Estrategias pedagógicas para la presentación de patrones al entrenamiento de redes neuronales artificiales de tipo MLP que utilizan backpropagation como algoritmo de aprendizaje

Adolfo Díaz Sardiñas, Rafael Bello Pérez *

Resumen

El problema de las estrategias de selección de patrones para el entrenamiento de redes neuronales no ha recibido mucha atención por parte de los investigadores de este campo. Durante el entrenamiento Backpropagation (BP) usualmente todos los patrones son presentados de forma secuencial en un orden uniforme, sin tener en cuenta posibles estructuraciones del conjunto de entrenamiento que posibilitarían drásticas reducciones del tiempo de duración del proceso de aprendizaje. Este trabajo presenta y compara algunas estrategias de selección de patrones para adaptar el proceso de aprendizaje a las características del problema en particular que se esté tratando. Estas estrategias favorecen la selección de patrones que producen valores de error altos, restando prioridad a aquellas que ya han sido "aprendidas" por la red (que producen errores muy bajos). Pueden existir dos tipos de estrategias: *aleatorias* y *determinísticas*. En las estrategias aleatorias, los patrones son seleccionados aleatoriamente atendiendo a alguna variable de probabilidad que depende del estado del proceso de aprendizaje. Por el contrario, las determinísticas siguen esquemas globales y predefinidos que incrementan la frecuencia de presentación de ciertos patrones forzando su repetición. Hemos formalizado cinco variantes de estrategias que se describen en la literatura (Cachin, 1994 y Munro, 1992) que son: (1) *probabilidad de presentación dependiente del error (PPDE),* de tipo aleatorio; (2) *repetición dependiente del error (RDE),* de tipo determinista; (3) *sistema de fichero de tarjetas (SFT),* que posee dos variantes, una aleatoria y otra determinista; (4) *partición del conjunto de entrenamiento (PCE),* de tipo determinista; y (5) *repetir hasta aprender (RHA),* de tipo aleatorio. Además proponemos dos nuevas estrategias: (1) *repetir el mayor error (RME),* de tipo determinista y (2) *sistema de fichero de tarjetas dependiente del error (SFTDE)* también de tipo determinista. Los experimentos han mostrado que en los mejores resultados para los problemas de prueba, el tiempo de convergencia y la calidad del aprendizaje pueden ser mejorados, pero solo mediante estrategias de tipo determinista.

---------- *Palabras clave*: redes neuronales, patrones de entrenamiento, estrategias de selección de patrones, procesos de aprendizaje, estrategias aleatorias, estrategias deterministas.

* Facultad de Matemática, Física y Computación, Universidad Central de Las Villas Santa Clara Cuba, rbellop@uclv.etecsa.cu.

Abstract

The problem of strategy selection for neuronal net training has not received much attention from researchers in this field. In Backpropagation training, usually patterns are presented in a sequential form and uniform order, without taking into account the possible structures in the training set, which would permit severe reductions in the learning process time. This paper presents and compares some strategies for pattern selection, in order to adopt the learning process to the characteristics of the problem in hand. These strategies favor pattern selection that yield high error values, giving less attention to those already "learned" by the net (that yield low errors).

It could exist two strategy types: random and deterministic. In the first, patterns are random selected following a probability variable which depends on the learning process stage of advance. On the contrary, deterministic ones follow global and determined schemes that increase presentation frequency of certain patterns that enforce repetition.

Five known strategy variants are formalized here: (1) Error Dependant Probability Presentation, (2) Error Dependant Repetition, (3) Card File System, (4) Training Set Partitioning and (5) Repeat until Learn. Also two new strategies are presented: (1) Repeat the Higher Error and (2) Error Dependant Card File System. Experiments showed that in the best results for test problems, convergence time and learning quality could be improved, but only by means of deterministic strategies.

---------- *Key words*: neuronal nets, training patterns, pattern slection strategies, learning processes, random strategy, deterministic strategy.

