



Visión por Computadora II - CEAI - FIUBA



Profesores:

- Alfonso Rafel - alfonsorafel93@gmail.com
- Cornet Juan Ignacio - juanignaciocornet@gmail.com
- Seyed Pakdaman - khodadad.pakdaman@gmail.com

Cuarta clase:

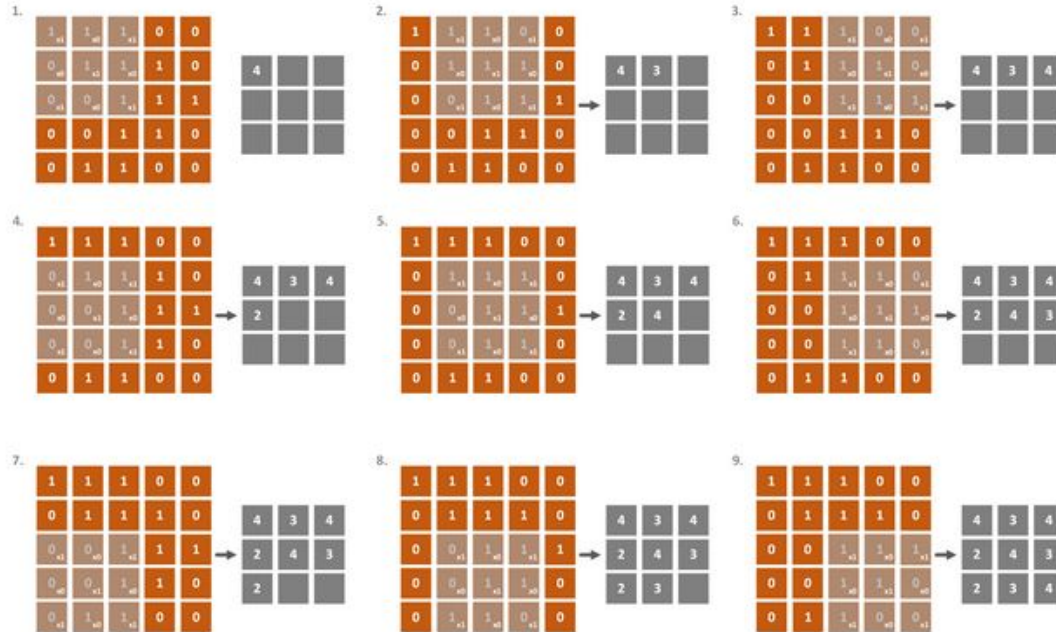


- Revisión temas TF
- Segmentación
 - Aplicaciones
 - Medidas de error
 - Acercamiento al problema de segmentación
 - Arquitecturas clásicas
 - U-Net
 - DeepLab
 - Mask R-CNN
 - Dataset clásicos
 - Explicando las redes convolucionales

Trabajo final



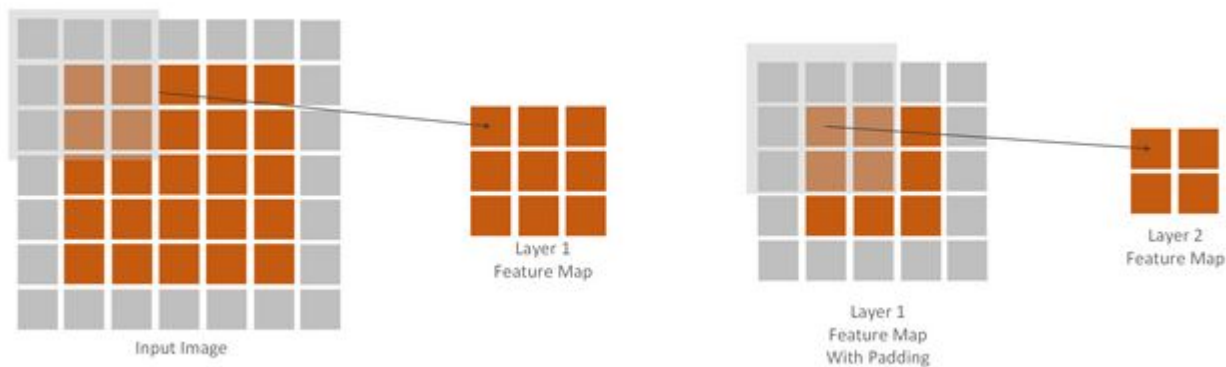
Redes convolucionales



Campo receptivo:

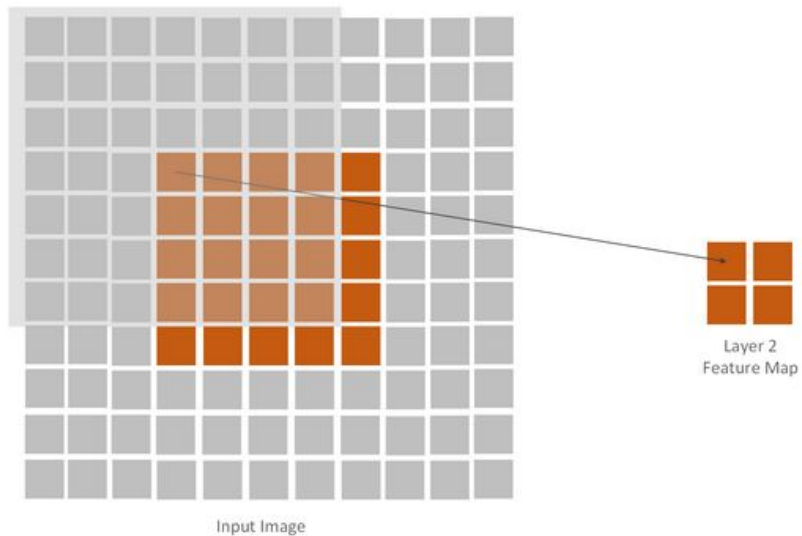
- tamaño
- centro

Redes convolucionales

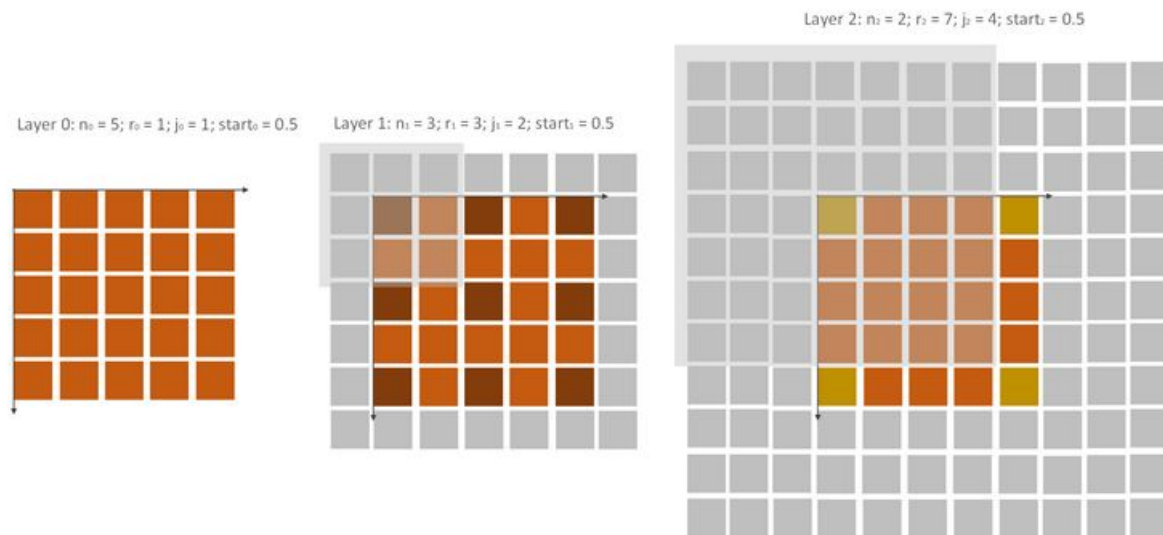


Campo receptivo se va complejizando y en principio expandiendo a medida que nos sumergimos más profundo en la arquitectura de la red.

Redes convolucionales



Redes convolucionales



$$n_i = \frac{n_{i-1} + 2p - k}{s} + 1$$

$$j_i = j_{i-1} * s$$

$$r_i = r_{i-1} + (k - 1)j_{i-1}$$

$$start_i = start_{i-1} + \left(\frac{k-1}{2} - p\right)j_{i-1}$$

Aplicaciones de image segmentation

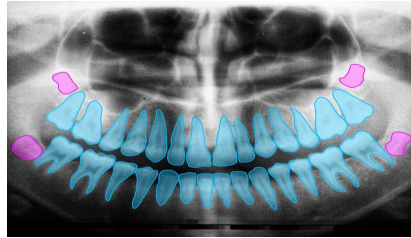


- Fotografía computacional
- Medición de área con satélites
- Identificación de tejidos (e.g. tumores vs. tejido sano)
- Cirugía guiada por computadora
- Realidad aumentada
- Encontrar límites geográficos con satélites
- Apps de celular para pintar regiones o aplicar filtros selectivamente
- Segregación de residuos

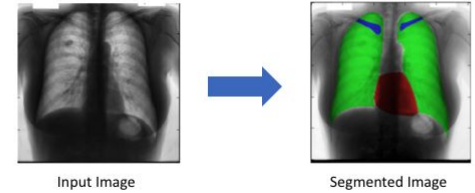
Ejemplos

- Segmentación semántica de campos: <https://www.youtube.com/watch?v=wfObVKKKJkE&t=2s>
- Reemplazo del cielo en realidad aumentada: <https://www.youtube.com/watch?v=B6X0q7YGUDQ>
- Segmentación de imágenes urbanas en tiempo real: <https://www.youtube.com/watch?v=qWI9idsCuLQ>

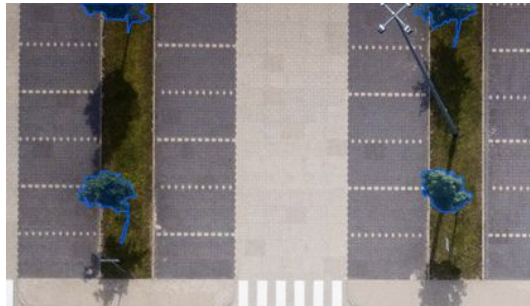
- Imágenes dentales:



- Imágenes médicas:



- Agricultura de precisión:



- Segmentación de residuos:



Medidas de error

- Pixel Accuracy: porcentaje de píxeles en la imagen clasificados correctamente.
Problemas: si hay mucho fondo el resultado de la métrica puede no ser significativo (en general un problema si las clases están muy desbalanceadas)

$$PA = \frac{\sum_{i=0}^K p_{ii}}{\sum_{i=0}^K \sum_{j=0}^K p_{ij}},$$

- Mean Pixel Accuracy: para salvar el problema anterior, una alternativa muy utilizada es calcular el Pixel Accuracy por clase, y luego promediarlos.

$$MPA = \frac{1}{K + 1} \sum_{i=0}^K \frac{p_{ii}}{\sum_{j=0}^K p_{ij}}.$$

Medidas de error



- IoU promedio: se toma la IoU para cada clase en la imagen y se saca el promedio

$$\text{IoU} = J(A, B) = \frac{|A \cap B|}{|A \cup B|},$$

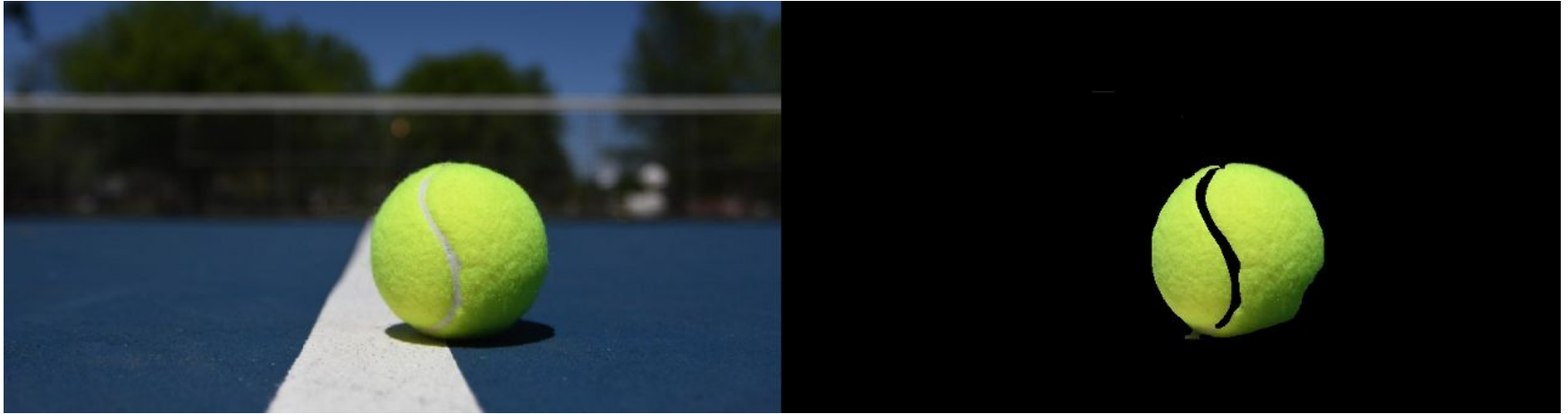
- Coeficiente Dice o F1 score: 2 veces el área de la intersección dividida la suma de las áreas

$$\text{Dice} = \frac{2|A \cap B|}{|A| + |B|}.$$

Otras medidas de error:

Shruti Jadon, A survey of loss functions for semantic segmentation [Link](#)

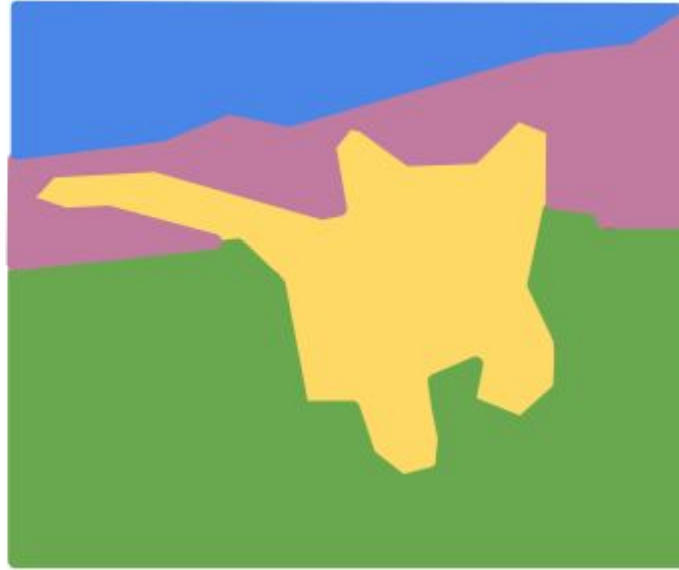
Segmentación Pixel a Pixel



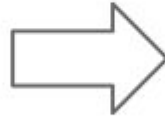
Segmentación Pixel a Pixel



Segmentación semántica



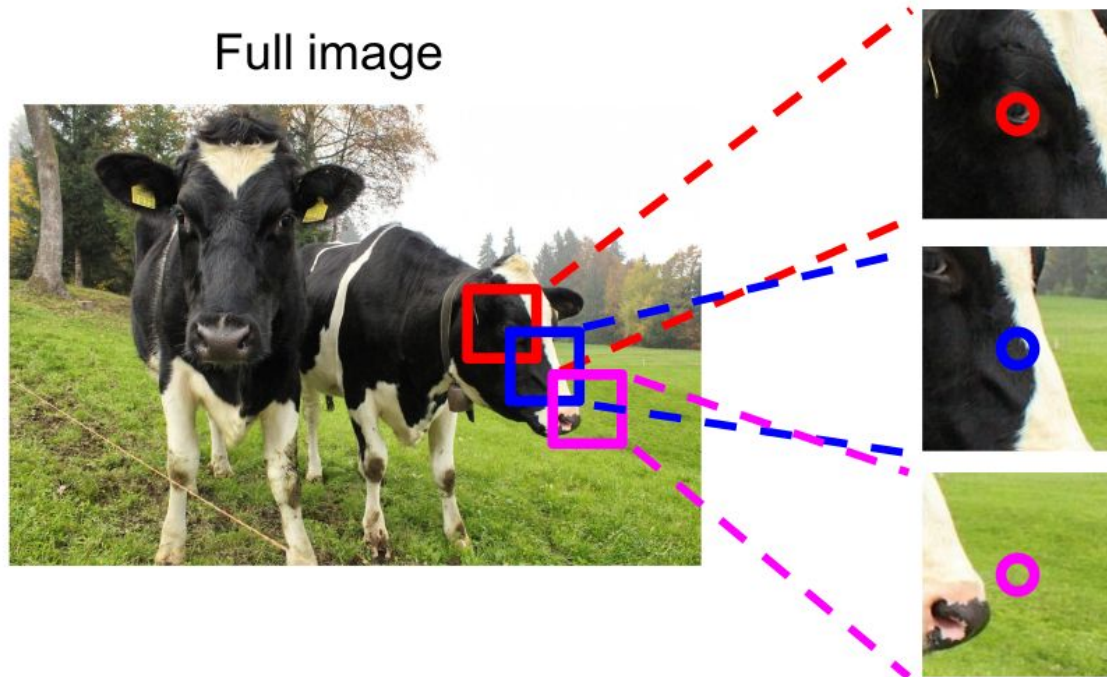
Segmentación semántica



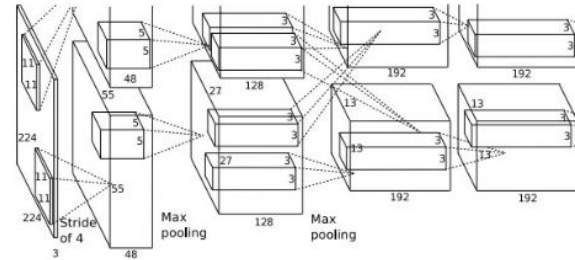
Enfoque con ventana deslizable



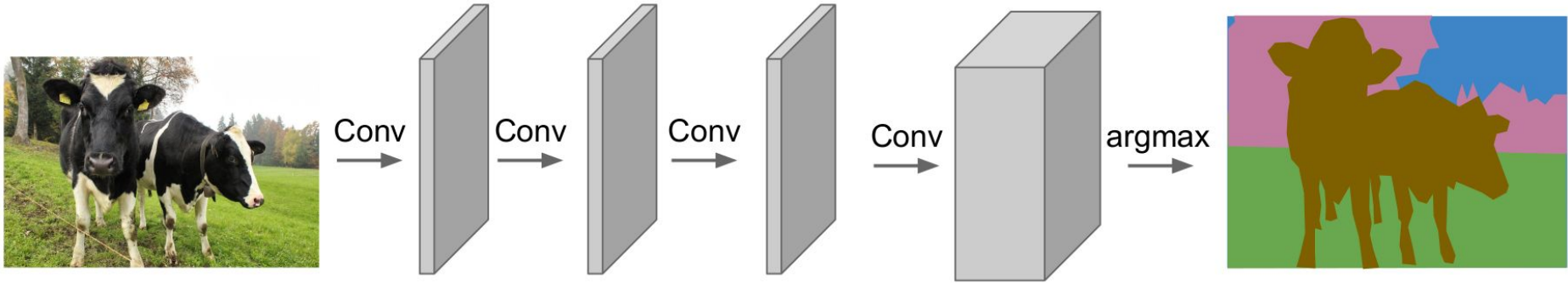
Enfoque con ventana deslizante



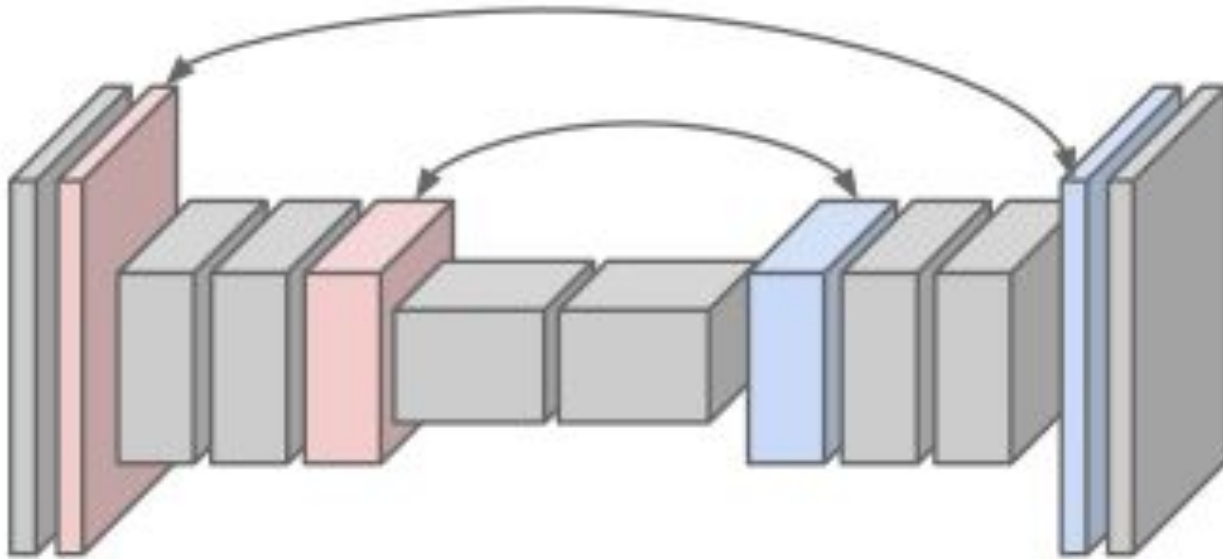
Enfoque con ventana deslizante



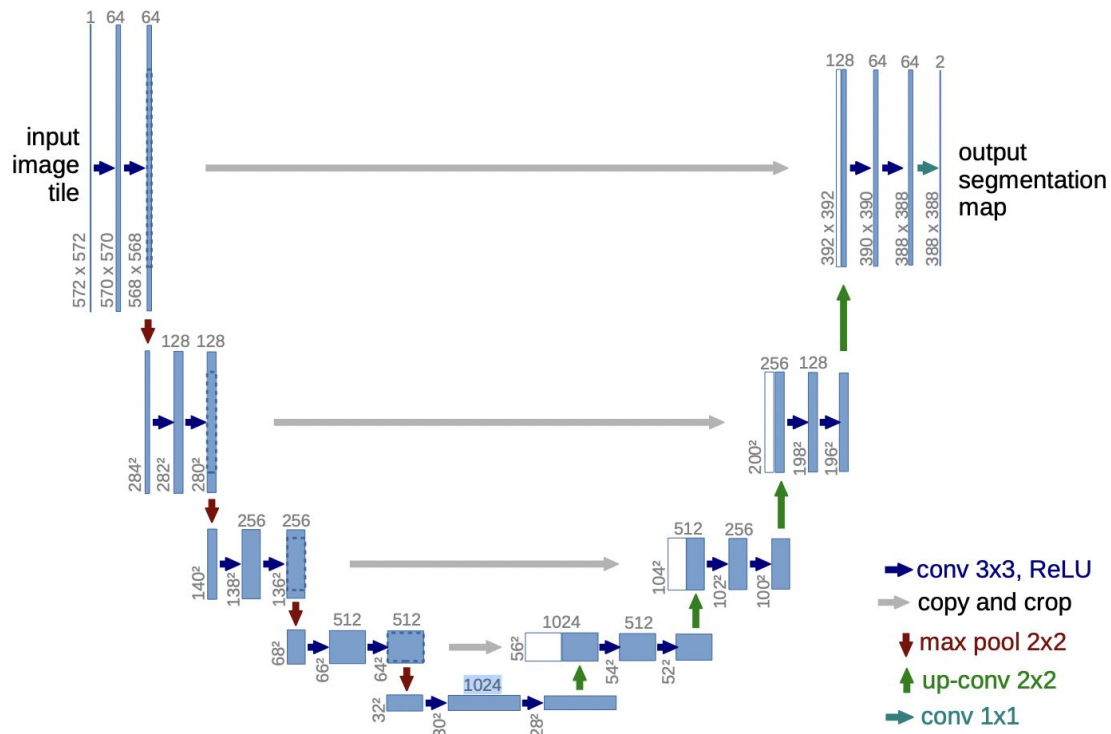
Fully Convolutional Network



Enfoque Codificador Decodificador



U-Net: Arquitectura



U-Net



- Es una red fully convolutional, esto quiere decir que convierte capas fully connected en convolucionales. De esta manera permite aplicar la red a imágenes de diverso tamaño.
- La arquitectura tiene forma de U y tiene un enfoque de codificador, decodificador. Hay una parte inicial que contrae la imagen y otra parte posterior que la expande nuevamente. Para ello se usa upsampling a través de transposed convolution (“convoluciones transpuestas”)
- Se propaga la información desde capas de la parte contractiva a capas de la parte expansiva de la misma “granularidad con “skip connections”
