TALLER NO. 3

JUAN FELIPE MARIN

TALLER NO. 3

TRABAJO DE INTELIGENCIA ARTIFICIAL

CARLOS ALBERTO LONDOÑO LOAIZA

INGENIERO DE SISTEMAS

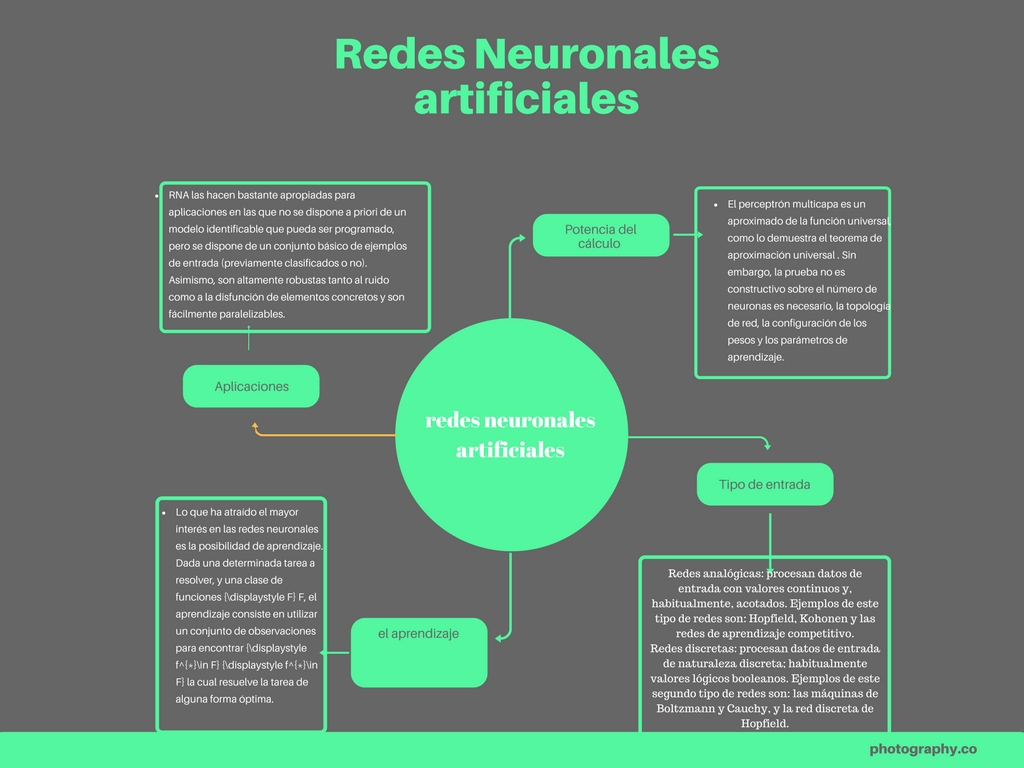
CORPORACIÒN DE ESTUDIOS TECNOLOGICOS DEL NORTE DEL VALLE

CIENCIAS INFORMÁTICAS, TECNOLÓGICAS E INGENIERÍA

[TECNOLOGIA EN SISTEMAS DE INFORMACION](http://www.cnotas.cotecnova.edu.co/servicios/servicios_est2.php)

CARTAGO VALLE

**1- consultar la historia de las redes neuronales artificiales, crear un mapa conceptual, que permita evidenciar los casos más importantes**



<https://www.canva.com/design/DACwu8GrEzY/L8zSAFLBNrd3u1QDwSgtyg/edit?utm_source=onboarding>

**2- nombre 6 0 más ventajas y desventajas que tiene el uso de las redes neuronales artificiales para la solución de problemas.**

**Ventajas**

**Aprendizaje:** Las RNA tienen la habilidad de aprender mediante una etapa que se llama etapa de aprendizaje. Esta consiste en proporcionar a la RNA datos como entrada a su vez que se le indica cuál es la salida (respuesta) esperada.

**Auto organización:** Una RNA crea su propia representación de la información en su interior, descargando al usuario de esto.

**Tolerancia a fallos:** Debido a que una RNA almacena la información de forma redundante, ésta puede seguir respondiendo de manera aceptable aun si se daña parcialmente.

**Flexibilidad:** Una RNA puede manejar cambios no importantes en la información de entrada, como señales con ruido u otros cambios en la entrada (ej. si la información de entrada es la imagen de un objeto, la respuesta correspondiente no sufre cambios si la imagen cambia un poco su brillo o el objeto cambia ligeramente)

**Tiempo real:** La estructura de una RNA es paralela, por lo cual, si esto es implementado con computadoras o en dispositivos electrónicos especiales, se pueden obtener respuestas en tiempo real.