1. Introducción

Aunque los mecanismos de aprendizaje para redes multicapas se conocen desde hace más de treinta años, la ausencia de algoritmos de entrenamiento adecuados ha conspirado contra su uso exitoso en aplicaciones prácticas. Investigaciones relativamente recientes en métodos de entrenamiento supervisados para redes de este tipo, utilizando para cada capa una función de discriminación lineal ha provocado la extensión y éxito que actualmente presenta. Las redes multicapas (MLP) pueden implementar mapas entrada/salida o separaciones de clases en regiones arbitrariamente complejas. El atributo más importante es que puede aprender a mapear cualquier complejidad. Su aprendizaje se basa en la repetición de presentaciones de los ejemplos de entrenamiento y ya entrenada puede producir resultados sorprendentes y generalizaciones en aplicaciones donde las derivaciones explícitas de mapeos y el descubrimiento de relaciones es casi imposible por otra vía.

Durante el entrenamiento de redes neuronales hacia adelante (feedforward) utilizando Backpropagation (BP) como algoritmo de aprendizaje, muchos factores afectan la calidad del aprendizaje (Cachin, 1994), entre los que se pueden señalar la arquitectura de la red, el tipo de neurona, los parámetros de entrenamiento, las medidas de error, así como la estrategia para seleccionar patrones.

Durante los últimos años se han propuesto diferentes métodos para contrarrestar estas deficiencias, entre los que están los que tratan de determinar el conjunto óptimo de patrones de entrenamiento, pero en la práctica en las aplicaciones de redes neuronales, especialmente en reconocimiento de patrones, es usual encontrarse con conjuntos de entrenamientos fijos sin tener en cuenta que en la actualidad tiene especial importancia el uso de estrategias para el entrenamiento. Con la excepción de Munro (1992) y Cachin (1994), el proceso de selección de patrones no ha recibido mucha atención en las investigaciones de redes neuronales. Todos los patrones son presentados usualmente de acuerdo con un esquema predefinido en igualdad de condiciones para todos, sin tener en cuenta el progreso del aprendizaje.

En este trabajo presentamos y analizamos algunas alternativas de estrategias de selección de patrones que adaptan el proceso de aprendizaje, también llamadas *pedagógicas* por Munro (1992) y Cachin (1994). Sus propiedades comunes es que incrementan la frecuencia de presentación de patrones que producen altos valores de error y restan prioridad a aquellos que ya han sido "aprendidos" por la red. Algunas de las estrategias están inspiradas en analogías del aprendizaje humano.

Las estrategias presentadas son de dos tipos, aleatorias y deterministas. En una estrategia aleatoria los patrones son seleccionados aleatoriamente de acuerdo con alguna variable de probabilidad dependiente del estado del proceso de aprendizaje. Por el contrario, las estrategias deterministas siguen un esquema global y predefinido que determina la frecuencia de presentación de ciertos patrones forzando su repetición.

Usualmente los patrones son seleccionados aleatoriamente con igual probabilidad o todo el conjunto de entrenamiento es presentado cíclicamente, eventualmente permutando su orden en forma aleatoria después de un ciclo completo. Como todos los ejemplos son presentados en igualdad de condiciones, la llamaremos estrategia *uniforme*.

En etapas avanzadas del aprendizaje, las estrategias uniformes consideran los ejemplos bien aprendidos (con valor de error bajo) de la misma forma que aquellos que aun no han sido aprendidos (valores altos de error), perdiendo tiempo de entrenamiento innecesariamente. Sin embargo no se puede restringir el proceso de entrenamiento únicamente al subconjunto de ejemplos con valores altos de error: debido a que como BP minimiza la función de error localmente, todos los demás ejemplos son gradualmente

"olvidados" si un ejemplo simple es repetido hasta su mínimo error. El objetivo del presente trabajo es explorar el espacio entre este extremo y su opuesto, la estrategia uniforme, y determinar qué posibles mejoras se lograrían utilizando estrategias de selección de patrones sofisticadas.

Utilizaremos el conjunto de entrenamiento denominado *Iris Data* (ver apéndice), para tener una visión lo más amplia posible.

2. Estrategias

2.1. Probabilidad de presentación dependiente del error (PPDE)

Una de las vías más simples de adaptar la selección de patrones al proceso de aprendizaje es directamente tomar la probabilidad de presentación de un ejemplo proporcional a su error de salida. Los ejemplos aun son presentados en orden aleatorio, pero no con una probabilidad de presentación uniforme. La probabilidad de presentación p^u del ejemplo u es proporcional a su error . Esto permite una frecuencia de selección mayor para los ejemplos que producen mayor error (Cachin, 1994).

