AIDA 2154 – Final Project: Deep Learning-based Monitoring of Solar Panels

Group 1: Mark, Herve, Kelsey
Course: Computer Vision (Fall 2025)
Instructor: Dr. Muhammad Tufail

Project Overview

This notebook implements an **Object Detection** system using **YOLOv11** to identify thermal anomalies in photovoltaic modules. The system is trained on the *Infrared Solar Modules* dataset to detect and localize 12 distinct classes of faults (e.g., Hotspots, Diode failures) and environmental issues (Vegetation, Soiling).

Methodology

1. **Data Preparation:** Dynamic configuration of the dataset paths.
2. **Model Training:** Fine-tuning a pre-trained `yolo11n` (Nano) model.
3. **Evaluation:** Analyzing Mean Average Precision (mAP) and confusion matrices.
4. **Inference:** Demonstrating the model on unseen test images.

[1]:
```
# Cell 1: Environment Setup & GPU Check
# Installs dependencies if missing, and imports necessary libraries.

import os
import sys
import yaml
import random
import glob
import cv2
import torch
import matplotlib.pyplot as plt

try:
    import ultralytics
    from ultralytics import YOLO
except ImportError:
    print("Installing Ultralytics YOLO...")
    !pip install -q ultralytics
    import ultralytics
    from ultralytics import YOLO

# Verify installation
ultralytics.checks()

# HARDWARE CHECK
if torch.cuda.is_available():
    print(f"✅ GPU Detected: {torch.cuda.get_device_name(0)}")
    BATCH_SIZE = 16  # Standard Production Batch Size
else:
    print("⚠️ GPU NOT Detected. Running on CPU.")
    BATCH_SIZE = 4

print(f"Project Root: {os.getcwd()}")
```

```
Ultralytics 8.3.235  Python-3.11.14 torch-2.10.0.dev20251205+cu128 CUDA:0 (NVIDIA GeForce RTX 5070 Laptop GPU, 8151MiB)
Setup complete  (20 CPUs, 31.1 GB RAM, 362.6/927.8 GB disk)
✅ GPU Detected: NVIDIA GeForce RTX 5070 Laptop GPU
Project Root: C:\Users\markm\Desktop\AIDA2154_Solar_Panel_Project
```

[2]:
```
# Cell 2: Dynamic Data Configuration
# This cell automatically locates the dataset and creates the YOLO config file.
# This ensures the notebook runs on any machine without manual path editing.

# 1. Locate the Dataset
possible_locations = [
    os.path.join("InfraredSolarModules", "ImageSet"),  # Expected location
    "datasets/InfraredSolarModules/ImageSet",          # Alternative
    "ImageSet"                                         # Flat structure
]

dataset_root = None
for path in possible_locations:
    if os.path.exists(os.path.abspath(path)):
        dataset_root = os.path.abspath(path)
```

```
        break

if dataset_root is None:
    raise FileNotFoundError("❌ CRITICAL: Dataset not found. Please ensure 'InfraredSolarModules/ImageSet' exists.")

print(f"✅ Dataset found at: {dataset_root}")

# 2. Create the Configuration Dictionary
config = {
    'path': dataset_root,
    'train': 'train/images',
    'val': 'valid/images',
    'test': 'test/images',
    'names': {
        0: 'Cell',
        1: 'Cell-Multi',
        2: 'Cracking',
        3: 'Diode',
        4: 'Diode-Multi',
        5: 'Hot-Spot',
        6: 'Hot-Spot-Multi',
        7: 'No-Anomaly',
        8: 'Offline-Module',
        9: 'Shadowing',
        10: 'Soiling',
        11: 'Vegetation'
    }
}

# 3. Save as YAML
config_filename = 'final_solar_config.yaml'
with open(config_filename, 'w') as f:
    yaml.dump(config, f)

print(f"✅ Config file '{config_filename}' generated.")
```

```
✅ Dataset found at: C:\Users\markm\Desktop\AIDA2154_Solar_Panel_Project\InfraredSolarModules\ImageSet
✅ Config file 'final_solar_config.yaml' generated.
```

