DAYANANDA SAGAR UNIVERSITY

Devarakaggalahalli, Harohalli
Kanakapura Road, Ramanagara - 562112, Karnataka, India

SCHOOL OF ENGINEERING

Bachelor of Technology
in
COMPUTER SCIENCE AND ENGINEERING

Project Phase-II Report

Onco-Vision: Deep Learning for Tumor Cell Detection in Breast Cancer

Batch: 98

By

JHANVI ACHARYA-ENG21CS0170
K VIDYASHREE-ENG21CS0175
KAMLESH N-ENG21CS0179
KAVITHA N-ENG21CS0185

Under the supervision of
Dr. T. POONGODI
Professor

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING,
SCHOOL OF ENGINEERING
DAYANANDA SAGAR UNIVERSITY,

(2024-2025)

DAYANANDA SAGAR UNIVERSITY

SCHOOL OF ENGINEERING

Department of Computer Science & Engineering

Devarakaggalahalli, Harohalli, Kanakapura Road, Ramanagara - 562112
Karnataka, India

CERTIFICATE

This is to certify that the Project Phase-II work titled **"Onco-Vision: Deep Learning For Tumour Cell Detection In Breast Cancer"** is carried out by **JHANVI ACHARYA (ENG21CS0170), K VIDYASHREE (ENG21CS0175), KAMLESH N (ENG21CS0179), KAVITHA N (ENG21CS0185)** bonafide students eighth semester of Bachelor of Technology in Computer Science and Engineering at the School of Engineering, Dayananda Sagar University, Bangalore in partial fulfillment for the award of degree in Bachelor of Technology in Computer Science and Engineering, during the year **2024-2025**.

Dr.T.Poongodi	**Dr. Girisha G S**	**Dr. Udaya Kumar Reddy K R**
Professor Dept. of CS&E, School of Engineering Dayananda Sagar University	Chairman CSE School of Engineering Dayananda Sagar University	Dean School of Engineering Dayananda Sagar University
Date:	Date:	Date:

Name of the Examiner **Signature of Examiner**

1.

2.

DECLARATION

We **JHANVI ACHARYA (ENG21CS0170), K VIYASHREE (ENG21CS0175), KAMLESH N (ENG21CS0179), KAVITHA N (ENG21CS0185)** are students of eighth semester B. Tech in **Computer Science and Engineering**, at School of Engineering, **Dayananda Sagar University**, hereby declare that the Project Phase-II titled **"Onco-Vision: Deep Learning for Tumour Cell Detection in Breast Cancer"** has been carried out by us and submitted in partial fulfilment for the award of degree in **Bachelor of Technology in Computer Science and Engineering** during the academic year **2024-2025**

Student	**Signature**
JHANVI ACHARYA	
ENG21CS0170	
K VIDYASHREE	
ENG21CS0175	
KAMLESH N	
ENG21CS0179	
KAVITHA N	
ENG21CS0185	

Place : Bangalore

Date :

ACKNOWLEDGEMENT

It is a great pleasure for us to acknowledge the assistance and support of many individuals who have been responsible for the successful completion of this project work.

First, we take this opportunity to express our sincere gratitude to School of Engineering , Dayananda Sagar University for providing us with a great opportunity to pursue our Bachelor's degree in this institution.

We would like to thank ***Dr. Udaya Kumar Reddy K R, Dean, School of Engineering, Dayananda Sagar University*** *for his constant encouragement and expert advice.*

It is a matter of immense pleasure to express our sincere thanks to ***Dr. Girisha G S, Department Chairman****,* ***Computer Science and Engineering****,* ***Dayananda Sagar University,*** *for providing the right academic guidance that made our task possible.*

We would like to thank our guide ***Dr.T.Poongodi ,Professor****,* ***Dept. of Computer Science and Engineering****,* ***Dayananda Sagar University****, for sparing her valuable time to help us with every step of our project work. This helped pave the way for smooth progress and fruitful culmination of the project.*

We also thank our ***Project Coordinators: Dr. Meenakshi Malhotra, Dr. Kumar Dilip, and Dr. Sivananda Reddy E,*** *and all the staff members of Computer Science and Engineering for their support.*

We are also grateful to our family and friends who provided us with every requirement throughout the course.

We would like to thank one and all who directly or indirectly helped us in the Project work.

TABLE OF CONTENTS

NOMENCLATURE USED

CNN	Convolution Neural Network
DL	Deep Learning
Dense Net	Densely Connected Convolution Network
Py	Python
ReLU	Rectified Linear Unit

LIST OF FIGURES

ABSTRACT

Profound learning emerged as a potent tool during the transformation of breast cancer detection at the stage of tumor cell identification in histopathological images. Among deep learning methods, Cnns-Densenet-201 has been found to be the most suited to complex image processing applications as it can effectively handle more intricate image information and extract a large number of important features from our dataset. Following a morphological layer, the enormous depth of the architecture combined with a Dense Net- 201 based residual learning system can extract the coarse features of progressive properties on very large volumes of annotated tissue tests at the interface from a number of perspectives. In a later step, based on its power, it separates kinds of cancerous cells and non-cancerous cells with a very high level of accuracy. To enhance the utilization of these systems for application building, image pre-processing plays a very important role. Normalization maintains pixel values within a range so that uniformity exists throughout the images; whereas, increasing the information, such as rotations, flipping, and altering brightness, makes the dataset diverse and hence promotes model generalization. Segmentation separates target regions of interest containing tumor cell boundaries from surrounding tissues so that the input information is clearer and tumor margins are even more accurate-a sine qua non in cancer research and got accuracy of 93%.

CHAPTER 1

INTRODUCTION

1.1 INTRODUCTION

Breast cancer continues to be one of the most common and life-threatening diseases affecting women globally. According to the World Health Organization (WHO), breast cancer has surpassed lung cancer as the most frequently diagnosed form of cancer worldwide. In 2020 alone, there were more than 2.3 million new cases and approximately 685,000 deaths attributed to breast cancer. These statistics underline the urgent need for effective early detection methods that can lead to timely diagnosis and treatment, ultimately improving patient survival rates. Despite the advances in oncology and medical imaging, early-stage diagnosis remains a critical challenge due to the subtlety of symptoms and variations in tumor morphology.

The survival rate of breast cancer is significantly higher when detected at an early stage. Early diagnosis not only improves the chances of effective treatment but also reduces the financial and emotional burden on patients and healthcare systems. However, many parts of the world still lack access to efficient diagnostic technologies and trained medical professionals. This global disparity calls for the development of affordable, accurate, and scalable diagnostic tools that can assist in early detection and reduce dependency on manual interpretation.

One of the traditional methods for breast cancer diagnosis is histopathological analysis, where tissue samples are examined under a microscope by pathologists. While this approach is widely used and clinically validated, it is often labor-intensive, time-consuming, and subject to inter-observer variability. Differences in pathologists' training, fatigue, and subjectivity in visual assessment can affect diagnostic consistency. Additionally, the increasing number of cancer cases places a growing burden on pathologists, increasing the risk of misdiagnosis and delayed treatment.

Histopathology imaging is a critical component in diagnosing various types of cancer, including breast cancer. It involves staining tissue samples to enhance contrast and magnifying them under microscopes to analyze cellular structures. In the case of breast cancer, pathologists look for characteristics such as cell density, nucleus shape, gland formation, and other morphological features to classify the tumor as benign

(non-cancerous) or malignant (cancerous). These images, when captured at high magnification levels such as 40x, 100x, 200x, or 400x, reveal intricate tissue patterns that are crucial for diagnosis.

The BreakHis (Breast Cancer Histopathological Image Classification) dataset, which was used in this project, contains thousands of such histopathology images at various magnifications. Specifically, this project utilized the 400x magnification set, which provides a high-resolution view of the cellular structures. The dataset is divided into benign and malignant classes, further categorized into different subtypes. These images serve as a valuable resource for training and testing deep learning models that can automate the classification process.

Manual examination of histopathology slides is not only time-consuming but also prone to variability in diagnosis. Even highly experienced pathologists may disagree on borderline cases or miss subtle signs of malignancy. As a result, the need for automated tools that can assist or even outperform human experts in diagnostic tasks has become increasingly important.

The past decade has seen a rapid rise in the application of Artificial Intelligence (AI) in healthcare, particularly in the domain of medical imaging. AI techniques, especially those rooted in Machine Learning (ML) and Deep Learning (DL), have demonstrated remarkable success in image classification, segmentation, and pattern recognition tasks. These capabilities make AI a promising solution for enhancing diagnostic accuracy and consistency in fields like radiology, pathology, and dermatology.

ML involves training algorithms on large datasets so they can learn patterns and make predictions or decisions without being explicitly programmed for specific tasks. However, traditional machine learning methods often require extensive feature engineering, where domain experts manually select and craft features from raw data. This approach can be limited in complex domains like medical imaging, where meaningful features are often difficult to extract manually. Deep learning, a subfield of machine learning, overcomes this limitation by automatically learning hierarchical features directly from the data using artificial neural networks. Among various deep learning architectures, Convolutional Neural Networks (CNNs) have become the standard for image analysis tasks due to their ability to learn spatial hierarchies and extract meaningful patterns from images.In medical image analysis, CNNs have shown exceptional performance in detecting diabetic retinopathy, classifying skin lesions, identifying lung diseases from X-

rays, and, most relevantly, detecting cancer from histopathology images. These models can learn to recognize complex visual cues—such as texture, shape, and structure—that are often indicative of disease. This capability is particularly valuable in breast cancer detection, where variations in cell arrangement and morphology are subtle but critical for diagnosis.

CNN are a specialized class of deep learning models designed to work with data that has a grid-like topology, such as images. CNNs have revolutionized computer vision by enabling machines to learn and extract spatial hierarchies of features directly from image pixels without the need for manual feature extraction. A typical CNN architecture comprises multiple layers, including convolutional layers, activation layers (such as ReLU), pooling layers, fully connected layers, and a final classification layer.

The convolutional layers apply a set of filters (kernels) across the image to detect low-level features like edges, corners, and textures. These are passed through activation functions to introduce non-linearity and then down sampled using pooling layers to reduce spatial dimensions and computational complexity. As the network goes deeper, it learns increasingly abstract features—combinations of simple patterns that represent higher-level concepts like shapes, regions, or specific tissue patterns relevant in medical imaging.

Despite their power, CNNs require large amounts of labeled data and computational resources to train from scratch. In many medical domains, especially histopathology, collecting vast labeled datasets is not feasible due to patient privacy issues, the cost of expert annotation, and limited sample availability. Moreover, training deep networks from scratch on limited data can lead to overfitting, where the model learns the training data too well and fails to generalize to unseen data.

To address this issue, researchers commonly adopt a technique known as **transfer learning**. Transfer learning involves using a model pre-trained on a large and diverse dataset (such as ImageNet) and adapting it to a specific task with a smaller dataset. In this project, transfer learning is implemented by using a pre-trained CNN architecture called DenseNet201. By leveraging learned representations from a large dataset, the model can extract general features like edges, colors, and textures in the initial layers and fine-tune the later layers to specialize in breast cancer histology classification.

Transfer learning not only reduces the computational cost and training time but also often improves model performance by providing a strong initialization, especially when labeled data is scarce. It has become a cornerstone of medical image classification tasks due to its efficiency and effectiveness.

DenseNet (Dense Convolutional Network) is a powerful deep learning architecture introduced to improve the flow of information and gradients through the network. In traditional CNNs, each layer has connections only to its immediate preceding layer. DenseNet changes this by connecting each layer to **every other layer** in a feed-forward fashion. That is, each layer receives inputs from all previous layers and passes its own feature maps to all subsequent layers. This design leads to the formation of dense connections and offers several advantages.

One key advantage is the **alleviation of the vanishing gradient problem**, which often hampers the training of very deep networks. Since each layer has direct access to the gradients from the loss function and the original input signal, training becomes more stable and efficient. DenseNet also encourages **feature reuse**, as each layer builds upon previously computed features, leading to more compact models with fewer parameters.

DenseNet201, as the name suggests, consists of 201 layers and has been pre-trained on the ImageNet dataset. Its densely connected structure ensures maximum information flow between layers, making it especially effective in learning complex representations. For histopathological image classification, where fine-grained features like cell shape and nucleus texture are crucial, DenseNet201's deep architecture provides a clear advantage.

Compared to other popular architectures like ResNet-101 (Residual Network), DenseNet-201 has fewer parameters and is less prone to overfitting, while maintaining or even improving accuracy. ResNet introduces identity shortcuts to skip layers and reduce degradation in deep networks, but it does not establish dense connections. DenseNet's architectural innovation makes it more suitable for feature-rich tasks such as medical image analysis, where subtle variations in features must be captured across multiple layers.

In this project, DenseNet201 was fine-tuned on the BreakHis 400x dataset using transfer learning. The

base layers of the model, which already possess rich feature-extraction capabilities, were retained, while the final classification layers were replaced and trained to distinguish between benign and malignant tissue samples. Additional techniques such as dropout and batch normalization were used to improve generalization and training stability.

The primary objective of this project is to develop an accurate, robust, and efficient deep learning-based model to automatically classify breast cancer histopathological images as benign or malignant. By utilizing DenseNet201 in a transfer learning setup, the project aims to demonstrate the viability of pre-trained CNNs in aiding medical professionals with diagnostic tasks.

The significance of this study lies in its potential impact on real-world clinical workflows. By reducing the reliance on manual analysis, which can be slow and inconsistent, automated systems like the one proposed can assist pathologists in making faster and more reliable diagnoses. This is especially critical in resource-limited settings where access to skilled professionals may be restricted.

Moreover, the project contributes to the growing body of evidence supporting the use of AI in healthcare. It showcases how carefully curated datasets, combined with state-of-the-art deep learning models, can lead to practical and deployable solutions for disease detection. Achieving a high classification accuracy of approximately **93%** on the BreakHis dataset further validates the effectiveness of the chosen methodology.

This project also provides a comparative analysis with other deep learning architectures like ResNet101 and InceptionV3. By exploring the strengths and limitations of each model, the study offers insights into how different architectures perform in the domain of medical image classification, thereby guiding future research directions.

Emphasis on Early Detection: The introduction clearly states the primary objective of early cancer detection. High Accuracy: The goal of achieving high accuracy in identifying complex and sensitive malignant features is explicitly mentioned. Focus on Deep Learning: The use of deep learning, specifically CNNs and the DenseNet-201 architecture, is highlighted. Addressing Limitations of Traditional Methods: The introduction acknowledges the limitations of current diagnostic methods, such as subjectivity and

potential for missed diagnoses. Potential Impact: The potential impact of the research on improving patient outcomes through earlier diagnosis is emphasized.

1.2 OBJECTIVE

The primary objective of this project is to develop an accurate and efficient deep learning-based diagnostic tool for the **automatic classification of breast cancer histopathological images** using the DenseNet201 architecture. This system aims to support pathologists and medical professionals by reducing diagnostic time, improving consistency, and enhancing early detection of breast cancer through high-resolution microscopy images.

