

# Redes de aprendizaje profundo

Deep Learning 15/11 Teóricas

☐ <https://drive.google.com/file/d/1HQ33J5jSY1FbmsUN90kdAllIVvIH1kRq/view>

☐ [https://drive.google.com/file/d/1q2KPJamWhExAqjkLIB\\_hNCYIp8seKT3Z/view](https://drive.google.com/file/d/1q2KPJamWhExAqjkLIB_hNCYIp8seKT3Z/view)

## Redes superficiales

### Redes de aprendizaje profundo

¿Para qué puedo usar una red de aprendizaje profundo?

Aprendizaje no supervisado

Aprendizaje supervisado

En general:

Clasificación

Análisis de series de tiempo

### Entrenamiento de Redes Neuronales Profundas

Problemas:

Tiempo de Entrenamiento

Backpropagation → Desvanecimiento del Gradiente

### Restricted Boltzmann Machines

### Deep Belief Nets

### Autoencoders

Usos

### Redes convolucionales

Ejemplo: reconocer esta imagen

Capa convolucional

¿Qué es Convolución?













¿Qué es RELU?

### Pooling

Como se ve por dentro una red convolucional

## Redes superficiales

Vimos las redes neuronales artificiales, que ahora las vamos a llamar superficiales

Estructura	Regiones de Decisión	Problema de la XOR	Clases con Regiones Mezcladas	Formas de Regiones más Generales
1 Capa 	Medio Plano Limitado por un Hiperplano			
2 Capas 	Regiones Cerradas o Convexas			
3 Capas 	Complejidad Arbitraria Limitada por el Número de Neuronas			

Con una sola capa podíamos solamente resolver problemas de separación lineal. Por ejemplo, compuerta AND, OR. Pero no podíamos resolver compuerta XOR, lo que sí se puede con 2 capas.

Hasta 3 capas estamos ante redes superficiales.

## Redes de aprendizaje profundo

Profundo → tiene varias capas

La complejidad de la red es mayor, el entrenamiento de la red es más complejo, pero así también puede resolver problemas más complejos.

Hay diferentes tipos o arquitecturas de redes de aprendizaje profundo:

- Restricted Boltzmann Machine
- Autoencoder
- Deep Belief Network
- Redes convolucionales (Convolutional Net)
- Redes recurrentes (Recurrent Net)
- Recursive Neural Tensor Nets (RNTN)

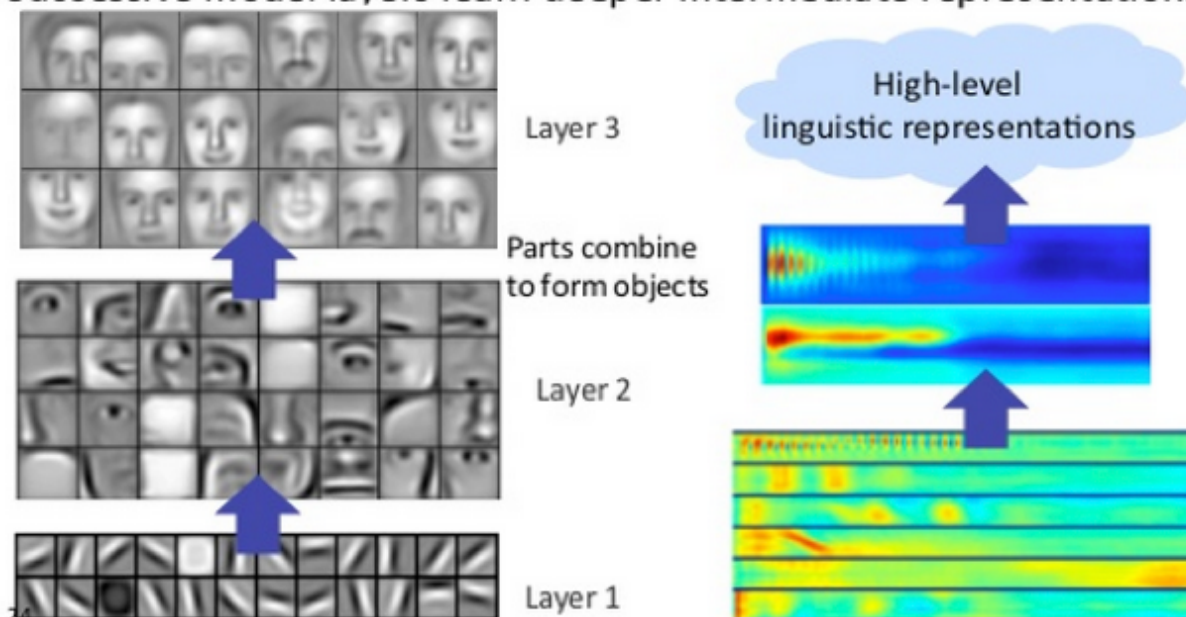
Hay problemas que son complejos, que no se pueden resolver de un solo “golpe” ⇒ esos problemas hay que partirlos en pequeños problemas



La idea principal es que cada capa de la red resuelva una parte del problema

### Ejemplo: reconocimiento de caras

Successive model layers learn deeper intermediate representations



**Prior: underlying factors & concepts compactly expressed w/ multiple levels of abstraction**

1era capa: detecta lineas y orientacion

2da capa: detecta objetos puntuales (ojos, boca, nariz)

3era capa: reconoce caras

# ¿Para qué puedo usar una red de aprendizaje profundo?

## Aprendizaje no supervisado

- Extracción de patrones
- Extracción de características



## Aprendizaje supervisado

- **Procesamiento de texto**  
Análisis de sentimiento, Parsing, Name Entity Recognition
  - Redes recurrentes (a nivel carácter)
  - RNTN
  - Redes convolucionales (para análisis de sentimiento)
- **Reconocimiento de imágenes**
  - Deep Belief Network
  - Redes convolucionales
- **Reconocimiento de objetos**
  - Redes convolucionales
  - RNTN

