UNIVERSITY OF PISA

MASTER DEGREE IN ARTIFICIAL INTELLIGENCE AND DATA ENGINEERING

DIPARTIMENTO DI INGEGNERIA DELL'INFORMAZIONE

Convolutional Neural Network for Medical Imaging Analysis : Abnormality detection in mammography

Authors
FEDERICO BALESTRI
GIULIO SILVESTRI

January 13, 2021

Contents

1 Introduction

The following paper presents a case study of implementing Abnormality Detection and Diagnosis in mammography using Convolutional Neural Networks (CNN). In the first part the most relevant works and state-of-art techniques related to the case study are presented, with particular attention regarding the diagnosis and tumor classification. In the second part various implementations and the corrispondent experimental results are reported.
According to recent studies[9], breast cancer is on the most dangerous type of cancer among women, being extremely common (accounting for 30% of newly diagnosed cancer in 2020) and having an high mortality rate (15% in 2020). For these reasons, being able to accurately detect and assess breast cancer in its early stages is crucial in order to fight this disease.
In order to help the decision-making process of radiologists, Computer-aided detection and diagnosis (CAD) systems were introduced and are largely used. Regarding mammography, these systems may address two different tasks: detection of suspicious lesions in a mammogram (CADe) and diagnosis of detected lesions (CADx). Deep learning represents a very valuable technology for implementing CAD systems, since deep learning methods are able to adaptively learn the appropriate feature extraction process from the input data with respect of a target output. The most commonly architecture used is the Convolutional Neural Network (CNN).
In image classification task, is essential to have a curated and sufficiently big dataset. Since most of the methods presented were tested on private datasets or unspecified subsets of public datasets, with the consequence of making hard to evaluate and compare the different solution proposed, in 2017 the CBIS-DDSM (Curated Breast Imaging Subset of DDSM) dataset has been introduced[5] as a standard.
Major challenges of abnormaly detection in mammography are the tumors dimensions themselves, that occupy only a small portion of the image of the entire breast, and the general lack of annotations for ROIs (Regions of Interest) in real life mammography datasets. Morover, the clinical significance of a False Negative (FN) for diseases is very high, so the model proposed should have higher sensitivity than the current standard of care.

2 Task 1: Abnormality Diagnosis state-of-art techniques report

When building a CNN, one of the main aspects to consider is if it is more appropriate to build the network From-Scratch or using Transfer Learning. All the paper analyzed demostrate that the second approach outperforms the first one. The most commonly used networks are:

- **AlexNet**[4]: the first CNN that exhibited performance beyond the state-of-the-art in the task of object detection and classification. It contains eight layers (the first five being convolutional and the remaining three being fully-connected). The network architecture is reported in the figure 1.

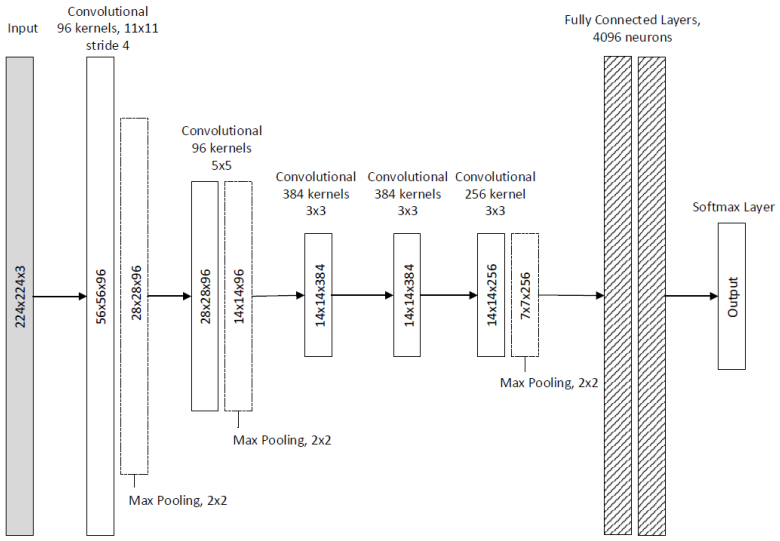


Figure 1: *AlexNet architecture.*

- **VGG**[10]: this network is considered an evoluted version of AlexNet. The number of layers was increased from 8 in AlexNet to 16 (VGG16) or 19 (VGG19), depending of the model. In general, VGG19 performs slightly better but consumes more memory. Figure 2 shows VGG architecture.

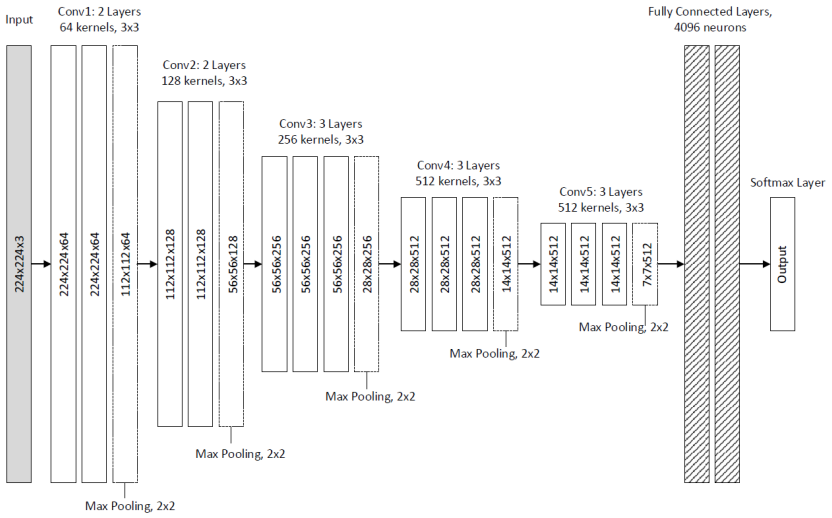
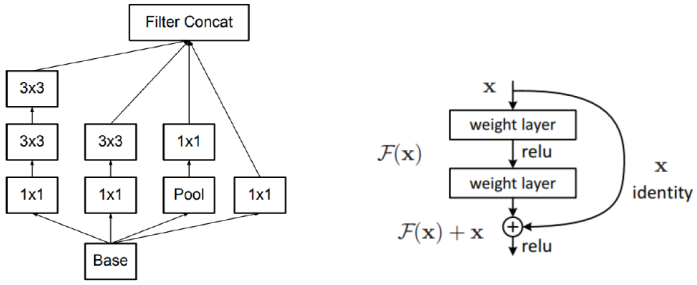


Figure 2: *VGG16 architecture.*

- **GoogLeNet**[11]: a very deep network characterized from some special modules called "inception modules" that make it very efficient in terms of computational cost, and allow the network to save a lot of parameters due to the absence of fully connected layers. The inception module currently used is "Inception v-2", reported in Figure 3a.
- **ResNet**[7]: implemented with the usage of “residual blocks". The block structure is reported in figure 3b.



(a) *Inception v-2 architecture.* (b) *Residual Block for ResNet.*

Figure 3: *The architecture of a Inception v-2 module(a) and a Residual Block (b)*

Regarding the classification between Begnin and Malignant masses starting from ROI patches, a paper from March 2019[13] draws a comparative study about some of the major results obtained in the previous years considering both

the possibilities of train From-Scratch and Fine-Tuining (with a pre-training on the ImageNet dataset[2]). Details about the specific adjustement can be found in the paper [13].
The following figure shows the results obtained by the networks in terms of AUC (Area Under the Curve) on the CBIS-DDSM dataset. The highest value of AUC (0.8) is obtained by AlexNet and ResNet-50.

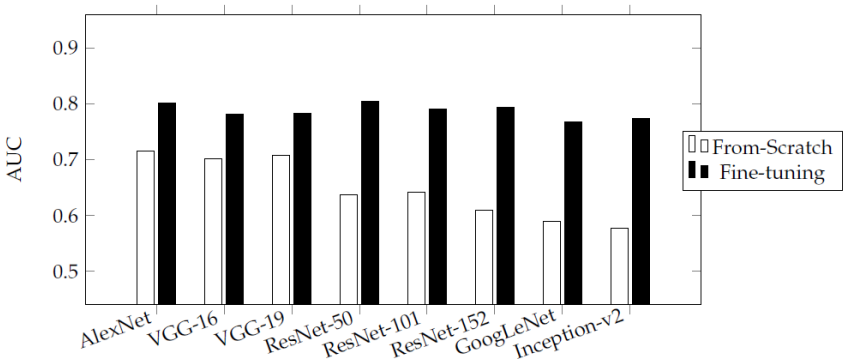


Figure 4: *Performance of convolutional neural networks for From-Scratch and Fine-Tuning scenarios in terms of AUC for CBIS-DDSM.*

For this classification problem, a recent study [3] showed a better result with the usage of transfer learning and fine-tuning of pre-trained network on ImageNet, applying also some specific preprocessing (increase image contrast) and data agumentation. Thanks to transfer learning (to improve accuracy), and in particular fine-tuning (to fight overfitting), the authors obtained an AUC of 0.844 with the VGG16-8-FT (fine tuned from layer 8) and an AUC of 0.818 with the VGG16-10-FT (fine tuned from layer 10) on a test set based on the CBIS-DDSM dataset.

