DS8013 – Deep Learning

# U-NET CONVOLUTIONAL NEURAL NETWORKS FOR IMAGE SEGMENTATION

Carlos Go, Smit Patel

**Abstract**

Working with datasets comprised of images can be complex and requires the use of many layers to process and make predictions with the given dataset. However, with the use of both Python as well as existing known CNN models, one can process, manipulate, and make prediction on the data in order to create functional models that are able to work with any image data. In this report, we will make use of the U-Net framework, which is a branch of CNN in order to process biomedical retinal vessel images and to create predictions on whether a disease can be identified for a given image.

## Table of Contents

## 1 Introduction

### 1.1 Problem Definition

There are a handful of eye diseases that have seen an upturn throughout recent years and more work is being done to identify these diseases as early as possible to avoid further retinal damage. Both blood vessels and the overall retinal vascular system can provide an abundance of information on the state of an eye and the use of medical image segmentation on these systems has become more prominent. As a result, this paper aims to use these existing segmentation methods to solve the problems of both segmentation errors as well as low accuracy in traditional retinal segmentation.

### 1.2 Suggested Solution

Convolutional Neural Networks (CNN) are a popular ML framework widely used for image classification. A common task would be outputting a single class label for an image, but for other visual tasks like biomedical image processing, the class label was assigned to each pixel. A solution was initially created to solve such a task by using a sliding window CNN but left room for improvement, specifically on time efficiency and accuracy. U-Net was thus created to build upon the current method with modifications that yields better results with few training images.

This project aims to reproduce a pre-trained U-Net framework and see its applications for biomedical image segmentation as well as other use cases. Making use of the existing dataset ‘CHASE’, which contains 20 images for retinal vessel segmentation, the end goal is to extract image patches around the vessel pixels and use this as an input to the existing U-Net framework.

# 2 Literature Review

Although there are many papers available on this topic of biomedical image segmentation involving the U-Net framework, we chose one that made improvements on the architecture such that is works with very few training images and yields more precise segmentations, as we felt this specific type of data is not easily accessible and using an architecture that supports smaller subsets of training images would prove useful. The paper can be accessed through arXiv[1].

## 2.1 Summary of Paper

The main idea of this paper is to build upon and improve on a previous network, which was trained in a sliding-window setup to predict the class label of each pixel by providing a local region or "patch" around that pixel. This network won a challenge and the paper "Deep Neural Networks Segment Neuronal Membranes in Electron Microscopy Images"[2] goes over the overall architecture, but there were two identified drawbacks. Namely, the model was slow due to the network needing to be run separately for each patch and there was a significant trade-off between localization accuracy and the use of context (i.e., larger patches reduced localization accuracy and small patches only allow network to see little context). Thus, Ronneberger et al. [1], established an architecture consisting of a downsampling path of unpadded convolutions, reLU and max-pooling operators, and replacing the latter pooling operators with upsampling operators, which in turns create the U-shape symbolizing the U-Net framework. In this manner, the feature channels are doubled (starting from a small number of features) through downsampling and are halved (starting with the largest number of features) through upsampling, which allows the network to propagate context information to higher resolution layers. Through replacing the pooling operators with upsampling, the upsampling layers are also able to increase the resolution of the output and combining the high resolution features from the downsampling allows for more precise localization. The network ends up using a total of 23 convolutional layers. Figure 1 shows how a typical U-Net model looks like.

