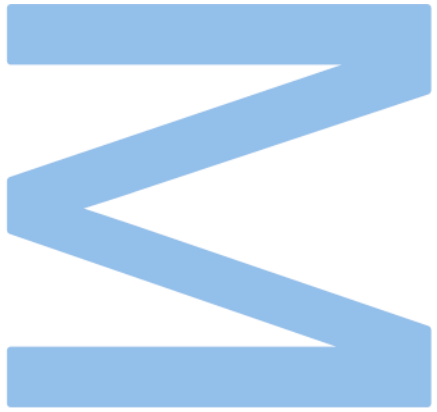
Automatic contrast generation from contrastless CTs



Rúben André Dias Domingues

Mestrado em Engenharia de Redes e Sistemas Informáticos

Departamento de Ciência de Computadores

2022

Orientador

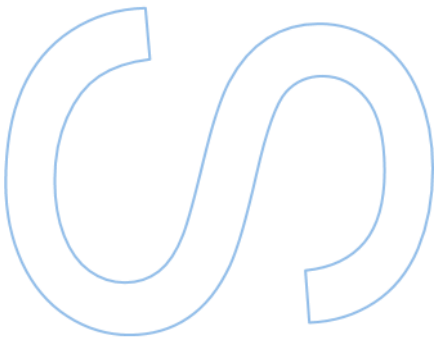
Francesco Renna, Professor Auxiliar Convidado, Faculdade de Ciências da Universidade do Porto

Coorientador

João Manuel Pedrosa, Professor Auxiliar Convidado, Faculdade de Engenharia da Universidade do Porto

Supervisor

Carlos Alexandre Nunes Ferreira, Investigador, INESC TEC




U.PORTO
FACULDADE DE CIÊNCIAS
UNIVERSIDADE DO PORTO

Declaração de Honra

Eu, Rúben André Dias Domingues, inscrito no Mestrado em Engenharia de Redes e Sistemas Informáticos da Faculdade de Ciências da Universidade do Porto declaro, nos termos do disposto na alínea a) do artigo 14.º do Código Ético de Conduta Académica da U.Porto, que o conteúdo da presente dissertação reflete as perspetivas, o trabalho de investigação e as minhas interpretações no momento da sua entrega.

Ao entregar esta dissertação, declaro, ainda, que a mesma é resultado do meu próprio trabalho de investigação e contém contributos que não foram utilizados previamente noutros trabalhos apresentados a esta ou outra instituição.

Mais declaro que todas as referências a outros autores respeitam escrupulosamente as regras da atribuição, encontrando-se devidamente citadas no corpo do texto e identificadas na secção de referências bibliográficas. Não são divulgados na presente dissertação quaisquer conteúdos cuja reprodução esteja vedada por direitos de autor.

Tenho consciência de que a prática de plágio e auto-plágio constitui um ilícito académico.

Rúben André Dias Domingues

Porto, 30 de Setembro de 2022

Abstract

Coronary Artery Disease (CAD) is a disease that occurs when the coronary artery is blocked by plaque, reducing the blood flow to the heart muscles. This disease may surge hereditary or by having high cholesterol and diabetes. This disease can be detected using computed tomography. This method is a non-intrusive look at the interior structures. The detection of this disease is made by calculating the amount of calcium in the coronary arteries. However, sometimes this disease appears without the presence of calcium, but using the injection of intravenous contrast agents, we can assess the level of stenosis of the coronary arteries. The use of contrast-enhanced tomography brings not only high hospital costs but also exposes patients to significant radiation. The use of Deep learning methods in the area of medicine, is appearing as one of the most promising strategies to help detect diseases that are hard to identify using only human intervention. The automatic generation of contrast from non-contrast tomography can avoid these problems. This dissertation uses two specific models of Generative Adversial Network (GAN) for image generation: The Pix2Pix-GAN and the Cycle-GAN.

The main contribution of this dissertation is the comparison of the performance of both the Pix2Pix-GAN and the Cycle-GAN, and the exploration of the trade-off of using 2D, 2.5D and 3D inputs. We have also used two different types of generators and two different datasets that will be described in this dissertation. For the first dataset using only the Structural Similarity Index Measure (SSIM), Peak Signal-to-Noise Ratio (PSNR) and Mean-Square Error (MSE), it could be concluded that Pix2Pix-GAN second generator using 2D data achieved better results obtaining 0.503 SSIM, 16.661 dB PSNR and 0.023 MSE. However, when using the Dice metric to evaluate the fidelity of synthetically generated high contrast region in the image, the model with the best performance was the Cycle-GAN second generator using 3D data, achieving 0.435. For the second dataset, the model with the best SSIM was the Pix2Pix-GAN first generator using 2.5D data, achieving a value of 0.381. The best PSNR and MSE were obtained using the Cycle-GAN first generator using 3D data. This model also achieved the best DICE value with 0.599. The Pix2Pix-GAN obtained better results in the SSIM, as was mentioned before. However, visual analysis of the output shows significant blur in the generated images, which is not the case for the Cycle-GAN models. This behaviour can be captured by the evaluation of the Fréchet Inception Distance (FID), which represents a fundamental performance metric that is usually not considered by related works in the literature.

Resumo

Coronary Artery Disease (CAD) é uma doença que ocorre quando a artéria coronária é bloqueada por placa, reduzindo o fluxo sanguíneo para os músculos do coração. Esta doença pode surgir de forma hereditária ou por ter colesterol e diabetes elevados. Esta doença pode ser detectada utilizando a tomografia computorizada. Este método é um olhar não intrusivo sobre as estruturas interiores. A detecção desta doença é feita através do cálculo da quantidade de cálcio nas artérias coronárias. No entanto, por vezes esta doença aparece sem a presença de cálcio, mas utilizando a injecção de agentes de contraste intravenosos, podemos avaliar o nível de estenose das artérias coronárias. A utilização de tomografia com contraste traz não só custos hospitalares elevados, mas também expõe os pacientes a radiações significativas. A utilização de métodos de aprendizagem profunda na área da medicina, está a aparecer como uma das estratégias mais promissoras para ajudar a detectar doenças que são difíceis de identificar utilizando apenas intervenção humana. A geração automática de contraste a partir de tomografias sem contraste pode evitar estes problemas. Esta dissertação utiliza dois modelos específicos de Generative Adversial Network (GAN) para a geração de imagens: A Pix2Pix-GAN e a Cycle-GAN.

A principal contribuição desta dissertação é a comparação do desempenho tanto da Pix2Pix-GAN como da Cycle-GAN, e a exploração do trade-off da utilização de entradas 2D, 2.5D e 3D. Também utilizámos dois tipos diferentes de geradores e dois conjuntos de dados diferentes que serão descritos na presente dissertação. Para o primeiro conjunto de dados utilizando apenas os conjuntos de dados Structural Similarity Index Measure (SSIM), Peak Signal-to-Noise Ratio (PSNR) e Mean-Square Error (MSE), poder-se-ia concluir que o segundo gerador Pix2Pix-GAN, utilizando dados 2D, obteve melhores resultados obtendo 0,503 SSIM, 16,661 dB PSNR e 0,023 MSE. Contudo, ao utilizar a métrica Dice para avaliar a fidelidade da região de alto contraste gerada sinteticamente na imagem, o modelo com melhor desempenho foi o segundo gerador Cycle-GAN usando dados 3D, obtendo 0,435. Para o segundo conjunto de dados, o modelo com o melhor SSIM foi o primeiro gerador Pix2Pix-GAN, utilizando dados 2,5D, atingindo um valor de 0,381. Os melhores PSNR e MSE foram obtidos utilizando o primeiro gerador Cycle-GAN, utilizando dados 3D. Este modelo também alcançou o melhor valor DICE com 0,599. O Pix2Pix-GAN obteve melhores resultados no SSIM, como foi mencionado anteriormente. No entanto, a análise visual da saída mostra um desfoque significativo nas imagens geradas, o que não é o caso dos modelos Cycle-GAN. Este comportamento pode ser capturado pela avaliação do Fréchet Inception Distance (FID), que representa uma métrica de desempenho fundamental

que normalmente não é considerada por obras relacionadas na literatura.

Acknowledgments

I want to thank my supervisors Professor Francesco Renna, Professor João Pedrosa and Professor Carlos Ferreira, for all the support given because without them, this dissertation would not have been possible. I also want to thank my parents for their support during this project. Finally, I want to thank all my friends who also supported and helped in every way they could.

Contents

List of Tables

List of Figures

Acronyms

ANN Artifical Neural Networks

BN Batch Normalization

CAD Coronary Artery Disease

CDC Centers for Disease Control and Prevention

CNN Convolutional Neural Network

CT Computed Tomography

DDB Dilated-Densely-Block

DL Deep Learning

DUN Densely-UNet-Nested

FID Fréchet Inception Distance

GAM Global Attention Model

GAN Generative Adversial Network

HU Hounsfield units

ICR Implicit Contrast Radiomics

ML Machine Learning

MAE Mean Absolute Error

MSE Mean-Square Error

PSNR Peak Signal-to-Noise Ratio

RFEM Radiomics-Feature Extraction Model

RgCL Radiomics-Guided Connection Layer

SFEM Semantic Feature Extraction Model

SSIM Structural Similarity Index Measure

TD Transition Down

TU Transition Up

Chapter 1

Introduction

Nowadays, ischemic heart diseases [5] are the cause of a huge amount of deaths in the most developed countries [6]. This disease is generated by the formation of plaques that can clog the arteries that surges due to the accumulation of cholesterol. There are some techniques to help calculate the risk of some cardiac situations. For example, a technique to measure the quantity of calcium helps to calculate an estimate of coronary atherosclerosis, and leads to predicting the risk of coronary artery disease.

Lately, non-contrast cardiac Computed Tomography (CT) has been used to evaluate the presence of coronary calcium, however it has the disadvantage of not detecting if the arteries are clogged. This disadvantage was overcome with the use of contrast-enhanced cardiac computed tomography. This method uses contrast agents to raise the density of the tissues, leading to a better attenuation which results in a brighter image.

This new method comes with some risks, the injection of contrast agents can lead to renal toxicity causing renal failure or some renal diseases (contrast-induced nephropathy). Besides this, we need to keep in attention that some regions (blood lumen and blood-thrombus interface) can not be distinguished with normal contrast agents, leading to the use of intravenous contrast agents. These agents improve the luminal density, attenuation and the intrinsic contrast between the vascular tree and the other soft tissues nearby. Reducing the need of doing contrast cardiac computed tomography helps to reduce kidney problems that come with the use of contrast agents.

Deep Learning (DL) [7] is a branch of Machine Learning (ML) methods. This method is based on the human brain, being constructed of what we call neurons and creating all the connections between them, and it is called artificial neural networks. These networks are composed mainly of layers, these layers contain all the neurons, and there are different types of layers that serve different purposes that will be described in the following chapter. DL can be used for medical imaging, in order to segment some areas of the patient body as well as generate new information that helps understanding more what is going on with the patient.

DL has been highly used recently in the medical area [8], in order to improve not only the quality of the exams, but it also helps to retrieve information that is not clear to the human

eye. This type of method is capable of using previous information in order to generate new types of exams, and also reducing the risk of doing certain exams. DL methods can also be used to classify exams by their disease. After giving the models enough data for them to distinguish some types of diseases, we can use that model to reduce human error.

