CORORACION DE ESTUDIOS TECNOLOGICOS DEL NORTE DEL VALLE

Trabajo de: REDES NEURONALES

Presentado a: CARLOS ALBERTO LONDOÑO LOAIZA

Presentado por: CARLOS DAVID MOSQUERA ASTIE

Programa: INGENIERIA DE SISTEMAS

Fecha:05/09/16

Cartago-Valle

**HISTORIA REDES NEURONALES**

Las redes neuronales como su nombre lo indica pretenden imitar a pequeñísima escala la forma de funcionamiento de las neuronas que forman el cerebro humano. Todo el desarrollo de las redes neuronales tiene mucho que ver con la neurofisiología, no en vano se trata de imitar a una neurona humana con la mayor exactitud posible. Entre los pioneros en el modelado de neuronas se encuentra Warren Mc Culloch y Walter Pitts.

Estos dos investigadores propusieron un modelo matemático de neurona. En este modelo cada neurona estaba dotada de un conjunto de entradas y salidas. Cada entrada está afectada por un peso. La activación de la neurona se calcula mediante la suma de los productos de cada entrada y la salida es una función de esta activación. La principal clave de este sistema se encuentra en los pesos de las diferentes entradas. Como se ha visto, las entradas son modificadas por el peso y las salidas son función de estas modificaciones. Esto nos lleva a concluir que los pesos influyen de forma decisiva en la salida y por lo tanto pueden ser utilizados para controlar la salida que se desea.

En realidad cuando se tienen interconectadas muchas de estas neuronas artificiales lo que se hace inicialmente es entrenar el sistema. El entrenamiento consiste en aplicar unas entradas determinadas a la red y observar la salida que produce. Si la salida que produce no se adecua a la que se esperaba, se ajustan los pesos de cada neurona para interactivamente ir obteniendo las respuestas adecuadas del sistema. A la red se le somete a varios ejemplos representativos, de forma que mediante la modificación de los pesos de cada neurona, la red va "aprendiendo".

**VENTAJAS**

Las redes neuronales artificiales (RNA) tienen muchas ventajas debido a que estan basadas en la estructura del sistema nervioso, principalmente el cerebro.

1. 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.
2. Auto organización: Una RNA crea su propia representación de la información en su interior, descargando al usuario de esto.
3. 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.
4. 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)
5. 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.
6. Auto-organización: una red neuronal puede crear su propia organización o representación de la información que recibe mediante una etapa de aprendizaje.

Hay muchas buenas razones para el uso de redes neuronales y los avances en este campo incrementarán su popularidad. Son excelentes como clasificadores/reconocedores de patrones – y pueden ser usadas donde las técnicas tradicionales no funcionan. Las redes neuronales pueden manejar excepciones y entradas de datos anormales, muy importante para sistemas que manejan un amplio rango de datos (sistemas de radar y sonar, por ejemplo). Muchas redes neuronales son biológicamente plausibles, lo que significa que pueden proveer pistas de como trabaja el cerebro según progresen. Avances en la neurociencia también ayudarán al avance en las redes neuronales y hasta el punto en que sean capaces de clasificar objetos con la precisión de un humano y la velocidad de una computadora! El futuro es brillante.

**DESVENTAJAS DE LAS REDES NEURONALES**

1. Complejidad de aprendizaje para grandes tareas, cuanto más cosas se necesiten que aprenda una red, más complicado será enseñarle.
2. 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.
3. 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.
4. 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.
5. Otros problemas con las redes neuronales son la falta de reglas definitorias que ayuden a realizar una red para un problema dado.
6. La mayoría de ellas, de todas maneras, padecen nuestra falta de hardware. La capacidad de las redes neuronales radica en su habilidad de procesar información en paralelo (esto es, procesar múltiples pedazos de datos simultáneamente). Desafortunadamente, las máquinas hoy en día son serie - sólo ejecutan una instrucción a la vez. Por ello, modelar procesos paralelos en máquinas serie puede ser un proceso que consuma mucho tiempo. Como todo en este día y época, el tiempo es esencial, lo que a menudo deja las redes neuronales fuera de las soluciones viables a un problema.

**APLICACIONES DE LAS REDES NEURONALES**   
Aplicaciones de las redes neuronales. Las redes neuronales pueden utilizarse en un gran número y variedad de aplicaciones, tanto comerciales como militares. Se pueden desarrollar redes neuronales en un periodo de tiempo razonable, con la capacidad de realizar tareas concretas mejor que otras tecnologías. Cuando se implementan mediante hardware (redes neuronales en chips VLSI), presentan una alta tolerancia a fallos del sistema y proporcionan un alto grado de paralelismo en el procesamiento de datos. Esto posibilita la inserción de redes neuronales de bajo coste en sistemas existentes y recientemente desarrollados. Hay muchos tipos diferentes de redes neuronales; cada uno de los cuales tiene una aplicación particular más apropiada. Algunas aplicaciones comerciales son:

Biología: - Aprender más acerca del cerebro y otros sistemas. - Obtención de modelos de la retina.

## Empresa: - Evaluación de probabilidad de formaciones geológicas y petrolíferas. - Identificación de candidatos para posiciones específicas. - Explotación de bases de datos. - Optimización de plazas y horarios en líneas de vuelo. - Optimización del flujo del tránsito controlando convenientemente la temporización de los semáforos. - Reconocimiento de caracteres escritos. - Modelado de sistemas para automatización y control.

## Medio ambiente: - Analizar tendencias y patrones. - Previsión del tiempo.

## Finanzas: - Previsión de la evolución de los precios. - Valoración del riesgo de los créditos. - Identificación de falsificaciones. - Interpretación de firmas.

## Manufacturación: - Robots automatizados y sistemas de control (visión artificial y sensores de presión, temperatura, gas, etc.). - Control de producción en líneas de procesos. - Inspección de la calidad.

## Medicina: - Analizadores del habla para ayudar en la audición de sordos profundos. - Diagnóstico y tratamiento a partir de síntomas y/o de datos analíticos (electrocardiograma, encefalogramas, análisis sanguíneo, etc.). - Monitorización en cirugías. - Predicción de reacciones adversas en los medicamentos. - Entendimiento de la causa de los ataques cardíacos.

## Militares: - Clasificación de las señales de radar. - Creación de armas inteligentes. - Optimización del uso de recursos escasos. - Reconocimiento y seguimiento en el tiro al blanco. La mayoría de estas aplicaciones consisten en realizar un reconocimiento de patrones, como ser: buscar un patrón en una serie de ejemplos, clasificar patrones, completar una señal a partir de valores parciales o reconstruir el patrón correcto partiendo de uno distorsionado. Sin embargo, está creciendo el uso de redes neuronales en distintos tipos de sistemas de control.

