Deep Learning Breast Cancer Detection

Final Year Project

Isaac Cilia Attard

A project report presented for the degree of
Bachelor of Science in Computer Science

Department of Computing
University of London
UK
August 13, 2025

Deep Learning Breast Cancer Detection

Final Year Project

Isaac Cilia Attard

Abstract

This project explores the application of convolutional neural networks (CNNs), and other architectures for detecting breast cancer using the Digital Database for Screening Mammography (DDSM) dataset. The devised system will incorporate data preprocessing, model training, and explainability through Grad-CAM heatmaps. The study evaluates the model's feasibility for clinical integration, indicating the potential difficulties in adopting such systems like class imbalance, and computational constraints. Future work will involve model optimisation, more advanced architecures, and enhanced data augmentation to aid in generalisability, and usability.

Contents

1 Introduction

Breast cancer remains as one of the leading causes of cancer mortality amongst women around the globe. Routine screening processes can prompt early detection of cancer, and significantly improve patient outcomes, yet they are time-consuming, and success rates vary depending on the observer. Across the world, many hospitals are taking part in large-scale research programmes to investigate the feasibility of using deep learning systems, like convolutional neural networks (CNNs), to match or exceed the diagnostic acumen of human radiologists in breast cancer screening.

Upon the premise of this initiative, project 3.2 titled —Deep Learning Breast Cancer Detection, explores the potential of applying CNN-based models to publicly available datasets such as the Digital Database for Screening Mammography (DDSM), as well as other technologies like vision transformers, and transfer learning. The objective of this project is two-fold: to develop a working proof-of-concept that achieves statistical significance in its results on the DDSM dataset, and to evaluate the integration of such a prototype into existing clinical workflows, identifying potential challenges and limitations.

This introduction begins by stating the clinical, and technical motivation for this project, then presents research questions, and objectives, as well as an outline of the report structure.

1.1 User Requirements

The main users of such a system would be radiologists, and clinicians within a clinical setting such as the NHS, particularly involved in breast cancer screening. Radiologists require a system that can provide accurate malignancy predictions which prioritises high-risk cases to reduce the workload, and offers explainable outputs (like through Grad-CAM heatmaps) to foster trust in the model's decisions. Clinicians require a system that can integrate seamlessly into existing systems, and supports rapid inference time, whilst minimising false positives that can lead to patient anxiety, and unnecessary follow-ups. As indirect stakeholders, patients would benefit from a system that can improve the accuracy of breast cancer screening, leading to earlier detection, and treatment, ultimately improving patient outcomes.

1.2 Stakeholder Requirements

This project aligns with the NHS' 2025 initiative to explore AI-assisted mammoraphy systems, that aim to replace of of two radiologists in the screening process (as detailed in the project brief). Stakeholders, therefore, include NHS radiologists, hospital IT departments (for integrating the system into existing workflows), and regulatory bodies (like the MHRA for clinical approval). The system must comply with medical device regulations, whilst ensuring data privacy in accordance with

GDPR regulations. It must also be computational feasible on hospital hardware through GPU-enabled servers. By understanding these needs, the system aims to reduce diagnostic, and turnaround times, alleviate radiologist workload, and improve patient outcomes through more accurate breast cancer screening.

1.3 Motivation

Mammography is widely known as the de facto standard for early breast cancer screening. Despite this, the visual similarity between benign, and malignant lesions can challenge even the most experienced of radiologists, leading to skewed results, and misdiagnosis. Often, false positives lead to unnecessary follow-ups, and false negatives result in delayed treatment, which can prove deadly. Deep learning can significantly reduce these issues by learning from large collections of labelled scans, complementing radiologists, and improving the accuracy of mammography.

Convolutional neural networks (CNNs), can be used to automatically extract features from an image, including edges, and complex textures, that can prove demonstrably effective at weeding out subtle patterns in pathology. Various studies have demonstrated the prowess of CNNs in tasks like tumour segmentation, and classification on medical imaging modalities like CT, and MRI scans, however, applications to mammography remain relatively untested, in part due to the challenges like standardisation, and generalisation.

The DDSM dataset is a large-scale collection of digital mammograms, annotated by radiologists for the presence or absence of malignant lesions. It is widely used in research, allowing for the effective benchmarking on model performance against established studies, and comparing CNN outputs to ground-truth labels. In addition, the use of the DDSM ensures that the prototype is developed on a dataset that is representative of real-world clinical data, and is reproducible, allowing for further extensions, and collaborations.

Aside from technical performance, an automated system must be understood in how it could augment clinical workflows. Thus, a CNN that is reliable enough, could be used as an assistant to radiologists, prioritising high-risk cases for human review, and reducing the burden of radiologist workload, and turnaround times. On the other hand, understanding how the system oculd fail, like misclassification issues, could guide refinements in data preprocessing pipelines, model architecture, and evaluation metrics.

1.4 Research Questions, and Objectives

This project is comprised of the following, central research questions.

1. **Model Feasibility** — Can a CNN trained on the DDSM dataset achieve classification metrics (like AUC, sensitivity, specificity) that approach or exceed those for human readers under similar conditions?

2. **Prototype Evaluation** — What do preliminary results demonstrate about the feasibility of integrating a CNN-based system into clinical workflows? How might these insights inform augmentations to model architecture or data augmentation methods to improve the system?

3. **Clinical Integration** — What are some of the practical considerations, like inference time, interpretability, and failures that would affect the adoption of such deep learning systems in a clinical setting?

In a bid to answer these questions, the following will be performed.

- Develop a CNN architecture using TensorFlow, and Keras, optimised for binary classification of patches, using practices like dataset splitting, transfer learning, and regularisation.
- Train the model on the DDSM dataset, and report performance on a test partition using metrics like accuracy, sensitivity, specificity, and AUC to determine diagnostic accuracy.
- Analyse the model's performance to identify the categories of issues where performance degrades, and identify mitigation strategies.
- Reflect on integration into clinical settings, discussing how such a system helps to reduce radiologist workload in reading mammograms, and error rates.

Note: Due to time-constraints, the previously mentioned Django web application was omitted. Instead, the focus is on developing a proper convolutional neural network that is able to accurately discriminate between benign and malignant lesions in mammograms, and demonstrate it's decisions through the Grad-CAM framework.

1.5 Report Structure

- **Chapter 1** — Introduction — This chapter introduces the motivation for the project, and outlines the research questions, and objectives.
- **Chapter 2** — Literature Review — This chapter reviews the literature on breast cancer screening, and deep learning applications to mammography.
- **Chapter 3** — Design — This chapter describes the design of the CNN architecture, and the data preprocessing pipeline.
- **Chapter 4** — Implementation — This chapter describes the manner in which the system was implemented, including a description of the model implemented, and the way in which it was trained.
- **Chapter 5** — Evaluation — This chapter recites the evaluation pipelines, and the attained scores from the model defined in the previous chapter.
- **Chapter 6** — Conclusion — The final chapter summarises the findings of the project, and discusses the implications of the results, as well as future work.

Note: This project borrows inspiration from *https://github.com/dustoff06/BreastCancers*.

2 Background

Breast cancer screening may use multiple imaging modalities, including X-ray mammography, digital ultrasound, and hispathological microscopy, each offering different diagnostic information. Deep convolutional neural networks (CNNs) have proven to be remarkably effective across these modalities, by learning complex image features from a given dataset [Jia+24] [Car+24]. Widely-used mammography datasets include the Digital Database for Screening Mammography (DDSM) [Car+24] and the Curated Breast Imaging Subset of Digital Database for Screening Mammography (CBIS-DDSM) containing around 10,000 images [Car+24], and the INbreast dataset containing around 400 FFDM (full-field digital mammography) images [Car+24]. For histopathology, the BreakHis dataset, containing about 9,000 breast tissue images at 40x-400x magnification, and BACH (Breast Cancer Histology) dataset are commonly used as benchmarks [Jia+24] [Sri+23]. For ultrasound imaging, the BUSI (Breast Ultrasound Image) dataset, has also been used in CNN studies [Lat+24]. Such public repositories have enabled various CNN-based breast cancer detection and classification studies.

2.1 History

Computer-aided detection (CAD) systems for breast cancer originated in the 1980s with the advent of digital mammography, marking the beginning of using computer systems to assist radiologists. By the mid-1980s, major strides were made in developing algorithms for use in mammography (and also chest X-rays) [GCB08]. These early CAD systems used hand-engineered image processing, and statistical classifiers to detection calcifications or masses. In the 1990s, the first studies involving neural networks for CAD appeared. These involved the application of the first convolutional neural network to mammographic microcalcification detection [GCB08]. In subsequent years, the use of more traditional machine learning (ML) methods like support vector machines (SVMs), and decision trees were also applied to breast pathology, and imaging, using handcrafted features like texture, shape, and histogram statistics to classify regions as benign or malignant. Nowadays, CNNs are dominant in both mammography, and histopathology tasks, without using classical pipelines that extract features for a separate classifier. The use of CNNs allows for the direct learning of features from pixels, allowing for better accuracy, and generalisation [Ara+17]. Namely, studies have shown that CNN-extracted features used in an SVM achieved a 95.6% sensitivity on histology slides [Ara+17]. In general, CNN-based models outperform traditional feature-based SVMs or random forest classifiers across many breast image datasets, though SVMs, and trees are still used for certain tasks such as radiomics-based subtype classification [Guo+24]. The disadvantages of using CNNs usually include the necessity of large, labelled datasets, and heavy computational resources for their training, and inference. This however comes at the advantage of automated feature learning, and the identification of sub-

tle, complex patterns that classical machine learning models struggle to come to terms with. On the other hand, SVMs and decision trees can work with smaller datasets, but depend on domain-expert feature design, and may not generalise as well to unseen data [Ara+17].

