Diabetic Retinopathy segmentation on digital fundus images

Yiwen Xu
The University of New South Wales
Sydney, Australia
z5224340@unsw.edu.au

Chengze Du
The University of New South Wales
Sydney, Australia
z5140893@unsw.edu.au

Sheng Du
The University of New South Wales
Sydney, Australia
z5171466@unsw.edu.au

Yichen Liu
The University of New South Wales
Sydney, Australia
z5233576@unsw.edu.au

Yiren Hu
The University of New South Wales
Sydney, Australia
z5148244@unsw.edu.au

***Abstract* —— Examinations of the retina and its associated parts, such as optic-nerve, cup, vessel and disc, are quite common in Ophthalmology to diagnose the retinal abnormalities. It is significant for experts to identify disease tissue before full medical check and retinal segmentation is a key step towards the accurate examinations. Recently, some deep learning based segmentation methods have reached the state of the art(SOTA) performance. We believed that there have been some potentials that some well-designed algorithm could assist human experts to give a better diagnosis. In this report, we explored several methods that were proposed by other researchers and did a comparison between them.**

Keywords — retinal image segmentation, deep learning, machine Learning, Alternating Sequential Filter, computer vision

I. Introduction

Diabetic Retinopathy (DR) is the most common cause of avoidable vision loss, predominantly affecting the working-age population across the globe. DR could be early diagnosed by visually inspecting retinal fundus images for the presence of retinal lesions like microaneurysms (MAs), hemorrhages (HEs), soft exudates (SEs) and hard exudates (EXs).

In this project, two tasks are given. The first task is to do lesion segmentations which are associated with DR. The second one is doing retinal blood vessel segmentation.

Different from our lab question which was also a segmentation problem, this project is actually requiring a pixel-wise segmentation. As a result, we should first stay away from some famous models such as LeNet, AlexNet or VGG because they are all doing segmentation for the whole image. They would not work properly in this specific project.

The provided datasets are Indian Diabetic Retinopathy Image Dataset (IDRiD) for project 1 and DRIVE database for project 2. We used both traditional methods and deep learning based methods during the project. More specifically, for task 1, the methods we used were U-Net, Region Growing Segmentation and Mask R-CNN. For task 2, we used Intensity Enhancement with Alternating Sequential Filter and ResNet18 embedded in U-net to do vessel segmentation.

II. Literature Survey

We first explored all 8 papers that were given for group tasks. Among those 8 papers, "IDRiD Diabetic Retinopathy Segmentation and Grading Challenge" [1] is the most comprehensive one towards retinopathy segmentation challenge. Methods mentioned in this essay are mainly based on deep learning.

Meanwhile, other papers provided methods with seeming simplicity, for example, "Multiscale Blood Vessel Segmentation in Retinal Fundus Images." [2] Since method proposed in this paper only had three steps, this is our first try. It ended up working pretty poor (using eigenvalues of neighbours' hessian matrix). To be honest, we found that most of the methods proposed in those given papers were performing badly. Thus we turned our sights to Internet. After doing some research, we chose and compared several methods.

The first idea we decided to try was U-net (and its variants) since it's SOTA in bio-medical image processing. After getting some decent outputs from U-net, we believed that we would need at least one traditional method in each task. Region growing segmentation [5] and alternating sequential Filtering [21][22] are our choices. In the end, we add MASK R-CNN method proposed in [3] as an addition to our project.

Beyond these, we certainly gained a lot of help from Google Scholar. It's somewhat hard to track all of our search results but we append some sites which gave us the best help at the end of the reports. They are basically some code templates that can be used to understand and construct a U-net.[26]

III. Method

A. Task 1: U-net

1) Model introduction: U-net was first introduced by Ronneberger, Olaf, et al.[8]. The U-net architecture looks exactly like an uppercase U. It does downsampling at first and then upsampling. By convention, we called its left hand

side as encoding layers and right hand side as decoding layers.

Unlike some CNN architectures with fully connected layers at the end of the networks, U-net is a fully convolutional network(FCN) which is able to do pixel-wise segmentations. Concatenating counterparts in encoding and decoding layers is another distinct feature of U-net. For the time being, U-net has been proved very effective in lots of bio-medical image segmentation tasks.

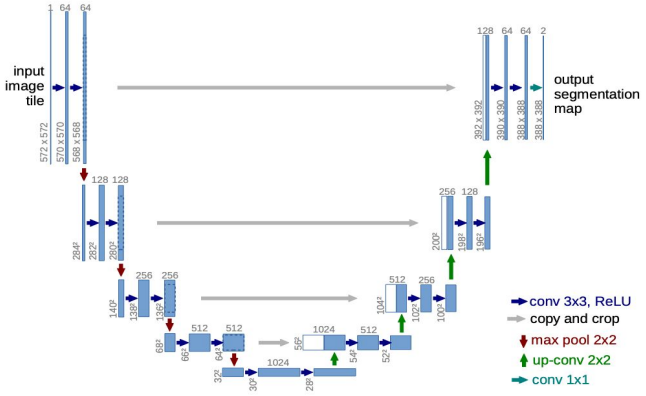


fig.1 Original U-Net Structure [8]

2) Image Preprocessing: Firstly, we cropped original images from 4288 × 2848 to 3500 × 2848 by reducing its width. Secondly, we padding the image to 3500 x 3500 and followed by resizing it to 640 × 640 through bicubic interpolation to protect image details. Thirdly, to do normalization, each image is divided by 255 as the network input. Also, we did data augmentations by doing flipping, re-scaling, rotation and adding Gaussian noise.

