Generalización en Redes Coulombianas

J.A.HORAS*, P.M.PASINETTI, E.A.BEA

UNIVERSIDAD NACIONAL DE SAN LUIS. FACULTAD DE CIENCIAS FÍSICO-MATEMÁTICAS Y NATURALES.
DEPARTAMENTO DE FÍSICA. INSTITUTO DE MATEMÁTICA APLICADA SAN LUIS.
EJÉRCITO DE LOS ANDES 950 - (5700) SAN LUIS - ARGENTINA.
e-mail: jhoras@unsl.edu.ar

Resumen

Se combina el uso del algoritmo de aprendizaje de la red coulombiana para determinar las conexiones entre neurodos con la aplicación de un proceso constructivo para generar la primer capa. Esta es construida simultáneamente con el proceso de aprendizaje y es tal que sus representaciones internas son linealmente separables.
Se analiza la capacidad de la red así construida para dar la respuesta correcta a patrones no presentados previamente. Se estudia la habilidad de generalización, analizando curvas típicas en problemas de clasificación (simetría).

Abstract

We combine the learning algorithm of the coulombian network to determine the connection weights between neurodes with the application of a constructive process to generate the first layer. It is constructed simultaneously with the learning process in that way that the internal representations are linearly separable.
We study the network ability to respond correctly to patterns never presented before, analizing curves in typical classification problems.

Introducción

En el trabajo sobre aprendizaje [1] se mostró que es posible resolver problemas simples (XOR) mediante el entrenamiento *independiente* de cada una de las capas de neurodos que forman la red, observándose que la red coulombiana puede realizar en la primer capa una transformación de los patrones de entrada de forma tal que sus representaciones internas quedan linealmente separables. Esto último hace atractiva a la red coulombiana puesto que si las representaciones internas son linealmente separables, cualquier problema de clasificación puede ser resuelto alimentando con ellas una segunda capa. El atractivo es aún mayor si esto se logra mediante el entrenamiento independiente de cada capa, siendo innecesario entonces cualquier tipo de propagación de errores entre ellas.

En este trabajo se muestra, en primer lugar, un procedimiento constructivo para generar la primer capa el cual asegura que sus salidas (representaciones internas) son linealmente separables. En segundo lugar, se describe la implementación del algoritmo de aprendizaje de la red coulombiana que se utiliza para determinar las conexiones entre neurodos, estudiándose la habilidad de genralización en problemas simples de clasificación.

Eliminar la restricción de una arquitectura fija, construyendo secuencialmente la red durante el aprendizaje, permite resolver en principio cualquier problema de clasificación [2]. La forma mas simple es colocar neurodos que responden solamente a la presencia de un patrón particular (estos son llamados neurodos "grandmother" [2]). Desafortunadamente, esta solución es equivalente a una "look-up table" y no capta ninguna de las *correlaciones* entre los patrones presentados. Por el contrario, una red

* Investigador del CONICET.

neuronal debe ser capaz de clasificar correctamente al menos algunos de los patrones nunca presentados a ella, de forma tal que haya aprendido y no solo memorizado. Esta propiedad de *generalización* es más pronunciada para redes mínimas (una red con un número mínimo de neurodos y conexiones).

El objetivo de encontrar la arquitectura absolutamente mínima es NP-completo [3], en consecuencia proponemos una heurística para construir redes que se aproximan a la mínima, pero cuyos tiempos de aprendizaje son aceptables. Para ello combinamos la utilización de un procedimiento constructivo para la primer capa y el uso del algoritmo de aprendizaje de la red coulombiana.

El algoritmo constructivo

Como en el trabajo anterior [1], tenemos M sitios de memoria $\mathbf{x}_1,\mathbf{x}_2,...,\mathbf{x}_M$. Nos interesan las representaciones internas (salidas de la primer capa) que tiene cada uno de ellos. Para asegurarnos que la segunda capa es alimentada con patrones linealmente separables, usamos el siguiente procedimiento secuencial para construir la primer capa:

a) Encontramos un conjunto de pesos $\mathbf{W}_1$ (obtenidos con el algoritmo de aprendizaje de la red coulombiana) tales que agrupen por un lado un número $n_1 \geq 1$ de patrones de una clase y por el otro al resto de los patrones de esta clase "mezclados" con todos los de la otra. Esto nos da el primer neurodo de la primer capa. Para el entrenamiento se utilizan los M patrones, llamado el *conjunto de trabajo*. Nótese que esto es siempre posible dada la existencia de la solución del neurodo "Grandmother".

b) Luego se extraen los n_1 patrones del grupo puro y tenemos un conjunto de $M-n_1$ patrones que es ahora el nuevo conjunto de trabajo, con el que obtenemos los pesos $\mathbf{W}_2$ (para el segundo neurodo) que clasifican correctamente $n_2 \geq 1$ patrones. Este procedimiento, ilustrado en la figura 1, se continua hasta que el conjunto de trabajo consista solo de patrones de la misma clase.