**Desventajas**

Complejidad de aprendizaje para grandes tareas, cuantas más cosas se necesiten que aprenda una red, más complicado será enseñarle.

Tiempo de aprendizaje elevado. Esto depende de dos factores: primero si se incrementa la cantidad de patrones a identificar o clasificar y segundo si se requiere mayor flexibilidad o capacidad de adaptación de la red neuronal para reconocer patrones que sean sumamente parecidos, se deberá invertir más tiempo en lograr que la red converja a valores de pesos que representen lo que se quiera enseñar.

No permite interpretar lo que se ha aprendido, la red por si sola proporciona una salida, un número, que no puede ser interpretado por ella misma, sino que se requiere de la intervención del programador y de la aplicación en si para encontrarle un significado a la salida proporcionada.

Elevada cantidad de datos para el entrenamiento, cuanto más flexible se requiera que sea la red neuronal, más información tendrá que enseñarle para que realice de forma adecuada la identificación.

Otros problemas con las redes neuronales son la falta de reglas definitorias que ayuden a realizar una red para un problema dado.

<http://redes-neuronales.wikidot.com/definicion-ventajas-desventajas>

**3-** nombre 10 aplicaciones de las redes neuronales

En este apartado mostraremos algunas de las aplicaciones que tienen este tipo de redes en el mundo de las telecomunicaciones. El ámbito de aplicación de las Redes Neuronales Artificiales en Telecomunicaciones puede ser muy importante, considerando por un lado el tipo de problemas a resolver, por ejemplo, relacionados al tráfico de datos, y por otro lado, las soluciones que pueden ser proporcionas por las redes neuronales para este tipo de problemas. Algunas de las características, de las redes neuronales, importantes para aplicar a esta área son: - Son capaces de determinar relaciones no lineales entre un conjunto de datos, asociando patrones de entrada o salidas correspondientes. -Los tipos de aprendizaje disponibles pueden utilizarse para tareas de predicción y clasificación.

Los modelos supervisados y no supervisados pueden ser aplicados para extraer y cancelar ruido de las señales. -Una vez que la red ha sido entrenada y probada puede adaptarse por sí misma a los cambios. -Una aproximación basada en redes neuronales artificiales puede aprender los modelos específicos de cada sistema de red y proporcionar aproximaciones aceptables de los sistemas

<http://www.it.uc3m.es/jvillena/irc/practicas/10-11/06mem.pdf>

**4- ¿Qué son funciones de activación, cuales existen y para cuales redes neuronales se aplican?**

Función de activación (activation function). Una neurona biológica puede estar activa (excitada) o inactiva (no excitada); es decir, que tiene un “estado de activación”. Las neuronas artificiales también tienen diferentes estados de activación; algunas de ellas solamente dos, al igual que las biológicas, pero otras pueden tomar cualquier valor dentro de un conjunto determinado. La función activación calcula el estado de actividad de una neurona; transformando la entrada global (menos el umbral, Θi) en un valor (estado) de activación, cuyo rango normalmente va de (0 a 1) o de (–1 a 1). Esto es así, porque una neurona puede estar totalmente inactiva (0 o –1) o activa (1). La señal Net(salida) generalmente se procesa por medio de una función de activación F, la cual producirá una señal que será la salida (Out) de la neurona. La función Ftambién puede ser alguna otra función que simule mejor las características no lineales de transferencia de una neurona biológica. Si Freduce el rango de Netde manera que Outnunca salga de algún límite, independientemente de lo grande que sea Net,entonces F es una función sigmoidal, una función de este tipo es la función [logística](http://www.monografias.com/trabajos15/logistica/logistica.shtml).

<http://www.monografias.com/trabajos95/redes-neuronales-artificiales-metodologia-desarrollo-y-aplicaciones/redes-neuronales-artificiales-metodologia-desarrollo-y-aplicaciones.shtml#ixzz4Idtpckc4>

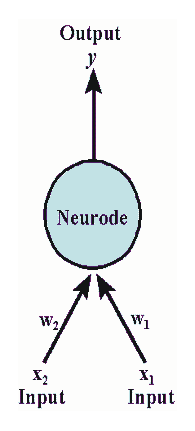
**5-** Perceptron

1. Historia
2. Fórmula matemática, explique sus términos
3. Dibuje la estructura
4. ¿para que se usa?
5. ¿Cuál es su función de activación?
6. ¿Cómo se entrena un perceptron, indique el nombre del algoritmo y sus pasos?
7. Nombre 5 ejemplos donde se evidencie el uso del perceptron.