## Desde el punto de vista de los casos de aplicación, la ventaja de las redes neuronales reside en el procesado paralelo, adaptativo y no lineal. El dominio de aplicación de las redes neuronales también se lo puede clasificar de la siguiente forma: asociación y clasificación, regeneración de patrones, regresión y generalización, y optimización

**QUE SON FUNCIONES DE ACTIVACIÓN, CUALES EXISTEN Y PARA CUALES REDES NEURONALES SE APLICAN**

Función de Activación (Función de neurona)

El valor de red, expresado por la función de base, u(w, x), se transforma mediante una función de activación no lineal. Las funciones de activación más comunes son la función sigmoidal y gausiana:

Función sigmoidal f (ui) = 1 1 + exp {−ui σ2}

Función gausiana f (ui) = c exp {−u 2 i σ 2}

Estructuras de conexión de atrás hacia delante Una red neuronal se determina por las neuronas y la matriz de pesos.

Hay tres tipos de capas de neuronas:

La capa de entrada, la capa oculta y la capa de salida. Entre dos capas de neuronas existe una red de pesos de conexión, que puede ser de los siguientes tipos: hacia delante, hacia atrás, lateral y de retardo.

**FUNCIONES DE ACTIVACIÓN**

En redes computacionales, la **Función de Activación** de un nodo define la salida de un nodo dada una entrada o un conjunto de entradas. Se podría decir que un circuito estándar de computador se comporta como una red digital de funciones de activación al activarse como "ON" (1) u "OFF" (0), dependiendo de la entrada. Esto es similar al funcionamiento de un [Perceptrón](https://es.wikipedia.org/wiki/Perceptr%C3%B3n) en una [Red neuronal artificial](https://es.wikipedia.org/wiki/Red_neuronal_artificial).

**FUNCIONES**

En las redes neurales inspiradas sobre la biología, la función de activación es usualmente una abstracción representando una tasa de *potencial de activación* gatillándose en la celda. En su forma simplificada, esta función es binaria, esto es, se activa la neurona o no. La función se ve como {\displaystyle \phi (v\_{i})=U(v\_{i})}, donde {\displaystyle U} es la función escalón. En este caso, un gran número de neuronas deben ser usadas en computación más allá de la separación lineal de las categorías.

Una función rampa también puede ser usada para reflejar el incremento del potencial de activación que ocurre cuando la entrada se incrementa. La función podría ser de la forma {\displaystyle \phi (v\_{i})=\mu v\_{i}}, donde {\displaystyle \mu } es la pendiente. Esta función de activación es lineal, y por consiguiente tiene los mismos problemas que la función binaria. En adición, las redes neurales construidas usando este modelo tienen convergencia inestable porque a la larga, las entradas a la neurona tienden a incrementarse sin límite, esta función no es normalizable.

Los problemas mencionados anteriormente, pueden ser manejados usando una [función de activación sigmoidal](https://es.wikipedia.org/wiki/Funci%C3%B3n_sigmoidea). Un modelo realístico permanece en cero hasta que una entrada es recibida, en este punto la frecuencia de activación se incrementa rápidamente, pero gradualmente llega a ser [asíntota](https://es.wikipedia.org/wiki/As%C3%ADntota) cuando la frecuencia es 100%.

Matemáticamente, esto se ve como {\displaystyle \phi (v\_{i})=U(v\_{i})\tanh(v\_{i})}, donde la función de [tangente hiperbólica](https://es.wikipedia.org/wiki/Tangente_hiperb%C3%B3lica) puede también ser cualquier función sigmoidal. Esta conducta es realísticamente reflejada en la neurona, ya que las neuronas no pueden físicamente activarse más rápido que una cierta tasa.

El modelo final que es usado en [perceptrones multicapa](https://es.wikipedia.org/wiki/Perceptr%C3%B3n_multicapa) es el modelo de activación sigmoidal en la forma de tangente hiperbólica. Dos formas de esta función son comúnmente usados: {\displaystyle \phi (v\_{i})=\tanh(v\_{i})} cuyos rangos son normalizados desde -1 hasta 1, y {\displaystyle \phi (v\_{i})=(1+\exp(-v\_{i}))^{-1}}es verticalmente normalizado desde 0 a 1. El último modelo es frecuentemente considerado más biológicamente realístico, pero tiene dificultades teóricas y experimentales en ciertos tipos de problemas computacionales.

**FUNCIONES DE ACTIVACIÓN**

Además de la regla de propagación es necesario poseer expresiones para las funciones de activación (calculan la activación en función de la entrada total) y funciones de salida (calculan la salida en función de la activación).

La función de activación calcula la activación de la unidad en función de la entrada total y la activación previa, aunque en la mayor parte de los casos es simplemente una función no decreciente de la entrada total. Los tipos de función más empleados son: la función escalón, función lineal y la función sigmoidal.

La función de salida empleada usualmente es la función identidad y así la salida de la unidad de procesado es idéntica a su nivel de activación.

Las redes neuronales están formadas por un conjunto de neuronas artificiales interconectadas. Las neuronas de la red se encuentran distribuidas en diferentes capas de neuronas, de manera que las neuronas de una capa están conectadas con las neuronas de la capa siguiente, a las que pueden enviar información.

La arquitectura más usada en la actualidad de una red neuronal, se presente en la Fig. 1.8, la cual consiste en:

1. Una primera capa de entradas, que recibe información del exterior.
2. Una serie de capas ocultas (intermedias), encargadas de realizar el trabajo de la red.
3. Una capa de salidas, que proporciona el resultado del trabajo de la red al exterior.

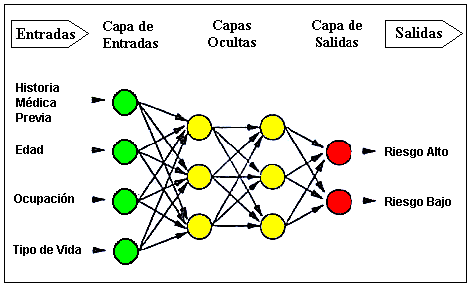
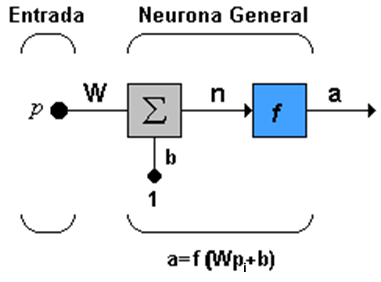


Figura 1.8 Esquema de una red neuronal antes del entrenamiento. Los círculos representan neuronas, mientras las flechas representan conexiones entre las neuronas.

El número de capas intermedias y el número de neuronas de cada capa dependerá del tipo de aplicación al que se vaya a destinar la red neuronal.

**FUNCIONES DE ACTIVACIÓN Y SALIDA**

Un modelo más académico que facilita el estudio de una neurona, puede visualizarse en la Fig.



Neurona de una sola entrada.

Donde:

1. nk: es la salida lineal de la neurona.
2. Wij: es la matriz de pesos.
3. Pi:es el patrón de entrenamiento.
4. bk:es el umbral de activación de la neurona.

