

Redes Neuronales Convolucionales

Deep Learning y Series de tiempo



Marco Teran
Universidad Sergio Arboleda

2023

Contenido

- 1 Visión por computador
- 2 Características visuales de aprendizaje
- 3 Extracción y convolución de características
- 4 Convolutional Neural Networks (CNNs)
 - CNN para la clasificación: ImageNet
- 5 Aplicaciones de las CNN
- 6 Impacto y resumen
- 7 Resumen



Lo que las computadoras ven

Las imágenes son números



(a)

157	153	174	168	150	152	129	151	172	161	156	156
155	182	163	74	75	62	93	17	110	210	180	154
180	180	50	14	34	6	10	33	48	106	159	181
256	109	5	124	131	111	120	204	166	15	56	180
194	68	137	251	237	239	239	228	227	87	71	201
172	105	207	233	233	214	220	239	228	98	74	206
188	88	179	209	185	215	211	158	139	75	20	169
189	97	165	84	10	168	134	11	31	62	22	148
199	168	191	193	158	227	178	143	182	106	36	190
205	174	155	252	236	231	149	178	228	43	95	234
190	216	116	149	236	187	85	150	79	38	218	241
190	224	147	108	227	210	127	102	36	101	255	224
190	214	173	66	103	143	95	50	2	109	249	215
187	196	235	75	1	81	47	0	5	217	255	211
183	292	237	145	0	6	12	108	200	138	243	236
195	206	123	207	177	121	123	200	175	13	96	218

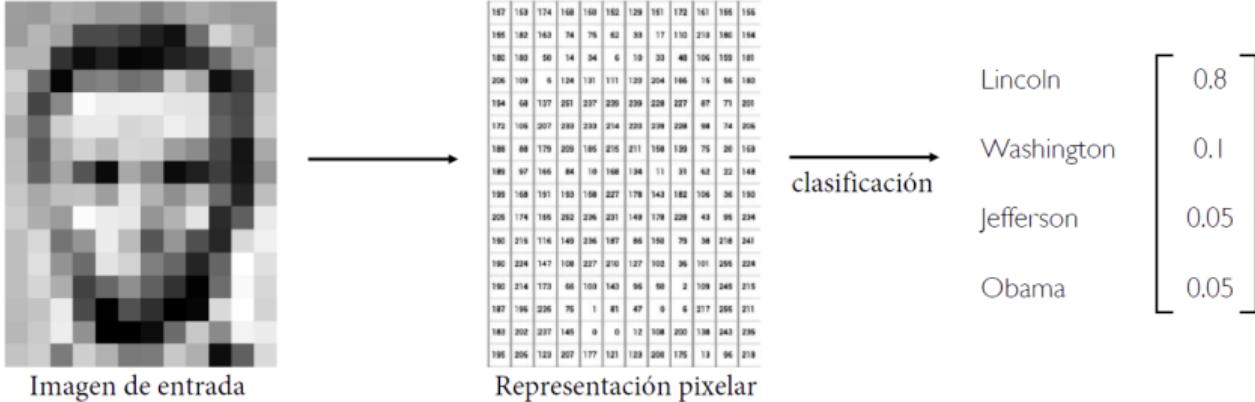
(b)

157	153	174	168	150	152	129	151	172	161	156	156
155	182	163	74	75	62	93	17	110	210	180	154
180	180	50	14	34	6	10	33	48	106	159	181
206	109	5	124	131	111	120	204	166	15	56	180
194	68	137	251	237	239	239	228	227	87	71	201
172	105	207	233	233	214	220	239	228	98	74	206
188	88	179	209	185	215	211	158	139	75	20	169
189	97	165	84	10	168	134	11	31	62	22	148
199	168	191	193	158	227	178	143	182	106	36	190
205	174	155	252	236	231	149	178	228	43	95	234
190	216	116	149	236	187	85	150	79	38	218	241
190	224	147	108	227	210	127	102	36	101	255	224
190	214	173	66	103	143	95	50	2	109	249	215
187	196	235	75	1	81	47	0	5	217	255	211
183	292	237	145	0	6	12	108	200	138	243	236
195	206	123	207	177	121	123	200	175	13	96	218

(c) Lo que el computador ve

¡Una imagen es sólo una matriz de números [0,255]!
es decir, 1080x1080x3 para una imagen **RGB**

Tareas de la visión por computador



- **Regresión:** la variable de salida toma un valor continuo
- **Clasificación:** la variable de salida se toma como la *etiqueta* de clase.
Puede producir la *probabilidad* de pertenecer a una clase particular

Detección de características de alto nivel

Identifiquemos las características de cada imagen



(d) Nariz, Ojos, Boca



(e) Llantas, Placa, Luces
delanteras



(f) Puerta, Ventana, Pisos

Extracción manual de características

Conocimiento
del dominio

Definición de
características

Detección de
características para
clasificación



Scale variation



Deformation



Occlusion



Illumination conditions



Background clutter

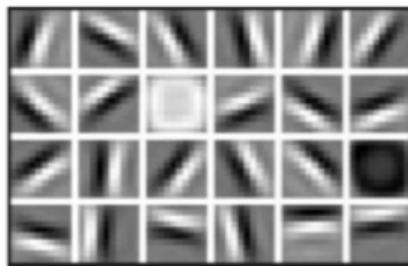


Intra-class variation



Representaciones de las características de aprendizaje

¿Podemos aprender una jerarquía de características directamente de los datos en lugar de la ingeniería manual?



(g) Características de bajo nivel: Líneas y bordes



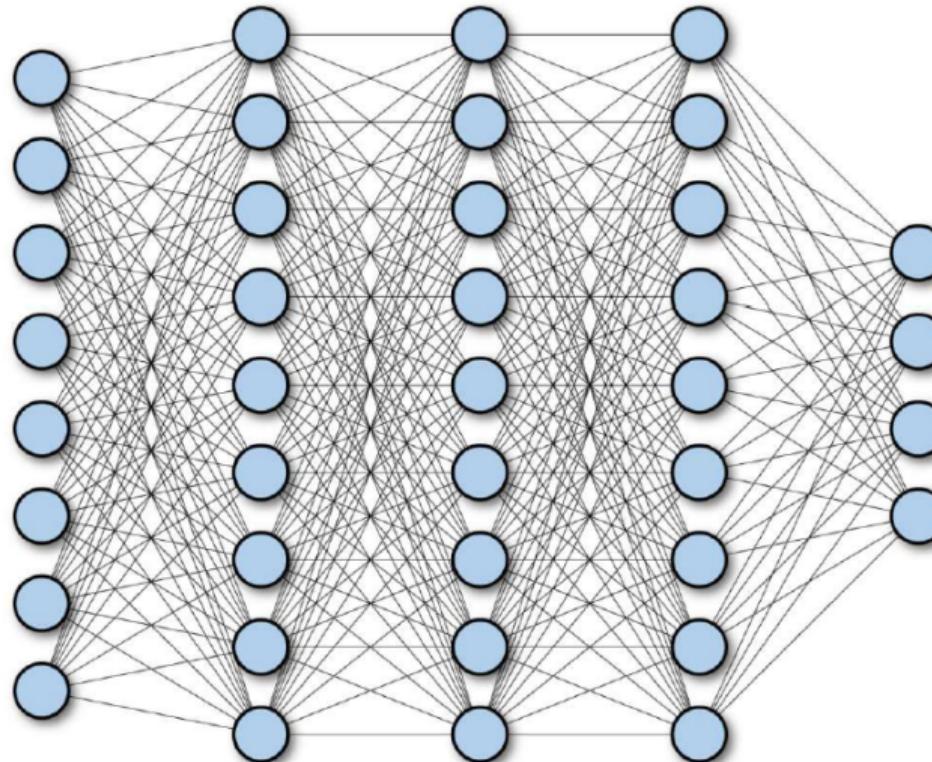
(h) Características de nivel medio: Ojos, nariz y oídos



(i) Características de alto nivel: Estructura facial

Características visuales de aprendizaje

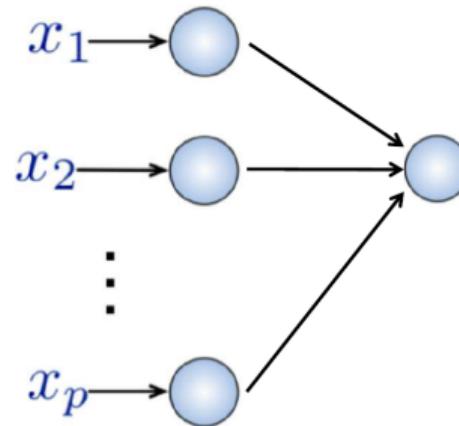
Red neuronal completamente conectada (DNN)



Red neuronal completamente conectada

Entrada:

- Imagen 2D
- Vector de valores de pixel



Totalmente conectada:

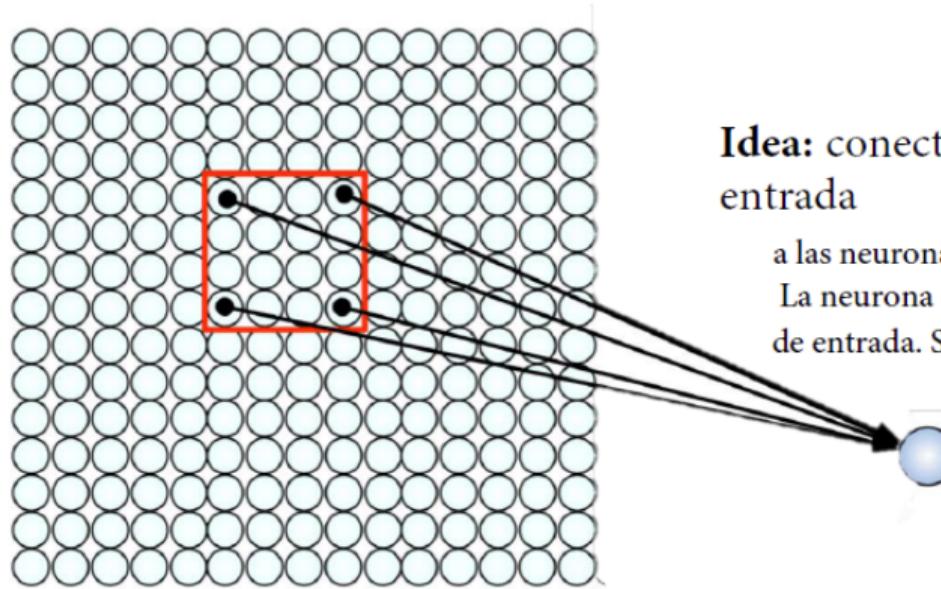
- Conectar la neurona de la capa oculta a todas las neuronas de la capa de entrada
- ¡No hay información espacial!
- ¡Y muchos, muchos parámetros!