In the idea of building a whole image classifier, an important 2019 study[8] demostrated the possibilities of improving the perfomances using pre-trained networks, in particular VGG16 e ResNet. The study followed a two step process: the goal is to first build a patch classifier and then an whole image classifier on top of it. For patches, two datasets has been extracted from the CBIS-DDSM: S1, that is a set of patches in which one is centered in the ROI and one is random background patch from the same image, and S10, in which there are 10 patches randomly sampled from around each ROI (minimum overlapping ratio of 0.9 with the ROI and inclusion of some background). Being not completely focused on the ROI, the S10 dataset is utilized in order to better capture the potentially informative region.
The patch classifier has been created using pre-trained networks (ResNet and

VGG16) on the ImageNet dataset. The following figure shows the results obtained on the S10 dataset.

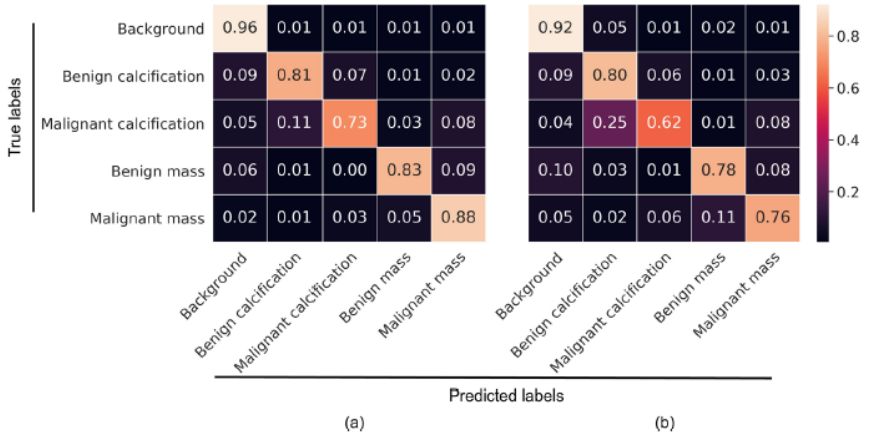


Figure 5: *Confusion matrix (normalized) of 5-class patch classification for Resnet50 (a) and VGG16 (b) in the S10 test set.*

The second step of the training process is to perform fine-tuning on the patch classifier obtained with ResNet and VGG16. Various configuration and ensemble models have been tested, with also the usage of many techniques like augmented prediction and model averaging, Max-pooling, Heatmap, ...
The study also showed that the model obtained with CBIS-DDSM dataset are also able to perform well on the INbreast[6] dataset with some fine-tuning, demostrating that the capabitilies of the models to learn general and useful features (for this task). The best models results are show in the following figure:

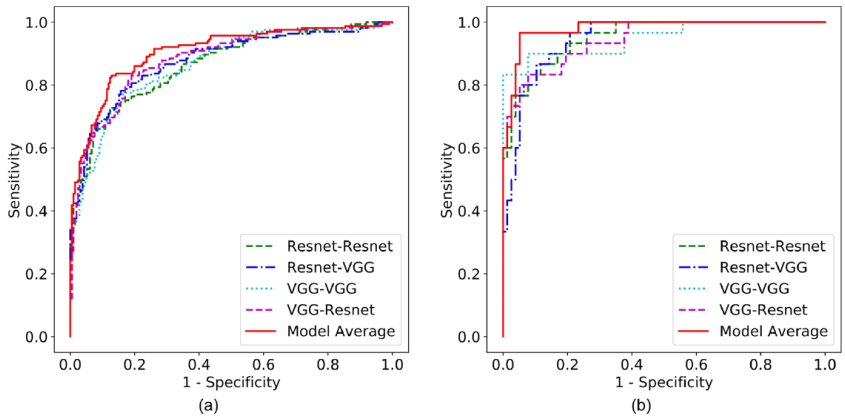


Figure 6: *ROC curves for the four best individual models and ensemble model on the CBIS-DDSM (a) and INbreast (b) test sets.*

3 CNN architecture for Abnormality classification: general methodology

For the development of CNN architecture, a general methodology has been followed in every experiment. We first developed some simple convolutional models, then moved towards more complex ones, capable of better capture the problem, and then added some regularization and others advanced techniques in order to improve the perfomances. Since for both the Mass/ Calcification and Benign/Malignant problems there is a pretty balanced classes distribution in the train set (1218 Masses and 1458 Calcifications, 1108 Benign and 1568 Malignant), no class rebalanced was performed at first. However, in order to improve the perfomance we introduced a strategy of class weighting in some experiments for Benign/Malignant classification.
For all experiments, we divided our train set into a training set and a validation set (validation split 20%), and evaluate the perfomance of the model on the test set. We adopted two callbacks: *"EarlyStopping"*, to monitor the validation loss and stop the training if it didn't change for a certain number of consecutive epochs (from 3 to 10, in general), and *"ModelCheckpoint"*, to save the network weights for the epoch with the lowest validation loss (considering all training epochs). This second callback sometimes allowed us to obtain a model that has not been trained too much, thus reducing the overfitting.
Regarding data agumentation, it has been observed that this strategy allowed the networks to obtain better results, but at the same time a too strong agumentation could provoke a phenomenon of underfitting, probably because the distortion of the images made the abnormalities unclear. For this reason, we didn't apply a very strong data agumentation and we mainly relied on small

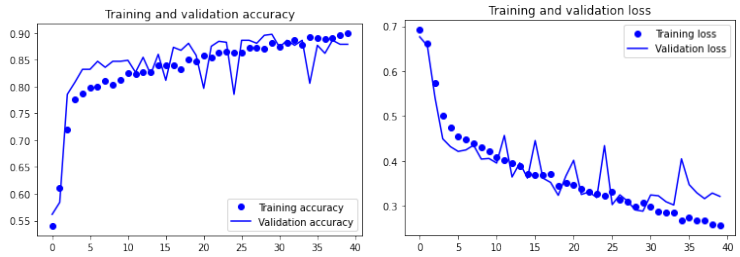
rotations, zooms, and vertical/horizontal flips.
As for evaluation methods, we manly relied on accuracy and confusion matrix (in a normalized form, where the TP represents the Recall for the positive class and the TN represents the Recall for the negative class). We also used ROC curves for classifier comparison.

4 Task 2: Design and develop an ad-hoc CNN architecture (training from scratch)

4.1 Task 2.1: train from scratch a network for Mass and Calcification

From the beginning, we noticed that this classification problem was not very hard and even some simple networks with few convolutional layers were able to obtain an accuracy of 0.75 on the test set. The following list reports the architectures tried and their best results.

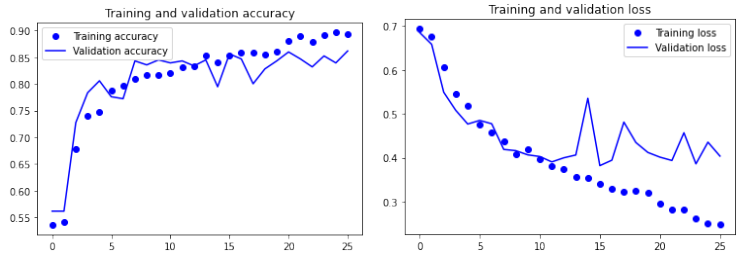
- **"Simple CNN"**: a simple CNN with 4 convolutional layers (filters going from 32 to 256). We tried learning rates from 1e-2 to 1e-5. The best accuracy obtained on the test set is 0.84.



