

# Invatare Automata in Vedere Artificiala

Curs 4: Detectia de obiecte

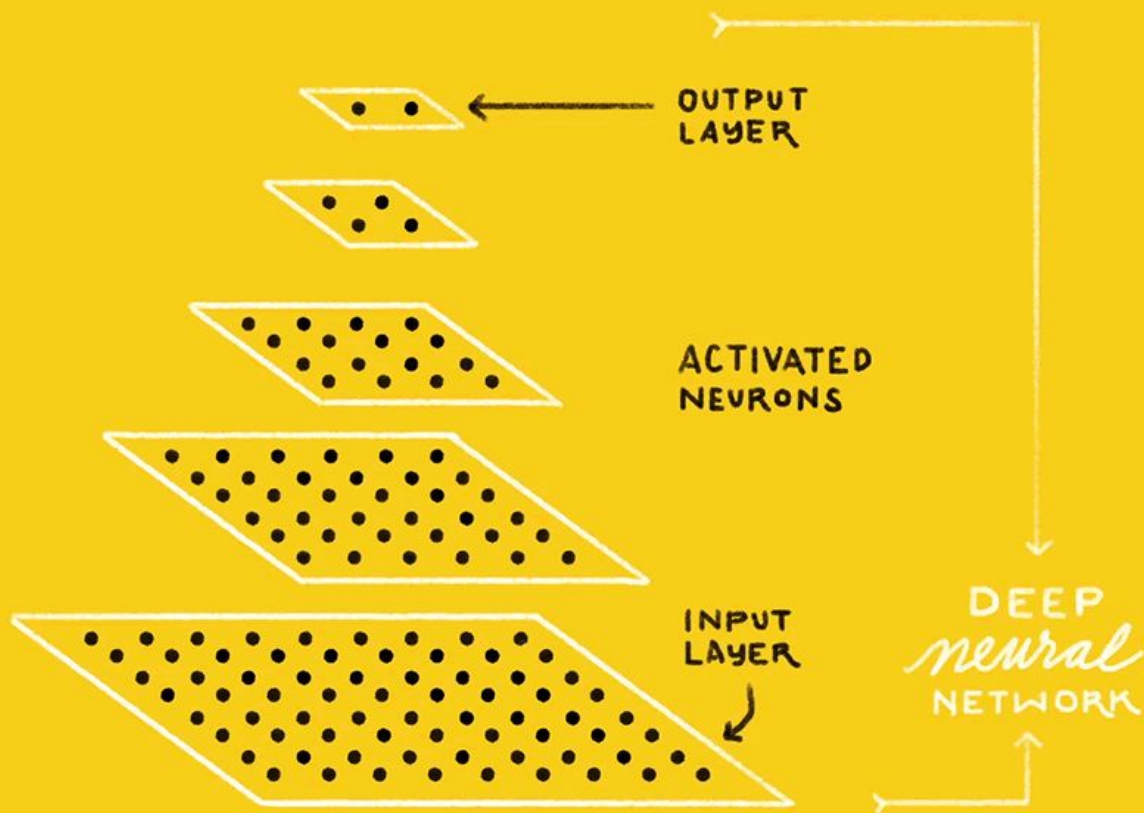


# Recapitulare

IS THIS A  
**CAT** or **DOG**?



