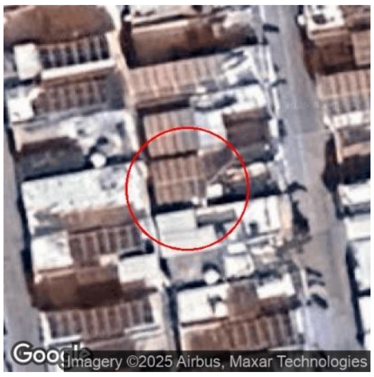
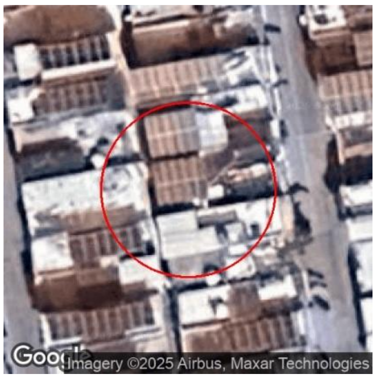
EcoInnovators Ideathon 2026 Challenge: AI-Powered Rooftop PV Detection – Round 2

Feedback (What to improve)

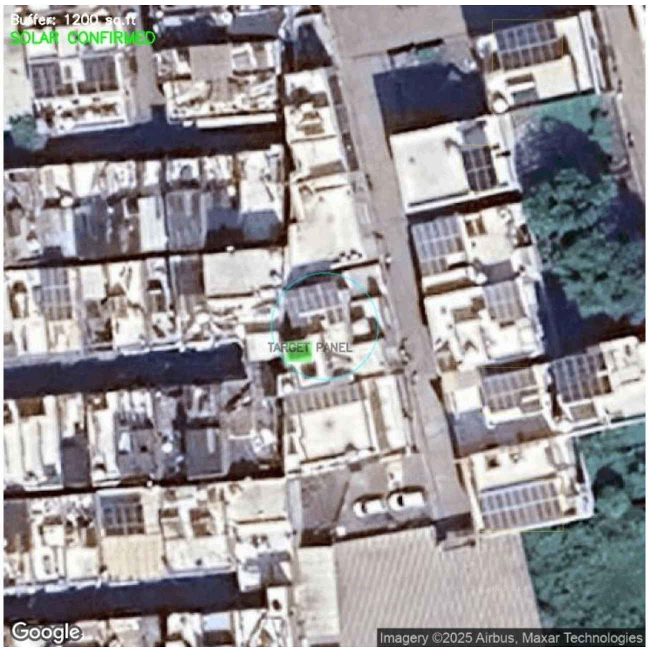
Listed below, are major findings from our evaluation of the solutions submitted in the first round. In the second round, we want you to optimize your solution to tackle these problems. Also there is a new parameter added in the evaluation criteria called "Verification metric" to measure the improved performance of your solution.

Type 1: In some images like shown below, the system is returning a 'No Solar Found' result despite solar panels being clearly visible. Even after expanding the search to a 2400 sq. ft. buffer radius, the model still fails to identify the solar panels. This is because model is not detecting any panel which lies inside the buffer zone. Model accuracy should be significantly improved.

