UNIVERSIDADE D
COIMBRA

Catarina Alexandra Almeida Carvalho

Evolutionary Machine Learning in Radiomics: A Case Study on Breast Cancer

Dissertation in the context of the Integrated Master's Degree in Physics Engineering, Specialization in Instrumentation, advised by Assistant Professor Nuno Lourenço of Department of Informatics Engineering of University of Coimbra, Principal Investigator Nickolas Papanikolaou of Champalimaud Foundation, Full Professor Leonardo Vanneschi of NOVA Information Management School and presented to the Faculty of Sciences and Technology / Department of Physics

September 2021

UNIVERSIDADE D
COIMBRA

Catarina Alexandra Almeida Carvalho

Evolutionary Machine Learning in Radiomics: A Case Study on Breast Cancer

Dissertation in the context of the Integrated Master's Degree in Physics Engineering, Specialization in Instrumentation, advised by Assistant Professor Nuno Lourenço of Department of Informatics Engineering of University of Coimbra, Principal Investigator Nickolas Papanikolaou of Champalimaud Foundation, Full Professor Leonardo Vanneschi of NOVA Information Management School and presented to the Faculty of Sciences and Technology / Department of Physics

September 2021

ABSTRACT

The mortality and incidence of cancer rates around the world are increasing every year, resulting in a push for new screening and diagnostic technologies. One screening technique is medical imaging, such as X-ray mammography. This method collects images that are later analysed and evaluated by a radiologist. Nevertheless, this analysis depended on the protocols and expertise of the radiologist and is subject to high variance. To improve the diagnostic results, and reduce the variability associated with it, one can resort to computational methods, such as Radiomics. Radiomics is the process that transforms medical images into radiomic features, which can afterwards be used to train Machine Learning (ML) algorithms to perform tasks automatically. However, the increase of the problems complexity, makes traditional Machine Learning ineffective when dealing with complex tasks such as cancer diagnostic. Therefore, to overcome this problem we can resort to Deep Learning (DL) algorithms, which allow the extraction and learning of a large amount of features automatically. DL models can be constructed from scratch or reusing models trained on other problems. Both the methods have some problems when dealing with small datasets, such as medical imaging datasets, which can lead to poor performance or overfitting. Taking this into account, there is a necessity to resort to new approaches to construct lightweight DL models. One novel and promising approach is to use DENSER, a rencently defined algorithm that combines Machine Learning and Evolutionary Computation approaches, to automatically design Deep Learning models and select the best one based on a quality metric.

In this work, we explore the application of DENSER in a digital screening mammography dataset. Two experiments were performed using different levels of information. In one we used cropped images of 90x90 pixels and in the other we used cropped images with more surrounding information (250x250 pixels). Each experiment includes three studies, in which the networks were evolved through 150 or 300 generations, with or without data augmentation. After, and considering the best networks evolved by DENSER, an attention heat map was implemented to get an interpretation of how the model worked. Finally,

we conducted a study using an ensemble of the 4, 3, or 2 fittest networks.

Our results show that the implementation that uses 300 generations and data augmentation obtained the lightest networks and the best results, with a test accuracy of 73.81% in the first experiment (90x90 images) and 71.96% in the second (250x250 images). Furthermore, the assembling of the 4 fittest networks resulted in increased accuracy and a decrease in the number of misclassification cases. For the case of the 90x90 images, we got an increase of 2.38% in the test accuracy value and 9 fewer misclassification cases, whilst for the 250x250 images the values were 2.64% and 10 fewer misclassified.

Our results outperform the state of the art obtaining an increase of 2.62% in accuracy, using less training time, fewer epochs, and not requiring domain or expert knowledge. Finally, the effectiveness of DENSER in handling small datasets is a remarkable outcome.

Keywords

DENSER; Breast Cancer; Machine Learning

RESUMO

A mortalidade e incidência do cancro em todo o mundo vem a aumentar todos os anos, resultando num incentivo para o desenvolvimento de novas tecnologias de rastreio e diagnóstico. Uma técnica de rastreio é a imagiologia médica, como por exemplo a mamografia de raio-X. Esta técnica recolhe imagens que são posteriormente analisadas e avaliadas por um radiologista. No entanto, esta análise depende dos protocolos e da perícia do radiologista, estando sujeita a uma grande variação. Para melhorar os resultados de diagnóstico, e reduzir a variabilidade, pode-se recorrer a métodos computacionais, como por exemplo a Radiómica. A Radiómica é o processo que transforma as imagens médicas em características radiómicas, que podem depois ser utilizadas para treinar algoritmos de Aprendizagem Computacional (AC) para executar tarefas automaticamente. Contudo, o aumento da complexidade dos problemas, torna o AC tradicional ineficaz quando se lida com tarefas complexas como o diagnóstico do cancro. Portanto, para ultrapassar este problema, podemos recorrer a algoritmos de Aprendizagem Profunda (AP), que permite a extração e aprendizagem automática de uma grande quantidade de caracterisiticas. Os modelos AP podem ser construídos de raiz ou através da reutilização de modelos treinados para outros problemas. Ambos os métodos têm alguns problemas ao lidarem com pequenos conjuntos de dados, como por exemplo os dados de imagens médicas, o que pode originar um desempenho inferior ou sobreajustamento. Tendo isto em conta, há necessidade de recorrer a novas abordagens para construir modelos leves de AP. Uma abordagem inovadora e possível é utilizar o DENSER, que combina abordagens de AC e Computação Evolucionária, para projetar automaticamente modelos de AP e selecionar o melhor com base numa métrica de qualidade.

Neste documento, exploramos a aplicação do DENSER num conjunto de dados de mamografia digital de rastreio. Duas experiências foram realizadas utilizando diferentes níveis de informação. Numa utilizámos imagens recortadas de 90x90 pixels e na outra utilizámos imagens recortadas com mais informação envolvente (250x250 pixels). Cada experiência inclui três estudos, nos quais as redes foram desenvolvidas ao longo de 150

ou 300 gerações, com ou sem aumento de dados. Depois, e considerando as melhores redes desenvolvidas pelo DENSER, foi implementado um mapa de calor de atenção para obter uma interpretação de como o modelo funcionava. Finalmente, realizámos um estudo utilizando a junção das 4, 3, ou 2 redes mais adequadas.

Os nossos resultados mostraram que a implementação de 300 gerações e aumento de dados obteve as redes mais leves e os melhores resultados, com uma precisão de teste de 73.81% na primeira experiência (90x90 imagens) e 71.96% na segunda (250x250 imagens). Além disso, a junção das 4 redes mais adequadas resultou numa maior precisão e numa diminuição do número de casos mal classificados. Para o caso das imagens 90x90, obtivemos um aumento de 2.38% no valor da precisão de teste e menos 9 casos mal classificados, enquanto que para as imagens 250x250 os valores foram 2.64% e menos 10 casos mal classificados.

Os nossos resultados superam os mencionados na literatura obtendo um aumento de 2.62% na precisão, utilizando menos tempo de treino, menos épocas, e não requerendo domínio ou conhecimento especializado. Finalmente, a eficácia do DENSER no tratamento de pequenos conjuntos de dados é um resultado notável.

Palavras-Chave

DENSER; Cancro da Mama; Aprendizagem Computacional

ACKNOWLEDGMENTS

This dissertation was partially supported by national Portuguese funds through the Portuguese Foundation for Science & Technology (FCT) under project PTDC/CCI-INF/29168/2017 (BINDER).

To my advisors, Professor Nuno Lourenço, Dr. Nikolaos Papanikolaou and Professor Leonardo Vanneschi, I would like to acknowledge all the support, guidance and advice. Thank you for your patience, availability and for all the teachings. You have given me opportunities for which I will always be grateful.

To my "Flores dos Combatentes", Alexandra, Sara, Joana, Catarina, Paula, Viktorya, Beatriz, Cláudia, Júnia, Carla, Inês, Margarida, Rita and Telma, who accompanied me on this beautiful journey. It has been a pleasure to share these 5 years of home and academic life with you. Thank you for the support, the happy moments, the thousands of ice creams and for all the conversations and ideas shared. I will carry you for life.

To Paula, who accompanied me in my first year and made a huge impact on me. You were an inspiration to me from the beginning.

Alexandra, the best roommate, you brought happiness and peace to this journey. Without you, it wouldn't have been the same. Thank you for all the long conversations, for all the support, for calming me down, and for teaching me so much. I will never forget you.

To Sara, Joana and Catarina, a thank you is not enough. Your support was unconditional and your friendship and loyalty were crucial in these 5 years. Since the beginning, you have been there, in the good and in the bad times, in the joys and in the tears. Catarina, thank you for your positive thinking and teachings. Joana, thank you for all the support and for never letting me down. Sara, thank you for always going along with my craziness and for the long talks. You are an inspiration and I want to carry you through life.

Rute, you were the best buddy I could have on this journey. There were many shared moments of happiness, companionship, stress and learning. You have made a difference in these last two years, and a thank you is not enough.

Pedro, thank you for the long friendship, the support and for making me see life with new eyes. You were an important pillar in my growth in these 5 years.

Finally, to my family who supported me unconditionally in all my decisions and stages. Who fought with me to get me this far and who gave me all the love, encouragement and protection. I couldn't thank you all in a lifetime. Without you, this would not be possible.

LIST OF ACRONYMS

AI Artificial Intelligence. 5, 10

ANN Artificial Neural Networks. 2, 6, 13, 15, 19

AUC Area Under the Curve. 23

AutoML Automatic Machine Learning. 14

BNF Backus-naur form. 16

CBIS-DDSM Curated Breast Imaging Subset of the Digital Database for Screening Mammography. xi, 27

CC Craniocaudal. xi, xii, xiii, 26, 27, 30, 31, 46, 58

CNN Convolutional Neural Networks. xi, xiii, 2, 8, 10, 14, 15, 16, 17, 19, 21, 22, 23, 28, 42, 50, 52, 55, 57, xxxviii, xxxix, xl, xli, xlii, xliii, xliv

CT Computed Tomography. 19

DENSER Deep Evolutionary Networks Structured Representation. xii, xiii, xiv, xv, xvi, xvii, 3, 4, 15, 16, 28, 31, 33, 38, 40, 41, 42, 43, 44, 45, 46, 49, 50, 52, 53, 55, 56, 57, 58, 61, 62, 64, xxxviii, xxxix, xl, xli, xlii, xliii, xliv, xlv, xlvii, l

DL Deep Learning. 1, 6, 10, 14, 21

DNN Deep Neural Networks. 2, 6, 14, 15, 16

DSGE Dynamic Structured Grammatical Evolution. xi, 15, 16, 17, 18

EA Evolutionary Algorithms. 10, 11, 14

EC Evolutionary Computation. 10, 13, 14

EDL Evolutionary Deep Learning. 14

EML Evolutionary Machine Learning. 2, 3, 13, 14

FC Fully Connect. 10

FN False Negative. 29, 40, 43, 48, 53, 58, 60, 64

FP False Positive. 29, 43, 53, 58, 60, 64

FPR False Positive Rate. 29, 40, 43, 46, 48, 50, 53, 57, 60, 64

GA Genetic Algorithms. xi, 11, 15, 16, 17, 18

GP Genetic Programming. 11, 12, 13, 14

GPU Graphics Processing Units. 14, 19

ML Machine Learning. 1, 5, 6

MLO Mediolateral Oblique. 26, 27

MRI Magnetic Resonance Imaging. 19, 20

NE NeuroEvolution. 13

NEAT NeuroEvolution of Augmenting Topologies. 13

NN Neural Networks. 6, 21

PCA Principal Component Analysis. 5, 21

PET Positron Emission Tomography. 19, 20

RL Reinforcing Learning. 14

RNN Recurrent Neural Networks. 22

ROC Receiver Operating characteristic. 29, 39, 40, 42, 45, 47, 57, 59

ROI Region of Interest. 19, 20, 21

SVM Support Vector Machine. 6, 21, 22, 23

TN True Negative. 29

TP True Positive. 29

TPR True Positive Rate. 29, 40, 43, 46, 48, 50, 53, 57, 60, 64

LIST OF FIGURES

LIST OF TABLES

LIST OF GRAMMARS

CONTENTS

CHAPTER 1

INTRODUCTION

Cancer mortality and incidence are growing worldwide, being one of the leading causes of death across the globe. According to GLOBOCAN 2020 [1], worldwide in 2020, there were 19.3 million new cancer cases that resulted in 10 million cancer deaths. According to the same source, female breast cancer was the most diagnosed (11,7%), and the deadliest was lung cancer (18%), followed by colorectal (9,4%). The trend is that these numbers will grow globally in the coming years. As such, it is necessary to invest in methods and techniques for early, rapid, and accurate cancer diagnostic.

X-ray mammography is a medical imaging modality that is non-invasive ant it has been used for screening breast [2]. Following image acquisition analysis and evaluation by breast radiologists is performed. A major problem is the fact that diagnostic assessement heavily depends on the radiologists' expertise and therefore it is often the case where double reading is performed to avoid medical errors. However, there is currently a mismatch regarding the number of available board certified breast radiologists and the diagnostic demands which promotes the idea, of introducing virtual AI based readers or assistants in the screening process. It is here that the combination of the Radiomics, which is the transformation of the clinical images into radiomic features, and the Machine Learning (ML) algorithms, which resort to these features to train computational models to perform tasks automatically, emerges [3].

Radiomic features can be attained through hand-crafted methods (hand-craft features), i.e., based on predefined mathematical transformations, or through Deep Learning (DL) algorithms (deep features), a subfield of ML. These features are subsequently used to train an ML algorithm that will make predictions based on this information. However, the increase of the problems complexity makes the use of traditional ML models ineffective when dealing with highly complex tasks, such as in medicine [4] and in computer vision [5]. To overcome this problem, we can resort to DL algorithms since they allow the extraction and learning of large sets of data representations automatically using

Artificial Neural Networks (ANN). When dealing with image datasets the most common DL architecture used is based on Convolutional Neural Networks (CNN). The DL models can be constructed from the scratch or using Transfer Learning [6]. The first one enables the networks to be completely adjusted to the problem, but vulnerable to the presence of overfitting and class imbalance. One way to tackle this problem is to resort to Transfer Learning using a Pre-trained Network on a large image dataset, and afterward, tune it to a medical dataset [6]. However, transfer learning can have some issues, namely concerned with the dissimilarity between the domain of application. Usually, networks are trained on datasets such as ImageNet that have nothing in common with the medical domain. As such, when transferring networks to the medical domain, the models will have poor performance if not adjusted [7]. Another problem is the fact that most of the networks are large and complex, and the medical datasets are small which can result in overfitting. Therefore, there is the need to make changes in the network, which most of the time are difficult or impossible to do. Taking all of this into account, there is an interest to resort to new approaches to construct lightweight DL models. One novel and possible approach is to use Evolutionary Algorithms, which are based on Darwinian natural selection principles, to automate the design of the DL models [8]. The combination of computational methods based on Evolutionary principles and Machine Learning is an active field of research collectively known as Evolutionary Machine Learning (EML).

In recent years, several proposals have emerged to automate the design of DL models, with DENSER [9] being one remarkable example. This evolutionary approach allows for the automatic design and parameterisation of Deep Neural Networks (DNN) and chooses the best network for each case.

The main objective of this work is to use DENSER to automate the design and parameterisation of the DNN, to deal with small medical imaging datasets, such as breast cancer datasets. The evolved networks will transform the medical images into features, then to build and train artificial neural networks. All the process is made without resorting to the domain or expert knowledge, i.e., fully automated.

This work is conducted within the context of the BINDER [1], which aims at improving the state of the art in Radiomics analysis of breast cancer, using existing and novel Machine Learning (ML) and Deep Learning (DL) methods. This project results from the collaboration of a multidisciplinary team of researchers from the University of Coimbra, Champalimaud Foundation, NOVA University of Lisbon, and University of Lisbon.

[1] https://www.cisuc.uc.pt/en/projects/binder-improving-bio-inspired-deep-learning-for-radiomics

1.1 Motivation

To build a Deep Learning model to a certain problem, practitioners and researchers can either develop and design the models from scratch, reuse models that have been successful in solving similar problems, or do a combination of both. However, when dealing with problems where the datasets are not very large, such as the medical domain, reusing models from other domains is problematic since they tend to be rather large and complex, which can lead to overfitting when dealing with small datasets. To overcome this obstacle, one needs to develop models from scratch, which can be a hard and laborious task. As such, one can resort to methods that are able of designing full DL models such as DENSER. DENSER relies on Evolutionary Computation to automate the design and parameterisation of the ML models and it has shown a remarkable performance on several computer vision problems (One dataset studied was the CIFAR-10, which is composed of 60000 images [9]). However, its effectiveness has never been measured on problems from the medical domain, where the datasets are small (in this study the dataset used was composed of 1696 images [10]).

Using an automatic approach has also its advantages since we can tailor the models to a problem at hand and we might even see novel models emerge from the automatic design process.

Finally, applying this approach can result in an important contribution to society as a whole, since we might discover models that can help improve the screening and detection of cancer.

1.2 Contributions

The contributions of this work are summarised below:

1. The application of DENSER to medical imaging datasets, i.e., using EML to automate the design and parameterisation of ML algorithms to deal with small medical imaging datasets. This is the first application of DENSER in such domain.

2. DENSER allowed us to acquired novel topologies achieving better performances with no domain knowledge. Comparing the results of our work with those from the literature [11], we obtained an improvement of 0.77% to 2.62% on unseen data. Additionally, the models require less training time and fewer number of epochs.

3. We developed attention heatmaps to understand if the model is "looking" in the right region of the input image to help understand the results. Moreover, this approach demonstrates that the inclusion of the surrounding information allows the networks to dismiss the images' black background, which does not happen when we focus more on the region of the lesion.

1.3 Structure

Chapter 2 of this document, **Background**, provides the knowledge bases to understand the work made in this study. It focuses on the topics of Machine Learning, Evolutionary Computation, DENSER and Radiomics.

Chapter 3, **Radiomics and DENSER**, describe the implementation of Radiomics in DENSER. It includes an explanation and description of the dataset used, the grammar, the experimental parameters, and the two image pre-processing works investigated. Furthermore, it contains an explanation of the model interpretability used.

Chapter 4 presents the **Experimental Results** of the two experiments made in this work and their comparison with the literature results. Each investigation contains three studies, which differ in the number of generations and the application or not of data augmentation approaches. Furthermore, it includes the results of the model interpretability of the best network obtain in each experiment and a study of the ensembling of the 2, 3, and 4 fittest networks.

Chapter 5, and the last one, presents the **Conclusion** of the work as a whole and looks through the possibilities for **Future Work** based on this work.

CHAPTER 2

BACKGROUND

This chapter provides the knowledge foundation to understand the experiments performed in this work. Therefore, it begins with an introduction to Machine Learning, which will be more focused on the Convolution Neural Networks, a Deep Learning architecture. Afterward, the Evolutionary Computation field will be introduced, more specifically Evolutionary Machine Learning and our approach, Deep Evolutionary Network Structured Representation. Finally, the state-of-art of the Radiomics field will be introduced.

2.1 Machine Learning

Machine Learning (ML) is a field of Artificial Intelligence (AI) that aims to create computational models to perform tasks automatically using complex large amounts of data. From these models, statistical structures are founded and rules will be created to automate the desired task [12, 13]. This process is time-consuming since it requires multiple trials in order to achieve the best set of features to train the model. This means that different tasks required different features to be extracted to train the model, making it a challenging problem [9].

ML can be divided into unsupervised or supervised learning. Both are fundamentally distinguished by the way the data is labelled.

In unsupervised learning the samples in the dataset are unlabelled. Hence, in this learning, the feature vectors extracted from the dataset are taken as an input, and they are transformed into other vectors using for example the Principal Component Analysis (PCA), or into a value by implementing Clustering, for example. Thus the algorithm learns the format of the dataset and the relation between the features [13, 14].

The dataset in supervised learning is labelled, i.e., we know beforehand what the target's model is. The label can belong to a set of classes or it can be a real number or a more complex structure, such as a vector, a matrix, a tree, or a graph [13, 14].

Most supervised learning algorithms use the training data to create a model where its parameters are learned. In this type of learning, a feature vector extracted from the dataset is the input's model. After learning, we get a prediction of the feature vector label. Therefore, the supervised learning' data come in pairs (input, output) [13].

The different classes are separated by decision boundaries. These boundaries can be linear or non-linear. If we want to use arbitrary non-linear boundary decisions, normally the kernel is used [13].

Two popular supervised learning algorithms are:

- Support Vector Machine (Support Vector Machine (SVM)) [14];
- Neural Networks (Neural Networks (NN)) - Composed of layers made of neurons, which are connected using weights [14].

Supervised learning is composed of classification or regression models.

Classification models are constructed based on labelled examples and based on a classification learning algorithm. After training the classification model, an unlabelled example is applied as input. Afterward, the model outputs a label or a value, such as a probability. Ultimately, the example is classified into one of the predefined labels of a set of classes. When the number of classes is two we are facing a binary classification problem [13].

Regression models, which are based on supervised learning, make use of regression learning algorithms. After training the model, and considering an unlabelled example given as input, a real-valued label is provided [13].

2.1.1 Deep Learning

As mentioned previously, in ML, experience and many hours of trial and error work are required in order to select and engineer the best features for a given model. To overcome this issue one can be use Deep Learning DL). DL is a subfield of ML that allows the extraction and learning of large sets of data representations automatically using Artificial Neural Networks (ANN). ANNs are composed of layers: an input layer, hidden layers, and an output layer. When the ANNs are composed of several non-output layers, i.e., hidden-layers, they are considered Deep Neural Networks (DNN). The layers are made up of units, called neurons, which are connected to the units of the following layer. These connections between neurons have associated weights that represent the strength of the connection. The first layer receives the input, which is divided into small batches, and the next layers (hidden layers) will apply transformations to generate more abstract representations. The output layer produces a prediction. An example of a network is displayed in Figure 2.1 [12, 13, 9].

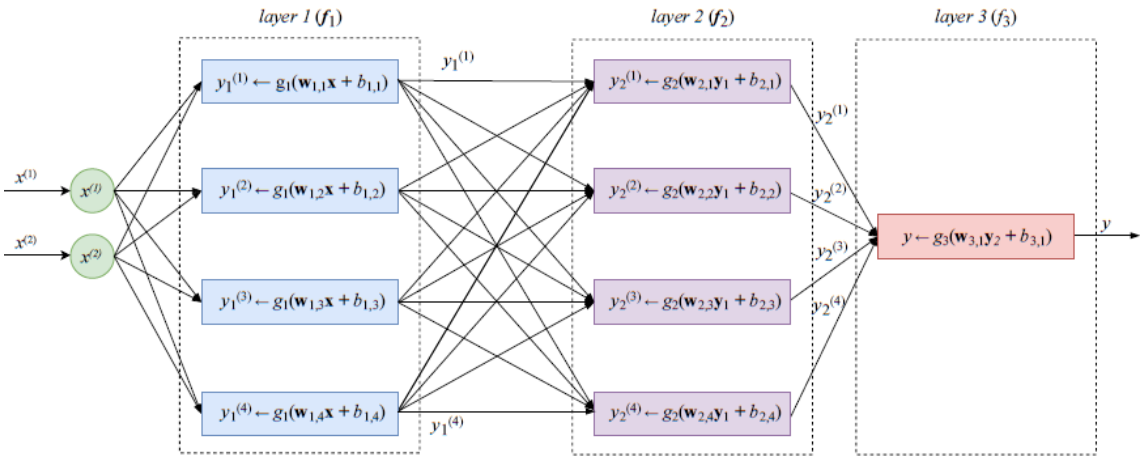


Figure 2.1: Example of a network with two-dimensional input, two layers with four units and one output layer with one unit [13].

Based on Figure 2.1 a 3-layer neural network, f_{NN}, can be represent like this [13]:

$$f_{NN} = f_3(f_2(f_1(x))) \tag{2.1}$$

The function f_3 and f_2 are represent as following [13, 12]:

$$f_L(i) = g_i(W_i z + b_i) \tag{2.2}$$

where i is the layer index, g_i is the activation function, W_i is a matrix of weights, z is the value of the neuron of the previous layer, and b_i a vector.

Finding the most accurate, efficient, and generalizable architecture, for a given task, in an acceptable time is a challenging task.

Based on [12], the construction of a training model consists of the following steps:

1. Definition of the training data: input and target multidimensional arrays, i.e., input and target tensors;
   (a) It is necessary to convert the integer lists into tensors lists;
2. Construction of the network;
   (a) Characterization of the activation function;
      i. This function allows the measurement of the importance of the weights in a specific neuron;
      ii. The most common activation function used in the hidden layers is the RELU activation function [13];

$$RELU(z) = \begin{cases} 0 \text{ , if z} < 0 \\ z \text{ , otherwise} \end{cases} \tag{2.3}$$

3. Configuration of the process hyper parameter selection, such as loss function, optimizer, and metrics to evaluate the model (example: accuracy);

   (a) The loss or cost function provides the metric to assess the performance during the training, which represents a measure of success or failure of the model. This value is achieved through forwarding propagation and comparing the output predictions with the ground truth labels. The chosen function will depend on the type of problem. One example of a loss function for multiclass classification is cross-entropy [15]. This function plays an important role, considering it guides the whole learning process;

   (b) The optimizer defines how the model will adjust its parameter, i.e., weights, taking into account the loss function;

      i. Gradient Descent is commonly used to obtain the minimum of the loss function, L [13]. It uses a backpropagation algorithm that enables us to know how much the function is sensitive to a change in the learnable parameters, w, such as weights and bias. The parameters are updated in the negative direction of the gradient, taking into account the learning rate, α, [15]. The update is formulated as follows [15]:

$$w := w - \alpha * \frac{\partial L}{\partial w} \tag{2.4}$$

      ii. It is required to choose a learning rate optimization algorithm.

