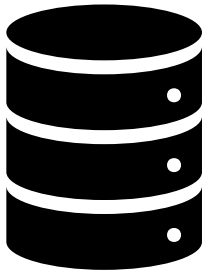
Introducción a las redes neuronales artificiales

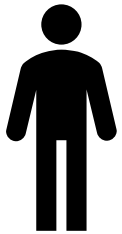
Analítica tradicional

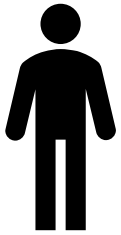
Datos	Transformaciones	Decisión	Acciones
	Personas calculan estadísticos, KPI gráficos con la data.	Personas toman las decisiones en base a los números, gráficos e indicadores observados.	Personas gatillan acciones a tomar en función de las decisiones tomadas.

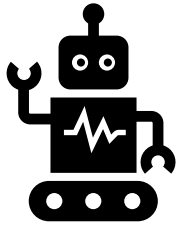
Ejemplo Las ventas en canales online han aumentado los últimos meses. En base a la información de ventas de las últimas semanas, se crearon paneles e indicadores, los cuales se usaron para el monitoreo de venta y la ejecución de campañas para el potenciamiento de la misma.

IA tradicional









Datos	Transformaciones	Decisión	Acciones
	Personas toman los datos, analizan eventos y procesos para…	…establecer reglas de acuerdo a la cual la máquina se guiará.	Con reglas y criterios de decisión bien definidos, la máquina realiza acciones.

Ejemplo Un sistema de verificación de respuesta automática, por ejemplo en caso que se valide la falta de stock de un producto le diga al usuario que no se encuentra disponible.

Analítica predictiva y Machine learning

Datos	Transformaciones	Decisión	Acciones
	Personas transforman la data de tal manera que el modelo funcione lo mejor posible.	La máquina, mediante un modelo toma las decisiones.	Personas o máquinas gatillan acciones a tomar en función del output del modelo.

Ejemplo Se crea un modelo de propensión a aumento de cupo. Se tomaron los datos, se crearon variables y un modelo retorna las probabilidades de que el cliente adquiera aumento de cupo. Utilizando esto se seleccionan algunos para campañas o para priorizar contacto.

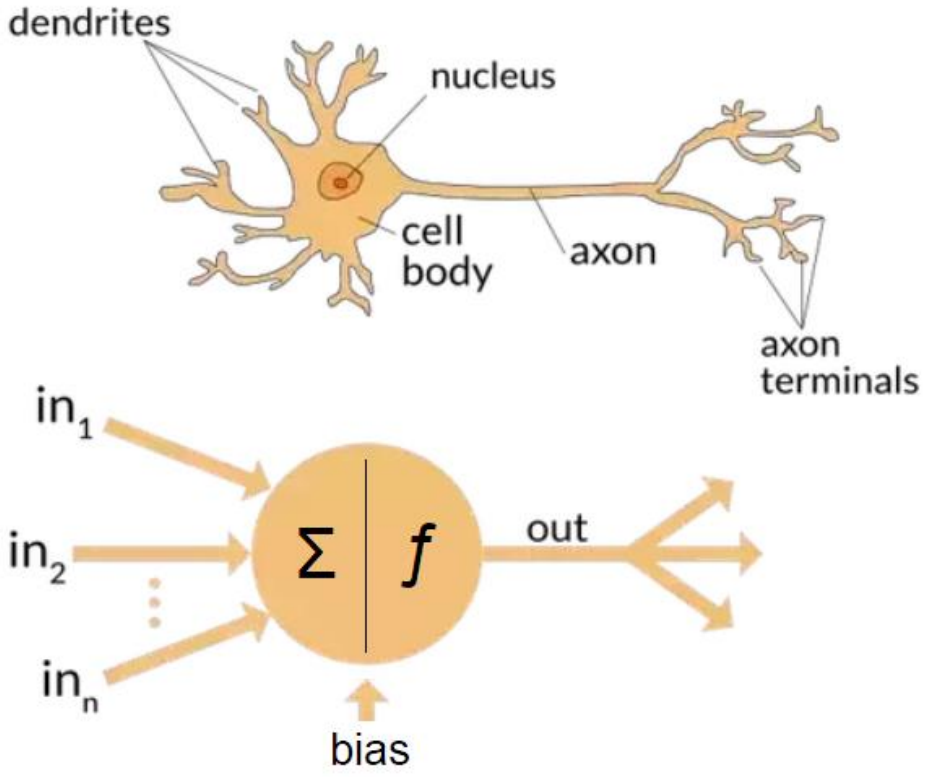
Deep learning

Datos	Transformaciones	Decisión	Acciones
	Mediante múltiples capas de redes, los datos se reprocesan obteniendo transformaciones implícitas.	La máquina, mediante una función de activación toma la decisión a la salida de la red	Con la red ajustada, se toma las decisiones de acuerdo al flujo de la red.

Ejemplo Un sistema que reconoce artefactos, recibe como input la imagen pura. La red mediante sus capas extrae las características de la imagen y luego la clasifica a un objeto. Finalmente la aplicación retorna la respuesta al usuario.

