Redes Neuronales Artificiales

**González Aguayo Víctor Manuel**

**Programación Orientada a Objetos y Eventos**

**Dr. Luis Carlos Padierna García**

PeriodoEnero-Junio 2020

**Introducción:**

Las Redes Neuronales Artificiales (ANN) son un modelo computacional inspirado en el comportamiento observado con las redes neuronales biológicas​. Consiste en un conjunto de unidades o nodos, conectados todos entre sí para transmitirse señales o valores de pesos asignados a cada nodo. Estas señales van desde las neuronas de entrada y pasan recorriendo todos los nodos mediante una function de activacion que le va arrojando el valor a la siguiente neurona o capa neuronal. Esto ultimo depende totalmente de la arquitectura de red neuronal que se tome, unas con más parecido al funcionamiento biológico que otras, en este Proyecto se optó por un funcionamiento más parecida biologicamente mediante una arquitectura llamada Feed Forward o Propagación Hacia Adelante.

Las aplicaciones de ANN son muy escalables y diversas, ya que se pueden llevar a entrenar para tener reconocimiento de patrones para imágenes, fenómenos estadísticos, optimización o hasta meramente artísticos, esto perteneciente a la inteligencia artificial y áreas del desarrollo computacional denominado Machine Learning o Aprendizaje de Máquina.

**Antecedentes**:

En un principio los esfuerzos estuvieron dirigidos a la obtención de autómatas, en el sentido de máquinas que realizaran, con más o menos éxito, alguna función típica de los seres humanos.

Las primeras explicaciones teóricas sobre el cerebro y el pensamiento ya fueron dadas ya por **Platón** (427-347 a.C.) y **Aristóteles** (348-422 a.C.). Las mismas ideas también las mantuvo **Descartes** (1569-1650) y los filósofos empiristas del siglo XVIII. La clase de las llamadas máquinas cibernéticas, a la cual la computación neuronal pertenece, tiene más historia de la que se cree.

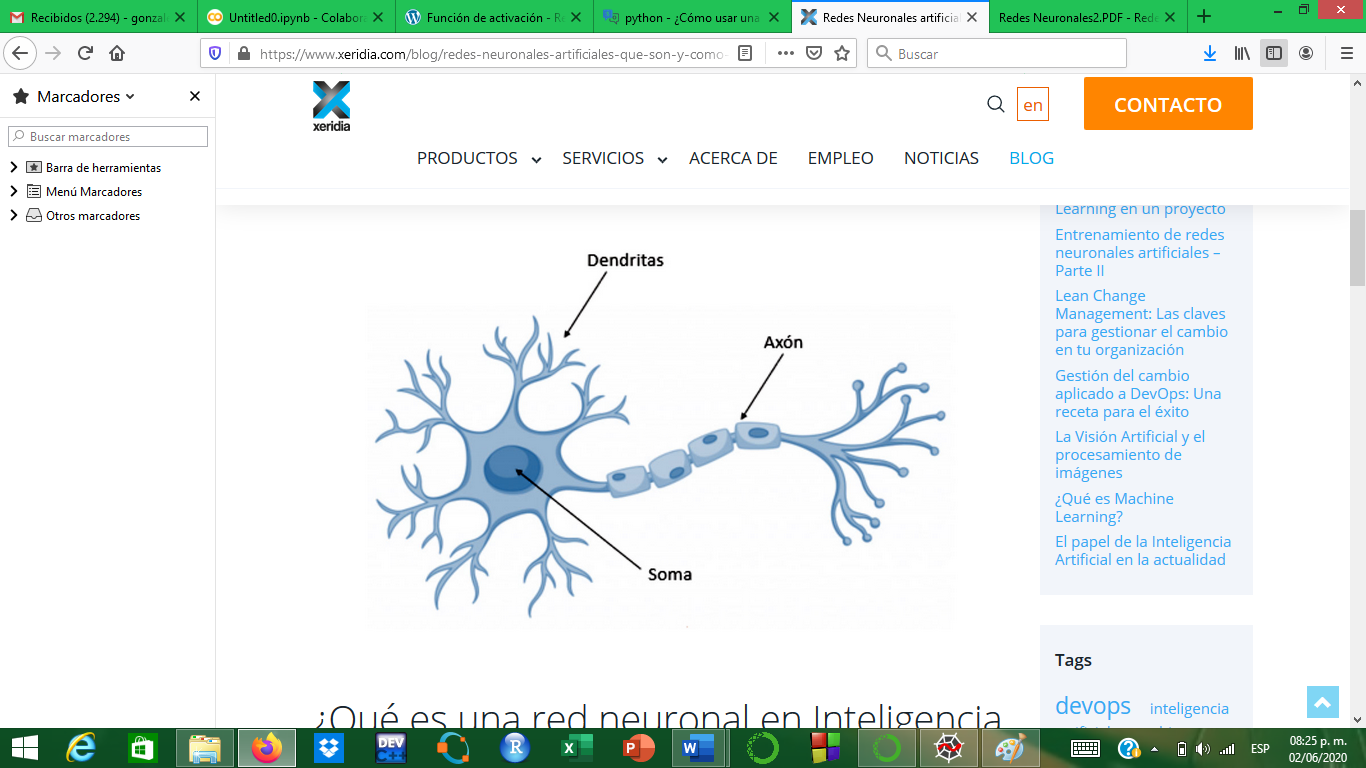
1936 - **Alan Turing**. Fue el primero en estudiar el cerebro como una forma de ver el mundo de la computación. Sin embargo, los primeros teóricos que concibieron los fundamentos de la computación neuronal fueron Warren McCulloch, un neurofisiólogo, y Walter Pitts, un matemático, quienes, en 1943, lanzaron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

1949 - **Donald Hebb**. Escribió un importante libro: La organización del comportamiento, en el que se establece una conexión entre psicología y fisiología. Fue el primero en explicar los procesos del aprendizaje (que es el elemento básico de la inteligencia humana) desde un punto de vista psicológico, desarrollando una regla de como el aprendizaje ocurría. Los trabajos de Hebb formaron las bases de la Teoría de las Redes Neuronales.

