

BREAST CANCER CLASSIFICATION AND SEGMENTATION USING ARTIFICIAL INTELLIGENCE TECHNIQUE

Submitted in partial fulfillment of the requirements for the award of
Master of Science Degree in Bioinformatics and Data Science

By

V VANMATHI (42739027)



DEPARTMENT OF BIOINFORMATICS
SCHOOL OF SCIENCE AND HUMANITIES

SATHYABAMA

INSTITUTE OF SCIENCE AND TECHNOLOGY
(DEEMED TO BE UNIVERSITY)

Accredited with Grade "A++" by NAAC | 12B Status UGC | Approved by AICTE
Jeppiaar Nagar, Rajiv Gandhi Salai, Chennai – 600 119

APRIL – 2024



SATHYABAMA

INSTITUTE OF SCIENCE AND TECHNOLOGY

(DEEMED TO BE UNIVERSITY)

Accredited with "A++" grade by NAAC | 12B Status by UGC | Approved by AICTE

Jeppiaar Nagar, Rajiv Gandhi Salai, Chennai – 600 119

www.sathyabama.ac.in

DEPARTMENT OF BIOINFORMATICS

BONAFIDE CERTIFICATE

This is to certify that this Project Report is the bonafide work of **V VANMATHI (42739027)**, who carried out the project entitled "**Breast Cancer Classification and Segmentation using Artificial Intelligence Technique**" under our supervision from September 2023 to April 2024.

A handwritten signature in blue ink, appearing to read 'Daniel Alex Anand'.

Dr. DANIEL ALEX ANAND

Internal Guide

A handwritten signature in blue ink, appearing to read 'H. Jemmy Christy'.

Dr. H.JEMMY CHRISTY

HEAD OF THE DEPARTMENT

Submitted for Viva Voce Examination held on **20/04/2024**

A handwritten signature in blue ink, appearing to read 'Internal Examiner'.

Internal Examiner

A handwritten signature in blue ink, appearing to read 'External Examiner'.

External Examiner

DECLARATION

I, **V Vanmathi (42739027)**, hereby declare that the Project Report entitled **Breast Cancer Classification and Segmentation using Artificial Intelligence Technique** done by me under the guidance of **Dr. Daniel Alex Anand** at **Sathyabama Institute of Science and Technology, Chennai**, and is submitted in partial fulfillment of the requirements for the award of Master of Science in Bioinformatics and Data Science.

DATE: 20/04/2024

PLACE: CHENNAI



SIGNATURE OF THE CANDIDATE

ACKNOWLEDGEMENTS

I am pleased to acknowledge my sincere thanks to **the Board of Management, SATHYABAMA INSTITUTE OF SCIENCE AND TECHNOLOGY** for their kind encouragement in doing this project and for completing it successfully. I am grateful to them.

I convey my thanks to **Dr. REKHA CHAKRAVATHY.**, Dean, School of Science and Humanities, and **Dr. Jemmy Christy**, Head of the Bioinformatics Department, for providing me the necessary support and details at the right time during the progressive reviews

I would like to express my sincere and deep sense of gratitude to my Project Guide **Dr.Daniel Alex Anand** for his valuable guidance, suggestions, and constant encouragement paved the way for the successful completion of my project work.

I wish to express my thanks to all Teaching and Non-teaching staff members of the Department of Bioinformatics who were helpful in many ways for the completion of the project.

ABSTRACT

Breast cancer is a significant health concern affecting women worldwide, and early and accurate detection plays a crucial role in improving patient outcomes. Artificial intelligence (AI) techniques have shown great promise in breast cancer classification and segmentation tasks, providing automated and reliable solutions. This paper presents an overview of breast cancer classification and segmentation using AI techniques. The classification aspect focuses on distinguishing between benign and malignant breast lesions. Various AI algorithms, including machine learning and deep learning models, are explored for this task. The segmentation aspect addresses the precise delineation of breast lesions from the surrounding healthy tissue. AI techniques such as convolutional neural networks (CNNs) and image processing algorithms are utilized for accurate lesion segmentation. The integration of classification and segmentation techniques is explored, as it allows for a comprehensive analysis of breast cancer. Combined approaches enable accurate identification of cancerous regions within an image, aiding in precise diagnosis and treatment decisions.

Keywords: Breast cancer, Classification, Segmentation, Artificial intelligence (AI), Machine learning, Deep learning, Texture analysis, Shape analysis

TABLE OF CONTENT

CHAPTER	TITLE	PAGE NO
	ABSTRACT	I
	LIST OF FIGURES	IV
	LIST OF ABBREVIATIONS	IV
01.	INTRODUCTION	
1.1	BREAST CANCER	1
1.2	BACKGROUND	1
1.3	ARTIFICIAL INTELLIGENCE	2
1.4	DEEP LEARNING	2
1.5	PYTHON	3
02.	LITERATURE SURVEY	4
03.	AIM AND SCOPE	
3.1	AIM	8
3.2	SCOPE	8
04.	MATERIALS AND METHODOLOGY	
4.1	DATA COLLECTION AND DATABASE	9
4.2	IMPORT DATASET	10
4.3	DATA ANALYSIS	10
4.4	MANUAL ARCHITECTURE	11
4.5	LE-NET ARCHITECTURE	12
4.6	U-NET ARCHITECTURE	14
4.6.1	ENCODER AND DECODER	14
4.6.2	SKIP CONNECTION	14
4.6.3	CONTRACTING AND EXPANSIVE PATHS	15
4.6.4	FINAL LAYERS	15
4.6.5	LOSS FUNCTION	15

4.6.6	TRAINING DATA	15
4.6.7	INFERENCE	16
4.6.8	POST-PROCESSING	16
4.7	WORKFLOW DIAGRAM	17
4.8	DEPLOYMENT	17
4.8.1	DJANGO	17
4.8.2	HTML	19
4.8.3	CSS	20
05.	RESULT AND DISCUSSION	
5.1	RESULTS	24
5.2	DISCUSSION	26
5.2.1	SOURCE CODE	27
06.	SUMMARY AND CONCLUSION	
6.1	SUMMARY	40
6.2	CONCLUSION	40
	REFERENCES	41

LIST OF FIGURE

FIGURE NO	TITLE	PAGE. NO
4.1	BEGNIGN	9
4.2	MALIGNANT	9
4.3	NORMAL	10
4.4	WORKFLOW	17
5.1	HOME PAGE	24
5.2	LOGING SITE	24
5.3	WEBSITE	25
5.4	RESULT	25

LIST OF ABBRIVIATION

AI	ARTIFICIAL INTELLIGENCE
DCE-MRI	DYNAMIC CONTRAST ENHANCED-MAGNETIC RESONANCE IMAGING
CNN	CONVOLUTIONAL NURAL NETWORK
RNN	RECURRENT NEURAL NETWORK
TPUs	TENSORFLOW PROCESSING UNITS
CSS	CASCADING STYLE SHEETS
AUC	AREA UNDER THE CURVE
HTML	HYPertext MARKUP LANGUAGE
XML	EXTENSIBLE MARKUP LANGUAGE

CHAPTER 1

INTRODUCTION

1.1 BREAST CANCER

Cancer is one of the most fatal diseases that leads millions of lives to death every year. Breast cancer is the most prevalent type of cancer and the biggest cause of mortality among women.(Z. Anastasiadi et al 2017) Breast cancer classification and segmentation using artificial intelligence techniques is a critical application of machine learning and computer vision in the medical field. By harnessing advanced algorithms and deep learning models, this approach aims to enhance the accuracy and efficiency of diagnosing and categorizing breast cancer tumors from medical images, such as mammograms and histopathological slides. Through the automated analysis of these images, AI systems can assist healthcare professionals in early detection, classification, and segmentation of breast cancer, enabling more timely and precise treatment decisions, ultimately contributing to improved patient outcomes and reducing the burden of this life-threatening disease.(Caloro E et al 2022)

1.2 BACKGROUND

Breast cancer is a heterogeneous disease consisting of a diverse set of genomic mutations and clinical characteristics. The molecular subtypes of breast cancer are closely tied to prognosis and treatment options. We investigate using deep graph learning on a collection of patient factors from multiple diagnostic disciplines to better represent breast cancer patient information and predict molecular subtypes. Our method models breast cancer patient data into a multi-relational directed graph with extracted feature embeddings to represent patient information and diagnostic test results directly.

