Automation of Vessel Segmentation for HEVs in Lymphatic 3D Image Analysis

By

Finlay Roy: 1629688

MSc Computing

School of Computer Science and Informatics, Cardiff University

Supervisor : Dr. Stefano Zappala

Date : 5th November 2021

Acknowledgements

I would like to thank my supervisor Dr. Stefano Zappala for providing excellent guidance and feedback throughout this project.
I would also like to thank Dr. Stefan Milutinovic for his help in outlining the problem to be solved as well as providing the images of the lymph nodes and their corresponding segmentation masks.

Abstract

This project surveys the existing literature surrounding vessel segmentation techniques and adapts multiple existing machine and deep learning frameworks to predict vessel segmentation masks. The predictions were performed upon a set of 3-dimensional High Endothelial Vessel (HEVs) networks within Lymph Node (LN) systems. The project aims to improve a current manual segmentation process pipeline by decreasing the time spent on producing segmentations while maintaining the high standard already achieved by the current pipeline. The current process involves a multi-step trial-and-error based approach that is very time-consuming. By improving the speed at which segmentation is performed, more time can be spent analysing data for Cancer Research and other scientific fields. The best quality predictions came from an implementation of the 3-D Unet framework that achieved a 0.717 Dice coefficient when compared with a pre-labelled segmentation mask of an LN image. This framework built upon the concepts of down-sampling and augmentation of image data as pre-processing techniques and showed their importance as steps for training segmentation models.

Table of Contents

Acknowledgements ii

Abstract ii

Introduction and Objectives 4

Background and Literature Review 7

Problem Statement 11

Approach 11

Application of Chosen Approaches 14

Results and Analysis 22

Conclusion 28

Reflection 31

References 33

Appendices 35

Introduction and Objectives

Research completed within Cardiff University, in partnership with the Medical Research Council, by Dr. Stefan Milutinovic investigated the role of High Endothelial Venules (HEVs) in cancer immunity (Milutinovic 2020). HEVs are a form of blood vessel typically found in lymph nodes (LN) and recent studies have found that they promote antitumour immunity through generation of 'cancerous tissue-destroying lymphocytes' (Ager and May 2015). To analyse HEVs impact on cancer immunity, light sheet fluorescence microscopy (LSFM) was used to obtain detailed 3-dimensional images of blood vessel networks within LNs.

The use of LSFM in this research (Milutinovic 2020) involved the fluorescent staining of blood vessels and HEVs within LNs using specific dyes (fluorophores) that attach to the sample. The sample LNs are illuminated in the microscope and the fluorophores absorb the illuminating light causing the blood vessels to emit a lower energy light than the light source. The emitted light is collected by the LSFM and a series of filters separate the illumination light from the sample light (Rice 2021). Image data from the LSFM is collected and used to produce a 3-D reconstruction of the sample, an example from Dr. Milutinovic's research can be seen in figure 1.

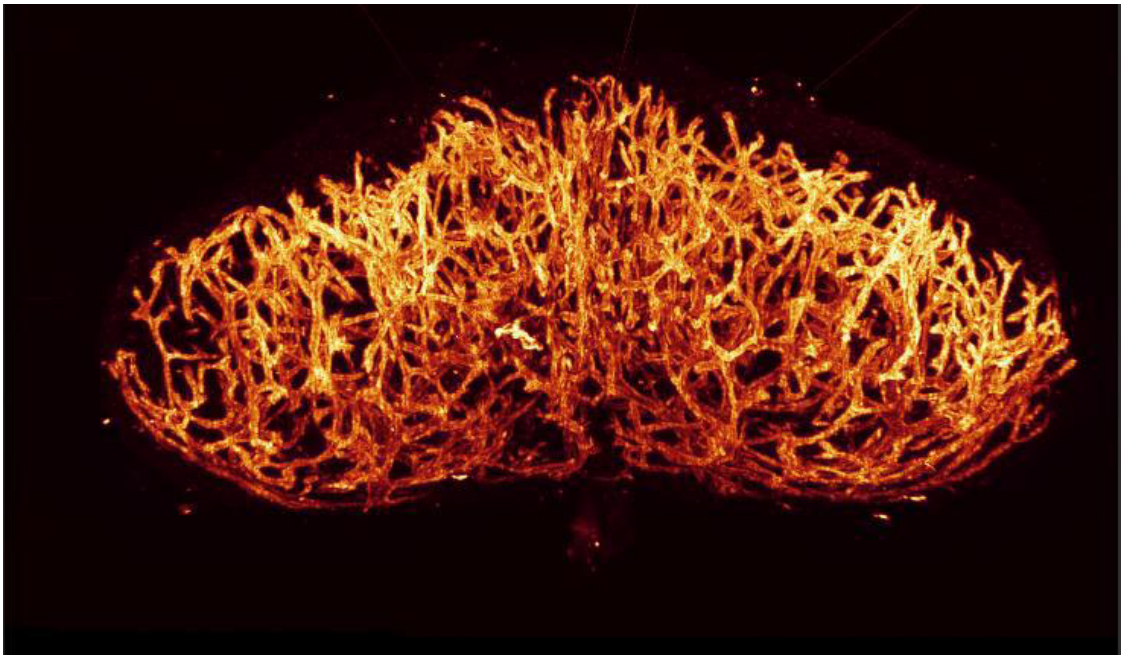

Figure 1 - 3-D reconstruction of a network of HEVs in a Lymph Node

Open-source command line computational 'Vessel Tools' were used to generate topological and numerical descriptors of the vessel networks inside the LNs, such as vessel diameter and length. These tools were developed by Dr. Gib Bogle (Kelch et al. 2015) and are accessible for public use (https://github.com/gibbogle/vessel-tools). A vital step in the collection of descriptors from the image data is the segmentation of the blood vessel networks. Segmentation can be described as an image processing technique involving the division of an image into regions which correspond to different objects within that image (BioSS 2021). In this paper, segmentation will refer to the separation of blood vessel from background LN tissue.

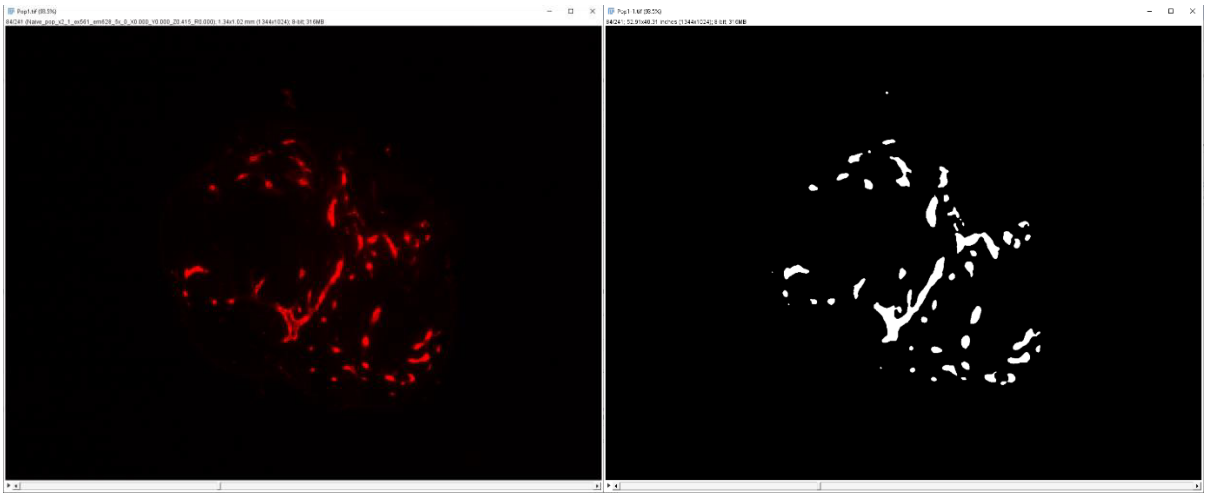

Figure 2 - (Left) 2-D section through a lymph node showing blood vessels (red) against faint lymph node background tissue. (Right) Segmentation of the same partial vessel network showing only the blood vessels (white).

The 'Vessel Tools' contain a series of tools developed to achieve accurate segmentation of vessels while using the staining process described before. The current pipeline involves an 8-bit 'tagged image file format' (TIFF) file being initially smoothed by one tool, then combined with the initial file in a thresholding algorithmic tool to produce a binary segmentation of the vessel network. However, an inconsistent staining of vessels can often be observed during the image analysis which leaves gaps in vessel walls and thus incomplete segmentations. A final filling tool utilises a basic 3-D algorithm to identify voxels within a vessel wall and is then applied to fill these gaps within the vessel walls.

The process of using these three tools (smoothing, thresholding, and filling) involves the user opening three separate instances of the command line and manually inputting user-chosen parameters for use in the tools' algorithms. The smoothing tool requires the

input of the radius of the mean smoothing filter. The thresholding tool requires the input of two parameters T and delta that affect the threshold level for what is considered a 'lit' voxel (value set to the 8-bit maximum 255) in a vessel wall depending upon a local average intensity. A low threshold value helps to account for faint areas within vessel walls. The filling tool requires the input of two parameters that measure the 'insideness' of a voxel, a specified radius around the voxel in question where the number of lit voxels is counted and a threshold value. If the number of lit voxels counted within the specified radius exceeds the threshold value, the voxel is considered within the vessel wall and is lit (Kelch et al. 2015).