- **Reconocimiento del habla**
  - Redes recurrentes

## En general:

Podemos agrupar en 2 grupos los problemas que podemos resolver:

- Clasificación
- Analisis de series de tiempo

## Clasificación

- Deep Belief Networks y Perceptrones Multicapa con RELU (Rectified Linear Units)

Dada una observacion, un registro, decidir a que clase pertenece de una serie de clases candidatas

Podemos hacer clasificacion de imagenes (si es un paisaje, si es una persona o un animal, si es de día o de noche)

O hacer reconocimiento de objetos dentro de la imagen



## Análisis de series de tiempo

Recurrent Net (redes recurrentes)

# Entrenamiento de Redes Neuronales Profundas

- ¿Por qué tardaron casi 50 años en aparecer?
- ¿Qué pasa con el método BackPropagation?

## Problemas:

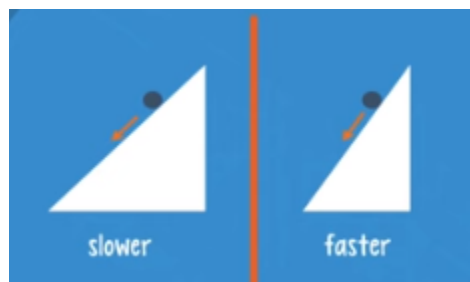
### Tiempo de Entrenamiento

Suelen tardar mucho tiempo en entrenarse las redes neuronales

El entrenamiento de una red de aprendizaje profundo puede tardar meses, pero con modernos GPUs puede tardar menos de una semana.

1 día vs 8 meses

### Backpropagation → Desvanecimiento del Gradiente



Backpropagation iba actualizando los pesos en la red neuronal de la parte de atrás, es decir, desde la capa de salida hacia las capas de entrada.

Va arrastrando los errores ya calculados hacia las capas de entrada. El error se calcula desde la salida.

Tenemos una salida dada, un valor predicho y un valor real, la diferencia es el error.

A medida que voy mostrando ese error para atrás, tengo que ir arrastrando los errores de cada una de las salidas de las redes. Eso hace que el gradiente se fuera haciendo más pequeño a medida que me iba acercando a las capas iniciales, que son las últimas

en actualizar

⇒ la actualización de los pesos era mínima.  $\text{Peso actual} + \text{factor de aprendizaje}$

Nunca se terminaban de actualizar los pesos.

Esto hace que entrenar redes de muchas capas sea mucho más costoso, hay más ciclos de GPU

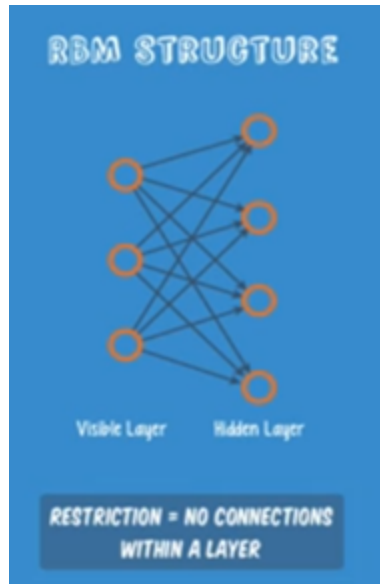
**Todo cambió entre 2006 y 2007 gracias a 3 papers publicados por Hinton, Lecun y Bengio**

## Restricted Boltzmann Machines

**Geoff Hinton** fue el primero en encontrar una solución al problema de las redes de aprendizaje profundo. Es considerado el **padre del Deep Learning**.

Esta red tiene dos capas

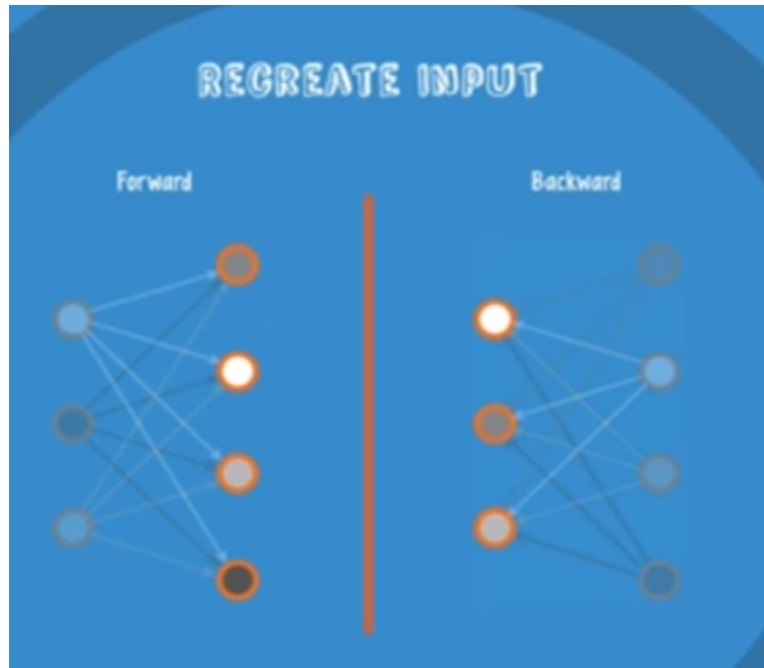
Es la clave para la construcción de la primera red de aprendizaje profundo



### Como funcionan?

1. Ejecutar una entrada en sentido directo (poner un dato, pasa por la red y da una salida)
2. Hacer la ejecución en sentido inverso (recuperar la entrada original)
3. Comparar con **KL Divergence** y ajustar pesos y bias, hasta que las salidas de la red invertida coinciden con las entradas, o se acercan lo más posible