Redes Neuronales

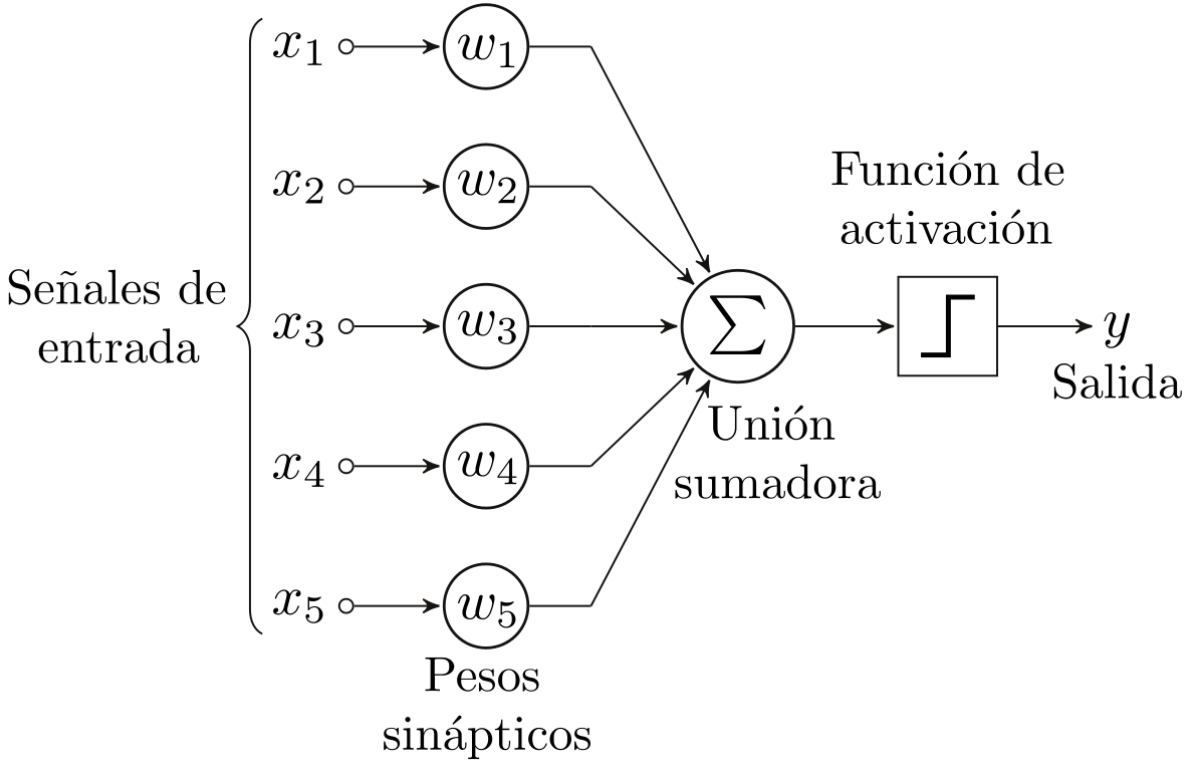
Las redes neuronales son una familia de algoritmos que basan su funcionamiento asemejando el funcionamiento de neuronas, es decir mediante el traspaso de señales de entrada a una salida que llega a otra neurona.



Al igual que la neurona es la unidad básica de una red neuronal biológica, las redes neuronales artificiales también tienen su unidad básica, esta es llamada **perceptrón**.

El perceptrón

Un perceptrón es una unidad que toma señales de entrada (datos), los optimiza y los pasa a una función de activación. La salida de un perceptrón es la predicción que buscamos. En el esquema, cada uno de los x serían nuestras columnas de datos y los w los betas.



En el esquema, la ecuación es

$$y = f(w_1x_1 + w_2x_2 + w_3x_3 + w_4x_4 + w_5x_5)$$

Donde f es una función de activación, quien es la encargada de pasar del espacio de los reales al conjunto de la respuesta. Es un análogo al la función de enlace

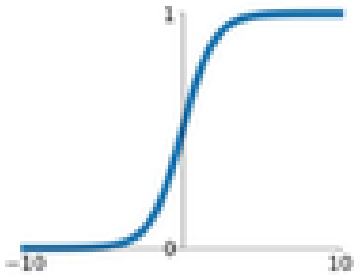
Ejemplo: perceptrón simple

El perceptrón

Activation Functions

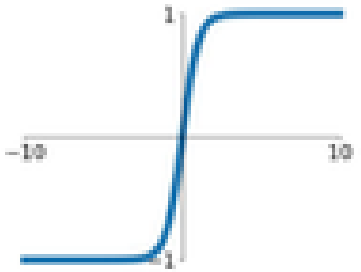
Sigmoid

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



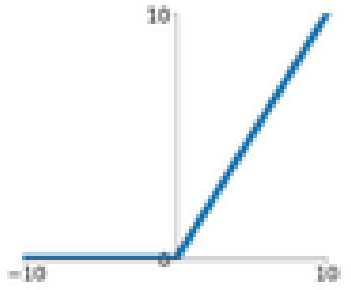
tanh

$$\tanh(x)$$



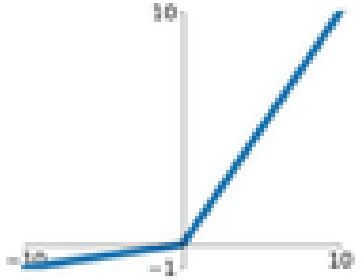
ReLU

$$\max(0, x)$$



Leaky ReLU

$$\max(0.1x, x)$$

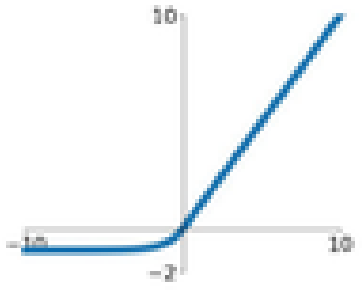


Maxout

$$\max(w_1^T x + b_1, w_2^T x + b_2)$$

ELU

$$\begin{cases} x & x \geq 0 \\ \alpha(e^x - 1) & x < 0 \end{cases}$$



El perceptrón

Intuitivamente la elección de los w, desde ahora en adelante **pesos**, consiste en hallar los valores que optimice una función de pérdida, es decir un indicador de que tan mal estamos prediciendo nuestra data.