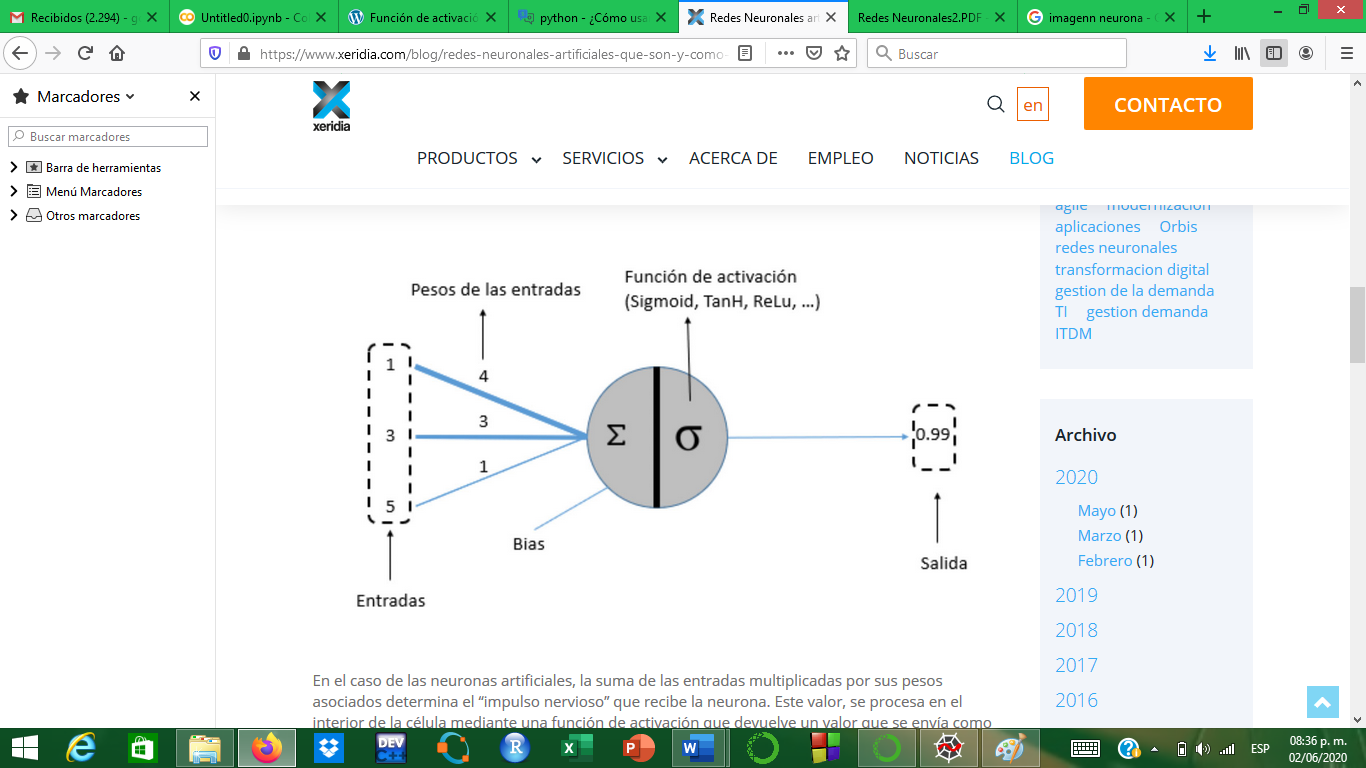
1956 - **Congreso de Dartmouth**. Este Congreso frecuentemente se menciona para indicar el nacimiento de la inteligencia artificial.

1957 - **Frank Rosenblatt**. Comenzó el desarrollo del Perceptrón. Esta es la red neuronal más antigua; utilizándose hoy en día para aplicación como reconocedor 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 anteriormente. Sin embargo, tenía una serie de limitaciones. Más tarde surgieron decenas de modelos y arquitecturas que resolverían ciertos problemas más especializados

