August 2021

Medical Image Segmentation Using Machine Learning

Masoud Khani
University of Wisconsin-Milwaukee

MEDICAL IMAGE SEGMENTATION USING MACHINE LEARNING

by

Masoud Khani

A Thesis Submitted in

Partial Fulfillment of the

Requirements for the Degree of

Master of Science

in Computer Science

at

University of Wisconsin-Milwaukee

August 2021

ABSTRACT

MEDICAL IMAGE SEGMENTATION USING MACHINE LEARNING

by

Masoud Khani

The University of Wisconsin-Milwaukee, 2021
Under the Supervision of Professor Zeyun Yu

Image segmentation is the most crucial step in image processing and analysis. It can divide an image into meaningfully descriptive components or pathological structures. The result of the image division helps analyze images and classify objects. Therefore, getting the most accurate segmented image is essential, especially in medical images. Segmentation methods can be divided into three categories: manual, semiautomatic, and automatic. Manual is the most general and straightforward approach. Manual segmentation is not only time-consuming but also is imprecise. However, automatic image segmentation techniques, such as thresholding and edge detection, are not accurate in the presence of artifacts like noise and texture.

This research aims to show how to extract features and use traditional machine learning methods like a random forest to obtain the most accurate regions of interest in CT images. In addition, this study shows how to use a deep learning model to segment the wound area in raw pictures and then analyze the corresponding area in near-infrared images.

This thesis first gives a brief review of current approaches to medical image segmentation and deep learning background. Furthermore, we describe different approaches to build a model for segmenting CT-Scan images and Wound Images. For the results, we achieve 97.4% accuracy in CT-image segmentation and 89.8% F1-Score For wound image segmentation.

TABLE OF CONTENTS

LIST OF FIGURES

ACKNOWLEDGMENTS

I would like to thank my advisor, Professor Zeyun Yu, who always helped me throughout my thesis and my master's studies. His advice helped me to find my interest in health informatics and encouraged me to continue my studies.

I would also like to thank Professor Jake Luo and Professor Jun Zhang for joining my thesis committee and giving me their guidance in their fields of expertise to improve my thesis. In addition, I would like to extend my gratitude to Professor Luo for giving me the opportunity to continue my studies for a Ph.D. in Biomedical and Health Informatics.

Thanks to Professor Priyantha Premnath from BHE at University of Wisconsin-Milwaukee and Dr. Jeff Niezgoda from the AZH center who provided the data for chapters three and four, respectively.

I would like to thank my family. Their support and dedication enabled me to complete my research and finish thesis writing. Finally, I would like to thank my wife, Parnia, who always supported me with her endless love.

CHAPTER ONE

Introduction

1.1 Motivation

Image segmentation is one of the most important computer vision problems. It divides the image into meaningful areas. Image segmentation is based on several factors in the image, including density, color, brightness, texture, pixel continuity, and object knowledge. Image segmentation can be expressed as pixels classification in the entire image with semantic labels (semantic segmentation). Semantic segmentation labels each pixel with a set of object categories (e.g., car, bike, wound). Hence, it is a more complex task than image classification, which predicts a single label for the entire image.[1] Semantic segmentation has solved several problems in the healthcare field, such as detecting tumors, and wound areas.

Medical images, such as computed tomography (CT) scans, magnetic resonance imaging (MRI) scans, and raw images, are essential in the healthcare industry. These photos provide vital information that hospitals and physicians can use to treat and diagnose patients. However, most medical pictures have noise, intense inhomogeneity, and weak boundaries that require complex segmentation. For these reasons, automated image segmentation techniques such as thresholding and region growing often generate less accurate segmented images.

Deep learning techniques came about in the 2000s as more computational resources became available, and they_began to prove their significant capabilities in image processing applications. Deep learning techniques have emerged as a primary choice for image segmentation, particularly for medical image segmentation, due to their promising ability to automate feature extraction. Image segmentation based on deep learning algorithms has recently gotten a lot of attention.[2]

The segmentation of wounds plays an essential function in the monitoring and healing of wounds. However, manual image segmentation is time consuming, and it may not reproduce the same segmented area each time. computer-aided diagnosis (CAD) like machine learning and deep learning models are helping us to automate the segmentation problem with the high accuracy and faster process. In the meantime, deep neural network approaches may successfully extract the picture features, but deep learning models first require a large number of training images in order to automate both feature extraction and segment Regions of Interest (ROI) with the high accuracy.

1.2 Problem Statement

In this thesis, we consider two different datasets for segmentation. The first dataset is computerized tomography (CT) scans of bone. In this project, we segment the bone area and tissue area with a machine learning model (Random Forest). The second dataset is different wound and NIR (Near Inferred) images. The main goal for this project is to train the deep learning model to segment the wound areas in color images that were taken from a regular camera. Afterwards, we apply the segmented mask into NIR images to analyze the wound area. Figure 1 shows a sample of a wound image and its corresponding NIR image.

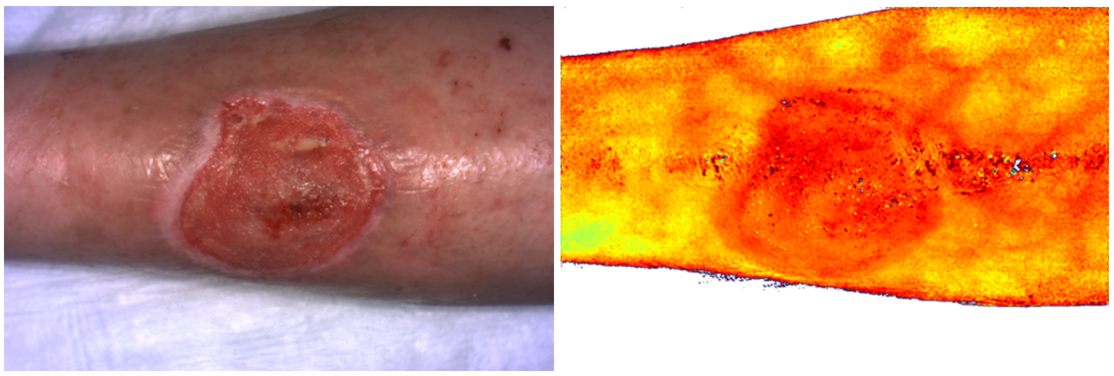


FIGURE 1: RAW WOUND IMAGE WITH CORRESPONDING NIR IMAGE

1.3 Thesis outline

The remainder of this thesis is divided into four chapter. Chapter two reviews the background of medical image segmentation methods such as manual, Threshold, Edge-Based, and Region-Based segmentation, as well as neural networks. The third chapter describes how to generate features for segmentation of CT scans using the Random Forest model (Chapter Three). In the fourth chapter, we express how to use a deep learning model (MobileNet 2) to segment wound areas and analyze NIR images (Chapter Four). Finally, the last part is the conclusion of the thesis: sentence or two (Chapter Five).

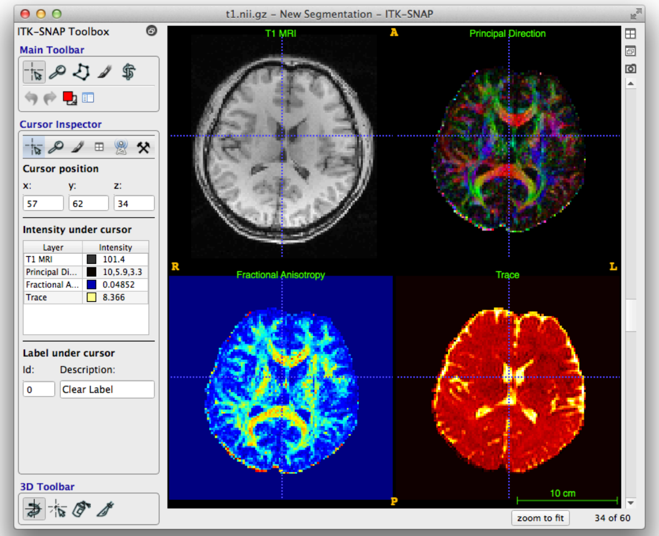
CHAPTER TWO

Background

2.1 Manual Segmentation

Manual segmentation is often conducted by a trained professional such as a radiologist or a specialist physician. These experts perform the segmentation by surrounding the ROI or marking the pixels of interest. Hence, the results of manual image segmentation are often the most accurate and most reliable. Automatic segmentation methods such as deep learning and machine learning require an accurate ground truth for the model to learn how to segment the images. Because of their accuracy, manual segmentation results are used as the ground truth. utilize expert knowledge.

To get the best results for manual segmentation and speed up the process, software with an easy-to-use user interface exist to assist with segmenting medical images. For manual medical image segmentation, there are several open-source software packages available that include various methods.[3] For instance, ITK-SNAP [4] is free and open-source software for manual and automatic segmentation. Figure 2 illustrates the ITK-SNAP program's interface. Even though these kinds of programs have some tools to assist with manually segmenting the ROI, it is time consuming to segment them manually, and it is not feasible in routine clinical practice. Given that the results are human generated, it is difficult to reproduce a ROI which can lead to inconsistency in diagnosis and treatment. In automatic segmentation methods like deep learning and machine learning, we need the accurate ground truth for model to learn how to segment the images, and manual segmentation results are used as the ground truth because we utilize expert knowledge for segmentation of target values and training machine learning and deep-learning models.



Cursor position

x: 57 y: 62 z: 34

Intensity under cursor

Layer	Intensity
T1 MRI	■ 101.4
Principal Di...	■ 10,5.9,3.3
Fractional A...	■ 0.04852
Trace	□ 8.366

Label under cursor

Id: 0 Description: Clear Label

FIGURE 2: USER INTERFACE OF MANUAL SEGMENTATION APPLICATION USER INTERFACE. HTTP://WWW.ITKSNAP.ORG

2.2 Threshold Method

Thresholding is one of the most common techniques for image segmentation. The thresholding technique works with pixel intensity and histogram analysis. This method uses a specific value as a thresholding value (T) to convert a grayscale level image into a binary image.

There are two different methods for thresholding: the global method and the local method.[5] The global method uses T as a threshold for the entire picture, then every pixel smaller than the T value becomes a background (black) and every pixel larger than the T value becomes the foreground (white). The CT image and corresponding histogram and segmented image by global thresholding are shown in figure 3. Furthermore, the local approach employs various threshold values, and these intensity levels are grouped together.

In general, establishing the precise threshold value might be challenging due to the similar intensity and pixel value in different regions. In the presence of artifacts, the performance of segmentation using the thresholding technique can be affected.[6]

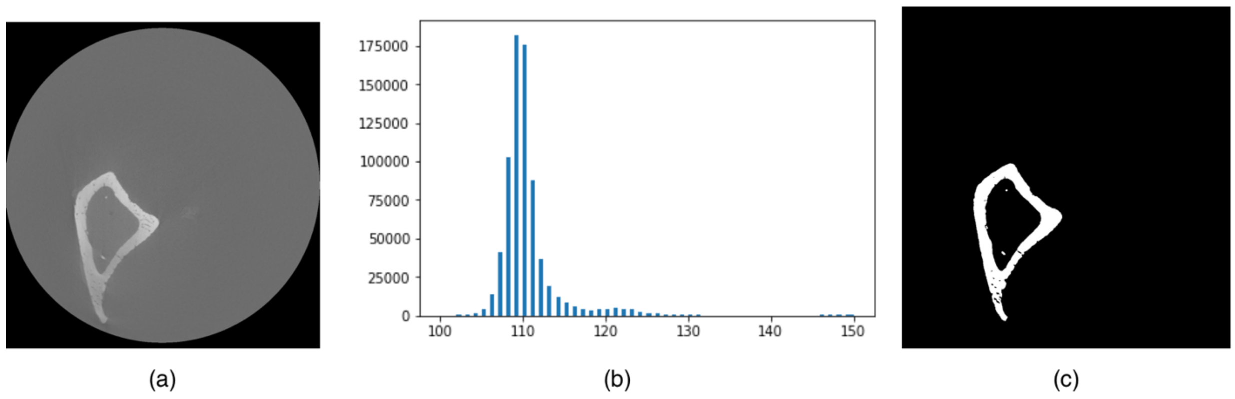


FIGURE 3: GLOBAL THRESHOLD METHOD, (A) THE INPUT CT IMAGE, (B) HISTOGRAM BETWEEN RANGE 70-170, (C) THRESHOLDING OUTPUT BINARY IMAGE

2.3 Edge-Based Segmentation

Edges are the boundaries of an object in an image; therefore, by obtaining the edge of each object, we can divide the objects in the image based on their edges. Edge segmentation uses the gradient (derivative) to find the edges in the image. Several functions are available that can find the edges based on the first or second derivative, such as Prewitt, Sobel, Roberts, Laplacian, Canny [7], and Marr-Hilclrath.

For instance, Canny works in four steps.[8] First, the image smoothed by a Gaussian filter to reduce noise. Second, the edge strength and direction are determined by performing a 2-D spatial gradient of the smoothed image with the Sobel operator. The third step is non-maximal suppression, which scans the image thoroughly to eliminate any unwanted pixel that may not be a part of the edges. To do this, every pixel is checked to be the local maximum in its neighborhood in the gradient direction. Step four, hysteresis, determines whether the edges are accurate, and removes any edges that are not real. To complete this step, we obtain the threshold value; any edge that is higher than the threshold is true, while any edge that is lower than the threshold is not.

However, edge detection has some limitations; for example, the presence of noise in a picture has an impact on its accuracy.[6] sometimes edges are not clear, and it is not easy to detect them. And it is difficult to do in clinical settings.

2.4 Region-Based Segmentation

Region-based segmentation approaches use pixel continuity and are based on the concept of homogeneity. These algorithms search for similarities between adjacent pixels by grouping pixels with similar characteristics such as pixel intensity, texture, color, and shape into single regions. There are two region-based methods: a. region growing and region-merging and region-splitting.

Region growing is considered the most straightforward technique, and it starts with some seed points, which have been grouped into different n sets and are chosen based on the feature and region of interest. Given the seeds, regions grow by connecting the adjacent pixels based on their characteristics. The regions are selected to be as homogeneous as possible. [9] If the criteria were a pixel intensity threshold value, the histogram knowledge might be utilized to determine the optimal threshold value for the stop point.

In the region-merging approach, similar intensity or color is utilized to link the similar components. The aim is to start with one region per pixel and then run a statistical test on nearby regions to determine if the mean intensities are similar enough to merge.

