1.2 Antecedentes.

En 1936 Alan Turing, fue el primero en estudiar el cerebro como una forma de ver el mundo de la computación. Sin embargo, los primeros teóricos que concibieron los fundamentos de la computación neuronal fueron Warren McCulloch, un neurofisiólogo, y Walter Pitts, un matemático, quienes, en 1943, lanzaron una teoría acerca de la forma de trabajar de las neuronas […]. Ellos modelaron una red neuronal simple mediante circuitos eléctricos. (Matich, 2001)

Casi 2 décadas después, en año de 1957, Frank Rosenblatt. Comenzó el desarrollo del Perceptron. Esta es la red neuronal más antigua; utilizándose hoy en día para aplicación como identificador de patrones. Este modelo era capaz de generalizar, es decir, después de haber aprendido una serie de patrones podía reconocer otros similares, aunque no se le hubiesen presentado en el entrenamiento. Sin embargo, tenía una serie de limitaciones, por ejemplo, su incapacidad para resolver el problema de la función OR-exclusiva y, en general, era incapaz de clasificar clases no separables linealmente. (Matich, 2001)

Los sistemas nerviosos de los seres vivos se organizan en forma de redes complejas con nodos (neuronas) y conexiones (sinapsis). Dichas redes son capaces de realizar tareas de percepción y detección de patrones de manera sumamente efectiva y eficiente (Rojas, 1996; London & Häusser, 2005). Por ello, han sido objeto de estudio en numerosas investigaciones y han servido de inspiración para el desarrollo de algoritmos y modelos computacionales de aprendizaje. Ejemplo de ello son las diversas aplicaciones en Visión Computacional, tal como detección de cáncer (Esteva et al., 2017; Mohsen et al., 2017) y manejo de vehículos autónomos (Bechtel et al., 2018); y en reconocimiento del habla, tal como reconocimiento de emociones (Chernykh et al., 2017); por solo mencionar algunas.

1.Redes Neuronales Artificiales.

1.1 Concepto.

Según (Gurney, 1997),

> Una red neuronal es un conjunto interconectado de elementos, unidades o nodos de procesamiento simples, cuya la funcionalidad se basa libremente en la neurona animal. La capacidad de procesamiento de la red se almacena en Fuerzas de conexión entrelazadas, o pesos, obtenidos por un proceso de adaptación o aprendizaje de un conjunto de patrones de entrenamiento.

(Dinamarca, 2018) define a una red neuronal como:

> Un conjunto de funciones $\{f^{(1)}, \dots, f^{(k)}\}$, conectadas comúnmente a la salida de cada una a las entradas de otras diferentes. De esta manera, las redes neuronales artificiales no son mas que redes de funciones, típicamente representadas mediante la composición de varias funciones $f(x) = f^{(k)}(\dots(f^{(1)}(x)))$. (p. 18)

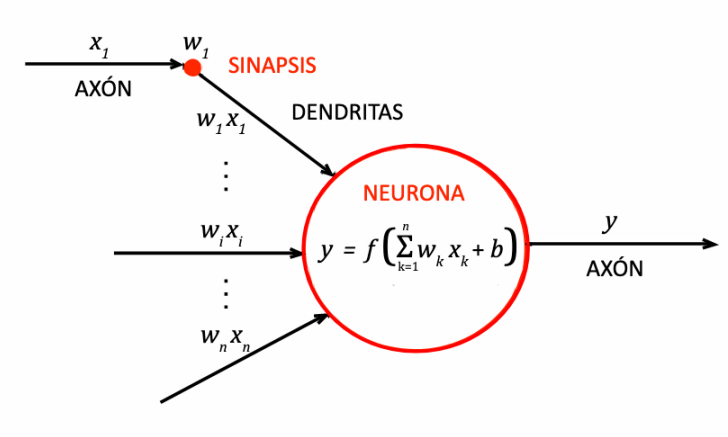

Fig. 1. Estructura lógica de una neurona. (Dinamarca, A)

(Gerhenson, s.f.) define a la red neuronal simple (según la [Fig.1]) como:

> Entradas (como sinapsis), que se multiplican por pesos (fuerza de las respectivas señales), y luego calculado por una función matemática que determina la activación de la neurona. Otra función (que puede ser la identidad) calcula la salida de la neurona artificial (a veces en dependencia de un cierto límite).

