NATIONAL UNIVERSITY OF SINGAPORE

BACHELOR THESIS

Development of An AI-based Region Detection App. for Wound Healing Therapy

Author:
Fan HAOLIN

Supervisor:
Prof. Fuh YING HSI

Examiner:
Prof.Lu WEN FENG

A thesis submitted in fulfillment of the requirements for the degree of Bachelor of Mechanical Engineering

in the

Department of Mechanical Engineering

May 14, 2021

Development of An AI-based Region Detection App. for Wound Healing Therapy

by

Fan HAOLIN
B.S. *NUS:https://www.eng.nus.edu.sg/me/*

A Thesis Submitted in Partial Fulfilment
of the Requirements for the Degree of
Bachelor of Mechanical Engineering

at

National University of Singapore
May, 2021

Acknowledgements

First and foremost, I would like to show my deepest gratitude to my supervisors, Professor Fuh Ying Hsi and Examiner Lu Weng Feng,who are respectable, responsible and resourceful scholars, has provided me with valuable guidance in every stage of the Final Year Project.

Without their enlightening instruction, impressive kindness and patience, I could not have completed the task or complete the process of research direction determination and research content enrichment. Their keen and vigorous academic observation enlightens me not only in the FYP project but also in my future study.

I shall extend my thanks to Professor Zeng Kaiyang, Mrs. Wang Hui and Mrs.Zhang Qin for all their kindness and help. I would also like to thank all my teachers who have helped me to develop the AI-based Region Detection App. The sincere appreciation also goes to my parents, who support me unconditionally.

Last but not least, I would like to thank the NUSRI and NUS for giving me an unforgettable experience of the enjoyable project.

Fan HAOLIN
National University of Singapore,SuZhou Institution
May 14, 2021

Abstract of thesis entitled

Development of An AI-based Region Detection App. for Wound Healing Therapy

Submitted by

Fan HAOLIN

in May, 2021

The main research content of this paper is *Development of an AI-based region detection App. for wound healing therapy.* Its research purpose is to segment and analyze wound images through traditional computer vision technology based on OpenCV and deep learning technology, thus locating the wound region by image segmentation. The analysis results provide corresponding wound information (wound characteristics, dressing area required for 3D printing, etc.) and auxiliary diagnosis and treatment suggestions, and package this analysis system with mobile applications to explore new ideas for medical image processing.

Research methods mainly focus on traditional computer vision based on OpenCV and deep learning technology. In OpenCV, methods such as Gaussian filter noise reduction and Canny edge detection are used to locate the wound region and calculate the area of the wound region. In terms of deep learning, deep learning models such as UNET, MobileNetV2, and SegNet were used to train the data set. The trained model was used to segment the wound image, and the posterior parameters of the model were calculated according to the segmentation results to evaluate the pros and cons of the model. In the application of deep learning, the labeled data set was used for training, and the training accuracy of the model reached 97.87%, 93.65%, and 96.26%.

All in all, this project takes wound image segmentation as the core. At the same time, it has completed the App with functions of registration, login, customer service, weather forecast, medical record query, and display,

and completed the development of a Web-App that can quickly segment and analyze wound images. Furthermore, it compares the advantages and disadvantages of different mobile application forms for processing the same task and explores the diversity of medical image processing carriers. Through the above research, it is concluded that: 1. In order to meet the needs of today's mobile communication era, it is feasible and practical to develop mobile applications with medical image processing functions, such as providing auxiliary reference opinions for doctors' treatment and improving the efficiency of medical treatment while reducing the human error rate. 2. Deep learning has a good effect on the processing of medical images. Among them, the lightness of the UNET model and the adaptability of the MobileNetV2 model to mobile applications are more significant advantages. 3. Different mobile application carriers have multiple scenarios. Advantages and disadvantages, you need to choose according to different application scenarios.

*Keywords:*Kivy, App, Wound Image Segementation, OpenCV, Deep Leaning,

Contents

List of Figures

List of Tables

List of Algorithms

List of Abbreviations

AI	**A**rtificial **I**ntelligence
UI	**U**ser **I**nterface
DL	**D**eep **L**earning
CNN	**C**onvolutional **N**eural **N**etwork
FCN	**F**ully **C**onvolutional **N**eural **N**etwork
ROI	**R**egion **O**f **I**nterest
MD	**M**aterial **D**esign
DFD	**D**ata **F**low **D**iagram
CCL	**C**onnected **C**omponent **L**abeling
WCSS	**W**ithin **C**luster **S**um of **S**quares
IOU	**I**ntersection **O**ver **U**nion

List of Symbols

Chapter 5

G_{ij}	(i, j) element of the Gaussian kernel	—
σ	stan-dard deviation of the Gaussian kernel elements	—
M_{xy}	element of the processed image from the data matrix f	—
G_x	x-direction Sobel kernels	—
G_y	y-direction Sobel kernels	—
G	edge gradient of each pixel in the image	—
θ	direction of the edge gradient	—

Chapter 7

n	observation numbers	—
k	aimed clustering set numbers	—
x_i	element in observation set	—
S_i	element in aimed clustering set	—
μ_i	mean of points in S_i	—

Chapter 1

Introduction

1.1 Subject Background

Acute and chronic non-healing wounds are a serious problem in health care facilities, impacting millions of people worldwide. The main issue at this point is how to detect and assess the wound, further provide theoretical support for 3D printed biological consumables information.

Unlike acute wounds, chronic wounds fail to advance in an orderly and timely manner across the healing stages, such that the cost to health care is increased to milliards annually by hospitalization and additional medication[1]. Chronic wounds are a significant concern to public health and the economy as they have a detrimental effect on the quality of life of patients, inducing stress, social separation, and high care costs.

A significant number of wound patients lack access to advanced wound care and new protocols due to a shortage of well-trained wound professionals in primary and rural healthcare settings. Telemedicine device advancements will greatly benefit patients in remote locations, especially in rural areas, by providing better diagnostic advice [2]. With increasing uses of artificial intelligence (AI) technologies and portable devices such as smartphones, it is now timely to develop remote and intelligent diagnosis and prognosis systems for wound care.

An intelligent system can be extremely beneficial for wound care in many ways: improved precision, reduced workload and financial burden, standardized diagnosis and management, and higher quality of patient care [3]. Meanwhile, in today's information era, with the high efficiency and portability of smartphones or other electronic equipment, doctors can get

rid of computer shackles and complete activities such as wound diagnosis and electronic medical records easily and effectively.

1.2 The Purpose & Significance of the Research

Through the **identification** and **analysis** of the patient's wound, the characteristics of the wound, such as circumference, area, infection, swelling, and suppuration, can be obtained, the above information is supposed to be stored as well. In this way, using the predicted wound region, the area of the dressing required for 3D printing can be estimated, and the auxiliary reference opinions for the treatment of doctors can be provided, improve the efficiency of medical treatment and reduce the rate of human error. On this basis, the target functions are encapsulated in applications to meet the needs of today's mobile communications era.

Thus, the research significance of this project can be summarized as the following three points: (a) Reduce the stress created by a lack of medical services and provide reliable assistance for the patient rehabilitation process. (b) Self-diagnosis and care of patients in places where medical services are limited. (c) Explore the feasibility of integrating AI with smartphone apps.

Chapter 2

Literature Review

2.1 Research Status on Wound Healing Therapy

The current research on **Wound Healing Therapy** focuses mainly on wound segmentation rather than wound feature extraction as the complexity of the type of wound and lesion, which means that this research is more of an **image segmentation problem** rather than an image classification problem. In other words, the result of this research is predicting the contour and other relative information about the wound image. The wound segmentation of the pictures is a common approach not only to simplify wound area measurements but also to facilitate efficient data entry into the electronic medical record. It can be seen from the analysis of existing literature, related studies on wound image segmentation can be roughly categorized into two groups: traditional computer vision methods[4] and deep learning methods[5].

2.1.1 Computer Vision Methods

Studies in the 1^{st} category concentrate on integrating computer technology vision with conventional methods of machine learning. These studies use manually crafted extraction features for creating a data set which is subsequently used to assist computer algorithms.

Hettiarachchi *et al.* showed in its HSV color model an energy mitigating discrete dynamic contour algorithm applied to the image's saturation plane. The wound area within the enclosed contour is then measured from a flood fill[6].

Hani *et al.* suggested that an Independent Component Analysis (ICA) algorithm be applied to the pre-processed RGB images to create hemoglobin-based images that are used as a K-means clustering input to segment the granulation tissue from the wound images[7].

Wantanajittikul *et al.* proposed a system to segment the burn wound from images. Cr-Transformation and Luv-Transformation are applied to the input images to remove the background and highlight the wound region. The transformed images are segmented with a pixel-wise Fuzzy C-mean Clustering (FCM) algorithm[8].

2.1.2 Deep Learning Methods

Since the AlexNet9 achievements at the Imagenet 2012 challenge[9] of large-scale visual recognition, deep learning in computer vision has created interest in semantic segmentation by means of deep convolutionary neural networks (CNN)[10]. The usual emphasis is on extracting functionality through traditional machine learning and vision approaches[11].A selection of essential characteristics and then advanced algorithms to catch these traits must be devoured to segment the area of concern.However, the CNN includes the retrieval of features and decisions. During network preparation, the convolutionary cores of CNN derive the characteristics and their significance. The input is interpreted by a series of convolutionary layers in a standard CNN architecture and its output is regulated by a completely linked layer requiring fixed input.

One successful variant of CNN is fully convolutional neural networks (FCN)[12]. For instance, Wang *et al.*, by segmenting wounds[13] with the vanilla FCN architecture, estimated the wound location. Using a Gaussian process regression function model, wound healing progress is predicted with time-series data consisting of the approximate wound areas and corresponding images. The mean dice precision of the segmentation, however, is only evaluated to be 64.2 %.

As research progresses, Liu *et al.* suggested the latest FCN architecture to replace the vanilla FCN decoder with a concatenation skip-layer upsampled by bilinear interpolation[14]. At the end of the network, a softmax layer

pixel-wise is attached to create an assumption that the final segmentation after the processing is over. A dice precision of 91.6 percent is reached with 950 images taken in an unregulated and dynamic background lighting environment. Yet pictures in their data set are annotated using an algorithm semi-automatically. This suggests that the profound learning paradigm is learning to mark wounds in contrast to human experts on the watershed algorithm.

2.2 Research Status on Mobile-based Wound Analysis Systems

Mobile devices for wound assessment are very convenient to use, but they can also be very difficult to use due to issues with ambient illumination, background noise, and picture quality[15]. Some smartphone apps created for wound evaluations are briefly outlined in this subsection. Some apps are made only for wound image capturing. For example, Yap *et al.* developed a foot imaging app called "FootSnap" using the iPad to standardize image capturing of the plantar surface of diabetic feet[16].

Some smartphone applications have been created especially for measuring and detecting wounds. Manuel Dujovny tested a smartphone wound analyzer software called MOWA in [17], which can measure three tissue colors (black, yellow, and red), ulcer scale, and wound edge dimensions. To produce the wound size at the edge of the wound, a freehand drawing is needed. This app will also include clinical care recommendations based on international standards.

Luay *et al.* created a mobile app to detect ulcers. This software includes a handheld thermal camera (FLIR ONE) for image acquisition and can detect ulcerous areas on the feet dependent on temperature differences[18]. About the fact that it can detect the ulcer and its position, this app would not operate on a person who only has one leg or if the features of two legs are identical and both have foot problems.Some other smartphone apps are designed for wound healing assessments.

Hettiarachchi *et al.* created an android smartphone framework for wound segmentation and chronic wound monitoring[6]. To eliminate the impact of camera distance and angle, wound areas are normalized. Friesen *et al.* developed an android app for chronic wound care, which allows add a new patient, store the patient information, update wound evaluations, store the records of chronic injuries, and compare them week-by-week[19].

Chapter 3

Software Requirement Specification

This chapter introduce the specification in software aspect,which means it gives an outline during designing and modifying the application.

3.1 Project Objective Description

3.1.1 Intended Audience

Main Target Audiences of the application are medical professionals and wound patients. Meanwhile, anyone else who might benefit from the features is the secondary users, where the App can be utilized as self-diagnostic tools. There specific domain for this application is medical research for wound healing. The user could upload or take the picture of the wound and then use the application to analyze the image, the report will be generated in PDF format based on the analysis results.

3.1.2 Product Scope

The scope of this project is to provide an efficient and enhanced software tool for the users to segment and analyze the wound image, thus providing auxiliary diagnosis and treatment opinions as much as possible based on the analysis results The major scopes of this project are demonstrated as follow:

- Upload wound image via album or taking photo by camera
- Create patient digital medical record
- Analyze the wound image, gain the wound image area and wound feature

- Store the analysis result under corresponding patient
- Traverse and access the electronic medical record

3.2 Overall Description

3.2.1 Design & Implementation Constrain

The following constrains are essential to be aware of during the design and implementation process, which are listed as below:

1. The pictures uploaded or taken by the user should be as good as possible to ensure that the picture is well lit, the image quality is clear, and the picture background is as small as possible.
2. When creating patient medical records or entering passwords, the user needs to pay attention to the correct filling of relevant information, to ensure timely and accurate traversal of relevant information.
3. When using the application, the correct connection to the Internet is needed to ensure, because the source of some asynchronously loaded images is COS (Object Storage Database).
4. The final result of the project should be adapted to the mobile terminal at least, which means that attention should be paid to the layout of the design and the compatibility of different devices.

3.3 External Interface Requirements

The following is a list of the external interfaces used by the application, which contains the user interfaces, hardware interfaces, and software interfaces:

3.3.1 User Interfaces

The user interfaces used for this system is mainly based on Kivy, and the material design is based on KivyMD(Material Design Standards as per Google). The detailed information is illustrated as the Table 3.1 below:

Table 3.1: User Interfaces for Desired Application

Front-end Software	Back-end Software
Kivy v2.0.0 KivyMD 0.104.2.dev0	Python v3.6.2 (PyCharm 2020.3.3 x64)

3.3.2 Hardware Information & Software Interface

Hardware Information

The realization of the project is based on the 64-bit operating system (x64-based processor) of the windows operating system, the hardware configuration of the computer are: CPU is Intel(R) Core(TM) i7-8700 CPU @ 3.2 GHz, and graphics card is configured to AMD Radeon R5 430.

Software Interface

Based on the operating system (operating environment) and the necessary dependencies, list the requirements of the software interface which used for the application maintenance as follows:

1. Operating system - Android for its best support and user-friendliness. But it also supports platforms like Mac OS, Windows for operation.
2. Python for basic coding with wrappers and modules like Kivy for interface design, Pillow and OpenCV for image processing, Tensorflow and Keras for deep learning image segmentation. Specificlly,the required python dependency library name and corresponding version number are shown in Table 3.2:

Table 3.2: Required Python Dependencies & Version

Required Site-packages	Corresponding Version Number
OpenCV for Python	4.5.1
Tensorflow	1.15
Keras	2.3.1

3.4 Non-Functional Requirements

3.4.1 Use Case Diagram

A complex or behavior diagram is referred to as an usage case diagram. Actors and use cases are used in use case diagrams to model a system's functionality. A collection of actions, programs, and functions that the device must perform is referred to as a use case. The use case for the App is illustrated as Fig. 3.1 below:

3.4.2 Safety & Security Requirements

When developing apps, especially for medical imaging apps, the protection of patient privacy is particularly important. Therefore, security (data storage, data access, etc.) should be considered when designing. Specifically, it can be summarized as the following two points:

- The App should be capable to prevent illegal access to camera, microphone, and other hardware when app is not used.
- Due to the privacy of the patient's wound image, the uploaded images must be safe which are inaccessible for the users who haven't logged in.