¿Cómo podemos usar la estructura espacial en la entrada para informar la arquitectura de la red?

Utilizando información espacial

Entrada:

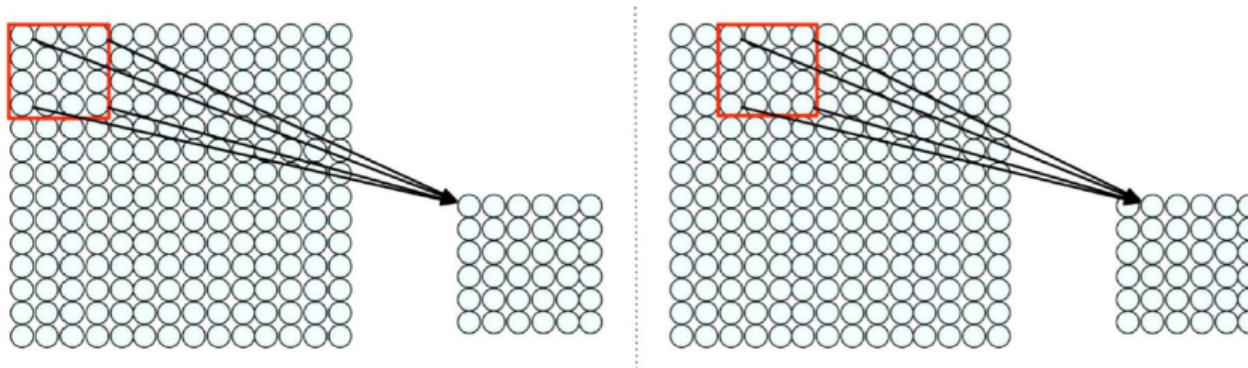
- Imagen 2D
- Vector de valores de pixel



Idea: conectar parches de entrada

a las neuronas en una capa oculta.
La neurona conectada a la región de entrada. Sólo "ve" estos valores.

Utilizando información espacial

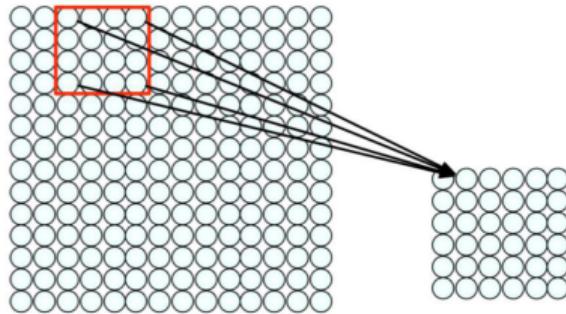


- Conecta el parche en la capa de entrada a una sola neurona en la capa siguiente.
- Utilice una ventana deslizante para definir las conexiones. ¿Cómo podemos ponderar el parche para detectar características particulares?

Aplicación de filtros para extraer características

- 1 Aplicar un conjunto de pesos - **un filtro** - para extraer las **características locales**
- 2 Usar **múltiples filtros** para extraer diferentes características
- 3 Compartir espacialmente los parámetros de cada filtro
 - (las características que importan en una parte del insumo deben importar en otra parte)

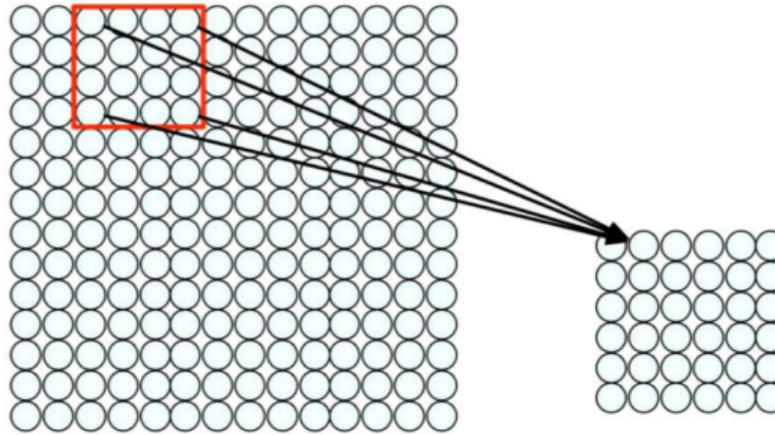
Aplicación de filtros para extraer características



- Filtro de tamaño 4x4: 16 pesos diferentes
- Aplica este mismo filtro a los parches de 4x4 en la entrada
- Cambie 2 píxeles para el próximo parche

Esta operación "irregular" es una convolución

Aplicación de filtros para extraer características



- 1 Aplicar un conjunto de pesos - un filtro - para extraer las **características locales**
- 2 Usar **múltiples filtros** para extraer diferentes características
- 3 Compartir espacialmente los parámetros de cada filtro

Extracción y convolución de características: Un caso de estudio

X o X?

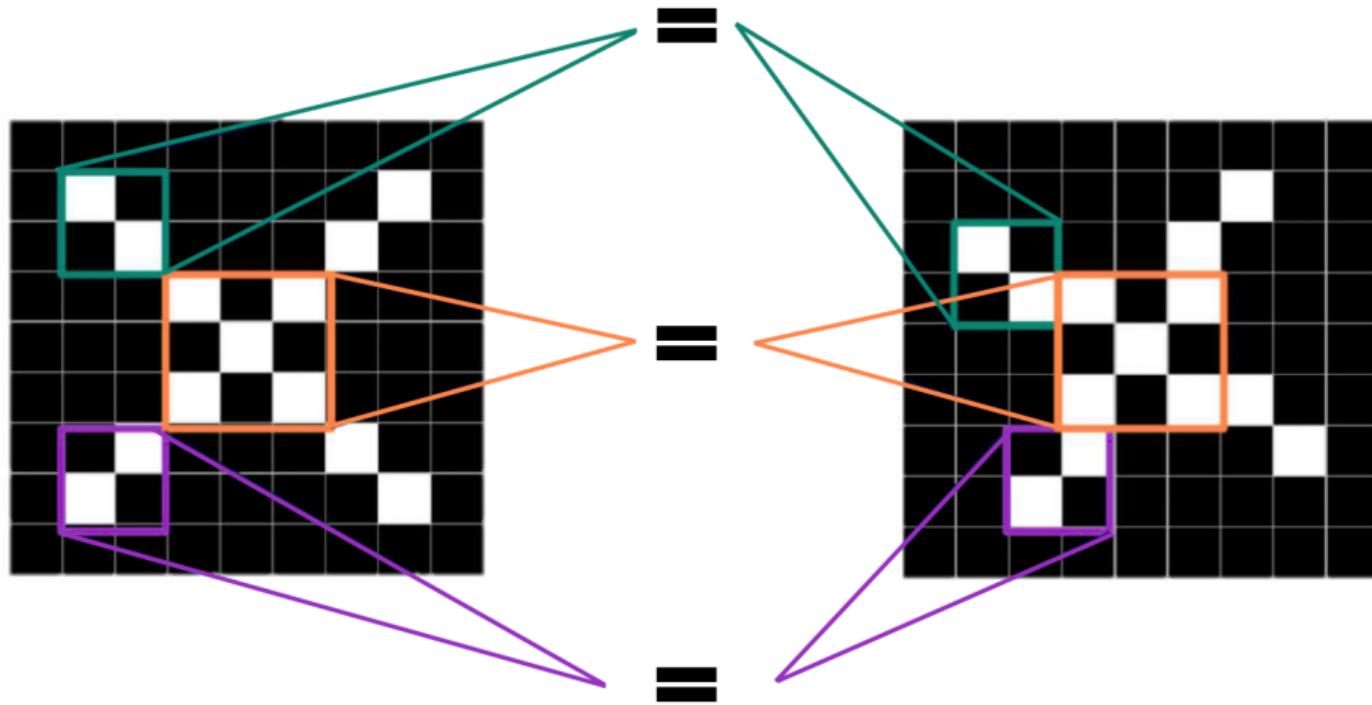


-1	-1	-1	-1	-1	-1	-1	-1	-1
-1	1	-1	-1	-1	-1	-1	1	-1
-1	-1	1	-1	-1	-1	1	-1	-1
-1	-1	-1	1	-1	1	-1	-1	-1
-1	-1	-1	-1	1	-1	-1	-1	-1
-1	-1	-1	1	-1	1	-1	-1	-1
-1	-1	1	-1	-1	-1	1	-1	-1
-1	1	-1	-1	-1	-1	1	-1	-1
-1	-1	-1	-1	-1	-1	-1	-1	-1

-1	-1	-1	-1	-1	-1	-1	-1	-1
-1	-1	-1	-1	-1	-1	-1	1	-1
-1	1	-1	-1	-1	-1	1	-1	-1
-1	-1	1	1	-1	-1	1	-1	-1
-1	-1	-1	-1	1	-1	-1	-1	-1
-1	-1	-1	-1	1	-1	1	-1	-1
-1	-1	-1	1	-1	1	1	-1	-1
-1	-1	-1	1	-1	-1	-1	1	-1
-1	-1	-1	-1	-1	-1	-1	-1	-1

La imagen se representa como una matriz de valores de píxeles... ¡y las computadoras son literales!
Queremos ser capaces de clasificar una X como una X aunque esté desplazada, encogida, rotada, deformada.

