

# Contents

<b>1</b>	<b>Introducción</b>	<b>1</b>
<b>2</b>	<b>Neurona Artificial</b>	<b>3</b>
2.1	Redes Neuronales Artificiales . . . . .	8
<b>3</b>	<b>Caso de Estudio</b>	<b>10</b>
3.1	Problema . . . . .	10
3.2	Justificacion . . . . .	11
3.3	Propuestas de Solucion . . . . .	12
3.3.1	Topologia . . . . .	12
3.3.2	Reglas . . . . .	12
3.3.2.1	Regla de propagación . . . . .	12
3.3.2.2	Regla de Activacion . . . . .	12
3.3.2.3	Regla de Salida . . . . .	12
3.3.2.4	Regla de Aprendizaje . . . . .	12
3.3.2.4.1	Back Propagation . . . . .	12
3.3.2.4.2	Segundo Orden . . . . .	13
3.3.2.4.3	R-PROP . . . . .	13
3.4	Desarrollo de la solucion . . . . .	13
3.4.1	Herramientas . . . . .	14
3.4.1.1	Python . . . . .	14
3.4.1.2	PyCharm . . . . .	15
3.4.1.3	Package . . . . .	15
3.4.2	Implementacion . . . . .	16
3.4.2.1	Back Propagation . . . . .	16
3.4.2.2	Segundo Orden . . . . .	16
3.4.2.3	R-PROP . . . . .	16
3.5	Resultados . . . . .	16
3.6	Conclusiones . . . . .	16

## 1 Introducción

El cerebro humano es un sistema de cálculo muy complejo, puede llevar a cabo procesamiento que a primera vista parecen sencillos, como por ejemplo,

el reconocimiento de imágenes. Esta capacidad que tiene el cerebro humano para pensar, recordar y resolver problemas ha inspirado a muchos científicos a intentar imitar estos funcionamientos.

Los intentos de crear un ordenador que sea capaz de emular estas capacidades ha dado como resultado la aparición de las llamadas Redes Neuronales Artificiales o Computación Neuronal.

El principal objetivo del Reconocimiento de patrones es la clasificación ya sea supervisada o no supervisada. Aplicaciones como Data Mining, Web Searching, recuperación de datos multimedia, reconocimiento de rostros, reconocimientos de caracteres escritos a mano, etc., requieren de técnicas de reconocimiento de patrones robustas y eficientes. Las redes neuronales, por su capacidad de generalización de la información disponible y su tolerancia al ruido, constituyen una herramienta muy útil en la resolución de este tipo de problemas.[1]

Este documento muestra los conceptos básicos de las Redes Neuronales y su regla de aprendizaje, en particular la configuración en *Perceptrón Multicapa* y el varios algoritmos de aprendizaje (Propagación hacia atrás, Métodos de segundo orden, RPROP, Algoritmos Genéticos).

## 2 Neurona Artificial

Uno de los retos más importantes a los que se enfrenta el ser humano de nuestra generación es el de la construcción de sistemas inteligentes, en su afán de conseguir este propósito aparecen las redes neuronales artificiales. Desde el punto de vista biológico las RNA son un modelo matemático acerca del funcionamiento del cerebro. *”Los sencillos elementos de cálculo aritmético equivalen a las neuronas-células que procesan la información en el cerebro- y la red en general equivale a un conjunto de neuronas conectadas entre sí”* [2]

Para la raza humana sigue siendo un misterio el funcionamiento del cerebro humano y como se genera el pensamiento, sin embargo años y años de investigación han dado ideas sobre el accionar del mismo. Si se quieren reproducir las acciones del cerebro humano, se debe tener la idea de como funciona. Una explicación sencilla y clara se encuentra en [2]:

*”Sabemos que la neurona, o célula nerviosa, es la unidad funcional básica de los tejidos del sistema nervioso, incluido el cerebro. Las neuronas están formadas por el cuerpo de la célula, o soma, en donde se aloja el núcleo de la célula. Del cuerpo de la célula salen ramificaciones de diversas fibras conocidas como dendritas y sale también una fibra más larga denominada axón. Las dendritas se ramifican tejiendo una tupida red alrededor de la célula, mientras el axón se extiende un buen tramo: por lo general, un centímetro (100 veces el diámetro del cuerpo de la célula) y, en casos extremos, hasta un metro. Finalmente, el axón también se ramifica en filamentos y subfilamentos mediante los que establece conexión con las dendritas y los cuerpos de las células de otras neuronas. A la unión o conexión se le conoce como sinapsis. Cada neurona establece sinapsis desde con una docena de otras neuronas hasta con cientos de miles de otras de ellas”*

