# Finding Void Space in Rock Fractures

Ethan Mendel

## 1 Introduction

Climate-dependent fracturing is recognized as a critical process for weathering, erosion, and geochemical cycling. More data from different regions on different time/spatial scales is needed to improve our understanding of the process. Currently, data is processed manually, making breakthroughs very slow and tedious.
Samples are cross-sections of sandstone from the Mullins Glacier in Antarctica. Rocks from the glacier give a chronosequence of rocks that have varying amounts of weathering.

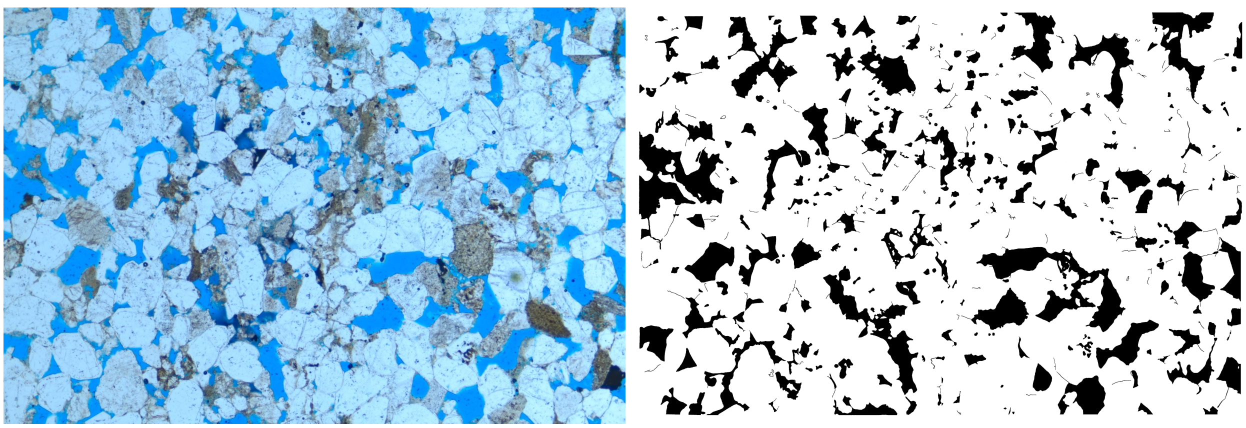


Figure 1.1: An example of a cross selection and its respective manually mapped image

## 2 Methods

A few methods were chosen to explore possible solutions to this problem.

### 2.1 Mean/Standard Deviation with Independent RGB Values

All void space looks to be around the same shade of blue. Additionally, this blue is vastly different than the other white/black splotches within the data.
The idea of this approach was to use the manually mapped images to build a distribution of the red, green, and blue color values identified as void space, then pick an acceptable standard deviation range that produces good results.
With a learned mean, standard deviation, and range, unseen void space could be classified based on the imputed red, green, and blue color values of any given pixel.