The study in this dissertation focuses on the use of DL methods to generate contrast CT from a non-contrast CT. This is a viable way to reduce the need of injecting contrast into the patients since the insertion of contrast agents may cause some discomfort and risk to the patient. Despite the developments made so far to solve this problem, some of them only focus on the segmentation of CT without the use of contrast agents, and the others focus on different areas of the patient. Our main target is the heart of the patient. This method has some limitations since we need both non-contrast and contrast images of the same patient for the training process. Those images are not perfectly aligned since the patient does some involuntary movements and his breathing is not the same. This results in errors in this training process that leads to miscalculation of contrast in some areas, so there is the need of a technique to align images based on his paired one using the position of the patient that comes in the medical files.

The whole community related to the CAD, namely the doctors and the patients, will benefit from the contributions in this dissertation, as we will analyze and implement a safe and efficient way to help detecting this disease resulting in a faster way to proceed to the treatment of the patient.

1.1 Main objectives

The following objectives were defined for the research work presented in this dissertation:

- To investigate and analyze existing methods in the literature that are concerned with producing contrast in non-contrast CT.
- To validate the possibility of using GAN-based methods to generate synthetic contrast CT from contrastless CT
- To create a dataset to train, validate and test the performance of different methods for contrast CT synthesis
- To analyze the trade-offs in the use of 2D, 2.5D, and 3D data for contrast CT generation
- To compare the performance and analyze the generalization ability of the different methods developed for contrast CT generation.

1.2 Dissertation layout

This dissertation is made of six chapters and they are organized as follows.

The key ideas required to comprehend the work are covered in Chapter 2, notably the principles of machine learning, deep learning, and the GAN approach.

A thorough examination of the state-of-the-art literature is provided in Chapter 3 to support the choices made. The methods for producing contrast using non-contrast images are highlighted, and the most pertinent research is described.

The general procedure for producing contrast CT utilizing non-contrast CT is described in Chapter 4. Two datasets, along with the pre-processing done on them, are also discussed in this chapter. This chapter describes the proposed architecture, including its key elements.

The metrics employed to assess the effectiveness of the developed models are presented in Chapter 5. The findings and the analysis that followed them are reported and analyzed in depth.

Finally, the conclusions made are reported in Chapter 6, along with possible lines of development that may be applied in future work.

Chapter 2

Background

In this Chapter, some fundamental background concepts of this document will be presented, such as a brief medical description of coronary artery disease, an introduction to ML and DL, what a GAN is and how it is constructed.

2.1 Coronary Artery Disease

CAD is the number one ranked cause of death in developing countries [6] with approximately 7.8 million deaths in 2020. Each year 659,000 people die from this type of disease in the USA, as informed by the Centers for Disease Control and Prevention (CDC).

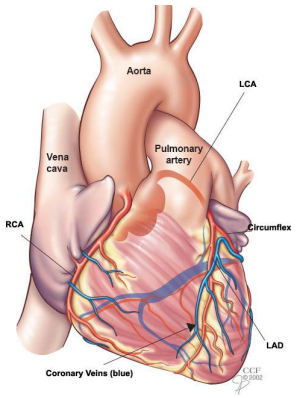


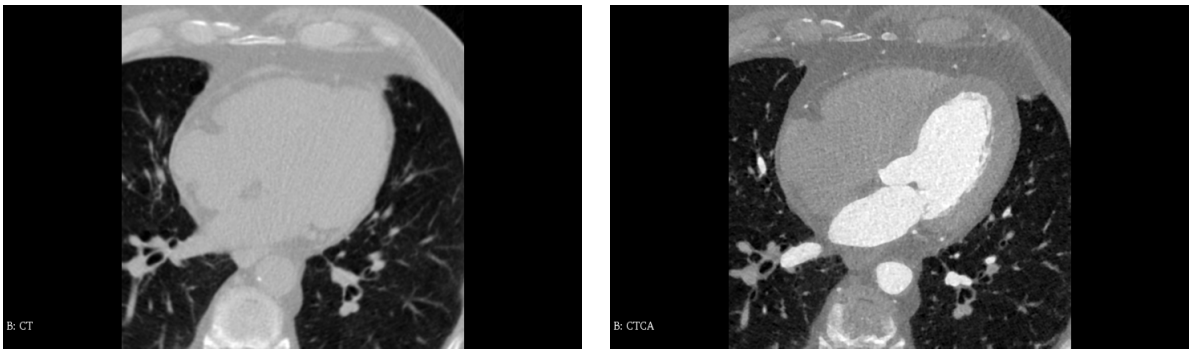
Figure 2.1: Front part of the Heart from [1]

.

CAD is the main focus of this dissertation. This disease [5], also known as ischemic heart disease, occurs when the coronary arteries are blocked by plaque (fatty material), reducing blood flow to the heart muscle and resulting in ischemia. These arteries (Figure 2.1) give blood-rich oxygen to our heart muscles and keeps it beating. This problem may result in heart attacks or chest discomfort. People can have this disease usually when they have high cholesterol, high blood pressure, diabetes, or the possibility that someone inside of their family has heart disease.

A common way to detect this disease is through the use of CT. This method is an existing non-intrusive look of interior structures. The principle of this method is to send x-rays or ionizing radiation through a current region of interest. Depending on the density of the tissue it passes through, the x-rays are attenuated during the same time they pass through it. When the attenuation is high, the CT gets brighter, and when it is low it gets darker.

Quantifying the amount of calcium using calcium scans, is an existing highly used method to detect this problem. These scans are obtained using an existing CT, the higher the amount of calcium detected, the higher the chances of having an existing heart attack, has it means that the artery is highly blocked. There is an existing problem with this method since some people have this disease without the presence of calcium. This happens more often in people below the middle age.



(a) Non-Contrast CT. (b) Contrast CT.

Figure 2.2: Computed tomography examples from the heart.

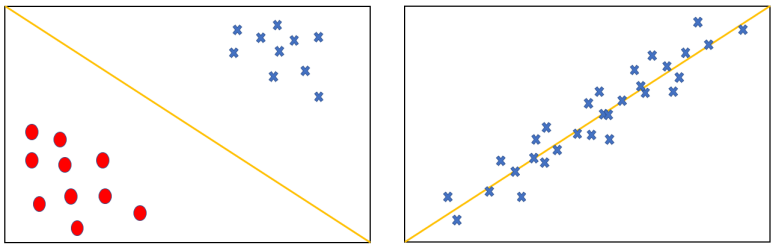
The contrast inside of the image (Figure 2.2(b)) is generated with the use of contrast agents increasing the attenuation of the near tissues. After examining the non-contrast CT (Figure 2.2(a)), on the possibility that is concluded the patient needs treatment, there is the necessity of a more detailed exam in order to comprehend how to proceed. This requires the use of an intravenous contrast agent, and this agent is used to increase the attenuation as well as the contrast between tissues and the artery. The use of this contrast also brings some downsides, given that some patients can be allergic to some agents, as some of them are based on iodine, and there are many patients with this allergy. The use of this agent requires needle insertion, and this may cause some discomfort to the patient as well as some skin irritation/damage. This agent is also nephrotoxic (poisonous or damaging to the kidney) and may cause acute kidney injury, meaning that your kidneys can stop working properly. This risk increases in older patients since they may have chronic renal disease, which means there is a high risk of complete kidney failure.

2.2 Machine Learning

ML [9] is a tool that allows the improvement of knowledge based on experience and existing data. This improvement can be seen as computer learning without the need for extensive programming

and outputting the expected result. ML automatic techniques can be divided into supervised, unsupervised, semi-supervised and reinforcement learning.

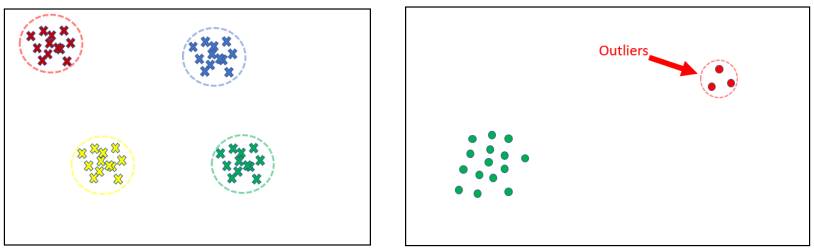
Supervised learning is one of the most used techniques. It involves the use of labelled data used in the training of the model so that afterwards, it can classify some unidentified data. In Supervised learning, we have classification methods that are used when there is a need to output a class from a set of predefined classes, as shown in Figure 2.3(a), and we have regression methods that are normally used when there is a need to output continuous values, as shown in Figure 2.3(b).



(a) Classification example. (b) Regression example.

Figure 2.3: Supervised methods.

Unsupervised learning involves data that does not have associated labels, and its objective is to obtain a description of the data. This technique involves finding patterns in the data without any knowledge of it. These patterns are used to either manipulate or understand the data, as we can see in Figure 2.4(a) or finding anomalies in the data like Figure 2.4(b).



(a) Cluster example. (b) Outlier example.

Figure 2.4: Unsupervised methods.

Semi-Supervised learning happens when we have a small amount of labelled data and a large amount of unlabeled data. This method is expected to produce better results than supervised learning, which uses only labelled data. As we can see in Figure 2.5, the black colour symbols are unlabeled data, and with the use of clustering, we can use the labelled data to label the clusters and make the boundary for the examples.

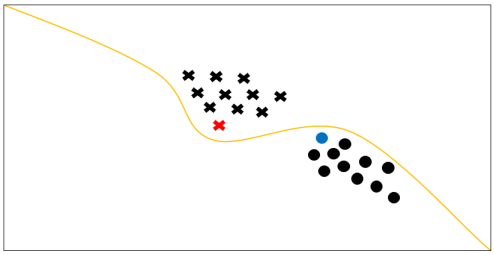


Figure 2.5: Semi-supervised learning structure example.

Reinforcement learning [10] is a technique that is based on the behaviour of an agent. This agent will take actions based on what it has learned so far, and these actions can reward or punish it, as we see in Figure 2.6, making its decisions sharper as it learns. This technique is used for resource management, personalized recommendations or even robotics.

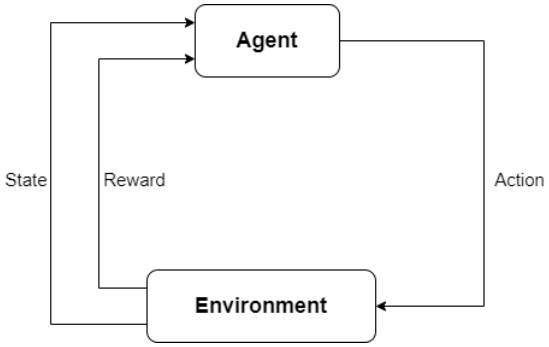


Figure 2.6: Reinforcement learning structure example.

2.3 Deep Learning

Machine learning includes a subset called DL [7]. These approaches do not need the user to provide features. Instead, they are based on a set of consecutive layers that find the features/patterns in data to make predictions or create the desired output. DL usually uses large datasets. This method makes it efficient and easier to interpret this amount of data. For the model to have a good prediction, it needs a wide amount of data.

2.3.1 Artificial Neural Networks

Artifical Neural Networks (ANN) are computer models that offer a mathematical model that takes its cues from the structure of intelligent animals. Simple processing units make up their structure, and these units store experimental knowledge. In order to obtain the desired outcome, the learning process modifies the network's weights by optimizing a given loss function.

