

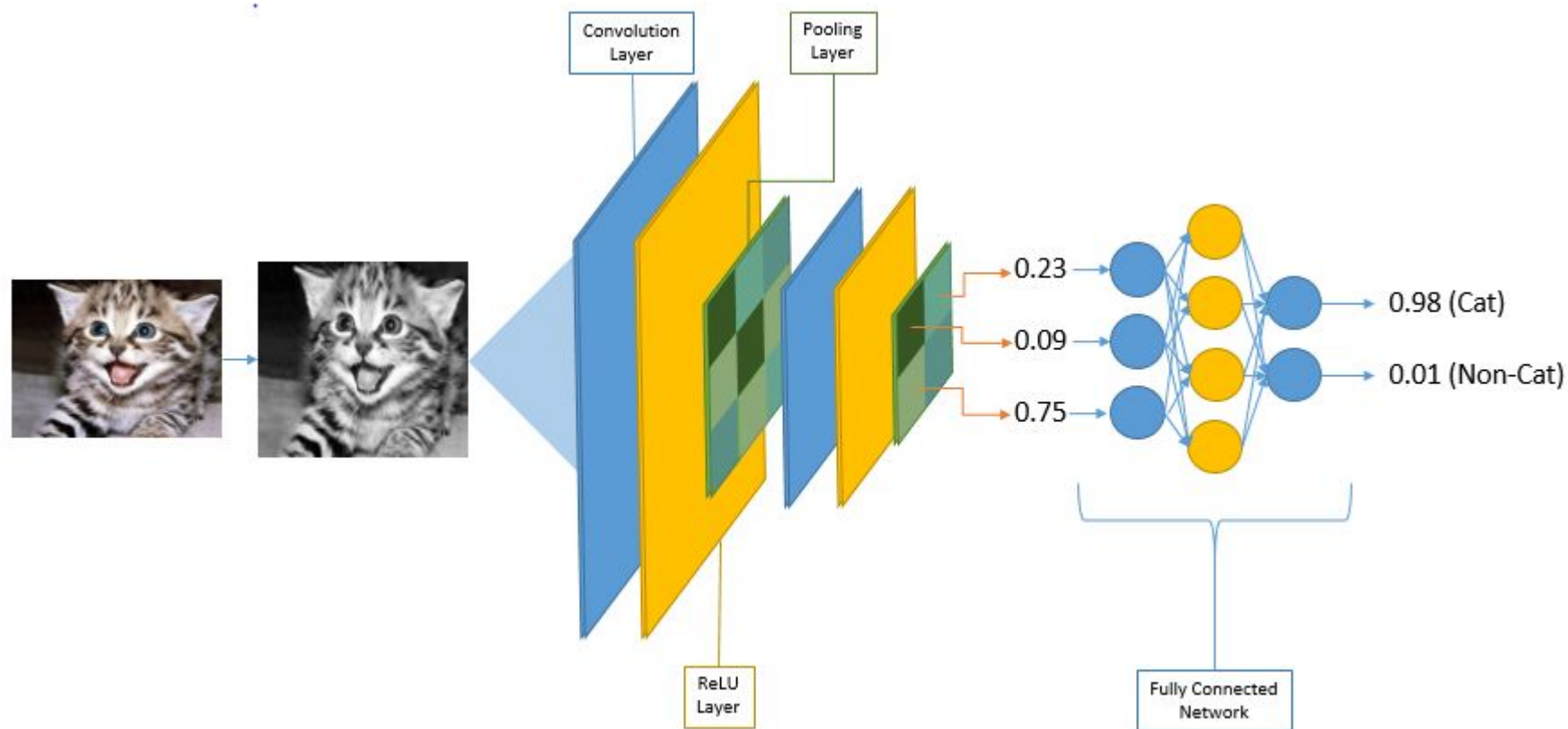
Object Detection Architectures

By Yassine Kriouile (PHD in Artificial supervised by Corinne
Ancourt)

Outline

- Introduction:
 - Classification Task reminder
 - Definition of object detection
- Two stage detection
- One stage detection
- Segmentation
- Metrics

Classification Task



Exercise

Create a classifier which classifies images to with_mask/without_mask

- Input images size: 180*180
- 1 Conv: 16 filters, 3*3 kernel size, strides (1,1), activation relu
- 2 Max pooling: size (2,2)
- 3 Conv: 32 filters, 3*3 kernel size, strides (2,2), activation relu
- 4 Max pooling: size (2,2)
- 5 Conv: 64 filters, 3*3 kernel size, strides (1,1), activation relu
- 6 Max pooling: size (2,2)
- 7 Dense: 128 units, activation relu
- 8 Classification layer

Link to colab:

https://colab.research.google.com/drive/1j9tqu8kgMrM0NZfRByaiwSlgXf40ko_r?usp=sharing

Code examples

- Convolution layer: **tf.keras.layers.Conv2D(*number of filters*, *kernel size*, **strides = *strides***, **padding = 'same'**, **activation = 'relu'**)**
- Pooling layer : **tf.keras.layers.MaxPooling2D()**
- Flatten layer: **tf.keras.layers.Flatten()**
- Dense layer: **tf.keras.layers.Dense(*number of units*, **activation = 'relu'**)**

Use `model.summary()` to describe the model

Explain the output shapes and number of parameters

How to calculate number of parameters for conv layer

Number of parameters =

$$(\text{filters \#} * \text{kernel size}^2 * \text{input channels \#}) + \text{filters \#}$$

Types of object recognition

Classification



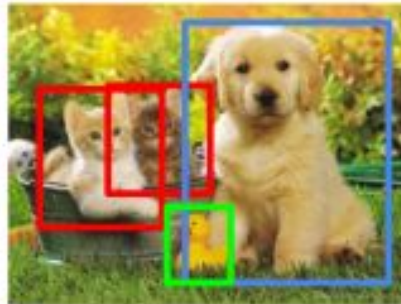
CAT

**Classification
+ Localization**



CAT

Object Detection



CAT, DOG, DUCK

**Instance
Segmentation**



CAT, DOG, DUCK

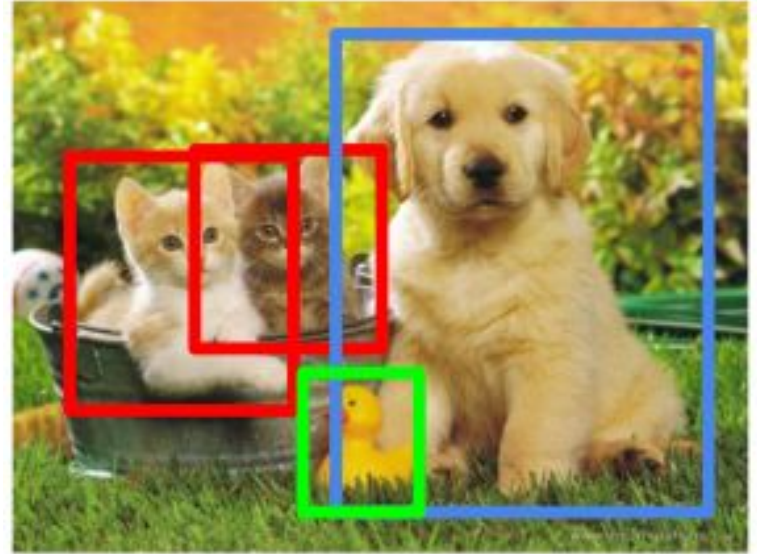
Single object

Multiple objects

Definition of object detection

The task of assigning a **label** and a **bounding box** to all objects in the image, without knowing the number of objects in advance.

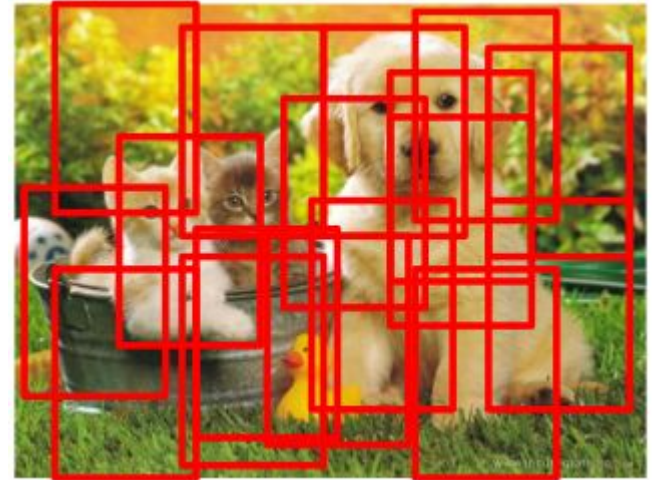
Suggest a method to solve this problem based on classification.



Classify all possible boxes

- Generate all possibilities
- Classify each one of them

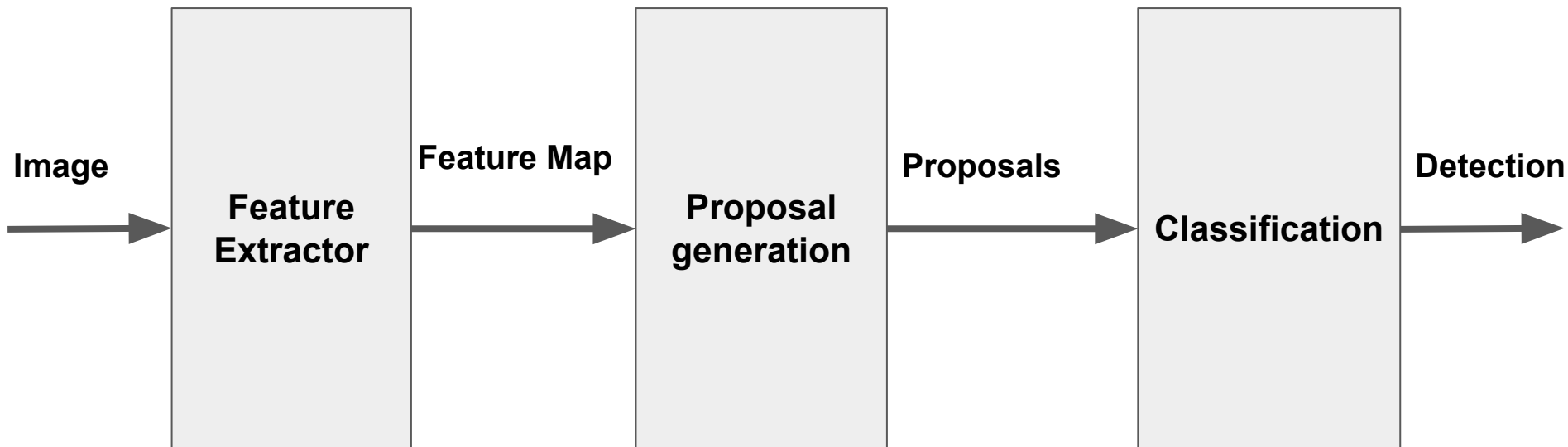
A very large number of possibilities (position, size, ratio). Do you think it is feasible?