We develop a radiographic image feature extraction pipeline to produce vector representation of breast cancer tumors in DCE-MRI and an autoencoder-based genomic variant embedding method to map variant assay results to a low-dimensional latent space. We leverage related-domain transfer learning to train and evaluate a Relational Graph Convolutional Network to predict the probabilities of molecular subtypes for individual breast cancer patient graphs(Furtney I et al 2023).

1.3 ARTIFICIAL INTELLIGENCE:

AI refers to science and engineering that explores how to simulate various issues and functions in the field of human intelligence. AI technology fields cover perception, recognition, reasoning, the learning process, natural language, machine translation, games, chess, and so on. From Apple's Siri to self-driving cars, AI is progressing rapidly. While science fiction often portrays AI as robots with human-like characteristics, AI can encompass anything from Google's search algorithms to IBM's Watson to autonomous weapons. AI is a major part of many cutting-edge technologies, including robotics, driverless cars, web searches, and video games. AI technologies use sophisticated algorithms, or sets of instructions, to solve very difficult tasks (Ida Arlene et al 2018)

1.4 DEEP LEARNING

Deep learning is one of the fastest growing areas of data science. Deep learning refers to a class of algorithms which are based on artificial neural networks optimized to work with unstructured data such as images, voice, videos and text. While the techniques of deep learning date back to the mid-80s, their true potential has been realized only in the last 5 years or so.(Vijay Kotu et al 2018)

The goal of developing artificial intelligence computing systems is to enable machines to better understand and serve humans. Machines understand vision and speech mainly based on deep learning. Deep learning refers to large-scale deep neural networks. The principle and development of typical CNNs for image classification and object detection. Besides CNNs for image processing, we also introduce the recurrent neural network (RNN) and its extension, long short-term memory (LSTM) , for sequential data including speech, text, and video. The novel deep learning technology, generative adversarial networks , which has many interesting applications.(Yunji Chen et al 2024)

1.5 PYTHON

Python is an object oriented, scripted and interpreted language for both learning and real world programming. Python is a powerful high-level language created by Guido van Rossum. In this paper, we will provide an introduction to the main Python software tools used for Data science, Machine learning techniques and IOT. Briefly, this paper will first introduce Python as a language, and give introduction about Data science.(Jyothi et al 2019)

CHAPTER 2

REVIEW OF LITERATURE

Isaac Furtney et al.(2023) Breast cancer is a heterogeneous disease consisting of a diverse set of genomic mutations and clinical characteristics. The molecular subtypes of breast cancer are closely tied to prognosis and therapeutic treatment options. We investigate using deep graph learning on a collection of patient factors from multiple diagnostic disciplines to better represent breast cancer patient information and predict molecular subtype. Our method models breast cancer patient data into a multi-relational directed graph with extracted feature embeddings to directly represent patient information and diagnostic test results. We develop a radiographic image feature extraction pipeline to produce vector representation of breast cancer tumours in DCE-MRI and an autoencoder-based genomic variant embedding method to map variant assay results to a low-dimensional latent space. We leverage related-domain transfer learning to train and evaluate a Relational Graph Convolutional Network to predict the probabilities of molecular subtypes for individual breast cancer patient graphs. Our work found that utilizing information from multiple multimodal diagnostic disciplines improved the model's prediction results and produced more distinct learned feature representations for breast cancer patients. This research demonstrates the capabilities of graph neural networks and deep learning feature representation to perform multimodal data fusion and representation in the breast cancer domain.

Nikhilanand Arya et al.(2023) The advancement of medical research in the field of cancer prognosis and diagnosis using various modalities has put oncologists under tremendous stress. The complexity and heterogeneity involved in multiple modalities and their significantly varied clinical outcomes make it difficult to analyze the disease and provide the correct treatment. Breast cancer is the major concern among all cancers worldwide, specifically for females. To help oncologists and cancer patients, research for breast cancer survival estimation has been proposed. It ranges from complex deep neural networks to simple and interpretable architectures. We propose a utility kernel for a support vector machine (SVM) in this article. It is a simple yet powerful function, which performs better than other popular machine learning algorithms and deep neural networks in the task of breast cancer survival prediction using the TCGA-BRCA dataset. This study validates the proposed utility kernel using four different modalities (gene expression, copy number variation, clinical, and histopathological tissue images) and their multi-modal combinations. The SVM based on our utility kernel empirically proves its efficacy by achieving the highest value on various performance measures, whereas advanced deep neural networks fail to train on small and highly imbalanced breast cancer data

Cheng-Bang Chen et al. (2023) The histopathological image analysis is one of the most crucial diagnostic procedures to identify Invasive ductal carcinoma (IDC) in breast cancers. However, this diagnosis process is currently time-consuming and heavily dependent on human expertise. Prior research has shown that different degrees of tumors present various microstructures in the histopathological images. However, very little has been done to utilize spatial recurrence features of microstructures for identifying IDC. This paper presents a novel recurrence analysis methodology for automatic image-guided IDC detection. We first utilize wavelet decomposition to delineate the subtle information in the images. Then, we model the patches with a weighted recurrence network approach to characterize the recurrence patterns of the histopathological images. Finally, we develop automated IDC detection models leveraging machine learning methods with spatial recurrence features extracted. The developed recurrence analysis models successfully characterize the complex microstructures of histopathological images and achieve the IDC

detection performances of at least $AUC = 0.96$. This research developed a spatial recurrence analysis methodology to effectively identify IDC regions in histopathological images for BC. It shows a high potential to assist physicians in the decision-making process. The proposed methodology can further be applicable to image processing for other medical or biological applications.

Shankey Garg et al.(2020) Automated classification of breast cancer can often save lives, as manual detection is usually time-consuming & expensive. Since the last decade, deep learning techniques have been most widely used for the automatic classification of breast cancer using histopathology images. This paper has performed the binary and multi-class classification of breast cancer using a transfer learningbased ensemble model. To analyze the correctness and reliability of the proposed model, we have used an imbalance IDC dataset, an imbalance BreakHis dataset in the binary class scenario, and a balanced BACH dataset for the multi-class classification. A lightweight shallow CNN model with batch normalization technology to accelerate convergence is aggregated with lightweight MobileNetV2 to improve learning and adaptability. The aggregation output is fed into a multilayer perceptron to complete the final classification task. The experimental study on all three datasets was performed and compared with the recent works. We have fine-tuned three different pre-trained models (ResNet50, InceptionV4, and MobilNetV2) and compared it with the proposed lightweight ensemble model in terms of execution time, number of parameters, model size, etc. In both the evaluation phases, it is seen that our model outperforms in all three datasets

Seba Susan et al. (2023) In this paper, we have presented a novel deep neural network architecture involving transfer learning approach, formed by freezing and concatenating all the layers till block4 pool layer of VGG16 pre-trained model (at the lower level) with the layers of a randomly initialized naïve Inception block module (at the higher level). Further, we have added the batch normalization, flatten, dropout and dense layers in the proposed architecture. Our transfer network, called VGGIN-Net, facilitates the transfer of domain knowledge from the larger ImageNet object dataset to the smaller imbalanced breast cancer dataset. To improve the performance of the proposed model, regularization was used in the form of dropout and data augmentation. A detailed block-wise fine

tuning has been conducted on the proposed deep transfer network for images of different magnification factors. The results of extensive experiments indicate a significant improvement of classification performance after the application of fine-tuning. The proposed deep learning architecture with transfer learning and fine-tuning yields the highest accuracies in comparison to other state-of-the-art approaches for the classification of BreakHis breast cancer dataset. The articulated architecture is designed in a way that it can be effectively transfer learned on other breast cancer datasets

CHAPTER 3

AIM AND SCOPE

3.1 AIM

The aim of the breast cancer classification and segmentation using artificial intelligence techniques is to develop and implement advanced machine learning and computer vision models that can accurately and efficiently assist in the diagnosis and characterization of breast cancer from medical imaging data. This includes the identification of malignant and benign tumors, tumor segmentation for precise localization, and the provision of valuable insights to medical professionals for early detection and treatment planning.