CAT DOG

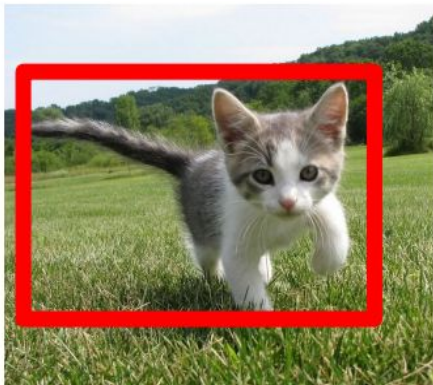


# Task-uri in Computer Vision

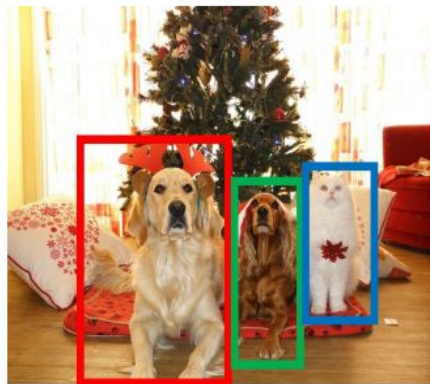
**Semantic  
Segmentation**



**Classification  
+ Localization**



**Object  
Detection**



**Instance  
Segmentation**

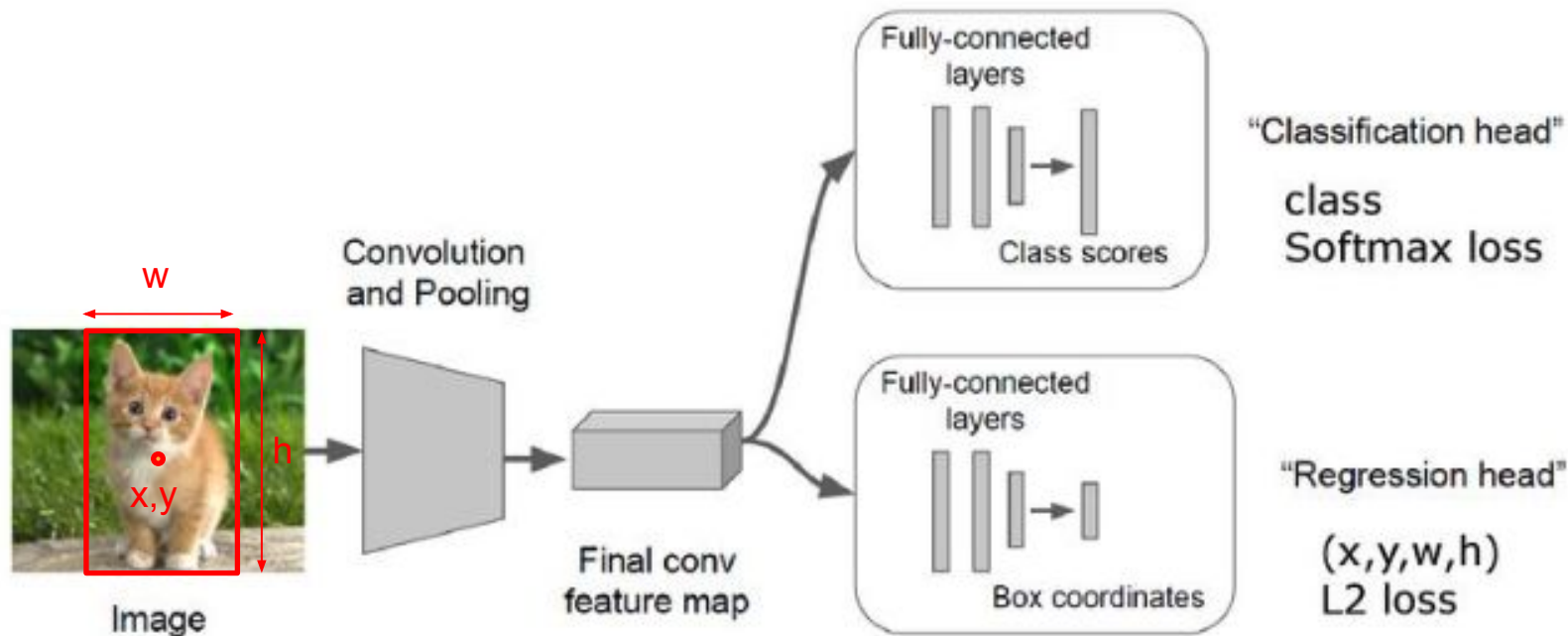


Un singur obiect

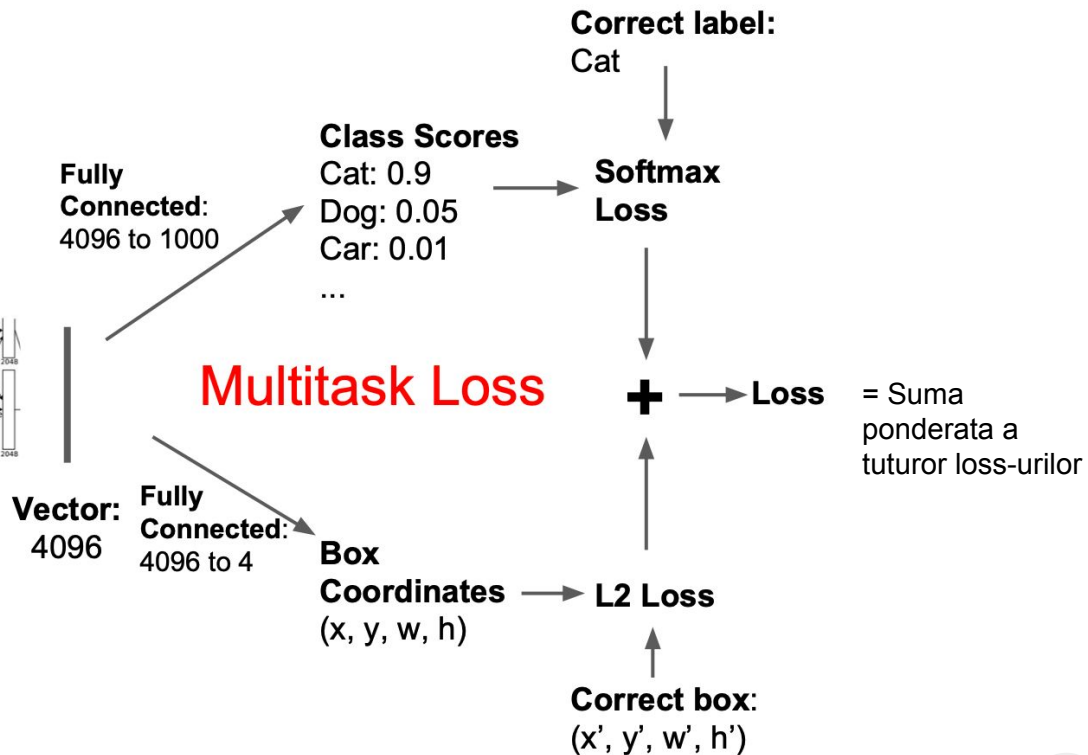
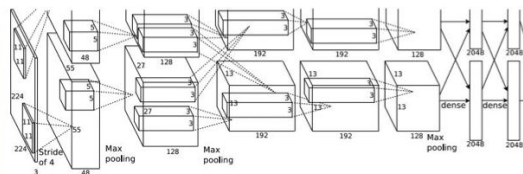
Mai multe obiecte

# Clasificare + Localizare

- In imagine este prezent un singur obiect. Q: este o pisica in imagine? Daca da, unde se afla aceasta in imagine?
- Output: Clasa obiectului + Coordonatele bounding box-ului ( $x, y, w, h$ )



# Clasificare + Localizare



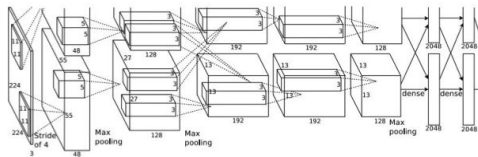
# Clasificare + Localizare

- Antrenam rețeaua pentru a obține un label pentru clasificare și un bounding box pentru localizare
- Localizarea este tratată ca o problemă de regresie
- Layerurile convoluționale sunt conectate la:
  - Layeruri fully-connected pentru clasificare
  - Layeruri fully-connected pentru cele 4 numere care definesc bounding box-ul
- $\text{Loss} = \alpha * \text{Loss}_{\text{Classification}} + (1 - \alpha) * \text{Loss}_{\text{Localization}}$ 
  - Localizare: L1, L2, smooth L1 etc.
  - Clasificare: Softmax

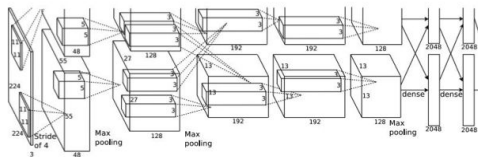


# Detectia de obiecte

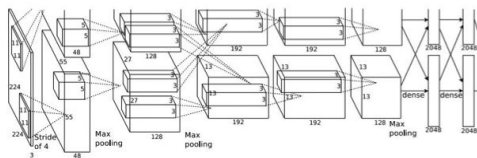
- Numar necunoscut de obiecte pe care vrem sa le detectam in imagine
- Modelul trebuie sa prezica un numar diferit de output-uri in functie de imaginea de input
- Output: Pentru fiecare obiect din imagine prezicem clasa obiectului si bounding box-ul acestuia (coordonatele)



CAT: (x, y, w, h)  $\longrightarrow 1 \times 4 = 4$  numere



DOG: (x, y, w, h)  
DOG: (x, y, w, h)  $\longrightarrow 3 \times 4 = 12$  numere  
CAT: (x, y, w, h)

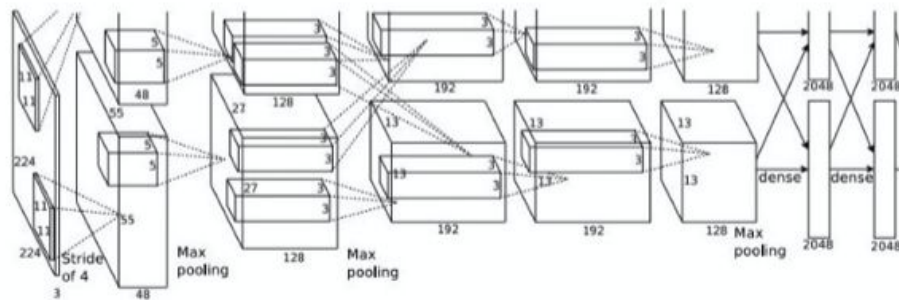


DUCK: (x, y, w, h)  
DUCK: (x, y, w, h)  $\longrightarrow n \times 4 = \dots$  numere