TABLE I. METRICS OF U-NET IN TASK 1

Block	U-Net Operation	U-Net Output size			
Input	fundus	(640,640,3)	up 2	up(conv7)	(160,160,256)
conv 1	{3 × 3 conv, batch-norm, ReLU} × 2	(640, 640, 32)	conv 8	{3 × 3 conv, batch-norm, ReLU} × 1	(160, 160, 128)
pool 1	2 × 2 maxpool	(320, 320, 32)	concat 2	[conv14, conv3]	(160, 160, 256)
conv 2	{3 × 3 conv, batch-norm, ReLU} × 2	(320, 320, 64)	conv 9	{3 × 3 conv, batch-norm, ReLU} × 2	(160, 160, 128)
pool 2	2 × 2 maxpool	(160, 160, 64)	up 3	up(conv9)	(320,320,128)
conv 3	{3 × 3 conv, batch-norm, ReLU} × 2	(160, 160, 128)	conv 10	{3 × 3 conv, batch-norm, ReLU} × 1	(320, 320, 64)
pool 3	2 × 2 maxpool	(80, 80, 128)	concat 3	[conv10, conv2]	(320, 320, 128)
conv 4	{3 × 3 conv, batch-norm, ReLU} × 2	(80, 80, 256)	conv 11	{3 × 3 conv, batch-norm, ReLU} × 2	(320, 320, 64)
pool 4	2 × 2 maxpool	(40, 40, 256)	up 4	up(conv11)	(640,640,64)
conv 5	{3 × 3 conv, batch-norm, ReLU} × 2	(40, 40, 512)	conv 12	{3 × 3 conv, batch-norm, ReLU} × 1	(640, 640, 32)
up 1	up(conv7)	(80,80,512)	concat 4	[conv12, conv1]	(640,640,64)
conv 6	{3 × 3 conv, batch-norm, ReLU} × 1	(80, 80, 256)	conv 13	{3 × 3 conv, batch-norm, ReLU} × 2	(640, 640, 32)
concat 1	[conv6, conv4]	(80, 80, 512)	conv 14	{1 × 1 conv, sigmoid} × 1	(640, 640, 1)
conv 7	{3 × 3 conv, batch-norm, ReLU} × 2	(80, 80, 256)			

3) Network Architecture: As shown in Table I, the U-net structure we used was U-net modified by team VRT [3]. The input fundus image size is 640 × 640 × 3, the details of convolutional blocks are in the table. After 4 times max-pooling, the image size is down to 40 × 40, and then the image is up-sampled to the original size with one channel. In the process of up-sampling, we concatenate the up-sampling layers with corresponding initial layers because the features are important in both initial layers and up-sampling layers for segmentation.

4) Implementation: Because the last layer of the model is sigmoid, the output of the image is in the range of [0, 1]. We used weighted binary cross entropy as loss function. In the function, k is the batch size. α is the weight which is calculated by N_0 and N_1. N_0 is the number of background and N_1 is the number of foreground. ɣ is the hyperparameter that we need to choose. The reason why we use the weight in the binary cross entropy is because the background area is significantly large in the image. We need to use the weight to reduce the false negative to reduce the loss. We use Adam optimizer as optimization and an initial learning rate of 1e-4. Every 10 epochs, learning rate would be 0.1 times as before.

$$L = \frac{1}{k}\sum_{i=1}^{k}\left[-\alpha \mathbf{y}^i_{true}\log \mathbf{y}^i_{pred} - (1-\mathbf{y}^i_{true})\log(1-\mathbf{y}^i_{pred})\right] \quad (1)$$

$$\alpha = \frac{N_0^i}{\gamma N_1^i} \quad (2)$$

5) Learning Curve:

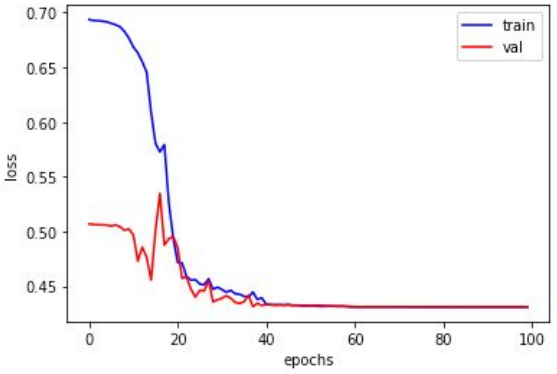


fig.2 Learning curve for HE

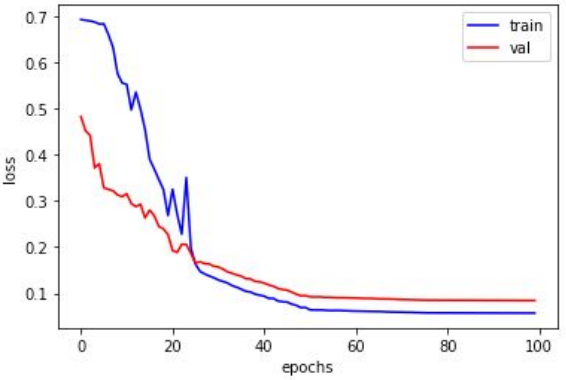


fig.3 Learning curve for EX

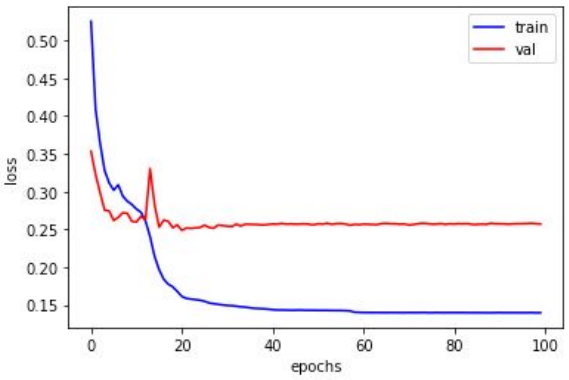


fig.4 Learning curve for MA

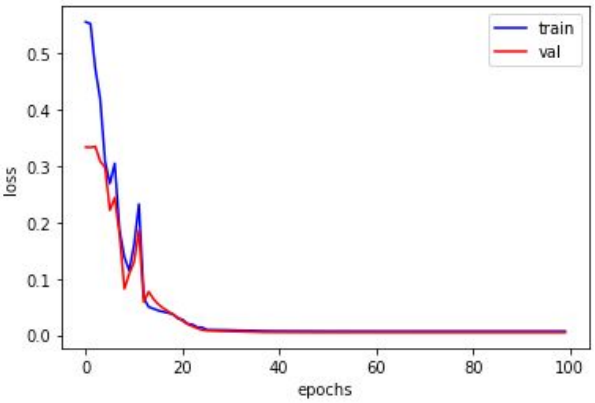


fig.5 Learning curve for SE

We can see from the picture, at the very beginning, val loss was sometimes actually diverging. The problem was likely caused by the initial learning rate was too high. After 20 epochs, loss started to converge yet some models suffered a lot from overfitting(MA). This could be the reason why our model performed badly in MA classification.