**Historia del Perceptrón**

La evolución de la neurona artificial ha progresado a través de varias etapas.   Las raíces de las cuales, están cimentadas dentro trabajo neurológica realiza principalmente por [Santiago Ramón y Cajal](http://www.uic.edu/depts/mcne/founders/page0076.html) y [Sir Charles Scott Sherrington](http://www.uic.edu/depts/mcne/founders/page0086.html) .   Ramón y Cajal fue una figura prominente en la exploración de la estructura de tejido nervioso y demostró que, a pesar de su capacidad para comunicarse entre sí, las neuronas se separan físicamente de otras neuronas.   con una mayor comprensión de los elementos básicos del cerebro, se hicieron esfuerzos para describir cómo estas neuronas básicas podrían dar lugar a conductas manifiestas, a la que [William James](http://www.emory.edu/EDUCATION/mfp/james.html) fue un destacado colaborador teórico.

Trabajando a partir de los comienzos de la neurociencia, [Warren McCulloch](http://www.csulb.edu/~cwallis/artificialn/warren_mcculloch.html) y [Walter Pitts](http://www.csulb.edu/~cwallis/artificialn/walter_pitts.html)   en su artículo de 1943, "Un cálculo lógico de las ideas inmanentes en la actividad nerviosa", sostuvieron que las neuronas con una función de activación de umbral binario eran análogas a las oraciones lógica de primer orden.   La McCulloch básica y Pitts neurona se veía algo como lo siguiente:



La neurona McCulloch-Pitts trabajado mediante la introducción de un 1 o 0 para cada una de las entradas, en donde 1 representa verdadero y 0 falsa.   Del mismo modo, el umbral se le dio un valor real, por ejemplo 1, lo que permitiría un 0 o 1 de salida si el umbral se alcanza o se supera.   por lo tanto, con el fin de representar la "y" función, fijar el umbral en 2,0 y llegar a la siguiente tabla de verdad:

|  |  |  |
| --- | --- | --- |
| **De entrada x 1** | **De entrada x 2** | **Salida** |
| **0** | **0** | **0** |
| **0** | **1** | **0** |
| **1** | **0** | **0** |
| **1** | **1** | **1** |

Esta tabla muestra la "y" función básica tal que, si x1 y x2 son falsas, entonces la salida de la combinación de estos dos también será falsa.   Del mismo modo, si x1 es verdadera o igual a 1 y x2 es verdadera o igual a 1 , entonces el umbral del 2 será recibido y la salida será 1.

Esto se deduce también de la función "o", si cambiamos el valor de umbral a 1.   La tabla de la "o" ser función,

|  |  |  |
| --- | --- | --- |
| **De entrada x 1** | **De entrada x 2** | **Salida** |
| **0** | **0** | **0** |
| **0** | **1** | **1** |
| **1** | **0** | **1** |
| **1** | **1** | **1** |

Este tipo de neurona artificial también se podría utilizar para resolver la función "no", lo que tendría sólo una entrada, así como, las funciones NOR y NAND.   La neurona McCulloch-Pitts, por lo tanto, fue muy instrumental en el progreso de la neurona artificial, pero tiene algunas limitaciones graves.   En particular, se podría resolver ni la función "exclusiva o" función (XOR), ni la "exclusiva ni" (XNOR).  Limitado a código binario, las siguientes tablas de verdad podría *no* ser resuelto con precisión el uso de esta neurona artificial temprano.

**XOR**

|  |  |  |
| --- | --- | --- |
| **De entrada x 1** | **De entrada x 2** | **Salida** |
| **0** | **0** | **0** |
| **0** | **1** | **1** |
| **1** | **0** | **1** |
| **1** | **1** | **0** |

**XNOR**

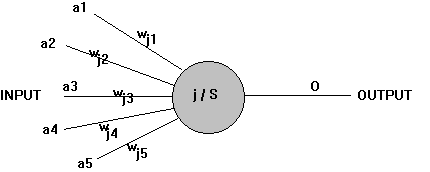
|  |  |  |
| --- | --- | --- |
| **De entrada x 1** | **De entrada x 2** | **Salida** |
| **0** | **0** | **1** |
| **0** | **1** | **0** |
| **1** | **0** | **0** |
| **1** | **1** | **1** |

Una de las dificultades con la neurona de McCulloch-Pitts fue su sencillez.   Sólo se permite para las entradas y salidas binarias, sólo se utiliza la función de activación de la etapa de umbral y no ha incorporado la ponderación de las diferentes entradas.