These networks are made of neurons organized in different layers for specific purposes. The neurons in the **hidden layers** are in charge of processing data. As the purpose of hidden neurons

is to mediate between input and output layers, they have no outward contact, as we can see in Figure 2.7. The network may extract more complex statistics when additional layers are added. As seen in Figure 2.7 we have the **input layer**, receiving the input that the hidden layers will process. The **output layer** is the last layer used in a network and it contains the model predictions. Recently, artificial neural networks with a large amount of hidden layers have been dubbed deep neural networks and have been shown to achieve the state-of-the-art of different tasks in machine learning [11].

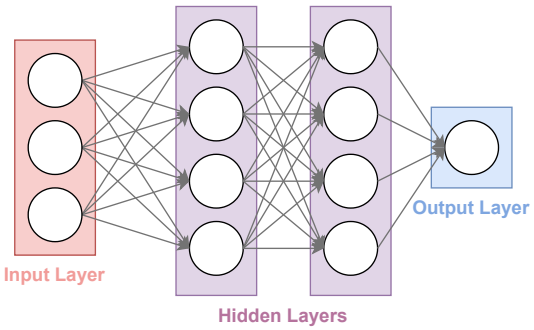


Figure 2.7: Artificial network example.

Both the input and output layers are non-trainable. In contrast, the hidden layers are trainable. After using a hidden layer, it is applied an **activation function**. This function is non-linear, which helps to detect non-linear features and decides what goes to the next neuron allowing us to see new patterns in data. There are multiple activation functions, some shown in Figure 2.8. The most used are, ReLu, LeakyReLu, sigmoid and tanh. The sigmoid function (Equation (2.1)) maps the inputs between 0 and 1 even if the values are negatives or large positives.

$$f(x) = \frac{1}{1 + e^{-x}} \tag{2.1}$$

In the intermediate layers, we use the activation function ReLU (Equation (2.2)), which assigns the zero value to the negative values of the output and replicates the positive values of the output of the previous layer identically.

$$f(x) = max(0, x) \tag{2.2}$$

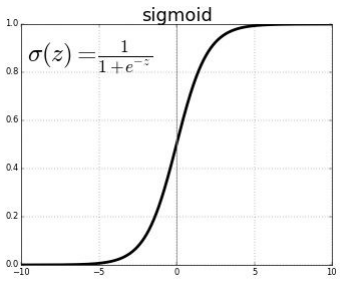
The LeakyReLu (Equation(2.3)) is similar to the ReLu function, but instead of transforming negative values to zero, it just reduces its extent.

$$f(x) = 1(x < 0)(\alpha x) + 1(x \geq 0)(x) \tag{2.3}$$

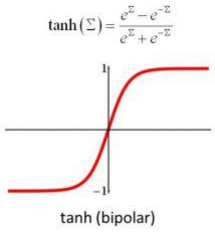
The tanh function (Equation (2.4)) maps the values between -1 and 1 and, because of that, results in greater gradient values that outcomes in an elevated number of weight updates in the

training phase.

$$f(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}} \tag{2.4}$$



(a) Sigmoid example.



(b) Tanh example.

Figure 2.8: Activation Functions from [2].

Loss Functions are functions that represent the error produced by the model in predicting the output values of training samples. The model focus on minimizing the loss to the minimum value possible. This results in a better prediction of the expected output.

This type of algorithm also has an **optimizer** that is used to change the attributes of the network, like the weights to minimize the loss and increase the accuracy. The hidden layers have weight matrices parameters that can be modified in the back-propagation. This modification is made by the optimizer chosen. **Learning rate** is a hyper-parameter that regulates how much the model adjusts each time the model weights are changed. If the value of the learning rate is high, that may cause the train to be unstable. On the other side, a low learning rate results in a more expensive time for training. This method also contains the **batch size**. This size represents the number of samples that will be used before the model parameters are refreshed. A high value of the batch size leads to an expensive memory cost, while a small value uses fewer memory resources. The number of **epochs** can also be defined. This number represents the amount of times the model has completed the training set.

During the training process, the model may suffer from two different problems, underfitting or overfitting. **Underfitting** occurs when the model can not have a good prediction of the training data, this may happen if the model is way too simple for the type of data. **Overfitting** occurs when the model predicts almost with perfection the training data, but when trying to predict new data can not perform that well. This may happen with a highly complex model for a simple task, the amount of data is low, or the data is not well distributed.

2.3.2 Convolutional Neural Network

The Convolutional Neural Network (CNN) [12] is a DL algorithm that can distinguish different features from an input image through the use of weights. This algorithm does not need complex

pre-processing in the input images because of its ability to learn their attributes and map them to obtain the most important of them.

CNN is composed of different types of components. First, we have the **convolutional layer**. This layer is used to learn the features from the input image, this type of layer has kernels (filters) that are combined with strides. The convolution layers consist of a dot product, where it shifts the input matrix by applying the kernel. These layers are not densely connected, meaning that not every input affects every output, and it has a much smaller number of weights in each layer. The strides basically move the filter around the image, and depending on its size, it can move, for example, one pixel at a time. It also has the number of filters in this layer. This filter is the number of neurons of that layer, also having the number of features map equal to the number of filters. This convolution layer is used when we want to downsample the input data. For upsampling, we use the **transposed convolution layer**, which basically reverts the spatial transformation of a normal convolution layer. In Equation(2.5) we can see the output size produced by the convolutional layer and in Equation(2.6) the output size of the transposed convolution layer. The i represents the input size, k the kernel size, s represents number of strides and p represents the padding.

$$\frac{(i + 2p - k)}{s} + 1 \tag{2.5}$$

$$(i - 1) * s + k - 2p \tag{2.6}$$

Pooling Layer is another type of layer used in CNNs. This layer focuses on lowering the feature maps resolution and obtaining shift-invariance. Shift-invariance occurs when a shift in the input signal independent variable results in a matching shift in the output signal. General usage is placed in the middle of two convolution layers, and feature maps of the pooling layer are linked to the feature maps of the previous convolution layer. The most two most common pooling layers are the **max pooling** and **average pooling**. The max pooling layer chooses the maximum value of a certain region of features covered by the filter. The average pooling instead chooses the average from each region.

2.3.3 Generative Adversarial Network

Generative Adversarial Network [13] are generative models, and their task is to produce new samples from a given distribution. These models are deep learning algorithms that are usually trained in an unsupervised manner using two models. Both these models learn from each other. The first one is called the generator and the second one is the discriminator. The generator is trained to generate new examples, while the discriminator is trained to evaluate that examples as real or fake. The objective of the generator is to fool the discriminator in order to produce fake examples that the discriminator classifies as true. The discriminator will have a set of the

real images and will compare the produced image to the ones on that set in order to classify them. In Figure 2.9, we can see what the function of the generator and the discriminator is.

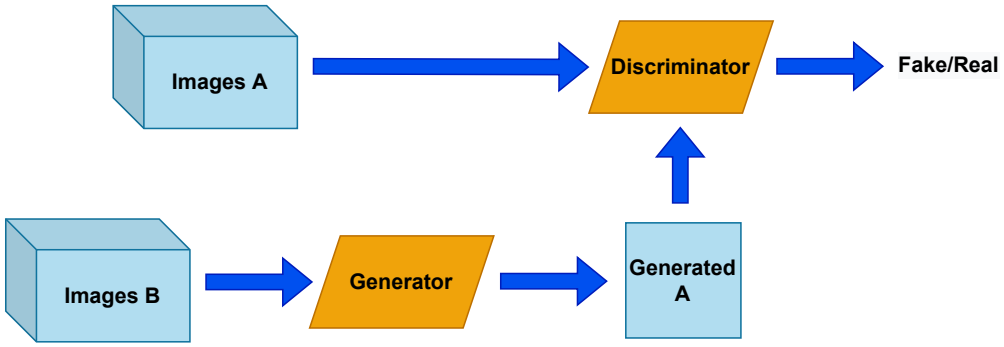


Figure 2.9: GAN scheme.

Every type of GAN can have two different types of loss functions, one for the generator and one for the discriminator. These losses are used when the generator is trying to fool the discriminator into improving himself. The generator mixes images that it produced with original images and sends them to the discriminator. The discriminator will then classify them as either real or fake. After this, the generator will update its weights with the measured loss obtained from the discriminator. The generator will generate new samples, and the loss it produced will be used by the discriminator in order to adjust its weights as well.

The most generic GANs do the generation by receiving a random vector and transforming it into a synthesised image. In this dissertation, we are focusing on GANs used for image-to-image translation. The image-to-image translation is to have one image from a certain domain and transform it to a different domain, for example, transform images captured during the day into images captured in the night. The two types of GANs that will be used in this dissertation are the Pix2Pix-GAN and the Cycle-GAN. The details of these two types of GANs will be described in the following, after providing information regarding some common architectures used for the discriminator and generator.

2.3.3.1 Patch-GAN

The PatchGAN [14] provides a new approach to image classification. This approach allows the classification of smaller images that we call patches. After classifying each patch, it averages all the outputs provided and that results in the final prediction. This method provides better precision and efficiency compared to the whole image classification, being the most used the 70×70 patch classification because of its efficiency and performance. In Figure 2.10, we can see that a small patch of the original image is one vector in the prediction matrix.

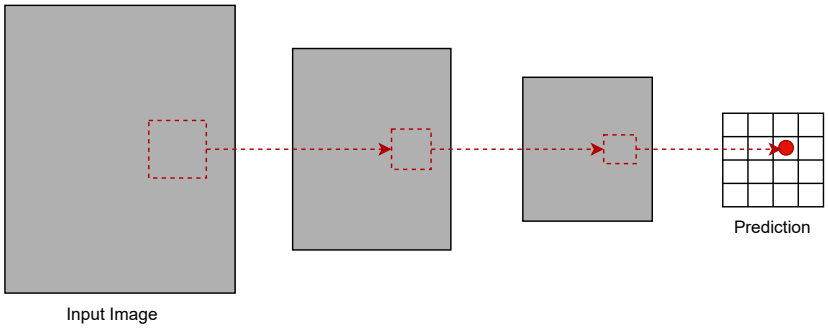


Figure 2.10: Patch-GAN scheme.

2.3.3.2 U-Net Generator

The U-Net [15] architecture is a semantic segmentation network that can also be used as a generator which is divided into an encoder and a decoder. The encoder uses convolution blocks along with a max pooling downsampling in the input image in order to extract the image features in multiple levels. These features are passed to the decoder that locates those features and upsamples them in a higher resolution allowing a dense classification. This decoder is composed mainly of upsampling and concatenation, having regular convolution operations. In Figure 2.11, we can see an example of a simple U-Net architecture.