Durante el entrenamiento, las probabilidades p^u deben ser actualizadas periódicamente debido a que la eficiencia de la red y los pesos cambian en cada iteración. Esto requiere presentar todos los ejemplos a la red al menos una vez, lo que puede causar sobrecarga, si se realiza con frecuencia. Los experimentos muestran que la longitud del intervalo entre actualizaciones consecutivas de las probabilidades debe ser del orden del número de ejemplos de entrenamiento.

Veamos ahora el algoritmo formalizado:

Sea N el número de ejemplos de entrenamiento, p^u la probabilidad de presentación de un ejemplo al entrenamiento, $u = 1,...,N$. Inicialmente, $p^u = 0$ para $1 \le u \le N$.

1. Realizar para cada ejemplo u un ciclo completo de BP y calcular su probabilidad de presentación proporcional a alguna de las siguientes funciones:
   - $p^u \propto \varepsilon^u$
   - $p^u \propto (\varepsilon^u)^2$
   - $p^u \propto (\varepsilon^u)^4$
   - $p^u \propto e^{\varepsilon^u}$
   - $p^u \propto 10^{\varepsilon^u}$

   Si la red converge, terminar; si no, continuar al paso 2.
2. Realizar N ciclos completos de BP, seleccionando en cada uno de ellos un ejemplo u aleatoriamente de acuerdo con su probabilidad .
3. Retornar al paso 1.

2.2. Repetición dependiente del error (RDE)

Esta variante presenta los ejemplos proporcionalmente a su error, pero a diferencia de PPDE, los patrones no son seleccionados aleatoriamente, sino con base en un esquema determinista. El procedimiento sería el siguiente: presentar inicialmente cada ejemplo u a la red y guardar su error ε^u, así como el ejemplo $u_{\max}$ con el valor mayor de error $\varepsilon_{\max} = \max \varepsilon^u$. Los patrones son repetidos siguiendo este esquema: el conjunto de ejemplos es examinado W veces, seleccionando en el i-ésimo examen todos los ejemplos u para los cuales $\varepsilon^u > i\varepsilon_{\max}/W$ hasta el W examen. El número de veces que un patrón es presentado depende de su error de salida. Después se realiza otra iniciación con todos los ejemplos y se repite el procedimiento. Los valores de error ε^u son ajustados siempre que un patrón es seleccionado para el entrenamiento. Una vez que su error está por debajo de cierto criterio, no es presentado hasta la próxima inicialización. Los experimentos muestran que W debe ser del orden del número de ejemplos de entrenamiento (Cachin, 1994).

El algoritmo quedaría de la siguiente forma: Sea N el número de ejemplos de entrenamiento.

1. Realizar para cada ejemplo u un ciclo completo BP y guardar sus respectivos errores ε^u, así como $\varepsilon_{max} = \max \varepsilon^u$. Si la red converge terminar; si no, continuar al paso 2.

2. Repetir N veces el siguiente procedimiento:

   Seleccionar los ejemplos u en la i-ésima iteración que cumplan $\varepsilon^u > i\varepsilon_{max}/N$, presentarlos al entrenamiento y actualizar su error ε^u.

3. Retornar al paso 1.

2.3. Sistema de fichero de tarjetas (SFT)

La idea de esta estrategia parte del método usado por algunos estudiantes para aprender gran cantidad de diferentes aspectos o preguntas. Ellos escriben cada aspecto en tarjetas separadas y las colocan en un fichero de tarjetas con diferentes índices. Para el aprendizaje, seleccionan todas las tarjetas bajo un determinado índice y tratan de responder en el primer intento; las tarjetas son regresadas al fichero bajo otros índices. Los índices mayores son seleccionados menos frecuentemente que los menores donde están las preguntas más difíciles (Cachin, 1994).

Podemos resumirlo de la siguiente forma. Sean $f_1 > f_2 > ... > f_k$ los enteros representantes de las frecuencias de presentación relativas a k índices. La selección de los ejemplos puede ser realizada de dos formas:

- **Determinista:** recorriendo todos los índices, donde i es repetido f_i veces. En la repetición de cada índice los patrones son seleccionados de acuerdo con una estrategia uniforme. Después los ejemplos son movidos a otros índices.

