Object Detection

Tomasz Stańczyk
tomasz.stanczyk@inria.fr

With slides from Andrew Ng and other sources (referenced)

Inria

3iA Côte d'Azur
Interdisciplinary Institute for Artificial Intelligence

Today's Agenda

Object detection fundamentals - based on DeepLearningAI materials by Andrew Ng

+ references for more information/self-study if desired

YOLOX object detection algorithm

+ references for more information/self-study if desired

Briefly about a few more recent approaches

+ discovering even more encouraged

Later today: YOLOX-based practical assignment

DeepLearning.AI

LearnOpenCV.com

EXPERT AI BLOG

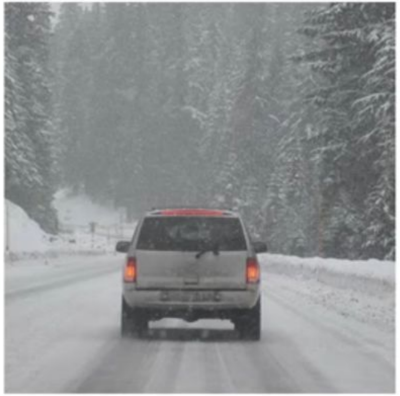
Object Detection Fundamentals

Selected slides by Andrew Ng

What are localization and detection?

Object Localization

Image classification

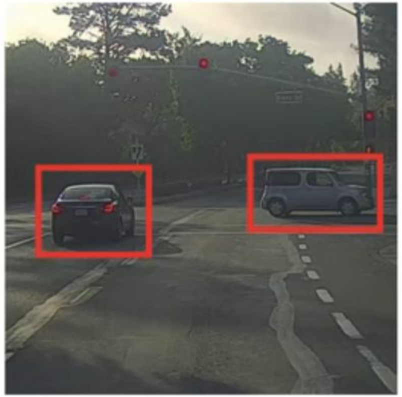


"Car"

Classification with localization

"Car"

Detection

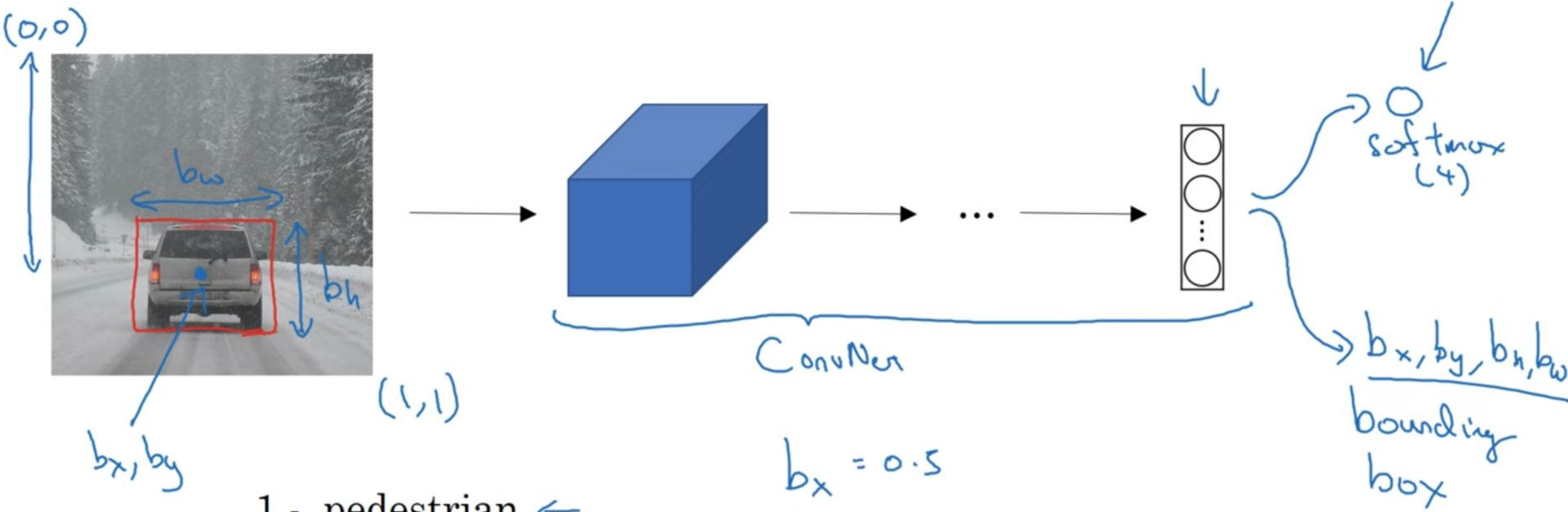


multiple objects

1 object

Classification with localization

Object Localization



1 - pedestrian

2 - car

3 - motorcycle

4 - $\boxed{\text{background}}$

$b_x = 0.5$

$b_y = 0.7$

$b_h = 0.3$

$b_w = 0.4$

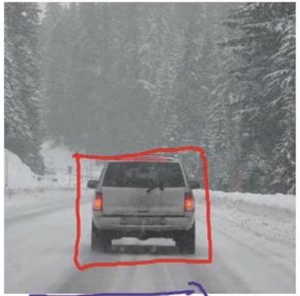
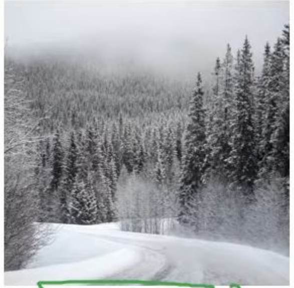
Andrew Ng

Defining the target label y

Object Localization

$$\begin{bmatrix} 1 - \text{pedestrian} \\ 2 - \text{car} \\ 3 - \text{motorcycle} \\ 4 - \text{background} \end{bmatrix}$$

Need to output b_x, b_y, b_h, b_w, class label (1-4)

$x =$  

$$\mathcal{L}(\hat{y}, y) = \begin{cases} (\hat{y}_1 - y_1)^2 + (\hat{y}_2 - y_2)^2 + \dots + (\hat{y}_8 - y_8)^2 & \text{if } y_1 = 1 \\ (\hat{y}_1 - y_1)^2 & \text{if } y_1 = 0 \end{cases}$$

$$y = \begin{bmatrix} P_c \\ b_x \\ b_y \\ b_h \\ b_w \\ c_1 \\ c_2 \\ c_3 \end{bmatrix}$$

is there any object?

$$(x, y) \quad \begin{bmatrix} 1 \\ b_x \\ b_y \\ b_h \\ b_w \\ 0 \\ 1 \\ 0 \end{bmatrix} \qquad \begin{bmatrix} 0 \\ ? \\ ? \\ ? \\ ? \\ ? \\ ? \\ ? \end{bmatrix}$$

P_c

"don't care"

Andrew Ng

Car detection example

Object Detection

Training set:

x	y
	1
	1
	1
	0
	0

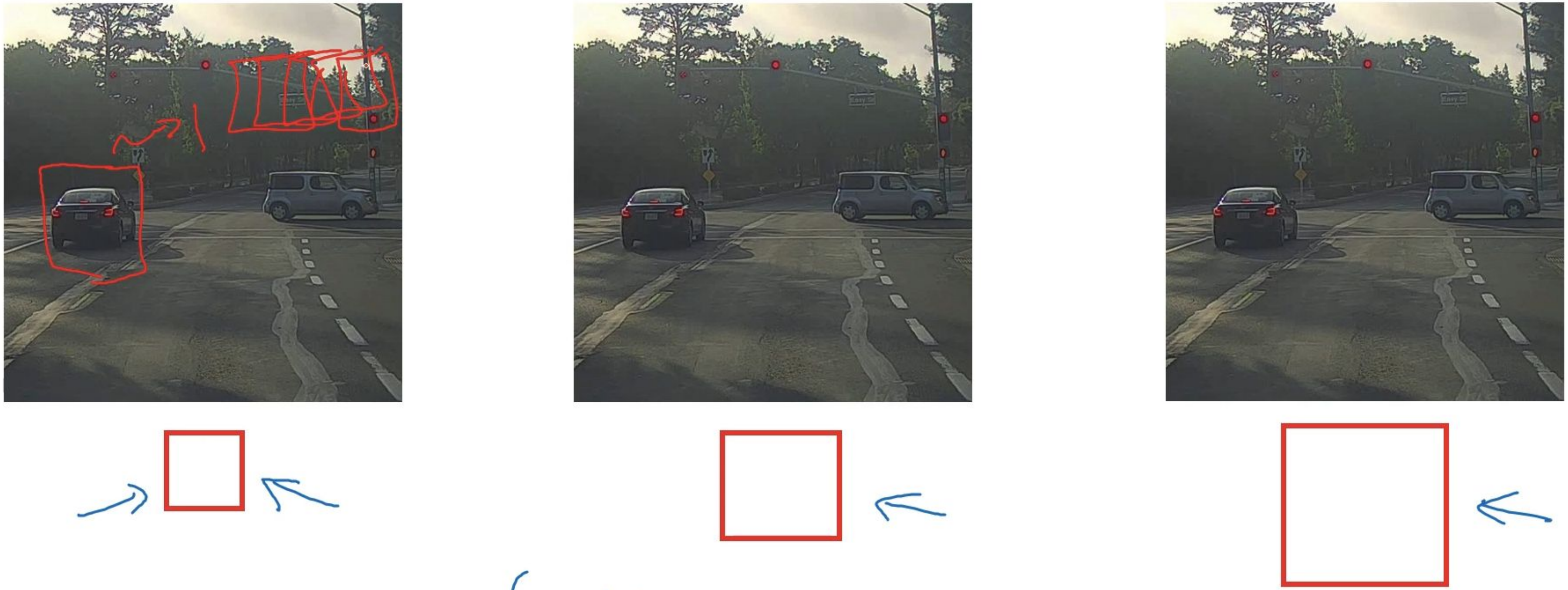
 → ConvNet → y

Sliding windows detection

Object Detection

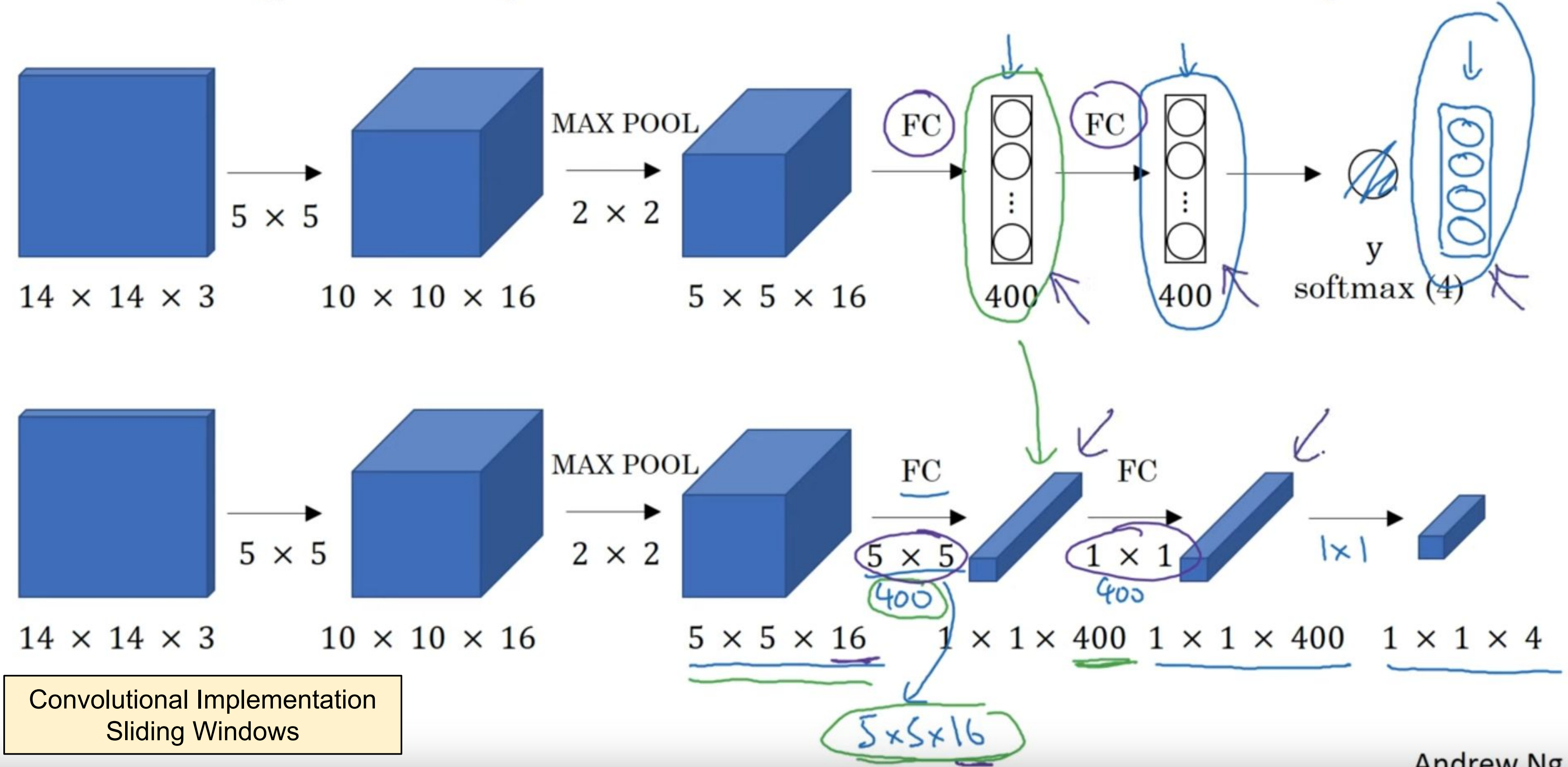
→ ConvNet → 0

→ ConvNet



Computation cost

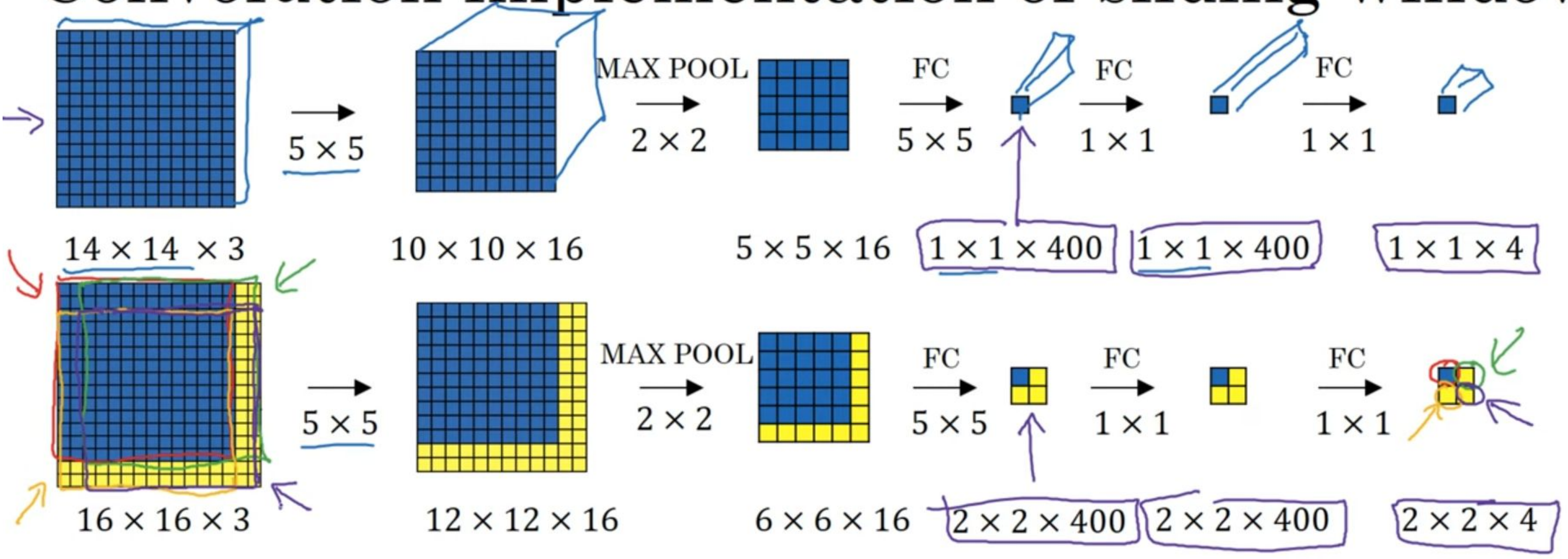
Turning FC layer into convolutional layers



Convolutional Implementation
Sliding Windows

Andrew Ng

Convolution implementation of sliding windows

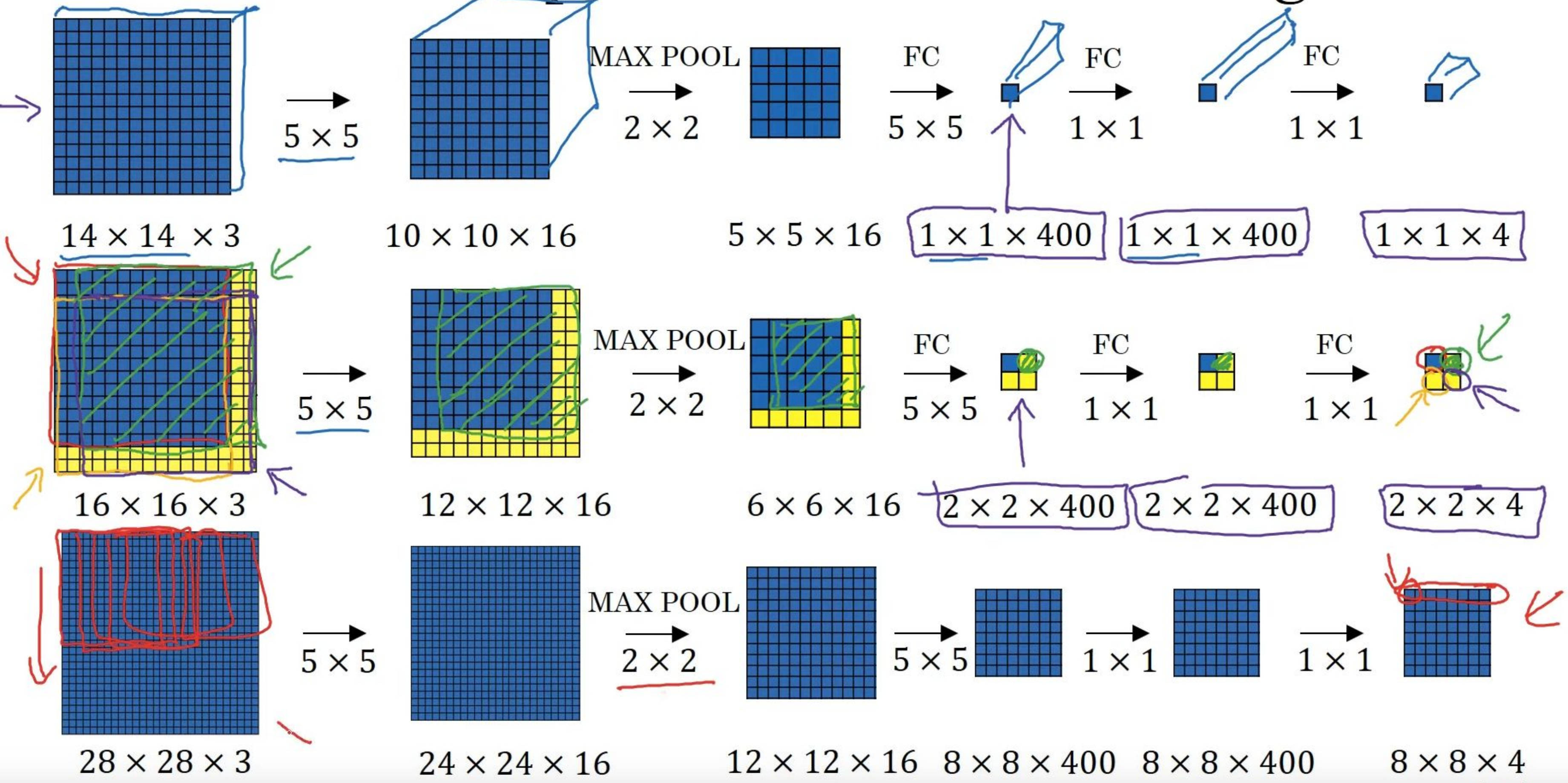


Convolutional Implementation
Sliding Windows

[Sermanet et al., 2014, OverFeat: Integrated recognition, localization and detection using convolutional networks]

Andrew Ng

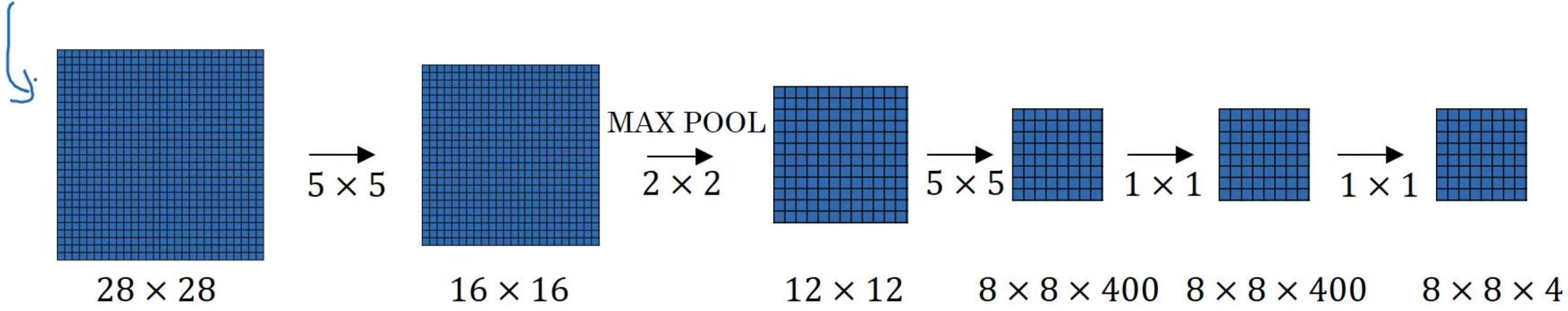
Convolution implementation of sliding windows

