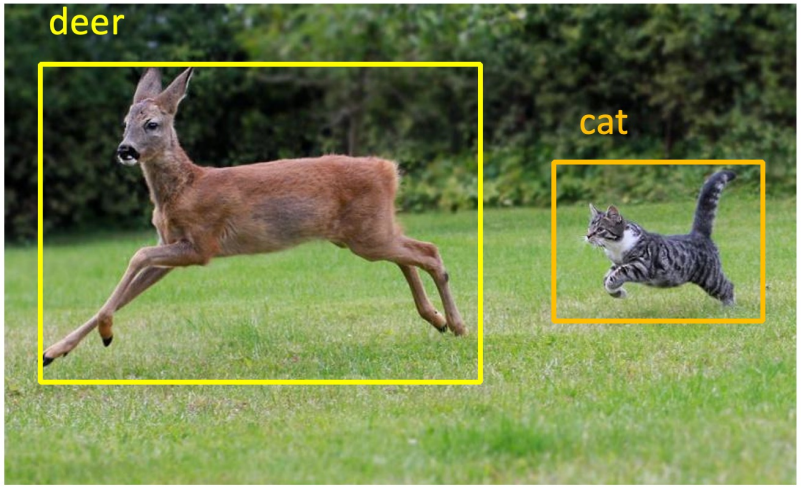
CS 2770: Object Recognition and Image Segmentation

PhD. Nils Murrugarra-Llerena
nem177@pitt.edu

Lab 8a: Object Recognition and Image Segmentation

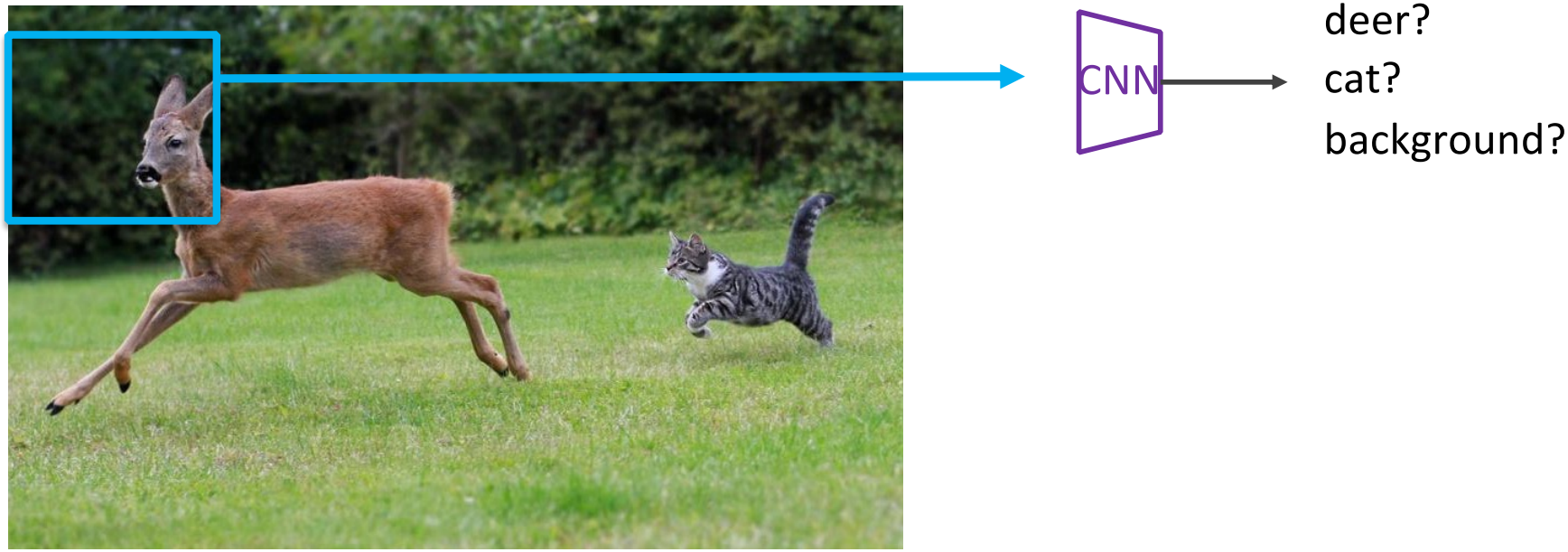
Duration: 10 min

What will be the output of an Object Detector?



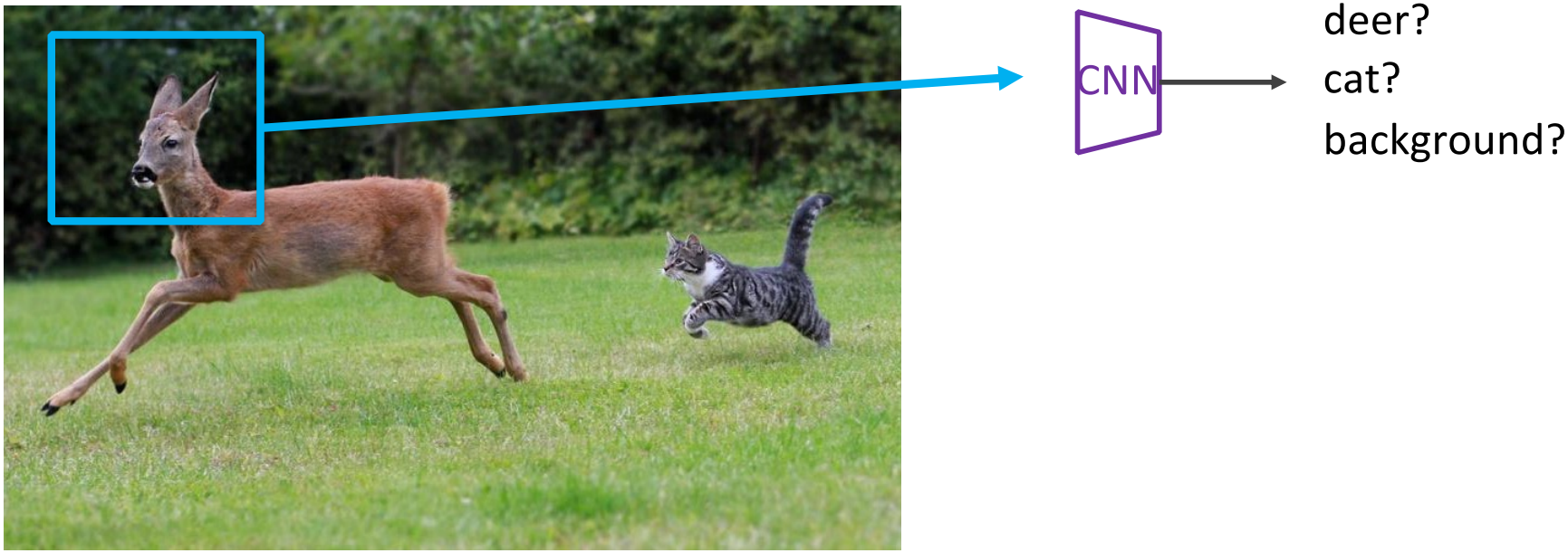
Adapted from Vicente Ordoñez

Object Detection as Classification



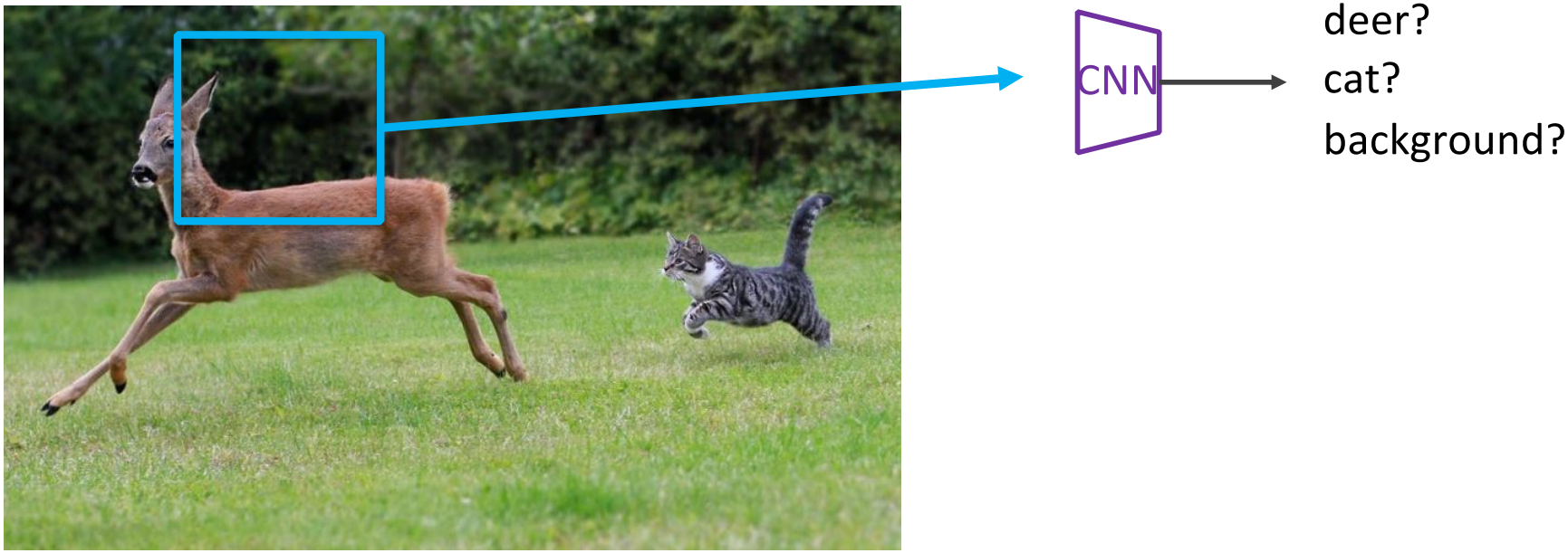
Adapted from Vicente Ordoñez

Object Detection as Classification



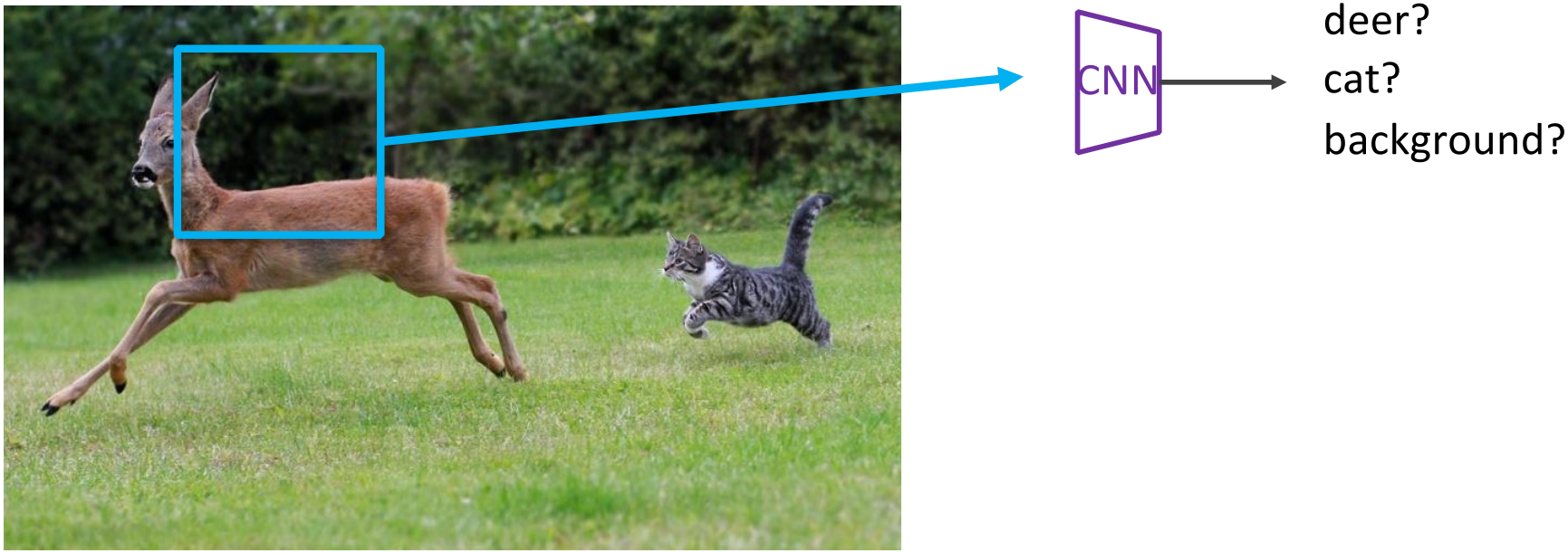
Adapted from Vicente Ordoñez

Object Detection as Classification



Adapted from Vicente Ordoñez

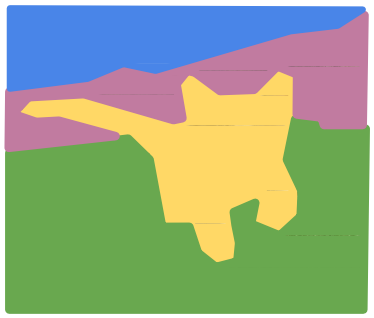
Object Detection as Classification with Sliding Window



Adapted from Vicente Ordoñez

Different Flavors of Object Recognition

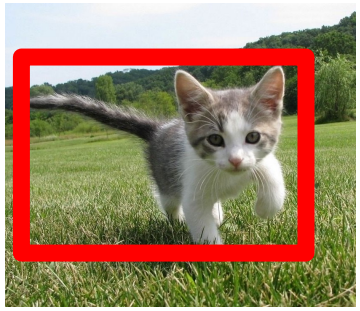
Semantic Segmentation



GRASS, CAT, TREE, SKY

No objects, just pixels

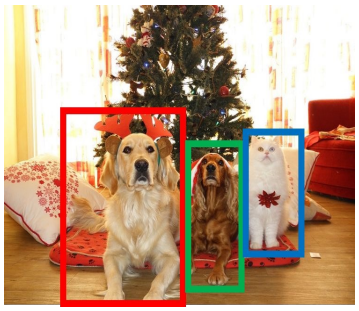
Classification + Localization



CAT

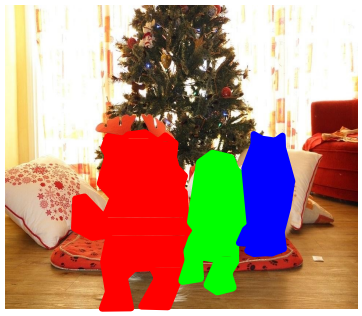
Single Object

Object Detection



DOG, DOG, CAT

Instance Segmentation



DOG, DOG, CAT

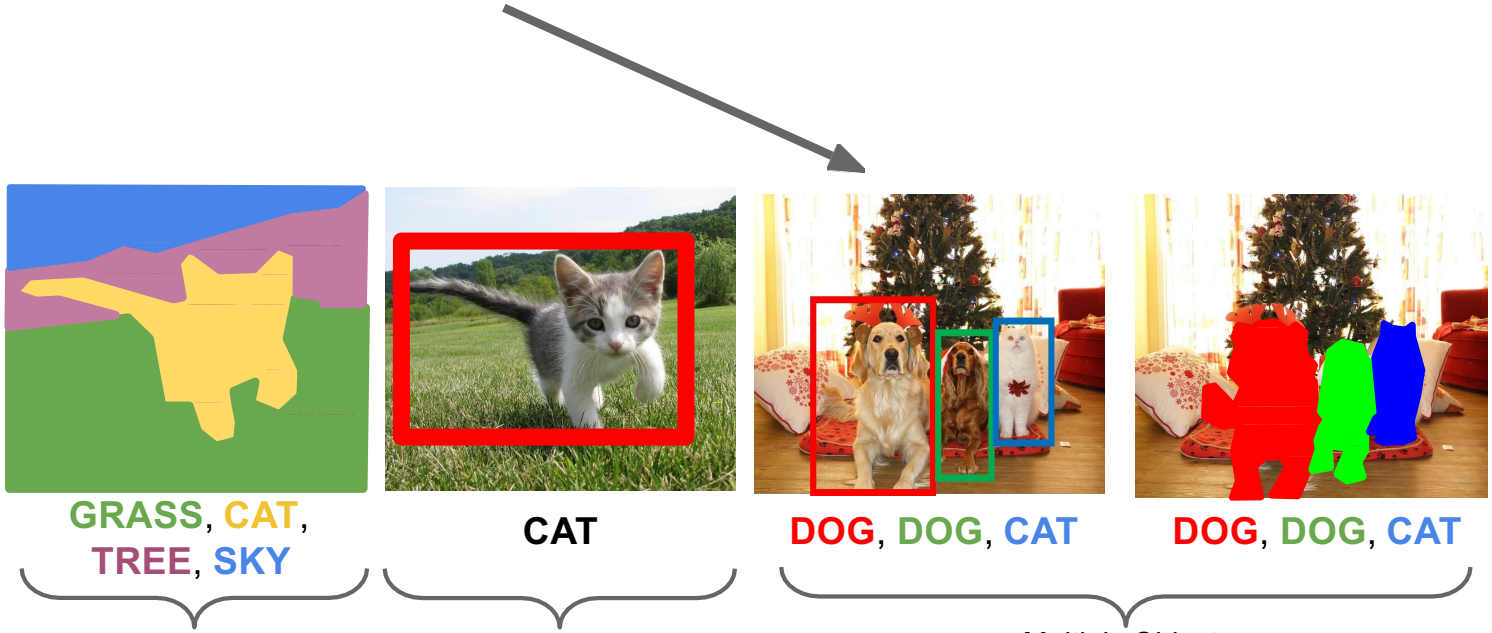
Multiple Object

Adapted from Justin Johnson

Plan for Today

- Detection approaches
  - Pre-CNNs
    - Detection with whole windows: Pedestrian detection
    - Part-based detection: Deformable Part Models
  - Post-CNNs
    - Detection with region proposals: R-CNN, Fast R-CNN, Faster-R-CNN
    - Detection without region proposals: YOLO, SSD, DETR
- Learning from noisy web image-text data
  - Contrastive Language-Image Pretraining (CLIP)
    - Prompting
  - Open-vocabulary object detection

Object Detection



Slide by: Justin Johnson

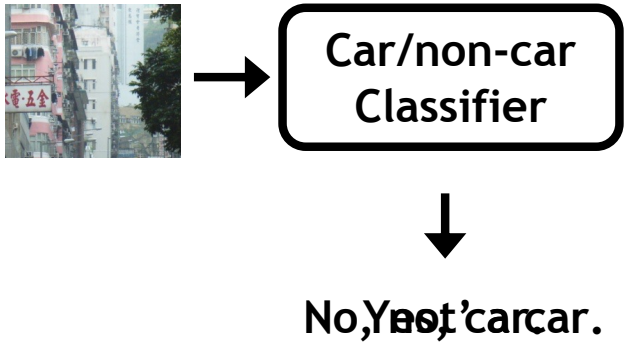
Object detection: basic framework

- Build/train object model
- Generate candidate regions in new image
- Score the candidates

Adapted from Kristen Grauman

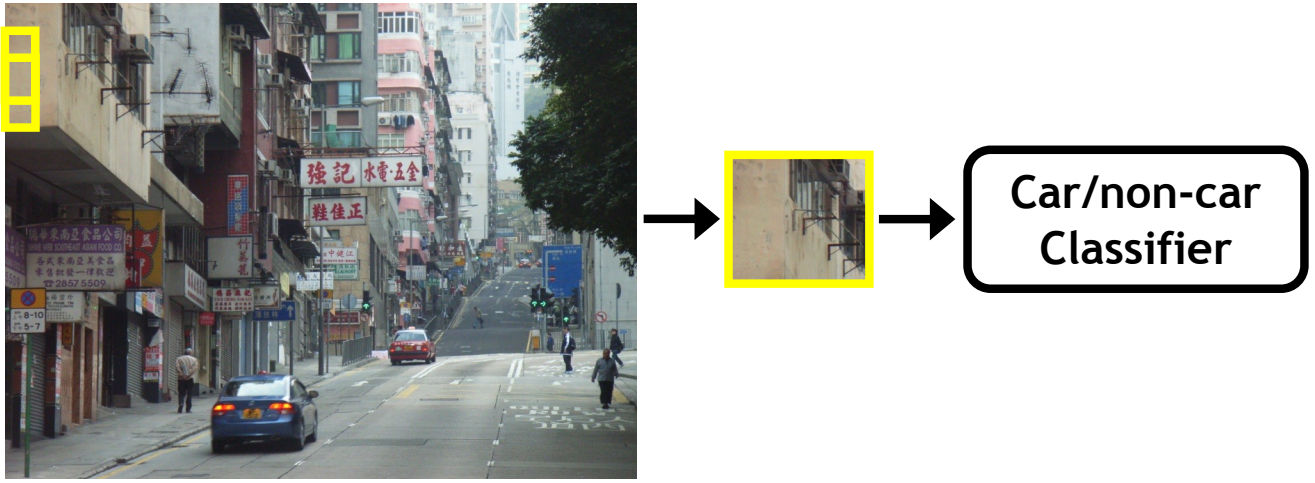
Window-template-based models Building an object model

Given the representation, train a binary classifier



Kristen Grauman

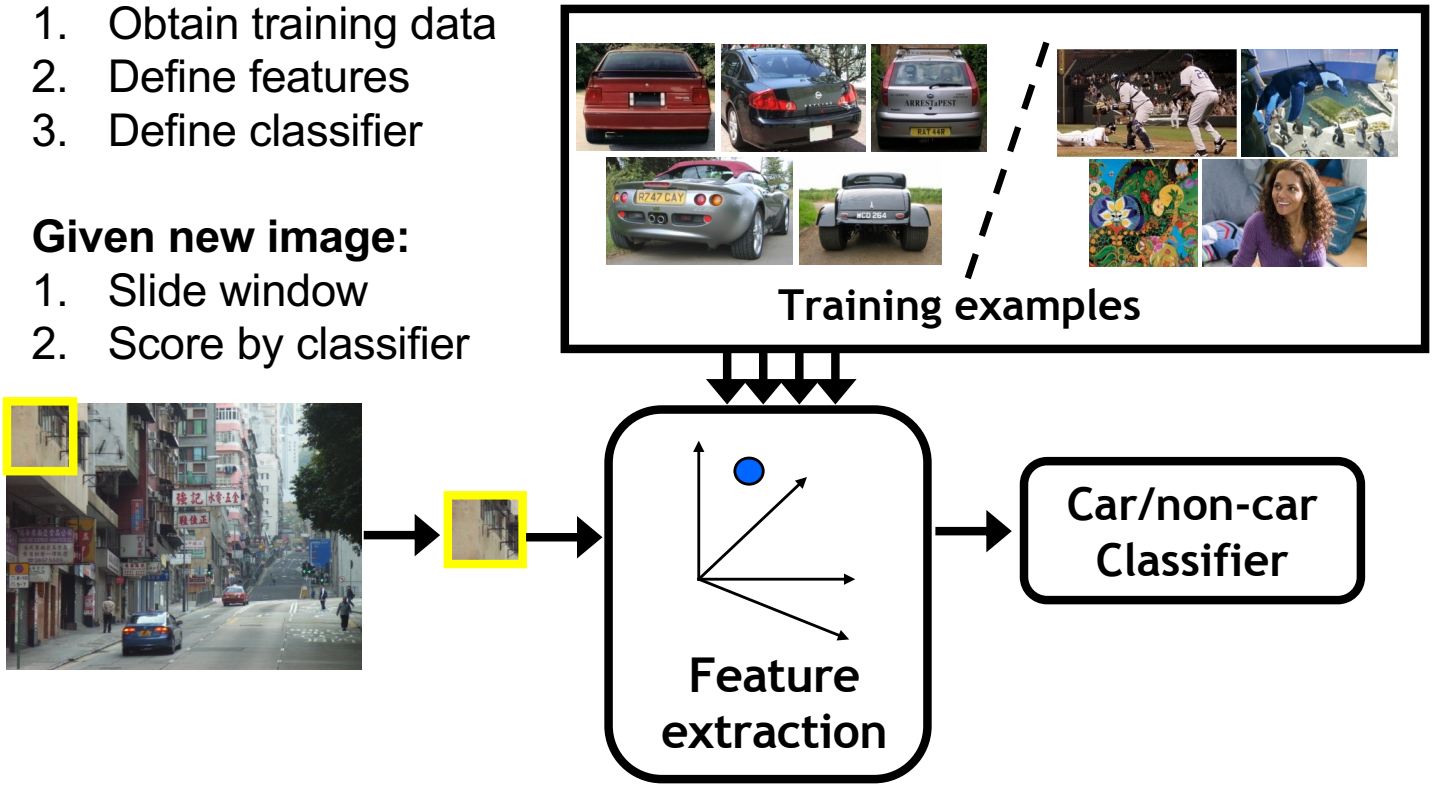
Window-template-based models Generating and scoring candidates



Window-template-based object detection: recap

Training:
1. Obtain training data
2. Define features
3. Define classifier

Given new image:
1. Slide window
2. Score by classifier

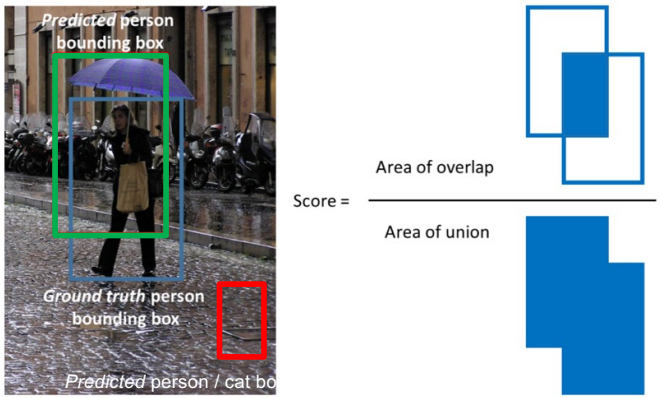


Kristen Grauman

Evaluating detection methods

$$mAP = \frac{1}{|classes|} \sum_{c \in classes} \frac{\#TP(c)}{\#TP(c) + \#FP(c)}$$

- True Positive - TP(c): a predicted bounding box (pred_bb) was made for class c, there is a ground truth bounding box (gt_bb) of class c, and IoU(pred_bb, gt_bb) >= 0.5.
- False Positive - FP(c): a pred_bb was made for class c, and there is no gt_bb of class c. Or there is a gt_bb of class c, but IoU(pred_bb, gt_bb) < 0.5.



Dalal-Triggs pedestrian detector

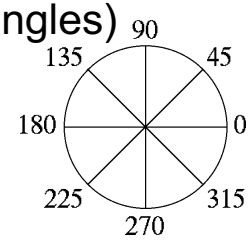
1. Extract fixed-sized (64x128 pixel) window at multiple positions and scales
2. Compute HOG (histogram of gradient) features within each window
3. Score the window with a linear SVM classifier
4. Perform non-maxima suppression to remove overlapping detections with lower scores

Navneet Dalal and Bill Triggs, Histograms of Oriented Gradients for Human Detection, CVPR05

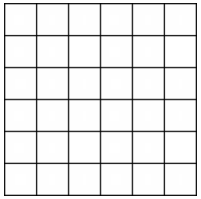
Histograms of oriented gradients (HOG)

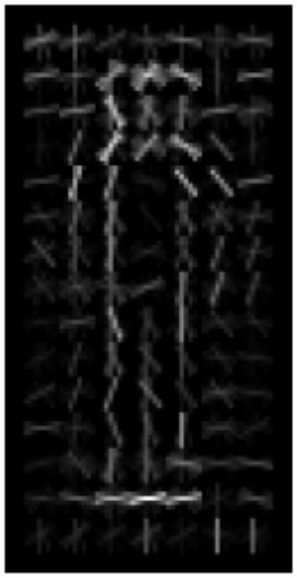
Divide image into 8x8 regions

Orientation: 9 bins (for unsigned angles)



Histograms in 8x8 pixel cells





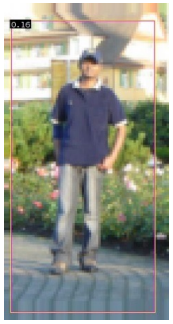
Votes weighted by magnitude

Adapted from Pete Barnum

Navneet Dalal and Bill Triggs, Histograms of Oriented Gradients for Human Detection, CVPR05

Train SVM for pedestrian detection using HoG

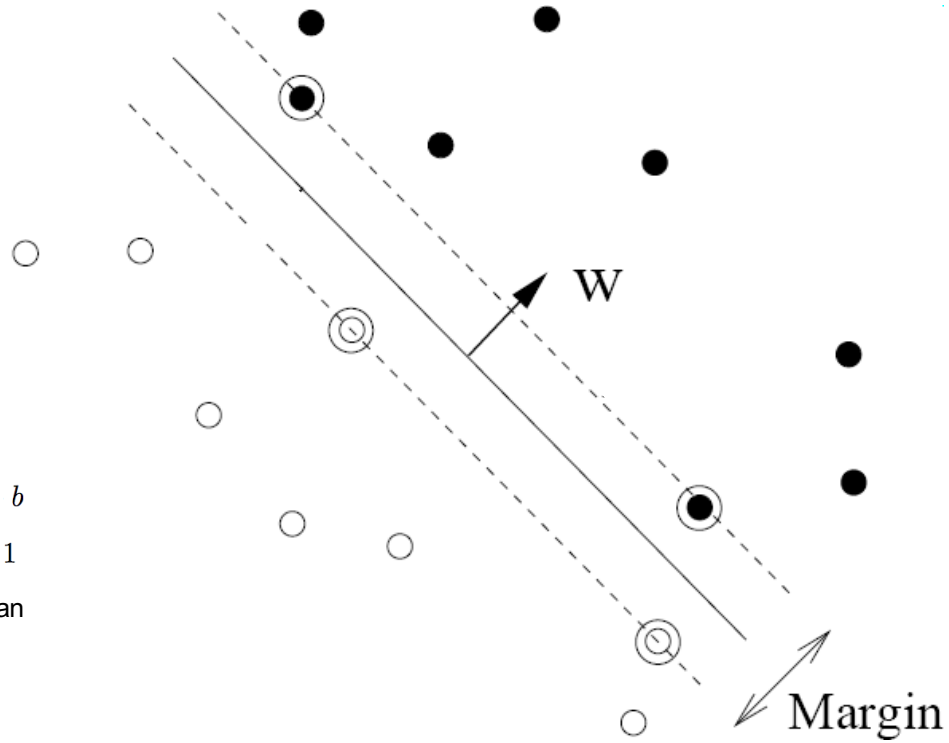
pos w



$0.16 = w^T x + b$

$sign(0.16) = 1$

=> pedestrian

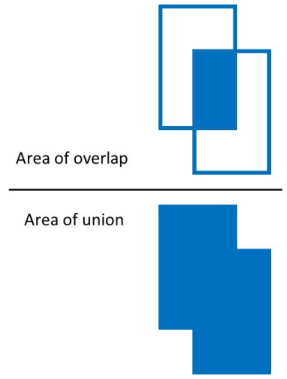


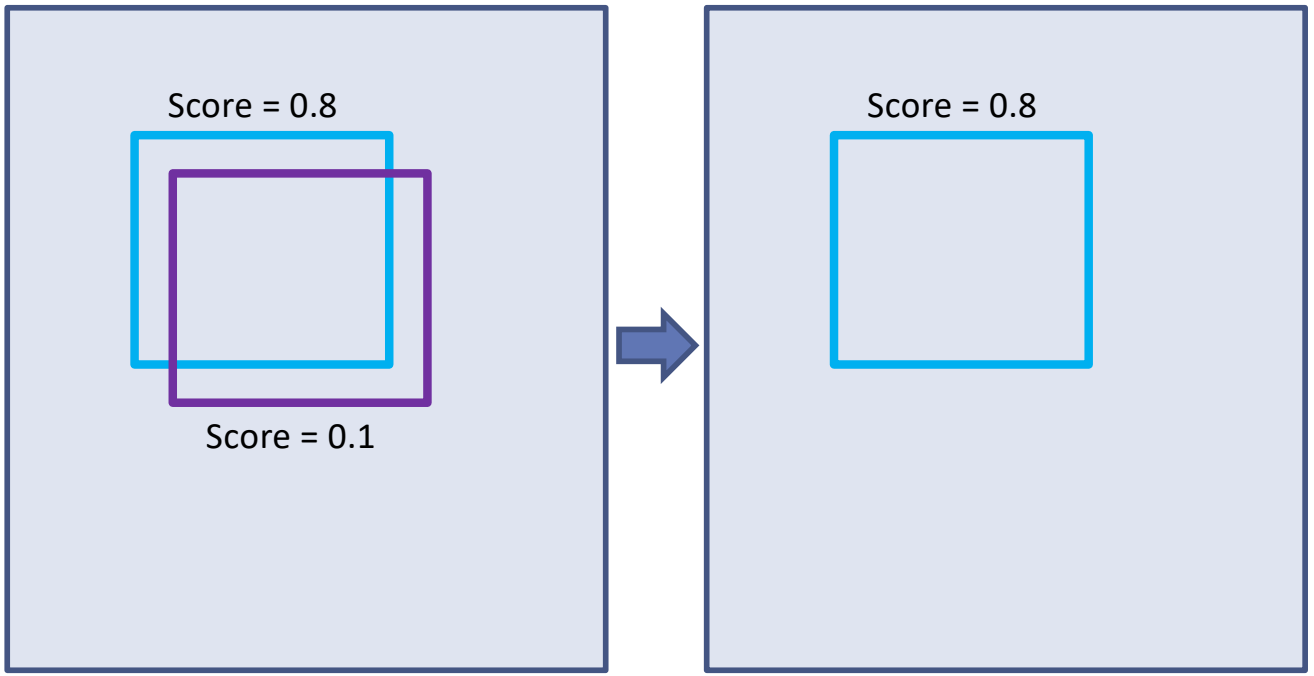
Adapted from Pete Barnum

Navneet Dalal and Bill Triggs, Histograms of Oriented Gradients for Human Detection, CVPR05

Remove overlapping detections

Non-max suppression





Adapted from Derek Hoiem

Are window templates enough?