Specific objectives include:

1. **To utilize transfer learning** by implementing DenseNet201, a pre-trained convolutional neural network, to classify breast tissue samples as benign or malignant.

2. **To apply image preprocessing and augmentation techniques** to enhance the quality, variety, and size of training data, thereby reducing overfitting and improving generalization.

3. **To fine-tune DenseNet201** on the BreakHis 400x dataset and evaluate its performance using metrics like accuracy, precision, recall, and confusion matrix.

4. **To compare the performance** of DenseNet201 with other deep learning architectures (ResNet101 and InceptionV3) to justify its effectiveness.

5. **To demonstrate the feasibility** of using deep learning models in real-world clinical decision support systems for cancer diagnosis.

1.3 SCOPE OF THE PROJECT

This project explores the application of advanced computer vision techniques in the healthcare domain, particularly focusing on breast cancer detection through histopathological image analysis. The scope includes the following:

- **Dataset**: The model is trained and tested using the *BreakHis 400x* dataset, which contains high-resolution microscopic images of benign and malignant breast tumors.

- **Model Architecture**: DenseNet201 is used as the core model due to its dense connectivity and high feature-reuse capability, which makes it suitable for medical image analysis.

- **Image Classification Task**: The project focuses solely on binary classification—distinguishing between benign and malignant classes—based on microscopic tissue images.

- **Implementation Platform**: The model is developed using Python, Keras, and TensorFlow in a Google Colab environment.

- **Performance Evaluation**: Model performance is evaluated on unseen test data using metrics such as training/validation accuracy, confusion matrix, false positives/negatives, and graphical

- visualization (accuracy/loss curves).

- **Comparative Analysis**: The DenseNet201-based model is compared with ResNet101 and InceptionV3 models to analyze strengths, weaknesses, and suitability for histopathological classification tasks.

- **Real-world Relevance**: The project addresses challenges in manual histological analysis, such as human error and subjectivity, and provides a foundation for developing AI-assisted diagnostic systems in hospitals and diagnostic labs.

CHAPTER 2

PROBLEM DEFINITION

Problems in Breast Cancer Detection

Finding breast cancer using traditional methods is hard. Doctors can make mistakes because the images are complicated and sometimes unclear. The quality of the images varies, making it harder to be accurate. Small problems are easy to miss, and checking many images takes a lot of time and money. Different doctors may also give different diagnoses.

Solutions with Deep CNN integrated with Dense Net 201 Models

Deep convolutional neural networks (CNNs) address the above problems by offering a more objective, efficient, and timely diagnostic approach, which increases survival chances. CNNs reduce subjectivity by providing standardized, data-driven analyses, minimizing human error. They handle variations in image quality effectively by introducing non-linearity through ReLU activation functions, which enhance high-level features to ensure robust detection. CNNs are adept at recognizing subtle patterns, improving the detection of early-stage cancer and precancerous changes. Their ability to process images quickly accelerates the diagnostic process, overcoming time and resource constraints.

The current diagnostic process for breast cancer relies heavily on a combination of methods, including:

- Mammography: While effective, mammography has limitations, particularly in women with dense breast tissue where it can be difficult to distinguish between benign and malignant masses.
- Clinical Breast Examination: This physical examination by a healthcare professional can detect lumps or other abnormalities, but its sensitivity can vary depending on the clinician's expertise and the patient’s breast tissue density.

- Histopathological Analysis: This involves microscopic examination of tissue samples obtained through biopsy. While considered the gold standard for diagnosis, manual analysis by pathologists is a time-consuming and subjective process.

To overcome the limitations of manual histopathological analysis, DenseNet-201, a powerful CNN architecture, offers a compelling solution. DenseNet-201 improves feature propagation by connecting each layer to every other layer in a feed-forward fashion, enabling efficient feature reuse and reducing the number of parameters. This architectural advantage allows DenseNet-201 to extract intricate patterns in breast tissue samples with high accuracy. Its dense connectivity also helps mitigate the vanishing gradient problem, leading to better performance in deep networks compared to traditional models like ResNet-101.

By leveraging DenseNet-201 for the automated analysis of histopathological images, we aim to develop a more accurate and reliable Computer-aided Diagnosis (CAD) system that supports early detection and timely treatment of breast cancer.

Challenges in Current Diagnostic Practices

Subjectivity and Inter-observer Variability: Manual interpretation of histopathological slides is inherently subjective. Different pathologists may interpret the same tissue sample differently, leading to inconsistencies in diagnosis. This variability can be influenced by factors such as experience level, fatigue, and cognitive biases. Time-consuming and Labor-Intensive: The manual analysis of histopathological slides is a time-consuming process, requiring pathologists to meticulously examine numerous tissue sections under a microscope. This can lead to significant delays in diagnosis, particularly in settings with limited access to experienced pathologists or high patient volumes. Limitations in Detecting Early-Stage Cancers: Early-stage breast cancers often present with subtle cellular changes that can be difficult for even experienced pathologists to detect. These subtle changes may be overlooked during manual examination, potentially leading to delayed diagnosis and missed treatment opportunities. Resource Constraints: The availability of skilled pathologists varies significantly across geographical regions and healthcare systems. Many regions, particularly in developing

countries, face a severe shortage of trained pathologists, creating significant bottlenecks in the diagnostic process. Emerging Challenges: The increasing complexity of breast cancer subtypes and the emergence of new treatment modalities necessitate more precise and nuanced diagnostic information. Current diagnostic methods may not always provide the level of detail required for optimal treatment planning and personalized medicine.

These challenges in breast cancer detection have significant implications for patient outcomes:

Delayed Diagnosis and Treatment: Delayed diagnosis can allow the cancer to progress to more advanced stages, making treatment more challenging and reducing the chances of successful outcomes. Increased Morbidity and Mortality: Delayed or inaccurate diagnosis can lead to increased morbidity and mortality rates among breast cancer patients. Increased Healthcare Costs: Delayed diagnosis can result in more complex and expensive treatment options, increasing the overall healthcare costs associated with breast cancer. Psychological Impact: The uncertainty and anxiety associated with a delayed or inaccurate diagnosis can have a significant negative impact on patients' mental and emotional well-being. Addressing these challenges requires innovative approaches that can improve the accuracy, efficiency, and accessibility of breast cancer diagnosis. Technological advancements in artificial intelligence (AI) and deep learning offer promising solutions to enhance the current diagnostic paradigm. CNN: Convolutional Neural Networks (CNNs) have demonstrated exceptional performance in image recognition tasks, making them well-suited for analyzing histopathological images. CNNs can effectively learn complex patterns and features within the tissue, such as subtle changes in cell morphology, nuclear characteristics, and tissue architecture, which may be indicative of malignancy. DenseNet-201 Architecture: The DenseNet-201 architecture, a state-of-the-art CNN model, is particularly promising for this application. DenseNet-201 addresses limitations found in earlier models by introducing dense connectivity, where each layer receives input from all preceding layers. This not only improves information flow and feature reuse but also helps prevent vanishing gradients during training. Its ability to learn intricate and hierarchical features makes it highly effective at extracting subtle and nuanced characteristics from histopathological images-critical for the accurate detection of early-stage breast cancer.

Benefits of Using DenseNet-201:Improve diagnostic accuracy: Identify subtle malignant features that may be missed during manual examination, leading to earlier and more accurate diagnoses. Increase diagnostic efficiency: Automate the analysis process, reducing the time required for pathologists to review slides and increasing diagnostic throughput. Enhance diagnostic consistency: Minimize inter-observer variability by providing consistent and objective assessments of histopathological images. Assist pathologists: Provide valuable insights to pathologists, enabling them to focus their attention on areas of concern and make more informed diagnostic decisions.

Conclusion: The early detection of breast cancer is crucial for improving patient outcomes and increasing survival rates. However, current diagnostic methods face significant challenges, including subjectivity, time constraints, and limitations in detecting subtle early-stage cancers. By leveraging the power of deep learning—specifically CNNs and the DenseNet-201 architecture—we can develop AI-powered systems that address these challenges and potentially revolutionize the landscape of breast cancer diagnosis. Our problem description now explicitly addresses how CNNs and DenseNet-201 can contribute to solving the challenges in early breast cancer detection.

CHAPTER 3

LITERATURE REVIEW

Paper 1: Cancer Unveiled: Breast Tumor Detection Using Deep Learning

The study *"Cancer Unveiled: A Deep Dive into Breast Tumor Detection Using Cutting-Edge Deep Learning Models"*, published on **21 November 2023**, explores the use of advanced CNN architectures, including InceptionV3, ResNet152V2, MobileNetV2, VGG-16, and DenseNet-121, for breast tumor detection. DenseNet-121 achieved the highest accuracy of 99%, outperforming other architectures. The dataset consisted of histopathological images, which were preprocessed through noise reduction, resizing to **50x50 pixels**, normalization to [0,1], and statistical analysis. Data augmentation was applied to enhance diversity and prevent overfitting, with the dataset divided into training, validation, and testing subsets. Pretrained models were evaluated using precision, recall, F1 score, and cross-entropy loss. The study highlights the effectiveness of DenseNet-121 and the importance of robust preprocessing in medical image analysis.

Paper 2: Deep Learning Based Methods for Breast Cancer Diagnosis

The paper *"Deep Learning Based Methods for Breast Cancer Diagnosis: A Systematic Review and Future Direction"*, published on **3 January 2023**, explores various deep learning techniques for breast cancer diagnosis. The study primarily employs Convolutional Neural Networks (CNNs) and Restricted Boltzmann Machines (RBM). RBM is used to generate unbiased estimates of maximum likelihood learning, with contrastive divergence (CD) applied during training to optimize the process. Additionally, Autoencoders (AEs) are utilized to replicate input values by reducing data dimensions, thereby minimizing error rates. The encoder converts input data into hidden features, which are then reconstructed by the decoder. This method enhances feature extraction while avoiding issues associated with handcrafted features. The paper highlights the effectiveness of these approaches in advancing breast cancer diagnosis through automated learning techniques.

Paper 3: Artificial Intelligence-Based Mitosis Detection in Breast Cancer Histopathology Images

The paper *"Artificial Intelligence-Based Mitosis Detection in Breast Cancer Histopathology Images Using Faster R-CNN and Deep CNNs"*, published on **10 March 2020**, investigates the use of AI techniques for mitosis detection in breast cancer histopathology images. The study employed **Faster R-CNN** and **deep CNNs** as core technologies. The dataset comprised two open datasets: **ICPR 2012** and **ICPR 2014 (MITOS-ATYPIA-14)**. Handcrafted features such as color, morphology, and texture were extracted from regions of interest (ROIs) using conventional image-processing techniques. These features were classified using algorithms like an **Artificial Neural Network (ANN)** and a **Support Vector Machine (SVM)**. Further, independently trained **ResNet-50** and **DenseNet-201** models were used, and their scores were fused for final classification. This staged process effectively distinguished mitotic and non-mitotic cells, demonstrating the potential of combining handcrafted and deep learning-based features for enhanced accuracy in breast cancer histopathology analysis.

Paper 4: A Multi-Phase Deep CNN-Based Mitosis Detection Framework for Breast Cancer Histopathological Images

The paper *"A Multi-Phase Deep CNN-Based Mitosis Detection Framework for Breast Cancer Histopathological Images"* presents a robust method for mitosis detection using a CNN classifier named **Mito Res-CNN** to filter false mitoses. The study utilized the **TUPAC16 dataset**, which consisted of **656 images collected from 73 patients**. The dataset was divided into training, validation, and testing subsets, with **cross-validation** performed to ensure reliability. The images were preprocessed and normalized before being fed into the model. This structured approach demonstrated the effectiveness of Mito Res-CNN in reducing false positives and improving mitosis detection accuracy in breast cancer histopathology analysis.

Paper 5: Deep Learning Applied for Histological Diagnosis of Breast Cancer

The paper *"Deep Learning Applied for Histological Diagnosis of Breast Cancer"*, published on **3 September 2020**, explores the application of CNNs, specifically **ResNet50** and **DenseNet121**, achieving an accuracy of **98%**. The study utilized the **BreakHis histopathological breast cancer dataset** to evaluate image classification algorithms and state-of-the-art deep learning methods. The research focused on selecting the most suitable algorithm for image recognition and designing a new CAD system tailored to the problem. A preprocessing technique was developed for dataset preparation and experimentation, followed by the definition and optimization of hyperparameters for the specific task. The chosen models were implemented with the optimized preprocessing techniques and hyperparameters for breast cancer detection. Extensive testing and analysis of the models were conducted to achieve the highest accuracy, demonstrating the effectiveness of the proposed framework.

Paper 6: Prediction of Breast Cancer: Comparative Review of Machine Learning Techniques and Their Analysis

The paper *"Prediction of Breast Cancer: Comparative Review of Machine Learning Techniques and Their Analysis"*, published on **14 August 2020**, explores various **Machine Learning, Deep Learning, and Data Mining techniques** for breast cancer prediction. The study analyzed datasets including the **Wisconsin Diagnostic Breast Cancer (WDBC)** and **MIAS database**. The research reviewed linear methods like **Linear Regression** and **Logistic Regression**, nonlinear methods such as **Naive Bayes** and **SVM**, and ensemble methods like **Random Forest** and **AdaBoost**. It compared these methods based on accuracy and performance, highlighting combinations like linear-nonlinear and nonlinear-ensemble approaches. SVM achieved **99% accuracy** with linear kernels and **97.13%** using Weka with cross-validation. **Random Forest** and **CNN** achieved accuracies of **92.2%** and **97%**, respectively. The study concluded with a detailed comparison to identify the most effective method for breast cancer prediction.

Paper 7: A Hybrid Dependable Deep Feature Extraction and Ensemble-Based Machine Learning Approach for Breast Cancer Detection

The paper *"A Hybrid Dependable Deep Feature Extraction and Ensemble-Based Machine Learning Approach for Breast Cancer Detection"*, published on **14 August 2023**, employs **deep learning with ResNet50V2** and **ensemble methods**, specifically **Light Gradient Boosting (LGB)** for classification. The study utilized the **Invasive Ductal Carcinoma (IDC)** dataset, consisting of histopathological breast cancer images. The images were resized to a fixed dimension of **224x224 pixels** for consistency, and the color space was converted from **BGR to RGB** for standardization. Sharpening filters were applied to enhance image details, and pixel values were normalized to ensure uniformity. Labels were encoded into numeric formats to make them compatible with machine learning models. This comprehensive preprocessing strategy helped improve the classification performance for breast cancer detection using deep learning and ensemble techniques.

