RetinaScan Documentation

Group 10

1 What is RetinaScan?

Glaucoma is a condition that causes progressive damage to the optic nerve due to increased intraocular pressure beyond the optic nerve's tolerance. This damage is irreversible and can lead to blindness. Thus, early diagnosis and treatment are crucial for improving patients' quality of life. The ratio of the optical cup diameter to the optical disc diameter, known as the vertical cup-to-disc ratio (CDR) is a clinical indicator to assess the risk of glaucoma. A higher CDR indicates a greater risk. Our software called **RetinaScan**, can be used to segment optical disk regions, detect whether images represent glaucoma or not, segment blood vessels, and calculate the overall length of vessels.

2 Key algorithms behind this software

The overall workflow of our software is shown in **Figure 1**.

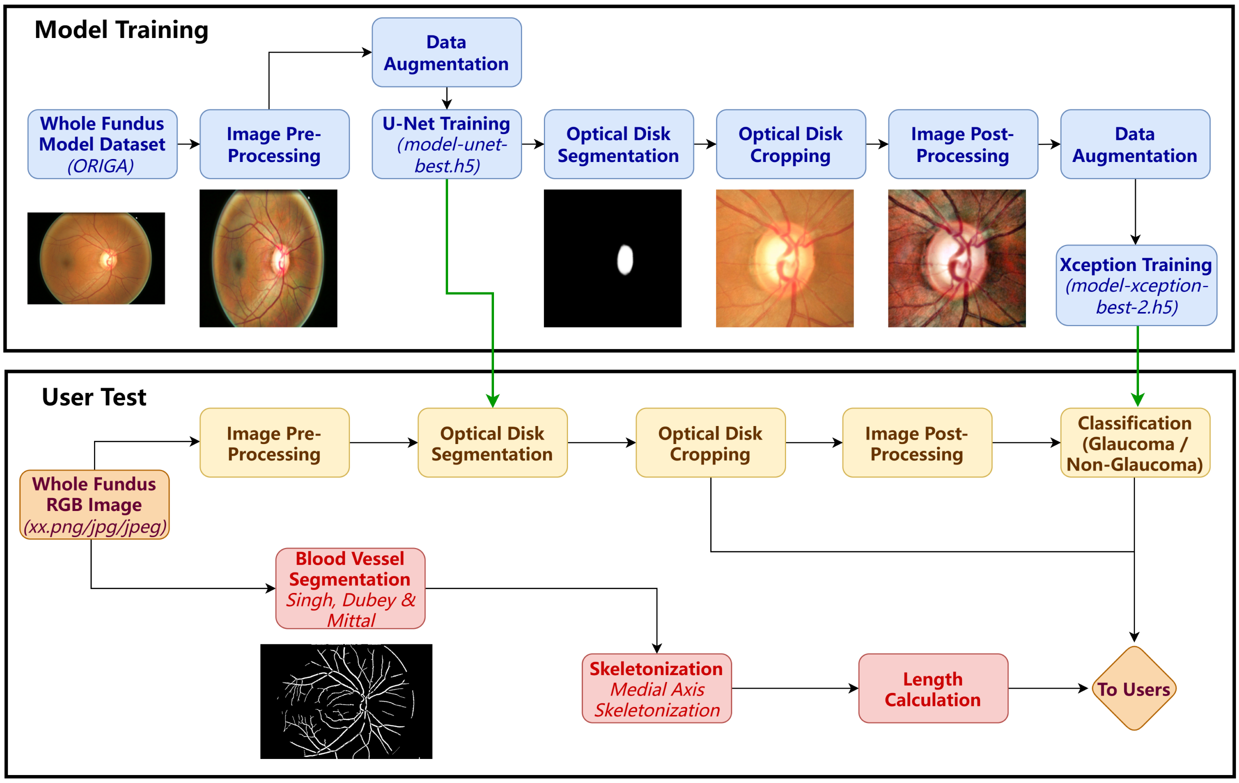

Figure 1: The workflow of the RetinaScan.

2.1 Optical Disk Segmentation

We first apply the U-Net with fewer filters to segment the optical disk region after image pre-processing. The pre-processing step includes contrast enhancement by Contrast Limited Adaptive Equalization **(CLAHE)** and size standardization. U-Net is a convolutional neural network **(CNN)** architecture appropriate for image segmentation. It applies a U-shaped framework, with down-sampling to extract features and subsequent up-sampling to localize spatial positions **(Figure 2)**.

