Artificial Neural Networks - diseño y variantes

Martín Gutiérrez

August 7, 2022

Introducción

Hemos ya hablado de los fundamentos de las ANN y de su forma de operar.

En este grupo de slides, se darán pautas básicas sobre el diseño de ANNs y se mostrarán distintos tipos de ANNs que están siendo investigadas y usadas hoy en día.

Sobre cómo diseñar una ANN...

Es preciso, evidentemente, saber qué es lo que se desea conseguir con la ANN. Se indicó previamente que la mayoría de las ANN siendo ocupadas actualmente son de tipo feed-forward. Sin embargo, existen otras clases de ANN, y dentro de las mismas feed-forward, subclasificaciones, que revisaremos más adelante en esta presentación.

Directrices base:

- Qué se quiere obtener de la ANN?
- Identificar cuál es el input y cómo representarlo. Asimismo, organizar un conjunto de entrenamiento (si es necesario).
- Determinar cuál es el output. Con ello se puede definir el formato y determinar la arquitectura a emplear.
- Seleccionar el conjunto de capas necesarias para resolver el problema y su tipo.

Qué se quiere obtener...

Recordemos que las ANN pueden cumplir una serie de distintas funciones. Funcionan como herramienta de aprendizaje supervisado así como de no-supervisado. Esto implica que pueden ser usadas para clasificar, predecir y además generar.

De ello, se desprende que es necesario emplear arquitecturas y tipos de ANN adecuadas para la finalidad buscada.

Y cuáles son esas arquitecturas?

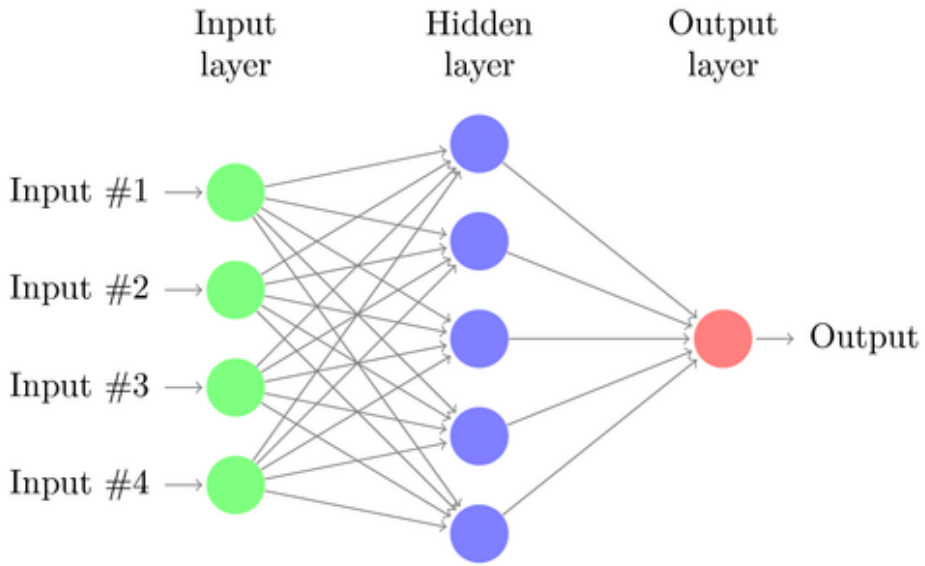
Existen muchas. En esta presentación abordaremos las más importantes y representativas:

- Multi-layer Perceptron (MLP)
- Convolutional Neural Networks (CNN)
- Recurrent Neural Networks (RNN)
- Generative Adversarial Networks (GAN)
- Self Organizing Maps (SOM)

... aunque la primera la vimos ya en la presentación anterior!!!

MLP (I)

MLP se consideran las ANN "de primera generación". Son ANNs compuestas de perceptrones. Típicamente usan entre una y dos capas escondidas (ver teorema de aproximación universal - Cybenko, 1989) más la capa de input y la capa de output.



