificial neural networks. This study mainly tries to show how these methods used in the nearest neighbour rule improve the classification accuracy and generalization of the most popular neural networks architectures. In order to accomplish this, we experiment with two class data sets and multiclass data sets.

The structure of the paper is as follows. Section 2 presents the learning algorithms. In Section 3, we briefly describe the editing algorithms used to reduce the training data sets. Section 4 consists of experiments on real data sets and an exhaustive discussion of results. Finally, we will conclude the main remarks and outline some directions for future work in Section 5.

2 The Classifiers

In this section, we briefly describe the classifiers selected for the present experimental study. All these algorithms work under the assumption that there exists a set of n previously labeled examples (training set, TS), say $\mathbf{X} = \{(\mathbf{x}_1, \omega_1), (\mathbf{x}_2, \omega_2), \ldots, (\mathbf{x}_n, \omega_n)\}$, where each element has an attribute vector $\mathbf{x}_i$ and a class label ω_i.

2.1 Multilayer Perceptron

The multilayer perceptron (MLP) neural network [6] usually comprises one input layer, one or more hidden layers, and one output layer. Input nodes correspond to features, hidden layers are used for computations, and output layers are the problem classes. A neuron is the elemental unit of each layer. It computes the weighted sum of its inputs, adds a bias term and drives the result thought a generally nonlinear (commonly, sigmoid) activation function to produce a single output.

The most popular training algorithm for MLP is the backpropagation, which takes a set of training instances for the learning process. For the given feedforward network, the weights are initialized to small random numbers. Each training instance is passed through the network and the output from each unit is computed. The target output is compared with the output estimated by the network to calculate the error, which is fed back through the network. To adjust the weights, backpropagation uses gradient descent to minimize the squared error between the target output and the computed output. At each unit in the network, starting from the output unit and moving down to the hidden units, its error value is used to adjust weights of its connections so as to reduce the error. This process of adjusting the weights is repeated for a fixed number of times or until the error is small or it cannot be reduced.

2.2 Radial Basis Function

The radial basis function (RBF) [7] neural network, which has three layers, can be seen as an especial kind of multilayer feedforward networks. Each unit in the hidden layer

employs a radial basis function, such as Gaussian kernel, as the activation function. The output units implement a weighted sum of hidden unit outputs. The input into an RBF network is nonlinear. The output is linear. The kernel is centered at the point specified by the weight vector associated with the unit. Both the positions and the widths of these kernels are learned from training instances. Each output unit implements a linear combination of these radial basis functions.

An RBF is trained to learn the centers and widths of the Gaussian function for hidden units, and then to adjust weights in the regression model that is used at the output unit. To learn the centers of the Gaussian functions, the k-means algorithm can be used, obtaining k Gaussian functions for each attribute in the instance. After the parameters for the Gaussian function at the hidden units have been found, the weights from these units to the output unit are adjusted using a linear regression model.

2.3 Support Vector Machine

Support vector machines (SVMs) [8] are a set of related supervised learning methods used for classification and regression. They belong to a family of generalized linear classifiers. A special property of SVMs is that they simultaneously minimize the empirical classification error and maximize the geometric margin; hence they are also known as maximum margin classifiers.

SVMs map input vectors to a higher dimensional space where a maximal separating hyperplane is constructed. Two parallel hyperplanes are constructed on each side of the hyperplane that separates the data. The separating hyperplane is the hyperplane that maximizes the distance between the two parallel hyperplanes. An assumption is made that the larger the margin or distance between these parallel hyperplanes the better the generalization error of the classifier will be.

3 Algorithms to Select Training Samples

The present section describes the procedures for handling outliers in a TS. This alternatives are based on the employment of a surrounding neighborhood to obtain a filtered TS, that is, to detect and remove outliers from the TS.

3.1 Edited Nearest Neighbor Rule

Wilson [5] developed the Edited Nearest Neighbor (ENN) algorithm in which **S** starts out the same as TS, and then each instance in S is removed if it does not agree with the majority of its k nearest neighbors (with k=3, typically). This edits out noisy instances as well as close border cases, leaving smoother decision boundaries. It also retains all internal points, which keeps it from reducing the storage requirements as much as most other reduction algorithms. Algorithmically, the ENN scheme can be expressed as follows:

1. Let $\mathbf{S} = \mathbf{X}$.
2. For each $\mathbf{x}_i$ in $\mathbf{X}$ do:
   - Discard $\mathbf{x}_i$ from $\mathbf{S}$ if it is misclassified using the k-NN rule with prototypes in $\mathbf{X} - \{\mathbf{x}_i\}$.

3.2 Editing with the Nearest Centroid Neighborhood

The nearest centroid neighborhood (NCN) [9] refers to a concept in which neighborhood is defined taking into account not only the proximity of prototypes to a given input sample but also their symmetrical distribution around it. From this general idea, the corresponding classification rule, the k-nearest centroid neighbours (k-NCN) [10], has been proven to overcome the traditional k-NN classifier in many practical situations. Now the NCN editing (NCNE) approach presented here corresponds to a slight modification of the original work of Wilson and basically consists of using the leaving-one-out error estimate with the k-NCN classification rule.

3.3 Proximity Graph Based Editing

Proximity graph based editing scheme are based on the concepts of Gabriel Graph (GG) and Relative Neighborhood Graph (RNG) [11]. The method consists of applying the general idea of Wilson's editing algorithm [5] but using the graph (GG or RNG) neighbors of each sample, instead of the Euclidean distance-based neighborhood, in order to estimate whether a sample is mislabeled or not. In a few words, the simplest GG and RNG prototypes based editing can be summarized as follows: after computing the graph neighborhood of every sample in the original training set, discard those samples that are misclassified by their graph neighbors (instead of their k nearest neighbors).

This editing technique provides some advantages as compared to conventional methods. Firstly, it considers the neighborhood size as a characteristic which depends on each one of the prototypes in the training set. Secondly, GGE and RNGE provides some kind of information about prototypes close enough but homogeneously distributed around a given sample, which can be specially interesting to detect outliers close to the inter-class or decision boundaries. A more detailed description of GGE and RNGE can be found in [12].

4 Experimental Results

The experiments were carried out on ten real data sets taken from the UCI Machine Learning Database Repository (`http://archive.ics.uci.edu/ml/`). A brief summary is given in the Table 1. For each database, we have estimated the overall accuracy by 5–fold cross–validation: each data set was divided into five equal parts, using four folds as the training set and the remaining block as independent test set. The Overall Accuracy were calculated from the equation:

$$Acc = 1 - \frac{n_e}{n_t} \tag{1}$$

where n_e is the number of misclassified examples and n_t is the total number of testing examples.

The experiments have been performed using the Weka Toolkit [13] with the learning algorithms described in Section 2, that is, MLP, SVM and RBF. Each classifier has been applied to the original training set and also to sets that have been preprocessed by the methods NNE, RNGE, NCNE and GGE. These editing approaches has been

Table 1. Data sets used in the experiments

Data set	Classes	Features	Samples training	Samples test
Diabetes	2	8	614	154
German	2	24	800	200
Heart	2	25	217	55
Liver	2	6	276	69
Phoneme	2	5	4323	1081
Sonar	2	60	166	42
Cayo	11	4	4815	1204
Ecoli6[1]	6	7	265	67
Fetwell	5	15	8755	2189
Satimage	6	36	5148	1287

[1] Ecoli6 is obtained from Ecoli. In this work classes 7 and 8 have been eliminated since these only have two samples.

Table 2. Size reduction rate using different editing techniques

Data set	NNE	RNGE	NCNE	GGE
Diabetes	29.75	22.60	32.03	23.08
German	31.71	23.47	33.70	26.46
Heart	35.48	29.03	41.47	31.33
Liver	35.74	29.96	38.98	31.04
Phoneme	11.40	7.83	9.85	12.18
Sonar	18.56	17.96	18.56	34.13
Average	27.11	21.81	29.1	26.37
Cayo	8.20	6.41	9.01	8.20
Ecoli6	14.66	14.28	17.66	18.42
Fetwell	1.77	1.49	1.31	7.38
Satimage	9.53	7.20	9.53	17.98
Average	8.54	7.35	9.38	12.99

improved the classification accuracy of an Nearest Neighbor Rule classifier [14]. Accordingly, this paper addresses the problem of selecting prototypes in order to improve the classification accuracy of an ANN classifier.

The Table 2 reports the percentage of size reduction yielded by the different editing algorithms over ten databases. The results show that in the case of the two-classes data sets, the average reduction rate was approximately 26.1% , meanwhile in the situation of multi-class data sets was 8% to 17% (approximately). From this, the NCNE and the GGE achieve the highest rates in the two-class and multiclass data sets, respectively. Consequently, this implies a important decrease in computational cost in the learning phase. This issue is important in applications with high-size data sets.

On other hand, in Feltwell the number of discards was minimal (except for the GGE method). This is due to the measuring criteria used by NNE, NCNE, and RNGE suggest

Table 3. Experimental results (Acc) for three different artificial neural network architectures

