Sistemas de inteligencia artificial

Informe preliminar n°3

Red neuronal con aprendizaje supervisado

Autores:
Pablo Ballesty - 49359
Nicolás Magni - 48008
Guillermo Liss - 49282

3 de mayo de 2012

Resumen

El objetivo del presente documento es detallar el diseño e implementación de diferentes mejoras aplicadas al aprendizaje supervisado de una red neuronal multicapa, para que la misma logre aprender y realizar generalizaciones aceptables de una función desconocida, de la cual se conocen solo un conjunto de puntos. También se realizan diferentes pruebas de arquitecturas con el fin de encontrar la configuración óptima.

1. Problema

El problema consiste en aproximar una función escalar $f : \Re^2 \to \Re$ de la cual se conocen 441 puntos. En la figura 1 se presenta el ploteado de los puntos conocidos. Dicha aproximación, se realiza mediante una red neuronal multicapa con dos funciones de activación diferentes.

2. Desarrollo

2.1. Funciones de activación

A continuación se definen las funciones de activación que se utilizan, junto con sus derivadas.

- **sigmoid-exp:**

$$g(x) = \frac{1}{1 + \exp^{-x}} \tag{1}$$

$$g'(x) = g(x)[1 - g(x)] \tag{2}$$

- **sigmoid-tanh:**

$$g(x) = tanh(x) \tag{3}$$

$$g'(x) = sech(x)^2 \tag{4}$$

En las figuras 2 y 3 se encuentran los gráficos de las mismas.

2.2. Normalización de la entrada y la salida

En la figura 2 se puede ver que las entradas x_1 y x_2 se encuentran en el rango $(-3, 4)$ respectivamente. Viendo que la función parece ser simétrica respecto al plano $x_2 = 0$, para normalizar las entradas consideramos que estas están en el rango $(-4, 4)$, de esta forma para aprovechar la sección sigmoidea de la función **sigmoid-exp** se divide la entrada recibida por 0.67, luego el rango $(-4, 4)$ se transforma a $(-5.97, 5.97)$. Para **sigmoid-tanh** no es necesaria ninguna modificación.

Con respecto a la salida, en el caso de utilizar **sigmoid-exp**, esta va a estar en el rango $(0, 1)$, y en el caso de utilizar **sigmoid-tanh** esta va estar comprendida en el rango $(-1, 1)$. Como se puede ver en la figura 1, podemos considerar que el rango de la salida de la función f es $(-10, 10)$, por lo cual, es necesaria la normalización de la salida de la red en ambos casos. Por lo tanto, cuando se utiliza **sigmoid-exp** la salida se normaliza mediante la ecuación 5, en cambio, cuando se utiliza **sigmoid-tanh** la salida se normaliza utilizando la ecuación 6.

$$n_{exp}(x) = (x - 0.5) * 20 \tag{5}$$

$$n_{tanh}(x) = x * 10 \tag{6}$$

2.3. Entrenamiento y mejoras implementadas

El entrenamiento de la red se llevó a cabo mediante el algoritmo de *backpropagation*, realizando la correción de pesos de forma iterativa.

En las siguientes secciones se explican las mejoras implementadas y detalles de implementación que caben la pena destacar.

2.3.1. η adaptativo

Una de las mejoras que se implementó, es la de η adaptativo. Dependiendo de la diferencia entre el error cuadrático de la última época corrida, con el error cuadrático de la anterior, se realizan pequeños cambios sobre η. Si esta diferencia es positiva se disminuye el valor de η en un porcentaje b, y si ocurren diez diferecias consecutivas negativas se aumenta el valor de η en un valor constate a. El fin de estos cambios es que cuando el error aumenta, se realicen cambios pequeños sobre los pesos con el objeto de que este deje de aumentar, mientras que cuando el error disminuye se busca realizar cambios más grandes con el objetivo de acelerar el aprendizaje.