**Marco Teórico:**



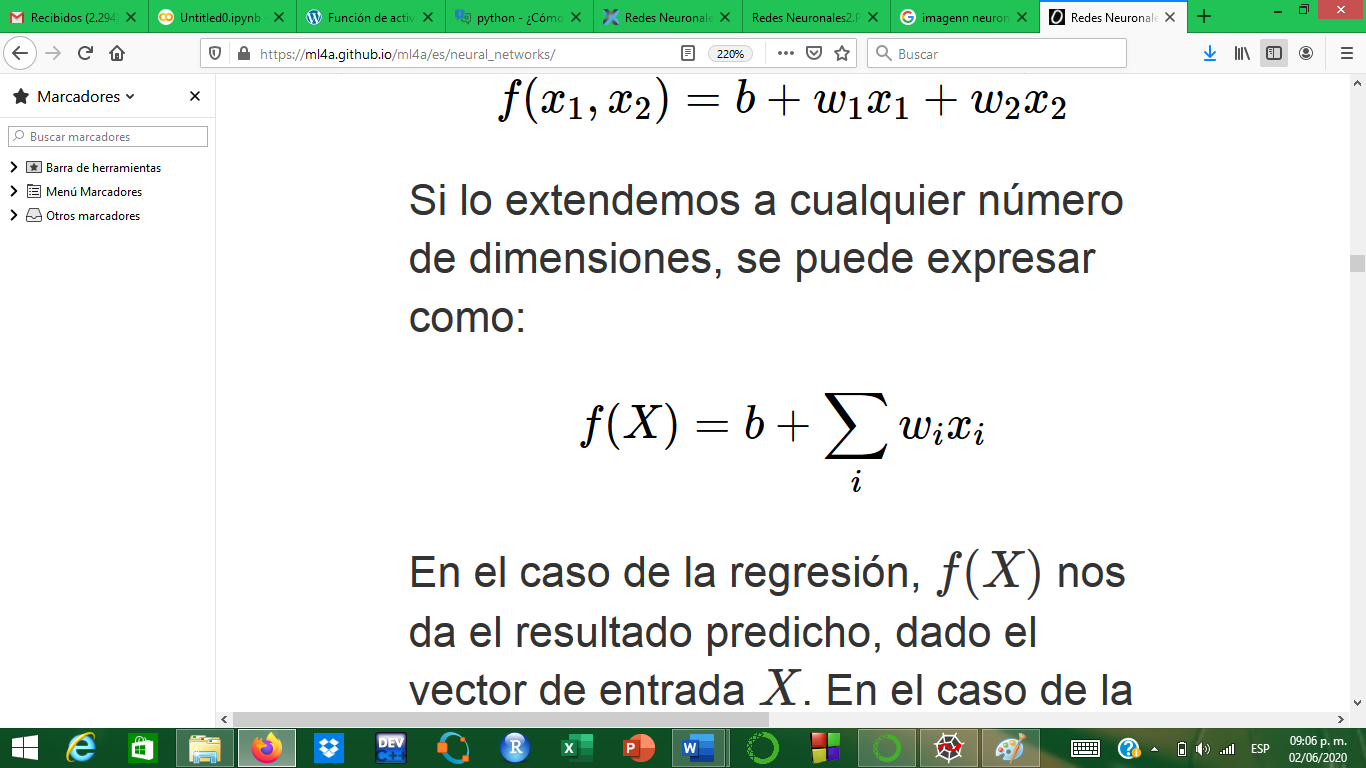
Imagen

Desde el nivel más básico de una neurona unitaria, biológicamente se sabe su función y como es el flujo de señales. En la imagen 1 se pueden observar las partes de una neurona biológica, en esta neurona el flujo de señales van desde las dendritas, pasan por el soma que procesa la señal, se dirige al axón y sale por alguna terminal del axón y esa señal se dirige hacia una nueva dendrita de otra neurona repitiendo el proceso. En una red neuronal Artificial no es muy diferente el proceso en especial una que está diseñada en una arquitectura de Perceptrón multicapa y Propagación hacia adelante.

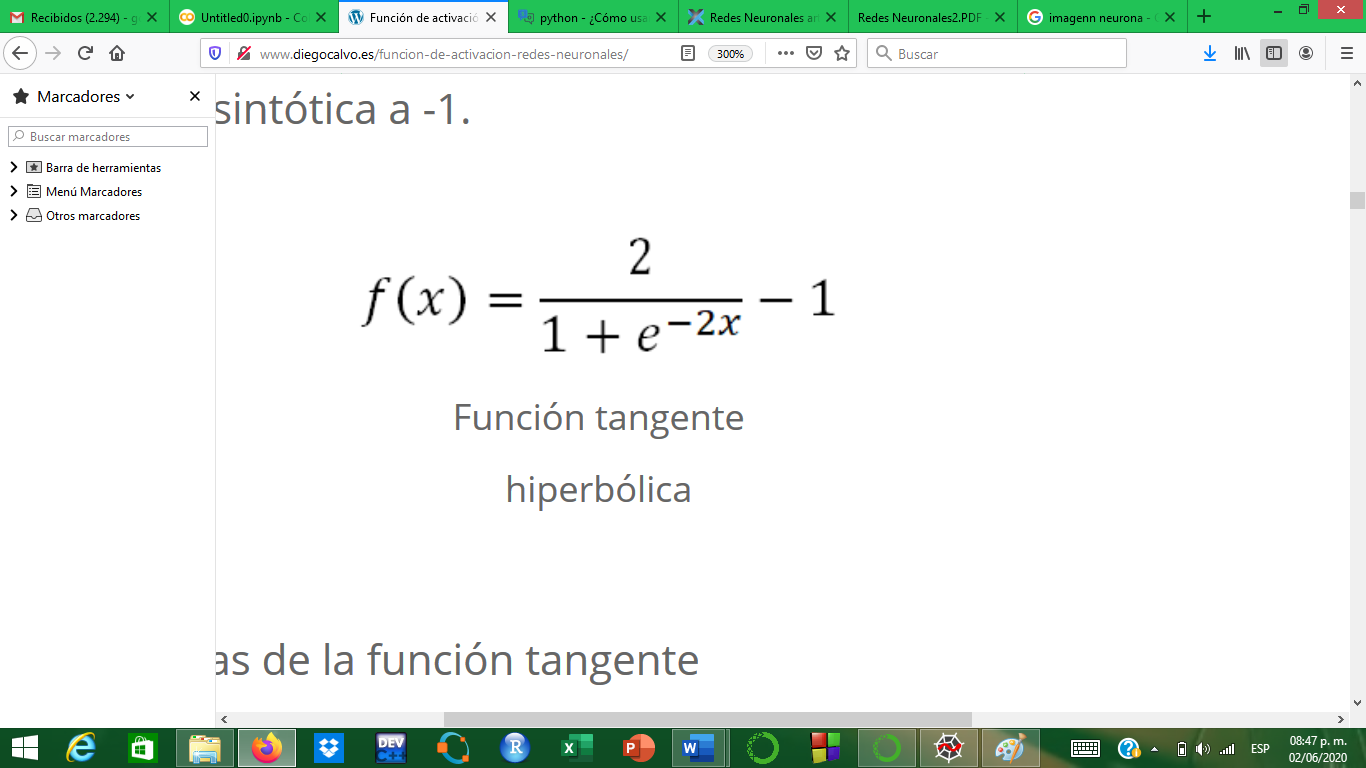
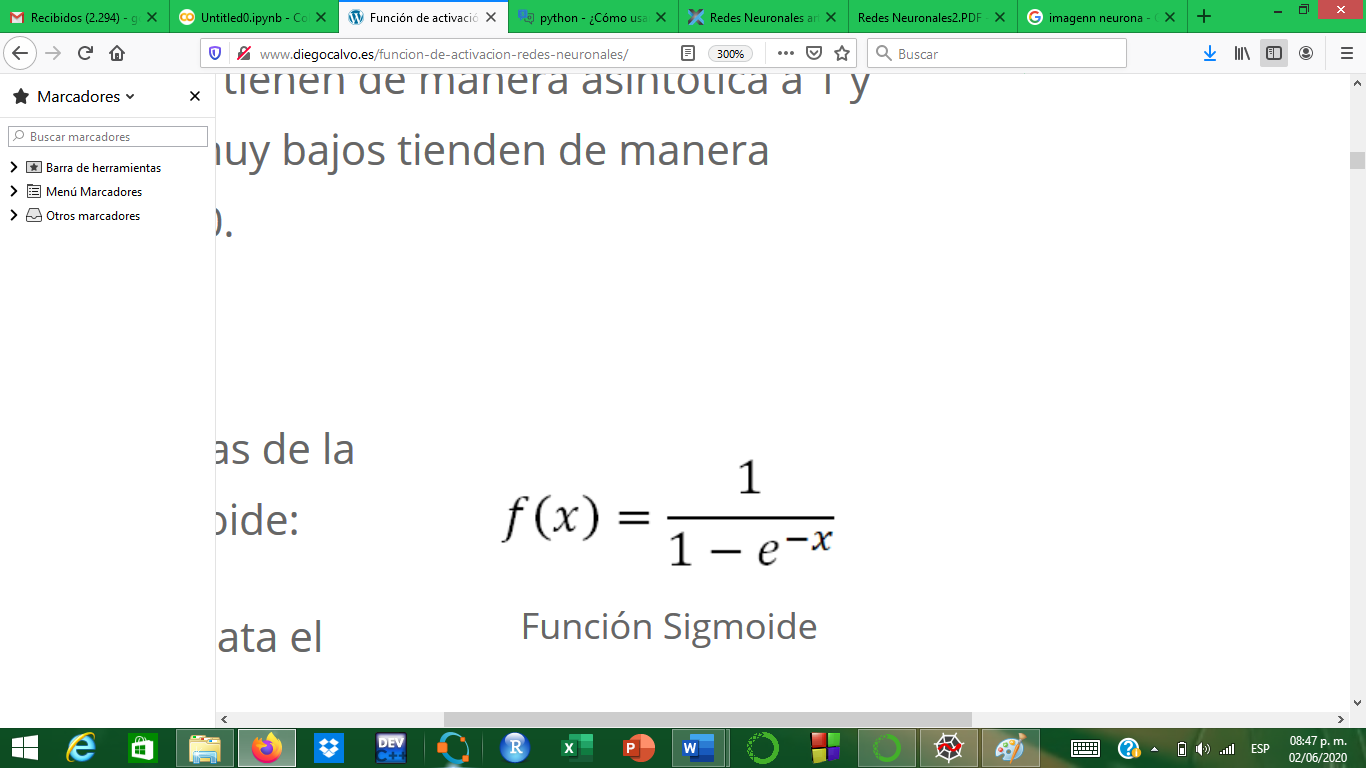
Imagen