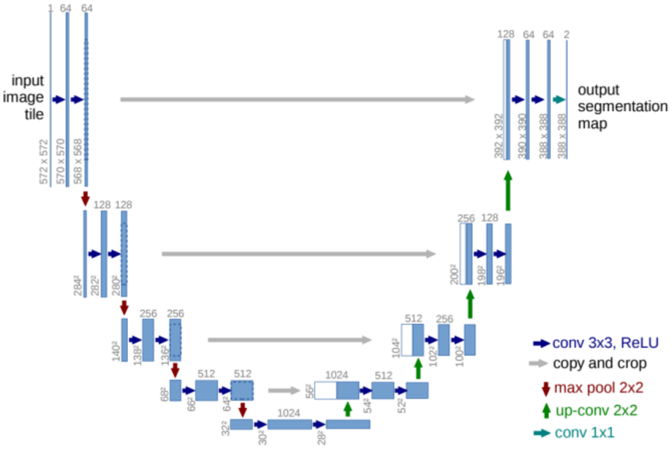


*Figure 1 – U-Net Model Architecture created by Ronneberger et al.[1]*

Majority of the training data used was through data augmentation by applying elastic deformations to the available training images, which allowed the network to learn invariance to such deformations, which is valuable to biomedical segmentation since deformation is known to be the most common variation in tissue and realistic deformations can be simulated efficiently. This is also essential in order to teach the network the desired invariances and robustness properties, using mainly shift and rotation invariances as well as robustness to deformations and gray value variations. The input images and their corresponding segmentation maps were used to train the network with the stochastic gradient descent implementation. Besides the data augmented images, there were 3 datasets of neuronal and cell microscopic images that were used. Each dataset ranged from 20-30 images of 512x512 pixels, which also came with a corresponding fully annotated ground truth segmentation map. The evaluation was obtained by sending by thresholding the map at 10 different levels and computing the "warping error", "Rand error" and "pixel error", which were all then compared to the previous model to determine if this newly developed model was indeed better. The conclusions of the comparison were that the u-net model achieved a lower warping error and lower Rand error as well as a much higher IOU ("intersection over union") than that of the sliding window model. Thus, the U-Net framework is now widely-used as the most popular network for biomedical image segmentation while also requiring less training images and much less training time.

### 2.2 Related Work

Further to the paper by Ronneberger et al. [1], we also looked at other datasets that were more specific to our space of biomedical segmentation, which is focused on retinal vessel segmentation, which brought us to papers constructed around "Improving Retinal Vessel Segmentation Trained with Noisy Labels[3]" as well as "Recurrent Residual Convolutional Neural Network based on U-Net (R2U-Net) for Medical Image Segmentation[4]". Both of these papers build upon the U-Net framework and involve processes of residual networks in order to create a more complex and more accurate U-Net, coined "Res-UNET" and "Attention U-NET". Although both of these modifications of the Simple U-Net framework would prove to improve both the Average Accuracy as well as Mean IoU of our models, the results were marginal and so we decided to proceed with the simple U-Net, and keep our focus on the transfer learning aspect for this project. A sample of how the average accuracy and Mean IoU changes between the HRF and DRIVE datasets mentioned in Section 3.2 can be seen in Table 1 below.

| Dataset | Model | Average Accuracy | Mean IoU |
|---|---|---|---|
| HRF | **Simple U-Net** | **0.965** | **0.854** |
| HRF | **Res-UNet** | **0.964** | **0.854** |
| HRF | **Attention UNet** | **0.966** | **0.857** |
| DRIVE | **Simple U-Net** | **0.9** | **0.736** |
| DRIVE | **Res-UNet** | **0.903** | **0.741** |
| DRIVE | **Attention UNet** | **0.905** | **0.745** |

*Table 1 – Results of each dataset based on different U-net model applications taken from Github[9]*

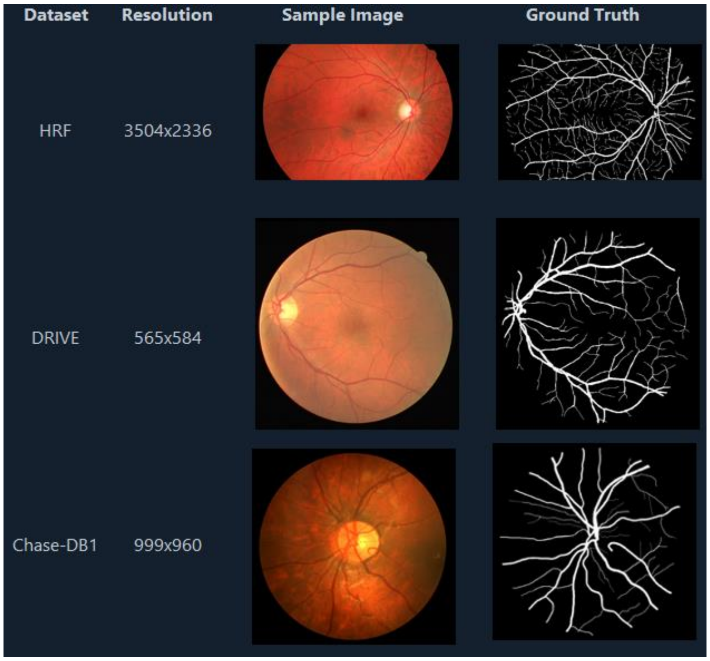
## 3 Data

### 3.1 Data Description

CHASE_DB1[5] is a dataset for retinal vessel segmentation which contains 28 color retina images with the size of 999×960 pixels which are collected from both left and right eyes of 14 school children. Also provided for each image was two sets of ground-truth vessel annotations, where each mask and input image were to be fed into the training and testing model. For the purpose of this project, only one set of ground truths was used.