(a) *Training and validation acc* (b) *Training and validation loss*

Figure 7: *Simple CNN (`"SimpleCNN_1_best.h5"`) accuracy and loss during training (learning rate: 1e-4). In the loss graph, we can notice some overfitting after epoch 30.*

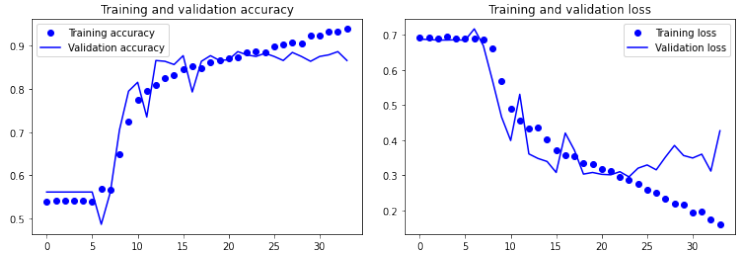
- **"SimpleVgg16"**: a simplified version of VGG16, with a reduced number of layers and parameters. The best accuracy obtained on the test set is: 0.8.



(a) *Training and validation acc* (b) *Training and validation loss*

Figure 8: *Simple Vgg16 (`"Simplevgg_1.h5"`)accuracy and loss during training (learning rate: 1e-4). Here we can notice some clear overfitting: after ten epochs the training loss keeps decreasing while the validation loss starts oscillating without decreasing anymore.*

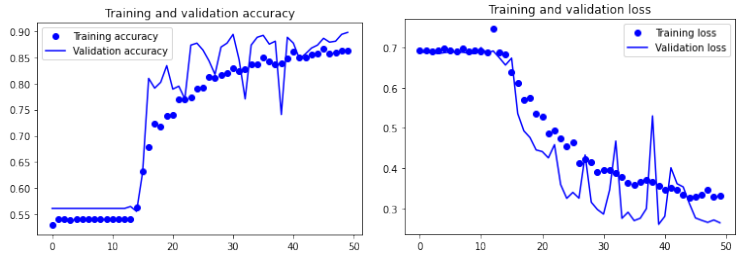
- **VGG16**: a VGG16 network with random weights. We tried learning rates from 1e-4 to 1e-5. The best accuracy obtained on the test set is: 0.82.



(a) *Training and validation acc* (b) *Training and validation loss*

Figure 9: *VGG16 (`"vgg16_1_best.h5"`) accuracy and loss during training (learning rate: 1e-5). Clear overfitting is present after epoch 25.*

Adding some data agumentation and dropout (0.5), we improve the perfomance of the VGG16 network. Figure 10 shows accuracy and losst during the training process.



(a) *Training and validation acc* (b) *Training and validation loss*

Figure 10: *Vgg16 (`"vgg16_4.h5"`) accuracy and loss during training (learning rate: 1e-4).*

This network was able to obtain an accuracy of 0.88 on the test set, with the following confusion matrix.

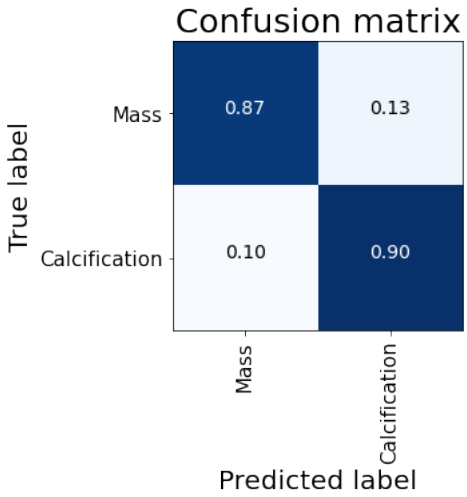
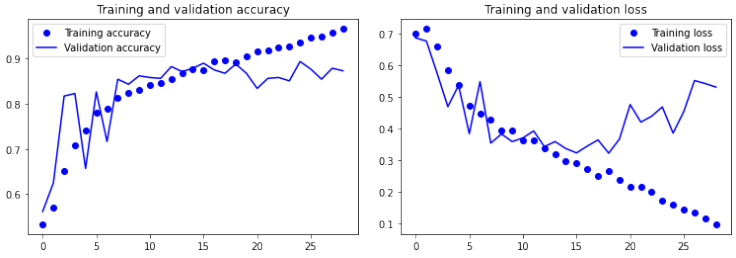


Figure 11: *Confusion matrix of VGG16*

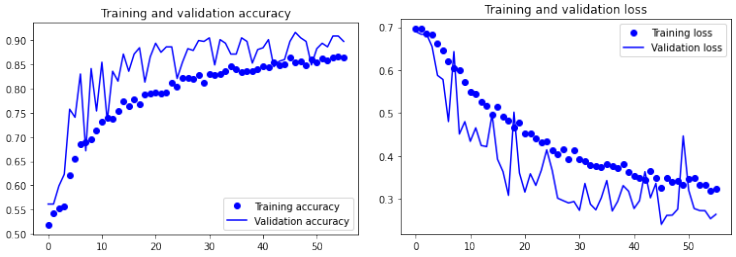
- **Simple AlexNet**: a simplified version of AlexNet,with 4 convolutional layer and kernel size reduced. The best accuracy obtained on the test set is: 0.86



(a) *Training and validation acc* (b) *Training and validation loss*

Figure 12: *AlexNet-like CNN (`"SimpleAlexNet_2_best.h5"`) accuracy and loss during training (learning rate: 1e-4). We can notice the overfitting after epoch 15.*

We improved the AlexNet network by adding some data agumentation and dropout (0.5), in order to reduce the overfitting. Some L2 regularization was also considered, as well as reducing/increasing the learning rate, but they weren't able to improve the perfomance.



(a) *Training and validation acc* (b) *Training and validation loss*

Figure 13: *AlexNet-like CNN (`"SimpleAlexNet_7_best.h5"`) accuracy and loss during training (learning rate: 1e-4). We can notice the network having slightly better results on the validation set than on the training set.*

In the end, the best accuracy obtained on the test set by this network is 0.88, with the following confusion matrix.

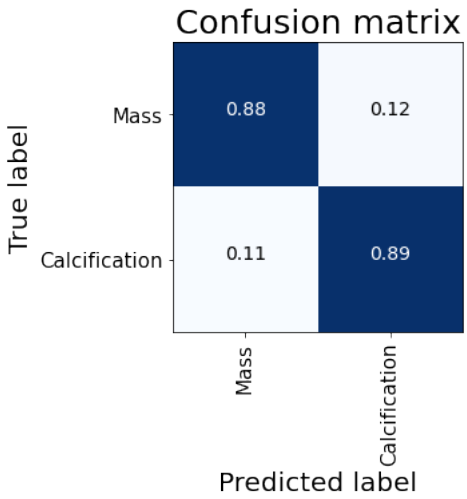


Figure 14: *Confusion matrix of SimpleAlexNet*

Considering that for the Mass/Calcification problem the missclassification error can be considered the same for both classes, none of the two best classifier (VGG16 (figure 10/11) and SimpleAlexNet (figure 13/14)) that obtained 0.88 accuracy can be considered better than the other. As shown in figure 15, no ROC curve is completely over the other.

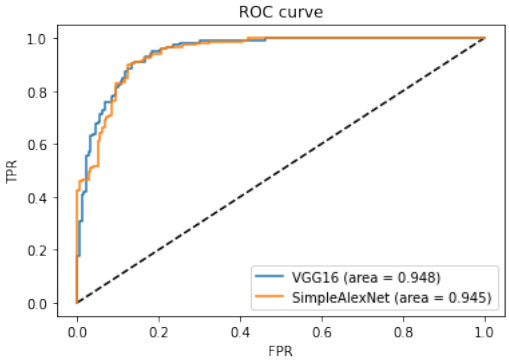


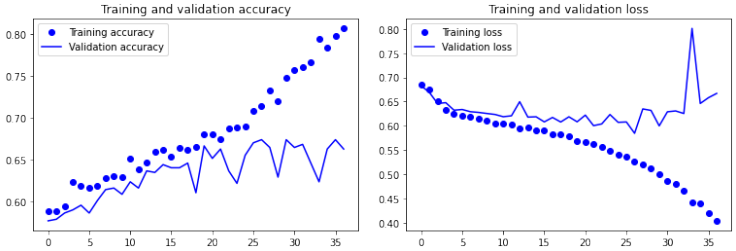
Figure 15: *ROC curves for VGG16 and SimpleAlexNet*

4.2 Task 2.1: train from scratch a network for Benign and Malignant

The Benign and Malignant classification problem presented itself as really complex. As a general consideration, we noticed that no network (even a complex one like VGG16) was able to learn much from the training, expecially with an

high learning rate. For this reason, we used lower learning rates in order to train the networks more slowly. The following list reports architectures and strategy used in our experiments:

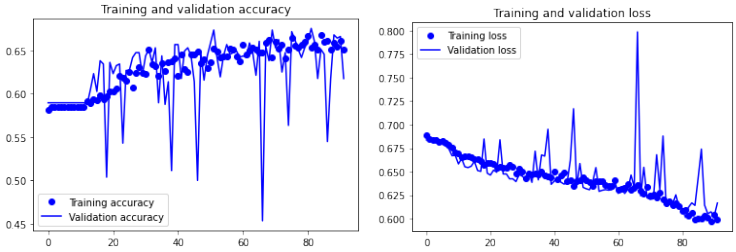
- **"Simple CNN"**: a simple CNN with 4 convolutional layers (filters going from 32 to 256). We tried learning rates from 1e-3 to 1e-5. The best accuracy obtained on the test set is: 0.63.



