Lunar Surface Image Segmentation

Sahara Ensley

December 12, 2022

George Washington University

Machine Learning II - DATS_6203_10

GitHub Repo

Joshua Ting

Going to the Moon!!

- Tasked to help land a lunar lander on the surface of the moon

- Many factors to consider in finding a safe landing location
  - Be nice to **not crash into rocks**

- Can we use ML/CV to help with this?

Figure 1: Lunar Surface[4]

Overview of Project Lifecycle

Research, Data, and Background → Data Download and EDA → Data Preprocessing → Model Training → Interpreting Model Results

Overview of Project Lifecycle

Background

- Semantic Segmentation: **Classifying the class per pixel**

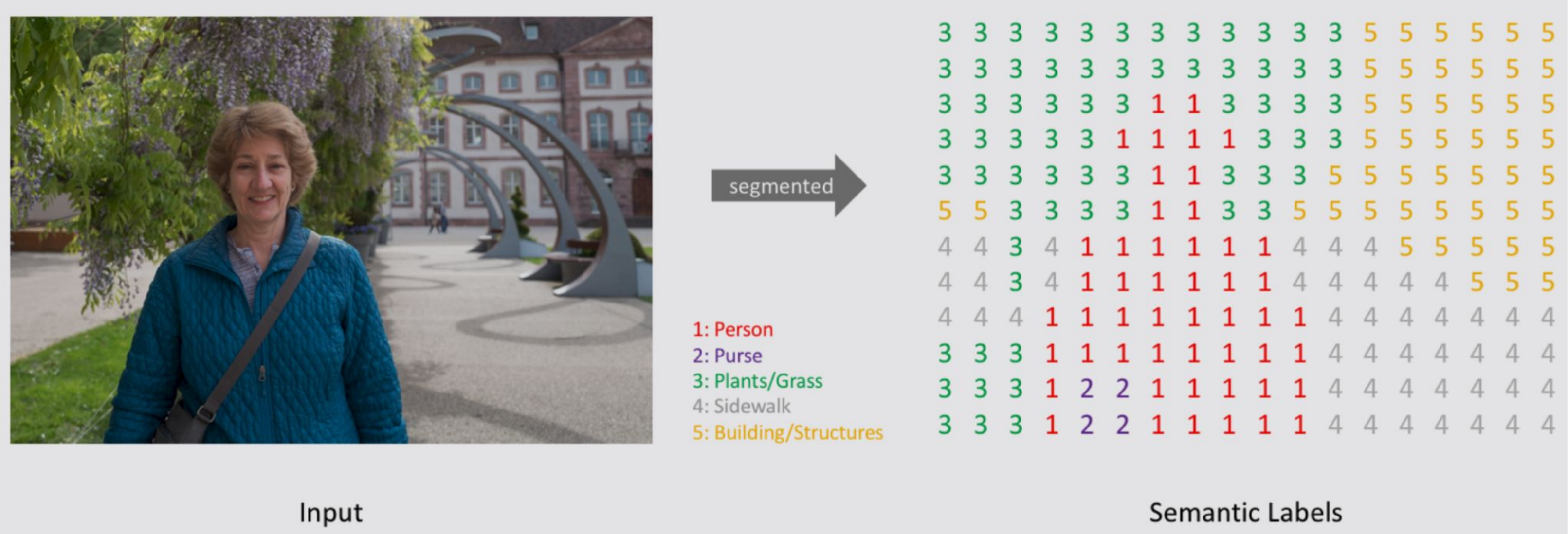


Figure 2: Overview of Semantic Segmentation[5]

Data Overview

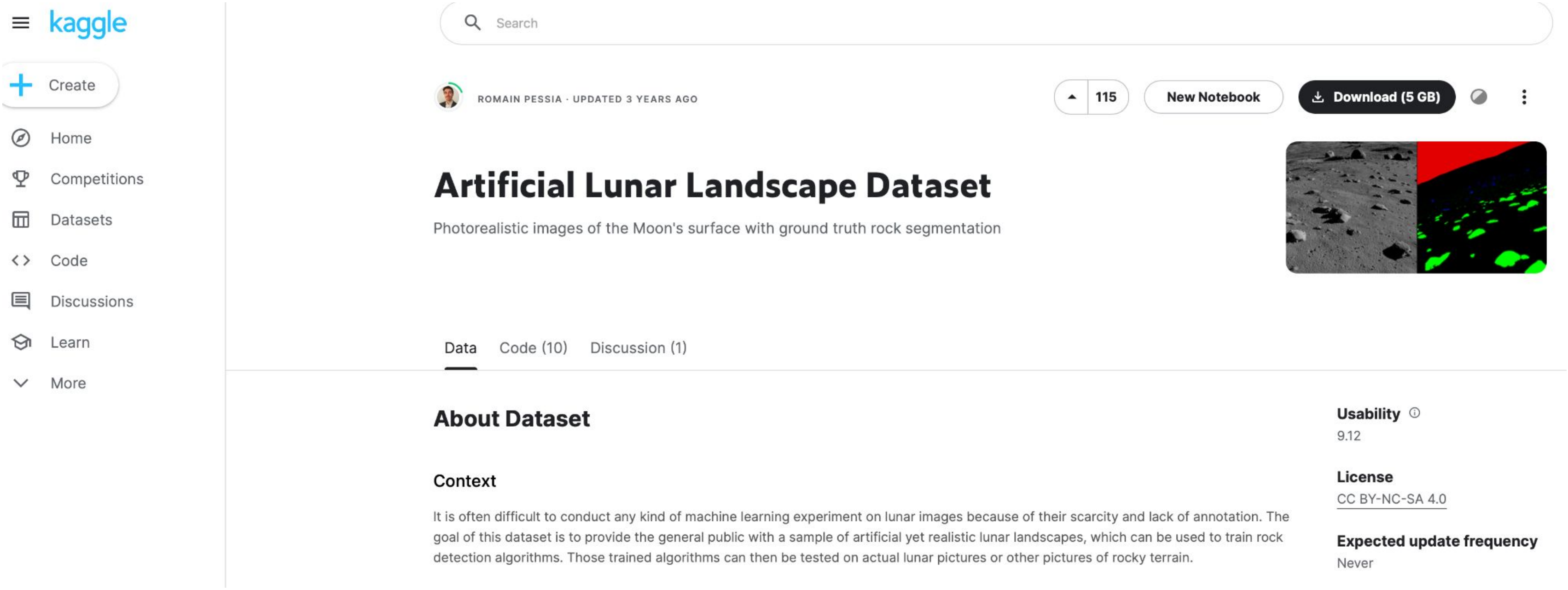
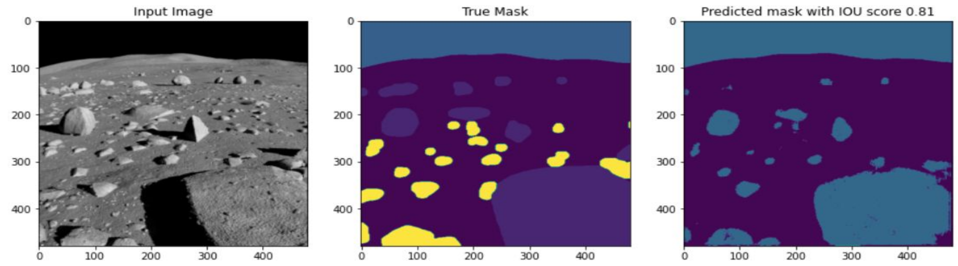


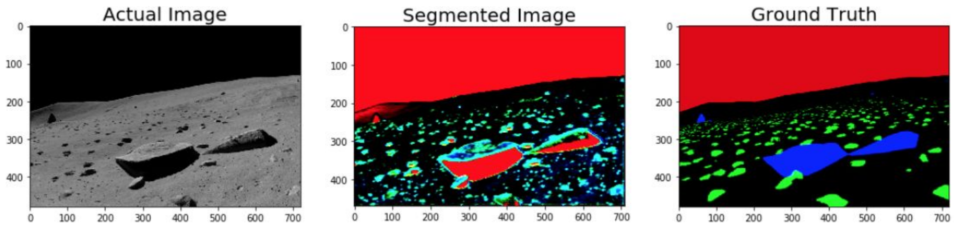
Figure 3: Artificial Lunar Landscape Dataset from Kaggle[3]

Prior Work on Dataset - Kaggle Notebooks[7]

- Gold: Tried VGG16 model



- Silver: Tried VGG16 model



Objectives and Contribution Goals

To build and train our **own U-Net** based architecture then compare results against a few **pretrained networks**

Dataset

- **Rendered Lunar Images:**
  - 9,766 Images of Rendered Lunar Landscapes
  - 9,766 Ground Truth Masks