### 3.2 Other Datasets

The model this project uses has been pre-trained with similar datasets, namely the HRF[6], DRIVE[7] and STARE[7] datasets, all of which contain retinal vessel images. Figure 2 shows sample images from the DRIVE, HRF and Chase-DB1 datasets and how they differ from each other. In the sample images shown, it is clearly seen that the images from the HRF and DRIVE datasets have the eye facing a certain direction, which isn't the case with Chase-DB1. It is important to note that not every image from HRF and DRIVE have the eyes facing the same direction, but all images from Chase-DB1 have the eyes set at the same position.



| Dataset | Resolution | Sample Image | Ground Truth |
|---|---|---|---|
| HRF | 3504x2336 | | |
| DRIVE | 565x584 | | |
| Chase-DB1 | 999x960 | | |

*Figure 2 – Sample images and Ground Truths from HRF, DRIVE and Chase-DB1 datasets*

## 4 Analysis

### 4.1 Architecture Design

As previously mentioned, the model this project uses has already been pre-trained with similar datasets. In this implementation, there are a total of 79 layers which are a combination of convolutional, dropout, max pooling and batch normalization layers. The encoder part works like a typical CNN, it performs feature extraction and reduces the spatial dimension of the feature maps using a convolutional and pooling layers respectively. The image is reduced to 32x32 before the decoder begins using up-convolutional layers to upsample the feature maps and finally outputs and image of the same size as the input. There are also skip connections that concatenate the high-resolution feature maps from the encoder with the upsampled feature maps from the decoder. This allows the network to use information from both high and low-resolution feature maps to make more accurate predictions.

## 4.2 Analysis

### 4.2.1 Preprocessing and Training

While the Chase-DB1 dataset is highly similar with the ones the model has already been trained with, the positions of the eyes may vary as soon in Figure 2. This may cause lower performance if the model is not fine-tuned with this new dataset. 18 images from the Chase-DB1 dataset are used for training and validation while 10 are kept for testing. Figure 3 and Figure 4 below show the outputs of some hidden layers in the encoder and decoder parts of the network to visualize how the image is processed.

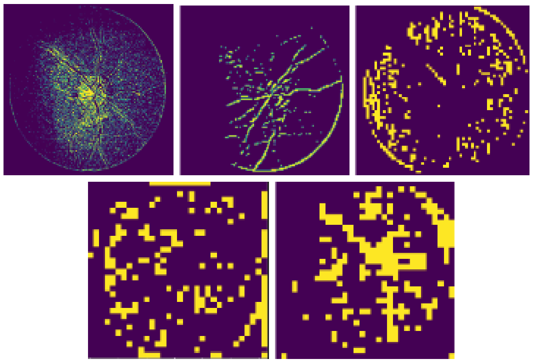


*Figure 3 – Encoder hidden layer outputs*

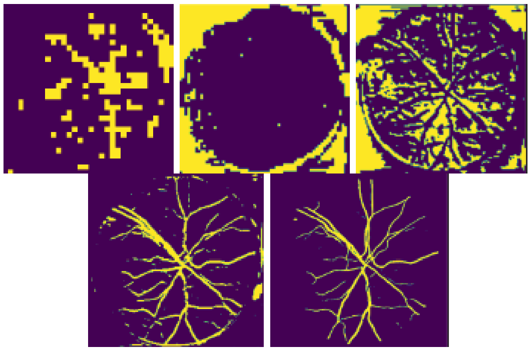
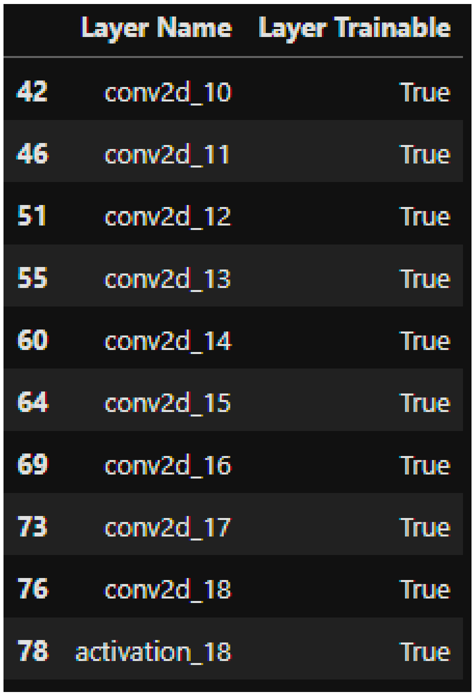


*Figure 4 – Decoder hidden layer outputs*

As seen in Figure 3 and Figure 4, there are steps taken when preprocessing the data. The main step is green channel selection and applying the Contrast-limited adaptive histogram equalization (CLAHE). The data is also cropped into nonoverlapping 512x512 patches so it can be fed into the model. This preprocessing step was the core initial step in making the training data usable for both the purposes of within the u-net model, as well as for transfer learning, which is applied in the latter steps of this project.

