Semantic Segmentation on Electron Microscopy Dataset

Sanchit Kabra

April 3, 2023

1 Introduction

This report details the implementations of semantic segmentation on Electron Microscopy Dataset acquired by Graham Knott and Marco Cantoni at EPFL. The dataset annotates mitochondria in two sub-volumes. Each sub-volume consists of the first 165 slices of the 1065x2048x1536 image stack. Semantic segmentation is the process of dividing an image into multiple segments, where each segment represents a different object or region in the image. This requires the network to learn to identify the boundaries of different objects and accurately segment them from the background. I use VGG-UNet as it has shown to achieve state-of-the-art performance on several benchmarks.

2 Dataset

The dataset is derived from a small 5x5x5µm segment that was extracted from the CA1 hippocampus region of the brain. This segment has a volume of 1065x2048x1536, providing a fairly large amount of data for analysis. The resolution of each voxel within this dataset is approximately 5x5x5nm.

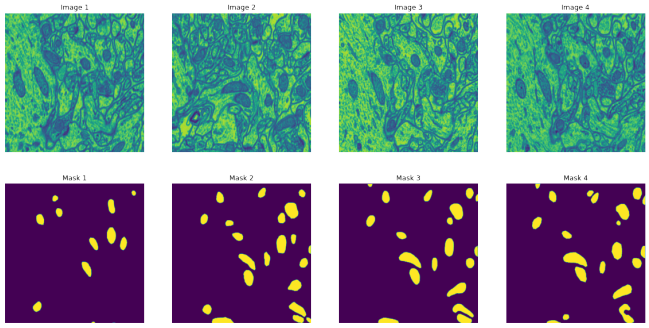

Figure 1: Slice of dataset

3 Approach

There are several models that can be used for semantic segmentation, some of which are: Fully Convolutional Networks (FCN), U-Net, DeepLab, Mask R-CNN, etc. We use The U-Net and VGG architectures as they can be combined to create a hybrid model that takes advantage of the strengths of both architectures.

The U-Net architecture is a fully convolutional neural network that consists of an encoder and a decoder. The encoder is a series of convolutional and pooling layers that gradually reduces the spatial resolution of the input image while increasing the number of channels. The decoder is a series of up-convolutional and concatenation layers that gradually increase the spatial resolution of the feature map while decreasing the number of channels. The U-Net architecture is designed to capture fine-grained spatial information while maintaining high-level semantic information.

The VGG architecture is a convolutional neural network that is used for image classification. VGG consists of a series of convolutional and pooling layers followed by fully connected layers. VGG is known for its ability to extract high-level features from images.

To combine the U-Net and VGG architectures for semantic segmentation, I can use the VGG network as the encoder in the U-Net architecture. This means that I replace the initial convolutional layers in the U-Net with the convolutional layers from the VGG network. This allows the U-Net to take advantage of the high-level features learned by the VGG network while still capturing fine-grained spatial information. The U-Net architecture with a VGG encoder (UnetVGG) is often used for medical computer vision tasks such as segmentation of medical images. I also leverage Transfer learning as the VGG encoder in UnetVGG uses pre-trained weights that have been trained on large dataset ImageNet. The pre-trained weights are used as a starting point for the training process on the medical dataset. This can significantly reduce the amount of training data required and improve the performance of the model.Medical datasets are often small due to the difficulty and cost of collecting and annotating medical images. The UnetVGG architecture has been shown to perform well even on small datasets, making it a suitable choice for medical computer vision tasks.

I use Binary Cross Entropy as Loss function as it is a common loss function used for binary classification tasks, which is essentially what semantic segmentation is - the classification of each pixel in an image as belonging to a specific class or not. Moreover, it is easily differentiable and can be efficiently optimized using gradient-based optimization algorithms. I pick BCE over dice or focal loss as they may be better than binary cross entropy loss in semantic segmentation tasks with imbalanced datasets, where the positive class is rare. Since this is not the case, Binary Cross Entropy performed decently well and converged over time. I also experimented with dice loss being the loss function. It resulted in slow convergence and an accuracy of 91.34I use the Adam optimizer over Adagrad and SGD owing to its faster convergence. I also experiment with Adagrad. It lead to slower convergence of loss function with accuracy of 93.46 % as compared to Adam optimizer resulting in accuracy of 95.2 %.