For SE, both training loss and val loss were low. Decent metrics are expected from this model.

B. Task 1: Region growing segmentation(RGS)

1) Algorithm Introduction: Region growing segmentation is an image segmentation method based on region features. It could be regarded as a pixel-wise segmentation method since the selection of the initial seeds is important. The algorithm aims at scanning through neighbors of initial seed pixels. Then it would use a criterion (a threshold value) to decide whether the neighbors should be added into a region.

Intuitively, this method is an DFS-like iterative searching algorithm. It started from an initial seed and ended up with all connected pixels being marked. The output is a seed-mark matrix which could offer the information about which pixel belongs to which region.

2) Image Preprocessing: The original 4K resolution images take a lot of space and the black background contains no useful information. Therefore, they are cropped into smaller size (3000 × 3000). This cropped images are converted from RGB to HIS color shape and we extracted I channels before passing them to following steps. Then CLAHE is applied to these single channel images to gain adaptive local contrast enhancement. In this way, we finished all the preparations we needed. Fig.6 below shows the original and preprocessed images.

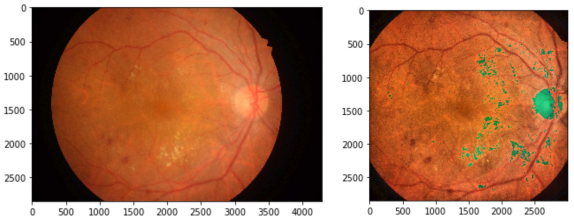


fig.6 Original and preprocessed images

3) Implementation: Based on the preprocessed images, the RGS processing is operating on the I channel's gray image. A seed is selected as the starting point and an empty stack is created for storing added pixels. Then we created a zero matrix having same size with original image for marking the seeds.

In python, a list object is used to simulate a stack. The initial seed [x,y] has four neighbors which are [x-1,y], [x+1,y], [x,y-1], [x,y+1], then we calculated the gap between seed and its four neighbours. If the absolute value of the result is less than or equal to the threshold (which is 10 in this case), the neighbouring pixel would be appended into the list and the seed-mark matrix would change from 0 to 1 at the same coordinates.

The region growing formula is clarified as fig.7. Iteration showed as fig.8 will not end until all pixels are marked. Then selected region's pixels' median intensity would replace all original intensities in this area, in other words, that region's pixels are merged. After using the segmented image to subtract the optic disc, HE regions can be obtained.

$$(a)\ \bigcup_{i=1}^{n} R_i = R.$$

(b) R_i is a connected region, i = 1, 2, …, n

$$(c)\ R_i \bigcap R_j = \varnothing, i \neq j$$

$$(d)\ P(R_i) = \text{TRUE for } i = 1, 2, \ldots, n.$$

$$(e)\ P(R_i \bigcup R_j) = \text{FALSE for any adjacent region } R_i \text{ and } R_j.$$

fig.7 Region growing formula

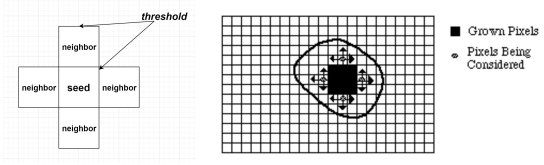


fig.8 Region growing iteration

C. Task1: Mask R-CNN

1) Model introduction: Mask R-CNN is a deep learning model modified from faster R-CNN by adding a branch for predicting segmentation masks on each Region of Interest, in parallel with the existing branch for classification and bounding box regression[1]. Mask R-CNN models replace

the RoI pooling layer with an RoI alignment layer. This allows the use of bilinear interpolation to retain spatial information on feature maps, making Mask R-CNN better suited for pixel-level predictions. The RoI alignment layer outputs feature maps of the same shape for all RoIs. This not only predicts the categories and bounding boxes of RoIs, but also allows us to use an additional fully convolutional network to predict the pixel-level positions of objects.

2) Image Preprocessing and model structure: Firstly, we convert the image into a png format that the model can read. Then downsample all original images and masks into same size as 536 × 365 because Mask R-CNN model causes memory overflow when dealing with large size images. RGB channels are remained to keep all the details of the images.

A brief structure of Mask R-CNN model has been shown in Figure 9. This model can be treated as two-stage procedure. The first stage is Region Proposal Network (RPN), proposes candidate object bounding boxes. In the second stage, in parallel to predicting the class and box offset, Mask R-CNN also outputs a binary mask for each ROI. During training a multi-task loss on each sampled RoI as formula (3).

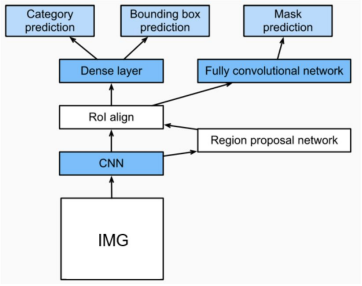


Fig.9 Mask R-CNN model.

$$L = L_{cls} + L_{box} + L_{mask} \quad (3)$$

Where Lcls and Lbox defined in [23] and define L_{mask} as the average binary cross-entropy loss.

3) Implementation: We set hyper parameters as same as existing sample model from Mask R-CNN source code. Considering the trade off between running time and effect, resnet 50 has been selected instead of resnet 101 as convolutional layer. Learning rate and learning momentum are set as 0.001 and 0.9 respectively. Train the Mask R-CNN for 20 epoch and each epoch have 500 training steps.

D. Task 2: Alternate sequential Filtering

1) Introduction: Retinal blood vessels extraction is also a primary step for detecting eye diseases including diabetic retinopathy which causes blindness. It also simplifies other image processing techniques such as classification we mentioned before. Since manual extraction is a time-consuming task, many automated methods have been proposed. In this report, we considered an algorithm for extracting blood vessels based on Alternate sequential Filtering (ASF). The proposed method has been tested on IDRiD dataset, and the result shows that this method capable of extracting blood vessels. Performance of the technique is evaluated using Jaccard metrics that are obtained as 80.41% on average on IDRiD dataset.