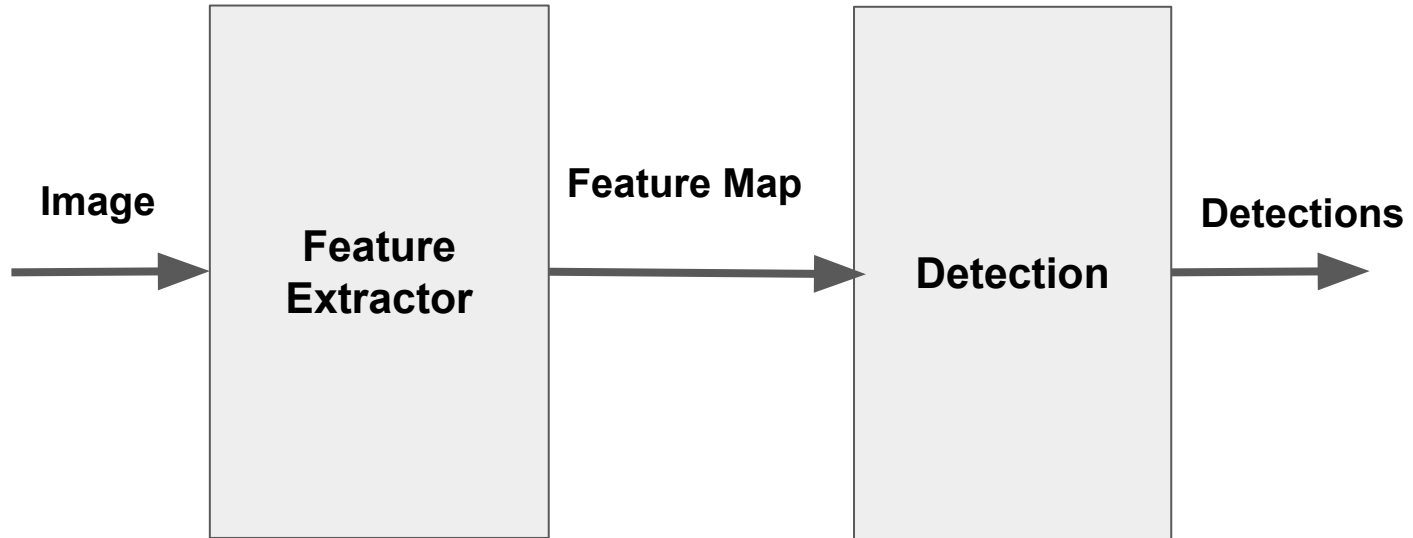
Types of object detection

- Two stage object detection: generate proposals + classification
- One stage object detection: generate objects directly