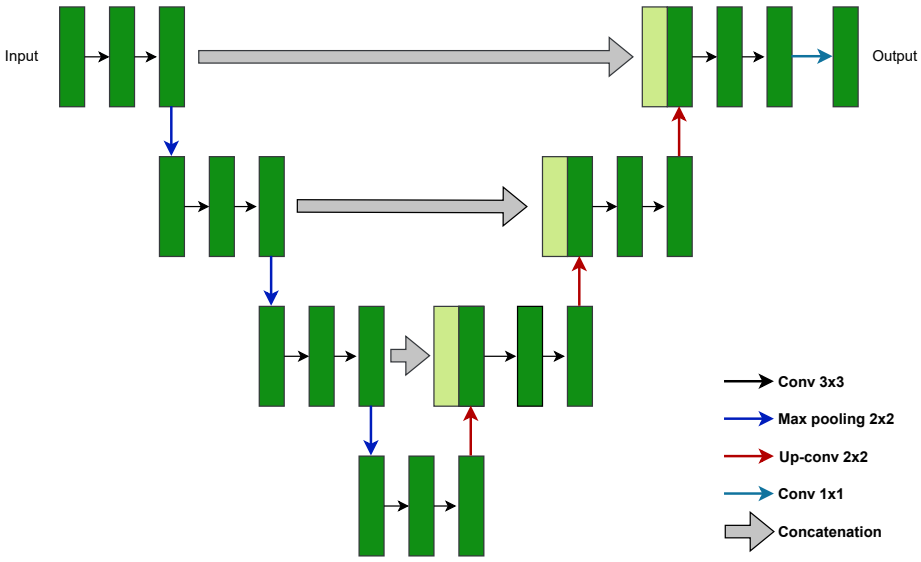


Figure 2.11: U-net architecture example

2.3.3.3 Pix2Pix-GAN

The Pix2Pix-GAN [16] is a deep learning method used in image-to-image translation. This type of GAN, in contrast to the Cycle-GAN, needs the use of paired data. This means that the data must be of the same spatial content, although with different characteristics. The requirement of

paired data forces a detailed analysis of the data, requiring the same amount of source images and target images during the training phase. This method is a type of conditional GAN that uses a conditional setup, meaning that the discriminator and generator depend on auxiliary data from different modalities, suggesting that contextual data is given to the model.

The architecture of this method, as we can see in Figure 2.12, is based on the use of only one generator and one discriminator. The difference between a generic GAN is that the discriminator needs the source image used to generate the output image in order to compare both and check if the output is a credible transformation of the source image.

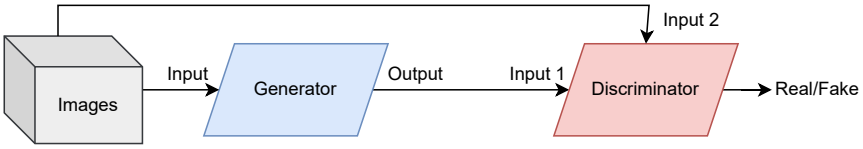


Figure 2.12: Pix2Pix-GAN scheme.

Through adversarial loss training, the generator is compelled to produce believable images in a particular domain. L1 loss, which is calculated between the created image and the desired output image, is another method of updating the generator and it is represented by the Equation (2.7), where $y_{true,i}$ is the i-th pixel of the real image and $y_{predicted,i}$ is the corresponding predict value.

$$\sum_{i} |y_{true,i} - y_{predicted,i}| \tag{2.7}$$

2.3.3.4 Cycle-GAN

The Cycle-GAN [17] is one of the deep learning methods used in image-to-image translation. This type of GAN provides a unique advantage against the other ones. It does not need paired data in order to produce the desired output. This means that, if we have two datasets A and B, we don't need to have the corresponding image from A on B. This method allows you to generate images from dataset B using A and also allows you to go back. This means that you can always reconstruct the original image from the generated one. This is called cycle consistency.

The architecture for this method basically consists of the use of both a generator and discriminator that allow you to generate new sets of images and another generator and discriminator that allows you to revert the image to the original one. The discriminators will receive two types of images during the training, some real images that allow them to later distinguish the fake images that we will provide in order to fool it.

In this type of GAN, creating the called "cycle", as we can see in Figure 2.13, is obtained by feeding the image generated from one generator to the other one. This means we generate an image from the first generator and feed it to the "reverse" generator in order to obtain the original one, and we compare both the image obtained from the "reverse" generator and the

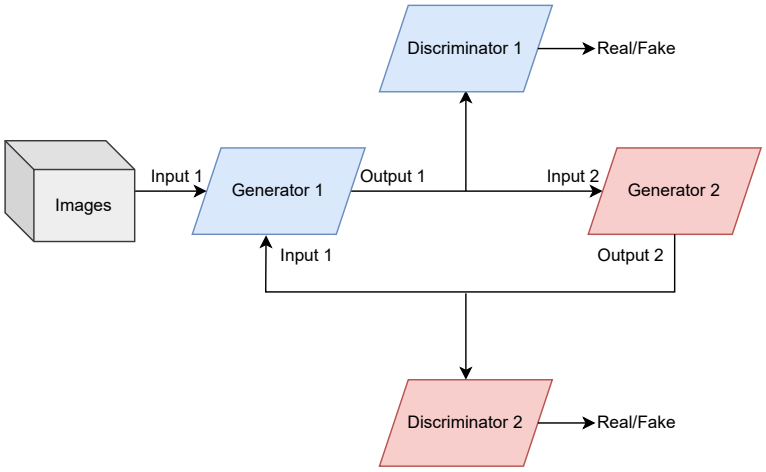


Figure 2.13: Cycle-GAN scheme.

original one in order to see the if they are really similar.

The first discriminator will classify Output 1 as either real or fake, and independently of the classification, the image will be fed to the second generator in order to reconstruct the original image. That reconstruction will then be classified by the second discriminator as real or fake as well.

Both the Pix2Pix-GAN and Cycle-GAN discriminators and generators will be trained in the adversarial zero-sum process. This means that for one model to get better, the other one needs to get worse. In this case, the objective of both models is to minimize their loss in order to improve each other.

Chapter 3

Literature review

Considering that CAD is the most common type of heart disease with a huge impact on our society, there are already algorithms applied in this area to detect it and check its development. This chapter starts with the fundamentals of the generation of contrast CT using non-contrast CT, also describing methods already implemented in order to create a new technique to generate contrast CT with more efficiency and precision.

Chandrashekar et al. [3] proposed a network (Figure 3.1) with a generator and discriminator that are simple neural networks, more explicitly a least-squares GAN and a PatchGAN. This framework (Cycle-GAN) removes the necessity of direct pair of specific samples while learning transformations of two distributions.

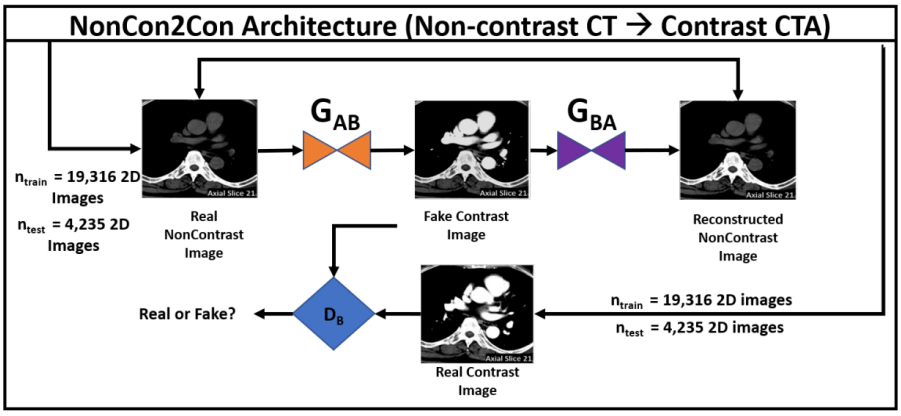


Figure 3.1: Cycle-GAN architecture from [3].

Least-squares GAN [18] is an improvement relative to the normal GAN because instead of using the sigmoid loss function, it uses the least-squares function. The sigmoid function has two main problems, vanishing gradient and saturation. In the vanishing gradient, the gradient starts getting closer to the value zero, which reflects almost no changes in the weights, resulting at the end of learning. The saturation happens when the function results in higher values where there is no more room for improvement. One way to solve both these problems is to not have all the activation functions using sigmoid. The least-squares function (Equation 3.1 and Equation 3.2)

prevents both the vanishing gradient and the saturation during training because it does not cause a small derivate. In this equations the a is the label for fake data and b the label for real data. The c is the value the generator wants the discriminator to believe for fake data. The x represents the input data and z is input variables sampled from uniform or Gaussian distribution.

$$\min_{\mathbf{D}} V_{LSGAN}(D) = \frac{1}{2} E_{x \sim Pdata(x)} \left[(D(x) - b)^2 \right] + \frac{1}{2} E_{z \sim Pz(z)} \left[(D(G(z)) - a)^2 \right] \tag{3.1}$$

$$\min_{\mathbf{G}} V_{LSGAN}(G) = \frac{1}{2} E_{z \sim Pz(z)} \left[(D(G(z)) - c)^2 \right] \tag{3.2}$$

The PatchGAN in this case classifies 70×70 size patches as real or fake. Twenty-six patients were selected with paired non-contrast and contrast images for this approach. This framework obtained low values of loss both in the generator and discriminator, having no values about the accuracy of the same.

Choi et al. [19] proposed the use of the Pix2Pix-GAN constituted by a U-Net architecture in the generator and a 70×70 PatchGAN as the discriminator, changing all the 2D layers by 3D layers. The PatchGAN used in this approach was similar to the one above.

The generator is a U-Net architecture. During the training phase, the chosen optimizer is Adam, that can handle sparse gradients on noisy problems. It also uses the L1 loss as the loss function. (Equation 2.7), this loss is defined when the model is being compiled, using the Mean Absolute Error (MAE) loss.

This approach used images from two different hospitals. The first hospital had twenty-five patients, and the second one had forty-two patients. To evaluate their performance, it was used the SSIM metric. This metric compares the generated contrast image with the original contrast image and returns the similarity between both. In the testing sets, it was obtained an average of 81 out of 100 is a good result for a small data size.

Kim et al. [20], and Woo et al. [21] proposed the use of the Conditional-GAN for this problem. Each approach has some minor changes in the base architecture of this algorithm. Both approaches used the same architecture for the generator, a U-Net, and differed in the choice of the function used in the discriminator. The discriminator in both architectures is a CNN classifier. This architecture has convolution layers that extract features from the input, mapping them. After this extraction, it proceeds to the downsampling of these features having layers that prevent overfitting by reducing the spatial size of the mapping of the features. Finally, it flats and transforms the data into a one-dimensional array giving this information to the fully connected layers to predict the output. One of the approaches[21] uses the minimax function (Equation 3.3) on the discriminator, and we have no information about the function used in the other, assuming a generic function. In this equation the x refers to sampled data from actual data distribution and the z refers to the data from the Gaussian distribution that is random noise.

$$\min_{\mathbf{G}} \max_{\mathbf{D}} V(D,G) = E_{x \sim Pdata(x))} [\log D(x))] + E_{z \sim Pz(z))} [\log(1 - D(z))] \tag{3.3}$$

The approach using the minimax function used 1560 slice images, obtaining a SSIM value of 95% in the training phase using 1170 slices and 90% in the test phase with 390 slices. These results were practically the same as the other approach, only having a 0.1% difference between the two using the same amount of slices both in training and testing.

Xiao et al. [4] proposed a framework based on the use of a Densely-UNet-Nested (DUN) segmentor and a radiomics-guided discriminator. These networks link with each other in order to improve the results obtained, and they have the objective of segmenting the lesions without the need for contrast agents. The DUN segmentor is composed of a total of twenty layers, as we can see in Figure 3.2. This segmentor can produce an improvement in the flow of data and give more details during this process, extracting Implicit Contrast Radiomics (ICR) features with more precision. These layers are the Dilated-Densely-Block (DDB), Transition Down (TD), Global and Transition Up (TU). The DDB is used to extract the features. Using a dilated convolution to expand the view of the image, we can extract a higher amount of features, having a huge impact on images with lower resolution. Every DDB layer uses a stride equal to 2, which means it will shift the pixels by 2 using a filter. It also uses dilation equal to 2. This means it expands the pixels by 2. Each dense block connects the layers with its successive layers using the density method. This resolves the problem of having to learn the features over and over again. After each DDB layer, we also have a TD layer, which consists of a group of operations, which are Conv, Batch Normalization (BN), ReLU and Pooling. The TU layer is the same group of operations but in reverse order. This segmentor receives as input the raw image with a resolution of 258x258 and outputs the segmented image with the features defined as most important with a lower resolution of 128x128.

