Object Detection

Computer Vision – Lecture 10

Further Reading

- Slides from F Li.
- Slides from A Zisserman.
- Slides from S Lazebnik.
- CVPR 2019 Tutorial.

Classification

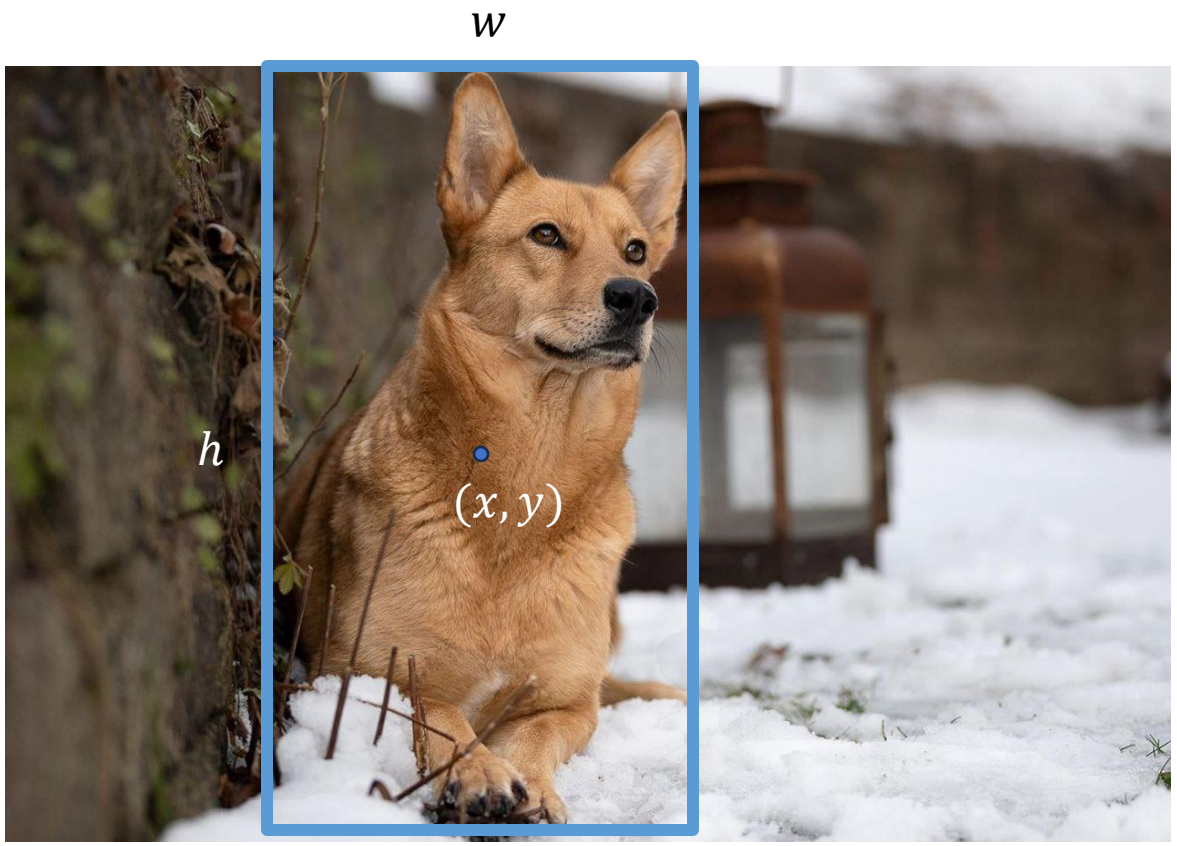
- So far: one class per image.
- Fixed output size: one image in, one probability distribution out.
- Real world: not that simple.
- Scene understanding:
  - what is in the image and where.
  - Object categories, instances, identities, properties, activities, relations, …

Object detection

Image source

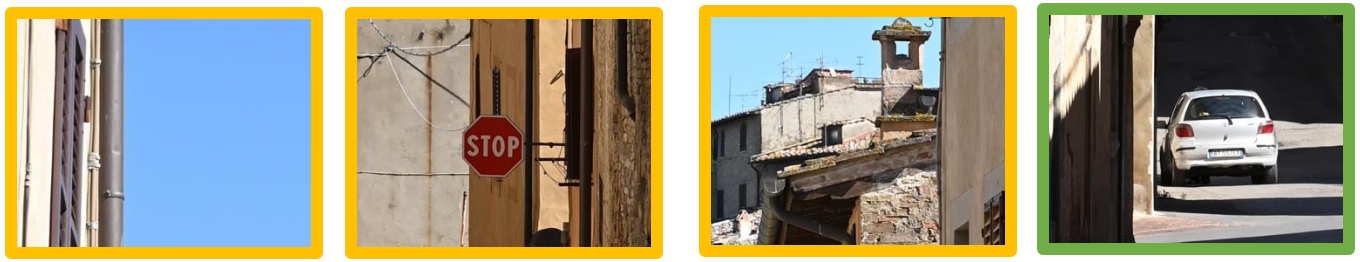
Object Detection

- Task: find the location of an object in the image.
- Typically, with a *bounding box* (x, y, w, h).
- Very often: add class prediction to the bounding box (x, y, w, h, c).



Sliding Window

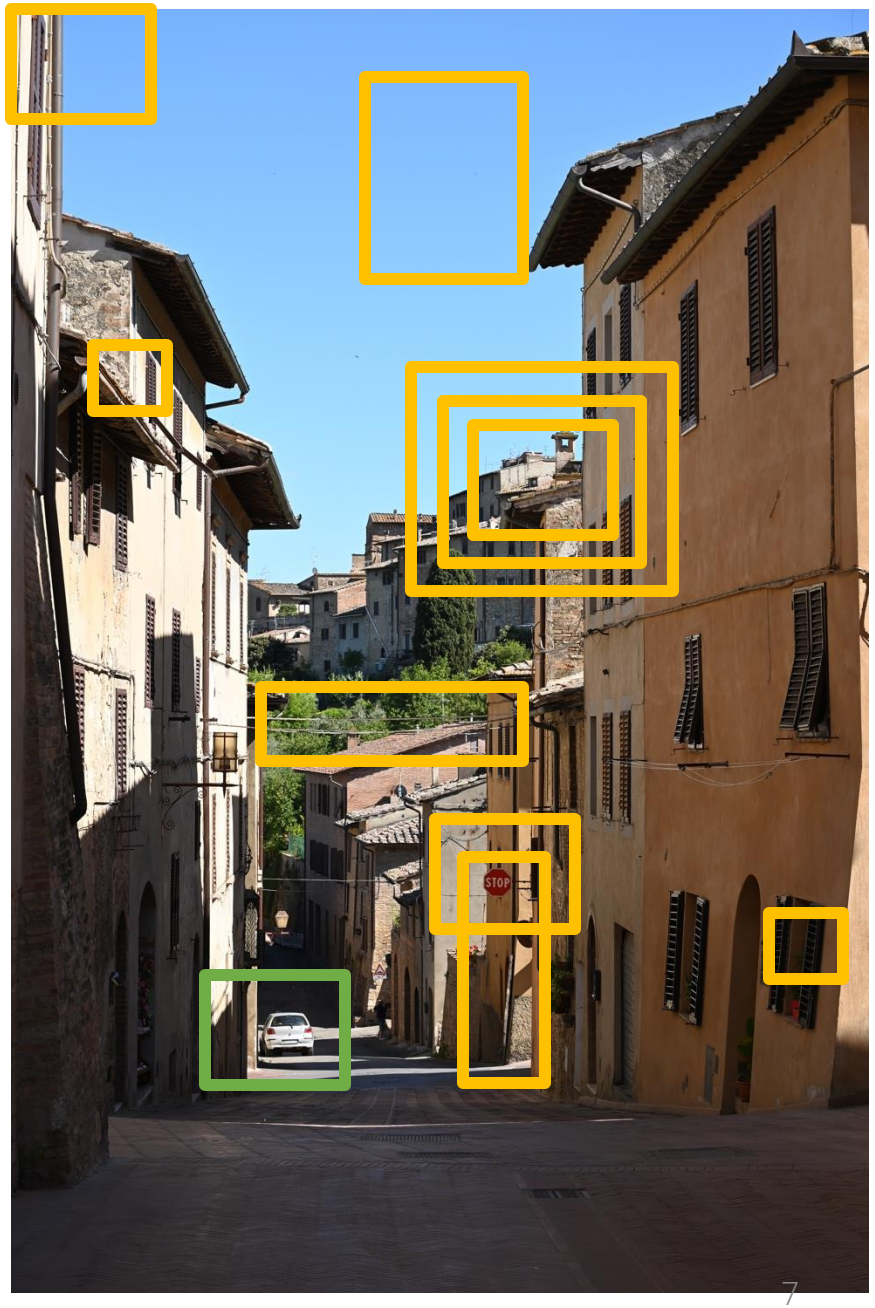
If we have a binary classifier to predict “object” or “background”, we can use it iteratively to detect objects.



background background background car

Sliding Window

- Exhaustive search over locations.
- Often also search over scale and aspect ratio!
- Needs to train on a dataset of object vs. background examples.
- What are representative negative examples?

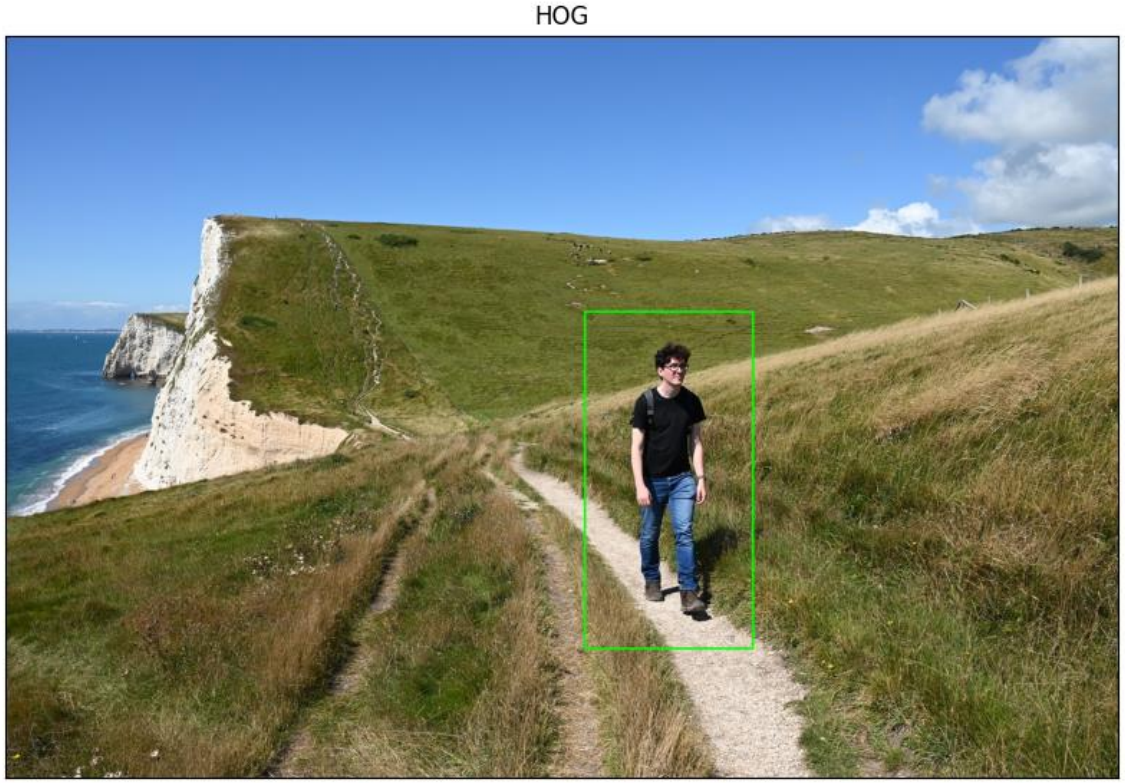


Sliding Window

Typical Problems:

- Computational cost
- Occlusion & truncation
- Multiple responses

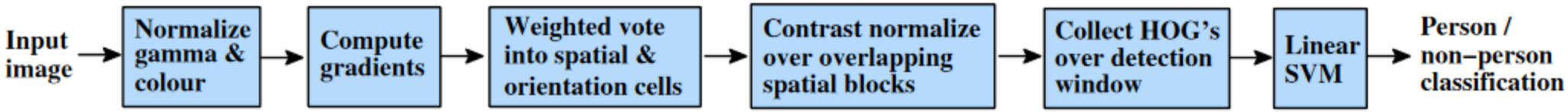
Pedestrian Detection



Dalal & Triggs, CVPR 2005

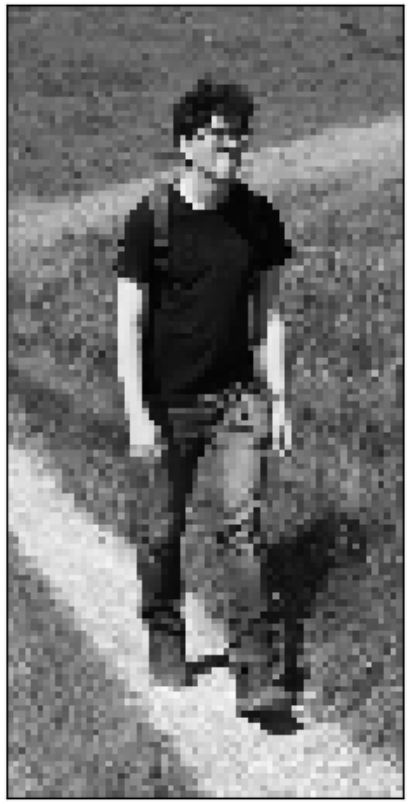
Objective: detect standing people

- Sliding window classifier
- Train a binary SVM classifier: person or not
- Histogram of Oriented Gradients (HOG) feature