Neural Network	Data set	Original	NNE	RNGE	NCNE	GGE
MLP	Diabetes	75.01(4.88)	74.74(3.56)	75.66(4.31)	75.65(3.05)	75.39(1.69)
	German	70.30(2.20)	72.50(3.81)	73.90(2.38)	72.50(2.57)	71.20(1.48)
	Heart	82.59(4.83)	81.11(7.57)	78.52(4.83)	78.52(4.65)	80.74(4.46)
	Liver	70.14(4.65)	69.28(5.46)	64.06(4.74)	68.12(8.51)	73.04(3.01)
	Phoneme	80.96(1.64)	81.48(1.19)	81.24(1.23)	80.92(1.34)	81.74(0.89)
	Sonar	86.52(5.03)	82.69(1.99)	86.53(2.78)	85.57(5.64)	76.91(2.86)
	Average	77.59(6.8)	76.97(5.55)	76.65(7.61)	76.88(6.20)	76.50(4.17)
	Cayo	86.59(0.29)	86.29(0.34)	86.18(0.33)	86.48(0.59)	86.48(0.71)
	Ecoli6	86.73(2.97)	87.65(3.27)	85.24(2.47)	87.35(1.67)	86.75(1.23)
	Fetwell	95.11(0.77)	95.53(0.48)	94.80(0.37)	95.88(0.44)	94.90(0.27)
	Satimage	88.78(0.67)	89.62(0.49)	89.49(0.66)	89.68(0.58)	87.29(1.56)
	Average	89.30(4.00)	89.77(4.07)	88.93(4.32)	89.85(4.24)	88.86(4.04)
Neural Network	Data set	Original	NNE	RNGE	NCNE	GGE
RBF	Diabetes	72.91(4.93)	75.13(4.76)	74.35(6.67)	72.78(5.55)	75.26(4.65)
	German	74.50(4.11)	70.80(2.46)	72.60(3.27)	72.30(3.83)	71.70(1.96)
	Heart	80.00(6.06)	81.11(5.14)	80.37(4.26)	82.59(4.65)	79.63(3.46)
	Liver	65.22(2.29)	61.45(3.64)	65.80(4.42)	61.45(2.20)	65.22(3.55)
	Phoneme	78.57(1.29)	77.81(1.20)	77.54(0.78)	77.98(1.71)	78.18(0.66)
	Sonar	74.52(8.18)	72.57(6.61)	74.97(5.68)	76.42(4.06)	77.89(6.25)
	Average	74.29(5.20)	73.15(6.81)	74.27(4.96)	73.92(7.17)	74.65(5.40)
	Cayo	87.26(0.58)	87.81(0.38)	88.14(0.54)	87.86(0.55)	87.71(0.47)
	Ecoli6	85.83(2.37)	84.94(2.09)	86.76(2.82)	87.35(2.75)	84.64(1.23)
	Fetwell	90.63(0.61)	89.99(1.72)	89.99(0.95)	89.99(1.25)	91.08(0.76)
	Satimage	84.01(0.48)	85.86(0.83)	86.05(0.99)	85.84(0.86)	85.70(1.12)
	Average	86.93(2.80)	87.15(2.24)	87.74(1.74)	87.76(1.72)	87.28(2.83)
Neural Network	Data set	Original	NNE	RNGE	NCNE	GGE
SVM	Diabetes	76.95(1.97)	75.65(4.15)	76.82(2.53)	76.56(3.01)	76.56(3.17)
	German	75.40(3.31)	72.30(1.89)	73.60(2.13)	74.00(1.90)	71.00(0.71)
	Heart	84.07(4.46)	81.48(6.93)	81.85(4.22)	81.11(5.14)	82.96(3.56)
	Liver	58.55(1.30)	57.97(0.00)	57.97(0.00)	58.26(0.65)	57.97(0.00)
	Phoneme	77.37(1.08)	77.00(1.14)	77.48(1.13)	77.15(1.23)	77.59(1.04)
	Sonar	79.28(7.07)	82.21(4.01)	81.72(2.81)	79.31(4.06)	78.37(5.57)
	Average	75.27(8.72)	74.42(8.90)	74.90(8.90)	74.40(8.27)	74.06(8.80)
	Cayo	66.94(0.48)	66.37(0.13)	66.62(0.48)	65.36(0.39)	69.13(1.00)
	Ecoli6	83.43(1.51)	83.13(1.37)	85.55(0.75)	84.32(3.03)	84.94(1.08)
	Fetwell	91.01(0.14)	91.25(0.23)	91.22(0.26)	91.15(0.28)	91.07(0.33)
	Satimage	86.53(0.99)	86.54(0.84)	86.40(0.86)	86.65(0.85)	85.45(0.99)
	Average	81.98(10.50)	81.82(10.83)	82.45(10.84)	81.87(11.37)	82.65(9.43)

that this data set is not overlapped, i.e., the decision boundary is well defined. However, it is doubtful whether this is really so or if necessary use other measures to locate the prototypes in the overlap region.

Examining the Table 3, the results shown that the editing algorithms obtain similar classification accuracy (Acc) to that of original data sets (i.e., without performing editing). This indicate that, the shrink of training set give a lower computational loads, but also not involve a loss performance. In this sense, although in the case of NCNE no outperform the original data sets (for three classifiers with two-class data set), the differences in such cases are not statistically significant. A final remark from the experiments and perhaps important, refers to that in certain cases the classification accuracy is improved.

In summary, the observations reported in Table 3 suggest that, on multiclass problems the performance editing techniques is better than that of two class problem, and illustrate that these techniques have a tendency to improve the classification accuracy.

5 Conclusion and Further Extensions

When using an ANN, the presence of mislabelled prototypes can degrade the corresponding classification accuracy. Many models for identifying and removing outliers have been proposed. This paper has reviewed some works in the frame of editing the nearest neighbor rule. A number of experiments over ten real data sets have been carried out in order to evaluate the accuracy of those editing methods. The experiments illustrate that editing techniques have a tendency improve the classification accuracy.

In the other hand, the editing techniques here studied, we can diminish the size of the data bases, and with this the diminution of the computational cost and the learning time in the ANN (RBFNN, MLP and SVM). Especially, in two class problem (was reported more of the 21% of reduction, approximately).

Future work is primarily addressed to investigate the potential of these editing methods applied to the hidden space of the neural network. This idea could generate great expectations. In order to obtain this one would be to use a dissimilarity measurement in the transformation space of the training sample and not in the feature space, such as commonly happens with the Wilson editing and its variants.

Acknowledgment

This work has been partially supported by the Spanish Ministry of Science and Education under project CSD2007-00018, UAEMCA-114 and SBI112 from the Mexican SEP, FE28/2009 (103.5/09/4195) of the Mexican PROMEP and the 2703/2008U from the UAEM project.

References

1. Foody, G.: The significance of border training patterns in classification by a feedforward neural network using back propagation learning. JRS 20(18), 3549 (1999)
2. Kavzoglu, T.: Increasing the accuracy of neural network classification using refined training data. Environmental Modelling & Software 24(7), 850–858 (2009)

3. Sánchez, J., Barandela, R., Marqués, A., Alejo, R., Badenas, J.: Analysis of new techniques to obtain quality training sets. Pattern Recognition Letters 24(7), 1015–1022 (2003)
4. Wilson, D.R., Martinez, T.R.: Reduction techniques for instance-based learning algorithms. Machine Learning 38(3), 257–286 (2000)
5. Wilson, D.: Asymptotic properties of nearest neighbor rules using edited data. IEEE Transactions on Systems, Man and Cybernetics 2(4), 408–420 (1972)
6. Bishop, C.M.: Neural Networks for Pattern Recognition, 1st edn., January 1996. Oxford University Press, USA (1996)
7. Buhmann, M.D.: Radial basis functions: theory and implementations. Cambridge University Press, United Kingdom (2003)
8. Vapnik, V.: Estimation of Dependences Based on Empirical Data. Springer Series in Statistics (Springer Series in Statistics). Springer, New York (1982)
9. Chaudhuri, B.B.: A new definition of neighborhood of a point in multi-dimensional space. Pattern Recogn. Lett. 17(1), 11–17 (1996)
10. Sánchez, J.S., Pla, F., Ferri, F.J.: On the use of neighbourhood-based non-parametric classifiers. Pattern Recogn. Lett. 18(11-13), 1179–1186 (1997)
11. Jaromczyk, J., Toussaint, G.: Relative neighborhood graphs and their relatives. Proceedings of the IEEE 80(9), 1502–1517 (1992)
12. Sánchez, J.S., Pla, F., Ferri, F.J.: Prototype selection for the nearest neighbour rule through proximity graphs. Pattern Recogn. Lett. 18(6), 507–513 (1997)
13. Witten, I., Frank, E.: Data Mining: Practical Machine Learning Tools and Techniques, 2nd edn. Morgan Kaufmann Series in Data Management Systems. Morgan Kaufmann Publishers Inc., San Francisco (2005)
14. Sánchez, J., Barandela, R., Marqués, A., Alejo, R.: Performance evaluation of prototype selection algorithms for nearest neighbor classification. In: SIBGRAPI 2001: Proceedings of the XIV Brazilian Symposium on Computer Graphics and Image Processing (SIBGRAPI 2001), Washington, DC, USA, pp. 44–50. IEEE Computer Society, Los Alamitos (2001)

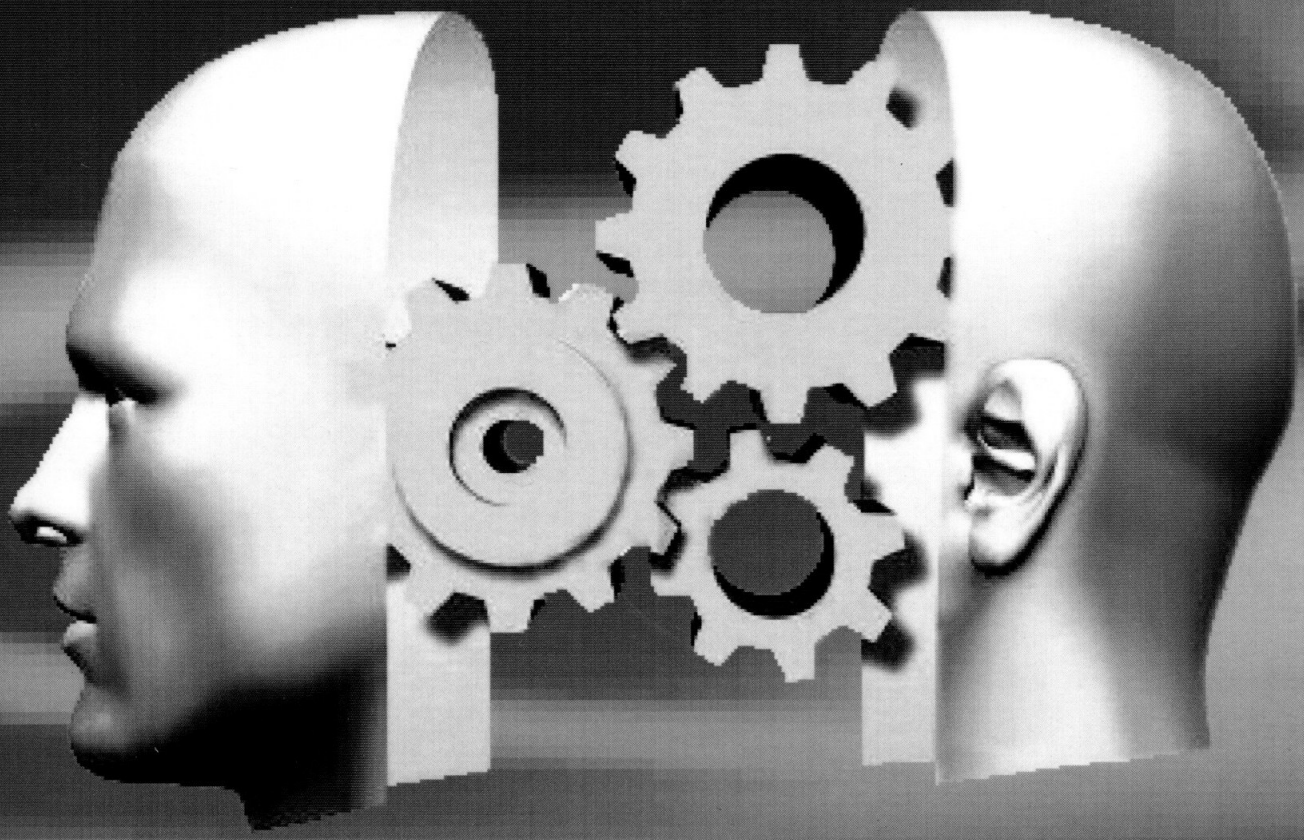
5.4 Reducción de la complejidad de los datos y su impacto en la precisión de clasificación de una RNA

El cuarto artículo fue presentado como ponencia y publicado el libro del **Congreso Mexicano de Inteligencia Artificial 2010**(COMIA2010) celebrado en las instalaciones de la Universidad Autónoma de Tlaxcala [4]. El titulo se llama "Reducción de la complejidad de los datos y su impacto en la precisión de clasificación de una RNA".

