20/11/2024

Convolutional Neural Networks for Object Detection

Giacomo Boracchi
giacomo.boracchi@polimi.it
Artificial Neural Networks and Deep Learning
Politecnico di Milano,
https://boracchi.faculty.polimi.it/

Object Detection

Object Detection, the problem

Assign to an input image $I \in \mathbb{R}^{R \times C \times 3}$:

- **multiple** labels $\{l_i\}$ from a fixed set of categories $\Lambda = \{$"wheel", "cars", ..., "castle", "baboon"$\}$, each corresponding **to an instance of that object**
- the coordinates $\{(x, y, h, w)_i\}$ of the bounding box enclosing **each** object

$$I \rightarrow \{(x, y, h, w, l)_1, \dots, (x, y, h, w, l)_N\}$$

Object Detection Task

$I: \rightarrow \{\{\Lambda \times \mathbb{R} \times \mathbb{R} \times \mathbb{R} \times \mathbb{R}\}\}$

Given a fixed set of categories and an input image which contains an unknown and varying number of instances

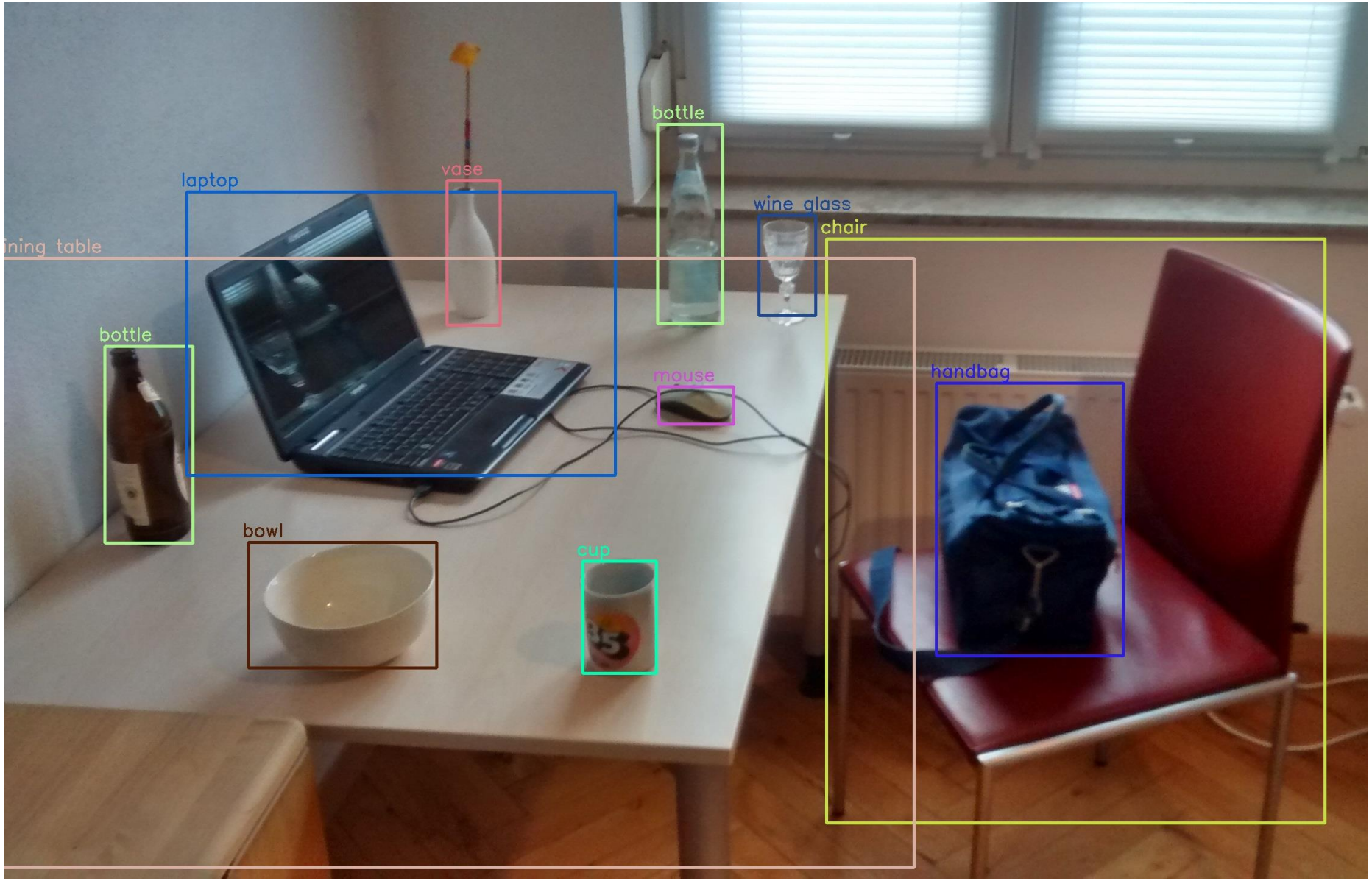
Draw a bounding box on each object instance

A training set of annotated images with labels and bounding boxes for each object is required

Each image requires **a varying number of outputs**

MAN: (x,y,h,w)
KID: (x,y,h,w)
GLOVE: (x,y,h,w)

Annotated Dataset for Object Detection



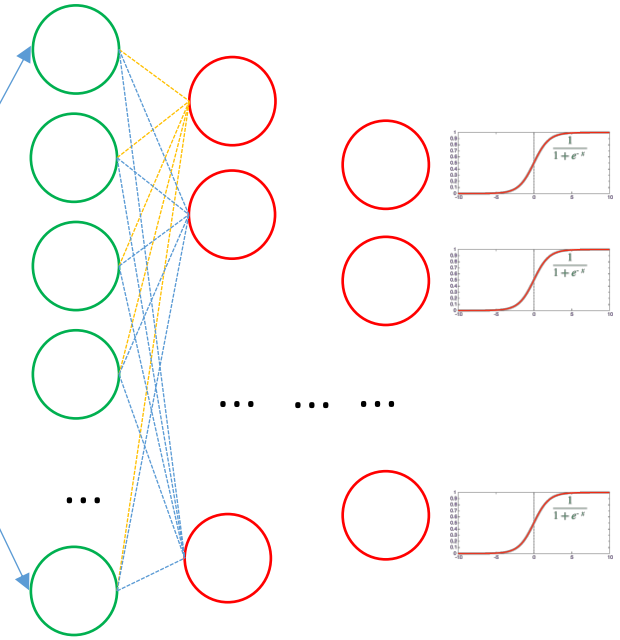
Excursus: Multi-label Classification

So far, all the models we have seen provide a fixed-sized output

CNNs (like other NN) can be modified to return **multiple classification output** by:

- Replace softmax activation in the output layer with sigmoid activation in each of the L neurons.
- The $i-$th neuron will predict the probability $p_i \in [0,1]$ of class membership for the input in a non-exclusive manner.
- The model must be trained with the binary cross-entropy loss function.

FEN + GAP/flatten



$p_1 \in [0,1]$

$p_2 \in [0,1]$

$p_L \in [0,1]$

These probabilities do not sum to 1, an input image might belong to multiple categories, depending on the content. This picture will likely be assigned to class «dog» and «cat»

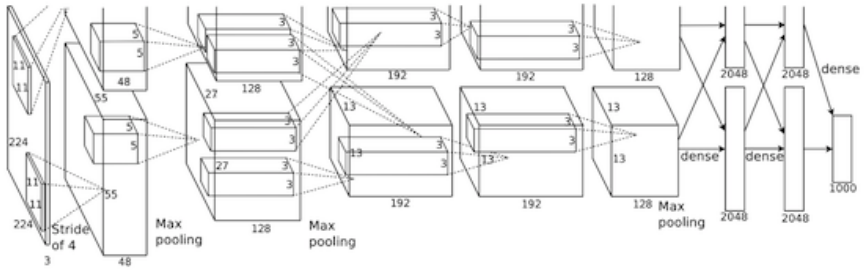
Image credit Sephiroty Magno Fiesta's
https://www.flickr.com/photos/sephiroty/317320544/

The Straightforward Solution: Sliding Window

1000 x 2000 pixels

- Similar to the sliding window for semantic segmentation
- A pretrained model is meant to process a fixed input size (e.g. 224 x 224 x 3)
- Slide on the image a window of that size and classify each region.

Adopt the whole machinery seen so far to each crop of the image



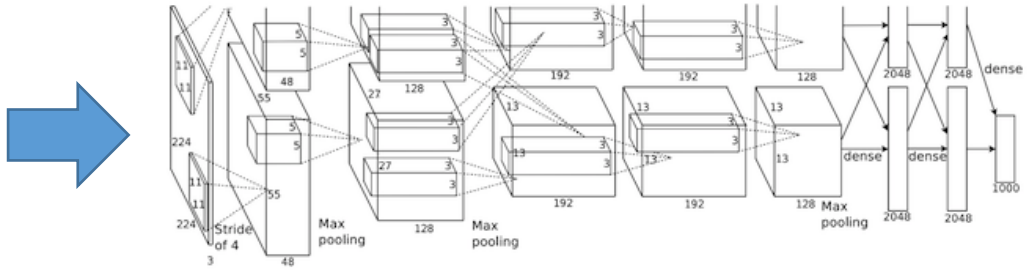
car

The Straightforward Solution: Sliding Window

1000 x 2000 pixels

- Similar to the sliding window for semantic segmentation
- A pretrained model is meant to process a fixed input size (e.g. 224 x 224 x 3)
- Slide on the image a window of that size and classify each region.

Adopt the whole machinery seen so far to each crop of the image



wheel

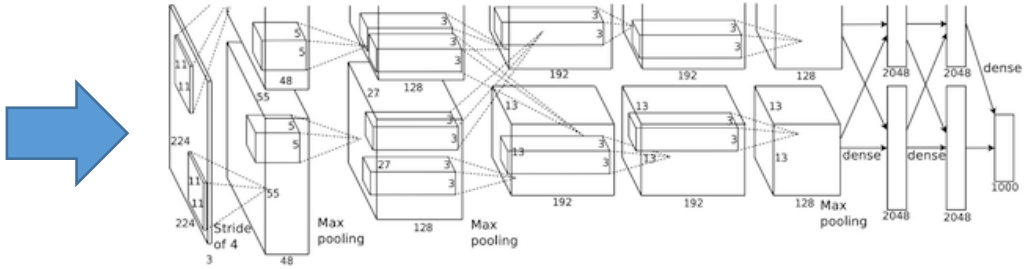
The Straightforward Solution: Sliding Window

1000 x 2000 pixels

- Similar to the sliding window for semantic segmentation
- A pretrained model is meant to process a fixed input size (e.g. 224 x 224 x 3)
- Slide on the image a window of that size and classify each region.

Adopt the whole machinery seen so far to each crop of the image

The background class has to be included!



background

Many drawbacks…

Cons:

- **Very inefficient!** Does not re-use features that are «shared» among overlapping crops
- How to choose the crop size?
- Difficult to detect objects at different scales!
- A huge number of crops of different sizes should be considered….

Plus:

- No need of retraining the CNN

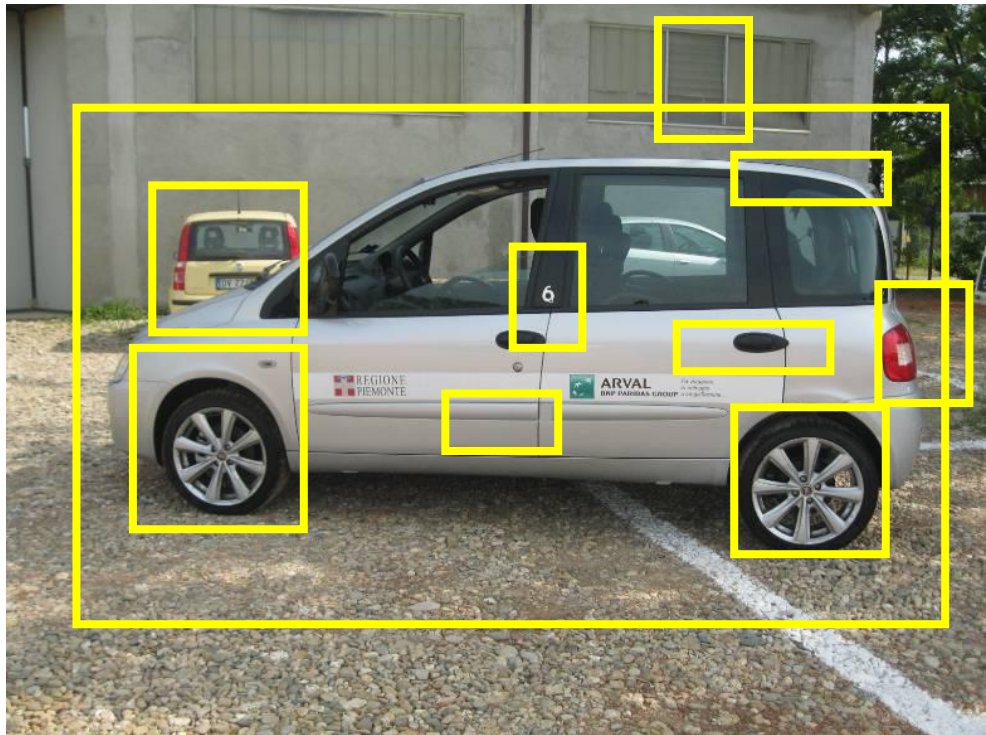
Region Proposal

Region proposal algorithms (and networks) are meant **to identify bounding boxes** that correspond to a **candidate object** in the image.

Algorithms with **very high recall** (but low precision) were there before the deep learning advent

The idea is to:

- Apply a region proposal algorithm
- Classify by a CNN the image inside each proposal regions

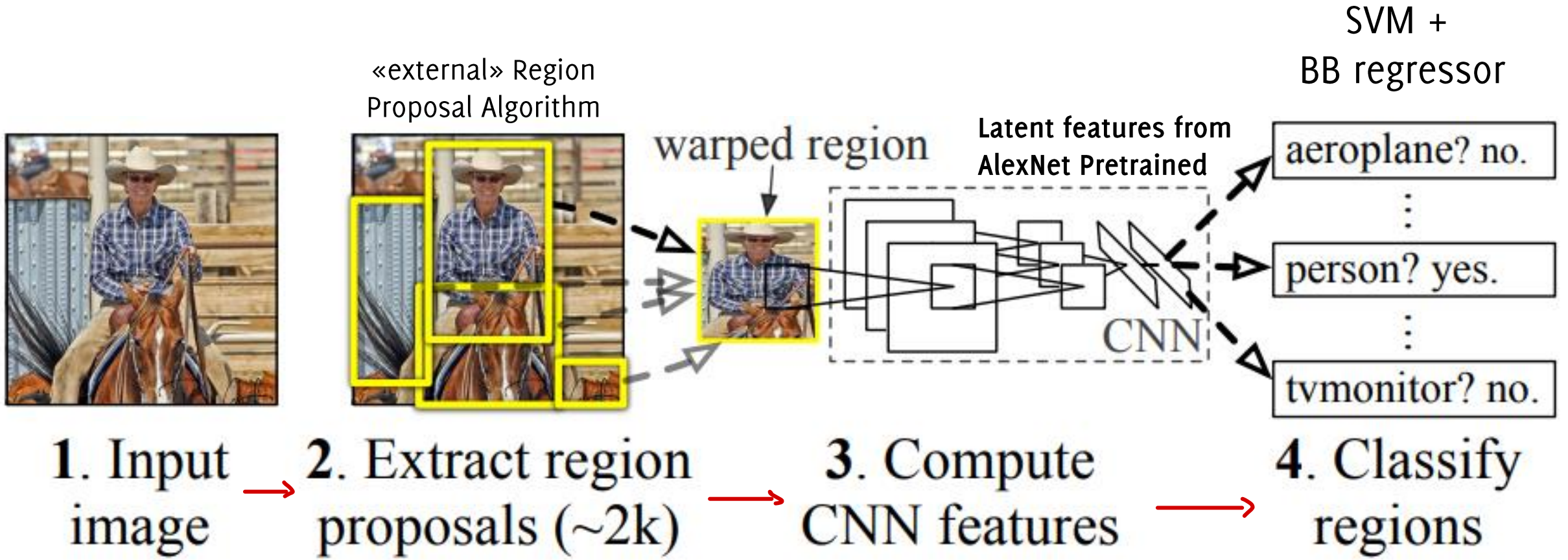


Rich feature hierarchies for accurate object detection and semantic segmentation

Ross Girshick[1] Jeff Donahue[1,2] Trevor Darrell[1,2] Jitendra Malik[1]
[1]UC Berkeley and [2]ICSI
{rbg,jdonahue,trevor,malik}@eecs.berkeley.edu

R-CNN

Object detection by means of region proposal (R stands for **regions**)

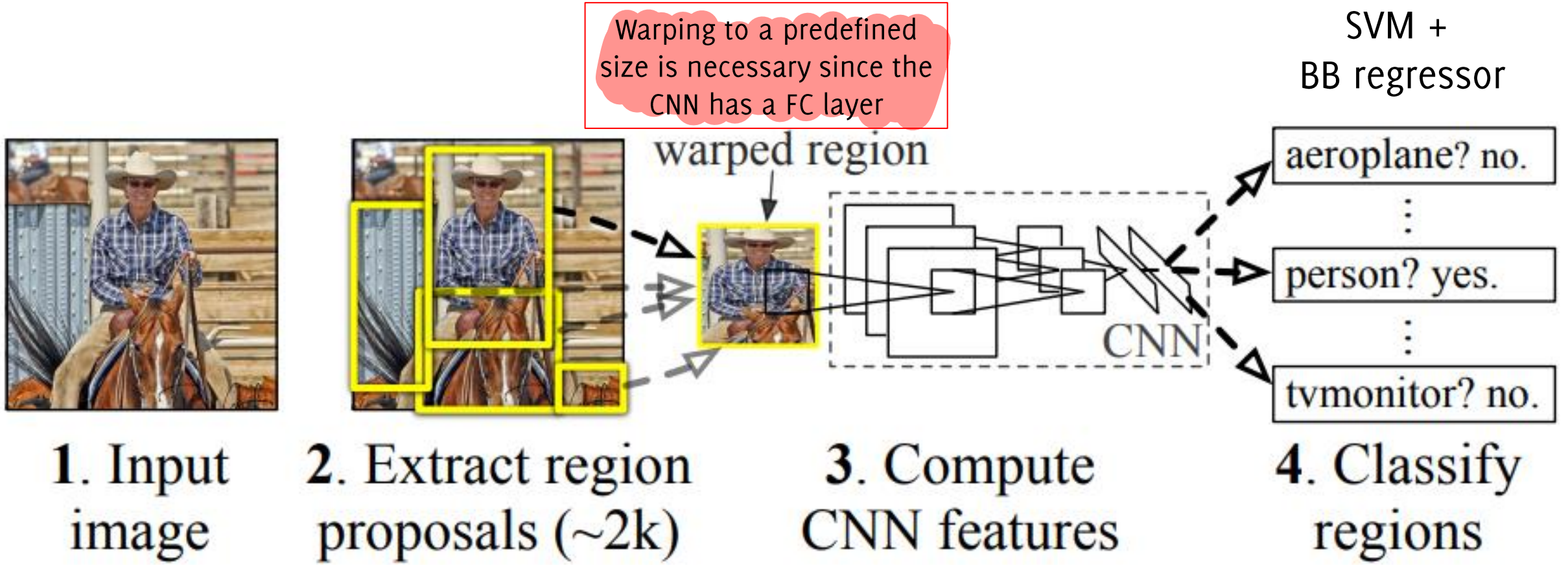


Girshick, Ross, et al. "Rich feature hierarchies for accurate object detection and semantic segmentation." *CVPR 2014*

R-CNN

Object detection by means of region proposal

Warping to a predefined size is necessary since the CNN has a FC layer



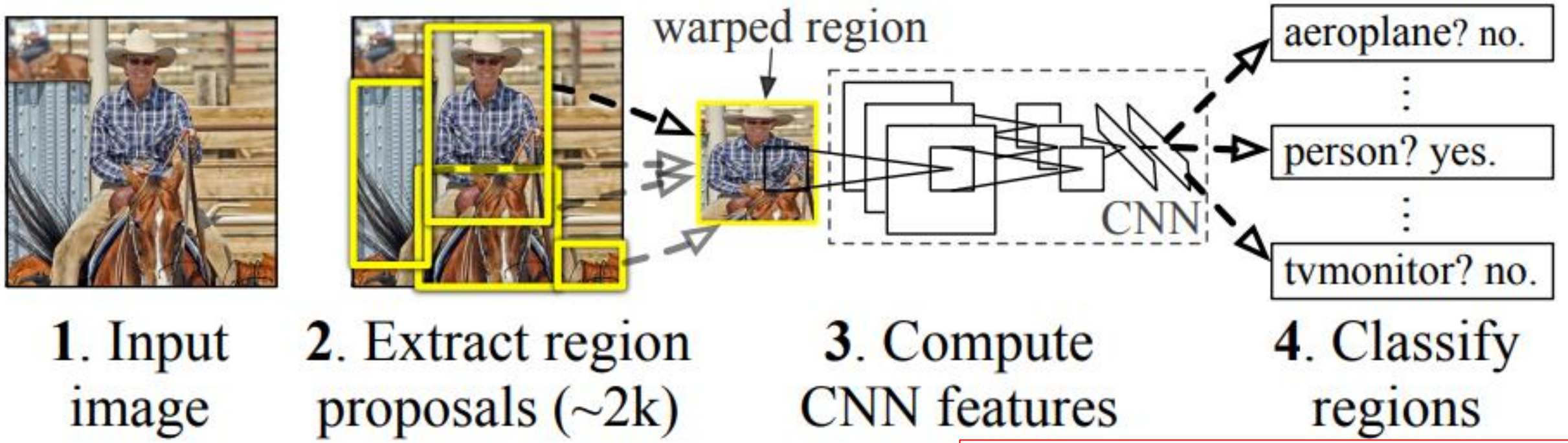
There is no learning in the region proposal algorithm, **very high recall (e.g. Selective Search)**

Girshick, Ross, et al. "Rich feature hierarchies for accurate object detection and semantic segmentation." *CVPR 2014*

R-CNN

This is AlexNet as a pre-trained network

SVM +
BB regressor



SVM trained to minimize the classification error over the extracted ROI

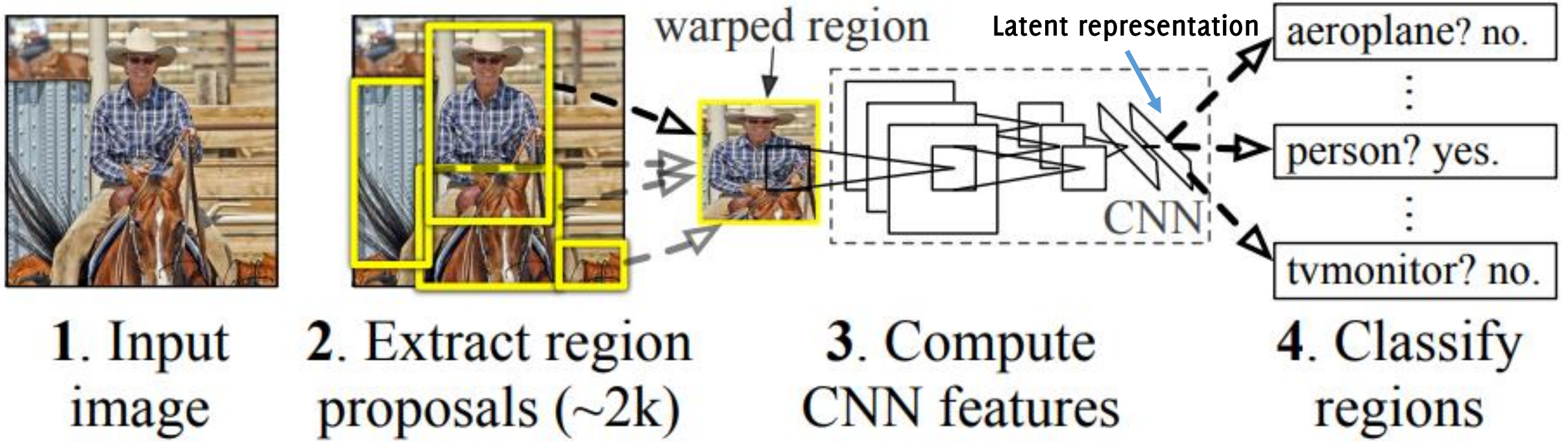
Girshick, Ross, et al. "Rich feature hierarchies for accurate object detection and semantic segmentation." *CVPR 2014*

R-CNN

(x, y, h, w)

The **regions are refined** by a regression network to correct the bounding box estimate from ROI algorithm

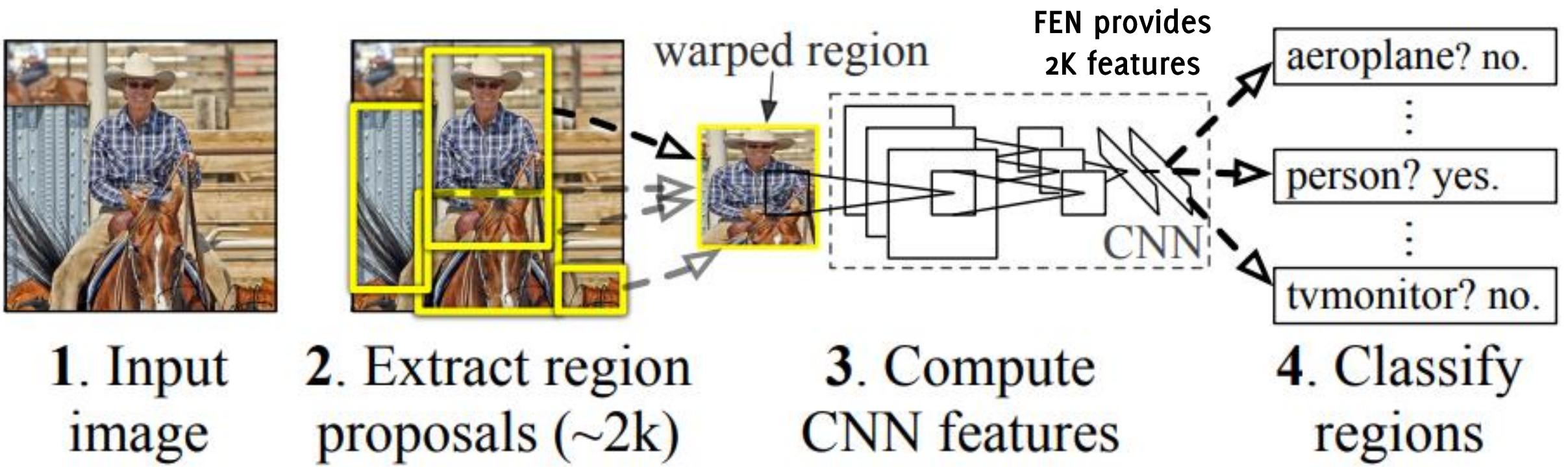
SVM + BB regressor



Girshick, Ross, et al. "Rich feature hierarchies for accurate object detection and semantic segmentation." *CVPR 2014*

R-CNN

The pretrained CNN is fine-tuned over the classes to be detected (21 vs 1000 of Alextnet) by placing a FC layer after feature extraction.
No end-to-end training of the SVM

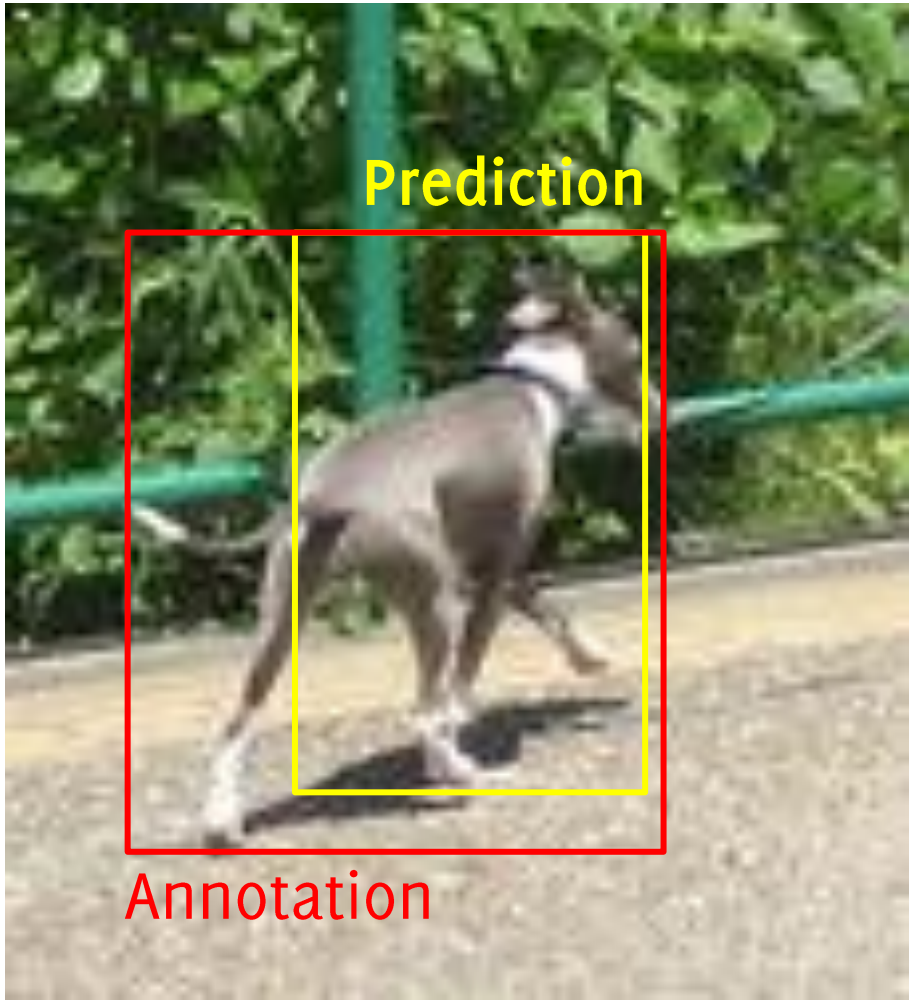


Include **a background class** to get rid of those regions not corresponding to an object

Girshick, Ross, et al. "Rich feature hierarchies for accurate object detection and semantic segmentation." *CVPR 2014*

The Loss Function (over bounding boxes)

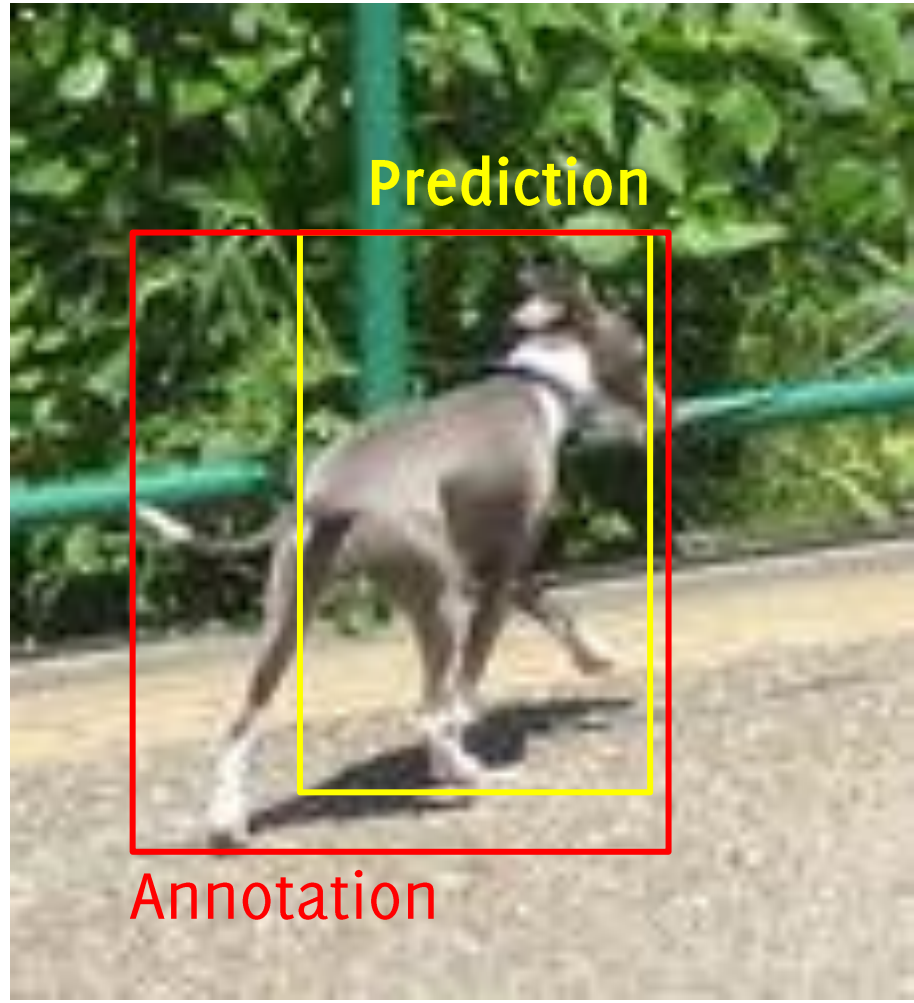
We need to quantitatively assess the network performance over each and every image in the test set



The Loss Function (over bounding boxes)

We need to quantitatively assess the network performance over each and every image in the test set

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



A number quantitatively assessing detection preformance

By Adrian Rosebrock - http://www.pyimagesearch.com/2016/11/07/intersection-over-union-iou-for-object-detection/

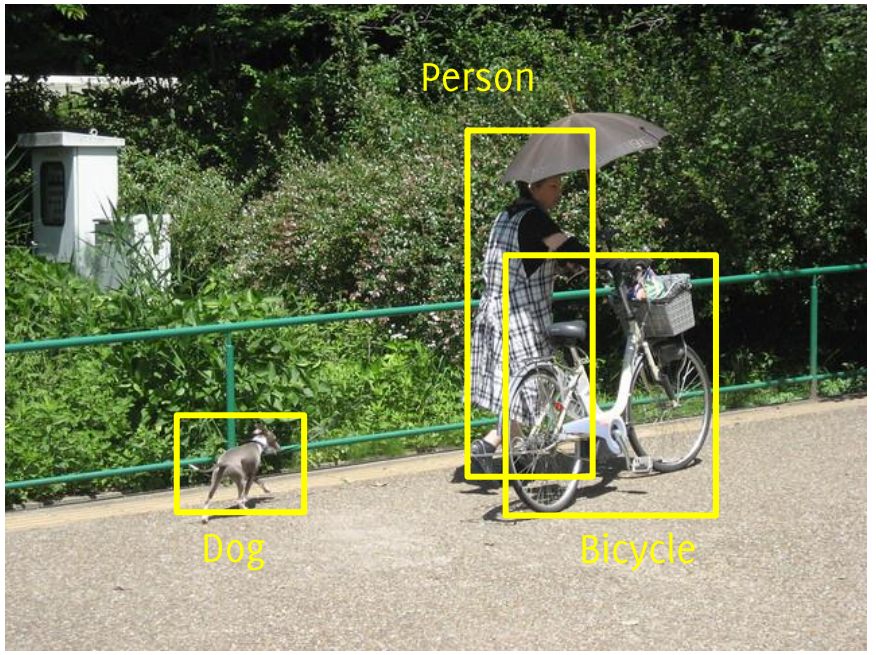
The Loss Function

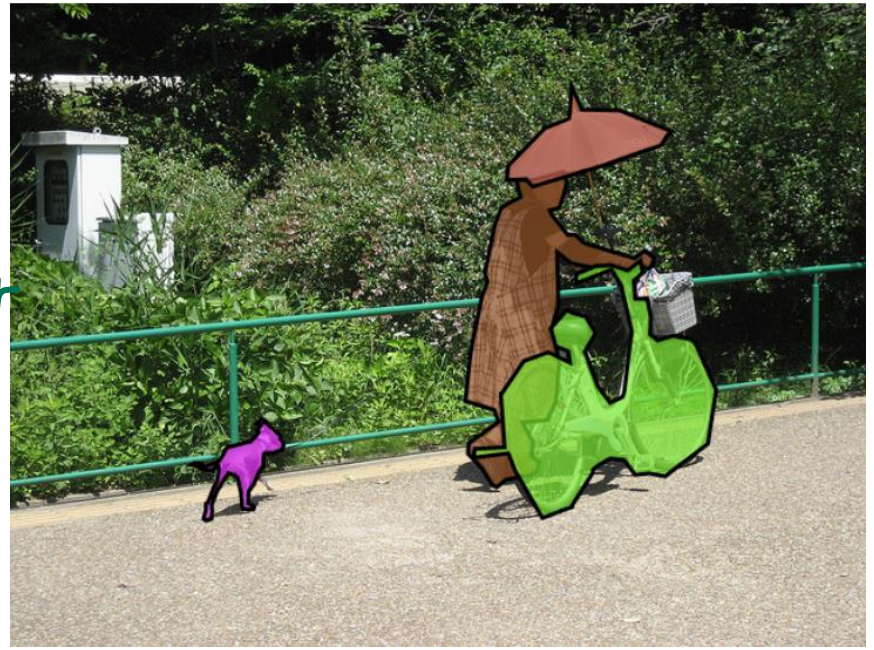
The loss function compares the detection results for an image x: $NN(x)$ against the annotations y and gives you a number

$$\mathcal{L}(y, \hat{y})$$

That is indicating «*how happy we are with the predictions*», the lower the better. This considers:

- how many missed items → classified as background but it was an object.
- how many false positives → classified as object but it was background.
- how close correct detections are to the annotation (IoU) → about bounding box.