On the other hand, the region-splitting method starts with the entire picture. A region-splitting algorithm splits the image into multiple regions with similar pixel values, these regions are subdivided into different classes until there are no more regions to split. Figure 4 shows an example of region-based segmentation of brain image. Three different stages of the region growing algorithm from left to right; seed selection, connecting seeded points to the neighbors with similar property, and completed stage. Overall, region-based segmentation has a better result than edge-based segmentation in noisy images where it is hard to detect the edges.

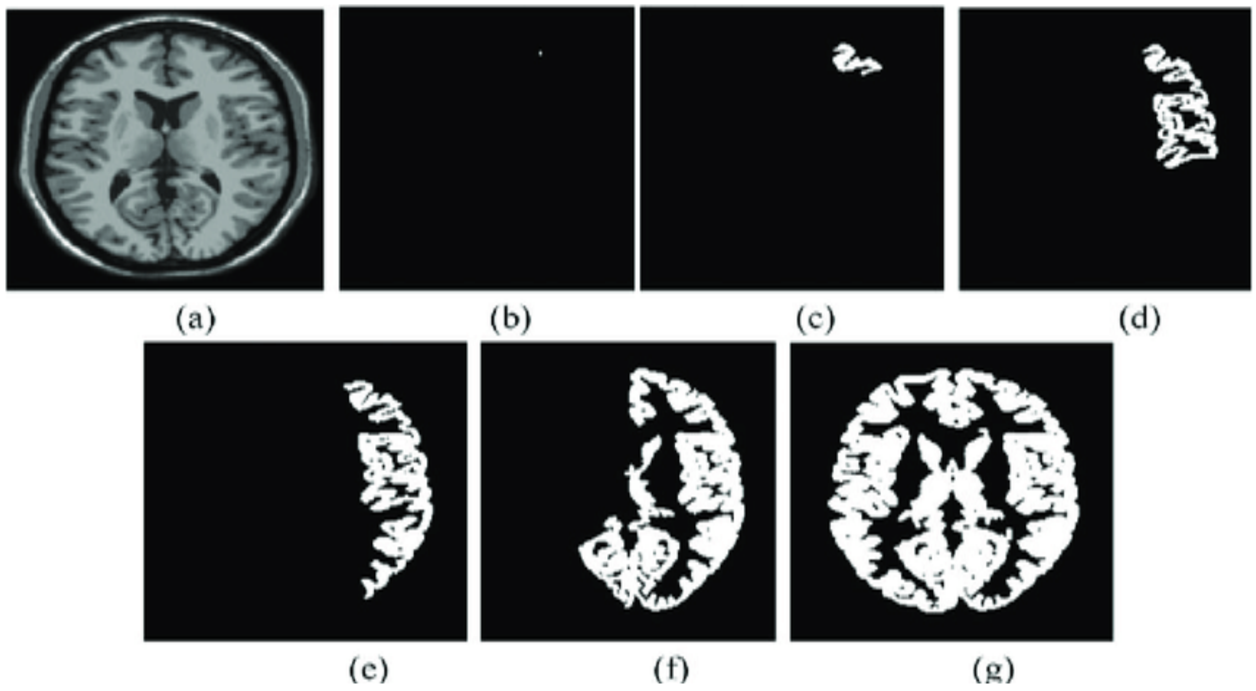


FIGURE 4: REGION GROWING SEGMENTATION PROCESS. (A) INPUT IMAGE (B) 1 SEED POINT SELECTED; (C) 10 NEAREST NEIGHBOR POINTS ADDED (D) 100 NEAREST NEIGHBOR POINTS ADDED (E) 1000 NEAREST NEIGHBOR POINTS ADDED (F) 2000 NEAREST NEIGHBOR POINTS ADDED (G) ALL NEAREST NEIGHBOR POINTS ADDED, REGION IS COMPLETED. [43]

2.5 Neural Networks

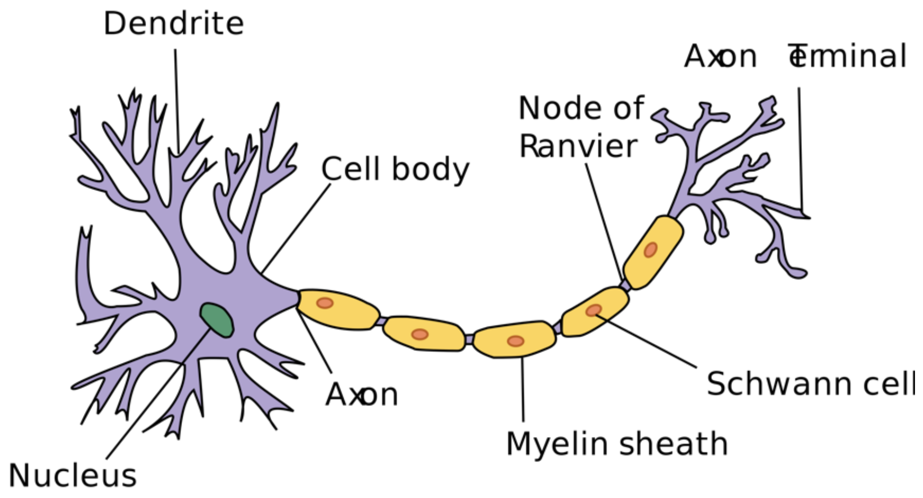


FIGURE 5: STRUCTURE OF NEURON [10]

Neural Networks (NNs), also known as Artificial Neural Networks (ANNs), are a subset of machine learning algorithms. In the past decade, they have become famous for their diverse applications, from finance to machine vision.[11] The neuron structure is shown in figure 5; an input is received by the dendrites and is transferred to the nucleus which transmits the input to the axon terminals. The axon terminals are linked to the next neuron dendrite.

The ANNs are mathematical model which simulate the human brain's behavior, replicate the way real neurons communicate with one another, and let computer systems recognize patterns and solve common issues in the domains of artificial intelligence and machine learning. Unlike the human Neuron structure, it goes back from the output to the input again to strengthening the connections.

Figure 6 shows a simple structure for NN models. A NN contains 3 different layers: input layer, hidden layer which may be one or more layers with several neurons, and output layer. Hidden layers are a series of mathematical functions, each meant to create an output that is particular to the desired outcome.

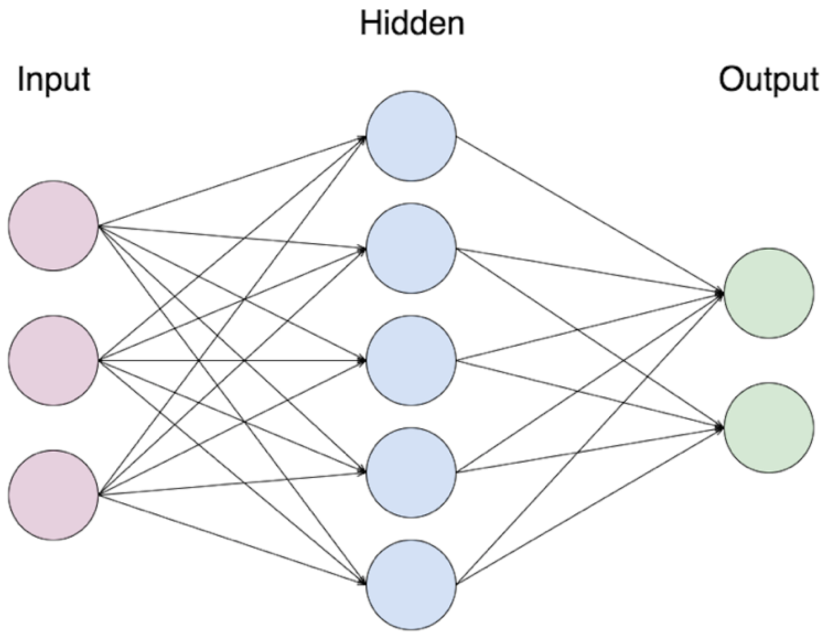


FIGURE 6: SIMPLE NEURAL NETWORK STRUCTURE [12]

One of the most important and oldest NN models is the Perceptron which was created by Marvin Minsky and Seymour Papert in 1969[13]. This model was not the first version of the Perceptron model, but it overcomes some of the limitations of the prior versions which was not having a scientific way to know whether a perceptron performs a specific task or not, by introducing numerical weight and mechanism for learning them to the Perceptron model.[13] These weights are representative of the importance of each input.

The Perceptron model is a binary classification algorithm which works in 4 steps. First, it initializes the weight vector and learning rate. Second, it calculates the function f based on the equation below, where w is the weight and x is the input. In addition, $\widehat{y_1^1}$ is the threshold function.

$$f = \sum_{j=0}^{k} w_j^1 x_{1j} = w_0^1 x_{10} + w_1^1 x_{11} + w_2^1 x_{12} + \ldots + w_k^1 x_{1k}$$

$$\widehat{y_1^1} = \{ 1\ if\ f > 0\ and\ 0\ otherwise$$

Function f here is a predicted output of the model. After getting the predicted output, the model then uses the function to find the error between the actual value and predicted value and updates the w to find a more accurate predicted output. This is an important step in all machine learning methods. Learning the w in the machine learning algorithms, the learning process is defined as an optimization problem. An algorithm is used to explore the space of possible weight settings that the model may employ to generate the best possible predictions. The loss function is the error of function f that the model tries to minimize or maximize with respect to w. There are several loss functions available that can be used based on the model and application. The neural network model is trained with the stochastic gradient decent optimization algorithm and weights are updated with the "backpropagation" error algorithm.

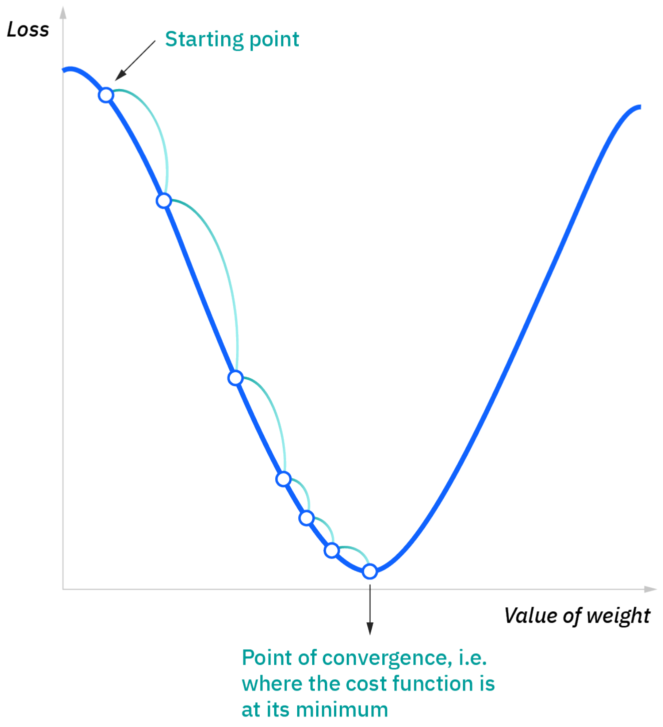


FIGURE 7: FINDING MINIMUM OF WEIGHT WITH RESPECT OF LOSS FUNCTION [52]

In figure 7, the starting point is the initial w and the learning rate η is the distance between each step. The initial w is used to calculate the prediction value and the error value. The gradient descent method attempts to adjust the weights so that the next estimation decreases the error, meaning that as the gradient descends, the predictions become more accurate. The batch gradient equation [28] is provided below where θ is the parameter (weight), η is the learning rate, and J is the loss function:

$$\theta = \theta - \eta \cdot \nabla \theta J(\theta)$$

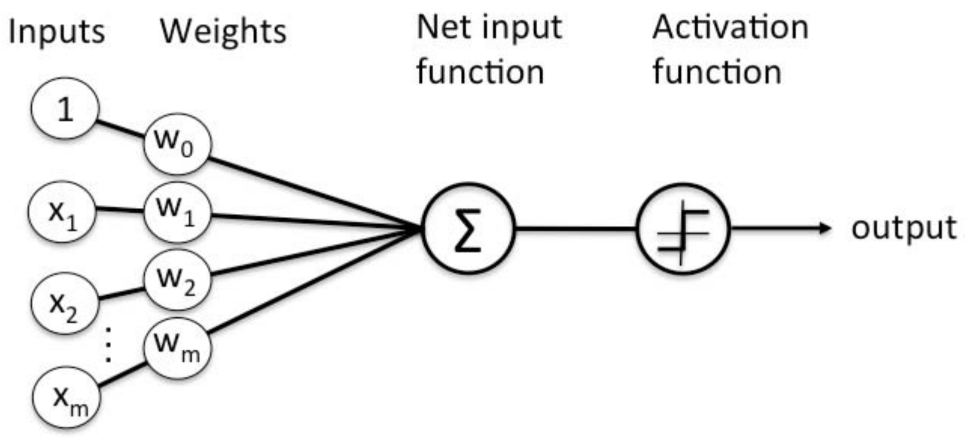


Figure* 8: *Architecture of perceptron model [51]

The fourth and final step in Perceptron model calculates the output based on the activation function. Figure 8 illustrates the Perceptron's architecture. The function f calculates the weighted sum of the inputs. In this case, f might be a signal or a simple linear function. Even though linear equations are simple and easy to solve, they can neither recognize nor learn complex data mappings. Furthermore, because the neuron cannot identify the output, the neural network performs like a linear regression with limited power and performance. Hence, activation functions are required to map the neuron output to the appropriate range, for instance, zero to one or minus one to one, depending on the activation function.

The accuracy and performance of neural networks rely on the activation function.[14] There are two different types of activation functions:

- Linear Activation Function
- Non-Linear Activation Function

The linear activation functions have a range of (-) infinity to (+) infinity. Therefore, these functions perform poorly in complex situations that use multivariable inputs in the neural network. For these reasons, linear activation functions are not frequently used.

On the other hand, non-linear activation functions work great in most cases and are the most used functions in the neural network field. Some of the most popular non-linear activation functions are ReLU (rectified linear unit) ($f(x) = \max(0, x)$), Tanh, and Sigmoid ($f(x) = 1/(1 + e^{-x})$). Figure 9 illustrates the shape of common activation functions.

