

# **RCNN, Fast R-CNN y Faster R-CNN**

Algoritmos para Localización y Detección de Objetos

---

Lesly Mita, Kate Olazabal y Gonzalo Coayla

Universidad Nacional de San Agustín

## 1 Introducción

---

- Preámbulo
- Detección y Segmentación

## 2 RCNN

---

- Modelo
- Métricas de rendimiento
- Búsqueda Selectiva
- Conjunto de Datos y Resultados
- Desventajas

## 3 Fast RCNN

---

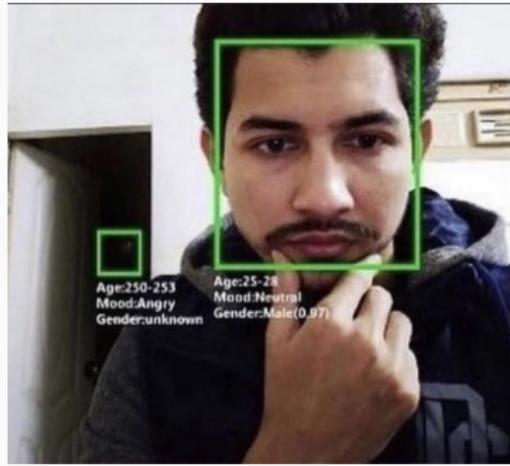
- Aportes
- Arquitectura
- RoI pooling layer
- Resultados