Paper 8: Deep Learning-Based Multi-Modal Ensemble Classification Approach for Human Breast Cancer Prognosis

The paper *"Deep Learning-Based Multi-Modal Ensemble Classification Approach for Human Breast Cancer Prognosis"*, published on **10 August 2023**, presents a novel approach for breast cancer prognosis prediction using multiple deep learning and machine learning techniques. The study utilized the publicly available **METABRIC dataset**, which includes **1980 breast cancer patient records** with multi-modal data, such as gene expression, clinical information, and copy number variation (CNV). The **mRMR algorithm** was applied for feature selection to reduce the dimensionality of the high-dimensional, low-sample-size (HDLSS) data. **Convolutional Neural Networks (CNNs)** were used for feature extraction from clinical and gene expression data, while **Deep Neural Networks (DNNs)** were employed for CNV data. The extracted features were combined into a stacked feature set and classified using the **Random Forest** algorithm. This hybrid model outperformed existing benchmarks, achieving higher accuracy, sensitivity, precision, and F1 score, significantly improving breast cancer prognosis prediction.

Paper 9: A Novel Deep Learning Approach for Accurate Cancer Type and Subtype Identification

The paper *"A Novel Deep Learning Approach for Accurate Cancer Type and Subtype Identification"*, published on **2 July 2024**, presents a robust methodology for cancer classification using **pre-trained Convolutional Neural Networks (CNNs)** combined with **Machine Learning (ML) classifiers** such as **KNN** and **SVM**, along with **Deep Learning (DL) classifiers**. The study utilized a secondary dataset from **Kaggle**, comprising over **130,000 images** representing **8 main cancer types** and **26 subtypes**. A series of image preprocessing techniques, including **morphological operations, histogram equalization, Gaussian blur**, and **Fourier transforms**, were applied to improve data quality. **CNNs** were combined with **Long Short-Term Memory (LSTM)** networks to capture both spatial and temporal cancer features, enhancing classification accuracy. The study also explored **merge CNN frameworks** and **multimodal fusion** with LSTM, using an **X-OR gate-based fusion method** to improve precision and minimize errors. **Principal Component Analysis** and **k-Nearest Neighbors (KNN)** were applied for better classification of lymphoma subclasses.

Paper 10: Hybrid Ensemble Deep Learning Model for Advancing Breast Cancer Detection and Classification in Clinical Applications

The paper *"Hybrid Ensemble Deep Learning Model for Advancing Breast Cancer Detection and Classification in Clinical Applications"*, published in **2023**, introduces a novel hybrid model combining **Gaussian Blur**, the **Ensemble Deep Random Vector Functional Link Neural Network (edRVFL)** algorithm, and **YOLOv5** for breast cancer detection. The study used a dataset of **5000 four-view mammography images** (totaling **20,000 images**) with a **70% training**, **20% testing**, and **10% validation** split. Data preprocessing involved **morphological erosion** to enhance images, reduce noise, and simplify the image complexity. The model integrates **deep learning** and **ensemble learning** concepts using **DenseNet-121** for feature extraction and **Random Forest (RF)**, **Logistic Regression (LR)**, and **Support Vector Machine (SVM)** classifiers for testing. This hybrid approach improves the performance of breast cancer detection and classification in clinical applications.

Literature Gap:

Traditional breast cancer diagnosis relies heavily on pathologists' expertise, which introduces subjectivity, human error, and diagnostic delays. While convolutional neural networks (CNNs) like ResNet 101 and DenseNet have shown promise in automating image analysis, there remains a gap in leveraging deeper and more advanced architectures such as **DenseNet-201**, which, due to its dense connectivity and depth, is particularly effective in capturing complex and subtle morphological features. In this study, we focus on the **BreakHis dataset at 400X magnification**, which provides the highest level of image detail, essential for identifying fine-grained structures in breast tissue. Many existing methods fail to tailor their models for specific resolutions or exploit the full potential of high-magnification images, limiting diagnostic accuracy. DenseNet-201's ability to promote feature reuse and maintain strong gradient flow makes it well-suited for such detailed analysis. However, limitations persist, including a reliance on single-resolution datasets, lack of multi-modal data integration (e.g., clinical and histological data), and limited use of explainable AI techniques. Incorporating DenseNet-201 with 400X BreakHis data, along with explainability and data fusion strategies, can significantly improve the accuracy, interpretability, and clinical utility of breast cancer diagnosis systems.

CHAPTER 4

PROJECT DESCRIPTION

Traditional breast cancer diagnosis through histopathology relies on the expertise of pathologists, making it prone to human error and delays, particularly when detecting early-stage or small cancers. Variations in image quality, such as differences in magnification and staining, can further complicate the interpretation of histopathology. Additionally, different pathologists may interpret the same images in various ways, leading to inconsistent diagnoses. Deep CNNs address these challenges by providing a more objective and efficient method of image analysis. CNNs are trained to recognize patterns in histopathology images, handling variations in quality and enhancing features for better accuracy. Among them, **DenseNet-201** stands out due to its dense connectivity and ability to reuse features, allowing it to detect subtle abnormalities and early-stage cancers that might be missed by the human eye. DenseNet-201 also accelerates the diagnostic process by efficiently analyzing large volumes of high-resolution images. With its consistent, data-driven approach, DenseNet-201 ensures more reliable diagnoses and minimizes human error. This leads to faster, more accurate diagnostic decisions and ultimately better outcomes for patients by improving early detection and enabling timely treatment, thereby enhancing survival rates.

4.1 PROPOSED DESIGN

- **Input Data:** Raw data is provided as input to the model.
- **Data Preprocessing:** Preprocessing techniques (e.g., resizing, normalization, augmentation) are applied to prepare the data for training or inference, ensuring compatibility with the model.
- **DenseNet-201 Convolution Layer:** A DenseNet-201 with 201 layers is used as the backbone of the deep neural network. DenseNet-201 is a powerful model that uses connections to solve the vanishing gradient problem, enabling deep networks to train

effectively.

- **Feature Extraction Layer:** The model extracts meaningful features from the input data through convolutional operations. This step focuses on identifying key patterns in the images, such as shapes, textures, and edges.
- **ReLU Activation:** After extracting features from the images, the model uses a ReLU (Rectified Linear Unit) activation function to help it learn complex patterns. ReLU works by keeping positive values as they are and setting negative values to zero, making the model more efficient and effective without adding much computational cost.
- **Fully Connected Layer:** The features are flattened and passed to one or more fully connected layers to perform high-level reasoning and combine extracted features.
- **Soft max Activation Layer:** The final output is normalized using the Softmax activation function, which converts the model's raw predictions into probabilities for each class
- **Classification Result:** Based on the probability scores, the model classifies the input into one of the following categories:
- **Benign:** Non-malignant tumors or abnormalities. It refers to non-cancerous growths or tumors that, while abnormal, do not spread to other parts of the body or invade nearby tissues. These are typically not life-threatening but may require monitoring or removal if they cause discomfort or complications
- **Malignant:** Cancerous and harmful tissues. It is tissue signifies cancerous growths that can invade nearby tissues and spread (metastasize) to other parts of the body, posing serious health risks. Accurate classification of these categories is critical for determining the appropriate medical intervention and ensuring timely treatment.

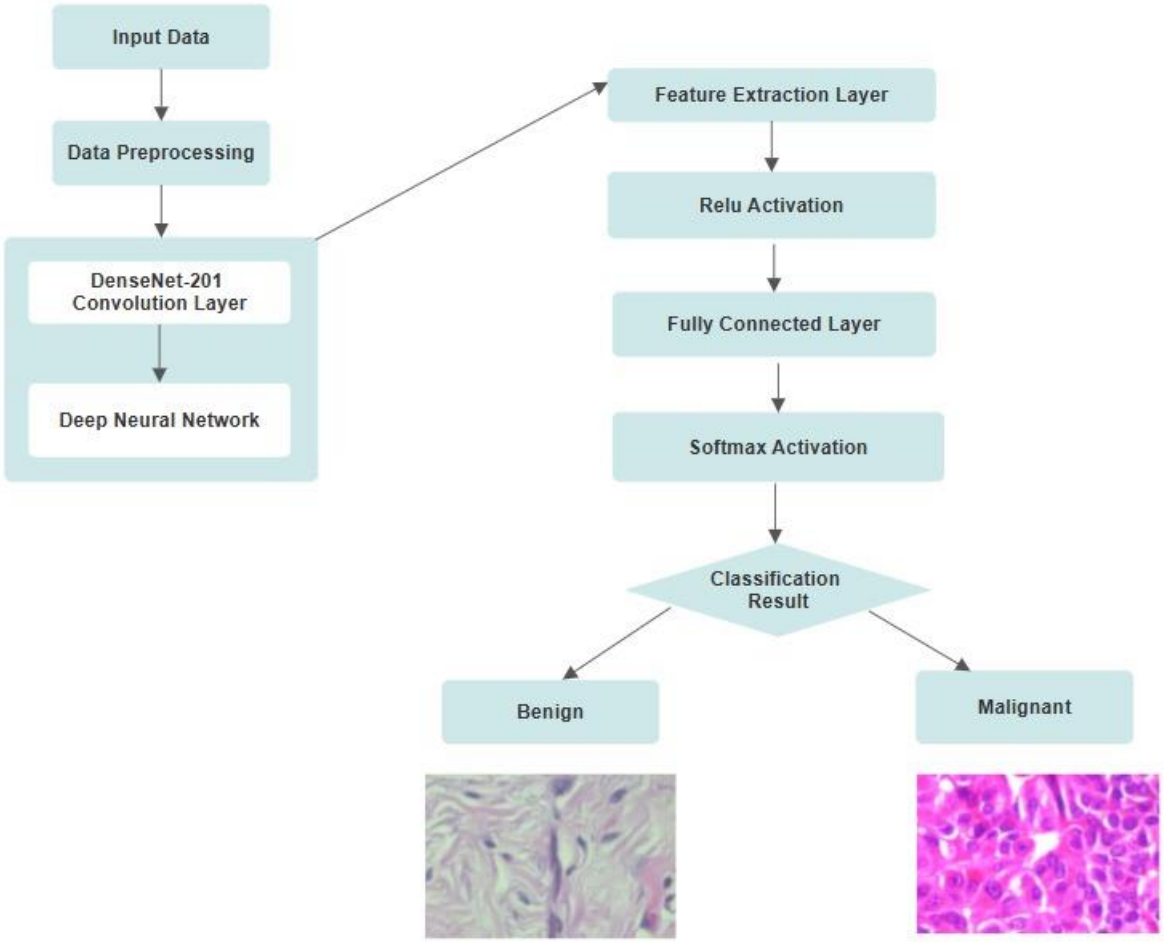

Fig 4.1 Proposed Design

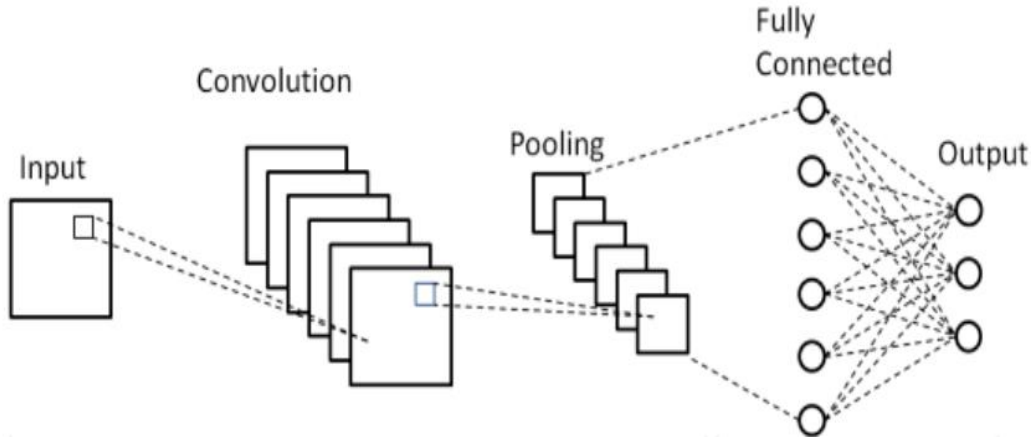

Fig 4.2 CNN Architecture

4.2 CNN ARCHITECTURE

Input Layer :

In this layer, the CNN takes the input of raw data(in our case it is dataset of breast cancer images) and provides data to the next layer for data preprocessing.

Convolutional Layers:

The convolution layer scans the image with small filters (i e kernel) to detect simple features like edges, shape, textures of the cells in the dataset. Each filter moves across the whole image and calculates a new feature map.

$$\text{Feature Map} = \sum(I * K) + b \quad (1)$$

Where:

I: Input image

K: Kernel (filter)

b: Bias term

The output of these layers is feature maps and the size of the feature map depends on stride and kernel.

Pooling Layers

This layer reduces the size of the image while keeping its most important features such as cell structure, pattern or edges of an image. Breast cancer images are usually large, with lots of details. Pooling layer reduces the size of these images to make the training and testing more faster and efficient.

There are various types of pooling:

Average Pooling: Computes the average of values in a region.

Max Pooling: Takes the maximum value in a region

Activation Layers:

An activation function decides whether a neuron in the network should "fire" (i.e., pass its output forward). It takes the information from the previous layers, applies a rule, and determines what should be passed to the next layer. This helps the network learn complex patterns in the data by introducing non-linearity, making the training process more effective.

ReLU Activation:

ReLU (Rectified Linear Unit) is a widely used activation function. It works by keeping all positive values as they are, while turning any negative values into zero. This allows the model to focus only on positive features, helping it learn faster and more efficiently.

$$f(x) = \max(0,x) \quad (1)$$

Where:

- f(x): It implies function of a variable x.
- max(0, x): This expression gives the maximum value between 0 and x.

Advantages:

Prevents Dead Neurons: One issue with some activation functions is that neurons can "stop learning" if they always output negative values. ReLU avoids this by turning negative values into zero, keeping the neurons active and helping the model keep learning.

Batch Normalization: ReLU also works well with batch normalization, which helps speed up the training process and makes the model more stable during learning.

Fully Connected (Dense) Layers:

After all the feature maps are flattened into a 1D vector, this layer connects all the features together to make decisions. Each neuron is connected to every neuron of the previous layer.It processes all the learnt patterns from the previous layers and combines them to predict the final output.

Softmax Layer (Output Layer)

The softmax layer plays an important role in classification tasks. Its key role is to transform raw output scores into probabilities that sum up to 1, representing cancerous or noncancerous of each class prediction.

$$s(x_i) = \frac{e^{x_i}}{\sum_{j=1}^{n} e^{x_j}} \quad (2)$$

where:

$s(x_i)$: The output probability for class ii after applying the softmax function.

x_i: The input (also called the logit or score) for class ii before softmax is applied.

e^{x_i}: The exponential of the input score x_i, used to make all values positive and scale them non-linearly.

$\sum_{j=1}^{n} e^{x_j}$: The sum of exponentials of all input scores x_j for all classes j=1 to n. This acts as a normalization factor.

n : The total number of classes.

j : An index variable used to iterate over all classes

.

CHAPTER 5

REQUIREMENTS

5.1 FUNCTIONAL REQUIREMENT

• **Image Data Acquisition:** The system must be able to acquire and process breast histopathology images at different magnifications for analysis.

