Object Detection with CNNs and Transformers

Applied Deep Learning — Chapter 20

Yin Yang

Foundation Books Project

Roadmap

From Classification to Detection

Boxes, Scores, and Metrics

The Applied Detection Pipeline

Two-Stage: Fast / Faster R-CNN

One-Stage: YOLO

DETR: Detection as Set Prediction

DINO and Grounding DINO

Choosing a Detector

Summary

From Classification to Detection

Why Detection?

Classification returns one label per image. Many applied questions also need *where* the evidence is.

- Chest X-ray: localize each lung opacity, not only "abnormal."
- Warehouse photo: positions of all boxes, pallets, workers.
- Traffic image: number and locations of cars, not only "car present."

Detection adds geometry. Output is a *set* of instances, each with a class label, a bounding box, and a confidence score.

Object Detection: Definition

Object detection
Supervised task of predicting a finite set of object instances. Each instance has a class label, bounding box, and confidence score.

Closed-set Fixed list of class names; the model is judged only on those.

Open-vocabulary Class names or text phrases beyond the supervised list (e.g. Grounding DINO).

Running example: chest X-ray detection (e.g. RSNA, VinDr-CXR). Educational, not a clinical system.

Boxes, Scores, and Metrics

Bounding Box Formats

Image origin at the upper-left; x right, y down. Width W, height H.

Corner form $b = (x_{\min}, y_{\min}, x_{\max}, y_{\max})$.

Center form (x_c, y_c, w, h); YOLO datasets normalize by W, H.

Example: $(256, 320, 640, 704)$ on a 1024×1024 image becomes

$$(0.4375,\ 0.5,\ 0.375,\ 0.375).$$

Box format is data. Record corner vs. center, pixel vs. normalized, and the conversion image size.

Intersection over Union

$$\mathrm{IoU}(a, b) = \frac{\mathrm{area}(a \cap b)}{\mathrm{area}(a \cup b)}.$$

Worked example: $g = (100, 80, 220, 200)$, $p = (130, 100, 260, 230)$.

- $\mathrm{area}(g) = 120 \cdot 120 = 14{,}400$; $\mathrm{area}(p) = 130 \cdot 130 = 16{,}900$.
- Intersection: width 90, height 100, area 9,000.
- Union: $14{,}400 + 16{,}900 - 9{,}000 = 22{,}300$.
- IoU ≈ 0.404.

IoU threshold matters. Acceptable at 0.5; fails strict localization at 0.7.

Precision, Recall

At a chosen score threshold and matching rule:

$$\text{precision} = \frac{\text{TP}}{\text{TP} + \text{FP}}, \qquad \text{recall} = \frac{\text{TP}}{\text{TP} + \text{FN}}.$$

Example: 3 ground-truth opacities, 5 ranked predictions, after IoU matching the sequence is TP, FP, TP, FP, TP.

- After 1: precision $= 1$, recall $= 1/3$.
- After 3: precision $= 2/3$, recall $= 2/3$.
- After 5: precision $= 3/5$, recall $= 1$.

Stricter score threshold raises precision, may lower recall.

AP and mAP

Average Precision (AP)
Summarizes the precision-recall curve for one class at a chosen IoU rule.

Toy AP from the previous example (precision at recall increases):

$$\frac{1 + 2/3 + 3/5}{3} \approx 0.756.$$

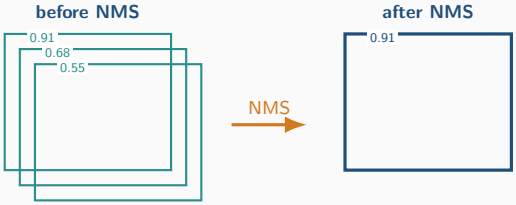
Mean Average Precision (mAP)
Average AP over classes, sometimes also over IoU thresholds.

COCO-style:

- mAP_{50}: IoU threshold 0.5.
- $\text{mAP}_{50:95}$: average over thresholds $0.50, \ldots, 0.95$ (step 0.05).

Loose-but-correct boxes help mAP_{50}; strict thresholds reward tight localization.

Non-Maximum Suppression (NMS)



NMS
Keep the highest-scoring box in a cluster; discard others with IoU above a threshold.