## 4 Faster RCNN

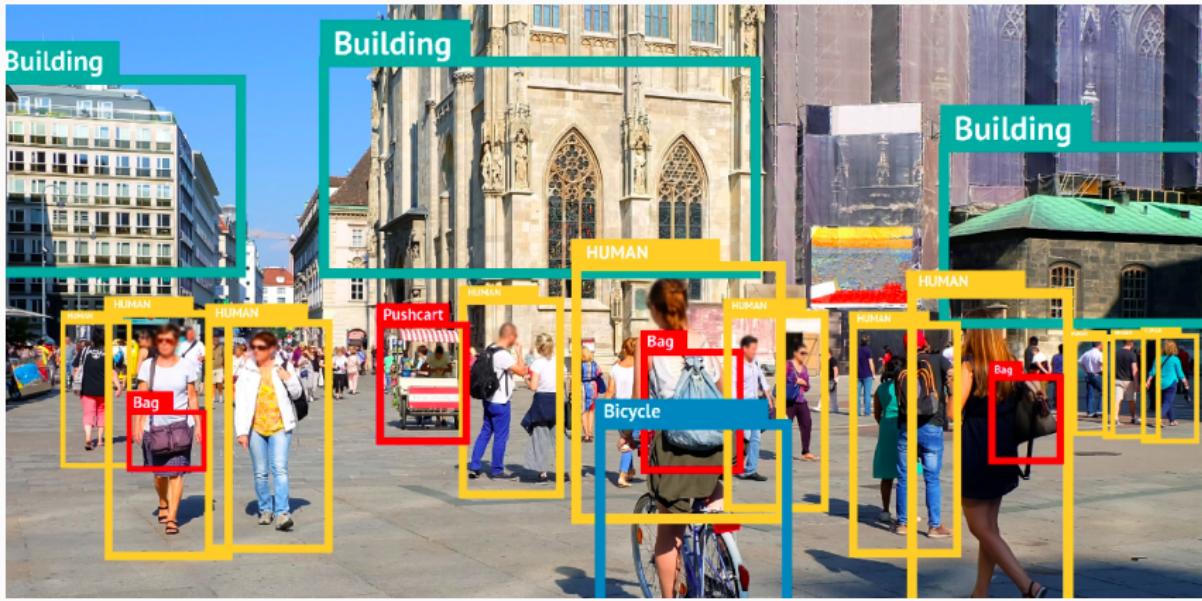
---

- Introducción
- RPM
- Resultados

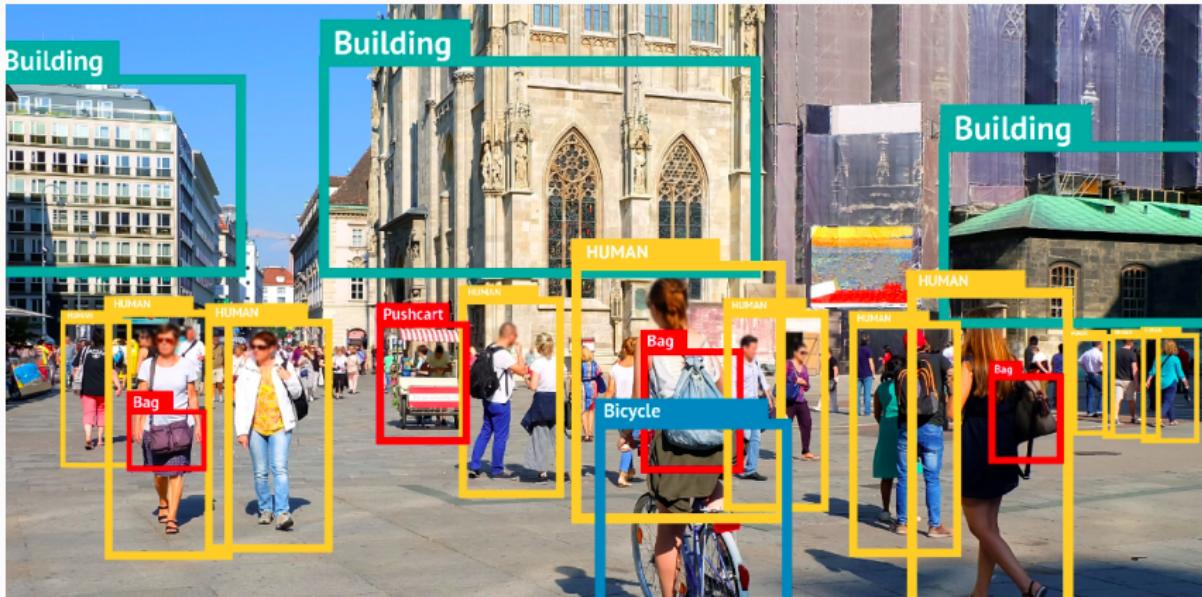
# Introducción Preámbulo



# Introducción Preámbulo

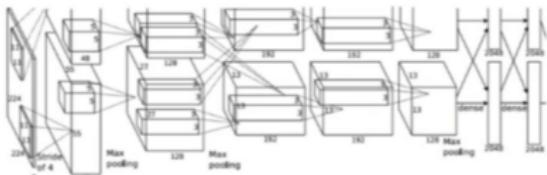


# Introducción Preámbulo



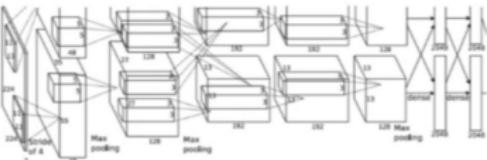
¿Un problema de clasificación de imágenes?

- Identificar múltiples objetos relevantes en una sola imagen
- Proporcionar la localización de los objetos
- Cuadro delimitador alrededor del objeto



CAT:  $(x, y, w, h)$

Figura 1: Cat [4]



DUCK:  $(x, y, w, h)$   
DUCK:  $(x, y, w, h)$

....

Figura 2: Duck [4]

- Diferentes ubicaciones espaciales dentro de la imagen
- Diferentes relaciones de aspecto

Costo computacional

Ross Girshick[6] - Paradigma CNN "reconocimiento usando **regiones**"

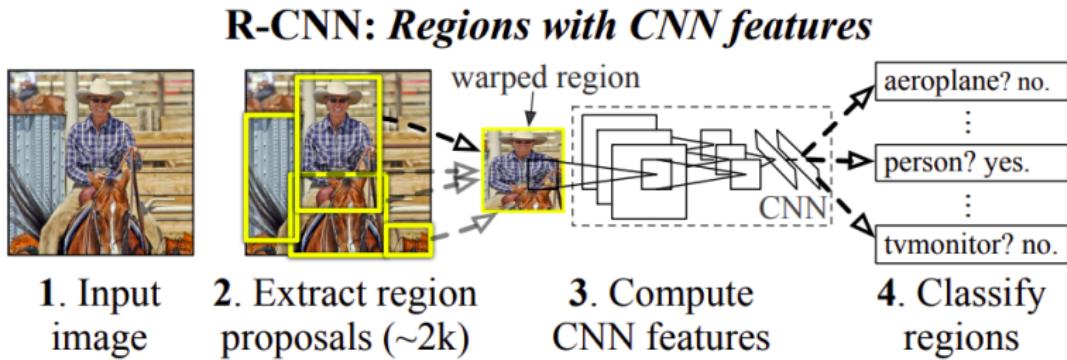


Figura 3: Object detection system overview [6]

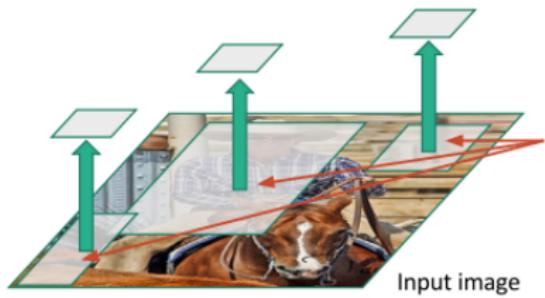


*Figura 4: "Rich feature hierarchies for accurate object detection and semantic segmentation" [7]*



*Figura 4: "Rich feature hierarchies for accurate object detection and semantic segmentation" [7]*

- Regiones de interés (RoI) de 2000 áreas.



- Regiones de imagen redimensionadas.
- Regiones de interés (RoI) de 2000 áreas.

