Department of Information Engineering - DII
MSc in Artificial Intelligence and Data Engineering
Academic Year 2020/2021
Computational Intelligence and Deep Learning

Convolutional Neural Network for Medical Imaging Analysis
Abnormality detection in mammography

Autors:
Davide Coccomini
Sina Gholami
Giuliano Zara

Contents

Introduction

The aim of the project is to train classifiers capable of distinguishing between masses and calcifications within mammograms and classifiers capable of distinguishing whether masses and calcifications, are benign or malignant. In order to fulfill this purpose, different techniques have been used exploiting also some approaches from the literature. The dataset used to train the deep neural networks is CBIS-DDSM (Curated Breast Imaging Subset of Digital Database for Screening Mammography): a digital breast imaging dataset performed during screening mammography. The original DDSM already contains a large amount of information for each of its 2,620 cases. However, some information is limited, specifically the ROI annotations, while other information is difficult to access. For that reason in the CBIS-DDSM dataset some improvements have been made and these issues have been solved by updating the ROI segmentations and by gathering and reformatting the metadata into a more accessible format.

Task 1: State of the art

In order to choose the best approach to address classification problems, we searched for computer-aided approaches capable of identifying masses and calcifications and discriminating between benign and malignant. Deep Learning has been widely used in the literature and we can often find uses of data augmentation, pre trained network (in particular VGG16, Inception, GoogLeNet and Resnet50) that have led in various combinations to the achievement of significant results in terms of accuracy of classification.

1 Abnormality Detection in Mammography using Deep Convolutional Neural Networks [3]

In order to avoid overfitting during training, in this study a data augmentation technique is applied on the training set with a random rotation between 0 and 360° and a reflection X and Y also random. The values are derived from several tests performed by the research team. In this study are exploited four of the best networks trained on ImageNet, a famous dataset containing over 15 million annotated images belonging to 22000 different categories, in an attempt to make these networks suitable for the analysis of mammograms identifying masses and calcifications. In particular, the networks considered are:

- AlexNet: The network was made up of 5 conv layers, max-pooling layers, dropout layers, and 3 fully connected layers. This work led to a series of deep CNN variants in the following years which consistently improved the state-of-the-art in the benchmark tasks.
- VGGNet: The network used very small 3x3 convolutional filters and showed significant improvement. This influential work indicated that CNNs need to have a deep network of layers in order for the hierarchical feature representations to work.
- GoogLeNet: Instead of sequentially stacking layers, this network was one of the first CNNs that used parallel structures in its architecture (9 Inception modules with over 100 layers in total)

- ResNet: 152-layer network architecture and set new records in ILSVRC. ResNet achieved 3.57% error rate in the classification task. The residual learning framework is 8 times deeper than VGGNet but still has lower complexity.

2 A novel deep learning based framework for the detection and classification of breast cancer using transfer learning [1]

The goal of this research is to identify malignant cells distinguishing them from benign ones within mammograms exploiting pretrained networks such as GoogLeNet, VGGNet and ResNet. Also in this study, data augmentation was performed by applying transformations and rotations to the images. The classification task results unaffected by rotation and whether the mass/calcification is benign or malignant should be understandable regardless of the rotation angle.

The results in this case are very promising both for the pretrained networks taken individually that reach an average accuracy of 90% and for the proposed method that reaches up to 97%. The result is clearly strongly influenced by the large amount of data available and some magnification techniques implemented on the initial dataset.

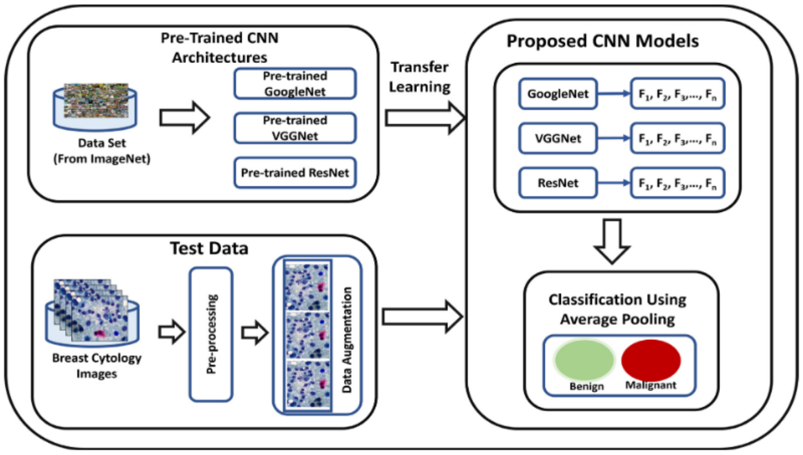


Figure 2.1: Proposed Deep Learning Framework

General Approach

1 Data Analysis

First of all, an analysis of the dataset was carried out in order to identify any features or manipulations needed in order to obtain better results in the various classification tasks. The training set containing examples of masses and calcifications is balanced as there are about 1300 masses and 1400 calcifications. The same is true for the dataset containing benign and malignant, even if with a higher level of difference in number between the two classes, with 1100 examples of the first class and about 1500 for the second. Therefore, no rebalancing technique was applicated. However, in the case of benign and malignant, given the complexity of the task and given the difference in instances in the two classes, tests were performed exploiting different class weights in order to improve classification. Unfortunately, in the majority of cases, the results did not improve in terms of overall accuracy but, if a different weight was given to the importance of false negatives rather than false positives, this technique could be applied giving better results. Indeed in some real cases may be more important to classify correctly the malignant cases also having a low accuracy because of wrong benign classification.

2 Data Augmentation

In general in all experiments data augmentation was considered necessary as the data is insufficient to obtain satisfactory accuracy results. This technique was tested with various settings inspired by the literature indicated in the state of the art. In fact, in all of the studies analyzed, data augmentation was found to be an excellent approach to achieving better performance. Random rotation between 0 and 360° was used as shown in [1] and horizontal and vertical flips were also applied. Several combinations were tried before arriving at this decision, for example we tried to use rotations of maximum 5° or to apply shifts on the images but all these settings led to worse accuracy.

3 Architectures and training

In each experiment we initially performed tests with simple structures and without applying further techniques in order to have a base accuracy value as a reference evaluation. Subsequently we apply further techniques to improve the performance of the classifiers such as fine tuning, more complex architectures etc. All the trainings were carried out using a 4-fold cross validation in order to make the accuracy results more reliable thus avoiding to obtain values dictated only by the random initialization of training. After the 4-fold cross validation, a model comes trained and used in order to carry out one evaluation on the supplied test set and the result is brought back in the following chapters. Many graphs have been realized during all the experiments in order to evaluate more accurately if we are overfitting, if it is necessary to apply regularization techniques etc.. All the confusion matrices considers the rows as the real classes and the columns as the predicted classes.

4 Memory Management

One of the principal problematic that we encountered was to manage in an intelligent way the use of the RAM that comes heavily exploited during the various phases of the training and for the memorization of the variables. Using a 4-fold validation, above all in the case of the pretrained in which more networks are trained in the same notebook, the RAM memory is saturated and the operation is not completed. Therefore, a multi-process approach was chosen in which each fold is performed on a separate process whose result is returned in the main process for further processing and the process is destroyed at the end of the single fold. With the destruction of the process also all the allocated memory is freed and returns to be available for the successive training phases.

Task 2: Scratch CNN

To start implementing our model we searched for different architectures form various articles[1-3] which used the same data set. We decided to test their used architecture model to evaluate the early results. In the first step to develop our model, we copied the 1.0.0.35 model in [4] which is also exploit VGG philosophy. They used 40 epochs for training the model. However, our preliminary results showed us 30 epochs is enough, so the network accuracy and loss would not change after 30 epochs. Note that the image size in the paper is 299x299 but our data image size is 150x150. Thus, we must remove the last two layers (3x3 Conv - 512 and 2x2 Max Pool) otherwise it is not possible to run the model. Our saved data are in numpy ndarray 16-bit format. Unfortunately, they did not mention anything about their augmentation parameters. So we tried our own augmentation approach which we explained before. In addition to create more robust model we consider the following parameters for the augmentation in scratch stage:

```
height_shift_range = 0.1,
width_shift_range = 0.1,
shear_range = 0.01,
zoom_range=0.1.
```

Code 4.1: Data augmentation parameters

All the values are very small, since we do not want shift or shear the image too much to lose important data in middle of image. Since the number of data is not large, we consider small batch size 16. We also developed another model which is very similar to 1.0.0.35 model. We called it regularized model since it uses regularization L1 and L2. The value of L1 and L2 are $1e^5$ and $1e^4$ since we faced bad accuracy results when they are close to 1. So we chose not very big nor very small values. In this part, we consider 70 percent of data for training and the rest for validation. We consider ”accuracy” as the metric of our training.

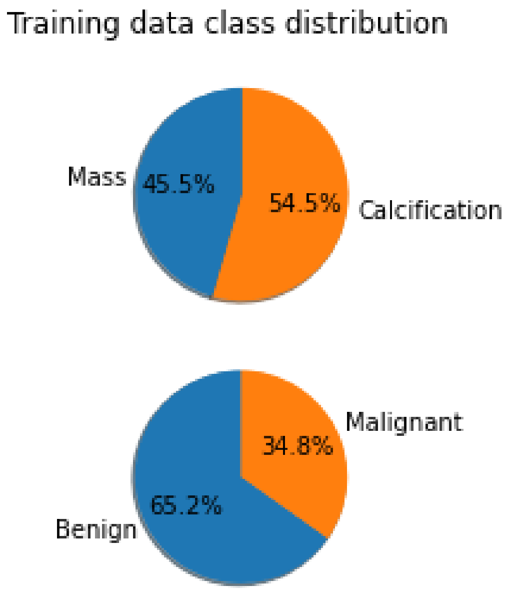


Figure 4.1: Classes distribution

According to the pie charts above, our data is not balanced, so we consider class weight for class X as follow:

$$weight(x) = \frac{frequency(Y)}{frequency(X) + frequency(Y)}$$

Since the the accuracy fluctuating as shown in the figures below so we introduced an early stop condition with 4 number of patience and increased the number of epochs to 80.

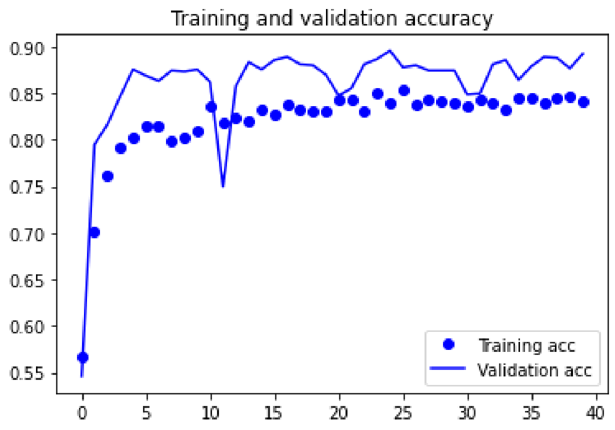


Figure 4.2: Training and validation accuracy (Mass/Calcification)

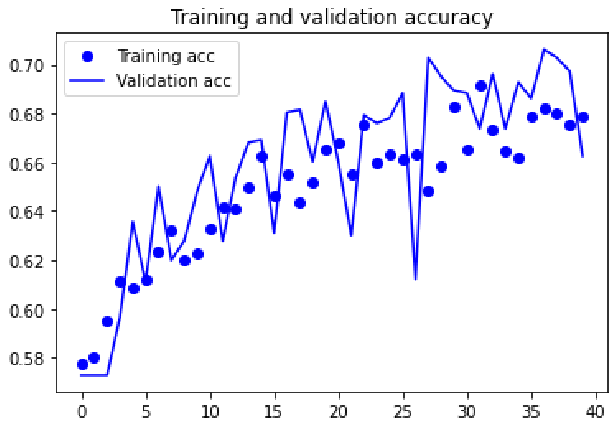


Figure 4.3: Training and validation accuracy (Benign/Malignant)

As a last step in order to validate the final model obtained we performed the 10-fold cross validation. The following tables show a summary of all the results obtained in the different steps. As we can see, after using class balancing the accuracy of test is more close to the accuracy of the validation. In the table we also mentioned the number of epochs, but since we employed early stopping the network may stop before reaching the specific number of epochs. M1 is referring to 1.0.0.35 model and M2 is our developed one. Consider that we put the Training and Validation accuracy of last epoch.

	Accuracy		
Model	Training	Validation	Test
Scratch (M1, 30 epochs)	0.7588	0.7397	0.6726
Scratch (M1, 30 epochs, class balancing)	0.7899	0.7460	0.7202
Scratch (M1, 80 epochs, class balancing, early stop)	0.7453	0.8020	0.7827
Scratch (M2, 30 epochs, class balancing, early stop)	0.7758	0.7584	0.7530
Scratch (M1, 25 epochs, 10-fold)		0.7543	0.8035

Table 4.1: Classification results with the scratch model (Mass/Calcification)

	Accuracy		
Model	Training	Validation	Test
Scratch (M1, 30 epochs)	0.5635	0.4969	0.4345
Scratch (M1, 30 epochs, class balancing)	0.5801	0.6189	0.6429
Scratch (M1, 80 epochs, class balancing, early stop)	0.5111	0.5903	0.6518
Scratch (M2, 30 epochs, class balancing, early stop)	0.5277	0.5890	0.6518
Scratch (M1, 30 epochs, 10-fold)		0.6180	0.6517

Table 4.2: Classification results with the scratch model (Benign/Malignant)

We save the last four models of each classification: models which used class balancing plus cross validated model.

Task 3: Use of Pretrained CNN

In order to obtain a more performing model we have identified some papers in which pretrained networks were used, identifying the most suitable for the purpose. Looking in particular at for example the research [3] were used pretrained networks such as VGG, Resnet and Inception, the latter particularly deep. We performed tests with both pretrained networks followed by two dense layers. All the pretrained networks considered were trained using ImageNet which contains RGB images as opposed to the images of the dataset considered which are instead in black and white. Therefore, before being able to proceed with the training it was necessary to transform the images into RGB by replicating the only layer present in the original images on three levels.

1 Classifiers

1.1 VGG16

VGG16 is a convolutional neural network model which achieves 92.7% top-5 test accuracy in ImageNet, which is a dataset of over 14 million images belonging to 1000 classes. It makes the improvement over AlexNet by replacing large kernel-sized filters (11 and 5 in the first and second convolutional layer, respectively) with multiple 3×3 kernel-sized filters one after another.