2.2 Convolutional Neural Network Architectures, and Methods

Modern CNN-based architectures for breast cancer detection are typically reliant on established architectures (like ResNet, Inception, VGG, DenseNet, and EfficientNet) designed for medical imaging. Transfer learning is a common approach, where pre-trained models are fine-tuned on medical scans to compensate for limited annotated data [Sri+23]. This approach has been shown to outperform using CNNs as fixed feature extractors. Data augmentation like flips, rotations, colour jittering, and GAN-based augmentation is also commonly used, particularly in ultrasound, and pathology where datasets are limited [Lat+24]. Namely, Gupta et al. state that the random flipping, and rotation of ultrasound images, helps to overcome class imbalance on the BUSI dataset [Lat+24]. Explainability methods like Grad-CAM heatmaps are incorporated to visualise salient regions, and build trust in the model [Lat+24] [GJ24]. In segmentation tasks (like in cases of lesions), encoder-decoder networks like U-Net remain the industry standard, yielding critical performance on medical image segmentation tasks [Jia+24].

2.3 Mammography-based detection

CNNs have demonstrated a high diagnostic accuracy by classifying whole images or localised patches of mammograms. Studies often use the DDSM or INbreast dataset, and the Digital Breast Tomosynthesis (DBT) dataset. Namely, a commercial deep CNN-based system, trained on about 1,000 mammograms reached an AUC score of 0.82, comparable to an expert radiologist [Car+24]. Retrospective evaluations indicate the Computer Aided Diagnosis (CAD) systems powered by AI can improve radiologist sensitivity, such that their cancer detection rate rose from 51% to 62% after the introduction of AI aid as evidenced by Watanabe et al. [Car+24]. In a similar study, Kim et al. reports that an unassisted AI-CAD system achieved an AUC of 0.94, outperforming the radiologist's AUC of 0.88 [Car+24]. Akselrod-Ballin et al. achieved an AUC of 0.91 using a large linked dataset of mammograms, and patient records [Car+24]. In large screening challenges (with around 85,000 training images, and 68,000 external images), top CNN ensembles reached an AUC of 0.90 on held-out mammograms [Car+24]. Although still below expert performance, combining radiologists with AI yielded an AUC of 0.94, indicating the performance gains experienced when the two are synergised. CNNs have also been applied to lesion detection, and segmentation, with a particular case in which a U-Net-like model, trained on a CBIS-DDSM dataset, detected masses with 95.7% sensitivity, and a Dice score of 74.5% [Jia+24].

Popular CNN models for mammography include ResNet50, VGG16/19, Inception, and EfficientNet. For tasks like the detection of mass classification or microcalcification detection, such networks often outperform more traditional, feature-based

methods [Wan24] [Car+24]. Namely, Shen et al. found a fine-tuned CNN achieved an 88% accuracy whilst classifying mammographic tumourss, outperforming radiologists which achieved 83% accuracy [Wan24]. In another study, Yala et al. reports CNN risk models with an AUC of 0.84 against an AUC of 0.77 for radiologists [Wan24]. Object detection models like Faster-RCNN, and RetinaNet were also used within the Agarwal et al. study, which achieved a 99% accuracy for malignant-mass detection on the INbreast dataset, when applying Faster-RCNN to full-field mammograms [Jia+24]. More recent works, use transformer-based models using a pretrained vision transformer, and segmentation, which reaches a 99.96% accuracy on INbreast, by first extracting a mass ROI [Kum+24]. All in all, CNNs on mammograms achieve a high AUC value, often similar to or exceeding the performance of an average radiologist [Car+24].

2.4 Histopathology-based detection

CNNs have also been applied to histopathology slides for cancer classification. The BreakHis dataset (containing microscopic tissue patches at 40x-400x magnification) has been the de facto benchmark for this. Basic CNNs like ResNet, and VGG achieve more than 90% accuracy on binary classification tasks upon the BreakHis dataset (when classifying benign against malignant cancers) [Jia+24]. In Han et al. a CNN was applied to the BreakHis dataset, achieving a 93.2% accuracy [Jia+24]. More complex architectures can improve performance significantly more, such as in Munikoti et al. which combines a CNN with an LSTM (utilising ImageNet transfer learning), and achieving a 99% accuracy on binary classification in the BreakHis dataset [Sri+23]. Such deep learning models significantly outperform more traditional machine learning methods like Rao et al. which found that CNNs (with transfer learning) superseded SVMs or random forests on histology data [Yus+23]. Apart from patch classification, CNNs like U-net are used to segment nuclei or tumor regions to help analysis [Jia+24]. Modern approaches use attention or multiscale to capture complex tissue contexts [Sri+23]. In conclusion, CNNs on histopathology often achieve very high accuracies of 90-99% on curated datasets [Sri+23] [Yus+23], taking advantage of deep architectures, and large-scale patch augmentation.

2.5 Ultrasound-based detection

When breast tissue is dense, ultrasound imaging is often used as an imaging modality. The use of this, posits issues relating to speckle noise, and operator variability. CNNs have shown remarkable results despite this. Since public ultrasound datasets are smaller, transfer learning from networks like VGG-16, VGG-19, and AlexNet are often used. Wang et al. applied a multi-view CNN with ImageNet pretraining to a breast ultrasound, and achieved an AUC of 0.9468 [Wan24]. In another study, Gupta et al. fine-tuned EfficientNet-B7 on the BUSI dataset, and reported a 99.1% accuracy [Lat+24]. Augmentation methods like rotations, and colour jitter were used, and Grad-CAM explanation was used to focus on lesion features [Lat+24]. In addition, Rahman et al. deduced that a fine-tuned model of InceptionV3 outperformed VGG-19 for ultrasound tumour classification [ADC21]. In general, ultrasound CNNs commonly exceed 90% accuracy, with transfer learning, and data

augmentation necessary to compensate for limited sample sizes.

2.6 Transfer Learning, Data Augmentation, and Explainability

Since annotated breast imaging data is limited, transfer learning, and augmentation are essential. Almost all recent studies initialise CNNs with ImageNet weights, and fine-tune on medical images [Sri+23]. Fine-tuning routinely outperforms training from scratch or using CNNs as pure feature extractors. Strategies for augmentation, including geometric transforms, colour perturbations, and GAN-based synthesis help to ease overfitting problems. Namely, EfficientNet-based classifiers for ultrasound used random flipping, rotation, and colour jitter to improve minority malignant cases [Lat+24]. Other histology studies have also used flips, rotations, and staining augmentation to improve the diversity of tissue patches [Sri+23].

Explainable AI is becoming increasingly important in medical imaging, where methods such as heatmaps (like Grad-CAM, and saliency maps) are applied to highlight image regions to make predictions [Lat+24] [GJ24]. This helps to verify that CNNs focus on tumours rather than artifacts. Studies report overlaying class activation maps on mammographs or ultrasound to show ROI localisation. Namely, EfficientNet-B7 visualised Grad-CAM heatmaps on ultrasound to confirm lesion focus [Lat+24]. Explainability is also explored via feature attribution, and partial dependence, but Grad-CAM remains popular [Lat+24] [GJ24].

2.7 Performance vs. Traditional Methods, and Radiologists

CNN-based systems have been shown to outperform classical machine learning algorithms, and approximate expert-level analysis. Deep models have reduced false positives relative to older CAD systems, and improved sensitivity. On mammography tasks, CNNs matched or superseded radiologist sensitivity as demonstrated by Becker et al., which reported an equal AUC of 0.82 for a commercial CNN, and human experts [Car+24]. In larger trials, deep learning models achieved an AUC of 0.86 to 0.94, while experts scored slightly better, but, when combining AI with human experts, recall rates were consistently reduced [Car+24]. In ultrasound, CNNs have outperformed conventional classifiers like SVMs, and decision trees when given enough data [Yus+23]. In histopathology, CNNs significantly outperform custom feature methods, where hybrid CNN and TL models reached an accuracy of 97% on BreakHis, and classical machine learning models rarely exceeded 90% on the same task [Yus+23]. Thus, studies indicate that CNNs offer excellent sensitivity, and specificity, across different modalities [Car+24] [Wan24], demonstrating their practical utility within CAD systems.