Figura 4: "Rich feature hierarchies for accurate object detection and semantic segmentation" [7]

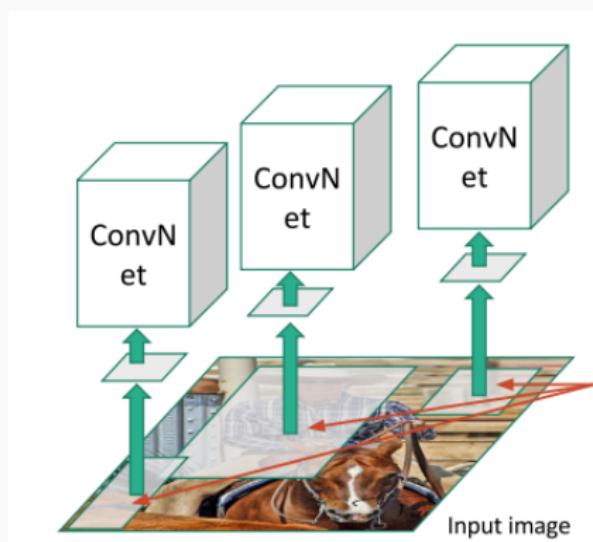
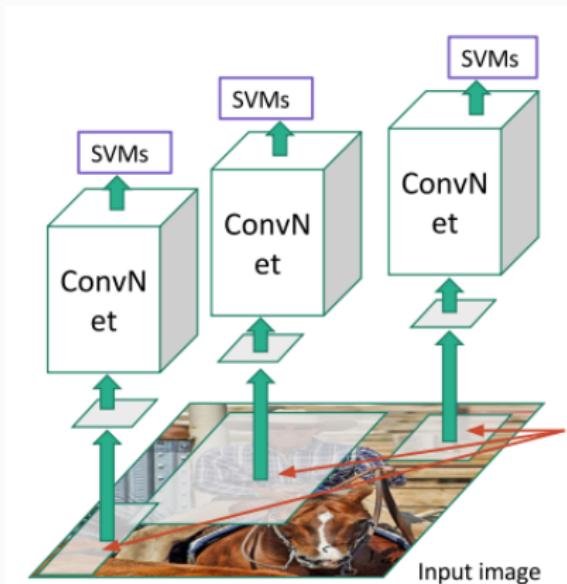


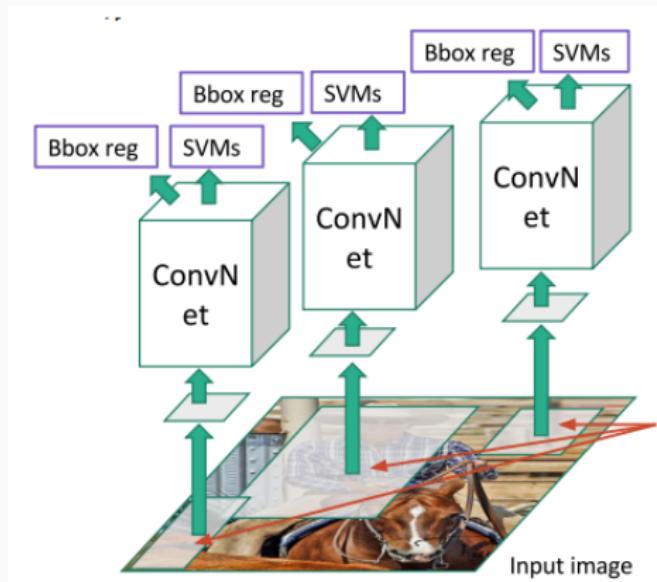
Figura 4: "Rich feature hierarchies for accurate object detection and semantic segmentation" [7]

- Reenviar cada región a través de CNN que produce un vector de características de 4096 dimensiones.
- Regiones de imagen redimensionadas.
- Regiones de interés (Roi) de 2000 áreas.



- Clasificar regiones con SVM
- Reenviar cada región a través de CNN que produce un vector de características de 4096 dimensiones.
- Regiones de imagen redimensionadas.
- Regiones de interés (RoI) de 2000 áreas.

Figura 4: "Rich feature hierarchies for accurate object detection and semantic segmentation" [7]



- Regresión lineal para desplazamientos de cuadro delimitador.
- Clasificar regiones con SVM
- Reenviar cada región a través de CNN que produce un vector de características de 4096 dimensiones.
- Regiones de imagen redimensionadas.
- Regiones de interés (RoI) de 2000 áreas.

Figura 4: "Rich feature hierarchies for accurate object detection and semantic segmentation" [7]

Intersección sobre Unión (IoU) =  $\frac{\text{superposicin}}{\text{union}}$

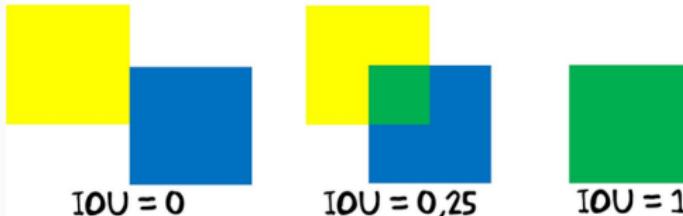


Figura 5: por Oleksii Sheremet con Microsoft Visio

## Precisión promedio media (mAP)

- Media de las precisiones promedio calculadas sobre todas las clases

## Puntuación de mAP

- Se calcula para un IoU fijo
- Media de las puntuaciones de mAP para valores de IoU variables
- Penalizar una gran cantidad de cuadros delimitadores con clasificaciones incorrectas.

## Algoritmo de búsqueda selectiva

1. Generar una subsegmentación inicial, generamos muchas regiones candidatas.
2. Usar un algoritmo voraz para combinar recursivamente regiones similares en otras más grandes.
3. Usar las regiones generadas para producir las propuestas finales de regiones candidatas.

