**Solar Guard: Intelligent Defect Detection on Solar Panels using  DeepLearning**

**Project Overview**

This web-based application aims to classify and detect various solar panel conditions using deep learning. The pipeline is built using:

* ResNet18 for image classification
* YOLOv8s for object detection
* Streamlit for user interaction
* Matplotlib/Seaborn for data visualization and exploratory analysis

**1. Exploratory Data Analysis (EDA)**

**1.1 Class Distribution Overview**

* Visualization: Bar charts for both training and validation splits.
* Observation: Significant class imbalance, with underrepresented categories like Physical\_Damage, Objects, and Snow.
* Suggestion: Employ data augmentation or class rebalancing techniques like oversampling or synthetic data generation (e.g., SMOTE for images or copy-paste augmentation).

**1.2 Sample Image Previews**

* Description: Representative images for each class provide visual validation of the dataset.
* Value: Allows manual inspection for data anomalies or labeling issues.
* Enhancement: Add hover tooltips or overlay labels showing:
  + Image dimensions
  + Channel info (RGB/RGBA)
  + Source metadata

**1.3 Image Properties**

* Computed Statistics:
  + Average Image Width & Height
  + RGB Mean & Std Dev for each channel
  + Number of Channels (3 vs 4)
* Action Required: Standardize image channels (e.g., convert 4-channel images to RGB).

**2. Image Classification (ResNet18)**

Model Configuration

* Architecture: ResNet18 with pre-trained weights (ResNet18\_Weights.IMAGENET1K\_V1)
* Training Set Size: 610 images
* Validation Set Size: 146 images
* Epochs: 20
* Loss Function: CrossEntropy
* Optimizer: Adam

**Performance Summary**

| Metric | Train (Final) | Validation (Best @ Epoch 16) |
| --- | --- | --- |
| Accuracy | 92.8% | 94.5% |
| Loss | 0.1966 | 0.2126 |

* Observation: Excellent generalization on a small dataset.
* Peak Performance: Achieved at epoch 16 → Ideal checkpoint for deployment.
* Slight Overfitting Signs: After epoch 16, validation loss rises while training accuracy improves.

**3. Object Detection (YOLOv8s)**

Dataset Details

* Validation Set: 23 images
* Classes: 7 (Bird\_Drop, Clean, Dust, Electrical\_Damage, Physical\_Damage, Snow, objects)
* YOLO Version: 8.3.166 (CPU inference)

**Performance Metrics (Validation)**

| Class | Instances | Precision | Recall | mAP@0.5 | mAP@0.5:0.95 |
| --- | --- | --- | --- | --- | --- |
| Bird\_Drop | 4 | 0.000 | 0.000 | 0.0398 | 0.0103 |
| Clean | 3 | 1.000 | 0.000 | 0.287 | 0.0995 |
| Dust | 3 | 0.000 | 0.000 | 0.048 | 0.0076 |
| Electrical\_Damage | 6 | 0.825 | 0.500 | 0.546 | 0.153 |
| Physical\_Damage | 2 | 0.818 | 0.500 | 0.499 | 0.249 |
| Snow | 4 | 1.000 | 0.000 | 0.000 | 0.000 |

* Overall mAP@0.5: 0.203
* Overall mAP@0.5:0.95: 0.0742
* Recall: 0.143
* Precision: 0.521

**Key Insights**

* Only Electrical\_Damage and Physical\_Damage are detected with moderate confidence.
* All other classes have 0 recall, indicating no detections matched ground truth.
* High precision + 0 recall → Overconfident false positives.