Object Detection - Data layer

Datasets — Benchmarks & Sources

COCO 2017 (Object Detection)

- **What it is:** ~118k train / 5k val / 20k test images with ~80 object classes, crowd labels, instance masks, keypoints.
- **Why it matters:** The de-facto general-purpose benchmark; nearly all modern detectors report mAP here.
- **Quirks:** Many small objects; "iscrowd" regions; multiple annotations per image; long-tail frequency even within 80 classes.
- **Where:** Hugging Face (`coco`, config `2017` → subsets `train`, `validation`); also via torchvision's `CocoDetection`.

Pascal VOC 2007/2012

- **What it is:** ~20 object classes; simpler images; VOC07 (~5k trainval / 5k test), VOC12 (~11k trainval).
- **Why it matters:** Lightweight baseline; great for quick iterations and educational demos.
- **Quirks:** Older annotation style; fewer small objects; AP computed at IoU=0.5 in classic protocol.
- **Where:** Hugging Face (`pascal_voc`).

LVIS v1 (Long-tail Visual Instance Segmentation)

- **What it is:** ~1k+ categories with extreme class imbalance; instance segmentation + boxes.
- **Why it matters:** Tests open-vocabulary/long-tail capability and rare class generalization.
- **Quirks:** Highly skewed label distribution; requires special sampling/reweighting.
- **Where:** Hugging Face (`lvis`).

Open Images V6 (Detection)

- **What it is:** Millions of images with ~600 categories; image-level + box labels; hierarchical ontology.
- **Why it matters:** Scale and label hierarchy; good for pretraining and robustness.
- **Quirks:** Noisy labels; partial annotations; class hierarchy requires careful mapping.
- **Where:** Hugging Face (`open_images_v6` with `object_detection` subset).

Objects365

- **What it is:** ~365 categories; ~600k images richly annotated with boxes.
- **Why it matters:** Large-scale pretraining to boost downstream COCO/LVIS performance.
- **Quirks:** Licensing/hosting can be heavier; class names differ from COCO.

- **Where:** Hugging Face (`objects365`) or official site (account needed).

Cityscapes (Boxes from Polygons)

- **What it is:** Urban street scenes; fine pixel labels for 8 categories (19 for segmentation). Boxes can be derived or use detection splits from community repos.
- **Why it matters:** Driving domain; medium-scale; consistent viewpoint.
- **Quirks:** Predominantly large objects; strong class bias (cars/persons).
- **Where:** Hugging Face (`cityscapes`) for segmentation polygons; convert to boxes or use detection versions from forks.

BDD100K (Detection)

- **What it is:** 100k driving images with detection, tracking, and lane/seg labels.
- **Why it matters:** Broad driving conditions (night, weather); good for domain robustness.
- **Quirks:** Class set differs from COCO; time-of-day imbalance.
- **Where:** Hugging Face (`bdd100k`).

KITTI (Detection)

- **What it is:** On-road images labeled for Car/Pedestrian/Cyclist in camera view.
- **Why it matters:** Classic autonomous driving detection benchmark.
- **Quirks:** Small dataset; strict evaluation protocol; depth cues from stereo.
- **Where:** Hugging Face (`kitti`).

CrowdHuman

- **What it is:** ~15k train images focusing on crowded human boxes (head/full body/visible body).
- **Why it matters:** Tests NMS/duplicate handling in crowded scenes.
- **Quirks:** Heavy occlusion; dense overlaps stress post-processing.
- **Where:** Hugging Face (`crowdhuman`).

Aerial/Remote Sensing (DOTA / xView)

- **What it is:** DOTA: oriented boxes over aerial imagery; xView: large-scale overhead boxes.
- **Why it matters:** Small, rotated objects; domain shift from natural images.
- **Quirks:** Rotated boxes (DOTA); extreme small object ratios; tiling needed.
- **Where:** Hugging Face (`dota`, `xview`).