El número de regiones producidas depende de la resolución de la imagen.

Ancho fijo a 500 píxeles

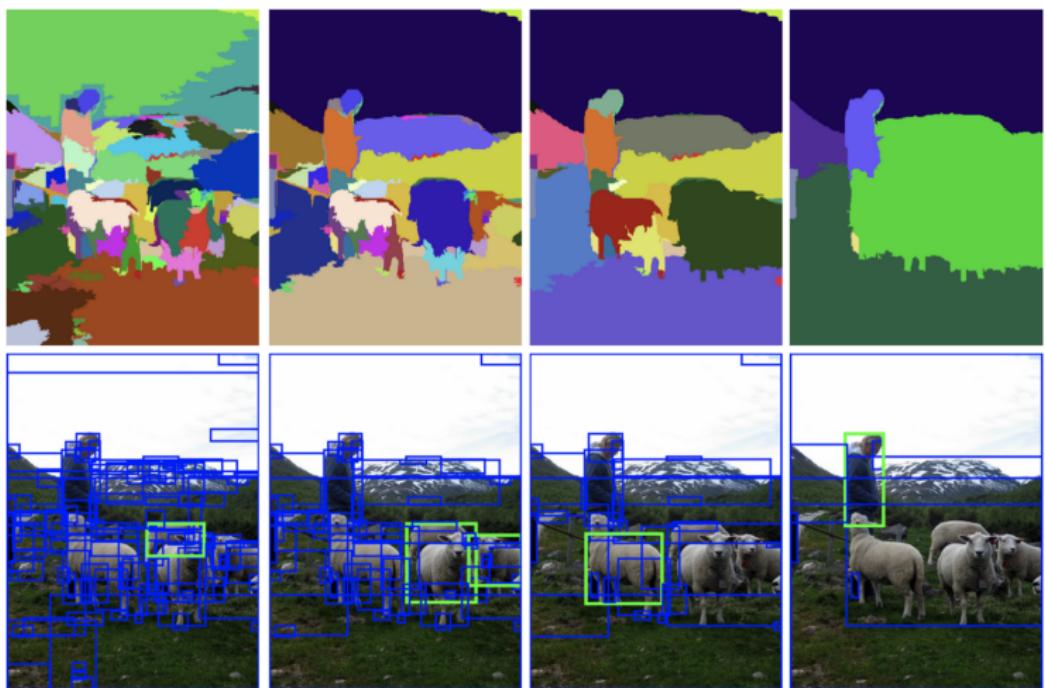


Figura 6: "Selective Search for Object Recognition" [11]

1. Dataset ImageNet de 2012
  - Puntuación de mAP del 31,4% **aumento de 22 puntos**
2. Dataset PASCAL VOC 2012
  - Puntuación de mAP del 62,4% **aumento de 7,1 puntos**
  - Se ajusta las regiones a un IoU > 0.5.
  - SVM se entrena para cada clase de cada conjunto de datos.

## Aplicación CNN

Propuestas de regiones ascendentes para localizar y segmentar objetos

## Pre-entrenamiento supervisado específico del dominio

- Efectivo entrenamiento supervisado para una tarea auxiliar con abundantes datos *clasificación de imágenes*
- Ajustar la red para la tarea de destino con datos escasos *detección*.

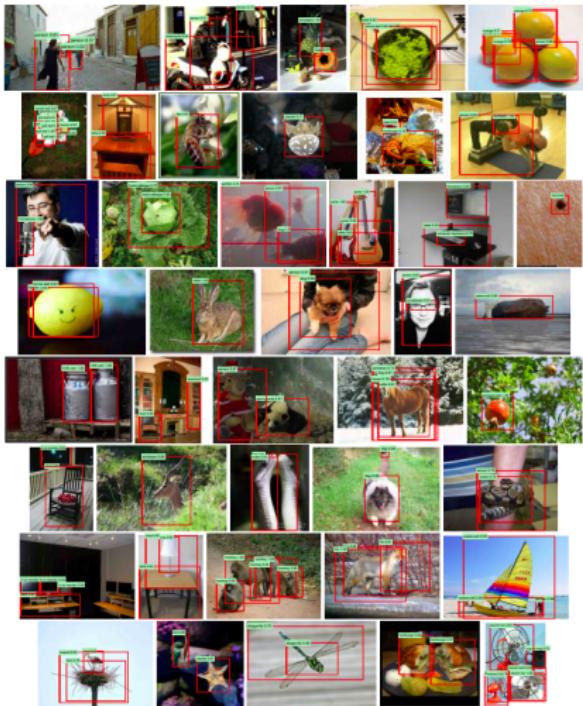


Figura 7: Curated examples. Each image was selected because we found it impressive, surprising, interesting, or amusing. [6]

- El **entrenamiento es lento** (84 h), ocupa mucho espacio en disco. Por clasificar 2000 propuestas de regiones por imagen.
- No se puede implementar en **tiempo real** ya que toma alrededor de 47 segundos para cada imagen de prueba.
- El algoritmo de **búsqueda selectiva** es un algoritmo fijo.

