An-Najah National University

Faculty of Graduate Studies

AUTOMATED OPTIC DISC SEGMENTATION FOR FUNDUS IMAGES BASED ON ARTIFICIAL NEURAL NETWORKS: U-NET

By

Nour Jamal Alhendi

Supervisor

Dr. Hadi Hamad

This Thesis is Submitted in Partial Fulfillment of the Requirements for the Degree of Master of Computerized Mathematics, Faculty of Graduate Studies, An-Najah National . University, Nablus - Palestine

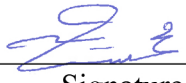
2024

AUTOMATED OPTIC DISC SEGMENTATION FOR FUNDUS IMAGES BASED ON ARTIFICIAL NEURAL NETWORKS: U-NET

By

Nour Jamal Alhendi

This Thesis\Dissertation was Defended Successfully on 27/08/2024 and approved by

Dr. Hadi Hamad	
Supervisor	Signature
Dr. Yousef Daraghmeh	
External Examiner	Signature
Dr. Mohammed Yaseen	
Internal Examiner	Signature

Acknowledgements

First, thanks and praises to Allah for all of this.

To my supervisor Dr. Hadi Hamad, I am grateful for his guidance, support, and patience. I am lucky to learn from his experience.

To my family, I am thankful for their support and encouragement.

Declaration

I, the undersigned, declare that I submitted the thesis entitled:

AUTOMATED OPTIC DISC SEGMENTATION FOR FUNDUS IMAGES BASED ON ARTIFICIAL NEURAL NETWORKS: U-NET

I declare that the work provided in this thesis, unless otherwise referenced, is the researcher's own work, and has not been submitted elsewhere for any other degree or qualification.

Student's Name: Nour Jamal Alhendi

Signature: 

Date: 27/8/2024

List of Contents

List of Tables

List of Figures

List of Appendices

AUTOMATED OPTIC DISC SEGMENTATION FOR FUNDUS IMAGES BASED ON ARTIFICIAL NEURAL NETWORKS: U-NET

By

Nour Jamal Alhendi

Supervisor

Dr. Hadi Hamad

Abstract

Optic disc (OD), located at the back of the eye, is a significant part of the retina. It represents the entry point for the optic nerve and blood vessels. Accurate OD segmentation provides critical information about the anatomy and health state of the retina, aiding in diagnosing and managing various eye conditions such as glaucoma, diabetic retinopathy (DR), and optic nerve abnormalities. With automatic OD segmentation, computer-based systems can efficiently analyze large numbers of retinal images, enabling early detection and monitoring of eye diseases. This automation not only enhances the speed and accuracy of diagnosis but also facilitates cost-effective and accessible healthcare, especially in areas with limited ophthalmic expertise.

In this study, an automatic method for OD segmentation in retinal images using a convolutional neural network (CNN) architecture, known as U-Net, was introduced. First, a region of interest (ROI) was extracted from the fundus images using the bounding box technique. For faster calculations, the cropped images were resized to 128 × 128 pixels. Then, these images were enhanced using the contrast limited adaptive histogram equalization (CLAHE) to eliminate the noise and improve their qualities. After that, a U-Net model was constructed and trained to obtain segmented images.

The proposed model was trained and evaluated using the public dataset ORIGA, and the predicted results were compared with the ground truth (GT) images. This method competed with other studies and achieved average accuracy of 98.42%, average precision of 97.46%, and average sensitivity of 95.33%. As the execution time is short, this enables the proposed method to be an online implemented method.

Keywords: Optic disc segmentation, CNNs, U-Net, ORIGA dataset.

Introduction

The retina is one of the main parts of the eye. It is responsible for translating the received images into signals and passing them to the brain to be processed and realized. Various conditions and diseases can affect and damage the retina causing partial or complete vision loss [21]. Examination and segmentation of the OD can provide valuable information about the health of the retina and help in diagnosing several types of these diseases affecting the retina, such as glaucoma, DR, optic neuritis, and macular edema.

Manual segmentation of the OD is a challenging task. It requires skilled experts to detect the OD boundaries accurately, making it time-consuming and expensive. Moreover, the segmentation can vary among experts; since the detection of the OD boundaries is affected by the noise, image quality, lightness, and several other factors. Additionally, the lack of experts in many countries and regions around the world makes it nearly impossible to diagnose and treat all patients with retinal diseases. Therefore, it is vital to develop automated or semi-automated methods for OD segmentation that is available for almost anyone to benefit from it.

Artificial neural network (ANN) was introduced in 1940 by Warren McCulloch and Walter Pitts as an imitation of the nervous system. ANNs are computational systems consisting of interconnected nodes (artificial neurons) that process and transmit information. ANNs are basically composed of input, hidden, and output layers [33]. The network's ability to learn from large datasets, adapt to complex relationships, and make predictions based on observed patterns, has made it effective in various fields such as finance, engineering, agriculture, medicine, etc.

CNNs are a suitable ANN architecture for image processing tasks. The basic structure of CNNs consists of convolutional, pooling, and fully-connected layers, and it is designed to extract features from images using filters or kernels and make predictions based on the extracted information. The architecture of the CNN allows the network to capture more complex features making it beneficial for image recognition, image segmentation, object detection, and other image processing tasks [44].

A particular type of CNNs, known as U-Net, was first introduced in 2015 by Olaf Ronneberger, Philipp Fischer, and Thomas Brox. The U-Net model was specifically

designed for biomedical image segmentation, and it proved to be effective in working with limited training data and producing detailed segmentation masks [55]. This network has a U-shaped structure, which consists of two paths: an encoding path that extracts the features from the input image, and a decoding path that upsamples the features to produce a segmentation map that matches the input image resolution. Since its introduction, the U-Net architecture has been widely adopted in various medical image analysis tasks, such as tumor segmentation, cell segmentation, retinal vessel segmentation, and optic disc segmentation.

Literature review

Due to the crucial role the OD segmentation plays in the detection and classification of several common fatal retinal diseases such as glaucoma and DR, many researchers have been developing various methods for automatic OD segmentation. Some of the recent studies that worked on the application of U-Net for OD segmentation are discussed.

Sudhan et al. [1] applied the U-Net for OD and optic cup (OC) segmentation. First, the ground truth (GT) (mask) is segmented into two separate regions (the OD and the OC) using the equations (disc = double (mask > 0)) and (cup = double (mask > 1)) respectively. After that, the two segmented regions are used to extract a ROI of size 256 × 256. For the segmentation, the used U-Net architecture consists of 3 blocks in both the encoding and the decoding paths with 112, 224, and 448 filters. This model was trained using the ORIGA dataset. Their work was used to identify Glaucoma and they obtained 98.82% training accuracy and 96.90% testing accuracy.

Due to the downsampling and upsampling operations performed during the encoding and decoding paths of the U-Net, the network may show low sensitivity to fine details and edges. This can cause a loss of information and make it difficult to detect the complex details of the OD edges accurately. To overcome this issue and improve the sensitivity to OD edges, Chen et al. [2] developed a BN-UNet model that combines the regular U-Net structure with a parameter normalization model (BatchNorm). For the preprocessing step, the input images are cropped to remove most of the background. For the segmentation step, a BatchNorm module is added to the U-Net after the convolutional layer and before the Rectified Linear Unit (ReLU) activation function. This helps stabilize and improve the training process and accelerate network convergence. This model used the IDRiD dataset for the OD segmentation, and achieved a sensitivity of 0.9835, a specificity of

0.9975, an accuracy of 0.9972, an area under the curve (AUC) of 0.9458, a dice coefficient of 0.9435, and an intersection over union (IoU) of 0.8932.

Almustofa et al. [3] developed a two-step method for OD segmentation. First, the OD is localized using the Normalized Correlation Coefficient (NCC) map and image brightness. The NCC map is a measure used to compare the similarity between a template and an image. Higher NCC values means stronger similarity, therefore, calculating the NCC at different regions of the images helps identify the regions in the retinal image that closely resemble the OD template. As a result, a ROI of size 550 × 550 is extracted from the original image. After that, for the segmentation part, the ROI is downsampled to 256 × 256, normalized and inputted into the U-Net model to generate a binary OD mask. This study used the Drishti-GS and REFUGE datasets. The performance of the OD segmentation was measured using the F-score and it achieved a value of 0.935 ± 0.031 and 0.950 ± 0.028 on the Drishti-GS and REFUGE datasets respectively.

Panahi et al. [4] proposed a simplified U-Net structure for OD and blood vessels segmentation that consists of 2 blocks in each path with 32 and 64 filters. This model used dropout regularization after the convolutional layer and before the ReLU activation function. The input images of this model are of size 256 × 256. Moreover, data augmentation techniques such as flipping, shifting and rotation were applied to the input images to increase the size of the training set. DRIONS-DB and RIM-ONE v.3 are the datasets used for OD segmentation and the performance was assessed using the dice score and IoU. This model demonstrated a dice score of 0.9401 and an IoU of 0.8754 on RIM-ONE v.3 dataset while it demonstrated a dice score of 0.9469 and an IoU of 0.89 on DRIONS-DB dataset.

Yu et al. [5] proposed a two-stage model that used a U-Net architecture in both stages (one for OD segmentation and the other for OC segmentation). To prepare the data for the OD segmentation in the first stage, the input images were converted from RGB to HSV, and the illumination of these images is adjusted using CLAHE, which reduces the effect of blood vessels when detecting the OD boundaries.

After that, a classic U-Net architecture, which takes input from the value channel and the CLAHE value channel, is used for the process of OD segmentation. To improve the performance of the model, two deep supervision blocks are added after the first two

blocks in the encoding path. Deep supervision refers to the process of adding extra connections to the network that helps improve the accuracy of the segmentation by allowing the low-level layers to gain more semantic knowledge. Instead of obtaining the output only at the final and comparing it with the ground truth using the loss function, intermediate outputs are generated in these deep supervision blocks and are compared to the ground truth during the training process. This provides the low-level layers with direct feedback from the loss function and enables them to learn further semantic information and enhances their ability to detect fine details and features. This study used Drishti-GS1 dataset that consists of 101 fundus images with four ground truth images manually segmented by four different experts for each fundus image. The four manual segmentations are combined, and the area marked by more than two experts is taken as the OD mask leaving the rest as the background. The AUC, recall, precision, and F1-score are the four parameters used to evaluate the performance of this model and they had values of 0.9956, 0.9811, 0.9536, and 0.9660 respectively for the OD segmentation.

Septiarini et al. [6] used the object detection model MobileNet Single Shot Detector (MobileNetSSDv2) to detect the OD and obtain an ROI of size 640 × 640. For the pre-processing stage, augmentation and normalization were applied to the input images. Moreover, the 640 × 640 images were resized to 128 × 128 pixels. The U-Net architecture used for the segmentation consists of 4 blocks in each path with 100 epochs, a batch size of 16, and a learning rate of 0.00001. The numbers of filters used in these blocks were 16, 32, 64, 128, and 256. This study used two datasets: a private dataset that contains 350 retinal images and the REFUGE dataset. The proposed method attained a precision of 0.9992, a recall of 0.9761, an F-score of 0.9880, a dice score of 0.9852, and an IoU of 0.9763 on the private dataset, while it attained a precision of 0.9982, a recall of 0.9718, an F-score of 0.9854, a dice score of 0.9838, and an IoU of 0.9712 on the REFUGE dataset.

Due to the difference in size, Hanifa Suwandoko et al. [7] used the ratios relative to the image size instead of a fixed size of the ROI after localizing the OD, and it was stated that after testing different ratios from 20% to 40%, the ratio 30% gave the optimal results. After segmenting the OD using a U-Net model, ellipses fitting is applied as a post-processing step to refine and accurately represent the shape of the segmented OD. The goal of this process is to find the best ellipse that represents the shape of the OD region

within the segmented area and improve the accuracy of the segmentation results. This study combined two datasets: the Drishti-GS and REFUGE datasets instead of using them separately. Training data consists of 450 images (all 50 images in the Drishti-GS dataset and 400 images from the REFUGE dataset), while both the validation and the testing data consist of 400 images from the REFUGE dataset. This method achieved the average F-Score of 0.945 ± 0.004 for OD segmentation.

Desiani et al. [8] used green channel only and combined the U-Net with augmentation techniques including rotating, flipping, and sharpening the retinal images to generate additional images and increase the size of the training dataset since the U-Net requires a large training set for accurate segmentation. This study used REFUGE dataset and a private dataset. The precision, recall, the F-score, the dice score, and IoU had values of 0.9992, 0.9761, 0.9880, 0.9852, and 0.9763 respectively on the private dataset and 0.9982, 0.9718, 0.9854, 0.9838, and 0.9712 respectively on REFUGE dataset.

Wang et at. [9] used a feature detection sub-network (FDS) and a cross-connection sub-network (CCS) for OD segmentation. FDS is a U-Net model used for features detection and extraction, while CCS gives more important features that help with the segmentation process. The U-Net architecture consists of 5 blocks in each path, and the numbers of filters used were 32, 64, 128, 256, and 512. A 4 × 4 max-pooling layer with a stride of 4 is applied to the first four blocks in the encoding path to construct the CCS, and these four components are combined with the features from the encoding path (from the FDS) and provided into the decoding path of the U-Net. The images were cropped, and field-of-view regions were resized to 256 × 256 pixels. After that, these images were normalized, and image augmentation was performed by randomly flipping the images along the horizontal and vertical axes, translating them by -15 to 15 percent per axis, and rotating them from -90º to 90º. The local disc regions in this training dataset were cropped and resized to 256 × 256 pixels, and both sets (the global field-of-view images and the local disc region images) are used for the OD segmentation. The performance of this model was evaluated using three public datasets: MESSIDOR, ORIGA, and REFUGE. For the ORIGA dataset, this model obtained a dice similarity coefficient (DSC) of 0.9392, an IoU of 0.8873, a Matthew's correlation coefficient (MCC) of 0.9401, and a balanced accuracy (BAC) of 0.9938 for segmenting the global field-of-view images, a DSC of

0.9797, an IoU of 0.9604, a MCC of 0.9692, and a BAC of 0.9869 for segmenting the local disc region images.

Nazir et al. [10] applied rotation and Gaussian blur as augmentation techniques (rotation at the angles of 0º, 90º, 180º, and 270º) to increase the diversity of the training data. After that, annotations for OD and OC are generated using VGG Image Annotator tool. The feature map was extracted using DenseNet-77, and then fed into a region proposal network (RPN) module to obtain ROIs. Lastly, the OD and OC are localized and segmented using these features by a custom Mask-RCNN model. ORIGA dataset was used to assess the performance of this method. For the OD segmentation, this model achieved an accuracy of 0.979, a precision of 0.959, a recall of 0.969, an F-measure of 0.953, and an IoU of 0.981.

In this study, we implemented the U-Net to segment the OD from the fundus images of the ORIGA dataset.

Chapter One

Optic Disc Review

1.1 Introduction to fundus photography

Fundus is the bottom or base of anything. In medicine, it generally means the part of a hollow organ farthest from the opening. The ocular fundus is the inner part of the eye opposite to the lens, which includes the retina, vitreous humor, choroid, and sclera.

A fundus camera is a specialized microscope with an attached camera used to capture retinal images. It is designed based on the principle of the indirect ophthalmoscope. Fundus cameras are described by the field of view, which can vary from 30° to 120° [11]. The German physiologist Hermann von Helmholtz introduced the first ophthalmoscope in 1851 [12]. Figure A.1.1.1 shows the early model of the Helmholtz ophthalmoscope.

Fundus photography documents the status of the eye and monitors if there is any abnormality or change. In addition to diagnosing ocular diseases, fundus imaging allows the detection and diagnosis of hypertensive and cardiovascular diseases. Moreover, the retinal microvasculature is the only part of the human circulation that can be directly visualized non-invasively. Digital fundus imaging offers easy and fast access to analyze the information in retina, which enables immediate access to the results. Fundus photography can be performed with colored filters, or with specialized dyes including fluorescein and indocyanine green [13]. Some retinal images databases are presented in Table 1.1.1.

1.2 The eye structure

The eye is an important organ in the human body which allows vision. It collects images from around and sends them as signals to the brain through the optic nerve. It is composed mainly of two parts: the anterior and the posterior, as shown in Figure A.1.2.1.

The anterior consists of the cornea, iris, ciliary body, and lens, where:

The cornea is the thin avascular transparent front part of the eye. The adult human eye cornea has an average diameter of about 11.5 mm horizontally and 10.5 mm vertically. It bends the light that enters the eye and helps focus it on the retina [20].

Table 1.1.1

A set of retinal images databases

Database name	No. of images	Image resolution (Pixels)	Camera	Field of View (FOV)	GT
DRIVE (Digital Retinal Images for Vessel Extraction) [14]	40	584 x 565	Canon CR5 non-mydriatic 3CCD	45º	Retinal blood vessels
STARE (Structured Analysis of the Retina) [15]	81	700 x 605	TopCon TRV-50	35º	Retinal blood vessels
IDRiD (Indian Diabetic Retinopathy Image Dataset) [16]	516	4288 x 2848	Kowa VX-10a	50º	Microaneurysms Hemorrhages Exudates Optic disc
DRISHTI-GS (Drishti - Retinal Image Dataset for Lesion Detection) [17]	101	2896 × 1944	A traditional high-resolution OD-centric camera	30º	Optic disc Optic cup
REFUGE (Retinal Fundus Glaucoma Challenge) [18]	1200	2124 × 2056 and 1634 × 1634	Zeiss Visucam 500 fundus camera and Canon CR-2	-	Optic disc Optic cup
ORIGA (Online Retinal fundus Image database for Glaucoma Analysis and research) [19]	650	3072 × 2048	-	-	Optic disc Optic cup

The iris is the annular colored structure that lies in front of the lens that has an opening in its center called the pupil. In addition to defining eye color, the iris regulates the amount of light reaching the retina by controlling the pupil's size [20].

The ciliary body is located behind the iris and attached to the lens by the zonular fibers. The ciliary body controls the shape of the lens to adjust focus on objects. It also produces aqueous humor, a transparent fluid responsible for supplying oxygen and nutrients for the lens and cornea and controlling the eye's pressure [21].

The lens, also known as the crystalline lens, is the transparent avascular biconvex structure located behind the iris and supported by the zonular fibers. It changes its shape

to help focus the light on the retina together with the cornea. The change in the curvature of the lens helps the eye focus on objects at various distances. This process is called accommodation.

On the other side, the posterior consists of sclera, choroid, vitreous humor, and retina, where:

The sclera is the white exterior surrounding the entire eye and protecting its interior components from injury.

The choroid is the vascular part of the eye that lies between the sclera and retina and supplies the outer part of the retina with oxygen and blood [20].