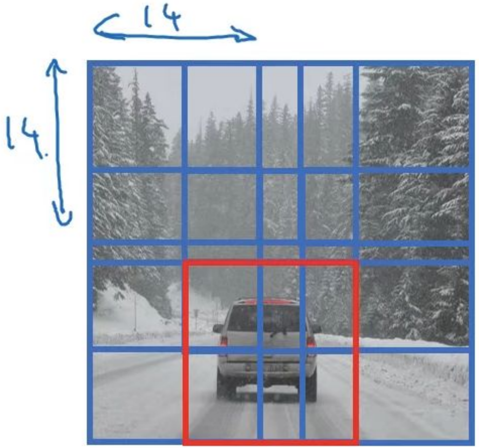


[Sermanet et al., 2014, OverFeat: Integrated recognition, localization and detection using convolutional networks]

Andrew Ng

Convolution implementation of sliding windows

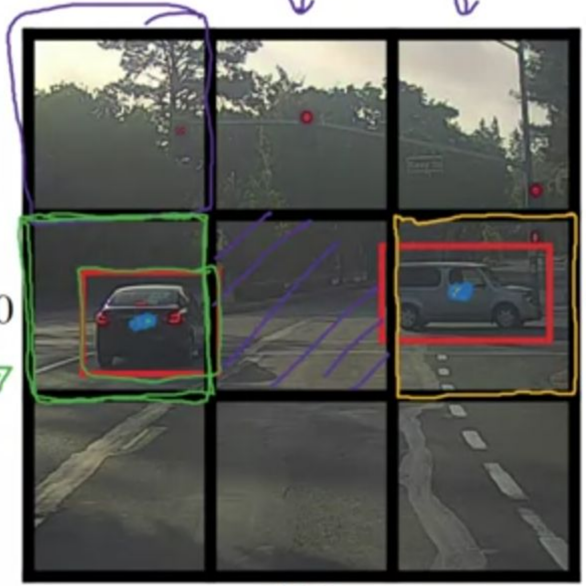




Convolutional Implementation
Sliding Windows

YOLO algorithm

Bounding Box Predictions



100

100

Labels for training

For each grid cell:

$$y = \begin{bmatrix} p_c \\ b_x \\ b_y \\ b_h \\ b_w \\ c_1 \\ c_2 \\ c_3 \end{bmatrix} \quad \begin{bmatrix} 0 \\ ? \\ ? \\ ? \\ ? \\ ? \\ ? \\ ? \end{bmatrix} \quad \begin{bmatrix} 1 \\ b_x \\ b_y \\ b_h \\ b_w \\ 0 \\ 1 \\ 0 \end{bmatrix} \quad \begin{bmatrix} 1 \\ b_x \\ b_y \\ b_h \\ b_w \\ 0 \\ 1 \\ 0 \end{bmatrix}$$

Target output: $3 \times 3 \times 8$

$\underline{x}$ ($100 \times 100 \times 3$) → CONV → MAXPOOL → .... → ($3 \times 3 \times 8$) y

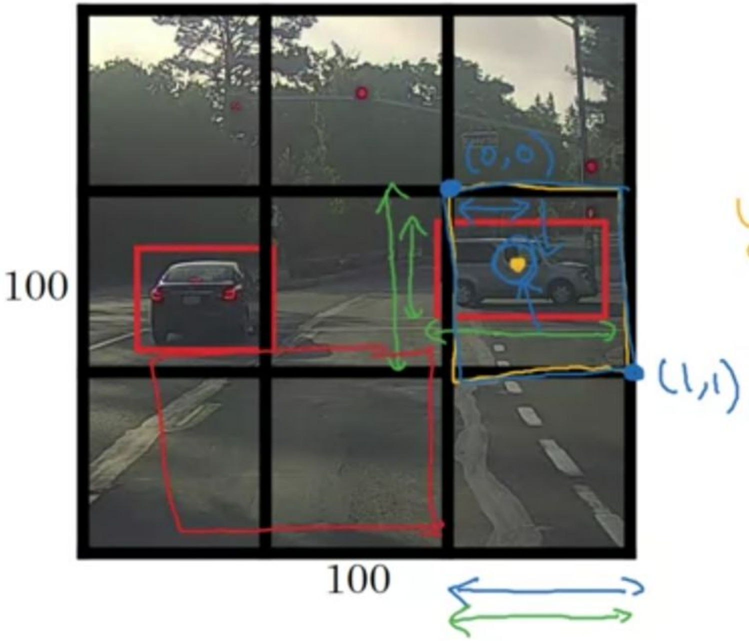
361

→ $19 \times 19 \times 8$

[Redmon et al., 2015, You Only Look Once: Unified real-time object detection]

Andrew Ng

Specify the bounding boxes



$$y = \begin{bmatrix} 1 \\ b_x \\ b_y \\ b_h \\ b_w \\ 0 \\ 1 \\ 0 \end{bmatrix}$$

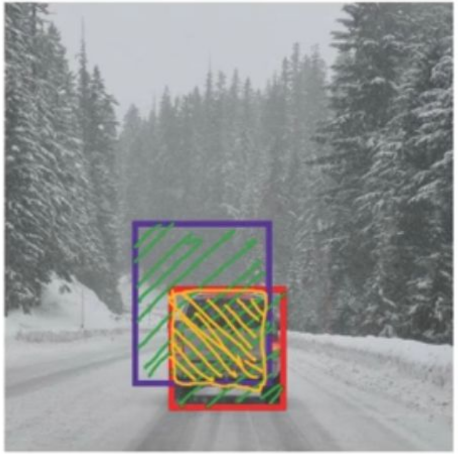
$\left.\begin{matrix} 0.4 \\ 0.3 \end{matrix}\right\}$ between 0 and 1

$\left.\begin{matrix} 0.9 \\ 0.5 \end{matrix}\right\}$ could be >1

[Redmon et al., 2015, You Only Look Once: Unified real-time object detection]

Evaluating object localization

Intersection Over Union



Intersection over Union (IoU)

$$= \frac{\text{size of}}{\text{size of}}$$

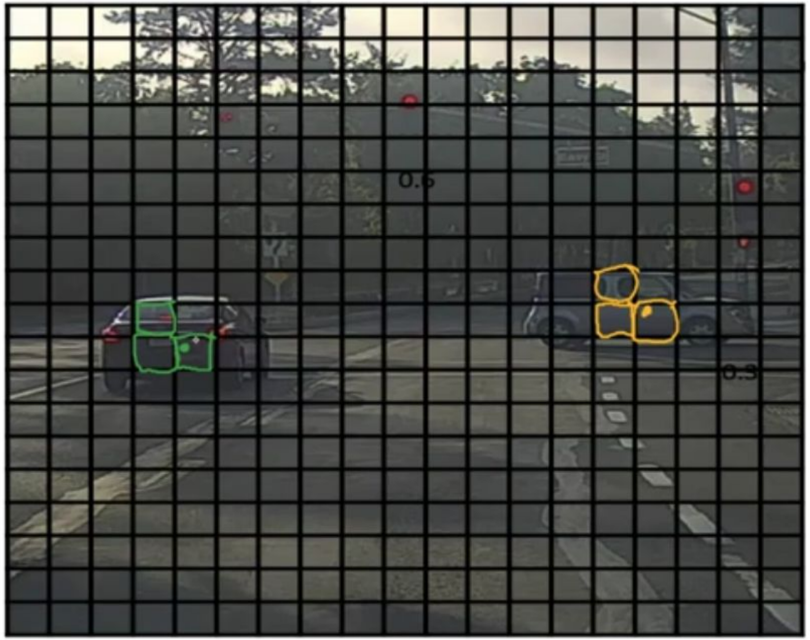
"Correct" if IoU $\geq \boxed{0.5}$

0.6

More generally, IoU is a measure of the overlap between two bounding boxes.

Non-max suppression example

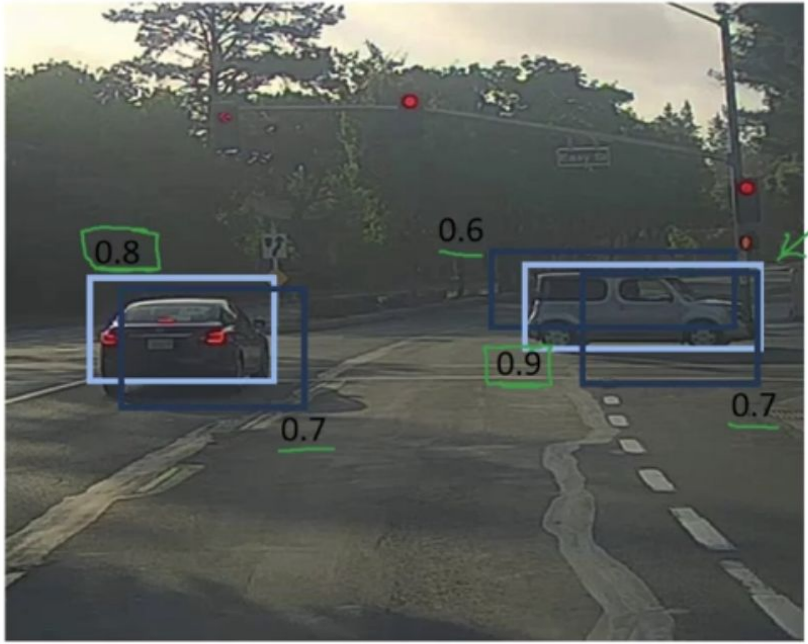
Non-max
Suppression



19x19

Non-max suppression example

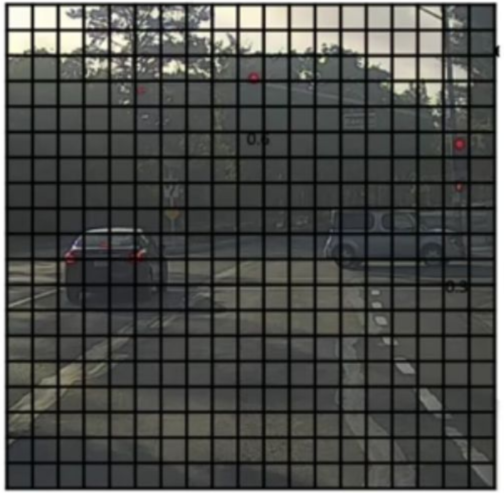
Non-max Suppression



$\underline{p_c}$

Non-max suppression algorithm

Non-max Suppression



19×19

Each output prediction is: $\begin{bmatrix} p_c \\ b_x \\ b_y \\ b_h \\ b_w \end{bmatrix}$

c_1
c_2
c_3

Discard all boxes with $p_c \leq 0.6$

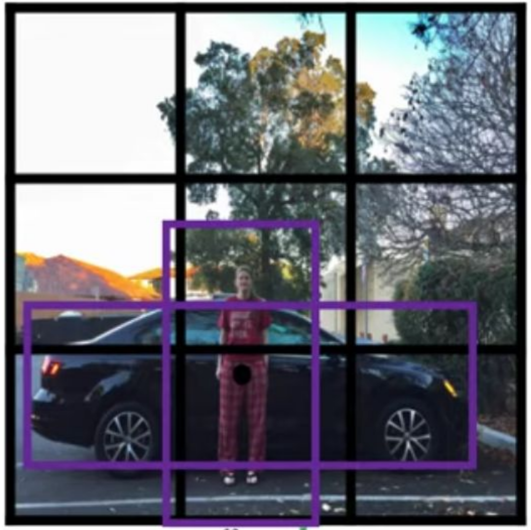
While there are any remaining boxes:

- Pick the box with the largest p_c
  Output that as a prediction.
- Discard any remaining box with
  IoU ≥ 0.5 with the box output
  in the previous step

Andrew Ng

Overlapping objects:

Anchor Boxes



$$y = \begin{bmatrix} p_c \\ b_x \\ b_y \\ b_h \\ b_w \\ c_1 \\ c_2 \\ c_3 \end{bmatrix}$$

Anchor box 1:

Anchor box 2:

$$y = \begin{bmatrix} p_c \\ b_x \\ b_y \\ b_h \\ b_w \\ c_1 \\ c_2 \\ c_3 \\ p_c \\ b_x \\ \vdots \\ c_3 \end{bmatrix} \begin{matrix} \left.\begin{matrix} \\ \\ \\ \\ \\ \\ \\ \\ \end{matrix}\right\} \text{anchor box 1} \\ \left.\begin{matrix} \\ \\ \\ \\ \end{matrix}\right\} \text{anchor box 2} \end{matrix}$$

[Redmon et al., 2015, You Only Look Once: Unified real-time object detection]

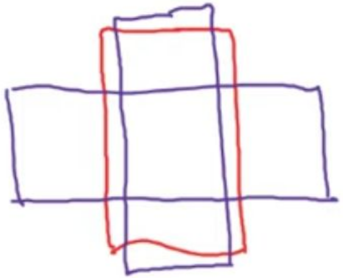
Anchor box algorithm

Previously:

Each object in training image is assigned to grid cell that contains that object's midpoint.

Output y:

$3 \times 3 \times 8$



With two anchor boxes:

Each object in training image is assigned to grid cell that contains object's midpoint and anchor box for the grid cell with highest IoU.

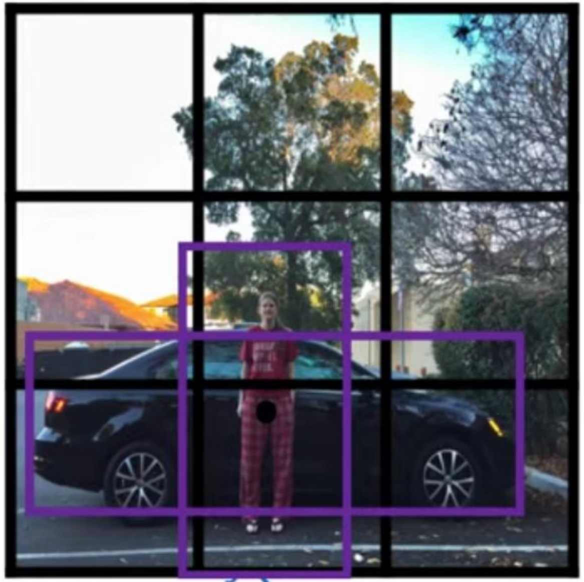
(grid cell, anchor box)

Output y:

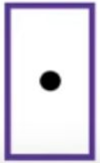
$3 \times 3 \times 16$

$3 \times 3 \times 2 \times 8$

Anchor box example



Anchor box 1: Anchor box 2:



car only?

$$y = \begin{bmatrix} p_c \\ b_x \\ b_y \\ b_h \\ b_w \\ c_1 \\ c_2 \\ c_3 \\ p_c \\ b_x \\ b_y \\ b_h \\ b_w \\ c_1 \\ c_2 \\ c_3 \end{bmatrix} \quad \begin{bmatrix} 1 \\ b_x \\ b_y \\ b_h \\ b_w \\ 1 \\ 0 \\ 0 \\ 1 \\ b_x \\ b_y \\ b_h \\ b_w \\ 0 \\ 1 \\ 0 \end{bmatrix} \quad \begin{bmatrix} 0 \\ ? \\ ? \\ ? \\ ? \\ ? \\ ? \\ ? \\ 1 \\ b_x \\ b_y \\ b_h \\ b_w \\ 0 \\ 1 \\ 0 \end{bmatrix} \begin{matrix} \left.\begin{matrix} \\ \\ \\ \\ \\ \\ \\ \\ \end{matrix}\right\} \text{anchor box 1} \\ \left.\begin{matrix} \\ \\ \\ \\ \\ \\ \\ \\ \end{matrix}\right\} \text{anchor box 2} \end{matrix}$$

Andrew Ng

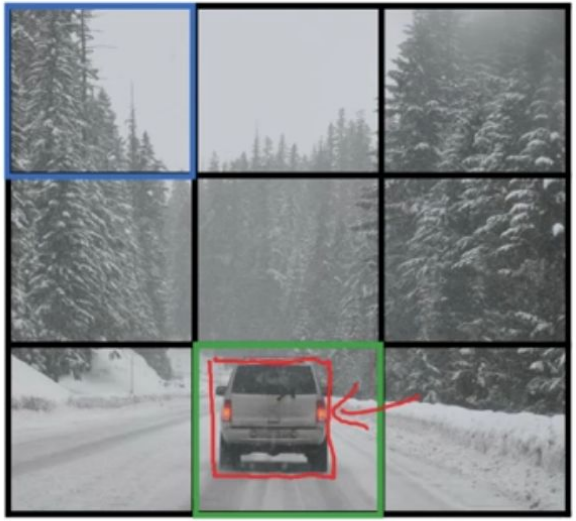
YOLO Algorithm

Training

1 - pedestrian

2 - car

3 - motorcycle



$$y = \begin{bmatrix} p_c \\ b_x \\ b_y \\ b_h \\ b_w \\ c_1 \\ c_2 \\ c_3 \\ p_c \\ b_x \\ b_y \\ b_h \\ b_w \\ c_1 \\ c_2 \\ c_3 \end{bmatrix} \quad \begin{bmatrix} 0 \\ ? \\ ? \\ ? \\ ? \\ ? \\ ? \\ ? \\ 0 \\ ? \\ ? \\ ? \\ ? \\ ? \\ ? \\ ? \end{bmatrix} \quad \begin{bmatrix} 0 \\ ? \\ ? \\ ? \\ ? \\ ? \\ ? \\ ? \\ 1 \\ b_x \\ b_y \\ b_h \\ b_w \\ 0 \\ 1 \\ 0 \end{bmatrix}$$

1 2

$3 \times 3 \times 16$

y is $\underbrace{3 \times 3}_{19 \times 19 \times 16,\; 19 \times 19 \times 40} \times 2 \times 8$

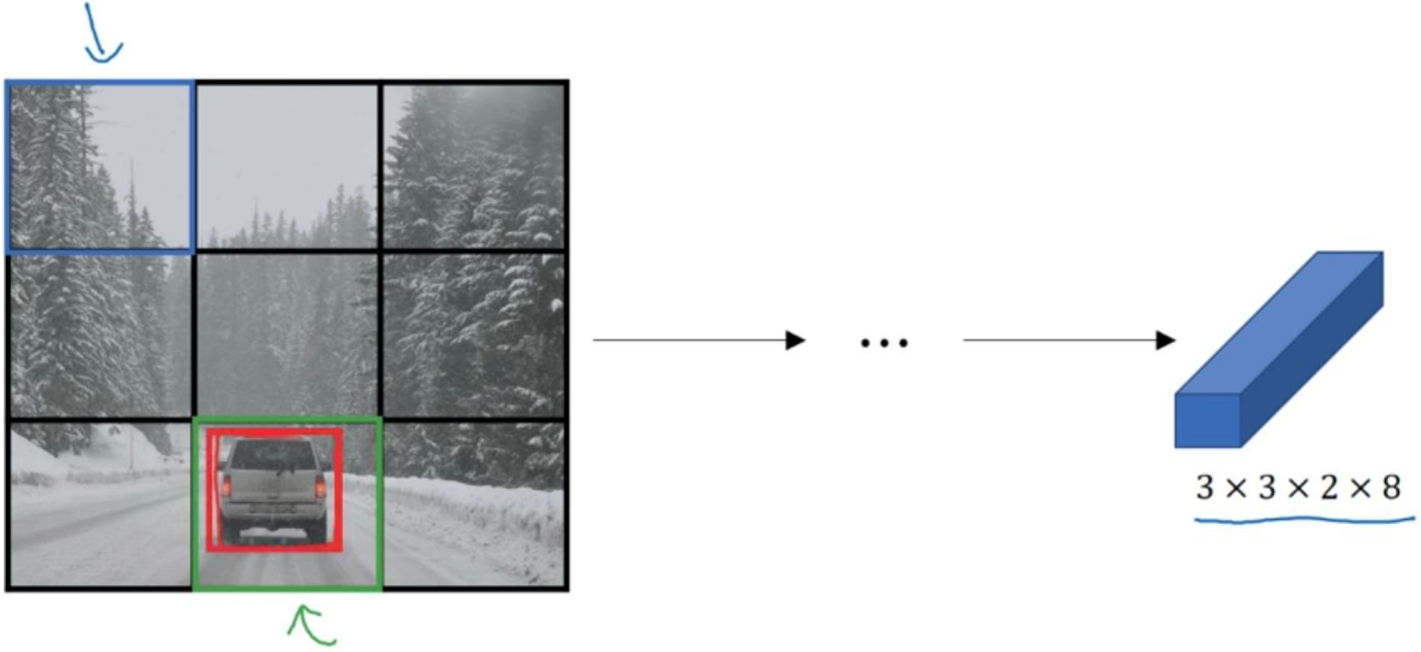
2: #anchors

8: 5 + #classes

[Redmon et al., 2015, You Only Look Once: Unified real-time object detection]

Andrew Ng

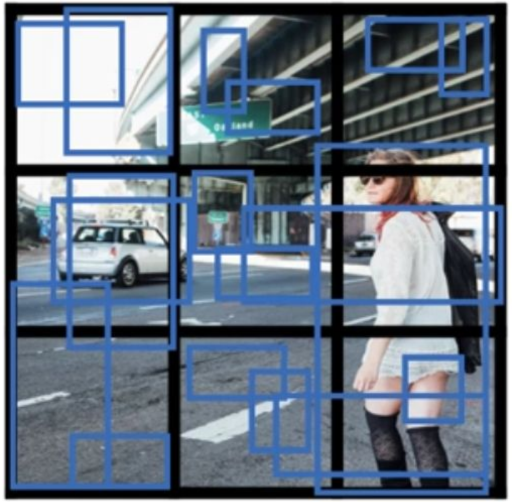
Making predictions



$$y = \begin{bmatrix} p_c \\ b_x \\ b_y \\ b_h \\ b_w \\ c_1 \\ c_2 \\ c_3 \\ p_c \\ b_x \\ b_y \\ b_h \\ b_w \\ c_1 \\ c_2 \\ c_3 \end{bmatrix} \quad \begin{bmatrix} 0 \\ \cdot \\ \cdot \\ \cdot \\ \cdot \\ \cdot \\ \cdot \\ \cdot \\ 0 \\ \cdot \\ \cdot \\ \cdot \\ \cdot \\ \cdot \\ \cdot \\ \cdot \end{bmatrix} \quad \begin{bmatrix} 0 \\ \cdot \\ \cdot \\ \cdot \\ \cdot \\ \cdot \\ \cdot \\ \cdot \\ 1 \\ b_x \\ b_y \\ b_h \\ b_w \\ 0 \\ 1 \\ 0 \end{bmatrix}$$

Outputting the non-max supressed outputs

YOLO
Algorithm



- For each grid call, get 2 predicted bounding boxes.