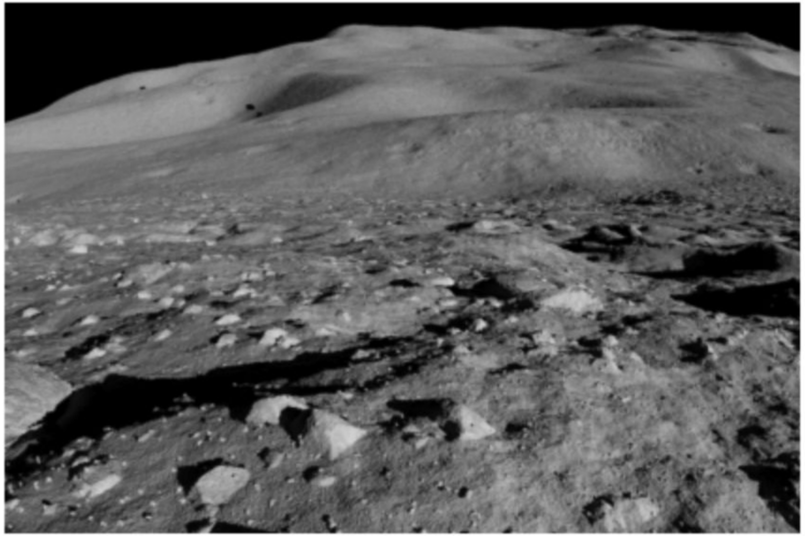
- **Real Lunar Images for Testing:**
  - 36 Images of Real Lunar Images
  - 36 Ground Truth Masks

Dataset Split	Type	Percentage of Type
Train	Rendered	49%
Validation	Rendered	21%
Test - Rendered	Rendered	30%
Test - Real	Real	100%

Figure 4: Dataset Split

Rendered Image/Mask Example

Original:
render0034.png



Ground Truth Mask:
clean0034.png

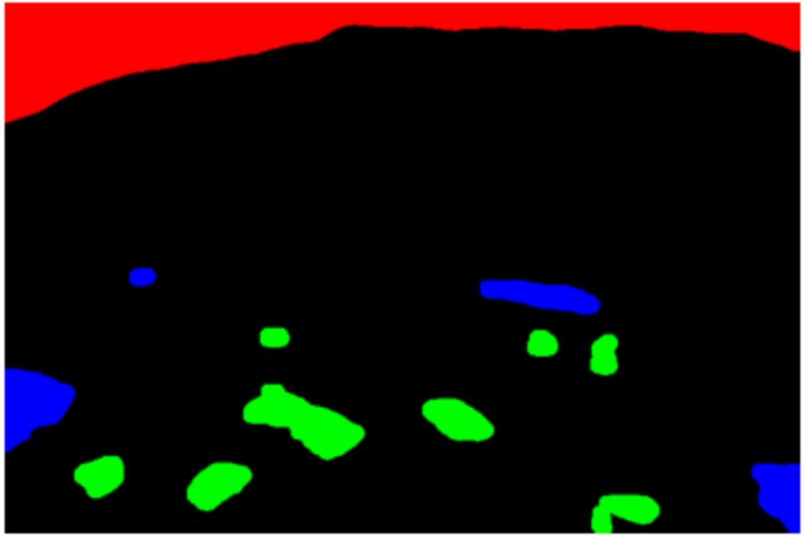


Figure 5: Rendered image and ground truth mask example[3]

Rendered Image/Mask Example w/ Color Key Labels

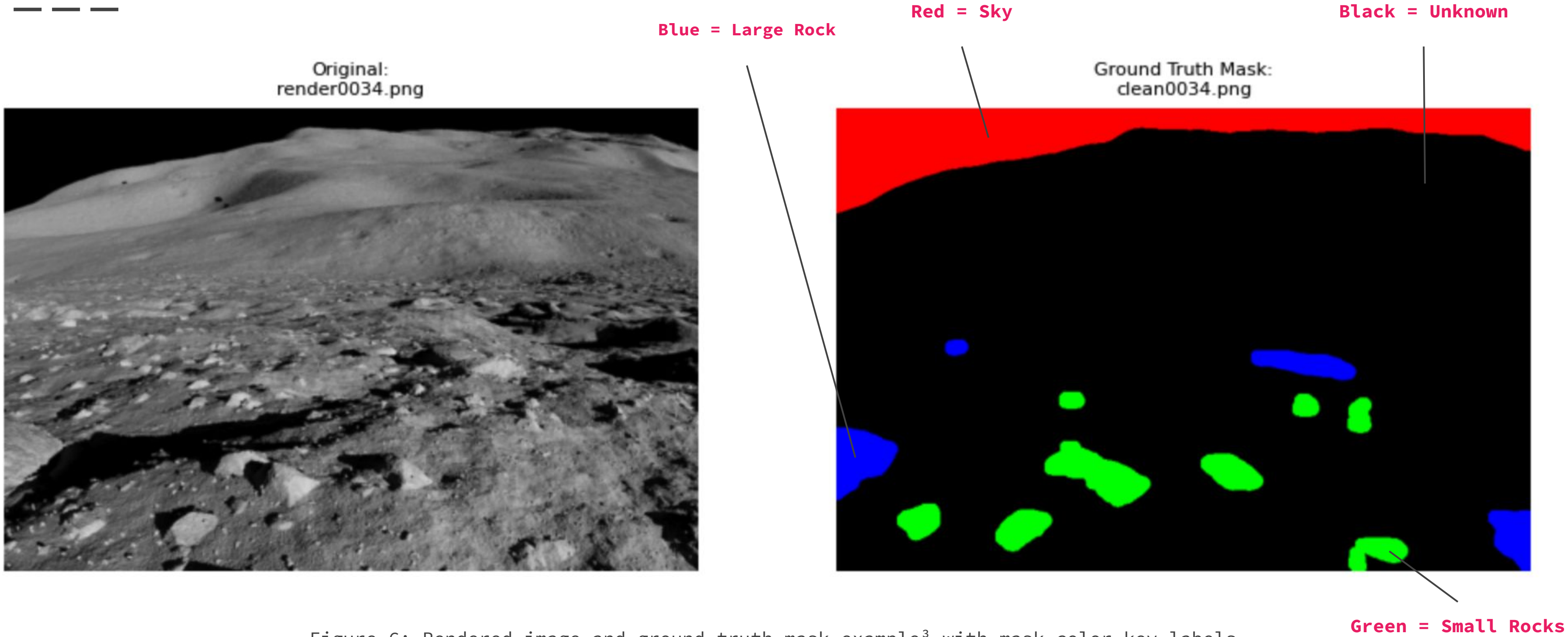


Figure 6: Rendered image and ground truth mask example[3] with mask color key labels

Rendered Image/Mask Example II

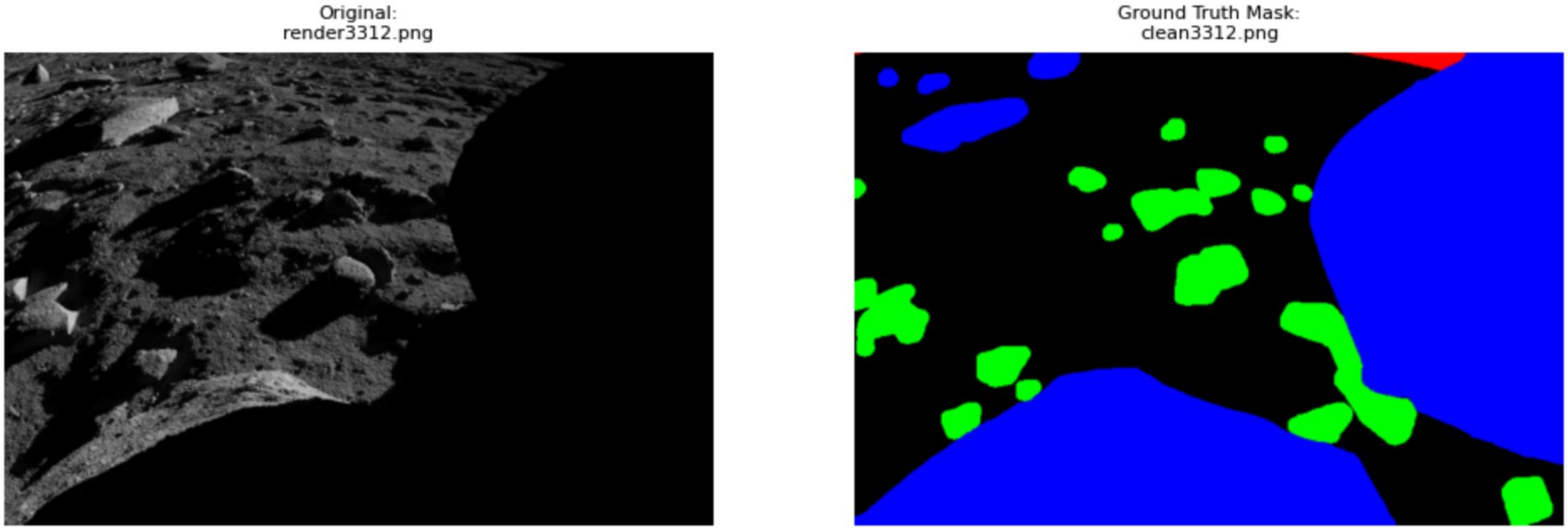
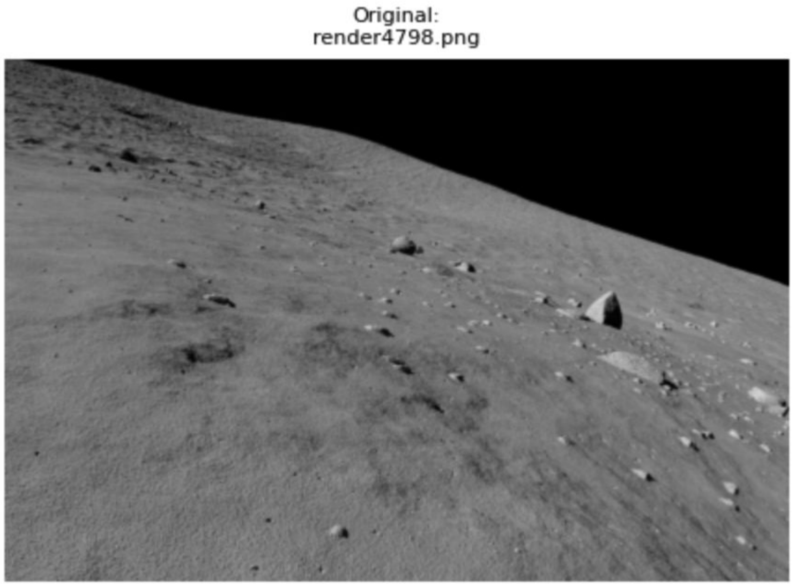