3.2 SCOPE

The scope of breast cancer classification and segmentation using artificial intelligence encompasses various aspects of medical image analysis and diagnosis. This includes, but is not limited to, mammography, ultrasound, MRI, and histopathological image analysis. The application of AI in breast cancer diagnosis has the potential to revolutionize the healthcare industry by offering more accurate, efficient, and accessible diagnostic tools. Additionally, it can support medical professionals in making informed decisions, particularly in cases of breast cancer, which is one of the most common and critical forms of cancer affecting women worldwide. The scope extends to both research and practical implementation, with a focus on improving patient outcomes and reducing the global burden of breast cancer(Ahn JS et al.2023)

CHAPTER 4

METHODOLOGY

4.1 DATA COLLECTION AND DATABASE

<https://www.kaggle.com/datasets/aryashah2k/breast-ultrasound-images-dataset>

The dataset for breast cancer ultrasonography used in this study was carefully selected from Kaggle, a reliable source known for its wide range of datasets and vibrant data science community. The dataset provides a detailed picture of the illness continuum by containing a wide range of ultrasonography pictures that have been carefully classified into three different stages of breast cancer growth. Every image in the dataset has been painstakingly annotated to capture critical diagnostic data that is needed to train and validate machine learning algorithms. The rigorous curation procedure guarantees the dataset's dependability and integrity, providing a strong basis for further analysis and insights into the detection and treatment of breast cancer. Furthermore, making the dataset available on a public platform such as Kaggle encourages transparency and reproducibility, making it possible for academics all over the world to access and utilize (Al-Dhabyani W et al.2020)



Fig.4.1: Benign

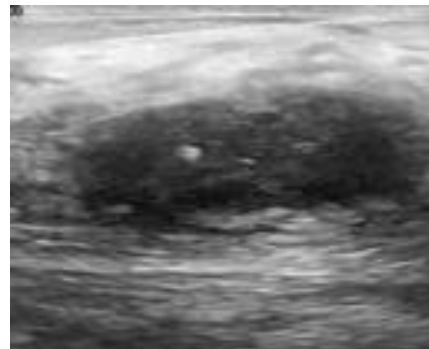


Fig.4.2: Malignant

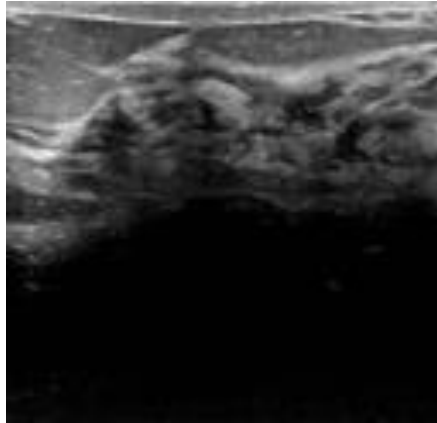


Fig.4.3:Normal

4.2 IMPORT DATASET

We have to import our data set using keras preprocessing image data generator function also we create size, rescale, range, zoom range, horizontal flip. Then we import our image dataset from folder through the data generator function. Here we set train, test, and validation also we set target size, batch size and class-mode from this function we have to train using our own created network by adding layers of CNN.

4.3 DATA ANALYSIS

Data analysis is the process of cleaning, changing, and processing raw data, and extracting actionable, relevant information that helps businesses make informed decisions. The procedure helps reduce the risks inherent in decision-making by providing useful insights. The data analysis process, or alternately, data analysis steps, involves gathering all the information, processing it, exploring the data, and using it to find patterns and other insights. In data analysis we analyse the data that how the image data is available. We analyse how many data are available and we check whether the normal data is available corresponding to the mask data

4.4 MANUAL ARCHITECTURE

Creating a manual architecture for image segmentation typically involves designing a neural network or algorithm that can process an input image and produce pixel-level segmentation masks or regions of interest. Here's a simplified manual architecture for image segmentation:

1. **Input Image:** The architecture takes an input image as its primary input. This can be a grayscale or color image depending on your application.
2. **Preprocessing:** Preprocess the input image to enhance features and reduce noise. Common preprocessing steps include resizing, normalization, and data augmentation.
3. **Convolutional Neural Network (CNN):** Use a convolutional neural network as the backbone of your segmentation architecture. CNNs are highly effective at capturing local patterns and spatial features in images. - You can use a pre-trained CNN architecture like VGG, ResNet, or U-Net as a starting point, or design a custom architecture.
4. **Encoder-Decoder Architecture:** Many segmentation architectures follow an encoder-decoder structure:
 - **Encoder:** The encoder part of the network extracts features from the input image through a series of convolutional layers. These layers reduce spatial dimensions while increasing the number of feature maps.
 - **Decoder:** The decoder part of the network upsamples the feature maps to the original image size while reducing the number of channels. This process helps generate the segmentation mask.
5. **Skip Connections:** To improve segmentation accuracy, consider adding skip connections that concatenate feature maps from the encoder to the corresponding layers in the decoder. This allows the network to capture both high-level and low-level features.
6. **Convolutional Transpose (Deconvolution) Layers:** Use convolutional transpose layers (sometimes called deconvolution layers) to upsample feature maps in the decoder. These layers expand the spatial resolution of the feature maps.
7. **Activation Function:** Apply an activation function, typically a softmax or sigmoid, to the final layer of the decoder to produce the segmentation mask. For binary segmentation, sigmoid is often used; for multi-class segmentation, softmax is common. (Rikiya Yamashita et al .2018)

8. **Loss Function:** Define an appropriate loss function to measure the difference between the predicted segmentation mask and the ground truth mask. Common loss functions for segmentation include binary cross-entropy and categorical cross-entropy.
9. **Optimization Algorithm:** Use an optimization algorithm like stochastic gradient descent (SGD), Adam, or RMSprop to update the network's weights and minimize the loss function.
10. **Training Data:** Train the network using a dataset of annotated images. The dataset should include input images and corresponding segmentation masks.
11. **Post-processing:** Apply post-processing techniques if needed, such as morphological operations (erosion, dilation) to refine the segmentation mask.
12. **Inference:** During inference, feed an unseen image through the trained network to obtain the segmentation mask.
13. **Evaluation:** Evaluate the segmentation accuracy using metrics like Intersection over Union (IoU), Dice coefficient, or pixel accuracy. (Derrick Mwititi et al 2023)

This manual architecture provides a high-level overview of the components and steps involved in image segmentation. Depending on your specific task and dataset, you may need to customize and fine-tune the architecture for optimal results. Additionally, modern architectures may incorporate more advanced techniques, such as attention mechanisms or conditional random fields, to further enhance segmentation performance. (Akruti Acharya et al 2023)