El método de la red convolucional basada en regiones (RCNN) [6] consigue una excelente precisión en la detección de objetos utilizando una ConvNet profunda para clasificar las propuestas de objetos. Sin embargo, R-CNN tiene notables inconvenientes:

- El training es un proceso de varias etapas
- El training es costoso en espacio y tiempo
- La detección de objetos es lenta

Fast RCNN [5] es un detector de objetos desarrollado exclusivamente por Ross Girshick, un investigador de IA de Facebook y antiguo investigador de Microsoft. Es un algoritmo de entrenamiento que soluciona las desventajas de R-CNN, al tiempo que mejora su velocidad y precisión. Se llamó Fast RCNN porque es comparativamente rápido de entrenar y probar. Tiene varias ventajas:

1. Higher detection quality (mAP) than R-CNN, SPPnet
2. Training is single-stage, using a multi-task loss
3. Training can update all network layers
4. No disk storage is required for feature caching

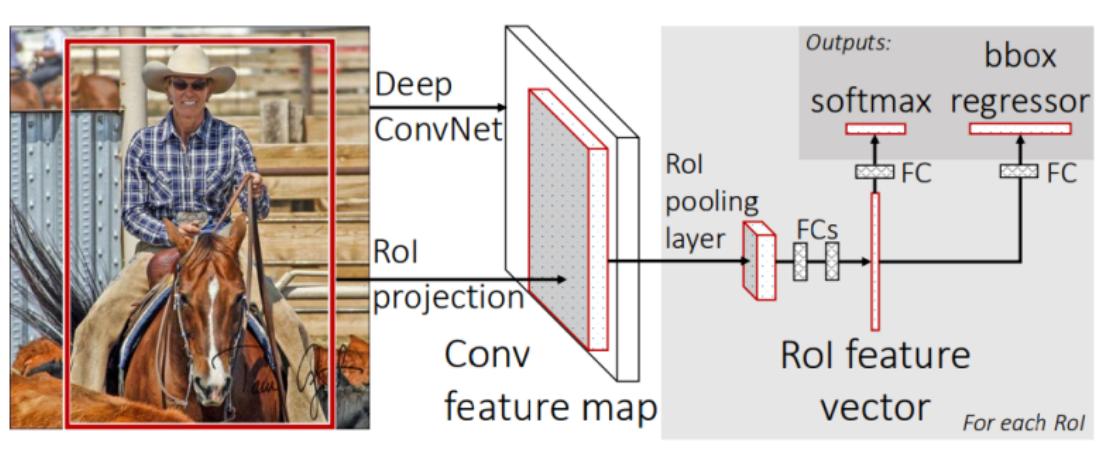


Figura 8: Fast R-CNN architecture [4]

Una imagen de entrada y múltiples regiones de interés (Rois) se introducen en una red totalmente convolucional. Cada región de interés se agrupa en un mapa de características de tamaño fijo y luego se asigna a un vector de características mediante capas totalmente conectadas (FC).

## Roi pooling layer

El mapa de características de la última capa convolucional se introduce en una capa de agrupación de ROI. La razón es extraer un vector de características de longitud fija de cada propuesta de región.