4 Results

Metrics used for evaluating the model performance are listed in the table below: article

Metric	Value
Accuracy	xx.xx
IoU	xx.xx
DICE Score	xx.xx

The DICE score is a metric used to evaluate the performance of image segmentation algorithms. The score ranges from 0 to 1, with a value of 1 indicating a perfect overlap between the predicted segmentation and the ground truth segmentation.

The DICE score is calculated as follows:

$$DICE = \frac{2\,|A \cap B|}{|A| + |B|} \tag{1}$$

where A is the predicted segmentation, B is the ground truth segmentation, and $|A|$ and $|B|$ are the number of pixels in each segmentation. The intersection between A and B is denoted by $|A \cap B|$. Intuitively, the DICE score measures the proportion of pixels that are correctly classified by the segmentation algorithm. A higher DICE score indicates a more accurate segmentation. The DICE score is commonly used in medical image analysis, where accurate segmentation is critical for tasks such as tumor detection and tracking.

IOU stands for Intersection over Union, which is another metric commonly used to evaluate the performance of image segmentation algorithms. Like the DICE score, the IOU metric is also based on the overlap between the predicted and ground truth segmentations. The IOU is calculated as the ratio of the number of pixels in the intersection of the predicted and ground truth segmentations to the number of pixels in their union. Mathematically, it can be expressed as:

$$IOU = \frac{|A \cap B|}{|A \cup B|} \tag{2}$$

The model was trained for 25 epochs each of 60 steps with average time taken per step being 9 seconds. I also trained it for 50 epochs but the training generally plateaued by the 15th epoch. The graphs of loss functions and accuracy are given below:

article graphicx

I did notice a lack of usual convergence of training loss. The final loss function values arent far from the ones which i obtained after the initial epochs. I suspected this indicating towards overfitting but the model

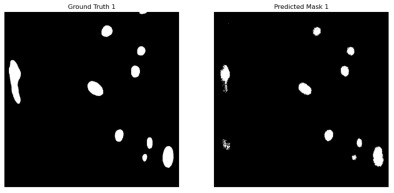

Figure 2: Test set example 1

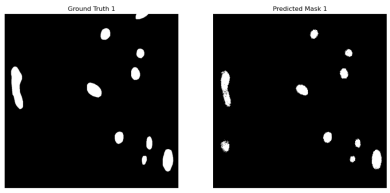

Figure 3: Test set example 2

perfomed well on validation sets with hardly any accuracy drops. Its test accuracy was 95.92% which further eliminated my doubts. But to further confirm my findings, i added l2 also called as ridge regression to the final two convolutional layers. The model training converged slowly as compared to when regularization isnt applied. Furthermore, it achieved an accuracy of 94.24% which is marginally less than when regularization isnt applied. Since application of regularization didnt impact the metrics, the model wasnt facing overfitting and was being trained ideally. It is also observed that the model is weaker at the edges of the images. The most probable i can think of resulting in this phenomenon is the limited context owing to cutting off of the section of interest. The final results of model on the test set are given below:

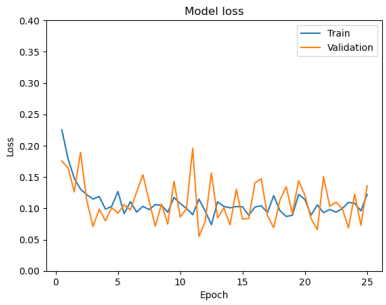

Figure 4: Loss function vs epoch

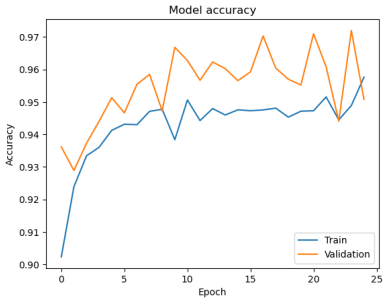

Figure 5: Accuracy vs epoch

The model achieved an accuracy of 95.13% on the test set with IoU of 73.74%.

5 Conclusion

In conclusion, the application of semantic segmentation using the U-NetVGG architecture on electron microscopy datasets has shown promising results. The model converges quickly on its loss during training an and is able to showcase a good accuracy over the test dataset of 95.92 % validating that model training is successfully generalizing over the distribution. More experimentation on data augmentation, loss functions and optimization were also performed which concluded that the best setting was using Binary Cross Entropy as loss function, Adam optimizer with 0.0001 as learning rate.