National College of Ireland

Configuration Manual

A Comparative Study Of Machine Learning And Deep Learning Models For Breast Cancer Detection

MSc Research Project
MSc Data Analytics

Anjaly Antony Achandy
Student ID: X21223661

School of Computing
National College of Ireland

Supervisor: Musfira Jilani

National College of Ireland

MSc Project Submission Sheet

School of Computing

Student Name: Anjaly Antony Achandy
Student ID: x21223661
Programme: MSc Data Analytics **Year:** 2024
Module: MSc Research Project
Lecturer: Musfira Jilani
Submission Due Date: 12/12/2024
Project Title: A Comparative Study Of Machine Learning And Deep Learning Models For Breast Cancer Detection
Word Count: 1818 **Page Count:** 12

I hereby certify that the information contained in this (my submission) is information pertaining to research I conducted for this project. All information other than my own contribution will be fully referenced and listed in the relevant bibliography section at the rear of the project.
ALL internet material must be referenced in the bibliography section. Students are required to use the Referencing Standard specified in the report template. To use other author's written or electronic work is illegal (plagiarism) and may result in disciplinary action.

Signature: Anjaly Antony Achandy
Date: 12th December 2024

PLEASE READ THE FOLLOWING INSTRUCTIONS AND CHECKLIST

Attach a completed copy of this sheet to each project (including multiple copies)	✓
Attach a Moodle submission receipt of the online project submission, to each project (including multiple copies).	✓
You must ensure that you retain a HARD COPY of the project, both for your own reference and in case a project is lost or mislaid. It is not sufficient to keep a copy on computer.	✓

Assignments that are submitted to the Programme Coordinator Office must be placed into the assignment box located outside the office.

Office Use Only	
Signature:	
Date:	
Penalty Applied (if applicable):	

Configuration Manual

Anjaly Antony Achandy
Student ID: X21223661

1 Introduction

The provided configuration manual details the hardware and software utilized, along with their implementation and application, for conducting a comparative analysis of machine learning and deep learning models for breast cancer detection.

2 System Specification

The System Specification showcases the specification into two subsections Hardware Specifications and Software Specifications.

2.1 Hardware Specifications

The Table 1 Hardware Design Specification showcases the computational needs for the execution of the comparative analysis of machine learning and deep learning models for breast cancer detection. Furthermore, the table delineates the necessary minimum computational components like 12 Cores, 24 Threads, 4.7 GHz Base, 5.6 GHz max frequency.

Table 1 Hardware Design Specification

Component	***Specification***
Processor	AMD Ryzen 9 7900X (12 Cores, 24 Threads, 4.7 GHz Base, 5.6 GHz Turbo)
Graphics Card	AMD Radeon RX 7900 XT (20 GB GDDR6)
RAM	64 GB DDR5 (4800 MHz)
Storage	2 TB PCIe 4.0 NVMe SSD, 4 TB SATA HDD
Operating System	Windows 10 Pro Dual Boot
Motherboard	MSI MPG B650 Carbon WiFi
Power Supply Unit	1000W Platinum Certified PSU
Cooling System	Noctua NH-D15 Chromax
Network Card	Intel AX200 Wi-Fi 6 and Bluetooth 5.2
External Storage	8 TB Seagate Expansion External Drive

Backup System	APC 1500VA Smart-UPS with LCD Display

2.2 Software Specifications

Similar to hardware specification it is essential to meet the package level requirement for the execution of the code. This further becomes challenging with the ever evolving and migrating structure of the code libraries. Thus, the manual presents a detailed software specification list provided by the Table 2 Software and Library Specifications. This allows faster readability and usability of the code within the academic community.