3 Design

Breast cancer is a leading cause of death amongst women worldwide, and early detection is paramount to improve the chances of survival. Computer-aided detection (CAD) systems have long been used to assist in screening, and diagnosis workflows for long, but recent developments in convolutional neural networks (CNNs) have significantly boosted the rates of detection on natural image benchmarks. The aim of this project is to leverage state-of-the-art CNN architectures to build a decision support system for breast cancer detection in mammograms. The Digital Database for Screening Mammography (DDSM) will be used to train, and evaluate a CNN model, and the results will be used to indicate the presence of malignant tumours. Thus, the primary objective is to develop, and evaluate a CNN, that matches or exceeds the single-reader accuracy commonly achieved by radiologists in large-scale screening programmes.

3.1 Domain Context, and User Analysis

Mammography screening programs capture images of the breast, and use them to detect early signs of breast cancer. Radiologists are responsible for viewing multiple images per week. In this context, false negatives lead to delayed diagnoses that may pose serious health consequences, whilst false positives impose unnecessary stress on patients, and the system itself, with procedures like biopsies, and follow-up imaging creating an operational burden. Therefore, any AI-powered decision support system must prioritise very high sensitivity while maintaining a reasonable specificity to avoid recalls.

The primary users of this system will be professional radiologists, who will utilise suggestions from it, and clinicians who monitor screening quality, and throughput values. Radiologists will require not only malignancy scores, but also explainable outputs demonstrating where the model has detected abnormalities for validation purposes, and to manage trust issues in the system. Many published works lack sufficient end-to-end system design, and evaluation. By concentrating on both model performance, and system usability, this project fills a gap in showing a deployable prototype complete with explainable outputs, and inference benchmarks.

3.2 System Architecture, and Design

The resultant system will be comprised of the following components: data preprocessing, model training, and evaluation, and explainability.

- **Data Preprocessing** — The DDSM dataset will be appropriately preprocessed to ensure that the images are in a suitable format for the model. This

will include resizing, normalisation, and augmentation of the images to improve the model's performance. Thus, horizontal flipping, rotations, and contrast adjustments may be applied on the fly to increase the diversity of the training data, and reduce overfitting. The dataset will also be split into training, validation, and test sets to ensure equal class balance across malignant, and benign classes.

- **Model Training, and Evaluation** — Multiple CNN models will be trained on the DDSM dataset, being constructed with TensorFlow, and Keras. The model will then be evaluated using metrics like accuracy, precision, recall, and F1-score.

- **Explainability** — Grad-CAM will be used to overlay heatmaps on input mammograms, demonstrating various regions that drive the model's prediction.

3.3 Technologies, and Methods

In light of the previous components, mature, and well-supported libraries were selected to ensure that the system is easy to maintain, and extend.

- **Data Handling** — Python as the fundamental language of choice for the entire stack, NumPy, and Pandas for data manipulation tasks, and OpenCV for image processing tasks.

- **Deep Learning** — TensorFlow for neural networks, with Keras frontend for ease of use, and fast prototyping.

- **Hyperparameter Tuning** — Optuna for automated, and efficient hyperparameter tuning, and model selection.

- **Explainability** — Grad-CAM through tf-keras-vis for visualising the model's attention on the input image, and highlighting the areas of interest.

3.4 Model

The core of the system is a deep feature extractor powered by a convolutional neural network (CNN). Multiple methods will be explored, including alternative architectures like vision transformers (ViTs), but initially, a custom CNN will be trialled. Alternative bases like VGG16, and ResNet50 will also be evaluated. This provides proven performance on breast image tasks, and manageable complexity [FS25]. VGG16's uniform 16-layer architecture adapts well via transfer learning from ImageNet, with prior work achieving a high accuracy on the BreakHis histopathology dataset by fine-tuning VGG16 [FS25]. Additionally, VGG16 has shown excellent generalisation on unbalanced data [FS25], making it suitable for the DDSM dataset's class imbalance.

3.5 Clinical Integration

The system could be integrated into existing radiology workflows, providing a decision support tool that assists radiologists in identifying potential malignancies. The model's outputs, including the malignancy score and Grad-CAM heatmaps, may be presented in the form of a user interface that is accessed by clinicians, and radiologists. Additionally, the system could log the model's predictions, and the corresponding images, to create a database of cases that can be used for further research, and development. This would be facilitated by the use of a two-tier architecture web application, with a RESTful API for the model, and a frontend for user interaction. Requests to the API will return the model's predictions, and Grad-CAM heatmaps, which can then be displayed in the frontend. As for the logging requests, a relational database such as PostgreSQL could be used to store the model's predictions, and the corresponding images, along with metadata such as the date, time, and user ID. This would allow for easy retrieval of cases for further analysis, and research. To set up the APIs, a web framework like FastAPI could be used, which is designed for building APIs quickly, and efficiently. FastAPI is built on top of Starlette, and Pydantic, and provides automatic generation of OpenAPI documentation, which can be used to document the API endpoints, and their parameters. The frontend could be built using a modern JavaScript framework like Next.js, which would allow for a responsive, and interactive user interface. The frontend would communicate with the backend API to retrieve the model's predictions, and Grad-CAM heatmaps, which can then be displayed to the user. Support for asynchronous requests, and real-time updates could be implemented which would allow for a more responsive user experience, and makes the system able to scale. The frontend could also include features such as user authentication, and authorisation, to ensure that only authorised users can access the system, and view logs. As for GPU acceleration, the model could be deployed on a cloud platform such as Google Cloud Platform (GCP), or Amazon Web Services (AWS), which provides CUDA-comptatible GPU instances that can be used to accelerate model inference. This would allow for faster processing of images, and reduce the time taken to generate predictions. Additionally, the cloud platform could be used to store the model's weights, and configuration files, which can be easily accessed by the frontend, and backend APIs. In this pursuit, a series of MLOps pipelines could be developed in order to facilitate the training, and deployment of models, and reduce the time taken in between prototyping, and production releases. Finally, horizontal scaling could be deployed through Kubernetes, which would allow for the system to handle a large number of requests, and scale up or down based on demand. However, the latter would mostly be important in cases with very high throughput that would likely not be seen within a single, or even a few hospitals. This would probably be of significance should the system be deployed as a form of SaaS, where multiple hospitals around the globe make use of the service simultaneously. In this case, the system may also need to use message queues, such as RabbitMQ, or Kafka, and be split into microservices to handle the high volume of requests, and ensure that the system remains responsive.

The above is a mock system architecture that incorporates some of the elements described previously. It offers a simplified version of the system that would provide an adequate starting point for the project, and can be extended as needed.

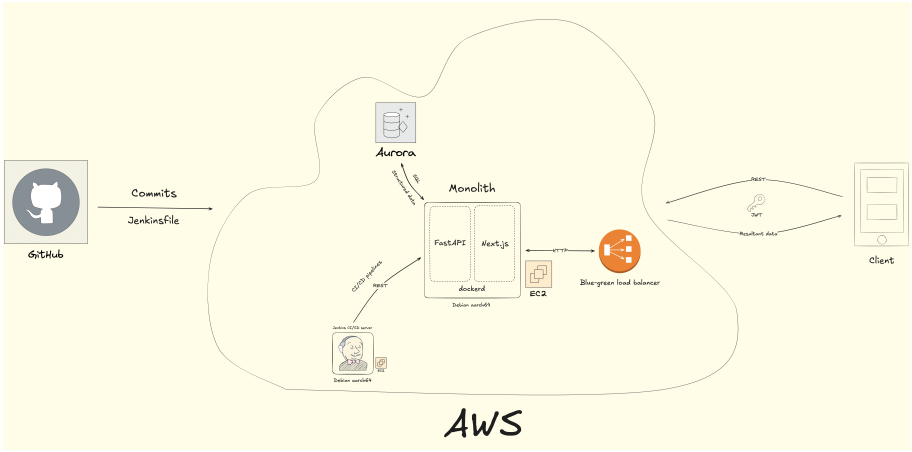

Figure 3.1: Mock System Architecture

CI/CD pipelines are even included through the use of Jenkins, and GitHub where the `jenkinsfile` is hosted within the project's repoistories. Blue-green load balancing is also introduced to manage API versions within requests, and JWT for authentication. Amazon Aurora is used as a database, which is PostgreSQL compatible.

3.6 Plan of Evaluation

The project as a whole will be evaluated using a variety of metrics. For the model itself, the most critical part of the system, the standard evaluation metrics of accuracy, precision, recall, and F1-score will be used to assess its performance on the DDSM dataset, including the Area Under the Receiver Operating Characteristic Curve (AUC). Together with Grad-CAM heatmaps, which will show the model's attention on the input images, these metrics will provide a comprehensive outlook on the model's classification performance. The evaluation will focus on the model's performance, and the explainability of the outputs.

4 Implementation

This chapter outlines the technical implementation of the breast cancer detection system, with a focus on data handling, model architecture, training methodology, and explainability. The system was implemented using TensorFlow, and Keras, running atop the Python interpreter. The data ingestion, and preprocessing steps will be covered alongside the architecture, and training of the convolutional neural networks used. Strategies to improve generalisation will be detailed, as well as the explainability module, and capturing, and visualising the results. Snippets of code are shown to demonstrate key points, with the full listings available in the appendix.