(a) *Training and validation acc* (b) *Training and validation loss*

Figure 16: *Simple CNN (`"SimpleCNN_4_best.h5"`) accuracy and loss during training (learning rate: 1e-4). We can notice a clear overfitting of the network from the first epochs.*

- **VGG16**: a VGG16 network with random weights. We tried learning rates from 1e-3 to 1e-5 and various batch sizes (from 32 to 80). The best accuracy obtained on the test set is 0.62. Data agumentation and dropout were not able to improve significantly the perfomances.



(a) *Training and validation acc* (b) *Training and validation loss*

Figure 17: *VGG16 (`"vgg16_4_best.h5"`)accuracy and loss during training (learning rate: 1e-5)*

Using this VGG16 network, the following confusion matrix has been obtained.

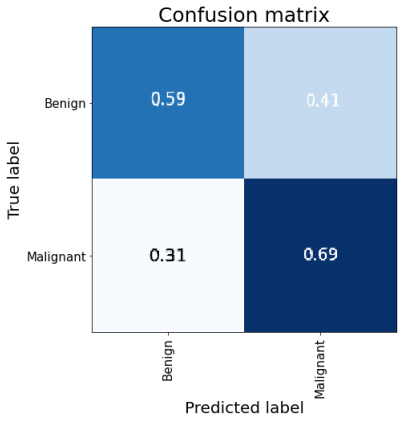
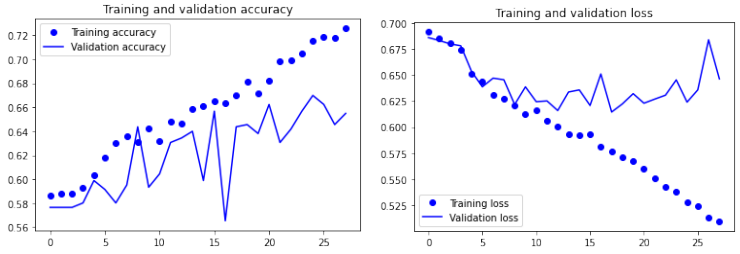


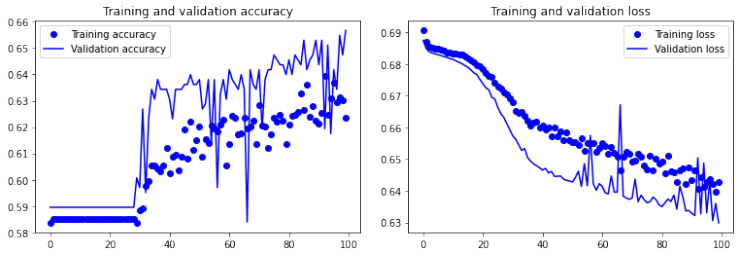
Figure 18: *Confusion matrix of VGG16*

- **Simple AlexNet**: a simplified version of AlexNet. The best accuracy obtained on the test set is 0.66.



(a) *Training and validation acc* (b) *Training and validation loss*

Figure 19: *Simple AlexNet (`"SimpleAlexNet_1_best.h5"`)accuracy and loss during training (learning rate: 1e-4). Notice some strong overfitting after epoch 5.*



(a) *Training and validation acc* (b) *Training and validation loss*

Figure 20: *Simple AlexNet (`"SimpleAlexNet_4_best.h5"`) accuracy and loss during training (learning rate: 1e-5) with data agumentation and dropout. We trained the netowork for an high number of epochs using a small learning rate.*

This last network was able to obtain a test accuracy of 0.68, with the following confusion matrix.

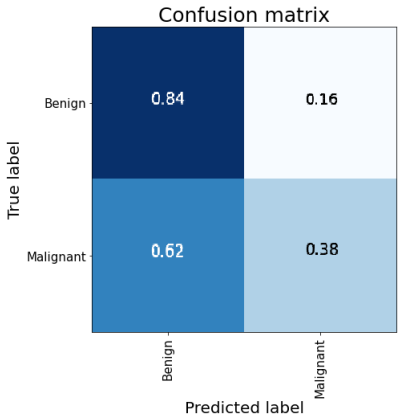


Figure 21: *Confusion matrix of SimpleAlexNet*

Despite having an overall lower accuracy (0.62) the VGG16 network (figures 17/18) can be considered the best since it was able to obtain a significantly better result in terms of discrimination of the Malignant class samples.

5 Task 3: Design and develop an ad-hoc CNN architecture (pre-trained networks)

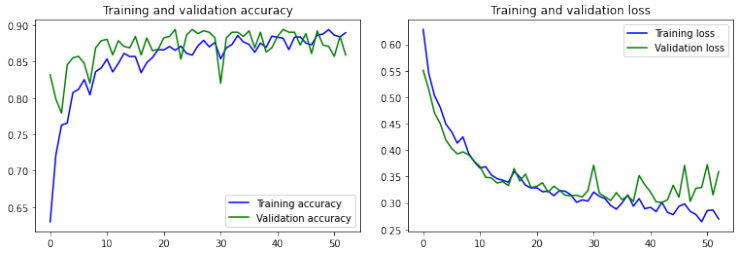
The following section reports the experiment carried out to develop a CNN for the two classification problems using pre-trained networks, following two strategies. In the first strategy (**Feature Extraction**) we taked an already trained network (on the *"Imagenet"* dataset), removed the top layers and added and

trained a classifier in the end, so we are able to exploit the general features already learned by the network to solve our classification problem. In general we added a fully connected layer followed by a classifier. The usage of a second fully connected dense layer (*ReLU, tanh*) made no observable difference to accuracy/loss plots or test metrics. The other approach used was **Fine Tuning**, in which we unfroze the last layers in a pre-trained network and then perform some new training with the mammography dataset.
The usage of pre-trained networks is most effective in cases where the new dataset is large and/or similar to the one used to pre-training, none of which apply to our case. As a matter of fact, the provided mammography dataset is not very big (2140 images) and it's very different from the *"Imagenet"* dataset. In the end, the results obtained with pre-trained networks were better to the ones obtained with CNNs trained from scratch (Task 2).

5.1 Task 3.1: pre-trained network for Masses and Calcifications

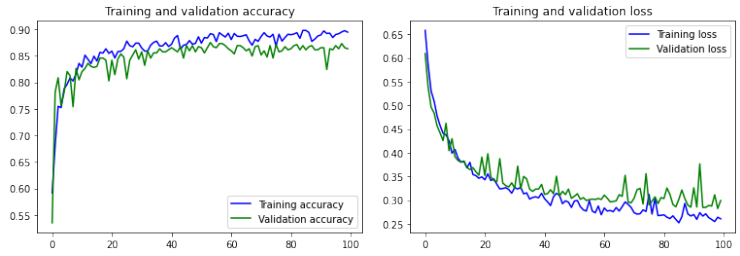
For the Mass/Calcificification problem, we used VGG16 and VGG19. Data agumentation and dropout have been implemented after few experiments because without them the networks were encountering some strong overfitting, having a training accuracy up to 20% higher than the validation one (same for training/validation loss). As previously explained, we didn't perform a very strong data agumentation because it would cause underfitting. As for dropout, we mostly setted it to 0.25, since we noticed some evident underfitting with values like 0.40 or 0.50. Regarding the learning rate, we mostly used values from 1e-4 to 1e-6. As a general rule, for Fine Tuning is expecially important to use a low learning rate. Batch sizes of 32 and 64 were tried: we noticed that a smaller batch size tends to cause higher oscillations amplitude in the training/validation accuracy and loss, but at the same time it allows the network to reach an higher accuracy.