En este se resumen una serie de experimentos sobre la reducción de la complejidad de los datos (específicamente ruido, patrones atípicos y solapamiento entre clases) y su impacto sobre la precisión del clasificador. Se retoman algunas métricas para medir la complejidad de los datos y se hace una comparación de las técnicas de edición y una eliminación aleatoria para demostrar la eficacia de estas. A continuación se incluye el articulo completo.

[4]ISBN:978-607-95367-2-5; más información en `http://ingenieria.uatx.mx/comia/`

AVANCES EN SISTEMAS INTELIGENTES EN MEXICO



EDITORES
MIGUEL GONZÁLEZ MENDOZA
OSCAR HERRERA ALCANTARA

ISBN: 978-607-95367-2-5

==========

Nº ID Tributaria: SMI871231

PRIMERA EDICIÓN: 2010

MIGUEL GONZALEZ MENDOZA
OSCAR HERRERA ALCANTARA
México, D. F.
DISEÑO DE PORTADA: Kronos Digital
PROCESOS EDITORIALES: Edgar Fernández C.

Impresiones / Printing: 500
Impreso en México / *Printed in Mexico*

Reducción de la Complejidad de los Datos y su Impacto en la Precisión de Clasificación de una RNA

P. Toribio*, R.M. Valdovinos**, R. Alejo ***

Universidad Autónoma del Estado de México,
www.uaemex.mx

Resumen. Las redes neuronales artificiales se han popularizado en la actualidad para realizar diversas tareas en el área de aprendizaje automático y reconocimiento de patrónes, sin embargo, ha observado en la literatura que la efectividad en su aprendizaje depende de la complejidad de los datos en la muestra de entrenamiento. En este trabajo se retoman algunas técnicas aplicadas al clasificador del vecino más cercano para reducir la complejidad de los datos y algunas métricas de complejidad para comprobar la efectividad de estas técnicas. Las métricas comparten la característica de ser independientes del clasificador. Por ultimo, se analizará el impacto sobre la precisión del clasificador al utilizar este tipo de técnicas.

Palabras Clave: Técnicas de edición, RNA, Complejidad de los Datos, Precisión Global, Precisión por Clase.

1 Introducción

Una Red Neuronal Artificial (RNA) es un modelo matemático que trata de emular a los sistemas neuronales biológicos en el procesamiento de la información [1]. Actualmente gozan de gran popularidad entre teóricos y especialistas en el área de aprendizaje automático, minería de datos y reconocimiento de formas. Una de las principales razones del auge de las RNA es que están dotadas de algoritmos que les permiten aprender a partir de ejemplos o datos de entrada [2], por lo que no precisan para su funcionamiento de un modelo a priori del problema a tratar.

Al día de hoy existen numerosos modelos de RNA entre los que se encuentran las RNA de Funciones de Base Radial (Radial Basis Function, RBF) y el Perceptron Multicapa (Multilayer Perceptron, MLP). Ambos modelos son considerados de propagación hacia adelante (feedforward), tienen capas no lineales [3] o pueden ser entrenados con métodos similares de descenso por gradiente. Sin embargo, presentan notables diferencias,por ejemplo, la función de activación de los nodos ocultos en las redes RBF [4] depende de la distancia entre los vectores de entrada y los centros de la red, mientras que en el MLP depende directamente del producto del vector de entrada y el vector de pesos.

* CU UAEM Texcoco, guffyprimo@hotmail.com
** CU UAEM Chalco, li_rmvr@hotmail.com
*** Tecnologico de Estudios Superiores de Jocotitlan, ralejoll@hotmail.com

El MLP y las redes RBF han sido ampliamente utilizadas en tareas de clasificación, aproximación de funciones, modelado y problemas de control [5]. No obstante, aún se conoce poco de estos modelos lo que se traduce en debilidades tales como la lentitud en el aprendizaje y la pobre capacidad de generalización que se observa en un número importante de aplicaciones prácticas [6].

El incremento de la rápidez del procedimiento de entrenamiento y la mejora de la precisión en el resultado, han motivado un esfuerzo importante en la investigación de criterios más adecuados para definir parámetros y algoritmos vinculados al proceso de aprendizaje.

En los ultimos años, surgió el interés por examinar la calidad de la muestra de entrenamiento (ME) y la validez de sus elementos, varios investigadores han coincidido en que la precisión de estos clasificadores depende de la calidad de los datos [7]. Por ejemplo, Barandela [6]dice que el conjunto de datos para entrenar la red contiene patrones atípicos y ruidosos que la afectan negativamente, en [7] tratan varios factores como el tamaño de la ME y el solapamiento de clases. Aunque, todavía no hay un acuerdo en que características afectan al clasificador y en que medida, en este trabajo se tratan a estos factores como "complejidad de los datos". También, se implementan algunas métricas que intentan medir desde el punto de vista geométrico algunas características (solapamiento, separabilidad de clases, entre otros) de las bases de datos.

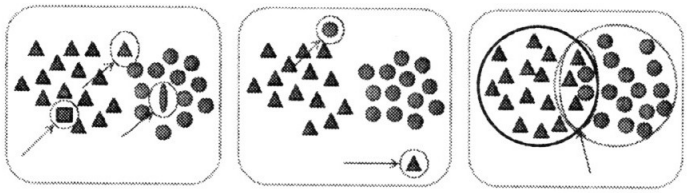
Sobre el enfoque del estudio de la calidad de la ME, en este trabajo se pretenden probar la efectividad de técnicas extraídas el contexto de la regla del vecino más cercano para reducir la complejidad de la ME y ver su impacto sobre la precisión del clasificador. Específicamente se utilizarán la Edición de Wilson y la variante que utiliza como regla de clasificación la del centroide más Cercano (Wilsoncn), los grafos de Gabriel y los de vecindad relativa, en este trabajo las llamaremos técnicas de edición. Además se comparará el resultado sobre el clasificador de estas técnicas con los de una eliminación aleatoria, con el objetivo de verificar la eficacia la técnicas de edición.

2 Complejidad de los Datos

De manera general, se definirá a la complejidad de los datos como todos aquellos factores o características que disminuyen la calidad de la ME y que por consiguiente disminuyen la precisión del clasificador. En este artículo solo se trataran tres características dentro de complejidad de los datos, ya que, dichas técnicas están enfocadas sobre estos, las características son:

- **Ruido.** Son Patrones con errores, originados en su medición o registro (mal etiquetados, es decir, cuando no pertenecen a la clase donde fueron colocados). En [6], se habla del ruido (ver Fig. 1a) como situaciones imperfectamente supervisadas. También en [8] se habla sobre el ruido y propone algunas metricas para medirlo.
- **Patrones Atípicos.** Son excepciones a la regla, es decir, que aunque han sido identificados correctamente estos son muy diferentes al resto de los patrónes de la misma clase (en su tratamiento comúnmente son confundidos como ruido). En [6], se habla sobre el efecto negativo sobre la presencia de patrones atípicos (ver Fig. 1b).
- **Solapamiento de clases.** Es cuando dos o más clases se encuentran interceptadas entre sí compartiendo elementos en común. En [9], [10] y [11], se define que es el

solapamiento (ver Fig. 1c), sus efectos sobre el clasificador y algunas soluciones a este problema (algunas basadas en el clasificador del vecino mas cercano).



(a) Ruido (b) Atípicos (c) Solapamiento

Fig. 1. Características de la ME incluidas dentro de la Complejidad de los Datos y tratadas en esta investigación.

3 Técnicas de Edición

También llamadas técnicas de muestreo dirigido, porque siguen una regla para elegir los patrones a descartar. La idea común de todos los algoritmos de Edición consiste en descartar patrones que se encuentren en la región correspondiente a una clase distinta a la suya, es decir, patrónes cuya probabilidad de pertenencia a su clase se vea superada por la probabilidad de pertenencia a alguna otra clase. Estos patrones descartados son aquellos que causan ruido, son atípicos o están dentro de la zona de solapamiento.

3.1 Edición de Wilson (NNE)

El algoritmo de edición de Wilson [12], se basa en la idea de que si un patrón es erróneamente clasificado usando la regla k-NN es eliminado de la ME, para este fin, se utilizarán todos los patrones del la ME para determinar los k-vecinos más cercanos (excepto el prototipo que se está considerando en cada momento), es decir, el método de estimación del error empleado corresponderá al *leaving-one-out*. El algoritmo de edición de Wilson se expresa de la siguiente manera:

1. Inicialización $S \leftarrow X$
2. Para cada prototipo $x_i \epsilon X$
   (a) Buscar los k vecinos más cercanos de x_i en $X - x_i$
   (b) Si $\delta_{k-NN}(x_i) \neq \theta_i$, entonces $S \leftarrow S - x_i$

3.2 Regla de los k-vecinos de Centroide más Cercanos (NCNE)

Valiéndose de la definición de vecindad en [13] se propuso la siguiente regla de clasificación que se denomino regla de los k-vecinos de Centroides más Cercanos(k-NCNE),

la cual se puede formalizar de la siguiente manera:

$$\delta_{k-NCNE}(x) = \bar{w}_i \Leftrightarrow d(x, P_i) = \min_{i=1,...,M} d_k(x, P_j) \quad (1)$$

Esta expresión significa que la clase asignada a la muestra x corresponderá a la clase más votada entre los k-vecinos de centride más cercano. En la práctica, al igual que ocurría con la regla de decisión k-NN, deberíamos considerar siempre un número impar de vecinos con el fin de evitar posibles empates.

3.3 Grafo de Gabriel (GGE)

La técnica del Grafo de Gabriel [13] dice que para un conjunto V de n puntos (refiriéndose a un vector) $V = \{p_1, p_2, ..., p_n\}$, dos puntos p_i y p_j son vecinos de Gabriel si:

$$dist^2(p_i.p_j) < dist^2(p_i, p_k) + dist^2(p_k, p_j) \quad \forall k \neq i, j \quad (2)$$

Uniendo todos los vecinos de Gabriel en pares mediante una arista se obtiene el grafo de Gabriel. En un sentido geométrico, los dos puntos p_i y p_j son vecinos de Gabriel, si el circulo con diámetro igual a la distancia entre p_i y p_j no contiene ningún otro punto $p_j \in V$.

3.4 Grafo de la Vecindad Relativa (RNGE)

La técnica Grafo de Vecindad Relativa [13], basa su funcionamiento en que si dos puntos x, y son relativamente cercanos sí el área definida por la intersección del circulo con centro x y radio xy, y el circulo con centro en y con el mismo radio no contiene otro punto. Formalmente, el grado de la vecindad relativa de un conjunto de puntos S tiene una arista entre x e y si:

$$dist(x.y) \leq max[dist(x, z), dist(y, z)] \quad \forall z \in S, z \neq x, y \quad (3)$$

4 Redes Neuronales Artificiales.

Una red neuronal artificial (RNA) es un prototipo de procesamiento de información inspirado en la estructura del cerebro humano, cuyo objetivo es reproducir sus mecanismo de aprendizaje. Esta red está conformada por un gran número de neuronas artificiales interconectadas en base a un modelo que asigna una respuesta a cada componente, donde lo que se pretende emular no es la estructura biológica de una neurona, sino su funcionamiento. Las RNA sobre las que se enfocará este trabajo son el Perceptron Multicapa y la Red de Funciones de Base Radial.