Table 2 Software and Library Specifications

SOFTWARE/LIBRARY	VERSION
PYTHON	3.10.5
NUMPY	1.24.2
PANDAS	1.5.3
MATPLOTLIB	3.6.3
SEABORN	0.12.2
OPENCV (CV2)	4.8.0
TENSORFLOW	2.14.0
SCIKIT-LEARN	1.3.1
XGBOOST	1.7.6
IMGAUG	0.4.0
PILLOW (PIL)	9.4.0
SKIMAGE	0.21.0
TQDM	4.66.0
WARNINGS (BUILT-IN)	N/A

3 Project Development

3.1 Data Loading

Data loading is an integral aspect of detection within the medical image diagnosis spectrum thus the Figure 1 Loading Images and Labels across 360 angle degree from the MIAS dataset delineates the process and angles associated to each mammographic sample, its loading and label generation.

```
no_angles = 360
url ='/kaggle/input/mias-mammography/all-mias/'
def save_dictionary(path,data):
    print('saving catalog...')
    #open('u.item', encoding="utf-8")
    import json
    with open(path,'w') as outfile:
        json.dump(str(data), fp=outfile)
      # save to file:
    print(' catalog saved')
def read_image():
    print("Reading images")
    import cv2
    info = {}
    for i in range(322):
        if i<9:
            image_name='mdb00'+str(i+1)
        elif i<99:
            image_name='mdb0'+str(i+1)
        else:
            image_name = 'mdb' + str(i+1)
        # print(image_name)
        image_address= url+image_name+'.pgm'
        #print(image_address)
        #print(image_address)
        img = cv2.imread(image_address,1)
        # print(i)
        img = cv2.resize(img, (100,100))   #resize image
        rows, cols,color = img.shape
        info[image_name]={}
        for angle in range(0,no_angles,8):
            M = cv2.getRotationMatrix2D((cols / 2, rows / 2), angle, 1)    #Rotate 0 degree
            img_rotated = cv2.warpAffine(img, M, (cols, rows))
            info[image_name][angle]=img_rotated
    return (info)
def read_lable():
    print("Reading labels")
    filename = url+'Info.txt'
    text_all = open(filename).read()
    #print(text_all)
```

Figure 1 Loading Images and Labels across 360 angle degree from the MIAS dataset

The functions created in the Figure 1 Loading Images and Labels across 360 angle degree from the MIAS dataset allows for the loading of the dataset, but since malignant samples are far less than benign or non-harmful breast cancer mammographic images creates the need for data balancing. The goal of the study is to develop the use of mammography to the identification of breast cancer. Thus, the Mammographic Image Analysis Society (MIAS) dataset is used in this work. The collection includes mammograms that radiologists have annotated to help identify breast cancer. The dataset consists of 1024 × 1024 photos that are categorised into two severity classes: benign and malignant; and background tissue types, such as fatty, fatty glandular, and dense glandular.

The Figure 2 Load the dataset and balance both benign and malign classes illustartes the process of selecting the maximum samples of malignant unit and creating a balanced randomly chosen dataset for training. This helps not only in creating different training sample units for cross validation during the statistcial analytics but also aids the balacning of feature vectors on both class feature domain aiding the reduction of biase within the training of the model.

```
X=[]
Y=[]
for id in ids:
    for angle in range(0,no_angles,8):
        X.append(image_info[id][angle])
        Y.append(lable_info[id][angle])
X=np.array(X)
Y=np.array(Y)
x_train, x_test, y_train, y_test = train_test_split(X, Y, test_size=0.15, random_state=2021,shuffle=True)
X.shape,Y.shape
```

[4]

```
((5175, 100, 100, 3), (5175,))
```

```
import pandas as pd

# Assuming X and Y are defined
df = pd.DataFrame()
df['X_index'] = range(X.shape[0])
df['Y'] = Y

# Group by 'Y' and balance the DataFrame
g = df.groupby('Y')
min_size = g.size().min()

baldf = pd.concat([df[df['Y'] == 0].sample(min_size), df[df['Y'] == 1].sample(min_size)])

# Check the shape of the balanced DataFrame
print(baldf.shape)
```