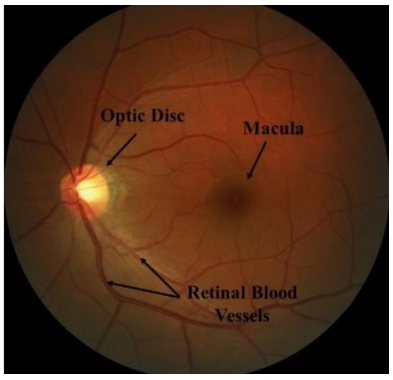
The vitreous humor, also known as the vitreous body, is the thick gel-like fluid that fills the space between the retina and the lens and gives the eye its shape.

The retina is the light-sensitive tissue of the eye that transforms images into neural signals and sends them to the brain to process them. The photoreceptor cells consist of rods and cones. These cells are involved in the process of transforming the images into signals. Neural retina is made up of approximately 7.7 million rods and 5 million cones. Retinal rod cells are responsible for differentiating colors in bright light whereas cone cells take care in distinguishing black and white color in dim light [21].

There are different retinal features like OD, OC, fovea, and macula. Figure 1.2.2 shows a fundus image from the REFUGE dataset.

Figure 1.2.2

A digital image of the retina from the REFUGE dataset



1.3 The Optic Disc

OD, also known as the optic nerve head (ONH), is one of the most important landmarks of the retina. It is made of 1.2 million ganglion cells and is divided into two main areas: neuroretinal rim, the OC (central area) [22]. In color fundus images, where the macula is in the center, the OD appears as a bright yellow or white elliptic area to the left-hand or right-hand side of the image and is about one sixth the width of the image in diameter. [23] OD represents the beginning of the optic nerve, where axons from retinal ganglion cells unite, and is also the entry point into the eye for the major blood vessels that supply the retina. Moreover, OD is called the blind spot; because it contains no photoreceptor cells, and therefore, light incident on the disc does not elicit a response. Figure A.1.3.1 shows the OD in several fundus images from IDRiD dataset [16].

OD plays a major role in detecting diseases, such as glaucoma and DR. Retinal neovascularization in DR appears on the OD. Glaucoma is diagnosed using the cup-to-disc ratio (CDR), that is the ratio of the vertical diameter of the cup to the vertical diameter of the disc. OD can also help locating other landmarks in the retina, such as macula and fovea [22]. The color of the disc, configuration and depth of the OC, CDR, and the rim tissue appearance and disc borders are considered during an ocular health examination. Most OD detection methods in fundus images are based on its shape, color, size, and the fact that it is the convergence point of blood vessels [23].

1.4 Optic disc abnormalities

OD abnormalities can be either congenital or acquired. Congenital abnormalities present from birth and are classified as benign or pathologic. Examples of congenital abnormalities, as shown in Figure A.1.4.1, are optic disc coloboma (ODC), optic pits (OP), and optic disc drusen (ODD). ODC is characterized by a bowl-shaped white excavation of the OD and causes visual field defects with severity depending on the size of the damage in the OD. OP is a round cavity near the margin of the OD that occupies ⅛ - ¼ the size of the disc. Many patients are asymptotic because the OP itself does not affect vision. Vision changes happen when the fluid starts accumulating under the macula causing a decrease in visual acuity. ODD are whitish-yellow dots that appear on the surface of the nerve caused by the abnormal accumulation of protein and calcium within the OD. Patients with drusen usually do not complain of visual symptoms but changes in visual acuity and visual fields can be expected in extreme cases [24].

Acquired abnormalities develop after birth and are assumed to be pathologic. Examples of acquired abnormalities, as shown in Figure A.1.4.2, are ischemic optic neuropathy (ION), and papilledema. ION is one of the main causes of impaired vision and blindness in which the blood flow to the optic nerve is blocked. When the blockage occurs with inflammation of the blood vessels, it's called arteritic (AION). Patients usually suffer from pain when chewing, and tenderness in the temples and scalp. If the blood flow is reduced without inflammation of the blood vessels, it's called nonarteritic (NAION). It causes sudden painless unilateral vision loss, and its severity varies from almost normal to profound vision loss [25]. Papilledema is OD swelling caused by high intracranial pressure (ICP). There are some conditions causing high intracranial pressure, including cerebral hemorrhage, head trauma, and brain tumor.

Patients with papilledema often experience blurred vision, nausea, dizziness, ringing in the ears, headaches and visual disturbances [26].

1.5 Some retinal diseases

The Eye is one of the most sensitive organs in the body. It is vulnerable to several retinal eye diseases, such as glaucoma, DR, and other optic neuropathies. Early detection of retinal diseases can help possibly cure it or prevent it from getting worse. If untreated for an extended period, the disease causes damage to the vision that may lead to blindness.

1.5.1 Glaucoma

Glaucoma is a chronic and progressive optic neuropathy that causes irreversible damage to the optic nerve, which supplies visual information from the eye to the brain. It is caused due to an abnormal increase of Intra-Ocular Pressure (IOP) inside drainage system of the eyes. Glaucoma has no cure, but early diagnosis can help prevent irreversible damage in the eyes [27]. Figure A.1.5.1 shows the difference in vision between a normal person and one with glaucoma.

In 2020, glaucoma was the second primary reason for blindness and the fourth primary reason for moderate and severe vision impairment (MSVI) [28]. The number of people (40-80 years of age) suffering from glaucoma is expected to increase to 111.8 million by 2040 [29].

In fundus images, OD size and the CDR are essential indications of glaucoma, and therefore, OD segmentation plays a major role in glaucoma detection [27]. Figure A.1.5.2 shows the difference between a healthy and glaucoma-suspicious eye.

1.5.2 Diabetic Retinopathy

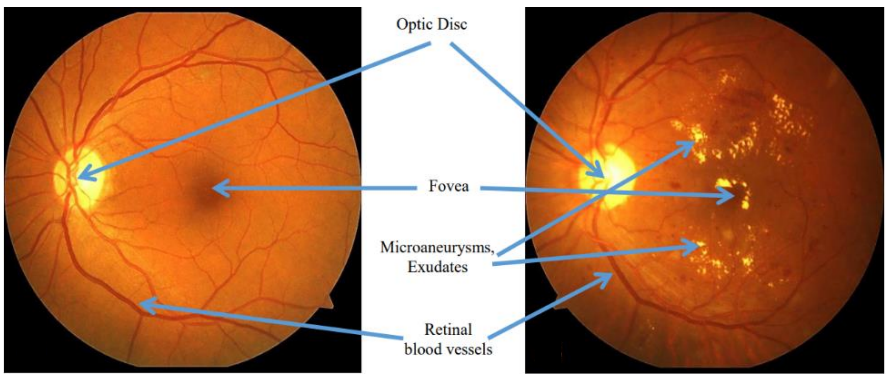
Constant high blood sugar levels caused by diabetes can damage the heart, blood vessels, kidney, nerves, and eyes. DR is a major microvascular complication of diabetes that affects the retina and can cause vision loss if left untreated. Globally, the number of DR patients is expected to grow to 191 million by 2030 [30].

DR patients in the early stages do not show symptoms, so it is hard to detect it in these stages. As the condition progresses, a patient may develop symptoms, such as floaters, blurred vision, and dark areas in their vision [31]. Figure A.1.5.3 shows the difference in vision between a normal person and one with DR.

DR is based on retinal lesions including microaneurysms (MA), hard exudates (EX), soft exudates (SE), and hemorrhage (HE), and it is diagnosed by examining and analyzing color fundus images manually. This technique is expensive and time-consuming due to the large number of diabetic patients around the world and the lack of experts and infrastructure in developing countries. Therefore, many automatic techniques, including OD localization, blood vessels segmentation, image boundary tracing, and others, have been developed for early DR detection [31]. Figure 1.5.4 shows a normal fundus image and one with DR.

Figure 1.5.4

Fundus images: (a) Normal fundus image (b) Fundus image with DR



(a) (b)

1.6 Medical image processing

Recently, artificial intelligence (AI) has played an essential role in the medical field, more specifically in medical image processing. Medical image processing is applying algorithms to analyze and extract information from medical images such as X-rays, MRI scans, CT scans, and others. Medical image processing helps with diagnosing, monitoring, and planning treatment of numerous diseases. It includes several techniques such as image enhancement, detection, segmentation, and classification [32], which will be explained as follows.

Image enhancement is the process of improving the quality of the image, highlighting specific details, and reducing the noise using several techniques such as spatial filtering and contrast enhancement. Enhancing medical images gives clearer and more informative detailed images for accurate diagnosis and analysis.

Detection means locating certain items in the image. In medical image processing, detection is used to locate specific anatomical structures and provide more information for further studies and research. Moreover, detecting any abnormalities in the medical image aids with early diagnosis, and sometimes prediction, of the disease.

Segmentation is the process of extracting a particular object or region, separating it from the background in the image, and creating a pixel-wise mask for this object. In medical images, segmentation provides vital information such as the size, shape, volume, and spatial relationships between anatomical structures, which allows precise measurements and analysis of these structures. Moreover, accurate segmentation aids in early recognition of any abnormalities that can help diagnose -or even predict- many diseases, monitor their progress, and set treatment plans.

Classification refers to the action of sorting objects into categories based on their similar features. In medical images, classification is used to determine the presence or absence of certain structures, conditions, abnormalities, or diseases by extracting relevant features and patterns from these images. Classification plays a major role in providing systems that can automatically analyze medical images and give an accurate diagnosis for the disease [32].

The OD segmentation in retinal images is a critical task in medical image analysis for diagnosing several eye diseases. Manual OD segmentation is time-consuming and

subjective. The main aim of this thesis is to contribute to the advancement of automated OD segmentation using artificial neural networks, specifically the U-Net architecture, in the field of medical image analysis.

The objectives of this research are as follows: Firstly, to develop a U-Net based model customized for OD segmentation in retinal images. Secondly, to optimize the network's performance by fine-tuning the U-Net model to achieve high performance, while considering the challenges posed by varying image qualities and pathological conditions. Additionally, to evaluate the model quantitatively using metrics such as the accuracy, precision, sensitivity, specificity, F-score, and intersection over union (IoU) on a diverse dataset of retinal images. Lastly, to compare the proposed U-Net model with other studies showing its competitivity in terms of accuracy, efficiency, and robustness.

Chapter Two

Artificial Neural Networks (ANNs)

2.1 Artificial Neural Networks

ANN is a branch of machine learning that was first proposed by Warren McCulloch and Walter Pitts in 1940 and aimed to mimic the functioning of a biological neuron [33].

2.1.1 Biological inspiration

Neurons, or nerve cells, are the fundamental units of the nervous system. When the neurons are connected, they form complex neural networks that can perform many parallel tasks. Therefore, scientists aim to develop computer-based programs that imitate the human brain in solving problems in different fields of life [34].

As shown in Figure A.2.1.1, the main parts of a biological neuron are the dendrite, the cell body (soma), and the axon [34]. The dendrites receive signals from other neurons and collect them in the cell body which in turn produces a response and sends it through the axon to the dendrites of other neurons [35].

2.1.2 ANN architecture

ANNs are designed to simulate the function of the brain, Figure A.2.1.2 shows the basic structure of an ANN. It consists of an input layer, one or more hidden layers and an output layer. The input layer contains the multidimensional vector provided to the network, and it will be passed to the hidden layers that will process it and make decisions [34].

The nodes in one layer are connected to the nodes in the next layer with adjustable coefficients called weights. In each node, all inputs modified by their respective weights are summed up together with the bias, which can be represented mathematically as:

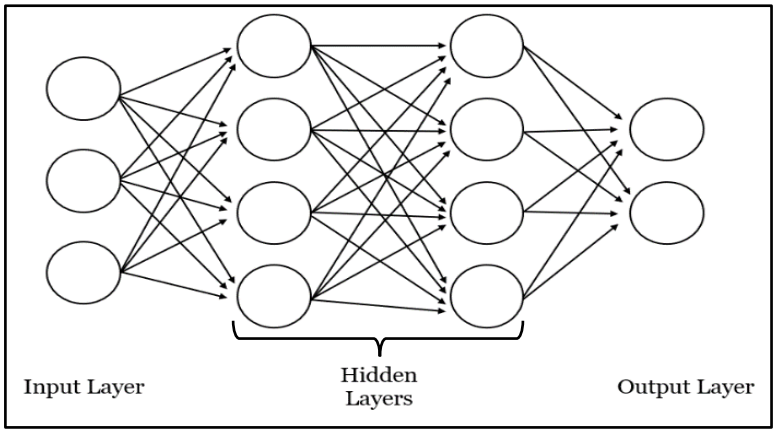
$$y_j = \sum_{i=1}^{m} w_{ji} x_i + b \qquad 2.1.1$$

where y_j : the j^{th} output, x_i : the i^{th} input, w_{ji} : the weight connecting to the input x_i with the output y_j, b : the bias, and m : the dimension of the input.

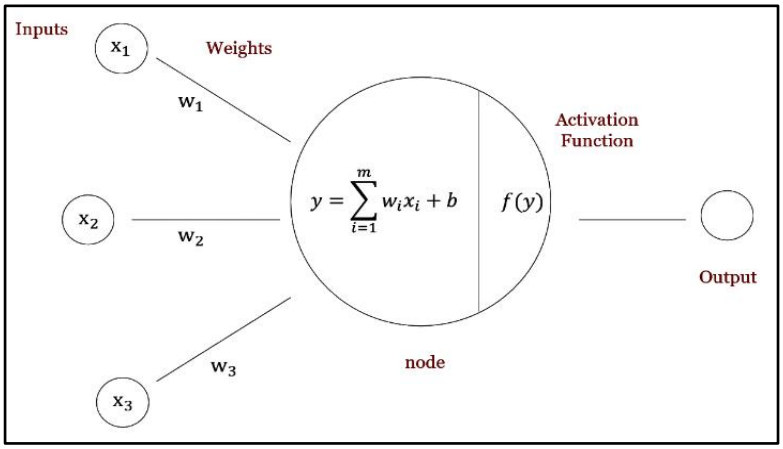
Activation function is used to transfer the inputs of a node into an output, which will be an input of the next layer [34]. If that sum exceeds a given threshold, it fires (or activates) the node, passing data to the next layer in the network, as shown in Figure 2.1.2.

Figure 2.1.2

An ANN: (a) The basic structure (b) A model of an artificial neuron (node)



(a)



(b)

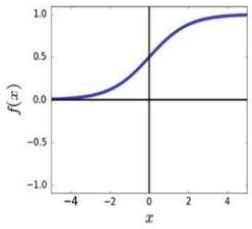
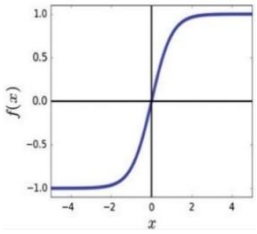
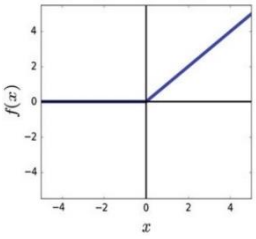
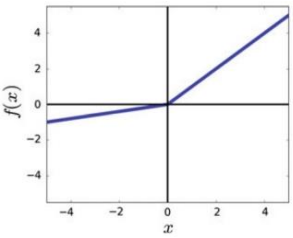
Activation function is represented in the equation:

$$output = f\left(\sum_{i=1}^{m} w_i x_i + b\right) \qquad 2.1.2$$

If a neural network doesn't contain an activation function, the output is going to be a simple linear function of the inputs and all the layers will behave the same way, and that is useless when the tasks are more complex. A set of common non-linear activation functions are represented in Table 2.1.1.

Table 2.1.1

Several types of non-linear activation functions

Sigmoid/ Logistic	Hyperbolic Tangent (Tanh)	Rectified Linear Unit (ReLU)	Leaky Rectified Linear Unit (LReLU)
$f(x) = \frac{1}{1+e^{-x}}$	$f(x) = \frac{e^{x}-e^{-x}}{e^{x}+e^{-x}}$	$f(x) = max(0, x)$	$f(x) = max(0.1\,x, x)$
			

2.1.3 ANN applications

ANNs have been widely applied in many fields, such as medicine and healthcare, industry, agriculture, business, finance, etc. In the field of medicine and healthcare, ANNs models are used in medical data analysis, which includes scalar data (heart rate, blood pressure, glucose level, etc.) and image data (MRI scans, CT scans, retinal images, etc.). Medical data analysis provides vital information that can help with medical diagnosis. Over the years, several researchers have used ANNs in the diagnosis of retinal diseases, diabetes, cancer, and cardiovascular diseases [36].

Applications of ANNs in finance involve stock market analysis, future price estimation, and financial fraud detection. Moreover, ANNs have been used to forecast bankruptcy of a company or a business and decide whether it deserves a loan or not. Additionally, ANNs help companies make investment decisions based on the prediction of foreign currency exchange rates [37].

Other applications of ANNs include handwriting, speech and facial recognition, signature authentication, fingerprint identification [38], and autopilot in planes and cars.

2.2 Machine learning algorithms

The neural network learning method is the process of adjusting or modifying the weights between layers and defining their biases. The main three learning algorithms are supervised learning, unsupervised learning, and reinforcement learning [35], as shown in Figure A.2.2.1.

2.2.1 Supervised learning

In supervised learning, the training dataset consists of the inputs and their corresponding actual outputs (labels). The goal is to build a machine learning model that can map each input in the training set to its matching label, and then, generalize this model and apply it on a new unlabeled data [39, 40]. Figure A.2.2.2 shows the process of supervised learning.

Supervised learning algorithms fall into two categories: classification and regression algorithms [40]. Classification algorithms categorize the inputs into a finite number of classes, as shown in Figure A.2.2.3. Classification can be binary, such as classifying a human face image into male or female, or multi-class, such as handwritten numbers recognition.

Meanwhile, the predicted output of the regression algorithms is a real-value variable [40], such as stock price prediction and weather forecasting. Figure A.2.2.4 shows the difference between classification and regression.

2.2.2 Unsupervised learning

In unsupervised learning the network is trained using a set of unlabeled data, which means that the output is not priorly known, and the system is supposed to find hidden patterns in the data on its own [35, 39]. Figure A.2.2.5 shows the process of unsupervised learning.

Clustering is the most common type of unsupervised learning algorithms. It maps similar data into the same group or cluster [39], as shown in Figure A.2.2.6. Clustering can be used in many applications. In marketing, for example, customers are clustered together according to their similar behavior.

2.2.3 Reinforcement learning

Reinforcement learning is a feedback-based type of learning in which the network is provided with an initial state of the environment. The network, also known as the agent, is trained by taking actions and getting feedback from the environment, as shown in Figure A.2.2.7. Reinforcement learning can be either positive or negative; the goal is to maximize the reward in positive reinforcement and minimize the risk (penalty) in negative reinforcement [35].

