Redes Neuronales

Gonzalo V. Castiglione, Alan E. Karpovsky, Martín Sturla
Estudiantes Instituto Tecnológico de Buenos Aires (ITBA)

19 de Abril de 2012

Entrega Preliminar 2 - Informe

***Resumen*—El presente informe busca analizar redes neuronales multicapa con aprendizaje supervisado que resuelvan el problema del *Simetría* y *Paridad* lógica para N bits con $2 \leq N \leq 5$ a través del uso de tres variantes de funciones de transferencia.**

***Palabras clave*—perceptrón, función de transferencia, red neuronal, aprendizaje supervisado, conjunto de entrenamiento, conjunto de testeo**

I. Introducción

Se analizó el comportamiento de distintas redes neuronales multicapa para el problema de la paridad y simetría booleana. Con este fin se implementó un algoritmo que permite definir la arquitectura y las propiedades de la red neuronal a generar de manera de hacer más simple y práctico su estudio. En las secciones próximas se explicará el desarrollo y modelado del problema como así también los resultados obtenidos.

II. Desarrollo

A. Modelado del problema

Se representó la red neuronal como una **matriz de pesos**. Cada neurona es una columna de pesos, cada capa de neuronas es una matriz de pesos, la red neuronal, por consiguiente, es un vector de matrices (en particular un perceptrón simple es solo una matriz). Lo interesante es que hallar el valor de la red neuronal con un cierto input se reduce a multiplicar el vector entrada por cada una de estas matrices. Cabe destacar que al vector de entrada se le agrega siempre un -1 como último valor para el sesgo. A su vez esta implementación facilita el manejo de errores.

Para las unidades escalón el error está definido como la **Distancia Hamming Levenshtein**, para los otros tipos de unidades se decidió utilizar el **error cuadrático medio**.

B. Features

Se implementaron tres tipos de funciones para modificar la variable η *(learn rate)*: La primera de ellas es a valor **constante**, , la segunda es ***annealed*** que reduce η exponencialmente y por último se tiene un *learning rate* **adaptative** que modifica η en función de los últimos errores obtenidos. El crecimiento es aritmético, con cota en $\eta = 0{,}5$, la reducción exponencial.

Debido a la existencia de mínimos locales se desarrolló un algoritmo ***persistent search*** que lo que hace es, dado un cierto error, entrenar la red neuronal la cantidad de épocas especificada y verificar si el error obtenido es mayor al deseado. En cuyo caso reinicia el algoritmo, asigna los pesos al azar y vuelve a comenzar el entrenamiento.

Otra estrategia que se utilizó para subsanar este problema es la de darle un impulso a η al detectar que el algoritmo está estancado en un mínimo local.

III. Resultados

A modo de ejemplo, en la sección ***Anexo A: Gráficos*** se muestran tres gráficos de los resultados obtenidos: En la *Figura 1* se visualiza un estancamiento en un mínimo local debido al uso de unidades lineales, en la *Figura 2* se puede aprenciar cómo un η dinámico puede, en algunos casos, ayudar a salir del mínimo local y por último en la *Figura 3* se propone un ejemplo que ilustra cómo el η varía a medida que el error decrece consitentemente.

IV. Conclusión

En base a las pruebas podemos concluír que utilizando una función de transferencia *lineal*, la red no aprenderá el problema. Esto se debe a que cada capa realiza una transformación lineal de la salida de la capa anterior y una transformacion lineal de una transformación lineal es lineal. Es decir que no existen coeficientes $a_1, a_2, \cdots, a_n$ que resuelvan, por ejemplo, mi ecuación de paridad para n bits, lo que hará que se tienda a un mínimo local (en 0) o diverger en una secuencia alterna por lo que el error subirá indefinidamente.

A su vez, para las funciones escalón o no lineales se observó que es posible que ciertas arquitecturas tiendan a atascarse mas en minimos locales u otras quizas resuelvan el problema.

Por lo visto en clase, para un XOR de N bits, la arquitectura óptima resulta ser una única capa oculta con N neuronas. Asimismo para el problema de simetría se sugiere utilizar una única capa oculta con 2 neuronas, independientemente de la cantidad de bits que contenga la entrada.

19 de Abril, 2012.