2.3.2. Momentum

Otra de las mejoras que se implementó, es la que se conoce como *momentum*. La mejora consiste en sumar un término más cuando se realiza la corrección de pesos. Este término va ser la diferencia del último peso con el penúltimo, multiplicada por un factor, denominado *factor momentum*, el cual se utiliza para graduar la importancia del término en la corrección. Esta mejora permite, acelerar el aprendizaje en los casos que los pesos anteriores hayan tenido grandes cambios, y sirve también para evitar caer en mínimos locales.

2.3.3. Conjuntos de prueba y entrenamiento

Para la selección de los conjuntos de prueba y entrenamiento, se dividió a los puntos conocidos de la función f, en dos conjuntos, los cuales denominamos *puntos de interés* y *puntos de poco interés*. La división se lleva a cabo de la siguiente forma, si el valor absoluto del punto es mayor a un umbral, al cual llamamos *umbral de interés*, el punto es colocado en el conjunto de *puntos de interés*, de lo contrario es colocado en el conjunto de *puntos de poco interés*.
Para conformar los conjuntos de prueba y entrenamiento, se tomaron diferentes porcentajes de estos dos conjuntos. Tomando mayor porcentaje de *puntos de interés* sobre *puntos de poco interés* para el conjunto de entrenamiento, y dejando el resto de los puntos para el conjunto de prueba.

El objetivo de esta selección de puntos, es tener más cantidad de puntos en las secciones más extremas de la función donde resulta más costoso que la red aprenda.
En la figura 4 pueden verse los dos conjuntos para un umbral de 4.

2.3.4. Condición de corte

Para finalizar el entrenamiento de una red, se utilizan dos condiciones de corte diferentes. Una mediante la fijación de un error cuadrático medio de entrenamiento a alcanzar, y otra por un máximo de épocas corridas. El corte se realiza si el error cuadrático medio del conjunto de entrenamiento alcanza al error fijado, o se han corrido tantas épocas como las máximas especificadas.

Durante la ejecución del entrenamiento, se imprimen cada 10 épocas, los errores cuadráticos medios para el conjunto de entrenamiento y el conjunto de prueba. El error cuadrático medio para el conjunto de prueba nos da la idea de cuanto afectan esas épocas, en puntos que no forman parte del conjunto de entrenamiento, es decir, nos da la idea de cómo generaliza la red. En las pruebas realizadas no hubo momentos en los cuales el error de prueba tienda a subir, lo que significaría que la red pierde generalización para memorizar el conjunto de entrenamiento, por lo tanto, no se implementó una condición de corte basada en este parámetro.

3. Resultados y conclusiones

Las arquitecturas que se utilizaron para las pruebas fueron[1]:

- *Arquitectura pequeña*: [2 6 2 1]
- *Arquitectura mediana*: [2 12 12 1]
- *Arquitectura grande*: [2 36 12 6 1].

[1] Las arquitecturas que se proponen fueron aquellas que arrojaron resultados aceptables durante pruebas realizadas en la implementación de las mejoras.

El conjunto de entrenamiento va a estar formado por el 95 % de los *puntos de interés* y el 70 % de los *puntos de poco interés*, esto es, 338 puntos. Por lo tanto, el conjunto de prueba queda con 103 puntos.

En el cuadro 1 se presentan los resultados obtenidos de entrenamientos con diferentes valores para a, b y *factor momentum*, el error cuadrático mínimo que se pide es 0.01, y la cantidad máxima de épocas es de 500. La función de activación utilizada en **sigmoid-tanh**. El objetivo de esta prueba, es obtener una combinación de parámetros conveniente, para luego escoger las arquitecturas que hayan obtenido el mejor comportamiento, y realizar entrenamientos más intensivos.

Se puede ver en los resultados, que la utilización de un *factor momentum* alto, no fue de gran utilidad en ninguna de las arquitecturas. Mientras que, los parámetros de η adaptativo que mejor resultaron fueron $a = 0.1$ y $b = 0.01$.

Se realizaron distintas pruebas utilizando **sigmoid-exp**, pero no se obtuvieron resultados para destacar.