Outputting the non-max supressed outputs

YOLO Algorithm

- For each grid call, get 2 predicted bounding boxes.
- Get rid of low probability predictions.
- For each class (pedestrian, car, motorcycle) use non-max suppression to generate final predictions.

For more details…

Check the Andrew Ng's videos on object detection

Available on YouTube

See the following playlist:
https://www.youtube.com/playlist?list=PLkDaE6sCZn6Gl29AoE31iwdVwSG-KnDzF

Videos: C4W3L**01**, C4W3L**03**, C4W3L**04**, C4W3L**06**, C4W3L**07**, C4W3L**08**, C4W3L**09**

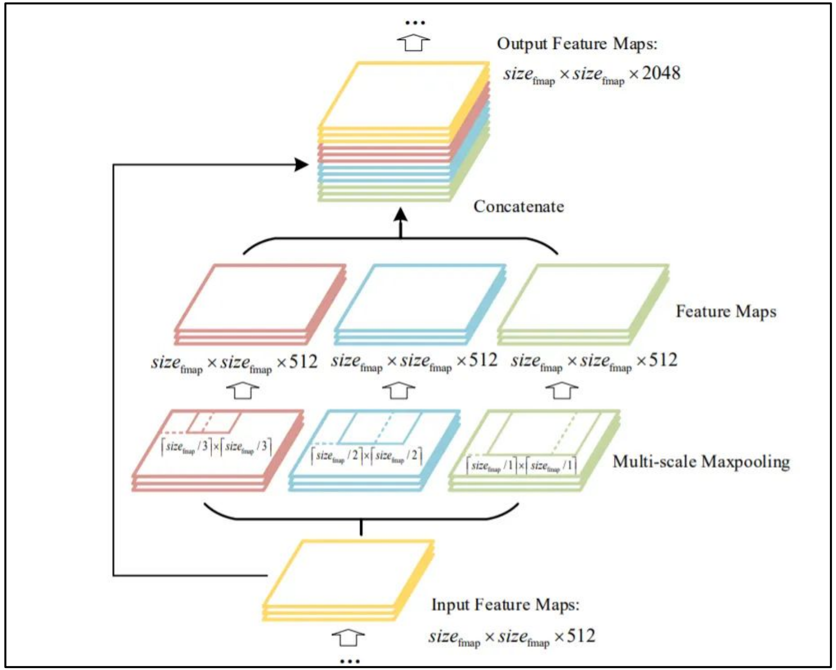
YOLOX
Object Detection Algorithm

Some information and figures
Courtesy of LearnOpenCV

YOLOX

Built on top of YOLOv3 with Darknet-53 backbone and SPP layer (YOLOv3-SPP)

Spatial Pyramid Pooling (SPP) layer →



Huang Z. et al. *DC-SPP-YOLO: Dense Connection and Spatial Pyramid Pooling Based YOLO for Object Detection.* 2019 https://arxiv.org/ftp/arxiv/papers/1903/1903.08589.pdf

YOLOX

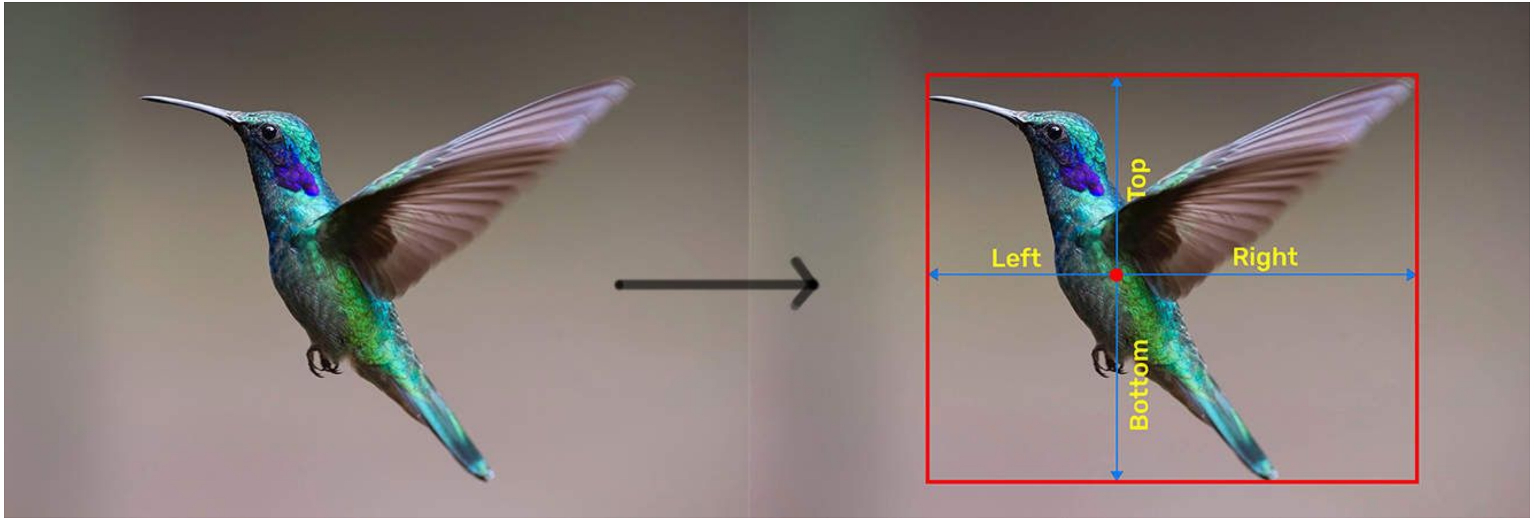
YOLOX’ distinctive features:

- Anchor free design
- Decoupled head
- simOTA label assignment strategy
- Advanced Augmentations: Mixup and Mosaic

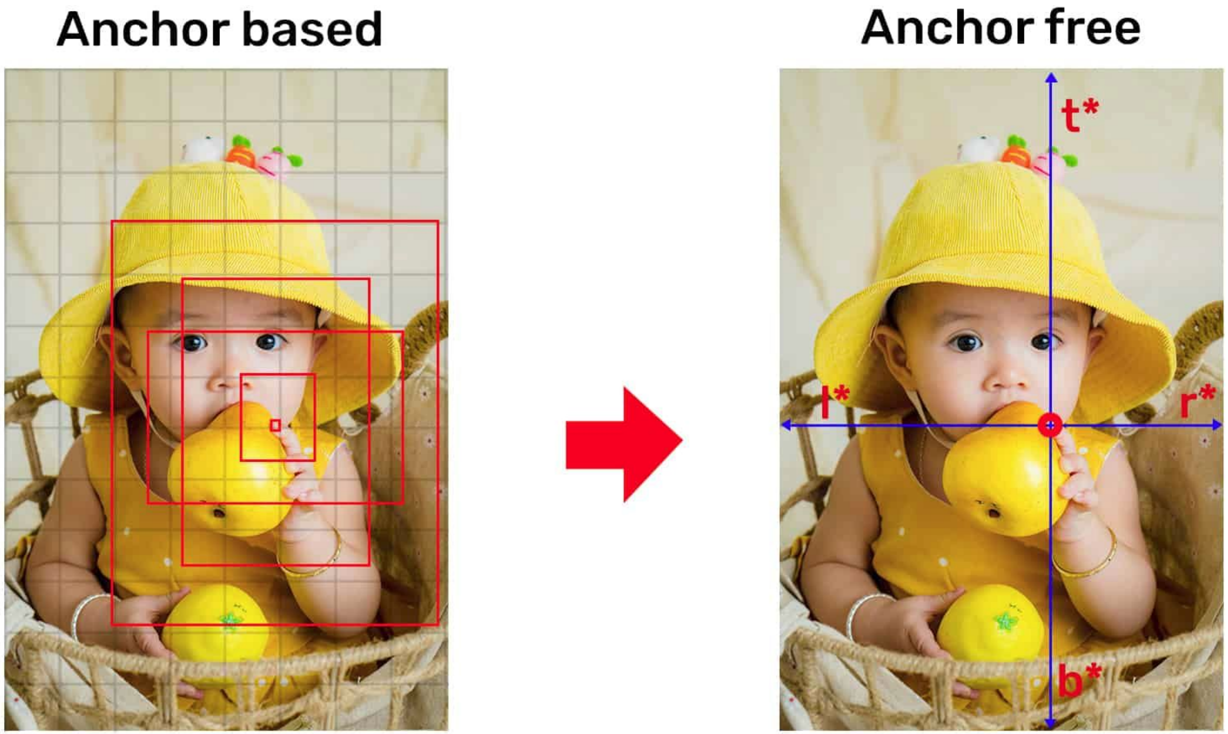
Center based detector

Find positive point in the center

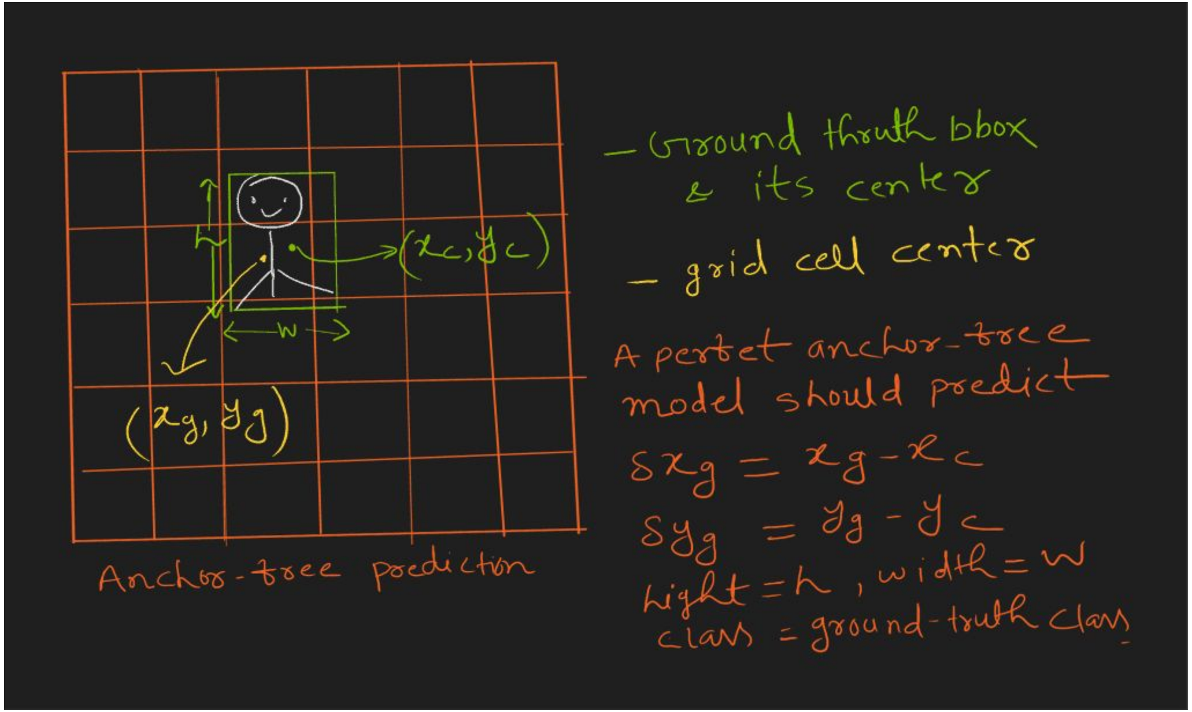
Predict four distances from the positive to the boundary



