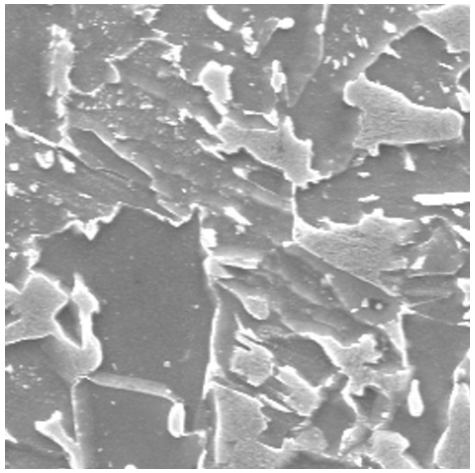
# Image Segmentation

## Progress Report

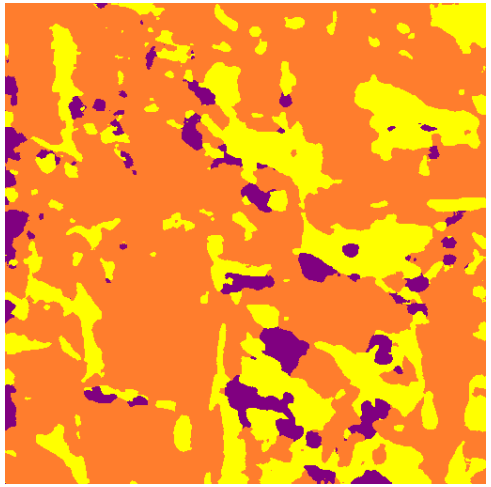
Bishal

# Model Output



Input Test Image

Input Label Image



Output Test Image

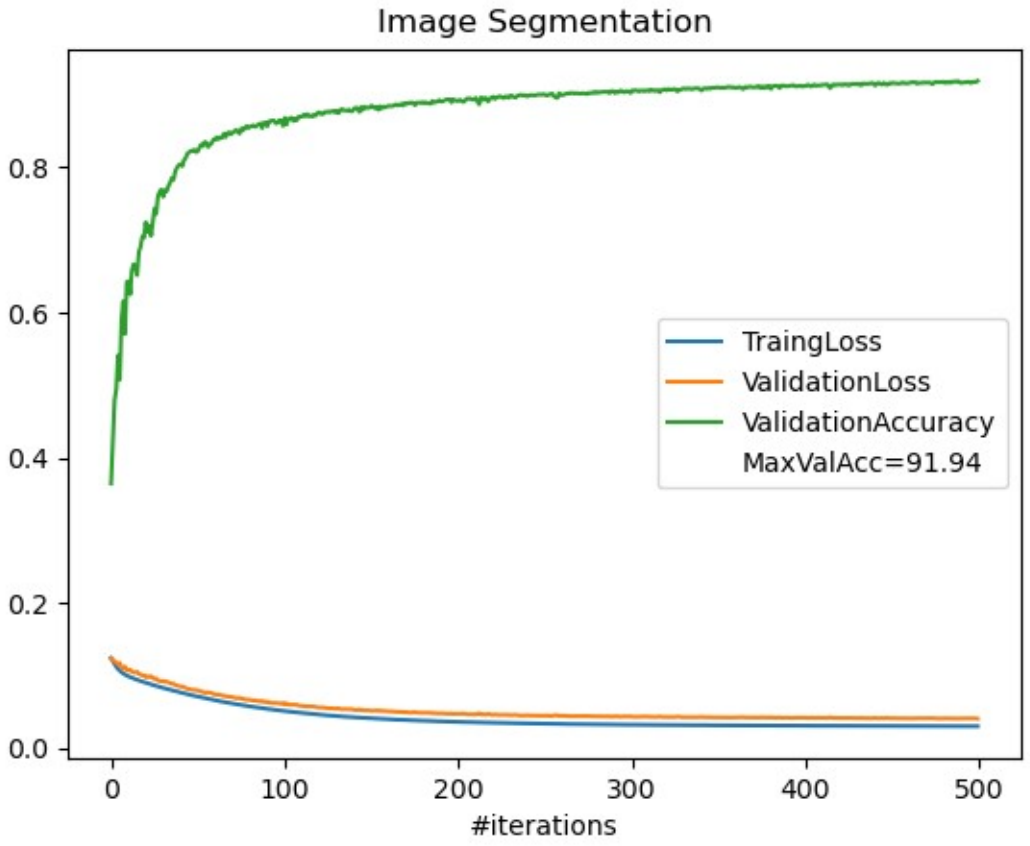
Validation mIOU → 0.4499319914225048 4

Validation Acc → 91.942

Dice Coefficient(Validation) → 0.8195625888211245

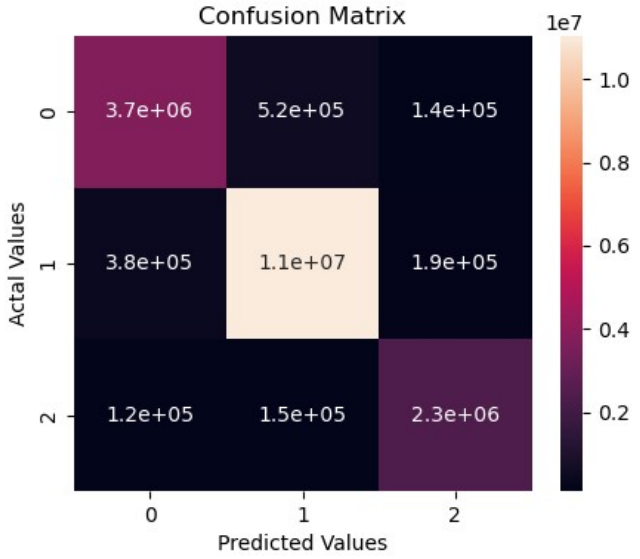
Dice Coefficient(Test) → 0.6842116581391179

# Model Metrics



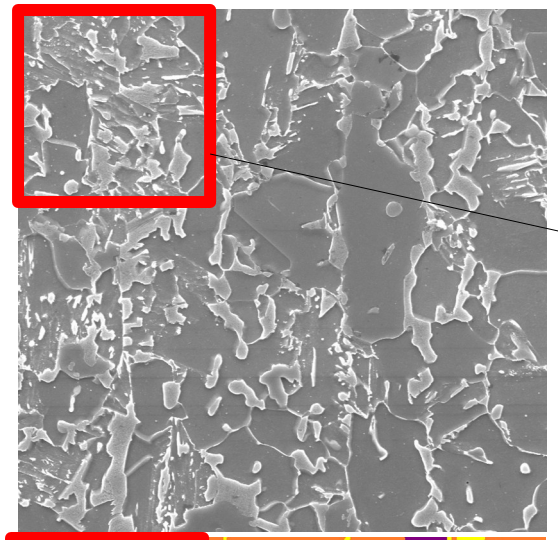

Confusion Matrix

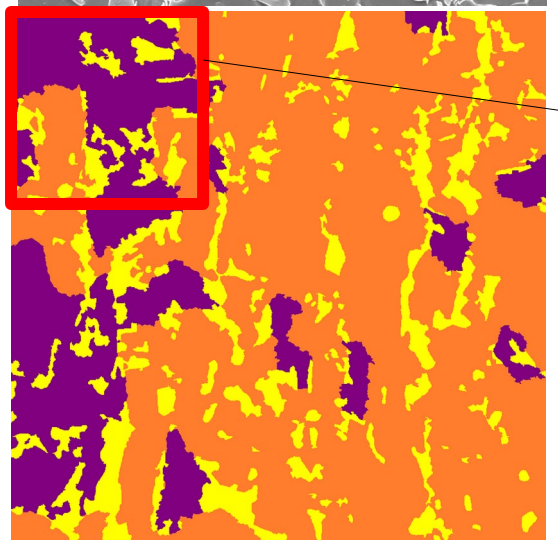
[[ 3737951.  522021.  135981.]
 [ 382343. 11028152.  186354.]
 [ 121146.  151955. 2346321.]]



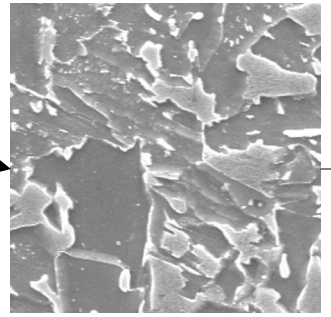
# Model Outlook

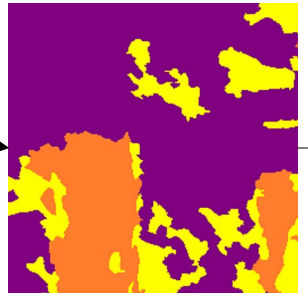
Image And Label





Augmented part of the Image





```
[[0.56862745 0.56862745 0.57254902 ... 0.94901961 0.95686275 0.96078431]
 [0.57254902 0.57254902 0.56862745 ... 0.91372549 0.9254902  0.93333333]
 [0.57254902 0.57254902 0.56078431 ... 0.8627451  0.87843137 0.89019608]
 ...
 [0.56078431 0.56078431 0.54509804 ... 0.6627451  0.67058824 0.67843137]
 [0.58039216 0.58039216 0.57254902 ... 0.69019608 0.68627451 0.68627451]
 [0.56470588 0.56470588 0.56078431 ... 0.70588235 0.70588235 0.71372549]]
```

Converted into Gray-scale

Formatted into corresponding classes of [0, 1, 2]

Ideal case

```
[[2 2 2 ... 0 0 0]
 [2 2 2 ... 0 0 0]
 [2 2 2 ... 0 0 0]
 ...
 [2 2 2 ... 0 0 0]
 [2 2 2 ... 0 0 0]
 [2 2 2 ... 0 0 0]]
```

