UNIVERSITÀ DI PISA

COMPUTATIONAL INTELLIGENCE AND DEEP LEARNING

Convolutional Neural Network for Medical Imaging Analysis

Professors: B. Lazzerini - A. Renda

2021

Students: A. Schiavo – M. Gómez – M. Daole

Contents

Chapter 1

1. Introduction

Breast cancer is one of the most common types of cancer in women. Early detection and treatment can effectively improve cure rates and reduce mortality. Detecting breast cancer using mammographic images is a cost-effective technique, and radiologists can make a diagnosis by analyzing these images. However, the large number of mammographic images produced day by day has brought a huge workload on radiologists and also increased the rate of misdiagnosis. Therefore, developing a computer-aided diagnosis (CAD) system can significantly relieve the pressure on radiologists and improve the diagnosis accuracy.

Machine learning therefore quickly enters the picture, based on large, heterogeneous data sets, the automatic analysis for mammography images needs to be analyzed and make predictions from the regions of interest and classify these regions into normal or abnormal (benign and malignant).

1.1. Notebooks

For this development, you can check and download all the collaborative work available on GitHub[1] with tasks solutions, adequately described and commented. The cells output reflect the results obtained on this report.

1. **Task 1 – Knowing the Dataset**
    a. Knowing_dataset.ipynb
2. **Task 2 – CNN from Scratch**
    a. Scratch_CNN_benign_vs_malign.ipynb
    b. Scratch_CNN_masses_vs_calc.ipynb
3. **Task 3 -Pretrained**
    a. PreTrained_CNN_benign_vs_malign.ipynb
    b. PreTrained_CNN_benign_vs_malign_InceptionV3.ipynb
    c. PreTrained_CNN_benign_vs_malign_ResNet50.ipynb
    d. PreTrained_CNN_benign_vs_malign_VGG16.ipynb

    ---

    e. PreTrained_CNN_masses_vs_calc.ipynb
    f. PreTrained_CNN_masses_vs_calc_InceptionV3.ipynb
    g. PreTrained_CNN_masses_vs_calc_ResNet50.ipynb
    h. PreTrained_CNN_masses_vs_calc_VGG16.ipynb
4. **Task 4 – Baseline**
    a. Baseline_CNN.ipynb
5. **Task 5 – Ensemble**
    a. Ensemble.ipynb

[1] GitHub public repository:
https://github.com/MarshaGomez/CNN-Medical-Imaging-Analysis

Chapter 2

2. Convolutional Neural Network for Medical Imaging Analysis

On this investigation, the main objective is to perform abnormality classification in mammography using Convolutional Neural Networks for Medical Imaging Analysis. This laboratory research will be development with a standard evaluation data set in the area of decision support systems in mammography, the *Digital Curated Breast Imaging Subset of Database for Screening Mammography* **(CBIS DDSM)**

2.1. Original Dataset

The dataset we will focus on is an updated and standardized version of the *Digital Database for Screening Mammography*[2] **(DDSM).** The DDSM is a database of 2,620 scanned film mammography studies. It contains normal, benign, and malignant cases with verified pathology information. Few well-curated public datasets have been provided for the mammography community. These include the DDSM, the Mammographic Imaging Analysis Society (MIAS) database, and the Image Retrieval in Medical Applications (IRMA) project. Although these public data sets are useful, they are limited in terms of data set size and accessibility.

The images have been decompressed and converted to DICOM format. Updated ROI segmentation and bounding boxes, and pathologic diagnosis for training data are also included. The data set contains 753 calcification cases and 891 mass cases, providing a data-set size capable of analyzing decision support systems in mammography.

In the subsequent sections, data source, data preprocessing, data augmentation, model development and evaluation will be delineated. A simple example of the image provided from the original dataset:

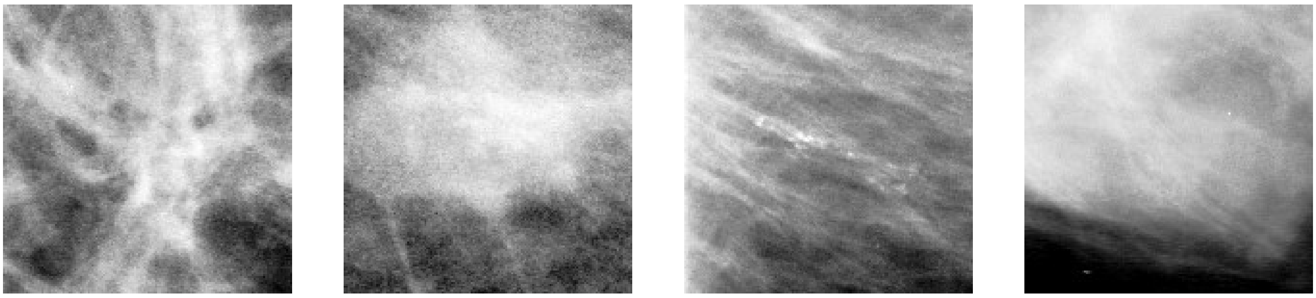
Table 1. Data Set DDSM description

Design Types	-Design Types and Parallel group design. -Feature extraction objective. -Image processing objective
Measurement Type	Mammography
Technology Type	Digital curation
Factor Type	Diagnosis
Sample Characteristic H	Homo sapiens

[2] Lee, Rebecca Sawyer, et al. 'A curated mammography data set for use in computer-aided detection and diagnosis research' Scientific data 4 (2017): 170177.

Considering the benefits of using deep learning in image classification problem (e.g., automatic feature extraction from raw data), develop a deep Convolutional Neural Network (CNN) that will be trained to read mammography images and classify them into the following five instances:

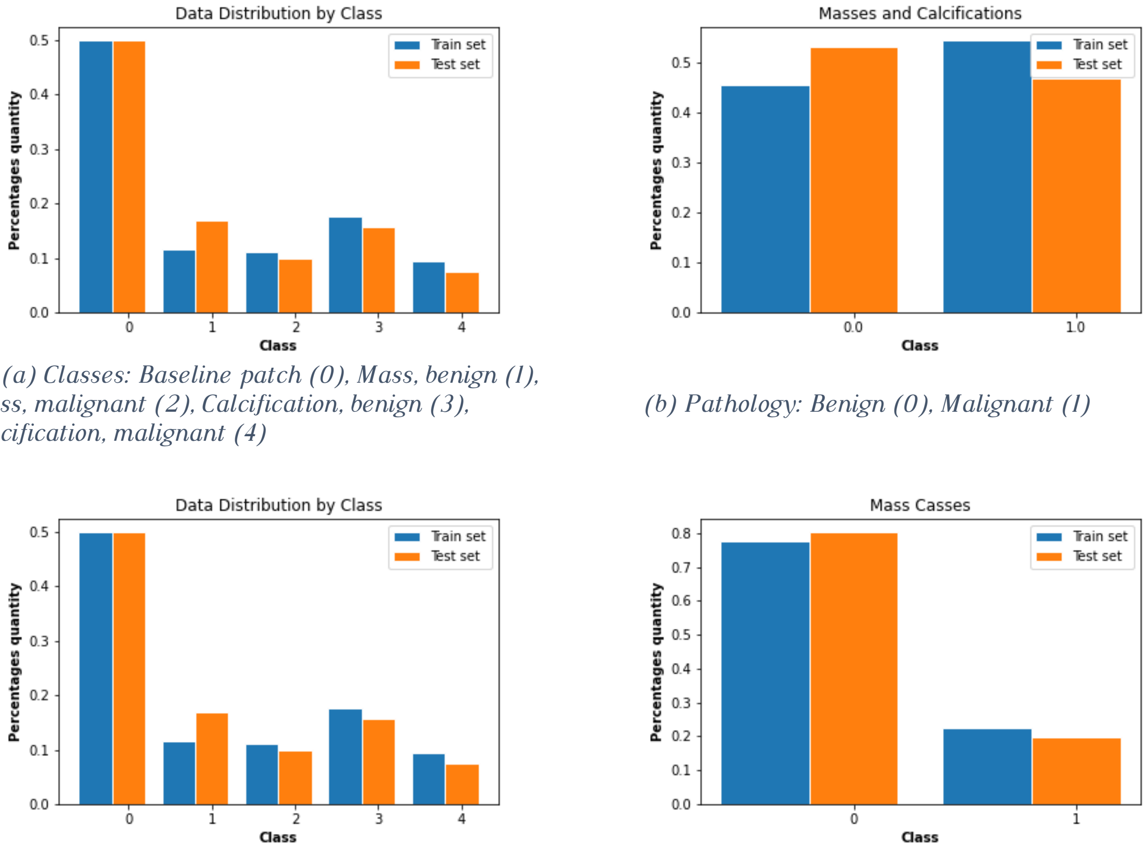
- Normal
- Benign Calcification
- Benign Mass
- Malignant Calcification
- Malignant Mass



Benign Mass Malignant Mass Benign Calcification Malignant Calcification

Figure 1. Medical Image Representation

The images are distributed at the baseline and abnormality level as NumPy arrays. A description of the dataset provided showing the distribution of training and test sets:



(a) Classes: Baseline patch (0), Mass, benign (1), Mass, malignant (2), Calcification, benign (3), Calcification, malignant (4)

(b) Pathology: Benign (0), Malignant (1)

(c) Pathology: Benign (0), Malignant (1)

(d) Pathology: Benign (0), Malignant (1)

Figure 2. CBIS DDSM *data Representation*

2.2. Paper Reference

The most relevant works and state-of-art techniques consulting for take inspiration to solve the project tasks comes from existing research works as following:

P. Xi, C. Shu and R. Goubran, "*Abnormality Detection in Mammography using Deep Convolutional Neural Networks,*" IEEE International Symposium on Medical Measurements and Applications (MeMeA), Rome, 2018, pp. 1-6, doi: 10.1109/MeMeA.2018.8438639

R. Agarwal, O. Diaz, X. Lladó, M. Hoon Yap, and R. Martí "*Automatic mass detection in mammograms using deep convolutional neural networks,*" Journal of Medical Imaging 6(3), 2019, doi: 10.1117/1.JMI.6.3.031409

M. K. Markey and A. Patel, "*Impact of missing data in training artificial neural networks for computer-aided diagnosis*," International Conference on Machine Learning and Applications, 2004. Proceedings, Louisville, KY, USA, 2004, pp. 351-354, doi: 10.1109/ICMLA.2004.1383534

R. Sawyer Leel, F. Gimenez, A. Hoogi, K. Kawai Miyake, M. Gorovoy and D. L. Rubin, "*A curated mammography data set for use in computer-aided detection and diagnosis research*", Sci Data, 2017, pp. 1-9, doi: 10.1038/sdata.2017.177

G. Carneiro, J. Nascimento, A.P. Bradley, "*Unregistered Multiview Mammogram Analysis with Pre-trained Deep Learning Models*.", Medical Image Computing and Computer-Assisted Intervention – MICCAI 2015. Lecture Notes in Computer Science, vol 9351. Springer, Cham, doi: 10.1007/978-3-319-24574-4_78

H. Chougrad, H. Zouaki, O. Alheyane, "*Deep Convolutional Neural Networks for breast cancer screening*", Computer Methods and Programs in Biomedicine, Volume 157, 2018, Pages 19-30, ISSN 0169-2607, doi: 10.1016/j.cmpb.2018.01.011

J. Y. Choi, D. H. Kim, K. N. Plataniotis, Y. M. Ro, "*Classifier ensemble generation and selection with multiple feature representations for classification applications in computer-aided detection and diagnosis on mammography,*" Expert Systems with Applications, Volume 46, 2016, Pages 106-121, ISSN 0957-4174, doi: 10.1016/j.eswa.2015.10.014

Reference Files

1 - Abnormality Detection in Mammoç

2 - Automatic mass detection in mammog

3 - Impact of missing data in training artifici

4 - A curated mammography data s

5 - Unregistered Multiview Mammogra

6 - Deep Convolutional Neural

7 - Classifier ensemble generation

2.2.1. Abnormality Detection in Mammography using Deep CNN

Abstract: To reduce the cost and workload of radiologists, it is proposed a computer aided detection approach for classifying and localizing calcifications and masses in mammogram images. To improve on conventional approaches, it is applied to deep convolutional neural networks (CNN) for automatic feature learning and classifier building. In computer-aided mammography, deep CNN classifiers cannot be trained directly on full mammogram images because of the loss of image details from resizing at input layers. Instead, on these classifiers are trained on labeled image patches and then adapted to work on full mammogram images for localizing the abnormalities. State-of-the-art deep convolutional neural networks are compared on their performance of classifying the abnormalities.