Anexo A: Gráficos

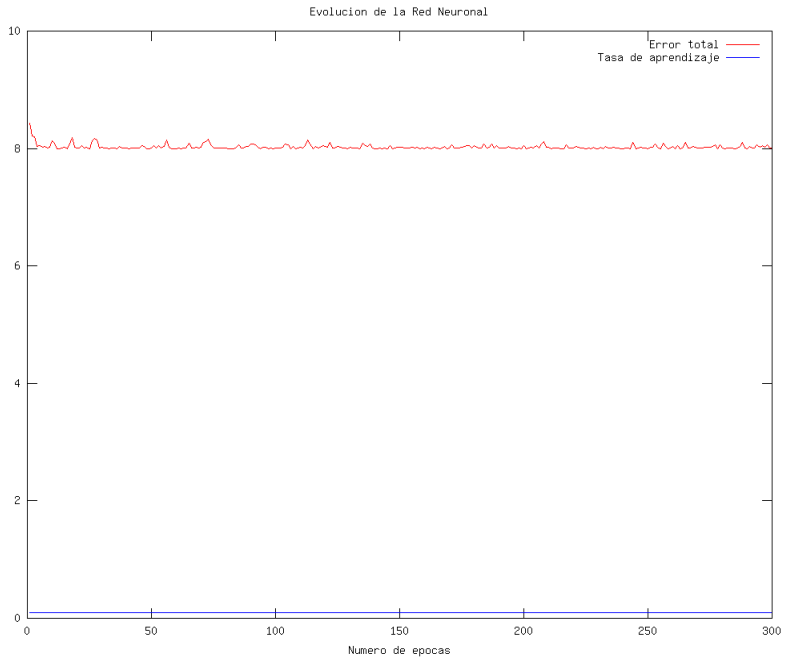

Figura 1: Ejemplo de estancamiento en mínimo local debido a la utilización de una función de transferencia lineal.

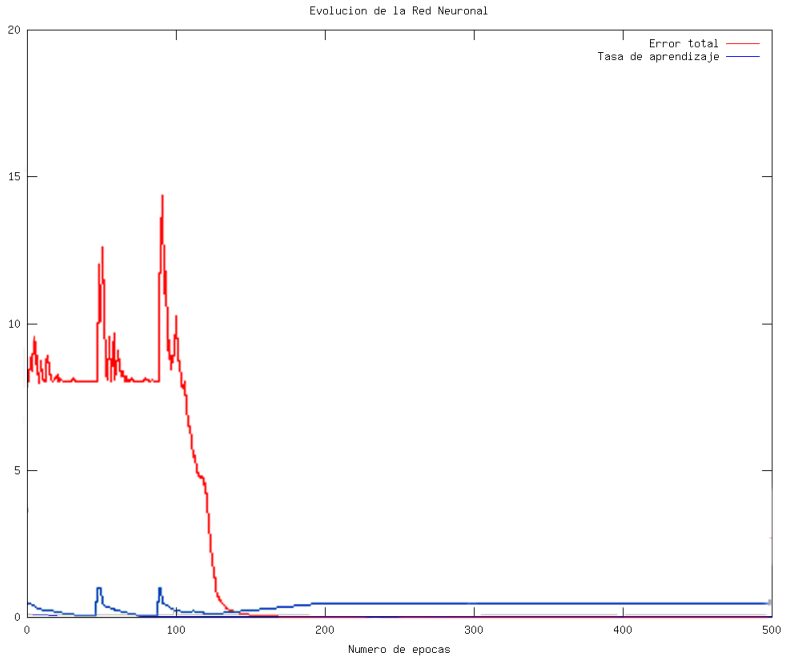

Figura 2: Escape de mínimo local utilizando eta dinámico para el problema de paridad de 4 bits usando 500 épocas y una arquitectura de una capa oculta con 4 neuronas.

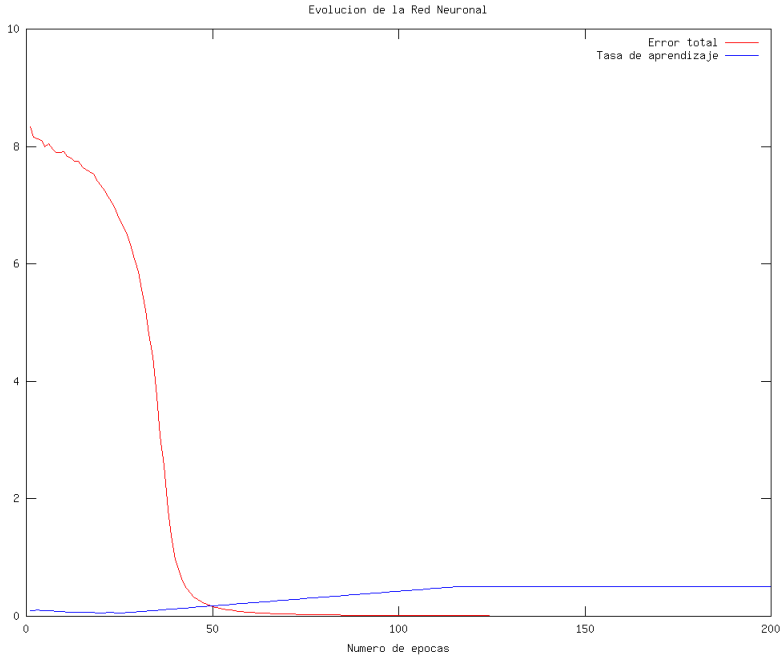

Figura 3: Muestra cómo se incrementa hasta el valor máximo eta cuando la red neuronal resuelve el problema con el error deseado y todavía quedan épocas.