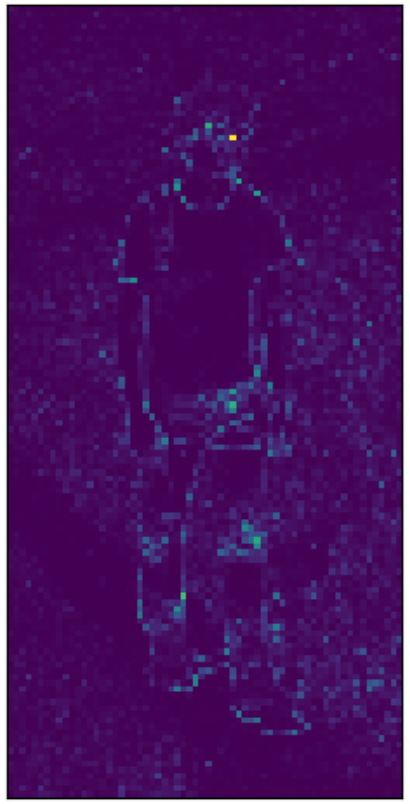


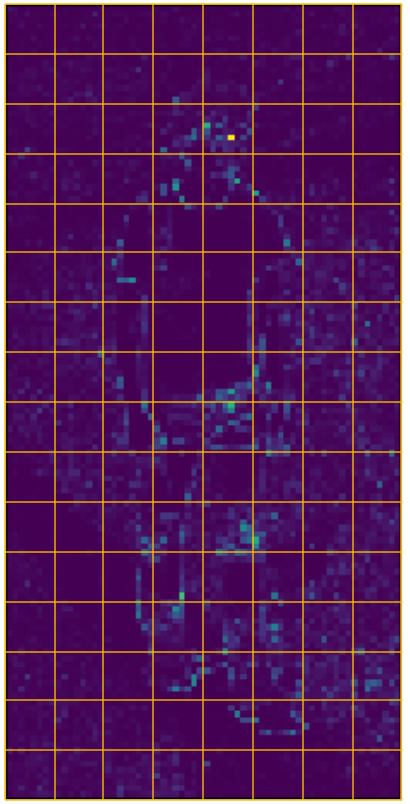
Histogram of Oriented Gradients

Crop

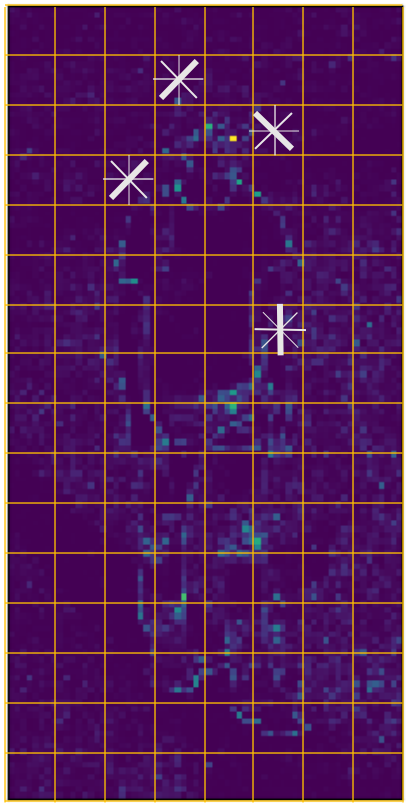


Gradient





split into 8x8 px patches

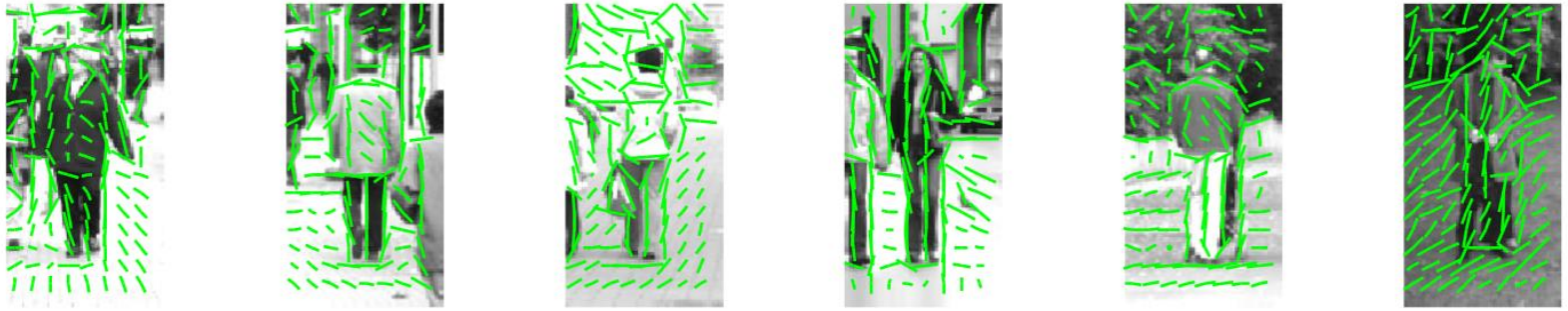


per patch: histogram of gradient directions

Examples

Image

Dominant gradient



HOG

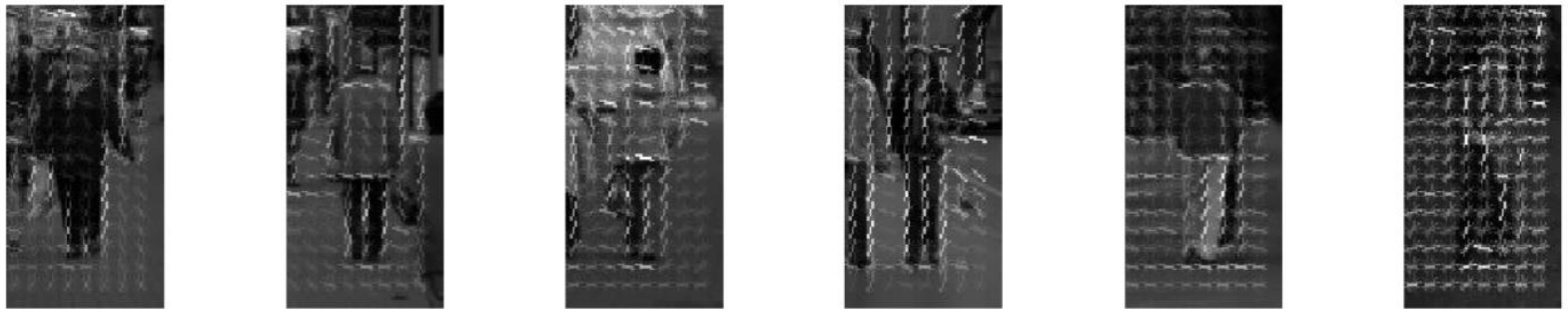
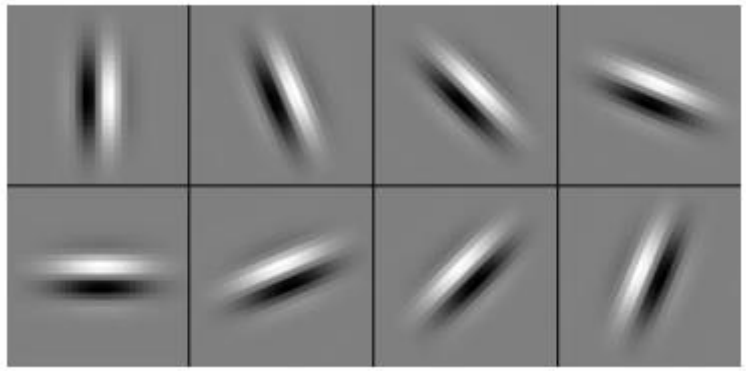
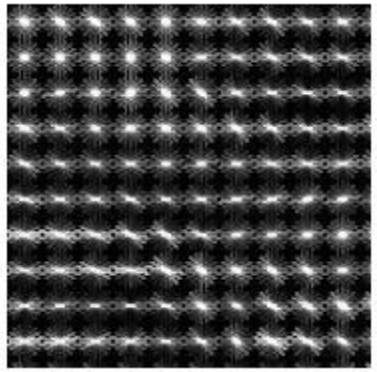


Image from A Zisserman

HoG as a CNN



Apply edge filters



8x8 average pooling with stride 8

LayerNorm to normalise histograms

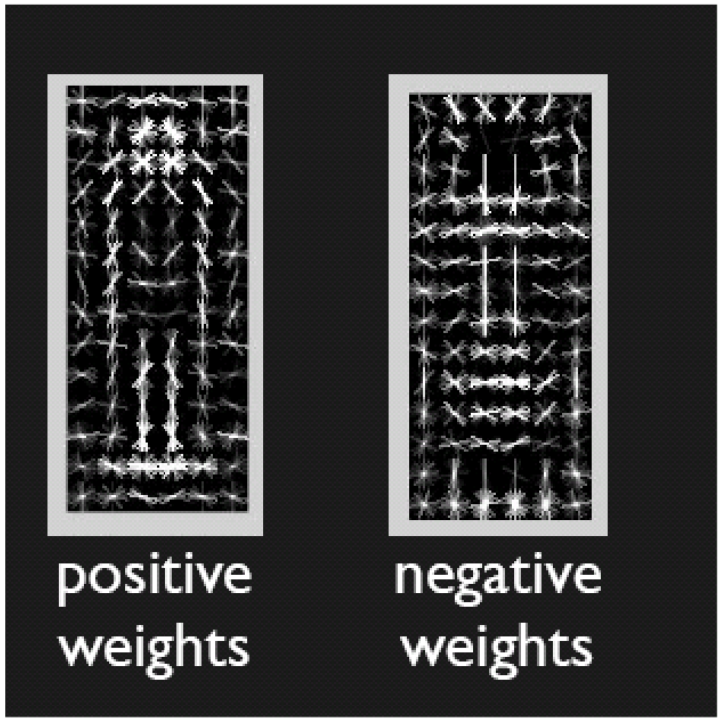
Flatten into a single feature vector

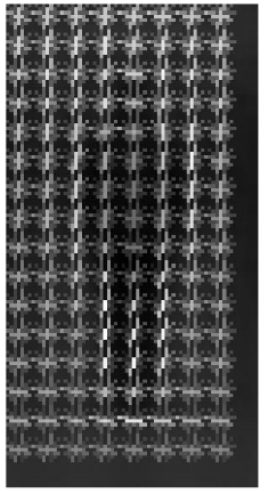
HoG with SVM

We can interpret the weights of the decision function by visualising their gradient histograms.

Positive weights vote for person, negative weights vote for background.

$$f(\mathbf{x}) = \mathbf{w}^\top \mathbf{x} + b$$





HOG averaged over positive training data

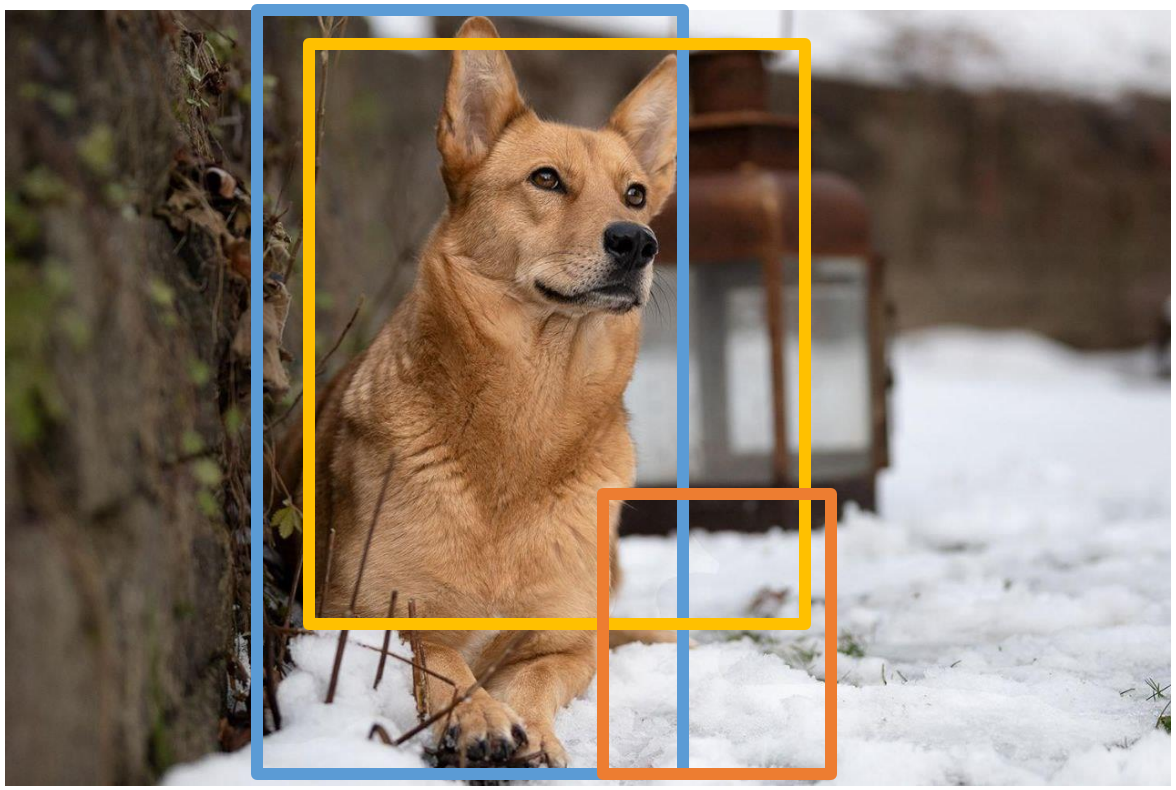
Slide from Deva Ramanan

HoG Summary

- Edges are useful to describe shapes.
- Spatial sub-sampling: invariant to small shifts.
- Grid structure allows spatial comparisons (in contrast to a single descriptor for the whole patch).
- Has been used for many objects afterwards!

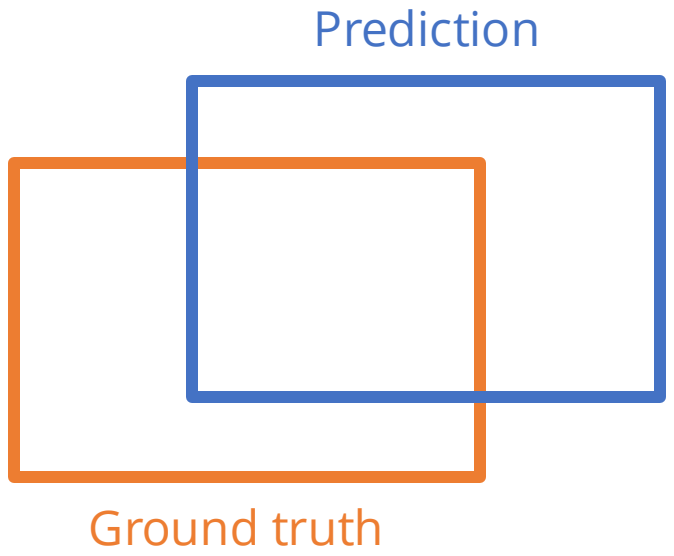
Measuring Performance

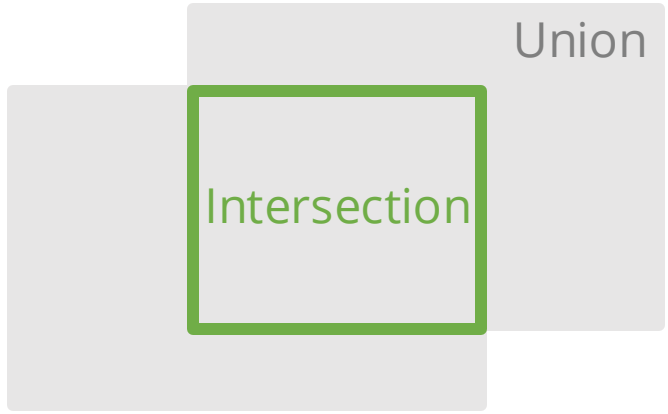
- Given the ground truth, how do we measure/score predictions?
- Continuous task: it is unlikely that the prediction will exactly match the ground truth.
- Yet, some predictions are better than others.