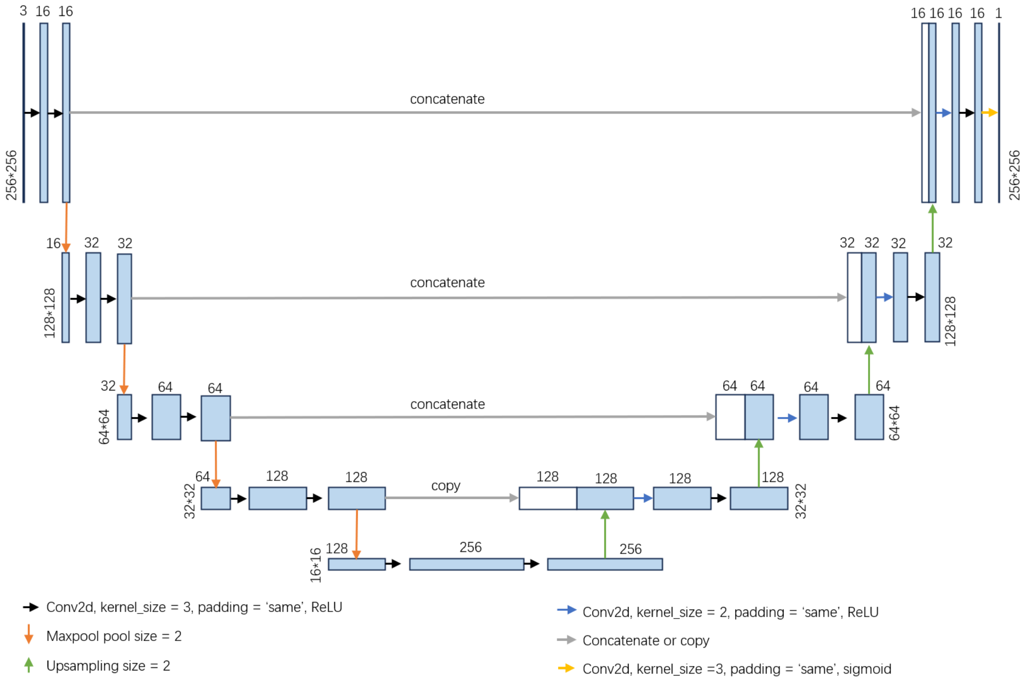

Figure 2: The architecture of U-Net used in RetinaScan.

2.2 Glaucoma Classification

Then we crop images to focus on the optical disk region based on the segmentation results. The cropped images are used for glaucoma classification. We use a modified Xception model, which is a deep-learning architecture that aims to improve the performance and efficiency of CNNs. Xception extends the Inception architecture by replacing the standard convolutions with depth-wise separable convolutions, leading to a more powerful and efficient model **(Figure 3)**.