2) Implementation: The flow-chart for the proposed scheme for Retinal blood vessel segmentation is as shown in fig.10.

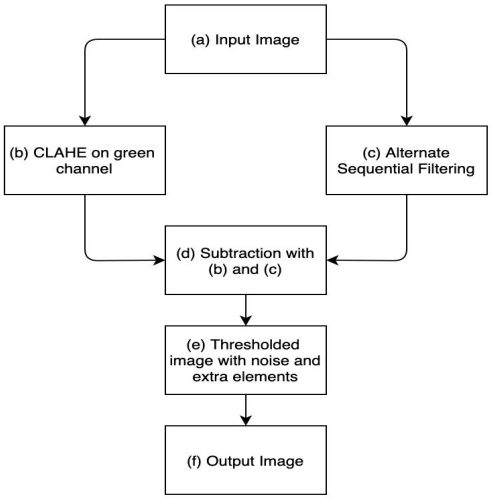


Fig.10 Blood Vessels Segmentation Flowchart.

The very first preprocessing step is to extract green channel out of the original image using as source image. This is because green channel would usually give us best contrast against blood vessel.

As the flow-chart shows, we can use Intensity Enhancement method such as Contrast Limited Adaptive Histogram Equalization (CLAHE) to enhancement vessel intensity. Then we can use the Alternative Sequential Filter (ASF) to smooth some parts of the blood vessels using the disk as a structuring element. [20] ASF is based on combining morphological opening and closing or vice versa. While the opening is dilation followed by erosion, closing is erosion followed by dilation. In this way, we can remove the speckled features which would mess up with the large structuring elements returned by this filter. Once we got CLAHE and ASF results, we could subtract these two result images to filter out blood vessel. [21] This would give us an image which contains faint traces of blood vessels with optic disk and other things removed. We can binarize this image with a thresholding using Otsu's method and get blood vessels segmented.

The final noises could be removed by morphological closing the images. Hopefully only blood vessels would remain in shape. [22] More details and results will be mentioned in Section IV.

E. Task 2: ResNet18 + U-Net

1) Model introduction: ResNet18 is short for 18-layer residual networks. ResNet was brought up by K He, et al. in 2015. [7] The main difference between a ResNet and

previous neural net architecture is that ResNet introduced "shortcut" between layers.

One common problem would occur during deep learning training is that, as the network goes deeper, the gradients would eventually converge to infinity or zero. Then it would become impossible to update the gradients using backpropagation algorithm. ResNet's shortcut solved the vanishing(gradients converge to zero) problem by retaining some of the previous weights when network is going to the next layer. Intuitively, since the layers are always connected to the previous layer, we can at least expect to get a similar result. In one word, ResNet is used to slow down the decay of gradients and allows network to go deeper.

The ResNet18 model we used in this task was a pre-trained model provided by pytorch. A basic ResNet block looks like this. [7]

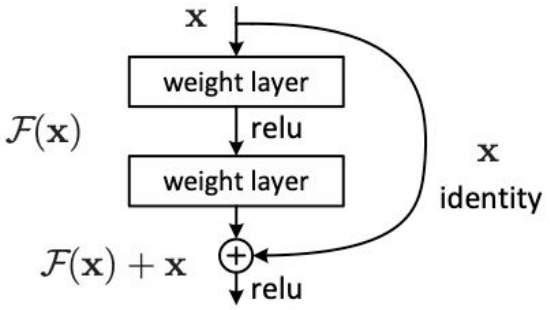
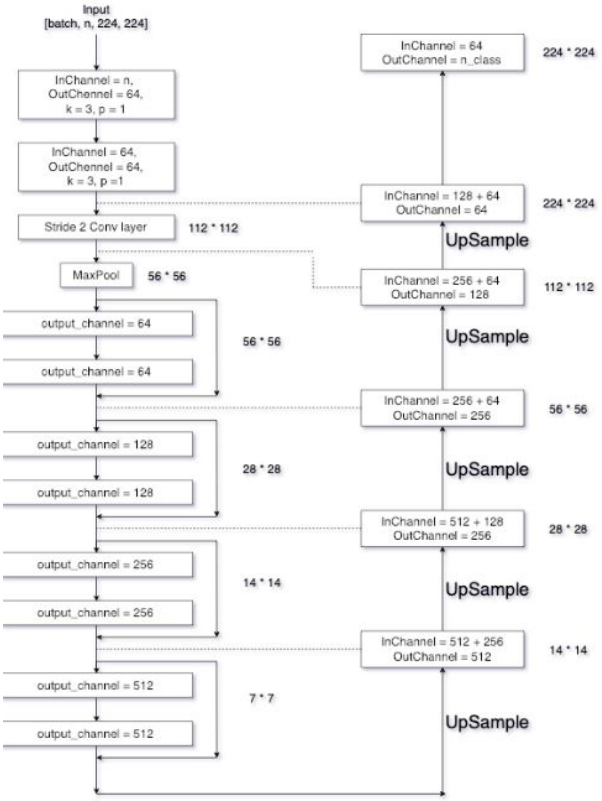


fig.11 basic ResNet block[7]

As for Unet, we've been discussing that in the previous section. It is a powerful architecture when doing pixel-wise segmentations.

This ResNet18 + U-Net model was inspired and derived from here[18]. This concept of combining two models together is called transfer learning. To be more concrete, pre-trained ResNet was used as the encoding layers of U-net, by doing backpropagation only on U-net layers, U-net is learning based on the output of ResNet.



2) Image Preprocessing: The preferred input of ResNet is 224 × 224 × 3 images. We ended up using 224 × 224 × 1 pictures as our input since we've already known that green channel would give us best blood vessel visualization. Those 224 × 224 patches were cropped from original images by selecting 100 sliding windows. Other than this, we also did CLAHE and gamma adjustment to enhance the blood vessels even harder. Then, normalization was applied to the inputs.

3) Network Architecture: See in fig.12

4) Implementation: We used PyTorch's built-in Adam algorithm as optimizer. Learning rate was getting 10 times smaller after every 10 epochs. We trained our model on Google Colab for 100 epochs. And it would take about 50 minutes. After seeing some promising results from the outputs, we continued to defrost ResNet's gradient to do fine-tuning, fine-tuning would take about 2.5 hours for 100 epochs.