2.3 Feed forward neural network

Neural networks are categorized based on the type of connections between neurons into two main types: feed forward neural networks (FFNNs), and recurrent neural networks (RNNs). FFNNs are networks with information processed in only one direction and without feedback from the outputs [41], while RNNs are networks with information processed in two directions (forward and backward) and with feedback from the outputs.

2.3.1 The perceptron rule

Frank Rosenblatt proposed the single layer perceptron (SLP) in the late 1950s [41]. It is the simplest type of FFNNs that consists of two layers, the input and the output layers. It is mainly applied in binary classification tasks, where data is categorized into two linearly separable classes using the unit step activation function. Figure A.2.3.1 shows the network of a SLP.

As shown in Figure A.2.3.2, linearly separable data means that data can be separated and categorized into two classes using a straight line. Otherwise, the data is said to be nonlinearly separable.

The output of a perceptron is defined as:

$$y = \begin{cases} 1, & if \ \sum_i w_i x_i + b \ \geq 0 \\ 0, & otherwise \end{cases} \qquad 2.3.1$$

where x_i are the inputs, w_i are the weights, and b is the bias [41].

In a neural network, the value that determines whether a neuron should be activated, and signals be transmitted to the next layer or not is called the threshold. For the perceptron,

if the linear combination of the inputs is greater than or equal some threshold $-b$, the neuron will be activated and its output y will equal 1. Otherwise, the neuron will not be activated and its output y will equal 0. The process of adjusting the weights and the bias of a perceptron is called the perceptron learning rule [41]. Starting with random initial parameters, the predicted output y is calculated using Equation 2.3.1 and compared to the given target t, that is the actual output. If they are equal, then there is no need for further adjustments to the parameters. If not, they must be modified as follows:

$$E = t - y$$
$$w_i^{new} = w_i^{old} + Ex_i \qquad 2.3.2$$
$$b^{new} = b^{old} + E$$

where E is the error. The new weights result from adding the inputs multiplied by the error to the old weights, and the new bias results from adding the error to the old bias [41].

2.3.2 Backpropagation algorithm

A multi-layer perceptron (MLP) has one or more extra layers, known as hidden layers, that allow the MLP to solve classification tasks where data is not linearly separable. Each hidden layer contains a differentiable nonlinear activation function. Figure A.2.3.3 shows the network of MLP.

The perceptron rule fails to solve nonlinearly separable classification tasks, and hence, a different learning rule called backpropagation (BP) is used to train MLP. It was first introduced in 1974 by Paul Werbos. However, it was not widely known until the 1980s when it was discovered independently by David Parker and Yann LeCun in 1985, and David Rumelhart, Geoffrey Hinton and Ronald Williams in 1986 [41]. BP is a supervised learning algorithm that modifies the weights and biases of the network using the mean square error (MSE) and gradient descent to get accurate output and minimize the error. This algorithm consists of two phases: forward and back propagation [42]. During the forward propagation phase, the weights and bias are constants, and the input is propagated to get the predicted output.

Switching to the back propagation phase if the error between the real output and the predicted one does not equal zero or close to it. Working backward from the output layer to the input layer to adjust the weights to minimize the error [42].

It is important to know how the change in the weights affects the error to minimize it, and that will be by using the gradient of the error with respect to the weights (∇E). Updating the weights will be in steps in the direction of the negative gradient to reach the weights that give the minimum value of error, and that is called the point of convergence, as shown in Figure 2.3.4.

The hyperparameter that determines the size of the step at which the weights are updated is called the learning rate. Some algorithms use a fixed learning rate for all parameters, while others adjust the learning rate during the training process [43]. Higher values of learning rate mean larger updates to the weights and faster training. However, the optimization process is more likely to diverge with larger learning rates, as shown in Figure 2.3.4.

Meanwhile, using lower learning rates means smaller updates to the weights and slower convergence. This gives the model better stability and reduces the risk of divergence. The weights are updated as follows:

$$\Delta w = -\eta \nabla E \tag{2.3.3}$$

$$\nabla E = \begin{bmatrix} \frac{\partial E}{\partial w_1} \\ \vdots \\ \frac{\partial E}{\partial w_n} \end{bmatrix}$$

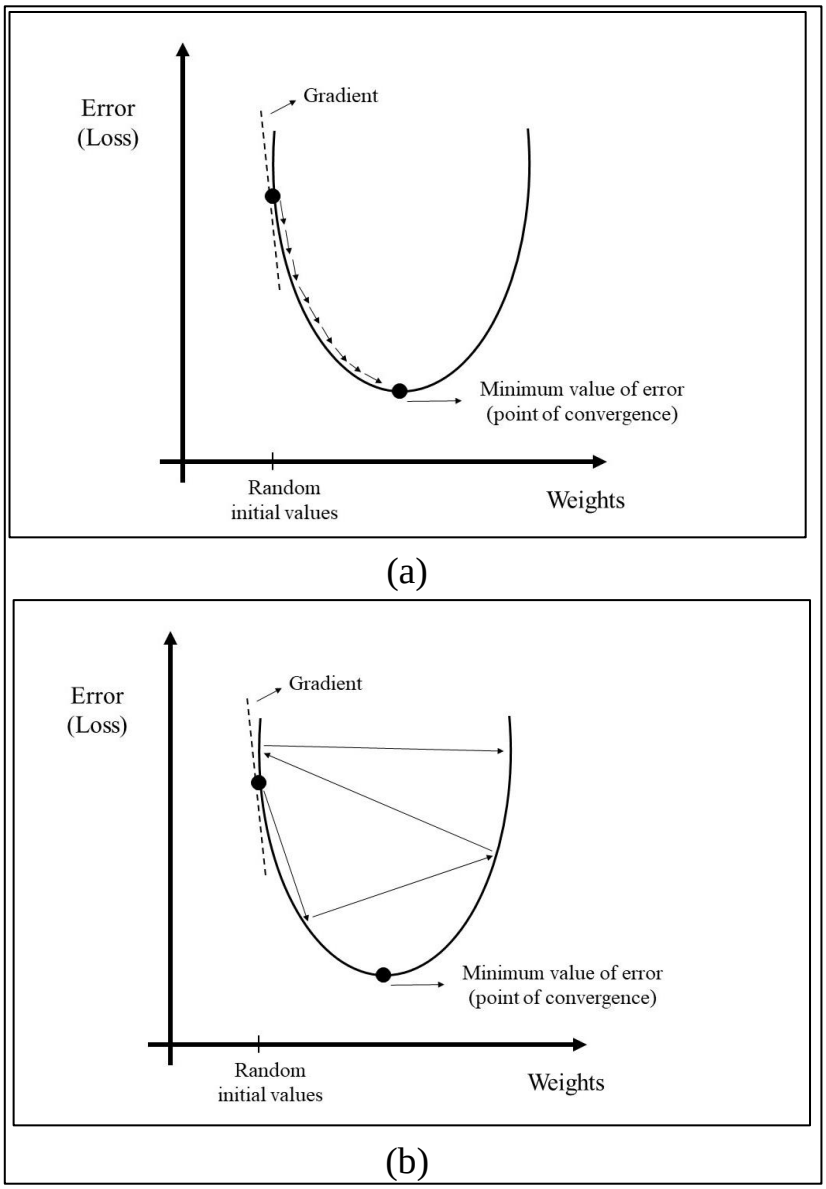
where n is the number of weights, and η is the learning rate.

Considering a three-layer network as illustrated in Figure A.2.3.5.

where x_i is the node of the input layer, z_j is the node of the hidden layer, y_k is the node of the output layer, w_{ji} is the weight from the input to the hidden layer, w_{kj} is the weight from the hidden to the output layer, $f(\cdot)$ is the activation function, b_j is the bias of the hidden layer, and b_k is the bias of the output layer.

Figure 2.3.4

(a) Minimizing the error of the output using the gradient of the error function with respect to the weights (b) The effect of using larger learning rates on the optimization process



The output of a neuron in the hidden layer z_j is [41, 42]:

$$z_j = f_j\left(\sum w_{ji}x_i + b_j\right) \qquad 2.3.4$$

The output of a neuron in the output layer y_k is:

$$y_k = f_k\left(\sum w_{kj} z_j + b_k\right) \quad 2.3.5$$

Given an actual output called target t_k for each predicted output y_k, the error can be written as:

$$E = \frac{1}{2}\sum (t_k - y_k)^2 \quad 2.3.6$$

Modifying a particular weight of a neuron in the output layer:

$$\Delta w_{kj} = -\eta \frac{\partial E}{\partial w_{kj}} \quad 2.3.7$$

Using the chain rule to redefine $\frac{\partial E}{\partial w_{kj}}$:

$$\frac{\partial E}{\partial w_{kj}} = \frac{\partial E}{\partial y_k}\frac{\partial y_k}{\partial w_{kj}} \quad 2.3.8$$

$$\frac{\partial E}{\partial y_k} = \frac{1}{2} * 2(t_k - y_k) * (-1) = (y_k - t_k) \quad 2.3.9$$

$$\frac{\partial y_k}{\partial w_{kj}} = \frac{\partial f_k\left(\sum w_{kj} z_j + b_k\right)}{\partial w_{kj}} = {f'}_k\left(\sum w_{kj} z_j + b_k\right) * z_j \quad 2.3.10$$

Substituting Equations 2.3.9 and 2.3.10 in Equation 2.3.8 gives:

$$\frac{\partial E}{\partial w_{kj}} = (y_k - t_k) * {f'}_k\left(\sum w_{kj} z_j + b_k\right) * z_j \quad 2.3.11$$

Hence, the updated weight in Equation 2.3.7 will be:

$$\Delta w_{kj} = -\eta\, \delta_k\, z_j \quad 2.3.12$$

where $$\delta_k = (y_k - t_k) * {f'}_k\left(\sum w_{kj} z_j + b_k\right)$$

Next, modifying a particular weight of a neuron in the hidden layer:

$$\Delta w_{ji} = -\eta \frac{\partial E}{\partial w_{ji}} \quad 2.3.13$$

Using the chain rule to redefine $\frac{\partial E}{\partial w_{ji}}$:

$$\frac{\partial E}{\partial w_{ji}} = \frac{\partial E}{\partial y_k}\frac{\partial y_k}{\partial z_j}\frac{\partial z_j}{\partial w_{ji}} \quad 2.3.14$$

$$\frac{\partial y_k}{\partial z_j} = \frac{\partial f_k\left(\sum w_{kj} z_j + b_k\right)}{\partial z_j} = {f'}_k\left(\sum w_{kj} z_j + b_k\right) * w_{kj} \quad 2.3.15$$

$$\frac{\partial z_j}{\partial w_{ji}} = \frac{\partial f_j\left(\sum w_{ji} x_i + b_j\right)}{\partial w_{ji}} = {f'}_j\left(\sum w_{ji} x_i + b_j\right) * x_i \quad 2.3.16$$

Substituting Equations 2.3.9, 2.3.15, and 2.3.16 in Equation 2.3.14 gives:

$$\frac{\partial E}{\partial w_{ji}} = (y_k - t_k) * {f'}_k\left(\sum w_{kj} z_j + b_k\right) * w_{kj} * {f'}_j\left(\sum w_{ji} x_i + b_j\right) * x_i$$

$$\frac{\partial E}{\partial w_{ji}} = \delta_k * w_{kj} * {f'}_j\left(\sum w_{ji} x_i + b_j\right) * x_i \quad 2.3.17$$

Hence, the updated weight in Equations 2.3.13 will be:

$$\Delta w_{ji} = -\eta\, \delta_j\, x_i \quad 2.3.18$$

where $\delta_j = \delta_k * w_{kj} * {f'}_j\left(\sum w_{ji} x_i + b_j\right)$

Using the same process to modify the bias of a neuron in the output and hidden layers:

$$\Delta b_k = -\eta\, \delta_k \quad 2.3.19$$

where $\delta_k = (y_k - t_k) * {f'}_k\left(\sum w_{kj} z_j + b_k\right)$

$$\Delta b_j = -\eta\, \delta_j \quad 2.3.20$$

where $\delta_j = \delta_k * w_{kj} * {f'}_j\left(\sum w_{ji} x_i + b_j\right)$

Therefore, the BP algorithm can be used to train MLP by modifying the weights and biases while minimizing the error between the predicted output and the given target.

2.3.3 A glance on convolutional neural networks

CNN is a subclass of ANNs used for image recognition in general; however, it is different from the traditional ANN models because it focuses on the knowledge of specific parts of input instead of all the problem domain, which helps us establish a simpler network architecture [44]. CNNs consist of different types of layers and are usually trained using backpropagation algorithm [45], as we will see in detail in chapter 3.

Chapter Three
Convolutional Neural Networks (CNNs)

3.1 Introduction

In 1959, Hubel and Wiesel discovered that several receptive fields stimulate the retina to send information to the brain. As a result of that discovery, in 1980, Kunihiko Fukushima proposed Neocognitron, the first multi-layer neural network model based on the visual nervous system. The modern structure of CNN was introduced in 1998 when Yann LeCun proposed LeNet-5, a simple CNN model trained using backpropagation algorithm and was successfully applied on handwritten numbers recognition. In 2012, Alex Krizhevsky developed a larger model called AlexNet and won the ImageNet competition for image classification. After that, several CNN architectures have been developed, such as, ZFNet (2013), VGGNet and GoogLeNet (2014), ResNet (2015), and DenseNet (2016) [44].

Through the years, CNN proved to be a powerful tool in many areas of life. One notable application is handwritten number recognition, where CNNs have been used to accurately identify and classify handwritten numbers using large training datasets such as MNIST and SVHN [46]. CNNs are also widely used in several computer vision tasks including object detection. They can identify objects within images or videos by extracting their features and then locate these objects with bounding boxes. Facial recognition is another important application of CNNs. They learn to extract unique facial features and use them for face matching and identification. DeepFace and FaceNet are examples of CNN-based models that have achieved remarkable performance in facial recognition tasks [47]. Additionally, CNNs are developed to enhance the safety and capability of autonomous driving systems. They are used to automatically detect lanes, pedestrians, and surrounding cars on the road without human intervention [48]. Lastly, CNNs have also achieved significant performance in medical imaging analysis. They assist in the detection and diagnosis of anomalies and diseases from various medical imaging modalities like X-rays, MRIs, and CT scans.

3.2 Advantages and disadvantages of CNNs

CNNs have been successfully used in several machine learning tasks with high accuracy rates. They have the benefit of hierarchical feature learning, which means that they can automatically detect simple features in earlier layers and gradually learn more complex

features in deeper layers. This allows the CNNs to extract effective information from the images and work on various tasks such as object recognition, image classification, and segmentation. Moreover, CNNs can locate the feature or pattern in the image regardless of its location, size, and color. Additionally, CNNs share the same weights across different regions in the image significantly reducing the number of calculations needed and making them fast and effective for working on large datasets and complex tasks [44].

Even though CNNs have achieved impressive results, they have some limitations and disadvantages. Containing multiple layers can be time-consuming due to the increase in the number and complexity of the calculations. Moreover, overfitting is a common problem that CNN encounters. It occurs when the network has learned the training data too well, but fails to generalize to new unseen data, as shown in Figure A.3.2.1. Working on small training datasets is one of the reasons that causes the overfitting; because the network can easily memorize the training examples, failing to capture the underlying patterns that generalize well to unseen data [49].

Additionally, CNNs sometimes require powerful hardware resources, such as GPUs or specialized accelerators when dealing with large training datasets and complex architectures, which may not be available for everyone. Finally, it can be difficult to interpret the reasoning behind the predictions and decisions of the network. This creates an issue in the cases where an explanation is required and important [50]. Researchers aim to overcome these shortcomings and improve the ability of CNNs to correctly solve more problems in various fields.

3.3 CNN structure

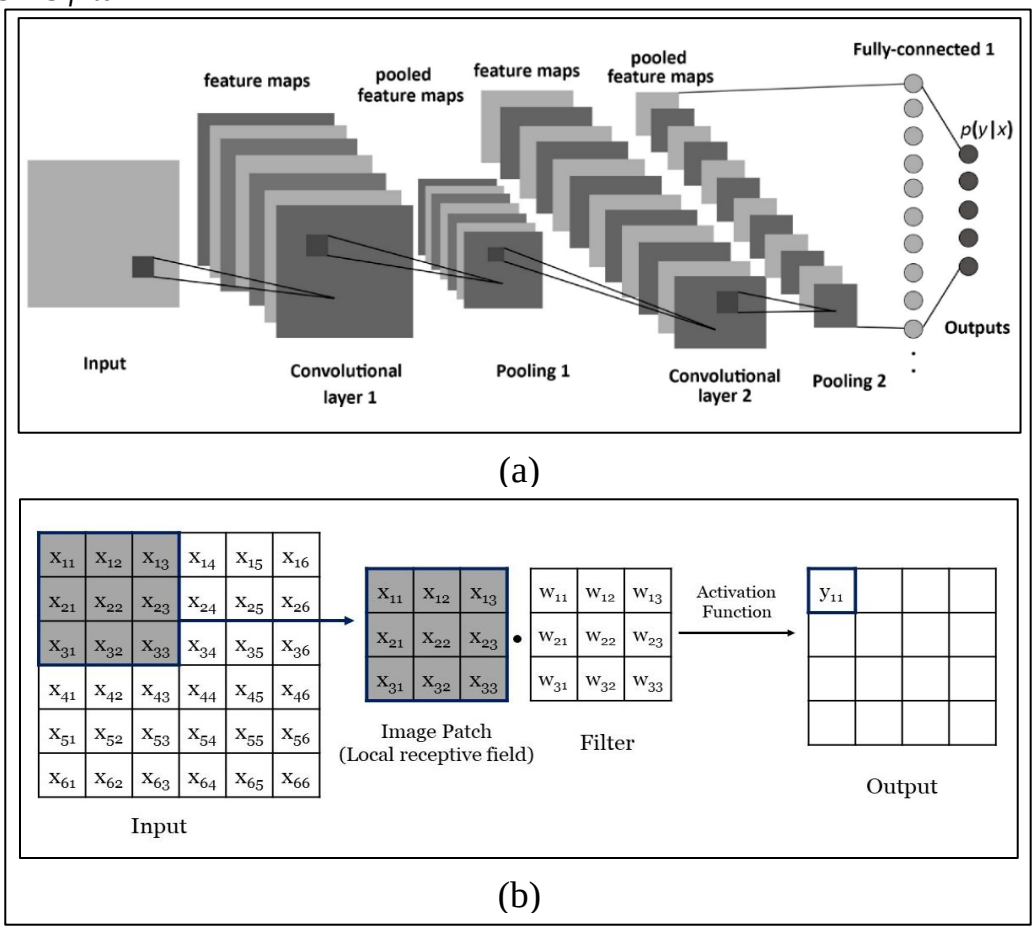
All different CNN architectures have a standard structure that consists mainly of convolutional, pooling, and fully-connected layers as shown in Figure 3.3.1. Other types of layers can be added to the network, such as batch normalization, dropout, and transposed convolutional layers.