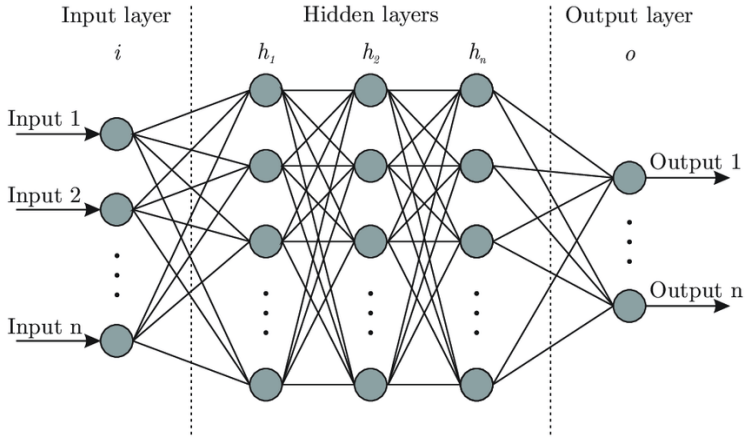

Fig. 2. Estructura de una red neuronal profunda. (Bre, 2017)

La estructura detallada en la [Fig.2] es detallada por (Dinamarca, 2018):

> La capa mas a la izquierda en esta red se llama capa de entrada, y las neuronas dentro de la capa se denominan neuronas de entrada. La capa mas a la derecha o de salida contiene las neuronas de salida o, como en este caso, una única neurona de salida. Las capas intermedias se llaman capas ocultas, ya que las neuronas en estas capas no son ni entradas ni salidas. Por otro lado, los diferentes modelos de redes neuronales difieren principalmente en las funciones de activación utilizadas, el patrón de interconexión, e inclusive el tiempo de trasmisión de la información.
> (p. 18)

1.2 Mecanismos de aprendizaje.

(Matich, 2001) define el concepto de aprendizaje de una red neuronal como "El proceso por el cual una red neuronal modifica sus pesos en respuesta a una información de entrada. Los cambios que se producen durante el mismo se reducen a la destrucción, modificación y creación de conexiones entre las neuronas".

De igual manera (Matich, 2001) menciona y define la existencia de 2 tipos de mecanismos que se pueden aplicar para el entrenamiento de una red neuronal.

1. **Aprendizaje Supervisado:**
   El proceso de aprendizaje se realiza mediante un entrenamiento controlado por un agente externo (supervisor, maestro) que determina la respuesta que debería generar la red a partir de una entrada determinada. El supervisor controla la salida de la red y en caso de que ésta no coincida con la deseada, se procederá a modificar los pesos de las conexiones, con el fin de conseguir que la salida obtenida se aproxime a la deseada. En este tipo de aprendizaje se suelen considerar, a su vez, tres formas de llevarlo a cabo, que dan lugar a los siguientes aprendizajes supervisados:
   - **Aprendizaje por corrección de error.**
     Consiste en ajustar los pesos de las conexiones de la red en función de la diferencia entre los valores deseados y los obtenidos a la salida de la red, es decir, en función del error cometido en la salida. Algunos ejemplos de estos algoritmos son:
     - Regla de aprendizaje de perceptron.
     - Regla de aprendizaje Delta o Regla del mínimo error cuadrado (LMS Error).
     - Regla de aprendizaje de propagación hacia atrás (Backpropagation o LMS Error multicapa)
   - **Aprendizaje por refuerzo.**
     Se basa en la idea de no disponer de un ejemplo completo del comportamiento deseado, es decir, de no indicar durante el entrenamiento exactamente la salida que se desea que proporcione la red ante una determinada entrada. […] La función del supervisor se reduce a indicar mediante una señal de refuerzo si la salida obtenida en la red se ajusta a la deseada (éxito = +1 o fracaso = -1), y en función de ello se ajustan los pesos basándose en un mecanismo de probabilidades.
   - **Aprendizaje estocástico.**
     Consiste básicamente en realizar cambios aleatorios en los valores de los pesos de las conexiones de la red y evaluar su efecto a partir del objetivo deseado y de distribuciones de probabilidad.
2. **Aprendizaje no supervisado.**
   No requieren influencia externa para ajustar los pesos de las conexiones entre sus neuronas. La red no recibe ninguna información por parte del entorno que le indique si la salida generada en respuesta a una determinada entrada es o no correcta. Estas redes deben encontrar las características, regularidades, correlaciones o categorías que se puedan establecer entre los datos que se presenten en su entrada. […] En algunos casos, la salida representa el grado de familiaridad o similitud entre la información que se le está presentando en la entrada y las informaciones que se le han mostrado hasta entonces (en el pasado). En otro caso, podría realizar una clusterización (clustering) o establecimiento de categorías, indicando la red a la salida a qué categoría pertenece la información presentada a la entrada, siendo la propia red quien debe encontrar las categorías apropiadas a partir de las correlaciones entre las informaciones presentadas. En cuanto a

los algoritmos de aprendizaje no supervisado, en general se suelen considerar dos tipos, que dan lugar a los siguientes aprendizajes:

1. **Aprendizaje hebbiano.**
   Pretende medir la familiaridad o extraer características de los datos de entrada. El fundamento es una suposición bastante simple: si dos neuronas N_iy N_j toman el mismo estado simultáneamente (ambas activas o ambas inactivas), el peso de la conexión entre ambas se incrementa.
2. **Aprendizaje competitivo y comparativo.**
   Se orienta a la clusterización o clasificación de los datos de entrada. Como característica principal del aprendizaje competitivo se puede decir que, si un patrón nuevo se determina que pertenece a una clase reconocida previamente, entonces la inclusión de este nuevo patrón a esta clase matizará la representación de la misma. Si el patrón de entrada se determinó que no pertenece a ninguna de las clases reconocidas anteriormente, entonces la estructura y los pesos de la red neuronal serán ajustados para reconocer la nueva clase.

1.3 Tipos de funciones de activación para la neurona artificial.

Las funciones de activación se utilizan especialmente en redes neuronales artificiales para transformar una señal de entrada en una señal de salida que a su vez se alimenta como entrada a la siguiente capa de la pila. En una red neuronal artificial, calculamos la suma de productos de las entradas y sus correspondientes pesos y finalmente aplicamos una función de activación para obtener la salida de esa capa en particular y suministrarla como entrada a la siguiente capa. (Siddharth & Sharma, 2020)

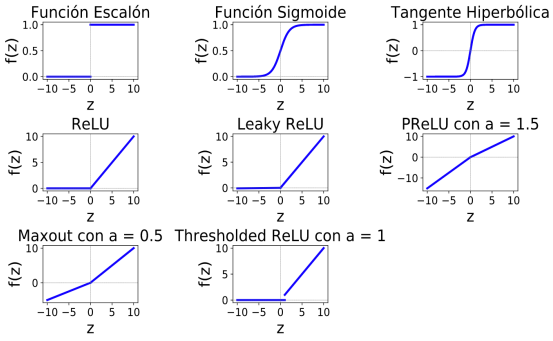

Fig. 3. Graficas de los comportamientos de las funciones de activación mas populares. (Dinamarca, 2018)

De igual manera (Siddharth & Sharma, 2020) definen algunas de las funciones presentes en la [fig.3] y sus propiedades, señalando lo siguiente:

- **Función Sigmoide.**
  Es la función de activación más utilizada ya que es una función no lineal. La función sigmoidea transforma los valores en el rango de 0 a 1. Se puede definir como:

$$f(x) = \frac{1}{1 + e^{-x}}$$

Además, la función sigmoidea no es simétrica con respecto a cero, lo que significa que los signos de todos los valores de salida de las neuronas serán las mismas. Este problema se puede mejorar escalando la función sigmoidea.

- **Función tanh (tangente hiperbólica):**
  Es similar a la función sigmoide pero es simétrica alrededor de el origen. Esto da como resultado diferentes signos de salidas de capas anteriores que se alimentarán como entrada a la siguiente capa. Se puede definir como:

$$f(x) = 2 * sigmoid(2x) - 1$$

  En comparación con el sigmoide función el gradiente de la función tanh es más empinado. Tanh es preferible a la función sigmoidea ya que tiene gradientes que no están restringidas a variar en una determinada dirección y también, es cero centrado.

- **Función ReLU:**
  ReLU significa unidad de revestimiento rectificado y no es lineal. Es una función de activación que es ampliamente utilizada en la red neuronal. La ventaja de usar la función ReLU es que todos los las neuronas no se activan al mismo tiempo. Esto implica que una neurona se desactivará sólo cuando la salida de la transformación lineal es cero. Puede ser descrita matemáticamente como:

$$f(x) = \max(0, x)$$

  ReLU es más eficiente que otras funciones porque como todas las neuronas no se activan al mismo tiempo, sino una cierto número de neuronas se activan a la vez. En algunos casos, el valor de gradiente es cero, por lo que los pesos y sesgos no se actualizan durante el paso de retropropagación en el entrenamiento de la red neuronal.