[5]

```
(4680, 2)
```

```
X.shape,Y.shape
```

[6]

```
((5175, 100, 100, 3), (5175,))
```

Figure 2 Load the dataset and balance both benign and malign classes

3.2 Data Preprocessing

Preprocessing remains as the central most important segment of the data pipeline. The preprocessing allows for the extraction of pixels with most relevant information or at least to domain the most crucial group of pixels. To achieve the same Figure 3 Application of Preprocessing techniques like gray scaling and bilateral filtering methods incorporates Gray scaling to reduce any noise and normalise the image and then bilateral filtering for spatial and temporal data extraction. The function preprocesses an input image by first converting it to a grayscale format, simplifying the image to focus on intensity values. A bilateral filter is then applied to reduce noise while maintaining the edges, ensuring the image remains sharp. Next, the pixel values are normalized to a range of [0, 1], enhancing compatibility with machine learning models. Finally, the image is reshaped to add a channel dimension, preparing it as a 3D input suitable for deep learning architectures. This preprocessing ensures the image is optimized for neural network analysis.

```python
# Function to convert to grayscale and apply bilateral filtering
def preprocess_image(image):
    # Convert to grayscale
    gray_image = cv2.cvtColor(image, cv2.COLOR_BGR2GRAY)

    # Apply bilateral filtering (d, sigmaColor, sigmaSpace)
    filtered_image = cv2.bilateralFilter(gray_image, d=9, sigmaColor=75, sigmaSpace=75)

    # Normalize the image to range [0, 1]
    normalized_image = filtered_image / 255.0

    # Reshape the image to add a channel dimension (for models that expect 3D inputs)
    normalized_image = normalized_image.reshape(normalized_image.shape[0], normalized_image.shape[1], 1)

    return normalized_image
```

Figure 3 Application of Preprocessing techniques like gray scaling and bilateral filtering methods

3.3 Data Modelling

The data modelling pipeline allows for the creation of machine and deep neural models, the Figure 4 Application of Transfer learning for machine and deep learning comparison code creates a transfer learning feature extractor using the VGG19 neural network. The Model allows for the creation of a pseudo 3D convolution network, post which the vectors are passed into VGG19 extractor and then flattened into the model.

The model then is used to create extractions for the mammographic imagery given in the feature variable X. Finally xop contains the feature vectors which are further analysed into machine and deep learning comparisons

```python
def FEX(name):
    input1 = Input(shape=(X.shape[1],X.shape[2],1,))
    FE_sigma_X = Conv2D(3, kernel_size=3, padding ='same', activation='relu')(input1)
    FE_sigma_X = tf.keras.applications.VGG19(input_shape=(X_op.shape[1],X_op.shape[2],3), weights='imagenet', include_top=False)(FE_sigma_X)
    FE_sigma_X_OP = Flatten()(FE_sigma_X)
    model_FV = Model(inputs=input1,outputs=FE_sigma_X_OP)
    return model_FV
def extr(name,saver):
    model_FV = FEX(name)
    print(model_FV.summary())
    tf.keras.utils.plot_model(model_FV, show_shapes=True)
    X_OP = model_FV.predict(X_op)
    dfFE = pd.DataFrame(X_OP)
    print(X_OP.shape,Y.shape)
    np.save('X_'+saver+'_flatten.npy',X_OP)
    np.save('Y_'+saver+'_flatten.npy',Y)
    return X_OP
xop = extr('VGG19','VGG19')
```

```
Downloading data from https://storage.googleapis.com/tensorflow/keras-applications/vgg19/vgg19_weights_tf_dim_ordering_tf_kernels_notop.h5
80134624/80134624 ━━━━━━━━━━━━━━━━━━━━ 2s 0us/step

Model: "functional_1"
```

Layer (type)	Output Shape	Param #
input_layer (InputLayer)	(None, 100, 100, 1)	0
conv2d (Conv2D)	(None, 100, 100, 3)	30
vgg19 (Functional)	(None, 3, 3, 512)	20,024,384
flatten (Flatten)	(None, 4608)	0

Figure 4 Application of Transfer learning for machine and deep learning comparison code

