# Semantic Segmentation with VGG16-UNet Architecture

## Table of Contents

1. Introduction

2. Experiments and Observations

2.1 Small Dataset Notebook

2.2 Session\_Crashes Notebook

2.3 IntroduceDataLoaders&RemoveAugmentation Notebook

2.4 VGG16\_UNET\_BEST Notebook

3. Lessons Learned

4. Challenges to Address in Future Work

5. Conclusion

## 1. Introduction

This report outlines the process of training a semantic segmentation model using VGG16 as an encoder and UNet as a decoder. The project involves a series of experiments across multiple notebooks, starting from handling a small dataset to resolving complex training issues. The following sections summarize the challenges faced, solutions implemented, and improvements achieved.

## 2. Experiments and Observations

### 2.1 Small Dataset Notebook

Dataset: Only 12 images (approximately 1 year of data).  
Epochs: 10.  
Results:  
- Training accuracy: ~60%.  
- Issue: Model overfitting due to the extremely small dataset size.

Final Training Accuracy: 0.6022

Final Validation Accuracy: 0.6167

Best Epoch (Validation Accuracy): 8

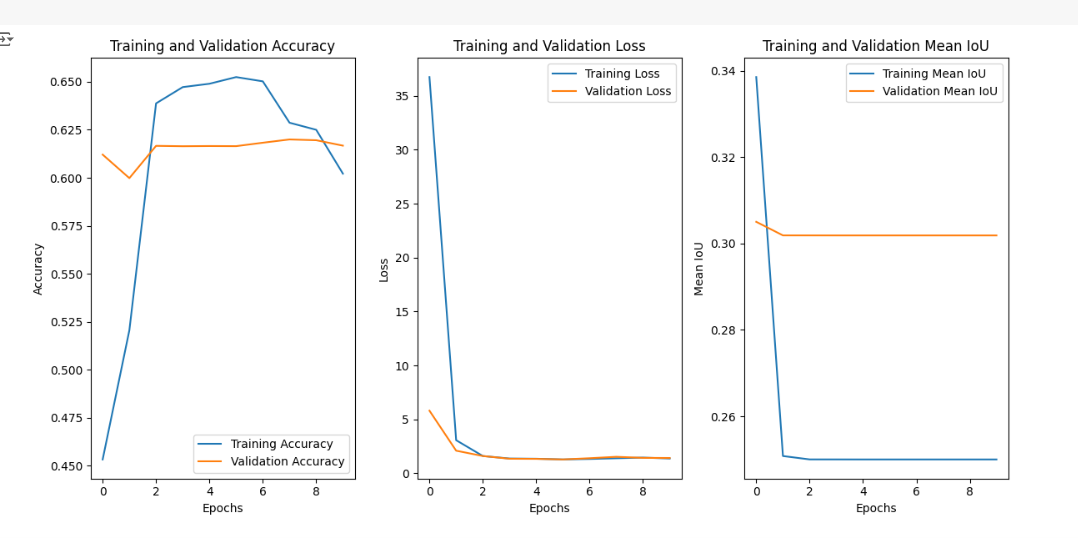
Best Validation Accuracy: 0.6200

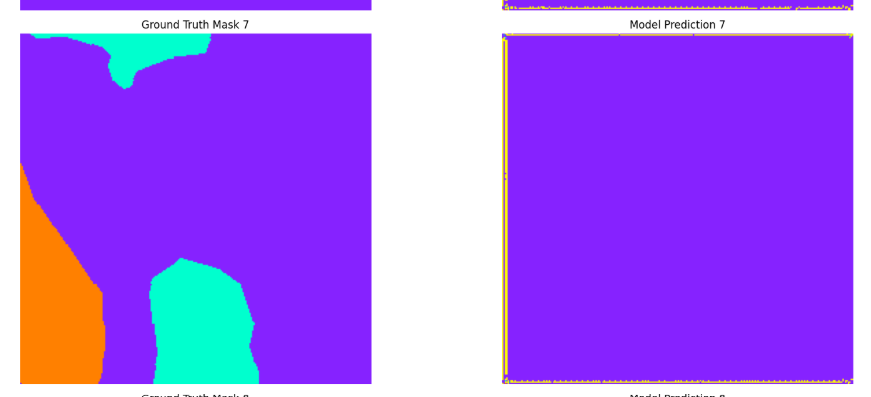
Final Training Loss: 1.3621

Final Validation Loss: 1.4113

Final Training Mean IoU: 0.2500

Final Validation Mean IoU: 0.3019



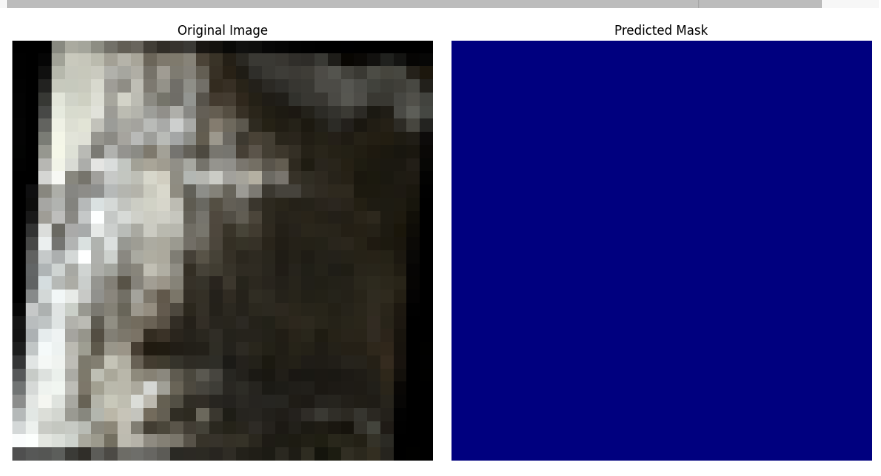


### 2.2 Session\_Crashes Notebook

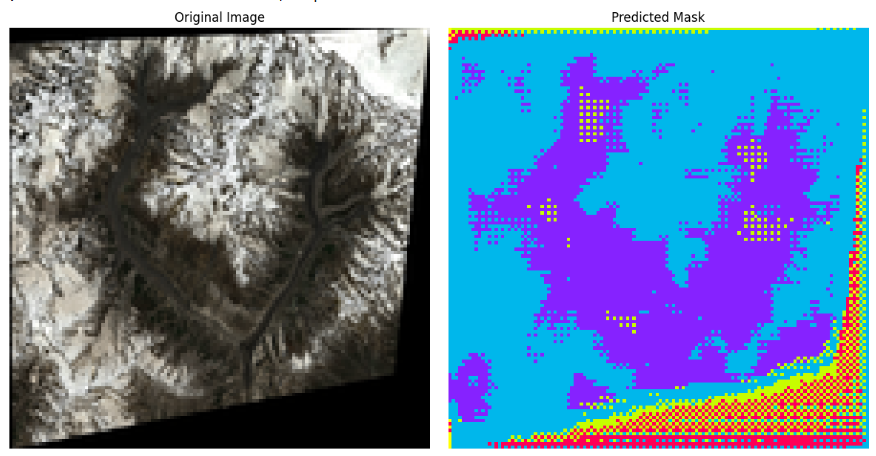
Challenges:  
1. Memory Overload:  
 - All data loaded at once, causing session crashes due to limited GPU availability on Colab.  
2. Batch Size:  
 - Initial batch size: 32.  
