Convolutional Neural Network for Medical Imaging Analysis - Abnormality

Christian Kevin Alvarado Rimas (585309)

Computation Intelligence, a.y. 2019/2020

1 Introduction

The following report is drafted after developing an assignment as part of the final project of Computational Intelligence course in the a.y. 2019/2020, it involves designing 5 classifiers and 1 composite classifier based on Convolutional Neural Networks (CNNs) from Curated Breast Imaging Subset of Digital Database for Screening Mammography (CBS:DDS) [10] for the following purposes:

– 1 Classifier for Mass/Calcification discrimination, 2 classes.
– 1 Classifier for Benign/Malignant Mass/Calcification discrimination, 4 classes.
– 1 Classifier using already-trained CNN model for Mass/Calcification discrimination, 2 classes.
– 1 Classifier using already-trained CNN model for Benign/Malignant Mass/Calcification discrimination, 4 classes.
– 1 Baseline-based classifier for Mass/Calcification discrimination, 2 classes.
– 1 Composite Classifier for Mass/Calcification discrimination, 2 classes.

In order to train the models, 2676 mammography screenings have been provided including their respective labels as follows:

– 0: Baseline.
– 1: Mass, benign.
– 2: Mass, malignant
– 3: Calcification, benign
– 4: Calcification, malignant

This document starts with state of the art of CNN on medical imaging application, the following subsections explain the reasoning behind the design of each classifier and its performance evaluation against its own variations. The last task, the composite classifier or ensemble system, was the most interesting in terms of creativity and reasoning demand.

2 State of the Art

A monumental performance of CNNs has been shown in the last decade for image classification and segmentation, such as ImageNet [8], VGG16 [9, 14] and InceptionV3 [11, 15]. Model after model has been designed in order to improve accuracy, avoid over-fitting and vanishing gradient effect due to their high dimensionality of the layers [15]. The field of application have been vast from videogames, such as Atari [6], up to medical diagnosis (or so called, Computer-Aided Diagnosis - CAD) [3].

In the field of breast detection anomaly, Arevalo [2] presents a two-stage method to detect mass lesion on mammography films; the first method lays on enhancing the details of the film by contrast correction and data augmentation, this output is fed onto the second stage, which is the training of a from-scratch CNN composed by 4 convolutional layers and 1 dense layer. This technique allows Arevalo to reach 0.822 of accuracy under the ROC curve for the Breast Cancer Digital Repository dataset.

On the other hand, Levy presented a transfer learning and a context-based approach [7]. Levy aims to discriminate benign masses from malignant ones. he trains a from-scratch model, which gives a validation accuracy of 0.66; however, by using transfer learning from a pre-trained AlexNet, such accuracy got boosted up to 0.9. He also studies, how much padding should be included, finding a proportional padding (he does not explain what "proportion" exactly means) gives better accuracy.

Chen [5] splits the mammography film into bit-spaces, being each bit-space a binary color channel (bit-plane) for 3-layers convolutional network with 1 flatten layer. With this approach Chen's model reaches a validation accuracy in the range 0.625-0.78. The mammography were splitting into different bit-spaces.

Finally, Hameed [1] reinforces Levy's statement: "pre-trained models perform higher accuracy on breast abnormality detection". Hameed fine-tunes VGG16 and VGG19 models, for later, merge them building this way an Ensemble CNN that reaches 95.29% validation accuracy in detecting carcinoma (a breast cancer type).

3 From-Scratch Models

In this section the design of two classifiers is described, 2-class classifier and 4-class classifier. All models were trained using 5-Fold cross-validation algorithm, Adam optimizer and after having acceptable results, the data have been augmented using the following parameters on the Keras API generator:

```
rotation_range=40,
width_shift_range=0.2,
height_shift_range=0.2,
zoom_range=0.2,
horizontal_flip=True
```

3.1 Mass/Calcification

Training and Design A First Model was implemented according to table 1. It is composed by 4 convolutional layers (*Conv*), each followed by a Max Pooling layers, 1 flatten layer to consolidate the feature map coming out from the CNN and the output, which is a single dense layer.

The kernel size in the first *Conv* layer is defined as 5x5, which is larger than the following layers (3x3) and it is because many information coming from the sample image are general and repetitive due to its neighbourhood, so a larger filter summarize such information. However, the first layer contains fewer (40) kernels, this is due to the fact the input layer does not contain abundant explicit and specific information; consequently, the deeper the networks is, the more kernels it has per layer and, the more specific features are extracted from the previous kernels. The number of kernels being doubled as deeper it goes, performs well as shown in AlexNet [8]. Also note, all *Conv* layers uses ReLU, because it was demonstrated by Glorot [4] this activation function enables fast computation due to its simplicity and better gradient propagation.

After each *Conv* layer a $max - pooling$ layer is added to extract features that are more relevant in the convolutional block of layers, as well as get rid of repetitive information in a small region (2x2). The $max - pooling$ layers are padded layers, which means, features at the edges are taken into the center of the kernel when max-pooling is performed, and by using *same* padding, the edges pass directly to the next layer, this means the features at the edges are kept between two consecutive *Conv*s blocks.

The final layer is a *Dense* layer of 1 neuron with *Sigmoid* activation function, because the case presented is a binary classification problem, the class is either a Mass or a Calcification. The *Sigmoid* will saturate the output limiting the values to one class or another, any value between 0 and 1.

Layer	Type	Filters	Filter Size	Padding	Activation
1	Convolutional	40	5x5	yes	ReLU
2	MaxPooling2D	-	2x2	yes	-
3	Convolutional	80	3x3	no	ReLU
4	MaxPooling2D	-	2x2	yes	-
5	Convolutional	160	3x3	no	ReLU
6	MaxPooling2D	-	2x2	yes	-
7	Convolutional	320	2x2	no	ReLU
8	MaxPooling2D	-	2x2	yes	-
9	Flatten	-	-	-	-
10	Dense		1	-	Sigmoid

Table 1. First Model, 2 classes

The learning curves of this model is plotted in figure 2, red-green pair, green being the training curve and red the validation curve. The model performs poorly in terms of learning speed and accuracy, in comparison to the Second Model, cyan-orange pair in figure 2. The second models implements the architecture of table 2.

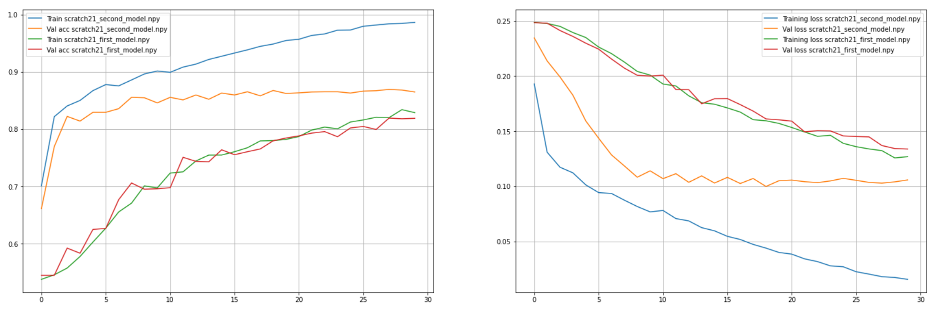

Fig. 1. First Model vs Second Model. Left: Accuracy. Right: Loss. cyan-orange pair: Second Model, green-red pair: First Model

Modifying some layers, the Second Model performed a better accuracy and a faster learning speed; although the learning rate was turned down to 0.00007 from 0.0001 in the First Model. This reduction was done because in the Second Model, the initial learning rate turned it into an unstable network giving high variability in accuracy between folds. On the other hand, a smaller value than 0.00007 turned the network slower; hence, 0.00007 is a trade-off design parameter.

Let us remark the relevant changes in the Second Model, the number of kernels being quadruplicated per *Conv* has shown better performance. The first *Conv* layer is not larger, but it *strides* 2 pixels when convolution is carried out over the input image, which is another way to summarize the initial features. A Batch Normalization is added after the first *Conv* in order to re-center and re-scale the input image, as proposed by Ioffe [13], which speeds up the learning processing, but abusing of it by placing it after each *Conv* aggravate the network's performance. Last remark, one more intermediate *dense* layer was added, this means the map feature coming out the CNN is not linearly separable for Mass and Calcification.

One more experiment was carried out using the Second Model, the ImageGenerator function from the Keras API was used to increase the number of training data, therefore the variety. The result is show in figure 18, where it is compared to the trained Second Model without the data augmentation. Parameters such as, flipping image horizontally and vertically, rotation, shifting and zooming were used; however the network's performance did not increase significantly and sometimes it even got worse. A zoom range of 0.2 slightly increased the accuracy of the network as seen in figure 18, this small gain in performance made the network 1 epochs slower, not relevant.

In the Computational Intelligence course, it was demonstrated that measuring a network's performance by simply taking a look into its accuracy is not enough, because the accuracy is taking into account the number False Positives

Layer	Type	Filters	Filter Size	Padding	Activation	Stride
1	Convolutional	40	3x3	yes	ReLU	2
2	Batch Normalization	-	-	-	-	-
3	MaxPooling2D	-	2x2	yes	-	1
4	Convolutional	160	3x3	no	ReLU	1
5	MaxPooling2D	-	2x2	no	-	1
6	Convolutional	640	3x3	no	ReLU	1
7	MaxPooling2D	-	2x2	yes	-	1
8	Convolutional	2560	2x2	no	ReLU	1
9	MaxPooling2D	-	2x2	yes	-	1
10	Flatten	-	-	-	-	-
11	Dense	-	75	-	ReLU	-
12	Dense	-	1	-	Sigmoid	-

Table 2. Second Model, 2 classes

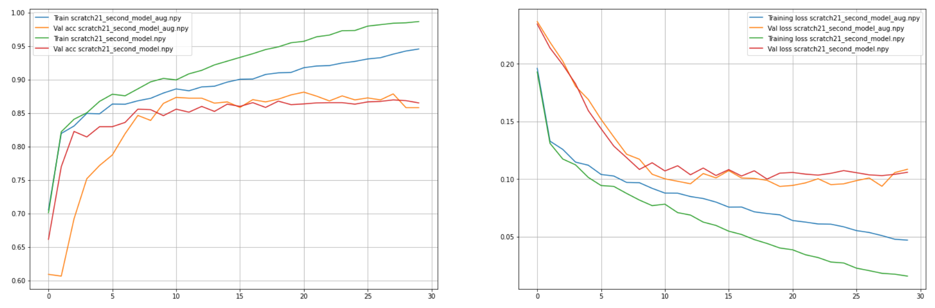

Fig. 2. Second Model vs Second Model with augmented data. Left: Accuracy. Right: Loss. cyan-orange pair: Second Model, green-red pair: Second Mode with augmented data

and True Positives over the entire dataset. To comprehend better a network is imperative to look at its Confusion Matrix and this will provide enough information either to choose the classifier with or without augmented data, in this case.

