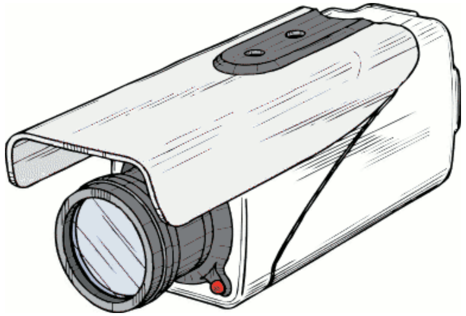
Univerza v Ljubljani

ViCoS
Visual Cognitive Systems Lab

# Machine Perception
# Recognition 2 & object detection




Matej Kristan

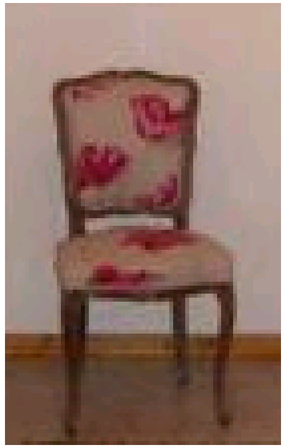



Laboratorij za Umetne Vizualne Spoznavne Sisteme,
Fakulteta za računalništvo in informatiko,
Univerza v Ljubljani

# Is detection/recognition really that difficult?

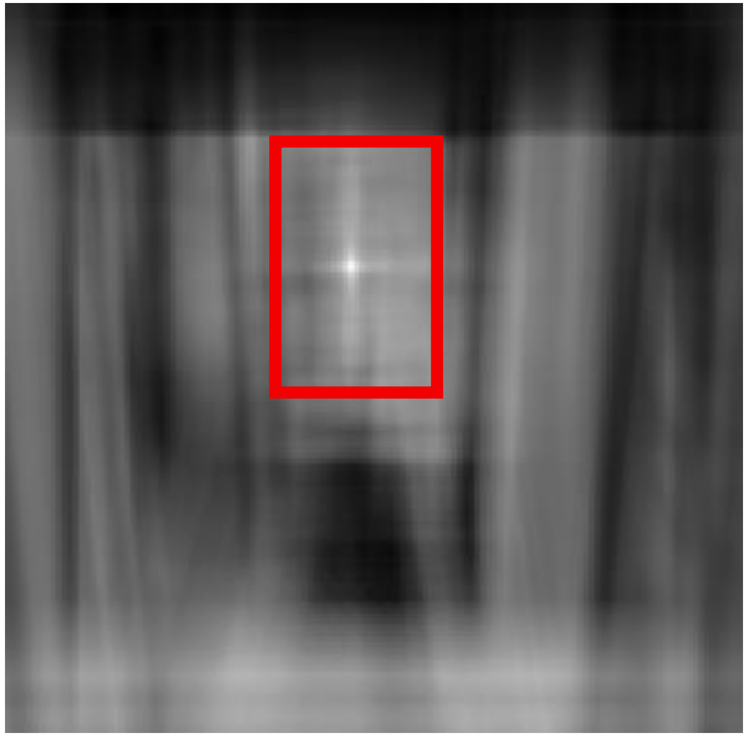
- A simple classifier: Comparison of a candidate patch to a template
- I.e., $\sum_i (I_i T_i) > \theta$ (is the dot product sufficiently large?)

Template: A chair



Find this chair in the image

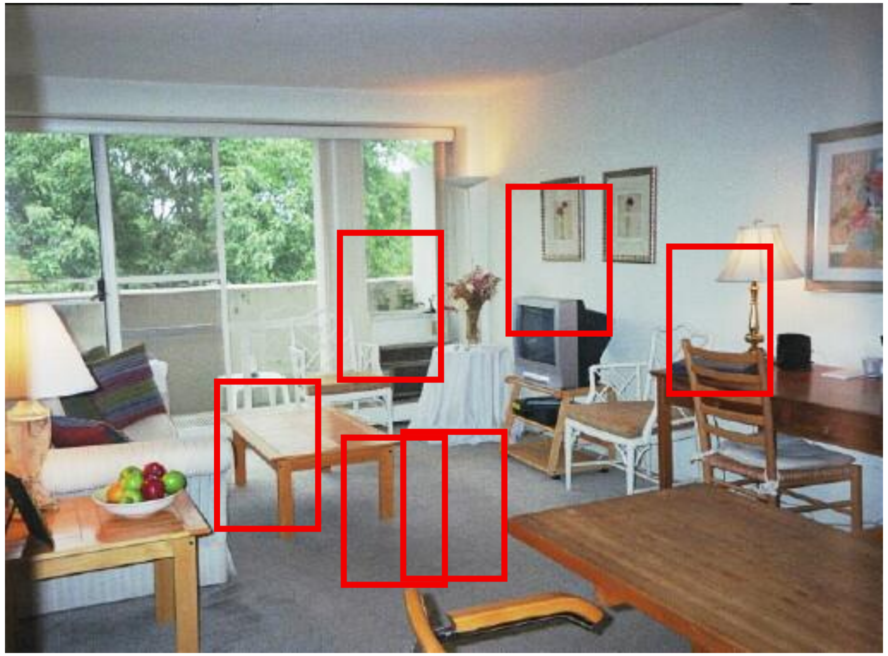
Output of the classifier evaluation.

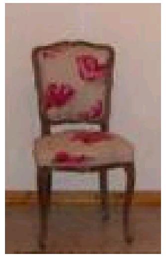


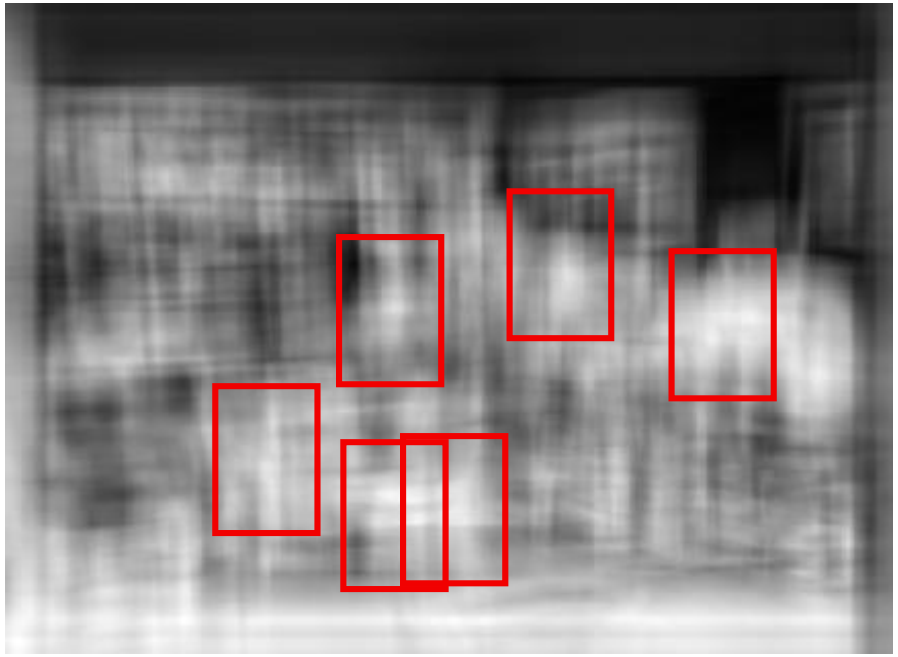
*Not really?!*

# Is detection/recognition really that difficult?

Analyze this!







Completely useless – does not work at all.

Main issue: Poor representation - Feature space!

# Challenges of feature construction

Illumination

Occlusion

Within-class variability

Object pose

Aspect

# How to come up with features?

1. Natural coordinate systems:

   *For some applications, it is enough just to linearly transform the input data.* PCA/LDA

2. Handcrafted nonlinear transforms:

   *Nonlinear transforms improve feature robustness.*

3. Feature selection:

   *Machine learning to select optimal features from a pool of several handcrafted transforms.*

4. End-to-end learning of feature transform:

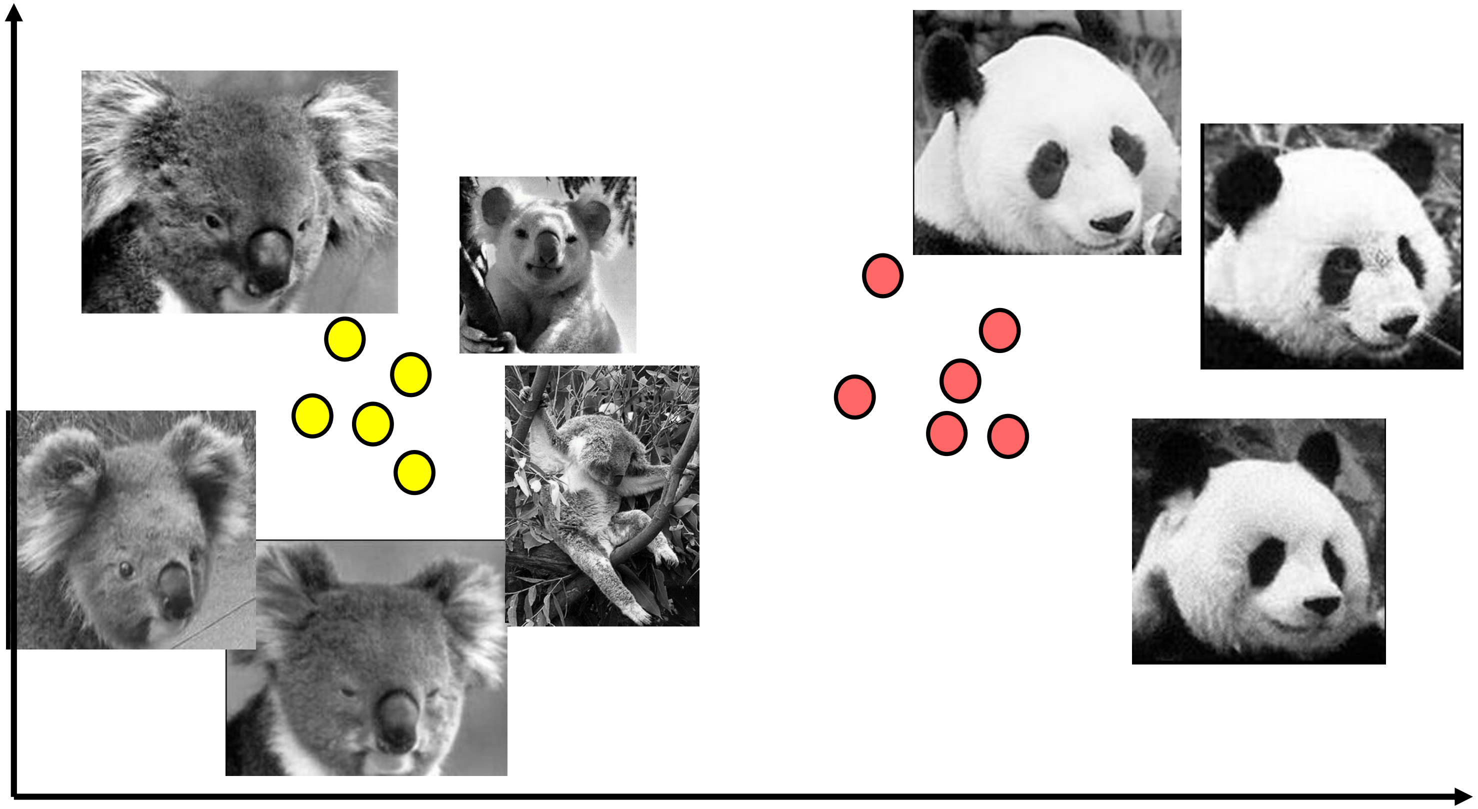
   *Have machine learn entire feature extraction and selection pipeline.*

Machine Perception

# HANDCRAFTED NONLINEAR TRANSFORMS

# Hand-crafting global features

- Require a representation that:
  - Accounts for intra-class variation
  - Distinguishes between different classes

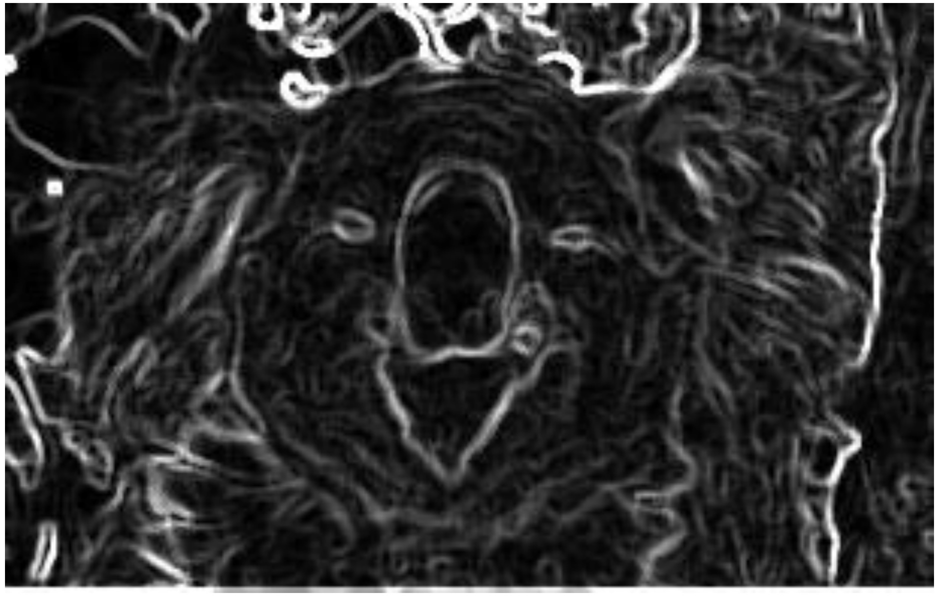


# Hand-crafting global features

- Problem: Color or gray-level representation is sensitive to illumination changes or within-class color variations.

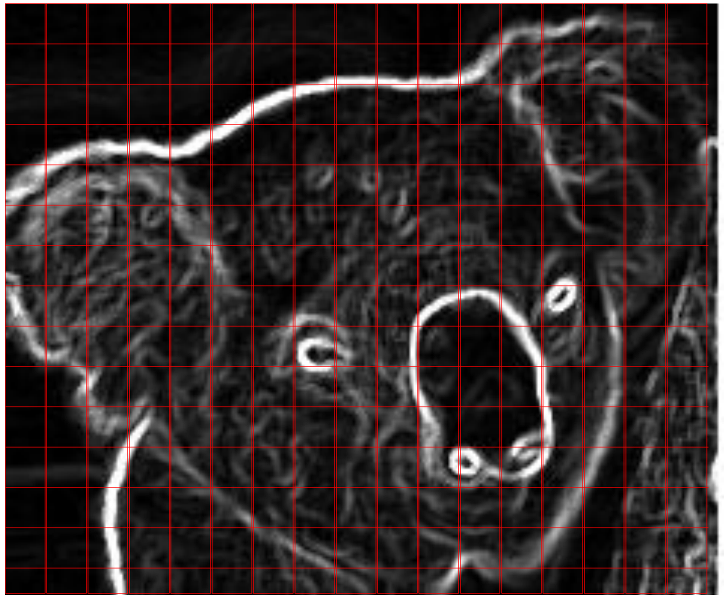
# Hand-crafting global features

- Solution: Edges, contours and oriented intensity gradients
- Change intensity to gradient-based features



# Hand-crafting global features

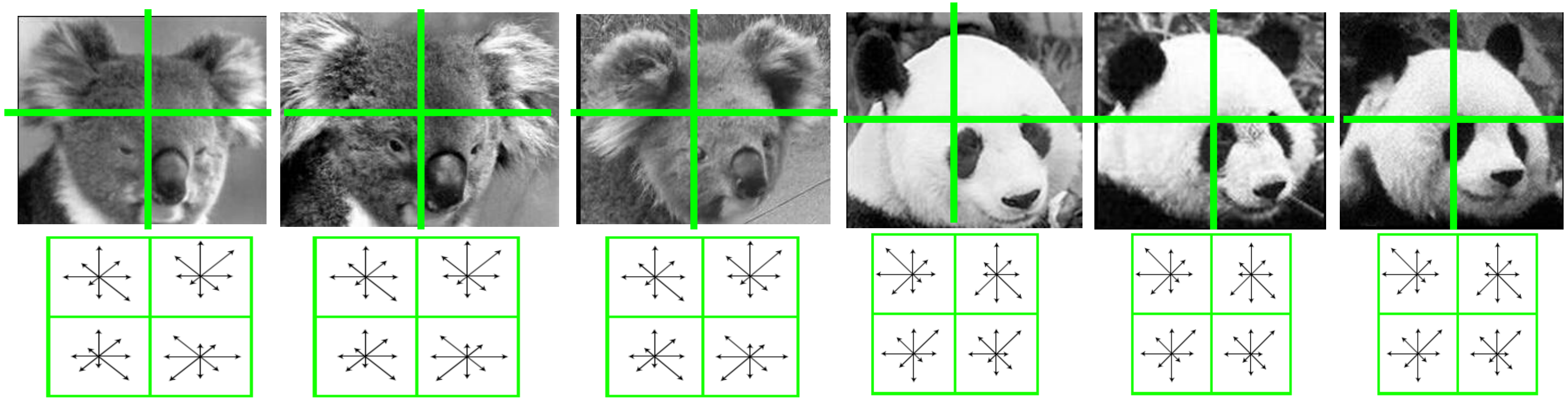
- Global descriptor: Templates vs Histograms



- Templates: often too specific – not robust to local deformations
- Histograms: robust to local deformations, but not enough specific

# Gradient-based representation

- Edges, contours and oriented intensity gradients



- Encode local gradient distributions using histograms
  - Locally unordered: invariant to small shifts and rotations
  - Contrast normalization: addresses non-uniform illumination and varying intensity.

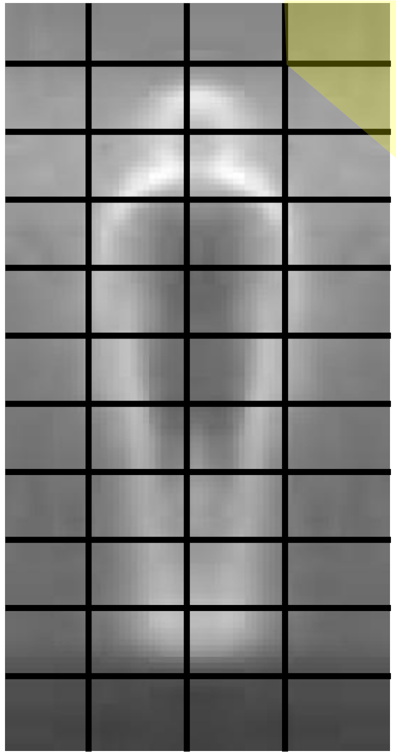
# Gradient-based representation: HOG

Input image → Normalize gamma & colour → Compute gradients → Weighted vote into spatial & orientation cells → Contrast normalize over overlapping spatial blocks → Collect HOG's over detection window

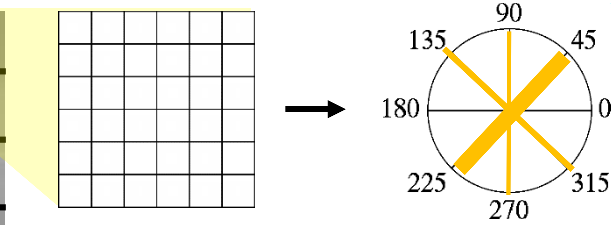
Navneet Dalal and Bill Triggs , Histograms of Oriented Gradients for Human Detection, CVPR 2005

Image

Sobel (gradient)

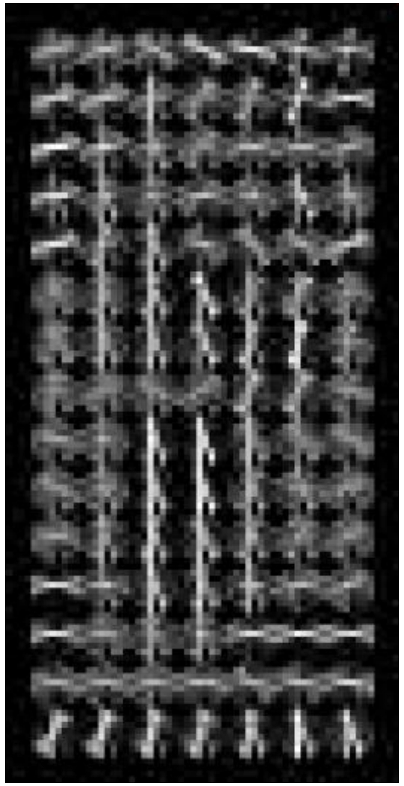


Calculate hog in $8 \times 8$ blocks and normalize




- Histogram of gradient orientations
  - Weighted by magnitude
  - Similar to SIFT

HOG descriptor

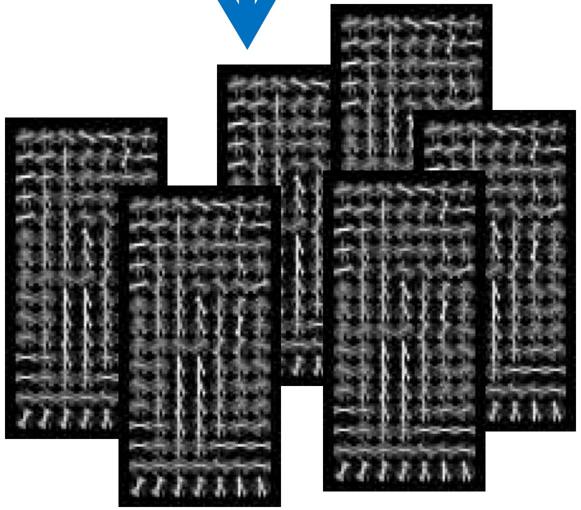


# Practical approach to learning a detector

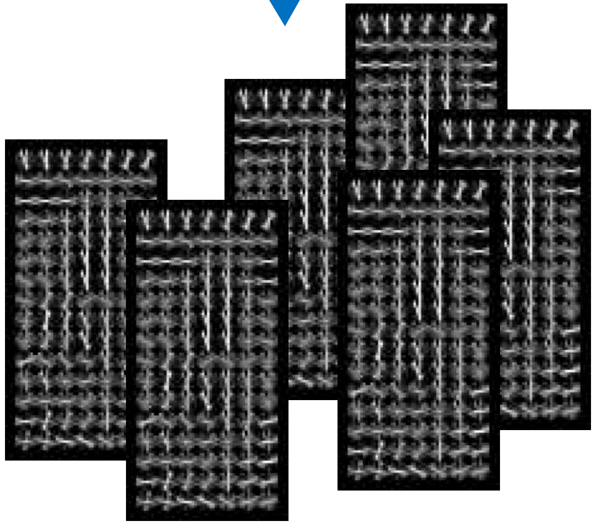
Persons

Background

Extract HOGs



Train a person/nonperson classifier



Extract HOGs

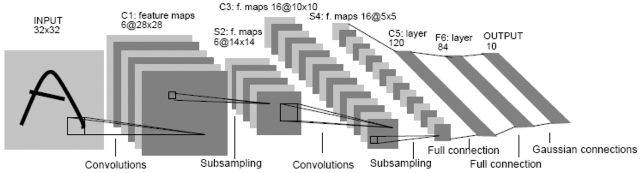
# Lots of choices for a classifier

## Nearest neighbor

$10^6$ examples

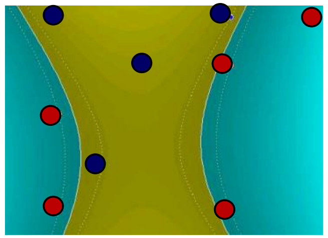
Shakhnarovich, Viola, Darrell 2003
Berg, Berg, Malik 2005...

## Neural networks



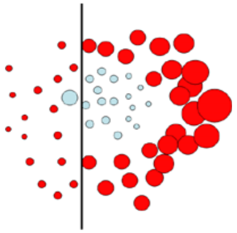

LeCun, Bottou, Bengio, Haffner 1998
Rowley, Baluja, Kanade 1998
…

## Support Vector Machines



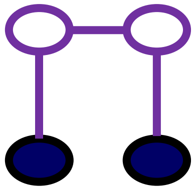
Guyon, Vapnik
Heisele, Serre, Poggio, 2001,…

## Boosting



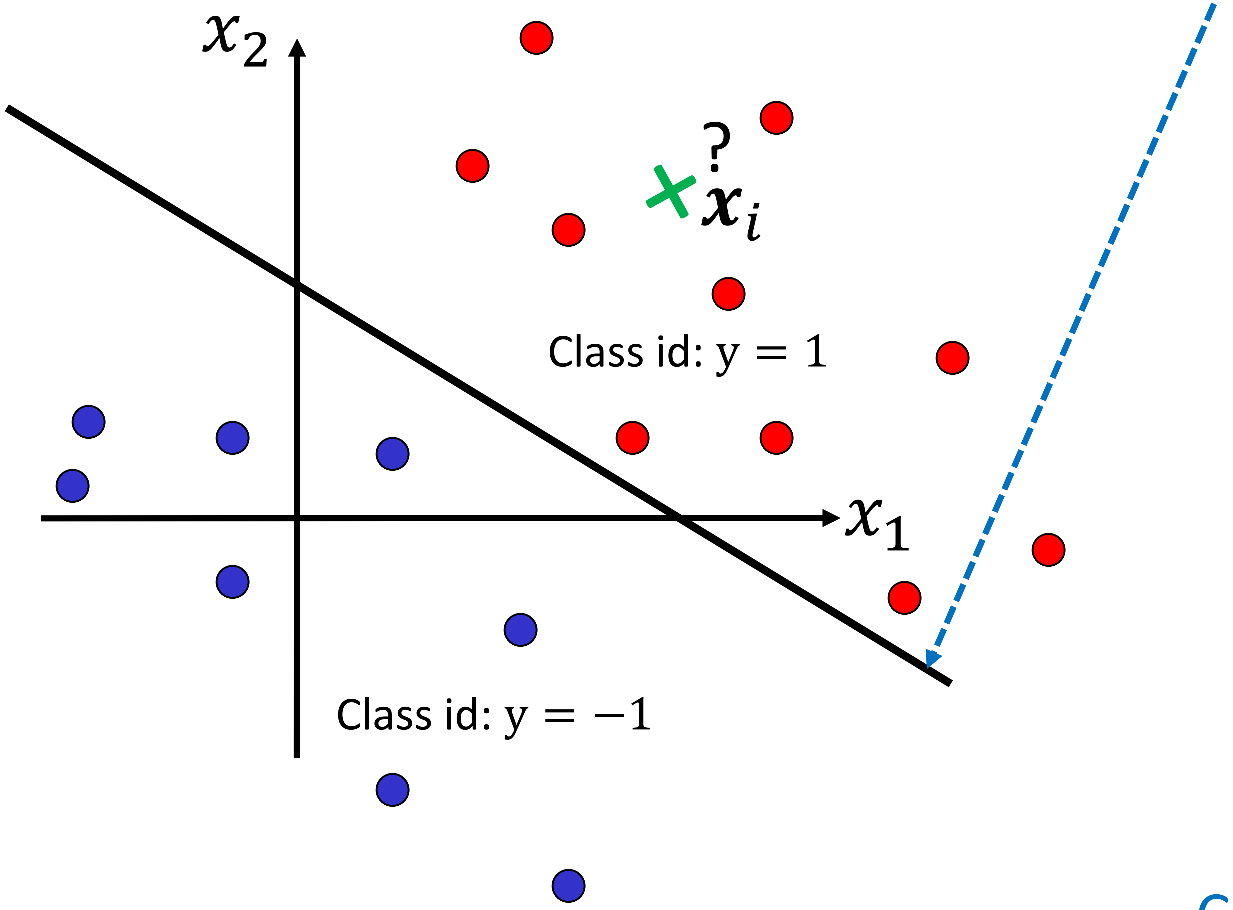
Viola, Jones 2001,
Torralba et al. 2004,
Opelt et al. 2006,…

## Conditional Random Fields



McCallum, Freitag, Pereira 2000; Kumar, Hebert 2003, …

# Consider a linear classifier




Learning = Choosing $\boldsymbol{w}$ and $b$!