Gradiente descendente

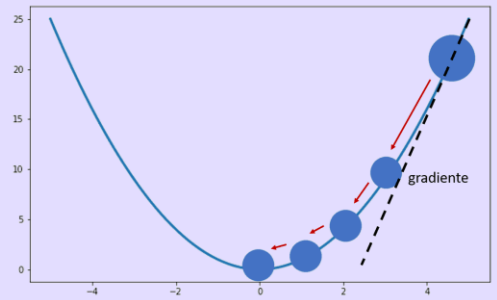
Método de optimización muy ligero que recorre en dirección del gradiente para llegar a un óptimo local

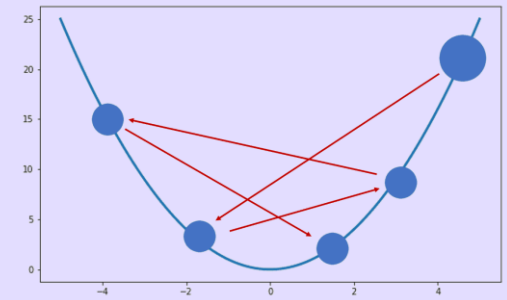
$$\boldsymbol{\theta}' = \boldsymbol{\theta} - \lambda \nabla_{\theta} L(\boldsymbol{y}; \boldsymbol{x}, \boldsymbol{\theta})$$

El parámetro λ se conoce como **tasa de aprendizaje.**
Este método es preferido pues es ligero, sin embargo frente a conjuntos de datos muy elevado aún es lento. Una opción es muestrear un *minibatch* de los datos y aplicar optimización con ese *minibatch* eso se conoce como **gradiente descendiente estocástico**.

Tasa de aprendizaje

La tasa de aprendizaje establece que tan rápido debemos avanzar en la dirección de mayor decrecimiento. La dificultad es que si se avanza de forma muy acelerada es más difícil converger al resultado. Como regla general se usa una tasa de aprendizaje pequeña, en caso que el tiempo de entrenamiento no sea problema.





Herramientas

En cuanto a frameworks para ajuste y diseño de redes, Python obtiene bastante ventaja con relación a otros lenguajes. Existe muchos, pero los tres principales son los siguientes.

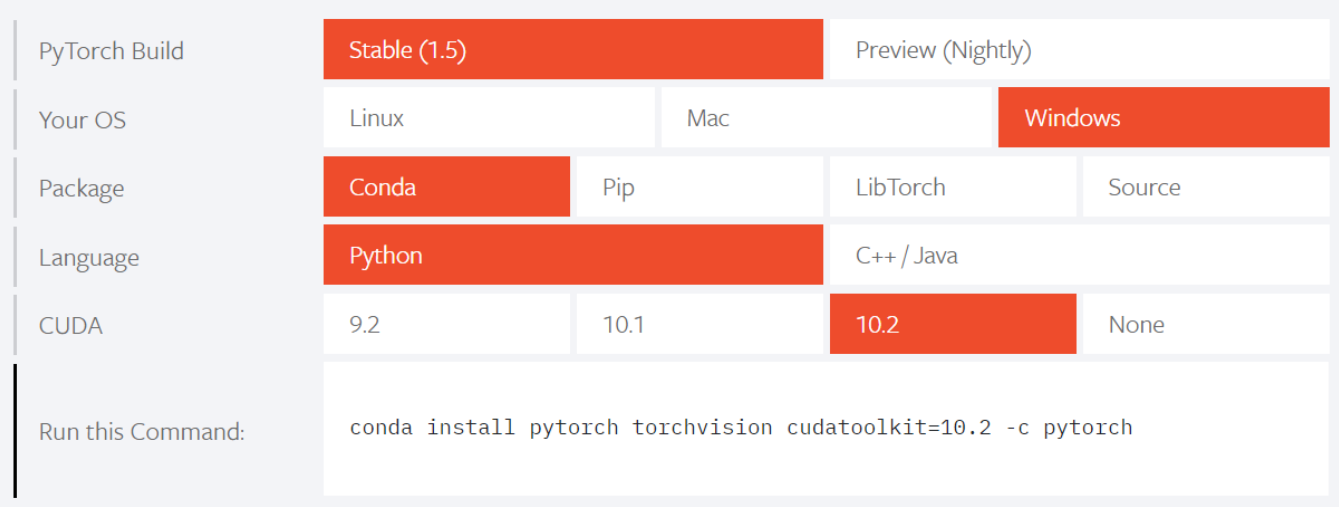
Keras es de uso más sencillo que los otros dos. Es una interface hacia tensorflow para ajustar redes de forma más sencilla

TensorFlow es un framework creado por Google. Destaca por su potencia, versatilidad y capacidad de poner en marcha modelos en producción

PyTorch, creado por Facebook permite el ajuste de forma más dinámica, sin ser muy invasivo con el lenguaje Python.

Pytorch

Desarrollado por Facebook, PyTorch es una interface de Python para Torch, un framework escrito en LUA. Como ventaja tiene que es el más flexible de los tres, pudiendo computar de forma dinámica, además es poco invasivo en el lenguaje Python.
Para instalarlo, basta ir a https://pytorch.org/, seleccionar su opción favorita e instalar.



En general, para instalarlo:

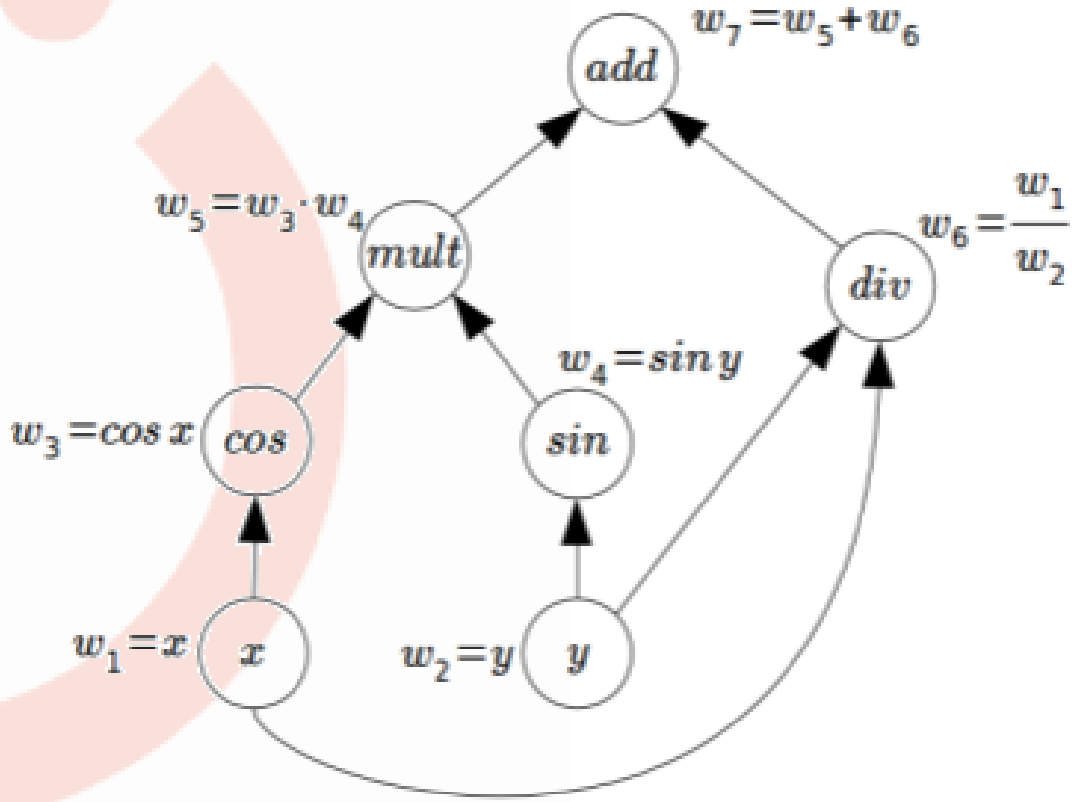
Pytorch Build indicar siempre el estable.
Your OS indicar su sistema operativo.
Package estamos utilizando Anaconda, por lo cual Conda es la selección. Si no usan Anaconda, usar Pip.
Language A menos que lo quieran usar en C++ o Java, indicar Python.
CUDA Indicar versión de CUDA, el cual es una interface para programar en GPU Nvidea. Si no tienen, no es obligatorio, Pytorch lo usa para ejecución en GPU.

Grafos computacionales

Un grafo computacional es un representación de una operación matemática, definiendo cada operador como los nodos del grafo y cada vértice indica a qué operador se le asigna el resultado.

Por ejemplo, el grafo de la derecha expresa la operación

$$Cos(x) \cdot Sin(y) + \frac{x}{y}$$



Notar que cada operación o función a aplicar es un nodo, incluyendo las variables (función identidad)

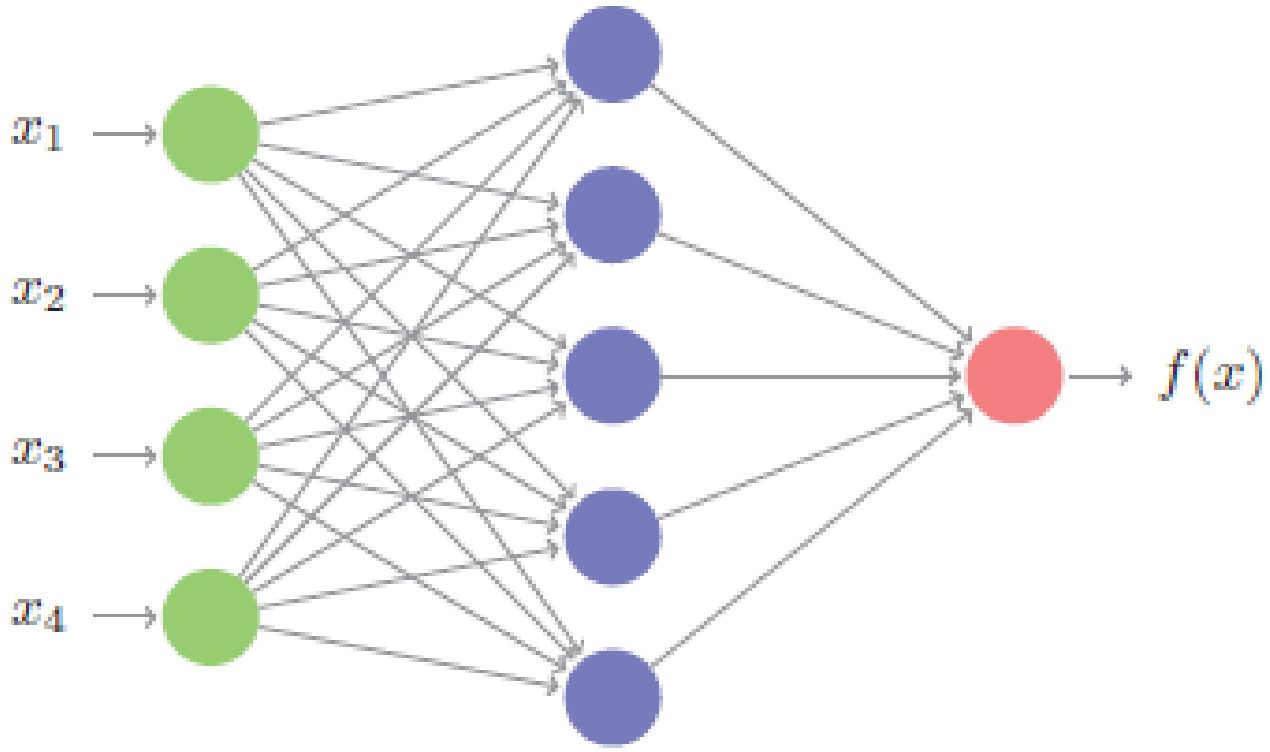
Glosario de términos

Antes de hablar de diseños de redes, es conveniente que estos conceptos estén claros.