The research also creates a hybrid neural network called Residual Convolutional Attention Neural Network which consists of multiple convo attention networks with 64, 16 and 1 variable neurons. The architecture is delineated in the Figure 5 Residual Convolutional Attention Neural Network. The Hybrid architecture allows for the better utilization of computation, explainable AI in breast cancer mammographic detection and the formulation of higher results in comparison to the machine and deep learning methods created from the Figure 4 Application of Transfer learning for machine and deep learning comparison code. Further, Convolutional and attention neural networks are combined to form RCANN. The network can solve the disappearing and expanding gradient problems, which were thought to be persistent across deep learning and machine learning techniques. Remaining convolutional attention mechanisms may be integrated. In order to decode characteristics within the cancerous samples that serve as the basis for detection, the model seeks to attain spatial attention.

```
in_lay = Input(X.shape[1:])
base_pretrained_model = VGG16(input_shape =  X.shape[1:], include_top = False, weights = 'imagenet')
base_pretrained_model.trainable = False
pt_depth = base_pretrained_model.get_output_shape_at(0)[-1]
pt_features = base_pretrained_model(in_lay)
from keras.layers import BatchNormalization
bn_features = BatchNormalization()(pt_features)

# here we do an attention mechanism to turn pixels in the GAP on an off

attn_layer = Conv2D(64, kernel_size = (1,1), padding = 'same', activation = 'relu')(bn_features)
attn_layer = Conv2D(16, kernel_size = (1,1), padding = 'same', activation = 'relu')(attn_layer)
attn_layer = Conv2D(1,
                    kernel_size = (1,1),
                    padding = 'valid',
                    activation = 'sigmoid')(attn_layer)
# fan it out to all of the channels
up_c2_w = np.ones((1, 1, 1, pt_depth))
up_c2 = Conv2D(pt_depth, kernel_size = (1,1), padding = 'same',
               activation = 'linear', use_bias = False, weights = [up_c2_w])
up_c2.trainable = False
attn_layer = up_c2(attn_layer)

mask_features = multiply([attn_layer, bn_features])
gap_features = GlobalAveragePooling2D()(mask_features)
gap_mask = GlobalAveragePooling2D()(attn_layer)
# to account for missing values from the attention model
gap = Lambda(lambda x: x[0]/x[1], name = 'RescaleGAP')([gap_features, gap_mask])
gap_dr = Dropout(0.5)(gap)
dr_steps = Dropout(0.25)(Dense(128, activation = 'elu')(gap_dr))
out_layer = Dense(2, activation = 'softmax')(dr_steps) # linear is what 16bit did
mammo_model = Model(inputs = [in_lay], outputs = [out_layer])

mammo_model.compile(optimizer = 'adam', loss = 'sparse_categorical_crossentropy',
                    metrics = ['sparse_categorical_accuracy'])

mammo_model.summary()
```

Figure 5 Residual Convolutional Attention Neural Network

After the creation of the RCANN model, the study incorporates a training procedure ensuring efficient and robust optimization. Model Check point saves the model's weights whenever there is an improvement in validation loss, ensuring the best model is retained. Reduce LR On Plateau adjusts the learning rate dynamically by reducing it when validation loss plateaus, enhancing convergence. Early Stopping stops the training early if validation loss does not improve after a specified patience period,

preventing overfitting. These callbacks help the model to be faster and efficient in computational utility. The aforementioned steps are highlighted vividly in the Figure 6 RCANN Modelling