[3]:

```
# Cell 3: Model Training (Task 3)
# We use YOLOv11 Nano (Detection) for speed and efficiency.

# Load Model
model = YOLO('yolo11n.pt')

print(f"🚀 Starting Training Pipeline with Batch Size: {BATCH_SIZE}...")

# Train
results = model.train(
    data='final_solar_config.yaml',
    epochs=15,
    imgsz=640,          # Standard detection size
    batch=BATCH_SIZE,   # Set to 16 (Standard)
    project='Solar_Final_Runs',
    name='yolo11_detect',
    exist_ok=True,
    plots=True,

    # --- COMPATIBILITY FLAGS ---
    # These settings prevent kernel crashes on Windows/Nightly builds
    workers=0,          # Forces main-process loading (Prevents DataLoader crashes)
    amp=False,          # Disables Mixed Precision (Prevents SegFaults on new GPUs)
    cache=False,        # Reads from disk to avoid memory spikes
)
```

```
🚀 Starting Training Pipeline with Batch Size: 16...
Ultralytics 8.3.235  Python-3.11.14 torch-2.10.0.dev20251205+cu128 CUDA:0 (NVIDIA GeForce RTX 5070 Laptop GPU, 8151MiB)
engine\trainer: agnostic_nms=False, amp=False, augment=False, auto_augment=randaugment, batch=16, bgr=0.0, box=7.5, cache=False, cfg=None, classes=None, close_mosaic=10, cls=0.5, compile=False, conf=None, copy_paste=0.0, copy_paste_mode=flip, cos_lr=False, cutmix=0.0, data=final_solar_config.yaml, degrees=0.0, deterministic=True, device=None, dfl=1.5, dnn=False, dropout=0.0, dynamic=False, embed=None, epochs=15, erasing=0.4, exist_ok=True, fliplr=0.5, flipud=0.0, format=torchscript, fraction=1.0, freeze=None, half=False, hsv_h=0.015, hsv_s=0.7, hsv_v=0.4, imgsz=640, int8=False, iou=0.7, keras=False, kobj=1.0, line_width=None, lr0=0.01, lrf=0.01, mask_ratio=4, max_det=300, mixup=0.0, mode=train, model=yolo11n.pt, momentum=0.937, mosaic=1.0, multi_scale=False, name=yolo11_detect, nbs=64, nms=False, opset=None, optimize=False, optimizer=auto, overlap_mask=True, patience=100, perspective=0.0, plots=True, pose=12.0, pretrained=True, profile=False, project=Solar_Final_Runs, rect=False, resume=False, retina_masks=False, save=True, save_conf=False, save_crop=False, save_dir=C:\Users\markm\Desktop\AIDA2154_Solar_Panel_Project\Solar_Final_Runs\yolo11_detect, save_frames=False, save_json=False, save_period=-1, save_txt=False, scale=0.5, seed=0, shear=0.0, show=False, show_boxes=True, show_conf=True, show_labels=True, simplify=True, single_cls=False, source=None, split=val, stream_buffer=False, task=detect, time=None, tracker=botsort.yaml, translate=0.1, val=True, verbose=True, vid_stride=1, visualize=False, warmup_bias_lr=0.1, warmup_epochs=3.0, warmup_momentum=0.8, weight_decay=0.0005, workers=0, workspace=None
Overriding model.yaml nc=80 with nc=12

                   from  n    params  module                                       arguments
  0                  -1  1       464  ultralytics.nn.modules.conv.Conv             [3, 16, 3, 2]
  1                  -1  1      4672  ultralytics.nn.modules.conv.Conv             [16, 32, 3, 2]
  2                  -1  1      6640  ultralytics.nn.modules.block.C3k2            [32, 64, 1, False, 0.25]
  3                  -1  1     36992  ultralytics.nn.modules.conv.Conv             [64, 64, 3, 2]
  4                  -1  1     26080  ultralytics.nn.modules.block.C3k2            [64, 128, 1, False, 0.25]
```

```
  5                  -1  1    147712  ultralytics.nn.modules.conv.Conv             [128, 128, 3, 2]
  6                  -1  1     87040  ultralytics.nn.modules.block.C3k2            [128, 128, 1, True]
  7                  -1  1    295424  ultralytics.nn.modules.conv.Conv             [128, 256, 3, 2]
  8                  -1  1    346112  ultralytics.nn.modules.block.C3k2            [256, 256, 1, True]
  9                  -1  1    164608  ultralytics.nn.modules.block.SPPF            [256, 256, 5]
 10                  -1  1    249728  ultralytics.nn.modules.block.C2PSA           [256, 256, 1]
 11                  -1  1         0  torch.nn.modules.upsampling.Upsample         [None, 2, 'nearest']
 12             [-1, 6]  1         0  ultralytics.nn.modules.conv.Concat           [1]
 13                  -1  1    111296  ultralytics.nn.modules.block.C3k2            [384, 128, 1, False]
 14                  -1  1         0  torch.nn.modules.upsampling.Upsample         [None, 2, 'nearest']
 15             [-1, 4]  1         0  ultralytics.nn.modules.conv.Concat           [1]
 16                  -1  1     32096  ultralytics.nn.modules.block.C3k2            [256, 64, 1, False]
 17                  -1  1     36992  ultralytics.nn.modules.conv.Conv             [64, 64, 3, 2]
 18            [-1, 13]  1         0  ultralytics.nn.modules.conv.Concat           [1]
 19                  -1  1     86720  ultralytics.nn.modules.block.C3k2            [192, 128, 1, False]
 20                  -1  1    147712  ultralytics.nn.modules.conv.Conv             [128, 128, 3, 2]
 21            [-1, 10]  1         0  ultralytics.nn.modules.conv.Concat           [1]