R-CNN Limitations

- **Ad-hoc training objectives** and not an end-to-end training
  - Fine-tuning network with softmax classifier (log loss) before training SVM
  - Train post-hoc linear SVMs (hinge loss)
  - Train post-hoc bounding-box regressions (least squares)
- **Region proposals are from a different algorithm** and that part has not been optimized for the detection by CNN
- **Training is slow** (84h), takes **a lot of disk space to store features**
- **Inference** (detection) **is slow** since the CNN has to be executed on each region proposal (**no feature re-use**)
  - 47s / image with VGG16

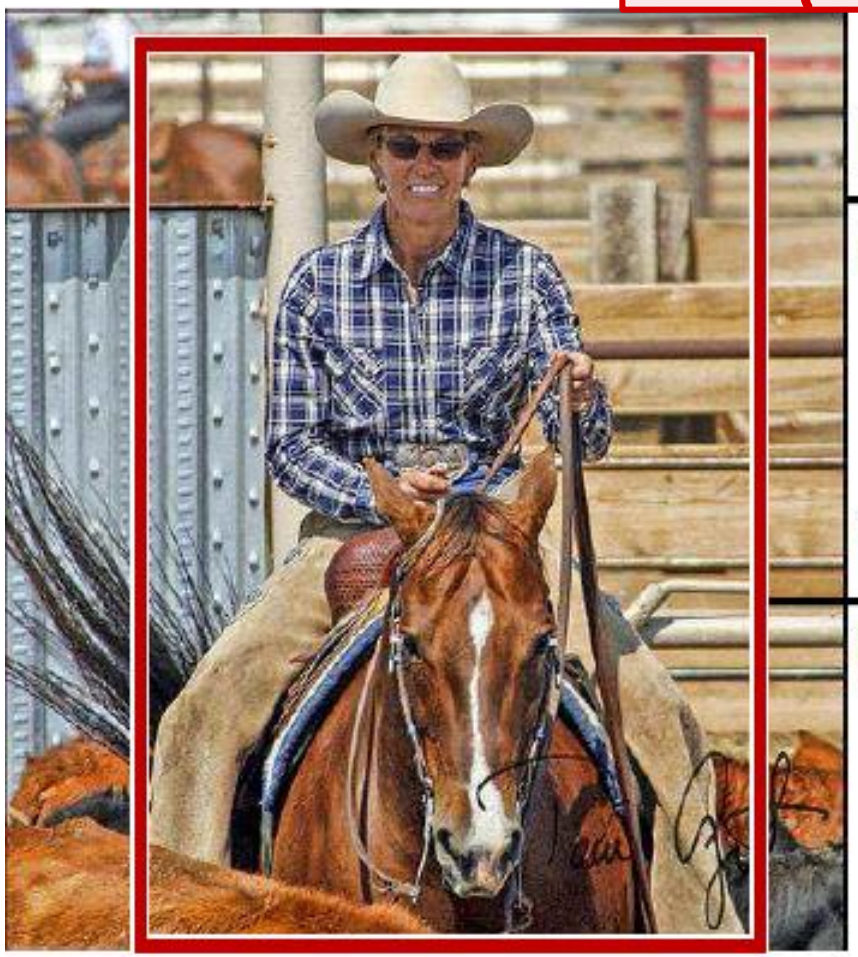
Rmk: efficiency in object detection network is key! Otherwise you might want to train a segmentation network instead!

Fast R-CNN

Ross Girshick
Microsoft Research
rbg@microsoft.com

Fast R-CNN

Explained in the next few slides



Region from external RPA

Convolutional features

Deep ConvNet

RoI projection

Conv feature map

brings each region to the same size (like 16x16).

RoI pooling layer

FCs

Outputs:

softmax

bbox regressor

FC

FC

RoI feature vector

For each RoI

«conv5»

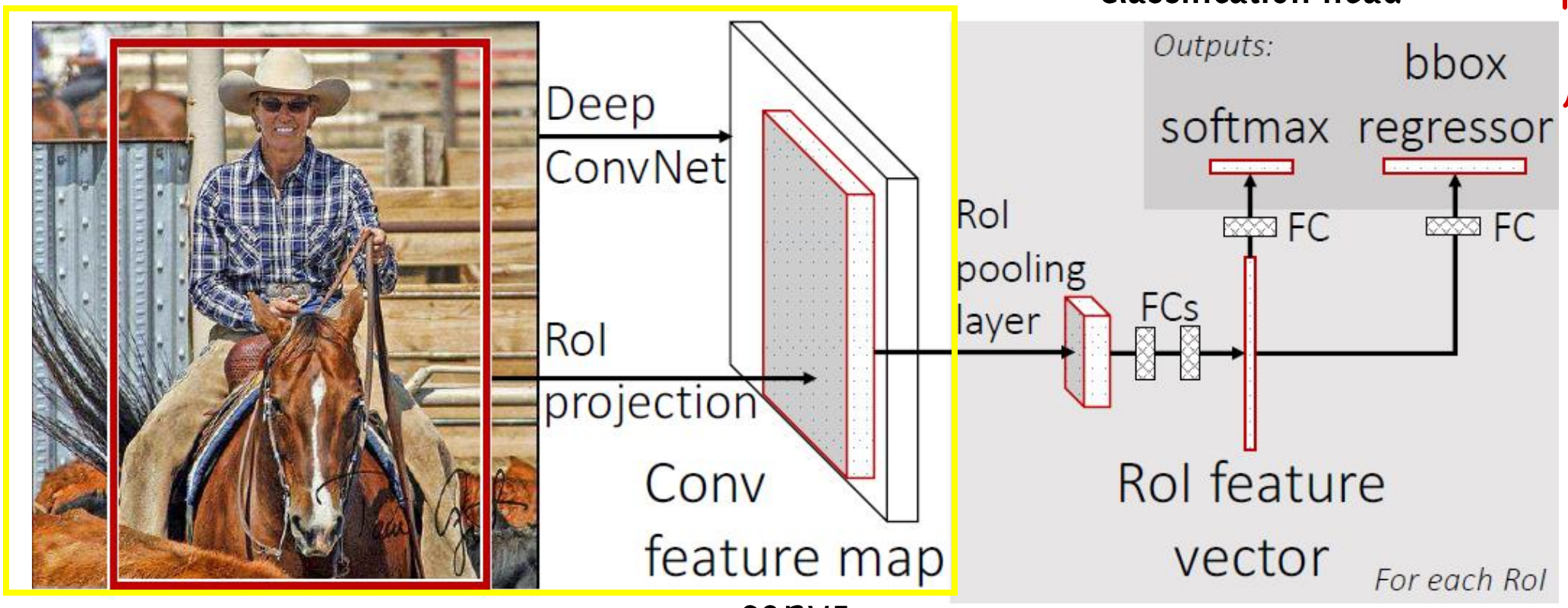
⚠ we get rid of the SVM.
↳ replace it by NN!

Girshick, Ross. "Fast r-cnn." ICCV 2015

Fast R-CNN

1. The whole image is fed to a CNN that extracts feature maps.
2. **Region proposals** are identified from the image and **projected into the feature maps.** Regions are directly cropped form the feature maps, instead from the image: →**re-use convolutional computation**.

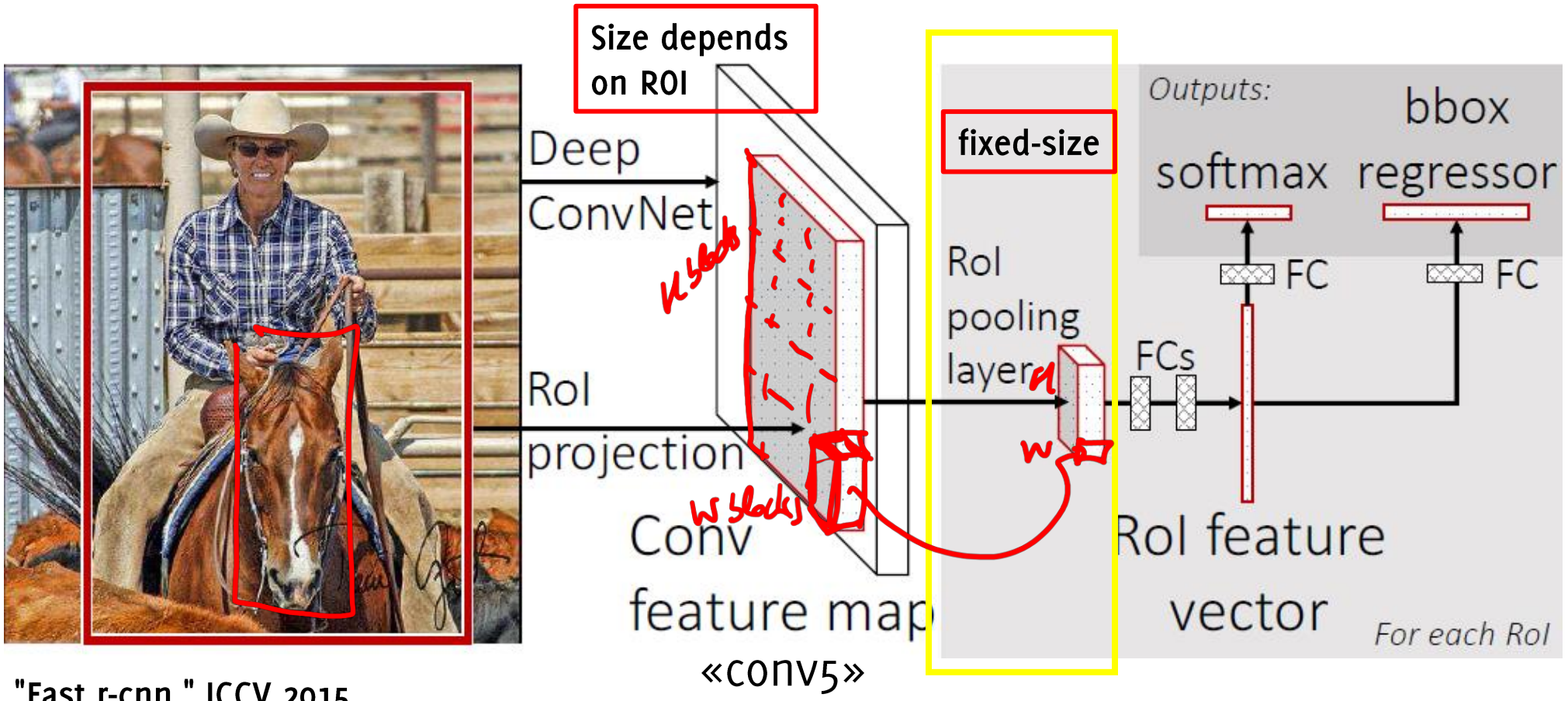
Ooo Smart! To avoid re computing the convolution.



Girshick, Ross. "Fast r-cnn." ICCV 2015

Fast R-CNN

3. Fixed size is still required to feed data to a fully connected layer.
   ROI pooling layers: extract a **fixed size $H \times W$ activation** from each region proposal. Each **ROI in the feature maps is divided in a $H \times W$ grid** and then **maxpooling** over the grid provides a fixed size input (vectorized) to the next step.



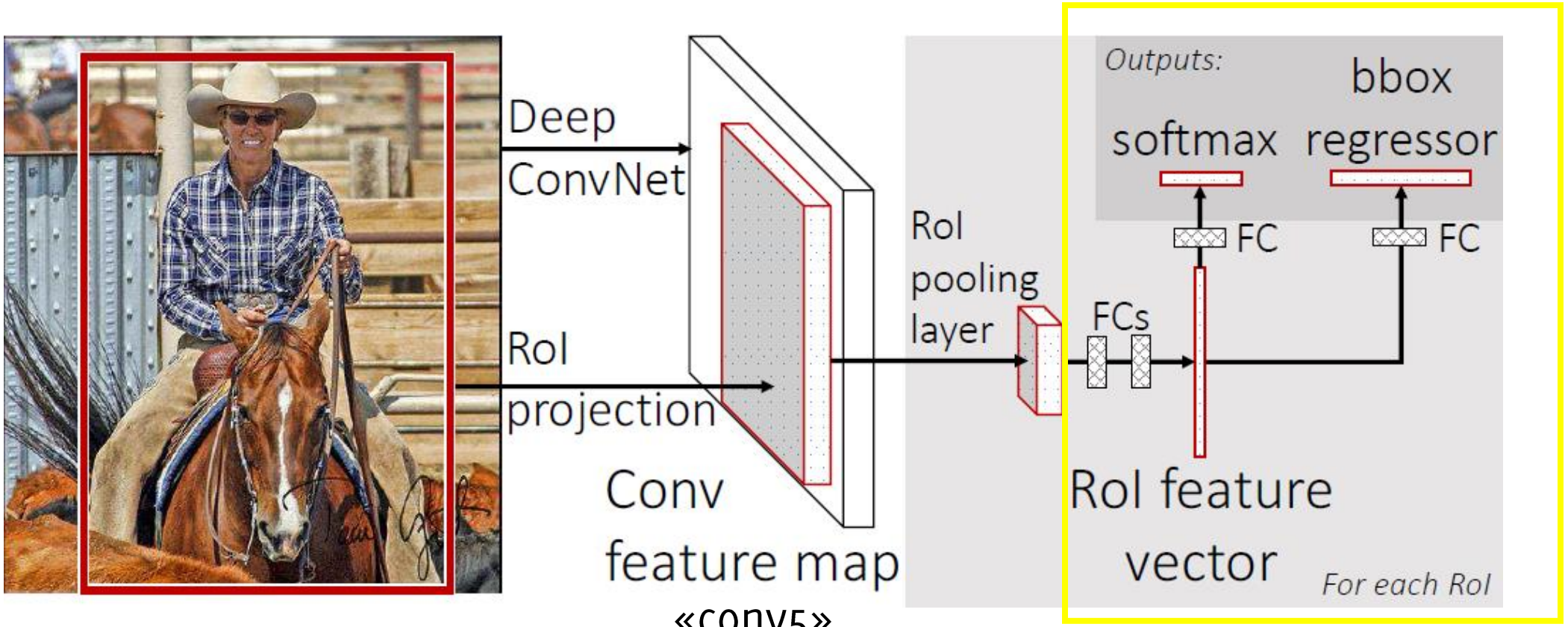
Girshick, Ross. "Fast r-cnn." ICCV 2015

Fast R-CNN

4. The **FC layers estimate both classes and BB location** (bb regressor)
   A **convex combination of the two** is used as a **multitask loss** to be optimized (as in R-CNN, but no SVM here).

↑ cf slide 21.

5. Training performed in an **end-to-end manner, convolutional part executed only once**



Girshick, Ross. "Fast r-cnn." ICCV 2015

Fast R-CNN

In this new architecture it is possible to **back-propagate through the whole network**, thus train the whole network in an end-to-end manner.

It becomes **incredibly faster than R-CNN during testing**.

Now that convolutions are not repeated on overlapping areas, **the vast majority of test time is spent on ROI extraction** (e.g. selective search).

Girshick, Ross. "Fast r-cnn." ICCV 2015

Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks

Shaoqing Ren* **Kaiming He** **Ross Girshick** **Jian Sun**

Microsoft Research
{v-shren, kahe, rbg, jiansun}@microsoft.com

Faster R-CNN

What is new: you manage to make learnable the ROI extraction.

Conv backbone
Classification

RPN

Detection head

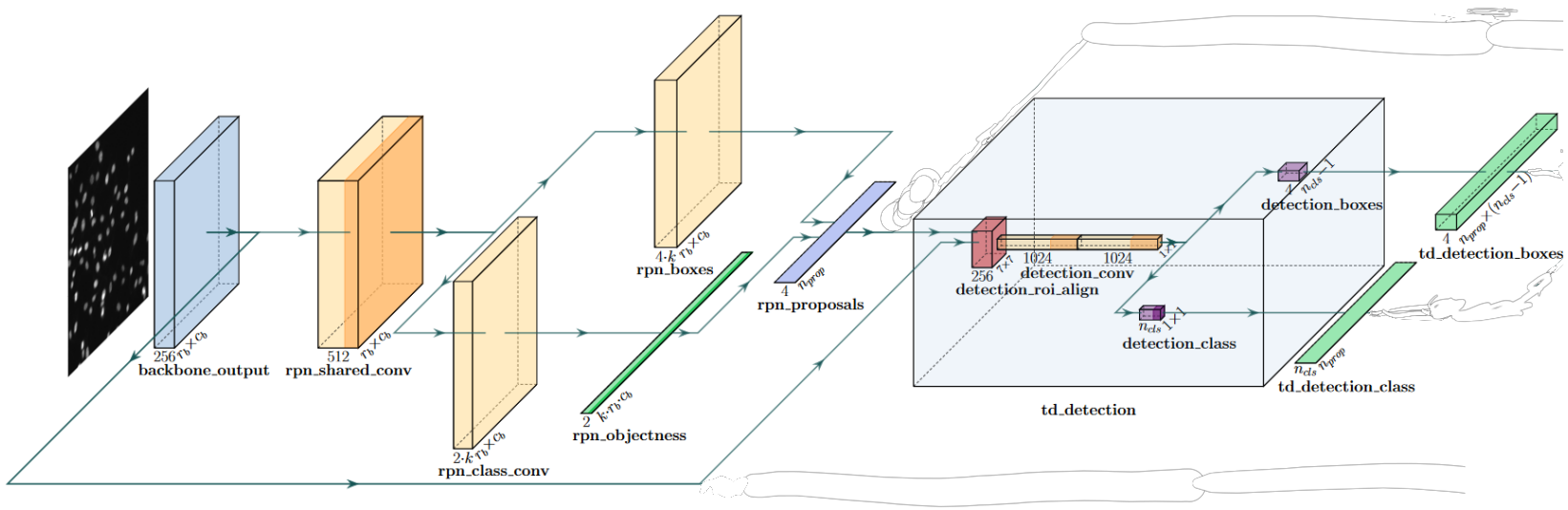
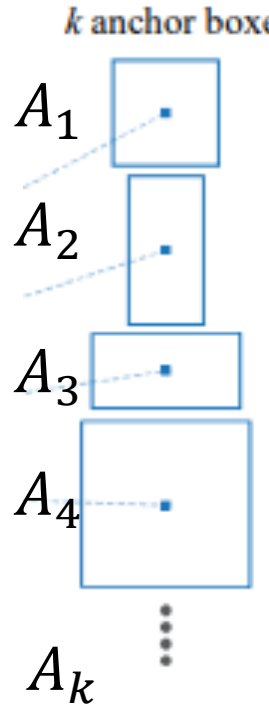


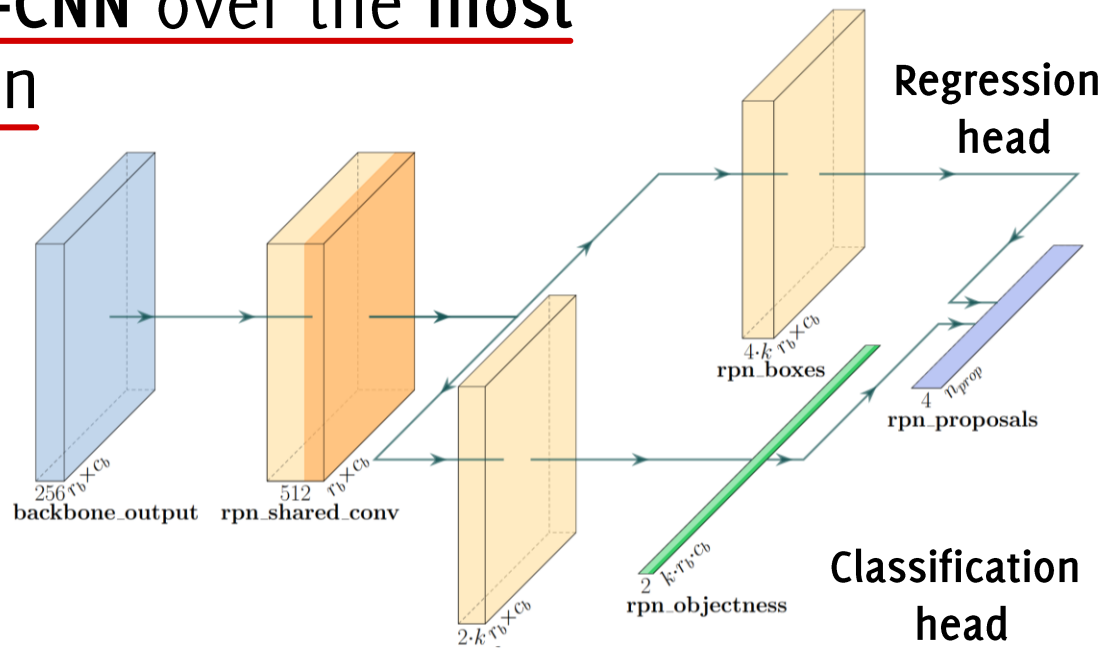
Image Credits Roberto Basla

RPN: The Region Proposal Network

Explained in the next few slides

- Instead of the ROI extraction algorithm, **train a region proposal network** (RPN), which is a F-CNN (3x3 filter size)
- **RPN operates on the same feature maps** used for classification, thus at the last conv layers
- RPN can be seen as **an additional (learnable) module** that **improves efficiency** and **focus Fast R-CNN** over the **most promising regions** for object detection



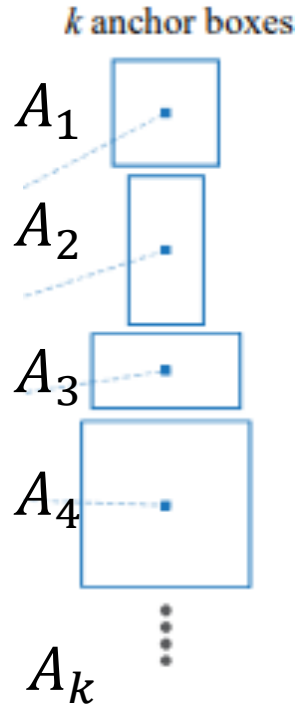


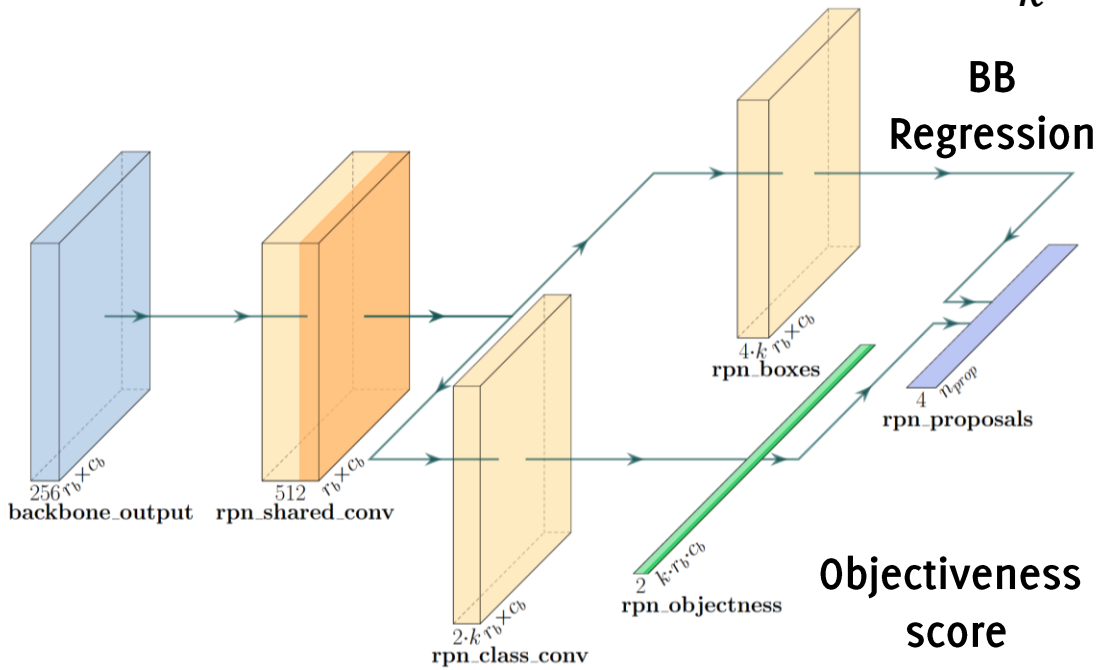
Ren, Shaoqing, et al. "Faster r-cnn: Towards real-time object detection with region proposal networks." *NIPS 2015*

RPN: Region Proposal Network

Goal: Associate to each spatial location k anchor boxes, i.e. ROI having different scales and ratios (e.g. $k = 3 \times 3$, 3 sizes of the anchor side, 3 height/width ratios). Assume the feature maps are $r_b \times c_b$.

The network outputs $r_b \times c_b \times k$ **candidates** anchor and estimate *objectiveness scores* for each anchor



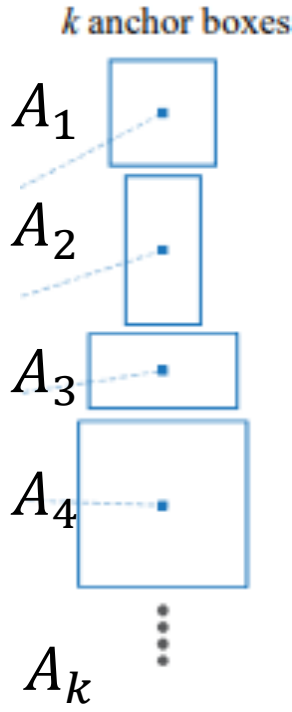


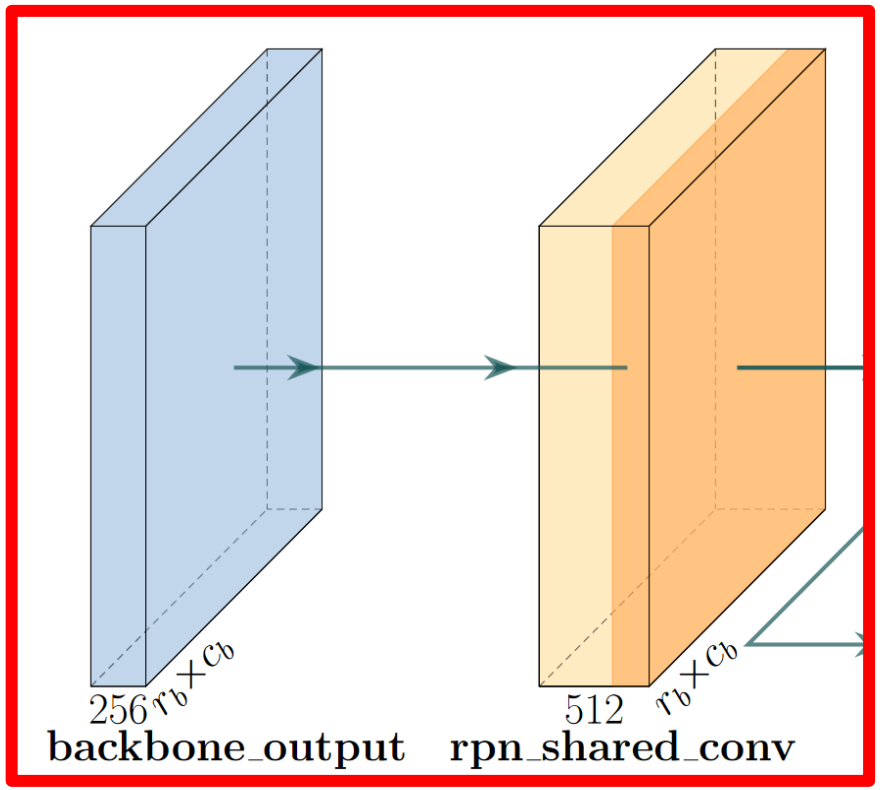
Ren, Shaoqing, et al. "Faster r-cnn: Towards real-time object detection with region proposal networks." *NIPS 2015*

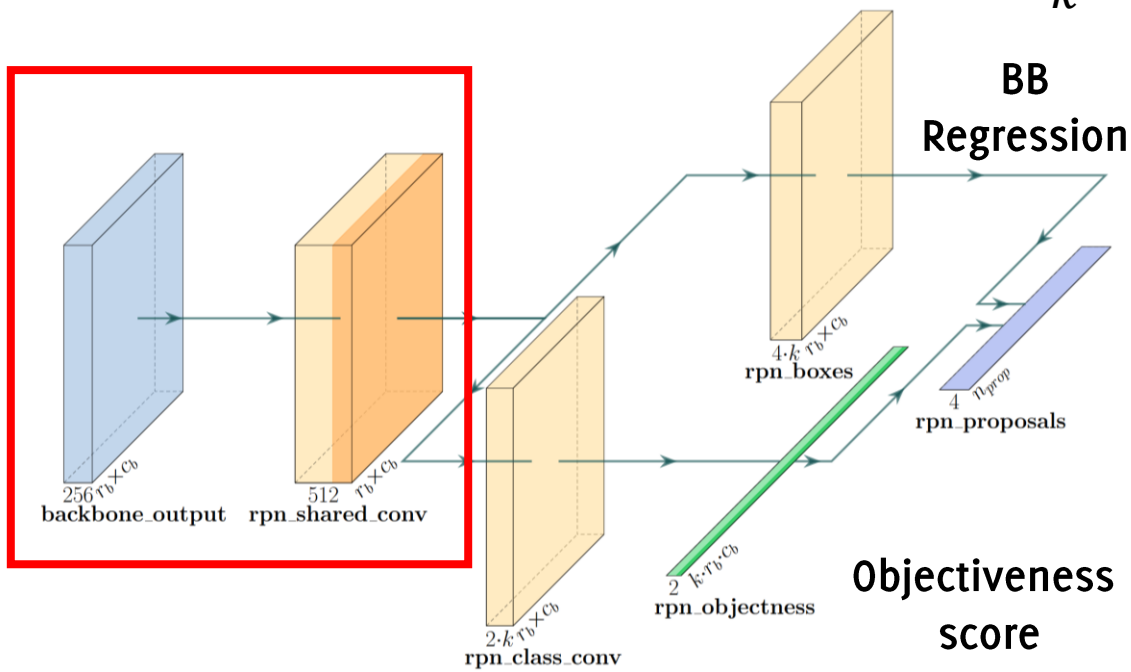
RPN: Intermediate Layer

Intermediate layer: conv2D that takes as input the last layer of the feature-extraction network and uses 256 filters of size 3×3.

Modifies the dimension of feature map, each region is embedded in a lower dimensional vector of size (output size $r_b \times c_b \times 512$)

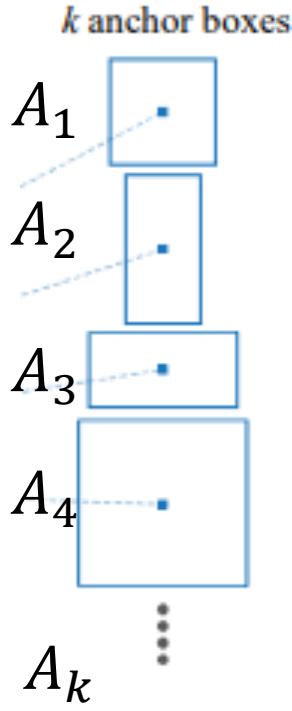


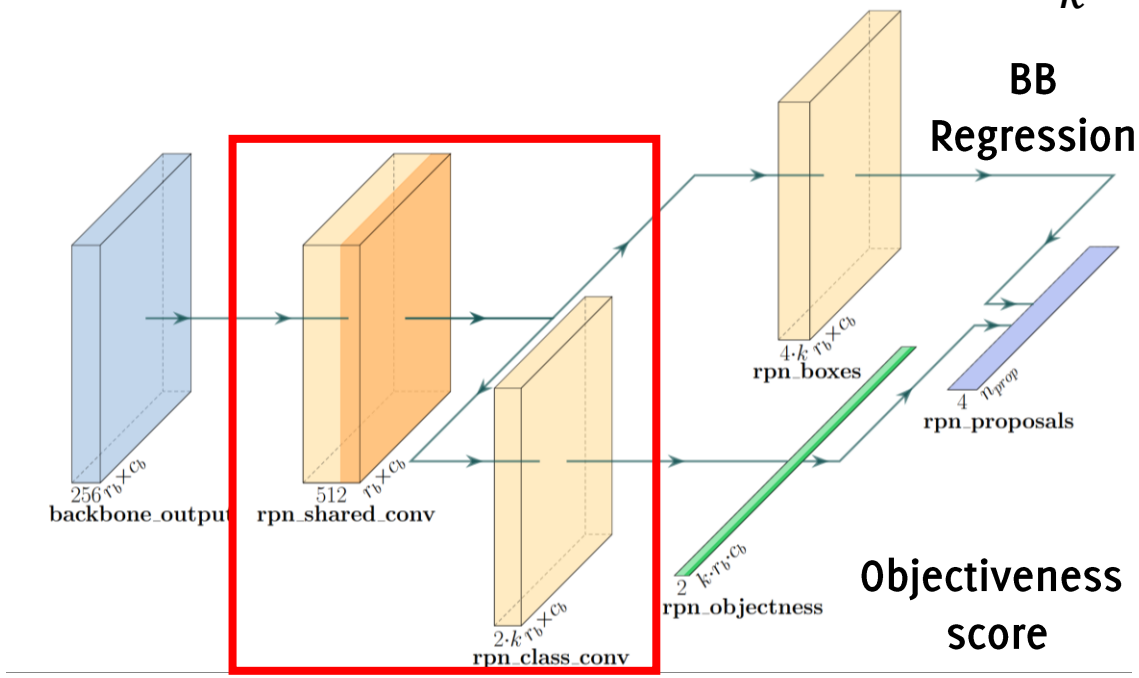




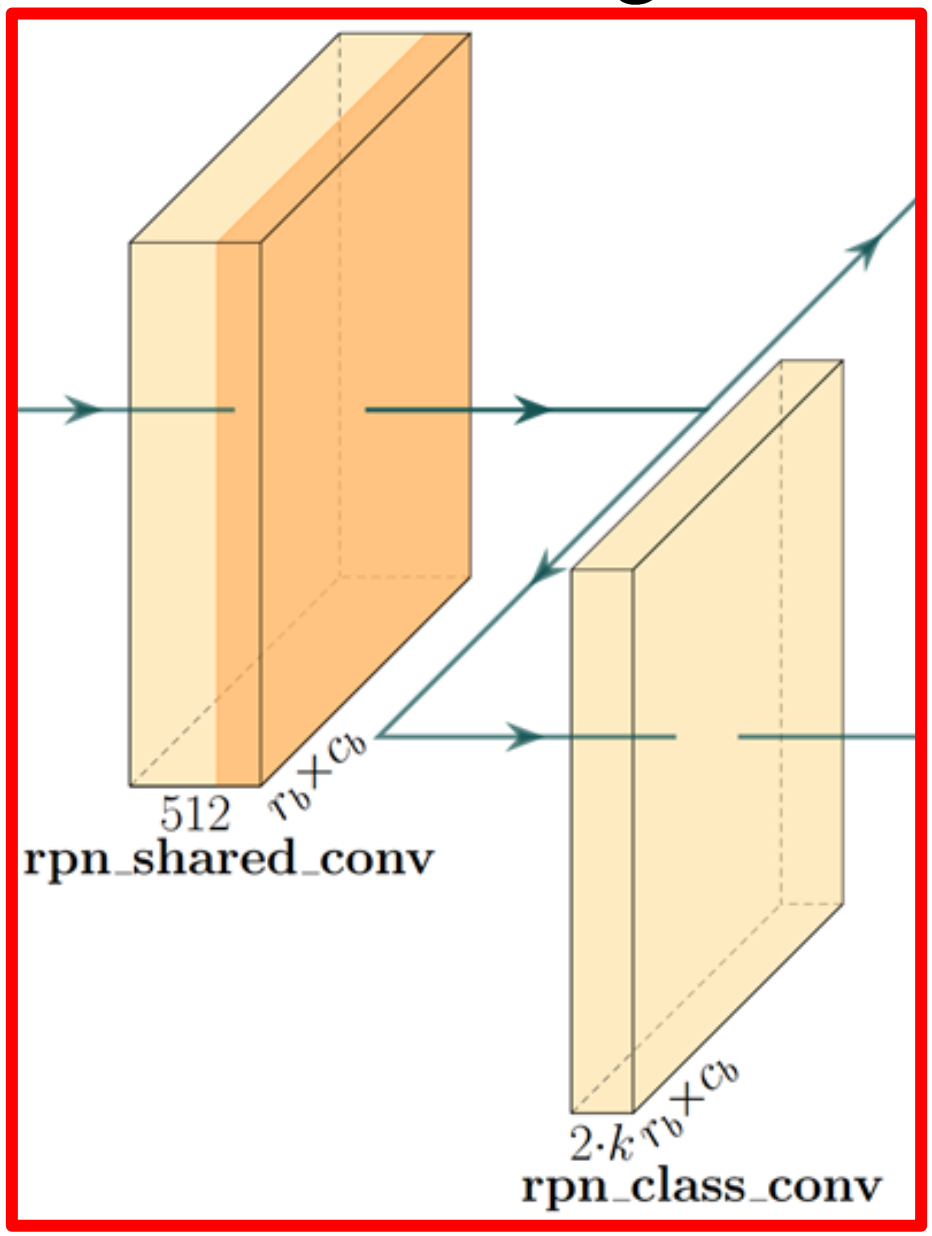
RPN Estimating k Anchors

- The **cls** (classification) **network** is trained to predict the ***object probability,*** i.e. **the probability for an anchor to contain and object** *[contains / does not contain]* $\rightarrow 2k$ probability estimates per each spatial location
- Made of a stack of 1×1 convolutional layers
- Each of these $\boldsymbol{k}$ **probability pairs** $[p, 1-p]$ corresponds to a specific anchor (having a specific dimension) **and expresses the probability for that anchor in that spatial location to contain *any object***