In table 4, three normalized confusion matrix are presented, say, a statistical double entry table. The control confusion matrix is the ideal output (or a perfect classifier confusion matrix), it is basically the data distribution of the training set. Given the fact the networks was trained by 5-fold algorithm, the output of the perfect classifier should be the same as the training labels. First, let us compare the Second Model with/without augmented data, the advantage of the augmented data lays on classifying calcification class, with 0.01 more accurate than its counter part. On the other hand, the augmented performs a thousandth worse at classifying Mass. By being numerically strict, the classifier with augmented data will be chosen for testing; Also,the usage of augmented data has shown better performance in [1, 2].

-	Actual M	Actual C	SM M	SM C	SM-AUG M	SM-AUG C
Actual M	0.455	0.0	0.398	0.057	0.397	0.058
Actual C	0.0	0.545	0.070	0.475	0.063	0.482

Table 3. Normalized Validation Confusion Matrices for from-scratch 2-class classifier. From left to right: Control confusion matrix, confusion matrix of the Second Model without augmented data, confusion matrix of the Second Model with augmented data. M: Mass class, C: Calcification class.

Test The SM-AUG is submitted under test, giving 86.3% of accuracy at discriminating between Masses and calcification. In table 4 the test confusion matrix is listed. Despite the validation confusion matrix, where SM-AUG did better classifying Calcification; on testing, the SM-AUG did better at classifying Mass, but this metric can be influenced by the ration mass-calcification of data, that could explain why mass is favoured in testing, and calcification favoured in training/validating.

-	Actual M	Actual C	SM-AUG M	SM-AUG C
Actual M	0.533	0.0	0.464	0.068
Actual C	0.0	0.467	0.068	0.399

Table 4. Normalized Test Confusion Matrices for from-scratch 2-class classifier. From left to right: Control confusion matrix, confusion matrix of the Second Model with augmented data. M: Mass class, C: Calcification class.

From table 4, the precision for discriminating mass is 0.872 and for calcification, 0.85. The high precision at discriminating mass over calcification might related with the testing data distribution that is also tilted towards the mass class, as it was previously commented.

3.2 Benign/Malignant Mass/Calcification

Training and Design Considering the performance of the Model from table 2, the first model for this 4-class classifier is similar, table 5, but changing the last 1-neuron *dense* for a 4-neuron *dense* layer with *softmax* activation function; due to the present case requires to classify the input image in 1 out of 4 classes mentioned above. By using *softmax* the energy of the barely predominant class gets boosted [12], and the remaining ones decrease. The usage of the softmax activation also implies to do one hot enconding on the labels.

Layer	Type	Filters	Filter Size	Padding	Activation	Stride
1	Convolutional	40	3x3	yes	ReLU	2
2	Batch Normalization	-	-	-	-	-
3	MaxPooling2D	-	2x2	yes	-	1
4	Convolutional	160	3x3	no	ReLU	1
5	MaxPooling2D	-	2x2	no	-	1
6	Convolutional	640	3x3	no	ReLU	1
7	MaxPooling2D	-	2x2	yes	-	1
8	Convolutional	2560	2x2	no	ReLU	1
9	MaxPooling2D	-	2x2	yes	-	1
10	Flatten	-	-	-	-	-
11	Dense	-	75	-	ReLU	-
12	Dense	-	4	-	Soft-Max	-

Table 5. Second Model Categorical, 4 classes

Figure 3 shows the performance of the Second Model Categorical, whose accuracy reaches 0.57 and there is even over-fitting. So, the idea of using such a model is abandoned. A variation of this model was created and named Third Model, which did not show any particular improvement, but it can be found in the Notebook.

The Forth Model in table 6 was built by putting away the building blocks of VGG19 (no transfer-learning). As it were a Jenga game, layers from the original stack were remove and changed until it reached the result of the figure 4.

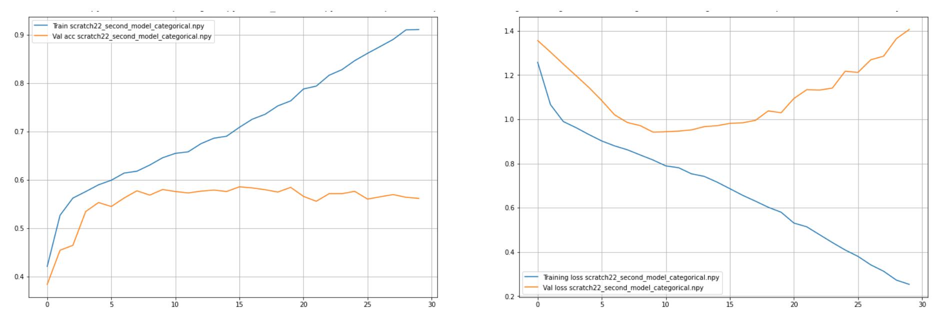

Fig. 3. Second Model Categorical. Left: Accuracy. Right: Loss

Layer	Type	Filters	Filter Size	Padding	Activation	Stride
1	Convolutional	64	3x3	yes	ReLU	1
2	Convolutional	64	3x3	yes	ReLU	1
3	MaxPooling2D	-	3x3	no	-	2
4	Convolutional	128	3x3	yes	ReLU	1
5	Convolutional	128	3x3	yes	ReLU	1
6	MaxPooling2D	-	3x3	no	-	2
7	Convolutional	512	3x3	yes	ReLU	1
8	MaxPooling2D	-	3x3	no	-	2
9	Convolutional	512	3x3	yes	ReLU	1
10	MaxPooling2D	-	3x3	no	-	2
11	Convolutional	512	3x3	yes	ReLU	1
12	MaxPooling2D	-	3x3	no	-	2
13	Flatten	-	-	-	-	-
14	Dense	-	4	no	Softmax	-

Table 6. Forth Model, 4 classes

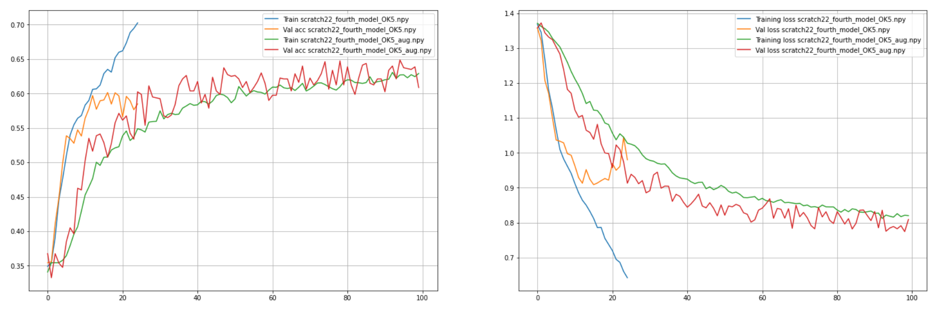

Fig. 4. Forth Model without augmentation data (cyan-orange) and Forth Model with augmented data (red-gree). Left: Accuracy. Right: Loss

The cyan-orange pair is the metrics from the non augmented data whose validation accuracy lays around at 60%. It is also observable that the network is slow, and increasing the learning rate lower the validation accuracy. The other disadvantage is the over-fitting that starts at epoch 15. At this point the network has been trained without data augmentation with a batch size = 32, and learning curve = 0.0001 with 20 epochs.

When augmentation data was introduced using the same hyper parameters configuration, the gap between the validation loss and training loss was vanished, and it showed a continuous loss decaying at epoch 20. But the network was still slow in both learning speed and in computation time (2 hours for 50 epochs) because it was training in small batches and generating images with 6 parameters. In order to know where the minimum loss was located, that batch size was increased to 128, this allowed the learning rate to change it to 0.0005 and the epochs = 100.

The networks is much slower, but it has better accuracy, no over-fitting and the minimum loss can be considered at epoch 80, because after that the change in the loss is not that significant. Another advantage is the computation time was reduced around 30-to-40 minutes.

Test The Chosen model to be tested was the forth model with augmentation data training, that showed higher accuracy and no over-fitting. The model was trained using early stop at 80, with batch size = 128, learning rate = 0.0005. The resulting of the test are seen in figure 5.

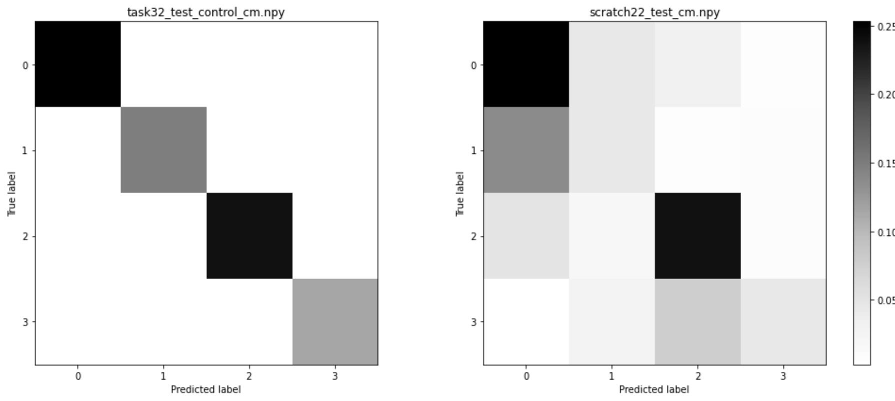

Fig. 5. Left: Test Confusion Matrix of perfect classifier, left: Test Confusion matrix of the Forth Model with augmented data

The testing gave an $accuracy = 0.58$, admissible because the validation accuracy is 0.62. From the confusion matrix, it's possible to compute the precision per class:

- 0.57 for Benign Masses
- 0.33 for Malignant Mass
- 0.68 for Benign Calcification
- 0.65 for Malignant Calcification

The Benign abnormalities are being discriminated with higher precision and it is worse at identifying Malignant Mass. This precision distribution can be related to the non-homogeneous distribution of the testing data as seen in figure 5 the benign samples are abundant in front of the malignant ones.