The global layer uses a Global Attention Model (GAM) that helps the network by revealing pixel-level features in images with low contrast, improving the results obtained with the segmentation. This is obtained using two convolution layers, using the ReLU activation function, the first one has half the size of the original image, and the second one has the size of one, in order to multiply the input map with the attention map. Then it merges local features with attention features providing guided information.

The radiomics-guided discriminator uses the following layers to detect if the image is real or not:

- Semantic Feature Extraction Model (SFEM)
- Radiomics-Feature Extraction Model (RFEM)
- Radiomics-Guided Connection Layer (RgCL)

The SFEM consists of the use of five convolution layers and three fully connected that

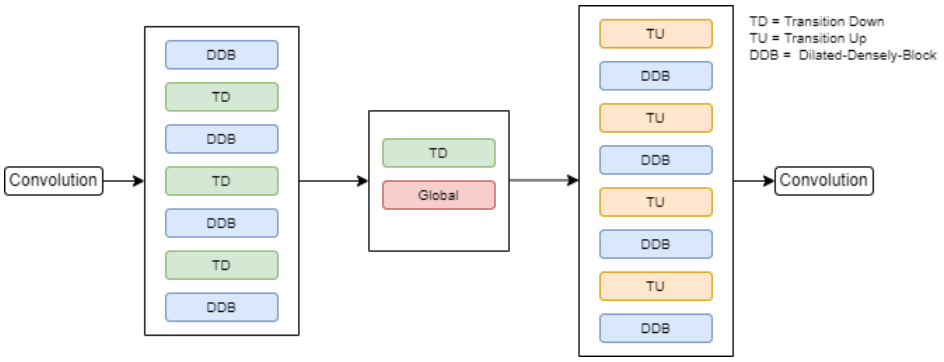


Figure 3.2: Adapted DUN segmentor strucutre from [4].

maximize the average across training cases of the logarithmic probability of the true label under the distribution of the prediction.

The fully connected layers are the layers where we apply a linear transformation in the input vector using a weights matrix. After this step, we apply a non-linear activation function. This means that every input affects every output.

The RFEM describes the diverseness and the microstructure of possible lesions. In need to extract and select these features, the open-source platform-PyRadiomics. This platform uses multiple algorithms to extract radiological features from medical data, allowing for the extraction of shapes and textures. It selects the 75 most relevant features and analyses them.

The RgCL is the connection layer that allows the union of the features obtained in the previous layer. This result is processed by the Softmax function computing the output, which can be fake or real. The loss function used in the training process is shown in Equation (3.4), where the y represents the segmentation ground truth and the x the predicted segmentation.

$$\min_{\mathbf{S}} \max_{\mathbf{D}} L_{GAN}(S, D) = E_{y \sim Pdata(y)} [\log D(y)] + E_{x \sim Px(x)} [\log(1 - D(s(x)))] \tag{3.4}$$

Finally, the data used in this framework was given by *McGill University Health Centre*, containing 250 participants. These participants are divided into 130 with hemangioma and 120 with hepatocellular carcinoma. This framework obtained 96.23 precision and 91.79 recall, being one of the best algorithms in this area.

3.1 Discussion

This search focused on deep learning applied to the generation of synthetic contrast on computed tomography. After understanding the existing approaches and their results, we decided to focus only on Cycle-GAN and Pix2Pix-GAN, trying to understand their limitations and how they can be overcome since both obtained the lowest performance. The reason for their low performance may be the small amount of data for training and testing or his distribution since the other

approaches used ten times more data, which can give better diversity, improving the algorithm capacity of generating contrast. In Table 3.1 we can see a small summary of the performance of each algorithm propose in this article.

Ref.	Objective	Algorithm	Data	Performance
[3]	A Deep learning Approach to Generate Contrast-Enhanced Computerised Tomography Angiography without the Use of Intravenous Contrast Agents	Cycle-Gan with least-squares loss generator and 70x70 PatchGan discriminator	26 subjects 23551 slices	Generator loss < 0.5 Discriminator loss < 0.3
[19]	Generating synthetic contrast enhancement from non-contrast chest computed tomography using a generative adversarial network	Pix2Pix-Gan with U-Net generator and 70x70 PatchGan discriminator	50 subjects	SSIM Mean: 82.51 PSNR Mean: 16.69
[20]	Contrast CT image generation model using CT image of PET/CT	Conditional-Gan with U-Net generator and CNN discriminator	1560 slices	SSIM: 0.9055
[21]	Generation of contrast enhanced computed tomography image using deep learning networks	Conditional-Gan with U-Net generator and CNN with minimax loss discriminator	1560 slices	SSIM: 0.90
[4]	Segmentation of liver lesions without contrast agents with Radiomics-guided Densely-UNet-Nested GAN	Radiomics-guided Densely-UNet-Nested GAN with a DUN generator and a Radiomics-guided discriminator	250 subjects	Accuracy: 96.23 Recall: 91.79

Table 3.1: Performance Summary.

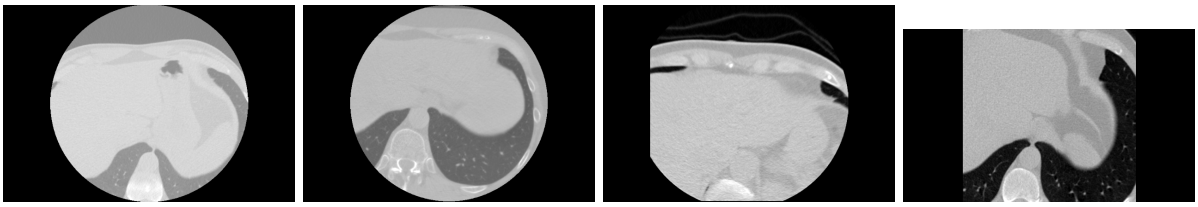
Chapter 4

Methods

This chapter aims are to consider different approaches based on GANs to synthesize contrast-enhanced CT from non-contrast CT. In particular, based on our literature review, we will be focusing our analysis on two specific families of models, the Cycle-GAN and Pix2Pix-GAN considering different setups for both.

4.1 Materials

One of the datasets used in this dissertation was obtained at the Orca Score in the Grand Challenge platform [22]. This data is already divided into training and testing, so there are thirty-two patients for training and thirty-two for testing, and each patient is also divided into their corresponding contrast and non-contrast CT. For each train and test set, there are four types of CT (different types of machines), as we can see in Figure 4.1.



(a) (b) (c) (d)

Figure 4.1: Different CT formats.

Each contrast and non-contrast information of each patient is found in pairs of mhd and raw formats that contain all the information of the CT. The mhd file works like a header file containing all the metadata from the image, whereas the zraw file contains all the compressed image data. This dataset slice thickness is divided into four groups, each group containing eight patients. The first group of patients had contrast images with a slice thickness of 0.6250mm and a non-contrast slice thickness of 2.5mm. The other three groups had a slice thickness of 3mm in non-contrast images, but they differed in the thickness in contrast images. The second group had

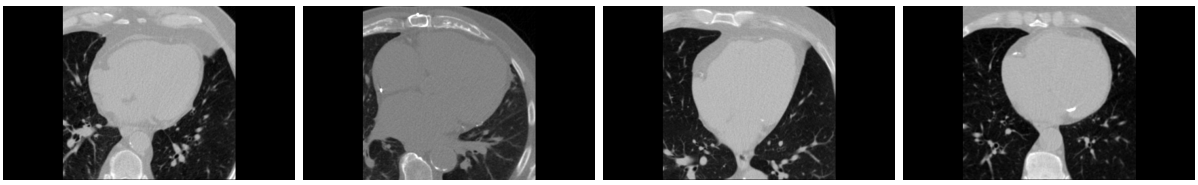
a slice thickness of 0.2500mm. On the other hand, the third group had a 0.4500mm thickness, and the fourth group had a slice thickness of 0.4000mm.

Using the tool Slicer [23], we converted all these files to the slices in dicom format. The dicom file consists of the image of a patient, and it may contain information about him. This file also includes information on the position of the patient and other metadata information. The final number of slices retrieved from Slicer can be observed in Table 4.1.

Folder	Nº Patients	Type of CT	NºSlices
Train	32	Contrast	10685
		Non-Contrast	1540
Test	32	Contrast	10614
		Non-Contrast	1551

Table 4.1: Orca Data Summary.

Our second dataset was provided by our project partners Centro Hospitalar de Vila Nova de Gaia. This dataset consisted of a total of thirty patients. This dataset was collected using the machine SIEMENS SOMATOM Force. The slice thickness used in the non-contrast CT is 3mm and for contrast CT 0.6mm. We can see some examples in Figure 4.2.



(a) (b) (c) (d)

Figure 4.2: Different CT examples.

Folder	Nº Patients	Type of CT	NºSlices
Train	25	Contrast	8118
		Non-Contrast	2541
Test	5	Contrast	1660
		Non-Contrast	502

Table 4.2: Centro Hospitalar de Vila Nova de Gaia Data Summary.

4.2 Data pre-processing

Before using these images in the considered model for this task, it is necessary to do some pre-processing in order to increase the model's overall quality. The first pre-processing done was the pairing of contrast and non-contrast CT. Since the number of images was mismatched

between contrast and non-contrast CT, it was important to filter the images in order to have one contrast CT for each non-contrast CT. There were two alignment pre-processing methods considered in this thesis: manual selection and image registration. The manual selection objective was to find the nearest contrast slice of each non-contrast slice, and we paired them with this. This selection was first made in the Orca Data because this dataset was used for most of the training models. Some non-contrast slices have been removed due to the fact that there was not any contrast slice nearest. As we can see in Table 4.3, the number of slices is now equal for both contrast and non-contrast CT but with a lower number of slices.

Folder	Nº Patients	Type of CT	NºSlices
Train	32	Contrast	1 406
		Non-Contrast	1 406
Test	32	Contrast	1 422
		Non-Contrast	1 422

Table 4.3: Orca Dataset Paired Filtration Summary.

For the second dataset we used an image registration technique in order to maximize the number of images that we had. Registration is a technique for geometrically aligning pictures for analysis that combines multiple images from different imaging devices or sensors obtained at different times or positions. In this case, since clinical images have information about the position and angle of the patient, we can use this information in order to obtain more similarities between non-contrast and contrast images. This was only applied to the second dataset, maintaining the manual selection for the first dataset.

The registration was made by using the nrrd files with the most zoom, this means, using the images with the closer view of the hearth. After reading these files, we got the images spatial coordinates that will be used to find the boundaries between the non-contrast and contrast slices. Before finding the boundaries, we first used the non-contrast image coordinates to find the nearest contrast image. After having the corresponding contrast image, we proceed to the boundary delineation. This boundary was divided into finding both the maximum and the minimum coordinates. For the minimum coordinates, we first found the minimum for each slice and then we chose the highest one. For the maximum coordinates, we found the maximum for each slice and then chose the lowest one. After the boundaries were found, we made the two interpolation functions, one function for the non-contrast image and one for the contrast image. This function used the RectBivariateSpline function from the spicy package that applies a bivariate spline over a rectangle, being this rectangle is the image that we are using. After this, we apply both the interpolation function to the respective images, obtaining the non-contrast and contrast slices aligned. Figure 4.3 shows an example of the application of this method. The first row represents the original images and the second row the images after the alignment. The contrast image suffer no alterations while the non-contrast image suffered alterations in the bottom right corner.