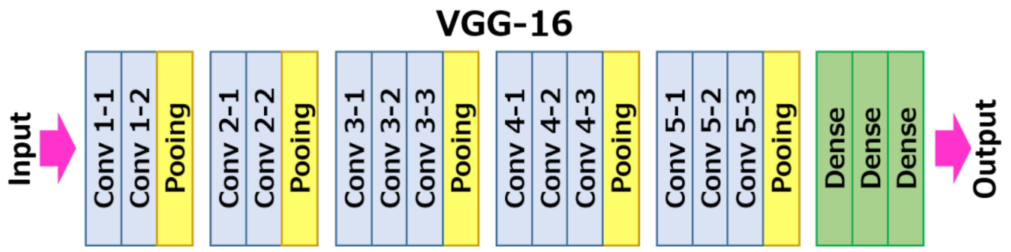


Figure 5.1: VGG16 Architecture

To obtain enough data to perform more effective training, data augmentation was applied with a value of 360 degrees of rotation as suggested in analyzed papers. Evaluations were done in terms of accuracy using a crossfold validation with k=4, 40 epochs and a batch size of 16. The resulting average accuracy on the validation set is 0.77.

In order to further improve accuracy, fine tuning was performed by unfreezing the last layers of the network.

1.2 Resnet50

ResNet50 is a variant of ResNet model which has 48 Convolution layers along with 1 MaxPool and 1 Average Pool layer. It has 3.8 x $10\hat{9}$ Floating points operations. It is a widely used ResNet model and we have explored ResNet50 architecture in depth.

1.2.1 Batch Normalization

As suggested in the literature, in order to obtain better results with Resnet50 it is good to use batch normalization. Batch normalization is a technique for training very deep neural networks that standardizes the inputs to a layer for each mini-batch. This has the effect of stabilizing the learning process and dramatically reducing the number of training epochs required to train deep networks.

1.3 InceptionV3

Inception-v3 is a convolutional neural network architecture from the Inception family that makes several improvements including using Label Smoothing, Factorized 7 x 7 convolutions, and the use of an auxiliary classifer to propagate label information lower down the network (along with the use of batch normalization for layers in the sidehead).

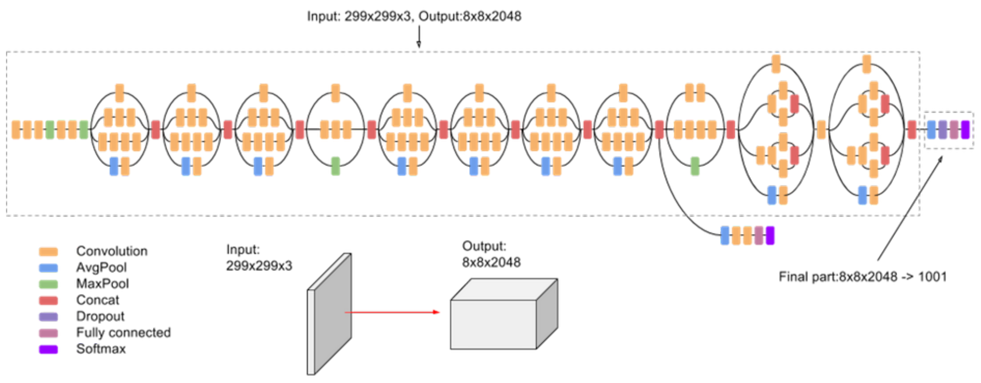


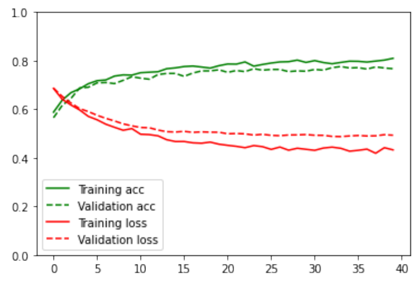
Figure 5.2: InceptionV3 Architecture

2 Mass and calcifications

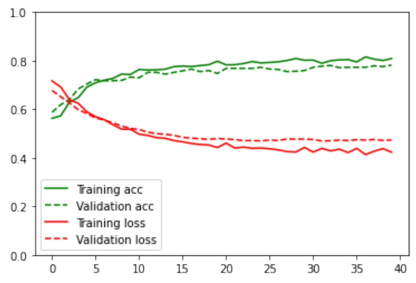
The first classification task concerns the problem of classifying masses and calcifications. This task in the various experiments proved much easier to accomplish probably due to the greater and more incisive differences between these two classes.

2.1 VGG16

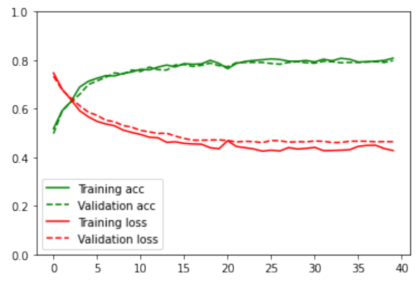
The model was trained by adding to the pretrained network a fully connected one with an internal layer of 256 and an output of 1 with a sigmoid activation function. The chosen optimizer, after multiple trials, is RMS with a learning rate of 2^{-6} and the loss function used is binary crossentropy. The network was validated using a 4-fold cross validation that led to an average accuracy on the validation set of 0.78 with a standard deviation of 0.01. The following graphs show the trend of the accuracy and the loss in the 4 folds over the 40 epochs considered for each of them.



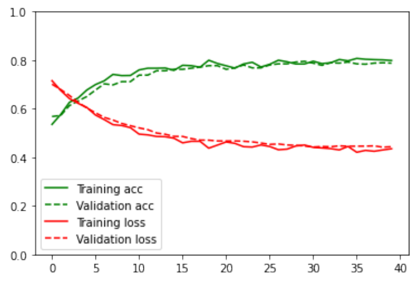
(a) Fold 1



(b) Fold 2



(c) Fold 3



(d) Fold 4

Figure 5.3: VGG16 accuracy and loss on k-fold

Looking at the graphs we can see that the accuracy on the validation set and on the training set go hand in hand and the same is true for the loss even if with small differences between a fold and the other linked to the random initialization of the weights. From this we can deduce the absence of overfitting and underfitting. The network was then retrained with 50 epochs to obtain a usable network on the test set and the resulting plot is as follows:

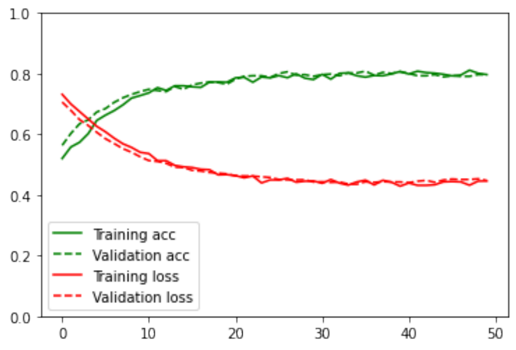


Figure 5.4: VGG16 accuracy and loss on test

As in the previous tests, there is neither overfitting nor underfitting and testing the network, stopping at epoch 50, on the test set achieves an accuracy of 0.79. The resulting confusion matrix is as follows from which an equally distributed error rate between masses and calcifications can be seen.

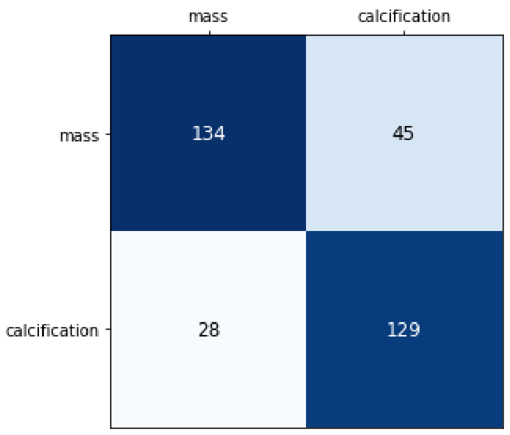


Figure 5.5: VGG16 test confusion matrix

2.1.1 Fine Tuning

In order to improve the performance we performed fine tuning on the last convolutional layer of the network achieving the following results:

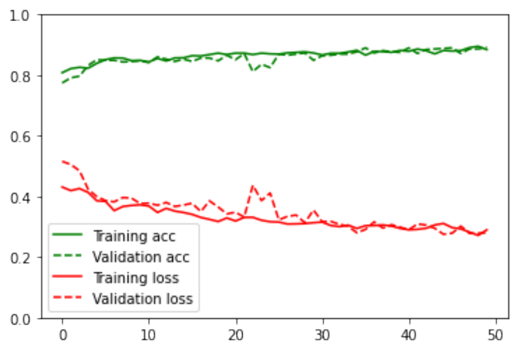
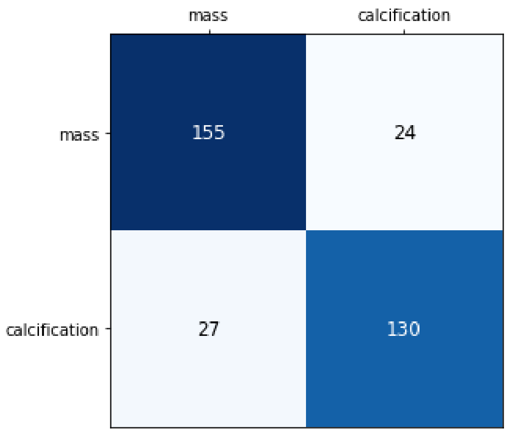


Figure 5.6: VGG16 fine tuning accuracy

As we can see from the plot the curve is very stable both on validation and training set and trying to classify on test set achieved an accuracy of 0.85, improving the accuracy of 0.06, with the following confusion matrix:

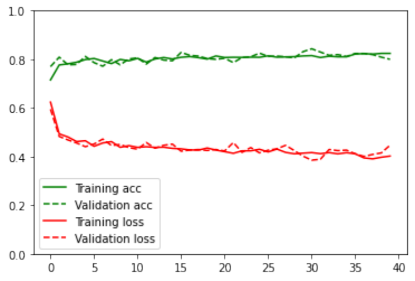


	mass	calcification
mass	155	24
calcification	27	130

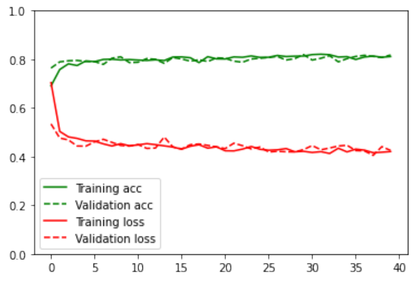
Figure 5.7: Fine tuned VGG16 confusion matrix on test

2.2 Resnet50

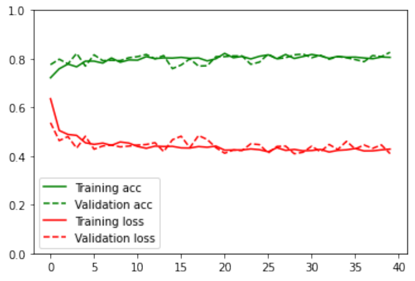
The model was trained by adding to the pretrained network a fully connected one with an internal layer of 256 and an output of 1 with a sigmoidal activation function. The optimizer selected, after multiple trials, is Adam with a learning rate of 0.001 and loss function binary crossentropy. The network was trained using a 4-fold cross validation that led to an average accuracy on the validation set of 0.81 with a standard deviation of 0.01. The following graphs show the trend of the accuracy and the loss in the 4 folds over the 50 epochs considered for each of them.



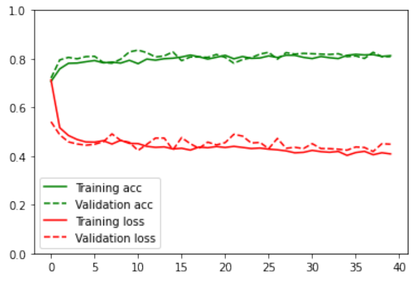
(a) Fold 1



(b) Fold 2



(c) Fold 3



(d) Fold 4

Figure 5.8: ResNet accuracy and loss on k-fold

Looking at the graphs we can see that the accuracy on the validation set and on the training set go hand in hand and the same is true for the loss even if with small differences between a fold and the other linked to the random initialization of the weights. From this we can deduce the absence of overfitting and underfitting.

The network was then retrained with 200 epochs to obtain a usable network on the test set and the resulting plot is as follows:

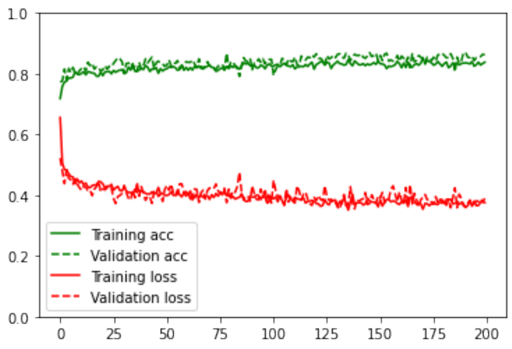


Figure 5.9: ResNet accuracy and loss final train

As in the previous tests, there is neither overfitting nor underfitting and testing the network on the test set achieves an accuracy of 81%. The resulting confusion matrix is as follows from which an equally distributed error rate between masses and calcifications can be seen.