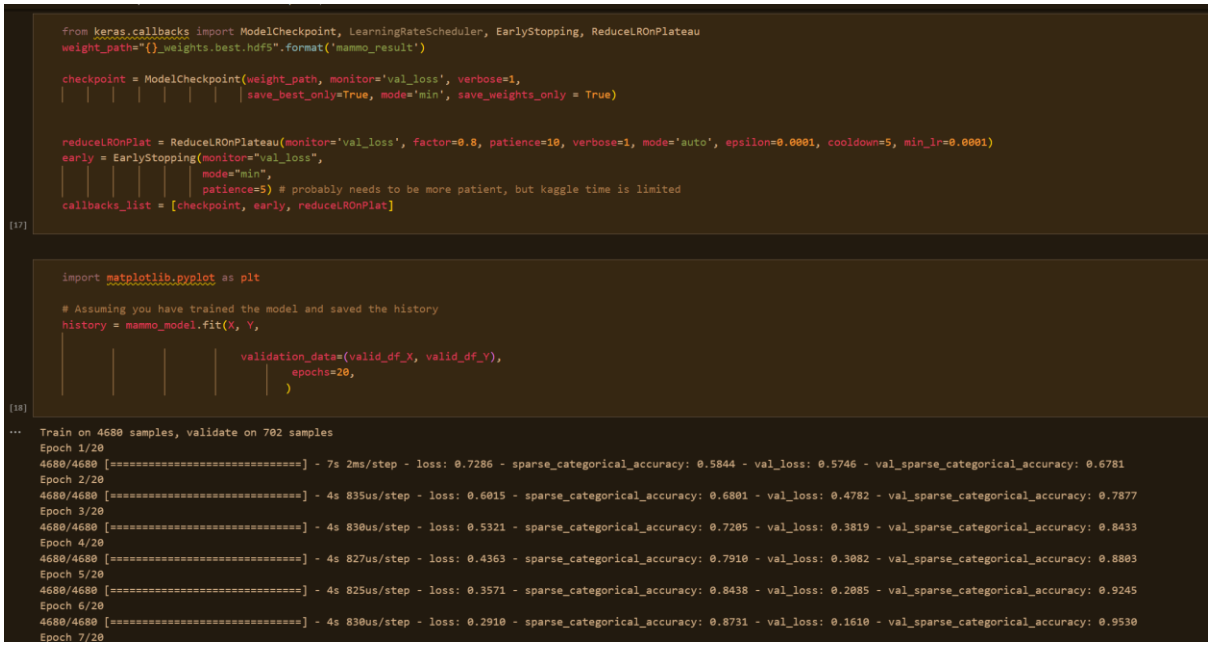

Figure 6 RCANN Modelling

3.4 Evaluation and Explainable AI

3.4.1 Evaluation

This section evaluates the model's parameters across various subheadings such as machine learning, deep learning comparison, RCANN performance evaluation and explainable AI visualization.

Figure 7 Class-wise Model Evaluation

Figure 7 Class-wise Model Evaluation showcases the evaluation process of the RCANN model. It also creates a vivid list of class wise computational metrics. The class wise comparison allows the model to help deduce the architectural flaws if any and look at both benign and malignant sample cases in case of an existing of inherent biased within the modelling tool.

Figure 8 Evaluation illustration - confusion matrix and Classification Report showcases the confusion matrix which tells the true positive, false positive, true negative and false negative number of samples within the test group. Further the Figure 8 Evaluation illustration - confusion matrix and Classification Report illustrates the classification report calculating class wise precision, recall, F1 score, accuracy. As well as micro, macro and mean scores for the metrics.

