FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

Retinal Vessel Segmentation - Project Report

Computer Vision in English (POVa) – 2019/2020

Jakub Svoboda (xsvobo0z) 10. 12. 2019

Introduction

Image segmentation is a process where an input image is processed and analyzed by and algorithm and the pixels of the image are separated into multiple segments. Each segment usually represents a pixels belonging to one specific object in the input picture. In a way, segmentation is can thought of as classification of each pixel, where pixels in the same class share certain characteristics.

With the increase in the amount of publicly available data and the exponential growth of computational capacities, image segmentation has seen application in wide range of fields. In video surveillance it allows for the implementation of more fine-grained surveillance systems which can be highly automated and do not require human monitoring. In self-driving car systems image segmentation can be used as an enhancement of more common object detection algorithms and can provide more robust data to the car's control mechanisms. Image segmentation has also seen large number of practical applications in the medical field. Here the machine learning algorithm can analyze huge amount of data gathered from x-rays, CT scans and magnetic resonance images.

The goal for this project is to implement image segmentation algorithm to segment the vessels of a human retina and evaluate their performance. For this task, I have selected a neural network which I have trained on publicly available datasets and compared the results to more naïve algorithms.

Datasets

The data acquired for this task came from several publicly available datasets. Several factors have been considered during the selection process to ensure viability of given dataset. Firstly, as this project requires makes use of neural network, it was necessary for the dataset to include masks which denote the vessels in human retina. Another factor was the sufficient resolution of the image data. These criteria have excluded several retinal imagery datasets. As the remaining datasets were rather limited in size, it was decided to merge multiple datasets into one. For this task, the DRIVE[1] dataset, the STARE[2] dataset and the CHASE DB1[3] have been utilized.

In order to merge these datasets, certain preprocessing tasks were undertaken to ensure the consistency of the data across the images from varying sources. In the DRIVE dataset, the test section has been deleted as there were no labels provided and therefore the accuracy of this part could not be measured. The CHASE dataset originally contained two sets of masks (from two people each manually annotating the set). As this is redundant, only the first set of label information has been used. Furthermore, images from all three sets have been resized to 1000x1000 pixels.

In total, the final dataset contained a total of 68 images and their labels. 20 of these came from the DRIVE dataset, 28 from the STARE dataset and 20 from the CHASE DB1. For the machine learning algorithm, the dataset has been split into training and testing subparts, with the training set containing 48 images and the testing set containing 20 images.

[1] DRIVE: Digital Retinal Images for Vessel Extraction: https://drive.grand-challenge.org/

[2] STARE: Structured Analysis of the Retina: http://cecas.clemson.edu/~ahoover/stare/

[3] CHASE DB1: https://blogs.kingston.ac.uk/retinal/chasedb1/

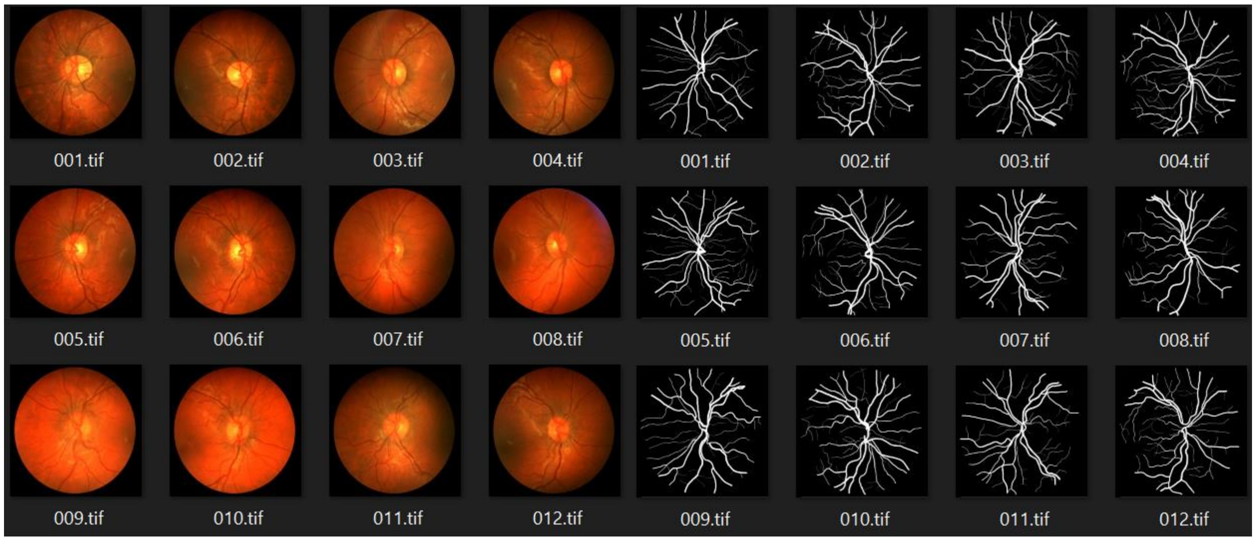

Figure 1: The dataset with the input data on the left and its labels on right. Here the vessels' segments are encoded as white pixels and the background as black.

Implemented Methods

As image segmentation algorithms' performance varies highly for each specific task, it was necessary to implement and test several naïve methods for image segmentation, which will serve as a baseline against which the final neural approach will be compared, with the expected result being that the neural network will outperform these basic algorithms.

For this several image processing methods have been implemented. First a simple thresholding method has been implemented. As expected, the hard-coded threshold proved to be not very robust, as variance in the overall illumination of the image impacts the pixels brightness and might shift them to an improper class.

Additionally, two adaptive threshold methods have been tested out. Adaptive thresholding changes the threshold value dynamically over the whole image and for this reason is much more robust to changing levels of illumination. The threshold value is selected either as the mean value or as the weighted sum of the neighborhood values (gaussian adaptive thresholding).

