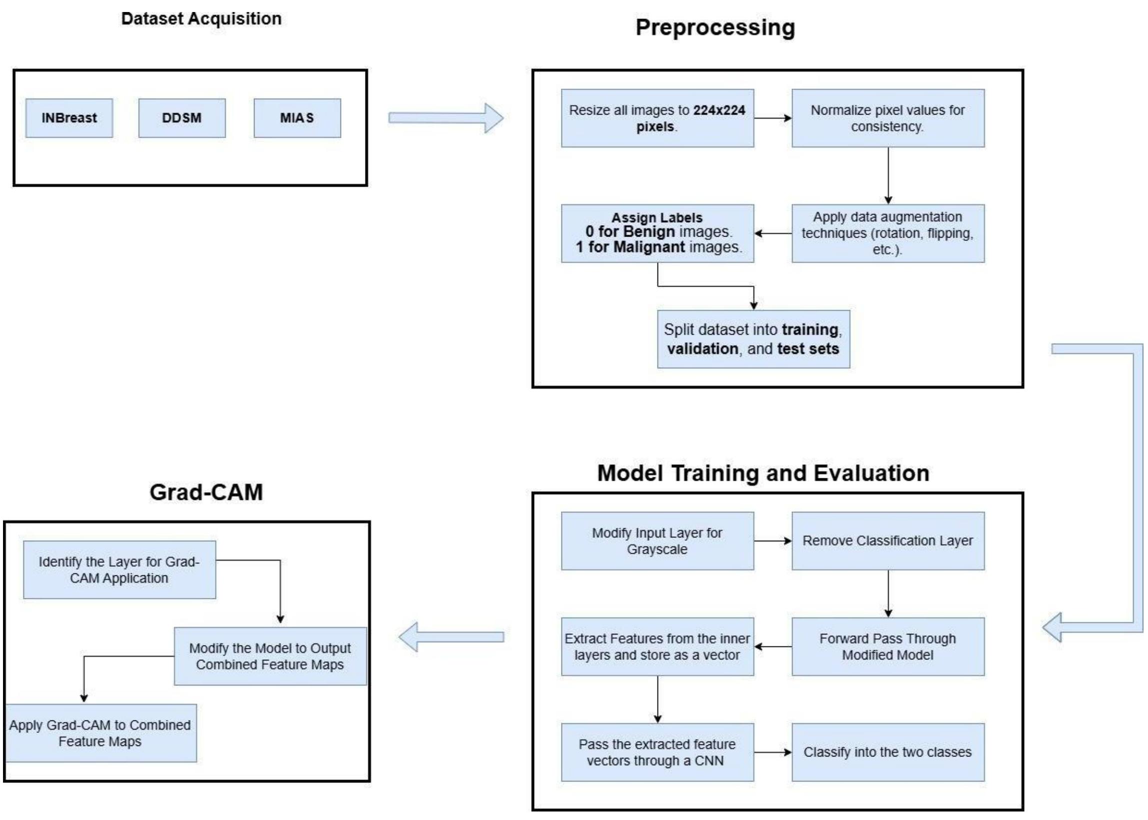
Methodology (Mammography Classification)

Overview:

In this project, the goal is to create a hybrid deep learning model that can classify mammography images as either benign or malignant. To do this, we use two convolutional neural networks: DenseNet121 and ResNet50. DenseNet121 is good at reusing features from earlier layers, while ResNet50 uses residual connections to solve the problem of vanishing gradients. We take the features from both models and combine them into one, which helps capture more detailed patterns from the images and improves the accuracy of the classification.

The combined features are then passed through fully connected layers to perform binary classification. To make the model more interpretable, we use Gradient- weighted Class Activation Mapping (Grad-CAM), which creates heatmaps showing which parts of the image are most important for the model's decision. This helps us understand how the model is making its predictions. Overall, the methodology covers everything from preparing the data, extracting features, training the model, and evaluating its performance, all while ensuring the results are both accurate and easy to interpret.

Methodology Flowchart:



Setting up Co-lab Environment:

To begin the project, I first set up a Google Colab environment, which provides an easy-to-use platform with free access to GPUs, making it ideal for deep learning tasks. After opening a new Colab notebook, I mounted Google Drive to access the datasets stored there. Once the drive was successfully mounted, I navigated to the specific directory containing the mammography dataset. The dataset is organized into two main subfolders: "Benign Masses" and "Malignant Masses," each containing the respective images. I used the os module in Python to load the paths to these folders, enabling me to load the images and labels for further processing. With the environment set up and the data loaded, I was ready to move on to the next steps in the project, including data preprocessing and model training.