To assess the quality of the model, there are three evaluation protocols: Hold-out, K-fold cross-validation and K-fold validation [12]. Focusing only on the Hold-out method, this method is implemented when a large amount of data is available. The data is held out into train, validation, and test set. The network is trained with the training set, and the evaluation of its generalisation ability during the training is performed using the validation set where the hyperparameters are tunned. The performance of the final model is acquired in the test set [15].

Our model can underfit or overfit and both situations need to be avoided. Overfitting occurs if the model performs better on the training set rather than on the validation set. If it performs poorly on training and validation, the model is underfitting. Overfitting can be recognized through monitorization of the loss and the accuracy of the training and validation sets. Some methods such as obtaining more training data, regularization with dropout, weight decay, batch normalization and data augmentation can be applied to mitigate overfitting [15].

Convolution Neural Networks

Deep Learning includes different architectures with Convolution Neural Networks (CNN) being one of them. This architecture is commonly used in computer vision and

image processing [13, 16].
Images are composed of a set of pixels with information and they are considered input of high dimensions [13]. This information must be converted into input tensors to be provided to the network. To do that the following instructions can be performed [12]:

1. Read the image;
2. Decode the image format in grids of pixels (RGB);
3. Convert into floating-point tensors, normally, 3D (height, width, depth);
4. Adjust the pixels values (0-255) into 0-1 interval;

After this, we get the input tensors and one or more convolutions will be applied to the inputs. Convolution is a linear operation used for feature extraction and it resorts to the dot-product between parts of the input, called patches, and a filter, W_iz. In each input patch, a kind of pattern will be detected and learned. The filter, called kernel, is a small array of numbers (typically 3x3 or 5x5) obtained by a linear transformation, and it is automatically learned through the training process. The convolution operation is also defined by the stride, the distance between two successive kernel positions. After the convolution, a 1D vector is obtained. This transformation is applied to all the patches, then we assemble all 1D vectors obtained and reach 3D output feature maps [12, 13, 6, 15].
The Figure 2.2 details this process [12].

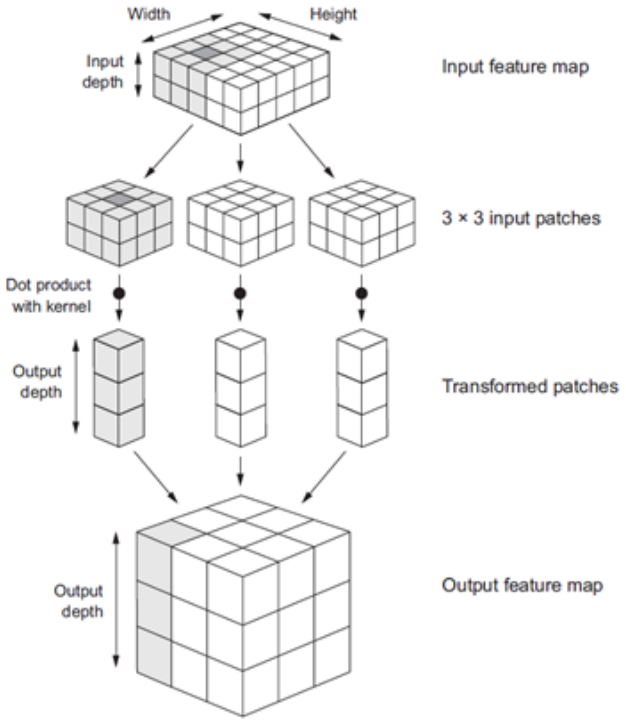


Figure 2.2: The convolution process [12].

Another way to downsample the feature map is by applying pooling (such as max or average pooling) that is similar to the convolution. The difference is in the filter, which in max pooling is obtained via a hardcoded max tensor operation that will output the max value of each patch, and in the average pooling is through average operations [12].

A basic structure of a CNN network consist of blocks of several convolution and a pooling layers, followed by fully connected layers [12, 15, 17]:

- Convolution layer;
- MaxPooling layers;
- Flatten layer - convert the tensor to a 1D tensor ;
- Fully Connect (FC) layer or dense layers - The neuron of one layer is connected to all the neurons of the next layer;

The most used CNN models in computer vision are AlexNet, VGG, GoogleNet and ResNet, which are composed of more than 20 million parameters [18, 15].

CNN has some properties, such as, if the pattern that was learned in a certain place, appears again, that pattern will be recognized. Another property is the ability to increase pattern recognition by increasing the number of convolution layers [12].

The CNN architecture can be divided into three categories: Standard architectures, Self-designed architectures, and Multiple CNNs. The architecture in our approach will be designed based on the specification of the problem. So, the architecture is self-designed [6].

Despite the fact that DL models extract and train the data automatically, the best model is obtained iteratively by trial-and-error. Accordingly, a big parameterization stage is needed which can be optimized by resorting to Evolutionary Computation (EC) methods [9].

2.2 Evolutionary Computation

Evolutionary Computation (EC) is an area of AI that research and develops nature-inspired algorithms [19]. Evolutionary Algorithms (EA) are computational models loosely based on Darwinian natural selection principles to find solutions to a problem. The implementation of these algorithms makes getting solutions faster and more flexible [8, 20].

When working with EA it is imperative to define a set of parameters: representation, fitness function, population, parent selection mechanism, variations operators, and survivor selection mechanism.

In natural selection, the individuals that fit the environmental condition more effectively are the ones that are favoured, i.e., the survival of the fittest. To evaluate the fitness in EAs a fitness function is used. The value obtained by the function for a certain individual will correspond to its quality [8, 20].

EA uses a population of solutions that later can exchange attributes among themselves. These solutions are referred to as phenotypes that are encoded into the corresponding genotype. A genotype represents each individual within the EA, and a multiset of genotypes represents a population. Due to mutations and recombination of genes (crossover) in reproduction, variations can occur in the genotype. The individual selection based on the behavioural and physical features that will affect the individual response to the environment occurs in the phenotype [20].

One of the pillars of evolutionary progress is competition-based selection that increases the quality of a population. The other one is related to the phenotypic random variations generated by genotypic variations. Both variation operators, crossover and mutation are applied stochastically. Crossover is a process that is applied with a higher probabilistic, combining the information of two or more solutions (parents) to create a new individual. The mutation is a variation that occurs on one individual and creates a new one by applying some randomized changes in the representation. The selection of the parents is based on the fitness value. The heuristic choice of the parents prevents premature convergence and loss of population diversity. Lastly, it is necessary to select the individuals that will create the next generation. The replacement mechanism can be based on two methods, fitness-based and age-based [20].
A candidate solution can be represented in four ways, depending on the method used [20];

- Fixed-length bit string representation – Genetic Algorithms (GA);
- Real-valued vectors – Evolution Strategies (ES);
- Finite state machines – Evolutionary Programming (EP);
- Trees and graphs with variable sizes – Genetic Programming (GP);

In this work, we will focus on GP.

2.2.1 Genetic Programming

This method uses instruction sets as attributes (for example a specific programming language) and it allows the automatic evolution of programs. This type of EA can be applied to machine learning and it is used to explore models with the maximum fit. The individuals are represented as trees and the fitness is the parameter to be maximized [8, 20].
Program trees are made up of nodes that are connected to other nodes (see Figure 2.3). If a node does not have a connection, it is called a leaf node or terminal (T). If it has is called a function (F) [8].

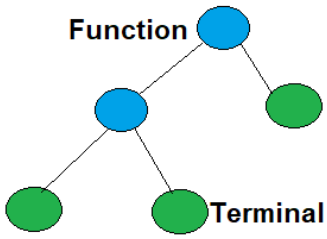
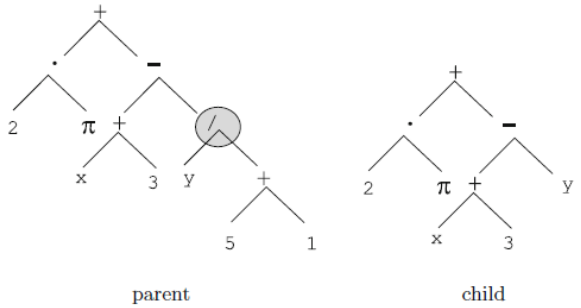
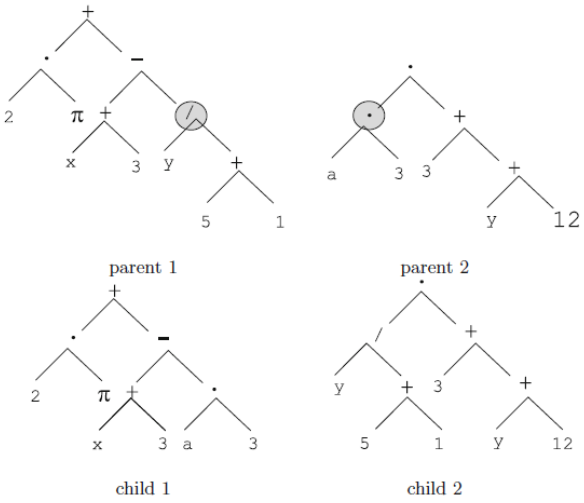


Figure 2.3: Structure of a program tree, where the blue nodes represent a function and the green nodes represent a terminal [8].



(a) Tree-based mutation.



(b) Tree-based recombination.

Figure 2.4: Example of the variations operators application: mutation and crossover. The operations act in the nodes designated by a circle in the parent trees [20].

Concerning initialization, the most common method to initialize a population is the ramped half-and-half method, where a maximum initial depth of the tree is chosen, D_{max}. Afterward, each individual is created from the two sets, F and T, employing the full method or the grow method, that are detailed in [20].
In GP, only one of the variation operators, crossover or mutation, is used at each reproductive step, and it is chosen stochastically. The tree-based mutation randomly selects

a node of the tree and replaces the subtree for a randomly generated tree (see Figure 2.4a). The tree-based recombination uses a binary operator to create children's trees from parents' trees by swapping subtrees between the parents (see Figure 2.4b) [20].

The most common selection method used in GP is tournament selection, where an arbitrary number of individuals is chosen to compete in small tournaments, and the fittest is selected to be a parent [8, 21].
The fitness in pattern recognition or object classification can be measured by the accuracy, although there are many ways to measure the fitness depending on the goals of the problem [21].
Some parameters in GP must be specified, such as population size, the performance probability of the genetic operations, the maximum programs size, and other details of the run. Other important parameters are the termination criteria that includes the maximum number of generations, the maximum time allowed to run the CPU, the maximum number of fitness evaluation, the periods where there is no improvement in the fitness and the threshold when there is decreased population diversity [20, 21].

2.3 Evolutionary Machine Learning

Evolutionary Machine Learning (EML) aims to use EC techniques to overcome some of the most common challenges in Machine Learning algorithms. These challenges include finding the optimal learning parameters and hyperparameters, and the structure which maximizes the prediction accuracy [14].

2.3.1 Neuroevolution

The EC can be used to automatically optimize ANN, and this set of methods is called NeuroEvolution (NE). Evolution can have different targets depending on the problem to be solved. Evolutionary algorithms can optimize the synaptic weights [22], search the activation's function and weights (EvoNN) [23], develop learning rules [24], and search for the network topology [25].

In their work, Soltanian et al. [26] used grammatical evolution to generate the network's architecture resorting to the backpropagation algorithm to train the network. Therefore, by implementing this method, expert knowledge is not necessary.

Turner et al. [27] studied the importance of evolving weight and/or the ANN's topology using NE. Using the topology evolution by itself is better than using weight evolution. However, the authors obtained more effective NE using both evolutions simultaneously. A method that applied both evolutions is NeuroEvolution of Augmenting Topologies (NEAT) [28].

Notwithstanding, optimizing weights and the topology is very hard when dealing with hundreds or thousands of weights and complex topologies. For this, we call upon evolutionary deep learning, where topology and learning hyperparameters are optimized

resorting to EC [19].

2.3.2 Evolutionary Deep Learning

A hot topic in EML is Evolutionary Deep Learning (EDL) that focuses on EC approaches to boost the performance of deep learning algorithms [19].
The proper configuration of DNN, which includes architecture and hyperparameters, will affect its performance and success. These structures are composed of hundreds of parameters and require more computational resources. Hence, to make configuration easier an Automatic Machine Learning (AutoML) system for deep learning was developed [29, 9].

The most recent researches have been focused on implementing EDL to discover architectures for specific tasks because there is not a guideline to choose the right one and it is a time-consuming task [29].

The automatization of the DNN's configurations can provide innovative, well-programmed, and small architectures, without the need for expert domain [29]. In addition, DNNs can achieve better performance, if they resort to hardware, such as Graphics Processing Units (GPU) [19].

Optimization of the hyperparameters can be made by grid search [29, 30], random search [30], and Bayesian optimization [30]. The search for the best architecture can be carried out by Reinforcing Learning (RL) [30] or EA [29, 30]. However, compared to RL, optimization using EAs enables dealing with larger search space [29, 31].

Alejandro et al. [32] propose EvoDeep, a new evolutionary algorithm to evolve the architecture and the parameters of a DNN.

Sun et al. [33] developed an optimizing method for deep learning architecture search resorting to evolution, called CoDeepNEAT. The optimization is implemented in the topology, components, and parameters of the DNN, and this method is applied to object recognition and language modelling.

EDL algorithms are divided into neural network-based (NN-EDL) and GP-based (GP-EDL) algorithms. The first one uses evolutionary algorithms to solve the most common problems of machine learning previously mentioned. The second one uses GP to achieve deep learning without neural networks [19, 34].

2.3.3 Evolutionary Convolutional Neural Networks

After applying EC to DL, these techniques are also starting to be applied to deep CNN for image classification [34].

Real et al. [35] evolved a large-scale evolution of image classifiers employing evolutionary algorithms to search automatically for the network. Later, Real et al. [31] developed AmoebaNet-A, which is also an image classifier but with a regularized evolution where an age property was introduced to benefit the younger genotypes.

Sun et al. [36] used genetic algorithms to evolve architectures, and the initial values for the connection weigh for a deep CNN. They represent the different blocks of the CNN

through variable-length gene encoding and propose a novel fitness evaluation method. The method allows speeding up the heuristic search with fewer computational resources. The mechanism outperforms state-of-the-art algorithms in terms of classification error. More recently, Sun et al. [37] propose an automatic CNN architecture research method by implementing genetic algorithms to do image classification tasks. This approach does not require domain knowledge, and it provides a promising architecture. Furthermore, the method outperforms the existing ones on classification accuracy, parameter numbers, and consumed computational resources.

2.4 Deep Evolutionary Network Structured Representation

During ANN construction, there are many decisions to take into consideration due to the fact that the models are very complex. The need to define the topology, structure, optimization parameters and the need for previous knowledge and domain expertise make these decisions hard to make as well as a time-consuming task. Therefore, there is the need to automate the design of these networks [38].

Deep Evolutionary Network Structured Representation (DENSER) is an evolutionary approach that allows us to evolve the structure and parameters of the DNNs. This novel approach combines the basics of Genetic Algorithms (GA) with Dynamic Structured Grammatical Evolution (DSGE) [38]. The code for DENSER is available at `https://github.com/fillassuncao/fast-denser3`.

Assunção et al. [9] tested the DENSER's capability in the classification task using object recognition. Therefore, this approach was applied to the automatic generation of CNNs. This approach was initially evaluated using the CIFAR-10 dataset. To test the CNN's robustness, generalization and scalability, obtained with CIFAR-10, other benchmarks were applied, more specifically, MNIST, Fashion-MNIST, SVHN, and CIFAR-100 datasets [9].

The results show that DENSER, trained on the CIFAR-10 dataset, outperforms previous evolutionary methods in the generations of CNNs using less prior knowledge and obtaining novel topologies that were never designed by a human being. The best performing CNN, obtained during the evolution, was composed of many dense layers which was a remarkable outcome. The results obtained by applying benchmarks enable us to conclude that the CNNs design by DENSER during evolution are robust, generalizable, and scalable [9, 38].

To overcome the challenge of automating the design of these networks, Assunção et al. [9] resort to evolutionary algorithms. These algorithms were implemented in the layers and network's structures. In the layer's type and parameters, the authors use DSGE, where the evolution acts on grammatical derivations. In the network's structures, the authors applied GA where the positions in the GA represent a layer and encode a list of genes [38]. So, there are two independent levels to represent the candidate solution: The GA level and the DSGE level [9].

The GA level is related to the macro structure of the DNNs and depicts the array of evolutionary units that produce the network. This requires a valid structure of the genotype (layers, learning, and/or data augmentation methods). An example of a GA structure for the evolution of CNNs is **[(features, 1, 10), (classification, 1, 2), (softmax, 1, 1), (learning, 1, 1)]**, composed by 4 modules, and where, feature, classification, softmax, and learning are the non-terminal symbols for the expansion into the DSGE level genotype. With this structure, we can have a candidate solution with 1 to 10 convolutions or pooling layers (features), 1 or 2 fully-connected layers (classification), 1 softmax layer (output), and a learning rule (algorithm and its parameters). This level facilitates the application of the genetic operators to evolve the candidate solution by allowing the encapsulation of the genetic information [9].

The DSGE level is implemented at the micro structures where the parameters for each GA evolutionary unit and its accurate length of parameters are defined. These parameters are defined in a Backus-naur form (BNF) grammar and are expressed as ranges or a closed set of possibilities. The use of this type of grammar allows the DENSER to be a general approach because it enables easy adaptation to different network types, layers, and tasks. This grammar also facilitated the use of domain specific knowledge straightforward. This grammar-based approach allows us to make these changes without having to make changes in the implementation details. An example of a grammar for the encoding of CNN is represented in Figure 2.5 and its interpretation is in Appendix A.1 [9].

```
<features> ::= <convolution>                                                    (1)
             | <pooling>                                                        (2)
<convolution> ::= layer:conv [num-filters,int,1,32,256] [filter-shape,int,1,1,5] (3)
                  [stride,int,1,1,3] <pad> <activation> <bias>                 (4)
                  <batch-normalisation> <merge-input>                          (5)
<batch-normalisation> ::= batch-normalisation:True                             (6)
                        | batch-normalisation:False                            (7)
<merge-input> ::= merge-input:True                                             (8)
                | merge-input:False                                            (9)
<pooling> ::= <pool-type> [kernel-size,int,1,1,5][stride,int,1,1,3] <pad>      (10)
<pool-type> ::= layer:pool-avg                                                 (11)
              | layer:pool-max                                                 (12)
<pad> ::= padding:same                                                         (13)
        | padding:valid                                                        (14)
<classification> ::= <fully-connected>                                         (15)
<fully-connected> ::= layer:fc <activation> [num-units,int,1,128,2048 <bias>   (16)
<activation> ::= act:linear                                                    (17)
               | act:relu                                                      (18)
               | act:sigmoid                                                   (19)
<bias> ::= bias:True                                                           (20)
         | bias:False                                                          (21)
<softmax> ::= layer:fc act:softmax num-units:10 bias:True                      (22)
<learning> ::= learning:gradient-descent [lr,float,1,0.0001,0.1]               (23)
```

Figure 2.5: Grammar example for CNN encoding [9]. The grammar interpretation is in the Appendix A.1.

These two types of genotypes and the decoding of each position into the GA level in the corresponding DSGE level are represented in Figure 2.6. In Fig.2.7 we can see the mapping phenotype for the DSGE genotype of Figure 2.6 [9].

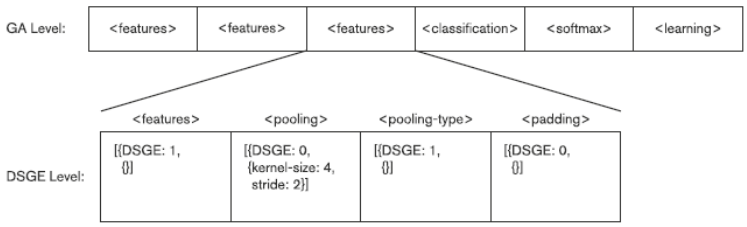


Figure 2.6: Example of the genotype of a candidate solution that encodes a CNN for both levels. The GA level is randomly built by the GA structure, [(features, 1, 10), (classification, 1, 2), (softmax, 1, 1), (learning, 1, 1)]. Subsequently, using the grammar in Figure 2.5 the DSGE level for the feature non-terminal is obtained [9].

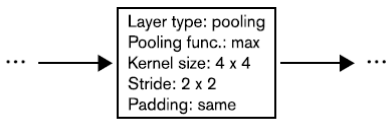


Figure 2.7: The corresponding phenotype of the genotype in Figure 2.6 [9].

To enable the evolution of the candidate's solutions, the authors rely on the probabilistic implementation of mutation and crossover operators. We have two crossover operators. The first one occurred in the GA level and, it changes layers within a specific layer module, i. e., a module is selected by tournament and is applied a one-point crossover in both parents' modules. On the DSGE level, the crossover operator swaps entire modules between individuals based on a binary representation. Figure 2.8 represents an application example of this operator in both levels. The mutation is also applied to both levels. At the GA level, the mutation occurs at the macro level, manipulating the structure, by adding, replicating, and removing a unit. In the DSGE level, the mutation happens at the micro level, changing the layer's parameters. These mutations include grammatical mutation, connection mutation, integer mutation, and float mutation [**DENSER.DENSER1**, 39].

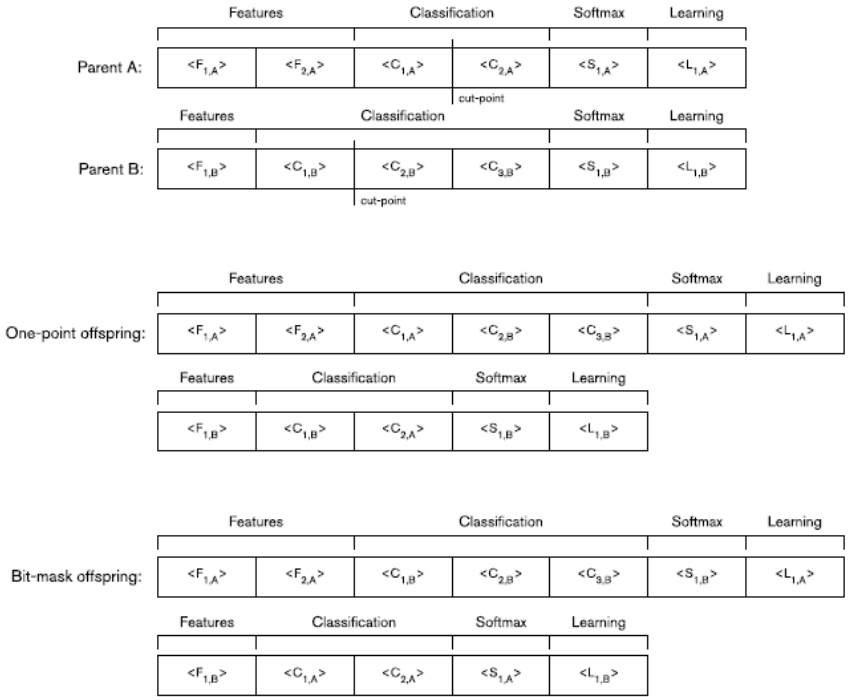


Figure 2.8: Example of both crossover operators. In GA level a cut-point in one of the modules of the parents is applied, originating two offspring by swapping. In DSGE level a bit-mask crossover, 1001, is applied where which position is associated with a module in the offspring [9].

The dataset used during the evolution is divided into three parts: train, validation, and test. This allows for getting unbiased results. The training dataset enables to tune the weights of the networks, and the validation dataset evaluates the performance during the evolution of the networks and based on the validation loss the early stopping can be trigger. The test dataset is used, after the evolutionary process, to evaluate the performance of the best models obtained during evolution in order to check the generalization of the networks [38].

In respect to the defined GA structure, the initial population is randomly generated, and for each layer, the parameters are set stochastically. However, the initial number of layers is limited by a low upper bound, such as the following outer level structure: "features":[5,10,15], "classification":[1]. So, the initial networks can have 5, 10, or 15 feature layers and 1 classification layer [9, 39].

During the evolution, the operators and parameters are applied to each candidate solution. To train the networks and obtain the best ones, based on the accuracy, Assunção et al. [9] resort to a backpropagation algorithm and a learning rate. After the evolution, and to get the robustness of the best networks, the authors re-trained the networks for several epochs using the training and validation set. In this training a variate learning rate and also the backpropagation algorithm was used. To improve the accuracy of the best networks data augmentation, like Padding, Random crop, and Horizontal flipping, were applied to the test data. After getting the two fittest networks they can both be

assembled increasing the accuracy of the approach [38, 9].

There are four steps to evaluate a candidate solution. After obtaining the genotype it is necessary to map it into its corresponding phenotype (1) and convert it into a trainable model (2). Afterward, the model is trained (3) and an evaluation is conducted on the model to determine the fitness of the model (4) [9]. To do this, Assunção et al. [9] resort to Keras [1], that runs in Tensorflow [2], and that is an API [3] with GPU support for ANNs.