- Many objects are articulated, or have parts that can vary in configuration

Images from Caltech-256, D. Ramanan

- Many object categories look very different from different viewpoints, or from instance to instance

Adapted from N. Snavely, D. Tran

Parts-based Models

Define object by collection of parts modeled by

1. Appearance
2. Spatial configuration

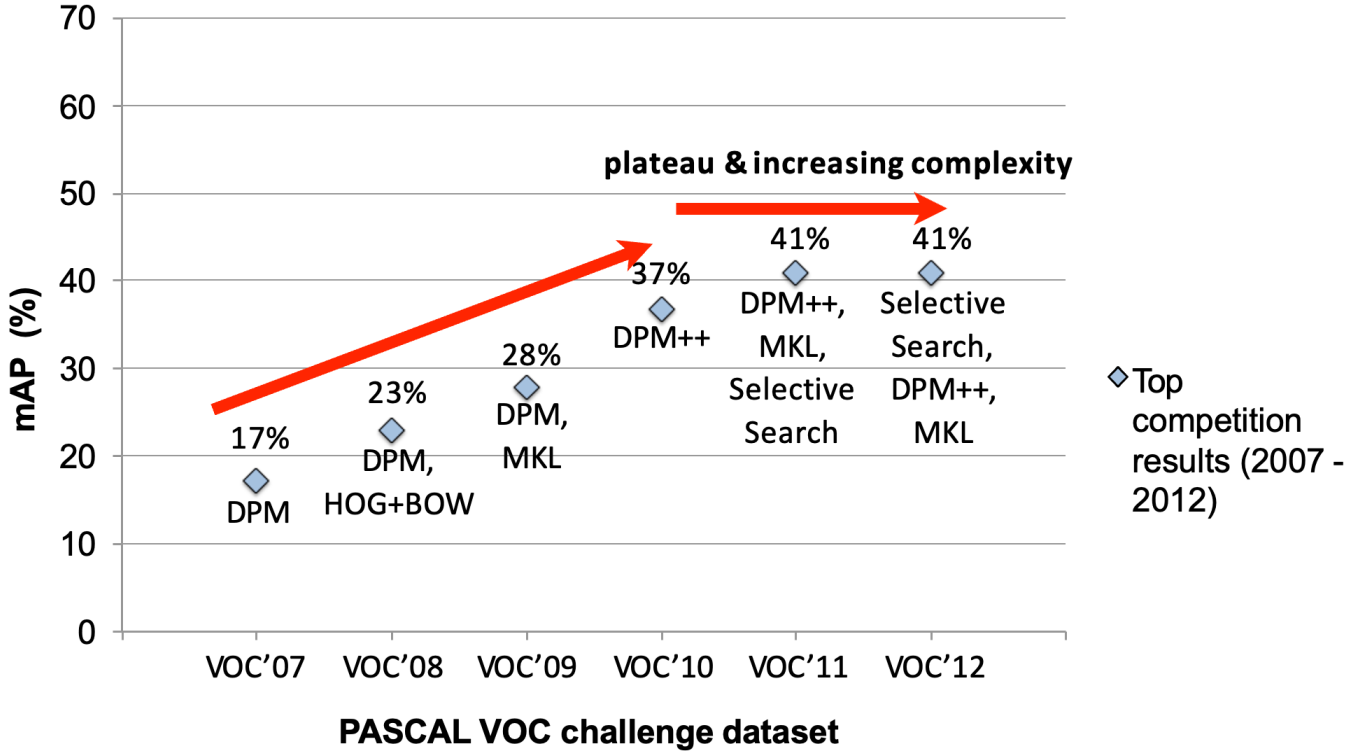
Slide credit: Rob Fergus

Plan for the next three lectures

- Detection approaches
  - Pre-CNNs
    - Detection with whole windows: Pedestrian detection
    - Part-based detection: Deformable Part Models
  - Post-CNNs
    - Detection with region proposals: R-CNN, Fast R-CNN, Faster-R-CNN
    - Detection without region proposals: YOLO, SSD
- Learning from noisy web image-text data
  - Contrastive Language-Image Pretraining (CLIP)
    - Prompting
  - Open-vocabulary object detection

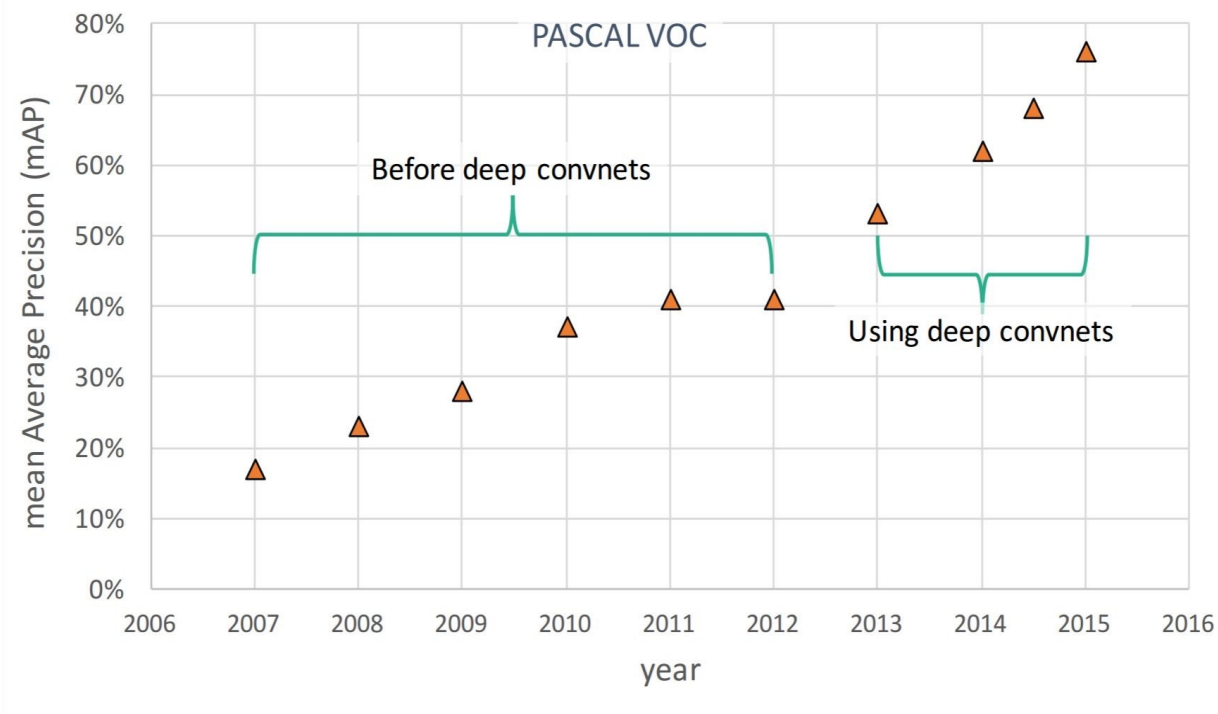
Complexity and the plateau

[Source: http://pascallin.ecs.soton.ac.uk/challenges/VOC/voc20{07,08,09,10,11,12}/results/index.html]



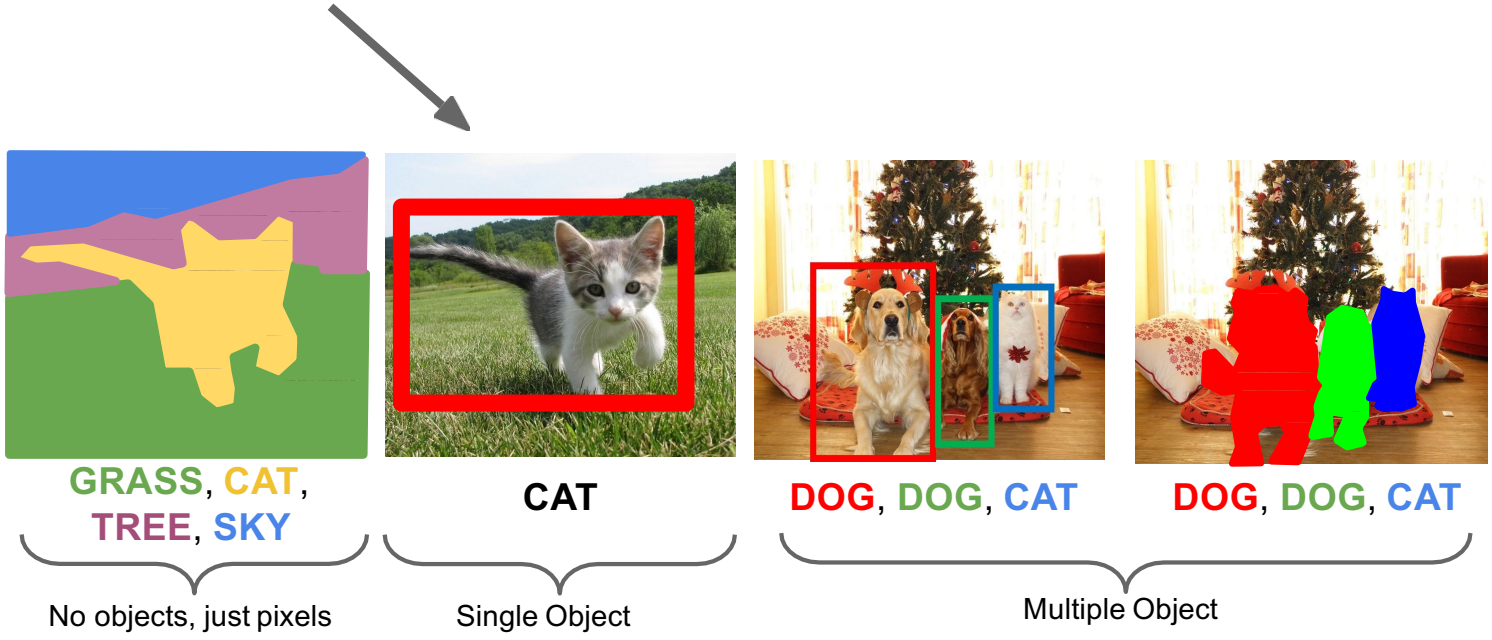
Girshick et al., "Rich Feature Hierarchies for Accurate Object Detection and Semantic Segmentation", CVPR 2014

Impact of Deep Learning



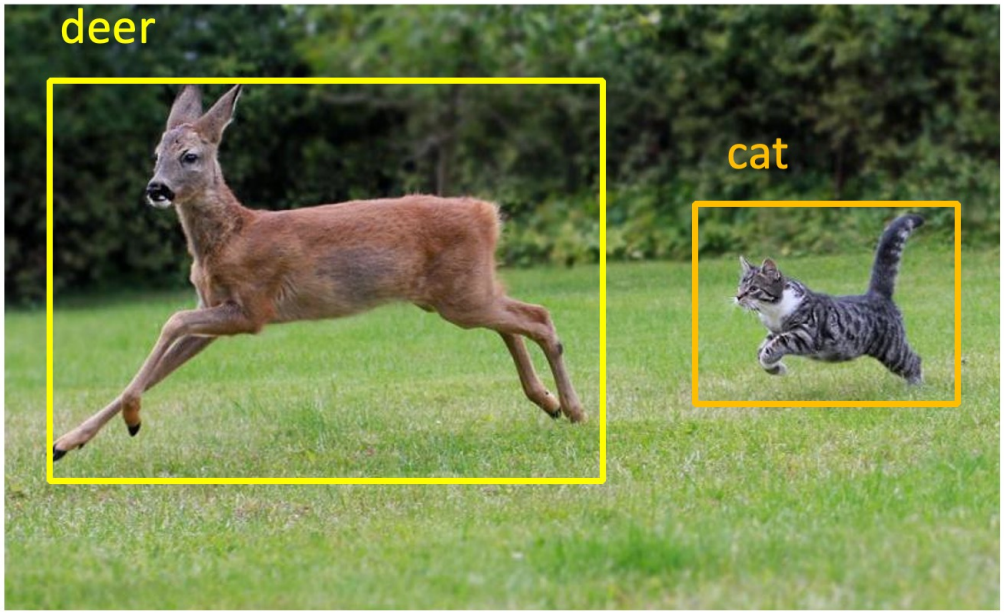
Slide by: Justin Johnson

Classification + Localization



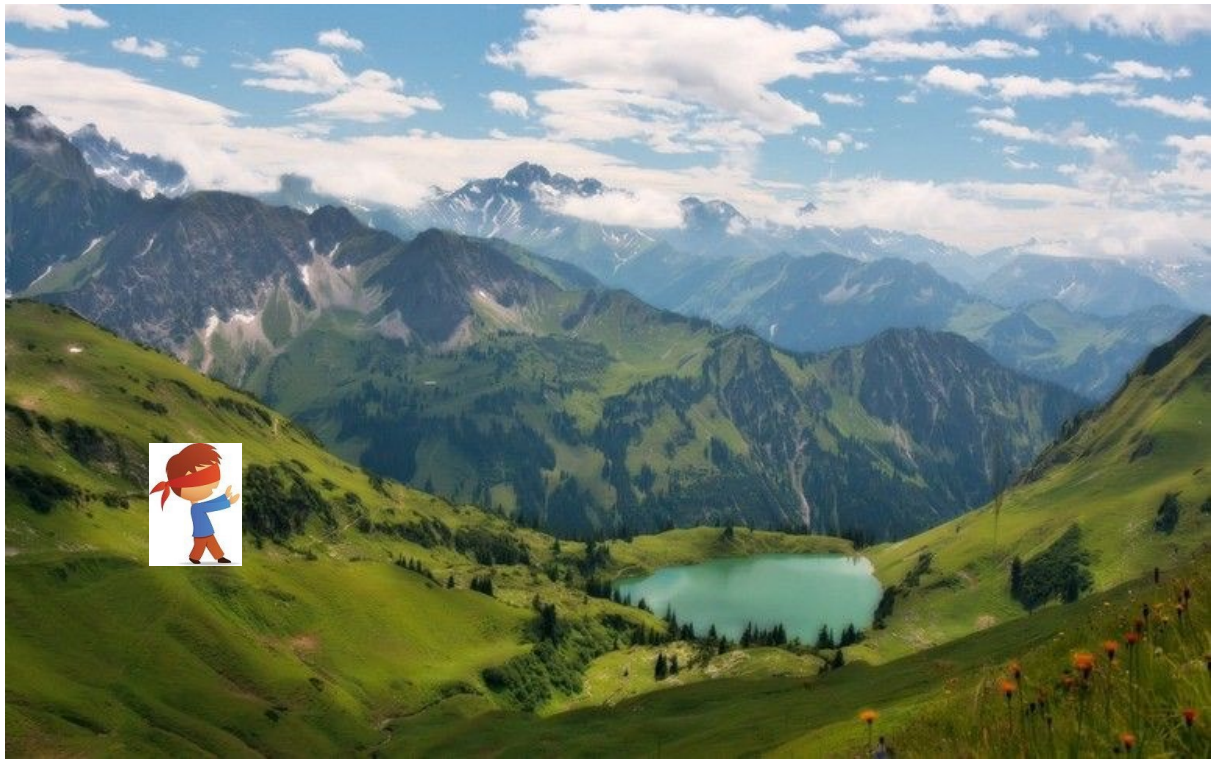
Slide by: Justin Johnson

What will be the output of a neural network for object detection?

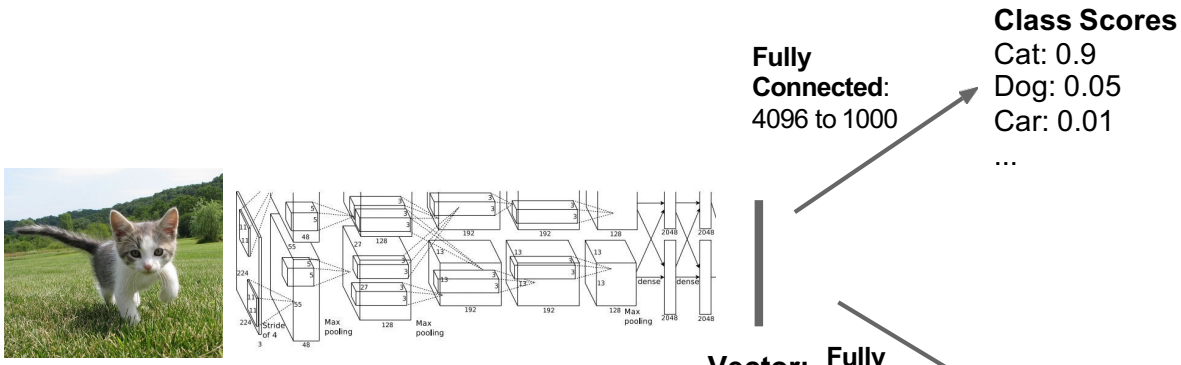


Adapted from Vicente Ordoñez

What guides the learning of a neural network?



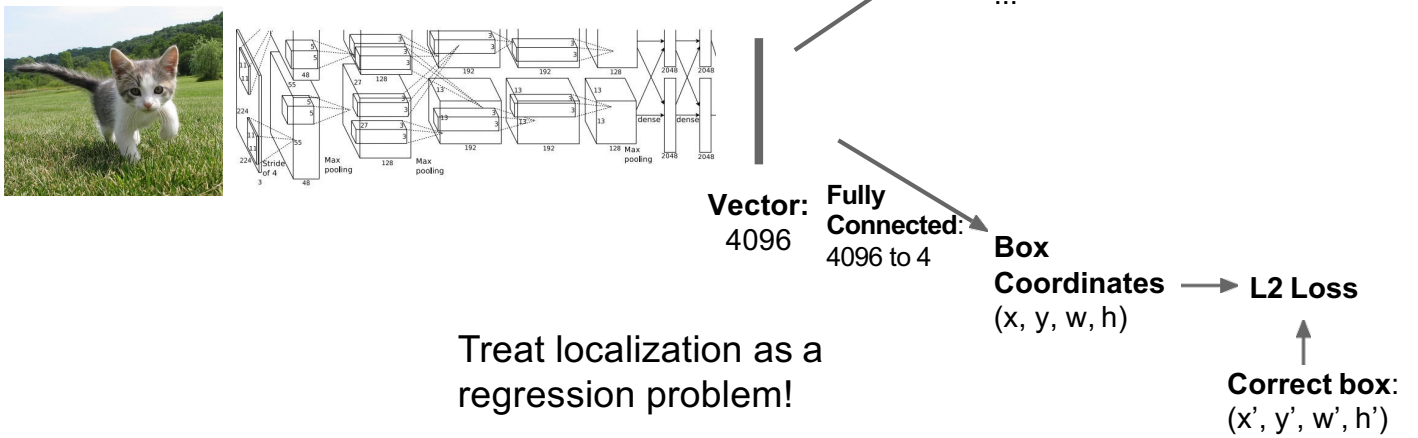
Classification + Localization



Treat localization as a regression problem!

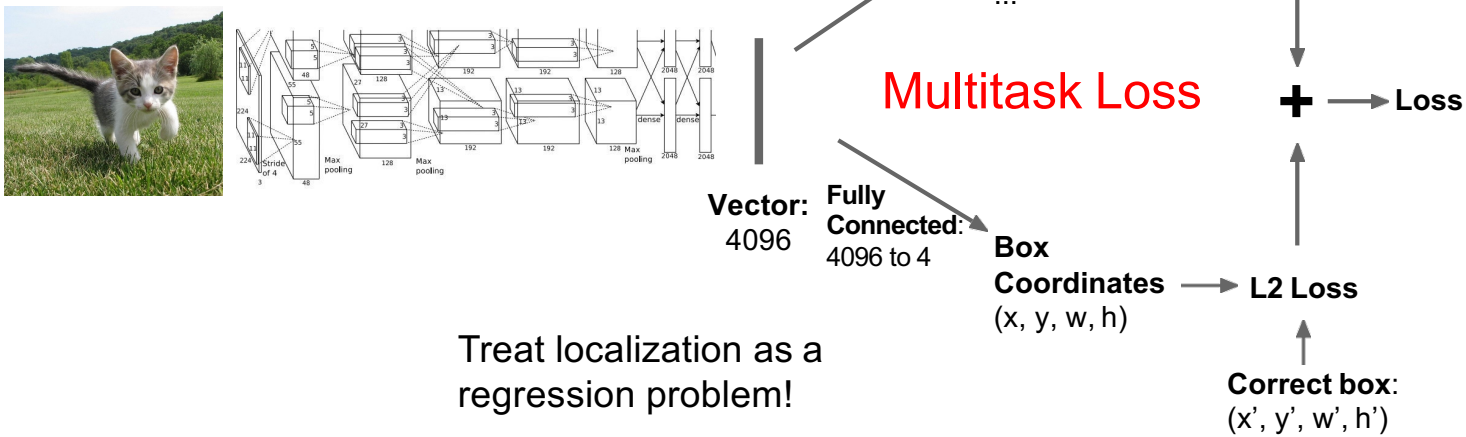
Slide by: Justin Johnson

Classification + Localization



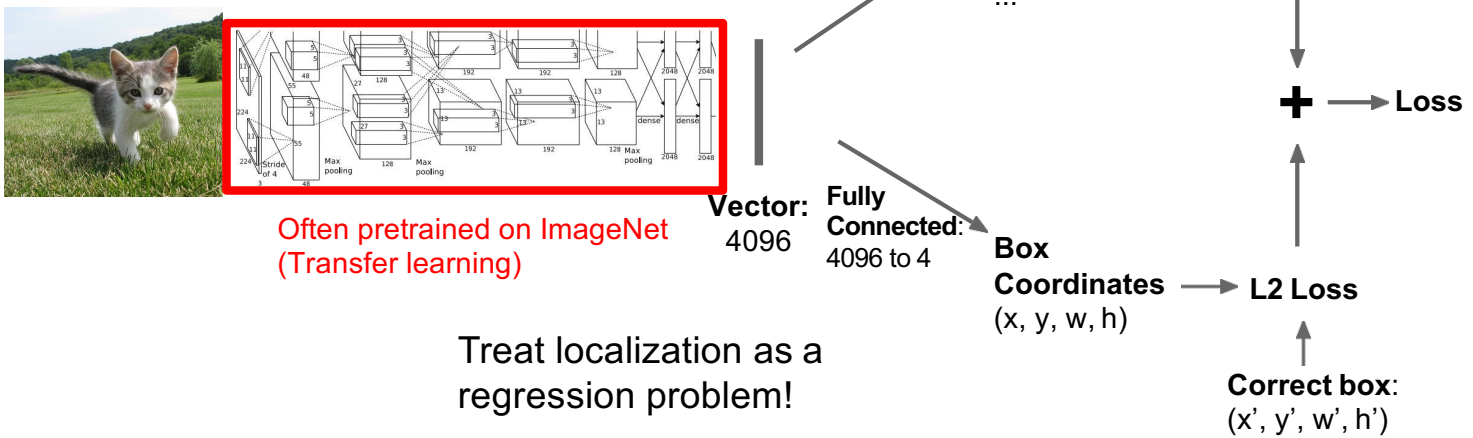
Slide by: Justin Johnson

Classification + Localization



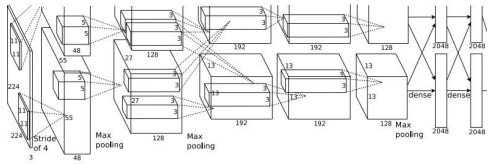
Slide by: Justin Johnson

Classification + Localization

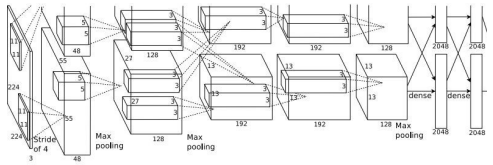


Slide by: Justin Johnson

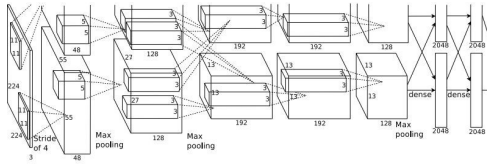
Object Detection as Regression?



CAT: (x, y, w, h)



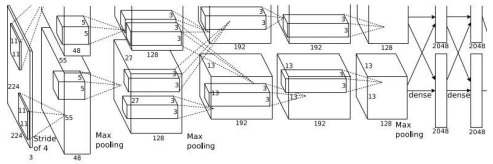
DOG: (x, y, w, h)
DOG: (x, y, w, h)
CAT: (x, y, w, h)



DUCK: (x, y, w, h)
DUCK: (x, y, w, h)
….

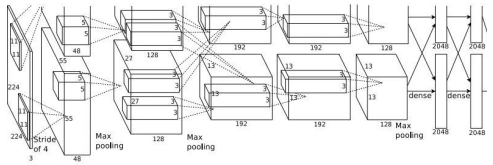
Slide by: Justin Johnson

Object Detection as Regression?



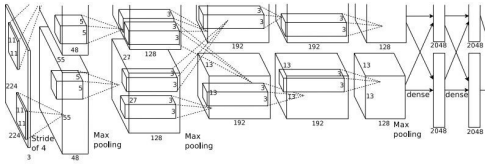
CAT: (x, y, w, h)

4 numbers



DOG: (x, y, w, h)
DOG: (x, y, w, h)
CAT: (x, y, w, h)

16 numbers



DUCK: (x, y, w, h)
DUCK: (x, y, w, h)
….

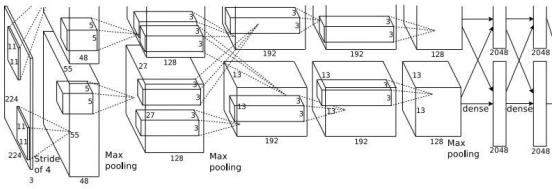
Many numbers!

Each image needs a different number of outputs!

Slide by: Justin Johnson

Object Detection as Classification: Sliding Window

Apply a CNN to many different crops of the image, CNN classifies each crop as object or background

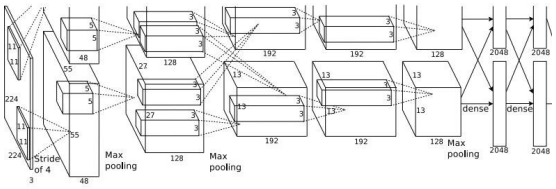


Dog? NO
Cat? NO
Background? YES

Slide by: Justin Johnson

Object Detection as Classification: Sliding Window

Apply a CNN to many different crops of the image, CNN classifies each crop as object or background

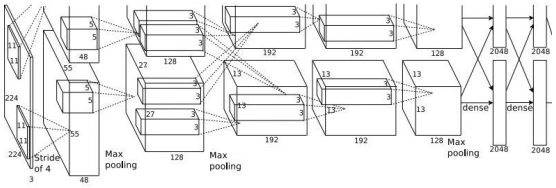


Dog? YES
Cat? NO
Background? NO

Slide by: Justin Johnson

Object Detection as Classification: Sliding Window

Apply a CNN to many different crops of the image, CNN classifies each crop as object or background

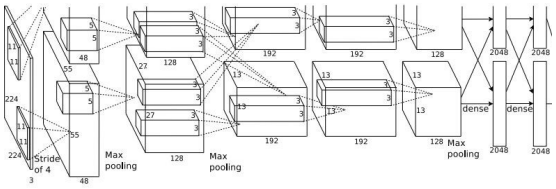


Dog? YES
Cat? NO
Background? NO

Slide by: Justin Johnson

Object Detection as Classification: Sliding Window

Apply a CNN to many different crops of the image, CNN classifies each crop as object or background

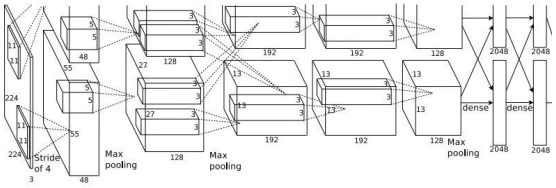


Dog? NO
Cat? YES
Background? NO

Slide by: Justin Johnson

Object Detection as Classification: Sliding Window

Apply a CNN to many different crops of the image, CNN classifies each crop as object or background



Dog? NO
Cat? YES
Background? NO

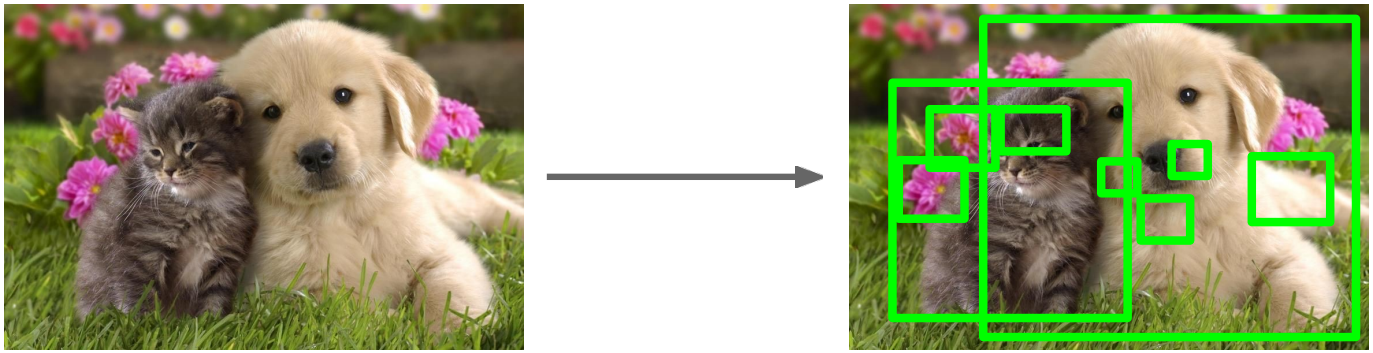
Problem: Need to apply CNN to huge number of locations and scales, very computationally expensive!

What can we do?

Slide by: Justin Johnson

Region Proposals

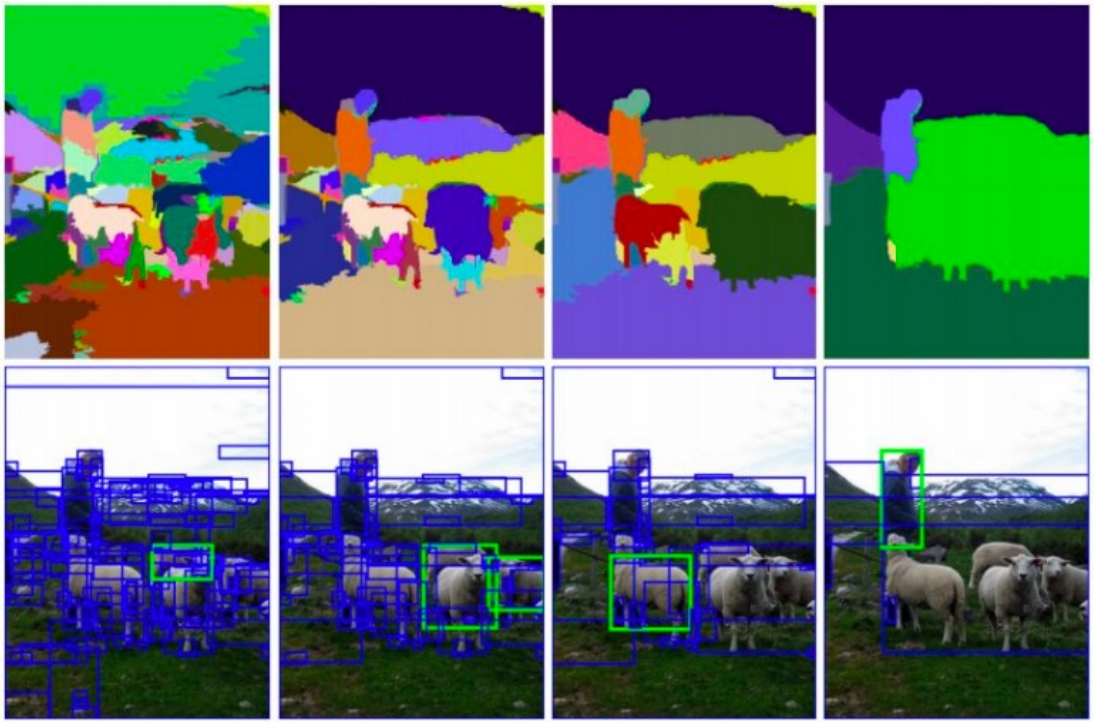
- Find “blobby” image regions that are likely to contain objects
- Relatively fast to run; e.g. Selective Search gives 1000 region proposals in a few seconds on CPU



Alexe et al, “Measuring the objectness of image windows”, TPAMI 2012
Uijlings et al, “Selective Search for Object Recognition”, IJCV 2013
Cheng et al, “BING: Binarized normed gradients for objectness estimation at 300fps”, CVPR 2014
Zitnick and Dollar, “Edge boxes: Locating object proposals from edges”, ECCV 2014

Slide by: Justin Johnson

Box Proposal Method – SS: Selective Search



Segmentation As Selective Search for Object Recognition. van de Sande et al. ICCV 2011

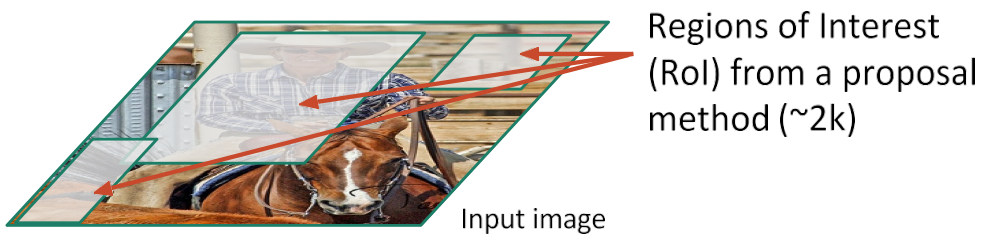
Adapted from Vicente Ordoñez

R-CNN

Input image

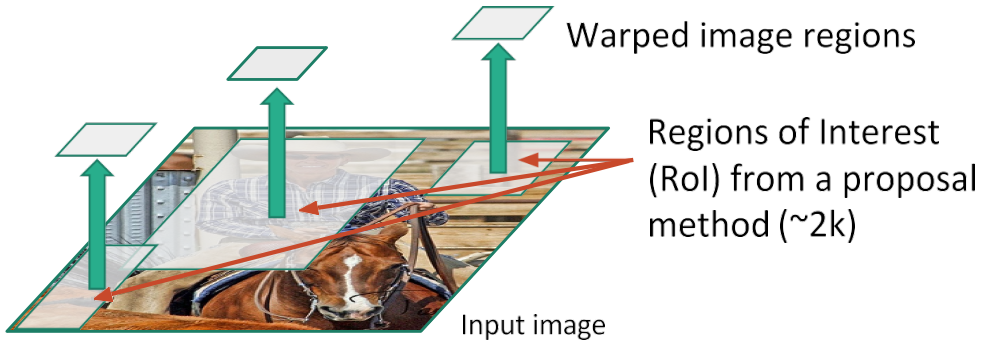
Girshick et al., "Rich Feature Hierarchies for Accurate Object Detection and Semantic Segmentation", CVPR 2014

R-CNN



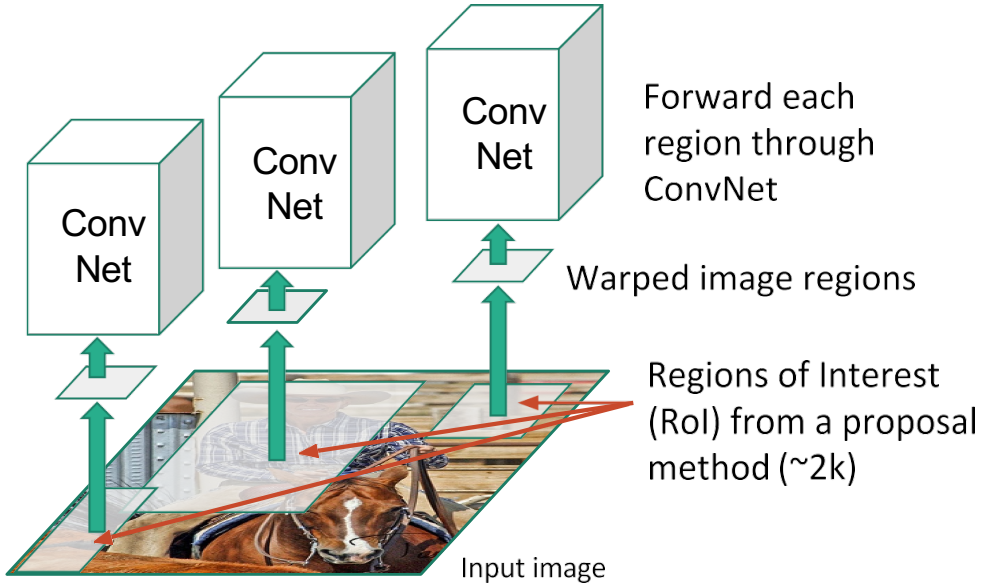
Girshick et al., "Rich Feature Hierarchies for Accurate Object Detection and Semantic Segmentation", CVPR 2014

R-CNN



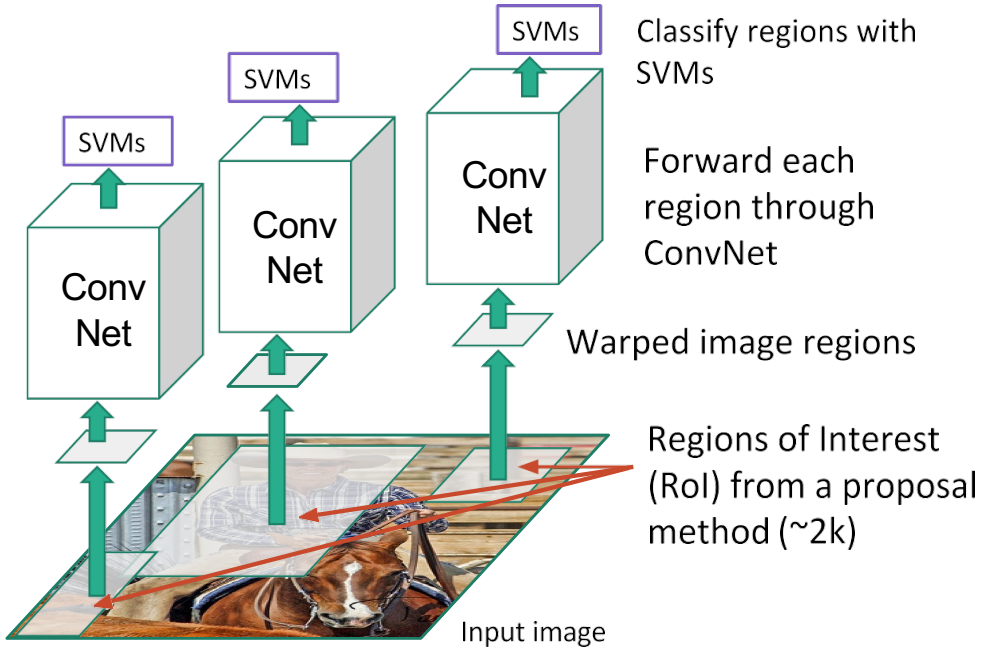
Girshick et al., "Rich Feature Hierarchies for Accurate Object Detection and Semantic Segmentation", CVPR 2014

R-CNN



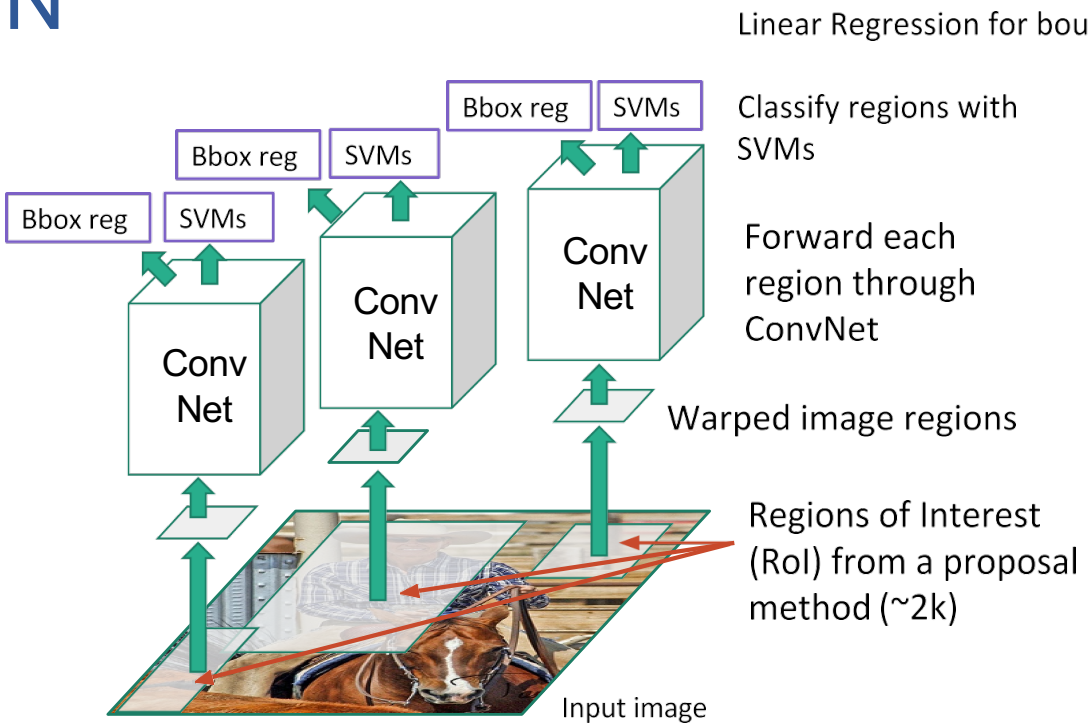
Girshick et al., "Rich Feature Hierarchies for Accurate Object Detection and Semantic Segmentation", CVPR 2014

R-CNN



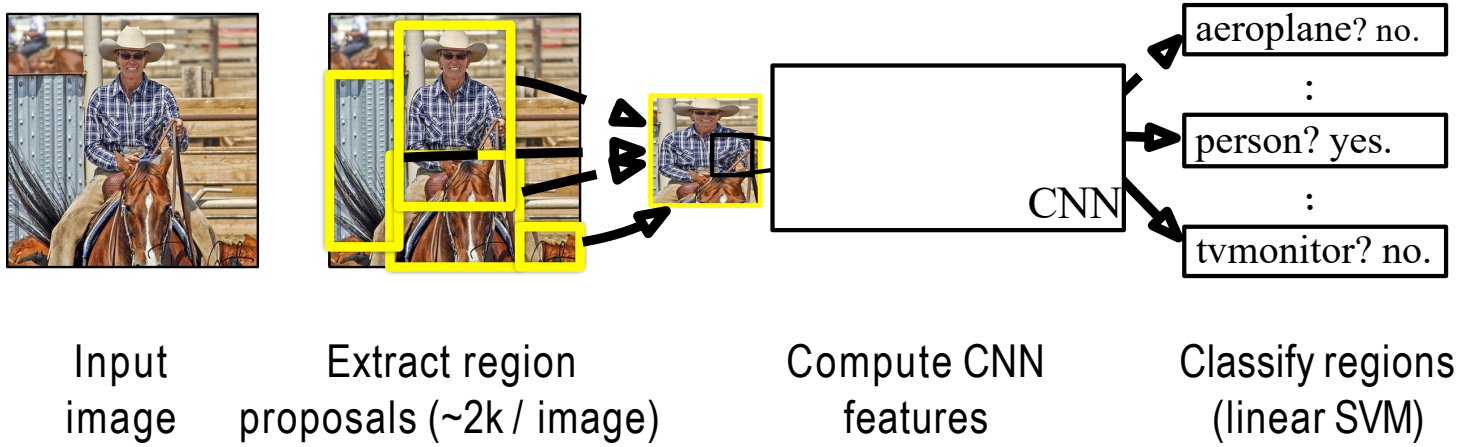
Girshick et al., "Rich Feature Hierarchies for Accurate Object Detection and Semantic Segmentation", CVPR 2014

R-CNN



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

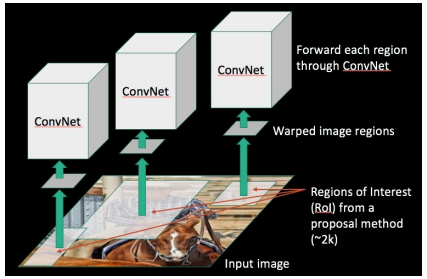
R-CNN: Regions with CNN features



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

What's wrong with slow R-CNN?

