Thesis submitted to the

UNIVERSITY OF MOHAMED BOUDIAF – MSILA

FACULTY OF MATHEMATICS AND COMPUTER SCIENCE

DEPARTMENT OF COMPUTER SCIENCE

In partial fulfillment of the requirements for the degree of

Master in **Computer science**

By

Hamdaoui Radja

Title of the thesis

Early Detection Of Breast Cancer Using AI Technologies

Under the supervision of

Pr. Said Gadri

Composition of the jury

Saleh Guesmia	University of Msila	President
Said Gadri	University of Msila	Reporter
Abdelbaki Bouguerra	University of Msila	Examiner

May, 2026

Abstract

Breast cancer is the most frequently diagnosed cancer among women worldwide. Traditional diagnostic methods remain limited by inter-observer variability and reduced sensitivity, motivating the development of AI-based solutions. This thesis proposes an automated early detection system based on a custom deep learning architecture. The proposed CNN Model has been trained on the IDC histopathological dataset using Focal Loss and cancer-boosted class weighting. After experimentation, it achieved strong classification performance, outperforming all evaluated architectures. The system is integrated into a graphical user interface designed to support clinical decision-making.

Keywords: Breast Cancer, Early Detection, Artificial Intelligence, Deep Learning, Convolutional Neural Network, Histopathological Image Classification.

ملخص

يُعدّ سرطان الثدي أكثر أنواع السرطان تشخيصًا بين النساء في جميع أنحاء العالم. لا تزال طرق التشخيص التقليدية محدودة بسبب التباين بين الملاحظين وانخفاض الحساسية، مما يدفع إلى تطوير حلول قائمة على الذكاء الاصطناعي. تقترح هذه الرسالة نظامًا آليًا للكشف المبكر يعتمد على معمارية تعلم عميق مخصصة. تم تدريب نموذج CNN المقترح على مجموعة بيانات IDC للصور النسيجية المرضية باستخدام دالة الخسارة البؤرية (Focal Loss)وآلية ترجيح الفئات المعززة لفئة السرطان. وبعد إجراء التجارب، حقق النموذج أداءً قويًا في التصنيف، متفوقًا على جميع المعماريات التي تم تقييمها. كما تم دمج النظام في واجهة مستخدم رسومية مصممة لدعم اتخاذ القرار السريري.

الكلمات المفتاحية: سرطان الثدي، الكشف المبكر، الذكاء الاصطناعي، التعلم العميق، الشبكة العصبية التلافيفية، تصنيف الصور النسيجية.

Résumé

Le cancer du sein est le cancer le plus fréquemment diagnostiqué chez les femmes dans le monde. Les méthodes diagnostiques traditionnelles demeurent limitées par la variabilité inter-observateurs et une sensibilité réduite, ce qui motive le développement de solutions basées sur l'intelligence artificielle. Ce mémoire propose un système automatisé de détection précoce fondé sur une architecture d'apprentissage profond personnalisée. Le modèle CNN proposé a été entraîné sur le jeu de données histopathologiques IDC en utilisant la fonction de perte Focal Loss ainsi qu'une pondération des classes renforcée pour la classe cancéreuse. Après expérimentation, il a obtenu de solides performances de classification, surpassant toutes les architectures évaluées. Le système est intégré dans une interface utilisateur graphique conçue pour soutenir la prise de décision clinique.

Mots-clés : Cancer du sein, Détection précoce, Intelligence artificielle, Apprentissage profond, Réseau de neurones convolutif, Classification d'images histopathologiques.

الإهداء

بِسْمِ اللَّهِ الرَّحْمٰنِ الرَّحِيمِ

﴿وَآخِرُ دَعْوَاهُمْ أَنِ الْحَمْدُ لِلَّهِ رَبِّ الْعَالَمِينَ﴾

الحمد لله الذي ما تمَّ جهدٌ، ولا اكتمل سعيٌ، ولا تحقَّق نجاحٌ إلا بفضله وتوفيقه. الحمد لله على ألطافه الخفية ونِعَمه الجليلة التي أحاطت بي طوال مسيرتي الدراسية، وعلى ما منحني من قوةٍ وصبرٍ وعزيمةٍ لتجاوز الصعاب والوصول إلى هذه المرحلة.

إلى أرواح المليون ونصف المليون شهيد.

إلى جميع مجاهدي الجزائر الأبطال الذين ضحّوا بالغالي والنفيس من أجل حرية الوطن واستقلاله، وبفضل تضحياتهم ننعم اليوم بالأمن والأمان والكرامة،

إلى عمتي المجاهدة لويزة، التي سطّرت بصدقها وتضحياتها صفحةً مشرقةً من صفحات الوفاء للوطن.

إلى أرواح شهداء طوفان الأقصى ومجاهديه ضحايا العدو الصهيوني، حراسِ المسجد الأقصى ومرابطيه، الذين سطّروا بدمائهم معاني الصمود والثبات وارتقوا دفاعا عن مقدساتنا.

إلى أمي وأبي، من كانا بعد الله سببًا في كل ما وصلت إليه ، من غرسا في قلبي حبَّ العلم والبذل والعطاء، أسأل الله أن يجعل هذا العمل في ميزان حسناتكما،
وأن يرفعكما به درجاتٍ في الجنة كما كنتما ترفعان همّتي دومًا جزاكما الله عني خير الجزاء، وبارك في عمركما وأدام عليَّ نعمة وجودكما.

إلى أختيَّ الحبيبتين جهينة وفطوم، وإخوتي الأحباء شعيب وإسحاق ومحمد ونضال، وصغير البيت وبهجته أحمد ياسين، الذين آمنوا بي، وساندوا خطواتي، وأمدّوني بالقوة والتوجيه والدعاء طوال هذه المسيرة وإلى يافا وشام، زهرتَي العائلة أسأل الله أن يحفظهما، ويجعلهما من أهل النجاح.

إلى العزيزة وصال، صديقتي في أصعب اللحظات قبل أسهلها، التي لم تبخل عليَّ يومًا بوقتها أو جهدها أو نصحها وإلى الصديقة دُرّة وكل بنات فوج راشدة الكشفي.

أهدي هذا العمل المتواضع لكل من كان له فضلٌ عليّ في هذه المسيرة، دعمًا أو دعاءً أو توجيهًا.

حمداوي رجاء

Acknowledgments

All praise and gratitude are due to Allah, the Almighty, for granting me the strength, patience, and perseverance to complete this work.

I extend my deepest expressions of gratitude and appreciation to my supervisor, Dr. Said Gadri, who spared no effort in providing guidance, support, and supervision throughout this research. His distinguished academic supervision, extensive expertise, sound advice, and precise feedback played a pivotal and essential role in completing this work and presenting it in its best possible form. Moreover, his continuous follow-up, great dedication to guidance, and ongoing support constituted a fundamental pillar in improving the quality of this research and bringing it to its final stage. His constant encouragement had a profound impact on the success of this work.

I also extend my appreciation to the members of the jury for their time, valuable remarks, and thoughtful evaluation of this dissertation.

My heartfelt thanks go to my family for their unwavering support, patience, and encouragement throughout my academic journey.

Finally, I would like to thank everyone who contributed, directly or indirectly, to the completion of this work.

May Allah reward you all with the best reward on my behalf, and bless your knowledge and your efforts.

Contents

List of figures

List of tables

General Introduction

Breast cancer is the most commonly diagnosed cancer among women worldwide. According to recent epidemiological data, over 2.3 million new cases were recorded in 2022, accounting for 23.8% of all new female cancer cases globally [1]. In Algeria, the situation is particularly concerning: breast cancer represents 41.3% of all new female cancer cases, with an age-standardized incidence rate of 61.9 per 100,000 women, one of the highest rates recorded across Africa [1]. These figures highlight the urgent need for improved early detection and management strategies.

Early detection is widely recognized as the most decisive factor in improving patient survival rates and reducing the severity and cost of treatment. Traditional diagnostic methods, including clinical breast examination, mammography, ultrasound, MRI, and biopsy, have long served as the standard of care. However, these approaches present significant limitations: they depend heavily on specialist expertise and availability, are subject to considerable inter-observer variability, and often exhibit elevated false-positive rates or reduced sensitivity, particularly in women with dense breast tissue. These shortcomings motivate the search for more objective, automated, and scalable diagnostic solutions.

Over the past decade, artificial intelligence technologies have emerged as a powerful and transformative paradigm in medical image analysis. Machine learning techniques, including classical algorithms such as Support Vector Machines, Random Forests, and K-Nearest Neighbors, have demonstrated strong potential in extracting meaningful patterns from clinical data and supporting diagnostic decision-making. More recently, deep learning approaches, and Convolutional Neural Networks in particular, have achieved state-of-the-art performance in the automated analysis of histopathological images, enabling complex feature extraction directly from raw image data without the need for hand-engineered descriptors. These AI-driven technologies offer the prospect of earlier, more consistent, and more accessible breast cancer diagnosis. It is within this context that the present thesis is situated, aiming to contribute to the advancement of AI-based breast cancer screening tools.

The main objective of this thesis is to design, develop, and evaluate an automated system for the early detection of breast cancer using AI technologies, with a particular focus on deep learning applied to histopathological images. The proposed system is

built around a custom Convolutional Neural Network trained to classify tissue patches as malignant or benign, with special emphasis on minimizing false negative predictions, which represent the most clinically dangerous type of misclassification in cancer screening. The system is further embedded within a graphical user interface designed to make it accessible and practical for use by medical professionals. This manuscript is organized into three chapters. The first chapter provides a comprehensive overview of breast cancer, covering its definition, global and Algerian epidemiology, traditional diagnostic methods, and the state of the art in AI-based detection, spanning both machine learning and deep learning techniques. The second chapter presents the complete methodology of the proposed system, including dataset preparation, the experimental evaluation of deep learning architectures considered in this work, the model selection process, the training strategy of the final model, and the corresponding performance analysis. The third chapter describes the technical implementation of the system, detailing the development environment, programming libraries, and the design of the graphical user interface.

Chapter 1: Overview of Breast Cancer

1.1 Introduction

In this chapter, we present the main concepts of breast cancer, starting with its definition and general statistics worldwide and in Algeria. We also introduce the traditional diagnostic methods used for breast cancer detection, including clinical breast examination, mammography, ultrasound, MRI, and breast biopsy. In addition, we briefly present the role of artificial intelligence techniques such as machine learning and deep learning in improving breast cancer diagnosis, with a focus on convolutional neural networks (CNNs).

Finally, we highlight the main challenges in breast cancer detection and the importance of improving diagnostic accuracy.

1.2 Definition

Breast cancer is a disease in which abnormal cells in the breast grow out of control and form tumours. If left unchecked, the tumours can spread throughout the body and become fatal [2].

Breast cancer cells begin inside the milk ducts. The earliest form (in situ) is not life-threatening and can be detected in early stages. Cancer cells can spread into nearby breast tissue (invasion). This creates tumours that cause lumps or thickening [2].

Invasive cancers can spread to nearby lymph nodes or other organs. Metastasis can be life-threatening and fatal. Treatment is based on the person, the type of cancer and its spread. Treatment combines surgery, radiation therapy and medications [2].

1.3 Some Statistics About Breast Cancer Around the World and Algeria

1.3.1 Global Statistics

Breast cancer ranks first among females worldwide, accounting for 23.8% of all new female cancer cases, with 2,296,840 new cases recorded in 2022. It caused 666,103 deaths globally, ranking 4th in cancer mortality [1].

1.3.2 Algeria

In Algeria, breast cancer is the leading cancer among females, representing 41.3% of all new female cancer cases, with 14,601 new cases and 4,893 deaths recorded in 2022. The age-standardized incidence rate (ASR) stands at 61.9 per 100,000 women, one of the highest rates in Africa. These figures are based on regional cancer registries (Sétif, Algiers, Batna, and Tlemcen), given the absence of a unified national registry [1].

1.4 Diagnostic methods

1.4.1 Traditional Diagnostic Methods

Traditional diagnostic methods are used to detect and clinically evaluate breast cancer. They are widely used in hospitals and medical centers to identify any abnormalities in breast tissue, determine the presence of tumors, and support treatment decisions. Early detection using traditional diagnostic techniques significantly increases the chances of successful treatment and improves patient survival rates. Common traditional methods include clinical breast examination, mammography, breast ultrasound, magnetic resonance imaging (MRI), and breast biopsy. Each technique provides specific information that helps clinicians assess breast changes and confirm a diagnosis of breast cancer.

1.4.1.1 Clinical Breast Exam

A clinical breast exam is a physical exam of the breast done by a health care provider. He or she will carefully feel the breasts and under the arms for lumps or anything else that seems unusual. A clinical breast exam alone is not an adequate screening test for breast cancer [3].

1.4.1.2 Mammogram

Mammograms are x-ray images of the breasts. Mammograms are used for breast cancer screening because they can find tumors at an earlier stage, before they cause symptoms.

Mammograms can also be used to check for breast cancer after a woman or her doctor finds a lump or other change.

Mammograms can show a mass (breast lump), deposits of calcium (called calcifications), and other changes. They also show breast density.

A radiologist will study the mammogram for unusual changes. When possible, the radiologist will compare your most recent mammogram with past mammograms to check for changes in breast tissue since your last mammogram [3].

1.4.1.3 Breast Ultrasound

Breast ultrasound uses sound waves and their echoes to make a computer picture of the inside of the breast. It can show certain breast changes, like fluid-filled cysts, that can be harder to see on mammograms.

Ultrasound is not typically used as a routine screening test for breast cancer. But it can be useful for looking at some breast changes, such as lumps (especially those that can be felt but not seen on a mammogram). Ultrasound can be especially helpful in women with dense breast tissue, which can make it hard to see abnormal areas on mammograms. It also can be used to get a better look at a suspicious area that was seen on a mammogram [4].

1.4.1.4 Breast Magnetic Resonance Imaging (MRI)

Breast MRI uses radio waves, atrong magnet, and a computer to create detailed pictures of the inside of the breasts. It does not involve radiation. If you are at high risk for developing breast cancer you may have breast MRI along with mammography. That is because MRI is more sensitive at detecting cancer than mammography and mammography alone may miss some cancers in high-risk women.

If you have dense breasts you may be offered breast MRI along with mammography [3].

1.4.1.5 Breast Biopsy

A breast biopsy is a procedure to remove a sample of breast tissue for testing. The tissue sample is sent to a lab, where doctors who specialize in analyzing blood and body tissue (pathologists) examine the tissue sample and provide a diagnosis.

A breast biopsy might be recommended if you have a suspicious area in your breast, such as a breast lump or other signs and symptoms of breast cancer. It can also be used to investigate unusual findings on a mammogram, ultrasound or other breast exam.

The results of a breast biopsy can show whether the area in question is breast cancer or if it's not cancerous. The pathology report from the breast biopsy can help your doctor determine whether you need additional surgery or other treatment [5].

1.4.2 Diagnosis Using Artificial Intelligence Tools

Over the past few decades, ML techniques have been widely used in intelligent healthcare systems, especially for breast cancer (BC) diagnosis and prognosis. Traditionally the diagnostic accuracy of a patient depends on a physician's experience, however, this expertise is built up over many years of observations of different patients' symptoms and confirmed diagnoses. Even then the accuracy still cannot be guaranteed. With the advent of computing technologies, it is now relatively easy to acquire and store a lot of data. The intelligent healthcare system is therefore a valuable and important domain.

The intelligent healthcare system can assist physicians to diagnose patients with greater accuracy or provide more meaningful benchmarks,

Lately, ML techniques are playing a significant role in diagnosis and prognosis of BC by applying classification techniques to identify people with BC, distinguish benign from malignant tumours and to predict prognosis [6].

1.4.2.1 Breast Cancer Based Machine learning (ML)

ML is a form of Artificial Intelligence (AI) that uses a variety of statistical, probabilistic and optimization tools to learn and improve performance automatically from new data and past experiences, without explicitly programmed instructions.