- **Aleatoria:** seleccionando un ejemplo de índice i con una probabilidad proporcional a f_i. Como cualquier estrategia aleatoria, las probabilidades deben ser actualizadas periódicamente.

Más formalmente tendríamos:

Sean $f_1 > f_2 > .... > f_k$ los enteros representantes de la frecuencia de presentación relativa a k índices en que se subdividen los N ejemplos de entrenamiento. La selección de los ejemplos puede seguir una de las variantes siguientes:

(a) Repetir k veces el siguiente procedimiento:

- Seleccionar en la i-ésima iteración todos los ejemplos de índice i y repetirlos f_i veces.

- Si la red converge terminar; si no, mover los ejemplos a otros índices.

(b) Repetir k veces el siguiente procedimiento:

- Seleccionar aleatoriamente un índice i con probabilidad proporcional a f_i y repetir los ejemplos que pertenecen al índice i, f_i veces.

- Si la red converge terminar; si no, recalcular los valores de f_i y repetir el proceso.

Los mejores resultados fueron logrados con pequeños valores de k (k = 3) y frecuencias de presentación 16:4:1. La variante determinista es preferible debido a que necesita menos tiempo y recursos.

2.4. Partición del conjunto de entrenamiento (PCE)

En esta estrategia el conjunto de N ejemplos de entrenamiento es dividido en una partición de M elementos $S_1, S_2, ..., S_M$. El entrenamiento comienza con los ejemplos de S_1 únicamente, luego se adiciona S_2 después de algún tiempo, y así sucesivamente. Dentro del subconjunto propiamente los ejemplos son seleccionados de acuerdo con una estrategia uniforme (Cachin, 1994). Existen dos variantes a la hora de adicionar un nuevo subconjunto S_i:

1. S_i es entrenado por algún tiempo exclusivamente y luego se continúa con $S_1 \cup S_2 \cup ... \cup S_i$

2. Continuar el entrenamiento inmediatamente con $S_1 \cup S_2 \cup ... \cup S_i$.

Más formalmente se tendría:

1. Dividir el conjunto de entrenamiento de N ejemplos en M subconjuntos S_1, S_2, ..., S_M.
2. Repetir M veces el siguiente procedimiento:

   En la i-ésima iteración entrenar k veces con $S_1 \cup S_2 \cup ... \cup S_i$ o entrenar primero exclusivamente S_i, k/2 veces y luego continuar con $S_1 \cup S_2 \cup ... \cup S_i$ otras k/2 veces.
3. Si la red converge terminar; si no regresar al paso 2.

Los mejores resultados fueron logrados con valores bajos de M (M = 4) y k = 500.

2.5. Repetir hasta aprender (RHA)

En esta estrategia los ejemplos son seleccionados aleatoriamente y repetidos hasta que su error esté por debajo de cierto criterio β. Los ejemplos no aprendidos son repetidos inmediatamente (Munro, 1992). Se selecciona en este caso un valor fijo para β que hace a la estrategia muy dependiente de la tarea específica de selección de este parámetro. Para superar este problema, Cachin (1994) propone adaptar β de acuerdo con el estado del proceso de entrenamiento. El valor de β puede ser $d \cdot \overline{\varepsilon^2}$, una constante d veces el valor del error cuadrado medio para todos los ejemplos. Para d, valores mayores que 1 pueden ser más útiles, por lo que en los experimentos se usa d = 1,5, lo que significa que solo los ejemplos con valores de error sustancialmente mayores que la media son repetidos.

El algoritmo sería:

1. Seleccionar un ejemplo u aleatoriamente.
2. Repetir su presentación hasta que $\varepsilon^u < \beta$.
3. Si la red converge terminar; si no, regresar al paso 1.

La sobrecarga causada por la evaluación repetida del promedio del error no puede evitarse.

2.6. Repetición dependiente del mayor error (RDME)

La idea central de esta estrategia es presentar al entrenamiento siempre el ejemplo que produzca mayor error. En este caso los patrones no son seleccionados aleatoriamente por lo que sería una estrategia de tipo determinista. Es necesaria la introducción de una estructura adicional dónde mantener ordenados los ejemplos de entrenamiento o simplemente implementar un proceso de búsqueda que permita en cada iteración seleccionar el ejemplo con el mayor nivel de error.