- Ad-hoc training objectives
  - Train post-hoc linear SVMs (hinge loss)
  - Train post-hoc bounding-box regressions (L2 loss)
- Training is slow (84h), takes a lot of disk space
  - Need to store all region crops
- Inference (detection) is slow
  - 47s / image with VGG16 [Simonyan & Zisserman, ICLR15]



~2000 ConvNet forward passes per image

Adapted from Girshick, "Fast R-CNN", ICCV 2015

Fast R-CNN

- One network, applied one time, not 2000 times
- Trained end-to-end (in one stage)
- Fast test time
- Higher mean average precision

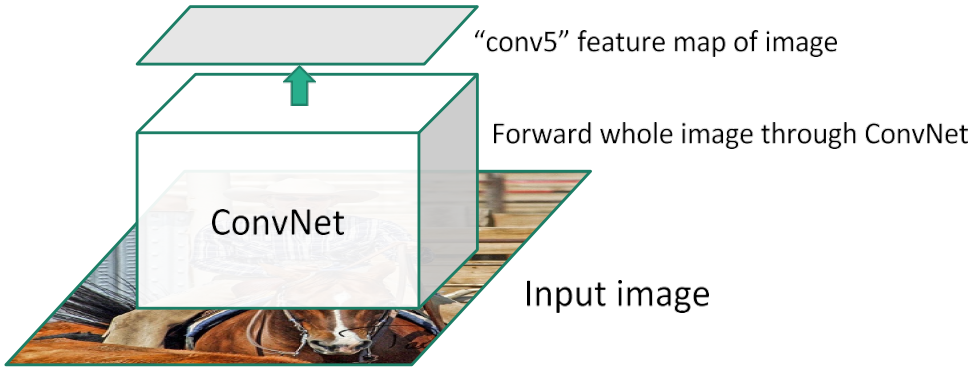
Adapted from Girshick, "Fast R-CNN", ICCV 2015

Fast R-CNN

Input image

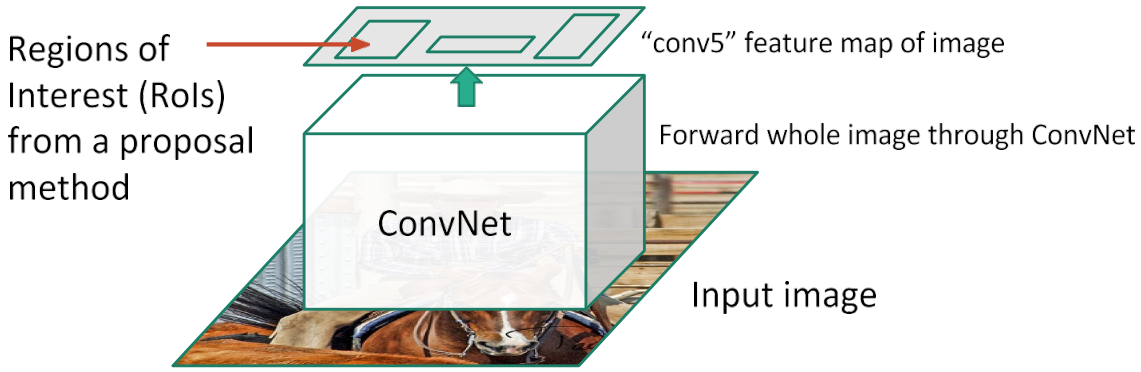
Adapted from Girshick, "Fast R-CNN", ICCV 2015

Fast R-CNN



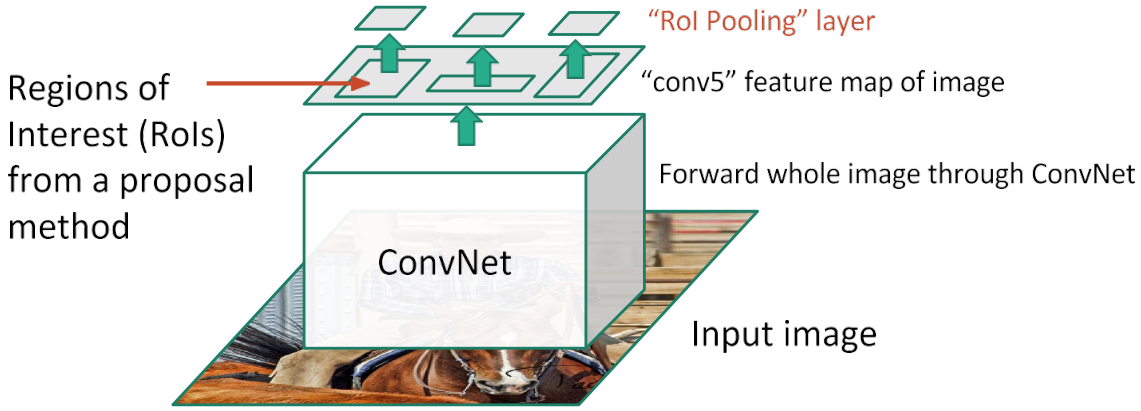
Adapted from Girshick, "Fast R-CNN", ICCV 2015

Fast R-CNN



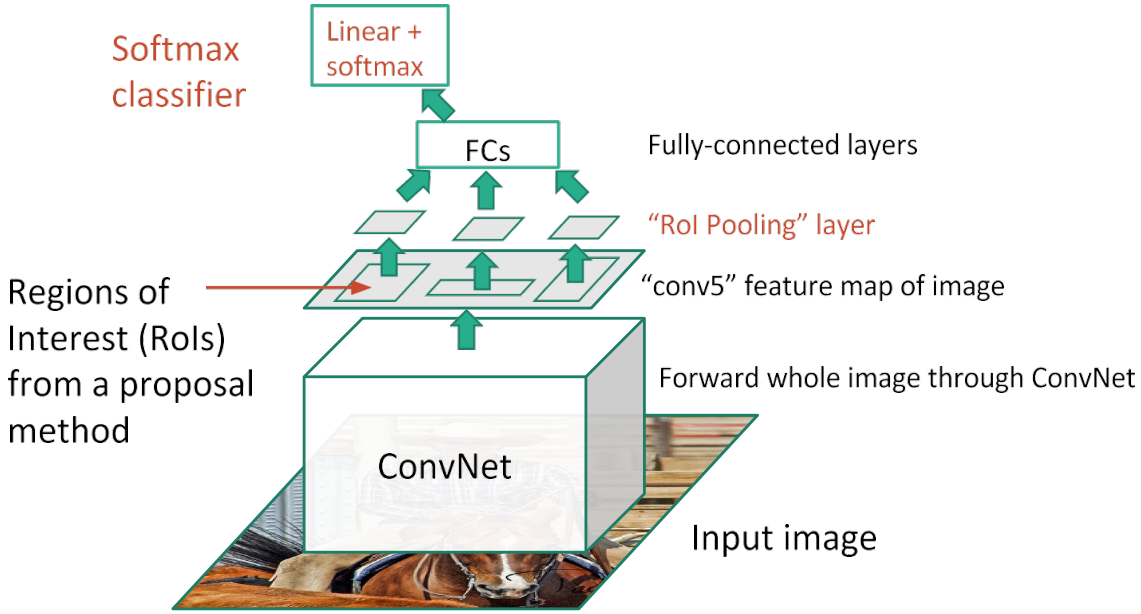
Adapted from Girshick, "Fast R-CNN", ICCV 2015

Fast R-CNN



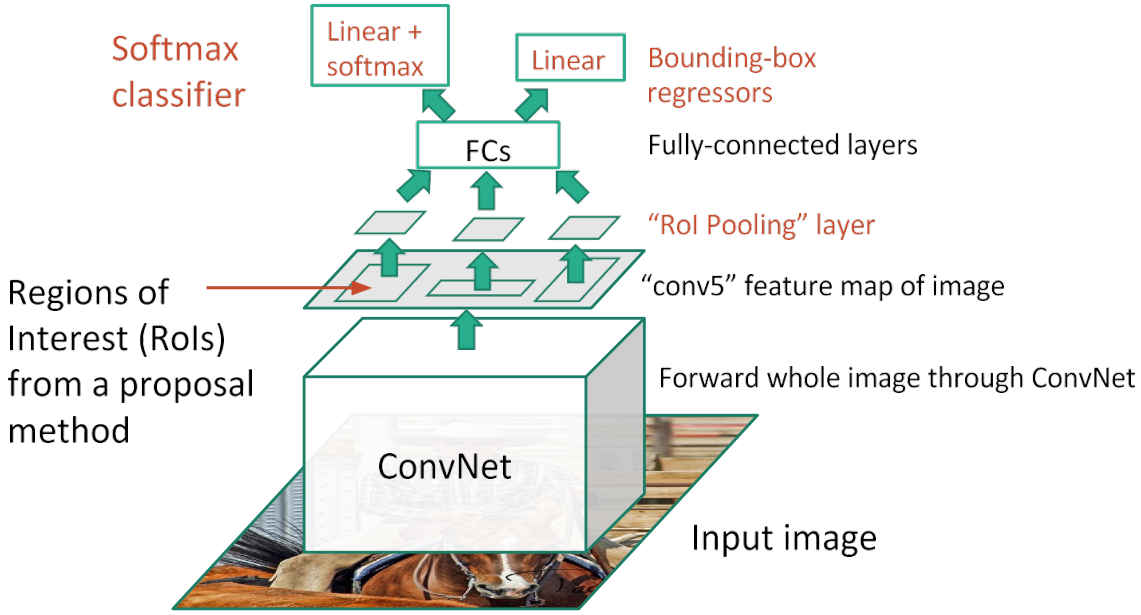
Adapted from Girshick, "Fast R-CNN", ICCV 2015

Fast R-CNN



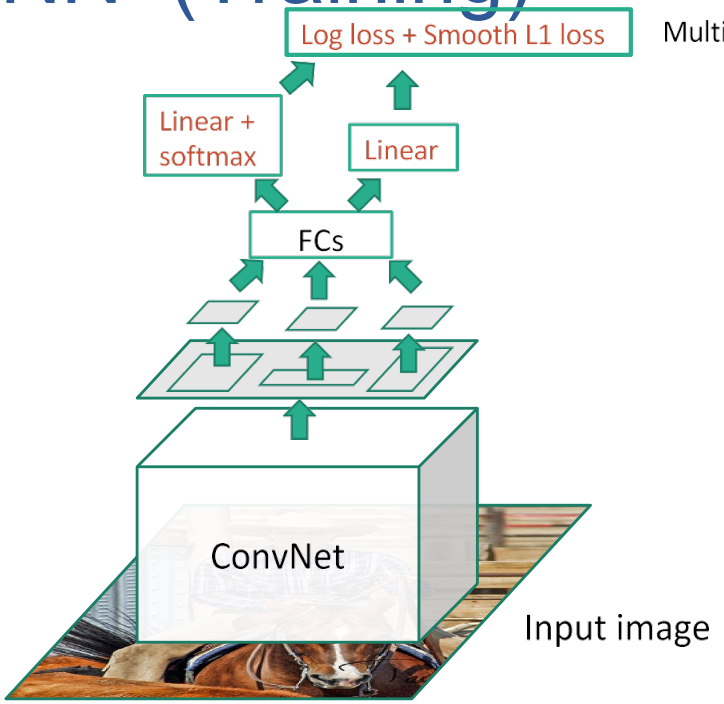
Adapted from Girshick, "Fast R-CNN", ICCV 2015

Fast R-CNN



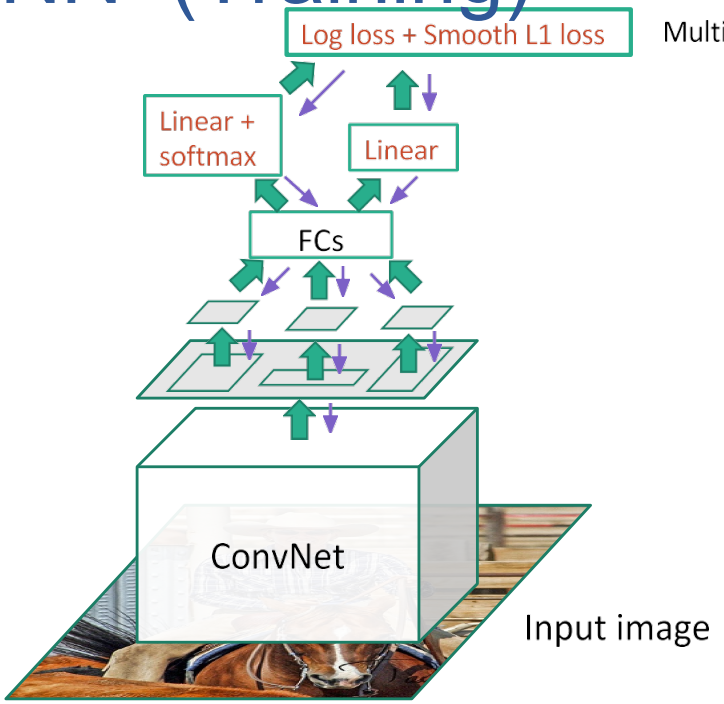
Adapted from Girshick, "Fast R-CNN", ICCV 2015

Fast R-CNN (Training)



Adapted from Girshick, "Fast R-CNN", ICCV 2015

Fast R-CNN (Training)



Adapted from Girshick, "Fast R-CNN", ICCV 2015

Fast R-CNN vs R-CNN

	Fast R-CNN	R-CNN
Train time (h)	9.5	84
Speedup	8.8x	1x
Test time / image	0.32s	47.0s
Test speedup	146x	1x
mAP	66.9%	66.0%

Timings exclude object proposal time, which is equal for all methods. All methods use VGG16 from Simonyan and Zisserman.

Adapted from Girshick, "Fast R-CNN", ICCV 2015

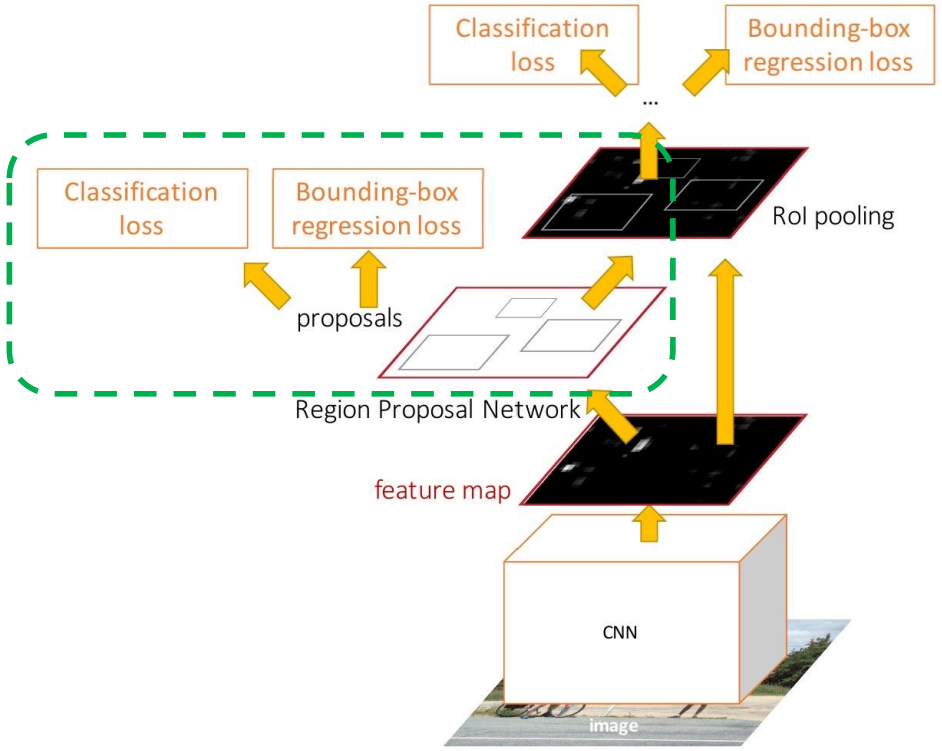
Faster R-CNN

Make CNN do proposals!

Insert **Region Proposal Network (RPN)** to predict proposals from features

Jointly train with 4 losses:

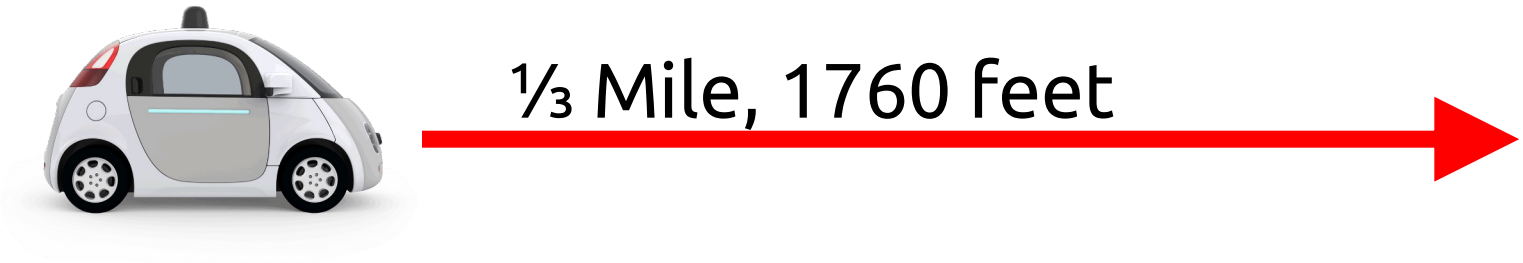
1. RPN classify object / not object
2. RPN regress box coordinates
3. Final classification score (object classes)
4. Final box coordinates



Ren et al, "Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks", NIPS 2015

Accurate object detection is slow!

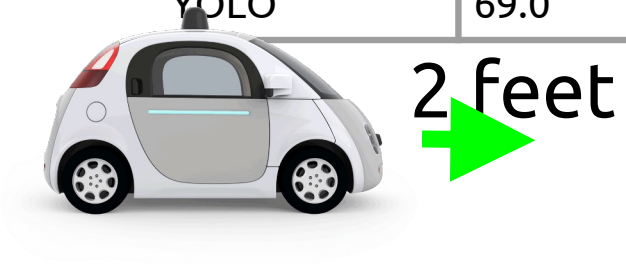
	Pascal 2007 mAP	Speed	
DPM v5	33.7	.07 FPS	14 s/img
R-CNN	66.0	.05 FPS	20 s/img



Redmon et al., "You Only Look Once: Unified, Real-Time Object Detection", CVPR 2016

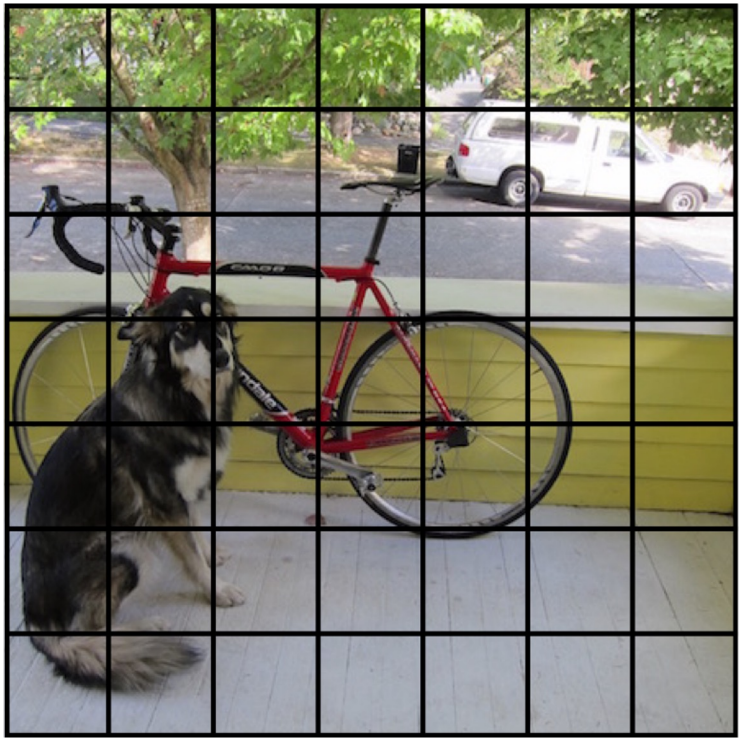
Accurate object detection is slow!

	Pascal 2007 mAP	Speed	
DPM v5	33.7	.07 FPS	14 s/img
R-CNN	66.0	.05 FPS	20 s/img
Fast R-CNN	70.0	.5 FPS	2 s/img
Faster R-CNN	73.2	7 FPS	140 ms/img
YOLO	69.0	45 FPS	22 ms/img



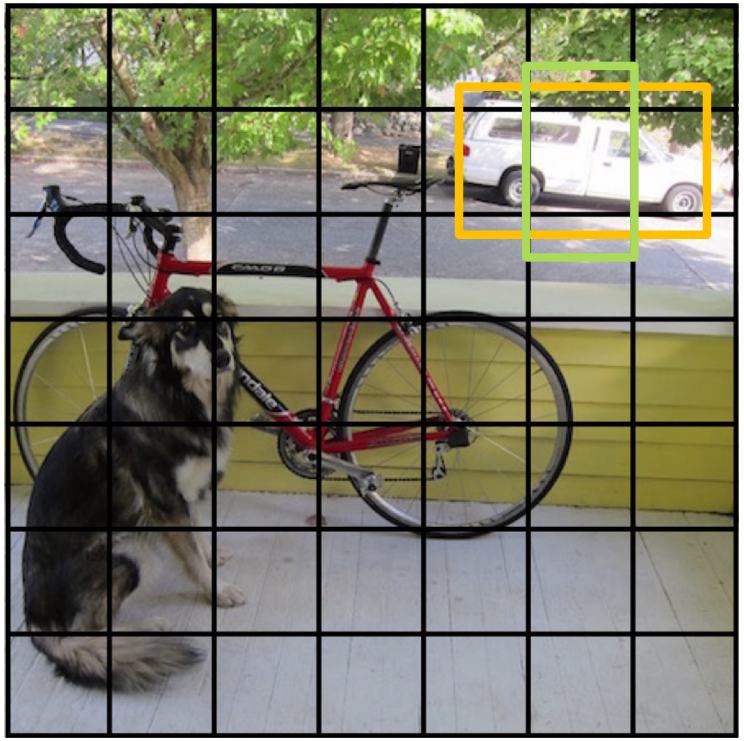
Redmon et al., "You Only Look Once: Unified, Real-Time Object Detection", CVPR 2016

Detection without Proposals: YOLO



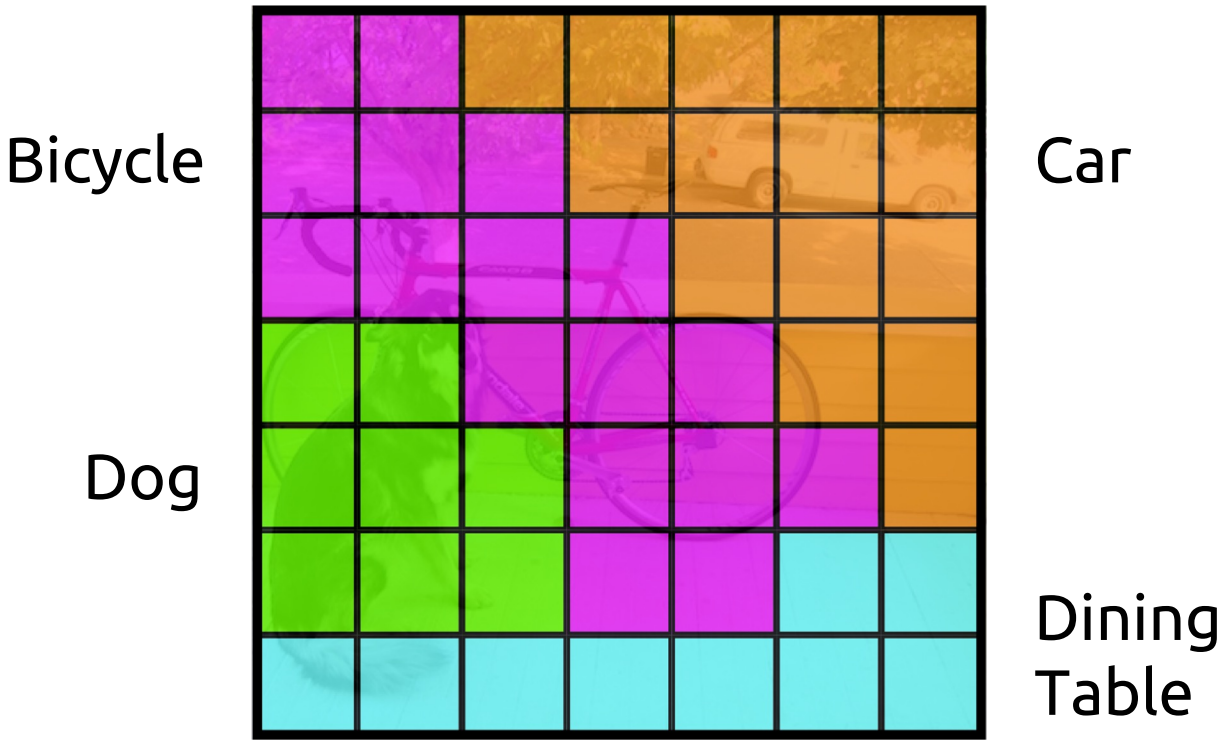
Redmon et al., "You Only Look Once: Unified, Real-Time Object Detection", CVPR 2016

Each cell predicts boxes and confidences: P(Object)



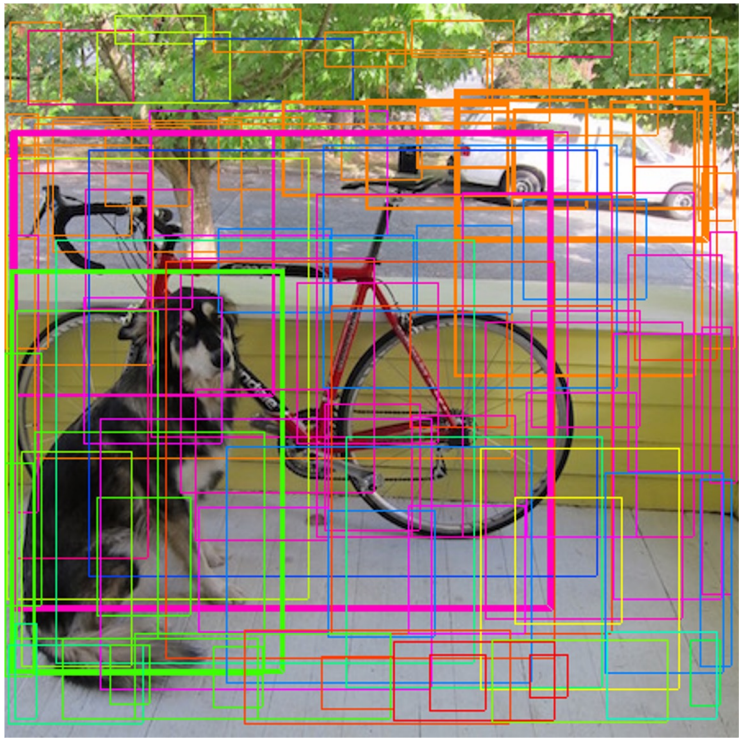
Redmon et al., "You Only Look Once: Unified, Real-Time Object Detection", CVPR 2016

Each cell also predicts a probability P(Class | Object)



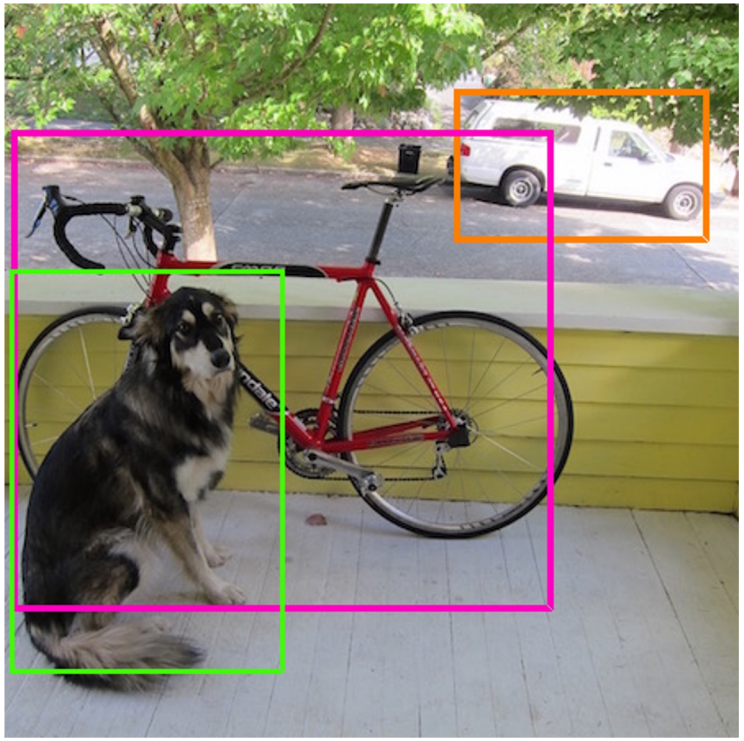
Redmon et al., "You Only Look Once: Unified, Real-Time Object Detection", CVPR 2016

Combine the box and class predictions



Redmon et al., "You Only Look Once: Unified, Real-Time Object Detection", CVPR 2016

Finally do NMS and threshold detections

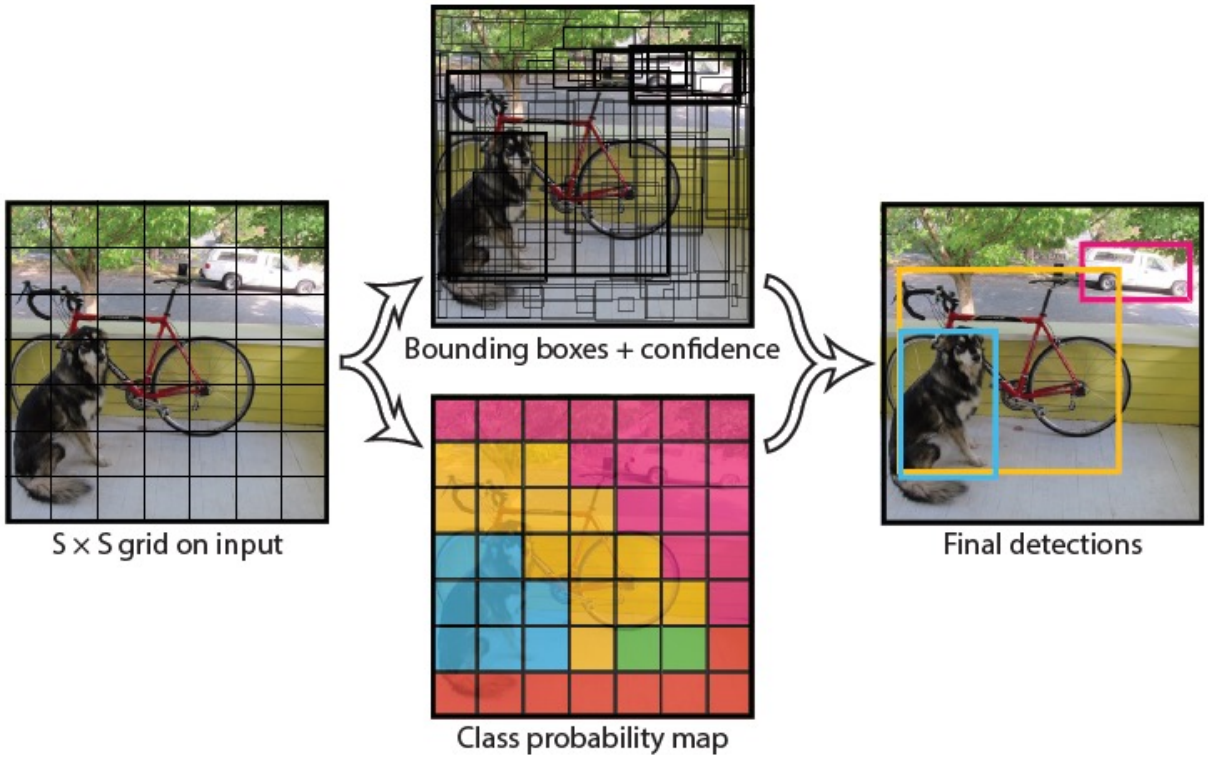


Redmon et al., “You Only Look Once: Unified, Real-Time Object Detection”, CVPR 2016

YOLO- You Only Look Once

Idea: No bounding box proposals.

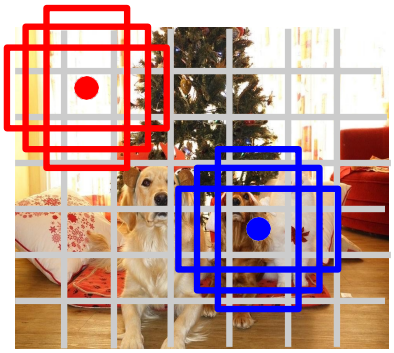
Predict a class and a box for every location in a grid.



https://arxiv.org/abs/1506.02640 Redmon et al. CVPR 2016.

Detection without Proposals: YOLO

Input image
3 x H x W



Divide image into grid
7 x 7

Image a set of **base boxes** centered at each grid cell
Here B = 3

Within each grid cell:

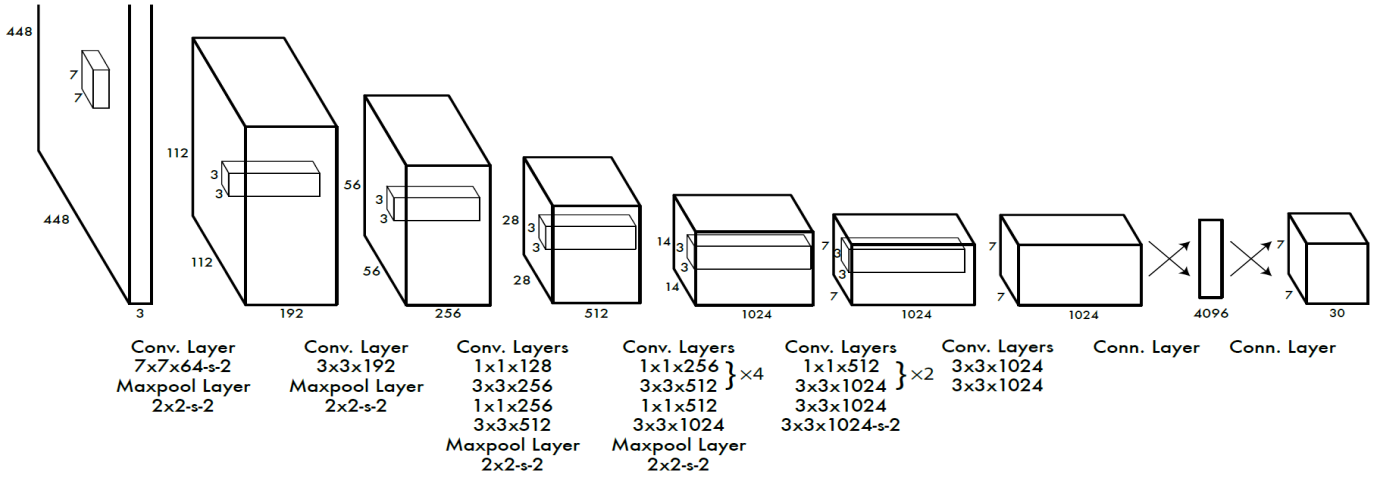
- Regress from each of the B base boxes to a final box with 5 numbers: (x, y, w, h, confidence)
- Predict scores for each of C classes (including background as a class)

Output:
7 x 7 x (5 * B + C)

Redmon et al, "You Only Look Once: Unified, Real-Time Object Detection", CVPR 2016
Liu et al, "SSD: Single-Shot MultiBox Detector", ECCV 2016

Slide by: Justin Johnson

YOLO- You Only Look Once



Divide the image into 7x7 cells.
Each cell trains a detector.
The detector needs to predict the object's class distributions.
The detector has 2 bounding-box predictors to predict bounding-boxes and confidence scores.

https://arxiv.org/abs/1506.02640 Redmon et al. CVPR 2016.