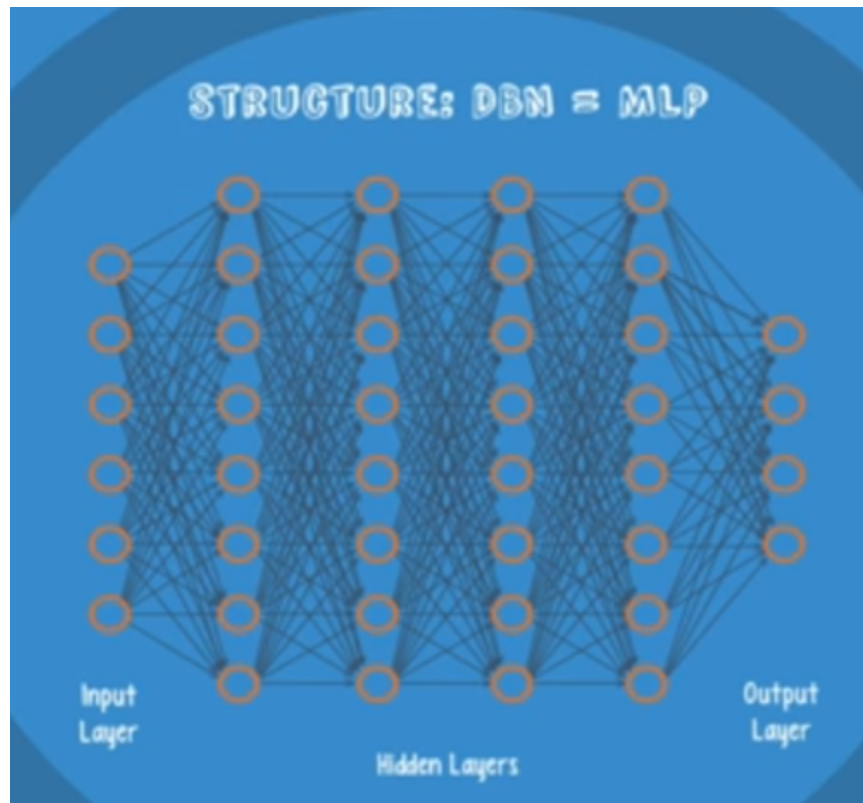




## Deep Belief Nets

Concebidas por **Geoff Hinton** como una alternativa al backpropagation. Por ello fue inmediatamente contratado por Google.

Una vez que tenemos entrenada una **Restricted Boltzmann Machines**, vamos a apilarlas y vamos a tener una red **Deep Belief Net** → es exactamente igual a un **perceptrón multicapa**, pero su método de entrenamiento es completamente diferente



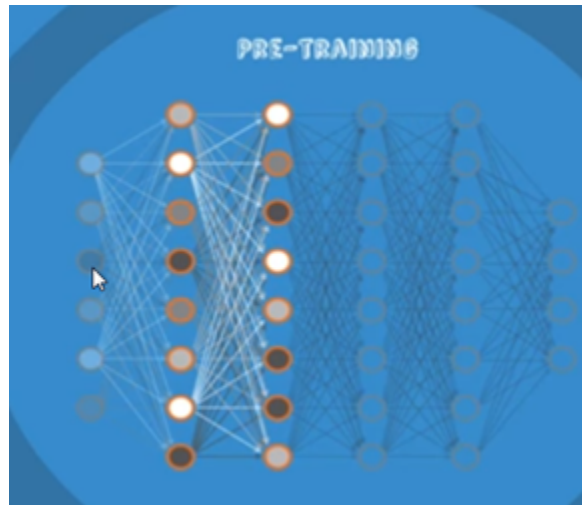
No se entrena con Backpropagation

Se entrena de a partes

Primera RBM



Sgunda RBM



Voy entrenando cada dos capas

- Cada capa aprende el input entero.  
(Distinto a las redes convolucionales)
- Cuando el entrenamiento concluye, la red aprendió a detectar patrones inherentes en los datos, pero aún no sabemos nada acerca de esos patrones.

Encontro mecanismos tales que le permiten recuperar los datos de entrada porque conoce las partes mas importantes, los patrones, pero todavia no sabemos que es lo que detecto, encontro

- Así que hay una segunda parte de entrenamiento supervisado. Pero con una cantidad pequeña de datos.  
Podemos meter un clasificador al final

### ▼ Una pregunta acerca de deep belief nets & autoencoders

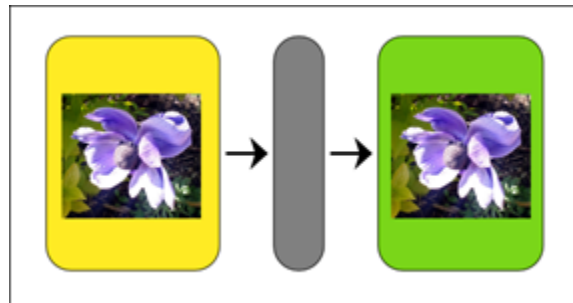
Por definicion un autoencoder en si tiene la misma cantidad de neuronas de entrada y de salida. Porque decimos que las deep belief nets son autoencoders si en este caso no tienen las mismas neuronas de entrada y salida??

Podes usar redes de tipo Deep Belief para problemas de clasificación, en ese caso tu salida será diferente a la entrada, tendras unas pocas neuronas de salida, una por cada categoria, igual que en un perceptron multicapa.