En 1949, [Donald Hebb](http://cogprints.ecs.soton.ac.uk/~harnad/Archive/hebb.html) podría ayudar a revolucionar la forma en que se perciben las neuronas artificiales.   En su libro, *La organización de la conducta*, propuso lo que ha llegado a ser conocido como la regla de Hebb.   Él afirma: "Cuando un axón de la célula A está cerca suficiente para excitar a una célula B y repetidamente o persistentemente toma parte en la cocción, algunos procesos de crecimiento o cambio metabólico se lleva a cabo en una o ambas células de tal manera que la eficiencia de a, como una de las celdas de disparo B, se incrementa. " [[1]](https://web.csulb.edu/~cwallis/artificialn/History.htm" \l "_ftn1" \o ")   Hebb se propone no sólo eso, cuando dos neuronas se activan conjuntamente la conexión entre las neuronas se fortalece, sino también que esta actividad es una de las operaciones fundamentales necesarias para el aprendizaje y la memoria.

Para la neurona artificial, esto significaba que la neurona de McCulloch-Pitts tuvo que ser modificado para permitir al menos la esta nueva propuesta biológica.   El método empleado fue ponderar cada una de las entradas.   Por lo tanto, una entrada de 1 puede darse más o menos peso, con respecto a la suma de umbral total.

Frank Rosenblatt, utilizando la neurona de McCulloch-Pitts y los resultados de Hebb, pasó a desarrollar el primer perceptrón.   Este perceptrón, lo que podría aprender en el sentido Hebbean, a través de la ponderación de los insumos, jugó un papel decisivo en la formación posterior de las redes neuronales.   se refirió a la perceptrón en su libro de 1962, *Principios de Neurodinámico* .   Un perceptrón básico se representa como sigue:



Este perceptrón tiene un total de cinco entradas A1 a A5 y cada una tiene un peso de W1 a través w5. [[2]](https://web.csulb.edu/~cwallis/artificialn/History.htm" \l "_ftn2" \o ")   Cada una de las entradas son ponderados y sumados en el nodo.   Si se alcanza el umbral, se produce una salida.   De gran importancia es que cada una de las entradas que no se puede dar el mismo peso.   El perceptrón puede haber "aprendido" a peso A1 a A2 y así sucesivamente.

La fórmula de sumación para determinar si es o no el umbral ( *θ* ) se reunió por la neurona artificial con N entradas (un 1 , un 2 ... a N ) y sus respectivos pesos de w 1 , w 2 , ... w Nes:

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  |  | **norte** |  |  |  |  |  |  |  |
| segundo | = | (Σw jun j ) | + | *θ* |  |  |  |  |  |
|  |  | **j = 1** |  |  |  |  |  |  |  |

La función de activación se convierte en:

                        x = f (b)

La función de activación utilizado por McCulloch y Pitts fue la función de paso umbral.   Sin embargo, otras funciones que se pueden utilizar son el sigmoide, lineal por partes y funciones de activación de Gauss.   Estas funciones se muestran a continuación. [[3]](https://web.csulb.edu/~cwallis/artificialn/History.htm" \l "_ftn3" \o ") (Ver el glosario adjunto a este subprograma para las fórmulas matemáticas correspondientes.)

|  |  |  |  |
| --- | --- | --- | --- |
| Paso umbral | Sigmoideo | Lineal a trozos | gaussiano |
| https://web.csulb.edu/~cwallis/labs/artificialn/History_files/image007.gif | https://web.csulb.edu/~cwallis/labs/artificialn/History_files/image009.gif | https://web.csulb.edu/~cwallis/labs/artificialn/History_files/image011.gif | https://web.csulb.edu/~cwallis/labs/artificialn/History_files/image013.gif |

A pesar de los numerosos cambios realizados en el original de la neurona de McCulloch-Pitts, el perceptrón todavía estaba limitado a la solución de ciertas funciones   Desafortunadamente, Rosenblatt era demasiado entusiasta sobre el perceptrón e hizo el anuncio inoportuno que:

"Dada una α-perceptrón elemental, un mundo de estímulos W, y cualquier clasificación C (W) para los que existe una solución, dejar que todos los estímulos en W se producen en cualquier orden, siempre que cada estímulo debe volver a ocurrir en un tiempo finito; a continuación, a partir de un estado inicial arbitrario, un procedimiento de corrección de errores siempre dará lugar a una solución a C (W) en un tiempo finito

