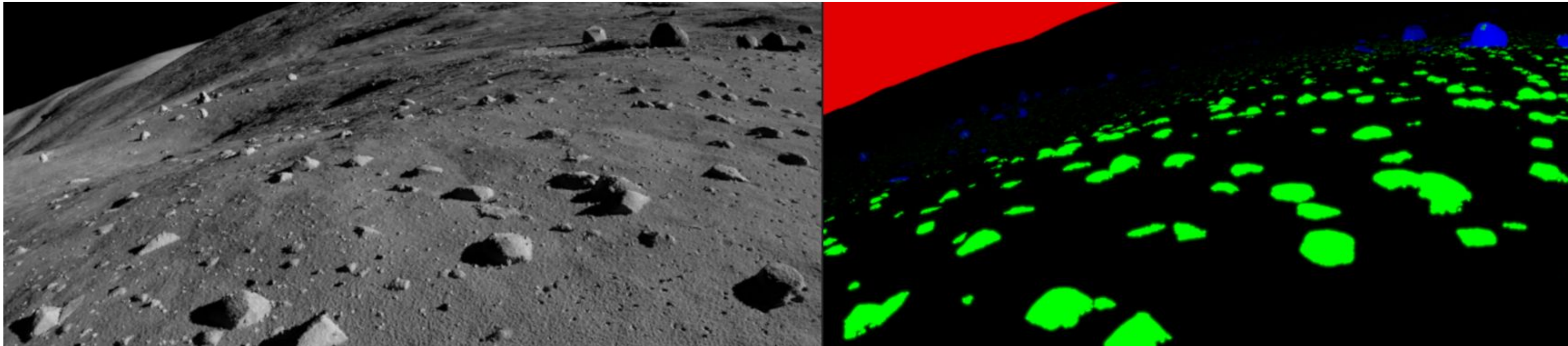
Multiclass Lunar Surface Segmentation Using U-Net in PyTorch

By:Ary Naim UNM anaim@unm.edu
Advisor: Dr Marios Pattichis



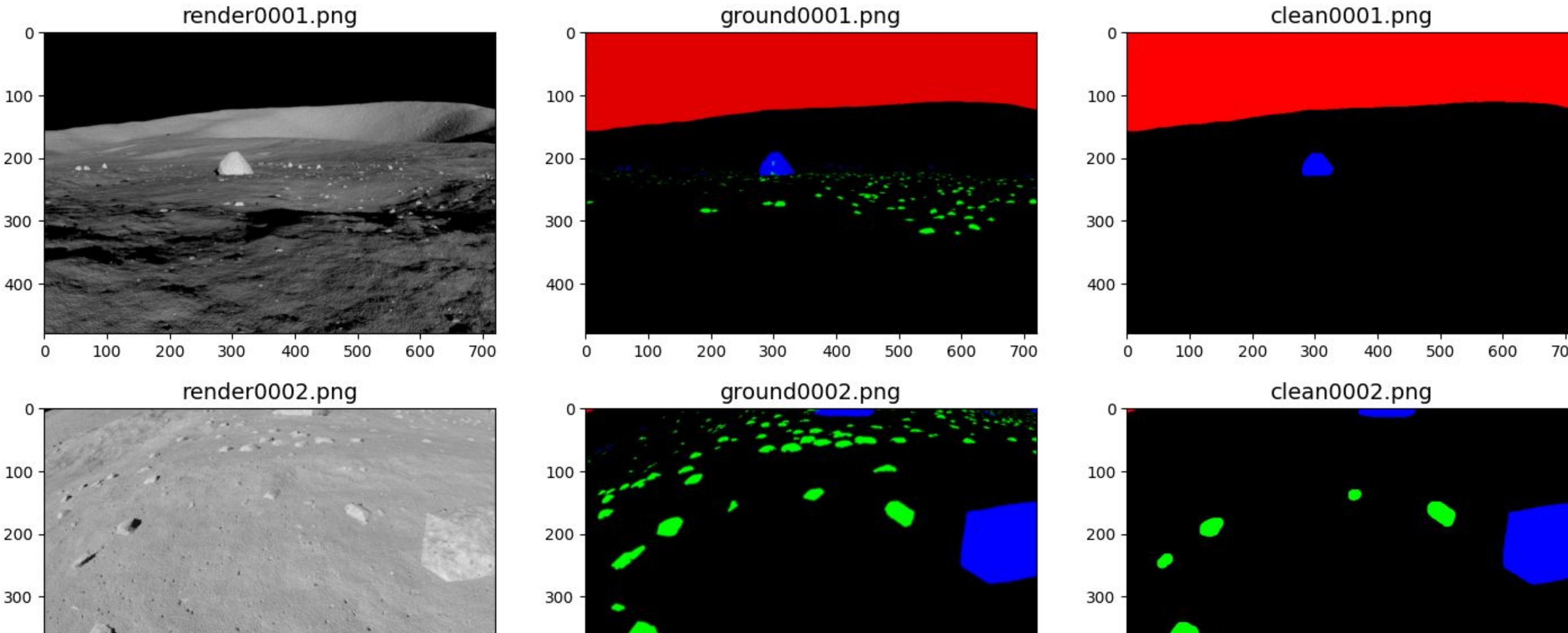
Problem

Earth’s Moon remains a frontier for exploration and a stepping stone for planetary exploration. Due to lack of atmosphere, distance from Earth’s resources, and infrastructure autonomous operations are an viable solution to build lunar infrastructure. I propose to implement the U-Net-only image segmentation lunar landscape segmentation for rock detection. The goal is to determine if U-Net is a suitable based on performance metrics such as (Dice an/or IoU).