- **Función Leaky ReLU (ReLU con Fuga).**
  Leaky ReLU es una versión improvisada de la función ReLU donde para valores negativos de x, en lugar de definir el ReLU valor de las funciones como cero, se define como extremadamente pequeño componente lineal de x. Se puede expresar matemáticamente como:

$$f(x) = \begin{cases} 0.01x, & x < 0 \\ x, & x \geq 0 \end{cases}$$

- **Función PReLU (ReLU Parametrizada).**
  También es una variante de Unidad Lineal Rectificada con mejor rendimiento y una ligera variación. Resuelve el problema del gradiente de ReLU que se vuelve cero para valores negativos de x introduciendo un nuevo parámetro de la parte negativa de la función, es decir, Pendiente. Se expresa como:

$$f(x) = \begin{cases} x, & x < 0 \\ ax, & x \geq 0 \end{cases}$$

- **Función SoftMax:**
  La función Softmax es una combinación de múltiples sigmoides funciones Como sabemos que una función sigmoidea devuelve valores en el rango de 0 a 1, estos pueden ser tratados como probabilidades de los puntos de datos de una clase en particular. Función Softmax a diferencia de las funciones sigmoideas que se utilizan para la clasificación binaria, se puede utilizar para problemas de clasificación multiclase. La función, por cada punto de datos de todas las clases individuales, devuelve la probabilidad. Se puede expresar como:

$$\sigma(z)_j = \frac{e^{z_j}}{\sum_{k=1}^{K} e^{z_k}} \; para\; j = 1, \dots, K.$$

  Cuando se construye una red o modelo para múltiples clasificación de clase, entonces la capa de salida del red tendrá el mismo número de neuronas que la número de clases en el destino. (p. 312-314)

Bibliography

Asanza, W. R., & Olivo, B. M. (2018). *Redes neuronales artificiales aplicadas al reconocimiento de patrones.* Editorial UTMACH.

Matich, D. J. (04 de 2001). *Redes Neuronales: Conceptos Básicos y Aplicaciones.* Facultad Regional Rosario: https://www.frro.utn.edu.ar/repositorio/catedras/quimica/5_anio/orientadora1/monograias/matich-redesneuronales.pdf

Gurney, K. (1997). *An introduction to neural networks.* UCL Press.

Siddharth, S., & Sharma, S. (2020). Activation Functions in Neural Networks. *International Journal of Engineering Applied Sciences and Technology, 4*(12), 310-316.

3. Esteva, A., Kuprel, B., Novoa, R. A., Ko, J., Swetter, S. M., Blau, H. M. y Thrun, S. (2017). *Dermatologist-level classification of skin cancer with deep neural networks*. Nature, 542 (7639), 115–118. DOI: 10.1038/nature21056.

4. Rojas, R. (1996). *Neural Networks: A Systematic Introduction [online]*. Recuperado de https://page.mi.fu-berlin.de/rojas/neural/neuron.pdf

5. London, Mausser, M. (2005). *Dendritic Computation. Annual Review of Neuroscience*, 28 (1), 503-532. DOI: 10.1146/annurev.neuro.28.061604.135703

6. Mohsen, H., El-Dahshan, E. A., El-Horbaty, E. M. y Salem, A. M. (2017). Classification using deep learning neural networks for brain tumors. Future Computing and Informatics Journal. DOI: 10.1016/j.fcij.2017.12.001.

7. Bre, F. (2017). *Artificial neural network architecture [Imagen]. ResearchGate.* *https://www.researchgate.net/figure/Artificial-neural-network-architecture-ANN-i-h-1-h-2-h-n-o_fig1_321259051*

8. Dinamarca, A. (2018). *Aprendizaje y Análisis de Redes Neuronales Artificiales Profundas* [Tesis de Obtención de grado]. Repositorio de la Universidad Nacional de Cuyo: https://bdigital.uncu.edu.ar/objetos_digitales/13989/dinamarca-agustina-tesina.pdf

9. Gerhenson, C. (s.f.). *Artificial Neural Networks for Beginners.* Repositorio de la Universidad Veracruzana https://www.uv.mx/mia/files/2012/10/Artificial-Neural-Networks-for-Beginners.pdf