Verification and Validation Plan for Retinal Vessel Segmentation System (RVSS)

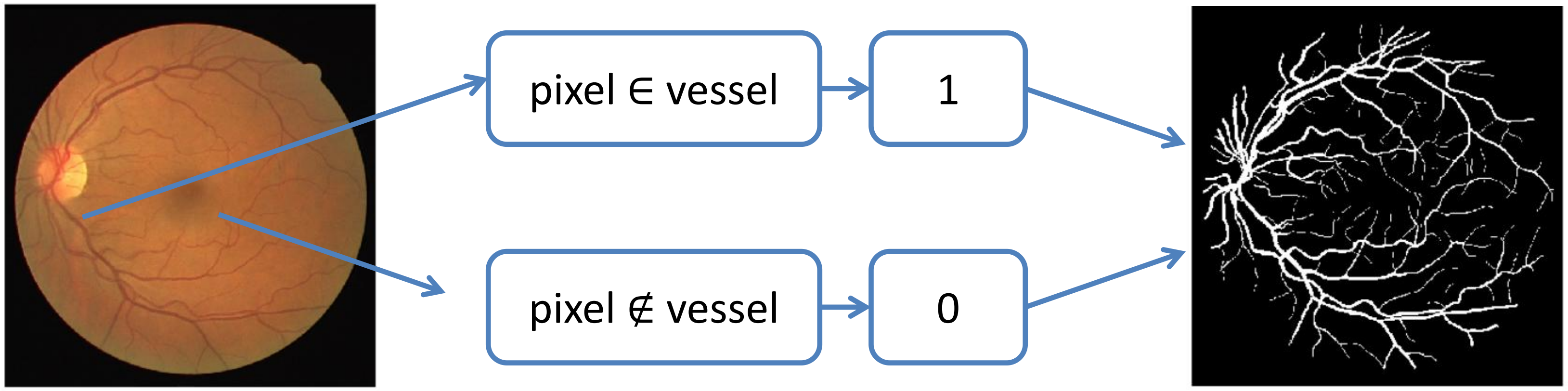
Xinyu Ma

February 16, 2024

Background

- vessel segmentation
  - an application of semantic segmentation in medical image analysis

associates a label or category with every pixel in an image



vessel segmentation --> to correctly classify the vessel pixels and background pixels in the retinal image