### 4.2.2 Transfer Learning and Fine-Tuning

Several experiments were performed to determine what set of hyperparameters would yield the best performance. Following the guidelines of transfer learning, it was determined that training the up-convolutional layers in the decoder yielded the best results. Figure 5 shows the layers that were set to “trainable” while keeping all other parts of the model frozen. The number of epochs were kept at 150, which was the same as the original number the model was pre-trained with. The learning rate was also lowered from 0.001 to 0.0005. Early stopping using validation accuracy with patience 10 was also used to ensure that our model would not overfit.



| | Layer Name | Layer Trainable |
|---|---|---|
| 42 | conv2d_10 | True |
| 46 | conv2d_11 | True |
| 51 | conv2d_12 | True |
| 55 | conv2d_13 | True |
| 60 | conv2d_14 | True |
| 64 | conv2d_15 | True |
| 69 | conv2d_16 | True |
| 73 | conv2d_17 | True |
| 76 | conv2d_18 | True |
| 78 | activation_18 | True |

*Figure 5 – Decoder hidden layer outputs*

### 4.3 Testing and Results

While all 150 epochs were needed during training, it is clearly visible that validation accuracy and loss peaks around the 25th epoch at 0.932. It is also evident that training loss and accuracy eventually end up with a better performance.

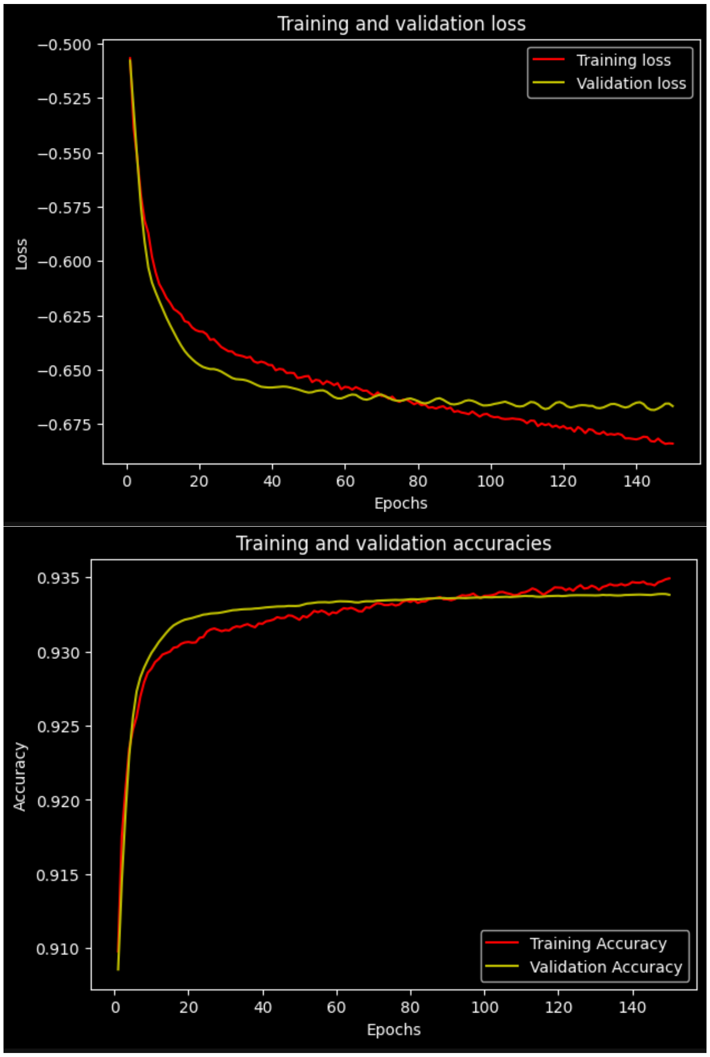


*Figure 6 – Training and Validation Loss and Accuracies*

In addition to accuracy, the Intersection over Union (IoU) loss or Jaccard similarity is also used to evaluate the model performance The IoU score represents how much overlap there is between the predicted image and the ground truth images. The higher the IoU, the better the prediction.