• **Image Preprocessing:** Perform necessary preprocessing steps on the input images such as resizing, normalization, and noise reduction to enhance image quality for accurate analysis.

• **Mitosis Detection:** Implement deep convolutional neural networks (CNNs) to detect the presence of mitosis in histopathology images, as mitotic activity is a key indicator of abnormal cell growth.

• **Cancer Cell Identification:** The system must classify the histopathology images into benign or malignant categories based on the presence of abnormal cells or mitosis.

• **Multi-scale Image Evaluation:** The system must support analysis at different magnifications to allow a detailed examination of cell structure and shape, assisting in detecting cancerous tissues.

• **Model Training and Optimization:** Use labeled histopathology datasets to train the CNN model. Optimize the model to improve detection accuracy, minimize false positives and false negatives, and enhance overall performance.

5.2NON-FUNCTIONAL REQUIREMENTS

- **Performance:** The system should be fast enough to analyze histopathology images and provide results within a reasonable time frame.
- **Throughput:** It needs to handle multiple images at once without slowing down or affecting performance.

- **Accuracy:** The system should be highly accurate, aiming for at least 95% precision and recall when identifying cancer and mitosis. It should minimize errors like false positives and false negatives, ensuring the results are reliable when distinguishing between benign and malignant samples.
- **Scalability:** As more users and images are added, the system must be able to grow without issues. It should be able to handle larger volumes of data, and if necessary, work across multiple systems or locations like hospitals.
- **Availability:** The system should be up and running most of the time, with minimal downtime. A target uptime of 99.9% is ideal to ensure that healthcare professionals can always access it when they need it.
- **Maintainability:** The system should be built in a way that makes it easy to update and maintain. Changes to the algorithms, models (like DenseNet-201), or other components should be straightforward. Clear documentation should be available for anyone who needs to work on the system.
- **Reliability:** The system should provide consistent, dependable results across different datasets, magnifications, and hardware setups. It should also be able to recover quickly from any minor issues without interrupting the process.
- **Efficiency:** The system should be resource-efficient, using CPU, GPU, and memory wisely to ensure it works smoothly even on machines with limited resources. It should also be designed to reduce unnecessary computation time.
- **Compliance:** The system needs to meet all necessary medical regulations and certifications to be used in healthcare settings. It should also follow legal and ethical guidelines when using AI for diagnosing medical conditions.

5.3. SOFTWARE/SYSTEM REQUIREMENTS

Hardware Requirement

- **RAM:** Minimum: 16 GB for larger medical image datasets (histopathological images), consider 32 GB or more.
- **Storage:** SSD storage is preferred for faster data loading. Storage size depends on dataset size. Medical imaging datasets can be large (100s of GBs or more), so having at least 1 TB of storage is recommended.

Software Requirements:

- **Operating System:** Windows 10 or above
- **Deep Learning Frameworks:** TensorFlow, PyTorch, or Keras for model implementation and training.
- **Image Processing:** CNN integrated with Dense Net 201 for processing and extraction of high-level features histopathology images.

Simulation:

- **Google Colab :** the platform for simulation due to its free access to GPUs and ease of use in a cloud environment.
- **Python 3.11 :** It provides the latest features and performance improvements for coding and libraries.
- **TensorFlow 2.x:** It is used for building and training deep learning models

Platform	Specification
Google Colab	-
Python Version	3.11
TensorFlow Version	2.x

Data and Preprocessing:

- **Dataset Used:** BreaKHis (400x magnification) – a histopathological breast cancer image dataset.
- **Image Size:** All images resized to 224x224 pixels to match model input requirements.
- **Batch Size**: Set to 16 to balance memory usage and training stability.
- **RGB Channels:** Used, indicating color images are preserved for model training.
- **Epochs:** The model was trained for 5 epochs, sufficient for initial evaluation.
- **Train/Validation Split**: 80% of the data used for training and 20% for validation to assess model generalization.

Item	**Specification**
Dataset	BreaKHis 400x
Image Size	224x224
Batch Size	16
RGB Channels	Yes
Epochs	5
Train/Val Split	80%/20%

Python Libraries:

- **Data Handling – Pandas, NumPy:** Used to load, clean, and manipulate structured data. Ensures efficient numerical computation and data preprocessing.

- **Data Visualization – Matplotlib:** Used to plot training metrics and data distributions. Helps visualize model performance and understand dataset trends.

- **Image Processing – OpenCV, Pillow (PIL), TensorFlow Keras:** Handles image resizing, normalization, and augmentation (e.g., flipping, rotating).Prepares images for input into the model and improves generalization.

- **Model Training – TensorFlow, Keras, DenseNet201:** Builds and trains deep learning models using the DenseNet201 architecture. Provides a powerful framework for developing and optimizing neural networks.

- **Performance Metrics – Scikit-learn (Sklearn):** Calculates accuracy, precision, recall, F1-score, and confusion matrix.Used to evaluate and validate the effectiveness of the trained model.

Algorithm	**Package Name**
Data Handling	Pandas, NumPy
Data Visualization	Matplotlib
Image Processing	OpenCV, Pillow, TensorFlow Keras
Model Training	TensorFlow, Keras, DenseNet201
Performance Metrics	Sklearn

CHAPTER 6

METHODOLOGY

CNN Integrated with DenseNet-201

Convolutional Neural Networks (CNNs) are deep learning models designed to process and analyze data like images and text. These networks automatically learn spatial patterns, such as edges, cell textures, shapes, and complex features, by applying filters to the input image (like a breast cancer dataset). CNNs are made up of multiple layers, each performing a specific task such as feature extraction, dimensionality reduction (keeping only the essential details), pooling, and introducing non-linearity to enhance the model's ability to make accurate predictions.

DenseNet-201

DenseNet is an advanced variation of CNN designed to address some of the challenges faced by deep networks. DenseNet, like CNNs, is built to process images, but it differs by using a technique called *dense connections*. In DenseNet, each layer is directly connected to every other layer in a way that improves the flow of information and gradients throughout the network. This method helps mitigate common issues like the vanishing gradient problem, allowing for better learning across all layers, even in very deep networks.

DenseNet comes in different configurations depending on the complexity and the task at hand. In particular, DenseNet-201 is a deeper and more powerful version of DenseNet, with 201 layers, making it suitable for complex image analysis tasks. This architecture not only captures a wide range of features but also ensures that important information is preserved and propagated efficiently across the layers.

DenseNet-121: This model is the lightest in the DenseNet family, with 121 layers. It’s designed to be more efficient by allowing each layer to receive input from all previous layers. This means it can reuse features and reduce the number of parameters compared to other models. DenseNet-

121 works well for smaller datasets and tasks that don't require heavy computing power, like real-time image classification or small-scale medical image analysis. However, it can struggle with larger datasets because its depth is limited, which can result in underfitting and lower accuracy when handling more complex tasks.

DenseNet-169: Building on the DenseNet-121, this model has 169 layers, allowing it to capture more intricate features in the data. It performs better in tasks that need intermediate complexity, such as feature extraction in images. DenseNet-169 improves on DenseNet-121 by providing more accuracy and handling moderately complex datasets better. But it's still not ideal for very large datasets, especially when you need to extract high-level, detailed features from the data.

DenseNet-201: DenseNet-201 takes it a step further with 201 layers, making it a more powerful model for handling more complex and larger datasets. The extra depth allows the model to recognize more intricate patterns and high-level features in the data, making it well-suited for tasks like medical image analysis. In particular, DenseNet-201 performs well when analyzing high-resolution images, where it can detect fine details that are important for tasks like identifying cancerous cells. Its deeper layers and efficient feature reuse help it provide higher accuracy compared to the lighter versions.

DenseNet-264: This version has an even greater depth with 264 layers, making it the most advanced model in the DenseNet family. It's designed to handle extremely large and complex datasets, where detecting very subtle differences in features is crucial. DenseNet-264 excels in capturing fine details, making it especially useful for high-level medical image analysis like breast cancer detection. However, with this increased complexity, it requires more computing resources to train and run.

Concept	DenseNet201	InceptionV3	ResNet101
Feature Sharing	Every layer shares features with future ones	Multiple views (filter sizes) at each layer	Learns the change (residual) from one layer to the next
Growth	Features keep increasing slowly	Multiple paths processed in parallel	Layers stack one after another deeply
Efficiency	Very high — avoids repeated learning	Medium — more computation, smart design	Less efficient — more parameters and memory
Accuracy	93%	86%	70%

- **Deeper Architecture (201 Layers):** DenseNet-201 has more layers than other variants like DenseNet-121 or DenseNet-169, enabling it to learn and detect finer, more complex features in histopathological images, such as slight abnormalities in cells that may indicate cancer.
- **Efficient Feature Reuse:** Thanks to its unique design where each layer connects to all previous layers, DenseNet-201 reuses features more effectively. This helps the model learn better with fewer parameters and reduces overfitting—making it well-suited for medical image analysis.
- **Better Accuracy for Subtle Differences:** Breast cancer cells often show very subtle changes in texture, shape, or color. The depth of DenseNet-201 allows it to capture these details more accurately than shallower models.
- **Optimized for High-Resolution Images:** Breast cancer diagnosis often uses high-resolution images. DenseNet-201 handles these better because of its ability to maintain detailed feature flow across many layers, without losing important information.
- **Balanced Performance vs. Computation:** Despite being deep, DenseNet-201 is designed efficiently. It doesn't require as much memory or computation as other deep networks, making it practical for real-world medical applications.

- **Improved Generalization:** It performs well across different datasets and imaging conditions, which is important because medical data often varies in quality, magnification, and format.
- **Supports Early Detection:** By identifying very small and early-stage abnormalities, DenseNet-201 helps in catching cancer at earlier stages—when treatment is more effective and patient outcomes are better.

Dataset description

The dataset used in this study comprises 1,693 histopathological images of breast tissue obtained at 400× magnification, providing high-resolution visual detail essential for accurate diagnosis. These images capture fine-grained cellular structures such as nuclei, cytoplasm, and tissue patterns, which are critical for distinguishing between benign and malignant lesions. The dataset is divided into 1,148 training images—consisting of 371 benign and 777 malignant samples—and 545 testing images, which include 176 benign and 369 malignant samples. The class distribution reflects the natural clinical imbalance and poses a realistic challenge for classification models. The high magnification enables deep convolutional neural networks to effectively extract discriminative features, enhancing the learning of subtle visual cues associated with malignancy. This level of detail is particularly valuable for developing robust and generalizable deep learning models for breast cancer detection, making the dataset a reliable benchmark for histopathological image analysis tasks.

Algorithm:

Importing modules: Libraries such as tensorflow, keras, numpy, and matplotlib are loaded in preparation for the work to be done herein. Densenet201 is used as the basis of the classification system from keras' pre-trained models.

Reading and preparing the data: Images are loaded from the benign and malignant folders, resized to a standard width and height of 224, respective of the pixel size, and standardized for

analysis. The dataset is split into training, validation, and test sets, with label encoding applied to the target and normalization to the data.

Train Deep Neural Network with PyTorch: Densenet201 is initialized with `include_top=false` to remove the default classifier layers. Then a custom classification head is attached that uses global average pooling and dense layers with a sigmoid activation for binary outputs.

Compile the model and define the data augmentation: Compilation of the model occurs with adam as the optimizer and binary_crossentropy as the loss function. Data augmentation is carried out for the images by randomly flipping, rotating, and zooming, in an attempt to make the data less prone to overfitting.

Model fitting: The fitting of the model is accomplished on the augmented images over several iterations, or epochs. Validation accuracy is frequently monitored and the best version of the model preserved through callbacks that implement early stopping and model checkpointing.

Analysis of Results: After the training phase, the model is interrogated on the validation set. Various parameters are reported (loss and accuracy), and sample predictions are visualized to qualitatively check the level of trustworthiness in classification.

Assessment metrics: Accuracy, precision, recall, and f1-score evaluate the quantitative performance. The evaluation results provide the model's validation accuracy, approximately 93% of classification legitimacy for medical images.

CHAPTER 7

RESULT

Potential Results:

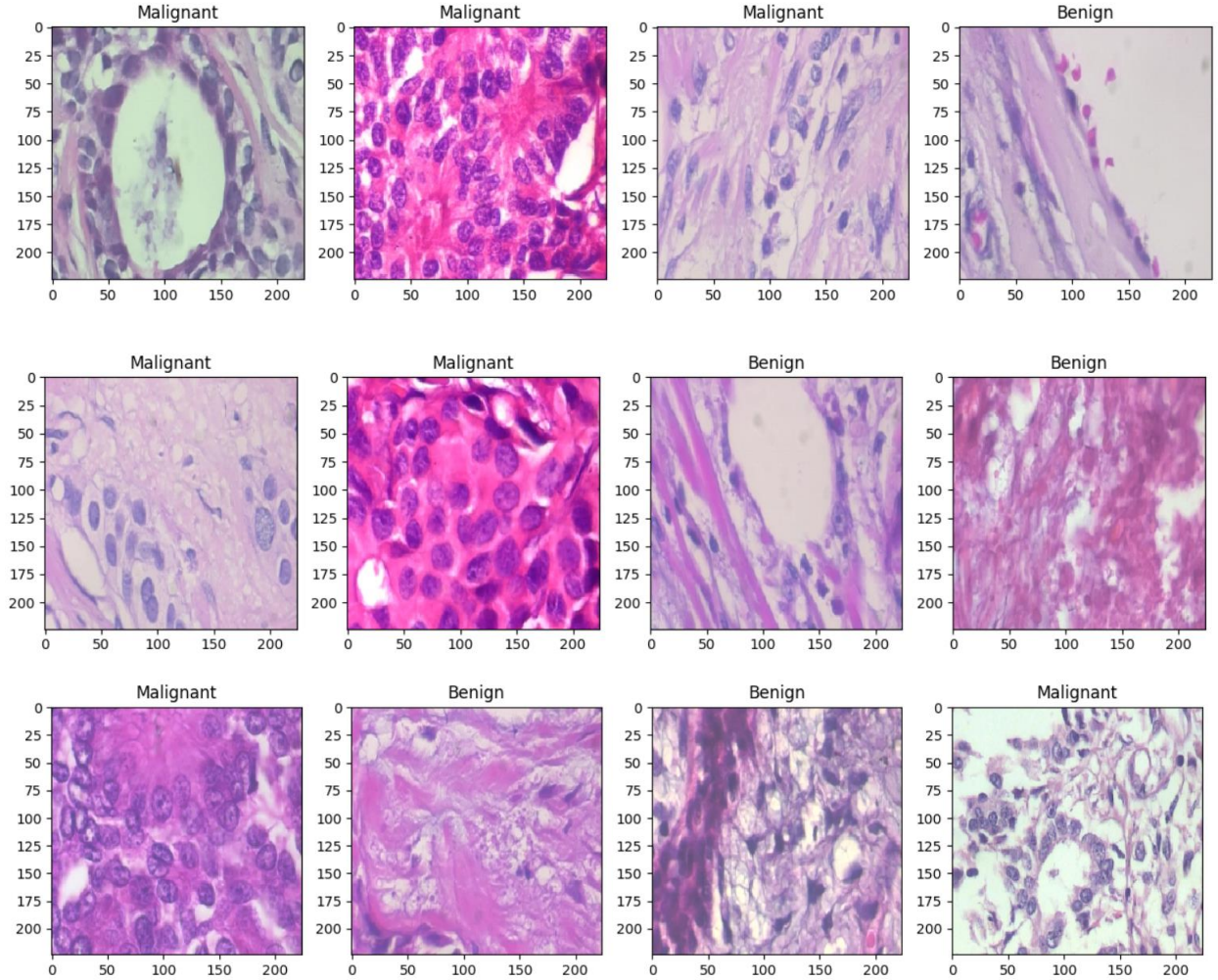

Fig 7.1 Classifying dataset as malignant or benign

The image displays a grid of histopathological breast tissue images, each labeled as either **Benign** or **Malignant**, and is part of a dataset used for classification using the deep learning model.