4 Pre-trained Models

In this section the design of classifiers built from pre-trained models are explained, these pre-trained models are used as Feature Extractor (FE) only or as Fine-Tuned (FT) and their respective performances are compared. Some modifications or extensions to these predefined models were added in order to increase the performance for the assigned classifier. the feature-extractor VGG16 model will be called $FE-VGG16$ for short, and the fine-tuned VGG16, $FT-VGG16$. Analogously, $FE-InceptionV3$ means feature extractor InceptionV3 and $FT-InceptionV3$, fine-tuned InceptionV3. All models were trained with 5-fold and cross-validation algorithm with augmented data using the following parameters on Keras API generator:

```
rotation_range=40,
width_shift_range=0.2,
height_shift_range=0.2,
zoom_range=0.2,
horizontal_flip=True,
vertical_flip=True,
```

All models are trained in two stages: first the pre-trained models are disabled for training, while only the output layer or an specify layer is trained. For the second stage, the pre-trained models are enabled to train their kernels to adjust to the new application.

Both training and testing set images have casted from monochromatic colours into a 3-channel images, because the intended pre-models have trained for RGB images. This casting was executed by triplicating its unique channel.

4.1 Mass/Calcification

Flatten Pre-Trained Model - VGG16 The only modification on this pre-trained model is to flat the output to perform binary classification, see table 7, then it was trained in two stages: first, the pre-trained model was kept fixed and then, the fine-tuning was applied to readjust the kernels on the VGG16.

Layer	Type	Dimension	Activation
1-19	VGG16	-	-
20	Flatten	-	-
21	Dense	1	Sigmoid

Table 7. Flatten VGG16, 2-classes classifier

Figure 6 shows the accuracy of the fine-tuned VGG16 (FT-VGG16) performs a higher validation accuracy ranging from 0.80 to 0.85 than FE-VGG16. Another advantage of the FT-VGG16, it is faster than FE-VGG16. However, observing the loss metrics, there exists clearly an over-fitting on FT-VGG16, an increasing gap over the training epochs, this can be solve by adding a dropout layer before the output layer.

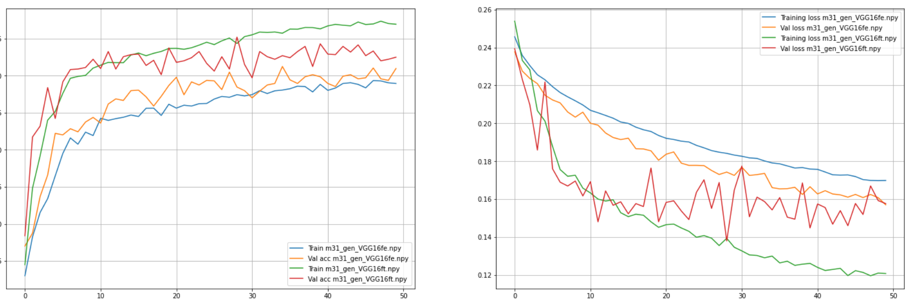

Fig. 6. Flatten VGG16: as Feature Extractor vs Fine Tuning. Left: Accuracy. Right: Loss

Densed Pre-Trained Model - VGG16 Before dropping away neurons, a dense layer with 2048 neurons was added between the pre-trained model and the output layer, see table 8. This modifications attempt to boost the accuracy of the model where VGG16 is a mere feature extractor, such assumption comes from the learning curves of the flatten FE-VGG16 where it does not show over-fitting, it follows a continuous growing but the low accuracy may suggest it is incapable of generalizing the mapping between input and output.

The results of this new approach is plotted in figure 7 where Flatten FE-VGG16 and Densed FE-VGG16 are compared in terms of accuracy and loss. In this plot, the densed version of the model over passes in accuracy (85%) as well as learning speed, however this results are still far from the Flatten FT-InceptionV3, 92.5% accuracy.

Layer	Type	Dimension	Activation
1-19	VGG16	-	-
20	Flatten	-	-
21	Dense	2048	ReLU
22	Dense	1	Sigmoid

Table 8. Densed VGG16, 2-classes classifier

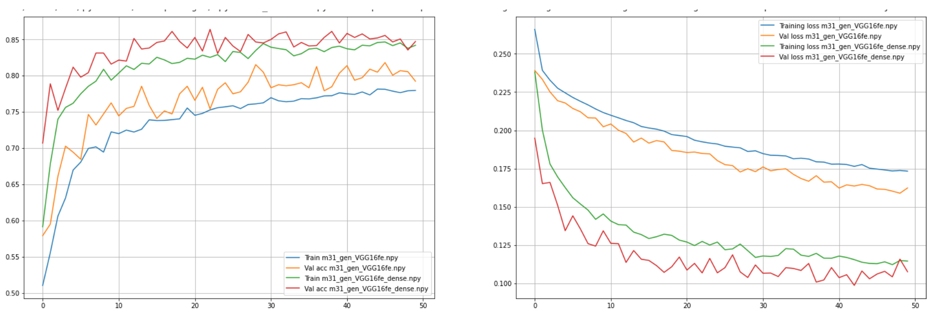

Fig. 7. Flatten FE-VGG16 vs Densed FE-VGG16, both as Feature Extractor. Left: Accuracy. Right: Loss

The following is to fine-tune the densed version in order to increase the performance. These results can be found in figure 8 where clearly the performance of the densed version did not increase but got aggravated, falling its validation accuracy from 85% to 58%, which makes it the worst model among all models presented for 2-class problem, including from-scratch model. This enormous decay tells more features needs to be extracted, which demands more trainable parameters in the dense side; however, the exploration over this hypothesis will not be followed in this assignment. Instead, by observing loss of the flatten FT-VGG16, as mentioned early, the gap at the 50th epoch training suggests over-fitting and by dropping out some features, the performance may be boosted up to the training curve. The results of such experiment are not included in this report but are available in the Colab Notebook submitted along this report, those results did not performed better than the InceptionV3.

Flatten Pre-Trained Model - InceptionV3 This model grabs the pre-trained InceptionV3 model and puts an output layer of a single neuron, see table 9. The model is trained in two stages as explained above.

Figure 9 shows the FT-InceptionV3 performs a higher validation accuracy (0.925) than the FE-InceptionV3, as well as, FT-InceptionV3 is faster. The disadvantage is similar as seen on FT-VGG16, the loss curve shows over-fitting, which might mean the FT-InceptionV3 has too many parameters, some have to be dropped away.

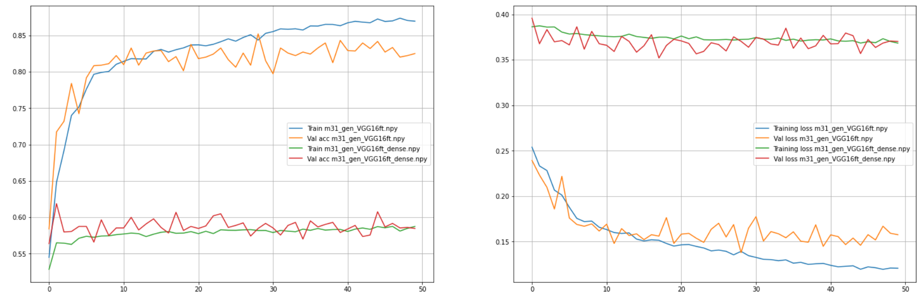

Fig. 8. Flatten FT-VGG16 vs Densed FT-VGG16, both Fine-Tuned. Left: Accuracy. Right: Loss

Layer	Type	Dimension	Activation
1-48	InceptionV3	-	-
49	Flatten	-	-
50	Dense	1	Sigmoid

Table 9. Flatten InceptionV3, 2-classes classifier

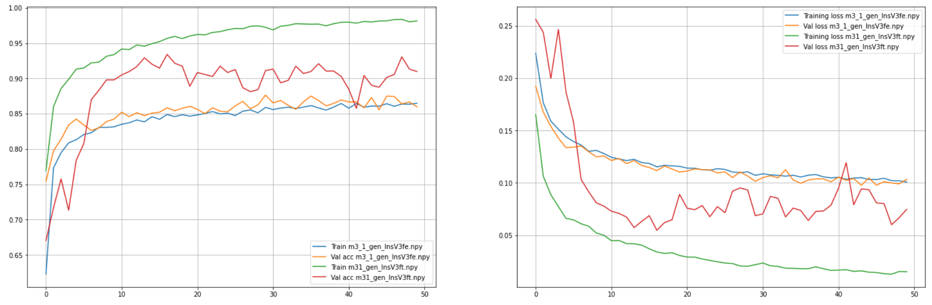

Fig. 9. Flatten InceptionV3: as Feature Extractor vs Fine Tuning. Left: Accuracy. Right: Loss

Dropout-ed Pre-Trained Model - InceptionV3 In this section, an exploration by dropping-out some trainable parameters away from InceptionV3 's last layer is executed. The architecture is found in table 10. As usual with pre-trained models in this report, the pre-trained models were trained in two stages.

Layer	Type	Dimension	Activation
1-19	InceptionV3	-	-
20	Flatten	-	-
21	Dropout	-	-
22	Dense	1	Sigmoid

Table 10. Dropout-ed InceptionV3, 2-classes classifier

Since the over-fitting issue has been pointed out in the Fine-Tuned InceptionV3 only (FE-InceptionV3 does not have over-fitting), the following reviews are carried out on those learning curves only, although the two-stage train was executed. In order to do the comparison, the flatten FT-InceptionV3 will be called as control model due to its best performance, so far.

The first experiment consisted in dropping out 50% of the output of the InceptionV3 model and as seen in figure 10, there is no significant improvement in the accuracy, red curve. The gap at the 50th epoch between the training loss and validation loss, curve green and red, are not significantly different from the non-dropout model, curve cyan and orange, where the over-fitting is still present.