A decision boundary, in general, a *hyper-plane*:

$$ax_1 + cx_2 + b = 0$$

Define:

$$\mathbf{w} = \begin{bmatrix} a \\ c \end{bmatrix} \qquad \mathbf{x} = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}$$

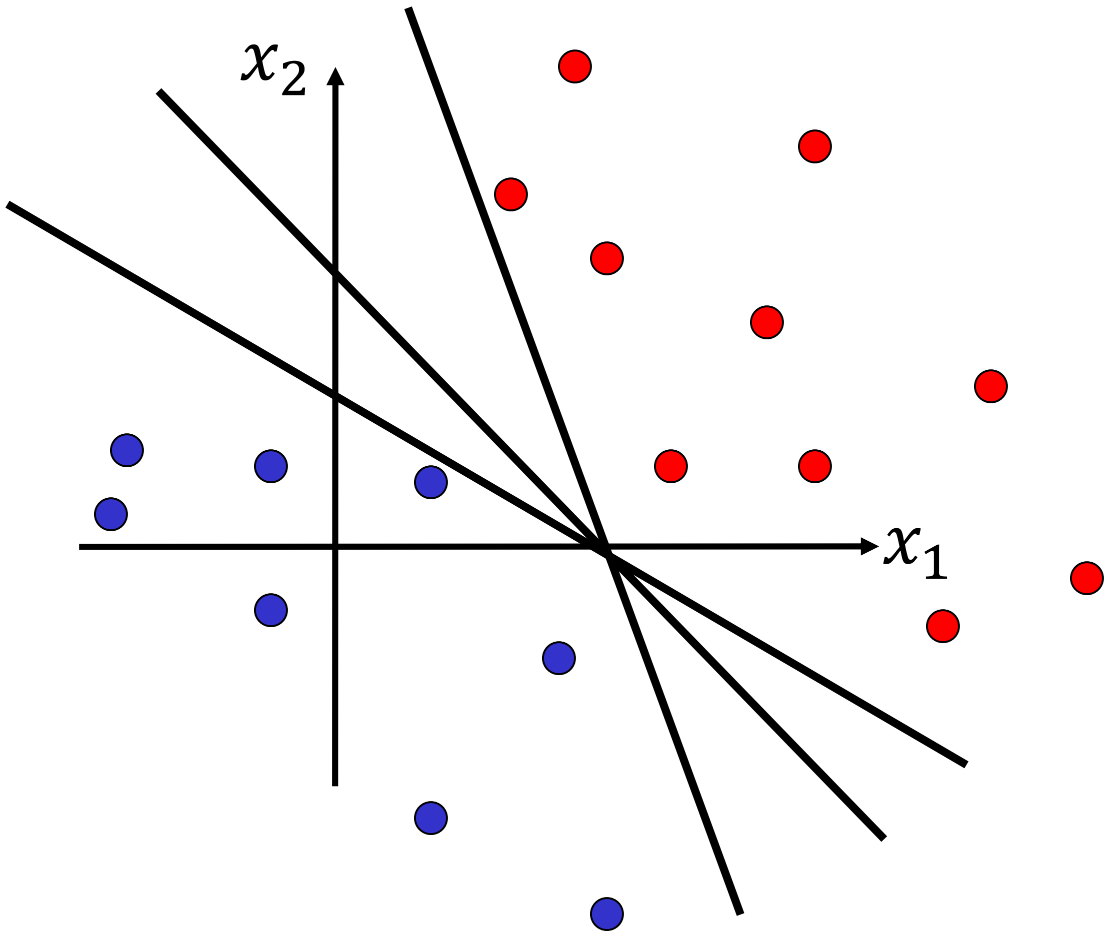
A general hyper-plane eq:

$$\mathbf{w}^T \mathbf{x} + b = 0$$

Classification of x = sign checking:

$$f(\mathbf{x}) = \text{sign}\,(\mathbf{w}^T \mathbf{x}_i + \mathrm{b})$$

# Best separation hyper-plane?




A general hyper-plane eq:

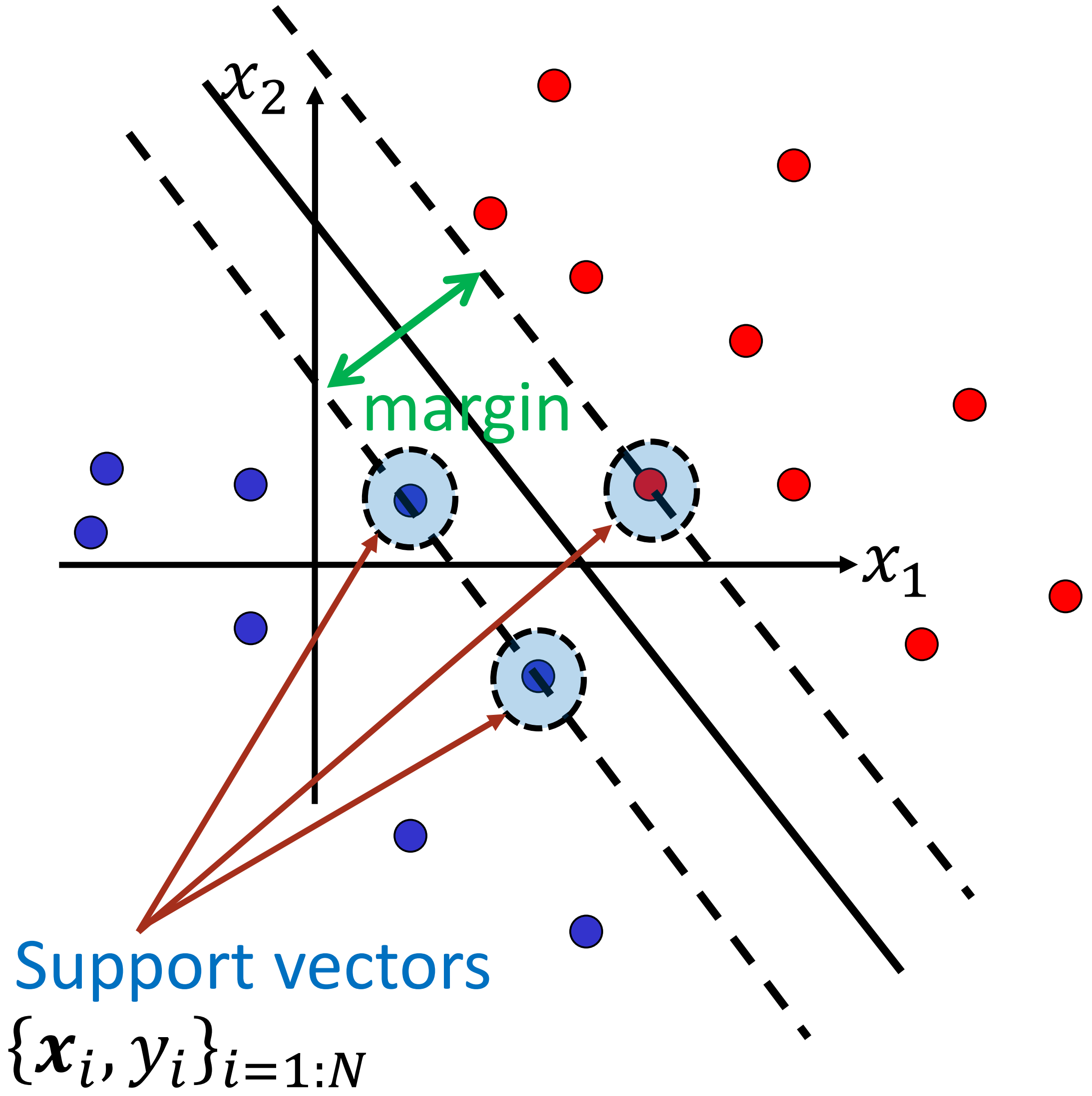
$$\mathbf{w}^T\mathbf{x} + b = 0$$

Classification of x = sign checking:

$$f(\mathbf{x}) = \text{sign}(\mathbf{w}^T\mathbf{x} + \text{b})$$

Choosing $\boldsymbol{w}$ and $b$?

# Best separation hyper-plane?




A general hyper-plane eq:

$$\mathbf{w}^T\mathbf{x} + b = 0$$

Classification of x = sign checking:

$$f(\mathbf{x}) = \text{sign}(\mathbf{w}^T\mathbf{x} + \text{b})$$

- The hyper-plane that maximizes the margin between positive ($y_i = 1$) and negative ($y_i = -1$) training examples.

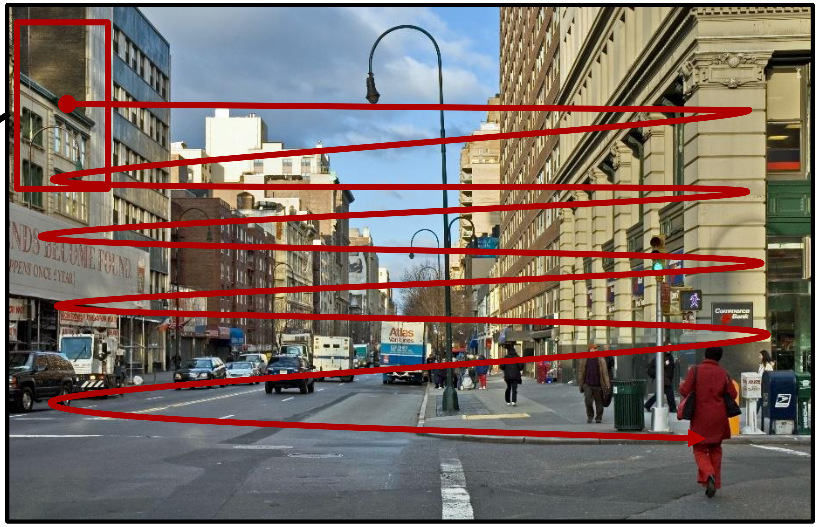
$$\mathbf{w} = \sum_{i=1}^{N} \alpha_i y_i \mathbf{x}_i$$

Have to select SVs and learn $\alpha_i$s!

# Application: Pedestrian detection

- Sliding window:

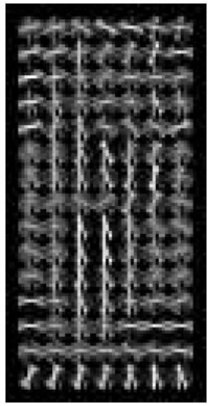
1. extract HOG at each displacement
2. classify by a linear SVM



$\boldsymbol{x}$

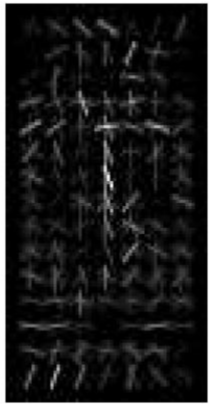
$$f(x) = \text{sign}\,(\mathbf{w}^T\mathbf{x} + \mathrm{b})$$

input



HOG

HOG cells weighted by the positive support vectors



HOG cells weighted by the negative support vectors

Dalal and Triggs, Histograms of oriented gradients for human detection, CVPR2005

# Pedestrian detection HoG+SVM


Navneet Dalal, Bill Triggs, Histograms of Oriented Gradients for Human Detection, CVPR 2005

# Time/computation criticality

- A lot of applications are time- and resources-critical
- Require efficient feature construction
- Require efficient classification
- A case study: Face detection

# How to come up with features?

1. Natural coordinate systems:

   *For some applications, it is enough just to linearly transform the input data.* PCA/LDA

2. Handcrafted nonlinear transforms:

   *Nonlinear transforms improve feature robustness.* HOG

3. Feature selection:

   *Machine learning to select optimal features from a pool of several handcrafted transforms.*

4. End-to-end learning of feature transform:

   *Have machine learn entire feature extraction and selection pipeline.*

Machine Perception

# LEARNING FEATURES BY FEATURE SELECTION

# Face detection

## Application specifics:

- Frontal faces are a good example, where the global appearance model + sliding window works well:
  - Regular 2D structure
  - Central part of the face is well approximated by rectangle.

Images from: Kristen Grauman

# Fast face detection

- To apply in real-time applications
  1. Feature extraction should be fast
  2. Classifier application should be fast
- These points addressed next

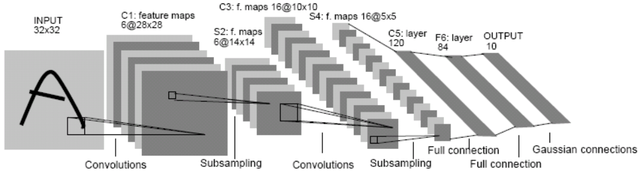
# Choosing the right classifier

## Nearest neighbor

$10^6$ examples

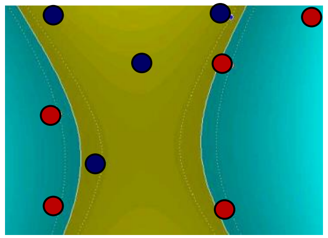
Shakhnarovich, Viola, Darrell 2003
Berg, Berg, Malik 2005...

## Neural networks



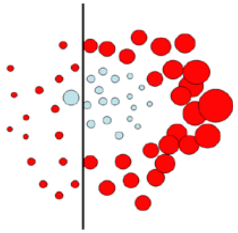

LeCun, Bottou, Bengio, Haffner 1998
Rowley, Baluja, Kanade 1998
…

## Support Vector Machines



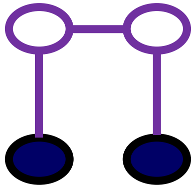
Guyon, Vapnik
Heisele, Serre, Poggio, 2001,…

## Boosting



Viola, Jones 2001,
Torralba et al. 2004,
Opelt et al. 2006,…

## Conditional Random Fields



McCallum, Freitag, Pereira 2000; Kumar, Hebert 2003, …

# Boosting

- Build a strong classifier from a combination of many "weak classifiers" – weak learners (each at least better than random)
- Flexible choice of weak learners
  - This includes fast but inaccurate classifiers!

- We'll have a look at the AdaBoost (Freund & Schapire)
  - Simple to implement.
  - Basis for the popular Viola-Jones face detector.

Y. Freund and R. Schapire, A short introduction to boosting, *Journal of Japanese Society for Artificial Intelligence*, 14(5):771-780, 1999.

# AdaBoost: Intuition

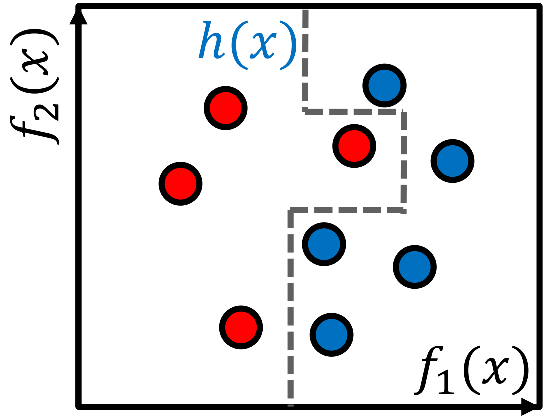
- Task: Build a classifier which is a weighted sum of many classifiers

Final (strong) classifier

weight

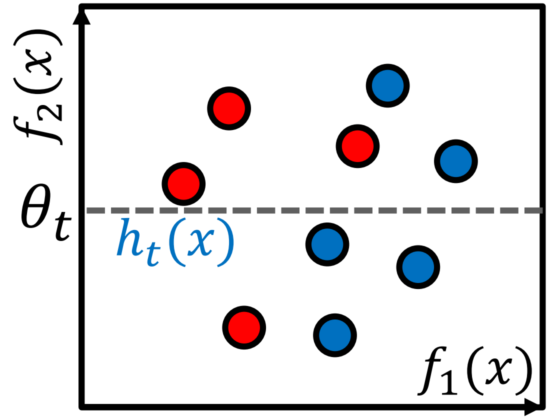
weak classifier

$$h(x) = \operatorname{sgn}\left(\sum_{t=1}^{T} \alpha_t h_t(x)\right)$$

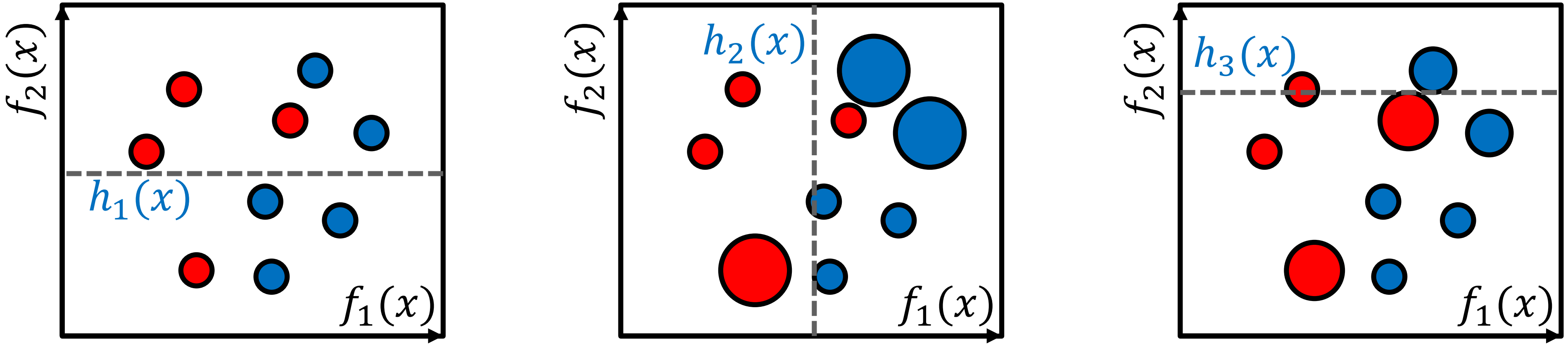



Example of a weak classifier:

$$h_t(x) = \begin{cases} +1 & \text{if } f_t(x) > \theta_t \\ -1 & \text{otherwise} \end{cases}$$



# AdaBoost: Intuition




…

- Train a sequence of weak classifiers.
- Each weak classifier splits training examples with at least 50% accuracy.
- Those examples that are incorrectly classified by the weak classifier, get more weight in training the next weak classifier.

The final classifier is a combination of many weak classifiers!

$$h(x) = \operatorname{sgn}\left(\sum_{t=1}^{T} \alpha_t h_t(x)\right)$$

# Face detection

- To apply in real-time applications

1. Feature extraction should be fast

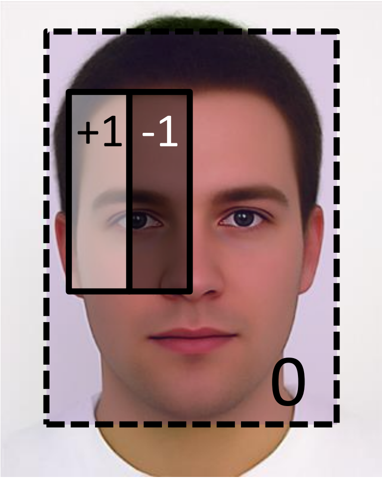
   (? How to calculate fast/strong features ?)

1. Classifier application should be fast
   (weak classifiers = fast evaluation)

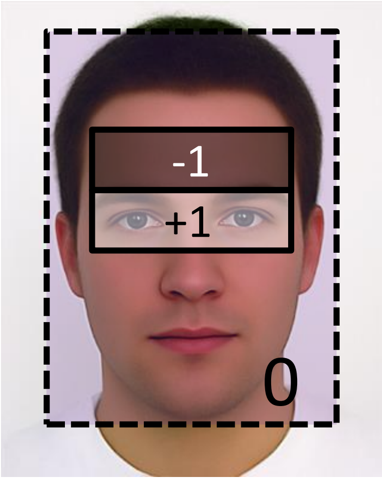
Viola, Jones, *"Rapid Object Detection using a Boosted Cascade of Simple Features"*, CVPR2001

# Computing features

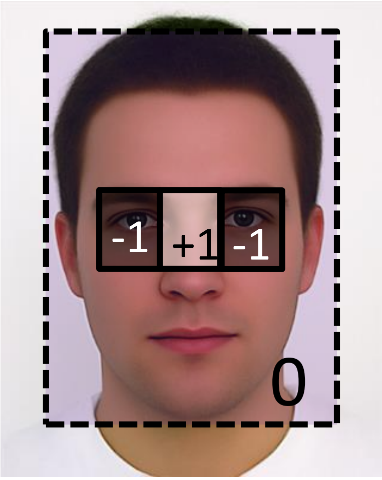
Simple rectangular filters as feature extractors (feature defined by filter type and position)




$f_1(x)$



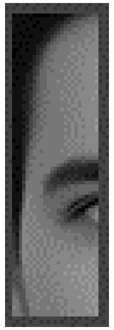

$f_2(x)$




$f_3(x)$

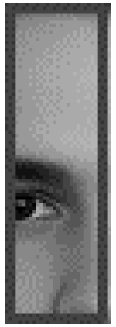

$f_4(x)$

…



Sum=1500

+



Sum=2000

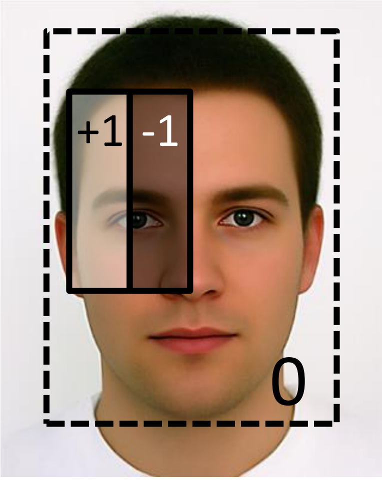
-

$f_1(x) = -500$

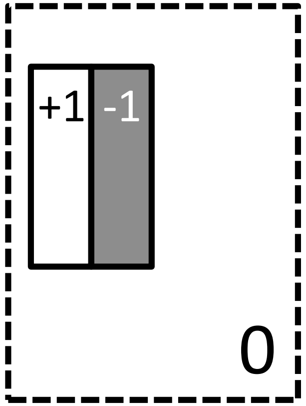
The value of each feature is the difference between the intensity in „black“ and „white“ regions.
Black is weighted as -1, white as +1.

# Computing features

Simple rectangular filters as feature extractors




$f_1(x)$

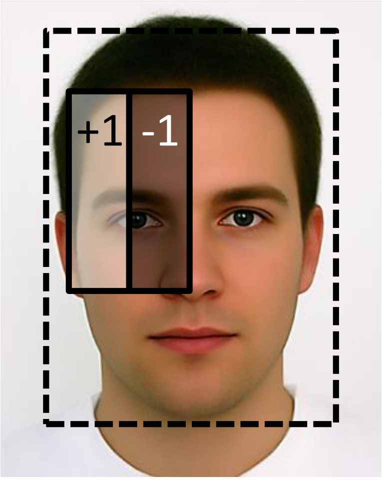


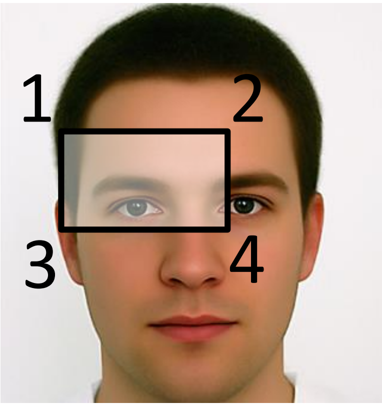

Require evaluation at many displacements and multiple scales!

Possible to evaluate such a simple filter efficiently!

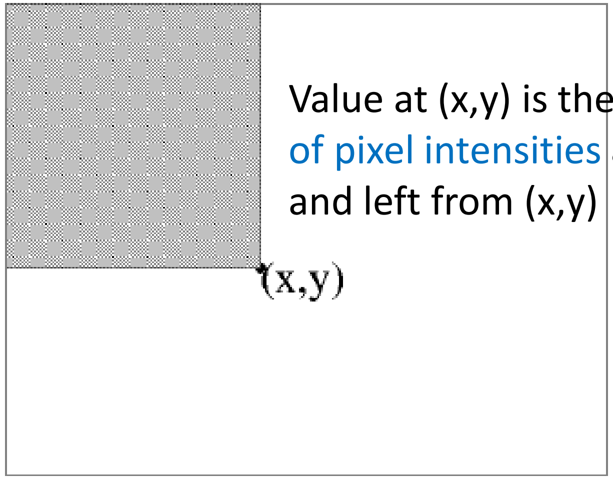
# Efficient computation – Integral images

- Our filters are based on sums of intensities within rectangular regions.
- This can be done in constant time for arbitrary large region!
- Require precomputing integral image.



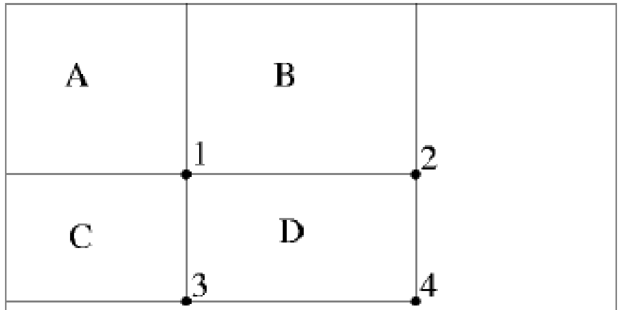




Integral image




Value at (x,y) is the sum of pixel intensities above and left from (x,y)

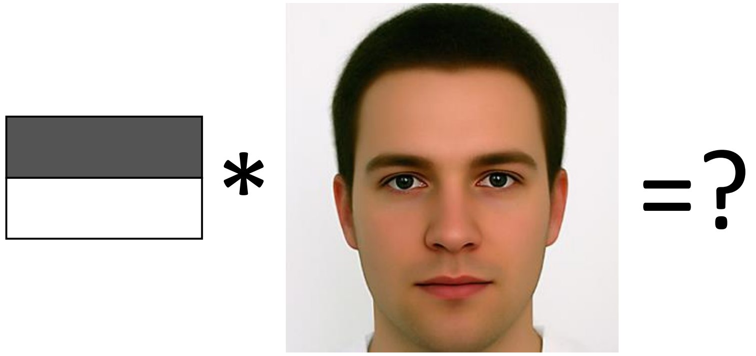
Integral image

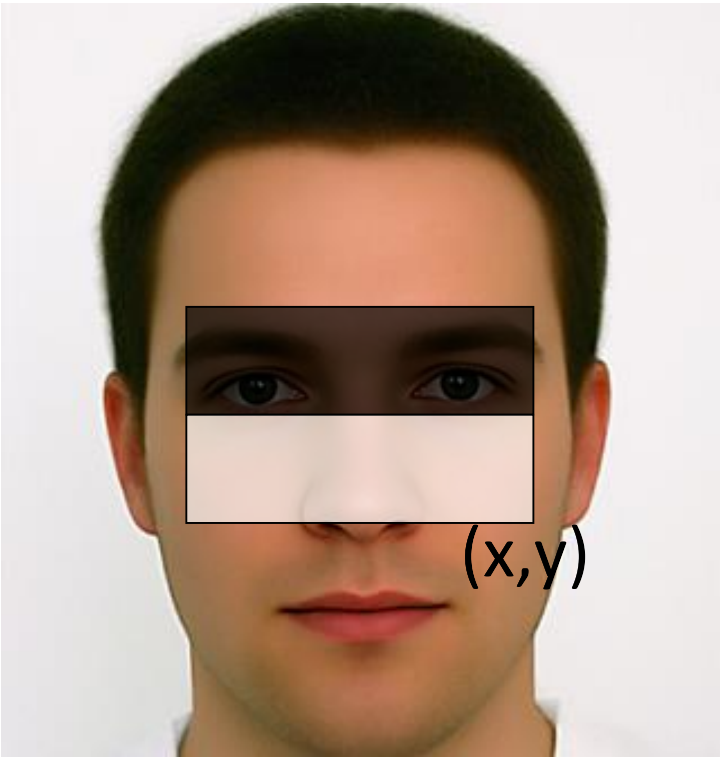


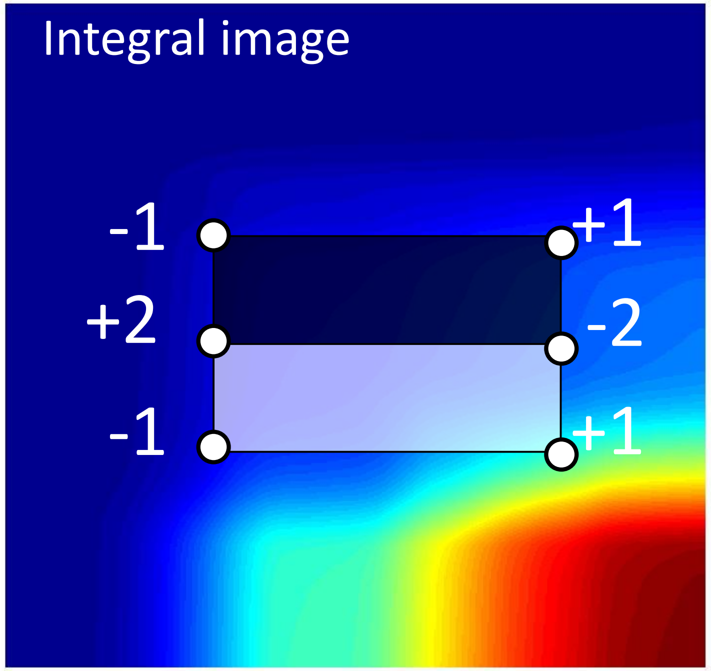

$$
\begin{aligned}
D &= 1+4-(2+3) \\
&= A+(A+B+C+D)-(A+C+A+B) \\
&= D
\end{aligned}
$$

# Efficient computation – Integral images

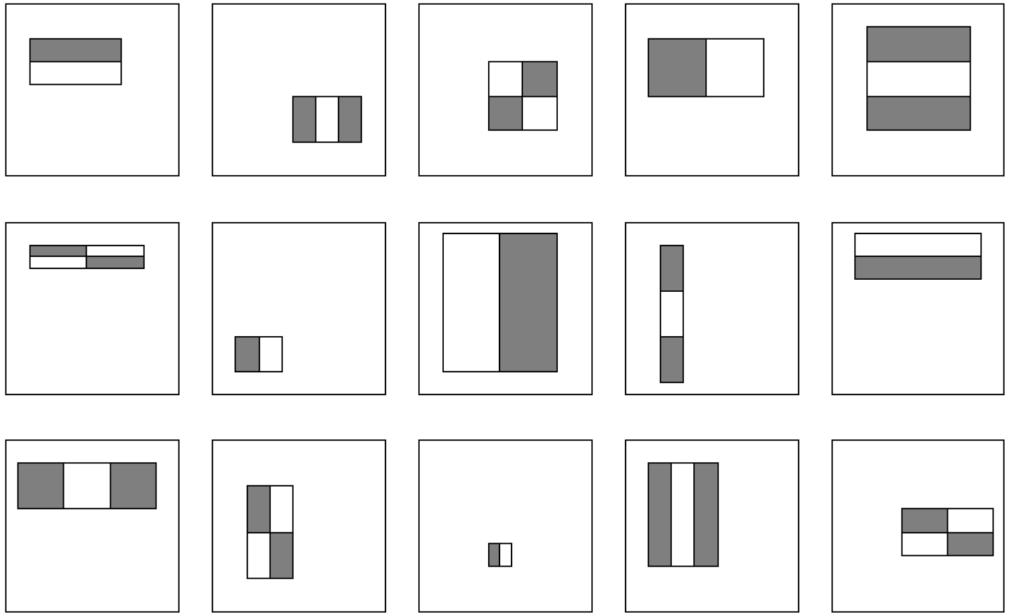
- Consider a more complex filter



# Large collection of filters



etc...

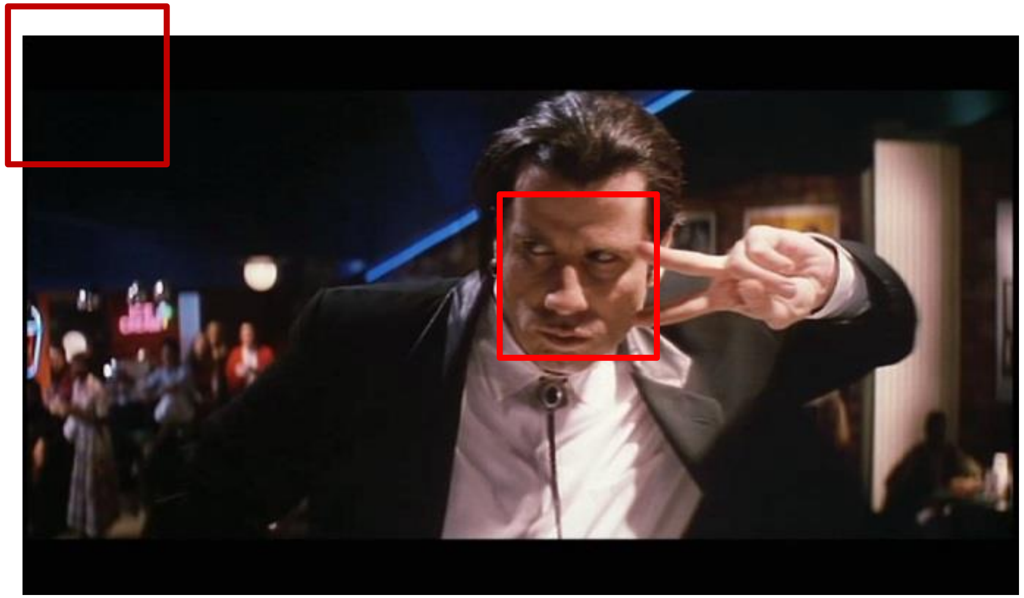
Account for all possible parameters:
position, scale, type

More than 180,000
different features in a 24x24 window.

Apply Adaboost for
(i) selecting most informative features and
(ii) composing a classifier (weights+thresholds).

[Viola & Jones, CVPR 2001]

# Efficiency issues

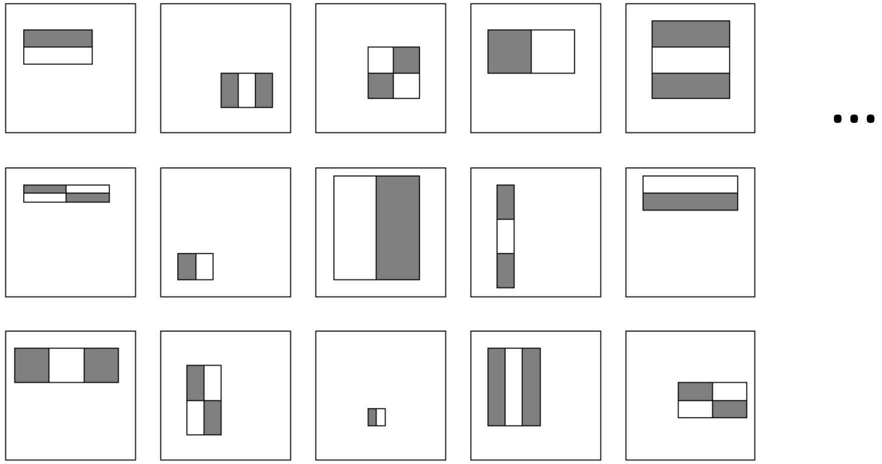


Extract features at each bounding box and apply Adaboost classifier.