4.1 Perceptron Multicapa

La arquitectura de un Perceptron Multicapa (*Multilayer Perceptron*, MLP) consta de una capa de entrada, uno o varias capas ocultas y una capa de salida. Cada capa tiene un determinado número de neuronas y cada una estas conectada a todas las neuronas de

la capa siguiente, cada conexión tiene un peso asociado y su valor será calculado por la red en la fase de entrenamiento [14]. El algoritmo de entrenamiento más utilizado para el MLP es el BackPropagation. Los pesos son inicializados con pequeños números aleatorios, después estos se recalcularan cada vez que pase cada patrón por la red. Para ajustar los pesos se compara la salida real con la salida ideal para estimar el error y mediante el método de descendente por gradiente se minimizara ese error.

4.2 Redes de Funciones de Base Radial

Una Red de Funciones de Base Radial (*Radial Basis Function*, RBF) tiene sólo una capa de entrada, una capa oculta y una capa de salida. Cada neurona de la capa oculta tiene como función de activación una función de base radial (de ahí el nombre de la red). Entre la capa oculta y la de salida cada conexión tiene un peso asociado. Cada neurona de la capa oculta es considerada un centroide y a diferencia del MLP lo que se calcula es una distancia.

En el entrenamiento de una red RBF [15], este consta de una fase no supervisada que trata de minimizar la distancia entre cada centroide y los patrónes, comúnmente se utiliza el algoritmo k-means [16]. La siguiente fase de una red RBF es supervisada, en la cual se deben calcular los pesos asociados a cada conexión, esto se puede hacer mediante un modelo de regresión linear o también se puede utilizar el algoritmo Back-Propagation.

5 Metricas de Complejidad

El conjunto de datos o ME se puede estudiar desde el punto de vista geométrico, tal que, cada patrón es un punto de n-dimensiones en un espacio real $\Re_n$ y cada punto es etiquetado a una clase. Se han investigado y propuesto algunas métricas para analizar el comportamiento y describir el desempeño de clasificador. Una métrica muy común, es la tasa de error del clasificador, sin embargo, la meta de muchos investigadores es encontrar otras métricas menos dependientes de clasificador para poderlas aplicarlas a diversos clasificadores. Además, estas métricas podrían dar indicios del origen de los errores y así mejorar el diseño del clasificador o depurar eficientemente la ME.

5.1 Indice Discriminante de Fisher (F1)

El indice discriminante de Fisher (F1) [17] analiza que tan separadas están dos clases de acuerdo a un patrón dado, en otras palabras, calcula la efectividad de un patrón para separar las clases correspondientes. F1 para dos clases, es definido como:

$$f = \frac{(\mu_1 - \mu_2)^2}{\sigma_1^2 + \sigma_2^2}, \tag{4}$$

donde μ_1, μ_2, σ_1^2 y σ_2^2 son las medias y las varianzas de las dos clases respectivamente. se calcula f para cada patrón y se toma el máximo para la métrica F1. Mientras más bajo sea el valor de F1, más alto es el solapamiento entre clases.

5.2 Volumen de la región de solapamiento (F2)

Esta métrica busca para cada patrón f_i el máximo ($max(f_i, c_j)$) y mínimo ($min(f_i, c_j)$) valor para cada clase c_j, después calcula el tamaño de la región de solapamiento normalizado por el tamaño de la región total donde están distribuidos todos los valores de ambas clases. F2 [17] se define como:

$$F2 = prod_i^d \frac{MINMAX_i - MAXMIN_i}{MAXMAX_i - MINMIN_i}, \tag{5}$$

donde $i = 1, ..., d$ para un problema de *n*-dimensiones y

$MINMAX_i = MIN(max(f_i, c_1), max(f_i, c_2))$
$MAXMIN_i = MAX(min(f_i, c_1), min(f_i, c_2))$
$MAXMAX_i = MAX(max(f_i, c_1), max(f_i, c_2))$
$MINMIN_i = MIN(min(f_i, c_1), min(f_i, c_2))$
Mientras más bajo sea el solapamiento de clases, más bajo es el valor de F2.

5.3 Máxima eficiencia de cada atributo (F3)

Es una métrica de la eficiencia de cada atributo en forma individual que describe cuánto cada atributo contribuye a la separación de dos clases. La eficiencia de cada característica es definida como la fracción de todos los puntos restantes separables por esa región. Para representar la contribución de la mayoría de las características útiles, se usa la máxima eficiencia de característica como la métrica F3 [17]. Entre más grande es el valor de F3, es mejor la eficiencia de cada atributo.

5.4 Índice de la distancia promedio intra/inter clase (N2)

Primero se calcula la distancia euclidiana de cada punto con su vecino más próximo dentro de la clase y también para su vecino más próximo fuera de la clase, se calcula el promedio de las distancias de los vecinos más próximos dentro de cada clase (interclase) y el promedio (sobre todos los puntos) de las distancias de los vecinos más próximos fuera de la clase. El índice sobre los dos promedios es utilizado como métrica N2 [17] y es lo suficientemente alto para un conjunto de datos totalmente solapado.

5.5 Índice de error del clasificador 1-NN (N3)

Esta es simplemente la tasa de errores de un clasificador del vecino próximo medido con el conjunto de entrenamiento. El índice de error es estimado por el método *leave-one-out*. Mientras mas grande sea N3 [17], menor es la proximidad entre las clases.

6 Resultados Experimentales y discusión

Con el objetivo de disminuir la complejidad de los datos en la ME y analizar su impacto sobre la precisión del clasificador se realizaron varios experimentos sobre 6 bases

Tabla 1. Características de las Bases de Datos.

DB	Clases	Atributos	ME	TST
Diabetes	2	8	614	154
German	2	24	800	200
Heart	2	25	217	55
Liver	2	6	276	69
Phoneme	2	5	4323	1081
Sonar	2	60	166	42

de datos de la UCI [18] (todas de dos clases). La Tabla 1 muestra las principales características de estas bases de datos.

A cada base se le aplico la *k-fold-cross-validation*[19] con $k = 5$, el 80% de los patrones del conjunto original fueron tomados para la ME y el 20% restante para la muestra de prueba (TST). Luego, se pre-procesaron las 5 particiones de cada DB (solo las ME, las TST se dejaron intactas) con las técnicas citadas en la sección 3. La Tabla 2 muestra el porcentaje de patrones eliminados por cada técnica, se puede observar que en el 66.66% de las bases de datos la técnica que elimino más patrones fue NCNE, mientras que en el resto fue GGE. Por el contrario, la técnica que elimino menos patrones fue RNGE para todas las bases de datos.

Tabla 2. Porcentaje de patrones eliminados por cada técnica.

DB	NNE	NCNE	GGE	RNGE	USRE
Diabetes	29.75	22.60	**32.03**	23.08	30.00
German	31.71	23.47	**33.70**	26.46	33.00
Heart	35.48	29.03	**41.47**	31.33	41.00
Liver	35.74	29.96	**38.98**	31.04	38.00
Phoneme	11.40	7.83	9.85	**12.18**	12.00
Sonar	18.56	17.96	18.56	**34.13**	34.00
Promedio	**27.11**	**21.81**	**29.10**	**26.37**	**31.33**

6.1 Análisis de la Complejidad

Mediante las métricas de complejidad mencionadas en la sección 5, se analizo si las técnicas de edición eran capaces de reducir la complejidad de los datos, es decir, limpiar la frontera entre clases (solapamiento) y mejorar la distribución de los datos (eliminar el ruido y patrones atípicos). La Fig. 2 muestra en forma de gráfica cada una de las métricas aplicadas a ME original y editadas. En resumen, se observa que la técnica GGE reduce mejor la complejidad de los datos según la métrica F1 y F2.

F3 nos dice que si se eliminan los patrones atípicos, el ruido y el solapamiento, se mejora la eficiencia de los atributos, dado que el valor más alto es para GGE. N2 nos muestra que hay una mejor separabilidad con NNE la cual se confirma con N3, y si observamos la Fig. 2 es la segunda técnica que elimina mejor el solapamiento.

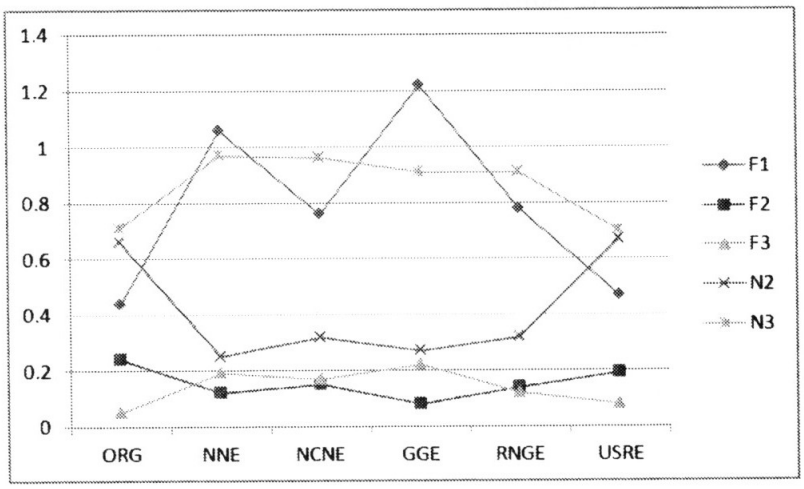


Fig. 2. Métricas de la complejidad de los datos.

6.2 Análisis de la Precisión de las RNA

Además de las muestras editadas, se construyo una nueva muestra basada en una eliminación aleatoria(se llamará USRE), para observar si hay diferencia entre eliminar patrones aleatoriamente o mediante una técnica de edición. La contrucción de la muestra USRE se basa en observar que técnica elimina más patrones, y entonces, eliminar de la ME original esa misma cantidad de patrones pero aleatoriamente (ver Tabla 2). Por ejemplo, en Heart GGE elimino un 41.47% de patrones, entonces USRE elimino aleatoriamente también el 41% (porcentaje redondeado para simplificar el proceso) de la ME original.

Tabla 3. Precisión del MLP al ser entrenado cada una de las ME.