Pero puedes usarlos para crear un autoencoder, en este caso (el de los autoencoders), la red es simétrica (generalmente) desde el centro hacia los bordes, teniendo la misma cantidad de neuronas de entrada que de salida

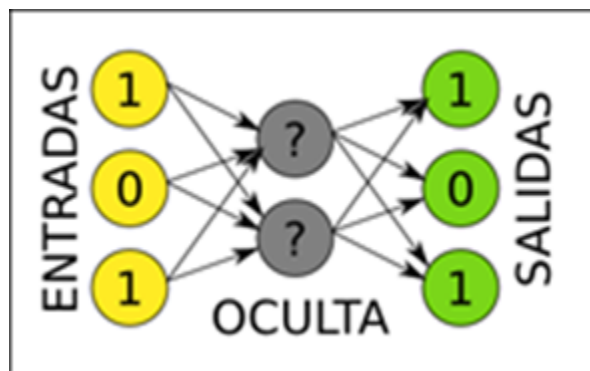
## Autoencoders

- Restricted Boltzmann Machines (**RBM**) y **Deep Belief Nets** son **Autoencoders**
- Un autoencoder aprende a producir en la salida exactamente la misma información que recibe a la entrada
- Entrada y salida igual número de neuronas



## Usos

- Compresor
- Reducción de dimensionalidad
  - 28x28 píxeles tiene 750 neuronas => puedo reducirla a solo 30 números teniendo toda la info importante contenida ahí

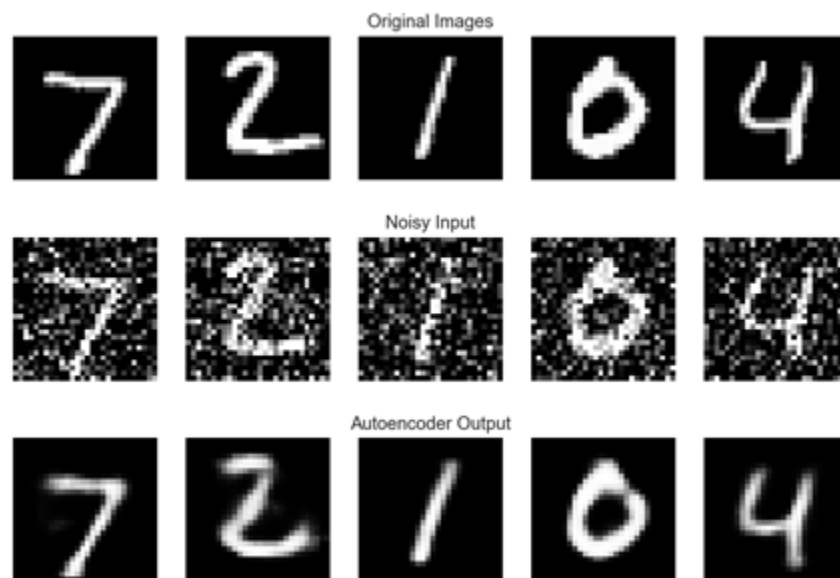


Capa de entrada es 750, las capas ocultas son cada vez mas chica hasta que llegan a 30 y la salida es de nuevo 750. Con lo cual recuperé la entrada original

- Eliminación de ruido

### Entrenamiento:

Se pueden entrenar con **Backpropagation**, pero utilizando una métrica particular llamada “**Loss**” (Cantidad de información que la red perdió al tratar de reconstruir el *input*)



## Redes convolucionales

### Convolutional Neural Nets (CNN)

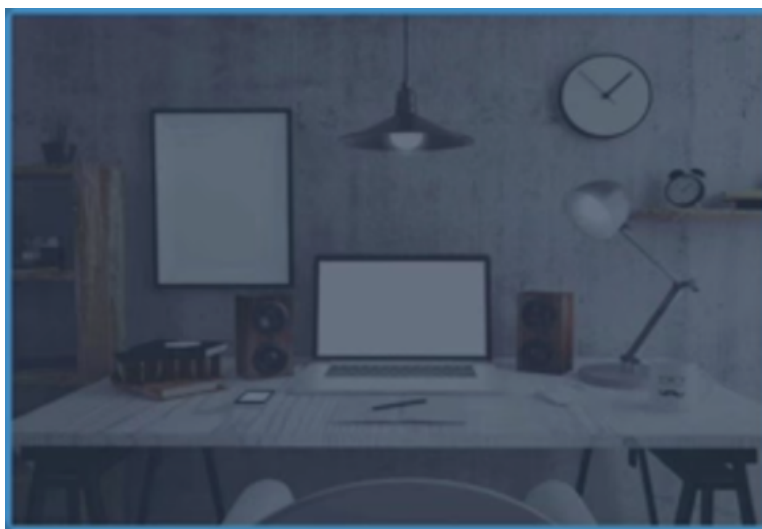
- Dominan completamente la visión espacial.  
Toda inteligencia artificial que trabaja con imagenes, reconocimiento de imagenes trabaja con CNN
- Desarrolladas por **Yann Lecun** of New York University.  
departamento de **AI de Facebook** (que usa estas redes para detectar caras en las fotos)
- Después de que aunaron esfuerzos en 2015, Microsoft, Google y Baidu lograron que una **computadora derrote a un humano** en un concurso de reconocimiento

visual de objetos. La primera vez en toda la historia de IA.