4.1 Setup

The necessary libraries are first imported, and the necessary helper functions are defined at the beginning of the notebook, to be used throughout. Namely, a garbage collection function is defined to clear as much memory as possible to prevent the system from crashing. This is called throughout the notebook. The function is called first to set the stage for the memory to be occupied solely by the subroutines in the following stages of the implementation.

4.2 Data Manipulation

This section focuses on loading, and preparing data for training, and testing from TFRecord files. A random seed is firstly set so that results are reproducibile. Due to the limitations of the hardware utilised for this project, a small subset of the DDSM dataset, sourced from Kaggle, was utilised, with pre-augmented images. The dataset was provided in the form of TensorFlow's TFRecord format, which is a binary format that allows for efficient storage and retrieval of large datasets. Each TFRecord file contains three fields, `image`, `label`, and `label_normal`. The `image` field contains the raw, flattened image data, the `label` field contains the label for the image in the form of a binary integer class (0 for benign, and 1 for malignant classifications), and the `label_normal` field contains a normalised version of the label for direct use in training.

Two empty lists are first initialised for images, and labels with a dictionary defined that specifies the structure of TFRecord data. A function is then defined to process the TFRecord files, and store the results into the previously defined `images`, and `labels` lists.

```
def read_file(files):
    mydata = (
        tf.data.TFRecordDataset(files, num_parallel_reads=4)
        .shuffle(buffer_size=10000)
        .cache()
    )
```

```
    mydata = mydata.map(
        lambda x: tf.io.parse_example(x, features), num_parallel_calls
    =4
    )
    for image_features in mydata:
        image = tf.io.decode_raw(image_features["image"], tf.uint8)
        image = tf.reshape(image, [299, 299])
        image = np.asarray(image)
        images.append(image)
        labels.append(image_features["label"].numpy())
```

Listing 4.1: Data ingestion from TFRecord files.

The garbage collection function is invoked before the files are parsed, to avoid any memory issues. The target TFRecord files are defined in a list, prior to being processed. The parsing function is finally called on the previously defined list.

4.2.1 Preparing Data for Training

This section will discuss the cells that process the already loaded training data to make it more understandable to the CNN later in the notebook. Each image is first saved as a `.jpg` file in the `train` directory with enumerated filenames. The list of labels is saved as a CSV file within the same directory. The labels are then loaded, and converted to boolean format, ideal for the binary classification task the CNN is to perform. The list of `.jpg` files from the `train` directory is retrieving using `glob`, with the numeric part of the filename extracted, and storing in the `mysplit` list for further sorting. A Pandas DataFrame is defined for training, with columns to store the image paths, and binary label, all sorted by their numeric value.

4.2.2 Preparing Data for Testing

This section details methods similar to those seen within the previous section, but more suited towards preparing testing data. The following cell begins by loading test images, and labels from NumPy files, and concatenates them to form `x_test` for images, and `y_test` for labels. The `y_test` array is then converted to boolean format as was done for training labels. The testing data is then saved as `.jpg` files within the `test` directory with enumerated filenames. The test labels are then loaded from the CSV file, and casted as strings. Similar to the previous section, the filenames are retrieved as a list of `.jpg` files using `glob`, with the numeric part of the filename extracted via `glob`. A Pandas DataFrame is finally created with this information.

4.2.3 Visualisation

The dataset is visualised within this section, with the garbage collection function being called first to alleviate the system's memory as much as possible.

4.2.4 Data Generators

The data processed in the previous sections is finally encapsulated into generators to feed it into the model during training, and testing. An `ImageDataGenerator`

object is first defined to preprocess images by rescaling their pixel values. Each pixel is normalised into the range 0, to 255 by being divided by 255. This is known to improve the model's training efficiency, and performance. The training generator yields batches of preprocessed training images, and their corresponding labels from the `traindf` DataFrame. This feeds the model during training. A similar generator is made for the testing set. As for the ViT model, the training, and testing data is further encapsulated into a HuggingFace dataset.

4.2.5 Data Augmentation

The dataset is already pre-augmented, but further augmentation is applied to the training data for the ViT model to improve its generalisation capabilities. This is done by applying random transformations to the training images, such as horizontal flipping, and rotations. The augmentation is done using the `transforms` function from the `torchvision` library, which allows for easy application of various transformations to the images. The augmentation is defined as follows, and is applied to the training, and testing data before it is fed into the model.

```
augment = transforms.Compose([
    transforms.RandomHorizontalFlip(),
    transforms.RandomRotation(10),
])

def transform(example_batch):
    images = [augment(Image.open(f).convert("RGB")) for f in
    example_batch["file"]]
    inputs = processor(images, return_tensors="pt")
    inputs["labels"] = [int(lbl) for lbl in example_batch["label"]]
    return inputs

prepared_train_ds = train_dataset.with_transform(transform)
prepared_test_ds = test_dataset.with_transform(transform)
```

Listing 4.2: Data augmentation for the ViT model.

4.2.6 Class Imbalance

The dataset utilised appears to be imbalanced with an 87:13 ratio of negative to positive samples. Without intervention, this leads to the model's outputs being skewed in a detrimental manner, where more false samples are likely to be produced. In order to address this, a callback is utilised from the standard TensorFlow API to compute class weights, and apply them during training. This scales the classes according to each one's scarcity in an attempt to offset the class imbalance, and correct the behaviour of the model. The corrections are computed by the following snippet.

```
class_weights = compute_class_weight(
    class_weight='balanced',
    classes=np.unique(y_train),
    y=y_train
)
class_weights = dict(enumerate(class_weights))
```

Listing 4.3: Correcting the class imbalance.

The resultant weights are then passed into the model during training, as an argument to the model's `fit` method.

4.3 Model Definition

Four models are defined, three of which are CNN-based, and another one which is transformer-based. Each CNN is defined with the same classification head, but an altered base. These models are made to handle binary classification tasks, which are to be trained on breast cancer tissue mammograms. The first model is an entirely custom CNN, which is based upon a series of `Conv2D`, `BatchNormalization`, and `MaxPooling2D` layers. For the classifier head, these are then followed by `Dense`, and `GlobalAveragePooling2D` layers, and fine-tuned accordingly.

```
input_tensor = Input(shape=(299, 299, 3))

x = Conv2D(32, (3, 3), activation='relu', padding='same')(input_tensor)
x = BatchNormalization()(x)
x = MaxPooling2D((2, 2))(x)

x = Conv2D(64, (3, 3), activation='relu', padding='same')(x)
x = BatchNormalization()(x)
x = MaxPooling2D((2, 2))(x)

x = Conv2D(128, (3, 3), activation='relu', padding='same')(x)
x = BatchNormalization()(x)
x = MaxPooling2D((2, 2))(x)

x = Conv2D(256, (3, 3), activation='relu', padding='same')(x)
x = BatchNormalization()(x)
x = MaxPooling2D((2, 2))(x)

x = GlobalAveragePooling2D()(x)
output = Dense(1, activation='sigmoid', name='classifier')(x)
```

Listing 4.4: Custom CNN definition.

The second model introduces transfer learning by supplanting the basis of the model, excluding the classifier head, with a VGG16 model. This significantly simplifies the model definition, and yields benefits in performance when classifying samples.

```
vgg16_base = VGG16(weights="imagenet", include_top=False, input_shape
    =(299, 299, 3))
classifier_head = vgg16_base.output
classifier_head = GlobalAveragePooling2D()(classifier_head)
classifier_head = Dense(1, activation="sigmoid", name="classifier")(
    classifier_head)
vgg16_model = Model(vgg16_base.input, classifier_head)
```

Listing 4.5: VGG16 model definition.

Something similar is done with the third model, where ResNet50 is used instead of VGG16.

```
resnet50_base = ResNet50V2(weights="imagenet", include_top=False,
    input_shape=(299, 299, 3))
classifier_head = resnet50_base.output
classifier_head = GlobalAveragePooling2D()(classifier_head)
```

```
classifier_head = Dense(1, activation="sigmoid", name="classifier")(
    classifier_head)
resnet50_model = Model(resnet50_base.input, classifier_head)
resnet50_model.summary()
```

Listing 4.6: ResNet50 model definition.

The fourth model is transformer-based, and uses the `ViT` or Vision Transformer architecture. The foundation model used is the `facebook/deit-tiny-patch16-224` model provided by Facebook, which is a small `ViT` model, and is more suited for the hardware used in this project. The transformer is fine-tuned on the breast cancer dataset. It is also worth noting that this model utilises a PyTorch backend, and is loaded using HuggingFace's `transformers` library. Due to this, the programmatic definitions for training will slightly differ from the other CNN-based models.

```
labels_list = ["Negative", "Positive"]
model = ViTForImageClassification.from_pretrained(
    'facebook/deit-tiny-patch16-224',
    num_labels=len(labels_list),
    id2label={str(i): c for i, c in enumerate(labels_list)},
    label2id={c: str(i) for i, c in enumerate(labels_list)},
    ignore_mismatched_sizes=True
)
```

Listing 4.7: Vision Transformer model definition.