In dealing with complex, large, high dimensional data, ML approaches are typically capable of extracting key features and potential rules which might be difficult to be discovered using traditional statistics.

ML can provide early diagnosis of these diseases by leveraging the power of artificial intelligence. There are various clinical ways to diagnose BC. Therefore, using ML powerful algorithms, this technology can recognize and classify images of new patients with high accuracy if presented based on examples of existing images from previous patients. Since the photos of new patients are in the early stages of their disease, it can be very important because determining which type of BC the patient is likely to be involved with can, in turn, guide the testing process and lead to a suitable

path, which can lead to a faster diagnosis of the disease. As a result, the process of treatment or preventing the development of the disease can be done more quickly [7].

Among the most commonly used algorithms in breast cancer classification tasks are [7]:

- Support Vector Machine (SVM)
- Naive Bayes (NB)
- Random Forest (RF)
- Decision Tree (DT)
- K-Nearest Neighbor (KNN)
- Logistic Regression (LR)
- Multilayer Perceptron (MLP)
- Linear Discriminant Analysis (LDA)
- XgBoost (XGB)
- Ada-Boost (ABC)
- Gradient Boosting (GBC)

Machine learning-based breast cancer diagnosis generally follows a sequence of steps including data preprocessing, feature extraction, data balancing, classification, and performance evaluation, as illustrated in Figure 1.1.

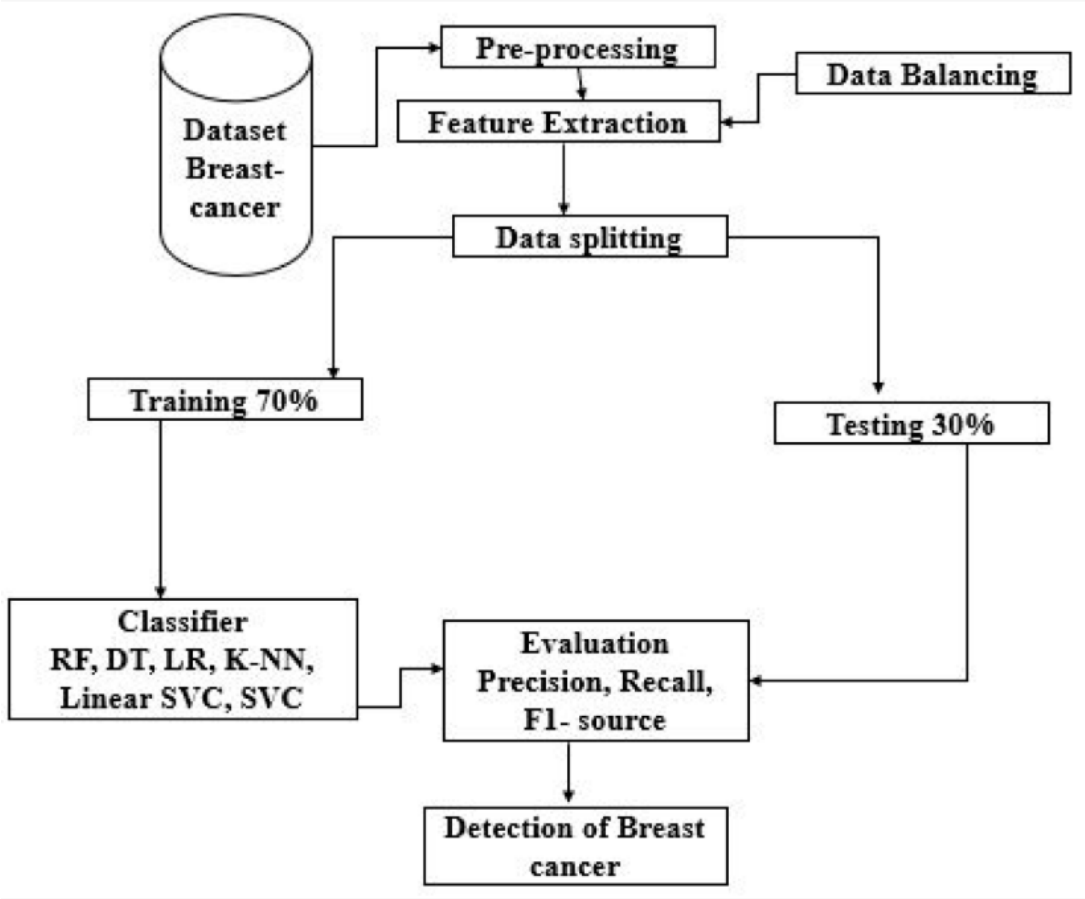

Fig. 1.1: General workflow of machine learning-based breast cancer diagnosis and classification [8]

As shown in Figure 1.1, the general workflow of ML-based breast cancer diagnosis includes data preprocessing, feature extraction, normalization, training/testing split, model evaluation using algorithms such as RF, DT, KNN, LR and SVM, and finally selection of the best performing classifier [8].

1.4.2.2 Breast Cancer Based Deep learning (DL)

Deep neural networks usually have several hidden layers in between input layer and output layer. These networks are used to retrieve features from images, unlike traditional ML algorithms which use hand-engineering features for breast cancer detection. Deep Learning is a subfield of machine learning based on artificial neural networks with multiple hidden layers structures mostly requiring a training stage to find optimum weights. The most frequently used learning rule is the backpropagation algorithm in which weights are updated systematically for every pass depending upon the error rate obtained from the production layer with the gradient and chain rule [9].

Deep learning methods have been demonstrated to be capable of diagnosing breast cancer up to 12 months earlier than those using conventional clinical procedures. In addition, the techniques can be used to learn the most pertinent features to best tackle the issue. In recent times, different deep learning-based methods have been introduced for breast cancer diagnosis, which include CNN-, DNN-, RNN-, DBN- and AE-based approaches.

CNN is the most popular deep-learning technique that has been utilized in several studies for breast cancer detection. The CNN is a deep learning model which represents hierarchical abstraction and consists of various layers which accept features as raw data [10].

Convolutional Neural Networks (CNNs) are widely used in breast cancer detection, particularly for histopathological image classification, due to their ability to automatically extract complex image features.

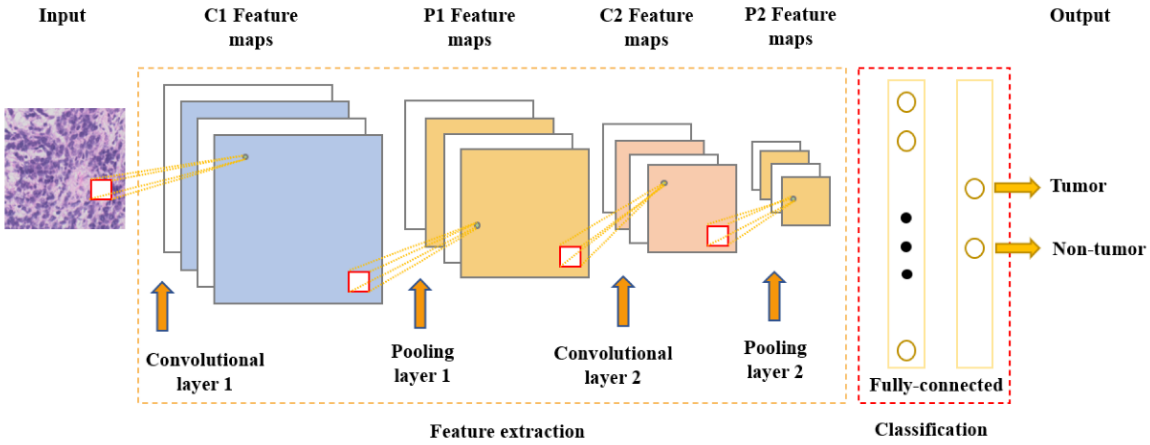

Fig. 1.2: Typical CNN architecture for automatic detection of IDC breast cancer [11]

As shown in Figure 1.2, the CNN architecture consists of convolution and pooling layers for automatic feature extraction, followed by fully connected layers and a softmax classifier for breast cancer classification.

1.5 Challenges of Breast Cancer Detection and the Role of Artificial Intelligence

1.5.1 Challenges of Breast Cancer Detection

- Traditional screening methods (mammography and clinical breast examinations) have several limitations, including high false-positive rates and reduced sensitivity in women with dense breast tissue [12].
- Classifying pathological tissue slides for cancer requires a larger number of pathologists and specialist physicians, and manually examining whole slide images is time-consuming [13].
- Traditional mammography screening may fail to identify interval breast cancers, which are fast-growing and biologically aggressive tumors that appear between scheduled screening intervals, thereby limiting the effectiveness of early detection strategies [14].
- As with any manual diagnostic process, it is subject to substantial intra- and inter-reader variability, adding to the risk of misdiagnosis or overdiagnosis [15].

1.5.2 The Role of Artificial Intelligence

The clinical fundamentals frame the role of AI and deep learning in augmenting detection, characterization, and treatment planning.

Recent advances in AI, particularly in deep learning models, have introduced transformative potential for breast cancer diagnosis. These models excel in processing complex imaging data, enhancing diagnostic precision across modalities such as mammography, ultrasound, and MRI. AI-driven techniques are effective not only for image analysis but also for risk assessment, tumor characterization, and treatment monitoring. For example, AI-based mammography has demonstrated superior diagnostic accuracy in dense breast tissue compared with traditional radiological assessments. AI models trained on extensive datasets also exhibit reduced false-positive and false-negative rates, thereby minimizing diagnostic inconsistencies and missed diagnoses.

In addition to imaging, AI is used for predictive modeling and personalized diagnosis. By integrating multimodal data, including genetic, histological, and clinical

information, AI models enable individualized risk assessment and outcome prediction based on patient-specific factors.

Such comprehensive insights support tailored treatment strategies and advance precision oncology in breast cancer [16].

This workflow is illustrated in the imaging-to-AI pipeline shown in Figure 1.3.

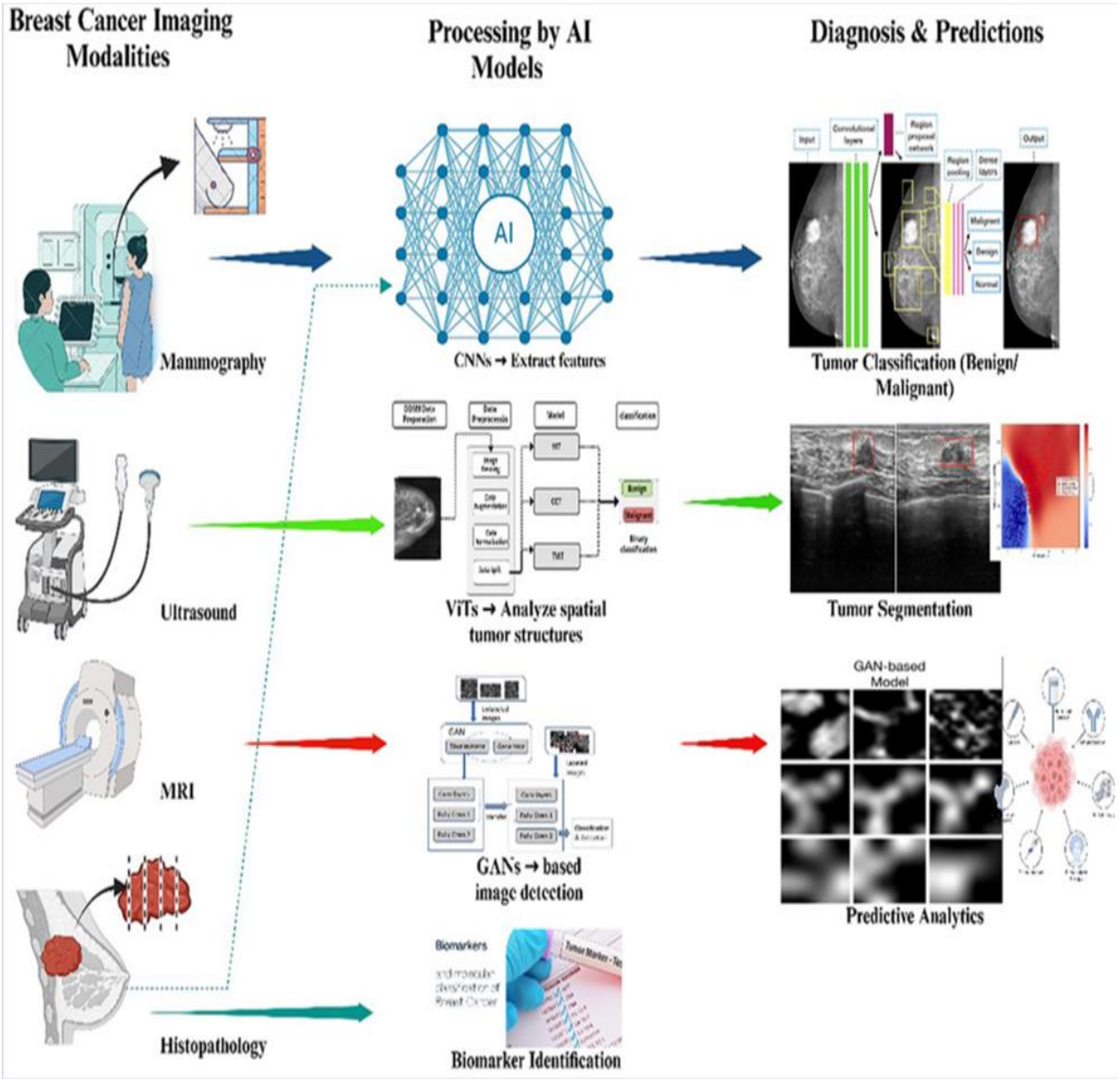

Fig. 1.3: AI pipeline for breast cancer diagnosis and prediction [16]

As shown in Figure 1.3, Imaging-to-AI pipeline for breast cancer diagnosis and prediction. Left: Clinical imaging modalities (mammography, ultrasound, MRI, and histopathology). Center representative AI families (CNNs, ViTs, GAN-based methods) and biomarker identification. Right downstream tasks: tumor classification, tumor segmentation, and predictive analytics [16].

1.6 Studies and Examples of Using AI in Breast Cancer Detection

1.6.1 Study by Spanhol et al. (2016)

Spanhol et al. introduced the BreakHis dataset, which is considered one of the most widely used datasets for breast cancer histopathological image classification. The study evaluated the effectiveness of Convolutional Neural Networks (CNNs) in distinguishing between benign and malignant breast tumours. The dataset contains microscopic breast tissue images captured under different magnification factors (40×, 100×, 200×, and 400×). The experimental results demonstrated that CNN-based approaches outperformed traditional machine learning methods based on handcrafted features, establishing a strong foundation for deep learning applications in breast cancer histopathological analysis [17].

1.6.2 Study by Voon et al. (2022)

Voon et al. conducted a comparative study of seven Convolutional Neural Network architectures using transfer learning for the classification of Invasive Ductal Carcinoma (IDC) in breast histopathological images. The study evaluated several pretrained CNN models to determine their effectiveness in identifying malignant tissue patterns. The results showed that transfer learning-based models achieved high classification performance and significantly improved the accuracy of breast cancer diagnosis in histopathological image analysis [13].

1.6.3 Study by Srikantamurthy et al. (2023)

Srikantamurthy et al. proposed a hybrid deep learning framework combining Convolutional Neural Networks (CNNs) and Long Short-Term Memory (LSTM) networks for breast cancer histopathological image classification. The proposed model was designed to automatically extract both spatial and sequential features from microscopic tissue images and classify them into benign and malignant categories. Experimental evaluation demonstrated high classification accuracy, highlighting the effectiveness of hybrid deep learning architectures in automated breast cancer diagnosis [18].

1.6.4 Study by Aldakhil et al. (2024)

Aldakhil et al. proposed an attention-based deep learning approach for multiclass classification of breast cancer histopathological images. The study employed attention mechanisms to enable the model to focus on important cancer-related regions within microscopic tissue images. The proposed framework achieved strong classification performance and demonstrated the importance of attention-based architectures in improving the accuracy and interpretability of breast cancer diagnosis systems [19].