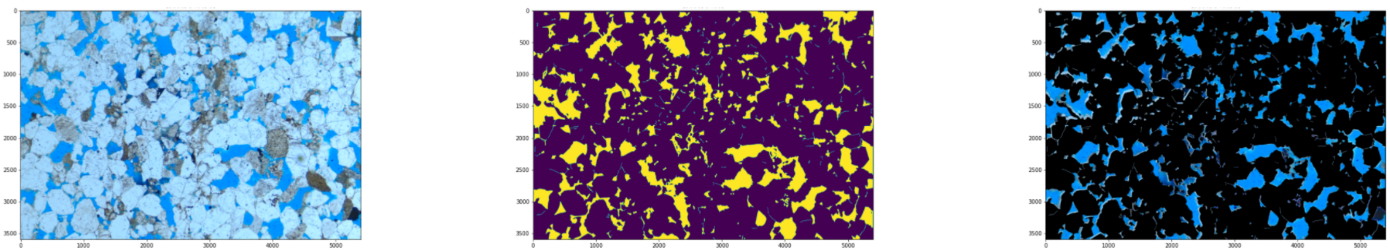


Figure 2.1.1: Isolating void space using matrix multiplication

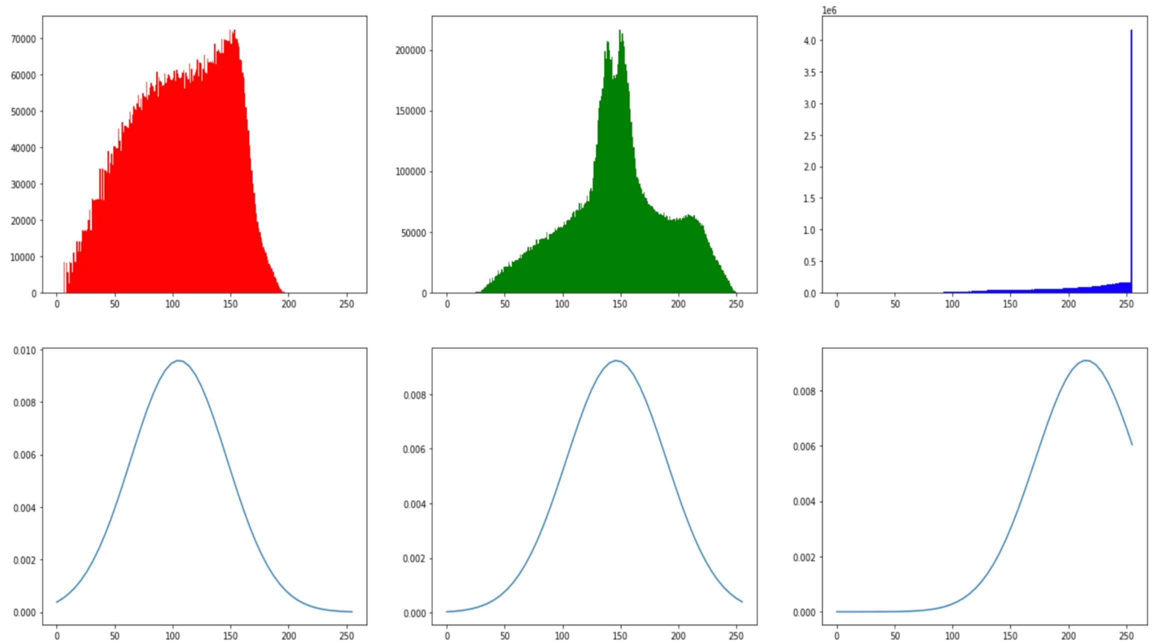
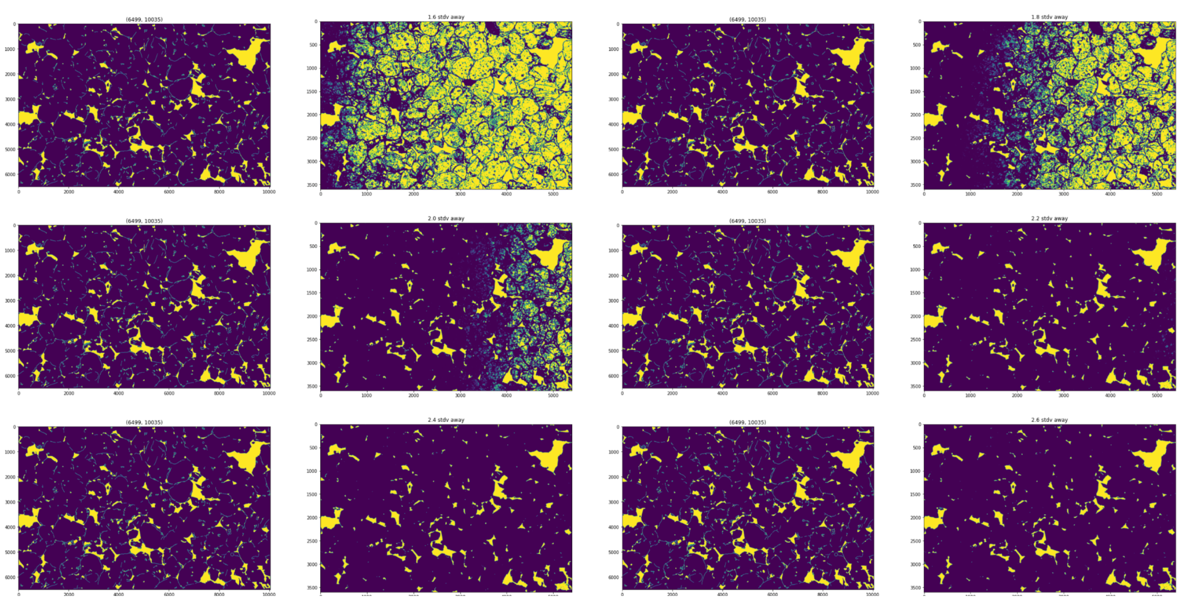