1. **Malignant** : These samples show messy, irregular cell patterns. The cells look abnormal and crowded, with large or strange-looking nuclei. This kind of tissue can grow fast and spread

to other parts of the body.

2. **Benign** : These samples look more normal and organized. The cells are neat, similar in size and shape, and don't invade other tissues. Benign tumors usually aren't life-threatening

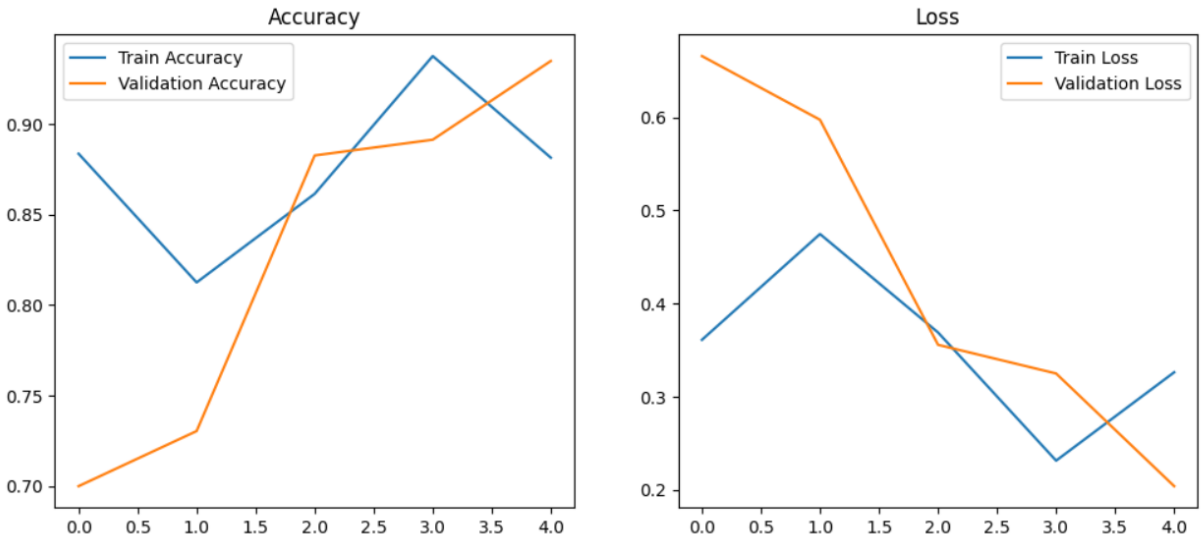

Fig 7.2 Accuracy and Loss of DenseNet 201

Accuracy :

As the training progressed over five epochs, the validation accuracy steadily increased, starting from around 70% and reaching close to 93% by the 5th epoch, which is a strong indication that the model is learning well from the data.

Loss :

Loss, which measures how far the predictions are from the actual labels, decreased consistently for both training and validation sets. The training loss dropped from about 0.35 to 0.22, while the validation loss had an even more dramatic drop from around 0.65 to just 0.20.

Fig 7.3 Accuracy and Loss in ResNet 101

Accuracy

This graph shows how the model's training accuracy improves sharply in the beginning, even reaching over 81%, but then drops and fluctuates, indicating some instability in learning. Meanwhile, the validation accuracy stays almost flat around 70% throughout all epochs. This gap between the two suggests that the model is overfitting—it's learning the training data well but isn't performing as well

Loss

This loss graph shows that while the training loss (blue line) initially drops—indicating the model is learning—it starts to increase again, peaking around the third epoch before dropping slightly. Meanwhile, the validation loss (orange line) steadily rises, ending significantly higher than the training loss. This growing gap between the two lines suggests **overfitting**, where the model is fitting the training data too closely and struggling to perform well on new data.

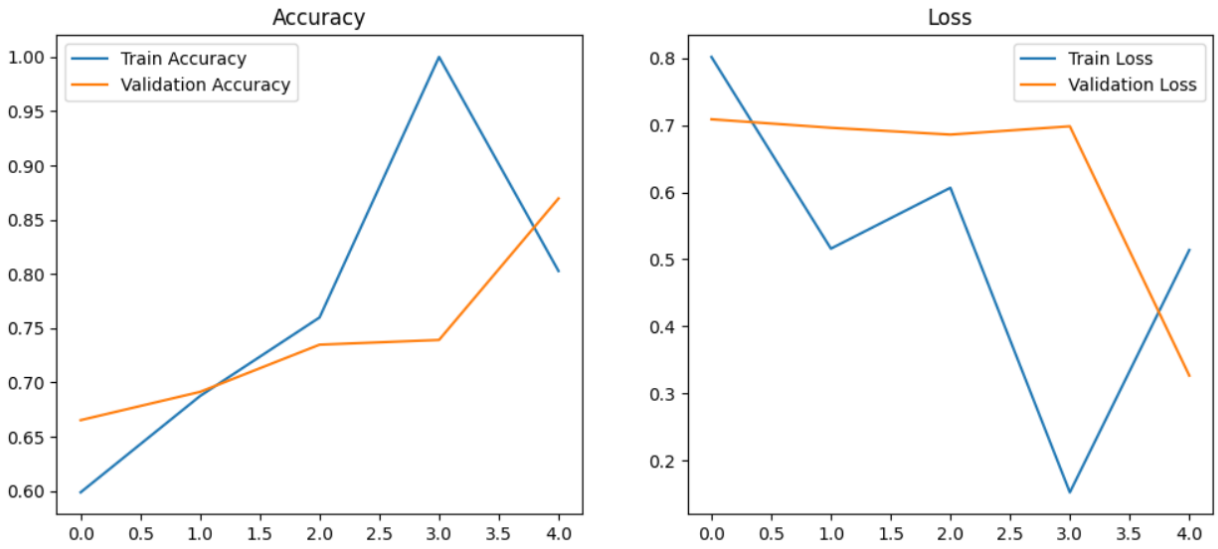

Fig 7.4 Accuracy and Loss in Inception V3

Accuracy

This accuracy graph shows a generally positive trend for both training and validation performance. The training accuracy increases sharply, reaching nearly 100% by the third epoch before dropping slightly. The validation accuracy also rises steadily, peaking around 87% by the final epoch.

Loss

This loss graph shows that the training loss steadily decreases, reaching a low point by the third epoch, which means the model is learning well. The validation loss stays mostly stable until the third epoch, then drops sharply, which is a good sign—it suggests the model is finally starting to generalize better to new data.

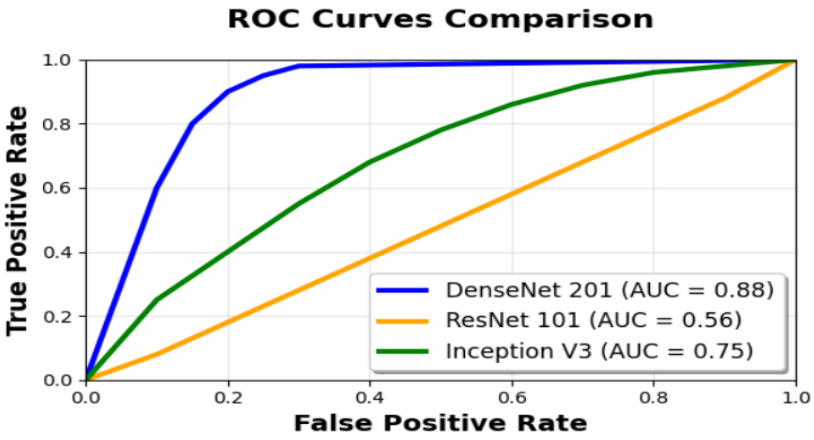

Fig 7.5 ROC Curve Comparison

ROC Curve

ROC curve comparison for three deep learning models—DenseNet 201, ResNet 101, and Inception V3—used for classification tasks. The ROC curve plots the True Positive Rate against the False Positive Rate, providing a visual assessment of a model's performance. Among the three models, DenseNet 201 shows the best performance with an AUC value of 0.88, indicating high classification accuracy. Inception V3 follows with a moderate AUC of 0.75, while ResNet 101 performs poorly with an AUC of 0.56, only slightly better than random guessing. This comparison highlights DenseNet 201 as the most effective model for the given classification problem.

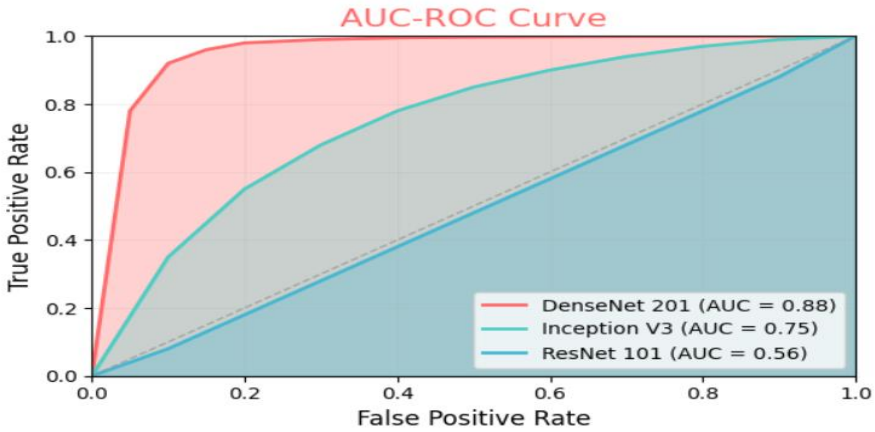

Fig 7.6 AUC-ROC Curve Comparison

AUC-ROC Curve

The image shows a comparison of classification performance for three deep learning models: DenseNet 201, Inception V3, and ResNet 101 using a curve that plots the true positive rate against the false positive rate. The curve visually represents how well each model can distinguish between classes. DenseNet 201, represented in red, shows the best performance with a score of 0.88, indicating high accuracy in predictions. Inception V3, shown in light blue, achieves a moderate score of 0.75, while ResNet 101, in blue, performs the worst with a score of 0.56, suggesting it is only slightly better than random guessing. The shaded regions under each curve highlight the difference in performance, clearly showing that DenseNet 201 is the most effective model among the three.

CHAPTER 8

CONCLUSION AND FUTURE WORK

8.1 CONCLUSION

In this project, we evaluated the performance of three advanced convolutional neural network architectures—DenseNet201, InceptionV3, and ResNet101—for breast cancer classification using transfer learning. Each model was fine-tuned with a custom classification head and trained on medical imaging data. Among the three, DenseNet201 emerged as the most effective, offering a strong balance between high accuracy, efficient parameter usage, and fast convergence—making it particularly well-suited for small and sensitive medical datasets. InceptionV3 also performed competitively, with slightly lower accuracy but significantly faster inference times, making it a good option for real-time or resource-constrained deployments. On the other hand, ResNet101, despite being a powerful and deep network, underperformed in this context. It showed slower convergence, higher risk of overfitting, and lower overall accuracy, likely due to its higher complexity and the relatively limited size of the training data. Overall, DenseNet201 proved to be the most suitable model for breast cancer classification in terms of both performance and practicality

8.2 FUTURE WORK

In the forthcoming stages of this research, a larger and more diverse histopathological image dataset will be utilized to improve the generalization capabilities of the model while reducing the risk of overfitting. Incorporating samples from various sources and patient demographics will allow the network to learn a broader range of morphological patterns, thereby increasing its applicability in real-world clinical settings. Furthermore, Explainable AI (XAI) techniques such as Grad-CAM, SHAP, or LIME will be explored not only to enhance model interpretability for clinical validation but also to guide model pruning and architecture optimization. This dual purpose aims to reduce model complexity and size without significantly compromising accuracy. Such lightweight yet interpretable models are particularly desirable for deployment in real-time

healthcare environments, including diagnostic support tools in hospitals and point-of-care applications. Moreover, the potential integration of the optimized model into mobile and edge devices will be investigated, enabling accessible and cost-effective breast cancer screening solutions in remote or resource-limited settings.