Given the fact that the gap in the loss curves are kept, the second experiment was more aggressive, it drops 80% away from the flatten InceptionV3 model. The result of this experiment is also plotted on figure 10, pair purple-brown. In the validation accuracy curve, curve brown, it is seen a improvement from 92.5% to an 94%. On the other hand, from the loss curve, the gap in the 50th epoch is still large, but observable shorter than 50%-dropout model. 80% of dropping is a huge amount of features being ignored, and even though the over-fitting is persistent. This can be explained by the 21 million trainable parameters that InceptionV3 has distributed over 48 non-sequential layers.

To summarize, two pre-trained models were investigated for developing this transfer-learning 2-classes classifier, VGG16 and InceptionV3. Initially both models were used only as feature extractor and later fine-tuned to improve their performance. Based on their responses an additional dense layer was added to both pre-trained models, which aggravated their performances at fine-tuning stage. This due to the number of trainable parameters have increase up to triple the conventional model's size. So, instead of adding more layers to extract more features, the dropping-out approached was evaluated, only on InceptionV3 because it has an accuracy of 5% higher than the best VGG16 variation. The dropping-out technique allowed FT-InceptionV3 reached 94% of validation accuracy.

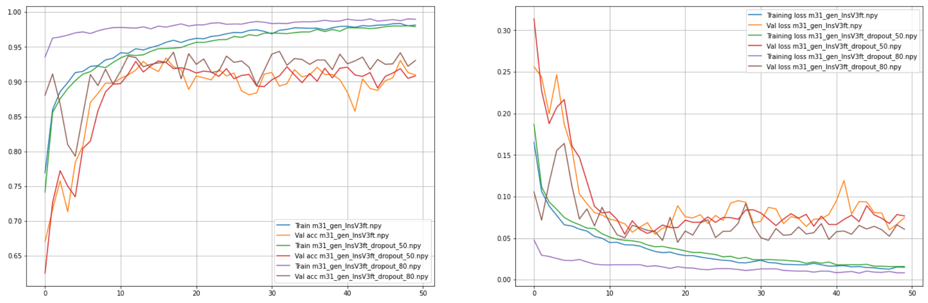

Fig. 10. Flatten FT-InceptionV3 vs Dropout@50 FT-InceptionV3 vs Dropout@80 FT-InceptionV3. Left: Accuracy. Right: Loss. Pair cyan-orange: Flatten FT-InceptionV3, pair green-red: Dropout 50% FT-InceptionV3, pair purple-brown: Dropout 80% FT-InceptionV3

Dropout80 FT-InceptionV3 Under Test This is the model which did the best at validation accuracy, now it is time to submit the model under a test with data that has not been presented to the model yet. First, the model has to be trained with the full training set in two stages as well.The first stage the number of epochs is 50, because the FE-InceptionV3 keeps a non-over-fitting characteristics in the loss curve until that epoch. The second stage is trained 19 epochs, according to the loss curve in figure 10 19 epochs is where the minimum loss has been reached, a technique called early stopping.

The test accuracy of the model is 0.845, in comparison with from-scratch model, the pre-trained model did it worse. From the confusion matrix in table 11 the precision for discriminating mass is 0.86 and the precision for calcification is 0.826. Even in precision metrics, the from-scratch model performed better for both classes.

-	Predicted Dropout80 FT-InsV3 M	Predicted Dropout80 FT-InsV3 C
Actual M	0.449	0.083
Actual C	0.071	0.396

Table 11. Confusion Matrix of Dropout 50% FT-InceptionV3

4.2 Benign/Malignant Mass/Calcification

In this section the design of the 4-class classifier based on pre-training models will be explained, two models were consider VGG16 and InceptionV3 and modified accordingly. All models have an 4-neurons output layer with $softmax$ activation function because of the 4 classes they are asked to classify, see tables 12, 13, 14 and 16. All models were trained using 5-Fold cross-validation algorithm and augmented data of 64 batch with the following parameters on the Keras API generator:

```
rotation_range=40,
width_shift_range=0.2,
height_shift_range=0.2,
zoom_range=0.2,
horizontal_flip=True
```

Unlike the 2-classes classifier where the augmented data did not help, for the 4-classes classifer augmented data certainly helped, that's why all the parameters listed above have been considered in this design.

A performance comparison was held by plotting the validation accuracy and loss of the models under study. By analysing this curves, the models can be modified to increase performance. The loss curve beyond describing the goodness of fit, also tells the learning stops in the epoch where the minimum loss is located, early stopping technique [12], already used in the design of the previous model.

Flatten Pre-Trained Model - VGG16 The first model under study is the VGG16 with the only modification of flatting the output to perform categorical classification into 4 classes, see table 12. As usual, a two stage training will be carried out for this model.

Layer	Type	Dimension	Activation
1-19	VGG16	-	-
20	Flatten	-	-
21	Dense	4	Softmax

Table 12. Flatten VGG16, 4-classes classifier

Figure 11 shows the performance of both the FE-VGG16 and FT-VGG16. As it was already observed in the 2-classes classifier, a fine-tuned model performs way better than its feature-extractor counterpart. This is seen in term of accuracy as well as learning speed. However, once again the better performance creates over-fitting, this is observed by the large gap between the loss curve of training and validation in figure 11. As mentioned earlier, in order to avoid overfitting, some features can be dropped by adding a dropout layer between the VGG16 model and the output layer.

Dropout-ed Pre-Trained Model - VGG16 the architecture for the following experiment is as seen in table 13. Just recall the training was a two stage training.

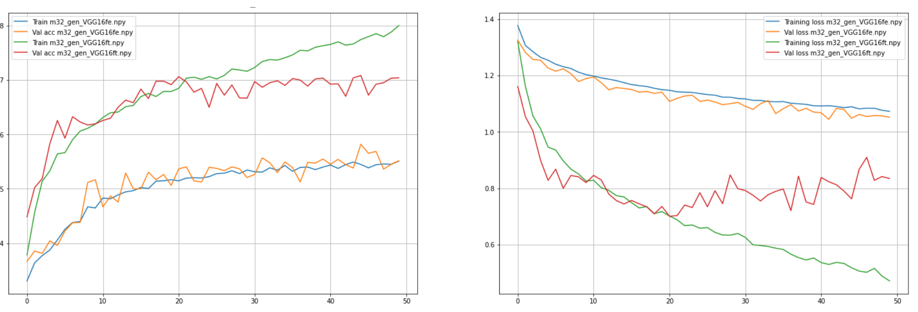

Fig. 11. Flatten FT-VGG16 vs Flatten FE-VGG16. Left: Accuracy. Right: Loss. Pair cyan-orange: Flatten FE-VGG16, pair green-red: Flatten FT-VGG16

Layer	Type	Dimension	Activation
1-19	VGG16	-	-
20	Flatten	-	-
21	Dropout	-	-
22	Dense	4	Softmax

Table 13. Dropout-ed VGG16, 4-classes classifier

Figure 12 shows the comparison between the Flatten FT-VGG16 and the Dropout FT-VGG16 with 50% dropping rate. This new approach loses accuracy and learning speed but the over-fitting decreased, this means 50% rate of dropping-out is too much. In figure 13 the result of another experiment is shown, the dropping rate has been changed down to 25%. However, neither significant accuracy or less over-fitting have been reached. Concluding, the best model, so far, for a 4-classes classifier is the flatten FT-VGG16 with 70% accuracy and with minimum loss at epoch 20.

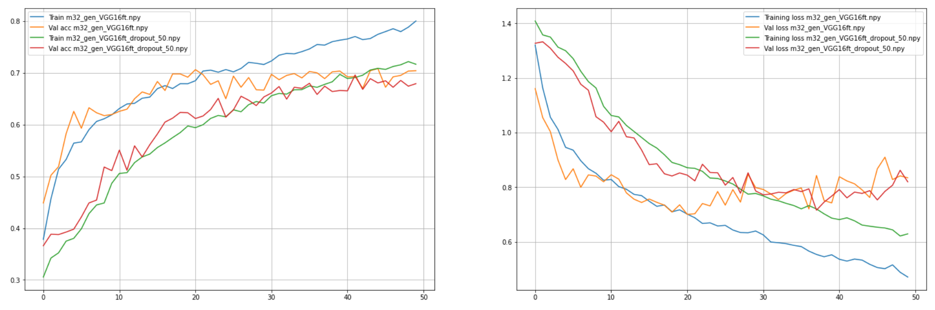

Fig. 12. Flatten FT-VGG16 vs Dropout@50% FT-VGG16. Left: Accuracy. Right: Loss. Pair cyan-orange: Flatten FE-VGG16, pair green-red: Dropout@50% FT-VGG16

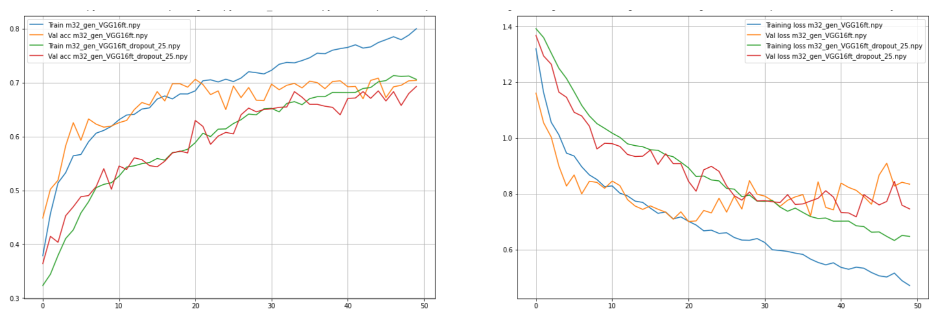

Fig. 13. Flatten FT-VGG16 vs Dropout@25% FT-VGG16. Left: Accuracy. Right: Loss. Pair cyan-orange: Flatten FE-VGG16, pair green-red: Dropout@25% FT-VGG16

Flatten Pre-Trained Model - InceptionV3 The next model to investigate is the InceptionV3; first, the one with flatting output is analysed, see table 14.