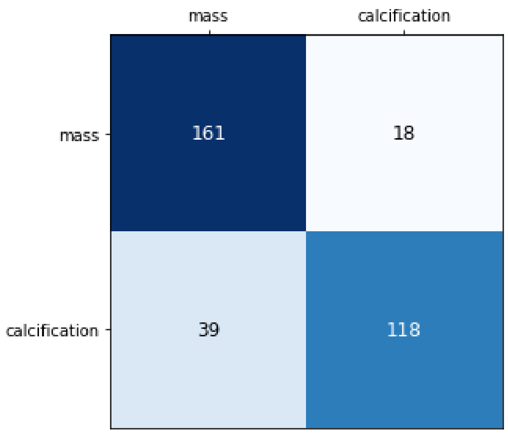


Figure 5.10: ResNet test confusion matrix

2.2.1 Fine Tuning

In order to improve the performance we performed fine tuning on the last convolutional layer of the network achieving the following results:

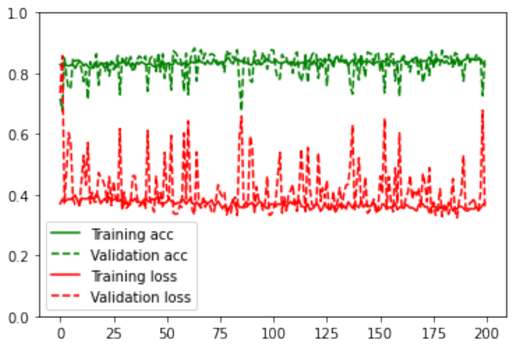


Figure 5.11: ResNet fine tuning accuracy

The curve is not very stable in this case but applying regularization techniques did not carried out to improvements in accuracy. Trying to classify on test set achieved an accuracy of 0.83, higher than the normal model, with the following confusion matrix:

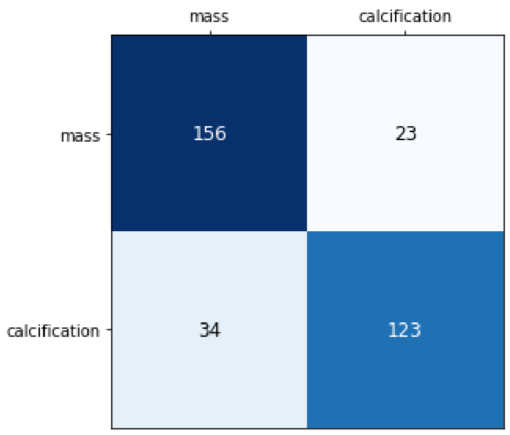
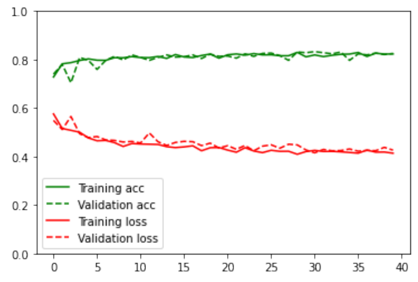


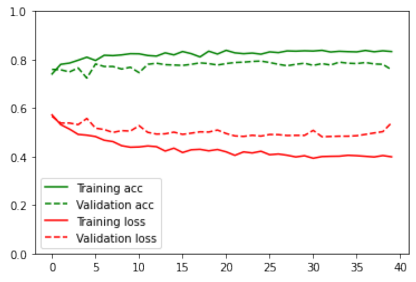
Figure 5.12: Fine tuned ResNet test confusion matrix

2.3 InceptionV3

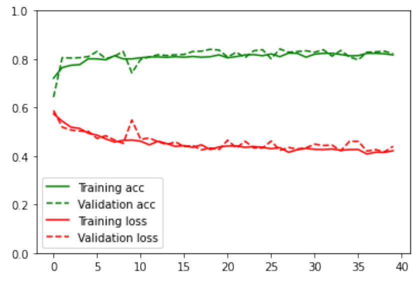
The model was trained by adding to the pretrained network a fully connected one with an internal layer of 256 and an output of 1 with a sigmoidal activation function. The optimizer selected, after multiple trials, is RMS with a learning rate of 2^{-5} and loss function binary crossentropy. The network was trained using a 4-fold cross validation that led to an average accuracy on the validation set of 0.81 with a standard deviation of 0.031. The following graphs show the trend of the accuracy and the loss in the 4 folds over the 40 epochs considered for each of them.



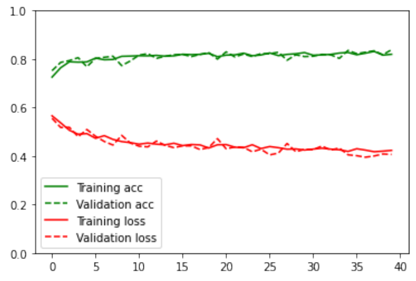
(a) Fold 1



(b) Fold 2



(c) Fold 3



(d) Fold 4

Figure 5.13: Inception accuracy and loss on k-fold

Looking at the graphs we can see that the accuracy on the validation set and on the training set go hand in hand and the same is true for the loss even if with small differences between a fold and the other linked to the random initialization of the weights. Only in fold 2 we can see a difference between training and validation accuracy and loss. From this we can deduce the absence of overfitting and underfitting. The network was then retrained with 50 epochs to obtain a usable network on the test set and the resulting plot is as follows:

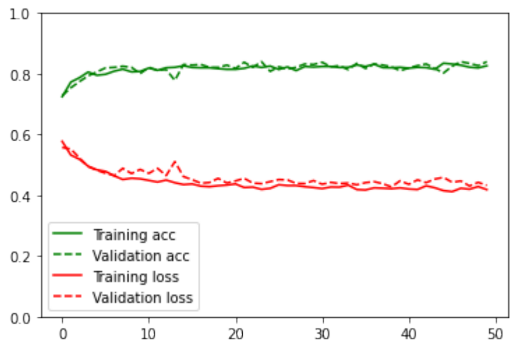


Figure 5.14: Inception test accuracy and loss

As in the previous tests, there is neither overfitting nor underfitting and testing the network on the test set achieves an accuracy of 0.79. The resulting confusion matrix is as follows from which an equally distributed error rate between masses and calcifications can be seen.

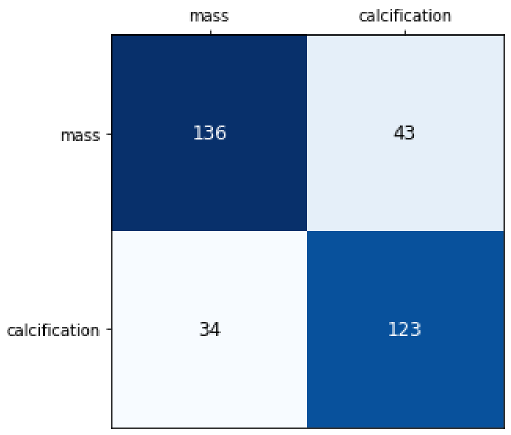


Figure 5.15: Inception test confusion matrix

2.3.1 Fine Tuning

In order to improve the performance we performed fine tuning on the last convolutional layer of the network achieving the following results:

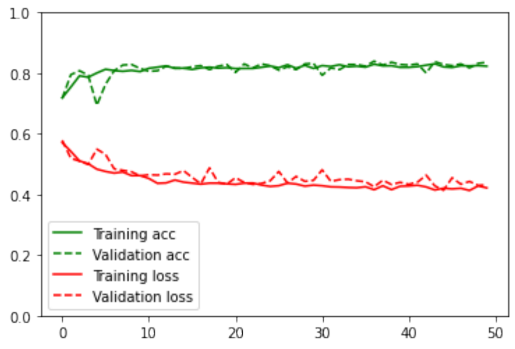


Figure 5.16: Inception fine tuning accuracy

As we can see from the plot the curve is stable both on validation and training set and trying to classify on test set achieved an accuracy of 0.75, becoming worse than the original one, probably because of the selected unfreezed layers, with the following confusion matrix:

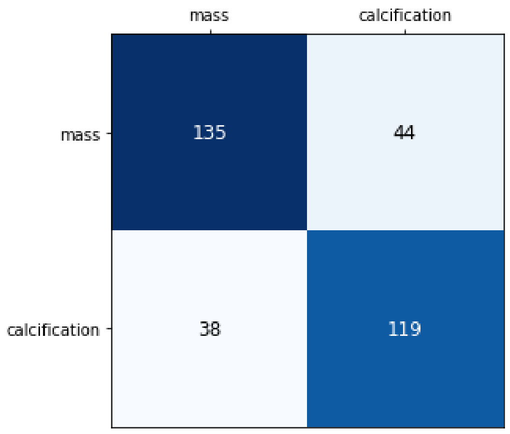


Figure 5.17: Fine tuned Inception test confusion matrix

2.4 Comparisons

For the purpose of identifying the best model, ROC curves were used. The ROC curves for the non-fine tuned models are as follows:

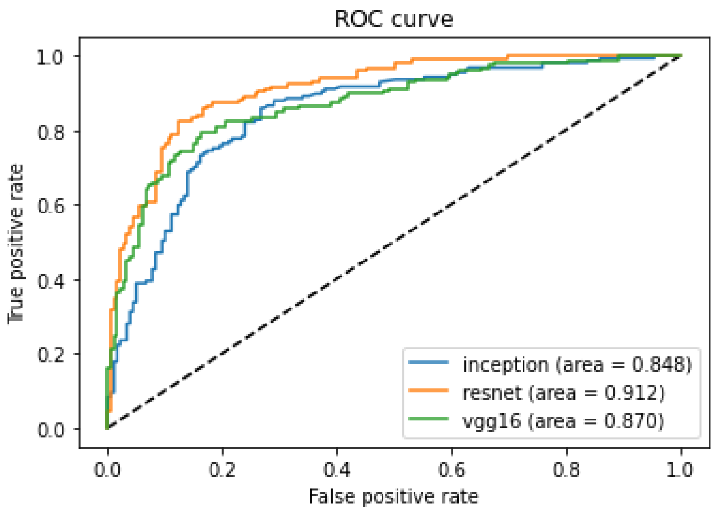


Figure 5.18: ROC Curves

As shown in the plot, the AUC of ResNet (0.912) is higher than the one of VGG16 (0.870) and Inception (0.848) and it is therefore preferable to select this model for classification.

Applying same method on the fine tuned networks we obtain the following plot:

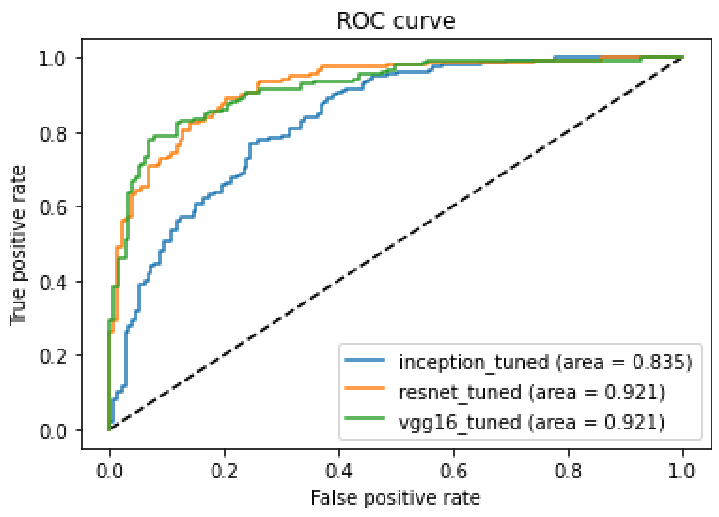


Figure 5.19: ROC Curves for Fine Tuned models

In this case there is a slight improvement in ResNet but a significant improvement in VGG16, leading to similar AUC. Looking at the graph we can see slight differences in the area delimited by the two classifiers and therefore it would be the case to select one rather than another based on the specific needs of the individual classification task.
A final comparison can be done on the obtained accuracies of all the models during the testing phase:

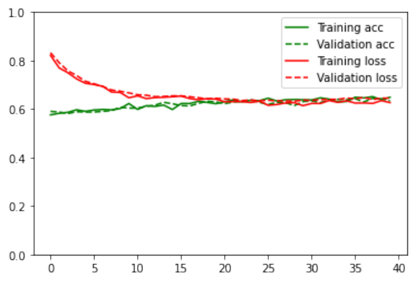
Model	Test Accuracy
VGG16	0.78
VGG16 Fine Tuned	**0.85**
ResNet	0.83
ResNet Fine Tuned	0.83
Inception	0.77
Inception Fine Tuned	0.75

Table 5.1: Classification results with the pretrained models (Mass/Calcification)

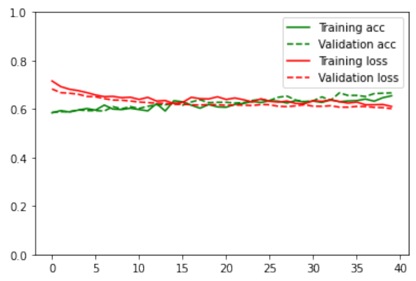
3 Benign and Malignant

3.1 VGG16

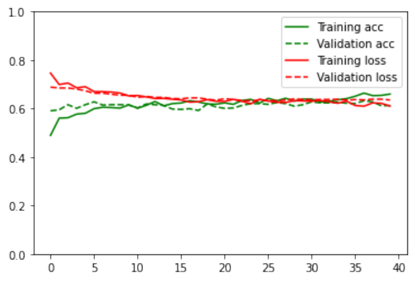
The model was trained by adding to the pretrained network a fully connected one with an internal layer of 256 and an output of 1 with a sigmoidal activation function. The optimizer selected, after multiple trials, is RMS with a learning rate of 2^{-6} and loss function binary crossentropy. The network was trained using a 4-fold cross validation that led to an average accuracy on the validation set of 0.64 with a standard deviation of 0.019. The following graphs show the trend of the accuracy and the loss in the 4 folds over the 40 epochs considered for each of them.



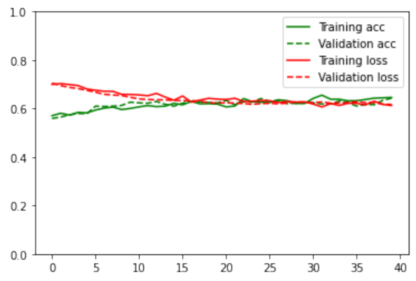
(a) Fold 1



(b) Fold 2



(c) Fold 3



(d) Fold 4

Figure 5.20: VGG16 accuracy and loss on k-fold

Looking at the graphs we can see that the accuracy on the validation set and on the training set go hand in hand and the same is true for the loss even if with small differences between a fold and the other linked to the random initialization of the weights. From this we can deduce the absence of overfitting and underfitting. The network was then retrained with 50 epochs to obtain a usable network on the test set and the resulting plot is as follows:

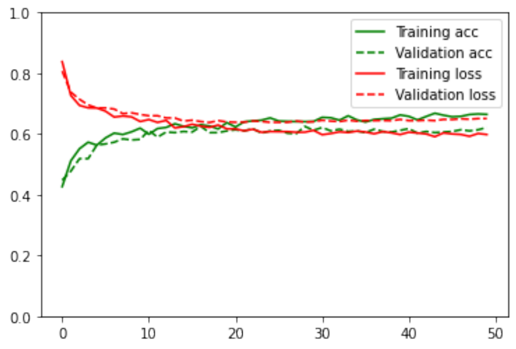


Figure 5.21: VGG16 accuracy and loss test

As in the previous tests, there is neither overfitting nor underfitting and testing the network on the test set achieves an accuracy of 64%. The resulting confusion matrix is as follows from which an equally distributed error rate between benign and malignant can be seen.