This approach makes it possible to create CNN without using previous knowledge. It uses novel topologies that were impossible to be thought of by a human being. Moreover, this approach uses limited computational resources and a low number of training epochs during evolution [38].

2.5 Radiomics

Medical imaging can provide valuable information about a given disease. Diagnostic imaging is a non-invasive technique used to extract information about a disease, such as cancer. It can provide a comprehensive macroscopic picture of the tumour's phenotype and its environment. This technique can provide some characteristics, such as the shape, growth, texture and heterogeneity, constituting an alternative to a biopsy [6, 40].

The diagnostic evaluation of clinical images is based on radiologists' expertise, consumes a lot of time, and depends on the institutions' protocol. These limitations produce a low reproducibility in the results and errors. Hence, radiomics emerge to fill in these gaps [3].

Radiomics is a non-invasive method applied to oncological images, that extract radiomic features from medical images. Therefore, a large amount of high-dimensional data is extracted from the clinical images, radiomic features, and they are combined with computational methods, such as ML algorithms, with the aim to develop models to predict clinical outcomes. The main objective of Radiomics is to obtain data with high fidelity and high throughput to provide the patient with valuable diagnostic, prognostic and predictive information [6, 3, 41, 42, 43, 44]. This process relies, firstly, on algorithms based on predefined mathematical transformations that obtain hand-crafted features. In this case, it is necessary to define a Region of Interest (ROI) predefined by an operator, normally a radiologist. However, nowadays, we can resort to deep learning, and extract automatically the features, deep features, taking into account, or not, the ROI [6, 40, 44].
In Radiomics, we can include biological or medical data to support evidence-based clinical decision-making [6, 41, 42].

Several imaging modalities can be used to extract the radiomics' features, the most common being Computed Tomography (CT), Positron Emission Tomography (PET) scans, and Magnetic Resonance Imaging (MRI). CT is used in the diagnosis of many diseases

[1] The most used deep learning framework.
[2] Complete open source platform for machine learning.
[3] Application Programming Interface

and can provide measures, like tissue density. PET can provide measures about metabolic activity and body function. MRI is a method that depends on many factors, such as the gradient, pulse sequence, and magnetic field strength [6, 40].

Radiomics has been successfully applied to cancer diagnosis, tumour detection and classification; survival, malignancy and recurrence predictions; and cancer staging [6].

2.5.1 Hand-Crafted Features

When dealing with hand-crafted features the radiomics workflow can be divided into distinct steps, such as pre-processing (image acquisition and reconstruction), segmentation, feature extraction, feature reduction, and statistical analysis. [6, 40, 3, 41].

Pre-processing

The data quality depends on the acquisition protocols which vary according to each institution's policies. Given this, it is necessary to pre-process the data to reduce noise and artifacts by applying smoothing and enhancement techniques [6, 40].

Segmentation

This is a critical step of the radiomics' process where the ROI is identified. This method should be as automatic as possible, time-efficient, and provide accurate and reproducible boundaries. The conventional segmentation techniques lie within three categories: intensity-based, model-based, and machine learning methods. The deep learning methods can also be used to do segmentation using deep networks. These methods include different variations of the U-Net, "LungNet" architecture, DenseNet, and hybrid dilated convolutions. To evaluate segmentation we can resort to three factors: accuracy, reproducibility, and consistency [6, 3, 42, 43].

Feature extraction

Different types of features can be extracted from the data. These features can be categorized into first order (intensity-based and shape-based features), second-order (texture-based features), and higher-order (wavelet and Fourier features). The first-order features are used to described pixel values without considering their spatial relationship. Second-order features can provide a spatial correlation between pixels of an image [6, 3].

Features reduction

After feature extraction, it is necessary to exclude redundant features to improve the quality of the data (reproducibility, informativeness, and relevancy), reduce their dimensionality, and avoid overfitting. In Radiomics, feature reduction techniques can be divided into two categories, supervised and unsupervised. Supervised methods include filtering, wrapper, and embedded methods, and take into account the discriminative ability of the

features. Unsupervised methods, such as PCA and Independent Component Analysis, are applied to reduce the redundancy of the features. In the embedded methods the feature selection and classification are performed simultaneously [6, 3].

Statistical analysis

The extracted features are then used in statistical analysis to reach a specific task like those mentioned earlier. The most common analysis methods are clustering (Hierarchical and partitional), classification (SVM, NN and k-nearest neighbour) and survival analysis, also called time-related analysis (Kaplan-Meier Survival Curve, Cox Proportional Hazards Model, and Log-Rank Test) [6, 3].

2.5.2 Deep Features

Deep learning models allows the extraction and selection of robust deep features to identify complex patterns automatically. The features can be analyzed in the deep network or go through a different analyzer, such as SVM. Different deep architectures can be used, such as CNN or Auto-Encoders [6, 44].

Some advantages of using DL models in Radiomics are: no need for prior knowledge; automatic feature extraction; and can resort or not to the segmentation of the ROI [6].

The studies concerning DL models have several aspects to take into account [6]:

1. **Input Hierarchy**
   There are three types of input images: Slice-level, volume level, and patient level. At slice-level, the image slices are analyzed and classified independently. At volume-level, all the slices associated with a volume are used as input. At patient-level is used as input all the volumes associated with a patient [6].

2. **Pre-trained and Raw Models**

   - **Training from scratch**, enables the networks to be completely adjusted to the problem. However, due to the limitation of medical datasets this approach can lead to overfitting and class imbalance (unequal number of positive and negative classes). Some strategies can be applied to fight these issues [6]:
     - **Data augmentation** – new data is formed based on the existing data, using random transformations to create more samples [12]. This method was employed in [16] and [45] to augment the training data and prevent overfitting in mammography images applying CNN. The methods implemented were cropping, translation, rotation, flipping, and scaling;
     - **Multitask training** – Simultaneous classification tasks are performed to decrease the number of free parameters [6];
     - **Loss function modification** – The function is modified to give more emphasis to the minority class.

- **Transfer Learning via a Pre-trained Network**, can also solve the problems of class imbalance and inadequate training data. The network is trained using a large image dataset, that transfers the information to the network. Afterward, the trained network is tunned by re-training the network using a medical dataset [6]. Transfer Learning is used in many studies applied to breast cancer, because of the large number of data needed to train a CNN from scratch and the small amount of breast cancer data [17]. However, negative transfer can happened because both the dataset tasks are too dissimilar, resulting in a reduction in the performance [7]. Another problem is because most of the networks used are large and complex, leading to overfitting. Therefore, some changes have to be performed to the network that can be very difficult to implement or even impossible. Some classical CNN architectures such as AlexNet [46], VGG19 [47] and GoogleNet [17], were used to distinguish between benign and malignant breast lesions. Alkhaleefah et al. [17] trained a CNN network from scratch using spine images dataset. Subsequently, the authors used this CNN model and the other two models, AlexNet and GoogleNet, to train on breast cancer data by fine-tuning to classify a breast lesion as benign or malign. The authors concluded that Transfer Learning between two similar CNNs in the domain structure is more effective than those with different domains. Zhang et al. [48] concluded that the InceptionV3 model provides a better classification of breast lesions than the CNN models VGG16, ResNet50, and VGG19.

3. **Deep Learning Network Architectures**
   The deep features can be extracted by discriminative and/or generative deep learning networks. In the second option, the learned weights of the generative model are used as initial weights on a discriminative model [6].

- **Discriminative models** make class distinguishable and reduce the prediction error. In Radiomics the most popular discriminative architectures are CNN and Recurrent Neural Networks (RNN) [6]. The CNN architecture was already explained in section 2.1.1 and RNN architecture is explained in [6].
  Arevalo et al. [49] used DL to extract radiomic features for the first time in mammography for breast cancer, applying CNN with two convolutional layers, two pooling layers, and one fully connected layer. The authors concluded that the use of a CNN, made from scratch, followed by a SVM model, outperforms the hand-crafted radiomics method and a pre-trained CNN model. Zhou et al. [50] also applied a CNN structure, made from scratch, to classify malignant and benign breast tumours. In [47], [46], [17] and [49] CNNs models were used to extract and select the features and, in the end, it was used a SVM classifier model to classify the feature vector extracted from the CNN.

- **Generative models** originate new samples from the distribution data based on the features learned. Some of the generative models used in Radiomics are Auto-Encoder, Deep Belief networks, and Deep Boltzmann machines [6].

Deep Learning models act as a black box. Therefore, it is important to increase the explainability of these models. Some of the techniques used are [6]:

1. Visualization of the features that the network is looking for. "Feature visualization" can be made by features maps;

2. Generating a heat-map where the image regions responsible for the output can be seen. This sensitivity analysis, in CNN, can be made using backpropagation where each input pixel is associated with weight;

3. Projection of the high-dimensional feature space to a bi-dimensional plane.

2.5.3 Hybrid Solution

The hybrid solutions can be acquired in two ways:

1. Combination of Radiomics [6]:

   (a) Extraction from different imaging modalities [51];

   (b) With other data sources: Clinical characteristics (eg. age and gender), blood bio-markers (eg. cholesterol level), prognostic markers(eg. tumour stage and size), and gene expression [6].

2. Fusion of hand-crafted and deep features:
   The fusion of both methods will allow benefiting from both domains' advantages and both types of features, improving the performance [6]. Huynh et al. [46] ensemble pre-trained CNN features and analytical extracted features, that were trained with the SVM classifier, and obtained a better Area Under the Curve (AUC) for the assembled model than the separated models. Antropova et al. [47] also concluded that the fusion between classifier outputs from the pooled CNN features and the handcrafted features outperforms both methods in separate.

CHAPTER 3

RADIOMICS AND DENSER

This chapter details the dataset used and the respective division into train and test to evolve the networks. Furthermore, the experimental setup is presented, and it can be summarised in 3 steps: Pre-Processing, Evolutionary Search and Testing. Two imaging pre-processing studies are presented, which originate the two different image datasets that will be studied later. In the Evolutionary Search we describe the grammar and experimental parameters. Finally, and in order to understand where the models are "looking" in the input image to reach a decision, the model interpretability method implemented is explained.

3.1 Dataset

In our work, we decided to use a dataset containing digital mammography, since Digital Mammography is one of the most popular medical imaging screening techniques to identify breast cancer. Additionally, this dataset is publicly available which will allow us to compare our results with those from the literature.

The dataset used was the Curated Breast Imaging Subset of the DDSM (CBIS-DDSM), from the Digital Database for Screening Mammography (DDSM) with 1,566 actual participants. This dataset can be found in Cancer Imaging Archive [1] and it is described in [10]. The CBIS-DDSM dataset was selected due to its quality and size since it is a well-curated public dataset provided for the mammography community with 2620 scanned film mammography studies. The dataset is divided into calcification (753 images) and mass (891 images) cases, and it includes malignant, benign, and benign without call-back cases (meaning that no additional films or biopsies were done to confirm). Each image in the dataset is in the Digital Imaging and Communications in Medicine (DICOM) format derived from several different scanners at different institutions. Additionally, each

[1] https://wiki.cancerimagingarchive.net/display/Public/CBIS-DDSM

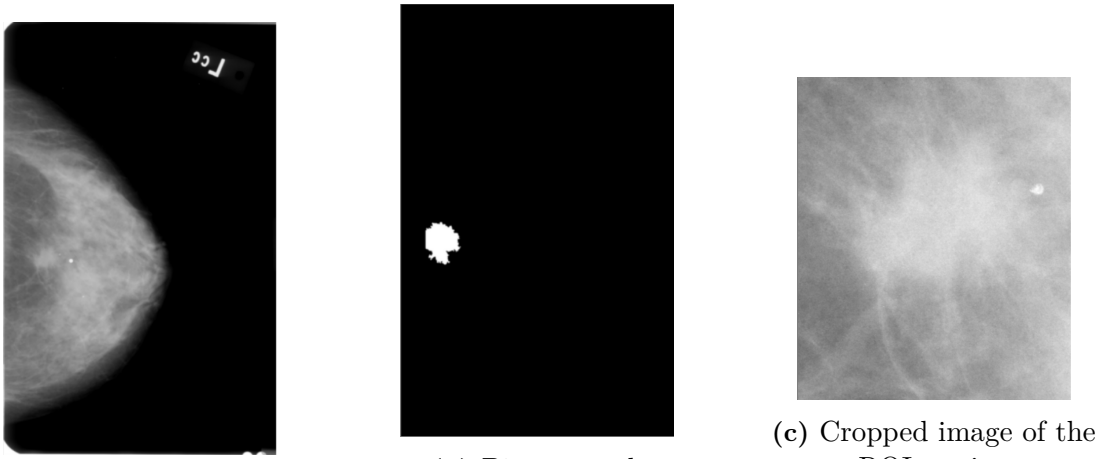
image is accompanied by a CSV file with additional information. The dataset has images obtained in the Craniocaudal (CC) and Mediolateral Oblique (MLO) views which are common for mammography screenings. The images are split into full mammography images, binary mask images delineating the ROI for each abnormality, and cropped images with the abnormalities [10]. Figure 3.1 shows an example of each type of image. In what concerns the information present in the CSVs it is as follows [10]:

- Patient ID: the first 7 characters of images in the case file
- Density category
- Breast: Left or Right
- View: CC or Mediolateral Oblique (MLO)
- Number of abnormalities in the image (This is necessary since there are some cases containing multiple abnormalities)
- Mass shape (when applicable)
- Mass margin (when applicable)
- Calcification type (when applicable)
- Calcification distribution (when applicable)
- BI-RADS assessment
- Pathology: Benign, Benign without call-back, or Malignant
- Subtlety rating: Radiologists' rating of difficulty in viewing the abnormality in the image
- Path to image files

In our experiments, we only use the mass cases cropped images, which were treated in a slice-level. The use of the cropped images speeds up the training since it catches the information about the masses. Our goal is to classify the lesion into benign and malignant. The benign without call back cases were considered as benign cases, so the study is a binary classification problem [10].

Through the Pathology information contained in the CSV file it is possible to obtain the target information, crucial to build the models. The target information was obtained by matching the information incorporated in the image's name, and the 'Patient ID', 'Breast', 'View' and 'Number of abnormality' informations in the CSV file. Subsequently, the information was converted into a binary classification, namely 1 to malignant and 0 to benign.

The dataset was split into 20% for testing and 80% for training ensuring that they were balanced. Since each abnormality can be in the CC or/and MLO view, in the end, the dataset is composed of 1318 images for training (681 benign and 637 malignant) and 378 images for testing (231 benign and 147 malignant) as shown in Figure 3.2a histogram. Figure 3.2b exhibits the images' distribution of the cases by views. Therefore, the MLO view is composed of 711 images to train (370 benign and 341 malignant cases) and 201 images for testing (121 benign and 80 malignant). The CC view has 607 images to train (311 benign and 296 malignant) and 177 images for testing (110 benign and 67 malignant).

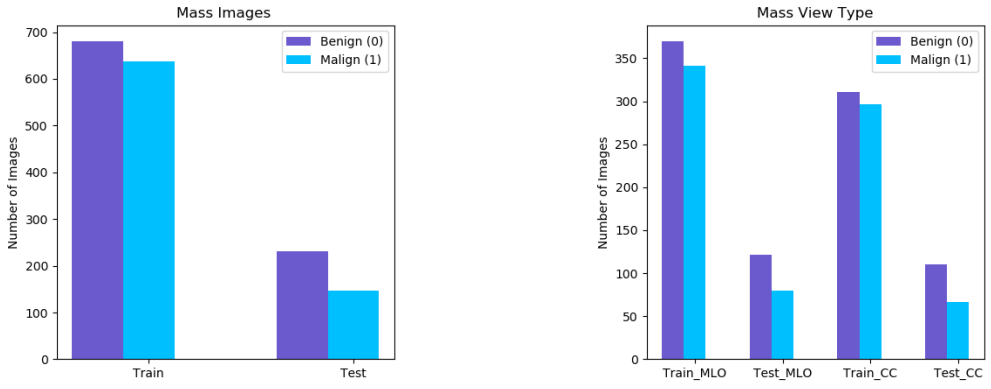


(a) Full mammography image

(b) Binary mask delineating the ROI of the abnormality

(c) Cropped image of the ROI section

Figure 3.1: Types of images provided by the CBIS-DDSM dataset. Images of CC view of the Patient '00001' left breast.



(a) **(b)**

Figure 3.2: Histograms of the images' distribution for the malignant and benign cases in the train and test sets (a) and the same distribution but for each view (CC and MLO) (b).

3.2 Experimental Setup

DENSER's goal is to evolve CNNs that give the best performance in the classification of benign and malignant mass cancer lesions.
Taking into consideration the DENSER's explanation presented in Section 2.4, and to better understand the process behind our approach, the scheme depicted in Figure 3.3 was constructed.

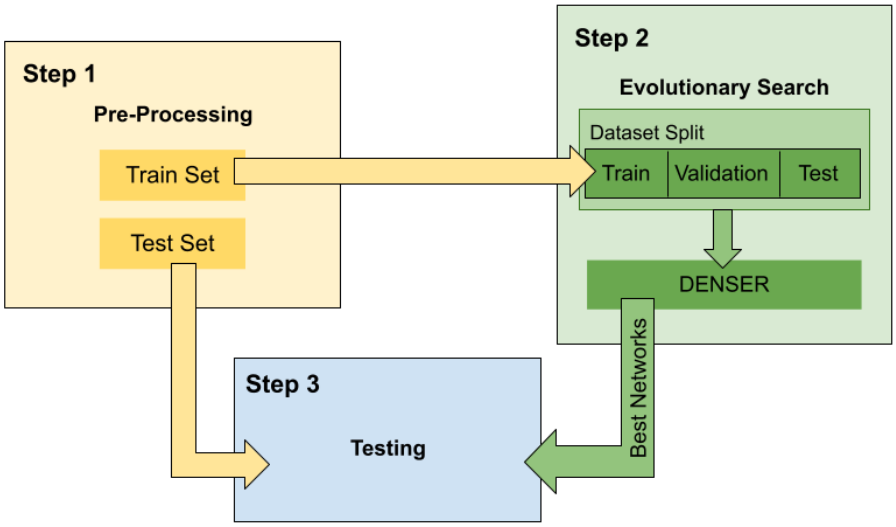


Figure 3.3: Summary scheme of our approach. It consists of 3 steps: Pre-Processing, Evolutionary Search and Testing.

The first step corresponds to acquiring the dataset, pre-process it and divided it into train (80%) and test(20%).

In step 2 we start by splitting the train dataset again into 70% for training, i.e., to tune the weights of the networks, 15% for validation, i.e., to evaluate the performance during the evolution of the networks. The remaining 15% of the data was used for testing the evolved networks. After splitting the dataset, the evolutionary search will be applied to find the best networks. This search is conducted by DENSER, with the grammar and evolution parameters described below.

Finally, in Step 3, we use the test set obtain in step 1 and the best networks of step 2 to assess the quality of the results. In this stage, we evaluate the generalisation ability of the models. The evaluation of the network will be set up using the following metrics [13]:

- Confusion Matrix - Table that presents how successful the model is in predicting the cases for each class. The benign class corresponds to the negative class (N) and the malignant class is the positive one (P). Therefore, each class can have false (F) and true(T) predictions;

True Classes	Predicted Classes: Benign	Predicted Classes: Malignant
Benign	True Negative (TN)	False Positive (FP)
Malignant	False Negative (FN)	True Positive (TP)

Table 3.1: Confusion matrix

- Accuracy - The ratio between the number of correct predictions and the number of all the predictions;

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{3.1}$$

- Precision - The ratio between the TP cases and the overall positive predictions;

$$Precision = \frac{TP}{TP + FP} \tag{3.2}$$

- Recall - The ratio between the TP cases and the overall positive cases in the test set;

$$Recall = \frac{TP}{TP + FN} \tag{3.3}$$

- Receiver Operating characteristic (ROC) Curve - Presents a summary graph of the classification performance, taking into account the True Positive Rate (TPR) and False Positive Rate (FPR) values;

$$TPR = \frac{TP}{TP + FN} \text{ and } FPR = \frac{FP}{FP + TN} \tag{3.4}$$

- F1-score - The F1-score value is the harmonic mean between the precision and recall;

$$F1 = \frac{2 \times precision \times recall}{precision + recall} \tag{3.5}$$

3.2.1 Image Pre-Processing

Before using the dataset images, we need to pre-process them. To do this, we used a set of Python libraries namely, such as Pydicom [2], SimpleITK [3], Python Imaging Library (PIL) [4] and Numpy [5].

The first issue we had to deal with was related to the fact that in the dataset the cropped images have different sizes. The height range was between 1099 to 126 pixels, and the width range was between 1089 to 95 pixels. To tackle this we employed, up or down

[2] https://pydicom.github.io/
[3] https://simpleitk.org/
[4] https://pillow.readthedocs.io/en/stable/
[5] https://numpy.org/

sampling. We acknowledge that upsampling will set unreal information in the images and downsampling will give very narrow images with loss of information, so both methods can impact the learning of the networks. Nevertheless, in medical imaging the introduction of synthetic information, by applying upsampling, has a more negative impact on the final results. Another aspect to take into account is the adoption of the original colour code, which in this case is the grayscale. The images are 16-bit grayscale with an intensity window of [0, 65535], making the image information very complex. To deal with this, the images were downsampled to 8-bit grayscale images.

Lastly, we decided to measure the impact of two imaging pre-processing methods: the use of the grayscale images employing downsampling (90x90 grayscale images) and grayscale images with more information about the surroundings implementing downsampling (250x250 grayscale images).

90x90 Grayscale Images

The images were resized to 90X90 pixels focusing on the lesion region. An image example is shown in Figure 3.4.

The pixel values were normalized to the range of [0, 1] and converted into a matrix format. In the end, the images will have 90 columns and lines, and since the images are grayscale they will have just one colour channel.

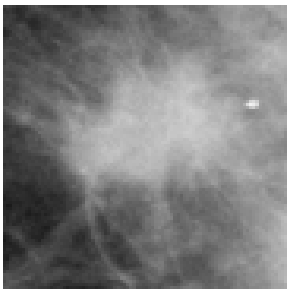


Figure 3.4: Cropped images of CC view of Patient '00001' left breast with 90x90 pixels.

250x250 Grayscale Images with more surrounding information

When using medical imaging the information surrounding the region of the lesion, such as the margins, forms, and structures are essential for the mammography diagnosis. Therefore, a new batch of cropped images was created where this information was included. The new cropped grayscale images were created based on the binary mask images and on the cropped images limiting box dimensions. The middle point coordinates of the lesion was obtained through the binary mask images, and the maximum bounding box width and height was defined considering all the cropped images. After getting the maximum values, 100 pixels were added to each measure, in order to get more information about the surrounding structure. However, some times these measures surpass the margins of the images. In these cases, the number of pixels that go beyond the margin was added to the

opposite side. Given that the original size was computationally expensive, and the images were resized to 250X250 pixels, which is approximately 25% of the original images' size. The colour of each image's pixel was normalized to the range [0, 1] and converted into a matrix format. In this case, the images will have 250 columns and lines, and since the images are grayscale they have just one channel. This process is explained in the scheme depicted in Figure 3.5. An example of the obtained image is showed in Figure 3.6.

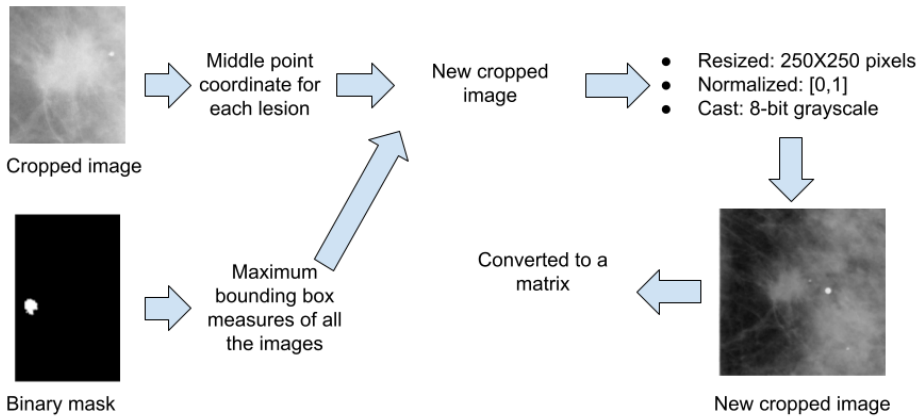


Figure 3.5: Scheme of the implemented pre-process work to acquire the new cropped images with more surrounding information.

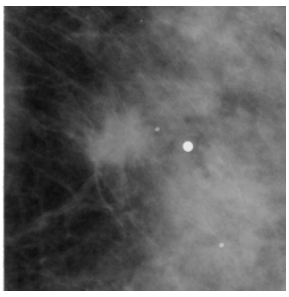


Figure 3.6: New cropped images of CC view of Patient '00001' left breast with 250X250 pixels.