But the observed formatting of some images turned out to be like this

random image

→

```
[[ 53  53  53 ... 226 226 226]
 [ 53  53  53 ... 226 226 226]
 [ 53  53  53 ... 226 226 226]
 ...
 [ 53  53  53 ... 226 226 226]
 [ 53  53  53 ... 226 226 226]
 [ 53  53  53 ... 226 226 226]]
```

MODEL

# Closer Inspection

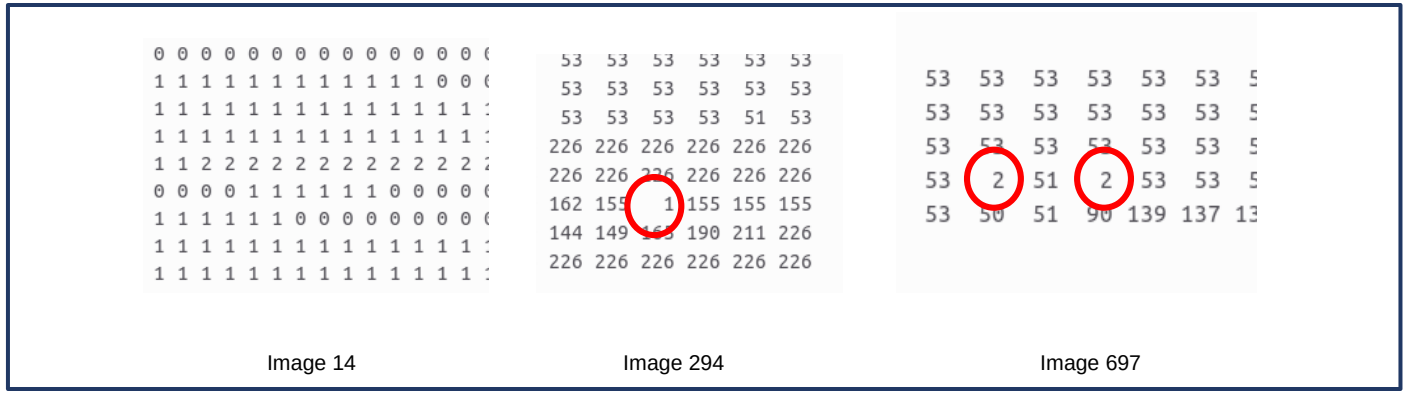
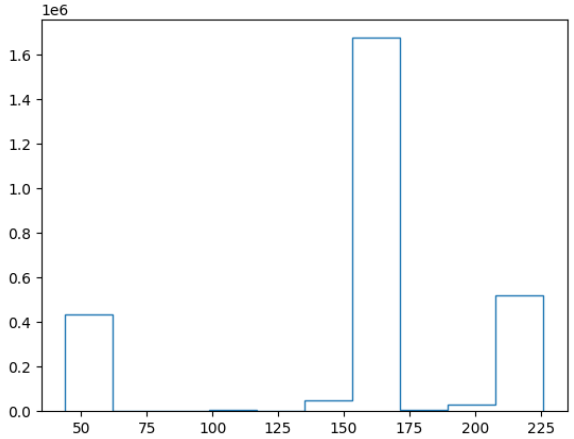


Image 14 Image 294 Image 697




Gray-scale Pixel Distribution

```
label[label==225] = 0
label[label==154] = 1
label[label==52] = 2
#label[label==149] = 3

label_b[label_b==225] = 0
label_b[label_b==154] = 1
label_b[label_b==52] = 2
label_b[label_b==149] = 3
```

Existing Corresponding code

**LM 0** → The Code tried to convert specific pixel values to be set to [0, 1, 2] under a given condition. But in reality such pixel values were not consistent in images.

**LM 1** → I tried to set a buffer of 3 pixels and the conversion was more consistent and with few outliers.

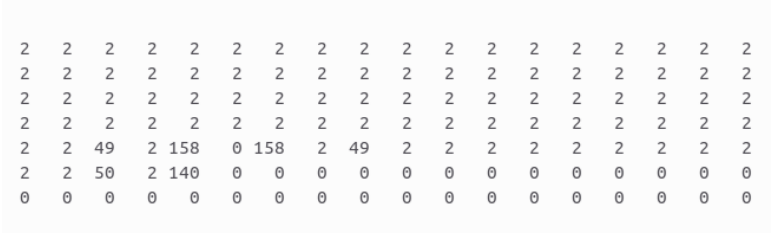


Image 381

# LM0 vs LM1

Comparative Output Results between LM0 and LM1 after 1 iteration

Existing code

**LM 0** → Val Accuracy = 36.41%
Time per epoch = 1m 48s

Modified code

**LM 1** → Val Accuracy = 64.23%
Time per epoch = 1m 57s

With 태호 's help, we thought of another approach (**LM2**) of removing color interpolation that could improve the results. And if implemented at the time of image augmentation could improve the running time of the model.

Will run the model after some modifications and give the result as soon as possible.