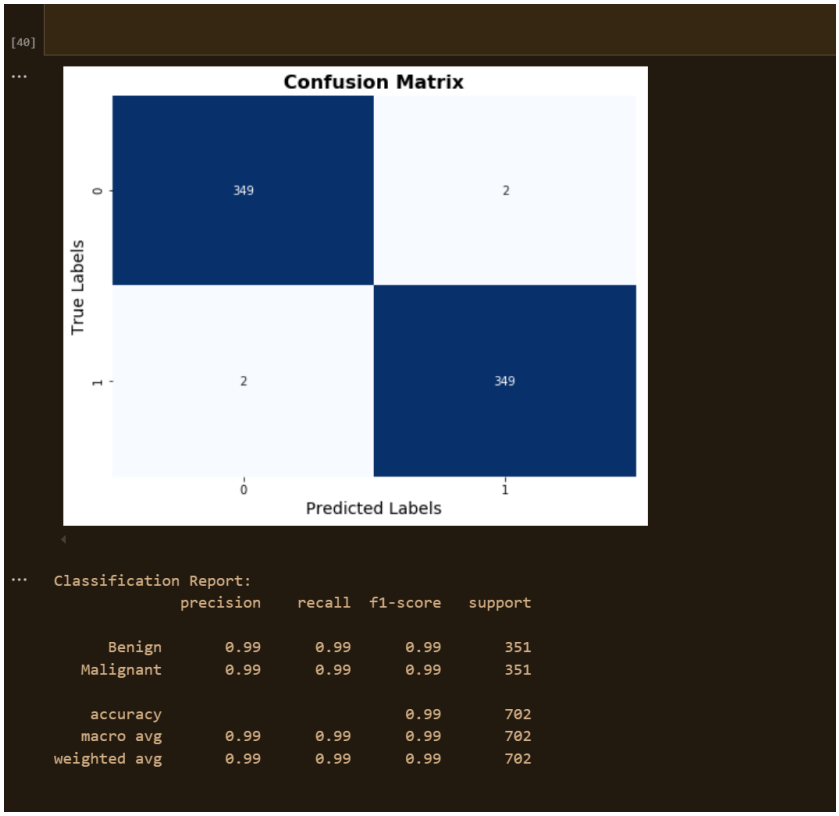

Figure 8 Evaluation illustration - confusion matrix and Classification Report

Another sample of the evaluation result is the One vs Rest Precision Recall Growth curve. This shows the balance within Precision rate and recall rate for a better nuanced view of the modelling technique showcased by the Figure 9 Precision Recall Curve.

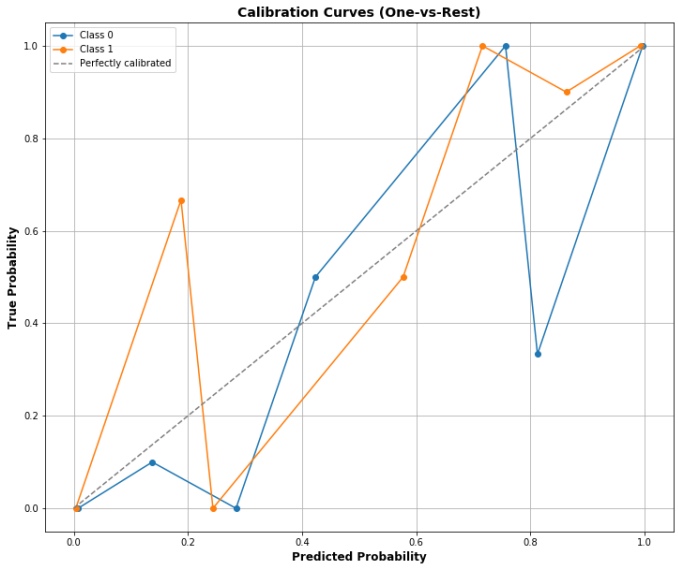

Figure 9 Precision Recall Curve

3.4.2 Explainable AI

The Explainable AI is one of the most important parts of the project this allows for the visualisation of the attention maps which are inherently responsible for the creation and detection of the benign and malignant samples. The attention maps highlight the inner working of the RCANN architecture thus the Figure 10 Attention Illustration where in the images are chosen at random form the test set, attention function is used to extract the output.

```
import keras.backend as K
import numpy as np
import matplotlib.pyplot as plt

# Select random indices for visualization  valid_df_X, valid_df_Y
rand_idx = np.random.choice(range(len(valid_df_X)), size=6, replace=False)

# Define the attention function
attn_func = K.function(
    inputs=[mammo_model.input, K.learning_phase()],
    outputs=[attn_layer.output]
)

# Set up the figure and axes
fig, m_axs = plt.subplots(len(rand_idx), 2, figsize=(8, 4 * len(rand_idx)))
[c_ax.axis('off') for c_ax in m_axs.flatten()]

# Loop through each selected index
for c_idx, (img_ax, attn_ax) in zip(rand_idx, m_axs):
    cur_img = valid_df_X[c_idx:(c_idx + 1)]
    attn_img = attn_func([cur_img, 0])[0]

    # Display the original image in RGB
    img_ax.imshow(cur_img[0], cmap='bone')
    attn_ax.imshow(attn_img[0, :, :, 0], cmap='viridis',
                   vmin=0, vmax=1, interpolation='lanczos')

    # Display real and predicted labels
    real_label = np.argmax(valid_df_Y[c_idx])
    img_ax.set_title(f'Mammography\nClass: {real_label}')
    pred_label = np.argmax(mammo_model.predict(cur_img)[0])
    attn_ax.set_title(f'Attention Map\nPred: {pred_label}')

# Save the figure
fig.savefig('attention_map.png', dpi=300)
plt.show()
```

[29]

...