Con estos tipos de observaciones Rosenblatt había trazado una línea en la arena entre los de apoyo de perceptrón estilo de investigación y los proyectos de manipulación de símbolos más tradicionales que se realiza por [Marvin Minsky](http://web.media.mit.edu/~minsky/) .   Como resultado, en 1969, Minsky coautoría con [Seymour Papert](http://papert.www.media.mit.edu/people/papert/) , *perceptrones : Una introducción a la geometría computacional* .   En este trabajo se atacó a las limitaciones del perceptrón.   demostraron que el perceptrón sólo se podía resolver funciones linealmente separables.   de particular interés fue el hecho de que el perceptrón todavía no podía resolver las funciones NXOR XOR y.   de la misma manera, Minsky y Papert indicaron que el estilo de la investigación que se realiza en el perceptrón estaba condenado al fracaso debido a estas limitaciones.   Esto fue, por supuesto, la observación igualmente inoportuna de Minsky.   Como resultado, hay muy poca investigación se llevó a cabo en la zona hasta acerca de la década de 1980.

Lo que vendría a resolver muchas de las dificultades fue la creación de redes neuronales.   Estas redes se conectan las entradas de las neuronas artificiales con las salidas de otras neuronas artificiales.  Como resultado, las redes fueron capaces de resolver los problemas más difíciles, pero han crecido considerablemente más complejo.   Sin embargo, muchas de las redes neuronales artificiales en uso hoy en día todavía se derivan de los primeros avances de la neurona de McCulloch-Pitts y el perceptrón de Rosenblatt.

<https://web.csulb.edu/~cwallis/artificialn/History.htm>

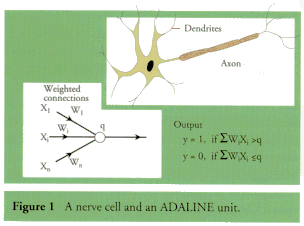
**6-** Adaline

1. Historia
2. Fórmula matemática, explique sus términos
3. Dibuje la estructura
4. ¿para que se usa?
5. ¿Cuál es su función de activación?
6. ¿Cómo se entrena una red Adaline, indique el nombre del algoritmo y sus pasos?
7. Nombre 5 ejemplos donde se evidencie el uso de Adaline.

En la década de 1950, con el avance de las computadoras, se hizo posible simular una red neuronal. Nathanial Rochester, de los laboratorios de investigación de la IBM, dio el primer paso en la simulación de la red neuronal, pero desafortunadamente fracaso. Posteriormente se hicieron otras simulaciones ya con éxito.

En 1959, Bernard Widrow y Marcian Hoff de la universidad de Stanford desarrollaron un modelo llamado “ADALINE” y “MADALINE”. ADALINE proviene de ADAptive LINear Element y MADALINE de Multiple ADAptive LINear Element (Many Adalines).

ADALINE fue desarrollado para el reconocimiento de patrones binarios, por ejemplo, predecir el siguiente bit en una línea telefónica. En la figura 1 podemos observar la representación de una ADALINE y su analogía con una Neurona.



ADALINE funciona tomando la suma de los pesos de las entradas y produce una salida con 0 o 1 dependiendo si pasa o no un umbral, esto haciendo analogía al funcionamiento de una neurona que se dispara si la actividad total procedente de las conexiones con las otras neuronas sobrepasa un nivel.

Varias ADALINE pueden ser organizadas en capas de tal manera que se obtengan grandes arquitecturas formando una red MADALINE la cual produce funciones más complicadas.

MADALINE fue la primera red neuronal aplicada a un problema real, se usó como un filtro para eliminar el eco en las líneas telefónicas.

En los años siguientes el neuro-biólogo Frank Rosenblatt comenzó a trabajar con el perceptrón. Se encontró que una simple capa de perceptrones era suficiente para clasificar un conjunto de valores continuos en una de dos clases.

Desafortunadamente el perceptrón estaba limitado y fue probado por Marvin Minsky y Seymour Papert durante los años de desilusión en el libro “Perceptrons”.

Durante los próximos 20 años se dejó atrás el estudio de las redes neuronales y fue hasta 1982 con John Hopfield que se retomó el interés en la materia. Hopfield presentó un documento a la Academia Nacional de Ciencias en el cual a través de análisis matemático mostraba como las redes neuronales funcionan y para que podrían servir.

# ADALINE

## ADALINE que por sus siglas en inglés significa ADAptive LINear Element es un dispositivo que consta de un solo elemento de procesamiento, por lo que técnicamente no es una red. Sin embargo, es un elemento muy importante, ya que de él se derivan redes más complejas. En la siguiente figura se muestran los elementos de una ADALINE.