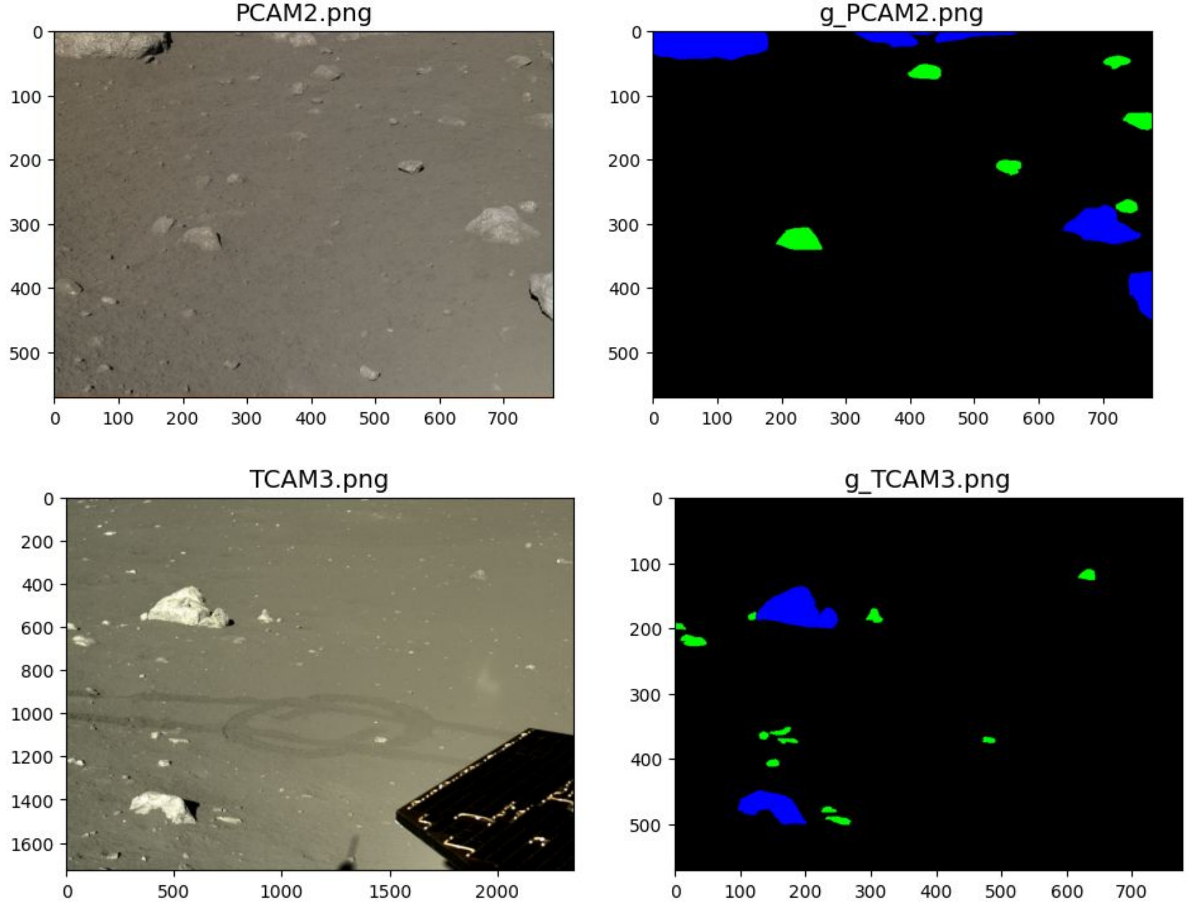
Refer to Github: https://github.com/naimaryan1/LunarMUnet

Data

There is a lack of real labeled lunar surface images with rocks. Researchers have focused on generating high-fidelity lunar surface images such as the widely used "artificial-lunar-rocky-landscape-dataset" created by Romain Pessia and Genya Ishigami of the Space Robotics Group, Keio University. Which is available at
https://www.kaggle.com/datasets/romainpessia/artificial-lunar-rocky-landscape-dataset [5]
Synthetic Data set:



Real Lunar Images



Run sections 1-3 then 6 in the Collab Notebook to see these visualizations.

Optional: K-Means Clustering Synthetic Images Results

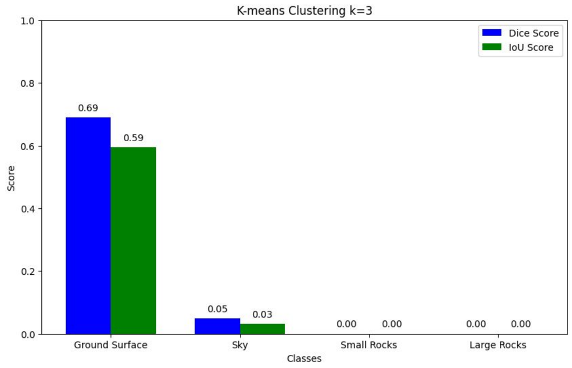
K-means clustering with k values from 3 to 8 showed minimal score variation. While k=4 aligns with the four-class structure, k=5 yielded the highest overall Dice & IoU values.