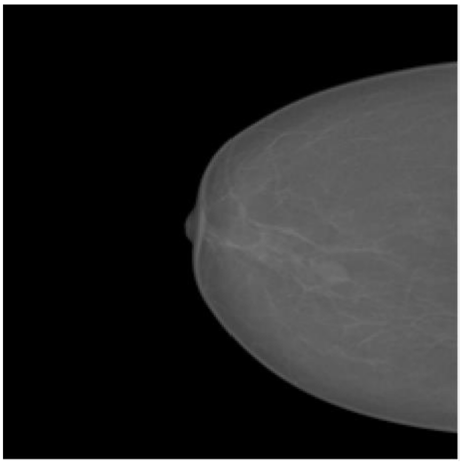
Dataset:

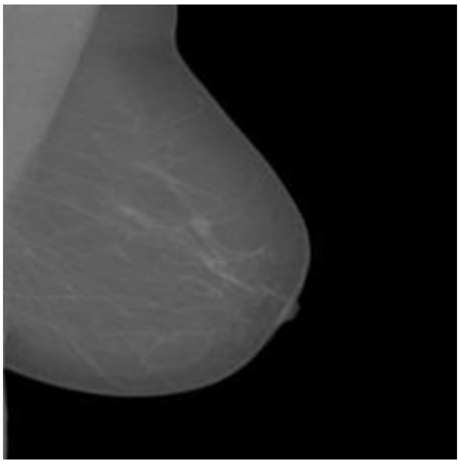
This project uses the Mendeley Mammogram dataset, which combines images from three major sources: INbreast, MIAS (Mammographic Image Analysis Society), and DDSM (Digital Database for Screening Mammography). The dataset is used to train a deep learning model that classifies mammography images as benign or malignant, which aids in the early identification of breast cancer. The combination of these three datasets yields a broad and well-annotated collection of photos, each with unique attributes and characteristics, ideal for training robust models.

1. **INbreast Dataset**:

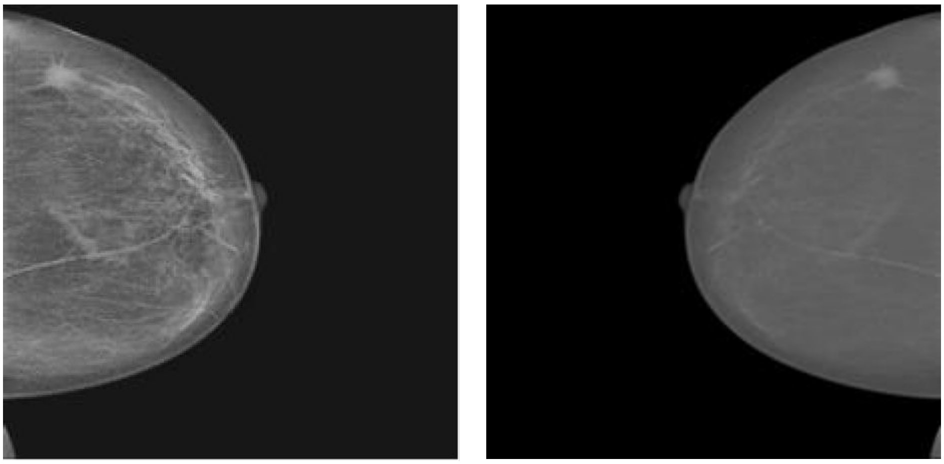
The INbreast dataset is a collection of high-quality digital mammogram images from 41 patients, containing both benign and malignant cases. Each image is labeled with relevant annotations, such as the presence of masses or microcalcifications.
Additionally, the dataset provides details about breast density, which can influence the ability to detect abnormalities. These annotations allow the dataset to be used for tasks involving the identification of different types of breast tissue patterns, which is crucial for accurate classification.