3.3.1 Convolutional layer

The convolutional layer is composed of a set of kernels or filters, which are 2-dimensional matrices, where their entries are the weights. To analyze an image, these filters are used to detect and extract specific characteristics or patterns, called features, from this image. The features can be as simple as lines or curves, or as complex as faces and objects. A feature map is obtained by sliding the filter over the input image starting from the upper-left corner and calculating the scalar product between the filter and a small region of the image called the local receptive field. After that, a non-linear activation function is applied to the result. The process can be seen in Figure 3.3.1. Using the same filters on all the receptive fields helps decrease the number of parameters to train and extract the same feature despite the change of its location, transformation, and direction in the input image [44, 45].

Figure 3.3.1

A CNN model: (a) A simple structure (b) A convolutional layer with a 6 × 6 input and a 3 × 3 filter



Some of the commonly used activation functions in CNNs are introduced here [44].
Rectified linear unit (ReLU) is a widely used non-linear activation function in neural networks that makes training the CNN faster than using sigmoid or tanh functions. Positive input of the ReLU function remains unchanged, while negative input becomes zero. ReLU function is mathematically represented as:

$$f(x) = \max(x\,, 0) \tag{3.3.1}$$

Leaky ReLU (LReLU) was introduced to overcome the dying ReLU issue, in which the nodes may become permanently inactive if they constantly receive negative inputs during training. It allows a small, non-zero slope for negative inputs, keeping the nodes partially active. LReLU function is mathematically represented as:

$$f(x) = \max(\lambda x\,, x) \tag{3.3.2}$$

where λ is a predefined parameter between 0 and 1.

Parametric ReLU (PReLU), similar to LReLU, allows the negative slope to be learned during training. However, instead of using a fixed parameter λ, PReLU adjusts the value of the parameter in each iteration through backpropagation to determine the optimal slope for negative inputs. PReLU function is mathematically represented as:

$$f(x) = \max(\lambda_t x\,, x) \tag{3.3.3}$$

where λ_t is the value of the parameter at the iteration t.

Exponential Linear Unit (ELU) has a smooth curve for negative values and a continuous gradient that helps with faster convergence of the network. ELU function is mathematically represented as:

$$f(x) = \begin{cases} x & x \geq 0 \\ \lambda(e^x - 1) & x < 0 \end{cases} \tag{3.3.4}$$

where λ is a positive constant that commonly ranges between 0 and 1.

The graphs of the activation functions ReLU, LReLU, PReLU, and ELU are illustrated in Figure A.3.3.2. Each one of these functions is non-linear and helps mapping complex relationships between inputs and outputs. Choosing the proper activation function depends on the specific task and network architecture.

In addition to the filter size, there are other parameters that control the output size, such as the depth, stride and padding.

The depth of the output is defined by the number of its layers based on the number of filters used in the network [51].

The stride is the number of steps the filter moves over the pixels of the input image. It determines how many pixels the filter jumps horizontally and vertically after each convolutional operation. The stride affects the output size of the convolutional layer. Using larger stride results in a significant reduction in the output size, but that can also reduce the resolution of the output [46]. Therefore, it is better to use smaller strides when working on fine-grained features.

Padding, or zero-padding, means adding rows and columns of zeros around the input image to control the decrease of the output size and prevent the loss of information in the corners of the image since the filters slide over them only once [46].

Therefore, the output size of a convolutional layer is:

$$output\ size = \frac{(M - K) + 2P}{S} + 1 \qquad 3.3.5$$

where M is the size of the input image, K is the filter size, P is the number of zero-padding added, and S is the stride. If the calculated output size is not integer, this means that the used stride is improper, and the filter does not fit across the input image because of this stride [51].

In general, padding the input image of size $M \times M$ gets a new input image of size $(M + P) \times (M + P)$. If a $K \times K$ filter with stride S is applied to the new padded input image, the $(ij)^{th}$ entry in the feature map (output) will be:

$$y_{ij} = f\left(\sum_{r=0}^{k-1}\sum_{t=0}^{k-1} w_{r+1,t+1}\tilde{x}_{S*i+r-1,S*j+t-1} + b_{ij}\right) \qquad 3.3.6$$

where $\tilde{x}$ are the entries of the new padded input image.

3.3.2 Pooling layer

A convolutional layer is usually followed by a pooling layer, also known as down-sampling layer. It reduces the size of the input image without losing its important features. Therefore, it reduces the number of parameters needed to learn and simplifies the model [51]. A sample of pooling is given in Figure A.3.3.3.

The pooling layer divides the image into sub-regions and performs a pooling operation on each one of them. The two popular pooling operations are max-pooling and average-pooling. In max-pooling, only the maximum value of the sub-region is selected, while average-pooling calculates the average of all the values in the sub-region [46].

Same as the convolutional layer, the filter size, stride, and padding are the parameters that control the output size of the pooling layer. Therefore, the output size of the pooling layer can be calculated using Equation 3.3.5. However, unlike the convolutional layer, the depth of the output in the pooling layer is the same as the depth of the input.

The $(ij)^{th}$ entry of the output after applying max-pooling to the padded input image with stride S will be:

$$z_{ij} = max\{\tilde{y}_{rt}, r = S*i-1, \dots, S*i+R-2, t = S*j-1, \dots, S*j+R-2\} \quad 3.3.7$$

where R is the size of the pooling and $\tilde{y}$ are the entries of the new padded input image.

Meanwhile, the $(ij)^{th}$ entry of the output of applying average-pooling to the padded input image with stride S will be:

$$z_{ij} = \frac{\sum_{r=S*i-1}^{S*i+R-2} \left(\sum_{t=S*j-1}^{S*j+R-2} (\tilde{y}_{rt}) \right)}{R^2} \quad 3.3.8$$

3.3.3 Fully-connected layer

The last layers of a CNN are fully-connected layers. Similar to the layout of the regular ANN, the nodes in a fully-connected layer are connected to all the nodes in the two adjacent layers [51]. For classification tasks, the final layer often uses softmax activation function to classify the input images into classes based on the extracted features [46].

Sometimes, a CNN model focuses on learning and memorizing the noise of the training data instead of learning the features and patterns needed to generalize the model to new data, this case is called overfitting, in which some nodes change to fix the mistakes of other nodes causing complex co-adaptions. As a result, the model will perform very well on the training data but will fail when applied on unfamiliar data. Several techniques, such as batch normalization and dropout, are used to reduce the overfitting in a CNN model [49].

3.3.4 Batch normalization layer

In RGB images, each pixel has a value that ranges from 0 to 225, and such a large range makes the training process more complicated. To avoid that, it is preferable to have all pixels in a smaller and more manageable range, and that is commonly done using either normalization or standardization.

Normalization, also known as min-max normalization, sets the minimum and maximum values of the data to 0 and 1 respectively, and the rest of the data will have values between 0 and 1 according to the equation:

$$\hat{x} = \frac{x - min}{max - min} \tag{3.3.9}$$

where x is the original value, and $\hat{x}$ is the normalized value.

Standardization, also known as z-score normalization, is the process of setting the mean and the standard deviation of the data to 0 and 1 respectively. The standardization process is described using the equation:

$$\hat{x} = \frac{x - \mu}{\sigma} \tag{3.3.10}$$

where μ is the mean value of the data, and σ is the standard deviation.

In batch normalization layer, the pixel values are normalized using the z-score normalization and multiplied by an arbitrary parameter α. Then, another arbitrary parameter β is added to the result. Mathematically:

$$\hat{x} = \left(\frac{x - \mu}{\sigma}\right) * \alpha + \beta \tag{3.3.11}$$

Batch normalization layer is normally added after a convolutional layer or fully-connected layer, but before the activation function. To make the process of normalization faster, the pixel values are normalized in small batches instead of the image as whole [52].

3.3.5 Dropout layer

Dropout is a technique used to cancel some of the nodes of the neural network temporarily by a dropout probability of p, removing all the connections to these nodes and preventing them from contributing to the training of the network. Randomly dropping nodes can help generalize the model and prevent co-adaption in the network, and thereby, decrease the network overfitting. Dropout layer can be applied to either the input or the hidden layers [46, 53].

When applying a dropout layer, each input value is multiplied by an independent Bernoulli random variable m that has two values: 0 or 1. The probability of m being 0 (also known as drop probability) is p. On the other side, the probability of m being 1 (also known as keep probability) is q, where $q = 1 - p$. Hence, Equation 3.3.6 will become [53]:

$$y_{ij} = f\left(\sum_{r=0}^{k-1}\sum_{s=0}^{k-1} w_{r+1,s+1}\hat{x}_{i+r,j+s} + b_{ij}\right) \qquad 3.3.12$$

where $$\hat{x}_{i,j} = m_{i,j} * x_{i,j}$$

3.3.6 Transposed convolutional layer

Transposed convolutional layer is an up-sampling layer used to create an output with size larger than the input. It is commonly used for image segmentation, image generation, and image resolution improvement. Sometimes a transposed convolutional layer is mistakenly referred to as deconvolutional layer. A deconvolutional layer reverses the process of a convolutional layer, which means that applying a deconvolutional layer after a convolutional layer gets the original input. Meanwhile, the output of a transposed convolutional layer will be of the same size but with different values, as shown in Figure A.3.3.4.

For a given $M \times M$ input and $K \times K$ filter with stride S and padding P, the output size of transposed convolutional layer is [54]:

$$output\ size = \big((M-1)\times S + K - 2P\big)\ \times\ \big((M-1)\times S + K - 2P\big) \quad 3.3.13$$

The first step to get the output of a transposed convolutional layer is to insert $\hat{S}$ number of zeros between each row and column of the input and then pad the input with $\hat{P}$ number of zeros,

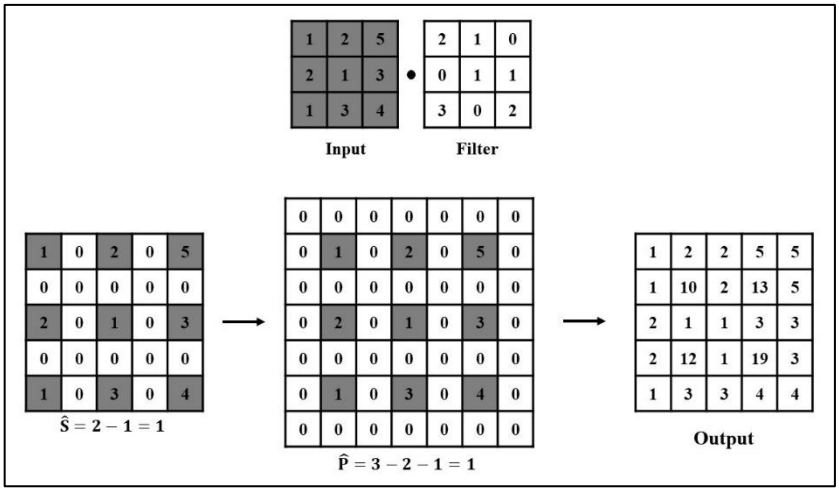
where $$\hat{S} = S - 1$$

$$\hat{P} = K - P - 1$$

After that, as in the convolutional layer, each value of the output is generated by sliding the $K \times K$ filter across the modified input and calculating the scalar product between the filter and the input. Figure 3.3.5 shows an example of a transposed convolutional layer with 3 × 3 input, 3 × 3 filter, stride 2 and padding 1.

Figure 3.3.5

An example of a transposed convolutional layer.



3.4 Training CNNs

Training a CNN involves optimizing its parameters to learn the underlying patterns and features within the input data. The steps of training a CNN are explained as follows [51]:

Preprocessing of the training data: The first step in training the network is collecting labeled data and performing several preprocessing techniques to reduce any type of noise that could affect the accuracy of the training. Sometimes dealing with small and insufficient training dataset, especially medical images, can cause overfitting as we

mentioned earlier. To increase the size of the training dataset, several techniques, called data augmentation, can be used to generate new datasets from the original set. These techniques, including rotation, scaling, and flipping, might be used if needed in case the number of input images is not sufficient.

Architecture design and initialization: Designing the CNN architecture means deciding the number and type of layers that will be used, and this depends on the type of the task, the size of the input data, and the available computational resources. After designing the network, the parameters are initialized either by using an initialization technique, such as random initialization, or using pre-trained weights from a different task or a pre-trained model.

Forward propagation: In this step, the CNN receives the input data and extracts the features from it by using the network's parameters (filters, weights, and biases) to perform convolutional operations, activation functions (e.g., ReLU), and pooling operations (e.g., max pooling). The network here processes the information from the input layer towards the output layer, and hence the name forward propagation.

Loss calculation and backpropagation: At the end of the network, a loss function is used to calculate the difference between the predicted output and the actual label. After that, during the backpropagation, the gradient of the loss function is calculated with respect to the network's parameters and used to adjust the parameters in a way that minimizes the loss function. The steps forward propagation, loss calculation, and backpropagation are repeated for multiple iterations or epochs until the network converges or reaches a predefined stopping criterion (e.g., a maximum number of epochs). Once the training is done, the model is tested on an independent set of data to evaluate its performance.

Training CNNs can be computationally intensive and requires substantial computational resources, especially for large-scale datasets and complex architectures. However, with the growing availability of GPUs and specialized hardware, training CNNs has become more accessible and efficient.

3.5 Applying CNNs

Once a CNN model has been trained, it can be implemented in several ways. This CNN model can be transformed into a software system or a mobile app that is available for anyone to use. New unseen data can be passed through the trained model to obtain real-time predictions. Moreover, If the trained CNN model has learned useful features from a large dataset, it can be used in transfer learning, which is the process of using a pre-trained model on the new task or dataset. This method helps to reduce the training time and is beneficial when dealing with limited labeled training datasets. Additionally, multiple trained models can be combined to enhance the system's performance and improve prediction accuracy.

To ensure optimal performance when using a trained model, it is important to apply the same preprocessing techniques that were used in training that model, such as normalization or data augmentation, on the new data.

3.6 A particular type of CNNs: U-Net

U-Net is a convolutional neural network architecture widely used for image segmentation tasks. Olaf Ronneberger, Philipp Fischer, and Thomas Brox introduced it in their paper in 2015 and used it for biomedical image segmentation [55]. The structure of a U-Net is symmetric and consists of two paths: the contracting path (encoder) and the expanding path (decoder) with skip connections, as illustrated in Figure A.3.6.1.

U-Net is widely used in segmentation tasks, and the structure used for this study will be discussed in detail in chapter 4.

Both CNNs and U-Net architectures serve as prominent tools in image processing, particularly in medical image analysis. CNNs excel in tasks such as image classification and object detection, using convolutional layers and pooling layers to extract features and establish hierarchical representations. On the other hand, U-Net represents a specialized CNN architecture designed explicitly for semantic segmentation tasks, where the precise classification of pixels is essential. The U-shaped design and skip connections aid in retaining spatial information, which is crucial for tasks like medical image segmentation that require detailed structure localization.

The decision to use U-Net over other ANN architectures for OD segmentation emerges from several reasons. Firstly, U-Net's design enables it to accurately localize complex

structures like OD within medical images. Secondly, the architecture's skip connections play a crucial role in maintaining spatial information throughout the segmentation process, ensuring precise localization despite noise or image quality variations. Moreover, U-Net's ability to learn from limited annotated data sets indeed makes it valuable in medical imaging tasks, where gathering extensive labeled data makes it challenging.

It is worth noting that the U-Net's user-friendly implementation and training processes make it more useful, facilitating accessibility for researchers in medical image analysis projects. By modifying the U-Net architecture for OD segmentation, researchers can use an effective approach to identify and locate the OD in retinal images, a critical step in the diagnosis and monitoring of eye diseases.

3.7 Future perspective of CNNs

The future of CNNs looks promising and exciting. CNNs are set to continue revolutionizing the field of computer vision, enabling machines to understand and interpret visual data with greater accuracy and efficiency. As technology advances, we can expect the development of more sophisticated CNN architectures that can handle complex tasks and datasets. Additionally, CNNs will likely become more accessible and applicable to various domains beyond image recognition, such as natural language processing and audio analysis. The future holds potential for CNNs to contribute to advancements in healthcare, autonomous vehicles, robotics, and other fields, making our lives easier and more efficient. With ongoing research and innovation, we can anticipate remarkable improvements in the performance, speed, and versatility of CNNs, opening new possibilities for intelligent systems and enhancing our understanding of the world around us.

Chapter Four

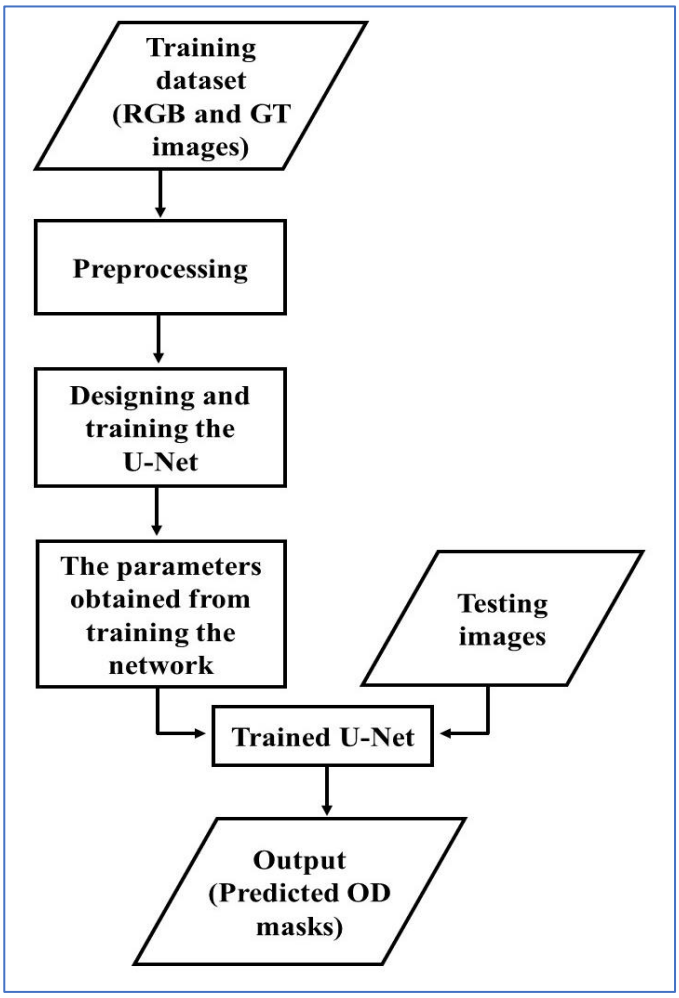
Methodology

4.1 Introduction

In this chapter, the proposed automatic OD segmentation method is introduced. Starting with the preprocessing stage, the input images were cropped and resized to make the computations more precise and faster than dealing with the whole image. After that, contrast adjustment was applied to the images to improve their quality and reduce the noise. The next step was designing the U-Net architecture that was used for OD segmentation. Finally, the performance of the proposed method was assessed, and the predicted results were compared with the GT images. A block diagram of the proposed method is shown in Figure 4.1.1.