Example: two predictions, IoU 0.8, scores 0.91 and 0.68, threshold 0.5 $\Rightarrow$ keep 0.91.

- Threshold can suppress correct boxes for adjacent findings.
- DETR-style detectors avoid NMS via set prediction.

The Applied Detection Pipeline

Pipeline Beats Architecture

- Output contract: image source, class list, box convention, split policy.
- Visual label inspection *before* training: catches swapped coords, wrong resize, missing negatives.
- Pretraining: a pretrained detector usually beats random init even on grayscale medical images.
- Image size: doubling side length from 640 to 1024 costs $1024^2/640^2 = 2.56\times$ more pixels.
- Augmentation respects domain (no horizontal flip if left/right anatomy matters).

Resolution is a budget. Larger inputs help small objects but spend memory, time, latency.

Split by Source

Standard random splits leak when images are correlated.

- Same patient $\Rightarrow$ patient-level split.
- Same video / study / slide $\Rightarrow$ group-level split.
- Validation: used repeatedly to choose hyperparameters.
- Test set: used *once* after final selection.

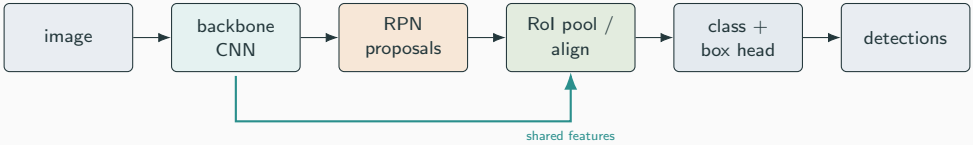
Symptom: validation improves but test drops.
Likely cause: model selection leaked into validation, or split policy was too loose.

Common Detection Symptoms

Symptom	Likely cause	First response
Low recall	Score threshold too high; small objects; sparse labels	Lower threshold for analysis; inspect missed boxes; larger image
Many FPs	Imbalanced classes; noisy labels	Inspect high-score FPs; tune threshold on val; add hard negatives
Duplicate boxes	NMS too loose	Check postprocessing; use framework evaluator
Val good, test bad	Leakage; small val set; distribution shift	Recheck split; per-source metrics; lock test set

Two-Stage: Fast / Faster R-CNN

R-CNN Family Idea



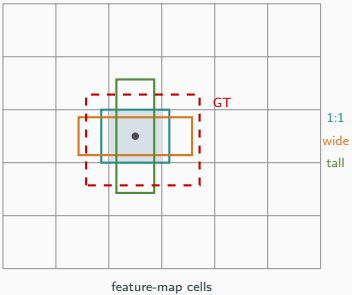
Two-stage pattern: (1) propose candidate regions; (2) classify and refine.

R-CNN CNN on each proposal $\Rightarrow$ slow.

Fast R-CNN CNN once; pool features per RoI.

Faster R-CNN RPN learns proposals from shared features.

Anchors and Box Offsets



Anchor: a reference at a feature-map location with width w_a, height h_a, center (x_a, y_a).
Predicted offsets:

$$t_x = \frac{x - x_a}{w_a}, \quad t_y = \frac{y - y_a}{h_a}, \quad t_w = \log \frac{w}{w_a}, \quad t_h = \log \frac{h}{h_a}.$$

RPN loss:

$$\mathcal{L}_{\text{RPN}} = \mathcal{L}_{\text{obj}} + \lambda \mathbb{1}_{\text{pos}} \mathcal{L}_{\text{box}}.$$

Negatives train objectness; positives also train offsets.

Two-Stage vs One-Stage

	Main idea	**Tradeoff**
Two-stage	Propose, then classify and refine each region	Often strong, interpretable; more complex, end-to-end latency higher
One-stage	Predict boxes and classes in one pass	Simpler to train and deploy; thresholds and postprocessing still matter

Measure the pipeline. Latency claims should include preprocessing, decoding, and postprocessing — not only the forward pass.

One-Stage: YOLO

YOLO: One-Stage Detection

One-stage detector
Predicts boxes and classes directly from image features without a separate proposal stage.