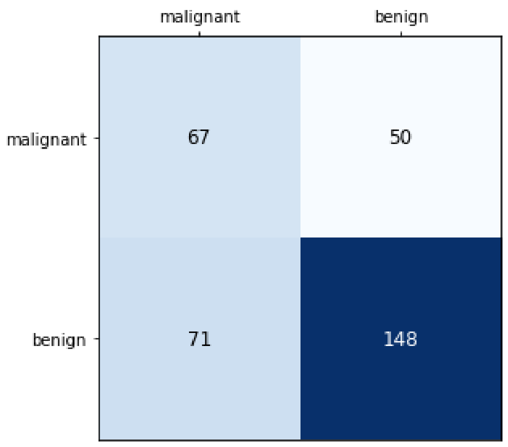


Figure 5.22: VGG16 test confusion matrix

3.1.1 Fine Tuning

In order to improve the performance we performed fine tuning on the last convolutional layer of the network achieving the following results:

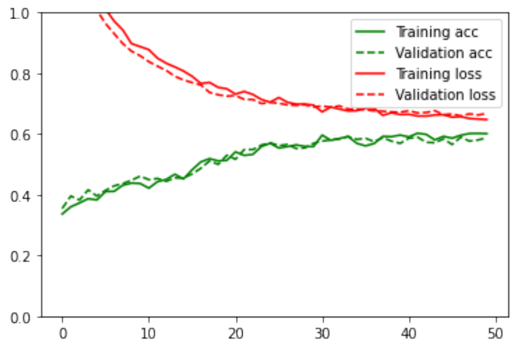
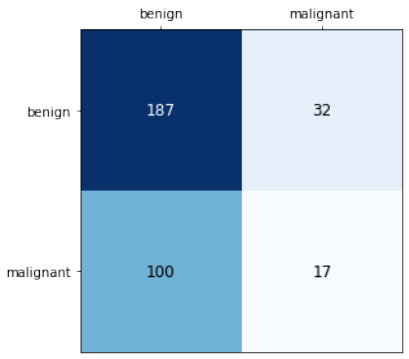


Figure 5.23: Fine Tuned VGG16 accuracy

As we can see from the plot the curve is very stable both on validation and training set with a quite high value of loss. Trying to classify on test set the network achieved an accuracy of 61% with the following confusion matrix:



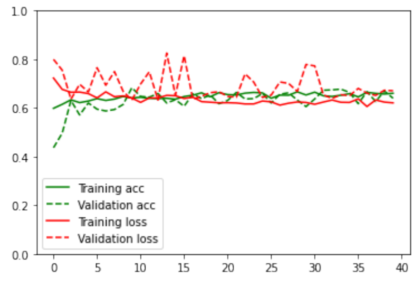
	benign	malignant
benign	187	32
malignant	100	17

Figure 5.24: Fine tuned VGG16 test confusion matrix

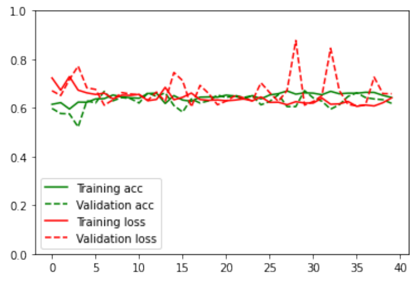
The network tends to classify more one class than another, this is because of the unbalancing of the dataset as well as the complexity of the classification task. In order to obtain better results have been made several tests using various values of class weights but have not led to better results.

3.2 Resnet50

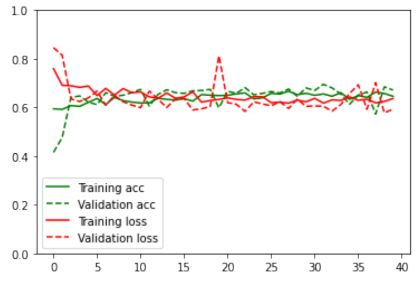
The model was trained by adding to the pretrained network a fully connected one with an internal layer of 256 and an output of 1 with a sigmoidal activation function. The optimizer selected, after multiple trials, is Adam with a learning rate of 0.001 and loss function binary crossentropy. The network was trained using a 4-fold cross validation that led to an average accuracy on the validation set of 0.65 with a standard deviation of 0.020. The following graphs show the trend of the accuracy and the loss in the 4 folds over the 50 epochs considered for each of them.



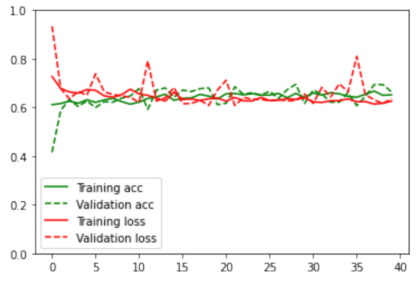
(a) Fold 1



(b) Fold 2



(c) Fold 3



(d) Fold 4

Figure 5.25: ResNet accuracy and loss on k-fold

Looking at the plots does not seems to be relevant overfitting or underfitting.

The network was then retrained with 200 epochs to obtain a usable network on the test set and the resulting plot is as follows:

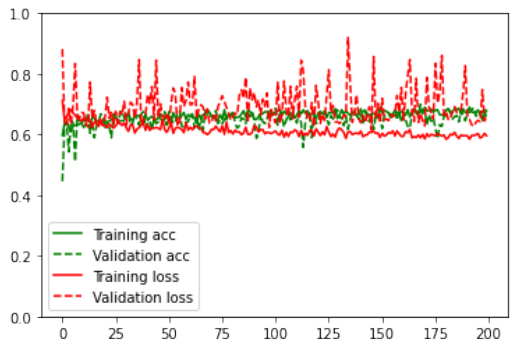


Figure 5.26: ResNet accuracy and loss test

As in the previous tests, there is neither overfitting nor underfitting and testing the network on the test set achieves an accuracy of 64%. The resulting confusion matrix is as follows from which an equally distributed error rate between benign and malignant can be seen.

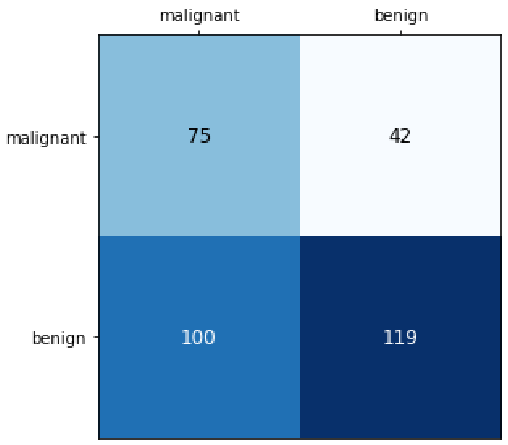


Figure 5.27: ResNet test confusion matrix

3.2.1 Fine Tuning

In order to improve the performance we performed fine tuning on the last convolutational layer of the network achieving the following results:

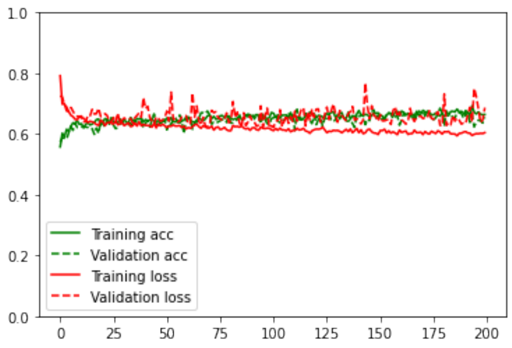


Figure 5.28: Fine Tuned ResNet accuracy

As we can see from the plot the curve is more stable than before both on validation and training set and trying to classify on test set achieved an accuracy of 65% with the following confusion matrix:

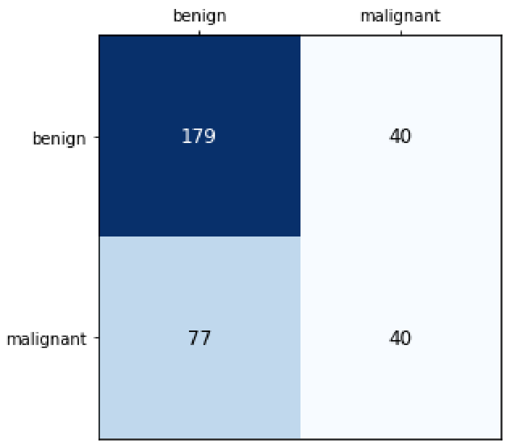
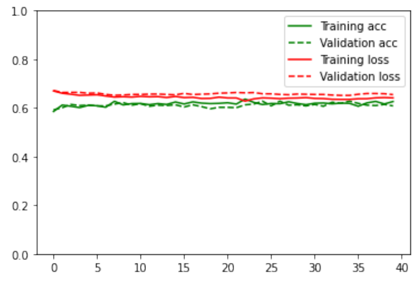


Figure 5.29: Fine tuned ResNet test confusion matrix

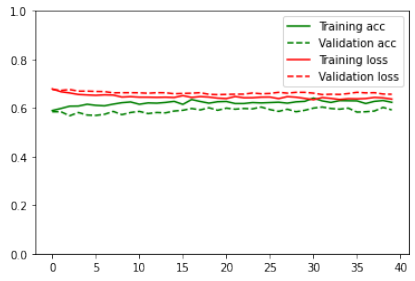
As in VGG16, because of classes balancing the classification error is not equally distributed between classes.

3.3 InceptionV3

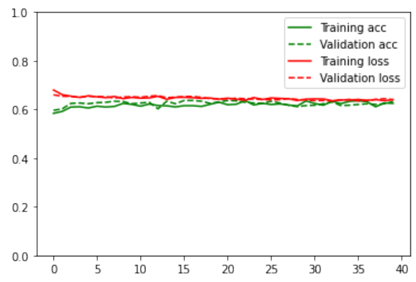
The model was trained by adding to the pretrained network a fully connected one with an internal layer of 256 and an output of 1 with a sigmoidal activation function. The optimizer selected, after multiple trials, is RMS with a learning rate of 2^{-6} and loss function binary crossentropy. The network was trained using a 4-fold cross validation that led to an average accuracy on the validation set of 0.61 with a standard deviation of 0.014. The following graphs show the trend of the accuracy and the loss in the 4 folds over the 40 epochs considered for each of them.



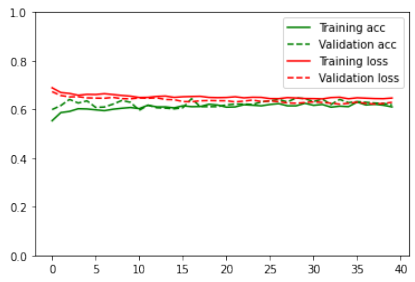
(a) Fold 1



(b) Fold 2



(c) Fold 3



(d) Fold 4

Figure 5.30: Inception accuracy and loss on k-fold

Looking at the graphs we can see that the accuracy on the validation set and on the training set go hand in hand and the same is true for the loss even if with small differences between a fold and the other linked to the random initialization of the weights. From this we can deduce the absence of overfitting and underfitting. The network was then retrained with 50 epochs to obtain a usable network on the test set and the resulting plot is as follows:

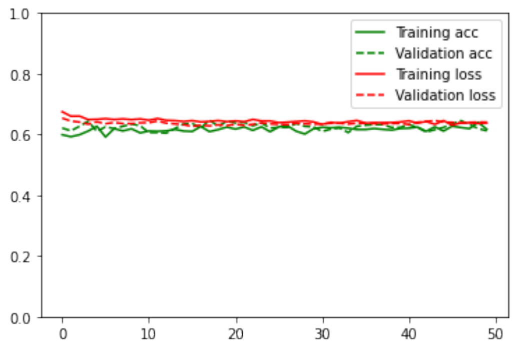


Figure 5.31: Inception test accuracy and loss

As in the previous tests, there is neither overfitting nor underfitting and testing the network on the test set achieves an accuracy of 61%. The resulting confusion matrix is as follows:

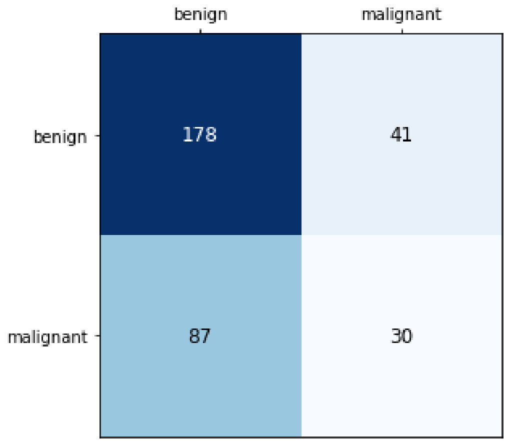


Figure 5.32: Inception test confusion matrix

As VGG16 and Resnet50, because of classes balancing, the error rate is not equally distributed between classes.

3.3.1 Fine Tuning

In order to improve the performance we performed fine tuning on the last convolutional layer of the network achieving the following results:

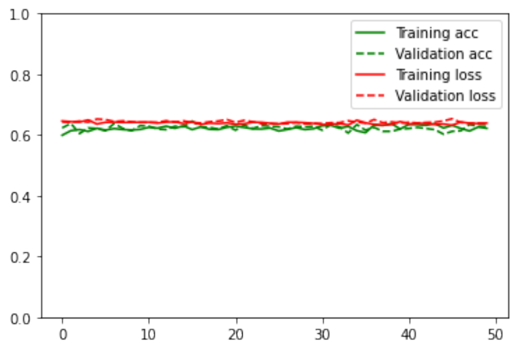
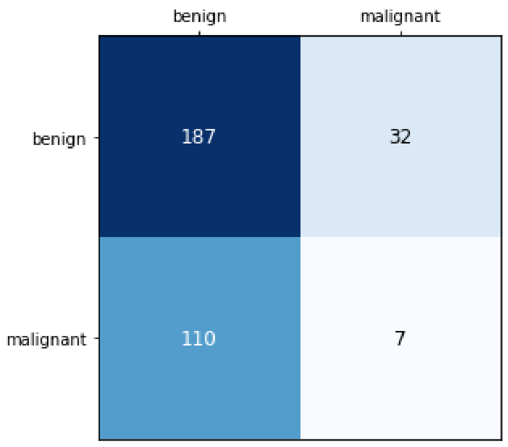


Figure 5.33: Fine Tuned Inception accuracy

As we can see the curve is very stable both on validation and training set and on test set achieved an accuracy of 0.58 with the following confusion matrix:



	benign	malignant
benign	187	32
malignant	110	7

Figure 5.34: Fine tuned Inception test confusion matrix

As can be seen from the confusion matrix, fine tuning had the effect of further unbalancing the network, which now classifies almost only benign and is almost totally unable to identify malignant samples. Such a network would be totally useless in the real world.