- **"VGG16"**: Feature Extraction with various batch sizes. The best results were obtained with a batch size of 32: accuracy on the test set was 0.85.



(a) *Training and validation acc* (b) *Training and validation loss*

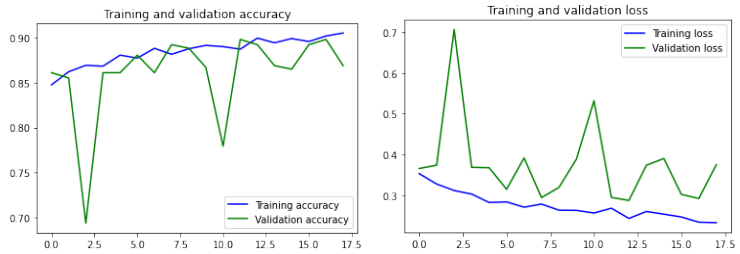
Figure 22: *VGG16 with Feature Extraction (`"vgg16_featextraction.h5"`) accuracy and loss during training (Optimizer: Adam; batch size: 32; learning rate: 1e-4)*



(a) *Training and validation acc* (b) *Training and validation loss*

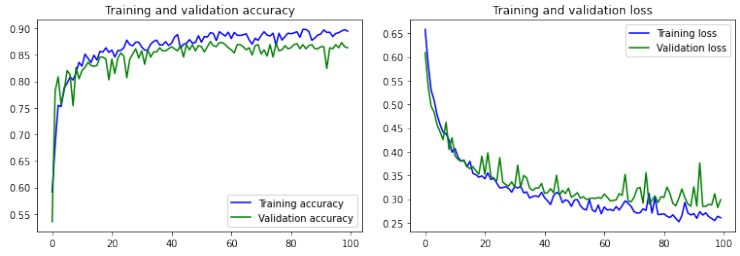
Figure 23: *VGG16 with Feature Extraction accuracy and loss during training (Optimizer: Adam; batch size: 64; learning rate: 1e-4)*

Fine Tuning experiments: We tried both unfrozing the last layer and the last two layers, obtaining slightly better results with the latter strategy.



(a) *Training and validation acc* (b) *Training and validation loss*

Figure 24: *VGG16 with Fine Tuning accuracy and loss during training (Optimizer: RMSprop; batch size: 32; learning rate: 1e-6). Notice the very strong oscillating behavior of the validation.*



(a) *Training and validation acc* (b) *Training and validation loss*

Figure 25: *VGG16 with Fine Tuning (`"vgg16_finetuned.h5"`) accuracy and loss during training (Optimizer: Adam; batch size: 64; learning rate: 1e-6)*

Using Fine Tuned VGG16 with optimizer Adam and batch size 64, we manage to get an accuracy score of 0.889 in the test set, with the following confusion matrix (figure 26)

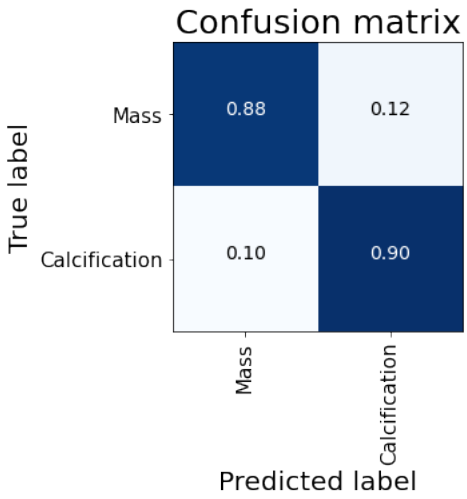
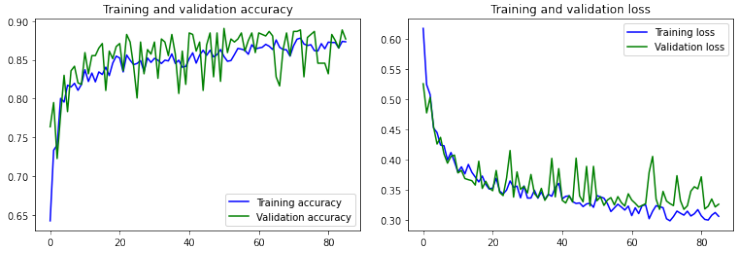


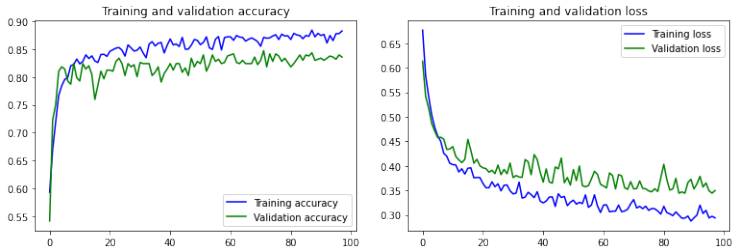
Figure 26: *Confusion matrix for FT VGG16* (`"vgg16_finetuned.h5"`)

- **"VGG19"**: Feature Extraction. The best results were obtained with a batch size of 64: accuracy on the test set was 0.842.



(a) *Training and validation acc* (b) *Training and validation loss*

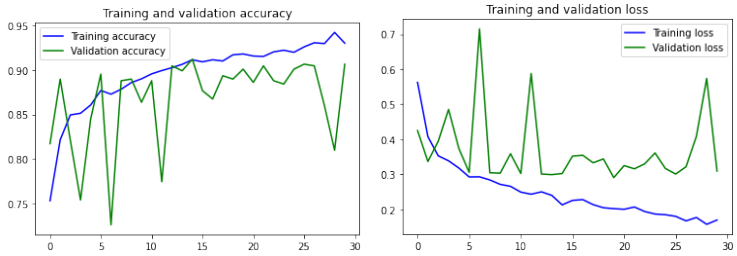
Figure 27: *VGG19 with Feature Extraction accuracy and loss during training (Optimizer: Adam; batch size: 32; learning rate: 1e-4)*



(a) *Training and validation acc* (b) *Training and validation loss*

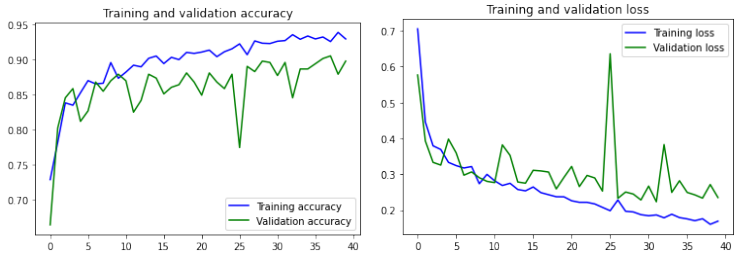
Figure 28: *VGG19 with Feature Extraction (`"vgg19_featextraction.h5"`) accuracy and loss during training (Optimizer: Adam; batch size: 64; learning rate: 1e-4)*

Fine Tuning experiments: Like with VGG16, we tried both unfrozing the last layer and the last two layers, obtaining slightly better results with the last strategy.



(a) *Training and validation acc* (b) *Training and validation loss*

Figure 29: *VGG19 Fine Tuning accuracy and loss during training (Optimizer: RMSprop; batch size: 32; learning rate: 1e-6). Notice the very high oscillations of the validationa accuracy/loss caused by a small batch size.*



(a) *Training and validation acc* (b) *Training and validation loss*

Figure 30: *VGG19 with Fine Tuning (`"vgg19_finetuned.h5"`) accuracy and loss during training (Optimizer: RMSprop; batch size: 64; learning rate: 1e-6)*

Using VGG19 and fine tuning, the best result obtained was 0.878, with the confusion matrix in the following figure.

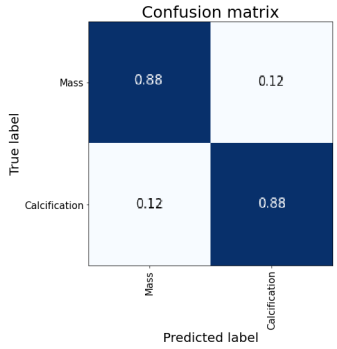


Figure 31: *Confusion matrix for FT VGG19* (`"vgg19_finetuned.h5"`)

Since we can assume that the misclassification cost for the Mass/Calcification problem is the same for both classes and the classifiers have a similar value of AUC (figure 32), we can't consider one as better than the other.