SAMPLE CODE

SAMPLE CODE FOR DENSENET-201

Import the Libraries

```
import json
import math
import os
import cv2
from PIL import Image
import numpy as np
from keras import layers
import tensorflow
from tensorflow.keras.applications import DenseNet201
from tensorflow.keras.preprocessing.image import ImageDataGenerator
from tensorflow.keras.utils import to_categorical
from keras.models import Sequential
from tensorflow.keras.optimizers import Adam
import matplotlib.pyplot as plt
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.metrics import cohen_kappa_score, accuracy_score
import scipy
from tqdm import tqdm
import tensorflow as tf
from keras import backend as K
import gc
from functools import partial
from sklearn import metrics
from collections import Counter
import itertools
```

Splitting Data Into Training and Testing:

```
benign_train = np.array(Dataset_loader('/content/review/BreaKHis 400X/train/benign',224))
malign_train = np.array(Dataset_loader('/content/review/BreaKHis 400X/train/malignant',224))
benign_test = np.array(Dataset_loader('/content/review/BreaKHis 400X/test/benign',224))
malign_test = np.array(Dataset_loader('/content/review/BreaKHis 400X/test/malignant',224))
benign_train_label = np.zeros(len(benign_train))
malign_train_label = np.ones(len(malign_train))
benign_test_label = np.zeros(len(benign_test))
malign_test_label = np.ones(len(malign_test))

X_train = np.concatenate((benign_train, malign_train), axis = 0)
Y_train = np.concatenate((benign_train_label, malign_train_label), axis = 0)
X_test = np.concatenate((benign_test, malign_test), axis = 0)
Y_test = np.concatenate((benign_test_label, malign_test_label), axis = 0)

s = np.arange(X_train.shape[0])
np.random.shuffle(s)
X_train = X_train[s]
Y_train = Y_train[s]

s = np.arange(X_test.shape[0])
np.random.shuffle(s)
X_test = X_test[s]
Y_test = Y_test[s]

Y_train = to_categorical(Y_train, num_classes= 2)
Y_test = to_categorical(Y_test, num_classes= 2)
x_train, x_val, y_train, y_val = train_test_split(
    X_train, Y_train,
    test_size=0.2,
    random_state=11)
```

Building A CNN Model Using Densenet201 for Image Classification

```
def build_model(backbone, lr=1e-4):
    model = Sequential()
    model.add(backbone)
    model.add(layers.GlobalAveragePooling2D())
    model.add(layers.Dropout(0.5))
    model.add(layers.BatchNormalization())
    model.add(layers.Dense(2, activation='softmax'))
    model.compile(
        loss='binary_crossentropy',
        optimizer=Adam(learning_rate=lr),
        metrics=['accuracy']
    )
    return model
K.clear_session()
gc.collect()
result = DenseNet201(
    weights='imagenet',
    include_top=False,
    input_shape=(224,224,3)
)
model = build_model(result ,lr = 1e-4)
model.summary()
```

Saving Best Weights During Training

```
from keras.callbacks import ModelCheckpoint
import numpy as np
from sklearn.metrics import accuracy_score


filepath = '/content/drive/MyDrive/best_model.keras'

checkpoint = ModelCheckpoint(filepath, monitor='val_accuracy', verbose=1, save_best_only=True, mode='max')

history = model.fit(
    train_generator.flow(x_train, y_train, batch_size=BATCH_SIZE),
    steps_per_epoch=int(x_train.shape[0] / BATCH_SIZE),
    epochs=5,
    validation_data=(x_val, y_val),
    callbacks=[checkpoint]
)

model.load_weights(filepath)

Y_val_pred = model.predict(x_val)
accuracy_score(np.argmax(y_val, axis=1), np.argmax(Y_val_pred, axis=1))
```

Model Accuracy And Model Loss

```
plt.figure(figsize=(12, 5))
plt.subplot(1, 2, 1)
plt.plot(history.history['accuracy'], label='Train Accuracy')
plt.plot(history.history['val_accuracy'], label='Validation Accuracy')
plt.legend()
plt.title('Accuracy')

plt.subplot(1, 2, 2)
plt.plot(history.history['loss'], label='Train Loss')
plt.plot(history.history['val_loss'], label='Validation Loss')
plt.legend()
plt.title('Loss')
plt.show()
```

Final Validation Accuracy

```
model.load_weights("/content/drive/MyDrive/best_model.keras")

Y_val_pred = model.predict(x_val)
accuracy_score(np.argmax(y_val, axis=1), np.argmax(Y_val_pred, axis=1))
```

```
/usr/local/lib/python3.11/dist-packages/keras/src/saving/saving_lib.py:757:
  saveable.load_own_variables(weights_store.get(inner_path))
8/8 ━━━━━━━━━━━━━━━━━━━━ 75s 8s/step
0.9347826086956522
```

Building Confusion Matrix

```
Y_pred = model.predict(X_test)
tta_steps = 10
predictions = []

for i in tqdm(range(tta_steps)):

    preds = model.predict(train_generator.flow(X_test, batch_size=BATCH_SIZE, shuffle=False),
                          steps = len(X_test) // BATCH_SIZE)
    predictions.append(preds)
    gc.collect()

Y_pred_tta = np.mean(predictions, axis=0)
from sklearn.metrics import confusion_matrix

def plot_confusion_matrix(cm, classes,
                          normalize=False,
                          title='Confusion matrix',
                          cmap=plt.cm.Blues):

    if normalize:
        cm = cm.astype('float') / cm.sum(axis=1)[:, np.newaxis]
        print("Normalized confusion matrix")
    else:
        print('Confusion matrix, without normalization')

    print(cm)
```

ROC Curve

```
from sklearn.metrics import roc_curve, auc


y_true = np.argmax(Y_test, axis=1)
y_pred = np.argmax(Y_pred, axis=1)

fpr, tpr, _ = roc_curve(y_true, y_pred)
roc_auc = auc(fpr, tpr)
plt.plot(fpr, tpr, label=f'AUC = {roc_auc:.2f}')
plt.xlabel('False Positive Rate')
plt.ylabel('True Positive Rate')
plt.title('ROC Curve')
plt.legend()
plt.show()
```

AUC ROC Curve

```
from sklearn.metrics import roc_curve, auc
import matplotlib.pyplot as plt
fpr, tpr, _ = roc_curve(y_true, y_pred)
roc_auc = auc(fpr, tpr)
plt.figure(figsize=(8,6))
plt.plot(fpr, tpr, color='#FF6F61', lw=2, label=f'AUC = {roc_auc:.2f}')
plt.fill_between(fpr, tpr, alpha=0.3, color='#FF6F61')
plt.plot([0, 1], [0, 1], color='grey', linestyle='--', lw=2)
plt.xlabel('False Positive Rate', fontsize=12, fontweight='bold', color='#333333')
plt.ylabel('True Positive Rate', fontsize=12, fontweight='bold', color='#333333')
plt.title(' AUC-ROC Curve', fontsize=14, fontweight='bold', color='#FF6F61')
plt.legend(loc='lower right', fontsize=12)
plt.grid(alpha=0.3)
plt.show()
```

SAMPLE CODE FOR RESNET-101

Import the Libraries

```
import json
import math
import os
import cv2
from PIL import Image
import numpy as np
from keras import layers
import tensorflow
from tensorflow.keras.applications import resnet
from tensorflow.keras.applications.resnet import ResNet101
from tensorflow.keras.preprocessing.image import ImageDataGenerator
from tensorflow.keras.utils import to_categorical
from keras.models import Sequential
from tensorflow.keras.optimizers import Adam
import matplotlib.pyplot as plt
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.metrics import cohen_kappa_score, accuracy_score
import scipy
from tqdm import tqdm
import tensorflow as tf
from keras import backend as K
import gc
from functools import partial
from sklearn import metrics
from collections import Counter
import itertools
```

Splitting Data Into Training and Testing

```
benign_train = np.array(Dataset_loader('/content/review/BreaKHis 400X/train/benign',224))
malign_train = np.array(Dataset_loader('/content/review/BreaKHis 400X/train/malignant',224))
benign_test = np.array(Dataset_loader('/content/review/BreaKHis 400X/test/benign',224))
malign_test = np.array(Dataset_loader('/content/review/BreaKHis 400X/test/malignant',224))
benign_train_label = np.zeros(len(benign_train))
malign_train_label = np.ones(len(malign_train))
benign_test_label = np.zeros(len(benign_test))
malign_test_label = np.ones(len(malign_test))

X_train = np.concatenate((benign_train, malign_train), axis = 0)
Y_train = np.concatenate((benign_train_label, malign_train_label), axis = 0)
X_test = np.concatenate((benign_test, malign_test), axis = 0)
Y_test = np.concatenate((benign_test_label, malign_test_label), axis = 0)

s = np.arange(X_train.shape[0])
np.random.shuffle(s)
X_train = X_train[s]
Y_train = Y_train[s]

s = np.arange(X_test.shape[0])
np.random.shuffle(s)
X_test = X_test[s]
Y_test = Y_test[s]

Y_train = to_categorical(Y_train, num_classes= 2)
Y_test = to_categorical(Y_test, num_classes= 2)
x_train, x_val, y_train, y_val = train_test_split(
    X_train, Y_train,
    test_size=0.2,
    random_state=11)
```

Building A CNN Model Using ResNet-101 for Image Classification

```
def build_model(backbone, lr=1e-4):
    model = Sequential()
    model.add(backbone)
    model.add(layers.GlobalAveragePooling2D())
    model.add(layers.Dropout(0.5))
    model.add(layers.BatchNormalization())
    model.add(layers.Dense(2, activation='softmax'))


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

    return model
K.clear_session()
gc.collect()

result = ResNet101(
    weights='imagenet',
    include_top=False,
    input_shape=(224,224,3)
)

model = build_model(result ,lr = 1e-4)
model.summary()
```

Saving Best Weights During Training

```
from keras.callbacks import ModelCheckpoint
import numpy as np
from sklearn.metrics import accuracy_score


filepath = '/content/drive/MyDrive/best_model.keras'

checkpoint = ModelCheckpoint(filepath, monitor='val_accuracy', verbose=1, save_best_only=True, mode='max')

history = model.fit(
    train_generator.flow(x_train, y_train, batch_size=BATCH_SIZE),
    steps_per_epoch=int(x_train.shape[0] / BATCH_SIZE),
    epochs=5,
    validation_data=(x_val, y_val),
    callbacks=[checkpoint]
)

model.load_weights(filepath)

Y_val_pred = model.predict(x_val)
accuracy_score(np.argmax(y_val, axis=1), np.argmax(Y_val_pred, axis=1))
```

Model Accuracy And Model Loss

```
plt.figure(figsize=(12, 5))
plt.subplot(1, 2, 1)
plt.plot(history.history['accuracy'], label='Train Accuracy')
plt.plot(history.history['val_accuracy'], label='Validation Accuracy')
plt.legend()
plt.title('Accuracy')

plt.subplot(1, 2, 2)
plt.plot(history.history['loss'], label='Train Loss')
plt.plot(history.history['val_loss'], label='Validation Loss')
plt.legend()
plt.title('Loss')
plt.show()
```

Final Validation Accuracy

```
model.load_weights("/content/drive/MyDrive/best_model.keras")

Y_val_pred = model.predict(x_val)
accuracy_score(np.argmax(y_val, axis=1), np.argmax(Y_val_pred, axis=1))
```

```
/usr/local/lib/python3.11/dist-packages/keras/src/saving/saving_lib.py:757: UserWarning: Skipping
  saveable.load_own_variables(weights_store.get(inner_path))
8/8 ━━━━━━━━━━━━━━━━━━━━ 50s 6s/step
0.7086956521739131
```

Building Confusion Matrix

```
Y_pred = model.predict(X_test)
tta_steps = 10
predictions = []

for i in tqdm(range(tta_steps)):

    preds = model.predict(train_generator.flow(X_test, batch_size=BATCH_SIZE, shuffle=False),
                          steps = len(X_test) // BATCH_SIZE)
    predictions.append(preds)
    gc.collect()

Y_pred_tta = np.mean(predictions, axis=0)
from sklearn.metrics import confusion_matrix

def plot_confusion_matrix(cm, classes,
                          normalize=False,
                          title='Confusion matrix',
                          cmap=plt.cm.Blues):

    if normalize:
        cm = cm.astype('float') / cm.sum(axis=1)[:, np.newaxis]
        print("Normalized confusion matrix")
    else:
        print('Confusion matrix, without normalization')

    print(cm)
```

ROC Curve

```
from sklearn.metrics import roc_curve, auc
y_true = np.argmax(Y_test, axis=1)
y_pred = np.argmax(Y_pred, axis=1)

fpr, tpr, _ = roc_curve(y_true, y_pred)
roc_auc = auc(fpr, tpr)
plt.plot(fpr, tpr, label=f'AUC = {roc_auc:.2f}')
plt.xlabel('False Positive Rate')
plt.ylabel('True Positive Rate')
plt.title('ROC Curve')
plt.legend()
plt.show()
```

AUC ROC Curve

```
from sklearn.metrics import roc_curve, auc
import matplotlib.pyplot as plt
fpr, tpr, _ = roc_curve(y_true, y_pred)
roc_auc = auc(fpr, tpr)
plt.figure(figsize=(8,6))
plt.plot(fpr, tpr, color='#FF6F61', lw=2, label=f'AUC = {roc_auc:.2f}')
plt.fill_between(fpr, tpr, alpha=0.3, color='#FF6F61')
plt.plot([0, 1], [0, 1], color='grey', linestyle='--', lw=2)
plt.xlabel('False Positive Rate', fontsize=12, fontweight='bold', color='#333333')
plt.ylabel('True Positive Rate', fontsize=12, fontweight='bold', color='#333333')
plt.title(' AUC-ROC Curve', fontsize=14, fontweight='bold', color='#FF6F61')
plt.legend(loc='lower right', fontsize=12)
plt.grid(alpha=0.3)
plt.show()
```

SAMPLE CODE FOR Inception V3

Import the Libraries

```
import json
import math
import os
import cv2
from PIL import Image
import numpy as np
from keras import layers
import tensorflow
from tensorflow.keras.applications import InceptionV3
from tensorflow.keras.preprocessing.image import ImageDataGenerator
from tensorflow.keras.utils import to_categorical
from keras.models import Sequential
from tensorflow.keras.optimizers import Adam
import matplotlib.pyplot as plt
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.metrics import cohen_kappa_score, accuracy_score
import scipy
from tqdm import tqdm
import tensorflow as tf
from keras import backend as K
import gc
from functools import partial
from sklearn import metrics
from collections import Counter
import itertools
```

Splitting Data Into Training and Testing

```
benign_train = np.array(Dataset_loader('/content/review/BreaKHis 400X/train/benign',224))
malign_train = np.array(Dataset_loader('/content/review/BreaKHis 400X/train/malignant',224))
benign_test = np.array(Dataset_loader('/content/review/BreaKHis 400X/test/benign',224))
malign_test = np.array(Dataset_loader('/content/review/BreaKHis 400X/test/malignant',224))
benign_train_label = np.zeros(len(benign_train))
malign_train_label = np.ones(len(malign_train))
benign_test_label = np.zeros(len(benign_test))
malign_test_label = np.ones(len(malign_test))

X_train = np.concatenate((benign_train, malign_train), axis = 0)
Y_train = np.concatenate((benign_train_label, malign_train_label), axis = 0)
X_test = np.concatenate((benign_test, malign_test), axis = 0)
Y_test = np.concatenate((benign_test_label, malign_test_label), axis = 0)

s = np.arange(X_train.shape[0])
np.random.shuffle(s)
X_train = X_train[s]
Y_train = Y_train[s]

s = np.arange(X_test.shape[0])
np.random.shuffle(s)
X_test = X_test[s]
Y_test = Y_test[s]

Y_train = to_categorical(Y_train, num_classes= 2)
Y_test = to_categorical(Y_test, num_classes= 2)
x_train, x_val, y_train, y_val = train_test_split(
    X_train, Y_train,
    test_size=0.2,
    random_state=11)
```

Building A CNN Model Using Inception V3 for Image Classification

```
def build_model(backbone, lr=1e-4):
    model = Sequential()
    model.add(backbone)
    model.add(layers.GlobalAveragePooling2D())
    model.add(layers.Dropout(0.5))
    model.add(layers.BatchNormalization())
    model.add(layers.Dense(2, activation='softmax'))


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

    return model
K.clear_session()
gc.collect()

res = InceptionV3(
    weights='imagenet',
    include_top=False,
    input_shape=(224,224,3)
)
model = build_model(res ,lr = 1e-4)
model.summary()
```