Intersection over Union

- The IoU metric is based on area overlap.
- $\text{IoU} = \frac{\text{Area}(\text{GT} \cap \text{Pred})}{\text{Area}(\text{GT} \cup \text{Pred})}$
- IoU is 1 for perfect overlap and 0 for no overlap.
- Often: threshold (e.g. 50% IoU) for counting as a positive match.





Intersection of Union

- High IoU requires tightly fitting predictions.
- Lower IoU allows loose fits.

- Examples

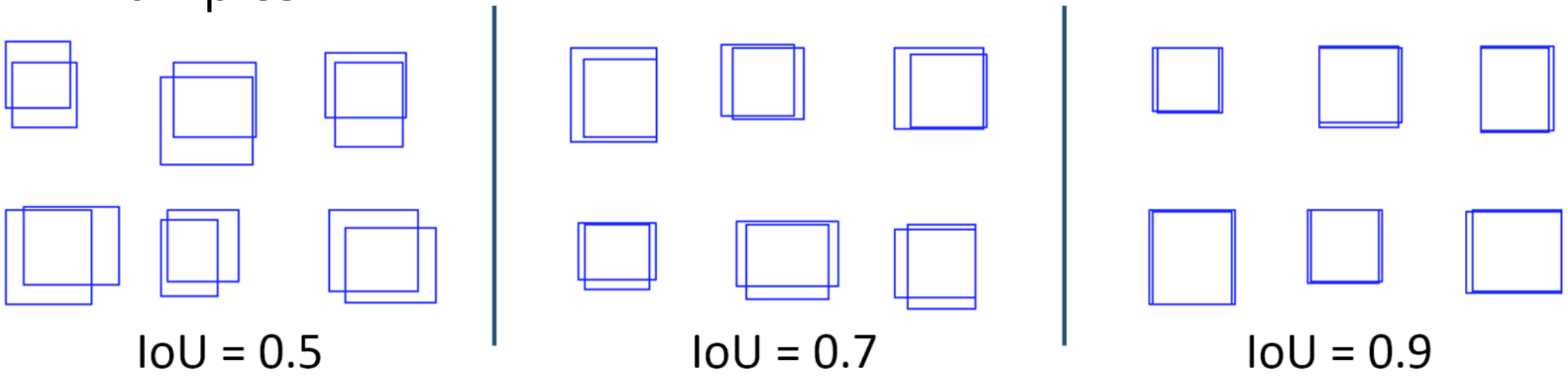
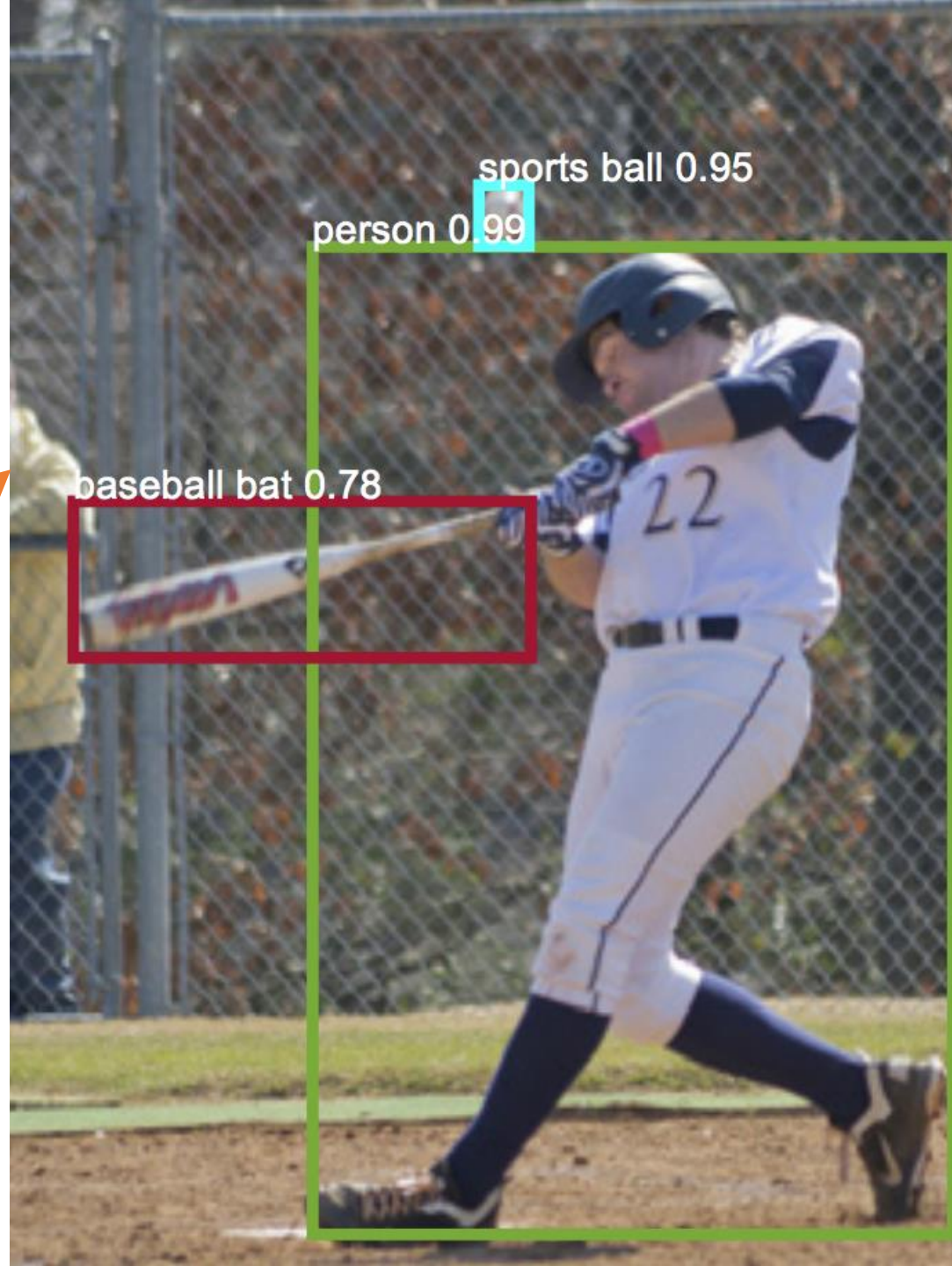


Image from Ross Girshick

False Negative

- False negatives are missed predictions.

- Common causes:
  - Occlusion
  - Truncation
  - Small size
  - …



False Negative
Person on the edge of the image was not detected.

False Positives

- Prediction that does not match a GT annotation
- Or, the GT annotation is already “covered” by a better-fitting prediction.

- Common cause: multiple detections for the same object.

Multiple Objects

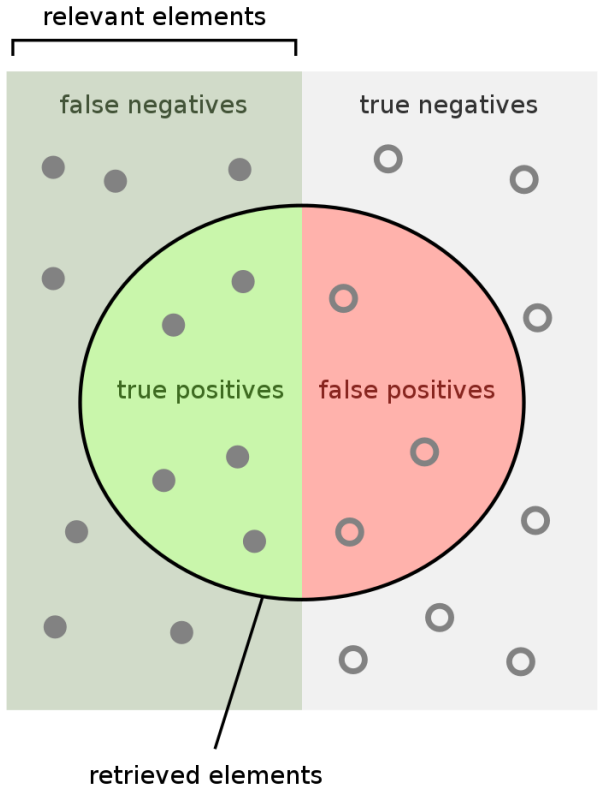
- During evaluation we are given a list of ground truth boxes and a list of predicted boxes.
- Per-class evaluation.
- For each GT box, find the best match in the predictions (IoU) and score it.
- Remove this box from predictions.
- Continue with next GT box.

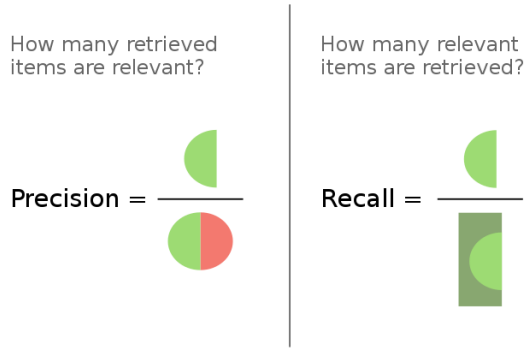
Recall: Precision and Recall

- Precision = TP/(TP+FP)
- Recall = TP/(TP+FN)

- Object detection: each predicted box has a confidence.

- Intuition: thresholding the confidence gives us more or less detections.