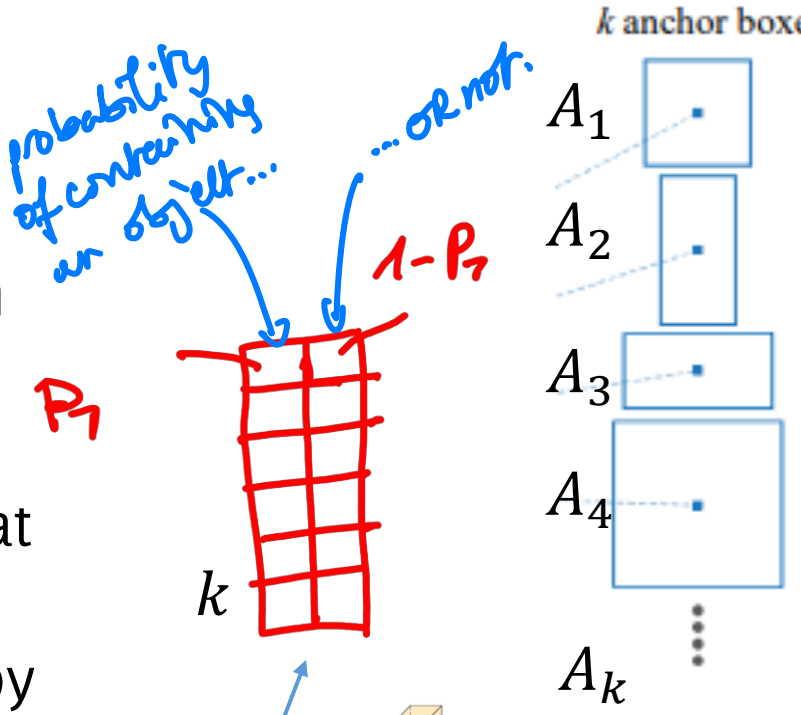


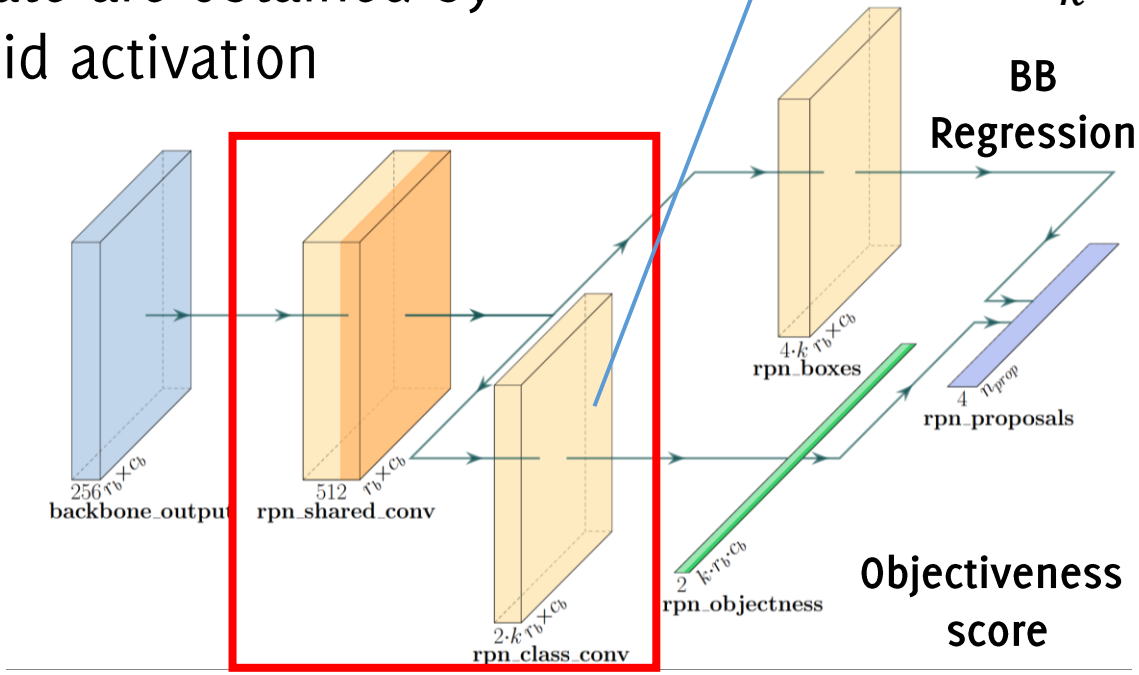


RPN Estimating k Anchors



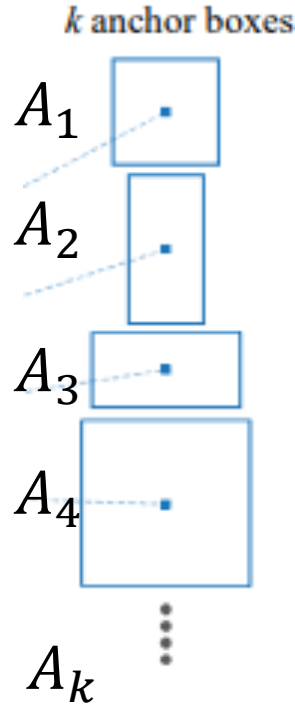
$2k$ probabilities per each spatial location. In each neuron we get an estimate of whether there is an object in that location for the anchor. Estimate are obtained by sigmoid activation

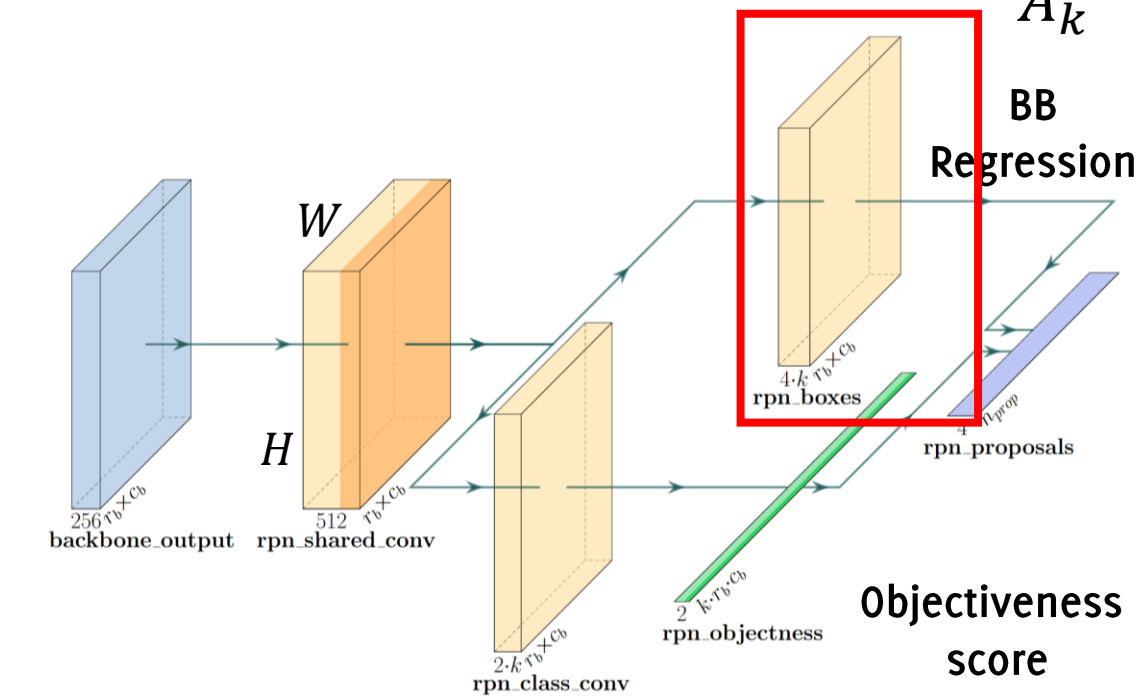




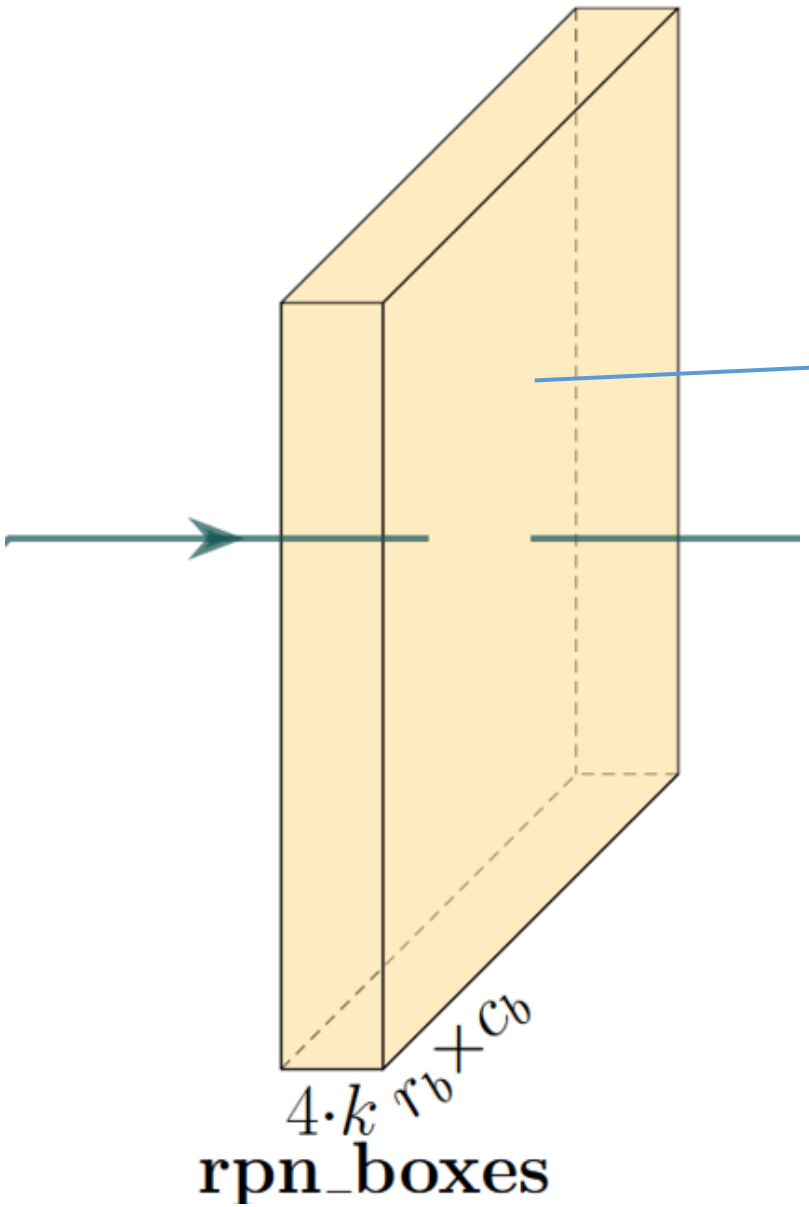
RPN Estimating k Anchors

- The **reg** (regression) **network is trained to adjust** each of the k anchors to better match object ground truth blob
  $\rightarrow 4k$ estimates for the «delta» over the 4 bounding box cooridnates (per each spatial location)
- Each of these k 4 –tuples **expresses the delta / refinements** for a specific anchor and are computed in a log measure to ease network convergence

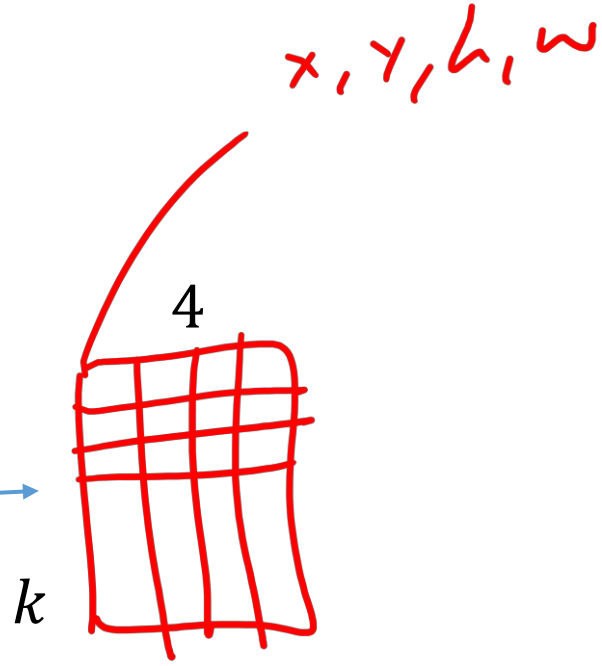


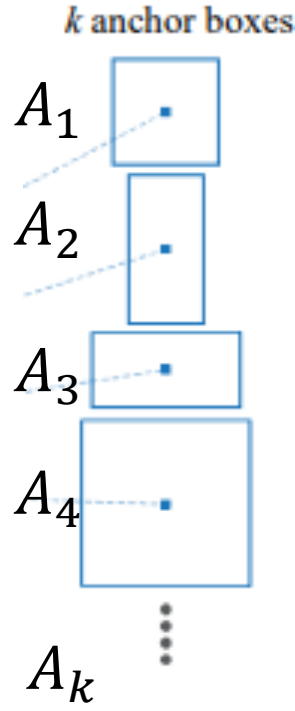


RPN Estimating k Anchors



BB Regression





$4k$ numbers predicted per each spatial location. In each neuron we get an estimate of the 4 bounding box correction factors for each anchor.

RPN output: the proposals

The RPN returns k **proposals for each location in the latent space**

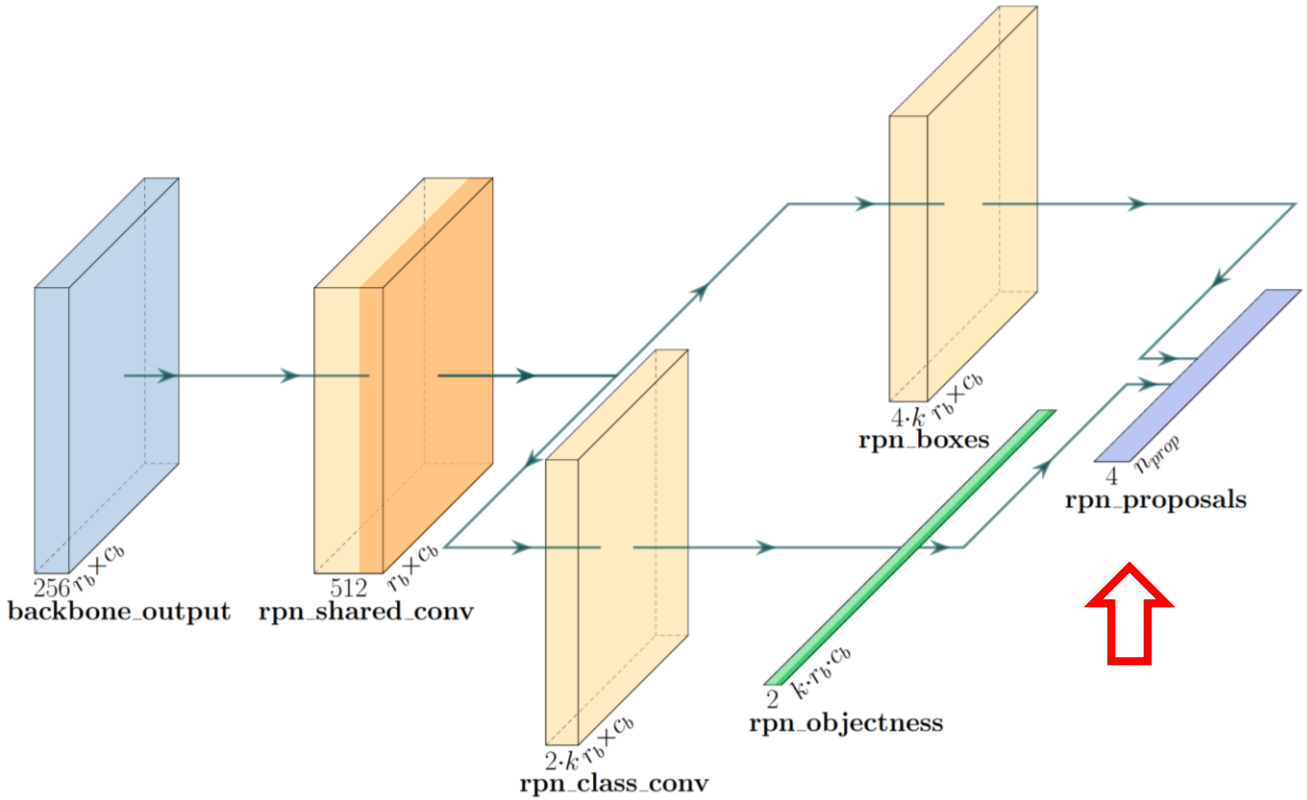
- $2k$ objectiveness scores
- $4k$ correction bounding boxes

Nonmaximum suppression: on the proposals to return only the first n_{prop} proposals

Proposals are selected analyzing:

- Non-maximum suppression based on the objectiveness score and the IoU considering corrected boxes
- Fixed threshold on the objectiveness score

Only the top n_{prop} proposals are propagated to the next layer



Faster R-CNN

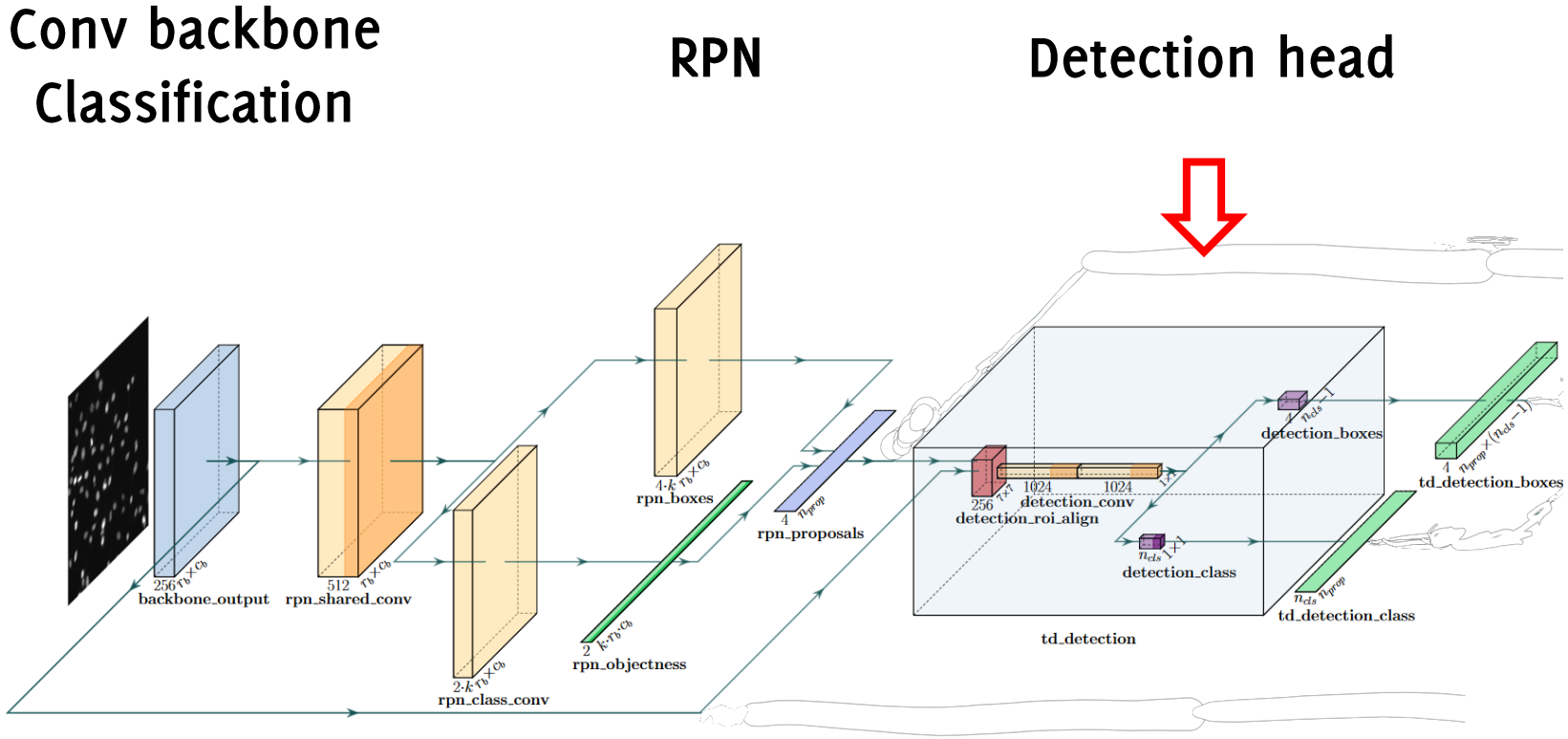
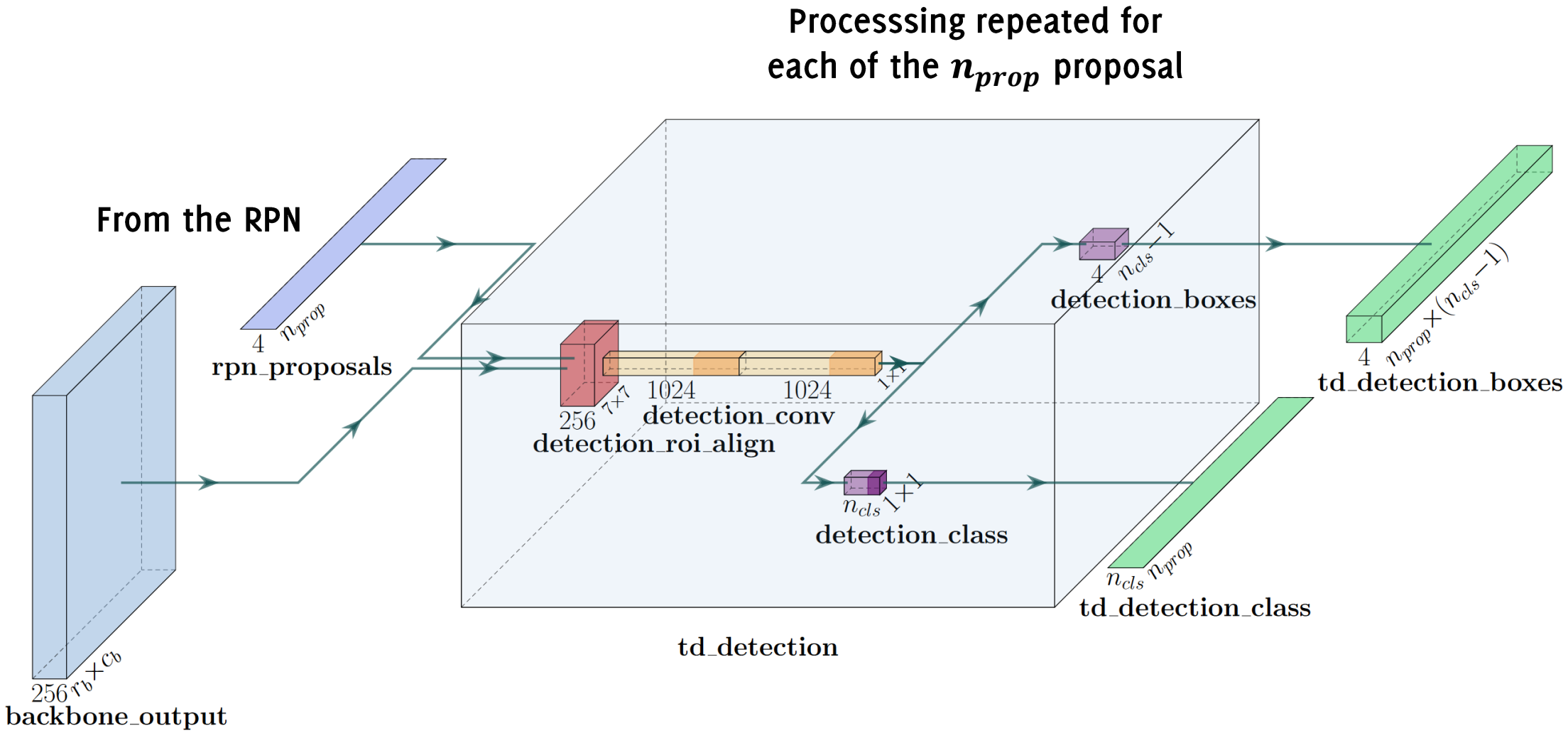


Image Credits Roberto Basla

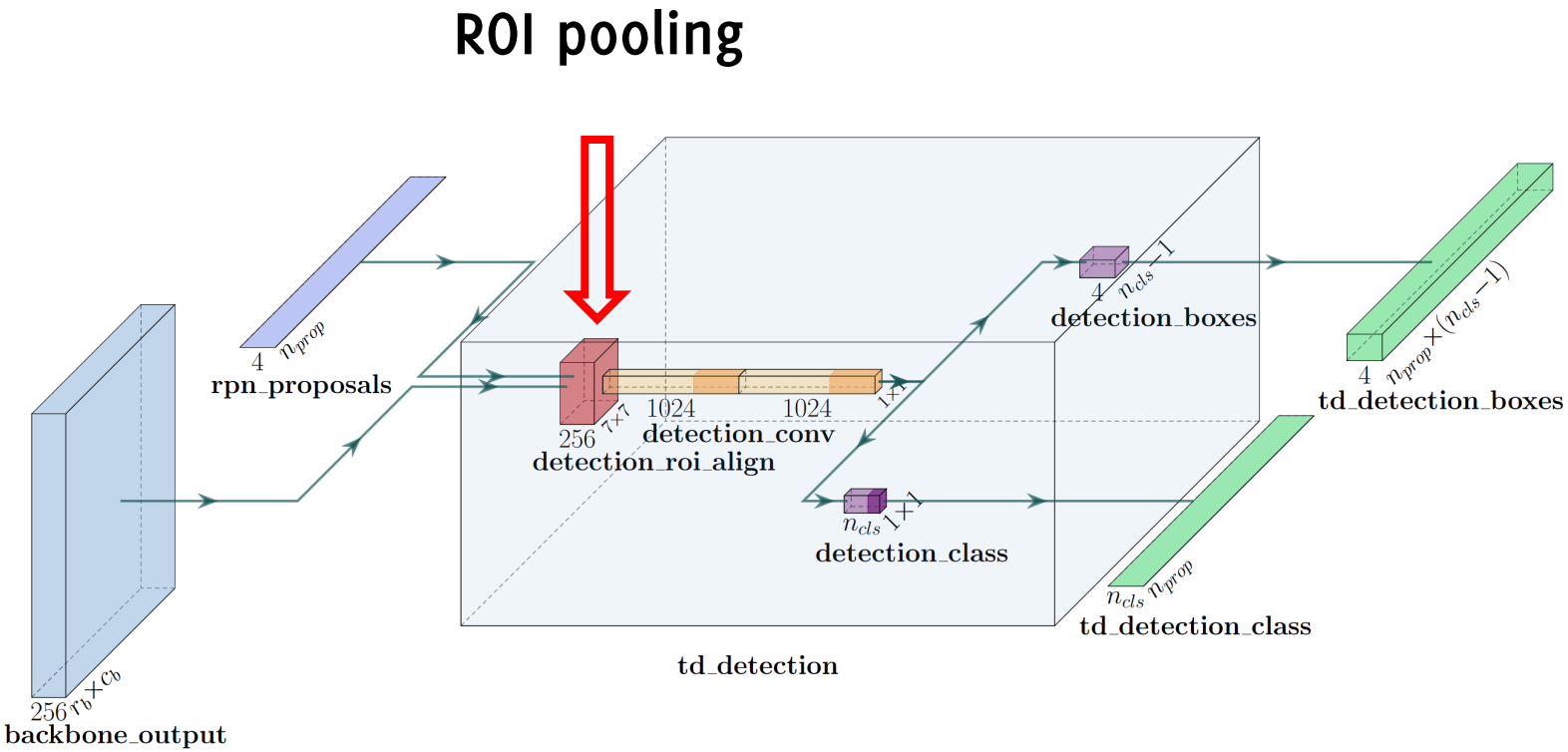
The detection head



From the CNN backbone

Faster R-CNN

RPN returns n_{prop} region proposals, thus replaces the region proposal algorithms (selective search) that are fed to the ROI pooling and then classified by the standard Fast-RCNN architecture.



Ren, Shaoqing, et al. "Faster r-cnn: Towards real-time object detection with region proposal networks." *NIPS 2015*

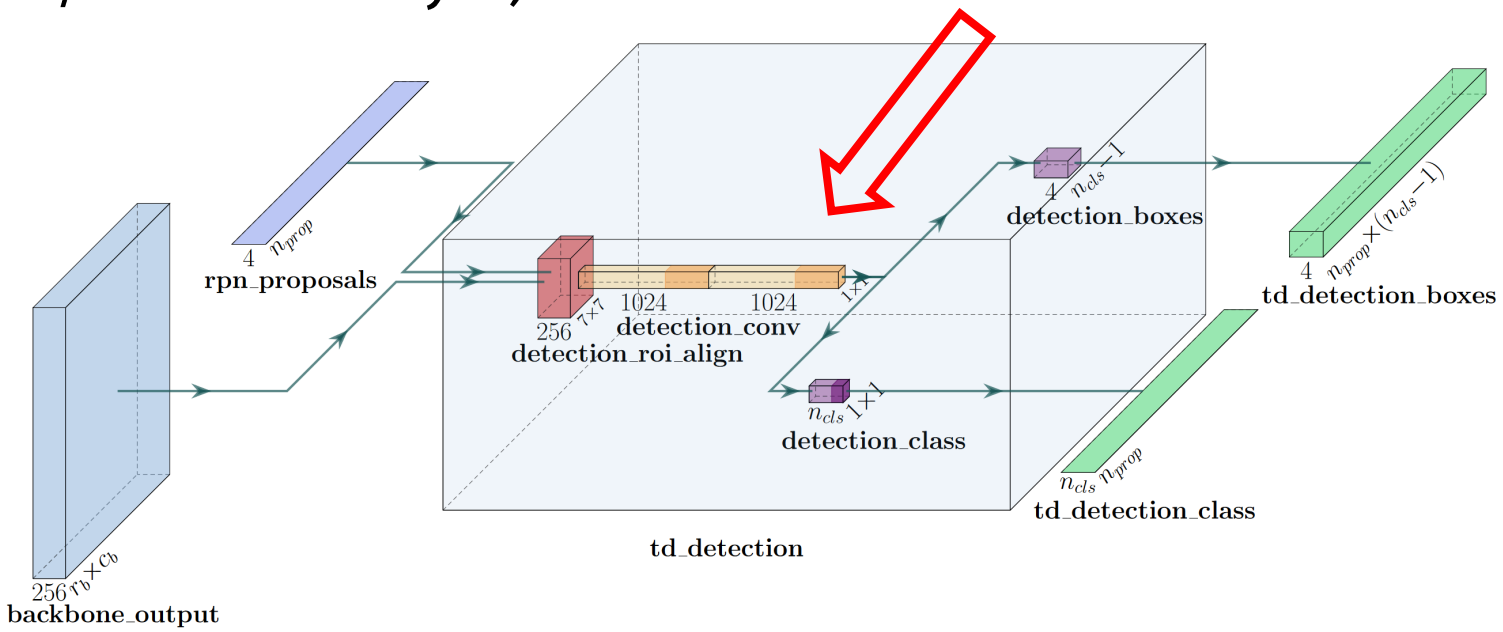
ROI Align and Pooling

This is meant to extract regions in the latent space referring to proposals

The features are then subsampled to a fixed size.

For each proposal there is a latent vector that is further processed by the detection head.

After some MLP/1x1 conv layer, it returns **ROI feature vector**



Ren, Shaoqing, et al. "Faster r-cnn: Towards real-time object detection with region proposal networks." *NIPS 2015*

RPN and the Anchor Shapes

- If you want to train the network to predict the objectiveness score and the offsets for different anchors, there is no need to design different RPN, bust just to **define different targets (associated to different anchor shape and size) when training the RPN.**
- You can in principle adopt **anchors having non-rectangular shapes** (e.g. ellipses, or simply rotated BBs)….
- However it can be difficult to compute IoU, which is instead extremely easy and fast in case of BBs… there are shapes that cannot be easily interesected (no closed form expression) to define the loss

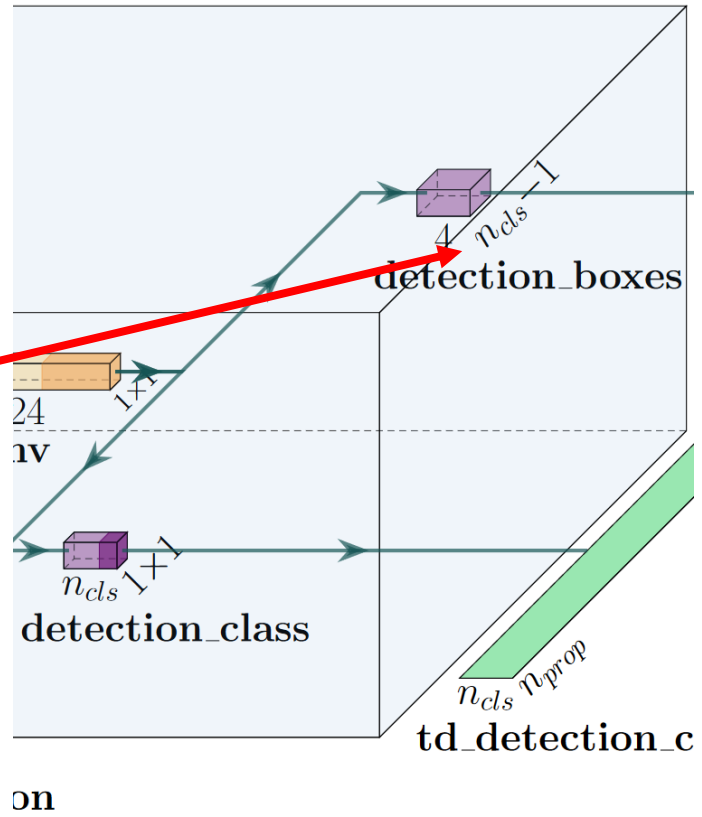
Ren, Shaoqing, et al. "Faster r-cnn: Towards real-time object detection with region proposal networks." *NIPS 2015*

Detection Head

The "detection head has two heads" and takes as input a single ROI feature vector.