Minibatch Subconjunto de los datos, es una muestra de ellos para acelerar las ejecuciones.
Tensor Arreglo multidimensional distribuido.
Epoch Iteración de ajuste de la red.
Grafo computacional es una operación especificada en términos de grafo
Pesos Ponderadores de cada variable que recibe la red.
Función de activación Función que permite pasar del conjunto obtenido a el conjunto de salida buscado.
Capa input Data de entrada, las variables puras.
Capa oculta múltiples capas en que se reprocesan las variables. No son observables
Capa output Salida, contiene la predicción
Célula En general un perceptrón pero puede tener modificaciones, célula es un término más general (una neurona es un tipo de célula)
Arquitectura de red se refiere al diseño de la red, cuántas capas, que tipo de capas, qué tipo de células, etc.
Dropout practica de eliminar neuronas aleatoriamente de las capas a fin de evitar sobreajustes.
Gradiente desvaneciente Problema que suele ocurrir al pasar por muchas capas, la función de activación puede provocar que la salida se estanque en un valor.

Feed forward net

Después del perceptrón este es el diseño de red más simple de todos, también llamado perceptrón multicapa consiste en aplicar múltiples capas de varios perceptrones cada una, conectando cada salida con otro perceptrón.

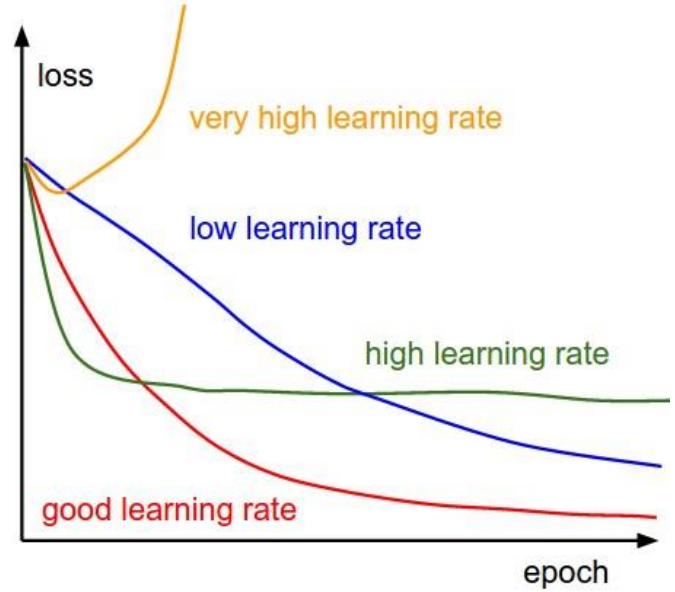


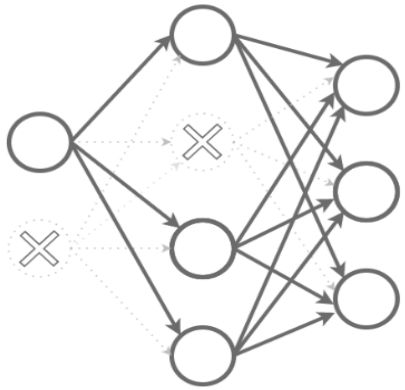
No existe regla definitiva para la configuración de red, sin embargo algunos consejos de ajuste.

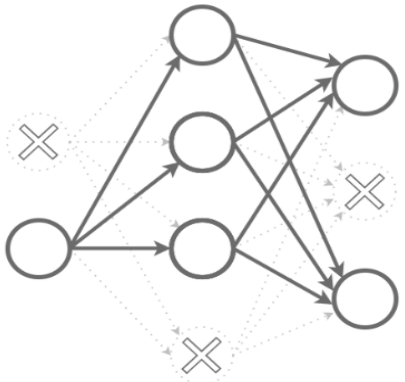
- No usar más de dos capas ocultas, en general una es suficiente, como máximo dos.
- Las capas ocultas debieran ir en reducción, es decir si son 600 variables la primera capa 300, la segunda 150, etc.
- En ocasiones cuando no son muchas columnas, ampliar la capa oculta permite encontrar nuevos patrones.

Configuración de la red

Al igual que con el perceptrón, una **tasa de aprendizaje** alta requerirá pocos epoch para llegar a la pérdida óptima (contra conjunto de validación) sin embargo puede evitar legar al óptimo. Se debiera comenzar con una pequeña y ver si la curva de decaimiento se estanca, y cuanto se puede subir sin ganar pérdida