# Roi pooling layer

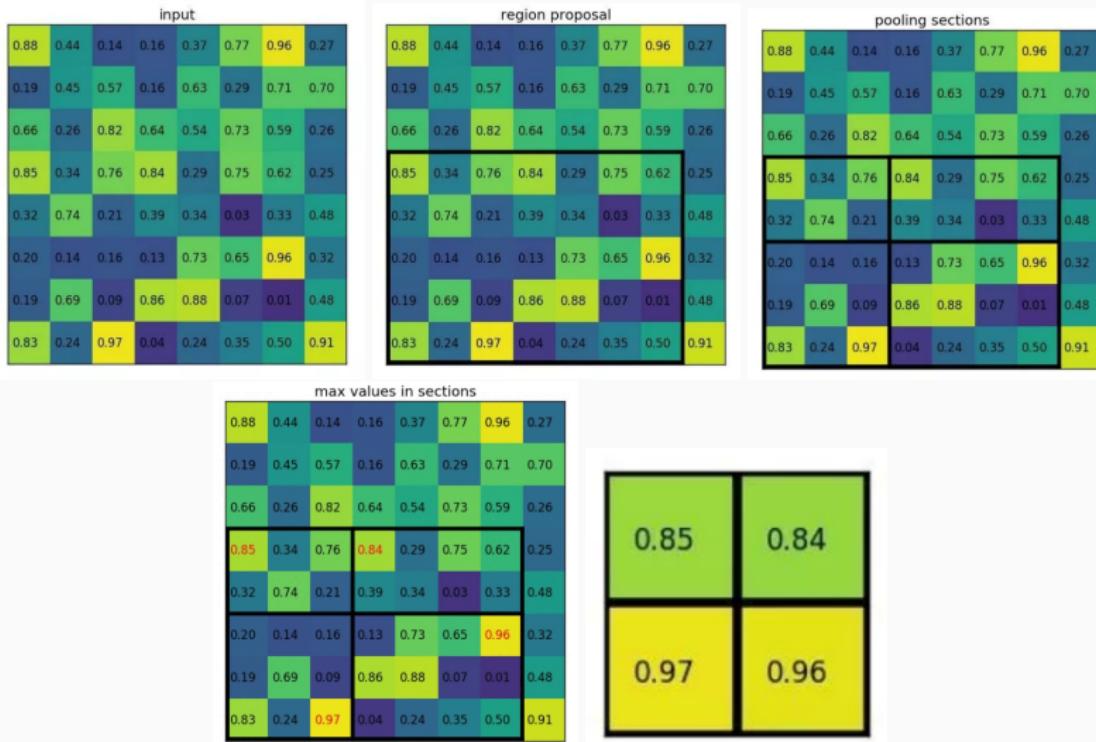


Figura 9: Example of Roi pooling layer

## Roi pooling layer

El vector de características extraído mediante el ROI Pooling se pasa a continuación a algunas capas FC. La salida de la última capa FC se divide en 2 ramas:

- Capa Softmax para predecir las puntuaciones de las clases
- Capa FC para predecir los cuadros delimitadores de los objetos detectados

# Resultados

- La Fast R-CNN entrena la red VGG16 9 veces más rápido que la R-CNN.
- En tiempo de ejecución, la red de detección procesa las imágenes en 0,3s (excluyendo el tiempo de propuesta de objetos)
- Con un mAP de 66% (vs. 62% de R-CNN)

## Principales contribuciones

- Propone una nueva capa llamada ROI Pooling que extrae vectores de características de igual longitud de todas las propuestas (es decir, ROIs) en la misma imagen.
- En comparación con R-CNN, que tiene múltiples etapas (generación de propuestas de regiones, extracción de características y clasificación mediante SVM), Faster R-CNN construye una red que sólo tiene una etapa.
- Faster R-CNN comparte los cálculos (es decir, los cálculos de la capa convolucional) entre todas las propuestas (es decir, las ROI) en lugar de realizar los cálculos para cada propuesta de forma independiente. Para ello se utiliza la nueva capa ROI Pooling, que hace que Fast R-CNN sea más rápida que R-CNN.
- Fast R-CNN no almacena en caché las características extraídas y, por tanto, no necesita tanto almacenamiento en disco en comparación con R-CNN, que necesita cientos de gigabytes.
- Fast R-CNN es más preciso que R-CNN.

- **RCNN y Fast RNCC** utilizan la búsqueda selectiva para hallar las regiones propuestas de la imagen, esta función es muy lenta y afecta el rendimiento de la red.
- La búsqueda selectiva funciona de manera separada a la red, esto causa que en caso este algoritmo llegue a algún error, la red no pueda aprender de este

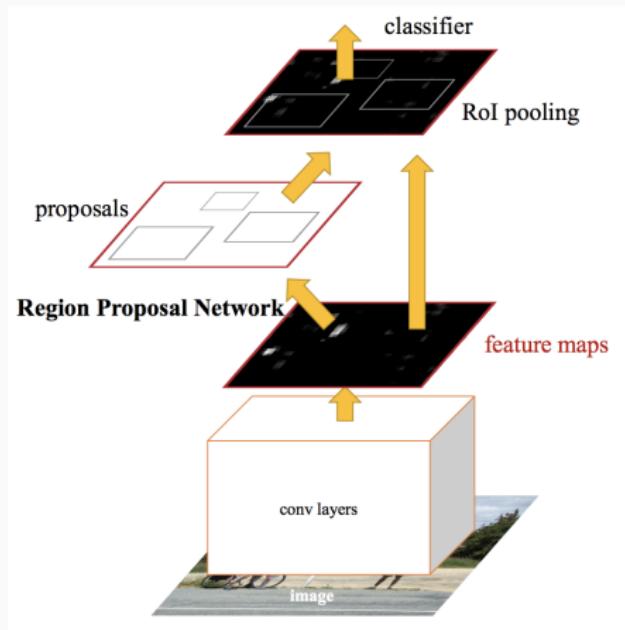


Figura 8: "Faster R-CNN" [9]

- Se elimino la búsqueda selectiva reemplazándola con una red llamada Region Proposal Network (RPN)
- Similar a Fast RCNN la CNN provee a la RPN de un mapa de características
- La RPN provee a la RoI pooling layer de las regiones propuestas y se continua con el proceso seguido en la Fast RCNN.

# Faster RCNN RPM

- La RPM se compone de una primera capa convolucional con un filtro de  $3 \times 3$ , y de dos capas convolucionales con filtro de  $1 \times 1$  una dedicada a la clasificación con salida  $2x$  y otra a la regresión con salida  $4x$
- Se introducen las anclas y la ventana deslizante, estas anclas serán el centro de la ventana deslizante

