Solar Panel Detection and Model Training Workflow

# Introduction

* Solar panels present in 20-30% frames (l0b and lower)

**A screenshot of a computer screen

Description automatically generated**

* Divide each frame into 6x6 smaller squares (boxes) and count the total number of boxes that contain solar panels (ignoring the 4 corners).

A black circle with green lines in the middle

Description automatically generated

**A graph of solar panel boxes

Description automatically generated**

For orbit 1, a maximum of 7 out of 36 smaller boxes contain solar panels, covering up to 20% of a frame. However, in most cases, only 3–4 boxes (~10%) are affected.

**Solar panels in satellite images, usually at the start and end of orbits, lead to data loss when entire frames are excluded. Instead of discarding these frames, we can mask solar panel regions, preserving valuable data. By using segmentation to isolate solar panels, we streamline preprocessing and save more data.**

# Viewing Images and Identifying Solar Panel Intervals

1. **Initial Division of Images:**
   * Divided x into 8 sections and y into 4 sections.

A screenshot of a graph

Description automatically generated

* + Observed insufficient pixels in the four corner boxes.

1. **Refined Division:**
   * Divided x into 6 sections with lengths: 42, 43, 43, 43, 43, and 42.
   * Divided y into 6 sections with lengths: 42, 43, 43, 43, 43, and 42.
   * The resulting boxes are square or nearly square, ensuring symmetry.

A black circle with white squares

Description automatically generated

Because of the symmetry, data augmentation can be applied:

* + rotations (90°, 180°, 270°).
  + flips (up-down, left-right) to increase the dataset size.

Corner Boxes:

* + Ignored the four corner boxes due to insufficient pixels.
  + Added a logic rule: If adjacent boxes (1,1), (0,1), or (1,0) have solar panels, infer presence in (0,0).

# Labeling Solar Panel Intervals

* **File:**

[Solar Panels 6x6.xlsx](https://usu-my.sharepoint.com/:x:/g/personal/a02400392_aggies_usu_edu/EX4Nz4NtY2BMmTFUG-LfpisBsHZLA6vwqk2cAphPrXobnA?e=NLR2TF)

Notes:

* + Sharp change from frame 0 to frame 1

A screenshot of a graph

Description automatically generated

* + Solar panels were mis identified in box (4,0).

# Generating Training Images

1. **Reading the intervals**

A screen shot of a computer

Description automatically generated

1. **Defining Boxes:**

A screenshot of a computer code

Description automatically generated

1. **Orientation:**

A black background with white text

Description automatically generated

* + Boxes were upside down, but this did not affect training or predictions.

1. **Decision Boundaries:**
   * Excluded images within 4 frames of decision boundaries between solar panel (SP) and no solar panel (No SP) regions.
2. **Dataset Statistics:**
   * Total files: 49,766 across 2 folders.
     + No SP: 45,890 files.
     + SP: 3,876 files.
   * Created a balanced dataset:
     + Took 10% of No SP images.
     + Final counts:
       - No SP: 4,568 files.
       - SP: 3,876 files.

# Model Training

* Developed and trained a machine learning model for solar panel classification.

A computer code on a black background

Description automatically generated

A graph with blue lines and red and green lines

Description automatically generated

# Processing .nc Files and Organizing by Orbit

1. **Script Functions:**
   * Processed all .nc files in a specified folder.
   * Generated images for predictions.
   * Organized images into subfolders named after orbit numbers.
2. **Prediction Pipeline:**
   * A second script:
     + Processed images in each orbit subfolder.
     + Made predictions for all images.
     + (Computed a running average probability for 9 frames.) I calculate running average later.
     + Saved results to a CSV file.

Completed processing for Orbit 00110. Time taken: 4933.82 seconds (**82 minutes**)

**Optimization Using Batch Processing**

* Implemented batch processing to improve prediction speed:
  + Batch size 64: **187.08 seconds** (best performance, consistent).

A screenshot of a computer

Description automatically generatedNoted variations in performance after leaving the environment (Spyder) on for extended periods.