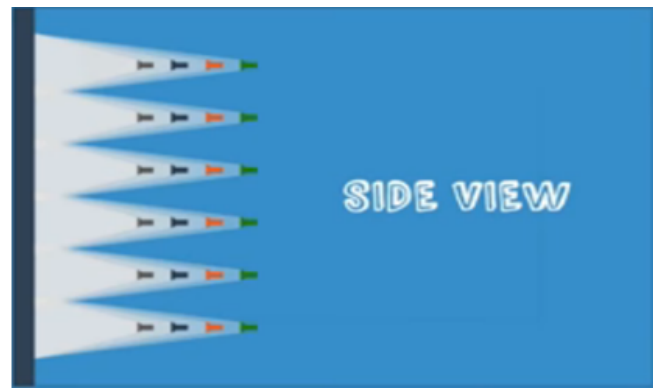
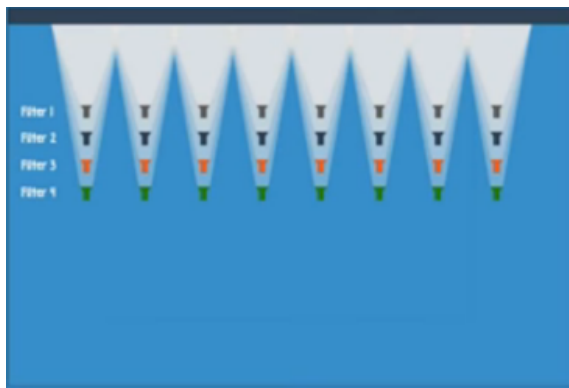
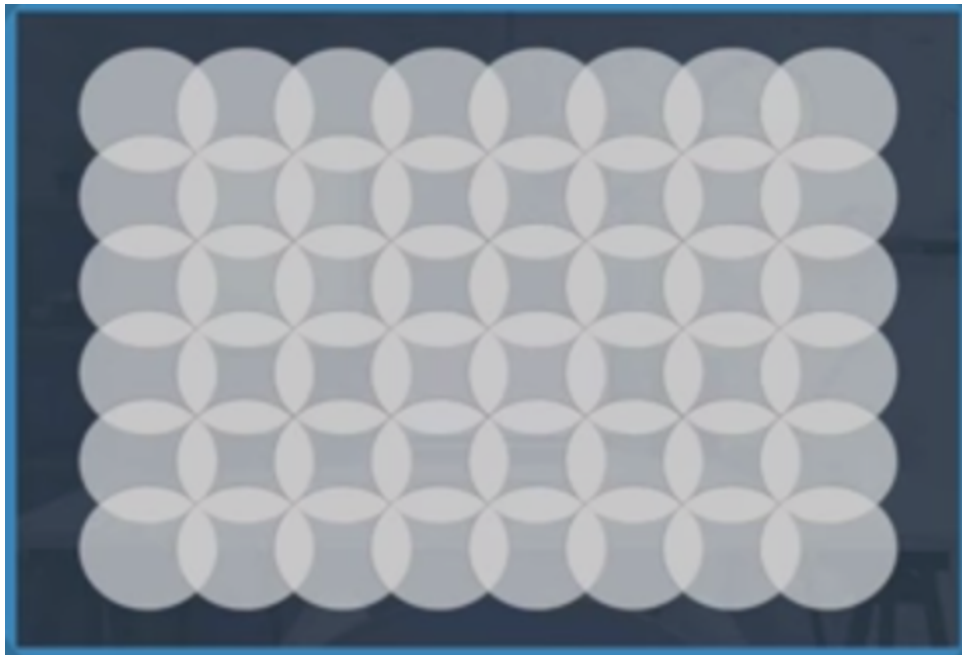


Replica de como funciona el circuito neuronal que va desde el ojo hasta el cerebro en la parte de reconocimiento de imagenes.

### Ejemplo: reconocer esta imagen



Tenemos pequeñas linternas, ojitos que cada uno va a mirar una parte de la imagen, va a iluminar un sector. Con superosicion



Estan varias linternas superpuestas, no esta conectadas entre si

Las linternas son como filtros

Es como una matriz tridimensional de pequeñas linternas iluminando la imagen

## Capa convolucional

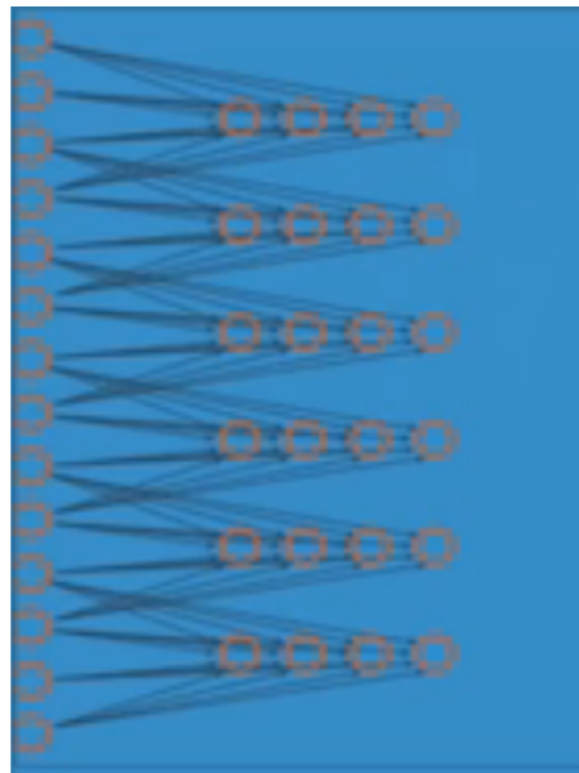
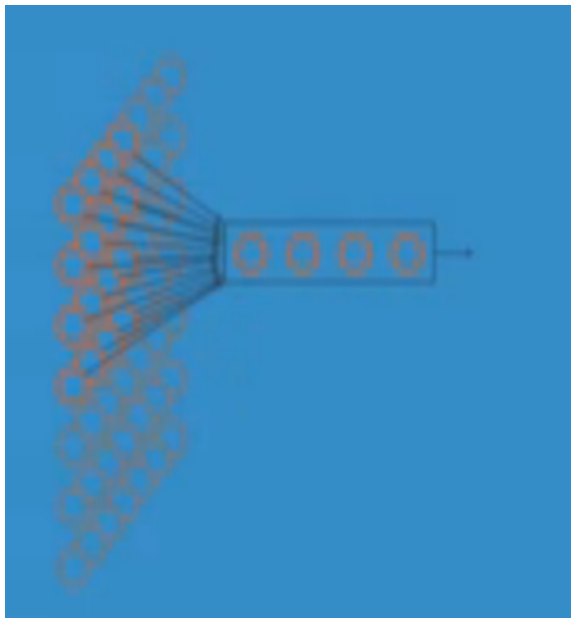
Una neurona que esta conectada con todas las neuronas de entrada (los pixeles)