2.2.2. Automatic mass detection in mammograms using deep CNN

Abstract: The aim of this paper is to propose a patch-based CNN method for automated mass detection in full-field digital mammograms (FFDM). In addition to evaluating CNNs pretrained with the ImageNet dataset, the investigation on the use of transfer learning for a particular domain adaptation. First, the CNN is trained using a large public database of digitized mammograms (CBIS-DDSM dataset), and then the model is transferred and tested onto the smaller database of digital mammograms (InBreast dataset). It is evaluated three widely used CNNs (VGG16, ResNet50, InceptionV3) and show that the InceptionV3 obtains the best performance for classifying the mass and no mass breast region for CBIS-DDSM.

2.2.3. Impact of missing data in training ANN for CADx

Abstract: Artificial neural networks (ANN) are frequently used in the development of Computer-Aided Diagnosis systems for breast cancer detection and diagnosis. One class of models uses descriptions of mammographic lesions encoded following the BI-RADS lexicon. Data sets that have been carefully curated to ensure completeness are generally used; however, in routine practice, some information is typically missing in clinical databases. The impact of missing data on the performance of a feedforward, back-propagation ANN, as measured by the area under the Receiver Operating Characteristic curve, was found to be much higher when data were missing from the testing set than when data were missing from the training set. This empirical study highlights the need for additional research on developing robust clinical decision support systems for realistic environments in which key information may be unknown or inaccessible.

2.2.4. A curated mammography data set for use in CADe and CADx

Abstract: Published research results are difficult to replicate due to the lack of a standard evaluation data set in the area of decision support systems in mammography; most computer-aided diagnosis (CADx) and detection (CADe) algorithms for breast cancer in mammography are evaluated on private data sets or on unspecified subsets of public databases. This causes an inability to directly compare the performance of methods or to replicate prior results. This paper seeks to resolve this substantial challenge by releasing an updated and standardized version of the Digital Database for Screening Mammography (DDSM) for evaluation of future CADx and CADe systems (sometimes referred to generally as CAD) research in mammography.

2.2.5. Unregistered Multiview Mammogram Analysis with Pre-trained DL Models

Abstract: The main conclusions on this CNN model that are pre-trained using computer vision databases (e.g., ImageNet) are useful in medical image applications, despite the significant differences in image appearance. Second, this paper is shown that the Multi view classification is possible without the preregistration of the input images. Rather, used the high-level features produced by the CNNs trained in each view separately. Focusing on the classification of mammograms using craniocaudal (CC) and mediolateral oblique (MLO) views and their respective mass and micro-calcification segmentations of the same breast, we initially train a separate CNN model for each view and each segmentation map using an ImageNet pre-trained model. Then, using the features learned from each segmentation map and unregistered views, it is training a final CNN classifier that estimates the patient's risk of developing breast cancer using the Breast Imaging-Reporting and Data System (BI-RADS) score.

2.2.6. Deep CNN for breast cancer screening

Abstract: In this paper is developed a Computer-aided Diagnosis (CAD) system based on deep Convolutional Neural Networks (CNN) that aims to help the radiologist classify mammography mass lesions. Deep learning usually requires large datasets to train networks of a certain depth from scratch. Transfer learning is an effective method to deal with relatively small datasets as in the case of medical images, although it can be tricky as it can easily start overfitting. In this work, it is explored the importance of transfer learning and it is experimentally determining the best fine-tuning strategy to adopt when training a CNN model.

2.2.7. Classifier ensemble with multiple feature representations for classification applications in CADe and CADx on mammography

Abstract: This paper presents a novel ensemble classifier framework for improved classification of mammographic lesions in Computer-aided Detection (CADe) and Diagnosis (CADx) systems. Compared to previously developed classification techniques in mammography, the main novelty of proposed method is twofold: (1) the "combined use" of different feature representations (of the same instance) and data resampling to generate more diverse and accurate base classifiers as ensemble members and (2) the incorporation of a novel "ensemble selection" mechanism to further maximize the overall classification performance. In addition, as opposed to conventional ensemble learning, this proposed ensemble framework has the advantage of working well with both weak and strong classifiers, extensively used in mammography CADe and/or CADx systems. Extensive experiments have been performed using benchmark mammogram dataset to test the proposed method on two classification applications: (1) false-positive (FP) reduction using classification between masses and normal tissues, and (2) diagnosis using classification between malignant and benign masses.

Chapter 3

3. Task 2: CNN from Scratch

The purpose of this task is to develop a Computer-Aided Detection and Diagnosis (CAD) system that aim is to assist in the detection and/or diagnosis of diseases through a "second opinion". CAD systems are classified into two groups: Computer-Aided Detection (CADe) systems and Computer-Aided Diagnosis (CADx) systems.

On first section: Development of a classification model for discriminating between Masses and Calcification class, we are going to develop a CADe system that geared for the location of lesions in medical images by exploiting an ad-hoc (from Scratch) designed and implemented CNN architectures. Moreover, on the second section: Development model classification for discriminating between Benign class and Malignant class, we are going to develop a CADx system perform the characterization of the lesion, on our case, the distinction between benign and malignant calcifications and masses by exploiting an ad-hoc (from Scratch) designed and implemented CNN architectures.

Which is the input data?

By visual inspection of the grid of images below you can try to detect differences between **benign masses examples** (Figure 3) against **malignant masses examples** (Figure 4). Also, try the same exercise to inspect the grid of images below trying to detect differences between **benign calcification examples** (Figure 5) against **malign calcification examples** (Figure 6).

As it can be noticed by comparing the images, **the task is challenging** as it is hard to detect significant differences, at least for a non physician. We are going to start with CADe system; masses and calcifications classification. Then, we are going to develop the CADx system for the diagnosis of malignant and benign lesion. Masses are very different from calcifications, which converts it easy to the system for the respective identification, but on the other hand, it is hard to diagnose whether you are dealing with a benign or malignant case considering both abnormality types. The problem we are going to face is a binary classification: we train deep learning models over labeled image samples.

Benign masses examples:

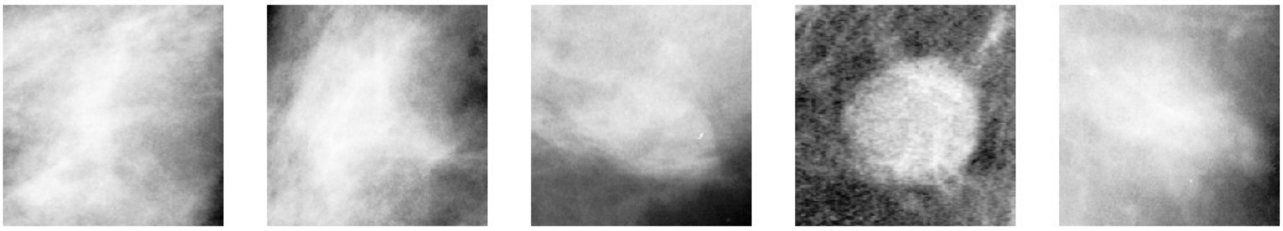


Figure 3. Benign masses Random examples

Malignant masses examples:

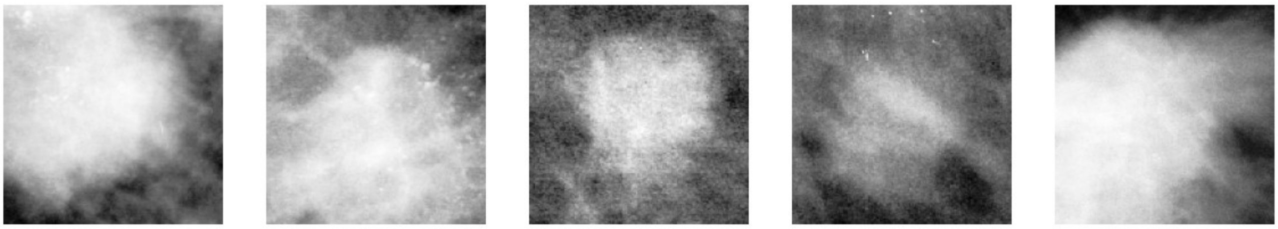


Figure 4. Malignant masses Random examples

Benign calcification examples:

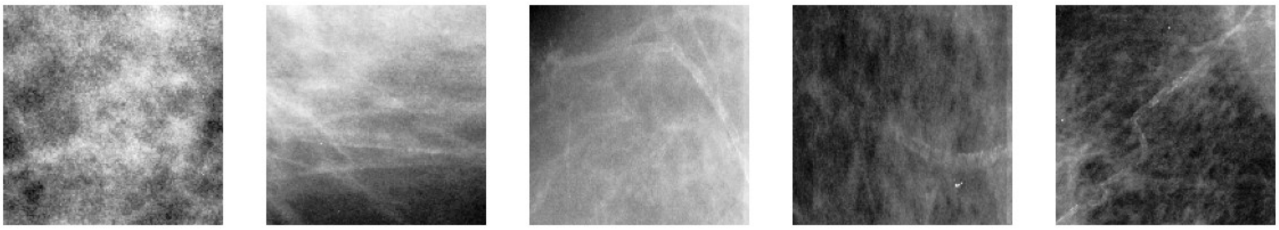


Figure 5. Benign calcifications Random examples

Malignant calcification examples

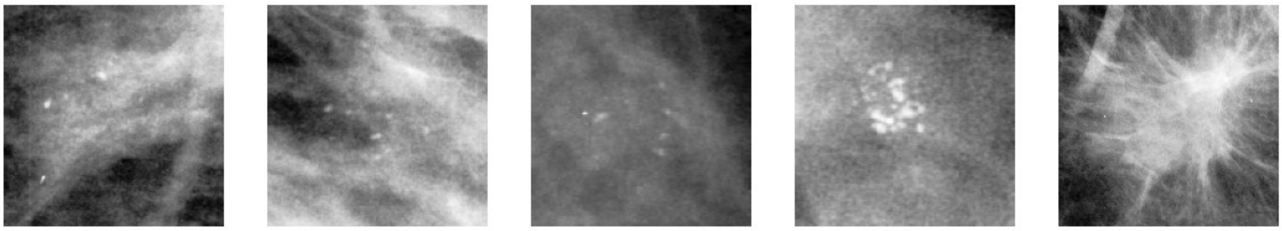


Figure 6. Malignant calcifications Random examples

3.1. Development of a classification model for discriminating between Masses and Calcification class

3.1.1. Data Loading

As a first step we need to prepare data in a way that can be fed to our machine learning models. First of all, we loaded data into four NumPy tensors: two NumPy tensors holding training images data and training images labels and other two holding test images and test labels:

Table 2. Classes distribution of images Train set

	Benign	Malignant	Total
Train Masses	*620*	*598*	*1218*
Train Calcification	*948*	*510*	*1458*
Total	*1568*	*1108*	*2676*

The masses class contains 1218 samples while the calcification class contains 1458 samples, the classes are quite balanced.

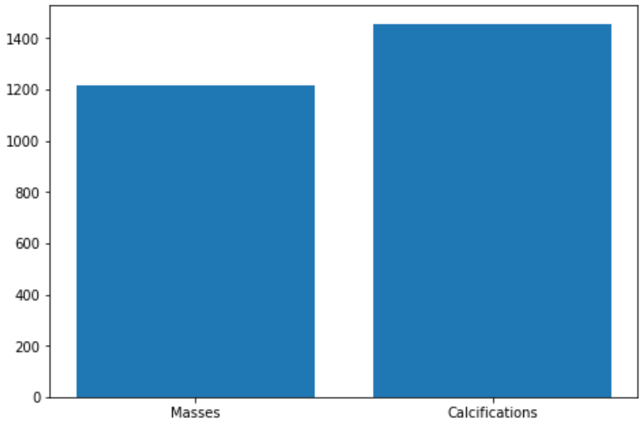


Figure 7. Masses vs Calcifications

3.1.2. Data Normalization

Computer vision usually requires relatively little of this kind of preprocessing. The images should be standardized, formatting images to have the same scale is the only kind of preprocessing that is strictly necessary. As optional, we add dataset augmentation because is an excellent way to reduce the generalization error of most computer vision models.

Since original tensors are of type *uint16* with values ranging in [0, 65335], we needed to rescale these to smaller values for training the NNs: we **rescaled in the range [0, 1]**

Subsequently, we divided our images dataset into **training set and validation set** according to the percentages **80%** and **20%**.

3.1.3. Choosing a measure of success

Being our classes balanced, we decided to compare models' performances by using the **accuracy values**. **Additionally**, for each model we computed[3]

- **Accuracy**
- **Precision**
- **Recall**
- **F-Score**
- **AUC**

3.1.4. Using Data Augmentation

Deep Learning models perform better when we have large datasets at our disposal. One very popular way to make our datasets bigger is data augmentation. Since **our dataset is quite small**, we adopted data augmentation as **help to prevent overfitting** so as to try to obtain a model which can generalize better.