The need for collection of large amounts of research data when working with medical research cannot be understated, and herein lies the main motivation for this project. The quality of the segmentations obtained in the previously described pipeline is judged solely upon the qualitative analysis of the user. This leads to a trial-and-error based approach in the selection of manually entered tool parameters. With each new LN, comes a multitude of attempts and failures to produce a satisfactory segmentation for analysis. The sheer amount of time used in segmentation with these tools, combined with the amount of time to become proficient with their use, creates a bottleneck on the amount of data that can be used for, and time that can be used in, analysis for important cancer research.

The primary aim of this project is to improve the speed of vessel segmentation by automating the process through machine and deep learning techniques, whilst maintaining the high-quality segmentation produced by the 'vessel tools'. The following objectives shall be strived for to meet this aim:

1. Identify a quantitative measure for use in comparing segmentations to maintain high-quality segmentations.
2. Improve the speed of the segmentation process.
3. Ensure the user input for the system/tool is limited.
4. Produce a solution that is easily relatable to the current tools and data in use, to enable ease of transfer from old tools to new.
5. Identify suitable existing models and frameworks from related literature to be adapted to this dataset.

Background and Literature Review

A review of current standard segmentation methods had been completed (Zhao et al. 2019) and gave a useful initial research area and overview of possible segmentation techniques to be adapted in this project. The paper reviewed segmentation methods of two categories, rule-based and machine learning based methods. Rule-based methods utilise designed rules and methods for classification of vessels and machine learning based methods use learned rules from training and testing datasets. The findings of the paper stated that machine learning-based methods were usually more computationally expensive than rule-based methods because of the large data sets. However, the results of the vessel segmentation from machine learning would be much more accurate. This paper provided introduction to many topics of further research such as K-Nearest Neighbours (kNNs), Random Forest and Convolutional Neural Networks (CNNs) etc. and their use in segmentation. The drawbacks of the results being they don't go into too much depth on any figures or statistics of any of the reports being reviewed. An issue of trusting the conclusions arises from this and further research would be required to make decisions on applicable topics.

A study focused on LSFM techniques to observe mouse brain vascular networks (Hu et al. 2017). The analysis of the images would involve the use of a hessian matrix and a feed forward neural network to deal with correlations between pixel values and utilize C-means clustering to segment vessel images. The results of this paper introduced the use of dice coefficients in evaluating vessel segmentation. The feed forward neural network was trained by a sample of 2.26 million pixels to deal with values in the Hessian matrix. This, combined with 85% dice coefficient results, shows the importance of a large sample in greatly improving the robustness of models. An area of concern here would be transferring methodology here from brain vascular systems and hoping to achieve similar results in lymphatic systems.

Similar to the C-means clustering in segmentation, other pixel classification approaches are seen to be used broadly throughout the literature. A comparative study on retinal vessel segmentation methods reviewed the performance of kNNs combined with Gaussian filtered features in segmentation (Niemeijer et al. 2004). The research team developed publicly available dataset called the DRIVE dataset consisting of testing and training retinal images and their corresponding segmentation masks (https://drive.grand-

challenge.org/) that can be seen in figure 3. The results from this study show that the kNN method outperformed the other automatic methods in this study but a second observer manually annotating an image still produces better results. However, it is stated that manual segmentation can take upwards of two hours for completion. This highlights a need for improved speed of segmentation to avoid fatigue and segmentation precision loss. Although this study provides a useful data source with the DRIVE database, it may be difficult to train models on datasets significantly different to the data in this project and achieve good segmentation results.

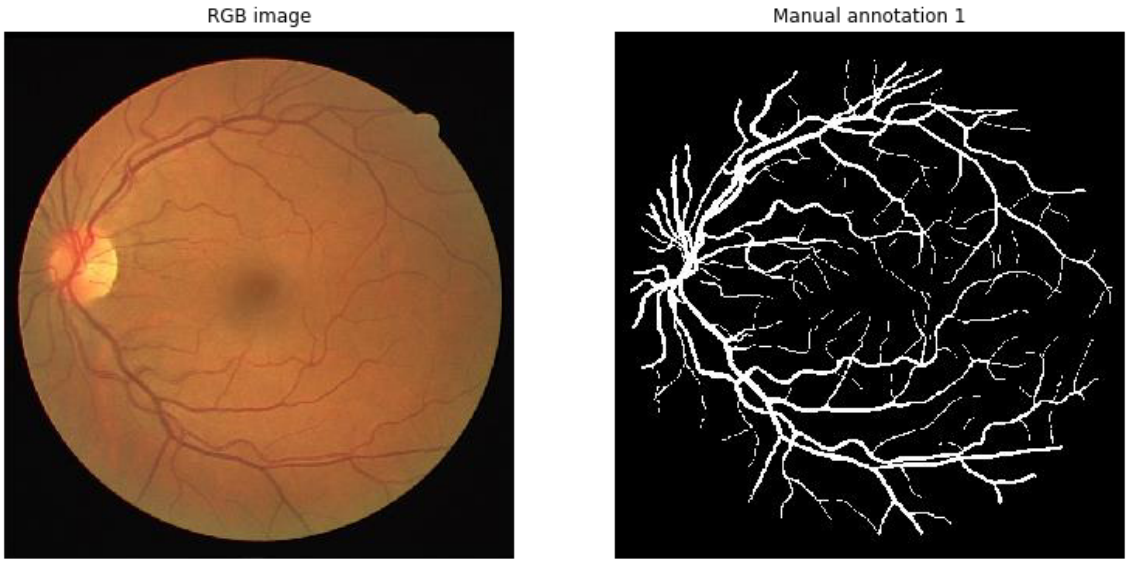

Figure 3 - Example retinal image (Left) and corresponding segmentation mask (Right) from DRIVE database

Several other papers approach the problem of retinal vessel segmentation by utilising the DRIVE database. Hassanpour (Hassanpour 2014) combined Gaussian edge detection features with random forest prediction trees for classification and achieved similar results to Niemeijer as described before. One study found the results of segmentation to be even greater when applying an Artificial Neural Network with gabor filters for feature extraction (Pal et al. 2012). These three studies performed on the DRIVE database all compare results using a maximum average accuracy metric, with the gabor filters method achieving the highest average accuracy metric on this Dataset of 96.16%. An issue may arise when comparing these results as both the filtering methods and classification methods differ across each study.

A study would apply a multi-scale CNN to integrate a network called DeepVessel (Fu et al. 2016). The experiments undertaken would show that the system would produce

excellent retinal vessel segmentation results. This article introduces some public domain datasets for retinal vessel images called STARE and CHASE_DB1 as well as the DRIVE database for training of models. Although a high level of accuracy is shown across the analysis on each of the datasets, it is mentioned that only the DRIVE database contains a training set, so parameters would only be fine-tuned on this database set. The parameters would then be used for segmentation of each of the database sets. As the DRIVE set only contains a small sample of 40 images of different vessel networks, this could be an issue when applying this to other models especially when considering deep learning models that require large inputs of data.

An article focused on a convolutional neural network (CNN), a hessian vesselness filter and intensity-thresholding to search nearby pixels and extract lymphatic shapes (Lai et al. 2020). The results obtained were concluded to contain minimal artifacts from this technique, while introducing a new application of a neural network framework called U-Net to be evaluated by dice coefficient. An important factor to note with these results is that the approach of gaining the image analysis is quite different. Optical coherence tomography (OCT) used which is a very different approach to LSFM in question, this could greatly skew any expected as OCT is considered non-invasive and does not involve the same staining process.

Another study looked to utilise the U-Net architecture for segmentation (Tetteh et al. 2020), however, this study looked to approach the segmentation of 3D volumes using deep learning. Their architecture replaced convolutional steps in the U-Net framework with their own designed crosshair filters that aimed to reduce computational load. To attempt to solve the problem of limited data availability they produced a synthetic dataset of 3-D vessel network structures using a method described (Schneider et al. 2012, 2014). This dataset consists of 136 3-D synthetic vessel structure image files and their corresponding 3D segmentation masks for use in training and testing. An example can be seen in figure 4. When training models using this dataset, it would be important to observe the differences to the data from the lymph nodes as the synthetic dataset is quite noisy and invariable it may produce different results to the extremely variable vessel staining process of imaging.