3.2.2 Grammar

The grammar is a central component in DENSER as it allows the definition of the building blocks of our models, i.e., the layers, the learning algorithms and its hyper-parameters. The grammar used for the evolution of learning and topology is depicted in Grammar 3.1. The grammar structure and hyper-parameters are explained in Annex A.1 [9].

```
<features> ::= <convolution> | <convolution> | <pooling> | <pooling> | <dropout> | <batch-norm>
<convolution> ::= layer:conv [num-filters,int,1,32,256] [filter-shape,int,1,2,5] [stride,int,1,1,3] <padding> <activation-function> <bias>
<batch-norm> ::= layer:batch-norm
<pooling> ::= <pool-type> [kernel-size,int,1,2,5] [stride,int,1,1,3] <padding>
<pool-type> ::= layer:pool-avg | layer:pool-max
<padding> ::= padding:same | padding:valid
<dropout> ::= layer:dropout [rate,float,1,0,0.7]
<classification> ::= <fully-connected> | <dropout>
<fully-connected> ::= layer:fc <activation-function> [num-units,int,1,128,2048] <bias>
<activation-function> ::= act:linear | act:relu | act:sigmoid
<bias> ::= bias:True | bias:False
<softmax> ::= layer:fc act:softmax num-units:2 bias:True
<learning> ::= <gradient-descent> <early-stop> [batch_size,int,1,50,500] epochs:10000 | <rmsprop> <early-stop> [batch_size,int,1,50,500] epochs:10000 | <adam> <early-stop> [batch_size,int,1,50,500] epochs:10000
<gradient-descent> ::= learning:gradient-descent [lr,float,1,0.0001,0.1] [momentum,float,1,0.68,0.99] [decay,float,1,0.000001,0.001] <nesterov>
<nesterov> ::= nesterov:True | nesterov:False
<adam> ::= learning:adam [lr,float,1,0.0001,0.1] [beta1,float,1,0.5,1] [beta2,float,1,0.5,1] [decay,float,1,0.000001,0.001]
<amsgrad> ::= amsgrad:True | amsgrad:False
<rmsprop> ::= learning:rmsprop [lr,float,1,0.0001,0.1] [rho,float,1,0.5,1] [decay,float,1,0.000001,0.001]
<early-stop> ::= [early_stop,int,1,5,20]
```

Grammar 3.1: Grammar used for the evolution of the learning and topology. The grammar structure and hyper-parameters are explained in Annex A.1 [9].

3.2.3 Experimental Parameters

The evolutionary parameters are divided into 4 categories [9]:

1. Evolutionary Engine - related to Evolutionary Algorithm used by DENSER;
2. Dataset - partitioning of the dataset;
3. Training - associated with the backpropagation algorithm;
4. Data Augmentation - processes to generate more data;

The evolutionary parameters explanation is in Annex A.2 and the employed values for each study is in Table 3.2. In our approach, the learning parameters, e.g., batch size and learning rate, are not fixed and they change through evolution.

To construct the networks, the outer level structure used for the hidden layers was [["features", 1, 10], ["classification", 1, 10]], i.e., we can have between 1 to 10 layers of feature extraction and 1 to 10 layers of classification. These layers are defined in Grammar 3.1 in the 1 and 8 lines, respectively. The initialisation is of the form "features":[5,10,15], "classification":[1], i.e., the initial network are composed by 5, 10 or 15 feature extraction layers and 1 classification layer. The output layer was always a softmax layer.
Since the classes are balanced, the balanced accuracy was used as a fitness metric and due to the stochastic nature of the DENSER framework, we executed each experiment 4 times. Each evolutionary process took approximately 1 to 2 weeks to terminate, using a GPU NVIDIA 1080 Ti.

Category	Parameter	Section 4.1.1	Section 4.1.2	Section 4.1.3	Section 4.2.1	Section 4.2.2	Section 4.2.3
Evolutionary Engine	Number of runs	4					
	Number of generations	300	150	300	150	300	
	Population size	20					
	Add layer rate	0.25					
	Reuse layer rate	0.15					
	Remove layer rate	0.25					
	Add connection rate	-					
	Remove connection rate	-					
	DSGE-level rate	0.15					
Dataset	Training set	75%					
	Validation set	15%					
	Test set	15%					
Training	Maximum Number of epochs	10000					
	Maximum Training time (segundos)	18000					
	Loss	Categorical Cross-Entropy					
	Batch size	*					
	Learning rate	*					
	Momentum	0.99					
Data Augmentation	Rotation range	-	190º		-		190º
	flip	-	horizontal		-		horizontal

Table 3.2: Table with the evolutionary parameters of each study of Sections 4.1 and 4.2. The parameters specification and explanation is displayed in Appendix A.2

3.2.4 Model Interpretability

One problem associated with deep learning models is interpretability since they are "black boxes". However, it is important to understand how the models reach a decision, i.e., where the model is "looking" in the input images. One way to achieve this goal is to use *saliency maps*, which plots the gradient of the predictions of the model taking into account the input. Implementing this visualization tool allows knowing how much a pixel contributes to class prediction [52].

This was implemented resorting to the visualize_saliency in the Keras Visualization Toolkit (Keras-vis)[6]. In Figure 3.7 we show a schematic explanation of this approach. The visualize_saliency function has 4 inputs: Images (Seed Input) and Pathology information (Filter Indices) of the test set, the model to obtain the prediction and to get the information of the last layer (layer_idx). Afterward, the attention heatmap is acquired taking into account these 4 parameters, and 2 processes are administered to improve the visualization: Gaussian filter with a standard deviation (sigma) of 5 and a 'jet' Colour map with a blending value (alpha) of 0.7.

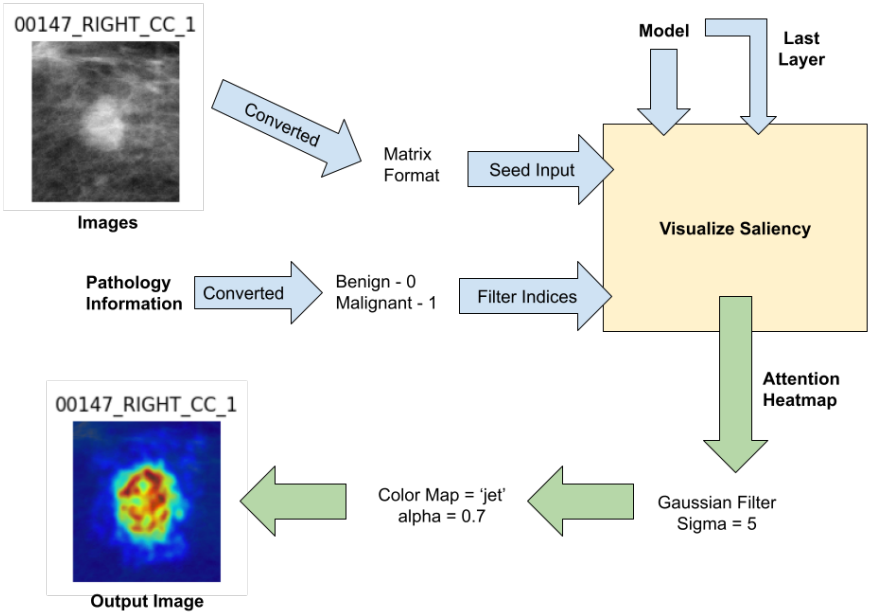


Figure 3.7: Schematic explanation of "visualize_saliency" implementation. It required 4 inputs: images (Seed Input) and Pathology information (Filter Indices) of the test set, the prediction made by the model and the last layer of the model. The output is an attention heatmap. Afterwards 2 processes are administered to improve the visualization: Gaussian filter with a standard deviation (sigma) of 5 and a 'jet' Colour map with a blending value (alpha) of 0.7.

[6]https://raghakot.github.io/keras-vis/vis.visualization/#visualize_saliency

CHAPTER 4

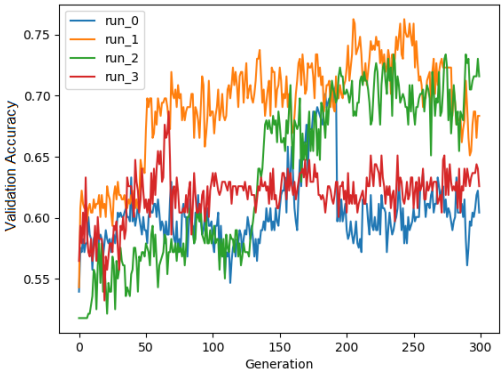
EXPERIMENTAL RESULTS

This chapter presents the experimental results of the two experiments conducted using the 90x90 and the 250x250 grayscale images. Each experiment contains three analysis, in which we investigate the impact of the number of generations and the application or not of data augmentation approaches. We also present the results of the model interpretability for the best networks, and a study where we build an ensemble of the 2, 3, and 4 fittest networks. Finally, we compare the results of our approach with the ones presented in the literature for the same dataset.

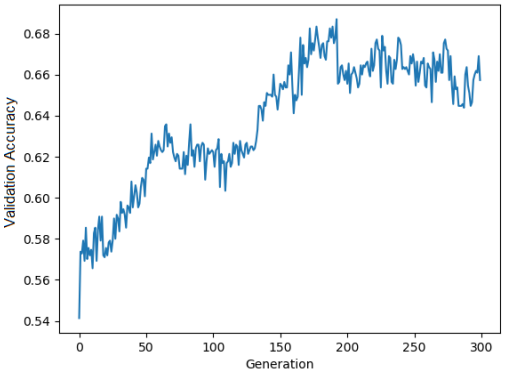
4.1 90x90 Grayscale Images

4.1.1 Evolution Analysis for 300 generations with no Data Augmentation

The first study conducted with the 90x90 images dataset was evolved over 300 generations.
The evolution of the validation accuracy over generations for each run is displayed in Figure 4.1a. The maximum accuracy values for run 0, 1, 2 and 3 were 70.86%, 76.26%, 73.38% and 68.70% as the Table 4.1 shows. The minimum value of all runs was 51.80% in generation 0 of run_2, and the maximum accuracy value was 76.26% in generation 205 of run_1.
Figure 4.1b exhibits the mean accuracy of all runs. Looking at the results it is possible to see that the models are evolving, i.e., their quality is improving, through the generations, especially until generation 192. The models in the first generation start with a mean minimum accuracy value of 54.14% and reach the maximum of 68.70% in generation 192.
The ups and downs present in both graphs are due to the re-evaluation of the networks.



(a) Graphic of the accuracy value of the validation set in each generation for each run during the evolution of the networks.



(b) Graphic of the mean accuracy values of all runs in graphic 4.1a.

Figure 4.1: Evolution of the runs over 300 generations taking as input 90x90 cropped images.

After the evolutionary search, we collect the best network evolved by DENSER in each run and measure its generalisation ability by applying it to the test set. The results of test accuracy are displayed in Table 4.1. Looking at the Table, one can see that two networks achieved a test accuracy value of 69.58%, (run_0 and run_2 networks). Even though the number of runs is small, it is possible to see that the performance exhibited by the networks in the validation set resembles the one in the test set (with the exception of the network from run_1). These result show that DENSER is able to evolve models that can capture the patterns that allow them to work beyond the training data.

Best Networks found in:	Validation Accuracy (%)	Test Accuracy (%)
run_0	70.86	69.58
run_1	76.26	68.25
run_2	73.38	69.58
run_3	68.70	59.52

Table 4.1: Accuracy results for the best networks found by DENSER for each run in the 300 generations study using the 90x90 cropped images test set.

In figure B.1 of Annex B.1.1 we show the topology of the best network found in the run_1. This network is composed of thirteen hidden-layers, one input and output layer, and 2319176 parameters. It has four convolution layers with filter sizes of 40, 253 and 114, kernel sizes of 5x5 and 4x4, stride sizes of 1x1 and 3x3, and sigmoid and linear as activation functions. The network has three max-pooling layers with 3x3 and 5x5 pool sizes, and 1x1 and 3x3 stride sizes. It is also composed of two batch normalization layers with a momentum of 0.99, three dropout layers with a rate of 0.36 and 0.0, and a dense layer with softmax as an activation function.

In general, the topology of the network does not differ from a typical CNN network. However, it has three dropouts in a row. The use of these three dropouts in a row is may be due to the fact that the dataset is small which can more easily lead to overfitting. Therefore, having three Dropouts can help to deal with this problem. Another curious aspect is the implementation of a rate of 0.0 in two of the dropout layers, which means that there is no dropout, i.e., there is no regularization. One possible explanation, and taking into account that we are dealing with an evolutionary process and these two layers do not have an impact on the network's performance but increase the its size, is that the network presents bloating. Bloat is defined by an increase of the network size without an increase in fitness. The appearance of the bloat is might be due to the fact that the small networks are much more susceptible to a decrease in quality when a mutation occurs. As such, by increasing the size of the individual with information that does not have an impact on fitness, the networks are protecting themselves against the effects of a possible bad mutation [53].

To better understand the classifications made by the best network, we use the confusion matrix of Table 4.2, the ROC curve in Figure 4.2 and the classification report in Table 4.3, which include the precision, recall and F1-score for each class.

		Predicted Classes	
		Benign	Malignant
True Classes	Benign	171	60
	Malignant	60	87

Table 4.2: Confusion matrix of the best evolve network in the 300 generations study using the 90x90 cropped images test set.

	Benign (%)	Malignant (%)
Precision	74	59
Recall	74	59
F1-score	74	59

Table 4.3: Classification report applying the 90x90 cropped images test set in the best network found by DENSER in the 300 generations study. The classification report presents the values of precision, recall and F1-score for each class.

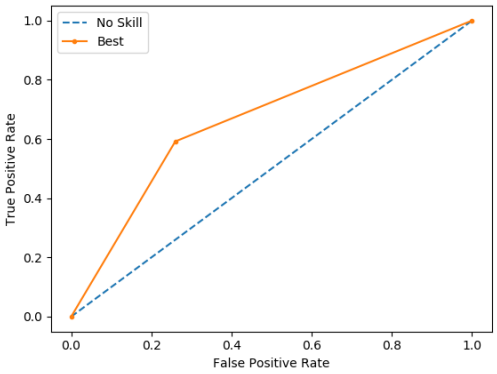


Figure 4.2: ROC curve of the best network acquired by DENSER in the 300 generations study applying the 90x90 cropped images test set.

In the ROC curve the TPR and FPR values are 0.59 and 0.26. Both values are poor, showing low sensitivity and specificity of the model towards malignant cases. Thus, the model does not reliably predict the malignant cases.

The precision values of 74% and 59% for the benign and malignant classes allow us to conclude that the predictions made by the model for the benign class are more reliable than the predictions made for the malignant class. The recall values for the benign and malignant classes, 74% and 59%, respectively, confirm that the model detects the benign cases better than the malignant ones. These results indicate that, 26% of the benign and 41% of the malignant cases were misclassified. However, and taking into account the nature of our problem, this is unfavourable, because the priority is to increase the detection of malignant cases, i.e., decrease the number of FN cases. Therefore, one approach that can be implemented to tackle this problem is data augmentation, since it will introduce new training samples in order to increase the generality of the model.

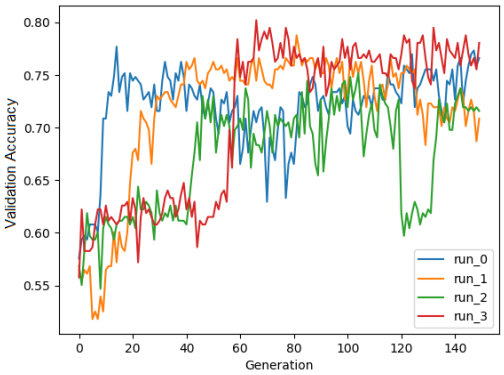
4.1.2 Evolution Analysis for 150 generations with Data Augmentation

One way to prevent overfitting, as mentioned in Section 2.5.2, is by implementing data augmentation. Data augmentation will allow the model to use more data during the evolutionary process. Two spatial augmentation operations were implemented [1]:

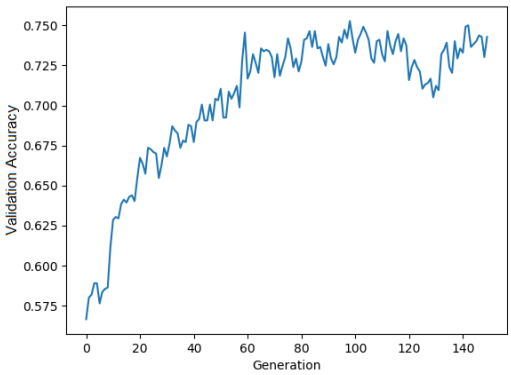
- Zoom range: 2 - Randomly zooms the image, adding pixels around the image or interpolating pixels values. The percentage of the zoom in this case is 0% to 300%, since the range is define as [lower, upper] = [1-zoom_range, 1+zoom_range] → [0,3].
- Rotation range: 90° - The image's pixels are randomly rotated 90° degrees clockwise. Some pixels will move outside the image, resulting in images with pixels with no data which are filled in by the nearest pixel values.

[1] https://keras.io/api/preprocessing/image/

Since we increase the size of the training dataset, we started by evaluating the performance with a reduce number of generations, 150, as showed in Figure 4.3. Figure 4.3a displays the maximum validation accuracy for each generation for each run. A brief perusal of the image show that the minimum validation accuracy value is 51.80% in run_1 generation 5, and the maximum value is 80.22% in run_3 generation 66. Figure 4.3b presents the mean accuracy of all runs over the generations, and the networks evolution is evident since the accuracy value is increasing along the generations. The minimum and maximum mean accuracy values are 56.65% and 75.27%.



(a) Graphic of the accuracy value of the validation set in each generation for each run during the evolution of the networks.



(b) Graphic of the mean accuracy values of all runs in graphic. 4.3a.

Figure 4.3: Evolution of the runs over 150 generations taking as input 90x90 cropped images with data augmentation.

The best networks of each run are the networks with the higher validation accuracy values. These networks were collected and used in the test set, containing a total of 378 new images. The test accuracy values for this study are displayed in Table 4.4, and it is noticeable a loss in the performance. Both run_1 and run_2 present the best test accuracy value of 71.16%. The network with the best validation accuracy value, the run_3 network, presents a test accuracy value of 68.52%, which is 2.64% below the best test accuracy value.

Best Networks found in:	Validation Accuracy (%)	Test Accuracy (%)
run_0	77.70	67.72
run_1	78.80	71.16
run_2	75.18	71.16
run_3	80.22	68.52

Table 4.4: Accuracy results for the best networks found by DENSER for each run in the 150 generations with data augmentation study using the 90x90 cropped images test set.

Figure B.2 in Annex B.1.2 shows the best network topology evolved in this experiment. The network has twelve hidden-layers, one input and output layer (softmax layer) and 2907498 parameters. Two of the hidden layers are convolution layers, with filter sizes of 139 and 41, and a relu function as the activation function. The kernel sizes were 4x4 and 6x6, and the stride size was 2x2. The network also had one max-pooling layer with a pool size of 4x4 and a stride size of 3x3, one average-pooling layer with a pool size of 2x2 and a stride size of 3x3, two batch normalization layers with a momentum of 0.99, three fully connected layers with sigmoid and relu activation functions and three dropout layers with rates of 0.0, 0.5 and 0.27.

Looking at the topology of the network it is possible to see that it does not have a typical CNN structure, since it has one batch normalization layer immediately after the input layer, three dropout layers in a row and a higher number of dense layers.

The application of a batch normalization at the beginning of the network is not a typical approach in the CNN since it will re-scale the input, modifying the initial information [13]. However, it is also commonly used to help overcome overfitting and speed up learning, as mentioned in Section 2.1.1. So, since this topology is the best network it can be seen as a standalone contribution and additional experiments are required.

The higher number of dense layers can also be observed in [9], where Assunção et al. trained DENSER with the CIFAR-10 dataset.

The best network was evaluated on the confusion matrix (Table 4.5), the ROC curve (Figure 4.4), and the precision, recall and F1-score values for each class (Table 4.6).

		Predicted Classes	
		Benign	Malignant
True Classes	Benign	161	70
	Malignant	49	98

Table 4.5: Confusion matrix of the best network acquired by DENSER in the 150 generations applying data augmentation study with 90x90 cropped images test set.

	Benign (%)	Malignant (%)
Precision	77	58
Recall	70	67
F1-score	73	62

Table 4.6: Classification report applying the 90x90 images test set in the best evolved network in the 150 generations applying data augmentation study. The classification report portrays the values of precision, recall and F1-score for each class.

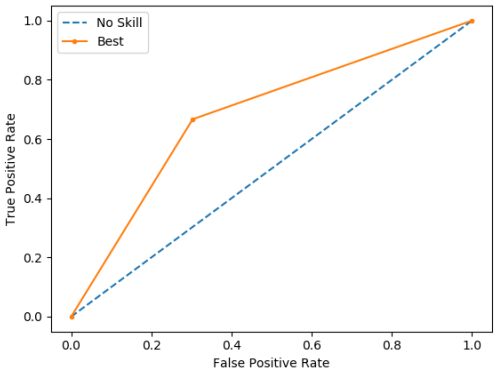


Figure 4.4: ROC curve of the best network acquired by DENSER in the 150 generations applying data augmentation study with 90x90 cropped images test set.

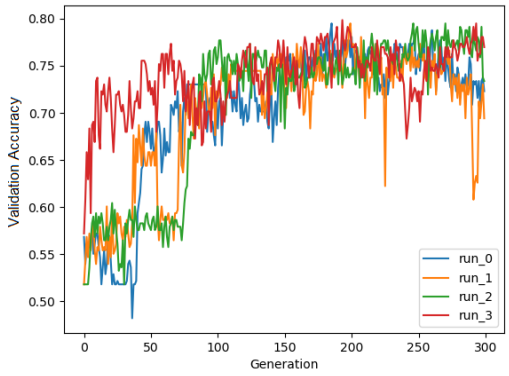
Looking at the results of ROC curve, we obtained TPR and the FPR values of 0.67 and 0.30, respectively.

Through the precision value achieved, it can be concluded that 77% and 58% of the predictions for the benign and malignant cases were true. Showing that the model has an a higher reliability level for benign cases than for malignant ones. The recall values for the benign class was 70% and for the malignant was 67%, which is in accordance with the higher effort made by the model in the prediction of the benign class rather than the malignant. Hence, and taking into account the 321 benign cases, 70 were classified as malignant. For the 147 malignant cases, 49 were classified as benign.

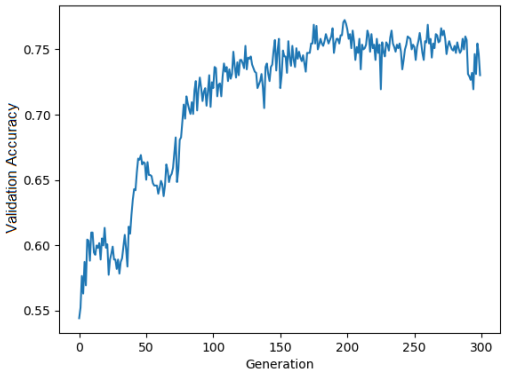
Comparing this study with the previous one without data augmentation, it can be concluded that the introduction of this approach reduces, in general, the number of FN cases, since the model has 11 fewer misclassified cases and a higher TPR value. However, the FP and the FPR increased. In medical imaging, both the FN and FP cases must be avoided as much as possible. Nevertheless, having a malign lesion classified as benign can cause a worse after-effect. Hence, and taking into account the problem we are working with, the reduction of FN is preferable making this model the best choice when compared to the previous one. The test accuracy was also slightly superior to the previous study. Therefore, a study for a larger number of generations will be performed.

4.1.3 Evolution Analysis for 300 generations with Data Augmentation

In this experiment we increased the number of generations from 150 to 300, as shown in Figure 4.5. As expected, increasing the number of generations resulted in networks that obtain a better performance. In Figure 4.5b it can be observed the networks evolution through the generations. The maximum mean accuracy value was 77.25% in generation 198 and from generation 160 to 300 it is visible that the accuracy values increase, even though at a slower rate.



(a) Graphic of the accuracy value of the validation set in each generation for each run during the evolution of the networks.



(b) Graphic of the mean accuracy values of all runs in graphic. 4.5a.

Figure 4.5: Evolution of the runs over 300 generations taking as input 90x90 cropped images with data augmentation.

Table 4.7 presents the validation and test accuracy values for the best network of each run. The test accuracy values were acquired using the test set, which is composed of 378 new images. The network of run_3 has the highest validation and test accuracy values. However, comparing the validation and test accuracy values is noticeable a slight degradation in the performance.

Best Networks found in:	Validation Accuracy (%)	Test Accuracy (%)
run_0	79.50	71.96
run_1	79.50	73.28
run_2	79.50	74.07
run_3	79.85	73.81

Table 4.7: Accuracy results for the best networks found by DENSER for each run in the 300 generations with data augmentation study using the 90x90 cropped images test set.

The best network topology is displayed in Figure B.3 in Annex B.1.3. The network has thirteen hidden layers, one input layer, one output layer (Dense layer) and 2808147 parameters. The network has five convolution layers with filter sizes of 202, 118 and 103, kernel sizes of 6x6, 4x4 and 3x3, stride sizes of 2x2 and 1x1, and linear and relu as activation functions. The network is also composed of two batch normalization layers with a 0.99 momentum, one max-pooling layer with a pool and stride size of 3x3, one dropout layer with a rate of 0.40 and four dense layers with linear, sigmoid, and softmax as activation functions.

Comparing to previous studies, this network has practically the same number of layers. However, it has a lower number of parameters than the one obtained in Section 4.1.2. This reveals that evolving for a larger number of generations might result in smaller but accurate networks.

To further evaluate the best network, the ROC curve was collected , which is depicted in Figure 4.6; the confusion matrix values exhibit in Table 4.8; and the precision, recall and F1-score values of each class, are displayed in Table 4.9.