The idea behind defining these models is to compare, and contrast between the different bases, and artchitectures for performance gains from the use of transfer learning, versus using an entirely custom model.

4.3.1 Training the Model

The model will now be trained, based on the previous definition. Garbage collection is called prior to training, to avoid memory issues. Important callbacks are first defined to stop training if parameters like validation loss do not improve, and for the model with best validation accuracy to be saved. The model is compiled with an `adagrad` optimisation function, and binary cross-entropy loss function. The model is finally trained, with data being fed from the previously defined generators.

```
custom_cnn.compile(
    optimizer=Adam(learning_rate=1e-4),
    loss='binary_crossentropy',
    metrics=['accuracy']
)
```

Listing 4.8: Compiling the custom CNN using the Adam optimiser, and a binary cross-entropy loss function.

The following snippet shows the point at which the model is trained, including epoch count, inclusion of callbacks, validation data, and class weights.

```
custom_cnn_history = custom_cnn.fit(
    train_generator,
    steps_per_epoch=len(train_generator),
    epochs=5,
    callbacks=[checkpoint, early],
    validation_data=test_generator,
    class_weight=class_weights
```

```
)
```

Listing 4.9: Model training of the custom CNN.

4.3.2 Evaluating the Model

This section evaluates the previously trained models using standard metrics like accuracy, and loss on the training, and validation sets. These metrics are extracted into variables for later use from the performance of the model across different epochs during training. These are then plotted, to visualise how the model's accuracy, and loss scores improved. Predictions are made on the test set in order to generate a final classification report, and confusion matrix. The previously-made predictions, and actual values are gathered, and fed into `scikit-learn` to generate the confusion matrix. The results are then plotted using `seaborn`. The same values are used to generate the final classification report. The performances of the different models are plotted on the same accuracy, and loss graphs in order to demonstrate a clear distinction of performance between the different models.

4.4 Model Explanation

This final section touches upon XAI or explainable AI, by implementing *Grad-CAM* through the `tf-explain` module. This provides a visual indication of the parts of the image that influenced the classification made of the model, explaining the results achieved. This is done for the first ten images within the testing set, but can easily be done for any other image as well. The following function accepts a path to an image, the model used, and it's final layer, and returns the explained version of the image, and the classification result. A series of images is then passed through the aforementioned function, and plotted into a singular figure. In the above plot, warmer colours indicate higher influence on the model's predicted class, and cooler colours the opposite.

5 Evaluation

This chapter details a comprehensive evaluation of the breast cancer classification models, implemented using transfer learning-based architectures, and including Grad-CAM visualisations for explainability. The objective of this is to critically determine the performance of the models across several metrics such as quantitative accuracy on the test data, robustness, and generalisability to unseen data, as well as interpretability through explainabile AI. A reflection on the strengths, and weaknesses of the prototype is also provided, along with a discussion on the potential for future work. The evaluation was guided by a combination of empirical metrics, and a quantitative analysis.

5.1 Evaluation Methodology

The evaluation methodology employed in this project seeks to answer the following research questions:

- How accurately do the models classify mammogram images?
- How robust and generalisable are the models to unseen data?
- How interpretable are the model predictions, and what insights can be derived from Grad-CAM visualisations?
- What potential improvements can be made for future work?

To answer these questions, the evaluation process involved:

- **Quantitative Analysis** — Assessing the performance of the models using metrics such as accuracy, precision, recall, and F1-score on the test dataset.
- **Robustness and Generalisability** — Evaluating the performance of the models on unseen data to determine its robustness and generalisability.
- **Explainable AI** — Using Grad-CAM visualisations to interpret the predictions of the models, and gain insights into the decision-making process.
- **Reflection** — Critically reflecting on the strengths and weaknesses of the project, and discussing potential improvements for future work.

5.2 Quantitative Results

The models were evaluated on a test dataset of mammogram images, which is distinct from the training dataset in order to measure generalisability more effectively. As discussed previously, the evaluation metrics utilised include accuracy, precision, recall, and F1-score, as well as AUC score.

5.2.1 Custom CNN-based Model

The first model, a Convolutional Neural Network (CNN) with the same classifier head as the other models, but a customised base architecture, yielded solid results. The model achieved an overall accuracy of 0.88, indicating that 88% of the predictions were correct. The precision, and recall scores for the negative class were 0.96, and 0.89 respectively, indicating a strong ability to identify negative cases, while the positive class had lower scores of 0.52 for precision, and 0.74 for recall, suggesting that the model struggled to identify positive cases effectively. The F1-score averaged at 0.77 across both classes, reflecting a balance between precision, and recall. The macro average scores of 0.74 for precision, and 0.82 for recall indicate a moderate performance across both classes, while the weighted averages of 0.90 for precision, and 0.88 for recall reflect the model's overall effectiveness.

	Precision	Recall	F1-score	Support
Negative	0.96	0.89	0.93	13360.00
Positive	0.52	0.74	0.61	2004.00
Accuracy	0.88	0.88	0.88	0.88
Macro Avg.	0.74	0.82	0.77	15364.00
Weighted Avg.	0.90	0.88	0.88	15364.00

Table 5.1: Quantitative results of the custom CNN model.

The confusion matrix reveals that out of the actual negative cases, 11,955 were correctly predicted as negative, while 1,405 were incorrectly predicted as positive. This indicates a low false positive rate. On the other hand, for actual positive cases, 512 were correctly classified as positive, while 1,492 were mistakenly classified as negative, showing a high false negative rate. This suggests that the model has a strong ability to identify negative cases but struggles with positive cases.

Actual	Predicted	
	Negative	Positive
Negative	11955	1405
Positive	512	1492

Table 5.2: Confusion matrix results of the custom CNN model.

The Receiver Operating Characteristic (ROC) curve for this model shows a moderate upward trend from a low false positive rate to a high true positive rate, indicating a reasonable ability to discriminate between positive, and negative classes. The area under the ROC curve (AUC) is 0.88, which indicates a good ability to distinguish between the two classes, but with room for improvement.

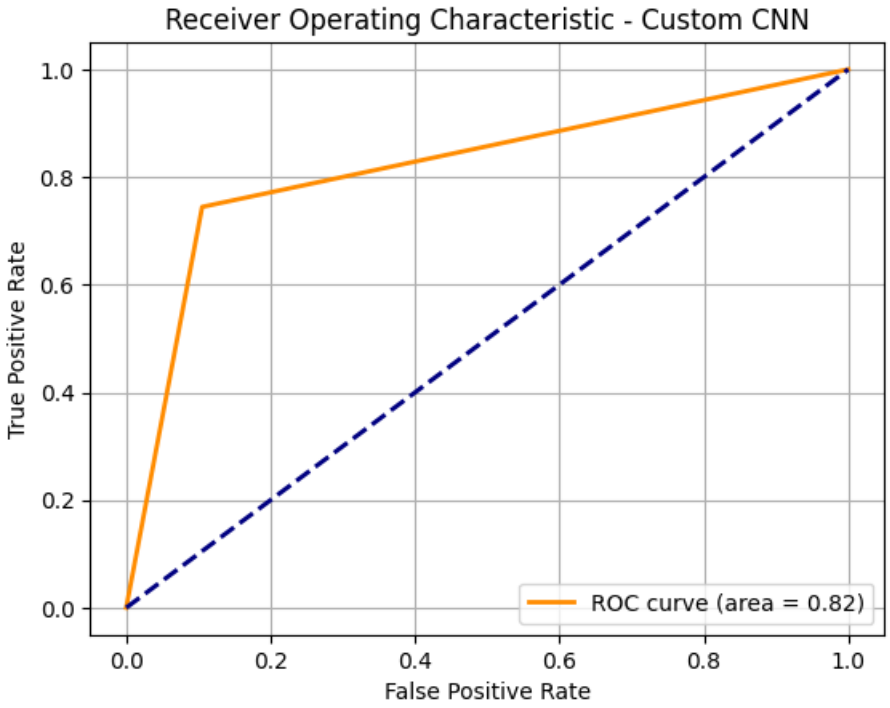

Figure 5.1: ROC curve for the custom CNN model.

5.2.2 VGG16-based Model

The VGG16-based model yielded the following results. These results demonstrate a robust performance with an overall accuracy of 0.96, indicating a 96% success rate in predicting actual outcomes. As for precision, and recall scores, both averaged 0.91, and 0.92 at the macro level, demonstrating a strong balance in identifying positive, and negative cases, with precision reflecting a 91% likelihood that positive predicitons are valid, and recall showing that 92% of actual positives were correctly categorised. The F1-score, also averaging 0.92, denotes the effectiveness of the model by harmonising precision, and recall, highlighting its reliability over both classes. These metrics altogether imply that the model is well-performing, although miniscule variations like higher precision, and recall scores for the negative class compared to those of the positive class indicate potential areas for improvement in handling the positive class more effectively.