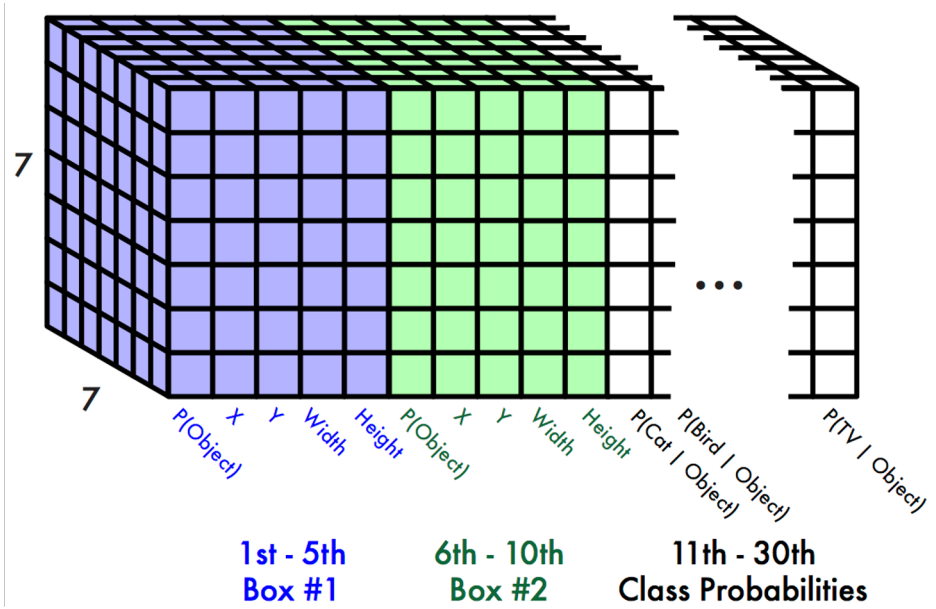
This parameterization fixes the output size

Each cell predicts:

- For each bounding box:
  - 4 coordinates (x, y, w, h)
  - 1 confidence value
- Some number of class probabilities

For Pascal VOC:

- 7x7 grid
- 2 bounding boxes / cell
- 20 classes



7 x 7 x (5 x 2 + 20) = 7 x 7 x 30 tensor = **1470 outputs**

Redmon et al., "You Only Look Once: Unified, Real-Time Object Detection", CVPR 2016

YOLO - Loss Function

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

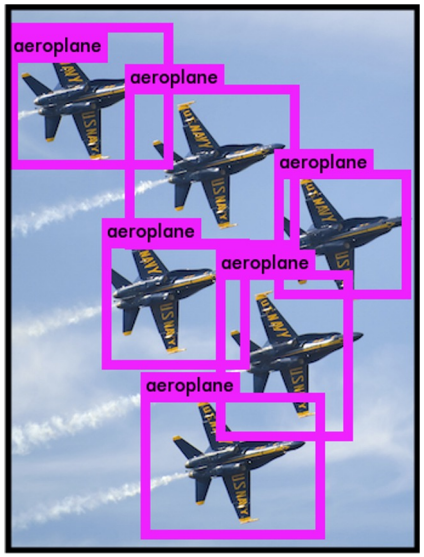
$$+ \lambda_{\text{coord}} \sum_{i=0}^{S^2} \sum_{j=0}^{B} \mathbb{1}_{ij}^{\text{obj}} \left[\left(\sqrt{w_i} - \sqrt{\hat{w}_i} \right)^2 + \left(\sqrt{h_i} - \sqrt{\hat{h}_i} \right)^2 \right]$$

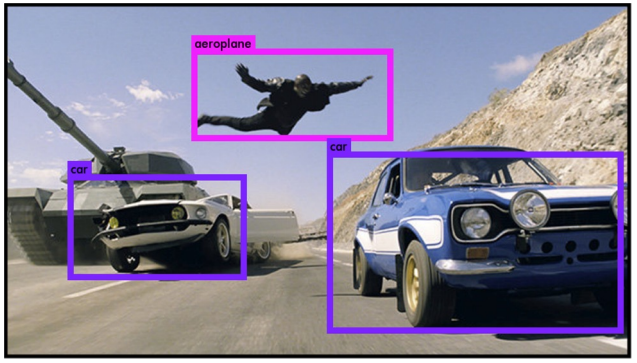
$$+ \sum_{i=0}^{S^2} \sum_{j=0}^{B} \mathbb{1}_{ij}^{\text{obj}} \left(C_i - \hat{C}_i \right)^2$$

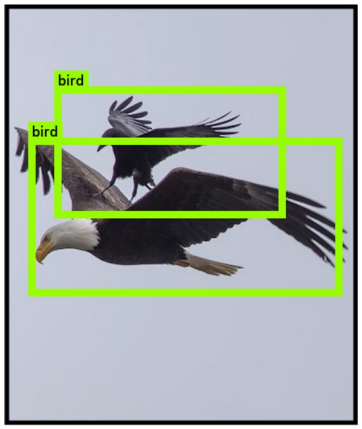
$$+ \lambda_{\text{noobj}} \sum_{i=0}^{S^2} \sum_{j=0}^{B} \mathbb{1}_{ij}^{\text{noobj}} \left(C_i - \hat{C}_i \right)^2$$

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

YOLO works across many natural images

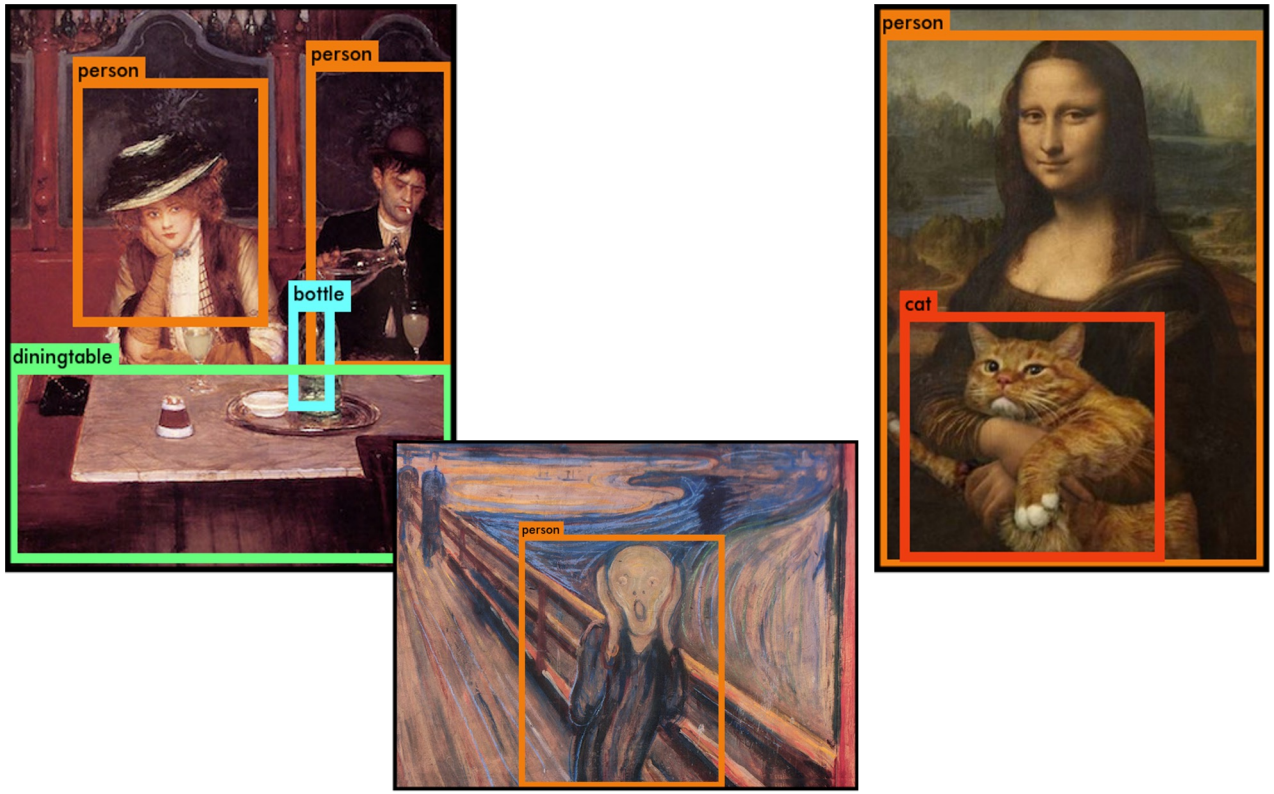






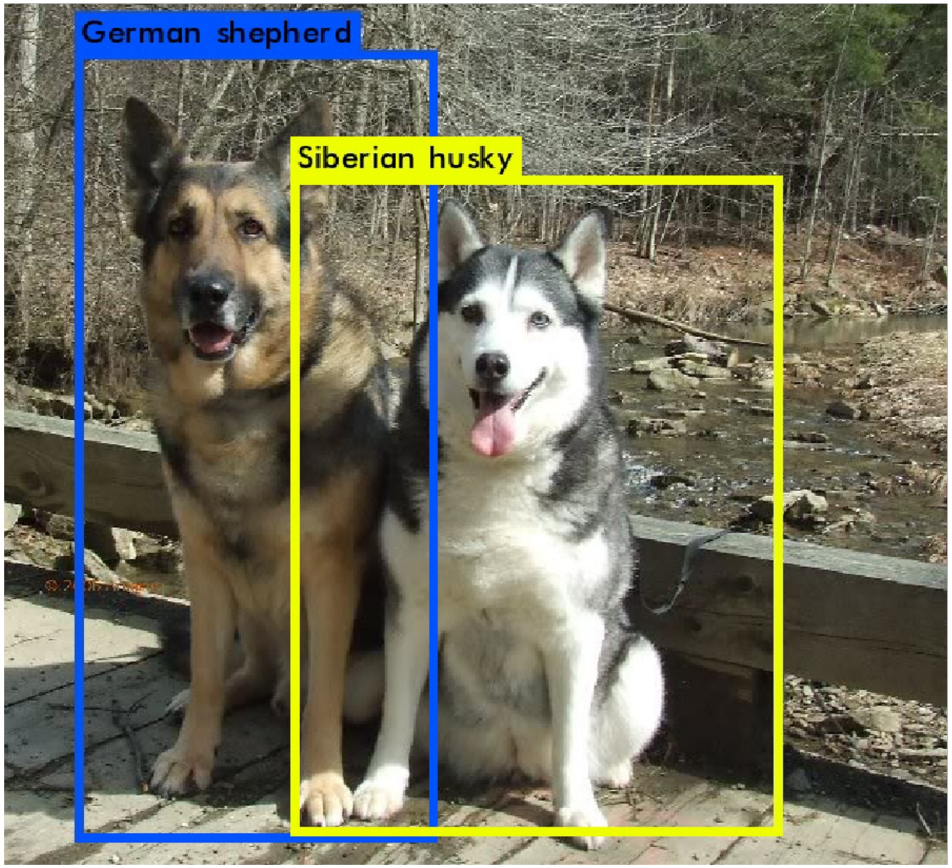
Redmon et al., "You Only Look Once: Unified, Real-Time Object Detection", CVPR 2016

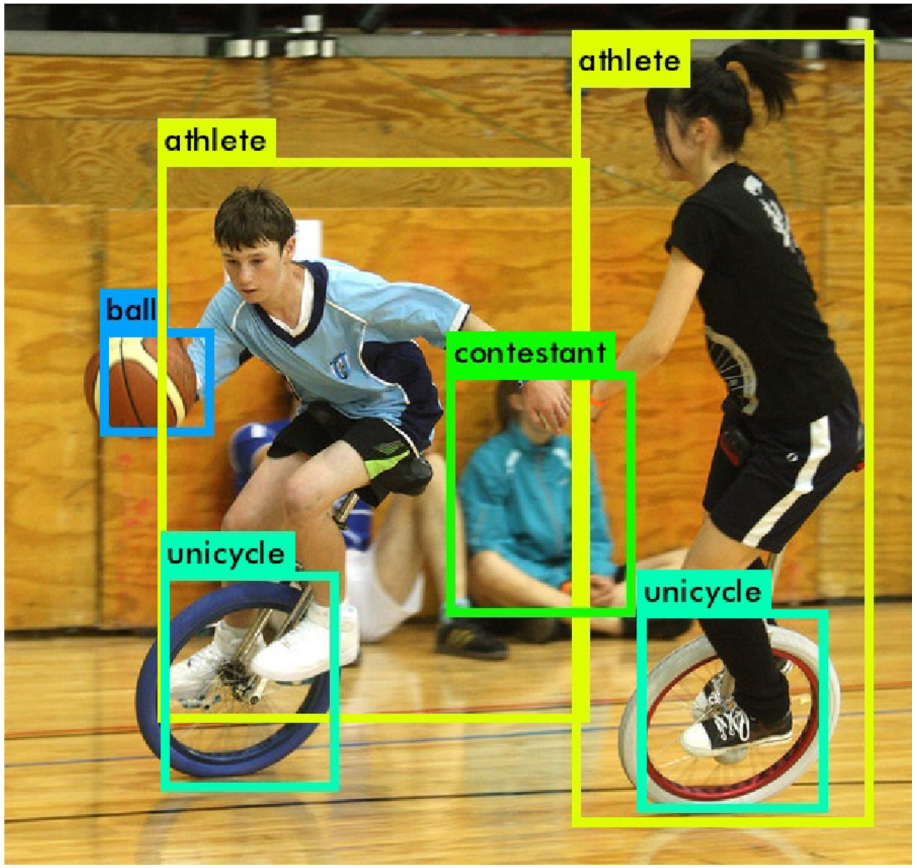
It also generalizes well to new domains



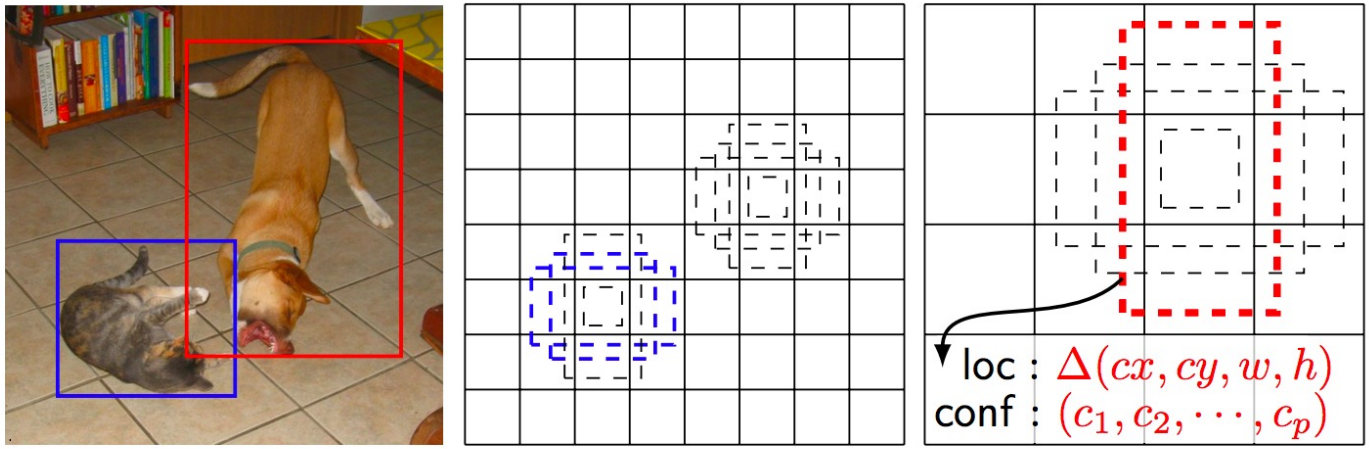
Redmon et al., "You Only Look Once: Unified, Real-Time Object Detection", CVPR 2016

Redmon and Farhadi, "YOLO9000: Better, Faster, Stronger", CVPR 2017


German shepherd
Siberian husky


athlete
athlete
ball
contestant
unicycle
unicycle

SSD: Single Shot Detector



(a) Image with GT boxes (b) 8×8 feature map (c) 4×4 feature map

Lab 8a

Idea: Similar to YOLO, but denser grid map, multiscale grid m aps. + Data augmentation + Hard negative mining + Other de sign choices in the network.

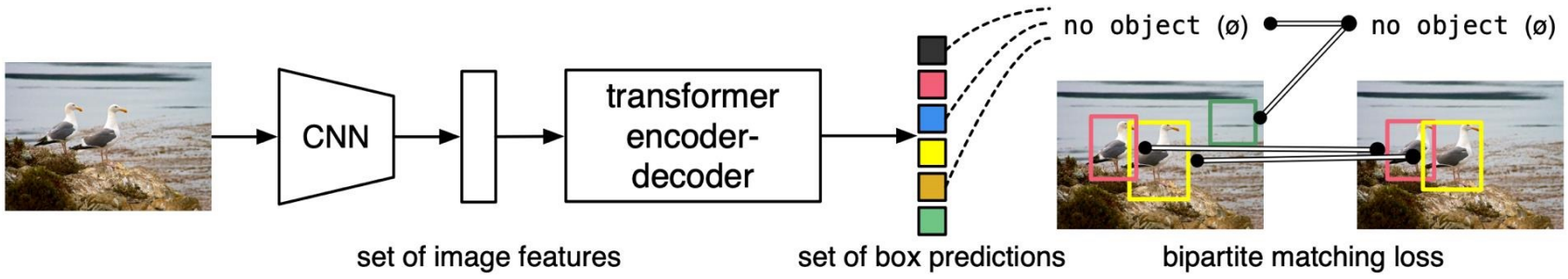
Liu et al. ECCV 2016.

Object **De**tection with **Tr**ansformers: DETR

Simple object detection pipeline: directly output a set of boxes from a Transformer

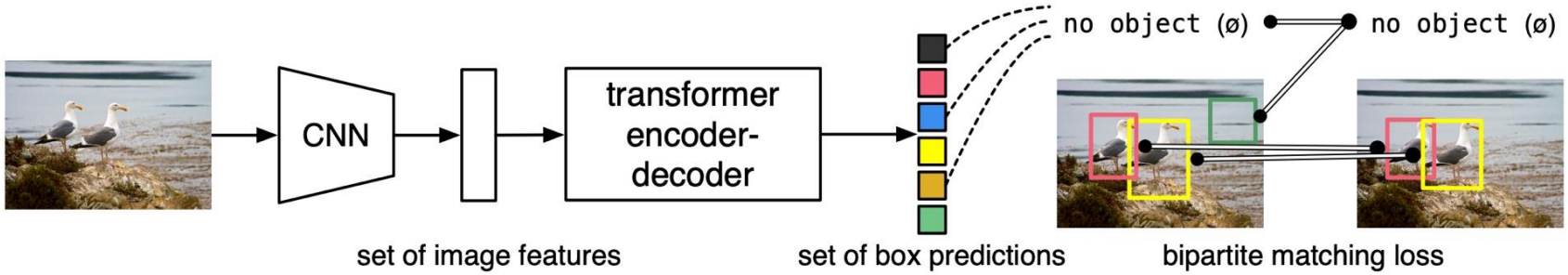
No anchors, no regression of box transforms

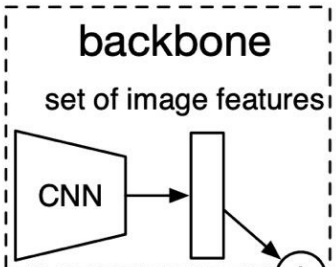
Match predicted boxes to GT boxes with bipartite matching; train to regress box c oordinates



Carion et al, “End-to-End Object Detection with Transformers”, ECCV 2020

Object **De**tection with **Tr**ansformers: DETR





Carion et al, “End-to-End Object Detection with Transformers”, ECCV 2020

Additional Resources

- TensorFlow Detection API [link]
  Faster RCNN, SSD, RFCN, Mask R-CNN, ...

TensorFlow

- Detectron2 (PyTorch) [link]
  Mask R-CNN, RetinaNet, Faster R-CNN, RPN, Fast R-CNN, R-FCN, ...

Detectron2

Plan for the next few lectures

- Detection approaches
  - Pre-CNNs
    - Detection with whole windows: Pedestrian detection
    - Part-based detection: Deformable Part Models
  - Post-CNNs
    - Detection with region proposals: R-CNN, Fast R-CNN, Faster-R-CNN
    - Detection without region proposals: YOLO, SSD, DETR
- Learning from noisy web image-text data
  - Contrastive Language-Image Pretraining (CLIP)
    - Prompting
  - Open-vocabulary object detection
- Segmentation approaches
  - Semantic segmentation
    - Fully-Convolutional Networks (FCN)
  - Instance segmentation
    - Mask R-CNN
    - Segment Anything

Learning from noisy web data

- Massive datasets of image-text pairs from the web
  - E.g. alt text, Flickr, Reddit, Wikipedia, etc
- Images and their co-occurring text assumed related (text provides a reasonable description of image?)
- Train text and image feature extractors using the objective that matched (co-occurring) image-text should be more similar than mismatched ones
- Great performance at a low annotation cost (data already existed)

Contrastive Language-Image Pretraining (CLIP)

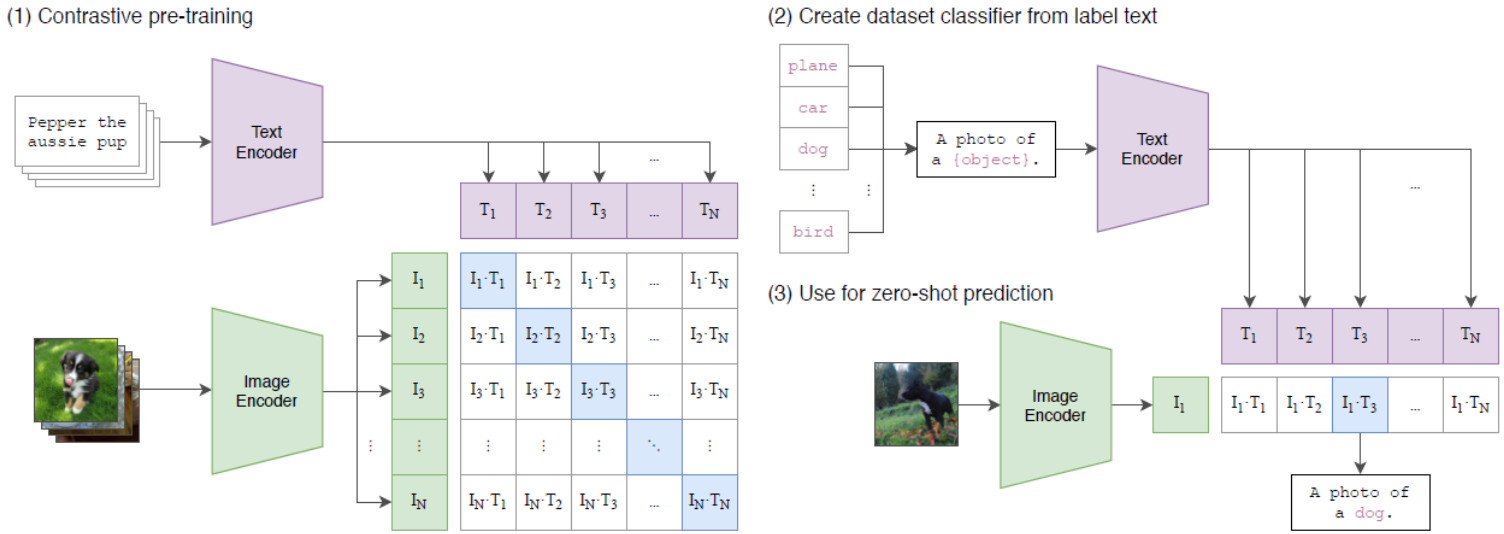
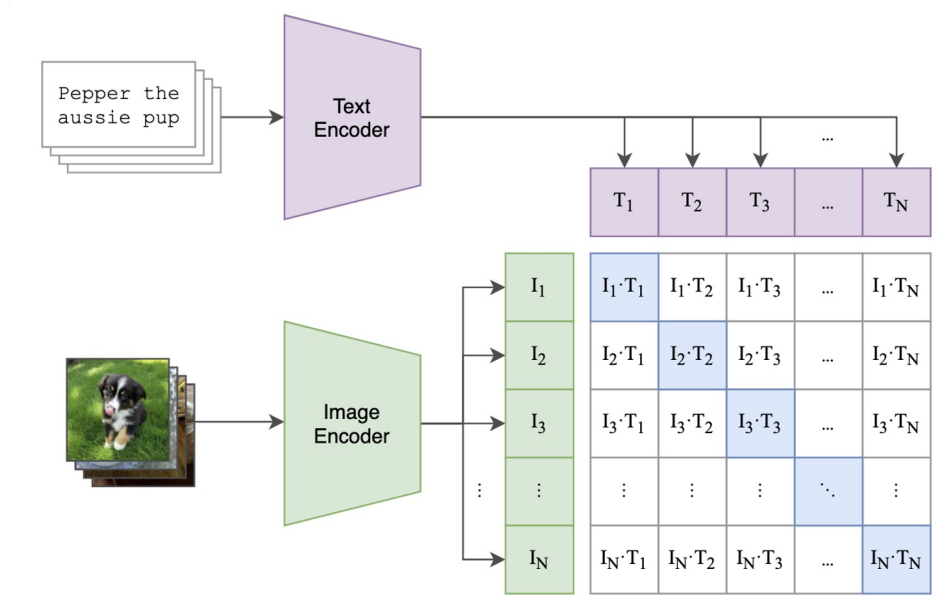


Figure 1. Summary of our approach. While standard image models jointly train an image feature extractor and a linear classifier to predict some label, CLIP jointly trains an image encoder and a text encoder to predict the correct pairings of a batch of (image, text) training examples. At test time the learned text encoder synthesizes a zero-shot linear classifier by embedding the names or descriptions of the target dataset's classes.

Radford et al. "Learning Transferable Visual Models From Natural Language Supervision." ICML 2021.

Contrastive Language-Image Pretraining (CLIP)



$$L = \sum_k \ell(I_k T_k)$$

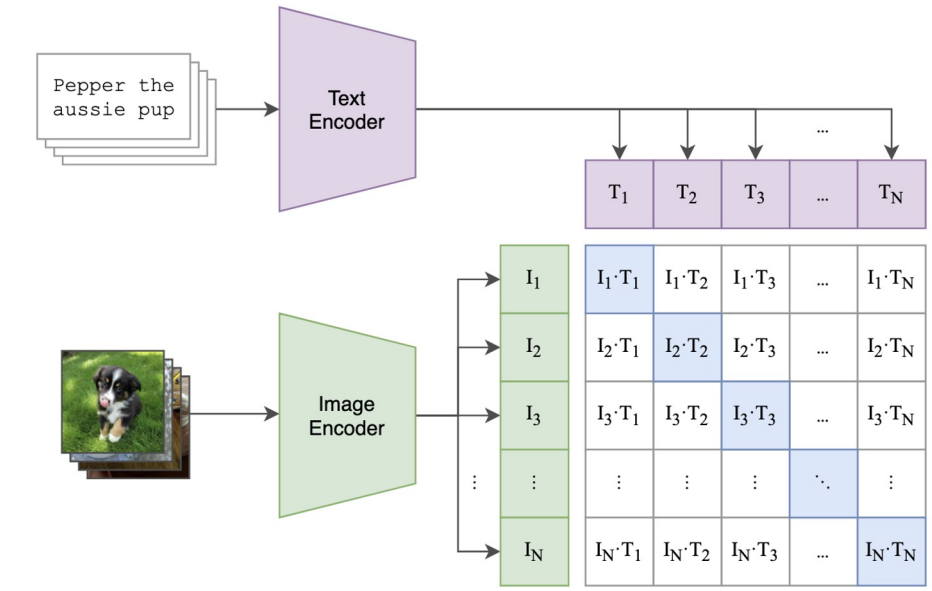
Softmax (cross-entropy) loss

$$\ell(I_k T_k) = -\log\left(\frac{\exp(sim(I_k, T_k))}{\sum_{t=1}^{2N} 1[k \neq i]\exp(sim(I_k, T_t))}\right)$$

Radford et al. "Learning Transferable Visual Models From Natural Language Supervision." ICML 2021.

Adapted from Vicente Ordoñez

Contrastive Language-Image Pretraining (CLIP)



$$L = \sum_k \ell_1(I_k T_k) + \ell_2(I_k T_k)$$

Softmax (cross-entropy) loss

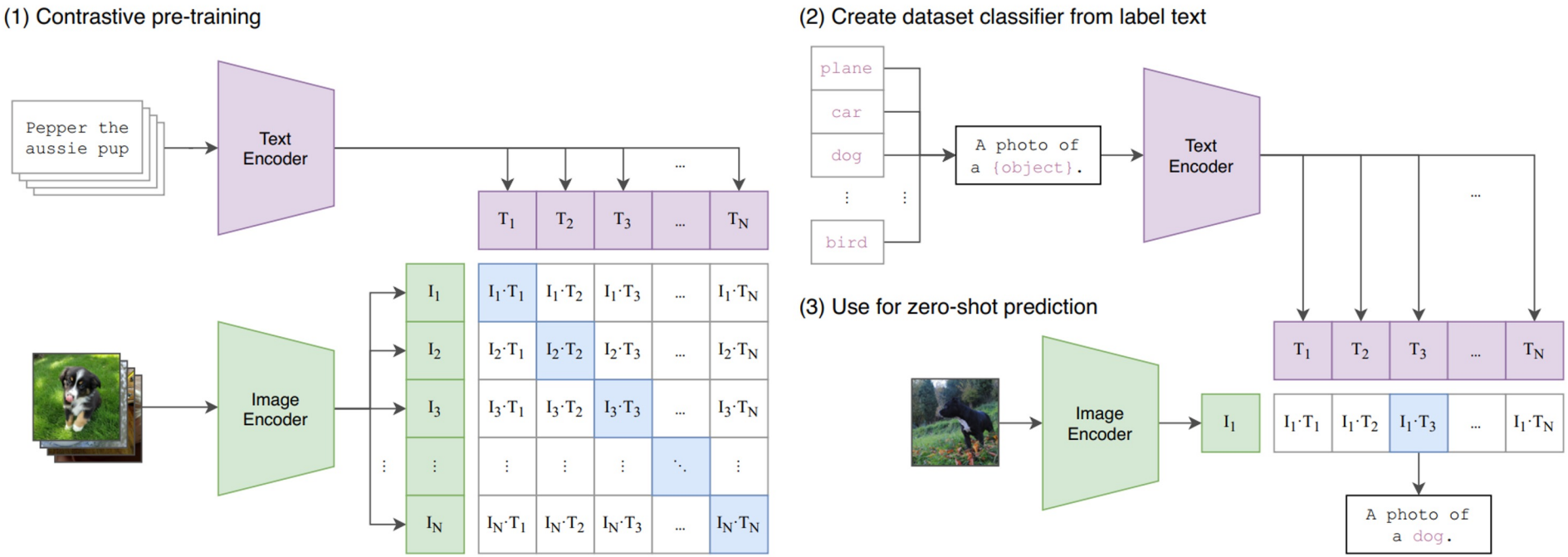
$$\ell_1(I_k T_k) = -\log\left(\frac{\exp(sim(I_k, T_k))}{\sum_{t=1}^{2N} 1[k \neq i]\exp(sim(I_k, T_t))}\right)$$

$$\ell_2(I_k T_k) = -\log\left(\frac{\exp(sim(I_k, T_k))}{\sum_{t=1}^{2N} 1[k \neq i]\exp(sim(I_t, T_k))}\right)$$

Radford et al. "Learning Transferable Visual Models From Natural Language Supervision." ICML 2021.

Adapted from Vicente Ordoñez

Zero-shot Image Classification with CLIP



Radford et al. "Learning Transferable Visual Models From Natural Language Supervision." ICML 2021.

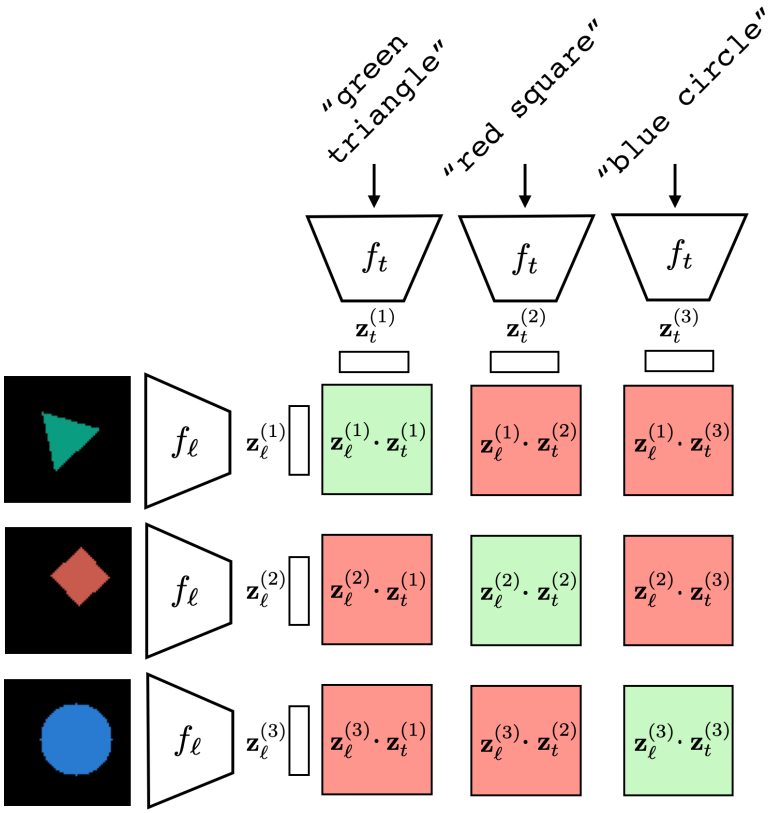
Adapted from Vicente Ordoñez

Zero-shot Image Classification with CLIP

- Image classification: given an image, predict its class name
- Image captioning: given an image, predict its caption
- Contrastive learning: align image and text embeddings that describe the same thing

[Radford*, Kim* et al., ICML 2021]

Zero-shot Image Classification with CLIP

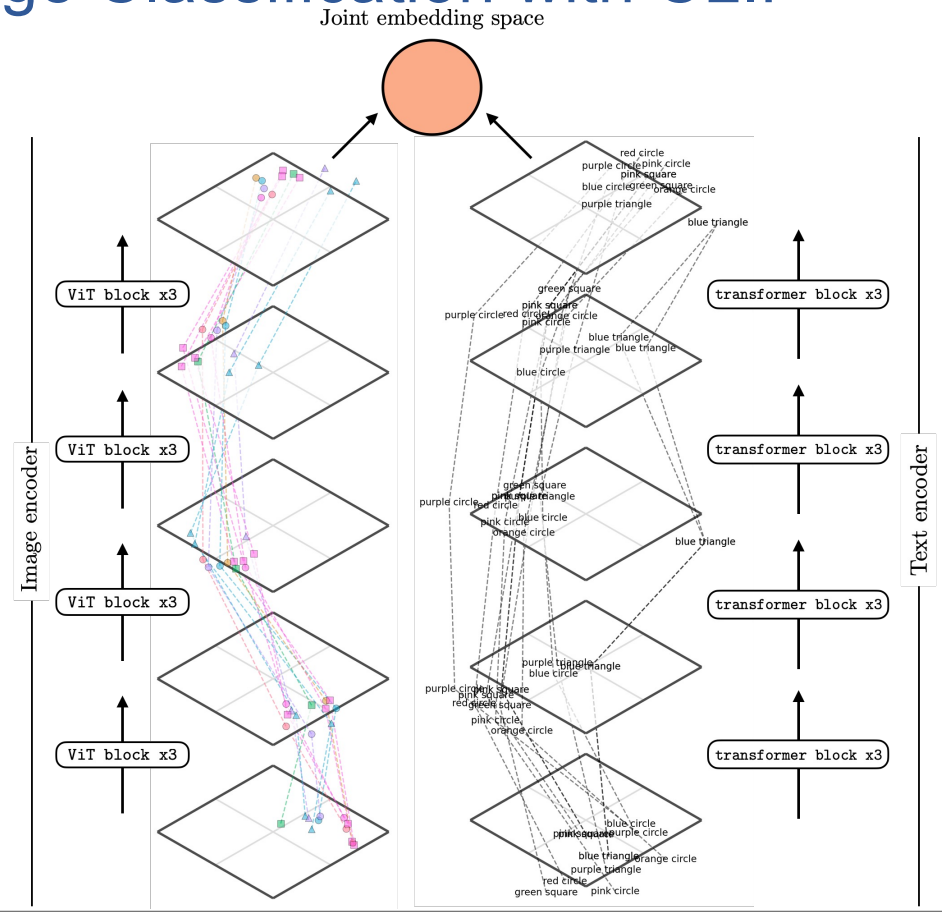


- Positive pairs: image and its caption.
- Negative pairs: image and a different image's caption.
- Learn a representation in which positives are pulled together, negatives are pushed apart.

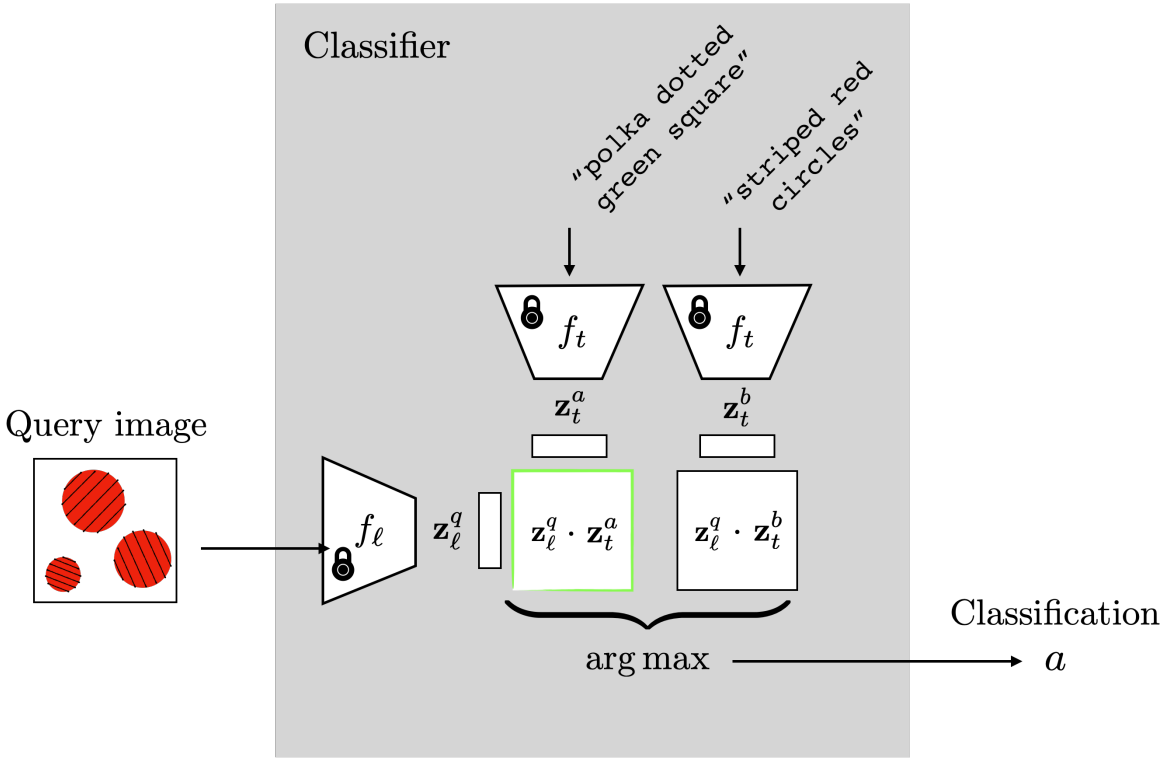
We learn this representation using Contrastive Learning

[Radford*, Kim* et al., ICML 2021]

Zero-shot Image Classification with CLIP



Zero-shot Image Classification with CLIP



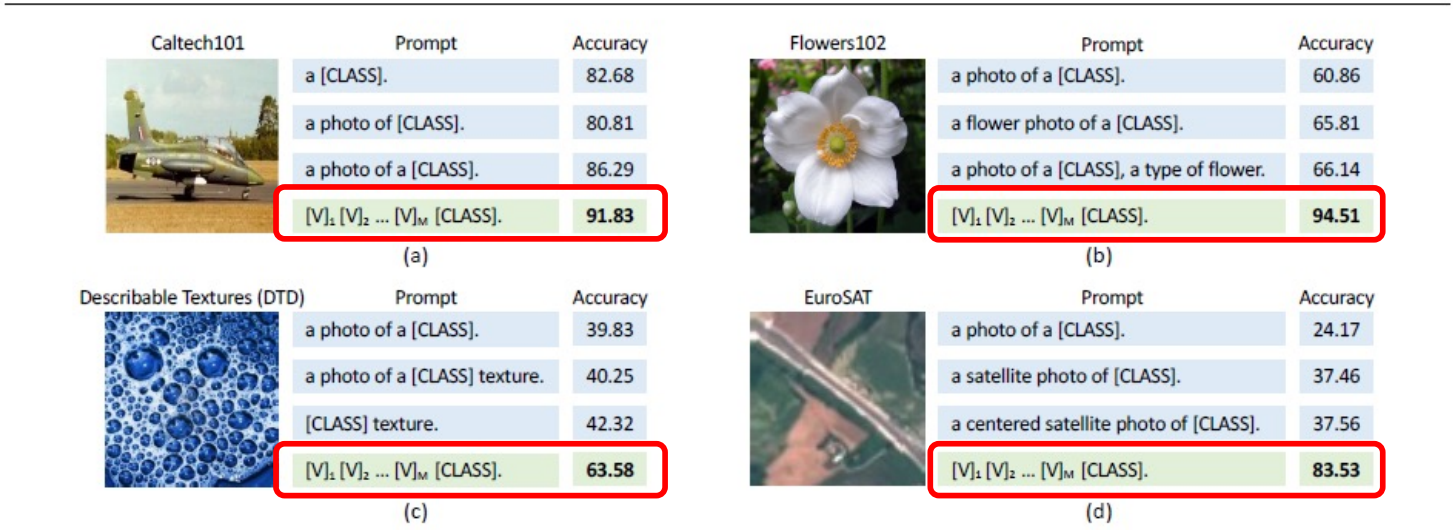
Using CLIP for Object Recognition

- Compute dot product of image and prompt for each class (“A photo of <class>”), e.g. “A photo of dog”
- Return class with highest dot product for each image
- Prompt can be optimized manually or through training
- Can extend idea for object detection

Variants of CLIP

- CLIP: https://github.com/openai/CLIP
- OpenCLIP: https://github.com/mlfoundations/open_clip
- MetaCLIP: https://github.com/facebookresearch/MetaCLIP
- CLIPA: https://github.com/UCSC-VLAA/CLIPA
- SigLIP: https://github.com/merveenoyan/siglip
- DFN-5b: https://huggingface.co/apple/DFN5B-CLIP-ViT-H-14-378

Using CLIP for Object Recognition



Caltech101

Prompt	Accuracy
a [CLASS].	82.68
a photo of [CLASS].	80.81
a photo of a [CLASS].	86.29
$[V]_1$ $[V]_2$ ... $[V]_M$ [CLASS].	**91.83**

(a)

Flowers102

Prompt	Accuracy
a photo of a [CLASS].	60.86
a flower photo of a [CLASS].	65.81
a photo of a [CLASS], a type of flower.	66.14
$[V]_1$ $[V]_2$ ... $[V]_M$ [CLASS].	**94.51**

(b)

Describable Textures (DTD)

Prompt	Accuracy
a photo of a [CLASS].	39.83
a photo of a [CLASS] texture.	40.25
[CLASS] texture.	42.32
$[V]_1$ $[V]_2$ ... $[V]_M$ [CLASS].	**63.58**

(c)

EuroSAT

Prompt	Accuracy
a photo of a [CLASS].	24.17
a satellite photo of [CLASS].	37.46
a centered satellite photo of [CLASS].	37.56
$[V]_1$ $[V]_2$ ... $[V]_M$ [CLASS].	**83.53**

(d)

Fig. 1 Prompt engineering vs Context Optimization (CoOp). The former needs to use a held-out validation set for words tuning, which is inefficient; the latter automates the process and requires only a few labeled images for learning.