La neurona artificial se ha diseñado como una abstracción de la neurona biológica y se muestra en la Figura 1. La figura representa la neurona  $j$  que recibe entradas. Sus partes son:

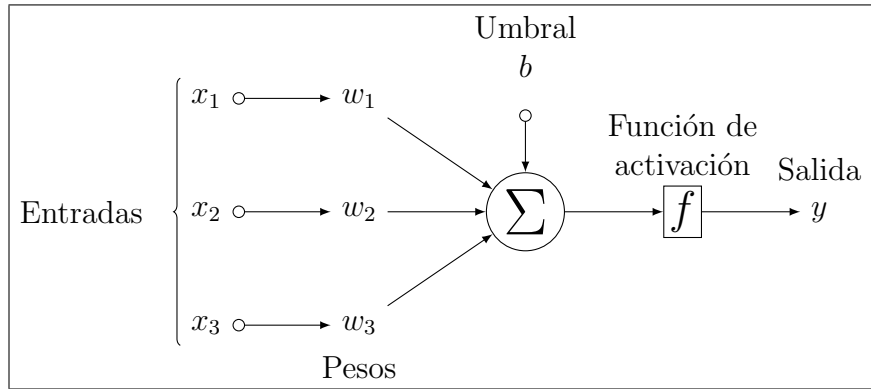


Fig. 1: Representación de una Red Neuronal Artificial.

1. Las **entradas**  $x_i$ , que son puntos por los que se reciben los datos provenientes del entorno o bien de otras neuronas. Para este caso se consideran  $n$  entradas, siendo el valor de  $n = 3$

$$X = (x_1, x_2, x_3)$$

2. La salida  $y_i$ . En una neurona biológica corresponde al axón
3. Al igual que en una neurona biológica, la neurona artificial debe permitir establecer conexiones (sinápsis) entre las entradas (dendritas) de una neurona y la salida (axón) de otra. Esta conexión representa con una línea que tiene asociado un valor llamado **peso sináptico**  $w_{ji}$ . Nótese que el primer subíndice indica la neurona a la que llega la conexión, mientras que el segundo subíndice indica de donde viene la conexión. El peso representa el factor de importancia de la conexión en la determinación del valor de salida. El valor  $w_{ji}$ , que es un número real, se modifica durante el entrenamiento de la red neuronal y es la variable que almacenará la información que indicará que la red ha aprendido algo y por tanto que sirva para un propósito u otro.
4. En la Figura 1 también se observa una entrada especial, llamada umbral, con un valor fijo que puede ser -1 o 1, y con un peso asociado llamado  $w_0$ . El valor del umbral se ajusta igual que cualquier otro peso durante el proceso de entrenamiento.
5. Una **regla de propagación**. Para cierto valor de las entradas  $x_i$  y los pesos sinápticos asociados  $w_{ji}$ , se realiza algún tipo de operación

para obtener el valor del potencial *post-sináptico* . Este valor es función de las entradas y los pesos. Una de las operaciones mas comunes es realizar la suma ponderada, que no es otra cosa que la sumatoria de las entradas, pero teniendo en cuenta la importancia de cada una (el peso sináptico asociado). Luego:

$$u = \sum_i^{n=3} w_{ji}x_i + w_0b$$

6. Una **función de activación**. Luego de realizar la suma ponderada, se aplica al resultado la función de activación, que se escoge de tal manera que permita obtener la forma deseada para el valor de la salida.

$$\begin{aligned} y &= f(u) \\ &= f\left(\sum_i^{n=3} w_{ji}x_i + w_0b\right) \\ &= f(W.X) \\ &= f(W^T X) \end{aligned} \tag{1}$$

donde las últimas dos ecuaciones están en notación vectorial. Es necesario especificar la función de activación  $f$ . Las funciones más usuales se observan en la 8