	Precision	Recall	F1-score	Support
Negative	0.98	0.96	0.97	13360.00
Positive	0.76	0.90	0.83	2004.00
Accuracy	0.95	0.95	0.95	0.95
Macro Avg.	0.87	0.93	0.90	15364.00
Weighted Avg.	0.96	0.95	0.95	15364.00

Table 5.3: Quantitative results of the VGG16-based model.

The confusion matrix reveals the performance of the classification model, where out of the actual negative cases, 13,055 were correctly predicted negative, while 305 were incorrectly predicted as positive. This indicates a low false positive rate. On the other hand, for actual positive cases, 1,718 cases were correctly labeled as positive, whilst 286 were mistakenly classified as negative, showing a moderate false negative rate. The model shows a strong accuracy in identifying negative cases, with a high true negative rate. This may be owed to the relatively larger number of negative cases within the training dataset. Overall, the confusion matrix indicates a reliable model, with further enhancements to be made to enhance sensitivity to positive cases.

Actual	Predicted	
	Negative	Positive
Negative	12805	555
Positive	198	1806

Table 5.4: Confusion matrix results of the VGG16-based model.

The Receiver Operating Characteristic (ROC) curve for this model demonstrates a significant upward trend from a low false positive rate to a high true positive rate, showing a strong ability for discrimination. The area under the ROC curve also known as AUC is excellent at 0.92, denoting a strong ability to distinguish between positive, and negative classes. Overall, the curve indicates that the model effectively balances between sensitivity, and specificity, implying that the model is reliable for classifying benign from malignant tissue.

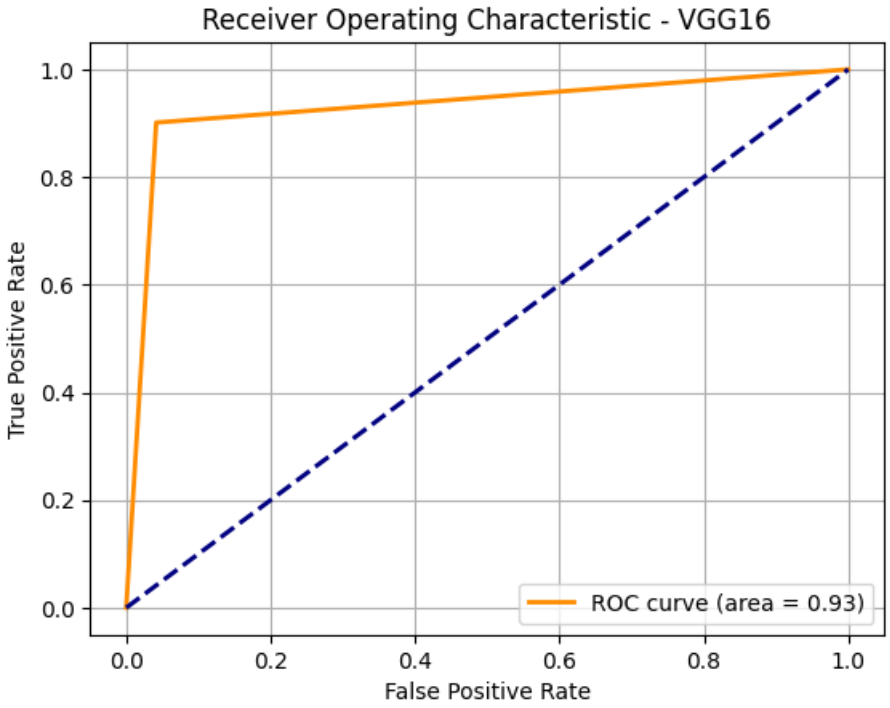

Figure 5.2: ROC curve for the VGG16 model.

5.2.3 ResNet50-based Model

The classification results for the ResNet-based model indicate a highly effective performance, with an overall accuracy of 0.97, implying that 97% of predictions are correct. Both precision, and recall scores for the negative class are robust at 0.98, meaning excellent identification of negative instances supported by a large sample size of 13.360. For the positive class, precision, and recall are slightly lower at 0.90, and 0.89 but are still strong, with a support of 2,004, showing consistent performance across both classes. The macro average of 0.94 for precision, recall, and F1-score demonstrates a balanced model performance, while the weighted average of 0.97 aligns with the overall accuracy of the model, accounting for class imbalance. These results together demonstrate a reliable model with some improvements to be made in positive class predictions.

	Precision	Recall	F1-score	Support
Negative	0.99	0.98	0.98	13360.00
Positive	0.85	0.91	0.88	2004.00
accuracy	0.97	0.97	0.97	0.97
Macro Avg.	0.92	0.94	0.93	15364.00
Weighted Avg.	0.97	0.97	0.97	15364.00

Table 5.5: Quantitative results of the ResNet50-based model.

The confusion matrix results demonstrate that 13,158 images were correctly predicted as negative, with only 202 incorrectly labelled as positive, reflecting a very low false positive rate. In terms of actual positive cases, 1,777 were correctly classified as posisive, while 227 were labelled incorrectly as negative showing a moderate false negative rate. This demonstrates a strongly performing model, with good precision in negative predictions.

Actual	Predicted	
	Negative	Positive
Negative	13040	320
Positive	182	1822

Table 5.6: Confusion matrix results of the ResNet50-based model.

The ROC curve for this model indicates a sharp rise from a low FPR, to a high TPR which shows a robust ability to discriminate. With an AUC score of 0.94, this model exhibits an excellent capability to distinguish between positive, and negative classes, reflecting a high balance of sensitivity, and specificity.

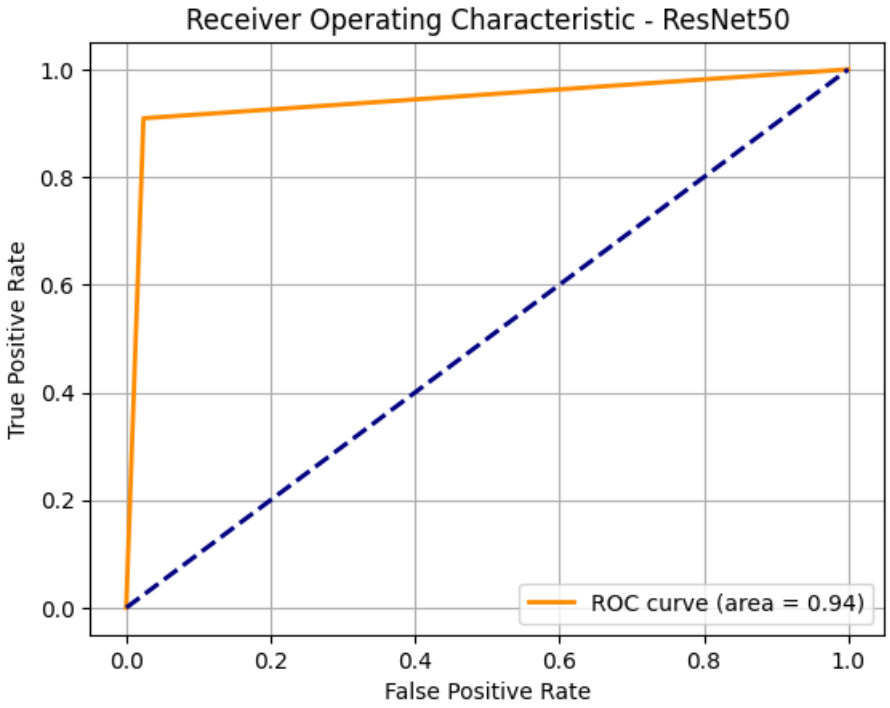

Figure 5.3: ROC curve for the ResNet model.

5.2.4 Vision Transformer Model

The Vision Transformer (ViT) model, which is a more recent architecture, also yielded strong results. The model achieved an overall accuracy of 0.97, indicating a high level of performance in classifying mammogram images. The precision, recall, and F1-score for the negative class were 0.98, 0.99, and 0.98 respectively, demonstrating excellent performance in identifying negative cases. For the positive class, the precision was 0.92, recall was 0.87, and F1-score was 0.89, indicating a solid performance but with room for improvement in identifying positive cases. The macro average scores of 0.95 for precision and 0.93 for recall indicate a balanced performance across both classes, while the weighted averages of 0.97 for all metrics reflect the model's overall effectiveness.

	Precision	Recall	F1-score	Support
Negative	0.98	0.99	0.98	13360.00
Positive	0.92	0.87	0.89	2004.00
accuracy	0.97	0.97	0.97	0.97
Macro Avg.	0.95	0.93	0.94	15364.00
Weighted Avg.	0.97	0.97	0.97	15364.00

Table 5.7: Quantitative results of the ViT-based model.

The confusion matrix for the ViT model shows that 13,214 images were correctly predicted as negative, with only 146 incorrectly labelled as positive, indicating a very low false positive rate. For the actual positive cases, 1,737 were correctly classified as positive, while 267 were labelled incorrectly as negative, showing a moderate false negative rate. This indicates that the model is performing well in identifying negative cases, but there is still room for improvement in identifying positive cases.

Actual	Predicted	
	Negative	Positive
Negative	13214	146
Positive	267	1737

Table 5.8: Confusion matrix results of the ViT-based model.