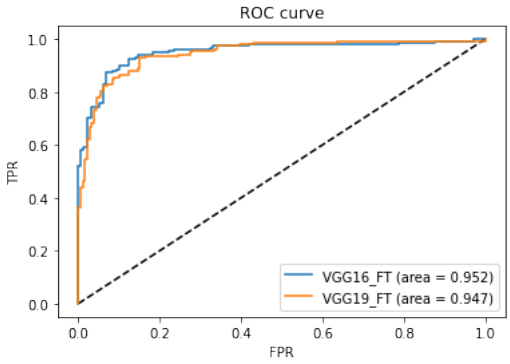
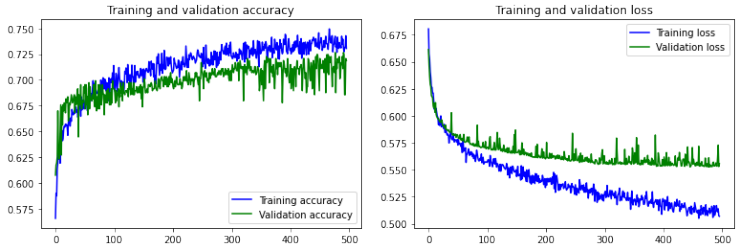


Figure 32: *ROC curves for VGG16 FT and VGG19 FT*

5.2 Task 3.2: pre-trained network for Benign and Malignant

For the Benign/Malignant problem, pretrained networks VGG16 and VGG19 has been used. The same considerations about data agumentation and dropout that were true for the Mass/Calcification problem also applied here. As we saw in the trained from scratch network experiments, the major problem for this classification task is the inability of the networks to learn from the dataset. For this reasons we tried to train the network for a very high number of epochs and a low learning rate, in order to let the CNN to learn as much as possible.

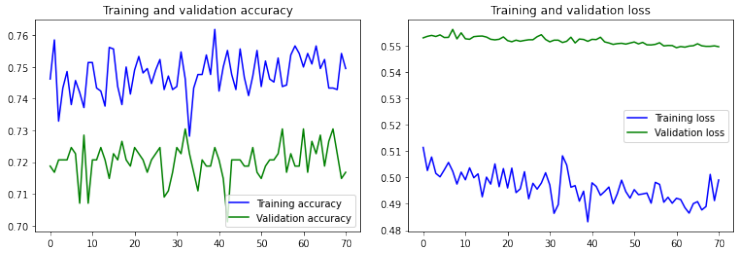
- **"VGG16"**: Feature Extraction with various batch sizes. The best results were obtained with a batch size of 32: accuracy on the test set was 0.71.



(a) *Training and validation acc* (b) *Training and validation loss*

Figure 33: *VGG16 with Feature Extraction (`"VGG16_featextraction.h5"`) accuracy and loss during training (Optimizer: RMSprop; batch size: 32; learning rate: 1e-5). We can notice that after about 50 epochs validation accuracy starts increasing very slowly. From the loss plot, we can see that the some overfitting is present*

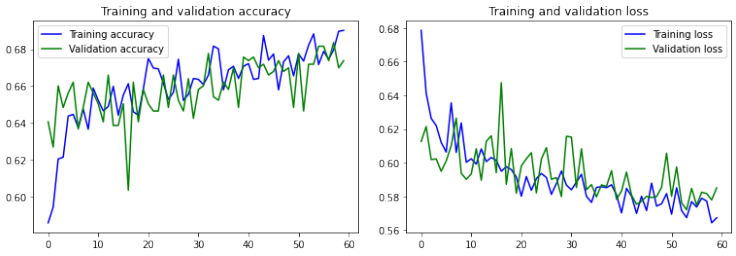
Fine Tuning experiments We tried both unfrozing the last layer and the last two layers. In this case the results were slightly inferior with respect of the feature extraction strategy.



(a) *Training and validation acc* (b) *Training and validation loss*

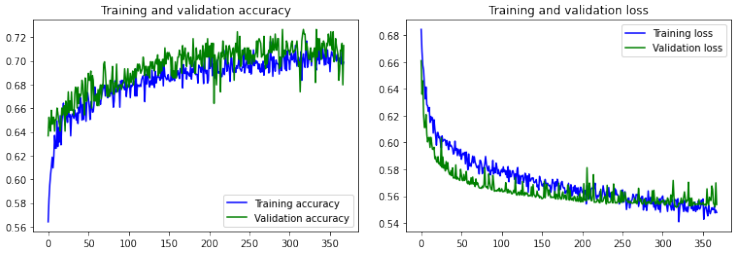
Figure 34: *VGG16 with Fine Tuning (`"VGG16_finetuned.h5"`) accuracy and loss during training (Optimizer: RMSprop; batch size: 32; learning rate: 1e-7). Validation accuracy and loss keep oscillating without any significant change for all the training process.*

- **"VGG19"**: Feature Extraction with various batch sizes. The best accuracy on the test set was 0.684, obtained with RMSprop optimizer and batch size 64 (figure shows the training process).



(a) *Training and validation acc* (b) *Training and validation loss*

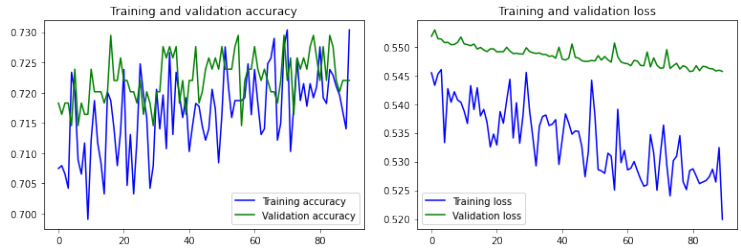
Figure 35: *VGG19 with Feature Extraction accuracy and loss during training (Optimizer: Adam; batch size: 32; learning rate: 1e-5)*



(a) *Training and validation acc* (b) *Training and validation loss*

Figure 36: *VGG19 with Feature Extraction (`"VGG19_featextraction.h5"`) accuracy and loss during training (Optimizer: RMSprop; batch size: 64; learning rate: 1e-5)*

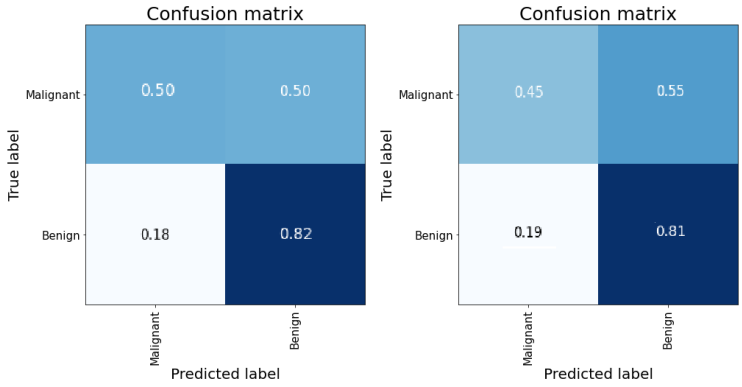
Fine Tuning experiments. The result obtained were slightly inferior with respect of the feature extraction. The following figure shows that validation loss and accuracy tends to not vary significantly during training.



(a) *Training and validation acc* (b) *Training and validation loss*

Figure 37: *VGG19 with Fine Tuning accuracy and loss during training (Optimizer: RMSprop; batch size: 32; learning rate: 1e-6)*

In the end, the highest accuracy (0.71) has been obtained with VGG16 and Feature Extraction, while the highest accuracy obtained with VGG19 (feature extraction) is 0.6845.



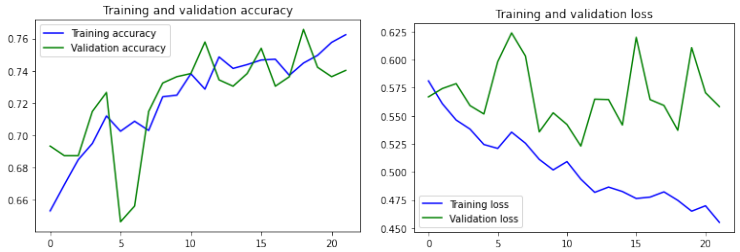
(a) *Confusion matrix VGG16* (b) *Confusion matrix VGG19*

Figure 38: *Confusion matrixes obtained from the best networks using feature extraction (`"VGG16_featextraction.h5"`) (`"VGG19_featextraction.h5"`). Unfortunately both the matrixes present a low value of recall for the Malignant class.*

Weighting classes experiments and initial bias Since the Malignant class can be considered more important to identify and it also represents the class with a lower amount of samples in the training set (1108 malignant vs 1568 benign), some class weighting has been implemented considering the class proportion in the training set.