# Visualization and Results

* **Promising Results**

A screenshot of a computer screen

Description automatically generated

Comparing the solar panel box count from MLSP (using machine learning) and CSV (manual), using threshold value of 80%:

A graph showing a number of solar panels

Description automatically generated

* Observed good results, consistent compared to manual labelling near the beginning and end of the orbit.

Can use logic to address mislabeling in the middle of the orbit.

# Using Consecutive Frames:

* Differential in the movement of borders, solar panels, and airglow.
* Improved separation led to higher accuracy with just two orbits in the training dataset.

A graph with blue lines

Description automatically generated

* + - **Accuracy went up**

# Cluster Analysis and Binary Representation

1. **Heat Map and 3D Plot:** A colorful squares with numbers

   Description automatically generated with medium confidence

A screenshot of a graph

Description automatically generated

A diagram of a graph

Description automatically generated

# Cluster Identification Script:

* + Identified clusters of SP predictions near the corners at the beginning and end of the orbit.
  + Planned to find the five largest clusters.
  + Developed a Python script to:
    - Identify the largest clusters.
    - Save binary representations into a new NetCDF file named MLSPB.

A graph of a graph with colored dots

Description automatically generated with medium confidence

A graph of solar panel

Description automatically generated

* + - Matched the manual labeling better

Watch the demo:

[Solar Panel Demo.mp4](https://usu-my.sharepoint.com/:v:/g/personal/a02400392_aggies_usu_edu/ET9qvyI7P9xGjbiNlX5L0cUBlS5yz60xDBf-kjpMvESuXA?e=JafVbl)

**Future Improvements**

* **Changing Connected Component Labeling**:

structure = np.ones((3, 3, 3)) # 26-connected in 3D

use Face-Connected (6-Connected) in 3D or Edge-Connected (18-Connected) in 3D

* Add more labeled data. (**Collaboration?)**
* Address glare in images.
* Reduce file size using chunking and compression (e.g., zlib).
* Consider relationships between neighboring boxes (other approaches than clustering).
* Supervised Learning?
* "smooth" clusters to reduce noise and irregularities.

**Incorporating Human Input**

Specify:

* The first and last No SP frames for each orbit.
* Boxes containing solar panels.

Logic to prevent the cases where there’s a solar panel box in the middle of the frame, not connected to any other boxes near the edge (they are still connected in 3D):

A screenshot of a graph

Description automatically generated

Update Dec 11:

Labels: Orbits 1, 30, 60, 90

A graph with blue lines

Description automatically generated

Update Dec 18:

**Centroid Calculation**:

* Computes the centroid of each cluster for every frame (time step) where the cluster exists.
* A centroid represents the geometric center of the cluster in the 2D grid for each frame.

**Line Fitting**:

* Fits a **3D line** through the centroids of each cluster using **Principal Component Analysis (PCA)**.
* The line minimizes the orthogonal distance from the centroids, representing the trajectory of the cluster in 3D space (time, y, x). A graph of a number

  Description automatically generated with medium confidence

Why?

* ML model misidentifies some frames near the boundary between SP and no SP.
* We can extend the clusters to cover the last few SP boxes.
* We can also increase the probability of boxes near the 3D line.