Benign sample images





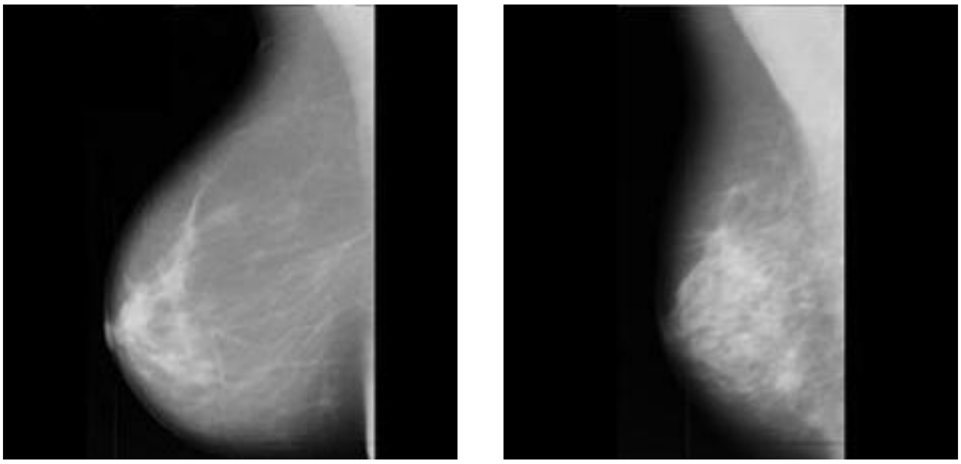
Malignant sample images



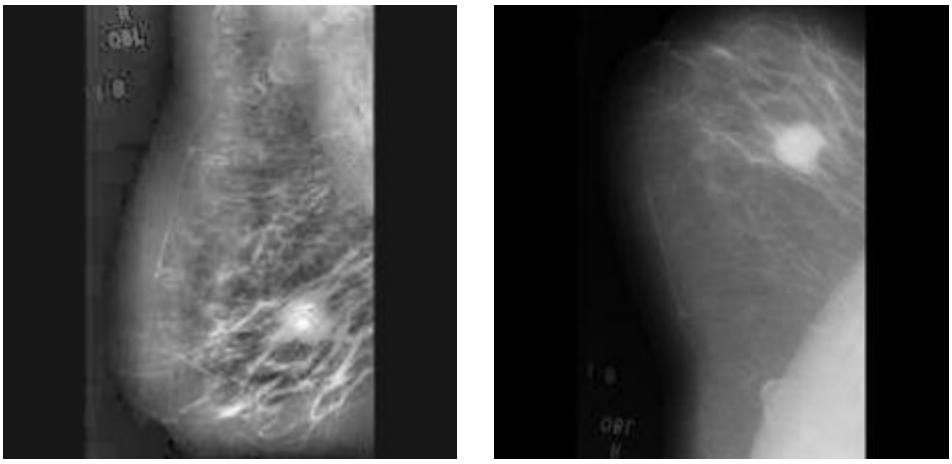
2. **MIAS Dataset**:

The MIAS dataset contains 322 mammogram images from 161 patients. These images are annotated with different types of abnormalities, such as masses, calcifications, and asymmetries, providing essential information for training classification models. One of the unique features of the MIAS dataset is its variety in image resolution, making it ideal for testing models that need to handle different qualities and sizes of mammograms. This helps ensure that the model is robust across varied input data.

Benign sample images



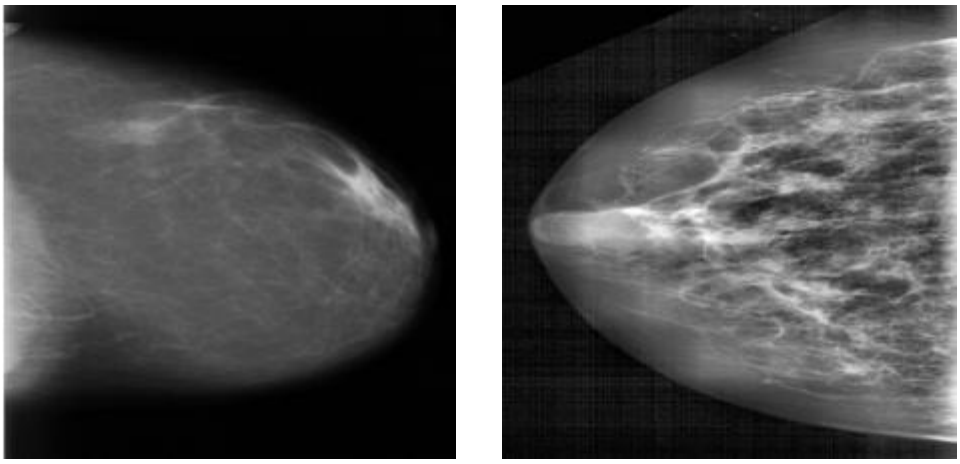
Malignant sample images

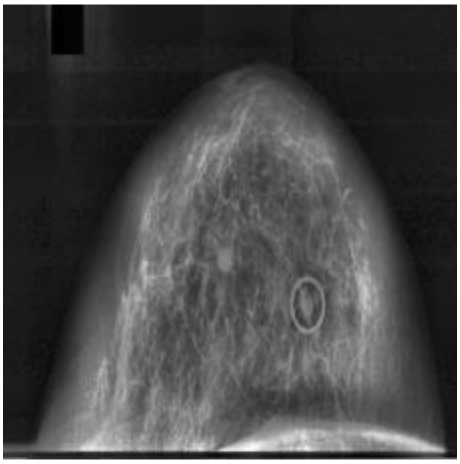
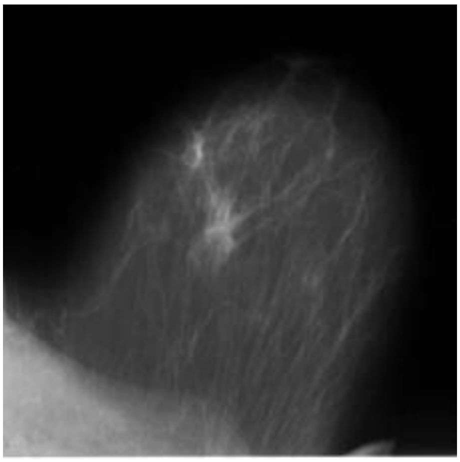


3. **DDSM Dataset**:

The DDSM dataset includes over 500 mammogram images from a diverse set of patients. It contains both standard and digitized film mammograms, offering a wide variety of cases to help train the model on different image types. The dataset is extensively annotated with information on masses, asymmetries, and calcifications, which are key features for breast cancer detection. The DDSM is one of the largest and most widely used mammography datasets, providing valuable data for developing deep learning models that require large and varied datasets for training.