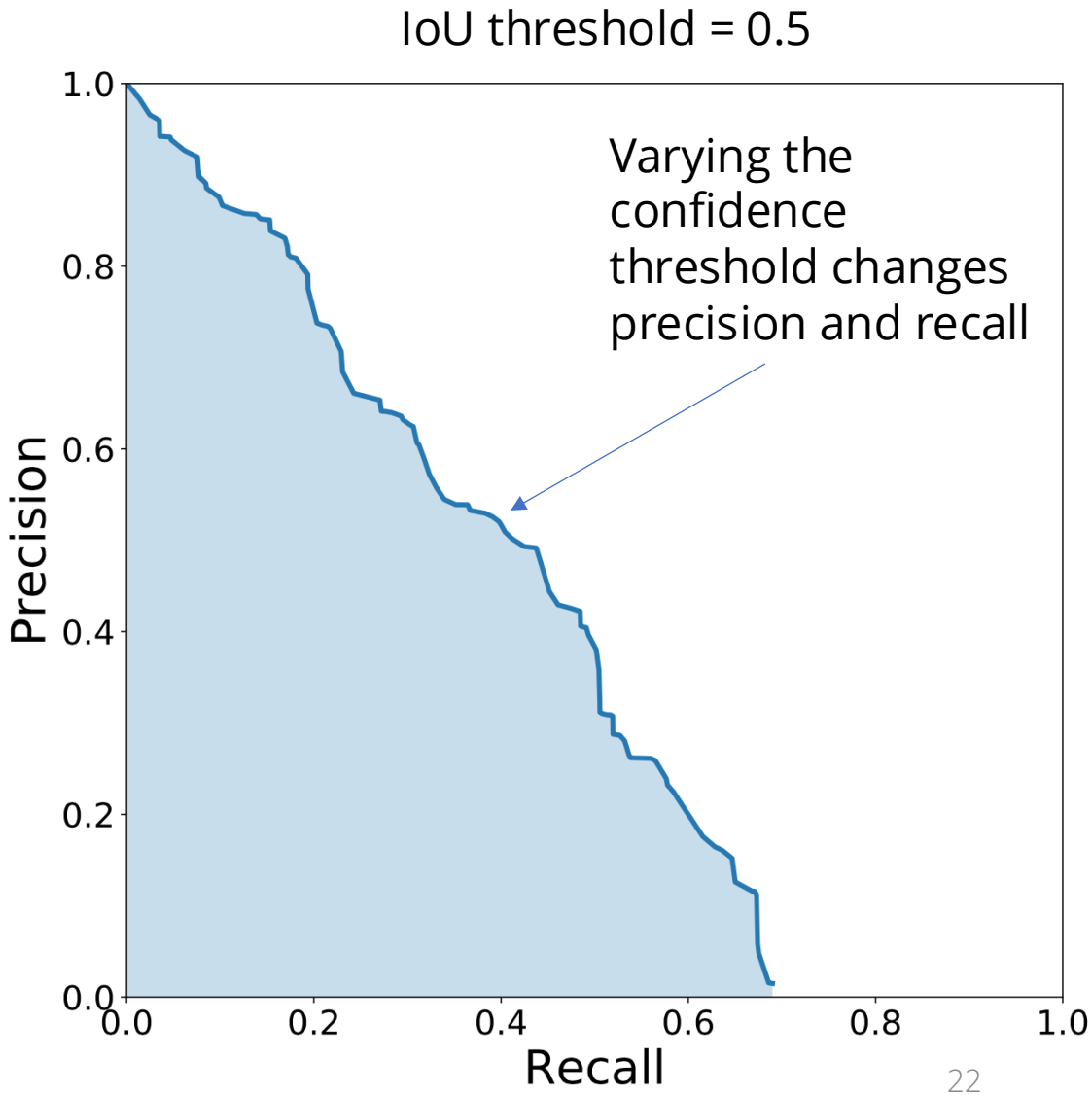




Precision Recall Curves

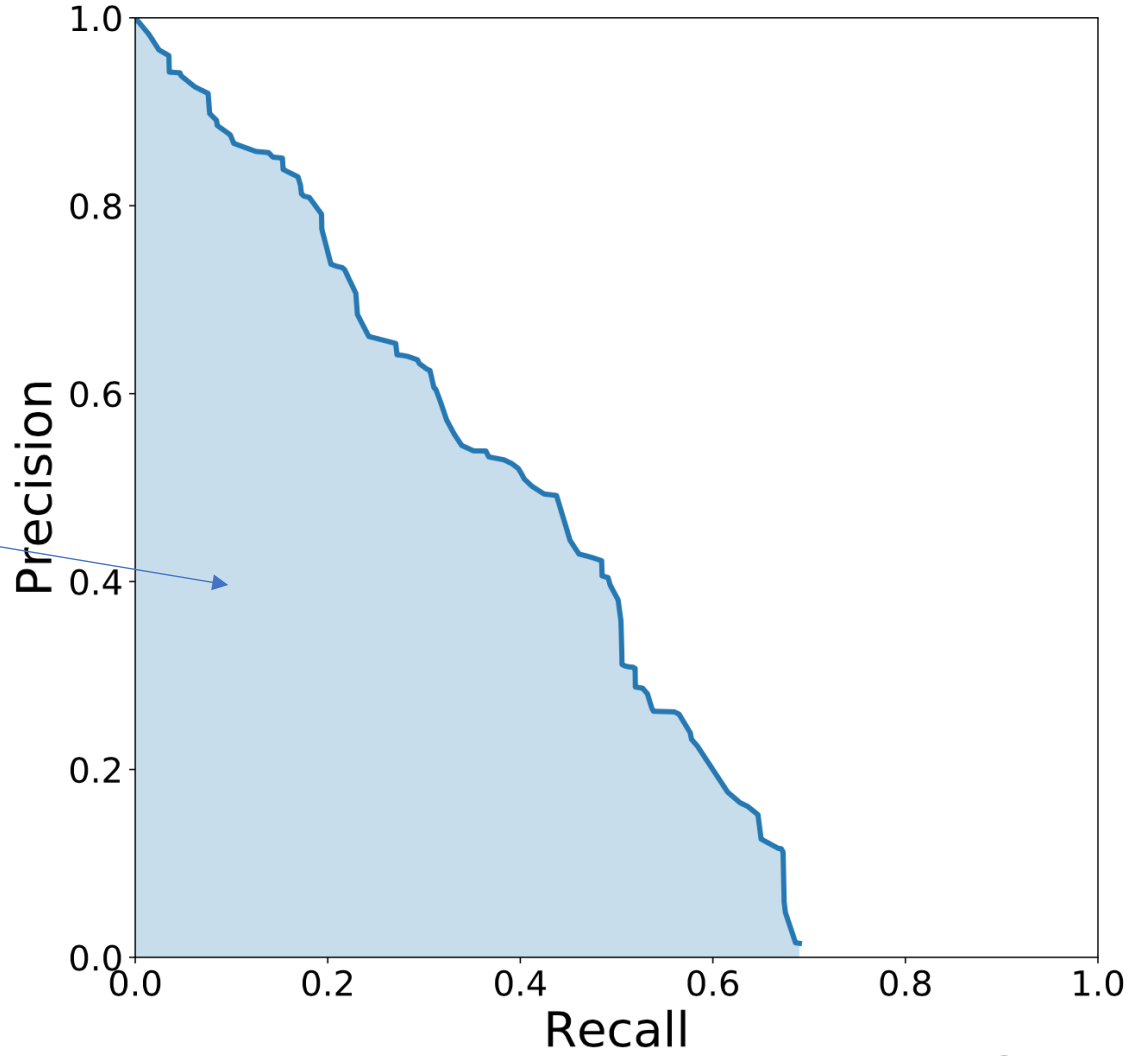
- Lower confidence threshold gives us more objects but many will be wrong (high recall, low precision).

- Higher confidence threshold yields less objects but they are more likely to be correct (low recall, high precision).



Average Precision (AP)

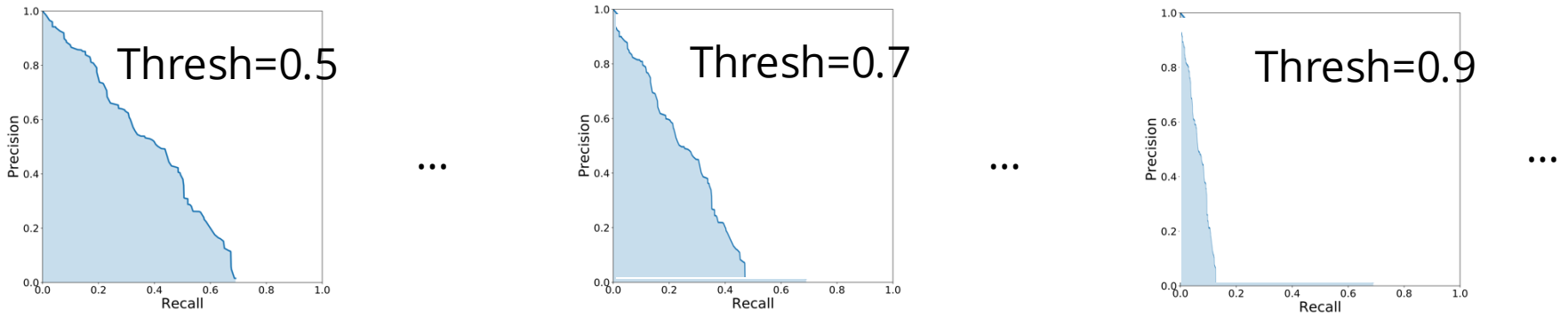
- We want: high precision and recall at all thresholds.
- AP metric: area under the precision recall curve.
- AP high: always good precision and recall.



Average Precision

- The metric still depends on the IoU threshold.
- Compute the average over many thresholds.

$$\text{AP(class)} = \frac{1}{\#\text{thresholds}} \sum_{\text{iou} \in \text{thresholds}} \text{AP(class,iou)}$$

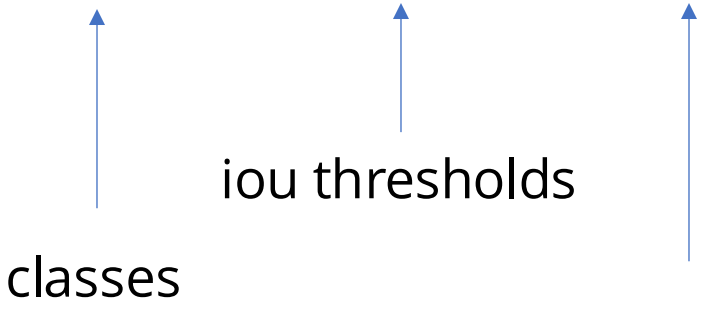


Overall Precision

- To combine AP from different classes, we average again.

$$\text{AP} = \frac{1}{\#\text{classes}} \sum_{\text{class} \in \text{classes}} \text{AP}(\text{class})$$

- AP is an *average, average, average* precision.

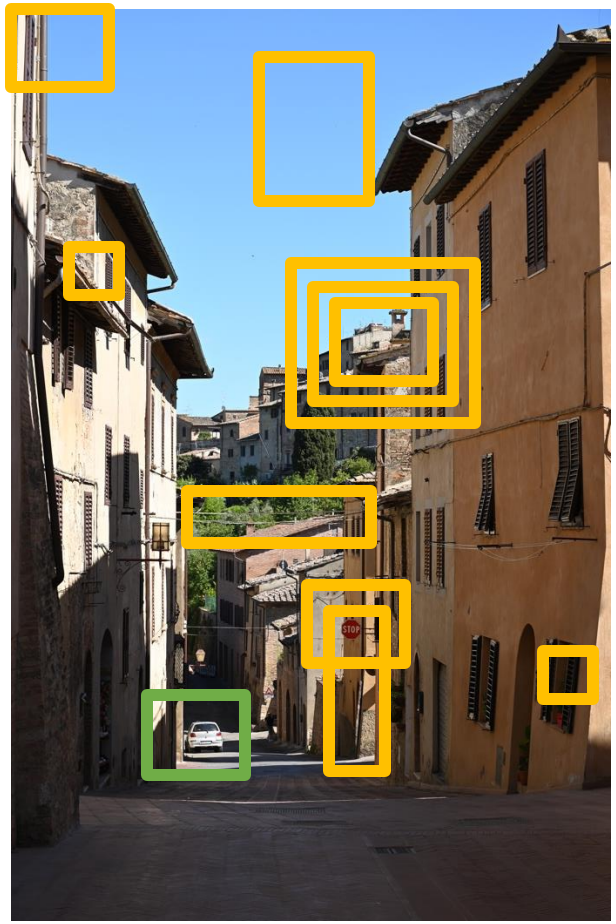


Detection Evaluation

- Task: given an image, predict objects as $(\hat{x}, \hat{y}, \hat{w}, \hat{h}, \hat{c})$. $\hat{c} \in \mathbb{R}^C$
- Evaluation: for each image, class c, IoU threshold t_{iou}
  - Set of predictions $(\hat{x}_i, \hat{y}_i, \hat{w}_i, \hat{h}_i, \hat{c}_{c,i})$ (class confidence $\hat{c}_{c,i}$)
  - For each confidence threshold t_{conf} :
    - Ignore any boxes with $\hat{c}_{c,i} < t_{\text{conf}}$.
    - For each GT annotation (x_j, y_j, h_j, w_j):
      - find highest IoU prediction
      - If $\text{IoU}\left(\left(\hat{x}_i, \hat{y}_i, \hat{w}_i, \hat{h}_i\right), \left(x_j, y_j, h_j, w_j\right)\right) \geq t_{\text{iou}}$: TP (remove from predictions) otherwise FN
    - Remaining predictions: FP
  - This gives us the PR-curve. Compute the area under the curve.
- Compute AP: average over classes and IoU thresholds of area under the PR-curve.

Object Detection as Classification

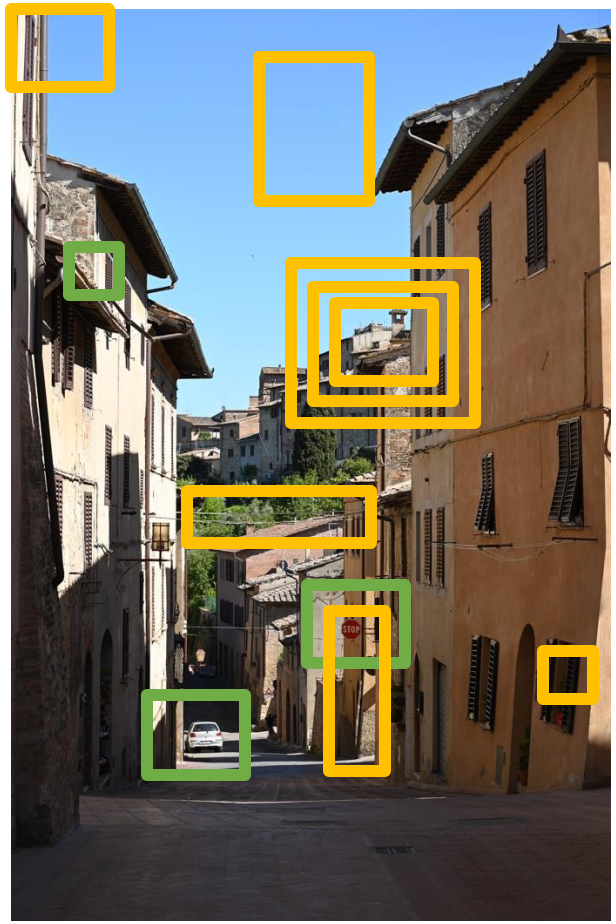
- Inherently unbalanced: many more negatives than positives.
- Background class is visually much more complex than the object.
- Good performance: low false positive rate.



Bootstrapping or Self-Training

1. Create a training dataset of positive and negative patches.
2. Train classifier.
3. Detect objects in *training* data.
4. Add false positives to training data.
5. Goto 2.

- Automatically includes difficult examples into the training set.
- Also called: hard negative mining.



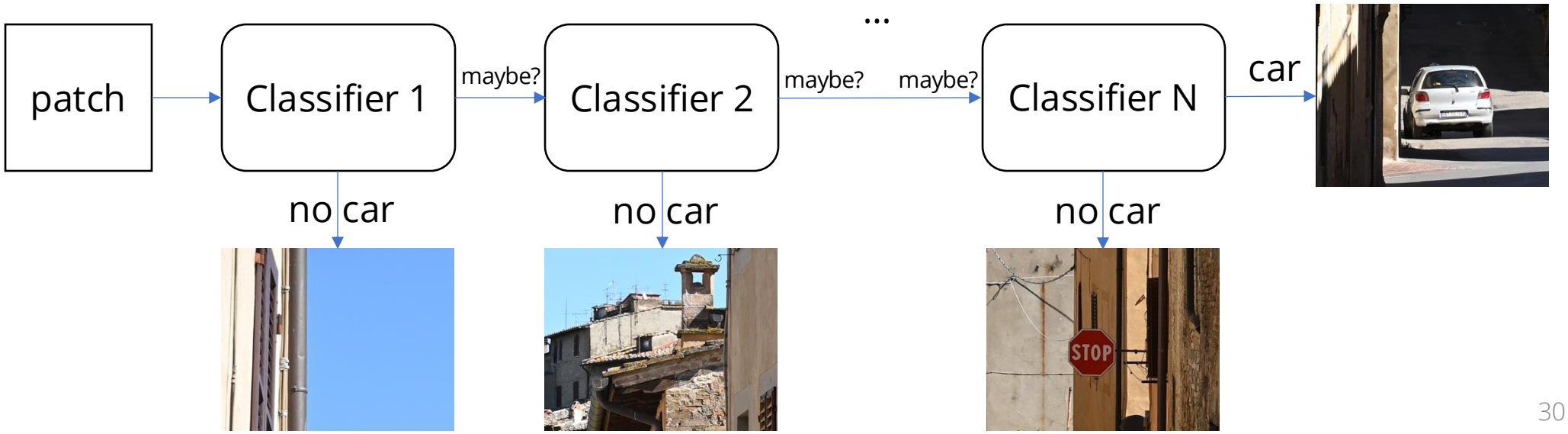
Non-Maximum Suppression

- Close-by patches look similar

- Often: multiple detections for the same object.
- For overlapping boxes: choose the one with highest confidence and remove/down-weigh others.

Cascaded Classifiers

- Sliding window detection is slow.
- Many windows are *clearly* not the object.
- Instead of one slow/big classifier, build a sequence to quickly throw out true negatives.