After using the registration, we obtained a elevated number of images keeping up the paired

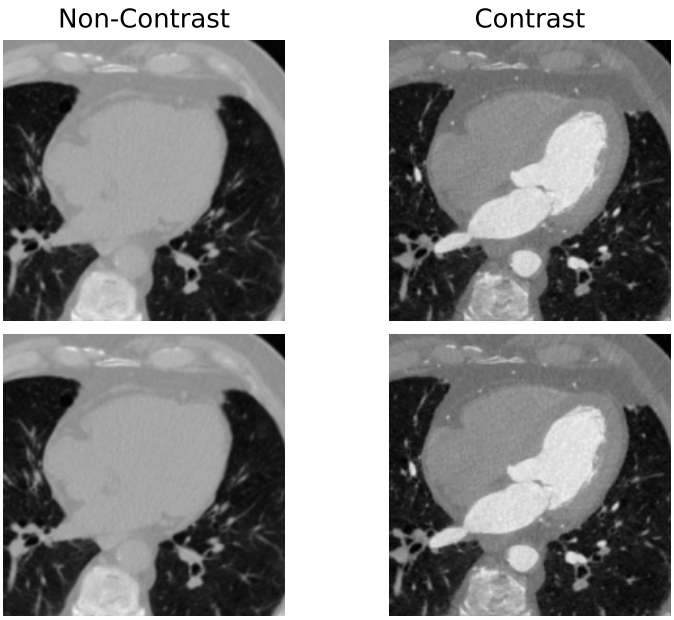


Figure 4.3: Registration application example.

contrast and non-contrast images. This happens because, in our manual segmentation, we discarded some slices because there wasn't any visual correspondence between the contrast and non-contrast image. In this method, we based on the non-contrast images, and for every non-contrast image, we found the contrast image with the most correlation with it. In Table 4.4 we can see the final number of slices we obtained for the training and testing set.

Folder	Nº Patients	Type of CT	NºSlices
Train	25	Contrast	2541
		Non-Contrast	2541
Test	5	Contrast	502
		Non-Contrast	502

Table 4.4: Centro Hospitalar de Vila Nova de Gaia Dataset Registration Summary.

The following pre-processing is the same for both datasets. The images from both datasets have the resolution 512×512, which means that the model we have more pixels to analyze and will result in more expensive memory requirements and also requires more time in the training phase. Reducing the image size, although it results in less expensive memory and time requirements, also means that we may lose some information. Still, we decided to reduce it to half the size, meaning that we will only use 256×256 resolution for all the images. This was done using the OpenCV package that provides the resize function. This function lets us decide which function we want to use for interpolation. The decided interpolation is the INTER_AREA[24]. This interpolation method uses the use of pixel area relation, being a good option for image shrinking.

Given that medical images have pixel values that can be higher than 255 or lower than 0, it is necessary to be sure that all the medical images are in the same range making them consistent with the ones from the Orca dataset. It was informed by the Centro Hospitalar de Vila Nova de Gaia that for non-contrast images, the pixel range was [-50, 350], and for the contrast images, it was [-150, 590]. This range is expressed in Hounsfield units (HU), this unit is obtained from a linear transformation of the measured attenuation coefficients being used in computed tomography.

After having our images with the same pixel range values as the ones used in the Orca dataset, it was necessary to convert that pixel values to a lower range in order to use them in our model. Since a higher range of pixel values results in a more expensive waste of resources and computing this high range may be a little more complex for the network. We chose to convert our pixel range to [-1, 1], and for this, there was implemented the function that uses a linear conversion to convert values to different ranges.

Finally, to prevent the need to do this pre-processing every time we want to use our datasets, it was decided that the best way to facilitate their usage was to save both datasets separately in the NumPy format (.npz). This format is a compressed way to save NumPy arrays, and since the images are in fact, NumPy arrays, we just need to create a main array that stores all the images. In our case, we have two main arrays for each dataset, one for non-contrast images and other for the contrast images. To save the arrays, we use the function savez_compressed from the NumPy package, and this offers us the option to save both arrays in the same file. To load the dataset, we just need to be sure to separate the arrays again in order to have the contrast and non-contrast images.

4.3 Model Architecture and Training

In this section we are going to explain the architecture used in the Cycle-GAN and the Pix2Pix-GAN that is adapted on the one implemented by Jun-Yan Zhu [17], the parameters used, how we created the designed GAN and the training process execution.

4.3.1 Discriminator

A PatchGAN convolutional classifier is used as the discriminator in the Pix2Pix-GAN and the Cycle-GAN. According to the Pix2Pix-GAN [14] and the Cycle-GAN [25] article, it makes an effort to categorize if each picture patch is legitimate or fake. Following the articles, we thought using the Instance Normalization for the Cycle-GAN and the Batch Normalization for the Pix2Pix-GAN.

The Batch Normalization [26] seeks to minimize the internal covariance shift, speeding up the training in the neural networks. It achieves this by fixing the means and variances of the layer inputs during the normalization stage, allowing the use of high learning parameters without

causing divergence. This type of normalization is applied to the batch size that is defined depending on the task and resources available. The Equations(4.1) are respectively the mean, variance and normalization. The T is the batch size, H and W are the spatial dimensions, t is the image in the batch, i is the feature channel, j and k are the span spatial dimensions.

$$\begin{aligned} \mu_i &= \frac{1}{HWT}\sum_{t=1}^{T}\sum_{l=1}^{W}\sum_{m=1}^{H} x_{tilm} \\ \sigma_i^2 &= \frac{1}{HWT}\sum_{t=1}^{T}\sum_{l=1}^{W}\sum_{m=1}^{H}(x_{tilm} - mu_i)^2 \\ y_{tijk} &= \frac{x_{tijk} - \mu_i}{\sqrt{\sigma_i^2 + \epsilon}} \end{aligned} \tag{4.1}$$

The Instance Normalization [27] works like the batch normalization but only works with one example independently of the batch size defined. The Equations(4.2) are respectively the mean, variance and normalization. The variables here represent the same as the ones above.

$$\begin{aligned} \mu_{ti} &= \frac{1}{HW}\sum_{l=1}^{W}\sum_{m=1}^{H} x_{tilm} \\ \sigma_{ti}^2 &= \frac{1}{HW}\sum_{l=1}^{W}\sum_{m=1}^{H}(x_{tilm} - mu_{ti})^2 \\ y_{tijk} &= \frac{x_{tijk} - \mu_{ti}}{\sqrt{\sigma_{ti}^2 + \epsilon}} \end{aligned} \tag{4.2}$$

Since, in this dissertation, we are working with a batch size equal to 1, the batch normalization works similar to the instance normalization [28], so using both the batch normalization and the instance normalization from the articles, is the same thing in terms of how the values are normalized. So we used the instance normalization.

The main difference between our discriminator and the one from the article is that they use a 70×70 patchGAN, and we modified that into a 142×142 patchGAN. Some articles suggest that usually, the 70×70 patchGAN obtains better results, but since that depends on the task, this modification was made, giving an overview of its limitations.

The activation function used in both discriminators is the LeakyReLu (Equation(2.3)). This function differs from the ReLu activation because instead of having zero for negative values, it has a small slope for them. It is normally used in GAN because this type of algorithm may suffer from sparse gradients.

The discriminator architecture (Figure 4.4) is quite similar in both models. The only difference is that the Pix2Pix-GAN receives not only the generated image but also the original output image in order to compare both. The convolution layers in blue colour use stride two by two. This means that we are downsampling the input while the convolution layers with no strides are used to maintain the spatial size.

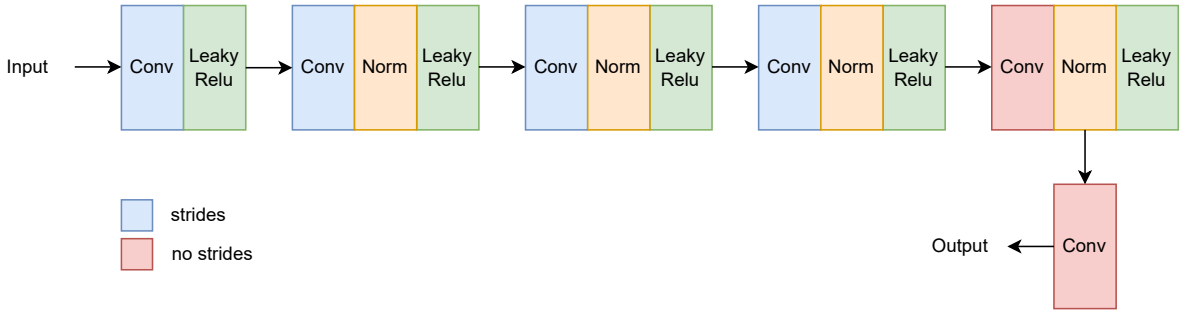


Figure 4.4: Discriminator architecture.

The output shape of our discriminator is (1, 16, 16, 1), the first value is the batch size, the second and third values are the image shape, and the last value represents the number of channels of the image. The output vector is 16 × 16 and each output vector classifies a 142 x 142 area of the input image. This area of the image input is also known as the receptive field size and can be calculated using Equation (4.3). In this Equation the L represents the number of layers and the l the current layer. The k represents the kernel size at the l layer and the s represents the strides at the corresponding layer l.

$$r_0 = \sum_{l=1}^{L}((k_l - 1)\prod_{i=1}^{l-1} s_i) + 1 \tag{4.3}$$

4.3.2 Generator

The first generator (Figure 4.5) used consists of an encoder-decoder also using multiple residual blocks in the middle. It first downsamples the input data, helping also speed up the training process. After the downsampling, the information goes through multiple residual blocks. These residual blocks are when we fast-forward an activation layer into a deep layer in the network. The input data is finally processed by the upsampling, obtaining the interpreted output.

The second generator (Figure 4.6) has the same structure as the first one but with the addition of a skip connecting from the first activation layer to the last convolution layer, helping preserve some of the initial information which may have been lost during the following layers.

The normalization function used in both generators is the instance normalization as used in the discriminator. The activation function is the ReLu function (Equation (2.2)) through the entire network except the output layer that uses the Tanh function (Equation (2.4)).

This choice was made because it makes the network more stable and learns to saturate and cover the training distribution's colour space more quickly, as explained in article [29].

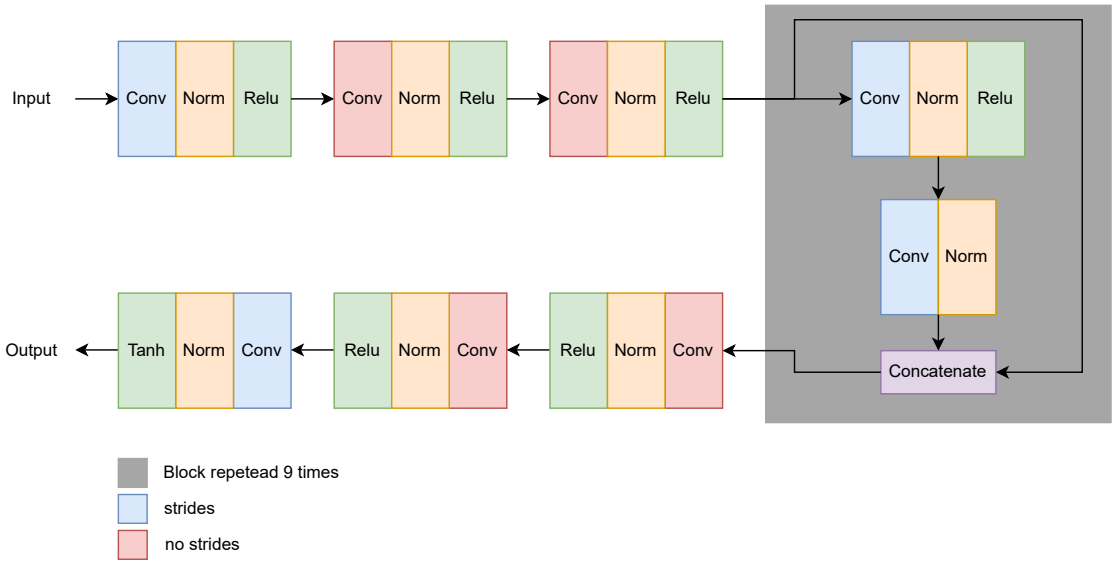


Figure 4.5: First Generator architecture (Residual).