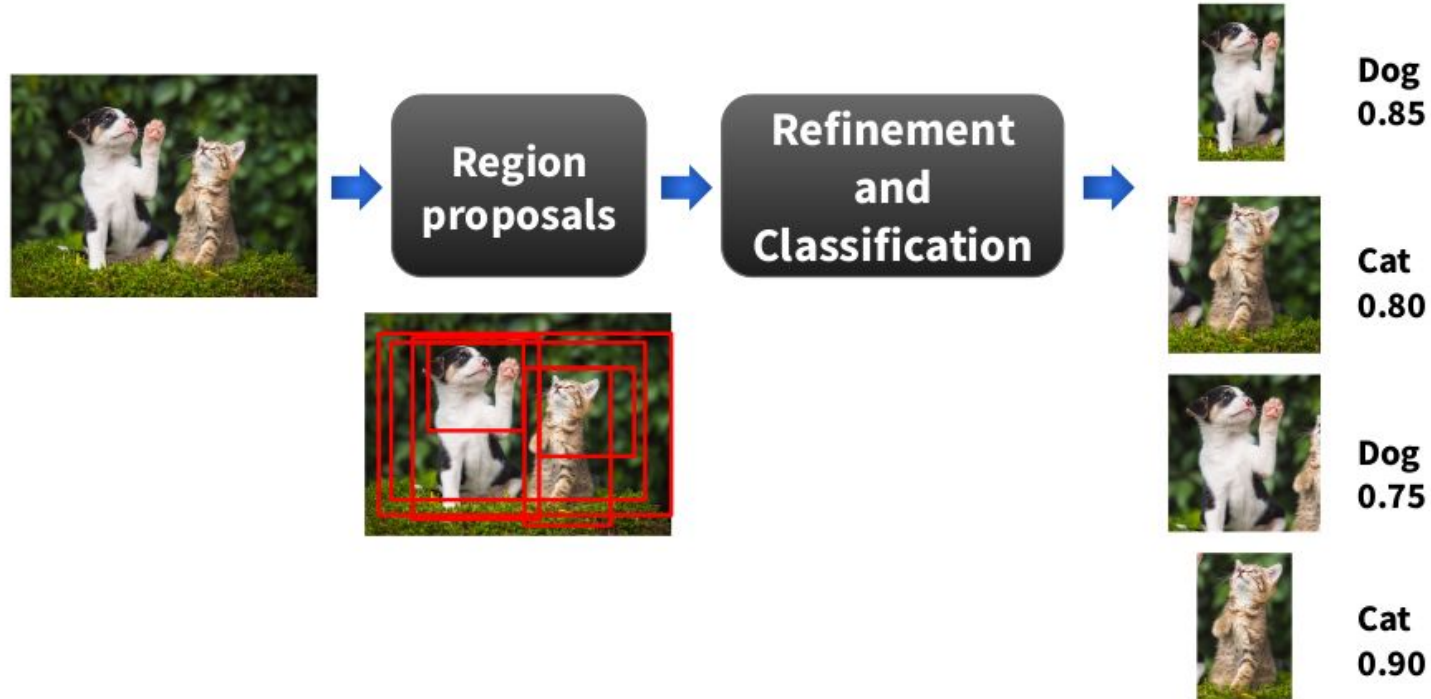
Two stage general architecture



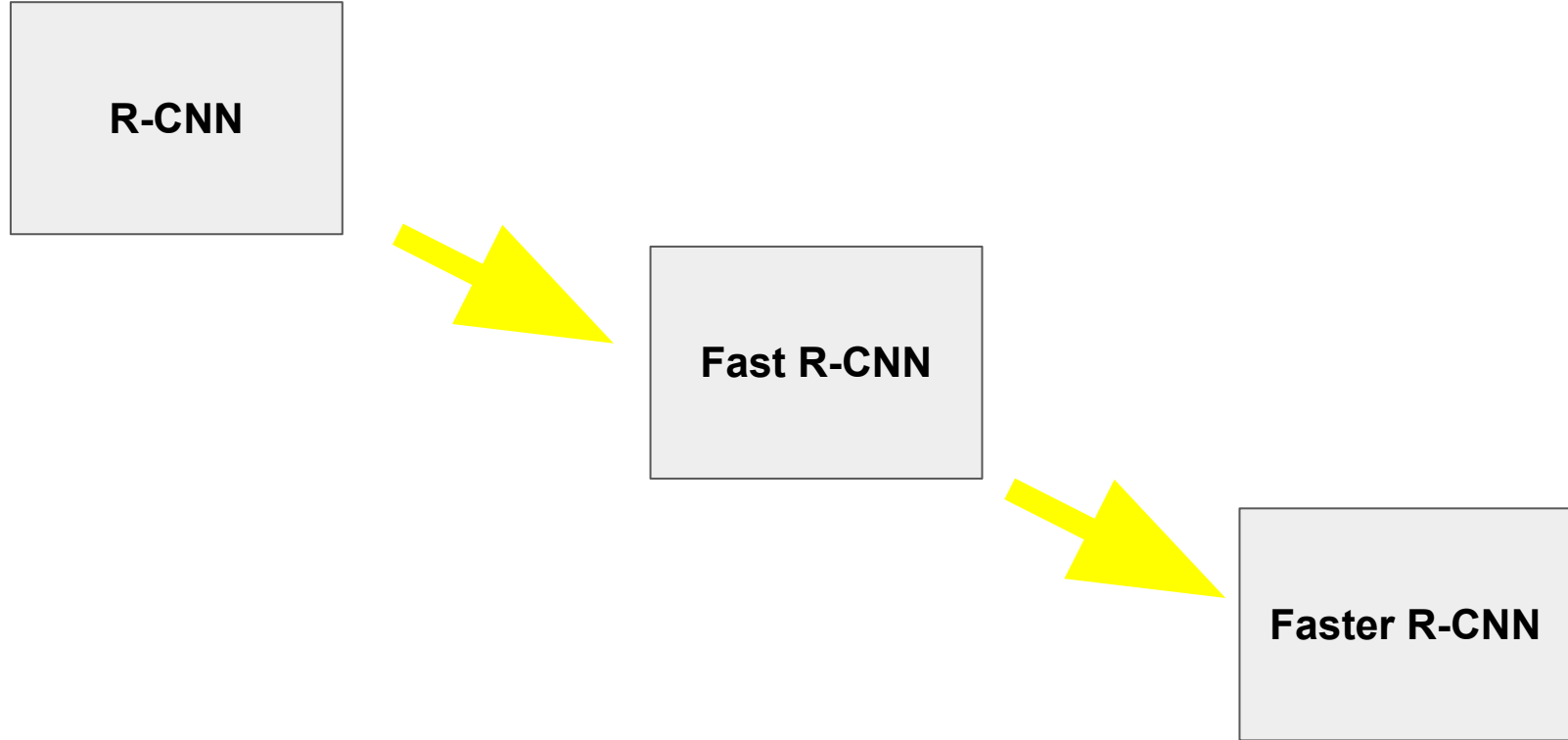
One stage general architecture



Two stage detection process

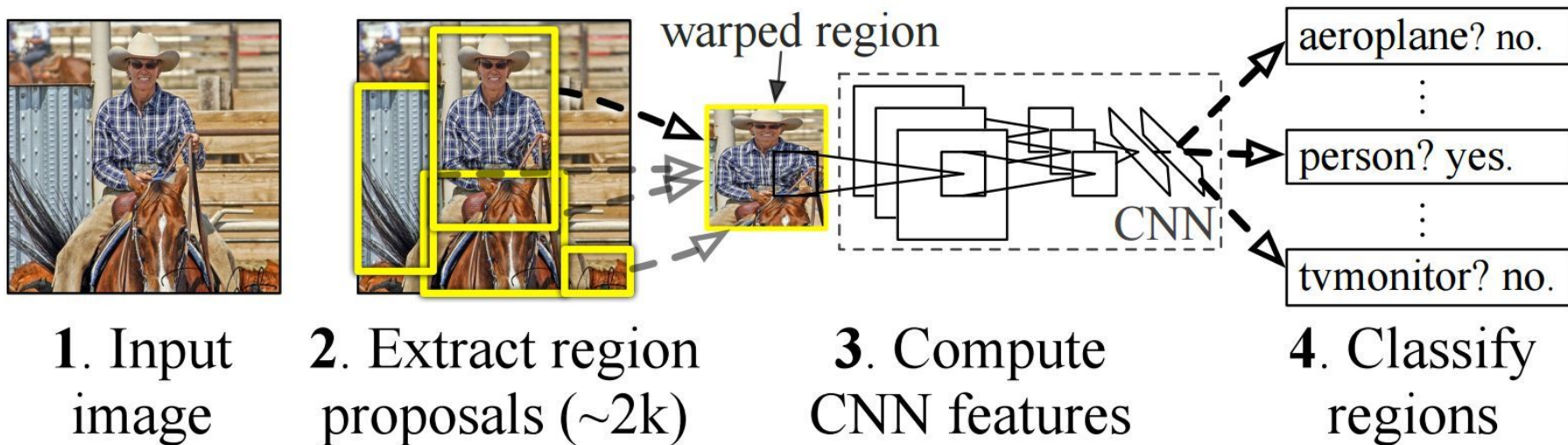


Two stage object detection development



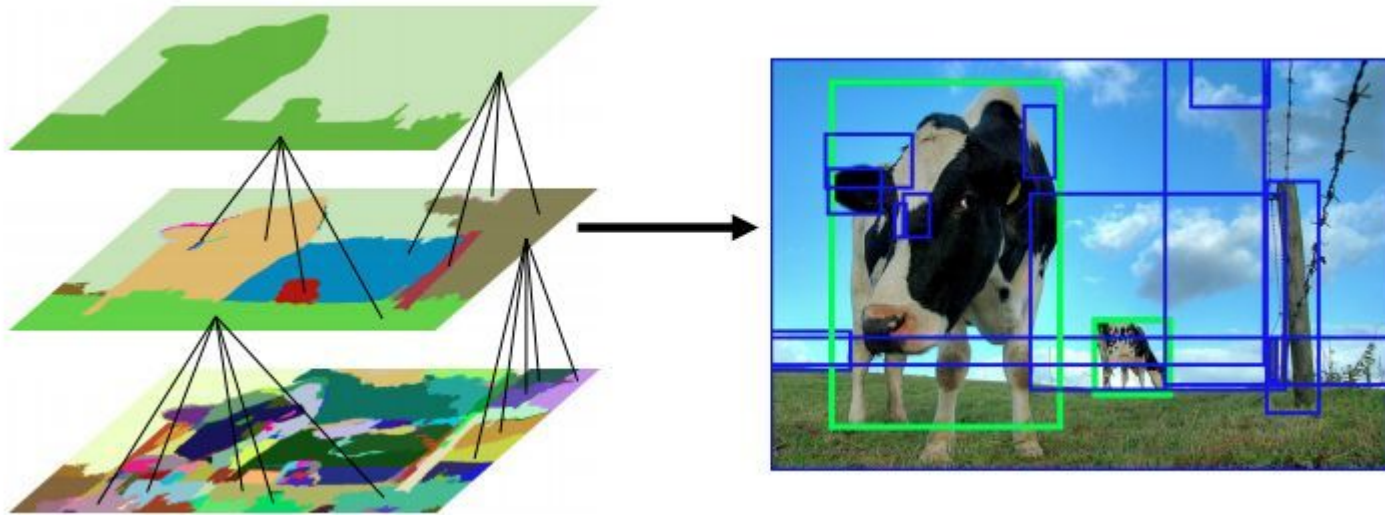
R-CNN

- Selective Search algorithm to generate proposals
- CNN to extract features
- SVM to classify



Selective search

Iterative algorithm to generate proposals from images

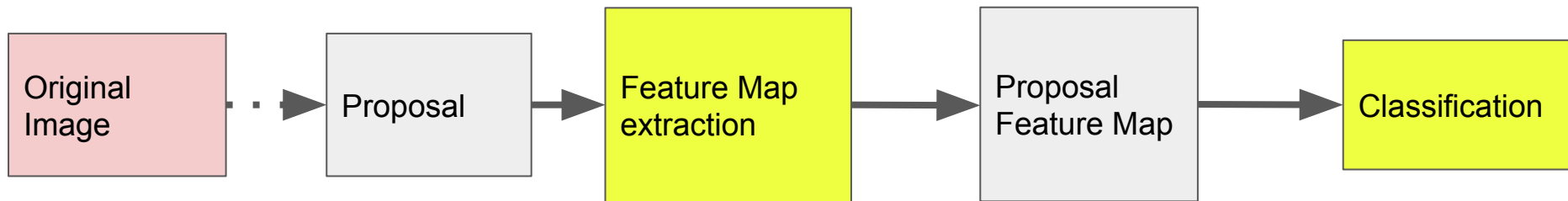


R-CNN limits

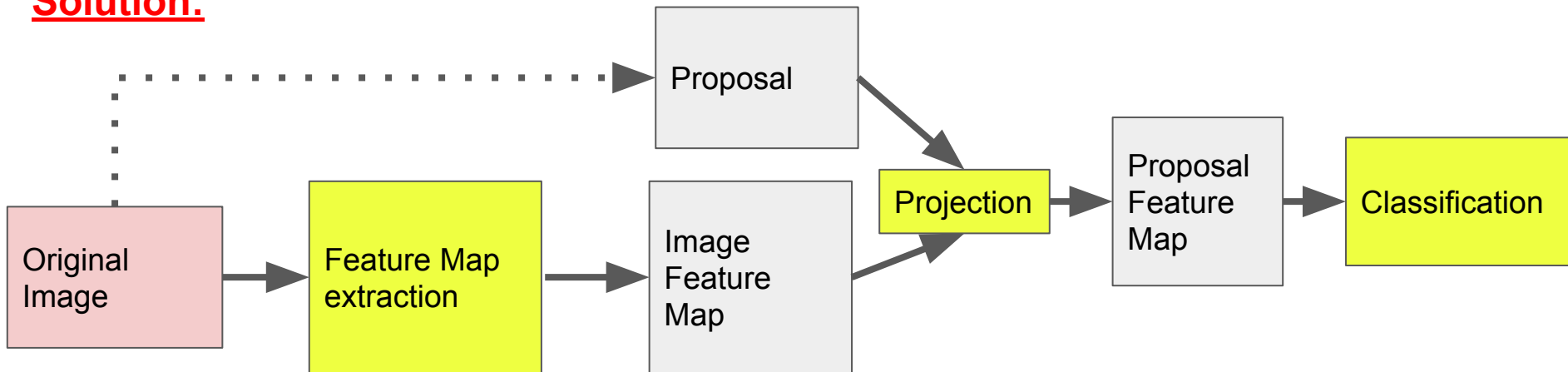
- Very slow because for each proposal features must be extracted
- Proposals are warped to respect the CNN input size constraint
- Two training phase: CNN and SVM
- Selective Search: static proposal generation

How to avoid feature extraction for each proposal?

How to avoid feature extraction for each proposal?

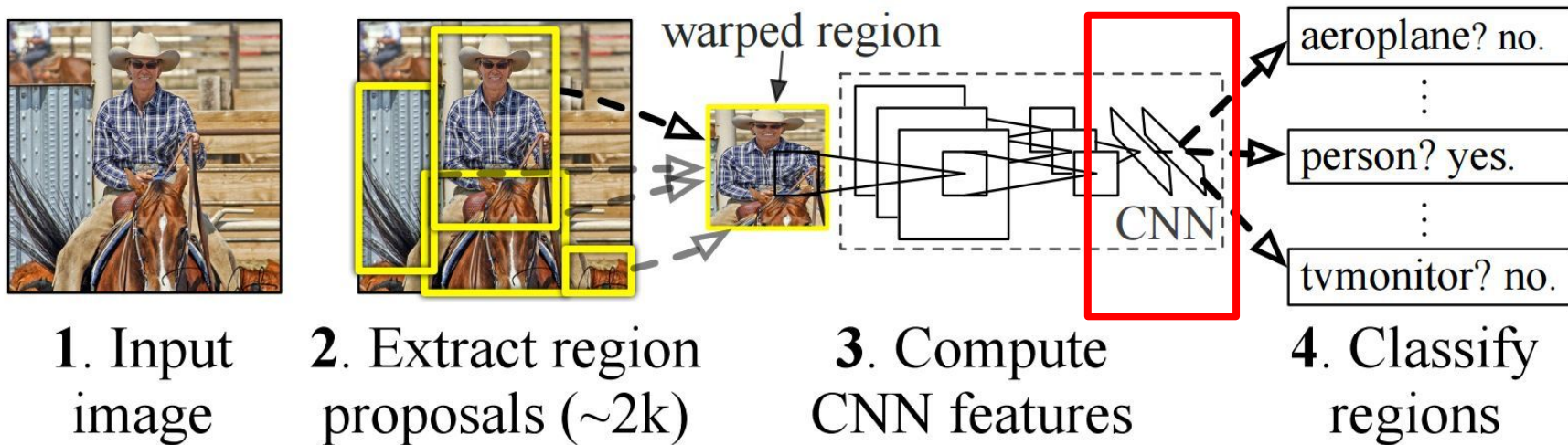


Solution:



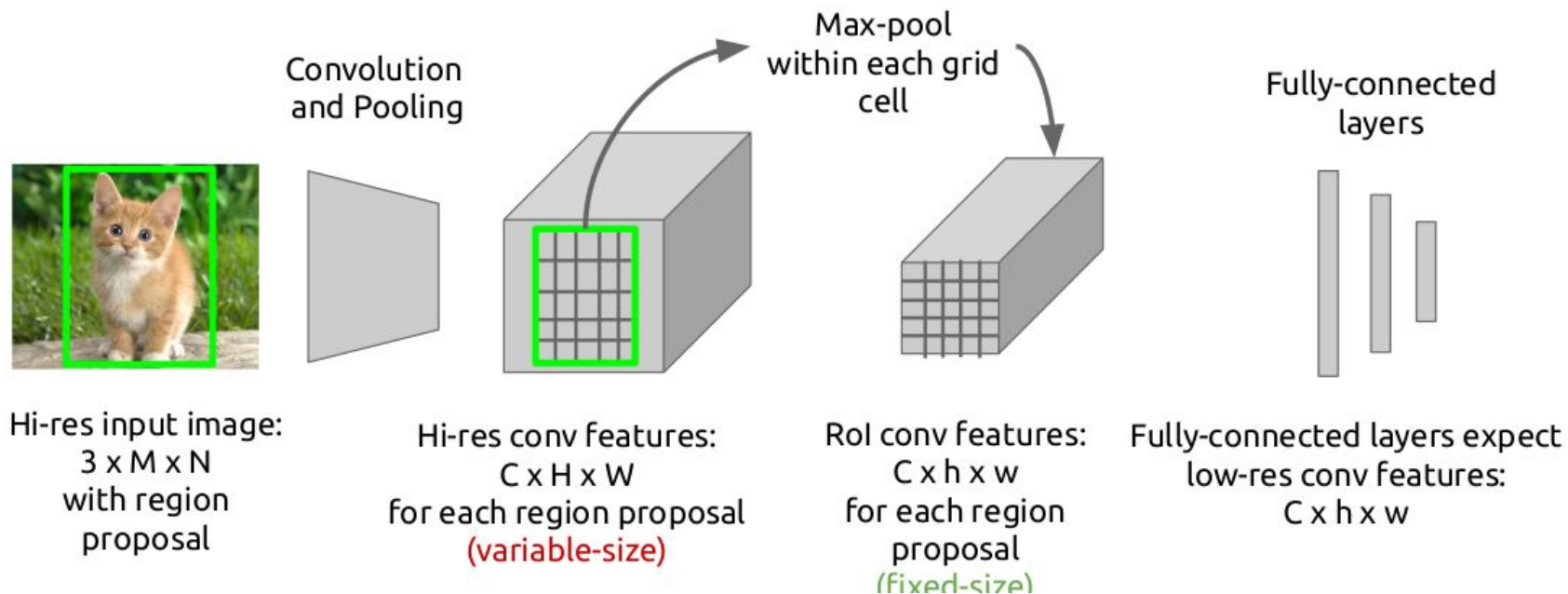
Warp problem:

- The convolutional layers are independant to input size
 - But it is not the case for fully connected layers
- > The proposal is warped



How to fix it?

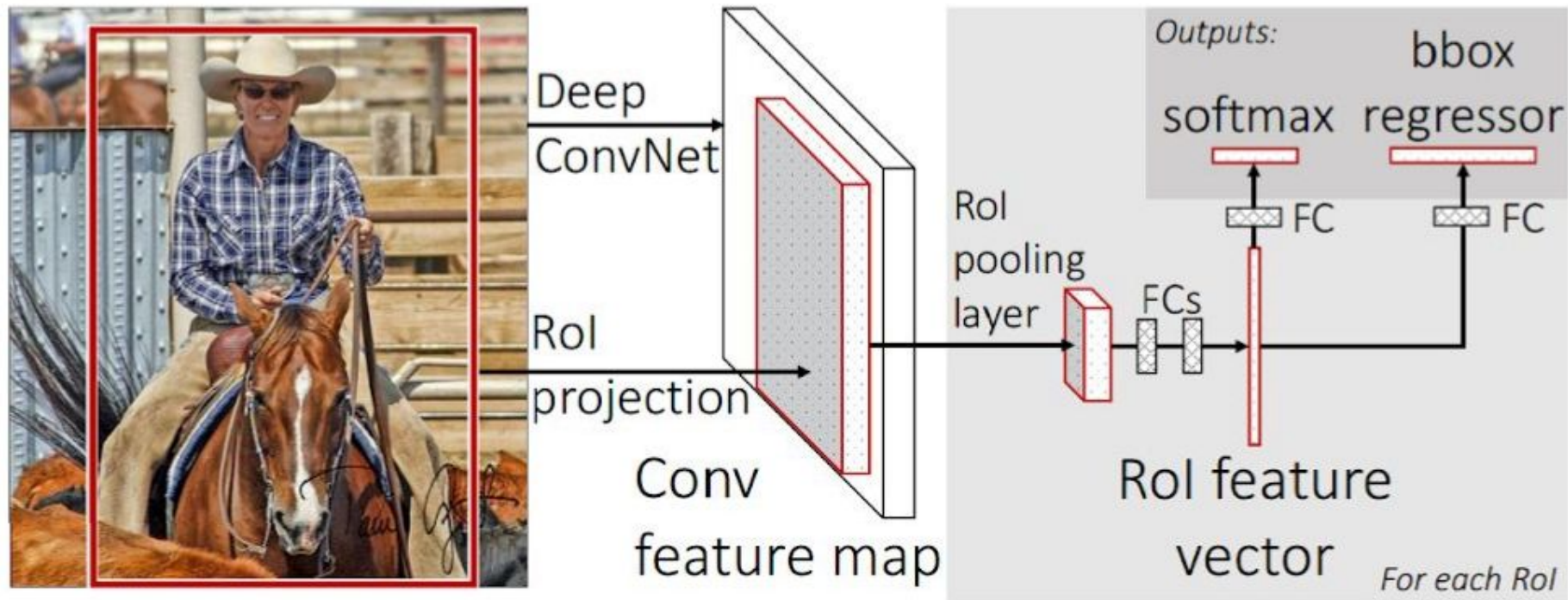
ROI pooling: region of interest pooling



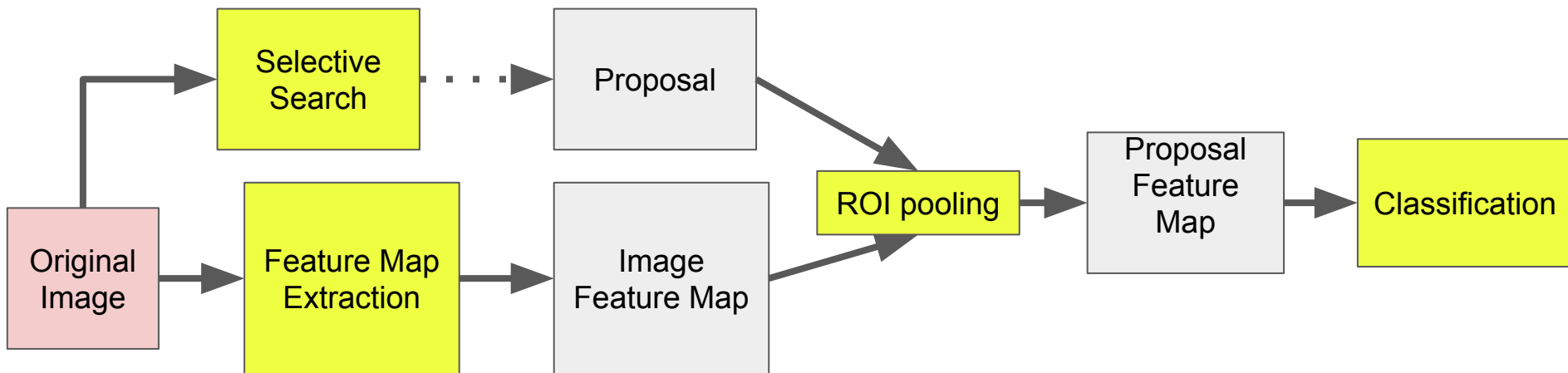
Remove SVM

- Direct Training
- More coherent (in RCNN, learned features are disconnected from SVM)
- Use softmax layer instead for classification

