Breast Cancer Classification

Fundamentals of Data Science
Sapienza University of Rome

Team members:
Andrea Di Vincenzo
Asia Montico
Emanuele Iaccarino
Mattia Mungo
Viktoriia Vlasenko

Task and Motivation

Motivation

Breast cancer is a leading cause of death among women worldwide. Early and precise diagnosis is critical for effective treatment and improving patient outcomes. Potentialy improve diagnostic tools and assist healthcare professionals in decision-making.

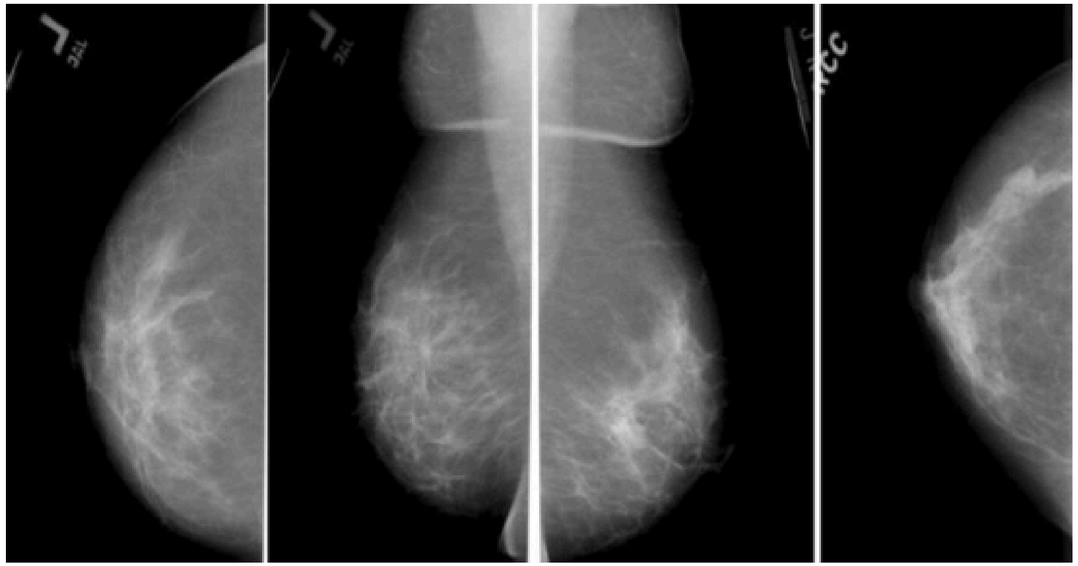


Figure 1: Images from CBIS-DDSM: Breast Cancer Dataset

Task

Initially CBIS-DDSM: Breast Cancer Image Dataset was categorized as Benign, Benign Without Recall and Malignant. After EDA we decided to define our classification task as binary, mapping Benign Without Recall into Benign.

Goal

Develop a custom CNN model to detect and classify breast cancer from histopathology images and its metadata.

The Dataset

BCBIS-DDSM: Breast Cancer Image Dataset contains 10237 **mammography images**, including 5122 **calcification** cases and **5115 mass** cases. Each case includes associated pixel-level annotations for lesions and metadata such as pathology (benign/malignant), assessment scores (BI-RADS), and lesion characteristics (e.g., shape, margin).

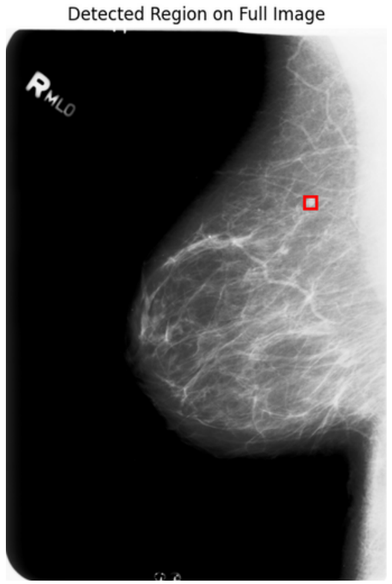


Figure 2: Cropped image detection experiment results

Cropped on Full Detection

To investigate the overlaying and scaling measure of the cropped image from the full mammogram X-Ray scan we run cv2 template matching. It was only suitable solution (comparing to the custom Canny Edge Detector from hw1). The similarity score is good:

```
Scale = 1.0, Location = (2335, 3894), Score = 0.9942139983177185
```

The other issue is that the ROI mask image has different shape from full image.