MLP (II)

Este tipo de ANN hacen un mapping entre una entrada y una salida mediante una aproximación de funciones, como tal:

1. 0 hidden layers: Solamente se pueden representar funciones linealmente separables o decisiones
2. 1 hidden layer: Aproxima cualquier función que contiene un mapping continuo de un espacio finito a otro
3. 2 hidden layers: Representa una frontera de decisión arbitraria con precisión arbitraria usando funciones de activación racionales y aproxima cualquier mapping diferenciable con precisión arbitraria
4. Más de 2 hidden layers: Se agrega aprendizaje de representaciones complejas para capas de la red

Fuente:

`https://www.heatonresearch.com/2017/06/01/hidden-layers.html`

CNN (I)

Son el tipo más frecuente de redes que compone el Deep Learning. Se pueden definir como el tipo base de Deep Neural Networks (DNN). Las CNN hallan su origen en aplicaciones de visión por computador.

A diferencia de redes MLP, normalmente se componen de un conjunto heterogéneo de neuronas organizadas por capas que cumplen una función específica y bien definida. Así pues, en CNN es posible interpretar la funcionalidad de cada capa. De hecho, se trabaja a la inversa que en las MLP: se diseñan las redes de acuerdo a las funcionalidades necesarias e integrando las capas.

CNN (II)

La idea detrás de las CNN es que se procesan (literalmente) volúmenes de datos. El procesamiento se plasma en los pesos, al igual que en otras ANN. Dichos pesos representan importancia de ciertos objetos o aspectos de la entrada (generalmente imágenes). Asimismo, cada neurona responde únicamente a estímulos provenientes de una región limitada.

Relaciones espacio-temporales provenientes de los datos se capturan en los volúmenes de input. Estas relaciones se preservan de forma reducida por medio de la aplicación de distintas funciones (capas) de frecuente uso que se revisarán a continuación.

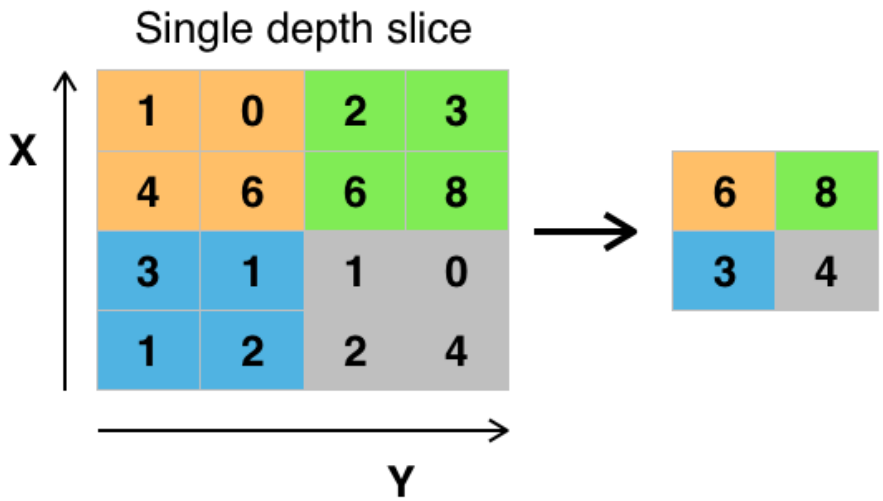
Convolution Layer

Este tipo de capas son las que principalmente extraen las características a partir de los datos que están procesando (un ejemplo pueden ser bordes, colores, orientaciones, etc.). Utilizan un Kernel/filtro. El Kernel trabaja como una ventana que se desliza sobre la data siendo procesada y aplica una multiplicación con la región observada.

Así, el resultado entrega una matriz que puede tener dimensiones menores, iguales o mayores. En los dos últimos casos, se habla del término de "Padding" (Ver Convolution.gif).

Pooling Layer (I)

Las capas de Pooling derechamente tienen por objetivo reducir la dimensionalidad de los datos. Trabajan también con un tamaño de ventana. Este elemento es clave en el ahorro del uso de recursos de procesamiento. Se puede asociar a un "resumen" de la región siendo procesada.