We perfomed Fine Tuning adopting a solution inspired by one of the paper previosuly mentioned[8] called "3-Stage training", in which the parameter learning is frozen for all but the final layer and progressively unfrozen from the

top to the bottom layers, while simultaneously decreasing the learning rate. An initial bias[12] for the classifier has been introduced in order to help with initial convergence; without this bias, the network would have to lose some epochs to learn it. The initial bias can be computed as: $b_0 = \log_e(pos/neg)$



(a) *Training and validation acc* (b) *Training and validation loss*

Figure 39: *VGG16 with Fine Tuning and Weighted Classes (`"VGG16_finetuned3step.h5"`) accuracy and loss during training (Optimizer: Adam; batch size: 64)*

With this strategy, we obtained an accuracy of 0.71 with an high recall value for Malignant, as shown in the following confusion matrix (figure 40).

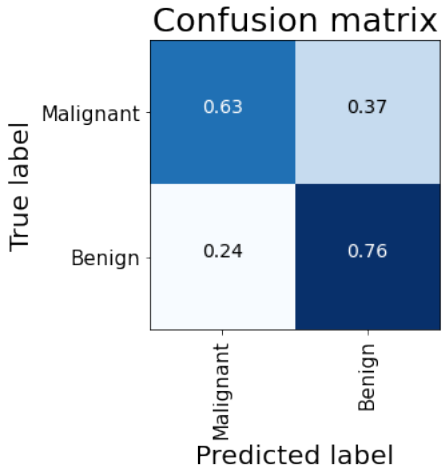
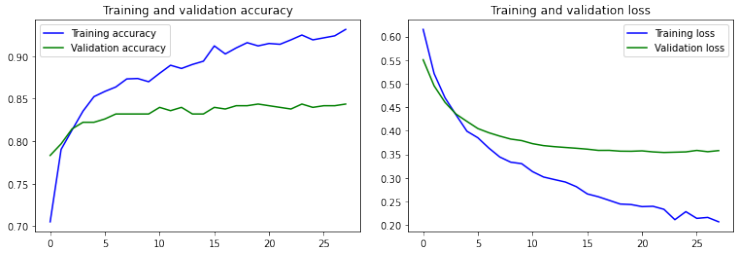


Figure 40: *Confusion matrix for FT VGG16 with 3 stage tuning*

6 Task 4: Classification exploiting baseline patches

Since the dataset provided contained both abnormalities and baseline patches (healty tissue), we tried to exploit the latter in order to obtain a better classifier. The idea, based on the Siamese network architecture (also called Twin), is to extract features from both types of images using a pretrained VGG16 (*"imagenet"* dataset), and then use a subtract layer to remove the baseline patches features from the abnormalities ones. In this way, only the features related to the tumors themselves should remain. In order to implement this idea, we create a training set containing couples of corrispondent images: baseline and abnormality.
Unfortunately, the above strategy didn't allow us to perform data preprocessing using keras standard ImageDataGenerator (we couldn't apply data agumentation without a custom agumentator), so we added a very high dropout (0.8) in order to reduce as much as possibile the overfitting.

6.1 Mass/Calcification problem exploiting baseline patches



(a) *Training and validation acc* (b) *Training and validation loss*

Figure 41: *Twin VGG16 (`"VGG16_baseline_2.h5"`) accuracy and loss during training (Optimizer: Adam; batch size: 32; learning rate: 1e-4). We can notice clear overfitting after a few epochs of training.*

The accuracy on the test set is 0.82.

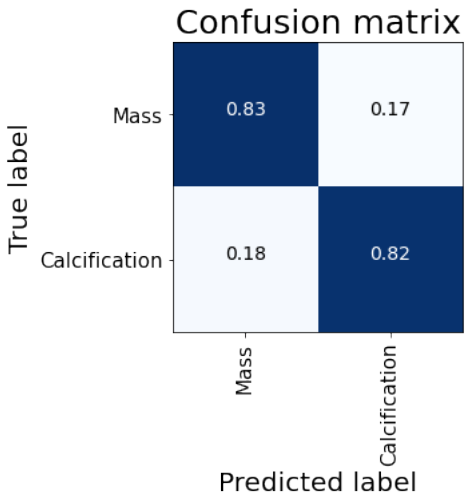
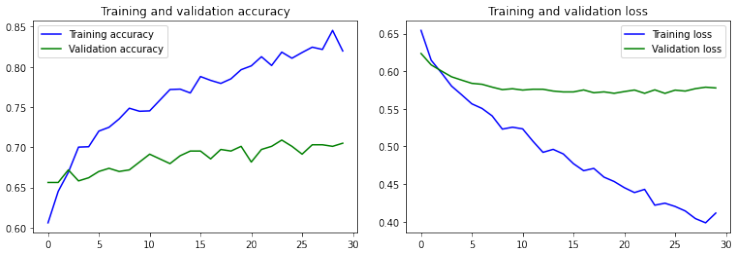


Figure 42: *Confusion matrix for Twin VGG16*

6.2 Benign/Malignant problem expoloiting baseline patches



(a) *Training and validation acc* (b) *Training and validation loss*

Figure 43: *Twin VGG16 (`"VGG16_baseline_2.h5"`) accuracy and loss during training (Optimizer: Adam; batch size: 32; learning rate: 1e-4). Notice the overfitting: validation accuarcy and loss change very little at each epochs, while training acc/loss keep increasing/decreasing.*

This network was able to obtain an accuracy of 0.63 on the test set, with the following confusion matrix.

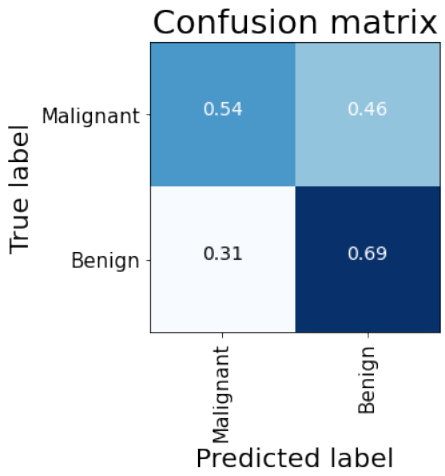


Figure 44: *Confusion matrix for Twin VGG16*

The results achieved for the both classification problems were definitely inferior than the ones obtained by other strategies. In practice, performing feature subtraction doesn't achieve exactly what we expected and in general drives to less precision in the classification. The main motivation behind this behaviour is the overfitting, as shown in the previous figures.

7 Task 5: Ensemble classifier

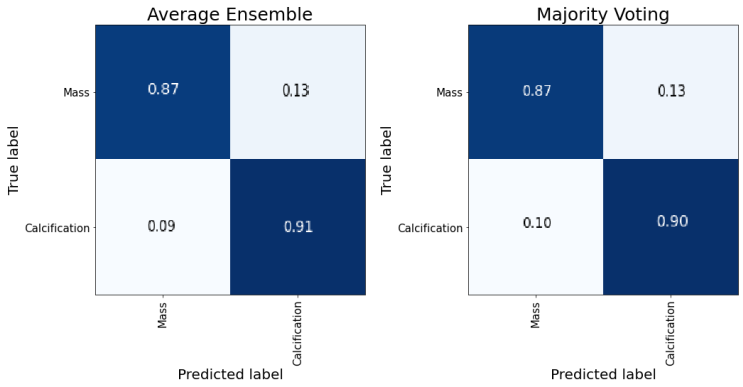
An ensamble classifier is a classifier that combines the predictions of various estimators in order to improve the perfomances and robustness over a single estimator. The main idea behind it is that different classifier will generally make different errors on the test set, so pooling together their votes may results in a more accurate final prediction thanks to the classifier errors "balancing" each other. We implemented an ensamble classifier for both classification problems, using the best networks that we developed in the previous tasks. The results of the different estimators were aggregated using both average and majority vote.
We also carried out a Weighted Average Ensamble, performing a grid search and assigning different weights to the various classifiers depending on their perfomance (on the test set). It is important to say that with this strategy we are training the weights on the test set, so the result is expected to be high in terms of accuracy. If we had more data at disposal, we could have used some of this data as training samples for the weight grid search and the rest as test set. For this reason, we are not considering this solution for the submission.
Code for this strategy is partially taken from an online tutorial [1]

7.1 Ensamble classifier for Masses and Calcifications

In the following list we report the networks used and their accuracy on the test set.