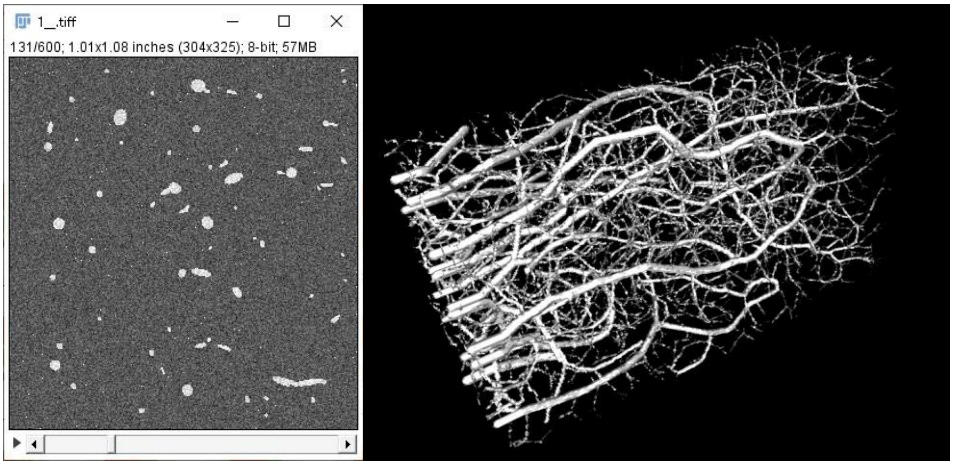

Figure 4 - Synthetic Dataset Example. (Left) 2-D slice through 3-D vessel network. (Right) 3-D vessel network representation.

It is clear from research that there is lots of outstanding methodology in this topic on the automation of vessel segmentation using both machine and deep learning networks. There is a clear indication to the use of public dataset libraries such as 'DRIVE', 'STARE' and 'CHASE_DB1', however, these datasets contain limited information for 2-D retinal imagery. This may produce issues when adapting methodologies concerning 3-D lymph vessel networks due to the limited data size and difference in imaging techniques. Appropriate methods of handling limited dataset problems are to be explored to ensure any trained model are effective. The dice coefficient seen here multiple times would act as a useful tool in deciphering the accuracy of any prototype methodology. A deep neural network is widely considered to have the potential for computational heavy actions, and this would need to be limited in the prototype to ensure ease of use in the laboratory. The method of training of a U-Net based CNN provided promising results with large datasets and would be an area of adaption in the project There appeared to be gaps in the literature on cases specifically involving the use of LSFM on lymphatic vessel networks and so care would need to be taken when adapting methodology specified for other biological networks.

Problem Statement

The initial sections of this paper outline the background and motivation of this project. In summary, the project hopes to solve the problem of achieving accurate and fast segmentations on large 3-D vessel networks when working with a limited dataset for training of artificial intelligence (AI) models. The current pipeline for the LN vessel networks provides high quality segmentation through a time-consuming trial-and-error based approach involving the manual input of user-chosen parameters. A solution developed for this problem looked to address the time-consuming nature of the process while maintaining the high quality of segmentation. The solution would need to pay close attention to specific difficulties faced by this problem. This includes the handling of differences between datasets for training and the target LN data, while ensuring parameters trained on models can be used for effective segmentation prediction. Limited dataset sizes will also pose a difficulty as training effective data-dependant AI models requires large datasets for training. Increasing the speed at which the segmentation process is completed will increase the throughput of the vessel network analysis pipeline allowing more data and more time to be spent on investigating the effects of HEVs on cancer treatment.

Approach

Datasets in Use

The training of AI models for vessel segmentation requires appropriate datasets that contain a raw input image of the vessel network and the corresponding segmentation binary mask. Dr Milutinovic provided images from his research of two separate types of LN (Milutinovic 2020). These are referred to as the Inguinal LN set and the Popliteal LN set. The sets include both the raw 3-D images of the LNs and the segmentation masks that were manually segmented using the tools developed by Dr Gib Bogle (Kelch et al. 2015). These sets of data were used as the gold standard for complete segmentations and thus were used for both training of models and evaluating predictions. Training models and predicting on the same type of data is widely practised and can be seen throughout the literature and will be used to ensure quality of segmentation predictions. An example of this dataset is seen in figure 2.

When working with Deep learning approaches it is important to have large amounts of data for training. There are several publicly available datasets that include images and segmentation masks for training and this project saw the use of the 3-D synthetic dataset as described (Tetteh et al. 2020). An example of this dataset can be seen in figure 4. Other datasets that were taken into consideration included the DRIVE database, as seen in figure 3, containing 2D retinal images and their corresponding segmentation masks. The decision to use the synthetic dataset was made due to two factors. Firstly, due to the sizes of the two sets, the DRIVE database only contains 40 584x565 pixel images and segmentation masks equating to around 13 million pixels for training. Whereas the synthetic dataset contains 136 600x304x325 pixel images equating to around $8x10^9$ pixels, therefore, a much greater amount of data for training of models. Secondly, due to the synthetic dataset being 3-D it is more like the target Popliteal and Inguinal datasets and therefore would make the training more applicable.

Machine Learning Approaches

In machine learning approaches to segmentation, it was observed from the research that appropriate feature extraction was an important factor in achieving effective segmentation. Suitable candidates for feature extraction included Gabor (Pal et al. 2012) and Gaussian (Hassanpour 2014) based filters. This project made use of Gabor filters for feature extraction due to the better results in segmentation seen across these research pieces, with the Gabor method achieving average accuracy of 0.9616 compared with the 0.9396 with the Gaussian method.

Previous studies had been done involving supervised pixel classification models using varying complexity techniques. For prediction of segmentation masks a kNN (Niemeijer 2004) classifier based model and Random Forest (Hassanpour 2014) based classifier model are to be compared. The decision to use these was based upon the issue of working with a limited dataset. These methods in their respective studies showed promising results even when only training with the small DRIVE dataset. By choosing methods that have a track record in achieving high quality segmentation from small datasets, more accurate results can be adapted in the small target Popliteal and Inguinal datasets.

Deep Learning Approach

The project looked to adapt elements of the work performed in the development of 'DeepVesselNet' (Tetteh et al. 2020) as this study also had a focus on vessel segmentation of 3-D volumes. This study along with others researched (Lai 2020) used adapted methods of a CNN called U-Net and achieved excellent segmentation results. This project investigated the use of the standard architecture 3-D U-Net model to perform vessel segmentation. To meet the large data needs of CNNs comparisons between methods to deal with limited datasets were made.

Taking inspiration from a study surveying techniques of improving deep learning networks (Shorten et al. 2019), this project incorporated image augmentation techniques upon the Inguinal and Popliteal datasets to increase the amount of useful data for training the U-Net model. The synthetic dataset was also used as a 'pretraining' dataset for a separate U-Net model, then the Popliteal and Inguinal datasets used to further train this model. To save time with the training of models, down-sampling of the training images was utilised and to ensure the effects of down-sampling did not decrease the quality of the segmentations, a control U-Net model was produced from training on the raw Popliteal and Inguinal datasets only with no down-sampling.

Evaluation

The trained models as described before were then used to make predictions of segmentation masks. The model would be passed a 3-D image from either the Popliteal or Inguinal dataset and would pass out a corresponding prediction of the segmentation mask. One image from the Popliteal (POP5) and one image from the Inguinal (ING3) were reserved for testing only and not included in the training sets. The decision on POP5's exclusion from the training set was due to it being much fainter than others in the datasets with lower pixel values in the vessel walls. Therefore, it could be used as a good comparison to see if models were able to predict a wider range of possible cases. ING3 on the other hand, is more like the rest of the dataset and so could be used as a control.

The evaluation of predictions would be measured in two ways. To ensure high quality segmentation, the prediction mask was compared with the ground truth segmentation mask provided in the Popliteal and Inguinal datasets. The comparison between the two masks was performed by calculating the DICE coefficient between the

overall of the segmentations and at times between individual slides of the masks. The Dice coefficient was chosen due to its presence in multiple studies; therefore, comparisons could be easily made to other works in the literature. The time for predictions was also measured to ensure the aim of achieving faster segmentations was met, however, the time for training models was not measured as it would not be expected for the user to have to train any model.

Application of Chosen Approaches

All code in this study was written in Python as it has multiple extremely useful libraries when dealing with both image processing and AI training and predicting. The Google Colaboratory development environment was used with access to a 'Colab Pro' account due to the provision of powerful computational resources including 26GB RAM and 16GB Video Memory which are vital in training and prediction with AI models. The FIJI environment from ImageJ was used for any image viewing or preparation. The items listed below include the scripts generated to achieve successful segmentation following the different methods outlined in the previous section. The details for these scripts can be seen in the appendix.

1. Feature_Extract_Gabor.py
2. Feature_Extract_Unet.py
3. Train_Model_Classifiers.py
4. Train_3D_Unet.py
5. Classifier_Predict.py
6. Unet_Predict.py

Data Preparation

To meet the objective of producing a solution relatable to the current manual tools in the pipeline, all images were of the TIFF file format and 8-bit grayscale. This matches the input format of the tools by Dr Gib Bogle (Kelch et al. 2015). For the Popliteal and Inguinal datasets this involved a simple tool in the FIJI environment that took an input 16-bit file and converted it to an 8-bit file. The synthetic dataset needed to first be converted from the NII file format to TIFF and then from a 16-bit RBG image into an 8-bit grayscale image.