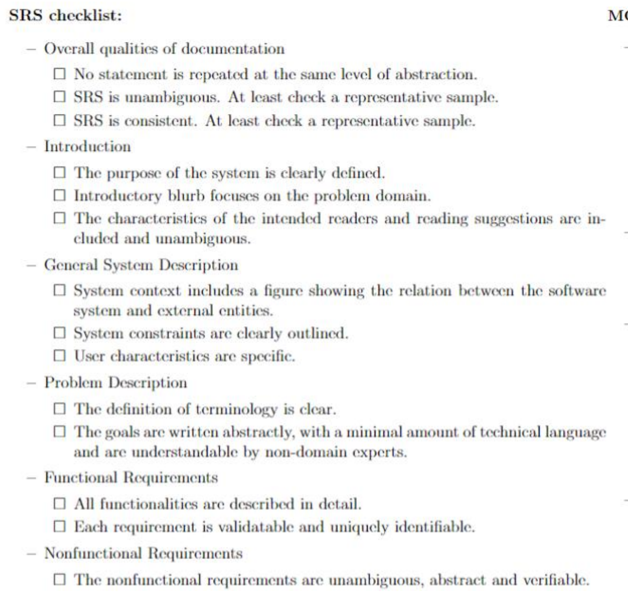
Plan

SRS Verification Plan

Design Verification Plan

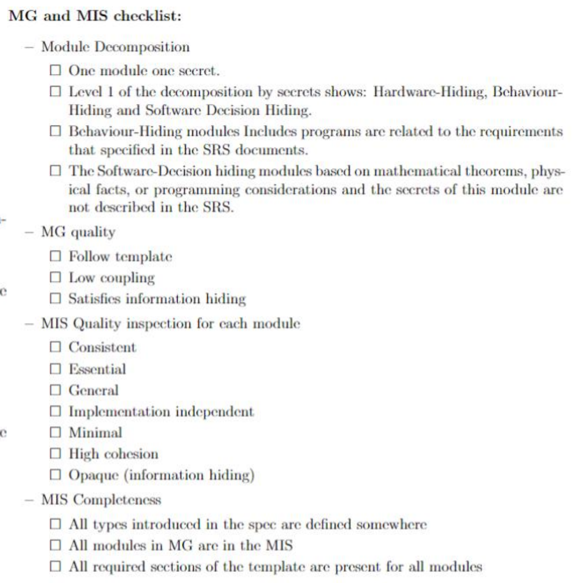
Verification and Validation Plan

1. Initial review from the assigned team members
2. Reviewers give feedback by creating issues in Github
3. Modify them according to the issues



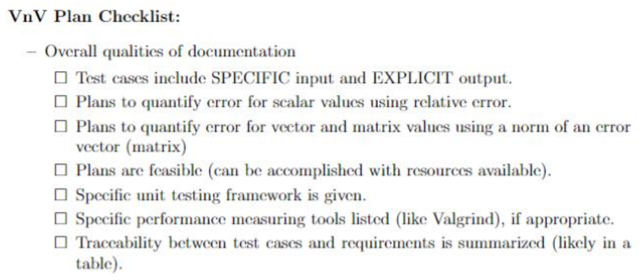
SRS checklist:

- Overall qualities of documentation
  - ☐ No statement is repeated at the same level of abstraction.
  - ☐ SRS is unambiguous. At least check a representative sample.
  - ☐ SRS is consistent. At least check a representative sample.
- Introduction
  - ☐ The purpose of the system is clearly defined.
  - ☐ Introductory blurb focuses on the problem domain.
  - ☐ The characteristics of the intended readers and reading suggestions are included and unambiguous.
- General System Description
  - ☐ System context includes a figure showing the relation between the software system and external entities.
  - ☐ System constraints are clearly outlined.
  - ☐ User characteristics are specific.
- Problem Description
  - ☐ The definition of terminology is clear.
  - ☐ The goals are written abstractly, with a minimal amount of technical language and are understandable by non-domain experts.
- Functional Requirements
  - ☐ All functionalities are described in detail.
  - ☐ Each requirement is validatable and uniquely identifiable.
- Nonfunctional Requirements
  - ☐ The nonfunctional requirements are unambiguous, abstract and verifiable.



MG and MIS checklist:

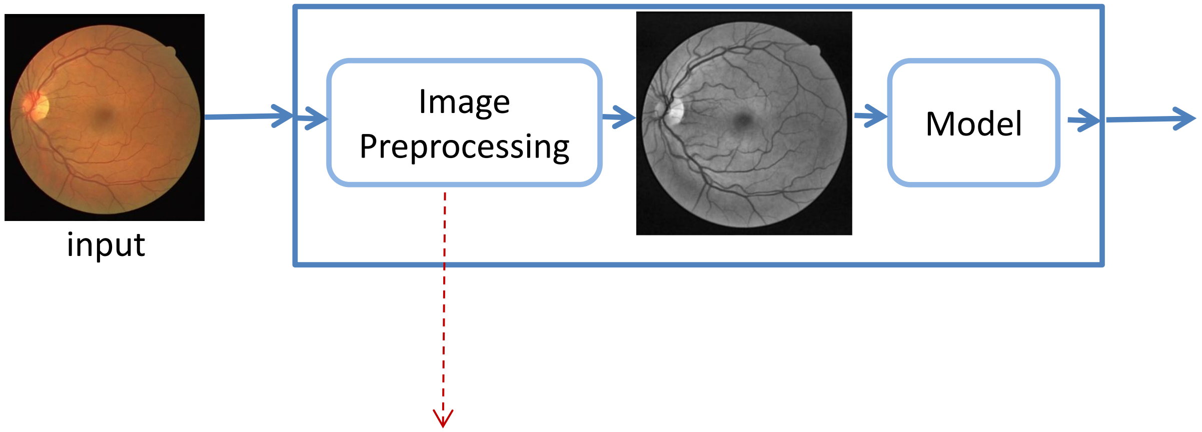
- Module Decomposition
  - ☐ One module one secret.
  - ☐ Level 1 of the decomposition by secrets shows: Hardware-Hiding, Behaviour-Hiding and Software Decision Hiding.
  - ☐ Behaviour-Hiding modules Includes programs are related to the requirements that specified in the SRS documents.
  - ☐ The Software-Decision hiding modules based on mathematical theorems, physical facts, or programming considerations and the secrets of this module are not described in the SRS.
- MG quality
  - ☐ Follow template
  - ☐ Low coupling
  - ☐ Satisfies information hiding
- MIS Quality inspection for each module
  - ☐ Consistent
  - ☐ Essential
  - ☐ General
  - ☐ Implementation independent
  - ☐ Minimal
  - ☐ High cohesion
  - ☐ Opaque (information hiding)
- MIS Completeness
  - ☐ All types introduced in the spec are defined somewhere
  - ☐ All modules in MG are in the MIS
  - ☐ All required sections of the template are present for all modules