We augmented images samples according to the **following random transformations**:

- Rotation in the range [0°, 40°];
- Width shift by a fraction of 0.2 w.r.t. the total width of the image;
- Height shift by a fraction of 0.2 w.r.t. the total height of the image;
- Shear range [0, 20];
- Zoom range [0, 0.2];
- Horizontal flip set to True;
- "Nearest" as fill mode strategy for filling in the newly created pixels which can appear after a rotation or a width/height shift.

3.1.5. Choosing Network Architecture: Baseline Model

As starting point we developed **a first CNN model** in order to get a first feel by checking how much better the model did with respect to random guessing: we wanted first of all to know whether it was possible to achieve statistical power in solving our task.

We tried out several CNN architectures, starting with relatively few layers and parameters and increasing time by time, in order to find a proper sized network: the best possible compromise between having too much capacity and not enough capacity. In the end we choose the architecture reported in the diagram below:

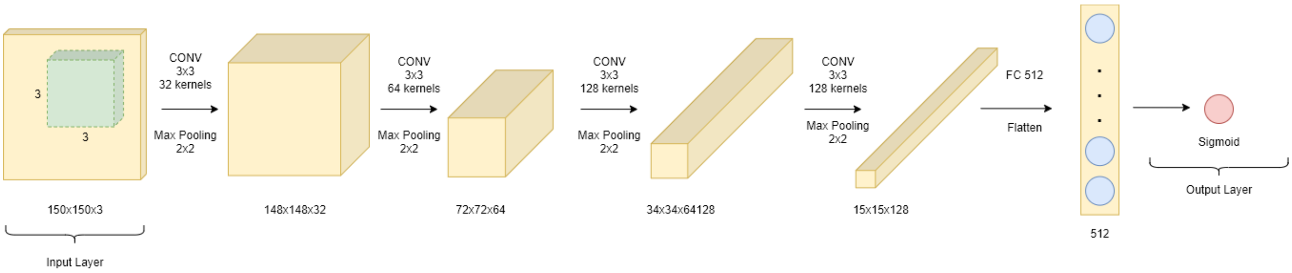
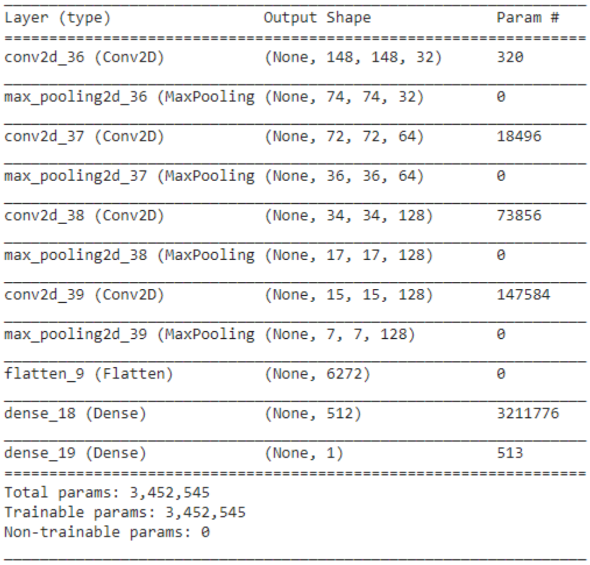


Figure 8. Network Architecture Chosen

Table 3. Configuration CNN Architecture Table

Loss Function	Optimizer	Learning Rate	Batch Size
Binary Cross entropy	Adam	0.0001	20



```
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
conv2d_36 (Conv2D)           (None, 148, 148, 32)      320
_________________________________________________________________
max_pooling2d_36 (MaxPooling (None, 74, 74, 32)        0
_________________________________________________________________
conv2d_37 (Conv2D)           (None, 72, 72, 64)        18496
_________________________________________________________________
max_pooling2d_37 (MaxPooling (None, 36, 36, 64)        0
_________________________________________________________________
conv2d_38 (Conv2D)           (None, 34, 34, 128)       73856
_________________________________________________________________
max_pooling2d_38 (MaxPooling (None, 17, 17, 128)       0
_________________________________________________________________
conv2d_39 (Conv2D)           (None, 15, 15, 128)       147584
_________________________________________________________________
max_pooling2d_39 (MaxPooling (None, 7, 7, 128)         0
_________________________________________________________________
flatten_9 (Flatten)          (None, 6272)              0
_________________________________________________________________
dense_18 (Dense)             (None, 512)               3211776
_________________________________________________________________
dense_19 (Dense)             (None, 1)                 513
=================================================================
Total params: 3,452,545
Trainable params: 3,452,545
Non-trainable params: 0
_________________________________________________________________
```

Table 4. Result table. Choosing Network Architecture: Baseline Model

Class	Precision	Recall	F-score
Mass	0.82	0.84	0.83
Calcification	0.81	0.78	0.80

Table 5. Result table. Extra computed values. Choosing Network Architecture: Baseline Model

AUC	TEST ACC
0.895	82%

3.1.6. Approaches Summary

3.1.6.1. Model Nr. 1: Majority Class Under sampling

Since masses and calcification classes are slightly unbalanced, we tried to under sample the calcifications class so as to have both the classes represented by exactly the same number of samples, 1218:

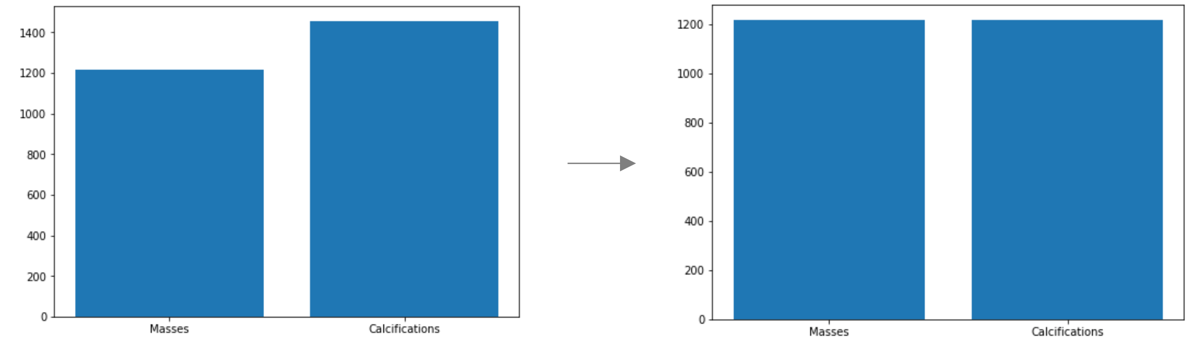


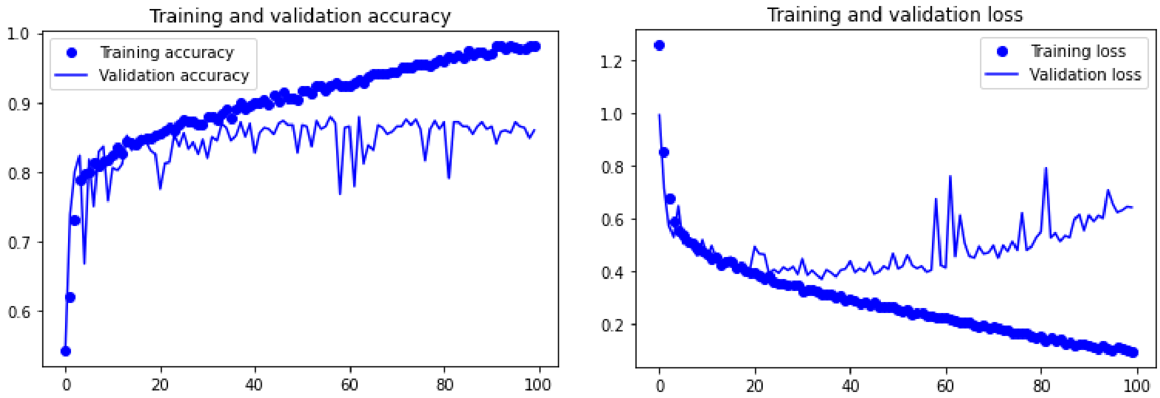
Table 6 Result table. Majority Class Under sampling

Class	Precision	Recall	F-score
Mass	0.88	0.85	0.86
Calcification	0.83	0.87	0.85

Table 7. Result table. Extra computed values.

AUC	TEST ACC
0.942	84%

With this trial we improved results with respect to the baseline model: test accuracy increased from 82% to 86%, AUC from 0.895 to 0.942.



3.1.6.2. Fighting Overfitting: Model Nr. 2

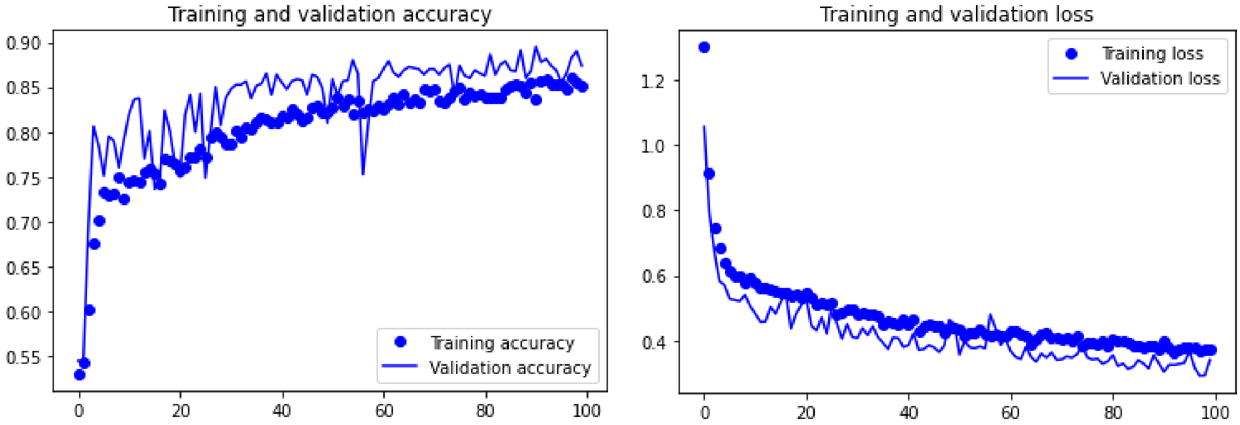
Our previous models are clearly affected by overfitting, in order to try to prevent this we exploited data augmentation. The random images transformations are those previously introduced.

Table 8 Result table. Majority Class Undersampling

Class	Precision	Recall	F-score
Mass	0.89	0.80	0.84
Calcification	0.80	0.89	0.84

Table 9. Result table. Extra computed values.

AUC	TEST ACC
0.93	84%



With this trial, although by observing the plots above reporting the training results you can notice that there is no more overfitting, the performances did not improve with respect to the previous model, we obtained no meaningful differences.

3.1.6.3. Hyperparameters Tuning throw Grid Search

In order to find out the best set of hyperparameters for our CNN, we carried out a grid search over these parameters grid:

PARAMETER	SET OF VALUES
Batch Size	[20, 32, 64, 128]
Learning Rates	[1e-2, 1e-3, 1e-4]
Units Per Layer	{[32, 64, 128, 128], [32, 64, 128, 256]}
Number of Epochs	100
Optimizers	[Adam, RMSProp]
Num of folds Crossvalidation	5

Thus, we carried out a total of 48 with 5-folds-CV saving all results in a csv file.

3.1.7. Top 3 Models Evaluation on Test Set

After sorting the training results obtained from the grid search by deceasing validation accuracy value, we picked the top 3 models in order to train each of these one more time on all the available data. Subsequently, we computed the accuracy of these models on the test set, **obtaining the following results**:

BATCH SIZE	OPTIMIZER	TEST LOSS	TEST ACC	PRECISION	RECALL	AUC
20	RMSProp	0.7444	0.8313	0.8519	0.7718	0.9022
20	ADAM	0.8893	0.8031	0.7829	0.7987	0.8760
20	ADAM	0.7338	0.8156	0.8156	0.8255	0.8803

3.2. Development model classification for discriminating between Benign class and Malignant class

3.2.1. Data Loading and Preprocessing

As a first step we need to prepare data in a way that can be fed to our machine learning models. First of all, we loaded data into four NumPy tensors: two NumPy tensors holding training images data and training images labels and other two holding test images and test labels:

Table 10. Classes distribution of images Train set

	Benign	Malignant	Total
Train Masses	*620*	*598*	*1218*
Train Calcification	*948*	*510*	*1458*
Total	*1568*	*1108*	*2676*

As it can be noticed, benign and malign classes are not exactly balanced we have **1568** benign samples against **1108** malignant samples.

3.2.2. Choosing a measure of success

Since we are dealing with a medical task aiming at classifying benign versus malignant breast cancer cases, and being the classes slightly unbalanced we chose not to adopt accuracy to compare model performances. The main idea is that **the misclassification costs for the two classes has not an equal weight**: misclassifying a malignant case as benign is much more dangerous with respect to misclassifying a benign case as malignant. In the former case, by carrying out further medical tests the error would be revealed immediately and the patient would be immediately relieved. Conversely, in the latter case the patient would risk his life.