Machine Learning Methods

The first stage of the machine learning methods used in this project was feature extraction. Both the kNN and Random Forest classifiers require labelled data as they are considered supervised learning algorithms. As outlined in the previous section, Gabor filters were used to generate feature banks for individual pixels. OpenCV was used extensively when generating Gabor filters as it contains useful methods for generating Gabor kernels. Images were also read in using OpenCV as this ensured filtering could be properly applied from the Gabor filters.

An iterative function was produced that read through each TIFF file in a target directory and its respective segmentation mask. The 3-D TIFF volumes were accessed in a slice-by-slice manner and for each pixel in the 2-D slice, a series of data points were generated and stored in a Pandas Dataframe. Pandas is a useful data analysis library and the use of Dataframe functionality was due to the ease of specifying data to be trained when handling pixels with multiple features. For each pixel in the image, the original pixel value was entered into the Dataframe, the same pixel location data from the segmentation mask was also added to the Dataframe determining whether this pixel would be considered in the vessel wall structure or not. A series of 8 Gabor filter values were also generated for each pixel by varying the input parameters for the OpenCV Gabor filter method. Images were resized using tools within the OpenCV library to ensure faster training of models.

The number of Gabor features to be generated was decided upon due to two separate factors. The first being due to time constraints as each iteration through the function would involve generating multiple features per pixel and slow down the feature extraction when using larger numbers of Gabor features. Also, when using a larger number of feature banks with lower values for Gabor input parameters, such as theta =0, it was observed that most Gabor feature values would equal 0. However, for higher values of input parameters, such as theta = ¼*pi, more variable values for the Gabor features were observed. By choosing a limited number of appropriate input parameters that produce variable feature values, the hope was to reduce the number of Gabor features generated without affecting their role in the training of the classifier models as it is the variance in the feature values for each pixel that the model is being trained to predict.

An example of a Dataframe containing training data can be seen in Figure 5 and the script for generating the set in Figure 6.

```
         Original Pix V  Final Pix V  Gabor1  ...  Gabor6  Gabor7  Gabor8
0                    77            0       0  ...      61     121     121
1                    88            0       0  ...      49     121     121
2                    88            0       0  ...      80     147     147
3                   133            0       0  ...      36     152     152
4                    92            0       0  ...      67     139     139
...                 ...          ...     ...  ...     ...     ...     ...
9154555              89            0       0  ...      65     136     136
9154556             112            0       0  ...      54     147     147
9154557             105            0       0  ...      66     150     150
9154558              97            0       0  ...      54     133     133
9154559              76            0       0  ...      71     128     128

[9154560 rows x 10 columns]
```

Figure 5 - Pixel Feature bank for model training

```
def feature_extract(raw_path, seg_path):
  df = pd.DataFrame()
  dimensions = (64,64)

  for file in tqdm((glob.glob(raw_path))):
    img = cv2.imreadmulti(file)
    segImg = cv2.imreadmulti(seg_path + ntpath.basename(file))

    #Loops through each 2D slice of 3D volume
    for n in range(len(img[1])):
      frame = pd.DataFrame()

      #resize to specified dimensions.
      imgre = cv2.resize(img[1][n], dimensions).reshape(-1)
      segImgre = cv2.resize(segImg[1][n], dimensions, interpolation=cv2.INTER_NEAREST).reshape(-1)

      #Read in pixel values from image and corresponding segmentation mask
      frame['Original Pix V'] = imgre
      frame['Final Pix V'] = segImgre

      #Generate Gabor features
      num = 1
      theta = 1 / 4. * np.pi
      for sigma in range(1,2):
        for lamda in np.arange(0, np.pi, np.pi / 4):
            for gamma in (0.05, 0.05):
                gabor_label = 'Gabor' + str(num)
                ksize=3
                kernel = cv2.getGaborKernel((ksize, ksize), sigma, theta, lamda, gamma, 0, ktype=cv2.CV_32F)
                #Filter the image and add values to a new column
                fimg = cv2.filter2D(imgre, cv2.CV_8UC3, kernel)
                filtered_img = fimg.reshape(-1)
                #Labels columns as Gabor1, Gabor2, etc.
                frame[gabor_label] = filtered_img
                #Increment for gabor column name
                num += 1

      #Add dataframe from individual 2D slice to dataframe containing the whole 3D volume
      df = df.append(frame, ignore_index = True)
```

Figure 6 - Method to generate training data for machine learning models including Gabor filtering

To train the models and make predictions t he machine learning library scikit-learn was used as it contains preloaded functions for defining both the kNN and Random Forest classifiers, as well as a useful method of splitting data into test and train sets called 'train_test_split'. The use of this data splitting method makes the validation steps easier to perform. The Dataframe containing the Gabor features and pixel values were split into dependant and independent variables for classification. The dependant variable referring to the binary segmentation mask pixel value, which would be either 0 if it was considered background or 255 if it was considered in the vessel wall structure. The independent variables were the original pixel value in the unsegmented image and its corresponding Gabor filter values. These values were then fit to the model in the same way for either classifier. An example from the kNN model training is seen in Figure 7.

```
from sklearn.model_selection import train_test_split
from sklearn.neighbors import KNeighborsClassifier

def train_knn(X,Y):
  k=2
  X_train, X_test, Y_train, Y_test = train_test_split(X,Y,test_size=0.4, random_state=20)
  knn_model = KNeighborsClassifier(n_neighbors=k)
  knn_model.fit(X_train, Y_train)
```

Figure 7 - kNN training function utilising sklearn

The model was then saved and could be loaded at any point through the Pickle library to make predictions of segmentation masks. To make a prediction on one of the images from the Popliteal or Inguinal sets, an adapted version of the 'feature_extract' method in figure 6 was used that generated only the independent pixel value and Gabor feature variables for the image to be segmented. A prediction using these variables was made on a 2D slice by slice basis with each predicted mask then being appended to a list to contain the full 3D segmentation volume prediction. The prediction method used can be seen in Figure 8.

```
load_model = pickle.load(open('/content/drive/MyDrive/masters_diss_data/Models/knn_model_v2', 'rb'))
result = load_model.predict(X)
```

```
[ ] for i in tqdm(range(len(sk3dimg[1]))):
      X = feature_extraction(sk3dimg[1][i])
      result = load_model.predict(X)
```

Figure 8 - Model prediction for kNN or Random Forest

Deep Learning Methods

The 3-D Unet models produced were adapted from work done in the 'DeepVesselNet' study (Tetteh et al. 2020). For each volume in the training set, extraction of non-overlapping 64x64x64 sub-volumes was completed. These sub-volumes were then passed into the models for training. This was achieved using the Patchify library. For the Patchify library to function correctly the volume being split's height, width and depth values would need to be exactly divisible by 64. A padding algorithm was designed that would take an input image, calculate the nearest multiple of 64 above the individual dimensions. It would then calculate the difference (n) between the nearest multiple and the dimension and then pad the training image along that dimension with pixels of value 0 n times. For example, an input volume of shape 300x940x550 would be padded to a shape of 320x960x576. The adapted method can be seen in Figure 9.

```
def pad_img(img):
  #defines dimensions of 64 times tables as max_dimensions * 64
  max_dimensions = 50
  sixfour_times = np.zeros(max_dimensions)

  #populate 64 times tables, used for test image padding
  for i in range(len(sixfour_times)):
    sixfour_times[i] = (64 * i) + 64

  #Find nearest value in 64 times tables, used for padding
  original_x = img.shape[2]
  original_y = img.shape[1]
  original_z = img.shape[0]

  nearest_x = int(min([v for v in sixfour_times if v >= img.shape[2]]))
  nearest_y = int(min([v for v in sixfour_times if v >= img.shape[1]]))
  nearest_z = int(min([v for v in sixfour_times if v >= img.shape[0]]))

  #add the padding to the dimensions to make it fit the model
  x_difference = nearest_x - original_x
  y_difference = nearest_y - original_y
  z_difference = nearest_z - original_z

  raw_pad = np.pad(img, [(0,z_difference), (0,y_difference), (0,x_difference)])
  seg_pad = np.pad(img, [(0,z_difference), (0,y_difference), (0,x_difference)])

  return raw_pad, seg_pad
```

Figure 9 - Image Padding Algorithm

The 3-D Unet model was compiled using the segmentation models library and is built upon the Ronneberger development of the UNet architecture (Ronneberger et al. 2015). In summary this model consists of two parts. The contracting path which follows a typical convolutional architecture of 3x3x3 convolutions with ReLU activation and 2x2x2 max

pooling operations. Also, there is an expanding path which utilises upsampling and concatenation layers. A summary of the model can be seen in the appendix containing supplementary material (Appendix 1).