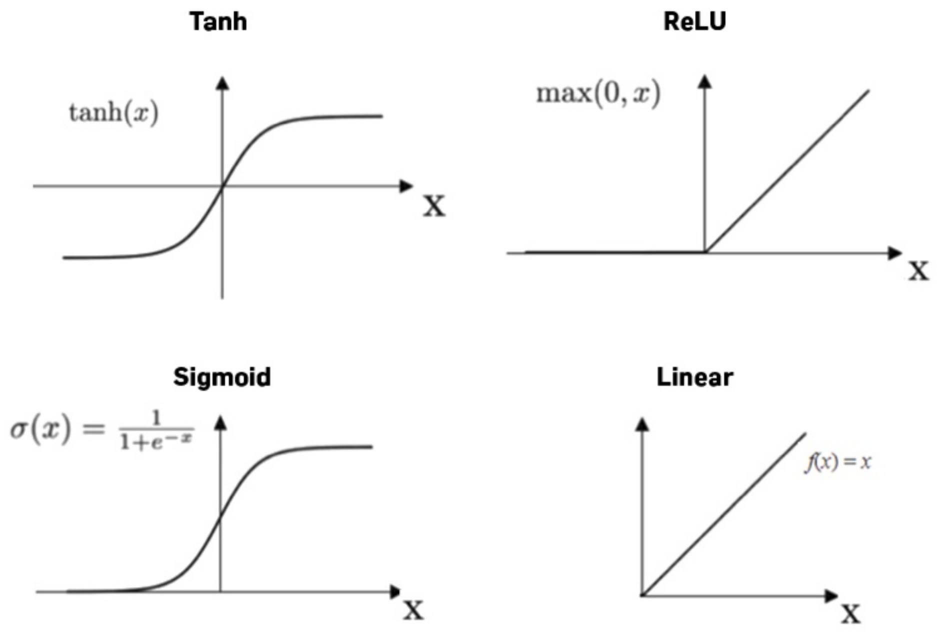


FIGURE 9: MOST COMMON ACTIVATION FUNCTIONS [15]

2.6 Deep Learning

Deep learning is a neural network with multiple hidden layers, which is also known as Deep Neural Networks. With the existence of big data and powerful computational resources, machines can use deep learning for knowledge-based applications like computer vision, voice recognition, and language translation. deep learning can extract high-level features from raw data during the learning process unlike traditional machine learning algorithms like logistic regression and decision tree which require a set of features or representative information to train the model. For example, in a facial recognition system, logistic regression does not examine a user's face directly. Instead, it uses pre-existing features such as edges or shape of facial parts to learn how these features correlate with one another. In addition, traditional machine learning models cannot influence how features are defined in any way. Meaning that, it depends on human interaction.

Deep learning, on the other hand, operate directly with raw data, without any prior feature extraction phase. Figure 10 represents a deep learning example of feature extraction in the hidden layer. In this example of face recognition, the first layer may learn to identify edges, the second layer may learn to detect facial components, and the third layer may learn to connect the facial parts to the person. However, this is only an example of what could happen in the layers. In fact, NNs (Neural Networks) are not interpretable (black box).

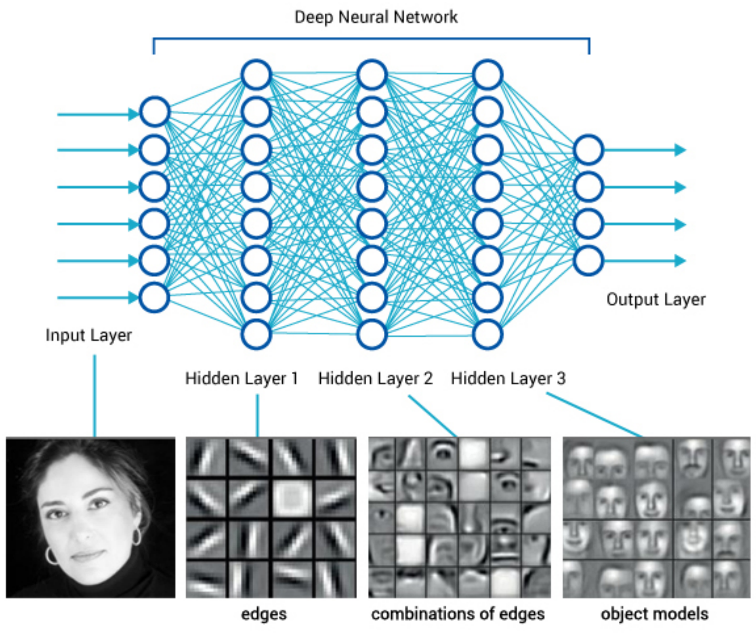


FIGURE 10: VISUAL REPRESENTATION OF DEEP LEARNING FEATURE EXTRACTION IN A FACE IMAGE [50]

2.7 Convolutional Neural Network

Convolutional neural networks (CNNs) are the most common deep learning-based models. CNNs have been used widely in image recognition, image classification, image segmentation, face recognition, and voice recognition.

Digital images consist of a 3-dimentional array which contains pixel values of the picture with the size of the array dependent on the image resolution. Images are made up of a grayscale and a color image. Gray scale images are represented as $Height \times Weight \times 1$ and color images $Height \times Weight \times 3$ where 3 represents the three RGB (Red, Green, Blue) channels. Figure 11 shows the example of a bird image with the RGB channels. The combination of these three channels produces the final color image.

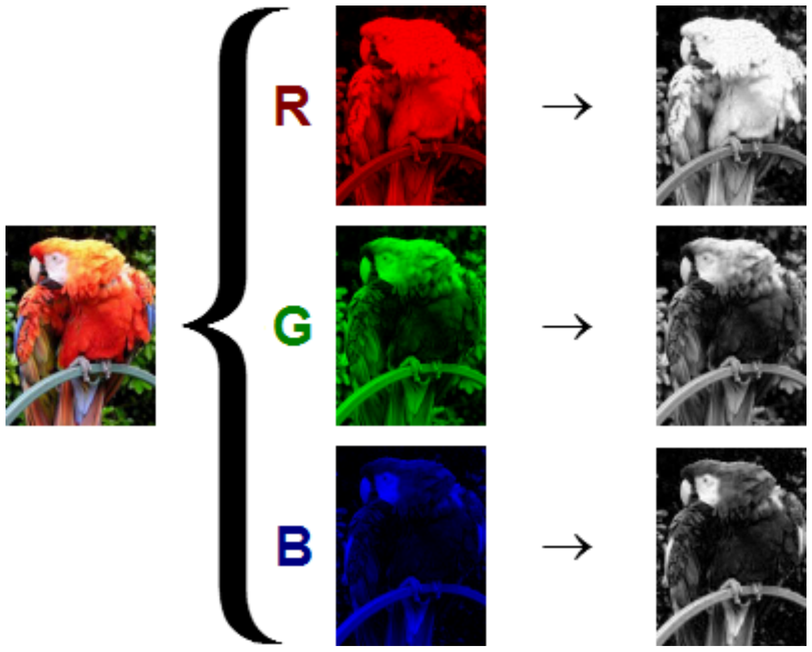


FIGURE 11: RGB CHANELL OF IMAGE [49]

CNN usually takes third-order tensors such as the height, width, and RGB channels of an image as an input. The picture pixel values are fed into the CNN as an input, then the CNN's neurons extract image information and features in the hidden layers. The hidden layers automatically learn the higher-level features of the corresponding input.

The hidden layers look for the local regions instead of the entire image, and the neuron in the next layer gets the input from the corresponding part of the image. This approach dramatically decreases the number of parameters (weights) that it needs to solve in the network. In addition, CNN keeps the local connection weights fixed for all the neurons in the following layer. This connects the neighboring neurons in the next layer with the same weight as the previous layer's local area. Therefore, it eliminates numerous unnecessary factors and decreases the number of weights. All these steps help to reduce the image into a form that is easier to process without losing features. Furthermore, it helps the network to recognize the features regardless of their position in the image. [16]

CNN architecture contains five different types of layers:

- Input layer
- Convolutional Layer
- Pooling Layer
- Fully Connected Layer
- Output Layer

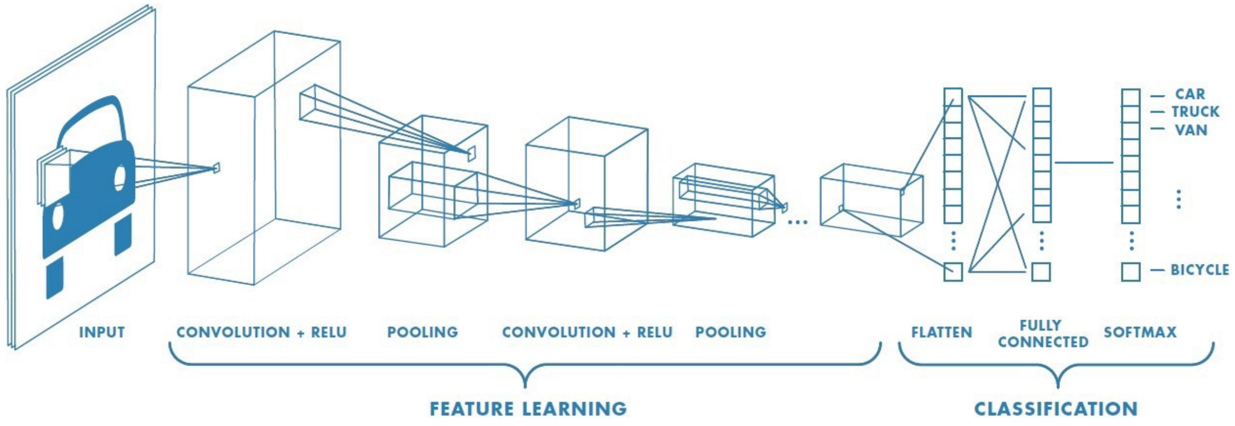


Figure 12: CNN Architecture [48]

Figure 12 shows the architecture of CNN for image classification. The first layer consists of the pixel values of the raw images which serve as the input. In the following layer, the convolutional layer calculates the scalar product between their weights and the region related to the input volume in order to identify the output of the neurons associated with a specific region. The rectified linear unit (ReLu) then applies an activation function, such as sigmoid, to the preceding layer's activation output. The RELU activation function is used in here to get the best performance. [17] Afterwards, the pooling layer reduces the number of parameters inside the activation output to reduce the space dimensions of the provided input. Finally, the fully connected layer acts similar to ANNs and generates a class score (between 0 to 1). Multiple CNN structures and models can be created by changing the design, layout, and/or quantity of data to achieve different classification goals. Changing the design means changing the shape and numbers of the input layer, convolutional layer, pooling layer, and fully connected layer. The next section will describe the different CNN layers.

2.7.1 Convolutional Layer

A fundamental element for CNN is the convolutional layer. The main goal of this layer is to extract features using a learnable kernel. Convolution is a mathematical process that requires two inputs: a kernel (filter) and an image. Figure 13 shows the convulsion a 7*7 image and a 3*3 filter.[18]

When data passes through a convolutional layer, the layer convolves each filter across the input's spatial dimensions to create a 2D activation (also known as a feature map). The scalar product is calculated for each value in that kernel as it progresses through each value.

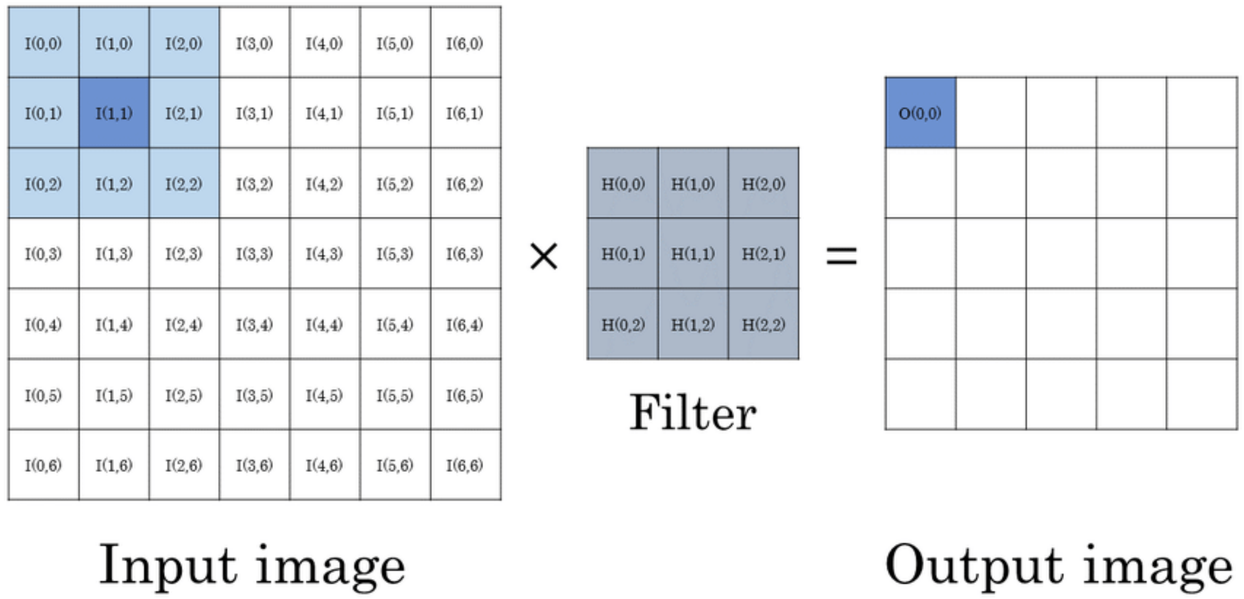

I(0,0) I(1,0) I(2,0) I(3,0) I(4,0) I(5,0) I(6,0)
I(0,1) I(1,1) I(2,1) I(3,1) I(4,1) I(5,1) I(6,1)
I(0,2) I(1,2) I(2,2) I(3,2) I(4,2) I(5,2) I(6,2)
I(0,3) I(1,3) I(2,3) I(3,3) I(4,3) I(5,3) I(6,3)
I(0,4) I(1,4) I(2,4) I(3,4) I(4,4) I(5,4) I(6,4)
I(0,5) I(1,5) I(2,5) I(3,5) I(4,5) I(5,5) I(6,5)
I(0,6) I(1,6) I(2,6) I(3,6) I(4,6) I(5,6) I(6,6)
Input image
×
H(0,0) H(1,0) H(2,0)
H(0,1) H(1,1) H(2,1)
H(0,2) H(1,2) H(2,2)
Filter
=
O(0,0)
Output image

FIGURE 13: CONVOLUTION OF 2D IMAGE. THE LEFT MATRIX IS THE INPUT IMAGE, THE FILTER STARTS WITH THE FIRST PIXEL IN THE TOP LEFT AND MOVES TOWARD THE BOTTOM RIGHT CORNER. THE FILTER CALCULATES THE CONVOLUTION OF THE PIXEL AND GENERATES THE OUTPUT. [18]

This layer also reduces the complexity of the model by zero padding and stride hyperparameter. The convolutional layer shifts the filter windows by a quantity called stride. For example, if stride is set to one, the kernel moves across each input one by one; hence, the feature map becomes much larger the lower the stride value. However, a lower stride value results in redundant overlapping pixel values between the kernels.