- Filter responses can be evaluated fast.
- But each image contains a lot of windows, that we need to classify – potentially great amount of computation!
- How to make detection efficient?

# Cascade of classifiers

- Efficient: Apply first few classifiers (fast), to reject the windows that obviously do not contain the particular category! Then re-classify the regions that survived with stronger classifiers.

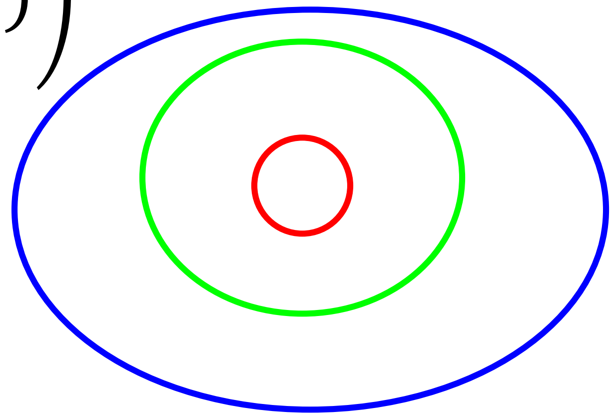


$$h(x) = \operatorname{sgn}\left(\sum_{t=1}^{T} \alpha_t h_t(x)\right)$$

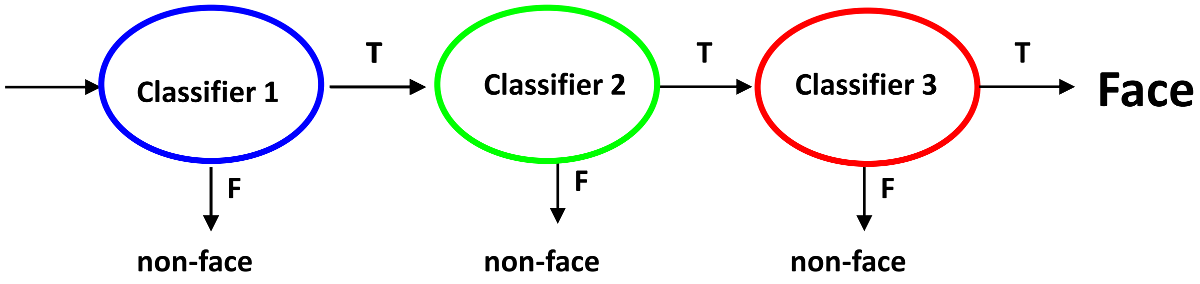
# Cascade of classifiers

- Chain classifiers from least complex with low true-positive rejection rate to most complex ones:

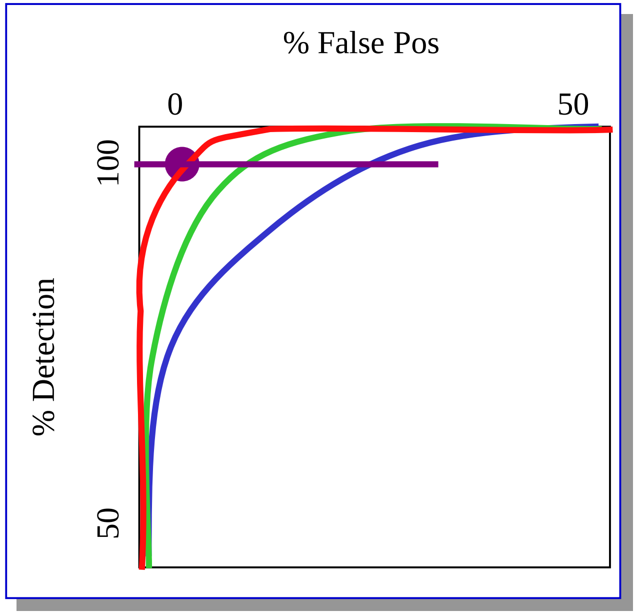
$$h(x) = \operatorname{sgn}\left(\sum_{t=1}^{T} \alpha_t h_t(x)\right)$$



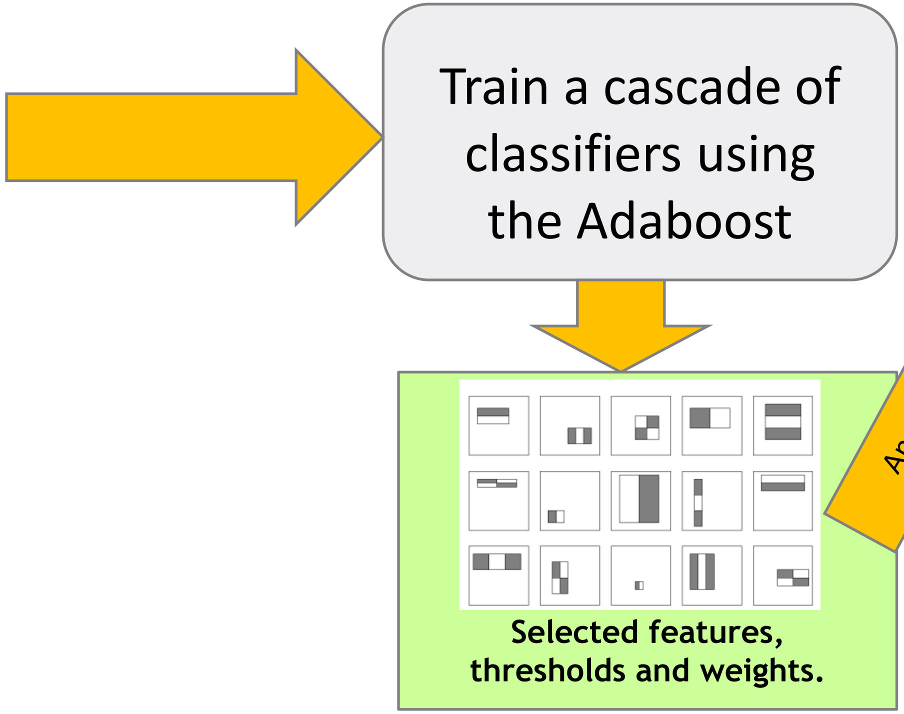
Window cropped from the image

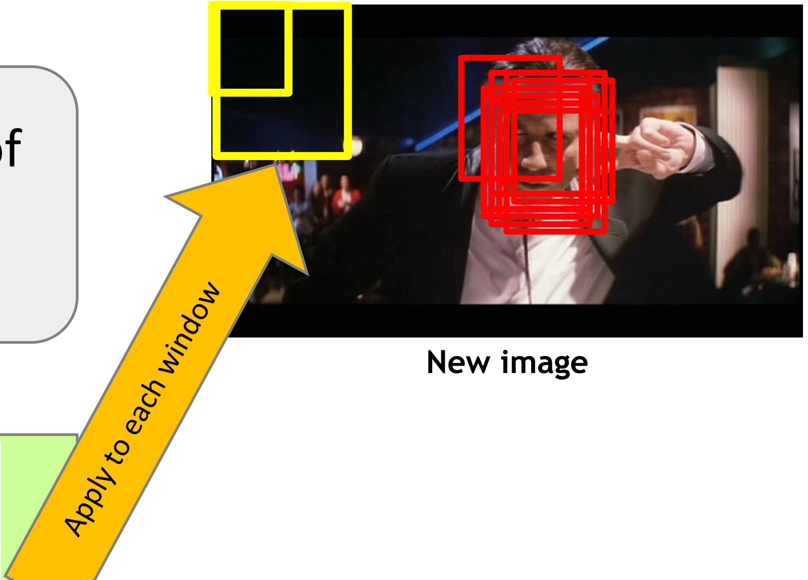



The ROC curve



# Viola-Jones face detector


Postprocess detections by non-maxima suppression.

- Train using 5k positives and 350M negatives
- Real-time detector using 38 layers in cascade
- 6061 features in the final layer (classifier)
- [OpenCV implementation:http://sourceforge.net/projects/opencvlibrary/]

384x288 images, detection 15 fps on 700 MHz Intel Pentium III desktop (2001).
Training time = weeks!

# Detection in progress

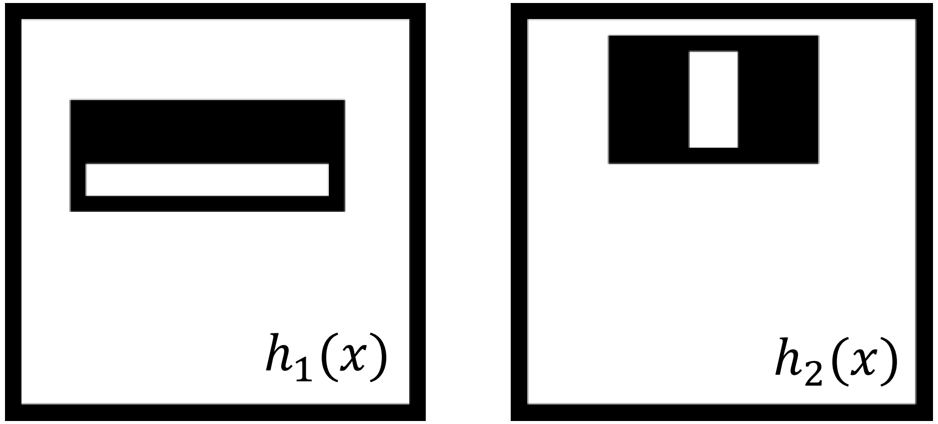
- The video visualizes all the “features”, i.e., filter responses checked in a cascade.
- Observe the increase of cascade once close to face.

http://cvdazzle.com/

# Viola-Jones: results

Guess what these correspond to!

Interesting:
First two selected features.




# Make your face invisible

- Know how it works? Brake it!

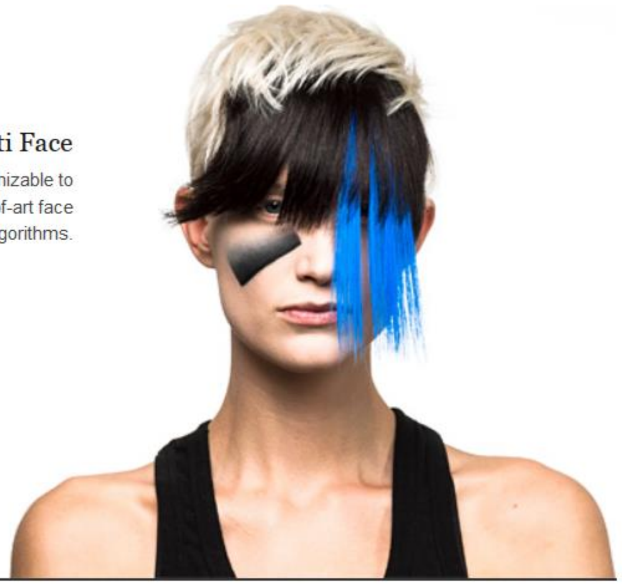
Face

Once computer vision programs detect a face, they can extract data about your emotions, age, and identity.

See how a face is detected

Anti Face

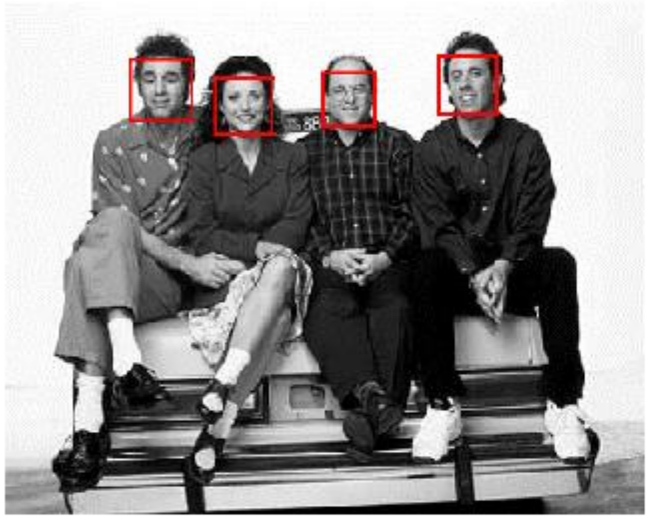
This face is unrecognizable to several state-of-art face detection algorithms.

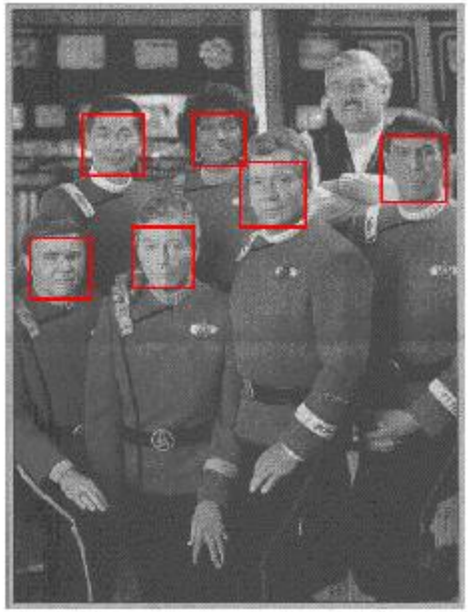


Camouflage from face detection.

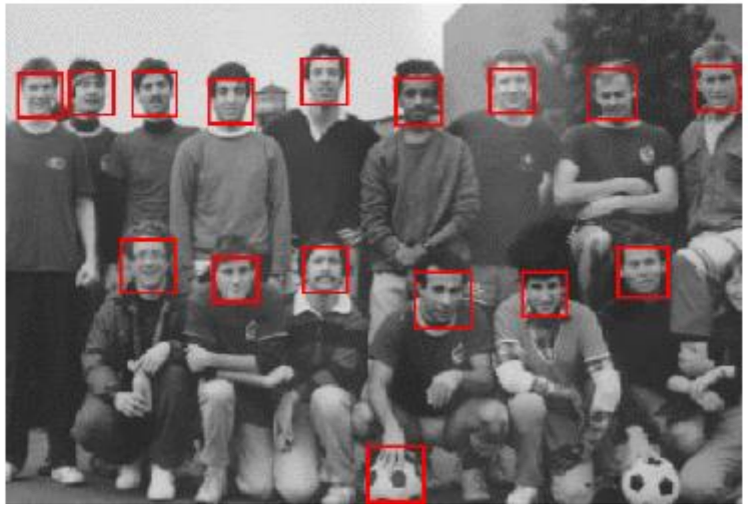
http://cvdazzle.com/

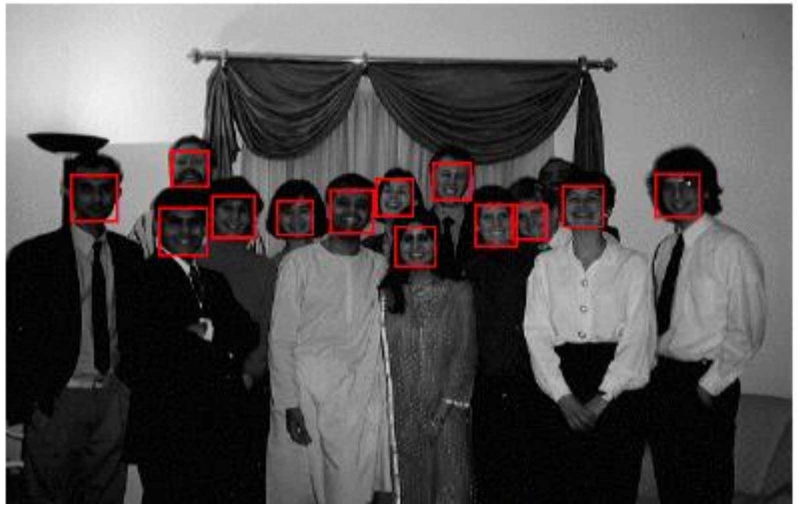
# Viola-Jones: results





Viola, Jones, *“Rapid Object Detection using a Boosted Cascade of Simple Features”*, CVPR2001



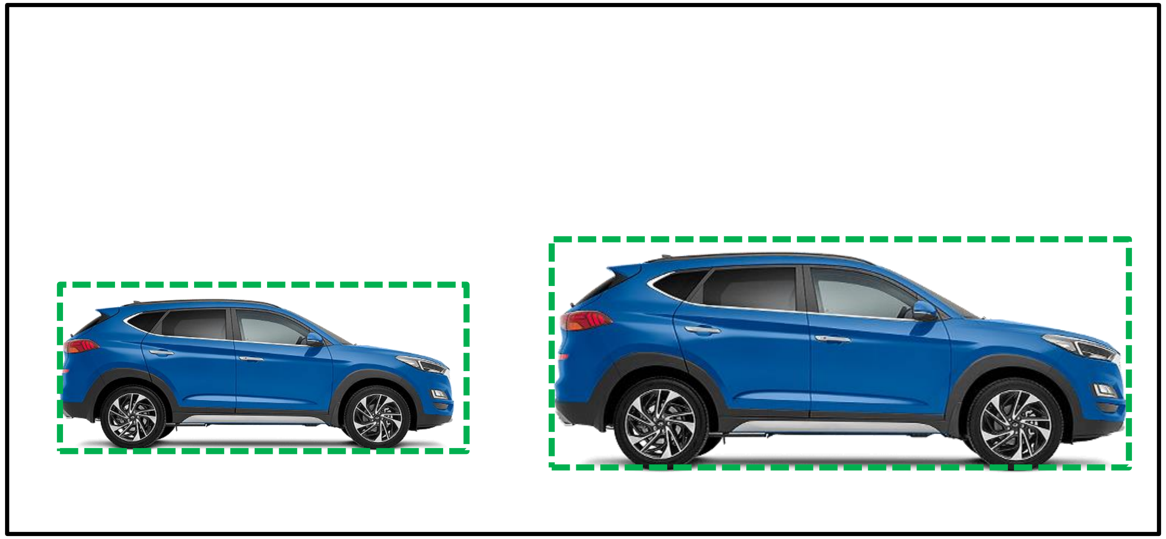


# Viola-Jones (2001

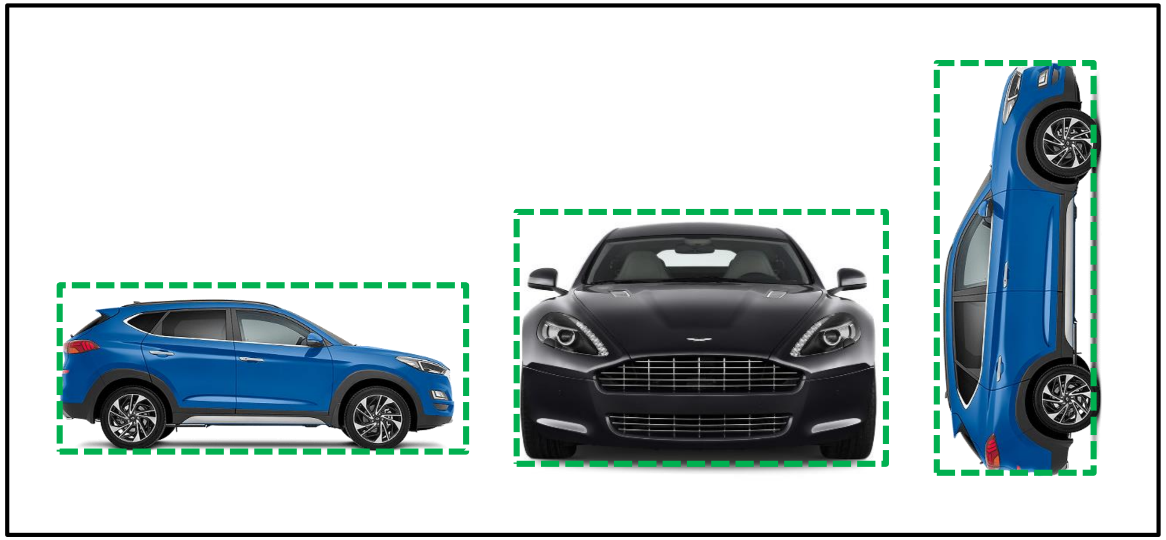

Viola, Jones, *"Rapid Object Detection using a Boosted Cascade of Simple Features"*, CVPR2001

# Sliding windows: Summary

- Strengths:
  - Simple to implement
  - Can deal with scale changes (e.g., by pyramid implementation)



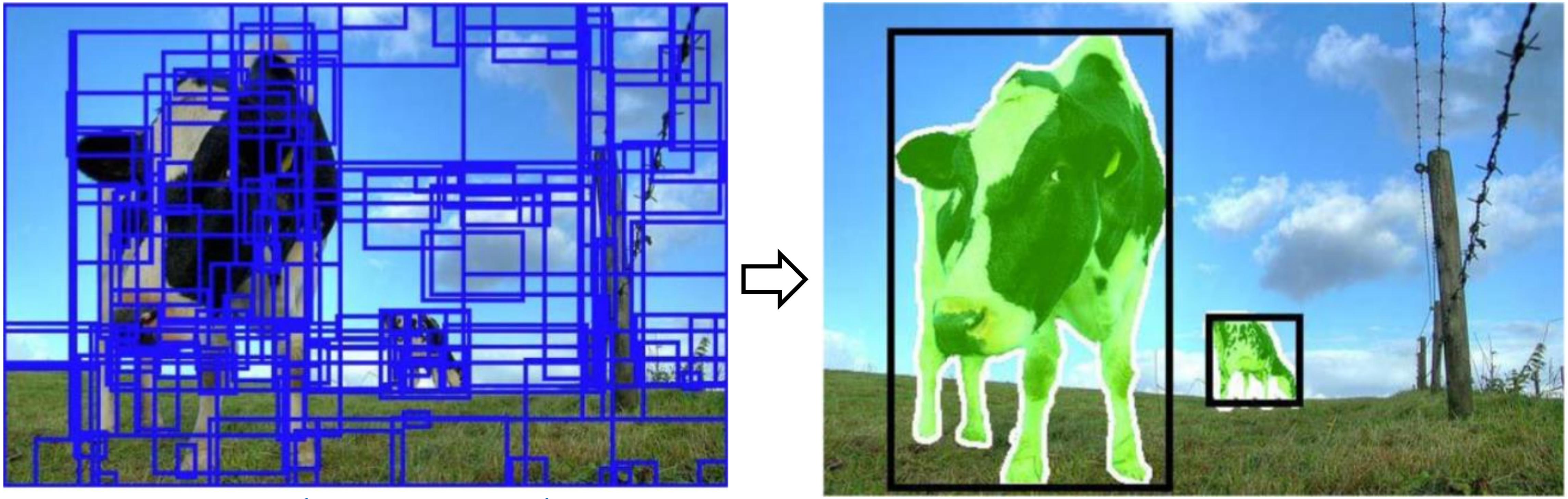
- Weaknesses:
  - Adding aspect change significantly increases computational complexity (bounding boxes do not share equal ratios between width and height)



# Region proposals

- Generate small number (~5000) of hypothesized object bounding boxes
- A potentially slow classifier may be applied to verify these

?



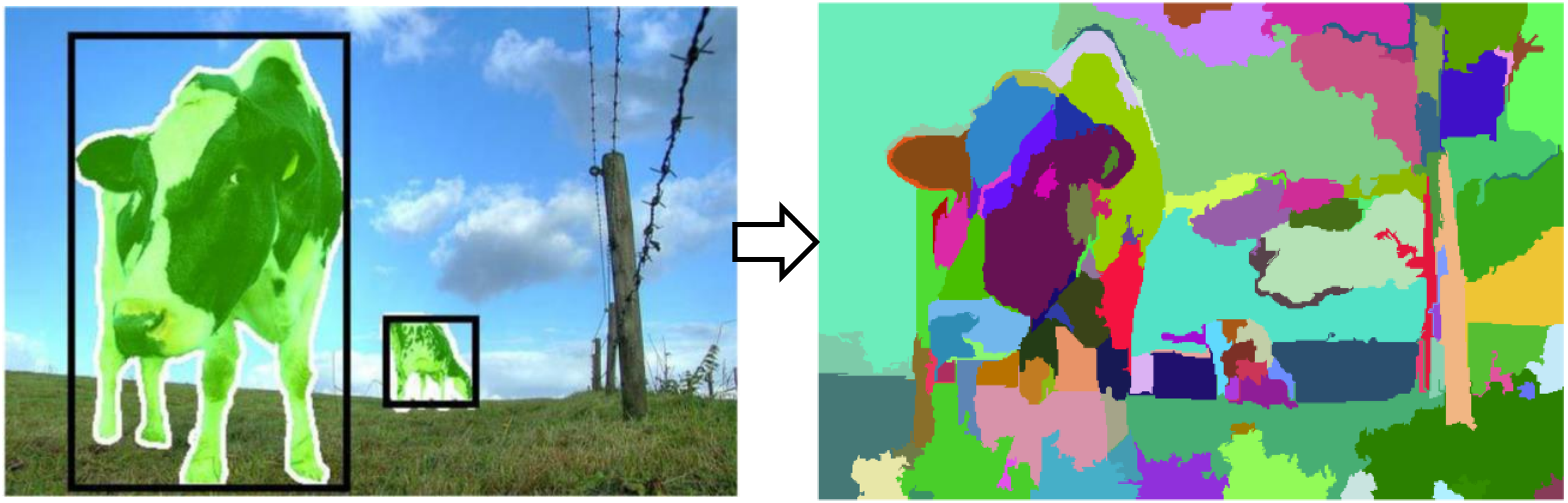
Generated region proposals

After verification with a “cow” classifier

*Sande, et al.*, *Segmentation* as Selective Search for Object Recognition, ICCV 2011

# Selective search

- Insight: Images are intrinsically hierarchical
- Start by over-segmentation into small regions

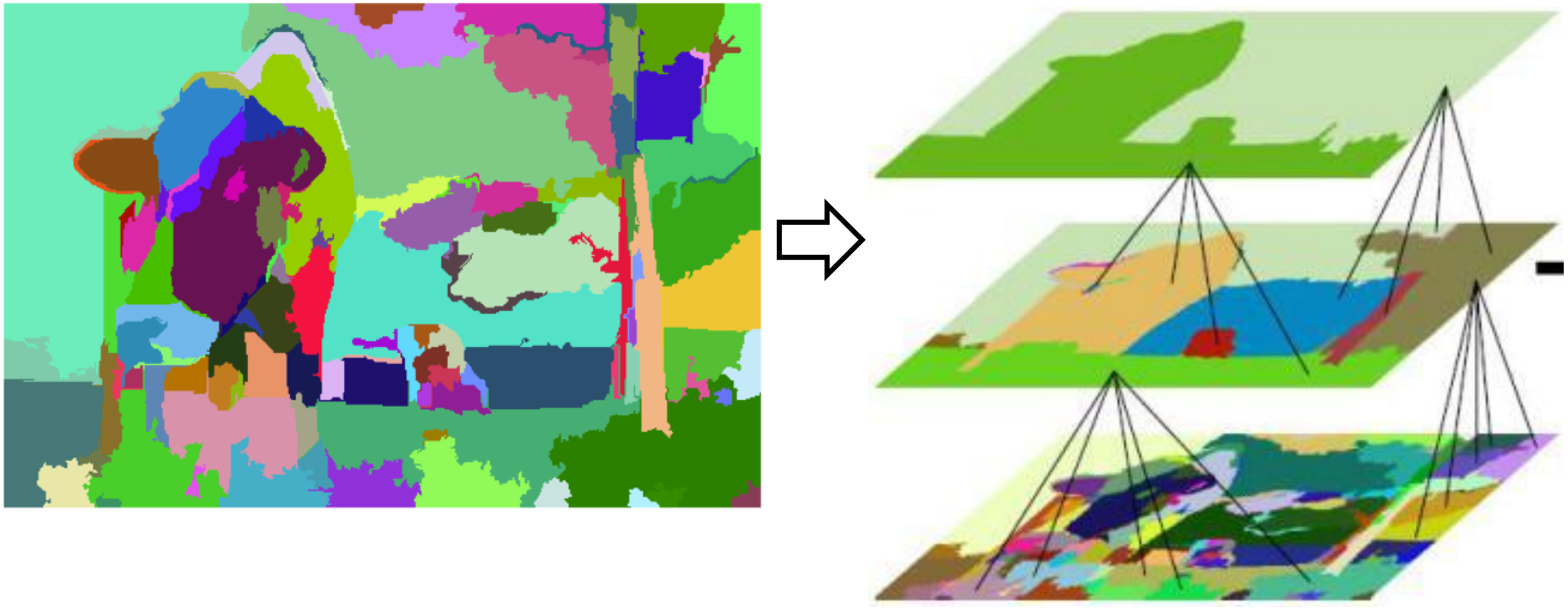


“Efficient graph-based image segmentation” Felzenszwalb and Huttenlocher, IJCV 2004

*Sande, et al.*, [Segmentation as Selective Search for Object Recognition](), ICCV 2011

# Selective search

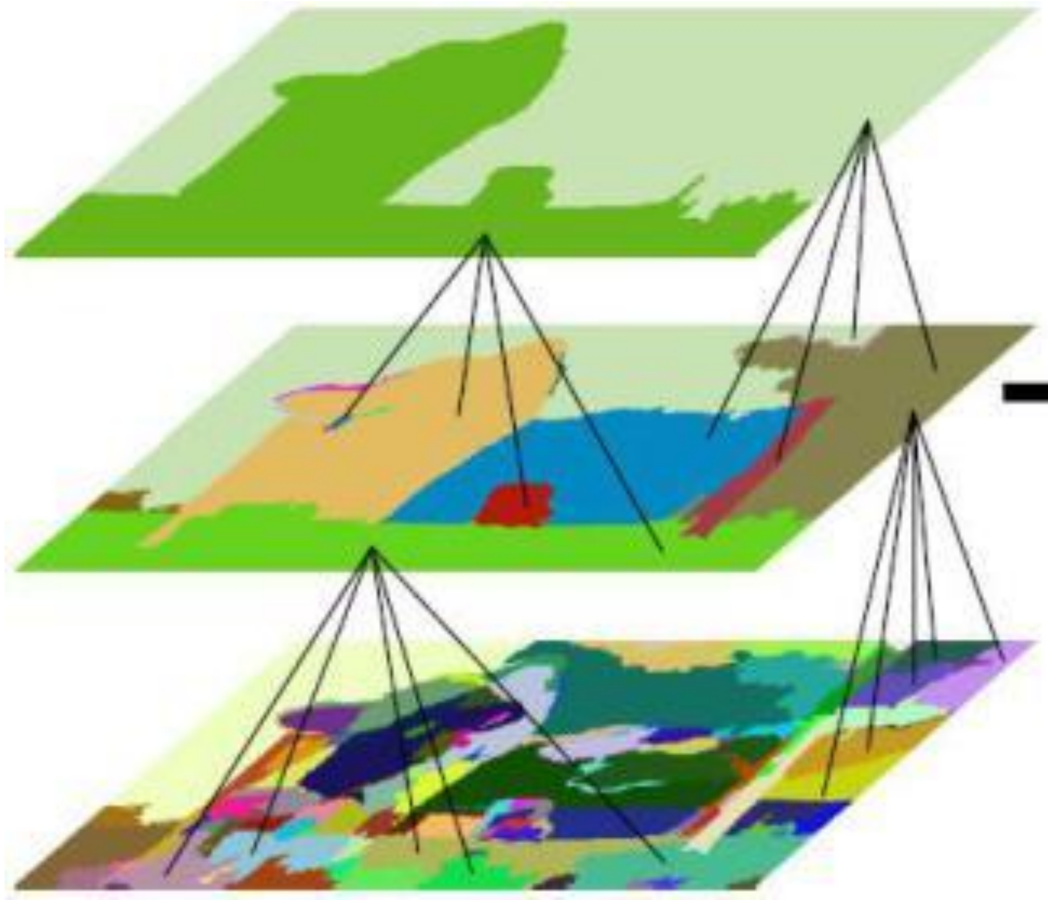
- Merge two most similar regions based on texture similarity and region size
- Continue until a single region remains.