Características de X



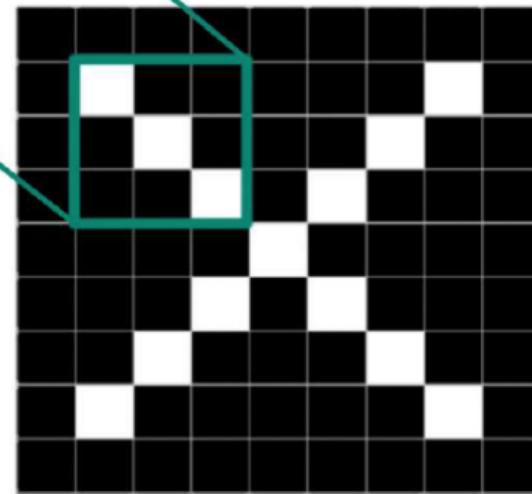
Filtros para detectar las características de X

filtros

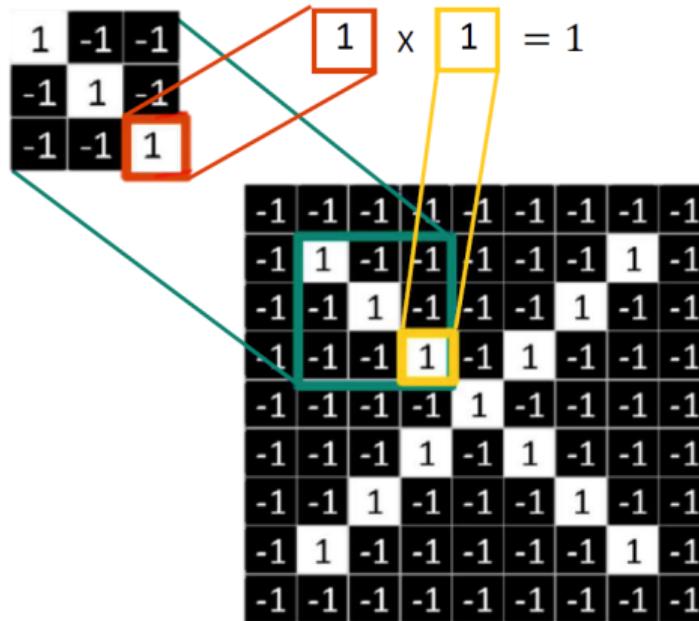
$$\begin{bmatrix} 1 & -1 & -1 \\ -1 & 1 & -1 \\ -1 & -1 & 1 \end{bmatrix}$$

$$\begin{bmatrix} 1 & -1 & 1 \\ -1 & 1 & -1 \\ 1 & -1 & 1 \end{bmatrix}$$

$$\begin{bmatrix} -1 & -1 & 1 \\ -1 & 1 & -1 \\ 1 & -1 & -1 \end{bmatrix}$$



La operación de convolución



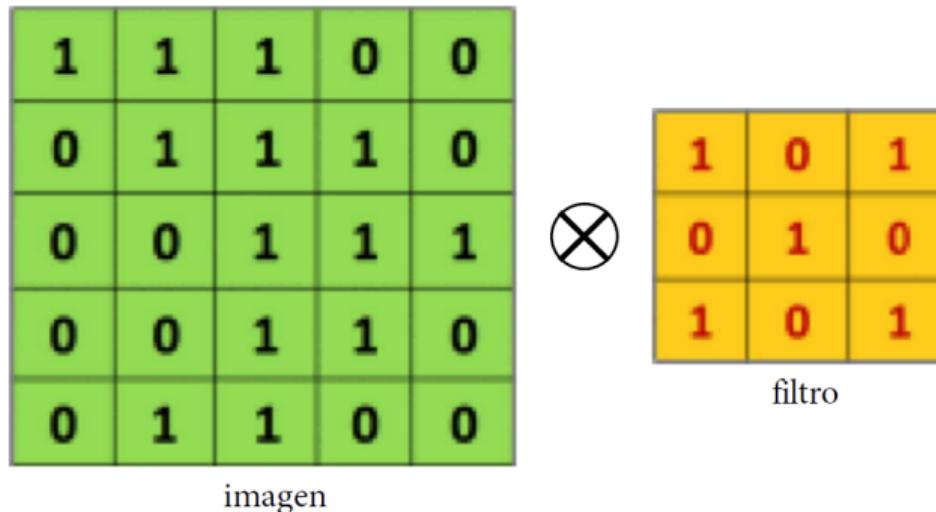
los elementos se multiplican

agregar salidas

$$\odot \begin{array}{|c|c|c|} \hline 1 & 1 & 1 \\ \hline 1 & 1 & 1 \\ \hline 1 & 1 & 1 \\ \hline \end{array} = 9$$

La operación de convolución

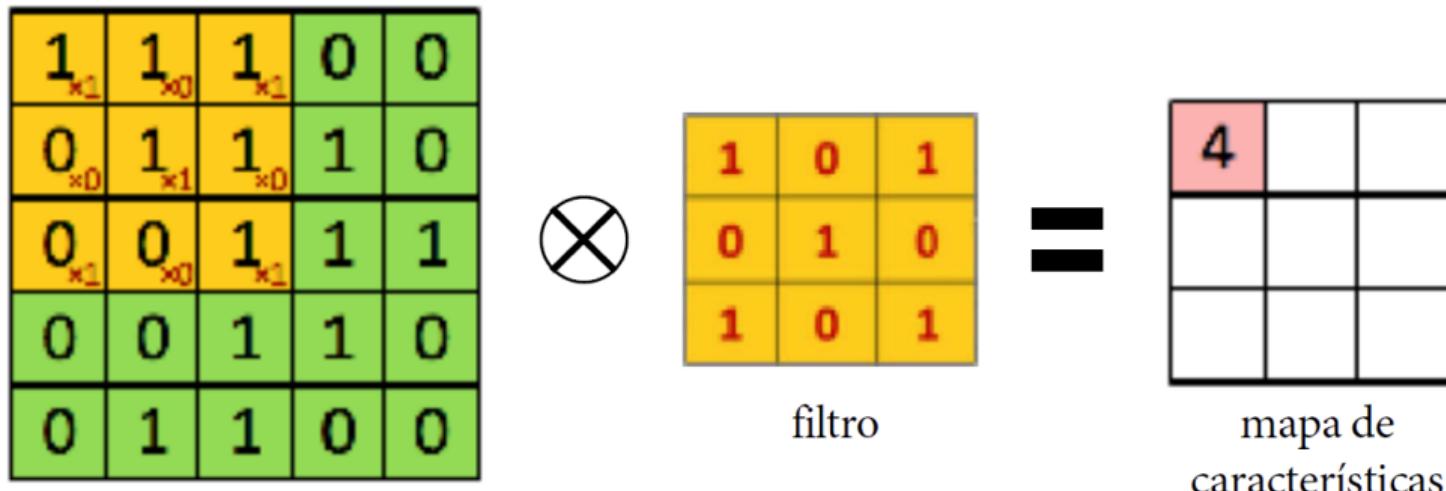
Supongamos que queremos calcular la convolución de una imagen de 5x5 y un filtro de 3x3:



Deslizamos el filtro 3x3 sobre la imagen de entrada, multiplicamos los elementos y sumamos las salidas...

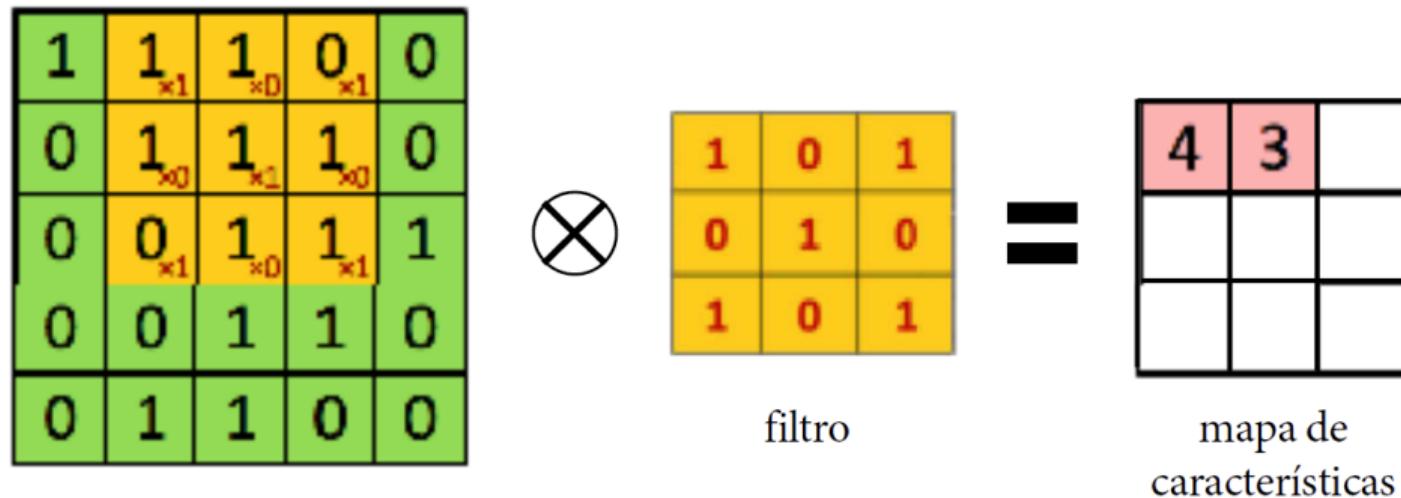
La operación de convolución

Deslizamos el filtro 3x3 sobre la imagen de entrada, multiplicamos los elementos y sumamos las salidas:



La operación de convolución

Deslizamos el filtro 3x3 sobre la imagen de entrada, multiplicamos los elementos y sumamos las salidas:



La operación de convolución

Deslizamos el filtro 3x3 sobre la imagen de entrada, multiplicamos los elementos y sumamos las salidas:

1	1	1	1×1	0	0×1
0	1	1	1×0	1	1×1
0	0	1	1×1	1	1×0
0	0	1	1	1×0	1×1
0	1	1	0	0	



1	0	1
0	1	0
1	0	1

filtro



4	3	4

mapa de
características

La operación de convolución

Deslizamos el filtro 3x3 sobre la imagen de entrada, multiplicamos los elementos y sumamos las salidas:

1	1	1	0	0
0 $\times 1$	1 $\times 0$	1 $\times 1$	1	0
0 $\times 0$	0 $\times 1$	1 $\times 0$	1	1
0 $\times 1$	0 $\times 0$	1 $\times 1$	1	0
0	1	1	0	0



1	0	1
0	1	0
1	0	1

filtro



4	3	4
2		

mapa de
características

La operación de convolución

Deslizamos el filtro 3x3 sobre la imagen de entrada, multiplicamos los elementos y sumamos las salidas:

1	1	1	0	0
0	1 _{x1}	1 _{x0}	1 _{x1}	0
0	0 _{x0}	1 _{x1}	1 _{x0}	1
0	0 _{x1}	1 _{x0}	1 _{x1}	0
0	1	1	0	0



1	0	1
0	1	0
1	0	1

filtro



4	3	4
2	4	

mapa de
características

La operación de convolución

Deslizamos el filtro 3x3 sobre la imagen de entrada, multiplicamos los elementos y sumamos las salidas:

1	1	1	0	0
0	1	1 _{*1}	1 _{*0}	0 _{*1}
0	0	1 _{*0}	1 _{*1}	1 _{*0}
0	0	1 _{*1}	1 _{*0}	0 _{*1}
0	1	1	0	0



1	0	1
0	1	0
1	0	1

filtro



4	3	4
2	4	3

mapa de
características

La operación de convolución

Deslizamos el filtro 3x3 sobre la imagen de entrada, multiplicamos los elementos y sumamos las salidas:

1	1	1	0	0
0	1	1	1	0
0	0	1	1	1
0	0	1	1	0
0	1	1	0	0



1	0	1
0	1	0
1	0	1

filtro



4	3	4
2	4	3
2		

mapa de
características

La operación de convolución

Deslizamos el filtro 3x3 sobre la imagen de entrada, multiplicamos los elementos y sumamos las salidas:

1	1	1	0	0
0	1	1	1	0
0	0	1	1	1
0	0	1	1	0
0	1	1	0	0



1	0	1
0	1	0
1	0	1

filtro



4	3	4
2	4	3
2	3	

mapa de
características

La operación de convolución

Deslizamos el filtro 3x3 sobre la imagen de entrada, multiplicamos los elementos y sumamos las salidas:

1	1	1	0	0
0	1	1	1	0
0	0	1 _{x1}	1 _{x0}	1 _{x1}
0	0	1 _{x0}	1 _{x1}	0 _{x0}
0	1	1 _{x1}	0 _{x0}	0 _{x1}



1	0	1
0	1	0
1	0	1

filtro



4	3	4
2	4	3
2	3	4

mapa de
características

Produciendo mapas de características



(j) Original



(k) Detallada

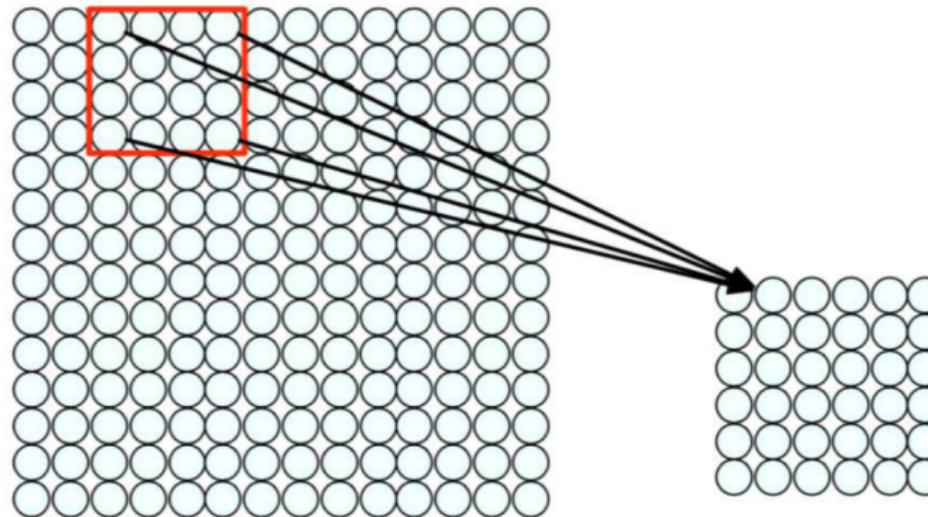


(l) Detección de
bordes



(m) Detección de
bordes "fuerte"

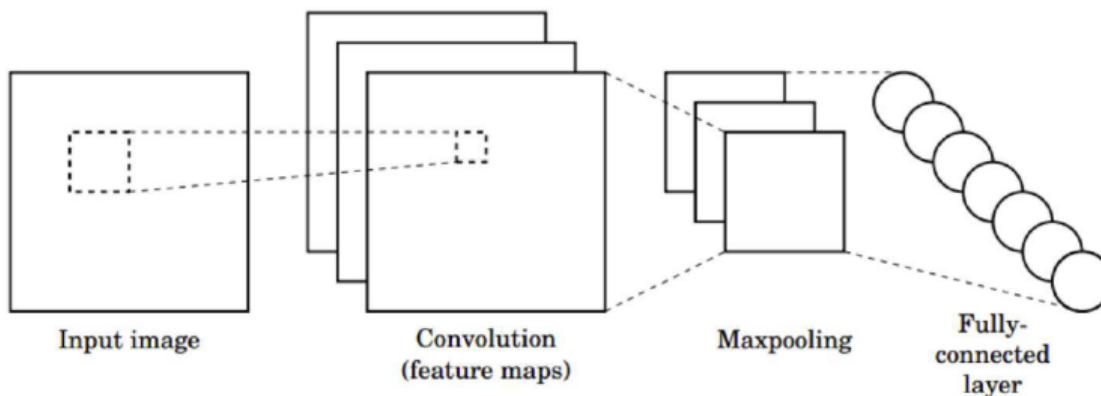
Aplicación de filtros para extraer características



- 1 Aplicar un conjunto de pesos - un filtro - para extraer las **características locales**
- 2 Usar **múltiples filtros** para extraer diferentes características
- 3 Compartir espacialmente los parámetros de cada filtro

Convolutional Neural Networks (CNNs)

CNNs para la clasificación

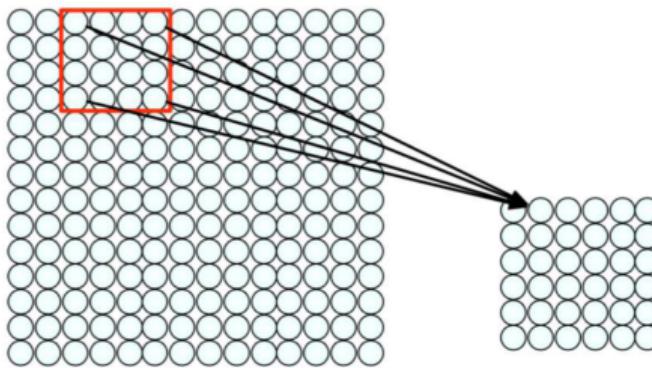


- 1 **Convolución:** Aplicar filtros con pesos aprendidos para generar mapas de características.
- 2 **No linealidad:** A menudo ReLU.
- 3 **Pooling:** (agrupación) operación de **submuestreo** (*downsampling*) en cada mapa de características.

Modelo de entrenamiento con datos de imagen.

Aprende los pesos de los filtros en las capas convolucionales.

Capas convolucionales: Conectividad local



Para una neurona en una capa oculta:

- Toma las entradas del parche
 - Calcular la suma ponderada
 - Aplicar el sesgo
- 1 aplicando una ventana de pesos
 - 2 computar las combinaciones lineales
 - 3 activando con la función no lineal

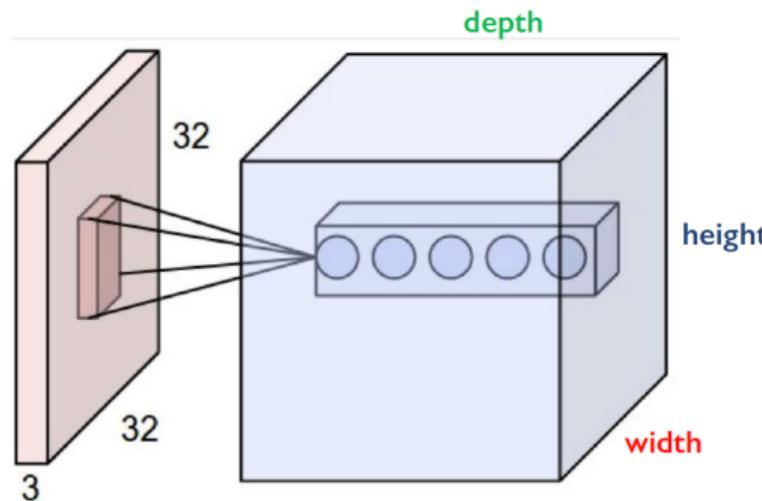
Filtro 4x4: matriz de pesos

w_{ij}

$$\sum_{i=1}^4 \sum_{j=1}^4 w_{ij} x_{i+p, j+q} + b$$

para la neurona (p, q) en la capa oculta

CNN: Disposición espacial del volumen de salida



Dimensiones de la capa:

$$h \times w \times d$$

- h y w son dimensiones espaciales
- d (*profundidad*) = número de filtros

Avance:

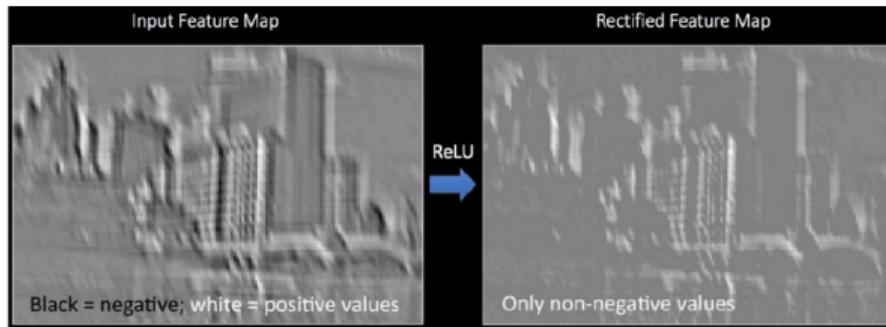
Tamaño del paso del filtro

Campo Receptivo:

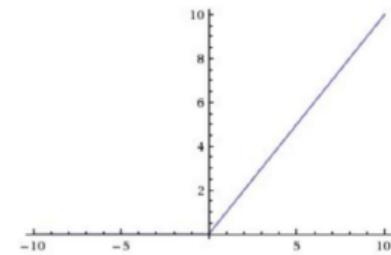
Ubicaciones en la imagen de entrada a la que un nodo está conectado.