# Detectia de obiecte: Sliding Window

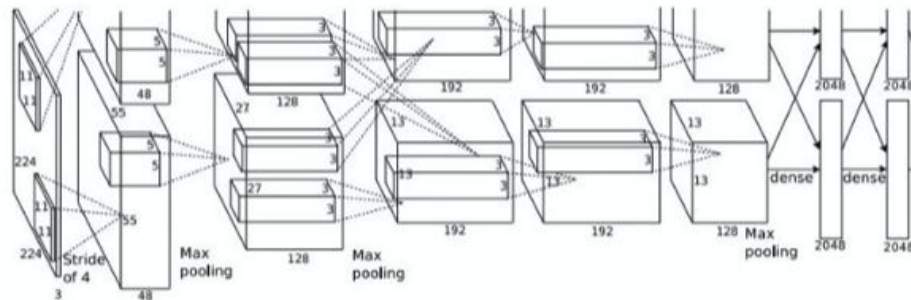
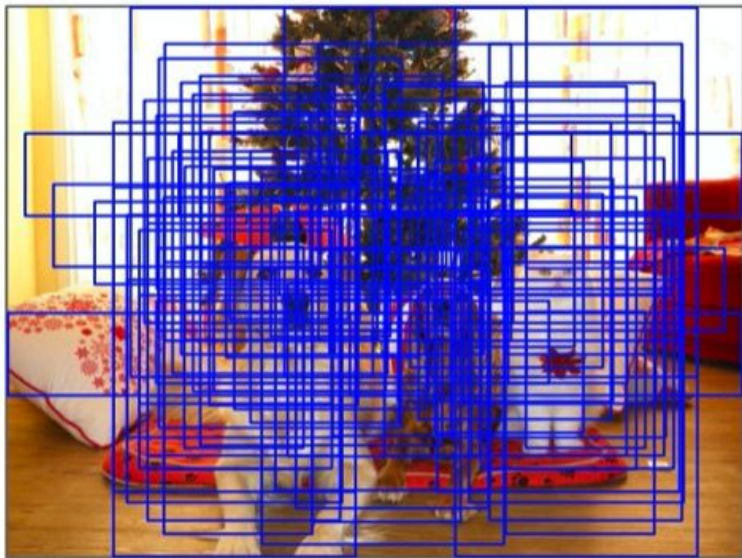
- Aplicarea unei rețele convolutive (CNN) pe mai multe crop-uri diferite din imagine
- Reteaua trebuie sa clasifice fiecare crop din imagine pe care il primeste.  
Output: obiect sau background





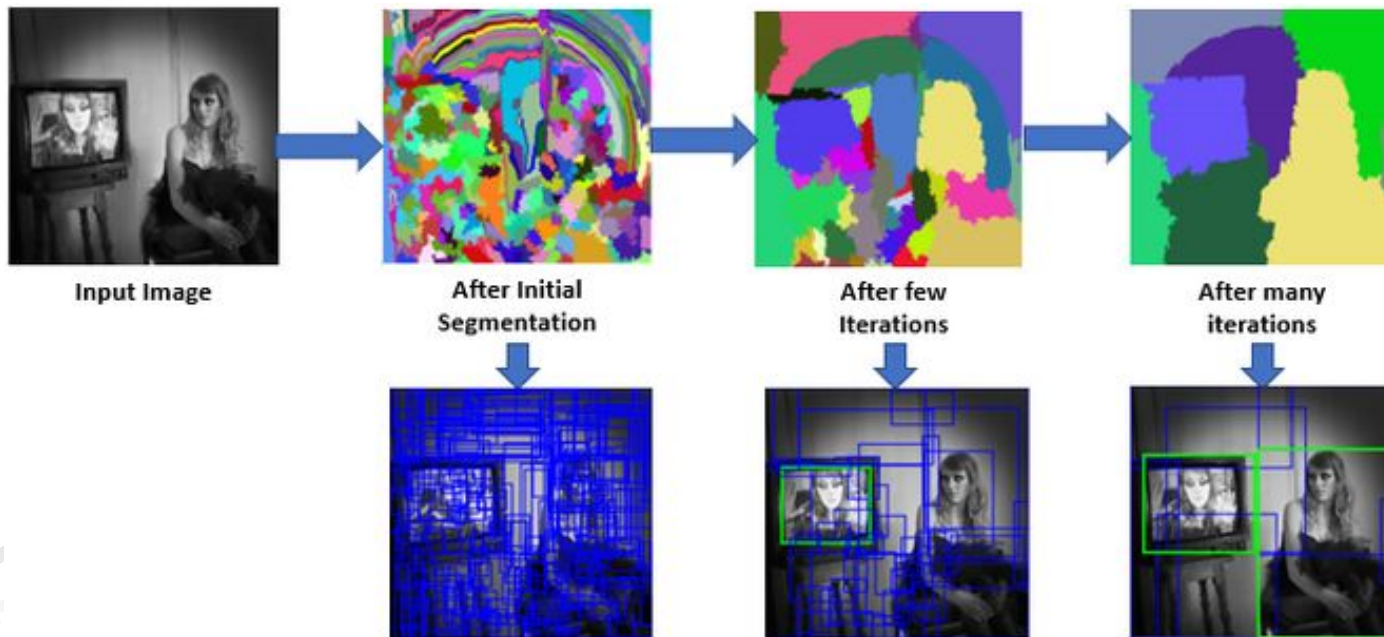
# Detectia de obiecte: Sliding Window

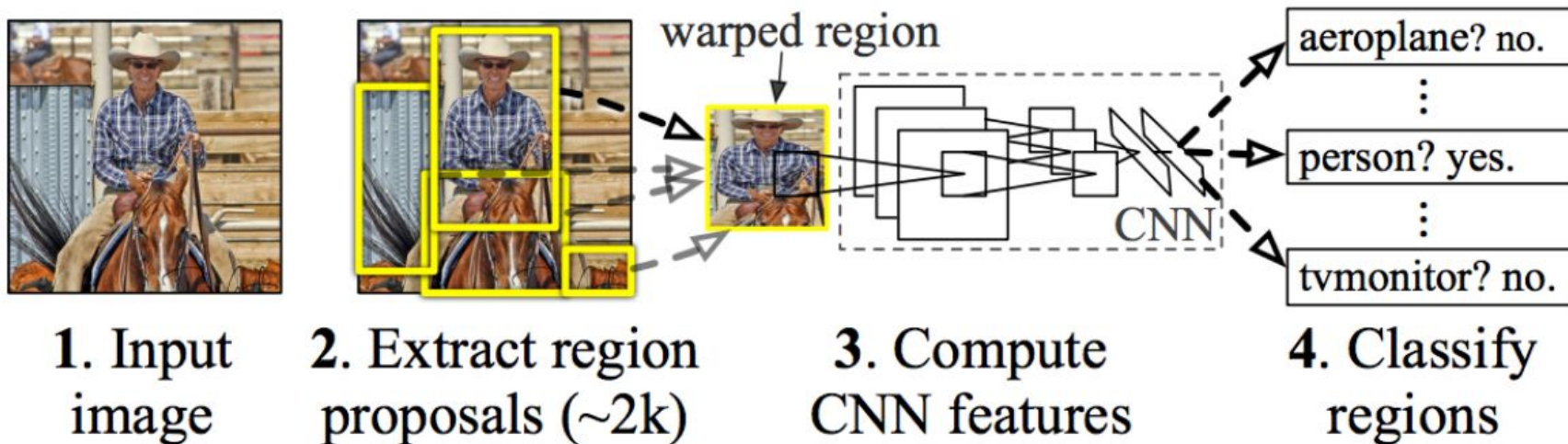
- Problema: Va trebui sa aplicam CNN-ul pe crop-uri de diferite scale-uri si aspect ratio-uri pentru un numar foarte mare de locatii => Computationally expensive!