- SimpleCNN (train from scratch; accuracy 0.845)
- VGG16 (train from scratch; accuracy 0.880)
- SimpleAlexNet (train from scratch; accuracy 0.880)
- VGG16 (fine tuning; accuracy 0.889)
- VGG19 (fine tuning; accuracy 0.878)

We obtained an accuracy on the test set of 0.889 using average ensamble and of 0.886 using majority voting.



(a) *Confusion matrix for average ensamble* (b) *Confusion matrix for majority voting*

Figure 45: *Confusion matrixes for average ensamble and majority voting for the Mass/Calcification problem. We can notice a slightly better result with the average ensamble strategy.*

Weighted Average Ensamble (Mass/Calcification) After performing the grid search for estimate weights, we observed that assigning the same weight to the two pretrained models will lead to a better result on the test set (accuracy is 0.907).

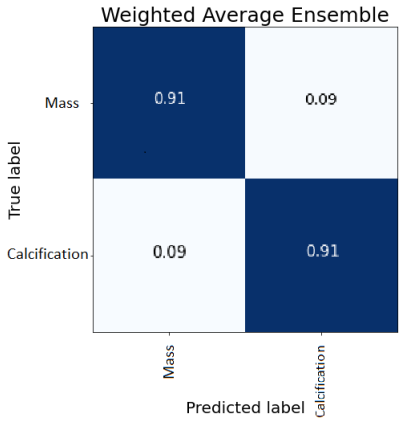


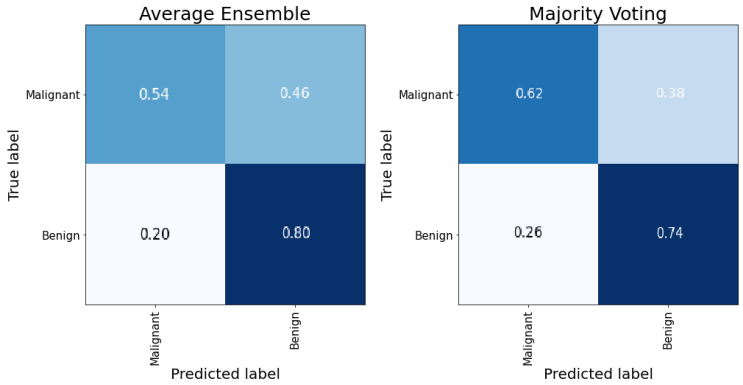
Figure 46: *Confusion matrix for Average Classifier using only the two pretrained networks.*

7.2 Ensamble classifier for Benign and Malignant

In the following list we report the networks used and their accuracy on the test set.

- SimpleCNN (train from scratch; accuracy 0.636)
- VGG16 (train from scratch; accuracy 0.627)
- SimpleAlexNet (train from scratch; accuracy 0.681)
- VGG16 (feature extraction; accuracy 0.711)
- VGG16 (3 stage fine tuned; accuracy 0.717)
- VGG19 (feature extraction; accuracy 0.6845)

We obtained an accuracy on the test set of 0.71 using average ensamble and of 0.70 using majority voting.



(a) *Confusion matrix for average ensamble* (b) *Confusion matrix for majority voting*

Figure 47: *Confusion matrixes for average ensamble and majority voting for the Benign/Malignant problem.*

Since the confusion matrix obtained with the majority voting strategy showed a better result in terms of recall for the malignant class, we can consider it as the best classification strategy between the two. Even if it has a sligthly lower accuracy, the ability to recognize the malignant samples is higher.

Weighted Average Ensamble (Benign/Malignant) We created a weighted ensamble using the three pre-trained networks. The result obtained in terms of accuracy on the test set is 0.735.

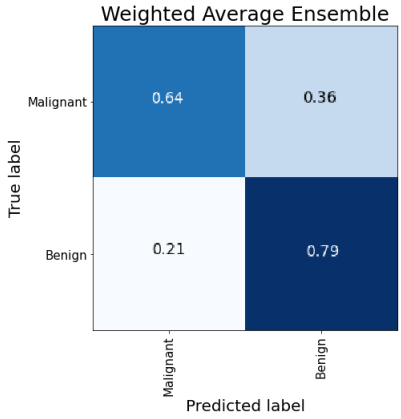


Figure 48: *Confusion matrix for Ensamble Weighted Classifier.*

8 Final Considerations

The two classification problems faced were very different: the Mass/Calcification problem was not particular difficult, since even a simple CNN was able to learn lots of important features from the training set and obtain an over 0.75 accuracy on the test set. For this reason, we mostly focused on setting hyperparameters capable of obtain the best results. The Benign/Malignant problem, on the other hand, was really complex since, in a sense, we had to fight both underfitting and overfitting. No CNN was able to learn very significant features from the images (we rarely obtained an accuracy on the training set higher than 0.75), resulting in poor perfomance on the validation/test sets.
Even if the papers analyzed were mostly focused on solving the Benign/Malignant problem for only masses or only calcifications, similar results were expected to be found in our experiments. Unfortunatly, the results obtained are inferior to the ones reported in the papers. In our opinion, the provided dataset is too small and doesn't represent well the problem, maybe because the images resolution is low(150x150) for our models to better discriminate between the two classes.

File Submission.csv For the context submission (Benign/Malignant) we compared the best classifier obtained in the previous experiments: VGG16 Feature Extraction (figure 38(a)), weighted classes VGG16 Fine Tuned with 3 stage (figure 40) and Ensamble with majority voting (figure 47(b)), considering accuracy, the confusion matrices and the ROC curves. Even if no ROC curve is completely dominating the others, we considered as best classifier the weighted classes VGG16 3 stage Fine Tuned newtork (`"VGG16_finetuned3step.h5"`), because it has the highest value for AUC (Area under the curve) and a good value of Recall for the malignant class (0.63).

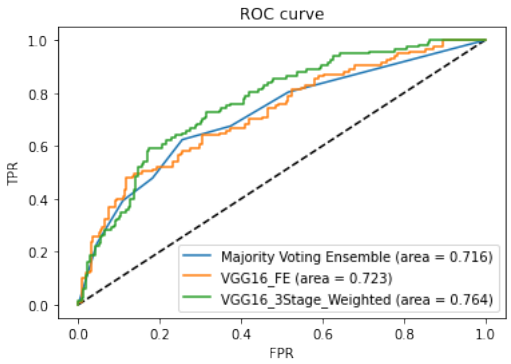


Figure 49: *ROC and AUC of our three best estimators.*

References

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

[2] Jia Deng et al. "Imagenet: A large-scale hierarchical image database". In: *2009 IEEE conference on computer vision and pattern recognition*. Ieee. 2009, pp. 248–255.

[3] LG Falconí et al. "Transfer learning and fine tuning in breast mammogram abnormalities classification on CBIS-DDSM database". In: *Advances in Science, Technology and Engineering Systems* 5.2 (2020), pp. 154–165.

[4] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. "Imagenet classification with deep convolutional neural networks". In: *Communications of the ACM* 60.6 (2017), pp. 84–90.

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

[6] Inês C Moreira et al. "Inbreast: toward a full-field digital mammographic database". In: *Academic radiology* 19.2 (2012), pp. 236–248.

[7] Cordelia Schmid, Stefano Soatto, and Carlo Tomasi. *Conference on Computer Vision and Pattern Recognition*. 2005.

[8] Li Shen et al. "Deep learning to improve breast cancer detection on screening mammography". In: *Scientific reports* 9.1 (2019), pp. 1–12.

[9] Rebecca L Siegel, Kimberly D Miller, and Ahmedin Jemal. "Cancer statistics, 2020". In: *CA: a cancer journal for clinicians* 70.1 (2020), pp. 7–30.

[10] Karen Simonyan and Andrew Zisserman. "Very deep convolutional networks for large-scale image recognition". In: *arXiv preprint arXiv:1409.1556* (2014).

[11] Christian Szegedy et al. "Going deeper with convolutions". In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2015, pp. 1–9.

[12] TensorFlow. *Classificazione su dati sbilanciati*. URL: https://www.tensorflow.org/tutorials/structured_data/imbalanced_data#optional_set_the_correct_initial_bias.

[13] Lazaros Tsochatzidis, Lena Costaridou, and Ioannis Pratikakis. "Deep learning for breast cancer diagnosis from mammograms—a comparative study". In: *Journal of Imaging* 5.3 (2019), p. 37.