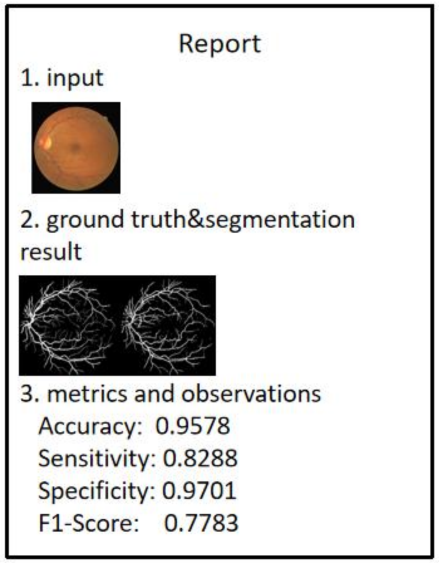


VnV Plan Checklist:

- Overall qualities of documentation
  - ☐ Test cases include SPECIFIC input and EXPLICIT output.
  - ☐ Plans to quantify error for scalar values using relative error.
  - ☐ Plans to quantify error for vector and matrix values using a norm of an error vector (matrix)
  - ☐ Plans are feasible (can be accomplished with resources available).
  - ☐ Specific unit testing framework is given.
  - ☐ Specific performance measuring tools listed (like Valgrind), if appropriate.
  - ☐ Traceability between test cases and requirements is summarized (likely in a table).

Tests for Functional Requirements




Report

1. input

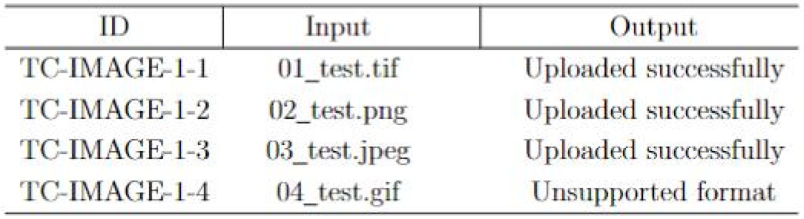
2. ground truth&segmentation result

3. metrics and observations

Accuracy:	0.9578
Sensitivity:	0.8288
Specificity:	0.9701
F1-Score:	0.7783

test-id1: Image Upload and Format Validation

The automation script will attempt to upload images in each specified format (JPEG, PNG, and GIF format). The system responds to each uploaded image and compares it with the expected results.