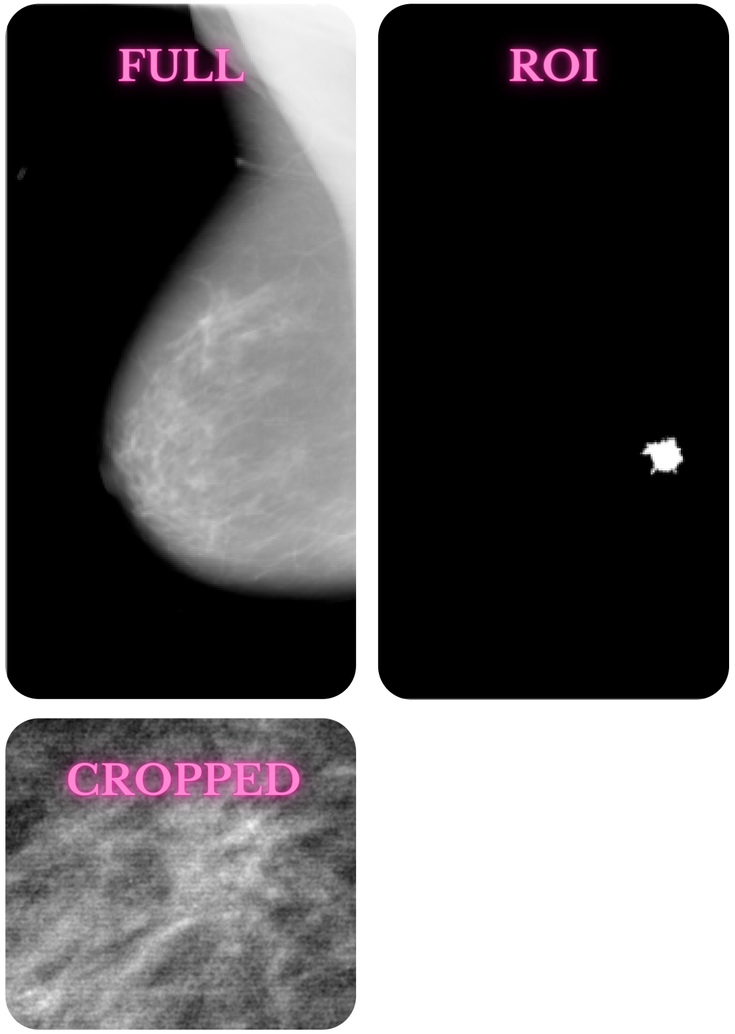


Figure 3-5: Images Types from CBIS-DDSM: Breast Cancer Dataset

EDA

HEATMAP OF MASS SHAPE VS PATHOLOGY

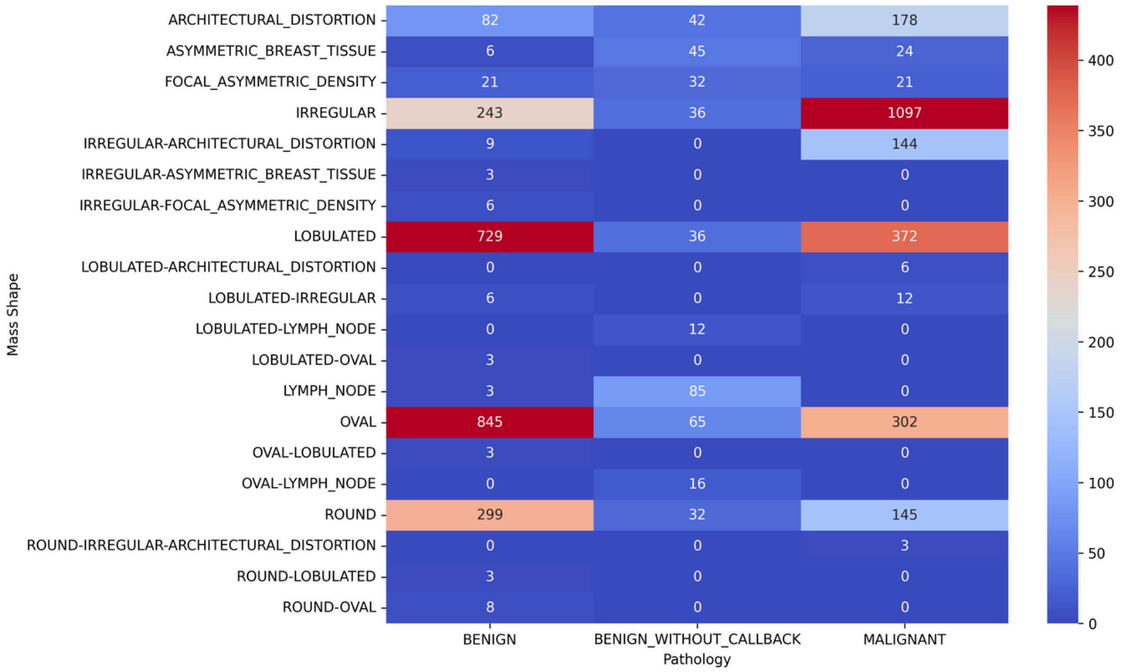