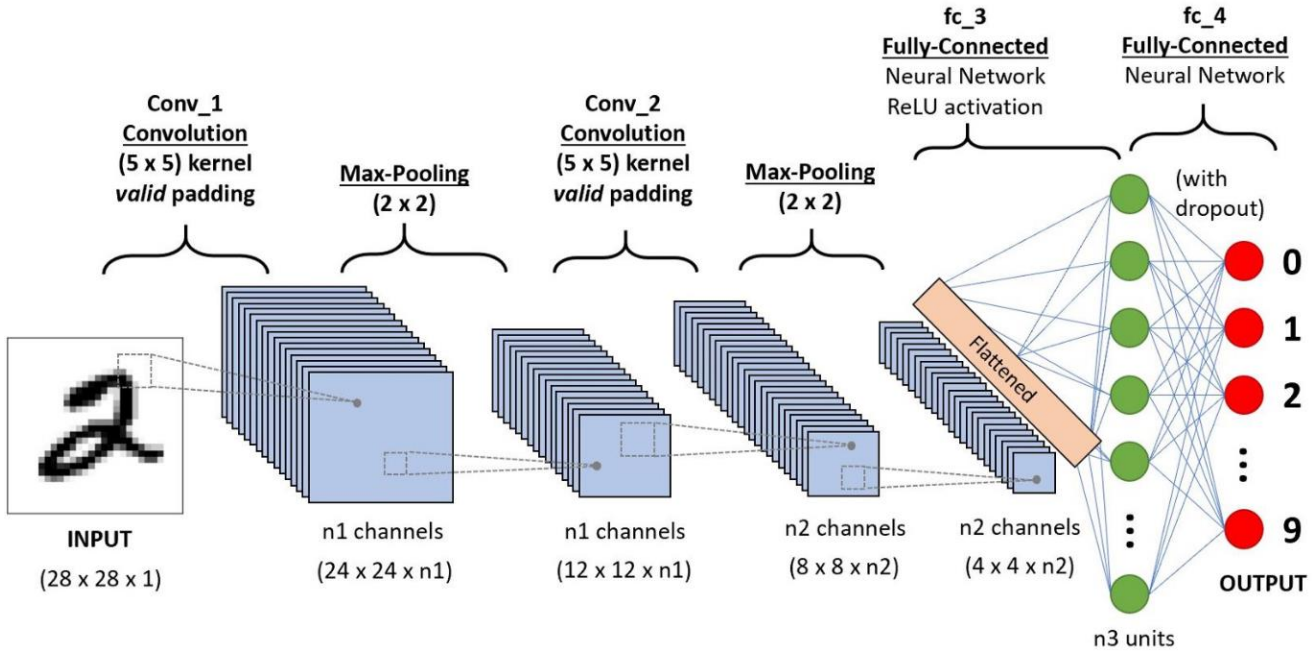




Dropout es una práctica que podría ser un análogo al podado de los árboles. Consiste en eliminar un porcentaje de neuronas de la capa para evitar aprendizajes de memoria, en general el porcentaje se va regulando en función del sobreajuste obtenido. Si la pérdida es extremadamente baja (llegando a cero) es indicio de un sobreajuste. Se sugiere eliminar la mitad de las neuronas y ahí ir ajustando en función del problema

Redes convolucionales

Las redes feed forward son las más habituales pues son bastante generales, sin embargo para superarlas se debe ocupar la herramienta adecuada para el problema adecuado. Las redes convolucionales (LeCun, 1989) son una familia de redes **especializadas en datos de grilla**, ejemplos de este tipo de datos son en contexto como reconocimiento de imágenes o minería de texto. El principio es extraer previamente características de la imagen en capas de convolución, para luego pasarlas a una red feed forward.



A partir de una imagen (matriz) se toman ventanas de esta matriz y se aplican funciones de convolución, las cuales extraen características en esa ventana, pueden ser varias. Posteriormente estas se agrupan en la operación pooling para obtener nuevas imágenes.

Configuración de la red

Sea **x(a)** un fenómeno observado en función de un parámetro **a** en un momento **t,** sea además **w(a)** una función de pesos, se define una **convolución** como

$$(\boldsymbol{x} * \boldsymbol{w})(\boldsymbol{t}) = \int \boldsymbol{x}(\boldsymbol{a}) \cdot \boldsymbol{w}(\boldsymbol{t} - \boldsymbol{a}) \boldsymbol{da}$$

En términos sencillos es sumar datos transformados dentro de una ventana

En nuestro caso, si tenemos una imagen e I(m,n) es un pixel en la posición m, n, y K(m,n) una función que llamaremos **kernel,** una convolución sería como

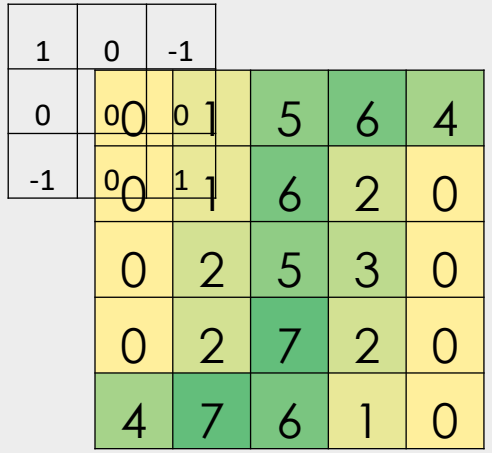
$$\boldsymbol{S}(\boldsymbol{i}, \boldsymbol{j}) = \sum_{\boldsymbol{m}=\boldsymbol{m_0}}^{\boldsymbol{m_0}+\boldsymbol{v}} \sum_{\boldsymbol{n}=\boldsymbol{n_0}}^{\boldsymbol{n_0}+\boldsymbol{v}} \boldsymbol{I}(\boldsymbol{i} - \boldsymbol{m}, \boldsymbol{j} - \boldsymbol{n}) \cdot \boldsymbol{K}(\boldsymbol{m}, \boldsymbol{n})$$

Ejemplo supongamos los siguientes datos

0	1	5	6	4
0	1	6	2	0
0	2	5	3	0
0	2	7	2	0
4	7	6	1	0

Al aplicar este kernel, conocido Edge kernel Nos queda la imagen que muestra con más detalles los bordes