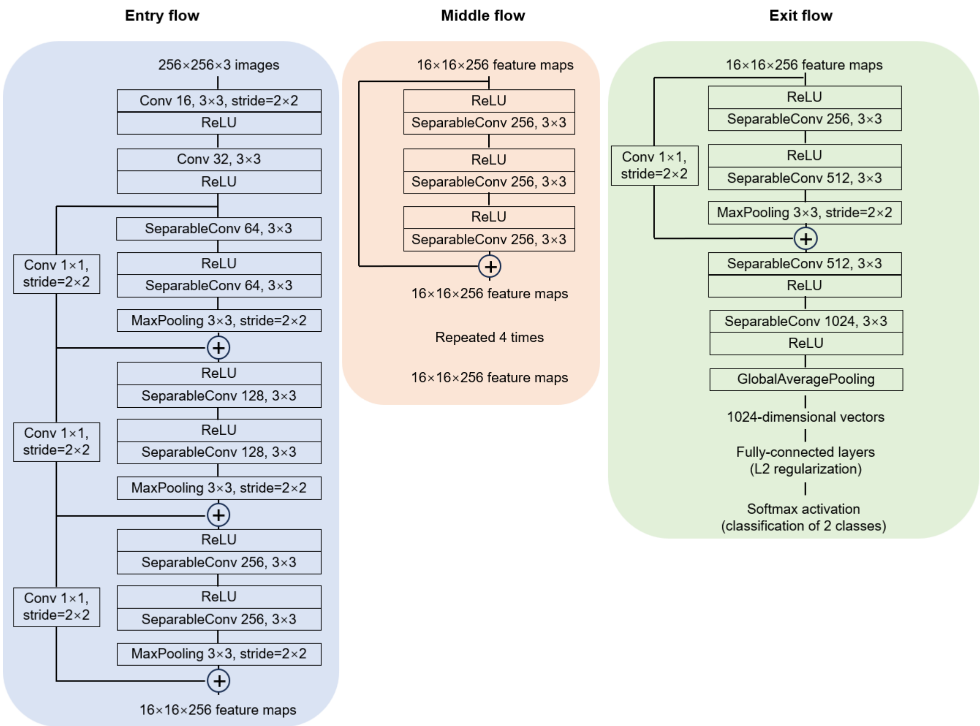

Figure 3: The architecture of Xception used in RetinaScan.

2.3 Vessel Segmentation & Length Calculation

For blood vessel segmentation, we opt to use a portion of the segmentation algorithm developed by Singh et al. in 2017. The Medial Axis algorithm is used for image skeletonization and lengths are calculated by counting the number of pixels occupied.

3 Running environment & reliable packages

The running environment of this software is ***Python 3.7***. Some other packages need to be downloaded before using this software, including ***numpy (v1.25.0)***, ***pandas (v2.0.2)***, ***scikit-image (v0.21.0)***, ***tensorflow (v2.14.0)***, ***keras (v2.14.0)***, ***opencv-python (v4.7.0.72)***, and ***matplotlib (v.3.7.1)***. We provide a *requirements.txt* file, which can be used to download all needed packages for software usage. Additionally, it also includes the packages needed to re-train the model using our appendix codes. Just run this command:

```
pip install -r requirements.txt
```

4 Parameter explanation (including input & output formats)

4.1 Input format

Our software allows for two types of input and will raise information if the input format is wrong (see the following examples for details):
(1) You can input one single image. The type of input images should be in **.png**, **.jpg**, or **.jpeg**, and encoded with three channels (RGB). If you input an image representing the whole fundus, our software will segment the optical disk region (return the cropped image focusing on the optical disk region), detect whether glaucoma exists or not, segment the blood vessel (return the segmented image), and calculate the vessel length (optional). If you input an already cropped image focusing on the optical disk region, our software will only detect whether glaucoma exists or not.
(2) You can also input a folder containing many images. Our software will only traverse **.png**, **.jpg**, or **.jpeg** files in this folder, and all images should be encoded with 3 channels. NOTA THAT you should make sure all images in this folder are in the same type (all represent the whole fundus or all represent the cropped images).

4.2 Parameter explanation

General usage: RetinaScan.py -i [-t {whole, cropped}] [-l] -o
Arguments:
-h, –help: Show the quick and brief help message.
-i, –input_path: The path of the input image or a folder containing many images.
-t, –type: The type of the input image or folder containing many images. Can only be set to two values: **whole** for whole fundus images, and **cropped** for cropped images focusing on the optical disk region. The default type is **whole**.
-l, –length: Whether to calculate the vessel lengths of your input images. If added, our software will include the vessel lengths in the final result file. NOTA THAT you can only set this parameter if your input images represent the whole fundus **(-t whole)**. It is impossible to segment the vessels and calculate lengths if your inputs are cropped images focusing on the optical disk region **(-t cropped)**.
-o, –output_folder: The output folder to save the cropped images, classification results, vessel segmentation images and vessel length.

4.3 Output format

(1) **Input one single whole fundus image:** Return 2 newly generated images (cropped image & vessel segmentation image) and 1 summary file containing classification & vessel length (if **-l/–length** is set).
(2) **Input one single cropped image**: Return 1 summary file containing classification information.
(3) **Input a folder with many whole fundus images:** Return 2 newly generated folders and 1 summary file.

(4) **Input a folder with many cropped images:** Return 1 summary file containing classification information.
You can see the example part for detailed information.

4.4 Additional information

The software will show the step it is currently running so that you can know whether your task is stuck and estimate how much time is needed to finish running the rest **(Figure 4)**.