Inicialmente se realiza un ciclo BP completo para cada ejemplo u del conjunto de entrenamiento lo que permitiría determinar para cada uno de ellos su error. Posteriormente se realizan M ciclos BP, presentando en cada uno de ellos el ejemplo que tenga mayor error $\varepsilon_{\max} = \max \varepsilon^u$. Después el proceso de inicialización es repetido para garantizar que todos los ejemplos sean presentados al entrenamiento por lo menos cada vez que se realiza dicha inicialización.

Formalmente se tiene:

Sean N ejemplos de entrenamiento.

1. Realizar un ciclo completo BP para cada ejemplo u y almacenar su error ε^u . Si la red converge terminar; si no continuar al paso 2.
2. Repetir M veces los siguientes pasos:
   - Seleccionar el ejemplo u tal que $\varepsilon^u = \varepsilon_{\max}$.
   - Realizar un ciclo completo BP con u y actualizar su error .
3. Regresar al paso 1.

Los experimentos muestran que la complejidad adicionada no es muy significativa debido fundamentalmente a que el tamaño del conjunto de entrenamiento usualmente no es muy grande, no obstante se recomienda optimizar al máximo el proceso de selección del ejemplo con mayor error, así como un valor para M del orden del número de ejemplos de entrenamiento.

2.7. Sistema de fichero de tarjetas dependiente del error (SFTDE)

Esta estrategia puede considerarse una variante de la propuesta por Cachin (1994), con la novedad de que la asignación de los k índices en que se divide el conjunto de entrenamiento está en correspondencia con el nivel de error que produce cada ejemplo y a su ubicación en el intervalo $[\varepsilon_{\min}, \varepsilon_{\max}]$. A diferencia del método original que asigna los índices de forma aleatoria. Otra novedad es que no se considera la variante aleatoria del algoritmo inicial, tomando en cuenta la superioridad de la determinista.

Podemos resumir la estrategia de la siguiente forma: Sean $f_1 < f_2 < \ldots < f_k$ los enteros representantes de la frecuencia de presentación relativa de k índices. Inicialmente se realiza un ciclo hacia adelante de BP para cada ejemplo con el objetivo de determinar a qué índice pertenece de acuerdo con $u \in \mathit{Índice}(i)$ si $(i-1)\alpha + \varepsilon_{\min} \le \varepsilon^u \le i\alpha + \varepsilon_{\min}$, siendo $\alpha = (\varepsilon_{\max} - \varepsilon_{\min})/k, \quad (1 \le i \le k)$.

Luego para i = 1, ..., k, presentar al entrenamiento todos los ejemplos u que pertenecen al índice i, f_i veces, siempre que cumpla $(i-1)\alpha + \varepsilon_{\min} \le \varepsilon^u \le i\alpha + \varepsilon_{\min}$. Los ejemplos que no cumplan esta condición son ignorados temporalmente, hasta que se realice otra inicialización.

El algoritmo formal sería:

Sean N ejemplos de entrenamiento y $f_1 < f_2 < \ldots < f_k$ los enteros representantes de la frecuencia de presentación relativa a los k índices en que se divide el conjunto de entrenamiento.

1. Asignarle a cada ejemplo u un índice i de la siguiente forma:
   - Realizar un ciclo hacia adelante para cada ejemplo u y determinar su error ε^u, así como $\varepsilon_{\min}$ y $\varepsilon_{\max}$. Si la red converge terminar; si no, continuar.
   - Determinar el índice i al que pertenece cada ejemplo u de forma que $u \in \mathit{Índice}(i)$ si $(i-1)\alpha + \varepsilon_{\min} \le \varepsilon^u \le i\alpha + \varepsilon_{\min}$ siendo, $\alpha = (\varepsilon_{\max} - \varepsilon_{\min})/k \quad (1 \le i \le k)$.
2. Para i = 1, ..., k, repetir f_i veces todos los ejemplos que pertenecen al índice i, si cumple que $(i-1)\alpha + \varepsilon_{\min} \le \varepsilon^u \le i\alpha + \varepsilon_{\min}$.