Video Detection/Tracking (YouTube-BB, TAO, BDD-Tracking)

- **What it is:** Frame-wise boxes over videos (YouTube-BB); long-tail, open-world (TAO); driving videos (BDD-Tracking).
- **Why it matters:** Temporal consistency and motion robustness.
- **Quirks:** Label sparsity per frame (YT-BB); identity switches (tracking); domain drift.
- **Where:** Hugging Face (`youtube_bounding_boxes`, `tao`, `bdd100k` with tracking splits).

Preprocessing (what to do and why)

Normalization

We adjust pixel values so they're centered and scaled, making training stable.

- **ImageNet stats:** `Normalize(mean=[0.485, 0.456, 0.406], std=[0.229, 0.224, 0.225])` → Matches the preprocessing expected by most pretrained backbones (ResNet/ViT/ConvNeXt).
- **From scratch:** `Standardize per dataset` → If no pretrained weights, compute dataset mean/std to reduce covariate shift.

Resizing & Aspect Strategy

We resize while keeping aspect ratio to fit model input limits without distorting boxes.

- **COCO/DETR style:** `Resize shortest_side∈{480…800}; max_size=1333` → Multi-scale training improves scale invariance; cap long side to control memory.
- **YOLO style:** `Letterbox to 640×640 (or 1024×1024)` → Pads to square without stretch; consistent batch shapes increase throughput.
- **Driving/Aerial:** `Tile or long-side=2048 + sliding windows` → Preserves tiny objects; tiling prevents shrinking small targets into oblivion.

Geometric Augmentations

We perturb geometry to improve invariance while updating boxes/masks accordingly.

- **RandomHorizontalFlip(p=0.5)** → Cheap diversity; must flip x-coords of boxes.
- **RandomAffine / RandomPerspective (small ranges)** → Adds viewpoint variance; keep boxes clipped and valid.
- **Mosaic/MixUp (YOLO-family)** → Combines images to enrich context and small-object exposure; tune to avoid label noise.

Photometric Augmentations

We vary color/lighting so the model learns robust features instead of memorizing illumination.

- **ColorJitter / HSV jitter / RandomBrightnessContrast** → Simulates different sensors/time-of-day; keep within moderate bounds.
- **Gaussian noise/blur (light)** → Models sensor noise or motion blur without hiding tiny objects.

Label Encoding & Filtering

We convert annotations to the target detector's format and drop unusable boxes.

- **Anchor-based (RetinaNet/YOLOv3):** encode to per-feature-map anchors → Precompute anchor boxes; match via IoU thresholds.
- **Anchor-free (FCOS/YOLOX/DETR):** center/point or set-based encoding → Simpler assignment; ensure correct class indices and [x_min, y_min, x_max, y_max] order.
- **Filter tiny/invalid boxes (e.g., area < 4 px²)** → Reduces label noise; prevents unstable gradients.

Evaluation Transforms

We disable stochastic augments and use a single, reproducible resize for fair metrics.

- **COCO/common:** `Resize shortest_side=800, max_size=1333` + `Normalize(...)` → Mirrors widely reported settings; no flips/crops at eval.
- **YOLO inference:** `Letterbox to train size` + `Normalize(...)` → Keeps NMS behavior consistent with training shape.

Dataloading tips

We prepare the dataset so training is fast, reproducible, and efficient.

- **Aspect-ratio batching:** group images by similar h/w before batching → Cuts padding waste and speeds up training.
- **Custom `collate_fn` (variable targets):** → Batches images while keeping a list of per-image dictionaries (`boxes`, `labels`, `areas`, `iscrowd`).
- **Prefetch & pin memory:** `DataLoader(pin_memory=True, prefetch_factor>1, persistent_workers=True)` → Ensures GPUs aren't starved while waiting for data.
- **Worker init functions:** `worker_init_fn=seed_all` → Makes sure random augmentations are reproducible across runs.
- **Cache decoded images / memmaps:** → Avoids repeated JPEG decode cost on large corpora.
- **COCO quirks:** respect `iscrowd` masks during training/eval → Use them to ignore regions in loss/evaluation; aligns with COCO mAP.