La función original ha sido mapeada entonces en una función definida por las representaciones internas, pudiendo probarse estrictamente que hay un conjunto de pesos que las hacen linealmente separables. Mostramos que mediante el uso del algoritmo de aprendizaje de la red coulombiana es posible obtener tal separabilidad con una red relativamente mínima.

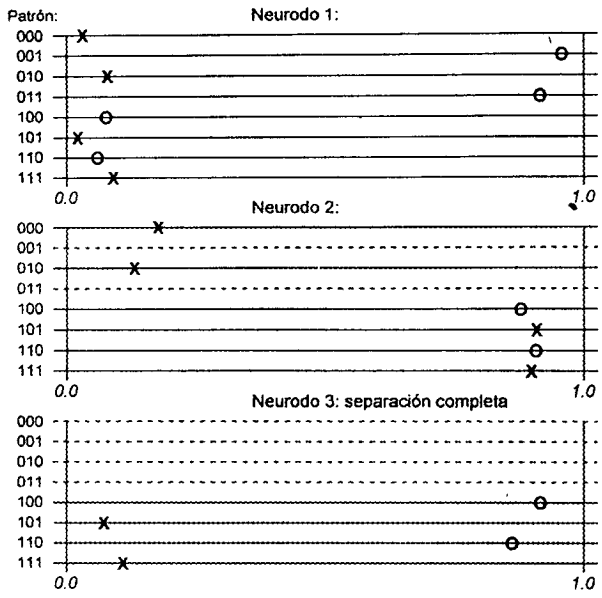

Figura 1: ilustración del procedimiento constructivo para el caso de 3 bits, en donde las cruces representan patrones simétricos y los círculos no simétricos (ver texto).

Simulaciones

Se analizan resultados obtenidos para generalización, o sea la capacidad de la red entrenada solamente en partes de una función, para dar la salida correcta a patrones nunca presentados. Estudiamos la velocidad y calidad de convergencia mediante el análisis de curvas típicas de generalización (número de errores en el conjunto test versus número de ejemplos en el conjunto de entrenamiento) para distintos valores de energía tanto en la primera como en la segunda capa, para estudiar la eventual presentación del fenómeno de sobre-entrenamiento (over-training).

El ejemplo utilizado es el de la función simetría [4] en que la salida debe ser *uno* si los bits de entrada son simétricos respecto de su centro y *cero* si no lo son. Se muestran aquí resultados para redes pequeñas con 8 y 10 bits de entrada.

La arquitectura de la red utilizada y las características del entrenamiento fueron las siguientes:

1) Dos capas. Se indica el número de neurodos en la primer capa (tabla 1), habiendo solo uno en la segunda.

2) Actualización de pesos tipo batch.

3) Los parámetros utilizados fueron:

- El exponente L=1, la "temperatura" T=1, inicialización de los pesos al azar entre -0.05 y 0.05, el parámetro de aprendizaje η=0.005 para la primer capa y η=0.003 para la segunda y cota mínima para el número de épocas por ensayo C_m=80 para la primer capa y C_m=40 para la segunda.

4) Otras condiciones:

- Para evitar divergencias en la actualización de los pesos, se procedió de igual forma que en la otra presentación [1].

- Relación de la magnitud de "carga eléctrica" de los sitios de memoria de distinta clase proporcional al cociente entre el número de patrones de cada clase.

- En la construcción de la primer capa los ensayos no exitosos no se contabilizan, mientras que en la segunda capa son tenidos en cuenta sumando C_m al intento exitoso subsecuente.

- En la actualización de pesos se utilizó momento con β=0.9.