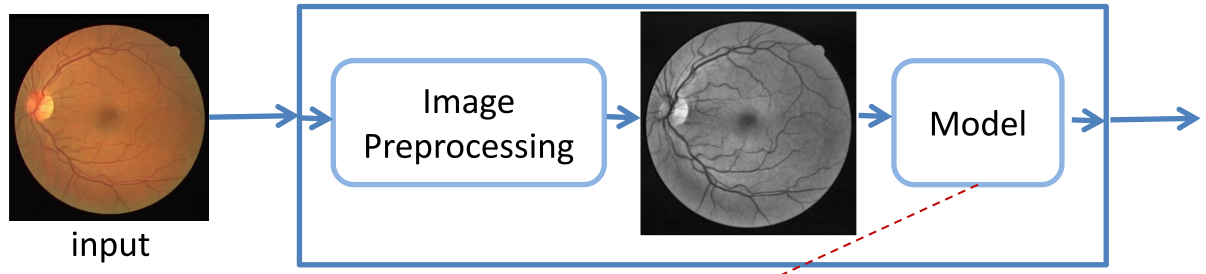


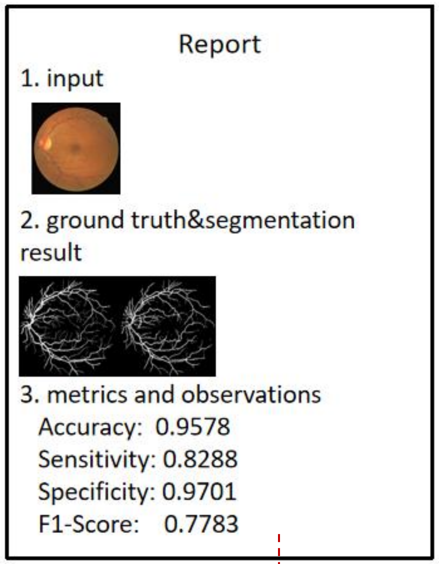
ID	Input	Output
TC-IMAGE-1-1	01_test.tif	Uploaded successfully
TC-IMAGE-1-2	02_test.png	Uploaded successfully
TC-IMAGE-1-3	03_test.jpeg	Uploaded successfully
TC-IMAGE-1-4	04_test.gif	Unsupported format

test-id2: Image Quality Preprocessing

A set of raw fundus images will be processed through the system's preprocessing function. The output will be analyzed using image analysis software to quantify changes in brightness, contrast, and noise levels

Tests for Functional Requirements




Report

1. input

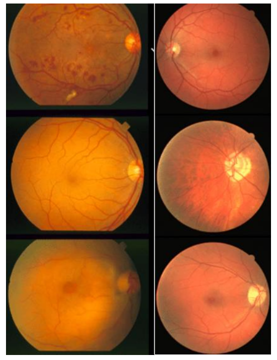
2. ground truth&segmentation result

3. metrics and observations

Accuracy:	0.9578
Sensitivity:	0.8288
Specificity:	0.9701
F1-Score:	0.7783

test-id3: Segmentation Image
The automation script will attempt to upload preprocessed images into the system and collecting the output segmented images.

test-id4: Segmentation Performance under Varying Conditions
Input fundus images with various conditions, including low contrast, presence of noise, and varying lighting conditions. Test the robustness of the segmentation network.



test-id5: Report Generation
The reports will include existing and commonly used analytical criteria to ensure that all necessary information is included and clearly presented.

Tests for Nonfunctional Requirements

Nonfunctional Requirements: Accuracy

test-id6: Accuracy Test

$$ACC = \frac{TP + TN}{\text{the number of all pixels}} \times 100\% > 90\%$$

TP (true positive)	the number of pixels that belongs to vessels and also classifies them as the same
TN (true negative)	pixels that are predicted as background and belong to it

Example: segmentation result and ground truth 584pixels $\times$ 565 pixels

compare each pixel, 310000pixels are correctly classified

$$ACC = \frac{310000}{584*565} \times 100\% = 93.95\% > 90\%$$

pass the test

Nonfunctional Requirements: Usability

test-id7: Usability Test

recruit volunteers, prepare a set of predefined tasks that cover the key functionalities of the system, develop a usability survey that includes questions on ease of use, interface design, functionality satisfaction, and overall user experience.