Layer	Type	Dimension	Activation
1-48	InceptionV3	-	-
49	Flatten	-	-
50	Dense	4	Softmax

Table 14. Flatten InceptionV3, 4-classes classifier

In figure 14, the learning curves of the FE-InceptinoV3 and FT-InceptionV3 are plotted, where it's observable, as usual, the fine-tuned model performs better in accuracy and learning speed with the disadvantage of creating over-fitting. On the other hand, in this case, the feature extractor model did not do way worse and it is not creating over-fitting; that is why, the next experiment explored by adding a dense layer.

Densed Pre-Trained Model - InceptionV3 For this experiment the architecture in table 15 was implemented and the results are shown in figure 15. From the similarities of the leaning curves none of them is significantly better; however, the Densed FT-InceptionV3 is composed of 59.5 million parameters and

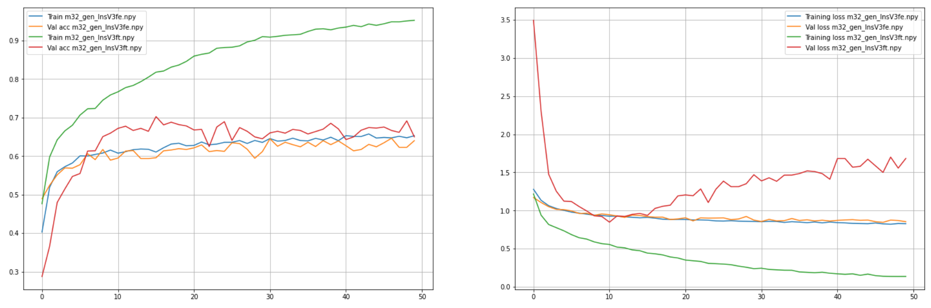

Fig. 14. Flatten FT-InceptionV3 vs Flatten FE-InceptionV3. Left: Accuracy. Right: Loss. Pair cyan-orange: Flatten FE-InceptionV3, pair green-red: Flatten FT-InceptionV3

the Flatten one, of 21.8 millions. the larger number of parameters on densed requires more computer capabilities, capabilities that are not well compensated in front of its flatten counterpart that with a third of the parameters does slightly better; therefore, FT-InceptionV3 alongside FT-VGG16 reach 70% of accuracy, becoming both the best model for this 4-class classification.

Layer	Type	Dimension	Activation
1-48	InceptionV3	-	-
49	Flatten	-	-
50	Dense	2048	-
51	Dense	4	Softmax

Table 15. Dropout-ed InceptionV3, 4-classes classifier

Dropout-ed Pre-Trained Model - InceptionV3 Although the flatten FT-InceptionV3 is one of the best models is not exempt of over-fitting; the reason why, this experiment look for improving its performance by placing a dropout layer between the pre-trained InceptionV3 model and the output layer, see table 16.

Its result is confronted against the flatten FT-InceptionV3 in figure 16. In here, the dropping-out rate was fixed in 50% and the resulting loss curves (green-red pair) shows a slight improvement in term of loss and accuracy; However in terms of over-fitting, it seems present yet. In order to enhance its performance one more experiment was carried out, where the dropping-out rate goes up to 80%, the learning curve of such aggressive dropping is portrayed in figure 17.

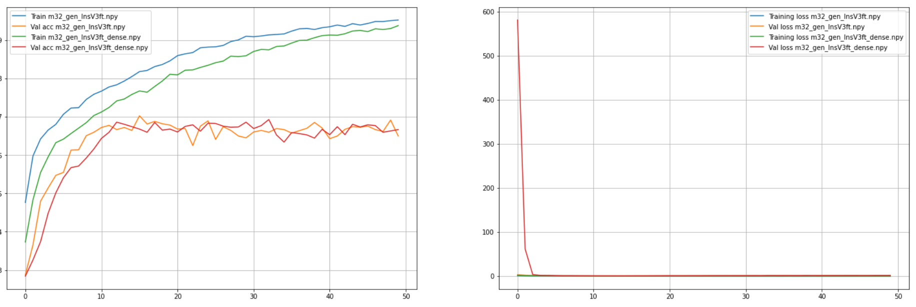

Fig. 15. Flatten FT-InceptionV3 vs Dense FT-InceptionV3. Left: Accuracy. Right: Loss. Pair cyan-orange: Flatten FT-InceptionV3, pair green-red: Dense FT-InceptionV3

Layer	Type	Dimension	Activation
1-48	InceptionV3	-	-
49	Flatten	-	-
50	Dropout	-	-
51	Dense	4	Softmax

Table 16. Dropout-ed InceptionV3, 4-classes classifier

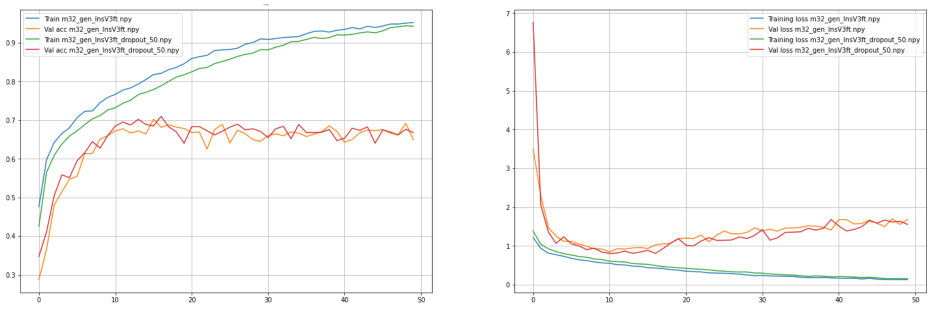

Fig. 16. Flatten FT-InceptionV3 vs Dropout@50% FT-InceptionV3. Left: Accuracy. Right: Loss. Pair cyan-orange: Flatten FT-InceptionV3, pair green-red: Dropout@50% FT-InceptionV3

No remarkable improvement is seen in terms of accuracy, other than a slightly bigger peak of the high dropping rate model at epoch 21. Therefore, the best InceptionV3 variation is the fine-tuned model with 50% dropping rate that reaches a validation accuracy of 70%, a value similar to the flatten FT-VGG16's. In order to choose or reject one of these models, an analysis of their confusion matrix was carried out and the results are shown in figure 18. Clearly the InceptionV3 with 50% dropping-out rate is not good for classifying any other class the benign Calcification.

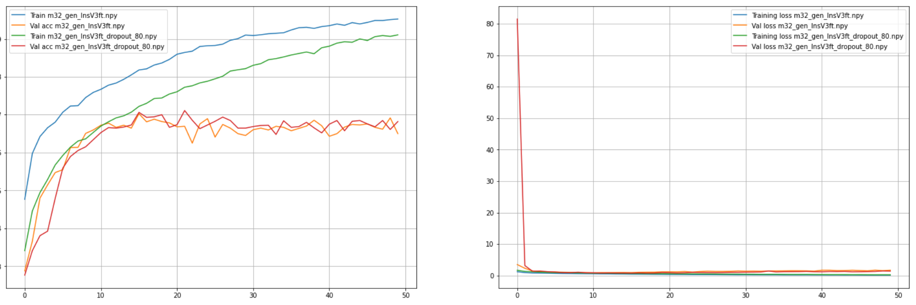

Fig. 17. Flatten FT-InceptionV3 vs Dropout@80% FT-InceptionV3. Left: Accuracy. Right: Loss. Pair cyan-orange: Flatten FT-InceptionV3, pair green-red: Dropout@25% FT-InceptionV3

Models Under Test From the figure 18, the model to be submitted for testing is the flatten FT-VGG16. The model was trained in two stages: first, in the feature-extractor-only the epoch was 50 because there is no clue about overfitting until that epoch, the loss keeps going down, but no further digging in this field has been done.Second, the fine-tuning stage, the model is trained with 20 epochs following the recommendations of early stopping technique.

In figure 19 the test confusion metrics is displayed, from here, the test accuracy is 0.54, with the following precision per class:

- 0.558 for Benign Masses
- 0.46 for Malignant Mass
- 0.576 for Benign Calcification
- 0.42 for Malignant Calcification

This model is better at discriminating benign abnormalities; even though, in terms of precision the numbers for benign abnormalities are slightly better

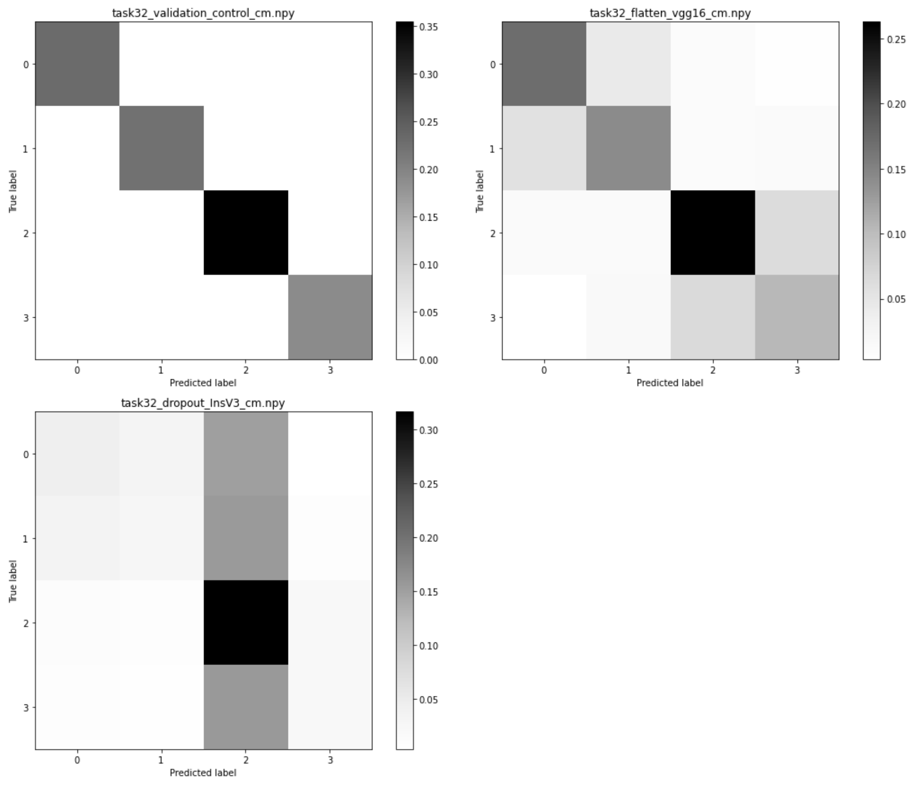

Fig. 18. Validation Confusion Matrix. Upper left: perfect classifier, upper right: flatten FT-VGG16, down left: dropout50 FT-InceptionV3

than random guessing. On the other hand, comparing it with the from-scratch model, the pre-trained network has worse accuracy and it is favoured to benign abnormalities.