Benign sample images



Attributes	INbreast	CBIS-DDSM	DDSM
Total Images	410	3,000	10,480 images
Image Resolution	High (up to 3328 × 4084 pixels)	Medium to High (varies, generally high for digitized)	Low to Medium (varies significantly)
Annotations	Yes (masks, detailed ROI, pathology)	Yes (detailed ROI, pathology, bounding boxes)	Limited (some ROI annotations, varies)
Classes	Benign, Malignant	Benign, Malignant	Benign, Malignant
File Format	DICOM	DICOM	LJPEG (digitized versions in DICOM available)
Dataset Focus	Digital mammography	Digitized film mammography	Screen-film mammography (digitized)
Availability	Restricted (license required)	Public (via TCIA,with approval)	Public (with request, through TCIA or collaborators)
Region of Interest (ROI)	Yes, detailed lesion annotations	Yes, detailed annotations	Limited or absent, depending on version
Augmentation Provided	No	No	No
Strengths	High-quality images, consistent format	Large size, well-annotated for ML tasks	Large size, useful for deep learning studies
Challenges	Small dataset size	Preprocessing required(e.g.,ROI extraction, normalization)	Inconsistent quality and preprocessing challenges

Table: Comparison of Mammography Datasets: INbreast, CBIS-DDSM, and DDSM

Preprocessing

The dataset is organized into two main categories: Benign and Malignant. These labels indicate the presence or absence of cancerous growths in the breast tissue. The images are primarily grayscale, with pixel values representing the intensity of the tissue. Before training the model, the dataset needs to undergo several preprocessing steps to ensure it is in the right format.

1. Resizing

First, the images are resized to a uniform size (224x224). Resizing is a crucial preprocessing step to ensure that all images in the dataset have the same dimensions. Since deep learning models like ResNet50 and DenseNet121 require input images of a fixed size, resizing all images to a standard dimension (such as 224x224 pixels) ensures compatibility with these models. Resizing also helps reduce computational load by standardizing the input, which allows the model to process the data more efficiently. This step is essential because images in the dataset might come in varying resolutions, and having consistent image dimensions improves the stability and performance of the model during training.

2. Normalizing

Normalization is another important step in the data preprocessing pipeline. It involves scaling the pixel values of images to a range of 0 to 1 by dividing the pixel values by 255 (since pixel values typically range from 0 to 255 in an 8-bit image). This helps the model learn more effectively by standardizing the input features and preventing large variations in pixel intensity from interfering with the learning process. Normalization also speeds up convergence during training by ensuring that the gradient updates during backpropagation are consistent across all input features, which ultimately helps the model to converge faster and achieve better performance.

3. Data Augmentation

Data augmentation is a powerful technique used to artificially expand the training dataset by applying random transformations to the original images. This helps improve the model's generalization capability by exposing it to a wider variety of image variations, preventing overfitting. In the case of the Mendeley Mammogram dataset, augmentation techniques such as rotation, zooming, and flipping are commonly applied. These transformations simulate different perspectives and conditions, making the model more robust when it encounters new, unseen images.

Labelling and storing the images

After preprocessing, the mammography images were saved as NumPy arrays to simplify storage and retrieval, allowing for more efficient data handling during model training. By transforming the photos to NumPy arrays, I ensured that the data could be easily retrieved and loaded, eliminating the need to repeat preprocessing processes each time the model was trained. This strategy dramatically increased the workflow's overall efficiency because NumPy arrays are fast and lightweight when processing massive datasets. Labels were assigned to each image based on its class, with '0' indicating benign masses and '1' indicating malignant masses. This label assignment helped to categorize the photos, which is necessary for training a supervised classification model.

After labelling the images, the images and their corresponding labels were saved as two separate NumPy arrays—one for the images and one for the labels. These arrays were then uploaded to Google Drive, which ensured that the data would be saved even if the session ended and made it easy to access when needed. Storing the data this way also made it quicker to load the images and labels during training, so we didn't have to process them again. Plus, saving the data as NumPy arrays helped with memory management, making it easier to load the entire dataset into memory when training and testing the model.

Model Training

Following the preprocessing, the mammography images were processed using convolutional neural networks (CNNs) to extract and classify features. We used the preprocessed dataset, which was scaled, normalized, and saved as numpy arrays, to make it easier to put into models. For this, we used two well-known models: DenseNet121 and ResNet50. These models were chosen because they are exceptionally good at detecting patterns and features in medical images. After extracting features from both models, we integrated them into a single set of features that allowed us to classify the photos as benign or malignant.