This said, our aim is to reach the **best possible malignant class recall** value **while keeping the False Positive Rate as low as possible**. Therefore, we decided to evaluate our models by **giving higher priority** to the following three measures:

- **TPR:** True Positive Rate (Malignant classified samples which actually are malignant)
- **FPR:** False Positive Rate (Malignant classified samples which actually are benign)
- **AUC:** Area Under the ROC Curve

Additionally, for each model we computed[4]

- Accuracy:
  - $\frac{TP+TN}{TP+TN+FP+FN}$
- Precision:
  - $\frac{TP}{TP+FP}$
- F-Score:
  - $\frac{TP}{TP+FN}$

[4] Where: TP = True positive; FP = False positive; TN = True negative; FN = False negative

3.2.3. Using Data Augmentation

Deep Learning models perform better when we have large datasets at our disposal. One very popular way to make our datasets bigger is data augmentation. Since **our dataset is quite small**, we adopted data augmentation as **help to prevent overfitting** so as to try to obtain a model which can generalize better.

We augmented images samples according to the **following random transformations**:

- Rotation in the range [0°, 40°];
- Width shift by a fraction of 0.2 w.r.t. the total width of the image;
- Height shift by a fraction of 0.2 w.r.t. the total height of the image;
- Shear range [0, 20];
- Zoom range [0, 0.2];
- Horizontal flip set to True;
- "Nearest" as fill mode strategy for filling in the newly created pixels which can appear after a rotation or a width/height shift.

3.2.4. Approaches Summary

3.2.4.1. Choosing Network Architecture: Baseline Model

As starting point we developed a first CNN model in order to get a first feel by checking how much better the model did with respect to random guessing: we wanted first of all to know whether it was possible to achieve statistical power in solving our task.

We tried out several CNN architectures, starting with relatively few layers and parameters and increasing time by time, in order to find a proper sized network: the best possible compromise between having too much capacity and not enough capacity. In the end we choose the architecture reported in the diagram below:

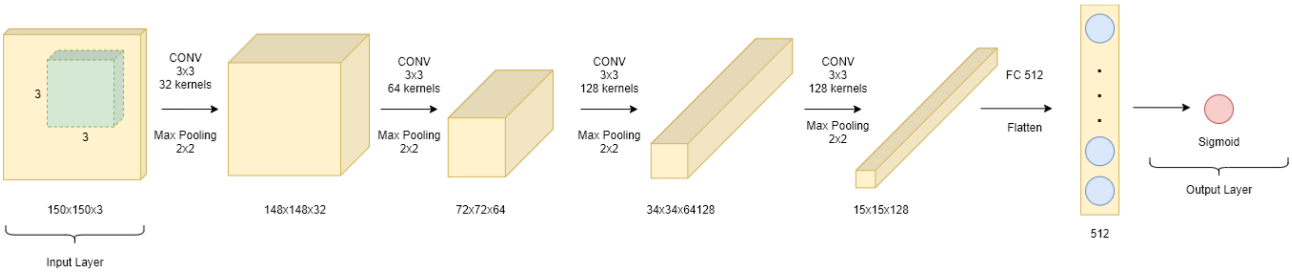
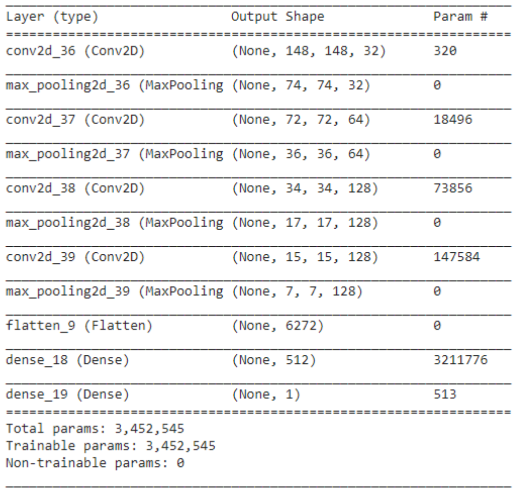


Figure 9. Network Architecture Baseline

Table 11. Configuration CNN Architecture Table

Loss Function	Optimizer	Learning Rate	Batch Size
Binary Cross entropy	Adam	0.0001	20



```
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
conv2d_36 (Conv2D)           (None, 148, 148, 32)      320
_________________________________________________________________
max_pooling2d_36 (MaxPooling (None, 74, 74, 32)        0
_________________________________________________________________
conv2d_37 (Conv2D)           (None, 72, 72, 64)        18496
_________________________________________________________________
max_pooling2d_37 (MaxPooling (None, 36, 36, 64)        0
_________________________________________________________________
conv2d_38 (Conv2D)           (None, 34, 34, 128)       73856
_________________________________________________________________
max_pooling2d_38 (MaxPooling (None, 17, 17, 128)       0
_________________________________________________________________
conv2d_39 (Conv2D)           (None, 15, 15, 128)       147584
_________________________________________________________________
max_pooling2d_39 (MaxPooling (None, 7, 7, 128)         0
_________________________________________________________________
flatten_9 (Flatten)          (None, 6272)              0
_________________________________________________________________
dense_18 (Dense)             (None, 512)               3211776
_________________________________________________________________
dense_19 (Dense)             (None, 1)                 513
=================================================================
Total params: 3,452,545
Trainable params: 3,452,545
Non-trainable params: 0
_________________________________________________________________
```

Figure 10. Architecture of the CNN Malignant and Benign from scratch

We report below the results obtained so as to use this as **baseline results**: this is our starting point, let us see by how much we can improve throughout the next trials.

Table 12. Result table. Choosing Network Architecture: Baseline Model

Class	Precision	Recall	F-score
Benign	0.72	0.76	0.74
Malign	0.50	0.45	0.48

Table 13. Result table. Extra computed values. Choosing Network Architecture: Baseline Model

TPR	FPR	AUC	TEST ACC
45	23	0.662	61%

3.2.4.2. Weighted Class Approach without data augmentation

To deal with the slight skew in the distribution of classes we tried to assign different weights to the majority and to the minority classes. The difference in weights will influence the classification the classification of the samples during the training phase of the network. Our aim is to penalize the misclassification made for the minority class by assigning it a higher weight w.r.t. the majority class.

Specifically, the weight values have been assigned to the 2 classes inversely proportional to their respective frequencies: 0.86 to benign class and 1.2 to the malign class.

Below are reported the results obtained:

Table 14 Result table. Weighted Class Approach without data augmentation

Class	Precision	Recall	F-score
Benign	0.71	0.62	0.66
Malign	0.42	0.52	0.47

Table 15. Result table. Extra computed values. Weighted Class Approach without data augmentation

TPR	FPR	AUC	TEST ACC
52	37	0.606	59%

3.2.4.3. Majority Class Under-sampling without data augmentation

In this trial, we under sampled the majority class so as to have the same number of samples in both the classes: 1108.

Below are reported the results obtained:

Table 16. Result table. Majority Class Under-sampling without data augmentation

Class	Precision	Recall	F-score
Benign	0.78	0.43	0.55
Malign	0.42	0.78	0.55

Table 17. Result table. Extra computed values. Majority Class Under-sampling without data augmentation

TPR	FPR	AUC	TEST ACC
78	57	0.606	55%

3.2.4.4. Using Data Augmentation

The first trials were affected by a clear overfitting. We adopted data augmentation as previously explained. With data augmentation, the results obtained improved significantly.

Below are reported the results obtained:

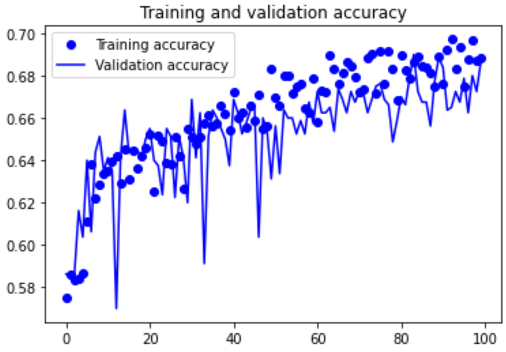


Figure 11. Model Training validation Accuracy Using Data Augmentation

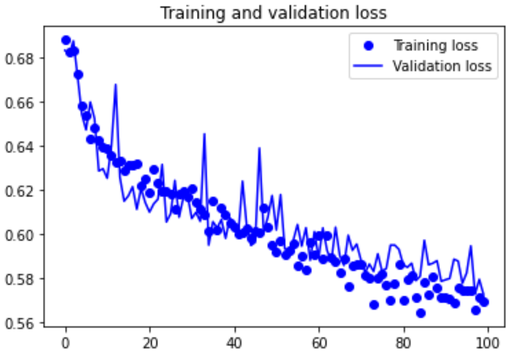


Figure 12. Model Training validation Loss Using Data Augmentation

Table 18. Result table. Majority Class Under-sampling using data augmentation

Class	Precision	Recall	F-score
Benign	0.77	0.72	0.75
Malign	0.54	0.61	0.57

Table 19. Result table. Extra computed values. Majority Class Under-sampling using data augmentation

TPR	FPR	AUC	TEST ACC
60	27	0.706	68%

3.2.4.5. Majority Class Under-sampling and Data Augmentation

In this trial beside under sampling the majority class as in model nr. 2, we exploited data augmentation.

Table 20. Result table. Majority Class Under-sampling without data augmentation

Class	Precision	Recall	F-score
Benign	0.76	0.46	0.57
Malign	0.42	0.73	0.53

Table 21. Result table. Extra computed values. Majority Class Under-sampling without data augmentation

TPR	FPR	AUC	TEST ACC
72	54	0.649	55%

3.2.4.6. Minority Class Oversampling and Data Augmentation

In this trial beside oversampling the minority class through random sampling with replacement, we exploited data augmentation.

Table 22. Result table. Minority Class Oversampling & Data Augmentation

Class	Precision	Recall	F-score
Benign	0.84	0.33	0.48
Malign	0.41	0.88	0.56

Table 23. Result table. Extra computed values. Minority Class Oversampling & Data Augmentation

TPR	FPR	AUC	TEST ACC
88	66	0.664	52%

3.2.4.7. Minority Class Oversampling through Data Augmentation

The main idea of this trial is to oversample the malign class by mean of data augmentation. The purpose is to augment the number of malignant samples with respect to the number of benign samples by a prespecified scale ratio, in such a way to try to obtain a model which does better in the classification of the malign class.

$$scale\ ratio = \frac{\#\ of\ malign\ samples}{\#\ of\ benign\ samples}$$

We tried out different scale ratio values, unfortunately even though the malignant class recall improved, the overall performances did not improve very much. Below you can see the performances obtained by using scale ratio of 1.35:

Table 24. Result table. Minority Class Oversampling through Data Augmentation

Class	Precision	Recall	F-score
Benign	0.77	0.57	0.65
Malign	0.46	0.68	0.55

Table 25. Result table. Extra computed values. Minority Class Oversampling through Data Augmentation

TPR	FPR	AUC	TEST ACC
68	43	0.685	61%

3.2.4.8. Oversampling though SMOTE & Data Augmentation

In this trial we exploited the Synthetic Minority Oversampling Technique so as to have the two classes having the same number of samples. Unfortunately, neither this approach provided much better results, we briefly report what we obtained below:

Table 26 Result table. Oversampling though SMOTE & Data Augmentation

Class	Precision	Recall	F-score
Benign	0.75	0.50	0.60
Malign	0.42	0.68	52

Table 27. Result table. Extra computed values. Oversampling though SMOTE & Data Augmentation

TPR	FPR	AUC	TEST ACC
68	50	0.638	52%

3.2.5. Models results comparison

The purpose of this section is to compare all the previous results of the tested models for make a Discussion about the determination of the best model from Scratch.