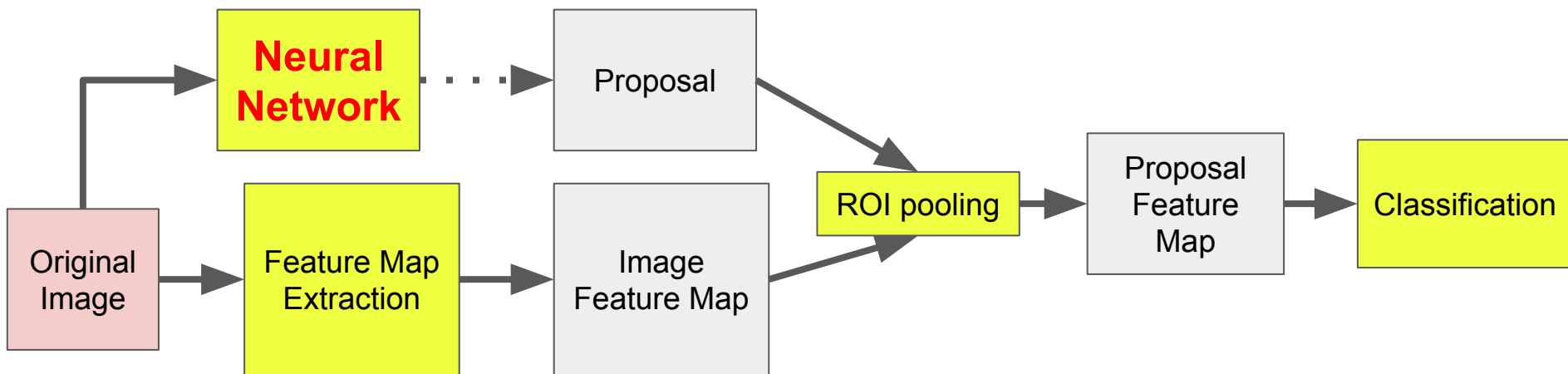
We obtain Fast-RCNN



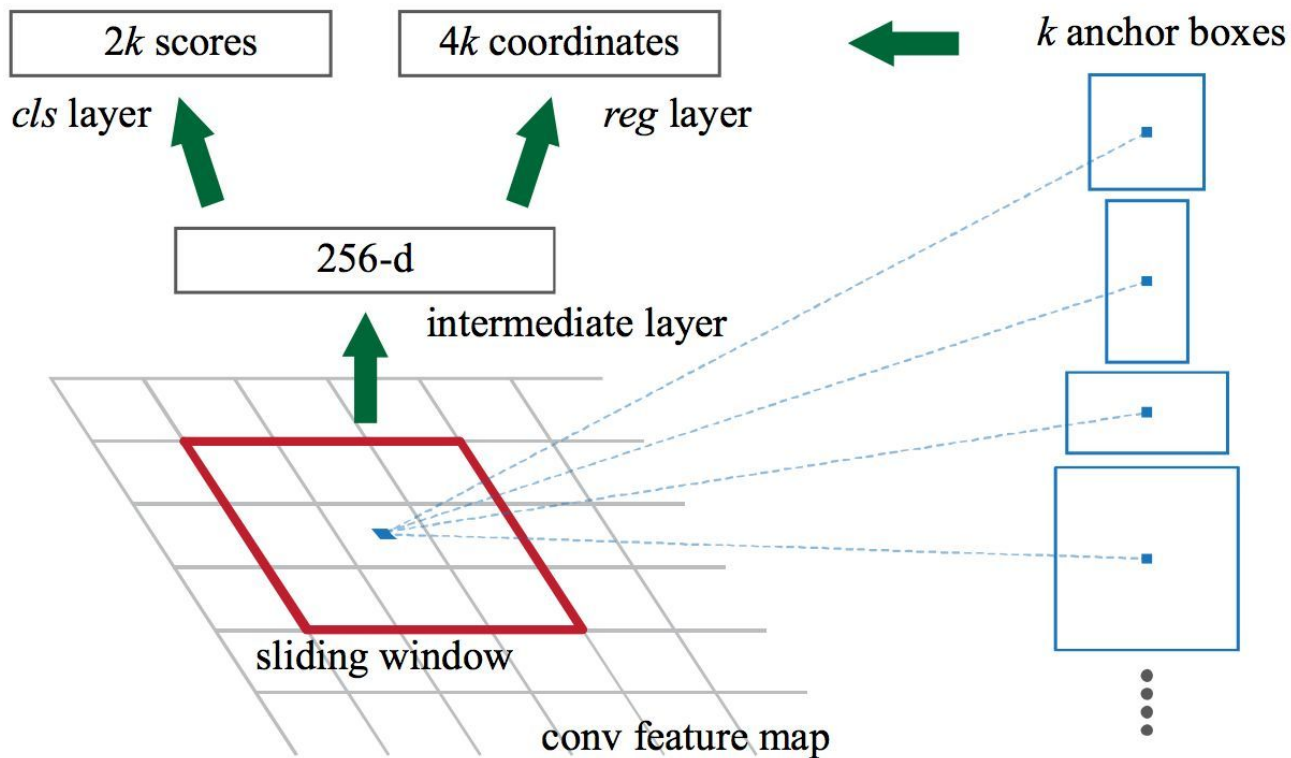
Fast R-CNN



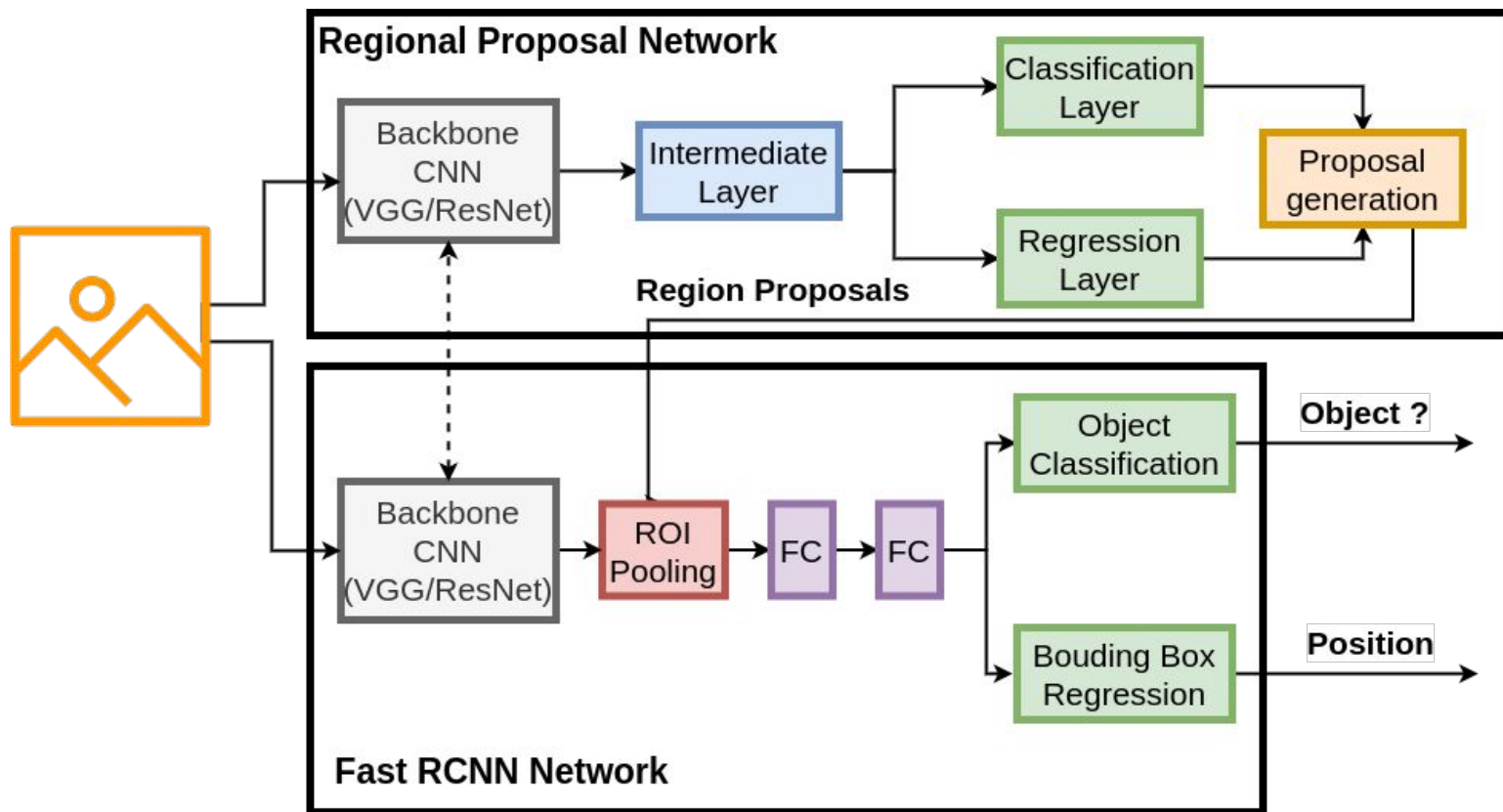
Still Selective Search Limit



Region Proposal Network

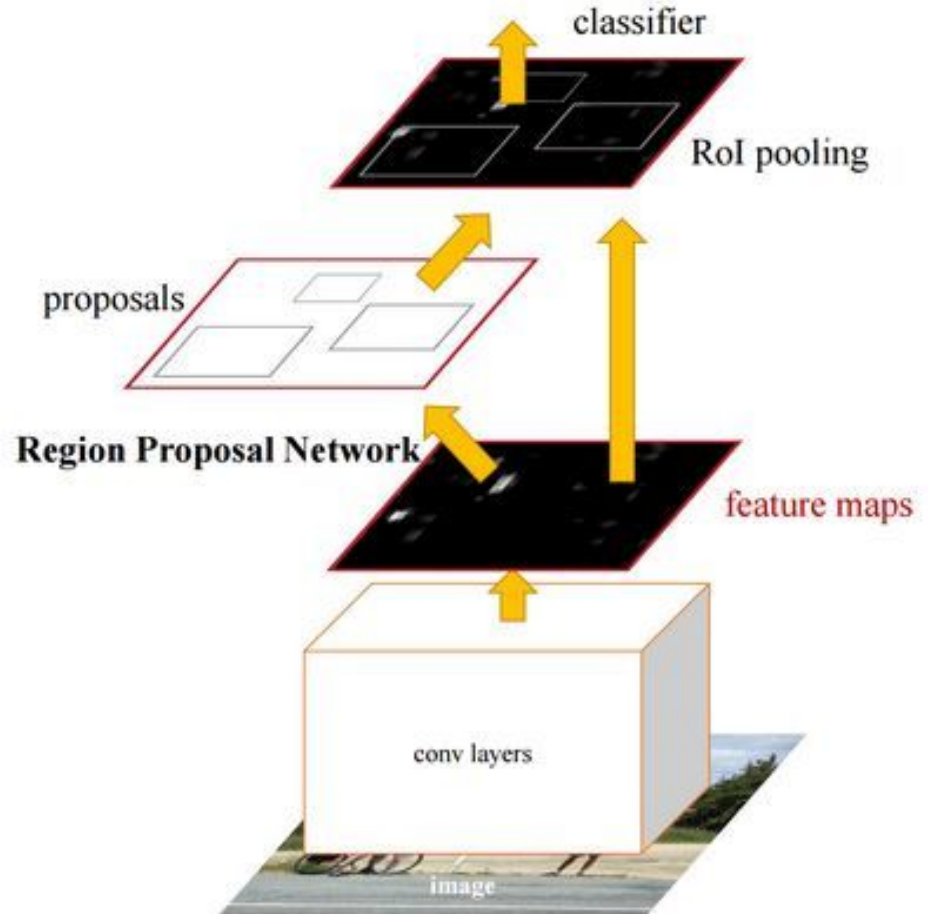


Faster RCNN



Faster-RCNN general architecture

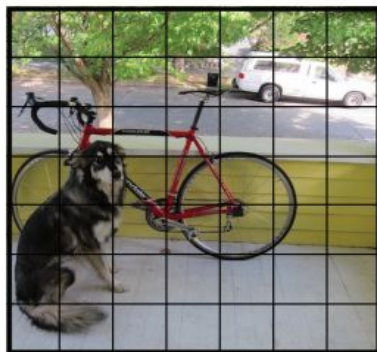
One backbone for extracting features



One stage detection intuition

- RPN generates bounding boxes through all feature map locations.
- It is possible thanks to parallel structure of RPN compared to Selective Search.
- RPN structure can be sufficient for detecting objects directly.