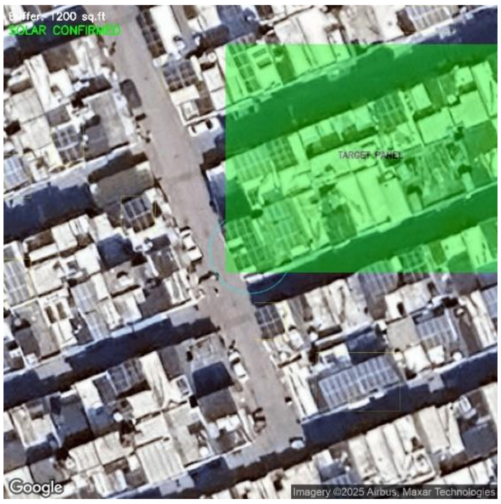


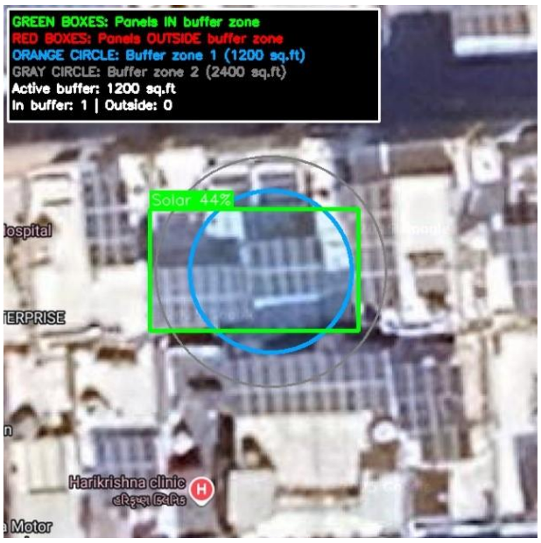


Type 2: In some cases like shown below, the system is returning the wrong panel. A smaller or more distant panel is being detected while the main target panel which has a much larger surface area is missed completely. The panel selection algorithm is correct. But the model is not accurate enough to detect both the panels inside the buffer zone. It is only detecting one panel and returning it. Model accuracy should be improved.

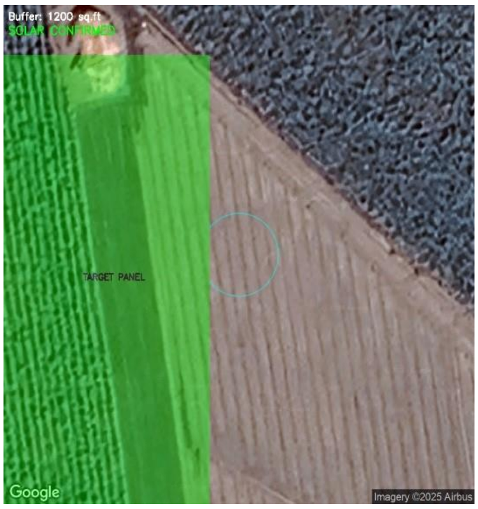


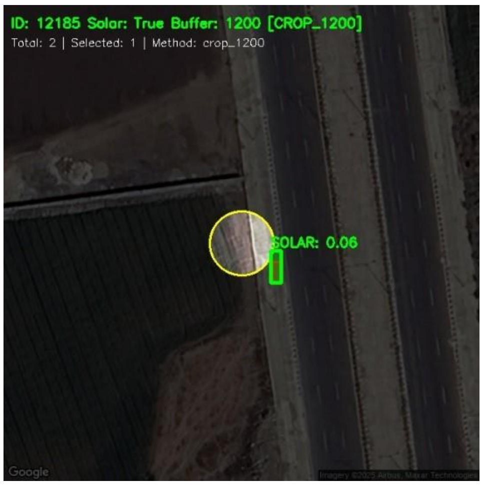
Type 3: In some cases, like shown below, the model incorrectly identifies large sections of the roof as a 'Target Panel'. This leads to faulty area calculation as well. Model accuracy needs to be significantly improved. Each panel must be labelled using a separate bounding box in the training data.