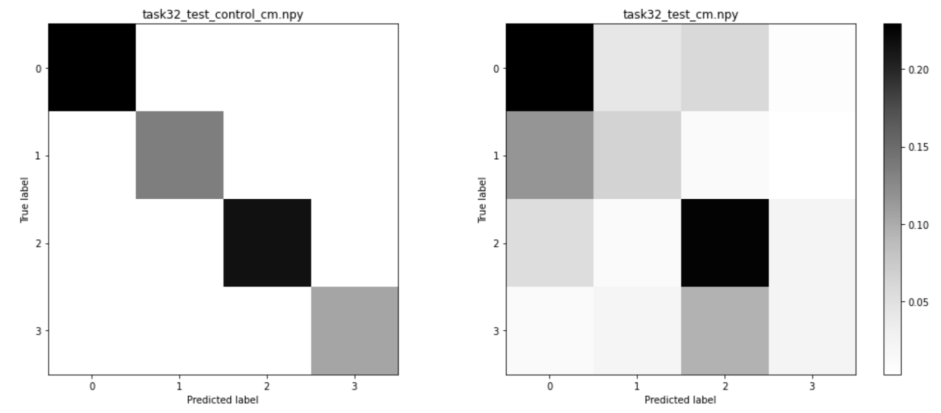

Fig. 19. Flatten 2FE-InceptionV3 vs Flatten 2FT-InceptionV3. Left: Accuracy. Right: Loss. Pair cyan-orange: Flatten 2FT-InceptionV3, pair green-red: Flatten 2FE-InceptionV3

5 Baseline Model

This classifier was designed to discriminate between two classes $Mass$ and $Calcification$, since the requirement is to take into account the baseline of the abnormality, a siamese alike model approach was adopted, see table 18. Later, this first proposed model was modified based on its performance in order to improve it.

Layer	Type	Dimension	Activation
1-48	[InceptionV3$_1$, InceptionV3$_2$]	-	-
49	Concatenate	-	-
50	Flatten	-	-
51	Dense	1	Sigmoid

Table 17. Dropout-ed InceptionV3, 4-classes classifier

The model has been trained in two stages: in the first training, the model works as a siamese network because both InceptionV3 models have the same parameters, and only the parameters on the dense layer are trained, meaning the

InceptionV3 models are feature extractors (named Flatten 2FE-InceptionV3). In the second training stage, a fine-tune training was performed to re-adjust the parameters of the two InceptionV3 models, independently, and boost the performance (named Flatten 2FT-InceptionV3). In the end, the whole model is not a siamese network because each CNN have adjusted their parameters based on their respective images. The results of this experiment are shown in figure 20. The learning curves evidence fine-tune double branch networks work better than a siamese one. In terms of learning speed the fine-tuned models reach a minimum loss in 9 epochs, meanwhile, the siamese does not reach the minimum even after 50 epochs; although this does not suggest the accuracy grows at high rates creating a gap of 5% accuracy below the fine-tuned model.

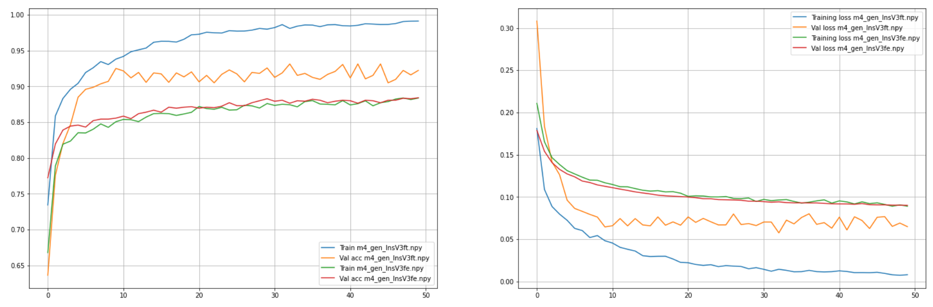

Fig. 20. Flatten 2FE-InceptionV3 vs Flatten 2FT-InceptionV3. Left: Accuracy. Right: Loss. Pair cyan-orange: Flatten 2FT-InceptionV3, pair green-red: Flatten 2FE-InceptionV3

The disadvantage of the fine-tuned model is a gap in the loss curve which suggest over-fitting at longer epochs, if this can be got rid of, the performance of the validation accuracy can go even higher; that is why, a second experiment was performed by adding a dropout layer with 50% rate and then 25% by implementing the architecture in table 18. The comparison is shown in figure 21. The 50% rate aggravates the validation accuracy and does not reduce the over-fitting. On the other hand, a 25% rate shows improvement on the validation accuracy, 93%. The gap on the loss curve at the epoch 8 also is being reduced to the minimum between the three models, however, at long term the gap is still present.

5.1 Dropout@25 2FT-InceptionV3 under Test

As the previous pre-trained networks, this network was trained by two stages: the feature-extractor-only was trained in 50 epochs, because up to 50 there is

Layer	Type	Dimension	Activation
1-48	[InceptionV3$_1$, InceptionV3$_2$]	-	-
49	Concatenate	-	-
50	Dropout	-	-
51	Dense	1	Sigmoid

Table 18. Dropout-ed InceptionV3, 4-classes classifier

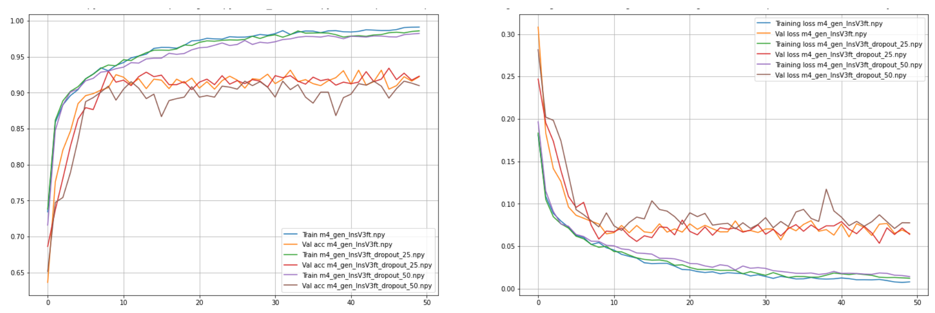

Fig. 21. Flatten 2FT-InceptionV3 vs Dropout@25 2FT-InceptionV3 vs Dropout@50 2FT-InceptionV3. Left: Accuracy. Right: Loss. Pair cyan-orange: Flatten 2FT-InceptionV3, pair green-red: Dropout@25 2FT-InceptionV3, pair purple-brown: Dropout@50 2FT-InceptionV3

no clue of over-fitting. The second stage is trained in 14 epochs, early stopping. The model was trained with augmented data as well, but unlike the previous 2-class classifiers, during training it was seen that input variety improved the networks performance, as consequence the following parameters were set on the ImageGenerator of the Keras API:

```
rotation_range=40,
width_shift_range=0.2,
height_shift_range=0.2,
zoom_range=0.2,
horizontal_flip=True,
```

Table 19 is the confusion matrix of the tested dropout25 2FT-InceptionV3 model. From that matrix, the precision per discriminating Mass is 0.8499 and for calcification is 0.83. The test accuracy reached was 0.842. In comparison with the from-scratch model, these metrics are not the best, and once again we can see the model is more precise at discriminating Mass during testing.

-	Predicted drop25 2FT-InsV3 M	Predicted drop25 2FT-InsV3 C
Actual M	0.455	0.077
Actual C	0.081	0.387

Table 19. Confusion Matrix of Weighted Average WAVG approach

6 Ensemble Model

In this section 5 methods for fusion the data coming out from the built models on previous tasks are proposed, executed and compared. For such a purpose, 3 different 2-class classifier are called: the from-scratch model, the pre-trained-based model and the baseline model. All this models were developed in previous sections and their testing raw outputs were saved in their respective files.

6.1 Average - AVG

The first approach is defined by the equation 1, being y_i the output of the from-scratch model, pre-trained bassed and the baseline: 1, 2, 3, respectively. This approach has reached an accuracy 87.798%.

$$y_{ensemble} = \frac{y_1 + y_2 + y_3}{3} \tag{1}$$

The normalized confusion Matrix of the model compared to a perfect classifier is shown in table 20. the two first columns are the output of perfect classifier, and the last two are the outputs of the $Average - AVG$ approach. For a better conclusion the precision for each class is computed: $Precision_{Mass} = 0.892$, $Precision_{Calcification} = 0.8625$. The precision quantity is independent of the data distribution, because as seen 20, the testing data is not homogeneous distributed among the two classes, therefore, it is not a fair comparison just looking the numbers inside the boxes. So, this Average AVG approach is more precise when the mammography contains a mass.

-	Actual M	Actual C	Predicted AVG M	Predicted AVG C
Actual M	0.533	0.0	0.467	0.065
Actual C	0.0	0.467	0.057	0.411

Table 20. Confusion Matrix of perfect classifier (left) and Average AVG approach (right)

6.2 Weighted Average - Accuracy Based - WAVG

In the last approach, the output coming from the three models are taking as equals even though the models are not perfect, the output comes with an accuracy of prediction. That's why this approach assigns a weight to each output, see equation 2. The wights are the accuracy of each model, describing a level of confidence of the model's output and punish that result with their accuracy.

$$\begin{aligned} y_{ensemble} &= \frac{w_1y_1 + w_2y_2 + w_3y_3}{w_1 + w_2 + w_3} \\ w_1 &= 0.8796 \\ w_2 &= 0.9062 \\ w_3 &= 0.9110 \end{aligned} \tag{2}$$

The confusion matrix of this approach in table 21 tells that punishing assumption did not work as it was thought. The accuracy reached with this approach is 0.875, slightly smaller than the AVG one. In terms of precision per class: $Precision_{Mass} = 0.887$, $Precision_{Calcification} = 0.8616$; both precision in this approach confirms that punishment by accuracy is not that path to follow.