The filter does not always fit the input image as it exceeds the picture's borders. This issue can be solved using one of two approaches. The first approach is known as valid padding which only applies the filter when it lies inside the input boundaries. However, this technique loses some information in the input image. The second approach, which is called zero-padding, pads the image borders so that the filter fits within the input image boundaries. The size of the zero-padding depends on the stride value, and if the stride value is not appropriately set with the value of the zero-padding, the neurons will not fit neatly over the input. Figure 14 illustrates how the filter goes beyond the image border and how zero-padding solves this issue.

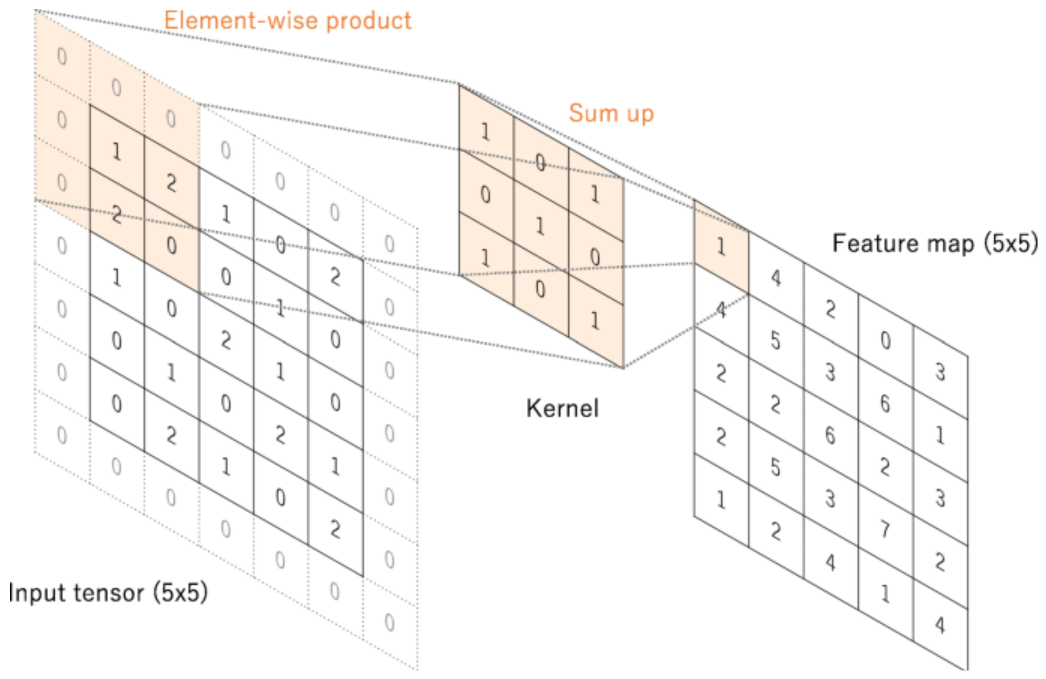


FIGURE 14: ZERO PADDING IN A CONVOLUTION OPERATION. THE INPUT AND THE FEATURE MAP HAVE THE DIMENSION OF 5X5. THE KERNEL SIZE IS 3X3 AND THE STRIDE SIZE IS 1 IN THIS EXAMPLE. [47]

2.7.2 Pooling Layer

The goal of the pooling layer is to reduce the complexity and parameters of the network by reducing the dimensions of the feature map. However, this layer retains the pixel values with one of two pooling methods: max pooling and average pooling. Max pooling is the most common type of pooling used in CNNs. In this technique, the pooling kernel moves across the input matrix by taking the largest pixel value and placing it into the corresponding output position. Instead of taking the largest pixel value, average pooling takes the average of the pooling kernel. Figure 15 shows an example of max pooling with a kernel size of *2*2* applied to an activation map of *4*4* resulting in an output with reduced dimensions.

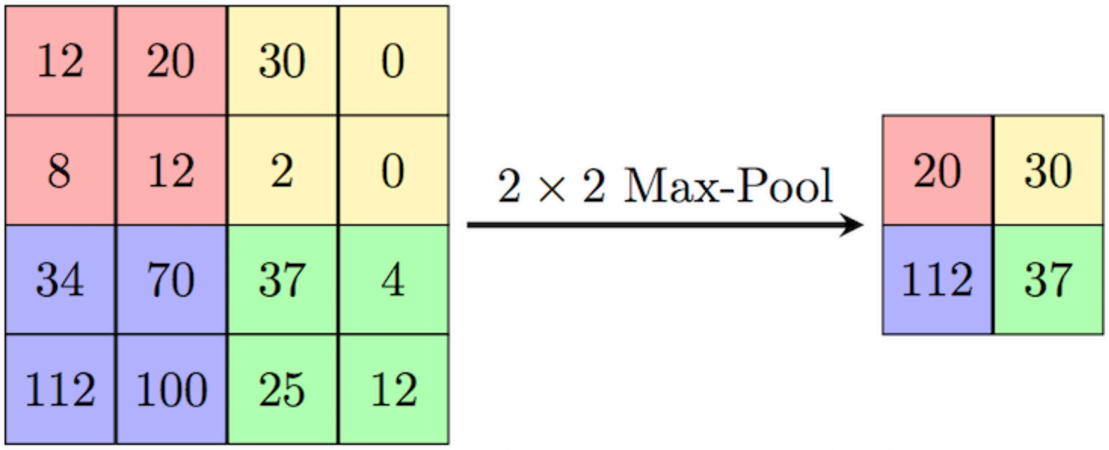


Figure 15: Max Pooling [46]

2.7.3 Fully Connected Layer

The last part of the CNN is the fully connected layer, which functions similarly to a feed-forward NN. The input of this layer is the output of either the pooling layer or the convolutional layer. In order to be fed into this layer, the input is first flattened, meaning the input matrix is converted into a one-dimensional array. Each layer of the fully connected layer uses the ReLU activation function and at the end of this layer, the network uses the SoftMax activation function to obtain a final probability value just like the NN. The CNN model uses the highest probability value as the final output.

2.8 Performance Metrics of Machine Learning Algorithms

Measuring how correctly the classification model predicts the intended outcome is critical when creating and optimizing the model. However, this measure is never the whole story since it can still produce unreliable results. Therefore, extra performance assessments are used to elicit more reliable results from a model. The accuracy, precision, recall, and f1-score for every model are all evaluated.

True Positive, True Negative, False Positive, and False Negative measurements are represented by TP, TN, FP, and FN in these equations, respectively.

2.8.1 Accuracy

The most important classification metric is accuracy. Accuracy is defined as the number of correct answers out of the total number of cases evaluated and is used for both binary and multi-class classification problems. Accuracy is calculated in the following equation:

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

2.8.2 Precision

Precision is the percentage of the predicted positives that are actually positive. In a segmentation problem, for example, precision evaluates the percentage of pixels in a segmentation that are successfully segmented. Precision is a relevant metric to consider when we want to be certain of our prediction. Precision is calculated in the following equation:

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

2.8.3 Recall

Another helpful metric is Recall, which addresses a different question: what percentage of real Positives is classified correctly. It computes the fraction of correctly segmented pixels in the ground truth by:

$$Recall = \frac{TP}{TP + FN}$$

2.8.4 F1-Score

The F1 score shows the segmentation's and the ground truth's similarity and is the harmonic mean of accuracy and recall, and it ranges from 0 to 1. F1-score is calculated by:

$$F1 - Score = 2\ X\ \frac{Precision * Recall}{Precision + Recal}$$

CHAPTER THREE

CT-scans segmentation using traditional machine learning approach

Introduction

A CT scan combines a sequence of X-ray pictures taken from various angles around a body with computer processing to generate cross-sectional images (slices) of the bones, blood arteries, and soft tissues inside the body. CT scan images contain more information than standard X-rays. CT scans can provide accurate three-dimensional information on the size and position of the target volume, and the position of any critical organs or structures of interest. With the growing use of CT scans for diagnosis, treatment planning, and clinical research, using computers to assist radiologists in clinical diagnosis and treatment planning has become necessary.

In the healthcare industry, the primary goal of using computers is to automate processes and decrease human error. Hence, automated algorithms and techniques are required to segment the ROI in CT and MR imaging. Medical image segmentation algorithms depend on the application. For instance, segmentation of the bone in the thorax differs from segmentation of a tumor in a brain image. Furthermore, both CTs and MRIs have many artifacts such as motion artifacts, ring artifacts, and noises due to electronic systems.[19] Hence, there is no single segmentation algorithm that works with all sorts of images. Model-based segmentation is one of the best choices to automate the segmentation.

In this chapter, we solved multi-class classification for CT-Scan dataset. The primary goal of this research is to segment bone and tissue areas. The bone region may be easily segmented using thresholding approaches. However, division of the tissue region is more challenging. For segmentation of these regions, we use the traditional machine learning model Random Forest. The rest of this chapter describes our dataset, features extraction using different filters, and the Random Forest model for segmenting these images.

3.1 Dataset

The dataset was provided by Professor Priyantha Premnath from the Biomedical Engineering department at the University of Wisconsin-Milwaukee which contains 3D Mini-CT scans of bone. Dataset contains 1,008 images and the dimension of each image is 984×1024. The figure 16 shows that the images comprise four distinct areas:

1. The dark zone, which provided no useful information about the image
2. The gray region, which is the background of the CT image
3. The dark gray, which is the tissue
4. The white region, which is the bone

The similarity between the tissue region and the background making the segmentation more complex. In addition, in some images like the middle image in the figure 16, the image contains texture area and lots of noises which makes some segmentation algorithms do not perform.

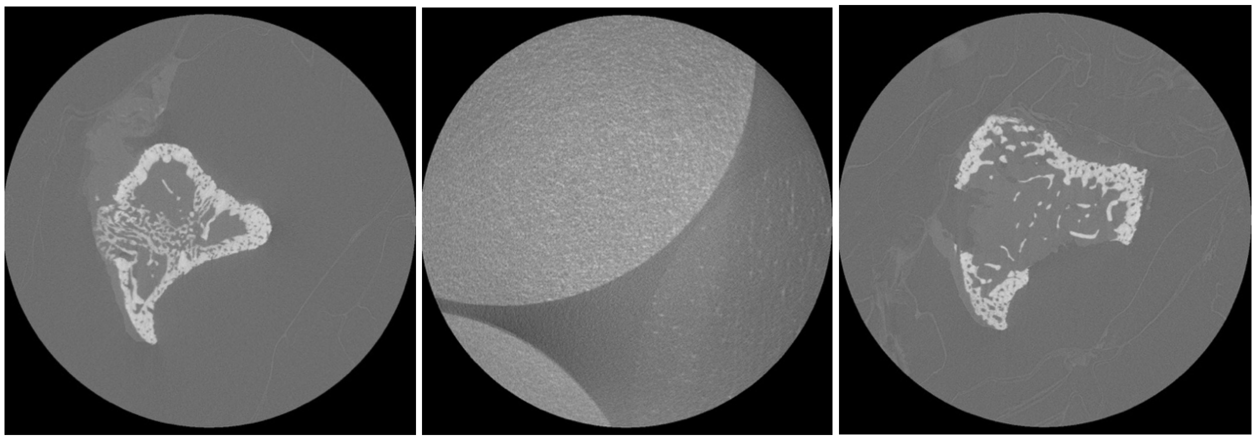


FIGURE 16: CT-SCAN SAMPLES OF OUR DATASET

3.2 Preprocessing

Preprocessing is the most crucial step in machine learning and deep learning projects. Because the quality of the data has a direct impact on machine learning model performance. In the CT Image segmentation problem, we split the data into training and testing sets of 30 and 10 images, respectively.

The CT images in our dataset contain a lot of noise. Therefore, for getting a better result in segmentation, we denoised the images before starting the feature extraction and training the model. The blur filter should be applied to the image to reduce the noise in the image. There are some blur filters exist that can smooth the image and reduce the noise of the images like Gaussian filter. However, for segmentation purpose we need to smooth the image in a way to not to lose too much information in the image like the edges. We use Bilateral filter which is one of the non-linear filters that preserve the edges while denoise the image.[20]

3.2.1 Feature Extraction

Feature extraction is the key point of segmentation by using tradition machine learning techniques. Feature extraction in the image processing is the process of transforming an image into set of vectors that describe the image. In addition, these features could be used for training a machine learning model.

The most important feature is the pixel values of the input image. For capturing the local appearance characteristics, we applied automated edge detection filters to capture edges, namely Robert, Sobel, Scharr, and Prewitt. To reduce noise in images we applied a median filter. In addition, we apply some automated segmented result such as thresholding and K-mean Clustering as a feature as well.

We generated 32 different 2D-Gabor filters. Gabor filters are widely used in image processing and computer vision for texture analyzing and edge detecting. The Gabor function is a complex sinusoid with Gaussian modulation that fulfills the monotonicity and differentiability requirements. [23] The equation below is the Gabor filter:

$$g\ (x, y, \omega, \theta, \psi, \sigma) = exp\ (-\ (x\ '2\ -\ y\ '2)\ /\ 2\sigma\ 2)\ exp\ (i\ (\omega x\ ' + \psi))$$

$$x\ ' = x \cos\theta + ysin\theta$$

$$y\ ' = -\ x\ cos\theta + ycos\theta$$

Gabor filters have shown to be an effective technique for obtaining spatially localized spectral characteristics, which are commonly used in pattern analysis. Figure 17 which has 40 different Gabor filter are look like the first layer of deep learning feature extraction in figure 10. For generating bank of Gabor filter, we set different values for each ω, θ, ψ, σ to generate the 32 different Gabor filter.

At the end we apply all these filters to all training and testing dataset to extract all the features and create a data frame for our random forest model.