Si tengo una img de 640x480  $\Rightarrow$  tendre un array de neuronas de entrada de 640x480 y a cada neurona le va a llegar la info de un pixel

Si tengo una img RGB voy a tener 3 canales de color  $\Rightarrow$  tengo lo mismo pero triplicado

Si tengo en blanco y negro es un solo canal, cada pixel puede solo tomar valor de 0 a 255 (la intensidad). En general, se utiliza para reducir los tiempos de la red neuronal y cuando no me interese el color

En la imagen una neurona esta conectada a 4 pixeles



### **Esto es una capa convolucional**

Tiene una particularidad

Todas las primeras neuronas tienen los mismos valores, los mismos pesos

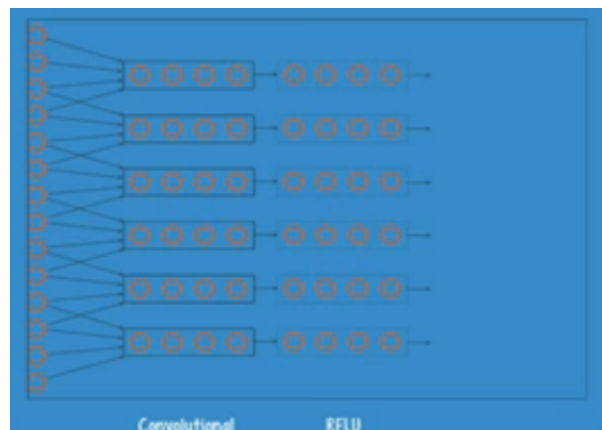




Despues de esta capa viene una capa de activacion

⇒ **Capa RELU: Rectified Linear Unit**

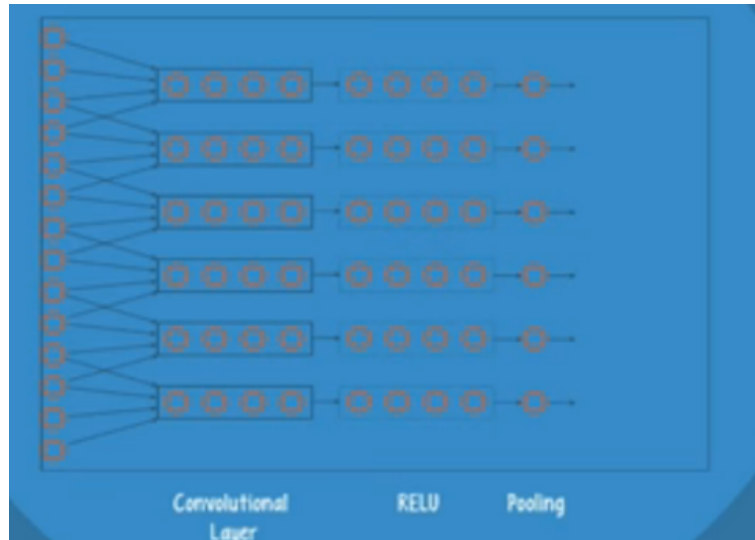
Esta capa se entrena con  
Backpropagation