3.4.3 Software Quality Attributes

The ideal App should have the following quality attributes, which are as stated below:

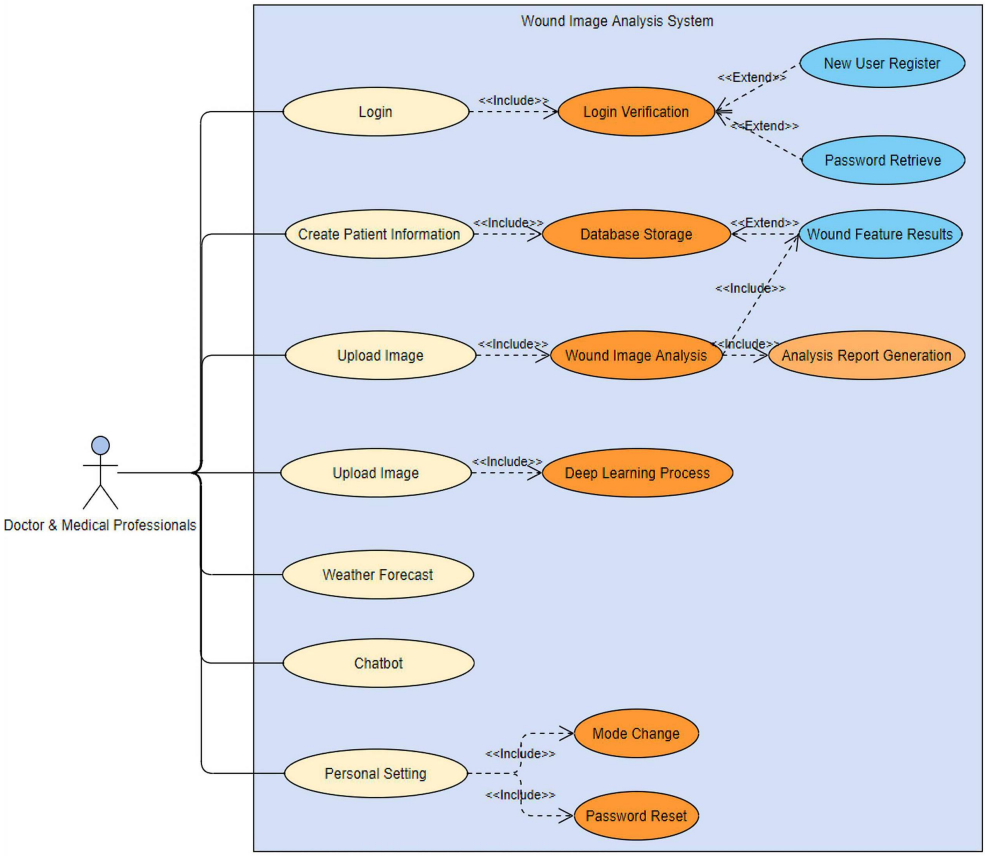


Figure 3.1: The Use Case Diagram for Desired Application.

- **Availability**: After the customer register and log in successfully in the app it can be used 24/7. This availability is the primary consistency ensured by this app.
- **Accuracy and Reliability**: The software's features are precise and dependable. The input revealed incorrect workings, which were corrected by the meantime.
- **Maintainability**: The app is maintained and regular update versions are provided, which is reflected in the core functions of application, response time and interactive experience.

3.5 System Design Using DFD

DFD (data flow diagram) is a diagram that represents the data flow of a process or system. DFD also contains data on each entity's outputs and inputs, as well as the process itself. There are no control flows, decision rules, or loops in the data flow graph. A flowchart can depict the basic activity based on the details.

3.5.1 Level 0 DFD

The structure of level 0 DFD is illustrated as Fig. 3.2,where the core functional element is Wound Image Analysis System.

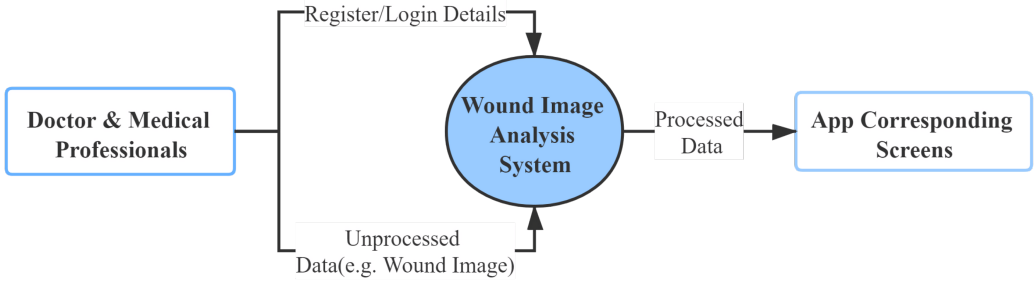


Figure 3.2: The illustration of Level 0 DFD diagram.

3.5.2 Level 1 DFD

Based on the level 0 DFD, expand 'Wound Image Analysis System' process into the following modules, which are aligned with the use case in Chapter 3.4.1:

1. Register
2. Password Retrieval
3. Login
4. Core Category
    - Image Segmentation module based on OpenCV
    - Image Segmentation module based on Deep Learning
5. Relevant Tool
    - Weather Forecast
    - Personal Setting
    - Chatbot Help Service Module
    - About Module

The level 1 DFD is drawn based on the above content, and the drawing result is shown as Fig. 3.3:

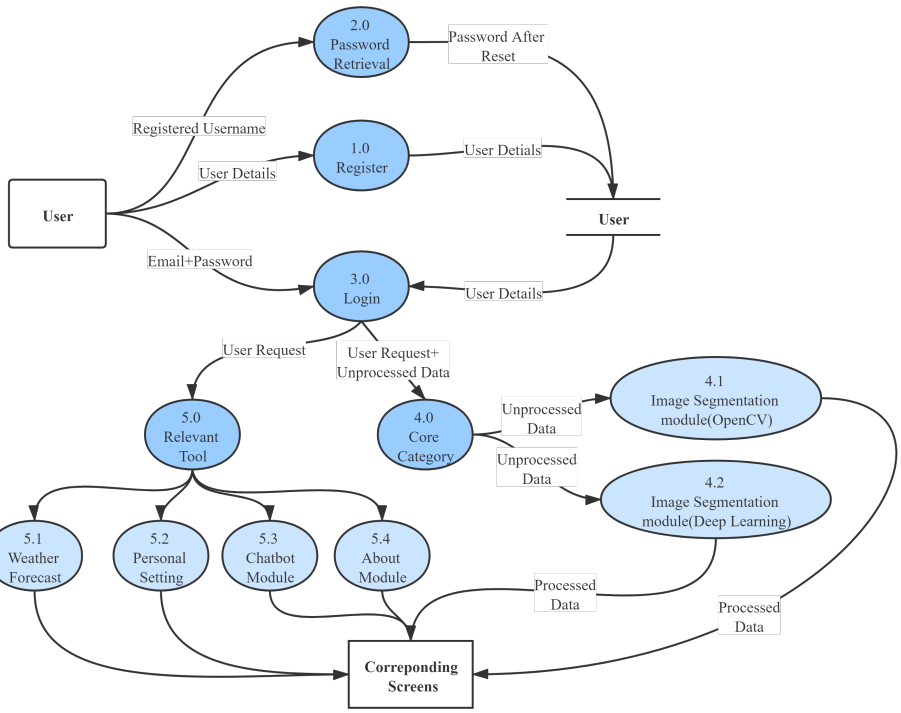


Figure 3.3: The illustration of Level 1 DFD diagram.

Chapter 4

Detailed Implementation of UI & Prompt

4.1 The Realization of Basic Component

The App is disassembled into separate components for presentation in order to briefly and intuitively explain the basic composition of the App. This section gives an introduction of the basic components(modules) in application, which contains the modules listed as follows:

4.1.1 Login Verification Module

In order to make each user's data stored independently and privately, the first basic component of the application is the login verification module,which include the User Creation, Password Retrieval,Login(via local database or Github).

The User are supposed to type the registered e-mail and corresponding password to log in the app. And the icon buttons on the top of the interface can be used to jump to the corresponding pages.

The detailed layout of the login verification includes the icon button at the right of each text-field.The drop-down icon on the right of the first text-field can be used to select a user email which is remembered in advance, the eyes on the right of second text-field can be clicked to hide the password.

Meanwhile,The user icon in the upper left corner can be used to create a new user and store it. At the same time, this page can jump to the password recovery module to provide password recovery services for registered users while clicking the button on the toolbar.

The detailed design process and updating history of the module will be posted on the personal website. The use case flow for login module which consists of User Creation and Password Retrieval is illustrated as Fig. 4.1.

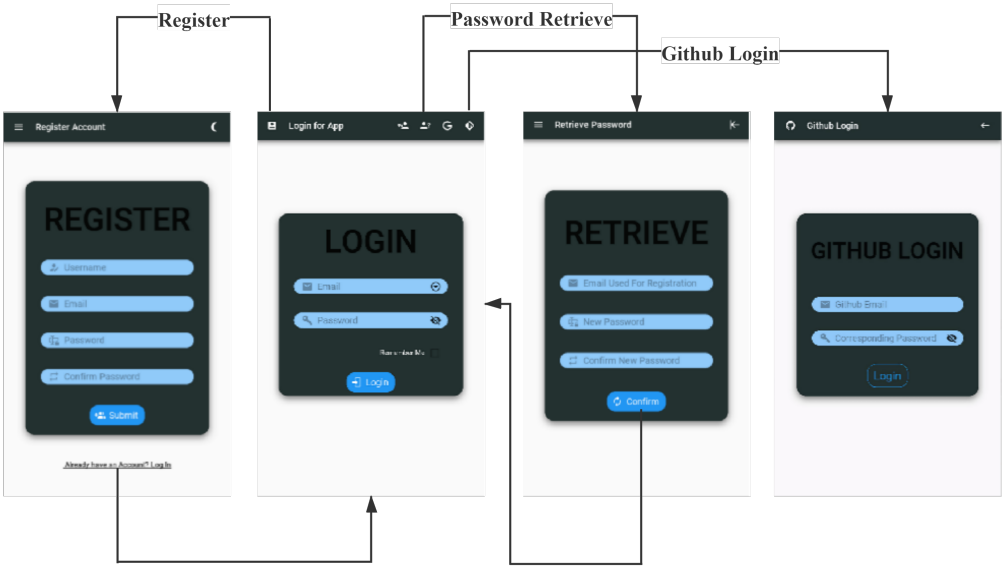


Figure 4.1: The illustration of Login Verification module.

4.1.2 Patient Creation Module

The Patient Creation Module is designed with the ability to store patients information via sqlite which is Python's built-in database.

This step is essential in the **wound image analysis system** in order to generate proper patient report. The basic patient information includes name, age, gender and creation date. The information will be stored in the database if input information is in right form.Otherwise,the system will reject the creation request if input information is malformed or missing (e.g.null or the input gender is not in the format with 'Male' or 'Female'). The case flow diagram for patient creation module is illustrated as Fig. 4.2.

At the same time, in the following analysis, the patient name entered at this stage is regarded as a unique identification until the analysis process completed, especially in the process of generating patient reports, displaying analysis result, and traversing the database to obtain all patient information.

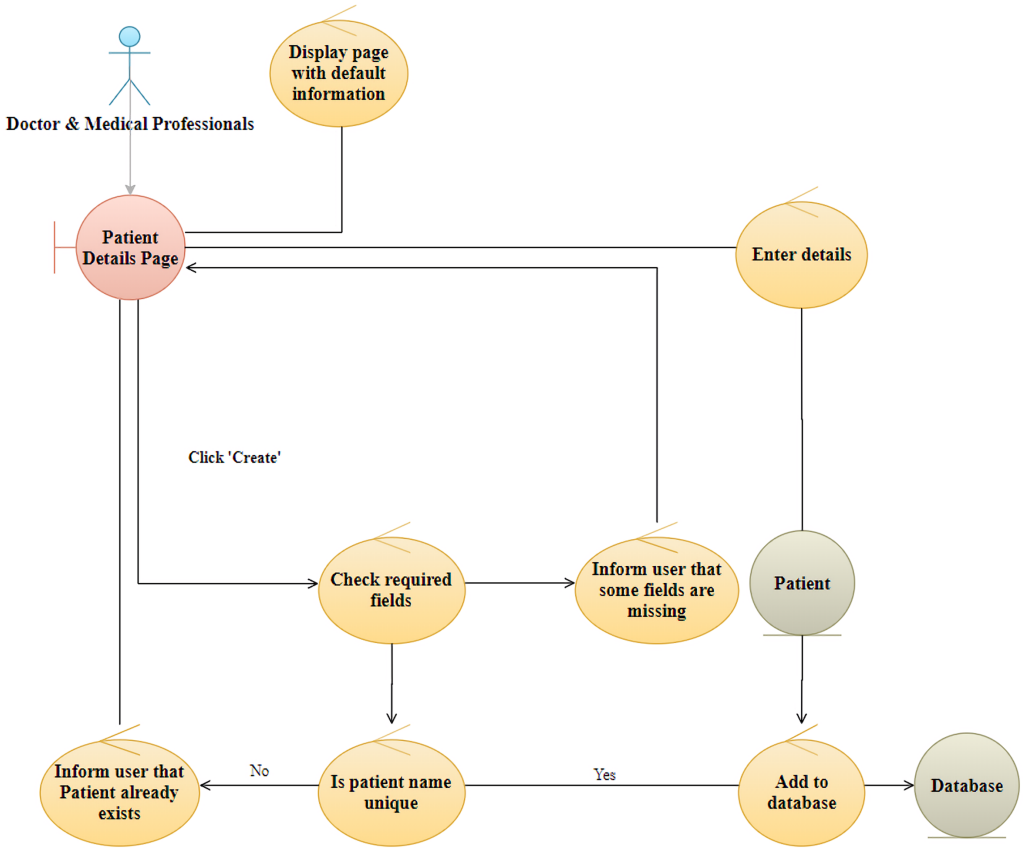


Figure 4.2: The illustration of Patient Creation flow diagram .

The interface of this module is demonstrated as Fig. 4.3 below, and the structure and data in the database after creation is illustrated in the Appendix B.

4.1.3 Home Page & Navigation Drawer

After finishing the creation of patient information, the user can reach to the app's home page, where shows the relevant knowledge using in this application with the form of tags with different colors. By clicking the button on the left corner of the home page, the user can get access to the navigation drawer.

To jump to the corresponding page with different functions, the navigation drawer can be taken into account. In the figure below, the page implementation is shown as Fig. 4.4.

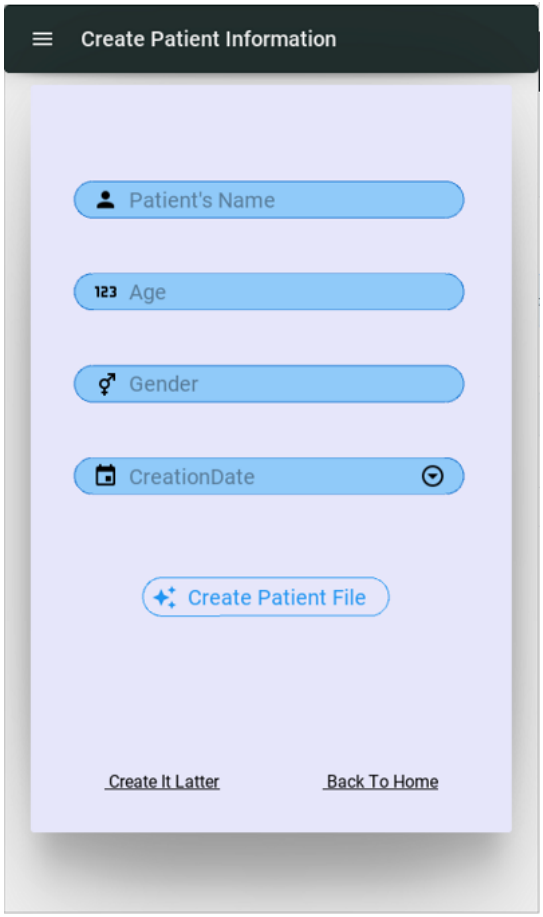


Figure 4.3: The illustration of Patient Creation Interface .