## 

#### ADAptive LINear Element

Una ADALINE consiste de un ALC (Adaptive Linear Combiner) y un cuantizador (función bipolar de salida). Se alimenta con un vector de entrada (valores observados) y con una entrada constante igual a 1 denominada sesgo (bias). Posteriormente se efectúa una suma ponderada de los valores de entrada con sus pesos asociados; si el resultado es positivo, la salida del ADALINE es 1, en caso contrario es 0 (o -1). En consecuencia, ADALINE sólo resuelve adecuadamente problemas binarios linealmente separables.

Algoritmo de Entrenamiento

El entrenamiento del ADALINE está basado en la regla LMS (Least Mean Square) que busca minimizar el error cuadrático medio por medio de la regla delta.

Dado un vector de entrada (x0, ..., xn) donde x0 =1, el correspondiente valor de pesos (w0, ..., wn) y el valor deseado de salida d, el error cuadrático es el siguiente:



El objetivo del método es minimizar dicho error mediante la modificación del vector de pesos (w0, ..., wn) sumándole un **w** de tal forma que nos acerquemos al error mínimo en la dirección del gradiente negativo, es decir, lo más rápidamente posible.

El procedimiento de derivación se presenta a continuación:



Por lo que el error local será reducido más rápidamente si se ajustan los pesos de acuerdo a la regla delta:



El algoritmo de entrenamiento se presenta a continuación:

1.Inicializar pesos (w1, ..., wn) y threshold (w0)

2.Presentar vector de entrada (x1,...,xn) y la salida deseada d(t)

3.Calcular la salida



donde Fh(a) = 1 sí a>0 y

= -1 sí a<=0

4. Adaptar los pesos



donde 0 < i < n y  es la tasa de aprendizaje

5. Repetir los pasos 2 a 4 hasta que las salidas reales y las deseadas sean iguales para todos los vectores del conjunto de entrenamiento

Siguiendo este método se garantiza que, para un conjunto de entrenamiento adecuado, después de un número finito de iteraciones el error se reduce a niveles aceptables. El número de iteraciones necesarias y el nivel de error deseado depende de cada problema particular.

Adaline/ADALINEYMADALINE2.doc

**7-** ¿Qué es y para que se usa las redes de retropropagación (backpropagation)

Al hablar de redes de retropropagación o redes de propagación hacia atrás hacemos referencia a un algoritmo de aprendizaje más que a una arquitectura determinada. La retropropagación consiste en propagar el error hacia atrás, es decir, de la capa de salida hacia la capa de entrada, pasando por las capas ocultas intermedias y ajustando los pesos de las conexiones con el fin de reducir dicho error. Hay distintas versiones o reglas del algoritmo de retropropagación y distintas arquitecturas conexionistas a las que pueden ser aplicados.

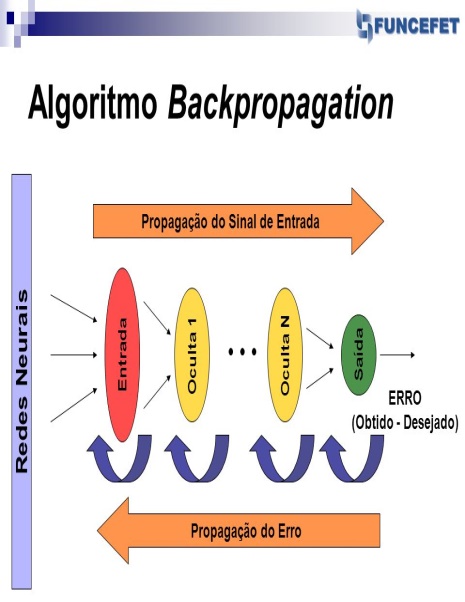
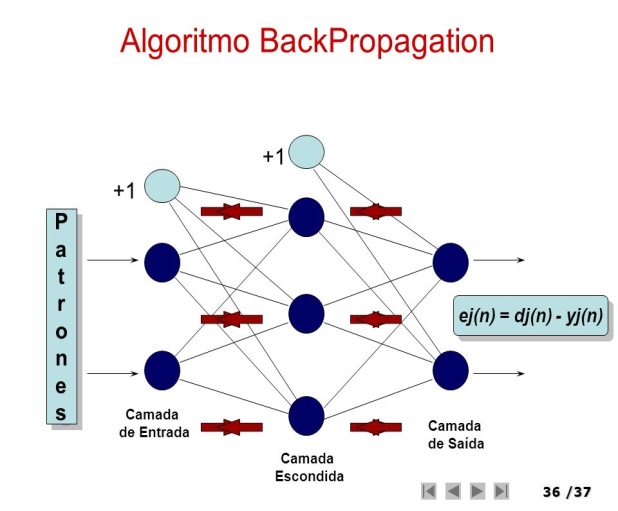
Durante mucho tiempo no se dispuso de algoritmos para entrenar redes multicapa, y como las redes de una capa estaban muy limitadas en cuanto a lo que eran capaces de representar, el campo de las redes neuronales artificiales estaba estancado. La invención y perfeccionamiento del algoritmo de retropropagación dio un gran impulso al desarrollo de este campo. Tiene un buen fundamento matemático y a pesar de sus limitaciones ha expandido enormemente el rango de problemas donde se aplican las redes neuronales artificiales.

