Perceptrón, su entrenamiento y el backpropagation

perceptron, its training and backpropagation

Autores:Juan Manuel Restrepo Urrego
Stiven Valencia Ramírez
IS&C, Universidad Tecnológica de Pereira, Pereira, Colombia
Correo-e: juanmanuel.restrepo@utp.edu.co
stiven.valencia1@utp.edu.co

***Resumen*— Este documento presenta una breve pero detallada información sobre lo que es, el cómo funciona y un ejemplo de lo que viene siendo el perceptrón, su entrenamiento y el backpropagation, también nos enfocaremos de forma detallada en el ejemplo de lógica difusa dando codigo para mayor entendimiento de este.**
Se presentará de forma detallada cómo funciona el backpropagation de manera formal, como también se explicará de forma aplicativa.
Recordar que todo esto se plantea sobre la base de red neuronal y sus aplicaciones como algoritmos para trabajarla.

***Palabras clave*— backpropagation, perceptrón, lógica difusa, código, entrenamiento, algoritmo, neurona.**

Abstract- This paper presents a brief but detailed information about what it is, how it works and an example of what is the perceptron, its training and backpropagation, we will also focus in detail on the example of fuzzy logic giving code for a better understanding of it.
It will be presented in detail how the backpropagation works, in a formal way, it is worth the redundancy, as well as it will be explained in an applicative way.
Remember that all this is raised on the basis of neural networks and its applications as algorithms to work it.

Keywords--- backpropagation, perceptron, fuzzy logic, code, training, algorithm, neuron.

I. INTRODUCCIÓN

El perceptrón surge en una de las ramas de la inteligencia artificial, la cual se le conoce como conexionista. Esta hace referencia a un mundo que pretende realizar modelos muy aproximados a las actividades de las neuronas biológicas, siendo el perceptrón la forma más básica de una neurona, el cual se puede conectar con muchas más y así formar un sistema más complejo y de mayor rendimiento como lo es el perceptrón multicapa.

I.1 PERCEPTRÓN

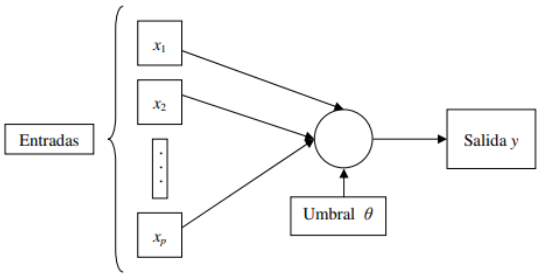

Figura 1. Perceptrón multicapa

Consideraciones básicas

Una neurona consiste en una implementación lineal de un combinador seguido de un limitador el cual nos arroja una salida la cual puede ser negativa o positiva, visualmente sería de la siguiente manera.

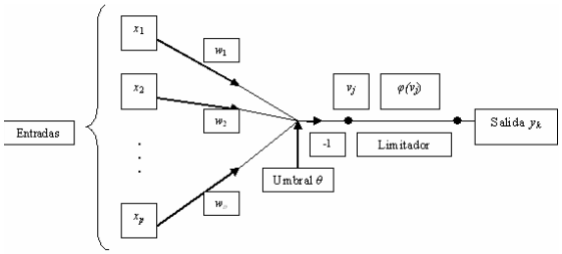

Figura 2. Implementación lineal de un combinador

La salida de este combinador lineal está determinada por una ecuación la cual es

$$v = \sum_{i=1}^{p} w_i x_i - \theta$$

Funcionalidad

Su funcionamiento es realmente simple pues la ecuación anterior nos indica que la salida y, está determinada por una sumatoria de 1 hasta p entradas de la multiplicación de los pesos wi por las entradas xi, eso menos el umbral teta. El propósito de este es el de clasificar los resultados en dos tipos, si son positivos o negativos (1 o -1)

Ahora parece ser que suene todo esto realmente difícil pero realmente no lo es y será mucho más sencillo de explicar con un ejemplo sobre el backpropagation.
entonces qué es el backpropagation

I.2 ENTRENAMIENTO

En qué consiste el entrenamiento muy sencillo, el entrenamiento del perceptrón consiste en alimentarlo con múltiples muestras de entrenamiento y el cálculo de la salida para cada uno de ellos. Después de cada muestra, los pesos w se ajustan de tal manera a fin de minimizar el error de salida, definido como la diferencia entre la salida deseada (objetivo) y la real. Hay otras funciones de error, como el error cuadrado medio, pero el principio básico de entrenamiento sigue siendo el mismo.[1]

tiene un gran inconveniente: sólo puede aprender funciones linealmente separables. ¿Qué tan importante es este inconveniente? Toma XOR, una función relativamente simple, y notarás que no puede ser clasificada por un separador lineal (nota el intento fallido, a continuación):