Las entradas a la red serán ahora presentadas en el vector **p**, que para el caso de una sola neurona contiene solo un elemento, **w** sigue representando los pesos y la nueva entrada **b** es una ganancia que refuerza la salida del sumador **n**, la cual es la salida neta de la red; la salida total está determinada por la función de transferencia , la cual puede ser una función lineal o no lineal de **n**, y que es escogida dependiendo de las especificaciones del problema que la neurona tenga que resolver; aunque las RNA se inspiren en modelos biológicos no existe ninguna limitación para realizar modificaciones en las funciones de salida, así que se encontrarán modelos artificiales que nada tienen que ver con las características del sistema biológico.

**Funcion Escalón (Hardlim)**

La Fig. 1.10, muestra como esta función de transferencia acerca la salida de la red a cero, si el argumento de la función es menor que cero y la lleva a uno si este argumento es mayor que uno. Esta función crea neuronas que clasifican las entradas en dos categorías diferentes, característica que le permite ser empleada en la red tipo Perceptrón.

http://www.hugo-inc.com/RNA/Unidad%201/imagen/ec1.3.jpg

(1.3)

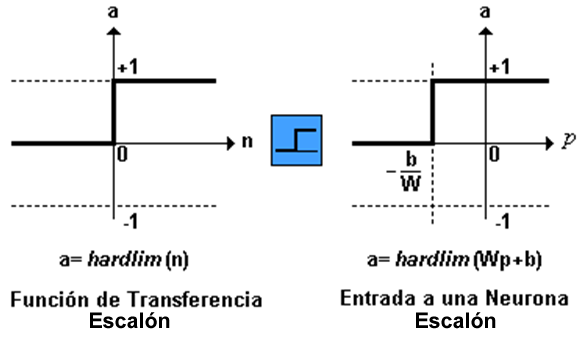


Figura 1.10 Función de Transferencia Escalón (Hardlim).

El ícono para la función escalón (Hardlim) reemplazará a la letra f en la expresión general, cuando se utilice la función Hardlim.

Una modificación de esta función puede verse en la Fig. 1.11, la que representa la función de transferencia Escalón Simétrica (Hardlims) que restringe el espacio de salida a valores entre 1 y –1.

http://www.hugo-inc.com/RNA/Unidad%201/imagen/image044.png

(1.4)

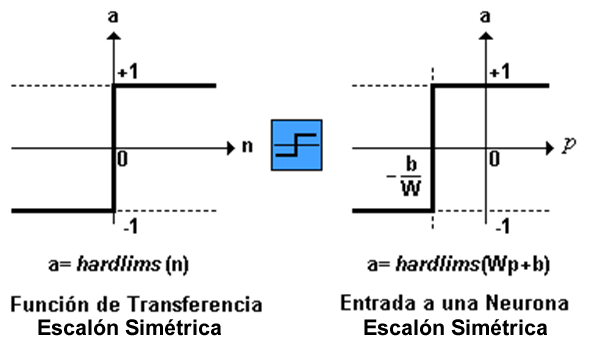


Figura 1.11 Función de Transferencia de Escalón Simétrica.

Función de Transferencia lineal (purelin)

La salida de una función de transferencia lineal es igual a su entrada, la cual se representa en la figura 1.12.

http://www.hugo-inc.com/RNA/Unidad%201/imagen/image047.png

(1.5)

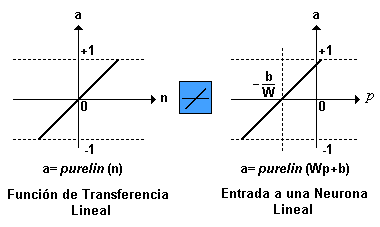


Figura 1.12 Función de Transferencia Lineal (purelin).

En la gráfica del lado derecho de la figura 1.12, puede verse la característica de la salida a de la red, comparada con la entrada p, más un valor de ganancia b, neuronas que emplean esta función de transferencia son utilizadas en la red tipo Adaline.

Funcion de Transferencia sigmoidal (logsig)

Esta función toma los valores de entrada, los cuales pueden oscilar entre más y menos infinito, y restringe la salida a valores entre cero y uno, de acuerdo a la expresión:

http://www.hugo-inc.com/RNA/Unidad%201/imagen/image050.png

(1.6)

Esta función es comúnmente usada en redes multicapa, como la Back propagation, en parte porque la función logsig es diferenciable, como se muestra en la Fig. 1.13.

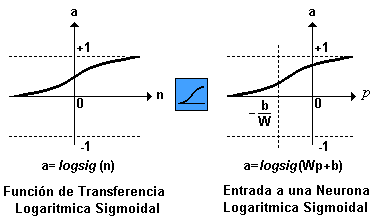


Figura 1.13 Función de Transferencia Sigmoidal.

La Fig. 1.14, hace una relación de las principales funciones de transferencia empleadas en el entrenamiento de redes neuronales.

|  |  |  |  |
| --- | --- | --- | --- |
| **Nombre** | **Relación Entrada /Salida** | **icono** | **Función en Matlab** |
| **Limitador Fuerte** | harlim | ic1 | hardlim |
| **Limitador Fuerte Simétrico** | harlims | ic2 | hardlims |
| **Lineal Positiva** | poslin | ic3 | poslin |
| **Lineal** | purelin | ic4 | purelin |
| **Lineal Saturado** | satlin | ic5 | satlin |
| **Lineal Saturado Simétrico** | satlins | ic6 | satlins |
| **Sigmoidal Logarítmico** | logsig | ic7 | logsig |
| **Tangente Sigmoidal Hiperbólica** | tangsig | ic8 | tansig |
| **Competitiva** | **a=1 Neurona con n max**  **a=0 El resto de las neuronas** | ic9 | compet |

**PERCEPTRON**

**HISTORIA:** El Perceptrón dentro del campo de las redes neuronales tiene dos [acepciones (conceptualizaciones)](https://es.wikipedia.org/wiki/Acepci%C3%B3n). Puede referirse a un tipo de [red neuronal artificial](https://es.wikipedia.org/wiki/Red_neuronal_artificial) desarrollada por [Frank Rosenblatt](https://es.wikipedia.org/w/index.php?title=Frank_Rosenblatt&action=edit&redlink=1) y, dentro de ésta teoría emitida por Rosenblatt, también puede entenderse como la [neurona artificial](https://es.wikipedia.org/wiki/Neurona_artificial) o unidad básica de inferencia en forma de [discriminador lineal](https://es.wikipedia.org/w/index.php?title=Discriminador_lineal&action=edit&redlink=1), a partir de lo cual se desarrolla un algoritmo capaz de generar un criterio para seleccionar un sub-grupo a partir de un grupo de componentes más grande. La limitación de este algoritmo es que si dibujamos en un plot estos elementos, se deben poder separar con un hiperplano únicamente los elementos "deseados" discriminándolos (separándolos) de los "no deseados". El Perceptrón puede utilizarse con otros perceptrones u otro tipo de neurona artificial, para formar redes neuronales más complicadas.