1.7 Conclusion

In this chapter, we reviewed the basic concepts of breast cancer and its impact on public health. We presented the main traditional diagnostic methods used in clinical practice and discussed the role of artificial intelligence techniques in improving breast cancer detection and classification. We also highlighted the main challenges in diagnosis and the need for more accurate and efficient methods. In the next chapter, we present the design and methodology of the proposed automated breast cancer detection system, including the dataset preparation strategy, the model architecture, the training pipeline, and the final evaluation results.

Chapter 2: Design and Methodology

2.1 Introduction

The early and accurate detection of breast cancer is one of the most critical challenges in modern medical diagnostics. Histopathological image analysis remains the gold standard for cancer diagnosis; however, manual examination by pathologists is time-consuming, subject to inter-observer variability, and increasingly difficult to scale given the growing volume of medical imaging data.

This chapter presents the complete design and methodology of an automated breast cancer early detection system based on a custom deep learning architecture. The proposed system is designed to classify histopathological image patches of breast tissue into two categories: cancer (malignant) and normal (benign) with particular emphasis on minimizing False Negatives, which represent the most clinically dangerous type of misclassification in oncological screening.

The chapter is structured to follow the logical progression of the system pipeline. It begins with the dataset preparation strategy, covering data sourcing, balancing, and augmentation. It then details the image preprocessing steps applied before model input, followed by a comparative evaluation of candidate architectures that motivated the final model selection. The training methodology is then described in depth, covering the model architecture, loss function, class weighting strategy, optimizer configuration, and training control mechanisms. Finally, the feature extraction and classification stages are explained, and the chapter closes with a comprehensive presentation of training results including accuracy metrics, confusion matrices, and training curve analysis.

2.2 Overview of the Proposed Solution

The proposed solution is based on an integrated pipeline that employs deep learning to automatically classify histopathological image patches of breast tissue into two categories: malignant (cancer) and benign (normal) (see Figure 2.1).

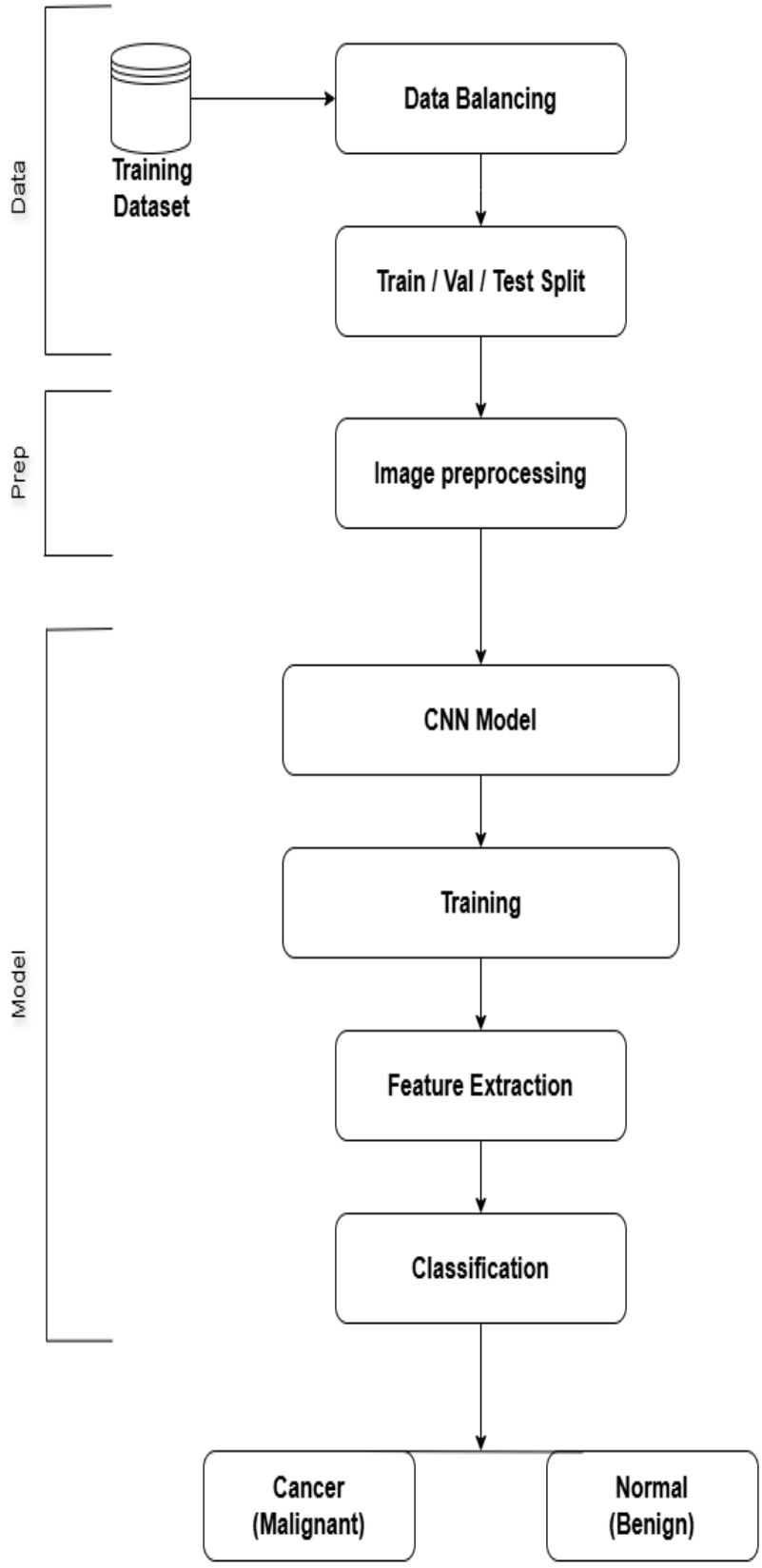

Fig. 2.1: General pipeline of the proposed breast cancer early detection system

The system is built around a custom Convolutional Neural Network (CNN) trained from scratch on the IDC histopathological dataset, without relying on any pre-trained weights.

The pipeline begins with data preparation, where images are preprocessed and augmented to improve generalization and address class imbalance. The preprocessed images are then fed into the CNN model, which automatically extracts relevant features from the tissue patches through its convolutional layers. The extracted features are subsequently passed to fully connected layers that produce the final binary classification output.

To enhance diagnostic reliability and minimize missed cancer cases, the model incorporates advanced training strategies including Focal Loss, class weighting. The system is fully implemented in Python using TensorFlow and Keras, and its performance is evaluated using standard metrics including accuracy, precision, recall, F1-score, and confusion matrix.

2.2.1 Training Dataset

2.2.1.1 Dataset Source

IDC Histopathology Dataset (Kaggle) The dataset used in this study is the Invasive Ductal Carcinoma (IDC) dataset, publicly available on Kaggle [36]. It consists of histopathological image patches extracted from whole slide images (WSIs) of breast tissue samples. The full dataset comprises 277,524 image patches, distributed across two classes: 198,738 benign patches (Class 0) and 78,786 malignant patches (Class 1).

Categories:

- Class 0: Benign (IDC Negative): Tissue patches that do not contain invasive ductal carcinoma cells.
- Class 1: Malignant (IDC Positive): Tissue patches that contain invasive ductal carcinoma cells.

The dataset consists of histopathological breast tissue images organized into two classes: cancer and normal. The raw dataset, downloaded from Kaggle, is originally organized into a patient-level directory structure, where label 1 corresponds to cancerous tissue and label 0 to normal tissue. These images were then reorganized into a unified folder structure using an automated script.

Sample Images from the IDC Histopathology Dataset:

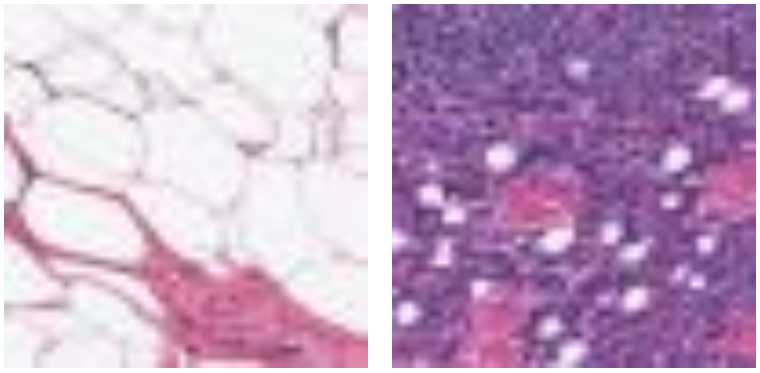

Fig. 2.2: Sample histopathological image patches from the IDC dataset.

- Left: Benign tissue sample (Class 0)
- Right: Malignant tissue sample (Class 1)

2.2.1.2 Data Balancing

Given the significant class imbalance in the original dataset (198,738 benign vs. 78,786 malignant samples), a random under-sampling strategy was applied: 78,786 images were randomly selected from each class, producing a balanced dataset of **157,572 images** with a strict 1:1 class ratio. This eliminates any numerical dominance of one class over the other at the data level.

The balanced dataset was then partitioned into three non-overlapping subsets:

Table 2.1: Dataset Partitioning Strategy and Split Ratios

Split	**Purpose**	**Percentage**	**Cancer**	**Normal**	**Total**
Training set	Model parameter optimization	70%	55,150	55,150	110,300
Validation set	Hyperparameter tuning and early stopping	15%	11,817	11,817	23,634
Test set	Final unbiased performance evaluation	15%	11,819	11,819	23,638
Total		**100%**	78,786	78,786	**157,572**

Images are loaded via Keras ImageDataGenerator using flow_from_directory, with a target resolution of 64×64 pixels and a batch size of 32.

2.2.1.3 Data Augmentation

To improve generalization and reduce overfitting, extensive data augmentation is applied exclusively to the training generator. The augmentation pipeline simulates real-world imaging variability such as orientation differences, staining inconsistencies, and illumination changes common in histopathological slides:

Table 2.2: Data Augmentation Techniques Used During Training

Augmentation Technique	**Parameter**
Random rotation	±30°
Width / height shift	±15%
Horizontal and vertical flip	Enabled
Zoom	±20%
Shear transformation	10%
Brightness variation	[0.75 – 1.25]
Channel shift	±20.0
Fill mode	Nearest neighbor

Validation and test generators applied pixel-value, normalizing image intensities to the range [0,1]. No augmentation was performed on validation or test data to ensure an unbiased evaluation of the model's performance on previously unseen samples.

2.2.2 Image Preprocessing

All images undergo a standardized preprocessing pipeline before being passed to the model.

Resizing: Every image is resized to 64×64 pixels. The input resolution was deliberately increased from an initial size of 48×48 pixels to 64×64 pixels in order to capture finer histopathological details such as nuclear morphology and cellular boundaries that are critical for distinguishing malignant from benign tissue, at the cost of a modest increase in computational load.

Normalization: Pixel values are rescaled from the integer range [0, 255] to the floating-point range [0, 1] by dividing by 255. This ensures numerical stability during gradient computation and aligns with the activation functions' expected input range.

Augmentation (training only): As described in Section 2.2.1, stochastic transformations are applied during training to simulate real-world imaging variability. No augmentation is applied at validation or test time.

2.2.3 Model Evaluation and Selection

To determine the most suitable architecture for histopathological breast cancer classification, several deep learning models were experimentally evaluated under the same dataset configuration and preprocessing pipeline. The evaluated architectures included a baseline CNN (Baseline CNN Model), an improved CNN (Enhanced CNN Model), EfficientNetB0, EfficientNetB3_Light_Model, EfficientNetB3, and the proposed final architecture Proposed CNN Model.

The comparison focused on multiple evaluation criteria, including classification accuracy, validation stability, convergence behavior, sensitivity toward cancerous samples, computational complexity, and reduction of clinically critical False Negatives. Generalization capability and training efficiency were also considered during the selection process.

2.2.3.1 Baseline CNN (Baseline CNN Model)

Building the Model

The first evaluated architecture is a lightweight sequential CNN built from scratch. The network accepts input images of size 48×48×3 and is organized into two convolutional blocks followed by a classification head. Block 1 consists of a SeparableConv2D layer with 32 filters, ReLU activation, Batch Normalization, MaxPooling (2×2), and Dropout (0.25). Block 2 consists of a SeparableConv2D layer with 64 filters, ReLU activation, Batch Normalization, MaxPooling (2×2), and Dropout (0.25). The classification head uses a Flatten layer, a Dense layer with 256 units, Batch Normalization, Dropout (0.5), and a final Dense layer with Softmax activation, as shown in Table 2.3.

Table 2.3: Architecture of Baseline CNN Model

Layer	**Filters / Units**	**Output Size**
SeparableConv2D + BN + ReLU	32	48×48×32
MaxPooling + Dropout (0.25)	—	24×24×32
SeparableConv2D + BN + ReLU	64	24×24×64
MaxPooling + Dropout (0.25)	—	12×12×64
Flatten	—	9216
Dense + BN + Dropout (0.5)	256	256
Dense + Softmax	2	2

Train the Model

The model was compiled using Categorical Cross-Entropy loss and the Adagrad optimizer with a learning rate of 1×10^{-2}. Class weights were applied to address class imbalance. Training was conducted for a maximum of 20 epochs with a batch size of 32, using ModelCheckpoint and EarlyStopping with a patience of 5 epochs.

Table 2.4: Training Configuration of Baseline CNN Model

Parameter	**Value**
Input size	48×48×3
Batch size	32
Max epochs	20
Optimizer	Adagrad ($lr=1\times10^{-2}$)
Loss function	Categorical Cross-Entropy
Early stopping patience	5

Obtained Results:

Table 2.5: Classification Results of Baseline CNN Model

Class	Precision	Recall	F1-score
Cancer	0.85	0.87	0.86
Normal	0.87	0.85	0.86

Test accuracy: 86.22% Macro F1-score: 0**.**86 False Negatives: 1**,**749

Although the model achieved good classification performance, its capability to handle difficult histopathological samples and reduce False Negatives remained limited.

2.2.3.2 Improved CNN (Enhanced CNN Model)

Building the Model

The second evaluated architecture is an improved version of the baseline CNN. The network accepts input images of size 48×48×3 and introduces a deeper structure with three convolutional blocks. Block 1 contains one SeparableConv2D layer with 32 filters. Block 2 contains two successive SeparableConv2D layers with 64 filters each, and Block 3 contains two successive SeparableConv2D layers with 128 filters each, providing richer feature extraction. Each block is followed by Batch Normalization, MaxPooling(2×2), and Dropout (0.25). The classification head replaces the Flatten layer with a GlobalAveragePooling2D layer to reduce overfitting, followed by two Dense layers (256 and 128 units) with Dropout regularization and a Softmax output layer, as shown in Table 2.6.

Table 2.6: Architecture of Enhanced CNN Model

Layer	**Filters/Units**	**Output Size**
SeparableConv2D + BN + ReLU	32	48×48×32
MaxPooling + Dropout (0.25)	—	24×24×32
SeparableConv2D ×2 + BN + ReLU	64	24×24×64
MaxPooling + Dropout (0.25)	—	12×12×64
SeparableConv2D ×2 + BN + ReLU	128	12×12×128
MaxPooling + Dropout (0.25)	—	6×6×128
GlobalAveragePooling2D	—	128
Dense + BN + Dropout (0.5)	256	256
Dense + Dropout (0.3)	128	128
Dense + Softmax	2	2

Train the Model

The model was compiled using Categorical Cross-Entropy loss and the Adam optimizer with an initial learning rate of 3×10^{-4}. Training was conducted for a maximum of 40 epochs with a batch size of 32. Data augmentation was applied during training including rotation (±20°), width/height shifting (±10%), horizontal and vertical flipping, and zooming (±10%). Three callbacks were used: ModelCheckpoint, EarlyStopping with patience of 8 epochs, and ReduceLROnPlateau with a patience of 3 and a reduction factor of 0.5.