Figure 2.1.2: Red, green, and blue void space color value distributions



Figurer 2.1.3: Predictions based on different standard deviation range
NOTE: 1st and 3rd columns are expected, 2nd and 4th columns are predictions

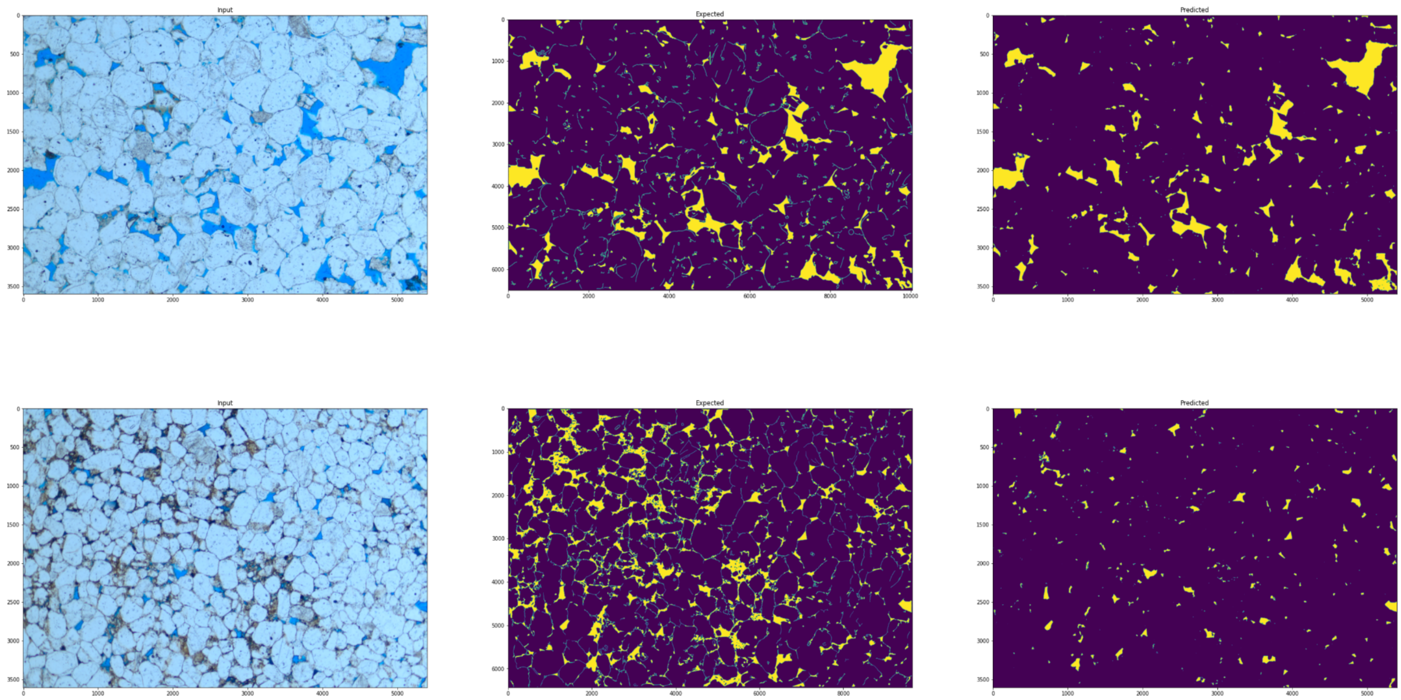



Figure 2.1.4: Input, Expected, Predicted

## 2.2 Mean/Standard Deviation with Dependent RGB Values

Initial findings with independent rgb values proved unpromising. This is when exploration of dependent rgb values started. The idea of this approach is that some colors are dependent on all color values being dependent on each other.

| | | | |
|---|---|---|---|
| | midnight blue | #191970 | (25,25,112) |
| | navy | #000080 | (0,0,128) |
| | dark blue | #00008B | (0,0,139) |
| | medium blue | #0000CD | (0,0,205) |
| | blue | #0000FF | (0,0,255) |
| | royal blue | #4169E1 | (65,105,225) |

Figure 2.2.1: Different shades of blue

NOTE: All blues would be within the standard deviation range, but are relatively different

With a dictionary of acceptable [R,G,B] values, unseen void space could be classified based on the imputed [R’,G’,B’] values of any given pixel.

In most iterations of running this code, ~30,000 different [R,G,B] combinations are found.