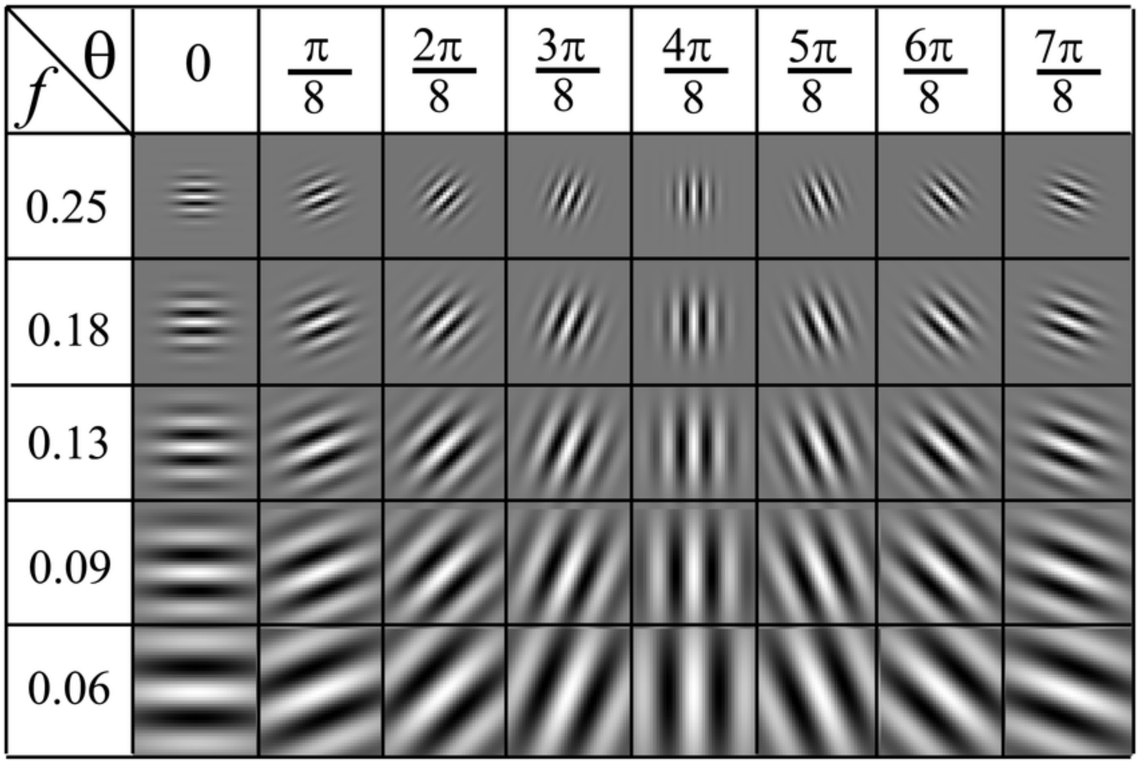


FIGURE 17: SAMPLE OF GABOR FILTER WITH COMBINATION OF θ (IN RADIANS) AND F (IN HZ) AND ψ = 0. [45]

3.3 Machine Learning Method

We used the Random Forest model to train our model to segment the CT-scans in our dataset. Random Forests, also known as random decision forests, are an ensemble learning method (bagging) which was first proposed by Tin Kam (1998). Random Forests use several randomized decision trees [29], where each tree is typically trained with a different subset of the training set; in other words, all trees are independent. For the output, Random Forests use the rule of majority votes to decide the final output. Random Forests have several appealing characteristics, including:

1. Computational efficiency in both training and classification
2. Probabilistic output
3. Seamless handling of a wide range of visual features (e.g., color, texture, shape, depth)
4. Inherent feature sharing of a multi-class classifier

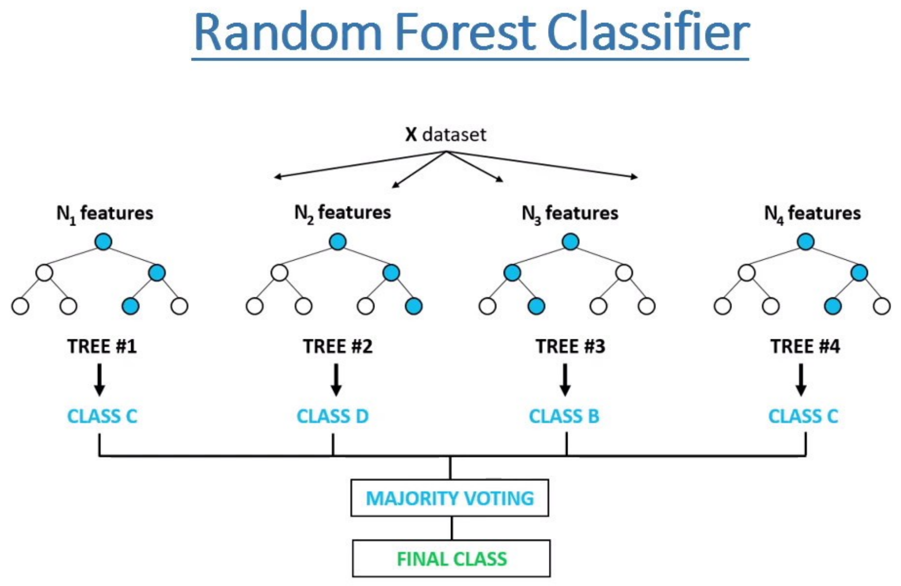


FIGURE 18: RANDOM FOREST CLASSIFIER. IN THIS EXAMPLE THE RANDOM FOREST HAS 4 INDEPENDENT DECISION TREES WHICH HAVE DIFFERENT RESULT FOR A SAME PROBLEM. AT THE END IT CHOOSES CLASS C AS AN OUTPUT BASED ON THE MAJORITY VOTING.[44]

In addition, Random Forests have the benefit of providing us with access to the information gained during the training phase, which can be interpreted by evaluating the importance of various features that we generated for the classification job.[22]

3.3.1 Training and implementation

For training the Random Forest model, we collected the ground truth of ROI in our dataset. Because we used a traditional machine learning model, only a small amount of labeling data is required. The annotation was done by manual image segmentation in Apeer[21] as it provides us the annotation tools to group different objects by color. Apeer also allows us to export the output images with different values based on their class. For example, in our case the background had a value of zero (black), the tissue area was one (gray), and the bone area was two (white). After segmentation, we normalized these data to fit within an image range from 0 to 255. The next figure shows the input image and the target values that we annotated.

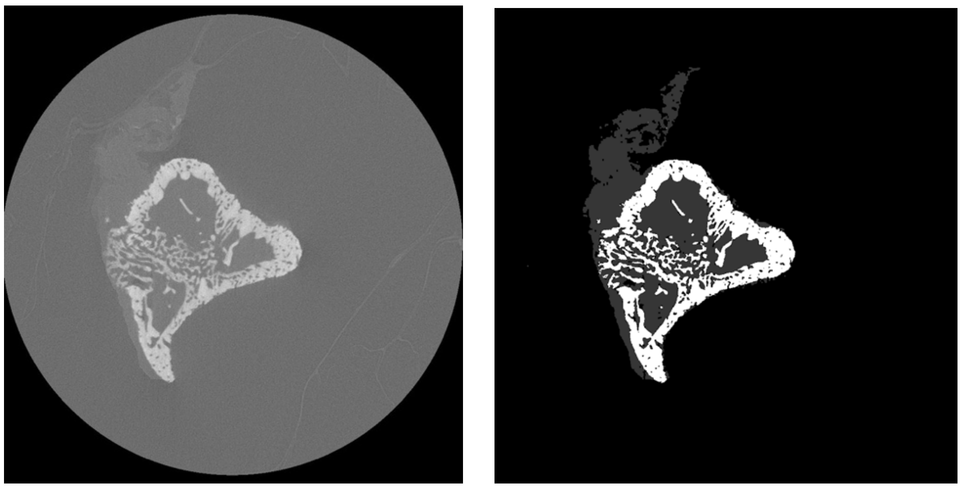


Figure 19: Input and label images for CT scan segmentation

To create a random forest model, we utilized Python programming [25] and a Scikit-Learn package [26]. The code was developed on a MacBook Pro with a 2.2GHz Intel i7 quad-core processor, 16 GB of 1600 DDR3 memory, and Intel Iris Pro 1535 Megabyte graphics.

Since the machine lacks a GPU, the code is executed using the CPU. Training the random forest model took 1 hour approximately.

We extract the feature of all the images in both training and testing sets. After extracting all the features, we create a data frame where each pixel and its corresponding features and output are in the same row. In the next step we use the Scikit-Learn to create and train a Random Forest model. After training the model, we evaluate the features which random forest model gave us and eliminate the less important features. This could help us to improve efficiency and speed of our training and predicting phase.

3.4 Results and discussion

The experimented Random Forest model is tested on ten images from the mini-CT scans dataset. In the test dataset, we observed accuracy of 97.4 percent. The precision and recall are also 97.3 and 95.1 percent, respectively, in the testing set. The result of our segmentation model is showing that even by using 30 images, our random forest model can generate the accurate segmentation result for all our dataset images.

Figure 20 illustrates inputs and segmented images from our model. By using a semi-automated segmentation method like thresholding, we need to analyze the histogram. Finding the appropriate threshold values could be time-consuming and not perform well in some images with noise. However, we automate this process by using the machine learning method, and each image can be segmented in just a second.

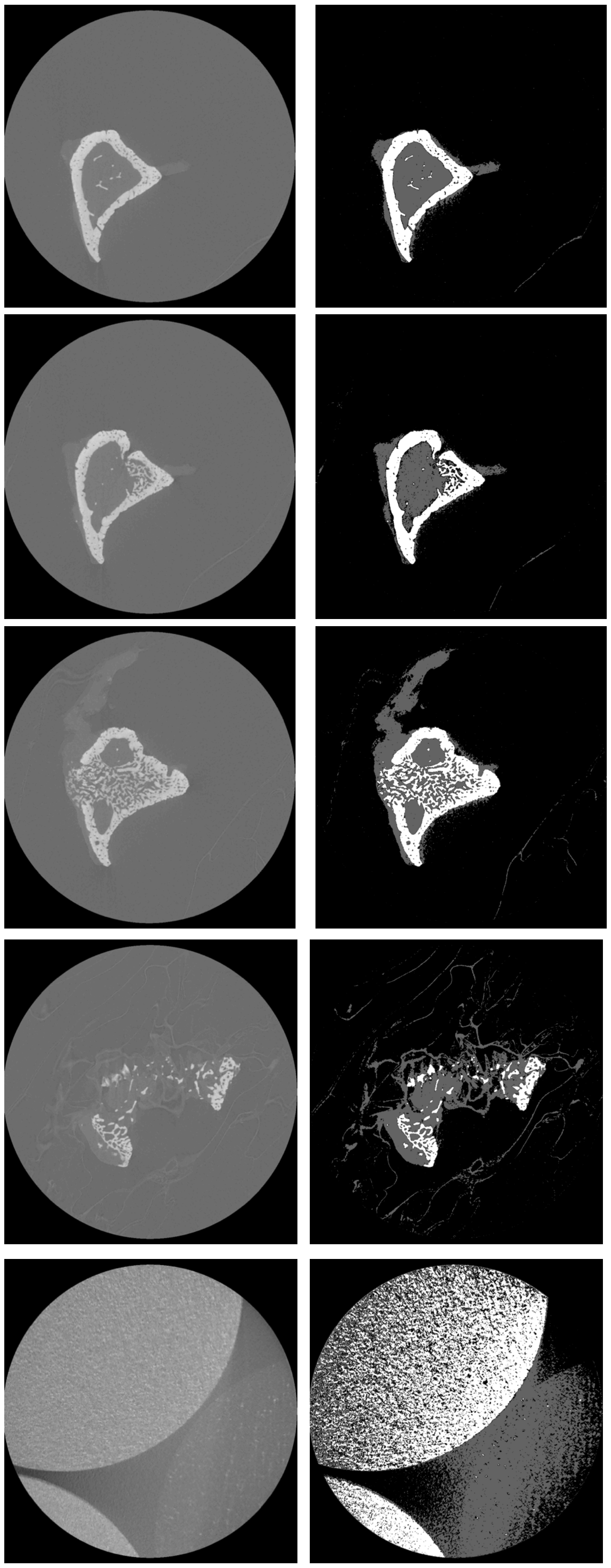


Figure 20: (A) is the input data in our Random Forest model and (B) is the output(segmented) images.

3.5 Conclusion

We proposed efficient machine learning techniques for segmenting CT images in this Chapter. While segmentation of these images using semi-automated algorithms limited the accuracy of the results, this approach provides insight into segmenting these medical images without analyzing them and with high accuracy. There is no requirement for a large amount of data when using traditional machine learning algorithms like Random Forest to train the model. Although we had a large number of images to use deep learning algorithms, labeling these data and training the deep learning was time-consuming and unnecessary. We generated features with filters such as 2D Gabor, different edge detection, median, k-mean clustering, and variance. The random forest model allows us to assess these features and determine how important the features in the training phase are. This evaluation allows us to identify the features that have either no effect or little effect on the model's accuracy and eliminate them. As a result, we obtained a model with 97.4 percent accuracy that can generate segmented images in just a second. The results indicate that we can generate a model that can be used in clinical practice with only a small number of images.

CHAPTER FOUR

Wound Analysis and Segmentation Using Deep Learning

4.1 Introduction

The largest organ in the human body, the skin, regulates water, electrolytes, and body temperature, and protects the internal organs from external noxious agents such as microorganisms.[30] When the skin is damaged for any reason, such as from ulcers, burns, tumors, or wounds, these functions are no longer appropriately performed. Therefore, it is crucial to restore its integrity as soon as possible.[30]

A wound is classified as acute or chronic based on the duration and type of the healing process. [31] An acute wound is a skin injury that arises suddenly because of an accident or surgical injury. It heals in a predictable way over an expected period depending on the size, depth, and severity of the skin damage. Chronic skin wounds, however, are caused internally by disease. Chronic wounds caused by vascular disease, venous insufficiency, unrelieved pressure, and diabetes are associated with conditions more common in older patients.[32] Wounds, both acute and chronic, are a challenge and a burden for healthcare systems worldwide. Each year, acute wounds afflict 11 million individuals in the United States, whereas chronic wounds impact more than 6 million people, resulting in annual Medicare costs of up to $96.8 billion (about $300 per person in the US).[34]

The basis of wound treatment is wound monitoring and analysis which consist of area measurement and tissue analysis. Manual wound treatment techniques are time-consuming and labor-intensive; the results are often incorrect, and the processes are unhygienic.