Table 2.7: Training Configuration of Enhanced CNN Model

Parameter	**Value**
Input size	48×48×3
Batch size	32
Max epochs	40
Optimizer	Adam ($lr=3\times10^{-4}$)
Loss function	Categorical Cross-Entropy
Early stopping patience	8
LR reduction patience	3 (factor=0.5)

Obtained Results:

Table 2.8: Classification Results of Enhanced CNN Model

Class	**Precision**	**Recall**	**F1-score**
Cancer	0.84	0.90	0.87
Normal	0.89	0.83	0.86

Test accuracy: 86.31% Test Loss: 0.3321 Macro F1-score: 0.86 False Negatives: 1,237

Compared to Baseline CNN Model, Enhanced CNN Model demonstrated a notable improvement in cancer recall (0.90 vs 0.87) and a significant reduction in False Negatives (1,237 vs 1,749), confirming the benefit of the deeper architecture and GlobalAveragePooling. However, further improvements in overall accuracy and sensitivity were still required.

2.2.3.3 EfficientNetB0

Building the Model

EfficientNetB0 was evaluated using transfer learning. The pretrained EfficientNetB0 backbone, originally trained on ImageNet, was used as the feature extractor with the top layers excluded. The added classification head consists of a GlobalAveragePooling2D layer, Batch Normalization, a Dense layer with 256 units and ReLU activation, Dropout

(0.4), a Dense layer with 128 units and ReLU activation, Dropout (0.3), and a final Dense layer with 2 units and Softmax activation, as shown in Table 2.9.

Table 2.9: Architecture of EfficientNetB0

Layer	**Units**	**Notes**
EfficientNetB0 backbone	—	Pretrained on ImageNet, top excluded
GlobalAveragePooling2D	—	Feature aggregation
BatchNormalization	—	—
Dense + ReLU	256	—
Dropout (0.4)	—	—
Dense + ReLU	128	—
Dropout (0.3)	—	—
Dense + Softmax	2	Binary output

Train the Model

Training was conducted in two phases. In Phase 1, the base model was fully frozen and only the classification head was trained for 10 epochs with a learning rate of 1×10^{-3}. In Phase 2, the last 30 layers were unfrozen for fine-tuning at a reduced learning rate of 1×10^{-4}, while all BatchNormalization layers remained frozen. ModelCheckpoint, EarlyStopping (patience=7), and ReduceLROnPlateau (patience=3, factor=0.5) callbacks were used in both phases.

Table 2.10: Training Configuration of EfficientNetB0

Parameter	**Phase 1**	**Phase 2**
Base model	Fully frozen	Last 30 layers unfrozen
Learning rate	1×10^{-3}	1×10^{-4}
Max epochs	10	30
Batch size	64	64
Loss function	Categorical Cross-Entropy	Categorical Cross-Entropy

Obtained Results:

Table 2.11: Classification Results of EfficientNetB0

Class	**Precision**	**Recall**	**F1-score**
Cancer	0.74	0.82	0.78
Normal	0.80	0.71	0.75

Test accuracy: 76.70% Macro F1-score: 0.77 False Negatives: 2,072

Although EfficientNetB0 showed reasonable cancer recall, the overall accuracy and F1-score were significantly lower than the custom CNN approaches, and the validation process showed instability during several epochs, confirming that ImageNet features do not transfer optimally to histopathological image classification at this resolution.

2.2.3.4 EfficientNetB3_Light_Model

Building the Model

EfficientNetB3_Light_Model was evaluated using a pretrained EfficientNetB3 backbone with input size 96×96×3, adapted using transfer learning. The classification head follows the same structure as EfficientNetB0GlobalAveragePooling2D, Batch Normalization, Dense (256) +ReLU, Dropout (0.4), Dense (128) +ReLU, Dropout (0.3), and a Softmax output layer, as shown in Table 2.12.

Table 2.12: Architecture of EfficientNetB3_Light_Model

Layer	**Units**	**Notes**
EfficientNetB3 backbone	—	Pretrained on ImageNet, top excluded
GlobalAveragePooling2D	—	Feature aggregation
BatchNormalization	—	—
Dense + ReLU	256	—
Dropout (0.4)	—	—
Dense + ReLU	128	—
Dropout (0.3)	—	—
Dense + Softmax	2	Binary output

Train the Model

Training was conducted in two phases with a fast configuration. In Phase 1, the base model was frozen and the head was trained for 10 epochs with a learning rate of 1×10^{-3}. In Phase 2, the last 30 layers were unfrozen for fine-tuning at 1×10^{-4} for 15 epochs. EfficientNet's built-in preprocessing function was applied to input images. SaveBestH5, EarlyStopping (patience=5), and ReduceLROnPlateau (patience=2, factor=0.5) callbacks were used.

Table 2.13: Training Configuration of EfficientNetB3_Light_Model

Parameter	**Phase 1**	**Phase 2**
Base model	Fully frozen	Last 30 layers unfrozen
Learning rate	1×10^{-3}	1×10^{-4}
Max epochs	10	15
Batch size	16	16
Input size	96×96×3	96×96×3

Obtained Results:

Table 2.14: Classification Results of EfficientNetB3_Light_Model

Class	**Precision**	**Recall**	**F1-score**
Cancer	0.85	0.85	0.85
Normal	0.85	0.85	0.85

Test accuracy: 84.79% Macro F1-score: 0.85 False Negatives: 1,779

EfficientNetB3_Light_Model outperformed EfficientNetB0 in overall accuracy but still fell short of the custom CNN architectures, particularly in cancer recall and False Negative reduction.

2.2.3.5 EfficientNetB3

EfficientNetB3 was also experimentally evaluated in order to investigate the effect of a deeper and more complex pretrained architecture. During training, EfficientNetB3 required significantly higher computational resources and memory consumption compared to the other evaluated models. Multiple memory allocation warnings were

observed during execution, indicating heavy hardware requirements. Although the training process initially showed promising learning behavior with training accuracy reaching approximately 76.7% during the early phase, the experiment could not be completed due to computational limitations and interruption of the training process.

Consequently, EfficientNetB3 was not selected because:

- The model required excessive computational resources
- Training stability was insufficient
- Execution time was considerably higher
- Practical deployment efficiency was reduced

Therefore, despite its theoretical representational power, EfficientNetB3 was considered unsuitable for the final system implementation.

2.2.3.6 Proposed Model: Proposed CNN Model

Building the Model

Based on the limitations observed in the previously evaluated architectures particularly in terms of cancer recall, False Negative rate, and classification robustness on difficult histopathological samples a custom deep learning architecture named Proposed CNN Model was proposed as the final model for this work.

Proposed CNN Model was specifically designed to address these shortcomings by integrating several complementary optimisation techniques, including residual connections, Squeeze-and-Excitation (SE) attention blocks, Separable Convolution layers, Focal Loss, cancer-boosted class weighting, Cosine Annealing learning rate scheduling, extensive data augmentation. The model also adopts a larger input resolution of 64×64 pixels compared to previous architectures, in order to preserve richer histopathological detail during feature extraction. The full architectural design and training configuration of Proposed CNN Model are described in detail in Section 2.2.4.

Obtained Results:

Table 2.15: Classification Results of Proposed CNN Model

Class	**Precision**	**Recall**	**F1-score**
Cancer	0.86	0.925	0.89
Normal	0.925	0.85	0.88

Test accuracy: 88.81% Test Loss: 0.0631 Macro F1-score: 0.89 False Negatives: 884

The cancer recall of 0.925 is particularly significant in a clinical context, as it indicates that 92.5% of all malignant patches were correctly identified, substantially reducing the risk of missed diagnoses compared to all previously evaluated models.

Final Model Selection

After comparing all evaluated architectures, Proposed CNN Model demonstrated the best balance between classification accuracy, robustness, sensitivity, computational efficiency, and generalization capability. The comparative summary is presented in Table 2.16:

Table 2.16: Comparative Evaluation of All Models on the Test Set

Metrics	**Proposed CNN Model**	**Enhanced CNN Model**	**Baseline CNN Model**	**EfficientNetB3_Light_ Model**	**EfficientNetB0**
Accuracy	88.81%	86.31%	86.22%	84.79%	76.70%
Precision (cancer)	0.86	0.84	0.85	0.85	0.74
Recall (cancer)	0.925	0.90	0.87	0.85	0.82
F1-Score (cancer)	0.89	0.87	0.86	0.85	0.78
False Negatives	884	1,237	1,749	1,779	2,072

Compared with all other evaluated architectures, Proposed CNN Model achieved higher overall accuracy, better cancer detection sensitivity, lower False Negative rates, more stable convergence behavior, lower computational cost, and better suitability for

practical deployment. Therefore, Proposed CNN Model was selected as the final architecture adopted in this work for the breast cancer early detection system.

2.2.4 Training

Model Architecture

The proposed model is a custom CNN built around Residual Squeeze-and-Excitation (SE) blocks. The architecture is organized as follows:

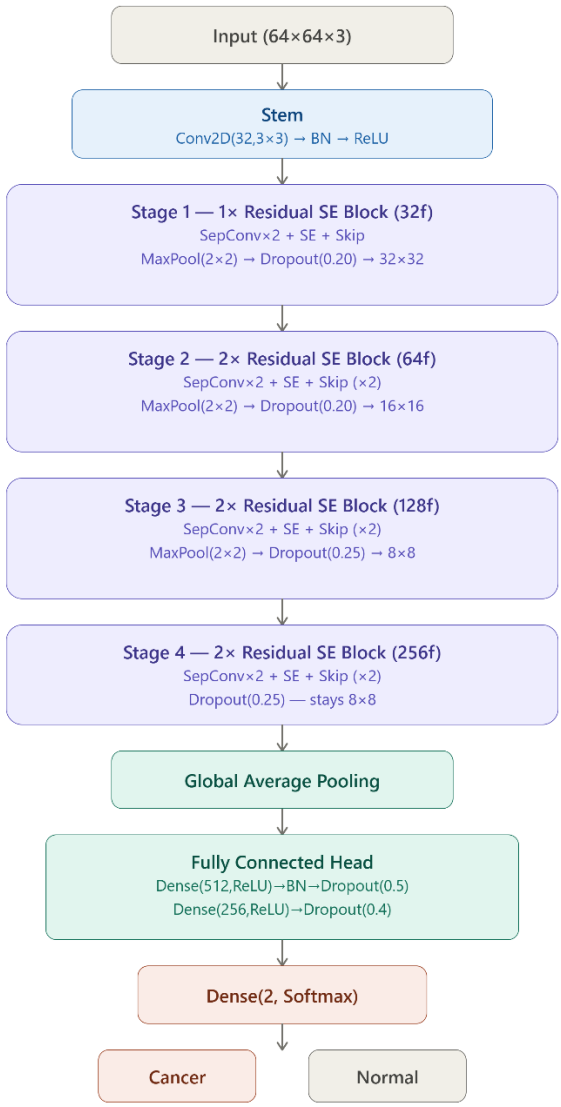

Fig. 2.3: Architecture of Proposed CNN Model

The architecture begins with an Input layer of size 64×64×3, followed by a Stem block (Conv2D 32 filters, 3×3 → BatchNorm → ReLU) that extracts initial low-level features. The network then passes through four progressive stages built from Residual SE Blocks: Stage 1 (1× block, 32 filters) → MaxPool → Dropout(0.20) → 32×32;

Stage 2 (2× blocks, 64 filters) → MaxPool → Dropout(0.20) → 16×16; Stage 3 (2× blocks, 128 filters) → MaxPool → Dropout(0.25) → 8×8; and Stage 4 (2× blocks, 256 filters) → Dropout(0.25), without pooling, preserving the 8×8 resolution for fine-grained feature retention.

Each Residual SE block consists of two SeparableConv2D layers with L2 regularization (λ=1e-4), Batch Normalization, and a Squeeze-and-Excitation module that recalibrates channel-wise feature responses through a learned attention mechanism, with a skip connection ensuring stable gradient flow. Following the convolutional stages, Global Average Pooling converts the final 8×8×256 feature maps into a 256-dimensional vector, which feeds a classification head of Dense(512, ReLU)→BN→Dropout(0.5) and Dense(256, ReLU, L2)→Dropout(0.4), ending in a Dense(2, Softmax) output layer producing the Cancer/Normal classification.

Table 2.17: Training Configuration of proposed CNN Model

Parameter	**Value**
Input size	64×64×3
Batch size	32
Max epochs	50
Optimizer	Adam (lr=3×10^{-4})
Loss function	Focal Loss (γ=2.0, α=0.75)
Early stopping patience	12
LR schedule	Cosine Annealing (warmup=5, lr_min=1×10^{-6})
Class weight (cancer)	×2.0
Class weight (normal)	×0.8

Loss Function -Focal Loss

Standard cross-entropy treats all samples equally regardless of prediction difficulty. To focus training on hard, misclassified examples, Focal Loss was adopted:

$$FL(p_t) = -\alpha_t(1-p_t)^{\gamma}\log(p_t)$$

with $\gamma = 2.0$and $\alpha = 0.75$, assigning 75% of the loss weight to cancer samples and 25% to normal samples, reinforcing the model's sensitivity toward cancerous cases.

Clinical Class Weighting Strategy

Although the dataset is numerically balanced (Section 2.2.1), a cancer-boosted class weighting strategy was additionally applied during training via the class_weight argument in model.fit(). This decision is motivated by the clinical asymmetry between error types: a False Negative failing to detect a cancerous case carries a significantly greater medical risk than a False Positive, which results only in additional follow-up examinations.

Class weights are computed dynamically from the training label distribution:

$$w_c = \frac{N_{total}}{K \cdot N_c}$$

where $N_{total}$is the total number of training samples, $K$is the number of classes, and $N_c$is the count for class c. An additional multiplicative factor is applied: the cancer class weight is scaled by ×2.0 to further penalize missed detections, while the normal class weight is reduced by ×0.8 to reflect its lower clinical priority. This instructs the model to treat each missed cancer case as twice as costly as a false alarm.

Optimizer and Learning Rate Schedule

The model is compiled with the Adam optimizer (initial lr = 3×10^{-4}). A Cosine Annealing with Warmup schedule is applied across 50 epochs:

- **Warmup phase** (epochs 1–5): learning rate increases linearly from 0 to lr_max = 3×10^{-4}.
- **Annealing phase** (epochs 6–50): learning rate follows a cosine decay down to lr_min = 1×10^{-6}.

This schedule prevents large destabilizing updates in early epochs while allowing fine-grained convergence in later stages.

Saving Best Model

Throughout training, three callbacks work simultaneously inside model.fit (). The SaveBestH5 callback monitors validation accuracy at the end of each epoch and automatically saves the model in .h5 format whenever an improvement is observed,

preserving the full architecture and learned weights. The EarlyStopping callback halts training if no improvement in validation accuracy is detected for 12 consecutive epochs, then restores the best weights automatically. The CosineAnnealingLR callback adjusts the learning rate at the beginning of each epoch according to the cosine annealing schedule described above. Together, these three callbacks ensure that the final model corresponds to the epoch of optimal generalization, prevents unnecessary training beyond convergence, and maintains a stable learning rate throughout.

2.2.5 Feature Extraction

Feature extraction is performed implicitly by the convolutional stages of the network. The hierarchical structure progressively extracts features of increasing abstraction:

Low-level features (Stages 1–2, 32–64 filters): edges, textures, color gradients, and staining patterns characteristic of tissue slides.

Mid-level features (Stage 3, 128 filters): cellular structures, nuclear morphology, and local tissue organization patterns.

High-level features (Stage 4, 256 filters): complex discriminative patterns differentiating malignant from benign tissue at a semantic level.

The Squeeze-and-Excitation module enhances this process by applying channel-wise attention: it globally pools each feature map, learns a recalibration weight per channel through a small bottleneck network, and rescales the feature maps accordingly allowing the model to focus on the most diagnostically informative channels and suppress irrelevant ones.

The final GlobalAveragePooling2D layer aggregates the spatial feature maps into a compact feature vector, which serves as the input to the classification head.