 22                  -1  1    378880  ultralytics.nn.modules.block.C3k2            [384, 256, 1, True]
 23        [16, 19, 22]  1    433012  ultralytics.nn.modules.head.Detect           [12, [64, 128, 256]]
YOLO11n summary: 181 layers, 2,592,180 parameters, 2,592,164 gradients, 6.5 GFLOPs

Transferred 448/499 items from pretrained weights
Freezing layer 'model.23.dfl.conv.weight'
train: Fast image access  (ping: 0.20.1 ms, read: 180.3124.6 MB/s, size: 46.0 KB)
train: Scanning C:\Users\markm\Desktop\AIDA2154_Solar_Panel_Project\InfraredSolarModules\ImageSet\train\labels.cache... 6924 images, 7 backgrounds,
0 corrupt: 100% ━━━━━━━━━━━━ 6924/6924 6.9Mit/s 0.0s
WARNING Box and segment counts should be equal, but got len(segments) = 189, len(boxes) = 19845. To resolve this only boxes will be used and all seg
ments will be removed. To avoid this please supply either a detect or segment dataset, not a detect-segment mixed dataset.
val: Fast image access  (ping: 0.10.1 ms, read: 204.161.7 MB/s, size: 59.9 KB)
val: Scanning C:\Users\markm\Desktop\AIDA2154_Solar_Panel_Project\InfraredSolarModules\ImageSet\valid\labels.cache... 400 images, 0 backgrounds, 0 c
orrupt: 100% ━━━━━━━━━━━━ 400/400  0.0s
Plotting labels to C:\Users\markm\Desktop\AIDA2154_Solar_Panel_Project\Solar_Final_Runs\yolo11_detect\labels.jpg...
optimizer: 'optimizer=auto' found, ignoring 'lr0=0.01' and 'momentum=0.937' and determining best 'optimizer', 'lr0' and 'momentum' automatically...
optimizer: AdamW(lr=0.000625, momentum=0.9) with parameter groups 81 weight(decay=0.0), 88 weight(decay=0.0005), 87 bias(decay=0.0)
Image sizes 640 train, 640 val
Using 0 dataloader workers
Logging results to C:\Users\markm\Desktop\AIDA2154_Solar_Panel_Project\Solar_Final_Runs\yolo11_detect
Starting training for 15 epochs...

      Epoch    GPU_mem   box_loss   cls_loss   dfl_loss  Instances       Size
       1/15      4.27G      1.597      3.806      1.275         43        640: 100% ━━━━━━━━━━━━ 433/433 2.0it/s 3:35<0.5s
                 Class     Images  Instances      Box(P          R      mAP50  mAP50-95): 100% ━━━━━━━━━━━━ 13/13 1.8it/s 7.2s0.6s
                   all        400       1119      0.288      0.278      0.195      0.127

      Epoch    GPU_mem   box_loss   cls_loss   dfl_loss  Instances       Size
       2/15      3.89G      1.448      2.535      1.182         82        640: 100% ━━━━━━━━━━━━ 433/433 2.1it/s 3:29<0.5s
                 Class     Images  Instances      Box(P          R      mAP50  mAP50-95): 100% ━━━━━━━━━━━━ 13/13 1.9it/s 6.9s0.6s
                   all        400       1119      0.338      0.408      0.328      0.216

      Epoch    GPU_mem   box_loss   cls_loss   dfl_loss  Instances       Size
       3/15         4G      1.412      2.198      1.158         45        640: 100% ━━━━━━━━━━━━ 433/433 2.6it/s 2:44<0.3s
                 Class     Images  Instances      Box(P          R      mAP50  mAP50-95): 100% ━━━━━━━━━━━━ 13/13 3.1it/s 4.1s0.3s
                   all        400       1119      0.322      0.378       0.32      0.208

      Epoch    GPU_mem   box_loss   cls_loss   dfl_loss  Instances       Size
       4/15      3.96G      1.366      1.977      1.136         40        640: 100% ━━━━━━━━━━━━ 433/433 3.1it/s 2:21<0.3s
                 Class     Images  Instances      Box(P          R      mAP50  mAP50-95): 100% ━━━━━━━━━━━━ 13/13 3.2it/s 4.1s0.3s
                   all        400       1119      0.385      0.546      0.395      0.258

      Epoch    GPU_mem   box_loss   cls_loss   dfl_loss  Instances       Size
       5/15      3.96G      1.338      1.825      1.117         40        640: 100% ━━━━━━━━━━━━ 433/433 3.1it/s 2:19<0.3s
                 Class     Images  Instances      Box(P          R      mAP50  mAP50-95): 100% ━━━━━━━━━━━━ 13/13 3.6it/s 3.6s0.3s
                   all        400       1119      0.387      0.488      0.398      0.278
Closing dataloader mosaic

      Epoch    GPU_mem   box_loss   cls_loss   dfl_loss  Instances       Size
       6/15      3.96G      1.304      1.711      1.125         27        640: 100% ━━━━━━━━━━━━ 433/433 3.4it/s 2:07<0.3s
                 Class     Images  Instances      Box(P          R      mAP50  mAP50-95): 100% ━━━━━━━━━━━━ 13/13 3.6it/s 3.6s0.3s
                   all        400       1119      0.458      0.538       0.51      0.351

      Epoch    GPU_mem   box_loss   cls_loss   dfl_loss  Instances       Size
       7/15      3.96G      1.263      1.601      1.101         19        640: 100% ━━━━━━━━━━━━ 433/433 2.6it/s 2:45<0.3s
                 Class     Images  Instances      Box(P          R      mAP50  mAP50-95): 100% ━━━━━━━━━━━━ 13/13 2.8it/s 4.6s0.4s
                   all        400       1119      0.527      0.562      0.539      0.391

      Epoch    GPU_mem   box_loss   cls_loss   dfl_loss  Instances       Size
       8/15      3.96G       1.23       1.51      1.088         50        640: 100% ━━━━━━━━━━━━ 433/433 3.2it/s 2:17<0.3s