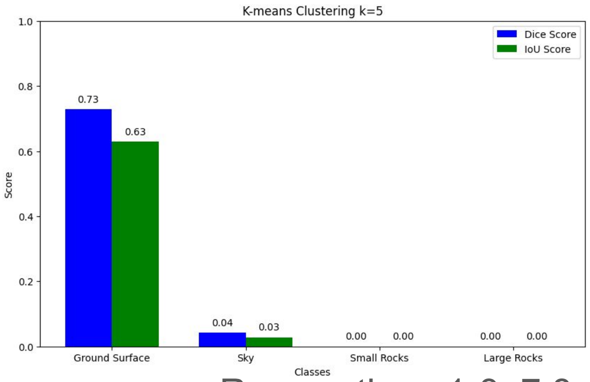
Initial analysis:
Best Overall (All Classes) → **k=5**, Mean Dice = **0.7285**, Mean IoU = **0.6295**

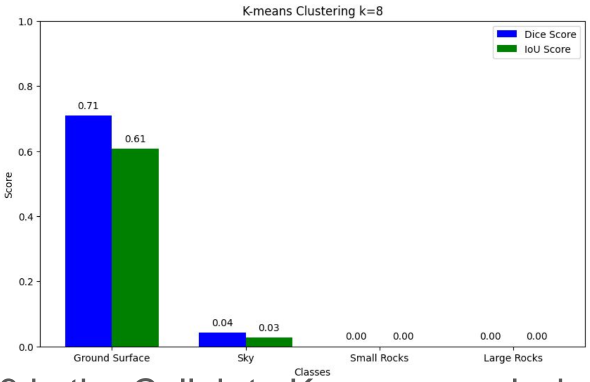
However, class importance is application specific. For surface obstacle avoidance, ground segmentation is more critical than sky. Re-evaluating with this priority confirmed the same optimal k.

Refined analysis (Ground Focused):
Best Surface Objects (No Sky) → **k=5**, Mean Dice = **0.7285**, Mean IoU = **0.6295**

Prioritizing ground segments reaffirmed that **k=5** is the optimal cluster count.