Figure 6: Pathology vs Mass Shape Heatmap

CALCIFICATION DISTRIBUTION

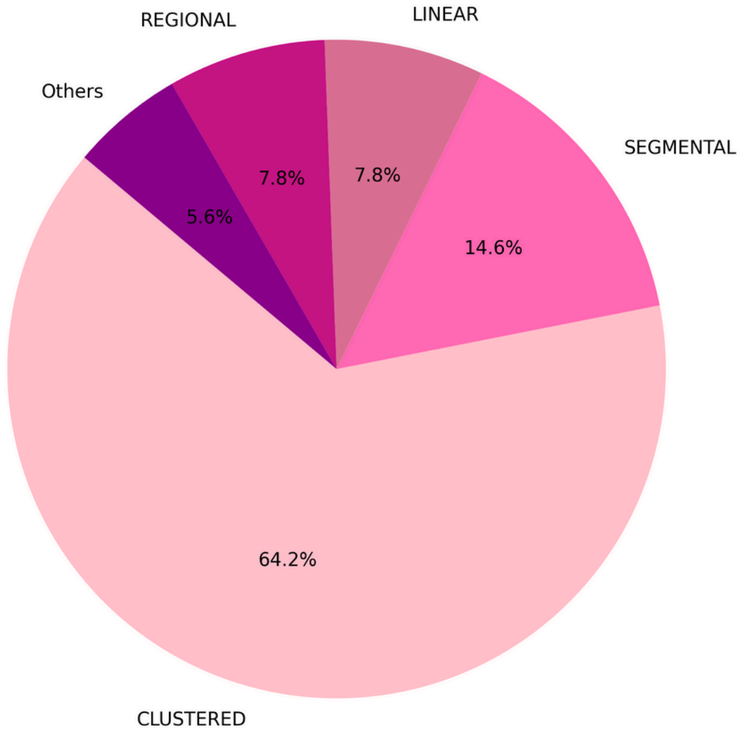
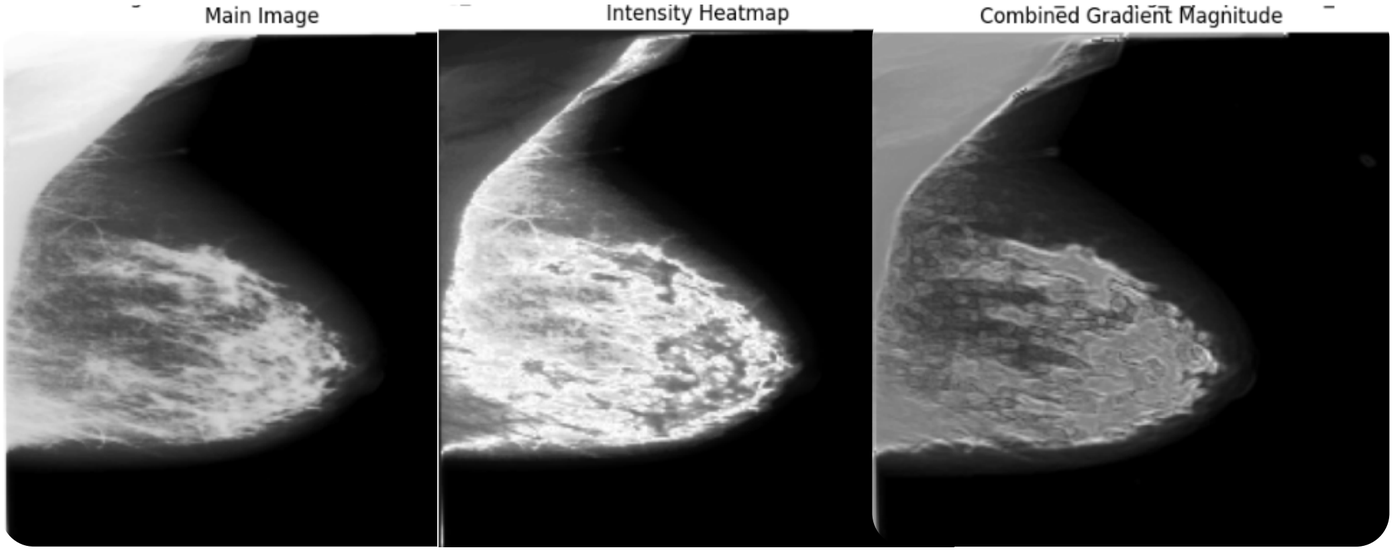