Table 28. Classes distribution of images Test set

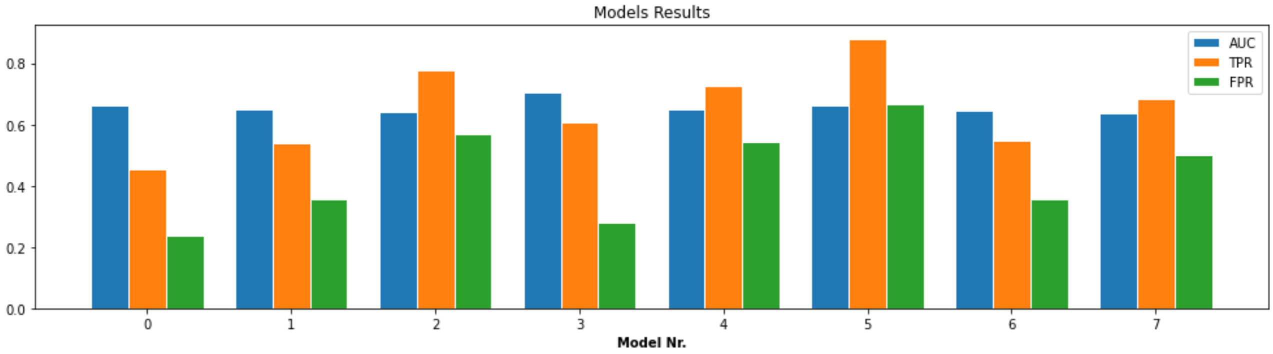


Table 29. Comparison results Benign and Malignant from Scratch

Model Nr.	TPR	FPR	TPR - FPR	AUC	TEST ACC
5	88%	67%	21	0.664	52%
2	78%	57%	21	0.643	55%
4	73%	54%	19	0.649	55%
6	68%	43%	25	0.685	61%
7	68%	50%	18	0.638	56%
3	61%	28%	33	0.706	68%
1	54%	35%	19	0.649	61%
0	45%	23%	22	0.662	65%

🟨 Model having the highest True Positive Rate

🟩 Best Model

⬜ Baseline Model (Worst)

Chapter 4

4. Pretrained CNN

4.1. Development of a classification model for discriminating between Masses class and Calcifications class using pre-trained architectures

In this task we imported the following CNN: "*VGG16, ResNet50V2, InceptionV3*", without the output layer to perform finetuning. Each one of them has been finetuned adding three different set of fully connected layers after the conv_base:

- *1 x Dense(256):* One layer with Dense(256)
- *3x [DropOut(0.5)+Dense(256)]:* Three layers with dropout layer with a fraction of the 0.5 input units to drop and Dense(256)
- *5x [DropOut(0.5)+Dense(256)]*: Five layers with dropout layer with a fraction of the 0.5 input units to drop and Dense(256)

Firstly, we analyzed the result using every possible combination of Data Augmentation and Global Contrast Normalization, adding only a Flatten / Dense(256) pair: We noted that using the GCN the performances drop considerably and that disabling the Data Augmentation would cause always overfitting in the networks.

We decided then to conduct all the successive experiments with Data Augmentation = True, and GCN = False. After that we computed some trial maintaining the same "***Dense layer architecture***" and changing the Flatten() layer with the ***GlobalAveragePooling2D***, which produce less features than the flatten.

The results were generally worse for the ***GlobalAveragePooling*** than the Flatten, so we used the results obtained with the flatten layer. In this problem we considered the test accuracy as the comparison metrics between the different architectures, because in this case both classes have the same importance, the number of samples is fairly balanced and the difference in misclassifications on the test set is small.

Table 30. Comparison Table Pretrained CNN. Mases and Calcifications

	0 Fine Tuning	1 Fine Tuning	2 Fine Tuning	3 Fine Tuning	All Fine Tuning
InceptionV3					
x1 [DropOut + Dense]	84.06	89.38	86.56	87.19	88.13
x3 [DropOut + Dense]	83.13	87.81	84.06	85.94	89.06
x5 [DropOut + Dense]	83.75	88.13	83.13	85.94	85.62
ResNet50V2					
x1 [DropOut + Dense]	88.44	89.39	90	89.39	87.50
x3 [DropOut + Dense]	85	89.69	86.56	88.13	90
x5 [DropOut + Dense]	85.94	88.44	85.62	87.50	88.44
VGG16					
x1 [DropOut + Dense]	83.44	84.38	86.87	87.50	89.69
x3 [DropOut + Dense]	70.31	86.25	86.56	88.75	89.06
x5 [DropOut + Dense]	82.81	89.06	87.19	85.62	85.62

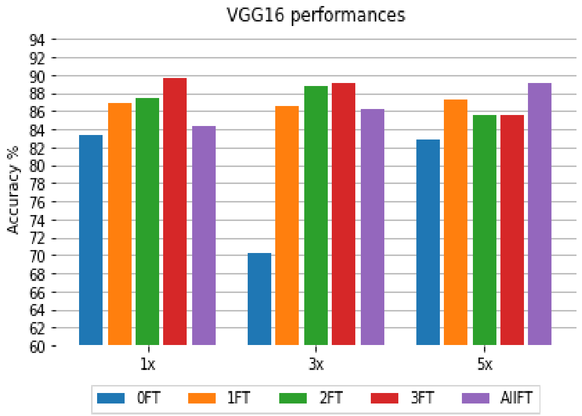


Figure 13. VGG16 Performance Masses and Calcifications

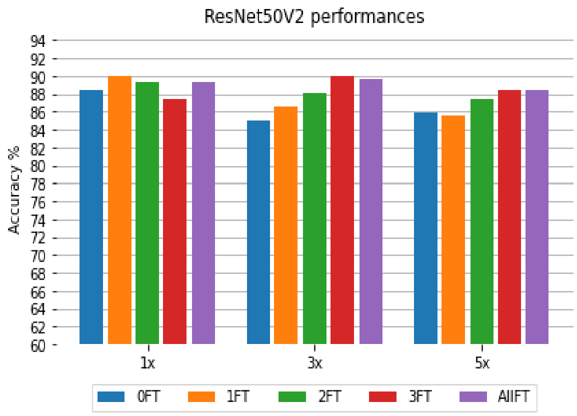


Figure 14. RestNet Version 2 Performances Masses and Calcifications

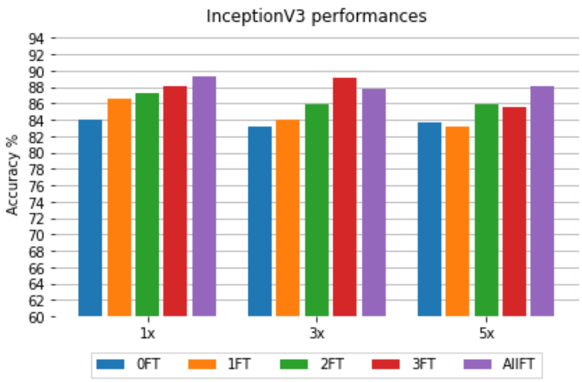
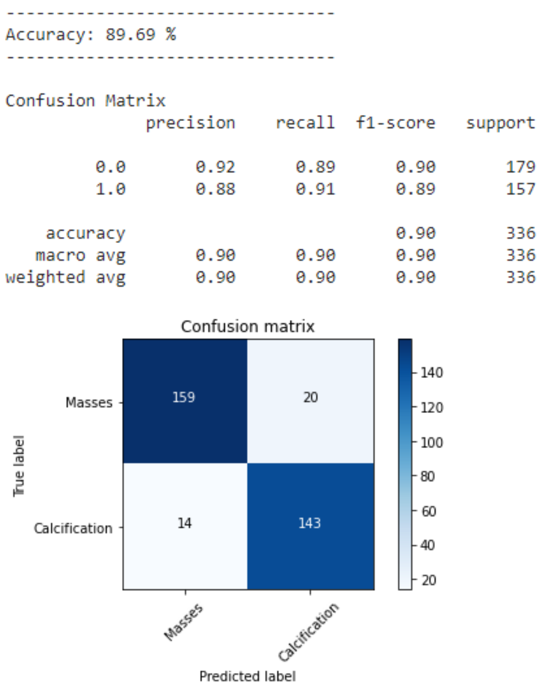


Figure 15 Inception Version 3 Performance Masses and Calcifications

As we can see from the table and the histograms the performances are fairly similar between the different architectures, many are apart only by 1-2% of test accuracy, so defining a definite “Best Model” is not easy. We can note that the “all Trainable” finetuned versions are generally among the best results (w.r.t.) the relative CNN used.



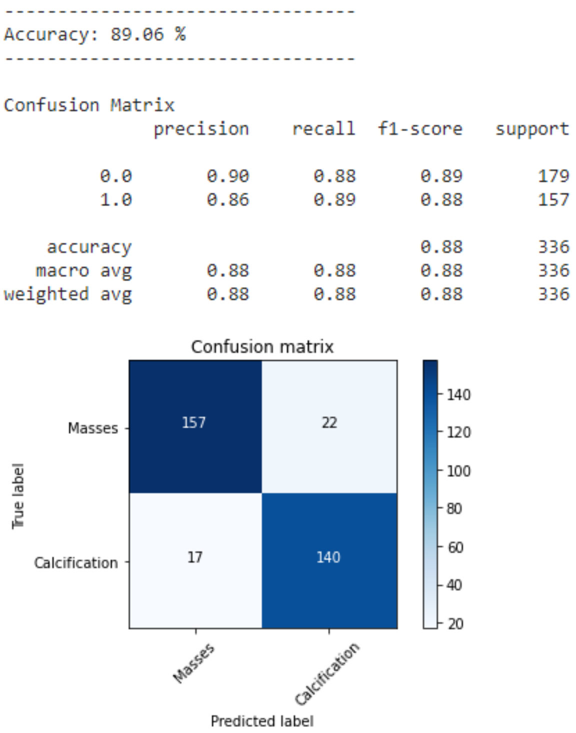
```
--------------------------------
Accuracy: 89.69 %
--------------------------------

Confusion Matrix
              precision    recall  f1-score   support

         0.0       0.92      0.89      0.90       179
         1.0       0.88      0.91      0.89       157

    accuracy                           0.90       336
   macro avg       0.90      0.90      0.90       336
weighted avg       0.90      0.90      0.90       336
```

*Figure 16. ResNet50V2 – Flatten + 3x [DropOut + Dense(256)] – **All Trainable***



```
--------------------------------
Accuracy: 89.06 %
--------------------------------

Confusion Matrix
              precision    recall  f1-score   support

         0.0       0.90      0.88      0.89       179
         1.0       0.86      0.89      0.88       157

    accuracy                           0.88       336
   macro avg       0.88      0.88      0.88       336
weighted avg       0.88      0.88      0.88       336
```

*Figure 17. InceptionV3-Flatten + 3x [DropOut + Dense(256)] – **Last 3 Blocks Trainable***

As we can see in the two confusion matrixes reported generally the prediction of the values tend to classify a bit more “masses as calcifications” than “calcifications as masses”, but the gap is a few samples.

4.2. Development of a classification model for discriminating between Benign class and Malignant class using pre-trained architectures

The purpose of this task is to develop a computer-aided diagnosis (CADx) system by **exploiting transfer learning**. Transfer learning is an effective method to deal with relatively small datasets as in the case of medical images, although it can be tricky as we can easily start overfitting. In this task we tried **different fine-tuning strategies** applied to the most powerful state of the art CNN architectures for computer vision: **VGG16**, **ResNet50V2** and **InceptionV3**.

We managed to improve results with respect to using from scratch CNN architectures: our best performing model achieved a **test accuracy** value of **75%**, and **0.797 AUC** with a **TPR** value of **73%** and a **FPR** value of **25%**. Although it is not an excellent result, it is a significant improvement w.r.t. from scratch CNNs.

4.2.1. Global Contrast Normalization (GCN)

As an additional pre-processing step, we tried to adopt the global contrast normalization technique as H. Chourag et. al did in their work.

Normalization, also called zero-centering is a standard step in medical image classification. It attempts to deal with external sources of data variation like illumination levels, the different scanners used in the digitalization process and how this can affect pixel values. GCN computes the mean of intensities for each image, and then subtracts it from each pixel of the image.

Let $x_{i,j}$ be the tensor of an image ($x \in R^{r\,x\,r}$) and $x_{avg} = \frac{1}{r^2}\sum_{i,j} x_{i,j}$, the tensor of the normalized image is $x'_{i,j} = x_{i,j} - x_{avg}$.

However, when we applied GCN as pre-preprocessing step, the CNNs were biased to misclassify a large number of malignant samples as benign, thus we abandoned this idea.

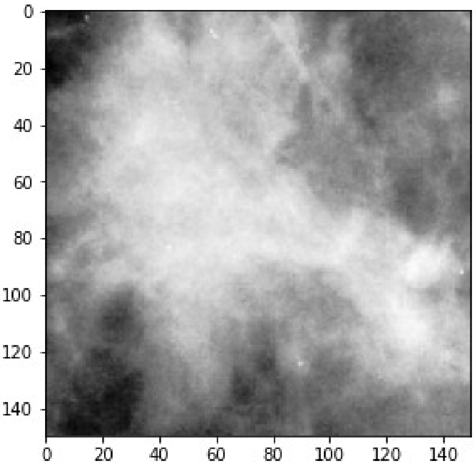


Figure 18. Original Sample

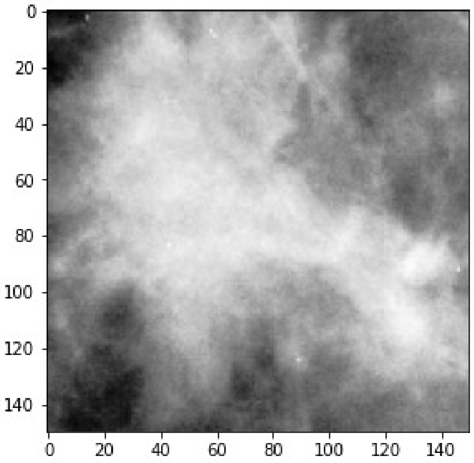


Figure 19. GCN Sample

4.2.2. Using Data Augmentation

Deep Learning models perform better when we have large datasets at our disposal. One very popular way to make our datasets bigger is data augmentation. Since **our dataset is quite small**, we adopted data augmentation as **help to prevent overfitting** so as to try to obtain a model which can generalize better.