2.2.6 Classification

The classification head maps the extracted feature vector to output class probabilities through two fully connected layers:

- **Dense (512)** with ReLU activation, Batch Normalization, and Dropout (0.5)
- **Dense (256)** with ReLU activation and Dropout (0.4)
- **Dense (2)** with Softmax activation producing a probability distribution over the two classes

The predicted class is determined by:

$$\hat{y} = \arg\max_{c} \; p(c \mid x)$$

2.2.7 Training Results

The model was trained for up to 50 epochs with early stopping. Final evaluation was performed on the held-out test set of 23,638 images (11,819 cancer / 11,819 normal).

Overall Performance :

```
Test Loss     : 0.0631
Test Accuracy : 88.81%
```

Fig. 2.4: Model Accuracy Results on Test data

Per-Class Performance:

```
========== CLASSIFICATION REPORT ==========
              precision    recall  f1-score   support

      Cancer       0.86      0.93      0.89     11819
      Normal       0.92      0.85      0.88     11819

    accuracy                           0.89     23638
   macro avg       0.89      0.89      0.89     23638
weighted avg       0.89      0.89      0.89     23638
```

Fig. 2.5: Performance Metrics on the Test Dataset

Confusion Matrix:

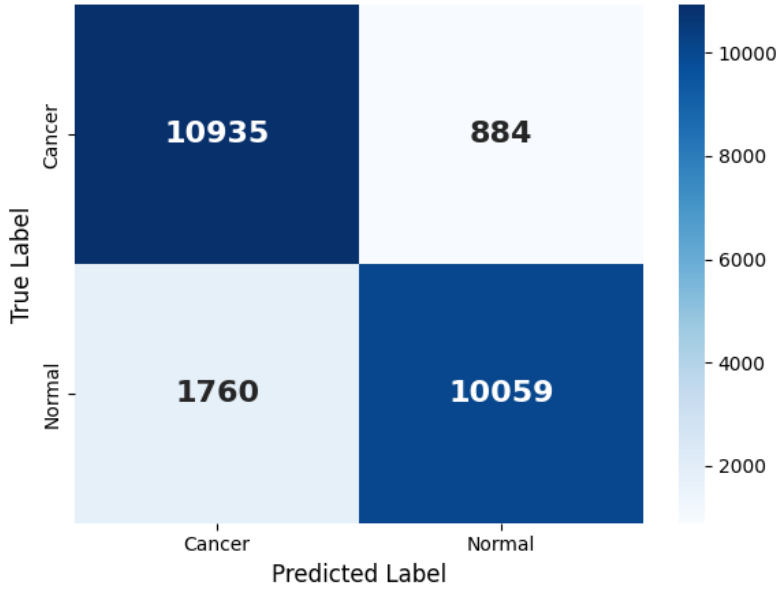

Fig. 2.6: Confusion Matrix on Test Set

The model achieves a cancer recall of 92.5%, correctly identifying 10,935 out of 11,819 cancerous cases. The False Negative count (884) is substantially lower than the False Positive count (1,760), confirming that the combined effect of Focal Loss and cancer-boosted class weighting successfully oriented the model toward minimizing the clinically most dangerous error type.

Training Curves Analysis

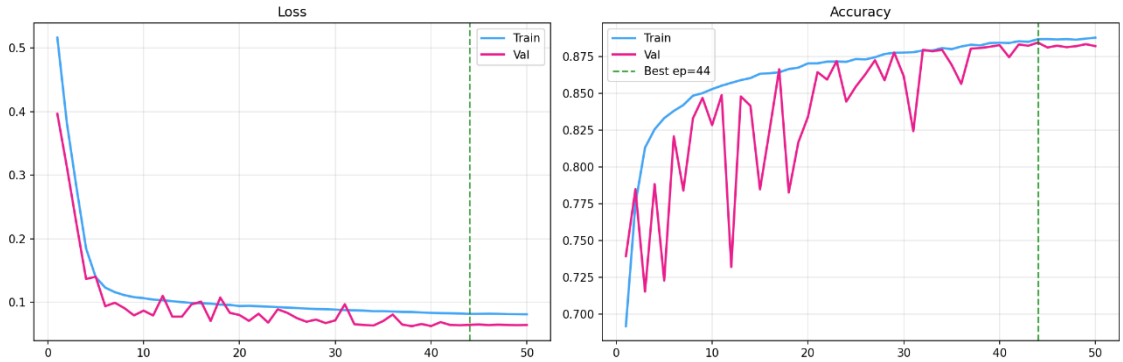

Fig. 2.7: Loss and accuracy curves over 50 training epochs best epoch = 44 (green dashed line)

Loss Curve: Both training and validation loss decrease sharply during the first 10 epochs, then converge smoothly toward low values (~0.07), with no significant

divergence between the two curves. This indicates that the model learned effectively without overfitting.

Accuracy Curve: Training accuracy increases steadily throughout, reaching approximately 88.8%. Validation accuracy exhibits higher variance during the early epochs (epochs 8–13), attributable to the cosine annealing warmup phase and the stochastic nature of data augmentation. After epoch 15, the validation curve stabilizes and converges, peaking at 88.44% at epoch 44.

The close alignment between training and validation curves confirms that the regularization strategy combining Dropout, L2 weight decay, Batch Normalization, and data augmentation successfully prevented overfitting, and the final test accuracy of 88.81% reflects the model's strong generalization capability.

2.3 Conclusion

This chapter presented a complete end-to-end methodology for automated breast cancer detection from histopathological images. The proposed system was developed through a systematic pipeline encompassing data preparation, image preprocessing, model evaluation and selection, training, feature extraction, and classification.

A comparative evaluation of multiple architectures including a baseline CNN, EfficientNetB0, EfficientNetB3, and the proposed Proposed CNN Model demonstrated that the custom Residual SE-CNN architecture achieved the best balance between classification accuracy, computational efficiency, and clinical suitability. The integration of Focal Loss, cancer-boosted class weighting, Cosine Annealing scheduling produced a model specifically oriented toward minimizing missed cancer detections.

The final model achieved a test accuracy of 88.81%, a cancer recall of 92.5%, and a macro-average F1-score of 0.89 on a balanced test set of 23,638 histopathological images. The close alignment between training and validation curves confirmed that the regularization strategy successfully prevented overfitting, while the confusion matrix analysis validated the clinical effectiveness of the weighting approach with False Negatives (884) remaining substantially lower than False Positives (1,760).

These results confirm the viability of the proposed system as a reliable decision-support tool for breast cancer screening, providing a reproducible and transparent foundation for further development and clinical validation.

Chapter 3: Development and Implementation

3.1 Introduction

This chapter presents the technical foundation upon which the intelligent system for early detection of breast cancer is built. It details the adopted development environment and the advanced programming libraries utilised to perform complex tasks such as image preprocessing, deep learning model training, and automatic classification of histopathological images. The chapter also addresses the design of the graphical user interface (GUI), which plays a key role in facilitating interaction between medical professionals and the system through a smooth and practical user experience.

3.2 Programming Language

Python was selected as the primary programming language for the development of this breast cancer detection system. Python is widely adopted in the field of artificial intelligence and medical image analysis due to its simplicity, readability, and extensive ecosystem of scientific libraries. Its open-source nature and strong community support make it particularly suitable for research and development in deep learning applications.

Python offers seamless integration with powerful libraries such as TensorFlow, Keras, NumPy, Matplotlib, Seaborn, and Scikit-learn, which are essential for implementing the various stages of the proposed system, including image preprocessing, model training, and classification. Furthermore, Python's compatibility with interactive development environments such as Jupyter Notebook and Google Colab enables efficient experimentation and rapid prototyping, which are critical requirements in medical imaging research [20].

3.3 Development Environment Adopted

This section introduces the tools and platforms used for writing and testing the breast cancer detection system.

3.3.1 Visual Studio Code

In this project, Visual Studio Code (VS Code) was selected as the primary integrated development environment (IDE) for writing and testing the deep learning code. VS Code is a lightweight yet powerful source code editor developed by Microsoft, widely

adopted in the scientific and engineering communities due to its extensive support for Python development through a rich collection of extensions, built-in debugging tools, and seamless integration with virtual environments.

VS Code facilitated the process of writing and organizing the various components of the project, including data preprocessing, model training, and result visualization, within a single unified workspace. Furthermore, its robust support for virtual environments helped manage Python packages and dependencies independently, ensuring a stable and reproducible development setup. Its advanced debugging tools allowed step-by-step code tracing and execution, making it easier to detect issues and improve the quality of the implemented models. Finally, VS Code is an open-source and free option that runs efficiently on Windows operating systems, enhancing accessibility throughout the project implementation [21].

3.3.2 Hardware Specifications

All model development, training, and evaluation were conducted locally on a personal computer with the following hardware specifications:

Table 3.1: Hardware Specifications of the Development Machine

Component	**Specification**
Operating System	Windows 11, 64-bit
Processor	AMD Ryzen 5 PRO 5675U with Radeon Graphics, 2.30 GHz
Installed RAM	16.0 GB (15.3 GB usable)
Storage	477 GB SSD — KIOXIA KBG40ZNV512G
Graphics Card	AMD Radeon Graphics (496 MB)
System Type	x64-based processor

3.4 Used Programming Libraries

Software libraries are fundamental components in modern software development, as they provide ready-to-use, well-tested functions that enable developers to accelerate the development process and reduce the amount of code required. Additionally, they

contribute to enhancing software quality and expanding the capabilities of the programming language in use. In the context of this project, several essential libraries were utilised to efficiently and effectively perform various tasks. The following sections highlight the most important of these libraries and their vital role in supporting the project's functionalities and achieving its goals.

3.4.1 TensorFlow

TensorFlow is an open-source machine learning library developed by Google, designed to facilitate the construction and deployment of machine learning and deep learning models across a wide range of applications. It provides a comprehensive ecosystem for building computational graphs, optimising model execution, and supporting deployment on diverse hardware platforms, including CPUs, GPUs, and TPUs. Its core functionalities include:

- Building and training machine learning models for tasks such as classification, regression, and clustering.
- Providing low-level tensor operations and automatic differentiation.
- Supporting hardware-accelerated computation across CPUs, GPUs, and TPUs.

In this project, TensorFlow served as the primary deep learning framework, providing the computational backbone for training the breast cancer classification model. Specifically, it was used to implement the custom Focal Loss function through tf.clip_by_value, tf.math.log, and tf.pow operations, and to load and run inference with the trained model within the graphical user interface [22].

3.4.2 Keras

Keras is a high-level Python library built on top of TensorFlow, designed to provide a user-friendly and modular interface for developing deep learning solutions. It abstracts many of the low-level complexities of deep learning, allowing for rapid prototyping and experimentation. Its key functionalities include:

- Constructing neural network architectures through a layered modular API.
- Supporting data preprocessing and augmentation via ImageDataGenerator.
- Defining loss functions, optimisers, and training callbacks.
- Enabling hyperparameter tuning and rapid experimentation.

In this project, Keras was used to design and build the full convolutional neural network architecture. This included defining the convolutional layers using Conv2D, SeparableConv2D, and BatchNormalization, constructing the residual and Squeeze-and-Excitation blocks using Add and Multiply, configuring the Adam optimiser with Focal Loss, and employing the EarlyStopping callback alongside a custom CosineAnnealingLR scheduler. The ImageDataGenerator was also used for data augmentation during training [23].

3.4.3 NumPy

NumPy, short for Numerical Python, is a fundamental Python library for scientific computing. It introduces support for large, multidimensional arrays and matrices, alongside a rich collection of mathematical functions to operate on these structures efficiently. Its functionalities include:

- Performing mathematical and statistical operations on arrays.
- Supporting array manipulation, reshaping, and indexing.
- Providing linear algebra computations and broadcasting operations.
- Enabling efficient numerical data processing for machine learning pipelines.

In this project, NumPy was employed throughout the model development pipeline for several critical tasks: computing class weights using np.bincount to address class imbalance, determining the best training epoch with np.argmax, and preprocessing individual images using np.expand_dims before feeding them into the model for inference [24].

3.4.4 Matplotlib

Matplotlib is an open-source Python plotting library that supports the creation of a wide variety of static, interactive, and animated visualisations. It offers a flexible interface for customising charts, graphs, and figures with fine-grained control over visual elements. Its primary functionalities include:

- Generating line plots, bar charts, scatter plots, and multi-panel subplots.
- Configuring axes, labels, titles, legends, and grid lines.
- Saving figures to disk in various formats such as PNG and PDF.
- Embedding plots directly into graphical user interfaces via backend toolkits.

In this project, Matplotlib was used in two contexts. During training, it was used to plot the training and validation accuracy and loss curves across epochs, with a vertical dashed line marking the best epoch. In the graphical user interface, it was used within the evaluation panel to render a styled confusion matrix heatmap embedded directly into the application window via FigureCanvasTkAgg [25].

3.4.5 Seaborn

Seaborn is a Python data visualisation library built on top of Matplotlib, offering a higher-level and more aesthetically refined interface for generating informative statistical graphics. It simplifies the creation of complex visualisations while integrating seamlessly with pandas data structures. Its main functionalities include:

- Generating heatmaps, distribution plots, and categorical plots.
- Providing built-in support for statistical estimation and annotation.
- Offering a variety of built-in colour palettes for clearer visual distinction.
- Simplifying the creation of publication-ready statistical figures.

In this project, Seaborn was used in the training script to generate confusion matrix heatmaps using sns.heatmap, enabling a clear and detailed visual comparison of the model's classification performance. The resulting figures were saved to disk as high-resolution PNG files for analysis and reporting [26].

3.4.6 Scikit-learn

Scikit-learn is an open-source Python library that provides a comprehensive suite of tools for machine learning and predictive data analysis. It is built on top of NumPy, SciPy, and Matplotlib, and offers consistent interfaces across a broad range of algorithms. Its functionalities include:

- Providing tools for classification, regression, clustering, and dimensionality reduction.
- Computing evaluation metrics such as precision, recall, F1-score, and accuracy.
- Generating confusion matrices for detailed classification analysis.
- Supporting model selection and cross-validation techniques.

In this project, Scikit-learn was employed in both the training script and the evaluation panel of the graphical user interface. The confusion matrix function was used to compute the matrix of true and predicted class labels, while classification report was used to generate detailed per-class performance metrics for both the Cancer and Normal classes [27].

3.4.7 Pillow (PIL)

Pillow is a Python imaging library that extends the original Python Imaging Library (PIL) with broader format support and an improved interface for image processing tasks. It enables opening, reading, manipulating, and saving images in a wide range of formats such as JPEG, PNG, and BMP. Its functionalities include:

- Opening and reading image files in a wide range of formats.
- Performing image resizing, cropping, and colour mode conversion.
- Integrating with NumPy arrays for efficient pixel-level manipulation.
- Supporting image rendering within graphical user interfaces.

In this project, Pillow was used in two distinct roles. In the prediction pipeline, it was used to open histopathological images, convert them to RGB format using Image.open().convert("RGB"), and resize them to 64×64 pixels before converting them to NumPy arrays for model inference. In the graphical user interface, it was also used to load the application icons and to display the uploaded image in the prediction panel after resizing it to fit the preview area [28].

3.4.8 CustomTkinter

CustomTkinter is a modern Python UI library built on top of the standard Tkinter framework, developed to overcome the visual limitations of traditional Tkinter by providing enhanced, customisable widgets with a contemporary aesthetic. It supports custom themes and offers a variety of updated graphical elements suited for building polished desktop applications. Its functionalities include:

- Providing modern widgets such as buttons, labels, input fields, and progress bars.
- Supporting scrollable frames for displaying large amounts of content.
- Enabling custom colour themes and both light and dark mode support.

- Allowing fine-grained control over widget appearance, corner radius, and borders.