1	0	-1
0	0	0
-1	0	1



1	6	-1	-6	-2
1	0	-4	-4	3
1	1	-1	-1	0
5	-3	-7	-1	2
-2	-7	0	7	2

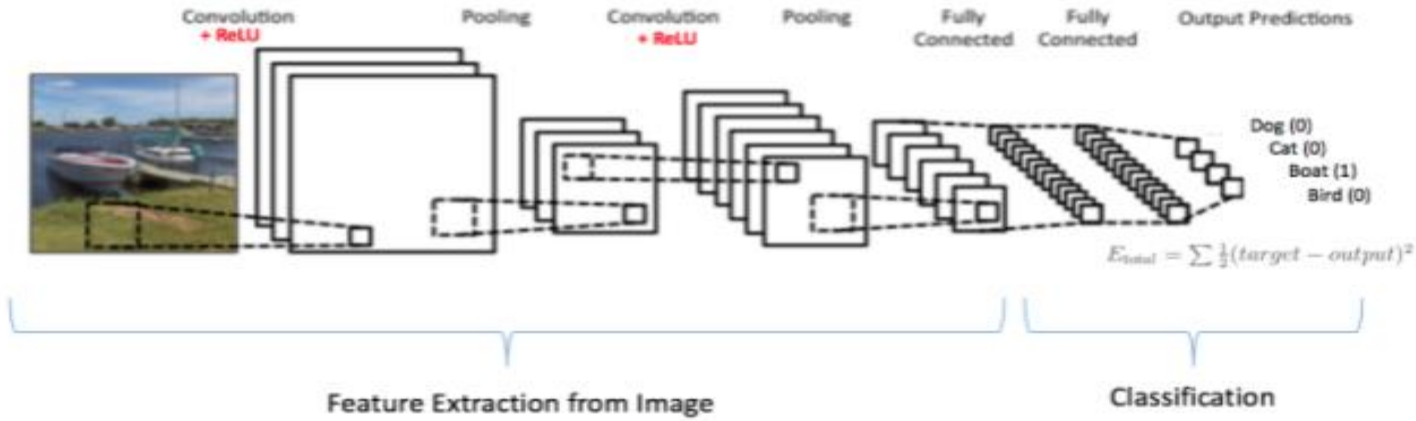
Configuración de la red

La operación conocida como **pooling**, consiste en ir tomando ventanas y reducir el tamaño de la imagen de acuerdo a un estadístico de resumen. Elecciones habituales para pooling son el promedio o el máximo. En este ejemplo si hacemos un max pooling de 2x2, queda

1	6	-1	-6
1	0	-4	-4
1	1	-1	-1
5	-3	-7	-1

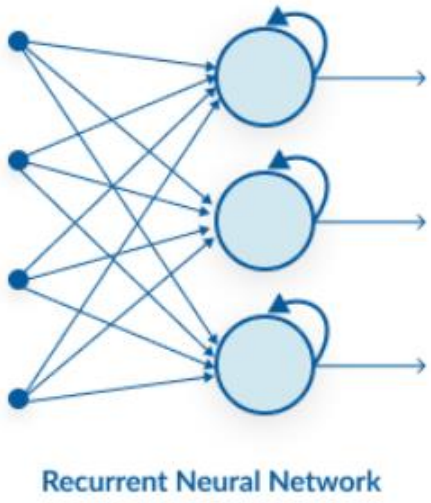
6	-1
5	-1

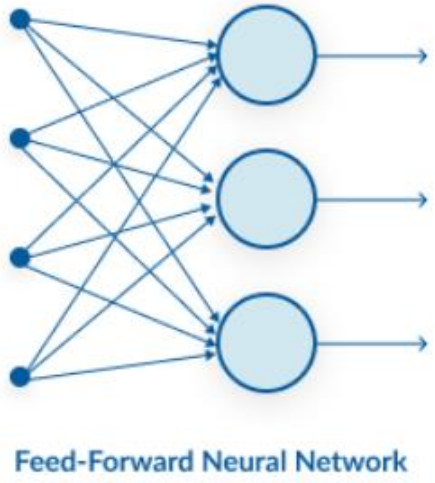
El último paso de una red convolucional, es el **flattening**, el cual es aplanar la imagen (tal como lo hicimos en el ejemplo del perceptrón), cada pixel entonces de las imágenes resultantes será una columna las cuales se le entregan a una red feed forward.

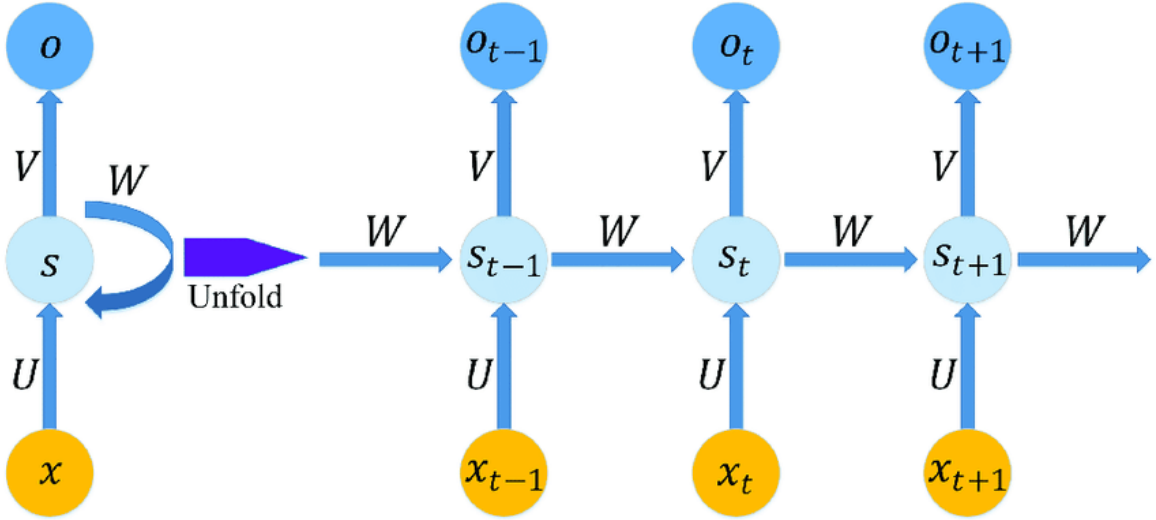


Otras redes: Redes recurrentes

Las redes recurrentes (Rumelhart et al. 1986) son otra arquitectura de redes en las cuales cada neurona retorna sobre si mismo además de pasar a la siguiente capa. Para cada neurona, estas se puedes "desdoblar" obteniendo un esquema como el de la imagen de la derecha. Por su naturaleza suelen se adecuadas para problemas de data indexada en el tiempo, pues estamos prediciendo outputs correspondiente a cada paso. Los contextos de uso son en series temporales, análisis de voz y minería de texto.
En general para problemas de poca volumetría (menos de 2 Gb de datos), no se aconseja su uso para pronósticos, pues suelen dar resultados más deficientes que métodos tradicionales (ARIMA, ETS, Holt-Winters, etc) y en caso de ser similares el entrenamiento es más lento.