Figure 7: Pie Chart of Calcification Distribution

Image Processing Workflow

Data Preprocessing → **Image Preprocessing and Data Augmentation** → **Model**

After preprocessing the CBIS-DDSM dataset, we decided to focus on the subset related to **breast masses**.



Pretrained CNN

Figure 8: Processed Breast Image into 3 channel .npy files

Data Augmentation

In the MASS dataset we have apx. 4900 images that we are processing into 3 channel *.npy* files.

To handle the dataset imbalancing we implemented 3 strategies that helped us increase the data 3 times and makes the dataset really heavy (17 GB from 6.3 GB initially) to compute in short time.

```
Benign images in mass dataset: 8028
Malignant images in mass dataset: 6924
```

Data Augmentation Strategies

The strategies included:

- **Overlaying ROI masks** onto full mammogram images.
- Applying **random geometric transformations** such as scaling, rotation, and shifting of a ROI mass.
- **Blending cropped images back onto full images** at specific coordinates.

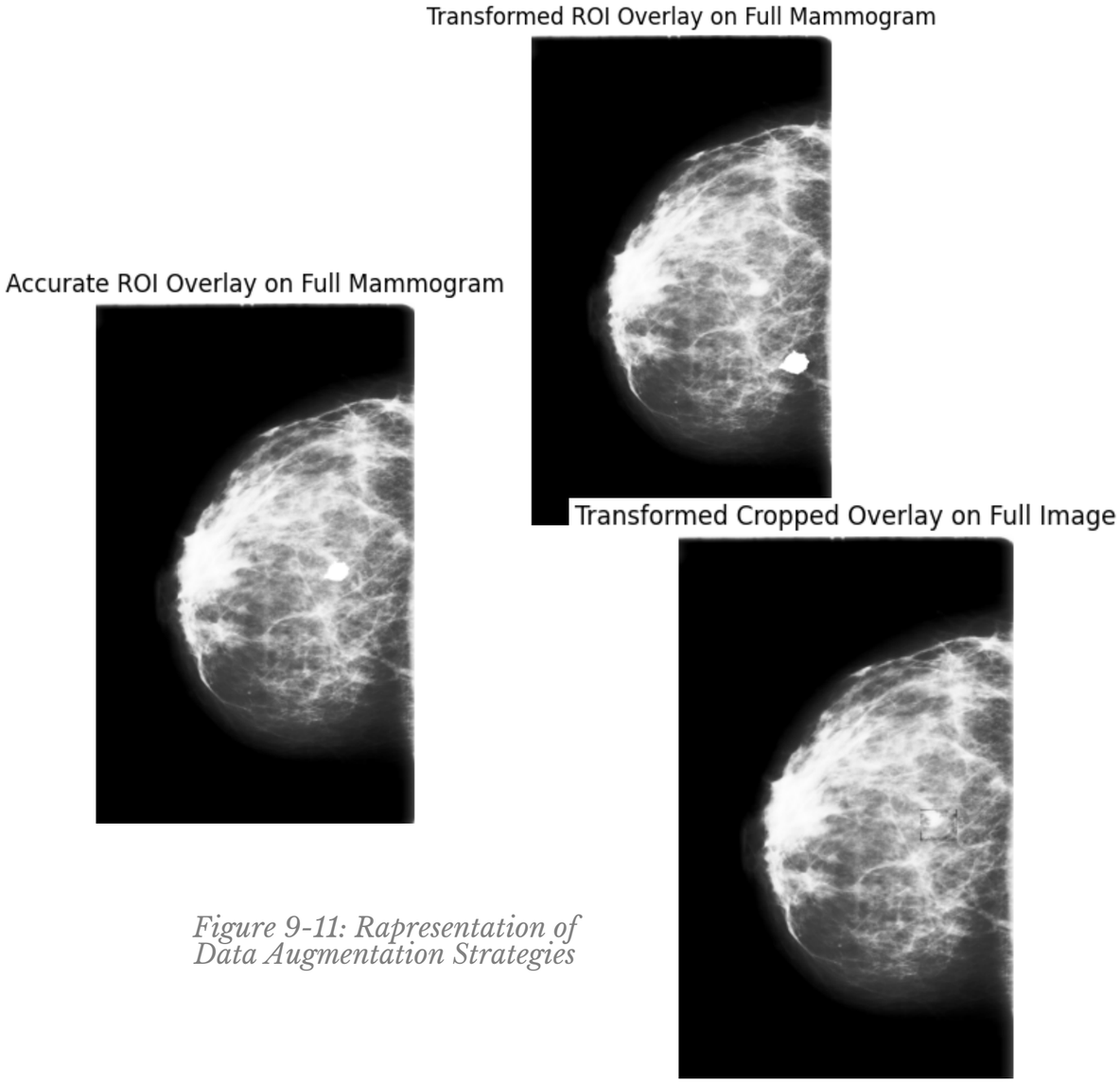
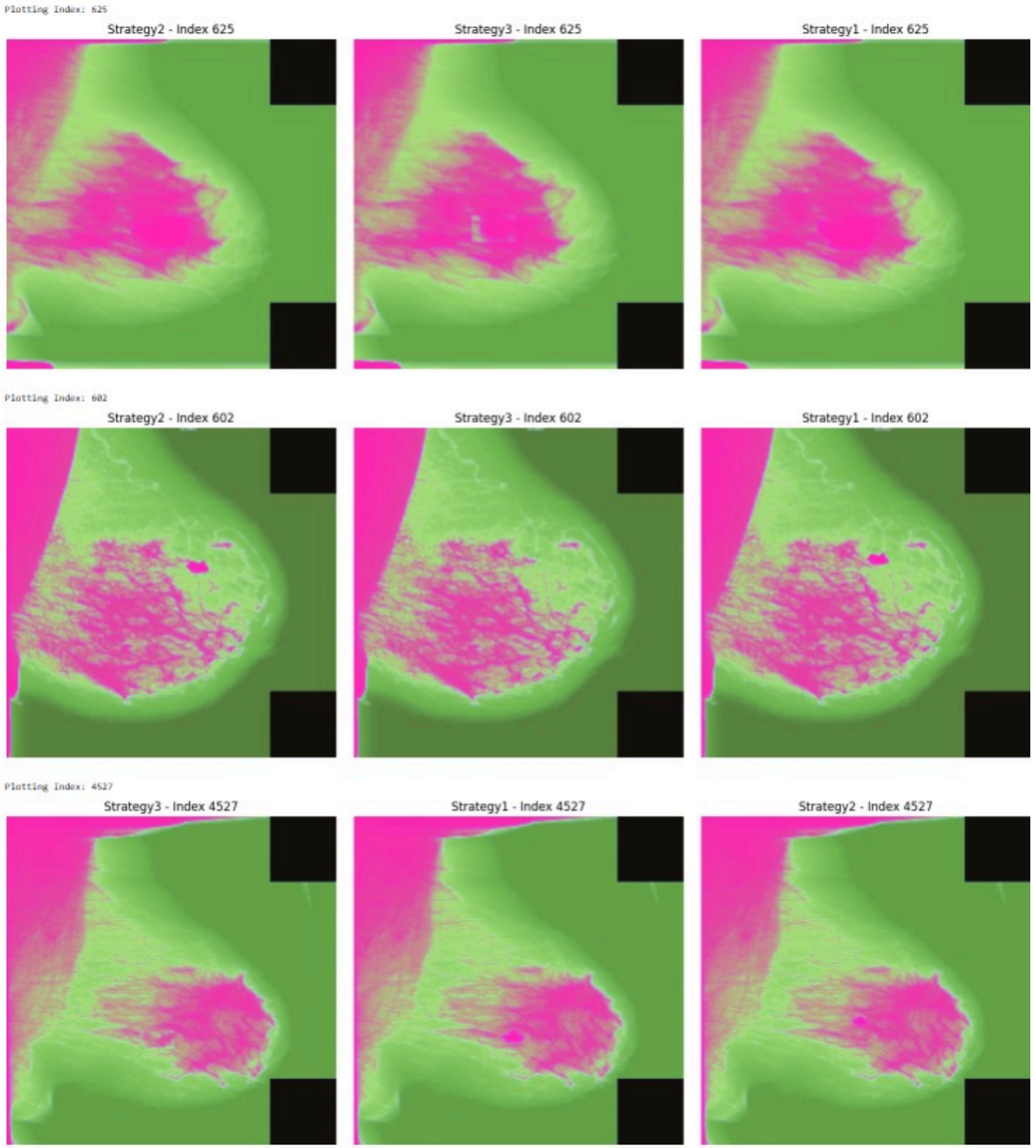