Anchor Free YOLOX



What is Anchor Free Object Detection

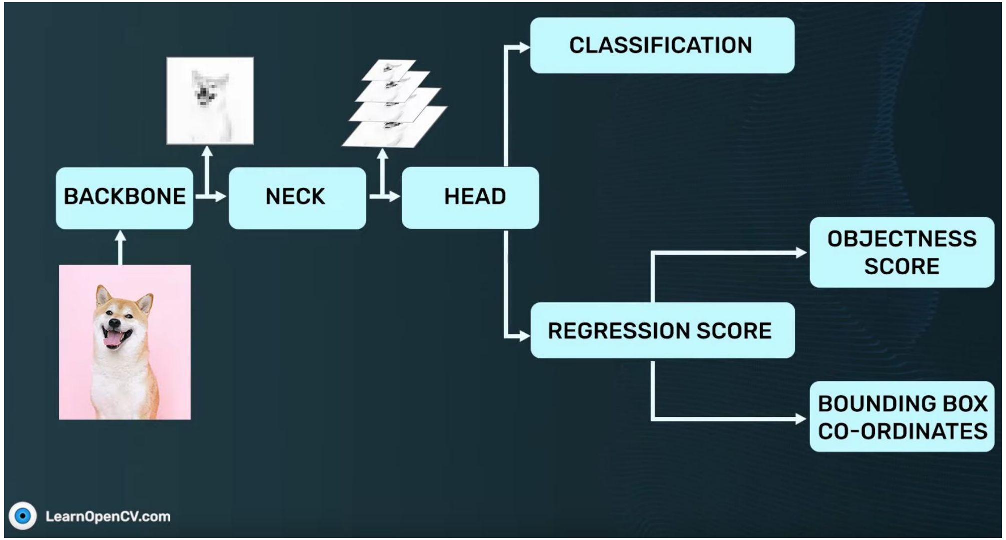


YOLO architecture

Backbone: extracts features of an image

Neck: producing feature maps with multiple scales

Head: outputs localization and classification scores



YOLOX - decoupled head

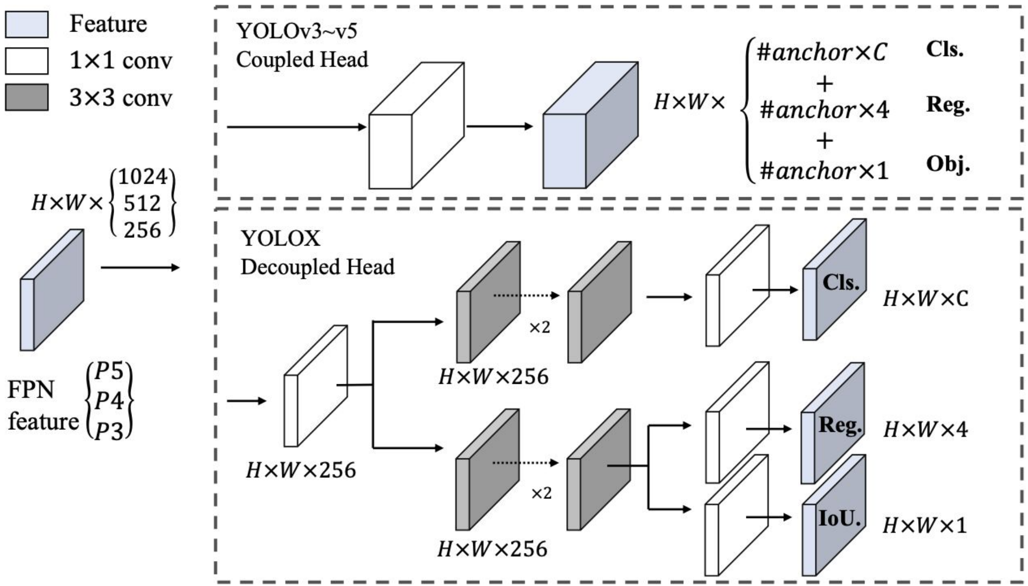
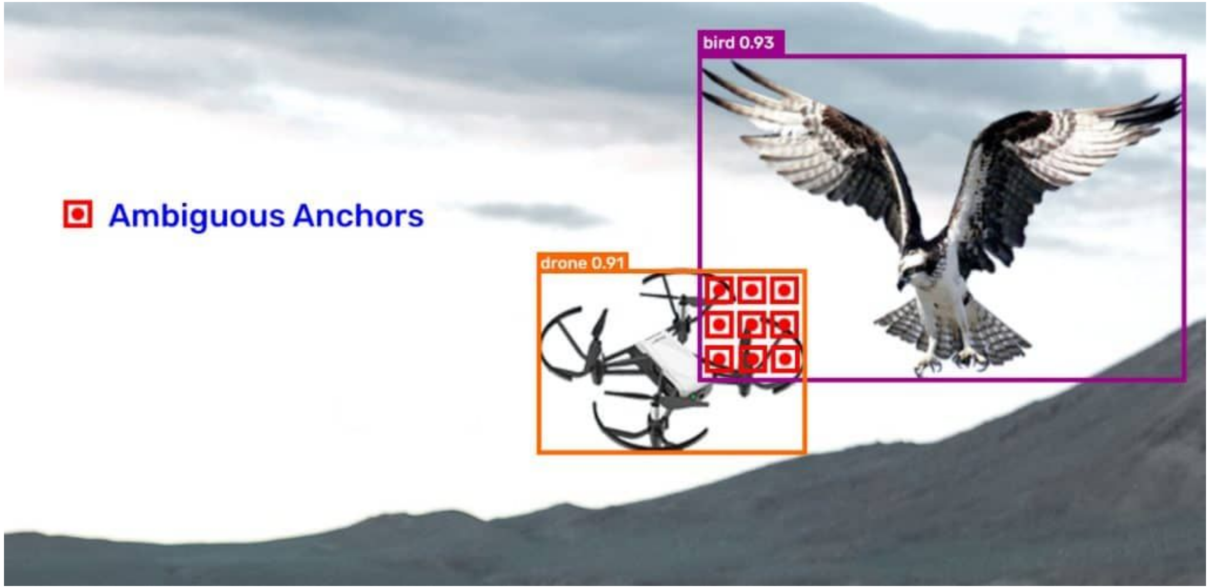


Figure 2: Illustration of the difference between YOLOv3 head and the proposed decoupled head. For each level of FPN feature, we first adopt a 1×1 conv layer to reduce the feature channel to 256 and then add two parallel branches with two 3×3 conv layers each for classification and regression tasks respectively. IoU branch is added on the regression branch.

SimOTA Advanced Label Assignment Strategy

OTA: Optimal Transport Assignment for Object Detection



SimOTA Advanced Label Assignment Strategy

Briefly explained by LearnOpenCV

What is simOTA in YOLOX?

Simplified OTA or simOTA is the redesigned Optimal Transport Assignment strategy. The training cost does not increase but average precision(AP) is definitely improved. It is shown with empirical evidence in the paper.

In simOTA, iteration is not performed for every positive label. A strategy called **Dynamic Top K** is used to estimate the approximate number of positive anchors for each ground truth. Here, only the top **K** number of positive labels are selected. This reduces the number of iterations by many folds.

The number of positive labels per ground truth (GT) varies due to the following factors.

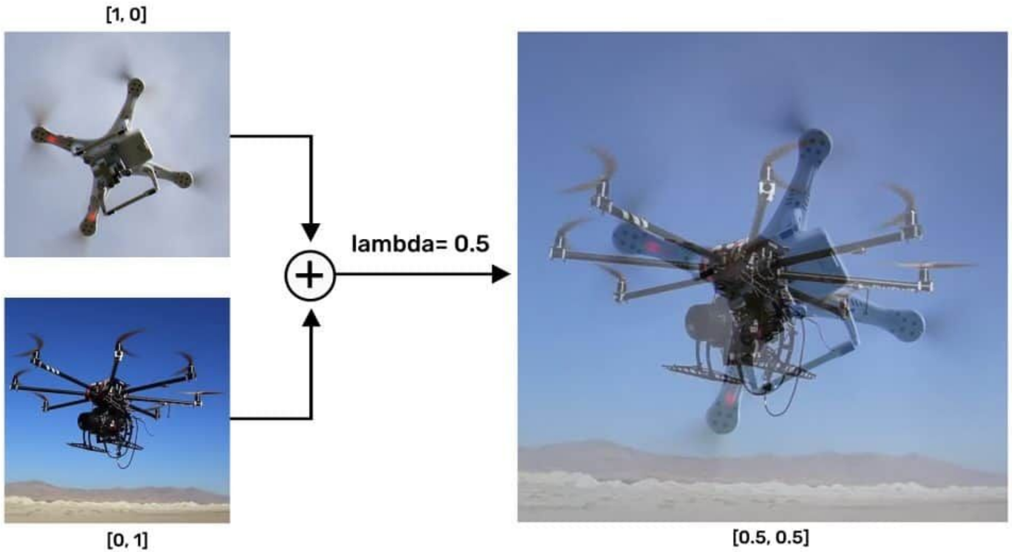
- Size
- Scale
- Occlusion conditions etc.

However, it is difficult to model a mapping function from these factors to the positive anchor number **K**. Hence it is done on the basis of IoU value. The IoU values of the **anchors** to the ground truth(GT) are summed up to represent the GT's estimated number of positive anchors.

The intuition is such that the number of positive anchors for a certain GT should be positively correlated with the number of anchors that have well regressed.