In this project, CustomTkinter was used to design and implement the complete graphical user interface of the breast cancer classification system. This encompassed the landing page, the login and registration pages with form validation, and the main application shell with a sidebar navigation panel. The inner pages included the Welcome dashboard, the Training results panel with embedded Matplotlib charts, the Evaluation panel with a live confusion matrix, and the Prediction interface supporting both single-image and batch-image classification with confidence progress bars and tabular result display [29].

3.4.9 OS

The os module is a built-in component of the Python standard library that provides a platform-independent interface for interacting with the underlying operating system. It abstracts operating-system-specific functionality, making code portable across different environments. Its functionalities include:

- Constructing and managing file and directory paths in a platform-independent manner.
- Creating directories and checking for the existence of files and folders.
- Retrieving and modifying environment variables.
- Joining and inspecting file paths using os.path utilities.

In this project, the os module was used to construct and manage all dataset directory paths including the training, validation, and test paths through os.path.join. It was also used to create the timestamped output folder for saving the best model checkpoint and evaluation plots with os.makedirs, and to check for the existence of the user database file with os.path.exists within the authentication system [30].

3.4.10 Datetime

The datetime module is a built-in component of the Python standard library that supplies classes for manipulating dates and times. It provides a straightforward interface for retrieving the current date and time, performing arithmetic on date and time objects, and formatting them as readable strings. Its functionalities include:

- Retrieving the current date and time using datetime.datetime.now().
- Formatting date and time values into custom string representations.
- Supporting arithmetic operations on date and time objects.
- Enabling time-based logging and file naming in applications.

In this project, the datetime module was used in the training script to generate a unique timestamp string at the start of each training run using datetime.datetime.now().strftime('%Y%m%d_%H%M%S'). This timestamp was appended to the name of the output directory where the best model checkpoint and evaluation plots were saved, ensuring that results from different training sessions are stored separately without overwriting one another [31].

3.4.11 Threading

The threading module is a built-in component of the Python standard library that enables concurrent execution of code through the use of multiple threads within a single process. It allows long-running tasks to be executed in the background without blocking the main execution flow. Its functionalities include:

- Creating and managing background threads using the Thread class.
- Running time-consuming tasks concurrently with the main application thread.
- Supporting daemon threads that terminate automatically when the main program exits.
- Preventing UI freezing by offloading heavy computations to separate threads.

In this project, the threading module was used in the graphical user interface to run both the model evaluation and the image prediction tasks in dedicated background threads using threading.Thread(target=..., daemon=True).start(). This ensured that the interface remained responsive and did not freeze while the model was processing images or evaluating the test dataset [32].

3.4.12 Tkinter (filedialog)

Tkinter is the standard Python library for building graphical user interfaces, and its filedialog submodule provides built-in dialogue windows for file and directory selection. It integrates natively with the operating system's file browser, offering a familiar and consistent experience across platforms. Its functionalities include:

- Opening a file selection dialogue to browse and choose one or more files.
- Opening a directory selection dialogue to choose a folder path.
- Filtering selectable files by extension type.
- Returning the selected path or paths as strings for further processing.

In this project, the filedialog submodule from Tkinter was used in the graphical user interface to allow users to browse and select histopathological images for prediction using filedialog.askopenfilenames, supporting both single and batch image selection. It was also used in the Training and Evaluation panels to allow users to choose a custom dataset folder through filedialog.askdirectory [33].

3.4.13 IO

The io module is a built-in component of the Python standard library that provides Python's core tools for working with streams of data, both text and binary. It allows data to be read from and written to in-memory buffers rather than physical files on disk. Its functionalities include:

- Creating in-memory binary streams using io.BytesIO.
- Reading and writing binary data without the need for temporary files.
- Serving as an intermediary buffer between data generation and consumption.
- Enabling efficient in-memory image handling and format conversion.

In this project, the io module was used in the evaluation panel of the graphical user interface to save the rendered confusion matrix figure into an in-memory binary buffer using io.BytesIO, instead of writing it to disk. The buffer was then passed directly to Pillow's Image.open() to convert it into an image object that could be displayed within the application window, avoiding unnecessary file I/O operations [34].

3.4.14 JSON

The json module is a built-in component of the Python standard library that provides functions for encoding and decoding data in JavaScript Object Notation (JSON) format. It enables seamless serialisation of Python objects into JSON strings and their deserialisation back into Python data structures. Its functionalities include:

- Serialising Python dictionaries and lists into JSON-formatted strings using json.dump.
- Deserialising JSON-formatted strings back into Python objects using json.load.
- Reading and writing JSON data to and from files.
- Enabling lightweight and human-readable data storage and exchange.

In this project, the json module was used within the user authentication system of the graphical user interface to manage the persistent storage of user account data. User credentials and names were saved to a local JSON file using json.dump and loaded back at application startup using json.load, enabling a simple and lightweight login and registration system without the need for an external database [35].

3.5 Graphical User Interfaces

This section presents the design of the system's graphical user interfaces, which were developed using the CustomTkinter library. These interfaces are intended to facilitate image upload, display classification results, and enhance the overall user experience for medical professionals interacting with breast cancer classification system. The system comprises distinct interfaces, each serving a specific functional role.

3.5.1 Welcome Screen

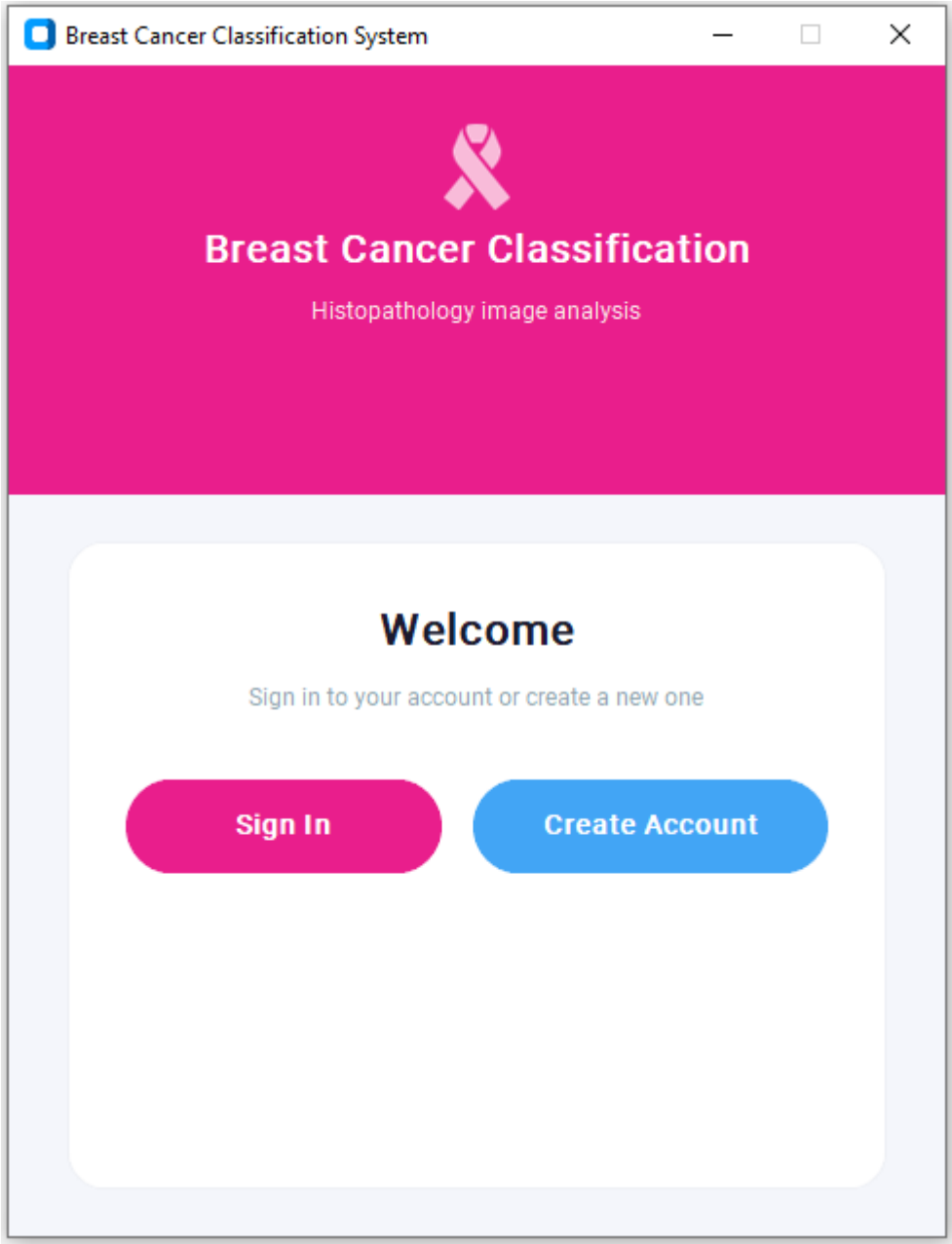

Fig. 3.1 Entry Screen

When the application is launched, the user is presented with the Welcome Screen, which displays the system name and logo. The screen provides two options: signing into an existing account or creating a new one.

3.5.2 Login Page

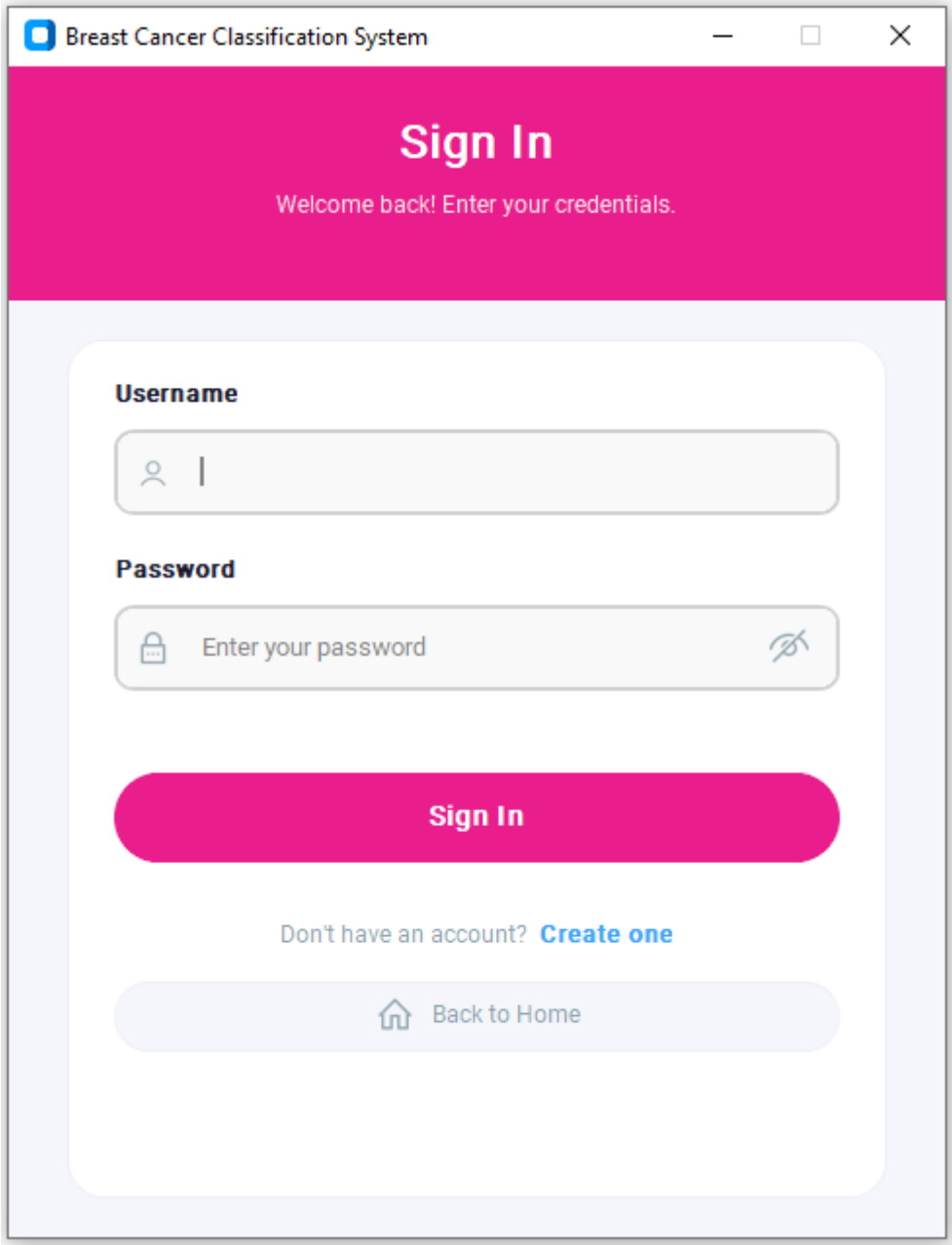

Fig. 3.2 Login Page

The Login page requires the user to enter a valid username and password to access the system. If the credentials are incorrect, an error message is displayed. The page also includes a link to the Registration page for new users and a button to return to the Welcome Screen.

3.5.3 Registration Page

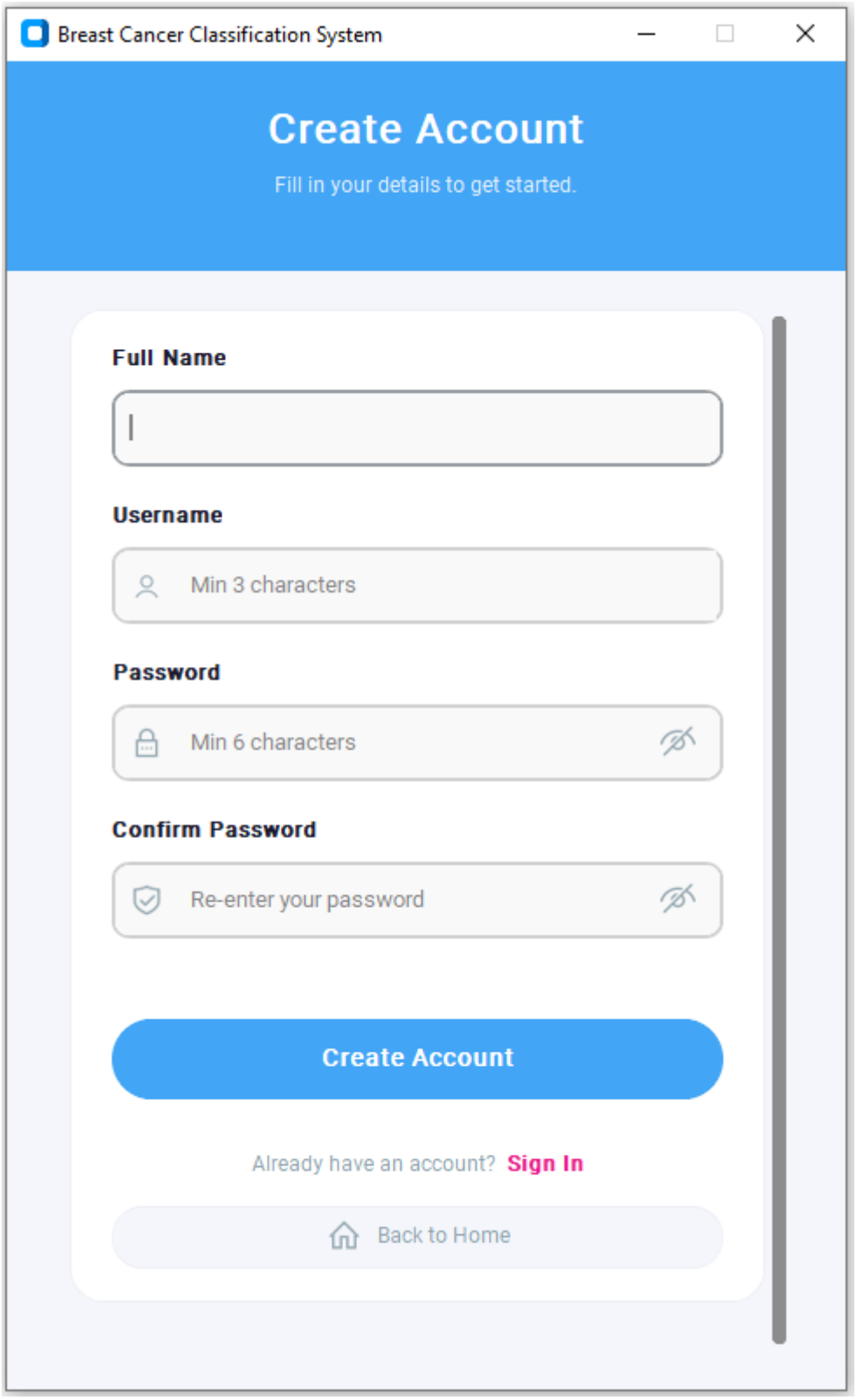

Fig. 3.3 Registration Page

The Registration page allows new users to create an account by entering their full name, username, password, and password confirmation. The system validates all fields (and displays an error message if any condition is not met). Upon successful registration, the user is redirected to the Login page.