It returns two outputs:

- L classification scores (class probabilities)
- $4 \times (L - 1)$ bounding box corrections



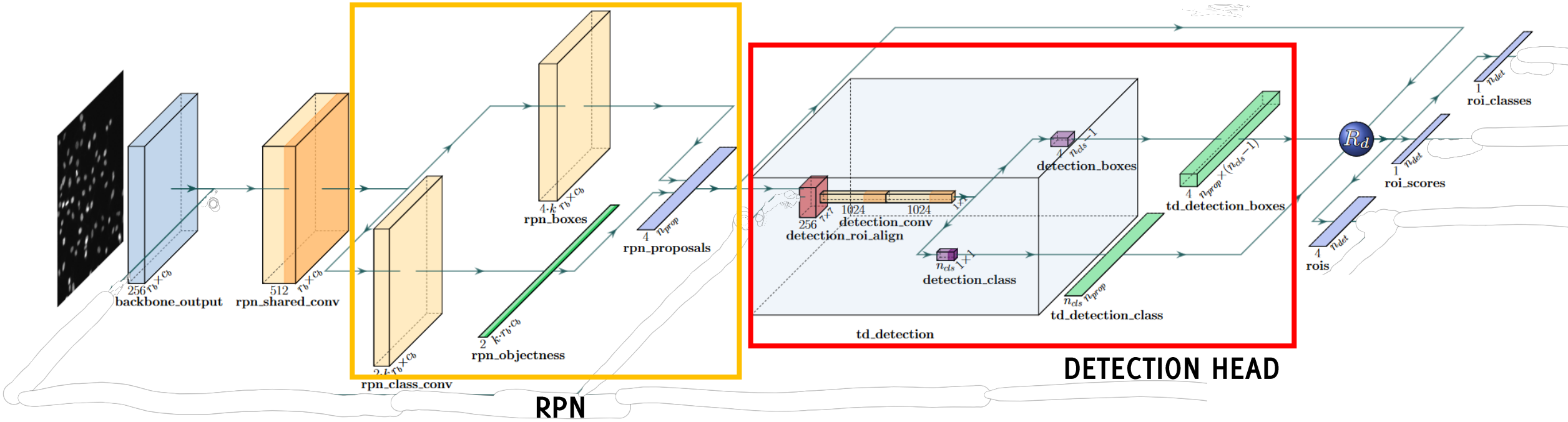
The classes include the "**background**" class for which we do not compute the bounding box corrections

The Faster output corresponds to the predicted objects with large confidence (fewer than the n proposals)

Ren, Shaoqing, et al. "Faster r-cnn: Towards real-time object detection with region proposal networks." *NIPS 2015*

Faster R-CNN Training

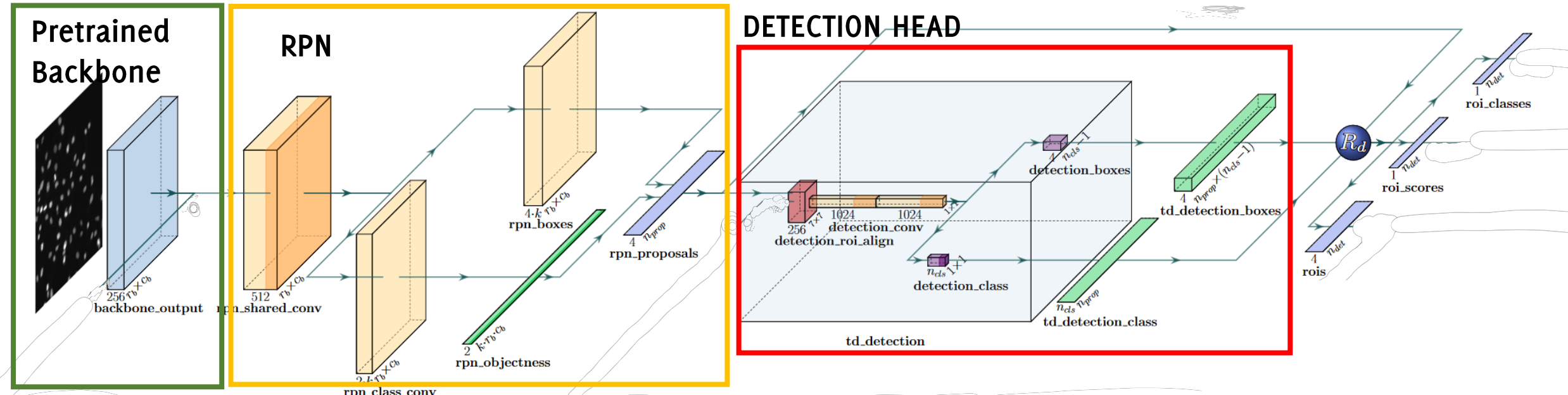
- **Training** now involves 4 losses, in a weighted sum:
  - RPN classify object/non object
  - RPN regression coordinates
  - Final classification score
  - Final BB coordinates
- During training, object/non object ground truth is defined by measuring the overlap with annotated BB
- The loss include a term of **final BB coordinates** computed form detection head



Faster R-CNN Training

Training procedure

1. **Train RPN** keeping backbone network frozen and training only RPN layers. This ignores object classes but **just bounding box locations** (Multi-task loss cls + reg)
2. **Train Fast-RCNN** using proposals from **RPN trained before.** Fine tune the whole Fast-RCNN including the backbone
3. **Fine tune the RPN in cascade of the new backbone**
4. **Freeze backbone and RPN and fine tune only the last layers of the Faster R-CNN**



Faster R-CNN

At test time,

- Take the top ~ 300 anchors according to their object scores
- Consider the refined bounding box location of these 300 anchors
- These are the ROI to be fed to a Fast R-CNN
- Classify each ROI and provide the non-background ones as output

Faster R-CNN provides as output to each image a **set of BB** with their **classifier posterior**

The network becomes much faster (0.2s test time per image)

Ren, Shaoqing, et al. "Faster r-cnn: Towards real-time object detection with region proposal networks." *NIPS 2015*

Faster R-CNN

⚠ It's still a **two stage detector**

First stage: SELECTING PROPOSALS.

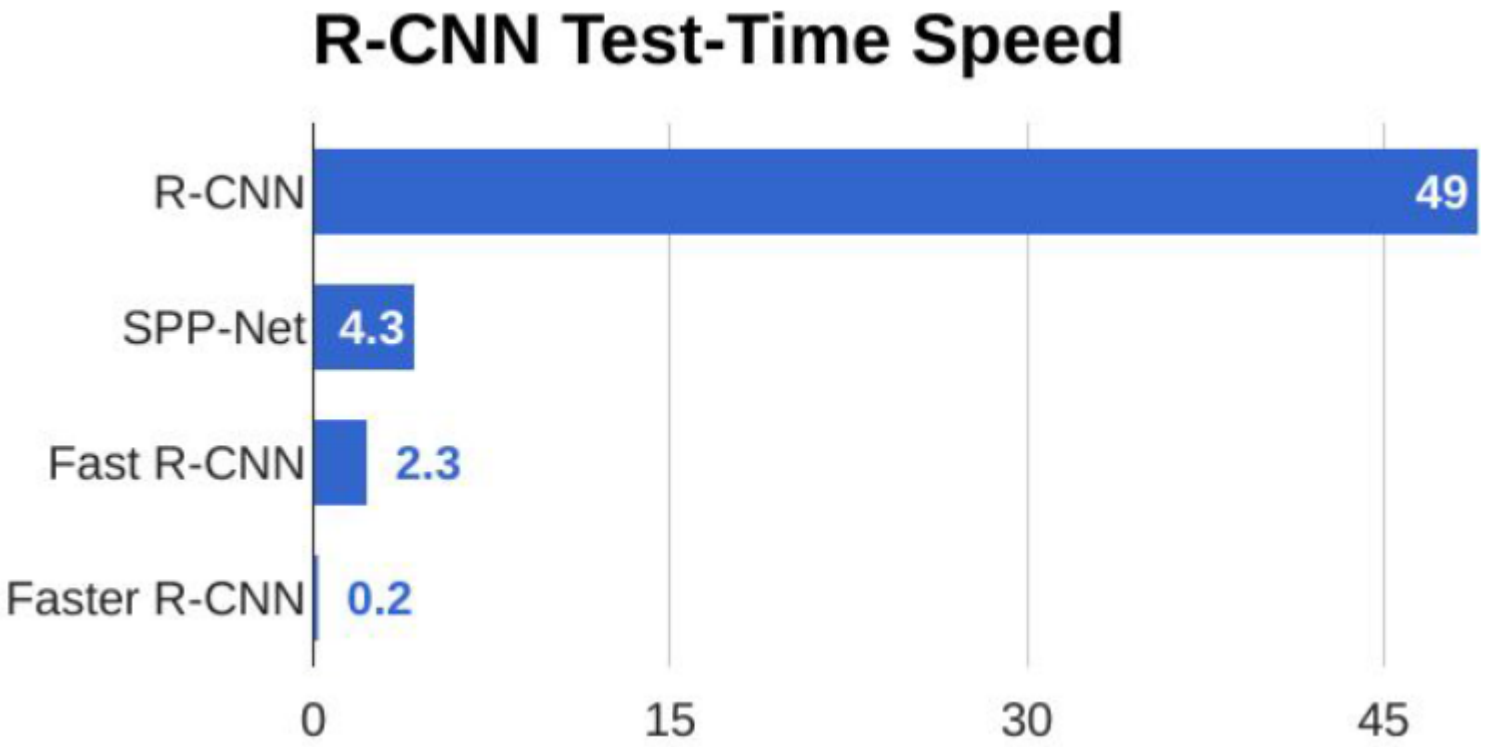
- run a **backbone network** (e.g. VGG16) to **extract featues**
- run the **Region Proposal Network** to **estimate** ~ 300 **ROI**

Second stage (the same as in Fast R-CNN): CLASSIFYING + BB.

- **Crop Features** through ROI pooling (with alignment)
- **Predict object class** using FC + softmax
- **Predict bounding box offset** to improve localization using FC + softmax

Ren, Shaoqing, et al. "Faster r-cnn: Towards real-time object detection with region proposal networks." *NIPS 2015*

Faster R-CNN



Ren, Shaoqing, et al. "Faster r-cnn: Towards real-time object detection with region proposal networks." *NIPS 2015*

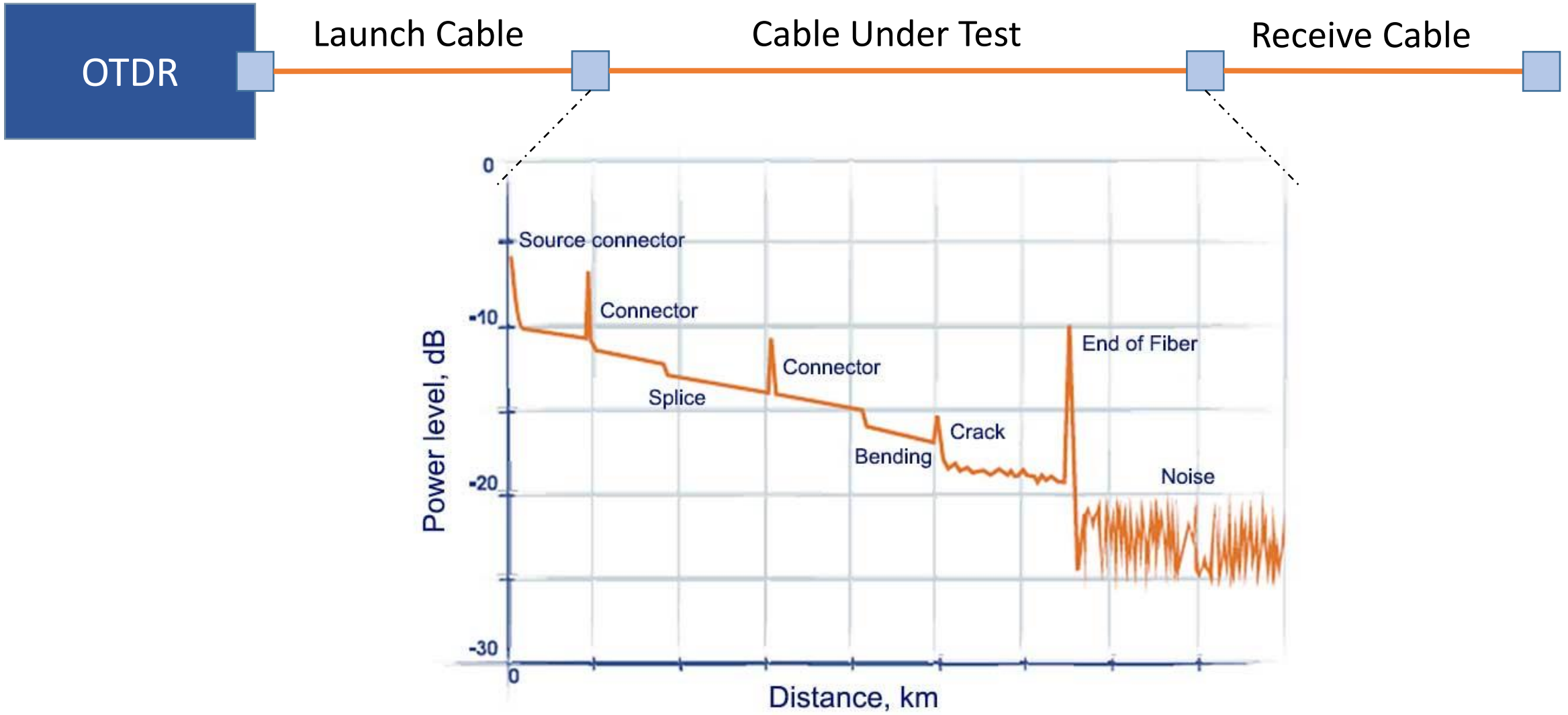
In collaboration with

An industrial application

Credits Davide Rutigliano

In collaboration with

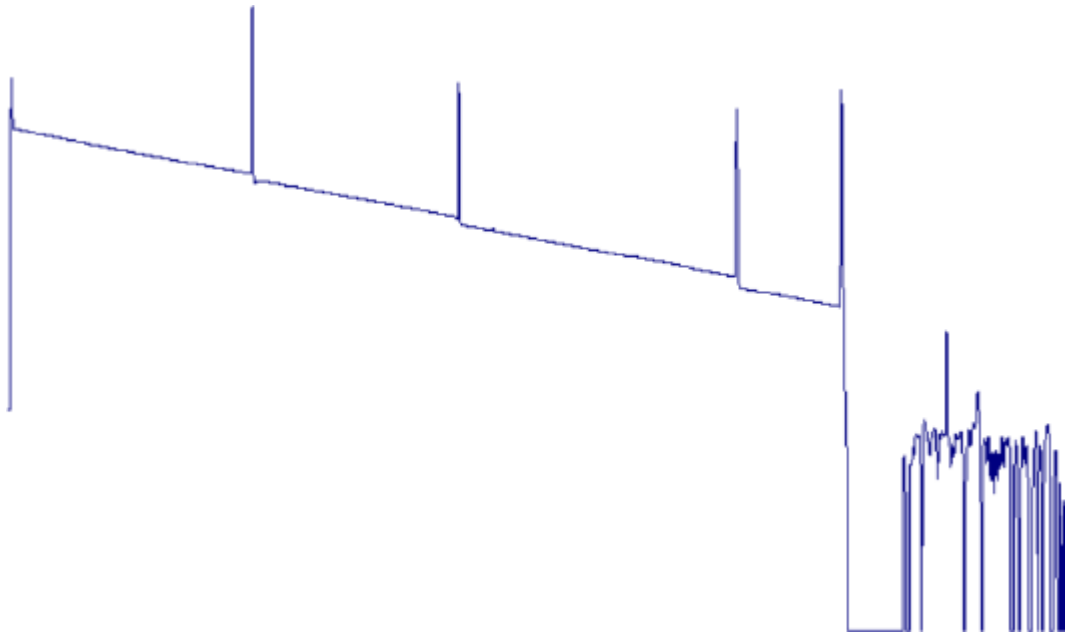
Event Detection in Optical Signals



A. M. Rizzo, L. Magri, D. Rutigliano, P. Invernizzi, E. Sozio, C. Alippi, S. Binetti, G. Boracchi, *"Known and Unknown Event Detection in OTDR Traces by Deep Learning Networks"*, NCAA, Springer 2021

Event Detection in Optical Signals

In collaboration with



OTDR trace: events indicate problems over a span of optical fiber

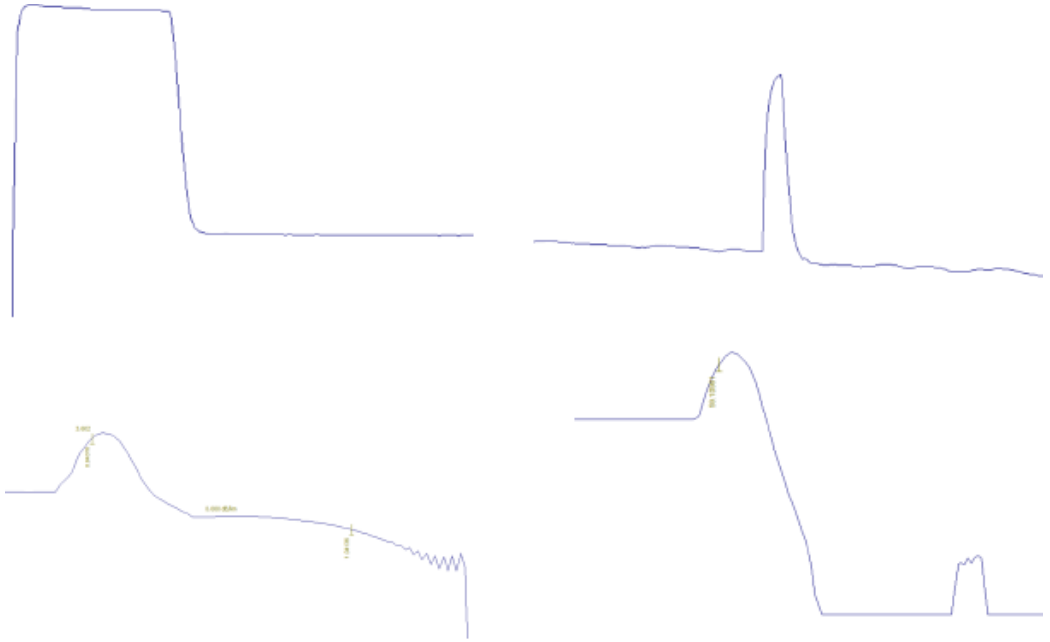


Fig. 1 (b): OTDR events examples: face-plate, pass-through, fiber-cut, fiber-end

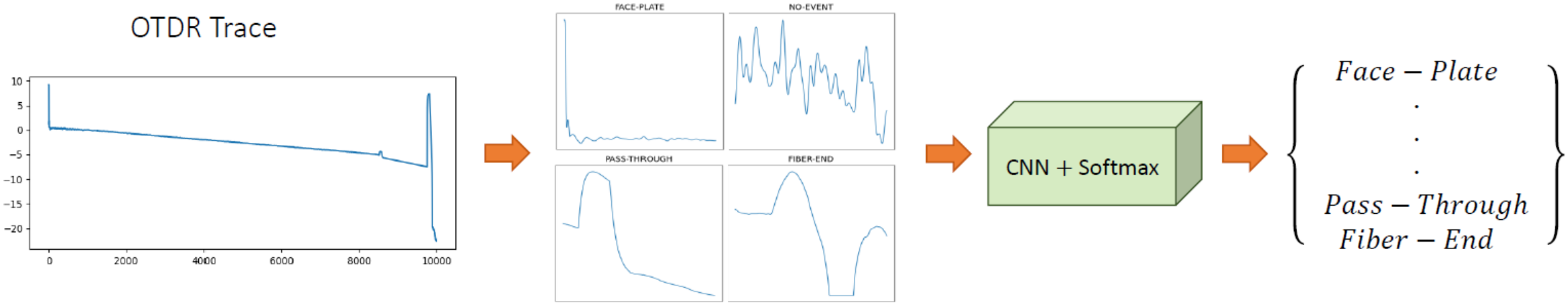
A. M. Rizzo, L. Magri, D. Rutigliano, P. Invernizzi, E. Sozio, C. Alippi, S. Binetti, G. Boracchi, *"Known and Unknown Event Detection in OTDR Traces by Deep Learning Networks"*, NCAA, Springer 2021

Event Detection in Optical Signals

In collaboration with

CISCO

Train a classification network over fixed windows (300 samples)



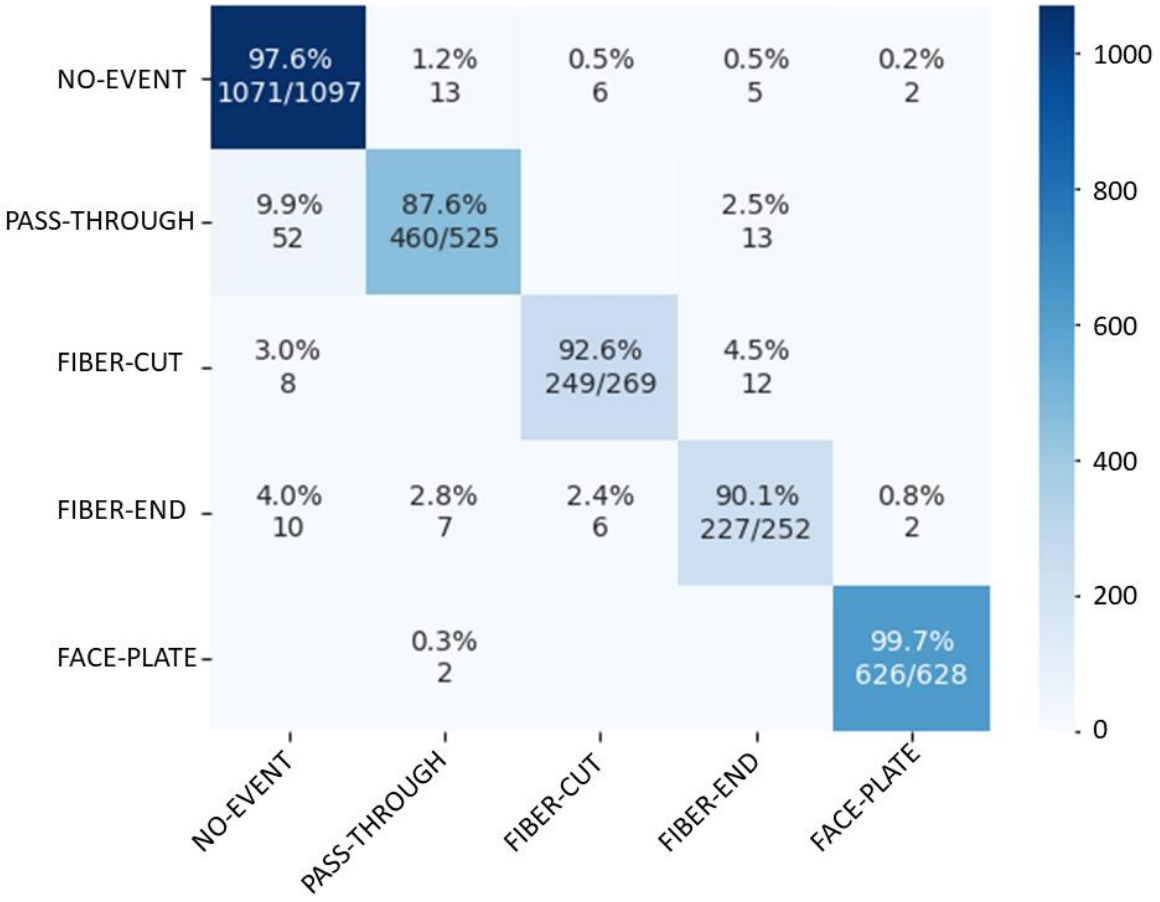
A. M. Rizzo, L. Magri, D. Rutigliano, P. Invernizzi, E. Sozio, C. Alippi, S. Binetti, G. Boracchi, *"Known and Unknown Event Detection in OTDR Traces by Deep Learning Networks"*, NCAA, Springer 2021

Event Detection in Optical Signals

In collaboration with

CISCO

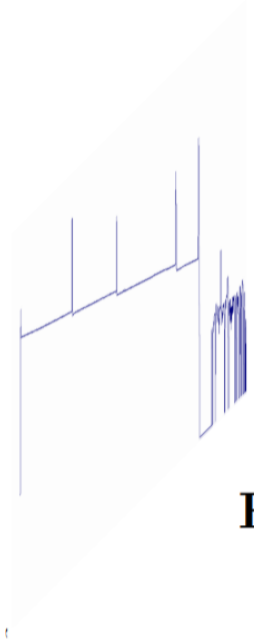
- Requires an additional step to split OTDR traces into windows
- Limited to single-scale events resolution
- Works under the assumption that each window includes at most one event
- Does not consider events position
- Does not share computations

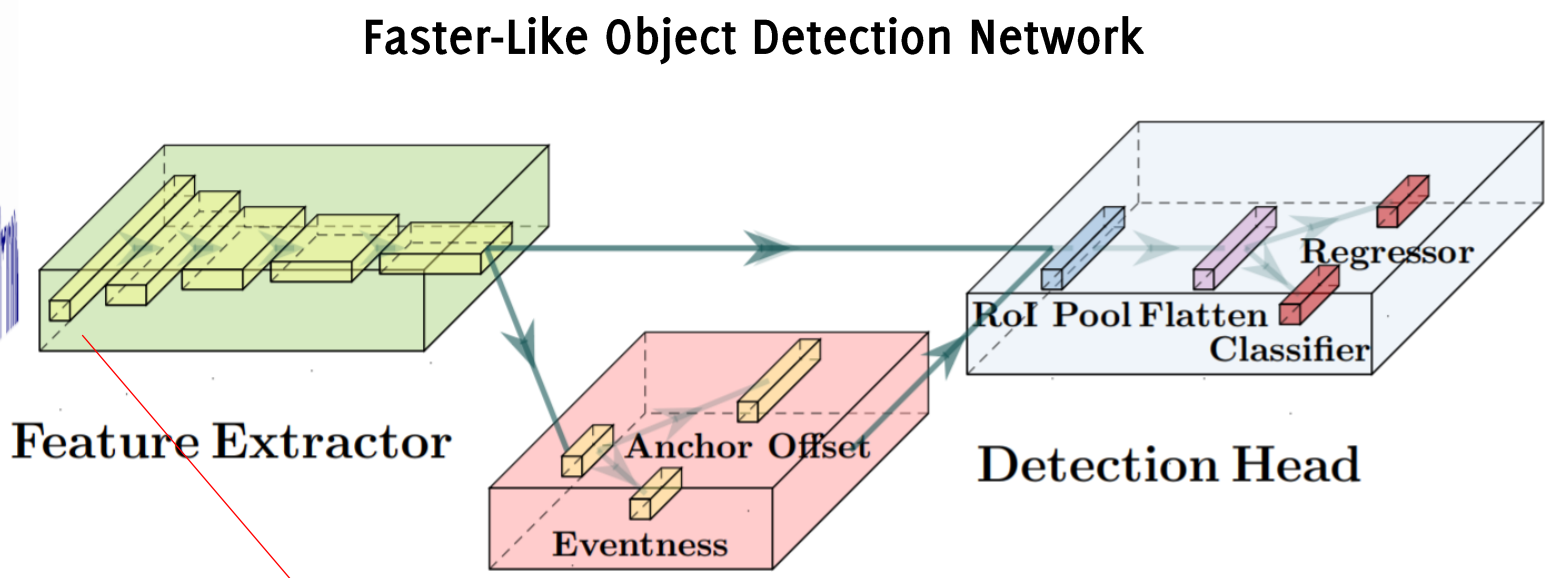


A. M. Rizzo, L. Magri, D. Rutigliano, P. Invernizzi, E. Sozio, C. Alippi, S. Binetti, G. Boracchi, *"Known and Unknown Event Detection in OTDR Traces by Deep Learning Networks"*, NCAA, Springer 2021

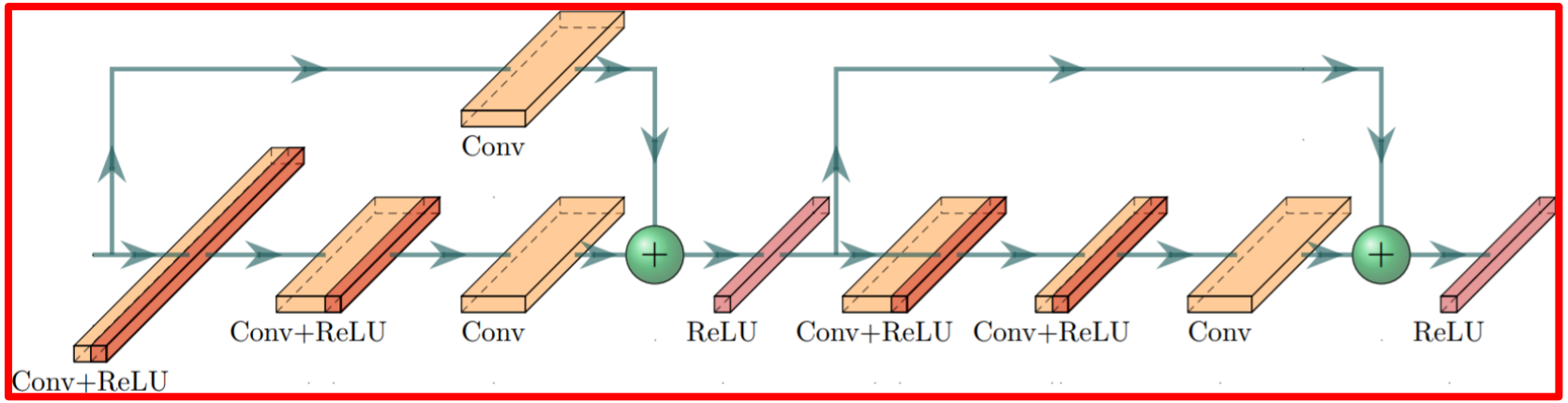
Event Detection in Optical Signals

In collaboration with



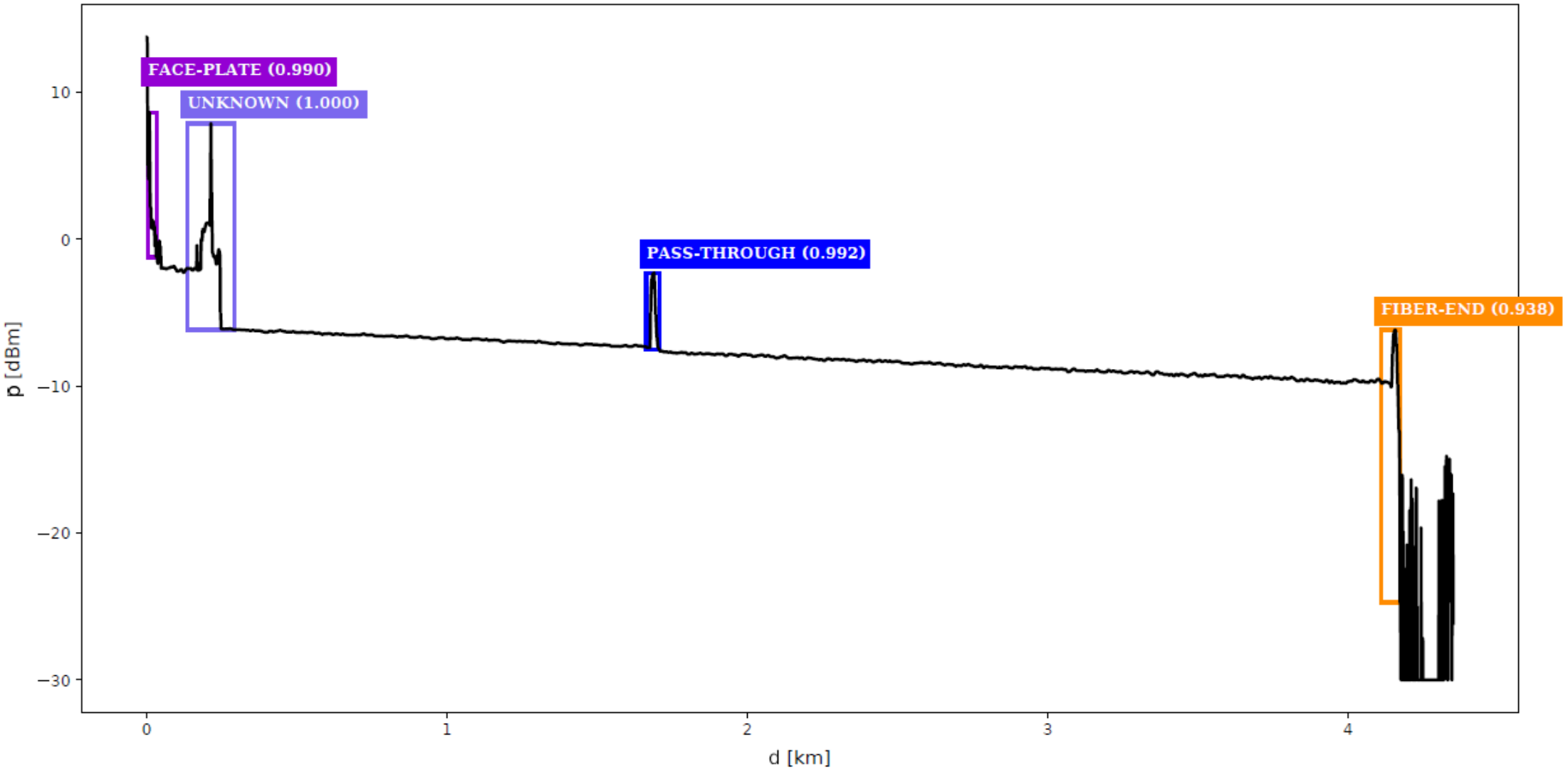


Block of the Feature Extraction Network



A.M. Rizzo, L. Magri1, D. Rutigliano, P. Invernizzi, E. Sozio, C. Alippi, S. Binetti, G. Boracchi *"Known and Unknown Event Detection in OTDR Traces by Deep Learning Networks"* NCAA, Springer

Able to detect both known and unknown events



A.M. Rizzo, L. Magri1, D. Rutigliano, P. Invernizzi, E. Sozio, C. Alippi, S. Binetti, G. Boracchi *"Known and Unknown Event Detection in OTDR Traces by Deep Learning Networks"* NCAA, Springer

You Only Look Once: Unified, Real-Time Object Detection

Joseph Redmon*, Santosh Divvala*†, Ross Girshick¶, Ali Farhadi*†

University of Washington*, Allen Institute for AI†, Facebook AI Research¶

http://pjreddie.com/yolo/

Redmon, Joseph, et al. "You only look once: Unified, real-time object detection." CVPR 2016.

YOLO/SSD

R-CNN methods are based on region proposals

There are also region-free methods, like:

YOLO: You Only Look Once

SSD: Single Shot Detectors

Redmon, Joseph, et al. "You only look once: Unified, real-time object detection." CVPR 2016.

The rationale

Detection networks are indeed a pipeline of multiple steps.

In particular, **region-based methods make it necessary to have two steps** during inference

This can be slow to run and hard to optimize, because each individual component must be trained separately.

In Yolo *"**we reframe the object detection as a single regression problem**, straight from image pixels to bounding box coordinates and class probabilities"*

And solve these regression problems **all at once**, with a large CNN

Redmon, Joseph, et al. "You only look once: Unified, real-time object detection." *CVPR* 2016.
Liu, Wei, et al. "SSD: Single shot multibox detector." *European conference on computer vision*. Springer, Cham, 2016.

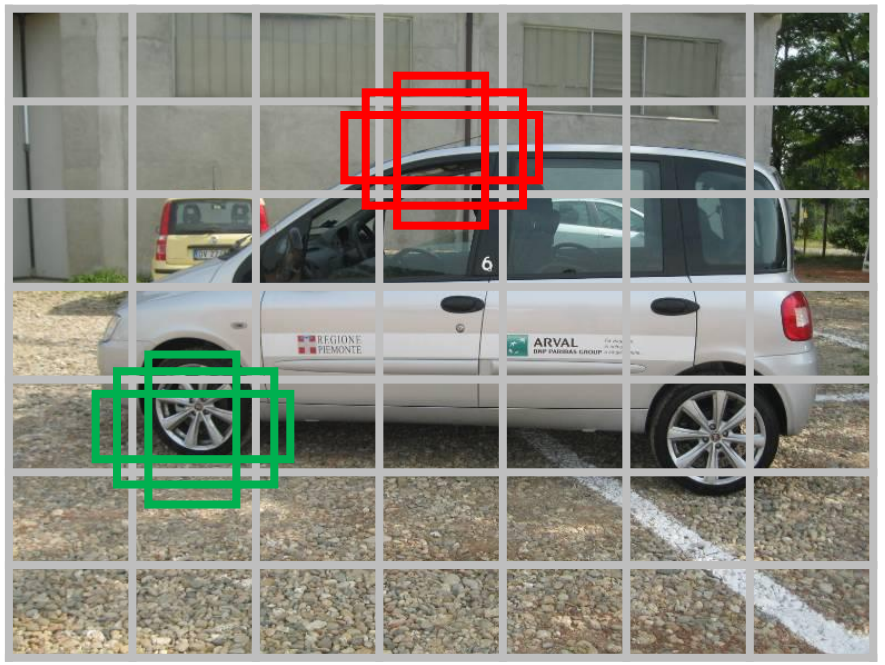
YOLO

1. divide the image in a coarse grid (e.g. 7x7)

Redmon, Joseph, et al. "You only look once: Unified, real-time object detection." *CVPR* 2016.

YOLO

1. divide the image in a coarse grid (e.g. 7x7)
2. each grid cell contains ***B anchors*** *(base bounding box)* associated



Redmon, Joseph, et al. "You only look once: Unified, real-time object detection." *CVPR* 2016.