 - High-resolution patches increased memory requirements, further exacerbating crashes.  
3. Patch Dimensions:  
 - Initial dimensions: 256x256, contributing to memory overload.  
4. Limited GPU Availability:  
 - Colab GPU availability was inconsistent.  
 - Faced issues with Kaggle: inability to register due to lack of Pakistani number support.  
 - Time wasted creating multiple Google accounts to access GPUs.  
  
Outcomes:  
- Poor predicted results due to noisy data augmentations.  
- Accuracy remained low, with blurred predictions.

### 2.3 IntroduceDataLoaders&RemoveAugmentation Notebook

Improvements:  
1. Data Loaders:  
 - Introduced to load one batch at a time, resolving memory overloads.  
2. Removed Data Augmentation:  
 - Eliminated augmentation to reduce noise, leading to sharper predictions.

  
3. Dataset:  
 - Added more images (Passu 2019 and Shisper 2023 datasets).  
  
Results:  
- Accuracy improved to the 80s (Epochs = 500).

accuracy: 0.8518 - loss: 0.1641 - mean\_io\_u\_15: 0.4518 - val\_accuracy: 0.0054 - val\_loss: nan - val\_mean\_io\_u\_15: 0.4375 - learning\_rate: 5.0000e-04



### 2.4 VGG16\_UNET\_BEST Notebook

Final Improvements:  
1. Optimized Data Loading:  
 - Data loaded in manageable batches.  
2. Batch Size:  
 - Reduced to 4 to handle high-resolution patches.  
3. Patch Dimensions:  
 - Adjusted from 256x256 to 128x128 for optimal performance.  
4. Expanded Dataset:  
 - Included Passu and Shisper datasets for 2019-2021.  
  
Results:  
- Achieved ~91% accuracy with 1000 epochs.  
- Predictions were highly accurate, resolving prior issues of blurriness.