Con estas especificaciones, se puede ahora explicar cómo funciona la neurona. Se supone en el modelo de neurona más simple, que corresponde a la función de activación escalón, también llamada limitador duro. En este caso, la salida puede tomar solo dos valores -1 y +1 donde la salida está determinada por

$$f(x) = \begin{cases} -1 & \text{if } u < 0 \\ +1 & \text{if } u \geq 0 \end{cases} \tag{2}$$

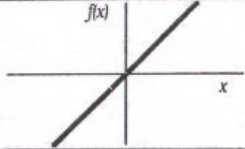
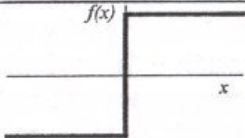
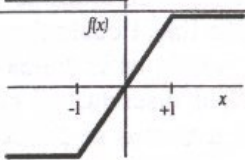
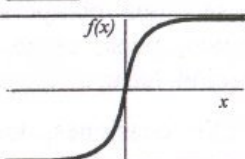
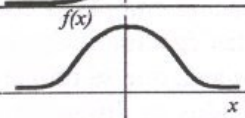
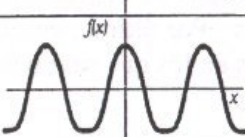
	<b>Función</b>	<b>Rango</b>	<b>Gráfica</b>
<b>Identidad</b>	$y = x$	$[-\infty, +\infty]$	
<b>Escalón</b>	$y = \text{sign}(x)$ $y = H(x)$	$\{-1, +1\}$ $\{0, +1\}$	
<b>Lineal a tramos</b>	$y = \begin{cases} -1, & \text{si } x < -l \\ x, & \text{si } -l \leq x \leq +l \\ +1, & \text{si } x > +l \end{cases}$	$[-1, +1]$	
<b>Sigmoidea</b>	$y = \frac{1}{1+e^{-x}}$ $y = \text{tgh}(x)$	$[0, +1]$ $[-1, +1]$	
<b>Gaussiana</b>	$y = Ae^{-Bx^2}$	$[0, +1]$	
<b>Sinusoidal</b>	$y = A \text{sen}(\omega x + \varphi)$	$[-1, +1]$	

Fig. 2: Funciones de activación.

Entonces, para la función sigmoidea, se tiene que

$$y = \left( \frac{1}{1 + e^{-u}} \right) \quad (3)$$

$$u = \sum_{i=1}^n w_{ji}x_i + w_0b$$

y para el segundo caso de la función sigmoidea

$$y = \tanh(u) = \left( \frac{e^u - e^{-u}}{e^u + e^{-u}} \right) \quad (4)$$

La expresión de la ecuación que almacena la neurona en virtud del vector de pesos  $W$  es el **modelo** que representa en mayor o menor grado el comportamiento del vector de salida y con respecto al vector de entrada  $X$

Entonces, una neurona artificial es un procesador elemental. Se encarga de procesar un vector de  $n$  entradas para producir un único valor de salida  $y$ . El nivel de activación depende de las entradas recibidas y de los valores sinápticos. Para calcular el estado de activación se ha de calcular en primer lugar la entrada total a la célula. Este valor se calcula como la suma de todas las entradas ponderadas por ciertos valores dados a la entrada.

## 2.1 Redes Neuronales Artificiales

La capacidad de modelar funciones más complejas aumenta grandemente cuando la neurona no trabaja sola, sino interconectada con otras neuronas, formando Redes Neuronales Artificiales (RNA), tal como se observa de manera simplificada en la Figura 3