Introduciendo la no linealidad

- Aplicar después de cada operación de convolución (es decir, después de las capas convolutivas)
- **ReLU:** operación píxel a píxel que reemplaza todos los valores negativos por cero. ¡Operación no lineal!

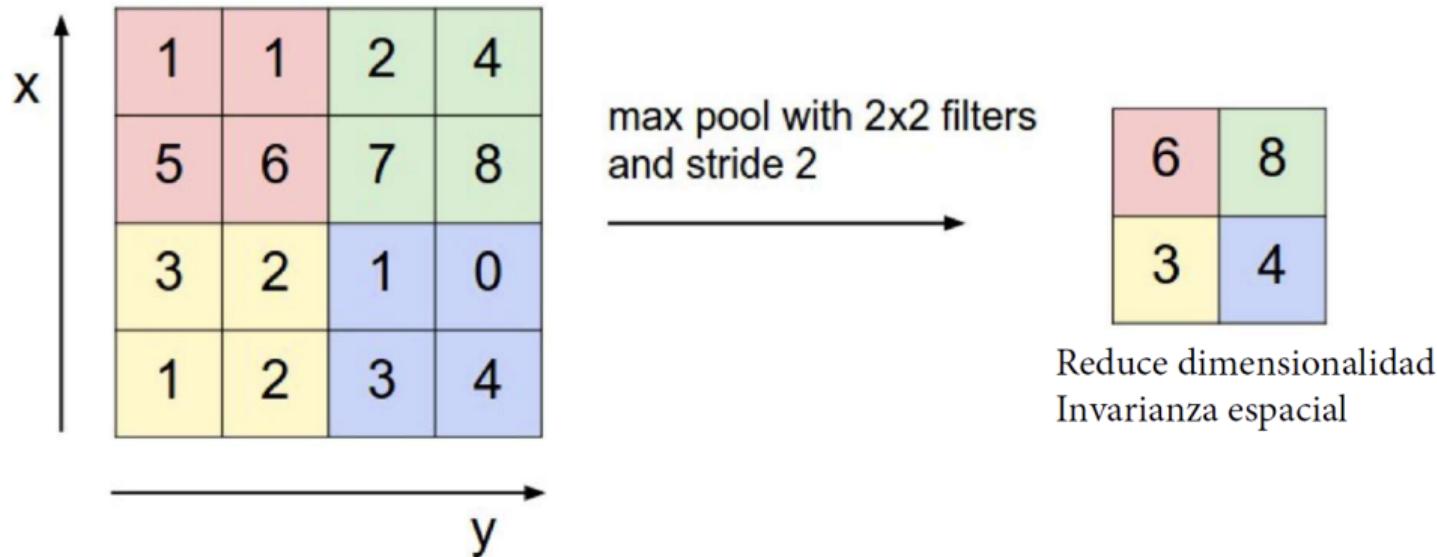


Unidad lineal rectificada (ReLU)



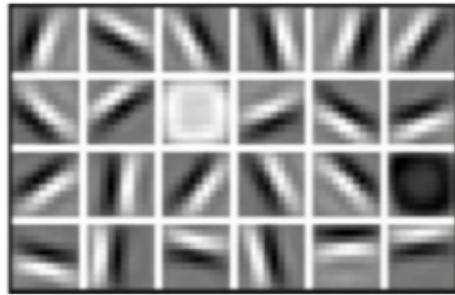
$$g(z) = \max(0, z)$$

Pooling



¿De qué otra forma podemos reducir la muestra y preservar la invariabilidad espacial?

Representaciones del aprendizaje con CNNs profundas



(n) Características de bajo nivel: Líneas y bordes. Capa conv. 1

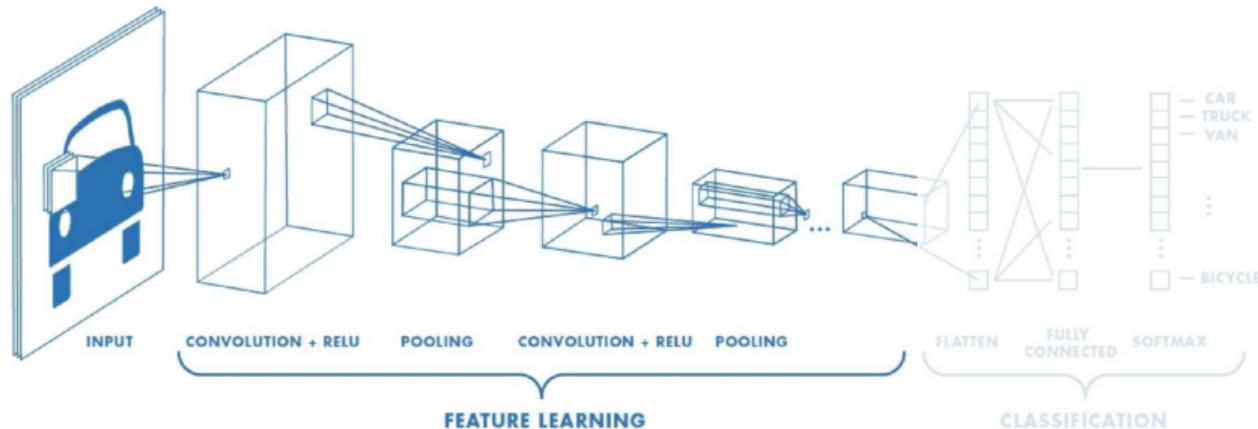


(o) Características de nivel medio: Ojos, nariz y oídos. Capa conv. 2



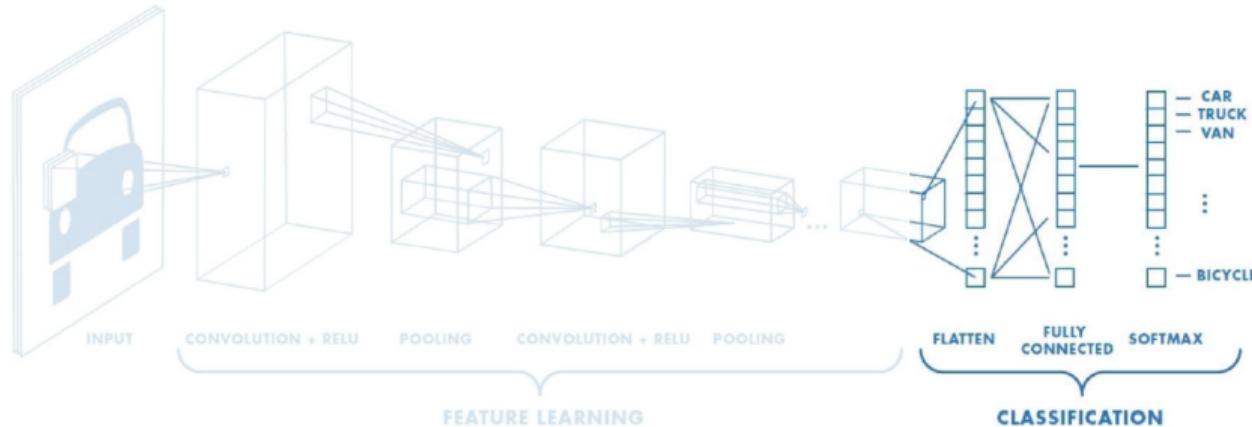
(p) Características de alto nivel: Estructura facial. Capa conv. 3

CNN para la clasificación: Aprendizaje de características



- 1 Aprende las características de la imagen de entrada a través de la **convolución**
- 2 Introducir la **no-linealidad** a través de la función de activación (¡los datos del mundo real son no-lineales!)
- 3 Reducir la dimensionalidad y preservar la invariabilidad espacial con el **pooling**

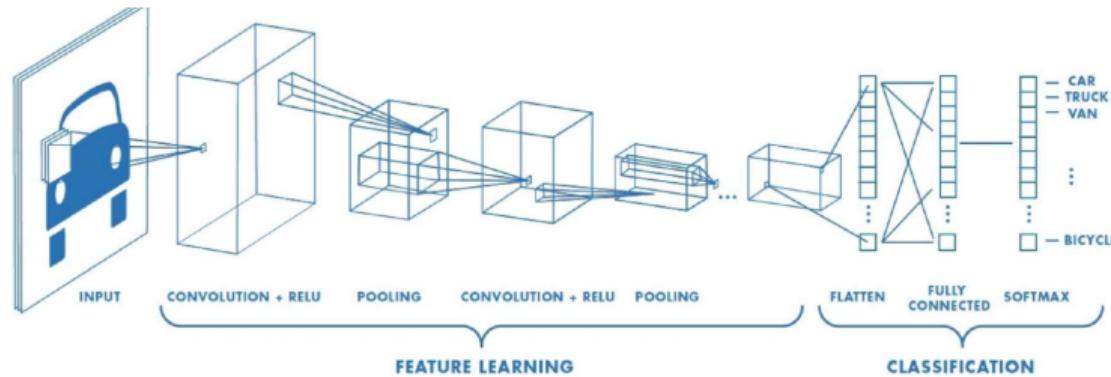
CNN para la clasificación: Probabilidades de la clase