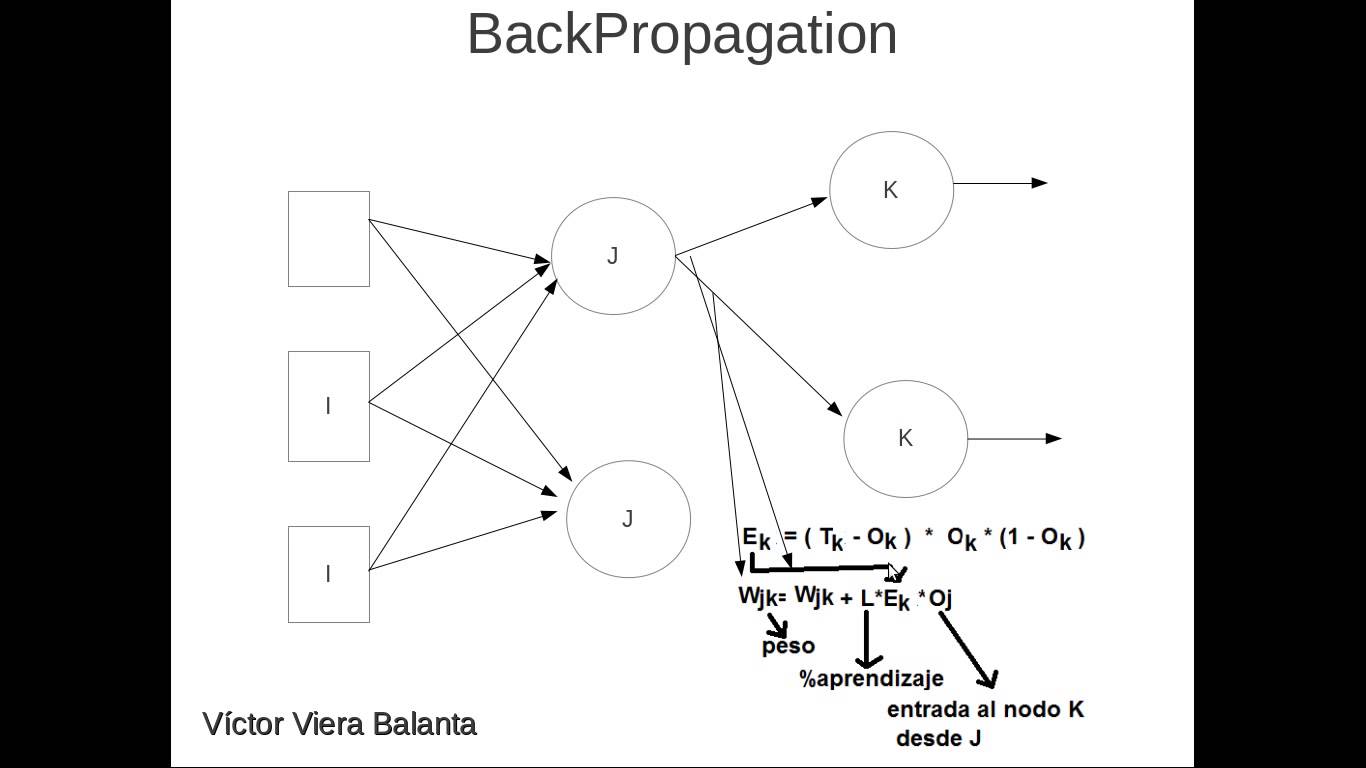
Al parecer el algoritmo fue ideado a principios de los 70 por Werbos, y redescubierto a principios de los 80 por Parker y Rumelhart independientemente, sin embargo, no se hizo popular hasta 1986, cuando Rumerlhart, Hinton y Williams presentaron una descripción clara y concisa del mismo. Y es que en un primer momento no se valoró como se merecía. El hecho de que permaneciera en el olvido tanto tiempo también debe ser una consecuencia de la condición interdisciplinar del campo, repartido entre las matemáticas y ciencias de la computación, las neurociencias y la psicología.