We augmented images samples according to the **following random transformations** as *Chourag et. al* did in their work:

- **Rotation** in the range [0°, 40°];
- **Width shift** by a fraction of 0.25 w.r.t. the total width of the image;
- **Height shift** by a fraction of 0.25 wr.t. the total height of the image;
- **Shear range** [0, 20];
- **Zoom range** [0.5, 1.5];
- **Horizontal flip** set to **True**;
- "**Nearest**" as **fill mode** strategy for filling in the newly created pixels which can appear after a rotation or a width/height shift;

4.2.3. Choosing a measure of success

Since we are dealing with a medical task aiming at classifying benign versus malignant breast cancer cases, and being the classes slightly unbalanced we chose not to adopt accuracy to compare model performances. The main idea is that the misclassification costs for the two classes has not an equal weight: misclassifying a malignant case as benign is much more dangerous with respect to misclassifying a benign case as malignant. In the former case, by carrying out further medical tests the error would be revealed immediately and the patient would be immediately relieved. Conversely, in the latter case the patient would risk his life.

This said, our aim is to reach the best possible malignant class recall value while keeping the False Positive Rate as low as possible. Therefore, we decided to evaluate our models by giving higher priority to the following three measures:

- **TPR:** True Positive Rate (Malignant classified samples which actually are malignant)
- **FPR:** False Positive Rate (Malignant classified samples which actually are benign)
- **AUC:** Area Under the ROC Curve

Additionally, for each model we computed[5]

- **Accuracy:**
  - $\frac{TP+TN}{TP+TN+FP+FN}$
- **Precision:**
  - $\frac{TP}{TP+FP}$
- **F-Score:**
  - $\frac{TP}{TP+FN}$

4.2.4. Transfer learning and fine-tuning

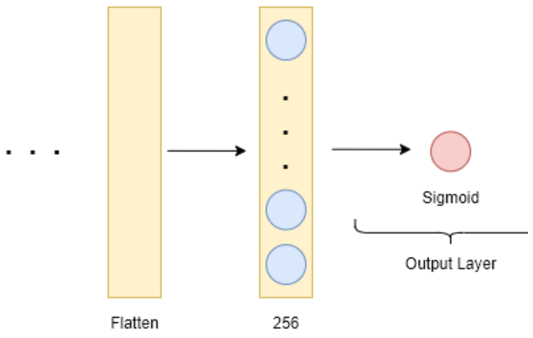
In this task we adopted as convolutional bases for our models **VGG16**, **ResNet50V2**, **InceptionV3** pre-trained on ImageNet, for transfer learning from natural images to breast cancer images.

We customized each of these base ImageNet pre-trained models by removing the fully connected part and by adding ourselves a new custom set of densely connected layers. Specifically, for each model we explored three different fully connected structures:

[5] Where: TP = True positive; FP = False positive; TN = True negative; FN = False negative

Figure 20 0. Flatten layer + 1 Dense layer having 256 neurons

Figure 21 1. Flatten layer + 3 pairs of 1 Droput (0.5) + 1 Dense layer having 256 neurons



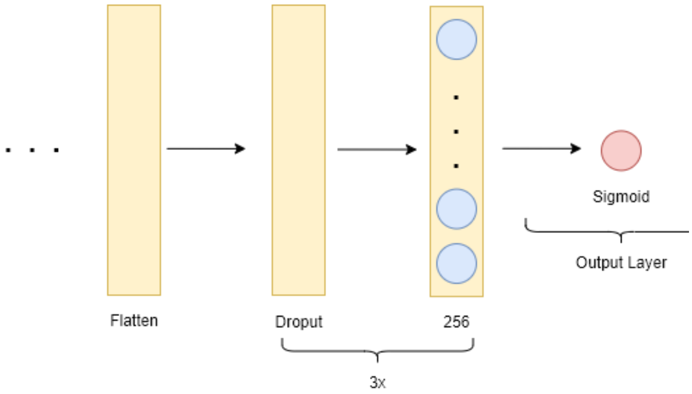
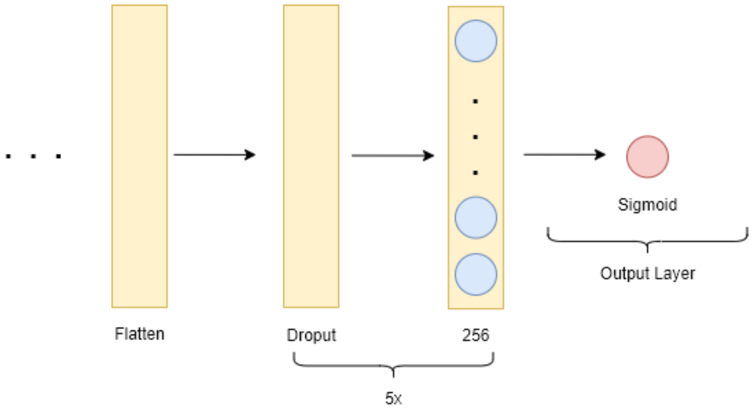


Figure 22. 2. Flatten layer + 5 pairs of 1 Dropout(0.5) + 1 Dense layer having 256 neurons



4.2.4.1. Which is the optimal number of layers to fine-tune?

Additionally, we adopted five different fine-tuning strategies: the idea is to try out different fractions of convolutional layers unfrozen to be fine-tuned so as to verify which is the best configuration for our specific problem and dataset. We investigated the optimal number of layers to fine-tune that would give us the best performance.

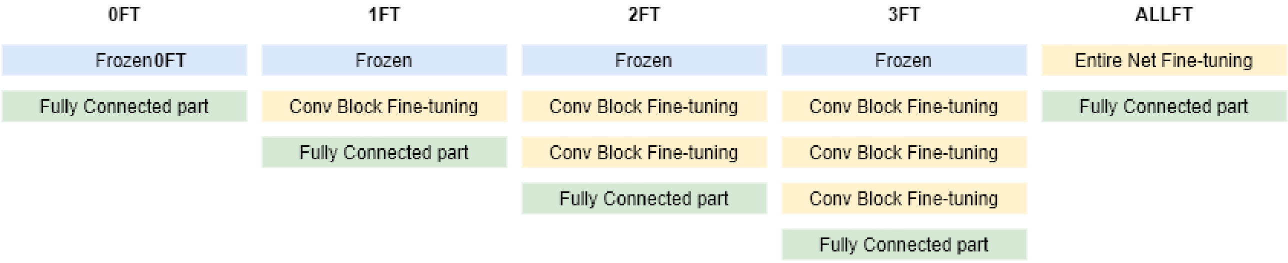
- 0 Fine-tuning: the convolutional base is entirely frozen, just train the custom fully connected part (Figure 20)
- 1 Fine-tuning: unfreeze only the last (top one) convolutional block and jointly fine-tune it with the custom fully connected part (Figure 21)
- 2 Fine-tuning: unfreeze the last two convolutional blocks and jointly fine-tune these with the custom fully connected part (Figure 22)
- 3 Fine-tuning: unfreeze the top three convolutional blocks and jointly fine-tune these with the custom fully connected part
- All Fine-tuning: weights are initialized as ImageNet trained, but train the entire network on our dataset

Note that since the three convolutional bases are very different, a convolutional block varies from one model to another resulting in a different number of keras layers to be unlocked.

4.2.4.2. Fine tuning procedure adopted

For each fine-tuning strategy adopted, we performed the following steps:

1. Add the custom network on top of an already-trained convolutional base network
2. Freeze the convolutional base
3. Train the fully connected added part
4. Unfreeze some convolutional blocks
5. Jointly train both the unfrozen conv blocks and the fully connected added part



0FT	1FT	2FT	3FT	ALLFT
Frozen0FT	Frozen	Frozen	Frozen	Entire Net Fine-tuning
Fully Connected part	Conv Block Fine-tuning	Conv Block Fine-tuning	Conv Block Fine-tuning	Fully Connected part
	Fully Connected part	Conv Block Fine-tuning	Conv Block Fine-tuning	
		Fully Connected part	Conv Block Fine-tuning	
			Fully Connected part	

Figure 23. Fine tuning strategy adopted from Scratch on Benign and Malignant CNN

4.2.4.3. Fine-tuning hyperparameters adopted

When training we fully connected model, we used **Adam** as optimizer with **1e-3 as learning rate**. On the other hand, when fine-tuning we chose a much **smaller learning rate**, 1e-5, so as to avoid the risk that the error signal propagating through the network during training would be too large thus destroying the representations previously learned by the layers being fine-tuned.

In both cases we used a batch of 20 samples and an early stopping procedure monitoring the validation loss with a patience set to 20.

4.2.5. Models Results

4.2.5.1. VGG16

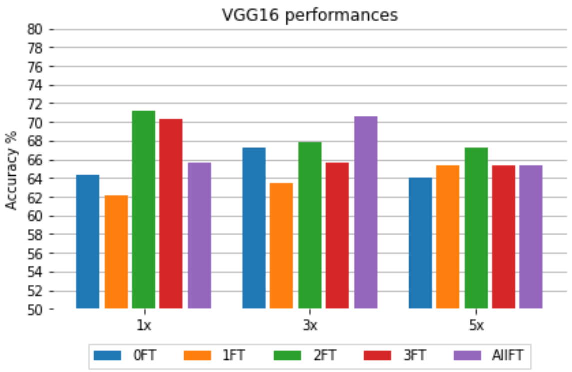


Figure 24. VGG16 performance Accuracy

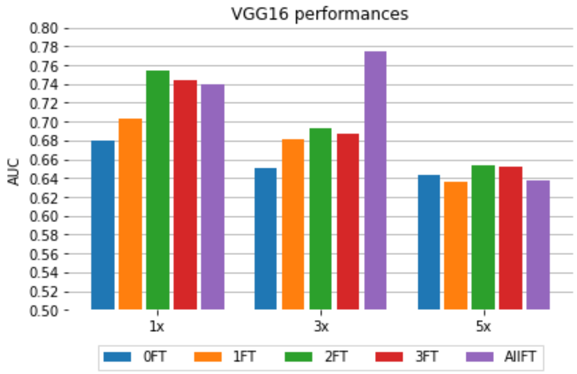


Figure 25 VGG16 performance AUC

Table 31. VGG16 TPR vs FPR

VGG16	1x		3x	
	TPR	FPR	TPR	FPR
0FT	55%	31%	33%	15%
1FT	58%	36%	51%	31%
2FT	64%	24%	46%	22%
3FT	60%	25%	62%	36%
ALLFT	67%	36%	68%	30%

Table 32. VGG16 Performance Accuracy and AUC

VGG16	1x		3x		5x	
	TEST ACC	AUC	TEST ACC	AUC	TEST ACC	AUC
0FT	64.38%	0.680	67.19%	0.651	64.06%	0.643
1FT	62.19%	0.703	63.44%	0.681	65.31%	0.636
2FT	71.25%	0.754	67.81%	0.693	67.19%	0.653
3FT	70.31%	0.744	65.62%	0.687	65.31%	0.652
ALLFT	65.62%	0.740	70.63%	0.775	65.31%	0.638