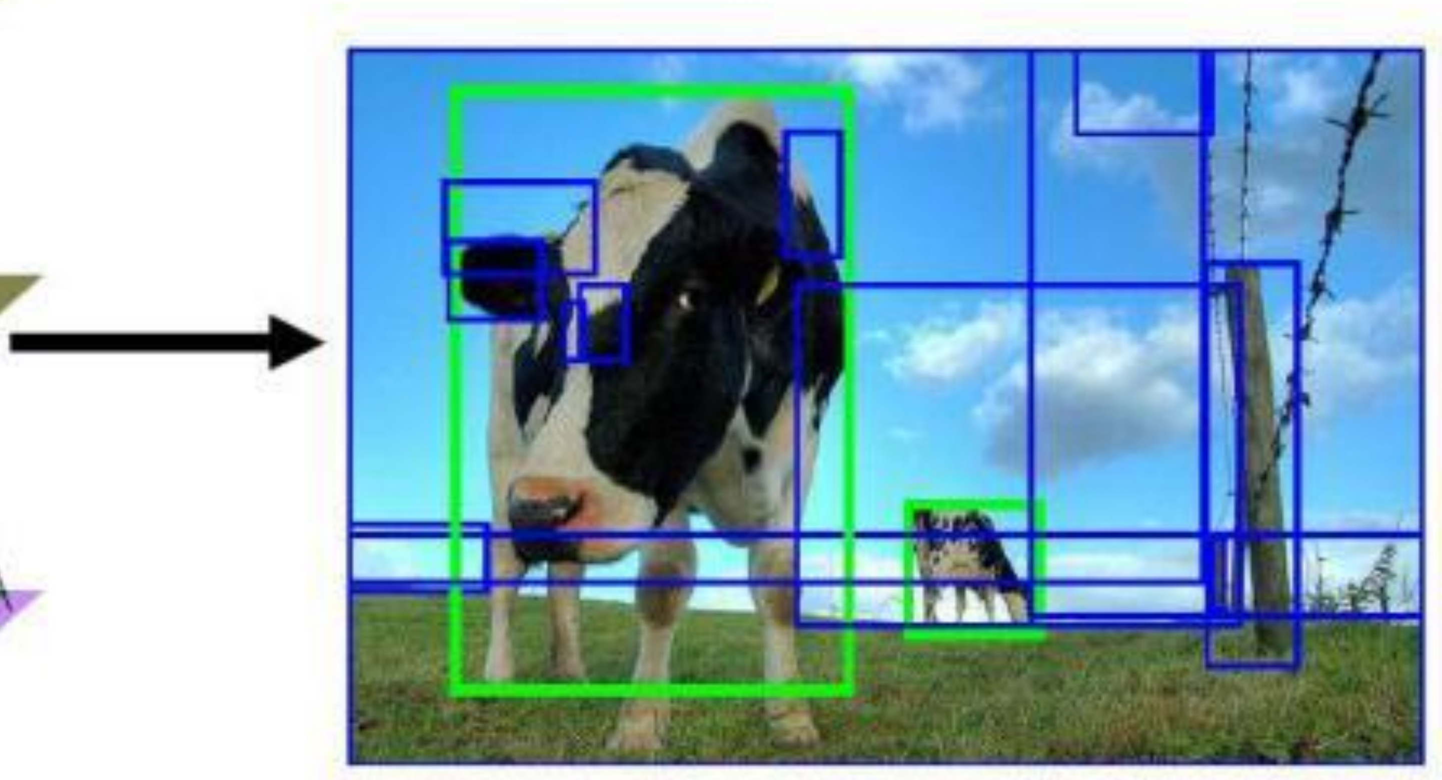


*Sande, et al.*, [*Segmentation* as Selective Search for Object Recognition](), ICCV 2011

# Selective search

- From each merged region generate a bounding box

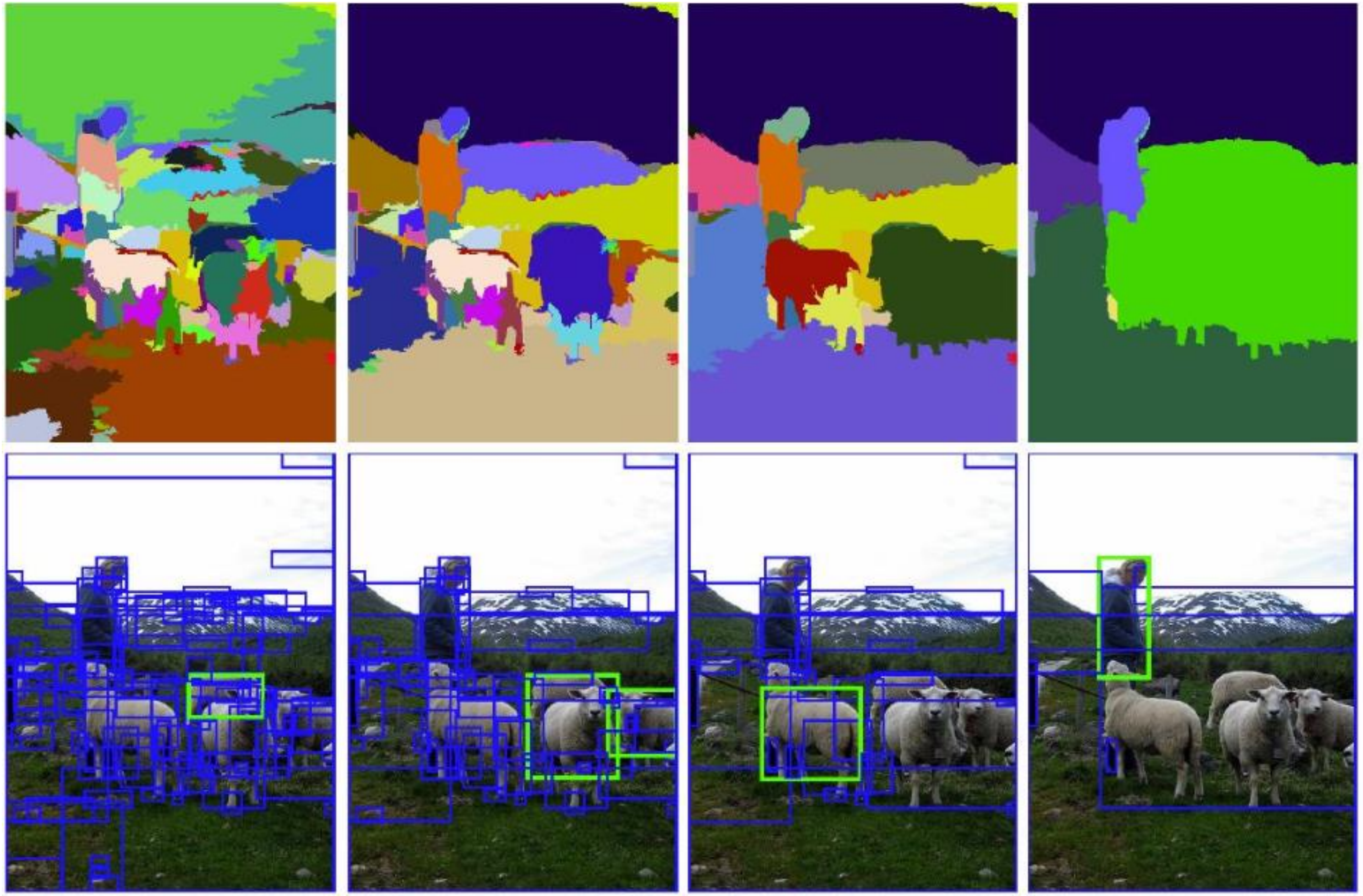




*Sande, et al.*, *Segmentation* as Selective Search for Object Recognition, ICCV 2011

# Selective search

- High recall
- Object-category agnostic!



*Sande, et al.,* [Segmentation as Selective Search for Object Recognition](), ICCV 2011

# How to come up with features?

1. Natural coordinate systems:

   *For some applications, it is enough just to linearly transform the input data.* PCA/LDA

2. Handcrafted nonlinear transforms:

   *Nonlinear transforms improve feature robustness.* HOG

3. Feature selection:

   *Machine learning to select optimal features from a pool of several handcrafted transforms.* Adaboost

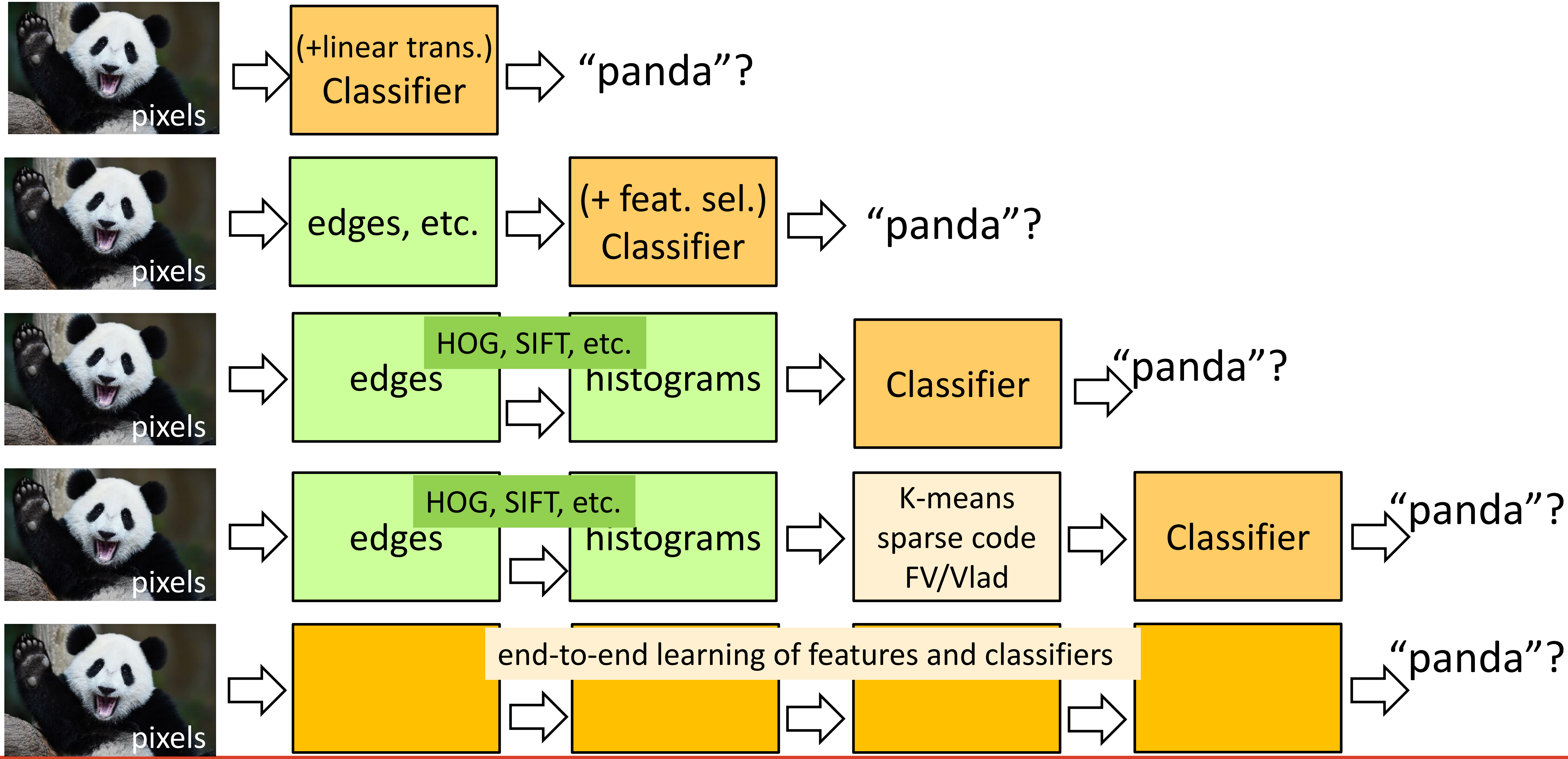
4. End-to-end learning of feature transform:

   *Have machine learn entire feature extraction and selection pipeline.*

Machine Perception

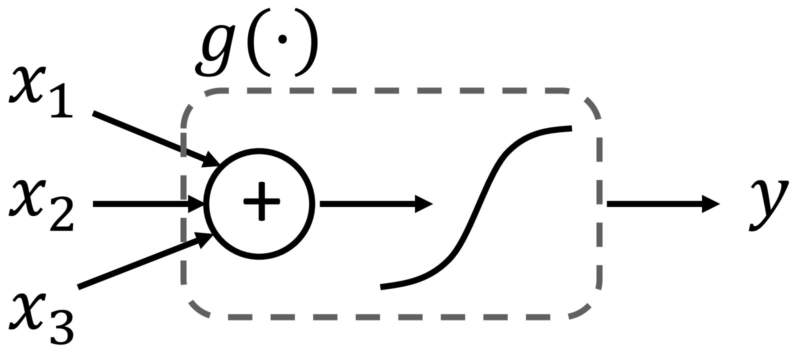
# END-TO-END FEATURE (AND CLASSIFIER ) LEARNING

# Modern representation learning



# Recall a simple neural network

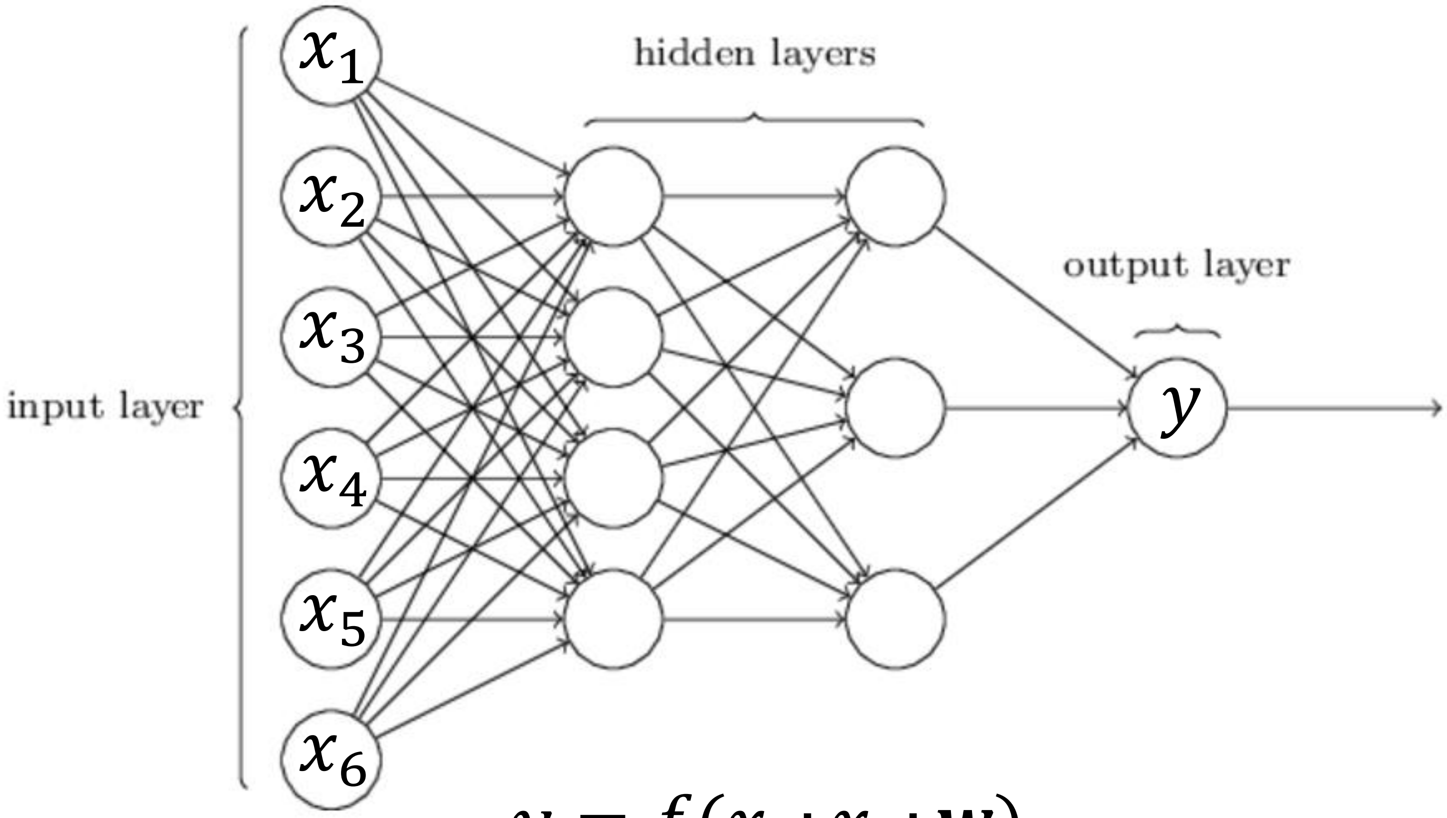
A sigmoid neuron:




A weighted sum of values $x_i$, transformed by a nonlinear function $g(\cdot)$:

$$y = g\left(\sum_i w_i x_i\right)$$

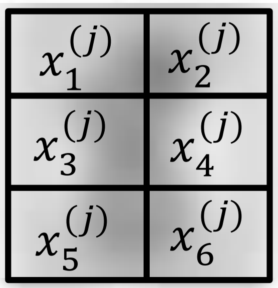
A network of neurons:



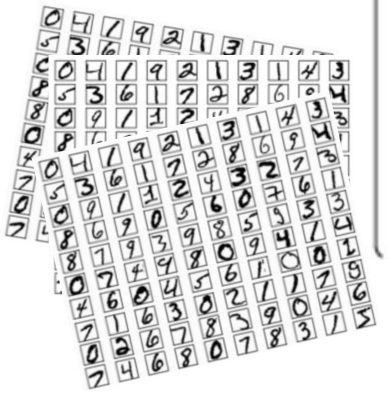

$$y = f(x_1 : x_6; \boldsymbol{w})$$

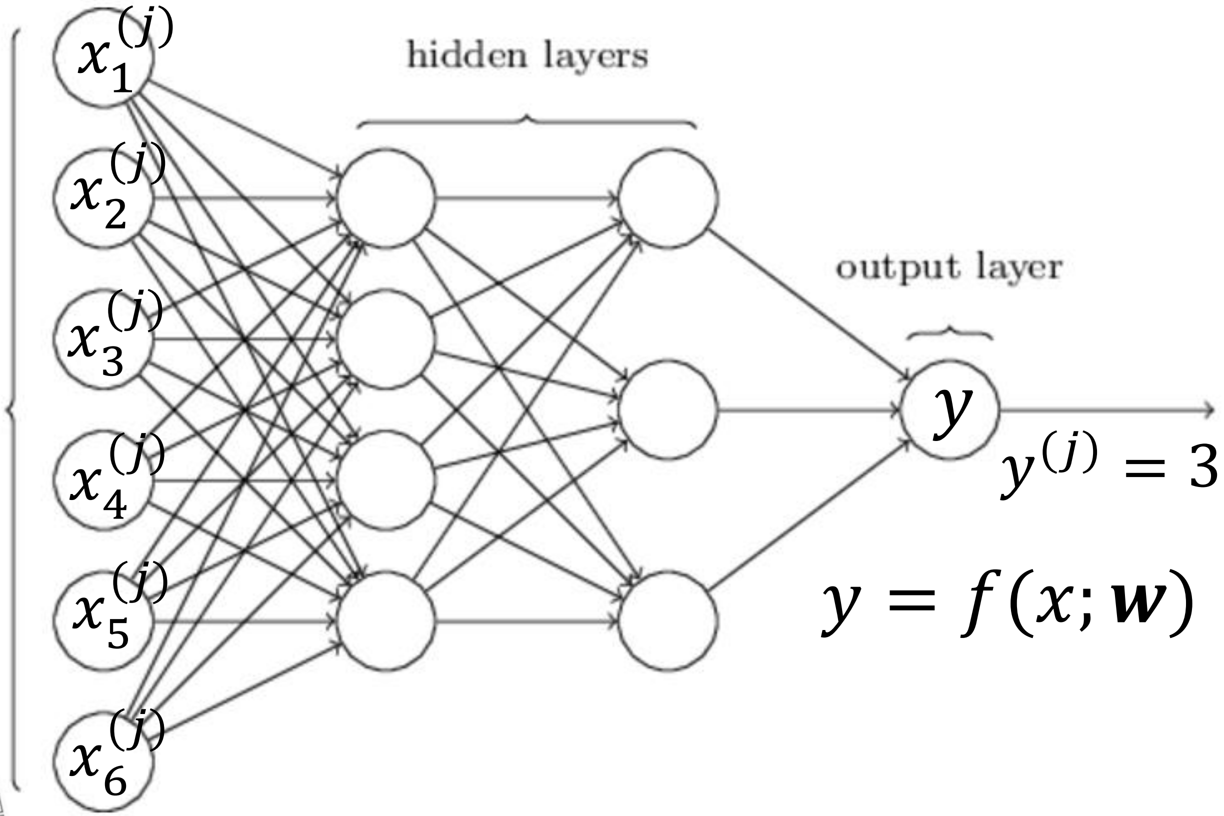
# Learning a simple neural network (in a nutshell)

$t^{(j)} = 8$




$\boldsymbol{x}^{(j)}$






$y^{(j)} = 3$

$y = f(x; \boldsymbol{w})$

Cost function example:

$$\epsilon(j) = \left(t^{(j)} - f\left(\boldsymbol{x}^{(j)}; \boldsymbol{w}\right)\right)^2$$

$$\epsilon(\boldsymbol{w}) = \sum \epsilon(j)$$

Find optimal parameters:

$$\boldsymbol{w}_{\mathrm{opt}} = \underset{\boldsymbol{w}}{\operatorname{argmin}}\, \epsilon(\boldsymbol{w})$$

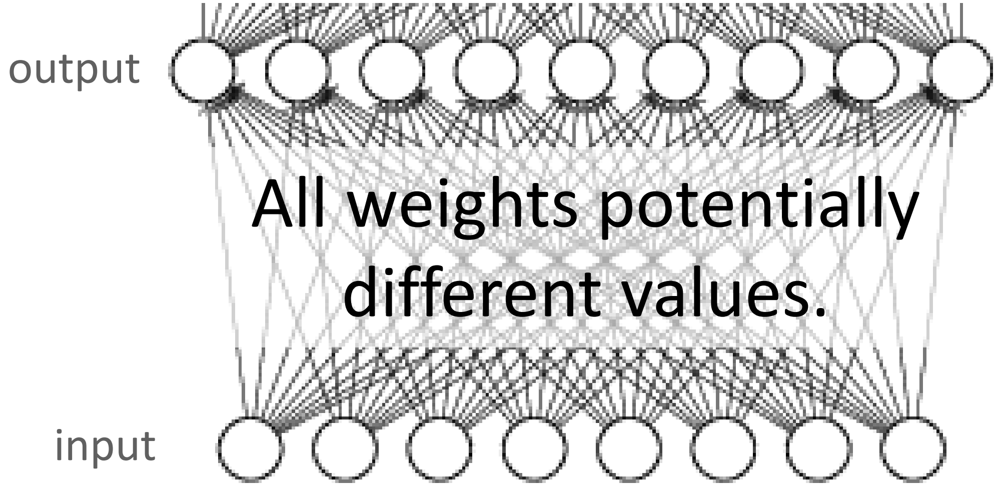
Iteratively adjust the weights to reduce the cost: gradient descent

$$\boldsymbol{w} \leftarrow \boldsymbol{w} + \alpha \frac{\partial \epsilon(\boldsymbol{w})}{\partial \boldsymbol{w}}$$

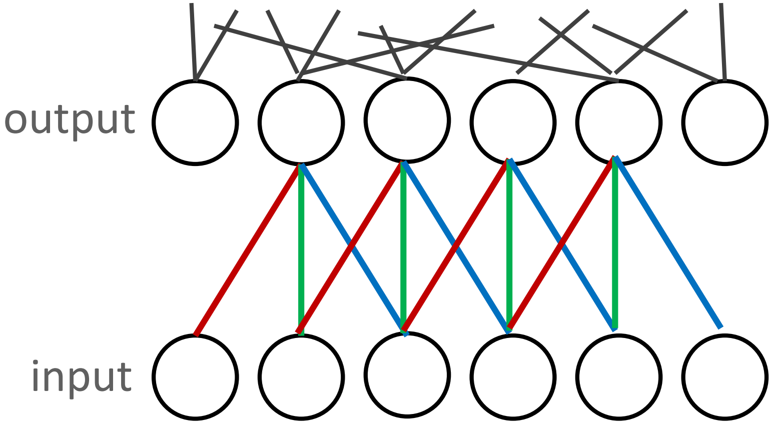
Efficient implementation: Backpropagation algorithm

# Put some structure in neural networks: CNN

- A fully connected neural network

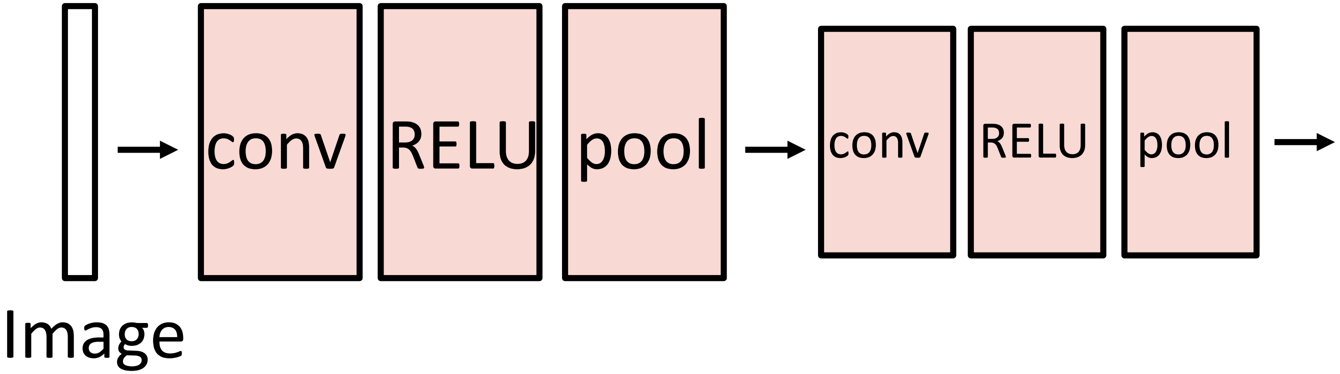



- A convolutional neural network




The basic building blocks of a CNN:

- Convolutional layers
- Nonlinearity (RELU)
- Pooling layers



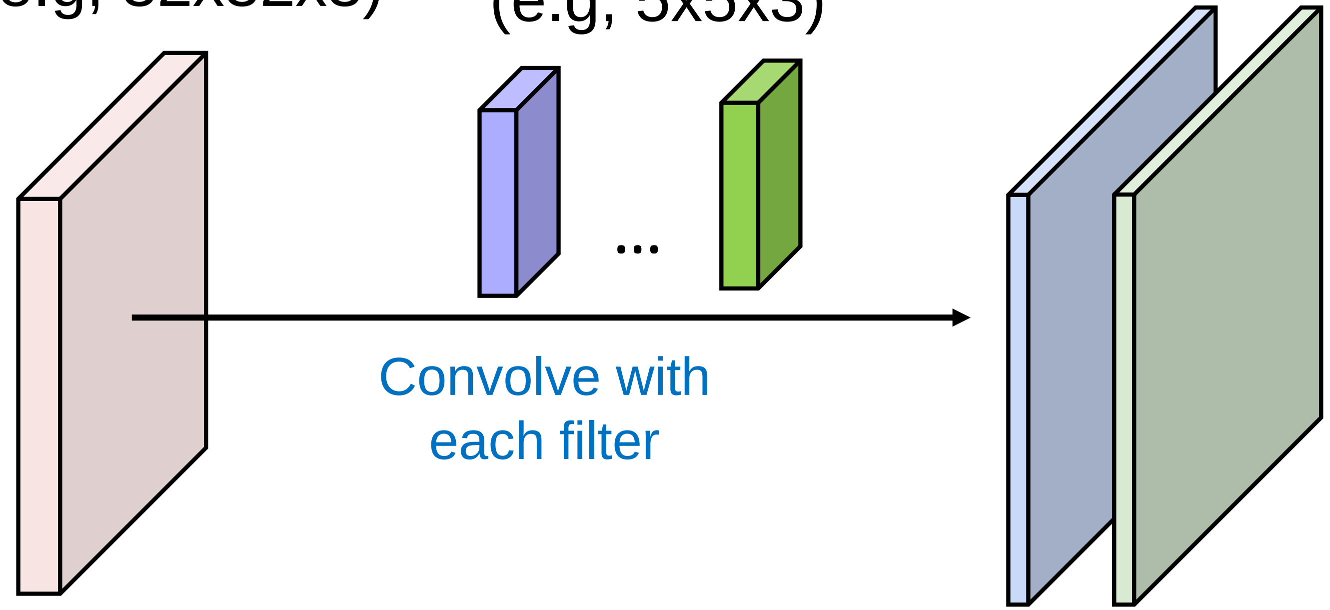

*New building blocks emerge each year…*

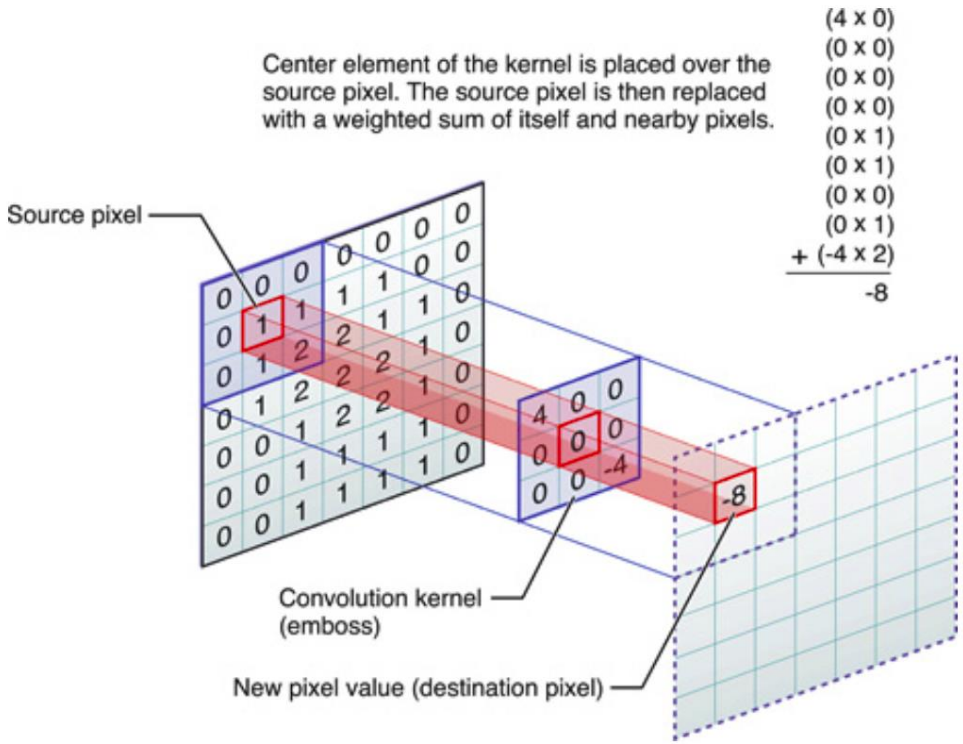
# Convolutional layer

Each filter generates a 2D output

Image
(e.g, 32x32x3)