5) Learning Curve:

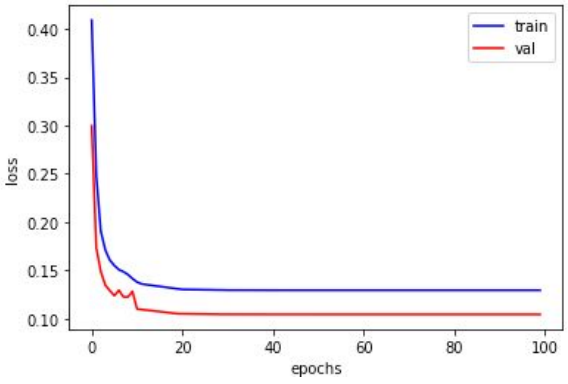


fig.13 Learning curve of training set and validation set for 100 epochs

We can see from the picture, our model got a pretty good result on this task. Both training and validation losses are quite low. No signs of overfitting or underfitting.

IV. Experimental Setup and Results

A. Task 1: U-net

In this method, we use deep learning to achieve the Retinopathy Segmentation. The neural network we use is the U-net structure.

1) Data preparation: After we applied data augmentation, we had 5400 pictures of size 640 × 640 × 3 for every task. As training set and test set has already been divided, we only set 400 pictures in the training set as validation set. By comparing the loss in the training and validation set, we could adjust our network to be more generalized.

The figure below is an example of image after image preprocessing.

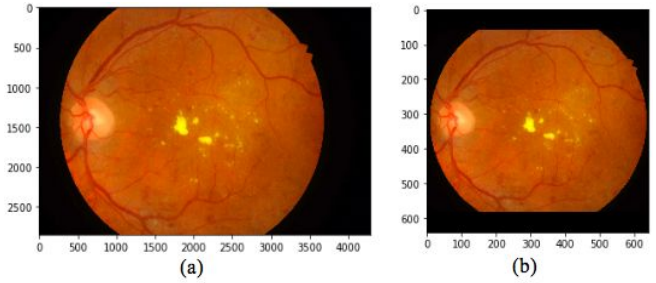


fig.14 (a) Image of original size. (b) Image of Input size.

2) Experiment: Totally, there are 4 classes. For each of the class, we consider it as a binary classification problem using the same U-net model and optimizer. Loss functions for these models had different hyper parameters. We trained 100 epochs for every class to get the results. The initial learning rate was 1e-3 and would decay 10 times after every 10 epochs. Referring to team VRT in the challenge, for EX, SE, HE and MA, we choose γ as 64, 512, 8 and 32 respectively to calculate the weights in BCEloss function to get relatively good performance.

3) Results and metrics: Here we post several examples of our intermediate results and outputs.

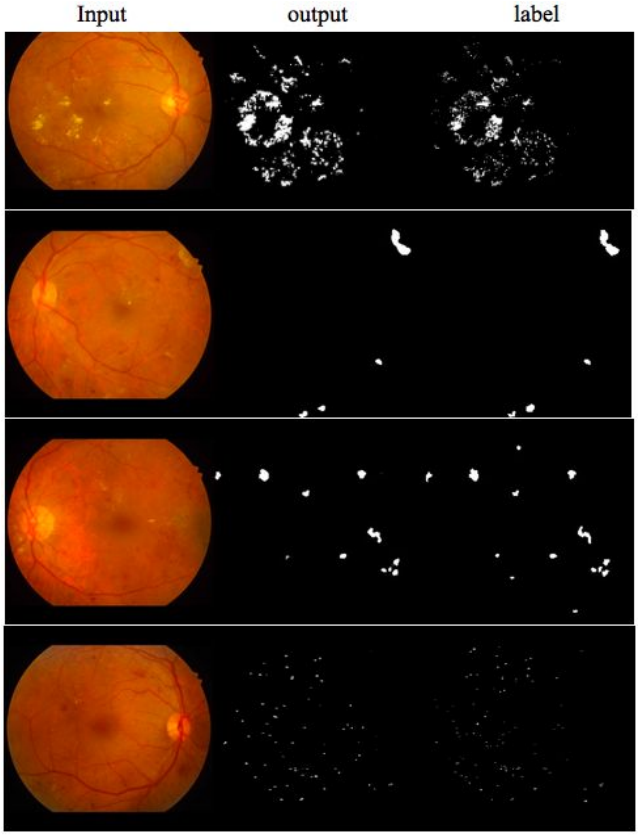


fig.15 The image from top to bottom are EX, SE, HE, MA.

Four common metrics were used in this section. We could compute true positive(TP), true negative(TN), false positive(FP) and false negative(FN) using bitwise operation between predictions and ground truth. After getting these four values.

We could compute recall using recall = TP/(TP + FN), precision using precision = TP/(TP + FP), accuracy using accuracy = (TP + TN)/(TP + TN + FP + FN) and f1-score using f1 = 2 * recall * precision/(recall + precision).

Details are in the table below.

TABLE II. METRICS OF U-NET IN TASK 1

	recall	***precision***	***accuracy***	***f1-score***
EX	69.9%	79.9%	99.4%	73.5%
HE	74.5%	83.7%	99.5%	78.7%
MA	33.6%	72.1%	99.9%	44.4%
SE	94.5%	92.0%	99.9%	93.2%

B. Task 1: Region Growing Segmentation

1) Experiment: During region growing segmentation procedure, the threshold is selected to be 10 for determining whether the neighbouring pixel needs to be merged. In addition, *python list* is used for convenience instead of building a stack class and *append/pop* are applied for stack operations. Generally, the hard exudates with other lesions and optic disc could be segmented together, thus the mask image of OD needs to be loaded and converted to the binary image. The HSI image's gray and I channels are both tested to get the mark of seeds.

2) Results: The result of segmentation is not great, and the seed-mark matrix outputs are presented as fig.16.

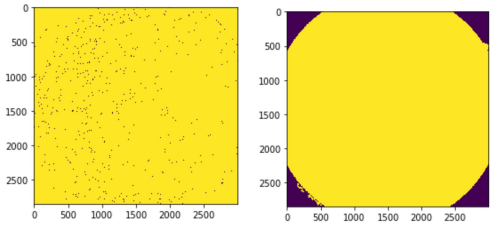


fig.16 seedmark: I channel (left) and HSI gray scale (right)

After segmentation, median intensity replaces the original values, in this way, HE and OD should be cropped out. However, it is based on the quality of segmentation and it is still not performing well. The fig.17 presents the result.