Figure 7: Rendered image and ground truth mask example[3]

Rendered Image/Mask Example III



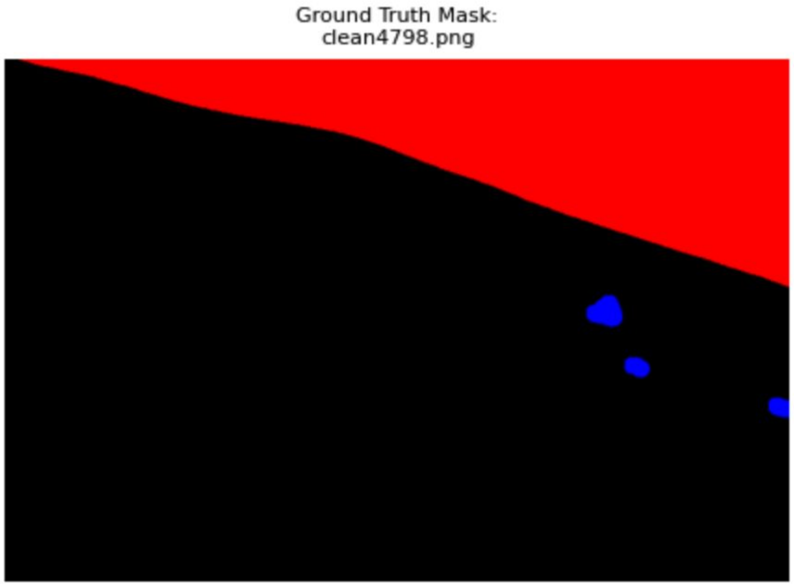


Figure 8: Rendered image and ground truth mask example[3]

Rendered Image/Mask Example IV

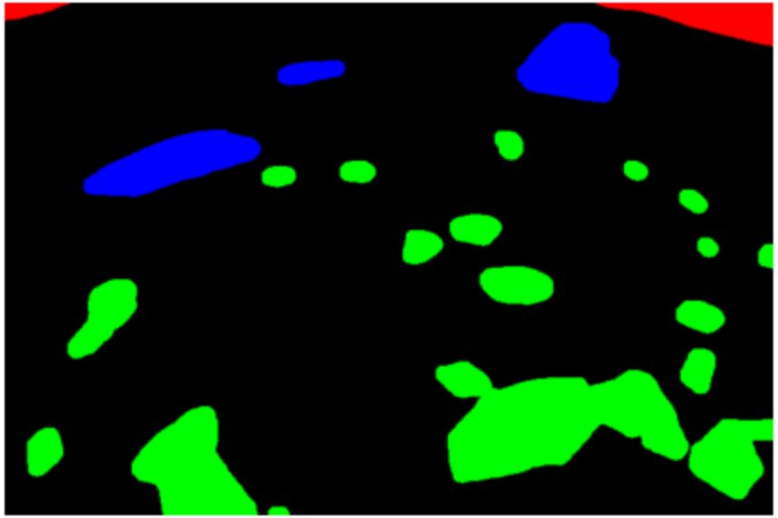
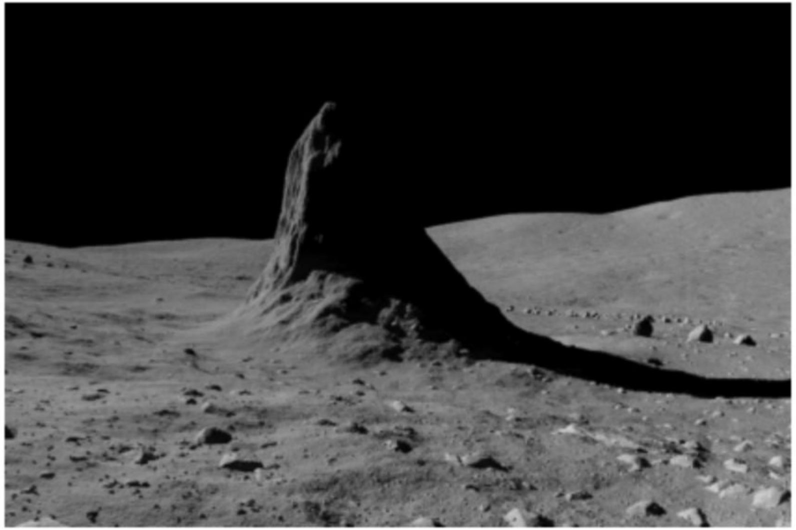


Figure 9: Rendered image and ground truth mask example[3]

Rendered Image/Mask Example V

Original:
render8455.png



Ground Truth Mask:
clean8455.png

Figure 10: Rendered image and ground truth mask example[3]

Real Image Example

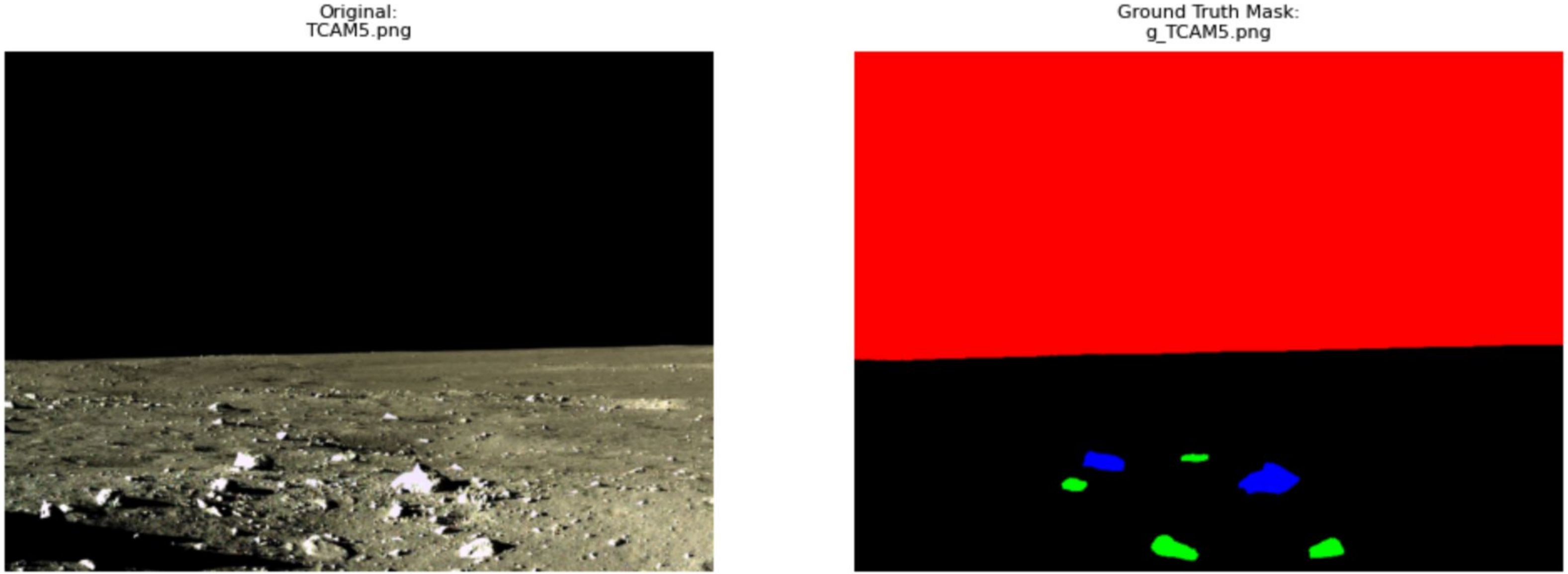


Figure 11: Real lunar image and ground truth mask example[3]