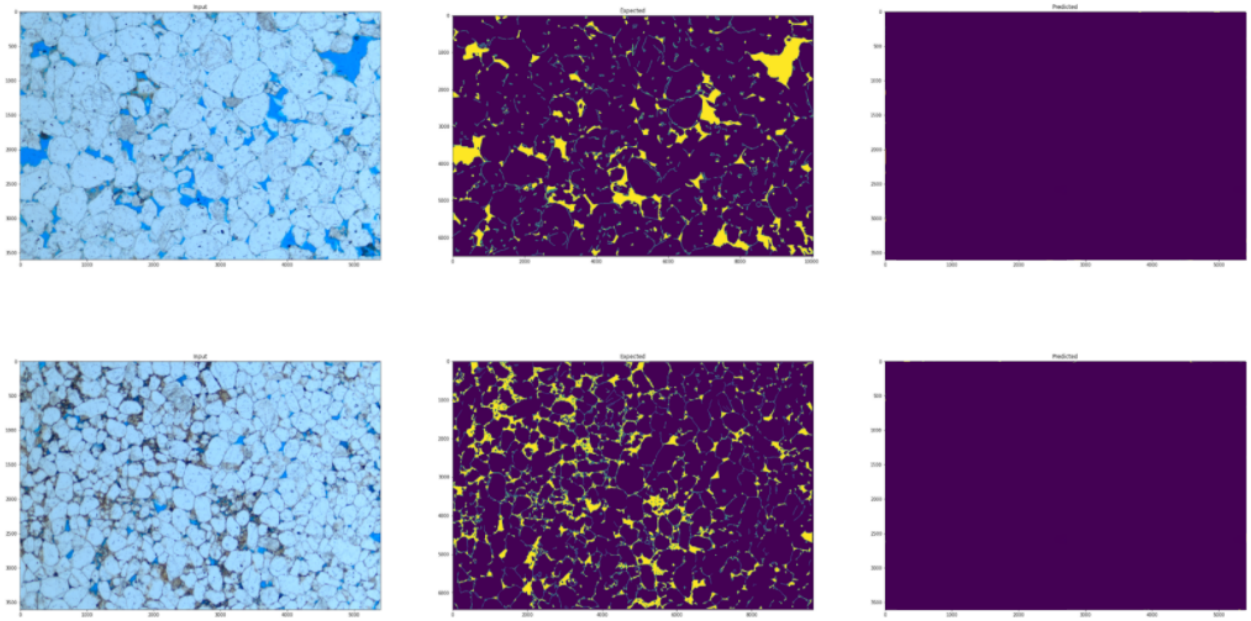


Figure 2.2.2: Input, Expected, Predicted

Even with 30,000 different [R,G,B] values, it seems there are no repeats. This might seem unlikely, but even with a constant blue value, there are still over 65,000 different combinations of only red and green values.

Because of this, it seems beneficial to check for similarities:

> Given value [r,g,b], check every combination of adding/subtracting from each:
>
> > [r,g,b],[r±1,g,b],[r,g±1,b],[r,g,b±1],[r±1,g±1,b],[r,g±1,b±1],[r±1,g,b±1],[r±1,g±1,b±1]
> >
> > ->15 checks per 1 value  difference.

```python
fig = figure(0, (48,50))

imgs = test_data[0]
orig = imread(imgs[0])
clas = convert_to_bin(plt.imread(imgs[1],0),invert=True)

orig = cv2.resize(orig,(orig.shape[1]//10,orig.shape[0]//10))
clas = cv2.resize(clas,(orig.shape[1],orig.shape[0]))
predicted_img = vf.predictImg(orig, update=True)

fig.add_subplot(1, 2, 1); title(clas.shape)
imshow(clas)
fig.add_subplot(1, 2, 2)
imshow(predicted_img)
clas = None
orig = None
predicted_img = None
```

11h 15m 2s    completed at 1:09 AM

Figure 2.2.3: Running for 11+ hours before disconnecting. 8.7 trillion iterations to predict 1 image. No further work on this approach because the runtime is not plausible

## 2.3 Convolution Neural Network (CNN)

The idea of this approach is to see if void space has detectable features. If it does, this also could spark new findings in the geology sector because that means there are similarities in the types of weathering at different points in the Earth's history.
Keras was used to make and train this model using binary cross entropy loss, and the mean intersection over union metric.

```
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
conv2d_2 (Conv2D)            (None, 356, 536, 10)      760
_________________________________________________________________
max_pooling2d_2 (MaxPooling2 (None, 118, 178, 10)      0
_________________________________________________________________
flatten_2 (Flatten)          (None, 210040)            0
_________________________________________________________________
dense_4 (Dense)              (None, 100)               21004100
_________________________________________________________________
dropout_2 (Dropout)          (None, 100)               0
_________________________________________________________________
dense_5 (Dense)              (None, 194400)            19634400
=================================================================
Total params: 40,639,260
Trainable params: 40,639,260
Non-trainable params: 0
_________________________________________________________________
None
```