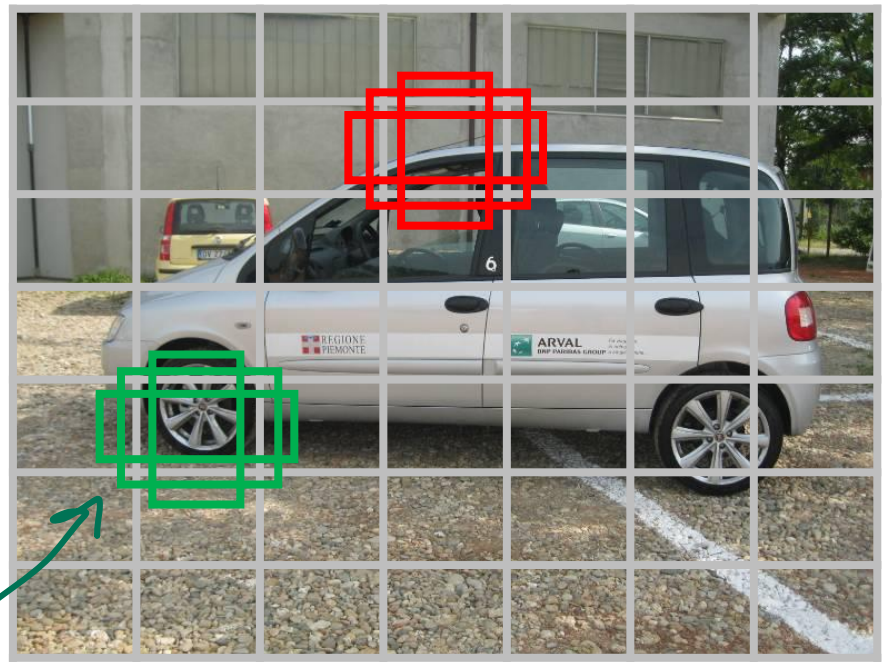
YOLO

3. For each cell and anchor **we predict:**
   - The **offset of the base bounding box**, to better match the object:
     $(dx, dy, dh, dw, objectness\_score)$
   - The **classification score of the base-bounding box** over the C considered categories (including background)

So, the output of the network has dimension

$$7 \times 7 \times B \times (5 + C)$$

"B basic bounding boxes"



Redmon, Joseph, et al. "You only look once: Unified, real-time object detection." *CVPR* 2016.

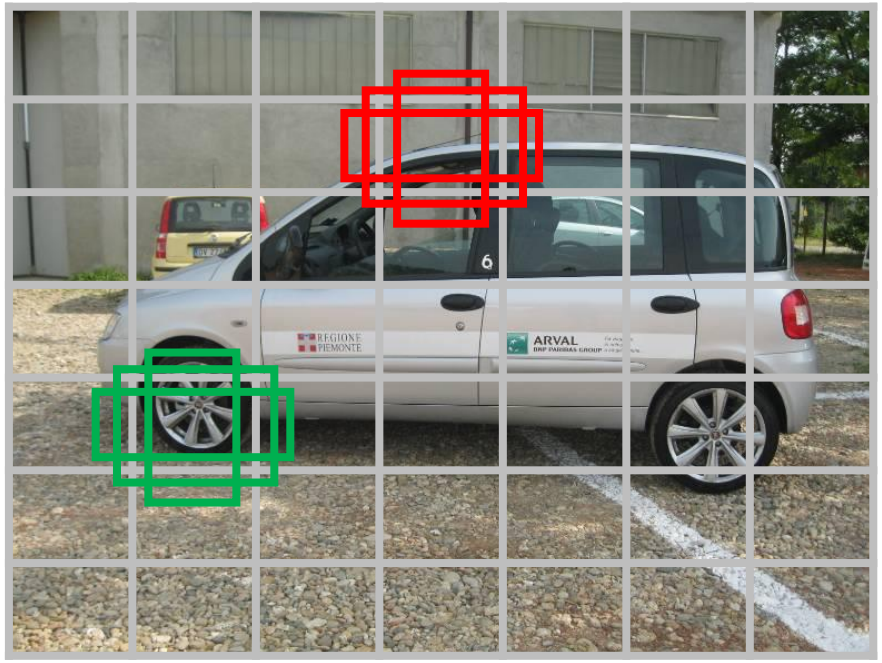
YOLO

The whole **prediction is performed in a single forward pass** over the image, **by a single convolutional network**

Training this network is sort of **tricky to assess the** loss (matched / not matched)

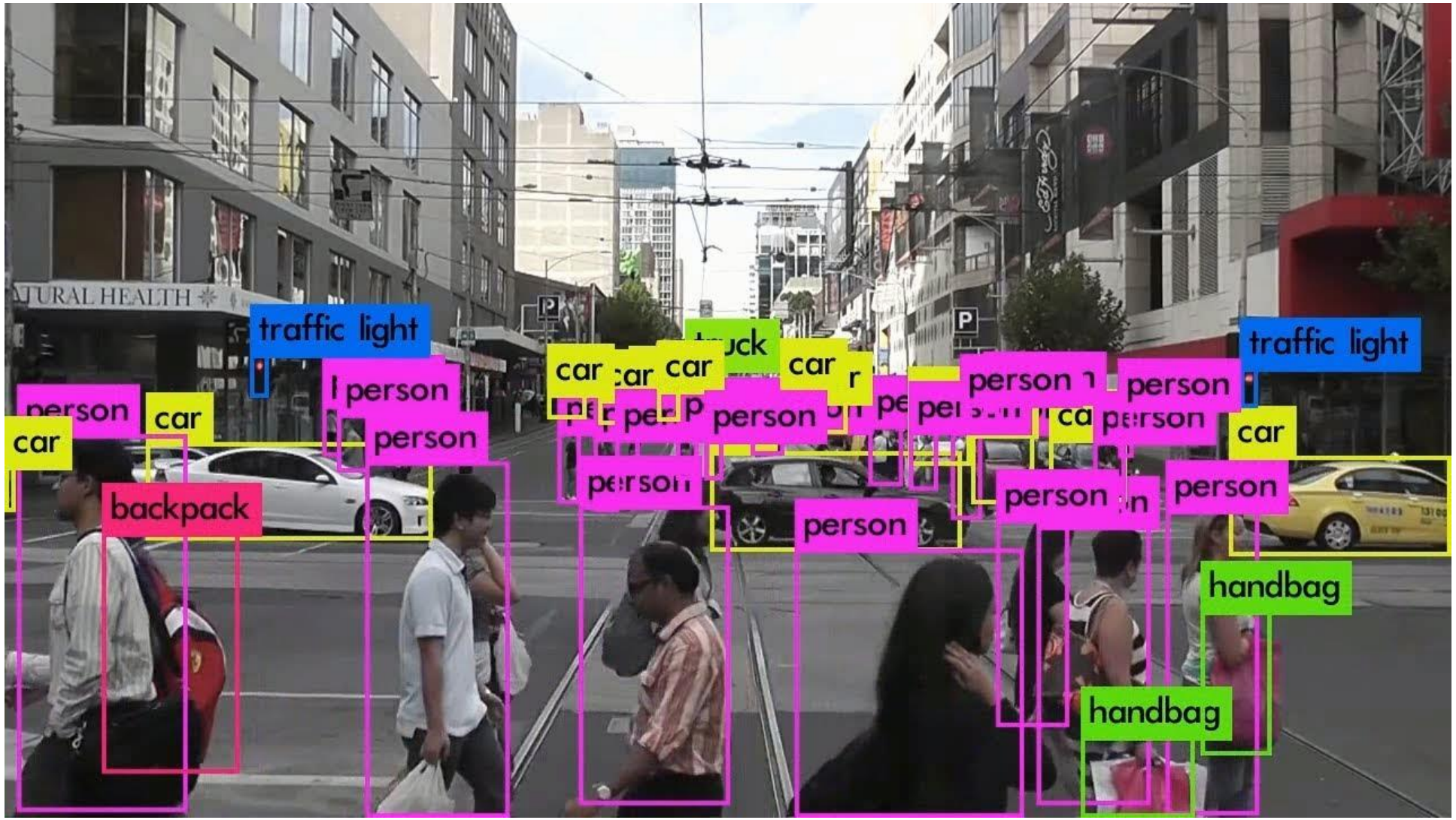
YOLO/SSD shares a similar ground of the RPN used in Faster R-CCN

Typically, networks based on region-proposals are more accurate, single shot detectors are faster but less accurate



Redmon, Joseph, et al. "You only look once: Unified, real-time object detection." *CVPR* 2016.

Object Detection



Redmon, J., & Farhadi, A. (2018). Yolov3: An incremental improvement. arXiv preprint arXiv:1804.02767.

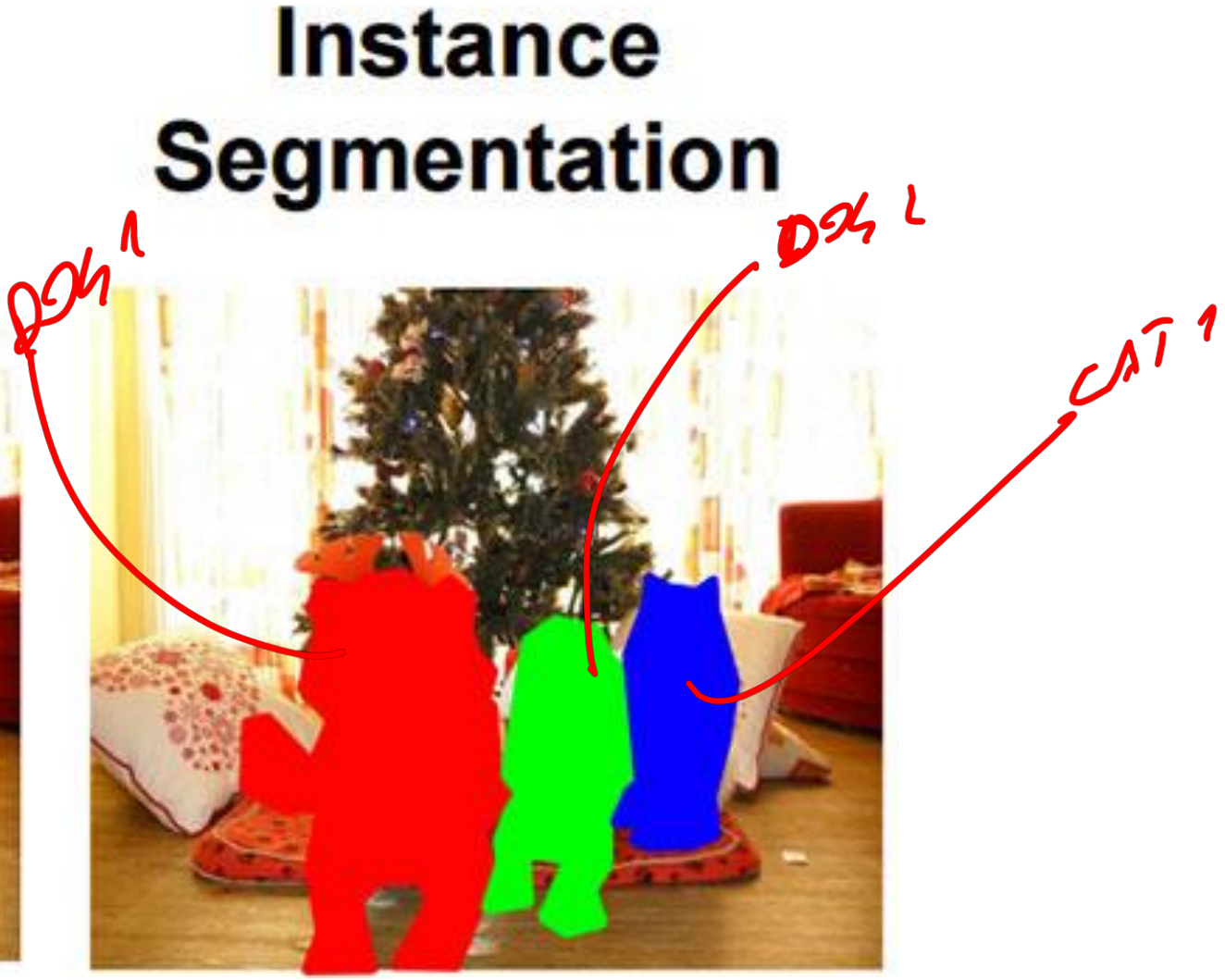
Do it yourself!

https://colab.research.google.com/drive/1chjXM6ckE1s-xoZZjGG_Q0dmlCeBPzPR

Perform inference using a pre-trained object detection network

Instance Segmentation

Object Detection vs Instance Segmentation



Instance Segmentation, the problem

Assign to an input image I:

- **multiple** labels $\{l_i\}$ from a fixed set of categories $\Lambda = \{$"wheel", "cars", ..., "castle", "baboon"$\}$, each corresponding to **an instance of that object**
- the coordinates $\{(x, y, h, w)_i\}$ of the **bounding box enclosing each object**
- the **set of pixels** S in each bounding box corresponding to that label

$$I \rightarrow \{(x, y, h, w, l, S)_1, \dots, (x, y, h, w, l, S)_N\}$$

Mask R-CNN

Kaiming He Georgia Gkioxari Piotr Dollár Ross Girshick
Facebook AI Research (FAIR)

He, K., Gkioxari, G., Dollár, P., & Girshick, R. *"Mask R-CNN"*. ICCV 2017

Instance Segmentation

It combines the challenges of:

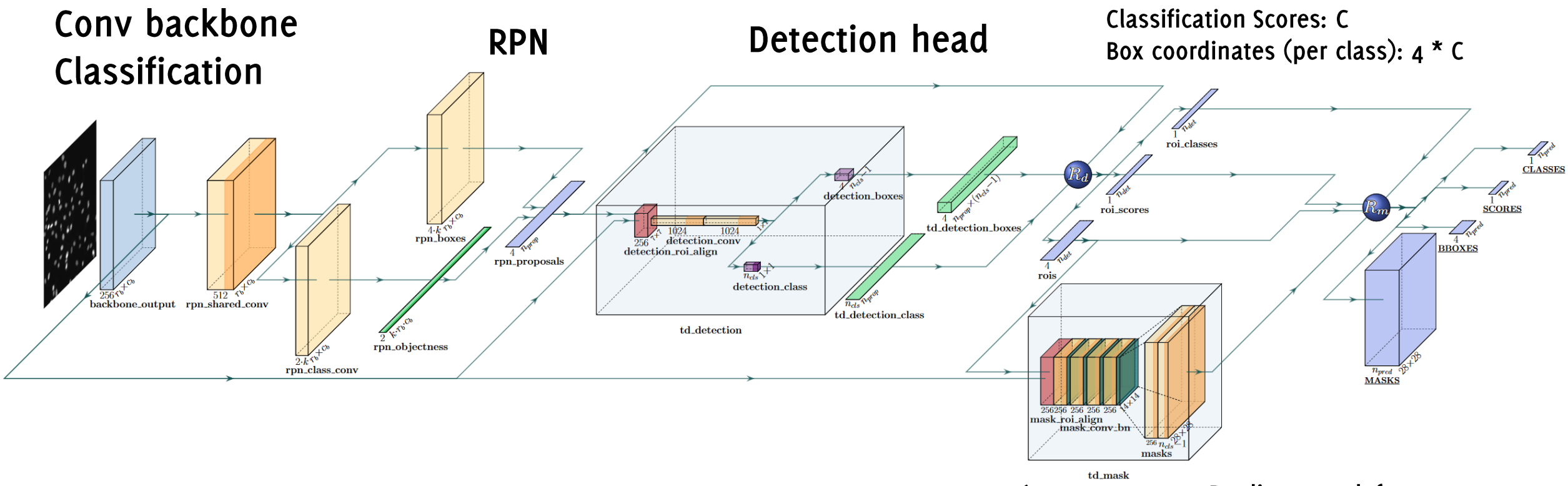
- **Object detection** (multiple instances present in the image)
- **Semantic segmentation** (associate a label to each pixel) separating each object instance

Mask R-CNN

Color the exact pixels ↳ mask.

As in Faster R-CNN, each ROI is classified and the bounding boxes are estimated

Semantic segmentation inside each ROI

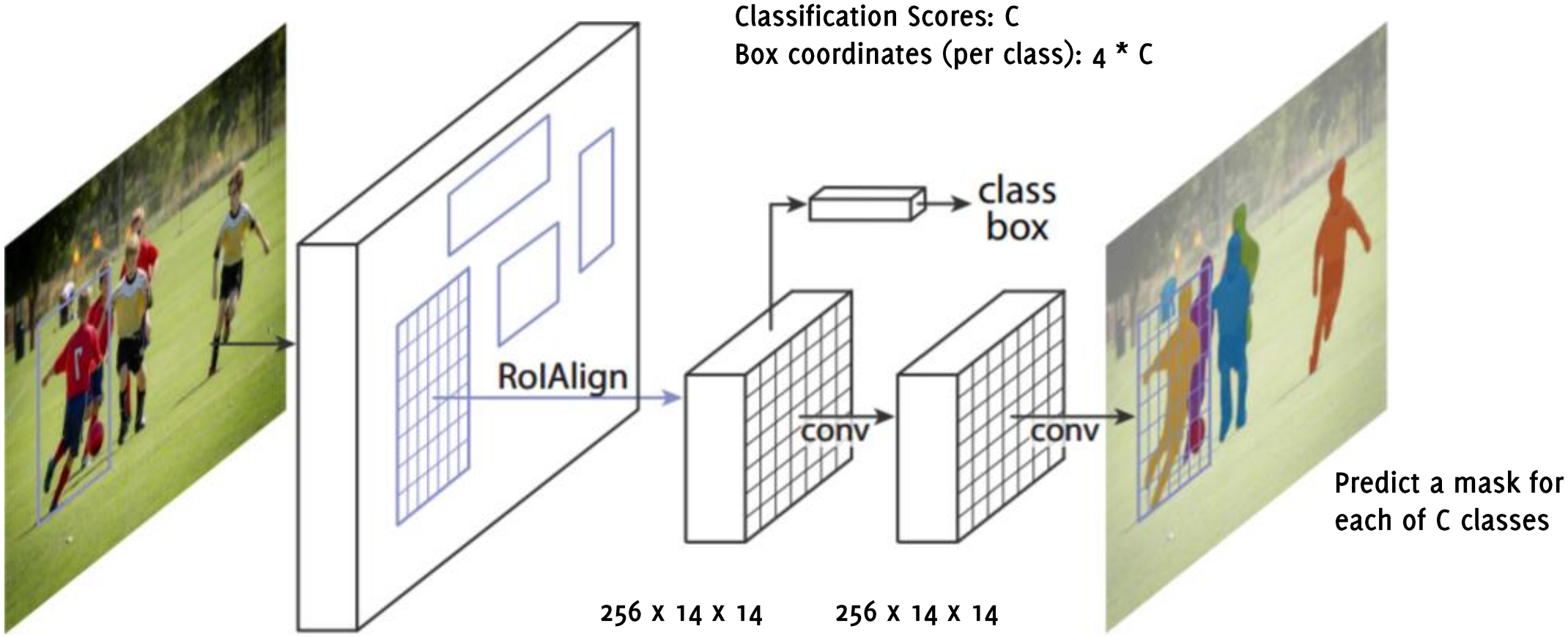


He, K., Gkioxari, G., Dollár, P., & Girshick, R. *"Mask R-CNN"*. ICCV 2017

Mask R-CNN

As in Faster R-CNN, each ROI is classified and the bounding boxes are estimated

Semantic segmentation inside each ROI

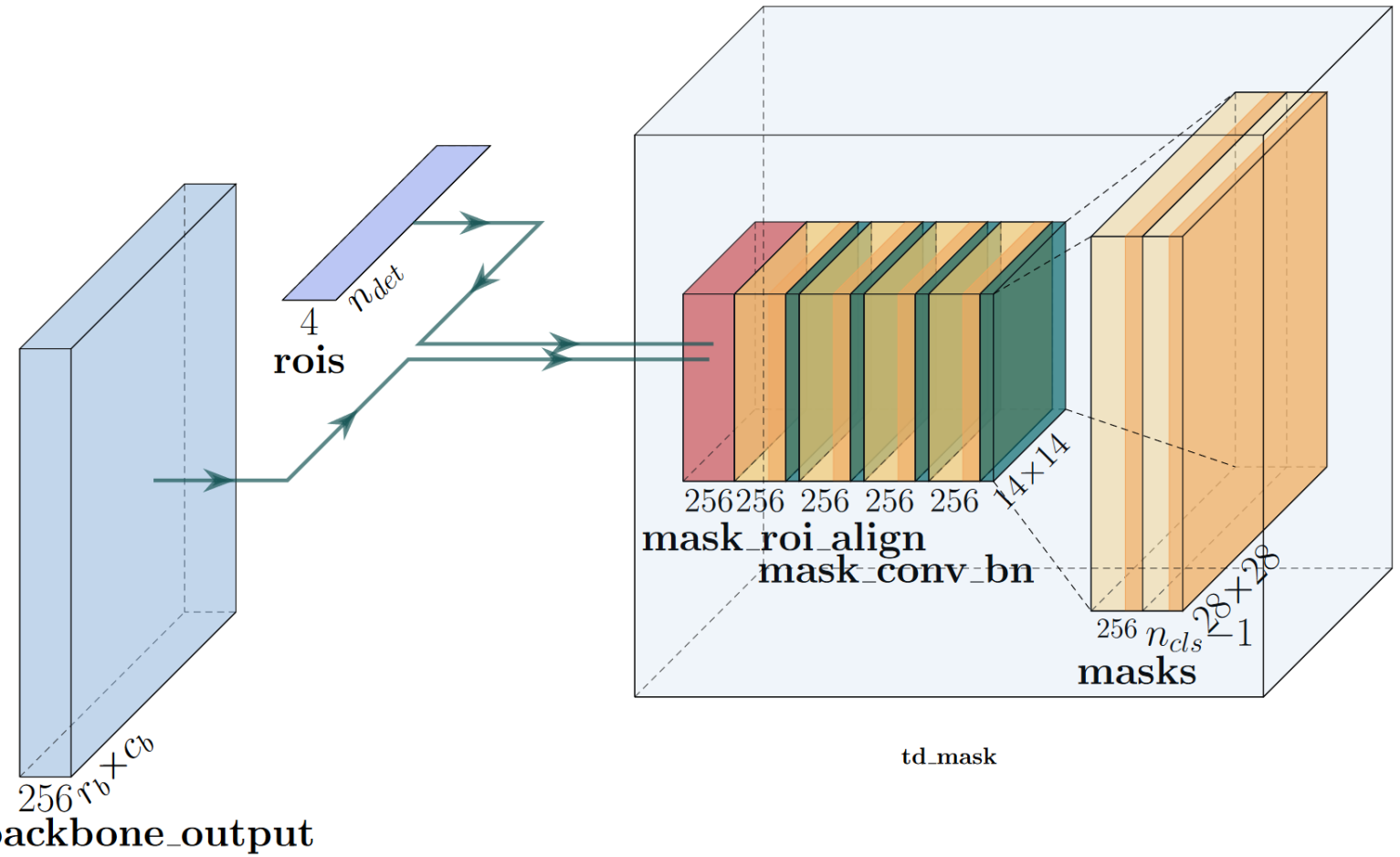


He, K., Gkioxari, G., Dollár, P., & Girshick, R. *"Mask R-CNN"*. ICCV 2017

Mask R-CNN

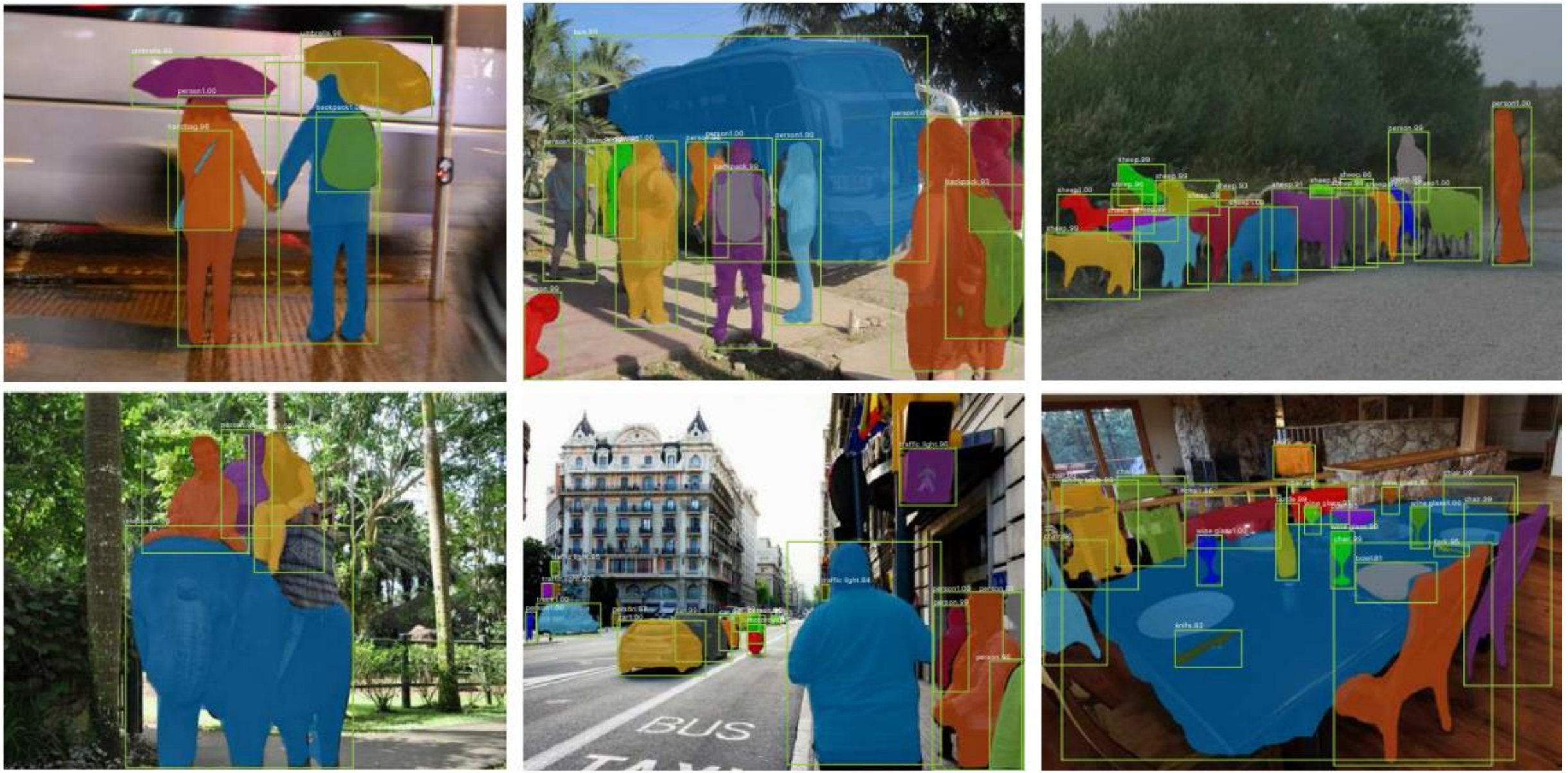
Mask is estimated for each ROI and each class

Extends Faster R-CNN *by adding a branch for predicting an object mask in parallel with the existing branch for bounding box recognition*



He, K., Gkioxari, G., Dollár, P., & Girshick, R. *"Mask R-CNN"*. ICCV 2017

Mask R-CNN (end-to-end training)

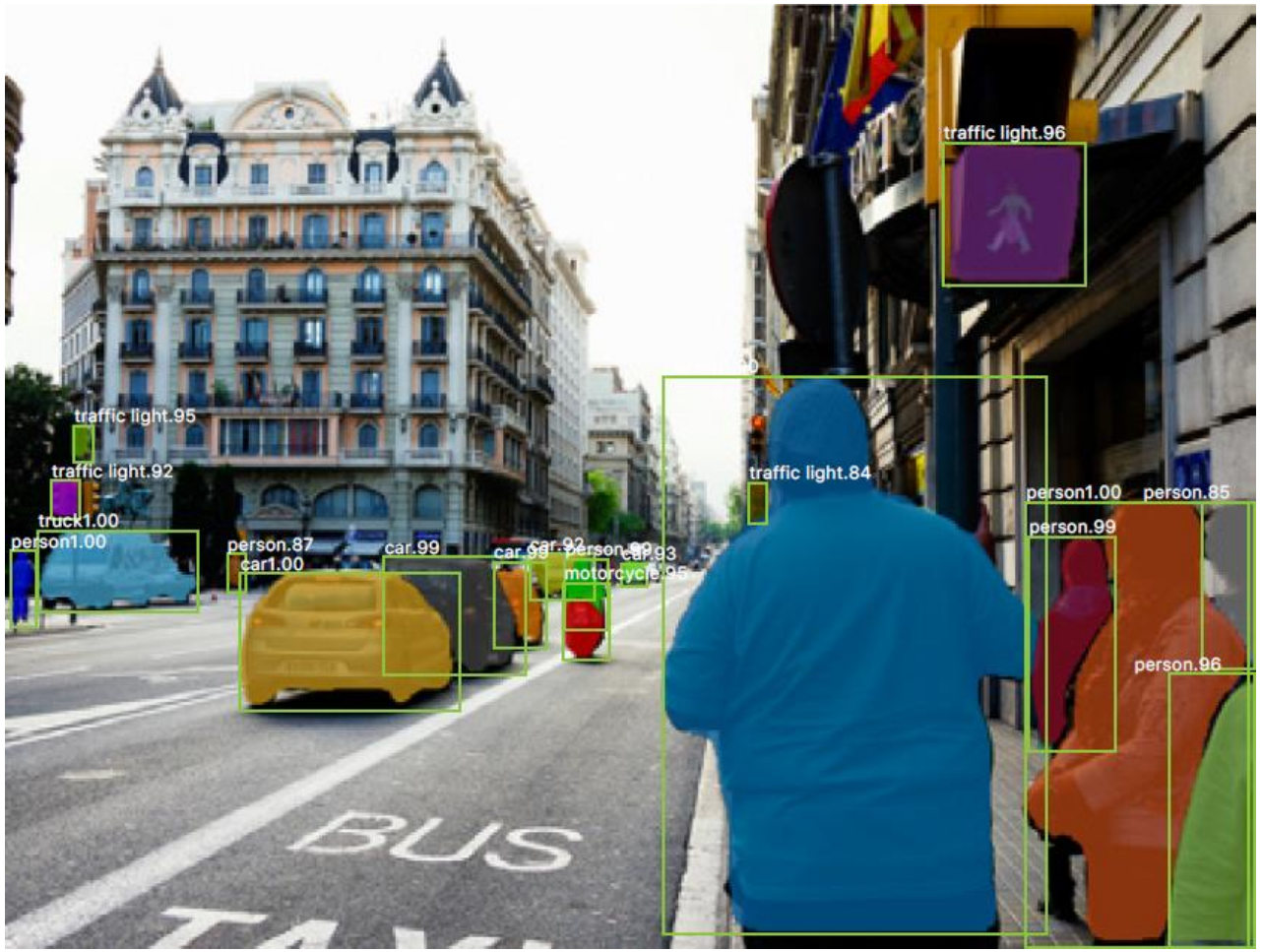


He, K., Gkioxari, G., Dollár, P., & Girshick, R. *"Mask R-CNN"*. ICCV 2017

Mask R-CNN (including Pose estimation)

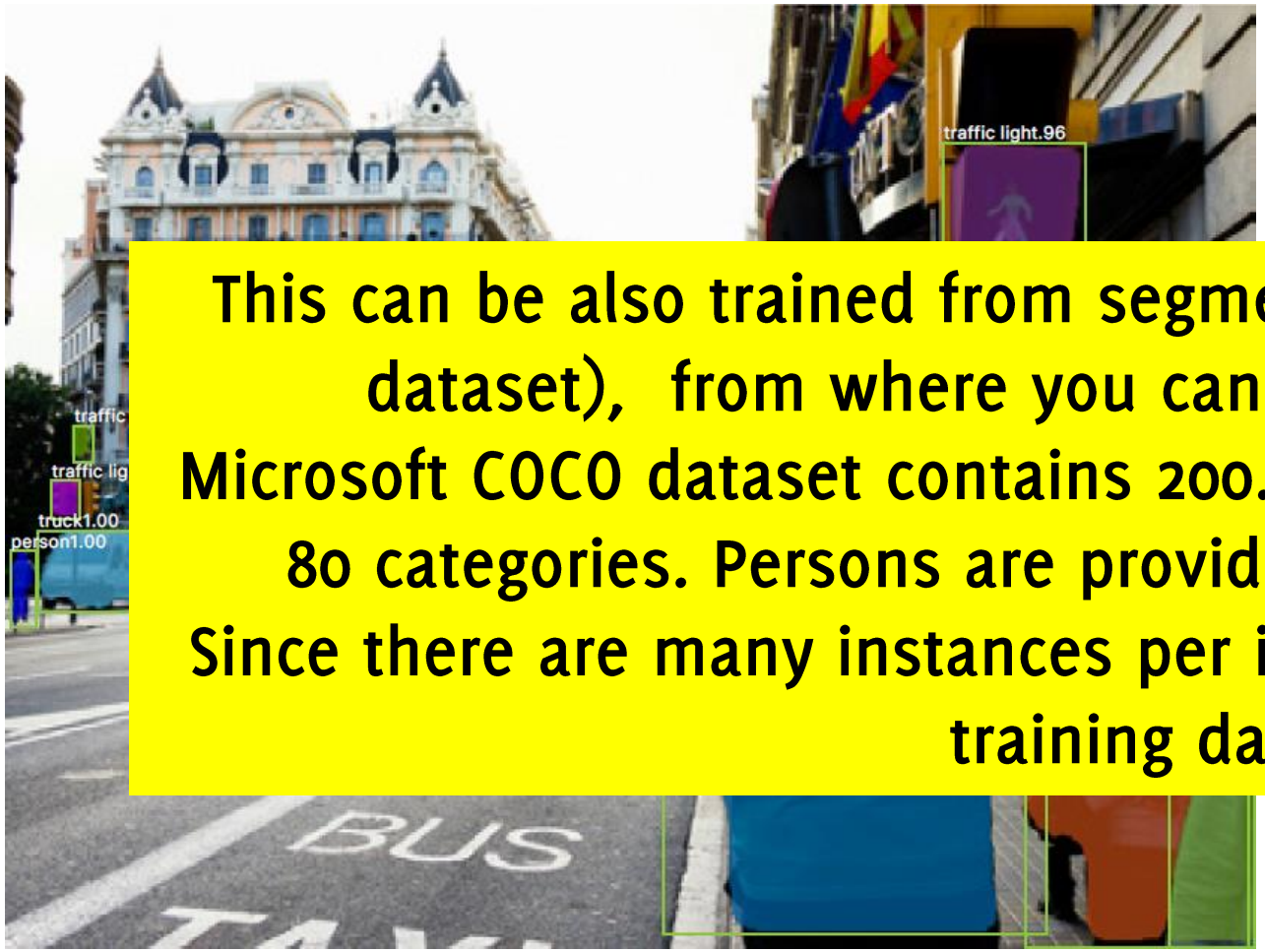
He, K., Gkioxari, G., Dollár, P., & Girshick, R. *"Mask R-CNN"*. ICCV 2017

Instance Segmentation / Human Pose Estimation



He, K., Gkioxari, G., Dollár, P., & Girshick, R. *"Mask R-CNN"*. ICCV 2017

Instance Segmentation / Human Pose Estimation



This can be also trained from segmentation dataset (like COCO dataset), from where you can infer bounding boxes
Microsoft COCO dataset contains 200.000 images segmented over 80 categories. Persons are provided with joints annotated
Since there are many instances per image, this provides a lot of training data

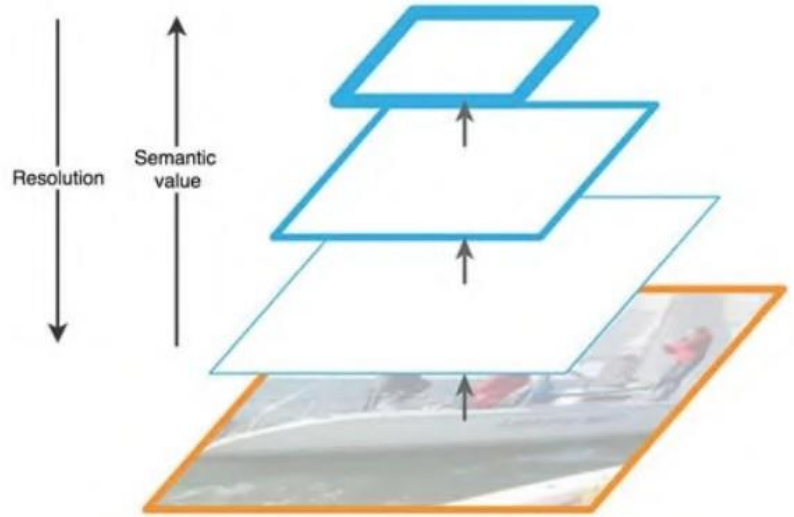
He, K., Gkioxari, G., Dollár, P., & Girshick, R. *"Mask R-CNN"*. ICCV 2017

Feature Pyramid Network

Slide credits: Daniele Casciani, Luca Alessandrini

Goal

The goal is to naturally **leverage the pyramidal shape** of a ConvNet's feature hierarchy while creating a feature pyramid that has **strong semantics at all scales**.



Combines

- semantically strong features (i.e low resolution)
- semantically weak features (i.e high-resolution)

Using

- top-down pathway
- lateral connections

Motivations

Needing of recognizing objects at vastly different scales. Feature pyramid built upon image pyramids (*featurized image pyramids*) were the baseline

Problem

At the time, state-of-the-art methods performed multi-scale testing on featurized image pyramids, but … inference time and memory increased a lot! Moreover, training in an end-to-end manner with image pyramids is infeasible. Many used them only at inference time: this leads to inconsistency w.r.t. training.

