Lunar Vision-SOI Problem Final Report

---

1. Problem Overview

The Lunar Vision Challenge aimed to automate the detection and classification of surface features on the Moon—specifically, identifying **craters** and **boulders** in high-resolution lunar satellite images. The task involved training a model on annotated datasets and evaluating its performance on unseen test data. This problem has relevance to real-world space exploration, aiding in terrain analysis for future lunar missions.

---

2. Our Approach

We used **YOLOv8** for object detection due to its speed, accuracy, and flexibility in custom training. Our approach involved the following stages:

1. **Dataset Preparation:**
    - Dataset was extracted from the provided zipfile and structured into standard YOLOv8 format.
    - Each label file contained class, x_center, y_center, width, and height normalized to the image size.
2. **Model Training:**
    - We trained a custom YOLOv8n (nano) model due to compute constraints.
    - The model was trained for 10 epochs with a batch size of 8 using an Adam optimizer.
    - Image augmentation and validation split were employed to improve generalization.
3. **Evaluation:**
    - Post-training, we evaluated the model on the validation set.
    - Key metrics : **mean Average Precision (mAP@0.5)**, **precision**, and **recall** were logged.

- We ensured that both **crater (class 0)** and **boulder (class 1)** detections were captured to the best of the model's capabilities.

4. **Testing Phase:**
    - The model was used to generate predictions on the test set.
    - Each prediction was saved as a `.txt` file following the `filename_label.txt` format.
    - Additionally, we saved visualizations of each prediction overlayed on the original image.

---

3. Implementation Details

- **Libraries Used:**
    - PyTorch
    - Ultralytics (YOLOv8)
    - OpenCV
    - matplotlib
    - numpy
    - tqdm
- **Model Configuration:**
    - `imgsz=640`
    - `epochs=10`
    - `device='cuda' if available else 'cpu'`
    - `optimizer=Adam`

- **Evaluation Snippet:**

```
results = model.val()
metrics_dict = results.results_dict

print(f"mAP@0.5: {metrics_dict['metrics/mAP50(B)']:.4f}")
print(f"mAP@0.5:0.95: {metrics_dict['metrics/mAP50-95(B)']:.4f}")
print(f"Precision: {metrics_dict['metrics/precision(B)']:.4f}")
print(f"Recall: {metrics_dict['metrics/recall(B)']:.4f}")
```

---

4. Results

Metric	Value
mAP@0.5	0.8596
mAP@0.5:0.95	0.6526
Precision	0.8142
Recall	0.7635

The model demonstrated strong performance on craters and acceptable precision for boulder detection, despite the imbalance in training samples.

---

5. Visualization

We saved two forms of results for the test set:

- `.txt` label files with bounding box predictions
- Image overlays with colored boxes (blue for craters, red for boulders)

These visuals aided in qualitative verification of detection accuracy.

---

6. Key Learnings & Insights

- **Data imbalance**: Boulders were less frequent in the dataset, leading to lower confidence predictions for class 1.
- **Augmentations** like flipping and brightness adjustments improved generalization.
- **Transfer learning** via pre-trained YOLOv8 weights boosted initial performance.

Real-world use cases:

- Terrain risk analysis for rover navigation.
- Automated crater count for geological age estimation.
- Resource extraction planning (e.g., locating safe flat zones).

---

7. Future Scope

- **Fine-tuning with high-res inputs** using YOLOv8m or YOLOv8l.
- Adding metadata such as elevation or radar data for better boulder prediction.
- Creating a Streamlit-based interface for public demonstrations.

---

8. Files in Submission

- `best.pt` – Trained model
- `label_generator.py` – Inference and label generation script
- `README.md` – Guide to run code
- `requirements.txt` – Dependency list
- `test_labels/` – Output `.txt` files
- `test_visuals/` – Prediction overlays

- `report.pdf` – This document

---

9. Conclusion

This project gave us hands-on experience in object detection, model training, metric evaluation, and real-world problem-solving. We learned to handle messy data, tune hyperparameters, and interpret spatial patterns from satellite images. It was both challenging and rewarding.

Thank you for the opportunity.

— Team Chaand Sitaare