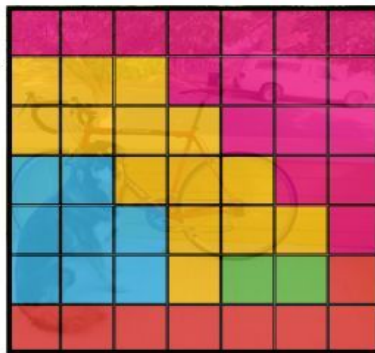
YOLO: you look only once



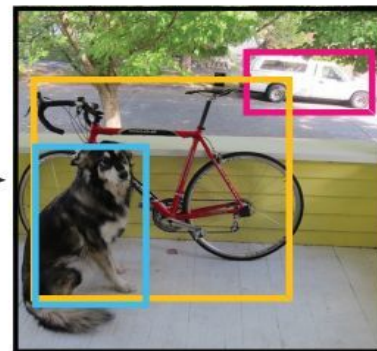
S x S grid on input



**Bounding boxes +
confidence scores**

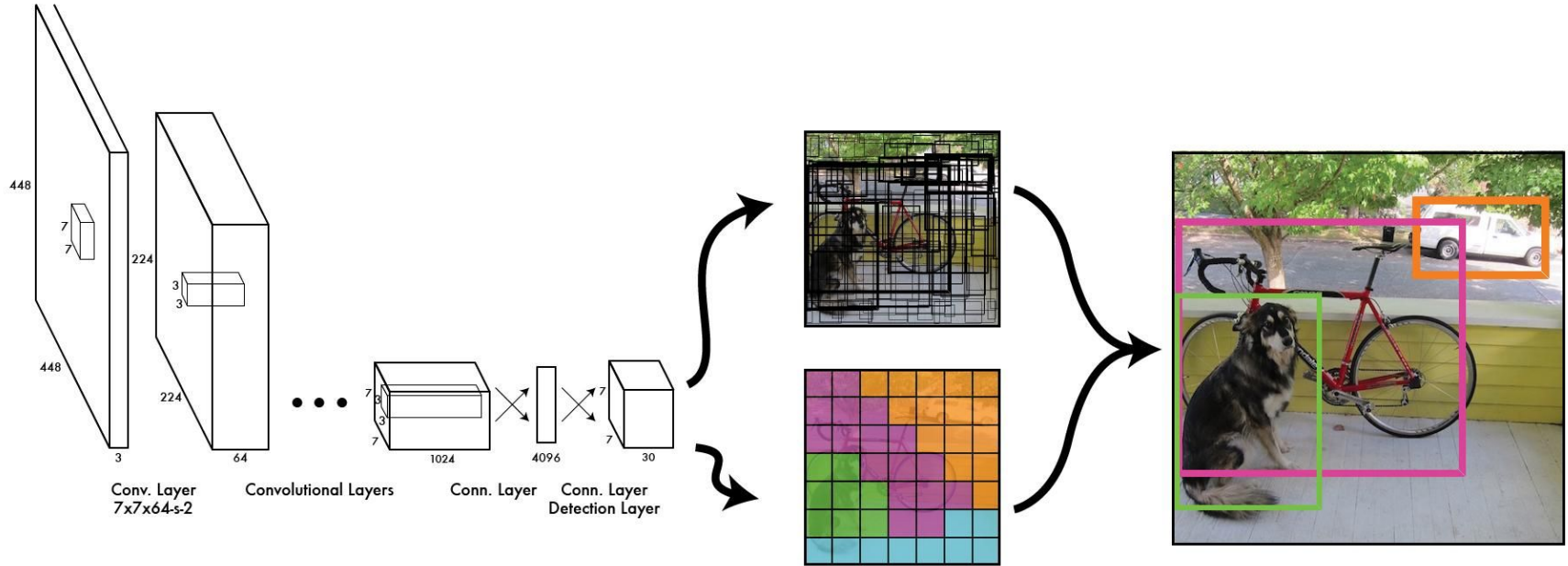


Class probability map



Final detections

YOLO general architecture



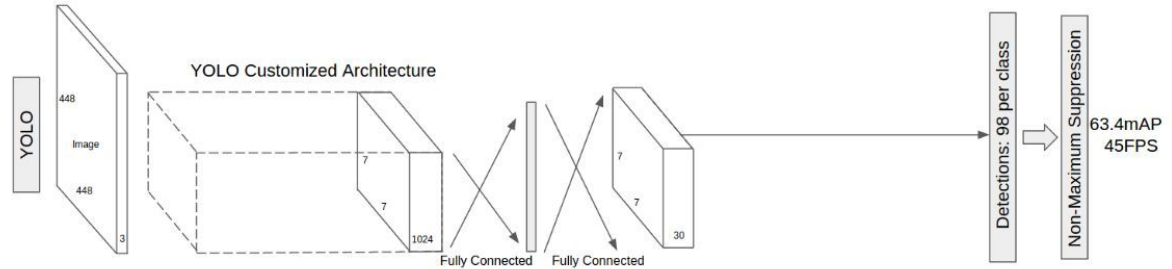
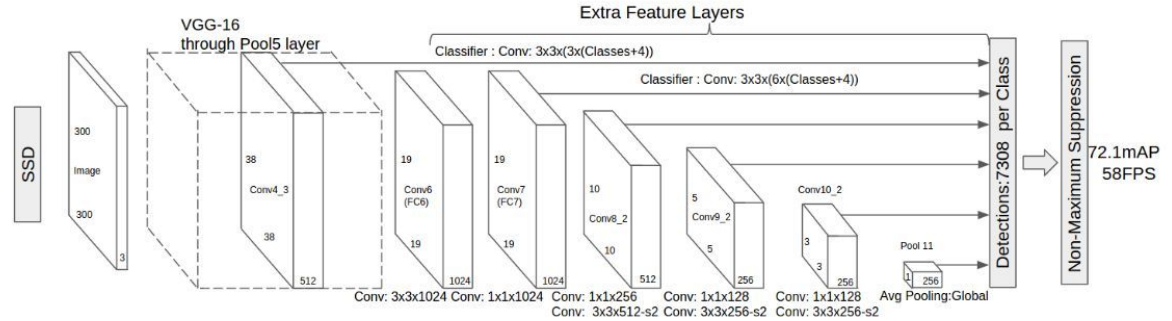
YOLO versions

- YOLO v2: BatchNorm, higher resolution, and anchor boxes.
- YOLO v3: objectness score to bounding box prediction, added connections to the backbone network
- YOLO v4, YOLOF, Scaled YOLO v4, PP-YOLO ...

SSD: Single Shot detector

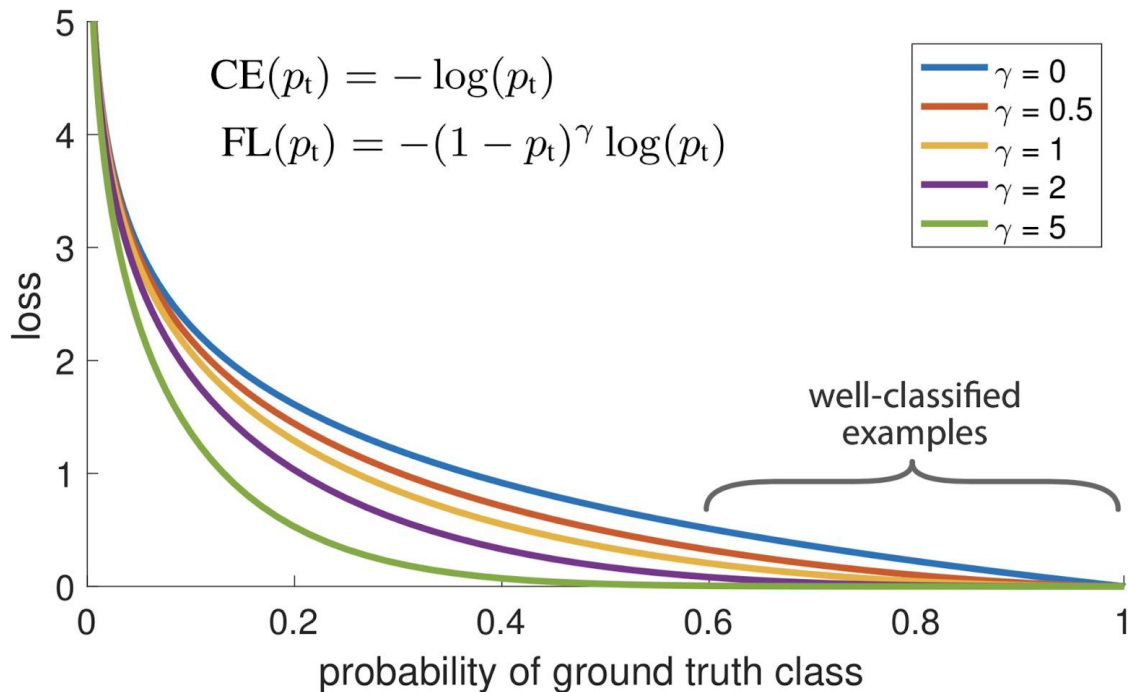
Based on same idea as YOLO

Use of different features maps to detect objects of different sizes



RetinaNET

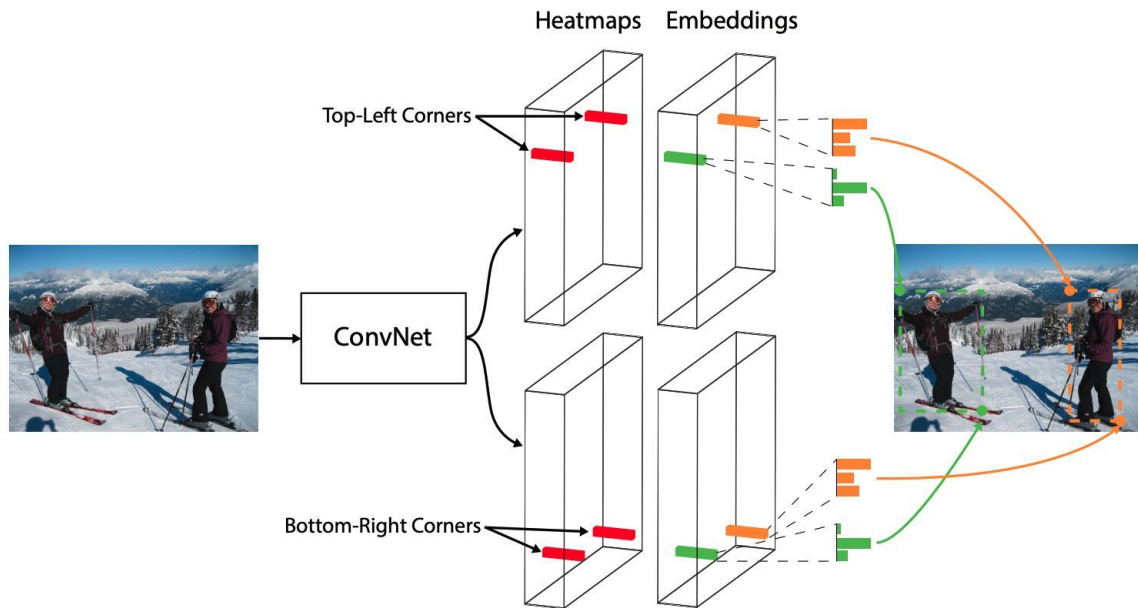
To deal with the imbalance between positive and negative examples in "one stage" methods, we suggest using Focal Loss: increase the loss of badly classified examples, compared to well classified examples.