The ROC curve for the ViT model shows a strong upward trend from a low false positive rate to a high true positive rate, indicating a robust ability to discriminate between positive, and negative classes. The AUC score of 0.93 reflects an excellent capability to distinguish between the two classes, demonstrating a high balance of sensitivity, and specificity.

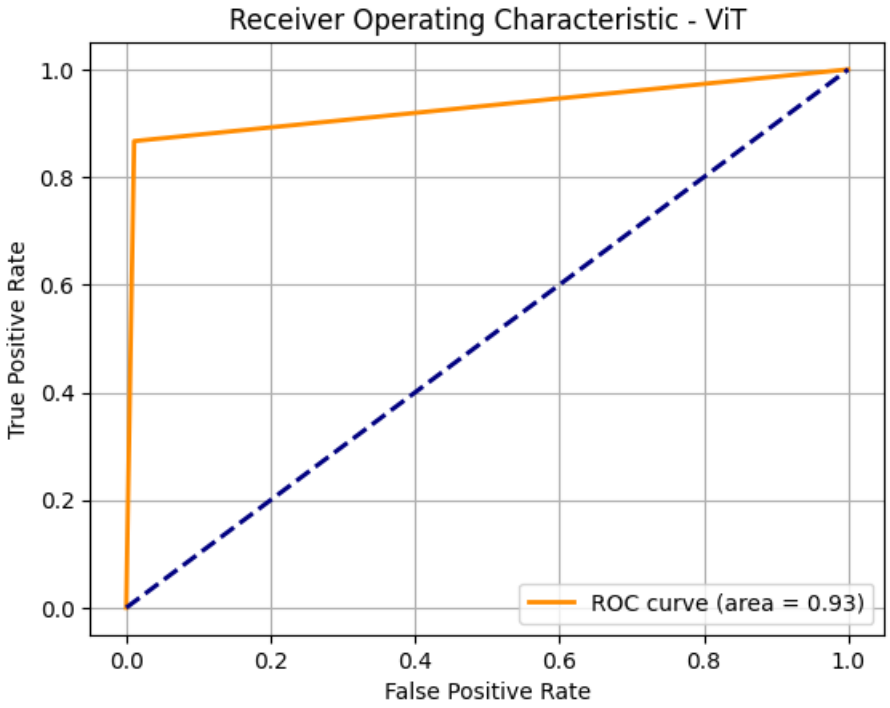

Figure 5.4: ROC curve for the ViT model.

5.3 Comparing the Models

The custom CNN network is considered as a baseline. The quantitative results of the four models indicate that all models perform well, with the ResNet50-based model achieving the highest accuracy of 0.97, with near identical results to the ViT model, followed closely by the VGG16-based model at 0.96, and the custom CNN-based model at 0.88. The precision, recall, and F1-score metrics also reflect strong performance across all models, with the ResNet50-based model showing the best balance between precision, and recall for both classes. The following chart demonstrates the comparison of the models across training epochs in terms of accuracy, and loss scores.

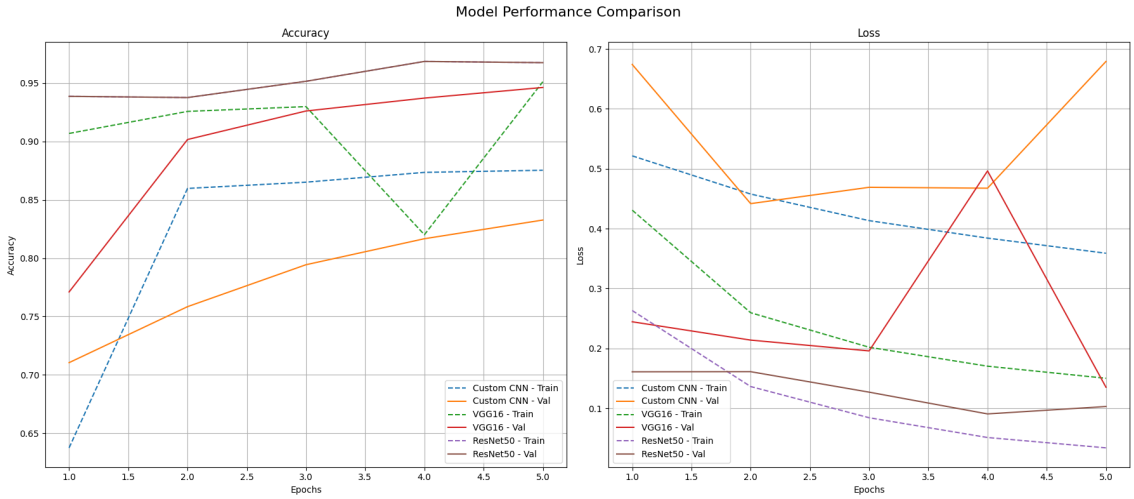

Figure 5.5: Comparison of the CNN models across training epochs in terms of accuracy, and loss scores.

The ROC curves for all models show a strong ability to discriminate between positive, and negative classes, with AUC scores ranging from 0.88 to 0.94, indicating a high balance of sensitivity, and specificity. The confusion matrices further highlight the strengths, and weaknesses of each model, with the ResNet50-based model showing the best performance in terms of true positive, and true negative rates.

5.4 Explainablity

In order to assess the model's performance on predictions aligned with human-interpretable features, a series of images is presented with Grad-CAM overlays. This analysis can be implemented in a clinical system, which would allow radiologists to interpret, and trust decisions made by a CNN-based system. The ability to visually validate the reasoning process behind each decision increases trust in the system, and reduces the opacity of models, and aids in the diagnosis of errors.

Overall, the use of Grad-CAM significantly improves the usability of the models in real-world situations, enabling the effective collaboration between human radiologists, and machines. Namely, in future studies, radiologists could rate the consistency, and usefulness of heatmaps on a Likert scale, or utilise them to flag low-confidence cases for further review. Below are some of the images with Grad-CAM

overlays, demonstrating the model's focus on relevant features in the mammogram images.

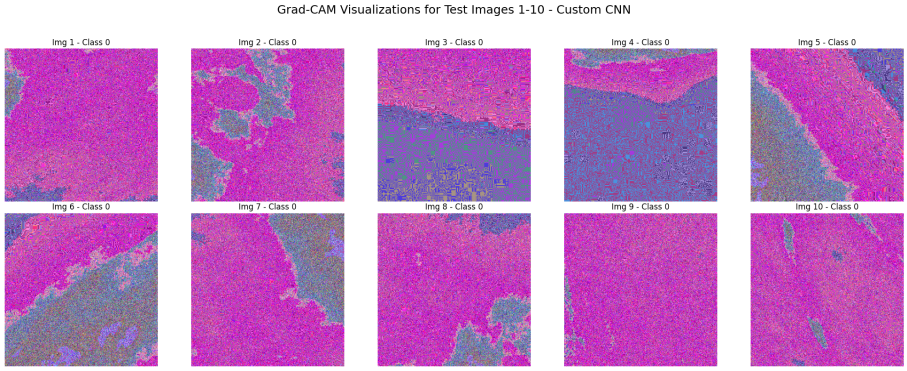

Figure 5.6: Example of Grad-CAM overlay on a mammogram image for the custom CNN.

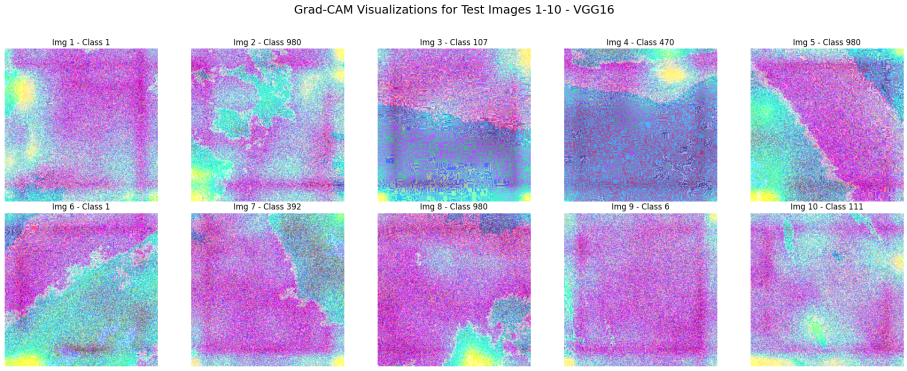

Figure 5.7: Example of Grad-CAM overlay on a mammogram image for the VGG16-based network.