Finally, during evaluation, the newly trained model with Chase-DB1 is pitted against the original pre-trained model without any changes to determine any differences in performance. After testing with the 10 images, the fine-tuned model had an average accuracy of 0.986 and mean IoU of 0.8974, while the untouched pre-trained model had 0.9568 and 0.8455, respectively. Both of these results can be seen in Table 2 below.

| | Pre-Trained Model | Fine-Tuned Model |
|---|---|---|
| Avg. Accuracy | **0.9568** | **0.986** |
| Mean IoU | **0.8455** | **0.8974** |

*Table 2 – Results of both Fine-Tuned and Pre-Trained Models*

Further, the results can be seen in Figure 7, which shows the difference between the 2 models' predictions and the clear advantage of the fine-tuned model. The output of the untouched model is also very similar to the output of one of the latter hidden layers of the fine-tuned model, seen in the 4th image in Figure 4, where some patched are seen on the edges of the image.

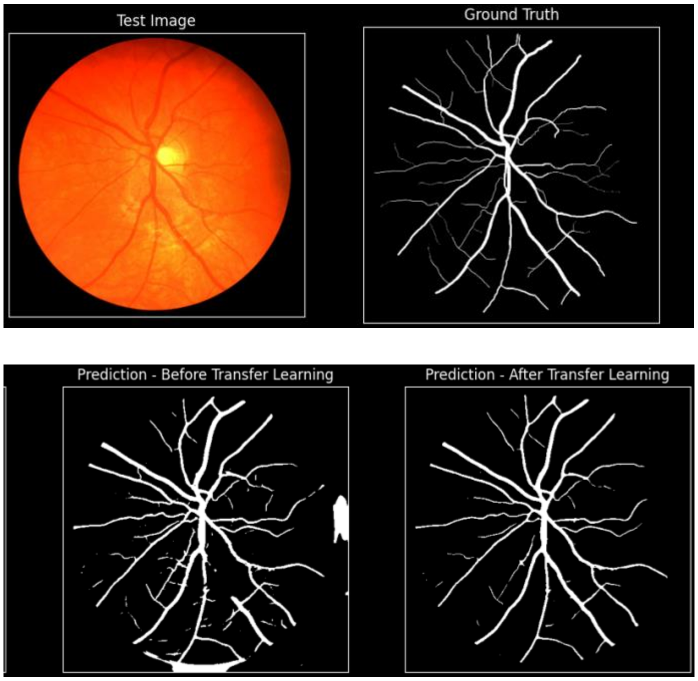

Test Image
Ground Truth
Prediction - Before Transfer Learning
Prediction - After Transfer Learning

*Figure 7 – Test Image and ground truth images, followed by prediction image before and after transfer learning*

## 5 Lessons Learned

Upon going through the stages of preprocessing the data, training the model using transfer learning, fine tuning the model, as well as testing the model, we felt we were able to accomplish our task of building upon the original U-Net model to a point where our average accuracy and mean IoU were better than the original model itself. Along the way, there were a few key learnings and conclusions which we were able to draw. Firstly, the U-Net model has proven to be a very efficient tool for image segmentation. This is especially the case for domains with very little training data such as biomedical imaging and as seen throughout this paper, the U-Net model is extremely capable of achieving high accuracies on segmenting various retinal images, regardless of the size of the dataset used for training. Secondly, we found that the model itself is very flexible, as we saw a variety of other variations of the model, which were mentioned in Section 2.2., which were able to achieve even better results on different image segmentation tasks. Lastly, we found that this paper was also able to demonstrate that, even with a new and small training set, transfer learning can be implemented to improve a pre-trained U-Net model by targeting the appropriate layers. In our case, training the convolutional layers in the decoder part of the network had achieved the best results.

## 6 References

[1] https://arxiv.org/pdf/1505.04597v1.pdf

[2] https://papers.nips.cc/paper_files/paper/2012/file/459a4ddcb586f24efd9395aa7662bc7c-Paper.pdf

[3] https://arxiv.org/pdf/2103.03451v1.pdf

[4] https://arxiv.org/pdf/1802.06955v5.pdf

[5] https://blogs.kingston.ac.uk/retinal/chasedb1/

[6] https://www5.cs.fau.de/research/data/fundus-images/

[7] https://cecas.clemson.edu/~ahoover/stare/

[8] https://drive.grand-challenge.org/

[9] https://github.com/arkanivasarkar/Retinal-Vessel-Segmentation-using-variants-of-UNET