Densenet121

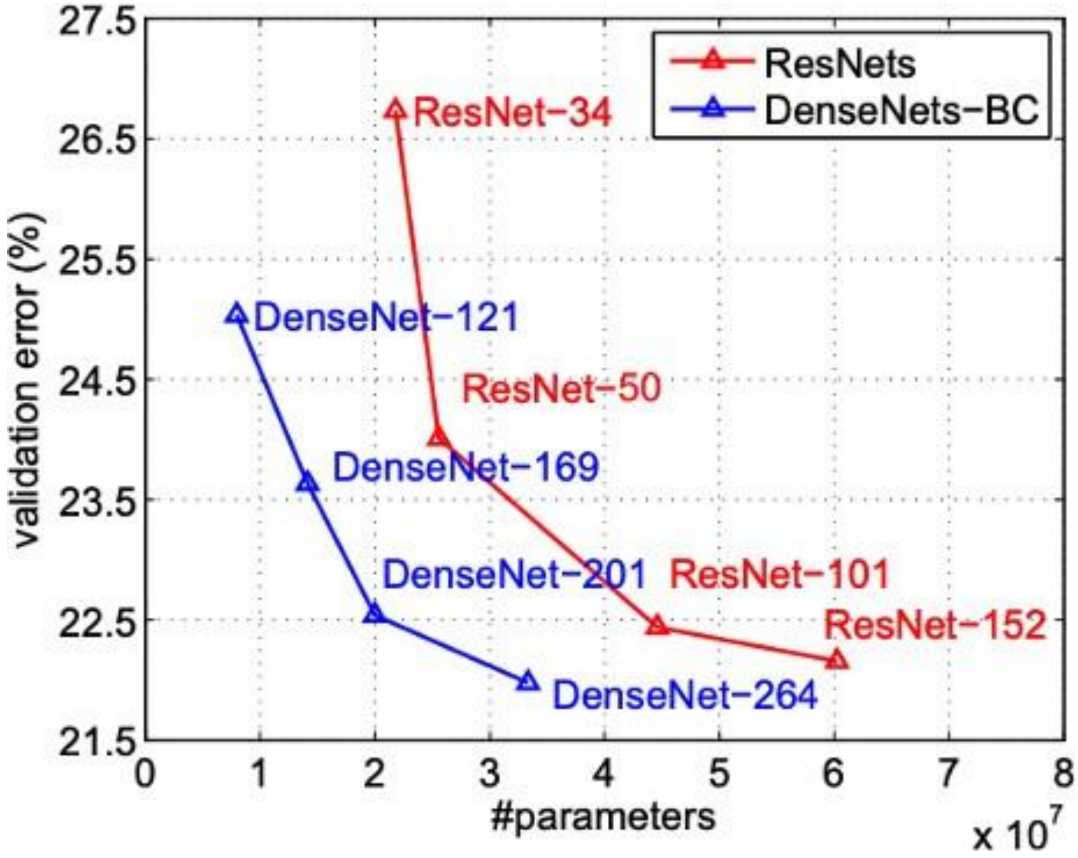
DenseNet121 is a convolutional neural network known for its innovative architecture that connects each layer to every other layer in a feed-forward manner. This unique structure, called dense connectivity, allows the model to pass information and gradients directly across layers, which significantly reduces the risk of vanishing gradients, a common challenge in deep networks. By reusing features throughout the network, DenseNet121 minimizes redundancy and improves efficiency. This characteristic makes it particularly useful for medical image analysis, where fine details and subtle patterns in images, such as masses or microcalcifications in mammograms, play a critical role in diagnosis. Its ability to focus on intricate image features while maintaining computational efficiency is why it is a strong choice for this project.

Resnet50

ResNet50, on the other hand, is a highly effective model based on the residual learning principle. Training can get difficult in deeper networks as gradients decrease, leading earlier layers to learn slower. ResNet50 addresses this issue by creating shortcut connections that skip specific layers, allowing the network to focus on learning the residuals, which are the differences between the present and desired outputs. This approach enables the model to learn deeper and more complex properties while maintaining efficiency. In mammography classification, ResNet50's capacity to identify high-level patterns, such as the overall structure of masses or asymmetries in breast tissue, complements DenseNet121's granular emphasis.

Hybrid Model Approach

We combine DenseNet121 and ResNet50 to take use of their respective capabilities, resulting in a more successful hybrid strategy. DenseNet121 excels at capturing fine- grained, low-level features required for medical imaging, such as subtle details in masses or microcalcifications in mammograms. ResNet50, on the other hand, is particularly good at extracting high-level, abstract properties like the overall shape or asymmetry of breast tissue. By combining the features extracted from these two models, we may obtain a more complete description of the data. This hybrid approach enables us to assess both local and global patterns in mammography images, yielding a model that is better able to distinguish between benign and malignant instances. The fusion of these feature sets not only enhances the classification accuracy but also ensures a more reliable diagnostic performance, making it a valuable method for addressing the challenges of medical image analysis.