		Predicted Classes	
		Benign	Malignant
True Classes	Benign	189	42
	Malignant	57	90

Table 4.8: Confusion matrix of the best evolved network in the 300 generations applying data augmentation study with 90x90 cropped images test set.

	Benign (%)	Malignant (%)
Precision	77	68
Recall	82	61
F1-score	79	65

Table 4.9: Classification report applying the 90x90 cropped images test set in the best network found by DENSER in the 300 generations applying data augmentation study . The classification report portrays the values of precision, recall and F1-score for each class.

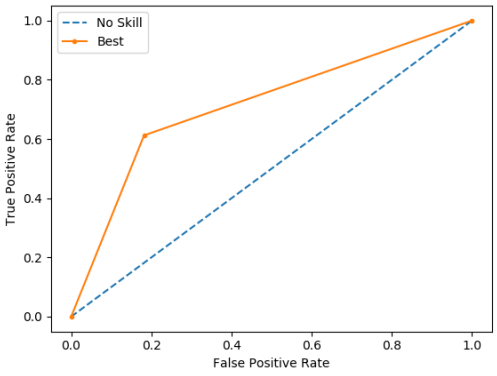


Figure 4.6: ROC curve of the best network acquired by DENSER in the 300 generations applying data augmentation study with 90x90 cropped images test set.

Looking at Figure 4.6, its possible to see that the TPR and FPR values are 0.61 and 0.18. Hence, applying a large number of generations originates a reduction in the FPR value. Confirmed by the confusion matrix in Table 4.8 which shows a reduced number of false positives.

This network acquired the best precision and recall values for the benign class of 77% and 82%, respectively, and the best precision value for the malignant class of 68%. The recall value for the malignant class was lower, comparing to the value of the previous experiment in Section 4.1.2. Furthermore, this network achieved more reliable results, and it got better prediction ability for both the classes, comparing to the two previous studies.

Moreover, and since this was the best network obtained, an additional validation mechanism was implemented to better understand which pixels the model is "looking" for to make a particular prediction, resorting to saliency maps.

Annex C reports the saliency maps of 50 images of the test set taking into consideration the best network of this study. Figure 4.7 presents two different images, one without the black background and the other with it, to detect the impact of the black background in the predictions. Since the images of this dataset are essentially based in the lesion region, it is expected that almost all of the image is used to generate the predictions, as the saliency of image a) confirms. The b) image, which has the black background, reveals that the model is not dismissing the background which might be an indication that our model still needs more training to avoid this element.

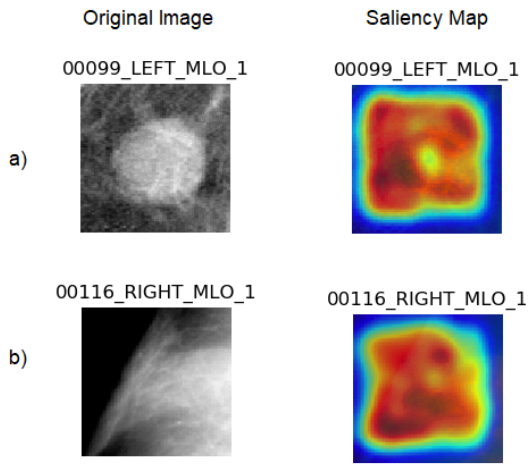


Figure 4.7: Saliency maps of two distinct images (a and b): CC view of patients '00099' and '00116' right breast. The first image is the cropped image and the second image is the saliency map resorting to the best network acquired by DENSER in the 300 generations applying data augmentation study.

4.1.4 Ensembling of the Best Networks

Deep learning models are nonlinear and can have high variance. One way to reduce the variance is by training a set of models and combine the predictions of those models, i.e., implementing an ensemble to improve the predictions by using majority voting [54].

We implemented an ensemble taking into consideration the 4 fittest networks evolved. The set of networks include the best network of Section 4.1.2 (Network A), the run_3 (Network B), run_2 (Network C) and run_1 (Network D) networks of Section 4.1.3.
Three types of ensembles were implemented, as Table 4.10 shows, namely the ensembling of the 4, 3 and 2 fittest networks. The ensembling of the 4 fittest networks achieved the best results, with a test accuracy value of 76.19%.
The ensemble of the 4 fittest networks was validated by the confusion matrix, the ROC curve and the classification report (precision, recall and F1-score values for each class) of Table 4.11, Table 4.12 and Figure 4.8 display, respectively.

Network	Single Network Accuracy Validation (%)	Ensemble Network Accuracy Test (%)		
A	80.22	76.19	75.92	73.54
B	79.85			
C	79.50			-
D	79.50		-	

Table 4.10: Test accuracy considering the ensembling of the 4, 3 and 2 fittest networks of Section 4.1. These networks are: the best network of Section 4.1.2 (Network A), the best network (Network B), run_2 (Network C) and run_1 (Network C) networks of Section 4.1.3.

		Predicted Classes	
		Benign	Malignant
True Classes	Benign	193	38
	Malignant	52	95

Table 4.11: Confusion matrix of the ensembling of the 4 fittest networks of Section 4.1 using the 90x90 new cropped images test set.

	Benign (%)	Malignant (%)
Precision	79	71
Recall	84	65
F1-score	81	68

Table 4.12: Classification report applying the 90x90 new cropped images test set in the ensembling of the 4 fittest networks of Section 4.1. The classification report presents the precision, recall and F1-score values for each class.

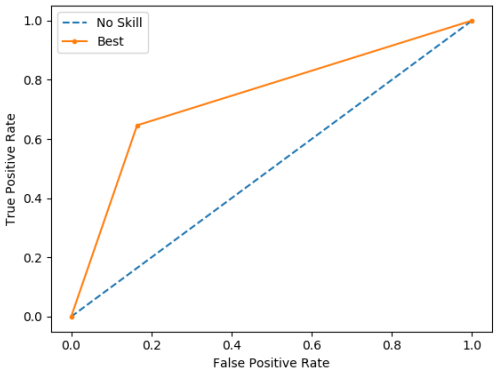


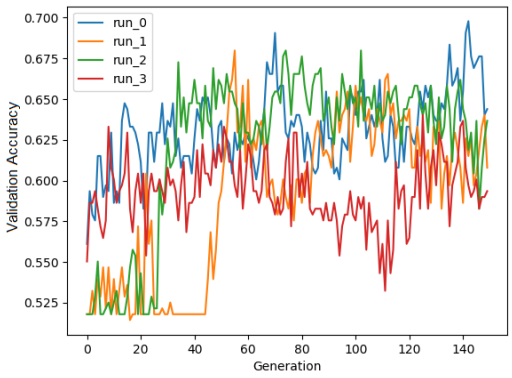
Figure 4.8: ROC curve of the ensembling of the 4 fittest networks of Section 4.1.3 using the 90x90 new cropped images test set.

The TPR was 0.65 and FPR was 0.16 which results in the lowest FPR value for this experiment. The number of FN cases is also closer to the best value obtained in Section 4.1.2.
The precision and recall values for the benign class and the precision value for the malignant class were also the best values obtained until now. In conclusion, using an ensemble increases the performance, improves the predicting power and decreases the misclassification cases.

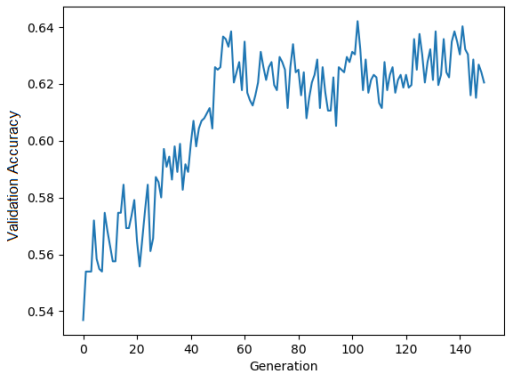
4.2 250x250 Grayscale Images with more surrounding information

4.2.1 Evolution Analysis for 150 generations

In the first set of experiments with 250x250 grayscale images we evolved networks for 150 generations. The evolution analysis can be observed in Figure 4.9a where the validation accuracy for each generation and run is plotted. Looking at the results it is possible to see that run_1 achieves the lowest value of 51.44% in generation 16 and run_0 achieves the maximum value of 69.78% obtained in generation 142. Figure 4.9b shows the mean accuracy of all runs of Figure 4.9a for each generation. In this Figure, it can be observed that there is an increase in performance up to generation 55 and after that, the accuracy values stabilize between 60.52% and 64.21%. The minimum and maximum values obtained were 53.69% and 64.21%, respectively.



(a) Graphic of the accuracy value of the validation set in each generation for each run during the evolution of the networks.



(b) Graphic of the mean accuracy values of all runs in graphic 4.9a.

Figure 4.9: Evolution of the runs over 150 generations taking as input 250x250 new cropped images.

After the evolutionary process finishes, the best network found in each run was applied to the test set. The test accuracy values obtained are in Table 4.13. These results show that run_0 and run_1 present a higher discrepancy between the validation and test accuracy values. On the contrary run_3 network obtains the closest values, in spite of having the smallest validation accuracy.

Best Networks found in:	Validation Accuracy (%)	Test Accuracy (%)
run_0	69.78	57.14
run_1	67.98	55.03
run_2	67.98	60.32
run_3	64.03	62.96

Table 4.13: Accuracy results for the best networks found by DENSER for each run in the 150 generations study using the 250x250 new cropped images test set.

The best network topology found is described in Figure B.4 in Annex B.2.1. The network is composed of seventeen hidden-layers, one input and output layer, and 49211716 number of parameters. The filter size of the convolution layers were 44, 160 and 158, and the activation functions used were relu and sigmoid. The strides in these layers were 1x1, 2x2 and 3x3, and the kernel sizes were 2x2, 3x3 and 4x4. The pooling layers used were three maximum pooling layers, with a stride of 1x1 and 2x2, and a pool size of 3x3 and 4x4. The batch normalization layer had a momentum of 0.99, the two dropout layers had rates of 0.5 and 0.0, and the eight dense layer activation functions were linear, relu, softmax and sigmoid.

In general, this network has a normal CNN structure. However, the fact that it has eight dense layers, six of which are in a row, is unusual. This already occurred in Section 4.1.2 and in [9]. Another situation that already occurred in Section 4.1.1 was a Dropout Layer that has a rate of 0.0.

Figure 4.10 displays the ROC curve, Table 4.14 shows the confusion matrix and Table 4.15 shows the classification report where the values of precision, recall and F1-score for each class were acquired considering the test dataset and the best network acquired.

		Predicted Classes	
		Benign	Malignant
True Classes	Benign	134	97
	Malignant	65	82

Table 4.14: Confusion matrix of the best evolved network in the 150 generations study using the 250x250 new cropped images test set.

	Benign (%)	Malignant (%)
Precision	67	46
Recall	58	56
F1-score	62	50

Table 4.15: Classification report applying the 250x250 new cropped images test set in the best network found by DENSER in the 150 generations study. The classification report presents the values of precision, recall and F1-score for each class.

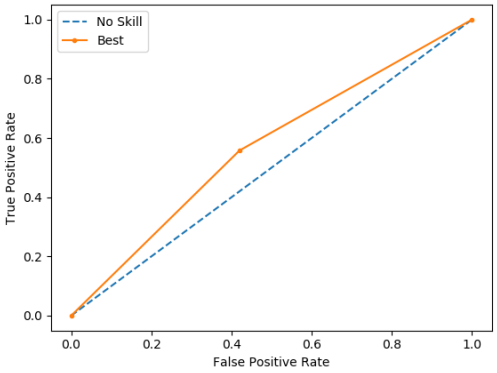


Figure 4.10: ROC curve of the best network acquired by DENSER in the 150 generations study applying the 250x250 new cropped images test set.

In the ROC curve shown in Figure 4.10 the FPR is 0.42 and the TPR is 0.56, which correspond to very poor rates.

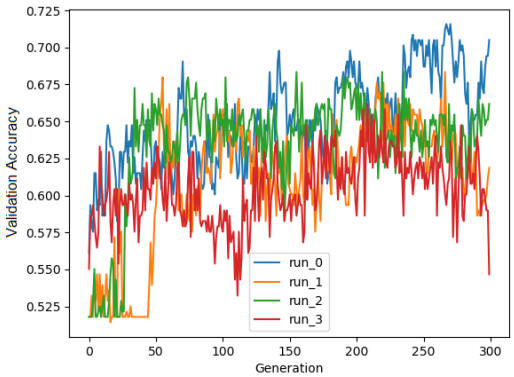
The classification report reveals that only 67% of the predictions made for the benign and 46% for the malignant classes were true. These percentages are explained by the fact that 97 out of 231 of the benign cases were classified as malignant and 82 out of 147 malignant cases were classified as benign. These values are low, especially for the malignant cases. The recall values were very similar for both classes. In the 231 benign cases, 58% were classified as such. In the 147 malignant cases, 56% were classified as malignant. These results show that the model is detecting the benign cases marginally better than the malignant ones. This result is worrying taking into consideration our

problem since our concern is to accurately detect the highest number of malignant cases.

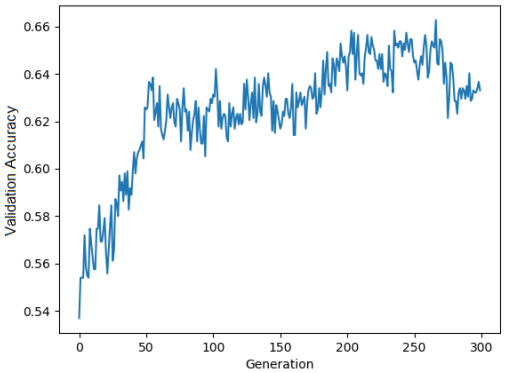
In light of these results and considering both the problems of the network's structure and the poor prediction of malignant cases, we decided to repeat this experiment for a larger number of generations.

4.2.2 Evolution Analysis for a large number of generations (300 generations)

In this study, we increased the number of generations to 300. The results for the evolutionary process are displayed in Figure 4.11. As observed in Figure 4.11a, run_1 attains a minimum value of validation accuracy of 51.43% in generation 16 and run_0 attains the maximum validation accuracy value in generation 267. In Figure 4.11b, the minimum mean validation accuracy value was 53.68% in generation 0 and the maximum was 66.28% in generation 266. The Figure also shows that between generation 0 and 266, the networks are evolving, since the accuracy value increases up until that point. However, after generation 266 a sudden drop is observed, mainly due to the run_3 performance. Comparing these values with the results of Section 4.2.1 is noticeable a slight increase in the mean maximum value. Therefore, and as expected, training for a larger number of generations increases the network's performance.



(a) Graphic of the accuracy value of the validation set in each generation for each run during the evolution of the networks.



(b) Graphic of the mean accuracy values of all runs in graphic 4.11a.

Figure 4.11: Evolution of the runs over 300 generations taking as input 250x250 new cropped images.

In the end of the evolutionary search the best networks were evaluated in the test set. The values of the test accuracy obtained are shown in Table 4.16. Comparing the values of the validation and test accuracy of runs 1, 2 and 3, it is noticeable a discrepancy between these values, i.e., there is a loss of the performance. On the contrary, the run_0 network presents a difference of 3.59% between the validation and test accuracy values, and it is the network with the higher values, with a test accuracy value of 67.99%.

Best Networks found in:	Validation Accuracy (%)	Test Accuracy (%)
run_0	71.58	67.99
run_1	68.34	57.14
run_2	68.34	55.29
run_3	66.19	58.20

Table 4.16: Accuracy results for the best networks found by DENSER for each run in the 300 generations study using the 250x250 new cropped images test set.

The best network topology is represented in Figure B.5 in Annex B.2.2 which is composed of eighteen hidden-layers, one fully connected layer as output and 11553067 parameters. The convolution layers' filter sizes are 227, 240 and 235, and the activation functions were relu and sigmoid. The convolution layers' kernel sizes were 6x6, 7x7 and 5x5, and the stride sizes were 3x3, 2x2 and 3x3, respectively. The maxpooling layers' pool sizes were 5x5, 3x3 and 4x4, and the stride sizes were 1x1, 3x3 and 3x3. The batch normalization layers have a momentum of 0.99, the dropout layers have a rate of 0.3865 and the dense layers' activation functions were linear, sigmoid and softmax.

Comparing to the previous study, training for a larger number of generations produce networks with approximately the same number of layers, but with a much lower number of parameters.

The network's topology is expected for a CNN, except for the large number of batch normalization and dense layers. Since the number of instances to train the networks is small, a higher number of batch normalization layers is used to prevent overfitting. The higher number of dense layers can also be observable in [9] and already happened in Sections 4.1.2 and 4.2.1.

The network's performance was also evaluated by the confusion matrix in Table 4.17, by the values of the classification report (precision, recall and F1-score values for each class) showed in Table 4.18 and the ROC curve, displayed in Figure 4.12.

		Predicted Classes	
		Benign	Malignant
True Classes	Benign	155	76
	Malignant	45	102

Table 4.17: Confusion matrix of the best network acquired by DENSER in the 300 generations study using the 250x250 new cropped images test set.

	Benign (%)	Malignant (%)
Precision	78	57
Recall	67	69
F1-score	72	63

Table 4.18: Classification report applying the 250x250 new cropped images test set in the best network found by DENSER in the 300 generations study. The classification report presents the precision, recall and F1-score values for each class.

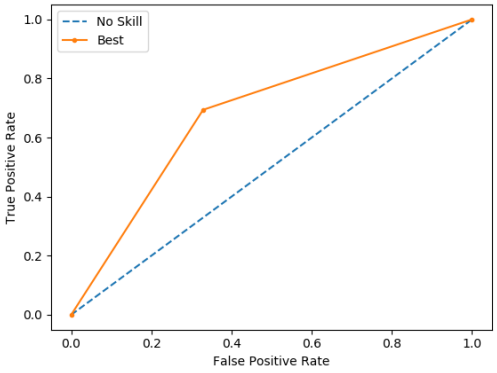


Figure 4.12: ROC curve of the best network acquired by DENSER in the 300 generations study applying the 250x250 new cropped images test set.

The ROC curve, in Figure 4.12, has a TPR of 0.69 and a FPR of 0.33. The TPR and FPR value are better than the ones obtained in the study of Section 4.2.1.

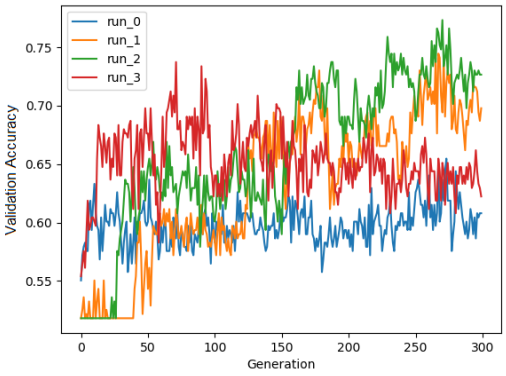
Looking at the network report, it can be seen that 78% and 57% of the benign and malignant predictions, respectively, were correct. Therefore, the model predictions continue to be higher in benign cases than the malignant ones. The recall values for both the classes are very close because the proportion of the cases for each class that was misclassified was approximately the same. For the 231 benign cases, 76 were classified as malignant, which corresponds to 67% of the cases being correctly classified. For the 147 malignant cases, 69% of the cases were classified correctly. In general, training for a larger number of generations produced better results since the accuracy, F1-score and TPR values increased, and the FPR, FN, FP values decreased when compared to the values of Section 4.2.1. However, the accuracy value is still low, considering that for a total of 378 cases, 32% were misclassified.

4.2.3 Evolution Analysis for 300 generations with Data Augmentation

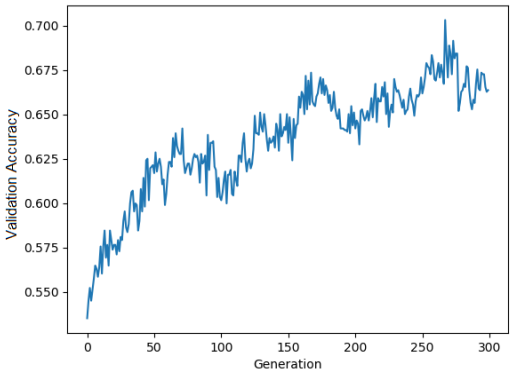
As mentioned previously one way to prevent overfitting is by applying data augmentation. Consequently, in order to get new, bigger and plausible data, two spatial augmentation operations were employed (The operations explanation was made in Section 4.1.3):

- Zoom_range : 2
- Rotation_range : 90

For this experiment, the evolutionary search ran for 300 generations since we already observed that a higher number of generations produces better results. The Figure 4.13 present the evolution of the validation accuracy through the generations. Table 4.19 contains the maximum validation accuracy values for each run of Figure 4.13a, which were 65.47%, 74.46%, 77.34% and 73.74% for run 0, 1, 2 and 3, respectively. In Figure 4.13b shows that evolution is occurring throughout the generations, since the minimum mean accuracy value was 53.51% and the maximum value was 70.32%.



(a) Graphic of the accuracy value of the validation set in each generation for each run during the evolution of the networks.



(b) Graphic of the mean accuracy values of all runs in graphic 4.13a.

Figure 4.13: Evolution of the runs over 300 generations taking as input the 250x250 new cropped images with data augmentation.

At the end of the evolutionary process, the best network of each run was collected. The networks were applied to the test set to gauge their generalization ability. The test accuracy by each network is shown in Table 4.19. The best accuracy value in the validation and test was obtained by the run_2 network (Network A), which has a difference between the accuracy values of 6.97%, which is acceptable.

Best Networks found in:	Validation Accuracy (%)	Test Accuracy (%)
run_0	65.47	58.73
run_1	74.46	62.43
run_2	77.34	70.37
run_3	73.74	68.78

Table 4.19: Accuracy results for the best networks found for each run in the 300 generation applying data augmentation study using the 250x250 new cropped images test set.

The topology of the best network (Network A) has one input layer, fifteen hidden-layers, one output layer, which is a fully-connected layer, and 5645647 parameters. This topology is displayed in Figure B.6 in Annex B.2.3. The convolution layers have filter sizes of 79, 176 and 142, with a kernel size of 5x5, stride sizes of 1x1, 2x2 and 3x3 and the activation functions were linear and relu. The pooling layers were max-pooling and average-pooling with pool sizes of 5x5 and 2x2, and stride sizes of 2x2 and 3x3. The batch normalization has a momentum of 0.99, a dropout rate of 0.5 and four fully-connected layers with linear, sigmoid and softmax as activation functions.

The network presents a number of parameters and layers lower than the ones in Section 4.2.2. These results confirm the effectiveness of using data augmentation techniques, since we obtained networks with better performance and with approximately half of the number of parameters.

In general, the topology obtained is an expected topology for a CNN.

To further evaluate the performance of the best network, we compute the ROC curve (Figure 4.14), the confusion matrix (Table 4.20) and the classification report, which is composed of the precision, recall and F1-score values for each class (Table 4.21).

		Predicted Classes	
		Benign	Malignant
True Classes	Benign	170	61
	Malignant	51	96

Table 4.20: Confusion matrix of the best network acquired by DENSER (Network A) in the 300 generations applying data augmentation study using the 250x250 new cropped images test set.

	Benign (%)	Malignant (%)
Precision	77	61
Recall	74	65
F1-score	75	63

Table 4.21: Classification report applying the 250x250 new cropped images test set in the best network found by DENSER (Network A) in the 300 generations applying data augmentation study. The classification report presents the precision, recall and F1-score values for each class.

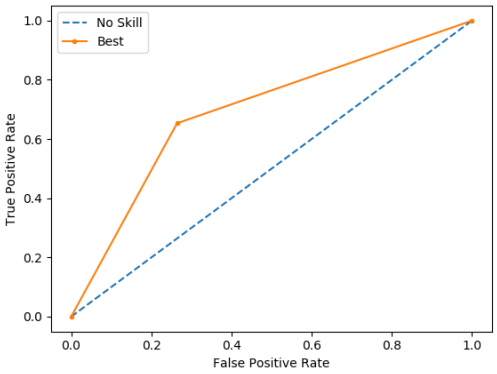


Figure 4.14: ROC curve of the best network acquired by DENSER (Network A) in the 300 generations applying data augmentation study using the 250x250 new cropped images test set.

This study obtained the best precision value for the malignant class and the best recall value for the benign class, of 61% and 74%, respectively. The values for the benign class precision and the recall of the malignant class were very close to the best ones reached until now, in the experiments described in Section 4.2.2. This was expected since the number of false positives was smaller, and the number of false negatives was nearby to the best one obtained until now (Section 4.2.2). However, taking into account the precision and recall values is possible to conclude that the model predicts better the benign cases, as already happened in the previous studies.

Another interesting aspect of this experiment, is that during the evolution, another good network was founded in run_2 (Network B). This network presents the closest values between the validation and test accuracy and the best test accuracy value.
This network was found in generation 229 in run_2, and it obtained the following accuracy values:

- Validation Accuracy value: 75.89%
- Test Accuracy value: 71.96%

The topology of this network is displayed in Figure B.7 in Annex B.2.3, and it is composed of one input layer, fourteen hidden-layers, one fully-connected layer and 4682245 parameters. Five of the hidden-layers were convolution layers with filter sizes of 79, 176, 203, 176 and 142, with kernel sizes of 7x7, 5x5 and 6x6, and stride sizes of 1x1, 3x3 and 2x2. The activation functions of these layers were linear and relu. The two max-pooling layers have pool sizes of 5x5 and 2x2, and a stride of 2x2. The batch normalization layers have a momentum of 0.99, the two dropout layers have rates of 0.5 and 0.3126, and the three fully-connected layers have as activation functions the sigmoid and softmax functions.