N filters
(e.g, 5x5x3)

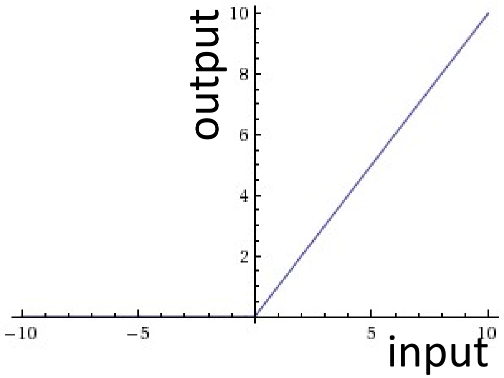


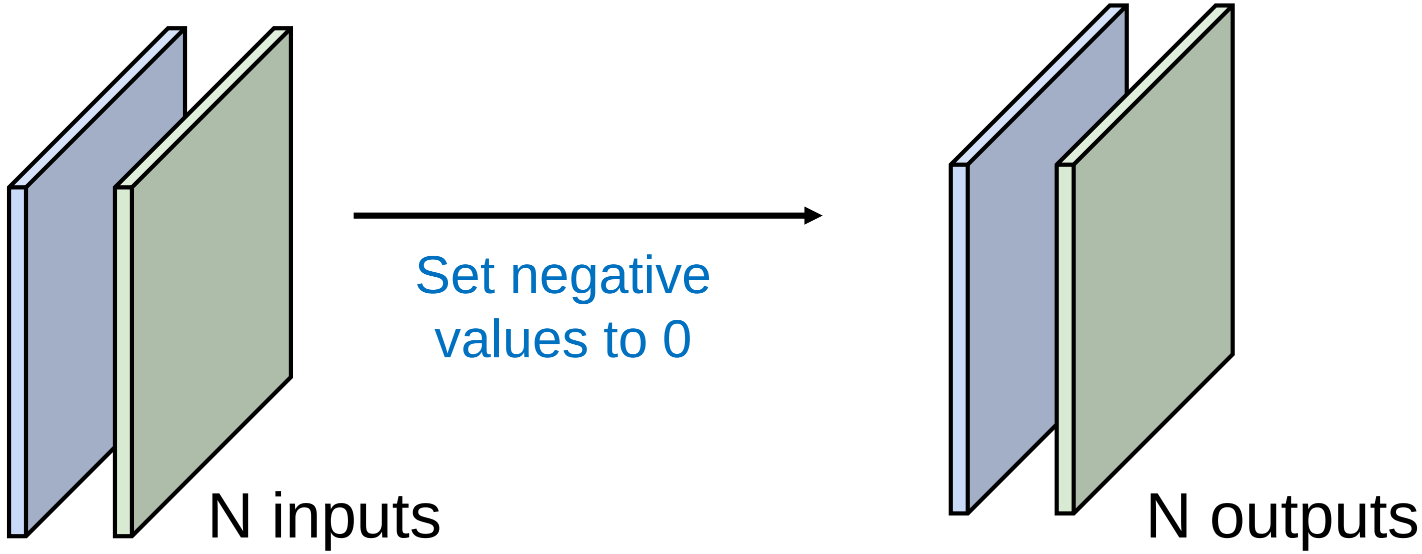




Output is a 32x32xN tensor

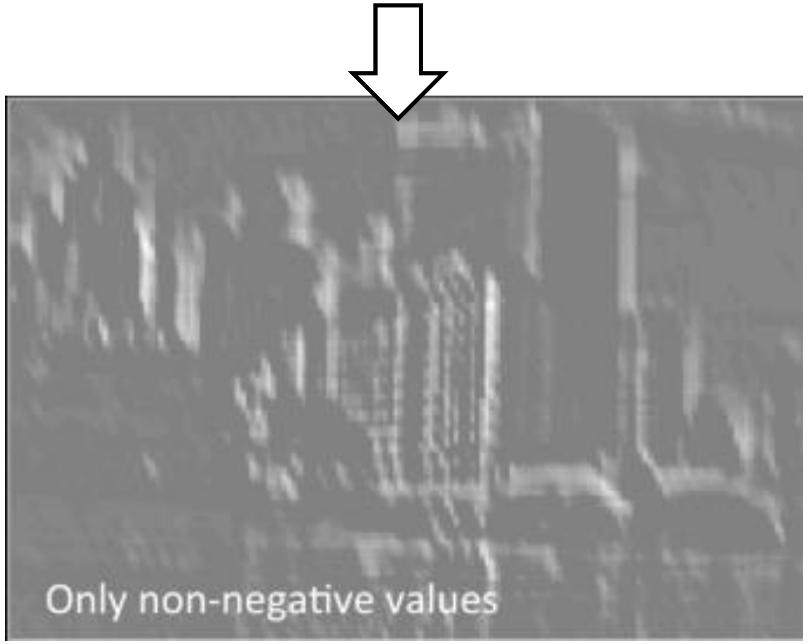
# Nonlinear layer (e.g., RELU)

- Rectified linear unit (RELU)
- Implement nonlinear feature transformations
- Specific form crucial for backpropagation to work!



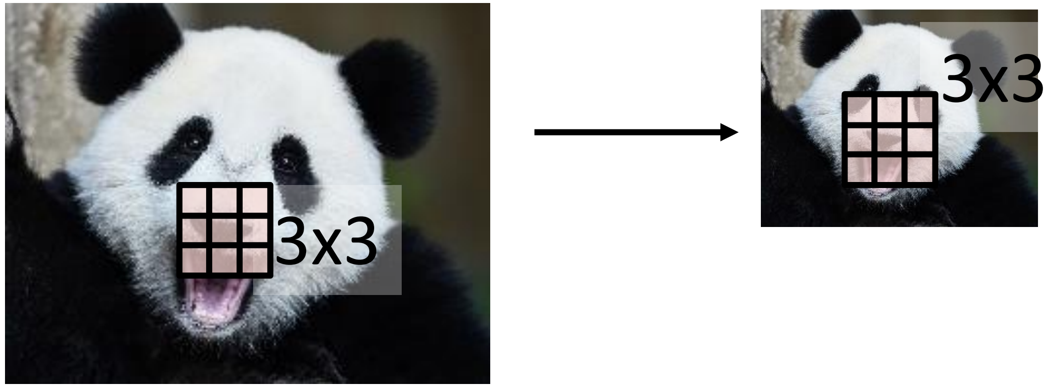



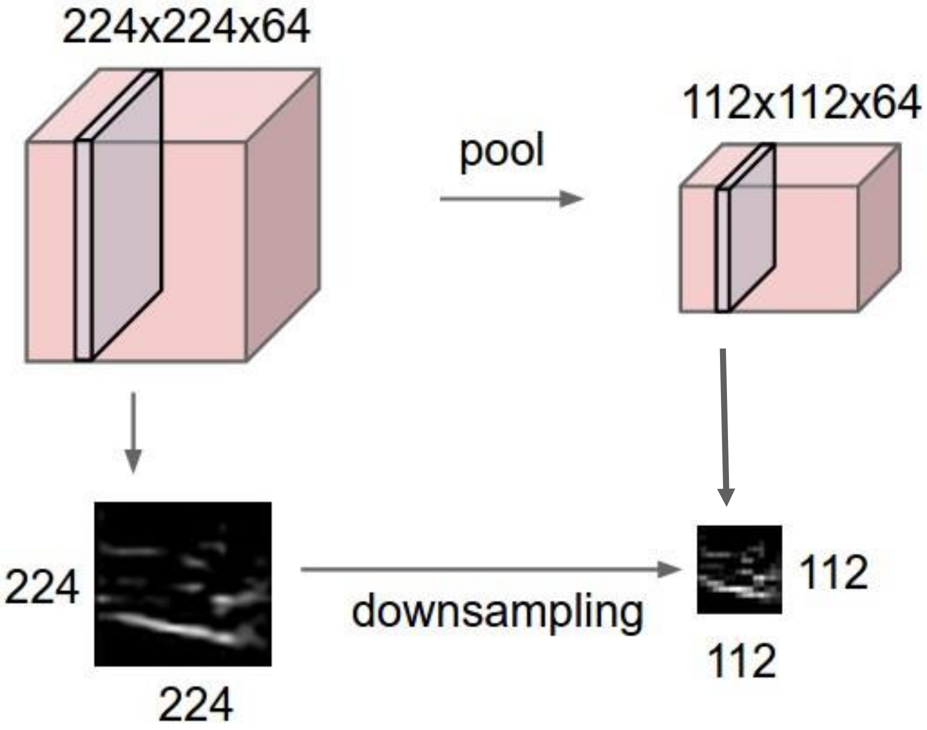




# Pooling layer

- Aim:
  - Increase the receptive field without significantly increasing the number of parameters.
- A popular pooling operation: max pool




Single depth slice

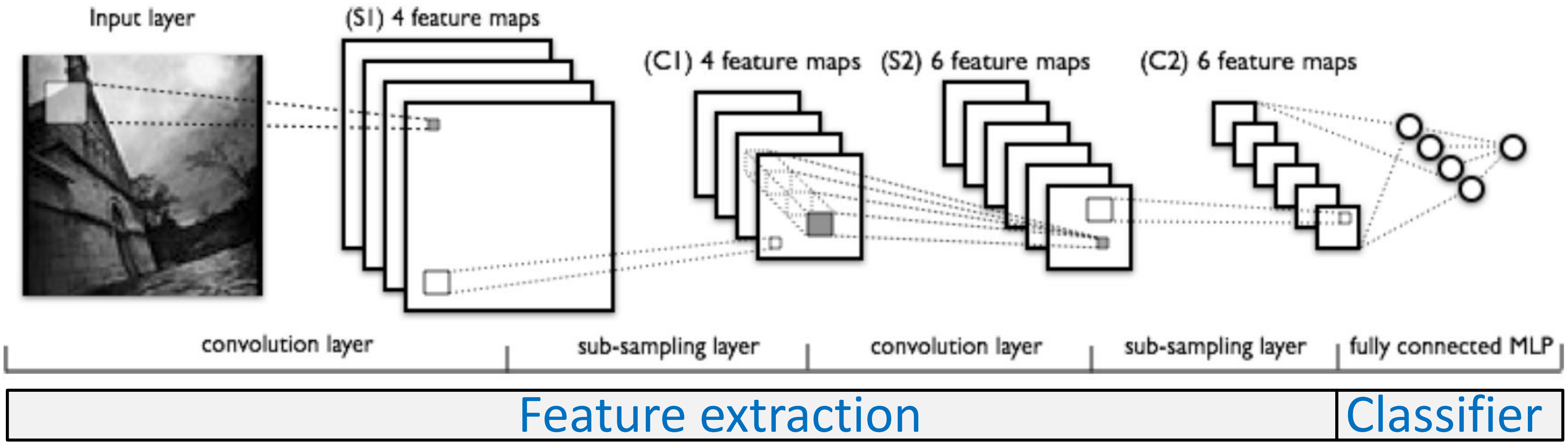
| | | | |
|---|---|---|---|
| 1 | 1 | 2 | 4 |
| 5 | 6 | 7 | 8 |
| 3 | 2 | 1 | 0 |
| 1 | 2 | 3 | 4 |

max pool with 2x2 filters and stride 2 →

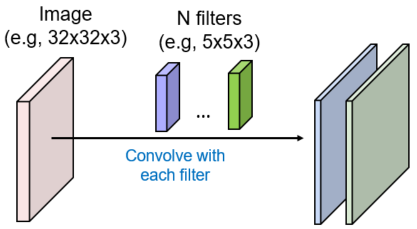
| | |
|---|---|
| 6 | 8 |
| 3 | 4 |

# A conceptual CNN architecture

- Architecture contains feature extraction as well as a classifier
- Learning means:
  - Learn feature extraction (convolution filter kernels)
  - Learn a classifier (e.g., a multi-layer perceptron)



# CNNs attract a significant attention in 2012




- The filters and biases in CNN are the parameters to be learned.
- The breakthrough came with the AlexNet (50-60 million parameters)

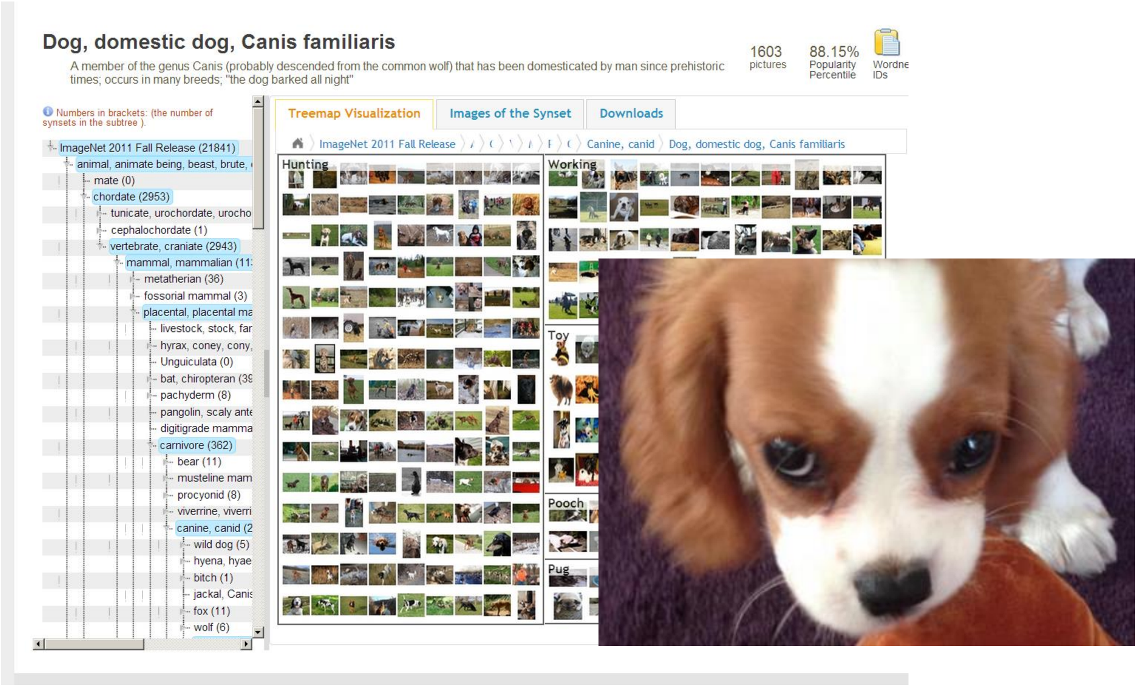
[1]Krizhevsky et al., ImageNet Classification with Deep Convolutional Neural Networks, NIPS2012

- Became possible due to HUGE labelled datasets (ImageNet )

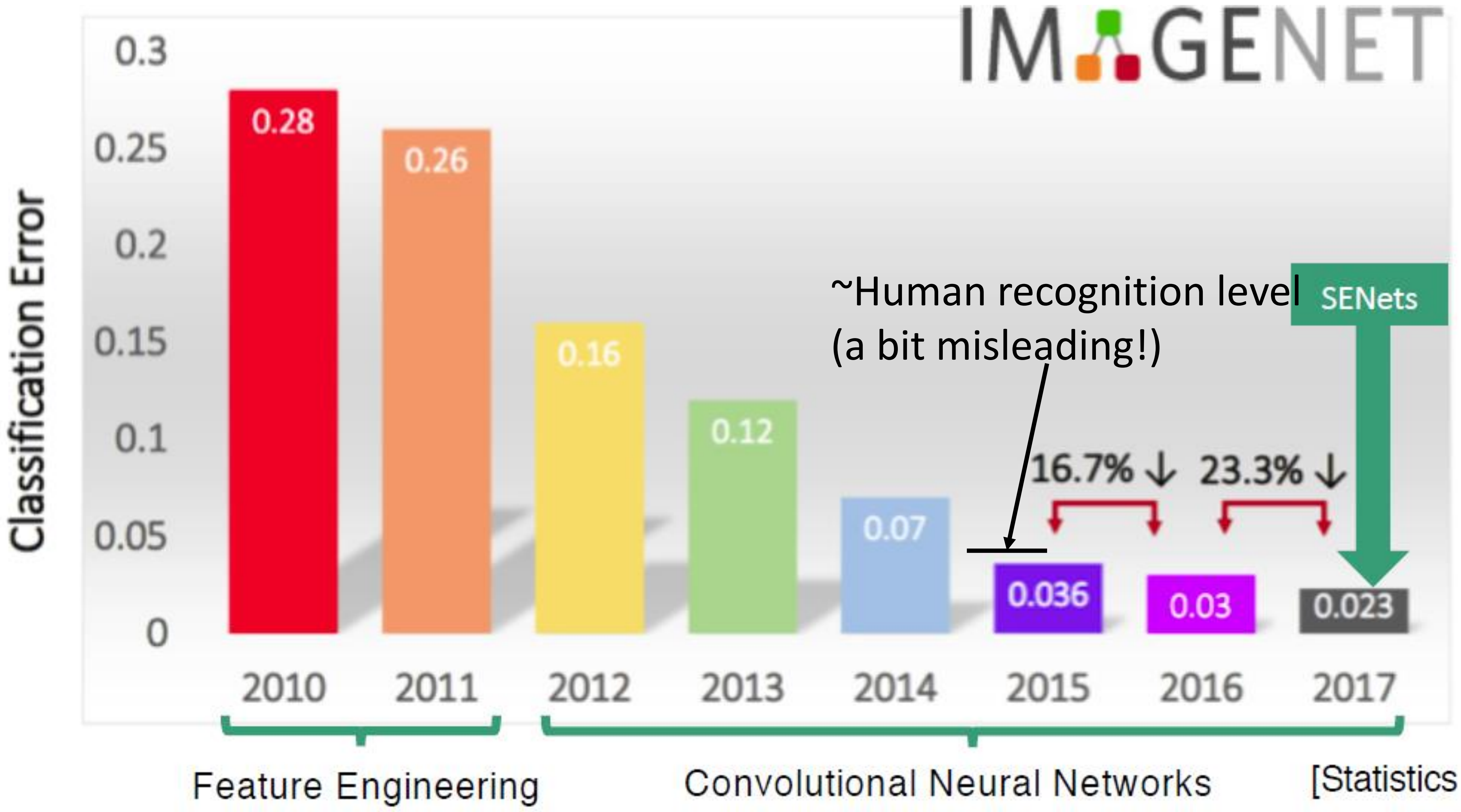
14 million labeled images, 20K categories

http://www.image-net.org/

| Model | Top-1 | Top-5 |
|---|---|---|
| *Sparse coding [2]* | *47.1%* | *28.2%* |
| *SIFT + FVs [24]* | *45.7%* | *25.7%* |
| CNN | **37.5%** | **17.0%** |



# Recognition paradigm shift

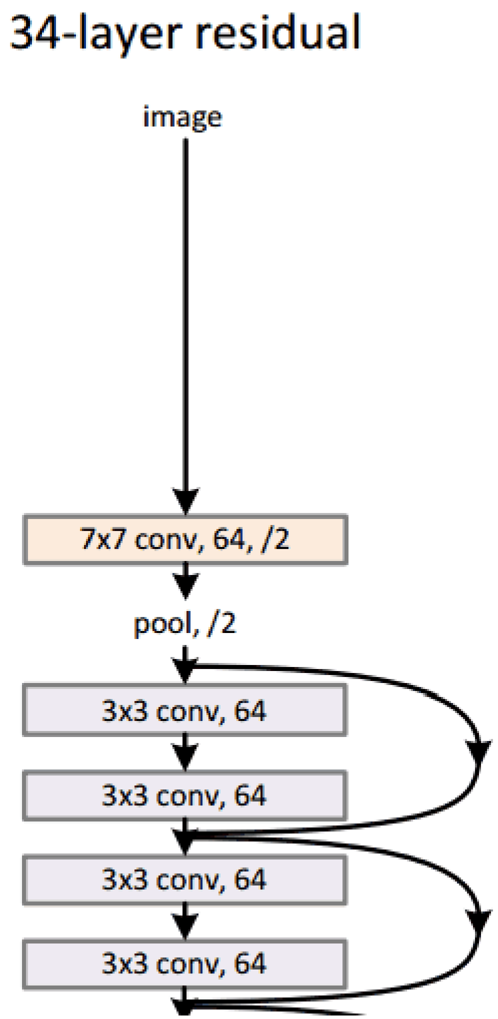



[Statistics provided by ILSVRC]

https://towardsdatascience.com/review-senet-squeeze-and-excitation-network-winner-of-ilsvrc-2017-image-classification-a887b98b2883

# Better approaches + depth

**Microsoft ResNet (2015)**




https://adeshpande3.github.io/adeshpande3.github.io/The-9-Deep-Learning-Papers-You-Need-To-Know-About.html

ResNet is a 152 layer network. It won ILSVRC 2015 with an incredible error rate of 3.6%

See more recent architectures:

Chollet, Xception: Deep Learning with Depthwise Separable Convolutions, CVPR 2017

Sandler et al., MobileNetV2: Inverted Residuals and Linear Bottlenecks, CVPR 2018

Howard et al., Searching for MobileNetV3, ICCV 2019

# Advances made over the years

- Initialization techniques (He et al., ICCV 2015)
- Optimization techniques

  (Adam, dropout, batch normalization, block normalization, etc.)
  [Krizhevsky et al. NIPS2012, Ioffe & Szegedy ICML2015, Wu et al. ECCV2018, …]
- Data augmentation

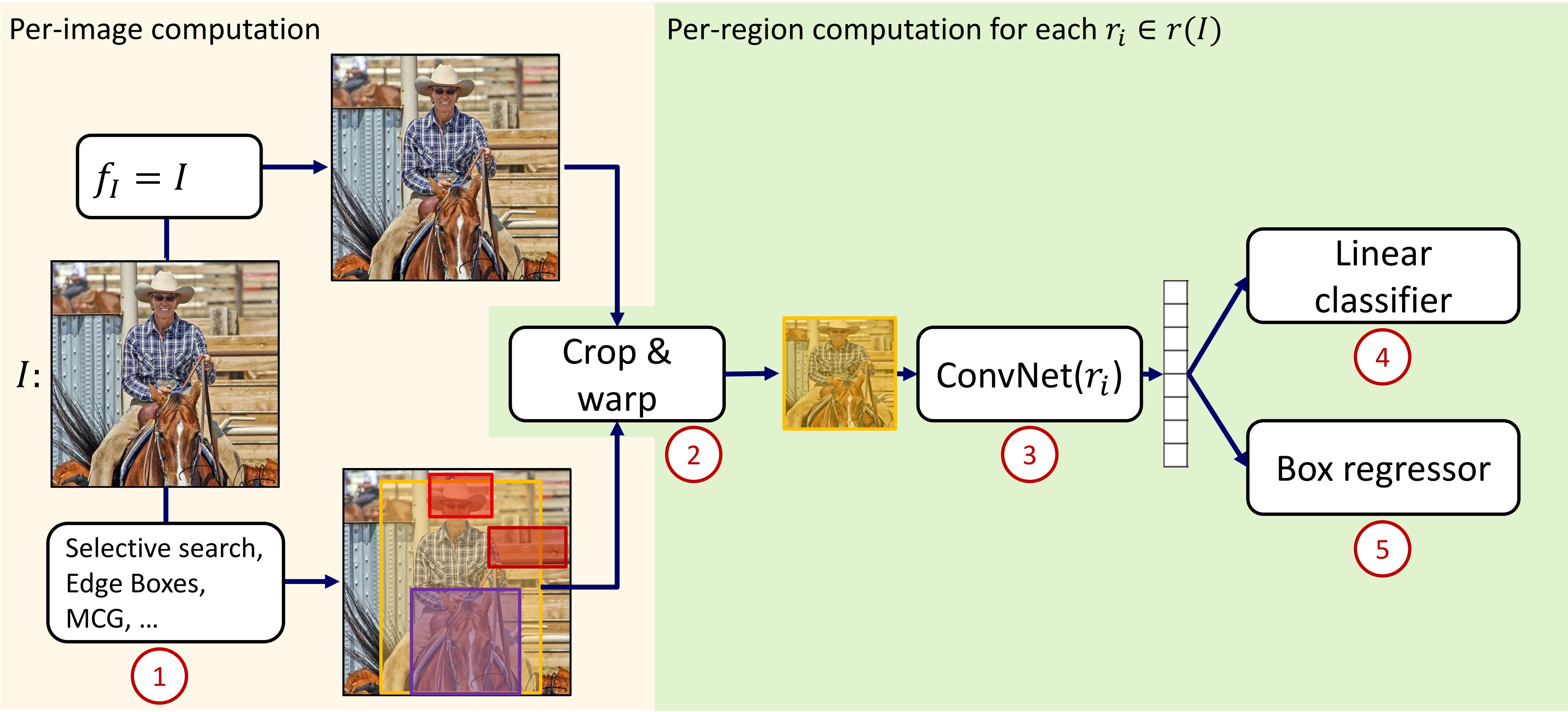
  (flipping, scaling, rotating images, adding noise, vary colors, etc.)
- Architectural changes

  (skip connections, batchnorm, multipath nets, bottlenecks, etc.)

  [Szegedy et al. CVPR2015, He et al. CVPR2016, Xie et al. CVPR2017, …]
- See COCO challenge for state-of-the-art (http://cocodataset.org)
- Applications to recognition, object detection, segmentation, etc.

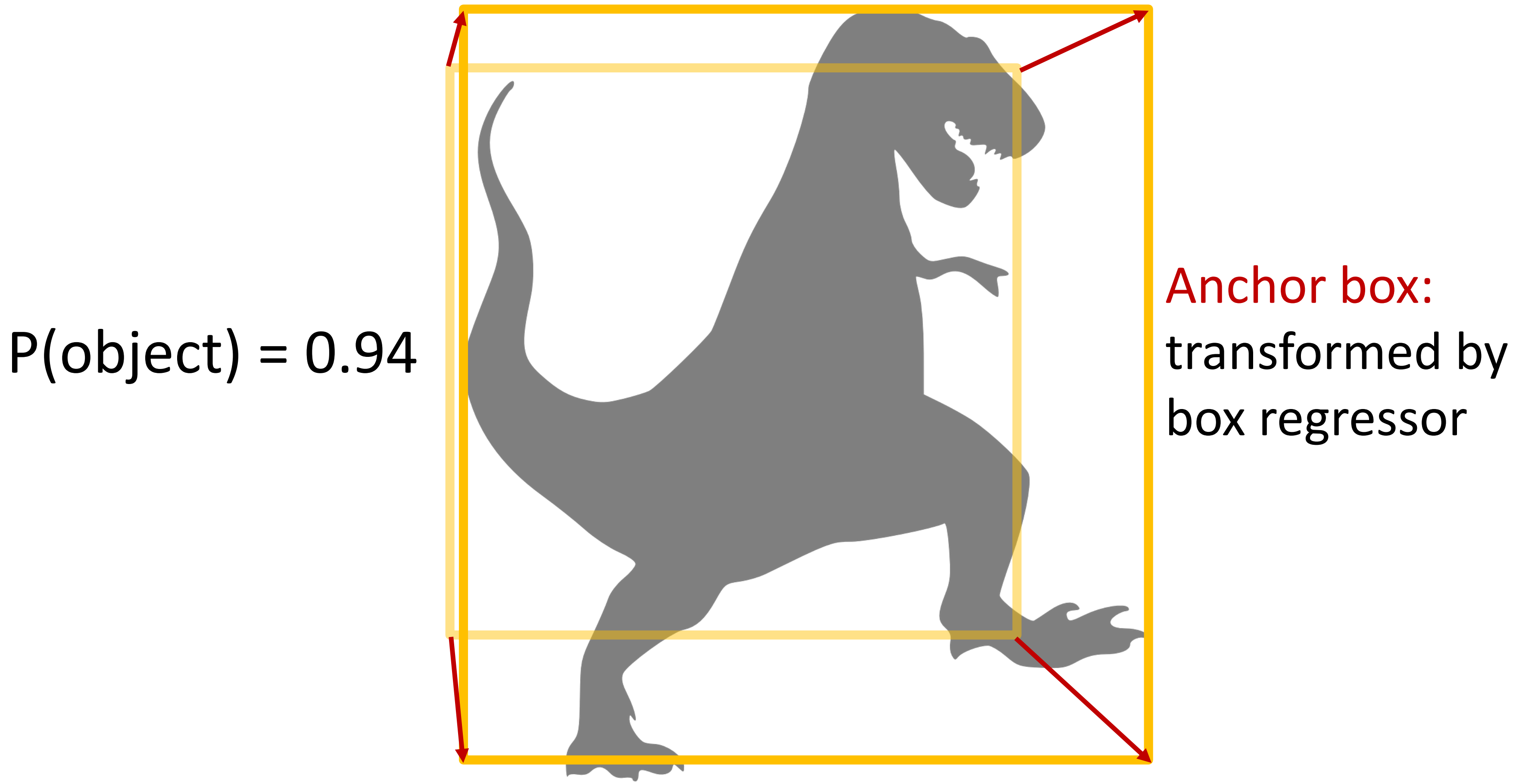
# Object detection by R-CNN

Girshick et. al, Rich Feature Hierarchies for Accurate Object Detection and Semantic Segmentation, CVPR 2014