-	Predicted WAVG M	Predicted WAVG C
Actual M	0.467	0.065
Actual C	0.059	0.407

Table 21. Confusion Matrix of Weighted Average WAVG approach

6.3 Logic Voting - LV

So far, the raw data coming from the models have been taking into account, but these predictions are classes, a certain mammography is either in one class or another. That's why the following approach is inspired in logic voting used in redundancy engineering, Triple modular redundancy. Translating it for model fusion terms, if two or more models predict a certain class then that is the real class.

This Logic Voting LV approach is a two-steps method: First, the output of each model is threshold-ed, if the value is bigger than 0.5 is set to 1, 0 otherwise. In here, the majority voting function is applied, equation 3.

$$y_{ensemble} = 1 - (1 - y_1y_2)(1 - y_1y_3)(1 - y2_y3) \tag{3}$$

This equation does not need normalisation because the maximum value can come from that equation is 1 and the minimum is 0.

From its confusion matrix, table 22, the accuracy is 0.869, which turns this approaches into the worst presented so far. The precision are as follows: 0.881 for mass and 0.855 for calcification. The low performance might lay on the fact that the outputs are threshold-ed without any punishment. So, if two models are wrong then the whole prediction will be wrong. A branch approach from this could be punishing the output values with their specific accuracy per class or precision per model; however, this is an unfair treat or 0-favoured class, since all model's accuracy and precision are smaller than 1, by multiplying these to the outputs, the outputs become even smaller tilting the classification to the class with a smaller number representation, which is 0.

-	Predicted LV M	Predicted LV C
Actual M	0.464	0.068
Actual C	0.062	0.404

Table 22. Confusion Matrix of Logic Voting LV approach

6.4 Weighted Average - Precision Based - PWAVG

Going back to the row values of the models. This approach goes further into the punishment, it is a weighted sum of the output, but output is multiply by the preciion of the class they claim to belong to, see equation 4 for mathematical understanding. In this way, the weights vary from output to output coming out from the same model, and the normalisation term varies as well.

$$\begin{aligned} y_{ensemble} &= \frac{p_1 y_1 + p_2 y_2 + p_3 y_3}{p_1 + p_2 + p_3} \\ p_i &= \begin{cases} P_{c0}^{net_i} & C_0 predicted \\ P_{c1}^{net_i} & C_1 predicted \end{cases} \end{aligned} \tag{4}$$

The results from table 23 tells, this approach does not work as expected, the accuracy is 0.863, the precision for mass is 0.872 and 0.854 for calcification, one of the worst approaches alongside logic voting.

-	Predicted PWAVG M	Predicted PWAVG C
Actual M	0.464	0.068
Actual C	0.068	0.039

Table 23. Confusion Matrix of Weighted Average Precision Based PWAVG approach

Four different approaches were presented and the one with the simplest methods has performed better than the over-thought ones. The Average AVG approach has the best accuracy and precision for predicting both classes; however, the mixing of the outputs by averaging did not average the test accuracy of the models: 0.86, 0.845, 0.842 (from-scratch, pre-trained, baseline, respectively), it boosted to 0.877. In fact, none of the presented approaches decreased the overall performance or averaged it from the individual models.

6.5 Why does Averaging perform better?

Let us take a deeper look in the meaning of a network's output, graphically, from the statistical point of view. As far as it is understood, the binary classifier with sigmoid function gives a probabilistic value, which is related with class membership; this definition turns the sigmoid function into a Cumulative Distribution Function (CDF), therefore, its derivative is the Probability Density Function (PDF), see (5)

$$\begin{aligned} PDF(z) &= \frac{e^{-z}}{(1+e^{-z})^2} \\ CDF(z) &= \int_{-\infty}^{+\infty} PDF(z)dz = \frac{1}{1+e^{-z}} \\ \text{z} &= \text{Representative Value of the Input} \end{aligned} \tag{5}$$

Let O_1, O_2, O_3 be the output of three binary classifiers that use sigmoid function. Given the fact that Sigmoid parameters is not a trainable parameter, all three classifier draw the same function, but z_i is unique on each network because it is its own way to describe the input, therefore, its probability (area under PDF) O_i will change.

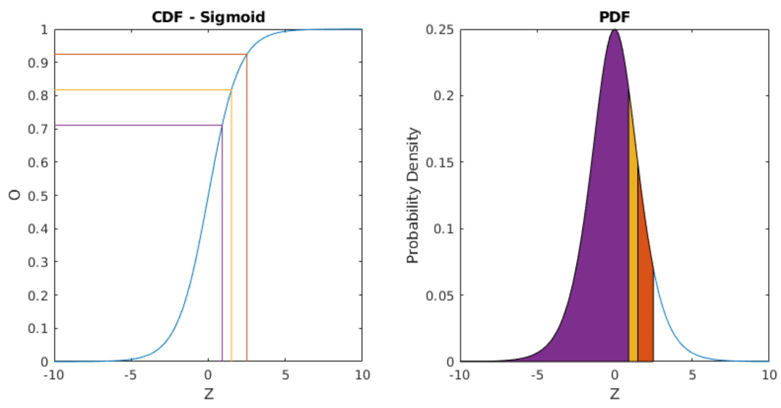

Fig. 22. Left: Cumulative Density Function. Right: Probability Density Function. Red: O_1, orange O_2, purple: O_3. Values are referential for explaining the concept

In figure 22, three networks' outputs O_i were assumed (they do not reflect real data). As it is seen, the output O_i can be mapped back into the z_i values and remapped into the PDF space. In this way, it is understood the output O_i

represents an area under $PDF(z_i)$. The more area of an certain input covers the bell curve, the more the probability is to belong to the class 1. It is an obvious assumption, but it makes easy to reason why arithmetic mean works better than punishment by accuracy, the first simply average areas. On the other hand, by multiplying these areas times their respective accuracy (a value less than 1) this areas are shrunken, so does the probability for class 1. See table 21 and 22, confusion matrix of AVG and WAVG method, the miss-classification of actual class 1 (predicted as class 0) is 0.002 bigger than than AVG method; this slight increment obliged the classification accuracy for class 1 to go down from 0.411 in AVG to 0.407 in WAVG; however, neither the accuracy or miss-classification of class 0 is affected. The WAVG forces the input to be classified as class 0; therefore, class 1 is unfairly treated.

Another technique to avoid is to choose the maximum O_i, this will force the classification for class 1, because the bigger O_i the higher the probability to belong to class 1 than class 0. On the contrary, taking the minimum area will tilt the classification in favour to class 0 (as WAVG did). So, let's take the middle value instead (median): probabilities near 0.5 might be adopted, those do not provide information, they are vague value; therefore, there will be more miss-classification on both classes. To clarify this reasoning, let be $O_1 < O_2 < 0.5 << O_3$ and O_1, O_2 are very close to 0.5. Taking the middle value will place the input in the class 0 without taking the opinion of O_3, even though the median probability of being in class 0 is uncertain (analogously for class 0). This also explains why logic voting performed not as good as expected, as in the previous assumption, the fact that both values are just under 0.5 means they are classifying the input in class 0 and since they represent the majority, the final output is class 0 as well, without double checking the classification with 0_3. The fact that majority votes for a certain class with tangling opinions does not mean they are right (Trump-2016 is a good example of this case).

Finally, the validation and precision cannot be taken into account because those values are biased by the unbalance training data set. the validation precision of class 0 is bigger than class 1 because there are more data of class 0, even if a random classification is performed, it will do good at classifying class 0.

6.6 Opinion Rating - OP

The learnt lecture: the ensemble system has to treat both classes fairly and the involved models' outputs have to be judged (called *model's opinion* O_i). If their *opinions* are close to 0.5, then they are meaningless; instead, if their opinions are close to 0 or 1, it means there is a solid support by the network for the chosen class.

Many functions can model this reasoning: *sinh*, *tan* and any odd-degree polynomial, the following opinion rating functions $R(O_i)$ are proposed:

$$
\begin{aligned}
R(O_i) &= \frac{1}{\tan(\frac{\pi}{2.2})}\tan((O_i - 0.5)\frac{\pi}{1.1}) \\
R(O_i) &= \frac{1}{\sinh \pi}\sinh\left(2(O_i - 0.5)\pi\right) \\
R(O_i) &= \frac{1}{0.5^n}(O_i - 0.5)^n | mod(n,2) = 1
\end{aligned} \tag{6}
$$

From their graphs in figure 23, these functions are intended to be limited between -1 to 1 for class 0 and class 1, respectively; hence, if an opinion O_i is minimised, it is minimised to the vague (0.5) point and not to the opposite class, treating this way both classes fairly.