Figure 4.1.1

A block diagram for the proposed method



ORIGA (Online Retinal fundus Image database for Glaucoma Analysis and research) dataset has been used to train and test the network. The dataset is open for online access and contains 650 RGB fundus images (482 healthy and 168 glaucoma) with the extension (jpg) and their corresponding GTs with the extension (mat), where OD pixels are represented as 1 and background pixels as 0. The resolution is 3072 × 2048 pixels. The data was collected by the Singapore Malay Eye Study (SiMES) that tested 3,280 Malay adults between the ages 40 and 80 [19].

The GTs were generated using a tool called ORIGA-GT, which was developed to help create accurate and reliable annotations for retinal images. It improves the accuracy of measurements and reduces the time and effort required for grading and segmentation tasks. The tool has a user-friendly interface and is used for image segmentation and grading. It assists in the diagnosis of glaucoma by measuring the CDR of the optic nerve. The tool provides features such as automatic detection of the ROI, which focuses on the OD area. It uses a technique called "fringe removal" to address brightness imbalances caused by unwanted reflections. The tool also employs "key nodes," which are specific landmarks on the disc and cup boundaries, to aid in drawing accurate boundaries. These key nodes are based on clinical or imaging features, such as blood vessels crossing [19]. An example of an RGB fundus image from ORIGA dataset and its corresponding ground truth is shown in Figure A.4.1.2.

After preprocessing, the data was divided randomly into 500 images for training, 100 images for validation, and 50 images for testing.

4.2 Preprocessing

Retinal images usually contain noise, which affects the accuracy of the segmentation process. Moreover, variations in illumination of retinal images can make it difficult to differentiate between the OD from the background and other retinal structures, particularly on the border of the OD. Therefore, to avoid these issues, it is essential to process and prepare the input images before the segmentation to improve the accuracy of the process and get more reliable detection and segmentation of the OD region.

First, in MATLAB, the bounding box technique is applied to extract ROIs from the original fundus images. This technique plays a major role in object detection and localization tasks, and it is based on drawing a rectangular box around a specific object

in the image, giving information about its position and size. It aims to identify multiple objects in the image and classify them into different categories. Object detection models, such as You Only Look Once (YOLO), Single Shot MultiBox Detector (SSD), or Faster R-CNN, are trained using a large dataset of images with bounding boxes manually drawn around the target objects. The model learns to identify each object and predict the coordinates of the bounding box of them during training. The predicted bounding box can be represented by the x and y coordinates of both the top-left corner and the bottom-right corner, or the x and y coordinates of the top-left corner, the height, and the width of the bounding box.

Here, ROI was extracted from the fundus images by applying the bounding box technique to the GT images, which were changed to binary so that they can be processed, to detect the area around the OD and obtain the coordinates of the bounding box (the x and y coordinates of the top-left corner of the box, as well as its width and height). The fundus images were cropped using these coordinates to get the required ROI. The bounding box method can be seen in Figure A.4.2.1.

After extracting a ROI, the input images were resized to 128 × 128 pixels, converted from RGB to LAB color space, and enhanced using CLAHE. This technique enhances the contrast of retinal images, which is crucial for improving the visibility of subtle details and boundaries of the OD and effectively highlighting important features in images with varying lighting conditions or uneven contrast levels. By employing CLAHE before segmentation, the model benefits from clearer input images, potentially leading to more accurate and reliable segmentation results. However, CLAHE has its limitations; One notable disadvantage is the potential amplification of noise in regions with excessive local contrast adjustments. Additionally, the computational overhead of CLAHE can be significant, especially when processing large retinal image datasets, which may impact real-time applications or require optimization for efficient implementation.

LAB color space is widely applied in image segmentation, color-based object detection, and color correction. As shown in Figure A.4.2.2, LAB color space consists of three components: L for lightness, A for green-red color opponent channel, and B for blue-yellow color opponent channel. The L component represents the lightness or brightness of a color and ranges from 0 to 100. A value of 0 represents black, while a value of 100 represents white. It is often used independently for image enhancement tasks. The A

channel represents the green-red opponent axis, where negative values indicate greenish colors and positive values indicate reddish colors. The B channel represents the blue-yellow opponent axis, where negative values correspond to bluish colors and positive values correspond to yellowish colors.

Converting the images to LAB color space can be more beneficial for image enhancement since the LAB color space separates the lightness channel (L) from the color channels (A and B). This helps manipulate the brightness components of the image using the lightness channel without changing the color channels. CLAHE divides the image into small non-overlapping blocks of equal size and separately applies the histogram equalization (HE) to each block, which means redistributing the pixel intensities to cover the entire range of values. However, to prevent noise amplification, the distribution of pixel intensities is limited to a specific range, reducing any value that exceeds this range. After that, all enhanced blocks are combined to form the final enhanced image [56]. After converting the input images into LAB color space, CLAHE is applied to the L channel only, and then, the images are converted back to RGB color space.

An example of an enhanced input image is shown in Figure A.4.2.3.

4.3 OD segmentation using U-Net

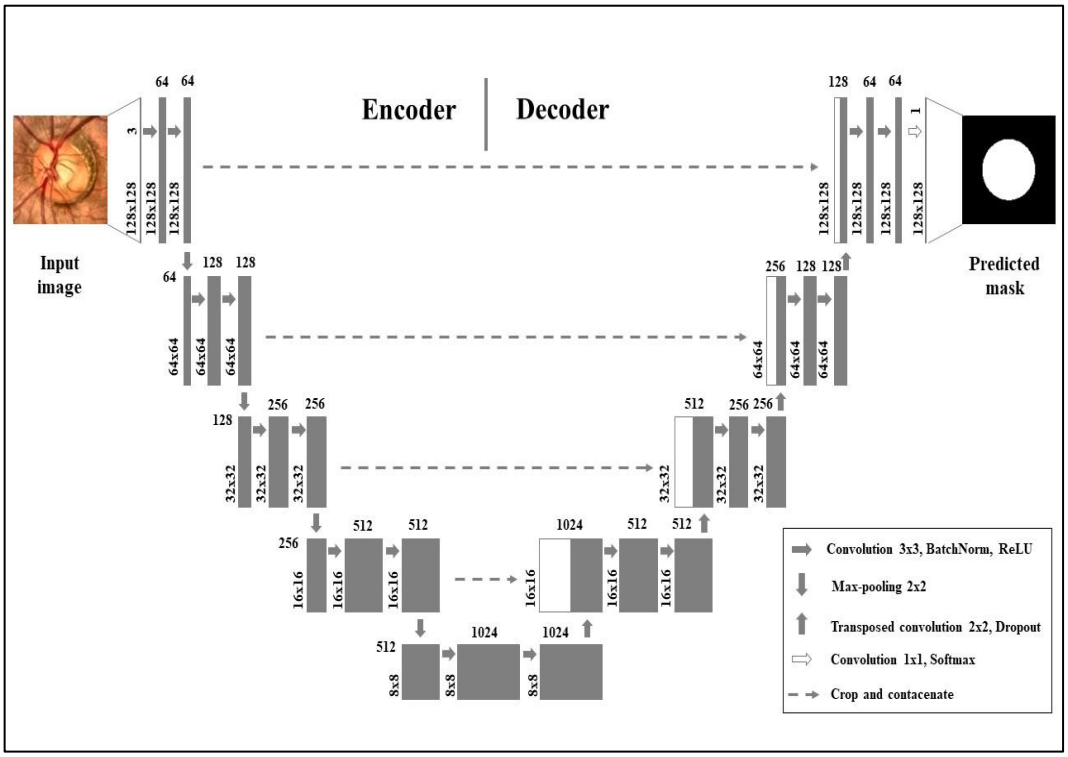
The U-Net architecture used for the OD segmentation is illustrated in Figure 4.3.1.

The network consists of 9 blocks in total: 4 blocks in the encoding path, a bridge block, and 4 blocks in the decoding path. Each block consists of two 3 × 3 convolutional layers, both followed by a batch normalization layer and a ReLU activation function. The encoder path is a typical CNN that extracts the features from the input image reducing its spatial dimensions and resolution.

Each block in the encoding path is followed by a 2 × 2 pooling layer with stride 2, and by the end of this path, the input size is reduced from 128 × 128 to 8 × 8 pixels. The number of filters used in this path gradually increases from 64 to 512. The bridge block is the block located at the bottom of the U-shape of the network connecting the two paths together, and the number of filters used in this block is 1024.

Figure 4.3.1

The proposed architecture of the U-Net



Meanwhile, the decoder path expands the encoded data to the original input size producing a pixel-wise segmentation map. In the decoding path, a 2 × 2 transposed convolutional layer and a dropout layer with probability 0.5 are applied before each block increasing the input size from 8 × 8 to 128 × 128 pixels.

The number of filters used in this path gradually decreases from 512 to 64. Finally, a 1 × 1 convolutional layer, followed by a softmax function is applied to generate the predicted binary masks.

Skip connections connect the corresponding layers in the encoding and decoding paths allowing the information to flow from the encoding to the decoding path. This helps recover the spatial details lost during the pooling process and localize the objects in the segmentation map accurately [57].

The variables that determine the behavior and performance of a neural network model during training are called hyperparameters. Unlike model parameters, which are learned from the data (such as weights and biases), hyperparameters are set before the training process begins and typically remain constant during training. Examples of

hyperparameters include learning rate, batch size, number of layers, activation functions, optimizer choice, number of epochs, loss function, etc. Initially, the hyperparameters used to train the network are defined. After that, the network model is created, complied with the optimizer and the loss function, and trained on the dataset. Finally, the model performance is monitored and the hyperparameters are adjusted as needed to improve segmentation accuracy and model generalization.

The proposed model was trained using the Adaptive Moment Estimation (Adam) optimizer and the binary cross-entropy loss function with 100 epochs, a learning rate of 0.001, and mini-batch of size 20. The optimizer and the loss function will be explained next.

Training the network means modifying its parameters to find the optimal values that gives better results and minimize the error. An optimizer is used to determine how are these parameters adjusted. Commonly used optimizers in U-Net include [58]:

Stochastic Gradient Descent (SGD) iteratively calculates the gradients of the loss function and adjusts the weights in the direction that leads to the most significant reduction in the loss function. However, instead of using the entire training dataset to calculate the gradients of the loss function like the traditional Gradient Descent (GD) method, SGD randomly selects small training batches reducing the calculations in each iteration.

Adaptive Gradient (AdaGrad) adjusts a separate learning rate for each parameter instead of using a fixed learning rate for all parameters like SGD; because not all parameters need the same number of updates to reach the optimal value. For each parameter, the AdaGrad algorithm computes the gradient of the loss function at the current iteration and adds the squared gradient to the sum of all squared gradients from previous iterations. Then, it calculates the adaptive learning rate by dividing the initial learning rate by the square root of the sum of squared gradients with a small constant added to it to avoid division by zero. Finally, to update the parameter, it subtracts the product of the adaptive learning rate and the gradient from the parameter's previous value. Equation 4.3.1 shows the steps of updating the parameter w at the iteration t using the AdaGrad algorithm.

$$w_t = w_{t-1} - \frac{\eta}{\sqrt{\alpha_t + \epsilon}} * \nabla L_t \qquad 4.3.1$$

where α_t can be calculated by:

$$\alpha_t = \sum_{i=1}^{t} (\nabla L_i)^2$$

where w_t and w_{t-1} are the new and previous values of the parameter respectively, η is the initial learning rate, α_t is the sum of all squared gradients from previous iterations, ∇L_t is the gradient of the loss function, and ϵ is a small constant added to avoid division by zero.

Root Mean Square Propagation (RMSProp), similar to AdaGrad, assigns a separate learning rate for each parameter. However, in AdaGrad, the learning rate excessively decreases as the denominator in Equation 4.3.1 becomes bigger with every iteration affecting the process of updating the parameter. To avoid this, RMSProp algorithm decreases this denominator and prevent it from increasing rapidly. For each parameter, the RMSProp algorithm computes the squared gradient of the loss function at the current iteration. Then, it calculates the running average of the squared gradients (also known as the exponentially weighted average of the squared gradients) using the equation:

$$v_t = \beta * v_{t-1} + (1 - \beta) * (\nabla L_t)^2 \qquad 4.3.2$$

where v_t and v_{t-1} are the running averages of the squared gradients of the current and previous iterations respectively, and β is a decay rate between 0 and 1. Higher value of β means more weight to the recent gradients and less weight to the previous ones, and it is often set to 0.9. After that, the learning rate is divided by the square root of the running average of the squared gradients with a small constant added to it to avoid division by zero, and the parameter is adjusted by subtracting the product of the new learning rate and the gradient from the parameter's previous value, as shown in Equation 4.3.3.

$$w_t = w_{t-1} - \frac{\eta}{\sqrt{v_t + \epsilon}} * \nabla L_t \qquad 4.3.3$$

Adaptive Moment Estimation (Adam) combines the benefits of both AdaGrad and RMSProp. In addition to calculating the running average of the squared gradients, the Adam algorithm also calculates the running average of the gradients as follows:

$$m_t = \beta_1 * m_{t-1} + (1 - \beta_1) * \nabla L_t \qquad 4.3.4$$

$$v_t = \beta_2 * v_{t-1} + (1 - \beta_2) * (\nabla L_t)^2$$

where m_t and m_{t-1} are the running averages of the gradients of the current and previous iterations respectively, β_1 and β_2 are the decay rates.

The running average of the squared gradients provides information about the magnitudes of the gradients for each parameter. Parameters with large gradients will have lower effective learning rates, while parameters with small and stable gradients will have higher effective learning rates. This helps the optimizer converge efficiently. Meanwhile, the running average of the gradients helps accelerate the convergence of the optimization process and improve its stability.

At the beginning of training, the running averages of gradients and squared gradients are initialized to zero. This can cause a bias towards the zero, especially in the initial steps. To avoid that, each running average (m_t and v_t) is divided by a correction factor that ensures that the running averages are unbiased and more accurate. Mathematically:

$$\hat{m}_t = \frac{m_t}{1 - {\beta_1}^t} \tag{4.3.5}$$

$$\hat{v}_t = \frac{v_t}{1 - {\beta_2}^t}$$

where $\hat{m}_t$ and $\hat{v}_t$ are the modified running averages of the gradients and squared gradients respectively.

Finally, the learning rate is divided by the square root of the modified running average of the squared gradients with a small constant added to it to avoid division by zero, and the parameter is adjusted by subtracting the product of the new learning rate and the modified running average of the gradients from the parameter's previous value, as shown in Equation 4.3.6.

$$w_t = w_{t-1} - \frac{\eta}{\sqrt{\hat{v}_t + \epsilon}} * \hat{m}_t \tag{4.3.6}$$

Calculating the running averages of both the gradients and the square gradients and combining them makes Adam optimizer an effective choice for updating and optimizing the network's parameters.

During the training process, a loss function is used to assess the model's performance by measuring the difference between the predicted results and the actual values. Lower values of the loss function mean better performance, and therefore, the goal is to minimize the loss function to get more accurate results. Common loss functions for classification tasks include softmax cross-entropy or binary cross-entropy, depending on the number of classes. In loss functions, entropy refers to a measure of uncertainty or disorder in a probability distribution. It quantifies the amount of information needed to describe or predict an outcome.

The binary cross-entropy function (also known as binary log loss or logistic loss) is the one we used in this segmentation task. It measures the difference between the predicted binary masks and the actual ground truth to update the weights through backpropagation, improving the network's performance. The binary cross-entropy loss is calculated for each pixel in the predicted mask and averaged over the entire image [51], which is represented mathematically as:

$$Loss = -\frac{1}{N}\sum_{i=1}^{N}(y_i * \log(p_i) + (1 - y_i) * \log(1 - p_i)) \quad 4.3.7$$

where N is the number of pixels in the image, y_i is the actual binary label for the pixel i (1 for OD and 0 for background), and p_i is the predicted probability of this pixel belonging to the OD region, which can be obtained using the softmax activation function.

Overall, after segmenting the retinal images and producing the predicted masks, binary cross-entropy function is used to calculate the difference between these masks and the original GT. Then the optimizer Adam updates the model's parameters based on the values obtained from the loss function to improve the model's performance.

4.4 Evaluation metrics

The performance of this model is assessed using evaluation methods including accuracy, precision, sensitivity, F-score, specificity, and IoU. These methods are obtained from the confusion matrix that shows the relation between the original GT and the predicted masks as shown in Figure A.4.4.1. True positive (TP), true negative (TN), false positive (FP), and false negative (FN) are used to calculate the evaluation methods, and they are defined as follows:

TP refers to the pixels that are OD and are correctly predicted as OD by the model.

TN refers to the pixels that are background and are correctly predicted as background by the model.

FP refers to the pixels that are background but are incorrectly predicted as OD by the model. FP is also known as Type I error.

FN refers to the pixels that are OD but are incorrectly predicted as background by the model. FN is also known as Type II error.