En una ANN se tienen que definir valores o pesos de entrada, pesos en cada una de las capas y con una función de activación de van obteniendo valores y se va obteniendo un flujo de tal manera que cada neurona de cada capa y su valor asociado depende totalmente de los pesos de capas anteriores hasta dar un valor de salida que dependió de todas las capas anteriores como se muestra en la imagen 2.

A menudo al término b se le llama sesgo (en inglés bias), ya que controla qué tan predispuesta está la neurona a disparar un 1 o un 0 independiente de los pesos. Un sesgo alto hace que la neurona requiera una entrada más alta para generar una salida de 1. Un sesgo bajo lo hace más fácil.

Tanto en las redes neuronales artificiales como biológicas, una neurona no sólo transmite la entrada que recibe. Existe un paso adicional, una función de activación, que es análoga a la tasa de potencial de acción disparando en el cerebro. La función de activación utiliza la misma suma ponderada de la entrada anterior:

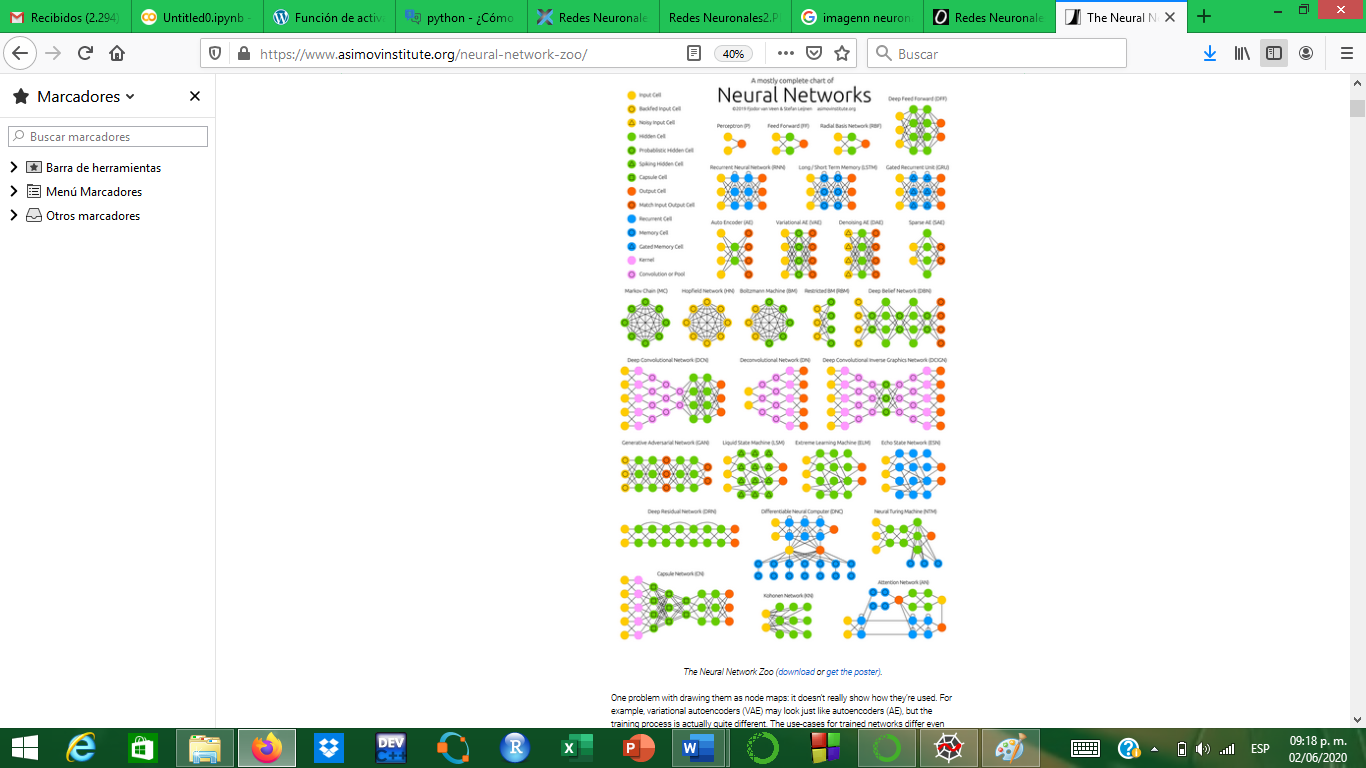
Hasta ahorita se ha mencionado lo que es una función de activación y realmente no es solamente una función, es una familia de funciones que su uso depende, de igual manera de su aplicación, están desde las funciones sigmoide, tangencial hiperbólica, escalonada, entre otras.



La función sigmoide transforma los valores introducidos a una escala (0,1), donde los valores altos tienen de manera asintótica a 1 y los valores muy bajos tienden de manera asintótica a 0.