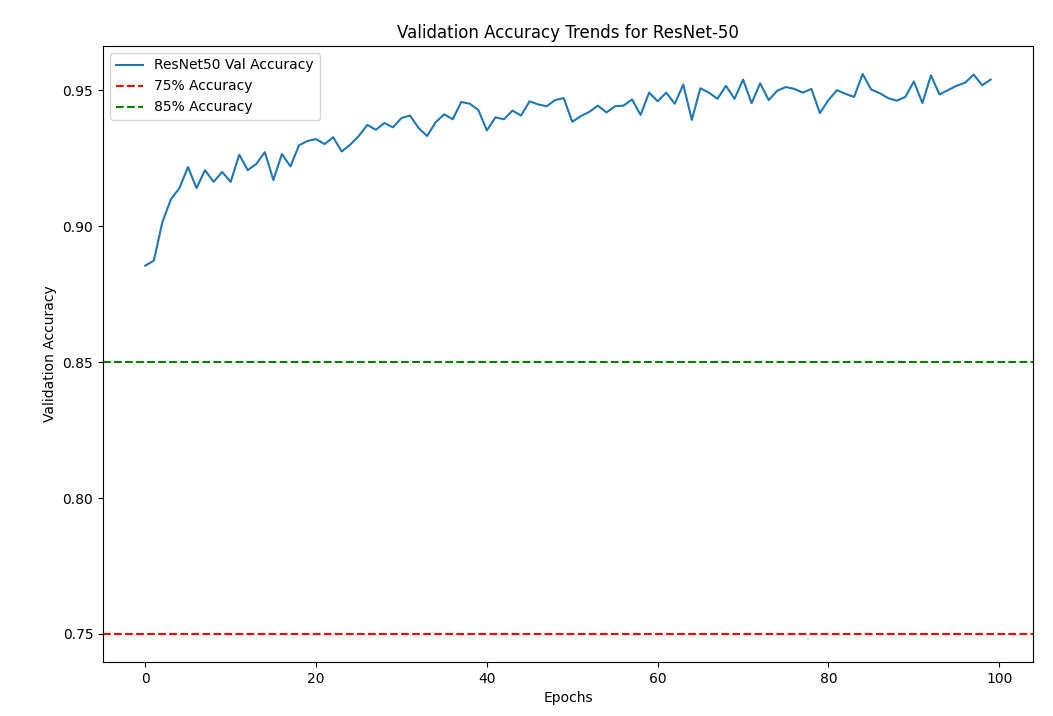
To do:

Modify the code. After getting the two clusters, find the fitted line through the centroids of each cluster.

Then, either lower the threshold for boxes near the line, or a similar way to expand the clusters.

Update Jan 10 2025

Added two new orbits



Jan 13

A graph of a graph

Description automatically generated with medium confidence

A graph of a box

Description automatically generated with medium confidence

Added orbits 1700, 1850, 1900 and retrained the model.

A graph with blue lines and green lines

Description automatically generated

Check confusion matrix threshold

Recall is defined as the ratio of correctly classified positive samples (True Positives) to the total number of actual positive samples. The formula for recall is:

Recall = True Positives / (True Positives + False Negatives)

Threshold the averaged MLSP data and remove points with x < 3

Tested orbits 1, 90, 110, 1550, 1600, 1650, 1700, 1750, 1800, 1850, 1900, 1950, 2000

It works pretty well in most cases.

Using both accuracy and recall when training:

A graph with blue lines

Description automatically generated

Re-evaluate the model for each value of threshold

* + - Try evaluating only once and then plot recall and accuracy for each value.

A graph with blue and orange lines

Description automatically generated

1750: NOT IN THE TRAIING DATA

2000: REALLY GOOD

1700: MAY AS WELL CUT HALF OF THEM OFF

models/DeepLearning\_resnet\_model\_sp\_acc.h5=> change this back

Use only accuracy: A graph with blue lines and red dots

Description automatically generated A graph with blue and orange lines

Description automatically generated

Radiance: [1, 300, 300]

Latitude: [1, 300, 300]

Longitude: [1, 300, 300]

Update Jan 21 2025

Orbits 1, 30, 60, 90, 113 (sp all orbit), 120, 1600, 1650, 1700, 1750, 1800, 1850, 1900, 1950, 2000

Use both acc and recall when training.

A graph showing a graph

Description automatically generated

A graph with blue and orange lines

Description automatically generated

1750:

in the training data,

works well (the best).

1775:

not in the training data,

works well but misses a few frames where the solar panels get small.

April data

2475:

not in the training data.

Near the end of April

Works okay, but misses more frames compared to the other orbits.

2375, 2175, 2075:

not in the training data

Works better than 1775

Focus on the cases when the model gets wrong.

Connor: switch from q20 to p12 or p14.

May data

2575, 2675: glare

2950, 2972: good