- Las capas **CONV** y **POOL** producen características de alto nivel de entrada
- La capa totalmente conectada (*fully connected layer*) utiliza estas características para clasificar la imagen de entrada
- Expresar la salida como la **probabilidad** de que la imagen pertenezca a una clase particular

$$\text{softmax}(y_i) = \frac{e^{y_i}}{\sum_j e^{y_j}}$$

CNN: Entrenamiento con Backpropagation



Aprende los pesos para los filtros convolucionales y las capas completamente conectadas

La retropropagación (Backpropagation): pérdida de entropía cruzada

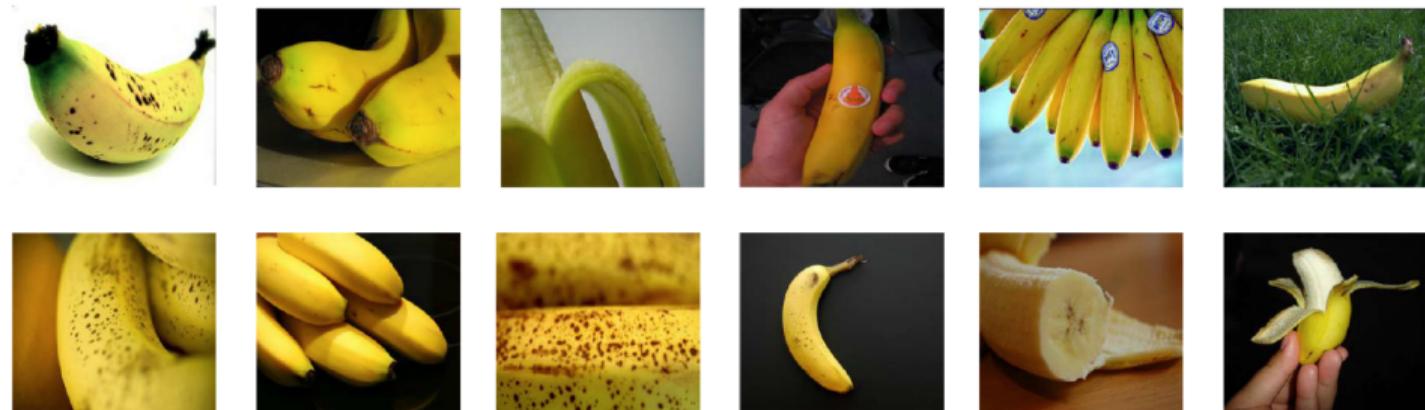
$$J(\Theta) = \sum_i y^{(i)} \log(\hat{y}^{(i)})$$

CNN para la clasificación: ImageNet

Conjunto de datos de ImageNet

Conjunto de datos de más de **14 millones** de imágenes en **21.841** categorías

Prompt: "Fruta amarilla en forma de media luna alargada con una suave y dulce pulpa"



1409 fotos de plátanos.

El desafío ImageNet

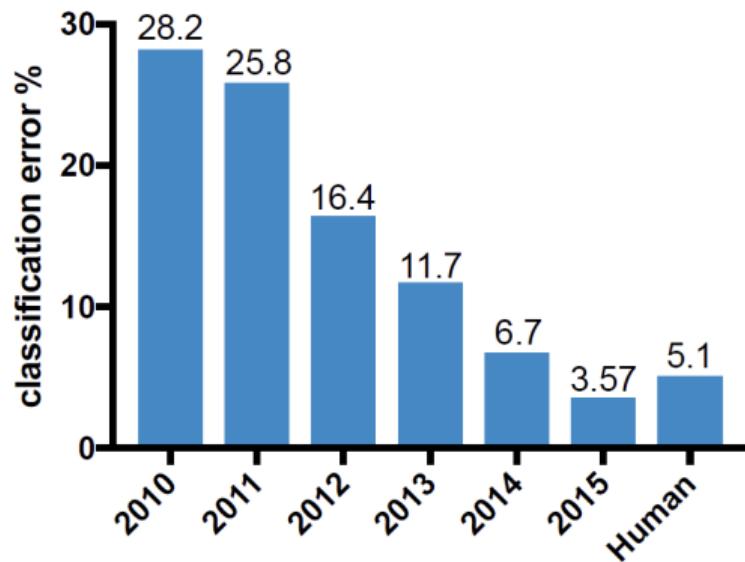


Problema de clasificación: producir una lista de categorías de objetos presentes en la imagen. 1000 categorías.

- "Error de los 5 principales": tasa a la que el modelo no da la etiqueta correcta en las 5 principales predicciones

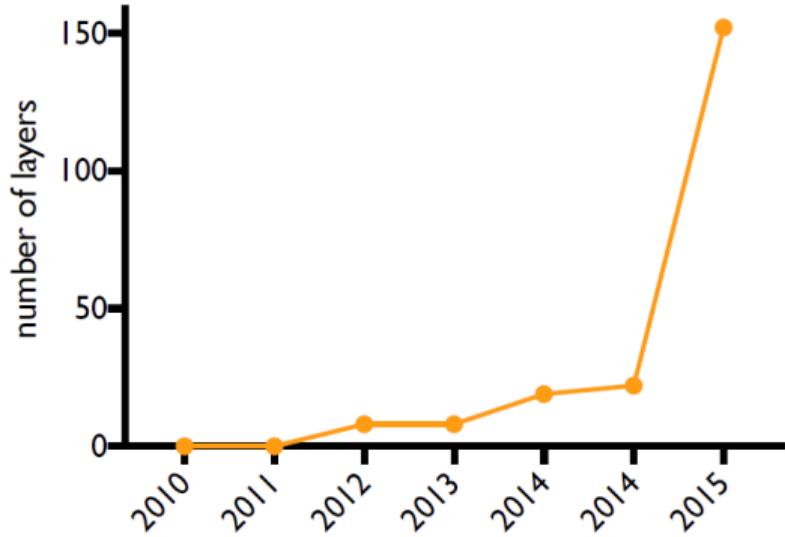
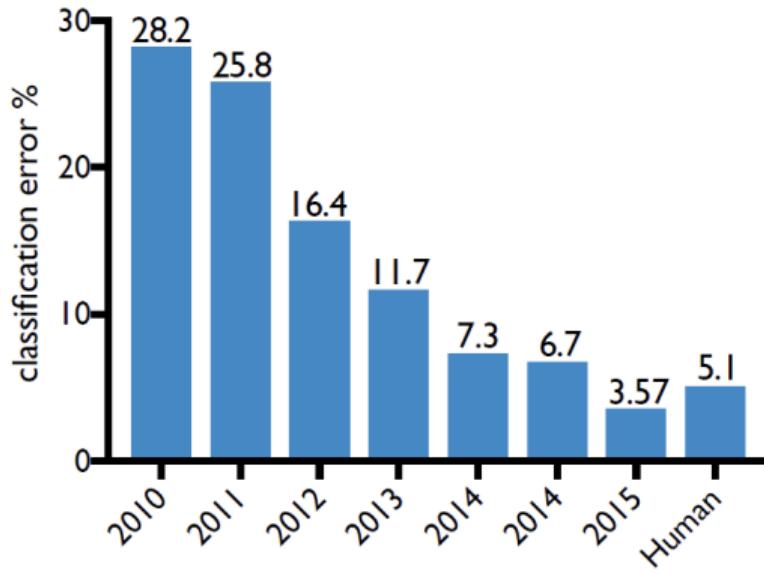
Otras tareas incluyen: Localización de un solo objeto, detección de objetos de vídeo/ímagenes, clasificación de escenas, análisis sintáctico de escenas

El desafío ImageNet: problema de clasificación



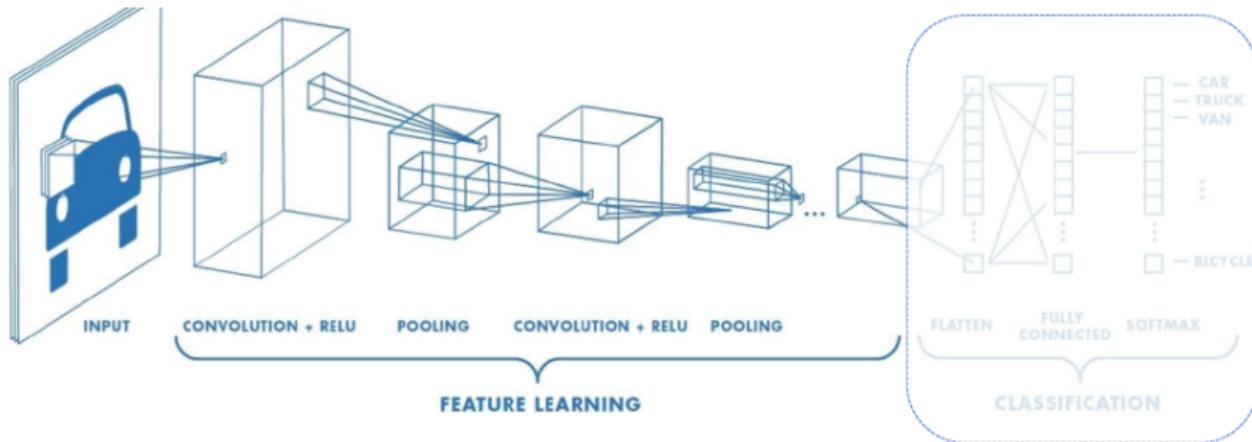
- 2012: **AlexNet**. La primera CNN en ganar.
 - 8 capas, 61 millones de parámetros
- 2013: **ZFNet**
 - 8 capas, más filtros
- 2014: **VGG**
 - 19 capas
- 2014: **GoogLeNet**
 - Módulos "*Inception*"
 - 22 capas, 5 millones de parámetros
- 2015: **ResNet**
 - 152 capas

El desafío ImageNet: problema de clasificación



**Una arquitectura para muchas
aplicaciones**

Una arquitectura para muchas aplicaciones



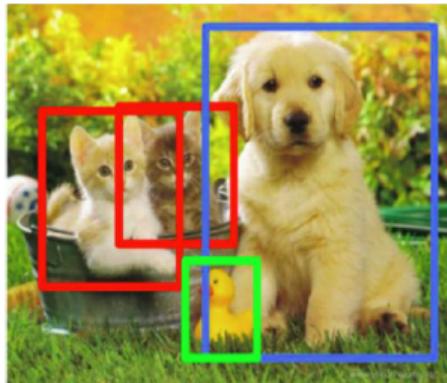
- Detección de objetos con R-CNNs
- Segmentación con redes totalmente convolutivas
- Subtitulado de imágenes con RNNs