Pooling Layer (II)

Típicamente se usa a continuación de la Convolution Layer. Cumple básicamente dos funciones aquello desde el punto de vista operativo:

1. Extrae las características dominantes e invariantes
2. Elimina el ruido en los datos procesados

Los tipos de pooling más frecuentemente usados son:

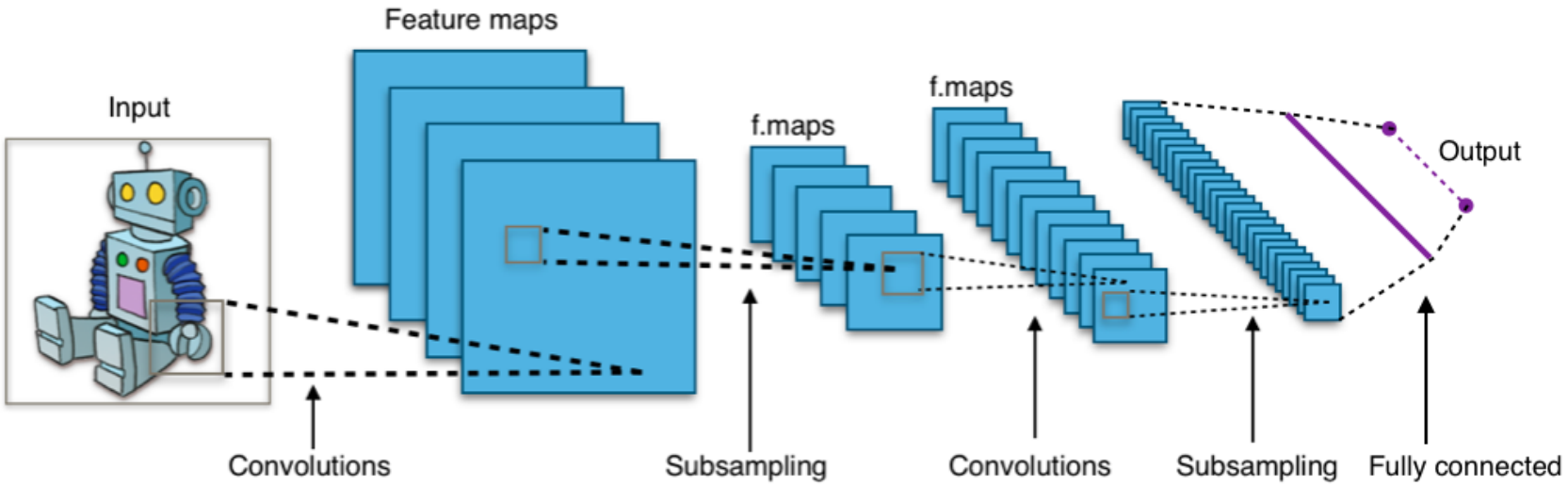
1. Max pooling
2. Average pooling

Fully Connected Layer (FCL)

Estas son conocidas: consisten en un MLP que clasifica a la data que sale de las capas anteriores. Dicha data se "aplana" y se procesa de igual forma que vimos en clases anteriores. Los cambios típicos que presentan estas MLP frente a las ya vistas son el reemplazo de las funciones de activación:

- ReLU: Función que se define como $ReLU(x) = max(0, x)$
- Softmax: Generalización de la función logística que se usa en capa de output

En resumen, una CNN se ve así



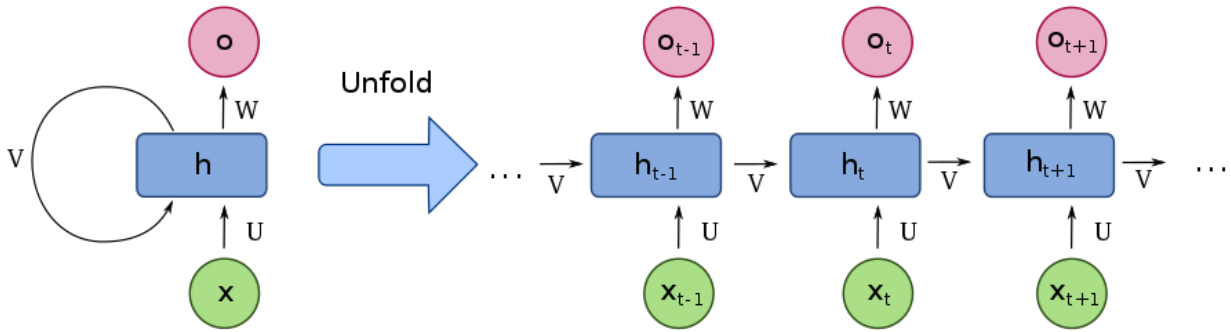
RNN (I)

Las RNN tienen el mismo formato general que las ANN vistas, sin embargo las diferencia la característica de recurrencia en su ejecución. Esto significa que el output de la ANN depende de las entradas, pero además una salida anterior (por medio de un ciclo).