As the diagram implies, the navigation drawer can reached to the pages which have the functions like:

- Wound image upload,image segmentation and analysis
- Viewing patient history and detailed information about wound image
- Weather forecast
- Personal setting
- Chat-bot used as customer service
- Viewing the relevant information about the application
- Exit the application

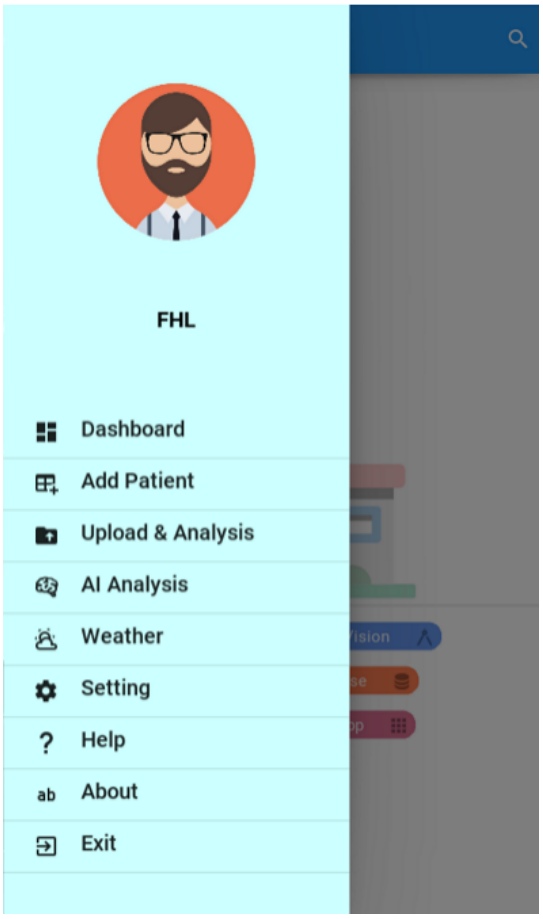


Figure 4.4: The illustration of Home Page & Navigation Drawer.

Meanwhile, the avatar and user name can be changed in personal setting page and callback to the navigation drawer. The detailed illustration of the home page and navigation drawer can be reached at the personal blog site as well.

4.1.4 Wound Image Upload & Analysis:OpenCV Method

The image upload and inspection process is the most important feature of this application. This process is realized using asynchronous processing tasks. It is noteworthy that the benefit of asynchronous processing is that it increases the usage rate of resources, thus increasing the overall quality of software activity.

The wound images uploaded by the doctors or medical professionals will be stored in local address and COS with the unique path of based on Patient Name.

The file manager looms large in this module, by adjusting parameter in (.kv) file,the manager could show the images only. It should be emphasized that when packaging the application, it is vital to point out the local storage address, which is default as '/storage/emulated/0'.The interface of the file manager is shown as Fig. 4.5 below:

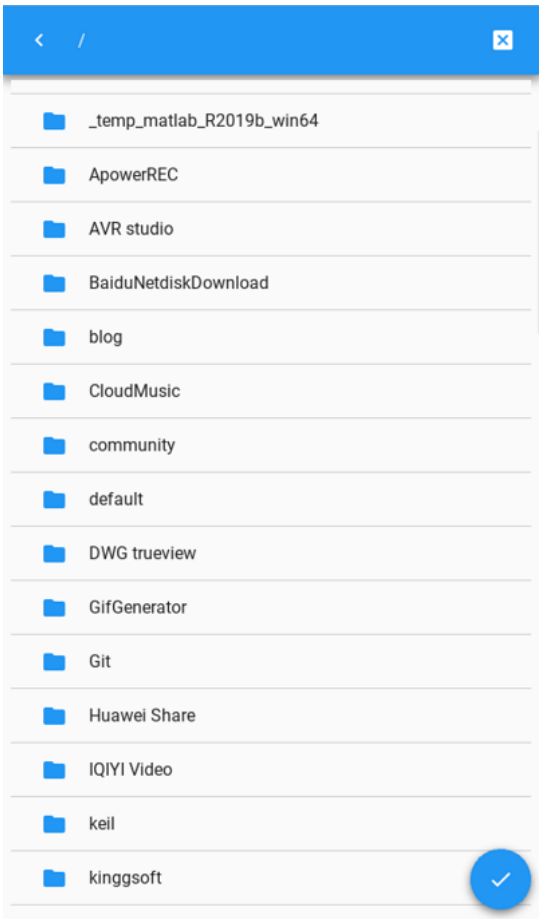


Figure 4.5: The interface of File Manager.

Meanwhile, the analytical result and segmentation result will be stored in the local database and then stored into the COS, which is easier to callback to the analytical result page with the image source based on the URL link. The detailed result will be introduced in Section 6.1.1 and 6.1.2.

The app also provide the function of generating medical report, doctors and the medical professionals can click the button to generate the wound image analytical report, which can be sent via e-mail to patient's personal e-mail box. The template of the generated report is illustrated in Fig. 6.2(b).

4.1.5 Wound Image Upload & Analysis:DL Method

The interface based on deep learning method is similar to the former OpenCV process.The drop-down page is utilized as menu which enable users to choose the different deep-learning model, choose the method of image upload (album or photo), click on the wound segmentation to gain the result. Similarly, the result page will show the segmentation result and the precision of the model while segmenting the uploaded image.

4.1.6 Others Functions Covered By The App

Besides the interface realization of the main task of the application which is introduced in 4.1.4 and 4.1.5, it also has some other useful tools like: Weather forecast, chat-bot as customer service, personal setting and the button used to exit the App.

In greater details, the personal setting function includes avatars change, password visibility change, password reset, mode change, and backup account switch.The detailed implementation of these functions and corresponding interfaces will be showed in the Appendix B .

4.2 The User Prompt Information & Dialog

Taking into account that in the actual use of the App, there may be an error message caused by the user's operation error, which affects the user's experience.

At this time, prompt information and dialog boxes are more important to ensure the reliability of related information acquisition and storage. In addition, when performing sensitive operations such as logout, password

reset, and account switching, the prompt box is used to confirm user operations to avoid misoperation or data loss and other abnormal situations. The following Fig. 4.6 is an example, which shows the confirmation prompt box while exiting the App.

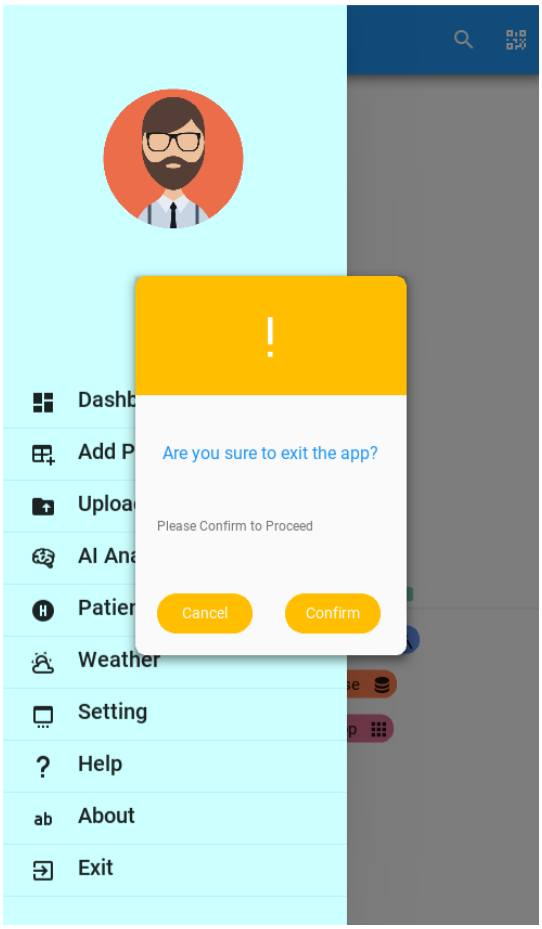


Figure 4.6: The exit app User Prompt.

Chapter 5

Experimental Methodologies

This chapter will introduce the basic principles and methods used in the process of wound image segmentation, which are related to 4.1.4 and 4.1.5 and acted as the back-end of wound analysis system. This chapter mainly consists of two methods used for wound image segmentation, which are OpenCV and DL.

5.1 Wound Image Analysis Based on OpenCV

5.1.1 Wound Contour Locating

Fundamental Process Flow

The beginning step is reading the image.which includes the detailed steps as follow. First, use the command *cv2.imread* to read the wound image with three channels in BGR. The second step is transferring the original image into RGB three channels. The third step is transferring the RGB three channels into gray-scale image.

The next step is utilize the threshold onto the image in gray-scale based on *cv2.thresh(src, thresh, maxval, type)*.It is worth noting that the setting of the maxval means that if the value of pixel is smaller than the maxval,the value of the pixel will be set to 0.The threshold operation will generate a binary image,where binary image occupies a very important position. The binary image of the image greatly reduces the amount of data in the image, which can highlight the contour of the target.

Further,the *bitwise-not* operator is used to perform binary "not" operation on each pixel value of the image.On this basics,Morphological manipulations of images such as dilation and corrosion are taken into account to process the element in the binary image.

The edge detection of the image is achieved by using two appropriate thresholds in the OpenCV Canny function based on Canny Edge[20], whose principle is illustrated in the subsection of 5.1.1.

The final contour of the wound image is realized by *cv2.findContours* in OpenCV, and contours whose contour area is less than a certain value are ignored, so as to locate the contour of the wound in the image.

Canny Edge Detection

The method of Canny operator edge detection is to find the local maximum of the image gradient.The Canny method uses two thresholds to detect strong and weak edges respectively, and only when the strong and weak edges are connected, the weak edges will be included in the output. Therefore, this method is not susceptible to noise interference and can detect real weak edges. The algorithm flow is shown in Fig. 5.1.

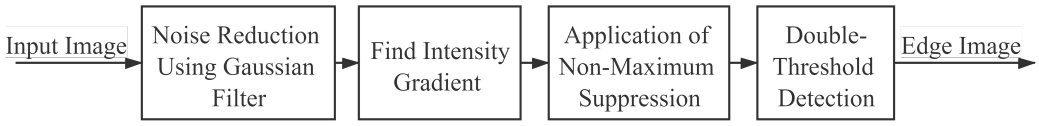


Figure 5.1: The flow chart diagram for Canny Edge Detection.

During the 1^{st} step, the Gaussian filter can be used to suppress noises because edge detection is vulnerable to picture noise.In this project, a $(2k+1) \times (2k+1)$ Gaussian kernel was used, as presented in Eq. 5.1 [21]:

$$G_{ij} = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{\{i-(k+1)\}^2+\{j-(k+1)\}^2}{2\sigma^2}}; 1 \leq i, j \leq 2k+1 \tag{5.1}$$

where G_{ij} is the (i, j) element of the Gaussian kernel, and σ is the standard deviation of the Gaussian kernel elements. The larger the Gaussian

kernel size, the lower the detector's sensitivity to noise. A 5×5 size kernel ($\sigma = 1.4$) was applied as presented in matrix 5.2:

$$G = \frac{1}{159} \begin{bmatrix} 2 & 4 & 5 & 4 & 2 \\ 4 & 9 & 12 & 9 & 4 \\ 5 & 12 & 15 & 12 & 5 \\ 4 & 9 & 12 & 9 & 4 \\ 2 & 4 & 5 & 4 & 2 \end{bmatrix} \tag{5.2}$$

Eq. 5.3 was operated on each pixel of the image to eliminate gaussian bleeds.If the distance (u, v) between the kernel's central position (x, y) and the kernel position $(x + u, y + v)$ grows, the weight decreases.where M_{xy} is the element of the processed image from the data matrix f .

$$M_{xy} = (f \otimes G)_{xy} = \sum_{W} f_{x+u,y+v} \cdot G_{uv} \tag{5.3}$$

The gradient of the image in which the noise has been removed by a Gaussian filter was calculated by applying 3×3 size x- and y-direction Sobel kernels obtained [22] (5.4):

$$G_x = \begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix}, G_y = \begin{bmatrix} 1 & 2 & 1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{bmatrix} \tag{5.4}$$

More detailed, the edge gradient and direction for each pixel can be calculated as Eq. 5.5 and Eq. 5.6, where G is the edge gradient of each pixel, and θ is the direction of the edge gradient:

$$\text{Gradient } (G) = \sqrt{G_x^2 + G_y^2} \tag{5.5}$$

$$\text{Angle } (\theta) = \tan^{-1}\left(\frac{G_y}{G_x}\right) \tag{5.6}$$

During the Canny Edge Detection,the 3^{rd} step's practical principle is: if the center point (that is, the access point) has the largest gradient amplitude in the neighborhood along its direction, it is retained; otherwise, it is suppressed. Thus the non-maximum suppression of the amplitude image is obtained. Briefly, pixels with smaller values than the maximum were then removed by nullifying those pixels with zero value(Fig. 5.2).

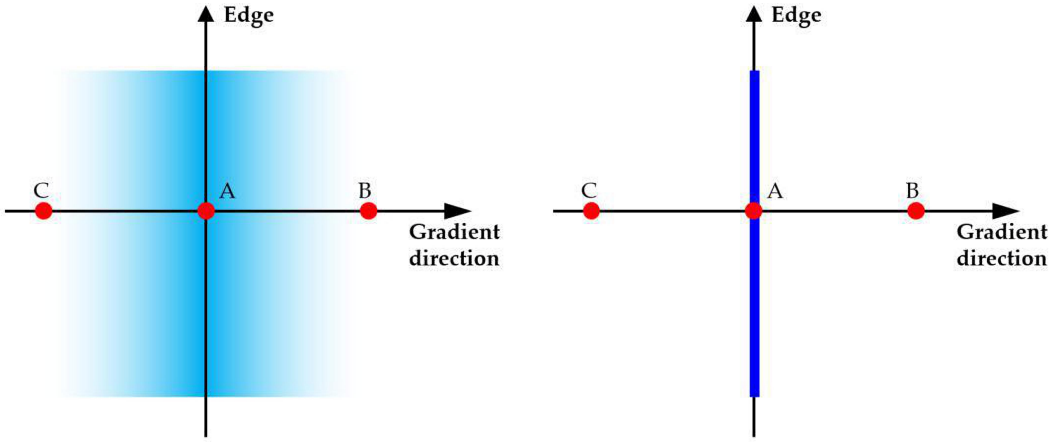


Figure 5.2: A graphic example of non-maximum suppression, where A, B, and C represent three gradient points. If the value at point A is the largest, then A becomes an edge pixel, and points B and C are nullified.

Typically, for the 4^{th} step, the Double-Threshold Detection looms large in the edge detection process, which is based on the algorithm 1[1] below.

Algorithm 1: Double-Threshold Detection in Canny Edge

Input: The content of a given input image

Output: Filter edge pixels with weak gradient values, and retain edge pixels with high gradient values

```
1 if G ≥ HighThreshold then
2     G is an strong edge ;                    // Justify Strong Edge
3     else if G ≤ LowThreshold then
4         G is an weak edge ;                    // Justify Weak Edge
5         else
6             G should be suppressed
7         end
8     end
9 end
```

5.1.2 Wound Feature Detection

After locating the wound region,it is vital to analyze the feature of the wound in practical medical treatment, the HSV three-channel-format[2] of the image is taken into account during the analysis of the feature.