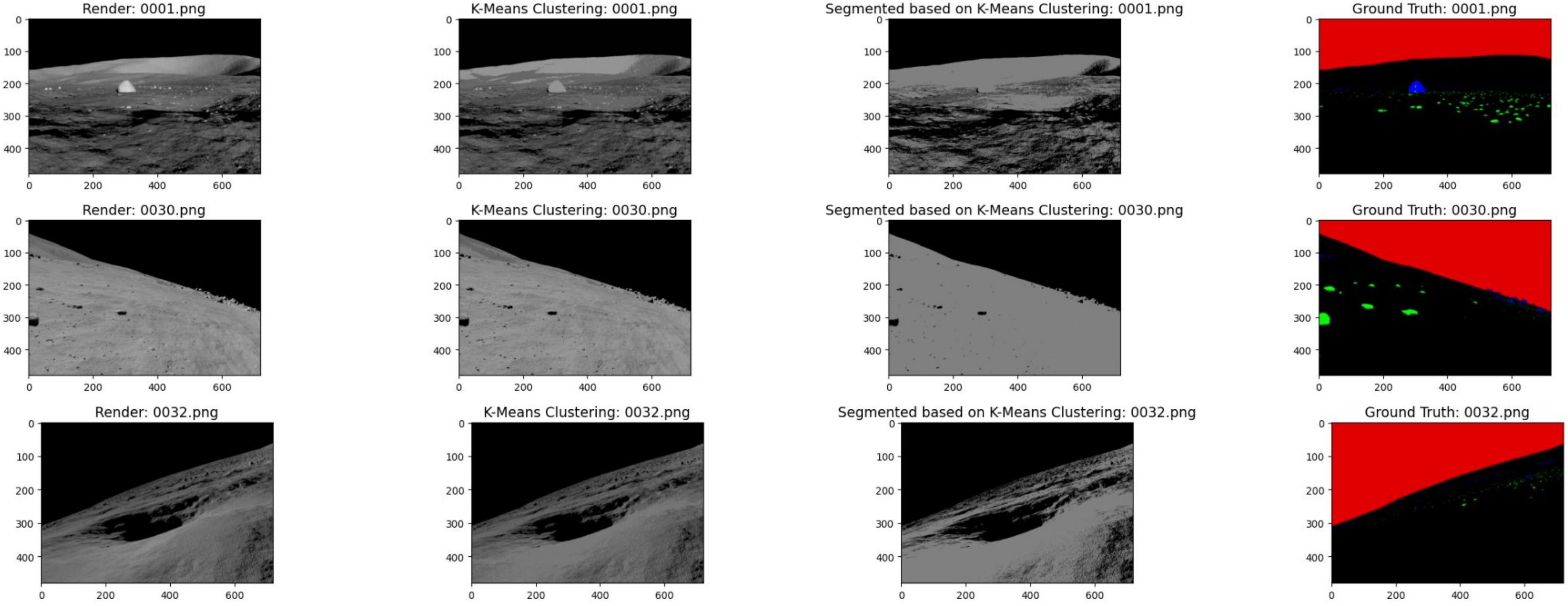




Run sections 1-3 ,7,8, 9 in the Collab to K-means analysis

Lets segment based on k-means

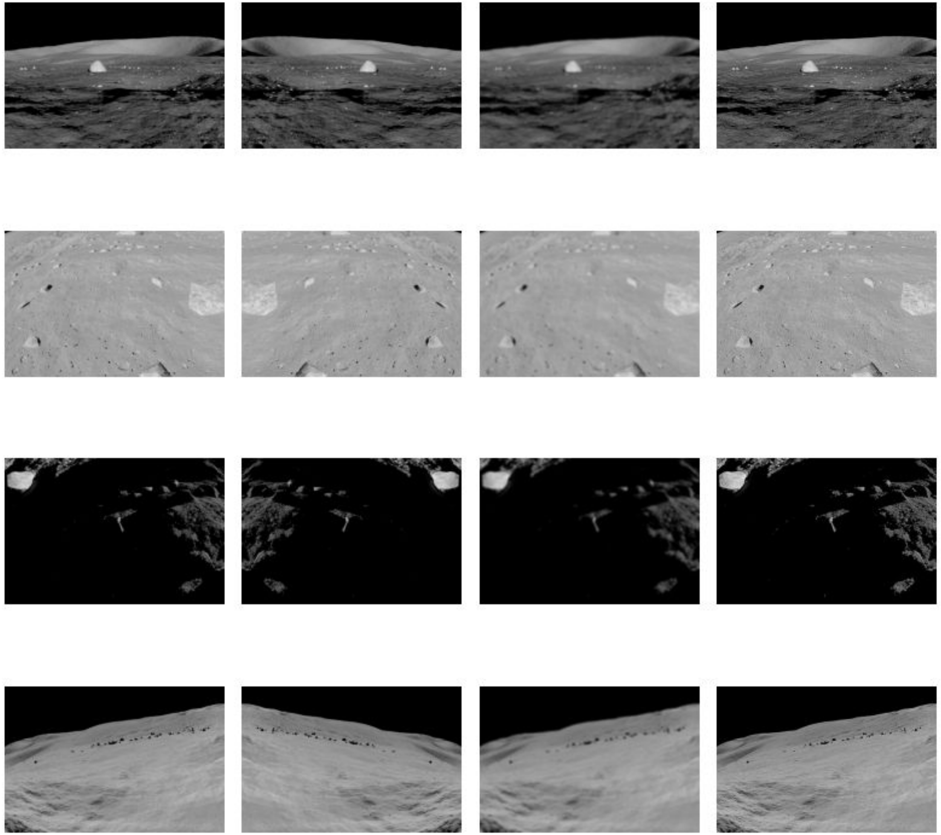
- Only ground surface Dice & IoU scores showed promise
- We k-means clustering alone can only detect the ground, so if we see ground we show else turn everything else off.
- Inference time was slow.
- Since we lack colors shadows were important & indicate areas to be avoided.



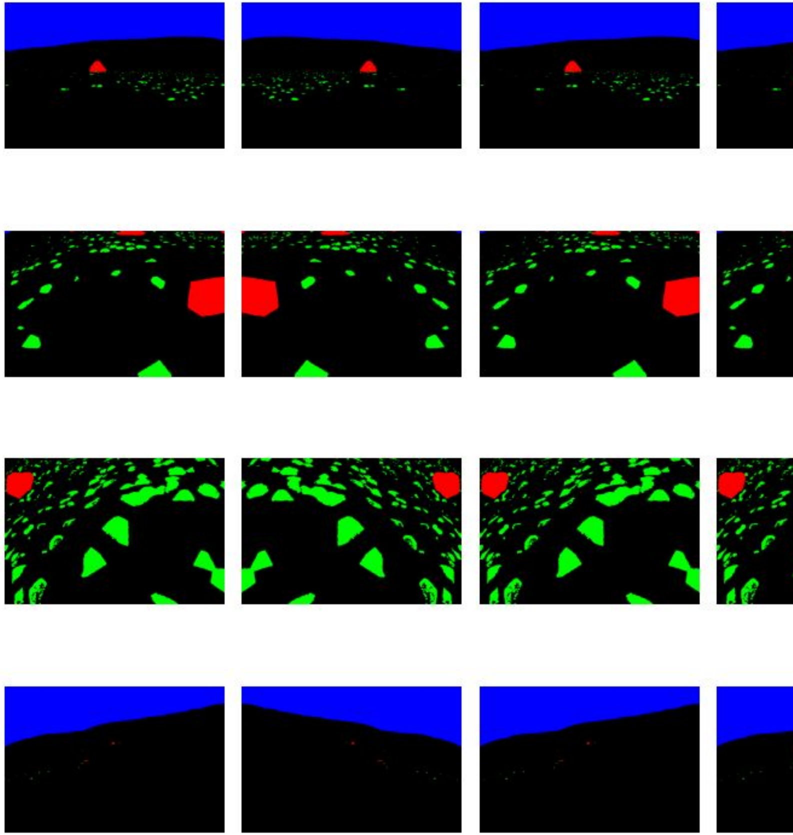
Run sections 1-3 ,7,8, 9 in the Collab

Basic Augmentation

Augmentations of the image:



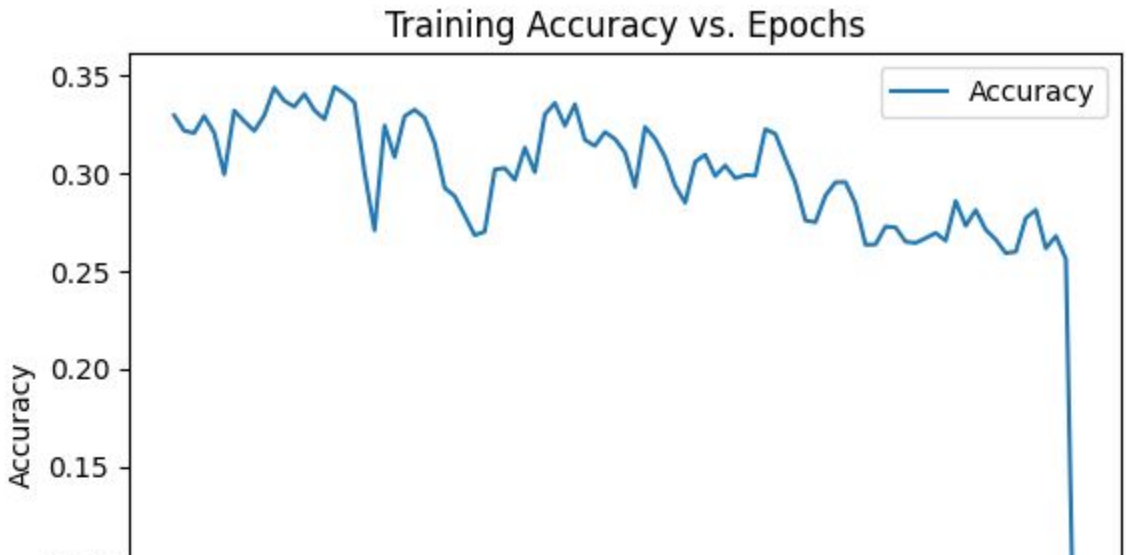
Corresponding masks:



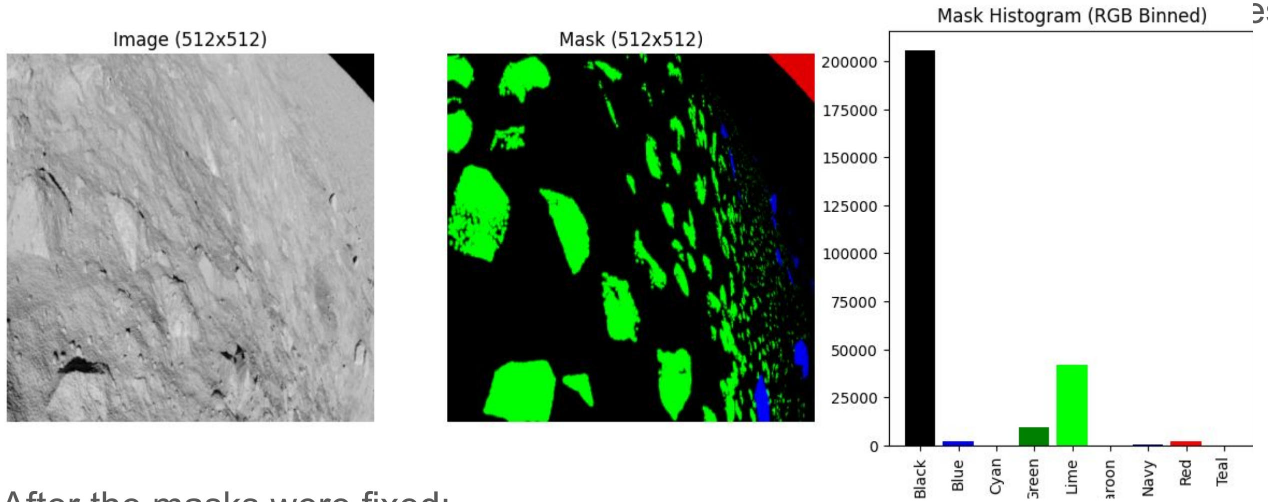
Initial Unet Run & Issues (Simple Augmentations)

Initial Data Issues. Unusual learning curve:

1) Mask size did not match input image (my mistake)
2) Mask had more than the 4 expected colors=classes.

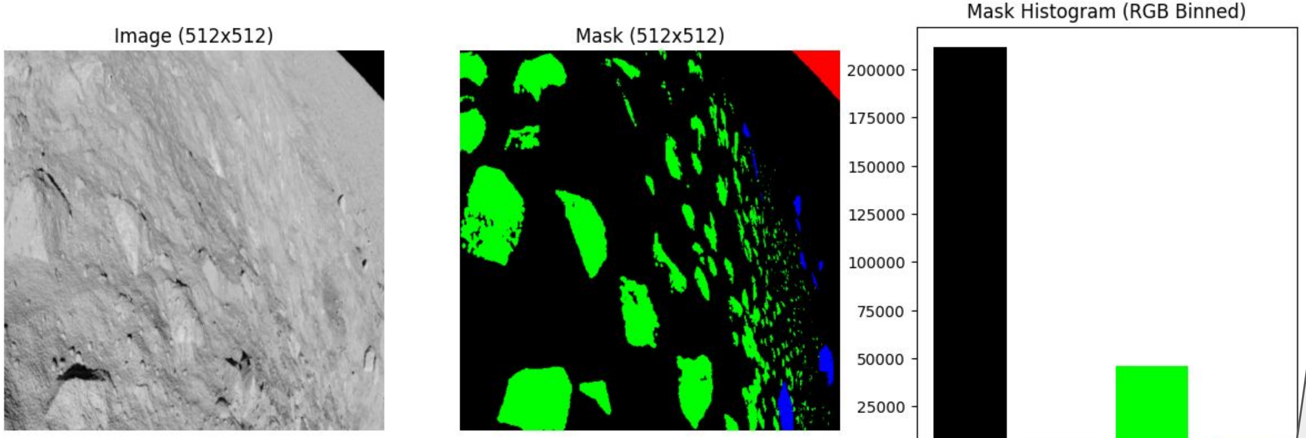


Investigation & Fix



There are surprisingly 9 colors or masks!
There should only be 4.

After the masks were fixed:

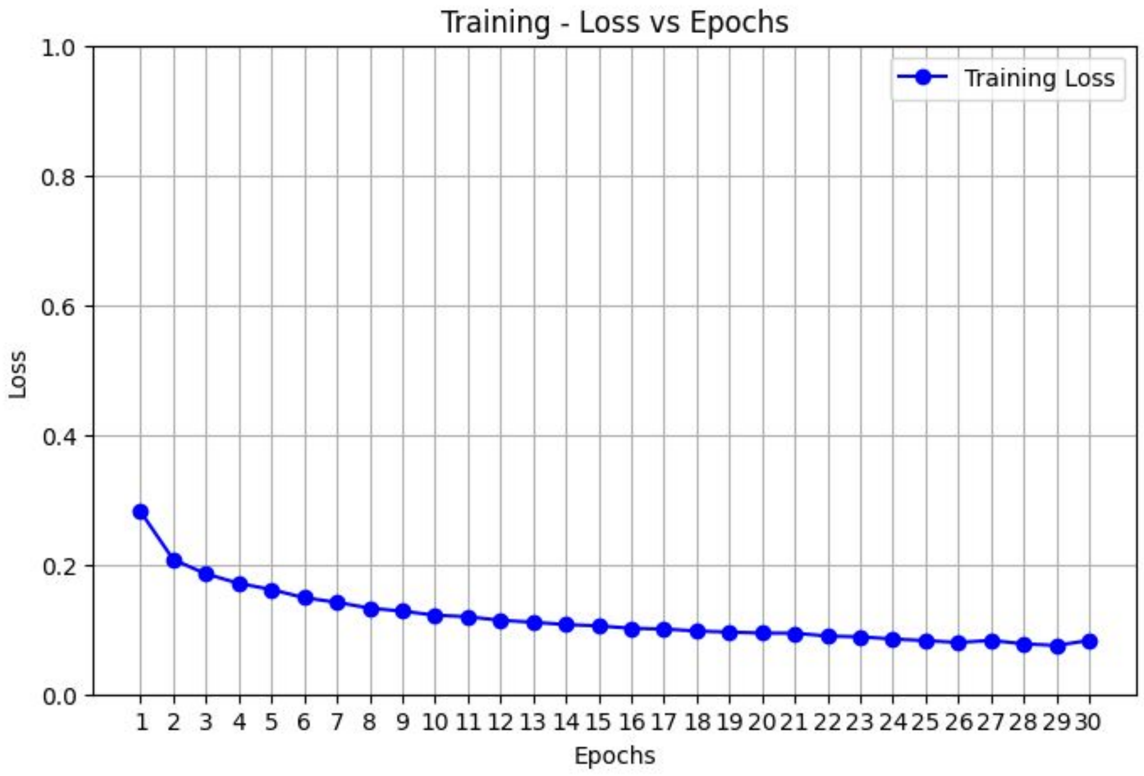


We processed that training data to return 4 masks.

Re-running UNet for 30 epochs

Much better Cross Entropy Loss! (`Loss: 0.2815 to Epoch 30 completed! Loss: 0.0830)`:

That's <u>350%+</u> loss in 30 epochs & sincere the goal is to minimize the loss then we are converging towards a better model very quickly.

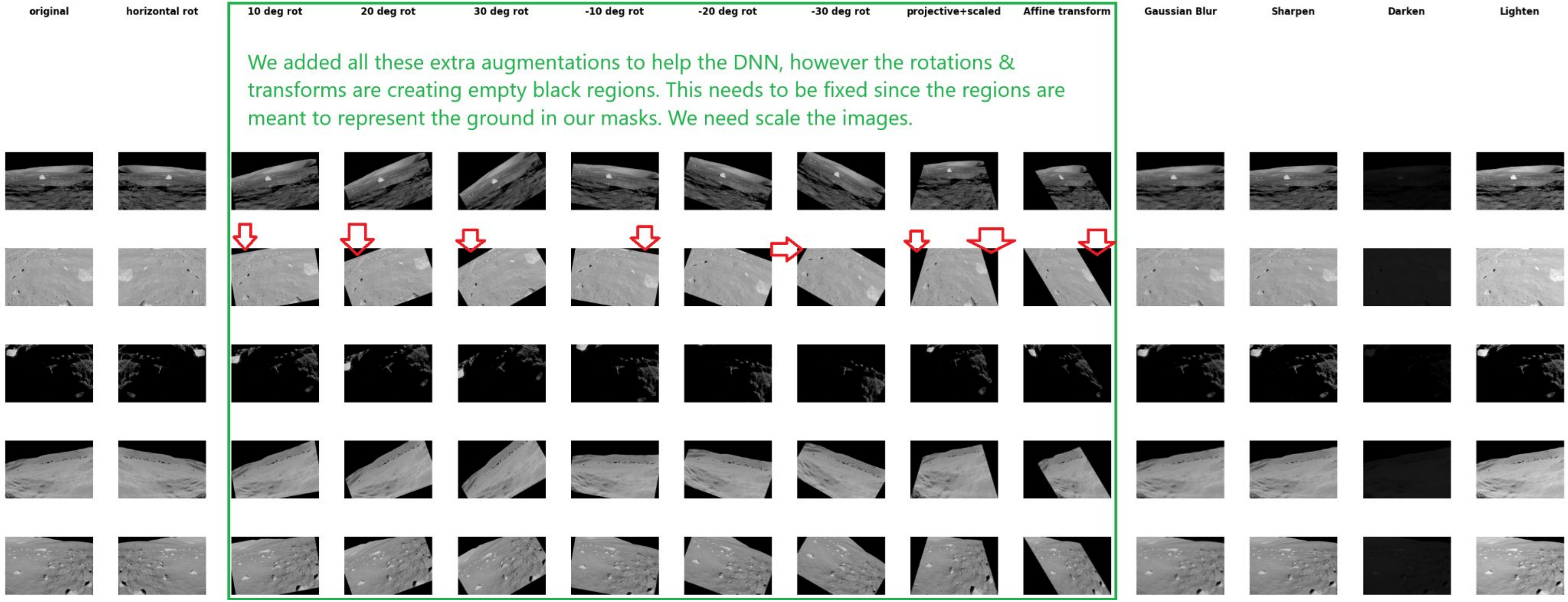


Run sections 1-3,8,10 in the Collab Notebook to train the model.