Real Image Example II

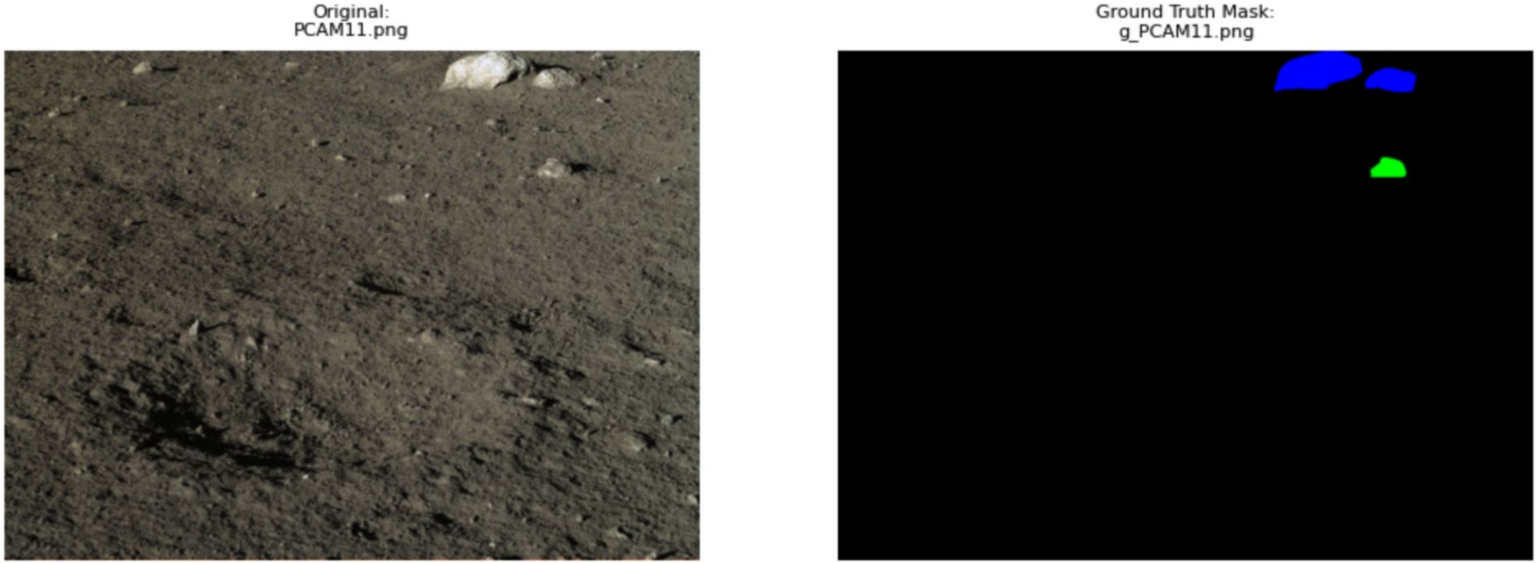


Figure 12: Real lunar image and ground truth mask example[3]

Real Image Example III

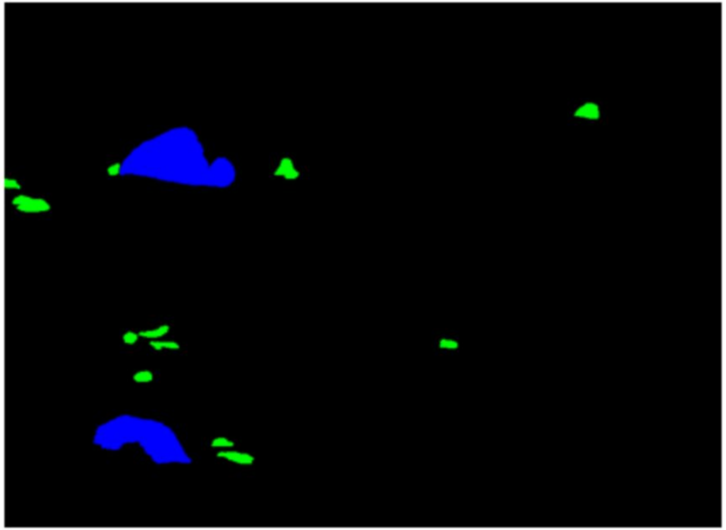


Figure 13: Real lunar image and ground truth mask example[3]

Overview of Project Lifecycle

Research, Data, and Background → Data Download and EDA → Data Preprocessing → Model Training → Interpreting Model Results