DB	Original	NNE	NCNE	GGE	RNGE	USRE
Diabetes	75.01(4.88)	74.74(3.56)	75.65(3.05)	75.39(1.69)	75.66(4.31)	74.73(2.02)
German	70.250(2.20)	72.50(3.81)	72.50(2.57)	71.20(1.48)	73.90(2.38)	69.80(5.40)
Heart	82.59(4.83)	81.11(7.57)	78.52(4.65)	80.74(4.46)	78.52(4.83)	81.84(2.41)
Liver	70.14(4.65)	69.28(5.46)	68.12(8.51)	73.04(3.01)	64.06(4.74)	69.85(3.75)
Phoneme	80.96(1.64)	81.48(1.19)	80.92(1.34)	81.74(0.89)	81.24(1.23)	81.69(1.10)
Sonar	86.52(5.03)	82.69(1.99)	85.57(5.64)	76.91(2.86)	86.53(2.78)	86.04(4.34)
Promedio	**77.59(6.90)**	**76.97(5.55)**	**76.88(6.20)**	**76.50(4.17)**	**76.65(7.61)**	**77.33(6.85)**

Con las 6 versiones de la ME obtenidas se entrenaron las redes neuronales MLP y RBF, los resultados del entrenamiento se muestran en la tabla 3 y 4, respectivamente. En ellas, se puede ver que ninguna técnica a perjudicado o mejorado la precisión del clasificador de manera significativa. Además, los resultados nos muestran que para el

clasificador le es indiferente si los patrones se eliminaron en base a una regla (NNE, NCNE, GGE, RNGE) o aleatoriamente (USRE).

Tabla 4. Precisión del RBF al ser entrenado cada una de las ME.

DB	Original	NNE	NCNE	GGE	RNGE	USRE
Diabetes	72.91(4.93)	75.13(4.76)	72.78(5.55)	75.26(4.65)	74.35(6.67)	73.69(4.88)
German	74.50(4.11)	70.80(2.46)	72.30(3.83)	71.70(1.96)	72.60(3.27)	71.80(4.95)
Heart	80.00(6.06)	81.11(5.14)	82.59(4.65)	79.63(3.46)	80.257(4.26)	81.47(2.26)
Liver	65.22(2.29)	61.45(3.64)	61.45(2.20)	65.22(3.55)	65.80(4.42)	64.92(3.14)
Phoneme	78.57(1.29)	77.81(1.20)	77.98(1.71)	78.18(0.66)	77.54(0.78)	78.04(1.38)
Sonar	74.52(8.18)	72.57(6.61)	76.42(4.06)	77.89(6.25)	74.97(5.68)	72.01(11.87)
Promedio	**74.29(5.20)**	**73.15(6.81)**	**73.92(7.17)**	**74.65(5.40)**	**74.27(4.96)**	**73.66(5.71)**

Analizando la precisión global (PG) del clasificador, para la red MLP en un 83% de las DB la PG con el conjunto de datos original es mejor que USRE y en un 50% de las DB se obtuvo la mejor PG con RNGE. Mientras que para la red RBF en un 33% de las DB la PG fue mejor con la muestra original, y en otro 33% con GGE. Como se puede observar, no hay ninguna tendencia hacia alguna técnica y parecen ser casos aislados.

7 Conclusiones

Numerosos trabajos, han concluido que la precisión de una red neuronal depende mucho de la calidad del conjunto de datos de entrenamiento. Por tal motivo, en este trabajo se tomaron de la literatura algunas técnicas de edición aplicadas a la regla k-NN como clasificador y se aplicaron a redes neuronales artificiales.

Mediante las métricas de complejidad se puede ver que se redujo el solapamiento, ruido y patrónes atípicos, es decir, las técnicas de edición son efectivas en cuanto a reducir la complejidad. Sin embargo, ninguna técnica de las utilizadas es esta investigación aumento la precisión de alguna de las dos RNA.

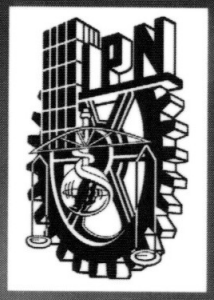
Analizando los resultados se llego a las siguientes conclusiones, las técnicas de edición no aumentaron la precisión porque son muy locales, es decir, analizan los datos en pequeñas regiones, por lo que se podría aumentar el valor de k para hacerlas más globales. Además, las bases de datos son muy pequeñas, lo que quiere decir que no están bien representadas y esto hace que el clasificador no converja satisfactoriamente, ademas, para bases de datos poco representadas como Sonar podría ser perjudicial aplicar las técnicas de edición, pues con un valor muy grande de k eliminaría más de un 90% de los patrones. Estos aspectos se abordaran en trabajos futuros con el objetivo de aumentar la precisión de las RNA.

8 Agradecimientos

Este trabajo a sido suvencionado parcialmente por el proyecto UAEMCA-114 y SBI112 de la SEP, el proyecto 2703/2008U de la UAEM y el proyecto SDMAIA-010 del Tecnológico de Estudios Superiores de Jocotitlan.

Bibliografía

1. Lippmann, R.: An introduction to computing with neural nets. SIGARCH Computer Architecture News **16**(1) (1988) 7–25
2. Russell, S. y Norvig, P.: Inteligencia Artificial. Un enfoque moderno. Prentice-Hall, EUA (1996)
3. Looney, C.: Pattern Recognition Using Neuronal Networks - theory and algorithms for engineers and scientists. 1 edn. Oxford University Press, New York (1997)
4. Jain, A., Mao, J., Mohiuddin, K.: Artificial neural networks: A tutorial. IEEE Computer **29**(3) (1996) 31–44
5. Ding, S., Xiang, C.: From multilayer perceptrons to radial basis function networks: a comparative study. In: IEEE CIS. Volume 1., Singapore, Singapore (2004) 69–74
6. Barandela, R., Gasca, E., Alejo, R.: Corrección de la muestra para el aprendizaje del perceptron multicapa. Inteligencia Artificial, Revista Iberoamericana de Inteligencia Artificial **13** (2001) 2–9
7. Visa, S., Ralescu, A.: Issues in mining imbalanced data sets - a review paper. In: Sixteen Midwest Artificial Intelligence and Cognitive Science Conference. (2005) 67–73
8. Murphey, Y., Guo, H., Feldkamp, L.: Neural learning from unbalanced data. Applied Intelligence **21** (2004) 117–128
9. Prati, R., Batista, G., Monard, M.: Class imbalances versus class overlapping: An analysis of a learning system behavior. In: MICAI. (2004) 312–321
10. Visa, S., Ralescu, A.: Learning imbalanced and overlapping classes using fuzzy sets. In: ICML. (2003) 91–104
11. García, V., Alejo, R., Sánchez, J., Sotoca, J., Mollineda, R.: Combined effects of class imbalance and class overlap on instance-based classification. In: IDEAL. (2006) 371–378
12. Wilson, D.: Asymptotic properties of nearest neighbor rules using edited data. IEEE Transactions on Systems, Man and Cybernetics **2**(4) (1972) 408–420
13. Snchez, J.S., F.P.y.F.J.F.: Using the nearest centroid neighbourhood concept for editing purpose. In: VII Simposium Nacional de Reconocimiento de formas y anlisis de imgenes. (1997) 175–180
14. Bishop, C.M.: Neural Networks for Pattern Recognition. 1 edn. Oxford University Press, USA (January 1996)
15. Buhmann, M.D.: Radial basis functions: theory and implementations. Cambridge University Press, United Kingdom (2003)
16. Hartigan, J.A.: Clustering Algorithms. Wiley (1975)
17. Mitra Basu, T.k.H.: Measures of Geometrical Complexity in Classification Problemsd. Springer-Verlag, London (2006)
18. Newman, D., Hettich, S., Blake, C., Merz, C.: UCI repository of machine learning databases. Repository, University of California, Irvine, Dept. of Information and Computer Sciences (1998) aviable at http://www.ics.uci.edu/~mlearn/MLRepository.html.
19. Devijver, P.A., Kittler, J.: Pattern Recognition: A Statistical Approach. Prentice-Hall, London (1982)



ISBN: 978-607-95367-2-5

Capítulo 6

Aportaciones, conclusiones y trabajos futuros

En la actualidad, las redes neuronales artificiales son ampliamente utilizadas en tareas de reconocimiento de patrones, minería de datos y aprendizaje automático. Lamentablemente su aplicación a problemas reales todavía está limitada por algunas debilidades tales como la lentitud en el proceso de aprendizaje y la pobre capacidad de generalización que han mostrado en diversas aplicaciones prácticas.

Numerosas investigaciones coinciden en que la efectividad del clasificador depende en gran medida de la calidad de los datos de entrenamiento, entendiéndose por esto suficiencia en la representatividad de los datos, cantidades apropiadas de patrones por clase, existencia mínima de patrones atípicos, ruido o solapamiento entre clases. En esta investigación se han estudiado algunos de estos factores y se han agrupado dentro del concepto al que se definió como *complejidad de los datos*. Asimismo se han estudiado algunas estrategias para tratar la complejidad de los datos, específicamente factores como el tamaño del conjunto de datos de entrenamiento, existencia de ruido, patrones atípicos y solapamiento entre clases

Otros factores que afectan la efectividad del clasificador como el desbalance de las clases o la dimensionalidad de los datos no fueron tratados en esta investigación por limitaciones de tiempo y espacio. Sin embargo, en futuras investigaciones pueden ser retomados para generar un estudio más completo sobre el concepto complejidad de los datos.

6.1 Aportaciones y conclusiones

El tema central de este trabajo es el estudio de la complejidad de los datos y su efecto sobre la precisión de las redes neuronales artificiales, específicamente, sobre la red MLP, RBF y como punto de comparación sobre SVM. Particularmente, esta investigación se centro en el estudio y tratamiento de problemas relacionados al tamaño de la muestra

de entrenamiento (ME), existencia de ruido, patrones atípicos y solapamiento de clases. En general, las aportaciones más importantes de este trabajo se pueden clasificar en los siguientes puntos de interés:

- En este proyecto se propone la definición del concepto complejidad de los datos de acuerdo a lo sugerido en la literatura especializada. Se definió como todos aquellos factores o características que disminuyen la calidad del conjunto de datos o muestra de entrenamiento y que por consiguiente reducen la efectividad y rendimiento del clasificador. Estos factores son el ruido, patrones atípicos, solapamiento de clases, desbalance de clases (con énfasis el problema de las clases poco representadas), el tamaño de la muestra de entrenamiento, la distribución de los datos (estos factores o características están definidas en la sección 3.1), o la dimensionalidad de los patrones de entrenamiento. La aportación más importante de esta primera parte de la investigación es la de establecer un punto de partida en la búsqueda de estrategias efectivas para reducir la complejidad de los datos. Así como el análisis de algunas métricas para su medición.

- Cuando la ME es muy grande en el entrenamiento de una RNA, el alto costo computacional asociado al proceso de aprendizaje se vuelve un problema serio. Es por esto, que se extrajeron del contexto de la regla del vecino más cercano las técnicas de la edición de Wilson, el grafo de Gabriel (GG) y la vecindad relativa (RNG) para aplicarlas sobre las conjuntos de datos utilizados en el entrenamiento de una RNA con el objetivo de disminuir el tamaño de la ME y el costo computacional asociado a este proceso. Los resultados de los experimentos muestran que la ME fue reducida hasta en un 22% (cuando una patrón era descartado si la mayoría de sus vecinos era de otra clase) a un 27% (cuando una patrón era descartado si al menos un vecino era de otra clase). Se observa que al aplicar estas técnicas en la mayoría de los casos de conserva o aumenta la precisión del clasificador.