Figure 2.3.1: cnn architecture

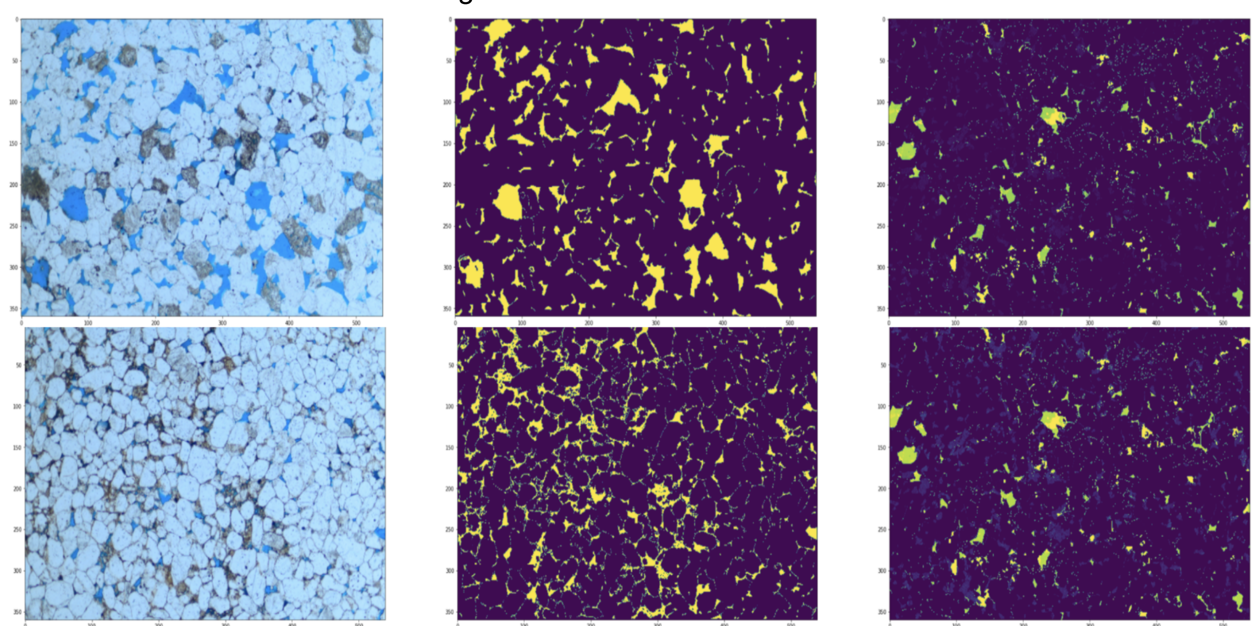


Figure 2.3.2: Input, Expected, Predicted; no validation set used

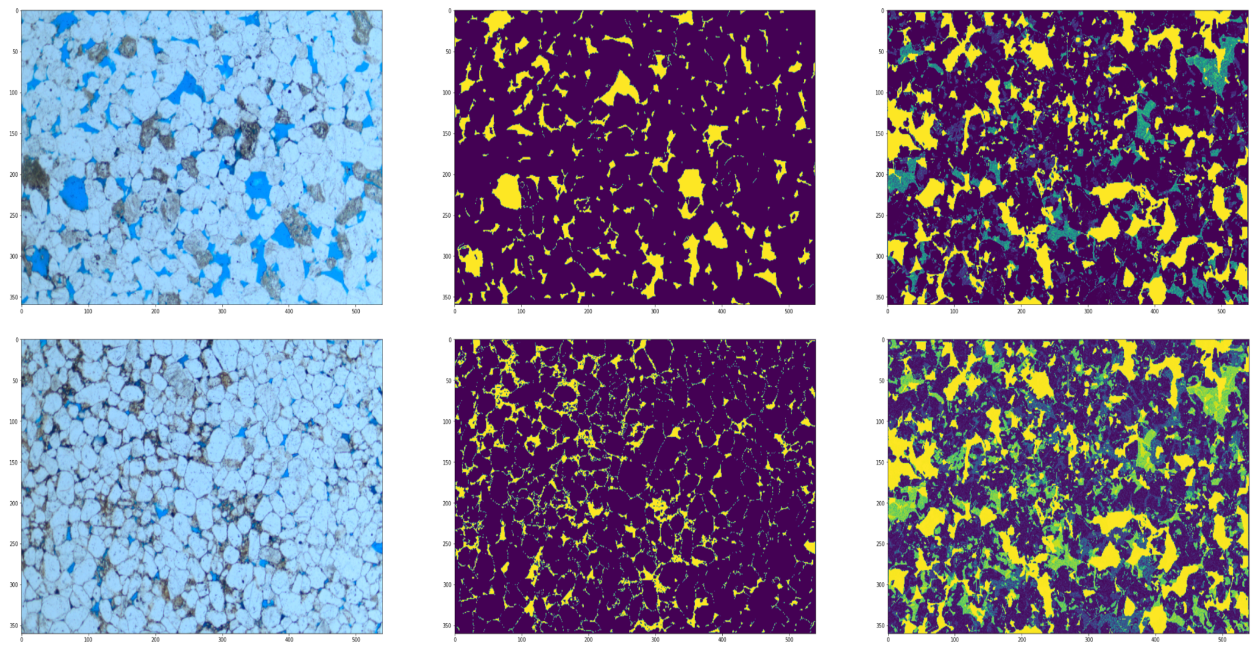


Figure 2.3.3: Input, Expected, Predicted; validation set used

*Both models were trained using 500 epochs.
It seems that predictions from the CNN look very similar. A validation set helps to diversify the predictions a bit, but the model is too biased towards the training data.

## 2.4 Semantic Segmentation

The idea of this approach is to segment different portions of the images as void or not void. Though this is a very attractive approach for a lot of areas of Computer Vision/Machine Learning, the level of complexity that comes with it is not required.
The desired model is searching for and identifying void space in rock fractures, producing binary output. If specific types of void space was desired, this method may prove more useful and worth exploring.

# 3 Struggles

## 3.1 Image Size

Images were gigantic and inconsistent. All input images were 3600x5400, but expected outputs varied from 3600x5400 to 10,000x15,000. Expected outputs were also saved as .tif files, which didn't allow some files to be loaded, shrinking the dataset size.
Resizing expected outputs was not too much trouble, but analyzing pixel values at that scale really slowed down development. Building the Mean/Standard Deviation model with Dependent RGB values wasn't even possible because of the processing time required for it.

Reducing image sizes was an option, but still requires 87 billion checks for dependent RGB values at 10% of the original input size. Additionally, up to 90% of valuable data could be lost with a reduction this large, which defeats the purpose of the project in the first place.