Figure 10 Attention Illustration

Figure 11 Attention or XAI sample view showcased the attention map for the mammographic samples from the attention neural network. This allows for the explainable AI calibration of contours and the region of interest for the based on the malignant and benign classes.

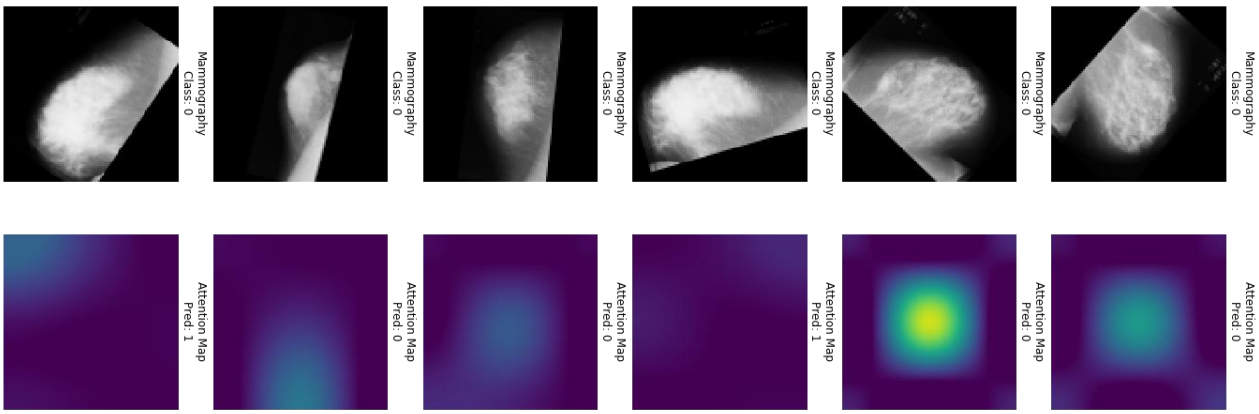

Figure 11 Attention or XAI sample view

4 Conclusion

The paper offers a thorough grasp of breast cancer, the present constraints on automated breast cancer diagnosis, and research needs related to the detection. The study suggests the RCANN module, a residual convolutional attention neural network, and fills in the research gaps found in the literature review. The network is extensively contrasted using machine learning methods such as adaptive boosting, decision trees, random forests, and voting classifiers. Additionally, convolutional neural networks, dense neural networks, and artificial neural networks are contrasted in order to have a whole simulation of deep neural architecture.

A number of performance criteria, including precision, recall, F1 score, accuracy, sensitivity, specificity, and support, have been used to compare the suggested RCANN. ROC-AUC curves, bar plots, and confusion matrices are used to better visualise the measurements. The study on the identification and use of AI as a tool for medical image processing in breast cancer sample pictures is empirically supported by the suggested RCANN model's ability to infer attention maps to provide powerful kernel views that support explainable AI.

References

Zebari, D. A., Zeebaree, D. Q., Abdulazeez, A. M., Haron, H. & Hamed, H. N. A. Improved threshold based and trainable fully automated segmentation for breast cancer boundary and pectoral muscle in mammogram images. IEEE Access 8, 203097–203116 (2020).

Liu, M. et al. A deep learning method for breast cancer classification in the pathology images. IEEE J. Biomed. Health Inform. 26(10), 5025–5032 (2022).
Hamed, G., Marey, M., Amin, S. E. & Tolba, M. F. Automated breast cancer detection, and classification in full field digital mammograms using two full and cropped detection paths approach. IEEE Access 9, 116898–116913 (2021).

Zhang, X. et al. Classification of whole mammogram and tomosynthesis images using deep convolutional neural networks. IEEE Trans. Nanobiosci. 17(3), 237–242 (2018).

Rastghalam, R. & Pourghassem, H. Breast cancer detection using MRF-based probable texture feature and decision-level fusion-based classification using HMM on thermography images. Pattern Recognit. 51, 176–186 (2016).