**FORMULA MATEMATICA**: El perceptrón usa una matriz para representar las redes neuronales y es un discriminador terciario que traza su entrada x (un vector binario) a un único valor de salida f(x) (un solo valor binario) a través de dicha matriz.

f(x)={1 si w . x – u >

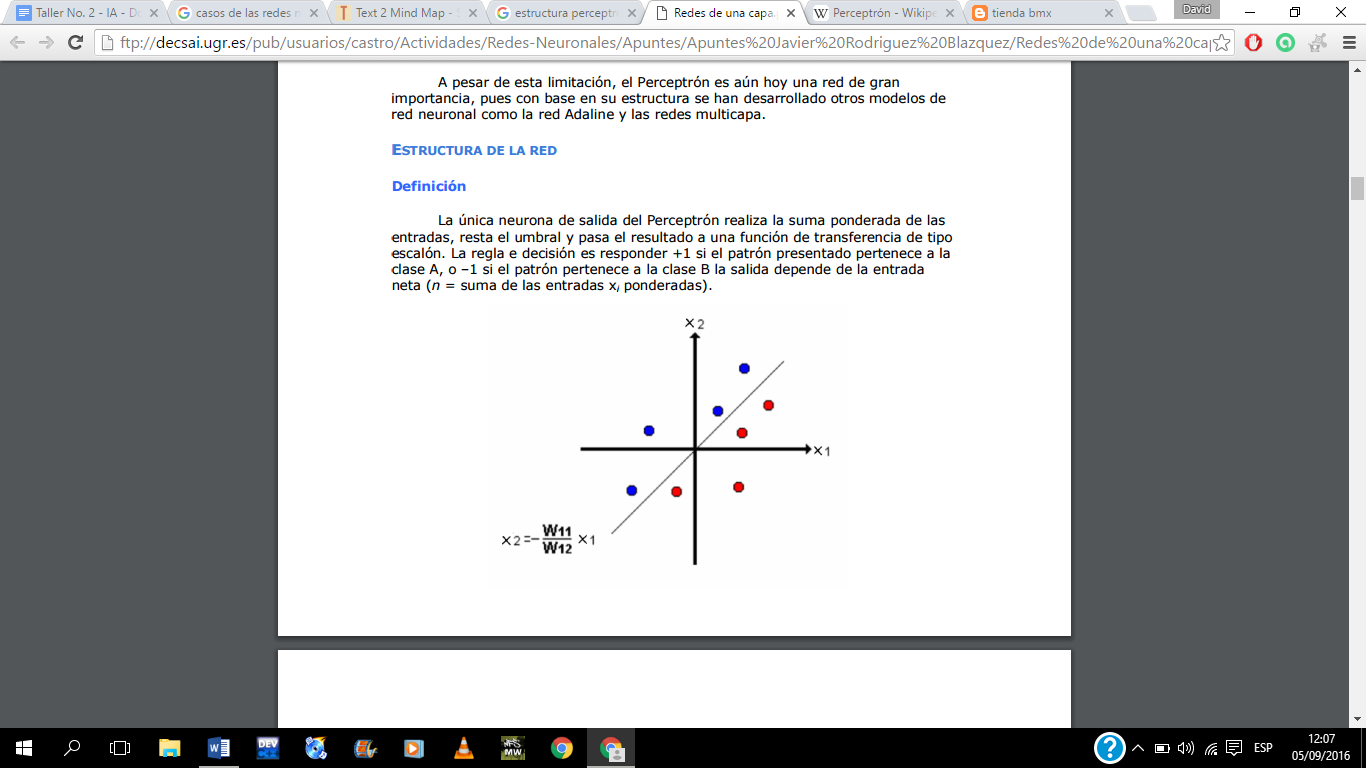
{0 en otro caso

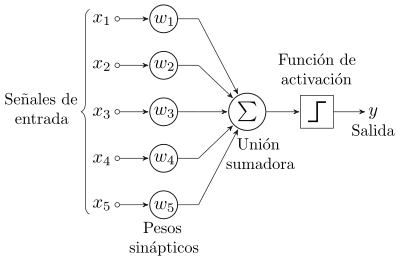
Donde w es un vector de pesos reales y **w . x**  es el producto escalar (que computa una suma ponderada). **u** es el 'umbral', el cual representa el grado de inhibición de la neurona, es un término constante que no depende del valor que tome la entrada.

El valor de f(x) (0 o 1) se usa para clasificar **x** como un caso positivo o un caso negativo, en el caso de un problema de clasificación binario. El umbral puede pensarse de como compensar la función de activación, o dando un nivel bajo de actividad a la neurona del rendimiento. La suma ponderada de las entradas debe producir un valor mayor que **u** para cambiar la neurona de estado 0 a 1

**ESTRUCTURA:** ESTRUCTURA DE LA RED

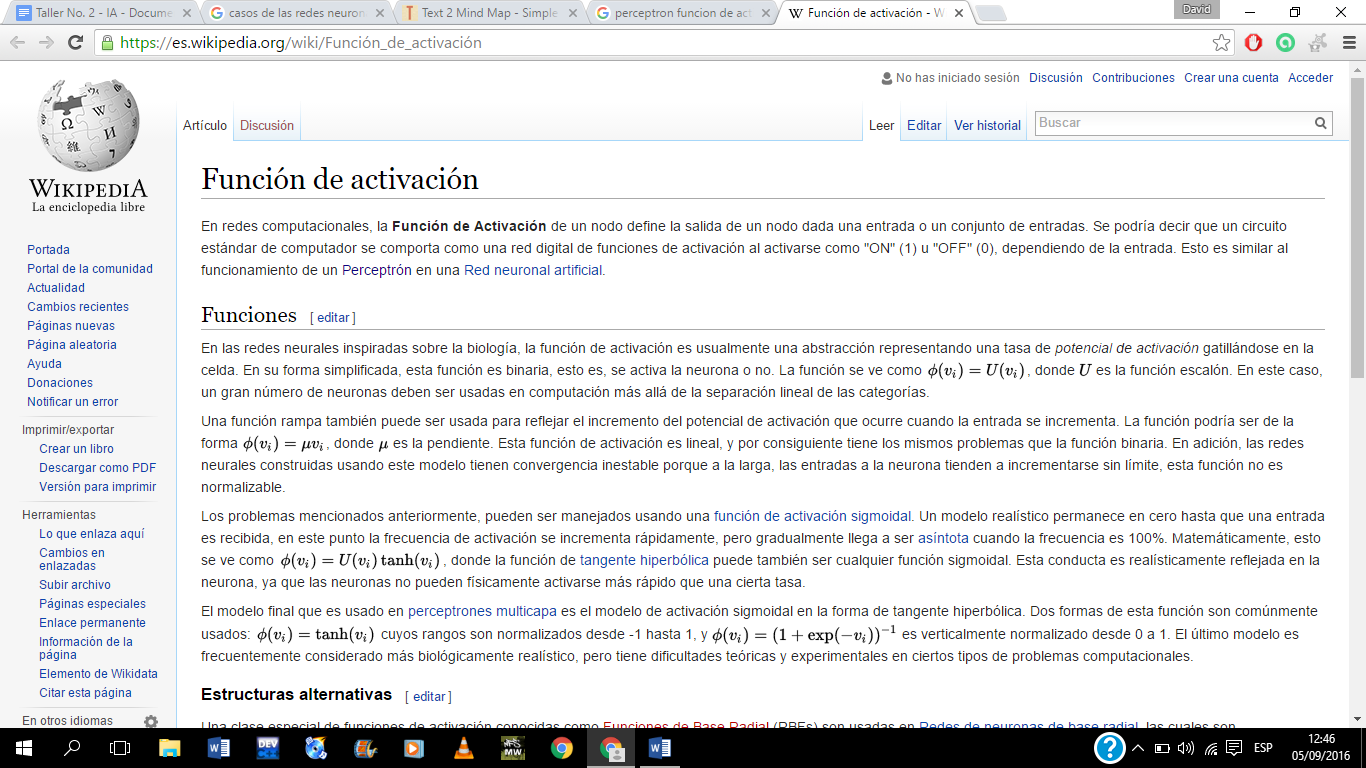
Definición La única neurona de salida del Perceptrón realiza la suma ponderada de las entradas, resta el umbral y pasa el resultado a una función de transferencia de tipo escalón. La regla e decisión es responder +1 si el patrón presentado pertenece a la clase A, o –1 si el patrón pertenece a la clase B la salida depende de la entrada neta (n = suma de las entradas xi ponderadas)