Desde la fecha clave de 1986 han surgido nuevas versiones que han tratado de aumentar la velocidad de convergencia del algoritmo y han tratado de superar algunos de sus inconvenientes, como la tendencia a alcanzar mínimos locales y no globales, punto que será discutido más tarde.

<http://rna.50webs.com/tutorial/RNA_backprop.html>

**8-** explique el algoritmo backpropagation, usando imágenes.





9- haciendo uso del lenguaje de programación Python graficar las siguientes funciones:

1. Función escalón
2. Función lineal y mixta
3. Función tangente hiperbólica
4. Función sigmoídal
5. Función de gauss

10- ¿Qué es el teorema de kolmogórov, explicar?

El teorema de Kolmogórov es un notable teorema de [A. Kolmogórov](https://es.wikipedia.org/wiki/Andr%C3%A9i_Kolmog%C3%B3rov) que establece que una función continua de diversas variables se puede presentar como una superposición de funciones de una única variable. Más concretamente establece que una función continua de *n* variables definida sobre el conjunto [0,1]*n* siempre puede expresarse mediante sumas y 2(*n*+1)(*n*+1) funciones de una sola variable en la forma:{\displaystyle f(x\_{1},x\_{2},\dots ,x\_{n})=\sum \_{i=1}^{2n+1}g\_{i}\left(\sum \_{j=1}^{n}h\_{ij}(x\_{j})\right)}

Donde las 2*n*+1 funciones *gi* y las 2(*n*+1)*n* funciones *hij* son funciones continuas de una variable. Kolmogórov declaró que este resultado fue el más difícil desde el punto de vista técnico al que se había enfrentado y el que le exigió un período más largo de concentración en el mismo problema.

**Teorema de** [Kolmogorov](https://www.uv.es/ceaces/autores/autores/kolmogorov.htm)**. (**[ley fuerte de los grandes números](https://www.uv.es/ceaces/tex1t/2%20conver/fuertes.htm)**)** Dada una sucesión de variables aleatorias independientes {Xn}0000.bmp (790 bytes) con medias https://www.uv.es/ceaces/tex1t/2%20conver/Image693.gif y varianza https://www.uv.es/ceaces/tex1t/2%20conver/Image694.gif      
estableciéndose https://www.uv.es/ceaces/tex1t/2%20conver/Image665.gif      y   Image746.gif (1230 bytes)  
se cumple que existe ley fuerte de los grandes números; así

Image695.gif (1294 bytes)           ó bien.      https://www.uv.es/ceaces/tex1t/2%20conver/Image696.gif

Por lo que la variable aleatoria media de una sucesión [converge de manera casi segura](https://www.uv.es/ceaces/tex1t/2%20conver/tipos.htm) a la media de las medias de las variables que forman la sucesión.

<https://www.uv.es/ceaces/tex1t/2%20conver/kolmogorov.htm>

[https://es.wikipedia.org/wiki/Teorema\_de\_superposici%C3%B3n\_de\_Kolmog%C%B3rov](https://es.wikipedia.org/wiki/Teorema_de_superposici%C3%B3n_de_Kolmog%25C%B3rov)

BIBLIOGRAFÍA

[https://www.canva.com/design/DACwu8GrEzY/L8zSAFLBNrd3u1QDwSgtyg/edit?utm source=onboarding](https://www.canva.com/design/DACwu8GrEzY/L8zSAFLBNrd3u1QDwSgtyg/edit?utm%20%20source=onboarding)

<http://redes-neuronales.wikidot.com/definicion-ventajas-desventajas>

<http://www.it.uc3m.es/jvillena/irc/practicas/10-11/06mem.pdf>

<http://es.slideshare.net/mentelibre/redes-neuronales-artificiales-historia-y-teoria>

<http://www.monografias.com/trabajos95/redes-neuronales-artificiales-metodologia-desarrollo-y-aplicaciones/redes-neuronales-artificiales-metodologia-desarrollo-y-aplicaciones.shtml#ixzz4Idtpckc4>

<https://web.csulb.edu/~cwallis/artificialn/History.htm>

Adaline/ADALINEYMADALINE2.doc

<http://rna.50webs.com/tutorial/RNA_backprop.html>

<https://www.uv.es/ceaces/tex1t/2%20conver/kolmogorov.htm>

[https://es.wikipedia.org/wiki/Teorema\_de\_superposici%C3%B3n\_de\_Kolmog%C%B3rov](https://es.wikipedia.org/wiki/Teorema_de_superposici%C3%B3n_de_Kolmog%25C%B3rov)