## 3.2 Dataset Size

Only 13 image pairs were attached to this project. This small number of data coupled with the aforementioned reading issues left 8 solid image pairs to work with. After removing two for a testing set, only six remained for training/validation purposes.

# 4 Final Steps and Results

The Mean and Standard Deviation of Independent RGB values seems to be the best approach. After running the exploration code a few times, its predictions consistently look the most similar to the expected outputs.

## Intersection Over Union

No real metric was needed when experimenting with the different approaches, though the Keras CNN model did require one to facilitate learning. Being that finding void space is similar to masking a background, comparing this project's predictions to expectations resembled the CNN assignment from this semester. In that assignment we used IoU, so it was used here as well.

## K-Fold Validation

Because of the insignificant amount of training data, k-fold validation is a no brainer for a final model. This helps the model reduce bias by evaluating itself using different training data each time, then picking the model that yields the best validation IoU.

## Standard Deviation Fitting

Mean and standard deviation are pretty straight forward parameters to learn, but as they change, the best standard deviation range changes as well. In each validation round, a different range will be found that maximizes the training set's IoU.

# 5 Conclusion

The final model was trained using 3-fold validation. After Isolating void space and removing zeros, independent red, green, and blue means and standard deviations were found. Next predictions were made for all training images with different standard deviation ranges. The average IoUs were saved and the best corresponding range was chosen. Lastly the validation images were predicted, and the average IoU was saved with the model's information to return to.