**PARA QUE SE USA:** se puede entrenar una red para formar el tiempo pasado de los verbos en inglés, leer texto en inglés y manuscrito. El Perceptrón multicapa (MLP) puede ser usado para la predicción de una serie de datos en el tiempo; tal a sido su éxito en la medición de la demanda de gas y electricidad, además, de la predicción de cambios en el valor de los instrumentos financieros.

NETtalk, es un Perceptrón que es capaz de transformar texto en inglés en sonido individual (representaciones fonéticas) y la pronunciación con la utilización de un sintetizador de voz; cuenta con aproximadamente 300 nodos de neuronas (siendo 80 en la capa oculta) y 20,000 conexiones individuales.

**FUNCION DE ACTIVACION:** El modelo final que es usado en perceptrones multicapa es el modelo de activación sigmoidal en la forma de tangente hiperbólica. Dos formas de esta función son comúnmente usados: ø (Vi) = tanh (Vi) cuyos rangos son normalizados desde -1 hasta 1, y  es verticalmente normalizado desde 0 a 1.

**ALGORITMO DEL PERCEPTRON:** Algoritmo de Aprendizaje

El algoritmo de aprendizaje es supervisado, en el cual, la regla es provista como un conjunto del comportamiento propio de la red:

ecuacion

Donde:

*pq*: Es la entrada de la red.

*tq*: Es la salida destino correspondiente.

La regla de aprendizaje es un procedimiento para modificar los pesos y los niveles de umbral de activación en una red neuronal.

La principal aportación de Rosenblatt fue el desarrollo de una regla de aprendizaje simple y automática aplicada al reconocimiento de patrones.

La red aprendía de sus errores y se inicializaba con valores aleatorios.

La regla de aprendizaje siempre converge a los pesos correctos de la red si es que existen los pesos que solucionen dicho problema.

El entrenamiento comienza asignándole valores iniciales pequeños en forma aleatoria a los parámetros de la red y , por medio de las siguientes ecuaciones:

Para el vector de pesos **W**:

ecuacion

Para el umbral **b**:

ecuacion

Para el error **e**:

ecuacion

**En donde:**

**W**: Es la matriz de pesos sinápticos.

**b**: Es el umbral de activación.

**p**: Es el patrón de análisis.

**e**: Es el error.

**t**: Es el valor objetivo relacionado con el patrón

**α**: Es la velocidad de aprendizaje.

El algoritmo de entrenamiento del perceptrón se encuentra dentro de los denominados algoritmos por corrección de errores. Este tipo de algoritmos ajustan los pesos de manera proporcional a la diferencia entre la salida actual proporcionada por la red y la salida objetivo, con el fin de minimizar el error producido por la red. Se puede demostrar que este método de entrenamiento converge siempre en un tiempo finito y con independencia de los pesos de partida, siempre que la función a representar sea linealmente separable. El principal problema de este método de entrenamiento es que cuando la función a representar no es linealmente separable el proceso de entrenamiento oscilará y nunca alcanzará la solución. Las funciones no separables linealmente no pueden ser representadas por un perceptrón.