Luego viene una capa  
adicional

⇒ **Capa Pooling**

Hace una Reducción en la  
dimensionalidad

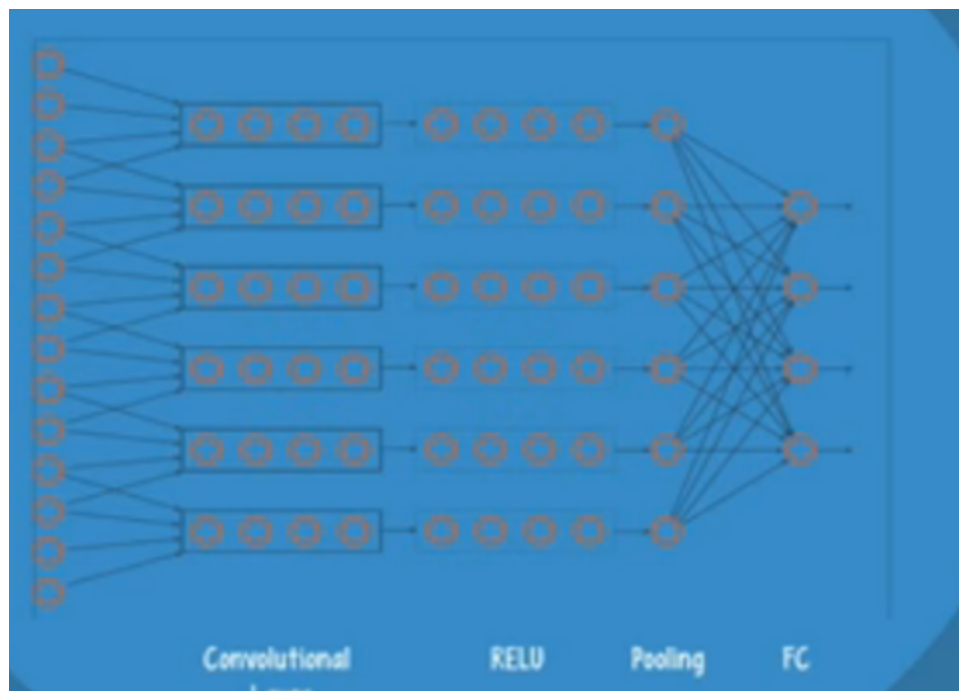


Finalmente, tenemos la **Capa Fully Connected** o capa densa

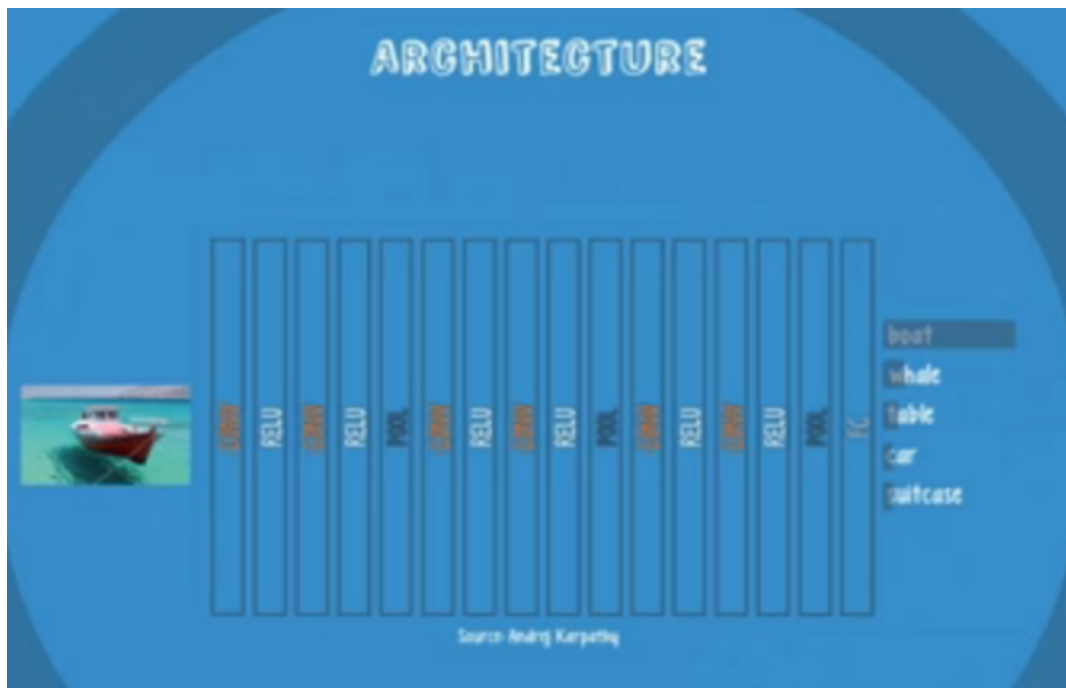
Es la capa de perceptron, sencillo de dos capas

Sirve para clasificar los ejemplos, la salida de la red

No va a estar conectada esta red neuronal a la imagen, a las neuronas de entrada, sino que la entrada es la salida de la red convolucional



La arquitectura típica de una **red convolucional** es varias veces la secuencia de capa convolucional, RELU, pooling, y así sucesivamente



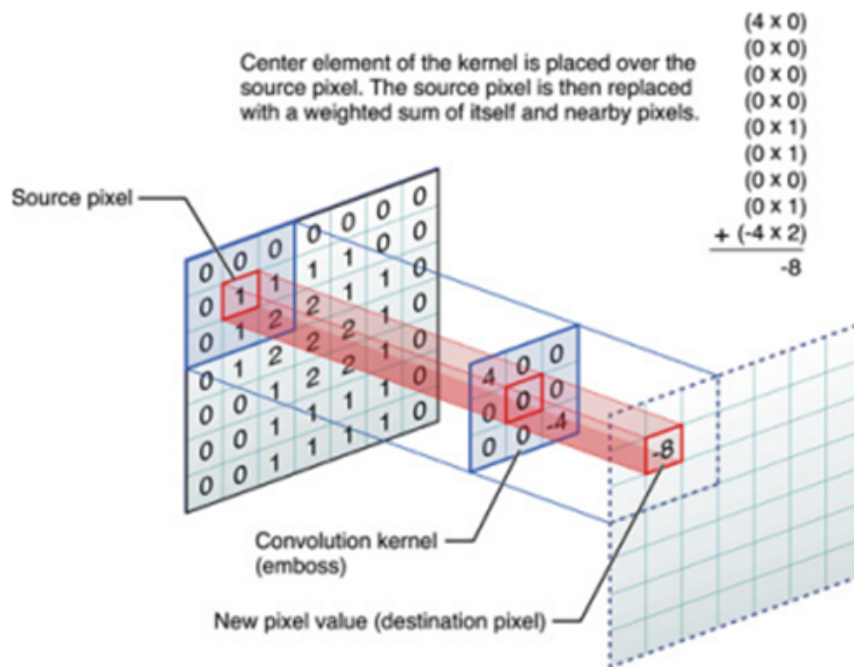
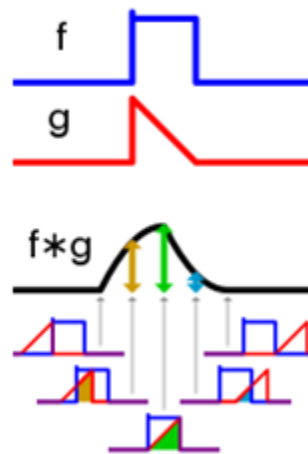
Múltiples capas: Convolucional, RELU y Pooling pero solo 1 FC