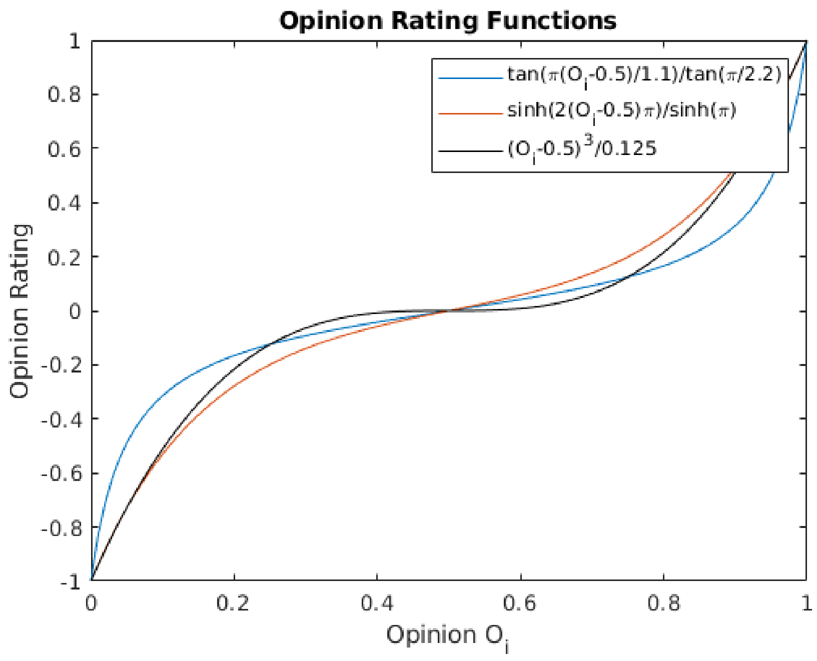

Fig. 23. Opinion Rating Functions. Black: n = 3

The second step is to average those opinion rating values $R(O_i)$, re-centring around 0.5 and re-scaling from 0-1; the last two operations are not compulsory but they help to understand the final output from a statistical point of view:

$$y_{merged} = \frac{1}{2} + \frac{1}{6R_N} \sum_{i}^{models} R(O_i) \tag{7}$$

R_N is a normalization parameter that limits the opinion rating functions around -1 to 1. The final merging equation per each proposed opinion rating function is (8).

$$\begin{aligned} y_{merged} &= \frac{1}{2} + \frac{1}{6\tan(\frac{\pi}{2.2})} \sum_{i}^{models} \tan((O_i - 0.5)\frac{\pi}{1.1}) \\ y_{merged} &= \frac{1}{2} + \frac{1}{6\sinh\pi} \sum_{i}^{models} \sinh(2(O_i - 0.5)\pi) \\ y_{merged} &= \frac{1}{2} + \frac{2^{n-1}}{3} \sum_{i}^{models} (O_i - 0.5)^n \end{aligned} \tag{8}$$

Table 24 lists the confusion matrix of all proposed methods. The worst opinion rating function over-passes the AVG's performance, as well as, the miss-classification per classes have been decreased. However, not all opinion rating function perform the same, the best opinion rating function is the *sinh* in terms of test accuracy and miss-classification. The OP method is the best method for merging binary classifiers based on judging their outputs independently of their validation accuracy, but, how large region around 0.5 can be considered meaningless?

In order to reply that inquiry and given the easy interpretability of the polynomial function: the higher the value of n, the more region around 0.5 will be devalued, as seen in figure 24. Three additional experiments have been executed with different values of n.

-	OP-tan M	OP-tan C	OP-sinh M	OP-sinh C	OP-pol3 M	OP-pol3 C
Actual M	0.470	0.0625	0.470	0.0625	0.467	0.0655
Actual C	0.0565	0.4107	0.0535	0.4136	0.0535	0.4137

Table 24. Confusion Matrix of Opinion Rating approach. OP-tan: tangent as opinion rating function, OP-sing: hyperbolic sine as opinion rating function, OP-pol3: 3-degree polynomial as opinion rating function

The results of these experiments are listed on table 25, three confusion matrix are portrayed with different value of n = 5, 29 and, 63. n = 63 was chosen because of the numpy library errors due to small quantities in the image of the polynomial function.

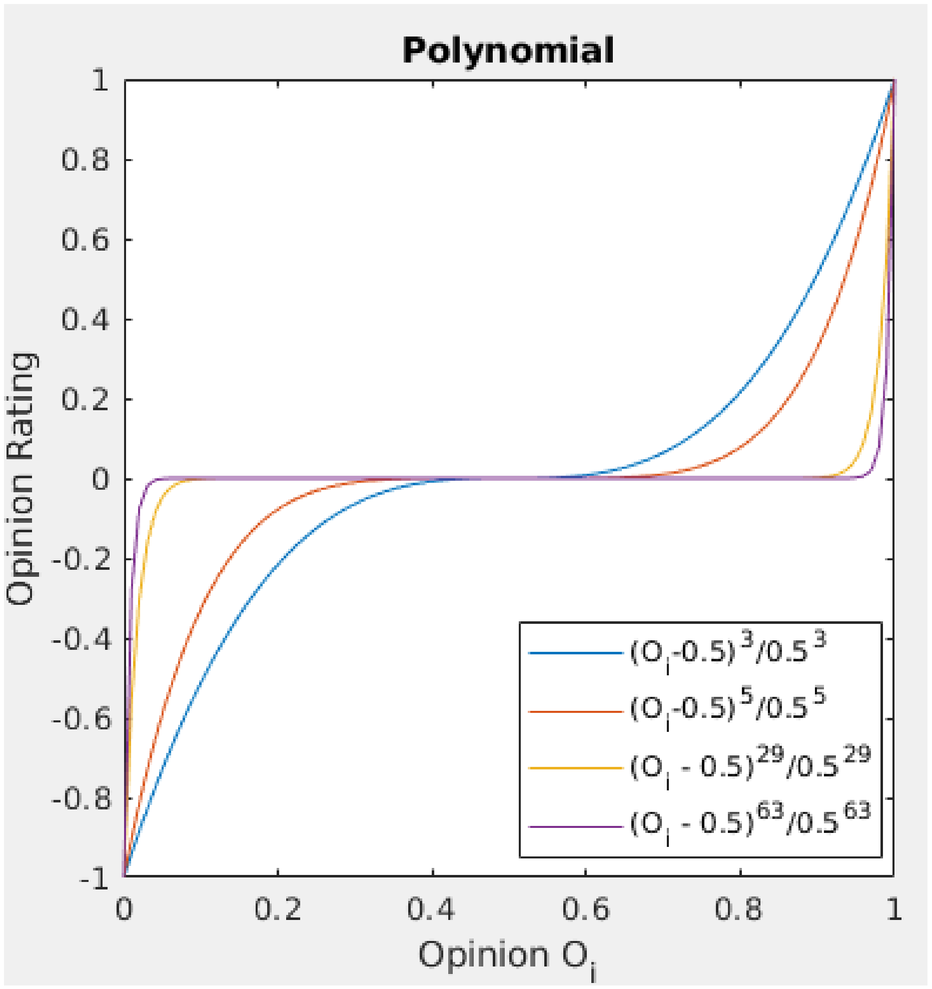

Fig. 24. Polynomial Opinion Rating Functions with n = 3, 5, 29 and, 63.

-	OP-pol5 M	OP-pol5 C	OP-pol29 M	OP-pol29 C	OP-pol63 M	OP-pol63 C
Actual M	0.470	0.0625	0.464	0.0685	0.425	0.1071
Actual C	0.0535	0.4136	0.04464	0.423	0.0327	0.4345

Table 25. Confusion Matrix of Opinion Rating approach. OP-pol5: n = 5, OP-pol29: n = 29, OP-pol63: n = 63

With $n = 5$, the polynomial approach has the same result as the proposed $sinh$. This means $n = 3$ was not devaluing many vague opinions; however, the larger the value of n does not mean a higher accuracy because eventually a big chunk of opinions are devalued and the input's class is decided by vague opinions as well.

To summarize, the Opinion Rating function is the best approach for merging models' output even with the worst OP function in comparison with simple AVG method. The OP function and its parameters are hyper parameters for the ensemble system; hence, they should be chosen at the validation stage, although in this work it was done with testing dataset due to lack of time. Based on the few experiments performed, the accuracy seems to be a concave curve with respect to the OP function and its parameters, so, it is not necessary another neural network to train to find the parameters of the OP functions, but a non-linear optimization methods can be performed to choose the right parameters and equation that maximise the accuracy and minimise the miss-classifications.

Bibliography

[1] Aguirre;, Z. H. S. Z. B. G.-Z. J. J. and Vanegas, A. M. (2020). Breast cancer histopathology image classificationusing an ensemble of deep learning models. https://www.mdpi.com/1424-8220/20/16/4373. Accessed: 2021-01-05.

[2] Arevalo, J., González, F. A., Ramos-Pollán, R., Oliveira, J. L., and Guevara Lopez, M. A. (2016). Representation learning for mammography mass lesion classification with convolutional neural networks. *Computer Methods and Programs in Biomedicine*, 127:248 – 257.

[3] Ball, J. E. and Bruce., L. M. (2007). Digital mammographic computer aided diagnosis (cad) usingadaptive level set segmentation. *2007 29th Annual International Conference of the IEEE Engineering in Medicine and Biology Society*, page 4973–497.

[4] Bengio, X. G. B. (2011). Deep sparse rectifier neural networks. http://proceedings.mlr.press/v15/glorot11a/glorot11a.pdf. Accessed: 2021-01-07.

[5] Chen, G., Chen, Y., Yuan, Z., Lu, X., Zhu, X., and Li, W. (2019). Breast cancer image classification based on cnn and bit-plane slicing. In *2019 International Conference on Medical Imaging Physics and Engineering (ICMIPE)*, pages 1–4.

[6] D., M. V. K. K. S. (2015). Human-level control through deep reinforcement learning. *Nature 518*, page 529–533.

[7] Daniel Lévy, A. J. (2016). Breast mass classification from mammograms usingdeep convolutional neural networks. https://arxiv.org/pdf/1612.00542.pdf. Accessed: 2021-01-03.

[8] E., K. A. S. I. H. G. (2012). Imagenet classification with deep convolutionalneural networks. https://papers.nips.cc/paper/2012/file/c399862d3b9d6b76c8436e924a68c45b-Paper.pdf. Accessed: 2021-01-01.

[9] Image-net.org (2014). Results of ilsvrc2014. http://www.image-net.org/challenges/LSVRC/2014/results. Accessed: 2021-01-01.

[10] Lee, Rebecca Sawyer, e. a. (2017). A curated mammography data set for use in computer-aided detection and diagnosis research. *Sci Data*, 4.

[11] MathoWorks (2020). inceptionv3. https://it.mathworks.com/help/deeplearning/ref/inceptionv3.html. Accessed: 2021-01-02.

[12] Ranzato;, Y. L. S. C. R. H. M. and Huang, F. J. (2006). A tutorial on energy-based learning. http://yann.lecun.com/exdb/publis/pdf/lecun-06.pdf. Accessed: 2021-01-07.

[13] Sergey Ioffe, C. S. (2015). Batch normalization: Accelerating deep network training by reducing internal covariate shift. https://arxiv.org/abs/1502.03167. Accessed: 2021-01-07.

[14] ul Hassan, M. (2018). Vgg16 – convolutional network for classification and detection. https://neurohive.io/en/popular-networks/vgg16/. Accessed: 2021-01-01.

[15] Wojna, C. S. V. V. S. I. J. S. Z. (2015). Rethinking the inception architecture for computer vision. https://arxiv.org/abs/1512.00567. Accessed: 2021-01-02.