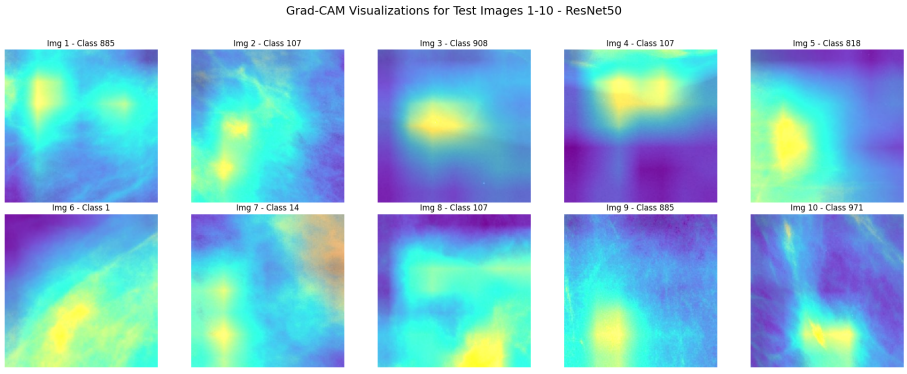

Figure 5.8: Example of Grad-CAM overlay on a mammogram image for the ResNet50-based network.

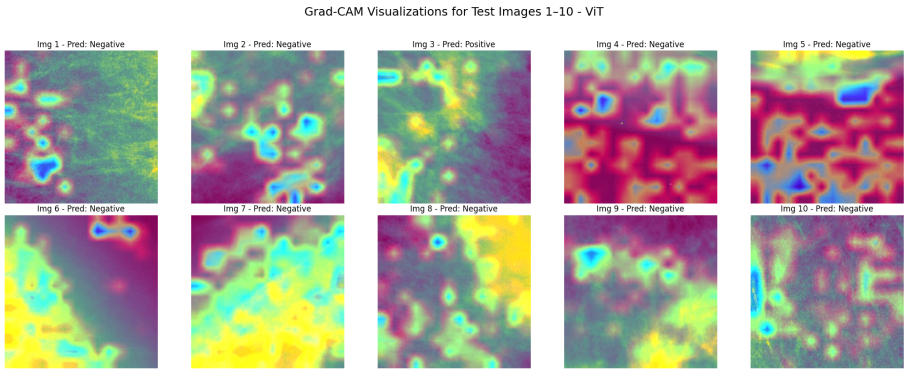

Figure 5.9: Example of Grad-CAM overlay on a mammogram image for the ViT-based network.

5.5 Achievements

This project has overseen the end-to-end development of a raw TFRecord ingestion system, that makes interpretable, and accurate predictions. A reasonably strong generalisation has been achieved using transfer learning on limited data given the AUC scores. Grad-CAM has also been effectively integrated for transparency, with minimal overfitting through the use of dropout, and early stopping.

5.6 Reflection

The project has successfully demonstrated the potential of deep learning-based models for breast cancer classification using mammogram images. The models have shown strong performance across various metrics, and the integration of Grad-CAM has provided valuable insights into the decision-making process of the models. However, there are several areas for improvement, including:

- **Dataset Limitations** — The dataset used for training, and testing the models is relatively small, and imbalanced, which may limit the generalisability of the models. Although this was partially accounted for by computing, and utilising class weights, future work should focus on acquiring larger, and more diverse datasets to improve model performance.
- **Model Complexity** — While the models have shown strong performance, there is potential for further refinement, and optimisation of the architectures to improve accuracy, and reduce overfitting. Most significantly, the transformer-based model could be further optimised by increasing the number of training epochs, and using a larger transformer model with more hyperparameters.
- **Evaluation Methodology** — The evaluation methodology could be enhanced by incorporating cross-validation techniques to provide more robust metrics, and reduce the risk of overfitting.
- **Collaboration with Domain Experts** — Engaging with domain experts such as radiologists can provide valuable insights into the clinical relevance of the model predictions, and help refine the models further.
- **Explainability** — While Grad-CAM has provided valuable insights into the model predictions, further research into explainable AI techniques can enhance the interpretability of the models, and improve trust in their predictions.

These areas for improvement highlight the potential for future work, and the ongoing development of deep learning-based models for breast cancer classification.

5.7 Summary

This evaluation indicates that the proposed deep learning-based system performs adequately on test data, and allows for the useful interpretability via Grad-CAM. While results are promising, the system would not be ready for a production scenario

due to dataset constraints that could be overcome given more powerful hardware to be able to train with it. Nonetheless, a solid foundation has been laid such that further refinement, and more intricate, and advanced loss functions, fine-tuning, and domain expert consultation can be made. In addition, further modifications to the evaluation of the models will be made to include cross-validation, which will give far more accurate metrics than the existing single-fold method of evaluation.

6 Conclusion

This project involves an end-to-end image classification pipeline for distinguishing two classes of mammographic scans using deep learning. Transfer learning is applied, with a pre-trained convolutional neural network being used for the classification of scans. The system ingests data from TFRecord files, originating from the DDSM dataset. These are then preprocessed for the model to use as input. The latter is then trained, and evaluated on this data. This approach allows for the efficient use of pre-trained models, which can significantly reduce the time and resources required for training a model from scratch. In this pursuit, this project has been successful, and may be further extended to meet the exigencies of a clinical setting. The project could be further extended to include additional features given enough time, and resources. What was achieved in this project was limited by the time constraints, and the resources available.

6.1 Future Work

This project demonstrates a solid proof-of-concept, and a foundation for future work. The final system will require further refinement, and optimisation, as well as additional features, and functionality to make it more user-friendly, and robust.

- **Model Optimisation** — The model can be further optimised by adjusting its architecture, and increasing the number of epochs. Additionally, a larger transformer model could be used with sufficient hardware. This will likely outperform the CNNs, and emerge as the most performant model.

- **Advanced Architectures** — Alternative architectures could be experimented with architectures like EfficientNet that may offer better performance.

- **Rigorous Testing** — More rigorous testing could be performed by having radiologists review the results, and providing feedback on the model's performance. This will help to identify areas for improvement, and ensure that the model is clinically relevant. This could also be aided by the use of Grad-CAM to visualise the model's predictions, and understand its decision-making process.

- **Data Augmentation** — tf.image operations like random flips, rotations, and brightness adjustments could be introduced to improve generalisation for the CNNs.

- **User Interface** — A user-friendly interface can be developed to allow users to easily upload images, and view results.

- **Deployment** — The model can be deployed as a web service through REST APIs, and be inferenced through them for user-friendly accessibility, particularly for clinical settings.
- **CI/CD** — Implementing continuous integration, and continuous deployment (CI/CD) practices such as the introduction of containerisation with Docker will ensure that the system is easily maintainable, scalable, and transferable across different platforms.

Bibliography

[ADC21] G Ayana, K Dese, and SW Choe. *Transfer learning in breast cancer diagnoses via ultrasound imaging. Cancers 13: 738*. 2021.

[Ara+17] Teresa Araújo et al. "Classification of breast cancer histology images using convolutional neural networks". In: *PloS one* 12.6 (2017), e0177544.

[Car+24] Alessandro Carriero et al. "Deep learning in breast cancer imaging: State of the art and recent advancements in early 2024". In: *Diagnostics* 14.8 (2024), p. 848.

[FS25] Tanjim Fatima and Hamdy Soliman. "Application of VGG16 Transfer Learning for Breast Cancer Detection". In: *Information* 16.3 (2025), p. 227.

[GCB08] Maryellen L. Giger, Heang-Ping Chan, and John Boone. "Anniversary Paper: History and status of CAD and quantitative image analysis: The role of Medical Physics and AAPM". In: *Medical Physics* 35.12 (2008), pp. 5799–5820. DOI: https://doi.org/10.1118/1.3013555. eprint: https://aapm.onlinelibrary.wiley.com/doi/pdf/10.1118/1.3013555. URL: https://aapm.onlinelibrary.wiley.com/doi/abs/10.1118/1.3013555.

[GJ24] Daraje kaba Gurmessa and Worku Jimma. "Explainable machine learning for breast cancer diagnosis from mammography and ultrasound images: a systematic review". In: *BMJ Health & Care Informatics* 31.1 (2024), e100954.

[Guo+24] Ya Guo et al. "Machine learning and new insights for breast cancer diagnosis". In: *Journal of International Medical Research* 52.4 (2024), p. 03000605241237867.

[Jia+24] Bitao Jiang et al. "Deep learning applications in breast cancer histopathological imaging: diagnosis, treatment, and prognosis". In: *Breast Cancer Research* 26.1 (2024), p. 137.

[Kum+24] Dip Kumar Saha et al. "Segmentation for mammography classification utilizing deep convolutional neural network". In: *BMC Medical Imaging* 24.1 (2024), p. 334.

[Lat+24] M Latha et al. "Revolutionizing breast ultrasound diagnostics with EfficientNet-B7 and Explainable AI". In: *BMC Medical Imaging* 24.1 (2024), p. 230.

[Sri+23] Mahati Munikoti Srikantamurthy et al. "Classification of benign and malignant subtypes of breast cancer histopathology imaging using hybrid CNN-LSTM based transfer learning". In: *BMC Medical Imaging* 23.1 (2023), p. 19.

[Wan24] Lulu Wang. "Mammography with deep learning for breast cancer detection". In: *Frontiers in oncology* 14 (2024), p. 1281922.

[Yus+23] Marina Yusoff et al. "Accuracy analysis of deep learning methods in breast cancer classification: A structured review". In: *Diagnostics* 13.4 (2023), p. 683.