Image data was read and resized through the skimage library. This library contains methods for resizing that can easily handle resizing of binary data. When resizing the segmentation mask, it was crucial that the correct interpolation type was used to preserve the original values. The 'order=0' parameter in the resizing, as seen in Figure 10, meant that the interpolation type was set to nearest neighbour and original values were not changed. The images for training when using the Popliteal and Inguinal sets were resized to 256x256 as this meant there were still around 20 million pixels per file for an image of 300 slide depth. The synthetic dataset images were resized to 64x64 as this meant there were still 2.5 million pixels per file for an image of 600 slides.

```
def data_prep(raw_path, seg_path):
  n=0
  dim=256
  raw_out = np.empty
  seg_out = np.empty
  for file in glob.glob(raw_path):
    seed = random.randint(1,100)
    raw_pop = io.imread(file)
    seg_pop = io.imread(seg_path + ntpath.basename(file))
    raw_vess = np.zeros((raw_pop.shape[0], dim, dim),dtype='uint8')
    seg_vess = np.zeros((raw_pop.shape[0], dim, dim),dtype='uint8')
    for i in tqdm(range(raw_pop.shape[0])):
      raw_vess[i] = resize(raw_pop[i], (dim,dim), preserve_range=True, anti_aliasing=True)
      raw_vess[i] = augment(raw_vess[i], seed)
      seg_vess[i] = resize(seg_pop[i], (dim,dim), preserve_range=True, anti_aliasing=False, order=0)
      seg_vess[i] = augment(seg_vess[i], seed)
    if n == 0:
      raw_out = raw_vess
      seg_out = label_gen(seg_vess)
      n+=1
    else:
      raw_out = np.concatenate([raw_out, raw_vess])
      seg_out = np.concatenate([seg_out, label_gen(seg_vess)])
      n+=1

  np.save('/content/drive/MyDrive/masters_diss_data/cardiff_lab_tiff_files/pop_train_np_arr/3DUnet_Resnet_256x256_raw.npy', raw_out)
  np.save('/content/drive/MyDrive/masters_diss_data/cardiff_lab_tiff_files/pop_train_np_arr/3DUnet_Resnet_256x256_seg.npy', seg_out)
```

Figure 10 - Data preperation method for UNet model

An important step in producing an effective deep learning models is through pre-processing of data (Shorten et al. 2019). To ensure accurate prediction it was crucial that the model training data be prepared correctly. The 3-D Unet model developed required labelled data from the segmentation masks used in training. The 'LabelEncoder' included within the sklearn pre-processing library was used to ensure segmentation classification could be understood by the Unet model. The method developed to encode the segmentation data is seen in Figure 11.

```python
from sklearn.preprocessing import LabelEncoder

def label_gen(seg):
  labelencoder = LabelEncoder()
  n,h,w = seg.shape
  seg_label = np.empty(0)
  for i in tqdm(range(seg.shape[0])):
    seg_temp = seg[i].reshape(-1,1)
    seg_temp = labelencoder.fit_transform(seg_temp)
    seg_label = np.append(seg_label, seg_temp)
    #print(seg_label.shape)

  seg_label = seg_label.reshape(n,h,w)
  return seg_label
```

Figure 11 - Label Encoder method

Another measure to ensure accurate segmentation data would be to increase the size of the training set by performing data augmentation (Shorten et al. 2019). A number of augmentation techniques were used to generate more data from the Popliteal and Inguinal datasets. These include rotation of the image; horizontally and vertically flipping the image; vertically and horizontally translating the image. A series of methods as seen in Figure 12 were designed to handle the augmentation of image data. These implementations were achieved using the numpy library, scipy library and random library. The use of seed in the random library meant that equal augmentations could be made to the target image data and its corresponding segmentation mask.

```python
from scipy.ndimage import rotate
import random

def rotation(image, seed):
    random.seed(seed)
    angle= random.randint(-180,180)
    r_img = rotate(image, angle, mode='reflect', reshape=False, order=0)
    return r_img

def h_flip(image):
    hflip = np.fliplr(image)
    return  hflip

def v_flip(image):
    vflip= np.flipud(image)
    return vflip

def v_translate(image, seed):
    random.seed(seed)
    no_pix = random.randint(-128,128)
    vtrans = np.roll(image, no_pix, axis=0)
    return vtrans

def h_translate(image, seed):
    random.seed(seed)
    no_pix = random.randint(-128,128)
    htrans = np.roll(image, no_pix, axis=1)
    return htrans

def augment(image, seed):
    image = v_flip(image)
    image = h_flip(image)
    image = v_translate(image, seed)
    image = h_translate(image, seed)
    image = rotation(image, seed)
    return image
```

Figure 12 - Augmentation methods

A series of different Unet models were trained in this project for comparison of results.

1. A control Unet model that was trained on the raw Popliteal and Inguinal datasets, this was neither down-sampled nor augmented.
2. A Unet model trained on the Popliteal and Inguinal datasets that had been down-sampled to contain 256x256 slices.
3. A Unet model trained on the Popliteal and Inguinal datasets that had been down-sampled to contain 256x256 slices and trained with 2x augmented further Popliteal and Inguinal datasets.
4. A Unet model pretrained on the down-sampled synthetic dataset containing 64x64 slices, and further trained with down-sampled and 2x augmented Popliteal and Inguinal datasets.
5. A Unet model utilising the pretrained ImagNet vgg16 (Shorten et al. 2019) weights with further training of Popliteal and Inguinal down-sampled and 2x augmented datasets

In the segmentation models library, there were metrics included for validation when training. To aid with deciding how long to train models, the Intersection Of Union (IOU) was tracked during training. The IOU is considered as a statistic for comparing the similarity of sample sets. As can be seen in Figure 13, after a certain number of epochs of training the validation IOU score can be seen to plateau. This plateauing effect will be observed for each of the models discussed previously to determine the length of their training. When fitting a model, a call-back function from the Keras library called 'EarlyStopping' was implemented to stop training when this plateau of IOU score was seen.

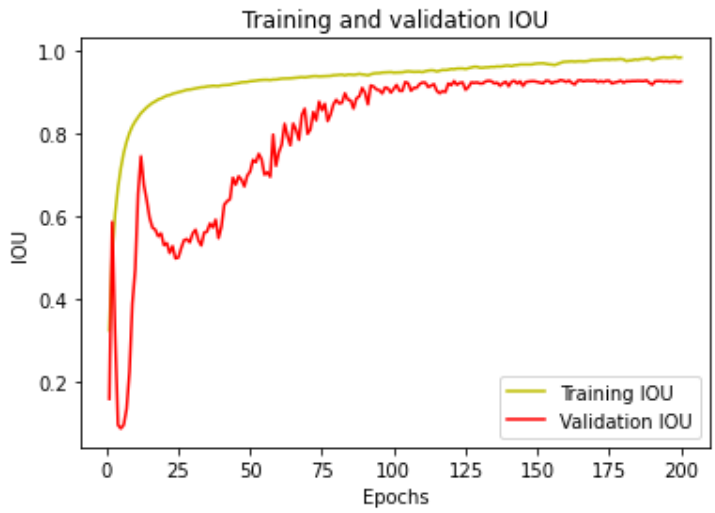

Figure 13 - Training and validation IOU for 3D Unet preliminary training

For predictions to be made on the test volumes, the process described before including padding of the image and splitting into 64x64x64 sub-volumes would be applied before prediction. This is due to the Unet model only accepting input of the same shape as the training dataset.

Evaluation

For both the machine learning models and deep learning models described a method to calculate the dice coefficient between a prediction segmentation and the ground truth segmentation mask was implemented from stackoverflow. The method takes two input image arrays and a value determining the truth value to be tested for. This was used in both whole volume comparisons as well as for slice-by-slice comparisons.

```
# code adapted from stack overflow example
# https://stackoverflow.com/questions/31273652/how-to-calculate-dice-coefficient-for-measuring-accuracy-of-image-segmentation-i
# Author : Dennis Beroetta
# Year : 2020
def dice(seg, true, k = 1):
    intersection = np.sum(seg[true==k]) * 2.0
    dice = intersection / (np.sum(seg) + np.sum(true))
    return dice
```

Figure 14 - Dice Coefficient calculation method

Results and Analysis

This section focuses on the comparison of the trained models produced in this project. An input image was fed into a model for prediction after the relevant feature extraction methods outlined in the previous section. The model produced a predicted segmentation mask for the image, and this was compared against the ground truth manual segmentation provided by Dr Milutinovic. These comparisons will be evaluated against the aims and objectives of this project using the Dice coefficient for image comparison and the time taken to perform a prediction over the given 3-D volume.

The use of the kNN model for prediction was discarded from the project. This was due to the predictions being produced not meeting the main aim of this project. The predictions would take upwards of 30+ minutes per slice when testing on the Popliteal LN image 5 (POP5). With an image depth of around 300 slices, this amounts to a time much greater than required in the current manual pipeline.

The Random Forest model produced more promising results with a volumetric prediction for POP5 taking only as long as 15 minutes. This would meet the objective of creating a fast segmentation model. However, the accuracy of the prediction when compared with the POP5 segmentation mask only scores a 0.248 Dice coefficient. Figure 15 shows the individual elements of the prediction.