Cascaded Classifiers

- Early classifiers can have low precision but high recall: eliminate easy negatives.
- Cascade from fast, simple classifiers to slower, complex classifiers with lower false positive rate.
- Viola & Jones, 2001: Face detection

Slide adapted from A Zisserman

Object Proposals

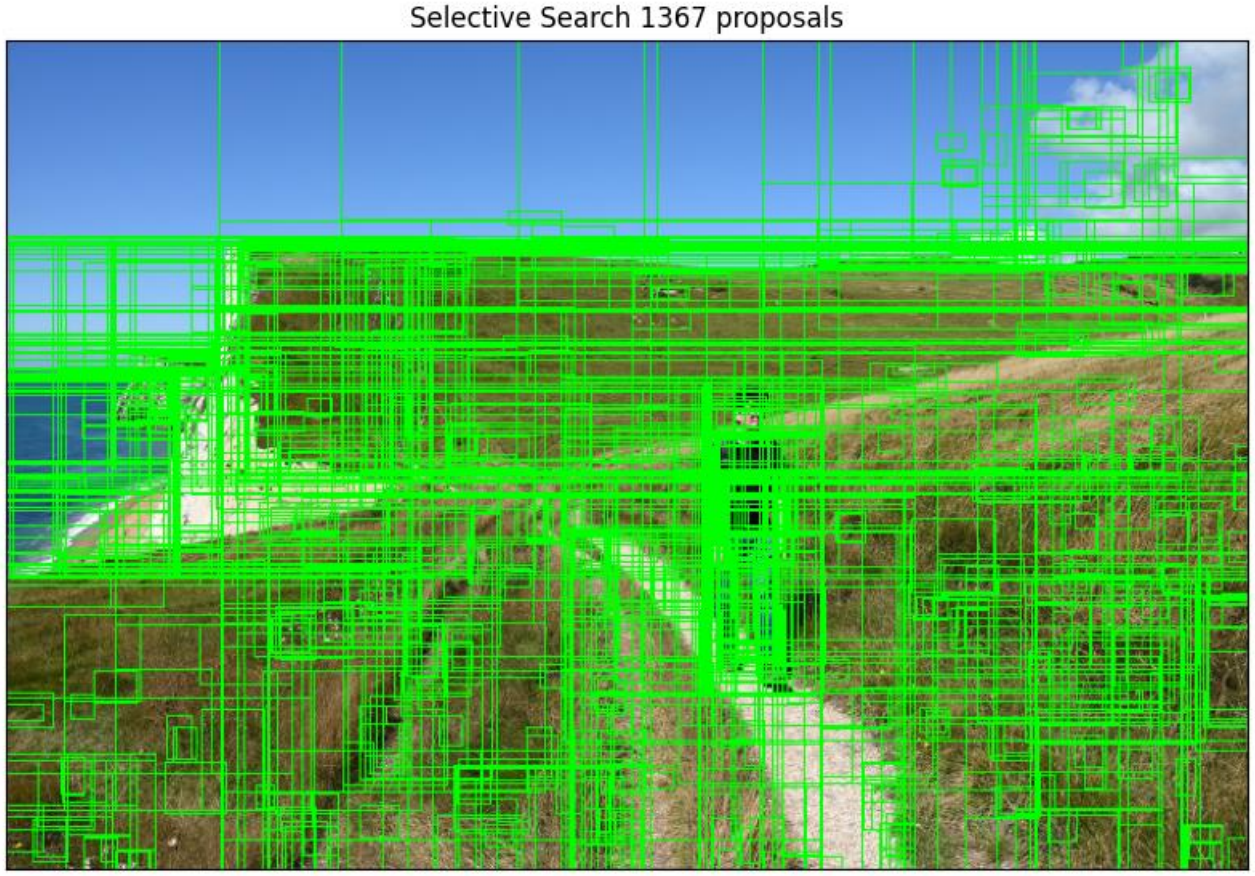
- Same idea: restrict the number of patches/windows to a better subset.
- Algorithm to suggest proposals for boxes.
- Needs high recall but can be low precision.
- Aim to cover all the objects in the image with a small number of proposals, e.g. 100-1000 per image.

Selective Search

Uijlings, van de Sande, Gevers, and Smeulders, IJCV 2013

- hierarchical segmentation
  - colour uniformity
  - image edges

- ca. 2000 regions / image.
- > 95% probability of hitting any relevant object in the image



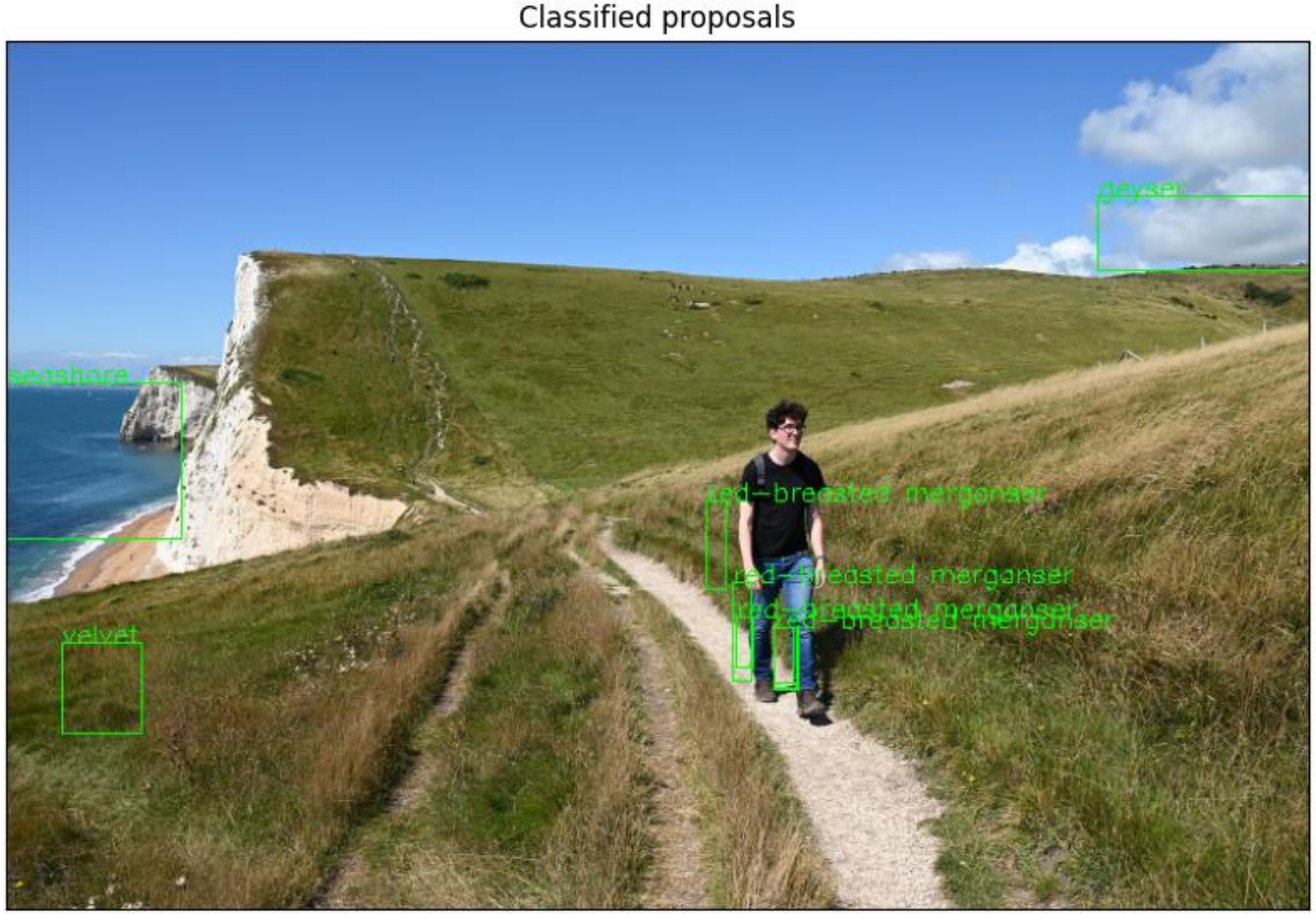
Object Detection with CNNs

Let’s build a simple detector:

1. Compute proposals with Selective Search.
2. For each proposal: feed to CNN classifier (ImageNet trained).
3. If class probability > 80%: detection.
4. Optional: non-maximum suppression.

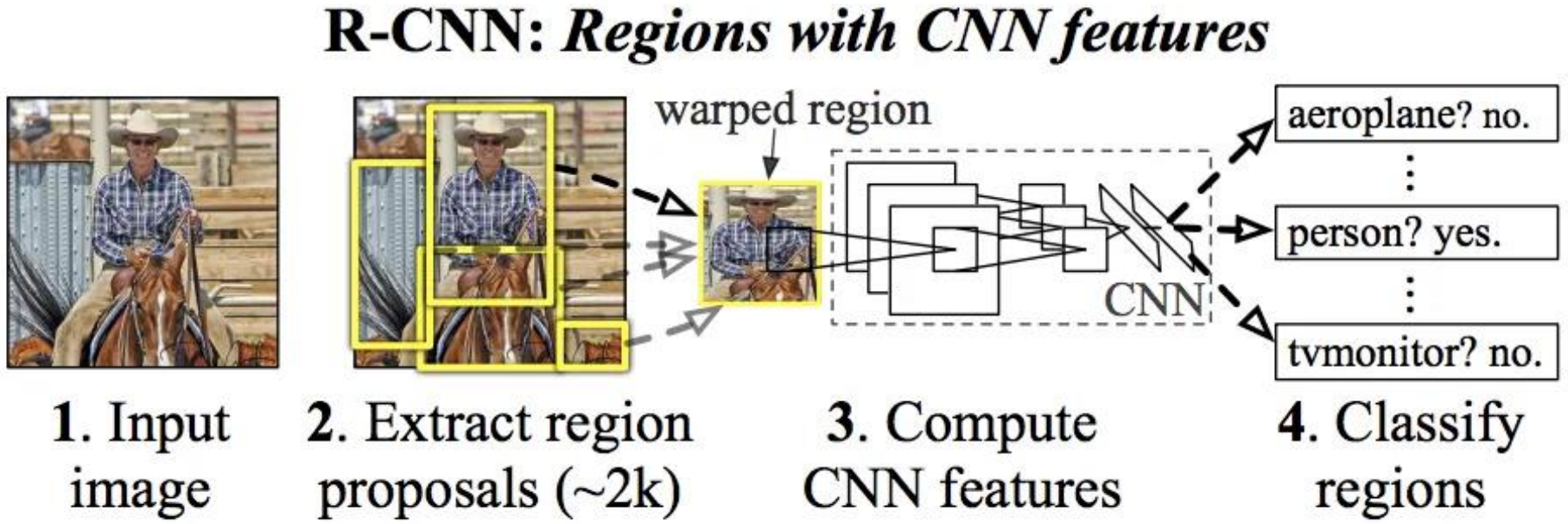
Simple CNN Object Detection

- ImageNet trained model is not trained to predict background.
- Simple thresholding confidences is not enough.
- Works to a certain degree: “seashore”