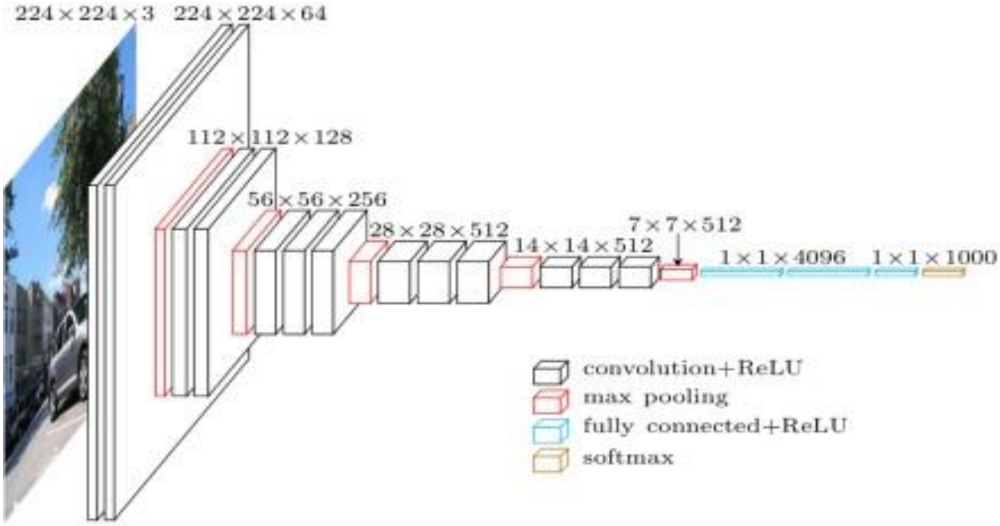


In our project, we aimed to utilize the pre-trained **DenseNet121** model for classifying mammogram images. DenseNet121 is a deep convolutional neural network typically trained on color images with three channels (RGB), but our dataset consists of **single-channel grayscale images** (mammograms). To use this pre-trained model, we needed to modify its input layer to handle single-channel images, while still leveraging the model's pre-trained weights.

Before passing the images through the model, the first step was to **preprocess** the mammogram images. We resized all images to a consistent size of 224x224 pixels, which is the input size expected by DenseNet121. Additionally, the pixel values were normalized by dividing them by 255.0, which scales the pixel values from the original range of 0 to 255 to a normalized range between 0 and 1. This normalization step is essential for faster model convergence during training, as it ensures that all features are on a similar scale.

Tweaking the models

DenseNet121, by default, expects an input shape of (224, 224, 3) to process three- channel RGB images. However, since our dataset consists of grayscale images with only a single channel, we needed to adjust the model to accept this format. Directly changing the input shape to (224, 224, 1) would cause issues because DenseNet121 is designed to work with three channels. To address this, we added a custom convolutional layer at the beginning of the model. This layer replicates the single-channel input into a three-channel format, effectively converting the grayscale images into an artificial RGB format. The convolution layer used a 3x3 kernel and three filters, ensuring the grayscale data is expanded to three channels while keeping the original information intact.



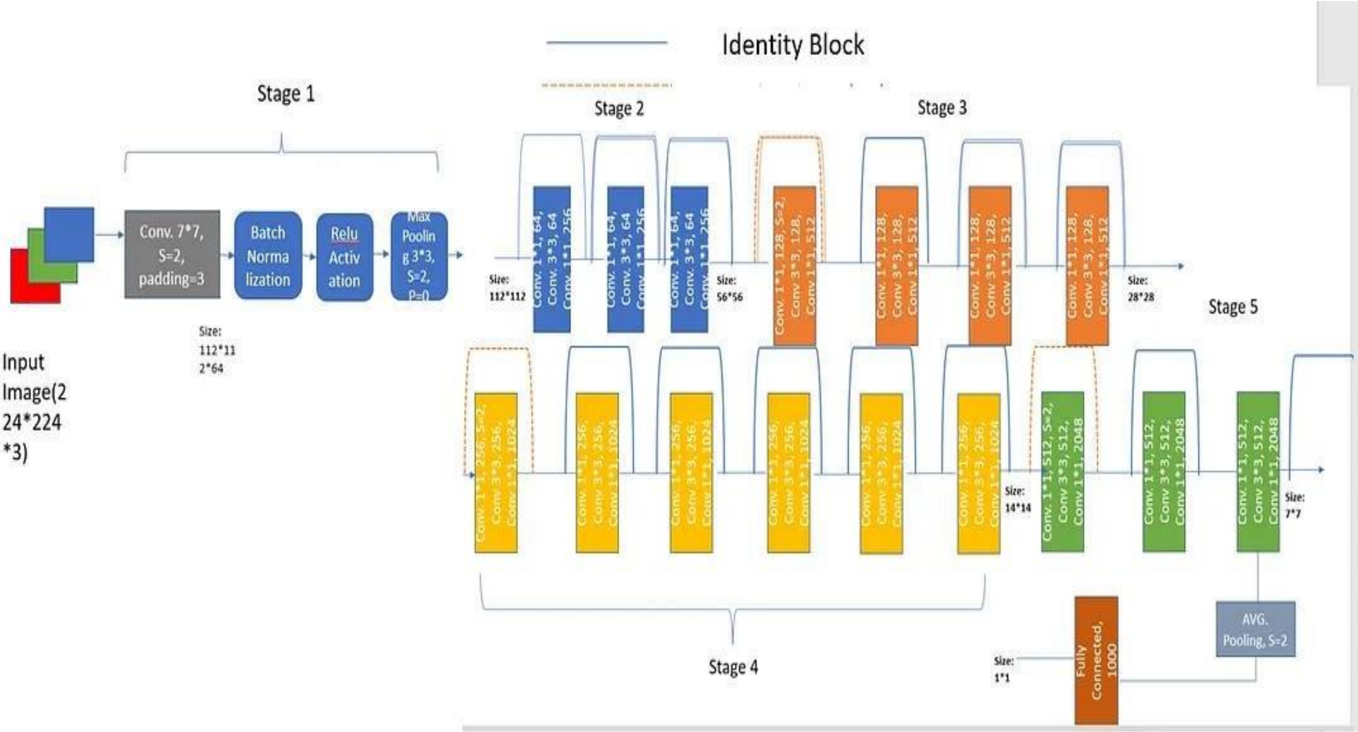
After modifying the input layer, we loaded the pre-trained DenseNet121 model using the Keras DenseNet121 function, with the include_top=False argument. This removed the final classification layers of the model, as they were originally designed for a different task. The goal was to use DenseNet121 as a **feature extractor** and add our own custom classification layers for our specific binary classification problem: identifying whether a mammogram image is benign or malignant. We used the pre-trained weights from ImageNet to initialize the model's convolutional layers, which helps the model detect general patterns in images like edges, textures, and shapes.