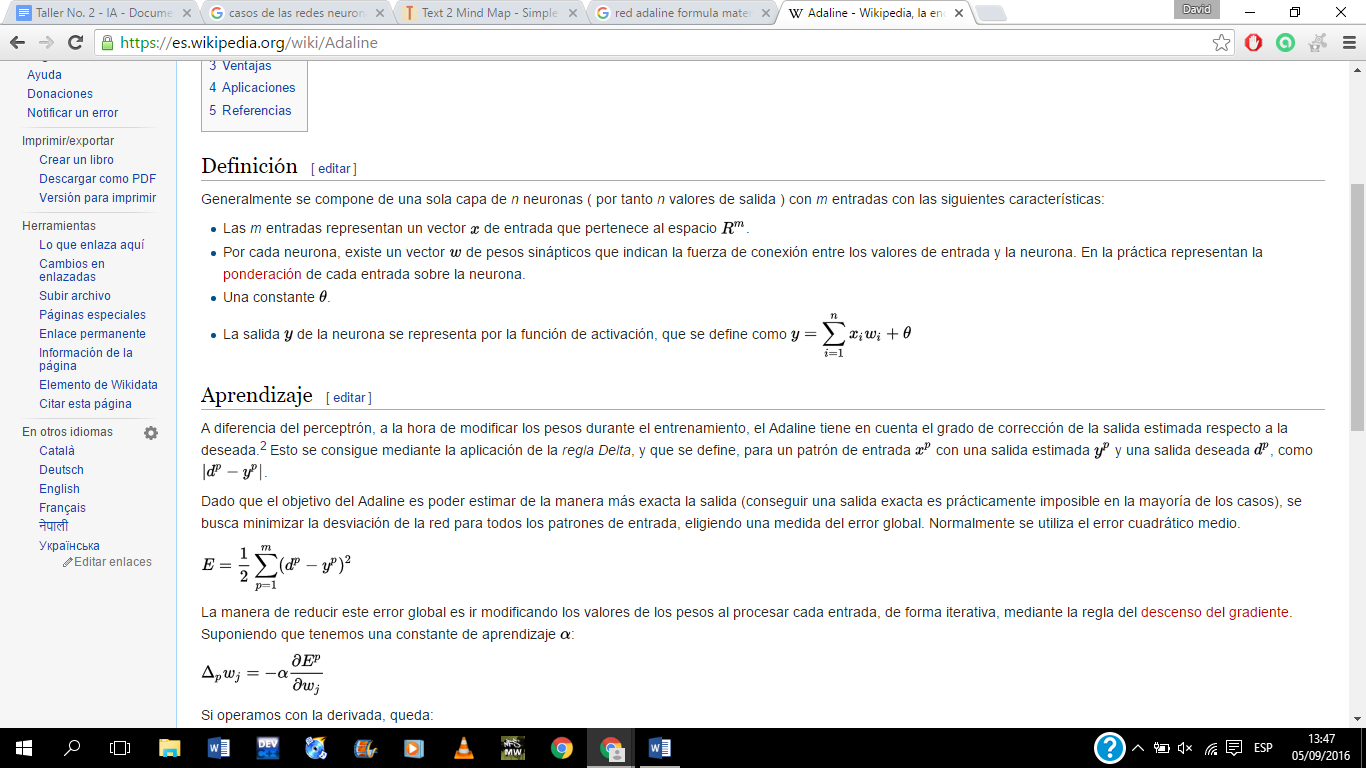
**ADALINE:**

**HISTORIA:** El **adaline** (de **ADA**ptative **LIN**ear **E**lement) es un tipo de [red neuronal artificial](https://es.wikipedia.org/wiki/Red_neuronal_artificial) desarrollada por el profesor [Bernie Widrow](https://es.wikipedia.org/w/index.php?title=Bernie_Widrow&action=edit&redlink=1" \o "Bernie Widrow (aún no redactado)) y su alumno [Ted Hoff](https://es.wikipedia.org/w/index.php?title=Ted_Hoff&action=edit&redlink=1) en la [Universidad de Stanford](https://es.wikipedia.org/wiki/Universidad_de_Stanford)en 1960.[1](https://es.wikipedia.org/wiki/Adaline#cite_note-Autor-1) El modelo está basado en la [Neurona de McCulloch-Pitts](https://es.wikipedia.org/wiki/Neurona_de_McCulloch-Pitts).

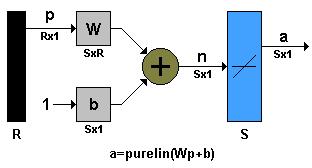
Otro de los modelos que tienen gran importancia es la red neuronal ADALINE. La topología de la red ADALINE es similar a la del perceptrón sólo que en este caso la función de salida de las neuronas es lineal. Dado que las señales de entrada pueden ser continuas, la red ADALINE es un dispositivo de entrada/salida analógica (continua) a diferencia del perceptrón que de acuerdo a lo dicho anteriormente es un dispositivo entrada/salida digital (binaria) La operación de una red ADALINE con n neuronas de entrada y m neuronas de salidas puede ser resumida de la siguiente manera:

**FORMULA MATEMATICA:** Generalmente se compone de una sola capa de *n* neuronas ( por tanto *n* valores de salida ) con *m* entradas con las siguientes características:

* Las *m* entradas representan un vector {\displaystyle x}**X** de entrada que pertenece al espacio {\displaystyle R^{m}}.
* Por cada neurona, existe un vector {\displaystyle w}**W** de pesos sinápticos que indican la fuerza de conexión entre los valores de entrada y la neurona. En la práctica representan la [ponderación](https://es.wikipedia.org/w/index.php?title=Ponderaci%C3%B3n&action=edit&redlink=1) de cada entrada sobre la neurona.
* Una constante {\displaystyle \theta }**0**.
* La salida {\displaystyle y}**Y** de la neurona se representa por la función de activación, que se define como {\displaystyle y=\sum \_{i=1}^{n}x\_{i}w\_{i}+\theta }



**ESTRUCTURA:** La estructura general de la red tipo Adaline puede visualizarse en la Fig. 2.11:



En donde:

*p*: Patrones de entrada

b: Umbrales de activación

*a*: Salida de la neurona

La salida de la red está dada por:

ecuacion

**PARA QUE SE USA:** En general, la Adaline es usada para ejecutar:

* Una aproximación lineal de un pequeño segmento de una superficie no lineal, la cual es generada por una función variable , En este caso, el bias es usualmente necesario, de aquí, *wp=1* .
* Filtrado lineal de datos (señales).
* Asociación de patrones, esto es, generación de m-elemento del vector de salida asociados con su respectivo p-elemento del vector de entrada.

**ALGORITMO DE APRENDIZAJE:** Algoritmo de Aprendizaje

1. Se aplica un patrón de entrada P.

2. Se obtiene la salida del Combinador Lineal Adaptable (ALC) y se calcula la diferencia con respecto a la deseada, es decir, el error.

3. Se actualizan los pesos.

4. Se repiten pasos 1 a 3 con todos los vectores de entrada.

Si el error es un valor aceptable, detenerse, si no repetir algoritmo.

La regla de aprendizaje de Widrow-Hoff o LMS (Least Mean Square), que utiliza la red Adaline para su entrenamiento, hace posible la realización del paso 3.

Por medio de las siguientes ecuaciones, se realiza la actualización de los parámetros de la red:

Para el vector de pesos ***W***

ecuacion

Para el umbral ***b***

ecuacion

Para el error ***e***

ecuacion

En donde se conoce como razón o tasa de aprendizaje, tal que, y 0 < *a* <= 1.

El cálculo de dicho parámetro se realiza mediante una matriz de correlación R:

ecuacion

Los eigenvalores li de la matriz de correlación serán útiles para la determinación del parámetro *a*, esto es:

ecuacion

**REDES DE RETROPROPAGACION**

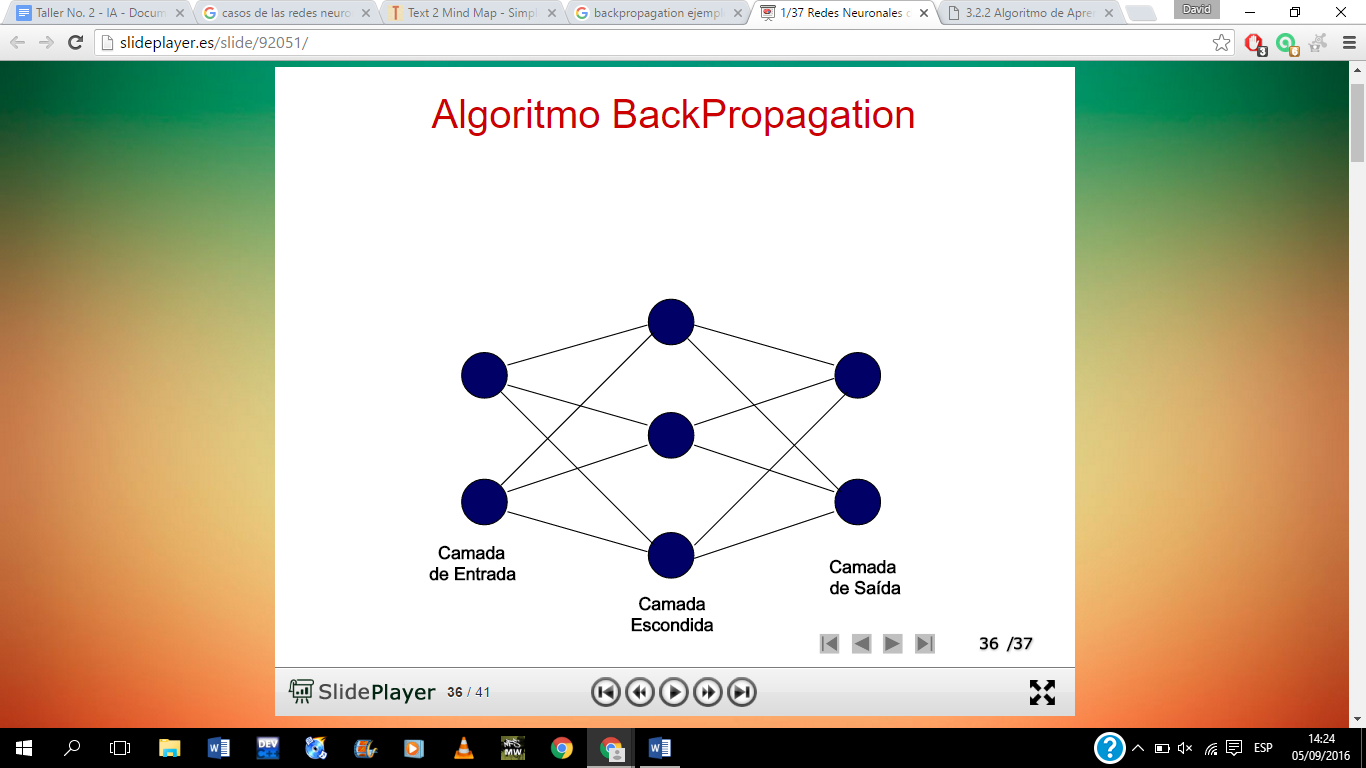
El método de *back-propagation* (o *entrenamiento hacia atrás*) es un sistema automático de *entrenamiento* de redes neuronales con *capas ocultas*, perfeccionado en la década de los 80.

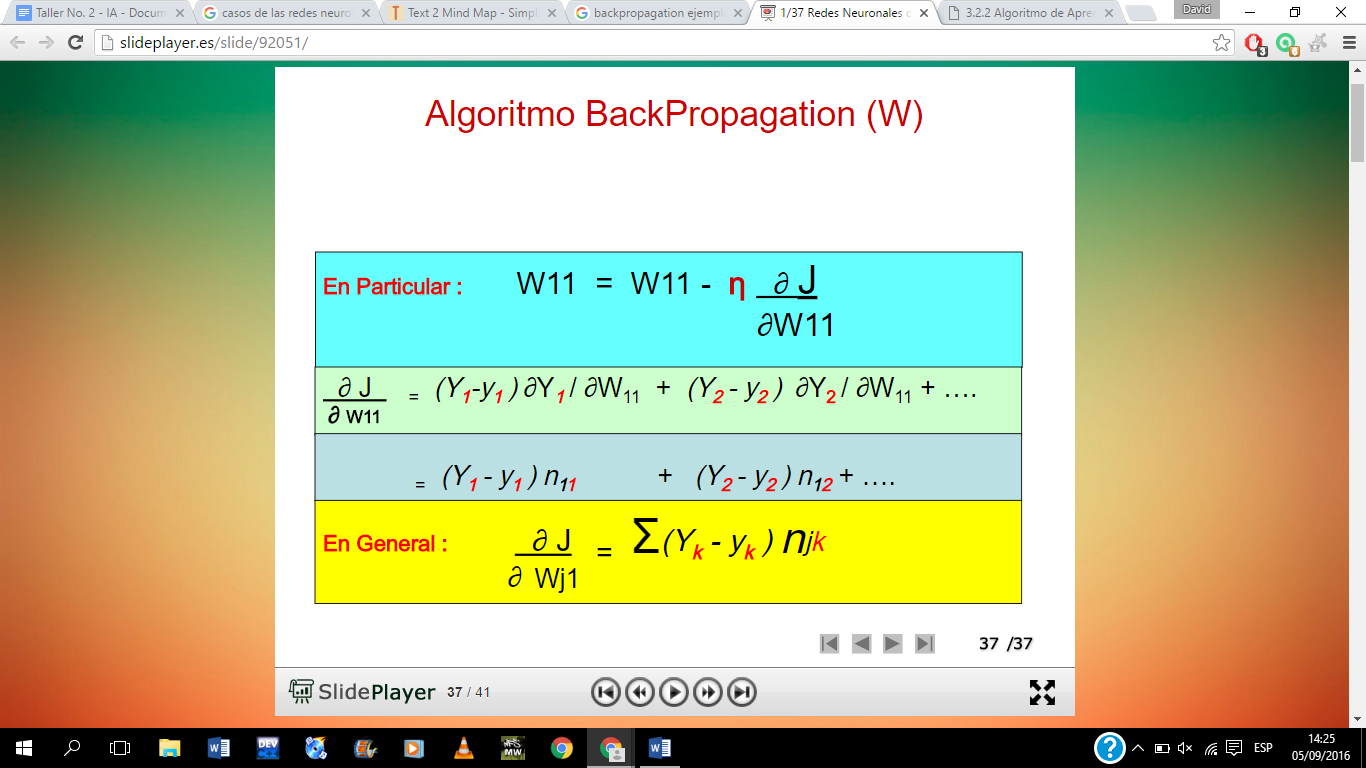
En este tipo de redes, el *problema* a la hora de *entrenarlas* estriba en que sólo conocemos la *salida* de la red y la *entrada*, de forma que no se pueden ajustar los pesos sinápticos asociados a las neuronas de las capas ocultas, ya que no podemos inferir a partir del estado de la capa de salida como tiene que ser el estado de las capas ocultas. 