First, the RGB three-channel image is converted into an HSV three-channel image, and then on the basis of the HSV three-channel image, an empty array is defined, and the values corresponding to the red area are stored in the array. Further, the values in the array are added to the three channels H, S, and V respectively. And the variances of the values in the HSV three-channel array are used *numpy.var* to plot and analyze. The HSV variance result will be demonstrated in Fig. 6.3.

After achieving the HSV variance diagram,the analytical result can be achieved via judging the range of each channels, which means the swelling, pus and infection of the wound are measured according to the features of

[1] G in the algorithm represents edge gradient of each pixel

[2] The meanings in the HSV three-channel color space are: H means the color/chroma of the image; S means the saturation of the image; V means the brightness of the image.

the variation. For instance, the Pus feature can be determined by the yellow component in H values and S values as well as the variance of the S values. Since the results of the wound characteristics evaluated by this system may include errors, the App adds the function of optimizing the findings depending on the real diagnosis for trained medical professionals, and the results are called back to the database.

5.2 Wound Image Analysis Based on DL

5.2.1 Data Set Establishment

Data Set Acquisition

During training, a deep learning model will learn the annotations of the training data-set. The accuracy of the annotations is therefore important. Automatic annotation developed using the computer vision anything similar is not an appropriate way to learn how human experts identify the area of the wound while training profound learning models. The images were manually annotated in the data set with segmentation masks.

The way to obtain this data set is the network and related research institutions, with 80% as the training set and 20% as the verification set, and the training set and the verification set are marked separately. At present, the commonly used labeling tool is Labelme [23], and the labeling results are stored in json format, and json files are output as png format images in batches.

Data Augmentation

Standard data augmentation techniques are added to the data set until it is fed into the deep learning model, in addition to the cropping and zero-padding used in the data set construction portion. Random rotations in the range of -25 to +25 degrees, random left–right and top–down flippings with a chance of 0.5, and random zooming within 80 percent of the initial image region are among the image transformations available.

Since it is suspected that other non-rigid transformations, such as shearings, do not reflect normal wound shape variants, random zooming is used as the only non-rigid transformation. The training data set is eventually expanded to about 3000 photographs.

5.2.2 Model Architecture Overview

UNET Model

The 1^{st} model used in wound segmentation is called UNET [24], and the architecture of the model is illustrates as Fig. 5.3 below:

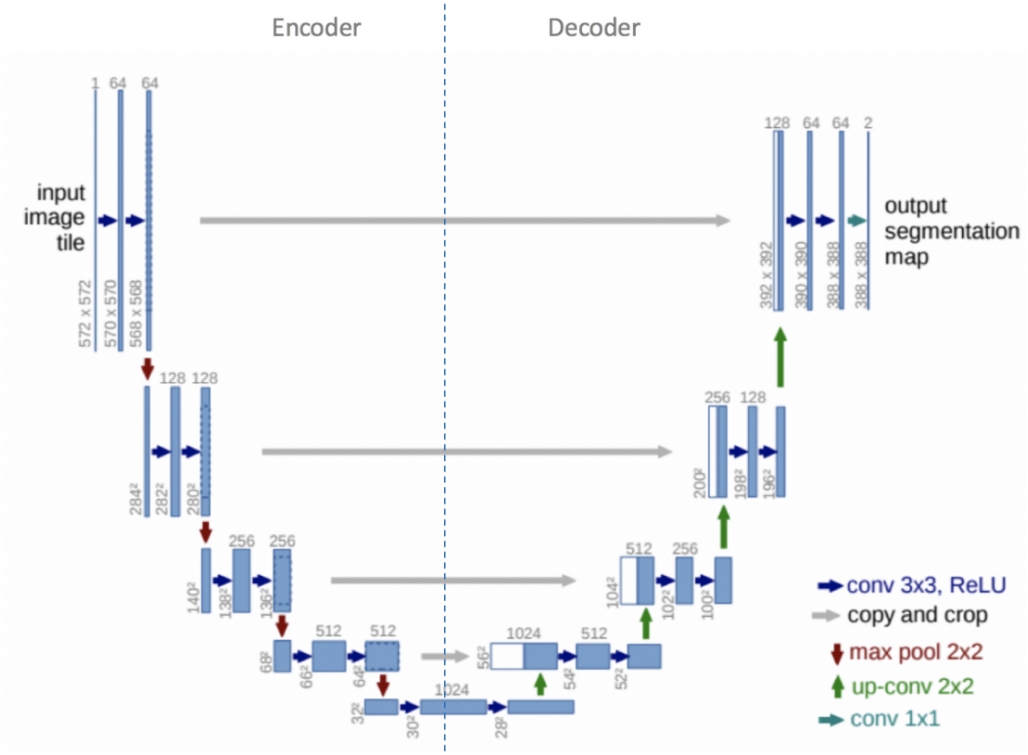


Figure 5.3: The structure of UNET model.

As it showed in the figure, it is made up of a contracting direction (left side) and an expansive path (right side). The convolutional network's contracting direction resembles the traditional architecture. It consists of two 3×3 convolutions (unpadded convolutions) that are implemented twice, each preceded by a rectified linear unit (ReLU) and a 2×2 max pooling operation

with stride 2 for down-sampling. the number of function channels are doubled with each down-sampling phase. An up-sampling of the feature map is followed by a 2×2 convolution ("up-convolution") that halves the number of feature channels, a concatenation with the correspondingly cropped feature map from the contracting direction, and two 3×3 convolutions, each followed by a ReLU in the expansive path. Due to the loss of boundary pixels in any convolution, cropping is needed. A 1×1 convolution is used at the final layer to assign each 64-component function vector to the desired number of groups. The network has a total of 23 convolutional layers.

The realization code for the model can be viewed in Appendix B realized by Tensorflow.

MobilenetV2 Model

The 2^{nd} model used is called MobilenetV2[25],whose structure is illustrated as Fig. 5.4.Meanwhile,MobileNetV2's versatility can support medical professionals and patients by enabling instant wound segmentation and wound area calculation using mobile devices such as smart-phones and tablets shortly after the picture is taken.

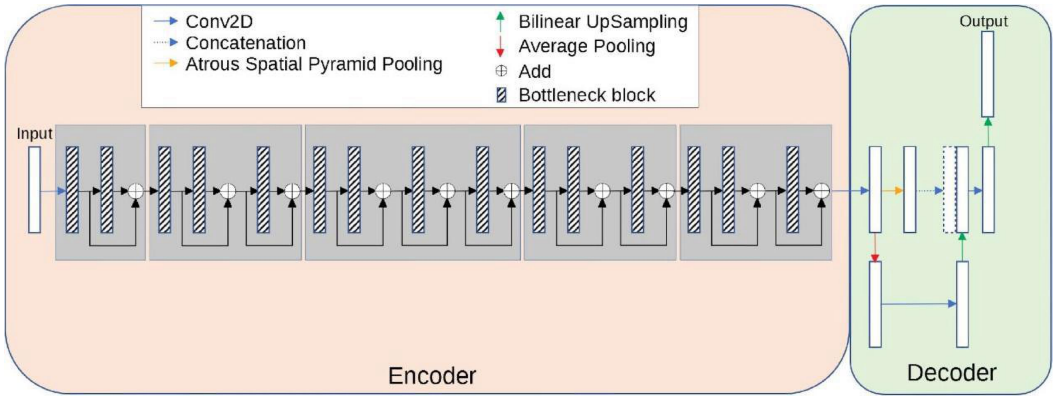


Figure 5.4: The structure of MobilenetV2 model.

The architecture of MobileNetV2 contains the initial fully convolution layer with 32 filters, followed by 19 residual bottleneck layers.The ReLU6 is used as the non-linearity because of its robustness when used with low-precision computation [26]. The kernel size 3 × 3 is used as standard for modern networks, and utilize dropout and batch normalization during training.

SegNet Model

The 3^{rd} model used is called SegNet[27],and the structure of SegNet is shown as the figure(Fig. 5.5) below.SegNet is a fully convolutional network for pixel-level image segmentation. The core component of segmentation is an encoder network and its corresponding decoder network, followed by a pixel-level classification network.

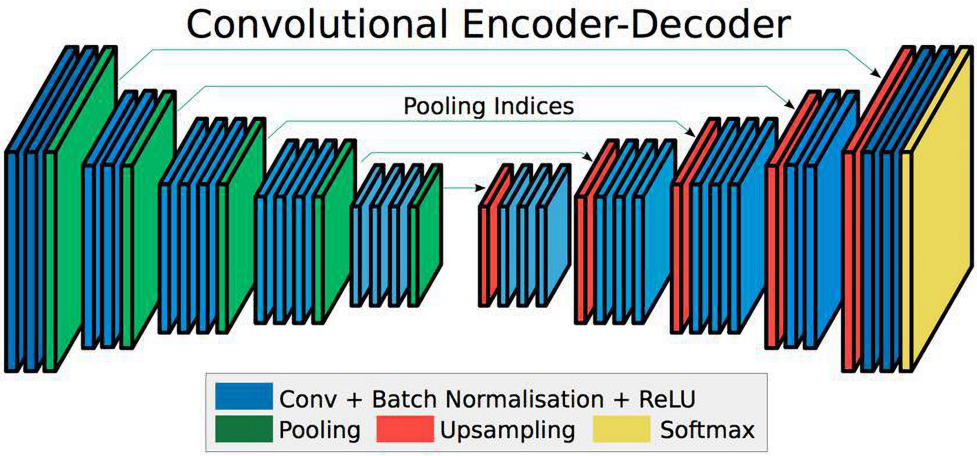


Figure 5.5: An illustration of the SegNet architecture. There are no fully connected layers and hence it is only convolutional. A decoder upsamples its input using the transferred pool indices from its encoder to produce a sparse feature map(s). It then performs convolution with a trainable fifilter bank to densify the feature map. The fifinal decoder output feature maps are fed to a soft-max classififier for pixel-wise classifification[26].

The structure of SegNet can be summarized as two parts. **Encoder**: (a)At the encoder, convolution and max pooling are performed. (b)VGG-16 has 13 convolutional layers. (Without a fully connected layer). (c)When performing 2×2 maximum pooling, store the corresponding maximum pooling index (position). **Decoder**: (a)At the decoder, upsampling and convolution are performed. Finally, each pixel is sent to the softmax classifier. (b)During upsampling, as shown above, the maximum pooling index at the corresponding encoder layer is called for upsampling. (c)Finally, a K-type softmax classifier is used to predict the category of each pixel.

5.2.3 Evaluation Metrics

Except for loss,precision, recall and the dice coefficient as assessment methods are taken to test the segmentation performance [28]. And the basic definition of confusion matrix[3] is demonstrated as the Table 5.1:

Table 5.1: Wound/Non-Wound Pixel Dichotomy

Sample	Positive	Negative
Predict Correctly(True)	TP	TN
Predict Wrongly(False)	FP	FN

Meanwhile, the specific meaning in confusion matrix is illustrated in Table 5.2[4] below:

Table 5.2: The Specific Definition of Elements in Confusion Matrix

Elements	Specific Meaning
TP	Predict the positive class as a positive class
FN	Predict the positive class as a negative class
FP	Predict the negative class as a positive class
TN	Predict the negative class as a negative class

Based on the Table 5.1 and Table 5.2, the customer matrix' definition is listed as follow, which contains the Precision,Recall, Dice Coefficient :

Precision

Precision calculates the right segmenting pixel percentage in the segment and the accuracy of segmentation is demonstrated by precision. In particular, it is calculated using Eq. 5.7:

$$P = \frac{TP}{TP + FP} \tag{5.7}$$

[3]Usually the class of interest is the positive class, and the other classes are the negative class.

[4]TP represents that the pixel that predicts the wound is actually marked as the wound.

Recall

Recall also shows the accuracy of segmentation. More specifically, it measures the percentage of correctly segmented pixels in the ground truth(ground truth is the correct label) and is computed by Eq. 5.8:

$$R = \frac{TP}{TP + FN} \tag{5.8}$$

Dice Coefficient

Dice Coefficient (Dice) shows the correlation between segmentation and ground truth. As a calculation combining Accuracy and Memory, Dice is also called F1 score. More precisely, the harmonic mean of Precision and Recall is determined by Dice [5] as Eq. 5.9 :

$$Dice = \frac{2 \times TP}{2 \times TP + FP + FN} \tag{5.9}$$

5.2.4 Experimental Setup

The deep learning model in the presented work was implemented in Python with Keras[29] and Tensorflow[30] back-end, as stated in Section 3.3.2. The Adam optimization algorithm [31] was used to update the network's parameters, which has become common in the field of stochastic optimization due to its rapid convergence compared to other optimization functions. The loss function was binary cross entropy, and the training phase also tracked Precision, Recall, and the Dice score defined in Section 5.2.3 as evaluation matrices. For balancing training precision and performance, the initial learning rate was set to 0.0001 and each mini-batch contained only two photographs.

[5] It is worth noting that in the actual code implementation, a smooth is often added to prevent the denominator from being 0.

In this experiment, the size of the convolution kernel is set to 32. Meanwhile,The early stopping to terminate the training so that the best result was saved when there was no improvement for more than 100 epochs in terms of Dice score. Eventually, the deep learning models were trained for around 1000 epochs for each before over-fitting.

5.2.5 Post-processing & Posterior Test

In order to reduce the noise in the segmentation results, this project intends to use post-processing to optimize the segmentation results. Here, connected component labeling(CCL) algorithm is used for screening to increase the true positive rate(refer to Table 5.1). Smaller false positive noises are also eliminated in the same way. By utilizing this way, noises were simply removed in the segmentation results by removing the connected component small enough based on adaptive thresholds. To be more specific, a connected region is removed when the number of pixels within the region is less than a threshold, which is adaptively calculated based on the total number of pixels segmented as wound pixels in the image.

After that, the post-processing segmentation results are compared with the annotations of the training set to measure the similarity between the segmentation results and the ground truth region, that is, the intersection and union of the "predicted bounding box" and the "real bounding box" ratio. That is to calculate the IoU[32]. The calculation formula is Eq. 5.10:

$$IoU = \frac{TP}{FP + TP + FN} \tag{5.10}$$

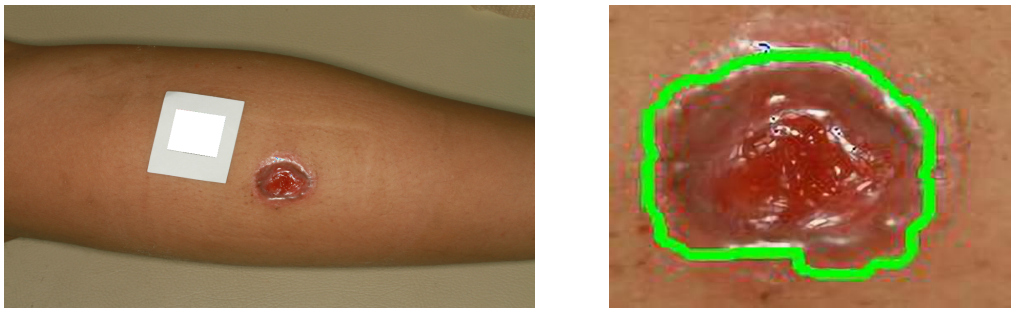
Chapter 6

Result & Discussion

6.1 Results Obtained by Utilizing OpenCV

6.1.1 Segmentation Result

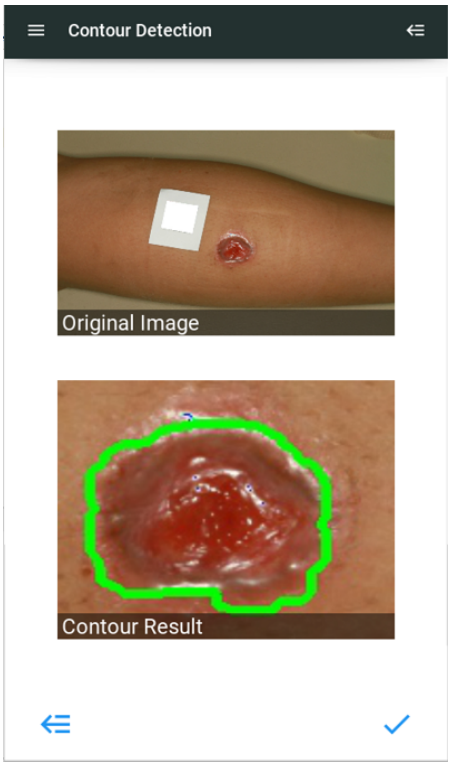
After using method mentioned in Section 5.1.1, the contour of the wound image can be achieved, the figure(Fig. 6.1) illustrates the original input image and corresponding segmentation result.