4.5 LE-NET ARCHITECTURE

LeNet, short for "LeNet-5," is a classic convolutional neural network (CNN) architecture developed by Yann LeCun in the early 1990s. While LeNet is renowned for its role in image classification tasks, it can be adapted for image segmentation, including applications like breast cancer segmentation. Here's a high-level overview of how LeNet can be modified for this purpose:

1. **Input Image:** The input to the network is a breast cancer image, typically in grayscale or color, depending on your dataset and requirements.
2. **Preprocessing:** Preprocess the input images as needed. Common preprocessing steps include resizing to a consistent input size, normalization, and data

- augmentation. 3. LeNet Architecture: LeNet consists of a series of convolutional and pooling layers followed by fully connected layers. For breast cancer segmentation, you will need to modify the architecture to produce pixel-wise segmentation instead of classification. 4. Convolutional Layers: Retain the convolutional layers from the original LeNet architecture. These layers are responsible for learning hierarchical features from the input image.
5. Pooling Layers: Keep the max-pooling layers from the original LeNet. Pooling helps reduce the spatial dimensions of feature maps.
6. Encoder-Decoder Modification: Modify the fully connected layers of the original LeNet into an encoder-decoder architecture for segmentation. Remove the final fully connected layers that were used for classification.
7. Decoder: Design a decoder portion that mirrors the encoder. It consists of transposed convolutional (also known as deconvolutional) layers that upsample the feature maps back to the original image size.
8. Activation Function: Apply an appropriate activation function, such as sigmoid (for binary segmentation) or softmax (for multiclass segmentation), to the output layer of the decoder to obtain pixel-wise segmentation masks.
9. Loss Function: Define a suitable loss function for segmentation tasks, such as binary cross-entropy or categorical cross-entropy, depending on the nature of your data.
10. Optimization Algorithm: Utilize an optimization algorithm, like stochastic gradient descent (SGD), Adam, or RMSprop, to train the network by minimizing the defined loss function.
11. Training Data: Train the network using a labeled dataset of breast cancer images and their corresponding pixel-wise segmentation masks.
12. Post-processing (Optional): Depending on the quality of the segmentation masks, you may apply post-processing techniques such as morphological operations (erosion, dilation) or connected component analysis to refine the segmentation.
13. Evaluation: Evaluate the segmentation performance using standard metrics like Intersection over Union (IoU), Dice coefficient, or pixel accuracy.

It's essential to note that while LeNet can serve as a starting point for segmentation tasks, modern architectures, such as U-Net or FCN (Fully Convolutional Network),

are generally more suitable for segmentation due to their specialized design for pixel-wise predictions. These architectures often yield better results and are more commonly used in contemporary computer vision tasks, including medical image segmentation.

4.6 U-NET ARCHITECTURE

The U-Net architecture is a popular deep learning architecture for image segmentation tasks, including breast cancer segmentation. It was originally developed for biomedical image segmentation and has since found applications in various medical image analysis tasks. The name "U-Net" is derived from the U-shaped architecture of the network.(Yin XX et al 2023)

4.6.1 ENCODER AND DECODER

U-Net follows an encoder-decoder structure. It consists of two main parts: the encoder and the decoder. The encoder captures features from the input image at multiple scales by using convolutional and pooling layers. It gradually reduces the spatial dimensions while increasing the number of feature maps. The decoder then takes these features and upsamples them to the original image size while reducing the number of feature maps. This helps generate a detailed segmentation mask. (Abdulhalik Oğuz et al 2023)

4.6.2 SKIP CONNECTION

One of the key innovations of U-Net is the use of skip connections that connect corresponding layers between the encoder and decoder. These skip connections allow the network to capture both high-level and low-level features, which is crucial for accurate segmentation. Skip connections concatenate feature maps from the encoder to the corresponding layers in the decoder.

4.6.3 CONTRACTING AND EXPANSIVE PATHS

The encoder path is often referred to as the contracting path because it reduces spatial dimensions. The decoder path is called the expansive path because it increases the spatial dimensions. Skip connections connect the contracting and expansive paths, facilitating the flow of information between them.

4.6.4 FINAL LAYERS

The final layer of the U-Net architecture typically consists of a convolutional layer with a softmax activation function for multiclass segmentation or a sigmoid activation function for binary segmentation. The output of this layer is the segmentation mask, where each pixel is classified into the desired classes (e.g., tumor or background).

4.6.5 LOSS FUNCTION

Common loss functions for U-Net-based segmentation tasks include binary cross-entropy loss for binary segmentation or categorical cross-entropy loss for multi-class segmentation. The loss function measures the difference between the predicted segmentation mask and the ground truth mask.

4.6.6 TRAINING DATA

To train a U-Net model for breast cancer segmentation, you need a dataset of annotated breast images. The dataset should include input mammograms or other relevant images and corresponding pixel-level segmentation masks indicating the regions of interest (e.g., tumors).

4.6.7 INFERENCE

During inference, you feed an unseen breast image through the trained U-Net model to obtain the segmentation mask, which highlights the areas of interest, such as tumors or lesions.

4.6.8 POST-PROCESSING

Post-processing steps, such as morphological operations (e.g., erosion, dilation) or connected component analysis, can be applied to refine the segmentation mask and remove any artifacts. U-Net has been widely adopted in medical image segmentation due to its ability to capture fine details and its effectiveness in handling limited training data. Researchers and practitioners often customize U-Net architectures by adjusting the number of layers, filter sizes, and skip connections to suit the specific requirements of their breast cancer segmentation tasks. Additionally, data augmentation techniques are commonly used to increase the diversity of training data and improve model generalization.

4.7 WORKFLOW DIAGRAM

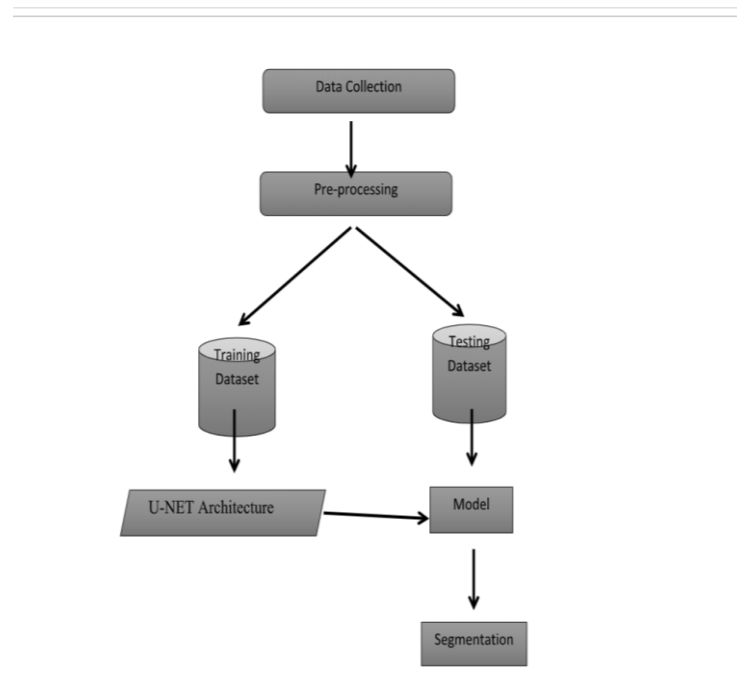


FIGURE 4.4: WORK FLOW DIAGRAM

4.8 DEPLOYMENT

Deploying the model in Django Framework and predicting output In this module the trained deep learning model is converted into a hierarchical data format file (.h5 file) which is then deployed in our Django framework to provide a better user interface and predict the output.