To adapt the model to our task, we added a **Global Average Pooling (GAP)** layer after the feature extraction layers. The GAP layer condenses the output from the DenseNet121 model into a single vector, representing the most important features from the entire image. Following this, we added a dense layer with a **sigmoid activation function** to predict a probability value between 0 and 1, indicating whether the image is benign or malignant. This architecture is well-suited for binary classification tasks, where the output is a probability representing the likelihood of the image belonging to one of the two classes.

Finally, the modified model was compiled with the **Adam optimizer** and **binary cross-entropy loss**. Adam is an efficient optimizer that adapts the learning rate during training, and binary cross-entropy is the appropriate loss function for binary classification tasks. We also tracked the model's **accuracy** during training to monitor its performance.

This approach allowed us to fine-tune the pre-trained DenseNet121 model for our specific mammogram classification task, benefiting from its ability to capture detailed features while adapting it to the grayscale format of our images. By making these adjustments, we were able to effectively use DenseNet121 for the task of mammogram classification, leveraging the power of transfer learning while customizing the model to handle single-channel images. These modifications ensured that the model could still recognize and classify subtle patterns in the mammogram images, improving diagnostic accuracy.

Like DenseNet121, ResNet50 typically expects an input shape of (224, 224, 3) for processing three-channel RGB images. However, since our dataset consists of grayscale images with only one channel, we needed to adjust the model accordingly. Directly changing the input shape to (224, 224, 1) would cause a mismatch with the model's architecture, so we used an alternative approach. We added a custom convolutional layer at the start of the model. This layer took the single-channel grayscale input and expanded it to a three-channel format, simulating RGB images. To achieve this, we used a 3x3 kernel with three filters, allowing the model to process the grayscale data while maintaining the original information.



Once the input layer was modified, we loaded the pre-trained ResNet50 model using the ResNet50 function from Keras, with the argument include_top=False. This excluded the original classification layers, as they were not suited for our task. Our goal was to utilize ResNet50 as a feature extractor, taking advantage of its ability to learn complex and high-level features, and then adding custom classification layers for our binary classification problem: distinguishing between benign and malignant mammograms. By using the pre-trained weights from ImageNet, we could leverage general features like edges, textures, and shapes, which are relevant for medical image analysis.

To adapt ResNet50 for our specific task, we inserted a Global Average Pooling (GAP) layer after the feature extraction layers. The GAP layer aggregates the feature maps into a single vector, representing the most important information from the image. After this, we added a fully connected dense layer with a sigmoid activation function, which outputs a probability value between 0 and 1, classifying the image as either benign (0) or malignant (1). This setup is well-suited for binary classification, as the output probability indicates the likelihood of the image belonging to either of the two classes.

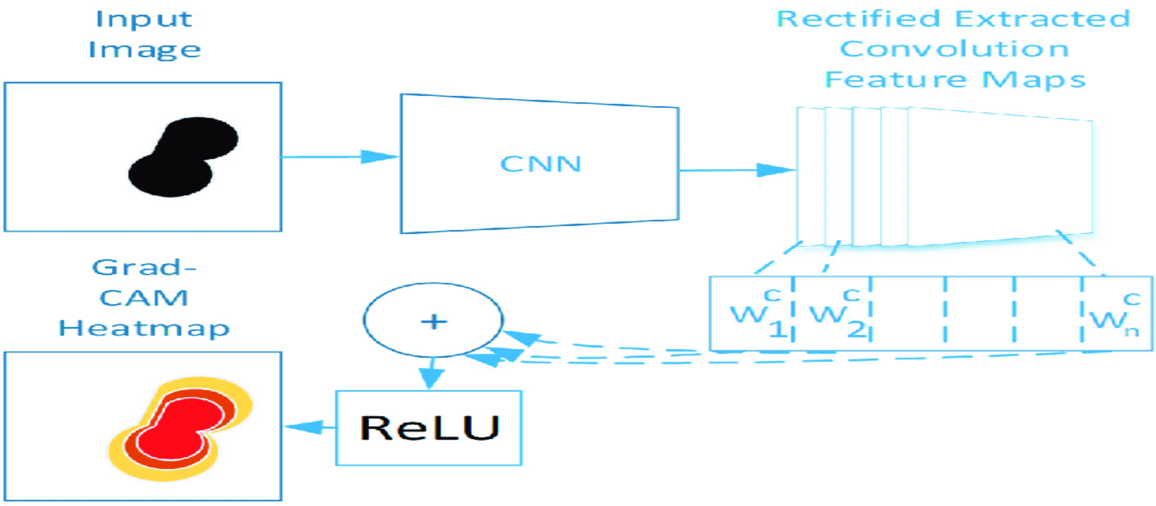
Lastly, we compiled the modified model using the Adam optimizer and binary cross- entropy loss. Adam is a popular optimizer known for adjusting the learning rate throughout training, which helps in speeding up convergence and avoiding overfitting. Binary cross-entropy was used as the loss function, which is standard for binary classification problems. We also tracked the model's accuracy during training to evaluate its performance. This combination of transfer learning with fine-tuning allowed us to tailor the pre-trained ResNet50 model to our mammogram classification task, effectively adapting it to handle single-channel images.Through these adjustments, we successfully leveraged ResNet50 for the mammogram classification task. The model was able to extract high-level features from the images while being adapted to handle grayscale input. This process ensured the model could accurately identify and classify subtle patterns in the mammogram images, ultimately enhancing its ability to assist in diagnostic analysis.