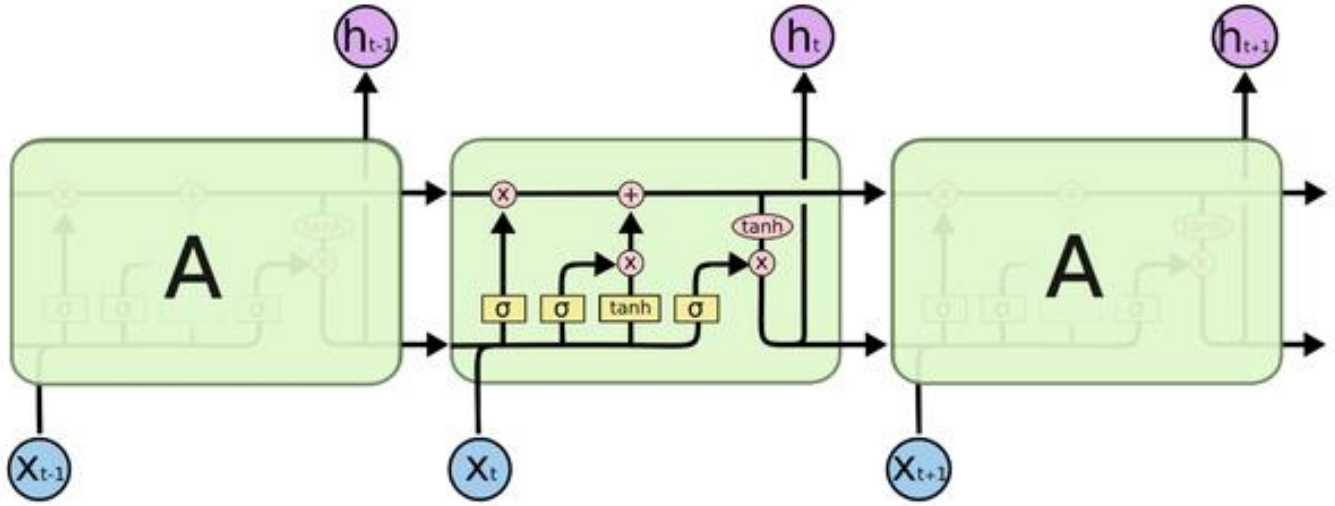






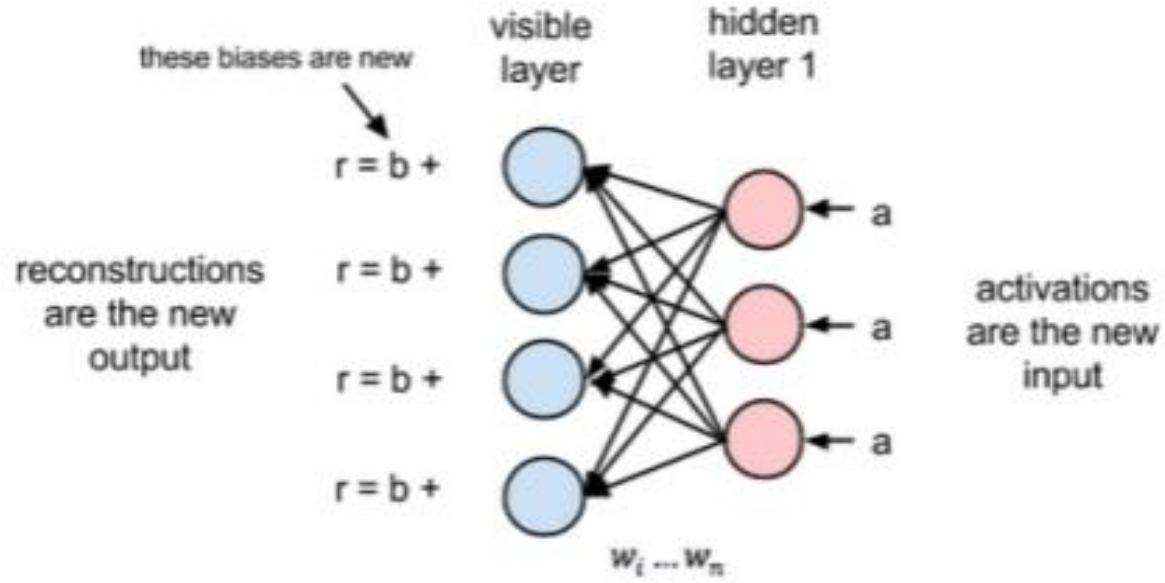
Otras redes: Redes LSTM

Uno de los principales defectos de las redes recurrentes es su **corta memoria** y el **gradiente desvaneciente**. Con el fin de solucionar en parte estos problemas, las redes LSTM (Long Short Time Memory) (Hochreiter and Schmidhuber, 1997) es una modificación de las redes recurrentes a fin de ir "olvidando" la información irrelevante e ir quedándose con la más importante. Funcionan igual que las recurrentes pero en cada paso, la neurona pasa por una célula LSTM, en lugar de ir directamente a la siguiente. Esta célula se encarga de llevar a cero información anterior que no explica fuertemente la respuesta e ir incorporando la información nueva que si explica la respuesta



Otras redes: Máquina de Boltzmann restringida

Una RBM (Restricted Boltzmann Machine) (Smolensky, 1986) es un tipo de red utilizada en problemas no supervisados. Esta arquitectura en particular consiste en una red probabilística, es decir pasa a otra neurona con una probabilidad.
La idea básica de esta arquitectura es "reconstruir el input", de tal forma que la capa de salida es análoga a la capa de input.
Sus principales usos son en filtros colaborativos, agrupación y reducción de dimensionalidad.



Otras redes: Redes generativas antagónicas

Las GAN, por sus siglas en inglés, son un tipo de red generativa, propuesta por Goodfellow el año 2014. Generó una rápida popularidad por la capacidad de "inventar" a partir de información recibida.
Su funcionamiento básico es mediante dos redes, una que se entrena para generar la data y otra que se encarga de discriminar cual data es generada y cual es real, lo cual le da la función de pérdida a la generadora.
Algunas de las cosas que han hecho estas redes es por ejemplo crear un rostro nuevo a partir de fotos o hasta escribir un capítulo de un libro.