Automated wound segmentation can avoid these problems,[33] and enables faster data entry into the electronic medical record, which improves patient care.[35] Moreover, appropriate wound evaluation is essential to ensure healing progress. Analysis of wound tissue oxygen saturation levels (StO2) is one of the indicators used to evaluate the progress of wound healing. The StO2 provides detailed information for healthcare practitioners to identify detrimental factors that delay wound healing.[36]

StO2 can obtained from the NIR wavelength. Kent Imaging recently developed the Snapshot$_{NIR}$ device for calculating Sto2 from NIR images.[37] Snapshot$_{NIR}$ uses the well-documented spectral characteristics of oxyhemoglobin and deoxyhemoglobin within the near-infrared range to determine Sto2. Snapshot$_{NIR}$ can assess relative quantities of oxygenated and deoxygenated hemoglobin based on these known spectral signatures and compute a tissue oxygenation ratio using various wavelengths of NIR light. Because it works with reflected light, the Snapshot$_{NIR}$ is completely non-invasive and does not need to touch the tissue.[37] When looking at chronic wounds, this is especially significant since measurements may be collected within the wound bed as well as the periwound from a quick-captured image. however, this device is proprietary**.** For generating NIR imaging, Advancing the Zenith of Healthcare (AZH) transforms the color code of the wound area into the hex code. In addition, they use a jet-color mapping table that has the corresponding color value for creating NIR images to map each color value and create a false-color NIR image.

In this chapter, we first describe a MobileNet deep learning model which was used for segmenting wound regions. In addition, we describe the transfer learning and post-processing methods which were used for getting more accurate results. In addition, we analyze the histogram and generate a K-mean clustering to predict what colors are appear the most in the wound area in the corresponding NIR images to track a wound healing process.

4.2 Dataset

The dataset was provided by Dr. Jeff Niezgoda from Advancing the Zenith of Healthcare (AZH) Wound and Vascular Center in the Milwaukee, WI. The dataset contains 47 wound images at a resolution of 960 by 640 from 4 different patients with corresponding NIR images. Figure 21 shows sample images from the dataset. These images were taken over the time from the same wound area to analyze the progress of wound healing.

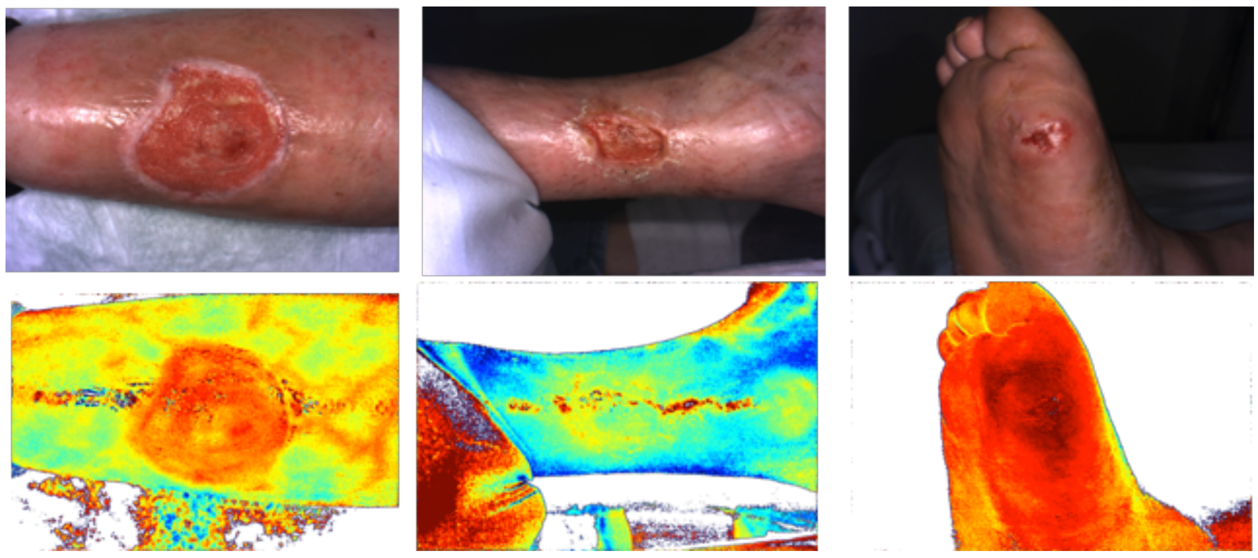


Figure 21: Sample wound dataset

4.3 Preprocessing

4.3.1 Data Annotation:

Unlike traditional machine learning methods, deep learning uses the raw image data to extract features and learn the correlation between input and output and find the higher knowledge to segment regions of interest. Hence, the quality and accuracy of the ground truth data for training and testing is essential. As we discussed in chapter 2, automated algorithms are not precise, therefore, manual annotation is the best choice to capture the ground truth. We used Apeer [21] to annotate the wound area in our dataset.

4.3.2 Data Augmentation and Data Splitting:

Having a large dataset is essential for obtaining more accurate predictions with a deep learning model. Since our dataset contained only 47 images, we used a data augmentation method to expand the number of images in the dataset. In order to obtain new data and expand our dataset, we only needed to make minimal changes to our existing dataset. Flips, translations, and rotations are examples of the minor modifications that we applied to the images. Our deep learning model recognized these augmented images as different pictures. We generated sixteen images from each image and the number of images in our dataset was increased from 47 to 756.

We separated the augmented data into 2 different training and testing datasets with 604 and 151 images, respectively.

4.4 Deep Learning Model – MobileNetV2

In this project, the CNN model MobileNetV2 was used to train and segment the wound area in images. Most CNNs have a deep structure that requires many computational resources like numerous, parallel CPU cores or a GPU. Since AlexNet [38] made deep CNNs popular by winning the 2012 ImageNet Challenge: ILSVRC [40], convolutional neural networks have become widespread in computer vision. In order to achieve better accuracy, the overall trend since 2012 has been to create deeper and more complex networks. However, these advancements in accuracy do not mean that networks have become more efficient in terms of size and speed. Computer vision tasks must be completed instantaneously on a computationally restricted platform in many real-world applications.

MobileNetV2 is a light model with only two hyperparameters that can be used on mobile hardware. The number of parameters considerably lowers compared to the regular CNNs with the same network depth, and this leads to lightweight neural networks.

The MobileNets model is built on depthwise separable convolutions. The depthwise separable convolution breaks down into two parts. The first part is the depthwise convolution, which performs as the filtering stage, and the second part is pointwise convolution, which performs as a combining stage. In the first part, unlike the standard convolution which applies the convolution to all channels, the depthwise convolution used by MobileNets applies a single filter to each input channel. After that, the pointwise convolution uses convolution to combine the depthwise convolution's outputs. This factorization that depthwise separable convolution use reduces computing time and model size significantly. In MobileNet, Except for the first layer, which is a complete convolution, the structure uses depthwise separable convolutions.

The MobileNet model use ReLU6($y = min(max(0, x),6)$) as the activation function after all batchnorm layers except the last layer which is use Softmax for classification. As the figure 22 shows, ReLU6 is similar to the well-known ReLU that we describe in chapter 2, except it limits larger activations.

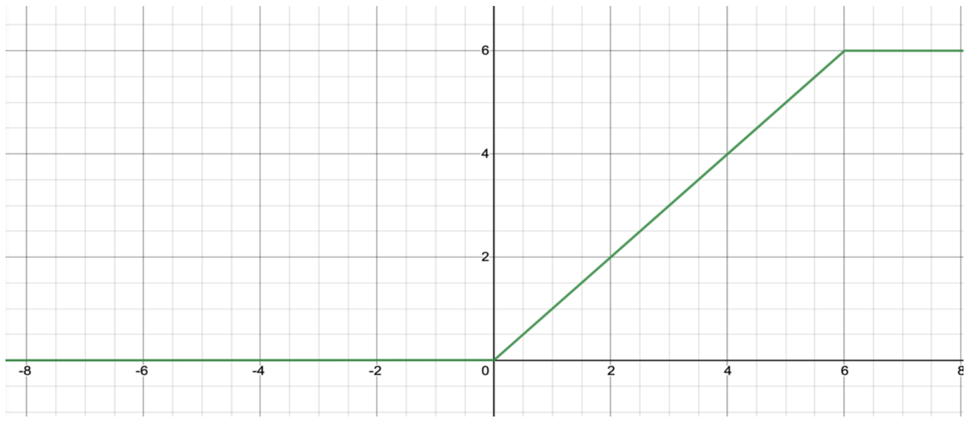


FIGURE 22: RELU6 ACTIVATION FUNCTION

The MobileNetV2 bottleneck Residual block structure and is illustrated in figure 23. This block contains three different layers: Expansion, Convolutional layer feature extraction, and compression. To map low-dimensional space to high-dimensional space, the expansion layer employs a $1X1$ convolution kernel. The projection layer comes after the depth separable convolution layer, and a 1X1 convolution kernel is utilized to remap high-dimensional features to low-dimensional space.

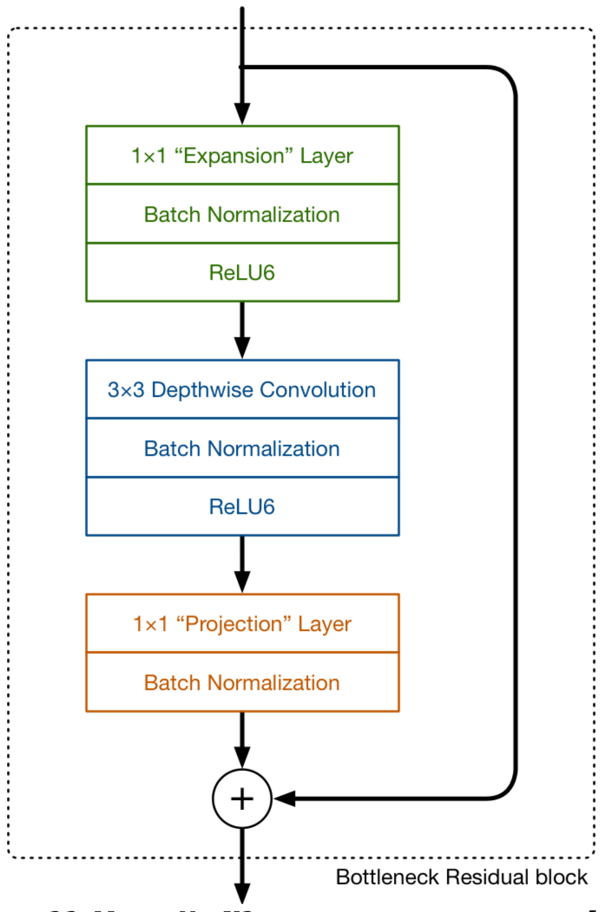


FIGURE 23: MOBILENETV2 BOTTLENECK RESIDUAL BLOCK [41]

The MobileNet balances model accuracy with performance; hence it performs great on devices with limited computational resources, such as smartphones and tablets. For example, medical professionals and patients may take a photo of a wound with any smartphone and instantly receive wound segmentation and wound area measurements which could help wound healing management.

4.4.1 Transfer Learning

Transfer learning is a strategy that re-uses a trained model created for a task as the starting point for a model for a second task. Machine learning models guess a number for the weights that they want to learn at the start point of the learning process. Transfer learning improves in preventing guesswork and obtaining a rough estimate of parameter values. This assists the model to find the actual weight values faster. In this project, we use a pre-trained model which was used for the chronic wound segmentation in 1,109 foot ulcer images. [35]

4.5 Implementation and Results of Wound Segmentation

The code was developed on a MacBook Pro with a 2.2GHz Intel i7 quad-core processor, 16 GB of 1600 DDR3 memory, and Intel Iris Pro 1535 Megabyte graphics. Since the machine lacks a GPU, the code is executed using the CPU. For Learning Process, we use a batch size of 2 and 2000 epoch with a learning rate of 1e-4, and we use Adam optimizer [42] to adapt the learning rate. In addition, for loss function since we have a binary classification, we use binary cross entropy. In addition, we use early stop technique to reduce overfitting. The early stop tracks the Validation F1-score and if it does not change after 200 epochs it stops the learning process. Training time of each epoch took around 90 second. The code was developed based on Python and Keras [27] package. Finally, after around 440 epochs, our deep learning model had been trained. After training the model, we obtain 89.8% F1-Score, 89.08% precision, and 91.4% recall in the testing dataset.

4.5.1Postprocessing

For getting a better result in our segmentation result, we use two different postprocessing method. The first method is the hole filling and small regions removal, which is performed in the same way as in [35]. In this technique, we look for connected components that are less than a threshold value that is determined adaptively based on the total number of pixels segmented as wound pixels. Finally, we remove such a connected component to increase the true positive rate in the segmentation model. [35]

The second method is Test-Time Augmentation (TTA). TTA is the same as data augmentation that we did for training step. For prediction, we transform the original image to 4 different images by flipping the image into right to left, up to down, and rotating by 90 degrees.

Afterward, we segmented all the augmented images with our model, after that, we undo the transformation (DE augmentation) for all segmented results. Moreover, we do the hole filling and small regions removal for all the predicted masks. Then we add all the mask and averaged all the segmented images to get the final result. The figure shows the diagram of how TTA works. In this figure, the average result without thresholding is shown. For getting a clear and final result we set a threshold of 0.3 to get the final output which we are going to use for prediction result.

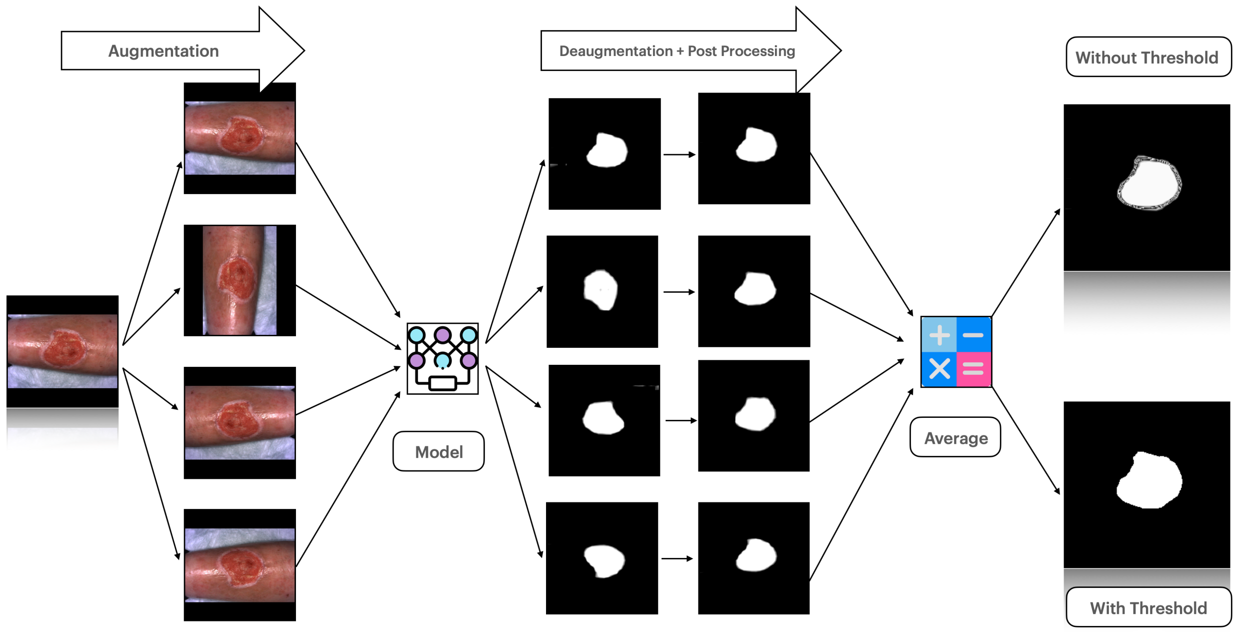


FIGURE 24: TTA. THIS IS A TTA DIAGRAM THAT SHOWS A PROCESS OF IMPROVING THE MODEL SEGMENTATION RESULT.