Tests for Nonfunctic   ıts

Nonfunctional Requirements: Accuracy

test-id6: Accuracy Test

$$ACC = \frac{TP + TN}{\text{the number of all pixels}} \times 100\% > 90\%$$

TP (true positive)	the number of pixels that belongs to vess
TN (true negative)	pixels that are predicted as background

Example: segmentation result and ground truth 584pixel
compare each pixel, 310000pixels are correctly

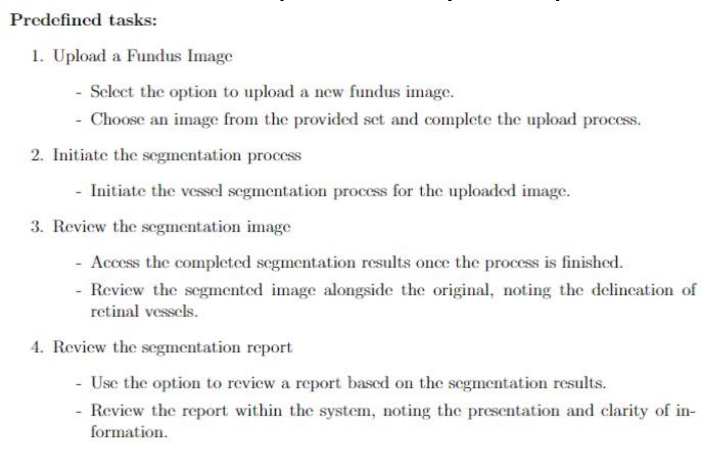
$$ACC = \frac{310000}{584 * 565} \times 100\% = 93.95\% > 90\%$$

pass the test

Nonfunctional Requirements: Usability

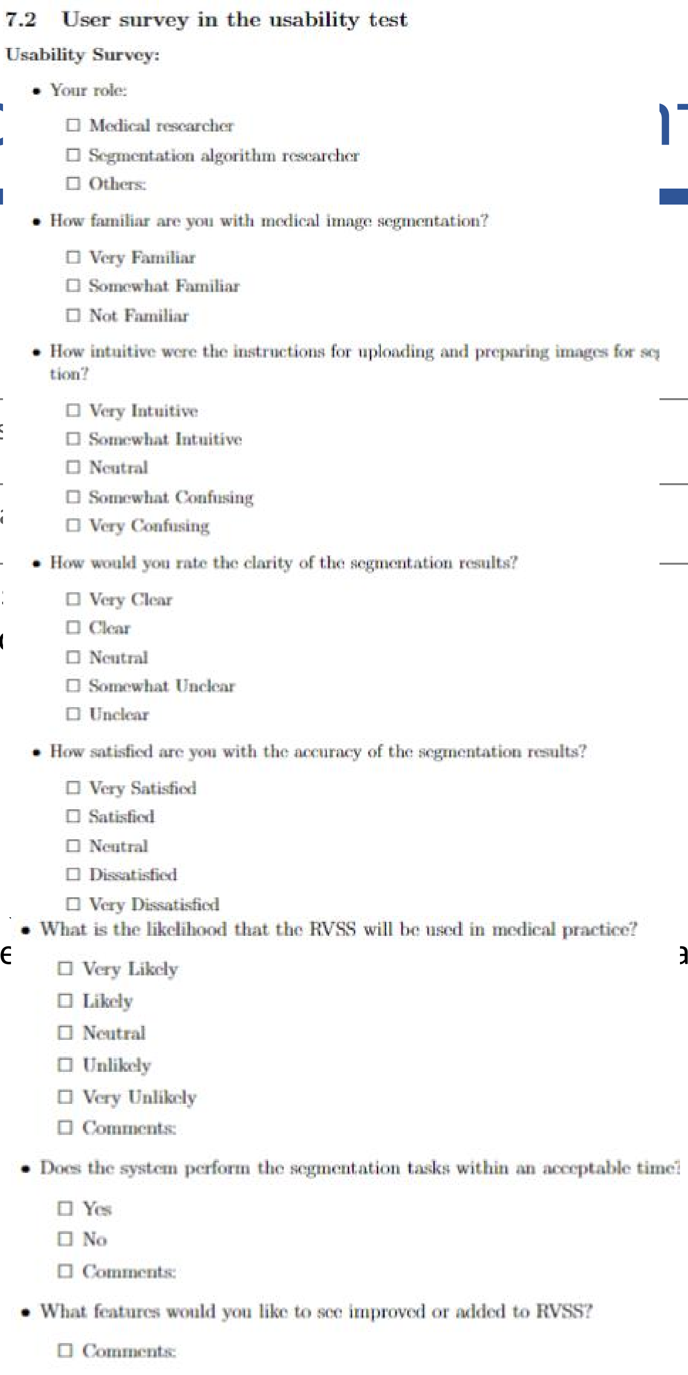
test-id7: Usability Test

recruit volunteers, prepare a set of predefined tasks that
develop a usability survey that includes questions on ease   action,