- Image divided into spatial locations; each contributes candidate boxes.
- Older YOLO: anchor-based. Newer: anchor-free / NMS-free variants.
- Modern workflow: pretrained checkpoint + dataset YAML + fine-tune.

Simplified YOLO Loss

For predicted box $\hat{b}_i$, objectness $\hat{o}_i$, and class probabilities $\hat{p}_i$:

$$\mathcal{L}_{\mathrm{YOLO}} = \sum_i \mathbb{1}_i^{\mathrm{obj}} \Big(\lambda_{\mathrm{box}} \mathcal{L}_{\mathrm{box}}(\hat{b}_i, b_i) + \mathcal{L}_{\mathrm{cls}}(\hat{p}_i, y_i) \Big) + \lambda_{\mathrm{obj}} \mathcal{L}_{\mathrm{obj}}(\hat{o}_i, o_i).$$

Modern variants change assignment rules and exact losses, but every run still balances localization, objectness, and class prediction. Names drift. Record installed package, model variant, and configuration: nano, small, medium, large, XL.

YOLO Decisions to Justify

Decision	Starting point	Evidence
Model size	Nano or small	val mAP, time, memory, latency
Image size	640 unless very small objects	Small-object recall, strict mAP
Epochs	Enough for val to stabilize	Curves, best epoch, overfit gap
Augmentation	Defaults; adjust with reason	Per-class AP, visual errors
Thresholds	Tune on val	Precision/recall, FPs, missed

YOLO's speed is a debugging asset: it shortens the path from data to evidence, so labels and splits get tested quickly.

DETR: Detection as Set Prediction

DETR: From Boxes to Sets

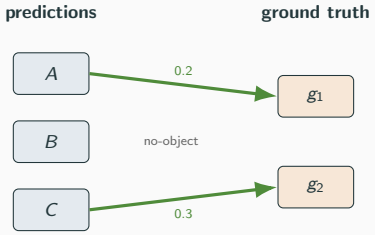
Set prediction
Output is an unordered finite set; ordering carries no semantic meaning.

DETR directly predicts a fixed-size set of object candidates and trains each ground-truth object to match *one* prediction.

Object query Learned vector used by the Transformer decoder as a slot for one possible object.

No-object class Unmatched queries are trained to predict "empty."

Bipartite Matching



Cost matrix:

	g_1	g_2
A	**0.2**	1.4
B	0.6	0.5
C	1.0	**0.3**

Best assignment $A \to g_1$, $C \to g_2$; total 0.5. B is unmatched, trains the no-object class.

One-to-one matching discourages duplicates at training time, reducing the need for NMS.

DETR Loss

Optimal assignment:

$$\hat{\sigma} = \arg\min_{\sigma} \sum_{j=1}^{M} \mathcal{C}(g_j, \hat{y}_{\sigma(j)}).$$

Training loss after matching:

$$\mathcal{L}_{\mathrm{DETR}} = \sum_{j=1}^{M} \Big[\mathcal{L}_{\mathrm{cls}} + \lambda_1 \| b_j - \hat{b}_{\hat{\sigma}(j)} \|_1 + \lambda_{\mathrm{giou}} \mathcal{L}_{\mathrm{GIoU}} \Big] + \mathcal{L}_{\text{no-object}}.$$

Strengths: clean formulation, less dependence on NMS/anchors.

Drawbacks of original DETR: slow convergence, weaker on small objects.

DINO and Grounding DINO

DINO: Training DETR Better

Note: *this* DINO is the detector (DETR with Improved deNoising anchOr boxes), not the self-supervised representation.

Denoising training Feed deliberately corrupted labels/boxes; train the model to recover the correct answer.

Box-aware queries Initialize queries from box-like priors; supervise decoder refinement directly.

Effect: stronger accuracy and faster convergence than original DETR-style baselines.

Denoising Example

Ground-truth opacity box: $(256, 320, 640, 704)$. Noisy version added during training:

$$(246, 335, 660, 690).$$

Training task: starting from this nearly-correct query, recover a prediction close to the true box and class. Why it helps: gives queries direct practice on the refinement step that DETR otherwise has to learn slowly from scratch.