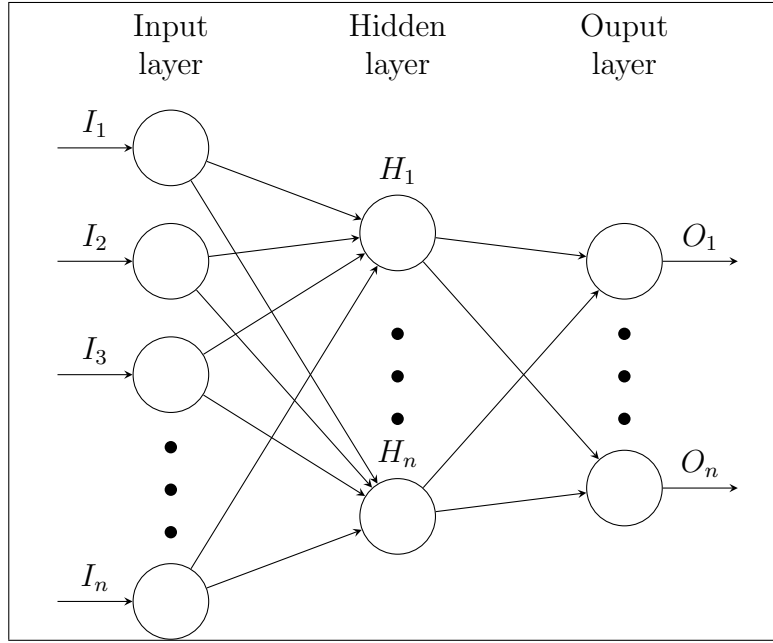


Fig. 3: Topología de la Red Perceptron Multicapa.

La red mas simple se llama **perceptron multicapa**. Esta red define una relación entre las variables de entrada y las variables de salida. Esta relación se obtiene *propagando* hacia adelante los valores de las variables de entrada. Cada neurona procesa la información recibida por sus entradas y produce una respuesta o activación que se propaga, a través de las conexiones correspondientes, hacia las neuronas de la siguiente capa.

Sea un perceptron multicapa con  $C$  capas, de las cuales una es la capa de entrada, una la capa de salida y  $C - 2$  capas ocultas. Se tiene  $n_c$  neuronas en la capa  $c$ , con  $c = 1, 2, 3, \dots, C$ . sea  $W^c = (w_{ji}^c)$  la matriz de pesos asociada a las conexiones de la capa  $c$  a la capa  $c + 1$  para  $c = 1, 2, 3, \dots, C - 1$ , donde  $w_{ji}^c$  representa el peso de la conexión de la neurona  $i$  de la capa  $C$  a la neurona



$j$  de la capa  $C + 1$ . Sea  $U^c = (u_j^c)$  el vector de umbrales de las neuronas de la capa  $c$  para  $c = 2, 3, \dots, C$ . Se denota  $o_j^c$  a la activación de la neurona  $j$  de la capa  $c$ ; estas activaciones se calculan del siguiente modo:

1. Entrada ( $o_i^1$ ). Estas neuronas transmiten hacia la red las señales recibidas del exterior

$$o_j^1 = x_j \quad \text{para} \quad j = 1, 2, 3, \dots, n_1 \quad (5)$$

Donde  $X = (x_1, x_2, \dots, x_n)$  representan el vector de entrada a la red.

2. Activación de las neuronas de la capa oculta  $c(o_j^c)$ : Las neuronas ocultas procesan la información recibida aplicando la función de activación  $f$  a la suma de los productos de las activaciones que recibe por sus correspondientes pesos:

$$o_j^c = f\left(\sum_{i=1}^{n_{c-1}} w_{ji}^{c-1} o_i^{c-1} + u_j^c\right) \quad (6)$$

Para  $j = 1, 2, 3, \dots, n_c$  y  $c = 2, 3, \dots, c - 1$

3. Salida ( $a_i^C$ ) : Al igual que en las capas ocultas, la activación de estas neuronas viene dada por la función de activación  $f$

$$y_j = o_j^c = f\left(\sum_{i=1}^{n_{c-1}} w_{ji}^{c-1} o_i^{c-1} + u_j^c\right) \quad (7)$$

Para  $j = 1, 2, 3, \dots, n_c$  donde  $Y = (y_1, y_2, y_3, \dots, y_c)$  es el vector de salida de la red.

Para el perceptrón multicapa las funciones de activación más usadas son la funciones **sigmoidal**:

$$f(u) = \frac{1}{1 + e^{-u}} \quad (8)$$

y la función **tangente hiperbólica**

$$f(u) = \left(\frac{1 - e^{-u}}{1 + e^{-u}}\right) \quad (9)$$

Estas funciones tienen una forma similar pero se diferencian en que la sigmoideal tiene un rango continuo de valores dentro de los intervalos  $[0, 1]$  mientras que la tangente hiperbólica tiene un rango continuo en el intervalo  $[-1, 1]$ .