Saving Best Weights During Training

```
from keras.callbacks import ModelCheckpoint
import numpy as np
from sklearn.metrics import accuracy_score


filepath = '/content/drive/MyDrive/best_model.keras'

checkpoint = ModelCheckpoint(filepath, monitor='val_accuracy', verbose=1, save_best_only=True, mode='max')

history = model.fit(
    train_generator.flow(x_train, y_train, batch_size=BATCH_SIZE),
    steps_per_epoch=int(x_train.shape[0] / BATCH_SIZE),
    epochs=5,
    validation_data=(x_val, y_val),
    callbacks=[checkpoint]
)

model.load_weights(filepath)

Y_val_pred = model.predict(x_val)
accuracy_score(np.argmax(y_val, axis=1), np.argmax(Y_val_pred, axis=1))
```

Model Accuracy and Model Loss

```
plt.figure(figsize=(12, 5))
plt.subplot(1, 2, 1)
plt.plot(history.history['accuracy'], label='Train Accuracy')
plt.plot(history.history['val_accuracy'], label='Validation Accuracy')
plt.legend()
plt.title('Accuracy')

plt.subplot(1, 2, 2)
plt.plot(history.history['loss'], label='Train Loss')
plt.plot(history.history['val_loss'], label='Validation Loss')
plt.legend()
plt.title('Loss')
plt.show()
```

Final Validation Accuracy

```
model.load_weights("/content/drive/MyDrive/best_model.keras")

Y_val_pred = model.predict(x_val)
accuracy_score(np.argmax(y_val, axis=1), np.argmax(Y_val_pred, axis=1))
```

```
/usr/local/lib/python3.11/dist-packages/keras/src/saving/saving_lib.py:757: UserWarning: Skipping
  saveable.load_own_variables(weights_store.get(inner_path))
8/8 ━━━━━━━━━━━━━━━━━━━━ 37s 4s/step
0.8695652173913043
```

Building Confusion Matrix

```
Y_pred = model.predict(X_test)
tta_steps = 10
predictions = []

for i in tqdm(range(tta_steps)):

    preds = model.predict(train_generator.flow(X_test, batch_size=BATCH_SIZE, shuffle=False),
                          steps = len(X_test) // BATCH_SIZE)
    predictions.append(preds)
    gc.collect()

Y_pred_tta = np.mean(predictions, axis=0)
from sklearn.metrics import confusion_matrix

def plot_confusion_matrix(cm, classes,
                          normalize=False,
                          title='Confusion matrix',
                          cmap=plt.cm.Blues):

    if normalize:
        cm = cm.astype('float') / cm.sum(axis=1)[:, np.newaxis]
        print("Normalized confusion matrix")
    else:
        print('Confusion matrix, without normalization')

    print(cm)
```

ROC Curve

```
from sklearn.metrics import roc_curve, auc
y_true = np.argmax(Y_test, axis=1)
y_pred = np.argmax(Y_pred, axis=1)
fpr, tpr, _ = roc_curve(y_true, y_pred)
roc_auc = auc(fpr, tpr)
plt.plot(fpr, tpr, label=f'AUC = {roc_auc:.2f}')
plt.xlabel('False Positive Rate')
plt.ylabel('True Positive Rate')
plt.title('ROC Curve')
plt.legend()
plt.show()
```

AUC ROC Curve

```
from sklearn.metrics import roc_curve, auc
import matplotlib.pyplot as plt
fpr, tpr, _ = roc_curve(y_true, y_pred)
roc_auc = auc(fpr, tpr)
plt.figure(figsize=(8,6))
plt.plot(fpr, tpr, color='#FF6F61', lw=2, label=f'AUC = {roc_auc:.2f}')
plt.fill_between(fpr, tpr, alpha=0.3, color='#FF6F61')
plt.plot([0, 1], [0, 1], color='grey', linestyle='--', lw=2)
plt.xlabel('False Positive Rate', fontsize=12, fontweight='bold', color='#333333')
plt.ylabel('True Positive Rate', fontsize=12, fontweight='bold', color='#333333')
plt.title(' AUC-ROC Curve', fontsize=14, fontweight='bold', color='#FF6F61')
plt.legend(loc='lower right', fontsize=12)
plt.grid(alpha=0.3)
plt.show()
```

REFERENCES

[1] Arshad, W., Masood, T., Mahmood, T., Jaffar, A., Alamri, F. S., Bahaj, S. A. O., & Khan, A. R. (2023). Cancer unveiled: A deep dive into breast tumor detection using cutting-edge deep learning models. *IEEE Access*, *11*, 133804-133824.

[2] Nasser, M., & Yusof, U. K. (2023). Deep learning based methods for breast cancer diagnosis: a systematic review and future direction. *Diagnostics*, *13*(1), 161.

[3] Mahmood, T., Arsalan, M., Owais, M., Lee, M. B., & Park, K. R. (2020). Artificial intelligence-based mitosis detection in breast cancer histopathology images using faster R-CNN and deep CNNs. *Journal of clinical medicine*, *9*(3), 749.

[4] Sohail, A., Khan, A., Wahab, N., Zameer, A., & Khan, S. (2021). A multi-phase deep CNN based mitosis detection framework for breast cancer histopathological images. *Scientific Reports*, *11*(1), 6215.

[5] Yari, Y., Nguyen, T. V., & Nguyen, H. T. (2020). Deep learning applied for histological diagnosis of breast cancer. *IEEE Access*, *8*, 162432-162448.

[6] Fatima, N., Liu, L., Hong, S., & Ahmed, H. (2020). Prediction of breast cancer, comparative review of machine learning techniques, and their analysis. *IEEE Access*, *8*, 150360-150376.

[7] Sharmin, S., Ahammad, T., Talukder, M. A., & Ghose, P. (2023). A hybrid dependable deep feature extraction and ensemble-based machine learning approach for breast cancer detection. *IEEE Access*, *11*, 87694-87708.

[8] Jadoon, E. K., Khan, F. G., Shah, S., Khan, A., & Elaffendi, M. (2023). Deep learning-based multi-modal ensemble classification approach for human breast cancer prognosis. *IEEE Access*, *11*, 85760-85769.

[9] Bappi, J. O., Rony, M. A. T., Islam, M. S., Alshathri, S., & El-Shafai, W. (2024). A novel deep learning approach for accurate cancer type and subtype identification. *IEEE Access*.

[10] Qasrawi, R., Daraghmeh, O., Qdaih, I., Thwib, S., Polo, S. V., Owienah, H., ... & Atari, S. (2024). Hybrid ensemble deep learning model for advancing breast cancer

detection and classification in clinical applications. *Heliyon*, *10*(19).

[11] Alotaibi, M., Aljouie, A., Alluhaidan, N., Qureshi, W., Almatar, H., Alduhayan, R., ... & Almazroa, A. (2023). Breast cancer classification based on convolutional neural network and image fusion approaches using ultrasound images. *Heliyon*, *9*(11).

[12] Thakur, A., Gupta, M., Sinha, D. K., Mishra, K. K., Venkatesan, V. K., & Guluwadi, S. (2024). Transformative breast Cancer diagnosis using CNNs with optimized ReduceLROnPlateau and Early stopping Enhancements. *International Journal of Computational Intelligence Systems*, *17*(1), 14.

[13] S. Cascianelli, R. Bello-Cerezo, F. Bianconi, M. L. Fravolini, M. Belal, B. Palumbo, and J. N. Kather, ``Dimensionality reduction strategies for cnn-based classication of histopathological images," in Proc. Int. Conf. Intell. Interact. Multimedia Syst. Services. Springer, 2018, pp. 2130.[Online].

[14] V. Gupta and A. Bhavsar, ``Sequential modeling of deep features for breast cancer histopathological image classication," in Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. Workshops, Jun. 2018, pp. 22542261.

[15] W. Zhi, H. W. F. Yueng, Z. Chen, S. M. Zandavi, Z. Lu, and Y. Y. Chung, ``Using transfer learning with convolutional neural networks to diagnose breast cancer from histopathological images," in Neural Information Processing_ICONIP (Lecture Notes in Computer Science), vol. 10637, D. Liu, S. Xie, Y. Li, D. Zhao, and E. S. El- Alfy, Eds. Cham, Switzerland: Springer, 2017. [Online]. Available: https://link.springer.com/chapter/10.1007%2F978-3-319-70093-9_71, doi: 10.1007/978-3-319-70093-9_71.

Github Link: https://github.com/JhanviAcharya/Onco-Vision-Deep-Learning-for-Tumor-Cell-Detection-in-Breast-Cancer

conference paper V3 plag check.docx

by Poongodi T

Submission date: 02-Jun-2025 09:52AM (UTC+0530)
Submission ID: 2690261811
File name: conference_paper_V3_plag_check.docx (495.25K)
Word count: 3316
Character count: 19079

Onco Vision: Deep Learning for Tumor Cell Detection in Breast Cancer

Abstract-Profound learning emerged as a powerful tool during the transformation of breast cancer detection at the stage of tumor cell identification in histopathological images. Among deep learning methods, CNN-Densenet-201 has been found to be the most suited for complex image processing applications as it can effectively handle more intricate image information and extract many important features from the dataset. Following a morphological layer, the enormous depth of the architecture combined with a Densenet-based residual learning system can extract the coarse features of progressive properties on very large volumes of annotated tissue tests at the interface. In a later step, based on its power, it separates kinds of cancerous cells and non-cancerous cells with 93% accuracy. The role of pre-processing of images in these systems' utilization enhancement is a significant one. Benign tumors are noncancerous growths, while malignant tumors are those cells that spread to different areas in the body. Normalization maintains pixel values within a range so that uniformity exists throughout the images. Segmentation separates target regions of interest containing tumor cell boundaries from surrounding tissues so that the input information is clearer and tumor margins are even more accurate.

Keywords: Deep Learning, CNN, DenseNet-201, ResNet -101,Inception V3, benign, malignant

I. INTRODUCTION

Even today, breast cancer changes as one of the most prevalent and fatal cancer types that challenge the lives of women all over the globe. Early detection is necessary as it enhances the effectiveness of treatment and survival rates. Histopathological examination remains the most trustworthy means of detecting breast cancer, where tissue samples are scrutinized under a microscope to detect any presence of abnormal cell structures. Nonetheless, this technique has several drawbacks. Tissue sample analysis is time-consuming, heavily relies on the expertise of pathologists, and can greatly vary from one pathologist to the other. With an increase in demand for precise and rapid histopathological image analysis, a new breed of computer-aided diagnostic systems has come into place to help the doctors.

With all the advancements in machine learning and deep learning, nowadays automated image analysis has become a promising tool for medical diagnosis. CNN have successfully pioneered image-classification tasks because they can learn and extract important features from raw image data. The advantage of CNN is that they do not require any manual feature extraction and can thus be applied to complicated problems like cancer detection in medical images. Even though it is highly successful, training a CNN from scratch requires a huge amount of labeled data that in most cases is not easily available in medical imaging. In the evolution of deep learning, the transition toward transfer learning attracted much attention as a solution to this problem. Transfer learning involves working with pre-trained models that have learned generic image features on large-scale datasets and adapting them to target problems such as disease classification. This way, the training time may be reduced and the model's accuracy improved on a more concise, focused dataset. This study adopts transfer learning on the DenseNet-201 architecture, which is a deep CNN known for its dense connectivity between layers. In contrast to traditional CNN where a layer is connected to its consecutive layer, DenseNet-201 connects each layer to all other layers in a feed-forward fashion, thereby promoting feature reuse, better gradient flow, and lower complexity for the model. This model is being utilized to classify breast cancer histopathological images into benign and malignant categories. The dataset used for this study is called the BreakHis dataset at 400x magnification. This dataset comprises a whopping 1,693 high-resolution images in total, with 1,148 for training and 545 for testing. The training set consists of 371 benign and 777 malignant images, while the test set is made up of 176 benign and 369 malignant examples. In this way, the training set can be artificially expanded, allowing the model to see a more extensive variety of images-it essentially helps the model become more generalized and dutiful. The techniques may include anything from somewhat straightforward manipulations like image rotation, flipping, or zooming. The DenseNet-201 based model is customized by adding miscellaneous classification layers, trained with an Adam optimizer and evaluated with categorical crossentropy as its loss function. The model, being evaluated on the test dataset, produced an outstanding accuracy of 93%, indicating the correctness of its predictions. The result shows that densenet201 is efficient in the recognition of very fine, subtle patterns of histopathological images that are difficult to identify manually.

This work highlights the potential of deep learning and transfer learning methods to aid or automate breast cancer diagnosis. The densenet201 with histopathological imaging data is envisioned to build a tool that is fast, accurate, and scalable for supporting clinicians in making decisions in real-world settings.

II. LITERATURE SURVEY

In very recent times, the deep learning methodologies-perhaps CNNs-were probably to generate an automated diagnosis for breast cancer with histopathological image analysis. Numerous researches were carried out against the most popular CNNs to assert their diagnostic possibilities.

The researchers had tested and compared a number of architecture models: inceptionv3, VGG16, ResNet152, MobileNet-V2, DenseNet-121, on histopathological datasets. Prior to putting the model to training, the images went through massive preprocessing techniques, such as normalization and augmentation, to improve their quality and to avoid overfitting of the model. DenseNet121 and VGG16 models proved to have very good classification accuracies of almost 99% [1]. The primary goal of the research was to merge CNN with probabilistic models such as Restricted Boltzmann Machine (RBM), autoencoders (AEs), and contrastive divergence to reduce data complexity. Their approach sought to reconstruct inputs across encoding-decoding units; in doing so, this methodology overcomes the inadequacy of designer features [2]. An analysis of handcrafted features extraction processes based on mitosis datasets was performed by Mahmood and Arsalan mainly in terms of color, texture, and morphology (ICPR 2012/2014). Then, hybrid deep learning techniques were highlighted with a hybrid classification framework in which ResNet-50 and DenseNet-201 were made to contribute to the greatest extent possible [3]. A mitosres-cnn classifier was initiated on the tupac16 dataset to rule out false mitoses. Preprocessing, normalization, and division of the datasets into training, testing, and validation subsets prepared the dataset for analysis, using cross-validation to test the validity of the model [4]. A CNN-based CAD system implemented ResNet50 and DenseNet121 on the breakhis dataset, reaching a high accuracy of 98%. The workflow comprised preprocessing, evaluation of algorithms, and hyperparameter tuning, thus highlighting the importance of systematic experimentation [5].