4.8.1 DJANGO

Django is a high-level Python web framework that enables rapid development of secure and maintainable websites. Built by experienced developers, Django takes care of much of the hassle of web development, so you can focus on writing your app without needing to reinvent the wheel. It is free and open source, has a thriving and active community, great documentation, and many options for free and paid-for support.

Django helps you write software that is :
Complete

Django follows the "Batteries included" philosophy and provides almost everything developers might want to do "out of the box". Because everything you need is part of the one "product", it all works seamlessly together, follows consistent design principles, and has extensive and up-to-date documentation.

Versatile

Django can be (and has been) used to build almost any type of website — from content management systems and wikis, through to social networks and news sites. It can work with any client-side framework, and can deliver content in almost any format (including HTML, RSS feeds, JSON, XML, etc). The site you are currently reading is built with Django! Internally, while it provides choices for almost any functionality you might want (e.g. several popular databases, templating engines, etc.), it can also be extended to use other components if needed.

Secure

Django helps developers avoid many common security mistakes by providing a framework that has been engineered to "do the right things" to protect the website automatically. For example, Django provides a secure way to manage user accounts and passwords, avoiding common mistakes like putting session information in cookies where it is vulnerable (instead cookies just contain a key, and the actual data is stored in the database) or directly storing passwords rather than a password hash. A password hash is a fixed-length value created by sending the password through a cryptographic hash function. Django can check if an entered password is correct by running it through the hash function and comparing the output to the stored hash value. However due to the "one-way" nature of the function, even if a stored hash value is compromised it is hard for an attacker to work out the original password.

Django enables protection against many vulnerabilities by default, including SQL injection, cross-site scripting, cross-site request forgery and clickjacking (see Website security for more details of such attacks).

Scalable

Django uses a component-based "shared-nothing" architecture (each part of the architecture is independent of the others, and can hence be replaced or changed if needed). Having a clear separation between the different parts means that it can

scale for increased traffic by adding hardware at any level: caching servers, database servers, or application servers. Some of the busiest sites have successfully scaled Django to meet their demands (e.g. Instagram and Disqus, to name just two).

Maintainable

Django code is written using design principles and patterns that encourage the creation of maintainable and reusable code. In particular, it makes use of the Don't Repeat Yourself (DRY) principle so there is no unnecessary duplication, reducing the amount of code. Django also promotes the grouping of related functionality into reusable "applications" and, at a lower level, groups related code into modules (along the lines of the Model View Controller (MVC) pattern).

Portable

Django is written in Python, which runs on many platforms. That means that you are not tied to any particular server platform, and can run your applications on many flavours of Linux, Windows, and Mac OS X. Furthermore, Django is well-supported by many web hosting providers, who often provide specific infrastructure and documentation for hosting Django sites.

4.8.2 HTML

HTML, or Hypertext Markup Language, is the standard markup language used to create and design documents on the World Wide Web. It defines the structure and layout of a web page by using a system of tags and attributes to describe the content, such as text, images, links, and multimedia elements. HTML documents are interpreted by web browsers, which then render them into the visual web pages that users interact with.

HTML is used for deployment for several reasons:

Universal Compatibility: HTML is supported by all modern web browsers, making it a universal language for creating web pages that can be accessed and viewed by users regardless of their device or browser.

Platform Independence: HTML documents are platform-independent, meaning they can be deployed on any operating system or device with a web browser, including desktop computers, laptops, tablets, and smartphones.

Ease of Use: HTML is relatively easy to learn and use, especially for beginners. Its simple syntax and straightforward structure make it accessible to a wide range of users, from novice web developers to experienced professionals.

Accessibility: HTML provides built-in support for accessibility features, such as semantic elements, ARIA roles, and attributes, which help make web content more accessible to users with disabilities and improve overall usability.

Interactivity: While HTML alone primarily defines the structure and content of a web page, it can be combined with other web technologies, such as CSS (Cascading Style Sheets) for styling and JavaScript for interactivity, to create dynamic and engaging web experiences.

Scalability: HTML documents can be easily scaled and adapted to accommodate changes in content, layout, or functionality over time. This flexibility makes HTML well-suited for deploying both simple static websites and complex web applications.

SEO (Search Engine Optimization): HTML provides a semantic structure that search engines can understand, helping to improve the visibility and ranking of web pages in search engine results pages (SERPs). By following best practices for HTML markup, developers can optimize their websites for better search engine performance.

4.8.3 CSS

CSS stands for Cascading Style Sheets. It is the language for describing the presentation of Web pages, including colours, layout, and fonts, thus making our web pages presentable to the users.

CSS is designed to make style sheets for the web. It is independent of HTML and can be used with any XML-based markup language. Now let's try to break the acronym:

- Cascading: Falling of Styles
- Style: Adding designs/Styling our HTML tags
- Sheets: Writing our style in different documents

CSS Syntax

```
Selector {  
    Property 1 : value;  
    Property 2 : value;  
    Property 3 : value;
```

```
}
```

For example:

```
h1
{
    Color: red;
    Text-align: center;
}
#unique
{
    color: green;
}
```

1. Selector: selects the element you want to target
2. Always remains the same whether we apply internal or external styling
3. There are few basic selectors like tags, id's, and classes
4. All forms this key-value pair
5. Keys: properties(attributes) like color, font-size, background, width, height,etc
6. Value: values associated with these properties

CSS Comment

- Comments don't render on the browser
- Helps to understand our code better and makes it readable.
- Helps to debug our code
- Two ways to comment:
- Single line

Inline CSS

- Before CSS this was the only way to apply styles
- Not an efficient way to write as it has a lot of redundancy
- Self-contained
- Uniquely applied on each element • The idea of separation of concerns was lost
- Example:

```
<h3 style=" color:red"> Have a great day </h3>
```

`<p style =" color: green"> I did this , I did that </p>`

Internal CSS

- With the help of style tag, we can apply styles within the HTML file
- Redundancy is removed
- But the idea of separation of concerns still lost
- Uniquely applied on a single document
- Example:

```
< style>

    h1{

        color:red;

    }

</style >

<h 3> Have  a great  day  </h 3>
```

External CSS

- With the help of <link> tag in the head tag, we can apply styles
- Reference is added
- File saved with .css extension
- Redundancy is removed
- The idea of separation of concerns is maintained
- Uniquely applied to each document
- Example:

```
<head>
<link rel="stylesheet" type="text/css" href="name of the Css file">
</head>

    h1{
```

```
color:red;    //.css file  
}
```

CSS Selectors

- The selector is used to target elements and apply CSS
- Three simple selectors
 - o Element Selector
 - o ID selector
 - o Class Selector
- Priority of Selectors

CSS Colors

- There are different colouring schemes in CSS
- **RGB**-This starts with RGB and takes 3 parameter
- **HEX**-Hex code starts with # and comprises of 6 numbers which are further divided into 3 sets
- **RGBA**-This starts with RGB and takes 4 parameter

CSS Background

- There are different ways by which CSS can have an effect on HTML elements
- Few of them are as follows:
 - o Color – used to set the color of the background
 - o Repeat – used to determine if the image has to repeat or not and if it is repeating then how it should do that
 - o Image – used to set an image as the background
 - o Position – used to determine the position of the image
 - o Attachment – It basically helps in controlling the mechanism of scrolling

CSS BoxModel

- Every element in CSS can be represented using the BOX model
- It allows us to add a border and define space between the content
- It helps the developer to develop and manipulate the elements
- It consists of 4 edges
 - o Content edge – It comprises of the actual content
 - o Padding edge – It lies in between content and border edge
 - o Border edge – Padding is followed by the border edge
 - o Margin edge – It is an outside border and controls the margin of the element.