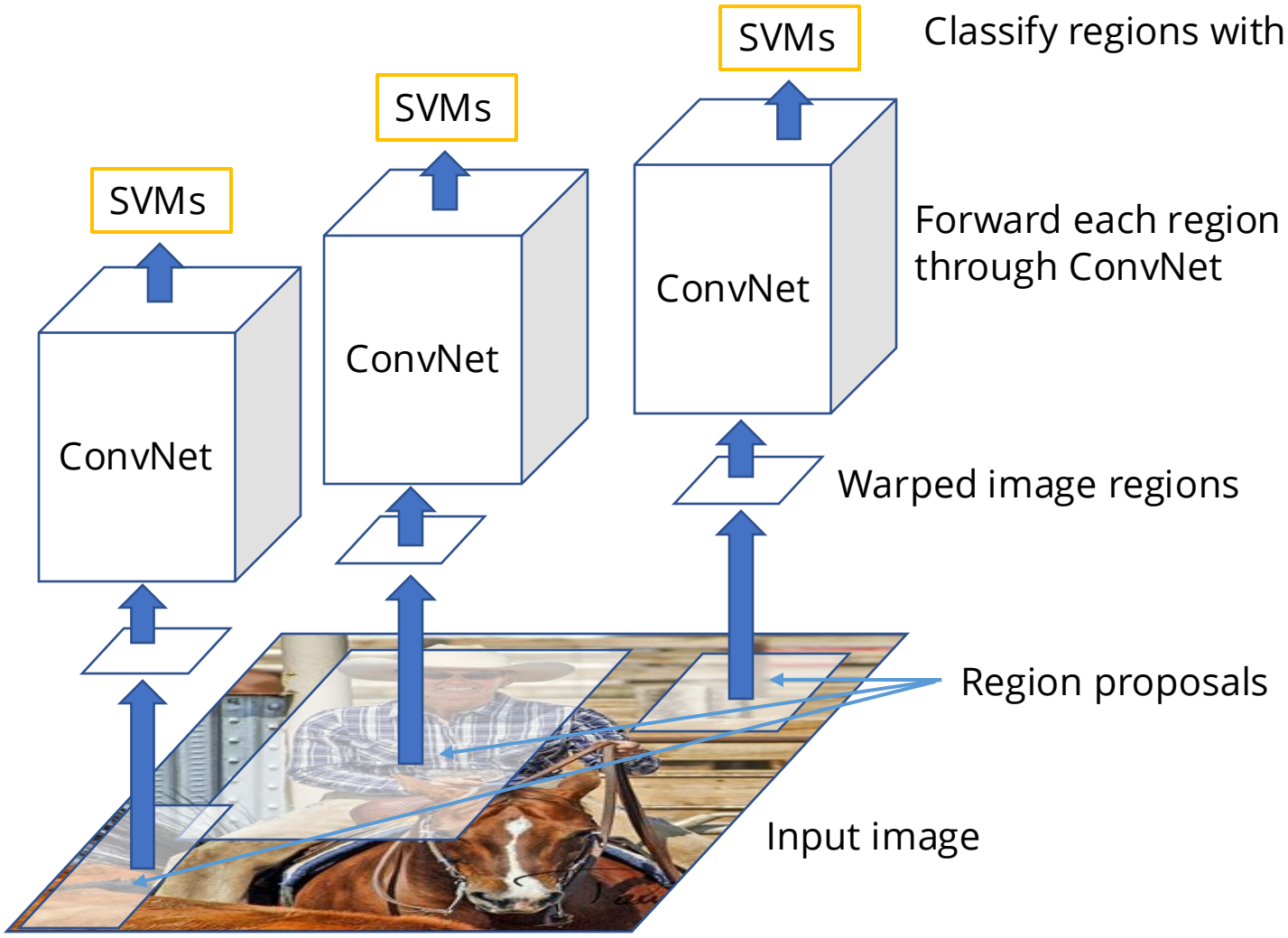


R-CNN

Girshick, Donahue, Darrel, Malik, 2013



R-CNN



Regions: ~2000 Selective Search proposals

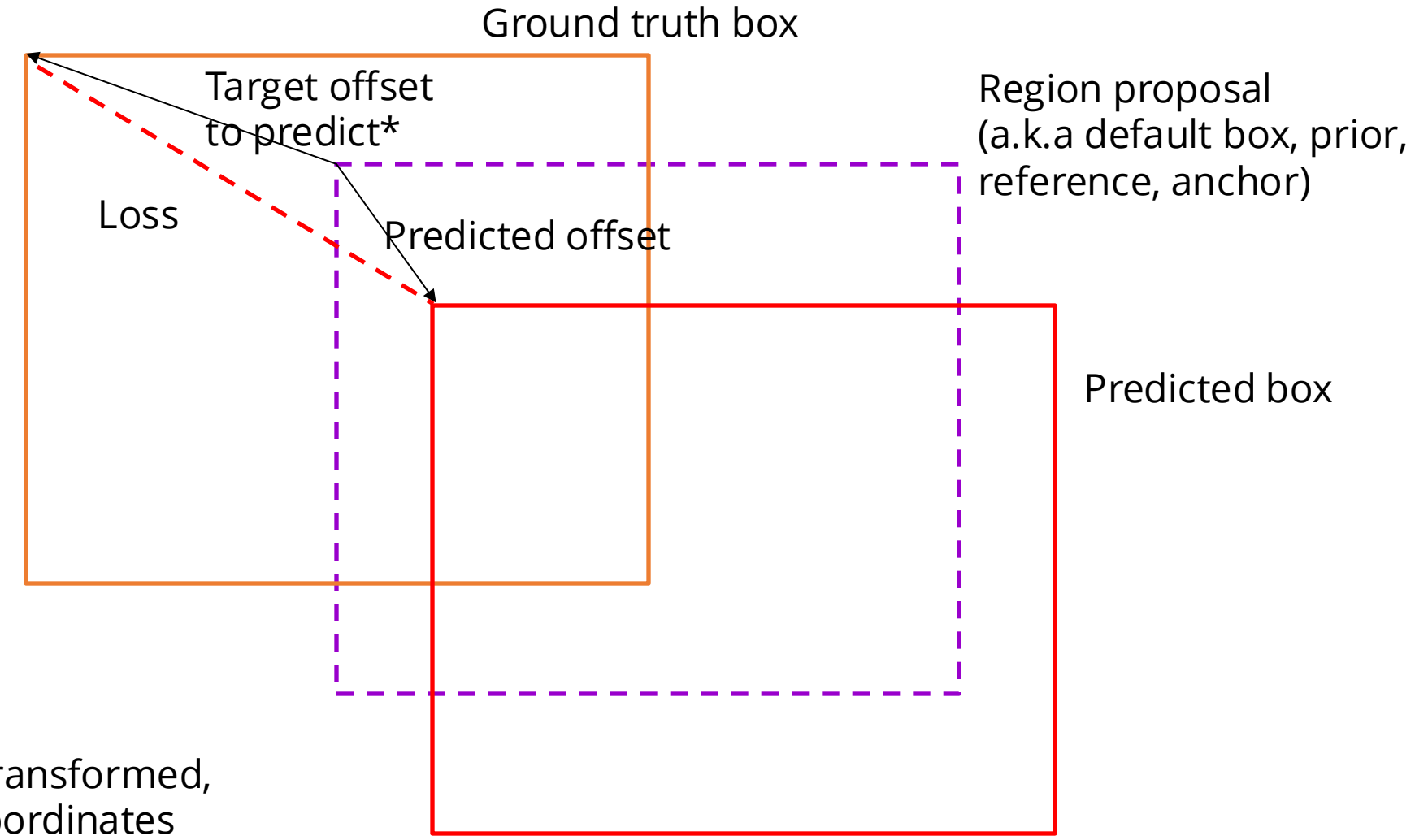
Network: AlexNet *pre-trained* on ImageNet (1000 classes), *fine-tuned* on PASCAL (21 classes)

Final detector: warp proposal regions, extract fc7 network activations (4096 dimensions), classify with linear SVM

Bounding box regression to refine box locations

Source: R. Girshick

Bounding box regression



*Typically in transformed, normalized coordinates

R-CNN

Pros

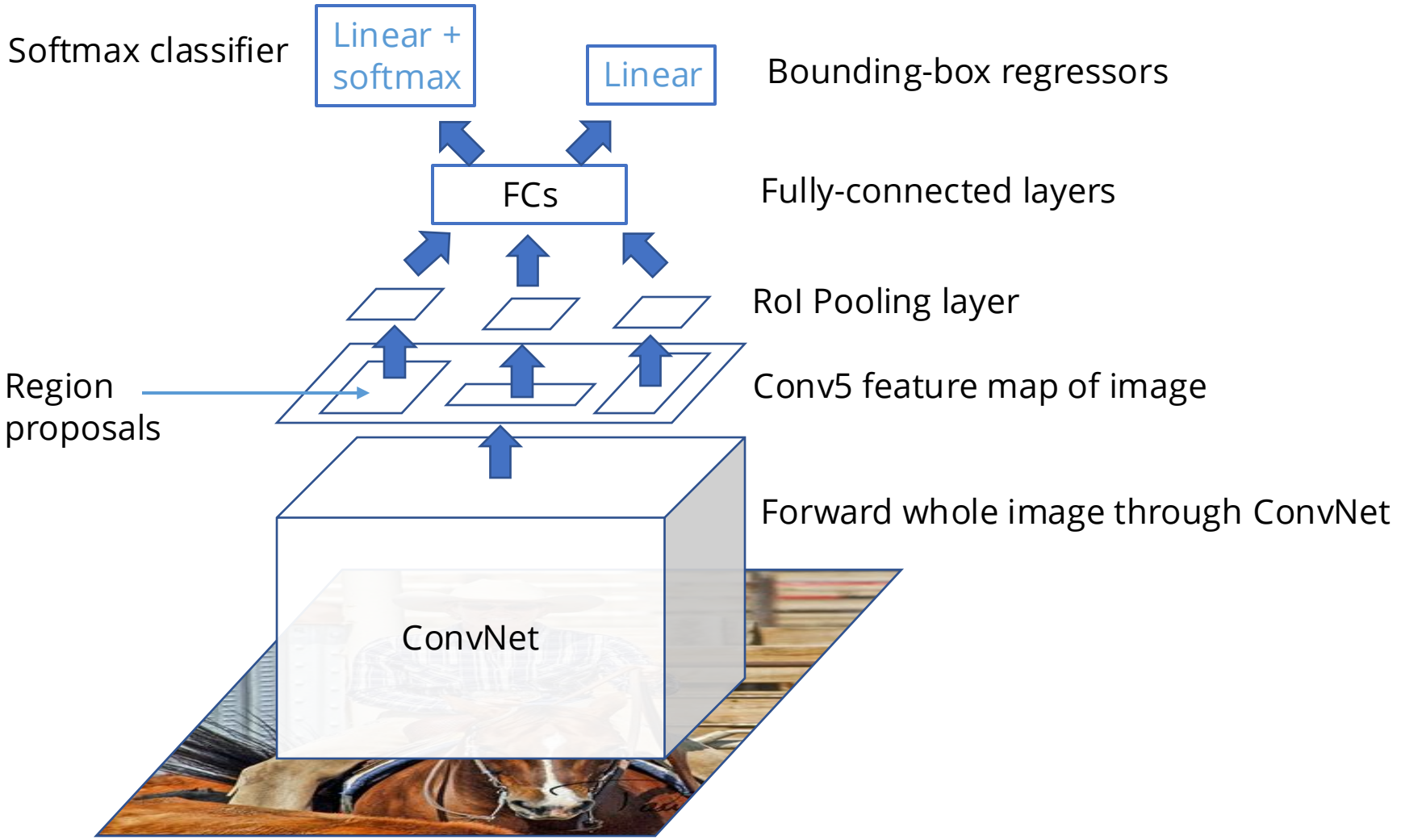
- Much more accurate than previous approaches!
- Any deep architecture can immediately be “plugged in”

Cons

- Not a single end-to-end system
  - Fine-tune network with softmax classifier (log loss)
  - Train post-hoc linear SVMs (hinge loss)
  - Train post-hoc bounding-box regressions (least squares)
- Training was slow (84h), took up a lot of storage
  - 2000 CNN passes per image
- Inference (detection) was slow (47s / image with VGG16)

Source: S. Lazebnik

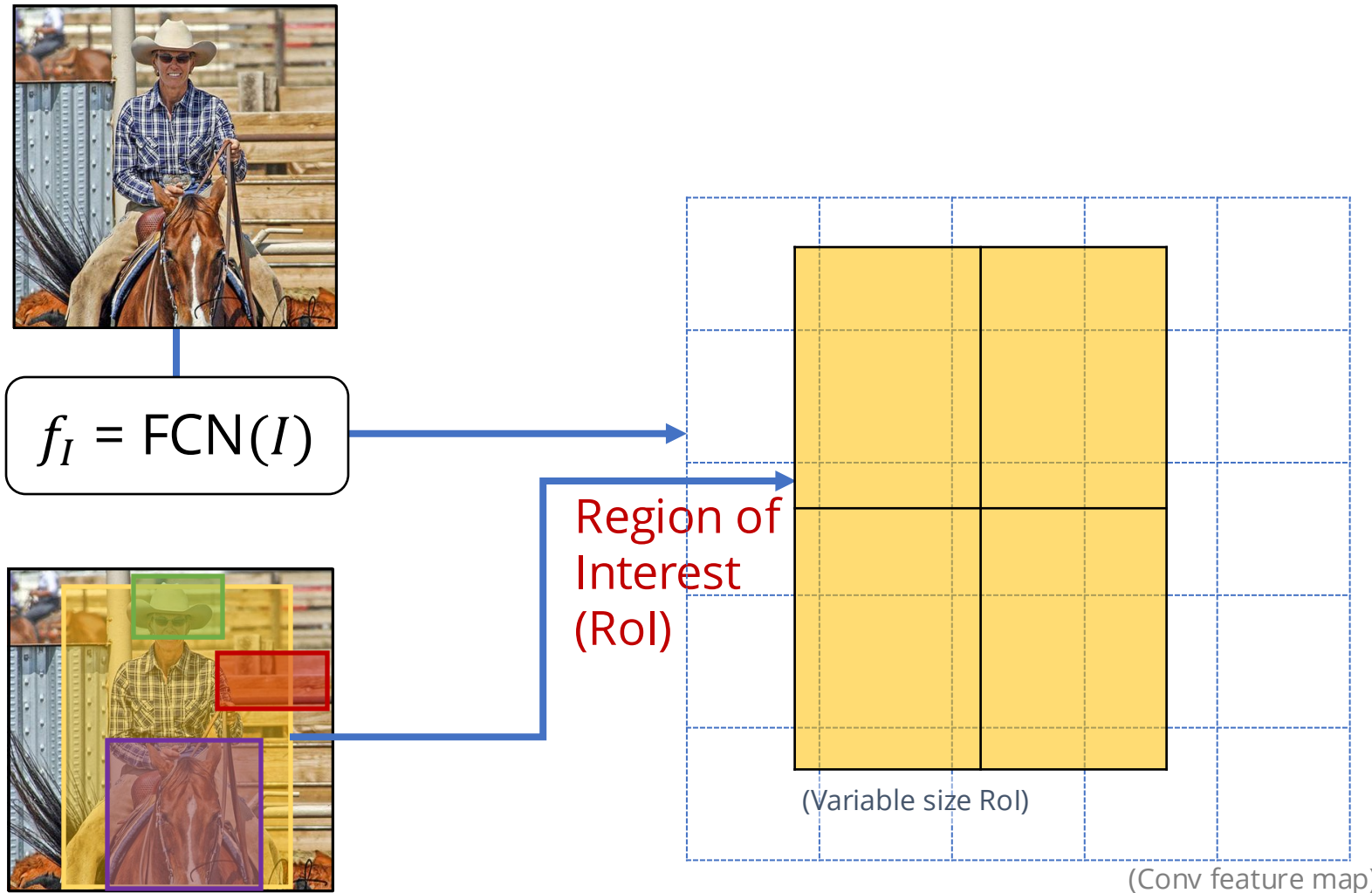
Fast R-CNN



R. Girshick, Fast R-CNN, ICCV 2015

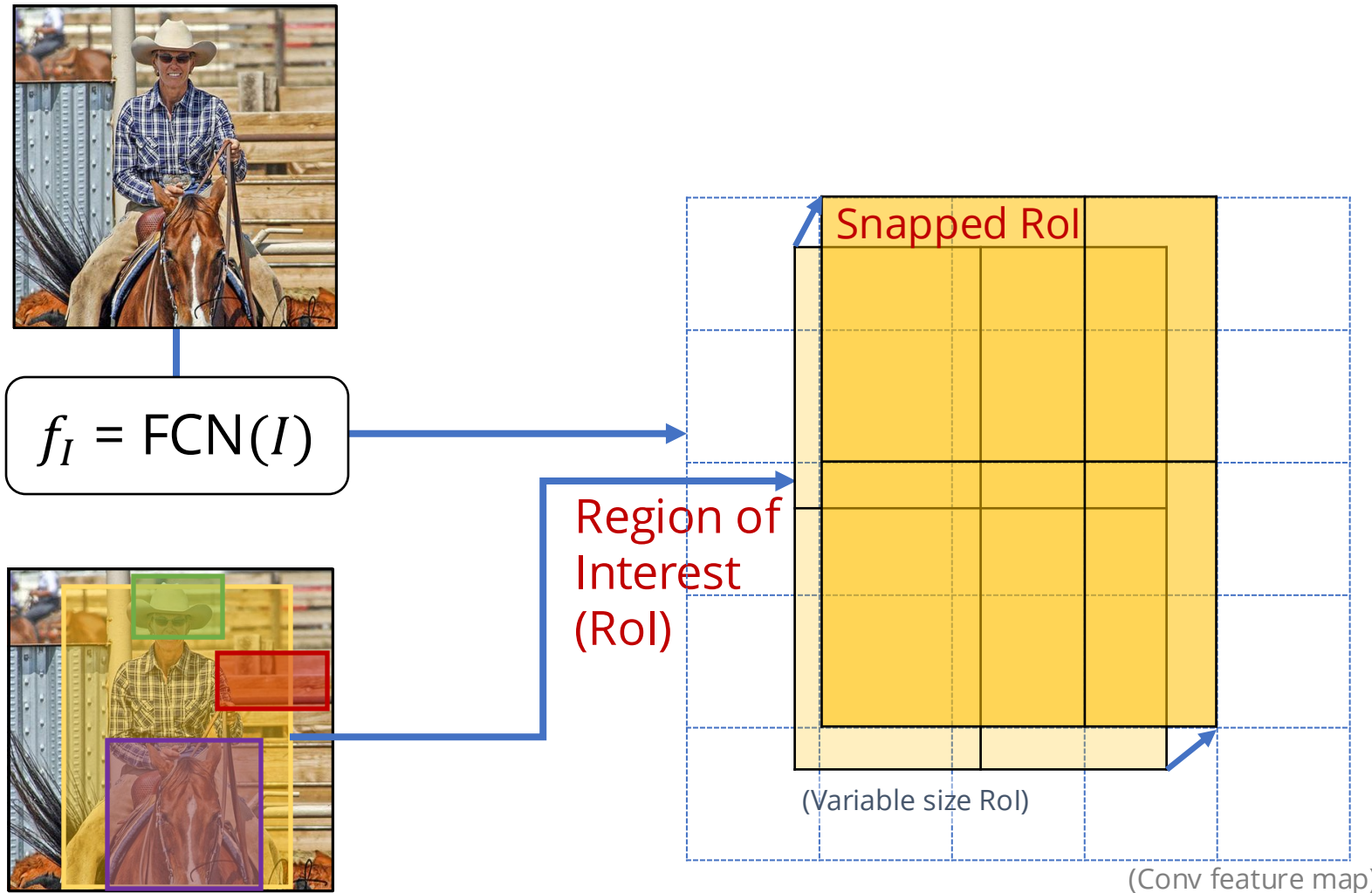
Source: R. Girshick

RoIPool Operation (on each Proposal)