# Detectia de obiecte: Region Proposal

- Alegerea “automata” a regiunilor din imagine care ar putea contine obiecte
- Selective search – algoritm pentru selectarea eficienta a regiunilor
- Tehnici clasice de computer vision (ex. Detectia blob-urilor)

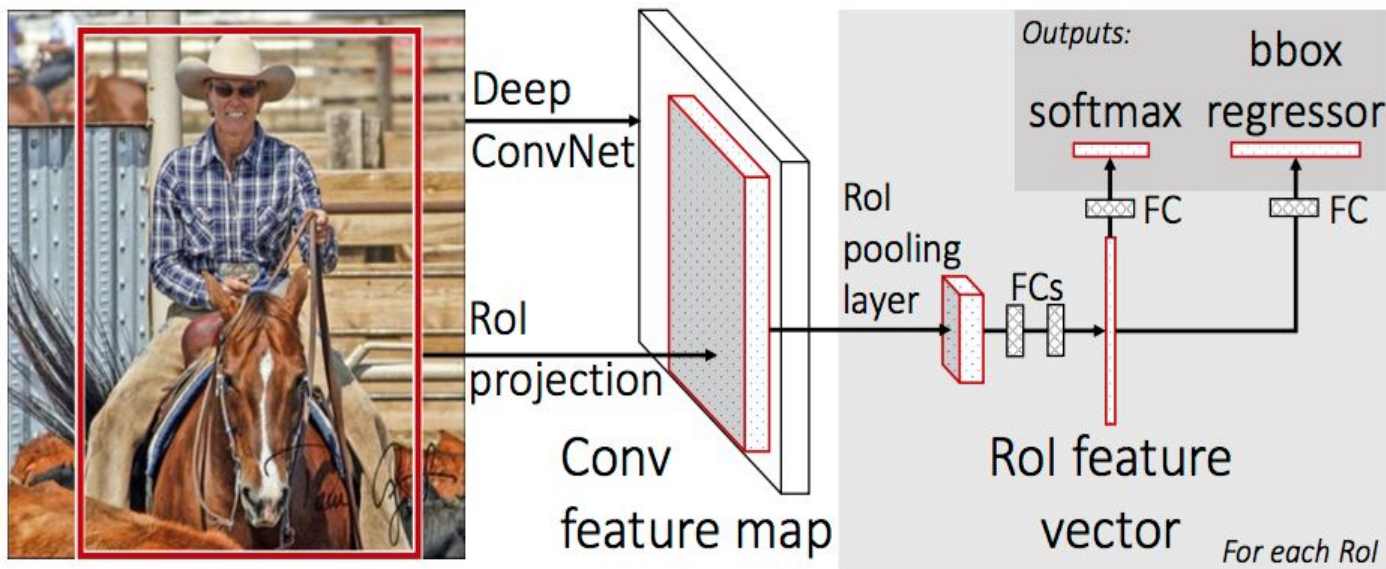




1. Region Proposal: Se genereaza si se extrag regiuni cu o probabilitate mare de a contine obiecte (agnostic – independente de categorii)
2. Feature Extractor: Se extrag trasaturile pentru fiecare regiune folosind o retea convolutionala
3. Clasificator: clasifica feature-urile si obtinem output-ul

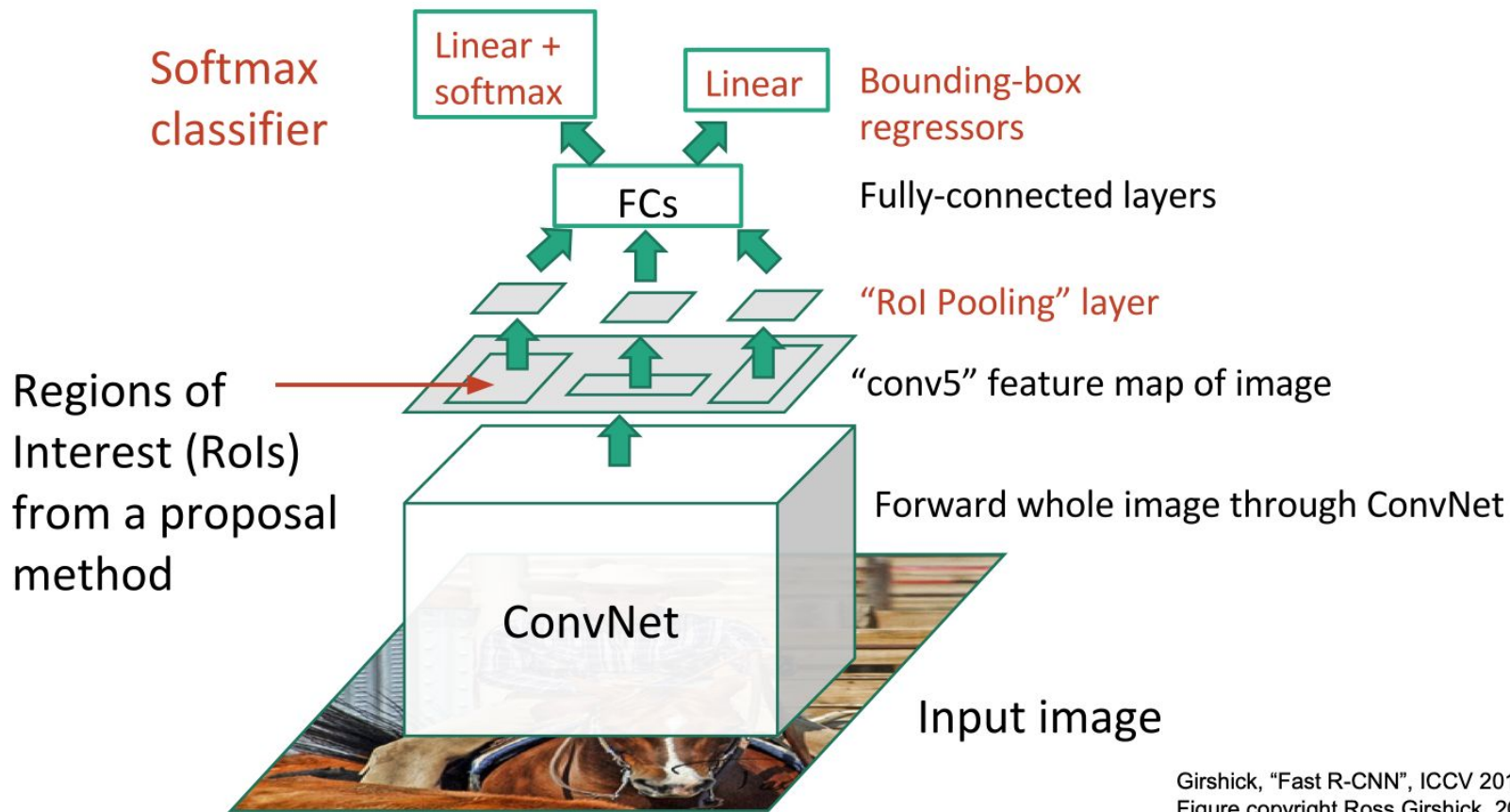
# Fast R-CNN

- R-CNN s-a dovedit a fi foarte incet (~2000 de regiuni per imagine sunt evaluate de CNN), iar antrenarea costisitoare d.p.d.v al timpului si spatiului.
- Fast R-CNN este versiunea imbunatatita si mai rapida a R-CNN
- Imaginea trece o singura data prin feature extractor