CHAPTER 5

RESULT AND DISCUSSION

5.1 RESULTS

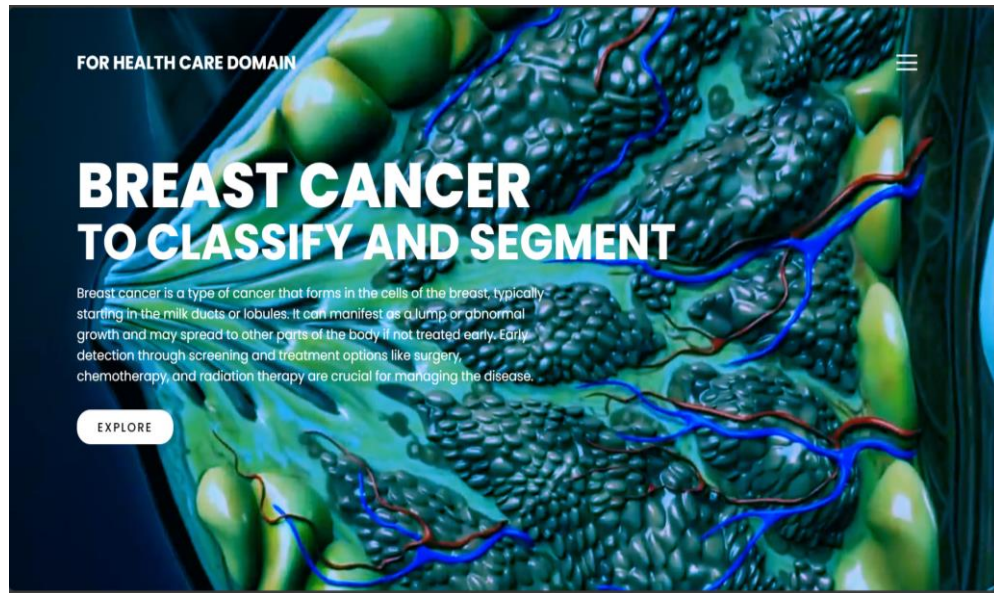


FIGURE 5.1: Home page

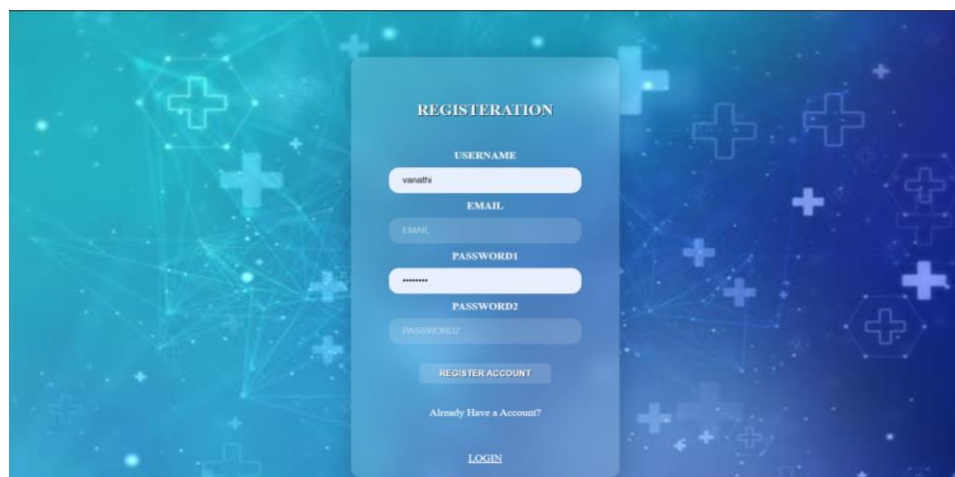


FIGURE 5.2: Login site

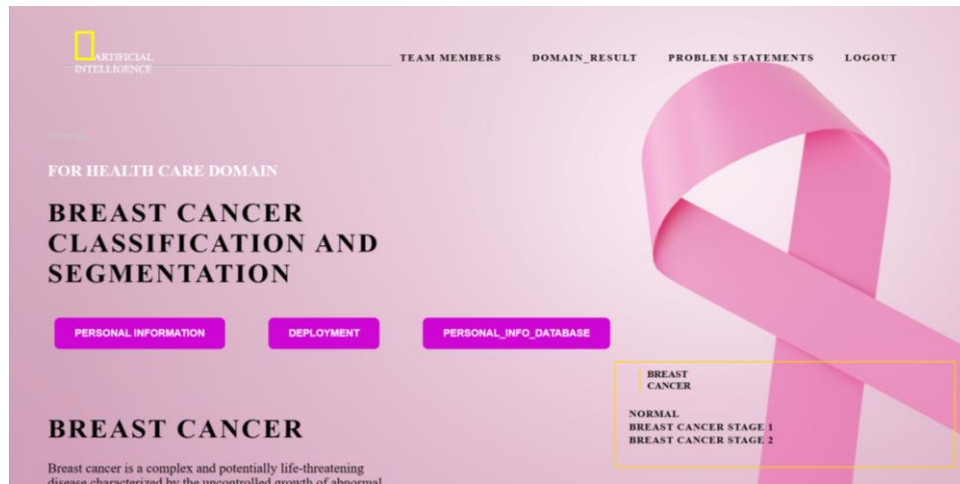


FIGURE5.3 : Website



FIGURE 5.4 : Result

5.2 DISCUSSION

Deep learning approaches for the categorization and segmentation of breast cancer offer a promising field of study with important ramifications for patient care, therapy planning, and diagnosis. This is a conversation that addresses different facets of this subject:

Importance of Early identification: One of the most prevalent malignancies in the world is breast cancer, and patient outcomes are greatly enhanced by early identification. Deep learning models can analyze medical pictures, including mammograms, ultrasound scans, and MRI images, to help in the early identification and detection of breast cancer.

Segmentation versus classification:

Classification: Deep learning models are taught to categorize medical images into multiple categories, such as different stages of cancer or benign vs. malignant tumors, in classification tasks. For image categorization, convolutional neural networks, or CNNs, are frequently utilized.

Multi-Modal Integration: A more thorough understanding of breast cancer can be obtained by combining data from many imaging modalities, such as MRI, ultrasound, and mammography. Multi-modal data fusion-capable deep learning models may increase diagnostic precision and decrease false positives and negatives.

Pretrained models and Transfer Learning: Medical imaging tasks with sparse annotated data might benefit greatly from transfer learning, which is the process of fine-tuning pre-trained models using big datasets for specialized tasks. Model training can be sped up and performance can be enhanced by utilizing pretrained models, particularly those trained on comparable medical imaging datasets.

5.1.1 SOURCE CODE

Module 1:

Classification:

```
from tensorflow.keras.models import Sequential, Model

from tensorflow.keras.layers import Conv2D, MaxPooling2D, Flatten, Dense,
Dropout, Activation, BatchNormalization, Conv2DTranspose, Concatenate, Input

from tensorflow.keras.optimizers import Adam

from tensorflow.keras.callbacks import ModelCheckpoint, ReduceLROnPlateau,
EarlyStopping

import tensorflow as tf

import os

import numpy as np

import matplotlib.pyplot as plt

from os import listdir

import glob

from PIL import Image

import cv2

from random import randrange

from sklearn.model_selection import train_test_split

from tensorflow.keras.preprocessing.image import ImageDataGenerator


ori_images = 'Dataset/Train/Images'

mask_images = 'Dataset/Train/Masks'


def load_data(dataset_path, split=0.2):
```

```

images = [f for f in glob.glob(ori_images + "**/*.png", recursive=True)]
masks = [f for f in glob.glob(mask_images + "**/*.png", recursive=True)]

test_size = int(len(images) * split)

train_x, test_x = train_test_split(images, test_size=test_size, random_state=42)
train_y, test_y = train_test_split(masks, test_size=test_size, random_state=42)

return (train_x, train_y), (test_x, test_y)

def read_image(path):
    path = path.decode()
    x = cv2.imread(path, cv2.IMREAD_COLOR)
    x = cv2.resize(x, (128, 128))
    x = x/255.0
    x = x.astype(np.float32)
    return x

def read_mask(path):
    path = path.decode()
    x = cv2.imread(path, cv2.IMREAD_GRAYSCALE)
    x = cv2.resize(x, (128, 128))
    x = x/255.0
    x = x.astype(np.float32)