La función tangente hiperbólica transforma los valores introducidos a una escala (-1,1), donde los valores altos tienen de manera asintótica a 1 y los valores muy bajos tienden de manera asintótica hacia -1.

Tipos de arquitecturas:

En la imagen 3 se presenta un poster de un panorama sumamente amplio de tipos de arquitecturas de ANN, y en la página web The Neural Net Zoo de The Asimov Institute se pueden observer con detalle y tarea que se les puede dar a cada ANN.

Imagen

Entrenamiento:

Para este proyecto, por cuestiones de tiempo, no se fijó un entrenamiento pero es importante saber lo que es.

Entrenar una red neuronal consiste en **ajustar cada uno de los pesos de las entradas de todas las neuronas** que forman parte de la red neuronal, para que las respuestas de la capa de salida se ajusten lo más posible a los datos que conocemos ya sea para un reconocimiento de patrones o para alguna tarea que se le asigne.

**Objetivo:**

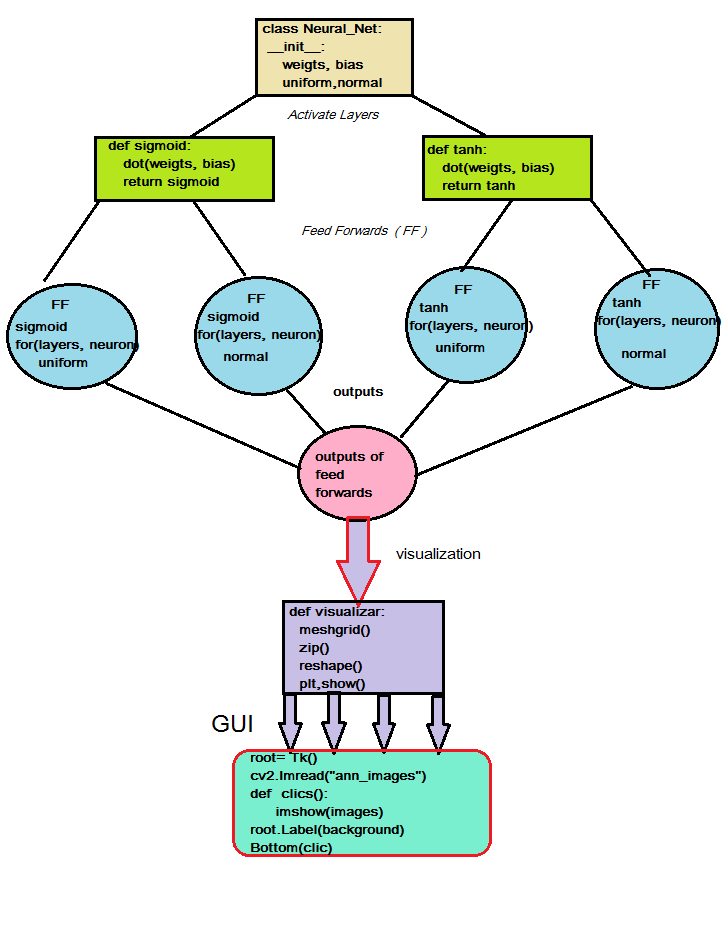
Mostrar y comparar gráficamente, con imágenes en una interfaz, cómo se ve el flujo de la comunicación de una ANN en una arquitectura de red neuronal por capas y de Propagación Hacia Adelante, con diferentes funciones de activación y con diferentes distribuciones en sus pesos.

**Justificación:**

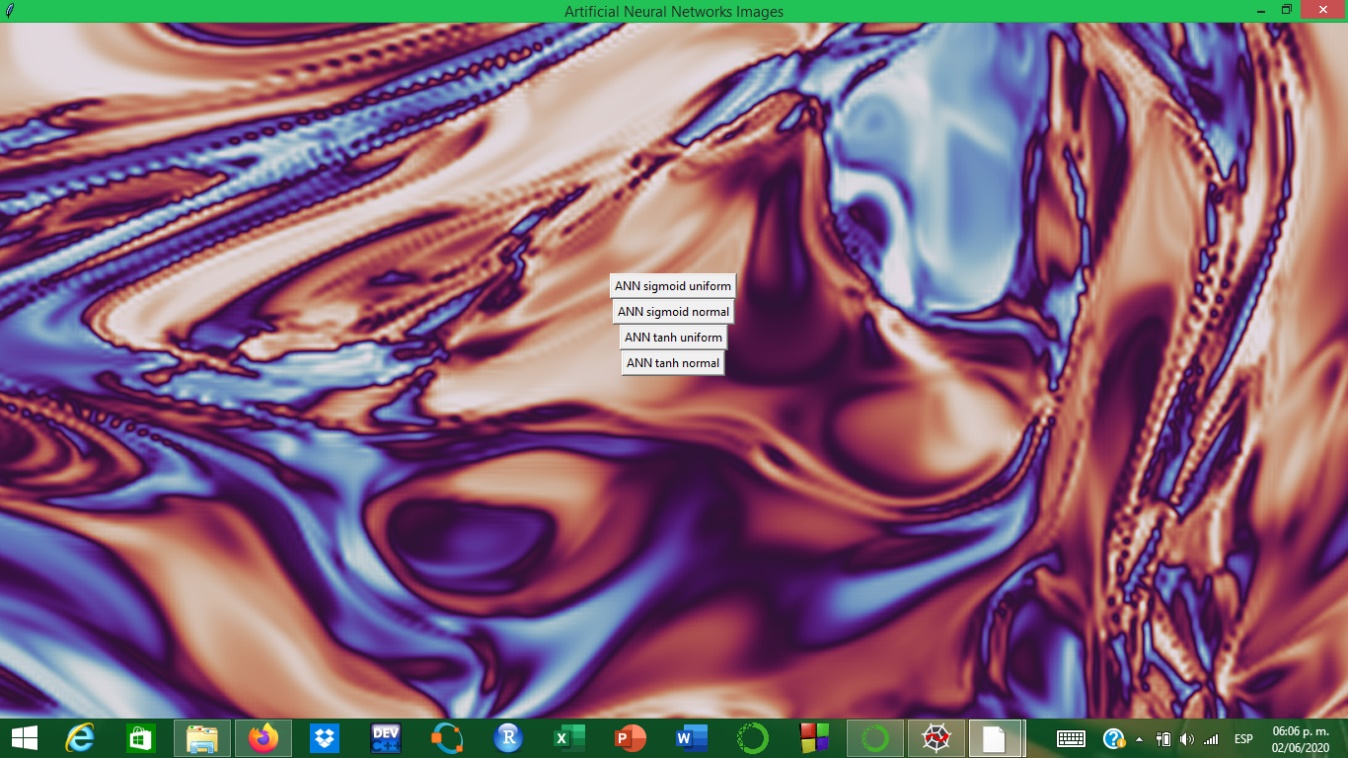
En ingeniería es muy importante obtener habilidades computacionales para poder desarrollar modelos interesantes que se han estado usando y funcionan de alguna manera. Simular una red neuronal no es una tarea fácil, pero tiene un amplio campo donde desarrollarse y dar una gran aportación desde aproximaciones probabilísticas y estadísticas hasta el reconocimiento de patrones.