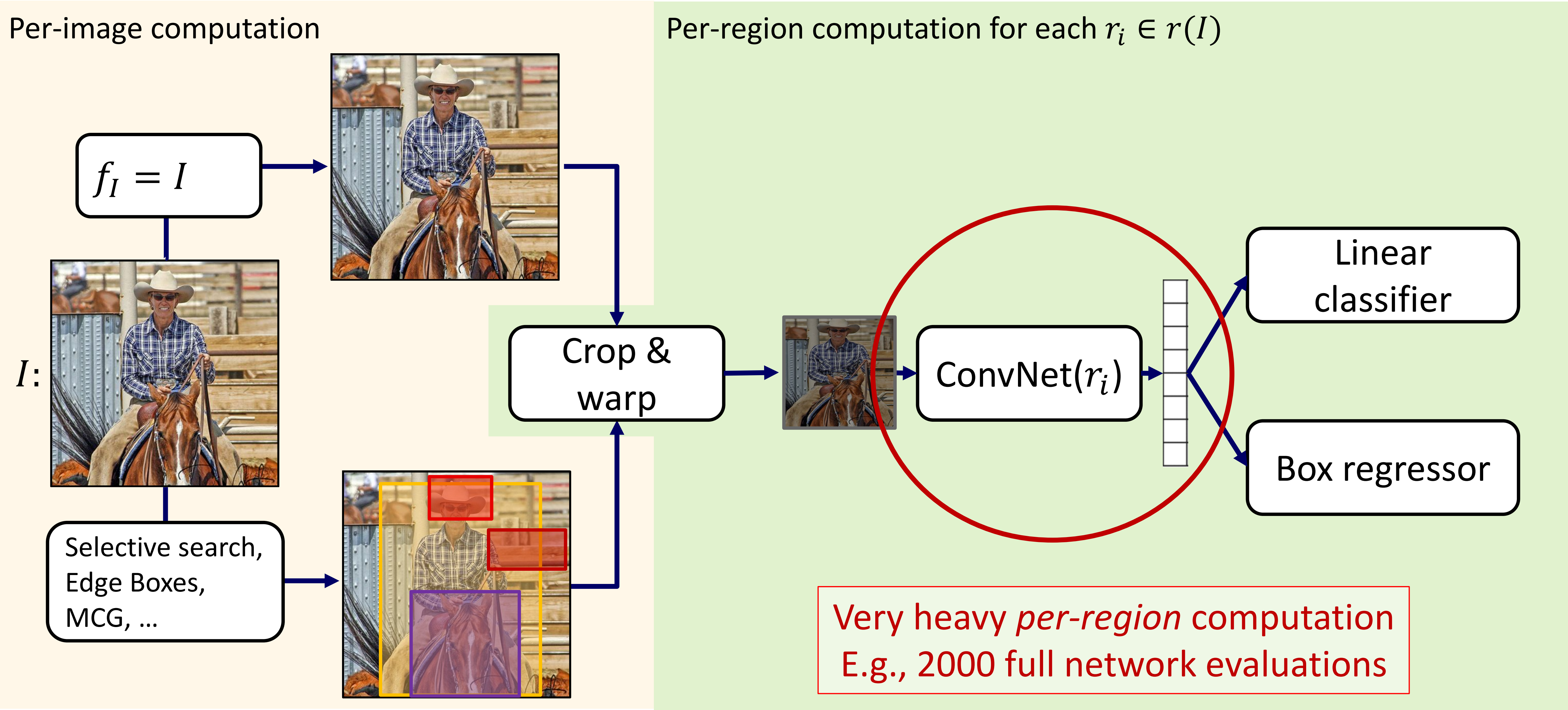


# Box regression

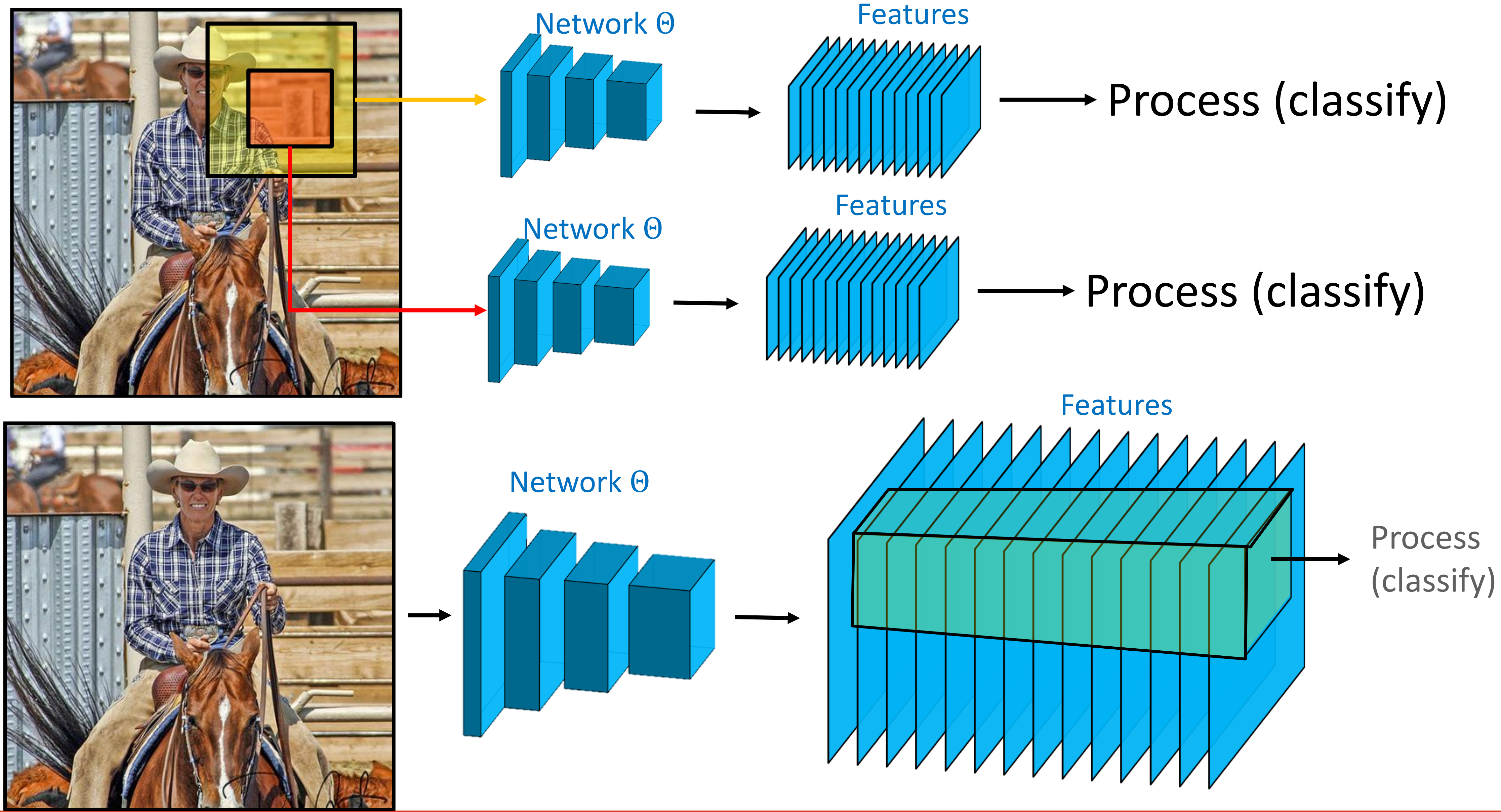
- Region proposal generates approximate bounding box
- The box regressor refines it



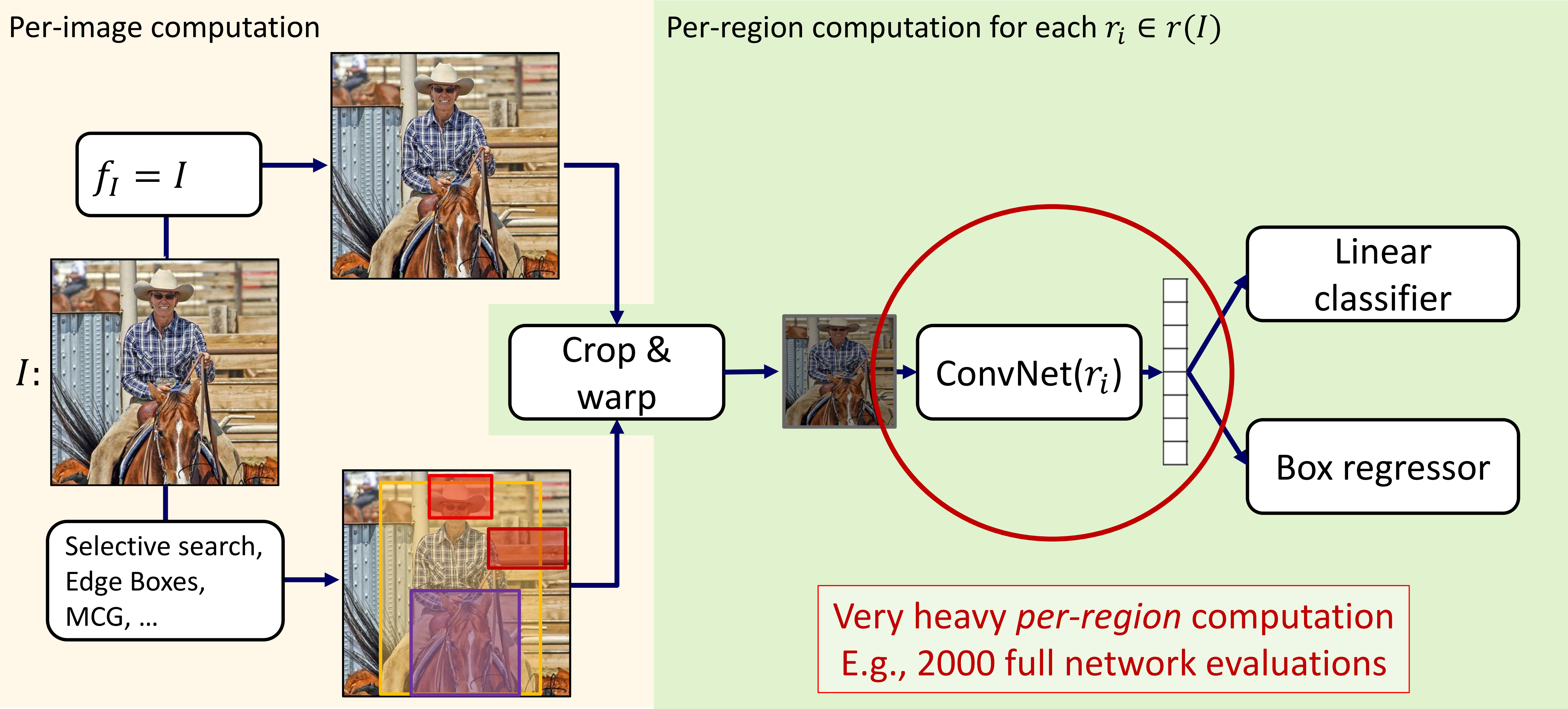
# “Slow” R-CNN


Per-image computation
Per-region computation for each $r_i \in r(I)$
$f_I = I$
$I$:
Selective search, Edge Boxes, MCG, …
Crop & warp
ConvNet($r_i$)
Linear classifier
Box regressor
Very heavy *per-region* computation
E.g., 2000 full network evaluations

# Why is it slow?

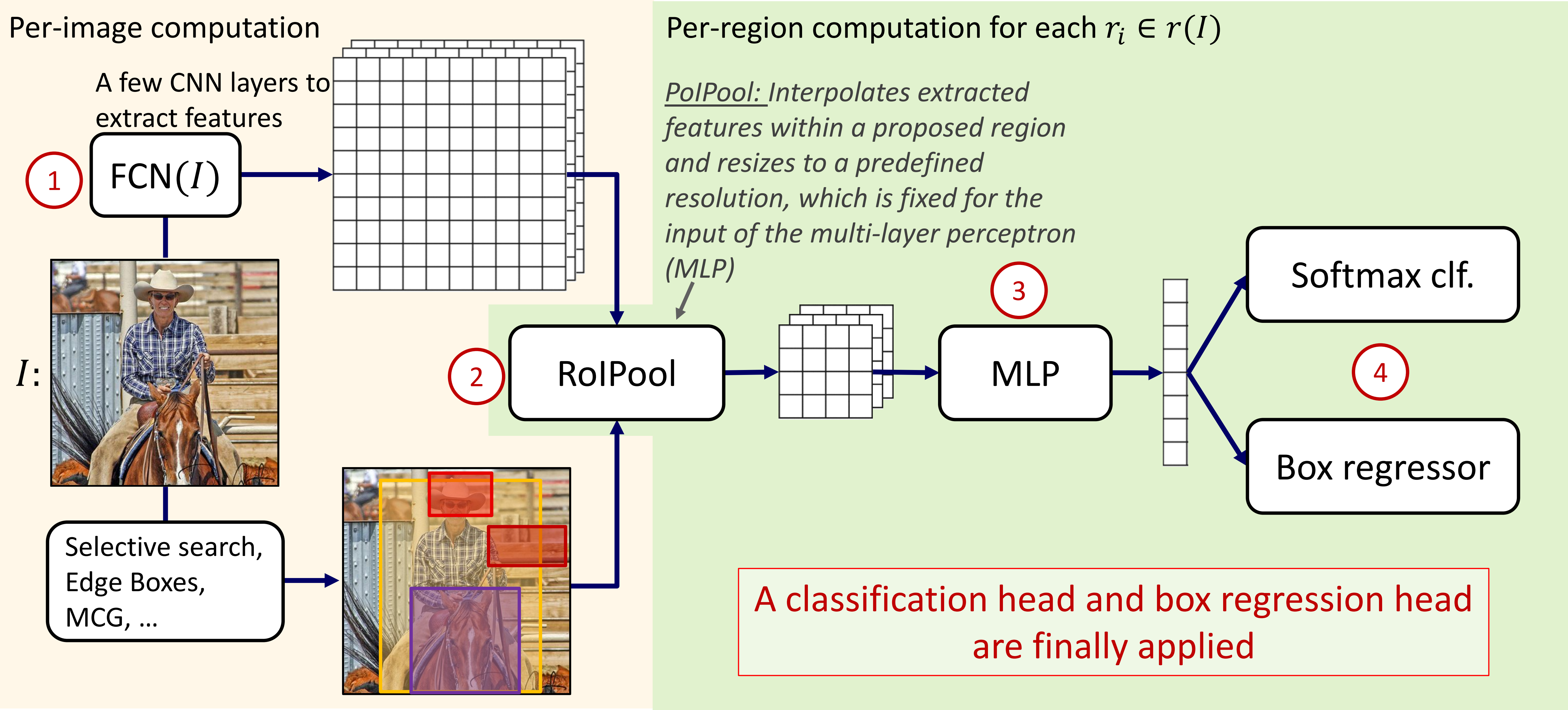


# “Slow” R-CNN

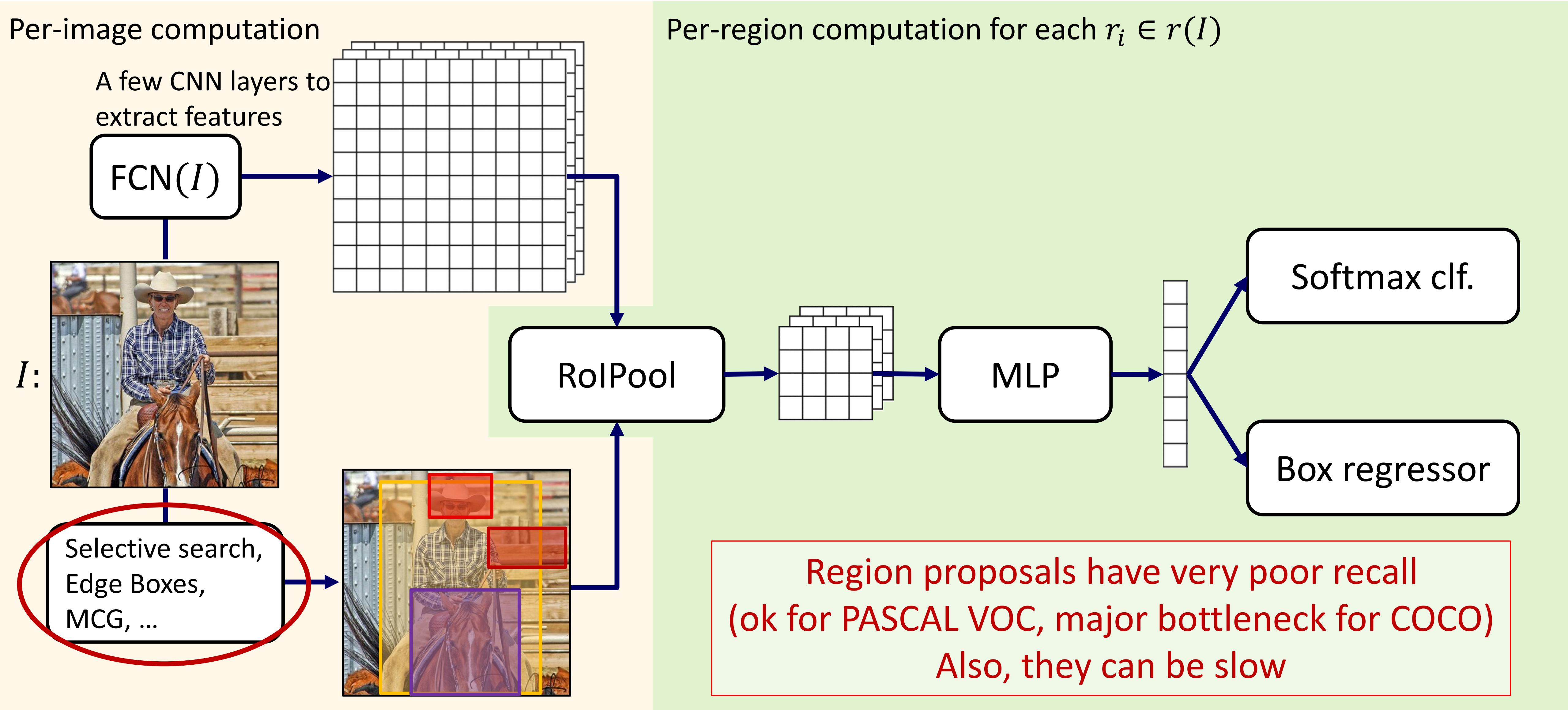

Per-image computation
Per-region computation for each $r_i \in r(I)$
$f_I = I$
$I$:
Selective search,
Edge Boxes,
MCG, …
Crop &
warp
ConvNet($r_i$)
Linear
classifier
Box regressor
Very heavy *per-region* computation
E.g., 2000 full network evaluations

Girshick. Fast R-CNN. ICCV 2015.

# Generalized R-CNN → Fast R-CNN



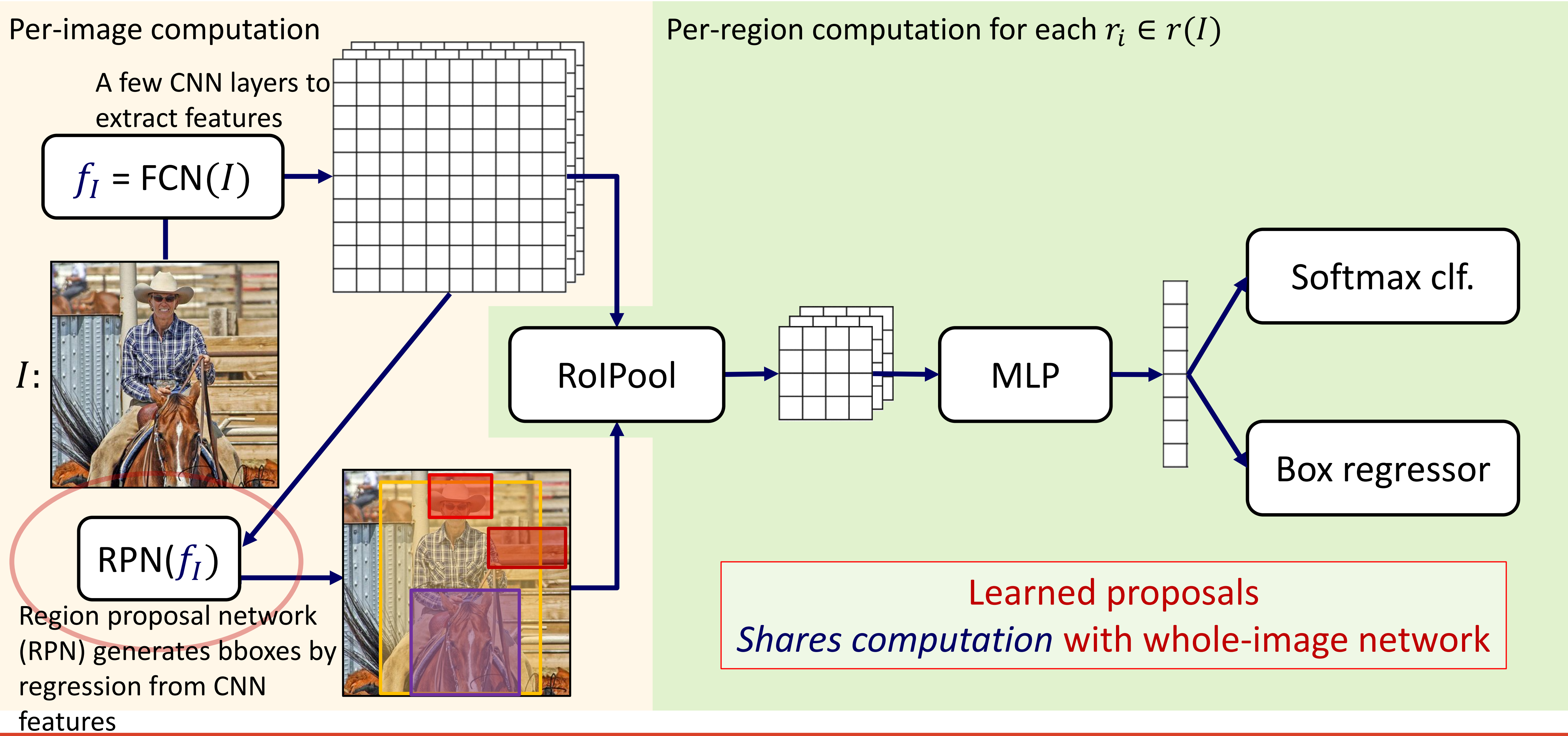
# The Problem with Fast R-CNN




Slide credit: Ross Girshick

Ren, He, Girshick, Sun. Faster R-CNN: Towards Real-Time Object Detection. NIPS 2015.

# Faster R-CNN

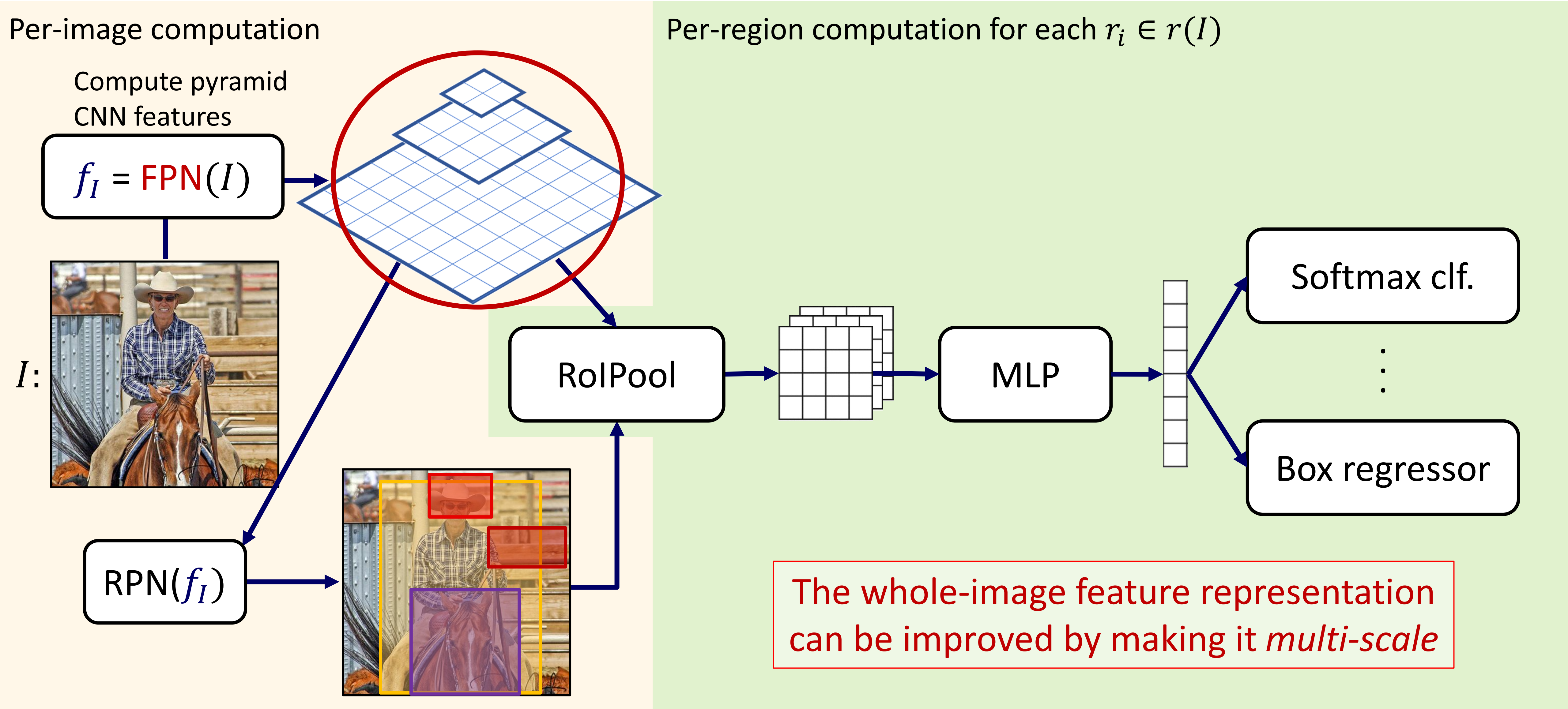



Learned proposals

*Shares computation* with whole-image network

Lin et al. Feature Pyramid Networks for Object Detection. CVPR 2017.

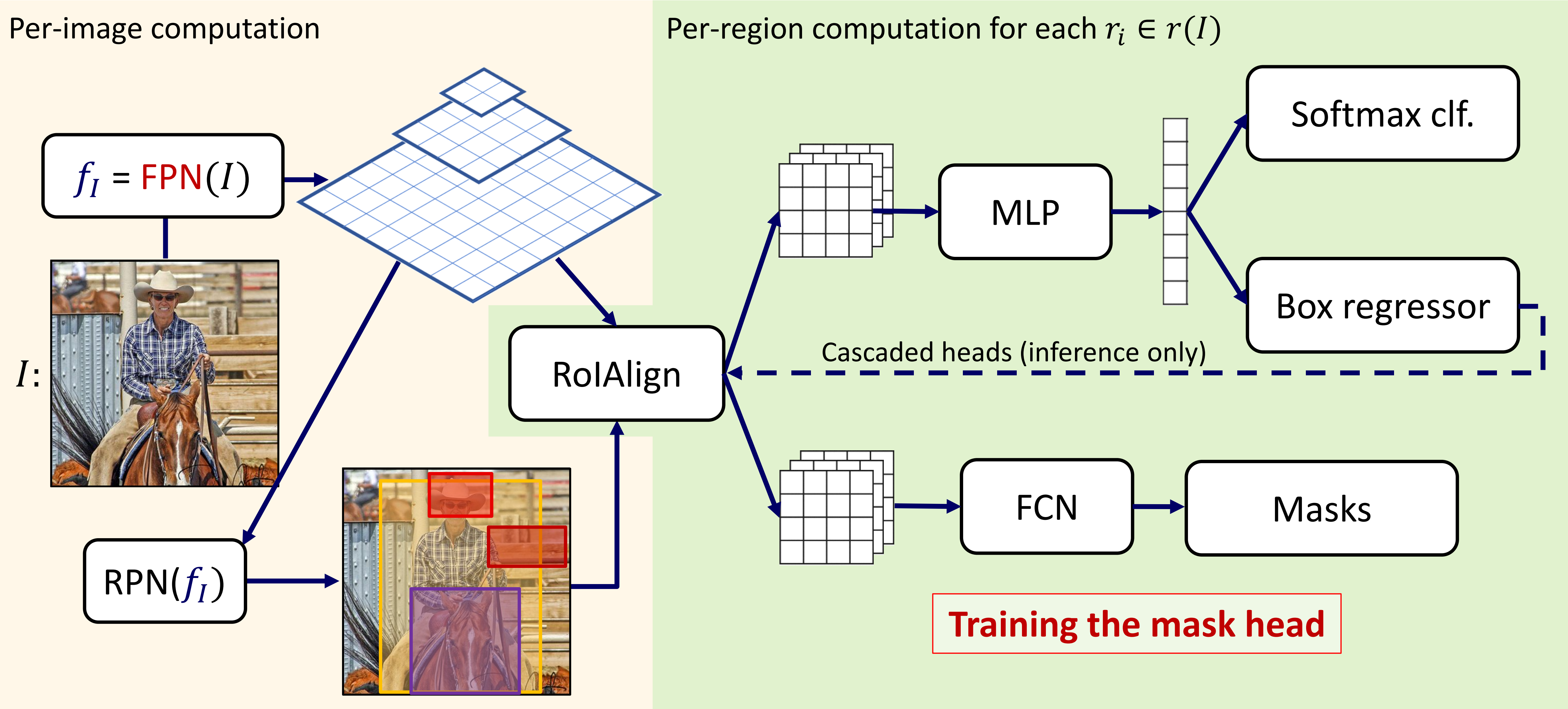
# Faster R-CNN with a Feature Pyramid Network




The whole-image feature representation can be improved by making it *multi-scale*

# Mask R-CNN

He et al. Mask R-CNN. ICCV 2017.