- Data augmentation: para el entrenamiento se realizan traslaciones, rotaciones, modificaciones de la escala de grises y **deformaciones elásticas aleatorias**
- Se usó para segmentación de estructuras neuronales y de células

Función de costo:



$$E = \sum_{\mathbf{x} \in \Omega} w(\mathbf{x}) \log(p_{\ell(\mathbf{x})}(\mathbf{x}))$$

- La función de costo se calcula sumando la contribución de cada pixel:
- **p_l(x)** es una softmax que usa las activaciones en cada canal para estimar una probabilidad que le asigna la red a que ese píxel esté en la clase **k**, y **l** es el índice de la verdadera clase del píxel
- La función **w(x)** tiene dos propósitos.
 - **w_c** compensa el desbalance de clases
 - el segundo término le da más peso a píxeles que están cerca de bordes, **d_1** es la distancia a la celda más cercana y **d_2** a la segunda celda más cercana, da más peso a píxeles cerca de bordes

$$w(\mathbf{x}) = w_c(\mathbf{x}) + w_0 \cdot \exp \left(-\frac{(d_1(\mathbf{x}) + d_2(\mathbf{x}))^2}{2\sigma^2} \right)$$

Código para construcción de UNet



Fuentes: <https://www.kaggle.com/phoenigs/u-net-dropout-augmentation-stratification>,
<https://towardsdatascience.com/unet-line-by-line-explanation-9b191c76baf5>

Unpooling

“Bed of Nails”