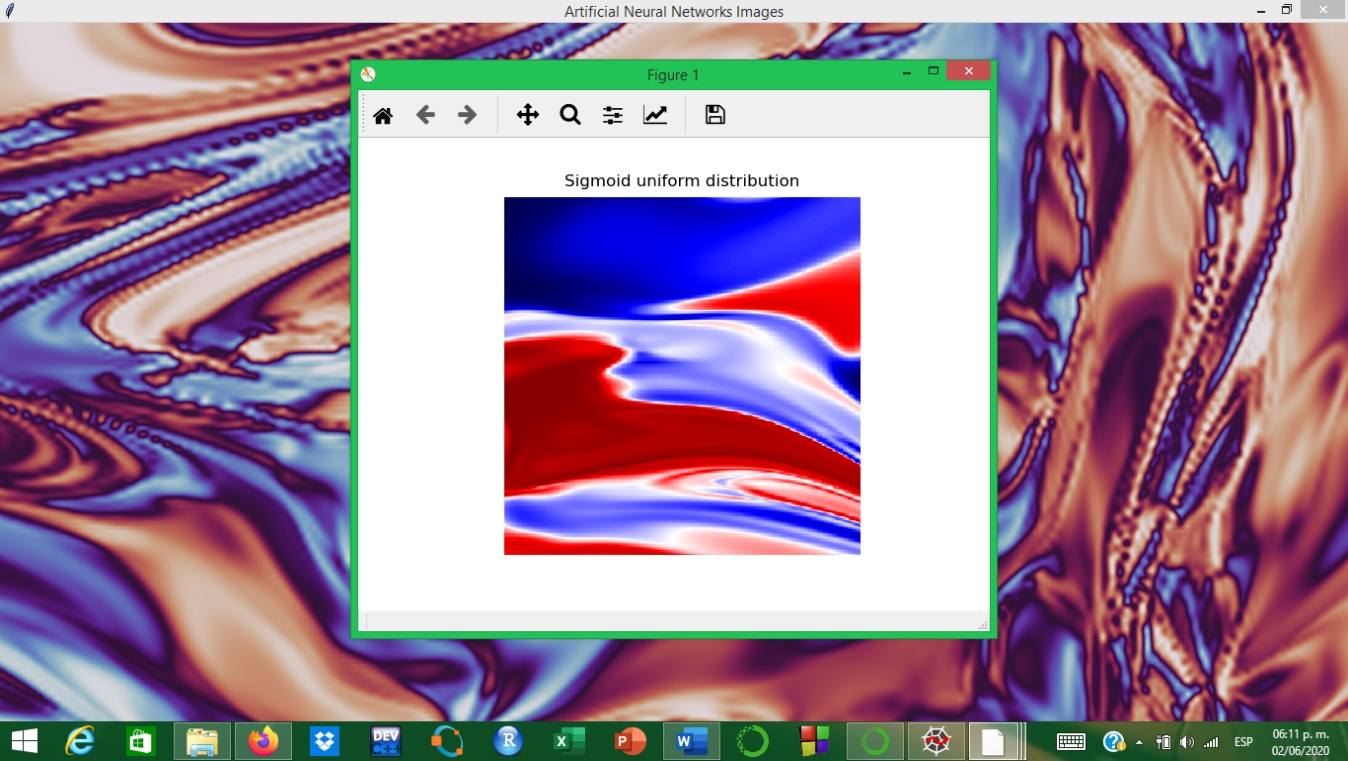
**Metodología:**

Se desarrolló un código en lenguaje Python, a través del ejecutador de aplicaciones Anaconda, donde se empleó Spyder como IDE. Se inicializaron vectores de pesos, y bias con distribuciones uniforme y normal, para después pasar esos pesos a una función que realiza los Feed Forwards o propagaciones hacia adelante que es la base de la arquitectura de propagación hacia adelante en donde se llamaban a diferentes funciones de activación, este proceso consistió en paras cada función de activación por cada neurona de cada capa con un ciclio for y retornando un solo valor para después pasarlo a una función de visualización que reiteraba el proceso creando un rejilla bidimensional y asignando el valor de un pixel coloreado por un colormap que está en la documentación de matplotlib.

Después de haber obtenido dichas imágenes, se desarrolló una interfaz gráfica (GUI) con botones para poder ver los resultados de explorar con dos funciones de activación y dos distribuciones diferentes, como los pesos y valores de entrada para cada nodo son completamente aleatorios, resultó que cada imagen de cada vez que se corriera el programa iba a ser diferente, pero según la función y distribución se podía predecir más o menos como iba a salir la imagen.

Resultados:

Se lograron obtener muchas imágenes explorando con diferentes colormaps y distribuciones en los pesos y bias pero en la interfaz gráfica solo se muestran 5 que es una de fondo que requirió mucho gasto computacional ya que es una red neuronal con 20 capas y 80 neuronas en cada capa, con una distribución normal y en una rejilla de 600x600 en decir 360 000 redes neuronales están detrás de la imagen de fondo y otras cuatro imágenes que salen al presionar botones con la descripción de la red neuronal seleccionada con su respectiva función y distribución.