Grounding DINO: Text-Guided Detection

Closed-set detectors require a fixed class list. Grounding DINO conditions the detector on *text*.

Open-vocabulary detection
Detection conditioned on class names or phrases beyond a small fixed label list.

- Prompts: “red cup,” “left lung opacity,” “surgical tube.”
- Useful for exploration, weak labeling, annotation assistance.
- Localized prediction contract: text must connect to image regions.

Pseudo-labels and Their Risks

Pseudo-label
A label generated by a model rather than a human annotator. Useful to expand data, but can copy the model's mistakes and biases.

Failure modes:

- Miss domain-specific findings.
- Latch onto the wrong phrase.
- Plausible box on tissue that is not the intended target.

Prompt is not proof. A text-conditioned box still needs validation against trusted annotations.

Choosing a Detector

Family Comparison

Family	Core idea	Strengths	Limits
Faster R-CNN	Propose, classify, refine	Strong baseline; mature	Complex; higher latency
YOLO	Boxes & classes in one pass	Fast; simple fine-tune; first baseline	Thresholds matter; small targets need resolution
DETR	Set prediction with queries	Less NMS reliance	Slow convergence; weak small objects
DINO det.	Denoising + better queries	Stronger accuracy and convergence	More complex than YOLO
Grounding DINO	Text-conditioned detection	Open vocabulary; annotation aid	Prompt sensitivity; needs validation

An RSNA Reference Run

Three short runs on a balanced RSNA pneumonia subset (512 train / 128 val / 128 test, single class `opacity`, 20 epochs).

Run	Config	Accuracy	Cost
Baseline	YOLO26n, 640	$\text{mAP}_{50} = 0.244$; $\text{mAP}_{50:95} = 0.091$	159 s; 58.6 ms/img
Larger model	YOLO26s, 640	$\text{mAP}_{50} = 0.318$; $\text{mAP}_{50:95} = 0.102$	246 s; 60.2 ms/img
Higher res	YOLO26n, 1024	$\text{mAP}_{50} = 0.182$; recall 0.515	326 s; 79.3 ms/img

Validation chose larger 640-pixel model. Test once: $\text{mAP}_{50} = 0.324$. Higher-res did *not* automatically win.

The Final Rule

Final means final. Use validation to choose the recipe; use the test set *once* to report the selected detector. Report should include:

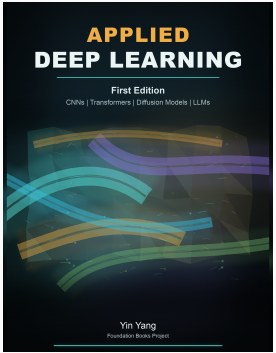
- Configuration (model, image size, epochs, augmentation, thresholds).
- Accuracy: mAP_{50}, $\text{mAP}_{50:95}$, per-class AP, precision, recall.
- Cost: training time, latency, hardware.
- Qualitative: a tight true positive, a missed small finding, a false positive, a duplicate prediction.

Summary

Summary

- **Detection** predicts a *set* of (class, box, score) instances; classification is not enough when location matters.
- **Metrics:** IoU for overlap; precision/recall/AP/mAP under a matching rule and IoU threshold; report thresholds and postprocessing.
- **Pipeline**: labels, splits, pretraining, image size, augmentation, error analysis decide most projects.
- **Two-stage** (Faster R-CNN): propose, classify, refine. **One-stage** (YOLO): predict in one pass.
- **DETR/DINO**: set prediction with bipartite matching. **Grounding DINO**: text-conditioned, open vocabulary.
- **Discipline**: split by source; choose by validation; report the test set once with qualitative evidence.

Next: segmentation — replace boxes with pixel-level masks.



Applied Deep Learning

Chapter overview slides for the book by Yin Yang.

Book page	`foundation-books.github.io/applied-deep-learning`
Code	`github.com/foundation-books/applied-deep-learning-code`
Print	`amazon.com/dp/B0GZF7QRSH`
Kindle	`amazon.com/dp/B0GZF9221X`
License	CC BY-NC-ND 4.0

These free overview slides may be shared non-commercially with attribution and without modifications. Full explanations, exercises, and companion-code context are in the book and linked repository.