1	2
3	4



1	0	2	0
0	0	0	0
3	0	4	0
0	0	0	0

Input: 2 x 2

Output: 4 x 4

Nearest Neighbor

1	2
3	4



1	1	2	2
1	1	2	2
3	3	4	4
3	3	4	4

Input: 2 x 2

Output: 4 x 4

Convolución transpuesta

Input

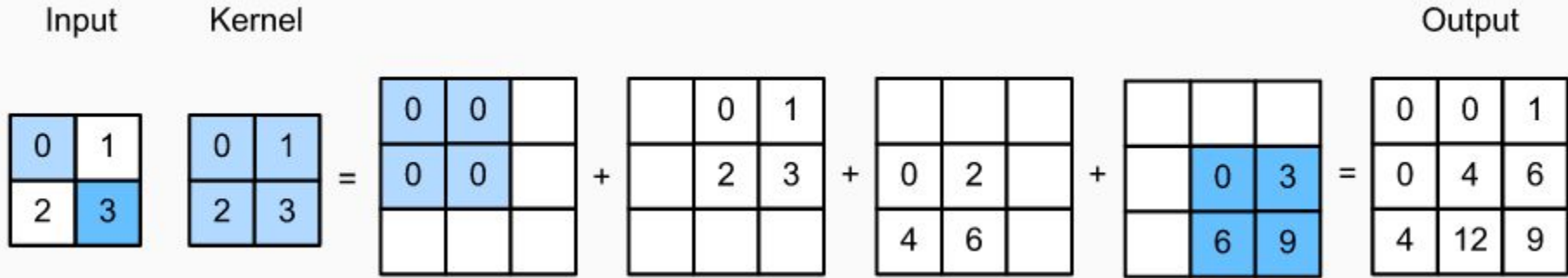
0	1
2	3

Kernel

0	1
2	3

Output

Convolución transpuesta

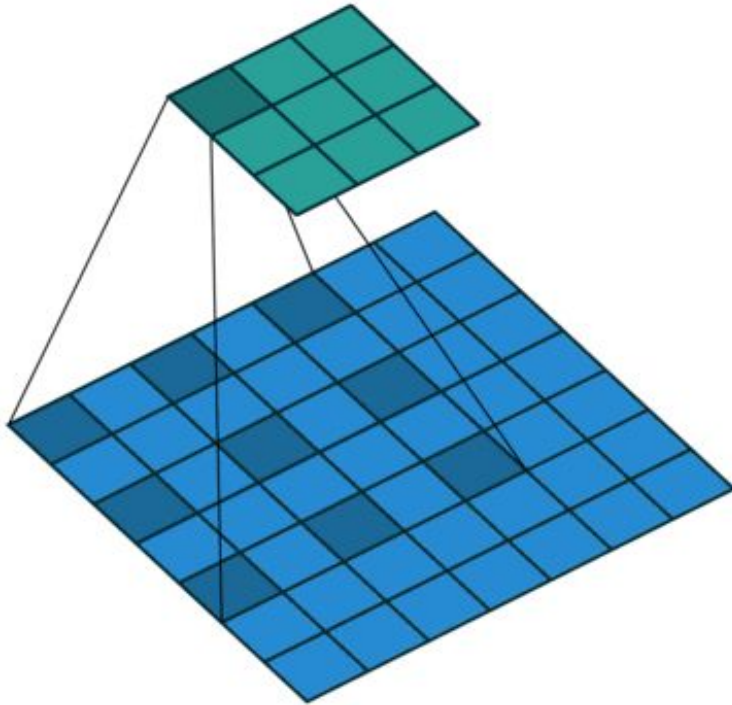


DeepLab



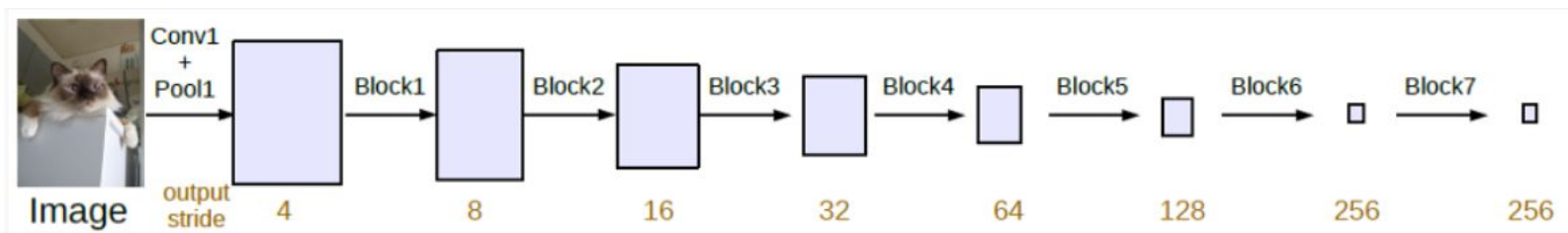
- Modelo moderno de segmentación semántica:
<https://github.com/tensorflow/models/tree/master/research/deeplab>
- De backend usa Resnet-101 y luego Aligned Xception (V3+)
- Introduce varias novedades:
 - Atrous convolution: permite controlar la resolución a la que las diferentes mapas de características son computados en DCNN
 - Atrous Spatial Pyramid Pooling: para segmentar objetos en múltiples escalas con filtros de diferentes sampling rates y field-of-views (campos de visión) efectivos.
 - En versiones posteriores además incluyen mejoras adicionales, como batch normalization, mejoras en la detección de bordes, y en el control del trade-off entre precisión y velocidad de procesamiento
- La versión más nueva es Deeplab V3+
- [Paper de Deeplab](#), [Paper Deeplab V2](#), [Paper Deeplab V3](#), [Paper Deeplab V3+](#)