Esta dinámica tiene dos efectos importantes:

1. La ANN se vuelve capaz de procesar secuencias
2. La ANN adquiere memoria

La operación de las RNN se esquematiza como varios estados de ellas en el tiempo (unfolding).



RNN (II)

Al tener las mismas características de base que las redes MLP, en realidad, los métodos de aprendizaje trabajan de forma similar. Esto significa que (una versión modificada de) backpropagation sigue siendo el algoritmo de aprendizaje empleado.

La adaptación de ese algoritmo se denomina Backpropagation Through Time (BPTT) y se hace en relación a cada corrida temporal que efectúa la RNN. El algoritmo trabaja igual, a excepción de que la propagación del error se hace siguiendo el desdoblado.

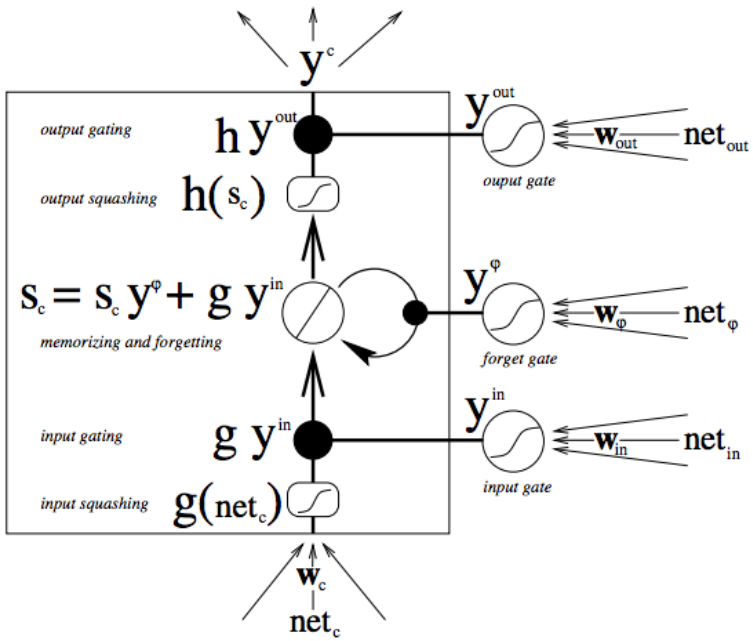
Problemas:

- Escala muy mal a secuencias largas
- Explosión y desaparición del gradiente

LSTM

Las unidades de tipo Long Short Term Memory (LSTM) buscan atacar los problemas mencionados previamente. Se refieren a unidades que tienen un mecanismo más sofisticado para el almacenamiento. Pueden retener secuencias más largas, pues mantienen un error más constante.

Las unidades de LSTM son más complejas. Un esquema se muestra a continuación:



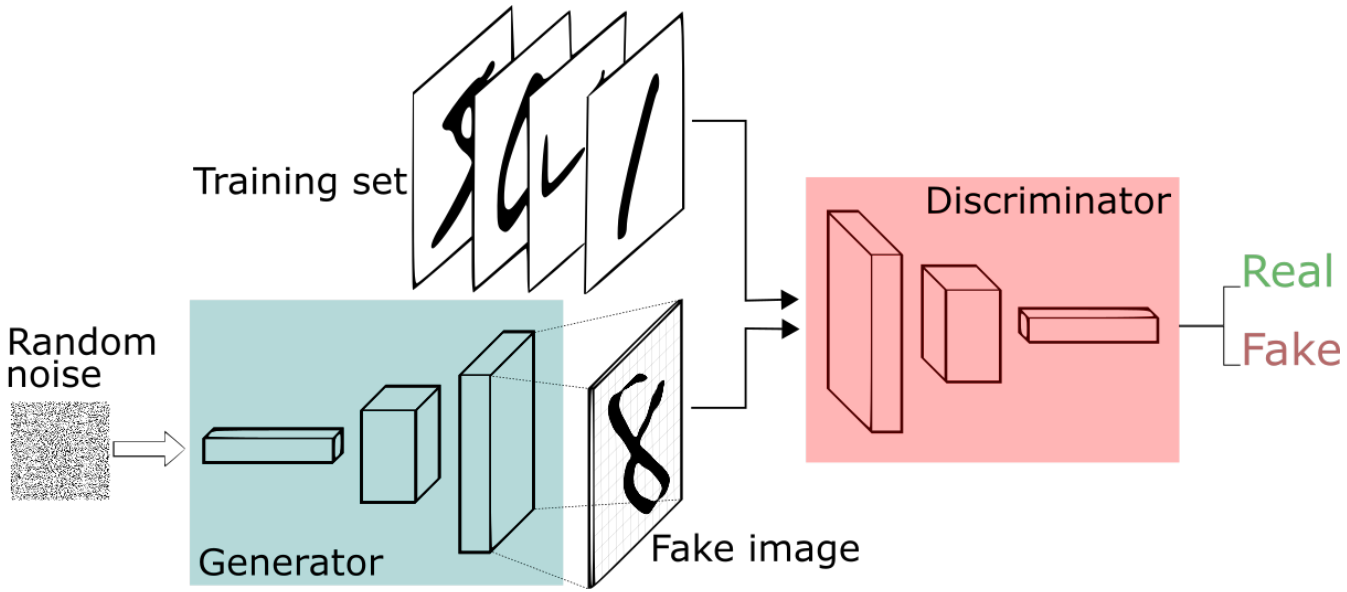
GAN (I)