EDA

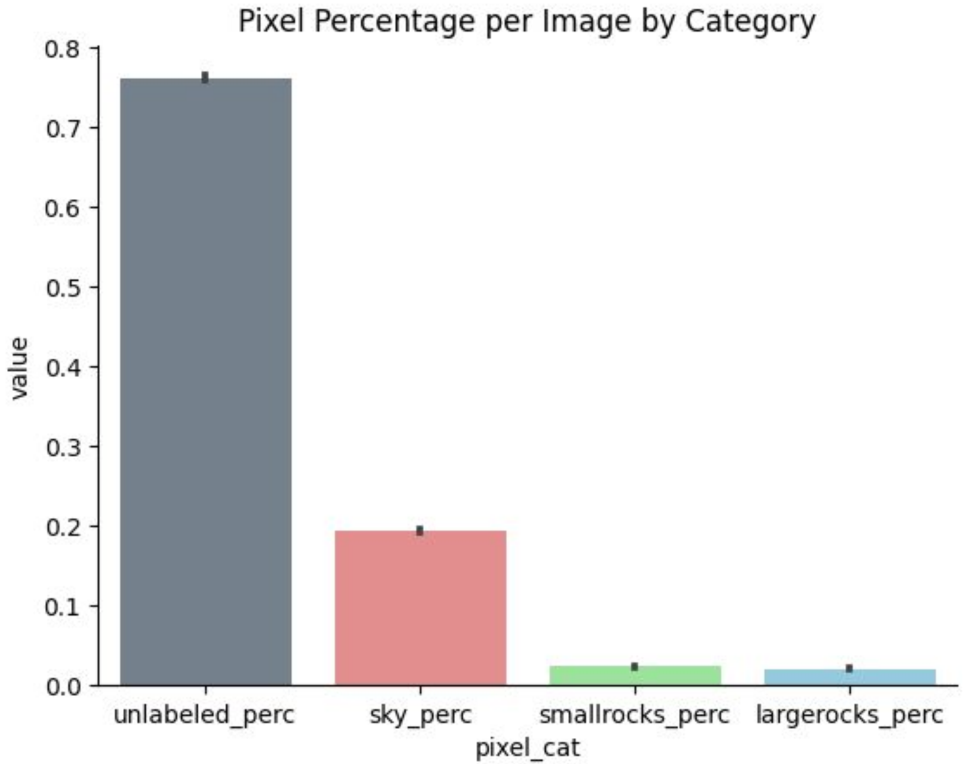


Figure 14: Class Imbalance

EDA

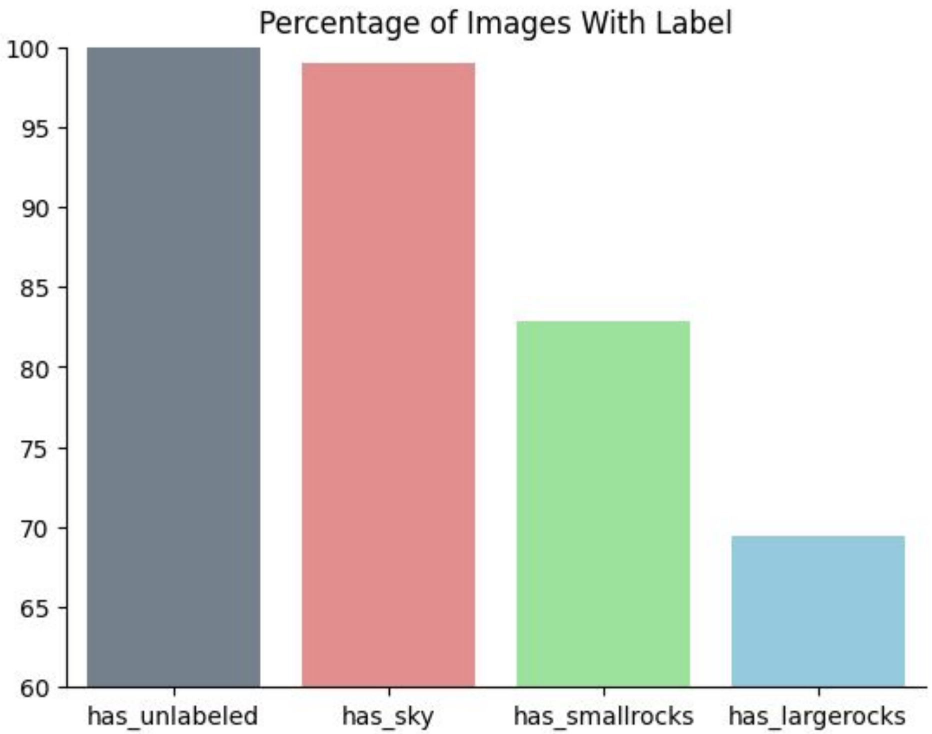


Figure 15: How Many Images Have Which Class Labels

EDA

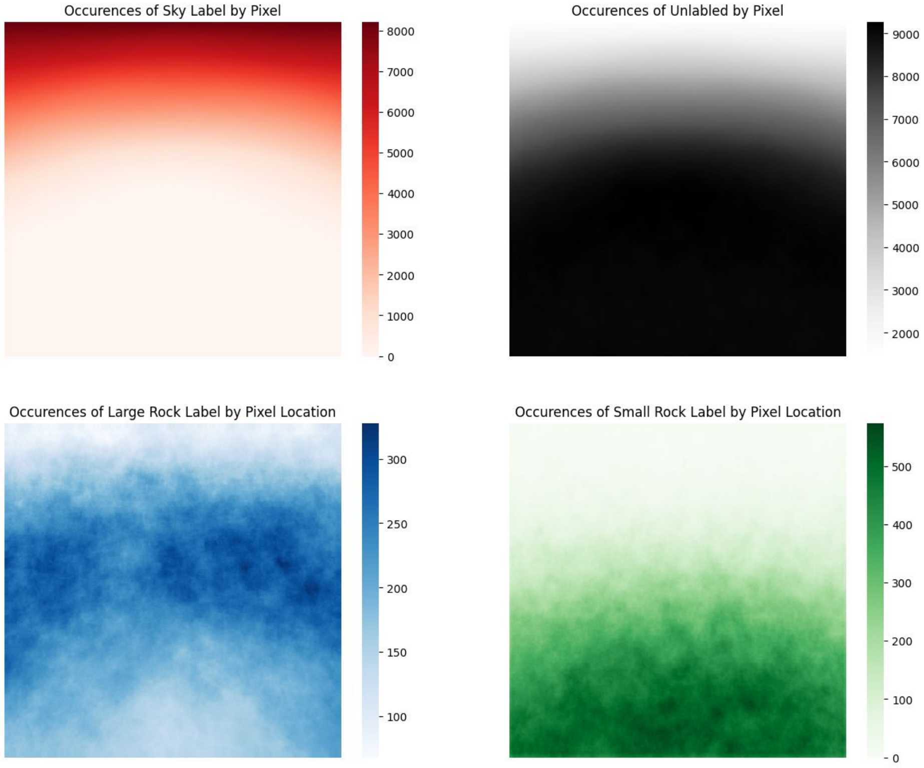


Figure 16: Overall Class Occurence Intensity by Image Location

Overview of Project Lifecycle

Data Preprocessing

- **Custom DataLoader**
  - Loads image and mask
  - Resize to 256x256
  - Applies data augmentation
  - Rescale pixels to between 0-1
  - One Hot Encode masks
  - Change ordering of channels - image dim last
  - Convert to torch tensor
- **Torch DataLoader**
  - Batch data

One Hot Encoding the Ground Truth Mask

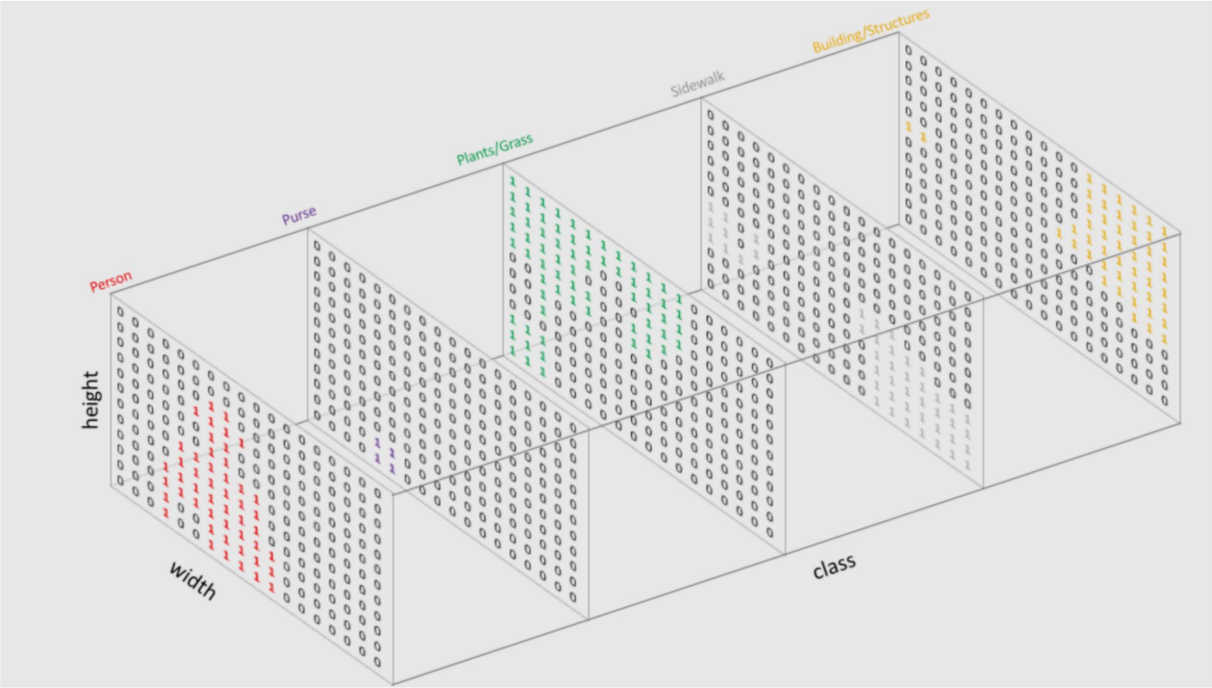


Figure 17: One Hot Encoding the Ground Truth Masks[5]

Data Augmentation

- Images & Masks
    - Random **Vertical Flip**
    - Random **Horizontal Flip**

- Images Only
    - Random **Color Jitters**
        - Brightness
        - Contrast
        - Saturation
        - Hue