In the past, models for traditional machine learning were analyzed, such as Support Vector Machine (SVM), CNN and random forest applied on datasets like the Wisconsin breast cancer dataset (WDBC) and the Mias dataset. The experiment saw Weka standing with the highest accuracy (99%), while CNN registered 97%, supporting CNN as being in between accuracy and flexibility [6]. The ResNet-50 V2 model and Light GBM algorithm were utilized to classify invasive ductal carcinoma (IDC). The preprocessing steps included resizing, conversion of color, filtering, and normalization, ending up with a highly potent ensemble model [7]. The authors introduced CNN, DNN, and Random Forest algorithms for multi-modal analyses on the metabric dataset. The mrmr and CNN-based feature extraction proved to provide stellar classification scores, especially with respect to gene expression data [8]. It combined CNN and LSTM architectures to classify cancer subtypes with the help of a dataset found in Kaggle. Enhancement techniques, including Gaussian blur and Fourier transforms, were implemented before the extraction of spatial and temporal features. The features were then combined for classification [9]. This research paper proposes a hybrid framework consisting of Gaussian blur area, YOLO V5, and a deep ensemble RVFL network for mammography images. Densenet121 was used in the feature extraction stage, while the classifiers, namely support vector machine, random forest, and logistic regression, were tested, giving considerable results [10]. From the literature review, it's observed that convolutional networks, especially VGG-19 and ensembles, have proven to be very accurate in classifying breast cancer from ultrasound images using transfer learning, spatial attention methods. Yet noise in image constitution, inter-dataset inconsistency, and poor fuzziness-nature restriction to real-world local grounds challenge them [11].In this article, a CNN-based model has been suggested to improve the early stopping mechanism and learning rate reduction on plateau for the accurate classifying breast cancer using mammography images. It achieved 95.2% accuracy, which is good because it means this model can generalize well over different data sets, hence allowing it to be presented as a reliable and explainable tool for early cancer detection [12].

DenseNet-201 deep architecture and dense connectivity make it a good candidate for reuses of features and gradient flow that may be well-suited for high-resolution images, such as those at BreaKHis 400x magnification. However, many papers either do not explore its full potential at this resolution or mass-apply methodologies at all magnifications. Such an abundance of cellular detail in the images demands tailored strategies for preprocessing and training. Also, poor integration of the clinical data and explainable AI constraints limit usefulness. Thereby, using DenseNet201 along with clinical data and explainability could significantly enhance diagnostic accuracy and applicability.

III. PROPOSED METHODOLOGY

The CNN with Densenet-201 for breast cancer diagnosis.

A truly comprehensive guide to get an unblemished grasp of convolutional neural networks. Convolutional Neural Networks are sophisticated machine learning techniques specially built for handling visual information. They analyze images with the help of filters that detect fundamental attributes such as edges, texture, and spatial relationships between elements. These features lead the model to discern irregular patterns, hence very important in medical imaging for detecting cancerous cells in breast tissue.

The main components of CNN are:

convolutional layers: they somehow add depth to the mode and detect local patterns like edges of cells or presence and absence of pattern structures

pooling layers: they reduce the data dimension so that it can dispose of information that is irrelevant yet important details are preserved Pooling also accelerates computation.

activation function: introduces nonlinear properties to our network and permits it to learn more complex functions

Recognizing cancerous and non-cancerous in breast tissue on digital microscopy images is mainly reliant on spotting very subtle differences. Perhaps Densenet-201 is deemed top-performing simply for these reasons:

Dense connectivity: By having direct connections from each layer to all other layers in a feed-forward manner, densenet-201 reuses the feature-maps throughout the network. This guarantees the presence of image features that may include anything from the finest high-level semantic cues to low-level visual cues, which behaves as a necessary step for understanding intricate tissue structures.
Feature propagation: As features propagate through the network, this allows each layer to utilize information coming from the gradients of the loss function and act as an efficient method of information exchange.
Enhanced performance: A dense network counteracts the vanishing gradient problem and also allows the training of much deeper models to go afoul of their power to do so. This then helps in more accurately identifying non-cancerous, benign, and malignant tissue.

Concept	Dense Net 201	Inception V3	ResNet 101
Feature Sharing	shares across layers	multi-filter views	learns residuals
Growth	slow, continuous	parallel paths	deep stacking
Efficiency	Very high	Medium	Less
Accuracy	93%	86%	70%

Therefore, due to the nature of its architecture, densenet-201 can conveniently process and analyze histopathological scans while allowing for inter-patient variation. Implementation of densenet-201 for breast cancer detection: images produced are histology images.
DenseNet tightly knit structure facilitates the efficient flow of information, combats the vanishing gradient problem, and encourages reuse of features. Hence, this advantage in architecture makes the model able to learn more specific and informative representations and helps make DenseNet 201 a better and more reliable choice for the classification task at hand.

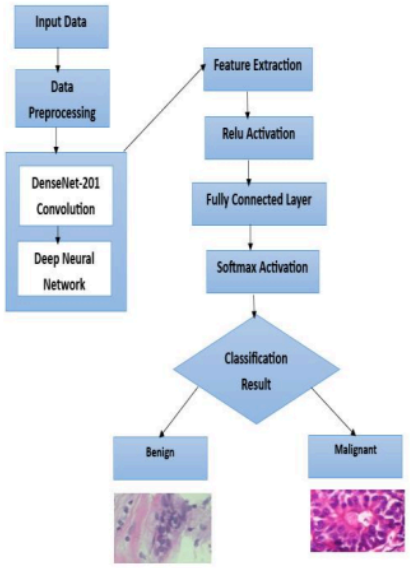

Fig 3.1 Proposed Design

It shows an illustration of a proposed deep learning model designed by utilizing densenet-201 for the classification of histopathological images. It will analyze the input data to check if the sample is benign or malignant.

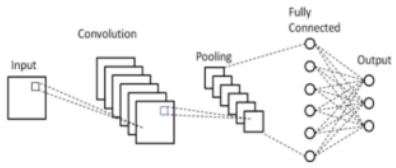

Fig 3.2 CNN Architecture

It shows the architecture of the convolutional neural network for image classification. The input image undergoes convolution and pooling operations to grasp and extract key spatial features. These features are passed on to the fully connected layers for the final prediction or classifying result.

In this layer, the convolutional neural network (cnn) receives raw data as input (in this case, images of breast cancer) and passes it to the subsequent layer for preprocessing.

Convolutional layers: to find the basic features like cell edges, shapes, and textures of the dataset, the convolution layer takes tiny filters or kernels to apply to the image. Each filter generates a new feature map as it travels all over the image.
Formula:

$$\text{Feature map} = \sum(i * k) + b \quad (1)$$

Where:

- i: input image.
- k: kernel (filter).
- b: bias term.

It generates the feature map as their output, and the size of this feature map is determined by the stride and kernel. It also reduces the size of an image somewhat without losing critical information, such as the boundaries of an image, patterns, or cellular structure. In general, breast cancer images are large and very detailed. So, it is a good option to connect such large images to a pooling layer to reduce their size; thus, this makes training and testing faster and more effective.
Average pooling: calculates the average value of a region.
Activation layers: An activation function is a function used to allow or disallow a neuron to fire. It takes the output from the preceding layer and applies a certain rule to make it useful to the next layer. By adding nonlinearity to the input, the network can then be trained to understand more complex patterns in the training dataset.
ReLU: ReLU stands for rectified linear unit, which is a form of activation and probably the most well-known one used within several different architectures for neural nets. It basically keeps any positive values as they are and converts the negative values to zero. This process is useful because the model learns and trains quicker by only focusing on positive features.
Formula:

$$f(x) = \max(0,x) \quad (2)$$

Where:

- f(x): It implies function of a variable x.
- max(0, x): This expression gives the maximum value between 0 and x.

Fully Connected (Dense) Layer: For decision-making, all features have been connected through this layer after all feature maps were flattened into 1D vectors. Each neuron on said input layer has a connection to every other neuron. The dense layer, integrating all patterns acquired at earlier levels, produces the final output.
Softmax Layer: The softmax layer is the core one for classification issues. It may be construed as converting raw output scores to probabilities that total exactly one, while simultaneously deciding on the malignant or non-malignant class prediction.
Formula:

$$P(y_i) = \sum e\text{^}zi \;/\; e\text{^}zj \quad (3)$$

Where:

- Zi is the logit value for class i.
- E denotes the exponential function.
- ∑e^zj represents the sum of the exponential scores for all classes.

Adam Optimizer: The learning rate for each parameter according to estimates of the first and second moments of the parameter gradient, finishing the process of faster convergence with better performance.
Formula:

$$\theta_{t+1} = \theta_t - \alpha \cdot \frac{\hat{m}t}{\sqrt{\hat{v}t} + \epsilon} \quad (4)$$

Where:

- θ_t: Model parameters at time step t
- α: Learning rate
- $\hat{m}t$: Bias-corrected first moment estimate
- $\hat{v}t$: Bias-corrected second moment estimate
- ϵ:Small constant to prevent division by zero

1. Algorithm:
Importing modules: Libraries such as tensorflow, keras, numpy, and matplotlib are loaded in preparation for the work to be done herein. Densenet201 is used as the basis of the classification system from keras' pre-trained models.
Reading and preparing the data: Images are loaded from the benign and malignant folders, resized to a standard width and height of 224, respective of the pixel size, and standardized for analysis. The dataset is split into training, validation, and test sets, with label encoding applied to the target and normalization to the data.
Train Deep Neural Network with PyTorch: Densenet201 is initialized with `include_top=false` to remove the default classifier layers. Then a custom classification head is attached that uses global average pooling and dense layers with a sigmoid activation for binary outputs.
Compile the model and define the data augmentation: Compilation of the model occurs with adam as the optimizer and binary_crossentropy as the loss function. Data augmentation is carried out for the images by randomly flipping, rotating, and zooming, in an attempt to make the data less prone to overfitting.
Model fitting: The fitting of the model is accomplished on the augmented images over several iterations, or epochs. Validation accuracy is frequently monitored and the best version of the model preserved through callbacks that implement early stopping and model checkpointing.
Analysis of Results: After the training phase, the model is interrogated on the validation set. Various parameters are reported (loss and accuracy), and sample predictions are

visualized to qualitatively check the level of trustworthiness in classification.

Assessment metrics: Accuracy, precision, recall, and f1-score evaluate the quantitative performance. The evaluation results provide the model's validation accuracy, approximately 93% of classification legitimacy for medical images.

IV. EXPERIMENTAL SETUP

Simulation:

Platform	Specification
Google Colab	
Python Version	3.11
TensorFlow Version	2.x

Data and Preprocessing:

Item	Specification
Dataset	BreaKHis 400x
Image Size	224x224
Batch Size	16
RGB Channels	Yes
Epochs	5
Train/Val Split	80%/20%

Python Libraries:

Algorithm	Package Name
Data Handling	Pandas, NumPy
Data Visualization	Matplotlib
Image Processing	OpenCV, Pillow, TensorFlow Keras
Model Training	TensorFlow, Keras, DenseNet201
Performance Metrics	Sklearn

DATASET

A set of 1,693 histopathological images at 400x magnification was utilized. It consists of 1,148 training images, with an almost 1:2 ratio of benign (371 images) and malignant (777 images). There are also 545 testing images, with 176 being benign and 369 malignant. The images exhibit detailed cellular attributes, which are extremely valuable for the correct classification of breast cancer. Their very high resolution enables the training of deep learning models efficiently.

(a) (b)

Fig. 5: Histopathological in the dataset:
(a) Benign (b) Malignant

V. RESULTS ANALYSIS AND DISCUSSION

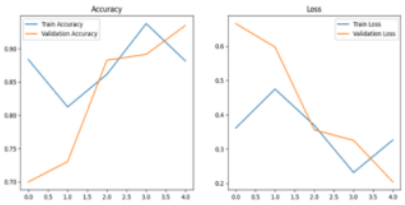

Fig 5.1 Accuracy and Loss in DenseNet-201

The plot in fig demonstrates the performance of the model through five iterations. In the left plot, we see an increase in training and validation accuracy, reaching roughly a peak in validation accuracy of 94%. Training and validation loss display a decreasing fashion in the right plot, indicating effective learning and slight to no overfitting.

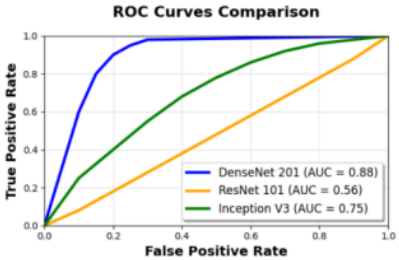

Fig 5.2 ROC Curve Comparison

The figure depicts the comparison between the receiver operating characteristic curves for DenseNet 201, Resnet 101, and inception v3. Out of all classifiers, DenseNet 201 gave the most excellent results, with an AUC of 0.88, followed by inception v3 (AUC = 0.75), while Resnet 101 showed almost no improvement (AUC = 0.56).

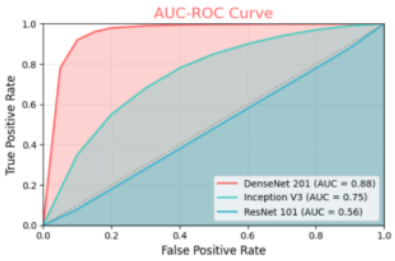

Fig 5.3 AUC-ROC Curve Comparison

The figure shows DenseNet 201, Inception V3, and Resnet 101 AUC-ROC curves. DenseNet 201, with an AUC of 0.88, outperforms the others and is thus a better discriminator. Inception V3 and ResNet 101, on the other hand, have AUC values of 0.75 and 0.56, respectively.

VI. CONCLUSION AND FUTURE WORK

The performance of DenseNet201, Inceptionv3, and ResNet101 in breast cancer classification were examined with transfer training. Densenet201 had the best performance since it attained training accuracy of ~93%, validation accuracy of ~91%, and AUC of 0.88, which establishes it as the most reliable and potent model. Inceptionv3 performs well (validation accuracy of ~86%, AUC 0.75) with low training loss, which proves it to be a good sim for lightweight applications. Worse performance was shown by ResNet101 with a validation accuracy of approximately 70%, while the AUC was 0.56, and therefore the model suffered from overfitting and could not generalize well. Further work will be directed toward enlarging the dataset and employing explainable AI methodologies to improve interpretability for real-world deployment.

During the forthcoming days, a bigger and more diversified dataset shall be harnessed to enhance the learning abilities of the model while avoiding overfitting. In addition to that, explainable AI methods shall be employed to reduce the size of the model so that it can be essentially deployed in real-time applications in the healthcare sector or even as a mobile application.

Commented [SN1]: Include good resolution graph

conference paper V3 plag check.docx

ORIGINALITY REPORT

2%	0%	1%	0%
SIMILARITY INDEX	INTERNET SOURCES	PUBLICATIONS	STUDENT PAPERS

PRIMARY SOURCES

1	Poonam Nandal, Mamta Dahiya, Meeta Singh, Arvind Dagur, Brijesh Kumar. "Progressive Computational Intelligence, Information Technology and Networking", CRC Press, 2025 Publication	1%
2	Agbotiname Lucky Imoize, Oleksandr Kuznetsov, Oleksandr Lemeshko, Oleksandra Yeremenko. "Advancements in Cybersecurity - Next-Generation Systems and Applications", CRC Press, 2025 Publication	<1%
3	www.ncbi.nlm.nih.gov Internet Source	<1%
4	Vidal, Alexander Robert. "Deep Learning Methods for Large-Scale Physics", Colorado School of Mines Publication	<1%
5	Yasha Hasija, Rajkumar Chakraborty. "Hands-On Data Science for Biologists Using Python", CRC Press, 2021 Publication	<1%

Exclude quotes	Off	Exclude matches	Off
Exclude bibliography	Off		