4.2.5.2. Resnet50v2

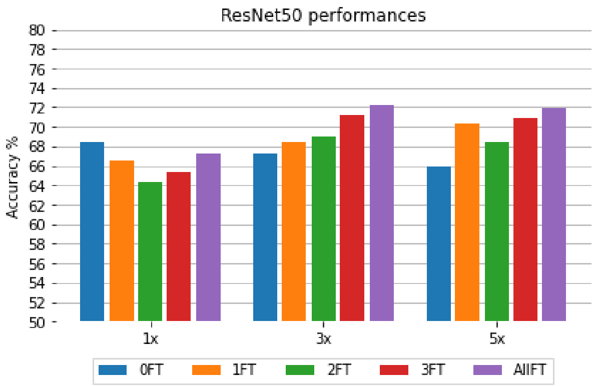


Figure 26 RestNet Version 2 performance Accuracy

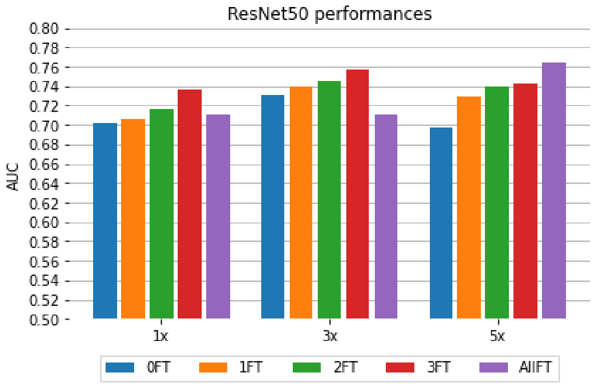


Figure 27 RestNet Version 2 performance AUC

Table 33. RestNet Version 2 TPR vs FPR

ResNet50V2	1x		3x		5x	
	TPR	FPR	TPR	FPR	TPR	FPR
0FT	49%	22%	54%	26%	46%	24%
1FT	62%	32%	52%	23%	37%	12%
2FT	61%	34%	65%	31%	68%	33%
3FT	63%	35%	68%	29%	63%	26%
ALLFT	54%	27%	71%	30%	57%	23%

Table 34. RestNet Version 2 Accuracy and AUC

ResNet50V2	1x		3x		5x	
	TEST ACC	AUC	TEST ACC	AUC	TEST ACC	AUC
0FT	68.44%	0.702	67.19%	0.731	65.94%	0.689
1FT	66.56%	0.706	68.44%	0.739	70.31%	0.729
2FT	64.38%	0.716	69.06%	0.745	68.44%	0.739
3FT	65.31%	0.736	71.25%	0.757	70.94%	0.742
ALLFT	67.19%	0.711	72.19%	0.771	71.88%	0.764

4.2.5.3. Inception v3

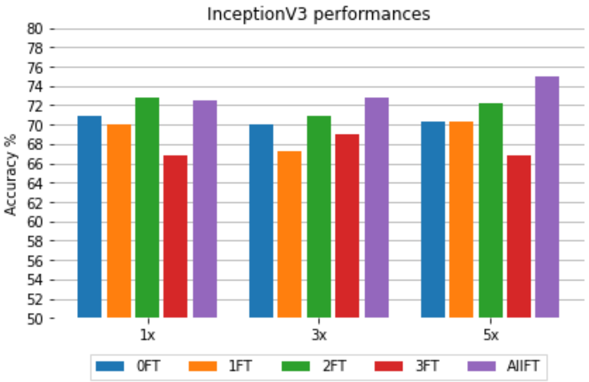


Figure 28. Inception Version 3 performance Accuracy

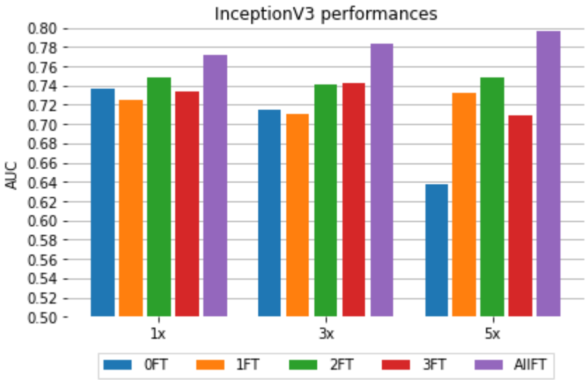


Figure 29. Inception Version 3 performance AUC

Table 35 Inception Version 3 TPR vs FPR

InceptionV3	1x		3x		5x	
	TPR	FPR	TPR	FPR	TPR	FPR
0FT	63%	26%	29%	9%	63%	26%
1FT	54%	22%	49%	24%	-	-
2FT	51%	16%	54%	21%	-	-
3FT	65%	33%	57%	26%	57%	29%
ALLFT			72%	29%	73%	25%

Table 36 Inception Version 3 Accuracy and AUC

InceptionV3	1x		3x		5x	
	TEST ACC	AUC	TEST ACC	AUC	TEST ACC	AUC
0FT	70.94%	0.737	70.00%	0.715	70.31%	0.737
1FT	70.00%	0.725	67.19%	0.711	70.31%	0.733
2FT	72.81%	0.749	70.94%	0.741	72.19%	0.748
3FT	66.87%	0.734	69.06%	0.742	66.87%	0.709
ALLFT	72.50%	0.771	72.81%	0.784	75.00%	0.797

4.2.5.4. Best model results

Convolutional Base	Inception V3 pretrained on ImageNet
Fully Connected Part	Flatten layer + 5 pairs of 1 Dropout(0.5) + 1 Dense layer having 256 neurons
Loss Function	Binary Crossentropy
Optimizer	Adam
Leaning Rate	[1e-3, 1e-5]
Batch Size	20
Epochs	200

TEST ACCURACY	75%
AUC	0.797
TPR	73%
FPR	25%
PRECISION	0.72
RECALL	0.74
F-SCORE	0.72

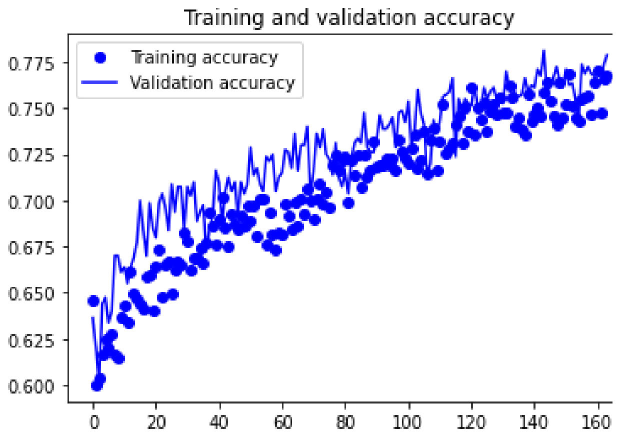


Figure 30. Training and validation accuracy Best Pre-training CNN

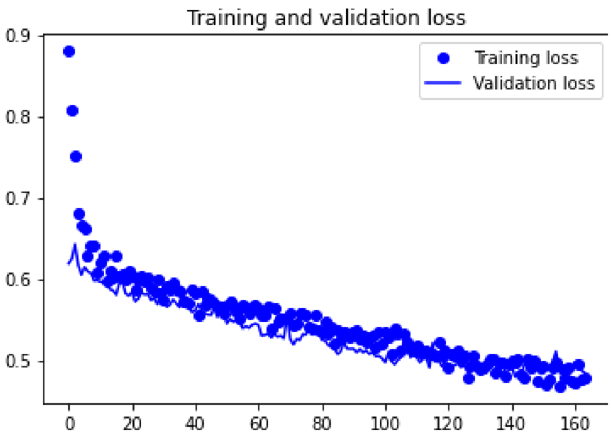


Figure 31. Training and validation loss Best Pre-training CNN

Confusion matrices comparison: best from scratch CNN VS best pre-trained and fine-tuned CNN:

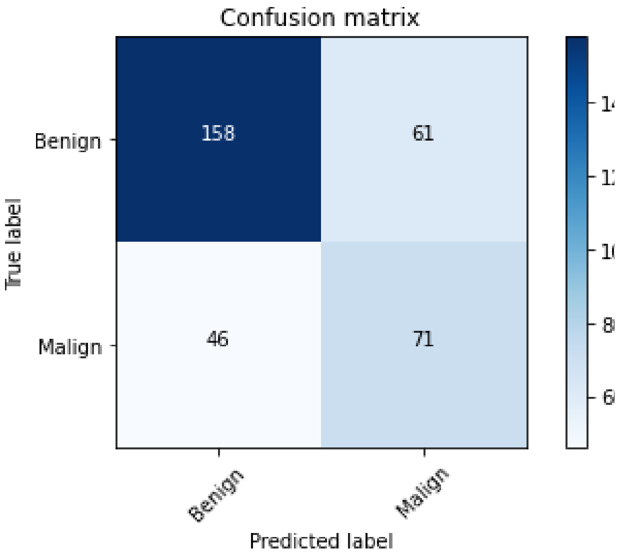


Figure 32. Best from Scratch CNN

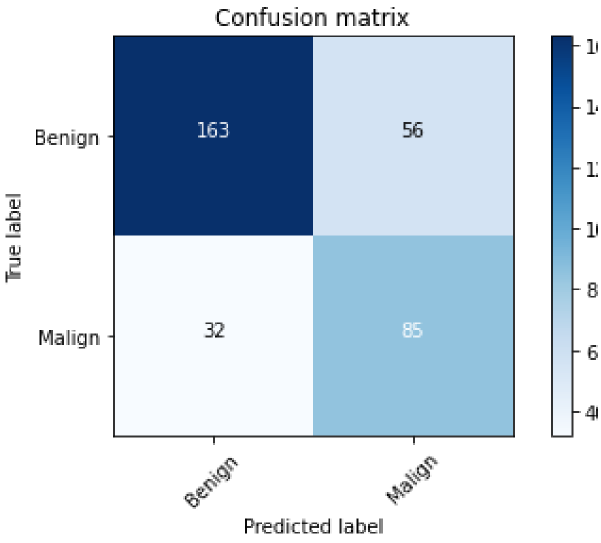


Figure 33. Best from Pre-Trained CNN

Chapter 5

5. Baseline Abnormality detection in mammography

5.1. Development of a classification model for discriminating the abnormality type or the diagnosis exploiting baseline patches

The main goal of this task it to try exploiting the additional pieces of information provided by the baseline parches so as to try to improve the performances of our models. Below we show some examples **of baseline vs abnormality samples**:

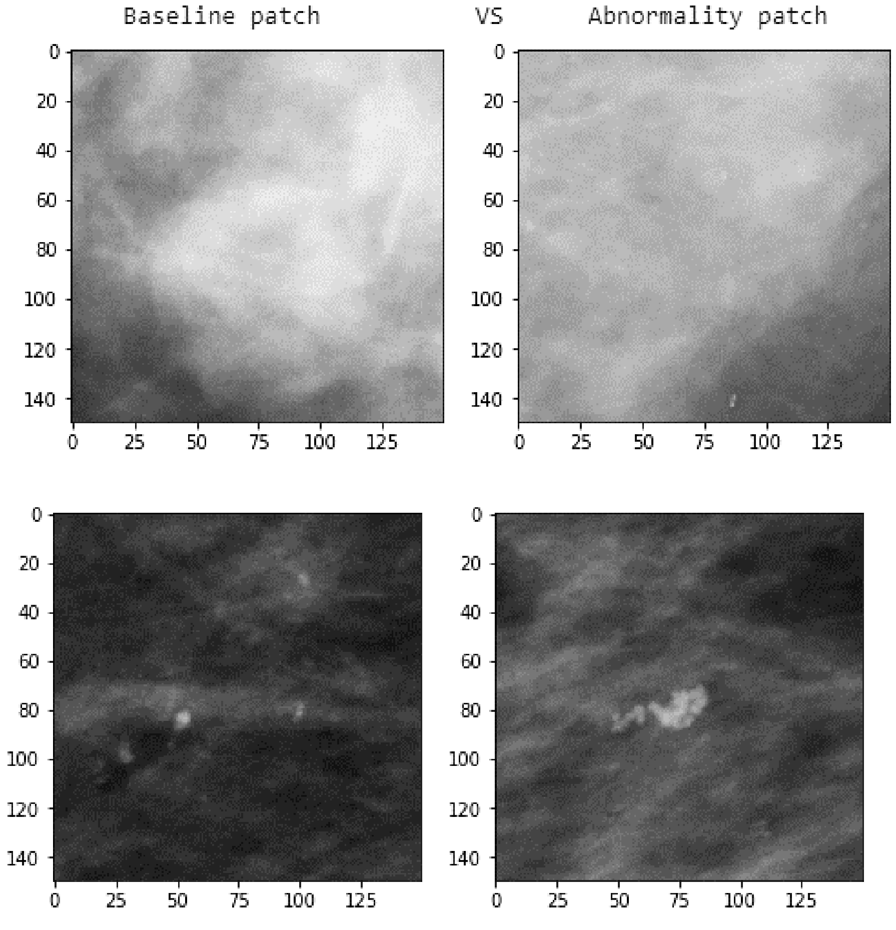


Figure 34. Baseline vs abnormality samples

We tried three different approaches for both systems (CADe) “Masses vs Calcification” and (CADx) “Benign vs Malign”:

1. Stacking Baseline as input layer below the relative Abnormality so as to obtain an input tensor shape like (#_of_samples, 150, 150, 3) as you can see below in (Figure 35)

2. Building rectangular images by concatenation of the Baseline and its relative Abnormality side by side so as to obtain input images shaped like (150, 300) as you can see in (Figure 36)

3. Exploitation of the Siamese Network using as CNN core our best performing models from the previous tasks (Figure 37)

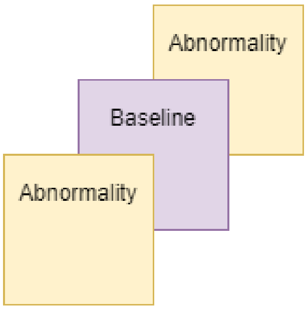


Figure 35. Stacking Baseline. [Abnormality, Baseline, Abnormality]

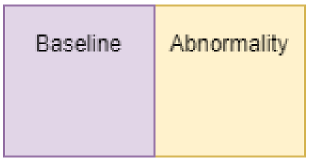


Figure 36. Concatenating Baseline. [Baseline, Abnormality]

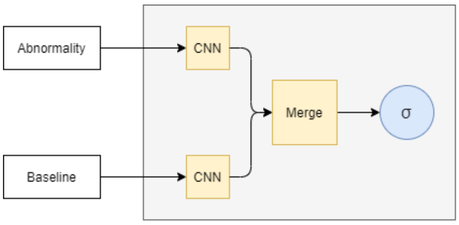


Figure 37. Siamese Network. [Abnormality, Baseline]

5.2. Stacking Baseline as layer below Abnormality

Given that our abnormality samples are greyscale images, and that we needed to repeat for all three layers of RGB the same value before passing the image to the network in the previous experiments, we tried to substitute one of the three layers with the baseline to see if the networks would be able to grasp some features useful to improve the classification results.