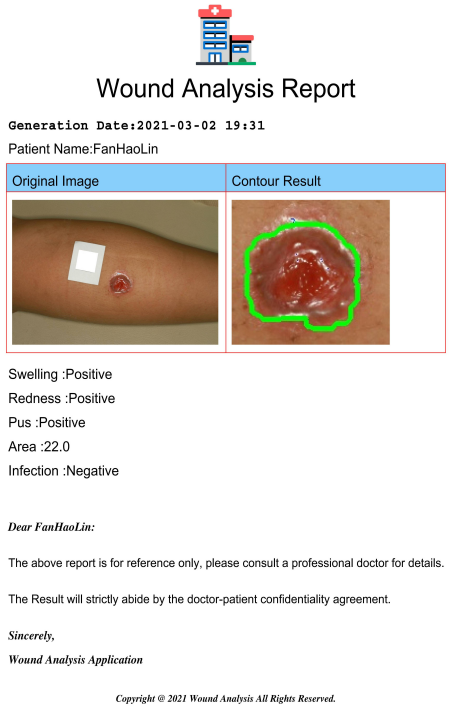
(a) A demo image uploaded by the user **(b)** The segmentation result

Figure 6.1: The original wound image and corresponding segmentation result achieved by OpenCV method, the green closure curve in the right chart outlines the wound region.

And the result in app is illustrated in Fig. 6.2(a), which is the actual interface during the use of the App.Meanwhile, according to the medical report generation function mentioned in Section 4.1.4, the result of wound analysis report is shown as Fig. 6.2(b).



(a) The interface of Contour Detection result.



(b) The wound analysis report.

Figure 6.2: The realization of wound contour locating during actual use of App and corresponding wound analysis report, the report can be sent via e-mail to patient's mailbox.

6.1.2 Feature Analysis Result

As the method introduced in Section 5.1.2, the HSV variance result of the demo image(Fig. 6.1(b)) is demonstrated as Fig. 6.3.

The analytical characteristics of the wound includes the swelling, redness, pus, infection and area.At present, the area is in pixel form.The relevent feature of the demo image is listed as Table 6.1 below:

6.1.3 Wound Analysis History Traversal

As it mentioned in Section 4.1.4, the characteristic effects of the analytical result above have been stored. At the same time, the app is equipped with wound analysis history traversal function: When the user request for the history search,and during the traversal process, if the input patient's name

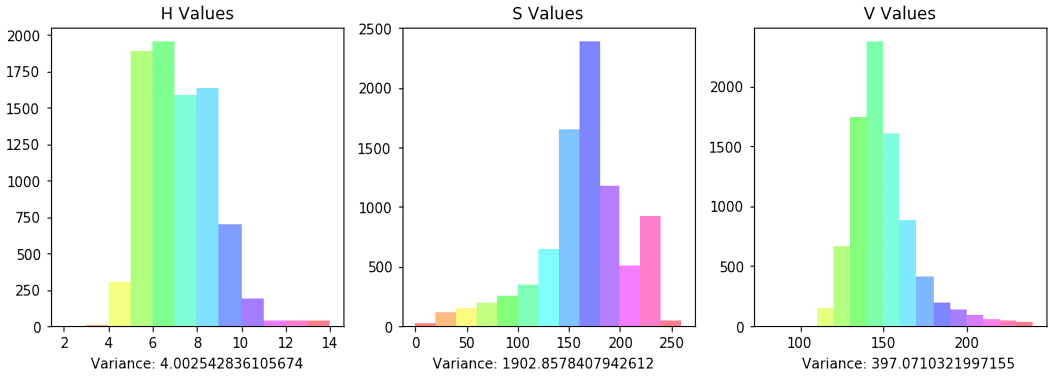


Figure 6.3: The figure illustrates the variance of the wound region in for each channel in HSV three-channel-diagram.

Table 6.1: The analytical feature result achieved by the HSV method

Wound Characteristic Type	Corresponding Reult
Swelling	Positive
Redness	Positive
Pus	Positive
Infection	Negative
Area	22

exists,the swipe-up item will contain the corresponding information of the patient. The Fig. 6.4 below illustrates the example usage of history traversal.

6.2 Results Obtained by Utilizing DL

This section summarizes the results of image segmentation based on deep learning, and elaborates on the model's training parameter results and image segmentation results. On this basis, the performance measures of different models in this project are compared, and then the advantages, disadvantages and feasibility of different models for mobile application scenarios are analyzed.

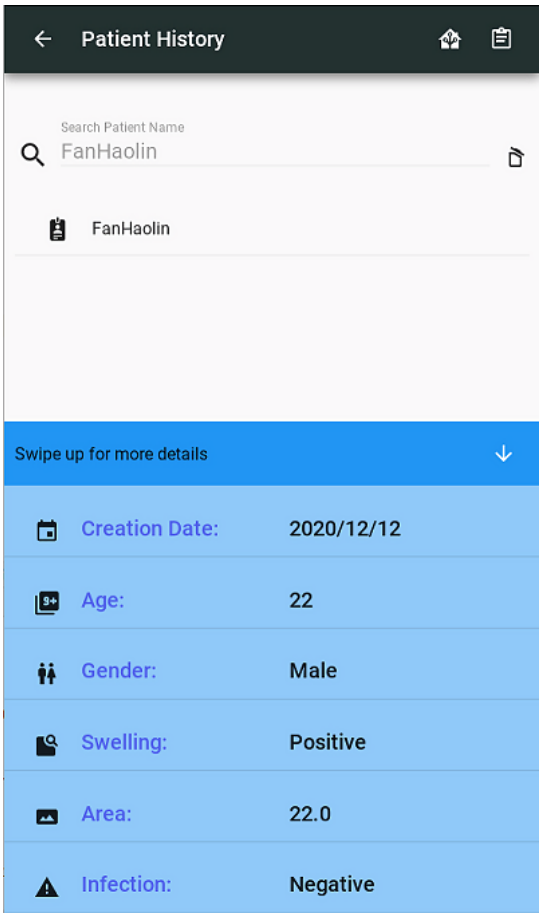


Figure 6.4: Patient history traversal.

6.2.1 Models' Training Parameters

Before the model performance is obtained, the training parameters of the model are summarized. The following table(Table 6.2) demonstrates the trainable parameters and non-trainable parameters among the models during the model training.

Table 6.2: Summary of Total Trainable Parameters

Model Name	U-Net	MobileNetV2	SegNet
Trainable Parameters	4,834,839	2,108,417	339,905
Non-trainable Parameters	0	33,088	0

6.2.2 Training Results

The **precision**, **recall** and **dice score** results assessed by different models will be given in this section. At the same time, the result of wound image segmentation predicted by training model will be presented intuitively by contrast graph.

Model Performance

The table(Table 6.3) below shows the quantitative outcomes of the different networks, with bold numbers indicating the best results of all the models.

Table 6.3: The train loss, precision, recall, and dice score evaluated using various models

Model Name	U-Net(%)	MobileNetV2(%)	SegNet(%)
Train Loss	**0.0017**	0.0057	0.0025
Precision	**97.87**	93.65	96.26
Recall	**98.45**	94.26	97.37
Dice	**98.15**	93.91	96.79

Segmentation , Post-processing & Posterior Results

The following contrast graph[6](Fig. 6.5) intuitively shows the original wound picture, the binary mask results predicted by different models, and the post-processing results, thus intuitively reflecting the performance of the models.In quantitative performance after utilizing

Meanwhile, as it mentioned in 5.2.5, the posterior results of the models are listed to compare the performance. Table 6.4 below illustrates the different IoU values achieved by U-Net, MobileNetV2, and SegNet.

[6]Only the segmentation results of the U-Net model with the best training results are shown here. For segmentation results of other models, please refer to Appendix C

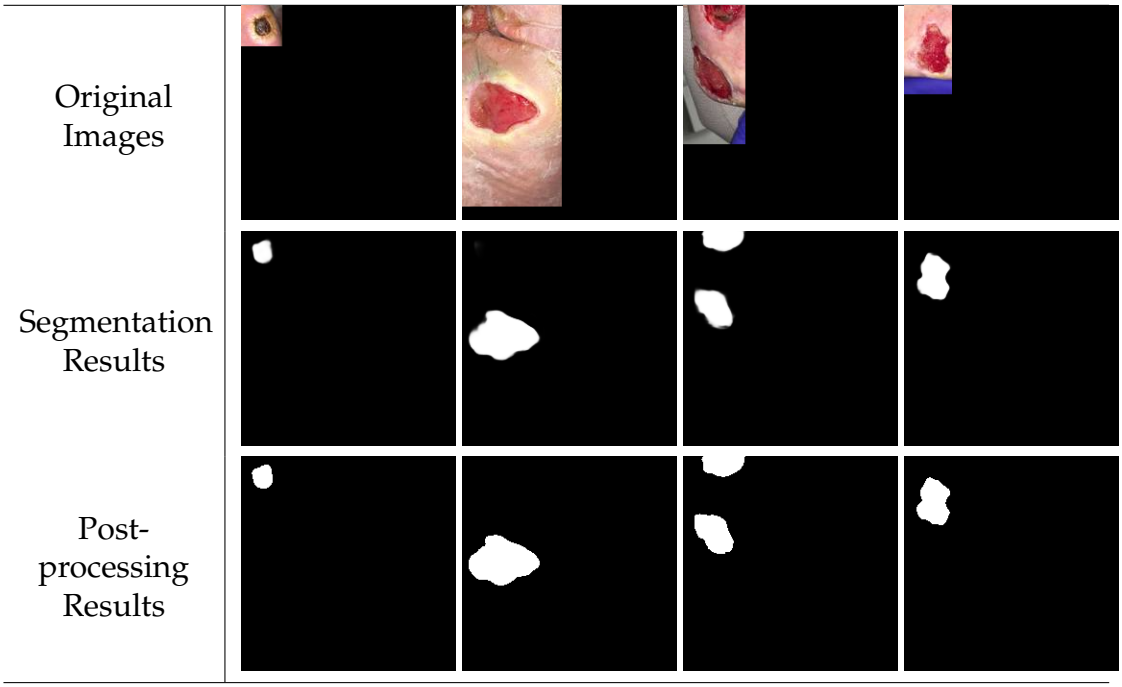


Figure 6.5: Above are the results of wound image segmentation based on U-Net. the 1^{st} row are the original wound images in the test set, the 2^{nd} row are the results of segmentation using the U-Net, and the 3^{rd} row are the outcomes of denoising the segmentation result.

Table 6.4: The posterior test evaluated via IoU using various models after post-processing

Model Name	U-Net(%)	MobileNetV2(%)	SegNet(%)
IoU	**73.31**	72.04	60.12

6.2.3 Discussion

Through the analysis and summary of the above quantitative results and visualization results, it is concluded that in the method of wound image segmentation based on deep learning in this project:

U-Net Achieves Highest Quantitative Performance. From the results, it is obvious that U-Net has smallest train loss, highest Precision, Recall, and Dice. This result fully proves the good performance of U-Net in medical image segmentation and its important role in medical image interpretability.

U-Net Comparison with SegNet. U-Net is mainly used for biomedical image segmentation. The U-Net comparison with SegNet shows a somewhat higher average Dice score for the former model. More specifically, in the performance measures, the Recall of Segnet was evaluated to be the second highest among all models, at 97.37%. This was 1.08% behind the highest Recall, 98.45%, which was achieved by U-Net. In order to replace pooling indices operation in SegNet's architecture, U-Net presents skip connections between convolutional layers. These skip connections concatenate the output of the transposed convolution layers with the feature maps from the encoder at the same level. Thus, the expansion section which consists of a large number of feature channels allows the network to propagate localization combined with contextual information from the contraction section to higher resolution layers. Intuitively, in the expansion section or "decoder" of the U-Net architecture, the segmentation results are reconstructed with the structural features that are learned in the contraction section or the "decoder"[33]. This allows the U-Net to make predictions at more precise locations. These comparisons have illustrated the effectiveness of skip connections for improving the accuracy of wound segmentation.

Comparison of Efficiency and Lightweight. Aside from performance, efficiency and lightweight features are also considered. The overall number of trainable parameters in the adopted MobileNetV2 was just a fraction of the numbers in U-Net, as seen in Table 6.3. As a result, the network needed less time to train and could be used on mobile devices with reduced capacity and computing resources. In contrast to the other versions, higher-resolution

input images could be fed into MobileNetV2 with less memory and processing capacity.

Comparison of IOU Value. For the post-processing process, the IOU value was calculated to evaluate the posterior results. Among three models, U-Net achieved the highest IOU value, which was 1.27% higher than the IOU value for second place. This means that U-Net has relatively excellent overall performance in the wound image segmentation of this project.

Chapter 7

Project Iteration:Web-App Implementation

In the process of developing the App, because the interactivity of image processing in Kivy is not particularly ideal, the operation is more cumbersome when selecting ROI and other operations, and when using deep learning for image segmentation, packaging deep learning dependencies such as Tensorflow needs to take up larger space.

Therefore, in order to realize the core content of this project-the segmentation and analysis of wound images, and the calculation of actual wound area, on the basis of App development, consider using the Web-App to implement and supplement functions.The implementation of this project iteration is based on the Streamlit[34] in Python. At present, the web-app has been successfully deployed, and readers can visit the web site to experience.

7.1 Wound Image Analysis Using OpenCV

The realization idea of this method is similar to that of Section 5.1. It mainly uses the image processing function in OpenCV to segment and analyze the uploaded image. The main implementation steps are: 1.Gaussian Blur and Edge detection using Canny Edge. 2.Morphological operations. 3.Contour locating. In the actual use process, after the user uploads the image, the specific operation can be summarized into the following 4 steps.

7.1.1 Calibrate Image

The calibration of the image can be realized by dragging the corners of

the marker over two points of known distance in the horizontal axis and enter the distance. The purpose of this step is to correspond the pixel size to the actual size of the image, which means the resolution of the image can be gained, so that the pixel area can be converted into an actual area. The Fig. 7.1 below demonstrates the realization of calibrating image.

Figure 7.1: The process of Calibrating Image .

7.1.2 Select ROI

After the resolution of the image is achieved, the next step is locating the contour of the wound image. There exists an important concept in image process which is ROI. The core operation is adjusting the bounding box realized by *streamlit_cropper* to select the The approximate region of the wound by inspection. Theoretically, the more accurate the selection of ROI, the better the segmentation effect accordingly. Fig. 7.2 below illustrates the realization of the ROI selecting.