For this reason, Fast and Fast R-CNN opt to not use them:

"*Recent and more accurate detection methods like Fast R-CNN [11] and Faster R-CNN [29] advocate using features computed from a single scale, because it offers a good trade-off between accuracy and speed. Multi-scale detection, however, still performs better, especially for small objects*."

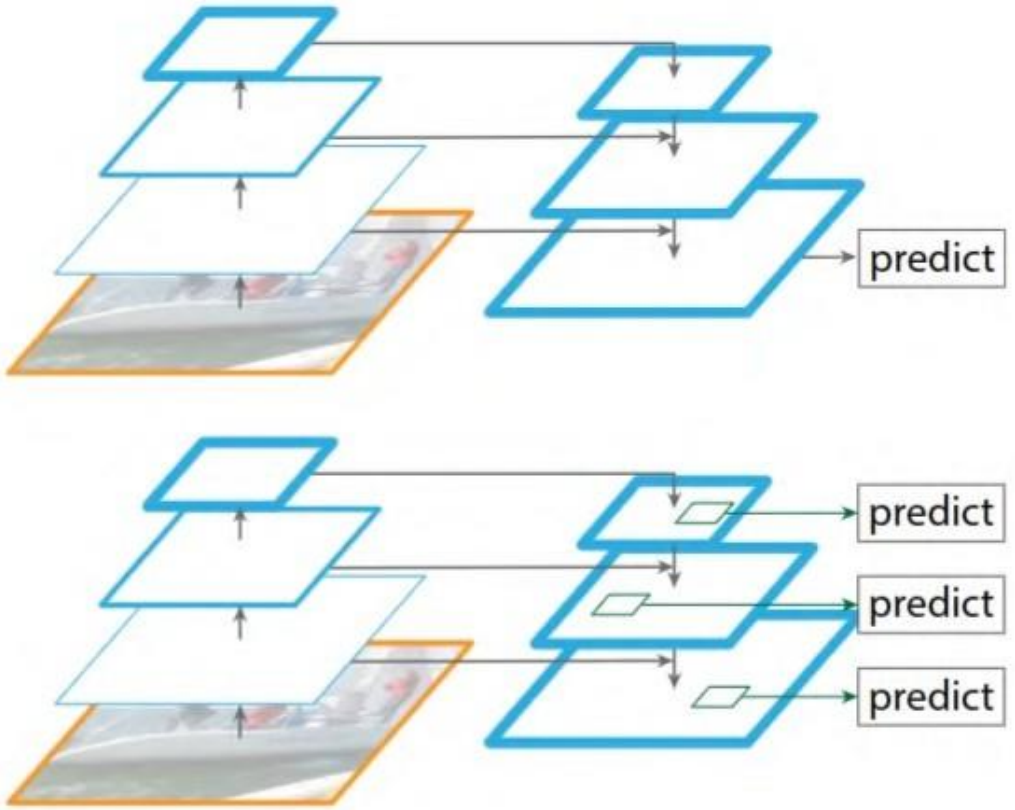
FPN vs Others

Others (e.g U-Net)

- Architectures also adopting top-down and skip connections
- goal is to produce a **single high-level feature map** on which make predictions
- What is missing: needed to recognize objects across multiple scales

Feature Pyramid Network

- predictions (e.g., object detections) are independently made on each level



Method

Input

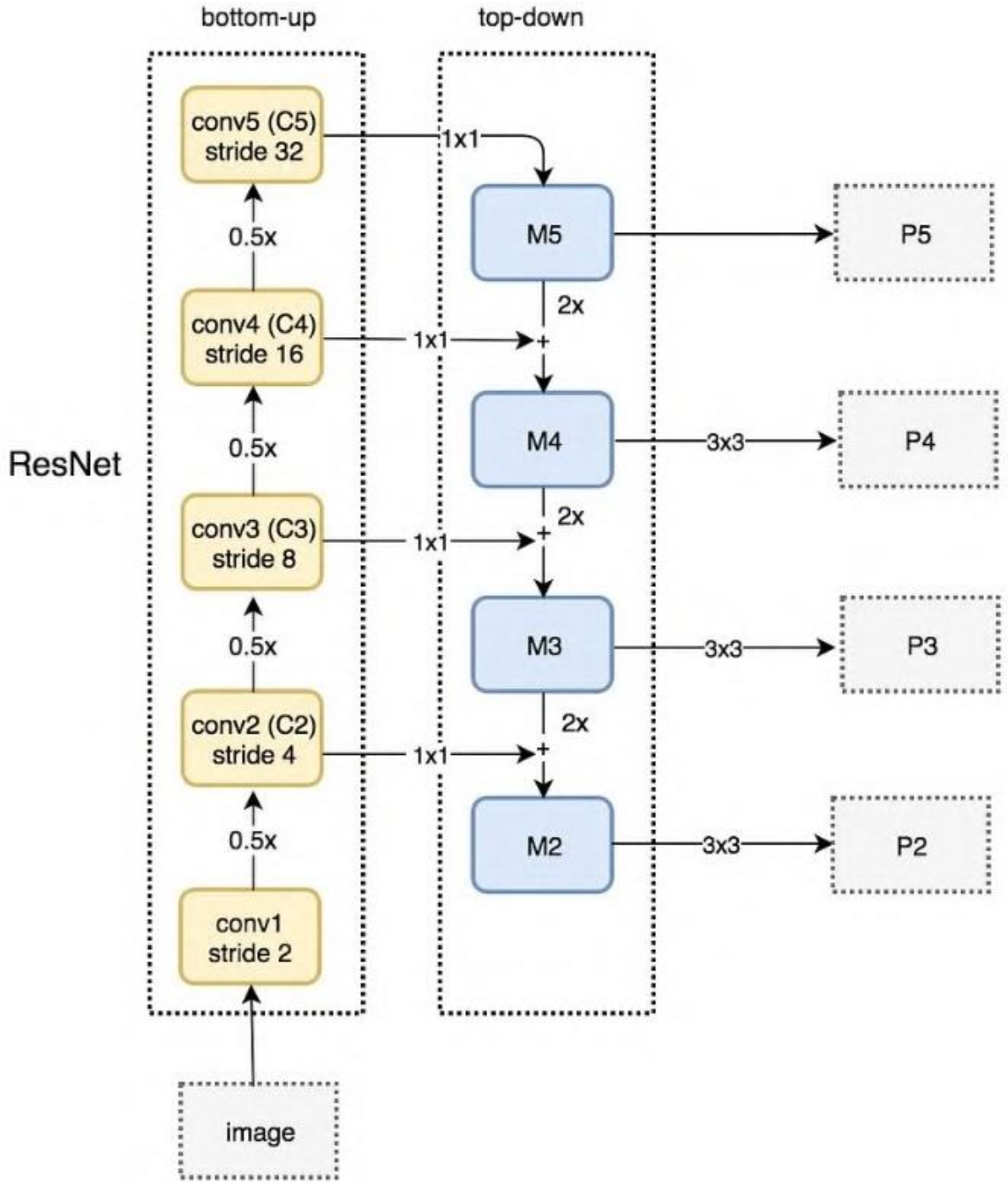
- A single-scale image of an arbitrary size

Output

- proportionally sized feature maps

Components

- Bottom-up pathway
- Top-down pathway
- Lateral-connections



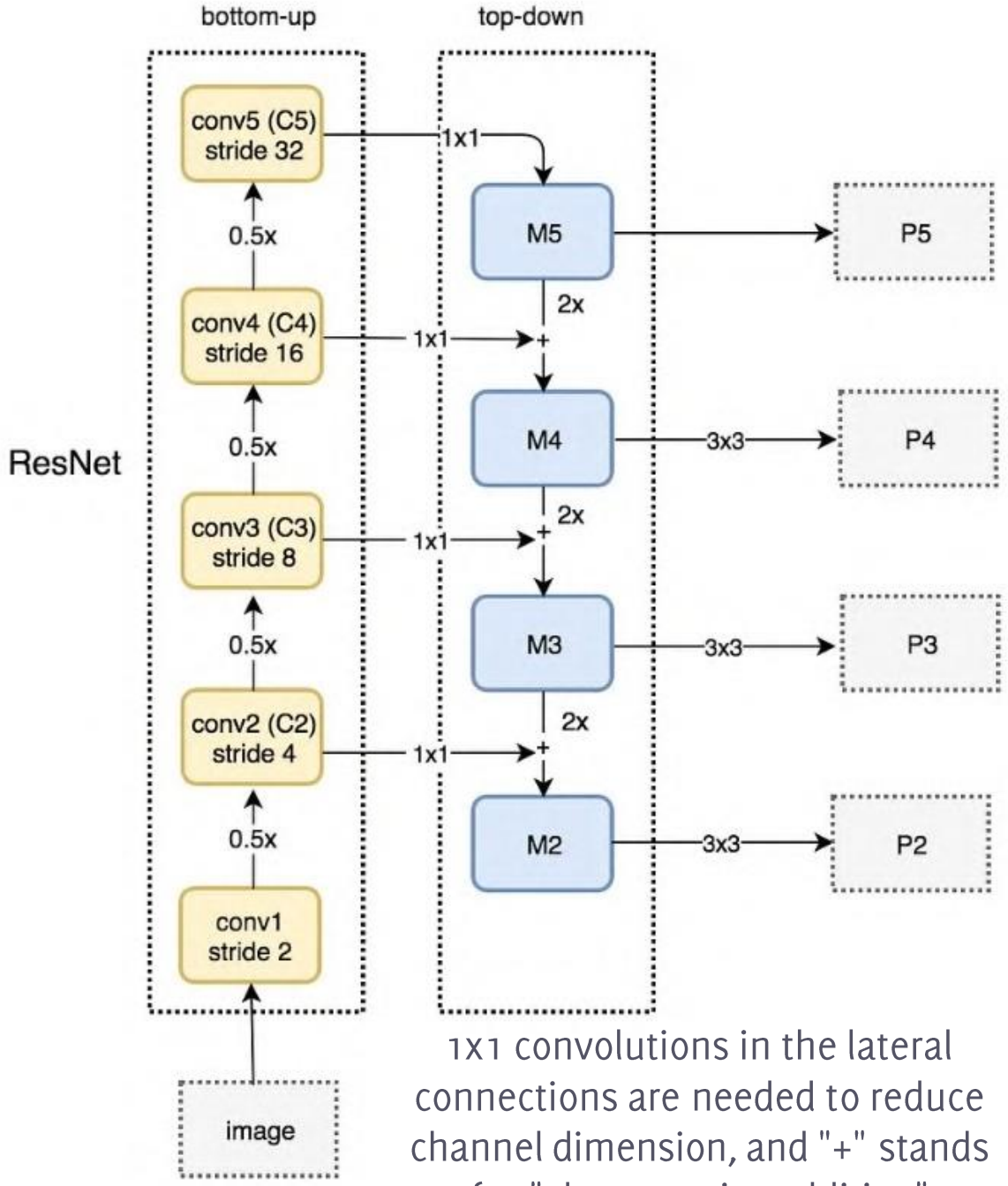
Method

Bottom-up

- Proportionally sized feature maps
- Moving up, the spatial dimension is reduced by 1/2
- Output of each convolution is used in the top-down pathway

Top-down with connections

- Derives higher resolution features by upsampling .
- Enhanced with features from the bottom up part through lateral connections (reduce channel dimensions).
- 3 ×3 convolution to merged layers, that reduces the aliasing effect.

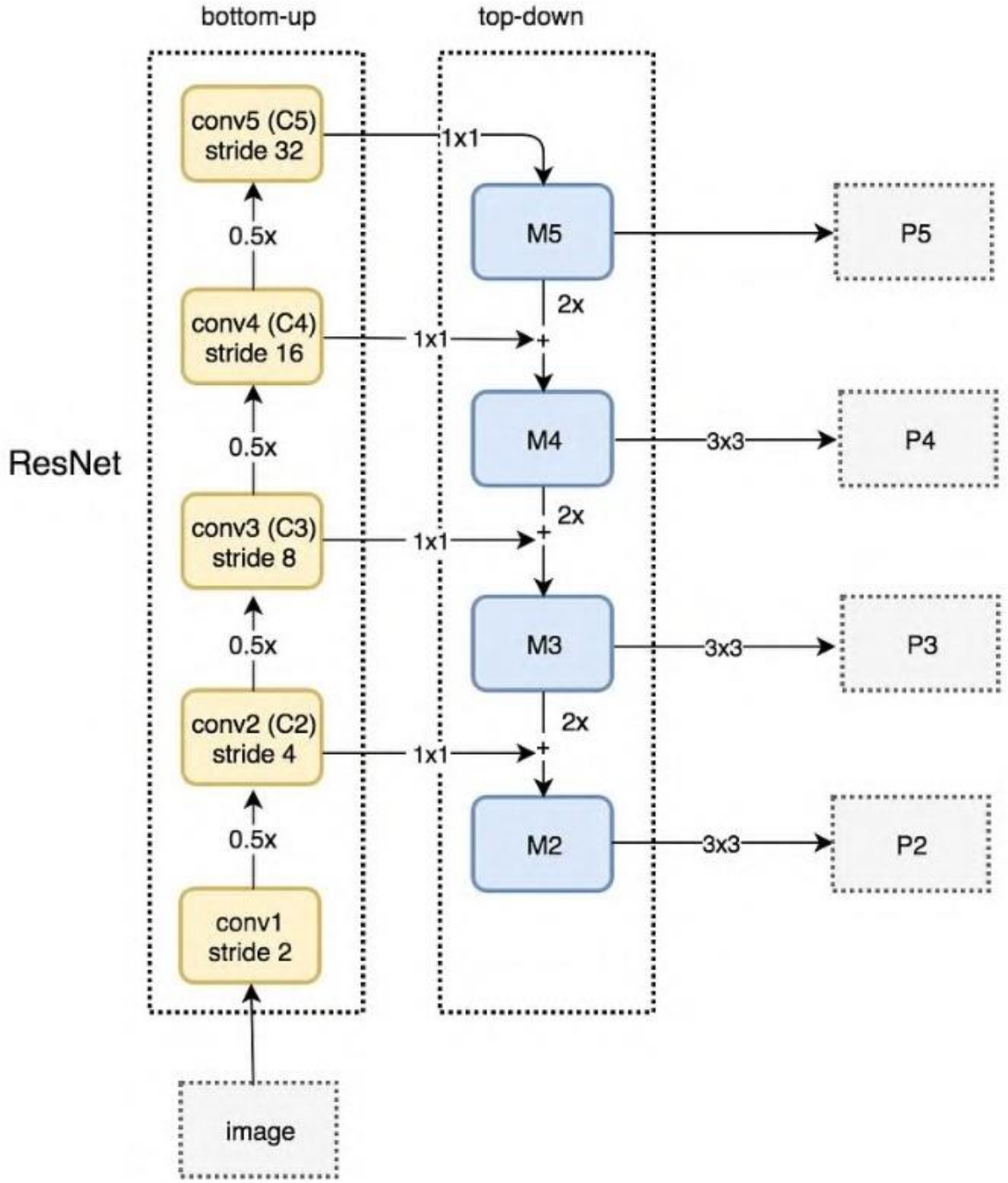


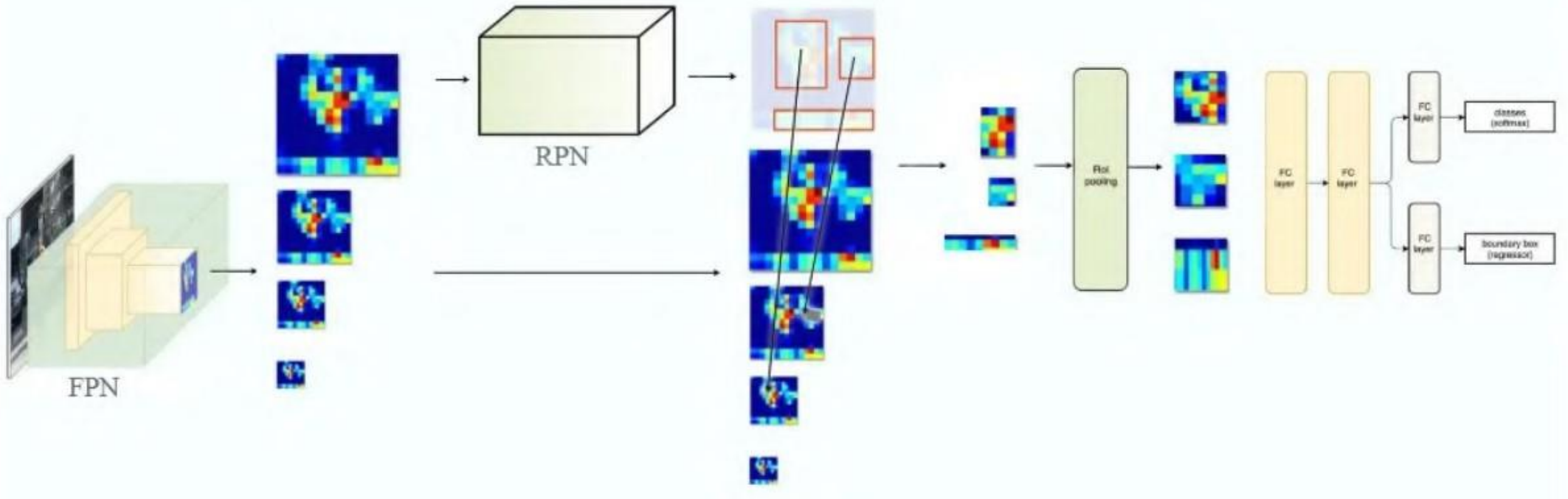
1x1 convolutions in the lateral connections are needed to reduce channel dimension, and "+" stands for "element-wise addition"

FPN with RPN

FPN is not an object detector by itself. It is a feature extractor that works with object detectors

RPN was adapted by replacing the single- scale feature map with FPN by attaching a **predictor head** (3×3 conv and two sibling 1×1 convs) **to each level** on the feature pyramid. Since the head slides over all locations in all pyramid levels, it is not necessary to have multi-scale anchors on a specific level.





FPN with Fast R-CNN or Faster R-CNN

We apply the RPN to generate RoIs. Based on the size of the RoI, we select the feature map layer in the proper scale to extract the feature patches. Formally, a RoI of width w and height h is assigned to the level k of the feature pyramid:

$$k = \lfloor k_0 + \log_2(\sqrt{wh}/224) \rfloor.$$

where:

$k_0 = 4$

k is the P_k layer in the FPN used to generate the feature patch.

So if k = 3, we select P3 as our feature maps. We apply the ROI pooling and feed the result to the Fast R-CNN head to finish the prediction.

References

1. https://jonathan-hui.medium.com/understanding-feature-pyramid-networks-for-object-detection-fpn-45b227b9106c
2. https://openaccess.thecvf.com/content_cvpr_2017/html/Lin_Feature_Pyramid_Networks_CVPR_2017_p aper.html

In collaboration with

ikonisys
FINDING THE CELLS THAT MATTER

Instance Segmentation

Research Project with Ikonisys

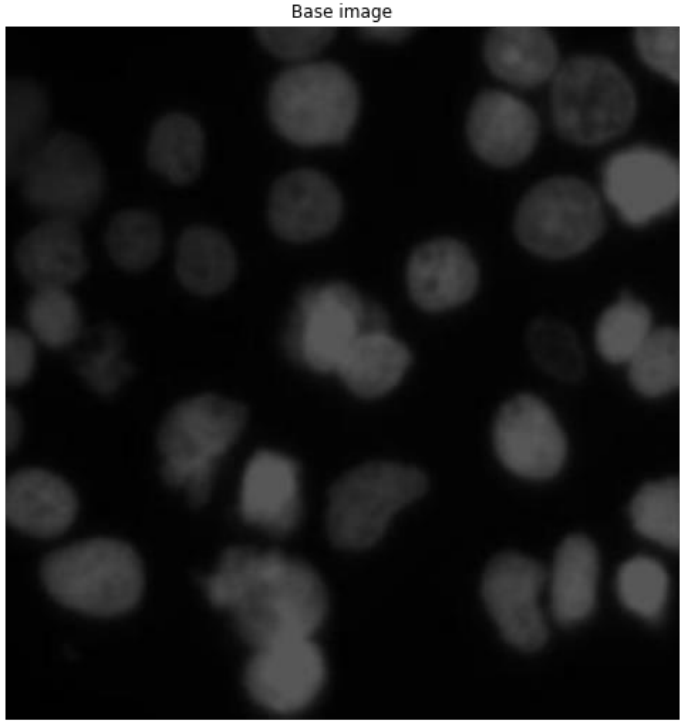
Credits Roberto Basla

In collaboration with

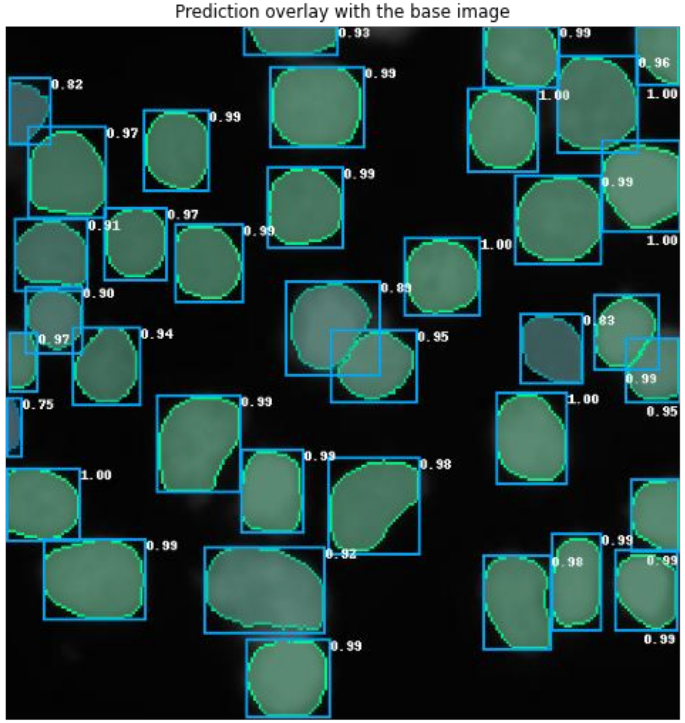
ikonisys
FINDING THE CELLS THAT MATTER

Instance Segmentation Network

The instance segmentation network is able to identify nuclei in fluoroscopy images.



Image



Prediction

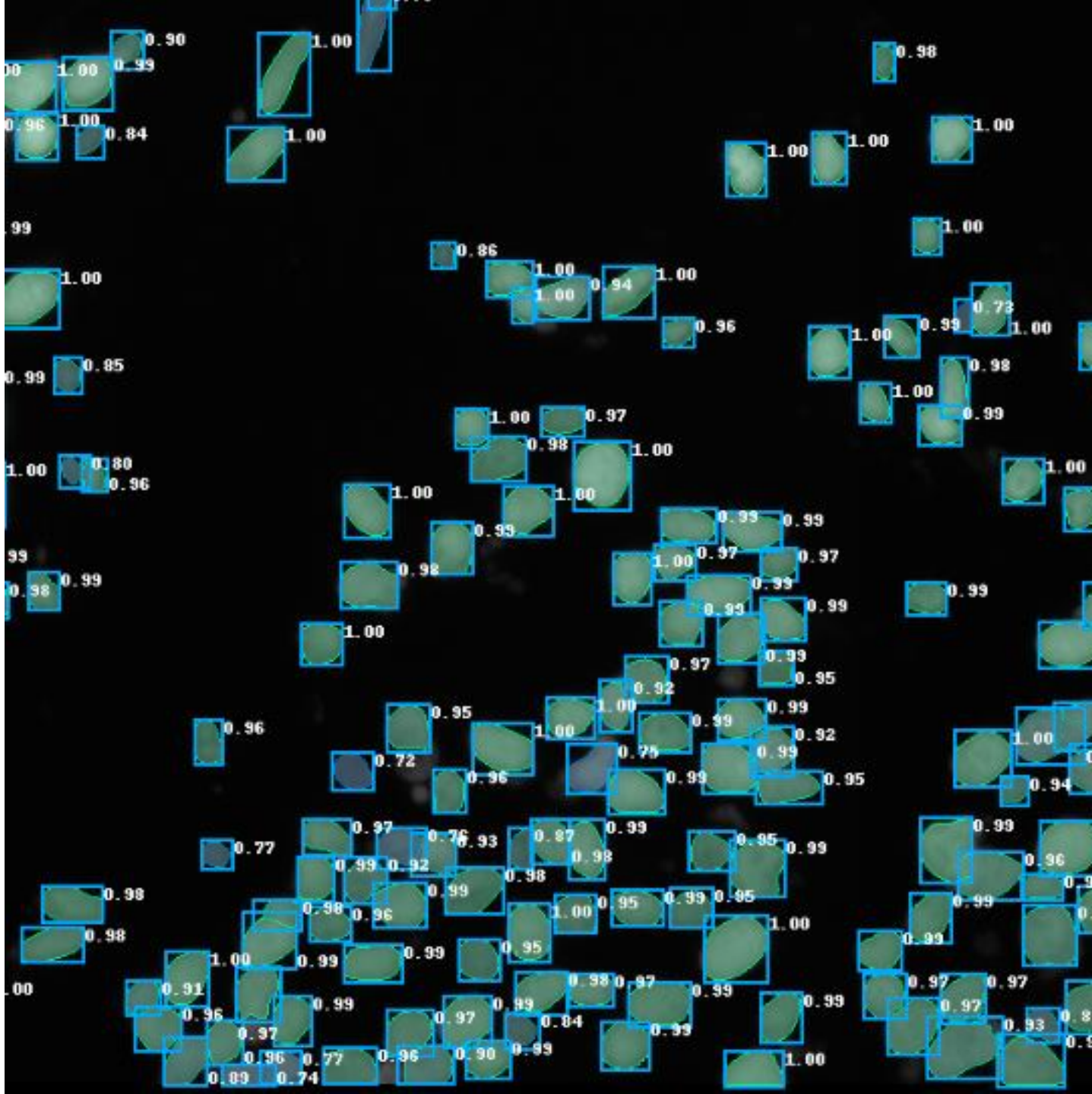
The network provides, for each identified nucleus, its segmentation mask, bounding box, and prediction confidence.

Credits Roberto Basla

Model Performance

The model achieves great results even when trained with **very few images from the target domain**, by making use of transfer learning from these datasets:

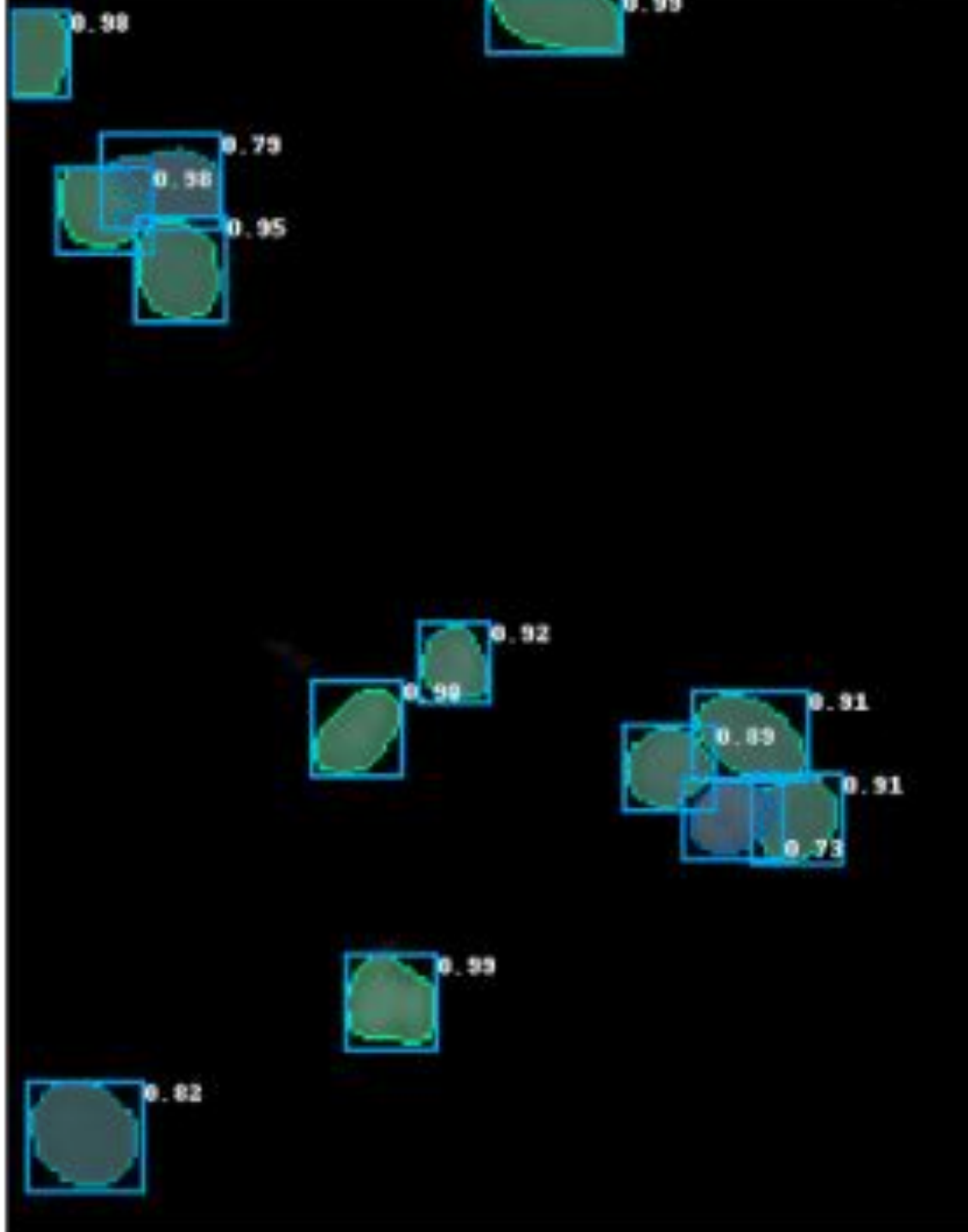
- DSB2018
- DSB2019
- BBBC039
- NSDE



Credits Roberto Basla

Overlapping segmentation

Overlapping cells can be identified and masked independently.



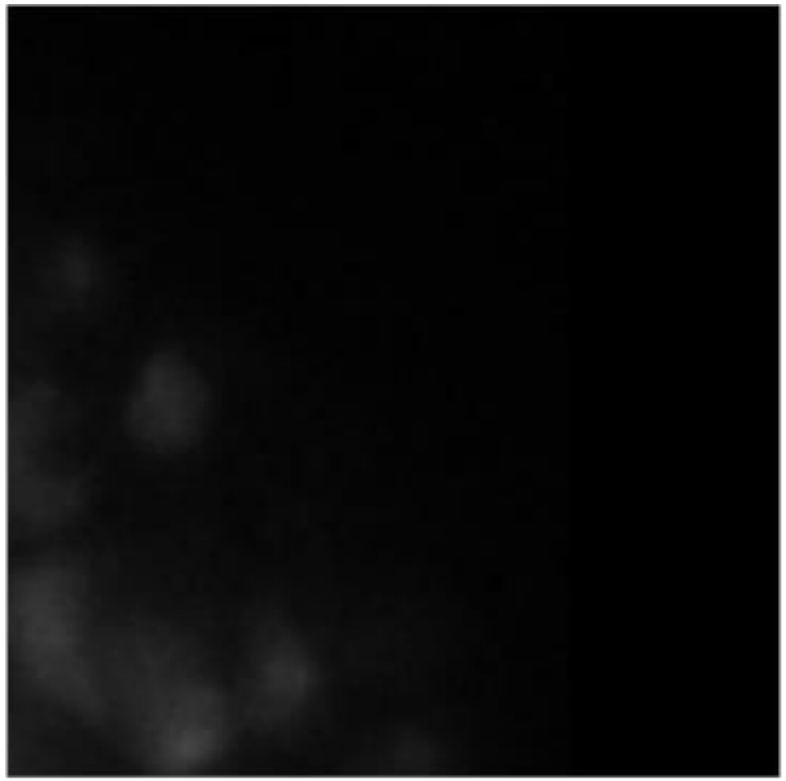
Credits Roberto Basla

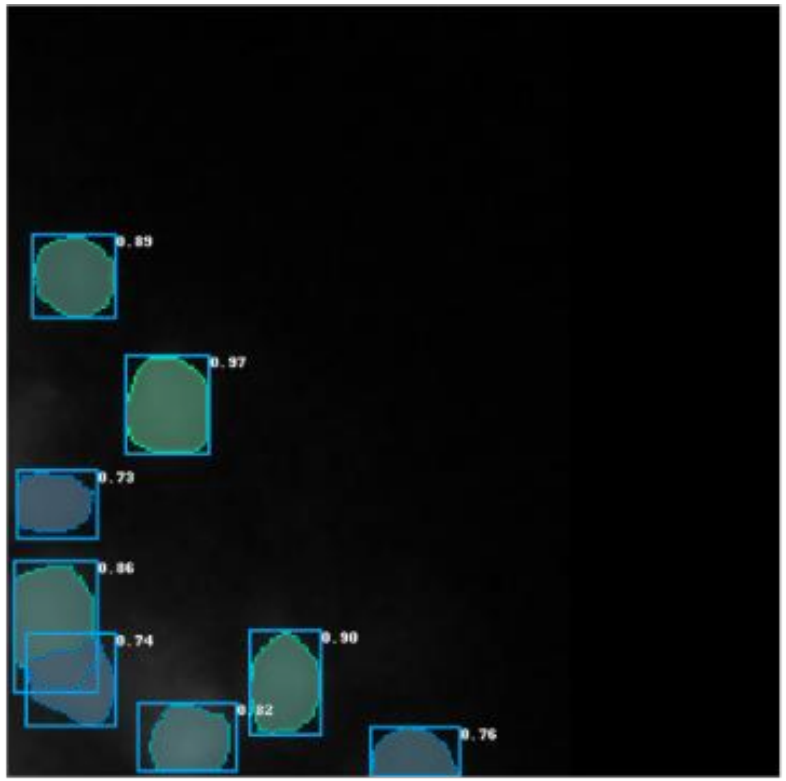
In collaboration with

ikonisys
FINDING THE CELLS THAT MATTER

Robustness

The model **can identify even the faintest of nuclei** in the image, further proving the performance of the system.





Credits Roberto Basla

Ongoing Research Activities

… good for thesis projects

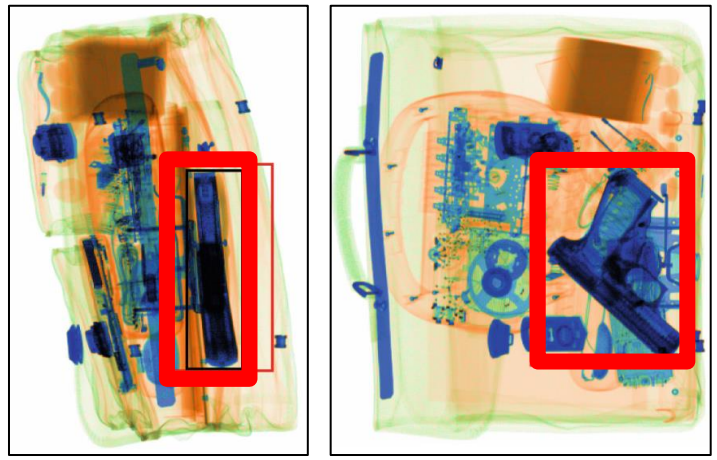
Multi-view object detection CNN

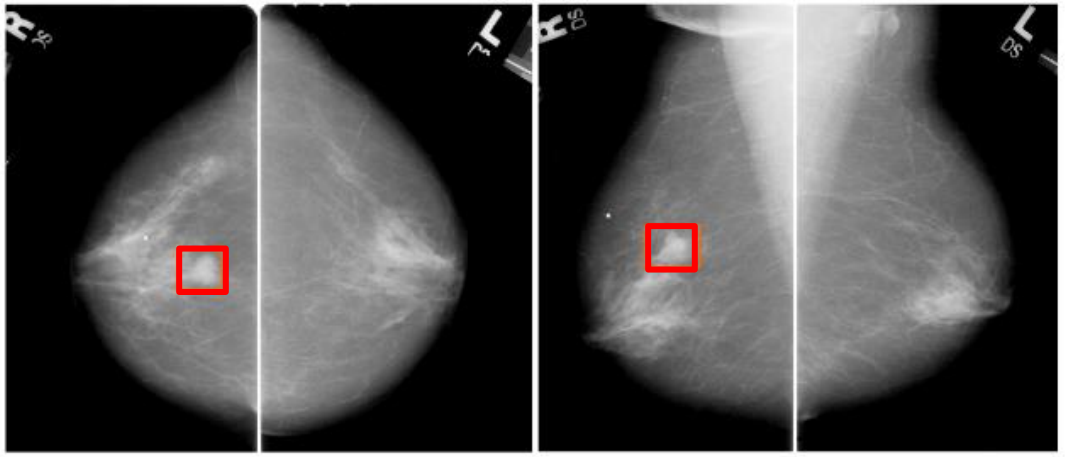
Detection systems usually take only a single input image. In some applications, such as

- Baggage inspection in airports
- Tumor detection in X-ray images

you have multiple views of the same objects.