## ▼ ¿Qué es Convolución?

Es una manera de combinar dos funciones en una nueva

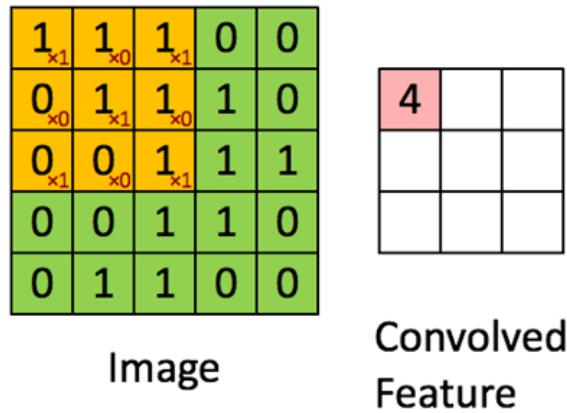
Operacion que se aplica entre funciones

## Convolution

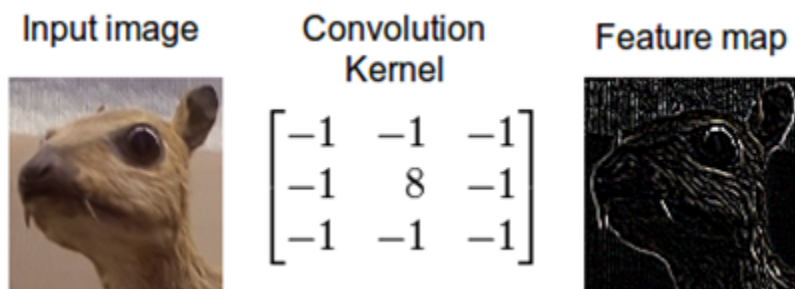


Lo que tenemos es una matriz de entrada de pixeles

Vamos a hacer una “convolucion” entre esa matriz y el kernel convolucional (pesos que conectan c/u de estas neuronas/pixeles a la neurona de activacion)



Suma los que son x1

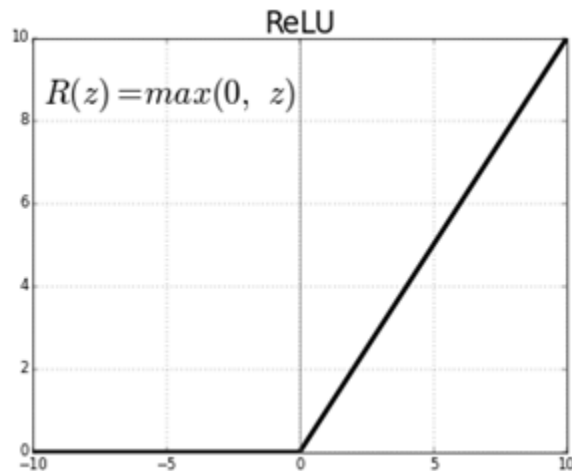


Este Kernel detecta las líneas

## ▼ ¿Qué es RELU?

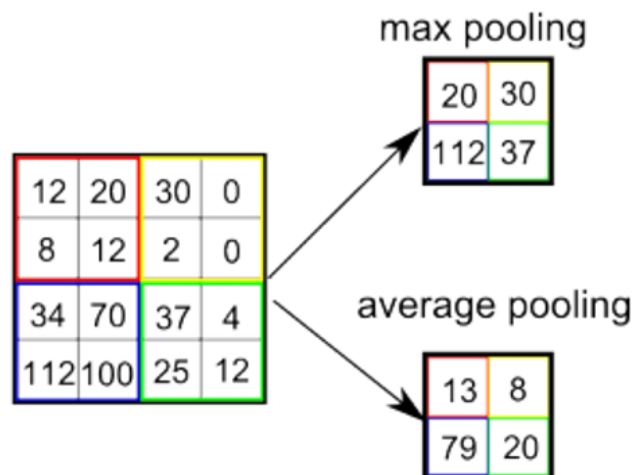
Relu es básicamente la función de activación la cual es:  $y = \max(0, x)$

Sado un x de entrada, devuelve x si  $x > 0$  sino devuelve 0:



## ▼ Pooling

La capa de reduccion de la dimensionalidad



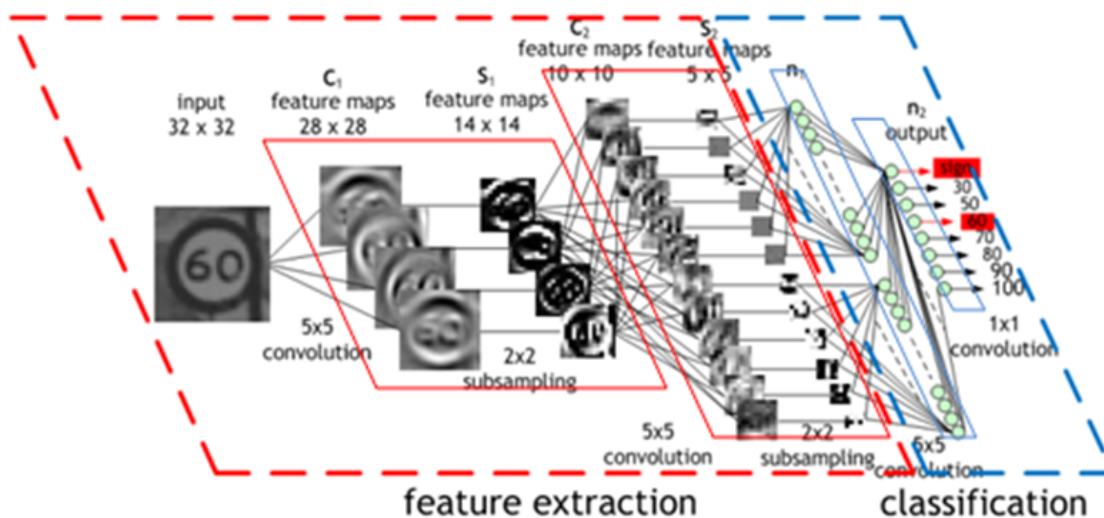
Tiene dos forma de funcionar:

**Max pooling:** de un recuadro de 2x2 me quedo con el de mayor valor

**Average pooling:** sumo todos los valores y los divido por 4, es decir, me quedo con el promedio

## Como se ve por dentro una red convolucional

Detectar señales de transito



Subsampling → RELU y pooling

Al final pone el perceptron multicapa, la capa fully connected. Donde voy a tener varias neuronas para poder procesar todas estas imagenes

La salida son 8 neuronas

- Si enciende la primera, significa que es una señal
- Las siguientes van a indicar que tipo de señal es