Notably, the network structure is a typical CNN structure and is very similar to the previous network. The differences are that this network has one more convolution and dropout layer, there is one less batch normalization and dense layer, and it does not have an average-pooling layer. The presence of two dropouts in a row already occurred in Section 4.1.1, and it helps in the regularization of the network's parameters.

The network was the best network found until now, since it has the fewer number of layers and parameters.

However, this network exhibits unusual behaviour that goes against what is described in existing literature because the number of neurons in the fully-connected layers is increasing with depth. Normally, the number of neurons decreases with depth. This can be a new approach since this network presents the best performance.

The network's performance was assessed through the ROC curve, in Figure 4.15, and the precision, recall and F1-score of each class, displayed in Table 4.23. Table 4.22 exhibits the confusion matrix.

		Predicted Classes	
		Benign	Malignant
True Classes	Benign	169	62
	Malignant	44	103

Table 4.22: Confusion matrix of the network with the best performance (Network B) acquired by DENSER in the 300 generations applying data augmentation study using the 250x250 new cropped images test set.

	Benign (%)	Malignant (%)
Precision	79	62
Recall	73	70
F1-score	76	66

Table 4.23: Classification report applying the 250x250 new cropped images test set in the network with the best performance (Network B) acquired by DENSER in the 300 generations applying data augmentation study. The classification report presents the precision, recall and F1-score values for each class.

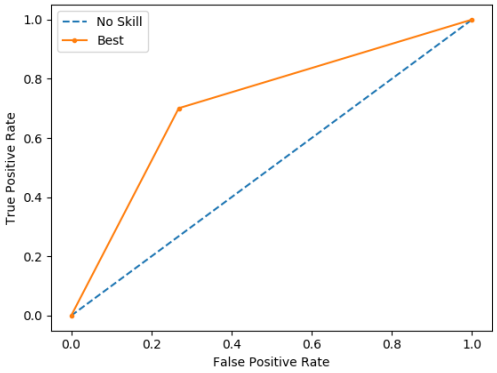


Figure 4.15: ROC curve of the network with the best performance (Network B) acquired by DENSER in the 300 generations applying data augmentation study using the 250x250 new cropped images test set.

In the ROC curve the TPR and FPR values were 0.70 and 0.27. The TPR was the best one achieved until now.

The classification report values were the best ones acquired for both the classes, comparing to previous studies, with the exception of the recall value for the benign cases which was 1% inferior to the best value. Notwithstanding, this network presents the smallest number of FN of 44, and the FP value is very near to the best one of 61.

Compared to the previous networks, this network has the smallest number of layers and parameters. Hence, the data augmentation process is an approach to keep into consideration since it improved the network training.

In addition to the previous assessments, it is important to test the model interpretability, by resorting to *saliency maps*.

Annex C reports the saliency maps of 50 images of the test set taking into account the two best networks of this study, Network A and B. In Figure 4.16 are shown only two images, one without black background and the other with a black background. It can be seen that both the models ignore the black background, focusing more on the breast region, as intended. This did not occur in the saliency maps of Section 4.1.3, which demonstrates that incorporating the surrounding information creates more effective models. The saliency map obtained with Network A is more scattered comparing to the saliency map of Network B for both images. In general, and as expected, considering the 50 images, Network B present more precise saliency maps comparing to Network A saliency maps.

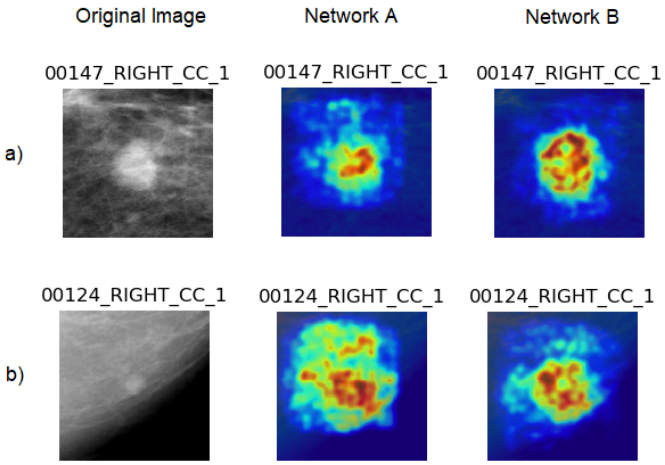


Figure 4.16: Saliency maps of two distinct images (a and b): CC view of patients '00147' and '00124' right breast. The first image is the cropped image, the second image is the saliency map resorting to the best network (Network A) acquired by DENSER and the third image is the saliency map resorting to the network with the best performance (Network B) acquired by DENSER.

4.2.4 Ensembling of the Best Networks

As already explained in Section 4.1.4 the creation of an ensemble of the best networks might help reduce the variance of the models and improve the predictions.

The ensembling was implemented considering the 4 fittest networks using majority voting. Table 4.24 shows the test accuracy values considering the ensembling of the 4, 3 and 2 fittest networks. The ensembling of the 4 fittest networks gives the best test accuracy of 74.60%. These results reflect the efficiency of this approach in attaining better predictions since the test accuracy increased by 3 values respectively to the best obtained value of 72.00%.

<table>
<tr><th>Network</th><th>Single Network
Accuracy Validation (%)</th><th colspan="3">Ensemble Network Accuracy Test (%)</th></tr>
<tr><td>A</td><td>77.34</td><td rowspan="3">74.60</td><td rowspan="2">71.43</td><td rowspan="2">73.54</td></tr>
<tr><td>B</td><td>75.89</td></tr>
<tr><td>C</td><td>74.46</td><td rowspan="2">-</td></tr>
<tr><td>D</td><td>73.74</td><td>-</td></tr>
</table>

Table 4.24: Test accuracy considering the ensembling of the 4, 3 and 2 fittest networks of the Section 4.2. This networks are: the best network (Network A), the network with the best performance (Network B), the run_1 network (Network C) and the run_3 network (Network D) of the Section 4.2.3.

To further evaluate the ensemble of the 4 fittest networks we computed the confusion matrix, the ROC curve and the classification report (precision, recall and F1-score values for each class) as Table 4.25, Table 4.26 and Figure 4.17 display.

		Predicted Classes	
		Benign	Malignant
True Classes	Benign	187	44
	Malignant	52	95

Table 4.25: Confusion matrix of the ensembling of the 4 fittest networks of Section 4.2 using the 250x250 new cropped images test set.

	Benign (%)	Malignant (%)
Precision	78	68
Recall	81	65
F1-score	80	66

Table 4.26: Classification report applying the 250x250 new cropped images test set in the ensembling of the 4 fittest networks of Section 4.2. The classification report presents the precision, recall and F1-score values for each class.

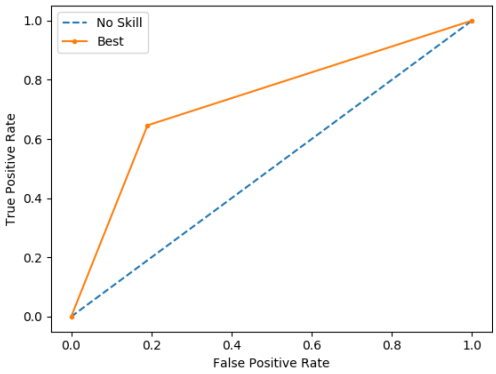


Figure 4.17: ROC curve of the ensembling of the 4 fittest networks of Section 4.2 using the 250x250 new cropped images test set.

The FPR and TPR values were 0.19 and 0.65. The FPR was the smaller value reached in this experiment. This is due to the fact that the FP value was the lower value acquired until now, out of 44. However, the FN value is 7 values higher than the best value achieved. The precision value for the malignant class and the recall value for the benign class were the best attained until now. The precision value for the benign class was very close to the best value. In general, and considering the values of the F1-score, the predictions for the malignant class maintain its performance, but the predictions for the benign ones increase. Hence, the model is detecting benign cases better than malignant cases.

4.3 DENSER vs An Ad Hoc Random Initialization Deep Neural Network Architecture

DENSER results were compared with another approach, an Ad Hoc random initialization CNN architecture [11], which used the same dataset.

The authors explore a total of 260 model architectures and used a hold-out protocol to evaluate the models. The evaluation was made by measuring the performance of the models in the validation set based on two separate criteria: (A) highest AUC (Model A) and (B) best F2 score (Model B). The model selection was based also on the reduced number of false negatives cases while maintaining an adequate accuracy value.

The dataset was split into 1158 images to train, 160 images for validation and 378 images for testing. Data augmentation was employed, namely, random rotations, rescaling and shear deformations.

In the end, both the criteria, achieved approximately the same model architecture: 3 convolutional layers with 64 kernels, sizes of 7×7, 5×5, and 3×3, respectively, and relu as the activation function; the dropout rate on each convolution was 0.25; 3 max-pooling layers with sizes 4×4, 3×3, and 2×2 (and same stride) were employed. The difference was in the input size and in the neuronal architecture. In the model obtained with the A

criteria (Model A) the images had 238x238 pixels and the fully connected layers had 50 to 10 neurons. In the model obtained with the B criteria (Model B) the images had 286x286 pixels and the fully connected layers had 50 to 20 neurons.

To train the models the authors resorted to 4000 training epochs, taking approximately 78 hours to train on a 40-CPU dedicated HP Blade system. Model A acquired an AUC of 0.785 and Model B an AUC of 0.774. Model A was the one with the highest AUC and it gets an accuracy of 71.19% on the test set.

Table 4.27 presents the parameters and results of Model A [11], of the best network from Section 4.1 (Network B) and the best network from Section 4.2 (Network C). Networks B and C have a higher number of instances to validate the network, getting a smaller number to train. The images size for Model A and Network C is relatively the same, and the images for Network B are approximately 70% smaller than Model A images size. Model A and Network B images are more focused on the lesion region, and Network C images have included more surrounding information. The three models implemented data augmentation. Even though Network B and C have used a restricted set of transformations. The number of hidden-layers is 12, 13 and 13, for Model A, Network C and Network B, respectively, Model A does not resort to Batch-Normalizations layers, which negatively impacts the training time. Network B and C disadvantage is the substantial number of neurons in the last layers. The biggest advantage of these networks is the lower training time and the number of training epochs, due to the fact that this study it was used a 1-GPU instead of 40-CPU. Both the Networks acquired by DENSER achieved better test accuracy values. Network C attained a smaller value because the inputs contain more information, therefore require more training. Therefore, the input of Model A and Network B only differs on the size, where the Network B images are 70% smaller (which is an advantage), and the test accuracy value of Network B is 2.62% higher, this demonstrates the efficacy of DENSER to provide better networks.

		An Ad Hoc Random Initialization DNN Architecture (Model A)	Section 4.1.3 (Network B)	Section 4.2.3 (Network C)
Data Division (images)	Train Set	1158	660	
	Validation Set	160	658	
	Test Set	378	378	
Images Size (pixels)		238x238	90x90	250x250
Data Augmentation	Rotation	✓	90º	
	Rescaling	✓	-	
	Shear Deformation	✓	-	
	Flip	-	Horizontal	
Model Architecture		Conv(3), MaxPool(3), Dropout(3), Dense(1 to 3)	Conv(5), MaxPool(1), BatchNorm(2), Dropout(1), Dense(4)	Conv(5), MaxPool(2), BatchNorm(2), Dropout(2), Dense(2)
Fully Connected Layer Neurons		50 to 10	1413 to 345	825 to 1275
Training Epochs		4000	537	313
Training Time (Hours)		78.00	0.83	1.16
CPU		40-CPU (HP Blade system)	1 GPU NVIDIA 1080 Ti	
Test Accuracy Value		71.19%	73.81%	71.96%

Table 4.27: Results and parameters comparison between the literature model, an Ad Hoc random initialization CNN architecture [11], and the best network acquired with DENSER in Section 4.1 (Network B), and the best performance network acquired by DENSER in Section 4.2 (Network C)

CHAPTER 5

CONCLUSIONS

The growing incidence and mortality worldwide demand for an investment and development on early, fast, and rigorous diagnostic methods and techniques. Some of these techniques rely on medical imaging to do the screening, such as X-ray mammography. After the collection of the images, they are analysed and evaluated by a human specialist, e.g., a radiologist. However, this process is highly dependent on expert knowledge and is subject to a high variance, which might result in the wrong diagnostic. To mitigate these problems, one can resort to computational methods. Hence, emerges the combination of Radiomics and Machine Learning (ML). This combination raises other challenges, related to the increase of problems complexity, making traditional Machine Learning ineffective when dealing with complex tasks such as cancer diagnostic. To overcome this challenge, we can resort to Deep Learning (DL) methods to automatically extract and learn from large amount of data features. Nevertheless, the methods used nowadays to construct DL models have some problems when dealing with small datasets, such as medical imaging datasets, which can lead to poor performance or overfitting. Taking this into account, it is necessary to search and develop new approaches to construct effective DL models. Therefore, one novel possibility is resort to methods that automatically design DL models such as DENSER, which combines Machine Learning and Evolutionary Computation.

In this work, we evaluate the capabilities of DENSER to build DL models to help in breast cancer diagnostic. In concrete, two different experiments were performed, one with the original cropped images, 90x90 grayscale images, in Section 4.1 and other with cropped images with more surrounding information, 250x250 grayscale images, in Section 4.2. For each experiment, we study the impact that the number of generations has on the quality of the model, as well as the impact of data augmentation techniques. Furthermore, the interpretability of the best network and the ensemble of the 2, 3 and 4 fittest networks in each experiment were studied.

In Section 4.1 the images focus more on the region of the lesion and have inferior

dimensions than the images of Section 4.2, resulting in images with less information to be processed by the networks. The best network evolved by DENSER for this experiment was obtained when using 300 generations and data augmentation, attaining validation and test accuracy values of 79.82% and 73.81%, respectively. The best network was composed of approximately 2 million parameters and 13 hidden-layers. Concerning its performance, it presents a FPR value of 0.18, the lowest number of FP cases, the best precision and recall values for the benign class and the best precision value for the malignant class. However, the acquired saliency maps are very unclear, since practically all of the image is used, including the black background to generate the predictions, which indicates that the model continues to be influenced by components beyond the lesion area. Concerning, the ensembling of the 4 fittest networks produces better results than the best network acquired, attaining a test accuracy value of 76.19% and a FPR value of 0.16.

In the second experiment, the images had more information (pixels) and in some cases, the lesion was not centred. The best network obtained in the study was evolved over 300 generations and with data augmentation techniques. This network presents validation and test accuracy values of 75.89% and 71.96%, and it has the smallest number of layers and parameters (approximately 4 million parameters). Furthermore, it presents the best test accuracy value, the smaller number of FN, the higher TPR and the higher F1-score values. The results of the saliency maps reflect the good performance of this network since the results were more precise and less scattered. Additionally, these maps show that the inclusion of surrounding information allows the networks to dismiss the images' black background and focus on the important parts. This emphasises the importance of including the surrounding information in this type of problem. The ensembling of the 4 fittest networks of this Section generates the best prediction value, with a test accuracy value of 75.00% and the lowest FPR.

Finally, the results obtained by our models outperform the existing ones in the literature for the same dataset. The best accuracy test value we obtained was 73.81% and the best reported in the literature was 71.19% [11]. Additionally, our models need less training time, fewer epochs, and no domain knowledge.

In summary, our results show that training for a larger number of generations and implementing data augmentation provide effective and lightweight networks. Additionally, some of the topologies obtained are novel and exhibit characteristics different from those designed by humans, which is remarkable. The saliency maps confirm that the introduction of the surrounding information enhances the network's performance helping to focus more accurately on the lesion region. The ensemble of the 4 fittest networks improves the predictions, as pretending. Moreover, an important outcome was the efficacy of DENSER when manipulating small datasets, originating lightweight networks with around 2 to 4 million parameters.

5.1 Future Work

Notwithstanding the obtained results, there is still space for improvement. In the future, and taking into account the results obtained, it is necessary to do more tests and experiments.

Firstly, the use of a more recente dataset with better quality images and information, since the dataset used was acquired using old technology. This dataset was chosen due to the fact it was the public breast dataset with more images. Furthermore, the use of a more recent dataset acquired by novel technologies might help achieve better results. Afterwards, in our studies, we resort to 8-bit images. However, medical images are 16-bit images. Therefore, the use of the original 16-bit images with more information is a procedure to be implemented. Another improvement is increasing the number of generations and runs to evolve the networks for a longer time to see if we get more accurate results.

Another study that has to be performed, is to split the view of the images and train it separately, and analyze the influence of each image view in the training process. These views catch different plans of the breast, which presents different lesion information. This can impair the training of the networks when combining both types of views since the information is set differently.

The introduction of clinical data can also produce better results, such as age, sex and historical medical information might help in the achievement of better results.

BIBLIOGRAPHY

[1] Hyuna Sung et al. "Global cancer statistics 2020: GLOBOCAN estimates of incidence and mortality worldwide for 36 cancers in 185 countries". In: *CA: A Cancer Journal for Clinicians* (). DOI: https://doi.org/10.3322/caac.21660.

[2] Etta D. Pisano et al. "Diagnostic Performance of Digital versus Film Mammography for Breast-Cancer Screening". In: *New England Journal of Medicine* 353.17 (2005). PMID: 16169887, pp. 1773–1783. DOI: 10.1056/NEJMoa052911.

[3] Nickolas Papanikolaou and João Santinha. "An Introduction to Radiomics: Capturing Tumour Biology in Space and Time". In: 2018.

[4] Simeon E. Spasov et al. "A Multi-modal Convolutional Neural Network Framework for the Prediction of Alzheimer's Disease". In: (2018), pp. 1271–1274. DOI: 10.1109/EMBC.2018.8512468.

[5] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton. "ImageNet Classification with Deep Convolutional Neural Networks". In: *Commun. ACM* 60.6 (May 2017), pp. 84–90. ISSN: 0001-0782. DOI: 10.1145/3065386. URL: https://doi.org/10.1145/3065386.

[6] P. Afshar et al. "From Handcrafted to Deep-Learning-Based Cancer Radiomics: Challenges and Opportunities". In: *IEEE Signal Processing Magazine*. Vol. 36. 4. 2019, pp. 132–160. DOI: 10.1109/MSP.2019.2900993.

[7] M. Rosenstein. "To transfer or not to transfer". In: *Neural Information Processing Systems 2005*. 2005.

[8] Felix Streichert. *Introduction to Evolutionary Algorithms*.

[9] Filipe Assunção et al. "DENSER: deep evolutionary network structured representation". In: *Genetic Programming and Evolvable Machines* 20.1 (Sept. 2018), pp. 5–35. DOI: 10.1007/s10710-018-9339-y.

[10] R. S. Lee et al. “A curated mammography data set for use in computer-aided detection and diagnosis research”. In: 4.170117 (2017). DOI: 10.1038/sdata.2017.177.

[11] Andrea Duggento et al. “An Ad Hoc Random Initialization Deep Neural Network Architecture for Discriminating Malignant Breast Cancer Lesions in Mammographic Images”. In: Mammographic Images. Contrast Media & Imaging, 2019. DOI: 10.1155/2019/5982834.

[12] François Chollet. *Deep Learning with Python*. Ed. by Manning Publications Co. 1st. 2017. Chap. 1-5.

[13] Andriy Burkov. “The Hundred-Page Machine Learning Book . Draft”. In: chap. 1,3,6.

[14] Seyedali Mirjalili, Hossam Faris, and Ibrahim Aljarah. “Introduction to Evolutionary Machine Learning Techniques”. In: *Evolutionary Machine Learning Techniques: Algorithms and Applications*. Singapore: Springer Singapore, 2020, pp. 1–7. DOI: 10.1007/978-981-32-9990-0_1.

[15] Rikiya Yamashita et al. “Convolutional neural networks: an overview and application in radiology”. In: *Insights into Imaging* 9 (June 2018). DOI: 10.1007/s13244-018-0639-9.

[16] M. G. Ertosun and D. L. Rubin. “Probabilistic visual search for masses within mammography images using deep learning”. In: *2015 IEEE International Conference on Bioinformatics and Biomedicine (BIBM)*. 2015, pp. 1310–1315. DOI: 10.1109/BIBM.2015.7359868.

[17] M. Alkhaleefah and C. Wu. “A Hybrid CNN and RBF-Based SVM Approach for Breast Cancer Classification in Mammograms”. In: *2018 IEEE International Conference on Systems, Man, and Cybernetics (SMC)*. 2018, pp. 894–899. DOI: 10.1109/SMC.2018.00159.

[18] T. Pang et al. “Deep learning radiomics in breast cancer with different modalities: Overview and future”. In: *Expert Systems with Applications* 158 (2020), p. 113501. ISSN: 0957-4174. DOI: https://doi.org/10.1016/j.eswa.2020.113501.

[19] Harith Al-Sahaf et al. “A survey on evolutionary machine learning”. In: *Journal of the Royal Society of New Zealand* 49.2 (2019), pp. 205–228. DOI: 10.1080/03036758.2019.1609052.

[20] A. E. Eiben and James E. Smith. “Introduction to Evolutionary Computing”. In: 2nd. Springer Publishing Company, Incorporated, 2015. Chap. I.

[21] Riccardo Poli, William B. Langdon, and Nicholas Freitag McPhee. “A Field Guide to Genetic Programming”. In: Lulu Enterprises, UK Ltd, 2008. Chap. 1-2.

[22] Omid E. David and Iddo Greental. “Genetic algorithms for evolving deep neural networks”. In: *Proceedings of the 2014 conference companion on Genetic and evolutionary computation companion - GECCO Comp ’14* (2014). DOI: 10.1145/2598394.2602287.

[23] Boris Shabash and Kay Wiese. “EvoNN: a customizable evolutionary neural network with heterogenous activation functions”. In: July 2018, pp. 1449–1456. DOI: 10.1145/3205651.3208282.

[24] A. Radi and R. Poli. “Discovery of backpropagation learning rules using genetic programming”. In: *1998 IEEE International Conference on Evolutionary Computation Proceedings. IEEE World Congress on Computational Intelligence (Cat. No.98TH8360)*. 1998, pp. 371–375. DOI: 10.1109/ICEC.1998.699761.

[25] Masanori Suganuma, Shinichi Shirakawa, and Tomoharu Nagao. *A Genetic Programming Approach to Designing Convolutional Neural Network Architectures*. 2017. arXiv: 1704.00764 [cs.NE].

[26] K. Soltanian et al. “Artificial neural networks generation using grammatical evolution”. In: *2013 21st Iranian Conference on Electrical Engineering (ICEE)*. 2013, pp. 1–5. DOI: 10.1109/IranianCEE.2013.6599788.

[27] Andrew James Turner and Julian Francis Miller. “The Importance of Topology Evolution in NeuroEvolution: A Case Study Using Cartesian Genetic Programming of Artificial Neural Networks”. In: *Research and Development in Intelligent Systems XXX*. Ed. by Max Bramer and Miltos Petridis. Cham: Springer International Publishing, 2013, pp. 213–226.

[28] Kenneth O. Stanley and Risto Miikkulainen. “Evolving Neural Networks through Augmenting Topologies”. In: *Evol. Comput.* 10.2 (June 2002), pp. 99–127. DOI: 10.1162/106365602320169811.

[29] Jason Liang et al. *Evolutionary Neural AutoML for Deep Learning*. 2019.

[30] J. Waring, C. Lindvall, and Renato Umeton. “Automated machine learning: Review of the state-of-the-art and opportunities for healthcare”. In: *Artificial intelligence in medicine* 104 (2020), p. 101822.

[31] Esteban Real et al. *Regularized Evolution for Image Classifier Architecture Search*. 2019.

[32] Alejandro Martín et al. “EvoDeep: A new evolutionary approach for automatic Deep Neural Networks parametrisation”. In: *J. Parallel Distributed Comput.* 117 (2018), pp. 180–191.

[33] Risto Miikkulainen et al. “Evolving Deep Neural Networks”. In: *Artificial Intelligence in the Age of Neural Networks and Brain Computing*. Ed. by Robert Kozma et al. Amsterdam: Elsevier, 2018.

[34] Ying Bi, Bing Xue, and M. Zhang. "An Evolutionary Deep Learning Approach Using Genetic Programming with Convolution Operators for Image Classification". In: *2019 IEEE Congress on Evolutionary Computation (CEC)* (2019), pp. 3197–3204.

[35] Esteban Real et al. "Large-Scale Evolution of Image Classifiers". In: *Proceedings of the 34th International Conference on Machine Learning*. Ed. by Doina Precup and Yee Whye Teh. Vol. 70. Proceedings of Machine Learning Research. International Convention Centre, Sydney, Australia: PMLR, June 2017, pp. 2902–2911.

[36] Esteban Real et al. *Regularized Evolution for Image Classifier Architecture Search*. 2019. arXiv: 1802.01548 [cs.NE].

[37] Yanan Sun et al. "Automatically Designing CNN Architectures Using the Genetic Algorithm for Image Classification". In: *IEEE Transactions on Cybernetics* 50.9 (Sept. 2020), pp. 3840–3854. DOI: 10.1109/tcyb.2020.2983860.