                 Class     Images  Instances      Box(P          R      mAP50  mAP50-95): 100% ━━━━━━━━━━━━ 13/13 3.7it/s 3.6s0.3s
                   all        400       1119      0.582      0.626        0.6      0.439

      Epoch    GPU_mem   box_loss   cls_loss   dfl_loss  Instances       Size
       9/15      3.96G      1.212       1.46      1.071         18        640: 100% ━━━━━━━━━━━━ 433/433 2.6it/s 2:45<0.4s
                 Class     Images  Instances      Box(P          R      mAP50  mAP50-95): 100% ━━━━━━━━━━━━ 13/13 2.5it/s 5.2s0.4s
                   all        400       1119      0.539      0.667      0.582      0.428

      Epoch    GPU_mem   box_loss   cls_loss   dfl_loss  Instances       Size
      10/15      3.96G      1.176      1.392      1.057         34        640: 100% ━━━━━━━━━━━━ 433/433 2.6it/s 2:47<0.3s
                 Class     Images  Instances      Box(P          R      mAP50  mAP50-95): 100% ━━━━━━━━━━━━ 13/13 2.6it/s 5.0s0.4s
                   all        400       1119       0.64      0.612      0.643      0.482

      Epoch    GPU_mem   box_loss   cls_loss   dfl_loss  Instances       Size
```

```
      11/15      3.96G      1.162      1.329      1.053         24        640: 100% ━━━━━━━━━━━━ 433/433 2.8it/s 2:35<0.3s
                 Class     Images  Instances      Box(P          R      mAP50  mAP50-95): 100% ━━━━━━━━━━━━ 13/13 2.9it/s 4.5s0.4s
                   all        400       1119      0.604      0.708      0.676      0.508

      Epoch    GPU_mem   box_loss   cls_loss   dfl_loss  Instances       Size
      12/15      3.96G      1.139       1.29      1.042         21        640: 100% ━━━━━━━━━━━━ 433/433 2.6it/s 2:46<0.4s
                 Class     Images  Instances      Box(P          R      mAP50  mAP50-95): 100% ━━━━━━━━━━━━ 13/13 2.5it/s 5.1s0.4s
                   all        400       1119      0.647      0.638      0.687      0.508

      Epoch    GPU_mem   box_loss   cls_loss   dfl_loss  Instances       Size
      13/15      3.96G      1.115       1.241      1.029         39        640: 100% ━━━━━━━━━━━━ 433/433 2.6it/s 2:49<0.4s
                 Class     Images  Instances      Box(P          R      mAP50  mAP50-95): 100% ━━━━━━━━━━━━ 13/13 2.4it/s 5.3s0.4s
                   all        400       1119      0.683      0.652      0.691      0.517

      Epoch    GPU_mem   box_loss   cls_loss   dfl_loss  Instances       Size
      14/15      3.96G      1.096       1.211      1.025         24        640: 100% ━━━━━━━━━━━━ 433/433 2.5it/s 2:51<0.3s
                 Class     Images  Instances      Box(P          R      mAP50  mAP50-95): 100% ━━━━━━━━━━━━ 13/13 2.5it/s 5.2s0.4s
                   all        400       1119      0.667      0.685      0.703      0.537

      Epoch    GPU_mem   box_loss   cls_loss   dfl_loss  Instances       Size
      15/15      3.96G      1.083       1.173      1.016         39        640: 100% ━━━━━━━━━━━━ 433/433 2.6it/s 2:48<0.3s
                 Class     Images  Instances      Box(P          R      mAP50  mAP50-95): 100% ━━━━━━━━━━━━ 13/13 2.5it/s 5.1s0.4s
                   all        400       1119      0.646      0.696      0.713      0.552

15 epochs completed in 0.705 hours.
Optimizer stripped from C:\Users\markm\Desktop\AIDA2154_Solar_Panel_Project\Solar_Final_Runs\yolo11_detect\weights\last.pt, 5.5MB
Optimizer stripped from C:\Users\markm\Desktop\AIDA2154_Solar_Panel_Project\Solar_Final_Runs\yolo11_detect\weights\best.pt, 5.5MB

Validating C:\Users\markm\Desktop\AIDA2154_Solar_Panel_Project\Solar_Final_Runs\yolo11_detect\weights\best.pt...
Ultralytics 8.3.235  Python-3.11.14 torch-2.10.0.dev20251205+cu128 CUDA:0 (NVIDIA GeForce RTX 5070 Laptop GPU, 8151MiB)
YOLO11n summary (fused): 100 layers, 2,584,492 parameters, 0 gradients, 6.3 GFLOPs
                 Class     Images  Instances      Box(P          R      mAP50  mAP50-95): 100% ━━━━━━━━━━━━ 13/13 2.2it/s 6.0s0.4s
                   all        400       1119      0.647      0.696      0.713      0.552
                  Cell         29         34      0.826      0.699      0.838      0.637
            Cell-Multi         39         77      0.461      0.506      0.505      0.383
              Cracking        138        209      0.577      0.752      0.725      0.591
                 Diode        142        263      0.795      0.863      0.884      0.728
           Diode-Multi         70        128      0.572      0.609      0.653      0.516
              Hot-Spot        122        323      0.717      0.789      0.787      0.622
         Hot-Spot-Multi         44         69      0.725      0.783      0.824      0.619
             No-Anomaly         14         16      0.499      0.562      0.492      0.317
Speed: 0.3ms preprocess, 3.6ms inference, 0.0ms loss, 2.0ms postprocess per image
Results saved to C:\Users\markm\Desktop\AIDA2154_Solar_Panel_Project\Solar_Final_Runs\yolo11_detect
```