Let’s add more augmentations

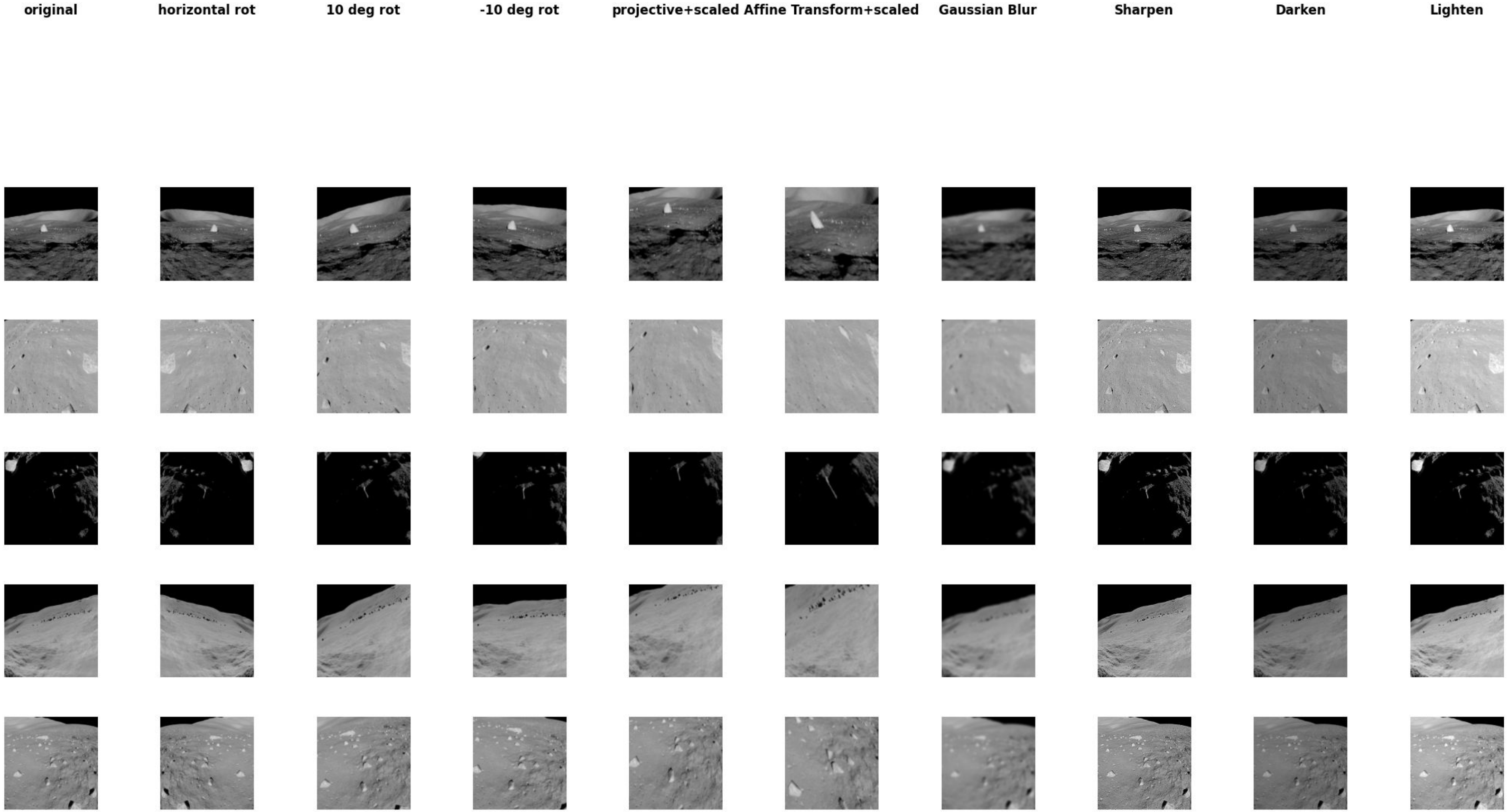
More Transforms

- We added 10-30 counter clock & clockwise including scaling factor to fix edges
- Project transform that made our original image look like isosceles parallelogram
- Affine transform
- Lighting



Augmentations scaled after transforms

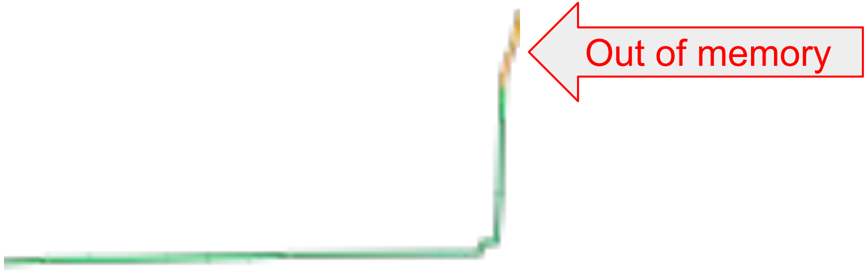
- Scaled the affine, projective, rotations unexpected artifacts at the edges



Run sections 1-3 then 6,7,8 in the Collab Notebook to see these visualizations.