Zhou et al. "Learning to Prompt for Vision-Language Models." IJCV 2022.

Using CLIP for Object Recognition

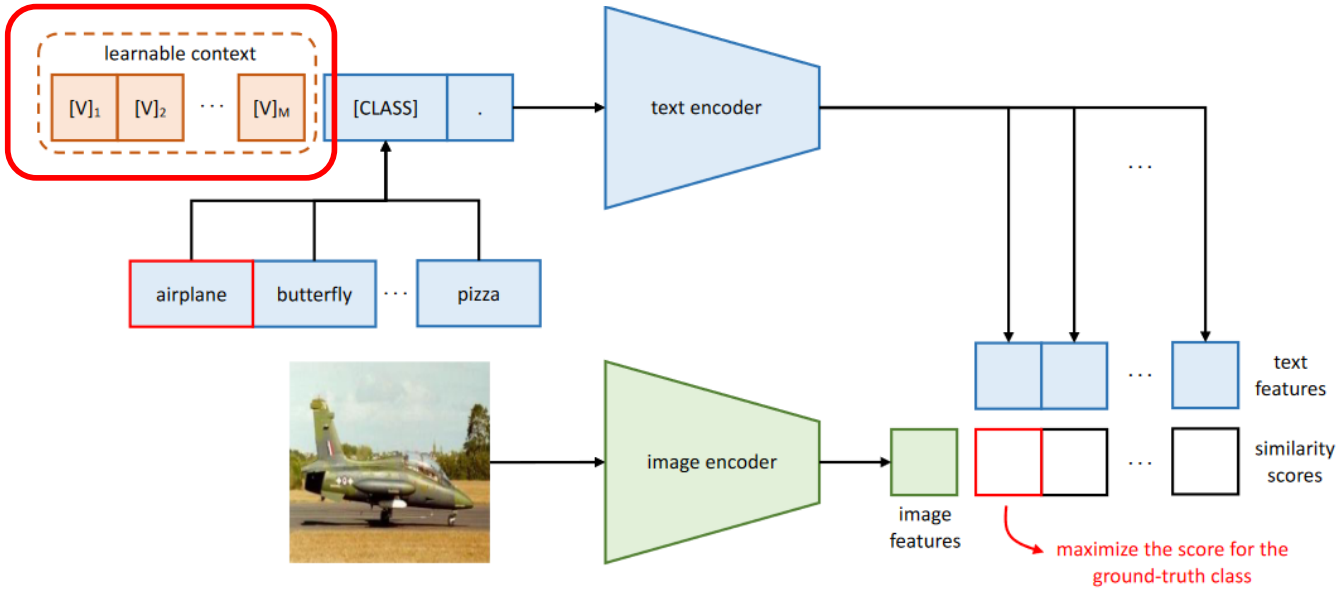


Fig. 2 Overview of Context Optimization (CoOp). The main idea is to model a prompt's context using a set of learnable vectors, which can be optimized through minimizing the classification loss. Two designs are proposed: one is unified context, which shares the same context vectors with all classes; and the other is class-specific context, which learns for each class a specific set of context vectors.

Zhou et al. "Learning to Prompt for Vision-Language Models." IJCV 2022.

Using CLIP for Object Recognition

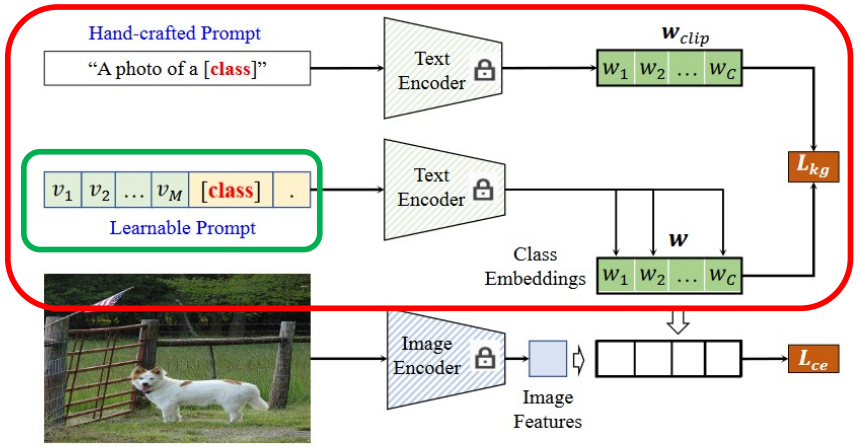


Figure 2. The framework of the Knowledge-guided Context Optimization for prompt tuning. $\mathcal{L}_{ce}$ is the standard cross-entropy loss, and $\mathcal{L}_{kg}$ is the proposed Knowledge-guided Context Optimization contraint to minimize the discrepancy between the special knowledge (learnable textual embeddings) and the general knowledge(the textual embeddings generated by the hand-crafted prompt).

degradation. Therefore, we can minimize the distance between $\mathbf{w}_i$ and $\mathbf{w}_i^{clip}$ for boosting the generability of the unseen classes,

$$\mathcal{L}_{kg} = \frac{1}{N_c}\sum_{i=1}^{N_c} ||\mathbf{w}_i - \mathbf{w}_i^{clip}||_2^2, \tag{3}$$

where $||\cdot||$ is the euclidean distance, N_c is the number of seen classes. Meanwhile, the standard contrastive loss is:

$$\mathcal{L}_{ce} = -\sum_{\mathbf{x}\in\mathbf{X}} \log \frac{\exp(d(\mathbf{x}, \mathbf{w}_y)/\tau)}{\sum_{i=1}^{N_c}\exp(d(\mathbf{x}, \mathbf{w}_i)/\tau)}, \tag{4}$$

where y is the corresponding label of the image embedding.

By combining the standard cross-entropy loss $\mathcal{L}_{ce}$, the final objective is:

$$\mathcal{L} = \mathcal{L}_{ce} + \lambda\mathcal{L}_{kg}, \tag{5}$$

where λ is used balance the effect of $\mathcal{L}_{kg}$.

Yao et al. "Visual-Language Prompt Tuning with Knowledge-guided Context Optimization." CVPR 2023.

What Objects do you see?

Source: Instagram

What Objects do you see?

https://faculty.cc.gatech.edu/~zk15/teaching/AY2025_cs8803vlm_fall/

Source: Instagram

What Objects do you see? Now you can only choose from one of the COCO dataset labels

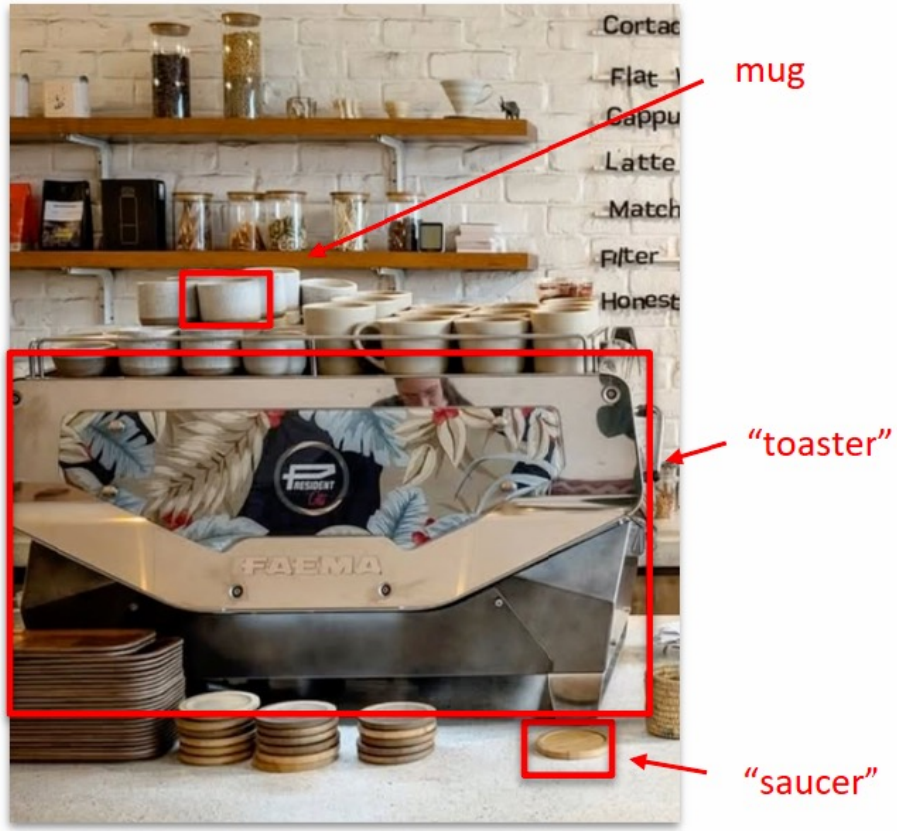
Sample COCO labels

person	wine glass	toaster
bicycle	cup	sink
car	fork	refrigerator
motorcycle	knife	book
airplane	spoon	clock
bus	bowl	vase
train	banana	scissors
truck	apple	teddy bear
boat	sandwich	hair drier
traffic light	orange	toothbrush

https://faculty.cc.gatech.edu/~zk15/teaching/AY2025_cs8803vlm_fall/

Source: Instagram
COCO - Common Objects in Context

[Problem] Object Detection in the Real World



Source: Instagram

Closed-Vocabulary Detection Problem:

- Models (e.g., COCO, LVIS) are trained on a **fixed set** of categories (80, 1,200, etc.)
- **Out-of-vocabulary** objects are either ignored or misclassified
- Scaling to cover "every object in the world" with manual labels is **impossible**

Need: An object detector that:

- Works with **natural language labels** (no fixed class list)
- Generalizes to **unseen categories** without retraining
- Retains **competitive** performance on **known** categories

https://faculty.cc.gatech.edu/~zk15/teaching/AY2025_cs8803vlm_fall/

[Solution 1] Using CLIP for Object Recognition [Open Vocab]

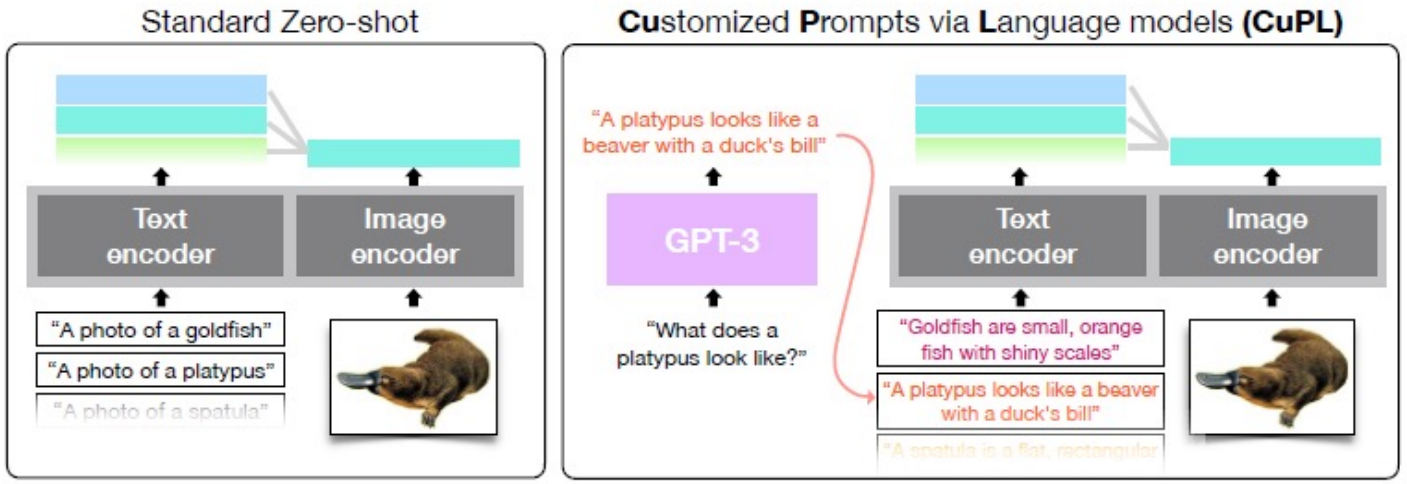


Figure 1: **Schematic of the method.** (Left) The standard method of a zero-shot open vocabulary image classification model (e.g., CLIP (Radford et al., 2021)). (Right) Our method of CuPL. First, an LLM generates descriptive captions for given class categories. Next, an open vocabulary model uses these captions as prompts for performing classification.

Pratt et al. "What does a platypus look like? Generating customized prompts for zero-shot image classification." ICCV 2023.

[Solution 1] Using CLIP for Object Recognition [Open Vocab]

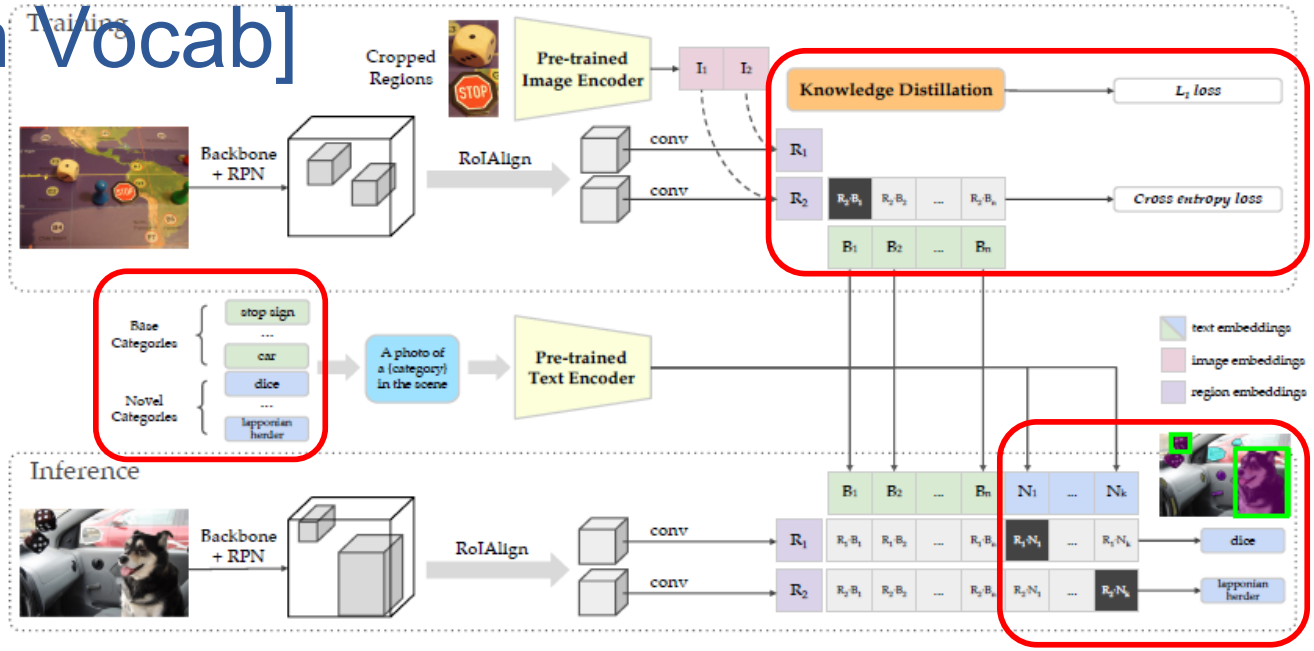


Figure 2: **An overview of using ViLD for open-vocabulary object detection.** ViLD distills the knowledge from a pretrained open-vocabulary image classification model. First, the category text embeddings and the image embeddings of cropped object proposals are computed, using the text and image encoders in the pretrained classification model. Then, ViLD employs the text embeddings as the region classifier (ViLD-text) and minimizes the distance between the region embedding and the image embedding for each proposal (ViLD-image). During inference, text embeddings of novel categories are used to enable open-vocabulary detection.

Gu et al. "Open-vocabulary Object Detection via Vision and Language Knowledge Distillation." ICLR 2021.

[Solution 2] OWL-ViT: Vision Transformer for Open-World Localization

Simple Open-Vocabulary Object Detection with Vision Transformers

Matthias Minderer*, Alexey Gritsenko*,
Austin Stone, Maxim Neumann, Dirk Weissenborn, Alexey Dosovitskiy,
Aravindh Mahendran, Anurag Arnab, Mostafa Dehghani, Zhuoran Shen,
Xiao Wang, Xiaohua Zhai, Thomas Kipf, and Neil Houlsby

Google Research
{mjlm,agritsenko}@google.com

Keywords: open-vocabulary detection, transformer, vision transformer, zero-shot detection, image-conditioned detection, one-shot object detection, contrastive learning, image-text models, foundation models, CLIP

Contributions

- **Open-Vocabulary Detection**: detects objects described in text, not limited to training labels.
- **Zero-Shot Generalization**: finds novel categories without retraining (e.g., “espresso machine”).
- **Simplicity + Scaling**: large-scale pre-training + ViT + end-to-end fine-tuning outperforms more complex architectures

https://faculty.cc.gatech.edu/~zk15/teaching/AY2025_cs8803vlm_fall/

[Approach] : Two Stages: Large-Scale Pre-Training + Detection Fine-Tuning

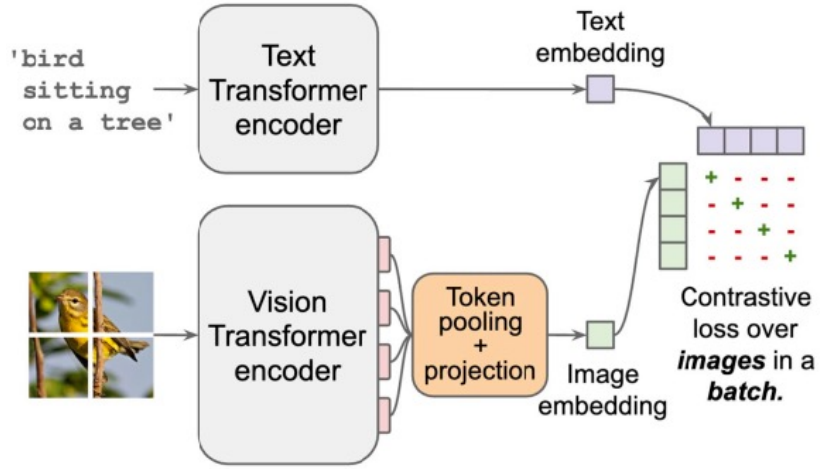


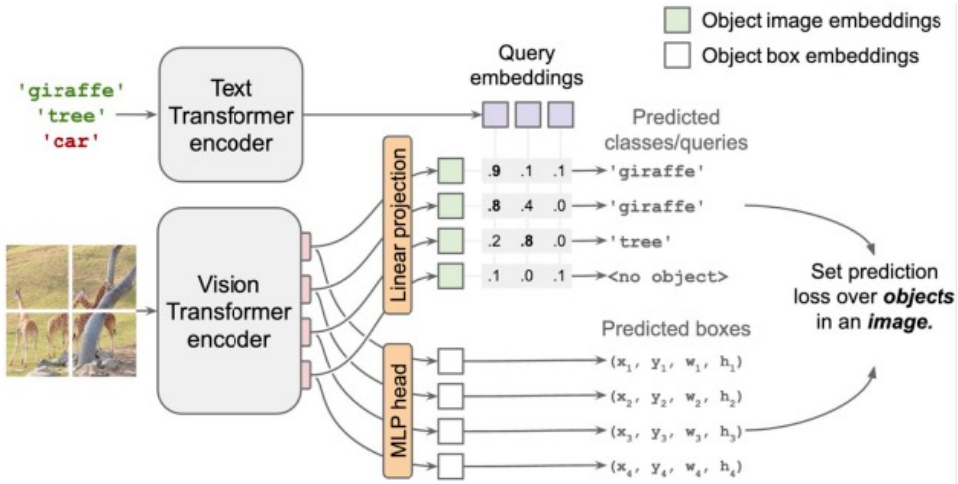
Image-level Contrastive Pre-training

Stage 1: Contrastively pre-train image and text encoders on large-scale image-text data

- Vision:
  Model: ViT: [B]ase, [L]arge, [H]uge / 16-32 (patch size); R50+H – ResNet50 + ViT H[uge]
- Text: Transformer with 12 layers & 8 heads
- Data: 3.6 billion image-text pairs; batch size 256
- Both Text and Image encoders are trained from scratch

https://faculty.cc.gatech.edu/~zk15/teaching/AY2025_cs8803vlm_fall/

[Approach] : Two Stages: Large-Scale Pre-Training + Detection Fine-Tuning



Transfer to Open-Vocabulary Detection

Stage 2: Add Detection Heads and fine-tune on medium-sized detection data

- **Text**: Text encoder from CLIP is retained. At inference, user supplies text query
- **Vision**:
  - Remove token pooling + projection layer
  - Linearly project each output token to obtain per-object image embeddings
  - Box coordinates come from a separate MLP head
- **Data**: Medium-scale detection datasets (e.g., LVis, COCO, Objects365)
- **Text** encoder is **frozen**; we're only retraining the ViT

https://faculty.cc.gatech.edu/~zk15/teaching/AY2025_cs8803vlm_fall/

[Data] : LVIS – Test-bed for RARE (“unseen”) categories

LVIS: A Dataset for Large Vocabulary Instance Segmentation

Agrim Gupta Piotr Dollár Ross Girshick
Facebook AI Research (FAIR)

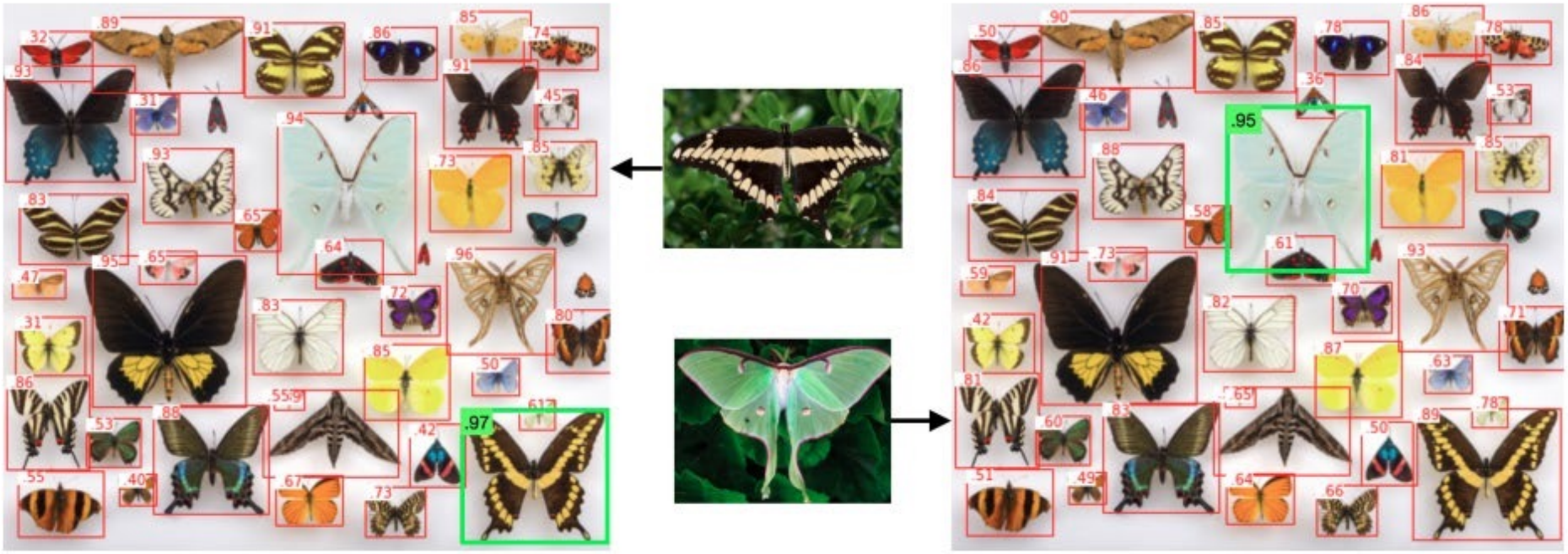
Figure 1. **Example annotations.** We present **LVIS**, a new dataset for benchmarking Large Vocabulary Instance Segmentation in the 1000+ category regime with a challenging long tail of rare objects.

LVIS: A Dataset for Large Vocabulary Instance Segmentation [arxiv]

https://faculty.cc.gatech.edu/~zk15/teaching/AY2025_cs8803vlm_fall/

[Results] Image-Conditioned Detection Performance

OWL-ViT strongly outperforms the best task-specific models by a 72% margin

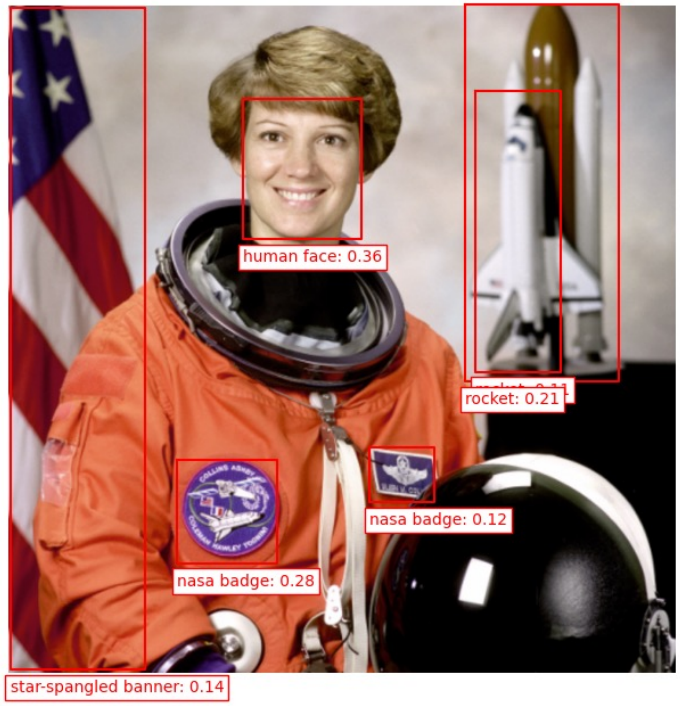


Idea: Use image embeddings (instead of text) to “query” the input image and find most relevant objects

https://faculty.cc.gatech.edu/~zk15/teaching/AY2025_cs8803vlm_fall/

[Live] OWL-ViT: open-vocabulary object detector

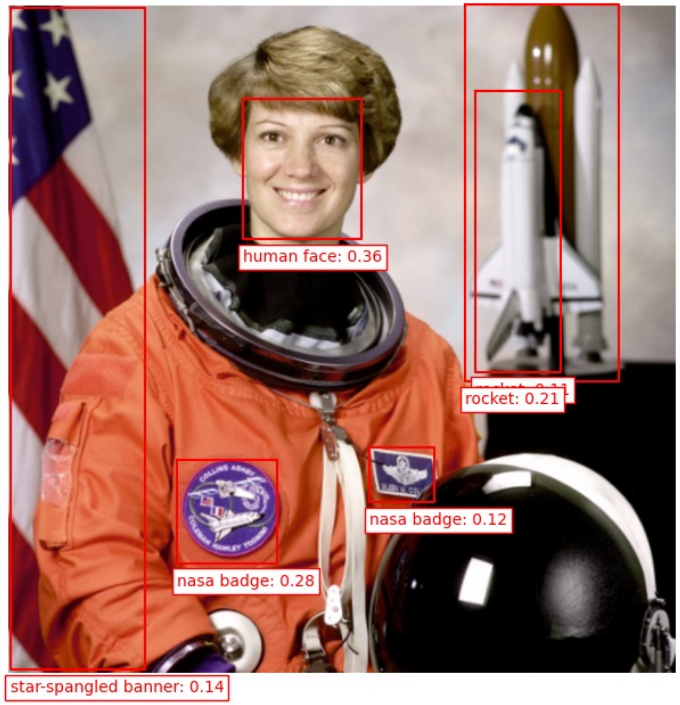
```
text_queries = ["human face", "rocket",
 "nasa badge", "star-spangled banner"]
```



Lab 8b: Open Vocabulary Object Recognition

Duration: 5 min

[Live] OWL-ViT: open-vocabulary object detector

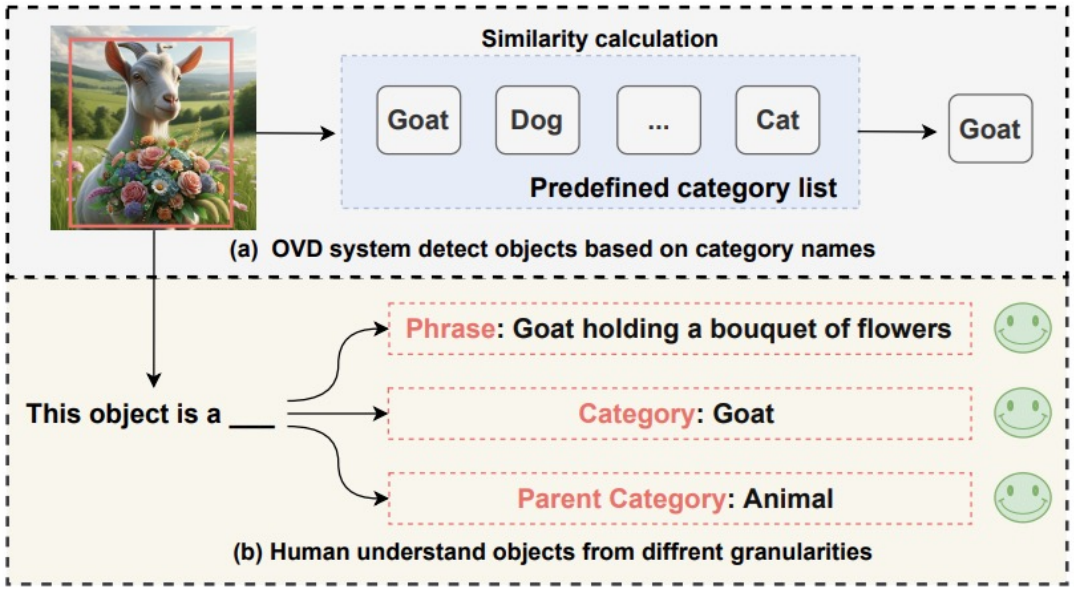


How Bounding Boxes are encoded?

DetCLIP-v3: Towards Versatile Generative Open-Vocabulary

- Existing OVD models are limited by their reliance on a predefined object category list, which hinders their usage in practical scenarios.

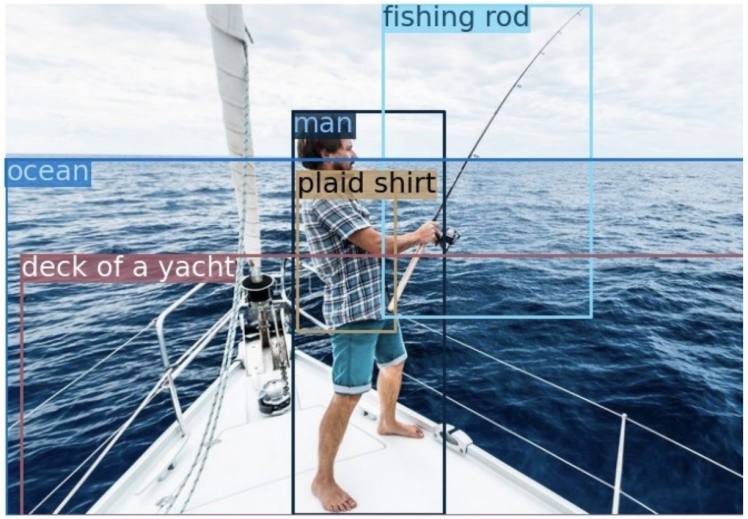
- In contrast, human cognition demonstrates much more versatility. For example, humans are able to understand objects from different granularities, in a hierarchical manner.

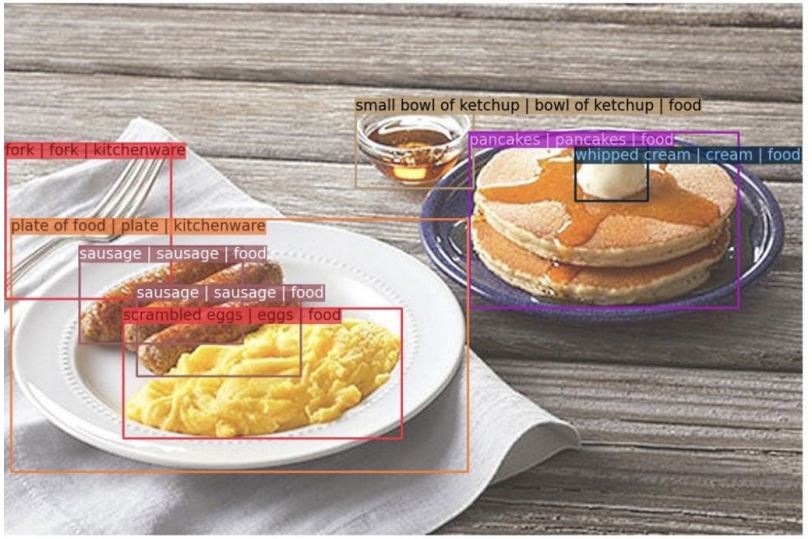


Slide inspired by Lewei Yao
https://faculty.cc.gatech.edu/~zk15/teaching/AY2025_cs8803vlm_fall/

DetCLIP-v3: Overview

DetCLIPv3 is a high-performing detector that excels not only at open-vocabulary object detection, but also generating hierarchical descriptions for detected objects.