Los resultados se muestran en la tabla 1 y en las figuras 2, 3 y 4 para los casos de 3 neurodos en la primer capa y 8 bits de entrada. En la figura 2 se grafica el número de errores normalizados a 1 en el conjunto de test, versus el número de ejemplos utilizados para entrenar. Cada punto es el promedio de 8 ensayos, graficándose también la dispersión de cada uno de ellos. Para ilustrar y mostrar la calidad de la convergencia, se muestran en las figuras 3 y 4 el gráfico de las energías de cada capa versus el número de épocas tomándose los pesos correspondientes a varios puntos, según se muestra en la figura 2. Como prueba adicional de convergencia se hicieron ensayos para el caso de 10 bits de entrada, o sea 1024 patrones de los cuales 32 son simétricos. Los resultados para generalización fueron: error de 6.92×10^{-2}, 6.32×10^{-2}, 1.05×10^{-2}, 6.85×10^{-3}, 1.98×10^{-3}, 0.0, 0.0 para un número de patrones usados para entrenamiento de 200, 280, 360, 440, 520, 680, 760 respectivamente, siendo el número de patrones del conjunto de test lo que resta en cada caso del total de 1024 patrones. Esto último corresponde a 500 épocas de entrenamiento en la primer capa (η=0.001) y 130 épocas en la segunda (η=0.0005).

NÚMERO DE PATRONES DE TRAINING.	NÚMERO DE NEURODOS.
50	2.3
70	3.2
90	3.3
110	3.1
130	3
150	3.7
170	3.3
190	3.3

Tabla 1: número de patrones usados en el entrenamiento y el correspondiente número de neurodos en la primer capa (promedio sobre 10 redes).

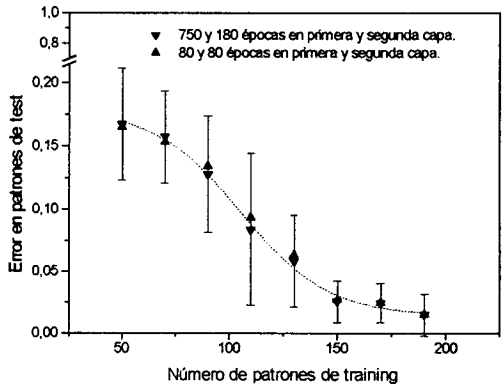

Figura 2: curva de generalización (ver texto). La curva que une los puntos es solo una guía para el ojo.

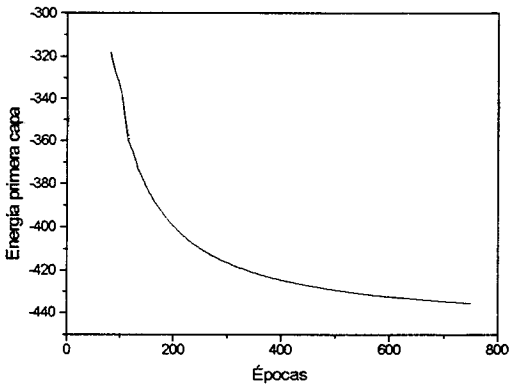

Figura 3: gráfica de la "energía" versus el número de épocas, en la primer capa

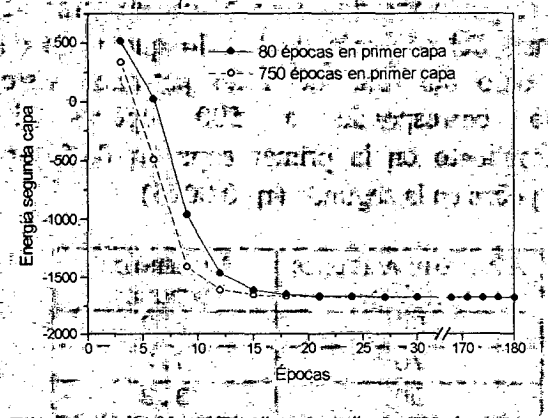

Figura 4: gráfica de la "energía" versus el número de épocas, en la segunda capa.

Conclusión

En este trabajo se han mostrado resultados interesantes y auspiciosos de la combinación de una metodología constructiva y del algoritmo de aprendizaje de la red coulombiana. En particular enfatizamos su capacidad de generalización, no detectándose en las pruebas realizadas, una degradación de la performance de generalización al aumentar el tiempo de entrenamiento medido en épocas en ambas capas.

Referencias

1 - J. Horas, E. Bea y M. Pasinetti, Anales de la AFA 1994.
2 - M. Marchand, M. Golea y P. Ruján, Europhys. Lett., **11** (6), pp. 487-492 (1990).
3 - A. Blum y R. L. Rivest, *Proceedings of the First Worshop on Computational Learning Theory* (Morgan Kaufmann) 1988, p. 9.
4 - D. E. Rumelhart y J. L. McClelland, *Parallel Distributed Processing*, Vol. 1-2 (Bradford Books, MIT Press, Cambridge, MA) 1986.