Atrous convolution

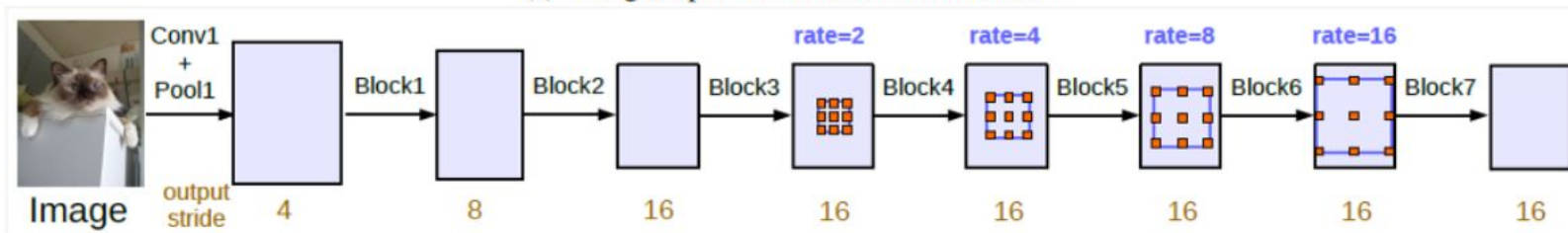


$$y[i] = \sum_k x[i + r \cdot k] w[k]$$

Atrous convolution

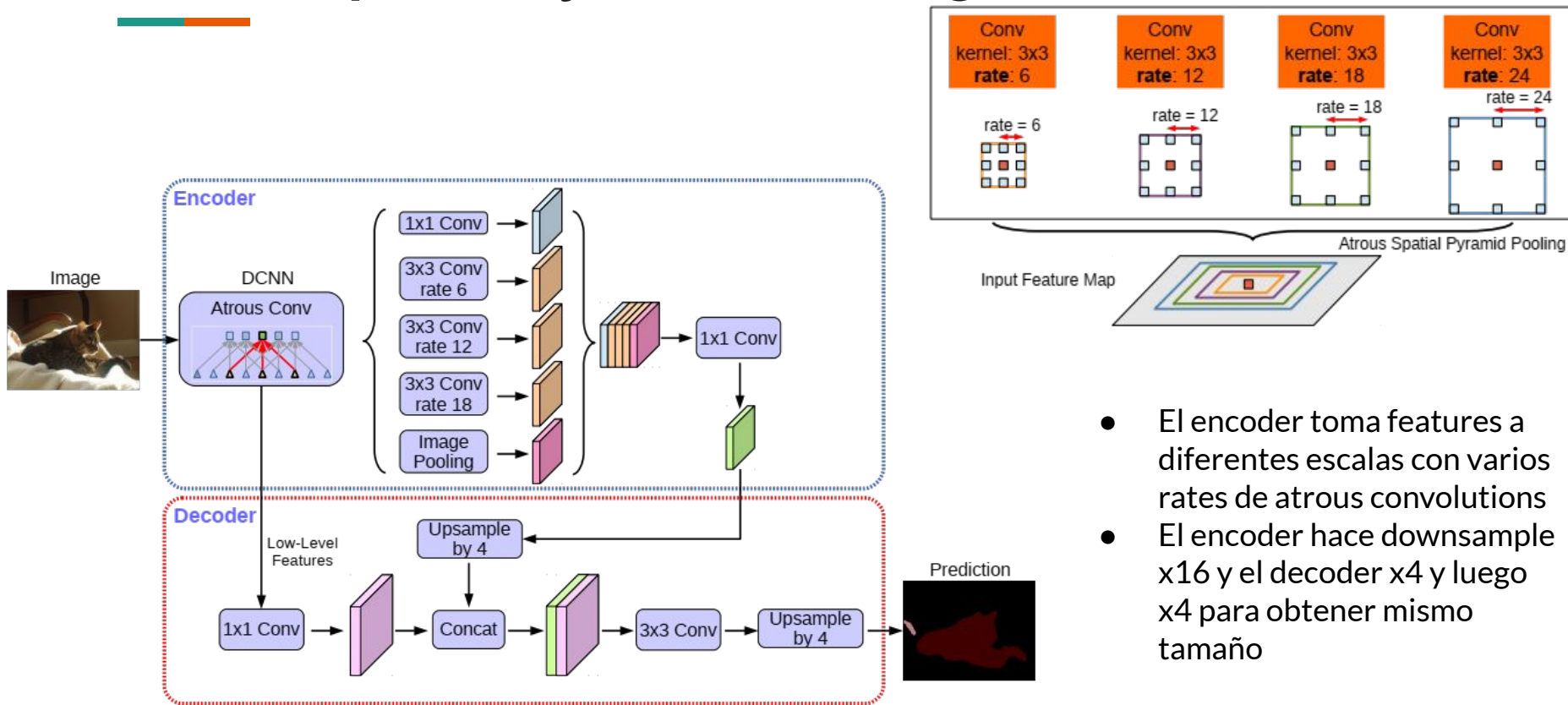


(a) Going deeper without atrous convolution.



(b) Going deeper with atrous convolution. Atrous convolution with $rate > 1$ is applied after block3 when $output_stride = 16$.