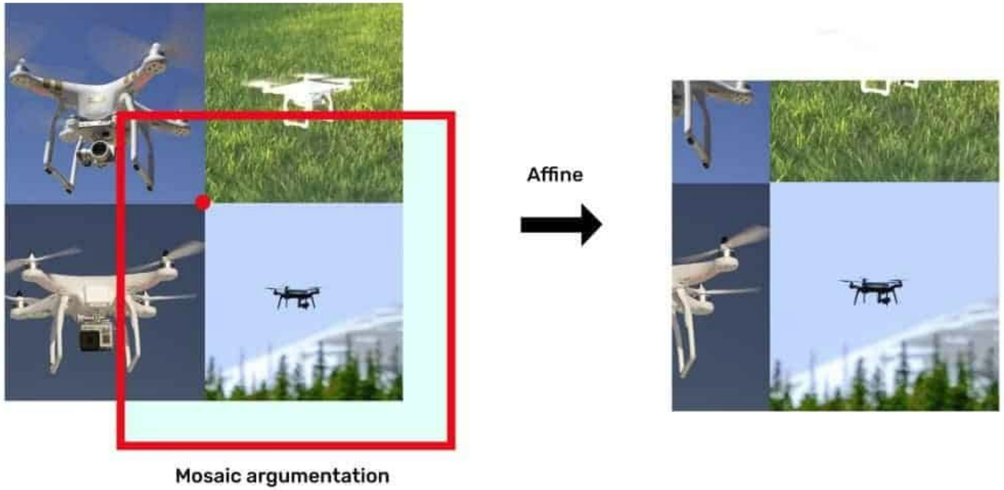
Strong Data Augmentation in YOLOX

Mixup Augmentation



Strong Data Augmentation in YOLOX

Mosaic Augmentation



Performance gain - step by step

Methods	AP (%)	Parameters	GFLOPs	Latency	FPS
YOLOv3-ultralytics[2]	44.3	63.00 M	157.3	10.5 ms	95.2
YOLOv3 baseline	38.5	63.00 M	157.3	10.5 ms	95.2
+decoupled head	39.6 (+1.1)	63.86 M	186.0	11.6 ms	86.2
+strong augmentation	42.0 (+2.4)	63.86 M	186.0	11.6 ms	86.2
+anchor-free	42.9 (+0.9)	63.72 M	185.3	11.1 ms	90.1
+multi positives	45.0 (+2.1)	63.72 M	185.3	11.1 ms	90.1
+SimOTA	**47.3** (+2.3)	63.72 M	185.3	11.1 ms	90.1
+NMS free (optional)	46.5 (-0.8)	67.27 M	205.1	13.5 ms	74.1

Table 2: Roadmap of YOLOX-Darknet53 in terms of AP (%) on COCO *val*. All the models are tested at 640×640 resolution, with FP16-precision and batch=1 on a Tesla V100. The latency and FPS in this table are measured without post-processing.

For more details…

Check the following:

YOLOX Object Detector Paper Explanation and Custom Training
https://learnopencv.com/yolox-object-detector-paper-explanation-and-custom-training/

CenterNet: Objects as Points – Anchor Free Object Detection Explained
https://learnopencv.com/centernet-anchor-free-object-detection-explained/

Paper Review: “YOLOX: Exceeding YOLO Series in 2021”
https://medium.com/mlearning-ai/paper-review-yolox-exceeding-yolo-series-in-2021-ffc1bd94a1f3

The YOLOX paper
https://arxiv.org/pdf/2107.08430.pdf

…more interesting blog posts of your choice! Plenty of stuff available online!

Briefly about a few more recent approaches

YOLO World

https://arxiv.org/pdf/2401.17270

YOLO-World

Real-Time Open-Vocabulary Object Detection

YOLO-World: Real-Time Open-Vocabulary Object Detection

Tianheng Cheng[3,2,*], Lin Song[1,*,✉], Yixiao Ge[1,2,†], Wenyu Liu[3], Xinggang Wang[3,✉], Ying Shan[1,2]

[*]**equal contribution** [†] **project lead** [✉] **corresponding author**

YOLO World

YOLO-World

Real-Time Open-Vocabulary Object Detection

Try it out: https://github.com/AILab-CVC/YOLO-World

Highlights & Introduction

This repo contains the PyTorch implementation, pre-trained weights, and pre-training/fine-tuning code for YOLO-World.

- YOLO-World is pre-trained on large-scale datasets, including detection, grounding, and image-text datasets.
- YOLO-World is the next-generation YOLO detector, with a strong open-vocabulary detection capability and grounding ability.
- YOLO-World presents a *prompt-then-detect* paradigm for efficient user-vocabulary inference, which re-parameterizes vocabulary embeddings as parameters into the model and achieve superior inference speed. You can try to export your own detection model without extra training or fine-tuning in our online demo!

YOLO World

https://arxiv.org/pdf/2401.17270

YOLO-World

Real-Time Open-Vocabulary Object Detection

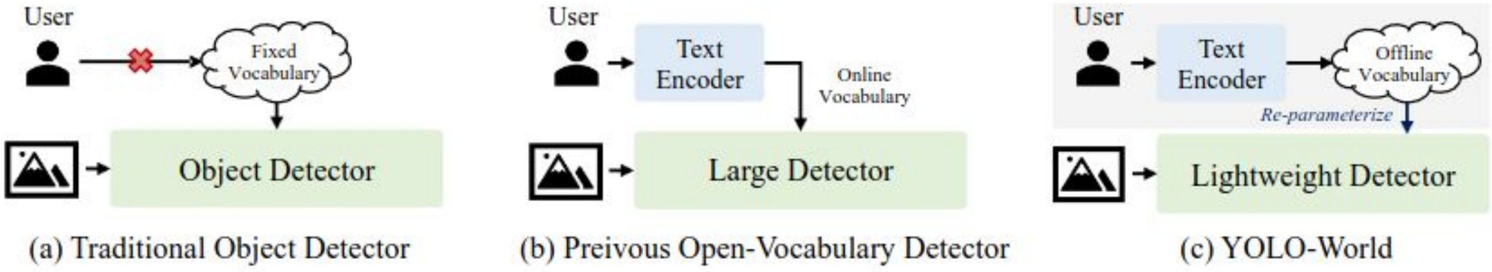
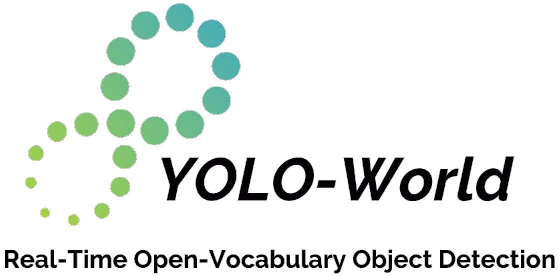


Figure 2. **Comparison with Detection Paradigms. (a) Traditional Object Detector**: These object detectors can only detect objects within the fixed vocabulary pre-defined by the training datasets, *e.g.*, 80 categories of COCO dataset [26]. The fixed vocabulary limits the extension for open scenes. **(b) Previous Open-Vocabulary Detectors:** Previous methods tend to develop large and heavy detectors for open-vocabulary detection which intuitively have strong capacity. In addition, these detectors simultaneously encode images and texts as input for prediction, which is time-consuming for practical applications. **(c) YOLO-World:** We demonstrate the strong open-vocabulary performance of lightweight detectors, *e.g.*, YOLO detectors [20, 42], which is of great significance for real-world applications. Rather than using online vocabulary, we present a *prompt-then-detect* paradigm for efficient inference, in which the user generates a series of prompts according to the need and the prompts will be encoded into an offline vocabulary. Then it can be re-parameterized as the model weights for deployment and further acceleration.

YOLO World

https://arxiv.org/pdf/2401.17270



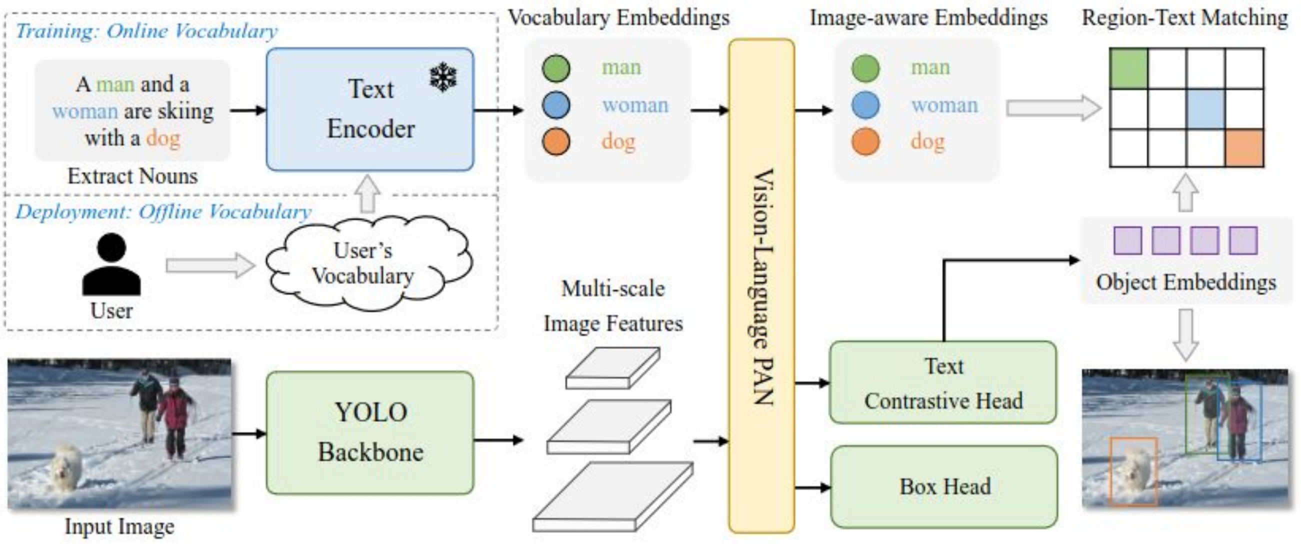


Figure 3. **Overall Architecture of YOLO-World.** Compared to traditional YOLO detectors, YOLO-World as an open-vocabulary detector adopts text as input. The *Text Encoder* first encodes the input text input text embeddings. Then the *Image Encoder* encodes the input image into multi-scale image features and the proposed *RepVL-PAN* exploits the multi-level cross-modality fusion for both image and text features. Finally, YOLO-World predicts the regressed bounding boxes and the object embeddings for matching the categories or nouns that appeared in the input text.

YOLO World

https://arxiv.org/pdf/2401.17270

YOLO-World

Real-Time Open-Vocabulary Object Detection

Figure 5. **Visualization Results on Zero-shot Inference on LVIS.** We adopt the pre-trained YOLO-World-L and infer with the LVIS vocabulary (containing 1203 categories) on the COCO `val2017`.

YOLO World

https://arxiv.org/pdf/2401.17270

YOLO-World

Real-Time Open-Vocabulary Object Detection

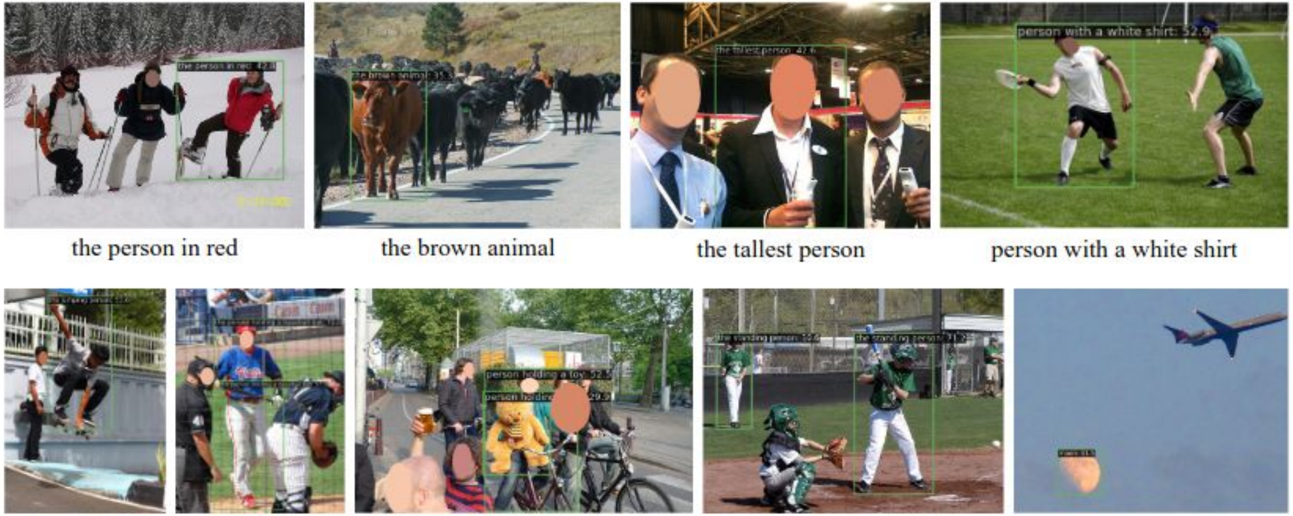
{men, women, boy, girl} {elephant, ear, leg, trunk, ivory} {golden dog, black dog, spotted dog} {grass, sky, zebra, trunk, tree}

Figure 6. **Visualization Results on User's Vocabulary.** We define the custom vocabulary for each input image and YOLO-World can detect the accurate regions according to the vocabulary. Images are obtained from COCO `val2017`.