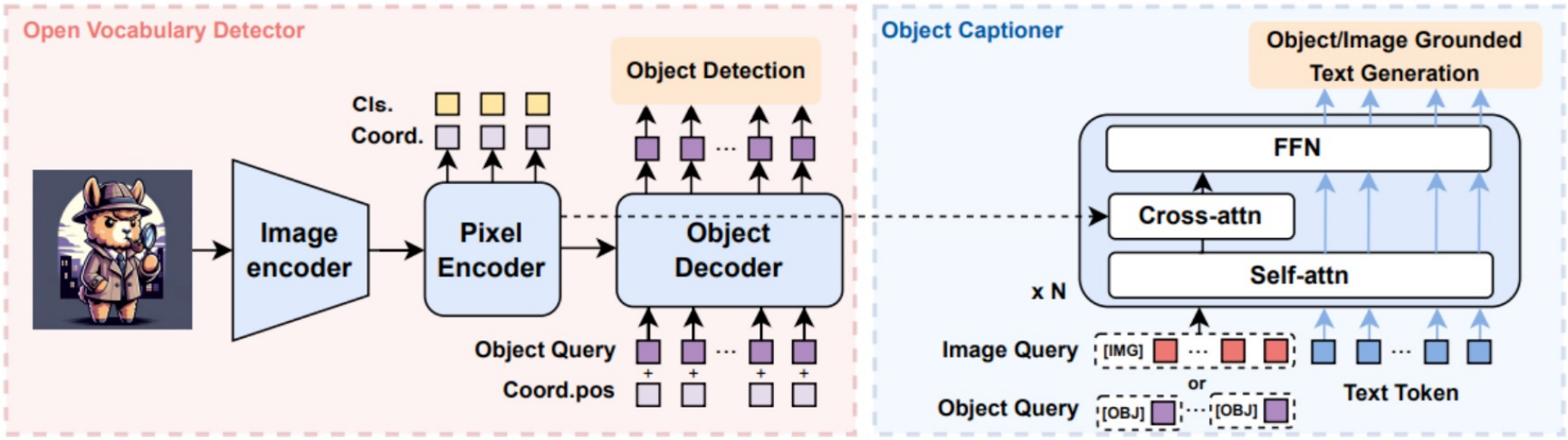




Slide inspired by Lewei Yao
https://faculty.cc.gatech.edu/~zk15/teaching/AY2025_cs8803vlm_fall/

DetCLIP-v3: Architecture

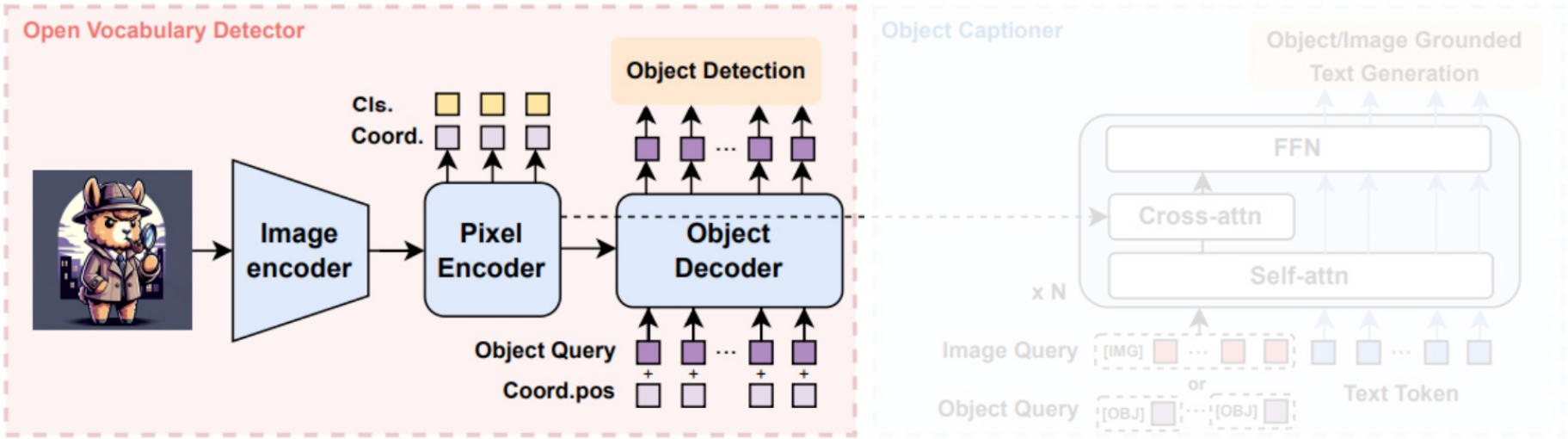
The model is powered by an open-vocabulary object detector, coupled with an object captioner for generating hierarchical and descriptive object concepts.



Slide inspired by Lewei Yao
https://faculty.cc.gatech.edu/~zk15/teaching/AY2025_cs8803vlm_fall/

DetCLIP-v3: Architecture

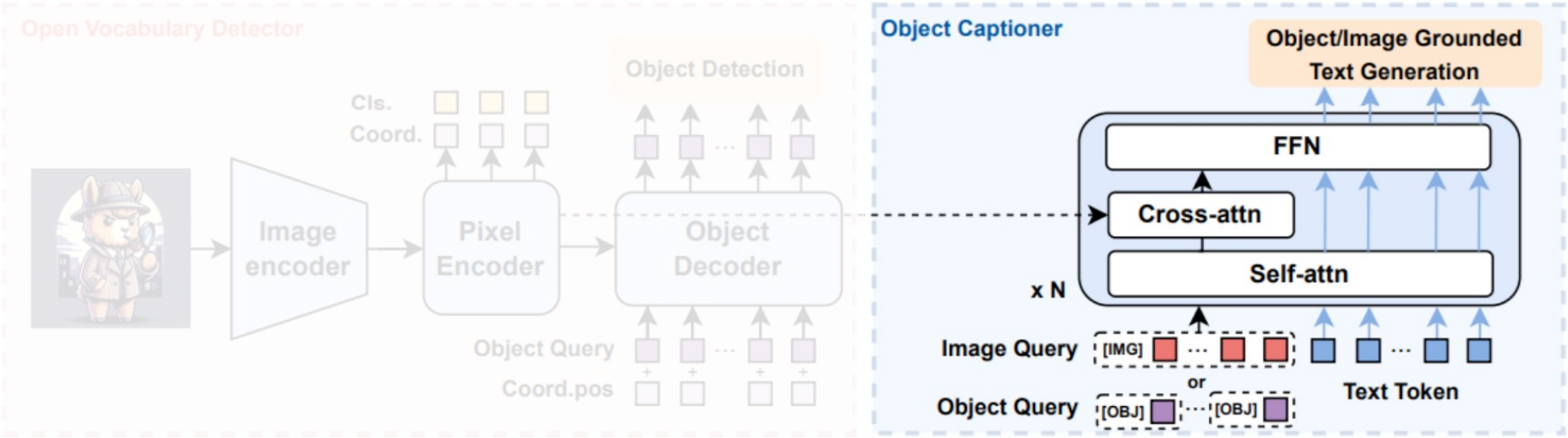
- A dual-path model comprising a **visual detector** and **text encoder**
- Visual object detector employs a **DETR-like** architecture
- Utilizes text features to select the top-k visual tokens from a pixel encoder based on **similarity**



Slide inspired by Lewei Yao
https://faculty.cc.gatech.edu/~zk15/teaching/AY2025_cs8803vlm_fall/

DetCLIP-v3: Architecture

- A Transformer-based architecture initialized with the weights of QFormer[1]
- 2 types of visual queries: **image and object-level** (provided by the OV detector)
- Visual queries interact with features from the pixel encoder via **deformable** cross-attention



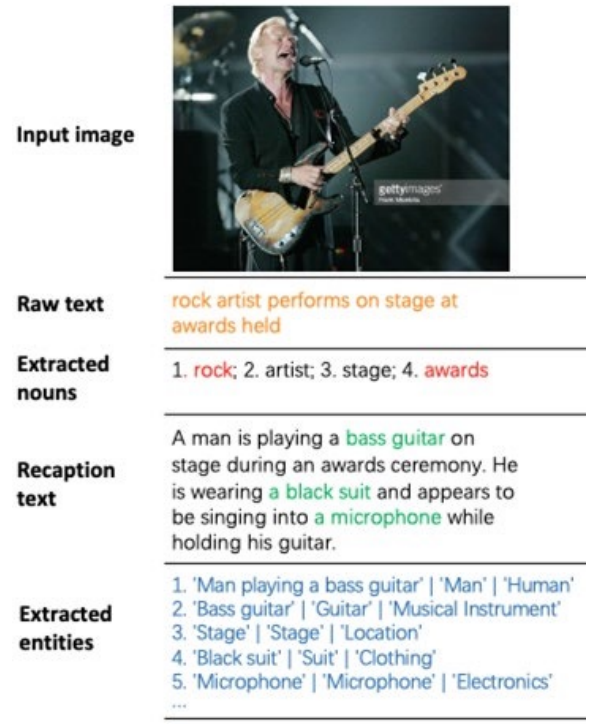
[1] Junnan Li, Dongxu Li, Silvio Savarese, and Steven Hoi. Blip-2: Bootstrapping language-image pretraining with frozen image encoders and large language models.

Slide inspired by Lewei Yao
https://faculty.cc.gatech.edu/~zk15/teaching/AY2025_cs8803vlm_fall/

DetCLIP-v3: Data

To construct a dataset with diverse object-level multi-granular descriptions, an auto-annotation pipeline is developed with 4 steps:

1. Re-captioning image-text pairs with a VLM (InstructBLIP)
2. Entity extraction using GPT-4
3. Fine-tuning the VLM (LLaVA) for large-scale annotation
4. Auto-labeling for bounding boxes



Slide inspired by Lewei Yao
https://faculty.cc.gatech.edu/~zk15/teaching/AY2025_cs8803vlm_fall/

DetCLIP-v3: Experiments

DetCLIPv3 achieves SoTA zero-shot OVD performance on a 1203-class dataset LVIS, surpassing previous methods by a large margin.

	Method	Backbone	Pre-training data	LVIS^{minival} AP_{all}	AP_r	AP_c	AP_f
1	GLIP [29]	Swin-T	O365,GoldG,Cap4M	26.0	20.8	21.4	31.0
2	GLIPv2 [65]	Swin-T	O365,GoldG,Cap4M	29.0	–	–	–
3	CapDet [38]	Swin-T	O365,VG	33.8	29.6	32.8	35.5
4	GroundingDINO [36]	Swin-T	O365,GoldG,Cap4M	27.4	18.1	23.3	32.7
5	OWL-ST [43]	CLIP B/16	WebLI2B	34.4	38.3	–	–
6	DetCLIP [58]	Swin-T	O365,GoldG,YFCC1M	35.9	33.2	35.7	36.4
7	DetCLIPv2 [60]	Swin-T	O365,GoldG,CC15M	40.4	36.0	41.7	40.4
8	DetCLIPv3	Swin-T	O365,V3Det,GoldG,GranuCap50M	**47.0**	**45.1**	**47.7**	**46.7**

Slide inspired by Lewei Yao
https://faculty.cc.gatech.edu/~zk15/teaching/AY2025_cs8803vlm_fall/

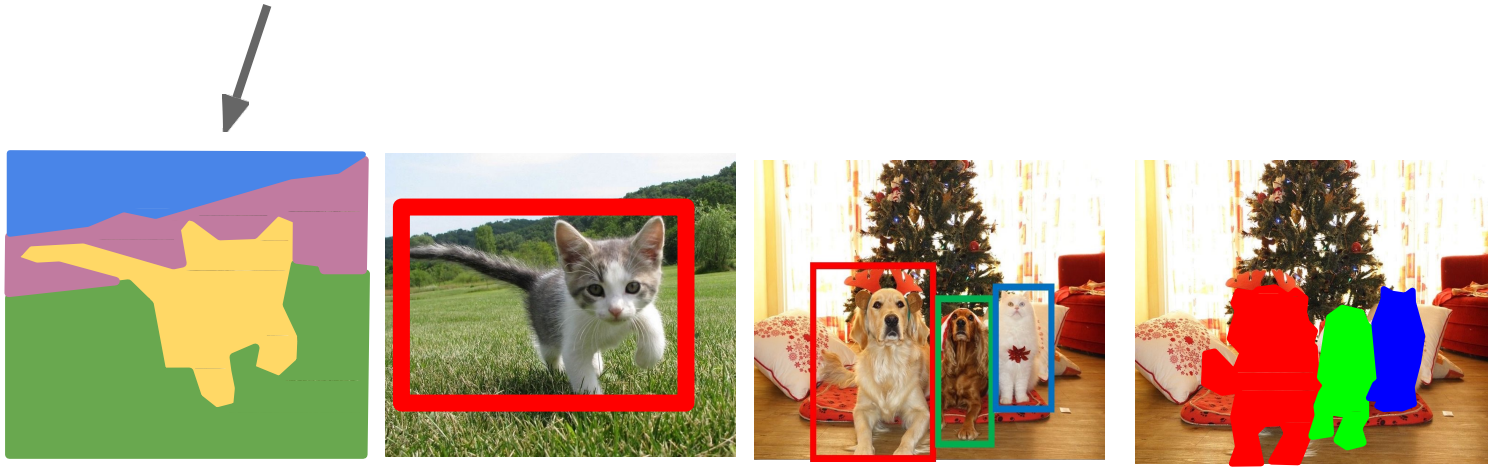
DetCLIP-v3: Results

Slide inspired by Lewei Yao
https://faculty.cc.gatech.edu/~zk15/teaching/AY2025_cs8803vlm_fall/

Plan for the next few lectures

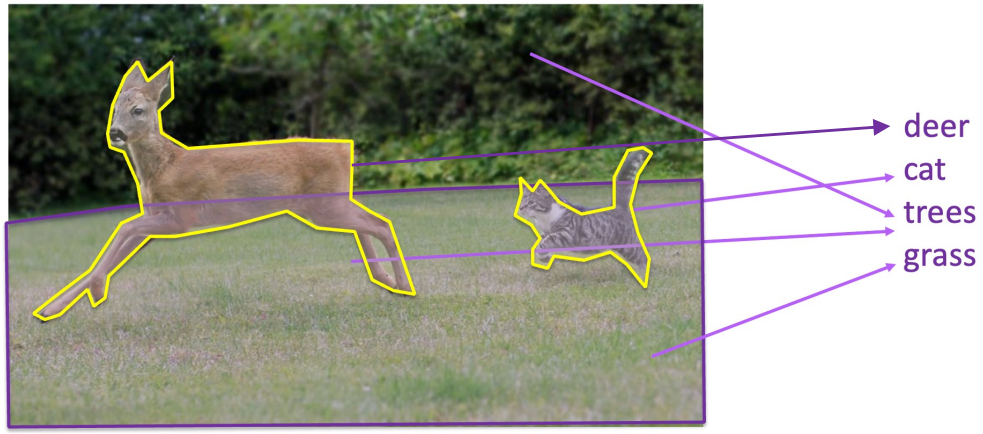
- Detection approaches
  - Pre-CNNs
    - Detection with whole windows: Pedestrian detection
    - Part-based detection: Deformable Part Models
  - Post-CNNs
    - Detection with region proposals: R-CNN, Fast R-CNN, Faster-R-CNN
    - Detection without region proposals: YOLO, SSD, DETR
- Learning from noisy web image-text data
  - Contrastive Language-Image Pretraining (CLIP)
    - Prompting
  - Open-vocabulary object detection
- Segmentation approaches
  - Semantic segmentation
    - Fully-Convolutional Networks (FCN)
  - Instance segmentation
    - Mask R-CNN
    - Segment Anything

Semantic Segmentation



Slide by: Justin Johnson

Semantic Segmentation

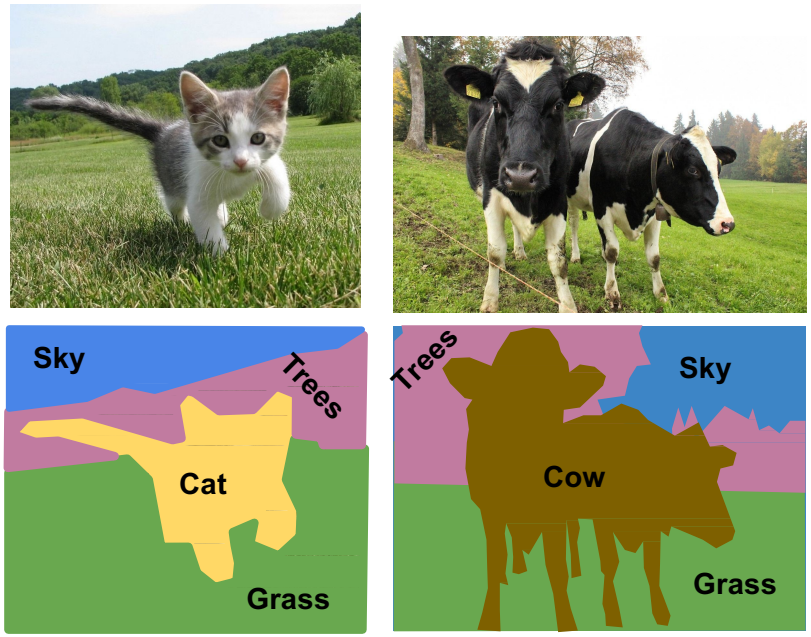


Adapted from Vicente Ordoñez

Semantic Segmentation

Label each pixel in the image with a category label

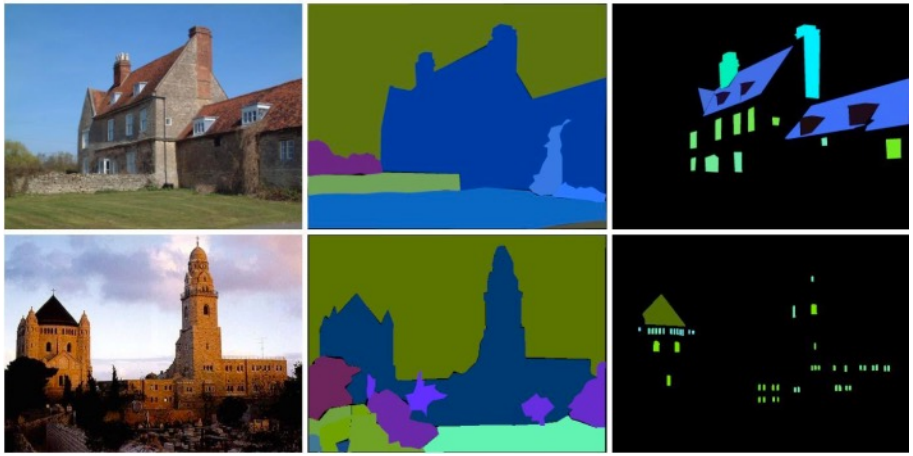
Don't differentiate instances, only care about pixels

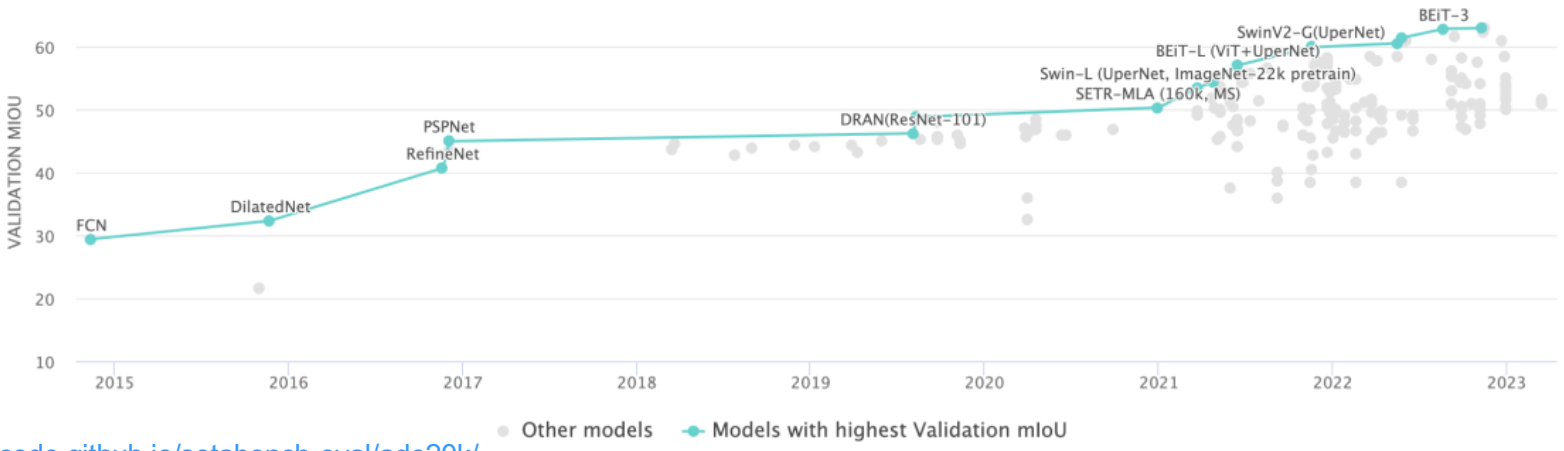


Slide by: Justin Johnson

Semantic Segmentation dataset: ADE20k

- 20K scene-centric images exhaustively annotated with pixel-level objects and object parts labels.
- 150 semantic categories, which include stuffs like sky, road, grass, and discrete objects like person, car, bed.

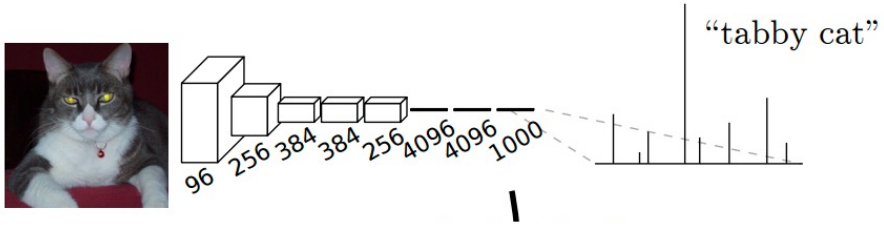




https://paperswithcode.github.io/sotabench-eval/ade20k/

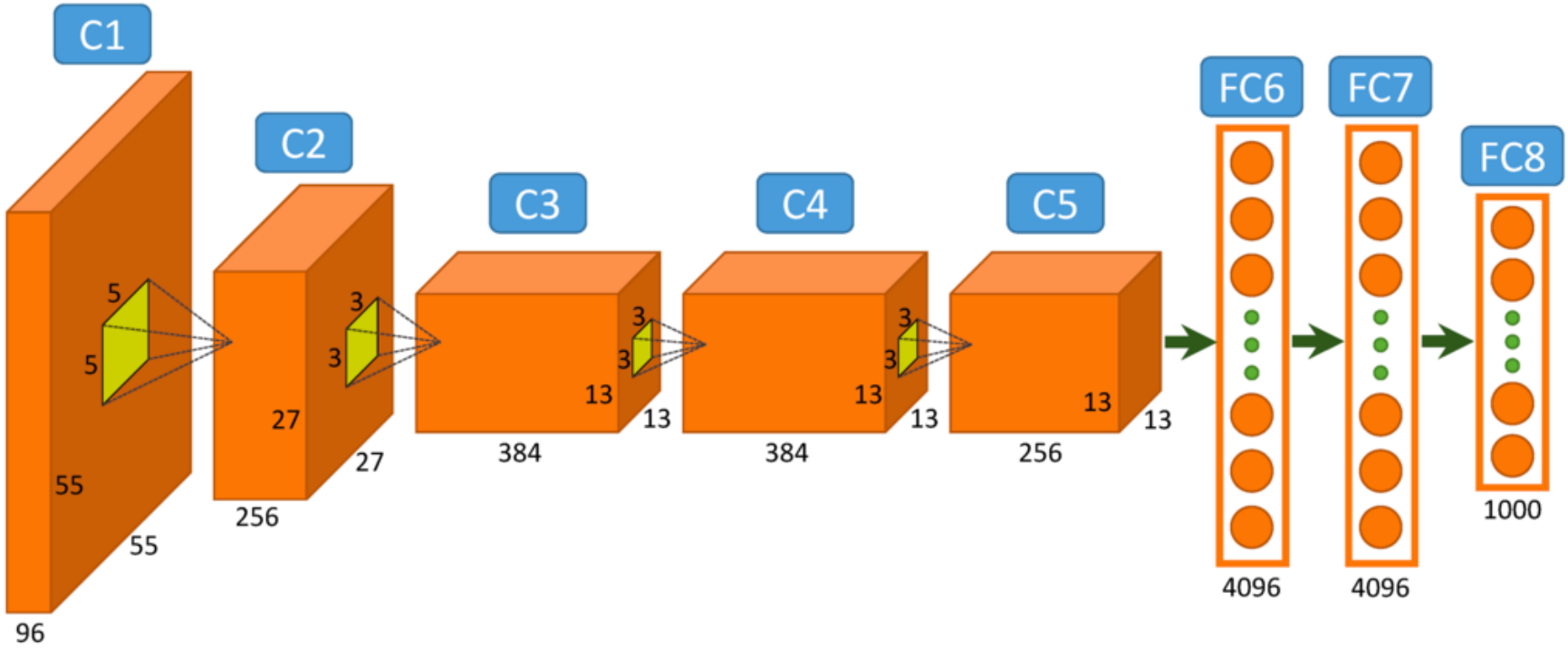
https://www.cs.unc.edu/~ronisen/teaching/fall_2024/intro2vision_fall2024.html

Idea 1: Convolutionalization



Adapted from Vicente Ordoñez

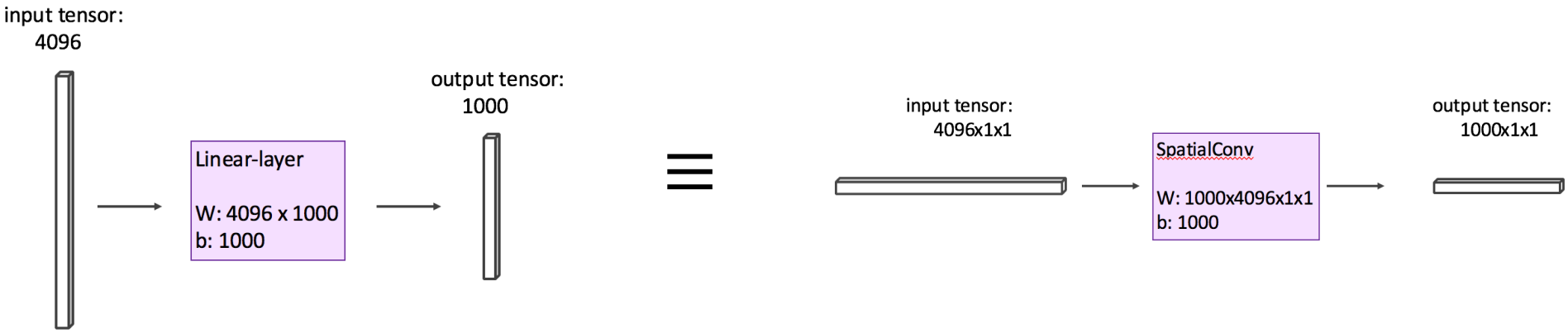
Idea 1: Convolutionalization



https://www.saagie.com/fr/blog/object-detection-part1

Idea 1: Convolutionalization

nn.Linear(4096, 1000) == nn.Conv2D(4096, 1000, kernel_size = 1, stride = 1)

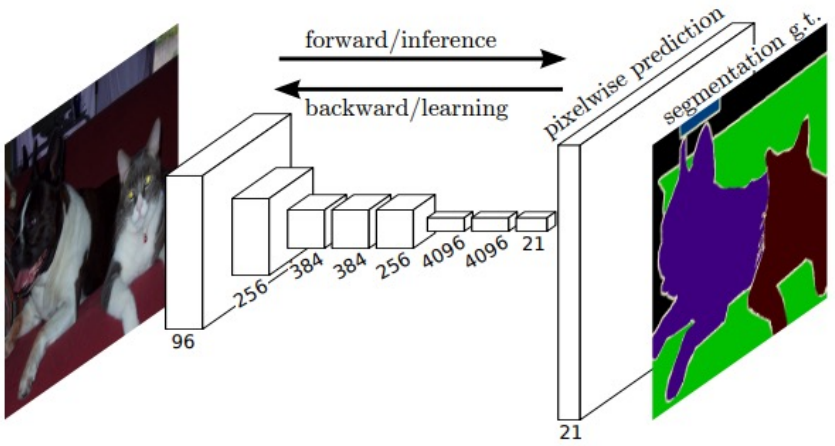


Adapted from Vicente Ordoñez

Idea 2: Fully Convolutional Networks (CVPR 2015)

Fully Convolutional Networks for Semantic Segmentation

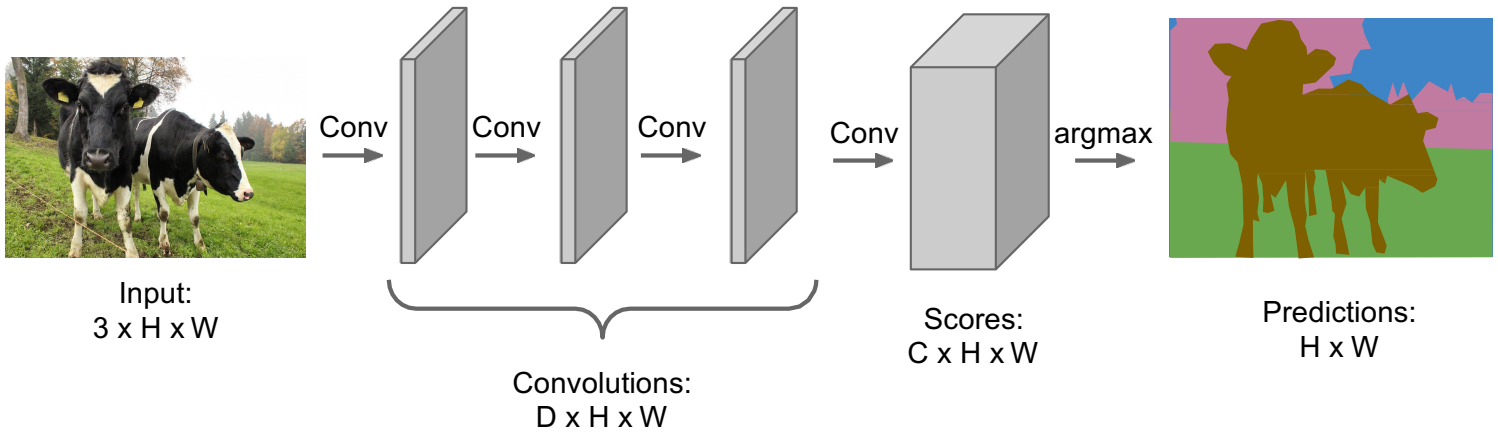
Jonathan Long* Evan Shelhamer* Trevor Darrell
UC Berkeley
{jonlong,shelhamer,trevor}@cs.berkeley.edu



Lab 8a

Idea 2: Fully Convolutional

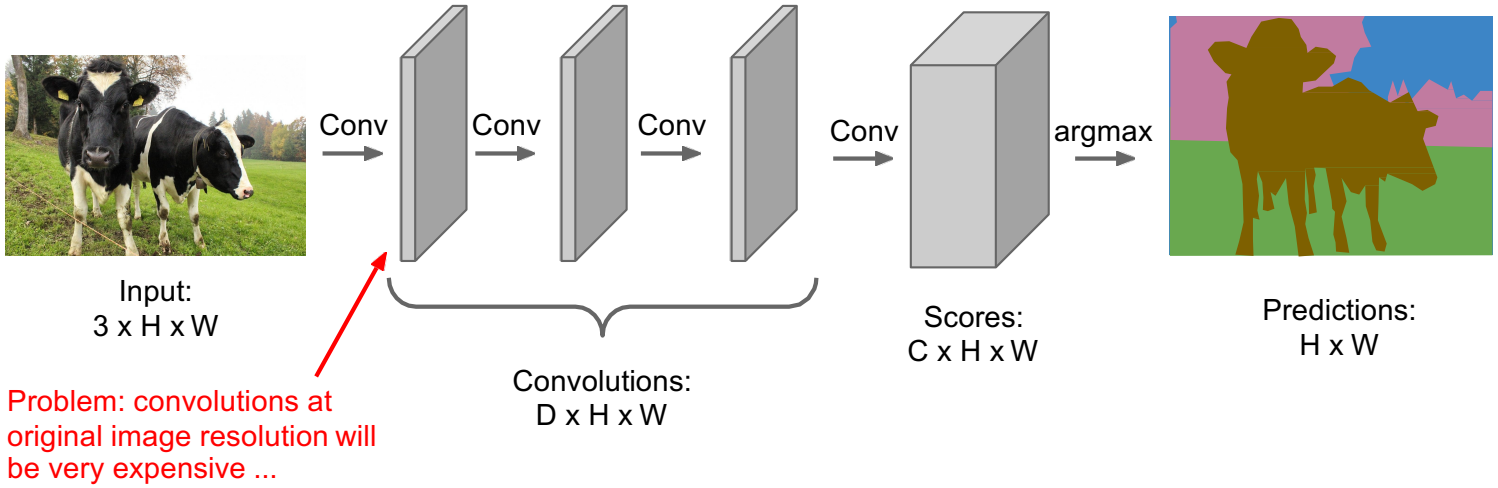
Design a network as a bunch of convolutional layers to make predictions for pixels all at once!



Slide by: Justin Johnson

Idea 2: Fully Convolutional

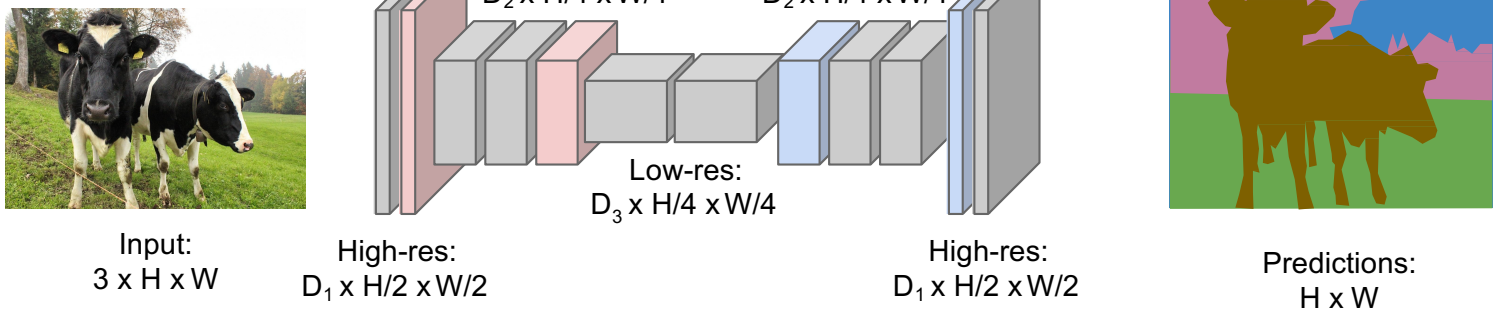
Design a network as a bunch of convolutional layers to make predictions for pixels all at once!



Slide by: Justin Johnson

Idea 2: Fully Convolutional

Design network as a bunch of convolutional layers, with **downsampling** and **upsampling** inside the network!



Long, Shelhamer, and Darrell, "Fully Convolutional Networks for Semantic Segmentation", CVPR 2015
Noh et al, "Learning Deconvolution Network for Semantic Segmentation", ICCV 2015

Slide by: Justin Johnson

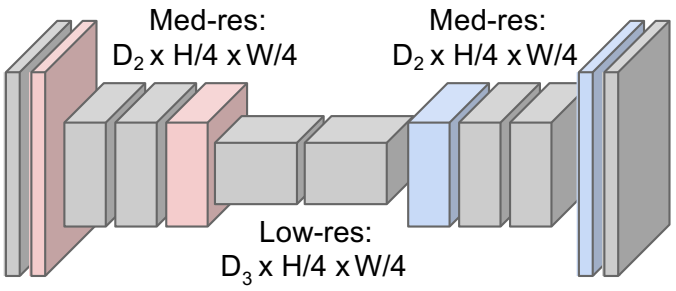
Idea 2: Fully Convolutional

Downsampling:
Pooling, strided convolution

Design network as a bunch of convolutional layers, with **downsampling** and **upsampling** inside the network!

Upsampling:
???