3.5.4 Welcome Page

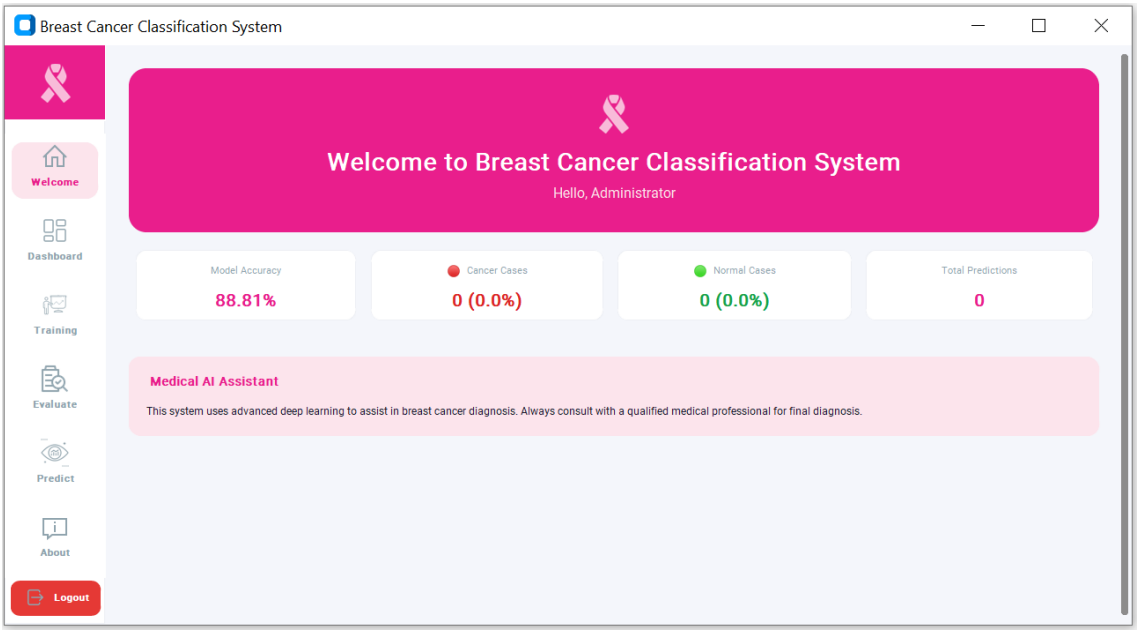

Fig. 3.4 Welcome Page

After login, the user is directed to the Welcome Page, which displays a personalised greeting along with four summary cards showing the model's test accuracy, the number of cancer predictions, the number of normal predictions, and the total predictions made in the current session.

3.5.5 Dashboard Page

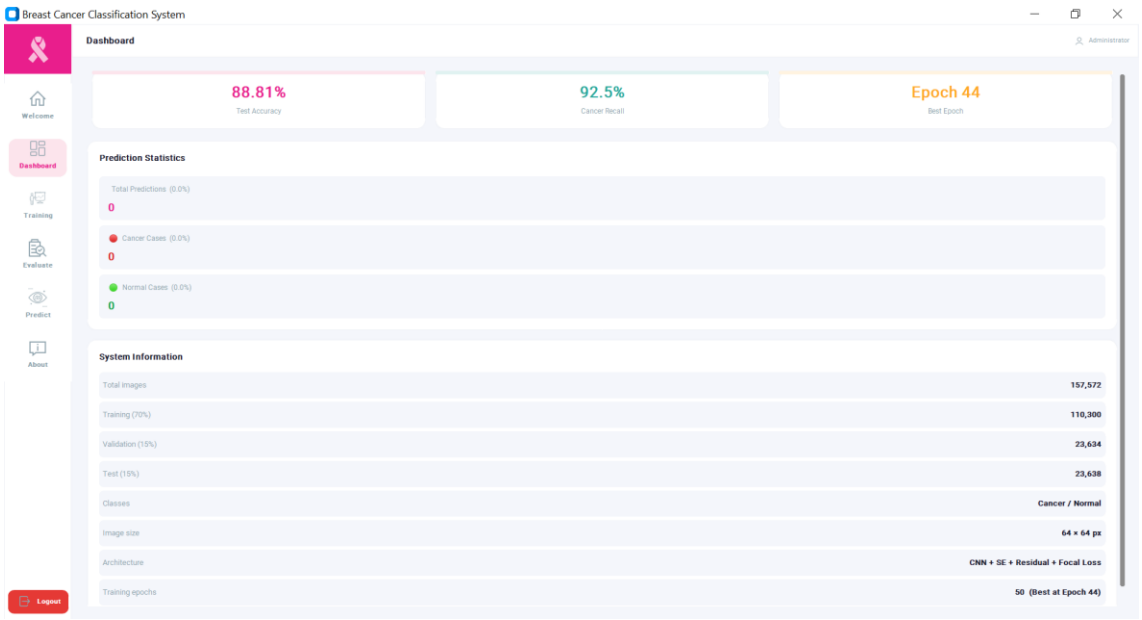

Fig. 3.5 Dashboard Page

The Dashboard page provides a summary of the model's performance metrics, including test accuracy, cancer recall, and the best training epoch. It also displays the session's prediction statistics and a system information section containing details about the dataset, image resolution, and model architecture.

3.5.6 Training Page

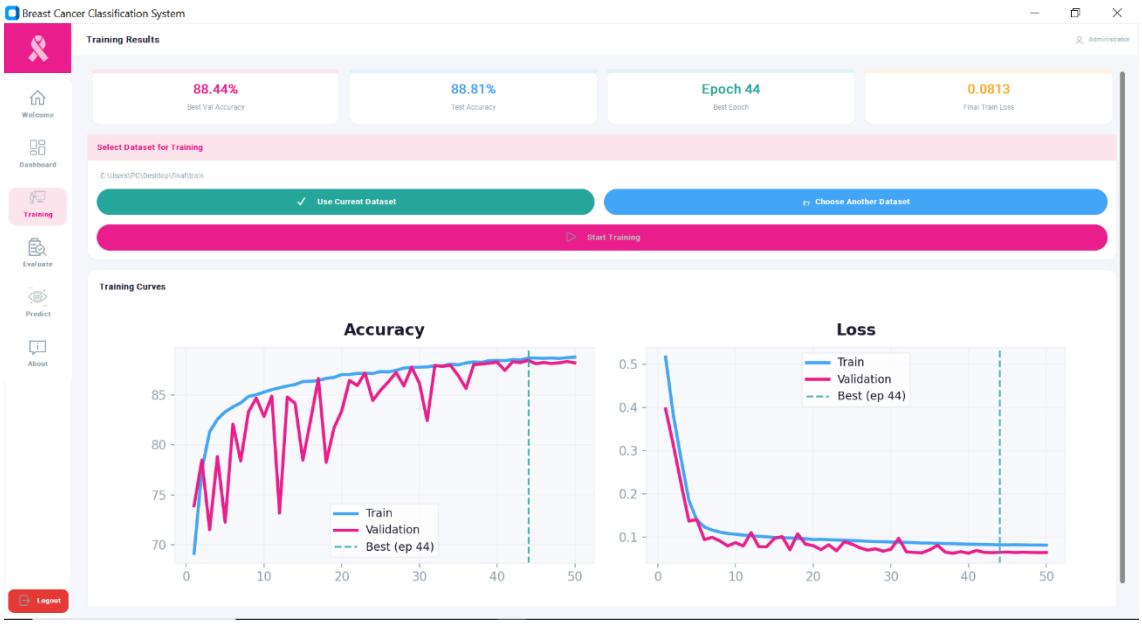

Fig. 3.6 Training Page

The Training page allows the user to select a dataset and launch the model training process. Once training is complete, it displays the best validation accuracy, test accuracy, best epoch, and final training loss, along with the accuracy and loss curves plotted over all training epochs.

3.5.7 Evaluation Page

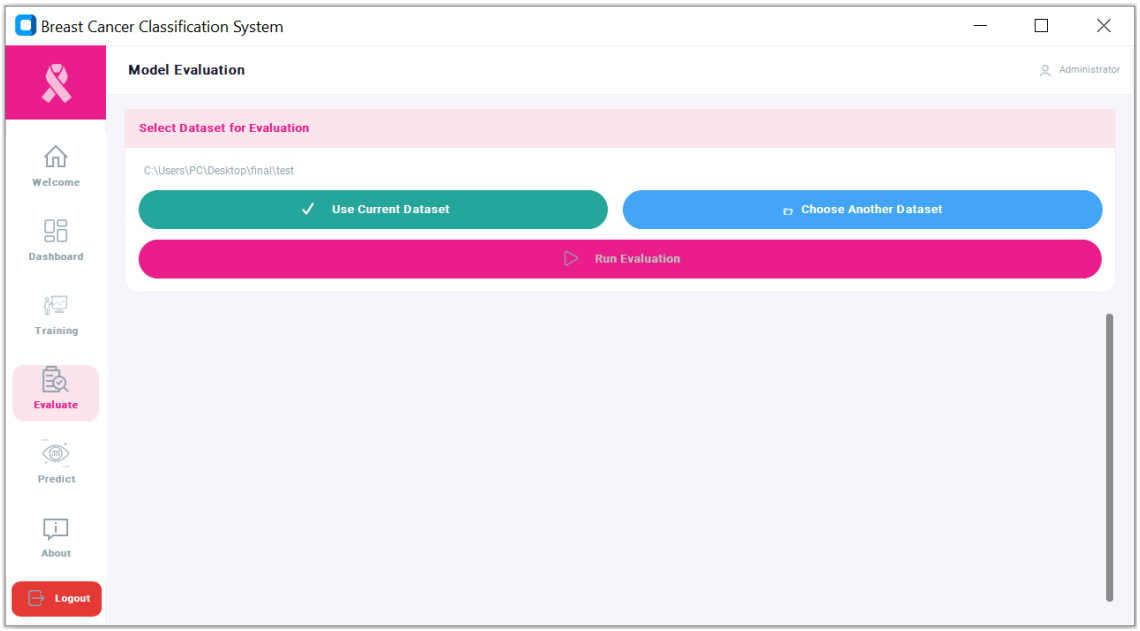

Fig. 3.7 Evaluation Page - Before Running Evaluation

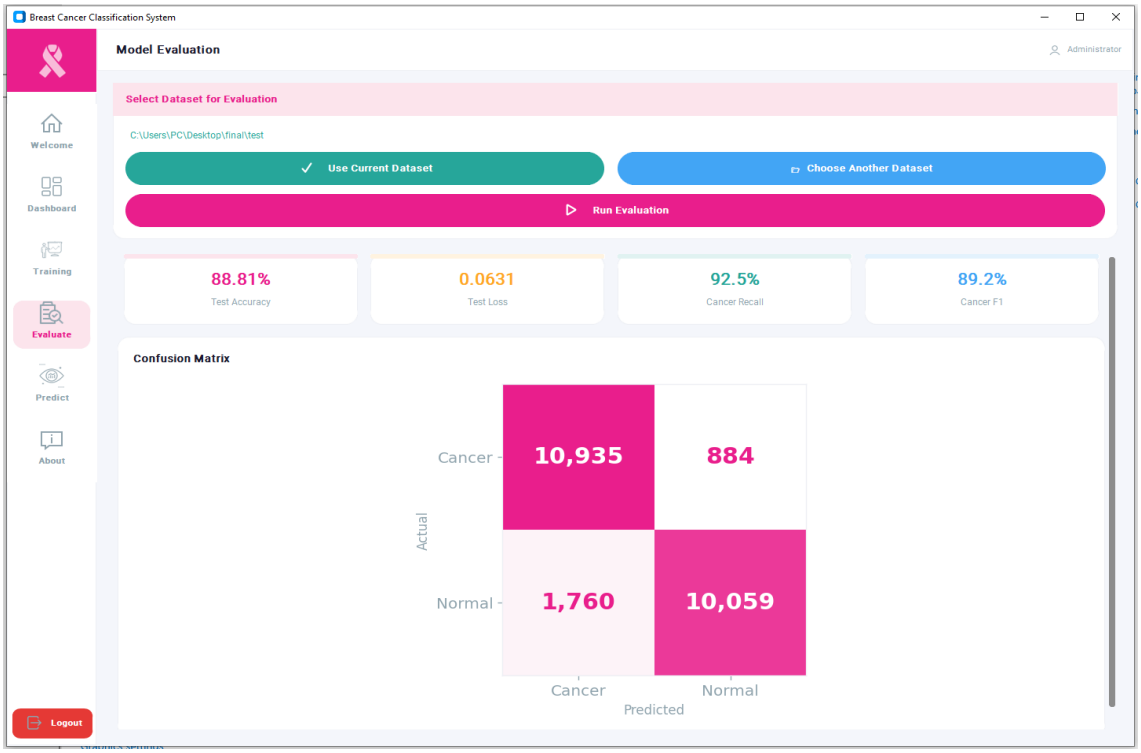

Fig. 3.8 Evaluation Page - After Running Evaluation

The Evaluation page allows the user to test the model on a selected dataset by clicking the "Run Evaluation" button. After evaluation, the system displays the test accuracy, test loss, cancer recall, and cancer F1-score, along with a confusion matrix showing the classification results for each class.