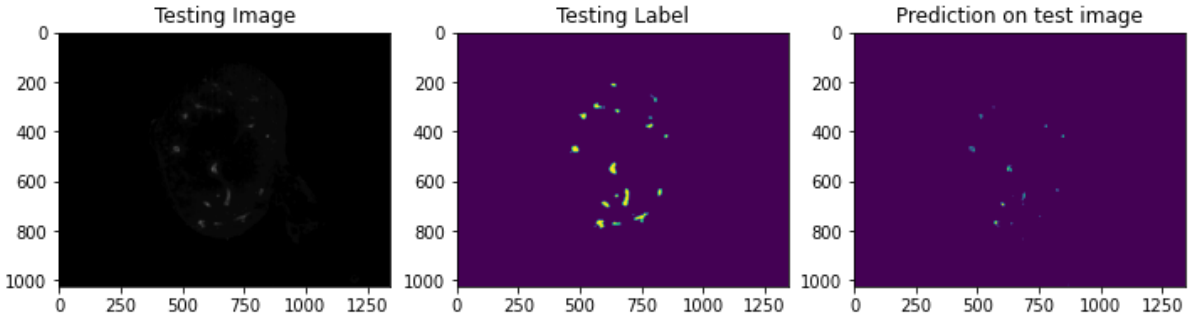

Figure 15 - 2-D slices throgh (Left) original LSFM image, (Middle) ground truth segmentation and (Right) Prediction of segmentation mask. Random Forest Model.

The first deep learning model results to be shown are from the 3-D Unet control group. The training dataset consisted of 64x64x64 sub-volumes of the entire Popliteal and Inguinal 3-D volumes. The accuracy of the prediction when compared with the POP5 segmentation mask scores a 0.483 Dice coefficient. Figure 16 shows the individual elements of the prediction. The time needed to complete the predictions using this 3-D Unet methodology is significantly lower, at only 2 minutes 44 seconds for POP5. Figure 17 shows the prediction method utilising the tqdm library to measure the time taken for the 3-D volume prediction to be made.

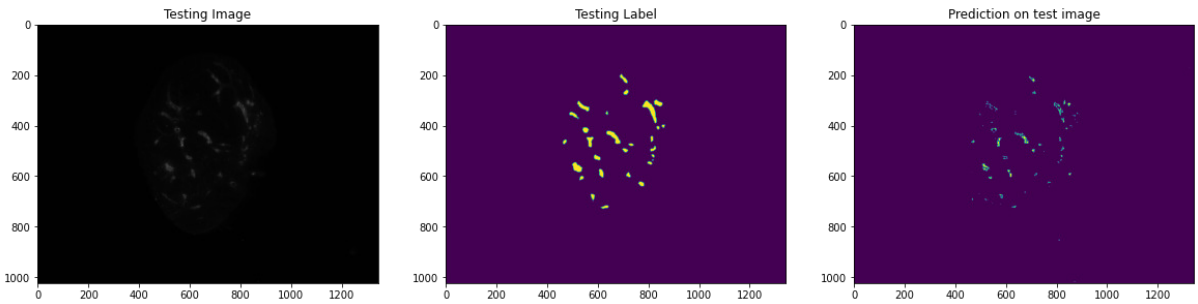

Figure 16 - 2-D slices throgh (Left) original LSFM image, (Middle) ground truth segmentation and (Right) Prediction of segmentation mask. 3-D Unet control.

```
predictions = []
for i in tqdm(range(patch_test_ready.shape[0])):
  tester = patch_test_ready[i]
  tester_input=np.expand_dims(tester, 0)
  test_pred = my_model.predict(tester_input)
  test_prediction = np.argmax(test_pred, axis=4)[0,:,:,:]
  predictions.append(test_prediction)
```

```
100%|██████████| 1680/1680 [02:44<00:00, 10.20it/s]
```

Figure 17 - 3-D Unet prediction method showing total time for 3-D volume prediction

The second deep learning model results to be shown are from 3-D Unet down-sampled dataset. This training dataset consisted of the Popliteal and Inguinal 3-D volumes with their width and height dimensions resized to 256x256 while keeping the same depth of slices. The accuracy of the prediction when compared with the POP5 segmentation mask scores a 0.682 Dice coefficient. Figure 18 shows the individual elements of the prediction. When comparing these results with the 3-D Unet control group, it can be observed that even though down-sampling is used, a much higher quality segmentation is produced. Due to the preparation of this dataset in the label generator algorithm (Figure 11) taking only 8 minutes per image, compared with 40 minutes per image for the control group, the decision to use only down-sampled data for training of other models was made.

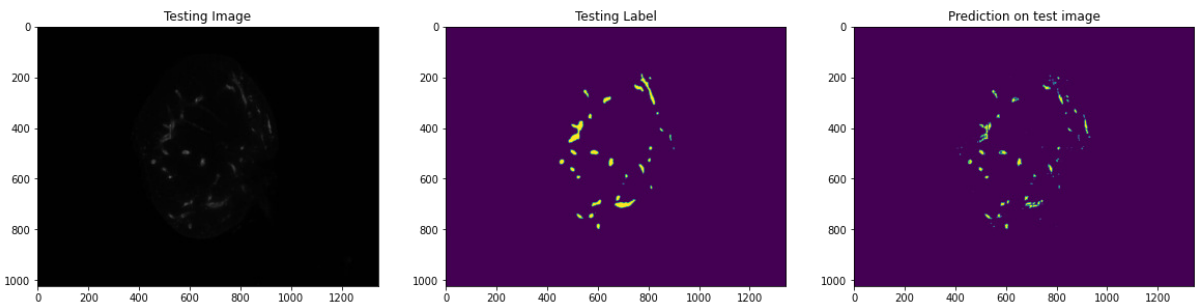

Figure 18 - 2-D slices throgh (Left) original LSFM image, (Middle) ground truth segmentation and (Right) Prediction of segmentation mask. 3-D Unet downsampled.

The third deep learning model results to be shown are from the 3-D Unet down-sampled dataset with 2x augmentation of Popliteal and Inguinal datasets. The augmentation follows the functions outlined in figure 12 where a sample is flipped horizontally and vertically, then a random seed generates a rotation degree and horizontal and vertical translation to produce a more varied set of data. The accuracy of the prediction when compared with the POP5 segmentation mask scores a 0.717 Dice coefficient. Figure 19 shows the individual elements of the prediction. This shows the positive effects of augmentation on producing more reliable prediction as this score is even higher than the just down-sampled model.

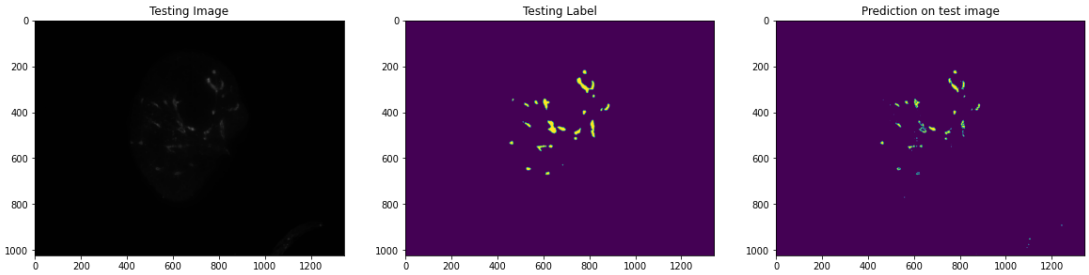

Figure 19 - 2-D slices throgh (Left) original LSFM image, (Middle) ground truth segmentation and (Right) Prediction of segmentation mask. 3-D Unet Downsampled and Augmented.

The fourth deep learning results to be shown are from the 3-D Unet model pretrained on the down-sampled synthetic dataset containing 64x64 slices, and further trained with down-sampled and 2x augmented Popliteal and Inguinal datasets. The accuracy of the prediction when compared with the POP5 segmentation mask scored a 0.294 Dice coefficient. Figure 20 shows the individual elements of the prediction. This score is much lower than the results reported in the 'DeepVesselNet' study in which a Unet finetuned model achieved a dice score of 0.844. However, this is most likely due to the amount of training the models have received as in this study where there were 1200 epochs of training compared to the average of around 100-200 for each of this project's adapted models (Tetteh et al.2020). This highlights the need for suitable iterations of training on a dataset.

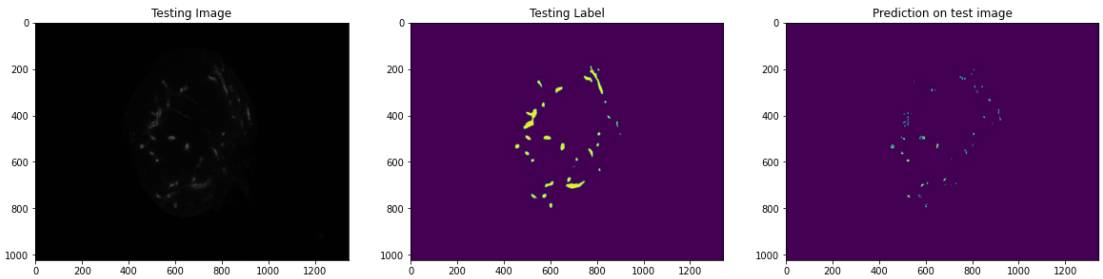

Figure 20 - 2-D slices throgh (Left) original LSFM image, (Middle) ground truth segmentation and (Right) Prediction of segmentation mask. 3-D Unet synthetic pretraining.

