

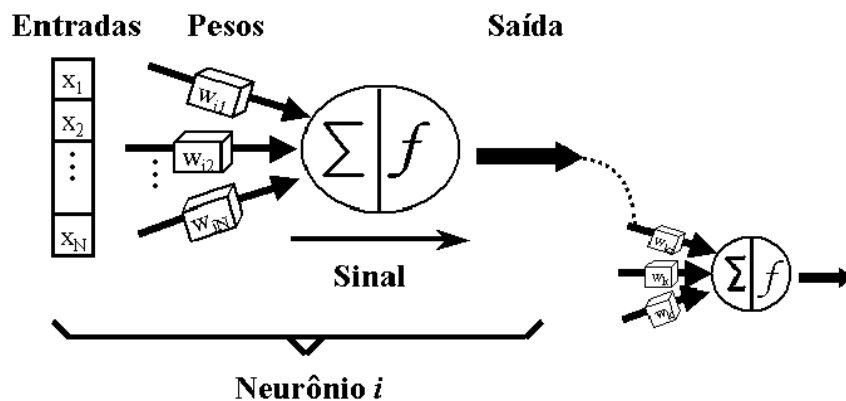
Redes Neuronales

Una red neuronal artificial (RNA) es un modelo computacional inspirado en redes neuronales biológicas que puede ser consideradas como un sistema de procesamiento de información con características como aprendizaje a través de ejemplos adaptabilidad, robustez, capacidad de generalización y tolerancia a fallas.

La RNA puede ser definida como una estructura distribuida, de procesamiento paralelo, formada de neuronas artificiales (llamados también elementos de procesamiento), interconectados por un gran numero de conexiones (sinapsis), los cuales son usados para almacenar conocimiento que esta disponible para poder ser usado.

Estructura de la Neurona Artificial

Una neurona artificial es una unidad de procesamiento de información de redes neuronales. El modelo de neurona mas conocido es de McCulloch-Pitts.



Puede observarse que N señales de entrada son representadas por las variables x_1, x_2, \dots, x_N las cuales están asociadas a pesos que son representados por las variables w_{ji} los cuales determinan el nivel de influencia de la neurona j para la neurona i .

Existen dos esta de procesamiento para cada neurona: suma y activación.

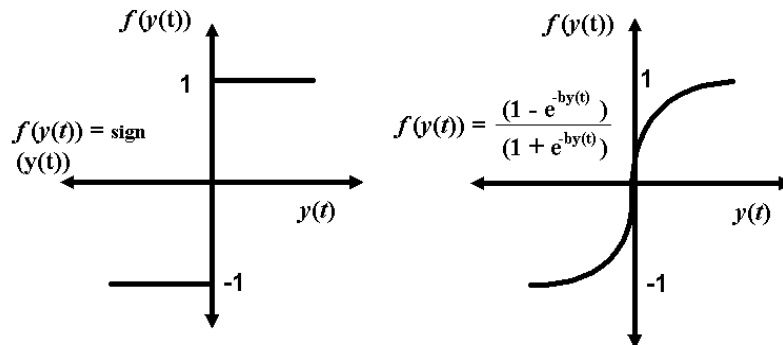
En la primera etapa, las señales de entrada x_j y los pesos w_{ij} son combinadas por el sumatoria:

$$y_i = \sum_{j=1}^N w_{ij}x_j$$

Donde y_i es llamado de estado interno de la neurona i . En la segunda etapa, la salida de la neurona es generada a través de la aplicación de una función llamada función de activación.

$$x_i = f(y_i)$$

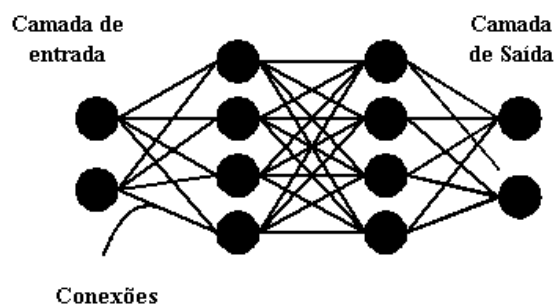
donde la salida de la neurona es representado por x_i y f corresponde a la función de activación aplicada al estado interno de la neurona, que tiene como objetivo limitar el nivel de activación de entre $[-1, 1]$ o $[0, 1]$, en el caso de x_i sea un valor continuo y si x_i es discreto entonces el puede ser : $\{-1, 1\}$ o $\{0, 1\}$



Existen varios tipos de función de activación. La figura muestra dos funciones de activación más usadas: la función de grado y la tangente hiperbólica. Como se vio en la primera figura la salida de una neurona puede ser la entrada de otra. Generalmente, una red neuronal se forma por muchas neuronas de alguna forma acoplados.