Each evaluation metric is explained in the context of OD segmentation task as follows:

Accuracy measures how precise the predicted mask is compared to the GT. It is calculated by:

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \quad 4.4.1$$

Precision measures the proportion of correctly predicted OD pixels among all the pixels predicted as OD. It indicates how much the model can avoid false positives, and how precise the model is in identifying OD pixels. It is calculated by:

$$Precision = \frac{TP}{TP + FP} \quad 4.4.2$$

Sensitivity (also known as recall or true positive rate) measures the proportion of correctly predicted OD pixels among all the actual OD pixels in the GT. It indicates how much the model can identify as many OD pixels as possible. It is calculated by:

$$Sensitivity = \frac{TP}{TP + FN} \quad 4.4.3$$

F-score (also known as the dice coefficient) measures the similarity between the predicted mask and the GT. It combines precision and sensitivity into a single metric, and it ranges from 0 to 1, where 1 indicates perfect precision and sensitivity. It is calculated by:

$$F - score = 2 \times \frac{Precision \times Sensitivity}{Precision + Sensitivity} = \frac{2 \times TP}{2 \times TP + FP + FN} \quad 4.4.4$$

Specificity (also known as true negative rate) measures the proportion of actual background pixels that are correctly predicted as background by the model. It is used in tasks that aim to minimize false positives, such as fraud detection. It is calculated by:

$$Specificity = \frac{TN}{TN + FP} \quad 4.4.5$$

IoU measures the overlap between the predicted mask and the GT. It calculates the ratio of the intersection to the union of the two masks. IoU ranges from 0 to 1, where 1 indicates perfect overlap between the predicted and ground truth masks. For binary classification, it is calculated by:

$$IoU = \frac{TP}{TP + FP + FN} \quad 4.4.6$$

Chapter Five

Results and Discussion

This study proposed an automatic OD segmentation method in retinal images using a CNN architecture known as U-Net. This model was trained and tested using the ORIGA dataset containing 650 images. It starts with image preprocessing and extracting a ROI using the bounding box technique to detect the area around the OD and crop the input images. Then, these images were resized and enhanced using CLAHE. After that, a U-Net model was constructed and trained using these preprocessed images. Finally, the model was tested using randomly selected 50 images, and the performance was assessed using the evaluation metrics mentioned in section 4.4. Our proposed method competes the similar studies on the same database giving the following results: average accuracy of 98.42%, average precision of 97.46%, and average sensitivity of 95.33%.

5.1 Experimental results

The proposed method was implemented on Intel® Core™ i5-7200 CPU @ 2.50 GHz with 12 GB RAM using MATLAB R2021a, which consumed approximately 14 hours for training the database using 600 images. An example set of the results of 6 testing images is represented in Table 5.1.1 sorted descending according to accuracy, and the images of these results are shown in Figure 5.1.1.

Table 5.1.1

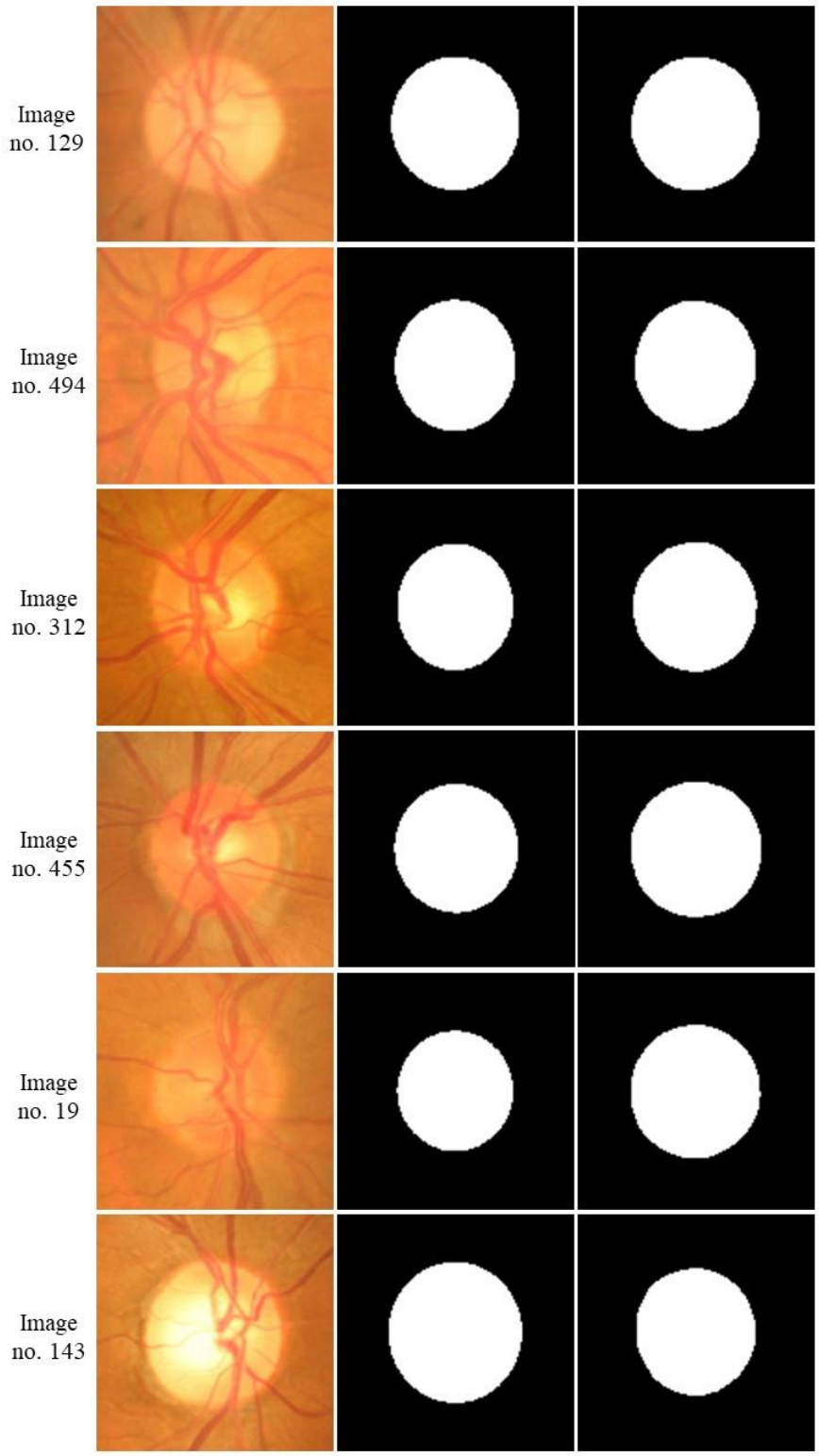
Results of OD segmentation of a sample of 6 testing images.

Testing image no.	Accuracy (%)	Precision (%)	Sensitivity (%)	Specificity (%)	F-score (%)	IoU (%)
129	99.73	99.24	99.62	99.76	99.43	98.86
494	99.70	99.58	99.06	99.88	99.32	98.65
312	98.23	92.04	100.00	97.77	95.86	92.04
455	97.37	89.43	100.00	96.62	94.42	89.43
19	95.25	80.38	100.00	94.10	89.12	80.38
143	94.99	100.00	80.96	100.00	89.48	80.96

This sample was selected to present the two best classified images in the testing set, the two worst images, in addition to two average images.

Figure 5.1.1

Results of OD segmentation of 6 testing images: the fundus images (left), the GT (middle), and the predicted binary masks (right)



Comparison of the OD segmentation results with some new previous studies that used U-Net is shown in Table 5.1.2, while Table 5.1.3 shows the comparison of the OD segmentation results with some previous studies on the ORIGA dataset. The overall performance of this method was calculated by taking the average of all tested images' results.

Table 5.1.2

Comparison of the OD segmentation results of the proposed method with some previous studies that used U-Net

	Accuracy (%)	Precision (%)	Sensitivity (%)	Specificity (%)	F-score (%)	IoU (%)
Chen et al. [2]	99.72	-	98.35	99.75	94.35	89.32
Yu et al. [5]	-	95.36	98.11	-	96.60	-
Hanifa Suwandoko et al. [7]	-	-	-	-	94.5	-
Almustofa et al. [3]	-	-	-	-	Drishti - GS 93.5 REFUGE 95.0	-
Proposed method	98.42	97.46	95.93	99.24	96.54	93.41

Table 5.1.3

Comparison of the OD segmentation results of the proposed method with some previous studies on the ORIGA dataset.

	Accuracy (%)	Precision (%)	Sensitivity (%)	Specificity (%)	F-score (%)	IoU (%)
Wang et al. [9]	-	-	-	-	93.92	88.73
Nazir et al. [10]	97.9	95.5	96.9	-	95.3	98.1
Proposed method	98.42	97.46	95.93	99.24	96.54	93.41

As we can see from Table 5.1.2, many recent papers used the U-Net to segment the OD as a step to use it in medical diagnosis. The results show diversity between the different papers, which can be explained according to the different datasets used, in addition to the different properties implemented inside the U-Net itself, such as the number of filters and number of blocks.

To realize in consistence of the method's performance regarding different datasets, we notice that Almustofa et al. [3] got two different F-score values for two different datasets even though the same U-Net structure was used on both databases. Regardless to the fact that the dataset we used was different from the other studies shown in Table 5.1.2, still the proposed method is competing with these recent works.

To make the comparison more reasonable, we compared our results with research that implemented the same ORIGA dataset (as shown in Table 5.1.3). It is interesting to say that the results of the performance metrics of our method were almost always better than the compared studies who used different techniques for OD segmentation.

While comparing our work with others, it is worth mentioning that they used the whole retinal image for the OD segmentation, which means that the entire background is present. That gives very large TN values, improving the accuracy and the specificity very remarkably. Meanwhile, in our work, we removed most of the background and used only a portion of the image (a ROI), which influences the metrics negatively compared to others work. Nevertheless, the proposed method is competing with the others, and sometimes, gives better results.

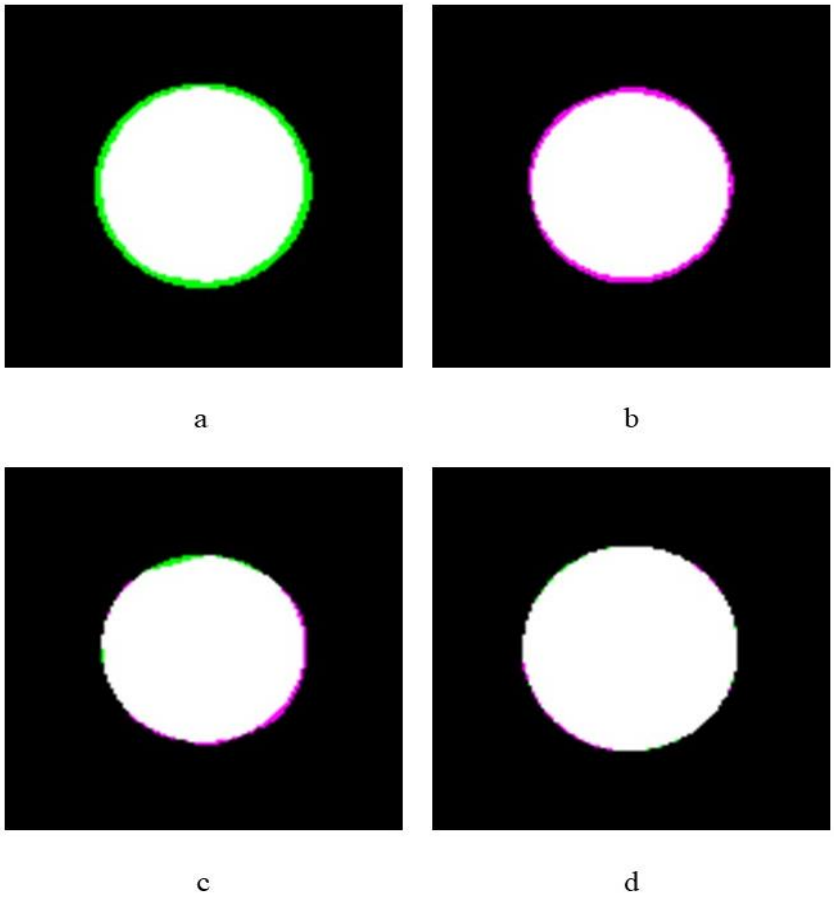
Figure 5.1.2 shows some examples of the overlap between the predicted mask and the ground truth of the OD segmentation results. The green areas represent the FN pixels, while pink areas represent the FP pixels.

5.2 Discussion and future work

OD segmentation task faces many challenges that affect the performance of the model and the accuracy of its results. For instance, the OD shape, size and color can differ among individuals, which makes generating a mask that fits all input data nearly impossible. Moreover, it is difficult sometimes to detect exactly the OD boundary due to several factors including the image quality, lightness, noise, and the presence of the blood vessels or other conditions such as hemorrhage, OD drusen, and exudates. That is why even the manually segmented GT by more than one expert can be slightly different from each other.

Figure 5.1.2

An example of the predicted mask and the GT: (a) Green pixels are FN (b) Pink pixels are FP (c) Green (FN) and pink (FP) pixels are both present (d) Approximately no colored pixels (exact matching)



The rapid development in computers makes it possible to improve AI technology to interfere in many different disciplines positively, and one of these important fields is medical diagnosis, specifically eye diseases. Here, we tried to tackle the detection and segmentation of the OD, which represents an essential key to monitoring a part of eye diseases.

The proposed method is a promising approach as it is a real-time algorithm that can be modified to support physicians in eye treatment. It is suggested to continue this work to make a user-friendly interface that enables doctors to use it easily in diagnosing and aids treating several eye diseases.

The U-Net structure has proven effective in various medical image segmentation tasks. Modifying the U-Net architecture could potentially enhance the accuracy of the segmentation results, by including additional features or layers that capture more details of the OD. Furthermore, to improve the overall performance, it would be beneficial to

consider the use of larger datasets, that include images from different populations, ethnicities, and eye conditions, for training and evaluation. Additionally, datasets with higher resolution and finer annotations could contribute to more precise and detailed segmentation results. Moreover, this work could be expanded beyond OD segmentation to include other retinal structures, such as blood vessels, and fovea. By incorporating multiple segmentation tasks into a single framework, the U-Net model could be trained to segment and analyze these structures simultaneously.

List of Abbreviations

Abbreviation	Meaning
AdaGrad	Adaptive Gradient
Adam	Adaptive Moment Estimation
AI	Artificial Intelligence
AION	Arteritic Ischemic Optic Neuropathy
ANN	Artificial Neural Network
AUC	Area Under the Curve
BAC	Balanced Accuracy
BP	Backpropagation
CCS	Cross-Connection Sub-Network
CDR	Cup-to-Disc Ratio
CLAHE	Contrast-Limited Adaptive Histogram Equalization
CNN	Convolutional Neural Network
DR	Diabetic Retinopathy
DRISHTI-GS	Drishti – Retinal Image Dataset for Lesion Detection
DRIVE	Digital Retinal Images for Vessel Extraction
DSC	Dice Similarity Coefficient
ELU	Exponential Linear Unit
EX	Hard Exudates
FDS	Feature Detection Sub-Network
FFNN	Feed Forward Neural Network
FN	False Negative
FP	False Positive
GD	Gradient Descent
GT	Ground Truth
HE	Hemorrhage
HE	Histogram Equalization
ICP	Intracranial Pressure
IDRiD	Indian Diabetic Retinopathy Image Dataset
ION	Ischemic Optic Neuropathy
IOP	Intra-Ocular Pressure
IoU	Intersection over Union
LReLU	Leaky Rectified Linear Unit
MA	Microaneurysms
MCC	Matthew’s Correlation Coefficient

MLP	Multi-Layer Perceptron
MobileNetSSDv2	MobileNet Single Shot Detector
MSE	Mean Square Error
MSVI	Moderate and Severe Vision Impairment
NAION	Nonarteritic Ischemic Optic Neuropathy
NCC	Normalized Correlation Coefficient
OC	Optic Cup
OD	Optic Disc
ODC	Optic Disc Coloboma
ODD	Optic Disc Drusen
ONH	Optic Nerve Head
OP	Optic Pits
ORIGA	Online Retinal fundus Image database for Glaucoma Analysis and research
PReLU	Parametric Rectified Linear Unit
REFUGE	Retinal Fundus Glaucoma Challenge
ReLU	Rectified Linear Unit
RMSProp	Root Mean Square Propagation
RNN	Recurrent Neural Network
ROI	Region of Interest
RPN	Region Proposal Network
SE	Soft Exudates
SGD	Stochastic Gradient Descent
SiMES	Singapore Malay Eye Study
SLP	Single Layer Perceptron
SSD	Single Shot MultiBox Detector
STARE	Structured Analysis of the Retina
Tanh	Hyperbolic Tangent
TN	True Negative
TP	True Positive
YOLO	You Only Look Once

References

[1] Sudhan MB, Sinthuja M, Pravinth Raja S, Amutharaj J, Charlyn Pushpa Latha G, Sheeba Rachel S, et al. Segmentation and classification of glaucoma using U-Net with deep learning model. J Healthc Eng. 2022 Feb 16;2022.

[2] Chen N, Zhao Y, Li J, Yang D, Zhou S, Xue L. The U-Net via batch norm model for optic disc extraction and segmentation in retinal image. In: ACM International Conference Proceeding Series. Association for Computing Machinery; 2022. p. 511–4.

[3] Almustofa AN, Handayani A, Mengko TLR. Optic disc and optic cup segmentation on retinal image based on multimap localization and U-Net convolutional neural network. Journal of Image and Graphics. 2022 Sep 1;10(3):109–15.

[4] Panahi A, Askari Moghadam R, Tarvirdizadeh B, Madani K. Simplified U-Net as a deep learning intelligent medical assistive tool in glaucoma detection. Evol Intell. 2022 Sep 10;

[5] Yu H, Ying W. Two-stage U-Net for optic disc/cup segmentation. In: 2022 IEEE 2nd International Conference on Data Science and Computer Application, ICDSCA 2022. Institute of Electrical and Electronics Engineers Inc.; 2022. p. 275–8.

[6] Septiarini A, Hamdani H, Setyaningsih E, Junirianto E, Utaminingrum F. Automatic method for optic disc segmentation using deep learning on retinal fundus images. Healthc Inform Res. 2023 Apr 1;29(2):145–51.

[7] Hanifa Suwandoko F, Handayani A, Latifah Erawati Rajab T. An optimized segmentation of optic disc and optic cup in retinal fundus images based on multimap localization and conventional U-Net. In: IEEE Region 10 Annual International Conference, Proceedings/TENCON. Institute of Electrical and Electronics Engineers Inc.; 2023. p. 125–9.

[8] Desiani A, Priyanta S, Ramayanti I, Suprihatin B, Al-Filambany MG, Salamah F. Improved U-Net performance with augmentation for retinal optic segmentation. In: 2023 International Conference on Informatics, Multimedia, Cyber and Informations

System (ICIMCIS). Institute of Electrical and Electronics Engineers (IEEE); 2023. p. 284–9.

[9] Wang L, Gu J, Chen Y, Liang Y, Zhang W, Pu J, et al. Automated segmentation of the optic disc from fundus images using an asymmetric deep learning network. Pattern Recognit. 2021 Apr 1;112.

[10] Nazir T, Irtaza A, Starovoitov V. Optic disc and optic cup segmentation for glaucoma detection from blur retinal Images using improved mask-RCNN. Int J Opt. 2021;2021.

[11] Saine PJ, Tyler ME. Ophthalmic photography: retinal photography, angiography, and electronic imaging. 2nd ed. Butterworth-Heinemann; 2001.

[12] Ivanišević M. First look into the eye. Eur J Ophthalmol. 2019 Nov 1;29(6):685–8.