Unfortunately, **results worsened** with respect to results obtained without using baselines for both “**Masses vs Calcification**” and “**Benign vs Malign**” **by** an amount **of 5-10% test accuracy**. Results obtained are summarized in the following table:

Table 37. Summarized Results obtained. Stacking Baseline as Layer

MODEL NR	CONV BASE	FT STRATEGY	FC PART	TEST ACC
1	Inception V3	2FT	3x256	77.19%
2	Inception V3	ALLFT	5x256	84.38%

5.3. Concatenating Baseline and relative Abnormality horizontally

The second tentative was to concatenate the baseline alongside the abnormality, increasing the size of the image. In this case we added the baseline always on the same side because we don't have the exactly information where the baseline was extracted between [left, up, down, right], but not having reference information to inverse the process we opted for a fixed side, the left one. In this case the **performance is comparable to the one of the pretrained models without the baseline, with a small drop of around 2-6%.**

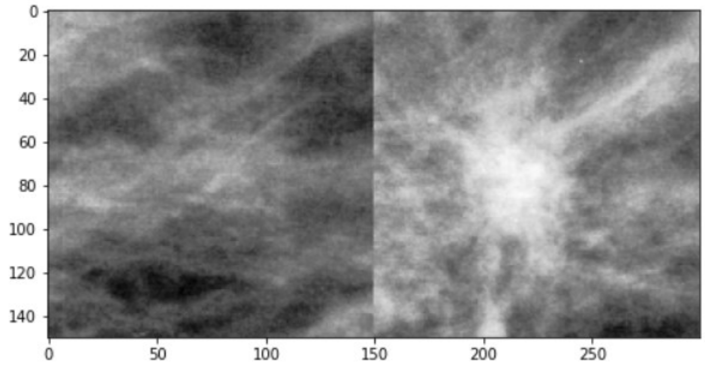
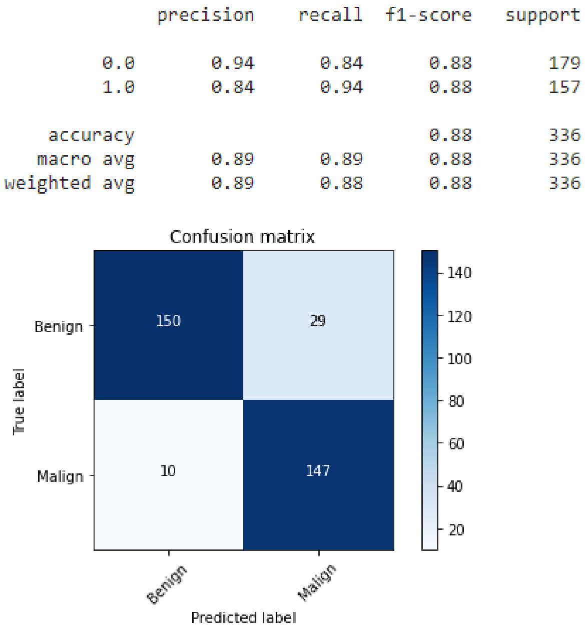


Figure 38. Concatenation Baseline with an abnormality layer example result. [Baseline, Abnormality]

Table 38. Masses VS Calcifications results:

MODEL NR	CONV BASE	FT STRATEGY	FC PART	TEST ACC
1	Inception V3	ALLFT	3x256	87.81%
2	Inception V3	ALLFT	5x256	87.72%

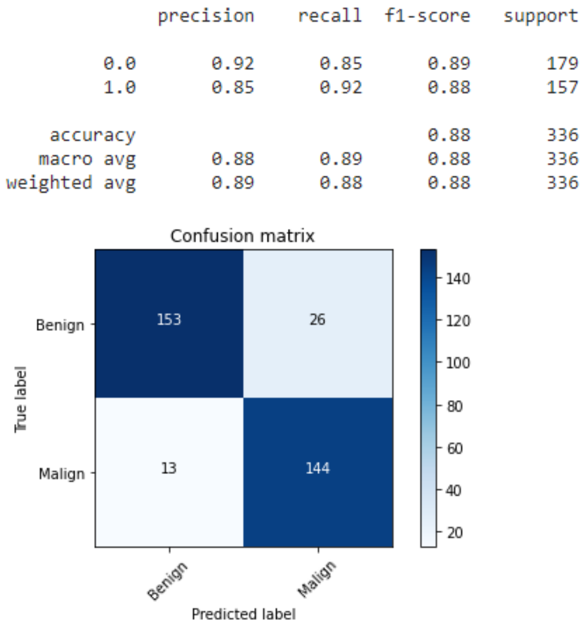


```
              precision    recall  f1-score   support

         0.0       0.94      0.84      0.88       179
         1.0       0.84      0.94      0.88       157

    accuracy                           0.88       336
   macro avg       0.89      0.89      0.88       336
weighted avg       0.89      0.88      0.88       336
```

Figure 39 - Model 1



```
              precision    recall  f1-score   support

         0.0       0.92      0.85      0.89       179
         1.0       0.85      0.92      0.88       157

    accuracy                           0.88       336
   macro avg       0.88      0.89      0.88       336
weighted avg       0.89      0.88      0.88       336
```

Figure 40 - Model 2

Table 39. Benign VS Malign results:

MODEL NR	CONV BASE	FT STRATEGY	FC PART	TEST ACC	TRP	FPR	AUC
1	Inception V3	ALLFT	3x256	64.06%	63%	36%	0.693
2	Inception V3	ALLFT	5x256	68.75%	-	-	0.733

5.4. Siamese Network

For the last experiment we tried to feed into a Siamese network both abnormality with the relative baseline as two input to the two networks used by the Siamese. Even for this case, there is a drop in the performances for both the classifications problem, in the order of a few percentile points in test accuracy. The usage of the baseline in these modes, within our experiments, did not bring improvements for our classification problem.

As last approach, we tried to exploit the Siamese Neural Network. We fed into the Siamese training pairs (X_i, Y_i) pairs where X_i = [baseline, abnormality], Y_i = abnormality label (0 if mass, 1 if abnormality). Unfortunately, neither in this case we obtained an improvement in performances, conversely, **there was a slight drop** (around 3-6%) with respect to the same task without using the baselines.

Table 40. Masses VS Calcifications results

MODEL NR	Core CNN	FT STRATEGY	FC PART	TEST ACC
1	ResNet50V2	ALLFT	3x256	83.98%
2	Inception V3	ALLFT	1x256	82.03%
3	Inception V3	ALLFT	1X256	84.77%

Table 41. Benign VS Malign results

MODEL NR	Core CNN	FT STRATEGY	FC PART	TEST ACC
1	Inception V3	ALLFT	3X256	66.41%

Chapter 6

6. Ensemble of Neural Networks

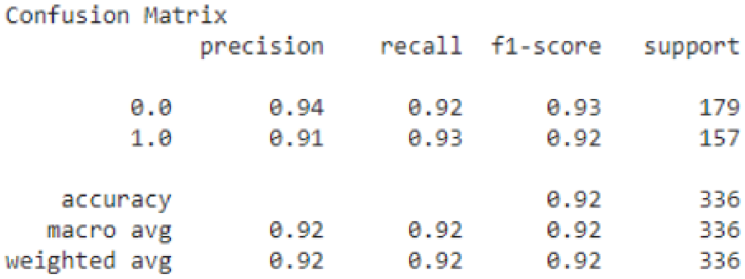
For the ensemble architecture we adopted a simple bagging approach where the output class predicted by the ensemble architecture is computed through a majority vote across different models. In particular we took the models that gave the best test accuracies and we used them all together.

6.1. Masses and Calcifications

We tried three different ensemble architecture, trying to take those architectures that scored the highest test accuracy:

Architecture	Score		
5 pretrained networks,	achieved	91.667%	test accuracy
11 pretrained networks,	achieved	92.26%	test accuracy
15 pretrained networks,	achieved	91.37%	test accuracy

All the architecture we tried obtained better result than the pretrained architectures alone. The best result achieved is using 11 pretrained networks, obtaining:



```
Confusion Matrix
              precision    recall  f1-score   support

         0.0       0.94      0.92      0.93       179
         1.0       0.91      0.93      0.92       157

    accuracy                           0.92       336
   macro avg       0.92      0.92      0.92       336
weighted avg       0.92      0.92      0.92       336
```

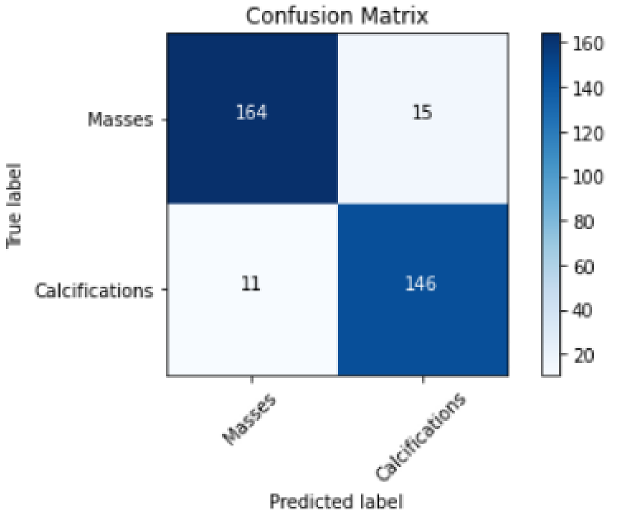
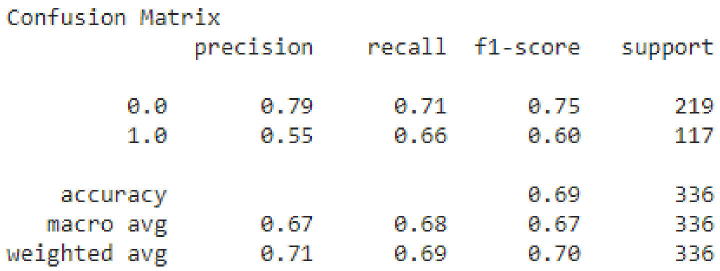


Figure 41. Ensemble Calcification and Masses Model

6.2. Benign and Malignant

In this case we assembled the ensemble using 5 pretrained model, obtaining results in line with the single pretrained, thus not gaining in terms of accuracy.



```
Confusion Matrix
              precision    recall  f1-score   support

         0.0       0.79      0.71      0.75       219
         1.0       0.55      0.66      0.60       117

    accuracy                           0.69       336
   macro avg       0.67      0.68      0.67       336
weighted avg       0.71      0.69      0.70       336
```

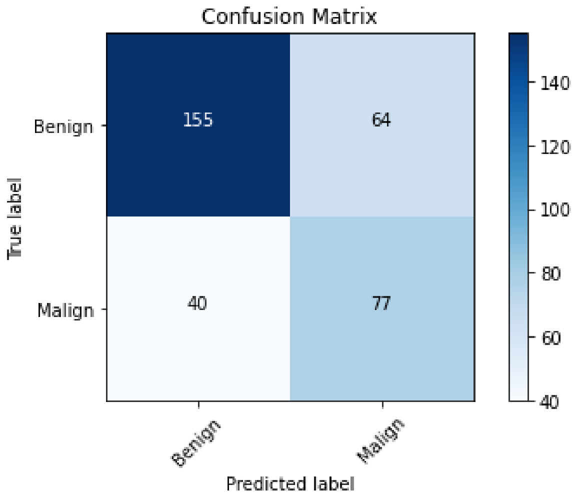


Figure 42. Ensemble Benign and Malignant Model