Training Loss (last epoch): 9.05%

Training Accuracy (last epoch): 91.15%

Mean Training Loss: 20.61%

Mean Training Accuracy: 80.29%

**2/2** ━━━━━━━━━━━━━━━━━━━━ **1s** 580ms/step - accuracy: 0.9032 - loss: 0.0990 - mean\_io\_u\_3: 0.6339

Validation Loss: 10.09%

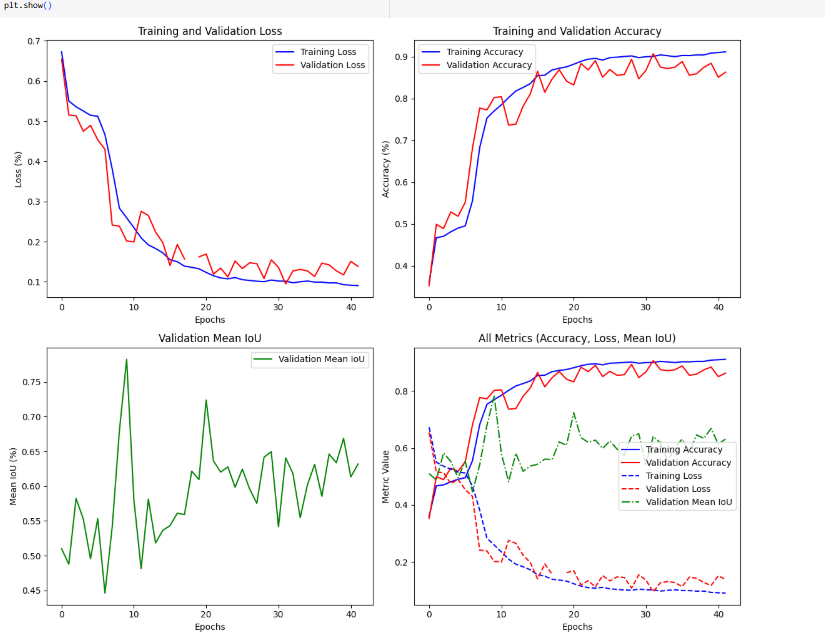
Validation Accuracy: 90.12%

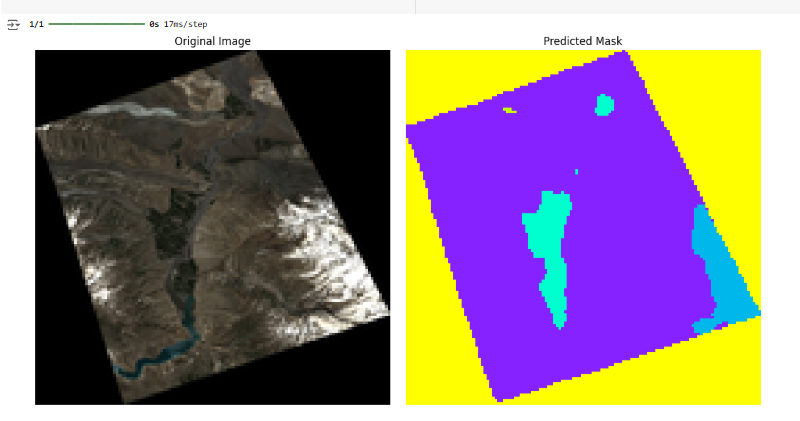
Validation Mean IoU: 63.07%

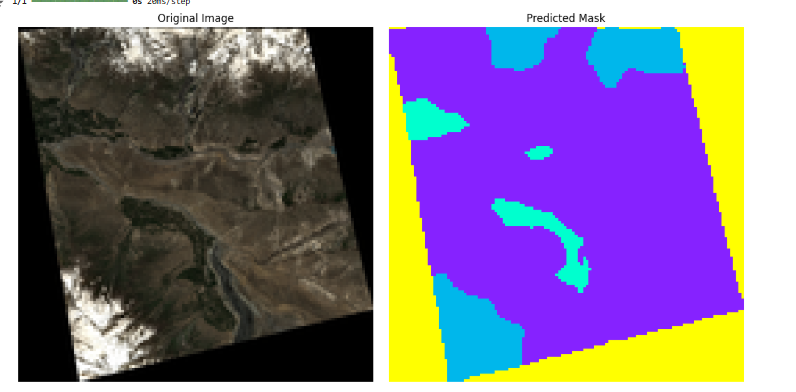
Mean Validation Loss: nan%

Mean Validation Accuracy: 79.06%

Mean Validation IoU: 59.25%







## 3. Lessons Learned

Prompt Engineering:  
- Optimized interaction with ChatGPT to refine model training steps.  
- Developed a structured approach for debugging and resolving issues effectively.

## 4. Challenges to Address in Future Work

1. Class Imbalance:  
- Dataset includes 8 classes, with some classes underrepresented.  
- Plan: Implement techniques like class weighting or oversampling to balance training.  
  
2. Dataset Expansion:  
- Add more data for better generalization.  
- Explore working with PPM files for higher image quality.  
  
3. Loss Functions:  
- Experiment with different loss functions (e.g., Dice Loss, Focal Loss) to handle class imbalance and improve performance further.

## 5. Conclusion

This project has demonstrated iterative improvements in semantic segmentation by addressing key issues related to memory, data augmentation, and model optimization. Future work will focus on resolving class imbalance and enhancing dataset quality to further boost accuracy.