[13] Patton N, Aslam TM, MacGillivray T, Deary IJ, Dhillon B, Eikelboom RH, et al. Retinal image analysis: concepts, applications and potential. Prog Retin Eye Res. 2006 Jan;25(1):99–127.

[14] Staal J, Abràmoff MD, Niemeijer M, Viergever MA, Van Ginneken B. Ridge-based vessel segmentation in color images of the retina. IEEE Trans Med Imaging. 2004 Apr;23(4):501–9.

[15] Hoover A, Kouznetsova V, Goldbaum M. Locating blood vessels in retinal images by piecewise threshold probing of a matched filter response. IEEE Trans Med Imaging. 2000;19(3):203–10.

[16] Porwal P, Pachade S, Kamble R, Kokare M, Deshmukh G, Sahasrabuddhe V, et al. Indian Diabetic Retinopathy Image Dataset (IDRiD): a database for diabetic retinopathy screening research. 2018; Available from: www.mdpi.com/journal/data

[17] Sivaswamy J, Krishnadas SR, Datt Joshi G, Jain M, Syed Tabish AU. DRISHTI-GS: retinal image dataset for optic nerve head (ONH) segmentation. In: 2014 IEEE 11th International Symposium on Biomedical Imaging (ISBI). Beijing, China; 2014. p. 53–6.

[18] Orlando JI, Fu H, Barbosa Breda J, van Keer K, Bathula DR, Diaz-Pinto A, et al. REFUGE Challenge: A unified framework for evaluating automated methods for glaucoma assessment from fundus photographs. Med Image Anal. 2020 Jan;59.

[19] Zhang Z, Yin FS, Liu J, Wong WK, Tan NM, Lee BH, et al. ORIGA-light : an online retinal fundus image database for glaucoma analysis and research. In: 32nd Annual International Conference of the IEEE EMBS. 2010. p. 3065–8.

[20] Willoughby CE, Ponzin D, Ferrari S, Lobo A, Landau K, Omidi Y. Anatomy and physiology of the human eye: Effects of mucopolysaccharidoses disease on structure and function - a review. Clin Exp Ophthalmol. 2010;38:2–11.

[21] Cholkar K, Dasari SR, Pal D, Mitra AK. Eye: Anatomy, physiology and barriers to drug delivery. In: Ocular Transporters and Receptors: Their Role in Drug Delivery. 1st ed. Woodhead Publishing; 2013. p. 1–36.

[22] Almazroa A, Burman R, Raahemifar K, Lakshminarayanan V. Optic disc and optic cup segmentation methodologies for glaucoma image detection: a survey. J Ophthalmol. 2015;2015.

[23] Rangayyan RM, Zhu X, Ayres FJ, Ells AL. Detection of the optic nerve head in fundus images of the retina with gabor filters and phase portrait analysis. J Digit Imaging. 2010 Aug;23(4):438–53.

[24] Amador-Patarroyo MJ, Pérez-Rueda MA, Tellez CH. Congenital anomalies of the optic nerve. Saudi Journal of Ophthalmology. 2015 Jan 1;29(1):32–8.

[25] Biousse V, Newman NJ. Ischemic optic neuropathies. Campion EW, editor. New England Journal of Medicine [Internet]. 2015 Jun 18;372(25):2428–36. Available from: http://www.nejm.org/doi/10.1056/NEJMra1413352

[26] Rigi M, Almarzouqi SJ, Morgan ML, Lee AG. Papilledema: Epidemiology, etiology, and clinical management. Eye Brain. 2015 Aug 17;7:47–57.

[27] Joshi S, Partibane B, Hatamleh WA, Tarazi H, Yadav CS, Krah D. Glaucoma detection using image processing and supervised learning for classification. J Healthc Eng. 2022;2022.

[28] Steinmetz JD, Bourne RRA, Saylan M, Mersha AM, Weldemariam AH, Wondmeneh TG, et al. Causes of blindness and vision impairment in 2020 and trends over 30 years, and prevalence of avoidable blindness in relation to VISION 2020: The Right to Sight: An analysis for the Global Burden of Disease Study. Lancet Glob Health. 2021 Feb 1;9(2):e144–60.

[29] Tham YC, Li X, Wong TY, Quigley HA, Aung T, Cheng CY. Global prevalence of glaucoma and projections of glaucoma burden through 2040: A systematic review and meta-analysis. Ophthalmology. 2014 Nov 1;121(11):2081–90.

[30] Zheng Y, He M, Congdon N. The worldwide epidemic of diabetic retinopathy. Indian J Ophthalmol. 2012 Sep 1;60(5):428–31.

[31] Jinfeng G, Qummar S, Junming Z, Ruxian Y, Khan FG. Ensemble framework of deep CNNs for diabetic retinopathy detection. Comput Intell Neurosci. 2020;2020.

[32] Moorthy J, Gandhi UD. A survey on medical image segmentation based on deep learning techniques. Big Data and Cognitive Computing. 2022 Dec 1;6(117).

[33] Mcculloch WS, Pitts W. A logical calculus of the ideas immanent in nervous activity. Bulletin of Mathematical Biophysics. 1943;5:115–33.

[34] Lin JW. Artificial neural network related to biological neuron network: a review. Advanced Studies in Medical Sciences. 2017;5:55–62.

[35] Dongare AD, Kharde RR, Kachare AD. Introduction to Artificial Neural Network. International Journal of Engineering and Innovative Technology (IJEIT). 2012;2(1):189–94.

[36] Weiss R, Karimijafarbigloo S, Roggenbuck D, Rödiger S. Applications of neural networks in biomedical data analysis. Biomedicines. 2022 Jul 1;10(7).

[37] Kamruzzaman J, Begg RK, Sarker RA. Artificial neural networks in finance and manufacturing. Idea Group Publishing; 2006.

[38] Basu JK, Bhattacharyya D, Kim T. Use of artificial neural network in pattern recognition. International Journal of Software Engineering and its Applications. 2010;4(2).

[39] van Engelen JE, Hoos HH. A survey on semi-supervised learning. Mach Learn. 2020 Feb 1;109(2):373–440.

[40] Zhou ZH. A brief introduction to weakly supervised learning. Natl Sci Rev. 2018 Jan 1;5(1):44–53.

[41] Hagan MT, Demouth HB, Beale MH, De Jesús O. Neural network design. 2nd Edition. Martin Hagan; 2014.

[42] Li J, Cheng JH, Shi JY, Huang F. Brief introduction of back propagation (BP) neural network algorithm and its improvement. In: Advances in Computer Science and Information Engineering. Berlin, Heidelberg: Springer; 2012. p. 553–8.

[43] Goodfellow I, Bengio Y, Courville A. Deep Learning. Cambridge, MA: The MIT Press; 2016.

[44] Gu J, Wang Z, Kuen J, Ma L, Shahroudy A, Shuai B, et al. Recent advances in convolutional neural networks. Pattern Recognition. 2018;77:354–77.

[45] Albelwi S, Mahmood A. A framework for designing the architectures of deep convolutional neural networks. Entropy. 2017 Jun 1;19(6).

[46] Ahlawat S, Choudhary A, Nayyar A, Singh S, Yoon B. Improved handwritten digit recognition using convolutional neural networks (CNN). Sensors (Switzerland). 2020 Jun 2;20(12):1–18.

[47] Qi X, Wu C, Shi Y, Qi H, Duan K, Wang X. A convolutional neural network face recognition method based on BiLSTM and attention mechanism. Comput Intell Neurosci. 2023 Jan 19;2023:1–14.

[48] Agnihotri A, Saraf P, Bapnad KR. A convolutional neural network approach towards self-driving cars. In: 2019 IEEE 16th India Council International Conference, INDICON 2019 - Symposium Proceedings. Institute of Electrical and Electronics Engineers Inc.; 2019.

[49] Yamashita R, Nishio M, Do RKG, Togashi K. Convolutional neural networks: an overview and application in radiology. Insights Imaging. 2018 Aug 1;9(4):611–29.

[50] Fan FL, Xiong J, Li M, Wang G. On interpretability of artificial neural networks: a survey. IEEE Trans Radiat Plasma Med Sci. 2021 Nov 1;5(6):741–60.

[51] Aghdam HH, Heravi EJ. Guide to convolutional neural networks: a practical application to traffic-sign detection and classification. Guide to Convolutional Neural Networks. Springer International Publishing; 2017.

[52] Schilling, F. The effect of batch normalization on deep convolutional neural networks [master's thesis on the Internet]. Stockholm: KTH Royal Institute of Technology; 2016. [cited 2023 Aug 31]. Available from: https://www.diva-portal.org/smash/get/diva2:955562/FULLTEXT01.pdf

[53] Lee S, Lee C. Revisiting spatial dropout for regularizing convolutional neural networks. Multimed Tools Appl. 2020 Dec 1;79(45–46):34195–207.

[54] Yu Y, Zhao T, Wang M, Wang K, He L. Uni-OPU: An FPGA-based uniform accelerator for convolutional and transposed convolutional networks. IEEE Trans Very Large Scale Integr VLSI Syst. 2020 Jul 1;28(7):1545–56.

[55] Ronneberger O, Fischer P, Brox T. U-Net: convolutional networks for biomedical image segmentation. In: Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics). Springer Verlag; 2015. p. 234–41.

[56] Chen RC, Dewi C, Zhuang YC, Chen JK. Contrast limited adaptive histogram equalization for recognizing road marking at night based on yolo models. IEEE Access. 2023;11.

[57] Ibtehaz N, Rahman MS. MultiResUNet: rethinking the U-Net architecture for multimodal biomedical image segmentation. Neural Networks. 2020 Jan 1;121:74–87.

[58] Kartowisastro IH, Latupapua J. A comparison of adaptive moment estimation (Adam) and RMSProp optimisation techniques for wildlife animal classification using convolutional neural networks. Revue d'Intelligence Artificielle. 2023 Aug 1;37(4):1023–30.

Appendix A

Figures

Figure A.1.1.1

Early model of the Helmholtz ophthalmoscope, 1851

Figure A.1.2.1

The human eye anatomy

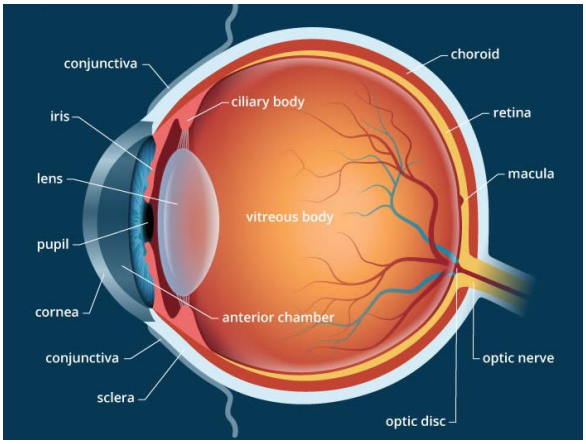


Figure A.1.3.1

The OD shown in several fundus images from the IDRiD dataset

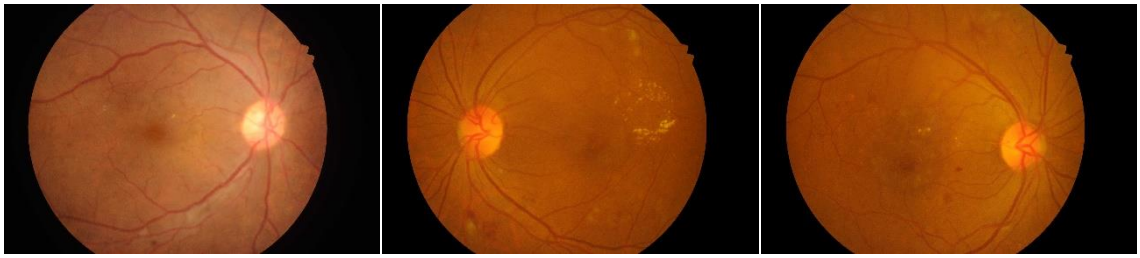


Figure A.1.4.1

Examples of congenital abnormalities (a) OD coloboma (b) Optic pit (c) OD drusen

(a) (b) (c)

Figure A.1.4.2

Examples of acquired abnormalities (a) Arteritic ischemic optic neuropathy (AION) (b) Nonarteritic ischemic optic neuropathy (NAION) (c) Papilledema

(a) (b) (c)

Figure A.1.5.1

The difference in vision between a normal person (left) and one with glaucoma (right)

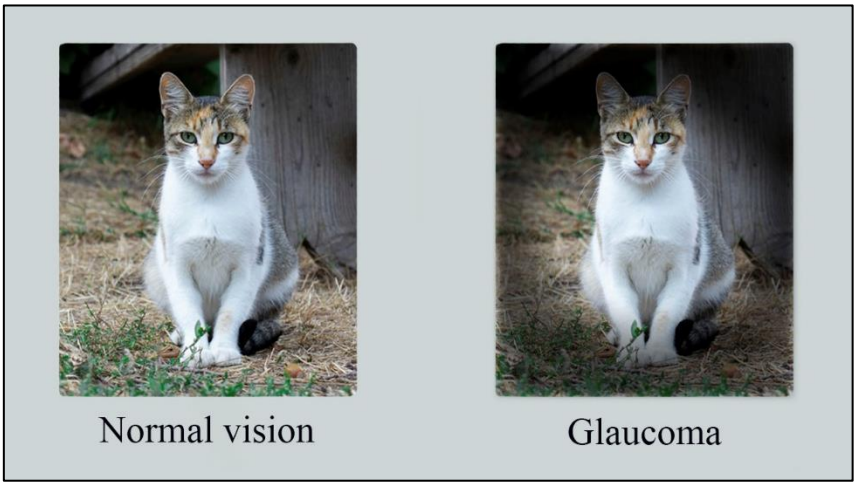
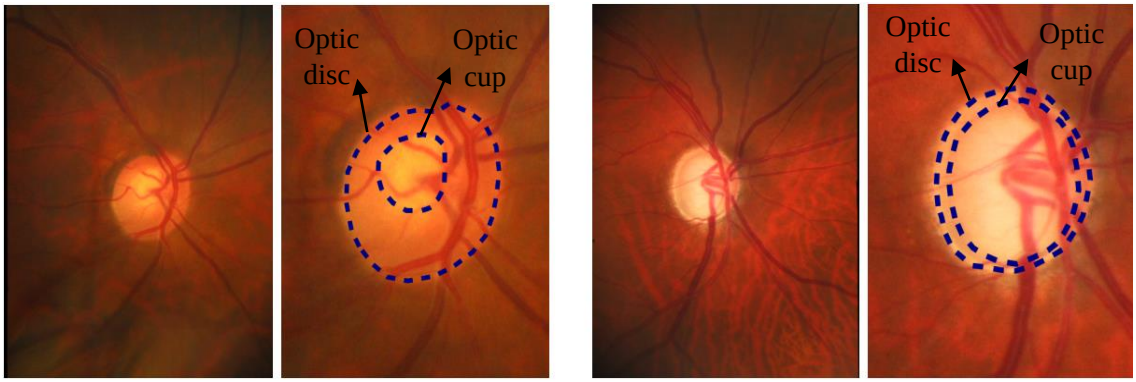


Figure A.1.5.2

Fundus images of (a) Healthy eye (b) Glaucoma-suspicious eye



(a) (b)

Figure A.1.5.3

The difference in vision between a normal person (left) and one with DR (right)

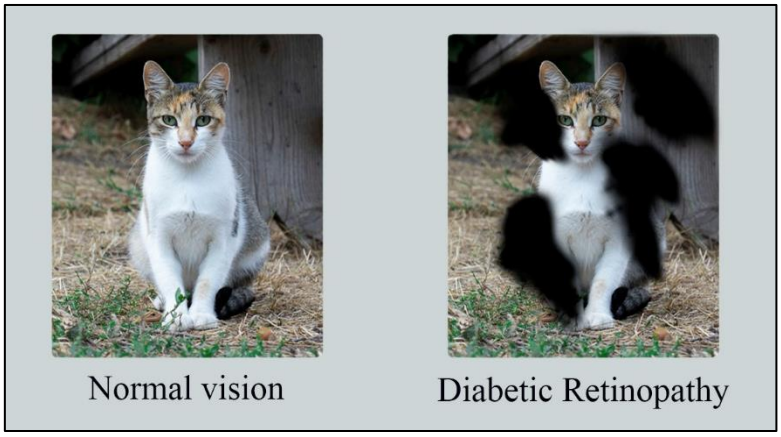


Figure A.2.2.3

Binary classification (left) and multi-class classification (right)

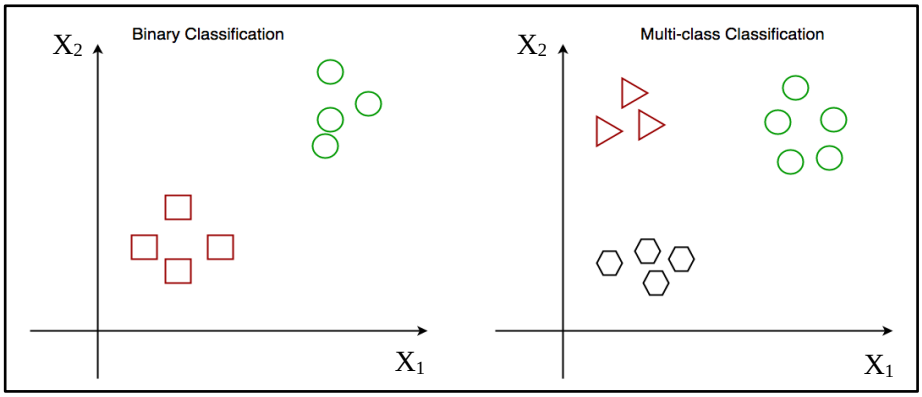


Figure A.2.2.4

The difference between classification (left) and regression (right)