Augmentations - Memory problem

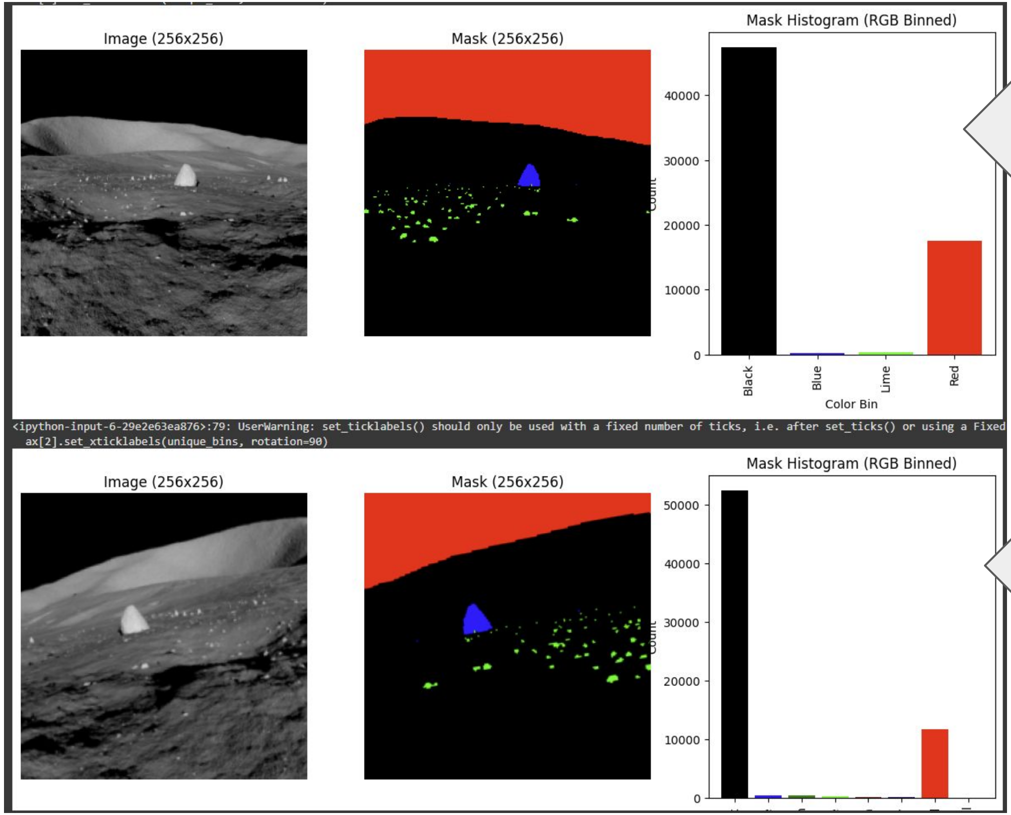
- We previously had about ~10k of data with 4 simple augmentations which means that after augmentations we had loaded ~40k images to RAM using about 45 GBytes of RAM which was 75% of our capacity.
- However after going from 4 augmentations to 14; the number of training images grew from ~10K to ~140K, which takes all our available memory in Collab which is ~60 GBytes & the notebook crashes:



- **Quick solution** just to test if the network is training after augmentation reduce the dataset size from 10k to 2k.
- **Long term solution** process augmentations in batches, write them to disk then, load then again as augmented training set.

Oh oh…extra classes are back!

- After affine, projective, and rotations extra classes are back. It seems some transforms create extra colors in opencv.

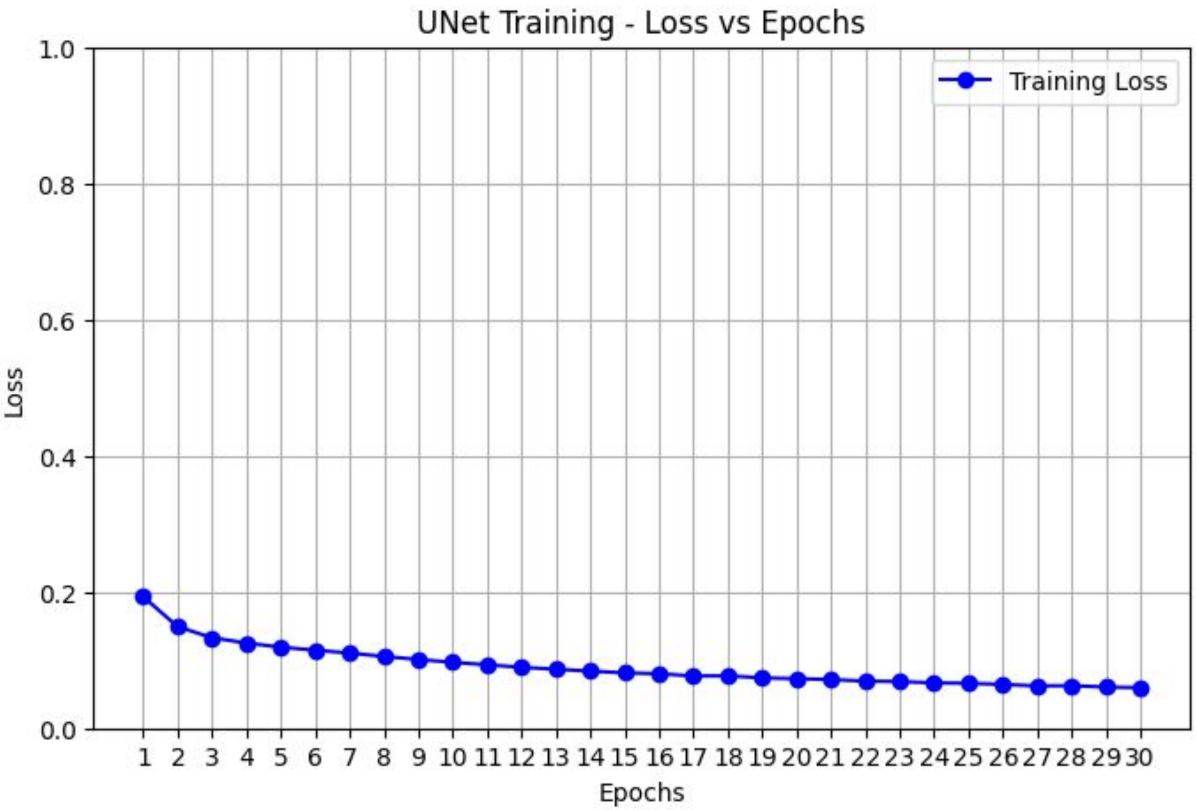


Flips are ok.

Rotations are not. So we fix them again.

Retrained U-Net again

- Retrained for 30 epochs after adding rotations, affine, and projective transforms & here are the results:
- Increased learning rate from 1e-4 to 3e-4,
- Epoch 1 completed! Loss: 0.1930 & Epoch 30/30....Epoch 30 completed! Loss: 0.0595



Run sections 1-3,8,10 in the Collab Notebook to train the model.

The Future - Real images & distortions

https://www.space.com/the-universe/moon/sunrise-on-the-moon-private-blue-ghost-lander-captures-amazing-shot-after-historic-lunar-touchdown-photo