Input:
3 x H x W



High-res:
D_1 x H/2 x W/2

High-res:
D_1 x H/2 x W/2

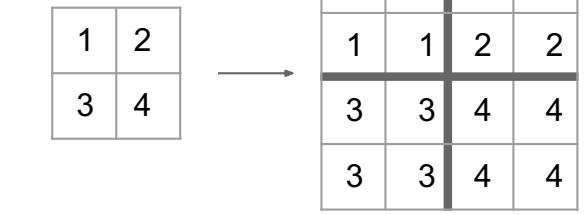
Predictions:
H x W

Long, Shelhamer, and Darrell, "Fully Convolutional Networks for Semantic Segmentation", CVPR 2015
Noh et al, "Learning Deconvolution Network for Semantic Segmentation", ICCV 2015

Slide by: Justin Johnson

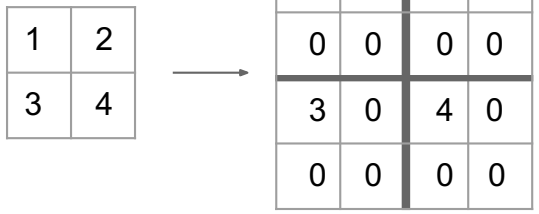
In-Network upsampling: “Unpooling”

Nearest Neighbor



Input: 2 x 2 Output: 4 x 4

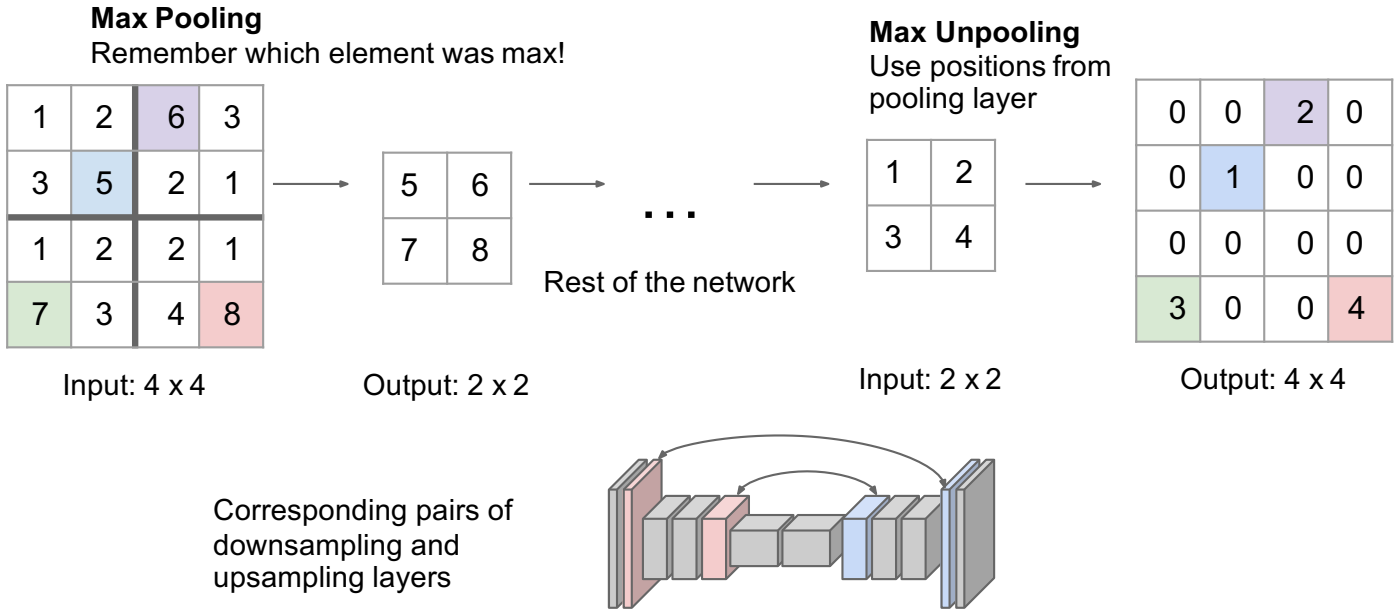
“Bed of Nails”



Input: 2 x 2 Output: 4 x 4

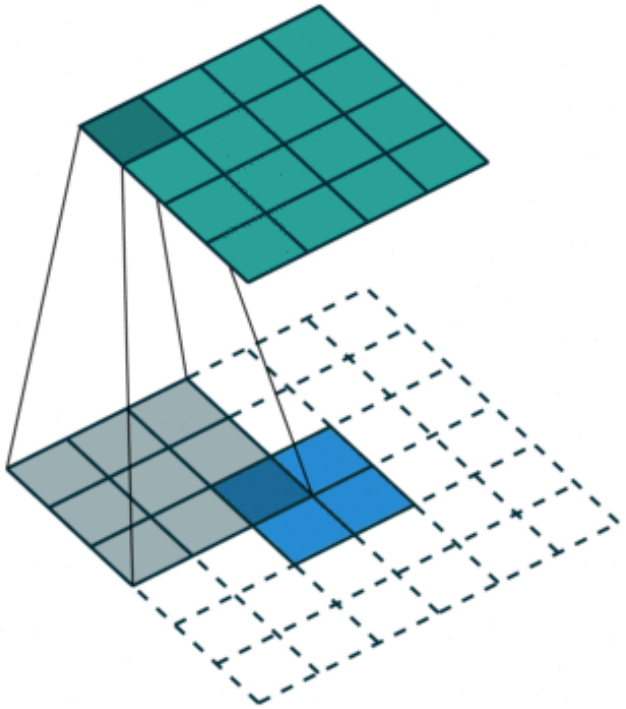
Slide by: Justin Johnson

In-Network upsampling: “Max Unpooling”



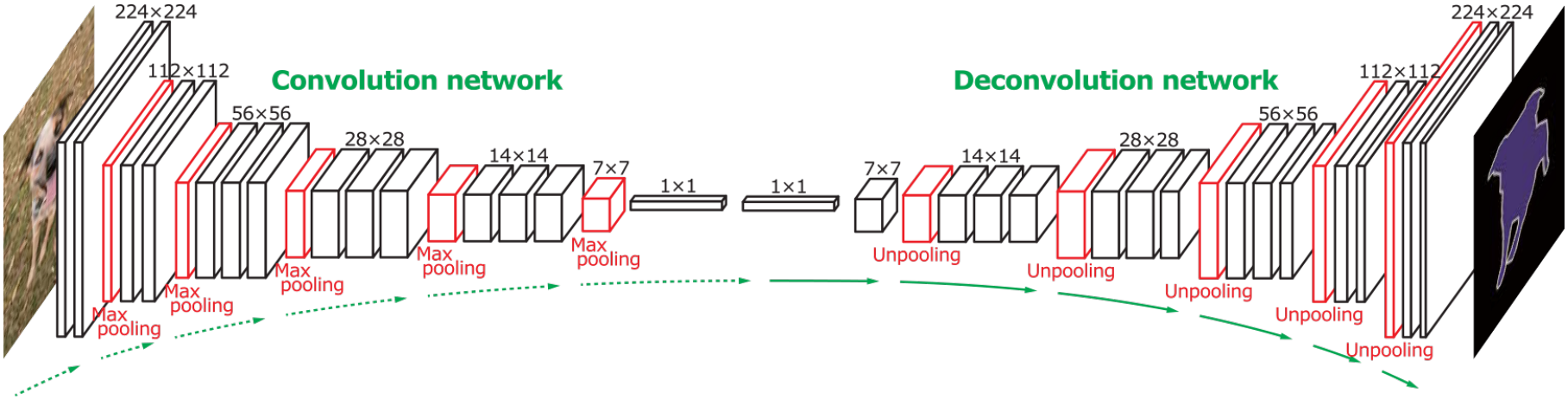
Slide by: Justin Johnson

In-Network Up-sampling Convolutions or ”Deconvolutions”



https://github.com/vdumoulin/conv_arithmetic

Idea 2: Fully Convolutional Networks



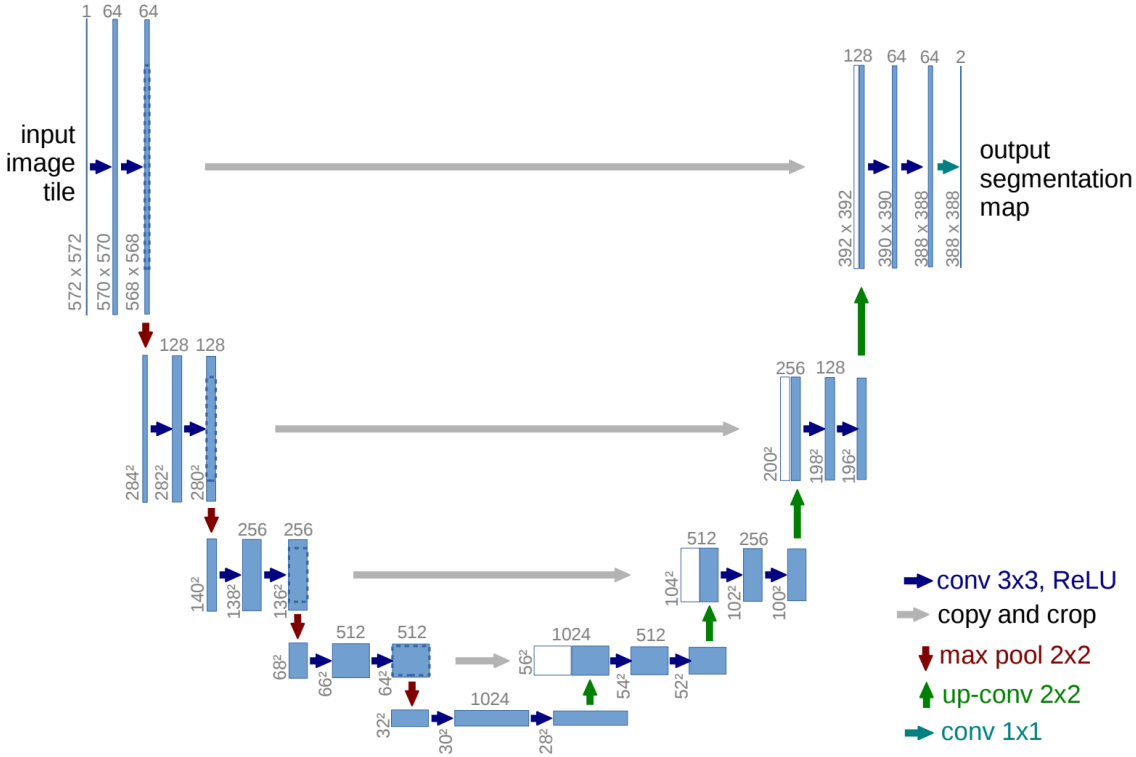
Learning Deconvolution Network for Semantic Segmentation

Hyeonwoo Noh Seunghoon Hong Bohyung Han
Department of Computer Science and Engineering, POSTECH, Korea
{hyeonwoonoh_,maga33,bhhan}@postech.ac.kr

http://cvlab.postech.ac.kr/research/deconvnet/

U-Net: Convolutional Networks for Biomedical Image Segmentation

Olaf Ronneberger, Philipp Fischer, and Thomas Brox



https://arxiv.org/abs/1505.04597

https://github.com/milesial/Pytorch-UNet

https://github.com/usuyama/pytorch-unet

UNet in Pytorch

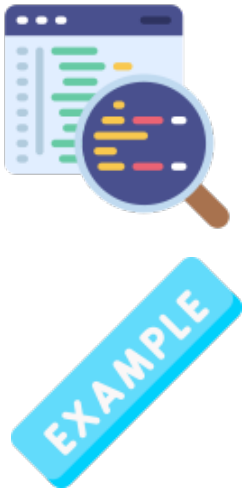
```
from .unet_parts import *


class UNet(nn.Module):
    def __init__(self, n_channels, n_classes, bilinear=False):
        super(UNet, self).__init__()
        self.n_channels = n_channels
        self.n_classes = n_classes
        self.bilinear = bilinear

        self.inc = (DoubleConv(n_channels, 64))
        self.down1 = (Down(64, 128))
        self.down2 = (Down(128, 256))
        self.down3 = (Down(256, 512))
        factor = 2 if bilinear else 1
        self.down4 = (Down(512, 1024 // factor))
        self.up1 = (Up(1024, 512 // factor, bilinear))
        self.up2 = (Up(512, 256 // factor, bilinear))
        self.up3 = (Up(256, 128 // factor, bilinear))
        self.up4 = (Up(128, 64, bilinear))
        self.outc = (OutConv(64, n_classes))

    def forward(self, x):
        x1 = self.inc(x)
        x2 = self.down1(x1)
        x3 = self.down2(x2)
        x4 = self.down3(x3)
        x5 = self.down4(x4)
        x = self.up1(x5, x4)
        x = self.up2(x, x3)
        x = self.up3(x, x2)
        x = self.up4(x, x1)
        logits = self.outc(x)
        return logits
```

https://github.com/milesial/Pytorch-UNet/blob/master/unet/unet_model.py

Image Segmentation Learning

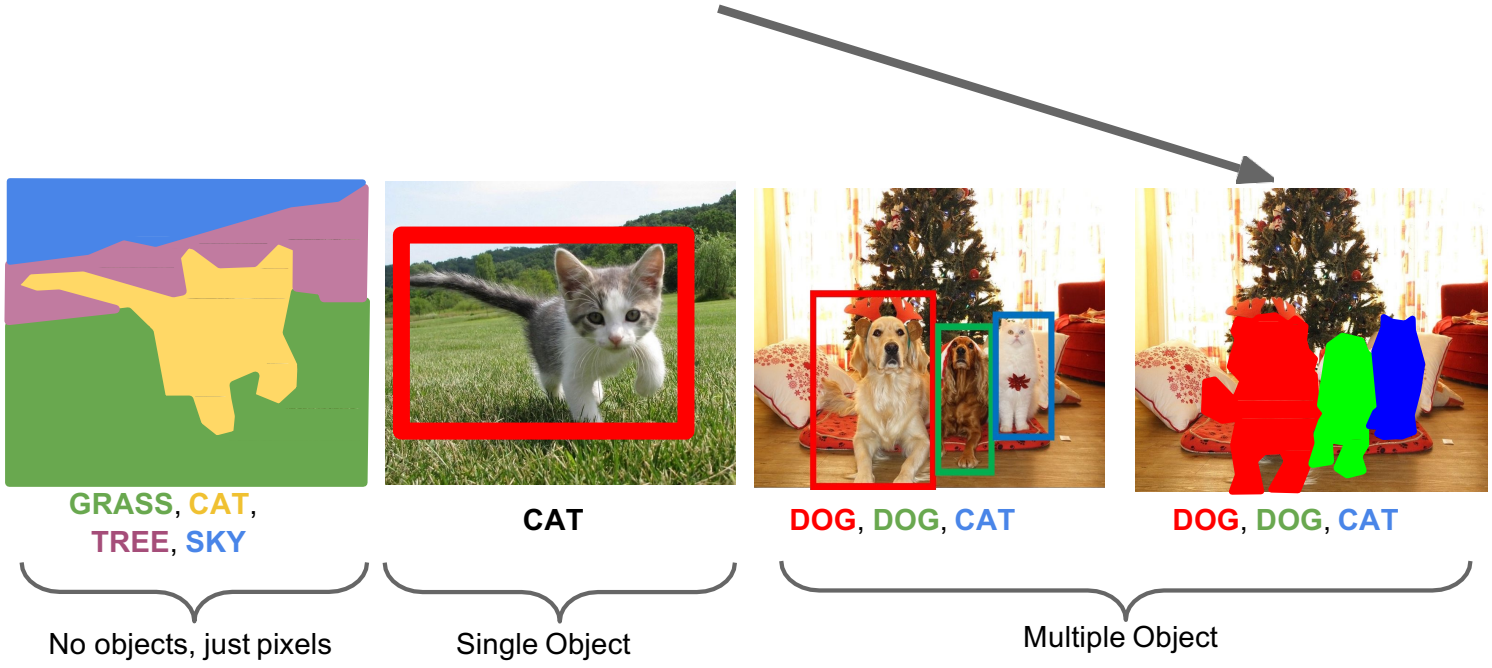
What loss can we use?

Additional Resources



- Image Segmentation with Synthetic Masks [link]
- Segmentation Models [link]
- Oxford Pets Image Segmentation [link]
- Cars Image Segmentation [link]
- Road Analysis Image Segmentation [link]

Instance Segmentation



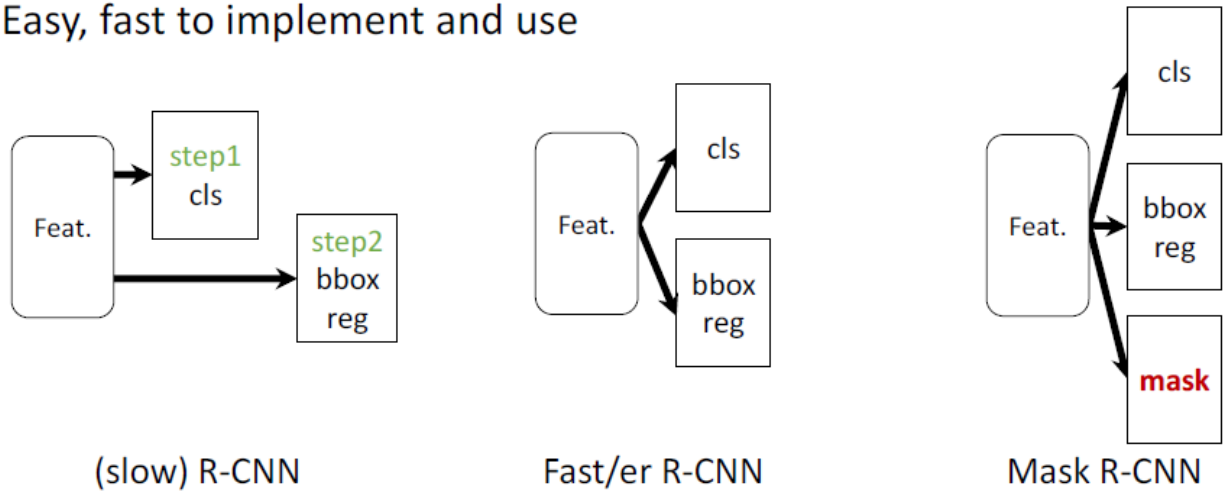
Slide by: Justin Johnson

Mask R-CNN

He et al, "Mask R-CNN", ICCV 2017

What is Mask R-CNN: Parallel Heads

- Easy, fast to implement and use

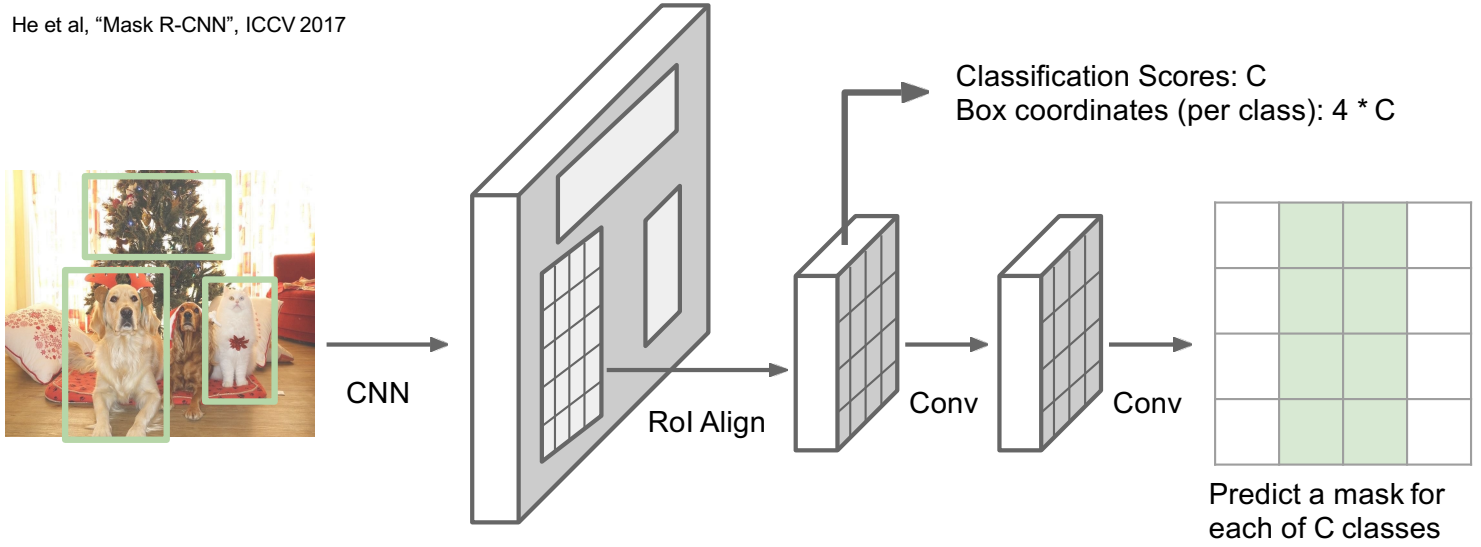


(slow) R-CNN Fast/er R-CNN Mask R-CNN

Slide by: Kaiming He

Mask R-CNN

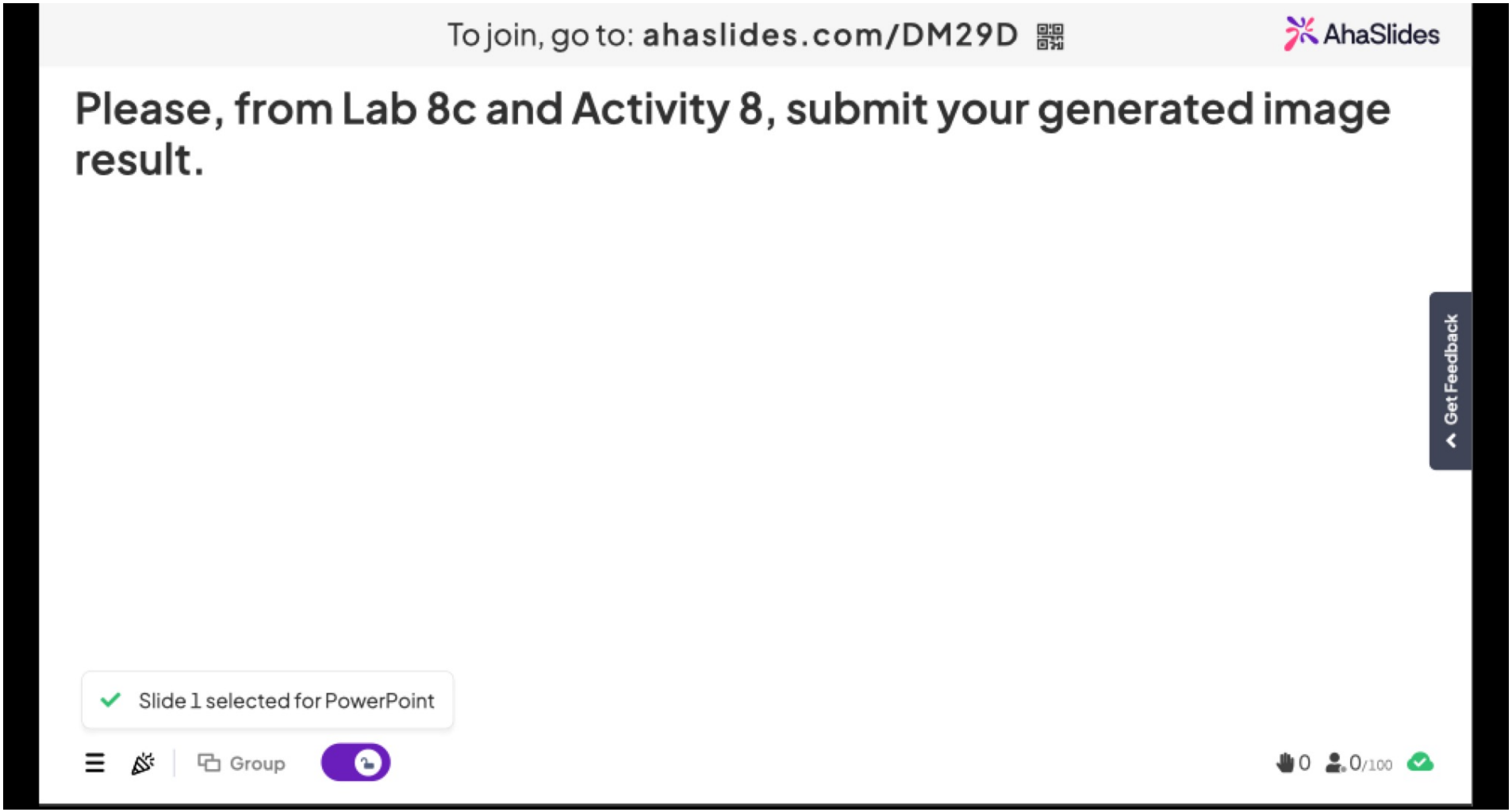
He et al, “Mask R-CNN”, ICCV 2017



Adapted from Justin Johnson

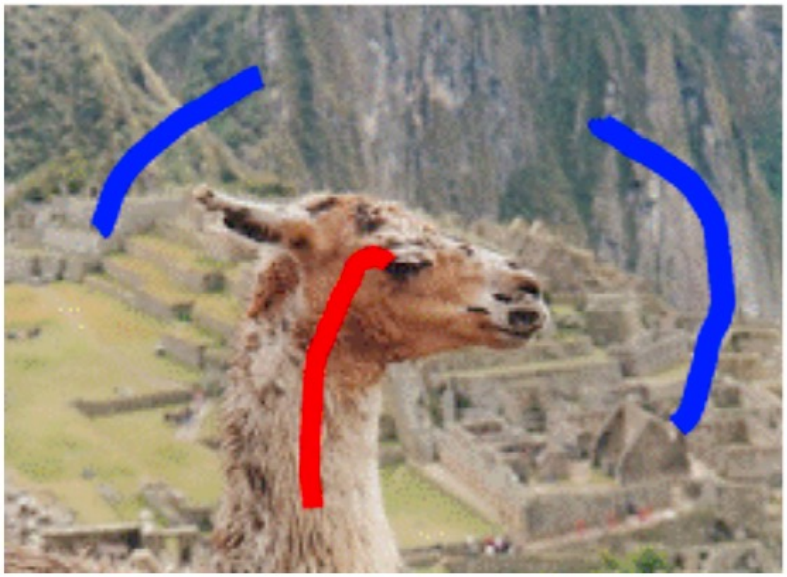
Lab 8c: Fine-Tuning Mask R-CNN for Object Recognition and Image Segmentation

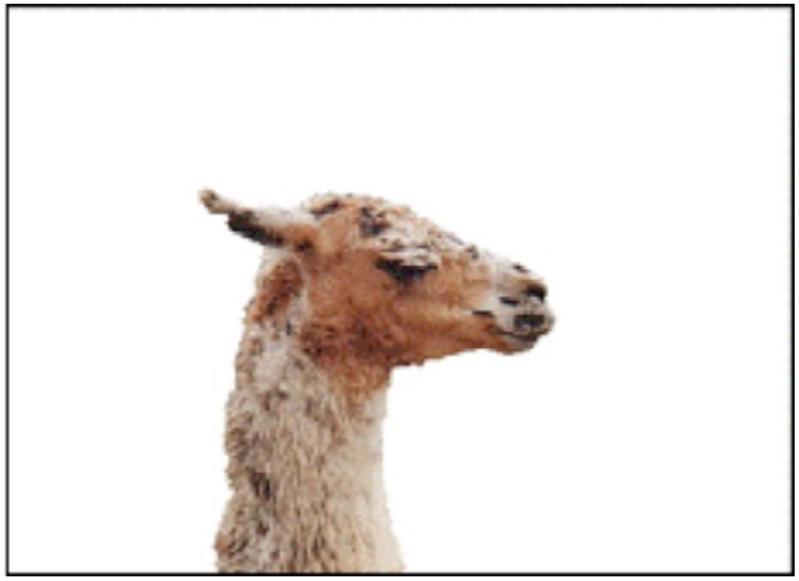
Duration: 10 min


To join, go to: ahaslides.com/DM29D
AhaSlides
Please, from Lab 8c and Activity 8, submit your generated image result.
Get Feedback
Slide 1 selected for PowerPoint
Group
0
0/100

Interactive Segmentation: With Scribbles

(Red = foreground, blue = background)





Earlier works of segmentation used Graph Cut techniques to solve this problem.

Normalized cuts and image segmentation

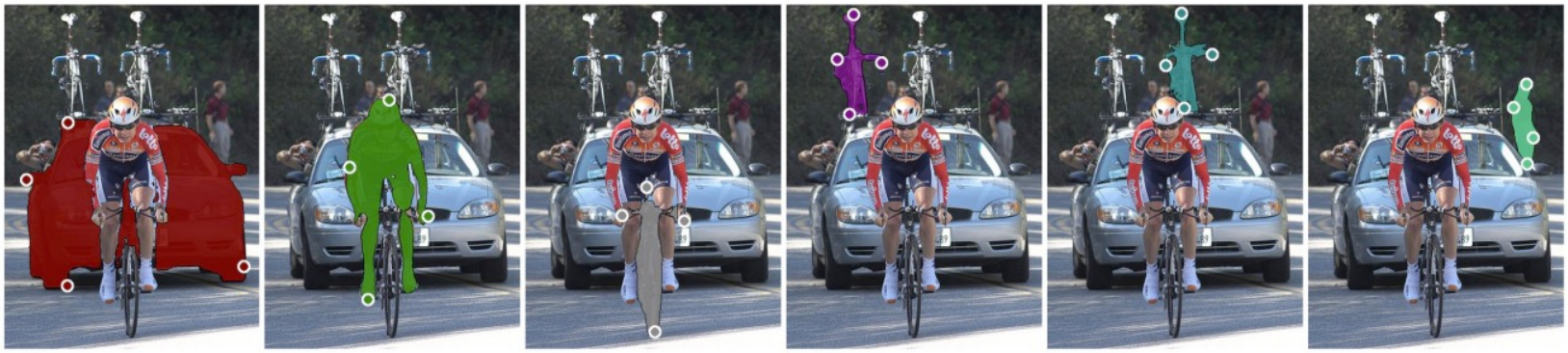
J Shi, J Malik - IEEE Transactions on pattern analysis and …, 2000 - ieeexplore.ieee.org

… have smaller **cut** value than the **cut** that partitions … **cut** cost as a fraction of the total edge connections to all the nodes in the graph. We call this disassociation measure the **normalized cut** (…

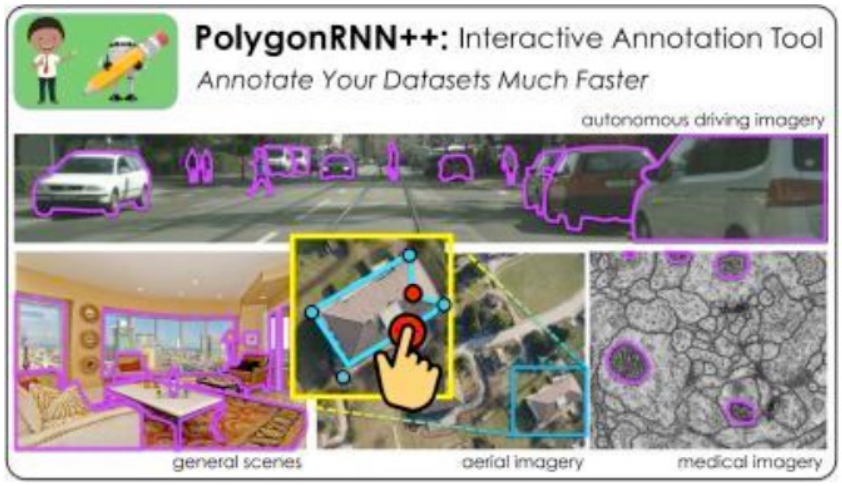
☆ Save 99 Cite Cited by 19705 Related articles All 34 versions

https://www.cs.unc.edu/~ronisen/teaching/fall_2024/intro2vision_fall2024.html

Interactive Segmentation: With Few Points



Deep Extreme Cut (DEXTR): From Extreme Points to Object Segmentation, CVPR 2018

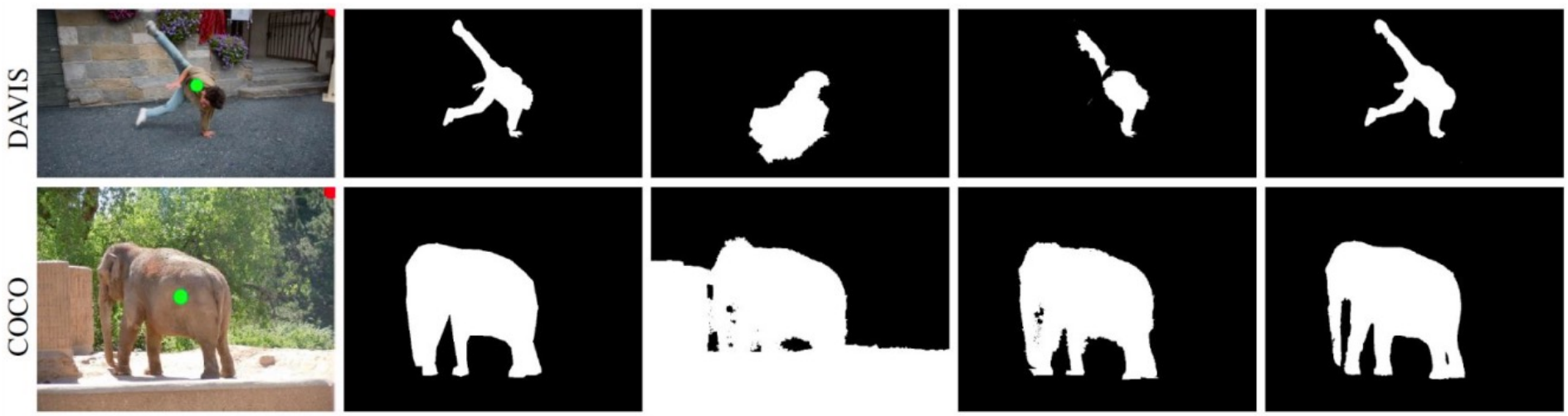


https://www.cs.unc.edu/~ronisen/teaching/fall_2024/intro2vision_fall2024.html

Efficient Annotation of Segmentation Datasets
with Polygon-RNN++, CVPR 2018

Interactive Segmentation: With 2 Points

(Green = foreground, red= background)



Interactive Image Segmentation with Latent Diversity, CVPR 2018

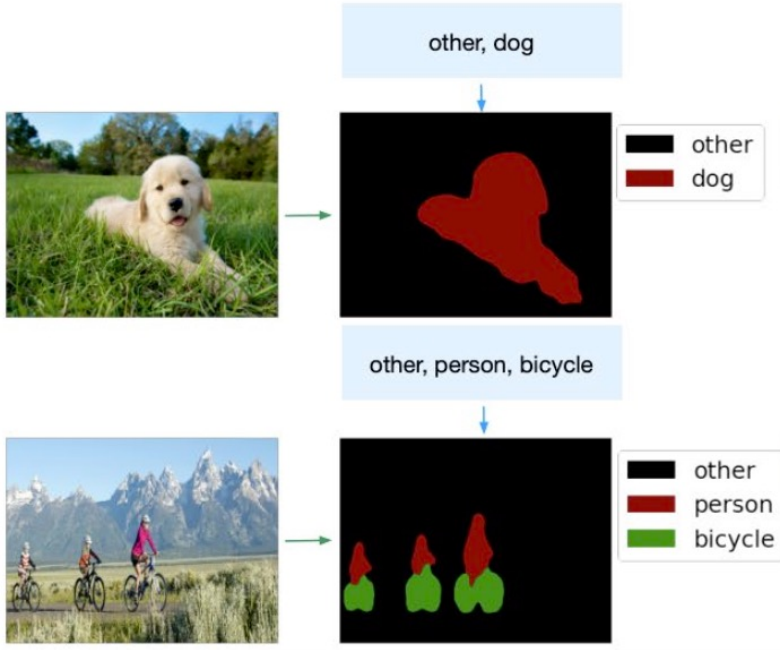
https://www.cs.unc.edu/~ronisen/teaching/fall_2024/intro2vision_fall2024.html

LSeg: Language-driven Semantic Segmentation

Problem

- CLIP at pixel-level segmentation
- Allows model to potentially learn more precise object recognition

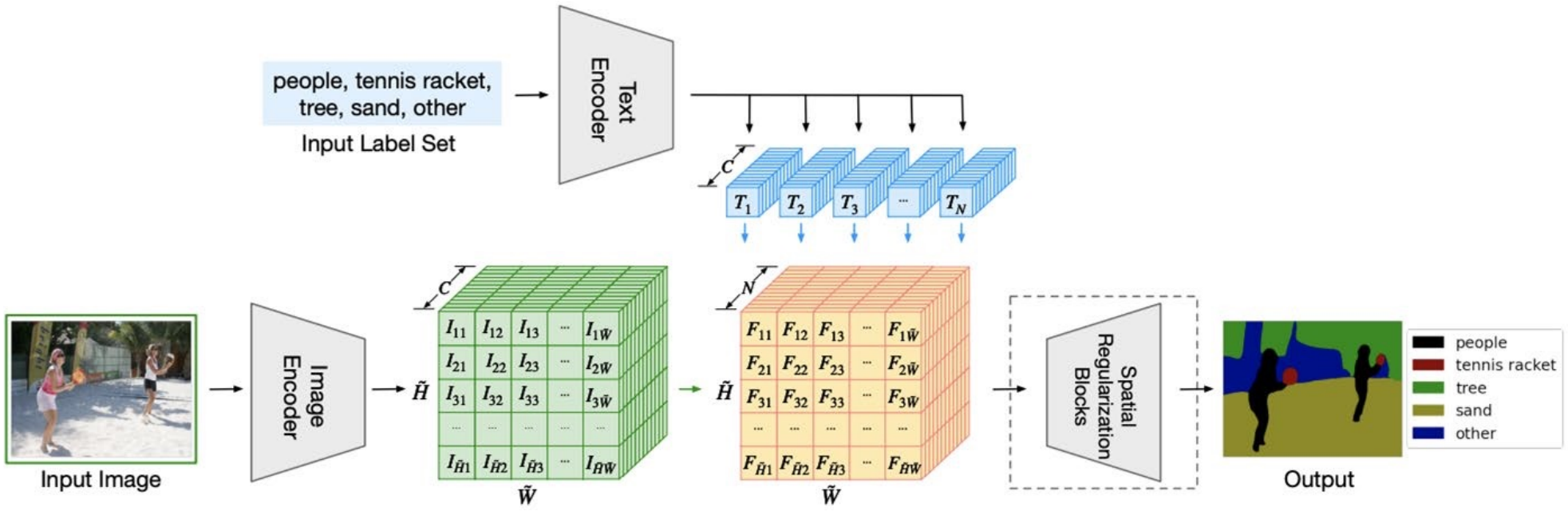
ICLR 2022



https://faculty.cc.gatech.edu/~zk15/teaching/AY2025_cs8803vlm_fall/

LSeg: Architecture

- Text embeddings per input word
- Image embedding per input pixel (after downsampling)

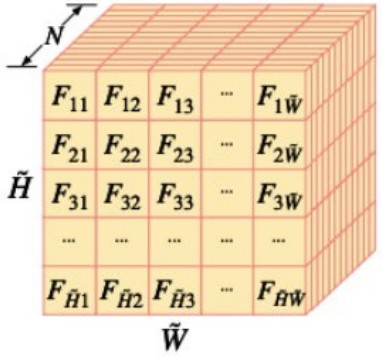


https://faculty.cc.gatech.edu/~zk15/teaching/AY2025_cs8803vlm_fall/

LSeg: Approach – Contrastive Learning

- Inner product between text and image embeddings
  Then Softmax (Over what dimension?)

$$f_{ijk} = I_{ij} \cdot T_k.$$

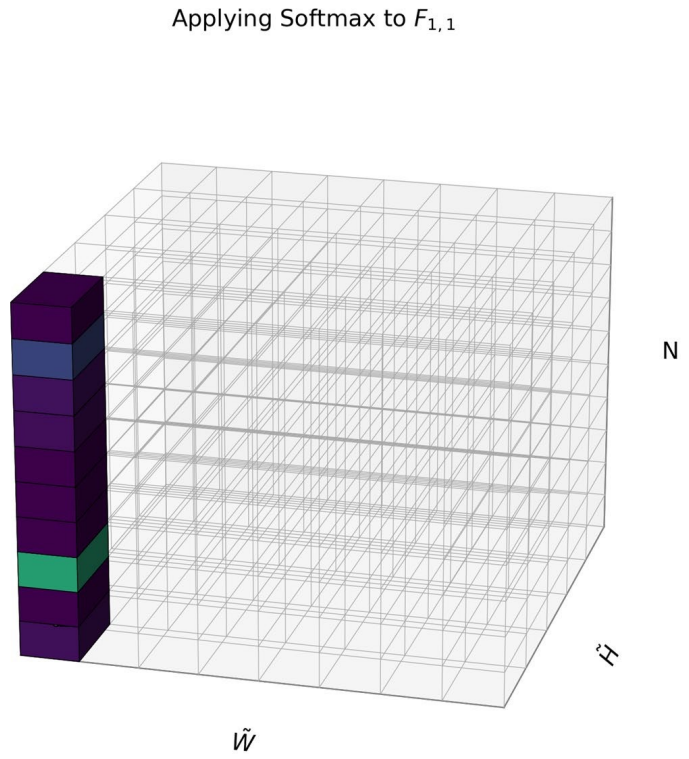


https://faculty.cc.gatech.edu/~zk15/teaching/AY2025_cs8803vlm_fall/

LSeg: Approach – Contrastive Learning

- Softmax over pixels with low temperature (t)
  Why low temperature?

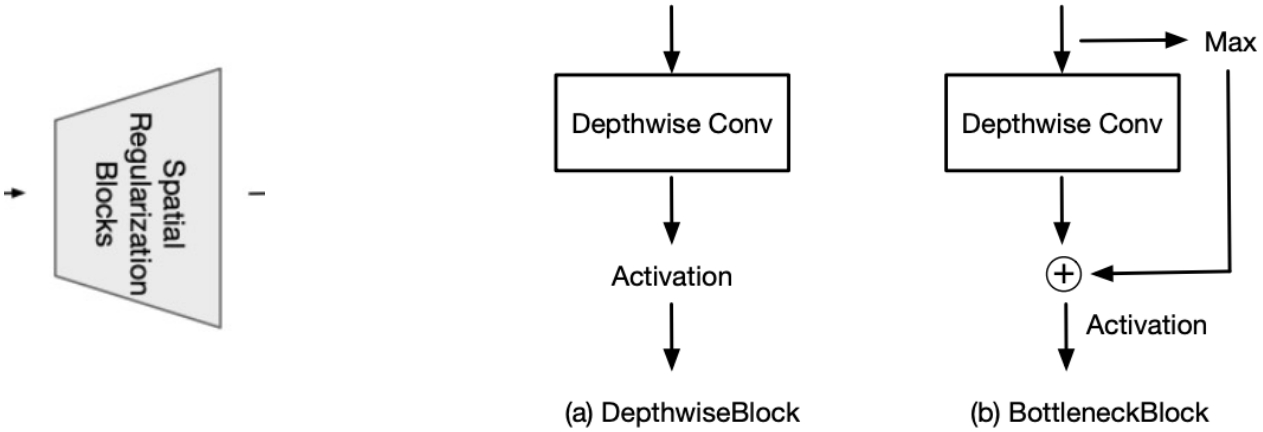
$$\sum_{i,j=1}^{H,W} \text{softmax}_{y_{ij}} \left(\frac{F_{ij}}{t} \right),$$



https://faculty.cc.gatech.edu/~zk15/teaching/AY2025_cs8803vlm_fall/

LSeg: Approach – Spatial Regularization

- Depthwise convolution for regularization
  Why do regularization at all?
- Then bilinear interpolation to recover original resolution



https://faculty.cc.gatech.edu/~zk15/teaching/AY2025_cs8803vlm_fall/

Segment Anything Model (SAM)

The first foundation model for promptable segmentation.