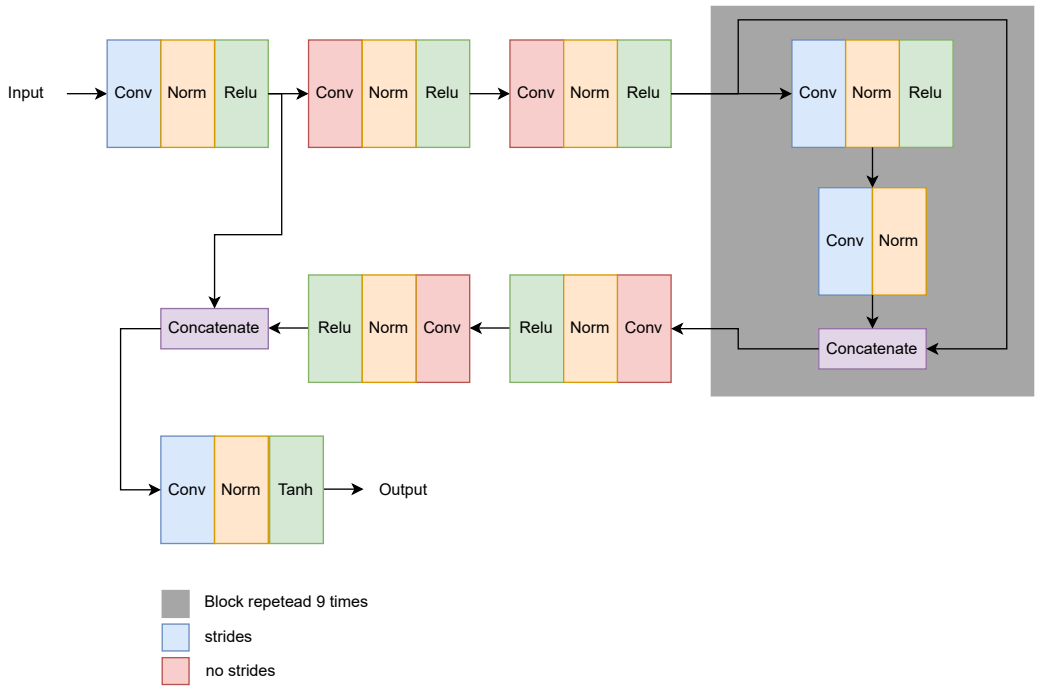


Figure 4.6: Second Generator architecture (SkipResidual).

4.3.3 Approaches

Using the architectures described above, 2D, 2.5D and 3D models were created. The 2D model uses independent slices, and we change both architectures to use the 2D convolution layers in the TensorFlow package. For the 2.5D and 3D models, a method was created that allows us to generate the dataset that will be used in these models. We must first define how many 2D images will be for each 3D image. Since we had limited computing resources, we opted only to

use three 2D images for each 3D image. After defining this parameter, we need to care about not mixing images from different patients, for example, using the last slice of a patient with the first of the next one. To solve this problem, we counted of many images there were for every patient, storing them in an array. This array will be used to prevent this problem and to help us cycle between all the images in the dataset. The creation of a 3D image basically consisted of the current image, the previous and the next images (Figure 4.7).

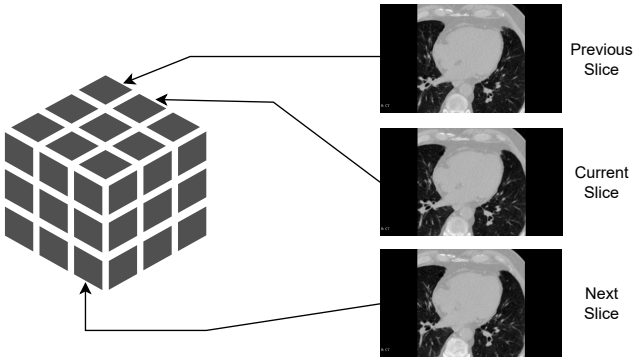


Figure 4.7: 3D image from 2D slices.

The generator in the 3D approach uses 3 consecutive slices to generate the corresponding 3 slices of contrast CT. On the other hand, in the 2.5D approach, 3 consecutive slices of non-contrast CT are used to generate only a single slice of contrast CT, i.e., the middle slice.

For the 3D model, we used the 3D convolution layers instead of the 2D layers. Therefore, the kernel size in the Z-axis has the value of 3 to use both the previous, current and posterior image. The 2.5D model uses the same layers that the 3D model but with the Z-axis of the kernel size with value 1, thus using the information from neighboring slices only via combining feature maps.

4.4 Performance Metrics

In order to evaluate the performance of our models, the use of some evaluation metrics was a need. Some of the metrics used were chosen from the state-of-art, and we have added some new ones as well.

The first metric used is the SSIM (Equation(4.4)). This metric is used to measure how similar two images are, comparing the reference image and the disturbed one. In this case, the reference image is the original contrast CT and the disturbed one is the contrast-enhanced CT. The c_1 and c_2 variables are only to stabilize the division. Theμ_x and μ_y are the pixel mean from each image. In this Equation we also have the σ_x^2 and σ_y^2 that represent the variance of each image, in the other hand, the σ_{xy} represents the covariance of both images. Finally the k_1 and k_2 are constants and the L is the dynamic range of the pixel-values. The higher the SSIM means that both images are highly similar. The implementation of this metric was the one in the TensorFlow package.

$$SSIM(x,y) = \frac{(2\mu_x\mu_y + c_1)(2\sigma_{xy} + c_2)}{(\mu_x^2 + \mu_y^2 + c_1)(\sigma_x^2 + \sigma_y^2 + c_2)}$$
$$c_1 = (k_1 L)^2$$
$$c_2 = (k_2 L)^2 \tag{4.4}$$

The second metric used is the PSNR (Equation (4.5)). This metric is used to measure the quality measurement between two images. This metric is used with the MSE, while the PSNR represents a measure of the peak error, the MSE represents the cumulative squared error. The implementation of the PSNR was the one in the TensorFlow package.

$$PSNR = \log(\frac{max\_value^2}{MSE}) \tag{4.5}$$

The MSE (Equation (4.6)) is also used in this dissertation. The higher the PSNR means the MSE is lower. In this metric for the denominator, the M and N are the resolution of the image. The M and N in the numerator are each pixel of the respective image.

$$MSE = \frac{\sum_{M,N} [Image_1(M,N) - Image_2(M,N)]^2}{M * N} \tag{4.6}$$

The fourth metric is the FID (Equation(4.7)) compares the distribution of generated images with the distribution of the real images. The mu_1 is the mean of the real images, and the mu_2 is the mean of the generated images. The C_1 and C_2 are the covariance matrices of the real and generated images, and the Tr is the sum of elements on the main diagonal. The higher the FID value, it means that the distribution is not similar, meaning a worse generation of images.

$$FID = ||mu_1 - mu_2||^2 + Tr(C_1 + C_2 - 2 * \sqrt{(C_1 * C_2)}) \tag{4.7}$$

The last metric used is the Dice similarity coefficient (Equation (4.8)), and it is used to compare the pixel-wise agreement between the original image and the generated image. We use the Dice in order to calculate how similar the contrast in both CT. The **X** is the original image and **Y** the generated image.

$$Dice = 2 * |X \cap Y| / (|X| + |Y|) \tag{4.8}$$

We converted both the real and generated image to binary (Figure 4.8), considering the pixels with a value greater than 250 are equal to 1, and all the other values are 0. The value 250 was chosen because the pixel value range was between -150 and 590 for the contrast images. We did tests with different thresholds and this value represented a good amount of information about the contrast in the image. This was made in order to evaluate how well the generated contrast image was reproducing the information given by the contrast in terms of lumen of the coronary arteries.

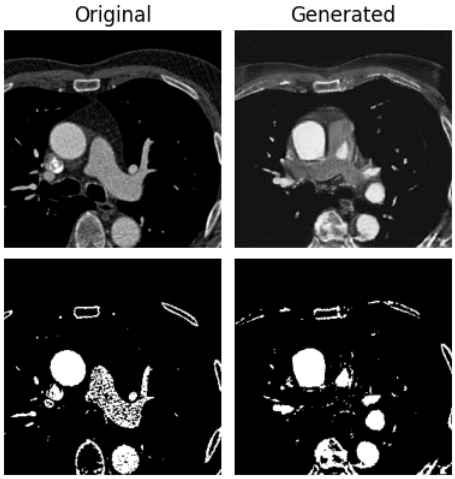


Figure 4.8: Conversion of image to binary.

Chapter 5

Results and Discussion

In this chapter, we will describe the different results obtained and the corresponding analysis, also visualizing some of the examples produced by different models.

Our objective was to compare the performance of the Cycle-GAN vs Pix2Pix-GAN and the trade-off between 2D, 2.5D and 3D. So to have these results, for both the Cycle-GAN and Pix2Pix-GAN, we trained using the 2D, 2.5D and 3D datasets for both the generator architectures.

5.1 Results with the ORCA dataset

The first comparison we will make is between the Residual(first generator) and the SkipResidual(second generator) generators using the Orca dataset, in the Table 5.1 and Table 5.2 we have the corresponding results.

As we can observe in Table 5.1, the Pix2Pix-GAN obtained better results in the SSIM, PSNR, DICE, and MSE. On the other hand, got worse results in the FID. The SSIM value was expected to be better in the Pix2Pix-GAN since it used not only the non-contrast image but also the corresponding contrast image during the training process in order to generate it better. The PSNR, since it uses the MSE and this metric obtained better results, was already expected the same for it. The DICE obtained similar results using the 2D dataset for both Pix2Pix-GAN and Cycle-GAN, but when using the 3D dataset, it obtained better results for the Pix2Pix-GAN. The 2.5D model increased the performance of the Pix2Pix-GAN compared to the 3D model.

On the other hand, both of these models decreased the performance of the Cycle-GAN. The FID was always better using the Cycle-GAN, making these quite interesting results since this represents the similarity between the distribution of the features in the generated and original images. This means that the Cycle-GAN generated images are generally way more similar to the originals compared to the Pix2Pix-GAN. The low performance of the Pix2Pix-GAN with this metric is because all the images generated from these models have blur, which makes them not

well clear compared to those generated from the Cycle-GAN.