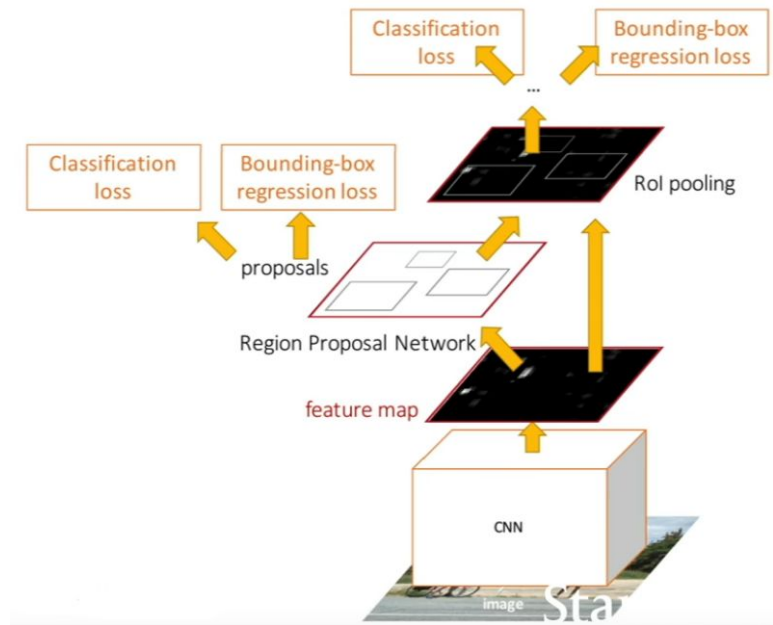
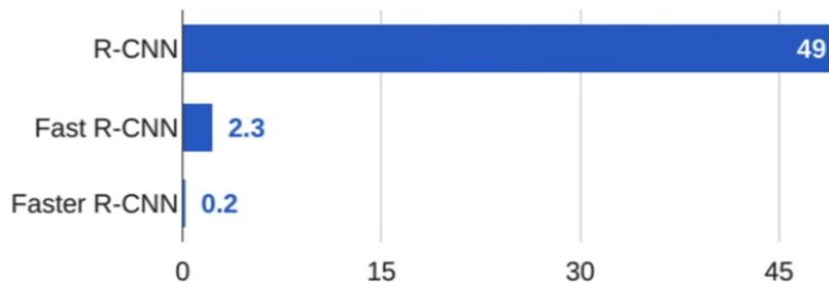


# Fast R-CNN



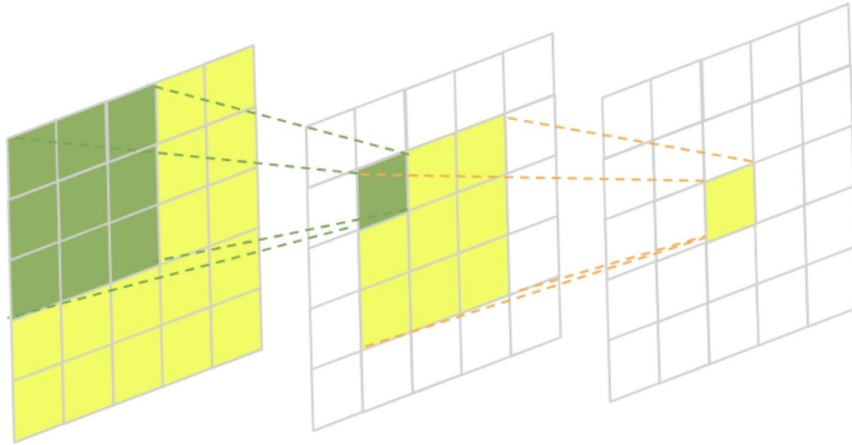
**Problema:** Selective Search dureaza prea mult

**Solutie:** Selective Search este inlocuit de o retea care invata sa propuna regiuni, Region Proposal Network (**RPN**)





# CNN - Receptive Field



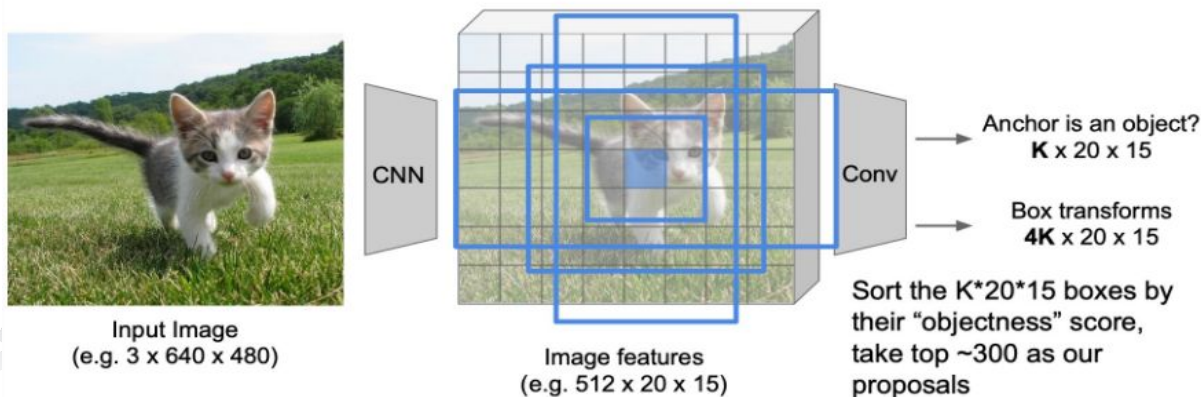
## First Stage (Region Proposal Network - RPN)

- Pentru fiecare imagine din dataset:
  - Backbone -> imaginea este trecuta prin rețeaua convolutionala si se obtine un feature map
  - Region Proposal Network (RPN)
- RPN:
  - Se genereaza ancore (regiuni) la locatii diferite, cu diverse marimi si aspect ratio-uri
  - Acelasi set de ancore este folosit pentru intreg dataset-ul; reprezinta un prior despre locatiile cel mai des intalnite ale obiectelor
  - Pentru fiecare ancora, clasificam regiunea ca obiect/background si rafinam coordonatele bounding-box ului (2 loss-uri)



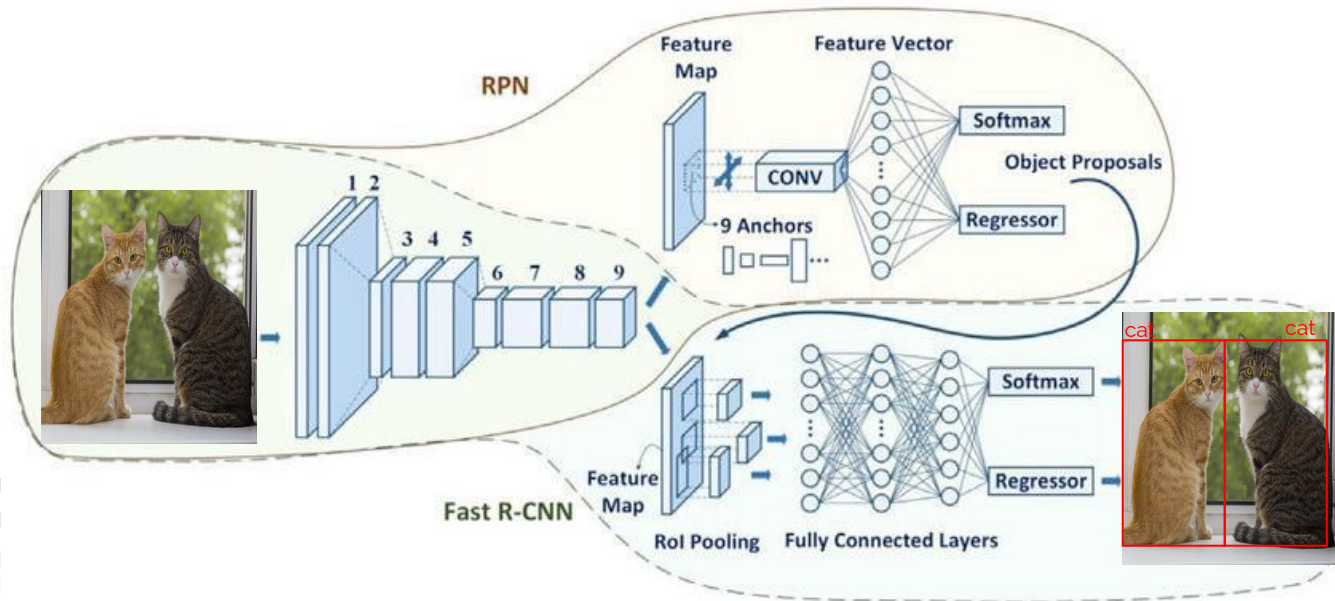
## First Stage (Region Proposal Network - RPN)