3.4 Comparisons

In order to identify which of the three models is the most balanced in the classification of the two classes, assuming that equal importance is given to the classification of both, ROC curves are used. The ROC curves for the non-fine tuned model are as follows:

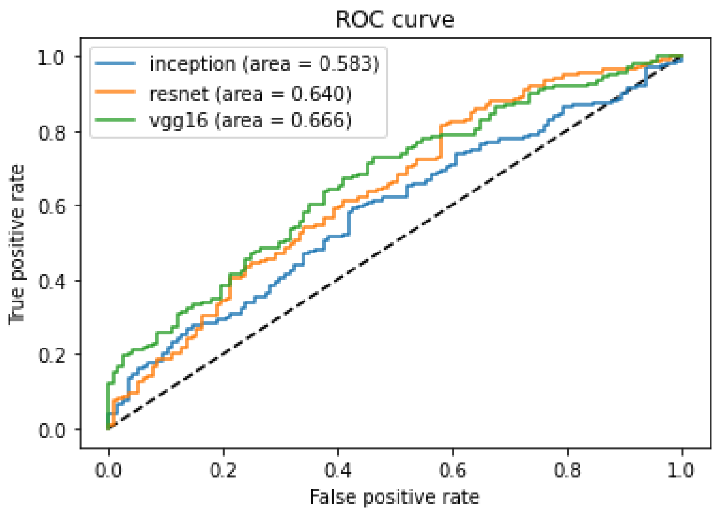


Figure 5.35: ROC Curves

As can be seen, the area under the curve of VGG16 (0.666) is higher than that of Resnet (0.640) and especially Inception (0.583).

Applying the same method on the fine tuned networks we obtain the following plot:

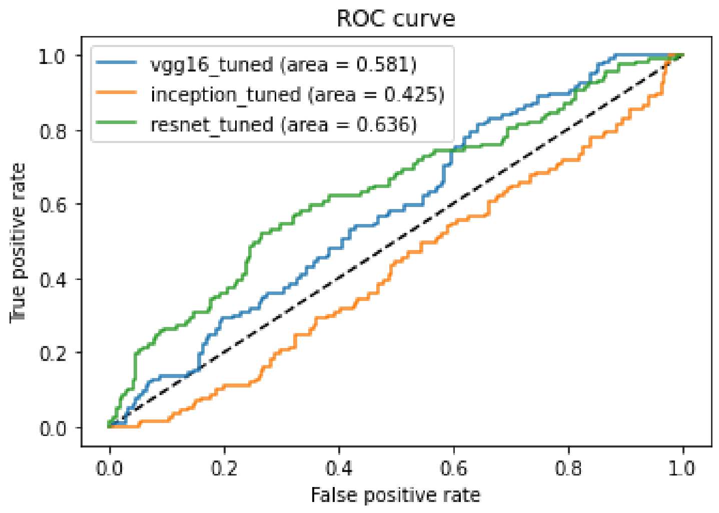


Figure 5.36: ROC Curves for Fine Tuned models

Although several levels of fine tuning have been applied, the original VGG16 model still appears to have the best AUC but with a very small difference.
A final comparison can be done on the obtained accuracies of all the models during the testing phase:

Model	Test Accuracy
VGG16	0.64
VGG16 Fine Tuned	0.61
ResNet	0.57
ResNet Fine Tuned	**0.65**
Inception	0.62
Inception Fine Tuned	0.57

Table 5.2: Classification results with the pretrained models (Benign/Malignant)

Task 4: Usage of Baseline Patches

In this phase of the project we focus on exploiting baseline patches to build a classifier capable of distinguishing calcifications and masses or benign or malignant cases, in the hope of obtaining better results by adding useful information to the network.

1 Siamese Network

To be able to exploit the baseline patches we used a structure known as Siamese network, often used in face-recognition but that could be useful also in our context. In particular it is composed by two neural networks with the same parameters. The global network has two images as inputs and ends up with a single output representing the predicted class. Both the images are first preprocessed through two convolutional neural networks that share the same weights and the extracted features are then composed and used as input for the final dense network classifier. In one branch of the network an image of baseline patches will be input, in the other branch a mass or calcification, or a benign or malignant case. The classification will be performed exploiting the information of the distance of the submitted abnormality image from the baseline image. The distances are calculated pixel by pixel providing the network with a large amount of useful information on the type of case to be classified. The network is trained to recognize masses and calcifications or benign and malignant, not on the basis of the image alone but on the basis of all distances of individual pixels from a baseline patch image.

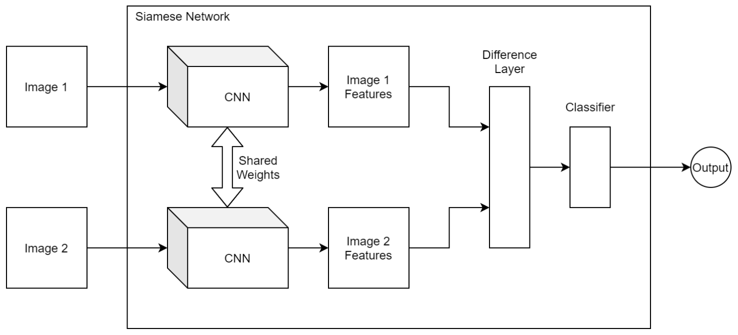


Figure 6.1: Siamese Network

2 Preliminary Operations

First, an analysis of visual differences between baseline patches and images containing abnormalities was performed. Below are two examples in which it is possible to see that there are different cases, some in which there is an obvious difference between a baseline image and one of calcification or mass while others in which the difference is almost imperceptible. The presence of these latter cases clearly negatively affects the performance of the networks.

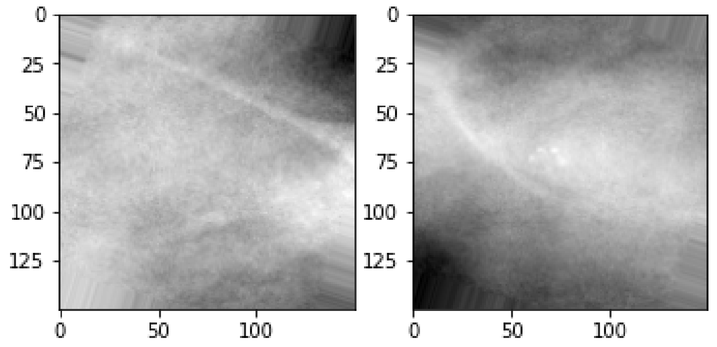


Figure 6.2: Baselines Patch and Abnormality not evident

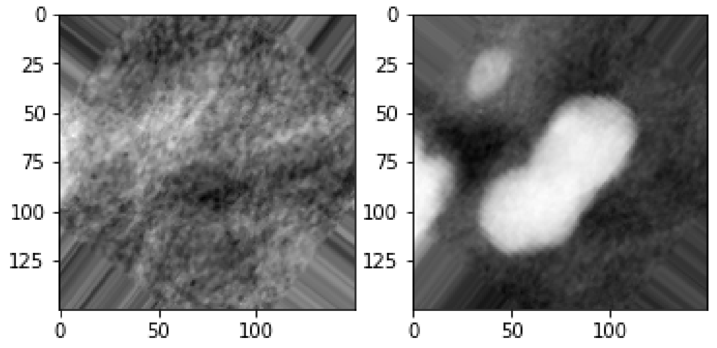


Figure 6.3: Baselines Patch and Abnormality evident

Also in this case it has been used data augmentation but to do it it has been necessary to realize a custom data generator able to generate data both for the baseline branch and for the abnormality branch. The result was obtained by declaring two different generators and making them work together using the code below:

```
while True:
    baseline = baseline_generator.next()
    abnormality = abnormality_generator.next()
    yield [baseline[0], abnormality[0]], abnormality
       [1]
```

Code 6.1: Custom Generator

A customized, non sequential, network has been generated to mimic the architecture of a siamese network. A single convolutional network has been used to extract the features from both the baseline and abnormality patches and then different versions of the difference layer were tested. The first version of the difference layer was the one performing the absolute difference between two tensors but this approach performed very bad.

```
L1_layer = layers.Lambda(lambda tensors:keras.backend.
   abs(tensors[0] - tensors[1])/65535)
L1_distance = L1_layer([encoded_baseline,
   encoded_abnormality])
```

Code 6.2: Subtraction Layer V1

The second and more successful approach removed the absolute value and improved significantly the accuracy of the models which exploits this function.

```
subtract_layer = keras.layers.Subtract()([
   encoded_baseline/65535, encoded_abnormality/65535])
```

Code 6.3: Subtraction Layer V2

3 Masses and Calcifications

3.1 Scratch CNN

Exploiting a similar architecture to the one implemented in Chapter 4, we employed a scratch CNN as a classifier. Having tried various structures, also exploiting dropout and regularization, the best model was identified consisting of three layers of convolutional followed by one of MaxPooling and another three layers of convolutional also followed by a MaxPooling. Then the output of this structure meets the baseline branch arriving on a difference layer and finally get the result based on the calculated distance.

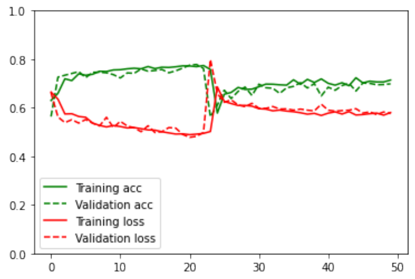


Figure 6.4: Siamese Scratch CNN accuracy and loss test

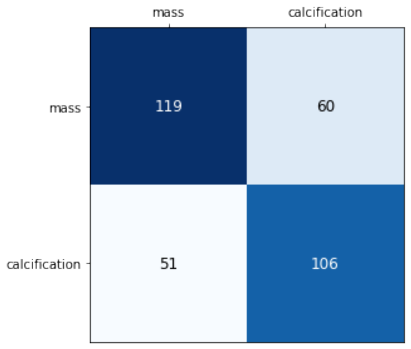

mass
calcification
mass
calcification
119
60
51
106

Figure 6.5: Siamese Scratch CNN test confusion matrix

As expected the accuracy of this model, 0.67, turns out to be much lower than previously trained pretrained networks due to the high number of parameters to train.

3.2 VGG16

The model was trained by adding to the pretrained CNN a difference layer followed by a fully connected one with an internal layer of 256 and an output of 1 with a sigmoidal activation function. The optimizer selected, after multiple trials, is Adam with a learning rate of 0.001 and as loss function binary crossentropy. The resulting accuracy on the test set is 0.81. The following plot shows the accuracy trends on validation and training and the confusion matrix:

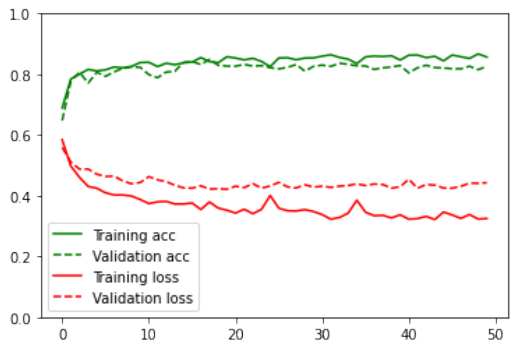


Figure 6.6: Siamese VGG16 accuracy and loss test

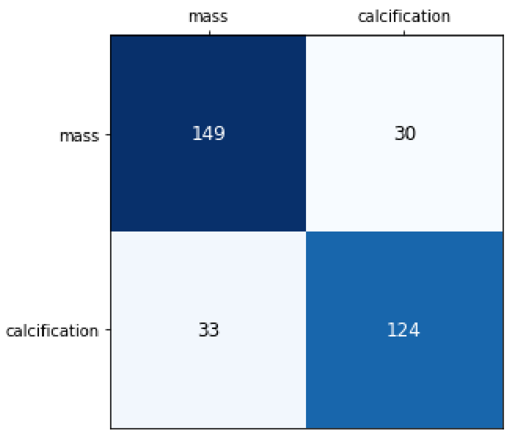


Figure 6.7: Siamese VGG16 test confusion matrix

3.2.1 Fine Tuning

In order to improve the performance we performed fine tuning on the last convolutional layer of the network achieving the following results:

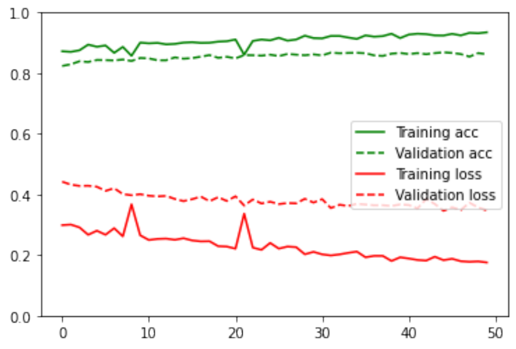
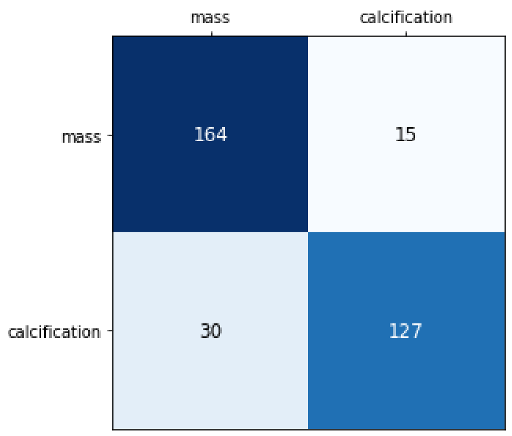


Figure 6.8: VGG16 fine tuning accuracy

As we can see from the plot the curve is quite stable both on validation and training set but the validation accuracy is less than the training accuracy. This is because there's a slight overfitting. The accuracy obtained on test set is 0.86 improving the accuracy of 0.05:



	mass	calcification
mass	164	15
calcification	30	127

Figure 6.9: Fine tuned VGG16 test confusion matrix

3.3 ResNet50

The model was trained by adding to the pretrained CNN a difference layer followed by a fully connected one with an internal layer of 256 and an output of 1 with a sigmoidal activation function. The optimizer selected, after multiple trials, is Adam with a learningrate of 0.001 and loss function binary crossentropy. The resulting accuracy on the test set is 0.84. The following plot shows the accuracy trends on validation, training and the confusion matrix:

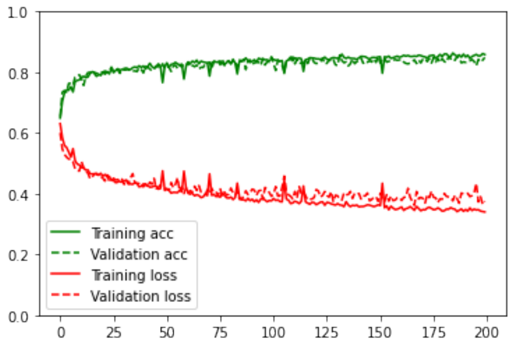


Figure 6.10: Siamese ResNet accuracy and loss test

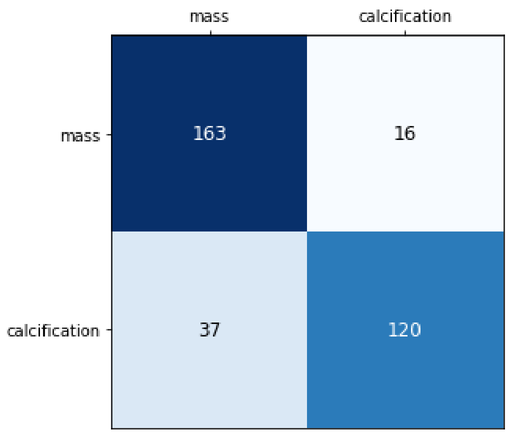


Figure 6.11: Siamese ResNet test confusion matrix

3.3.1 Fine Tuning

In order to improve the performance we performed fine tuning on the last convolutational layer of the network achieving the following results:

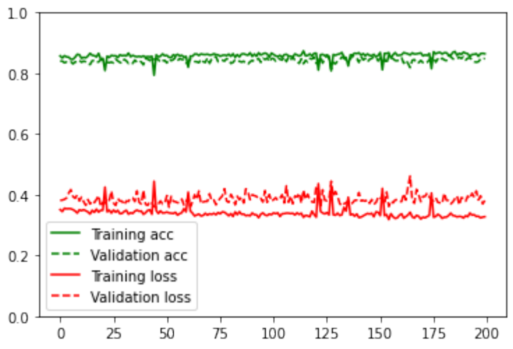


Figure 6.12: ResNet fine tuning accuracy

As we can see from the plot the curve is quite stable both on validation and training set. The accuracy obtained on test set is 0.83.