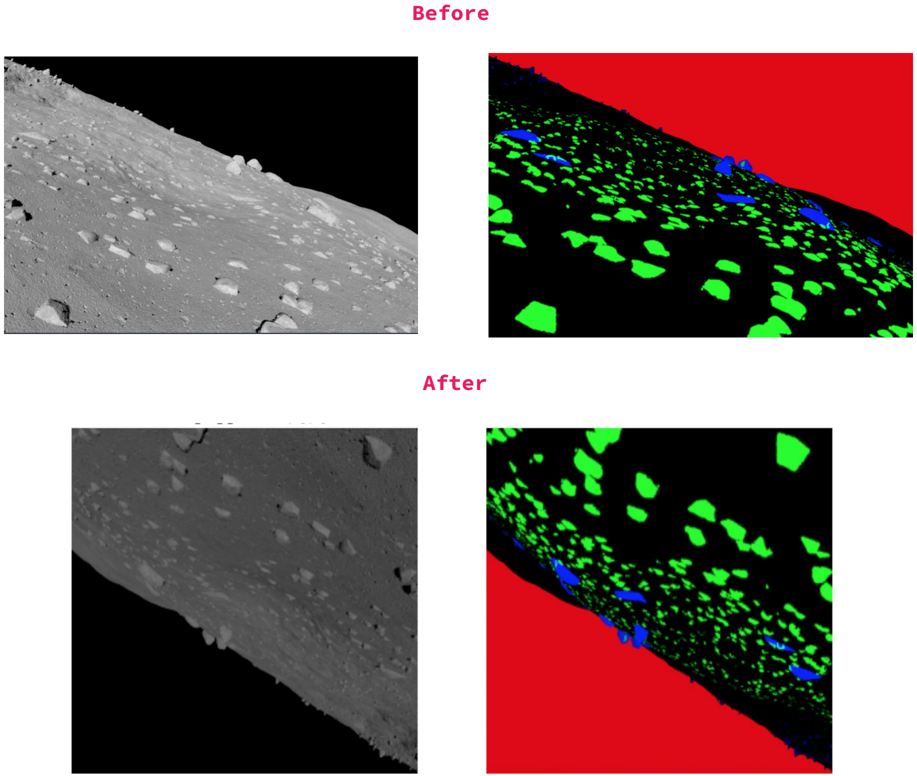


Figure 18: Before and after data augmentation

Overview of Project Lifecycle

Model Selection: U-Net

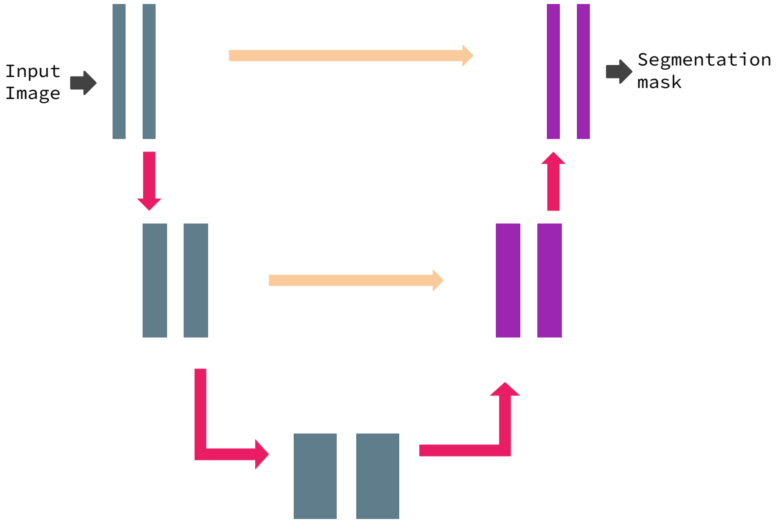


Figure 19: General U-Net.

Encoder - Decoder model

- **Encoder**:
  - Down samples the image with convolutional layers and max pooling
- **Decoder**:
  - Up samples the image with transposed convolutional layers

Feature maps from the encoder of the same level are concatenated and passed through the decoder

Our Model

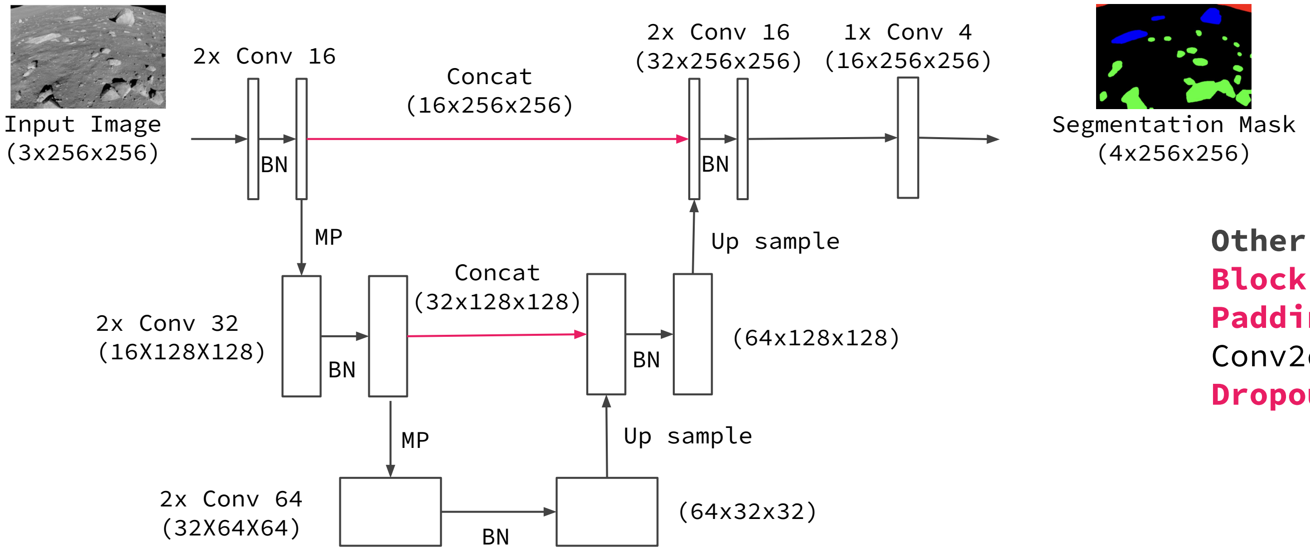


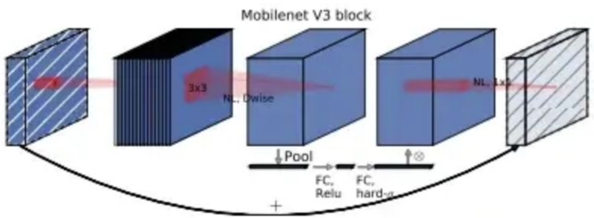
Figure 20: Our detailed U-Net diagram.

*MP = Max Pooling
*BN = Batch Norm

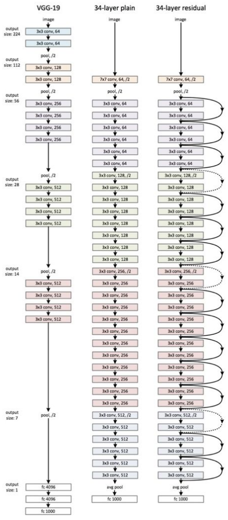
Other Notes:
Block Activation: ReLU
Padding: Same
Conv2d **Filters**: 3x3
Dropouts between all blocks

Pre-Trained Models

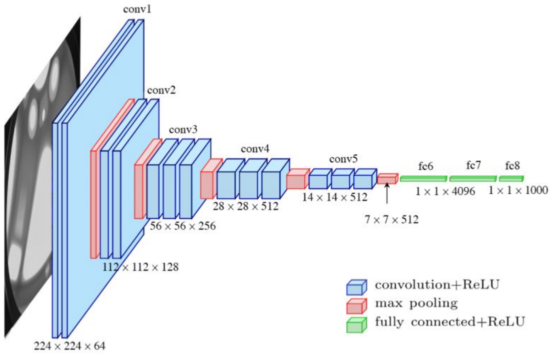
MobileNetv3 Large (Google)



ResNet18



VGG11



Parameters	5.4 M	11 M	18 M

Figure 21: 3 different pre-trained models.

Metrics

Jaccard's Index (IOU):

- Measures how much overlap exists between the predicted mask and the true mask

$$J(A,B) = \frac{|A \cap B|}{|A \cup B|} = \frac{|A \cap B|}{|A| + |B| - |A \cap B|}$$

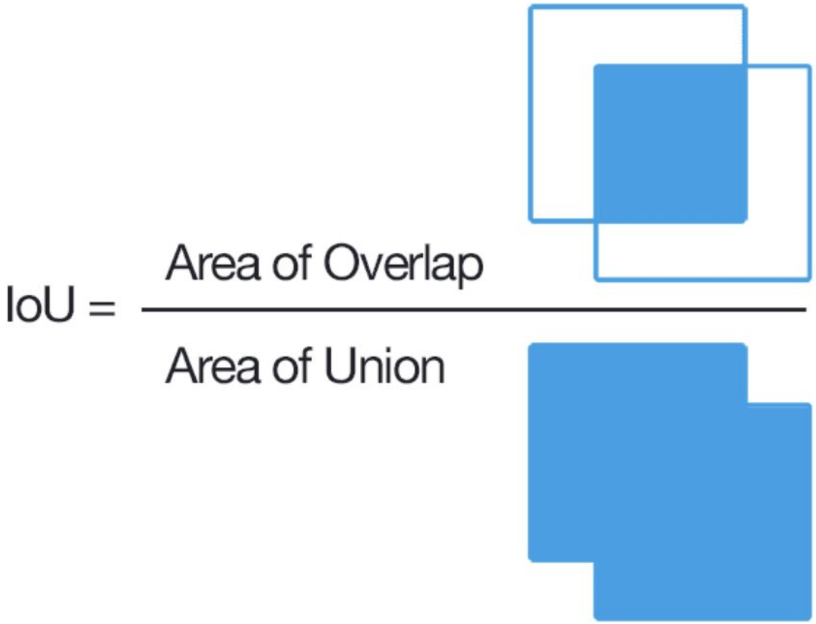
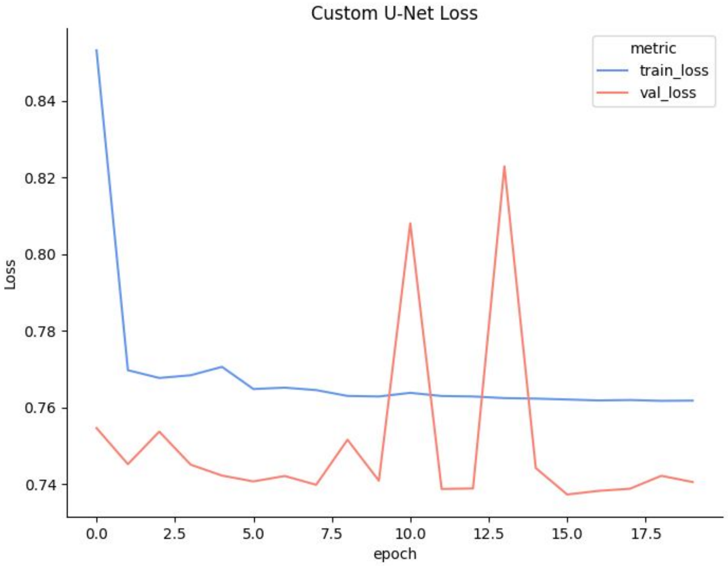
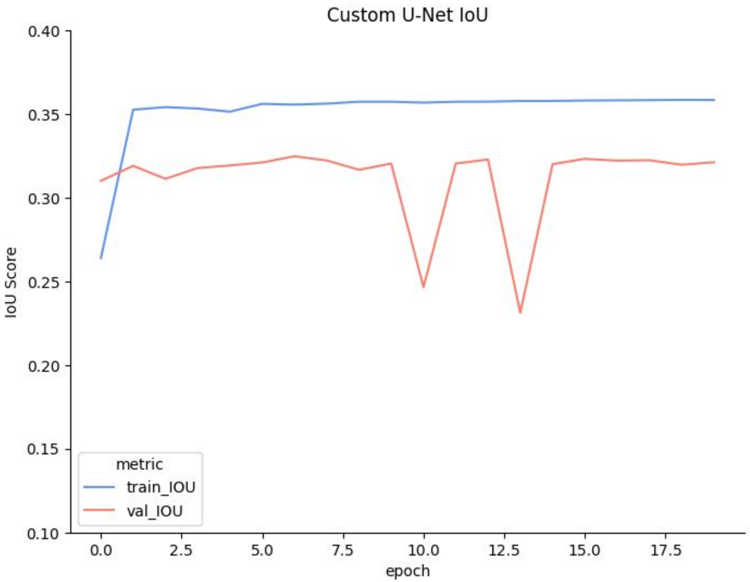