Exploiting multiple-views can increase detection power.





[1] Steitz et al., "Multi-view x-ray r-cnn." (2018)
[2] Wimmer et al. "Multi-task fusion for improving mammography screening data classification" (2021)

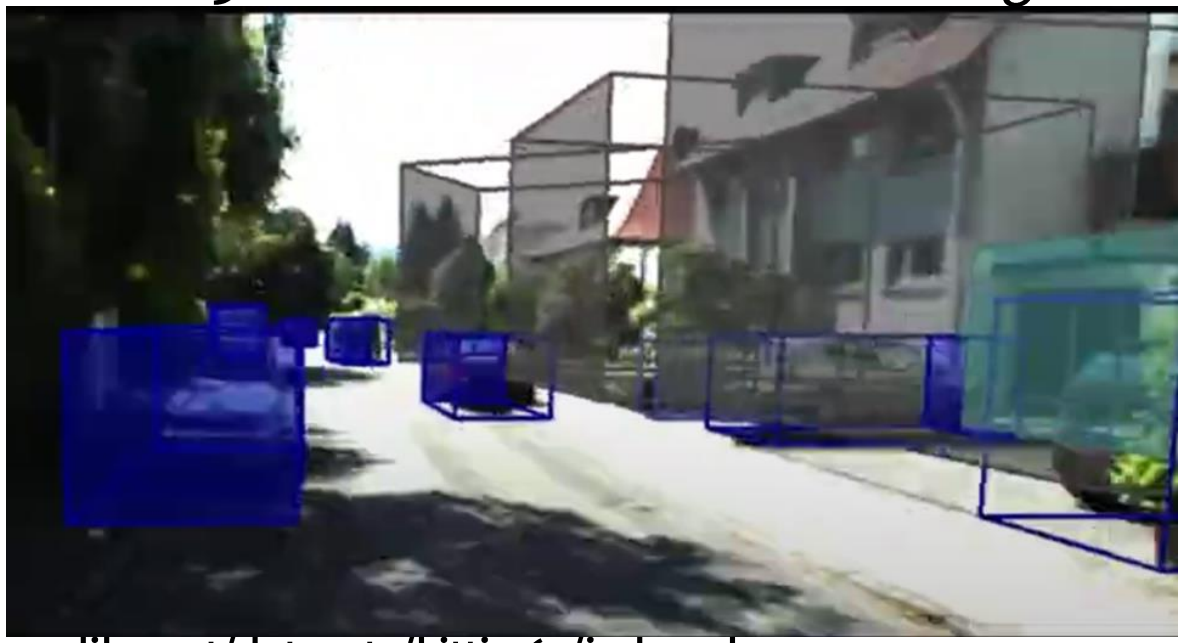
Multimodal object detection CNN

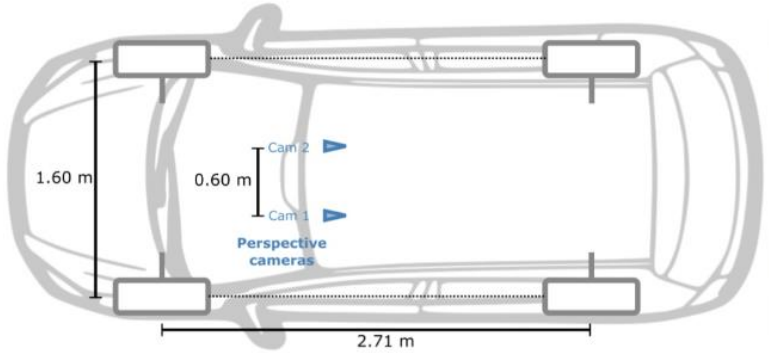
In collaboration with Prof. Savaresi, MOVE Research Group

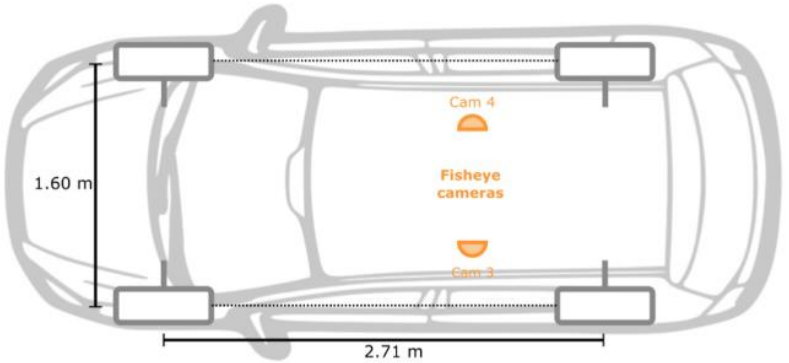
Object detection can leverage multiple streams:

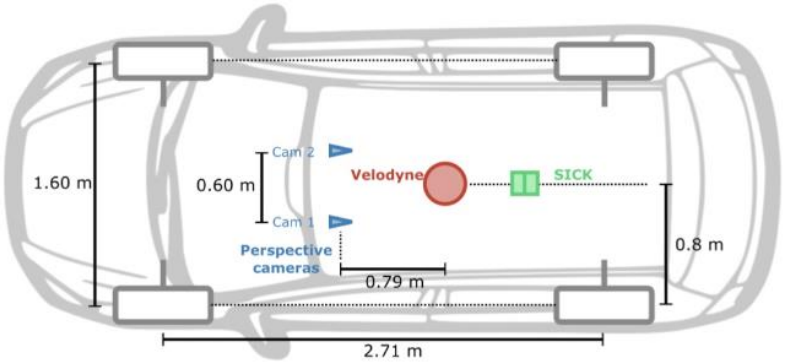
- RGB images (possibly stereo pairs)
- Point Clouds from lasers scanners
- event cameras..

To infer 3D coordinates of bounding boxes

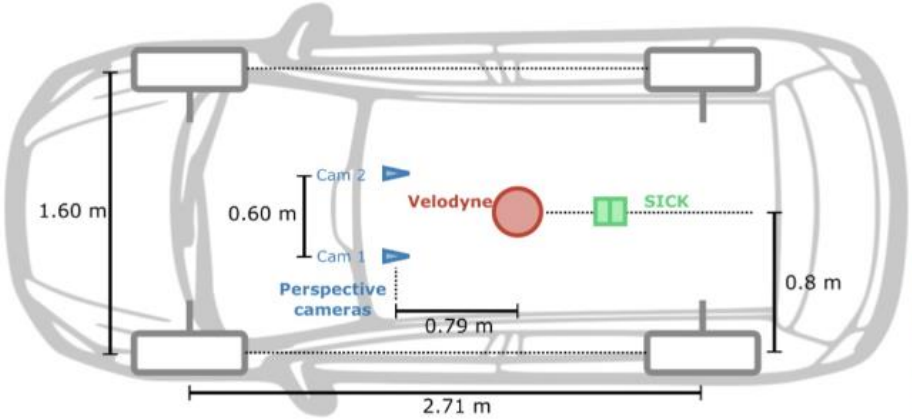


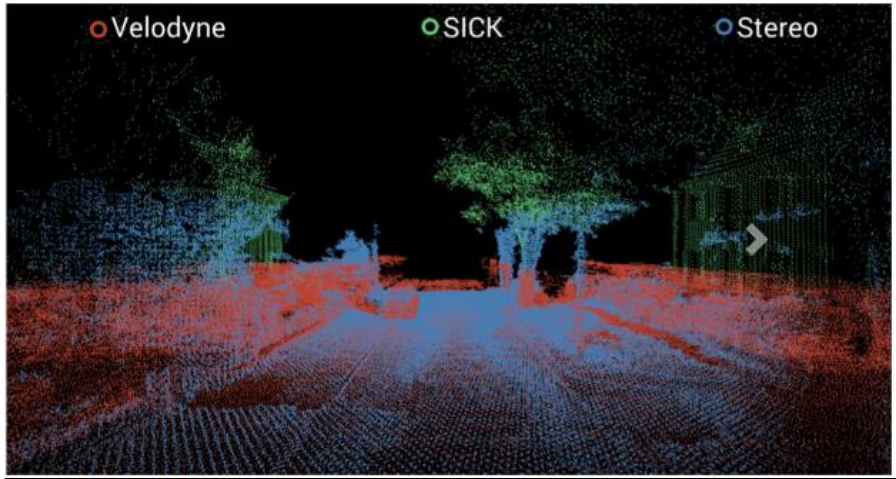


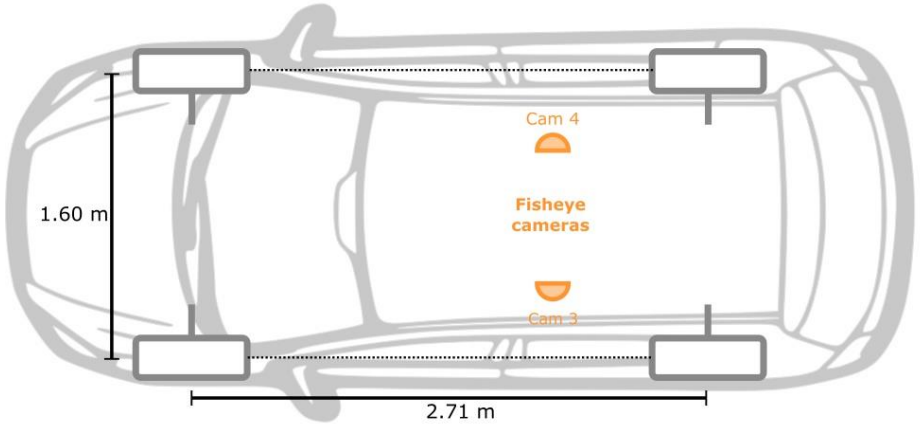


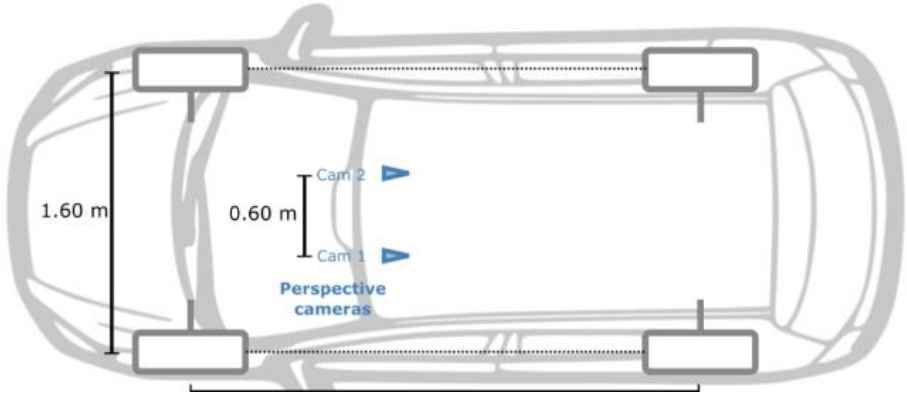


https://www.cvlibs.net/datasets/kitti-360/index.php

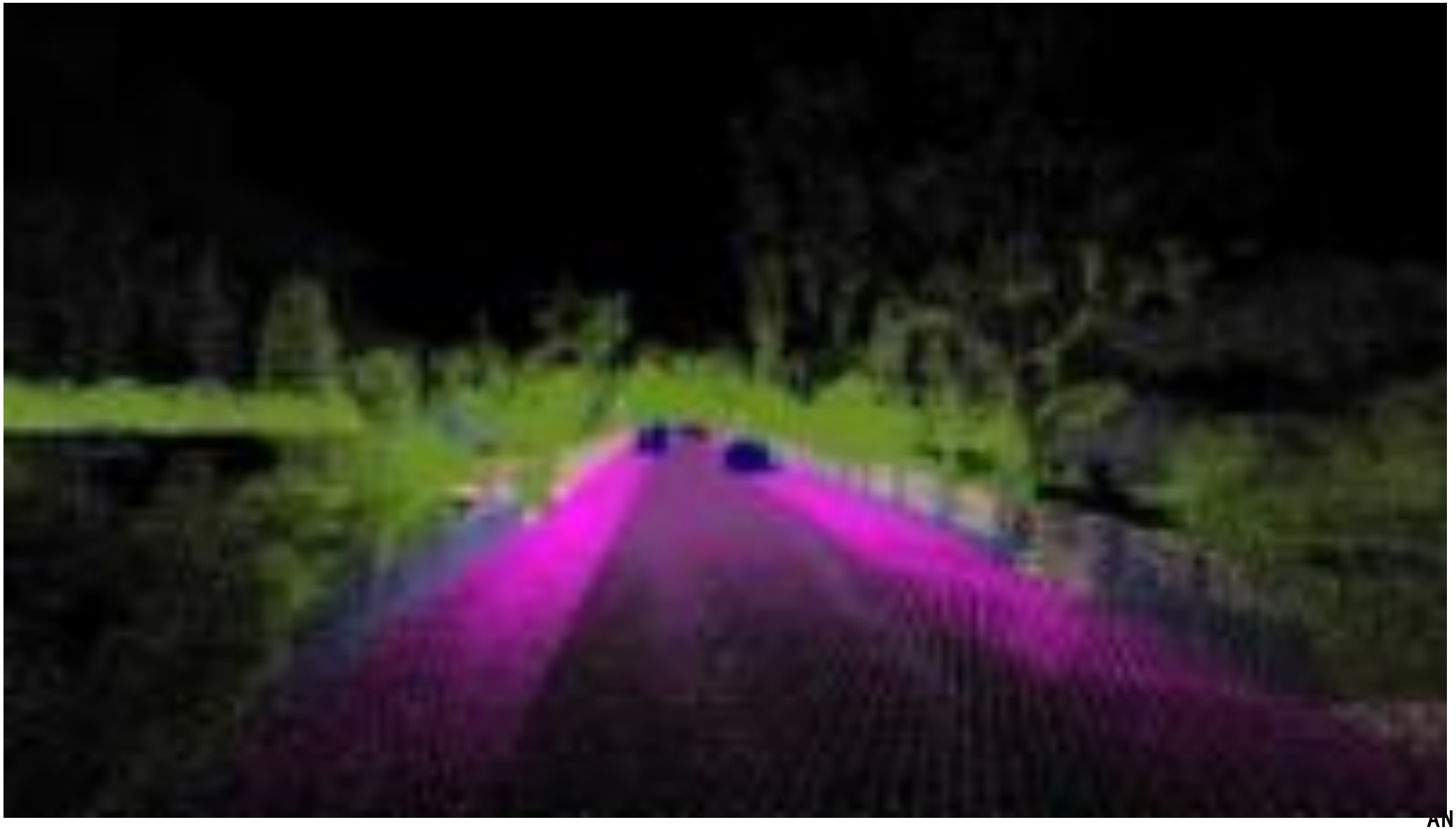








https://www.cvlibs.net/datasets/kitti-360/index.php

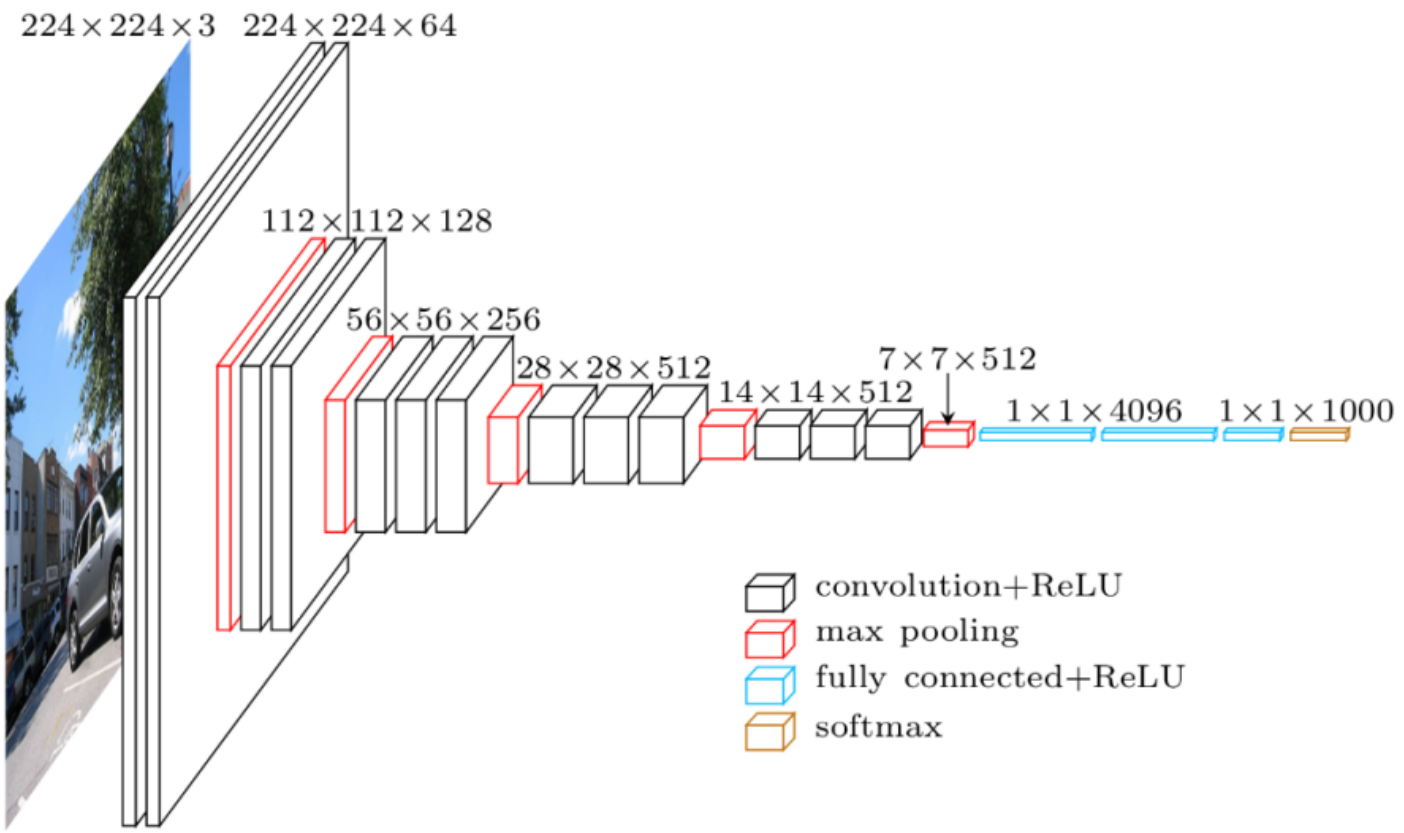


Metric Learning

Say you are asked to implement a face identification system
to open Bocconi door!

Classification? Detection? Segmentation?

Let's start fresh from a Convolutional Neural Network for classification



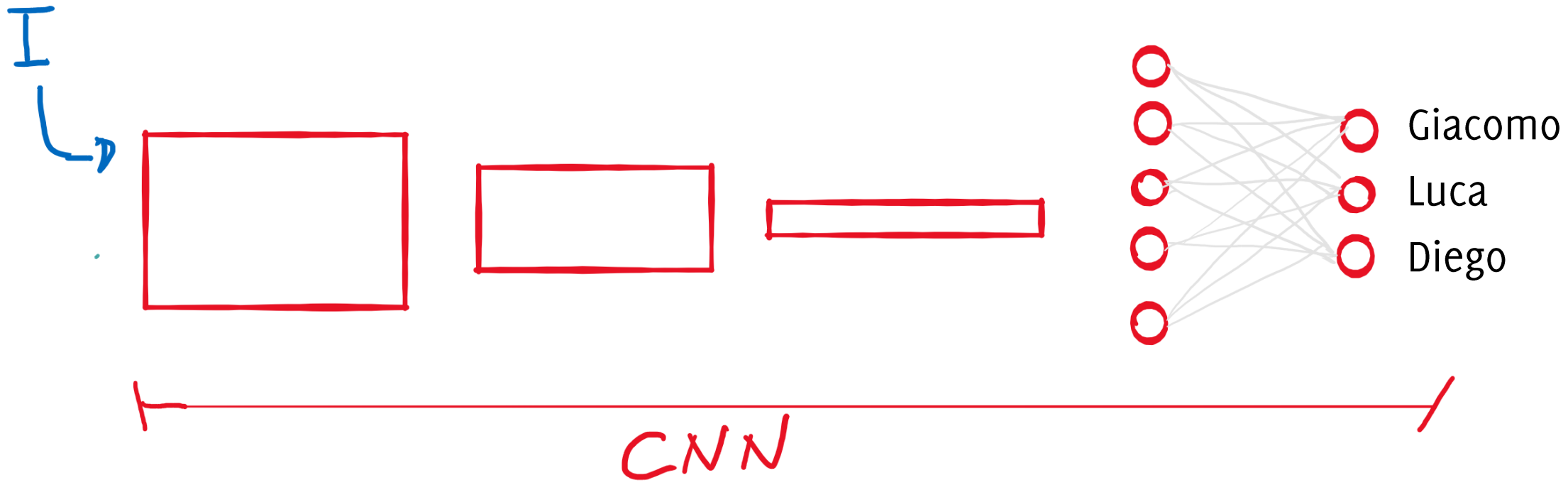
1 person

=

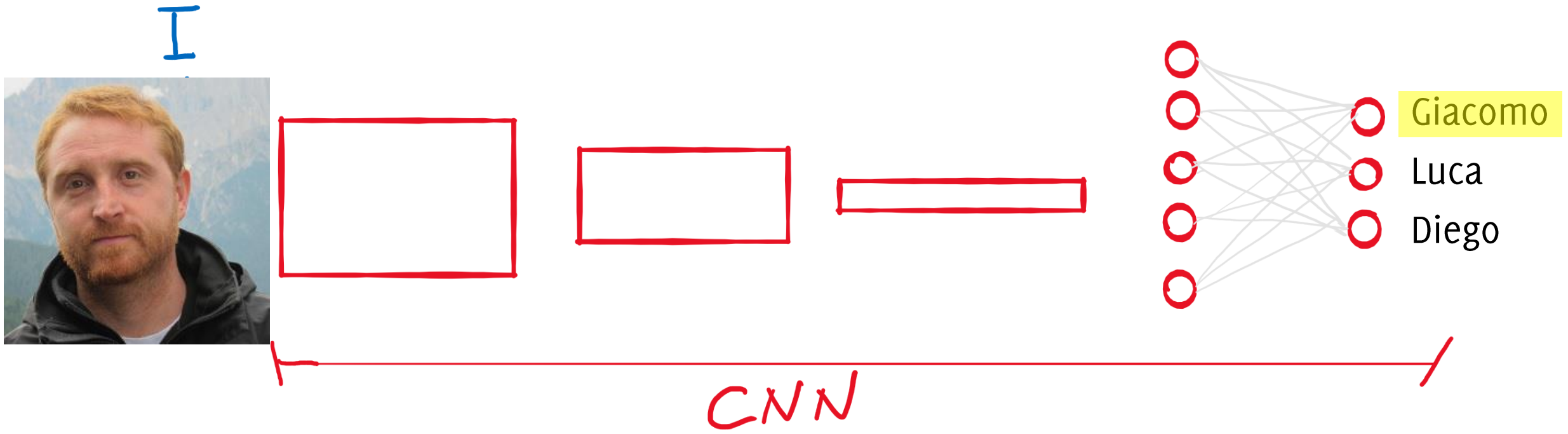
1 class

Simonyan, Karen, and Andrew Zisserman. "Very deep convolutional networks for large-scale image recognition." ICLR 2015

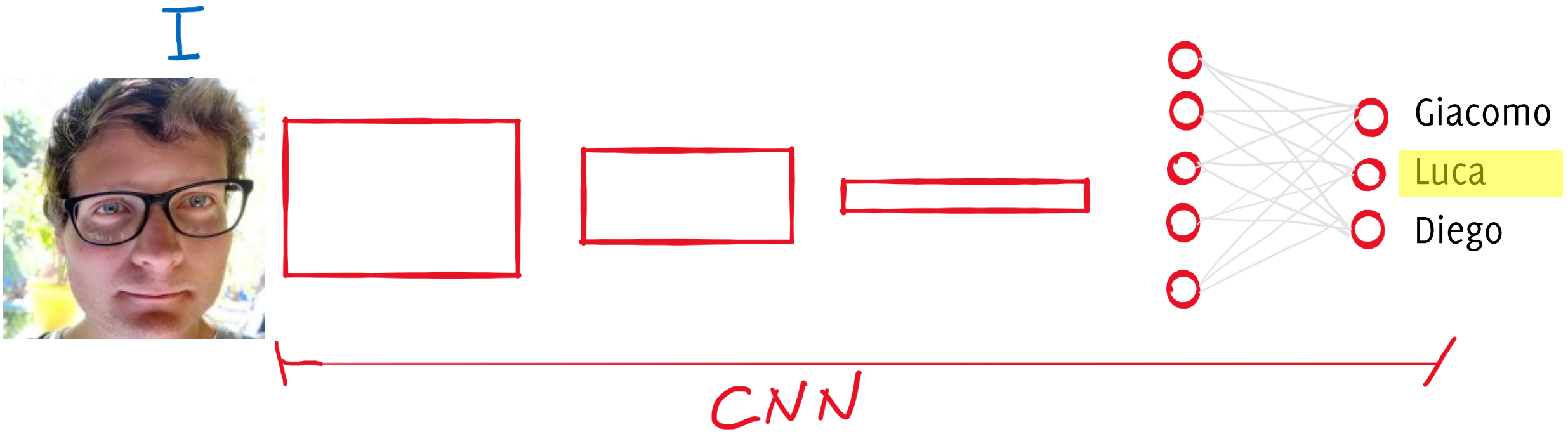
Face identification by Classification



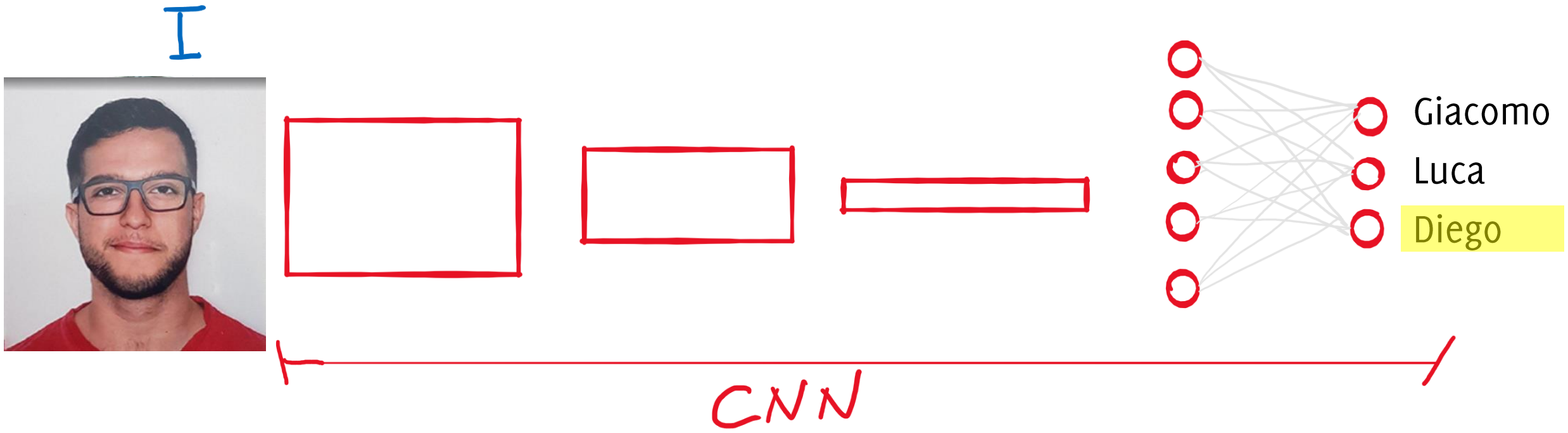
Face identification by Classification



Face identification by Classification



Face identification by Classification



What do we need?

A training set

- A few images per class / person
- Possibly Images in different conditions (position, light, facial expression, clothes …)

A few Py snippets and a GPU…

That’s easy…

What do we need?

A training set

- A few images per class / person
- Possibly Images in different conditions (position, light, facial expression, clothes …)

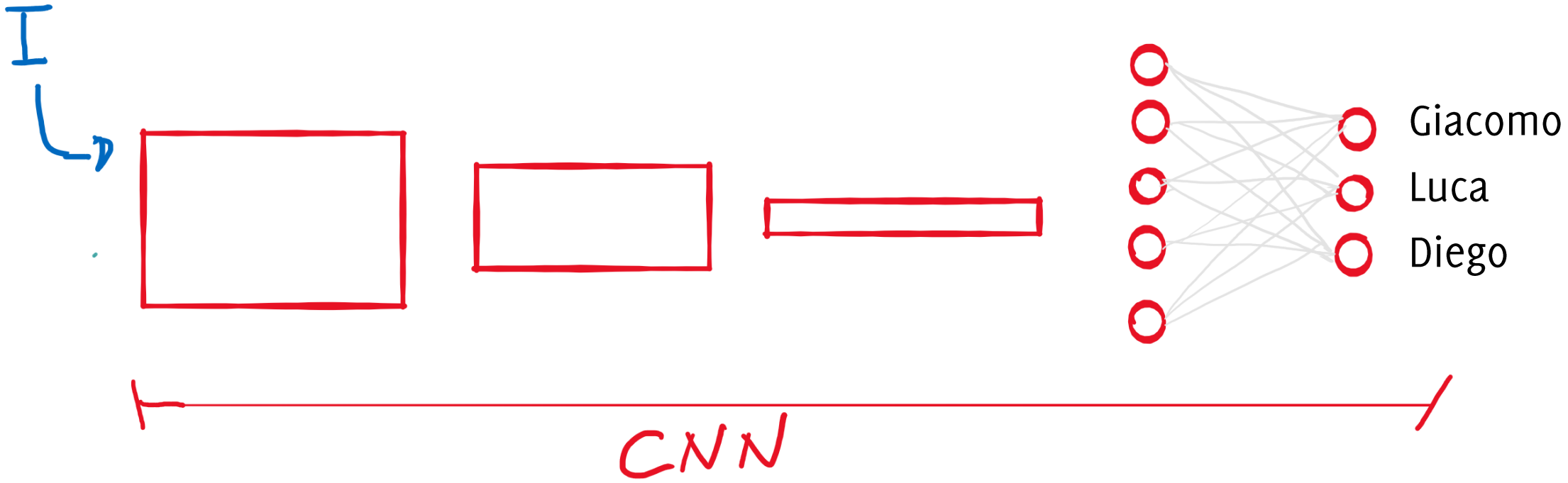
A few Py snippets and a GPU…

That’s easy…

Are we happy with this solution?

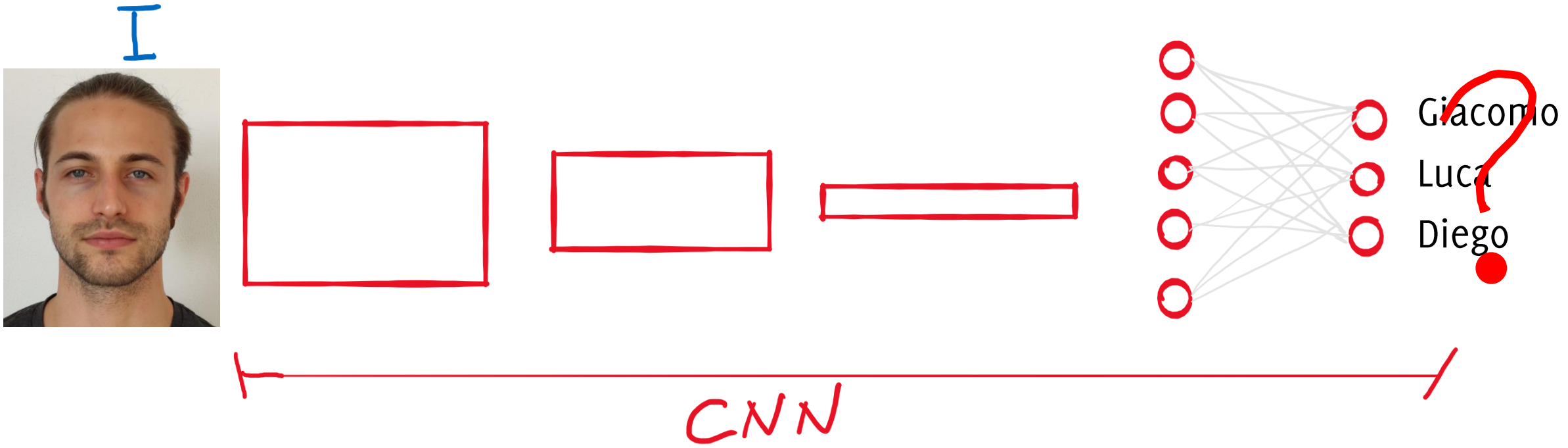
… Not quite

What happens when you need to enroll a new employee?



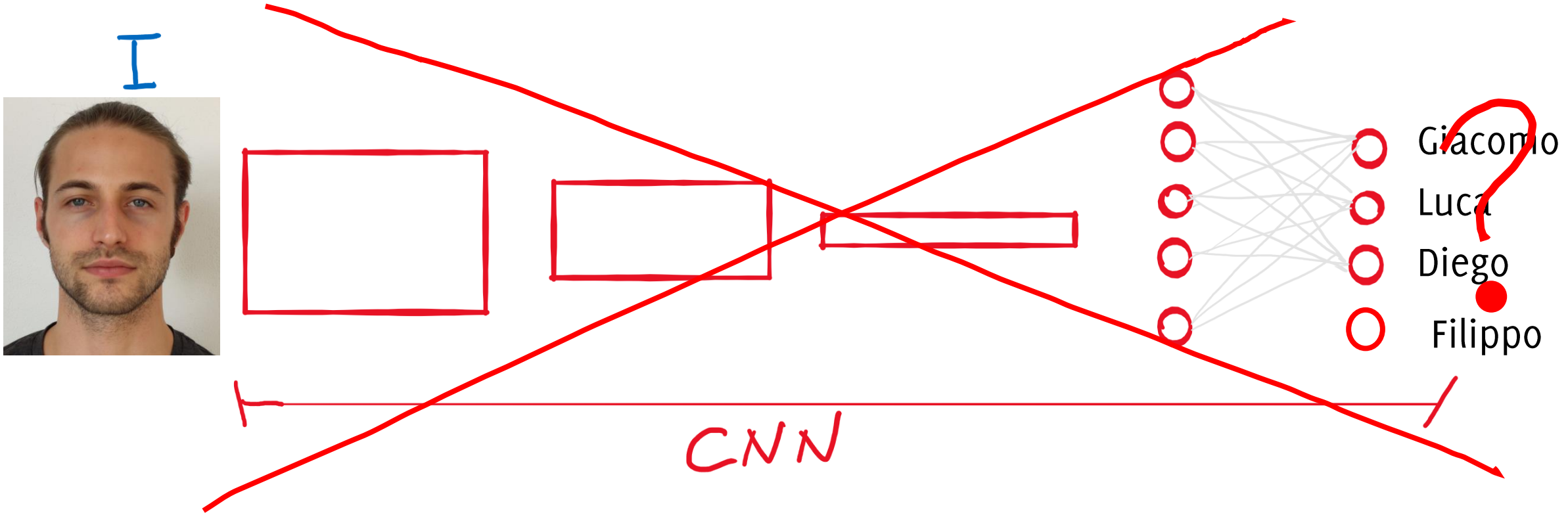
… Not quite

What happens when you need to enroll a new employee?



… Not quite

What happens when you need to enroll a new employee?



The whole network has to be retrained for each new person to be identified

A Different Approach is Needed!

What about image comparison?

Why don’t we store a picture for each employee (**template**), and then perform identification by **pairing the input to its closest template**?

T_1 T_2 T_3

What about image comparison?

So, identification becomes:

$$\hat{\imath} = \underset{j=1,..,3}{\operatorname{argmin}} \left\| I - T_j \right\|_2$$

I

T_1

T_2

T_3

What about image comparison?

Enrolling a new individual would be straightforward

T_1

T_2

T_3

What about image comparison?

Enrolling a new individual would be straightforward… it is enough to add a template to the database!

T_1 T_2 T_3 T_4

What about image comparison?