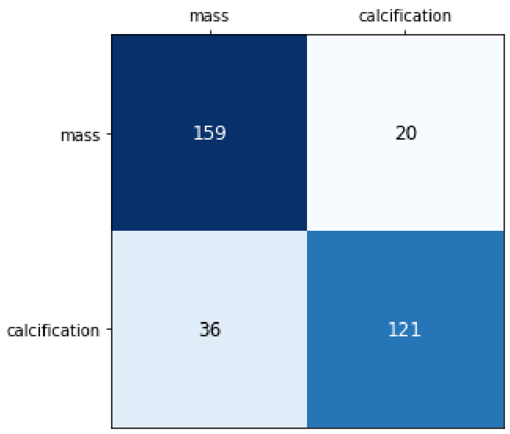


Figure 6.13: Fine tuned ResNet test confusion matrix

3.4 Comparisons

For the purpose of identifying the best model, ROC curves were used. The ROC curves for the non-fine tuned model are as follows:

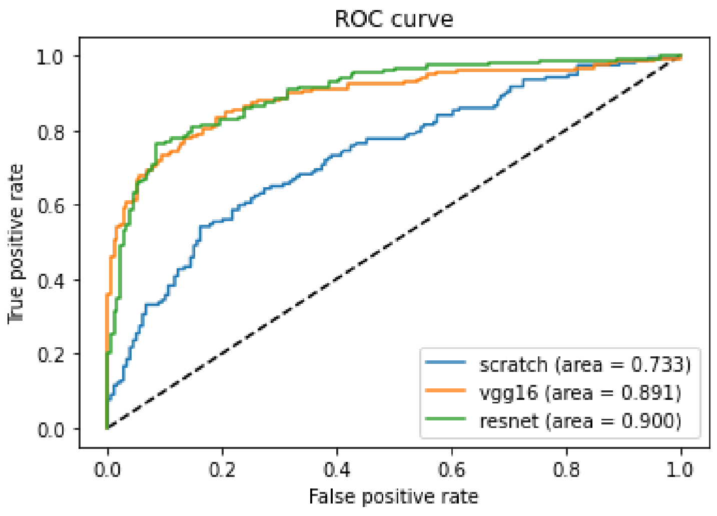


Figure 6.14: ROC Curves

As shown in the plot, the AUC of ResNet (0.900) is slightly higher than that of VGG16 (0.891) and very higher than Scratch (0.733) and it is therefore preferable to select this model for classification.

Applying the same method on the fine tuned networks we obtain the following plot:

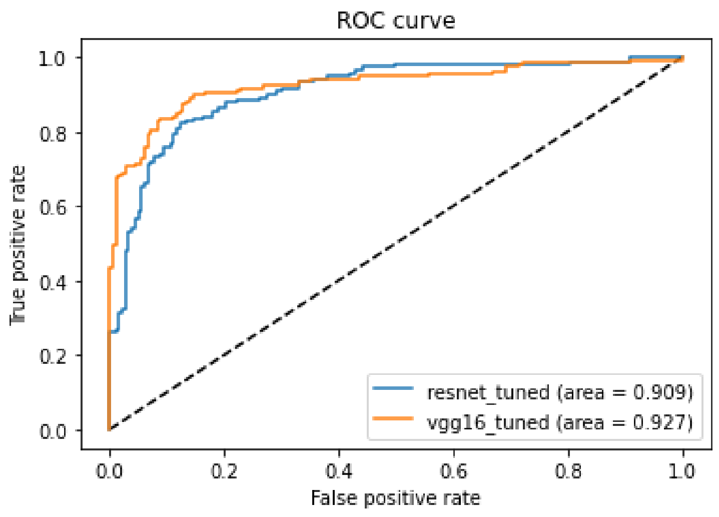


Figure 6.15: ROC Curves for Fine Tuned models

In this case there is a slight improvement in both ResNet and VGG16 but the last one now is the best one. Looking at the graph we can see slight differences in the area delimited by the two classifiers and therefore it would be the case to select one rather than another based on the specific needs of the individual classification task.
A final comparison can be done on the obtained accuracies of all the models during the testing phase:

Model	Test Accuracy
Scratch	0.67
VGG16	0.81
VGG16 Fine Tuned	**0.86**
ResNet	0.84
ResNet Fine Tuned	0.83

Table 6.1: Classification results with the siamese models (Mass/Calcification)

4 Benign and Malignant

4.1 Scratch CNN

Exploiting a similar architecture to the one implemented in Chapter 4, we employed a scratch CNN as a classifier. Having tried various structures, also exploiting dropout and regularization, the best model was identified consisting of three layers of convolutional followed by one of MaxPooling and another three layers of convolutional also followed by a MaxPooling. Then the output of this structure meets the baseline branch arriving on a difference layer and finally get the result based on the calculated distance.

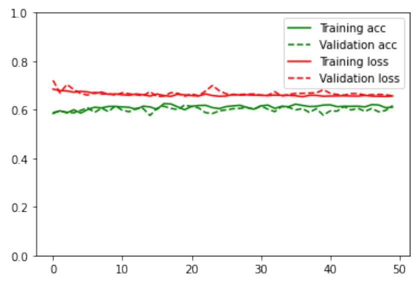


Figure 6.16: Siamese Scratch CNN accuracy and loss test

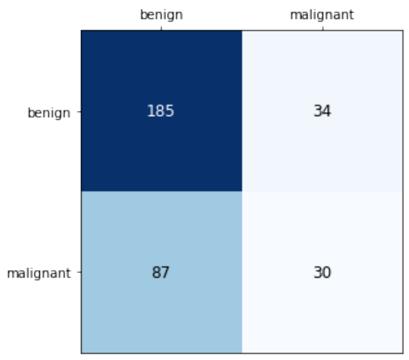


Figure 6.17: Siamese Scratch CNN test confusion matrix

As expected the accuracy of this model is 0.64, not a bad result considering the difficulty of the task and the characteristic of the dataset.

4.2 VGG16

The model was trained by adding to the pretrained CNN a difference layer followed by a fully connected one with an internal layer of 256 and an output of 1 with a sigmoidal activation function. The optimizer selected, after multiple trials, is Adam with a learningrate of 0.001 and loss function binary crossentropy. The resulting accuracy on the test set of 0.67. The following plot shows the accuracy trends on validation, training and the confusion matrix:

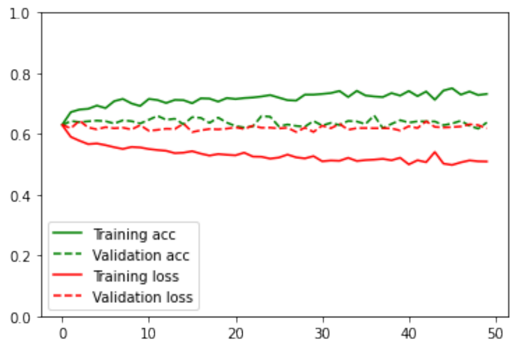


Figure 6.18: Siamese VGG16 accuracy and loss test

The model is overfitted in fact the accuracy on the training set is higher than the one on validation set and test set but the resulting accuracy is quite good for the task.

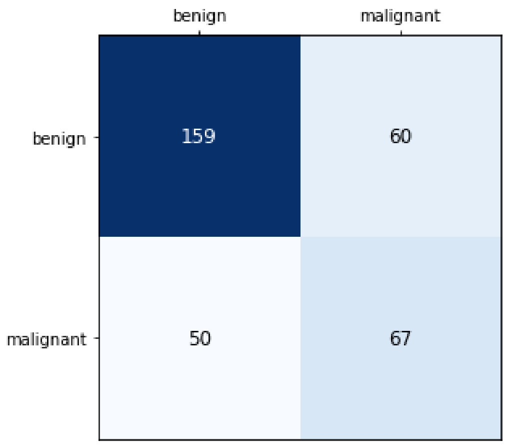


Figure 6.19: Siamese VGG16 test confusion matrix

4.2.1 Fine Tuning

In order to improve the performance we performed fine tuning on the last convolutional layer of the network achieving the following results:

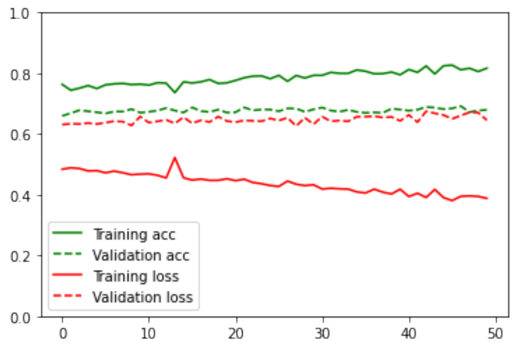


Figure 6.20: VGG16 fine tuning accuracy

As we can see from the plot the curve is quite stable both on validation and training set but the validation accuracy is less than the training accuracy. This is because there's a slight overfitting. The accuracy obtained on test set is 64% getting worse than the previous result:

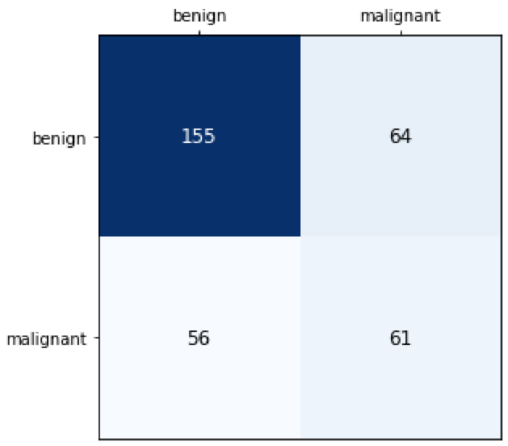


Figure 6.21: Fine tuned VGG16 test confusion matrix

4.3 ResNet50

The model was trained by adding to the pretrained CNN a difference layer followed by a fully connected one with an internal layer of 256 and an output of 1 with a sigmoidal activation function. The optimizer selected, after multiple trials, is Adam with a learningrate of 0.001 and loss function binary crossentropy. The resulting accuracy on the test set of 0.67. The following plot shows the accuracy trends on validation, training and the confusion matrix on test set:

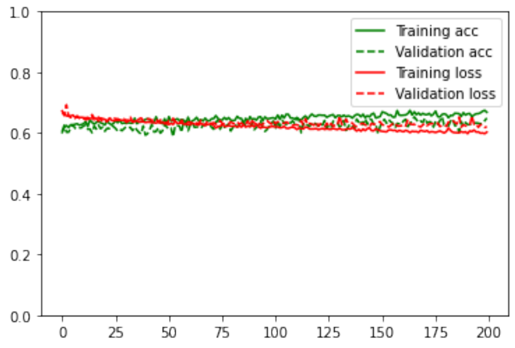
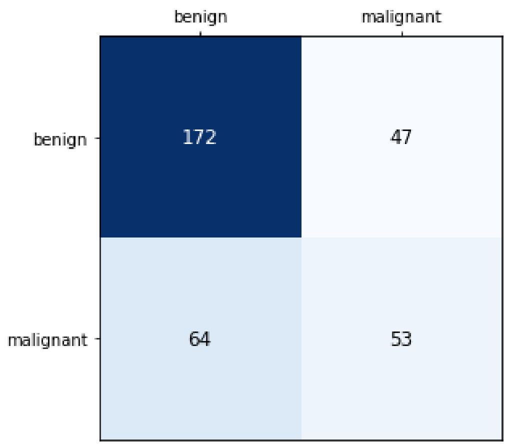


Figure 6.22: Siamese ResNet accuracy and loss test



	benign	malignant
benign	172	47
malignant	64	53

Figure 6.23: Siamese ResNet test confusion matrix

4.3.1 Fine Tuning

In order to improve the performance we performed fine tuning on the last convolutational layer of the network achieving the following results:

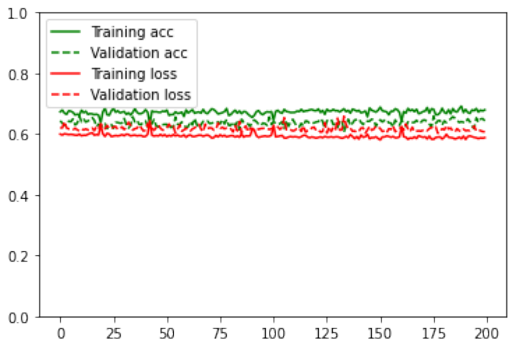
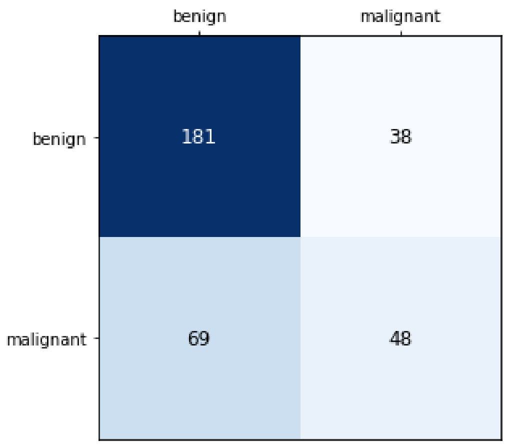


Figure 6.24: ResNet fine tuning accuracy

As we can see from the plot the curve is quite stable both on validation and training set. The accuracy obtained on test set is 0.68, increasing the accuracy of 0.01.



	benign	malignant
benign	181	38
malignant	69	48

Figure 6.25: Fine tuned ResNet test confusion matrix

4.4 Comparisons

For the purpose of identifying the best model, ROC curves were used. The ROC curves for the non-fine tuned model are as follows:

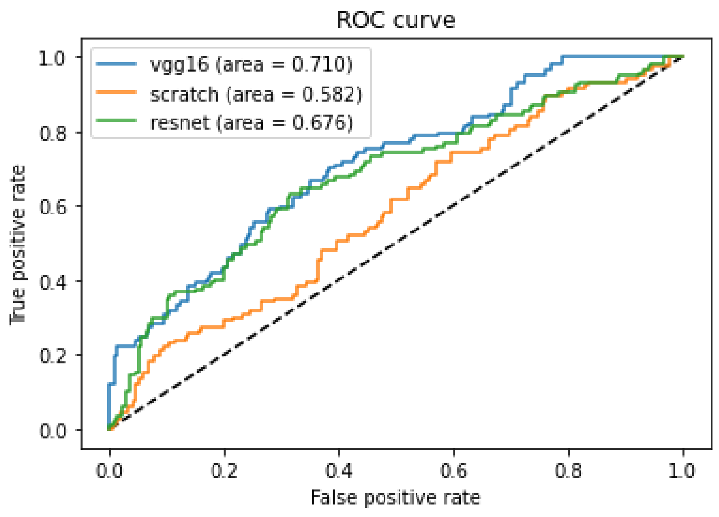


Figure 6.26: ROC Curves

As shown in the plot, the AUC of VGG16 (0.710) is higher than that of ResNet (0.676) and Scratch (0.582) and it is therefore preferable to select this model for classification.