Figure 22: IOU calculation.

Model Training: Custom U-Net





Epochs - 20
Optimizer - Adam
Loss - Cross Entropy Loss

Figure 23: Custom U-Net training curves.

Model Training: Pre-trained U-Nets

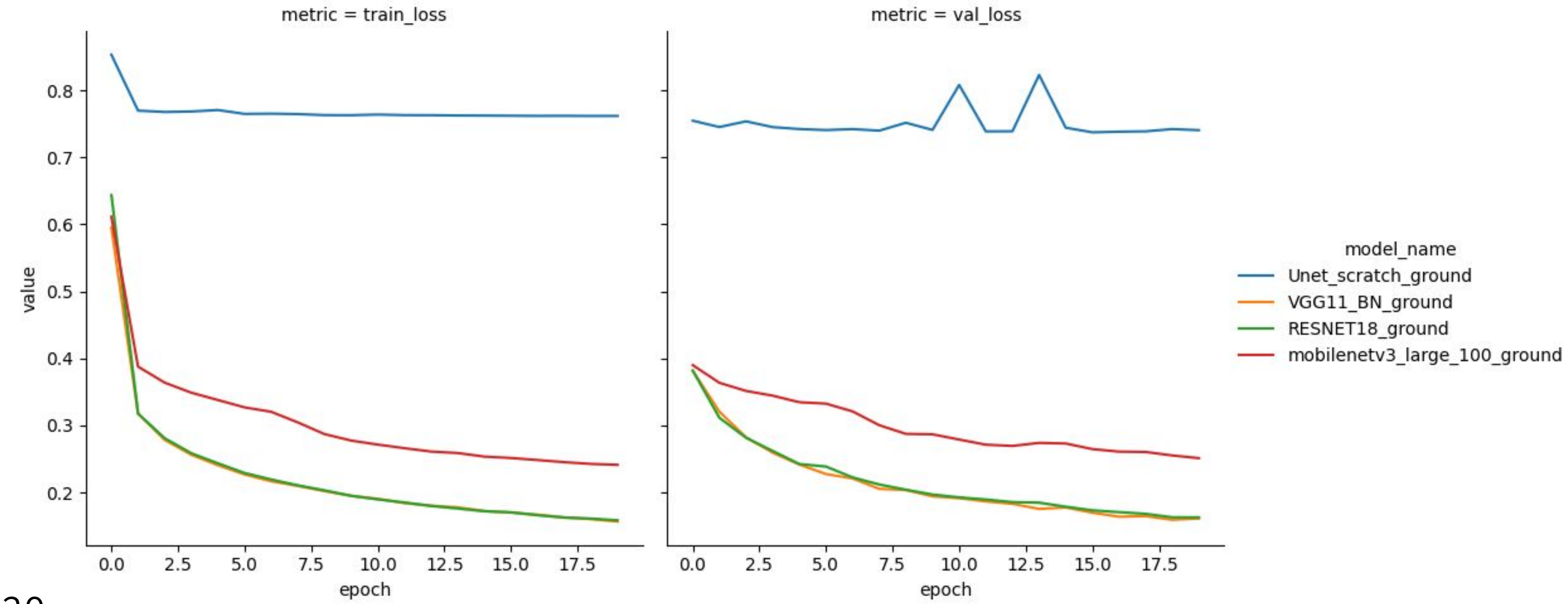


Figure 24: Pre-trained models training curves - loss.

Epochs - 20
Optimizer - SGD
Loss - Cross Entropy Loss

Model Training: Pre-trained U-Nets

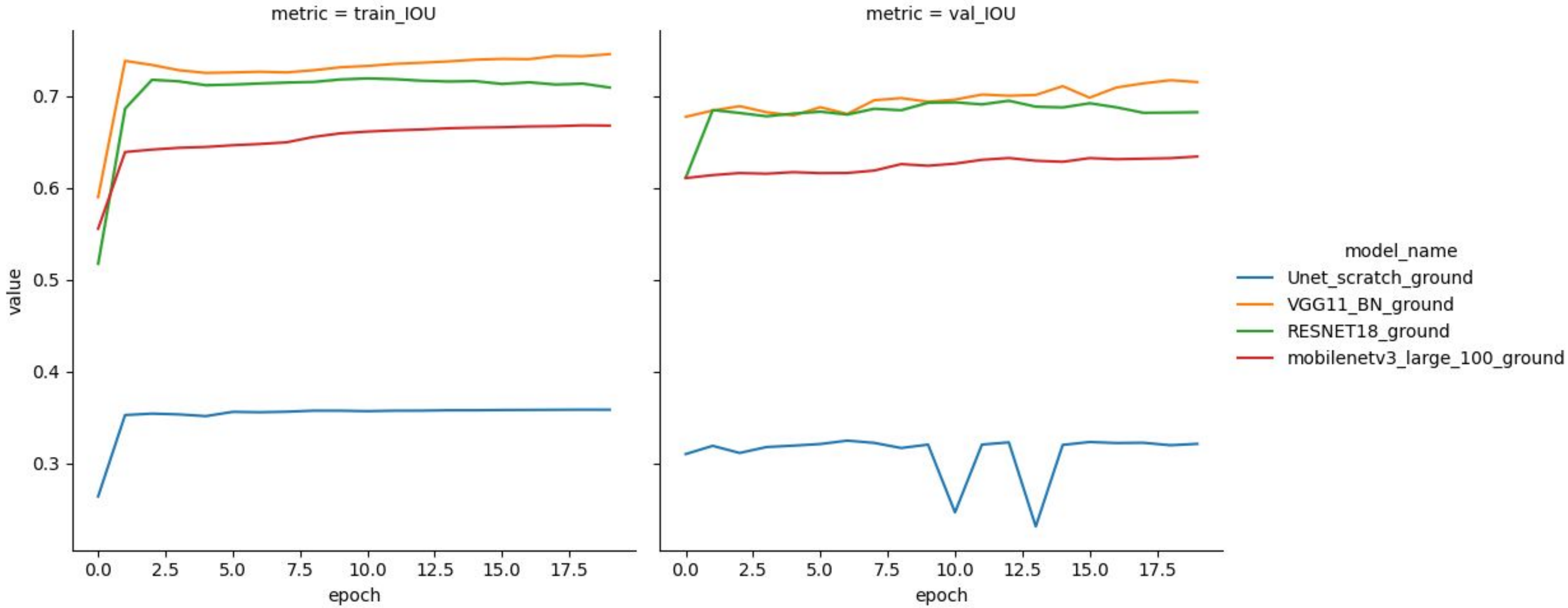


Figure 25: Custom U-Net training curves - IOU.