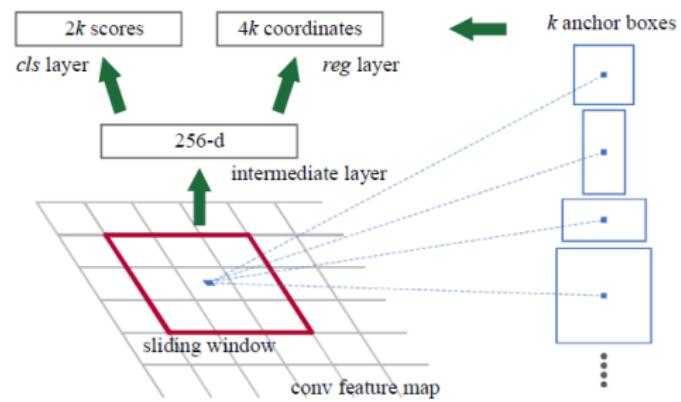
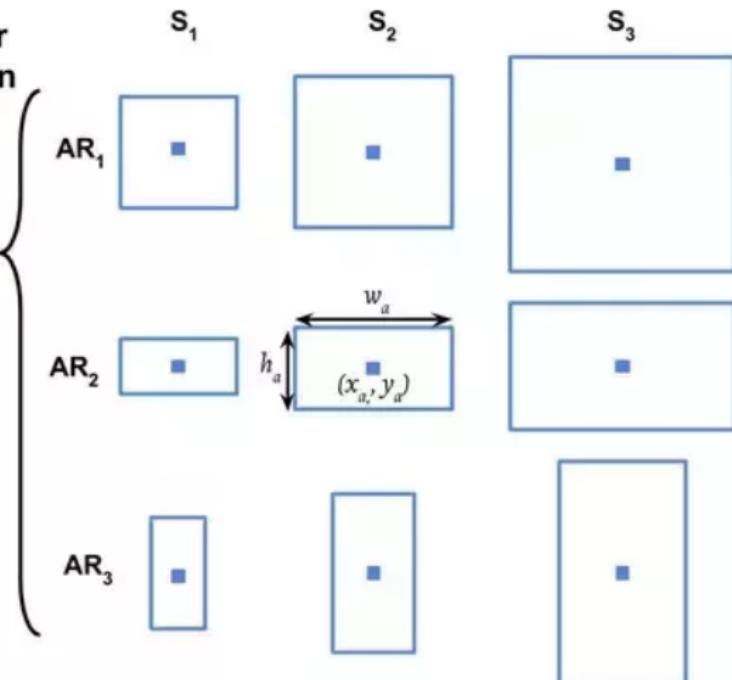
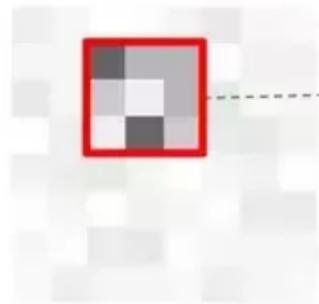


Figura 9: "Mapa y salida de la RPM" [9]

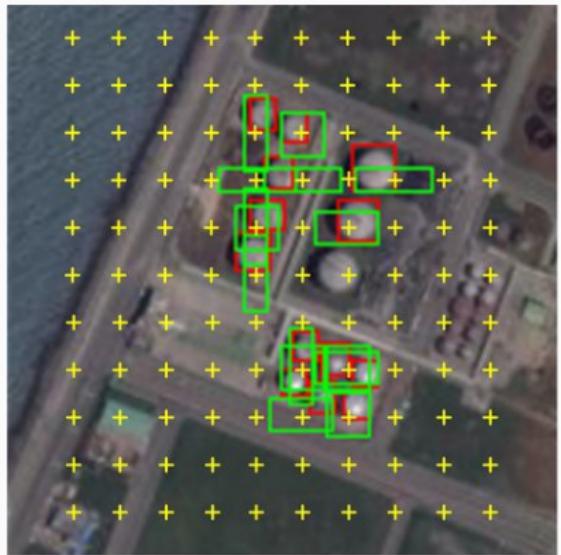
Generate 9 anchors for each **sliding window** on conv. feature map



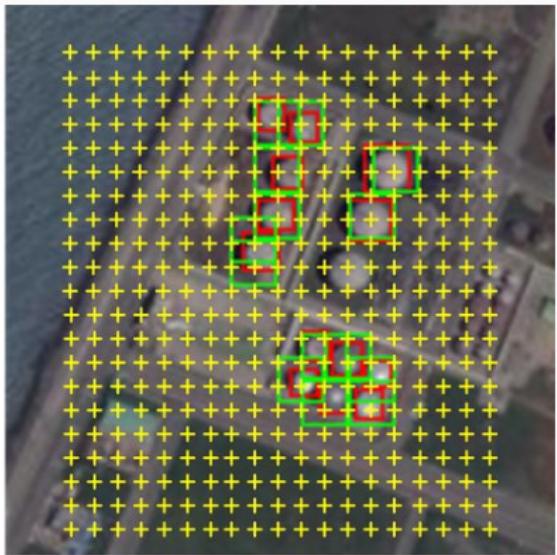
$w_a$ : anchor's width  
 $h_a$ : anchor's height  
 $x_a, y_a$ : anchor's center

@vmirly

Figura 10: Generación de anclas [10]



(a)  $S_A = 16$



(b)  $S_A = 8$

Figura 10: Anclas en la imagen [10]