La arquitectura que mejor se comportó en las pruebas fue la *arquitectura grande*, es por esto, que se realizó un entrenamiento intensivo que consta en correr el entrenamiento con los siguientes parámetros $a = 0.1$, $b = 0.01$, *factor momentum* $= 0.2$, cantidad máxima de épocas $= 5000$ y un error cuadrático mínimo aceptable de 0.017 (que se deriva de aceptar un error de 0.01 en cada punto de los 338 puntos).
El entrenamiento alcanzó un error de 0.0715993, cortando por cantidad máxima de épocas. En la figura 5, se encuentra una comparación de los valores que arroja la red, contra los valores de los puntos recibidos. En la figura 6 se realiza el ploteado de los puntos recibidos junto con los arrojados por la red.

A. Cuadros y Figuras

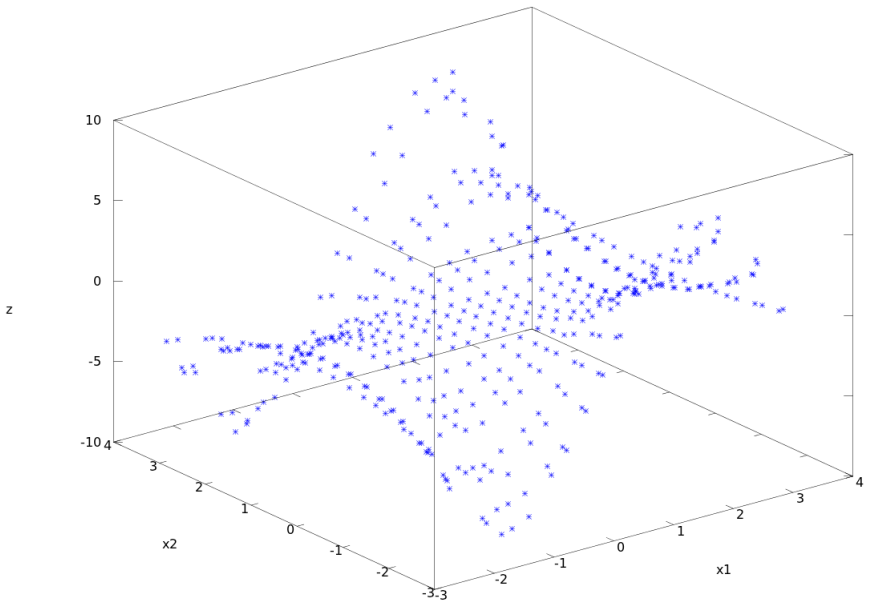

Figura 1: puntos conocidos de la función escalar $f : \Re^2 \to \Re$.

Arquitectura	η	a	b	factor momentum	Error de entrenamiento	Época
pequeña	0.5	0.1	0.01	0.2	0.170463	500
mediana	0.5	0.1	0.01	0.2	0.388060	500
grande	0.5	0.1	0.01	0.2	0.103301	500
pequeña	0.5	0.1	0.01	0.5	0.396113	500
mediana	0.5	0.1	0.01	0.5	3.32919	500
grande	0.5	0.1	0.01	0.5	0.104419	500
pequeña	0.5	0.025	0.0025	0.7	13.1008	500
mediana	0.5	0.025	0.0025	0.7	110.037	500
grande	0.5	0.025	0.0025	0.7	45.6437	500

Cuadro 1: resultados obtenidos para diferentes parámetros de las mejoras.