7.1.3 Threshold Adjusting & Result Display

On the basic of the selecting ROI, the step is focus on adjusting the Threshold and Width sliders to get the ideal segmentation. The principle is using *cv2.threshold* as well.In this step, the slider component on the web

Figure 7.2: The outcome of selecting ROI .

side is used and the parameters are called back to realize the dynamic adjustment of the threshold. Fig. 7.3 below illustrate the threshold adjusting process and the final segmentation result as well(In this example, the ideal threshold is around to 52).

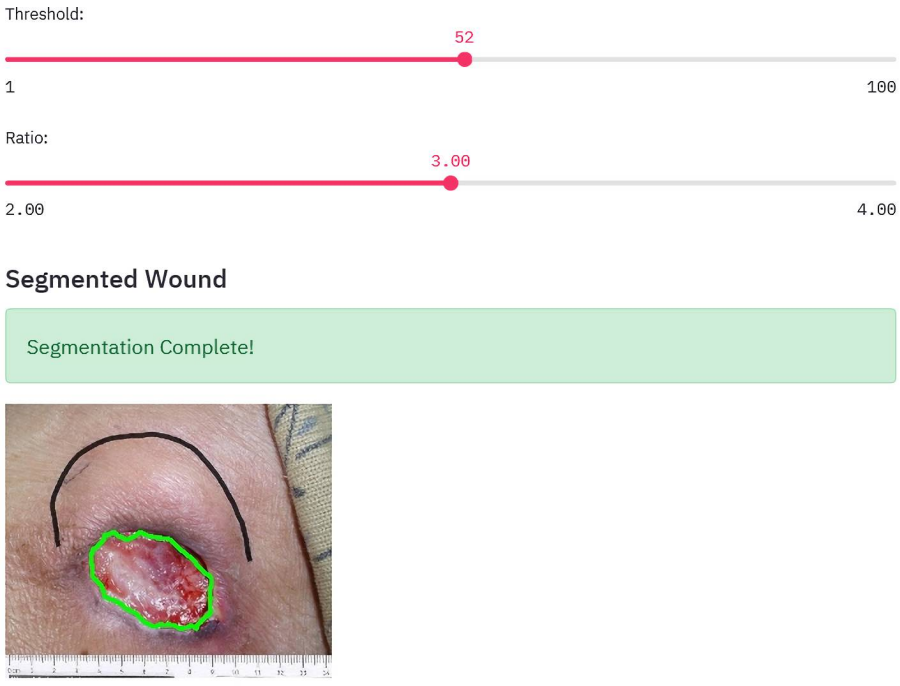


Figure 7.3: The outcome of wound contour locating by adjusting threshold .

7.1.4 Area Calculation & Feature Identifying

As the Fig. 7.3 shows, it can be inspected that the segmentation is ideal,In

order to calculate the area of the wound area and obtain the area of consumables required for 3D printing accordingly, the relationship between the areas of the two parts here is: $Area_{3D} = 1.5 \times Area_{woundregion}$.

Meanwhile,the feature of the wound is gained by K- means clustering algorithm[35] realized by *cv2.kmeans*, where k equals to 4 in the analytical process, whose algorithm is briefly summarized as follows :

Given a set of observations $(x_1, x_2, ..., x_n)$, where each observation is a d-dimensional real vector, k-means clustering aims to partition the n observations into $k(\leq n)$ sets $S = \{S_1, S_2, ..., S_k\}$, so as to minimize the within-cluster sum of squares (WCSS) (i.e. variance). Formally, the objective is to find, where μ_i is the mean of points in S_i:

$$\arg\min_{S} \sum_{i=1}^{k} \sum_{x \in S_i} \|x - \mu_i\|^2 = \arg\min_{S} \sum_{i=1}^{k} |S_i| \operatorname{Var} S_i \tag{7.1}$$

By utilizing the K-means clustering algorithm, the feature of wound like Granulation Tissue[36] and Slough could to gained .The calculation and analysis results of this step are shown as Fig. 7.4. The user can clearly and intuitively see the area size of the auxiliary materials required for the wound and the characteristics of the wound. Compare to the wound analysis function in former app. Here, only two characteristics are analyzed.

7.2 Wound Image Analysis Using DL

Similar to the method and model mentioned in Section 5.2, Web-App also uses different model training results to segment the input wound image, but at this time, it is easier to select and call the model compared to the App side. At the same time, the site-packages like Tensorflow, Kears can be deployed to the cloud server as well the corresponding environment and dependencies, thereby reducing the computational pressure of the local processor. Web-App is temporarily unable to realize the database storage of

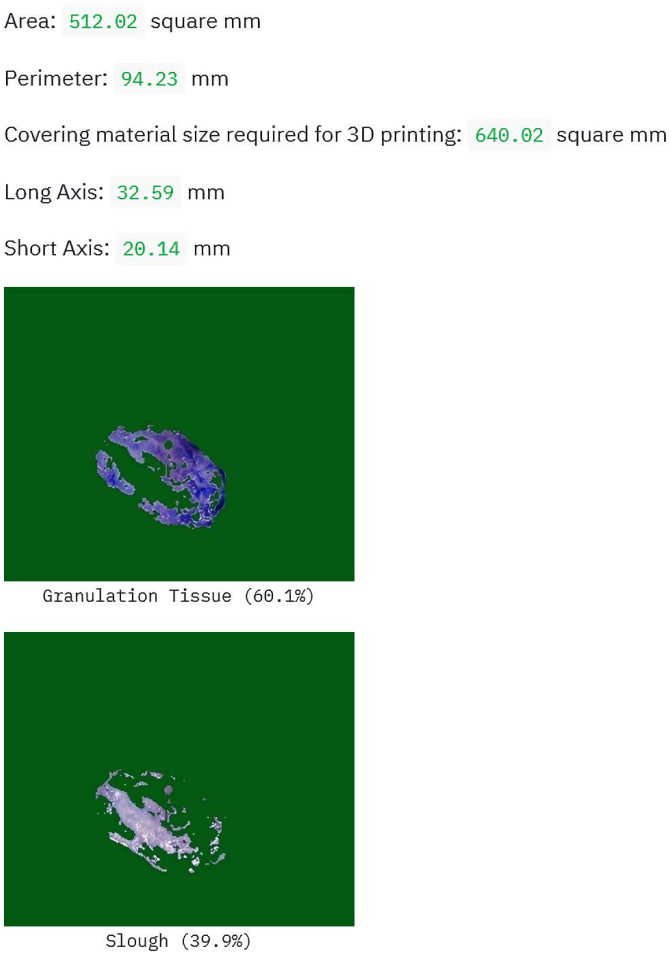


Figure 7.4: The wound image analysis result, which illustrates information about dimensions and corresponding issues.

wound segmentation and analysis results, but those information and file can be stored manually by the user.

7.2.1 Model Selection

First, upload the wound image[7], which can be achieved by code below:

```
1 import numpy as np
2 import cv2
```

[7]Here is a brief description of the upload of the image on the web side. The upload of web image files is uploaded in bytes and decoded accordingly, where the form of data flow is quite different from that of App

```
file_bytes = np.asarray(bytearray(bytes_data), dtype=np.uint8)
input_image = cv2.imdecode(file_bytes, 1)
cv2.imwrite("segmentation/test/images/1.png", input_image)
```

Next, use the select-box on the Web-App side to select the model for segmentation tasks, and the back-end loads the corresponding model for the image segmentation.At present, the available models are U-Net, MobileNetV2, SegNet. Fig. 7.5 shows the process of uploading image by dragging or dropping files in certain formats(.jpg, .png, .jpeg) and function of the model selection:

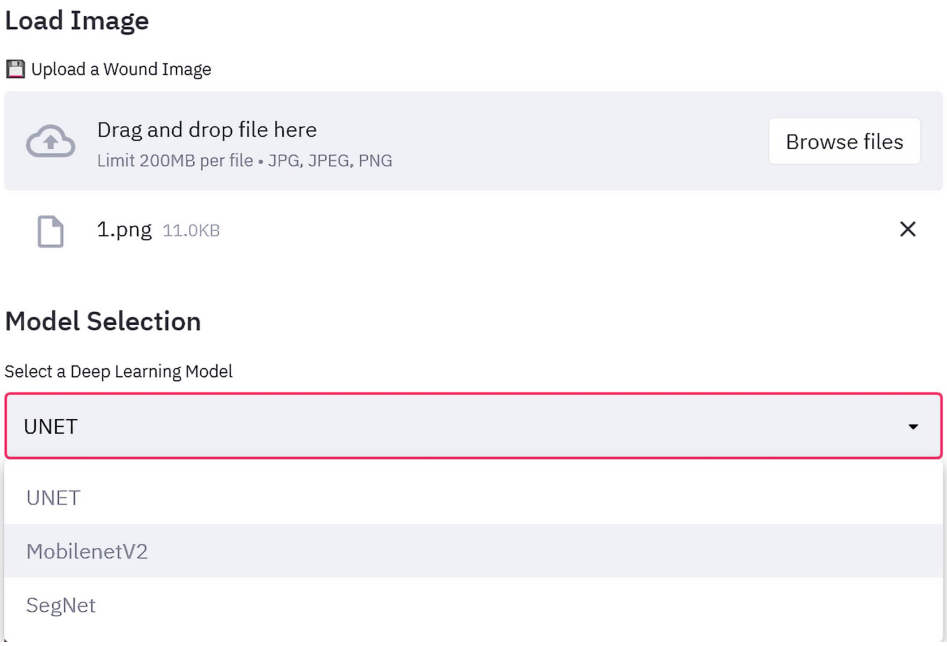


Figure 7.5: The model selection function in Web-App,as shown in figure,the alternative models are U-NET, MobileNetV2, and SegNet .

7.2.2 Wound Image Segmentation

In terms of the page layout, the image uploaded by the user will be displayed first. By clicking the 'predict' button, the corresponding response event is obtained, that is, the image is segmented using the corresponding model, and the predicted result will be displayed below the original image, both original image and segmented result image can be zoomed in and saved

as a local picture. Furthermore, as with the post-processing method mentioned at Section 5.2.5 above, by clicking the 'Denoise Prediction' button, the segmented image is denoised on the Web to improve the overall performance of the segmentation. The figure below(Fig. 7.6) is an example of prediction and post-process:

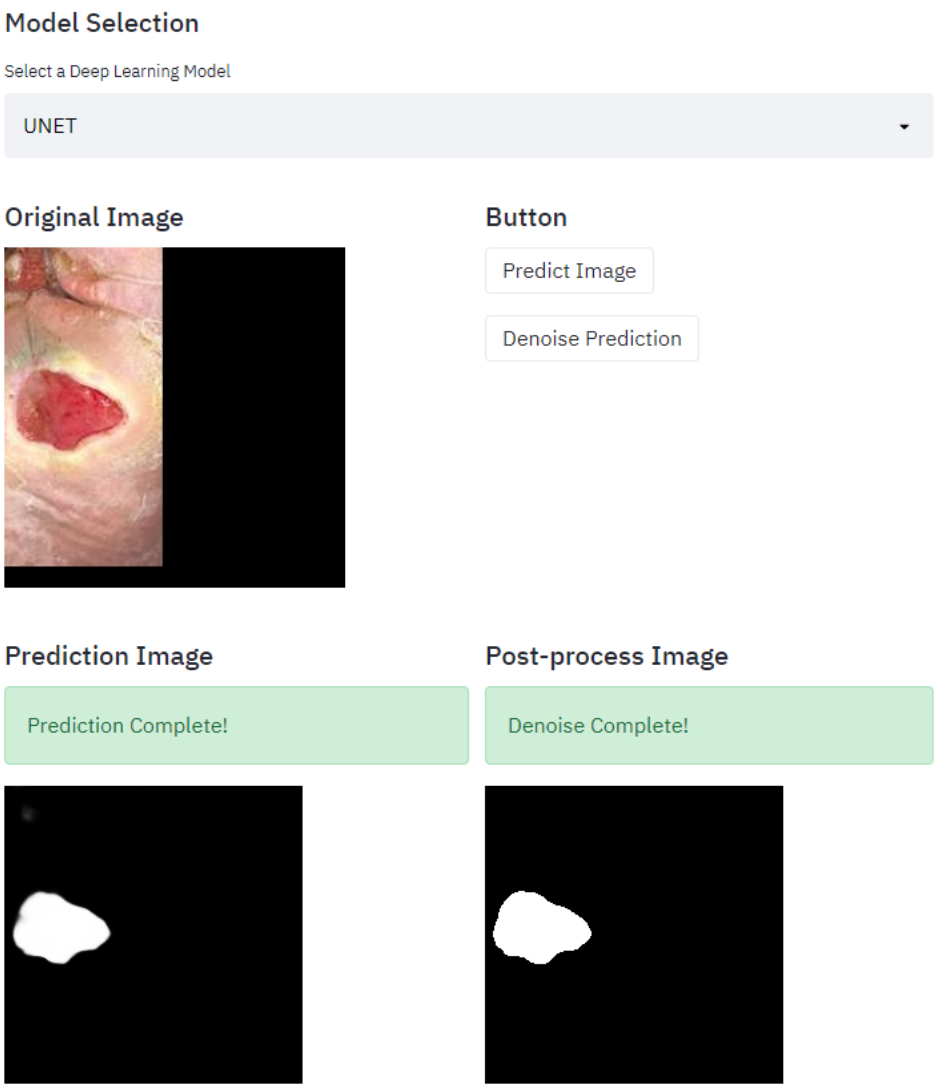


Figure 7.6: The Segmentation Result achieved by utilizing UNET model.

Chapter 8

Summary & Conclusion

8.1 Achievement & Summary of Project Objectives

The content introduced in the above chapters relies on the mobile app and Web-App two types of mobile carriers, using traditional computer vision and deep learning technology to segment and analyze wound images, and explore the possibility of medical image segmentation based on mobile. It provides a reference idea for the combination of auxiliary diagnosis and treatment technology and mobile terminal. The content of the above chapters can be summarized into four parts: App hardware and software specifications and design ideas, UI interface design and implementation, back-end image segmentation and analysis function implementation, Web App-based project iteration. Among them, the third part covers the realization method based on computer vision and the realization method based on deep learning. Especially, the core component(wound image analysis system) of the App can be concluded as Fig. 8.1 below.

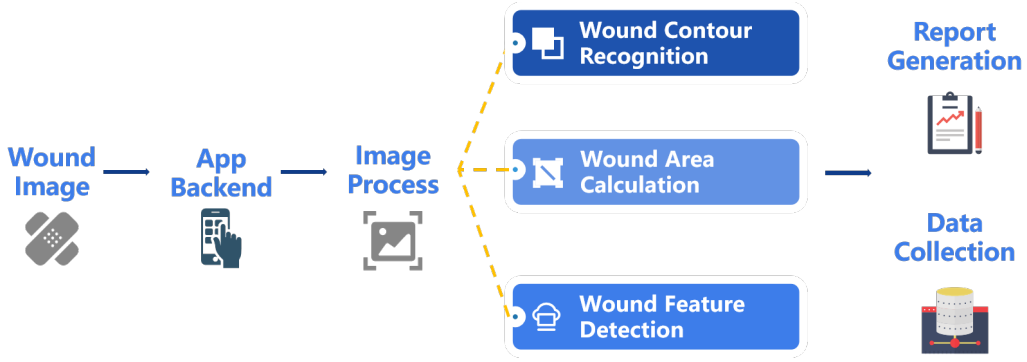


Figure 8.1: An overall flow chart to illustrate the realization process of App.