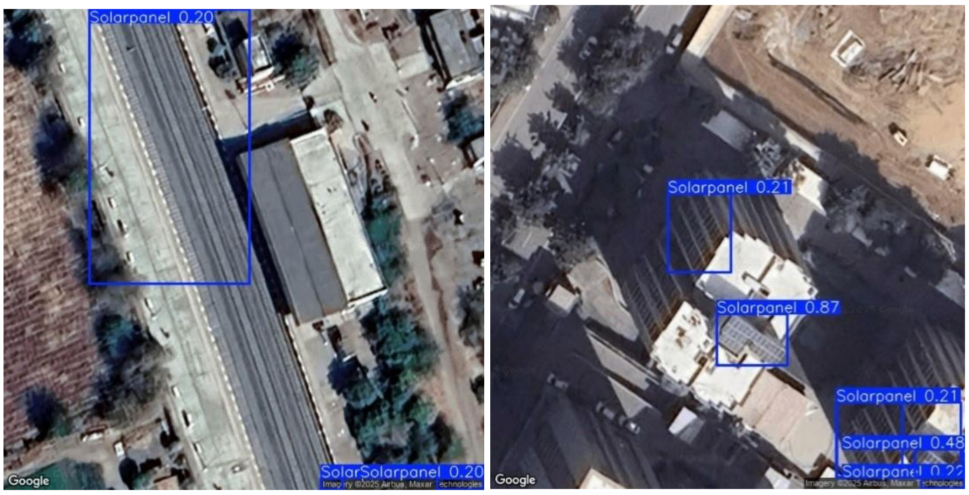



GREEN BOXES: Panels IN buffer zone
RED BOXES: Panels OUTSIDE buffer zone
ORANGE CIRCLE: Buffer zone 1 (1200 sq.ft)
GRAY CIRCLE: Buffer zone 2 (2400 sq.ft)
Active buffer: 1200 sq.ft
In buffer: 1 | Outside: 0
Solar 44%
Hospital
ERPRISE
Harikrishna clinic
Motor

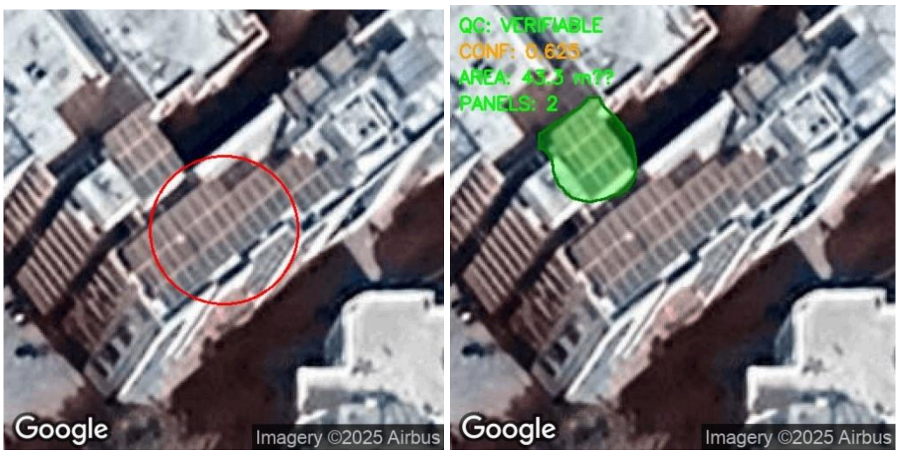
Type 4: In some cases, like shown below, the model incorrectly identifies non-solar features—such as roads, crop fields, or unrelated roof sections—as solar panels, resulting in a false 'Solar Confirmed' status. More images which have no solar panel have to be added to the training data to improve model accuracy for these cases.



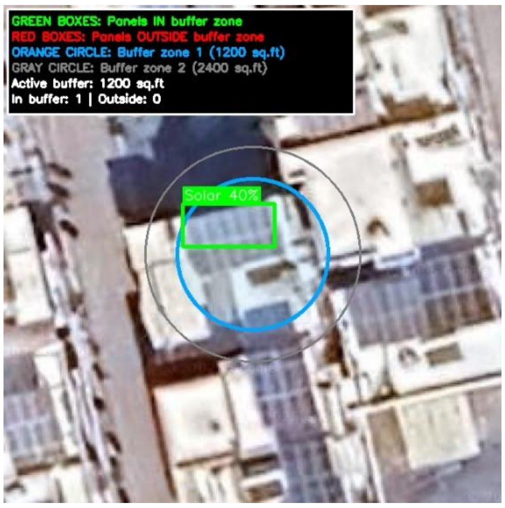
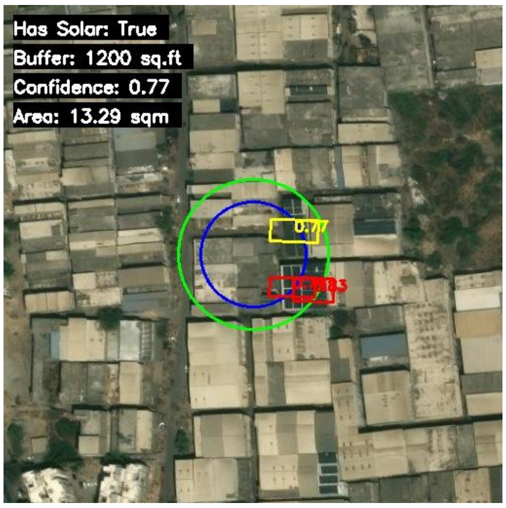