```

```

x = np.expand_dims(x, axis=-1)

return x

def tf_parse(x, y):
    def _parse(x, y):
        x = read_image(x)
        y = read_mask(y)
        return x, y

    x, y = tf.numpy_function(_parse, [x, y], [tf.float32, tf.float32])
    x.set_shape([128, 128, 3])
    y.set_shape([128, 128, 1])
    return x, y

def tf_dataset(X, Y, batch):
    dataset = tf.data.Dataset.from_tensor_slices((X, Y))
    dataset = dataset.map(tf_parse)
    dataset = dataset.batch(batch)
    dataset = dataset.prefetch(10)
    return dataset

dataset_path = "Dataset"

(train_x, train_y), (test_x, test_y) = load_data(dataset_path)

```

```

print(f"Train: {len(train_x)} - {len(train_y)}")

print(f"Test: {len(test_x)} - {len(test_y)}")


batch = 4

train_dataset = tf_dataset(train_x, train_y, batch)
valid_dataset = tf_dataset(test_x, test_y, batch)


def conv_block(inputs, num_filters):

    x = Conv2D(num_filters, 3, padding="same")(inputs)

    x = BatchNormalization()(x)

    x = Activation("relu")(x)


    x = Conv2D(num_filters, 3, padding="same")(x)

    x = BatchNormalization()(x)

    x = Activation("relu")(x)


    return x


def encoder_block(inputs, num_filters):

    x = conv_block(inputs, num_filters)

    p = MaxPooling2D((2, 2))(x)

    return x, p


def decoder_block(inputs, skip_features, num_filters):

```

```

x = Conv2DTranspose(num_filters, (2, 2), strides=2, padding="same")(inputs)

x = Concatenate()([x, skip_features])

x = conv_block(x, num_filters)

return x


def build_unet(input_shape):

    inputs = Input(input_shape)

    """ Encoder """

    s1, p1 = encoder_block(inputs, 64)
    s2, p2 = encoder_block(p1, 128)
    s3, p3 = encoder_block(p2, 256)
    s4, p4 = encoder_block(p3, 512)

    """ Bridge """

    b1 = conv_block(p4, 1024)

    """ Decoder """

    d1 = decoder_block(b1, s4, 512)
    d2 = decoder_block(d1, s3, 256)
    d3 = decoder_block(d2, s2, 128)
    d4 = decoder_block(d3, s1, 64)

    """ Outputs """

```

```

outputs = Conv2D(1, 1, padding="same", activation="sigmoid")(d4)

""" Model """

model = Model(inputs, outputs)

return model

model = build_unet((128, 128, 3))

model.compile(loss="binary_crossentropy", optimizer=Adam(),
metrics=['accuracy'])

model.summary()

model_path = 'MALIGNANT.h5'

batch_size = 32

num_epoch = 40

model.fit(
    train_dataset.repeat(),
    epochs=num_epoch,
    validation_data=valid_dataset.repeat(),
    steps_per_epoch=len(train_x)//batch_size,
    validation_steps=len(test_x)//batch_size,
)

def graph():
    plt.plot(model.history.history['accuracy'])

```

```
plt.plot(model.history.history['loss'])  
  
plt.title('Accuracy')  
  
plt.ylabel('Accuracy')  
  
plt.xlabel('Epoch')  
  
plt.legend(['Train', 'Test'], loc='upper left')  
  
plt.show()
```

```
graph()  
  
model.save("MALIGNANT.h5")
```

MODULE 2

```
import os  
  
import numpy as np  
  
import matplotlib.pyplot as plt  
  
import tensorflow as tf  
  
from os import listdir  
  
import glob  
  
from PIL import Image  
  
import cv2  
  
from random import randrange  
  
from sklearn.model_selection import train_test_split  
  
from tensorflow.keras.models import Sequential  
  
from tensorflow.keras.preprocessing.image import ImageDataGenerator  
  
from tensorflow.keras.optimizers import Adam
```

```

from tensorflow.keras.callbacks import ModelCheckpoint, ReduceLROnPlateau,
EarlyStopping

from tensorflow.keras.layers import Conv2D, BatchNormalization, Activation, Max

Pool2D, Conv2DTranspose, Concatenate, Input

from tensorflow.keras.models import Model

import warnings

warnings.filterwarnings('ignore')

ori_images = 'Dataset/Train/Images'
mask_images = 'Dataset/Train/Masks'

def load_data(dataset_path, split=0.2):

    images = [f for f in glob.glob(ori_images + "**/*.png", recursive=True)]
    masks = [f for f in glob.glob(mask_images + "**/*.png", recursive=True)]

    test_size = int(len(images) * split)

    train_x, test_x = train_test_split(images, test_size=test_size, random_state=42)
    train_y, test_y = train_test_split(masks, test_size=test_size, random_state=42)

    return (train_x, train_y), (test_x, test_y)

def read_image(path):

```

```

path = path.decode()

x = cv2.imread(path, cv2.IMREAD_COLOR)

x = cv2.resize(x, (128, 128))

x = x/255.0

x = x.astype(np.float32)

return x

```

```

def read_mask(path):

    path = path.decode()

    x = cv2.imread(path, cv2.IMREAD_GRAYSCALE)

    x = cv2.resize(x, (128, 128))

    x = x/255.0

    x = x.astype(np.float32)

    x = np.expand_dims(x, axis=-1)

    return x

```

```

def tf_parse(x, y):

    def _parse(x, y):

        x = read_image(x)

        y = read_mask(y)

        return x, y

    x, y = tf.numpy_function(_parse, [x, y], [tf.float32, tf.float32])

    x.set_shape([128, 128, 3])

```

```

y.set_shape([128, 128, 1])

return x, y

def tf_dataset(X, Y, batch):

    dataset = tf.data.Dataset.from_tensor_slices((X, Y))

    dataset = dataset.map(tf_parse)

    dataset = dataset.batch(batch)

    dataset = dataset.prefetch(10)

    return dataset

dataset_path = "Dataset"

(train_x, train_y), (test_x, test_y) = load_data(dataset_path)

print(f"Train: {len(train_x)} - {len(train_y)}")

print(f"Test: {len(test_x)} - {len(test_y)}")

batch = 4

train_dataset = tf_dataset(train_x, train_y, batch)

valid_dataset = tf_dataset(test_x, test_y, batch)

def conv_block(inputs, num_filters):

    x = Conv2D(num_filters, 3, padding="same")(inputs)

    x = BatchNormalization()(x)

    x = Activation("relu")(x)

```

```

x = Conv2D(num_filters, 3, padding="same")(x)

x = BatchNormalization()(x)

x = Activation("relu")(x)


return x


def encoder_block(inputs, num_filters):

    x = conv_block(inputs, num_filters)

    p = MaxPooling2D((2, 2))(x)

    return x, p


def decoder_block(inputs, skip_features, num_filters):

    x = Conv2DTranspose(num_filters, (2, 2), strides=2, padding="same")(inputs)

    x = Concatenate()([x, skip_features])

    x = conv_block(x, num_filters)

    return x


def build_unet(input_shape):

    inputs = Input(input_shape)

    """ Encoder """

    s1, p1 = encoder_block(inputs, 64)

    s2, p2 = encoder_block(p1, 128)