[38] F. Assunção et al. "Evolving the Topology of Large Scale Deep Neural Networks". In: *European Conference on Genetic Programming*. Springer, 2018, pp. 19–34.

[39] Filipe Assunção et al. "Fast-DENSER: Fast Deep Evolutionary Network Structured Representation". In: *SoftwareX* 14 (2021), p. 100694. ISSN: 2352-7110. DOI: https://doi.org/10.1016/j.softx.2021.100694.

[40] H.J. Aerts. "The Potential of Radiomic-Based Phenotyping in Precision Medicine: A Review". In: *JAMA Oncol*. Vol. 2. 12. 2016, pp. 1636–1642. DOI: 10.1001/jamaoncol.2016.2631.

[41] S. Rizzo et al. "Radiomics: the facts and the challenges of image analysis". In: *European Radiology Experimental*. Vol. 2. 1. 2018. DOI: 10.1186/s41747-018-0068-z.

[42] V. Kumar et al. "Radiomics: the process and the challenges". In: *Magn Reson Imaging*. Vol. 30. 9. 2012, pp. 1234–1248. DOI: 10.1016/j.mri.2012.06.010.

[43] R.J. Gillies, P.E. Kinahan, and H. Hricak. "Radiomics: Images Are More than Pictures, They Are Data. Radiology". In: 278.2 (2016), pp. 563–577. DOI: 10.1148/radiol.2015151169.

[44] A. Hosny et al. "Deep learning for lung cancer prognostication: A retrospective multi-cohort radiomics study". In: *PLoS Med*. Vol. 15. 11. 2018. DOI: 10.1371/journal.pmed.1002711.

[45] C. Lehman et al. "Mammographic Breast Density Assessment Using Deep Learning: Clinical Implementation." In: *Radiology* 290 1 (2019), pp. 52–58. DOI: 10.1148/radiol.2018180694.

[46] B. Q. Huynh, H. Li, and M. L. Giger. "Digital mammographic tumor classification using transfer learning from deep convolutional neural networks". In: *J Med Imaging (Bellingham)* 3.3 (July 2016), p. 034501.

[47] N. Antropova, B. Q. Huynh, and M. L. Giger. "A deep feature fusion methodology for breast cancer diagnosis demonstrated on three imaging modality datasets". In: *Med Phys* 44.10 (Oct. 2017), pp. 5162–5171.

[48] H. Zhang et al. "Diagnostic Efficiency of the Breast Ultrasound Computer-Aided Prediction Model Based on Convolutional Neural Network in Breast Cancer". In: *J Digit Imaging* 33.5 (Oct. 2020), pp. 1218–1223.

[49] J. Arevalo et al. "Convolutional neural networks for mammography mass lesion classification". In: *2015 37th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC)*. 2015, pp. 797–800. DOI: 10.1109/EMBC.2015.7318482.

[50] Y. Zhou et al. "A Radiomics Approach With CNN for Shear-Wave Elastography Breast Tumor Classification". In: *IEEE Transactions on Biomedical Engineering* 65.9 (2018), pp. 1935–1942. DOI: 10.1109/TBME.2018.2844188.

[51] Xiao-Yan Zhang et al. "Predicting Rectal Cancer Response to Neoadjuvant Chemoradiotherapy Using Deep Learning of Diffusion Kurtosis MRI". In: *Radiology* 296 (Apr. 2020), p. 190936. DOI: 10.1148/radiol.2020190936.

[52] Andrei Margeloiu et al. *Improving Interpretability in Medical Imaging Diagnosis using Adversarial Training*. 2020. arXiv: 2012.01166 [cs.LG].

[53] Leonardo Trujillo et al. "NEAT Genetic Programming: Controlling bloat naturally". In: *Information Sciences* 333 (2016), pp. 21–43. ISSN: 0020-0255. DOI: https://doi.org/10.1016/j.ins.2015.11.010.

[54] M. A. Ganaie et al. *Ensemble deep learning: A review*. 2021. arXiv: 2104.02395 [cs.LG].

Annexes

ANNEX A

GRAMMAR AND EVOLUTION PARAMETERS

The networks' evolution depends on the grammar and the evolutionary parameters. Therefore, the comprehension and interpretation of these parameters and their relation are critical to understanding DENSER [9].

A.1 Grammar Parameters

The grammar is based on Backus-Naur Form (BNF). Grammar 3.1 is an example of grammar. The initialization symbols of this grammar are called non-terminal symbols, such as feature and classification. These symbols contain the hyper-parameters of each of the evolutionary units, which can be integer, float or closed values. The float and integer values, such as stride, are encoded taking into account a 5-tuple: variable name, variable type, number of values to generate, minimum and maximum values. An example of a closed value is the padding [9].
The evolutionary units can encode layers or learning algorithms. The layers start by layer:layer_type and learning start with learning:learning_algorithm. In Tables A.1 and A.2 is display the hypothesis for the layer and learning algorithms.

Layer type	Float or integer hyper-parameters	Non-terminal	Closed values
Convolution	Number of filters, Filter shape and Stride	Padding	Same or Valid
		Activation function	Linear, Relu or Sigmoid
		Bias	True or False
Pooling	Kernel size and Stride	Pool type	Average or Max Pooling
		Padding	Same or Valid
Batch-normalization	-	-	-
Dropout	Rate	-	-
Fully-connected	Number of units	Activation function	Linear, Relu or Sigmoid
		Bias	True or False
Softmax	Number of units, Bias:True, Activation function: softmax		

Table A.1: Hyper-parameters required by each layer type

The layers types and its hyper-parameters:

- Convolution - Apply convolution to the input;
  - Number of filters - filters applied to the input resulting into a feature map;
  - Filter shape - Define the filter dimension;
  - Stride - The amount that the filter shifts;
  - Padding - Allows that the input and output dimensions stay the same. "Valid" padding is not applied, and "Same" padding is applied;
  - Activation Function - function implemented to the filters to obtained the feature map. In this case, the function can be linear, relu or sigmoid;
  - Bias - Used to adjust the layer's output;
- Pooling - Used to reduce the size of the spatial dimensions when increasing the number of channels;
  - Kernel size - size of the filter to be applied;
  - Stride
  - Pool type - Defines the way that the dimensions are reduced, taking into account the maximum value of each patch of the feature map or its average.
  - Padding
- Dropout - Drops the input units to 0 with a certain rate, randomly;

- Fully-connected - layer where all the inputs of the previous layer are connected to all the activation units of the next layer;
  - Number of units
  - Activation Function
  - Bias
- Softmax - Normalizes the input into a probabilistic distribution;
  - Number of units
  - Bias
  - Activation function

Learning Algorithm	Float or integer hyper-parameters	Non-terminal	Closed values
Gradient Descent	Learning rate (lr), momentum, Learning decay, Batch size and number of epochs	Nesterov	True or False
		Early stop	-
Adam	Learning rate (lr), Learning decay, beta1, beta2, Batch size and number of epochs	Early stop	-
RMSProp	Learning rate (lr), Learning decay, rho, Batch size and number	Early stop	-

Table A.2: Hyper-parameters required by each learning algorithm

The learning algorithms and its hyper-parameters:

- Gradient Descent - Optimization algorithm used to update the parameters of the model;
  - Learning rate - controls the update speed;
  - momentum - It is a variant of the gradient descent, which speeds up the training;
  - Learning decay
  - Batch size - Number of samples which will be propagated through the network;
  - Number of epochs - controls the number of complete transitions through the training set;
  - Nesterov
  - Early stop - method used to stop training when the model's performance stop improving (hold out method);
- Adam - Optimization algorithm
  - Learning rate

  - Learning decay
  - beta1 and beta2 - parameters to control the decay rate;
  - Batch size
  - Number of epochs
  - Early stop

- RMSprop - Optimization algorithm
  - Learning rate
  - Learning decay
  - rho - parameter to control the decay rate;
  - Batch size
  - Number of epochs
  - Early stop

A.2 Evolution Parameters

The experimental parameters are divided into four categories, as explained in section 3.2.3. The evolutionary parameters are related to the evolutionary algorithms. The parameters and its respective explanation, are in table A.3.

Evolutionary Engine	**Explanation**
Number of runs	Number of studies made in parallel
Number of generations	Number of generations of the best network evolve
Population size	Number of evolve individuals
Add layer rate	Likelihood of randmoly adding a layer*
Reuse layer rate	Likelihood of reusing a layer*
Remove layer rate	Likelihood of randmoly removing a layer*
Add connection rate	Likelihood of randmoly adding a connection*
Remove connection rate	likelihood of randmoly removing a connection*
DSGE-level rate	Rate of changing the DSGE genotype*

Table A.3: Evolutionary parameters used to evolve the networks (*Mutation rates).

ANNEX B

NETWORK TOPOLOGY

B.1 Grayscale Studies (90x90 Images)

B.1.1 Evolution Analysis for 300 generations

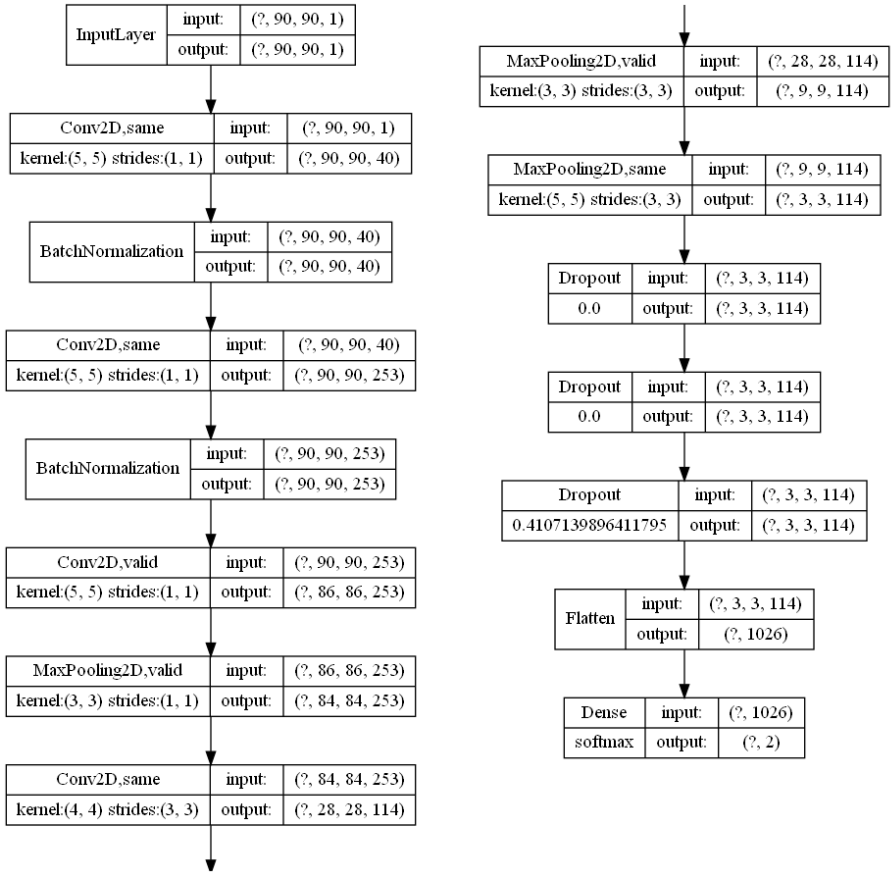


Figure B.1: Topology of the fittest CNN evolved by DENSER in the 300 generations study with 90x90 cropped images.

B.1.2 Evolution Analysis for 150 generations with Data Augmentation

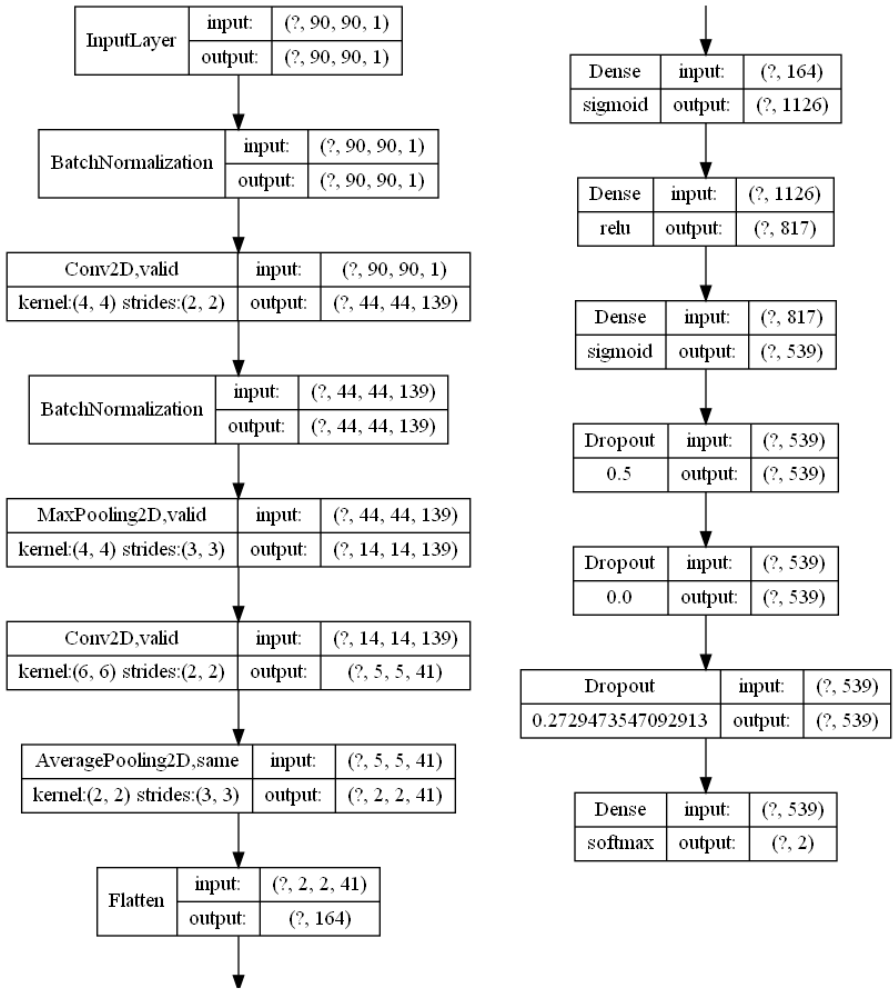


Figure B.2: Topology of the fittest CNN evolved by DENSER in the 150 generations applying data augmentation study with 90x90 cropped images.

B.1.3 Evolution Analysis for 300 generations with Data Augmentation

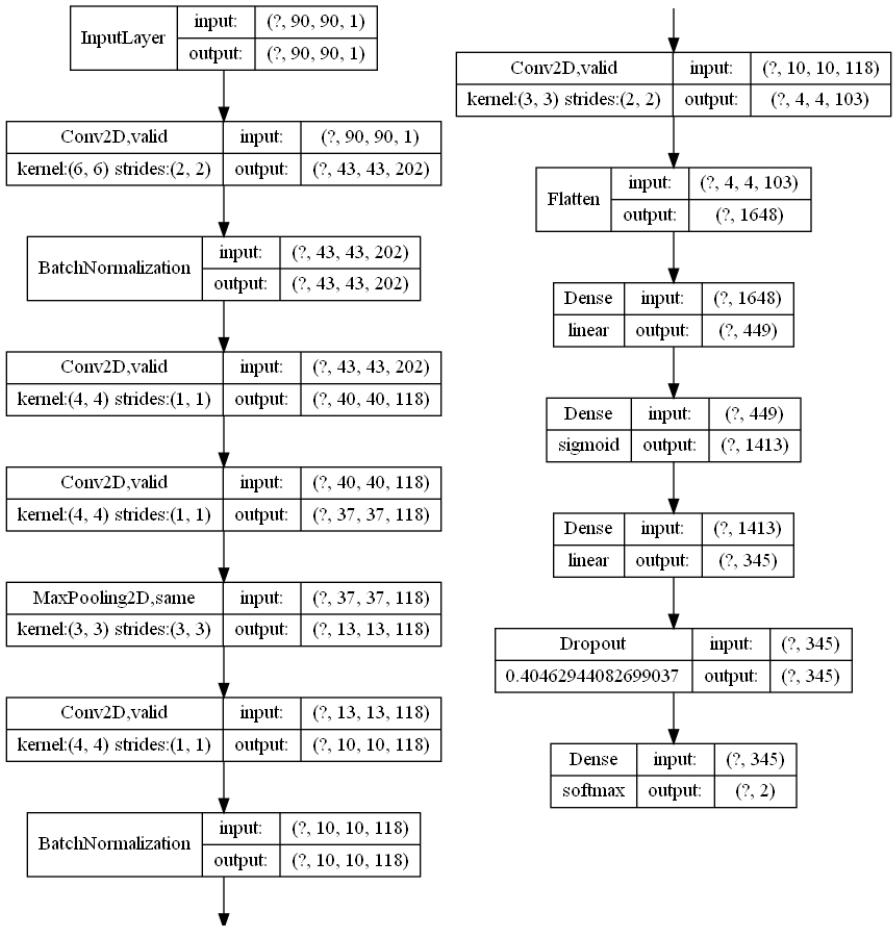


Figure B.3: Topology of the fittest CNN evolved by DENSER in the 300 generations applying data augmentation study with 90x90 cropped images.

B.2 Grayscale Images with more surrounding information Studies (250x250 Images)

B.2.1 Evolution Analysis for 150 generations

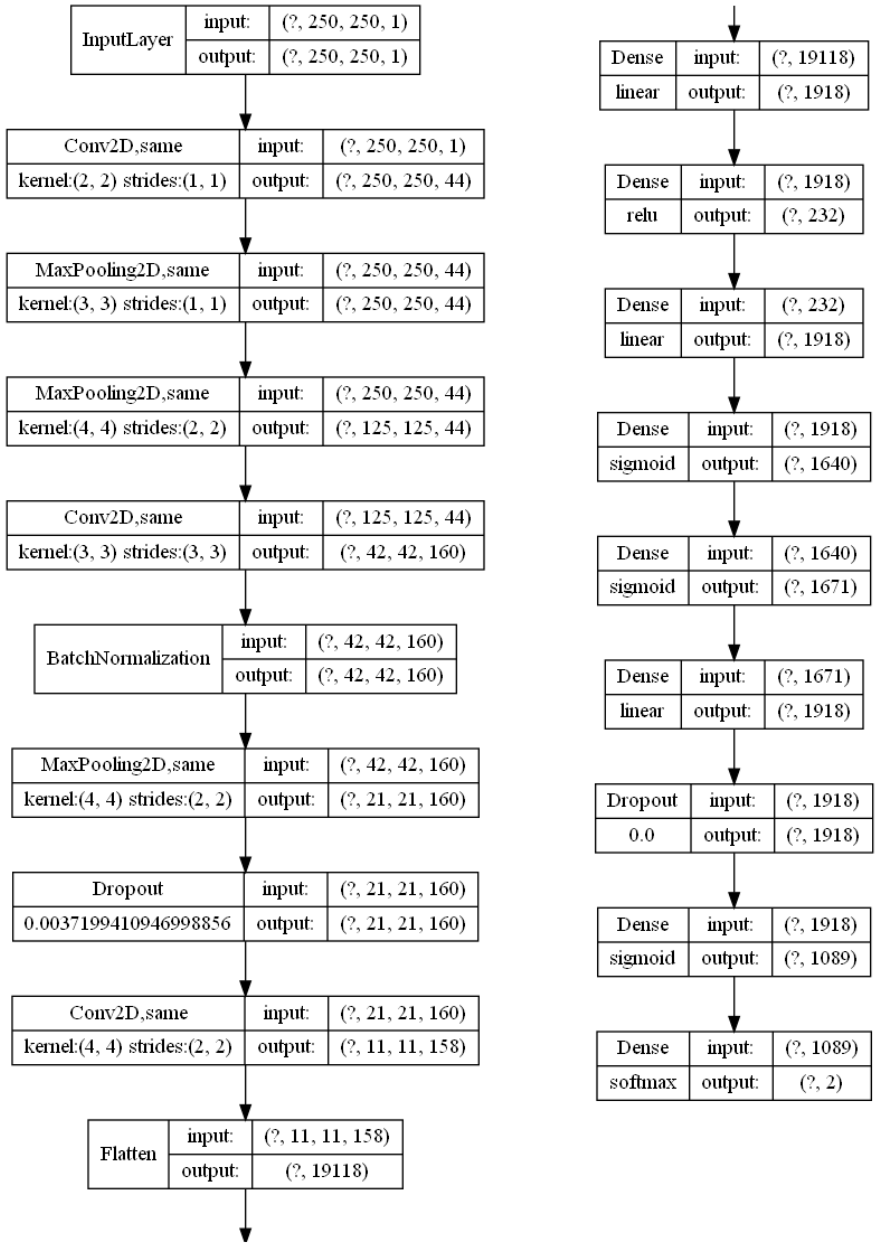


Figure B.4: Topology of the fittest CNN evolved by DENSER in the 150 generations study with 250x250 new cropped images.

B.2.2 Evolution Analysis for a larger number of generations (300 generations)

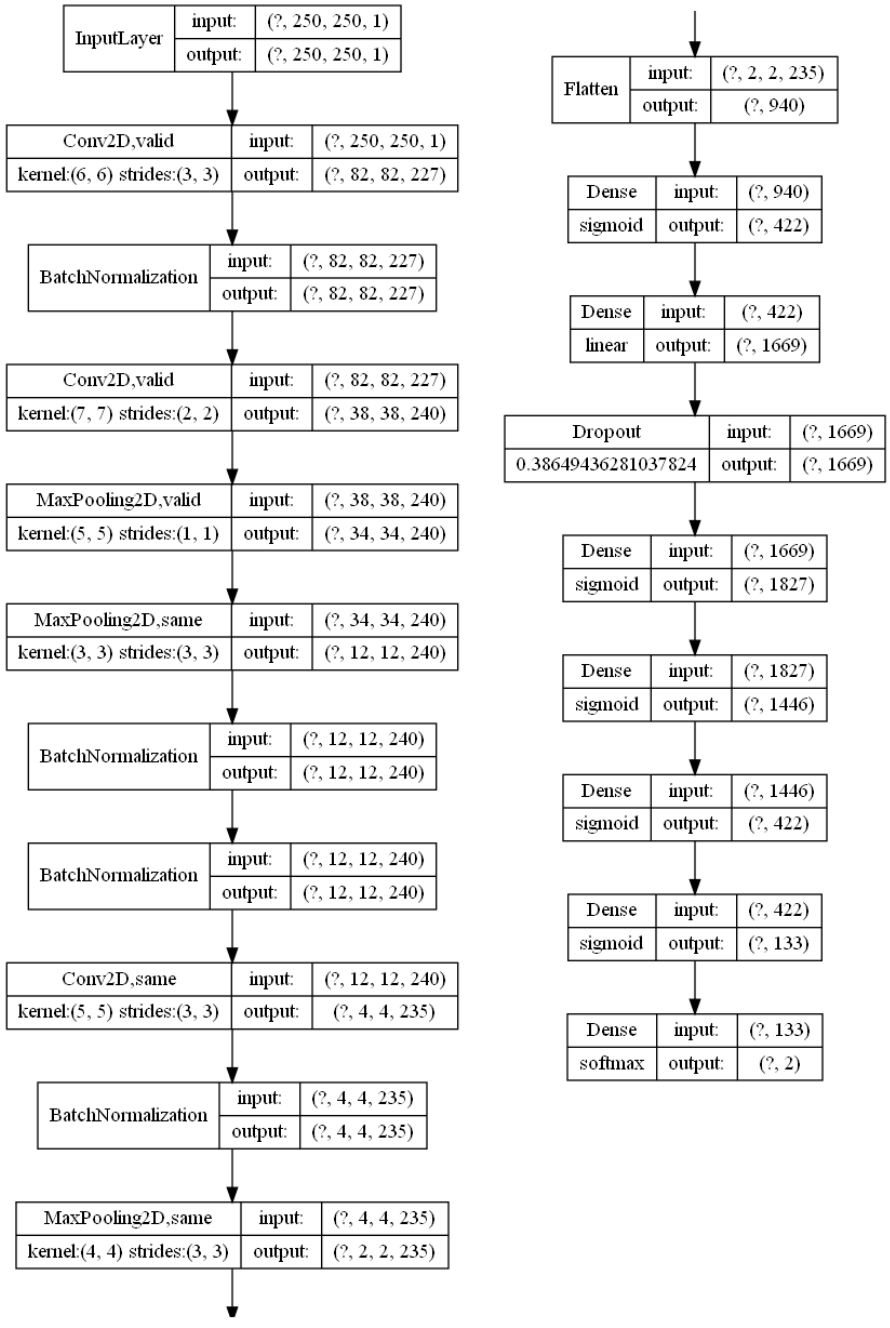


Figure B.5: Topology of the fittest CNN evolved by DENSER in the 300 generations study with 250x250 new cropped images.