Figure 9-11: Rapresentation of Data Augmentation Strategies



Data Augmentation P.2

bad news

To prevent overfitting we refused to use data augmentation and run our CNN model on preprocessed images without augmentation.

RESULTS

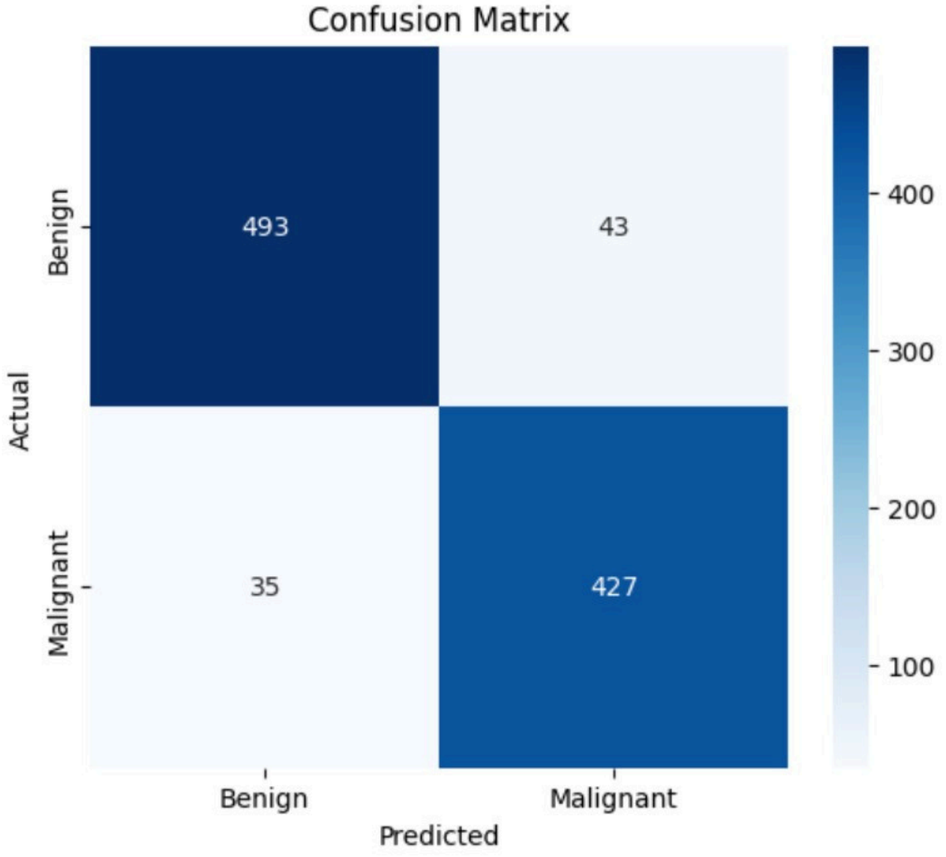


Figure 13-15: Results (Metrics Evaluation)

```
Classification Report:
              precision    recall  f1-score   support

      Benign       0.93      0.92      0.93       536
   Malignant       0.91      0.92      0.92       462

    accuracy                           0.92       998