- En esta parte del trabajo se reutilizaron las técnicas de pre-procesado de la ME (edición de Wilson y su variante del centroide más cercano, el GG y el RNG) con el objetivo de reducir el ruido, patrones atípicos y el solapamiento entre clases, y así aumentar el nivel de acierto del clasificador. También se evalúo la utilidad de algunas métricas extraídas de la literatura para medir la complejidad de los datos. Específicamente se estudiaron las siguientes: F1 (el índice discriminante de Fisher), F2 (volumen de la región de solapamiento), F3 (la máxima eficiencia de cada atributo), N2 (índice de la distancia promedio intra/inter clase) y N3 (índice de error del clasificador 1-NN).

  Mediante el úso de estas métricas de complejidad se pudo observar que se redujo el solapamiento, ruido y patrones atípicos cuando se utilizaron las estrategias de pre-procesado propuestas en este trabajo. En otras palabras la implementación de las métricas permitió observar la efectividad de las técnicas de pre-procesado para

reducir la complejidad de los datos. Sin embargo, ninguna de las técnicas de preprocesado utilizadas en esta parte de la investigación, aumentó significativamente la precisión del clasificador.

Para la red MLP, se observa que si dada alguna técnica se logra aumentar la precisión de una clase, como efecto secundario la precisión de la otra clase disminuye, esta es una causa del porque no hay ningún impacto significativo de las técnicas sobre la precisión global del clasificador. Por otra parte, en la red RBF se puede observar que en algunos casos tiene un comportamiento similar pero en otros logra aumentar la precisión de las dos clases (ver Apéndice A).

6.2 Trabajos Futuros

A lo largo de esta investigación se observaron resultados interesantes, que permitieron vislumbrar posibles escenarios hacia donde encaminar futuras investigaciones sobre el problema de la complejidad de los datos. Algunas de estas ideas son de especial interés porque han sido poco exploradas y obedecen a problemáticas actuales.

En este trabajo se utilizaron los criterios de medida F1 y el de los k-NN para medir de alguna forma el nivel de solapamiento o separabilidad entre clases. No obstante, en algunos casos estos criterios no fueron suficientes para relacionar estos factores con la efectividad de la red neuronal. Es necesario buscar estrategias específicas para medir los niveles de solapamiento o separabilidad entre clases dentro del contexto de las redes neuronales.

Si se cuenta con estos mecanismos se pueden optimizar las estrategias diseñadas para reducir la región de confusión o identificar con mayor precisión el tratamiento que se le deba dar a cada conjunto de datos en función de sus características propias.

Por otro lado, la trascendencia real de esta propuesta esta en que los criterios empleados para identificar los elementos redundantes o en la frontera de decisión consideren características particulares de la red neuronal artificial. Por ejemplo, la búsqueda de los elementos representativos puede darse en el espacio oculto o de salida en lugar del espacio de características, y el criterio de medida puede estar basado en valores de MSE (Error cuadrático medio).

Apéndice A

Precisión por clase

En el articulo presentado en la sección 5.4 no se incluye el análisis de la precisión por clase por motivos un tanto fuera del alcance de nuestras manos. Para poder entender los resultados de una manera más apropiada en necesario analizar la precisión por clase, porque parte de las conclusiones de este proyecto se derivan de su análisis. La Tabla A.1 muestra la Precisión por clase de las redes MLP y RBF, de cada una de las bases de datos y las diferentes variantes de la ME (Muestra Original, NNE, NCNE, GGE, RNGE y USRE). Para la red MLP, se observa que si dada alguna técnica se logra aumentar la precisión de una clase como efecto secundario la precisión de la otra clase disminuye, esta es una causa del porque no hay ningún impacto significativo de las técnicas sobre la precisión el clasificador. Por otra parte, en la red RBF se puede observar que que en algunos casos tiene un comportamiento similar pero en otros casos logra aumentar la precisión de las dos clases.

MLP												
BD	Original		NNE		NCNE		GGE		RNGE		USRE	
	Cl1	Cl2	Cl1	Cl2	Cl1	Cl2	Cl1	Cl2	Cl1	Cl2	Cl1	Cl2
Diabetes	0.553(0.107)	0.856(0.033)	0.559(0.100)	0.848(0.015)	0.593(0.099)	0.844(0.036)	0.530(0.054)	0.874(0.030)	0.604(0.087)	0.838(0.066)	0.596(0.131)	0.828(0.085)
German	0.470(0.072)	0.803(0.034)	0.353(0.084)	0.884(0.033)	0.410(0.038)	0.860(0.024)	0.177(0.069)	0.942(0.014)	0.387(0.080)	0.890(0.021)	0.477(0.086)	0.796(0.058)
Heart	0.808(0.087)	0.840(0.068)	0.717(0.130)	0.887(0.038)	0.692(0.152)	0.860(0.064)	0.742(0.112)	0.860(0.037)	0.700(0.075)	0.853(0.030)	0.833(0.084)	0.807(0.076)
Liver	0.820(0.078)	0.538(0.079)	0.830(0.078)	0.503(0.099)	0.765(0.135)	0.566(0.093)	0.880(0.027)	0.524(0.051)	0.755(0.116)	0.483(0.069)	0.775(0.088)	0.593(0.086)
Phoneme	0.871(0.020)	0.661(0.050)	0.884(0.026)	0.649(0.074)	0.876(0.031)	0.648(0.098)	0.876(0.027)	0.676(0.062)	0.859(0.014)	0.700(0.020)	0.853(0.015)	0.729(0.032)
Sonar	0.855(0.079)	0.874(0.067)	0.763(0.068)	0.883(0.024)	0.824(0.032)	0.884(0.079)	0.691(0.126)	0.838(0.094)	0.805(0.063)	0.919(0.068)	0.814(0.050)	0.901(0.060)
Prom	**0.729(0.172)**	**0.762(0.134)**	**0.684(0.197)**	**0.776(0.162)**	**0.693(0.171)**	**0.777(0.135)**	**0.649(0.265)**	**0.786(0.156)**	**0.685(0.170)**	**0.780(0.164)**	**0.725(0.153)**	**0.776(0.105)**

RBF												
BD	Original		NNE		NCNE		GGE		RNGE		USRE	
	Cl1	Cl2	Cl1	Cl2	Cl1	Cl2	Cl1	Cl2	Cl1	Cl2	Cl1	Cl2
Diabetes	0.522(0.131)	0.840(0.019)	0.567(0.118)	0.850(0.030)	0.519(0.127)	0.840(0.020)	0.537(0.118)	0.868(0.011)	0.530(0.144)	0.858(0.033)	0.481(0.155)	0.874(0.045)
German	0.493(0.093)	0.853(0.051)	0.284(0.042)	0.890(0.032)	0.340(0.025)	0.887(0.052)	0.167(0.062)	0.953(0.017)	0.323(0.052)	0.899(0.050)	0.467(0.041)	0.826(0.070)
Heart	0.809(0.109)	0.793(0.055)	0.750(0.121)	0.860(0.044)	0.825(0.146)	0.827(0.043)	0.733(0.152)	0.847(0.090)	0.733(0.124)	0.860(0.028)	0.817(0.130)	0.813(0.102)
Liver	0.745(0.076)	0.524(0.075)	0.745(0.045)	0.435(0.058)	0.740(0.063)	0.441(0.123)	0.835(0.049)	0.400(0.079)	0.775(0.059)	0.496(0.058)	0.730(0.057)	0.538(0.079)
Phoneme	0.845(0.032)	0.644(0.053)	0.868(0.019)	0.563(0.037)	0.861(0.016)	0.584(0.024)	0.852(0.018)	0.612(0.039)	0.858(0.019)	0.576(0.038)	0.862(0.018)	0.584(0.030)
Sonar	0.795(0.126)	0.702(0.167)	0.680(0.087)	0.766(0.086)	0.764(0.084)	0.766(0.109)	0.713(0.088)	0.839(0.108)	0.732(0.063)	0.765(0.119)	0.647(0.157)	0.783(0.118)
Prom	**0.701(0.154)**	**0.726(0.128)**	**0.649(0.204)**	**0.727(0.186)**	**0.675(0.203)**	**0.724(0.174)**	**0.640(0.257)**	**0.753(0.207)**	**0.659(0.197)**	**0.742(0.168)**	**0.667(0.167)**	**0.736(0.140)**

Tabla A.1: Precisión por clase del entrenamiento de la redes MLP y RBF.

Bibliografía

[Alejo 2006] R. Alejo, V. García, J.M. Sotoca, R.A. Mollineda and J.S. Sánchez. *Improving the classification accuracy of RBF and MLP neural networks trained with imbalanced samples.* En IDEAL, Volumen 4224, Pág. 464–471, Burgos, España, 2006.

[Alejo 2007] R. Alejo, V. García, J.M. Sotoca, R.A. Mollineda and J.S. Sánchez. *Improving the Performance of the RBF Neural Networks with Imbalanced Samples.* En IWANN, Pág. 162–169, San Sebastián, España, 2007. Springer Berlin / Heidelberg.

[Alejo 2008] R. Alejo, J.M. Sotoca and G. A. Casañ. *An Empirical Study for the Multiclass Imbalance Problem with Neural Networks.* En CIARP, Pág. 479–486, 2008.

[Anand 1993] R. Anand, K.G. Mehrotra, C.K. Mohan and S. Ranka. *An improved Algorithm for Neural Network Classification of Imbalanced Training Sets.* IEEE Transactions on Neural Networks, Vol. 4, Pág. 962–969, 1993.

[Anand 1995] R. Anand, K. Mehrotra, C.K. Mohan and S Ranka. *Efficient classification for multiclass problems using modular neural networks.* IEEE Transactions on Neural Networks, Vol. 6, No. 1, Pág. 117–124, 1995.

[Anderson 1977] J.A. Anderson, J.W. Silverstein, S.A. Ritz and R.S. Jones. *Distinctive features, categorical perception, and probability learning: Some applications of a neural model.* Psychological Review, Vol. 84, Pág. 413–451, 1977.

[Ariza 1996] F.J. Ariza, C. Pinilla and M.J. Borque. *Control de Calidad del Proceso de Clasificación de Imágenes de Satélite.* Mapping, No. 34, Pág. 74–86, 1996.

[Atiya 1997] A. y C. Ji Atiya. *How initial conditions affect generalization performance in large networks.* IEEE, Vol. 8, No. 2, Pág. 448–451, 1997.

[Barandela 2000] R. Barandela and E. Gasca. *Decontamination of Training Samples for Supervised Pattern Recognition Methods.* En SSPR/SPR, Pág. 621–630, 2000.

[Barandela 2001] R. Barandela, E. Gasca and R. Alejo. *Corrección de la Muestra para el Aprendizaje del Perceptron Multicapa.* Inteligencia Artificial, Revista Iberoamericana de Inteligencia Artificial, Vol. 13, Pág. 2–9, 2001.