- RPN:
  - Rafinarea coordonatelor implica calcularea offset-urilor ( $\Delta x$ ,  $\Delta y$ ,  $\Delta w$ ,  $\Delta h$ ) pentru fiecare ancora
  - GT pentru RPN - o ancora e considerata pozitiva, daca are  $iOU > 0.5$  (hiper-parametru) cu cel mai apropiat bounding-box real
  - Se aleg Top 300 (hiper-parametru) ancore (regiuni) sortate descrescator in functie de probabilitatea de a contine un obiect



## Second Stage

- Asemănător cu Fast R-CNN:
  - RoI pooling layer
  - Clasificarea finala (label)
  - Corectia spatiala a bounding box-ului: regresia ne ofera offset-urile pentru ancore



# One Stage vs Two Stage Detector

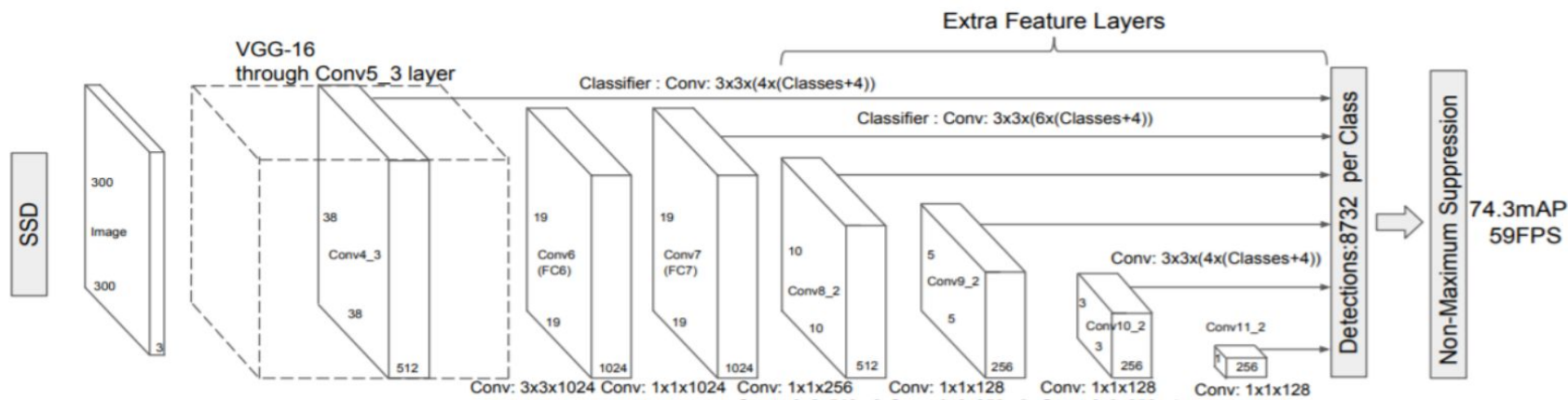
- Faster R-CNN: detectia are loc in doua etape:
  - Prima, in care modelul propune o serie de regiuni de interes folosind RPN
  - A doua, in care clasificatorul doar proceseaza regiunile pe care le primeste din first stage.
- SSD/YOLO: detectia are loc intr-o singura etapa => mai rapid
- One Stage Detectors – mai potrivite pentru aplicatii real-time



# Single-Shot Detector (SSD)

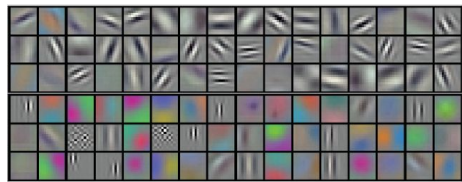
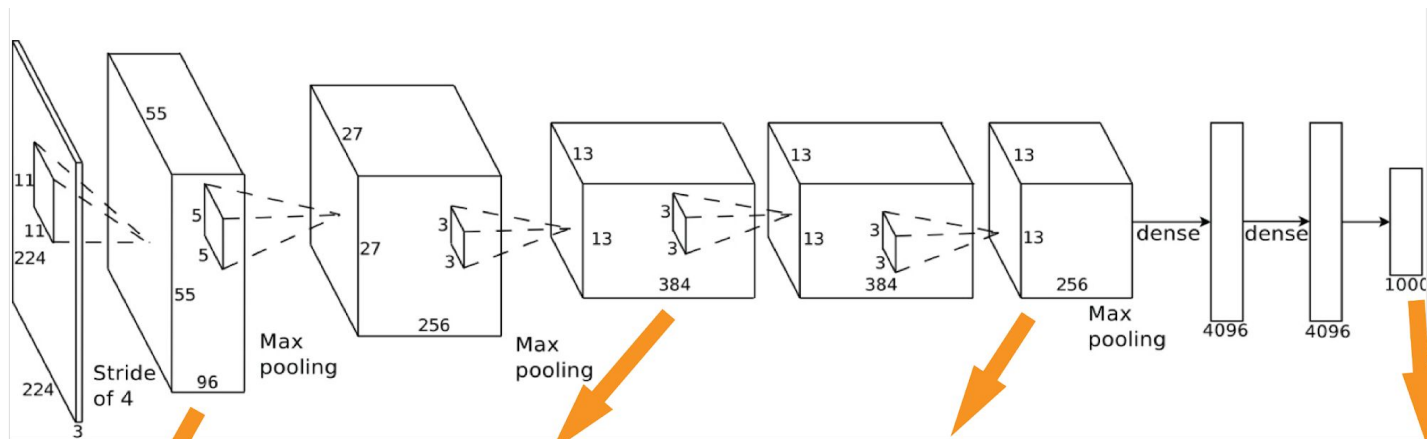
SSD are 2 componente:

- Un **backbone** (o retea pre-antrenata de clasificare folosita ca feature extractor)
- **SSD head**: una sau mai multe convolutii adaugate la backbone
- Receptive field-ul este ideea de baza a SSD: acesta ne ajuta sa detectam obiecte la diferite scale-uri. Primele layer convolutionale ne ajuta sa detectam obiecte mici, iar pe masura ce avansam in retea, putem detecta obiecte din ce in ce mai mari