After practice and development, the summary of this project is as follows: In order to analyze the wound images, identify the forms of wound

tissue, measure wound measurements and track improvements in the injury over time computational methods and related applications are in high demand. When performed manually, this procedure takes a long time and is subject to intra- and inter-rater fluctuations. As a result, developing an intelligent method for wound assessment, diagnosis, and prognosis will save a lot of time for clinicians, lower costs, and improve patient quality of life. Relying on artificial intelligence, there is hope for efficient auxiliary diagnosis and treatment, such as quickly locating the wound area, calculating the area area, and analyzing the characteristics of the wound. Combining artificial intelligence technology with mobile applications is feasible and will be an important development trend of the current era. More generally, in actual development, different mobile application carriers need to be selected according to different needs and target products.

8.2 Contrast & Reflection

On this basis, this chapter compares the different frameworks, platforms and technical principles used to lay the foundation for the optimization of this project. The specific comparison contents are as follows:

8.2.1 Comparison of Kivy & Native Android Development

In terms of App development, this project chose Pyhton as the development language. After Kivy's App development practice, it was found that compared with the Java language required by the native development tool Android Studio, Python has a lower threshold to use. However, if it is oriented to enterprise-level or needs to achieve more interactive application development, Kivy still has many drawbacks.

Compared with native development tools and languages (taking Android system and Java as examples), there are gaps in areas like calling API interfaces(Use of cloud databases or login authentication) and components usage. For example, an interactive selection box is difficult to implement in Kivy, but it is easier in Android Studio. Therefore, if you want to avoid unexpected problems, Java language is more friendly to App development.

Figure 8.2: An interesting image which shows the relationship between android and Kivy.

8.2.2 Comparison of DL & OpenCV in Wound Image Segmentation Tasks

For image segmentation tasks, Opencv comes with a variety of edge recognition and contour extraction functions. If you select an appropriate threshold and perform morphological operations, you can quickly locate the wound region, but the difficulty lies in the selection of the threshold, the application of edge recognition algorithms and the picture its own noise reduction processing, while the recognition effect also depends on the quality of the input image.

In contrast, deep learning extracts wound pixel features. Although it takes a lot of time to acquire, label, and train data sets to obtain a model with higher accuracy, when the process is completed,this model can segment multiple images. The difficulty of its realization lies in the accuracy requirements for the labeling of medical images, and the establishment of the data set requires a lot of time. If it involves the classification of wound characteristics, professional labeling information is also required.

8.2.3 Comparison of App & Web-App

In several ways, the following table(Table 8.1) shows the benefits and drawbacks of the app and web-app, which offers suggestions for the choice of mobile carriers type for the segmentation and study of medical images.

Specifically, mobile carriers can be selected according to different requirements, or the development of platform-wide mobile applications can be considered.

Table 8.1: Comparison of App and Web-App

	App	Web-App
Development Language & Tool	Python & Kivy	Python & Streamlit
Access	Developers pack and publish, and users use them by downloading and installing local apk.	Can be used when connected to the Internet.
Configuration of DL Environment	In most cases, the environment needs to be configured locally, and if installed in the cloud, it is more complex.	Can be arranged in web cloud server.
Functional Diversity	Its advantage lies in comprehensive development.	Streamlit has advantages in research-oriented (such as data analysis, machine learning, deep learning, etc.) Web-side app development.
Interface Interaction	Kivy have drawbacks in interactive components, such as slider.	The interaction function is perfect, and the callback response of interaction time is convenient.
Data Security	Local or cloud storage, not easy to lose.	There may be errors in network transmission.

Chapter 9

Future Recommendations

In the process of realizing the project, there are still some problems. This chapter briefly discusses the existing problems of the project to provide ideas for the improvement and optimization of the project:

1. In terms of the implemented platform and framework, Kivy still has a gap in App development compared to Android Studio, and the function implementation is more complicated than that of web-app.
2. When using deep learning for image segmentation, the segmentation of the wound image depends on the accuracy of the training model, and there are relatively strict requirements for the camera environment and angle of the input image. Currently, the problem is that the Precision, Recall, and Dice could reach a satisfactory outcome, however, the posterior results were not ideal.
3. Furthermore, in the traditional computer vision method, the location of the wound contour and the calculation of the area depend on the connected domain, which means that if the type of wound is more complex and scattered, the recognition effect will be unsatisfactory.
4. At present, the treatment of wounds is limited to the segmentation of the wound area and simple wound feature recognition. For complex wound types, the analysis results are not ideal.
5. In order to estimate the actual area of the wound and the area of consumables required for 3D printing, it is necessary to correspond the pixel area to the actual area, so the horizontal or vertical length of the wound area needs to be measured in reality. Therefore, the accuracy of wound size measurement in reality is often very critical, and a suitable method should be adapted.

In summary, regarding the above-mentioned problems, the optimization direction of the project could be:

1. Consider using Java as a development language. To improve the efficiency of wound image segmentation and analysis, and simplify the process, you can directly use the Web App for development.
2. Focus on the filtering of image noise (the input image can be denoised through preprocessing and different filtering algorithms to reduce the impact of the external environment on the photo)
3. Before using the deep learning method, preprocess the images in the training set, consider whether there is the possibility of overfitting the training model, and consider using transfer learning to improve the training efficiency of the model, to ensure the training model based on the good performance of Precision, Recall and other parameters, the comprehensive performance of the model and the posterior results of image segmentation are improved. Meanwhile, consider including more data in the dataset to improve the robustness and prediction accuracy.
4. Consider using image classification in deep learning to predict and distinguish different characteristics of wounds, so as to achieve the purpose of auxiliary diagnosis and treatment.
5. The optimization of the actual area calculation method can consider placing a standard reference object (for example, a 2cm×2cm square) when taking pictures, and then obtain the relationship between the pixel area and the actual area, but the problem of this solution is still similar to the second problem above, that is, it depends on the environment and angle of the photo. For actual auxiliary diagnosis and treatment, this method is also difficult to implement.

Appendix A

Other Functions Covered by App

Appendix A mainly introduces other functions covered by the application that are not introduced in detail in 4.1.6, whose main components are weather forecast(Fig. A.1), chatbot as customer service(Fig. A.2), personal setting(Fig. A.3), and relative information display(Fig. A.4).

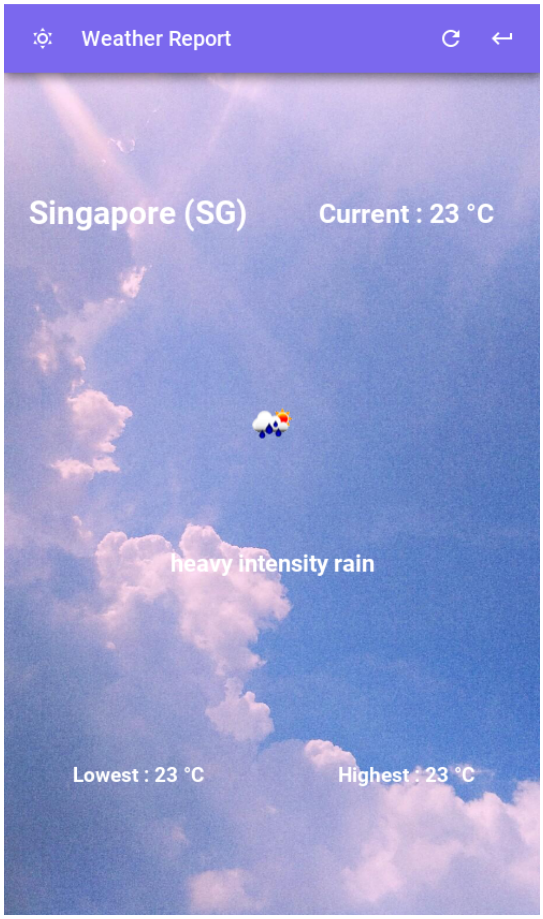


Figure A.1: The UI for weather forecast, it displays the current weather at Singapore.

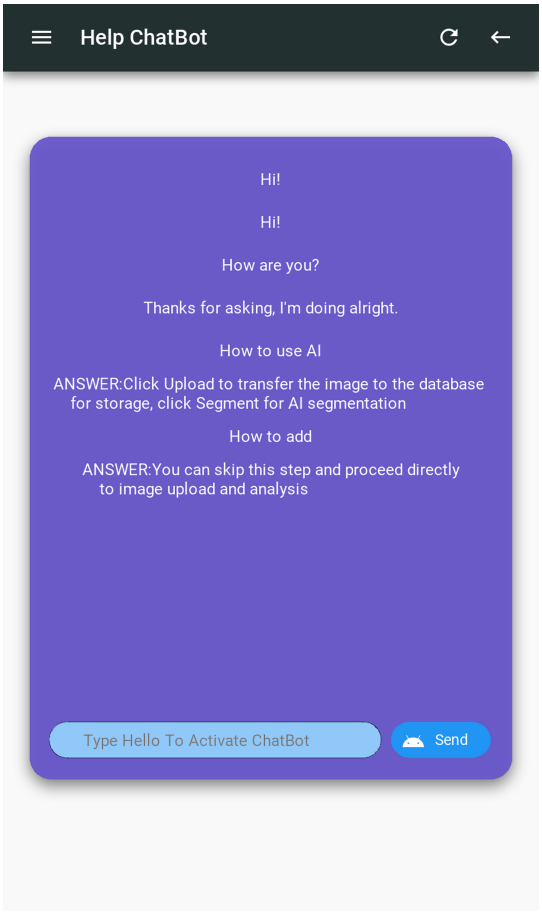


Figure A.2: The UI for help chatbot, and the chatbot is able to answer some basic questions about the usage of the App.

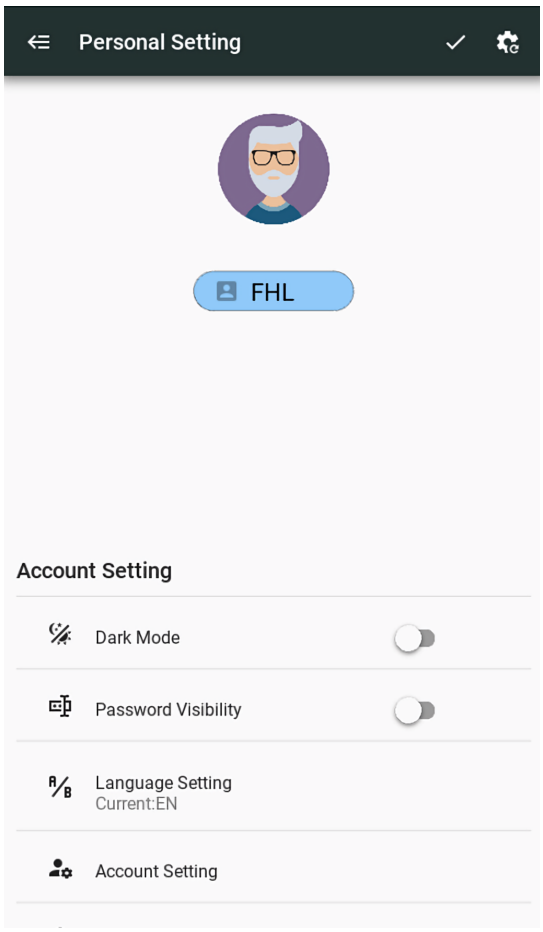


Figure A.3: The UI for personal setting module, which includes mode change, password visibility change, account setting, and password reset for current account .

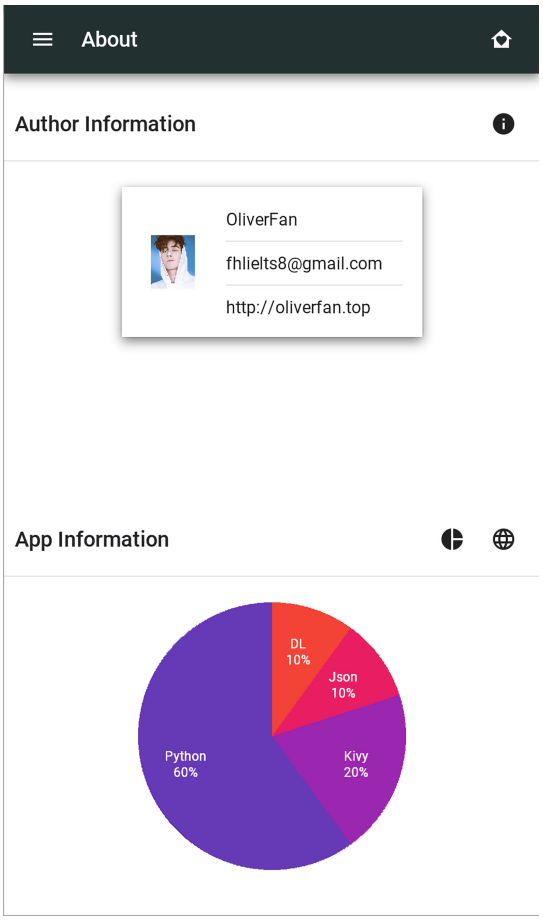


Figure A.4: The about information of the App.

Appendix B

Source Code for Realization of U-Net

The code for realization of U-Net can be summarized as **Up Sampling**, **Down Sampling** and **Softmax**.

```
def unet_model(self):
    unet_input = Input(shape=(self.input_dim_x, self.input_dim_y,
    ↪ self.num_channels))

    conv1 = Conv2D(self.n_filters, kernel_size=3,
    ↪ activation='relu', padding='same')(unet_input)
    conv1 = Conv2D(self.n_filters, kernel_size=3,
    ↪ activation='relu', padding='same')(conv1)
    pool1 = MaxPooling2D(pool_size=(2, 2))(conv1)

    conv2 = Conv2D(self.n_filters * 2, kernel_size=3,
    ↪ activation='relu', padding='same')(pool1)
    conv2 = Conv2D(self.n_filters * 2, kernel_size=3,
    ↪ activation='relu', padding='same')(conv2)
    pool2 = MaxPooling2D(pool_size=(2, 2))(conv2)

    conv3 = Conv2D(self.n_filters * 4, kernel_size=3,
    ↪ activation='relu', padding='same')(pool2)
    conv3 = Conv2D(self.n_filters * 4, kernel_size=3,
    ↪ activation='relu', padding='same')(conv3)
    conv3 = Conv2D(self.n_filters * 4, kernel_size=3,
    ↪ activation='relu', padding='same')(conv3)
    pool3 = MaxPooling2D(pool_size=(2, 2))(conv3)