Al igual que en el algoritmo original se recomienda valores pequeños para k (k = 3) y frecuencias 16:4:1.

3. Resultados

Para el entrenamiento de redes neuronales de tipo MLP utilizando BP como algoritmo de aprendizaje las dos cantidades de interés general que podemos medir son la *velocidad* y *exactitud* del proceso de aprendizaje. La velocidad es medida como el porcentaje de memorización o aprendizaje de ejemplos correctos en algún intervalo de tiempo predefinido. Por otra parte la exactitud es medida como la capacidad de generalización de la red evaluando el porcentaje de ejemplos correctos para el conjunto de prueba de la misma forma que para el conjunto de entrenamiento.

Los experimentos fueron realizados con el conjunto de datos *IRIS DATA*, reconocido por la eficiencia en el entrenamiento de sistemas de este tipo. Para cada versión de una estrategia se realizaron 100 ciclos de entrenamiento cada uno compuesto por 10^4 ó 10^5 presentaciones.

La tabla 1 muestra los resultados para las estrategias PPDE y RDE, las cuales difieren considerablemente, a pesar de su similitud. Únicamente la mejor variante de PPDE sobrepasa la eficiencia de la estrategia uniforme y las demás poseen resultados peores. Por el contrario la variante RDE sobrepasa la estrategia uniforme logrando los mejores resultados con la función de error proporcional a $(\varepsilon^u)^2$. Sin embargo la estrategia RDME demostró ser la mejor, incluyendo al resto de las estrategias que también se muestran en la tabla 2, en la cual para los dos prime-

ros problemas solo RHA y SFT muestran resultados superiores a la estrategia uniforme. En caso del SFT muestra una evidente superioridad con respecto al resto de las estrategias.

En la segunda tabla también puede comprobarse que para el caso de las estrategias PCE y SFT aleatorio, éstas no muestran mejoras significativas con respecto a la estrategia uniforme.

Los experimentos muestran que una estrategia de selección cuidadosa para los patrones puede resultar en un significativo aumento de la eficiencia del proceso de aprendizaje. Esto confirma nuestra motivación inicial para la selección pedagógica de patrones para el entrenamiento. Ejemplos con pequeño error pueden ser excluidos del entrenamiento parcialmente, evitando desperdiciar tiempo en ellos y centrando la atención en aquellos que producen niveles altos de error.

Tabla 1 Resultados para las estrategias PPDE y RDE

Estrategia	*IRIS DATA*
Uniforme	82,3
PPDE usando:	
ε^u	65,9
$(\varepsilon^u)^2$	78,1
$(\varepsilon^u)^4$	70,4
e^{ε}	85,8
10^{ε}	80,2
RDE usando:	
ε^u	91,3
$(\varepsilon^u)^2$	96,5
$(\varepsilon^u)^4$	95,4
e^{ε}	93,4
10^{ε}	92,3
RDME	96,9

La tabla 1 muestra los porcentajes de clasificación correcta de los ejemplos después de 10^4, 10^5 y 10^4. Aunque mostró mejores resultados con respecto a las demás estrategias analizadas.

Tabla 2 Resultados para las estrategias SFT, PCE, RHA y SFTDE

Estrategia	*IRIS DATA*
Uniforme	82,3
SFT Determinista:	
k = 2	85,7
k = 3	86,5
k = 3	90,6
SFT Aleatorio:	
k = 2	80,2
k = 3	80,9
k = 3	81,7
PCE(a)	83,5
PCE(b)	83,6
RHA	93,4
SFTDE:	
k = 2	90,4
k = 3	91,3
k = 3	94,8

La tabla 2 muestra los porcentajes de clasificación correcta después de 10^4 presentaciones. Los mejores resultados son logrados por las estrategias RHA, SFT determinista y SFTDE.

Sin embargo, solo las estrategias deterministas muestran este efecto. Las estrategias probabilísticas no alcanzan resultados similares y algunos casos funcionan peor que la estrategia uniforme. De las estrategias de selección de patrones vistas, RDE, RDME y SFTDE, que repiten ejemplos proporcionales al error que producen, son las que mejor se han comportado. Esto brinda un buen balance entre el énfasis en los patrones no aprendidos y la reelección de los ya aprendidos como lo confirman los resultados de los tres problemas de prueba.

