# Procesamiento de imágenes: CNN

**Diapositivas**

En la parte anterior del curso analizamos el funcionamiento de las redes neuronales. Cómo estos modelos son capaces de clasificar puntos que vienen representados en forma tabular.

En esta parte del curso vamos a entrar de lleno en el campo de la visión por computador.

Enseñar a un ordenador a que sea capaz de ver.

**Diapositiva 1**

Esto nos permitiría resolver problemas de clasificación aplicados a imágenes.

¿Es esta imagen un gato o un perro?

Aparece un coche en esta imagen o no?

Qué tipo de animal se muestra en la imagen?

O problemas que van más allá de la clasificación

¿Dónde está el coche en la imagen? (detección de objetos)

¿Cuáles son las diferentes secciones de esta imagen? (segmentación)

¿Qué diferencia hay entre estas imágenes? (detección de anomalías)

**Diapositiva 2 (aplicaciones)**

Es muy útil en el día a día. Reconocimiento de caras, filtros, cámaras de vigilancia autónoma…

**Diapositiva 3 (qué es una imagen)**

Vamos a analizar primero cómo es una imagen y los problemas de utilizar una red neuronal estándar para procesarla.

Esta imagen por ejemplo, nuestro cerebro es capaz de procesarla muy bien. Sabemos que esta imagen representa la cara de una persona. Esto lo sabemos porque nuestros ojos y nuestro cerebro han escaneado la imagen y han visto una boca, dos orejas, dos ojos, una forma esférica que parece una cabeza, cejas… Hemos detectado patrones en lo que estamos viendo y hemos llegado a la conclusión correcta de que es una cara.

Nosotros, como seres humanos, somos capaces de ver la imagen desde nuestro espacio 3D. A simple vista podemos ver que en esta imagen hay una oreja, una boca, un ojo… sin tener que analizar el valor de cada píxel. Pero tenemos que recordar que a un ordenador le llega, para cada imagen, una lista de valores de cada píxel.

Una imagen viene dada normalmente en un espacio bidimensional donde cada píxel puede representarse en tonalidad de gris (gray image) utilizando un único valor o en forma de color (RGB), utilizando tres valores.

Si en vez de una imagen os doy una lista de valores de píxlees, ninguno sería capaz de decirme qué representa esta imagen.

Pero bueno para eso están las redes neuronales, para pasarle datos a mansalva y que haga lo que tenga que hacer para aprender. Cómo podemos diseñar una red neuronal que pueda procesar imágenes de forma eficiente.

Podríamos utilizar la misma arquitectura que vimos en el curso pasado. Cada píxel se le asocia una neurona de entrada.

Dos problemas:

1. Eficiencia: En los datos tabulares estamos hablando de que usábamos instancias con dimensiones pequeñas. Una imagen pequeña tiene 256 x 256 x 3 píxeles, es decir, 196.608. Y si cogemos imágenes de alta definición, 1024 x 1024 x 3 = 1.048.576 valores. Si cada neurona tiene un peso asociado que debe aprender para cada valor de entrada, el número de pesos se dispara desde las primeras capas, impidiendo utilizar redes grandes.
2. Visión espacial: es difícil que una red estándar sea capaz de captar las relaciones espaciales entre los píxeles de la imagen y, detectar por ejemplo, que una agrupación concreta de píxeles forman un ojo. No hay mecanismos específicos que permitan adquirir este conocimiento.

Surgen las redes neuronales convolucionales para resolver estos dos problemas.

La principal diferencia entre una capa densamente conectada y otra que se basa en la operación de convolución, que llamaremos capa convolucional, es que la capa densa aprende patrones globales en su espacio global de entrada, mientras que una capa convolucional es capaz de aprender patrones locales en pequeñas ventanas de dos dimensiones. Es un mecanismo que permite a la red neuronal aprender características o rasgos en imágenes como aristas, líneas, formas,...

**Kernel (D.13)**

Sirve para Analizar un píxel en base a sus vecinos

Dependiendo del valor que se le asigne al kernel, este hará una cosa u otra.

D.15

Así es como funcionaba el algoritmo de Viola-Jones para detección de caras.

Así que, en general, los kernel aplicados a imágenes sirven para extraer características de las mismas que luego puedan utilizarse para otras tareas, como detectar caras, bordes, objetos…

y no nos suena de algo esa representación del conocimiento? Es justamente lo que hace una red neuronal. Dada una entrada, en las capas intermedas consigue obtener una representación de la entrada, cada vez más abstracta, hasta que llega a la capa de salida en la que se clasifica o se predice a partir de la entrada.

Entonces, cómo podemos unir este concepto de aprender representaciones en la red neuronal y los kernel?

Hiperparámetros de una CNN

<https://poloclub.github.io/cnn-explainer/>

**Notebook**

Construcción de una CNN en pytorch lightning

**Caso práctico**

<https://web.archive.org/web/20220706122225/https://torres.ai/deep-learning-inteligencia-artificial-keras/>

<https://github.com/mrdbourke/pytorch-deep-learning/blob/main/03_pytorch_computer_vision.ipynb>

<https://github.com/rasbt/deeplearning-models/blob/master/pytorch-lightning_ipynb/cnn/cnn-alexnet-cifar10.ipynb>

3D

<https://github.com/OValery16/Tutorial-about-3D-convolutional-network>