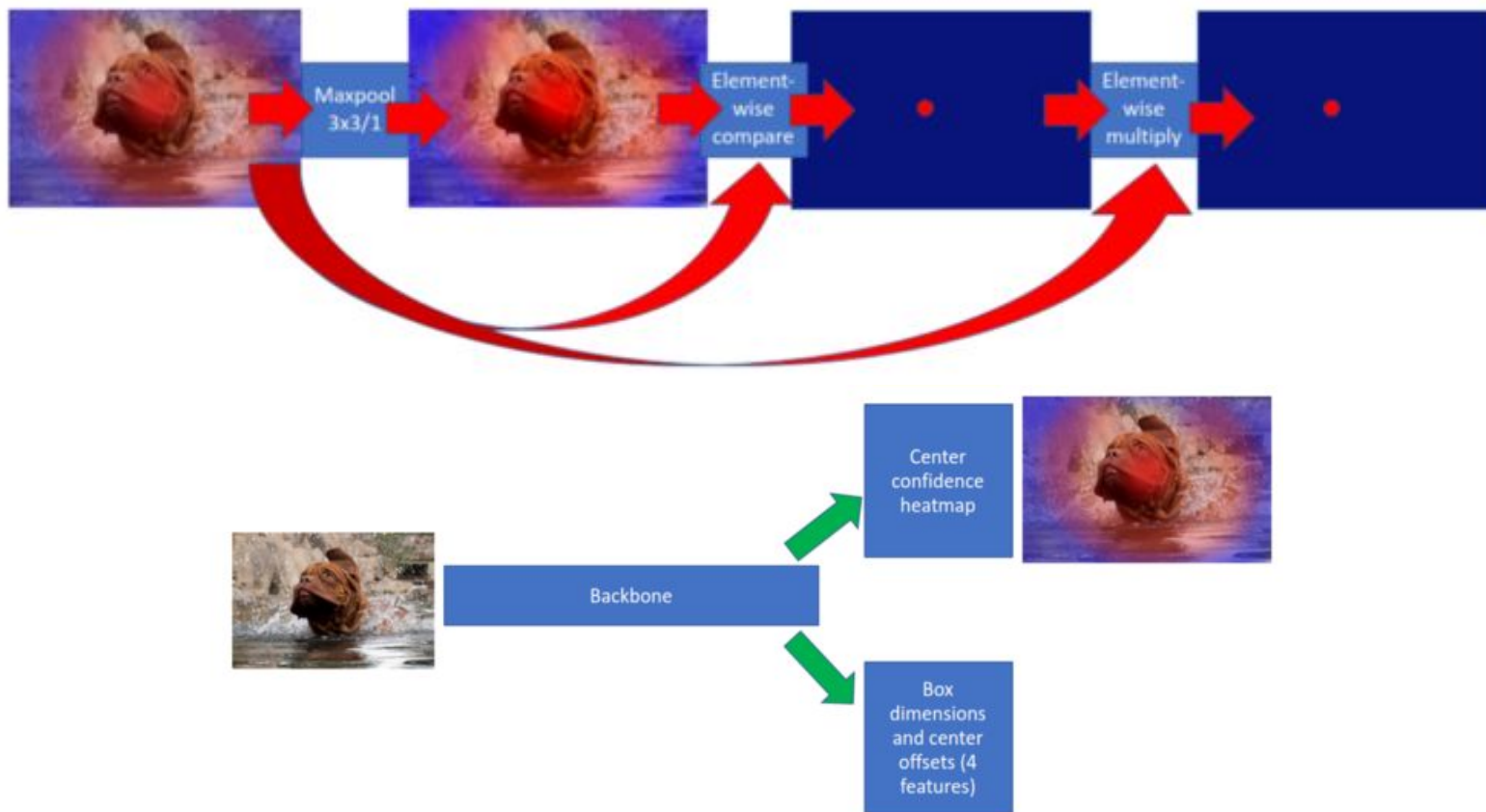
CornerNET

Use of heatmaps to predict corners.

The network predicts corner embedding. Embeddings are compared to find corners of same objects.

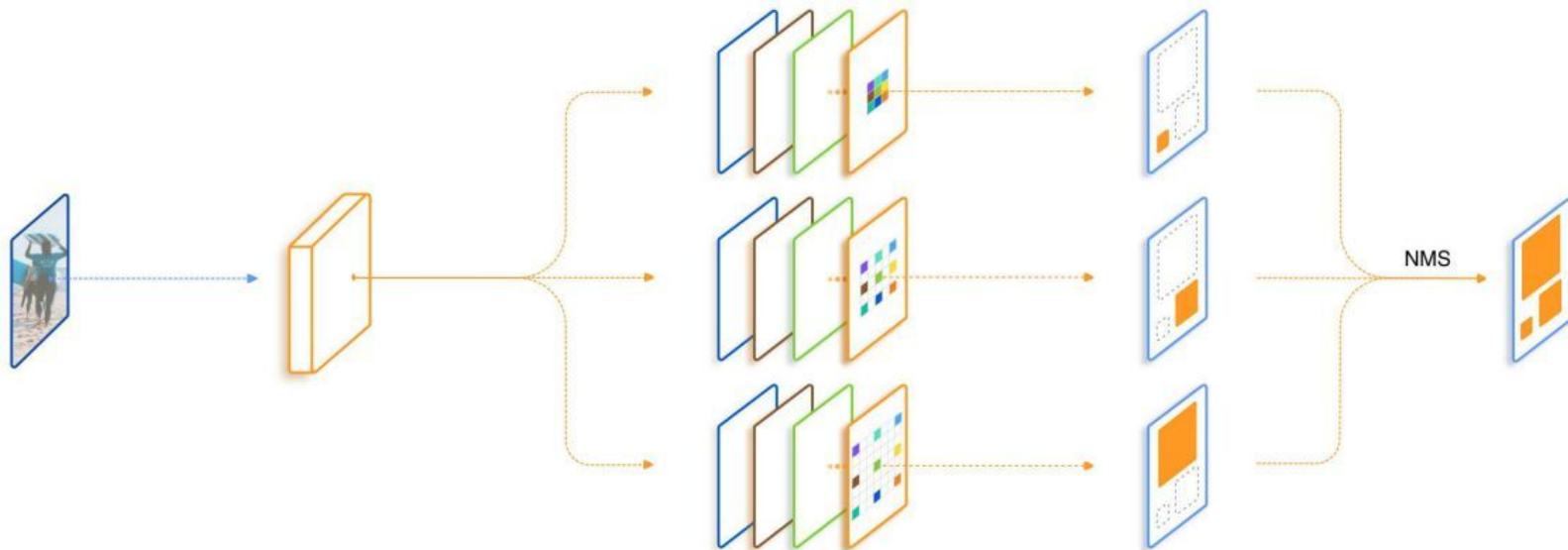


Center Net



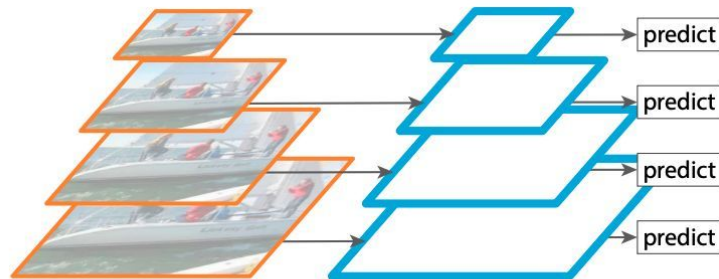
Trident Blocks

We use dilated convolutions to enlarge the “receptive field” of each point of the feature map. The size of the objects to be detected for each branch depends on the size of the reception field

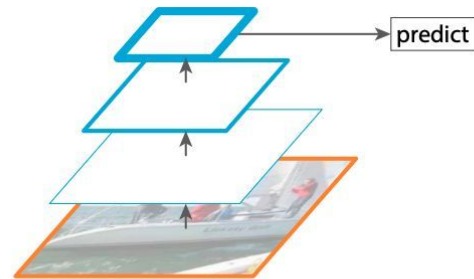


FPN: Feature Pyramid Network

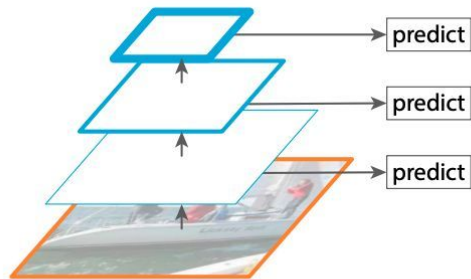
To detect objects
of different sizes



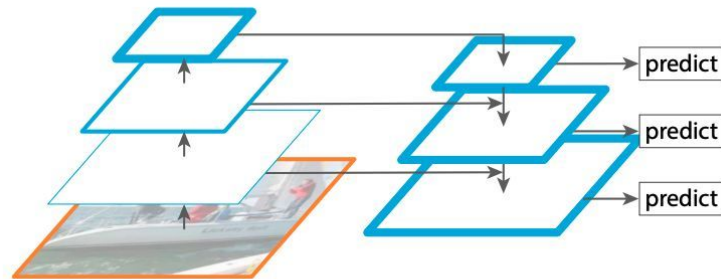
(a) Featurized image pyramid



(b) Single feature map



(c) Pyramidal feature hierarchy

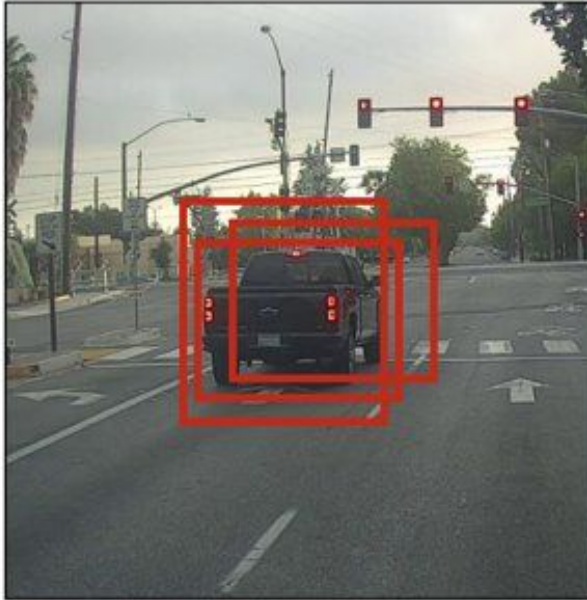


(d) Feature Pyramid Network

NMS: Non Maximum Suppression

Used for deleting duplicate detections

Before non-max suppression



**Non-Max
Suppression**



After non-max suppression



Other neural networks

- Transformers: originally created for NLP
- Context RCNN: using Attention system to detect objects of fixed cameras using past data
- FCN: fully convolutional network
-

Transfer Learning

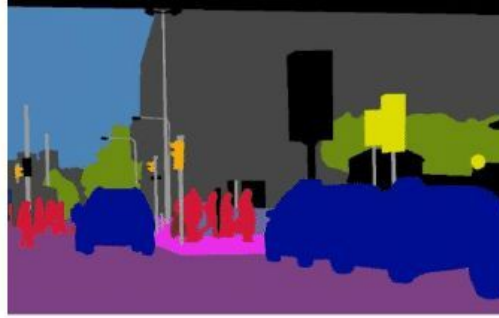
Use of classification neural networks known as ResNet and VGG pre-trained on reference image bases.

Fine-tuning: training of a few layers of the neural network to adapt it to the problem to be solved.