## Estructura de RPM

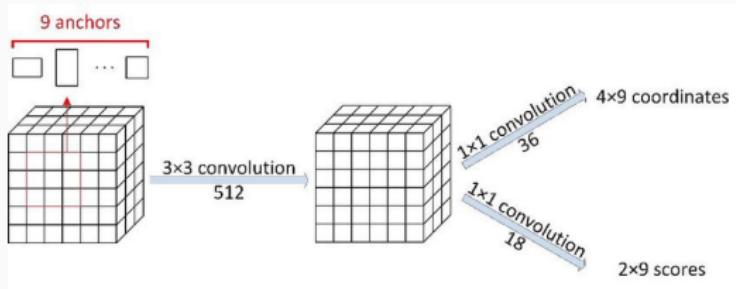


Figura 5: Estructura de una RPM [8]

- La primera capa nos devuelve 4 valores por ancla, x,y los centros y w,h el alto y ancho con un desplazamiento que ajusta el tamaño de la caja
- La segunda capa nos devuelve dos valores, cada uno con la probabilidad de que el ancla tenga o no un objeto

## Entrenamiento

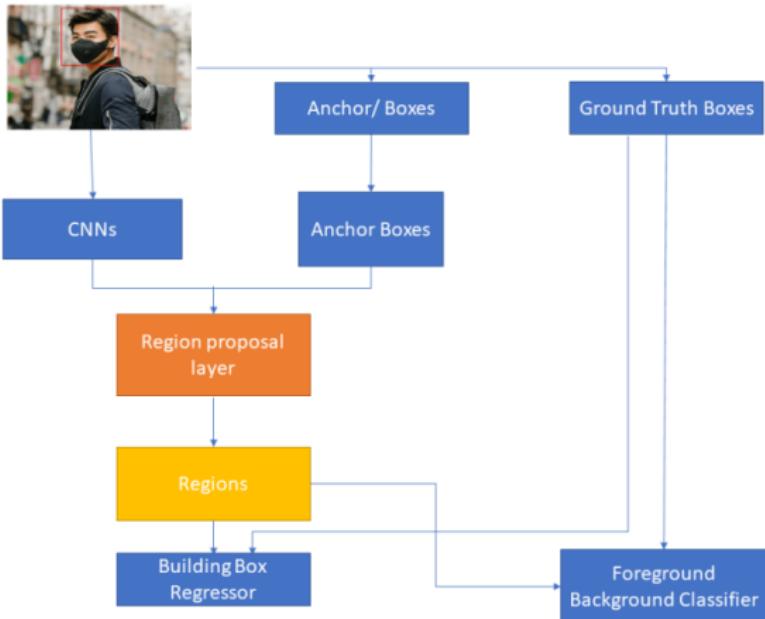


Figura 11: RPM en entrenamiento [8]

## Función de perdida

$$\begin{aligned} L(\{p_i\}, \{t_i\}) = & \frac{1}{N_{cls}} \sum_i L_{cls}(p_i, p_i^*) \\ & + \lambda \frac{1}{N_{reg}} \sum_i p_i^* L_{reg}(t_i, t_i^*). \end{aligned}$$

*Función de perdida*

- Se divide en dos partes, la primera relacionada a la clasificación que utiliza la probabilidad P de que el ancla contenga o no un objeto
- La segunda relacionada con la regresión donde solo tienen efectos las anclas que fueron clasificadas como que contienen un objeto

# Resultados

- En pruebas realizadas Faster RCNN es capaz de detectar hasta en 59ms con una red ZFNet y 198ms con una red VGGNet haciendo posible su uso en casos donde se necesite procesamiento cercano al tiempo real
- En pruebas con datasets populares obtuvo en general igual o mayor mAP en comparación a Fast RCNN

- [1] P. Adams.  
**The title of the work.**  
*The name of the journal*, 4(2):201–213, 7 1993.  
An optional note.
- [2] P. Babington.  
**The title of the work, volume 4 of 10.**  
The name of the publisher, The address, 3 edition, 7 1993.  
An optional note.
- [3] P. Eston.  
**The title of the work, volume 4 of 5, chapter 8, pages 201–213.**  
The name of the publisher, The address of the publisher, 3 edition, 7 1993.  
An optional note.
- [4] R. Gandhi.  
**R-cnn, fast r-cnn, faster r-cnn, yolo – object detection algorithms**, 12 2018.
- [5] R. Girshick.  
**Fast r-cnn.**  
In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, December 2015.

## References II

- [6] R. Girshick, J. Donahue, T. Darrell, and J. Malik.  
**Rich feature hierarchies for accurate object detection and semantic segmentation.**  
*Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 11 2013.
- [7] R. Girshick, J. Donahue, T. Darrell, and J. Malik.  
**Rich feature hierarchies for accurate object detection and semantic segmentation.**  
In *2014 IEEE Conference on Computer Vision and Pattern Recognition*, pages 580–587, 2014.
- [8] S. K.  
**Region proposal network – a detailed view**, 12 2019.
- [9] S. Ren, K. He, R. Girshick, and J. Sun.  
**Faster r-cnn: Towards real-time object detection with region proposal networks.**  
In C. Cortes, N. Lawrence, D. Lee, M. Sugiyama, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 28. Curran Associates, Inc., 2015.
- [10] V. Sagar, S. Jain, and V. BS.  
**Yield estimation using faster r-cnn.**  
05 2018.

## References III

- [11] J. Uijlings, K. Sande, T. Gevers, and A. Smeulders.  
**Selective search for object recognition.**  
*International Journal of Computer Vision*, 104:154–171, 09 2013.