ID: 12185 Solar: True Buffer: 1200 [CROP_1200]
Total: 2 | Selected: 1 | Method: crop_1200
SOLAR: 0.06
Google



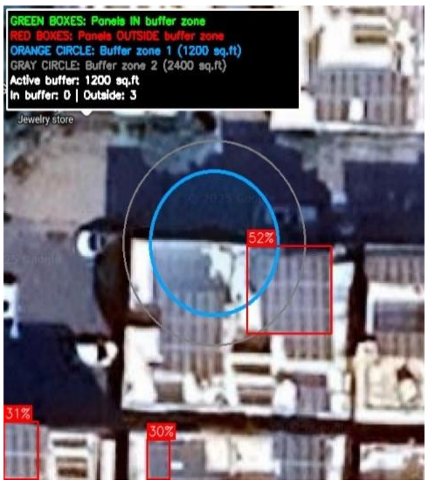
Type 5: In some cases like shown below, the system is returning the wrong panel. Like in Type 2, a smaller or more distant panel is being detected while the main target panel which has a much larger surface area is missed completely. However in this case, the smaller panel actually does not lie inside the buffer zone, its mask inaccurately lies in the buffer zone leading to error. Object detection could be used for detecting panel and image segmentation could be used for area calculation.



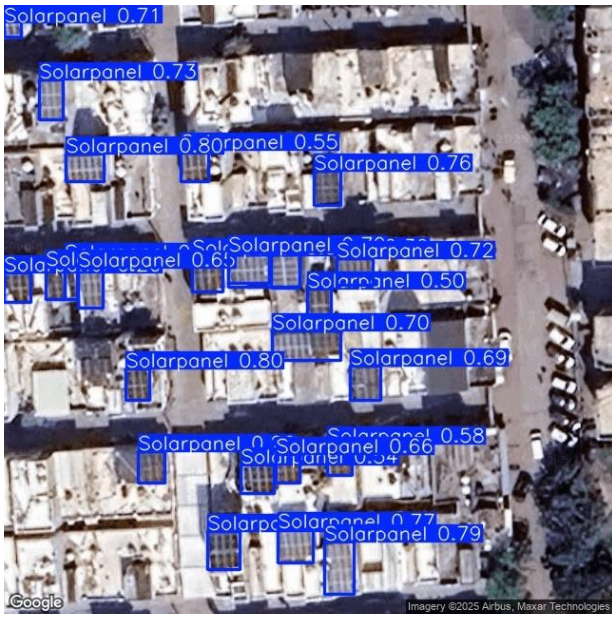
Type 6: In some instances, as seen below, the model successfully identifies the presence of a solar panel but fails to draw the bounding box accurately leading to inaccurate area calculation. The quality of labelling training data should be improved.


GREEN BOXES: Panels IN buffer zone
RED BOXES: Panels OUTSIDE buffer zone
ORANGE CIRCLE: Buffer zone 1 (1200 sq.ft)
GRAY CIRCLE: Buffer zone 2 (2400 sq.ft)
Active buffer: 1200 sq.ft
In buffer: 1 | Outside: 0
Solar 40%


Type 7: In some images, as seen below, the system returns no solar panel detected even though panel inside buffer zone is being detected. The algorithm needs to be rechecked.


GREEN BOXES: Panels IN buffer zone
RED BOXES: Panels OUTSIDE buffer zone
ORANGE CIRCLE: Buffer zone 1 (1200 sq.ft)
GRAY CIRCLE: Buffer zone 2 (2400 sq.ft)
Active buffer: 1200 sq.ft
In buffer: 0 | Outside: 3
Jewelry store
52%
31%
30%

Type 8: In some cases, all the panels in the image are highlighted. The buffer zone and selected panel are not highlighted. Overlays have to be redrawn.



