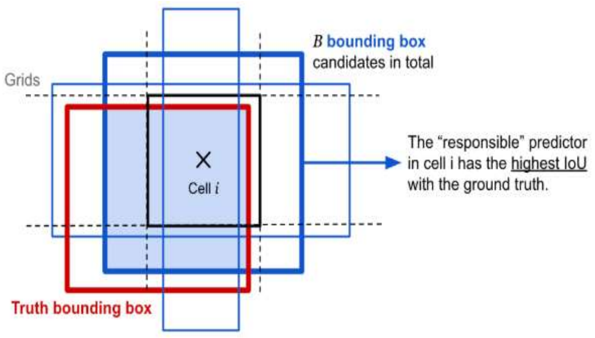
OBJECT DETECTION CHEAT SHEET

Object detection is identifying and locating objects within an image or video by assigning class labels (e.g., "cat," "car") & drawing bounding boxes around them, using machine or deep learning techniques.

Key Concepts

Bounding Boxes: Rectangular coordinates (x_min, y_min, x_max, y_max) that enclose detected objects in an image.
Annotations: Labels or metadata (e.g., name, bounding box coordinates) assigned to objects for training models.
Confidence Scores: Probability (0-1) indicating model's certainty to which specific class detected object belongs.
Intersection over Union (IoU): Metric to evaluate detection accuracy. ***Formula:*** IoU = Area of Intersection / Area of Union (IoU > 0.5 typically indicates a good detection.)

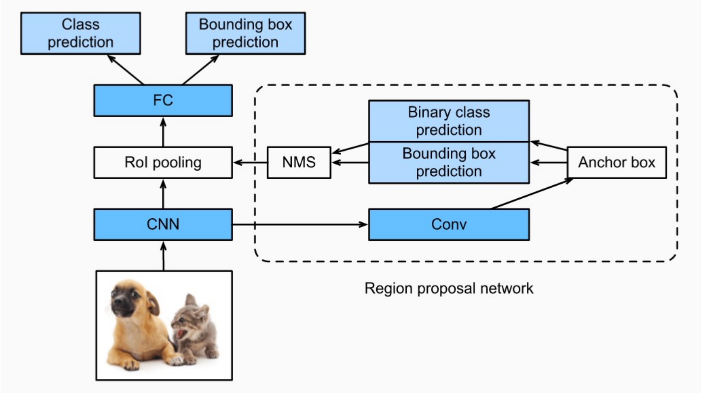


B bounding box candidates in total
Grids
Cell *i*
The "responsible" predictor in cell i has the highest IoU with the ground truth.
Truth bounding box

Algorithm	Speed	Accuracy	Key Features
R-CNN	🐢	🎯	Region-based CNN; proposes regions, extracts features, classifies objects.
Fast R-CNN	MODERATE	🎯	Shared Computation across regions.
Faster R-CNN		🎯	Region Proposal Network (RPN) for real-time performance.
SSD (Single Shot Detector)	FASTER	🎯	Single-pass detection; balances speed and accuracy.
YOLO	SUPER SPEED	🎯	Single-stage detector; predicts boxes and classes in one go. Variants: YOLOv3, YOLOv4, YOLOv8.

Workflow of Object Detection

Data Preparation: Collect and annotate images (e.g., using tools like LabelImg).
Model Selection: Choose an algorithm (e.g., YOLO for speed, Faster R-CNN for accuracy).
Training: Feed annotated data into the model; adjust hyperparameters (e.g., learning rate).
Inference: Run the trained model on new images to detect objects.
Evaluation: Measure performance using metrics like mAP (mean Average Precision) and IoU.
Post-Processing: Apply Non-Maximum Suppression (NMS) to remove duplicate detections.



Tools & Libraries

Open-source ML framework

Install: pip install tensorflow
Usage: model = tf.keras.models.load_model('path')
Docs: tensorflow.org

High-level API – neural networks

Install: pip install keras
Usage: model = load_model('m')
Docs: keras.io

Computer Vision library

Install: pip install opencv-python
Usage: img = cv2.imread('image.jpg')
Docs: opencv.org

Pytorch

Install: pip install torch
Usage: torch.nn.Module
Docs: pytorch.org

Challenges & Troubleshooting

Occlusion: Objects partially hidden. ***Tip:*** Use models with context awareness (e.g., Faster R-CNN).
Small Objects: Hard to detect. ***Tip:*** Increase image resolution or use multi-scale detection (e.g., SSD).
Class Imbalance: Rare objects underrepresented. ***Tip:*** Augment data or use weighted loss functions.
False Positives: Incorrect detections. ***Tip:*** Adjust confidence threshold or improve training data quality.

Additional Resources

- Books: 'Deep Learning for Computer Vision' by Adrian Rosebrock, 'Computer Vision: Algorithms and Applications' by Szeliski, "Deep Learning" by Goodfellow, Bengio, and Courville
- Tutorials: Coursera, Udacity, and Fast.ai courses
- Websites: tensorflow.org/tutorials, https://www.pyimagesearch.com/, pjreddie.com/darknet/yolo towardsdatascience.com