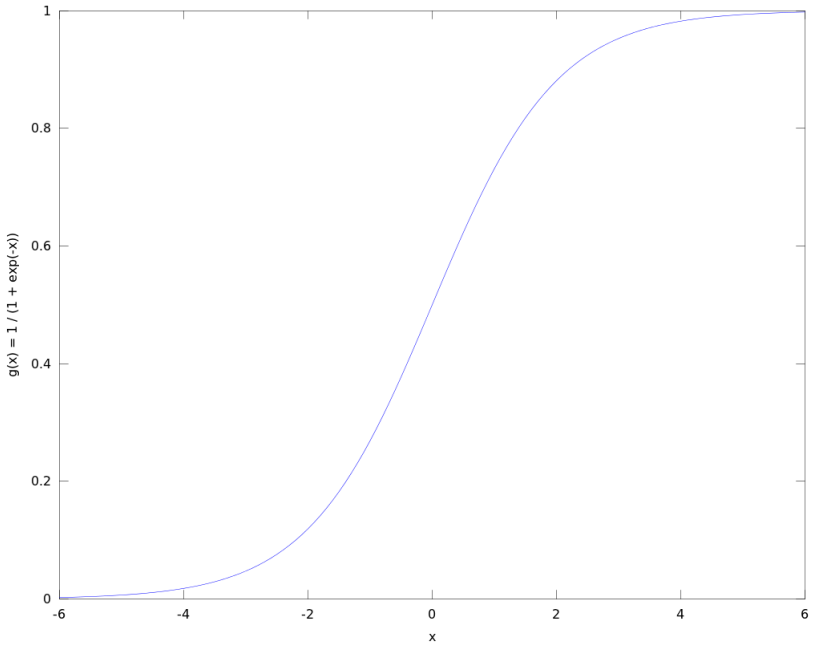

Figura 2: gráfico de **sigmoid-exp**.

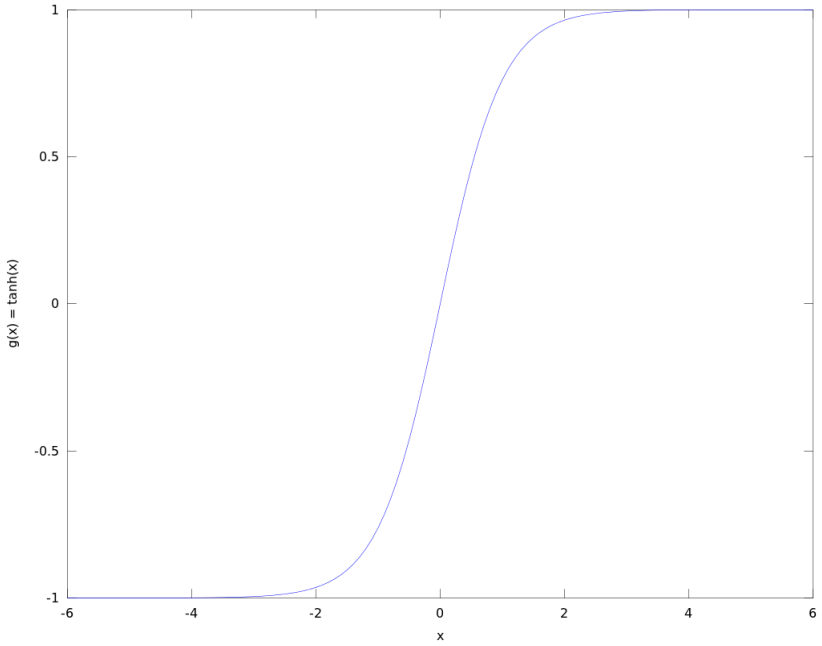

Figura 3: gráfico de **sigmoid-tanh**.

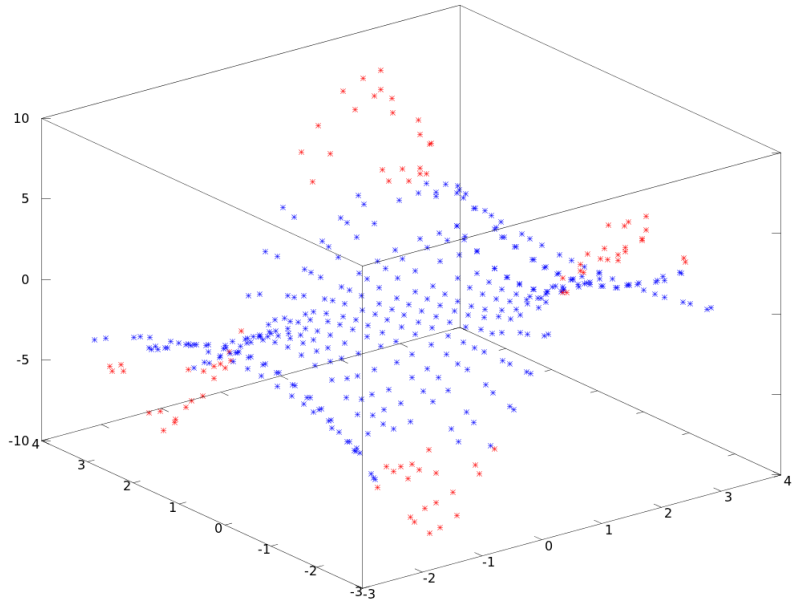

Figura 4: los puntos en rojo conforman el conjunto de *puntos de interés*, con un *umbral de interés* = 4.