Más allá de la clasificación



CAT

(q) Segmentación semántica



CAT, DOG, DUCK

(r) Detección de objetos

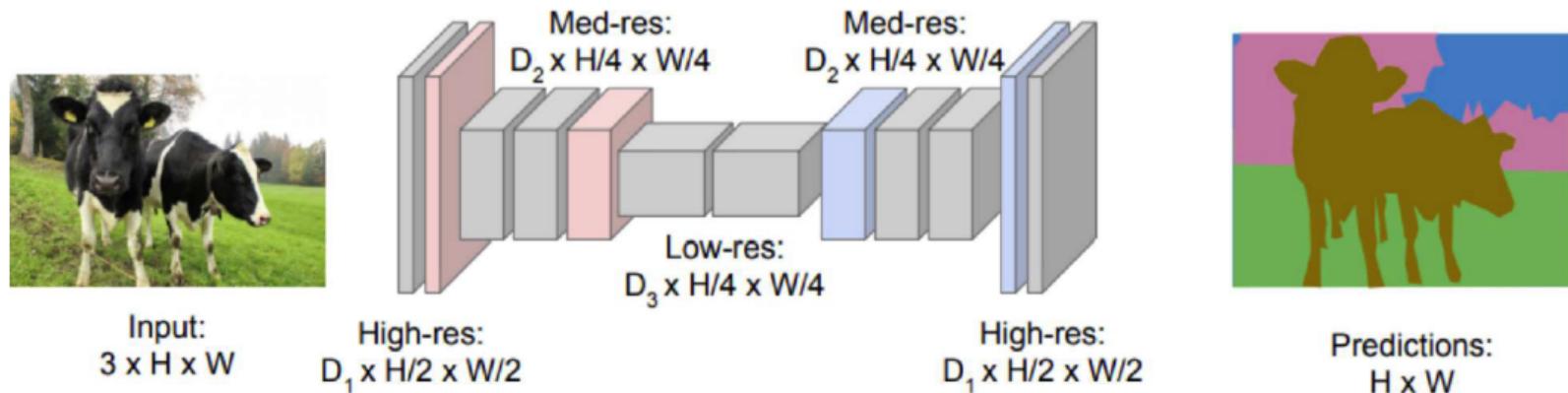


El gato está en el pasto.

(s) Subtitulado de la imagen

Segmentación semántica: FCNs

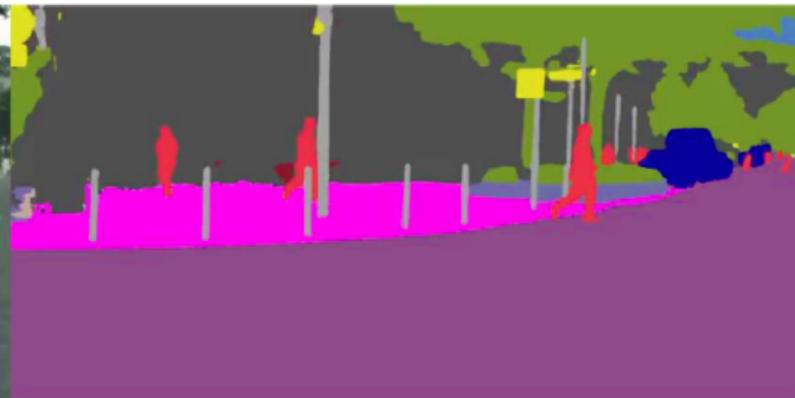
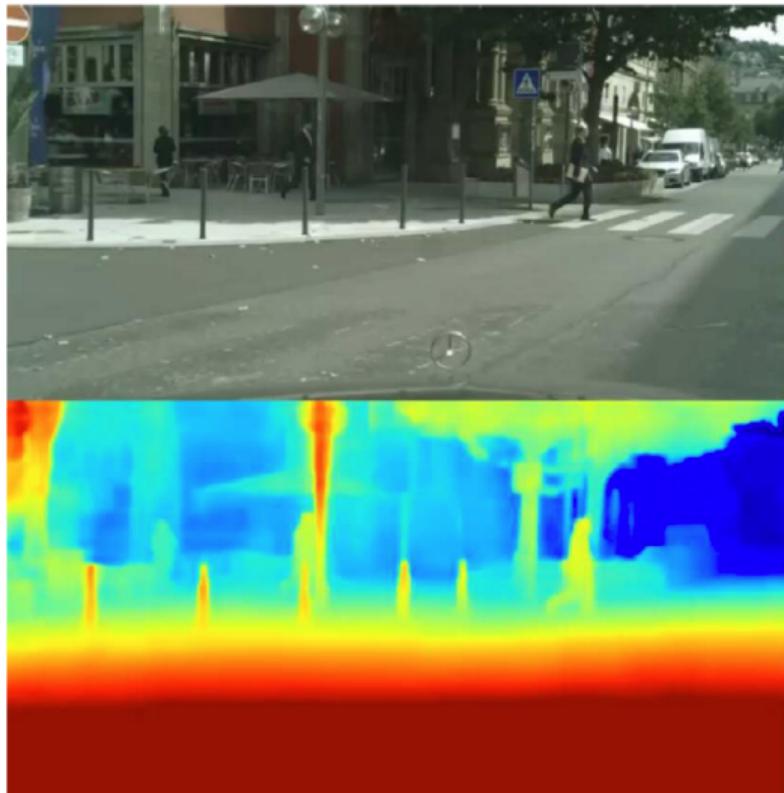
- FCN (*Fully Convolutional Network*): Red totalmente convolutiva.
- Red diseñada con todas las capas convolucionales, con operaciones de **disminución** y **aumento** de la muestra.



Segmentación de una escena de conducción

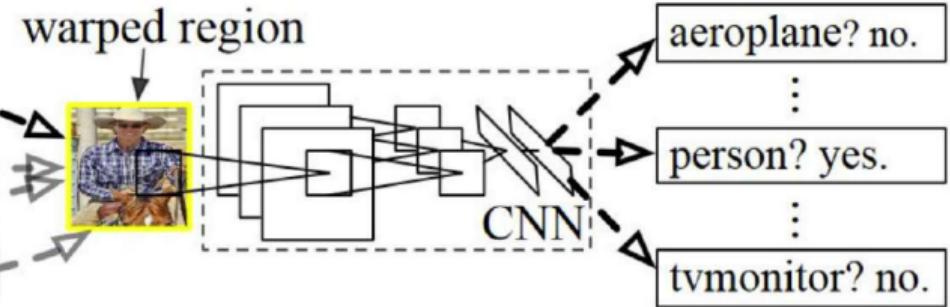
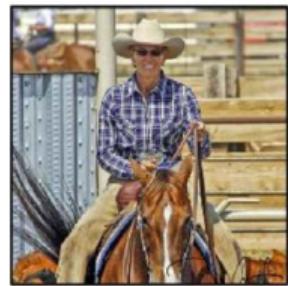


Segmentación de una escena de conducción



Detección de objetos con R-CNN

R-CNN: Encontrar regiones que creemos que tienen objetos. Usa la CNN para clasificar



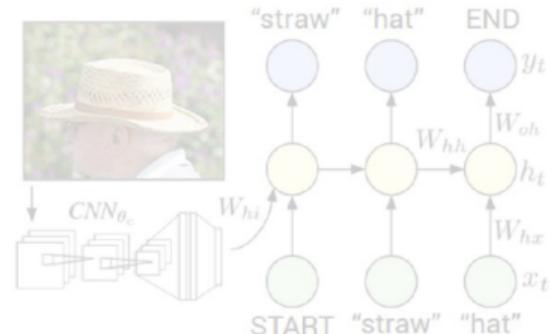
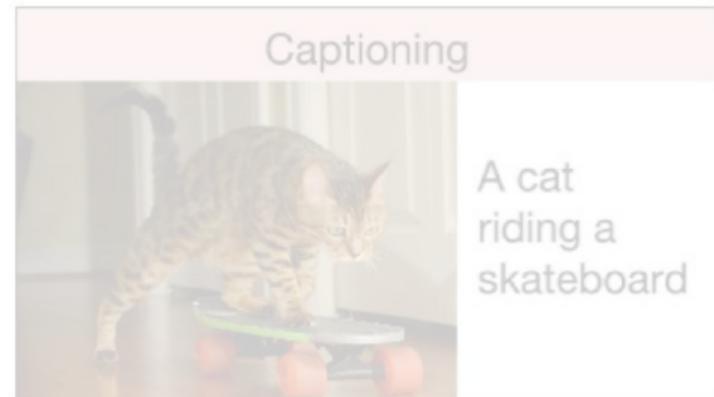
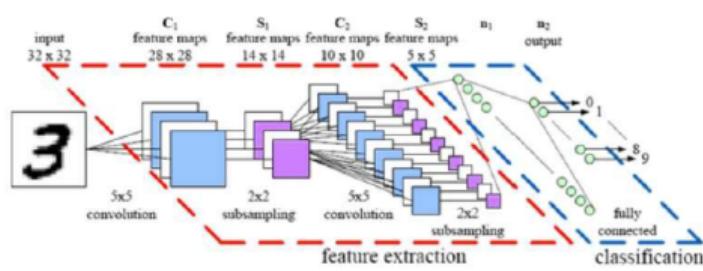
1. Input image