[2]:
```python
import yaml
import torch
from ultralytics.data import build_dataloader
from ultralytics.cfg import get_cfg
from ultralytics.data.utils import check_det_dataset

print("🔍 Starting Data Loader Stress Test...")

try:
    # 1. Load default YOLO configuration (avoiding the 'val' key conflict)
    # We set the data config in overrides, not as the base config
    cfg = get_cfg(cfg=None, overrides={'data': 'final_solar_config.yaml', 'imgsz': 640, 'batch': 16})

    # 2. Parse the dataset file explicitly to get the image path
    data_info = check_det_dataset('final_solar_config.yaml')
    train_image_path = data_info['train']

    print(f"   Reading images from: {train_image_path}")

    # 3. Build the data loader (CPU only, no GPU math)
    loader = build_dataloader(cfg, batch=16, img_path=train_image_path, stride=32, rank=-1, mode="train")

    # 4. Stress Test: Read 50 batches from disk into RAM
    print("   Iterating through 50 batches (this tests your CPU, RAM, and Disk)...")
    for i, batch in enumerate(loader):
        if i >= 50: break
        # Print progress every 5 batches to keep output clean
        if i % 5 == 0:
            print(f"   ✅ Loaded batch {i}/50")

    print("\n🎉 SUCCESS: Data Loading is STABLE.")
    print("   CONCLUSION: Your Data, Disk, and RAM are healthy.")
    print("   The kernel crash is confirmed to be the Python 3.13 <-> PyTorch Nightly incompatibility.")

except Exception as e:
    print(f"\n❌ FAILURE: Data Loader crashed.")
    print(f"   Error details: {e}")
```

```
🔍 Starting Data Loader Stress Test...

❌ FAILURE: Data Loader crashed.
   Error details: argument of type 'NoneType' is not iterable
```