Model Results: Model Testing

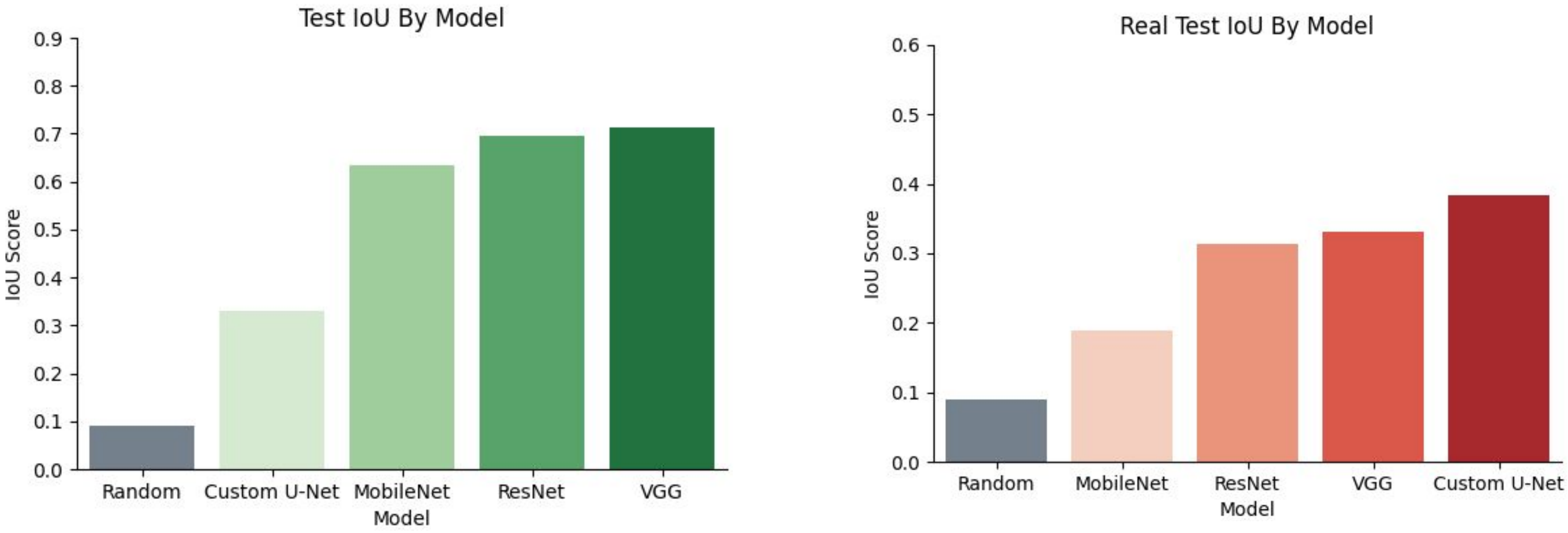


Figure 26: Testing evaluation comparison.

Model Results: Example Rendered Segmentations

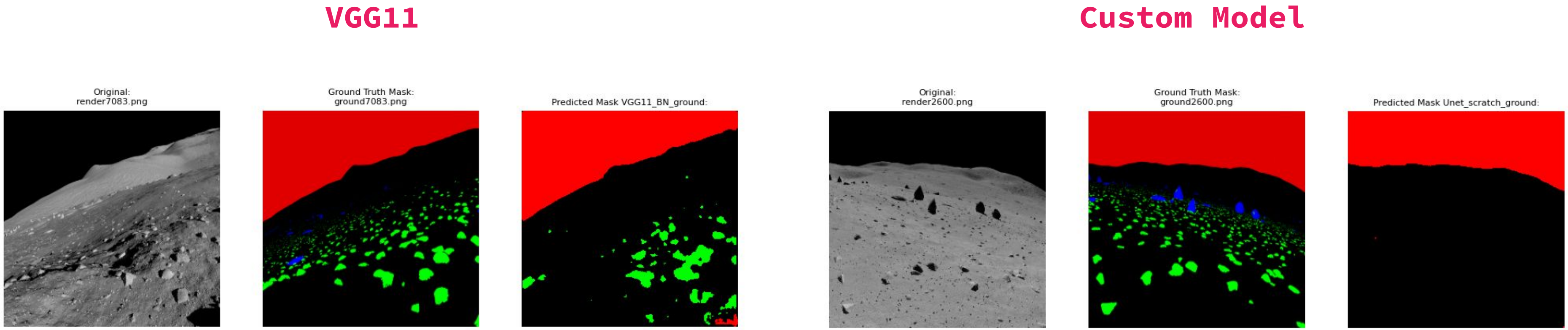
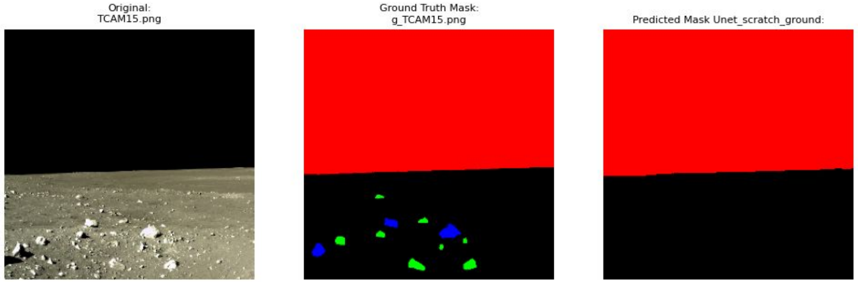


Figure 27: VGG11 vs custom U-Net testing results.

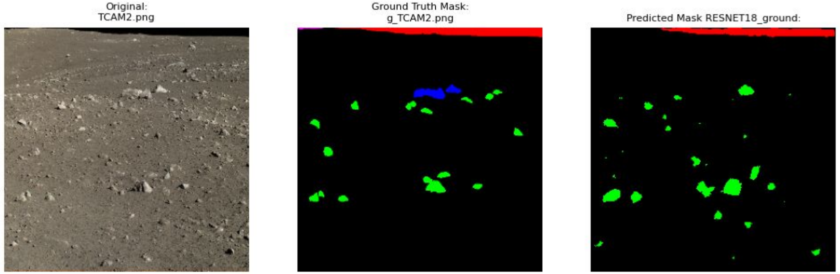
Model Results: Example Real Image Segmentation

Custom Model



IOU: 0.48

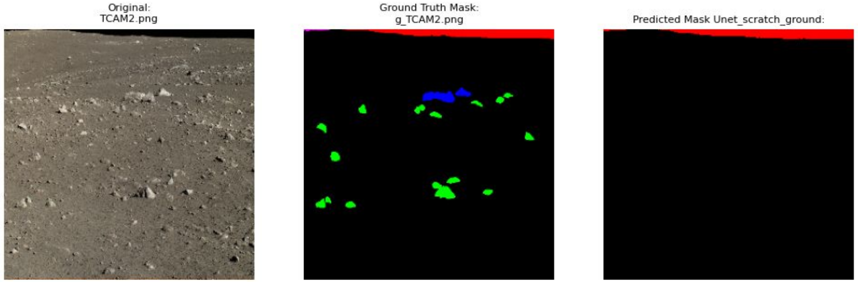
ResNet



IOU: 0.46

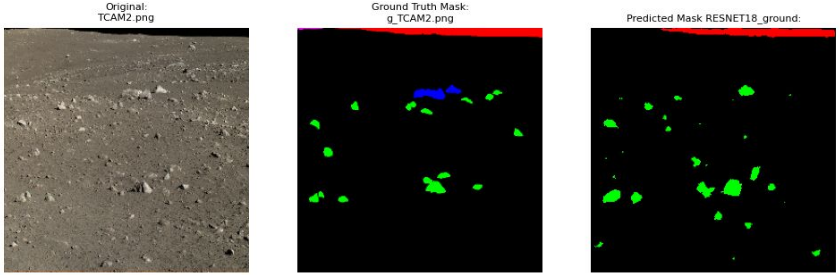
Model Results: Example Real Image Segmentation

Custom Model



IOU: 0.44

ResNet



IOU: 0.46

Figure 28: ResNet vs custom U-Net testing results.

Overview of Project Lifecycle

Interpreting Results

- The **VGG11 Model** was the strongest model we trained
  - This is likely due to being the largest model
- Our custom model underfit the dataset
- None of the models performed well on the real lunar images

Final Words and Areas of Improvement

- We succeeded in **comparing multiple different pre-trained U-Nets**
    - Although the conclusion we came to was consistent with previous attempts to solve this problem
- In the future:
    - Run full hyperparameter grid-search
    - Attempt adding attention to the models
    - Increase the complexity of the scratch model to attempt and compete with the pre-trained models

Conclusion

We are **not** getting hired by NASA.

QUESTIONS?

References

1. Jonathan Long et. al (2014) - Fully Convolutional Networks for Semantic Segmentation
2. Ronneberger et. al (2015) - UNet: Convolutional Networks for Biomedical Image Segmentation
3. Artificial Lunar Landscape Dataset on Kaggle
4. Lunar Surface Image - thespaceacademy.org
5. An Overview of Semantic Segmentation
6. Stanford CS231: Detection and Segmentation
7. Kaggle - Artificial Lunar Landscape Dataset
8. Kaggle - Artificial Lunar Landscape Dataset - Silver Notebook
9. Jaccard Index
10. Understanding and Visualizing ResNets
11. Architecture and Implementation of VGG16
12. MobileNet v3
13. Metrics to Evaluate Semantic Segmentation
14. Cross Entropy Loss