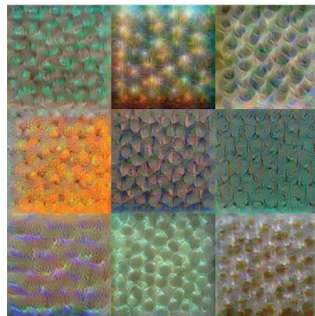




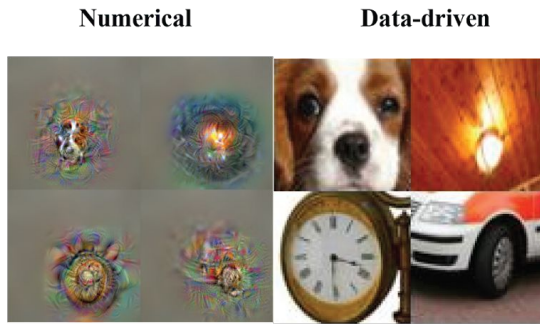
# Single-Shot Detector (SSD)



Conv 1: Edge+Blob



Conv 3: Texture



Conv 5: Object Parts

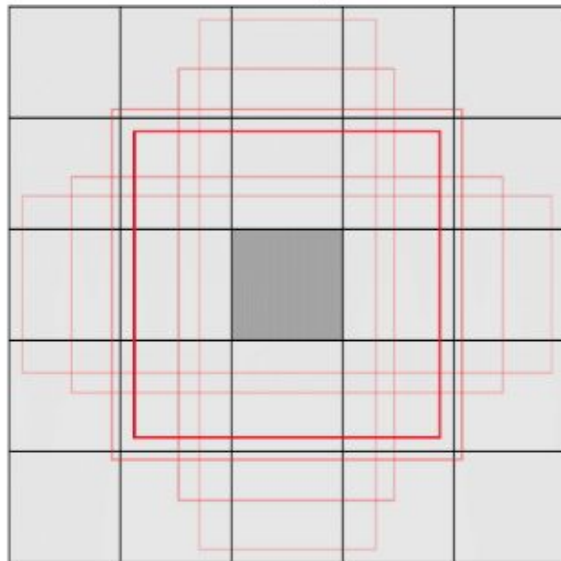


Fc8: Object Classes

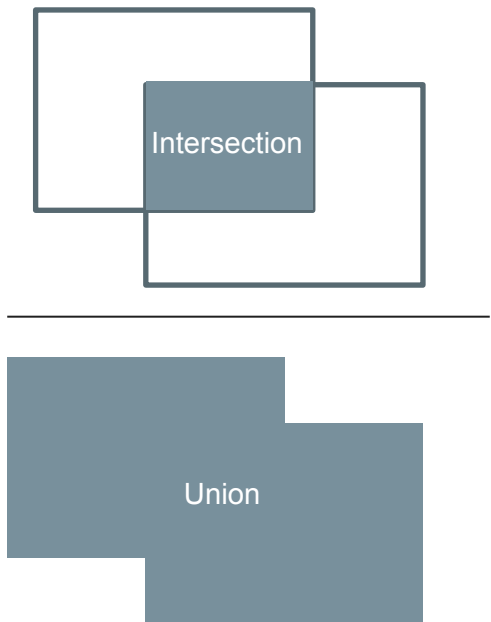
# Single-Shot Detector (SSD)

- Definirea ancorelor:
- Prior-urile (ancorele) vor fi aplicate pe feature map-uri de diverse dimensiuni
- Ancorele sunt diferite pentru fiecare feature map in parte
- Daca o ancora are un scale  $s$ , atunci aria ei este egala cu aria unui patrat cu latura  $s$ .
- La fiecare pozitie din feature map vor exista ancorae cu diferite aspect ratio-uri.

Feature Map From	Feature Map Dimensions	Prior Scale	Aspect Ratios	Number of Priors per Position	Total Number of Priors on this Feature Map
conv4_3	38, 38	0.1	1:1, 2:1, 1:2 + an extra prior	4	5776
conv7	19, 19	0.2	1:1, 2:1, 1:2, 3:1, 1:3 + an extra prior	6	2166
conv8_2	10, 10	0.375	1:1, 2:1, 1:2, 3:1, 1:3 + an extra prior	6	600
conv9_2	5, 5	0.55	1:1, 2:1, 1:2, 3:1, 1:3 + an extra prior	6	150
conv10_2	3, 3	0.725	1:1, 2:1, 1:2 + an extra prior	4	36
conv11_2	1, 1	0.9	1:1, 2:1, 1:2 + an extra prior	4	4
<b>Grand Total</b>	—	—	—	—	<b>8732 priors</b>



# Intersection over union (IoU)

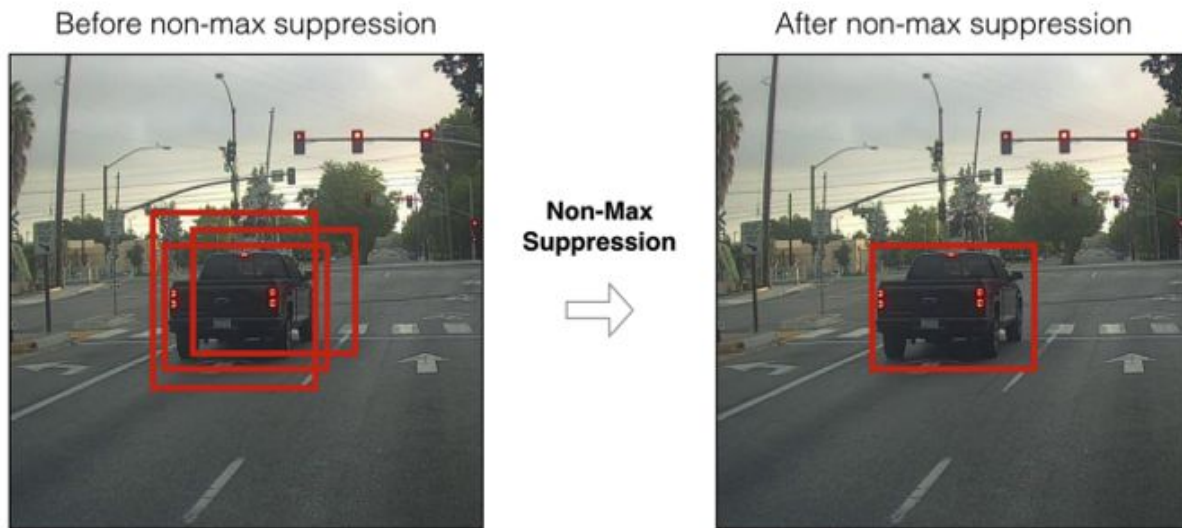


$$\text{IoU} = \frac{\text{Intersection}}{\text{Union}} = \frac{\text{Area of overlap}}{\text{Area of union}}$$

- $\text{IoU} = 0 \Rightarrow$  nu exista suprapunere intre box-uri
- $\text{IoU} = 1 \Rightarrow$  intersectia box-urilor este egala cu reuniunea lor, deci acestea se suprapun perfect

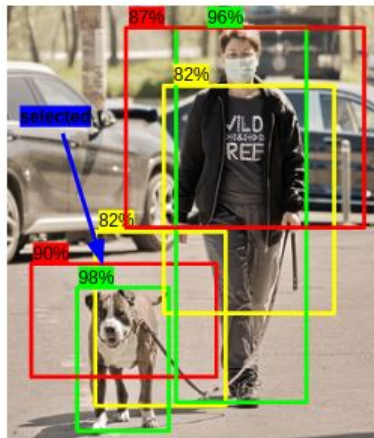
# NMS

- Input: o lista de box-uri propuse  $B$ , confidence score-ul  $S$  corespunzator fiecaruia si pragul de suprapunere (overlap threshold)  $N$ .
- Output: O lista filtrata de box-uri propuse  $D$ .
- Scop: elimina predictiile redundante, pastrand detectia cu cel mai mare scor.