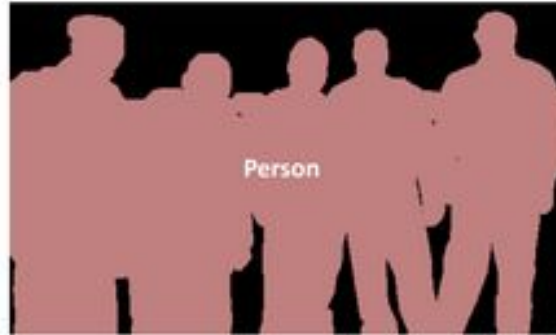
Atrous Spatial Pyramid Pooling



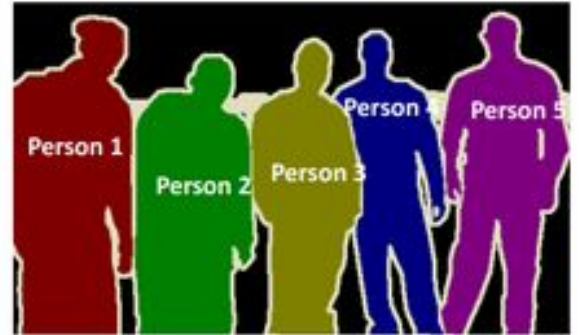
Segmentación de instancias



Object Detection

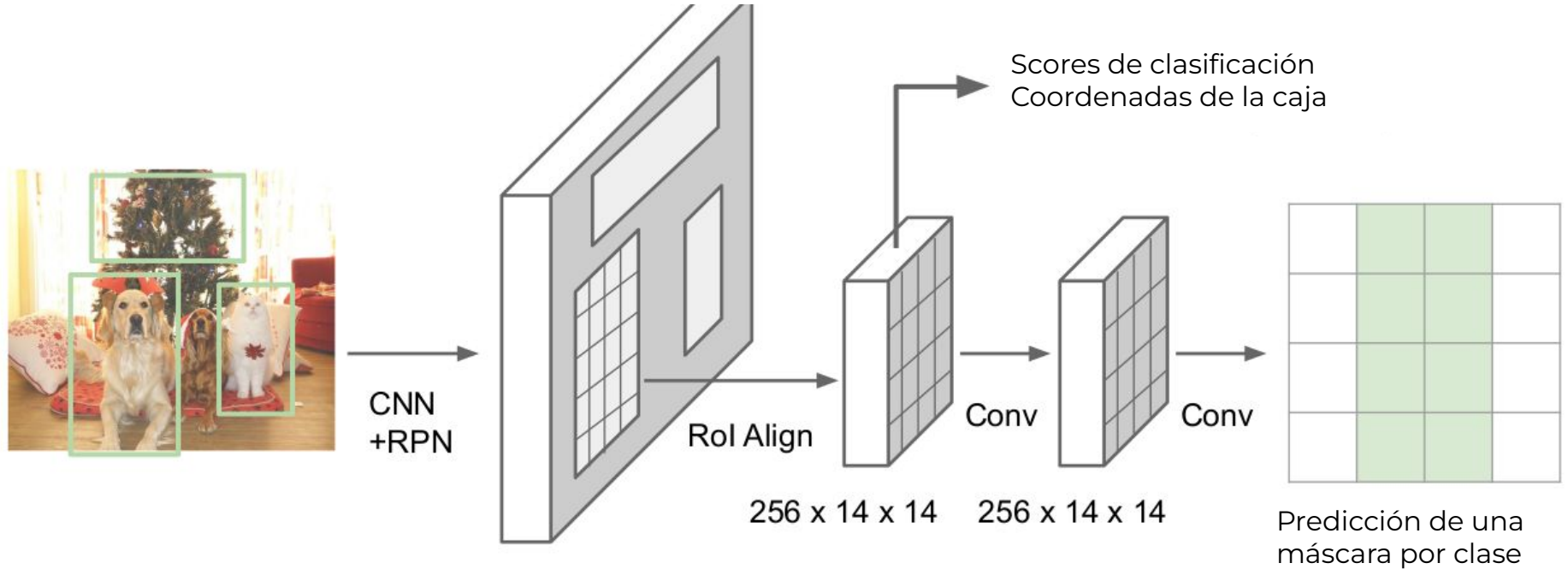


Semantic Segmentation



Instance Segmentation

Mask R-CNN



Mask R-CNN



Proviene de la familia R-CNN, Fast R-CNN y Faster R-CNN.

Como ellas es una red de dos etapas.

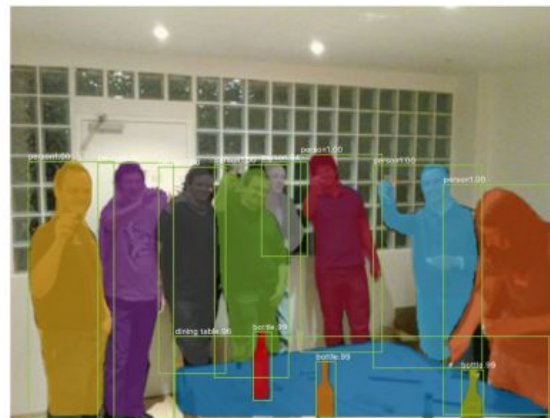
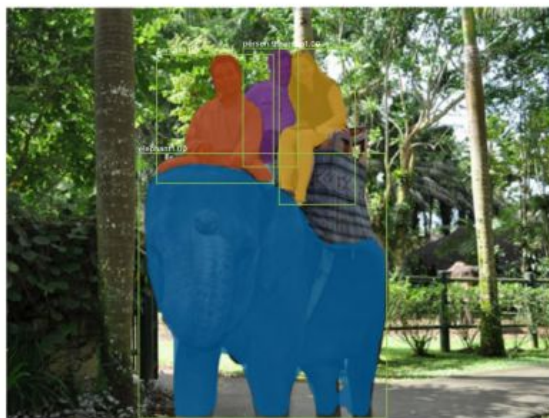
En la segunda etapa se predice una máscara binaria para cada clase en cada una de las RoI, la cual es procesada en paralelo con la predicción de clase y bounding box.

Se define una función de costo multitarea, contemplando un término para la clase, un término para el bounding box y otro para la máscara. Los primeros son idénticos a los de Fast R-CNN.

El tercer término es la binary cross-entropy entre la máscara predicha y la ground truth, pixel a pixel (en la resolución de la máscara).

Solo la máscara de la clase relacionada a la RoI en cuestión contribuye a la función de costo.

Mask R-CNN



Open Source Frameworks



Hay muchas buenas implementaciones de código abierta, en varios lenguajes.

TensorFlow Detection API:

https://github.com/tensorflow/models/tree/master/research/object_detection

Faster RCNN, SSD, RFCN, Mask R-CNN, ...

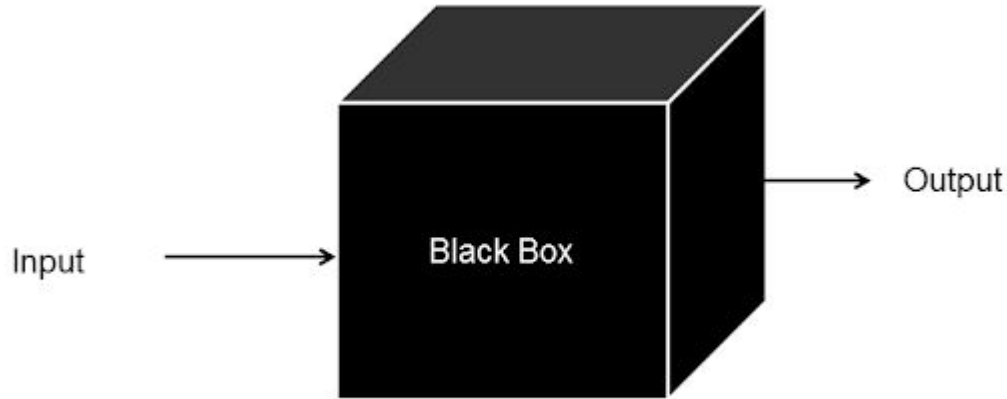
Detectron2 (PyTorch)

<https://github.com/facebookresearch/detectron2>

Mask R-CNN, RetinaNet, Faster R-CNN, RPN, Fast R-CNN, R-FCN, ...

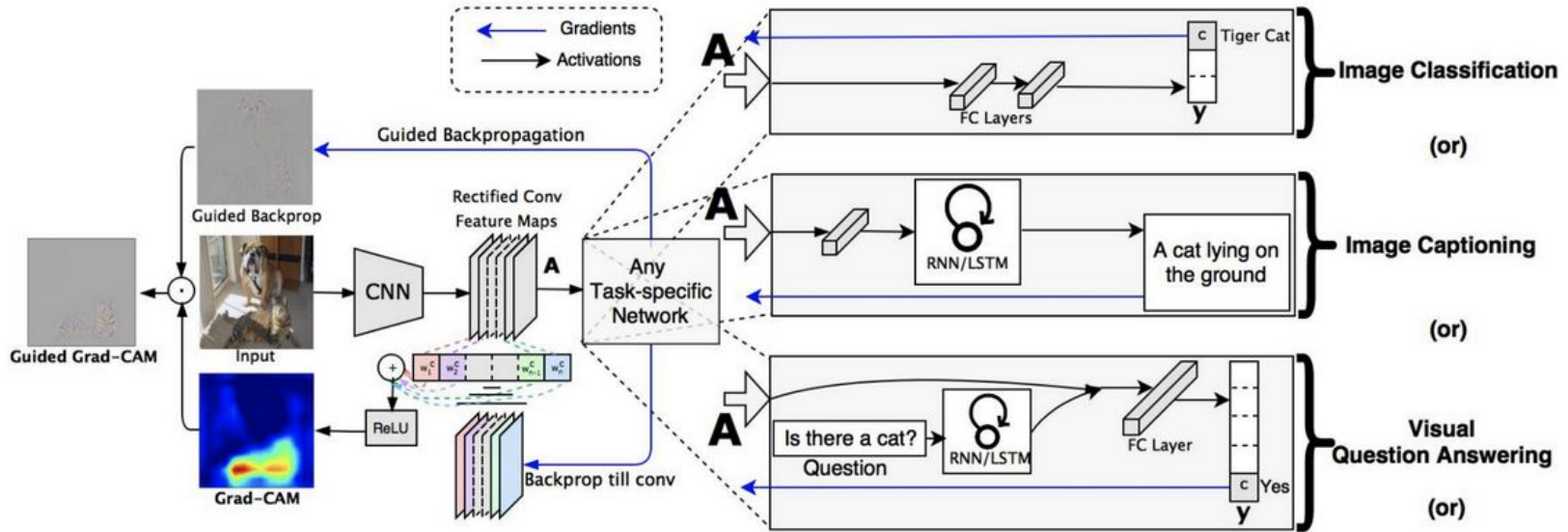
Pueden ser usados para reentrenar modelos haciendo Fine Tuning o Transfer Learning.