… but how to perform identification?

$$\hat{\imath} = \underset{j=1,..,4}{\operatorname{argmin}} \left\| I - T_j \right\|_2$$

I

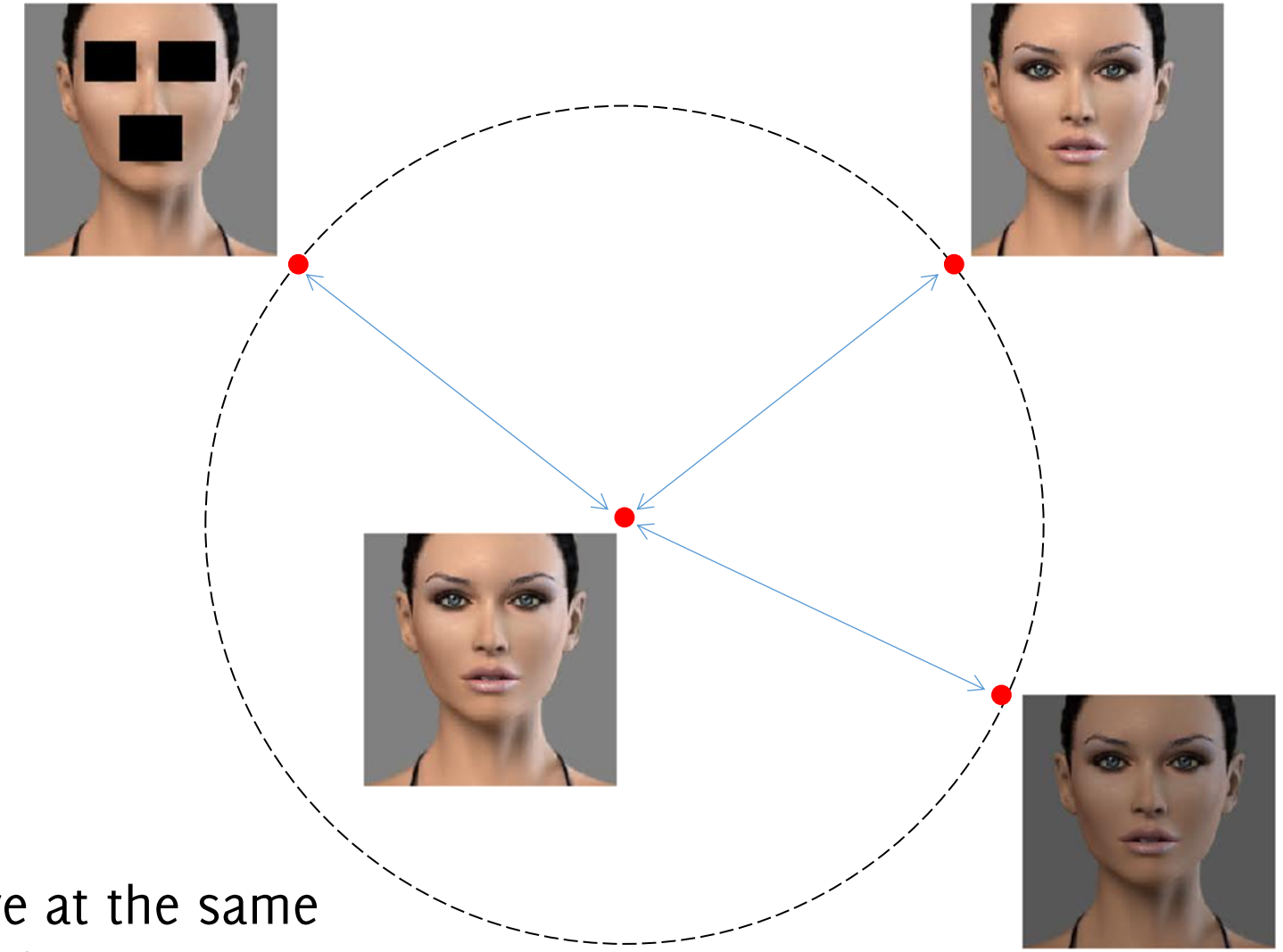
T_1

T_2

T_3

T_4

Bad News…



The three images are at the same distance from the reference at the center

CS231n: Convolutional Neural Networks for Visual Recognition http://cs231n.github.io/

What about image comparison?

… but how to perform identification?

$$\hat{\imath} = \underset{j=1,..,4}{\operatorname{argmin}} \left\| I - T_j \right\|_2$$

I

T_1

T_2

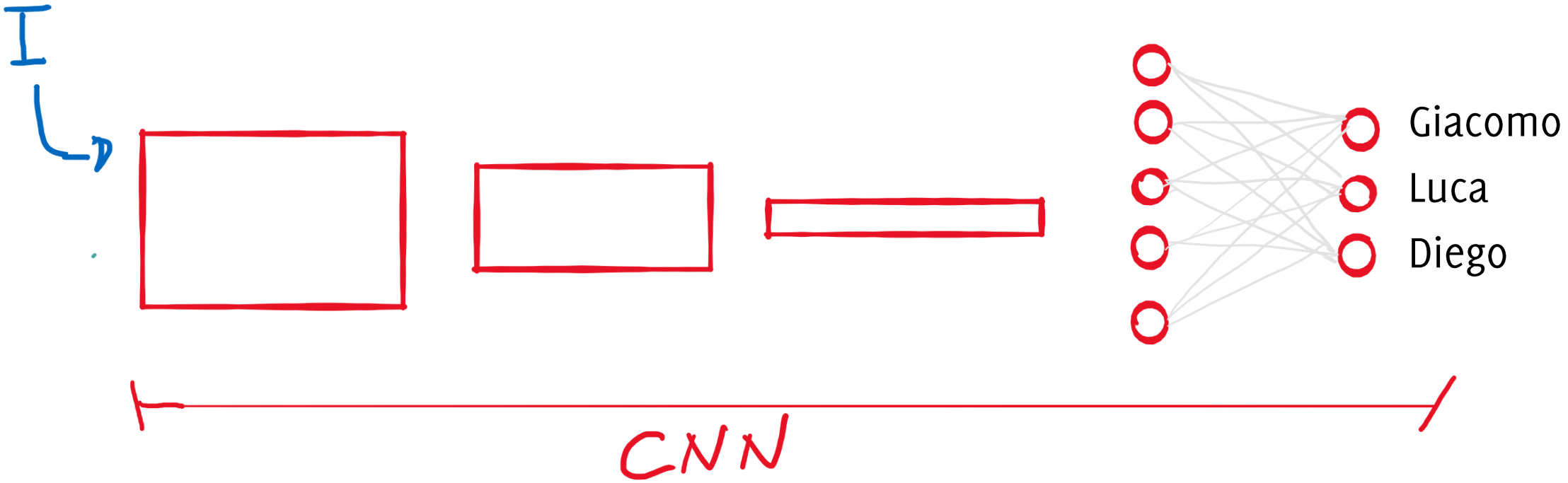
T_3

T_4

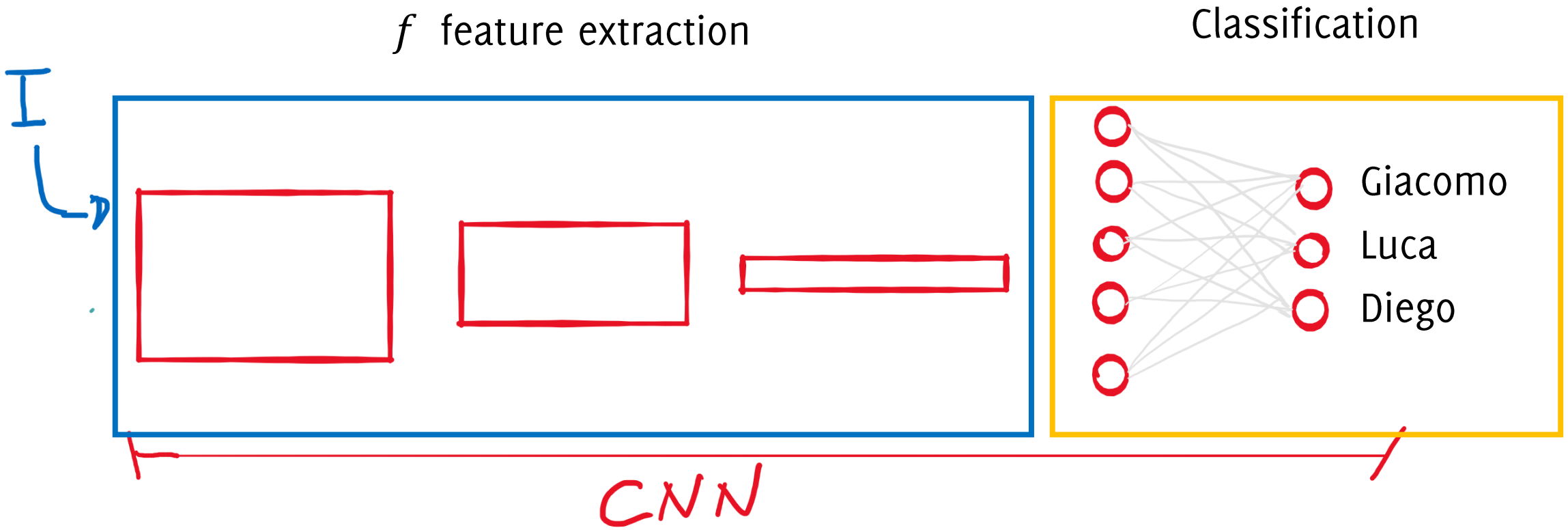
Is it possible to move to a learned distance,
and to train tihs relying only on these examples?

...we need a better distance measure for face identification!

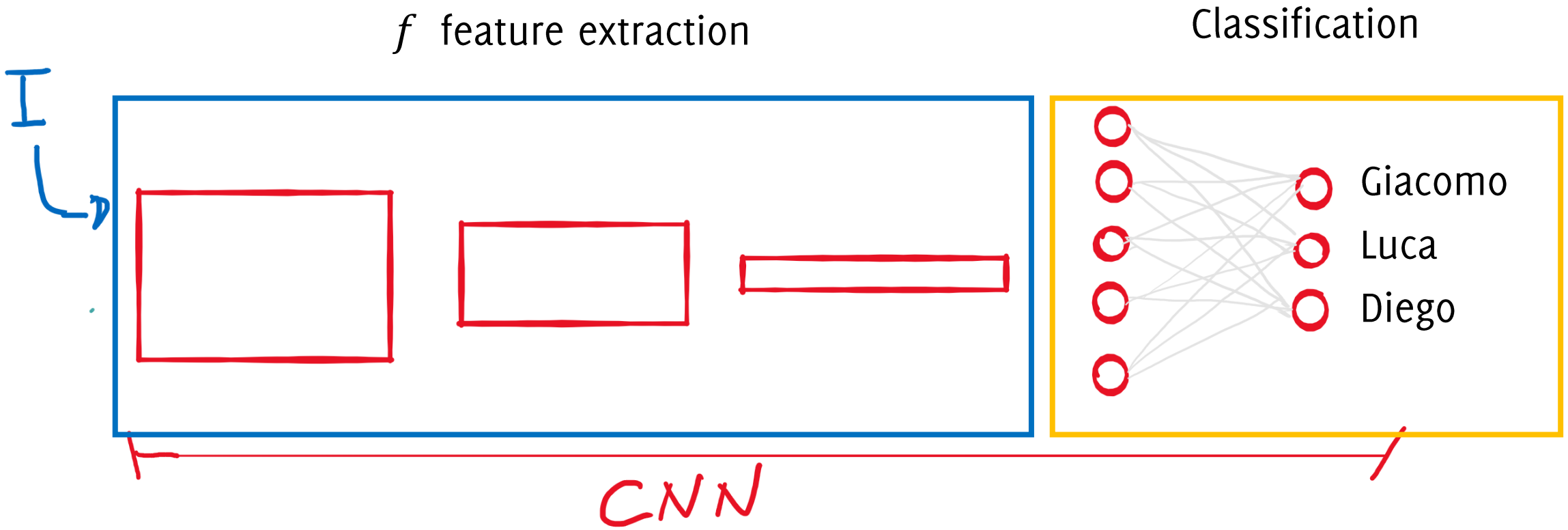
Dissecting CNN



Dissecting CNN

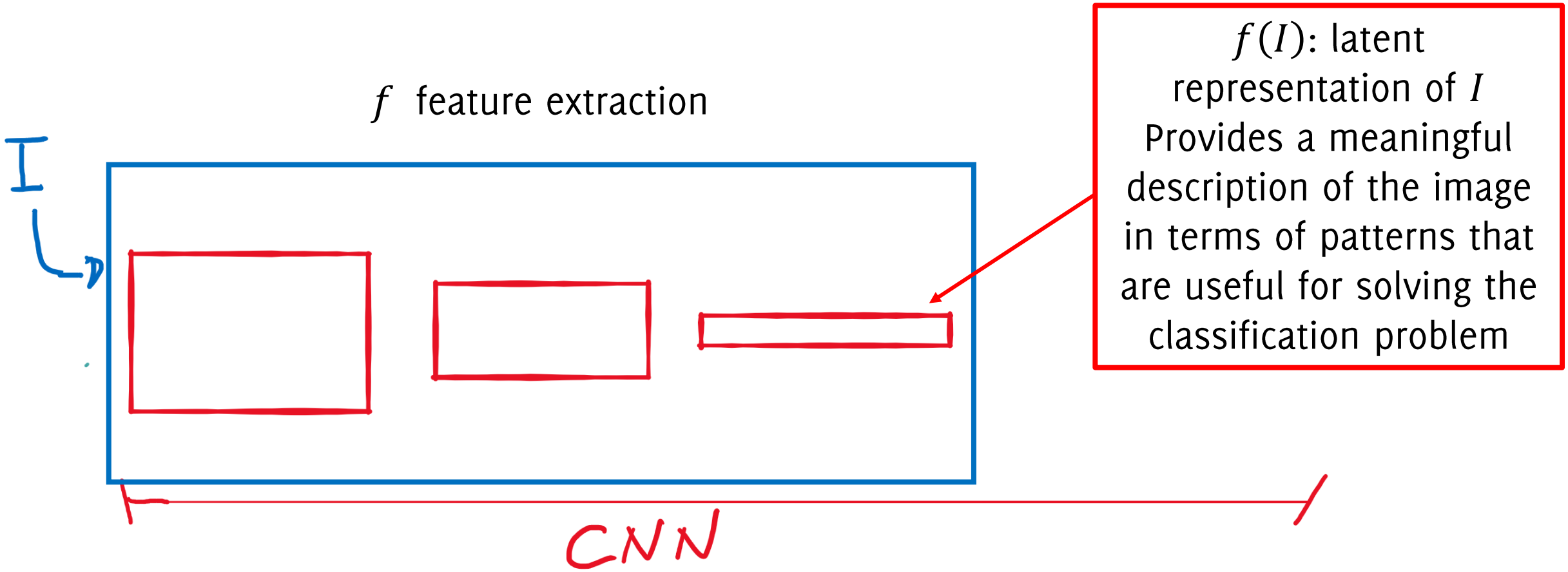


Dissecting CNN



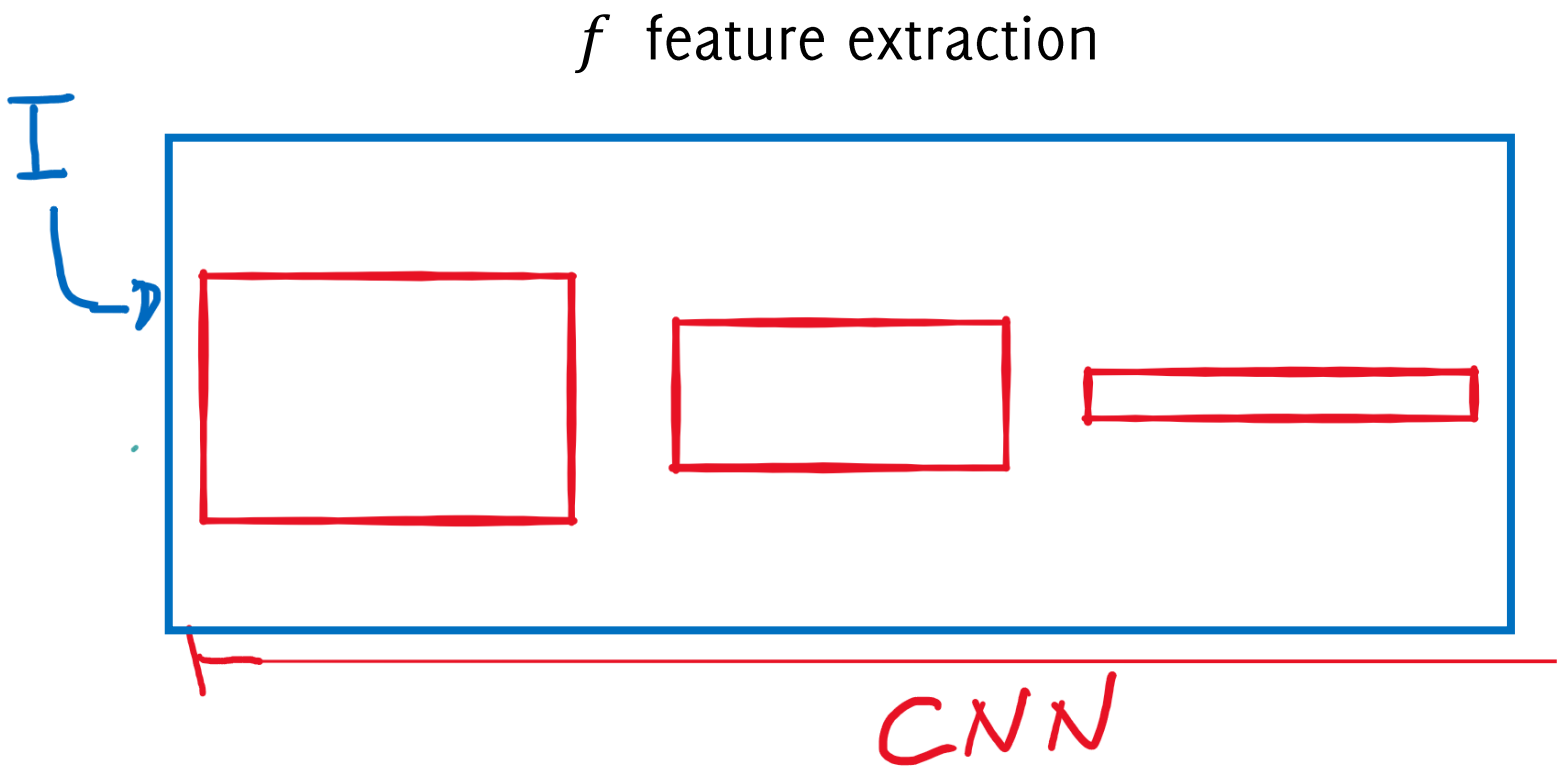
The feature extraction part is typically more general-purpose than the classification part (see Transfer Learning / Domain Adaptation)

Latent representation



$f(I)$: latent representation of I
Provides a meaningful description of the image in terms of patterns that are useful for solving the classification problem

Distance among latent representations



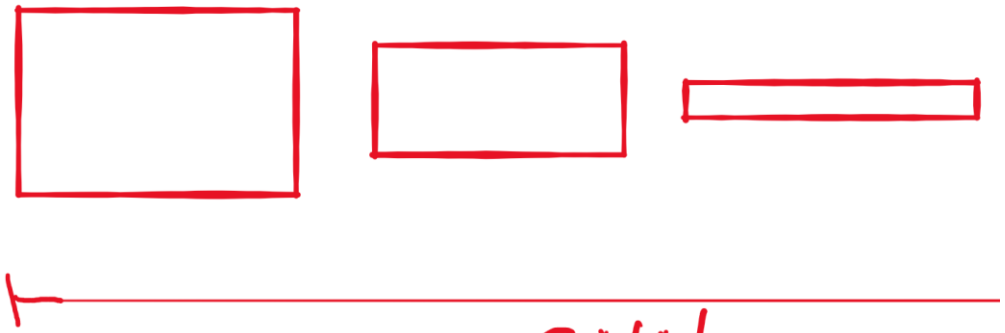
What about using?

$$\hat{\imath} = \underset{j=1,..,4}{\operatorname{argmin}} \left\| f(I) - f(T_j) \right\|_2$$

Distance among latent representations

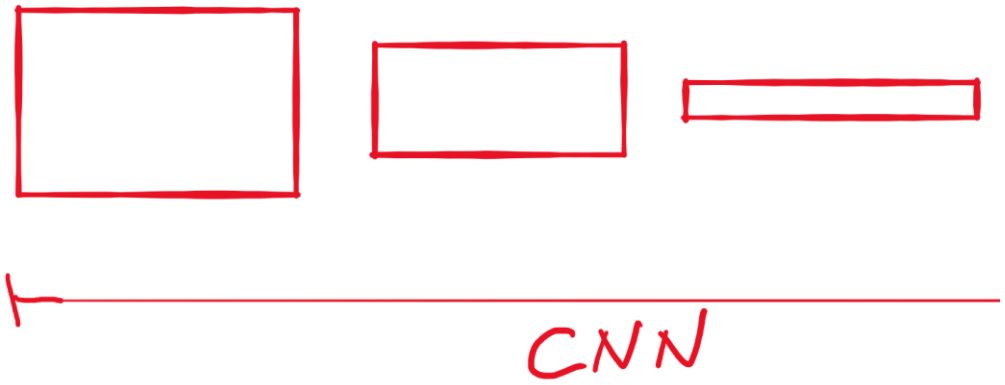
I

f feature extraction



$f(I)$

T_1



$f(T_1)$

$$\|f(I) - f(T_1)\|_2$$

Distance among latent representations

I

f feature extraction

CNN

$f(I)$

T_2

CNN

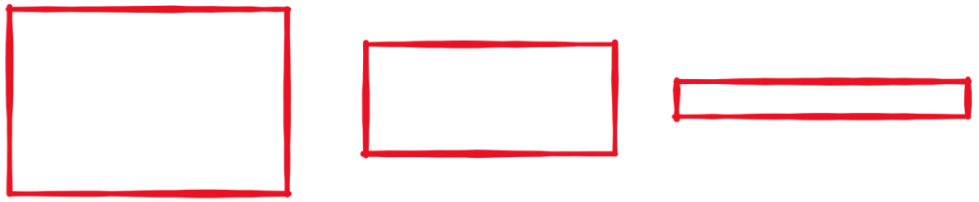
$f(T_2)$

$\|f(I) - f(T_2)\|_2$

Distance among latent representations

I

f feature extraction

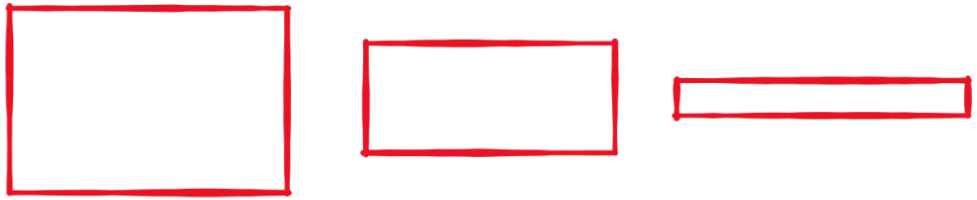


$f(I)$

And hopefully now

$$\|f(I) - f(T_1)\|_2 < \|f(I) - f(T_2)\|_2$$

T_2



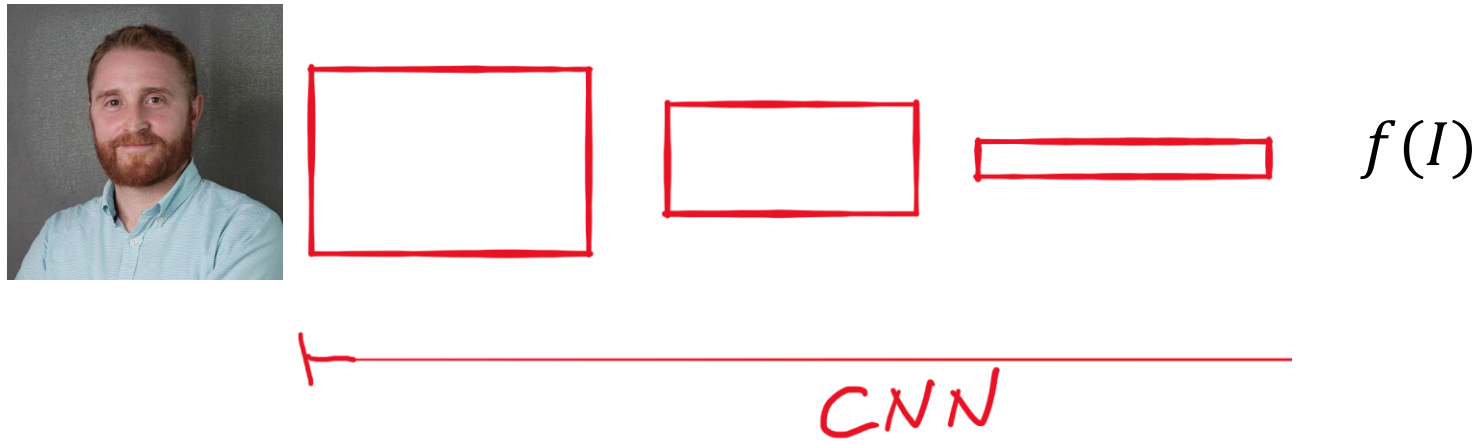
$f(T_2)$

$\|f(I) - f(T_1)\|_2$

CNN

In practice

I f feature extraction

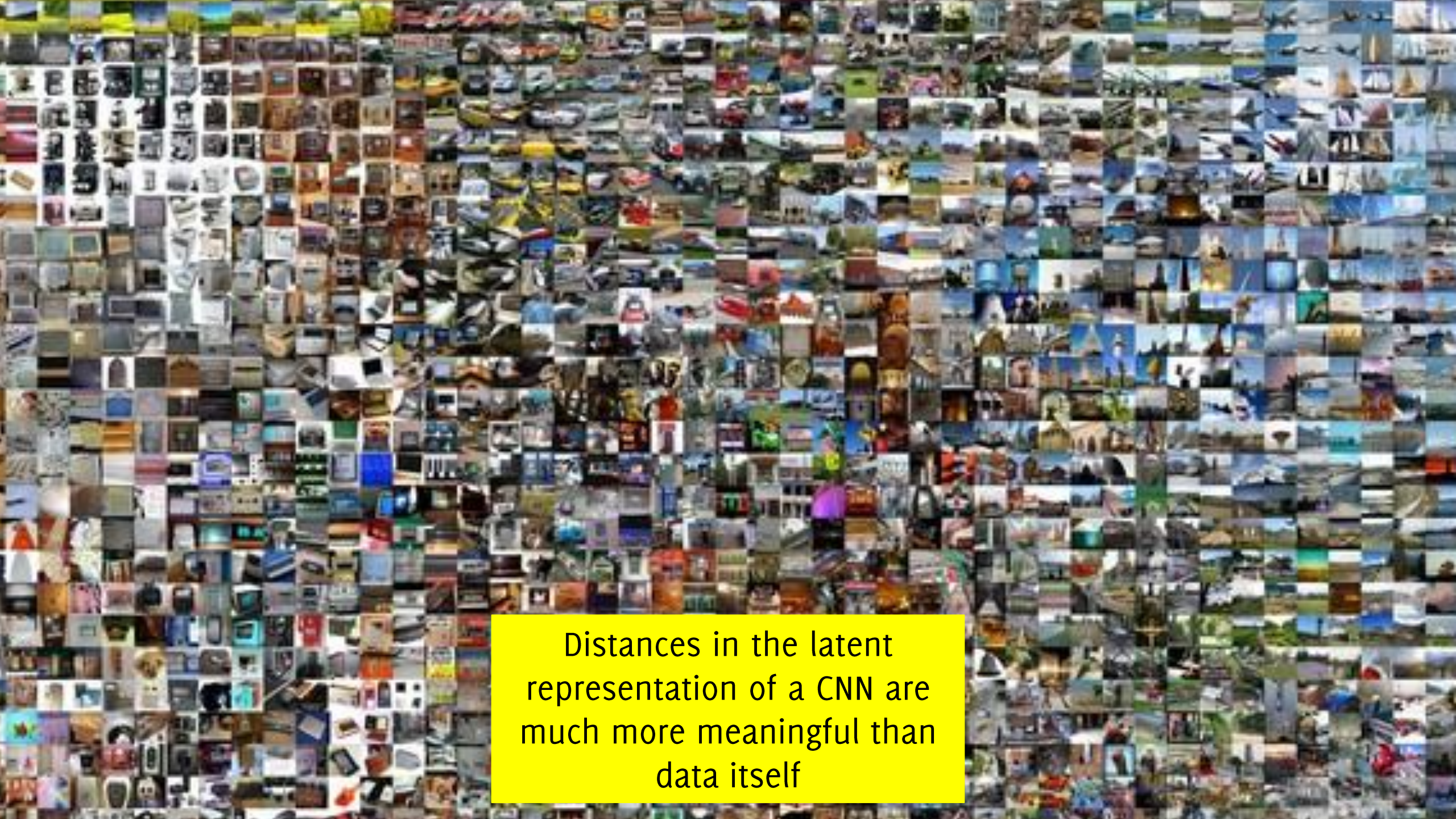


$f(I)$

1. Extract features from the first image $f(I)$
2. Perform identification as $\hat{\imath} = \underset{j=1,..,4}{\operatorname{argmin}} \left\| f(I) - f(T_j) \right\|_2$

No need to compute $f(T_j)\ \forall j$, these can be directly stored

This is equivalent to perform image retrival in the latent space… we know this can work!


Distances in the latent representation of a CNN are much more meaningful than data itself

However… can we do any better?
…. After all, the network $f(\cdot)$ was not trained for this purpose…

Metric Learning

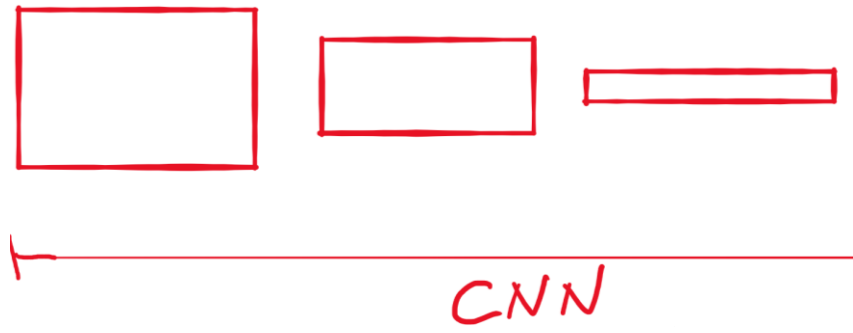
We are still **comparing latent representations trained for classification** (over a few persons?)**, while not for comparing images.**

A more appealing perspective would be to **train the network to measure distances between images.**

We would like to **train the weights** W of our CNN such that

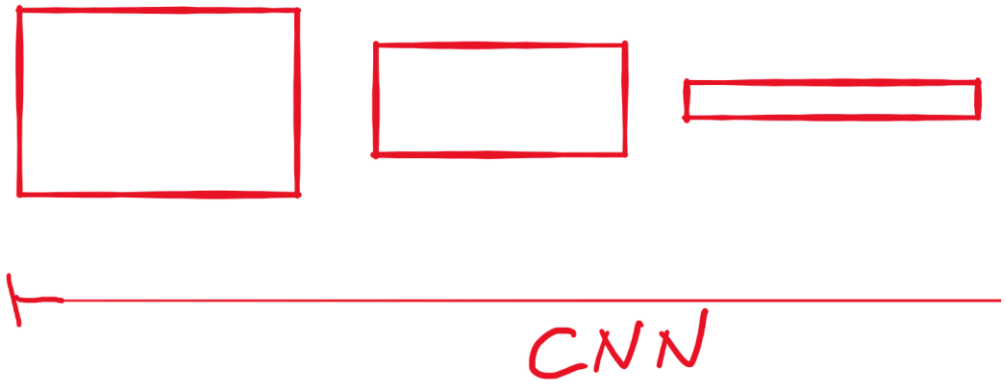
$$\|f_W(I) - f_W(T_i)\|_2 < \left\|f_W(I) - f_W(T_j)\right\|_2 \quad \forall j \neq i$$

When I belongs to class i



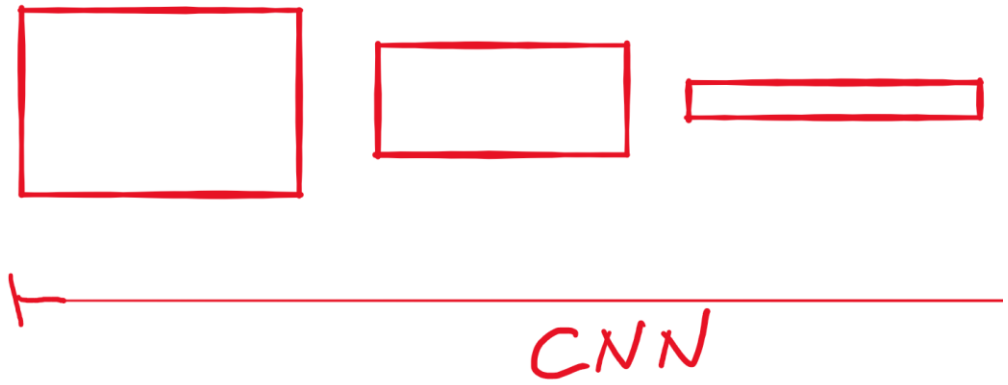
Siamese Networks

I



$f_W(I)$

T_2



$f_W(T_2)$

The «two» networks have to **perform the same operations**, using the same weights W. Hence the term **Siamese**

How to Train the Siamese Network?

During training, the siamese network is fed with **pairs of images (I_i, I_j) that might refer or not to the same individual**

I

CNN

$f_W(I)$

T_2

CNN

$f_W(T_2)$

How to learn the parameters W **provided a training set** $\{I_i, i = 1 \ldots\}$ of many images each one annotated to the respective template?

Contrastive Loss

The *contrastive loss* function is defined as follow:

$$W = \underset{\omega}{\operatorname{argmin}} \sum_{i,j} \mathcal{L}_\omega\left(I_i, I_j, y_{i,j}\right)$$

Where:

$$\mathcal{L}_\omega\left(I_i, I_j, y_{i,j}\right) = \frac{\left(1 - y_{i,j}\right)}{2} \left\| f_\omega(I_i) - f_\omega(I_j) \right\|_2 + \frac{y_{i,j}}{2} \max\left(0, m - \left\| f_\omega(I_i) - f_\omega(I_j) \right\|_2\right)$$

- $y_{i,j} \in \{0,1\}$ is the label associate with the input pair (I_i, I_j):
  - 0 when (I_i, I_j) refers to the same person
  - 1 otherwise
- m is a hyperparameter indicating the margin we want (like in Hinge Loss)
- $\left\| f_\omega(I_i) - f_\omega(I_j) \right\|_2$ is the distance in the latent space.

Triplet Loss

A loss function such that a training sample I is compared against

- P a positive input, referring to the same person
- N a negative input, referring to a different person

We train the network to minimize the distance from the positive samples and maximize the distance from the negative ones

$$\mathcal{L}_\omega(I,P,N) = \max(0, m + (\|f_\omega(I) - f_\omega(P)\|_2 \ - \|f_\omega(I) - f_\omega(N)\|_2))$$

Triplet loss forces that a pair of samples from the same individual are smaller in distance than those with different ones.

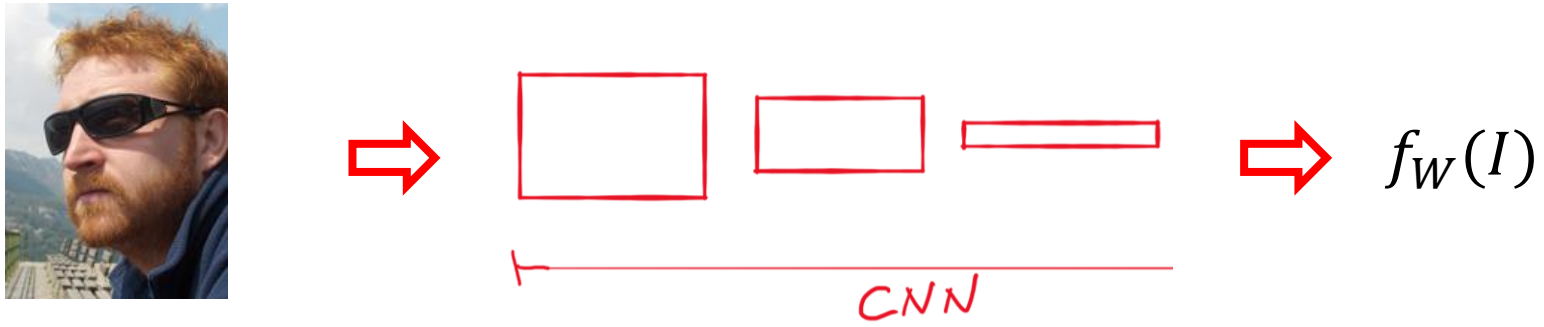
m always play the role of the margin.

The selection of triplets for training is a matter of study

Just to Recap

When a new image has to be verified

1. We feed the image I to the trained network, thus compute $f_W(I)$



2. Identify the person having average minimum distance from templates (in case there are many associated to the same individual)

$$\hat{\imath} = \underset{u}{\operatorname{argmin}} \frac{\sum_{T_{u,j}} \left\| f_W(I) - f_W(T_{u,j}) \right\|_2}{\#\{T_u\}}$$

3. Assess whether

$$\frac{\sum_{T_{i,j}} \left\| f_W(I) - f_W(T_{i,j}) \right\|_2}{\#\{T_{i,j}\}} < \gamma$$

is sufficiently small, otherwise **no identification**

Other decision rules can be adopted (e.g. searching for the person giving the a template with a minimum distance)