The final deep learning results are from the 3-D Unet model that makes use of the publicly available ImageNet weights discovered in the research (Shorten et al. 2019) combined with further training using the Popliteal and Inguinal down-sampled and 2x augmented datasets. The accuracy of the prediction when compared with the POP5 segmentation mask scored a 0.662 Dice coefficient. Figure 21 shows the individual elements of the prediction. Table 1 contains the results for all the models' predictions and their corresponding Figure number.

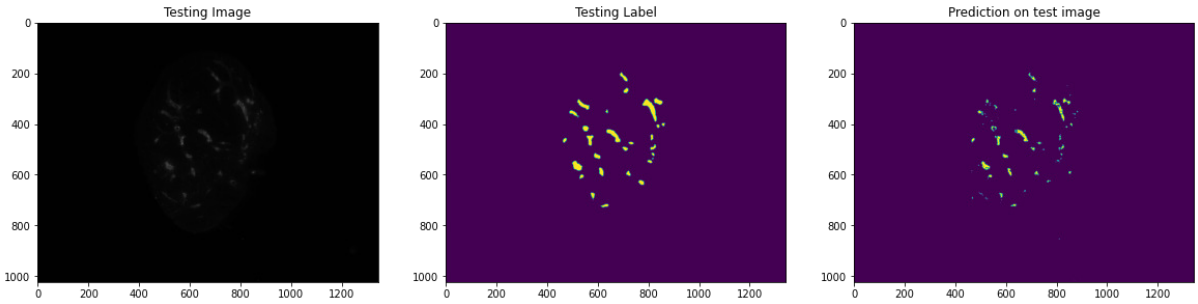

Figure 21 - 2-D slices throgh (Left) original LSFM image, (Middle) ground truth segmentation and (Right) Prediction of segmentation mask. 3-D Unet Imagenet pretraining.

Table 1 - Overview of performance of adapted models

Model Description	Figure	Dice Coefficient
3D Unet Control	16	0.483
3D Unet Down-sampled	18	0.682
3D Unet Down-sampled + Augmented	19	0.717
3D Unet Synthetic pretrain	20	0.294
3D Unet imagenet pretrain	21	0.662
Random Forest	15	0.248

Limitations

One of the main limitations to the segmentation quality was due to the staining method used in the medical research. Figure 22 shows a slice through a prediction made on Popliteal 4 (POP4) which is an image part of the training set. The dice coefficient for this individual slice prediction is 0.896. Figure 23 shows a slice through the same prediction on POP4 at a

slice that contains some LN tissue that has been stained accidentally. The dice coefficient for this slice is only 0.399. As the model is based solely on the intensity of pixels within the image, it is unable to distinguish the difference between when a blood vessel is stained and when tissue is stained. This makes the overall prediction of a volume worse, even when there are excellent predictions in areas.

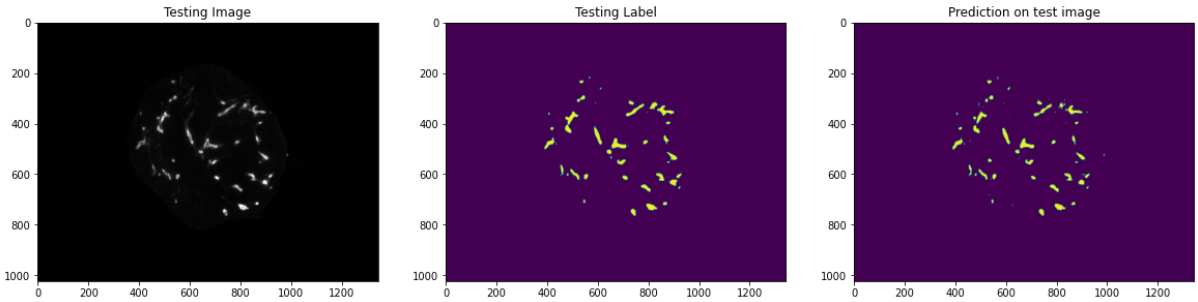

Figure 22 – Segmentation Prediction on POP4

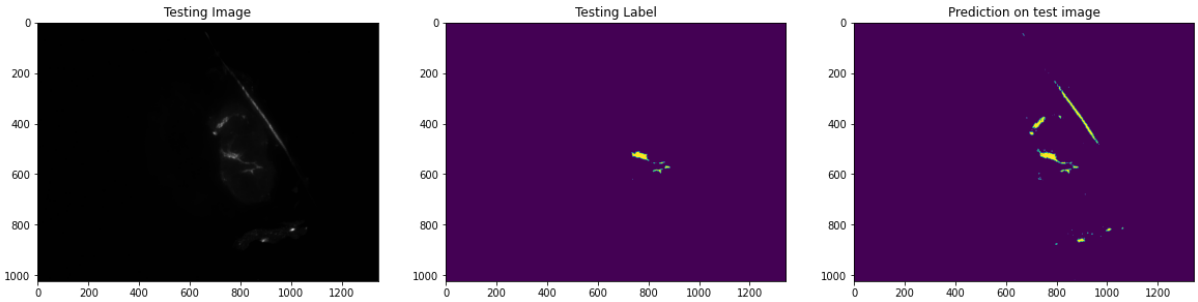

Figure 23 - Segmentation Prediction on POP4 showing tissue staining

Another main limitation to the results produced in this dissertation project was related to the researcher's prior knowledge of the fields in both artificial intelligence and image processing. This meant that a large sum of time in this project was dedicated to learning and familiarising with the basic concepts and libraries of these topics. With a greater amount of time on the project, perhaps better models and segmentations would have been achieved.

Conclusion

Meeting of objectives

The first objective was to Identify a quantitative measure for use in comparison to help maintain the high-quality segmentations achieved in the current 'Vessel Tools' pipeline. The use of the Dice coefficient identified in the literature was used in comparisons between predicted segmentation masks and the ground truth segmentations produced from the current tools. In the results however, the highest Dice coefficient score when predicting on an unseen image (POP5) was only 0.717 and more work would need to be done to improve the effectiveness of the models.

The second objective was to improve the speed of the segmentation by making a single step process. As seen in the results, the predictions from both deep learning and machine learning based approaches provided a fast segmentation. 15 minutes with the Random Forest Classifier and 3 minutes with the deep learning models. When compared with the current pipeline that involves a multi-step trial and error based approach that may take hours per image, this objective is seen as met.

The third objective was to ensure the user input for the tool is limited. Further work would need to be done to achieve this as currently the models require feature extraction and data preparation in separate steps. However, this could easily be compiled into a single input image pipeline with more time available.

The fourth objective was to produce a solution that is easily relatable to the current tools and data in use, to enable ease of transfer from the current tools to the new. All image data handled, and predictions produced in this project were of TIFF 8-bit grayscale file format and so could be transferred and used with the current tools. However, as the predictions made in this project were performed in the Google Colab development environment, an issue may arise with the same predictions being performed on in-lab workstations that may have much lower computer specifications such as dedicated GPU memory. Lower specifications may lead to predictions taking much longer or perhaps may not even run.

Several suitable existing frameworks were identified and tested upon in the results section and so the fifth objective of identifying these from related literature is met.

Methodology

As the project progressed it became apparent the project scope was too large. The project looked to compare several different model techniques to try and find an effective solution. However, this led to the research being spread too thin. Lots of time had to be used researching the implementation of multiple different methods and so the training time for models suffered. The project could have instead focused upon one solution, such as 3-D Unet, and attempted to optimise the modelling here. With more training and approaches focused on honing a single model, more accurate segmentations could have been achieved.

From the results seen in this project, down-sampling of images is a useful tool in saving time when training models while also being able to produce high quality segmentations. If I was able to repeat the project, I would use down-sampling in model training from the beginning rather than wasting a lot of time preparing the training of data from whole images.

The methodology used in deciding the number of epochs to train the deep learning models involved utilising a Keras call-back called 'EarlyStopper'. The number of epochs models would be fit on depended on when there was a plateau in the increase of validation IOU. This led to different models receiving a different number of epochs of training. In future when comparing different models in this way, a control number of epochs would be used to ensure this wasn't affecting the quality of the segmentations.

Further work

The choice of testing all models against POP5 was made due to it having a lower overall intensity of staining and hence pixel values in the vessel walls. This allowed for testing against the extremes of possible test cases and showed a worst-case scenario Dice score for the model. Figure 24 shows a prediction made on ING3 by the 3-D Unet down-sampled and augmented model. ING3 was also chosen for testing due to its similarity in intensity of staining to the rest of the training images. This prediction received a Dice score of 0.891 which is much higher than the 0.717 achieved with POP5. This highlights the importance of accounting for differences in pixel intensity within training images. Although augmentation of data produced great results as a pre-processing step. Further work with other pre-processing techniques, such as normalisation or standardisation (Bhandara 2020), that

would look at handling the variation in pixel intensities between different images could be performed.