[ ]:
```python
# Cell 4: Validation & Metrics (Task 4)
# Evaluate the model on the validation set.
```

```
print("📊 Running Validation...")
metrics = model.val()

print(f"\n🏆 Results:")
print(f"   mAP@50:    {metrics.box.map50:.3f}")
print(f"   mAP@50-95: {metrics.box.map:.3f}")
print(f"   Precision: {metrics.box.mean_results()[0]:.3f}")
print(f"   Recall:    {metrics.box.mean_results()[1]:.3f}")
```

[ ]:
```
# Cell 5: Inference Demonstration (Task 4 Demo)
# Visualizing results on the unseen 'Test' set.

test_images_path = os.path.join(dataset_root, 'test', 'images')
test_files = glob.glob(os.path.join(test_images_path, "*.jpg"))

if test_files:
    # Pick 3 random images
    samples = random.sample(test_files, min(len(test_files), 3))

    print(f"🔍 Running inference on {len(samples)} random test images...")

    for img_path in samples:
        # Run Inference
        results = model.predict(img_path, conf=0.25, verbose=False)

        # Plotting
        result_plot = results[0].plot()

        # Convert BGR to RGB for Matplotlib
        result_rgb = cv2.cvtColor(result_plot, cv2.COLOR_BGR2RGB)

        plt.figure(figsize=(10, 8))
        plt.imshow(result_rgb)
        plt.axis('off')
        plt.title(f"Test Image: {os.path.basename(img_path)}")
        plt.show()
else:
    print("⚠️ No test images found in the dataset folder.")
```