Sobre la explicabilidad de las redes convolucionales



Internal behavior of the code is unknown

Grad CAM



Grad CAM

Método para analizar el funcionamiento de casi cualquier red convolucional.

Analiza qué activaciones son más relevantes para la activación de la salida (o alguna neurona intermedia)

Dada las activaciones de interés $A(k)$ se calcula la importancia de cada canal del mapa de características dado para la salida de interés, y luego se calcula la combinación lineal de todos los mapas y sus importancias.

Generalmente se hace sobre las últimas capas que tienen un buen compromiso entre conservación espacial y contenido semántico.

$$\alpha_k^c = \overbrace{\frac{1}{Z} \sum_i \sum_j}^{\text{global average pooling}} \underbrace{\frac{\partial y^c}{\partial A_{ij}^k}}_{\text{gradients via backprop}}$$

$$L_{\text{Grad-CAM}}^c = \text{ReLU} \left(\underbrace{\sum_k \alpha_k^c A^k}_{\text{linear combination}} \right)$$

Grad CAM



Grad CAM: Comparación de modelos



Robot A based it's decision on



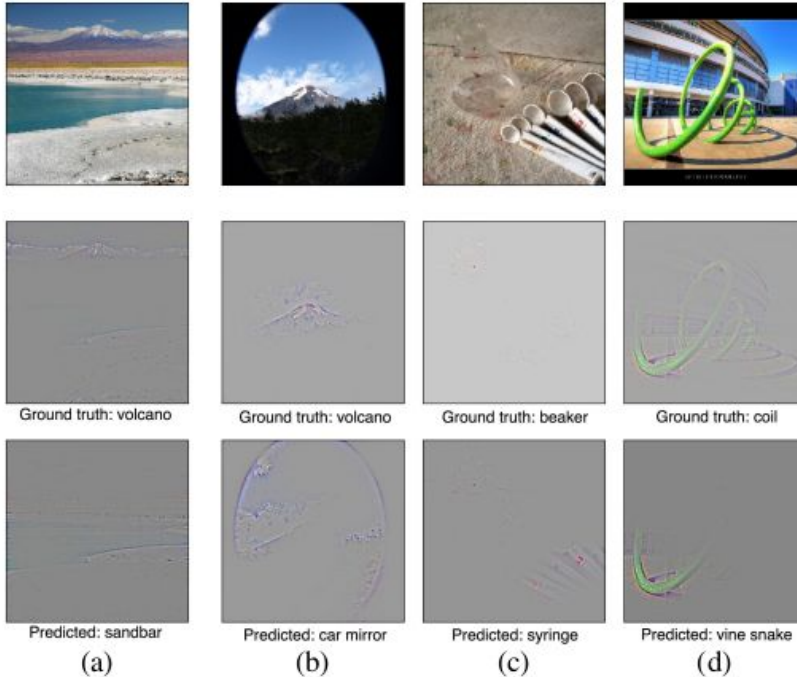
Robot B based it's decision on



Which robot is more reasonable?

- ☐ Robot A seems clearly more reasonable than robot B
- ☐ Robot A seems slightly more reasonable than robot B
- ☐ Both robots seem equally reasonable
- ☐ Robot B seems slightly more reasonable than robot A
- ☐ Robot B seems clearly more reasonable than robot A

Grad CAM: Análisis de modo de fallas

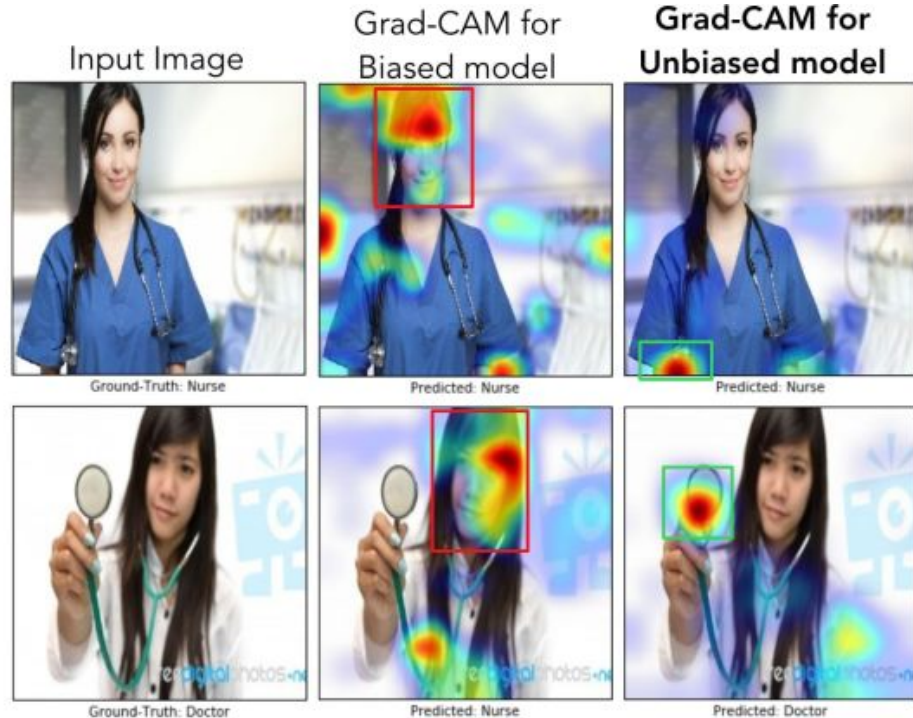


Se comparan los mapas de activación dados por Grad CAM de las clases correspondientes a los Ground Truth y a las salidas predicha, para los casos en los que el modelos dio falsas categorizaciones para las imágenes dadas.

Grad CAM: Análisis de modo de fallas

Un problema muy frecuente en Deep Learning, y uno de los que más se le critica al área es el de los sesgos.

Con Grad CAM podrían analizarse los sesgos en los que incurre determinado modelo, para tomar acciones correctivas.





¡GRACIAS POR SU ATENCIÓN!