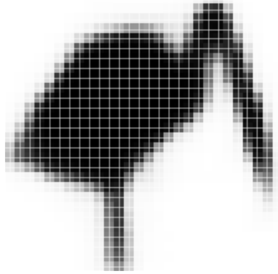


# Mask predictor

- Per-pixel occupancy map is predicted at the regressed bounding box

Validation image with box detection shown in red

28x28 soft prediction

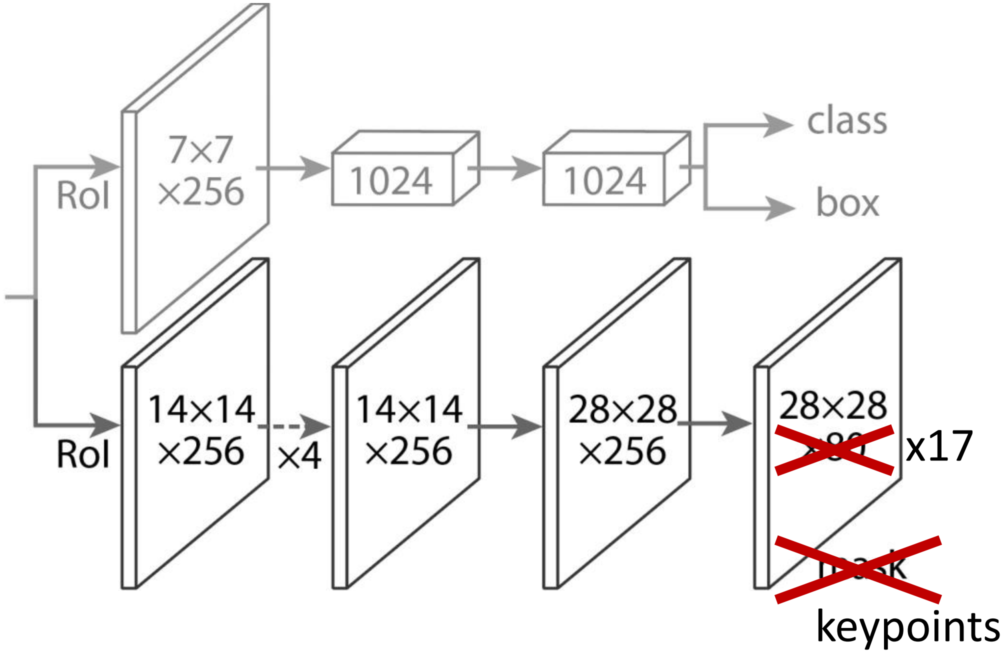


Resized Soft prediction

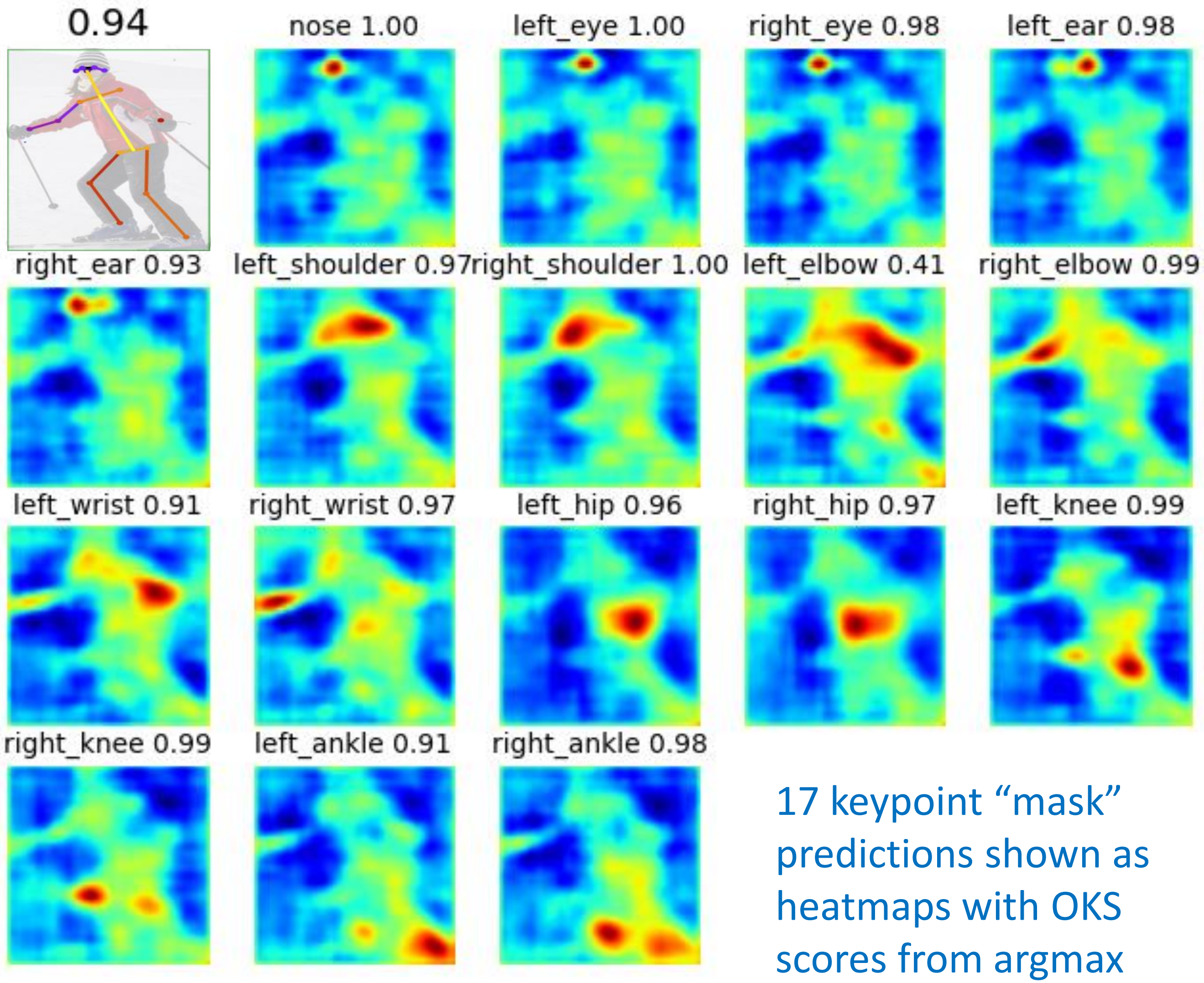
Final mask

# Mask R-CNN application

# Human pose estimation




(Not shown: Head architecture is slightly different for keypoints)

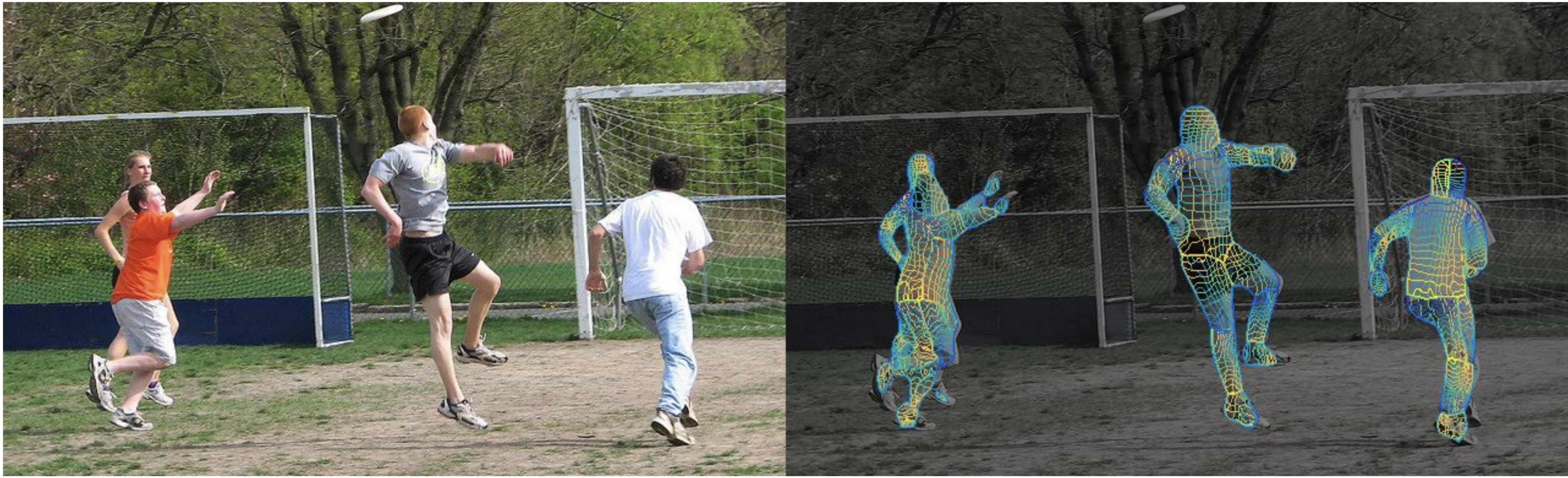



- Add keypoint head (28x28x17)
- Predict one “mask” for each keypoint
- Softmax over spatial locations (encodes one keypoint per mask “prior”)

17 keypoint “mask” predictions shown as heatmaps with OKS scores from argmax positions

# Human pose estimation

# Human “surface” estimation



The CVPR2018 oral available here: https://www.youtube.com/watch?v=Dhkd_bAwwMc

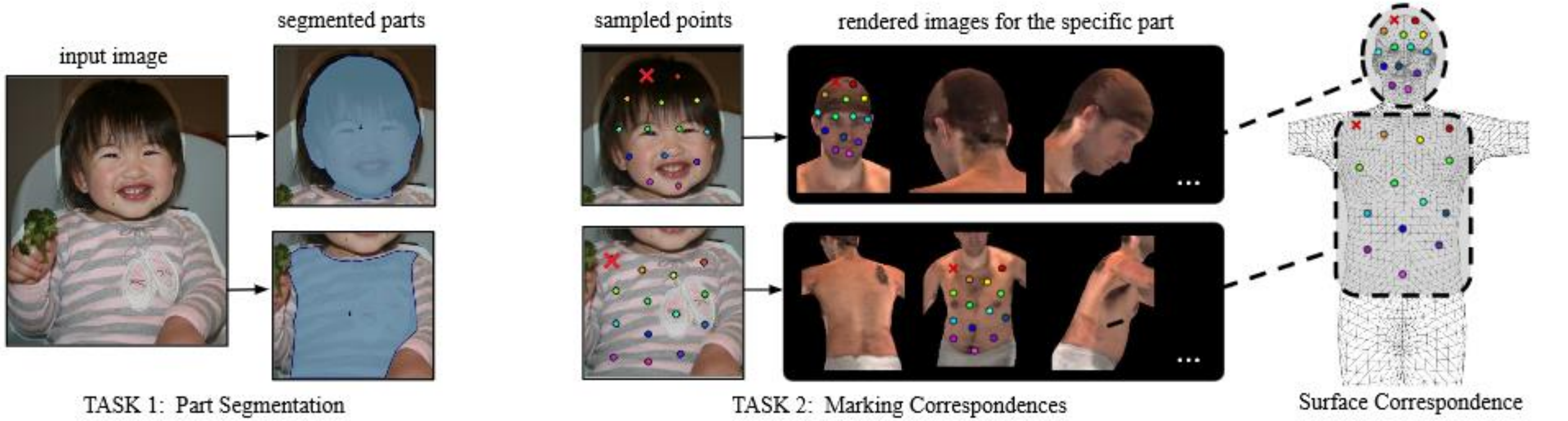
http://densepose.org/

Güler, Neverova, Kokkinos, **DensePose: Dense Human Pose Estimation In The Wild,** CVPR 2018

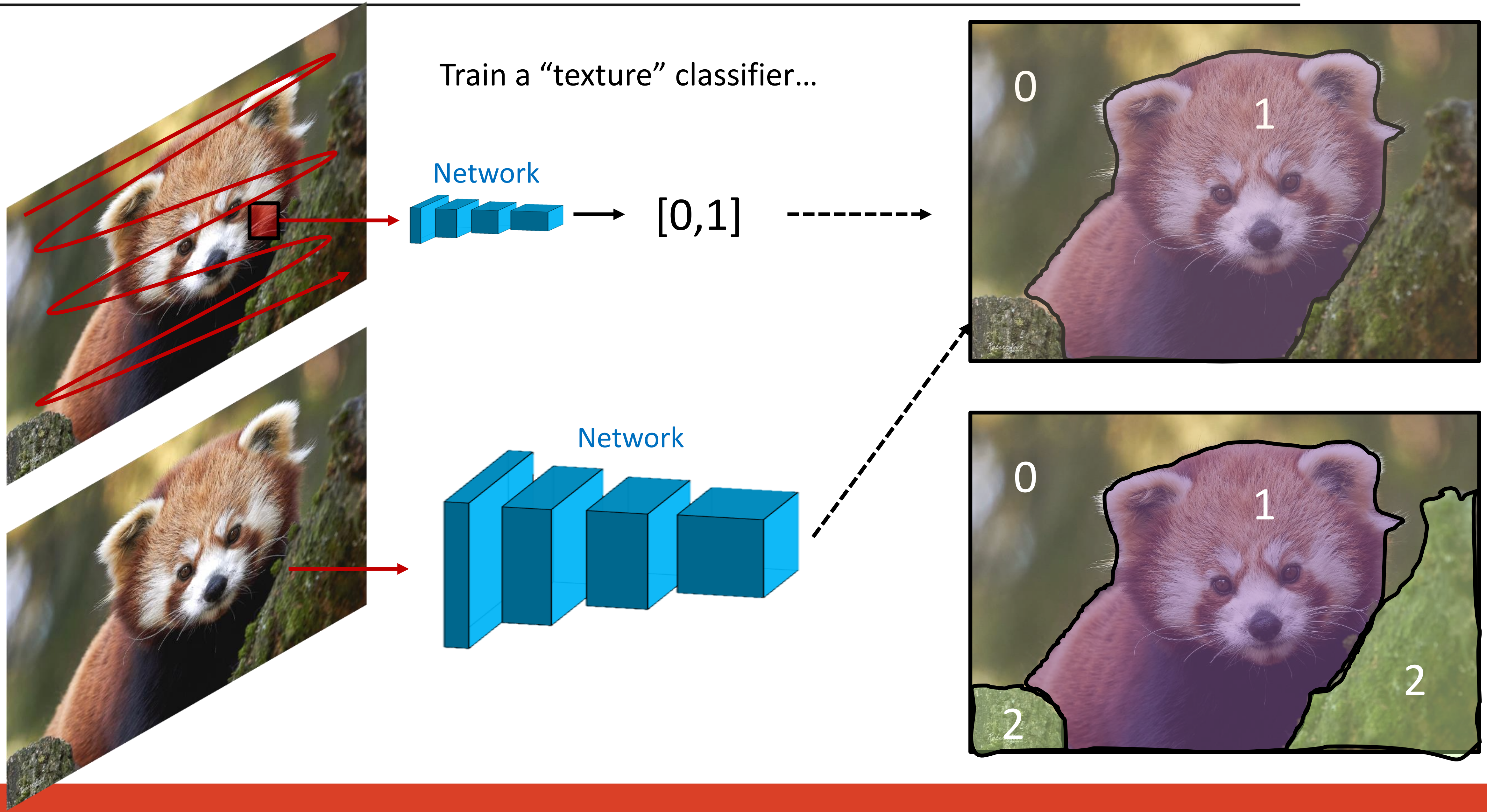
# Human "surface" estimation

Huge effort made to come up with manual ground truth annotations!

**50K humans**, over **5 million** manually annotated correspondences.



# "Semantic" Segmentation

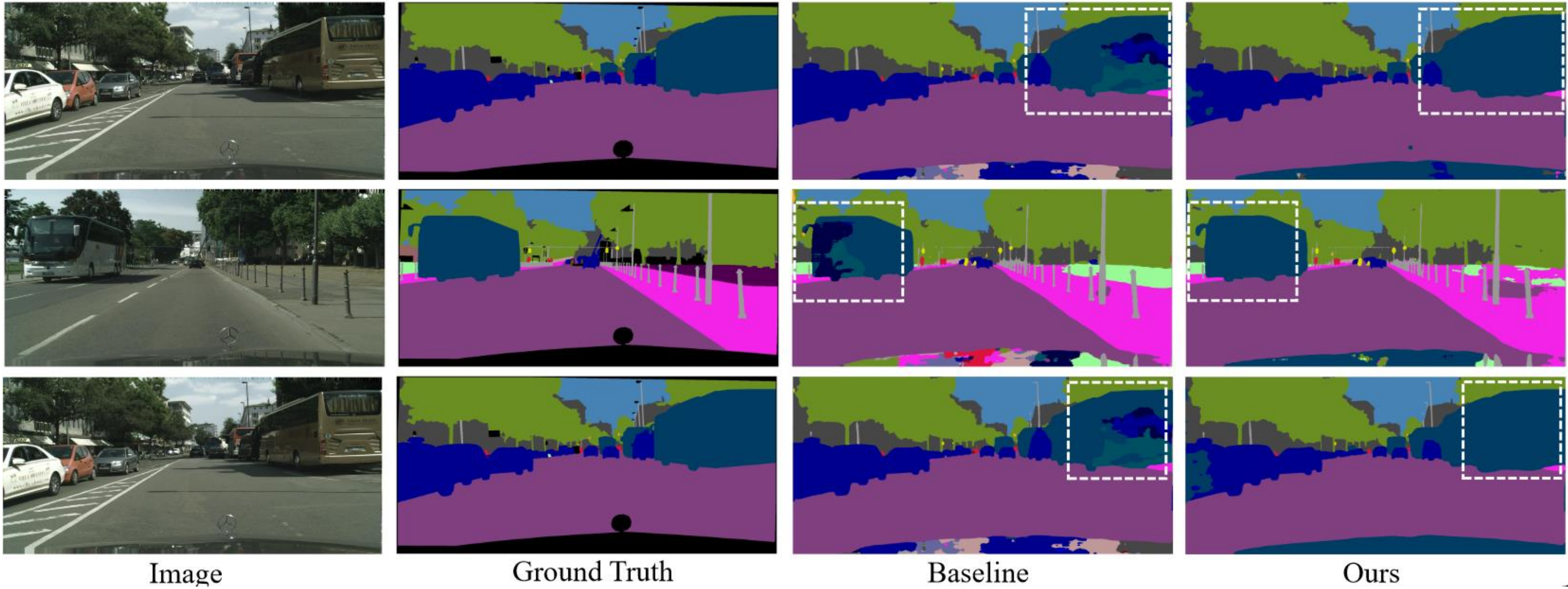


# Autonomous cars

https://blog.mapillary.com/update/2018/06/14/robust-cv.html

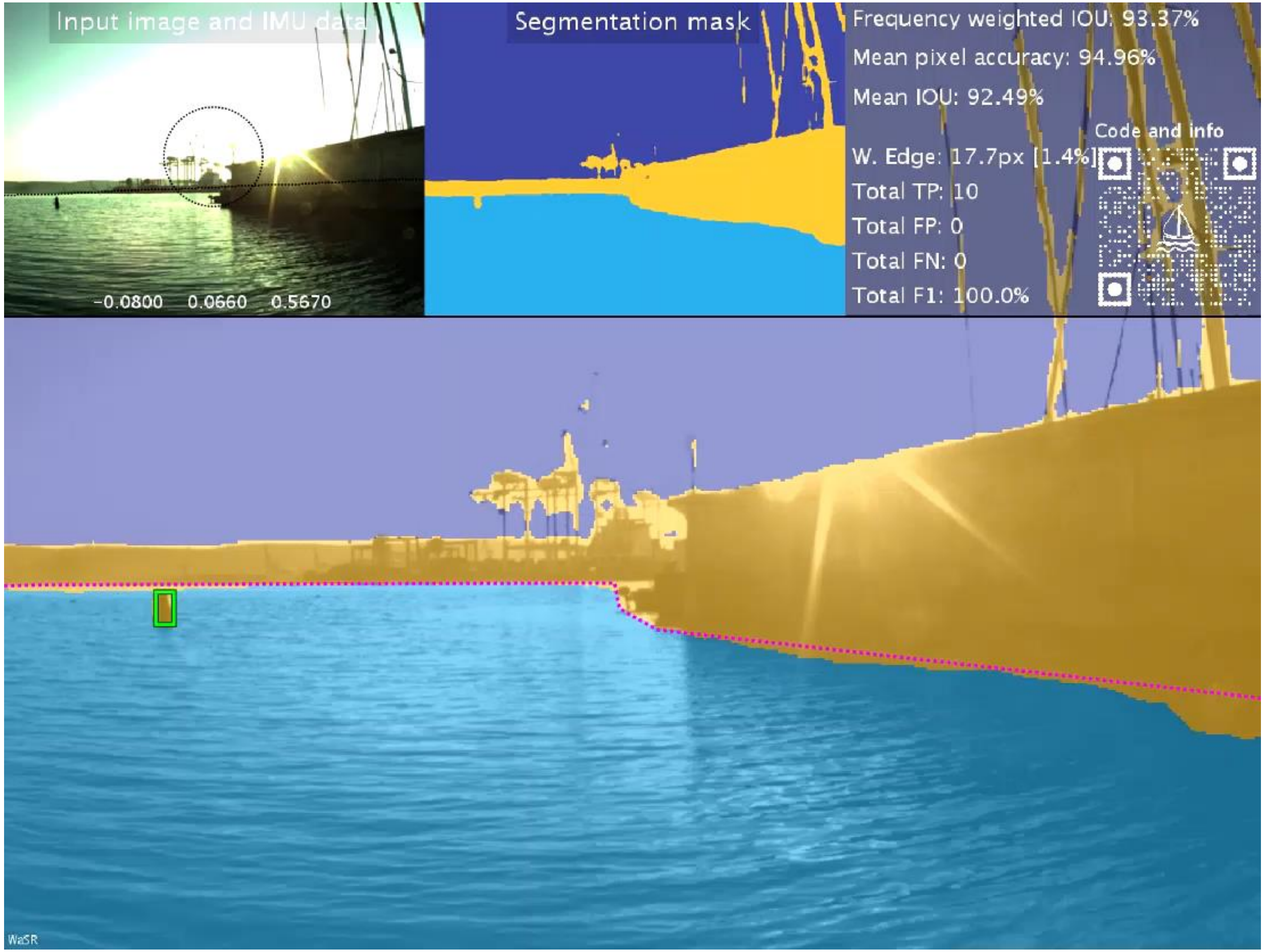
# Autonomous cars

Yuhui Yuan, Xilin Chen, Jingdong Wang, Object-Contextual Representations for Semantic Segmentation, Arxiv (unpublished) 2019




Top performers on the major autonomous cars benchmark Cityscapes in 2019.

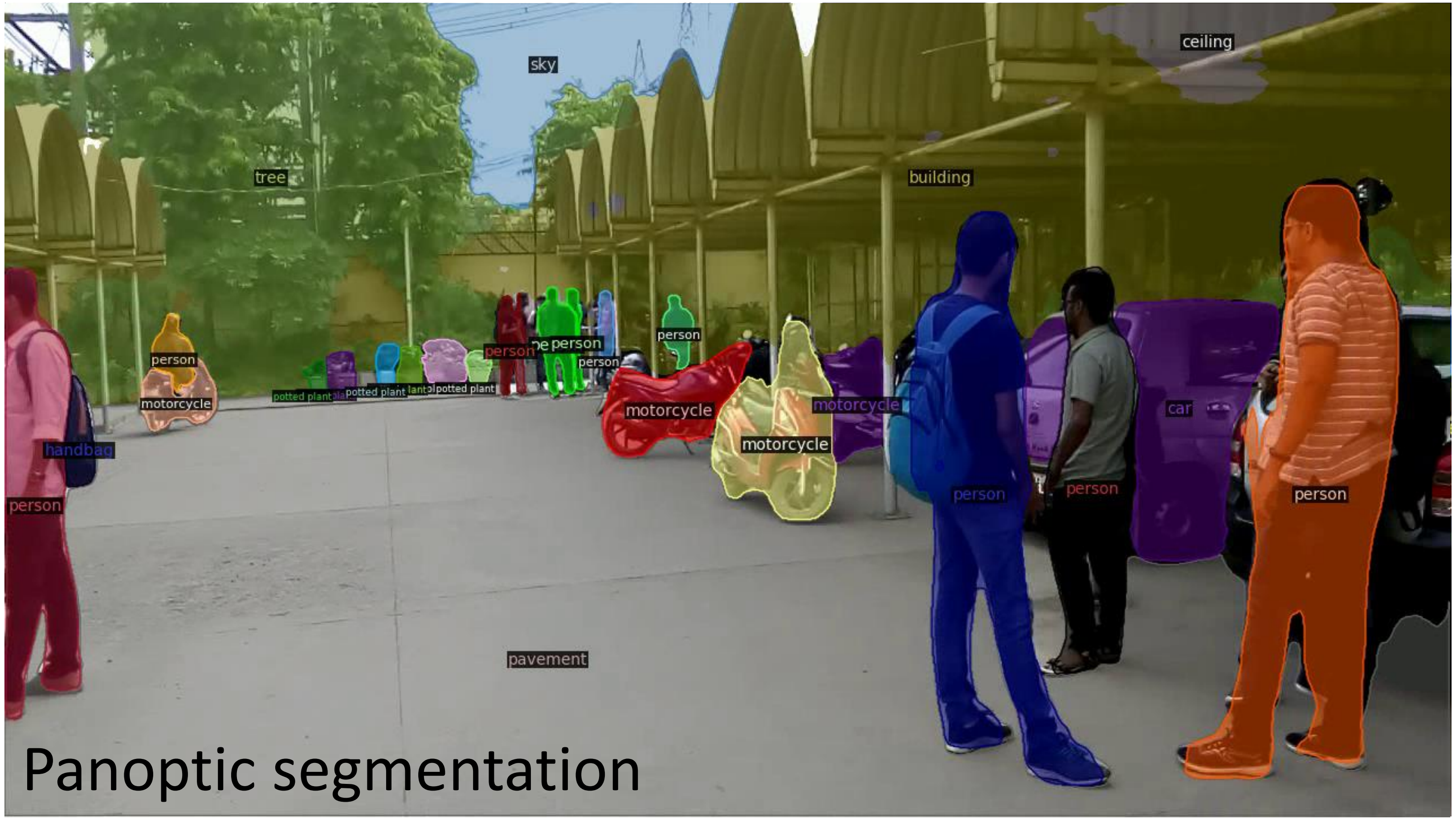
# Autonomous boats




Borja Bovcon, Matej Kristan, A water-obstacle separation and refinement network for unmanned surface vehicles, ICRA 2020

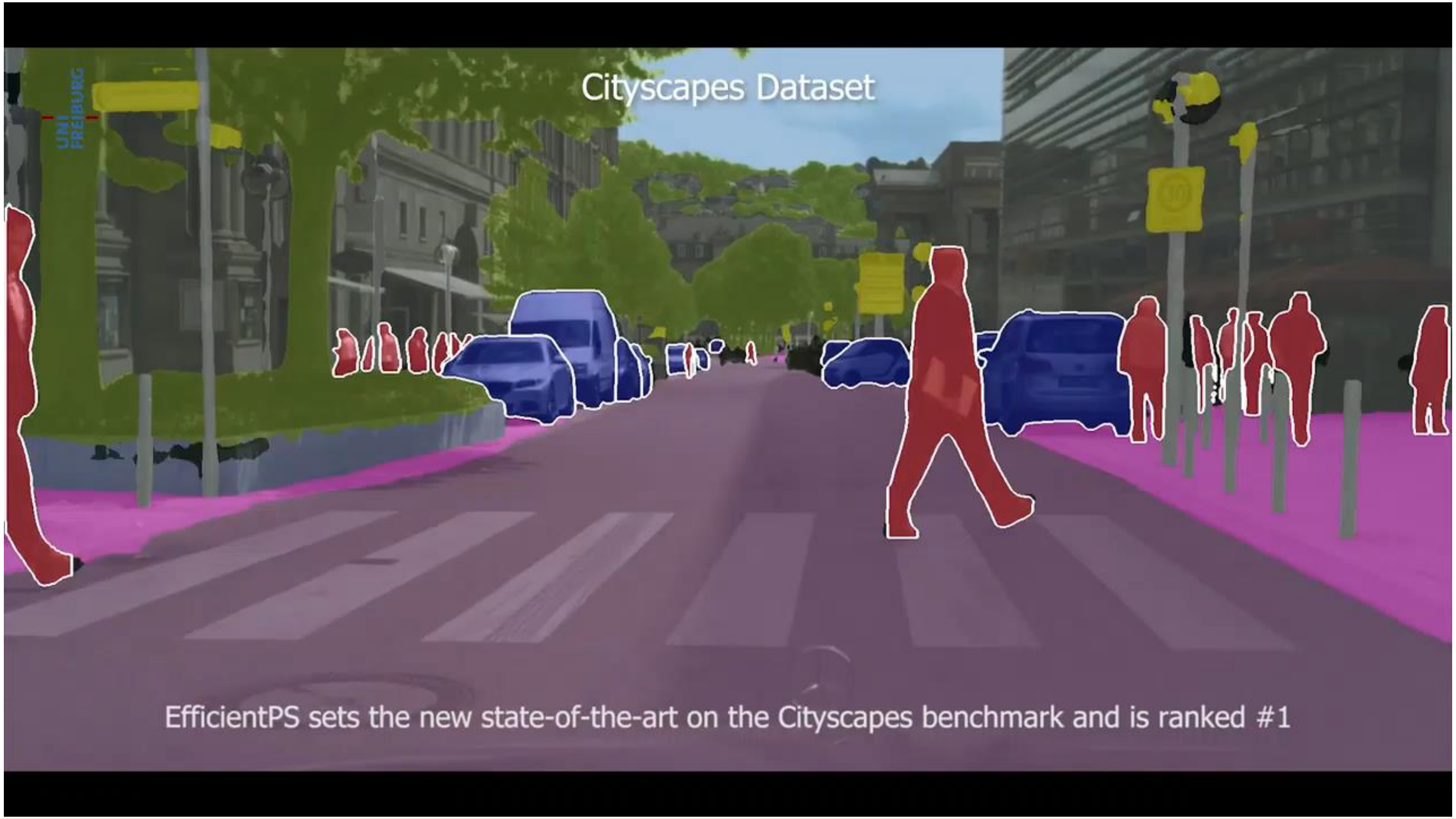
# Panoptic segmentation

- Combines object detection and stuff segmentation



# Panoptic segmentation

- Combines object detection and stuff segmentation

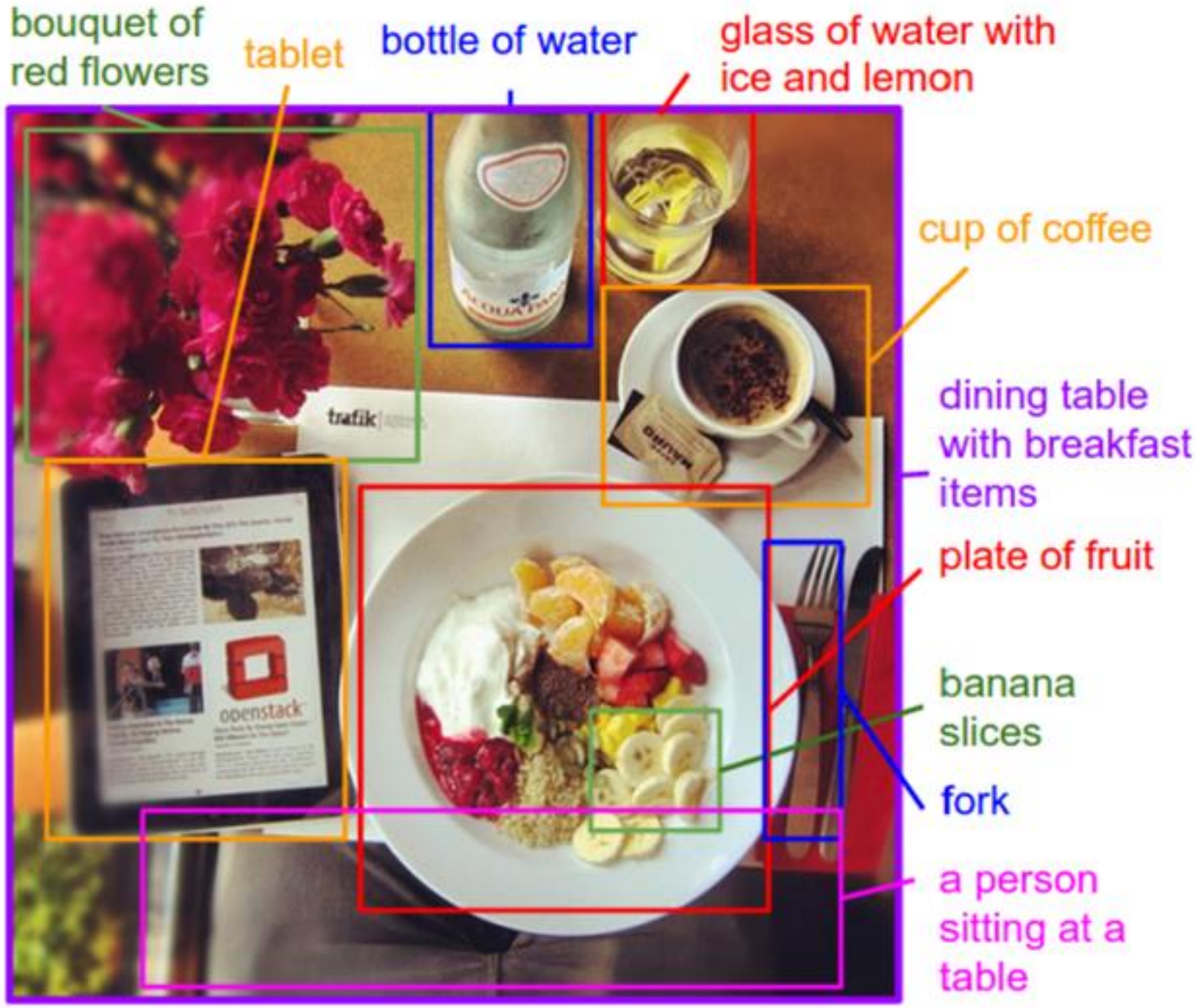



https://www.youtube.com/watch?v=j11mvFqFmfA&t=110s

# CNN generated image descriptions

- Use CNN for feature extraction
- Use RNNs (recurrent neural networks) for word generation
- Take a huge number of images with captions manually annotated and learn by backprop