Las Generative Adversarial Networks (GAN) se refiere a un modelo relativamente reciente (publicado en 2014) que involucra dos ANNs que se enfrentan y en que una de ellas (Generadora) intenta "engañar" a otra (Discriminadora). En la jerga de nuestro curso, una red Discriminadora busca clasificar los X hacia una etiqueta Y, y la red Generadora hace lo inverso, intenta, a partir de Y, reproducir valores de X.

Ambas ANN generalmente son CNN, y pasan por un proceso de entrenamiento. La Discriminadora se entrena como una CNN normal que clasifica los datos ingresados. Por otra parte, la Generadora se construye como una CNN invertida que produce datos que ingresan a ser procesados por la red Discriminadora. Esta dinámica se enmarca dentro de un loop de feedback entre ambas ANNs, lo que hace que se entrenen mutuamente.

GAN (II)

El esquema de organización general de GAN es como a continuación:



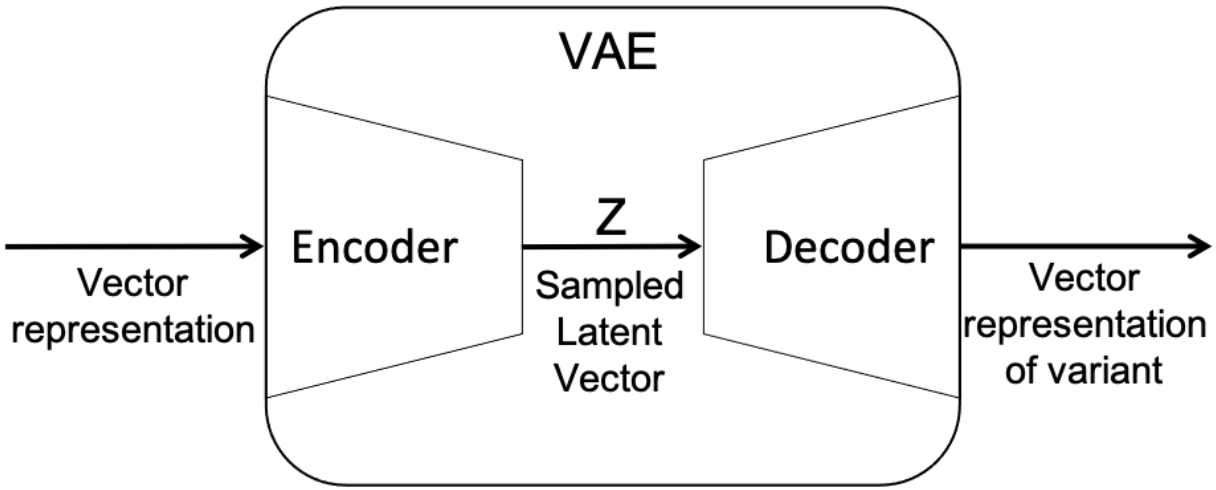
Problemas de las GAN:

- Tiempos de entrenamiento bastante largos
- Requiere un balance de "nivel" de las redes Discriminadora y Generadora

AE

Los Auto Encoder (AE) son otra clase de ANN de tipo generativa. Lo fundamental de este tipo de ANN es que se codifica una entrada, se obtiene una representación comprimida, y luego se decodifica, de modo de obtener una representación aumentada (en algún aspecto) del dato ingresado. Se asemeja a la GAN en el sentido que se generan datos nuevos, sin embargo emplea un elemento que trabaja sobre el mismo dato ingresado.