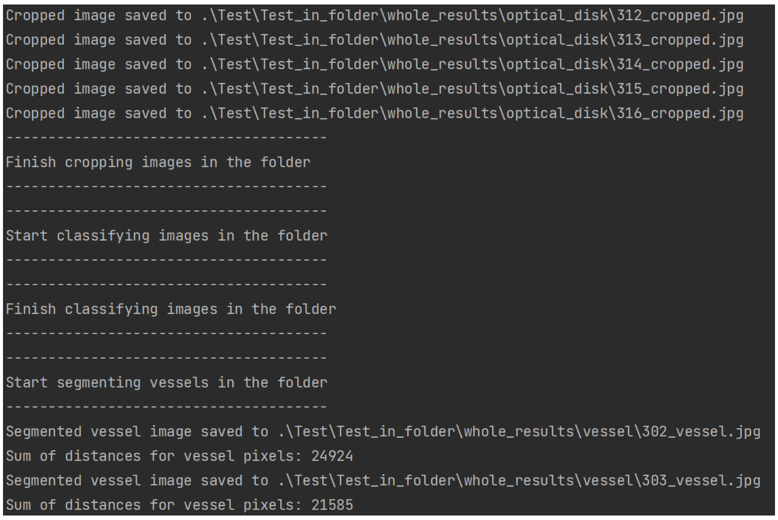

Figure 4: An example to show the process of RetinaScan.

5 Examples to use this software

Here we will include some examples of using RetinaScan (both correct usage & incorrect usage).

5.1 Error tracking

If our software raises the following messages, please check whether your input or the parameters are set correctly.
(1) **Error: The path [your path name] does not exist:** Your input path does now exist. Please enter the correct path.
(2) **Error: Your input file should be .png/.jpg/.jpeg:** You input a file with the wrong type.
(3) **Error: Your input image should be encoded in 3 channels (RGB):** It seems that you input a gray image with only one channel or an image with more than 3 channels.
(4) **Error: Your input image [your file name] should be encoded in 3 channels (RGB):** It seems that one image in your input folder is not encoded with 3 channels. Please replace it.
(5) **Error: We can not calculate the vessel length if your inputs are cropped images. Please use whole fundus images instead or do not set -l/–length to ignore the length calculation:** You select that your input represents cropped images **(-t cropped)** and also set **-l/–length** to calculate the vessel lengths. The cropped images contain very few vessels and it makes no sense to segment the vessels and calculate their lengths. Therefore, please only input whole fundus images to calculate lengths.

5.2 Example 1: The input is a single whole fundus image

```
python RetinaScan.py -i .\Test\Test_in_image\whole_RGB.jpg
                     -t whole -l
                     -o .\Test\Test_in_image\whole_results
```

The **whole_RGB.jpg** is a whole fundus image encoded with 3 channels. By running the above code, our software will generate 3 files in the output folder. :
(1) **whole_RGB_cropped.jpg:** The cropped image focusing on the optical disk region.
(2) **whole_RGB_vessel.jpg:** The vessel segmentation image.

(3) **results.csv**: The classification of this image and the total vessel length.
Additionally, the path of newly generated images, classification information, and vessel length will be shown in the terminal.

```
Cropped image saved to .\Test\Test_in_image\whole_results\whole_RGB_cropped.jpg
This image may represent non-Glaucoma.
Segmented vessel image saved to .\Test\Test_in_image\whole_results\whole_RGB_vessel.jpg
Sum of distances for vessel pixels: 31405
```

Figure 5: The terminal output of example 1

5.3 Example 2: The input is a folder containing many whole fundus images

```
python RetinaScan.py -i .\Test\Test_in_folder\whole_all_correct\
                     -t whole -l
                     -o .\Test\Test_in_folder\whole_results
```

The **whole_all_correct** is a folder containing 15 whole fundus images encoded with 3 channels. By running the above code, our software will generate 2 folders and 1 file in the output folder. :
(1) **optical_disk** This folder will contain all cropped images focusing on the optical disk region.
(2) **vessel:** This folder will contain all vessel segmentation images.
(3) **results.csv**: The classification results of all images and the total vessel lengths.
Additionally, the path of newly generated images, vessel lengths, and the process of software will be shown in the terminal.

5.4 Example 3: The input is a folder containing many cropped images

```
python RetinaScan.py -i .\Test\Test_in_folder\cropped_all_correct\
                     -t cropped
                     -o .\Test\Test_in_folder\cropped_results
```

The **cropped_all_correct** is a folder containing 28 cropped images encoded with 3 channels. By running the above code, our software will only return a **results.csv** file containing the classification results of all images in this folder.