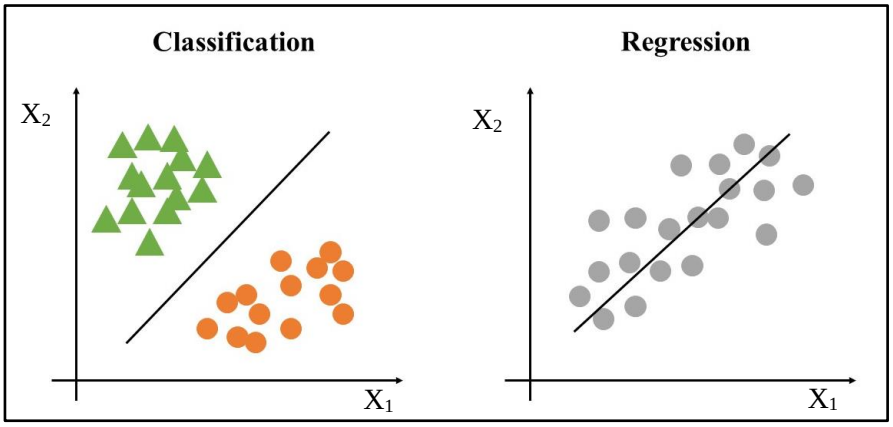


Figure A.2.2.5

The process of unsupervised learning

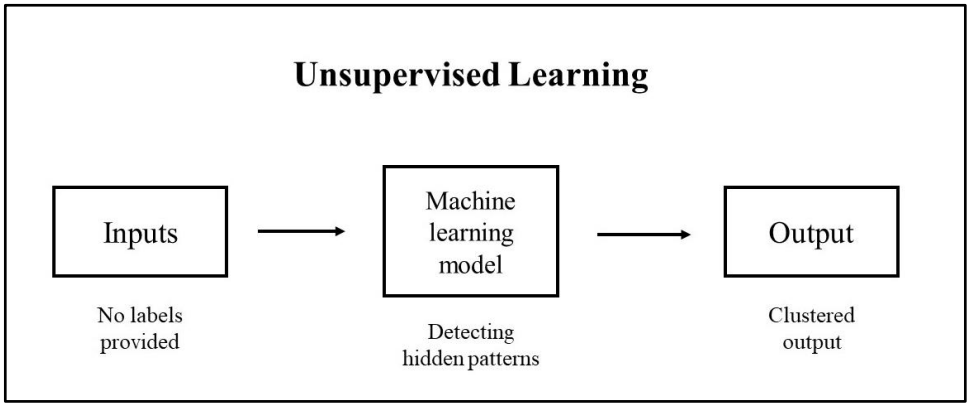


Figure A.2.2.6

Clustering algorithm

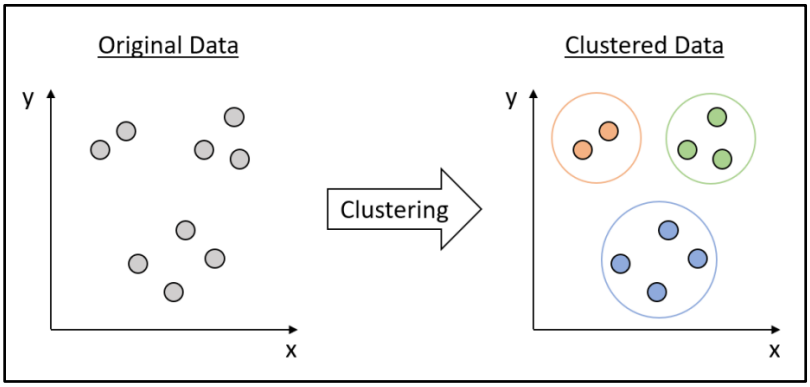


Figure A.2.2.7

Reinforcement learning

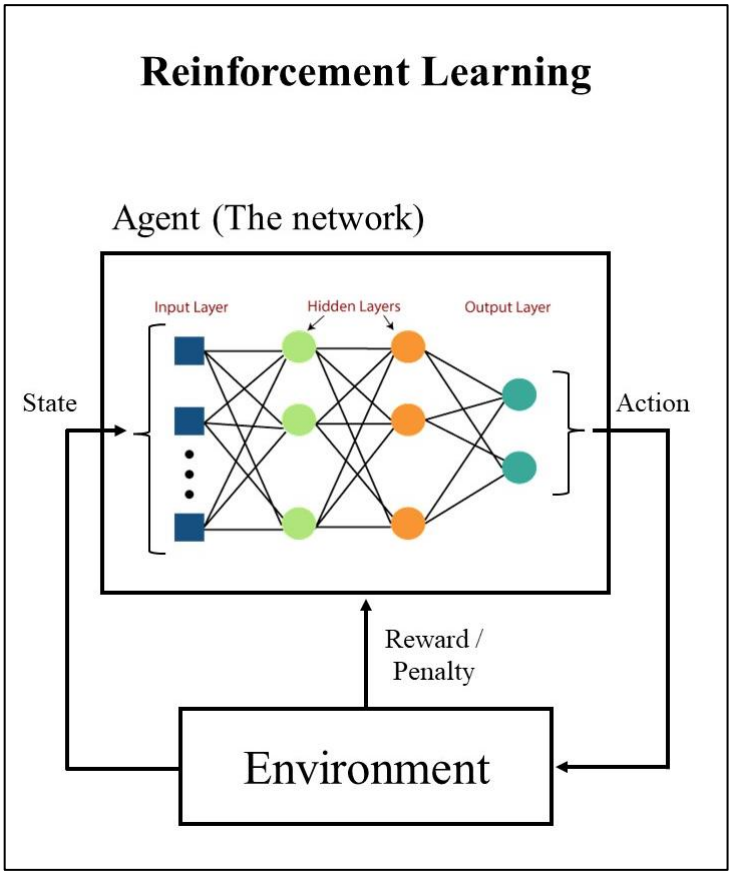


Figure A.2.3.1

The network of a SLP

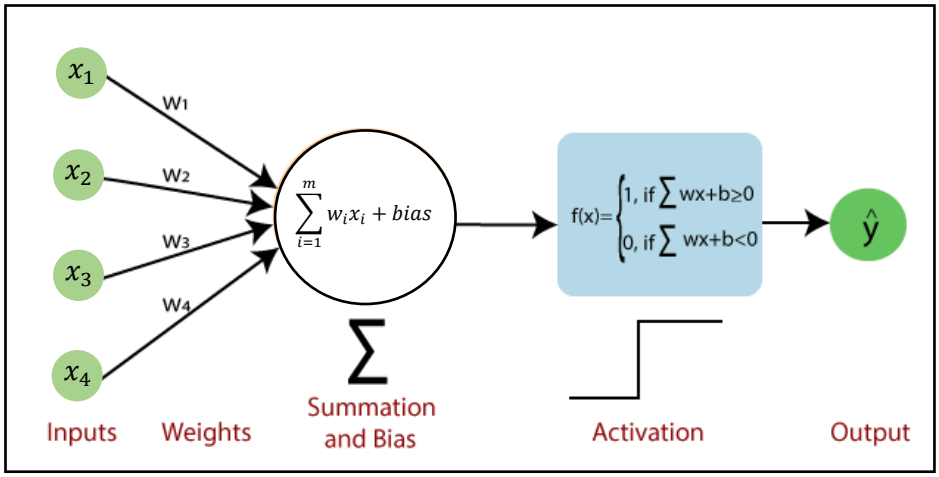


Figure A.2.3.2

Linearly separable data (left) and nonlinearly separable data (right)

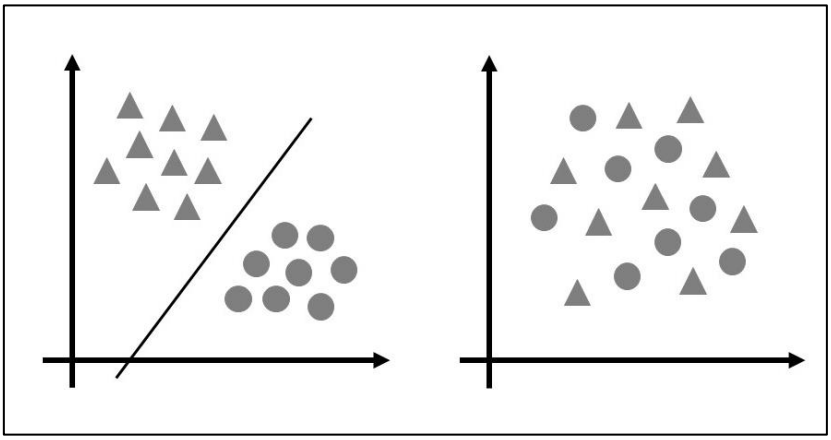


Figure A.2.3.3

The network of a MLP

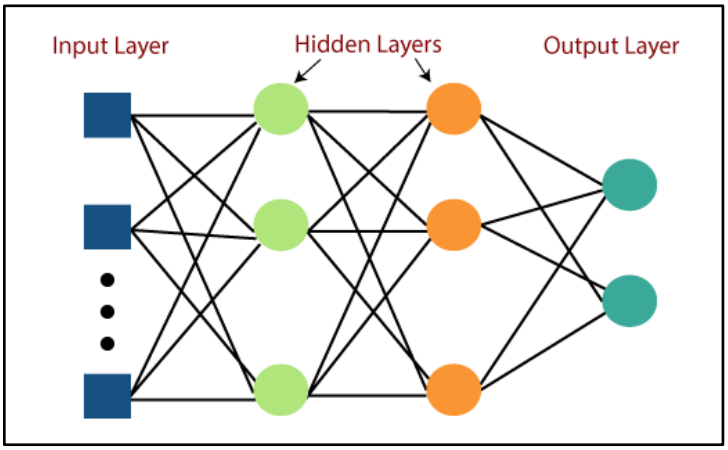


Figure A.2.3.5

Three-layer MLP

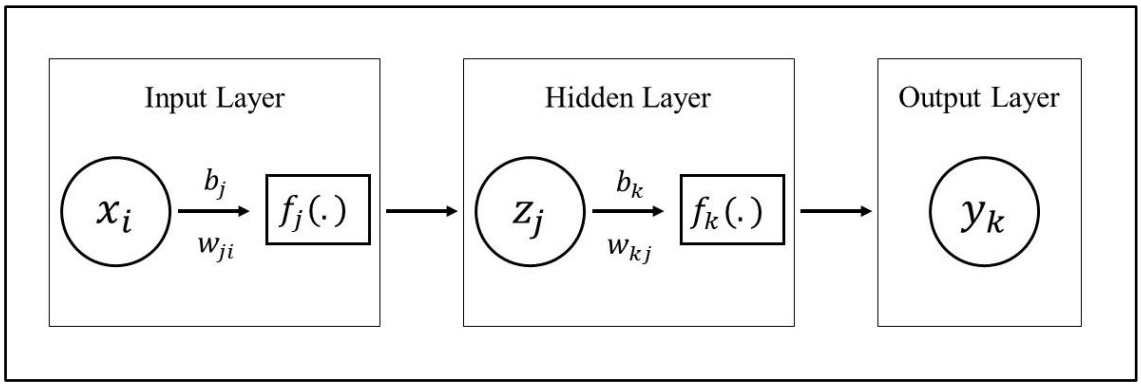


Figure A.3.3.4

A comparison between the outputs of applying a deconvolutional layer and a transposed convolutional layer after a convolutional layer

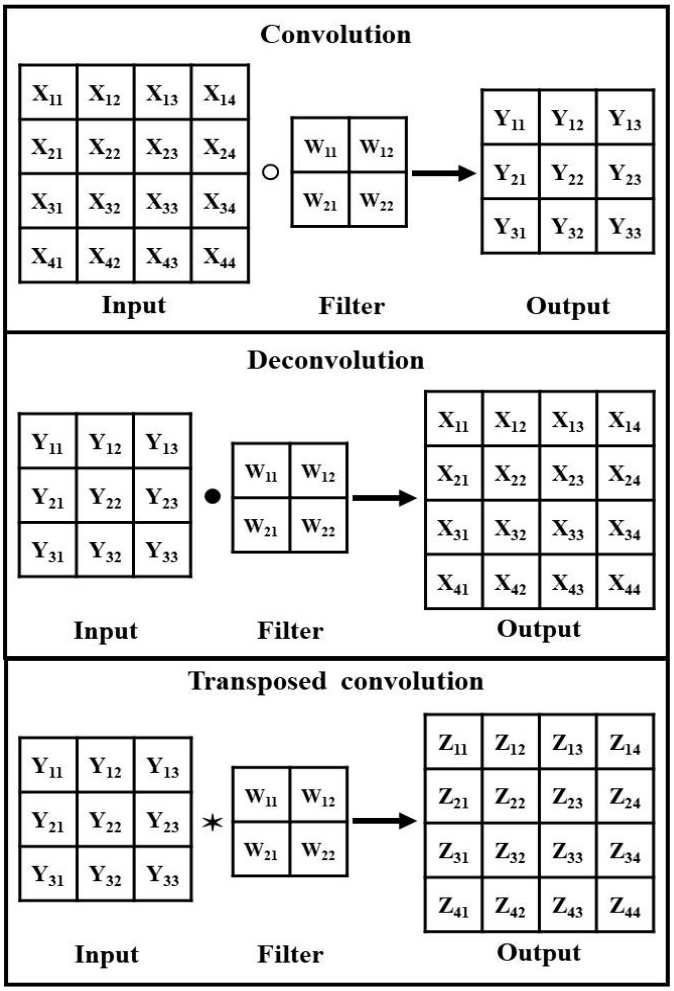


Figure A.3.6.1

A typical U-Net structure

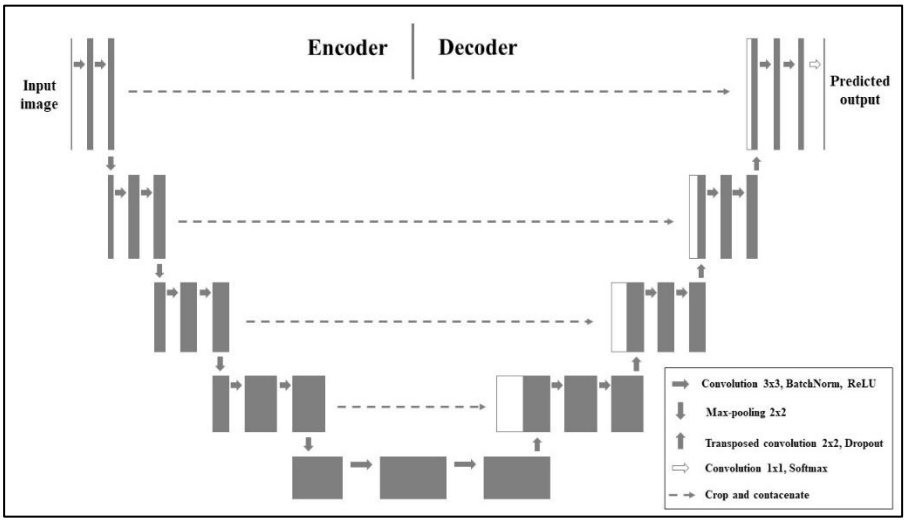


Figure A.4.2.3

A cropped retinal image before and after contrast enhancement (left and right respectively)

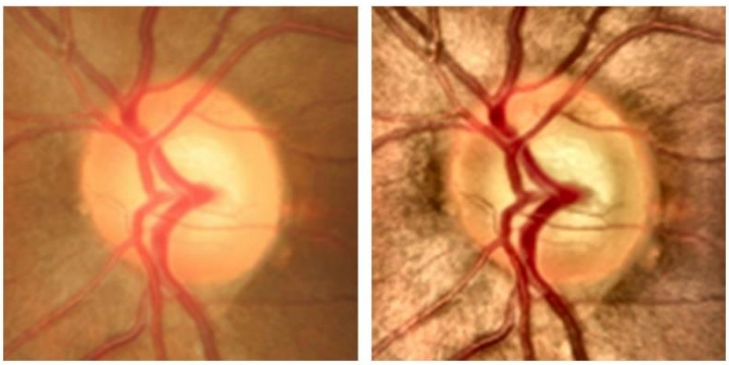


Figure A.4.4.1

The confusion matrix used for evaluating the performance of the model

		Predicted mask: OD	Predicted mask: Background
Ground Truth	OD	TP	FN
Ground Truth	Background	FP	TN

جـامعـــة النجاح الوطنية

كليـــة الدراســـات العليـــا

التجزئة الآلية للقرص البصري في صور العين بالاعتماد على الشبكات العصبية الاصطناعية: شبكة U

إعداد

نور جمال الهندي

إشراف

د. هادي حمد

قدمت هذه الرسالة استكمالا لمتطلبات الحصول على درجة الماجستير في الرياضيات المحوسبة، من كلية الدراسات العليا، في جامعة النجاح الوطنية، نابلس– فلسطين.

2024

التجزئة الآلية للقرص البصري في صور العين بالاعتماد على الشبكات العصبية الاصطناعية: شبكة U

إعداد

نور جمال الهندي

إشراف

د. هادي حمد

الملخص

قرص العين يقع في الجهة الخلفية من العين وهو أحد الأجزاء المهمة في شبكية العين، ويمثل نقطة عبور السيالات العصبية والأوعية الدموية لعصب العين. تمنح التجزئة الدقيقة لقرص العين معلومات مهمة حول بنية شبكية العين وحالتها الصحية وتساعد هذه المعلومات في تشخيص ومعالجة عددٍ من أمراض العين مثل الزَرَق واعتلال الشبكية بمرض السكري وتشوّهات عصب العين. ومع توفّر التجزئة التلقائية لقرص العين تستطيع الأنظمة المحوسبة تحليل عدد كبير من صور الشبكية متيحةً الفرصة لتحديد ومراقبة أمراض العين. لم تقتصر هذه الأتمتة على تحسين سرعة ودقة التشخيص وإنما سهّلت الوصول إلى الرعاية الصحية وجعلتها فعّالة من حيث التكلفة، وخاصّةً في المناطق ذات الإمكانات المحدودة في طب العيون.

في هذه الدراسة قمنا بتقديم طريقة تلقائية لتجزئة قرص العين باستخدام نوع من أنواع الشبكات لعصبية الالتفافية (CNN) تسمى U-Net. بدايةً تم استخراج المنطقة المطلوبة وقصّها من صور الشبكية باستخدام تقنية المربع المحيط، ولجعل العمليات الحسابية أسرع تم تغيير حجم الصور المقتطعة إلى 128 × 128 بكسل. بعدها تم معالجة الصور باستخدام معادلة الرسم البياني التكيفي محدود التباين (CLAHE) للتخلص من التشويش وتحسين جودة الصور، بعد ذلك تمّ إنشاء نموذج U-Net وتدريبه للحصول على الصور المجزئة.

تمّ تدريب النموذج المقترح وتقييمه باستخدام مجموعة البيانات العامة الصورية ORIGA والتي تحتوي على 650 صورة مختلفة لشبكية العين، وتم مقارنة النتائج المتوقعة مع صور التجزئة التي رسمها الأخصائيون. وكانت نتائج النموذج المقترح واعدة ونافست دراسات شبيهة من حيث الطريقة والبيانات، حيث ظهرت النتائج بمعدل دقة 98.42%، ومعدل ضبط 97.46%، ومعدل حساسية 95.33%.

الكلمات المفتاحية: قرص العين، تجزئة قرص العين، الشبكات العصبية الالتفافية (CNN)، U-Net، مجموعة البيانات ORIGA.