Las siguientes imagenes muestran una red neuronal de 20 capas y 50 neuronas por capa, mediante un rejilla de 400 x 400

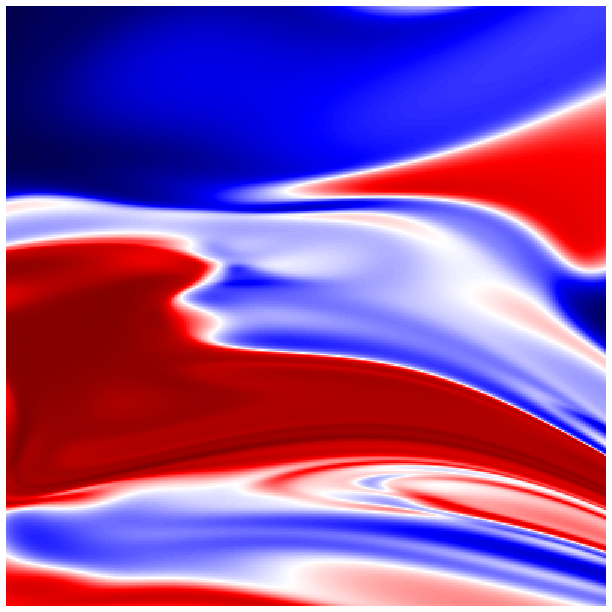
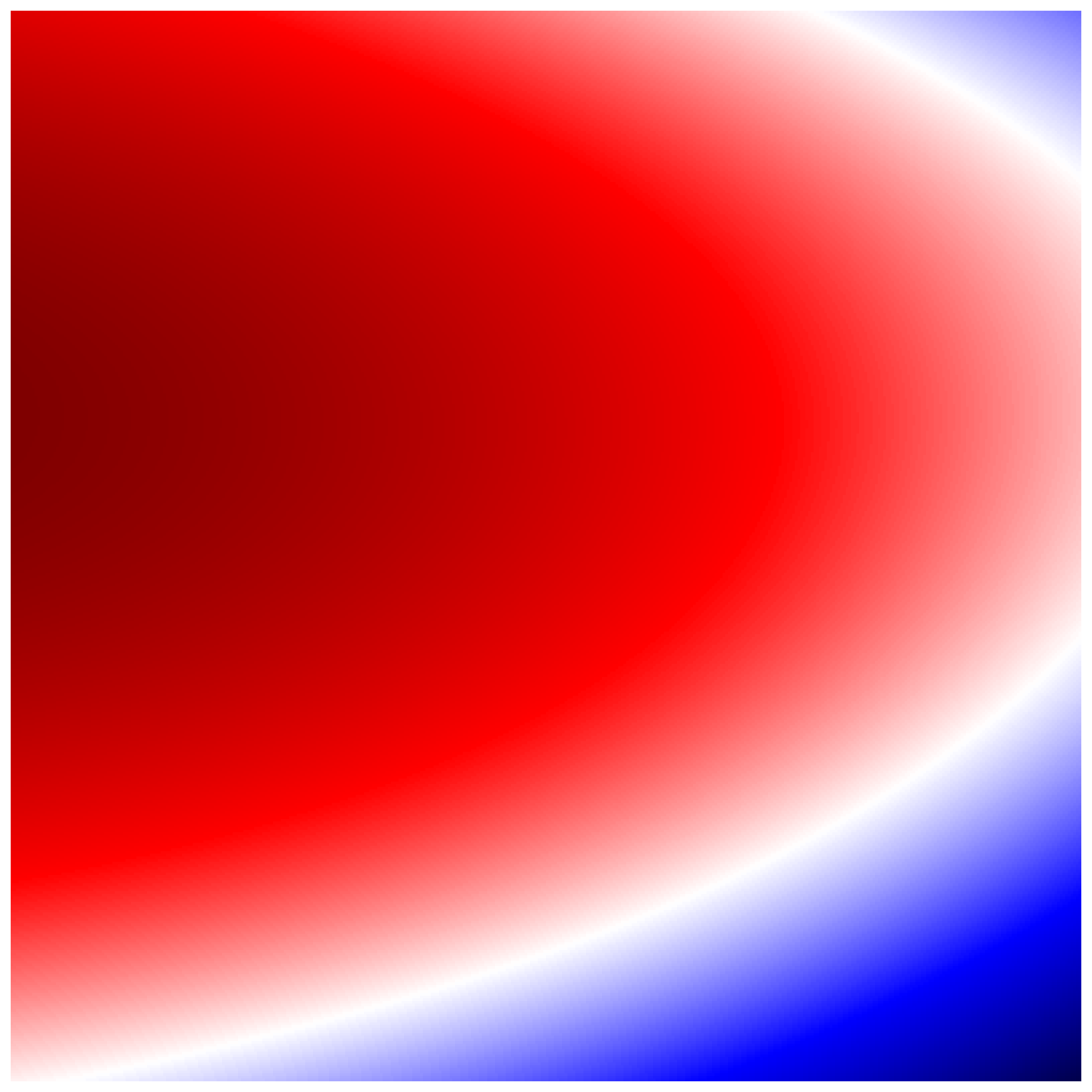
 La imagen 5 y 4 muestran como se ven las salidas de redes neuronales con un colormap seismic, ya que los valores más bajos se ven de color azul, los más cercanos a cero de color blanco y los más grandes de color rojo imagen 4 y 5 con función sigmoide y una distribución uniforme y normal respectivamente.