4.6 Analyze the Wound Area in NIR Images

NIR images provide a vital role in wound healing analyses. However, just by looking at the image itself we may not be able to find the wound area and analyze it. Hence, after segmenting the regular wound images with our model, we applied the masks into the NIR images to capture the wound area.

Afterward, we generated a histogram of NIR images to analyze pixel intensity values across the different RGB channels. The colors indicate the percentage of Sto2 in the NIR images. Hence, by analyzing both the colors and the histogram, we can extract more information from these images and observe the wound healing process. As shown in figure 25, the dark blue color is the lowest amount of Sto2, and the dark red is the highest. This figure has two wound images from the same patient. The left image is the oldest, and the right image is the newest. By looking at the image in the wound area, we can acquire information about the level of Sto2. In this example, the left image shows that the Sto2 levels are around 60 to 80 percent, while the right image shows the Sto2 levels are around 80 to 100 percent. Therefore, the results indicate that the Sto2 level increased over time, hence, the wound is healing.

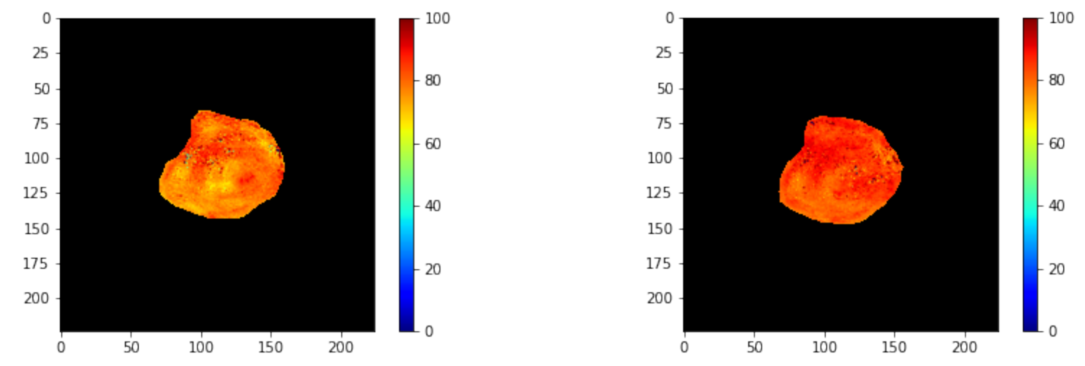


FIGURE 25: SEGMENTED NIR IMAGES SAMPLE FOR ANALYZING

We sorted 4 photos from oldest to newest of the same wound area image that have been taken from the same patient (see figure 26). We segmented the wound area in these images using our segmentation model. After segmentation, we applied the mask in the NIR image to capture the wound area. In figure 26 the top row is the NIR image, and the second row is the segmented wound. The leftmost image is the oldest and the rightmost image is the newest. We analyzed each masked image to track the healing process by plotting the histogram channels of each image and comparing their pixel intensity value.

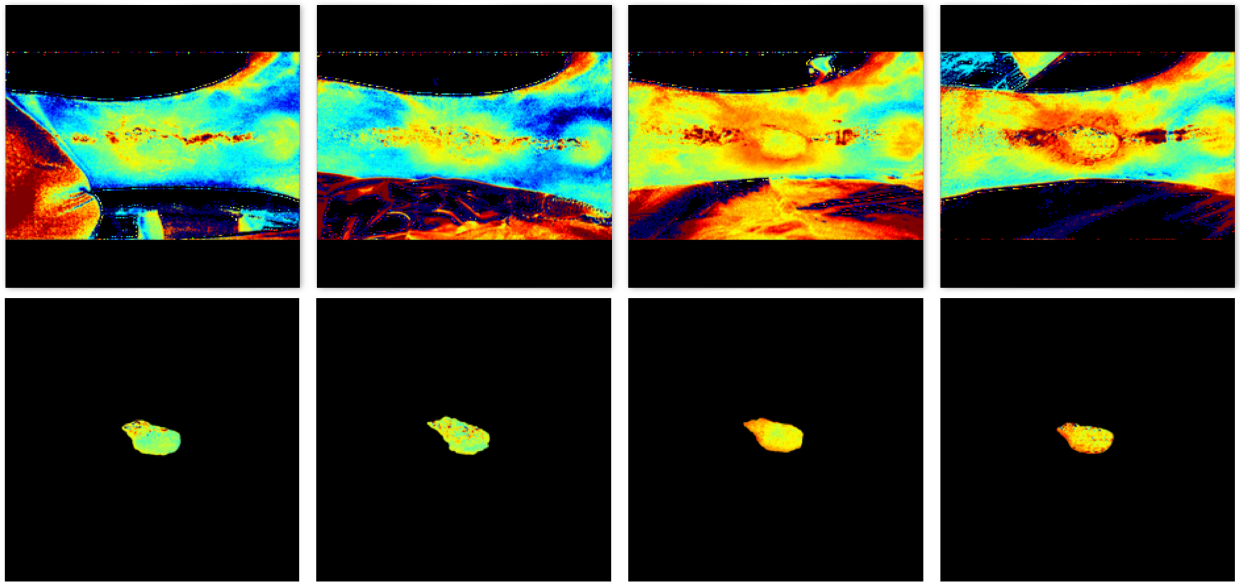


FIGURE 26: NIR MASKED IMAGES. THIS FIGURE SHOWS 4 IMAGES FROM THE SAME WOUND AREA OVER THE TIME. IT SORTED FROM LEFT TO RIGHT WHERE THE LEFT ONE IS THE OLDEST AND THE RIGHT ONE IS THE NEWEST. THE FIRST ROW THE WOUND CORRESPONDING NIR IMAGE, AND THE SECOND ROW IS THE MASKED NIR.

Figure 27 shows the histogram of RGB channels of all images where the first row is the oldest and the fourth row is the newest. The histogram shows the different pixel values on the x-axis from 0 to 255 where 0 is the darkest (black) pixel value and 255 is the lightest (white) pixel value. The y-axis marks the intensity of each pixel value. The distribution of the pixel values in the first blue and red histograms are relatively symmetrical while the distribution for the first green histogram is skewed left, and all the histograms are unimodal. In this row, the green histogram shows brighter pixel values than the blue and red. The blue channel has the greatest concentration of dark pixel values while the red channel is mostly concentrated among lighter pixel values. Over time, the red channel shows a greater concentration of lighter and brighter pixel values, the blue channel shows a greater concentration of darker pixel values, and the green channel covers a wider range of dark and light pixel values. These changes in the histograms over time indicate that the images display more red color and less blue color signifying increased Sto2 levels in the wound area.

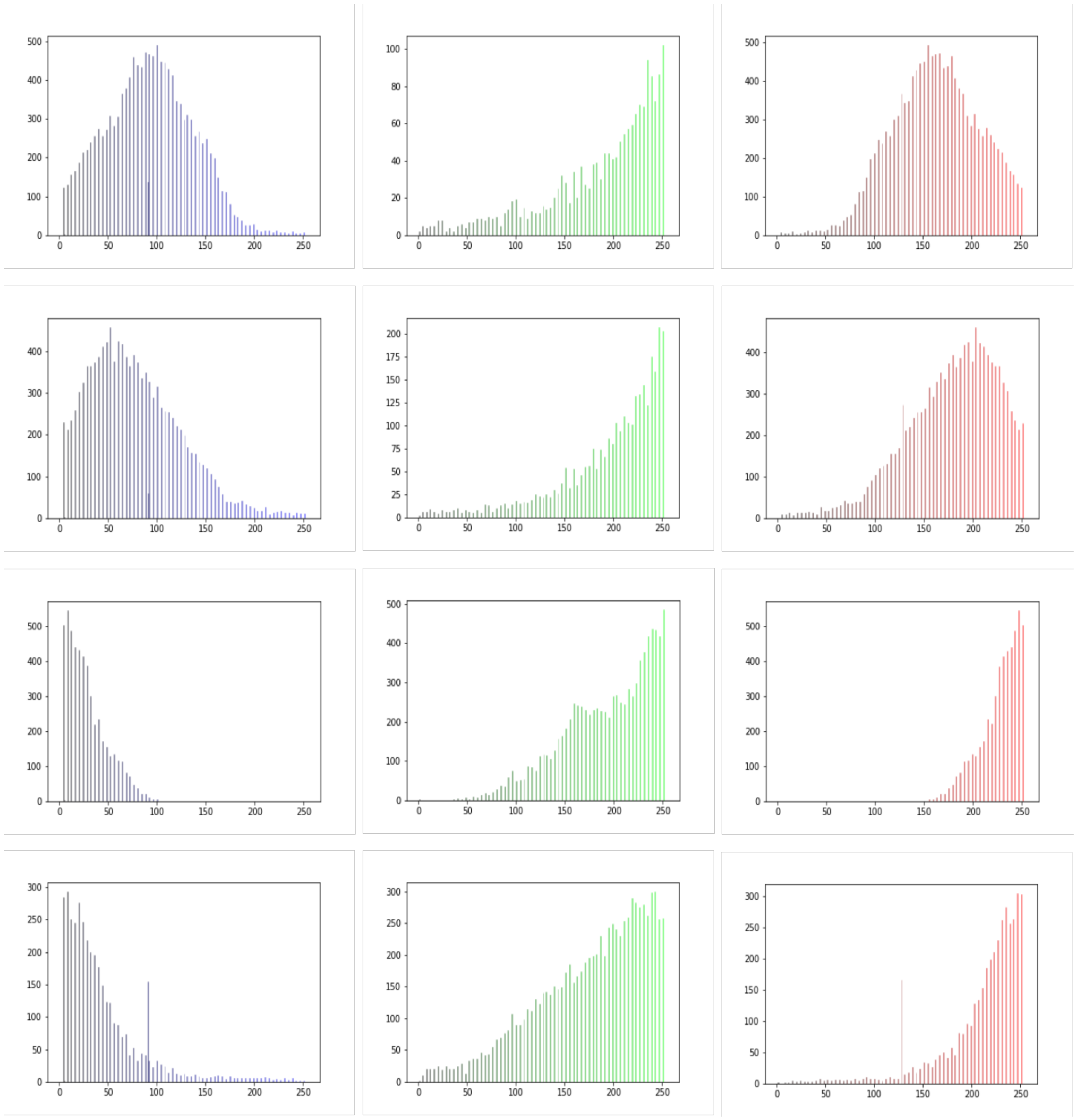


Figure 27: Histogram of four masked image. The first row is the oldest image and the last row is the newest image. Each color represent the corresponding Channel

Figure 28 represents the most common colors in the oldest and newest NIR images from figure 26. Since the colors represent the percentage of Sto2 in the NIR images, we utilized K-Mean clustering to predict the most common colors. We set the K-mean clustering to group the colors into six different clusters. Next, we used K-Mean clustering to predict which color appears most frequently. The results of our K-Mean clustering for color extraction are shown in figure 28. In this figure, each pie chart illustrates the colors that appear most in the image. The left chart, which represents the first NIR image, shows mostly green color tones, while the right chart shows mostly orange and yellow color tones. Green tones indicate Sto2 levels ranging from 40 to 60 percent, and the orange tones indicate Sto2 levels from 60 to 80 percent. Hence, we can conclude that in image four, the levels of Sto2 have increased over time.

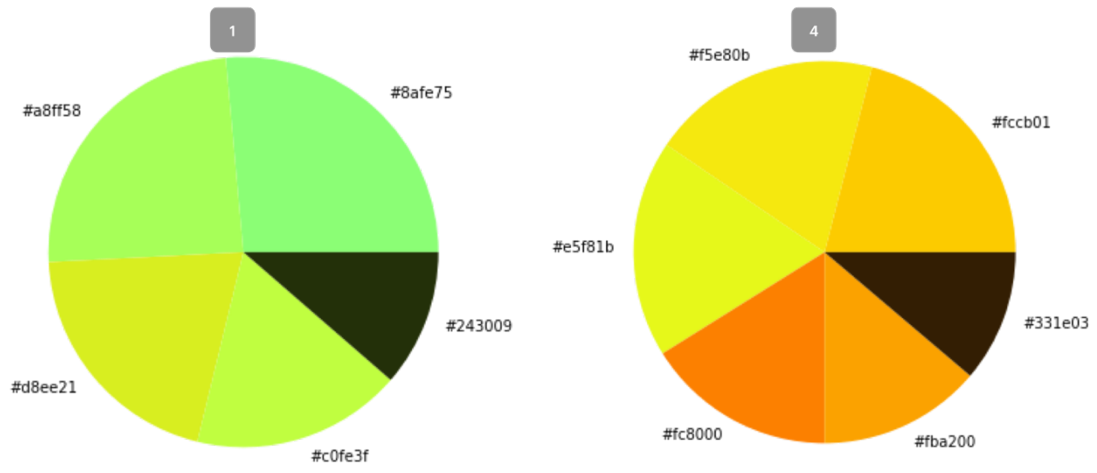


FIGURE28: COLOR ANALYSIS FOR NIR IMAGES

We discovered that the Sto2 levels increase throughout time, which is necessary for producing new cells and aiding in bacterial defense for optimal healing. This vital information could be used to improve wound care by allowing physicians to better track the wound healing process, resulting in better wound treatment.

4.7 Conclusion

In this chapter, we described and generated a deep learning model to segment a wound region in photographs. We used MobileNetV2 to train a model to segment the wound area in these images. We generated a model which improved and balanced the efficiency and training speed. With only 750 images, we obtained a model with an f1-score of 89.8%, which is an impressive result considering similar models require at least twice as many images. Next, we used two post processing techniques to improve the model's prediction leading to a better segmentation result: 1. hole filling and small noise removal, and 2. test time augmentation.