Types of segmentation



(a) Image



(b) Semantic Segmentation



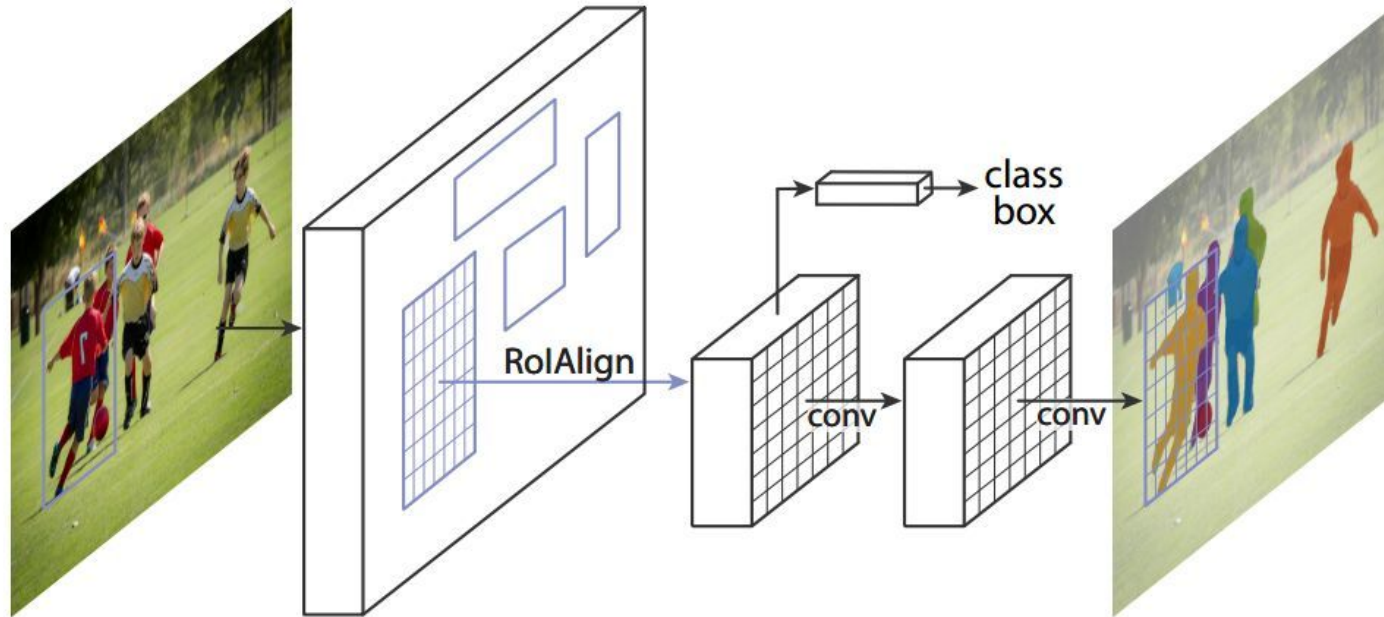
(c) Instance Segmentation



(d) Panoptic Segmentation

Mask RCNN

Object Detection + Instance Segmentation

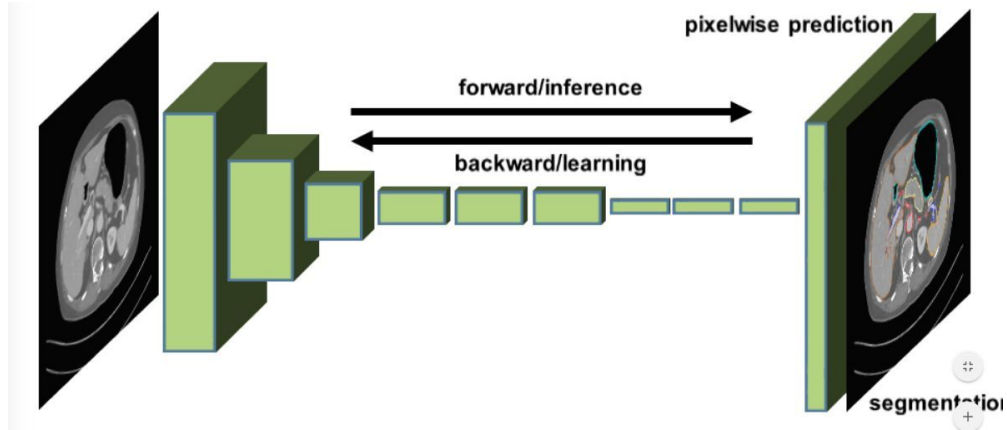
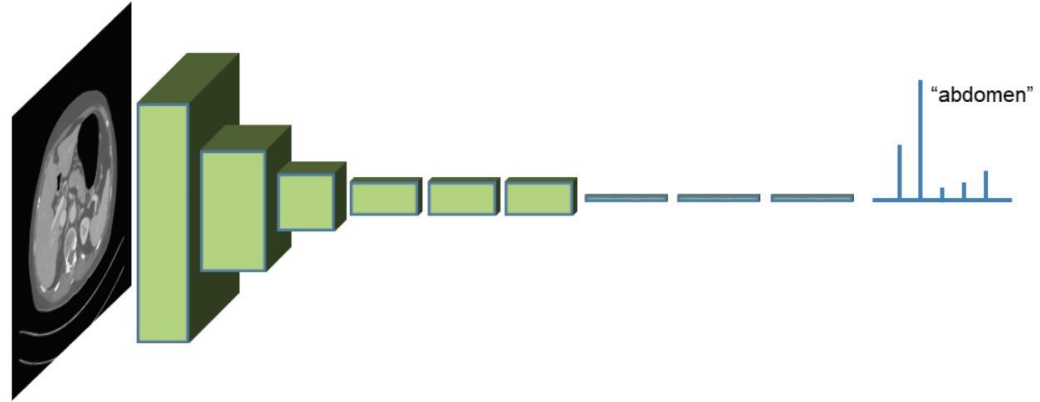


FCN: Fully Convolutional Network

Use the structure of a classification neural network

Replace FC by convolution (independent to the size of the input) and decrease in the number of parameters

Use of size increase layers: unpooling / deconvolution

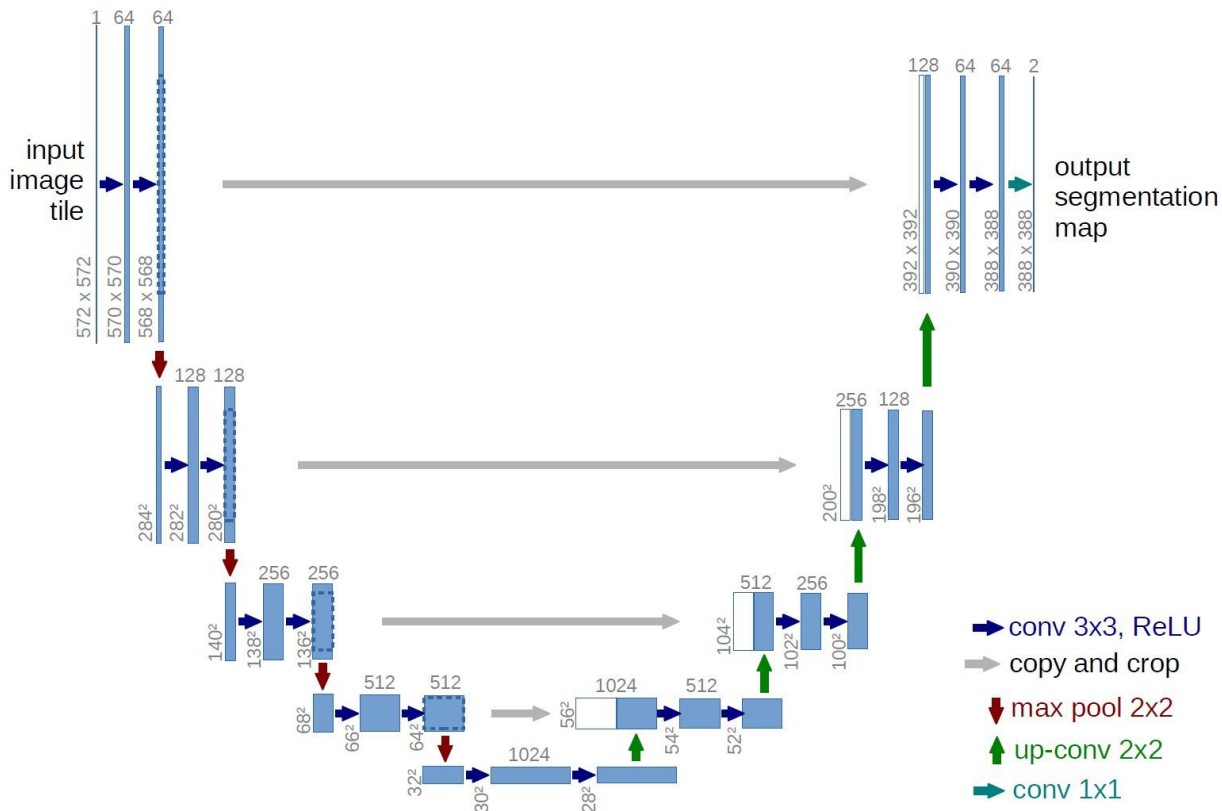


U-NET

Most often used in the medical field

Easy to implement and train

A basis for segmentation neural networks



Public dataset



PASCAL

20 categories
6k training images
6k validation images
10k test images



COCO

80 categories
200k training images
60k val + test images

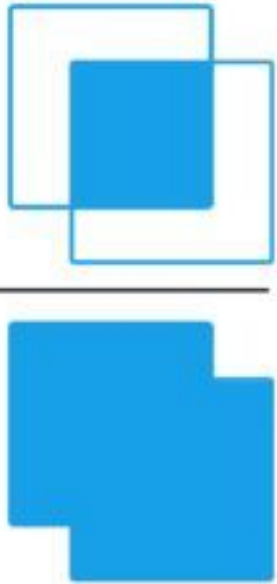


ILSVRC

200 categories
456k training images
60k validation + test images

Metrics: IOU

- Metric used to compare between two bounding boxes.
- Used in NMS to eliminate multidetections.
- Used in the evaluation to verify whether a detection corresponds to a ground truth object.

$$\text{IoU} = \frac{\text{Area of Overlap}}{\text{Area of Union}}$$


The diagram illustrates the calculation of Intersection over Union (IoU) between two bounding boxes. It shows two overlapping rectangles: a light blue rectangle and a dark blue rectangle. The intersection of the two rectangles is highlighted in a darker blue. Below the rectangles, the formula for IoU is shown as a fraction: the numerator is 'Area of Overlap' and the denominator is 'Area of Union'.

Métriques: mAP Mean Average Precision

Average precision:

- 10 recall intervals
- all categories
- different IOUs