```

```
17      conv4 = Conv2D(self.n_filters * 8, kernel_size=3,
        ↪  activation='relu', padding='same')(pool3)
18      conv4 = Conv2D(self.n_filters * 8, kernel_size=3,
        ↪  activation='relu', padding='same')(conv4)
19      conv4 = Conv2D(self.n_filters * 8, kernel_size=3,
        ↪  activation='relu', padding='same')(conv4)
20      pool4 = MaxPooling2D(pool_size=(2, 2))(conv4)
21
22      conv5 = Conv2D(self.n_filters * 8, kernel_size=3,
        ↪  activation='relu', padding='same')(pool4)
23      conv5 = Conv2D(self.n_filters * 8, kernel_size=3,
        ↪  activation='relu', padding='same')(conv5)
24      conv5 = Conv2D(self.n_filters * 8, kernel_size=3,
        ↪  activation='relu', padding='same')(conv5)
25
26      up6 = Conv2D(self.n_filters * 4, 2, activation='relu',
        ↪  padding='same')(UpSampling2D(size=(2, 2))(conv5))
27      feature4 = Conv2D(self.n_filters * 4, kernel_size=3,
        ↪  activation='relu', padding='same')(conv4)
28      concat6 = Concatenate()([feature4, up6])
29      conv6 = Conv2D(self.n_filters * 4, kernel_size=3,
        ↪  activation='relu', padding='same')(concat6)
30      conv6 = Conv2D(self.n_filters * 4, kernel_size=3,
        ↪  activation='relu', padding='same')(conv6)
31
32      up7 = Conv2D(self.n_filters * 2, 2, activation='relu',
        ↪  padding='same')(UpSampling2D(size=(2, 2))(conv6))
33      feature3 = Conv2D(self.n_filters * 2, kernel_size=3,
        ↪  activation='relu', padding='same')(conv3)
34      concat7 = Concatenate()([feature3, up7])
35      conv7 = Conv2D(self.n_filters * 2, kernel_size=3,
        ↪  activation='relu', padding='same')(concat7)
36      conv7 = Conv2D(self.n_filters * 2, kernel_size=3,
        ↪  activation='relu', padding='same')(conv7)
37
```

```
38      up8 = Conv2D(self.n_filters * 1, 2, activation='relu',
        ↪  padding='same')(UpSampling2D(size=(2, 2))(conv7))
39      feature2 = Conv2D(self.n_filters * 1, kernel_size=3,
        ↪  activation='relu', padding='same')(conv2)
40      concat8 = Concatenate()([feature2, up8])
41      conv8 = Conv2D(self.n_filters * 1, kernel_size=3,
        ↪  activation='relu', padding='same')(concat8)
42      conv8 = Conv2D(self.n_filters * 1, kernel_size=3,
        ↪  activation='relu', padding='same')(conv8)
43
44      up9 = Conv2D(int(self.n_filters / 2), 2, activation='relu',
        ↪  padding='same')(UpSampling2D(size=(2, 2))(conv8))
45      feature1 = Conv2D(int(self.n_filters / 2), kernel_size=3,
        ↪  activation='relu', padding='same')(conv1)
46      concat9 = Concatenate()([feature1, up9])
47      conv9 = Conv2D(int(self.n_filters / 2), kernel_size=3,
        ↪  activation='relu', padding='same')(concat9)
48      conv9 = Conv2D(int(self.n_filters / 2), kernel_size=3,
        ↪  activation='relu', padding='same')(conv9)
49      conv9 = Conv2D(3, kernel_size=3, activation='relu',
        ↪  padding='same')(conv9)
50      conv10 = Conv2D(1, kernel_size=1,
        ↪  activation='sigmoid')(conv9)
51
52  return Model(outputs=conv10, inputs=unet_input), 'unet_model'
```

Appendix C

Segmentation Results Achieved by MobileNetV2 & SegNet

The following figures are the prediction results of the training set based on MobileNetV2 and SegNet. According to the prediction result, it can be intuitively seen that the segmentation result is consistent with the quantitative performance of the models.

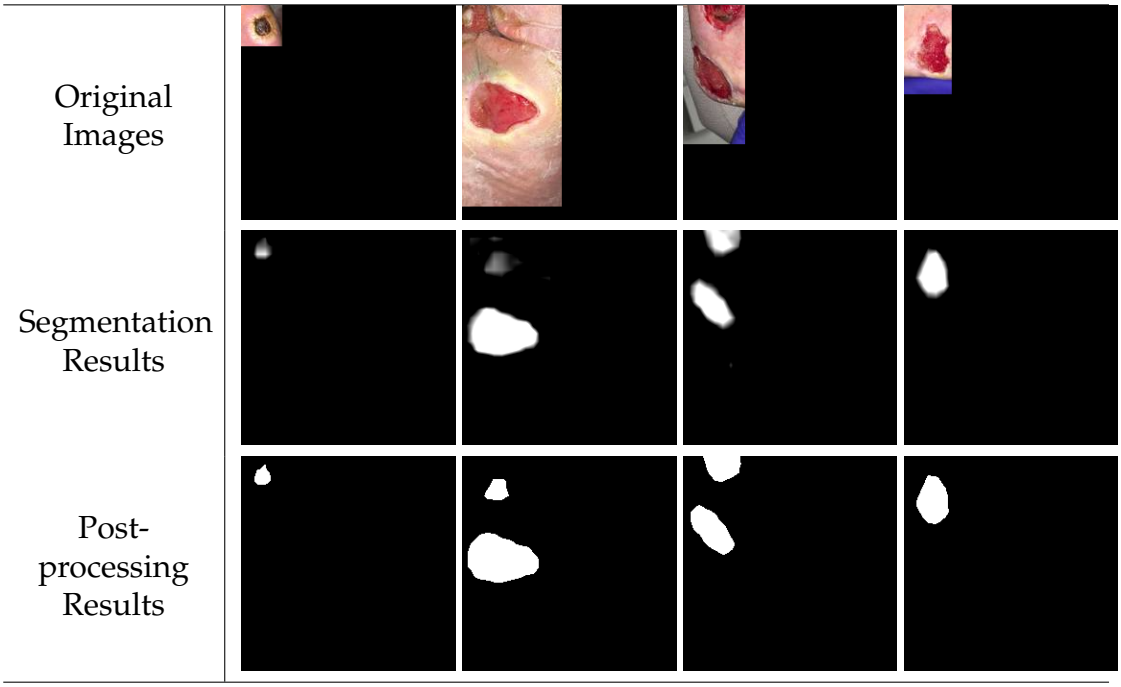


Figure C.1: Above are the results of wound image segmentation based on MobileNetV2. the 1^{st} row are the original wound images in the test set, the 2^{nd} row are the results of segmentation using the U-Net, and the 3^{rd} row are the outcomes of denoising the segmentation result.

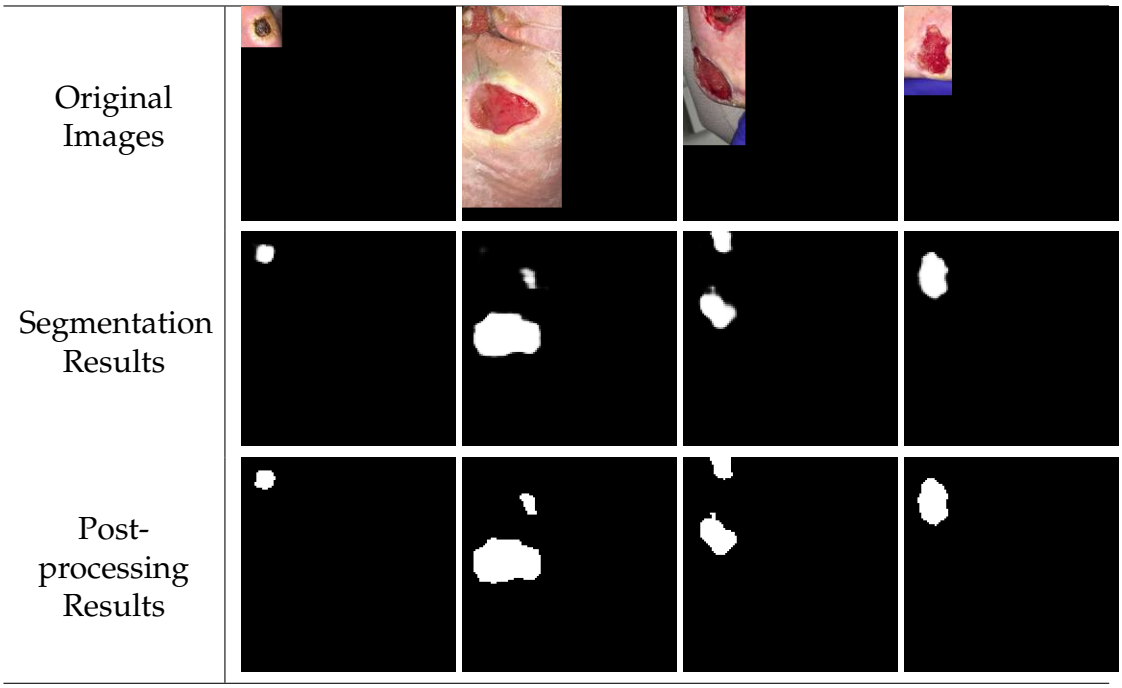


Figure C.2: Above are the results of wound image segmentation based on SegNet. the 1^{st} row are the original wound images in the test set, the 2^{nd} row are the results of segmentation using the U-Net, and the 3^{rd} row are the outcomes of denoising the segmentation result.

Bibliography

[1] L. K. Branski, G. G. Gauglitz, D. N. Herndon, and M. G. Jeschke. "A review of gene and stem cell therapy in cutaneous wound healing". In: *Burns* 35.2 (2009), pp. 171–180.

[2] F. L. Bowling, L. King, J. A. Paterson, J. Hu, B. A. Lipsky, D. R. Matthews, and A. J. Boulton. "Remote assessment of diabetic foot ulcers using a novel wound imaging system". In: *Wound Repair and Regeneration* 19.1 (2011), pp. 25–30.

[3] E. Pasero and C. Castagneri. "Application of an automatic ulcer segmentation algorithm". In: *2017 IEEE 3rd International Forum on Research and Technologies for Society and Industry (RTSI)*. IEEE. 2017, pp. 1–4.

[4] M. Sonka, V. Hlavac, and R. Boyle. *Image processing, analysis, and machine vision*. Cengage Learning, 2014.

[5] L. Deng. *Deep learning: methods and applications. Foundations Trends Signal Process 7 (3–4): 197–387*. 2014.

[6] N. Hettiarachchi, R. Mahindaratne, G. Mendis, H. Nanayakkara, and N. D. Nanayakkara. "Mobile based wound measurement". In: *2013 IEEE Point-of-Care Healthcare Technologies (PHT)*. IEEE. 2013, pp. 298–301.

[7] A. F. M. Hani, L. Arshad, A. S. Malik, A. Jamil, and F. Y. B. Bin. "Haemoglobin distribution in ulcers for healing assessment". In: *2012 4th International Conference on Intelligent and Advanced Systems (ICIAS2012)*. Vol. 1. IEEE. 2012, pp. 362–367.

[8] K. Wantanajittikul, S. Auephanwiriyakul, N. Theera-Umpon, and T. Koanantakool. "Automatic segmentation and degree identification in burn color images". In: *The 4th 2011 Biomedical Engineering International Conference*. IEEE. 2012, pp. 169–173.

[9] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, et al. "Imagenet large scale visual recognition challenge". In: *International journal of computer vision* 115.3 (2015), pp. 211–252.

[10] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner. "Gradient-based learning applied to document recognition". In: *Proceedings of the IEEE* 86.11 (1998), pp. 2278–2324.

[11] C. Wang, Y. Guo, W. Chen, and Z. Yu. "Fully automatic intervertebral disc segmentation using multimodal 3d u-net". In: *2019 IEEE 43rd Annual Computer Software and Applications Conference (COMPSAC)*. Vol. 1. IEEE. 2019, pp. 730–739.

[12] J. Long, E. Shelhamer, and T. Darrell. "Fully convolutional networks for semantic segmentation". In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2015, pp. 3431–3440.

[13] C. Wang, X. Yan, M. Smith, K. Kochhar, M. Rubin, S. M. Warren, J. Wrobel, and H. Lee. "A unified framework for automatic wound segmentation and analysis with deep convolutional neural networks". In: *2015 37th annual international conference of the IEEE engineering in medicine and biology society (EMBC)*. IEEE. 2015, pp. 2415–2418.

[14] X. Liu, C. Wang, F. Li, X. Zhao, E. Zhu, and Y. Peng. "A framework of wound segmentation based on deep convolutional networks". In: *2017 10th International Congress on Image and Signal Processing, BioMedical Engineering and Informatics (CISP-BMEI)*. IEEE. 2017, pp. 1–7.

[15] D. Anisuzzaman, C. Wang, B. Rostami, S. Gopalakrishnan, J. Niezgoda, and Z. Yu. "Image Based Artificial Intelligence in Wound Assessment: A Systematic Review". In: *arXiv preprint arXiv:2009.07141* (2020).

[16] M. H. Yap, C.-C. Ng, K. Chatwin, C. A. Abbott, F. L. Bowling, A. J. Boulton, and N. D. Reeves. "Computer Vision Algorithms in the Detection of Diabetic Foot Ulceration: A New Paradigm for Diabetic Foot Care?" In: *Journal of diabetes science and technology* 10.2 (2016), pp. 612–613.

[17] M. Dujovny. "Pressure Injury Evolution: Mobile Wound Analyzer Review". In: *J Biol Med Res* 2.2 (2018), p. 12.

[18] L. Fraiwan, J. Ninan, and M. Al-Khodari. "Mobile application for ulcer detection". In: *The open biomedical engineering journal* 12 (2018), p. 16.

[19] M. R. Friesen, C. Hamel, and R. D. McLeod. "A mHealth application for chronic wound care: Findings of a user trial". In: *International journal of environmental research and public health* 10.11 (2013), pp. 6199–6214.

[20] J. Canny. "A computational approach to edge detection". In: *IEEE Transactions on pattern analysis and machine intelligence* 6 (1986), pp. 679–698.

[21] N. Hagen and E. L. Dereniak. "Gaussian profile estimation in two dimensions". In: *Applied optics* 47.36 (2008), pp. 6842–6851.

[22] J. R. Dim and T. Takamura. "Alternative approach for satellite cloud classification: edge gradient application". In: *Advances in Meteorology* 2013 (2013).

[23] B. Russell, A. Torralba, K. Murphy, and W. Freeman. "Labelme: a database and web-based tool for image annotation. Int". In: *Journal of Computer Vision* 77 (2007).

[24] O. Ronneberger, P. Fischer, and T. Brox. "U-net: Convolutional networks for biomedical image segmentation". In: *International Conference on Medical image computing and computer-assisted intervention*. Springer. 2015, pp. 234–241.

[25] M. Sandler, A. Howard, M. Zhu, A. Zhmoginov, and L.-C. Chen. "Mobilenetv2: Inverted residuals and linear bottlenecks". In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2018, pp. 4510–4520.

[26] A. G. Howard, M. Zhu, B. Chen, D. Kalenichenko, W. Wang, T. Weyand, M. Andreetto, and H. Adam. "Mobilenets: Efficient convolutional neural networks for mobile vision applications". In: *arXiv preprint arXiv:1704.04861* (2017).

[27] V. Badrinarayanan, A. Kendall, and R. Cipolla. "Segnet: A deep convolutional encoder-decoder architecture for image segmentation". In: *IEEE transactions on pattern analysis and machine intelligence* 39.12 (2017), pp. 2481–2495.

[28] K. H. Zou, S. K. Warfield, A. Bharatha, C. M. Tempany, M. R. Kaus, S. J. Haker, W. M. Wells III, F. A. Jolesz, and R. Kikinis. "Statistical validation of image segmentation quality based on a spatial overlap index1: scientific reports". In: *Academic radiology* 11.2 (2004), pp. 178–189.

[29] F. Chollet et al. *Keras*. `https://keras.io`. 2015.

[30] S. S. Girija. "Tensorflow: Large-scale machine learning on heterogeneous distributed systems". In: *Software available from tensorflow. org* 39.9 (2016).

[31] D. P. Kingma and J. Ba. "Adam: A method for stochastic optimization". In: *arXiv preprint arXiv:1412.6980* (2014).

[32] M. A. Rahman and Y. Wang. "Optimizing intersection-over-union in deep neural networks for image segmentation". In: *International symposium on visual computing*. Springer. 2016, pp. 234–244.

[33] C. Wang, D. Anisuzzaman, V. Williamson, M. K. Dhar, B. Rostami, J. Niezgoda, S. Gopalakrishnan, and Z. Yu. "Fully automatic wound segmentation with deep convolutional neural networks". In: *Scientific Reports* 10.1 (2020), pp. 1–9.

[34] A. K. Adrien Treuille Thiago Teixeira. *Streamlit*. https://streamlit.io. 2020.

[35] D. J. MacKay and D. J. Mac Kay. *Information theory, inference and learning algorithms*. Cambridge university press, 2003.

[36] S. Bhat. "Srb's Manual of Surgery. 4e". In: *Jaypee Brothers Medical Pub* (2013), p. 17.