Prompt it with interactive points and boxes

Automatically segment everything in an image

Generate multiple valid masks for ambiguous prompts

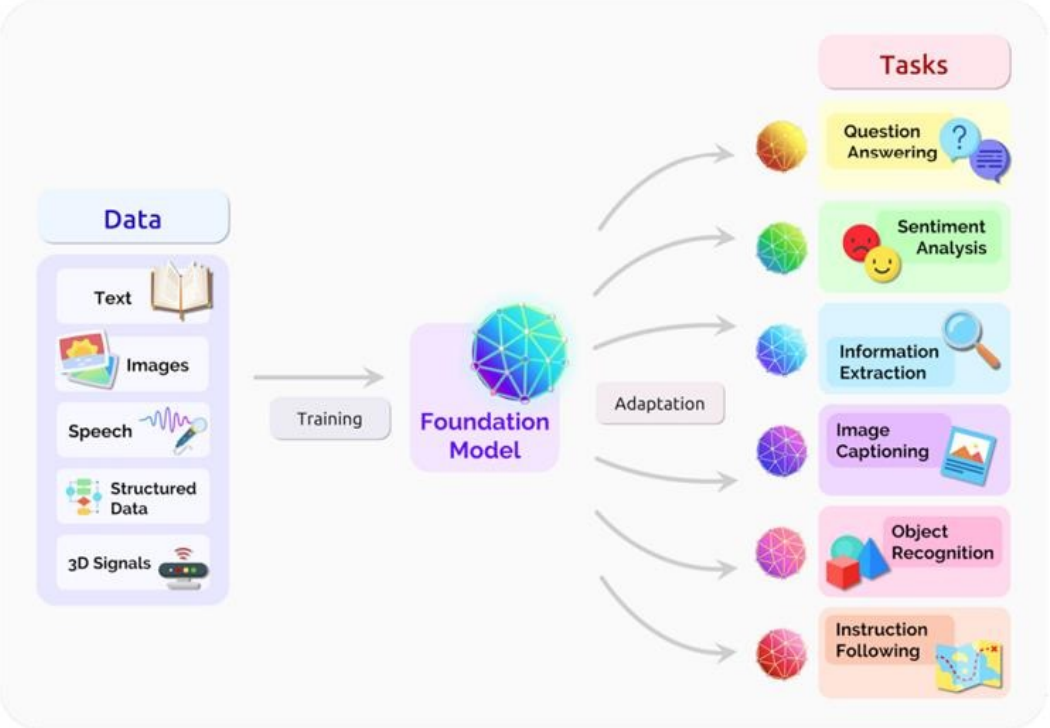
https://aidemos.meta.com/segment-anything

https://blog.roboflow.com/what-is-sam3/

https://www.cs.unc.edu/~ronisen/teaching/fall_2024/intro2vision_fall2024.html

Segment Anything Model (SAM)

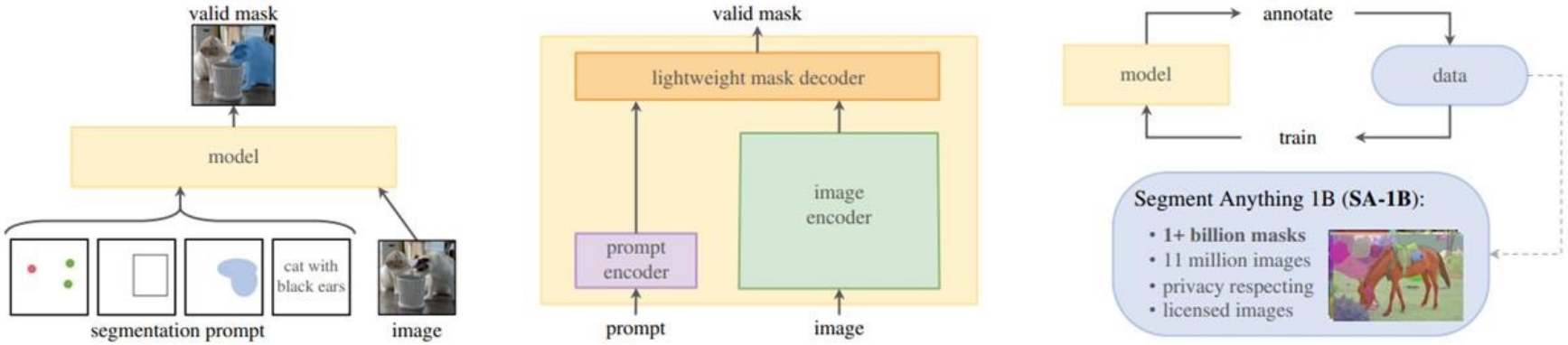
- The first foundation model for promptable segmentation.
- A foundation model can centralize the information from various modalities.



https://www.cs.unc.edu/~ronisen/teaching/fall_2024/intro2vision_fall2024.html

Segment Anything Model (SAM)

SAM is built with three interconnected components: A task, a model, and a data engine.



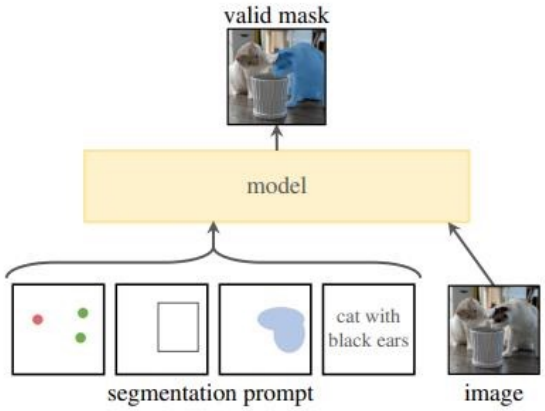
(a) **Task**: promptable segmentation (b) **Model**: Segment Anything Model (**SAM**) (c) **Data**: data engine (top) & dataset (bottom)

https://www.cs.unc.edu/~ronisen/teaching/fall_2024/intro2vision_fall2024.html

Segment Anything Model (SAM): Task

Promptable Segmentation

- SAM considers two sets of prompts: **sparse** (clicks, boxes, text) and **dense** (masks).
- SAM's promptable design enables flexible integration with other systems (i.e., used as **component** in larger systems).

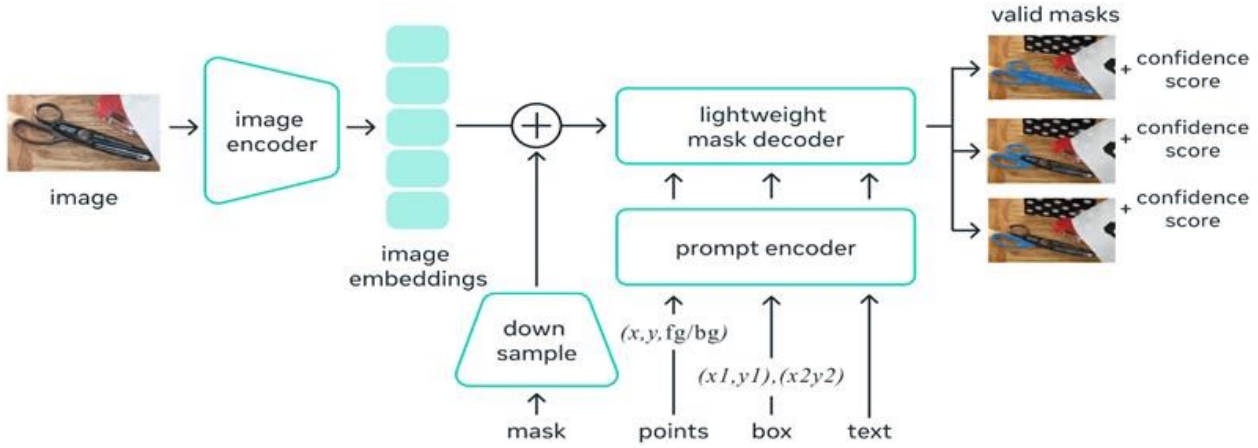


(a) **Task**: promptable segmentation

https://www.cs.unc.edu/~ronisen/teaching/fall_2024/intro2vision_fall2024.html

Segment Anything Model (SAM): Model

- A heavyweight **image encoder** outputs an image embedding.
- A lightweight **prompt encoder** efficiently queries the image embedding.
- A lightweight **mask decoder** produces object masks and confidence scores.

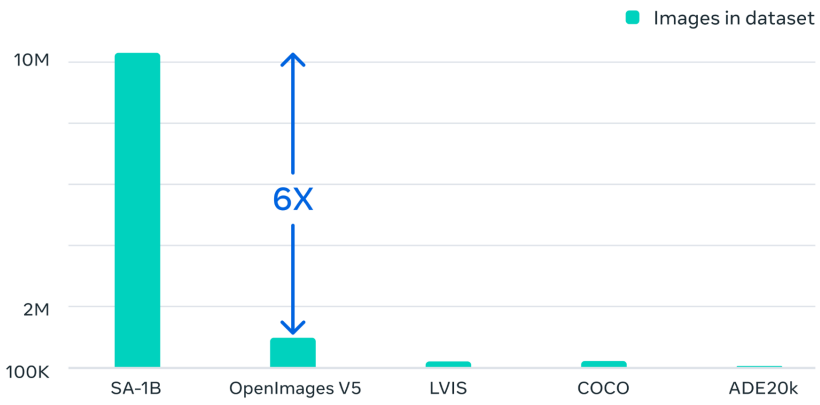


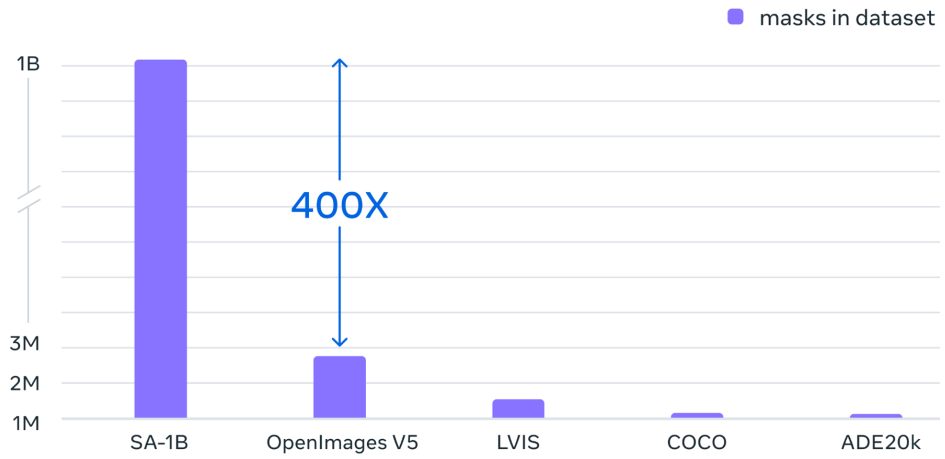
https://www.cs.unc.edu/~ronisen/teaching/fall_2024/intro2vision_fall2024.html

Segment Anything Model (SAM): Data

Dataset: SA-1B

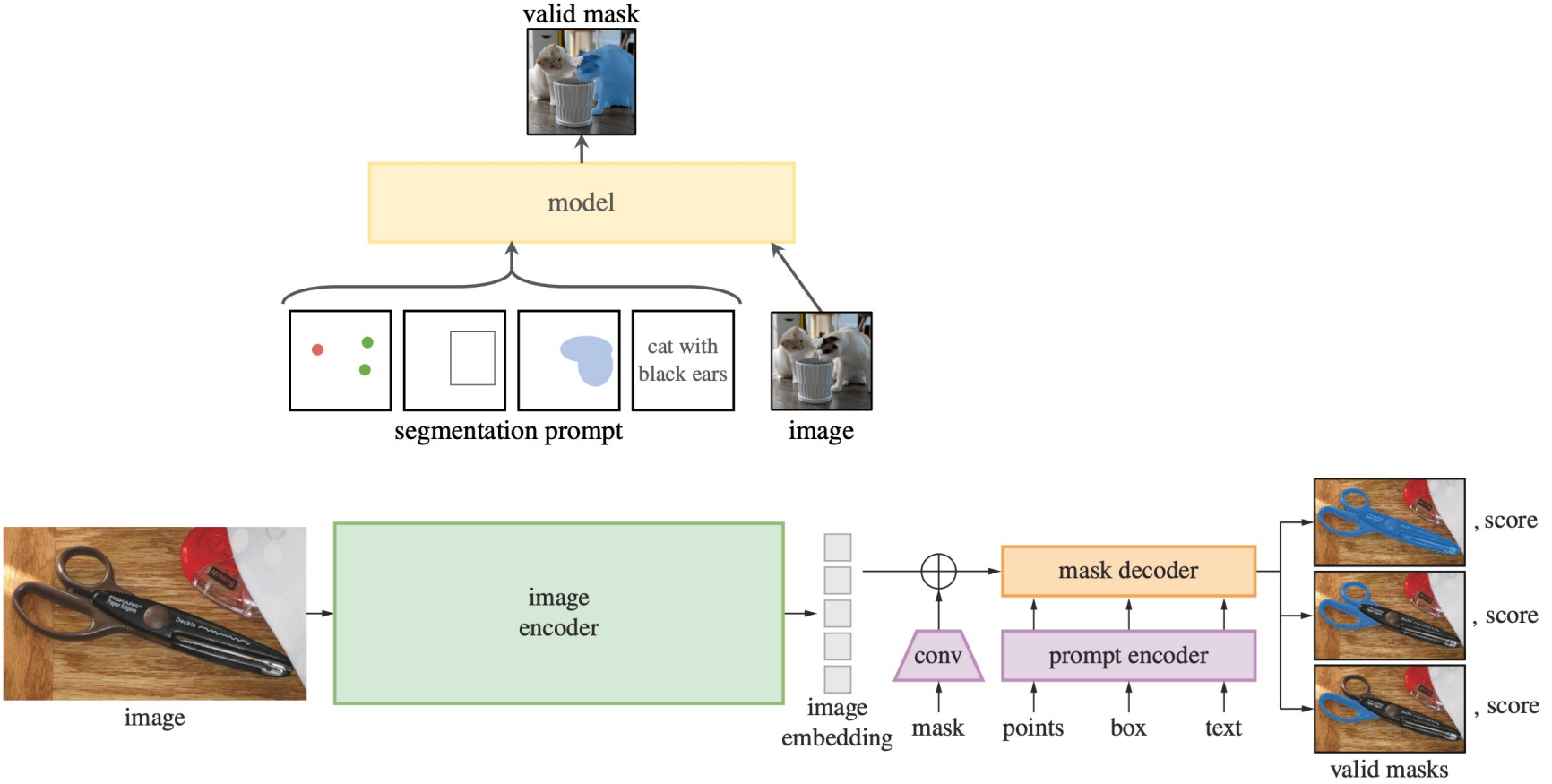
- Built with a SAM model in the loop
- 11M images with 1.1B segmentation masks
- 400x more masks than any prior segmentation dataset





https://www.cs.unc.edu/~ronisen/teaching/fall_2024/intro2vision_fall2024.html

Segment Anything Model (SAM)

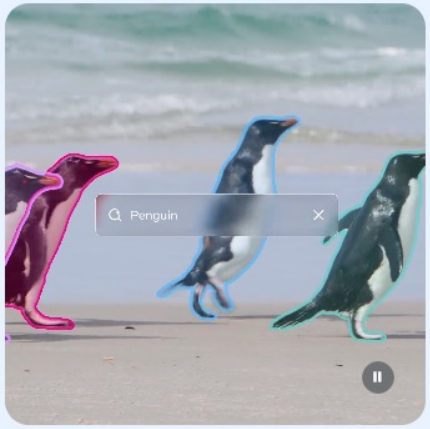


https://arxiv.org/abs/2304.02643

Lab 8d: Segment Anything

Duration: 10 min

[SOTA] Segment Anything

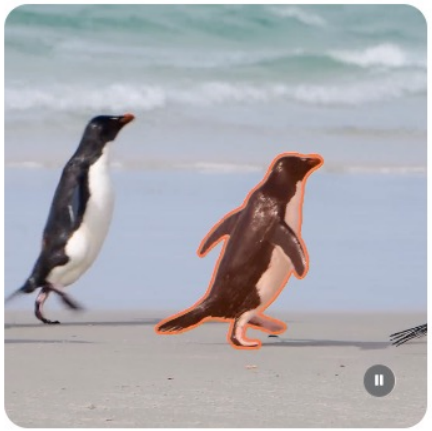


SAM 3

Detect, segment and track every example of any object category in an image or video, using text or examples

- Segment an object from a click
- Track segmented objects in videos
- Refine prediction with follow up clicks
- **Detect and segment matching instances from text**
- **Refine detection with visual examples**

ICLR 2026

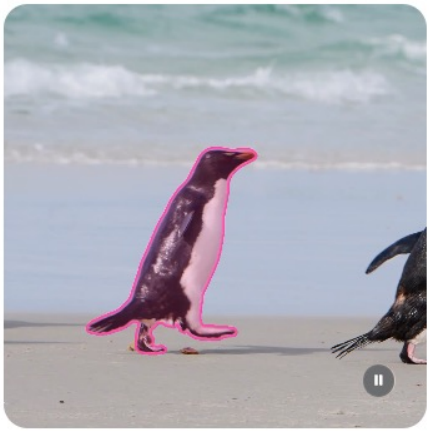


SAM 2

Segment and track any object in any image or video using click, box or mask prompts

- Segment an object from a click
- Track segmented objects in videos
- Refine prediction with follow up clicks

ICLR 2025



SAM 1

Segment any object in any image with as little as a single click

- Segment an object from a click
- Refine prediction with follow up clicks

JUNE 18-22, 2023 CVPR VANCOUVER, CANADA

https://ai.meta.com/research/sam3/

Lab 8e: **Segment Anything 3 (SAM3)**

Duration: 10 min

[SOTA] Object Recognition and Segmentation with LLMs

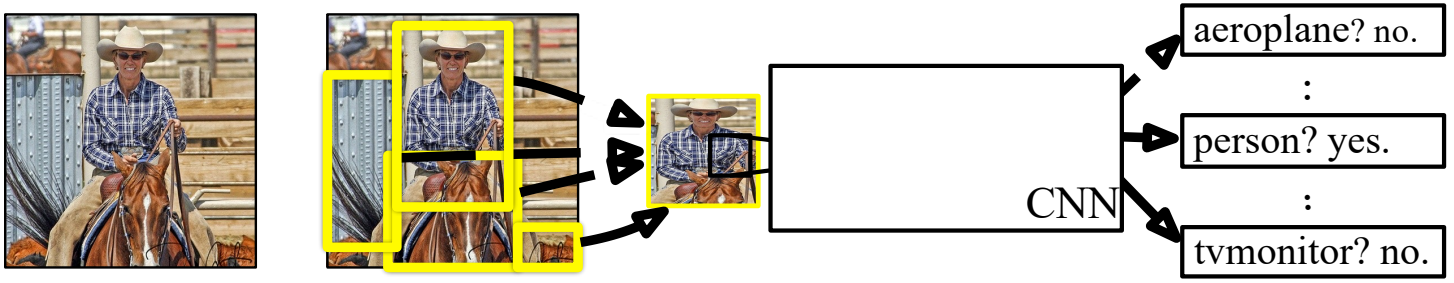
https://github.com/roboflow/notebooks

Lab 8f: Zero-Shot Object Detection and Segmentation with Google Gemini 2.5

Duration: 10 min

Extra

R-CNN at test time: Step 1



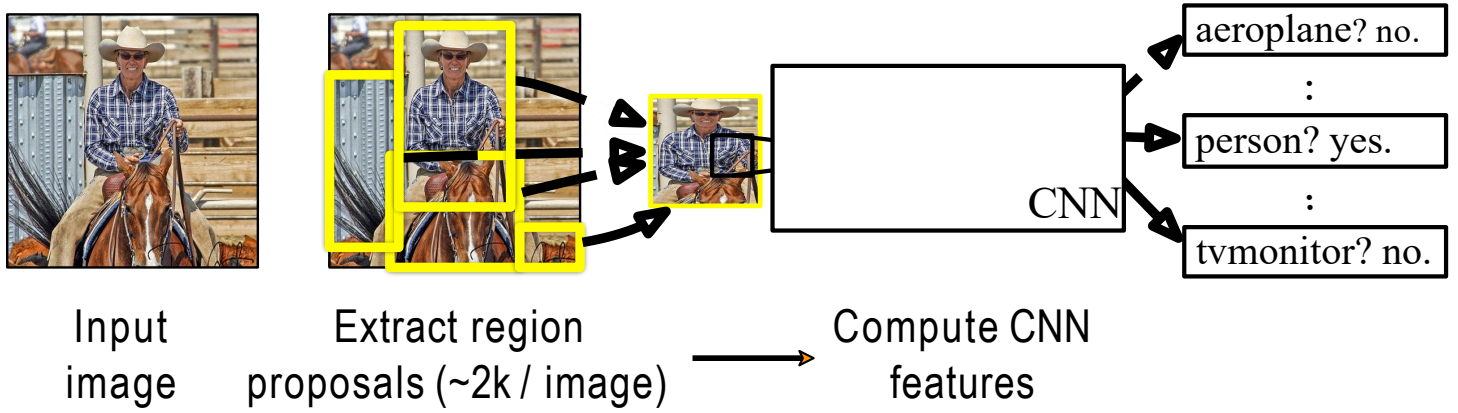
Input image → Extract region proposals (~2k / image)

Proposal-method agnostic, many choices

- Selective Search [van de Sande, Uijlings et al.] (Used in this work)
- Objectness [Alexe et al.]
- Category independent object proposals [Endres & Hoiem]
- CPMC [Carreira & Sminchisescu]

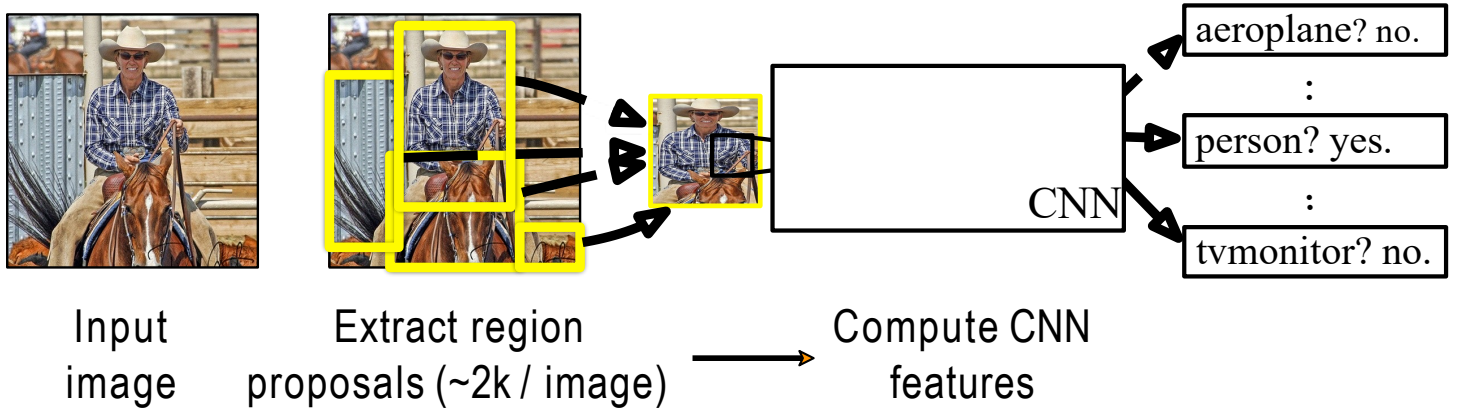
Girshick et al., "Rich Feature Hierarchies for Accurate Object Detection and Semantic Segmentation", CVPR 2014

R-CNN at test time: Step 2



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

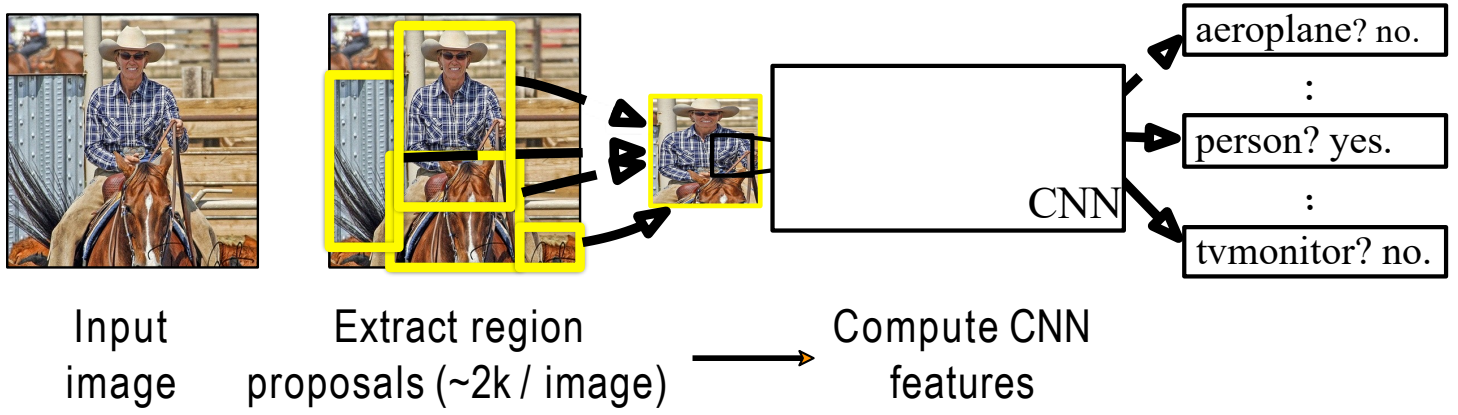
R-CNN at test time: Step 2



Dilate proposal

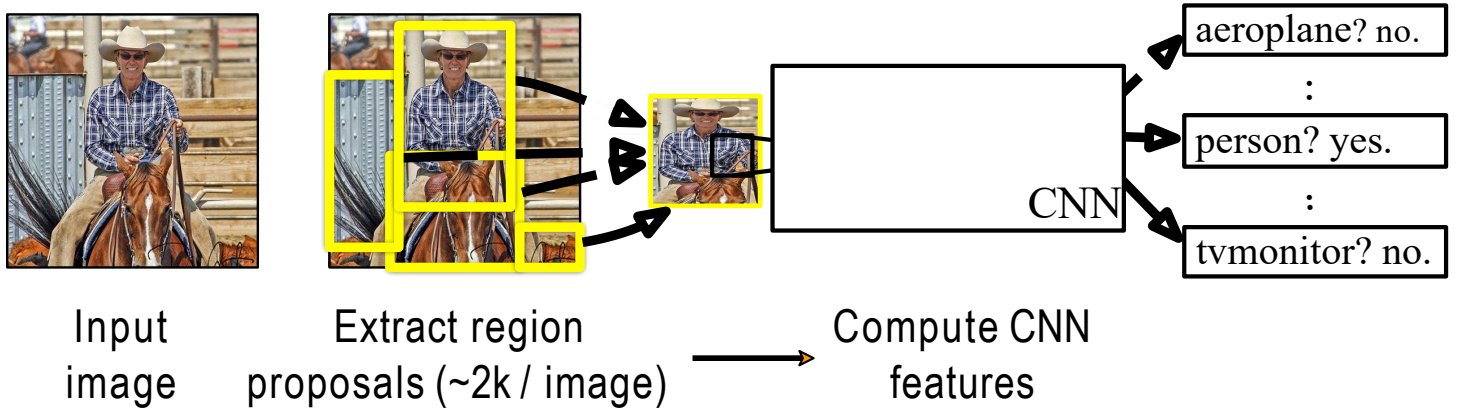
Girshick et al., "Rich Feature Hierarchies for Accurate Object Detection and Semantic Segmentation", CVPR 2014

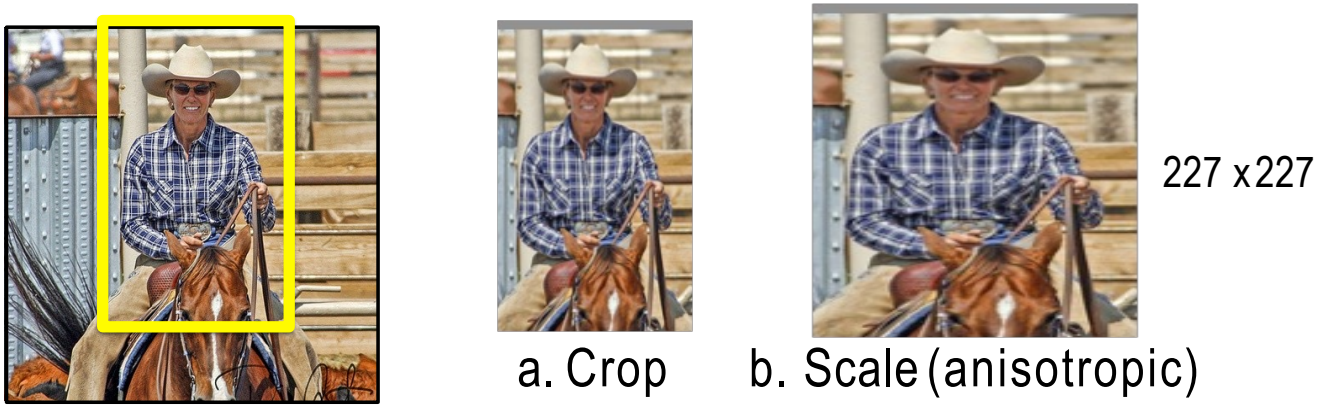
R-CNN at test time: Step 2



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

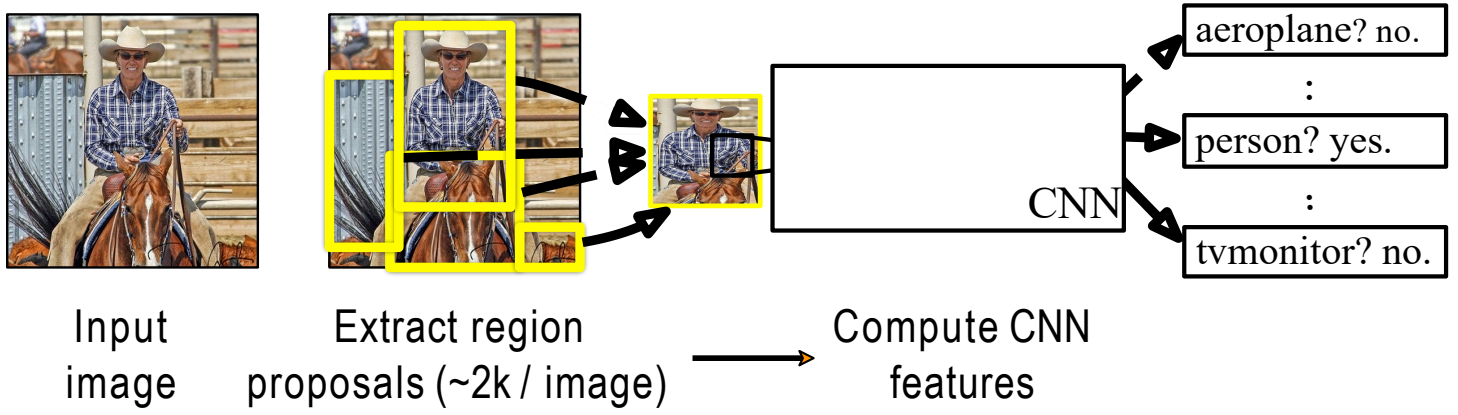
R-CNN at test time: Step 2

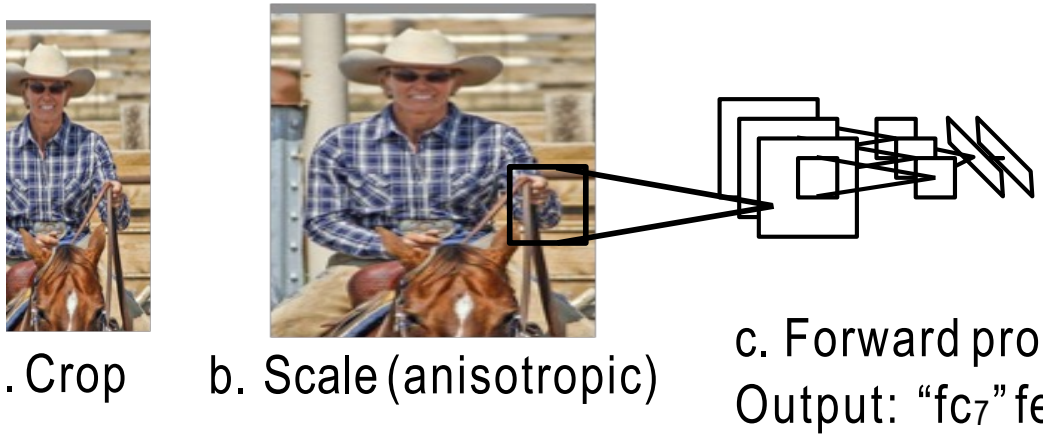




Girshick et al., "Rich Feature Hierarchies for Accurate Object Detection and Semantic Segmentation", CVPR 2014

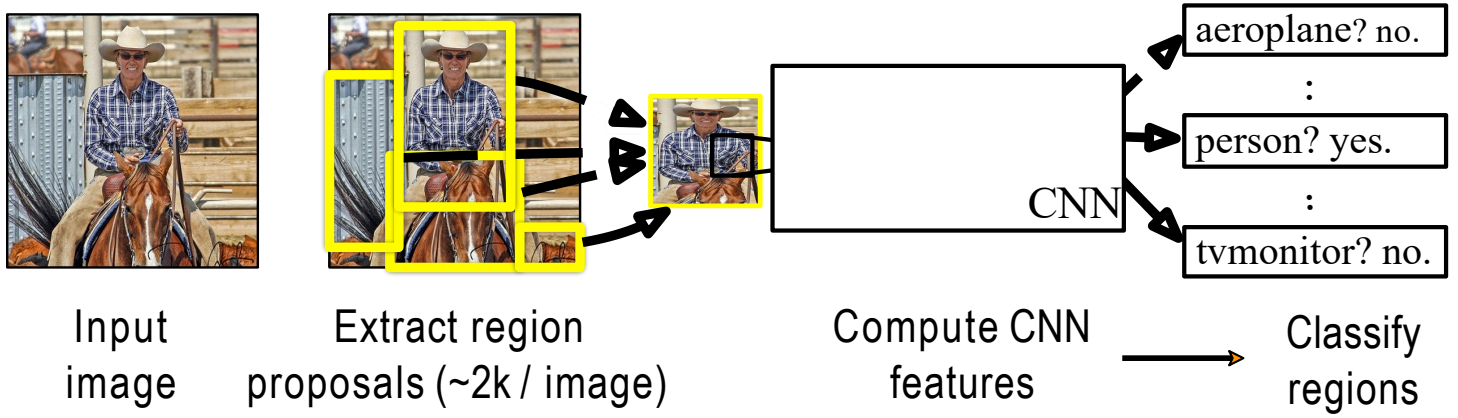
R-CNN at test time: Step 2

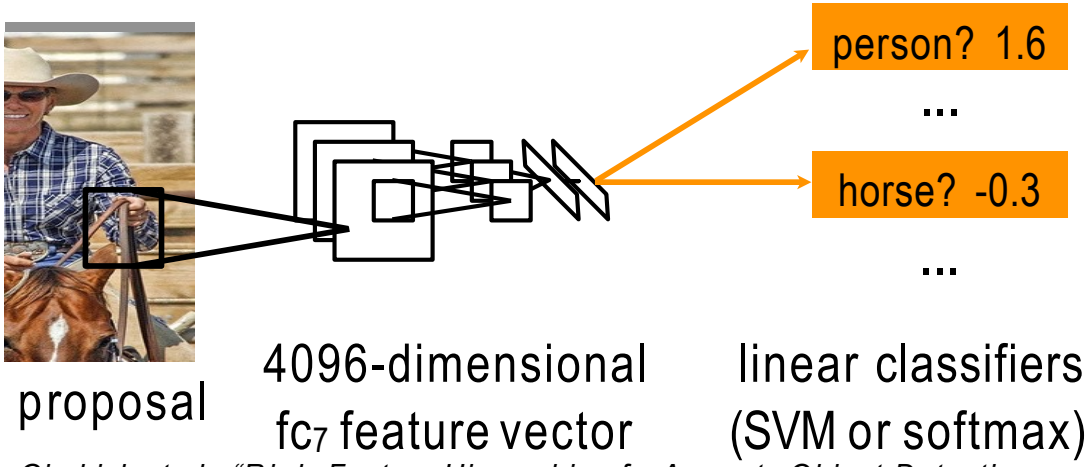




Girshick et al., "Rich Feature Hierarchies for Accurate Object Detection and Semantic Segmentation", CVPR 2014

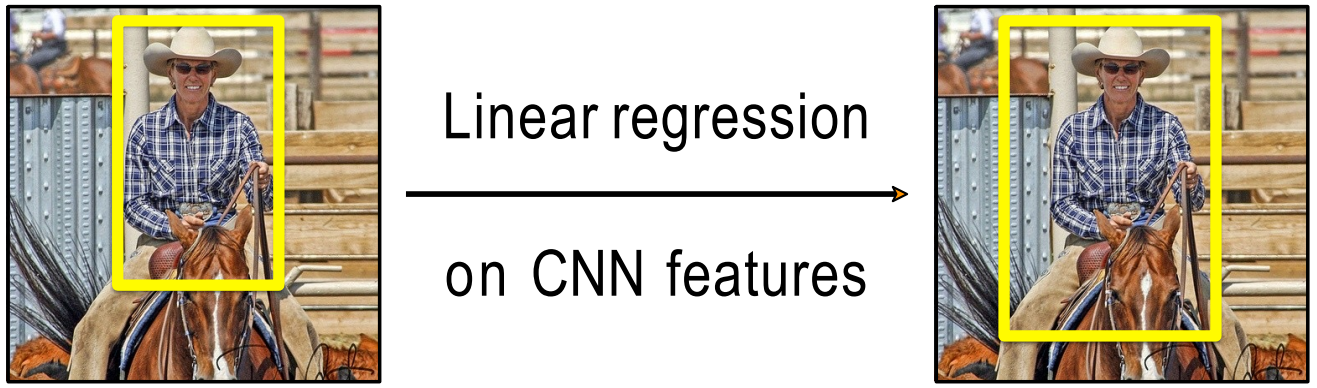
R-CNN at test time: Step 3





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

Step 4: Object proposal refinement



Bounding-box regression

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