For the neural network approach a U-Net[4] based architecture has been selected. Other architecture considered was the Mask-RCNN, but this architecture was rejected, due to the U-Net being several times faster and easier to train on small datasets. For this task, an open source implementation[5] has been used as a staring point and the code has been modified to fit the task. Forthermore, the network has been trained on the 48 images from the test subsection of the dataset over 10 epochs. The training was performed on an Google Colab instance.

Evaluation and results

Each algorithm's performance has been measured with the Jaccard Similarity Index (commonly known as the Intersection over Union). The method performs a binary AND and a binary OR function on the result image and the ground truth mask. The number of pixels in the intersection and in the union are then summed

[4] https://arxiv.org/pdf/1505.04597.pdf
[5] https://github.com/zhixuhao/unet

and divided. This simple metric results in range from zero to one, with higher numbers denoting better accuracy.

The results show that in the naïve methods, the basic thresholding method performed worst with the accuracy being only 0.08. For the adaptive thresholding methods, the mean adaptive thresholding reached the accuracy of 0.23 and outperformed the gaussian method which reached 0.21. For the U-Net architecture, the 20 image from the testing subsection of the dataset were passed through the network and the results were evaluated using the same method. On the test dataset, the network reached the accuracy of 0.38.

Overall the results show that the neural network outperforms other methods, which aligns with the expectations as well as with the subjectively perceived differences when comparing the resulting images.

Method:	**Average IoU**:
Thresholding (fixed threshold)	0.08
Adaptive Thresholding (mean)	0.23
Adaptive Thresholding (Gaussian)	0.21
U-Net	0.38

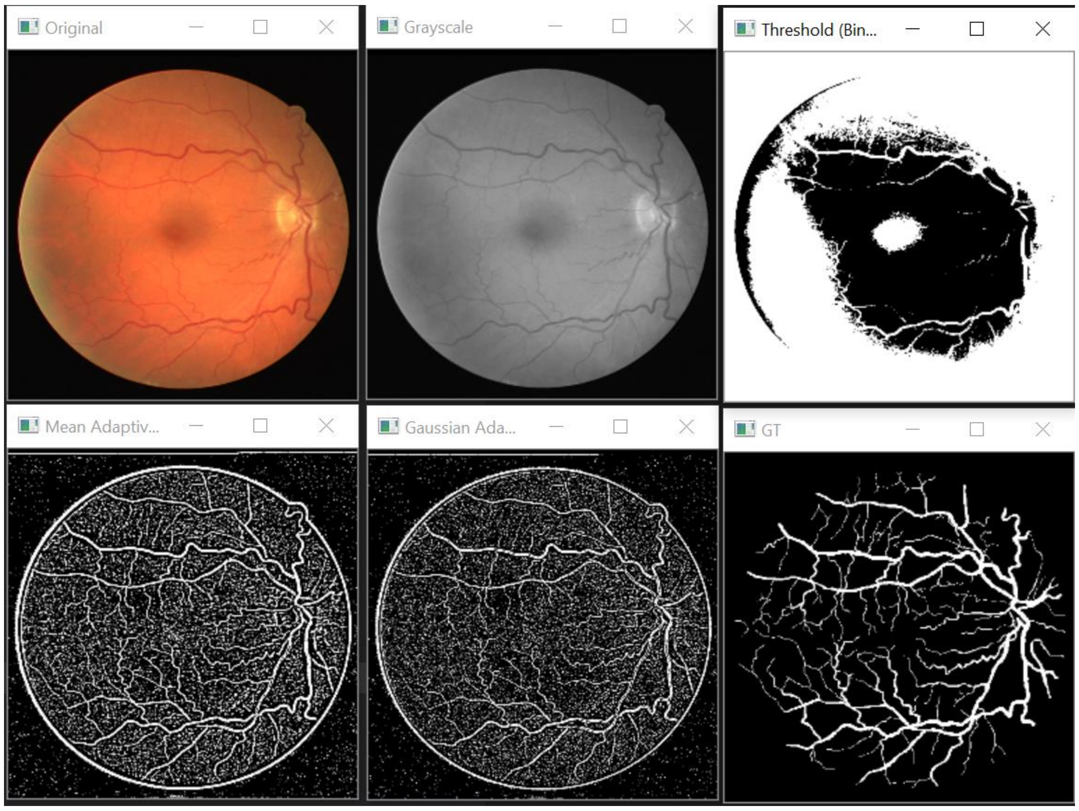

Figure 2: The results of various thresholding methods. Top row from left: The original image, grayscaled image, threshold (128). Bottom row: Mean Adaptive Threshold, Gaussian Adaptive Threshold, the ground truth from the dataset.

Installation instructions

While it is possible to gather the data and source code into workable workspace on your own, it is HIGHLY recommended that you clone the Github repository for this project where all the images are already gathered and preprocessed and the source code is ready to be used. There are also pre-trained weighs attached, which can be used for immediate testing.

```
git clone https://github.com/Jakub-Svoboda/Retinal-Vessel-Segmentation
cd Retinal-Vessel-Segmentation
python3 ./thresholding.py
python3 ./unet/main.py
```

This code has been tested and implemented for Tensorflow 1.15.0, scikit-image 0.16.2, Opencv 4.1.2.30, Keras 2.2.5

Conclusion

Several methods for pixel wise image segmentation have been implemented and tested on the task of segmenting blood vessels in human retinal images. The naïve thresholding methods were outperformed by adaptive thresholding by a significant margin. One U-Net based neural network has also been trained and its performance was measured at about double of the best thresholding algorithm. The measured accuracies fit the predicted performic expectations.