[Barandela 2002] R. Barandela, E. Gasca and R. Alejo. *Correcting the Training Data.* Combinatorial Optimization -Dordrecht-, Vol. 13, Pág. 1–42, 2002.

[Benoudjit 2003] N. Benoudjit and M. Verleysen. *On the Kernel Widths in Radial-Basis Function Networks.* Neural Processing Letters, Vol. 18, No. 2, Pág. 139–154, 2003.

[Brunak 1990] S. Brunak and B. Lautrup. Neural networks: computers with intuition. World Scientific Publishing Co., Inc., River Edge, NJ, USA, 1990.

[Bruzzone 1997a] L. Bruzzone and S.B. Serpico. *Classification of imbalanced remote-sensing data by neural networks.* Pattern Recognition Letters, Vol. 18, Pág. 1323–1328, 1997.

[Bruzzone 1997b] L. Bruzzone and S.B. Serpico. *Training of neural networks for classification of imbalanced remote-sensing data.* En IGARSS, Volumen 3, Pág. 1202–1204, Singapore, Singapore, 1997.

[Burges 1998] C. J Burges. *A Tutorial on Support Vector Machines for Pattern Recognition.* Data Min. Knowl. Discov., Vol. 2, No. 2, Pág. 121–167, 1998.

[Chan 1999] P.K. Chan, W. Fan, A.L. Prodromidis and S.J. Stolfo. *Distributed Data Mining in Credit Card Fraud Detection.* IEEE Intelligent Systems, Vol. 14, No. 6, Pág. 67–74, 1999.

[Cybenko 1989] G. Cybenko. *Approximation by superpositions of a sigmoidal function.* Mathematics of Control, Signals, and Systems (MCSS), Vol. 2, No. 4, Pág. 303–314, 1989.

[DARPA-USA 1988] DARPA-USA. Darpa neural network study. AFCEA Intl, 1988.

[Ding 2004] S.Q. Ding and C. Xiang. *From multilayer perceptrons to radial basis function networks: a comparative study.* En IEEE CIS, Volumen 1, Pág. 69–74, Singapore, Singapore, 2004.

[Duda 2001a] Hart Peter E. Stork David G. Duda Richard O. Pattern classification. Wiley, 2001.

[Duda 2001b] R.O. Duda, P.E. Hart and D.G. Stork. Pattern classification and scene analysis. Wiley, New York, 2 edición, 2001.

[Fahlman 1988] S.E. Fahlman. *Faster-Learning Variations on Back-Propagation: An Empirical Study.* En CMSS, Pág. 38–51, Carnegie Mellon University, Pittsburgh, PA, 1988.

[Fawcett 1997] T. Fawcett and F. Provost. *Adaptive Fraud Detection.* Data Mining and Knowledge Discovery, Vol. 1, No. 3, Pág. 291–316, 1997.

[Foody 1995] McCulloch M. Foody G. and W. Yates. *The effect of training set size and composition on artifial neural networks classification.* International Journal of Remote Sensing, Vol. 16, No. 9, Pág. 1707–1723, 1995.

[Freeman 1991] J.A. Freeman and D.M. Skapura. Neural networks: algorithms, applications, and programming techniques. Addison Wesley Longman Publishing Co., Inc., Redwood City, CA, USA, 1991.

[Fukushima 1979] K. Fukushima. *Neocognitron: A self-organizing neural network model for a mechanism of pattern recognition unaffected by shift in position.* Biological Cybernetics, Vol. 36, Pág. 193–202, 1979.

[Funahashi 1989] K. Funahashi. *On the approximate realization of continuous mappings by neural networks.* Neural Networks, Vol. 2, No. 3, Pág. 183–192, 1989.

[García 2006] V. García, R. Alejo, J.S. Sánchez, J.M. Sotoca and R.A. Mollineda. *Combined Effects of Class Imbalance and Class Overlap on Instance-Based Classification.* En IDEAL, Pág. 371–378, 2006.

[Ghodsi 2003] A. Ghodsi and D. Schuurmans. *Automatic basis selection techniques for RBF networks.* Neural Networks, Vol. 16, No. 5-6, Pág. 809–816, 2003.

[Golub 1996] G.H. Golub and C.F. Van Loan. Matrix computations. Johns Hopkins University Press, Baltimore, MD, USA, 3 edición, 1996.

[Gopalakrishnan 1995] V. SridhR Y H. KrishNamurthy Gopalakrishnan M. *Some application of clustering in the design of neural networks.* Patter Recognition Letters, Vol. 16, No. -, Pág. 59–65, 1995.

[Gray 1984] R. Gray. *Vector quantization.* ASSP Magazine, IEEE Signal Processing Magazine, Vol. 1, No. 2, Pág. 4–29, 1984.

[Grossberg 1987] S. Grossberg. *Competitive learning: from interactive activation to adaptative resonance.* Cognitive Science, Vol. 11, Pág. 23–63, 1987.

[Guérin-Dugué 1995] A. Guérin-Dugué*et al. Deliverable R3-B1-P - Task B1: Databases.* Reporte Técnico 6891, Elena-NervesII "Enhanced Learning for Evolutive Neural Architecture", ESPRIT-Basic Research Project, Junio 1995. FTP: /pub/neural-nets/ELENA/Databases.ps.Z en ftp.dice.ucl.ac.be.

[Haibo 2009] He y Edwardo A. Garcia Haibo. *Learning from Imbalanced Data.* IEEE transactions on knowledge and data engineering, Vol. 21, No. 9, Pág. 1263–1284, 2009.

[Han 2006] J. Han and M. Kamber. Data mining: Concepts and techniques. Cambridge University Press, 500 Sansome Street, Suite 400, San Francisco, CA 94111, 2006.

[Harpham 2004] C. Harpham, W. Dawson and R. Brown. *A review of genetic algorithms applied to training radial basis function networks.* Neural Computing and Applications, Vol. 13, No. 3, Pág. 193–201, 2004.

[Haykin 1999] S. Haykin. Neural networks. a comprehensive foundation. Pretince Hall, New Jersey, 2 edición, 1999.

[He 2009] H. He and E.A. Garcia. *Learning from Imbalanced Data.* IEEE Transactions on Knowledge and Data Engineering, Vol. 21, No. 9, Pág. 1263–1284, 2009.

[Hebb 1949] D. Hebb. The organization of behavior - a neurophysiological theory. John Wiley & Sons, New York, USA, 1 edición, 1949.

[Hecht-Nielsen 1990] R. Hecht-Nielsen. Neurocomputing. Addison-Wesley, Massachusetts, 1990.

[Hilera 1995] V.J. Hilera J.R.and Martínez. Redes neuronales artificiales. fundamentos, modelos y aplicaciones. Ra-Ma, Madrid, España, 1 edición, 1995.

[Hilera 2000] Víctor J. Hilera José R. y Martínez. Redes neuronales artificiales. Alfaomega, 2000.

[Hoekstra 1996] Duin R. P. W. Hoekstra A. *On the nonlinearity of pattern classifiers.* En 13th International Conference on Patter Recognition, Pág. 271–275, 1996.

[Hopfield 1982] J.J. Hopfield. *Neural networks as physical systems with emergent collective computational abilities.* Proc. National Academy of Sciences of USA, Vol. 79, No. 8, Pág. 2554–2558, 1982.

[Horton] P. Horton and K. Nakai. *A Probabilistic Classification System for Predicting the Cellular Localization Sites of Proteins.*

[Hutchinson 1995] J.M. Hutchinson, A. Lo and T. Poggio. *A Nonparametric Approach to Pricing and Hedging Derivative Securities Via Learning Networks.* NBER Working Papers 4718, National Bureau of Economic Research, Inc, February 1995. available at http://ideas.repec.org/p/nbr/nberwo/4718.html.

[Ian 2005] H.W. Ian and F. Eibe. Data mining: Practical machine learning tools and techniques. Morgan Kaufmann, San Francisco, CA, 2 edición, 2005.

[Jacobs 1988] R.A. Jacobs. *Increased rates of convergence through learning rate adaptation.* Neural Networks, Vol. 1, No. 4, Pág. 295–308, 1988.

[Jadid 1996] M.N. Jadid and D.R. Fairbairn. *Neural-network Applications in Predicting Moment-curvature Parameters from Experimental Data.* Engineering Applications of Artificial Intelligence, Vol. 9, No. 3, Pág. 309–319, 1996.

[Jain 1996] A.K. Jain, J. Mao and K.M. Mohiuddin. *Artificial Neural Networks: A Tutorial.* IEEE Computer, Vol. 29, No. 3, Pág. 31–44, 1996.

[Japkowicz 2002] N. Japkowicz and S. Stephen. *The class imbalance problem: a systematic study.* Intelligent Data Analysis, Vol. 6, Pág. 429–449, 2002.

[Joachims 1999] T. Joachims. *Making large-scale support vector machine learning practical.* En In A.S.B. Scholkopf, C. Burges, editor, Advances in Kernel Methods: support vector machine, Pág. pp.169–184, 1999.

[Jordan 1994] M.I. Jordan and R.A. Jacobs. *Hierarchical mixtures of experts and the EM algorithm.* Neural Computation, Vol. 6, No. 2, Pág. 181–214, 1994.

[Jordan 1995] M.I. Jordan and L. Xu. *Convergence results for the EM approach to mixtures of experts architectures.* Neural Networks, Vol. 8, No. 9, Pág. 1409–1431, 1995.

[Kanellopoulos 1997] I. Kanellopoulos and G.G. Wilkinson. *Strategies and Best Practice for Neural-Network Image Classification.* International Journal of Remote Sensing, Vol. 18, No. 4, Pág. 711–725, March 1997.

[Kemley 1992] David H. Kemley and Tony R. martinez. *A Survey of Neural Network Research anf Fielded Application.* In International Journal of Neural Networks: Research and Applications, Vol. 2, No. 2/3/4, Pág. 123–133, 1992.

[Kohonen 1977] T. Kohonen. Associative memory. a system-theoretical approach. Springer-Verlag, New York, 1977.

[Kohonen 1990] T. Kohonen. *The Self-Organizing Map.* En Proceedings of the IEEE, Pág. 1464–1480, 1990.

[krishman 1988] D. Walters krishman G. *Psychologically Plausible Features for Shapes Recognition in a Neural-Networks.* En IEEE international Conferece on Neural Networks 2, Pág. 127–134, 1988.

[Kubat 1998] M. Kubat. *Decision Trees Can Initialize Radial-Basis Function Networks.* IEEE Transactions on Neural Networks, Vol. 9, No. 5, Pág. 813–821, September 1998.

[Kuncheva 2004] L.I. Kuncheva. Combining pattern classifiers: Methods and algorithms. Wiley-Interscience, July 2004.

[Lachtermacher 1995] G. Lachtermacher and J.D. Fuller. *Article Back propagation in time-series forecasting.* Journal of Forecasting, Vol. 14, No. 4, Pág. 381–393, 1995.