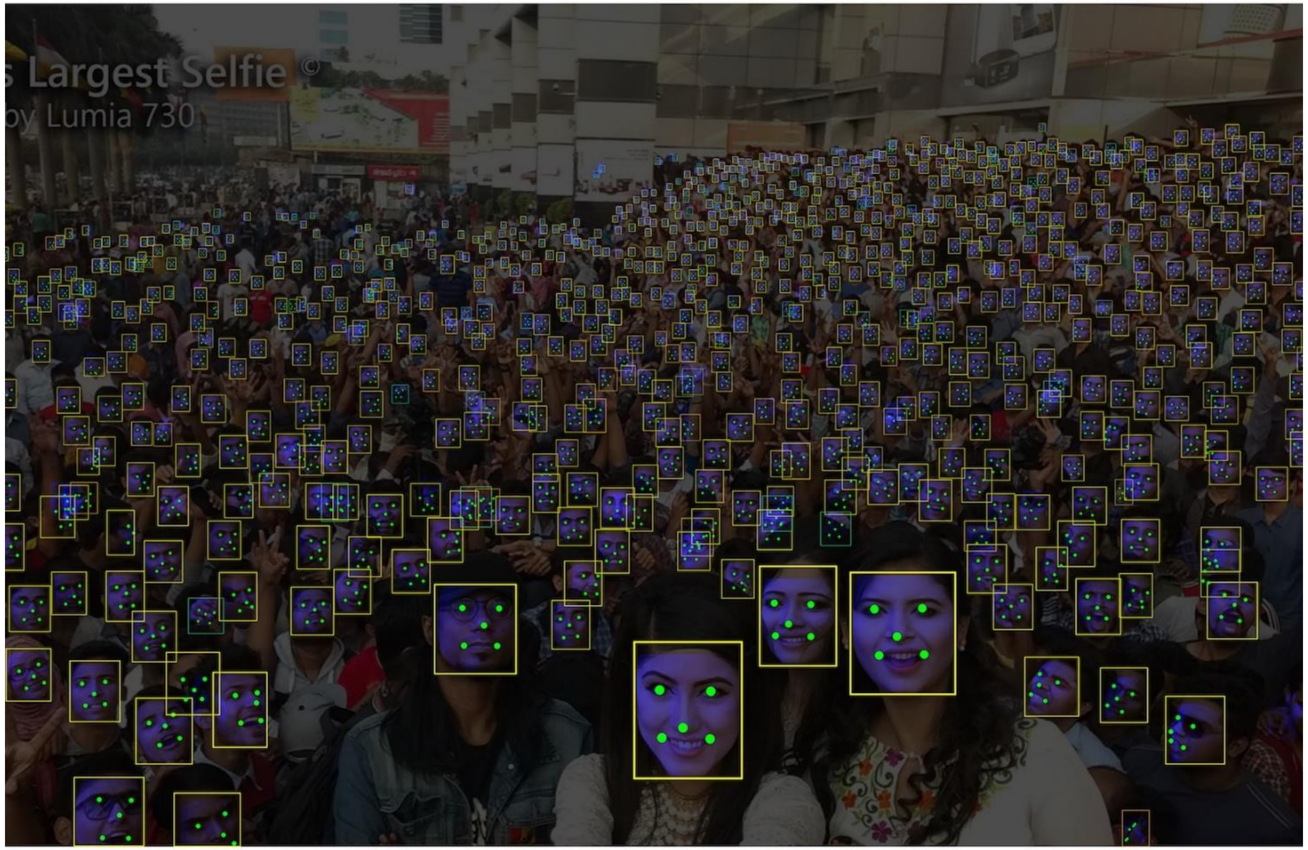
Karpathy and Li, Deep Visual-Semantic Alignments for Generating Image Descriptions, CVPR 2015




Example output of the model

# Beyond Viola Jones (CNN face detection)

https://paperswithcode.com/sota/face-detection-on-wider-face-hard

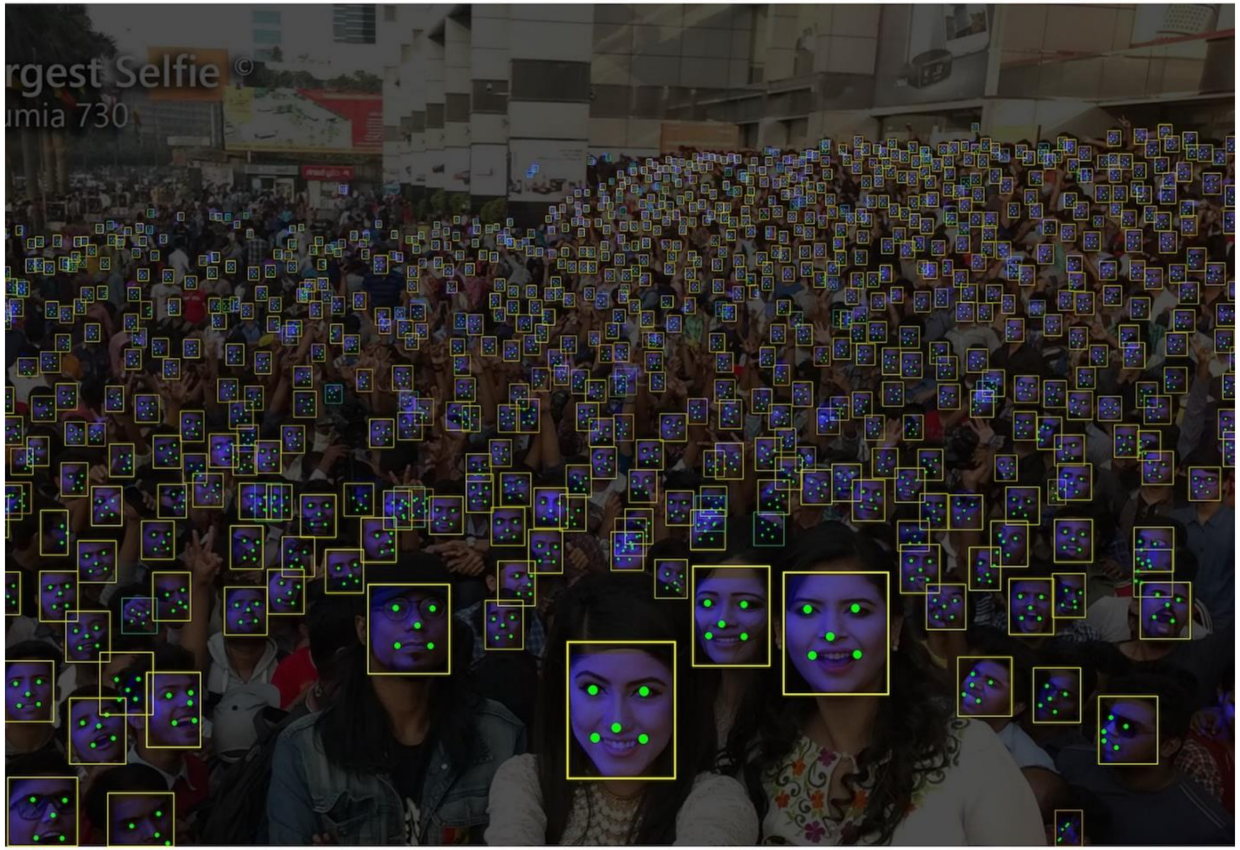



Deng et al., RetinaFace: Single-stage Dense Face Localisation in the Wild, Arxiv2019


Bai et al., Finding Tiny Faces in the Wild with Generative Adversarial Network, CVPR2019

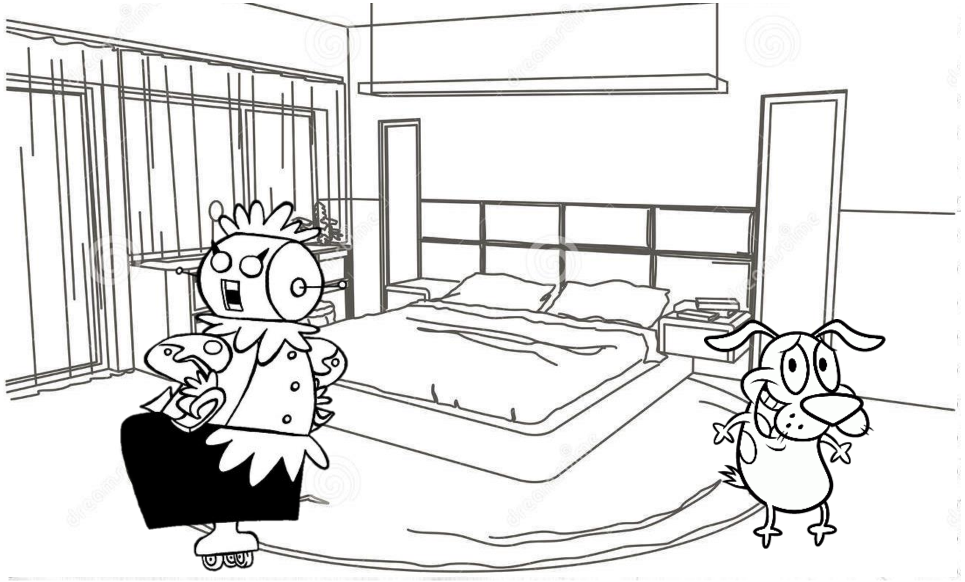
# Viola-Jones (2001) vs RetinaFace (2019)

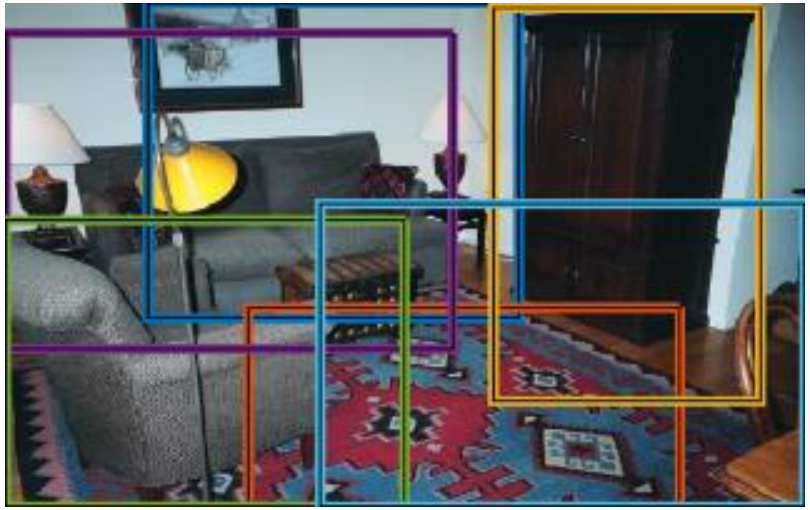

Viola, Jones, *"Rapid Object Detection using a Boosted Cascade of Simple Features"*, CVPR2001




Deng et al., RetinaFace: Single-stage Dense Face Localisation in the Wild, Arxiv2019

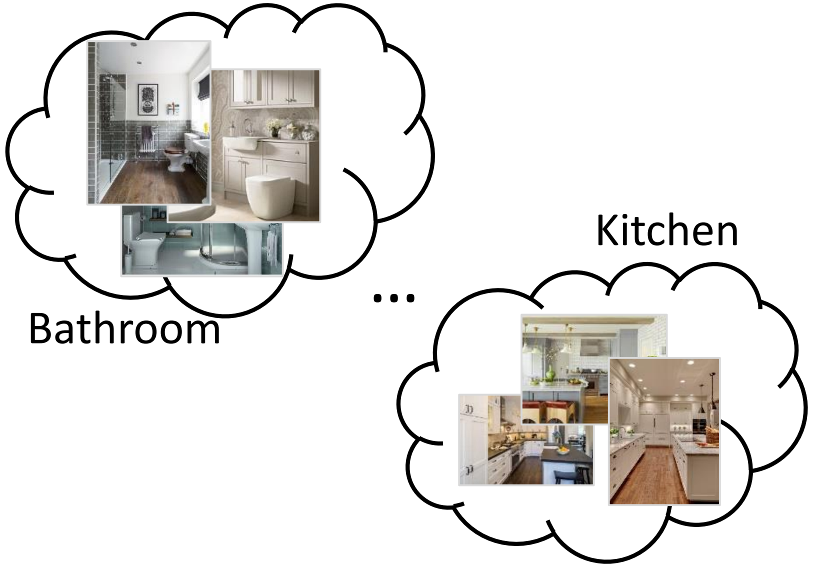
# Robotic vision: Household robots





Place recognition for service robots

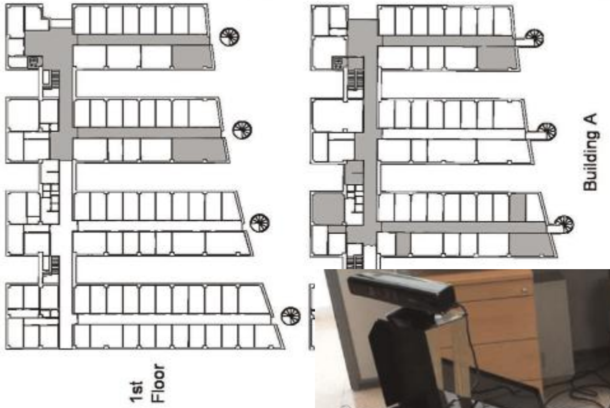
Uršič, Tabernik, Boben, Skočaj, Leonardis, Kristan, IJRAS 2013 ;
Uršič, Leonardis, Skočaj, Kristan, ICRA 2016 ;
Uršič, Mandeljc, Leonardis, Kristan, ICRA 2016
Uršič, Leonardis, Skočaj, Kristan, IJRR 2017



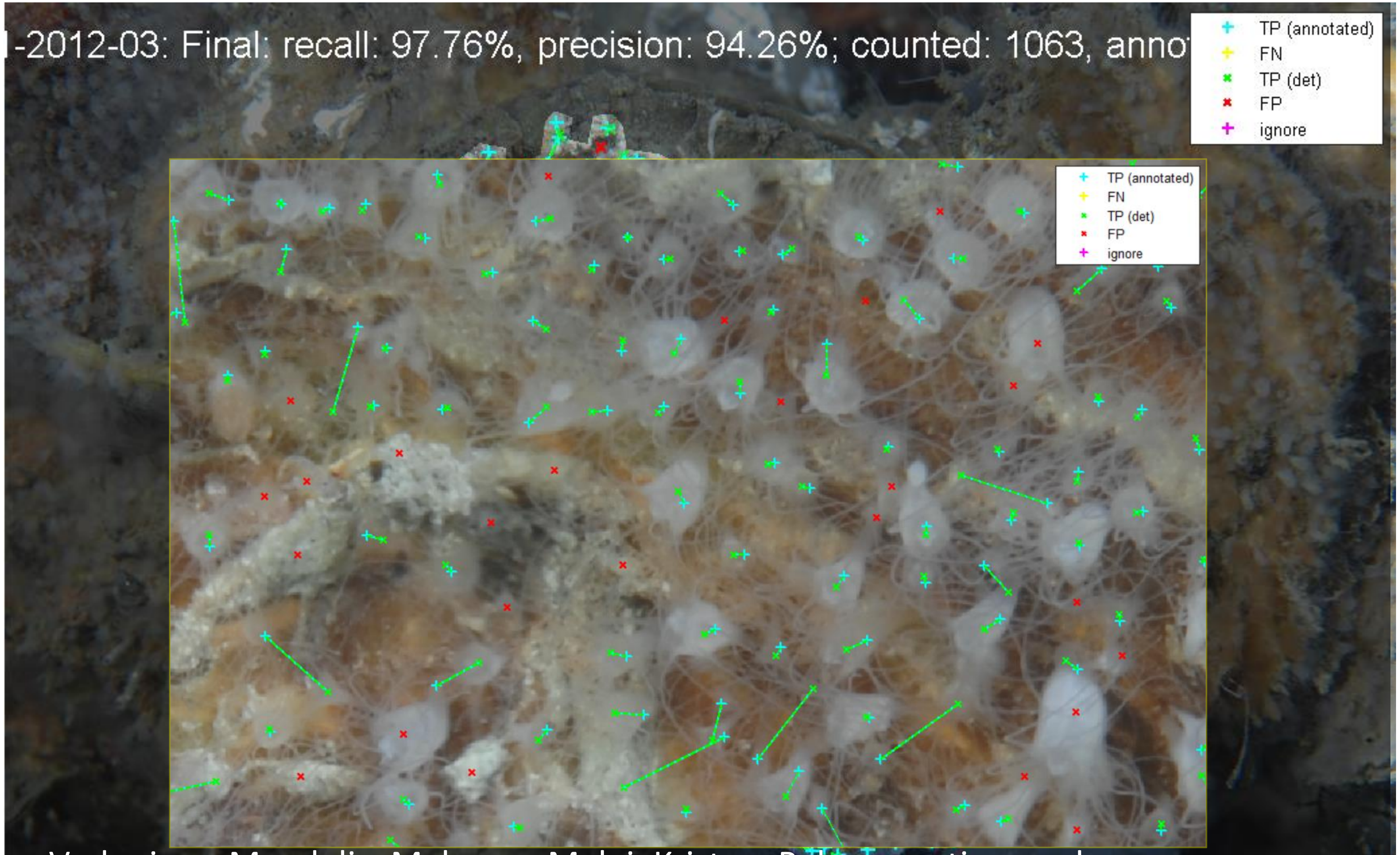

# Robotic vision: Household robots

## Place recognition for service robots

Mandeljc, Uršič, Leonardis, Skočaj, Kristan, (ICRA 2016)



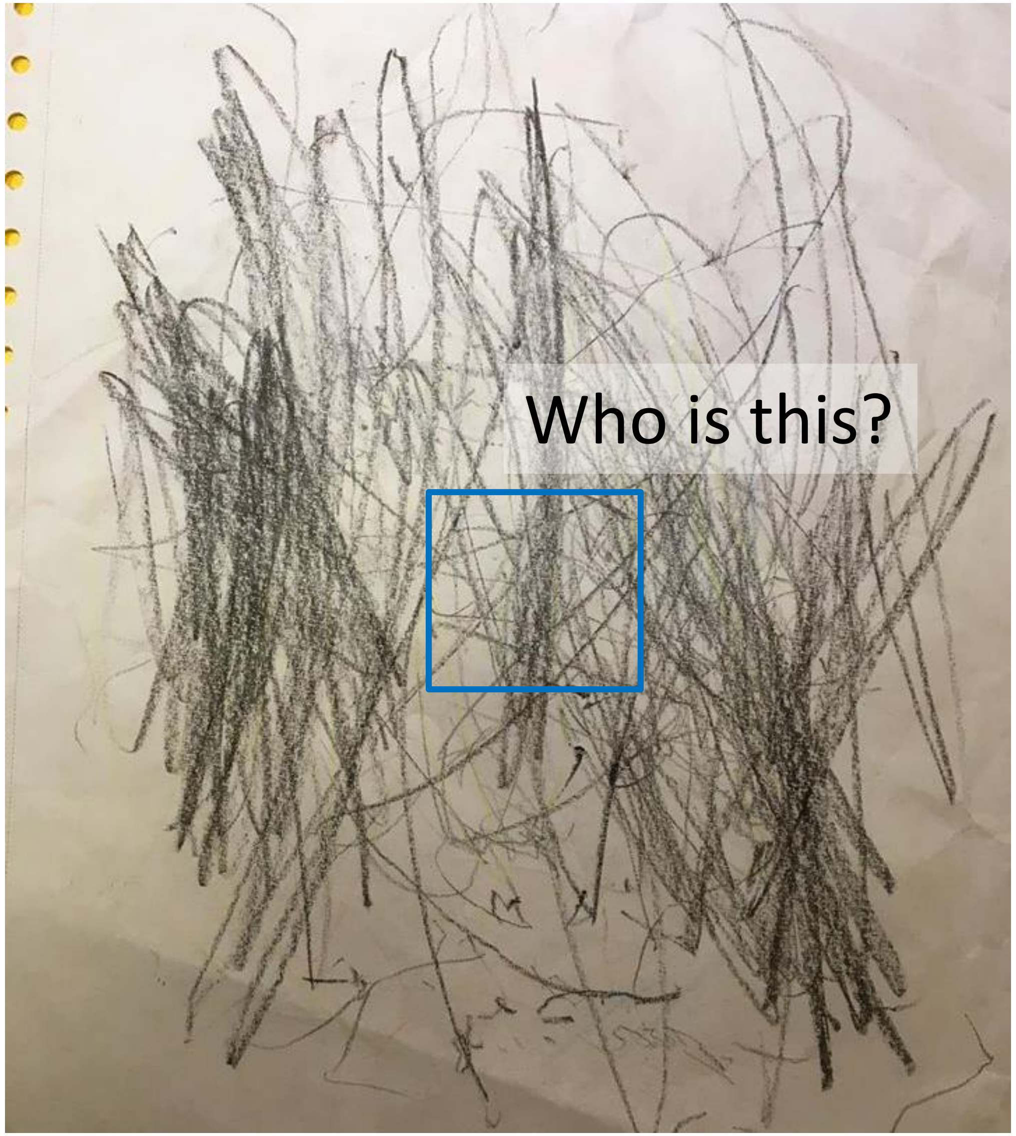

# Automating boring tasks: Counting polyps




Vodopivec, Mandeljc, Makovec, Malej, Kristan, Polyp counting made easy: towards automated scyphistoma census in underwater imagery, (2018)

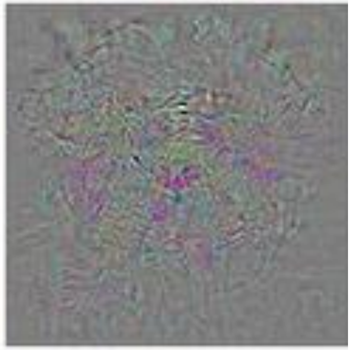
# Facebook Face detection (CNN-based)

“The lady that comes at night
when we’re all asleep…”



# CNNs and “human performance” fallacy

- Adding small (but specific!) perturbations to images

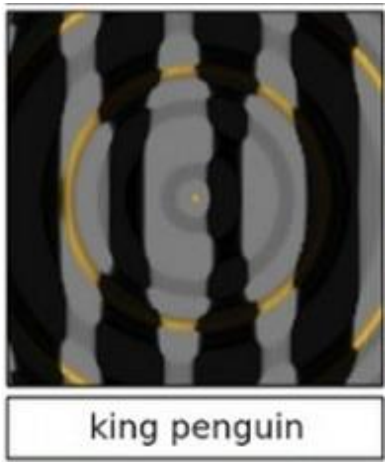


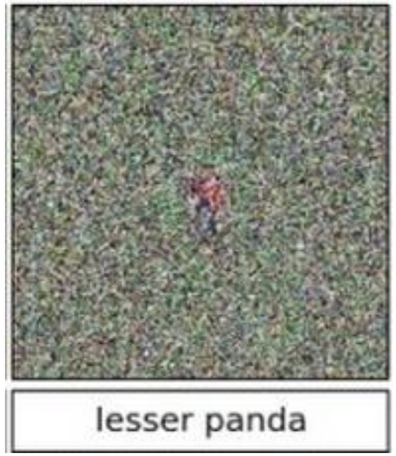
=

Ostrich!

dog + perturbation

- Generating „adversary“ images




Over 99.6% confidence in decision!

Nguyen et al., Deep Neural Networks are Easily Fooled: High Confidence Predictions for Unrecognizable Images, CVPR 2015 (http://www.evolvingai.org/fooling)

https://karpathy.github.io/2015/03/30/breaking-convnets/

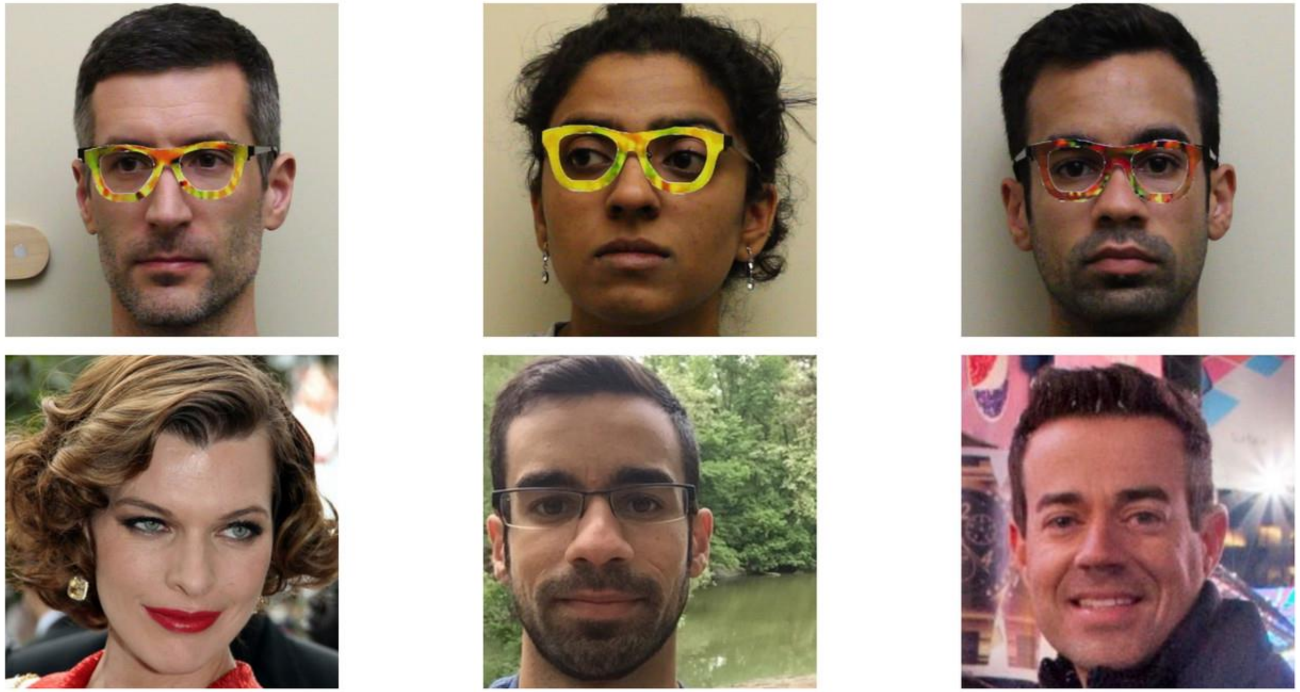
# Not only images, 3D objects too


Vincent, J. **Google's AI thinks this turtle looks like a gun, which is a problem**
https://www.theverge.com/2017/11/2/16597276/google-ai-image-attacks-adversarial-turtle-rifle-3d-printed

# CNN invisibility glasses

- ANY current learning algorithm extracts features to do a classification
- Recall the Adaboost face detector

Identity change

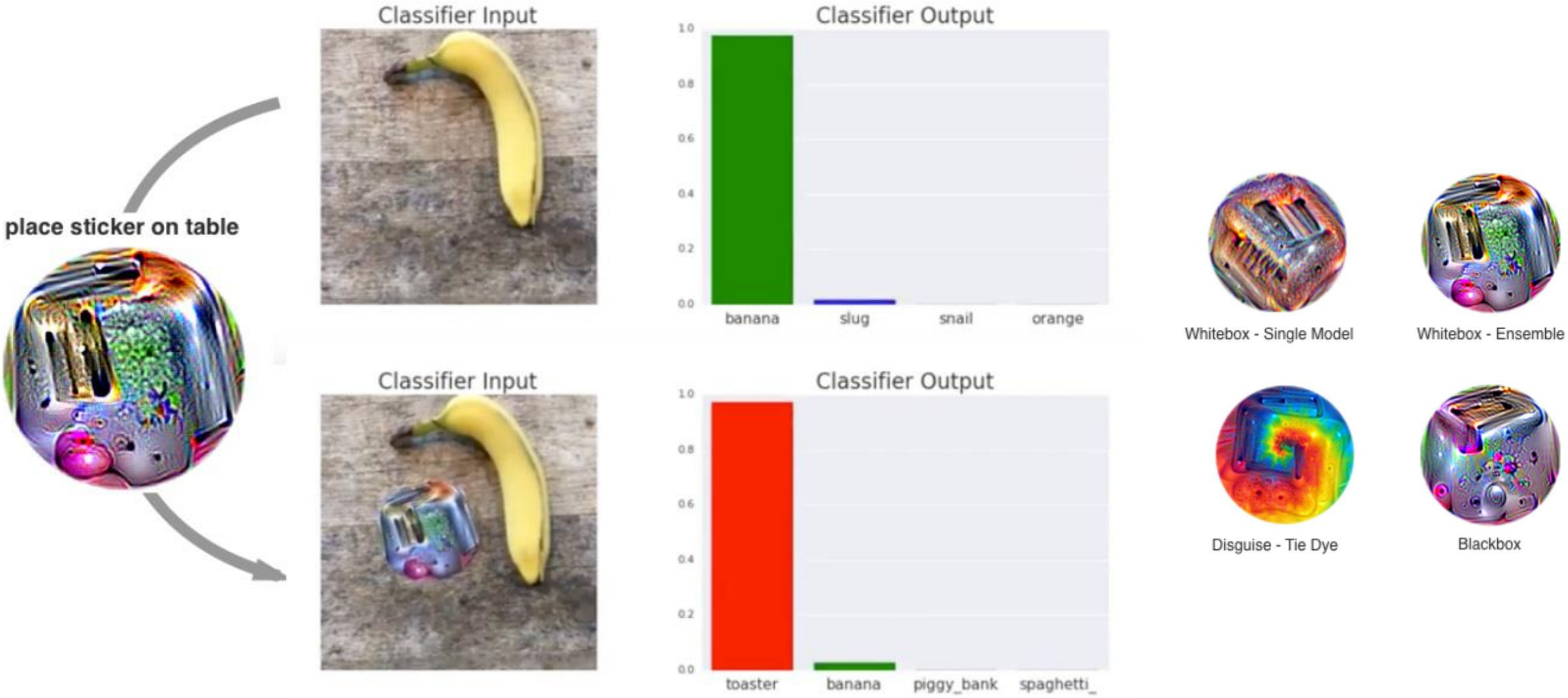


Cloaking device

https://www.youtube.com/watch?v=6Xh1vuwnVhU

Sharif et al., Accessorize to a Crime: Real and Stealthy Attacks on State-of-the-Art Face Recognition, CCS '16

# CNN false classification stickers




https://techcrunch.com/2018/01/02/these-psychedelic-stickers-blow-ai-minds/

# References


- David A. Forsyth, Jean Ponce, Computer Vision: A Modern Approach (2nd Edition), (prva izdaja dostopna na spletu)
- R. Szeliski,Computer Vision: Algorithms and Applications, Springer, 2011
- Viola, M. Jones, Robust Real-Time Face Detection,  IJCV, Vol. 57(2), 2004.
- Viola-Jones Face Detector
  - C++ implementation in OpenCV [Lienhart, 2002]
    - http://sourceforge.net/projects/opencvlibrary/
  - Matlab wrappers:
    - http://www.mathworks.com/matlabcentral/fileexchange/19912
- Convolutional neural networks
  - Yan LeCun, http://yann.lecun.com/
  - Caffe, Torch, Tensor flow, etc.
  - Ross Girshick, The Generalized R-CNN Framework for Object Detection, ECCV2018 tutorial (link)
  - Kaiming He, Learning Deep Representations for Visual Recognition, ECCV2018 tutorial (link)