The metrics are missing since the outputs are not good enough. So we didn't add metrics of this part in the comparison.

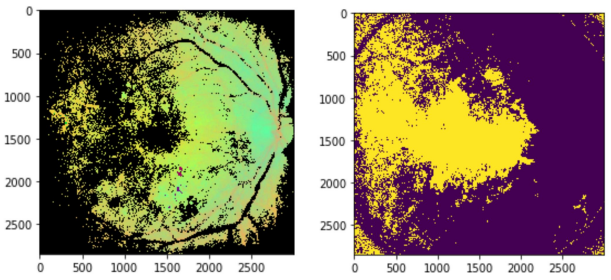


fig.17 segmented images :I channel (left) and HSI gray scale (right)

C. Group task 1: Mask R-CNN

1) Data preparation: We used all dawn sampling images from training datasets, and try to use images from data augmentation datasets for testing. Input images can be augmented in Mask R-CNN model limited by 8 different augmentation methods.

2) Experiment: For the first stage, Region Proposal Network runs a lightweight binary classifier on a lot of boxes (anchors) over the image and returns object/no-object scores as shown in Figure 22. Anchors with high objectness score (positive anchors) are passed to the stage two to be classified (fig.23).

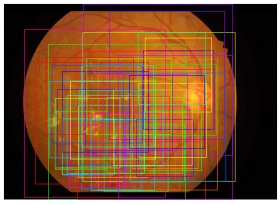


fig.22 (a) Proposal boxes

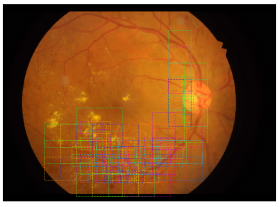


fig.23 Top100 score boxes

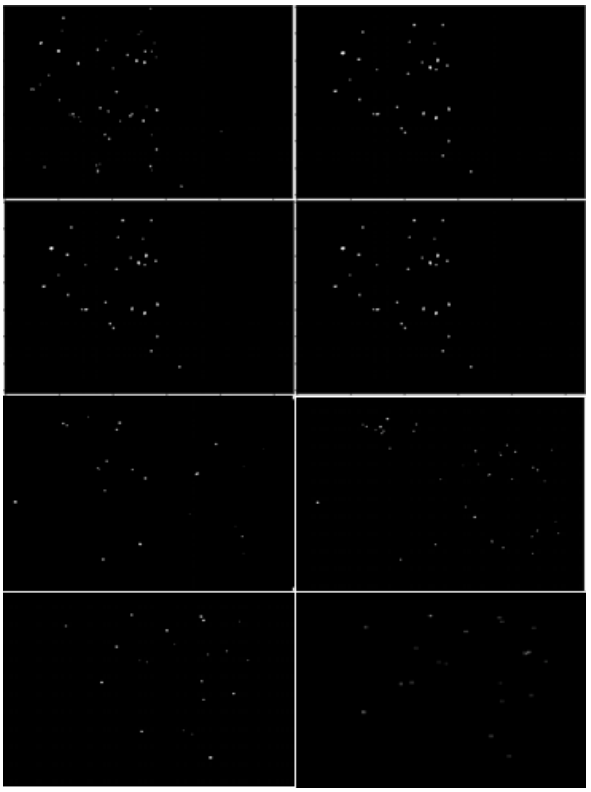
Next stage takes the region proposals from the RPN and classifies them, followed by applying Bounding Box Refinement.

3) Results:

The prediction results of mask Figure 24.a and the ground-truth Figure 24.b corresponding 4 classes of disease have been shown as an example. As we can see that the mask image predicted from Mask R-CNN model based on resnet 50 as convolutional layer has a good performance visually, evaluation matrix has been shown in table III.

TABLE III. METRICS OF ASF IN TASK 2

	recall	***precision***	***accuracy***	***f1-score***
average	41.1%	39.0%	99.6%	40.0%



(a) (b)

Fig.24 (a) predictions and (b) ground-truth

D. Task 2: Alternate sequential Filtering

1) Experiment: In this method we can use the training image directly, we can use the origin image to generate CLAHE result first, select the correct kernel size and clip limit can help us get more accurate result in the final. We propose the use of kernel size with 8*8 and clip limit with 2.0, which are estimated by comparing the blood vessel segmentation results experimentally. Figure 18 shows a few example CLAHE results generated by kernel size 8*8 and different clip limit.

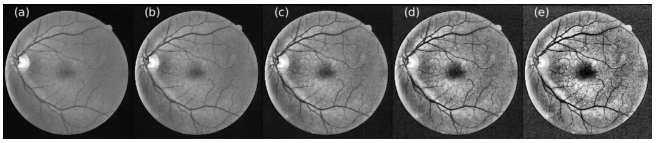


Fig.18 Example results of CLAHE: (a) clip limit = 1.0, (b) clip limit = 2.0, (c) clip limit = 4.0, (d) clip limit = 8.0, (e) clip limit = 16.0

We can see above example output images, larger clip limit value may produce more noise even in the background.

Before we do subtraction we also need to use ASF to remove the speckle features. As we mentioned before, ASF is based on combining opening and closing or vice versa. Due to this method result will also affect the subtraction directly, we need to repeated verificate the closing and opening times and the corresponding kernel size. In the final we choose 3 times closing opening and the corresponding kernel size as Figure 19 which are compared by the final result accuracy.

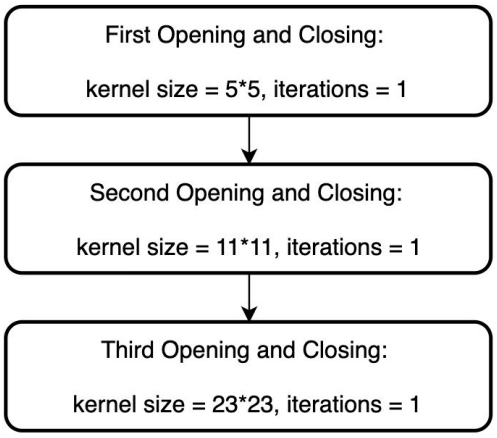


Fig.19 Opening and Closing times and corresponding parameters.