[Lee 1997] C. W. Lee. *Training feedforward Neural Networks: An algorithm giving improved generalization.* Neural networks, Vol. 10, No. 1, Pág. 61–68, 1997.

[Lippmann 1988] R.P. Lippmann. *An introduction to computing with neural nets.* SIGARCH Computer Architecture News, Vol. 16, No. 1, Pág. 7–25, 1988.

[Looney 1997] C. Looney. Pattern recognition using neuronal networks - theory and algorithms for engineers and scientists. Oxford University Press, New York, 1 edición, 1997.

[Lowe 1989] D. Lowe. *Adaptive radial basis function non linearities, and the problem of generalisation.* En ICANN, Pág. 171–175, 1989.

[Lu 1998] Y. Lu, H. Guo and L. Feldkamp. *Robust neural learning from unbalanced data examples.* En IJCNN, Pág. 1816–1821, 1998.

[Marques de Sa 2001] J.P Marques de Sa. Pattern recognition, concepts methods and applications. Springer, Alemania, 2001.

[Martín 2001] B. Martín and A. Sanz. Redes neuronales y sistemas borrosos. Alfaomega-Rama, 2001.

[Masters 1993] T. Masters. Practical neural network recipes in c++. Academic Press Professional, Inc., San Diego, CA, USA, 1993.

[McCulloch 1943] W.S. McCulloch and W. Pitts. *A logical calculus of the ideas immanent in nervous activity.* Bulletin of Mathematical Biophysics, Vol. 5, Pág. 115–133, 1943.

[Mease 2007] Wyner A. J. y Buja J. Mease D. *Boosted Classification Trees and Class Probability/ Quantile estimation.* J. machine learning research, Vol. 8, No. -, Pág. 409–439, 2007.

[Midorikawa 1988] H. Midorikawa. *The Face Pattern Identificación by Bak-Propagation Learning Procedure.* Neural Networks, Vol. 1, No. 1, Pág. 515, 1988.

[Minsky 1969] Marvin Minsky and Seymour Papert. Perceptrons: An introductions to computacional geometry. -, 1969.

[Mitra 2006] Basu Mitra and Kam Ho Tin. Data complexity in pattern recognition. pringer-Verlag, London, 2006.

[Moody 1989] J. Moody and C. Darken. *Fast learning in networks of locally-tuned processing units*. Neural Computation, Vol. 1, No. 2, Pág. 281–294, 1989.

[Murphey 2004] Y. Murphey, H. Guo and L. Feldkamp. *Neural Learning from Unbalanced Data*. Applied Intelligence, Vol. 21, Pág. 117–128, 2004.

[Nakai 1991] K. Nakai and M. Kanehisa. *Expert system for predicting protein localization sites in gram-negative bacteria*. Proteins: Structure, Function, and Bioinformatics, Vol. 11, No. 2, Pág. 95–110, 1991.

[Newman 1998] D.J. Newman, S. Hettich, C.L. Blake and C.J. Merz. *UCI Repository of Machine Learning Databases*. Repository, University of California, Irvine, Dept. of Information and Computer Sciences, 1998. aviable at http://www.ics.uci.edu/∼mlearn/MLRepository.html.

[Pao 1989] Y.-H. Pao. Adaptive pattern recognition and neural networks. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 1989.

[Papik 1998] K. Papik, B. Molnar, R. Schaefer, Z. Dombovari, Z. Tulassay and J. Feher. *Application of neural networks in medicine - a review*. Diagnostics and Medical Technology, Vol. 4, No. 3, Pág. 538–546, 1998.

[Parker 1985] D.B. Parker. *Learning-logic*. Technical Report 47, Center for Comp. Research in Economics and Management Sci., MIT, April 1985.

[Platt 1998] J. Platt. *Fast Training of support vector machine using sequential minimal optimization*. En In A.S.B. Scholkopf, C. Burges, editor, Advances in Kernel Methods: support vector machine, 1998.

[Platt 1999] John C. Platt. Advances in kernel methods - fast training of support vector machines using sequential minimal optimization. MIT Press, Cambridge, MA, USA, 1 edición, 1999.

[Powell 1987] M. J. D. Powell. *Radial basis functions for multivariable interpolation: a review*. Algorithms for approximation, Pág. 143–167, 1987.

[Prati 2004] R.C. Prati, G.E.A.P.A. Batista and M.C. Monard. *Class Imbalances versus Class Overlapping: An Analysis of a Learning System Behavior.* En MICAI, Pág. 312–321, 2004.

[Roli 1996] F. Roli, S.B. Serpico and L. Bruzzone. *Classification of Multisensor Remote-Sensing Images by Multiple Structured Neural Networks*. En ICPR, Volumen 4, Pág. 180, Washington, DC, USA, 1996. IEEE Computer Society.

[Rosenblatt 1958] F. Rosenblatt. *The perceptron: A probabilistic model for information storage and organization in the brain.* Psycological Review, Vol. 65, No. 6, Pág. 386–408, 1958.

[Rumelhart 1986] D.E. Rumelhart, G.E. Hinton and R.J. Williams. Learning internal representations by error propagation. MIT Press, Cambridge, MA, USA, 1986.

[Russell 1996] P. Russell S. y Norvig. Inteligencia artificial. un enfoque moderno. Prentice-Hall, EUA, 1996.

[Russell 1999] G.C. Russell and G. Kass. Assessing the overall accuracy of remotely sensed data: Principles and practice. Lewis Publishers, New York, Washington, D.C., 1999.

[Saha 1989] A. Saha and J.D. Keeler. *Algorithms for Better Representation and Faster Learning in Radial Basis Function Networks.* En NIPS, Pág. 482–489, 1989.

[Saito 1988] R. Nakato Saito K. *Medical Diagnostic Expert System based on PDP Model.* En IEEE international Conferece on Neural Networks 1, Pág. 255–262, 1988.

[Sánchez 2001] J.S. Sánchez, R. Barandela, A.I. Marqués and R. Alejo. *Performance Evaluation of Prototype Selection Algorithms for Nearest Neighbor Classification.* En SIBGRAPI, Pág. 44–50, Washington, DC, USA, 2001. IEEE Computer Society.

[Sánchez 2006] García Alma Y. Sánchez Camperos Edgar N. y Alanís. Redes neuronales: Conceptos fundamentales y aplicaciones a control automático. Pearson-Prentice Hall, 2006.

[Schölkopf 1997] B. Schölkopf, Sung K.-K., Burges C.J.C., F. Girosi, P. Niyogi, T. Poggio and V. Vapnik. *Comparing support vector machines with Gaussian kernels to radial basis function classiers.* IEEE Transactions on Signal Processing, Vol. 45, No. 11, Pág. 2758–2765, 1997.

[Schwenker 2001] F. Schwenker, H.A. Kestler and G. Palm. *Three learning phases for radial-basis-function networks.* Neural Networks, Vol. 14, No. 4-5, Pág. 439–458, 2001.

[Serpico 1993] S.B. Serpico, F. Roli, P. Pellegretti and G. Vemazza. *Structured neural networks for the classification of multisensor remote-sensing images.* En IGARSS, Pág. 907–909, Tokyo, Japan, 1993.

[Shawe-Taylor 2000] J. Shawe-Taylor and N. Cristianini. An introduction to support vector machines and other kernel-based learning methods. Cambridge University Press, Cambridge, 2000.

[Simpson 1990] P. K. Simpson. Artificial neural systems: Foundation, paradigms, applications and implementations. Pergamon press, New York, USA, 1990.

[Smith 1968] F. W. Smith. *Patter classifier design by linear programming.* IEEE Transactions on Computers, Vol. C-17, No. 4, Pág. 367–372, 1968.

[Sánchez 1997] F. Pla y F. J. Ferri Sánchez J.S. *Using the nearest centroid neighbourhood concept for editing purpose.* En VII Simposium Nacional de Reconocimiento de formas y análisis de imágenes, Pág. 175–180, 1997.

[Tong 2001] Simon Tong and Edward Chang. *Support vector machine active learning for image retrieval.* En Proceedings of the ninth ACM international conference on Multimedia, MULTIMEDIA '01, Pág. 107–118, New York, NY, USA, 2001. ACM.

[Upadhyaya 1992] B.R. Upadhyaya and E. Eryurek. *Application of Neural Networks for Sensor Validation and Plant Monitoring.* Nuclear Technology, Vol. 97, No. 2, Pág. 170–176, 1992.

[Uykan 2000] Z. Uykan, C. Guzelis, M.E. Celebi and H.N. Koivo. *Analysis of input-output clustering for determining centers of RBFN.* IEEE Transactions on Neural Networks, Vol. 11, No. 4, Pág. 851–858, 2000.

[Valdovinos 2006] R.M. Valdovinos. *Técnicas de Submuestreo, Toma de Decisiones y Análisis de Diversidad en Aprendizaje Supervisado con Sistemas Múltiples de Clasificación.* Tesis doctoral, Universitat Jaume I, Castellón, España, Septiembre 2006.

[Visa 2003] S. Visa and A. Ralescu. *Learning imbalanced and overlapping classes using fuzzy sets.* En ICML, Pág. 91–104, 2003.

[Visa 2005] S. Visa and A. Ralescu. *Issues in Mining Imbalanced Data Sets - A Review Paper.* En Sixteen Midwest Artificial Intelligence and Cognitive Science Conference, Pág. 67–73, 2005.

[Werbos 1974] P. Werbos. *Beyond Regression: New Tools for Prediction and Analysis in the Behavioral Sciences.* Tesis doctoral, Harvard University, Cambridge, MA, 1974.

[Wettschereck 1992] D. Wettschereck and T.G. Dietterich. *Improving the performance of Radial Basis Function Networks by Learning Center Locations.* En J.E. Moody, S.J. Hanson and R.P. Lippmann, editores, NIPS 4, Volumen 4, Pág. 1133–1140. Morgan Kaufmann, San Mateo, CA, 12 1992.

[Widrow 1960] B. Widrow and M.E. Hoff. *Adaptative switching circuits.* IRE WESCON Convention Records, Pág. 96–104, 1960.

[Wilson 1972] D.L. Wilson. *Asymptotic properties of nearest neighbor rules using edited data*. IEEE Transactions on Systems, Man and Cybernetics, Vol. 2, No. 4, Pág. 408–420, 1972.

[Xu 1995] L. Xu, M.I. Jordan and G.E. Hinton. *An Alternative Model for Mixtures of Experts*. En G. Tesauro, D. Touretzky and T. Leen, editores, NIPS, Volumen 7, Pág. 633–640. The MIT Press, 1995.

[Xu 1998] L. Xu. *RBF nets, mixture experts, and Bayesian Ying-Yang learning*. Neurocomputing, Vol. 19, No. 1-3, Pág. 223–257, 1998.

[Zhou 2006] Z.-H. Zhou and X.-Y. Liu. *Training cost-sensitive neural networks with methods addressing the class imbalance problem*. IEEE Transactions on Knowledge and Data Engineering., Vol. 18, Pág. 63–77, 2006.