Un AE (específicamente un Variational Auto Encoder) se ve así:



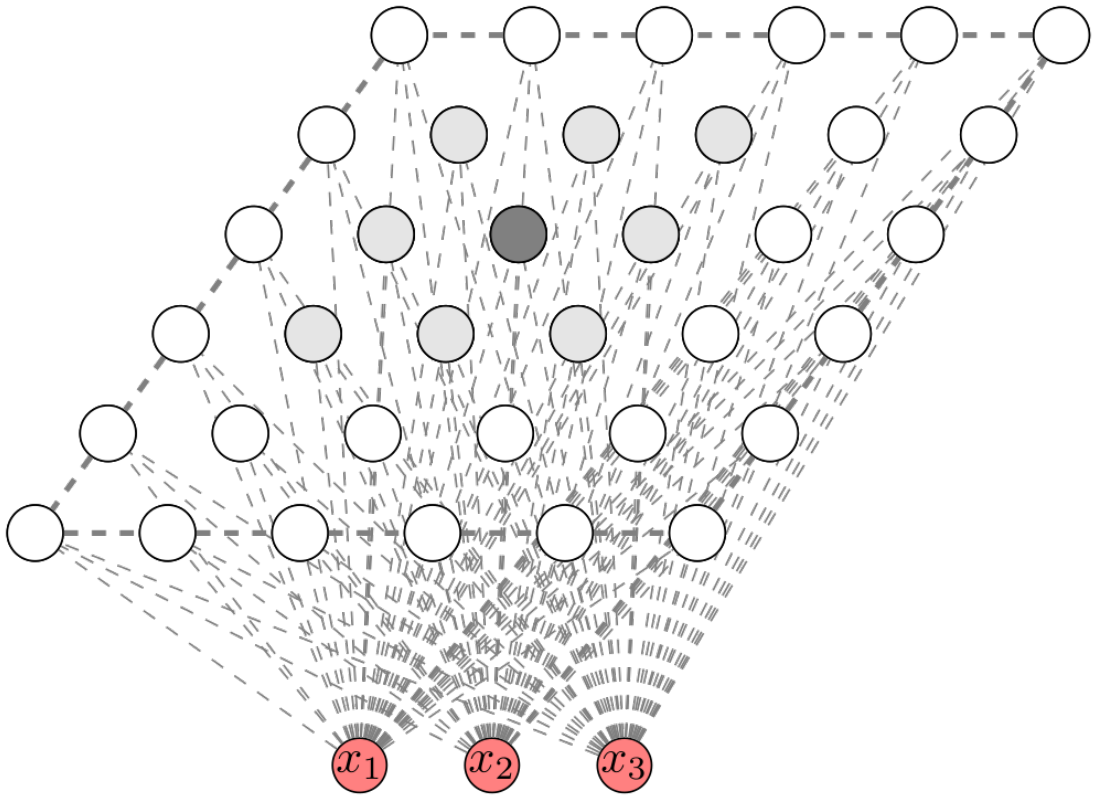
SOM (I)

Los Self-Organizing Maps (SOM) son un tipo de ANNs que se basan en la localidad de función del cerebro. Son redes que funcionan como herramientas de aprendizaje no-supervisado. El input es un vector de características, y la SOM opera en base a la comparación respecto a los pesos (aprendizaje competitivo).

Los pesos en este modelo dirigen la similitud hacia una neurona, por lo que se hace una lectura gráfica y espacial de la asignación de los vectores de entrada. Asimismo, el hecho de seleccionar un peso por su similitud afecta en alterar el peso para que se asemeje más al vector de entrada, pero también incide en los valores de los vecinos.

SOM (II)

Muchas veces, la representación de una SOM es a través de un mapa 2D de neuronas como se muestra a continuación:



Próxima clase

Ufff!!! Creo que esta ha sido la clase más larga que he dado en mucho tiempo!!!

Solamente nos queda la parte de algoritmos bio-inspirados :)