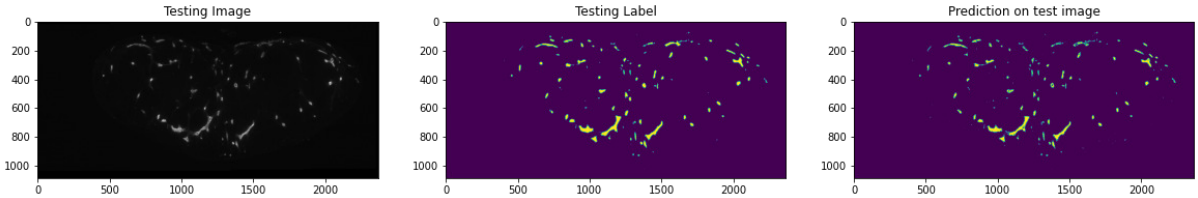

Figure 24 - Prediction on ING3. 3-D Unet Down-sampled + Augmented model.

To meet objective number four several approaches in further work could be adapted. More time on the project would allow for more optimisation of a model that would produce satisfactory vessel segmentations. If the trained model was unable to do this however, the combination of the 'Vessel Tools' used for manual segmentation could be utilised. Research could be done into the effects of using such tools as filler on the prediction masks. This approach would still satisfy speeding up the overall process as the automated model prediction could replace the manual thresholding and smoothing tools. This work would also require investigation into the use of the prediction models outside of the Google Colab environment.

<u>Contributions to existing material</u>

The results from this project contribute evidence to the literature supporting the effectiveness of both down-sampling and augmenting as pre-processing steps for vessel segmentation AI models. This is shown by the best prediction results in this project from the models adapted using these techniques. The research for this project also identified gaps in the literature relating to 3-D segmentation of lymphatic vessel networks. The methodology adapted combined Unet with datasets provided by Dr Milutinovic to add to this lacking area of literature.

Reflection

Over the course of this project, I learned many lessons relating to the fields of AI and image processing as well as learning many lessons about myself and my goals. One of the lessons I learned to do with these topics is the amount of time necessary to complete tasks. The pre-processing of data through augmentation, label encoding and feature extraction etc. took up most of the time in this project. The amounts of training necessary for models to be able to accurately predict also became apparent through this project achieving good results with limited data. This showed to me that even though it is the prediction that is under evaluation, in AI the preparation of data for training is the most vital step.

In the research completed into the literature surrounding this project it became clear that although there was lots of research on vessel segmentation, most of the research was performed on a very limited number of small datasets such as the DRIVE database. The results from this project showed the importance of having data for training that is relevant to the problem being solved. When training data is not representative of the testing data, such as in Figure 20, worse predictions are made. This showed to me the importance of publicly available data and open-source libraries in AI research to develop useful models and beyond.

This research topic required proficient understanding of Python and its libraries. One of my strengths identified during this process was my ability to understand the ways in which Python works and so I was able to utilise several previously unseen external libraries to achieve good results. I also began this project with practically no understanding of AI and image processing. I was able to utilise another strength of mine of being able to learn new topics quickly to adapt and produce a variety of different machine learning models.

In deciding upon the methodology for my research, some good and bad decisions were made. One of the good decisions I made was in deciding to test on the POP5 image data. I made this decision due to the image quality being much fainter than others in the testing dataset. This proved to be a good decision as it showcased the necessity of training on fully representative datasets and revealed areas of further study. I learnt from this that there is always more that can be done in the optimisation of AI models and will hope to bring this forward to further work. A bad decision I made was deciding on the scope of the project in the early stages. By deciding to attempt to implement and compare multiple

methodologies, I believe my segmentation results suffered due to this and so this hindered my project achieving its aims. In the future, I would hope to set my research sights on more focused tasks in the beginning and then expand to other comparisons if time would allow.

As I began this MSc Computing course last year, I thought one of the only disappointing aspects of this course was that it did not contain any modules dedicated to teaching AI and its concepts. This was an area I had always been interested to see if I enjoyed it. This project topic gave me the opportunity to explore AI in a field of research as important as Biomedicine and Cancer Research. Working through all the processes involved with producing models and predictions showed to me how much I would enjoy working in a field such as this, especially when achieving that amazing feeling of producing working predictions.

References

Ager, A., & May, M. J. 2015. *Understanding high endothelial venules: Lessons for cancer immunology*. Oncoimmunology, 4(6), e1008791. https://doi.org/10.1080/2162402X.2015.1008791

Bhandari, A. 2020. *Feature Scaling for Machine Learning: Understanding the Difference Between Normalization vs. Standardization*. Analytics Vidhya. Available at: https://www.analyticsvidhya.com/blog/2020/04/feature-scaling-machine-learning-normalization-standardization/ [Accessed 03/11/2021]

Biomathematics & Statistics Scotland (BioSS). 2021. *Chapter 4: Segmentation.* Available at: https://www.bioss.ac.uk/people/chris/ch4.pdf [Accessed 03/11/2021]

Fu H. et al. .2016. *DeepVessel: Retinal Vessel Segmentation via Deep Learning and Conditional Random Field*. Lecture Notes in Computer Science, vol 9901. Springer, Cham. https://doi.org/10.1007/978-3-319-46723-8_16

Hassanpour, N. 2014. *Retinal Vascular Segmentation using Random Forest.* University of British Columbia. Available at: https://web.unbc.ca/~hassanp/files/Retinal_Vascular_Segmentation_using_Random_Forest.pdf [Accessed 03/11/2021]

Hu, C. et al. 2017. *Brain vascular image segmentation based on fuzzy local information C-means clustering.* Proc. SPIE 10068, Imaging, Manipulation, and Analysis of Biomolecules, Cells, and Tissues XV, 100680Q. https://doi.org/10.1117/12.2251725

Kelch, I. D. et al. 2015. *Organ-wide 3D-imaging and topological analysis of the continuous microvascular network in a murine lymph node.* Scientific reports. England, 5, p. 16534. doi: 10.1038/srep16534.

Lai, P.Y. et al. 2020. *Lymphatic vessel segmentation in optical coherence tomography by adding U-Net-based CNN for artifact minimization*. Biomed. Opt. Express 11, 2679-2693. https://doi.org/10.1364/BOE.389373

Milutinovic, S. 2020. *Investigating the Role of High Endothelial Venules in Cancer Immunity*. Ph.D. Thesis, Cardiff University.

Niemeijer, M. et al. 2004. *Comparative study of retinal vessel segmentation methods on a new publicly available database*. Proc SPIE. 5370. https://doi.org/10.1117/12.535349.

Pal, N. et al. (2012). *Retinal vessel segmentation using Gabor filter and artificial neural network.* Proceedings - 2012 3rd International Conference on Emerging Applications of Information Technology. EAIT 2012. p157-160. https://doi.org/10.1109/EAIT.2012.6407885.

Rice, G. 2021. *Fluoresecent Microscopy: What Is Fluorescent Microscopy?* Available at: https://serc.carleton.edu/microbelife/research_methods/microscopy/fluromic.html [Accessed: 03/11/2021]

Ronneberger, O. et al. 2015. *U-Net: Convolutional Networks for Biomedical Image Segmentation.* BIOSS Centre for Biological Signalling Studies. Available at: https://arxiv.org/pdf/1505.04597.pdf [Accessed: 03/11/21]

Schneider, M. et al. 2012. *Tissue metabolism driven arterial tree generation*. Med. Image Anal. p1397–1414. https://doi.org/10.1016/j.media.2012.04.009

Schneider, M. et al. 2014. *TGIF: topological gap in-fill for vascular networks-a generative physiological modeling approach.* International Conference on Medical Image Computing and Computer-Assisted Intervention, Vol. 17, p89–96. https://doi.org/10.1007/978-3-319-10470-6_12ca

Shorten, C et al. 2019. *A survey on Image Data Augmentation for Deep Learning*. Journal of Big Data. 6. https://doi.org/10.1186/s40537-019-0197-0.

Tetteh, G. et al. 2020. *DeepVesselNet: Vessel Segmentation, Centerline Prediction, and Bifurcation Detection in 3-D Angiographic Volumes.* Frontiers in Neuroscience 14. https://doi.org/P1285. 10.3389/fnins.2020.592352

Zhao, F. et al. 2019. *Segmentation of blood vessels using rule-based and machine-learning-based methods: a review.* Multimedia Systems 25, 109–118. https://doi.org/10.1007/s00530-017-0580-7

Appendices

Appendix 1: Supplementary Materials.

File Name	Description
feature_extract_gabor.py	Extracts gabor features and pixel values for training of Random Forest and kNN.
feature_extract_unet.py	Pre-Processing of data for Unet models
train_model_classifiers.py	Training module for kNN and Random Forest models
train_unet.py	Training module for Unet models
classifier_predict.py	Prediction module for kNN and Random Forest models
unet_predict.py	Prediction module for Unet models
Model_summary.txt	3-D Unet model framework summary including number of trained parameters etc.