### 3 Caso de Estudio

Todos los días, millones de e-mails invaden las bandejas de entrada de los usuarios de Internet. De todos éstos, una cantidad muy importante es considerada "correo basura". Compuesto por mensajes publicitarios no solicitados, cadenas de la suerte o incluso virus que se autoenvían, el spam afecta a más de un usuario, y hace que la tarea de revisar el correo sea una verdadera molestia. El problema fundamental lo representan los spams, que son mensajes publicitarios no solicitados. Ya no resulta raro para quienes contamos con una dirección de correo electrónico recibir a diario varios mensajes con propagandas de las más variadas temáticas. A pesar de que ningún método de detección de Spam es totalmente efectivo, consideramos que si es posible mejorar los existentes y reducir considerablemente las deficiencias que actualmente presentan las herramientas disponibles. Es un hecho que parte de los mensajes no deseados escapan a los sistemas de detección de correo basura constituyendo así un "falso negativo", igualmente existe la posibilidad de identificar un mensaje como Spam sin serlo, lo que se conoce como "falso positivo". La idea es tomar las máximas precauciones posibles para minimizar este efecto, y para ello se debe ser consciente de este hecho antes de adoptar las posibles medidas de filtrado que se propondrá.[3]

#### 3.1 Problema

El crecimiento de Internet a nivel mundial esta cambiando nuestra forma de comunicación entre otros, por lo que cada vez la gente utiliza más el correo electrónico. A causa de un número tentativo de correos electrónicos los publicistas y spammers se ven los modos para obtener un listado grande de correos y así poder enviar spam. Todos los días, billones de e-mails invaden las bandejas de entrada de los usuarios de Internet. De todos éstos, una cantidad muy importante es considerada "correo basura". Compuesto por mensajes publicitarios no solicitados, cadenas de la suerte o incluso virus que se auto envían, el spam aqueja a más de un usuario, y hace que la tarea de

revisar correo sea una verdadera molestia.[3]

Los principales problemas son los siguientes:

1. Pérdida de productividad y dinero en las empresas
2. Amenaza la viabilidad de Internet como un medio efectivo de comunicación.
3. Incremento de costos relacionados con el tiempo.
4. Genera importantes costos de seguridad a empresas ISP's.
5. Incremento de propagación de virus informáticos.
6. Saturación de servidores. Muchos servidores dedicados para uso privado o para uso general son congestionados implicando una reducción de calidad de servicio.
7. Denegación de servicios (Deny of services). Una cantidad excesiva de correos no deseados pueden congestionar totalmente el servicio y así denegarlo al mismo.
8. Buzón de entrada incontrolable por parte del receptor. Causado por la cantidad masiva que los spammers envían a los correos electrónicos.
9. Daño de imagen a terceros.
10. Molestias por parte del receptor.

### **3.2 Justificación**

El correo electrónico, es sin duda un medio que nos permite comunicar rápidamente ofreciéndonos reducción de tiempo y costo Sin embargo muchas personas aprovechan esto para utilizarlo de forma no legítima con fines publicitarios, ocasionando una serie de problemas a nivel personal como empresarial. Como contramedida a esta acción se necesitan herramientas capaces de reducir el spam. De esta manera es muy importante la elaboración de anti-spams, ya que es la forma más viable de acabar con el spam y ofrecer a los usuarios seguridad y tranquilidad en los correos electrónicos, y por otra parte reducir los costos para las empresas ISP's y controlar la saturación de

servidores de correo electrónico. **El desarrollo de una herramienta informática capaz de aminorar con los problemas que causa el spam,** no es solamente capaz de **ahorrar mucho dinero** en aquellas empresas que suelen estar perjudicadas con el spam, sino también es capaz de permitir una mejor utilización y minimizar los dolores de cabeza a cualquier usuario del correo electrónico.[3]

### **3.3 Propuestas de Solucion**

asd

#### **3.3.1 Topologia**

asd

#### **3.3.2 Reglas**

asd

##### **3.3.2.1 Regla de propagación**

asd

##### **3.3.2.2 Regla de Activacion**

asd

##### **3.3.2.3 Regla de Salida**

asd

##### **3.3.2.4 Regla de Aprendizaje**

asd

##### **3.3.2.4.1 Back Propagation**

asd

#### 3.3.2.4.2 Segundo Orden

asd

#### 3.3.2.4.3 R-PROP

asd

### 3.4 Desarrollo de la solución

Para el desarrollo del perceptrón multicapa planteado anteriormente hemos se escogio como primera herramienta  $R$ ( Figura 5) y con el IDE Rstudio( Figura 4), pero no se utilizo porque **el tiempo de aprendizaje para un numero mayor de 10 epocas era muy elevado**, a comparacion de la segunda herramienta, python.