Predefined tasks:

1. Upload a Fundus Image
   - Select the option to upload a new fundus image.
   - Choose an image from the provided set and complete the upload process.
2. Initiate the segmentation process
   - Initiate the vessel segmentation process for the uploaded image.
3. Review the segmentation image
   - Access the completed segmentation results once the process is finished.
   - Review the segmented image alongside the original, noting the delineation of retinal vessels.
4. Review the segmentation report
   - Use the option to review a report based on the segmentation results.
   - Review the report within the system, noting the presentation and clarity of information.



7.2 User survey in the usability test

Usability Survey:

- Your role:
  - ☐ Medical researcher
  - ☐ Segmentation algorithm researcher
  - ☐ Others:
- How familiar are you with medical image segmentation?
  - ☐ Very Familiar
  - ☐ Somewhat Familiar
  - ☐ Not Familiar
- How intuitive were the instructions for uploading and preparing images for se tion?
  - ☐ Very Intuitive
  - ☐ Somewhat Intuitive
  - ☐ Neutral
  - ☐ Somewhat Confusing
  - ☐ Very Confusing
- How would you rate the clarity of the segmentation results?
  - ☐ Very Clear
  - ☐ Clear
  - ☐ Neutral
  - ☐ Somewhat Unclear
  - ☐ Unclear
- How satisfied are you with the accuracy of the segmentation results?
  - ☐ Very Satisfied
  - ☐ Satisfied
  - ☐ Neutral
  - ☐ Dissatisfied
  - ☐ Very Dissatisfied
- What is the likelihood that the RVSS will be used in medical practice?
  - ☐ Very Likely
  - ☐ Likely
  - ☐ Neutral
  - ☐ Unlikely
  - ☐ Very Unlikely
  - ☐ Comments:
- Does the system perform the segmentation tasks within an acceptable time?
  - ☐ Yes
  - ☐ No
  - ☐ Comments:
- What features would you like to see improved or added to RVSS?
  - ☐ Comments:

Tests for Nonfunctional Requirements

Nonfunctional Requirements: Performance

test-id8: Performance Test

evaluate the efficiency of the system

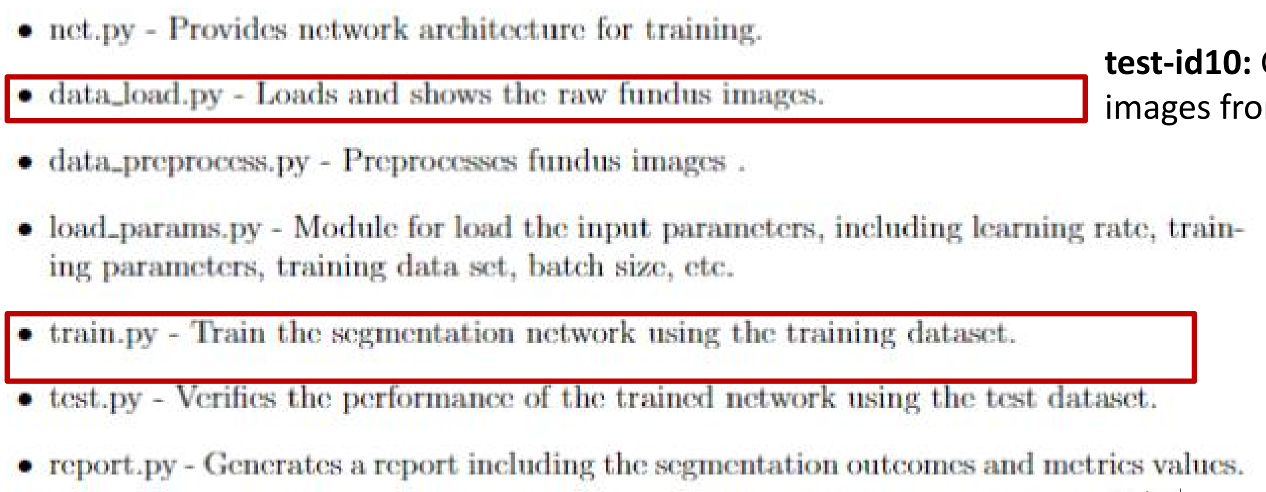
$$T_{single\_image} < 5s$$
$$T_{batch\_image} < min\{number\ of\ images\ \times 5s, 2\ \mathrm{minutes}\}$$

