Session 4: Hands-on Object Detection Lab

Transfer Learning for Building Detection (Metro Manila)

Stylianos Kotsopoulos
EU-Philippines CoPhil Programme

Lab Overview

Duration: 2.5 hours
Type: Hands-on Coding Lab (Colab)
Goal: Fine-tune a pre-trained detector for buildings/settlements

You will do: - Load pre-trained detector (TF Hub / PyTorch Hub) - Prepare Sentinel-2 urban patches + COCO annotations - Fine-tune detector (10–30 epochs) - Evaluate with mAP, Precision, Recall - Visualize detections and export results

Prerequisites: - Day 3 Session 3 (object detection theory) - Colab GPU enabled

Resources: - Notebook: `Day3_Session4_Object_Detection_STUDENT.ipynb` - Reference PDF: “Session 4_ Hands-on – Feature/Object Detection from Sentinel Imagery.pdf”

Case Study: Metro Manila

Urban Monitoring Focus

- AOI: Quezon City and Pasig River corridor
- Data: Sentinel-2 RGB + NIR (10 m)
- Task: Building / settlement detection (bounding boxes)

Why object detection?

- Count and localize discrete objects (buildings)
- Track growth/change over time
- Prioritize vulnerable areas (DRR, planning)

Workflow

Syntax error in text
mermaid version 11.6.0

Model Options

- SSD MobileNet V2 (fast, lightweight)
- Faster R-CNN ResNet50 (accurate, slower)
- YOLOv5/v8 (balanced)

Data Preparation (30 min)

Inputs

- Images: 320×320 / 512×512 S2 patches
- Annotations: COCO JSON (`bbox`, `category_id`)

Steps

1. Load images and annotations
2. Visual inspection of boxes
3. Train/val/test split (70/15/15)
4. Convert to model’s expected format

Demo dataset included

- ~100 urban patches with pre-annotated buildings
- Ready to fine-tune without long setup

Fine-Tuning (40 min)

Strategy

- Freeze early backbone layers
- Train detection head with low LR (1e-4 to 1e-3)
- 10–30 epochs with early stopping

Monitor

- Loss curves (train/val)
- mAP rising then plateauing

```
# Pseudocode
optimizer = Adam(learning_rate=1e-4)
for epoch in range(20):
    for batch in train_loader:
        preds = model(batch.images)
        loss = detection_loss(preds, batch.targets)
        # backprop + step
    val_map = evaluate_map(model, val_loader)
```

Evaluation (25 min)

Metrics

- mAP@0.5 (VOC), mAP@[0.5:0.95] (COCO)
- Precision, Recall

IoU Thresholds

- Correct detection if IoU ≥ 0.5
- NMS threshold: 0.4–0.5 typical

```
# Evaluate
results = evaluate_model(model, test_loader)
print(results["mAP_50"], results["precision"], results["recall"])
```

Visualization (20 min)

Show detections

- Draw boxes + confidence scores
- Compare pre- vs post-fine-tuning
- Inspect false positives/negatives

```
for det in detections:
    if det["score"] > 0.5:
        draw_box(image, det["bbox"], label=f"{det['score']:.2f}")
```

Export & Integration (10 min)

Export

- Save model checkpoint
- Export detections (GeoJSON with centroids / footprints)

GIS Workflow

- Merge tile detections
- Export to QGIS/ArcGIS for mapping and stats

Troubleshooting (10 min)

- OOM → smaller batch, smaller inputs, SSD MobileNet
- Low mAP → more epochs, LR tuning, annotation checks
- Too many FPs → higher confidence threshold (0.6–0.7)
- Slow training → ensure GPU, try lighter model

Time Plan

Block	Minutes
Intro & Setup	15
Data Prep	30
Fine-Tuning	40
Evaluation	25
Visualization	20
Export	10
Troubleshooting	10
Buffer	10

Start the Lab

Open the notebook in Colab:

`day3/notebooks/Day3_Session4_Object_Detection_STUDENT.ipynb`