4. Conclusiones

Se ha presentado aquí un aspecto fundamental del entrenamiento de las redes neuronales de tipo

MLP utilizando BP como algoritmo de aprendizaje: la estrategia de selección de ejemplos al entrenamiento. Pocas investigaciones han sido llevadas a cabo en esta área anteriormente, por lo que no se cuenta aún con las herramientas teóricas necesarias para su tratamiento y constituye por tanto un interesante campo de investigación. Nuestras comparaciones han mostrado que las estrategias pedagógicas de selección de patrones pueden lograr mejores resultados que las uniformes. En el mejor de los casos, el tiempo de entrenamiento puede ser reducido por un factor de dos y la memorización es cercana a la mitad.

Las estrategias satisfactorias fueron todas las de tipo determinista. Siguiendo estas ideas puede verse la fascinante analogía entre la pedagogía de redes neuronales y la humana, lo que puede presuponer la construcción de otros métodos.

Otro aspecto que se debe señalar es que la naturaleza de la aplicación no parece tener incidencia en la eficiencia de estas estrategias, por lo que resultaría beneficiosa su aplicación sobre todo en aquellos problemas donde la eficiencia de la red se ve muy comprometida debido a las características del problema en sí.

Bibliografía

1. Cachin, C., "Pedagogical Pattern Selection Strategies", In: *Neural Networks*, Vol. 7, No. 1, 1994, pp. 175-181.
2. Looney, C. G., "Stabilization and speedup of convergence in training feedforward neural networks", In: *Neurocomputing*, Vol. 10, No. 1, 1996.
3. Munro, P. W., *Repeat until bored: A pattern selection strategy*. San Mateo, CA: Morgan Kaufmann Publishers, 1992.
4. Sarkar, D., "Method to speed up error back-propagation learning algorithm", *ACM Computing Surveys*, Vol. 27, No. 4, 1995.
5. Setiono R. and Liu H, "Improving backpropagation with feature selection", *Applied Intelligence*,. Vol. 6, No. 2, 1996.
6. Van Ooyen, A. y Nienhuis B, "Improving the convergence of back-propagation algorithm", In: *Neural Networks*, Vol. 5, No. 3, 1992.
7. Zurada, J. M., *Introduction to Artificial Neural Systems*, West Publishing Company, 1992.

Apéndice

Conjunto de entrenamiento Iris Data: posee tres clases con 50 ejemplos cada una y cuatro rasgos por ejemplo.

Clase No. 1				*Clase No. 2*				*Clase No. 3*			
5	3,5	1,4	0,2	7	3,2	4,7	1,4	6,3	3.3	6	2,5
4,9	3	1,4	0,2	6,4	3,2	4,5	1,5	5,8	2,7	5,1	1,9
4,7	3,2	1,3	0,2	6,9	3,1	4,9	1,5	7,1	3	5,9	2,1
4,6	3,1	1,5	0,2	5,56	2,3	4	1,3	6,3	2,9	5,6	1,8
5	3,6	1,4	0,2	6,5	2,8	4,6	1,5	6,5	3	5,8	2,4
5,4	3,9	1,7	0,4	5,7	2,8	4,5	1,3	7,6	3	6,6	2,1
4,6	3,4	1,4	0,3	6,3	3,3	4,7	1,6	4,9	2,5	4,5	1,7
5	3,4	1,5	0,2	4,9	2,4	3,3	1	7,3	2,9	6,3	1,8
4,4	2,9	1,4	0,2	6,6	2,9	4,6	1,3	6,7	2,5	5,8	1,8
4,9	3,1	1,5	0,1	5,2	2,7	3,9	1,4	7,2	3,6	6,1	2,5
5,4	3,7	1,5	0,2	5	2	3,5	1	6,5	3,2	5,1	2
4,8	3,4	1,6	0,2	5,9	3	4,2	1,5	6,4	2,7	5,3	1,9
4,8	3	1,4	0,1	6	2,2	4	1	6,8	3	5,5	2,1