After obtaining the features from both ResNet50 and DenseNet121, the next step is to save them for later use in the feature fusion and classification stage. Since these models are used for feature extraction, we will store the output features (which are typically in the form of vectors after the Global Average Pooling layer) in a structured format that can be easily loaded for further processing. The next step is to combine the features extracted from both models. Since the features from ResNet50 and DenseNet121 are both in the form of feature vectors (e.g., each model outputs a vector of size 2048), we can concatenate them to form a combined feature vector that captures information from both models.

Now that we have the combined features, we can use them to train a classification model. Since our task is binary classification (benign vs malignant), a simple neural network or logistic regression model can be used for this task. Here, we'll use a basic fully connected (dense) neural network model, which is commonly used for classification tasks after feature extraction. After training the model, we can evaluate its performance on a separate test dataset or validation set to see how well it generalizes to new data.

Grad-CAM

Next Step is to implement GRAD-CAM for visualization. A potent visualization method for comprehending and interpreting convolutional neural networks' (CNNs') decision-making process, especially in image classification applications, is Grad- CAM (Gradient-weighted Class Activation Mapping). Grad-CAM gives information on the areas of an image that are most crucial for a model's predictions. It is particularly helpful for comprehending the decision-making process of deep learning models, which are sometimes regarded as "black-box" systems. This openness is crucial in domains such as medical image analysis, where credibility and validation may depend on the capacity to justify predictions.



How Grad-CAM works?

Grad-CAM works by analyzing the gradients flowing through the CNN during the prediction process. It calculates the importance of each feature map in the final convolutional layer, and then creates a heatmap showing which areas of the image influenced the model's decision the most. This heatmap can be overlaid on the original image, helping us understand which regions the model used to make its prediction.

Here's a breakdown of the Grad-CAM process:

1. **Forward Pass**: The input image is passed through the CNN model to generate predictions.
2. **Compute Gradients**: After the forward pass, Grad-CAM calculates the gradients of the output for the target class (benign or malignant) with respect to the feature maps of the final convolutional layer. These gradients show how much each feature map contributes to the final prediction.
3. **Global Average Pooling (GAP)**: The gradients are averaged across the spatial

dimensions to get a set of importance weights for each feature map.

4. **Weighted Sum of Feature Maps**: These weights are then used to perform a weighted sum of the feature maps from the final convolutional layer.
5. **Generate Heatmap**: The weighted feature maps are passed through a ReLU activation function, which ensures that only positive importance values are kept. This generates a heatmap showing which regions of the image are most important for the model's decision.
6. **Overlay Heatmap**: Finally, the heatmap is overlaid on the original image, highlighting the regions that influenced the model's decision the most.

How Grad-CAM is applied

To apply Grad-CAM to our mammogram classification task, we need to slightly modify the pre-trained ResNet50 and DenseNet121 models, as we are using them for feature extraction and binary classification (benign vs. malignant). Before using Grad-CAM, we need to make sure the models are set up properly. Both ResNet50 and DenseNet121 were fine-tuned to classify mammograms as either benign or malignant. Grad-CAM will be applied to the last convolutional layers that we added after the model features were saved

Here's how we prepare the model for Grad-CAM:

- **Choose the Layer Before Final Classification**: Grad-CAM works by focusing on the last convolutional layer, where high-level features like shapes or textures are captured. Since we are using combined feature vectors from both DenseNet121 and ResNet50, we need to identify the layer just before the final classification layer in our model, which is the point where both feature sets are merged. This will be the layer we apply Grad-CAM to, as it holds the most relevant information for the final decision.

- **Modify the Model to Output Combined Feature Maps**: Since Grad-CAM uses feature maps to generate the heatmap, we need to modify the model so that it outputs the combined feature maps from both DenseNet121 and ResNet50 before they are passed to the final fully connected layers for classification. This way, Grad-CAM can focus on the feature vectors derived from both networks, capturing the most important features before they are classified as benign or malignant. This setup will allow Grad-CAM to highlight the regions of the image that influenced the combined decision from both networks.

Once the Grad-CAM heatmap is generated, it can be overlaid on the original mammogram image. The heatmap shows the areas that contributed the most to the model's prediction. Hotter colors like red and yellow indicate regions of the image that were most influential for the decision. This visualization allows us to see where the model focused its attention when classifying the image.

For example, if the model predicts a malignant tumor, the heatmap will likely highlight the suspicious areas (e.g., masses or calcifications) in the mammogram. This helps clinicians understand which regions the model considers important, making the model's decision more interpretable.