YOLO World

https://arxiv.org/pdf/2401.17270

Real-Time Open-Vocabulary Object Detection



the person in red | the brown animal | the tallest person | person with a white shirt

the jumping person | person holding a baseball bat | person holding a toy | the standing person | moon

Figure 7. **Visualization Results on Referring Object Detection.** We explore the capability of the pre-trained YOLO-World to detect objects with descriptive noun phrases. Images are obtained from COCO `val2017`.

Grounding DINO

https://arxiv.org/pdf/2303.05499

Grounding DINO: Marrying DINO with Grounded Pre-Training for Open-Set Object Detection

Shilong Liu[1,2⋆], Zhaoyang Zeng[2], Tianhe Ren[2], Feng Li[2, 3], Hao Zhang[2, 3],
Jie Yang[2, 4], Qing Jiang[2, 6] Chunyuan Li[5], Jianwei Yang[5],
Hang Su[1], Jun Zhu[1⋆⋆], Lei Zhang[2⋆⋆].

Grounding DINO

Try it out: https://github.com/IDEA-Research/GroundingDINO

💡 Highlight

- **Open-Set Detection.** Detect **everything** with language!
- **High Performance.** COCO zero-shot **52.5 AP** (training without COCO data!). COCO fine-tune **63.0 AP**.
- **Flexible.** Collaboration with Stable Diffusion for Image Editting.

Grounding DINO

https://arxiv.org/pdf/2303.05499

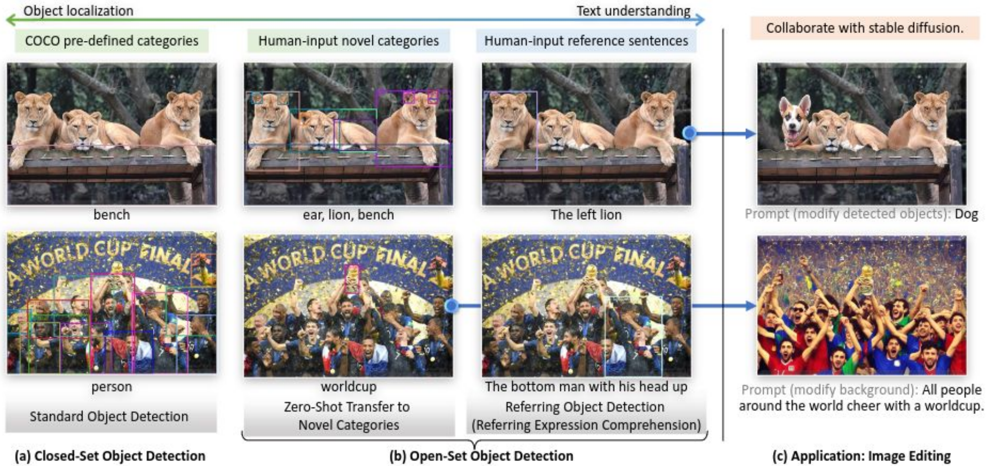


Fig. 1: (a) Closed-set object detection requires models to detect objects of pre-defined categories. (b) We evaluate models on novel objects and standard Referring expression comprehension (REC) benchmarks for model generalizations on novel objects with attributes. (c) We present an image editing application by combining Grounding DINO and Stable Diffusion [41]. Best viewed in colors.

Grounding DINO

https://arxiv.org/pdf/2303.05499

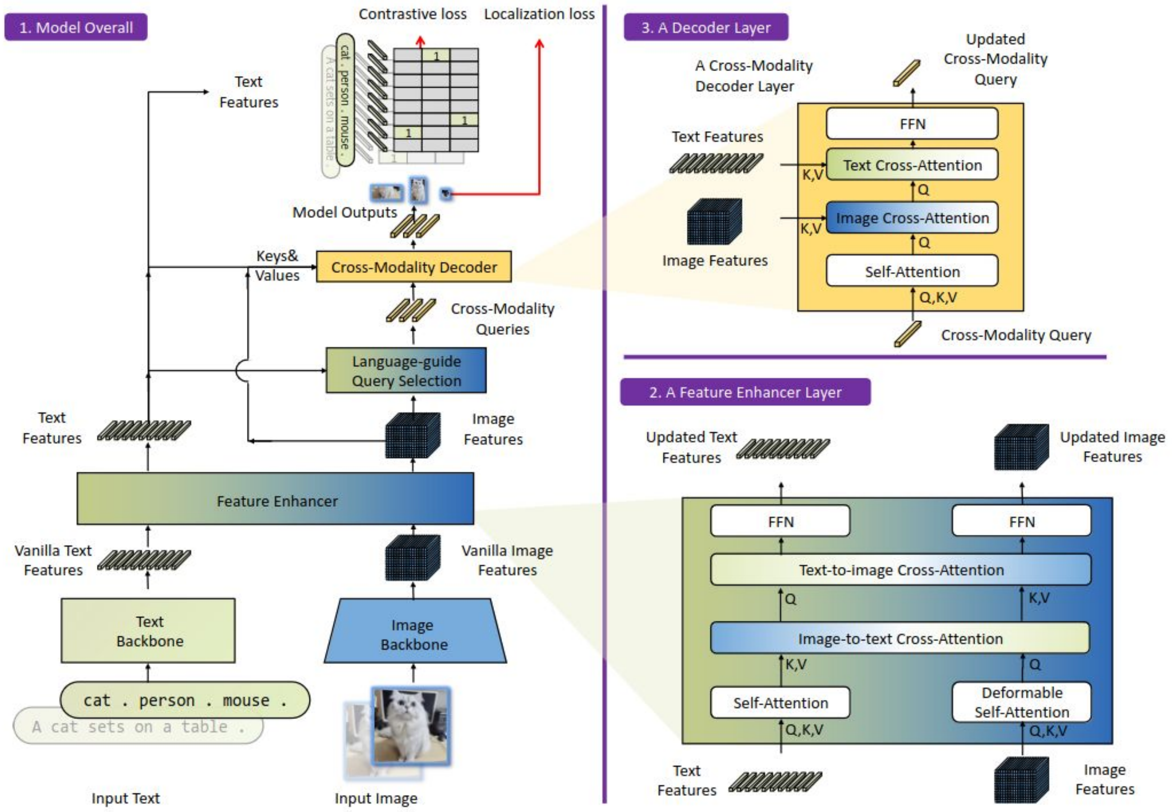


Fig. 3: The framework of Grounding DINO. We present the overall framework, a feature enhancer layer, and a decoder layer in block 1, block 2, and block 3, respectively.

Grounding DINO

https://arxiv.org/pdf/2303.05499

D.1 Detection Visualizations

We present some visualizations in Fig. 6. Our model presents great generalization on different scenes and text inputs. For example, Grounding DINO accurately locates `man in blue` and `child in red` in the last image.

Fig. 6: Visualizations of model outputs.

Grounding DINO

https://arxiv.org/pdf/2303.05499

Fig. 8: Our model predictions and ground-truths in RefCOCO.

Detic

https://arxiv.org/pdf/2201.02605

Detecting Twenty-thousand Classes using Image-level Supervision

Xingyi Zhou[1,2] ⋆ Rohit Girdhar[1] Armand Joulin[1]
Philipp Krähenbühl[2] Ishan Misra[1]

Detic

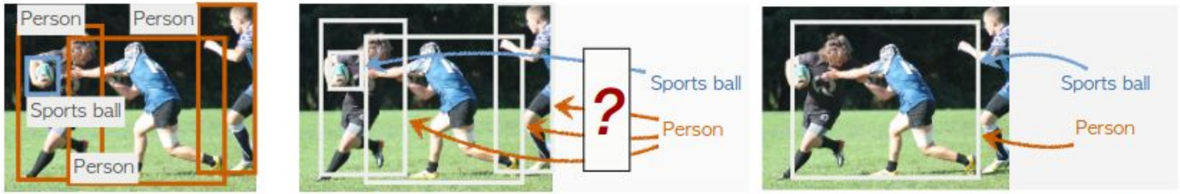
Try it out: https://github.com/facebookresearch/Detic

Features

- Detects **any** class given class names (using CLIP).
- We train the detector on ImageNet-21K dataset with 21K classes.
- Cross-dataset generalization to OpenImages and Objects365 **without finetuning.**
- State-of-the-art results on Open-vocabulary LVIS and Open-vocabulary COCO.
- Works for DETR-style detectors.

Detic

https://arxiv.org/pdf/2201.02605



(a) Standard detection **(b)** Prediction-based label assignment **(c)** Our non-prediction-based loss

Fig. 2: Left: Standard detection requires ground-truth labeled boxes and cannot leverage image-level labels. **Center:** Existing prediction-based weakly supervised detection methods [3, 44, 45] use image-level labels by assigning them to the detector's predicted boxes (proposals). Unfortunately, this assignment is error-prone, especially for large vocabulary detection. **Right:** Detic simply assigns the image-labels to the *max-size* proposal. We show that this loss is both simpler and performs better than prior work.

Detic

https://arxiv.org/pdf/2201.02605

(a) Detection data

(b) Image-labeled data

Fig. 3: Approach Overview. We mix train on detection data and image-labeled data. When using detection data, our model uses the standard detection losses to train the classifier (**W**) and the box prediction branch (**B**) of a detector. When using image-labeled data, we only train the classifier using our modified classification loss. Our loss trains the features extracted from the largest-sized proposal.

Detic

https://arxiv.org/pdf/2201.02605

Fig. 5: Qualitative results of our 21k-class detector. We show random samples from images containing novel classes in OpenImages (top) and Objects365 (bottom) validation sets. We use the CLIP embedding of the corresponding vocabularies. We show LVIS classes in purple and novel classes in green. We use a score threshold of 0.5 and show the most confident class for each box. Best viewed on screen.

And more object detectors to discover

The field is evolving rapidly

Highly satisfying scores

…yet take the results and visualizations with a grain of salt when you want to apply these detectors to your projects

You are encouraged to discover it more in detail on your own!

How? Check https://paperswithcode.com/ and recent top tier conferences (CVPR, ICCV, ECCV, NeurIPS, etc.) papers

Questions and Answers