Imagen 5

Imagen 4

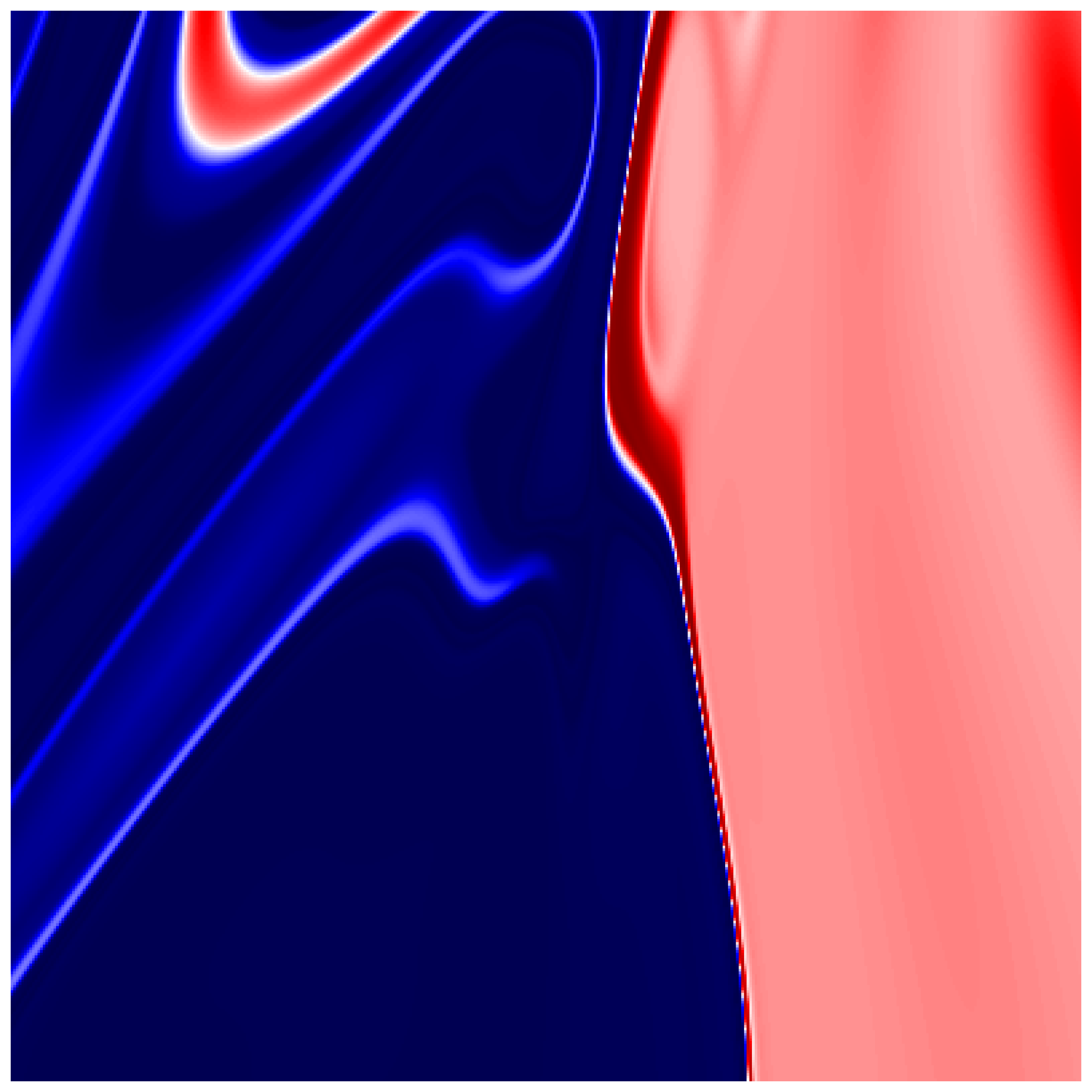
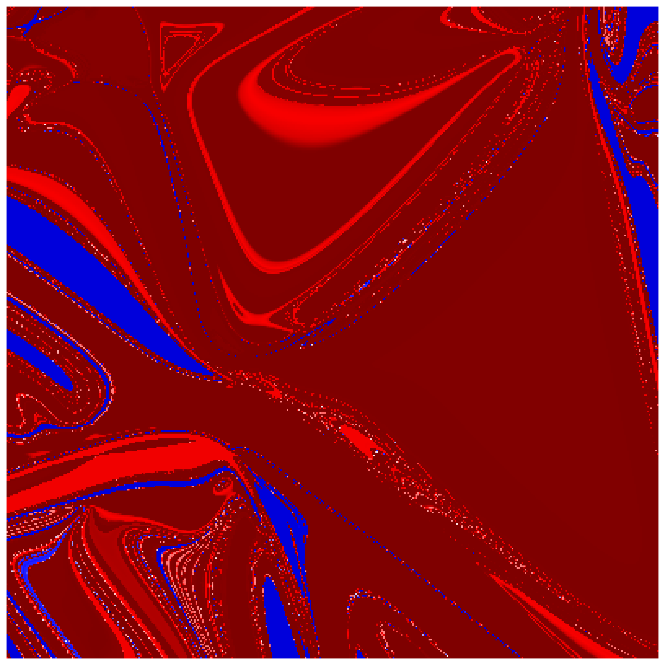


Imagen 6

Imagen 7

En las imágenes 6 y 7 es lo mismo respecto a colormap y distribuciones, uniforme y normal respectivamente, solo que esta vez con la función tangencial hiperbólca.

**Conclusiones:**

Las ANN tienen una gran escalabilidad y diversidad de aplicaciones que nos acercan cada vez más a automatizar procesos y actividades, un ejemplo muy grande son los autos que se manejan sin necesidad de ser tripulados, ya que estos van detectando los patrones de la carretera y entre más se manejen van a ir aprendiendo más de los caminos y reacciones a diversos cambios durante el manejo, también se podrían predecir eventos mediante estadística y un correcto entrenamiento de una red neuronal.

Por otro lado en nuestras imágenes vimos que usando la función sigmoide se obtienen flujos más diversos en especial con una distribución uniforme, mientras que con una función tangente hiperbólica se pueden obtener resultados más polarizados y va más orientado a decisiones binarias de si y no para sus aplicaciones.

Se obtuvieron imágenes muy artísticas y agradables a la vista.

**Agradecimientos:**

Un especial agradecimiento para Rodolfo Ferro del equipo de Future Lab quien me motivo a saber más sobre este tema y profundizar en ello y por su ayuda para guiarme prácticamente desde cero de este tema mediante sus cursos cuyo tema yo estaba nada familiarizado pero con gran curiosidad y al Dr. Luis Carlos ¨Padierna García por su apoyo en la documentación para profundizar y entender este tema que para mí, reitero, es complejo.

Bibliografía:

[1] Recuperado el 19 de mayo del 2020 de:

<https://towardsdatascience.com/introduction-to-artificial-neural-networks-ann-1aea15775ef9>

[2 ]recuperado el 19 de mayo del 2020 de: <https://www.google.com/url?sa=t&rct=j&q=&esrc=s&source=web&cd=&cad=rja&uact=8&ved=2ahUKEwj4-ay5lsjpAhUqIzQIHfqACAoQFjACegQIARAB&url=http%3A%2F%2Fconceptos.sociales.unam.mx%2Fconceptos_final%2F598trabajo.pdf&usg=AOvVaw2gbWEJ6WoQCxwd9mqwQe7m>

(tipos de redes multicapa) recuperado el 21 de mayo del 2020 de:

<https://www.asimovinstitute.org/neural-network-zoo/>

(perceptrón multicapa)recuperado el 21 de mayo del 2020 de:

<http://bibing.us.es/proyectos/abreproy/12166/fichero/Volumen+1+-+Memoria+descriptiva+del+proyecto%252F3+-+Perceptron+multicapa.pdf>

(funciones de activación) recuperado el 20 de mayo del 2020 de:

<https://medium.com/the-theory-of-everything/understanding-activation-functions-in-neural-networks-9491262884e0>