2. Extract region proposals (~2k)

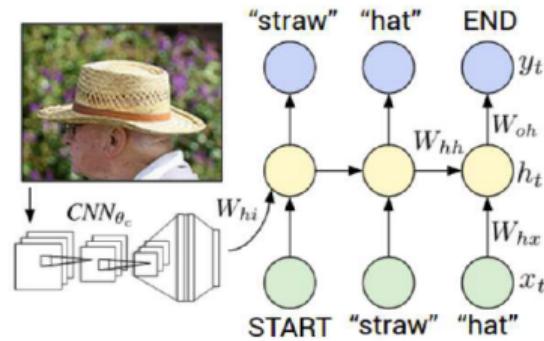
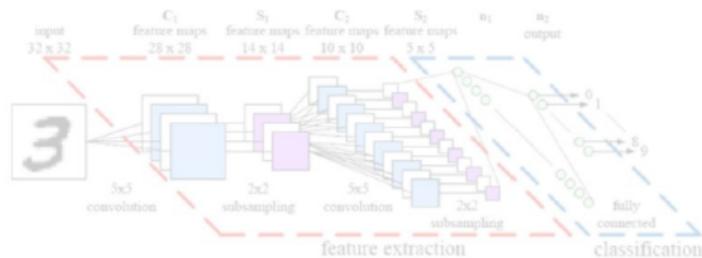
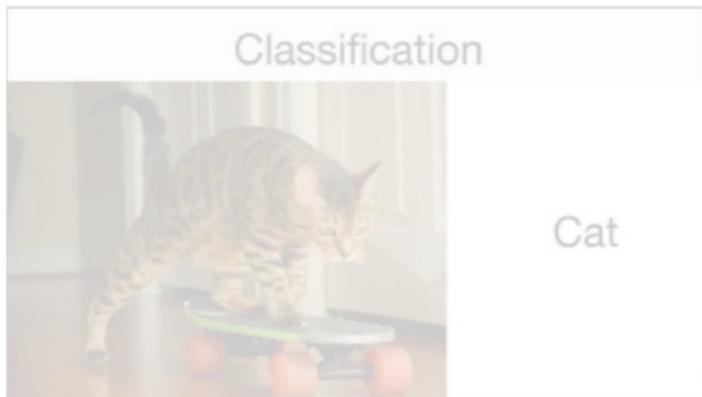
3. Compute CNN features

4. Classify regions

Subtitulado de imágenes usando RNNs



Subtitulado de imágenes usando RNNs



Aprendizaje profundo para la visión por ordenador: Impacto y resumen

Datos, Datos, Datos



ImageNet: categorías 22K. 14M imágenes.

Avión
Automóvil
Pájaro
Gato
Venado
Perro
Rana
Caballo
Nave
Camión

CIFAR-10

3	4	2	1	9	5	6	2	1	8
8	9	1	2	5	0	0	6	4	4
6	7	0	1	6	3	6	3	7	0
3	7	7	9	4	6	6	1	8	2
2	9	3	4	3	9	8	7	2	5
1	5	9	8	3	6	5	7	2	3
9	3	1	9	1	5	8	0	8	4
5	6	2	6	8	5	8	8	9	9
3	7	7	0	9	4	8	5	4	3
7	2	6	4	1	0	6	9	2	3

MNIST: dígitos escritos a mano

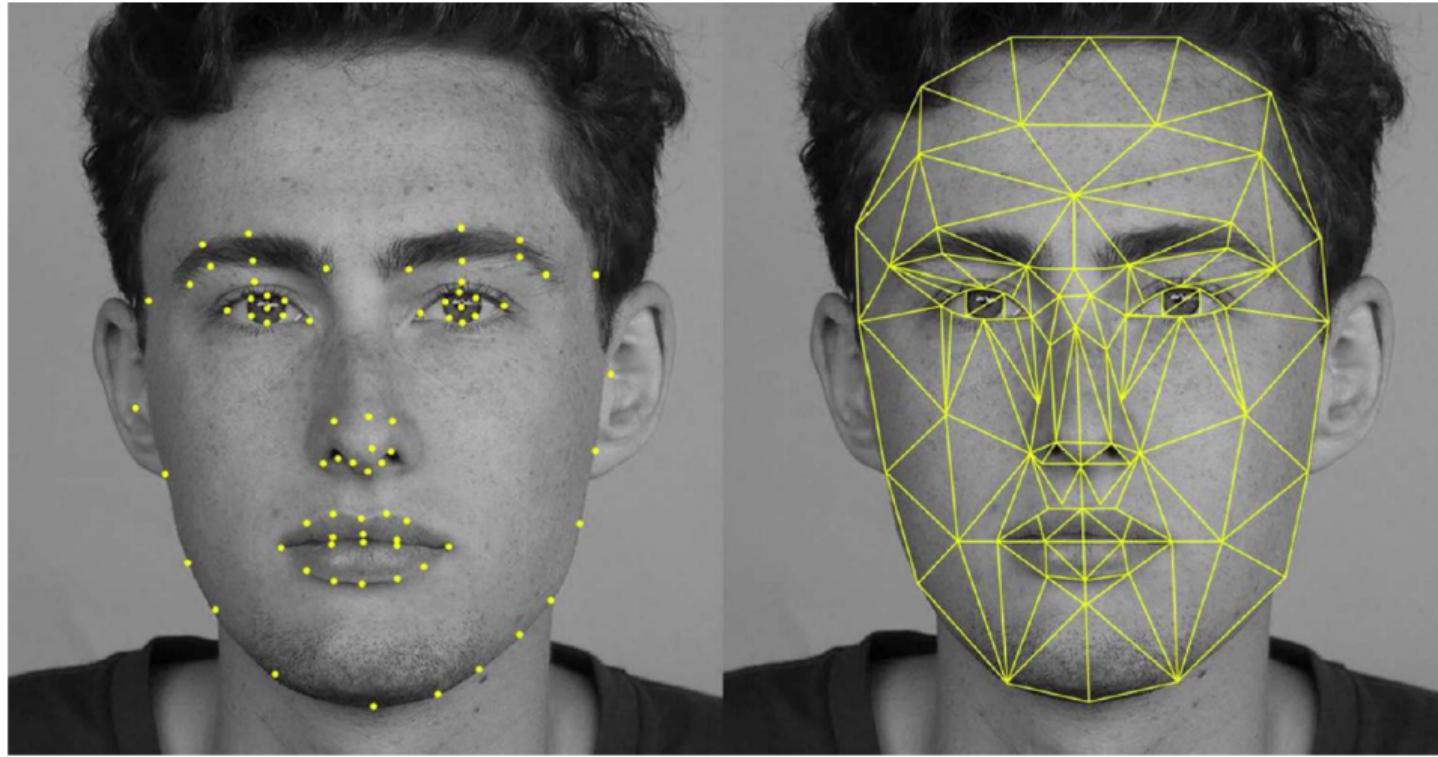


places: escenas naturales

Aprendizaje profundo para la visión por computador: Impacto



Impacto: Detección de rostros

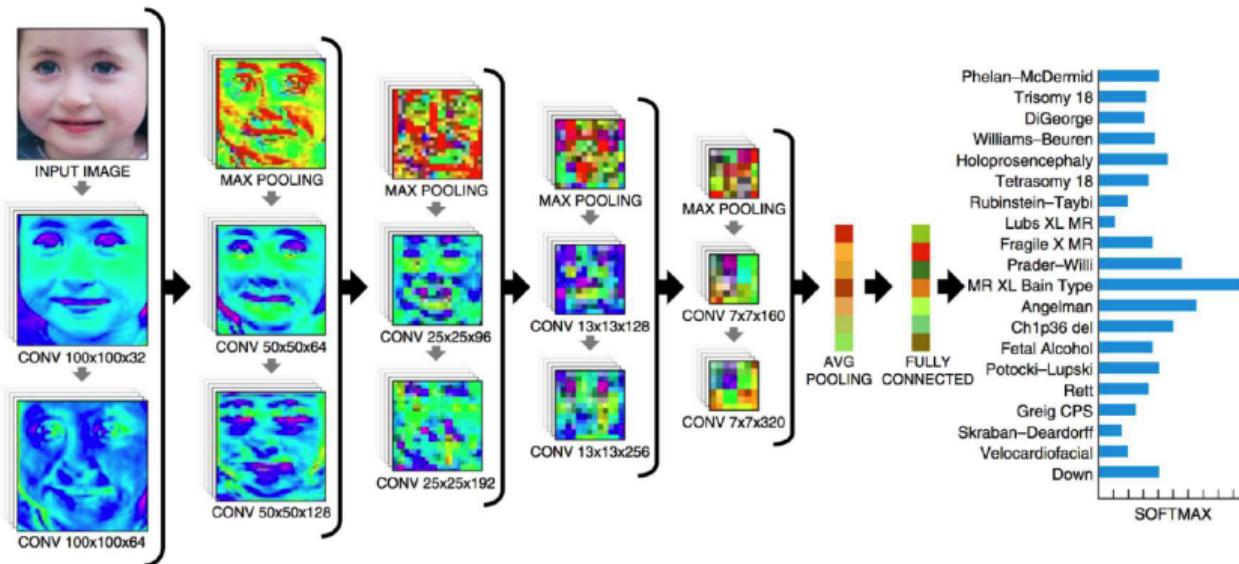


Impacto: carros autónomos



Impacto: Medicina

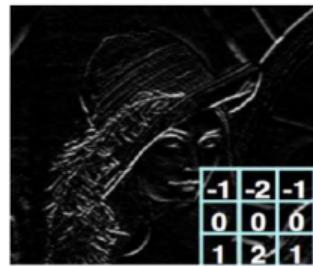
Identificar los fenotipos faciales de los trastornos genéticos utilizando el aprendizaje profundo
Gurovich y otros, Nature Med. 2019



Aprendizaje profundo para visión por computador: review

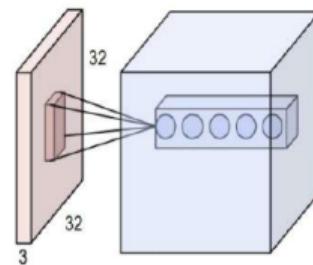
Fundamentos

- ¿Por qué la visión por ordenador?
- Representar imágenes
- Convoluciones para la extracción de características



CNNs

- La arquitectura de la CNN
- Aplicación a la clasificación
- ImageNet



Entrenamiento en práctica

- Aplicaciones
- Segmentación, detección de objetos, subtítulo de imágenes
- Visualización



¡Muchas gracias por su atención!

¿Preguntas?



Contacto: Marco Teran
webpage: marcoteran.github.io/
e-mail: marco.teran@usa.edu.co