Applying the same method on the fine tuned networks we obtain the following plot:

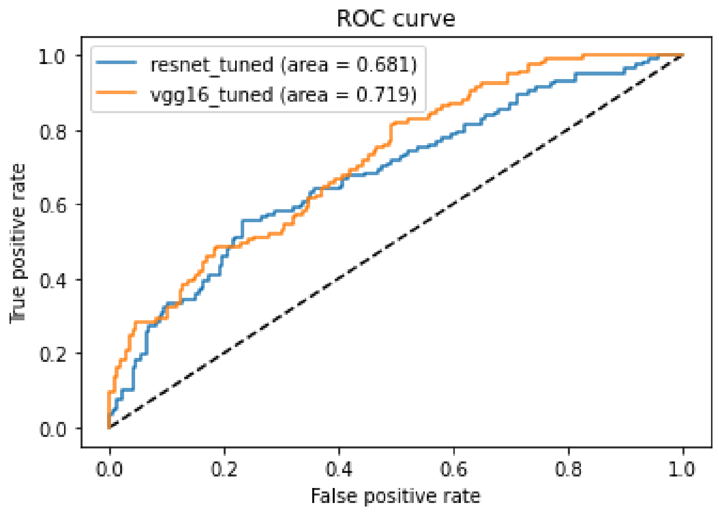


Figure 6.27: ROC Curves for Fine Tuned models

In this case there is a slight improvement in both ResNet and VGG16 but the last one continue to be the best one. Looking at the graph we can see slight differences in the area delimited by the two classifiers and therefore it would be the case to select one rather than another based on the specific needs of the individual classification task.
A final comparison can be done on the obtained accuracies of all the models during the testing phase:

Model	Test Accuracy
Scratch	0.64
VGG16	0.67
VGG16 Fine Tuned	0.64
ResNet	0.67
ResNet Fine Tuned	**0.68**

Table 6.2: Classification results with the siamese models (Benign/Malignant)

Task 5: Ensemble Classifier

1 Ensemble

To create our ensemble model, we consider two main approach:

- Stack model;
- Weighted Average Model.

We have trained three different set of models which we called it:

- Scratch: contains 4 models;
- Pretrained: contains 3 models;
- Tuned: contains 3 models.

Although, we only consider the Pretrained and Tuned models for creating of ensemble because they have higher accuracies and more reliable.

2 Stack Model

Basically, what we do in stack model is combining the predictions of multiple trained models and pass it to a shallow classifier. The shallow classifier can be KNN, SVM or MLP and tries to predict the label according to its input which is also the output of the trained models. However, the drawback of this model is that all trained classifiers have the same amount of contribution in predicating the final result. Thus, to overcome this issue, weighted ensemble model is introduced which we will explain about later. Generally, there are two main levels inside the stack model:

- Level 0: Which contains the multiple trained models which evaluate the input data. Finally, it passes the output to the level 1;
- Level 1: A shallow classifier which takes the output of the level 0 and tries to find the best combination level 0 outputs to classifies the original inputs.

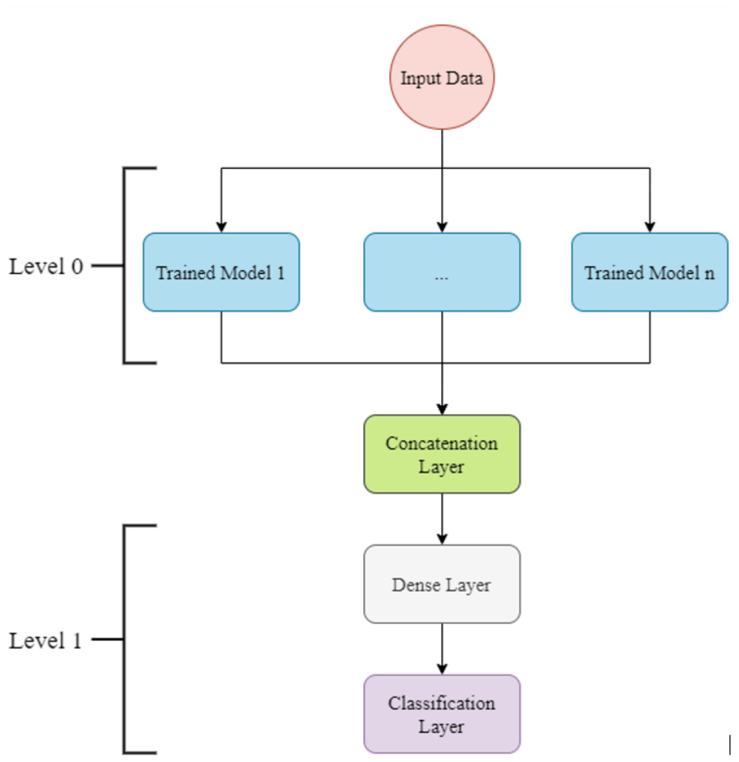


Figure 7.1: Stack Model Architecture

For level 0, we consider two set of models with three number of classifiers which we have trained in previous steps:

- Pretrained: containing Inception V3, Resnet50 and VGG16;
- Tuned: same as Pretrained set.

For these two sets we consider "sigmoid" as the activation of last layer since we exploited it in the training of pretrained and tuned models. We freeze all the layers of each model in level 0, because we have already trained those models and we do not want to repeat the same procedure again. Data preparation procedure (loading, labeling and augmenting) is same as we developed in previous steps. The number of epochs is 40 since we have already trained the models and we just want to find the best classification decision among them. To avoid the overfitting and get the minimum validation loss, we added the early function with three patience to the fitting process.

In the table below we can see the results in the case of Mass and Calcification classification:
*Note that we mention the accuracy of last epoch for training and validation

	Accuracy		
Model	Training	Validation	Test
Pretrained Stack Model	0.8094	0.8022	0.7798
Pretrained Stack Model Tuned	0.8708	0.8657	0.8423

The following plots shows the accuracy values on validation and training sets for masses and calcifications classification:

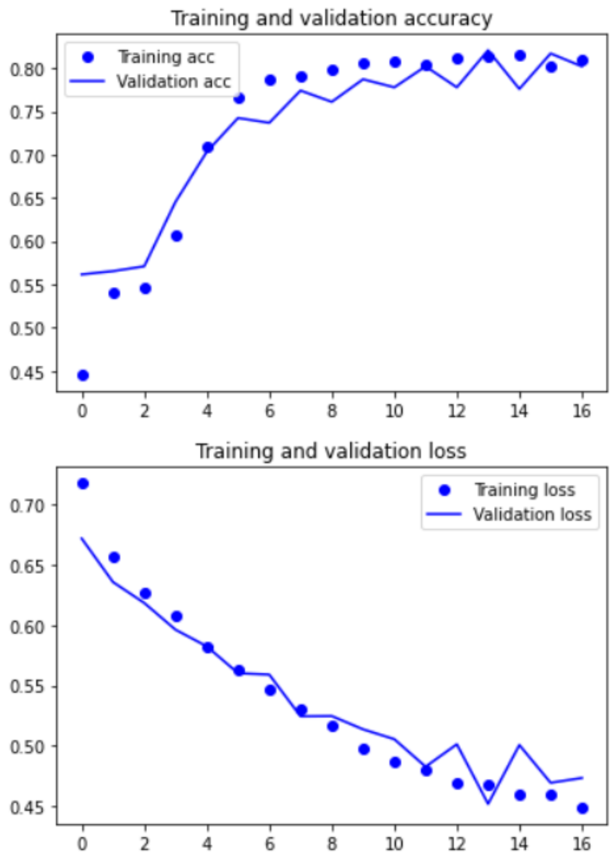


Figure 7.2: Accuracy and loss on training and validation sets

In the table below we can see the results in the case of Benignant and Malignant classification:
*Note that we mention the accuracy of last epoch for training and validation

	Accuracy		
Model	Training	Validation	Test
Pretrained Stack Model	0.6826	0.6530	0.6667
Pretrained Stack Model Tuned	0.6253	0.6194	0.6429

The following plots shows the accuracy values on validation and training sets for benign and malignant classification:

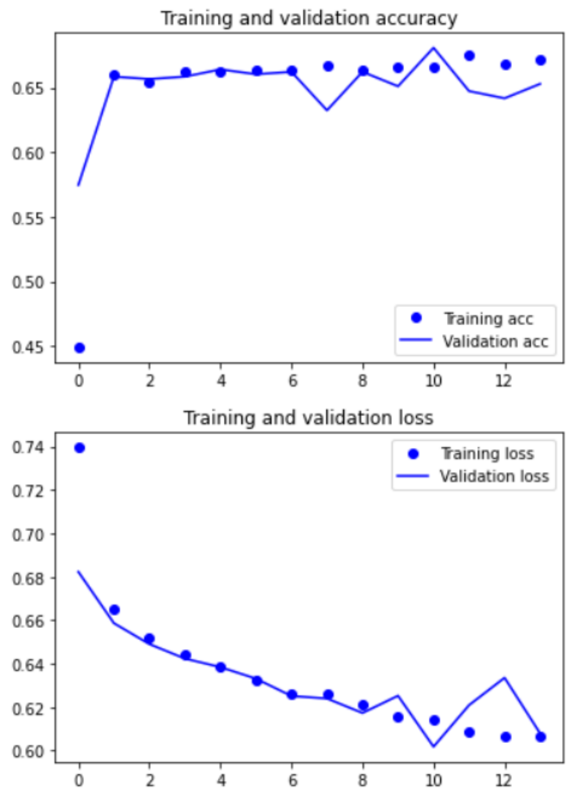


Figure 7.3: Accuracy and loss on training and validation sets

From the plots we can see that each training has different number of epochs, and the reason is that we used early stop with 3 number of patience. The trends of the accuracies are increasing while the loss trends are decreasing with respect to both validation and training. This shows our models are not overfitted and we can consider them as a reliable model.

3 Weighted Average Model

We start developing simple average ensemble to check the preliminary results. First, we evaluated the performance of each model (We have already trained two sets of models) on the test set. This evaluation is useful for comparing with the average ensemble model. The idea of the average ensemble is that each model predicts the probabilities for each class label. Then we summed up these all these probabilities from models inside of the set. Finally, if our activation is softmax we use the argmax function otherwise if it is sigmoid we use division and np.where(predicted $>$ 0.5, 1, 0) to return the class label. The following plot is the comparison between the accuracy of the pretrained models and average ensemble of pretrained set on Mass vs Calcification test dataset.

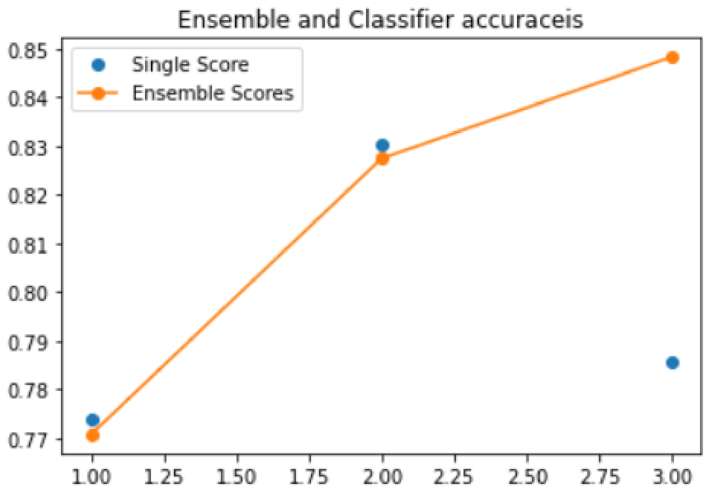


Figure 7.4: Average Ensemble classifier accuracies for Masses and Calcifications

The blue points are the accuracies of each model of pretrained set and the orange ones are the accuracy tracking of average ensemble. As we can see by increasing the number of models, our average ensemble outperformed the models.