When we get the use CLAHE and ASF process the image separately, we can use these results to do subtraction. However, there may still have some very small contours and blobs of unwanted bigger chunks taking into consideration they may not straight lines like blood. Therefore, we can use area parameter noise removal to remove these features. We only remove the noise contours which the contours area smaller than 200.

2) Results and metrics: Above parameters we mentioned are all estimated by comparing the blood vessel segmentation results experimentally. Therefore, we get a considerable accuracy which is 80.41% on average on the IDRiD dataset evaluated by using Jaccard metrics. Figure 20 shows the final results of this method and the Table IV shows the metrics.

fig.20 Final result of Alternate sequential filtering and CLAHE subtraction

TABLE IV. METRICS OF ASF IN TASK 2

	recall	***precision***	***accuracy***	***f1-score***
Best Example	83.2%	57.8%	93.0%	64.7%
Average	73.4%	49.2%	91.0%	58.4%

E. Task 2: ResNet18 + U-Net

1) Data preparation: In this section, we also prepared 5400 pictures derived from the original dataset. Instead of doing some data augmentations, we simply cropped the images into 224×224 patches to feed to the model. Again, we used 400 of them as the validation set to track the training process.

Then in evaluation procedure, we padded test images to make each of them into nine 224×224 patches, then feed those patches into our trained model and concatenate the outputs to restore the original size.

2) Experiment: This is a standard binary classification problem. We used binary cross entropy as loss function and added a sigmoid function to let it vary between [0,1]. Learning rate was starting from 1e-3. It would become 10 times smaller after every 10 epochs. Both SGD and Adam were tried as optimizer yet no obvious differences occurred. Here we only recorded the results for Adam.

3) Results and metrics: Same as task 1. Intermediate results and outputs are shown below.

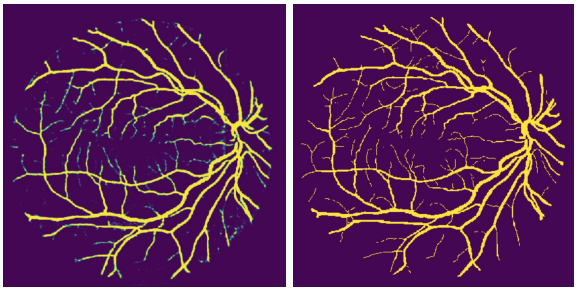


fig.20 Raw output and ground truth

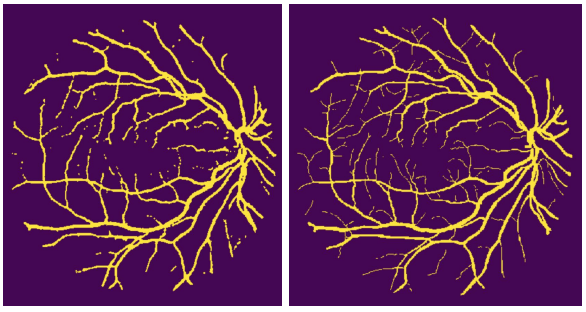


fig.21 Predictions with thresholding and ground truth

TABLE V. METRICS OF RESNET18+U-NET IN TASK 2

	recall	***precision***	***accuracy***	***f1-score***
Best Example	86.0%	85.8%	97.6%	81.6%
Average	76.0%	79.4%	97.2%	77.4%
Average after closing	75.9%	79.5%	97.2%	77.4%
Average after closing and opening	73.4%	80.5%	97.1%	76.6%

V. DISCUSSION AND CONCLUSIONS

A. Part1

1) U-net: This deep learning method in the lesion segmentation task can achieve ideal performance. However, a good neural network model needs to be trained reasonably. The deep learning normally needs to be fed with a large amount of data for training. Also, a multiple layer neural network such as our model is computely complicated. We may use the same model in different tasks but we need to change parameters and train separately for different tasks. After built a good model, we typically can get good prediction results by that.

2) Region growing segmentation: This region growing segmentation method in extracting lesions cannot obtain promising performance. The advantage of RGS is intuitive but it is computationally expensive which has $O_{(n^2)}$ time complexity. Besides, the algorithm is noise sensitive and the task images are complex with many spots similar to lesions, so the excess pixels are merged inside causing a fuzzy region. The poor quality of segmentation can lead to the bad result.

3) Mask R-CNN: The training of Mask R-CNN is surely not good, that maybe because we using single GeForce GTX 1050 GPU to train our model, also we just train 20 epoch, each epoch contain 500 train step. That may occur the loss function not convergence enough. Moreover, there is a possible issue is the high dispersion among mask in MA class, we should use using resnet 101[27] with smaller mask to achieve better performance. To solve the problems we mentioned before, tune hyperparameters like DETECTION MIN CONFIDENCE or RPN NMS THRESHOLD could be a possible improvement for this task, but we did not have enough time to come up with interesting conclusions. Because 20 epoch cost us about 40 hours, and the training time could be longer if using more complex convolutional layer.

4) In task 1, the best result was achieved in SE datasets while MA datasets got the worst. The reason could be MA pixels are so sparse and tiny, we lost some of the information during compressing the original images.

TABLE VI. PERFORMANCE (F1-SCORE) COMPARISON IN TASK 1

	EX	***HE***	***MA***	***SE***
U-Net	73.5%	78.7%	44.4%	93.2%
Mask R-CNN	/	/	40.0%	/

B: Part 2

1) Alternate sequential Filtering: When we remove the small contours and blobs of unwanted bigger chunks, we also lose some vessels feature, that occur our result vessel not contain capillary. Besides, the subtraction of CLAHE and ASF result may lead to the result contain retinal contours. Therefore we may use machine learning methods such as Support Vector Machines[24][25] and Extreme Learning Machine[26] to solve these problems and also keep the velocity.

2) ResNet18 + U-net: Even though good results are obtained from raw outputs. It's somewhat harder to make further improvement. If one used morphological open to deduct noises, some details of the blood vessel would also be removed. If one used morphological close to connect disconnected bloodvessel, some tiny gaps between vessels would also be filled.