We used these segmentation results to find the wound area in the NIR images and capture them for further analysis. We used a histogram to analyze and compare RGB channels to see the changes in pixel value intensity. Then we generated a K-Mean clustering model to predict the colors that appear the most in the images. These results provide information about changes in blood oxygenation saturation levels which could help healthcare providers to track the healing process and develop a better wound treatment plan.

CHAPTER FIVE

Conclusion

In this thesis, we attempted to solve two different medical image segmentation problems using Machine Learning approaches. Chapter three presents an image segmentation approach that uses the knowledge gained during the training phase of Random Forest classifiers for CT-scan segmentation. We described a feature extraction method for preprocessing the data for training. In addition to learning discriminative features, Random Forest classifiers allow us to analyze the function and contribution of various features. We can therefore evaluate which features the model uses to classify each pixel in an image. We eliminated features that had a minor influence on classification accuracy, such as a handful of Gabor filters. This allows us to obtain a better generalization of Random Forest. As a result, we obtain a model with a 97.4 percent accuracy.

In chapter four, wound segmentation, we presented a wound image segmentation approach that uses a MobileNetV2 Deep learning structure. We described how the MobileNet structure would help us to reduce computational complexity. Although we had a small amount of data, by using transfer learning and data augmentation techniques, we achieve a model with an f1-score of 89.8 percent. Afterward, we produce two different post-processing techniques which improved the model outcome. Next, we applied these masks to NIR images that provide vital information about the Sto2 level of wound tissue. In NIR images, colors represent the percentage of Sto2; therefore, we used histogram and color analysis to capture more information to track the wound healing process. Future studies may include the use of a wider range of wound samples to train the model to get better accuracy and track the different stage of wound healing process in NIR images. The result of our study could help the physician to have a better wound treatment plan.

We could use the MobileNet model that we trained to develop a mobile application that will benefit patients using their mobile phones for future works. Patients or omissions could use the camera directly to segment the wound area, create False-Color-NIR images, and provide the analyzing tool for both specialists and patients to track the wound healing process. This application could reduce the cost of treatment and make the wound healing tracking process more convenient.

REFRENCES

[1] S. Minaee, Y. Boykov, F. Porikli, A. Plaza, N. Kehtarnavaz, and D. Terzopoulos., "Image Segmentation Using Deep Learning: A Survey," arXiv.org, arXiv:2001.05566v1, 2020. https://arxiv.org/abs/2001.05566v1. (accessed: Jul. 13, 2021).

[2] M. H. Hesamian, W. Jia, X. He, and P. Kennedy, "Deep learning techniques for medical image segmentation: achievements and challenges," Journal of Digital Imaging, vol. 32, no. 4, pp. 582–596, May 2019, doi: 10.1007/s10278-019-00227-x.

[3] H. McGrath et al., "Manual segmentation versus semi-automated segmentation for quantifying vestibular schwannoma volume on MRI," International Journal of Computer Assisted Radiology and Surgery, vol. 15, no. 9, pp. 1445–1455, Jul. 2020, doi: 10.1007/s11548-020-02222-y.

[4] P. A. Yushkevich et al., "User-guided 3D active contour segmentation of anatomical structures: Significantly improved efficiency and reliability," NeuroImage, vol. 31, no. 3, pp. 1116–1128, Jul. 2006, doi: 10.1016/j.neuroimage.2006.01.015.

[5] Al-amri, Salem Saleh, K. N. V, and D., Khamitkar S, "Image Segmentation by Using Threshold Techniques," arXiv.org, 2021. https://arxiv.org/abs/1005.4020v1 (accessed Jul. 14, 2021).

[6] N. Sharma et al., "Automated medical image segmentation techniques," Journal of Medical Physics, vol. 35, no. 1, p. 3, 2010, doi: 10.4103/0971-6203.58777.

[7] P. Bao, Lei Zhang, and Xiaolin Wu, "Canny edge detection enhancement by scale multiplication," IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 27, no. 9, pp. 1485–1490, Sep. 2005, doi: 10.1109/tpami.2005.173.

[8] H. G. Kaganami and Z. Beiji, "Region-Based Segmentation versus Edge Detection," 2009 Fifth International Conference on Intelligent Information Hiding and Multimedia Signal Processing, Sep. 2009, doi: 10.1109/iih-msp.2009.13.

[9] R. Adams and L. Bischof, "Seeded region growing," IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 16, no. 6, pp. 641–647, Jun. 1994, doi: 10.1109/34.295913.

[10] Contributors to Wikimedia projects, "Nervous system cells," Wikipedia.org, May 30, 2004. https://simple.wikipedia.org/wiki/Neuron (accessed Jul. 14, 2021).

[11] M. Anthony and P. L. Bartlett, Neural network learning: theoretical foundations. Cambridge: Cambridge University Press, Feb, 2010.

[12] T. Wiederer ,"Neural Networks in Javascript - webkid blog," Webkid.io, 2016. https://webkid.io/blog/neural-networks-in-javascript/ (accessed Jul. 14, 2021).

[13] M. Minsky and S. A. Papert, Perceptrons. The MIT Press, 2017.

[14] S. Sharma, S. Sharma. "Activation functions in neural networks." Towards Data Science. Sep, 2017, pp. 310-316.

[15] "Activation Function," AI Wiki, 2021. https://docs.paperspace.com/machine-learning/wiki/activation-function (accessed Jul. 14, 2021).

[16] S. Albawi, T. A. Mohammed, and S. Al-Zawi, "Understanding of a convolutional neural network," 2017 International Conference on Engineering and Technology (ICET), Aug. 2017, doi: 10.1109/icengtechnol.2017.8308186.

[17] K. O'Shea and R. Nash, "An Introduction to Convolutional Neural Networks," arXiv.org, 2015. https://arxiv.org/abs/1511.08458v2 (accessed Jul. 14, 2021).

[18] Chaim Baskin, Natan Liss, Avi Mendelson, and Evgenii Zheltonozhskii, "Streaming Architecture for Large-Scale Quantized Neural Networks on an FPGA-Based Dataflow Platform," ResearchGate, Jul. 31, 2017. https://www.researchgate.net/publication/318849314_Streaming_Architecture_for_Large-Scale_Quantized_Neural_Networks_on_an_FPGA-Based_Dataflow_Platform (accessed Jul. 14, 2021).

[19] N. Sharma et al., "Automated medical image segmentation techniques," Journal of Medical Physics, vol. 35, no. 1, p. 3, 2010, doi: 10.4103/0971-6203.58777.

[20] W. Piao, Y. Yuan, and H. Lin, "A Digital Image Denoising Algorithm Based on Gaussian Filtering and Bilateral Filtering," ITM Web of Conferences, vol. 17, p. 01006, 2018, doi: 10.1051/itmconf/20181701006.

[21] APEER ,"APEER – from image to information," APEER, 2021. https://www.apeer.com/app (accessed Jul. 14, 2021).

[22] D. Mahapatra, "Analyzing Training Information from Random Forests for Improved Image Segmentation," IEEE Transactions on Image Processing, vol. 23, no. 4, pp. 1504–1512, Apr. 2014, doi: 10.1109/tip.2014.2305073.

[23] H. Guo et al., "Gait Recognition Based on the Feature Extraction of Gabor Filter and Linear Discriminant Analysis and Improved Local Coupled Extreme Learning Machine," Mathematical Problems in Engineering, vol. 2020, pp. 1–9, Apr. 2020, doi: 10.1155/2020/5393058.

[24] J. M. S. Prewitt, "Object enhancement and extraction," Picture Processing and Psychopictorics, B. Lipkin and A. Rosenfeld, Eds., New York Academic Press, 1970,

pp.75-149.

[25] Python, “Welcome to Python.org,” Python.org, Jul. 12, 2021. https://www.python.org/ (accessed Jul. 14, 2021).

[26] Scikit-Learn, “Scikit-learn machine learning in Python — scikit-learn 0.24.2 documentation,” Scikit-learn.org, 2021. https://scikit-learn.org/stable/ (accessed Jul. 14, 2021).

[27] Keras Team, “Keras: the Python deep learning API,” Keras.io, 2021. https://keras.io/ (accessed Jul. 14, 2021).

[28] S. Ruder, “An overview of gradient descent optimization algorithms,” arXiv.org, 2016. https://arxiv.org/abs/1609.04747 (accessed Jul. 14, 2021).

[29] Y.-Y. Song and Y. Lu, “Decision tree methods: applications for classification and prediction,” Shanghai Archives of Psychiatry, vol. 27, no. 2, pp. 130–5, 2015, doi: 10.11919/j.issn.1002-0829.215044.

[30] S. Enoch and P. Price, “Cellular, molecular and biochemical differences in the

pathophysiology of healing between acute wounds, chronic wounds and wounds in the aged,” World Wide Wounds 13, 2004, pp.1–17.

[31] S. Dhivya, V. V. Padma, and E. Santhini, “Wound dressings – a review,” *BioMedicine*, vol. 5, no. 4, Nov. 2015, doi: 10.7603/s40681-015-0022-9.

[32] L. Gould *et al.*, “Chronic Wound Repair and Healing in Older Adults: Current Status and Future Research,” *Journal of the American Geriatrics Society*, vol. 63, no. 3, pp. 427–438, Mar. 2015, doi: 10.1111/jgs.13332.

[33] F. Li, C. Wang, X. Liu, Y. Peng, and S. Jin, “A Composite Model of Wound Segmentation Based on Traditional Methods and Deep Neural Networks,” *Computational Intelligence and Neuroscience*, vol. 2018, pp. 1–12, May 2018, doi: 10.1155/2018/4149103.

[34] C. K. Sen, “Human Wounds and Its Burden: An Updated Compendium of Estimates,” *Advances in Wound Care*, vol. 8, no. 2, pp. 39–48, Feb. 2019, doi: 10.1089/wound.2019.0946.

[35] C. Wang *et al.*, “Fully automatic wound segmentation with deep convolutional neural networks,” *Scientific Reports*, vol. 10, no. 1, Dec. 2020, doi: 10.1038/s41598-020-78799-w.

[36] S. P. Philimon, A. K. C. Huong, P. E. Ong, and X. T. I. Ngu, “Multispectral imaging system for clinical assessment of superficial wound tissue oxygenation,” *2016 IEEE*

EMBS Conference on Biomedical Engineering and Sciences (IECBES), Dec. 2016, doi: 10.1109/iecbes.2016.7843405.

[37] M. G. Sowa, "SnapshotNIR: a handheld multispectral imaging system for tissue viability assessment," *Photonics and Education in Measurement Science 2019*, Sep. 2019, doi: 10.1117/12.2534989.

[38] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet classification with deep convolutional neural networks," *Communications of the ACM*, vol. 60, no. 6, pp. 84–90, May 2017, doi: 10.1145/3065386.

[39] A. G. Howard *et al.*, "MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications," *arXiv.org*, 2017. https://arxiv.org/abs/1704.04861 (accessed Jul. 19, 2021).

[40] O. Russakovsky *et al.*, "ImageNet Large Scale Visual Recognition Challenge," *International Journal of Computer Vision*, vol. 115, no. 3, pp. 211–252, Apr. 2015, doi: 10.1007/s11263-015-0816-y.

[41] Matthijs Hollemans, "MobileNet version 2," *Machinethink.net*, 2018. https://machinethink.net/blog/mobilenet-v2/ (accessed Jul. 20, 2021).

[42] Kingma, Diederik P and J. Ba, "Adam: A Method for Stochastic Optimization," *arXiv.org*, 2014. https://arxiv.org/abs/1412.6980 (accessed Jul. 27, 2021).

[43] Imdad, Ulfat & Asif, Muhammad & Ahmad, Tahir & Sohaib, Osama & Hanif, Muhammad & Chaudary, Muhammad. (2019). Three Dimensional Point Cloud Compression and Decompression Using Polynomials of Degree One. Symmetry. 11. 209. 10.3390/sym9030614.

[44] K. Recommended Citation Kirasich, T. Smith, and B. Sadler, "Random Forest vs Logistic Regression: Binary Classification for Heterogeneous Datasets," *SMU Data Science Review*, vol. 1, no. 3, p. 9, 2018, [Online]. Available: https://scholar.smu.edu/cgi/viewcontent.cgi?article=1041&context=datasciencereview.

[45] Rajath Soans, D. C. Lim, B. T. Keenan, and J. Shackleford, "Automated Protein Localization of Blood Brain Barrier Vasculature in Brightfield IHC Images," *ResearchGate*, Feb. 2016. https://www.researchgate.net/publication/292671765_Automated_Protein_Localization_of_Blood_Brain_Barrier_Vasculature_in_Brightfield_IHC_Images (accessed Aug. 05, 2021).

[46] "Max-pooling / Pooling - Computer Science Wiki," *Computersciencewiki.org*, 2018. https://computersciencewiki.org/index.php/Max-pooling_/_Pooling (accessed Aug. 05, 2021).

[47] R. Yamashita, M. Nishio, R. K. G. Do, and K. Togashi, “Convolutional neural networks: an overview and application in radiology,” *Insights into Imaging*, vol. 9, no. 4, pp. 611–629, Jun. 2018, doi: 10.1007/s13244-018-0639-9.

[48] Prabhu, “Understanding of Convolutional Neural Network (CNN) — Deep Learning,” *Medium*, Mar. 04, 2018. https://medium.com/@RaghavPrabhu/understanding-of-convolutional-neural-network-cnn-deep-learning-99760835f148 (accessed Aug. 05, 2021).

[49] “RGB channels separation.png - Wikimedia Commons,” *Wikimedia.org*, Mar. 2008. https://commons.wikimedia.org/wiki/File:RGB_channels_separation.png (accessed Aug. 05, 2021).

[50] Yali Nie, “A Multi-stage Convolution Machine with Scaling and Dilation for Human Pose Estimation 사람 자세 추정을 위한 스케일링 및...,” *ResearchGate*, Feb. 22, 2018. https://www.researchgate.net/publication/326531654_A_Multi-stage_Convolution_Machine_with_Scaling_and_Dilation_for_Human_Pose_Estimation_salam_jase_chujeong-eul_wihan_seukeilling_mich_hwagjang_giban_dadan_konbollusyeon_meosin (accessed Aug. 05, 2021).

[51] “Single-Layer Neural Networks and Gradient Descent,” *Dr. Sebastian Raschka*, Mar. 24, 2015. https://sebastianraschka.com/Articles/2015_singlelayer_neurons.html (accessed Aug. 05, 2021).

[52] IBM Cloud Education, “What is Gradient Descent?,” *Ibm.com*, Oct. 27, 2020. https://www.ibm.com/cloud/learn/gradient-descent (accessed Aug. 05, 2021).