3.2.2 Arquitectura de Red

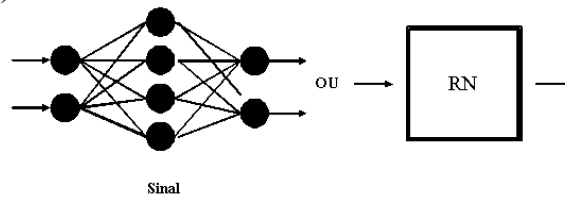
La definición de arquitectura es un punto importante en el modelaje de una red neuronal, por que ella restringe un tipo de problema que puede ser tratado. Por ejemplo las redes de una capa. Una red también puede estar formada por múltiples capas, las que pueden ser clasificadas en tres grupos: capa de entrada, capas intermediarias u ocultas y capas de salida



Basado en flujo de las señales, las redes neuronales también pueden ser clasificadas en dos tipos: *FeedForward* y redes *Recurrentes*.

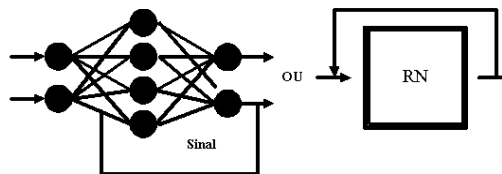
- Redes *FeedForward*

Como podemos ver la estructura de una red *FeedForward* consiste en capas de neuronas donde la salida de una neurona de una capa, alimenta todas las neuronas de la capa siguiente. El aspecto fundamental de esta estructura es que no existen las uniones de retroalimentación. La red *MultiLayer Perceptron* (MLP) de un tipo de red *feedforward* (D. Rumelhart, 1986).



- Redes *Recurrentes*

Redes recurrentes son aquellas que poseen conexiones de realimentación, como es visto en la figura, las cuales proporcionan un comportamiento dinámico. El modelo de Hopfield es un ejemplo de red neuronal recurrente y será presentado más adelante.



En general los siguientes parámetros son importantes para definir la arquitectura de una red neural: número de capas, número de neuronas en cada capa y tipo de conexión entre dos neuronas, que definen la red de *feedforward* o *Recurrentes*.

3.2.3 Algoritmos de Aprendizaje de un RNA

Una propiedad importante de las redes neuronales es la habilidad de aprender a partir de su ambiente. Eso es realizado a través de un proceso interactivo de ajustes aplicado a sus pesos de conexión entre dos neuronas, denominados entrenamiento. Existen muchos algoritmos de aprendizaje. Cada uno sirve para determinar redes neuronales. Entre los principales se tienen:

- **Aprendizaje por Corrección de Error:** Algoritmo muy conocido basado en la regla Delta, que busca minimizar la función de error

usando un gradiente descendente. Este es el principio usado no algoritmo *BackPropagation*, muy utilizado para el entrenamiento de redes de múltiples capas como la *Multilayer-Perceptron* (MPL)(James A. Freeman, 1991);

- **Aprendizaje Competitivo:** La cual dos neuronas de una capa compiten entre si por el privilegio de permanecer activos, tal que una neurona con mayor actividad será el único que participará del proceso de aprendizaje. Es usado en mapas de Kohonen (Kohonen, 1988) y en redes ART (Gail A. Carpenter, 1992);
- **Aprendizaje Hebbiano:** Son dos neuronas que están simultáneamente activos a conexiones entre ellos que debe ser fortalecida caso contrario será debilitada (Hebb,1949) utilizada en el Modelo de Hopfield (Hopfield, 1982);
- **Aprendizaje de Boltzmann:** Es una regla de aprendizaje estocástico obtenido a partir de principios de teórico de información y de termodinámica. El objetivo de aprendizaje de Boltzmann es ajustar los pesos de conexión de tal forma que el estado de las unidades visibles satisfaga una distribución de probabilidades deseada en particular (D. Ackley, 1985);

Otro factor importante es la manera por la cual una red neuronal se relaciona con el ambiente .

A partir de ese concepto existen los siguientes paradigmas de aprendizaje:

- **Aprendizaje Supervisado:** Se utiliza un agente externo que indica a la red la respuesta deseada para el patrón de entrada;
- **Refuerzo:** Es una variante de aprendizaje supervisado a la cual se informa a la red solamente una crítica de corrección de salida de red y no la respuesta correcta en si;
- **Aprendizaje No Supervisado (auto-organización):** No existe un agente externo indicando la respuesta deseada para los patrones de entrada. Este tipo de aprendizaje es utilizado en los modelos de Mapas de Kohonen (Kohonen, 1988), redes ART1, ART2 (Gail A. Carpenter, 1992) (G. Carpenter, 1987).