Source: R. Girshick

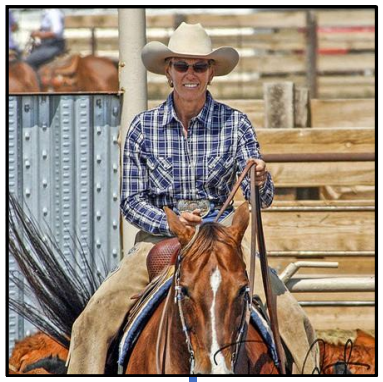
RoIPool Operation (on each Proposal)



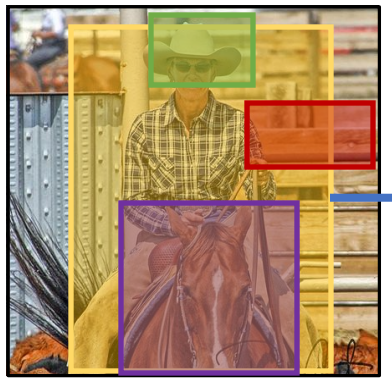
Source: R. Girshick

RoIPool Operation (on each Proposal)

Transform arbitrary size proposal into a fixed-dimensional representation (e.g., 2x2)



f_I = FCN(I)



Region of Interest (RoI)

Snapped RoI

(Variable size RoI)

(Conv feature map)

RoIPool transform

(Fixed dimensional representation)

MLP

Feature value is max over input cells

Source: R. Girshick

RoI Pooling

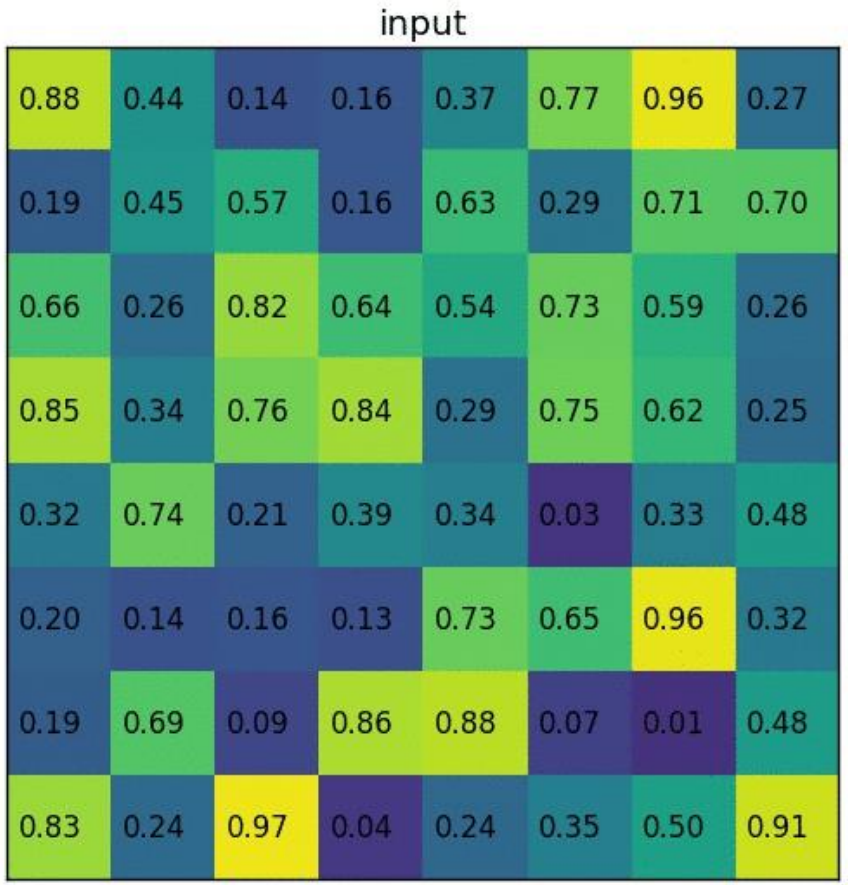


Image source

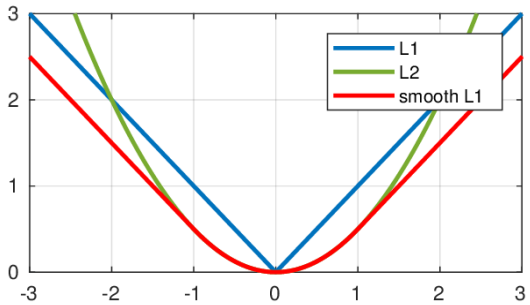
Multi-task loss

Loss for ground truth class y, predicted class probabilities $P(y)$, ground truth box b, and predicted box $\hat{b}$:

$$L(y, P, b, \hat{b}) = \underbrace{-\log P(y)}_{\text{softmax loss}} + \lambda \mathbb{I}[y \geq 1] \underbrace{L_{\text{reg}}(b, \hat{b})}_{\text{regression loss}}$$

Regression loss: *smooth* L_1 *loss* on top of log space offsets relative to proposal

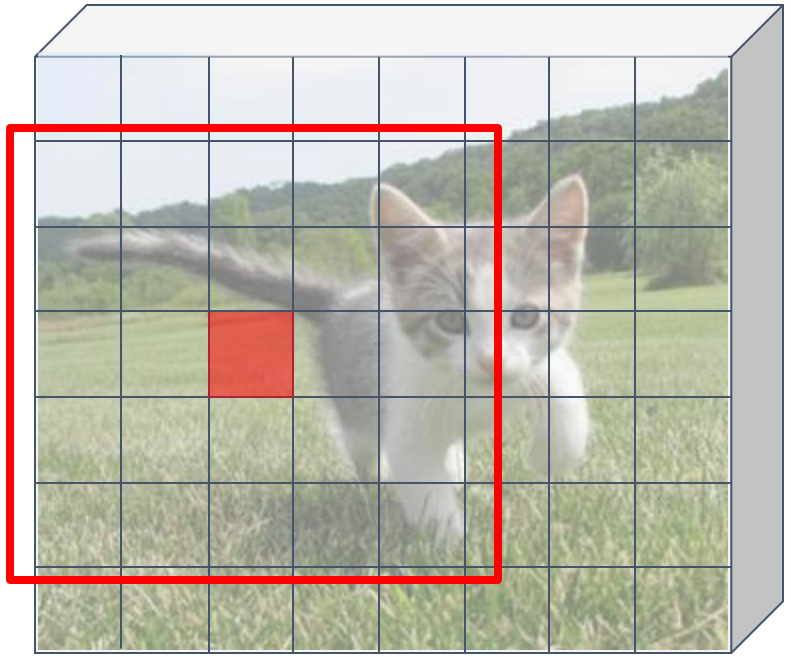
$$L_{\text{reg}}(b, \hat{b}) = \sum_{i=\{x,y,w,h\}} \text{smooth}_{L_1}(b_i - \hat{b}_i)$$



$$\text{smooth}_{L_1}(x) = \begin{cases} 0.5x^2 & \text{if } |x| < 1 \\ |x| - 0.5 & \text{otherwise} \end{cases}$$

Region proposal network (RPN)

Idea: put an “anchor box” of fixed size over each position in the feature map and try to predict whether this box is likely to contain an object

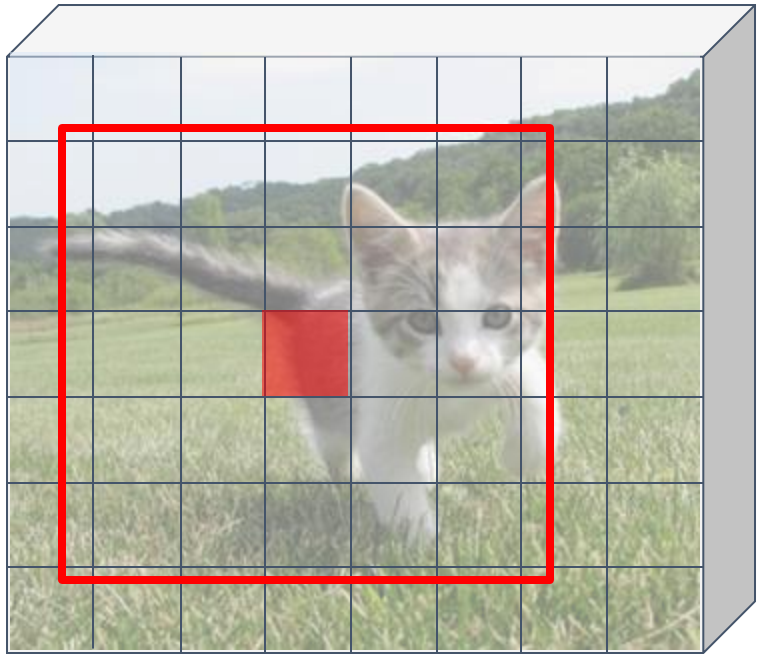


Anchor is an object?

Figure source: J. Johnson

Region proposal network (RPN)

Idea: put an “anchor box” of fixed size over each position in the feature map and try to predict whether this box is likely to contain an object



Anchor is an object?

Figure source: J. Johnson

Region proposal network (RPN)

Idea: put an “anchor box” of fixed size over each position in the feature map and try to predict whether this box is likely to contain an object

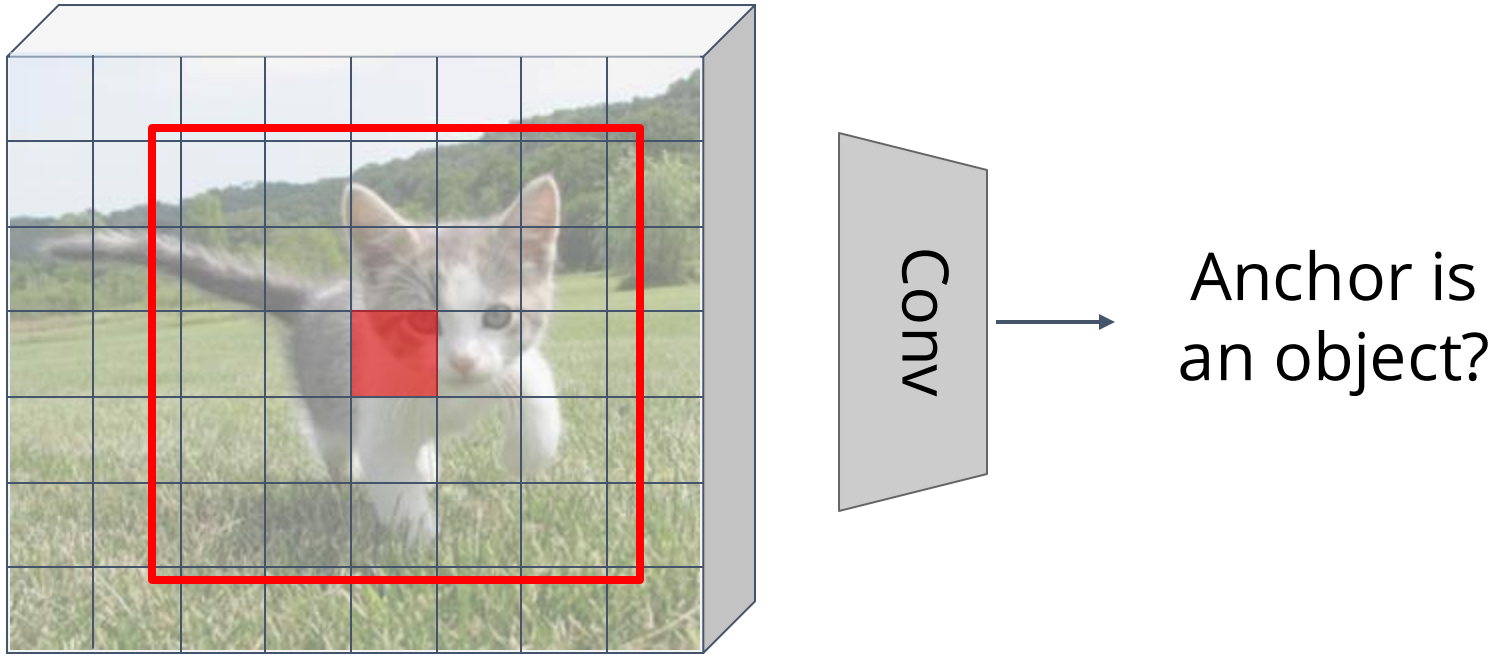


Figure source: J. Johnson

Region proposal network (RPN)

Introduce anchor boxes at multiple scales and aspect ratios to handle a wider range of object sizes and shapes