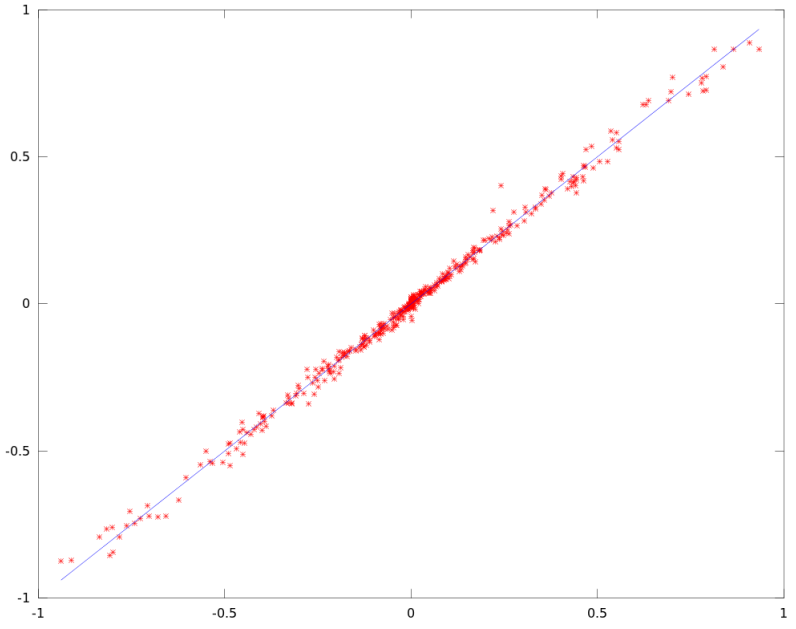

Figura 5: comparación de los valores de la red con los recibidos. En el eje x se coloca la salida esperada, en el eje y la salida de la red. La recta azul representaría un entrenamiento perfecto.

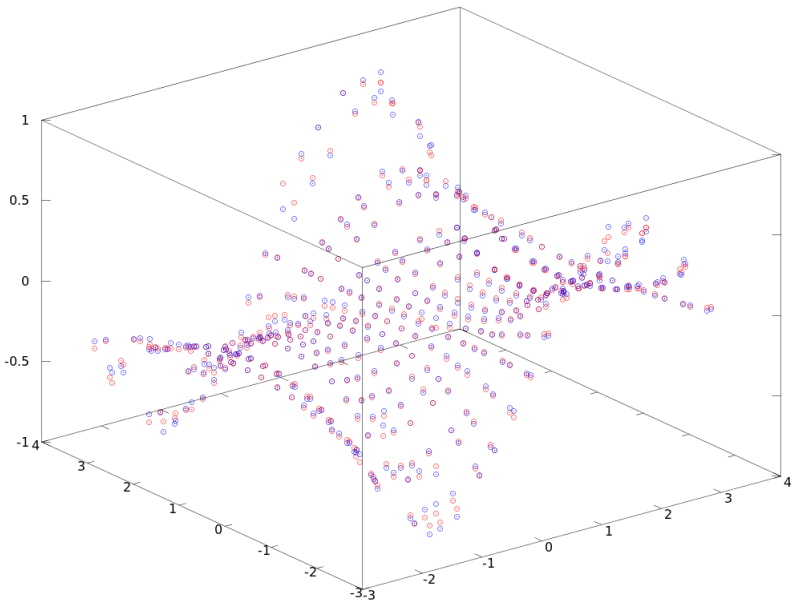

Figura 6: Comparación de los valores de la red con los recibidos. Los puntos azules son los recibidos, los rojos son los arrojados por la red.