3) Both methods got good visualizations. ASF is much faster than neural network. If we need some fast outputs, training a new neural network would be too time-consuming to finish the task. Also traditional methods are more explainable. However, if one concerns performance over time, he should choose DL based method.

TABLE VII. PERFORMANCE COMPARISON IN TASK 2

	f1-score	***time***
ASF	58.4%	75 seconds
ResNet + U-Net	77.4%	3~4 hours

VI. CONTRIBUTIONS OF TEAM MEMBERS

In the process of doing the project. Every member in our team are united and responsible. We divide work according to the project methods and we all complete our corresponding part very well including coding, experimenting, making PPT and writing the report. We communicate and cooperate with each other and all participate in the whole process.

A. Part 1

1) U-Net: Chengze Du & Yichen Liu
2) Mask R-CNN: Yiwen Xu & Sheng Du
3) Region growing: Yiren Hu

B. Part 2

1) *Alternate sequential Filtering: Yiwen Xu*
2) ResNet18 + U-Net: Yichen Liu

REFERENCES

[1] Porwal, Prasanna, Samiksha Pachade, Manesh Kokare, Girish Deshmukh, Jaemin Son, Woong Bae, Lihong Liu et al. "IDRiD: Diabetic Retinopathy–Segmentation and Grading Challenge." Medical Image Analysis (2019): 101561.

[2] Budai, Attila, Georg Michelson, and Joachim Hornegger. "Multiscale Blood Vessel Segmentation in Retinal Fundus Images." Bildverarbeitung für die Medizin. 2010.

[3] He, K., Gkioxari, G., Dollár, P., & Girshick, R. (2017, October). Mask r-cnn. In Computer Vision (ICCV), 2017 IEEE International Conference on (pp. 2980-2988). IEEE.

[4] Long, J., Shelhamer, E., & Darrell, T. (2015). Fully convolutional networks for semantic segmentation. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 3431-3440).

[5] Sinthanayothin, C et al, 'Automated Detection of Diabetic Retinopathy on Digital Fundus Images' (2002) 19(2) Diabetic Medicine 105.

[6] Farokhian, Farmaz, and Hasan Demirel. "Blood vessels detection and segmentation in retina using Gabor filters." 2013 High Capacity Optical Networks and Emerging/Enabling Technologies. IEEE, 2013.

[7] He, Kaiming, et al. "Deep residual learning for image recognition." Proceedings of the IEEE conference on computer vision and pattern recognition. 2016.

[8] Ronneberger, Olaf, Philipp Fischer, and Thomas Brox. "U-net: Convolutional networks for biomedical image segmentation." International Conference on Medical image computing and computer-assisted intervention. Springer, Cham, 2015.

[9] Somfai, Gábor Márk, et al. "Fractal-based analysis of optical coherence tomography data to quantify retinal tissue damage." BMC bioinformatics 15.1 (2014): 295.

[10] Soares et al., “Retinal vessel segmentation using the 2-d Gabor wavelet and supervised classification,” *Medical Imaging, IEEE Transactions on*, vol. 25, no. 9, pp. 1214–1222, 2006.

[11] Azzopardi et al., “Trainable cosfire filters for vessel delineation with application to retinal images,” *Medical image analysis*, vol. 19, no. 1, pp. 46–57, 2015.

[12] Osareh et al., “Automatic blood vessel segmentation in color images of retina” *Iran. J. Sci. Technol. Trans. B: Engineering*, vol. 33, no. B2, pp. 191–206, 2009.

[13] Roychowdhury et al., “Blood vessel segmentation of fundus images by major vessel extraction and subimage classification,” *Biomedical and Health Informatics, IEEE Journal of*, vol. 19, no. 3, pp. 1118–1128, 2015.

[14] Fraz et al., "An Ensemble Classification-Based Approach Applied to Retinal Blood Vessel Segmentation", *IEEE Transactions on Biomedical Engineering*, vol. 59, no. 9, pp. 2538-2548, 2012.

[15] Qiaoliang et al., "A Cross-Modality Learning Approach for Vessel Segmentation in Retinal Images", *IEEE Transactions on Medical Imaging*, vol. 35, no. 1, pp. 109-118, 2016.

[16] Melinscak et al., "Retinal vessel segmentation using deep neural networks", *In Proceedings of the 10th International Conference on Computer Vision Theory and Applications (VISIGRAPP 2015)*, (2015), pp. 577–582.

[17] Liskowski et al., "Segmenting Retinal Blood Vessels with Deep Neural Networks", *IEEE Transactions on Medical Imaging*, vol. PP, no. 99, pp. 1-1, 2016.

[18] A. S. Chaudhuri, S. Chatterjee, N. Katz, M. Nelson, M. Goldbaum, "Detection of blood vessels in retinal images using two-dimensional matched filters", IEEE Trans. Med. Imag., vol. 8, pp. 263-269, Sept. 1989.

[19] F. Zana, J. Klein, "Segmentation of vessel-like patterns using mathematical morphology and curvature evaluation", IEEE Transactions on Image Processing, vol. 10, no. 7, pp. 1010-1019, 2001.

[20] A. Hoover, V. Kouznetsova, M. Goldbaum, "Locating blood vessels in retinal images by piecewise threshold probing of a matched filter response", IEEE Transactions on Medical Imaging, vol. 19, no. 3, 2000.

[21] R. Girshick. Fast R-CNN. In ICCV, 2015

[22] N. Cristianini, J. Shawe-Taylor, An Introduction to Support Vector Machines, U.K., Cambridge:Cambridge Univ. Press, 2000.

[23] E. Ricci and R. Perfetti, "Retinal blood vessel segmentation using line operators and support vector classification," IEEE Trans. Med. Imag., vol. 26, no. 10, pp. 1357-1365, Oct. 2007. (Pubitemid 47525297)

[24] Guang-bin Huang,"Introduction to Extreme Learning Machine", workshop on Machine learning for Biomedical Informatics, Nov 2006.

[25] S. Ohleyer, "Building segmentation on satellite images", Web: https://preject.inria.fr/aerialimagelabeling/files/2018/01/fp_ohleyer_compressed.pdf

[26] U - Net/FCN PyTorch: https://github.com/usuyama/pytorch-unet