3.5.8 Prediction Page

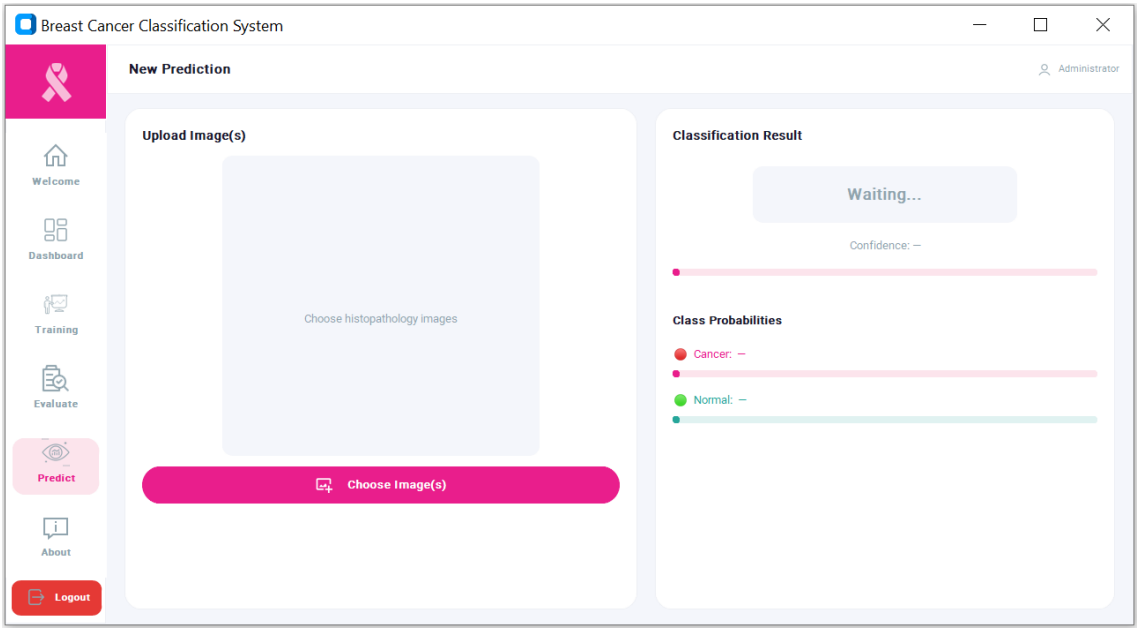

Fig. 3.9 Prediction Page -Before Prediction

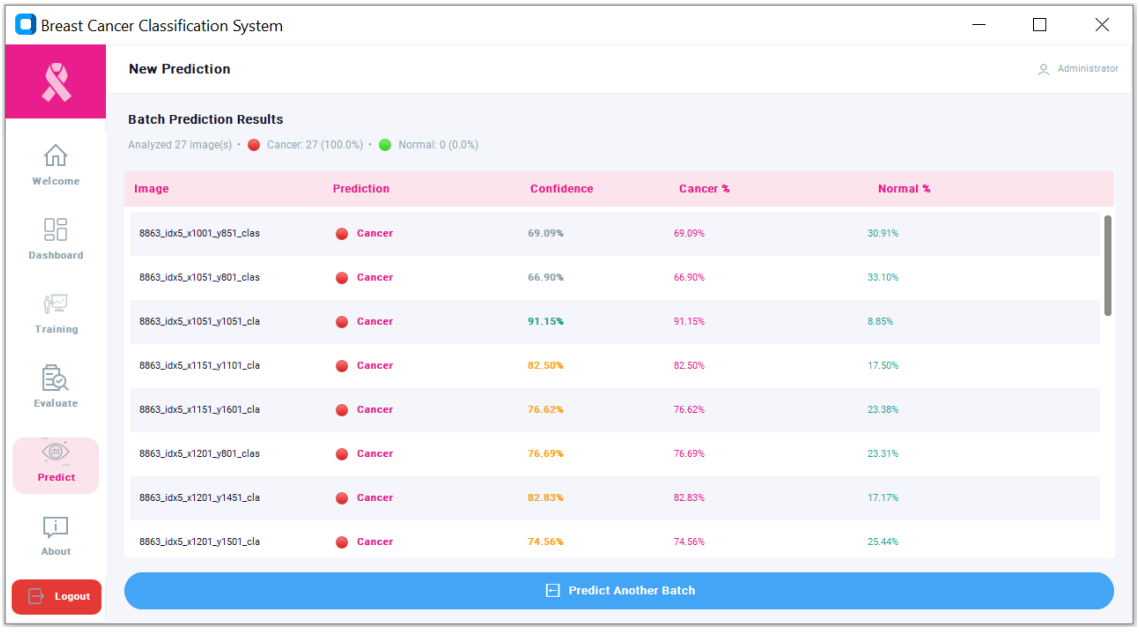

Fig. 3.10 Prediction Page -Batch Prediction Results

The Prediction page is where the user submits histopathological images for classification. The user selects one or more images using the file browser. For a single image, the predicted class, confidence score, and class probabilities are displayed. For multiple images, the results are presented in a table listing the filename, predicted class, confidence, and class probabilities for each image.

3.5.9 About Page

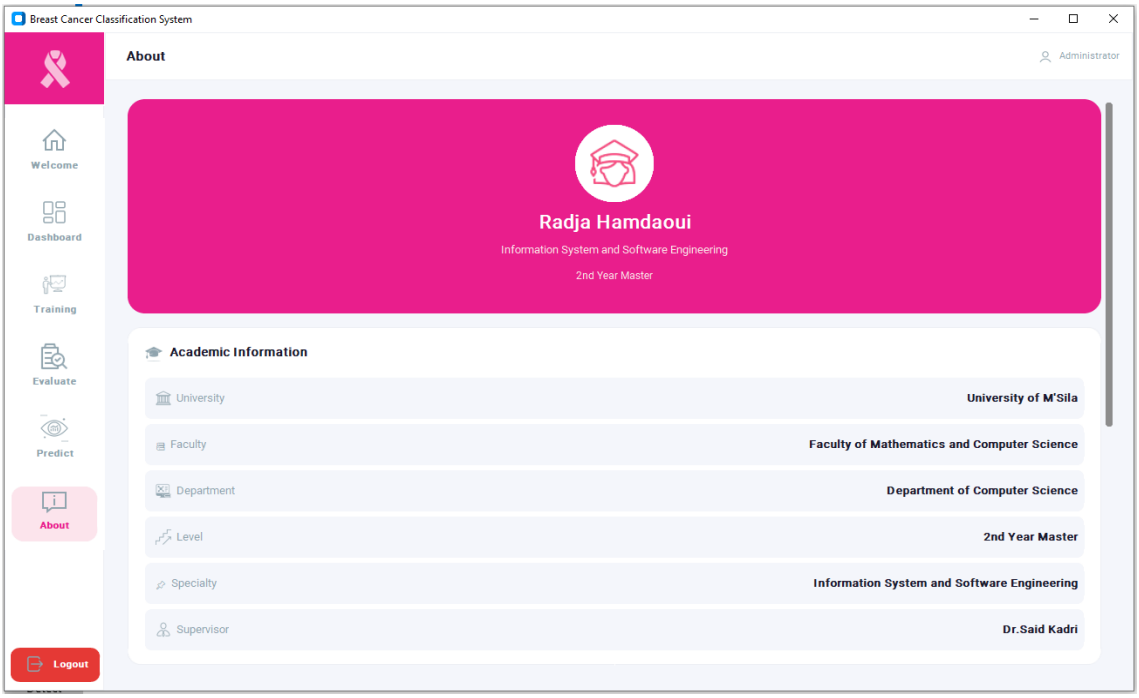

Fig. 3.11 About Page -Developer and Academic Information

Fig. 3.12 About Page -Project Information

The About page presents the developer's academic information, including the university, faculty, department, specialty, and supervisor name. It also provides a brief description of the project and a summary of its key technical characteristics such as the model architecture, test accuracy, cancer recall, and dataset size.

3.6 Conclusion

This chapter covered the implementation details of the breast cancer classification system, starting with the programming language and development environment, followed by the hardware specifications used during training and evaluation. It then described the programming libraries employed in both the training pipeline and the graphical user interface, highlighting the specific role of each library within the system. Finally, it presented the graphical user interface, describing each page and its functionality within the overall system.

Future Work Perspectives

The results achieved in this work are encouraging, yet several directions remain open for future investigation and improvement.

Regarding data diversity, the current model was trained and evaluated solely on the publicly available IDC dataset. Future work should incorporate multi-institutional datasets to assess generalizability across different imaging conditions and patient demographics, with particular relevance given to data from Algerian clinical institutions.

From the perspective of AI technologies, more advanced architectures such as Vision Transformers or hybrid CNN-Transformer models could further improve classification performance. The integration of multimodal approaches combining histopathological images with genomic or clinical data could also enable more comprehensive and personalized risk assessment.

In terms of explainability and clinical trust, the incorporation of interpretability tools such as Grad-CAM would allow the system to visually highlight the tissue regions driving each prediction, a necessary step toward responsible deployment in clinical practice.

Finally, prospective clinical validation in collaboration with oncology and pathology departments, alongside optimization for GPU-accelerated or cloud-based inference, remain critical steps toward real-world deployment of the proposed system.

General Conclusion

The present thesis has addressed the challenge of early breast cancer detection through the design and implementation of an automated system based on artificial intelligence technologies. Guided by the dual objective of achieving high classification accuracy and reducing the clinical risk of missed cancer diagnoses, the work has followed a rigorous and structured methodology from the literature review to the final system deployment.

This thesis covered the scientific and clinical foundation of breast cancer detection, the complete methodology of the proposed deep learning system, and its full technical implementation within a graphical user interface designed for medical professionals.

The proposed system is built around a custom Convolutional Neural Network integrating Residual connections for stable gradient flow, Squeeze-and-Excitation attention blocks for channel-wise feature recalibration, Focal Loss to focus training on hard misclassified samples, and a cancer-boosted class weighting strategy to minimize missed cancer detections. Trained from scratch without relying on any pretrained weights, the proposed model outperformed all evaluated architectures, achieving a test accuracy of 88.81%, a cancer recall of 92.5%, a macro F1-score of 0.89, and a False Negative count of 884 the lowest among all compared models confirming both its classification strength and its clinical relevance.

During the development of this thesis, several challenges were encountered. One of the main difficulties was the class imbalance present in the original dataset, which could bias the model toward the majority class and negatively affect cancer detection performance. Another challenge was achieving a balance between overall accuracy and clinical sensitivity, particularly reducing False Negatives, as missed cancer detections represent the most critical type of error in medical diagnosis. Furthermore, multiple deep learning architectures and training strategies were evaluated and compared before selecting the final model. Additional challenges included preventing overfitting, optimizing hyperparameters, and managing the computational limitations encountered during experiments with more complex architectures. Despite these difficulties, the proposed approach successfully achieved robust and clinically relevant performance.

In summary, this thesis has demonstrated that AI technologies, and deep learning in particular, constitute a promising and effective approach to the early detection of breast

cancer from histopathological images. The proposed system provides a solid, reproducible, and transparent foundation for further research and clinical development in this field.

References

[1] International Agency for Research on Cancer (IARC), "Global Cancer Observatory (GCO)," 2024. Accessed on: May, 23, 2026. [Online]. Available: https://gco.iarc.who.int/en

[2] World Health Organization, "Breast cancer," Fact Sheets, Apr. 16, 2026. Accessed on: Apr, 25, 2026. [Online]. Available: https://www.who.int/news-room/fact-sheets/detail/breast-cancer

[3] National Cancer Institute, "Breast Cancer Screening," Dec. 2, 2025. Accessed on: Apr, 17, 2026. [Online]. Available: https://www.cancer.gov/types/breast/screening

[4] American Cancer Society, "Breast Ultrasound," Jan. 14, 2022. Accessed on: Apr, 18, 2026. [Online]. Available: https://www.cancer.org/cancer/types/breast-cancer/screening-tests-and-early-detection/breast-ultrasound.html

[5] Mayo Clinic, "Breast biopsy," Tests & Procedures, Aug. 25, 2023. Accessed on: Apr, 18, 2026. [Online]. Available: https://www.mayoclinic.org/tests-procedures/breast-biopsy/about/pac-20384812

[6] W. Yue et al., "Machine Learning with Applications in Breast Cancer Diagnosis and Prognosis," Designs, vol. 2, no. 2, Art. no. 13, 2018.

[7] M. Ghorbian and S. Ghorbian, "Usefulness of machine learning and deep learning approaches in screening and early detection of breast cancer," Heliyon, vol. 9, no. 12, Art. no. e22427, Nov. 2023.

[8] A. Khalid et al., "Breast Cancer Detection and Prevention Using Machine Learning," Diagnostics, vol. 13, no. 19, Art. no. 3113, 2023.

[9] S. A. Alanazi et al., "Boosting Breast Cancer Detection Using Convolutional Neural Network," Journal of Healthcare Engineering, vol. 2021, Art. ID 5528622, 2021.

[10] M. Nasser and U. K. Yusof, "Deep Learning Based Methods for Breast Cancer Diagnosis: A Systematic Review and Future Direction," Diagnostics, vol. 13, no. 1, Art. no. 161, 2023.

[11] D. Petríková and I. Cimrák, "Convolutional Neural Network in Histopathology," Encyclopedia, May 3, 2023. Accessed on: Apr, 24, 2026. [Online]. Available: https://encyclopedia.pub/entry/43691

[12] A. Q. Khan et al., "Advances in breast cancer diagnosis: a comprehensive review of imaging, biosensors, and emerging wearable technologies," Frontiers in Oncology, vol. 15, 2025.

[13] W. Voon et al., "Performance analysis of seven Convolutional Neural Networks (CNNs) with transfer learning for Invasive Ductal Carcinoma (IDC) grading in breast histopathological images," Scientific Reports, vol. 12, Art. no. 19200, 2022.

[14] N. Houssami and K. Hunter, "The epidemiology, radiology and biological characteristics of interval breast cancers in population mammography screening," npj Breast Cancer, vol. 3, Art. no. 12, 2017.

[15] J. Sandbank et al., "Validation and real-world clinical application of an artificial intelligence algorithm for breast cancer detection in biopsies," npj Breast Cancer, vol. 8, Art. no. 129, 2022.

[16] A. Ali et al., "Exploring AI Approaches for Breast Cancer Detection and Diagnosis: A Review Article," Breast Cancer: Targets and Therapy, vol. 17, pp. 927–947, 2025.

[17] F. A. Spanhol et al., "A Dataset for Breast Cancer Histopathological Image Classification," IEEE Trans. Biomed. Eng., vol. 63, no. 7, pp. 1455–1462, 2016.

[18] Z. Rehman et al., "Classification of benign and malignant subtypes of breast cancer histopathology imaging using hybrid CNN-LSTM based transfer learning," BMC Medical Imaging, vol. 23, Art. no. 964, 2023.

[19] L. A. Aldakhil et al., "Attention-Based Deep Learning Approach for Breast Cancer Histopathological Image Multi-Classification," Diagnostics, vol. 14, no. 13, Art. no. 1402, 2024.

[20] Python Software Foundation, "Python." Accessed on: May, 15, 2026. [Online]. Available: https://www.python.org

[21] Microsoft, "Visual Studio Code." Accessed on: May, 15, 2026. [Online]. Available: https://code.visualstudio.com

[22] Google, "TensorFlow." Accessed on: May, 15, 2026. [Online]. Available: https://www.tensorflow.org

[23] Keras Team, "Keras." Accessed on: May, 15, 2026. [Online]. Available: https://keras.io

[24] NumPy Developers, "NumPy." Accessed on: May, 15, 2026. [Online]. Available: https://numpy.org

[25] Matplotlib Development Team, "Matplotlib." Accessed on: May, 16, 2026. [Online]. Available: https://matplotlib.org

[26] M. Waskom, "Seaborn." Accessed on: May, 16, 2026. [Online]. Available: https://seaborn.pydata.org

[27] Scikit-learn Developers, "Scikit-learn." Accessed on: May, 16, 2026. [Online]. Available: https://scikit-learn.org

[28] Pillow Developers, "Pillow Documentation." Accessed on: May, 16, 2026. [Online]. Available: https://pillow.readthedocs.io

[29] T. Schimansky, "CustomTkinter." Accessed on: May, 16, 2026. [Online]. Available: https://customtkinter.tomschimansky.com

[30] Python Software Foundation, "os — Miscellaneous operating system interfaces," Python 3 Documentation. Accessed on: May, 17, 2026. [Online]. Available: https://docs.python.org/3/library/os.html

[31] Python Software Foundation, "datetime — Basic date and time types," Python 3 Documentation. Accessed on: May, 17, 2026. [Online]. Available: https://docs.python.org/3/library/datetime.html

[32] Python Software Foundation, "threading — Thread-based parallelism," Python 3 Documentation. Accessed on: May, 17, 2026. [Online]. Available: https://docs.python.org/3/library/threading.html

[33] Python Software Foundation, "tkinter.filedialog — File and directory selection dialogs," Python 3 Documentation. Accessed on: May, 17, 2026. [Online]. Available: https://docs.python.org/3/library/tkinter.filedialog.html

[34] Python Software Foundation, "io — Core tools for working with streams," Python 3 Documentation. Accessed on: May, 19, 2026. [Online]. Available: https://docs.python.org/3/library/dialog.html

[35] Python Software Foundation, "json — JSON encoder and decoder," Python 3 Documentation. Accessed on: May, 19, 2026. [Online]. Available: https://docs.python.org/3/library/json.html

[36] P. Mooney, "Breast Histopathology Images," Kaggle, 2017. Accessed on: Mar. 2, 2026. [Online]. Available:

https://www.kaggle.com/datasets/paultimothymooney/breast-histopathology-images