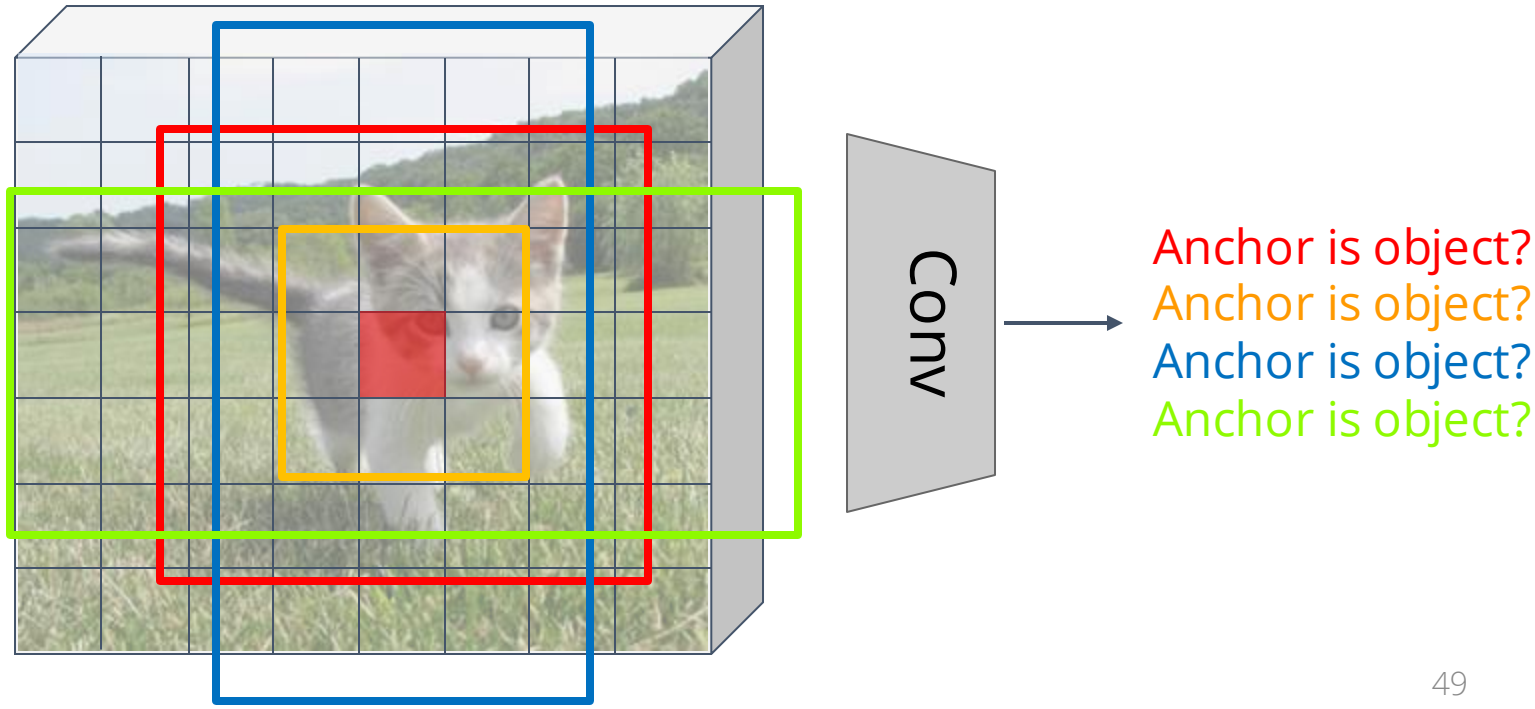
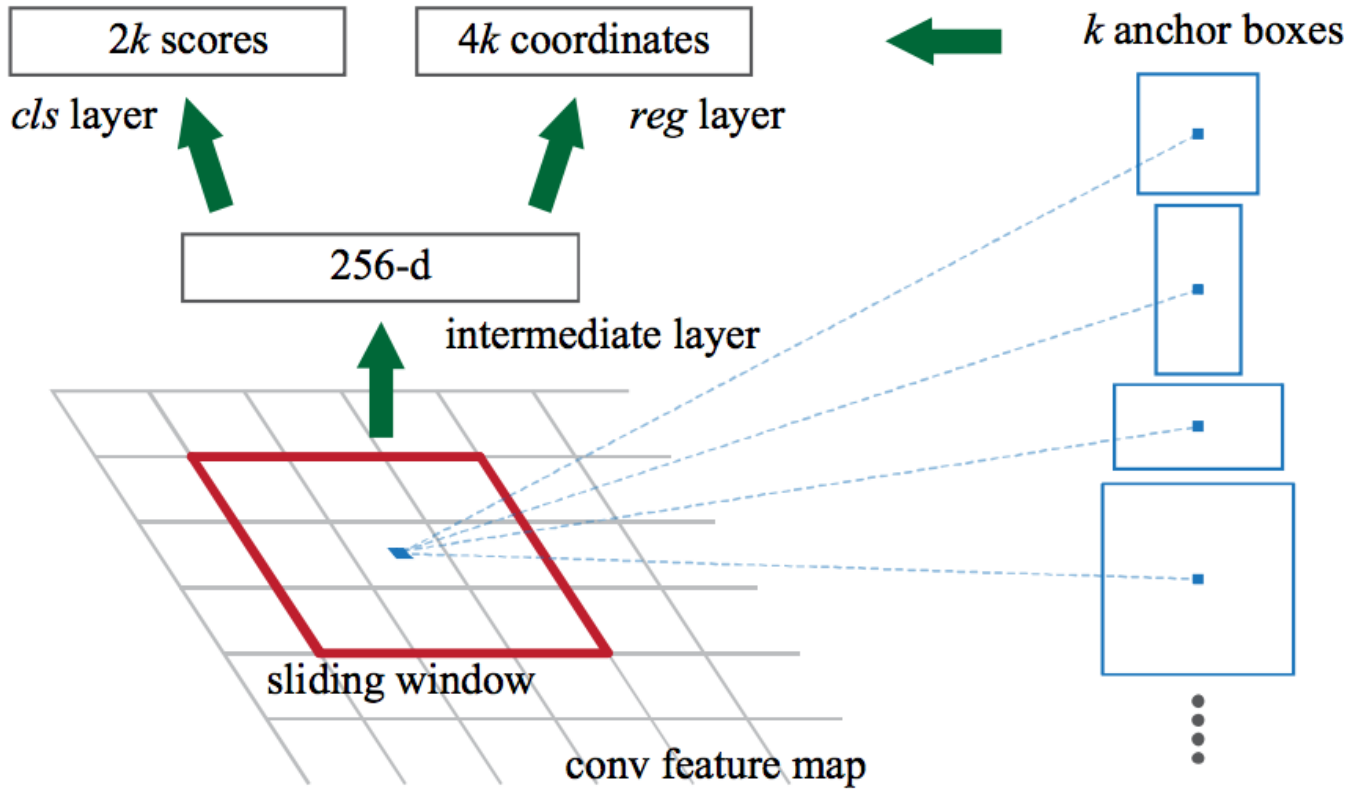


Figure source: J. Johnson

Faster R-CNN RPN design

Ren, He, Grishick, Sun, 2015



Two-Stage Detectors

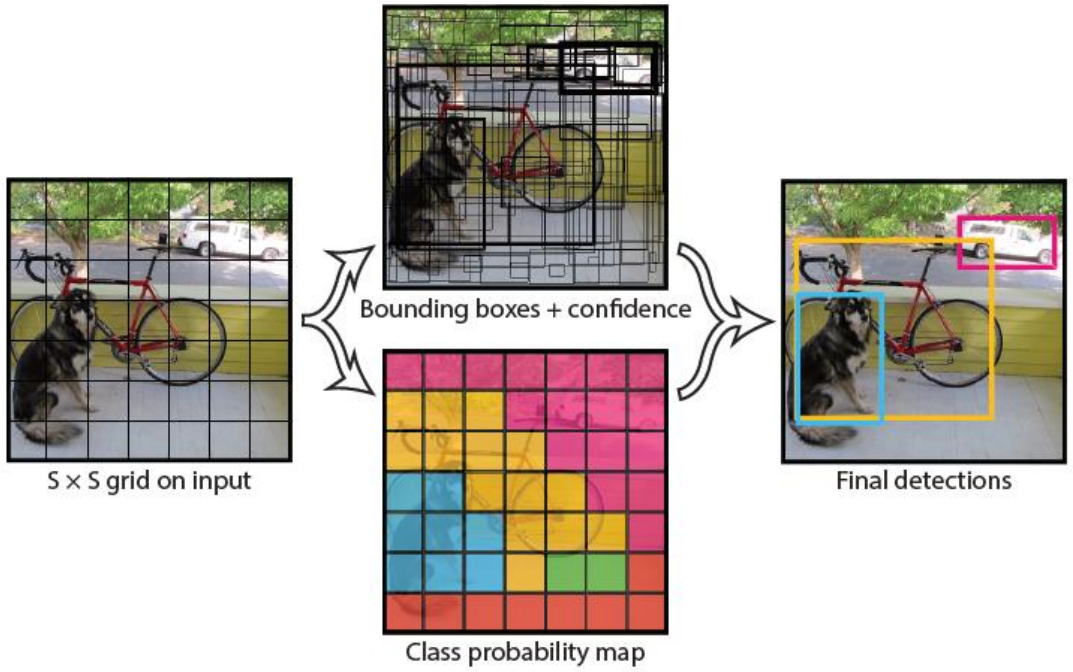
So far: two-stage detectors.

1. Create proposals
2. Classify & update/move proposals

Can we do it in one step?

Single Stage Detector: YOLO

Divide the image into a coarse grid and directly predict class label and a few candidate boxes for each grid cell



J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, You Only Look Once: Unified, Real-Time Object Detection, CVPR 2016

YOLO – Training Loss

$$\lambda_{\text{coord}} \sum_{i=0}^{S^2} \sum_{j=0}^{B} \mathbb{1}_{ij}^{\text{obj}} \left[(x_i - \hat{x}_i)^2 + (y_i - \hat{y}_i)^2 \right]$$

$$+ \lambda_{\text{coord}} \sum_{i=0}^{S^2} \sum_{j=0}^{B} \mathbb{1}_{ij}^{\text{obj}} \left[\left(\sqrt{w_i} - \sqrt{\hat{w}_i} \right)^2 + \left(\sqrt{h_i} - \sqrt{\hat{h}_i} \right)^2 \right]$$

Regression

$$+ \sum_{i=0}^{S^2} \sum_{j=0}^{B} \mathbb{1}_{ij}^{\text{obj}} \left(C_i - \hat{C}_i \right)^2$$

$$+ \lambda_{\text{noobj}} \sum_{i=0}^{S^2} \sum_{j=0}^{B} \mathbb{1}_{ij}^{\text{noobj}} \left(C_i - \hat{C}_i \right)^2$$

Object/no object confidence

$$+ \sum_{i=0}^{S^2} \mathbb{1}_{i}^{\text{obj}} \sum_{c \in \text{classes}} \left(p_i(c) - \hat{p}_i(c) \right)^2$$

Class prediction

Source: S. Lazebnik

YOLO – Training Loss

Cell i contains object, predictor j is responsible for it

$$\lambda_{\text{coord}} \sum_{i=0}^{S^2} \sum_{j=0}^{B} \mathbb{1}_{ij}^{\text{obj}} \left[(x_i - \hat{x}_i)^2 + (y_i - \hat{y}_i)^2 \right]$$

$$+ \lambda_{\text{coord}} \sum_{i=0}^{S^2} \sum_{j=0}^{B} \mathbb{1}_{ij}^{\text{obj}} \left[\left(\sqrt{w_i} - \sqrt{\hat{w}_i} \right)^2 + \left(\sqrt{h_i} - \sqrt{\hat{h}_i} \right)^2 \right]$$

Small deviations matter less for larger boxes than for smaller boxes

$$+ \sum_{i=0}^{S^2} \sum_{j=0}^{B} \mathbb{1}_{ij}^{\text{obj}} \left(C_i - \hat{C}_i \right)^2$$

Confidence for object

$$+ \lambda_{\text{noobj}} \sum_{i=0}^{S^2} \sum_{j=0}^{B} \mathbb{1}_{ij}^{\text{noobj}} \left(C_i - \hat{C}_i \right)^2$$

Confidence for no object

$$+ \sum_{i=0}^{S^2} \mathbb{1}_{i}^{\text{obj}} \sum_{c \in \text{classes}} \left(p_i(c) - \hat{p}_i(c) \right)^2$$

Class probability

Down-weight loss from boxes that don't contain objects ($\lambda_{\text{noobj}} = 0.5$)

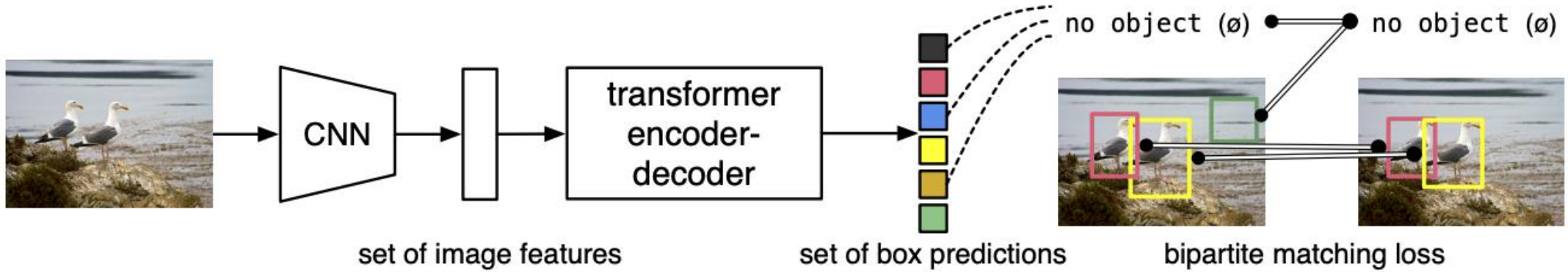
Source: S. Lazebnik

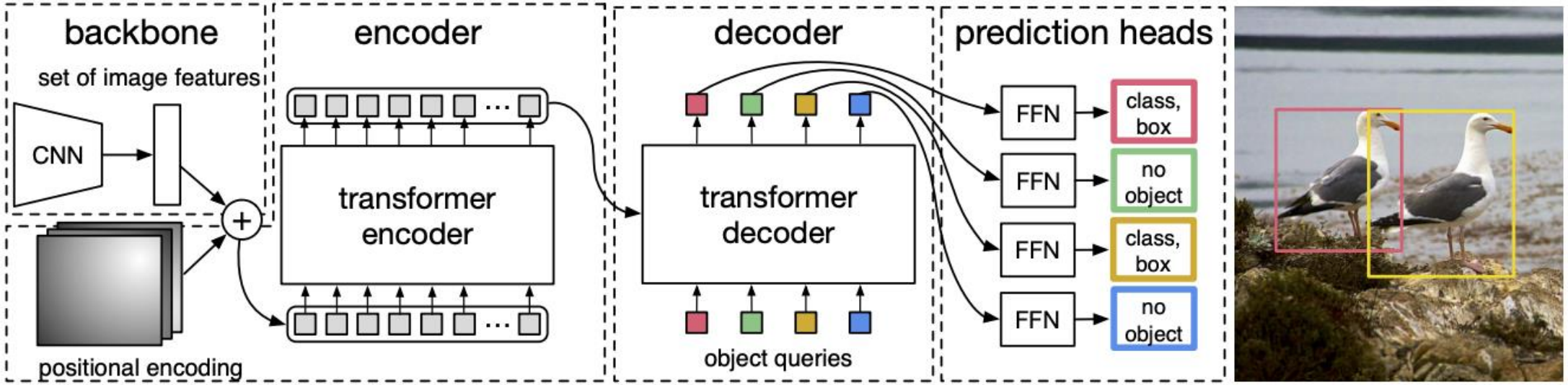
Single vs Two-Stage Detectors

- Single stage is usually faster.
- Two stage is usually better.

- Application dependent: speed vs. accuracy.

Detection Transformer (DETR)





N. Carion et al., End-to-end object detection with transformers, ECCV 2020