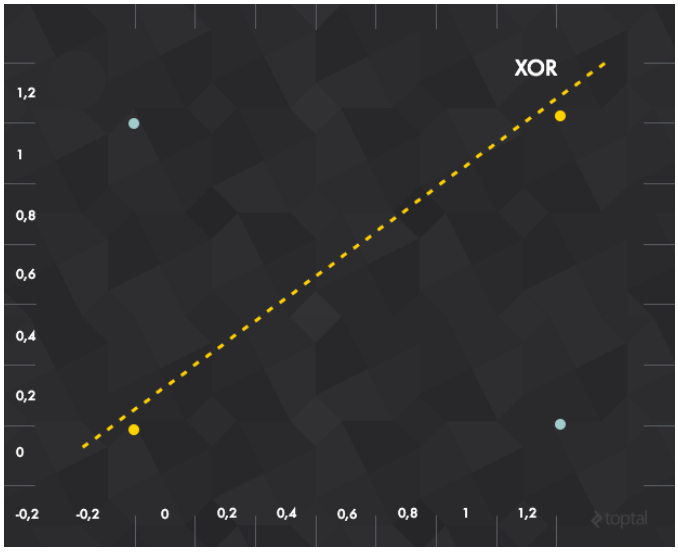

Figura 3. Compuerta XOR

La solución para este problema es realmente sencilla, consta de agregar otra neurona para así tener otra función lineal separable. De esta manera podríamos acotar los puntos que deseamos y separar ambas poblaciones. Esto es el indicio de un sistema más complejo conocido como redes neuronales.

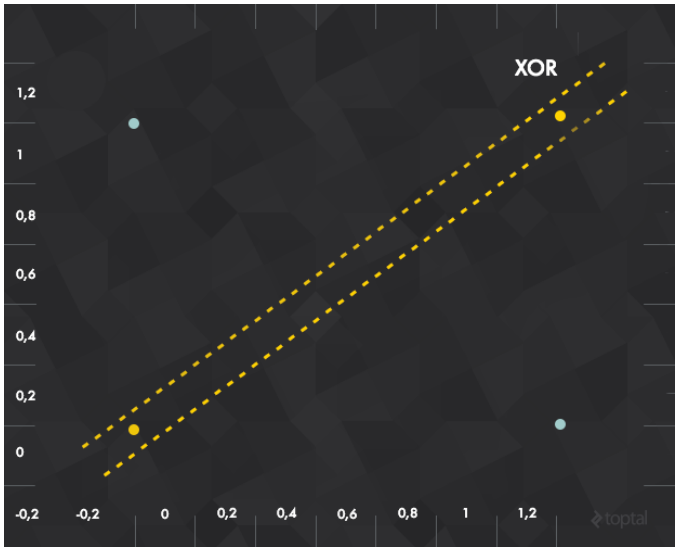

Figura 4. Solución Compuerta XOR

I.3 BACKPROPAGATION

La propagación hacia atrás de errores o retropropagación (del inglés backpropagation) es un método de cálculo del gradiente utilizado en algoritmos de aprendizaje supervisado utilizados para entrenar redes neuronales artificiales. El método emplea un ciclo propagación – adaptación de dos fases. Una vez que se ha aplicado un patrón a la entrada de la red como estímulo, este se propaga desde la primera capa a través de las capas siguientes de la red, hasta generar una salida. La señal de

salida se compara con la salida deseada y se calcula una señal de error para cada una de las salidas.

Las salidas de error se propagan hacia atrás, partiendo de la capa de salida, hacia todas las neuronas de la capa oculta que contribuyen directamente a la salida. Sin embargo las neuronas de la capa oculta sólo reciben una fracción de la señal total del error, basándose aproximadamente en la contribución relativa que haya aportado cada neurona a la salida original. Este proceso se repite, capa por capa, hasta que todas las neuronas de la red hayan recibido una señal de error que describa su contribución relativa al error total.

¿Cómo funciona?

El backpropagation, abreviatura de "propagación hacia atrás de errores", es un algoritmo para el aprendizaje supervisado de redes neuronales artificiales mediante el descenso de gradientes . Dada una red neuronal artificial y una función de error , el método calcula el gradiente de la función de error con respecto a los pesos de la red neuronal. Es una generalización de la regla delta para perceptrones a redes neuronales de alimentación directa multicapa.[2]

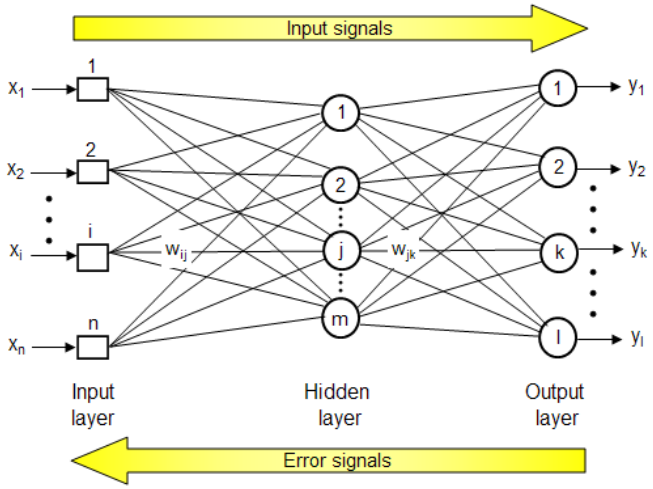

Figura 5. Backpropagation

REFERENCIAS

Referencias en la Web:

[1] https://www.toptal.com/machine-learning/un-tutorial-de-aprendizaje-profundo-de-perceptrones-a-redes-profundas

[2] https://brilliant.org/wiki/backpropagation/