Continuación apéndice

Clase No. 1				*Clase No. 2*				*Clase No. 3*			
4,3	3	1,1	0,1	6,1	2,9	4,7	1,4	5,7	2,5	5	2
5,8	4	1,2	0,2	5,6	2,9	3,6	1,3	5,8	2,8	5,1	2,4
5,7	4,4	1,5	0,4	6,7	3,1	4,4	1,4	6,4	3,2	5,3	2,3
5,4	3,9	1,3	0,4	5,6	3	4,5	1,5	6,5	3	5,5	1,8
5,1	3,5	1,4	0,3	5,8	2,7	4,1	1	7,7	3,8	6,7	2,2
5,7	3,8	1,7	0,3	6,2	2,2	4,5	1,5	7,7	2,6	6,9	2,3
5,1	3,8	1,5	0,3	5,6	2,5	3,9	1,1	6	2,2	5	1,5
5,4	3,4	1,7	0,2	5,9	3,2	4,8	1,8	6,9	3,2	5,7	2,3
5,1	3,7	1,5	0,4	6,1	2,8	4	1,3	5,6	2,8	4,9	2
4,6	3,6	1	0,2	6,3	2,5	4,9	1,5	7,7	2,8	6,7	2
5,1	3,3	1,7	0,5	6,1	2,8	4,7	1,2	6,3	2,7	4,9	1,8
4,8	3,4	1,9	0,2	6,4	2,9	4,3	1,3	6,7	3,3	5,7	2,1
5	3	1,6	0,2	6,6	3	4,4	1,4	7,2	3,2	6	1,8
5	3,4	1,6	0,4	6,8	2,8	4,8	1,4	6,2	2,8	4,8	1,8
5,2	3,5	1,5	0,2	6,7	3	5	1,7	6,1	3	4,9	1,8
5,2	3,4	1,4	0,2	6	2,9	4,5	1,5	6,4	2,8	5,6	2,1
4,7	3,2	1,6	0,2	5,7	2,6	3,5	1	7,2	3	5,8	1,6
4,8	3,1	1,6	0,2	5,5	2,4	3,8	1,1	7,4	2,8	6,1	1,9
5,4	3,4	1,5	0,4	5,5	2,4	3,7	1	7,9	3,8	6,4	2
5,2	4,1	1,5	0,1	5,8	2,7	3,9	1,2	6,4	2,8	5,6	2,2
5,5	4,2	1,4	0,2	6	2,7	5,1	1,6	6,3	2,8	5,1	1,5
4,9	3,1	1,5	0,1	5,4	3	4,5	1,5	6,1	2,6	5,6	1,4
5	3,2	1,2	0,2	6	3,4	4,5	1,6	7,7	3	6,1	2,3
5,5	3,5	1,3	0,2	6,7	3,1	4,7	1,5	6,3	3,4	5,6	2,4
4,9	3,1	1,5	0,1	6,3	2,3	4,4	1,3	6,4	3,1	5,5	1,8
4,4	3	1,3	0,2	5,6	3	4,1	1,3	6	3	4,8	1,8
5,1	3,4	1,5	0,2	5,5	2,5	4	1,3	6,9	3,1	5,4	2,1
5	3,5	1,3	0,3	5,5	2,6	4,4	1,2	6,7	3,1	5,6	2,4
4,5	2,3	1,3	0,3	6,1	3	4,6	1,4	6,9	3,1	5,1	2,3
4,4	3,2	1,3	0,2	5,8	2,6	4	1,2	5,8	2,7	5,1	1,9
5	3,5	1,6	0,6	5	2,3	3,3	1	6,8	3,2	5,9	2,3
5,1	3,8	1,9	0,4	5,6	2,7	4,2	1,3	6,7	3,3	5,7	2,5
4,8	3	1,4	0,3	5,7	3	4,2	1,2	6,7	3	5,2	2,3
5,1	3,8	1,6	0,2	5,7	2,9	4,2	1,3	6,3	2,5	5	1,9
4,6	3,2	1,4	0,2	6,2	2,9	4,3	1,3	6,5	3	5,2	2
5,3	3,7	1,5	0,2	5,1	2,5	3	1,1	6,2	3,4	5,4	2,3
5	3,3	1,4	0,2	5,7	2,8	4,1	1,3	5,9	3	5,1	1,8