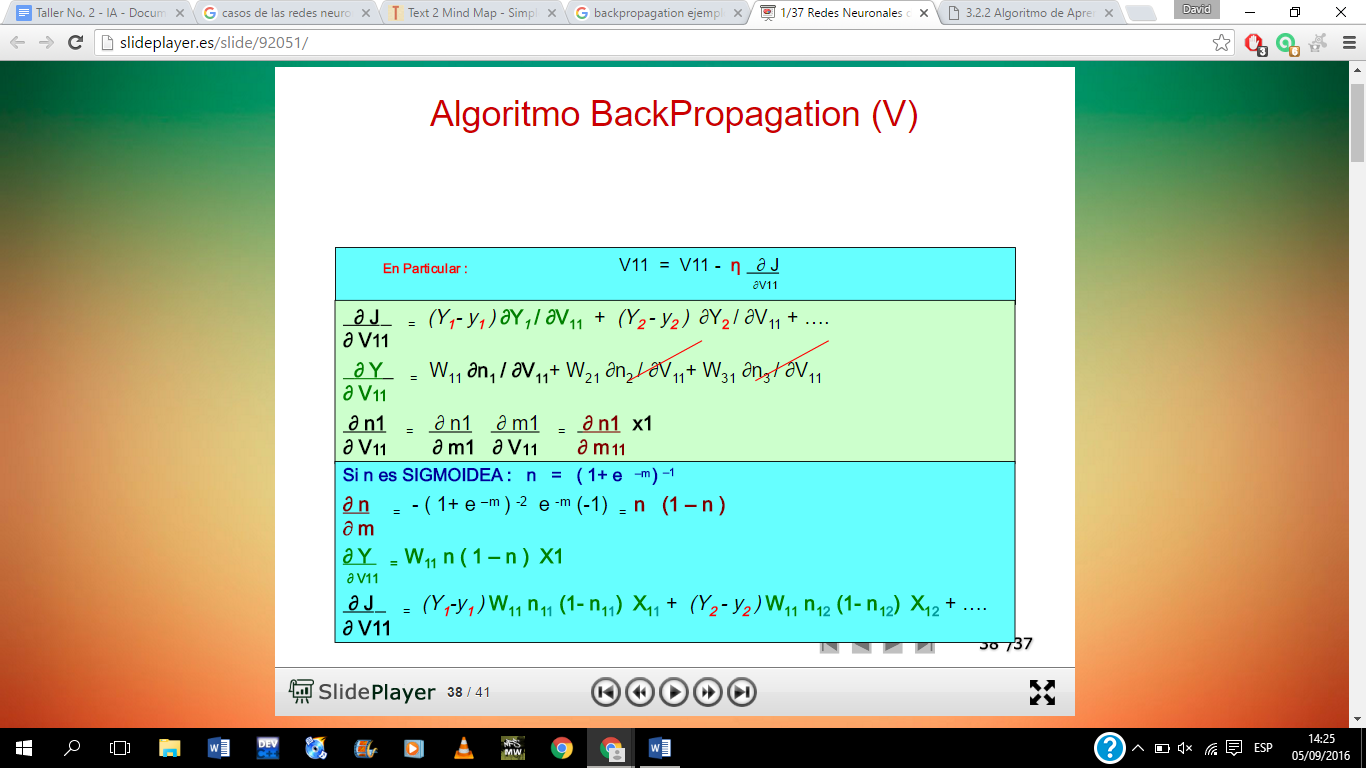
El sistema de entrenamiento mediante back-propagation consiste en:

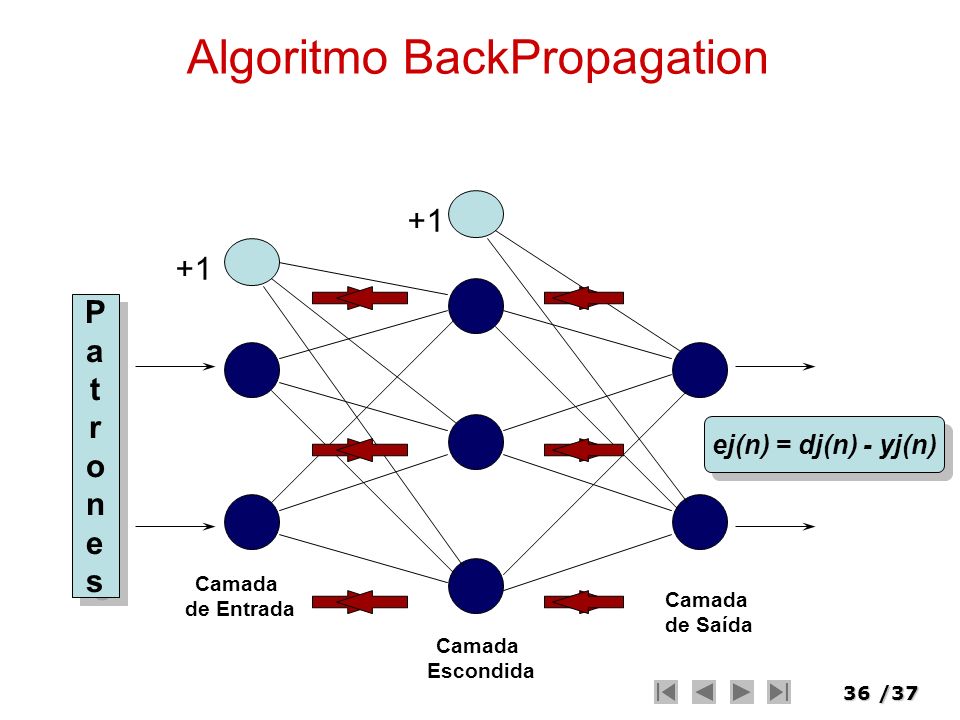
* Empezar con unos pesos sinápticos cualquiera (generalmente elegidos al azar).
* Introducir unos datos de entrada (en la capa de entradas) elegidos al azar entre los datos de entrada que se van a usar para el entrenamiento.
* Dejar que la red genere un vector de datos de salida (propagación hacia delante).
* Comparar la salida generada por al red con la salida deseada.
* La diferencia obtenida entre la salida generada y la deseada (denominada *error*) se usa para ajustar los pesos sinápticos de las neuronas de la capa de salidas.
* El error se *propaga hacia atrás* (back-propagation), hacia la capa de neuronas anterior, y se usa para ajustar los pesos sinápticos en esta capa.
* Se continua propagando el error hacia atrás y ajustando los pesos hasta que se alcance la capa de entradas.

 Este proceso se repetirá con los diferentes datos de entrenamiento.  
  
El método por el cual el error es propagado hacia atrás fue descubierto por D.E. Rumelhart, G.E. Hinton y R.J. Williams en 1986, y casi al mismo al tiempo por Parker (1985) y Le Cun (1985).

{\displaystyle y=\sum \_{i=1}^{n}x\_{i}w\_{i}+\theta }







**TEOREMA DE KOLMOGOROV:** El **teorema de Kolmogórov–Arnold–Moser** o **teorema KAM** es un resultado de [sistemas dinámicos](https://es.wikipedia.org/wiki/Sistemas_din%C3%A1micos) sobre la persistencia de [movimientos cuasiperiódicos](https://es.wikipedia.org/wiki/Movimiento_cuasiperi%C3%B3dico). Este teorema resuelve parcialmente el [problema de los divisores pequeños](https://es.wikipedia.org/w/index.php?title=Problema_de_los_divisores_peque%C3%B1os&action=edit&redlink=1) (que origina problemas de convergencia en sistemas con múltiples frecuencias). El teorema explica como se modifica el aspecto de las trayectorias de un sistema integrable bajo pequeñas perturbaciones.

**Bibliografias:**

(http://ingenieria.udea.edu.co/investigacion/mecatronica/mectronics/redes.htm)

(http://redes-neuronales.wikidot.com/definicion-ventajas-desventajas)

(https://www.frro.utn.edu.ar/repositorio/catedras/quimica/5\_anio/orientadora1/monograias/matich-redesneuronales.pdf)

(http://halweb.uc3m.es/esp/Personal/personas/jmmarin/esp/DM/tema3dm.pdf)