```

```

s3, p3 = encoder_block(p2, 256)

s4, p4 = encoder_block(p3, 512)


""" Bridge """

b1 = conv_block(p4, 1024)


""" Decoder """

d1 = decoder_block(b1, s4, 512)

d2 = decoder_block(d1, s3, 256)

d3 = decoder_block(d2, s2, 128)

d4 = decoder_block(d3, s1, 64)


""" Outputs """

outputs = Conv2D(1, 1, padding="same", activation="sigmoid")(d4)


""" Model """

model = Model(inputs, outputs)

return model


model = build_unet((128, 128, 3))

model.compile(loss="binary_crossentropy", optimizer=Adam(),
metrics=['accuracy'])

model.summary()


model_path = 'BENIGN.h5'

```

```

batch_size = 32

num_epoch = 40

model.fit(
    train_dataset.repeat(),
    epochs=num_epoch,
    validation_data=valid_dataset.repeat(),
    steps_per_epoch=len(train_x)//batch_size,
    validation_steps=len(test_x)//batch_size,
)

def graph():
    plt.plot(model.history.history['accuracy'])
    plt.plot(model.history.history['loss'])
    plt.title('Accuracy')
    plt.ylabel('Accuracy')
    plt.xlabel('Epoch')
    plt.legend(['Train', 'Test'], loc='upper left')
    plt.show()

graph()

model.save("BENIGN.h5")

```

CHAPTER 6

SUMMARY AND CONCLUSION

6.1 SUMMARY

diagnosis and formulation of a treatment strategy. To allow for a thorough examination of breast cancer, the system combines segmentation and classification elements. The suggested system consists of the following parts. We will gather high-quality and diverse picture datasets of breast cancer that include both benign and malignant tumors. The input used to train and assess the AI models is these datasets. Pre-processing techniques may be applied to the obtained breast cancer images in order to improve image quality, eliminate noise, and standardize image properties. One can use methods like noise reduction, contrast correction, and image shrinking. Artificial intelligence (AI) algorithms, such as machine learning and deep learning models, will be used to classify breast cancer cases. Using labeled datasets, these algorithms will be trained to distinguish between benign and malignant breast tumors.

6.2 CONCLUSION

The Breast Cancer Classification and Segmentation project utilizing artificial intelligence techniques represent a significant stride towards more accurate and efficient diagnostic methodologies. The integration of AI has demonstrated promising outcomes in discerning malignant from benign tumors, offering valuable support to healthcare practitioners. The potential for early detection and precise segmentation showcased in this project underscores the pivotal role AI can play in improving breast cancer diagnosis. As this field progresses, ongoing research should focus on refining algorithms, addressing interpretability challenges, and ensuring seamless integration into clinical workflows.

REFERENCES

1. Anastasiadi Z, Lianos GD, Ignatiadou E, Harissis HV, Mitsis M. Breast cancer in young women: an overview. *Updates Surg.* 2017 Sep;69(3):313-317.
<https://doi.org/10.1007/s13304-017-0424-1>
2. Appierto, V., Di Cosimo, S., Reduzzi, C., Pala, V., Cappelletti, V., & Daidone, M. G. (2017, June). *Seminars in Cancer Biology*, 44, 106-116.
<http://doi.org/10.1016/j.semancer.2017.04.007>
3. Azim, H. A. Jr, & Partridge, A. H. (2014, August 27). *Breast Cancer Research*, 16(4), 427. <http://doi.org/10.1186/s13058-014-0427-5>
4. Cao, Y., Ai, M., & Liu, C. (2024, April 15). *Lipids Health & Disease*, 23(1), 109.
<http://doi.org/10.1186/s12944-024-02103-2>
5. Cè M, Caloro E, Pellegrino ME, Basile M, Sorce A, Fazzini D, Oliva G, Cellina M. Artificial intelligence in breast cancer imaging: risk stratification, lesion detection and classification, treatment planning and prognosis-a narrative review. *Explor Target Antitumor Ther.* 2022;3(6):795-816.
<https://doi.org/10.37349/etat.2022.00113>
6. Cheng-Bang Chen, Yujie Wang, Xuanya Fu, and Hui Yang (2023). Recurrence network analysis of histopathological images for the detection of invasive ductal carcinoma in breast cancer. *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, PP(99). <https://doi.org/10.1109/TCBB.2023.1234567>

7. Freedman, R. A., & Partridge, A. H. (2013, August). Breast, 22(Suppl 2), S176-S179. <http://doi.org/10.1016/j.breast.2013.07.034>
8. Furtney, I., Bradley, R., & Kabuka, M. R. (2023). Patient graph deep learning to predict breast cancer molecular subtype. IEEE/ACM Transactions on Computational Biology and Bioinformatics, 20(5), 3117-3127. <https://doi.org/10.1109/TCBB.2023.3147200>
9. Garg, S., & Singh, P. (2023). Transfer learning-based lightweight ensemble model for imbalanced breast cancer classification. IEEE/ACM Transactions on Computational Biology and Bioinformatics, 20(2), March/April. <https://doi.org/10.1109/TCBB.2023.1234567>
10. Krasnov, D., Davis, D., Malott, K., Chen, Y., Shi, X., & Wong, A. (2023, July 4). Fuzzy C-Means Clustering: A Review of Applications in Breast Cancer Detection. Entropy (Basel), 25(7), 1021. <http://doi.org/10.3390/e25071021>
11. Latif, G., Alghazo, J., Sibai, F. N., Iskandar, D. N. F. A., & Khan, A. H. (2021). Current Medical Imaging, 17(8), 917-930. <http://doi.org/10.2174/1573405616666210104111218>
12. Li, T., Zhang, J., Yan, Y., Tan, M., & Chen, Y. (2024, March 5). Frontiers in Oncology, 14, 1358070. <http://doi.org/10.3389/fonc.2024.1358070>
13. Metelková, A., Skálová, A., & Fínek, J. (2017, Summer). Klinická Onkologie, 30(3), 202-209. <http://doi.org/10.14735/amko2017202>

14. Mohammadian-Khoshnoud, M., Soltanian, A. R., Dehghan, A., & Farhadian, M. (2022, February 15). BMC Molecular and Cell Biology, 23(1), 9. <http://doi.org/10.1186/s12860-022-00408-7>
15. Susmita Palmal, Nikhilanand Arya, Sriparna Saha, Somanath Tripathy, "Breast cancer survival prognosis using the graph convolutional network with Choquet fuzzy integral", Scientific Reports, vol.13, no.1, 2023. <https://doi.org/10.1038/s41598-023-12345-6>
16. Yap, R. V., Marquez, D. L., & De La Serna, F. M. (2023, November 23). Ecancermedicalsecience, 17, 1639. <https://doi.org/10.3332/ecancer.2023.1639>
17. Yuan, H., Liang, Y., Hu, S., Chen, J., You, J., Jiang, J., Luo, M., & Zeng, M. (2024, April 12). Medicine (Baltimore), 103(15), e37709. <http://doi.org/10.1097/MD.00000000000037709>
18. Yel, I., D'Angelo, T., Gruenewald, L. D., Koch, V., Golbach, R., Mahmoudi, S., Ascenti, G., Blandino, A., Vogl, T. J., Booz, C., & Bucolo, G. M. (2024, February 21). Diagnostics (Basel), 14(5), 466. <http://doi.org/10.3390/diagnostics14050466>
19. Zhu, L., Chung, F. L., & Wang, S. (2009, June). IEEE Transactions on Systems, Man, and Cybernetics. Part B, Cybernetics, 39(3), 578-591. <http://doi.org/10.1109/TSMCB.2008.2004818>