Nonfunctional Requirements: Compatibility

test-id9: Compatibility Test

install and run whole software in different operating systems.

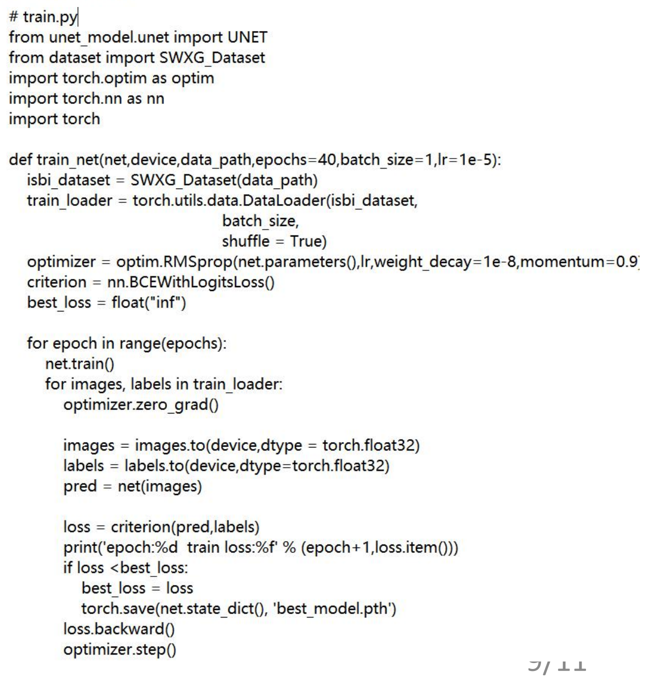
Unit Test Description



- net.py - Provides network architecture for training.
- data_load.py - Loads and shows the raw fundus images.
- data_preprocess.py - Preprocessses fundus images .
- load_params.py - Module for load the input parameters, including learning rate, training parameters, training data set, batch size, etc.
- train.py - Train the segmentation network using the training dataset.
- test.py - Verifies the performance of the trained network using the test dataset.
- report.py - Generates a report including the segmentation outcomes and metrics values.

test-id10: Check whether we can read images from the specified folder

test-id11: Check the model training process, output the value of each round of training process. For example, the output of the i-th round is “Epoch i: train loss is 0.214”.



```
# train.py
from unet_model.unet import UNET
from dataset import SWXG_Dataset
import torch.optim as optim
import torch.nn as nn
import torch

def train_net(net,device,data_path,epochs=40,batch_size=1,lr=1e-5):
    isbi_dataset = SWXG_Dataset(data_path)
    train_loader = torch.utils.data.DataLoader(isbi_dataset,
                                               batch_size,
                                               shuffle = True)
    optimizer = optim.RMSprop(net.parameters(),lr,weight_decay=1e-8,momentum=0.9
    criterion = nn.BCEWithLogitsLoss()
    best_loss = float("inf")

    for epoch in range(epochs):
        net.train()
        for images, labels in train_loader:
            optimizer.zero_grad()

            images = images.to(device,dtype = torch.float32)
            labels = labels.to(device,dtype=torch.float32)
            pred = net(images)

            loss = criterion(pred,labels)
            print('epoch:%d  train loss:%f' % (epoch+1,loss.item()))
            if loss <best_loss:
                best_loss = loss
                torch.save(net.state_dict(), 'best_model.pth')
            loss.backward()
            optimizer.step()
```

Automated Testing and Verification Tools

Pytest library: a Python testing framework

Flake8 : enforces coding standards and identifies stylistic errors, making the code more consistent and easier to maintain

Questions?