   macro avg       0.92      0.92      0.92       998
weighted avg       0.92      0.92      0.92       998
```

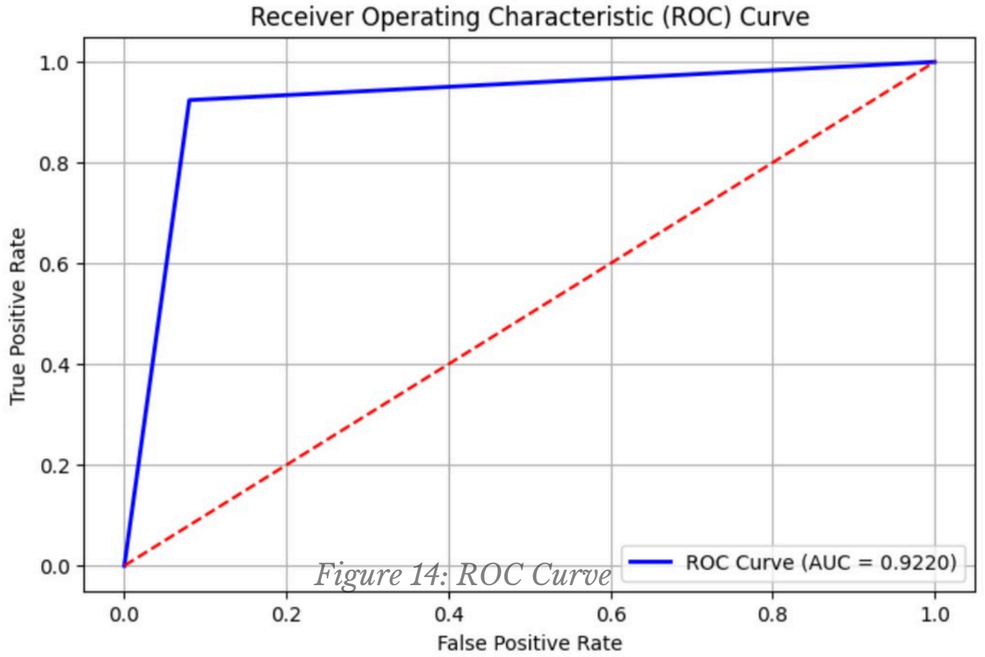


Figure 14: ROC Curve

Thank You

Relevant Links:

- An article about our dataset with the parsing of medical metadata for it in python: https://www.nature.com/articles/sdata2017177
- Dataset: https://www.kaggle.com/datasets/awsaf49/cbis-ddsm-breast-cancer-image-dataset/data
- ResNet implementation for Medical imgs: https://stackoverflow.com/questions/58151507/why-pytorch-officially-use-mean-0-485-0-456-0-406-and-std-0-229-0-224-0-2