B.2.3 Evolution Analysis for 300 generations with Data Augmentation

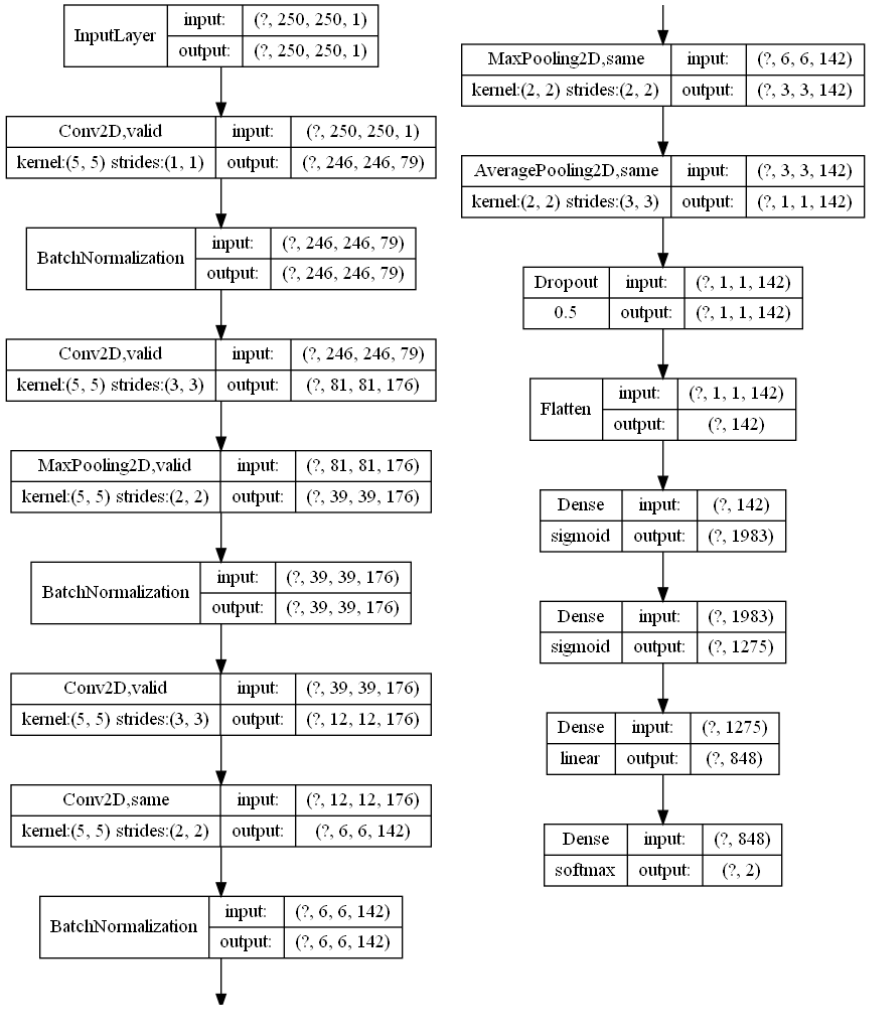


Figure B.6: Topology of the fittest CNN evolved by DENSER in the 300 generations applying data augmentation study using the 250x250 new cropped.

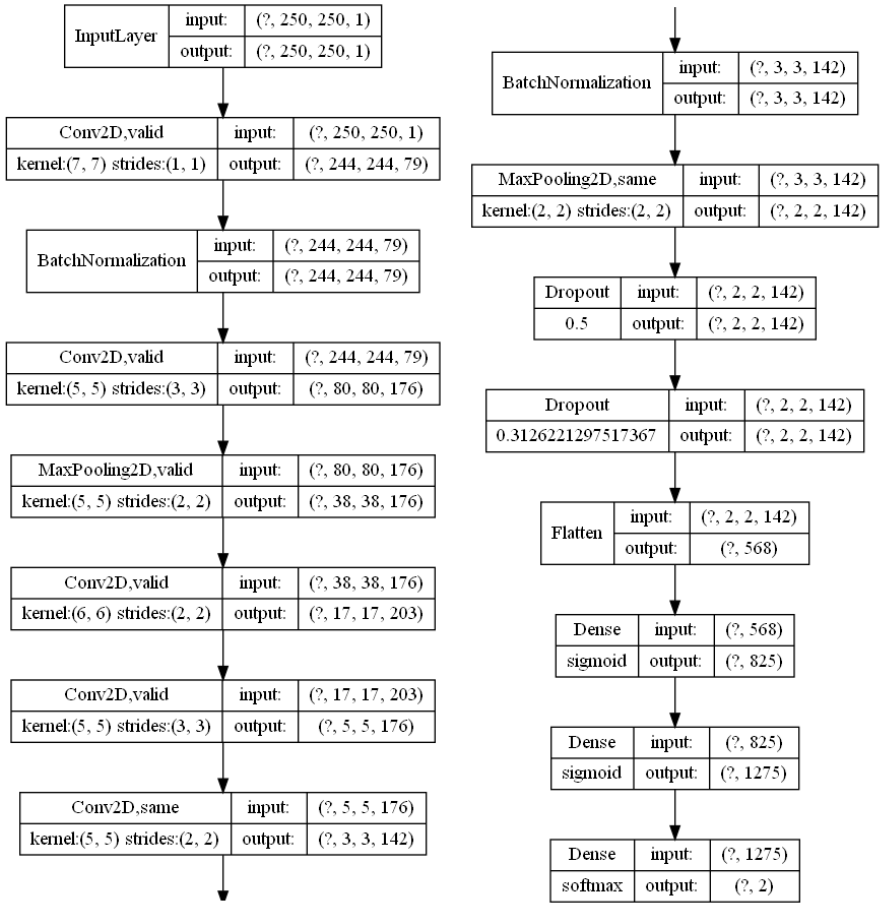


Figure B.7: Topology of the CNN with the best performance evolved by DENSER in the 300 generations applying data augmentation study using the 250x250 new cropped images.

ANNEX C

SALIENCY MAPS

Saliency maps are a visualization tool to facilitate model interpretability. A more accurate description of the process is given in Section 3.2.4.
This approach was tested in 50 images of Sections 4.1 and 4.2 test set. In the first one, only the best network acquired was investigated and in the other, the best performance network (Network A) and the best network acquired by DENSER (Network B) were investigated.

C.1 Section 4.1 Saliency Maps

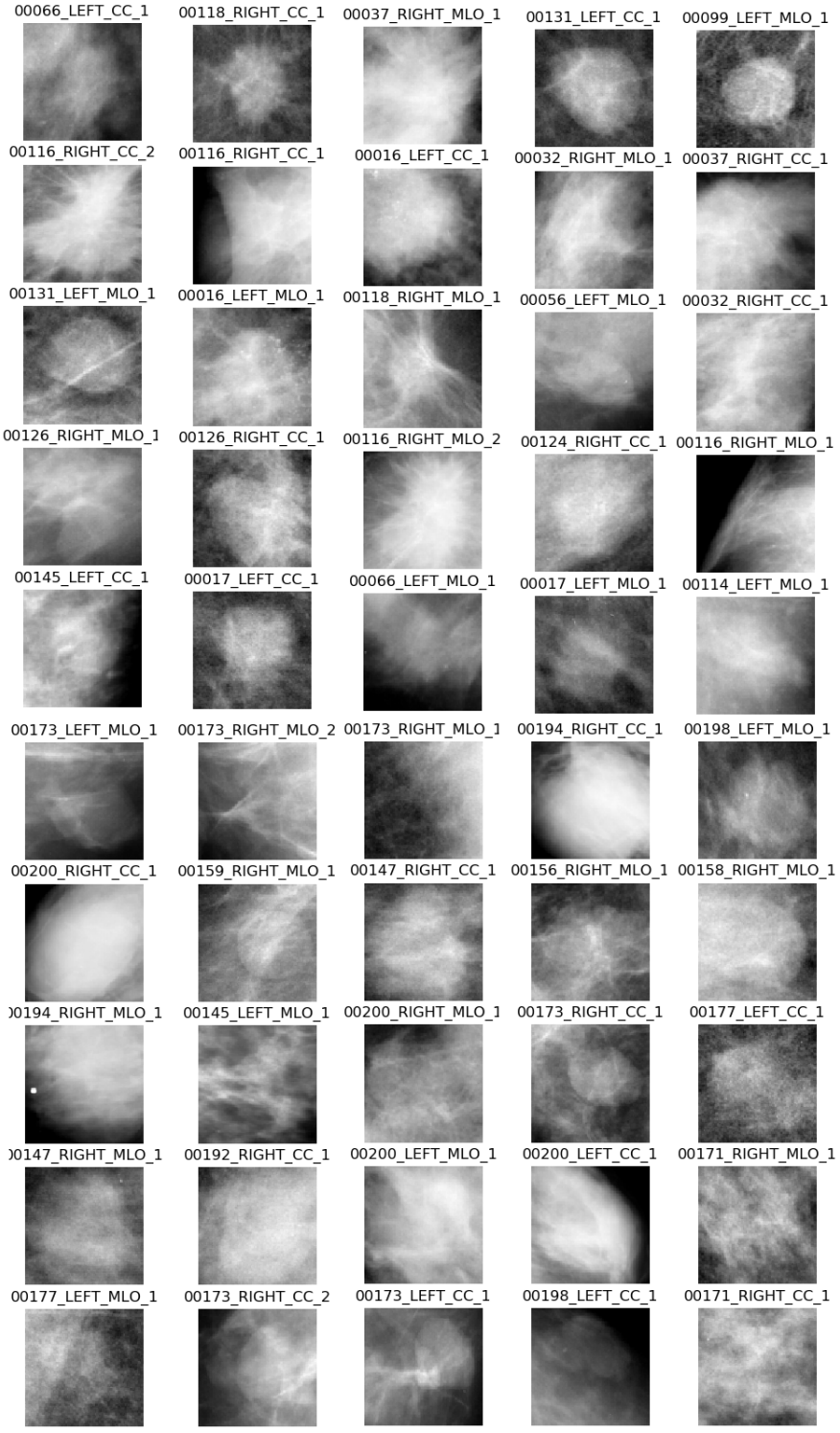


Figure C.1: 50 cropped images of the test set for the 90x90 dataset

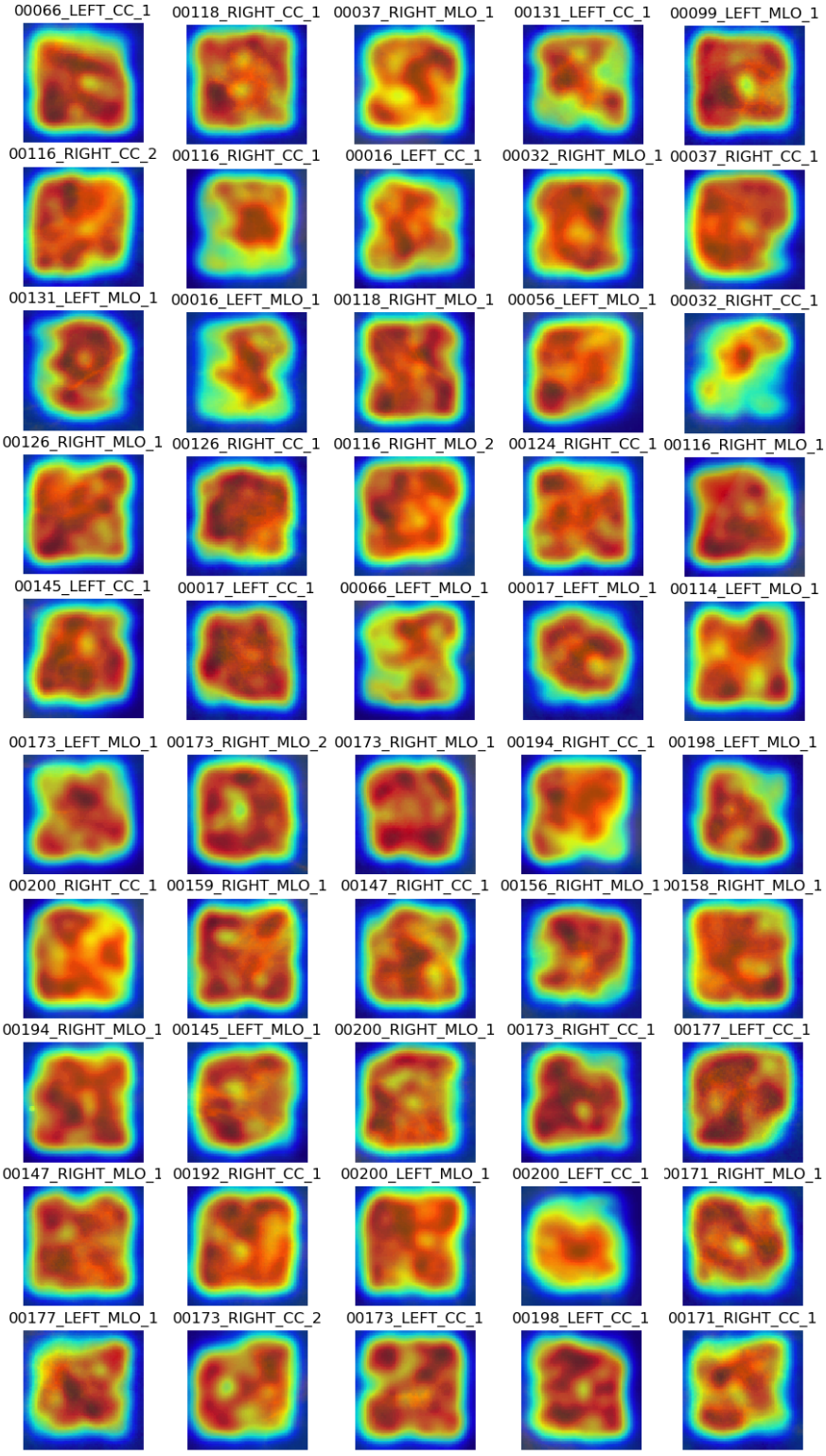


Figure C.2: Saliency maps of 50 images of the test set resorting to the best network acquired and chosen by DENSER in Section 4.1.

C.2 Section 4.2 Saliency Maps

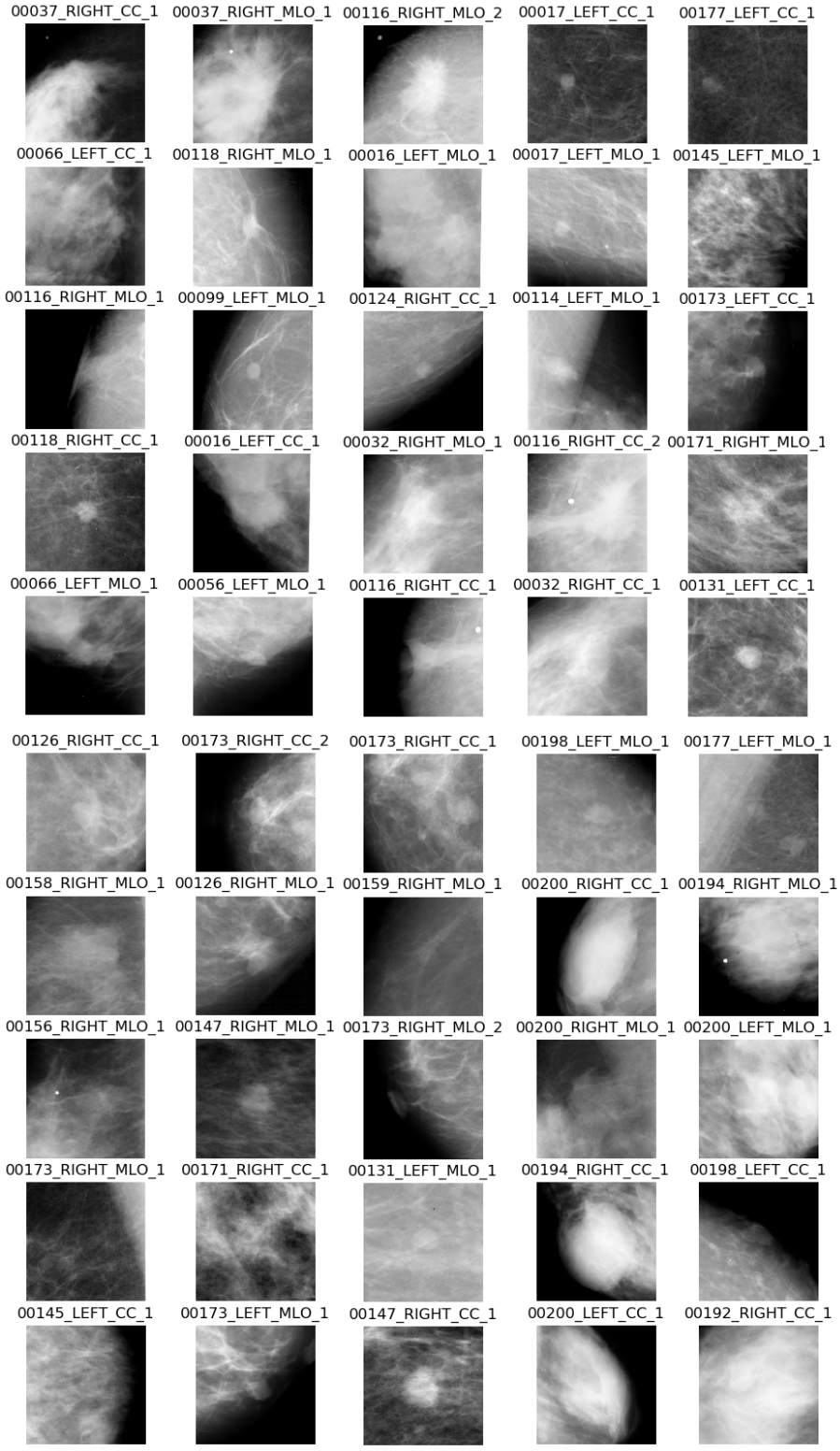


Figure C.3: 50 cropped images of the test set for the 250x250 dataset.

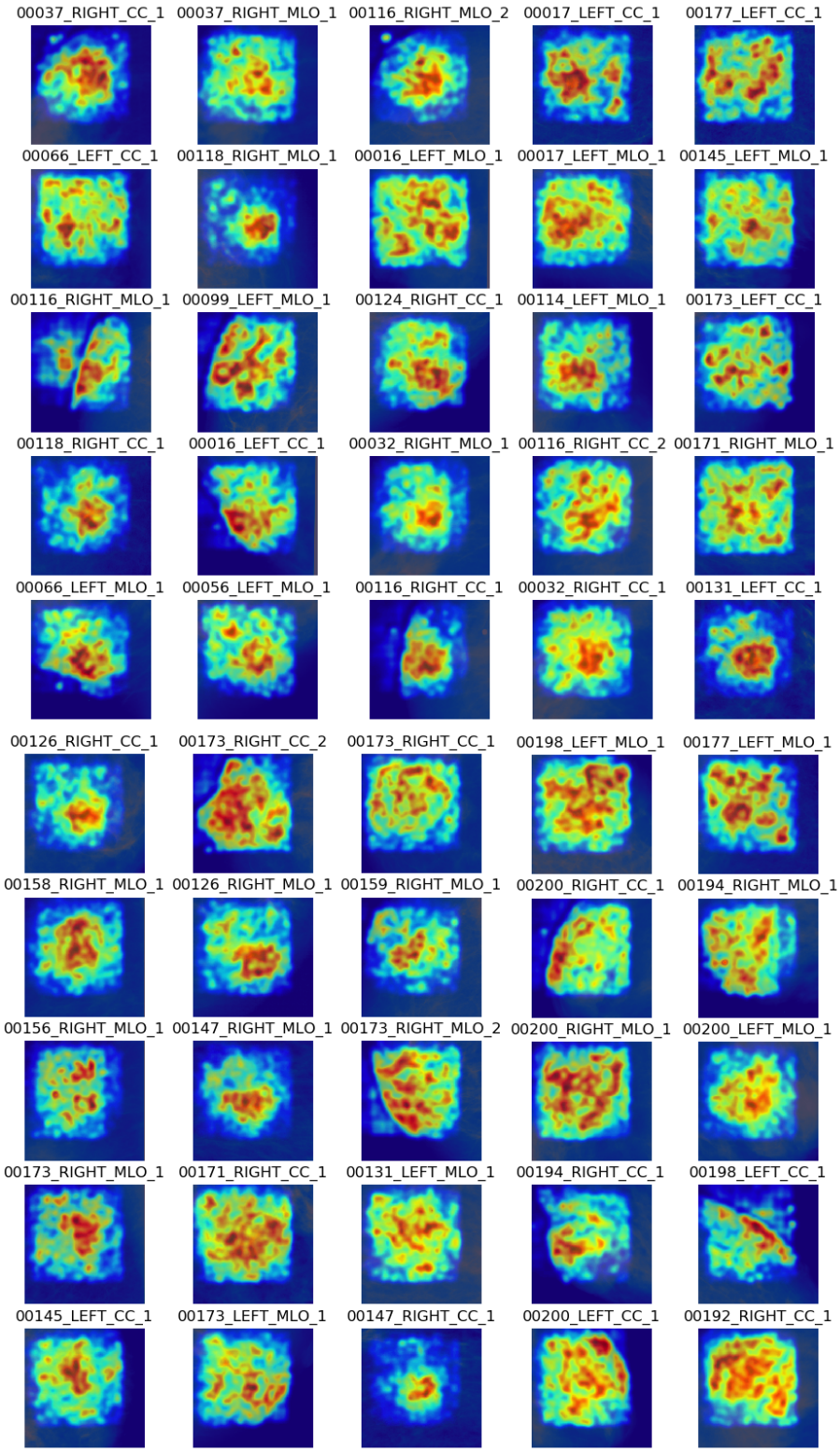


Figure C.4: Saliency maps of 50 images of the test set resorting to the network with the best performance (Network A) in Section 4.2.

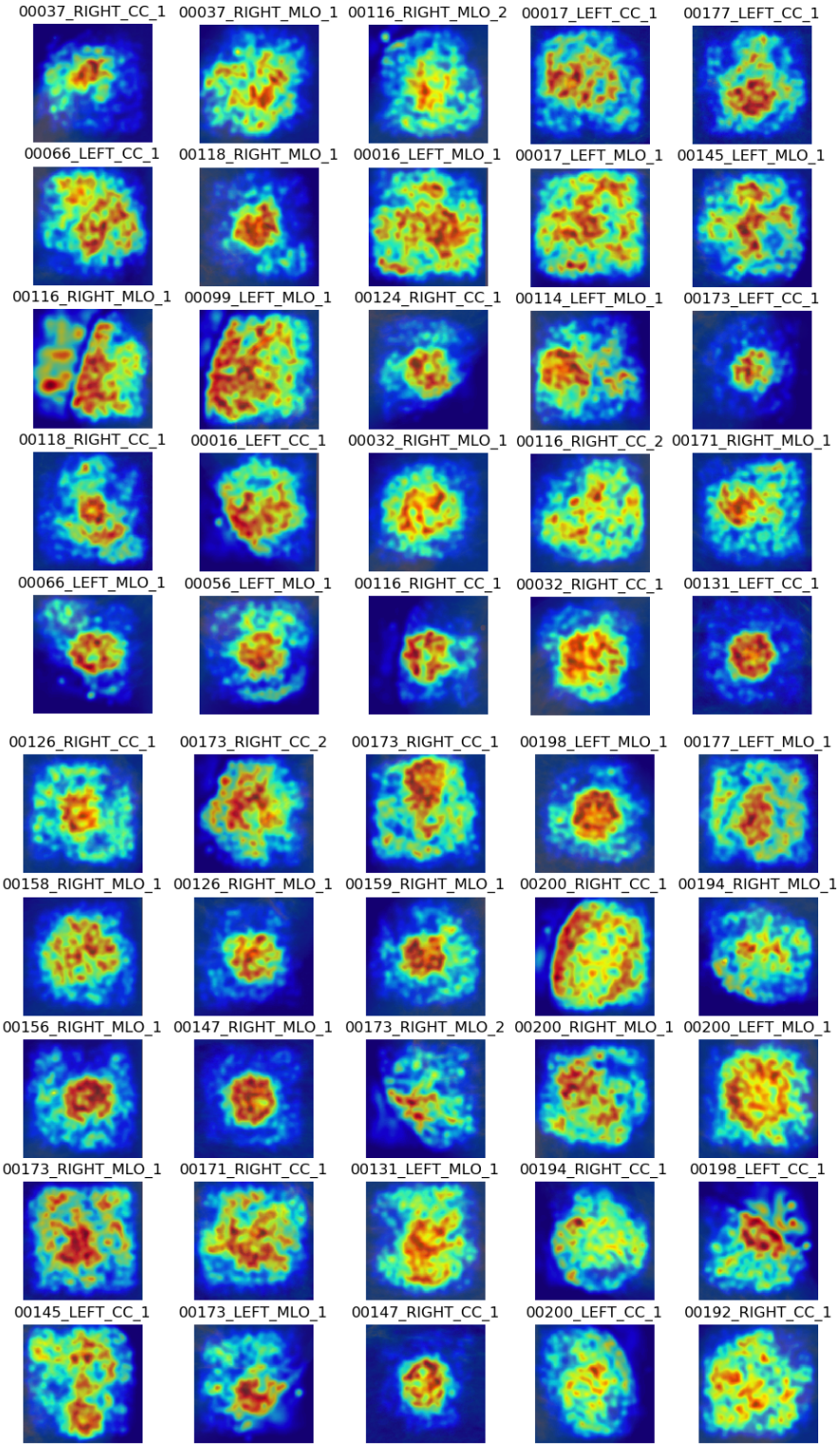


Figure C.5: Saliency maps of 50 images of the test set resorting to the best network acquired and chosen by DENSER in Section 4.2.