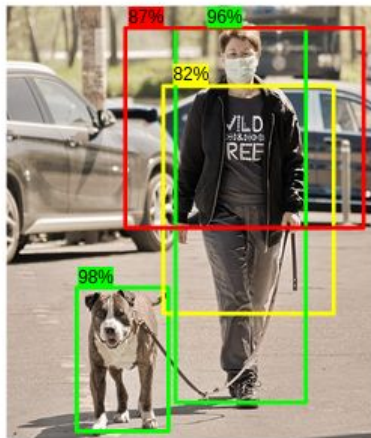


# NMS

1. Selecteaza box-ul cu cel mai mare confidence score
2. Apoi, compara suprapunerea acestuia cu alte box-uri (IoU)
3. Elimina bounding box-urile cu suprapunerea (IoU)  $>50\%$
4. Apoi, mergi la urmatorul box cu cel mai mare confidence score
5. Repeta pasii 2-4.



Step 1: Selecting Bounding box with highest score



Step 3: Delete Bounding box with high overlap



Step 5: Final Output

# Evaluarea Retelelor de Detectie

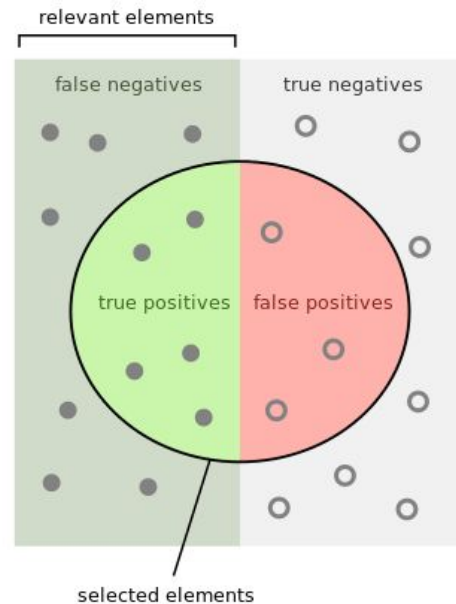
- Precision

$$\text{Precision} = \frac{tp}{tp + fp}$$

- Recall

$$\text{Recall} = \frac{tp}{tp + fn}$$

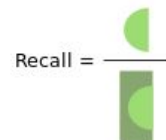
Metricile sunt calculate pe boxuri. Consideram un box ca fiind *true positive* daca are IoU cu boxul de Ground Truth peste o anumita valoare, altfel este *false positive*. *False negative* sunt boxurile din Ground Truth pentru care modelul nu a prezis nimic.



How many selected items are relevant?

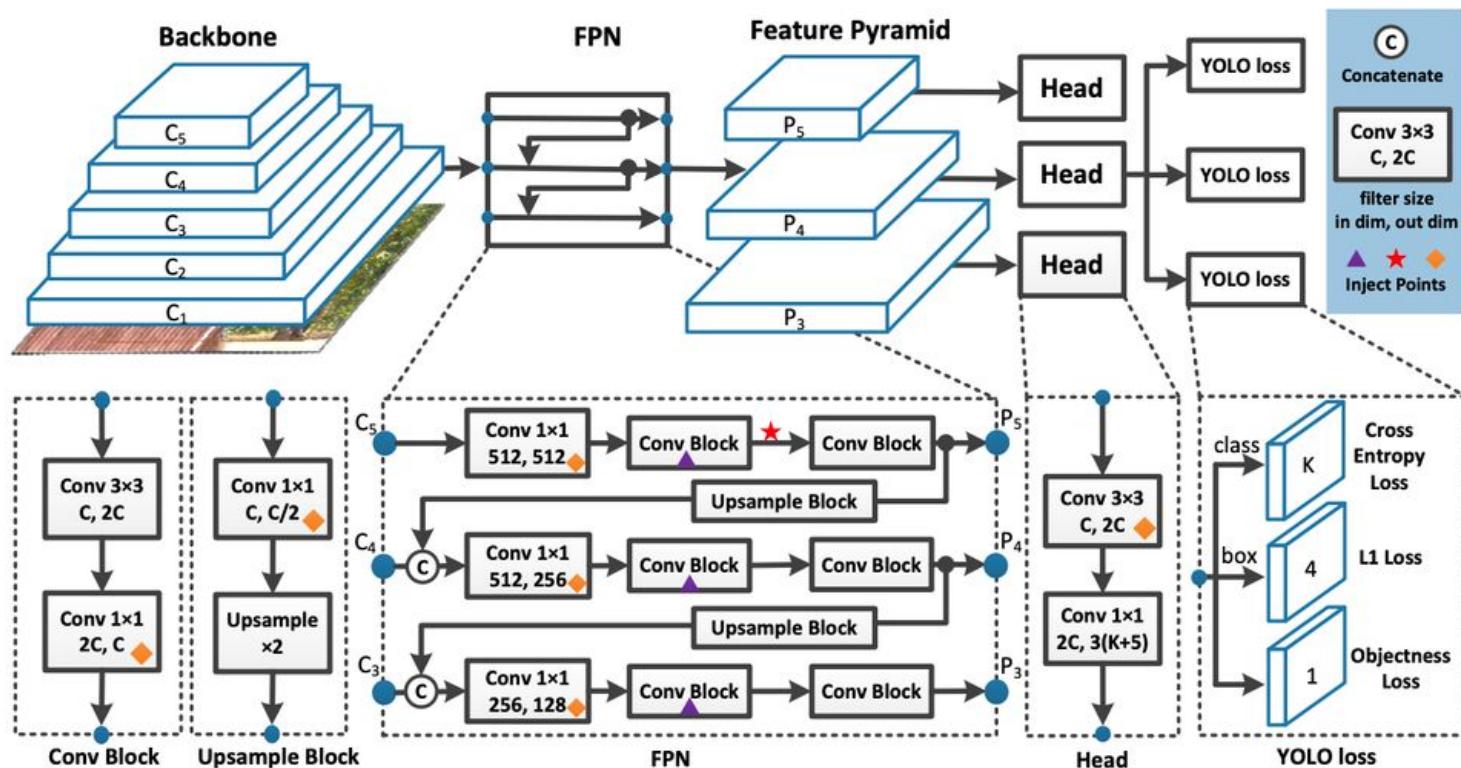


How many relevant items are selected?





# YOLO - You Only Look Once (2020)



# Key Developments in YOLO Evolution

## Early Improvements (YOLOv1 to YOLOv3):

- Introduced grid division and anchor boxes
- Enhanced backbone with Darknet-53
- Multi-scale predictions using Feature Pyramid Network (FPN)

## Efficiency and Speed (YOLOv4 to YOLOv6):

- CSPDarknet53 and Spatial Pyramid Pooling (SPP) for efficiency
- Simplification and speed optimization
- Anchor-free training

## Recent Advances (YOLOv7 to YOLOv12):

- Unified architectures for multiple tasks
- Transformer integration and Non-Maximum Suppression (NMS) elimination
- Attention mechanisms for efficiency and accuracy

# YOLO Evolution