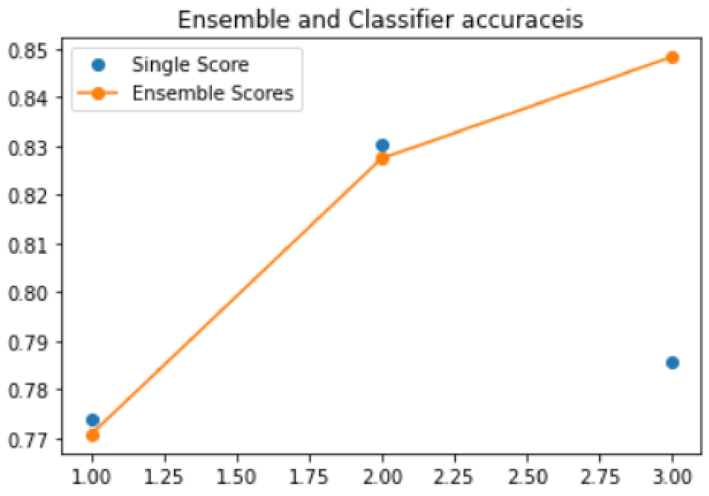


Figure 7.5: Average Ensemble tuned classifier accuracies for Masses and Calcifications

After developing the average ensemble for two sets (Pretrained and Tuned) in both Mass vs Calcification and Benignant vs Malignant, we implemented the weighted average one. We want to assign different weights to each classifier inside the set to improve the final classification accuracy. The procedure of obtaining the class label is similar to the average ensemble except during the summation we also multiply each classifier predictions to its weights. In order to find the best combination of weights, we developed grid search algorithm [2]. In grid search we initialized a vector of weights containing values in range [0 – 1]. Then we searched for the best set of weights which have the highest accuracy for our ensemble model. Due to our time limitation, we cannot test different values for initialization the weight vector. We initialized the weight vector with four values 0, 0.34, 0.67 and 1. In the results we can see the weights of the models inside each set and the accuracy of the ensemble model comprised by the set:

Mass vs Calcification

- Pretrained ensemble:
  - M1 = pretrained InceptionV3
  - M2 = pretrained ResNet50
  - M3 = pretrained VGG16

  Grid Search Weights: [M1 = 0.2, M2 = 0.4, M3 = 0.4], Accuracy: 0.8542

- Tuned pretrained ensemble:
  - M1 = pretrained InceptionV3 tuned
  - M2 = pretrained ResNet50 tuned
  - M3 = pretrained VGG16 tuned

  Grid Search Weights: [M1 = 0.0, M2 = 0.5, M3 = 0.5], Accuracy: 0.8780

Benign vs Malignant

- Pretrained ensemble:
  - M1 = pretrained VGG16
  - M2 = pretrained InceptionV3
  - M3 = pretrained ResNet50

  Grid Search Weights: [M1 = 0.0, M2 = 0.5, M3 = 0.5], Accuracy: 0.6667

- Tuned pretrained ensemble:
  - M1 = pretrained VGG16
  - M2 = pretrained InceptionV3
  - M3 = pretrained ResNet50

  Grid Search Weights: [M1 = 0.5, M2 = 0.5, M3 = 0.0], Accuracy: 0.6845

These results are promising and shows that if initialized the weight vector with more elements, we may find even better combination.

Add the Mass/Calcification information to the Benign and Malignant classification

1 Introduction

Since the task of classification of benign and malignant is particularly complex and having achieved results not particularly satisfactory we went in search of other possible approaches to improve the accuracy. The question we try to answer in this chapter is: ”Is it possible to improve the classification of benign and malignant providing among the various features also the type of abnormality (mass or calcification) considered? In this chapter we analyze the impact of including this additional feature on accuracy.

2 Approach A

2.1 Architecture

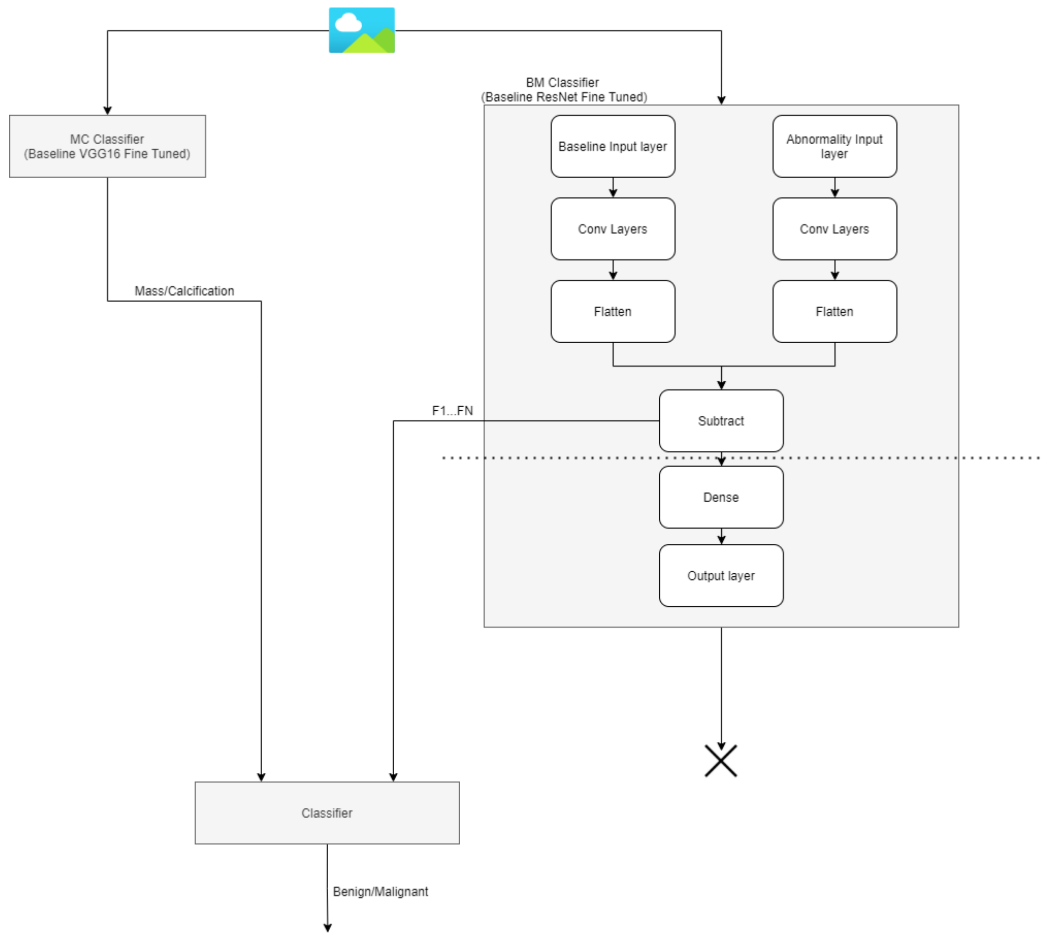


Figure 8.1: Proposed Architecture

To build the proposed architecture, the best models trained in previous experiments were selected, namely the VGG16-based mass and calcification classifier, with fine tuning and baseline patches, and the ResNet50-based benign and malignant classifier model also with fine tuning and baseline patches. The VGG16-based model is used to obtain a prediction on the image by classifying it as mass or calcification. The ResNet-based model is used to extract features by removing the last part of the network, then combining them with the class extracted by the other model and passing them all together to a new classifier. The hope is that providing this additional feature to a new model will allow for more accurate classification.

2.2 Results

Various tests have been carried out with different final architectures and exploiting or not the class weights and the structure resulted more effective has been that composed, in the final classifier, two Dense Layers one of 256 and one final of 1, with class weights of 1 for the benign and 1.3 for the malignant. In this way the following results were obtained:

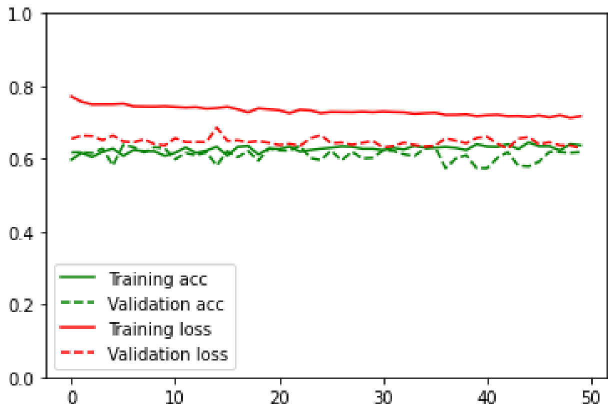


Figure 8.2: Model Accuracy and Loss on training and validation set

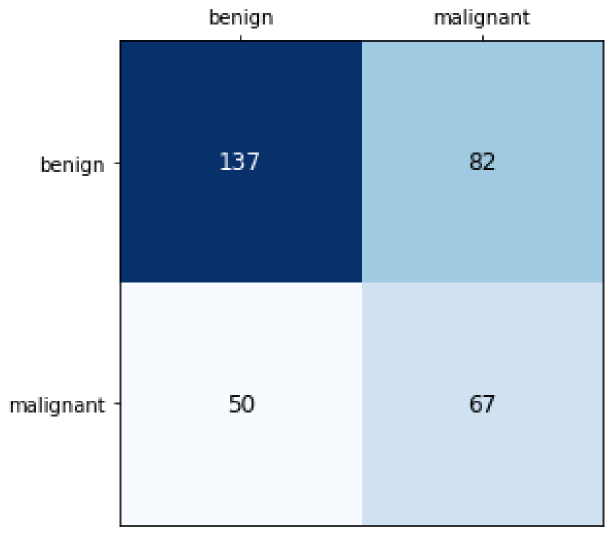


Figure 8.3: Model Confusion Matrix on test set

As can be seen from the graph, the curves are quite smooth and there does not seem to be any overfitting or underfitting when comparing the accuracy on the training set and that on validation. The confusion matrix is much more balanced than usual however the overall accuracy is not higher than those previously obtained. The value of accuracy on the test set is in fact 0.60, lower than other models previously trained and therefore this model should be used only if you are interested in a more balanced classification between the two classes.

3 Approach B

3.1 Architecture

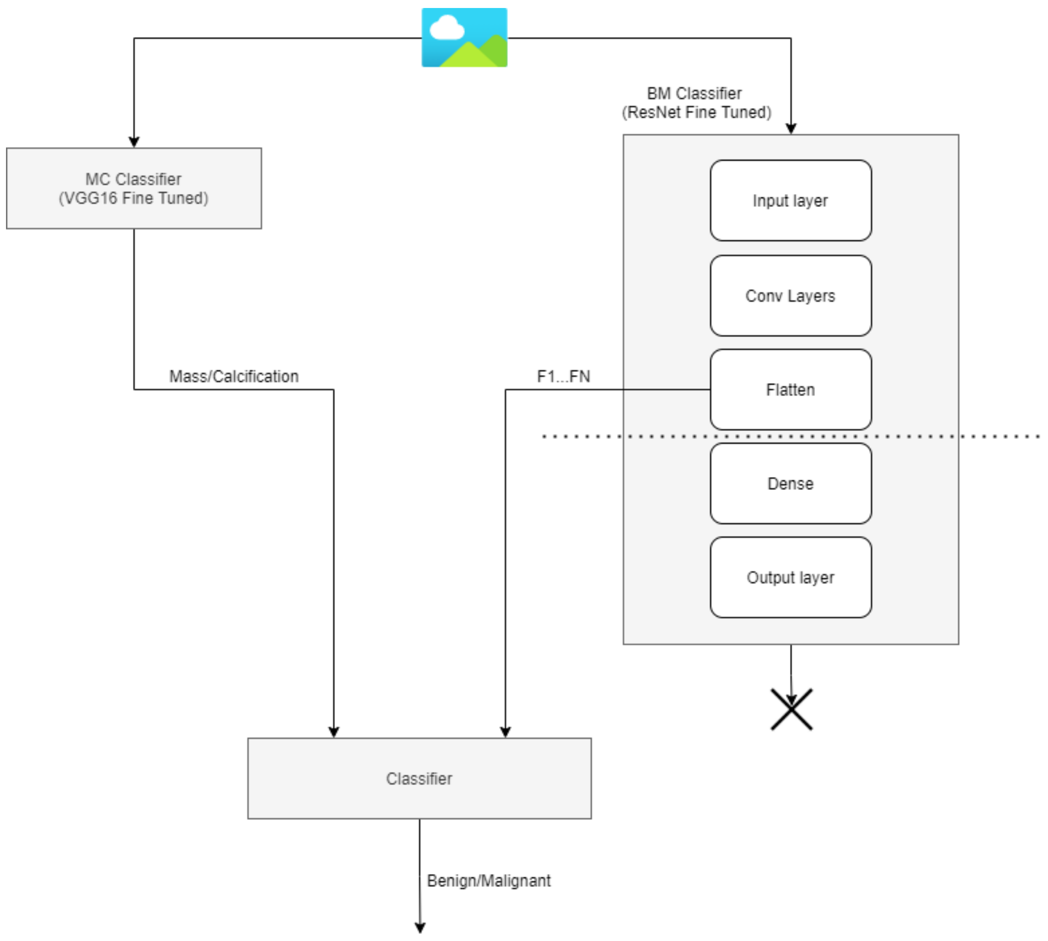


Figure 8.4: Proposed Architecture

In an attempt to improve the accuracy of the previous model we tried to couple the feature resulting from the classifier of masses and calcifications with those resulting from the normal pretrained network, without the use of baseline patches. Therefore we used the pretrained networks VGG16 fine tuned to classify masses and calcifications and ResNet50 Fine Tuned as classifier for benign and malignant from which we extracted the features from the level following the convolutional and combined with the label of mass and calcification.

3.2 Results

This experiment unfortunately did not lead to any improvement, managing to correctly identify only one of the two classes and therefore leading to unsatisfactory results, with low accuracy and unbalanced classifications.

4 Approaches Evaluation

The experiment if evaluated in terms of accuracy achieved did not lead to any improvement and the feature of mass and calcification was not particularly effective to correctly discriminate benign or malignant abnormalities. It seems from the experiment that discriminating between benign and malignant is independent of the type of abnormality that is being evaluated and using this information does not lead to real improvements in classification. However, other approaches could be tried in the future such as exploiting more accurate mass and calcification evaluations derived from human control rather than classification output, or other architectures with other combinations of features could be tried.

Final Considerations

Observing the various experiments conducted, it is evident that the task of classifying masses and calcifications is much simpler than that of benign and malignant. As a result of our experiments we find a better classification in terms of accuracy, in both tasks considered, of the approach using the baseline patches. In fact, the highest accuracy on the test set reached for Masses and Calcification was 0.86 with VGG16 Fine Tuned using the baseline patches and for Benign and Malignant 0.68 with ResNet50 fine tuned also using the baseline patches. The experiments conducted also allowed us to evaluate the relevance of the type of abnormality to define a malignant or benign and to analyze how various techniques may affect the classification.

Bibliography

[1] SanaUllah Khan et al. *"A novel deep learning based framework for the detection and classification of breast cancer using transfer learning"*. 2019. URL: https://www.sciencedirect.com/science/article/pii/S0167865519301059.

[2] Jason Brownlee. *"Weighted Average Ensemble for Deep Learning Neural Networks"*. 2020. URL: https://machinelearningmastery.com/weighted-average-ensemble-for-deep-learning-neural-networks/.

[3] Chris (Kitae) Kim. *"Abnormality Detection in Mammography using Deep Learning"*. 2020. URL: https://nycdatascience.com/blog/student-works/capstone/abnormality-detection-in-mammography-using-deep-learning/.

[4] Eric Antoine Scuccimarra. *"ConvNets for Detecting Abnormalities in DDSM Mammograms"*. 2018. URL: https://medium.com/@ericscuccimarra/convnets-for-classifying-ddsm-mammograms-1739e0fe8028.