Core Objectives (What your system must do)

When given an input folder location with a .xlsx containing list of sample id and corresponding geographic coordinates (latitude and longitude) and output folder location, your code must

1. Fetch: Automatically retrieve or accept a recent, high-resolution rooftop image for a given (lat, lon). You can use Google static maps Places API more info or ESRI API more info to extract the image.
2. Classify: Run inference on the extracted image by importing your trained model and determine whether rooftop PV is present (binary: present / not present) within a 1200 sq. ft radius buffer zone. If not present, determine whether rooftop present within a 2400 sq. ft radius buffer zone. Return a calibrated confidence score (if applicable).
3. Quantify: If PV is present, estimate total area of panel (m^2) which has the largest overlap with the selected buffer zone.
4. Verify: If PV is present, estimate the Euclidean distance (m) of the centroid of the selected solar panel from the given coordinate (lat,lon).
5. Explainability: Produce audit-friendly artifacts — image overlays, logic to compute area and a quality control (QC) status.
6. Store: JSON file and artifacts into output folder location

Required QC status values:

- VERIFIABLE — Clear evidence either way (present/not present).
- NOT_VERIFIABLE — Insufficient evidence (e.g., low resolution, heavy shadow/cloud, occlusion by tanks/trees, stale imagery).

Inputs & Outputs

Input data (available in .xlsx file):

- sample_id
- latitude, longitude (WGS84) — may include small geocoding jitters

Mandatory output per site (JSON record):

```
{
  "sample_id": 1234,
  "lat": 12.9716,
  "lon": 77.5946,
  "has_solar": true,
  "confidence": 0.92,
  "buffer_radius_sqft": 1200,
  "pv_area_sqm_est": 23.5,
  “euclidean_distance_m_est”:0,
  "qc_status": "VERIFIABLE",
  "bbox_or_mask": "<encoded polygon or bbox>",
  "image_metadata": {"source": "XYZ", "capture_date": "YYYY-MM-DD"}
}
```

Human-readable artifacts:

PNG/JPEG image overlays with polygon/bounding boxes. Highlight the selected panel in green and the remaining detected panels in red. Highlight the valid buffer zone using a yellow circle.

Evaluation Criteria (Weighted)

Your final score is a weighted aggregate across accuracy, auditability, efficiency, and ethics/operability.

Criterion	Metric / Evidence	Weight (%)
Detection accuracy	F1 score on has_solar	20
Quantification quality	RMSE for PV area (m^2)	20
Verification metric	RMSE for Euclidean distance (m)	20
Generalization & robustness	Performance across diverse cities/terrain	20
Others – code quality, documentation, usability		20

Deliverables (What to submit)

Please create a github repository more info and share the link. In the github repository, please create the following folders,

- Pipeline code: Must contain system code to run inference (.py)
- Environment details: Must contain requirement.txt for pip, environment.yml for conda, .txt file containing python version
- Trained model file: Must contain .pt, .joblib, .pkl etc file of trained model (if applicable)
- Model card: Must contain model card (.pdf, 2–3 pages). Model card could include data used, assumptions, logic, known limitations/bias, failure modes, and retraining guidance. more info
- Prediction files: Must contain prediction files for the training dataset (.json)
- Artefacts: Must contain artefacts for training dataset (.jpg, .png etc)
- Model Training Logs: Must contain a report or clear export of training logs (e.g., via CSV or MLflow export) documenting key metrics (Loss, F1 Score, RMSE) across training epochs/steps. more info
- README: Must contain clear run instructions for running the code.

Please use the format given below.

```
my-app/
├── pipeline_code/
│   ├── a.py
│   └── b.py
├── environment_details/
│   ├── python_version.txt
│   ├── requirements.txt
│   └── environment.yml
├── trained_model/
│   └── model.pt
├── model_card/
│   └── model_card.pt
├── prediction_files/
│   ├── train/
│   │   └── 1.json
│   └── test/
│      └── 2.json
├── artefacts/
│   ├── train/
│   │   ├── image.jpg
│   │   └── overlay.jpg
│   └── test/
│       ├── image.jpg
│       └── overlay.jpg
├── training_logs/
│   └── logs.csv
└── README.md
```