Fig. 4: IDE Rstudio



Fig. 5: Lenguaje R

### 3.4.1 Herramientas

Como se dijo anteriormente no se utilizó la herramienta R debido a su alto tiempo en procesar la información, a continuación se describen las herramientas utilizadas.

#### 3.4.1.1 Python

Python es un lenguaje de programación interpretado cuya filosofía hace hincapié en una sintaxis que favorezca un código legible.

Se trata de un lenguaje de programación multiparadigma, ya que soporta orientación a objetos, programación imperativa y, en menor medida, programación funcional. Es un lenguaje interpretado, usa tipado dinámico y es multiplataforma.

Es administrado por la Python Software Foundation. Posee una licencia de código abierto, denominada Python Software Foundation License,<sup>1</sup> que es compatible con la Licencia pública general de GNU a partir de la versión 2.1.1, e incompatible en ciertas versiones anteriores.[4]



Fig. 6: Lenguaje Python.

### 3.4.1.2 PyCharm

El entorno de desarrollo integrado que se utilizo es PyCharm, es un entorno muy popular utilizado en la comunidad de python que provee una manera sencilla de programar en python eligiendo la version de python que deseamos compilar y agregar los paquetes necesarios para el desarrollo de nuestros proyectos.



Fig. 7: IDE Pycharm.

### 3.4.1.3 Package

Para el desarrollo de los algoritmos de aprendizaje para el perceptron multi-capas se utilizo una libreria para Inteligencia Artificial llamada NeuPy. Esta libreria actualmente en su version 0.6 soporta diferentes tipos de Redes Neuronales desde un perceptron simple hasta modelos de deep learning [5].

Actualmente soporta la siguientes características:

1. Deep Learning
2. Reinforcement Learning (RL)
3. Convolutional Neuronal Networks (CNN)
4. Recurrent Neuronal Networks (RNN)
5. Restricted Boltzmann Machine (RBM)
6. Multilayer Perceptron (MLP)
7. Networks based on the Radial Basis Functions (RBFN)
8. Ensemble Networks

- 9. Competitive Networks
- 10. Basic Linear Networks
- 11. Regularization Algorithms
- 12. Step Update Algorithms

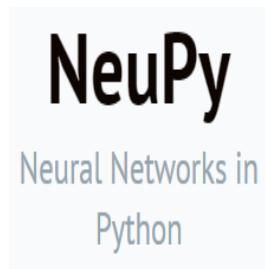


Fig. 8: Funciones de activación.

### **3.4.2 Implementacion**

#### **3.4.2.1 Back Propagation**

asd

#### **3.4.2.2 Segundo Orden**

asd

#### **3.4.2.3 R-PROP**

asd

### **3.5 Resultados**

asd

### **3.6 Conclusiones**

asd



## References

- [1] Laura Lanzarini. Redes neuronales aplicadas al reconocimiento de patrones. [http://sedici.unlp.edu.ar/bitstream/handle/10915/22061/Documento\\_completo.pdf?sequence=1](http://sedici.unlp.edu.ar/bitstream/handle/10915/22061/Documento_completo.pdf?sequence=1), 2017. [Online; accessed 15-October-2017].
- [2] P Russell, S.J.; Norvig. *Inteligencia Artificial*. Prentice Hall Hispanoamerica, 1996.
- [3] Hugo Galán Asensio. Inteligencia artificial.redes neuronales y aplicaciones. *Estudiantes*, 2010.
- [4] Python Software Foundation. The official home of the python programming language. <https://www.python.org/>, 2017. [Online; accessed 16-October-2017].
- [5] Yurii Shevchuk. Python library for artificial neural networks. <http://neupy.com/pages/home.html>, 2017. [Online; accessed 16-October-2017].