After 3 iterations, the best validation IoU was found, that model's information was loaded, and the testing data was predicted.

```
training with 3-fold validation..
        validation 0
                training   size: 4
                validation size: 2
training..
        isolating void space..
        void space isolated
        removing zeros..
        zeros removed
        getting means and standard deviations..
        found means and standard deviations
        getting best range..
        2.8 was over best range
        got best range: 2.6
training complete!
                validating..
        validation round 0 complete!
                range:2.6
                trainingIOU:0.21019456652274152
                validateIOU:0.2525157593612215
        validation 1
                training   size: 4
                validation size: 2
training..
        isolating void space..
        void space isolated
        removing zeros..
        zeros removed
        getting means and standard deviations..
        found means and standard deviations
        getting best range..
        2.8 was over best range
        got best range: 2.6
training complete!
                validating..
        validation round 1 complete!
                range:2.6
                trainingIOU:0.20689310160144547
                validateIOU:0.2610181999953311
        validation 2
                training   size: 5
                validation size: 1
training..
        isolating void space..
        void space isolated
        removing zeros..
        zeros removed
        getting means and standard deviations..
        found means and standard deviations
        getting best range..
        2.4 was over best range
        got best range: 2.2
training complete!
                validating..
        validation round 2 complete!
                range:2.2
                trainingIOU:0.21971021890979012
                validateIOU:0.3665355154857852
validation complete!
getting best model..
best model selected
        range:2.2
        trainingIOU:0.21971021890979012
        validateIOU:0.3665355154857852
```

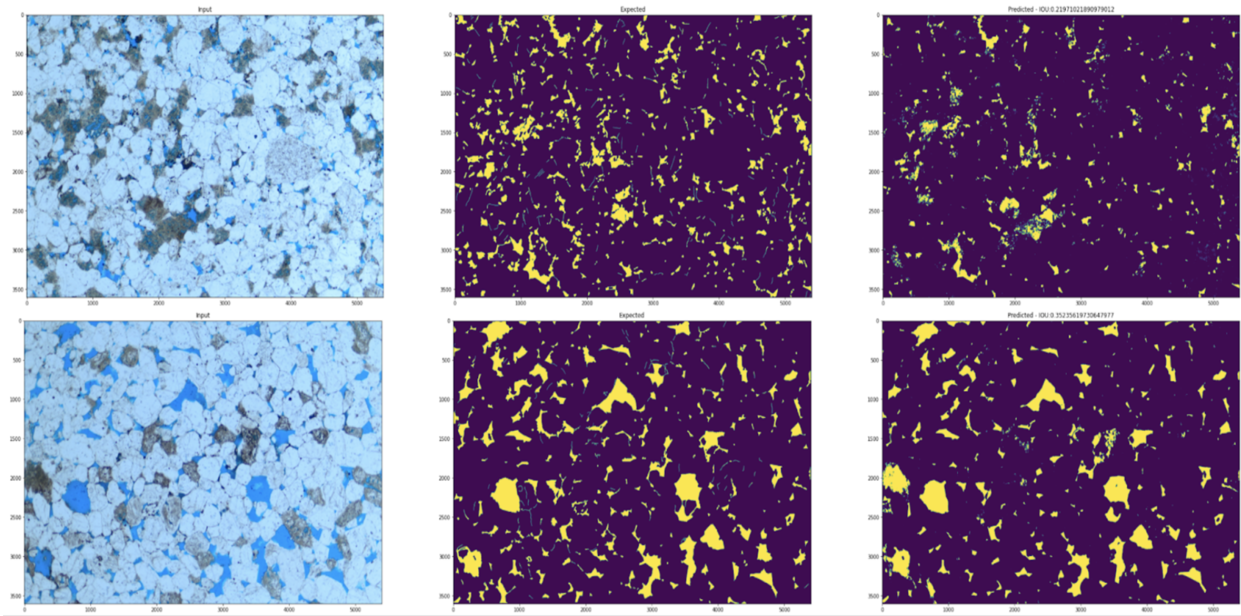



Figure 5.1: training output/updates

Figure 5.2: Input, Expected, Predicted

# 6 Links

Project Introduction - https://drive.google.com/file/d/1IdCPDMsKSTwdYxMfMhU6Q25cjA7Looo4/view
Github Repository - https://github.com/EthanMendel/RockFractures
Project Data - https://drive.google.com/file/d/1Xa7i6XsDcZ2-vRLZe02Q4a5E4a28qZG8/view