Method	SSIM	PSNR	FID	DICE	MSE
Pix2Pix-GAN 2D	0,492	16,375	239,522	0,398	0,025
Pix2Pix-GAN 2.5D	0,477	15,966	224,496	0,430	0,027
Pix2Pix-GAN 3D	0,458	15,736	238,78	0,412	0,028
Cycle-GAN 2D	0,402	15,519	103,86	0,397	0,030
Cycle-GAN 2.5D	0,433	15,569	138,361	0,284	0,030
Cycle-GAN 3D	0,350	15,213	136,774	0,381	0,032

Table 5.1: Residual Generator results with the Orca dataset.

Allowing a better perception of the results, we also analyzed the images generated. Some examples can be visualised in Figure 5.1. The first row represents the 2D model, the second row the 2.5D and the third row the 3D model. In this example, we can see the blur produced by the Pix2Pix-GAN, also noticing the Pix2Pix-GAN 3D outputs almost have a similar shape to the original image.

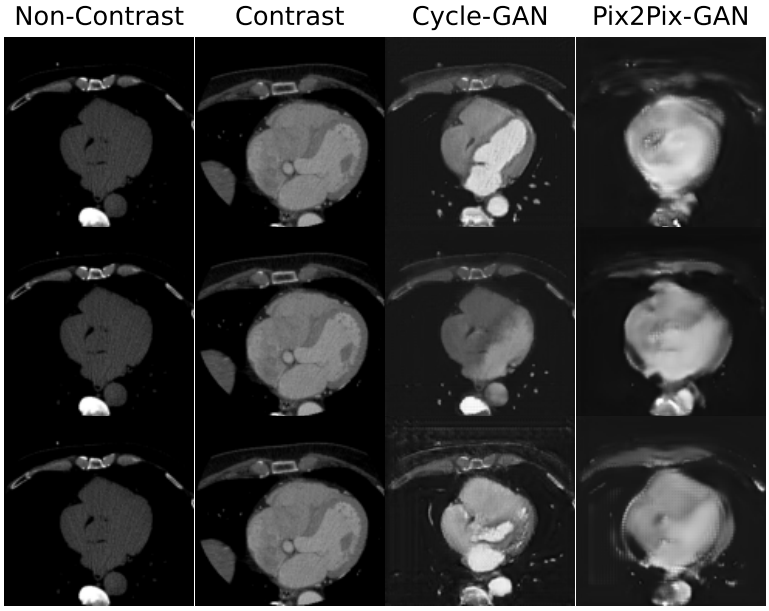


Figure 5.1: Residual Generator Output Example trained and tested with Orca dataset.

In Figure 5.2 compared to the previous example it has more difficult to produce that shape. That happens because the shape from this example has more details that the previous one, needing more images from this type in order to produce such detailed imaged. This happens for both the Cycle-GAN and the Pix2Pix-GAN. The blur present in the Pix2Pix-GAN is more easier to read since the pixel-value in the image are more different.

In Table 5.2, we can observe that the SSIM, PSNR and MSE still perform better in the Pix2Pix-GAN. The DICE increased the performance in the Cycle-GAN. This may happen

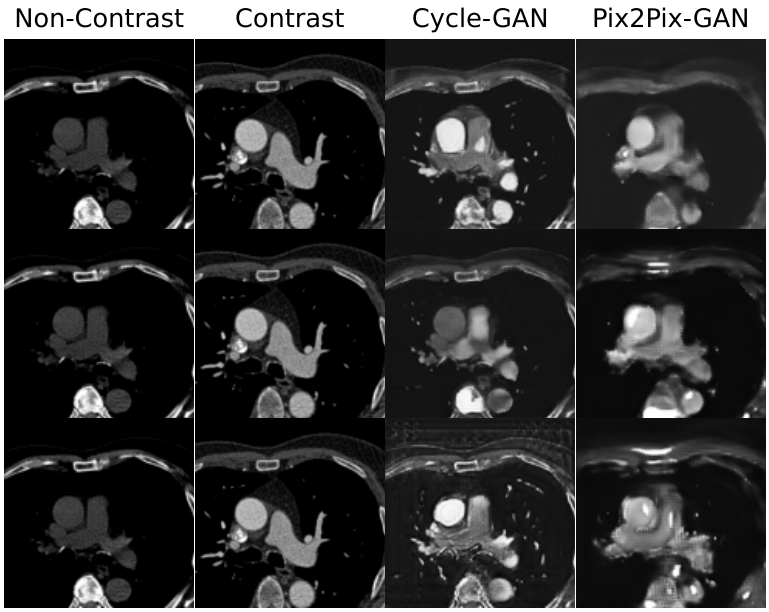


Figure 5.2: Residual Generator Second Output Example trained and tested with Orca dataset.

because this generator concatenates the output of the first layer with the input of the last layer, making them use the initial features to produce a more precise image. The FID continues to perform better in the Cycle-GAN since the Pix2Pix-GAN still has blur, making the features' distribution more different from the original images.

Method	SSIM	PSNR	FID	DICE	MSE
Pix2Pix-GAN 2D	0,503	16,661	239,445	0,323	0,023
Pix2Pix-GAN 2.5D	0,491	16,243	219,253	0,407	0,025
Pix2Pix-GAN 3D	0,474	16,051	221,577	0,421	0,026
Cycle-GAN 2D	0,394	15,669	105,22	0,424	0,029
Cycle-GAN 2.5D	0,398	14,967	114,32	0,413	0,034
Cycle-GAN 3D	0,299	15,265	137,738	0,435	0,032

Table 5.2: SkipResidual Generator results with the Orca dataset.

In order to give a better visualization of the models performance we can see in Figure 5.3 the graph of each metrics with all the models used trained and tested with the orca dataset.

In Figure 5.4, we have some examples of the model output. The rows follow the same sequence as the previous example. These examples show that with this concatenation, the output of the Pix2Pix-GAN starts to become more different from the original.

We can also see in Figure 5.5 another example of the SkipResidual Generator. In this example happens the same from the one in Figure 5.2, both models still have difficulty producing the shape of the contrast image and the intensity of the contrast.

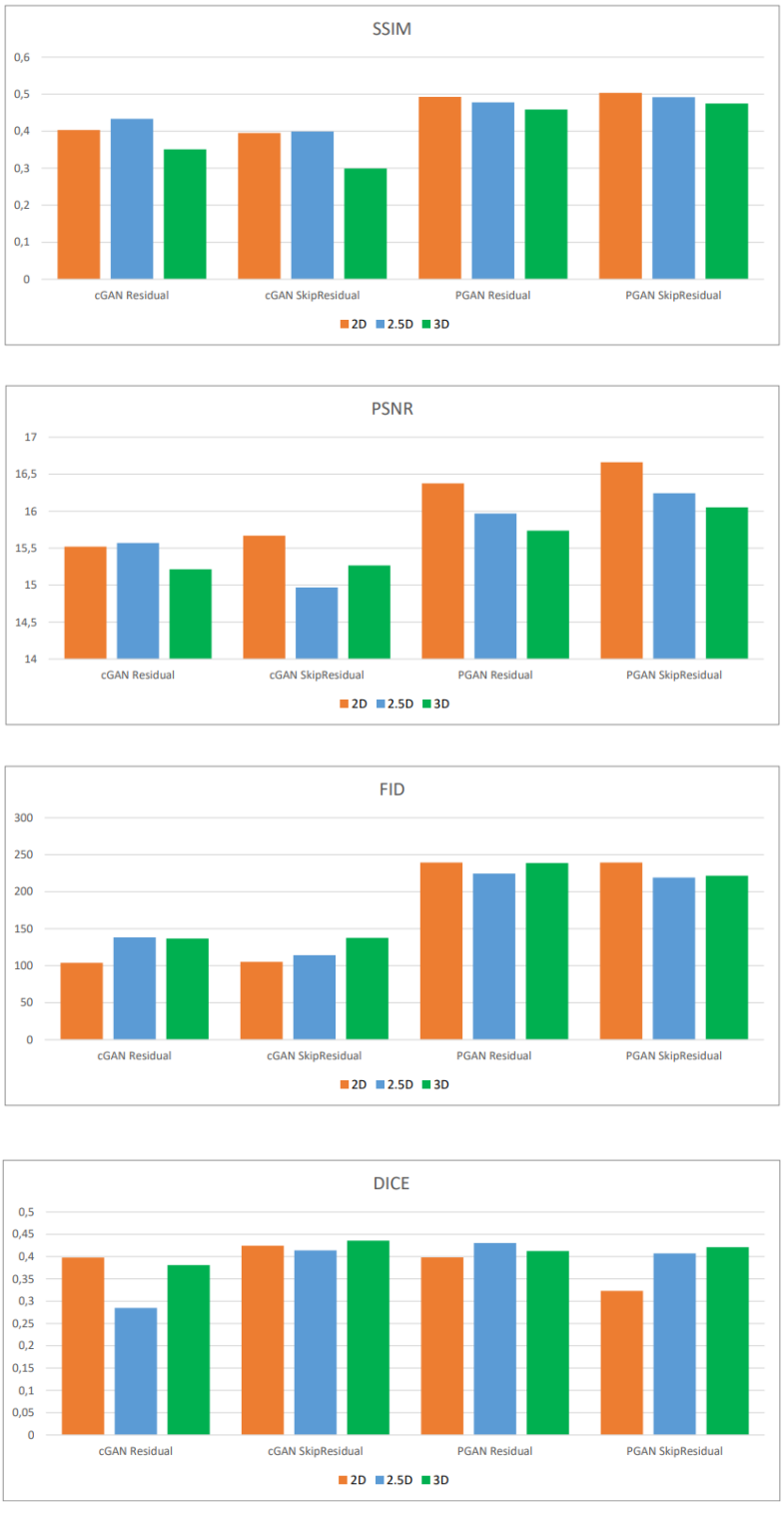


Figure 5.3: Graph performance for both generators only using orca dataset.

5.2 Results with the Hospital dataset

After using the Hospital dataset to test the models trained with the Orca dataset, we used the Hospital dataset to train new models as well. In Table 5.3 as we expected, after training

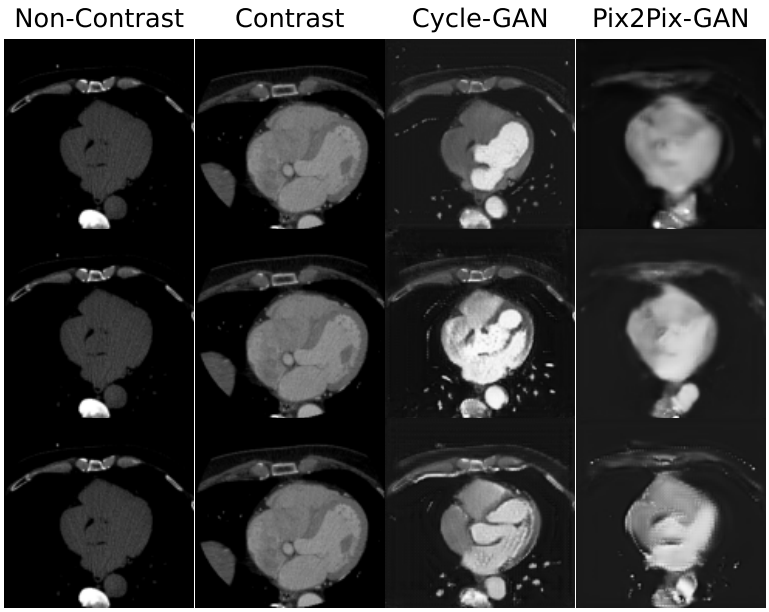


Figure 5.4: SkipResidual Generator Output Example trained and tested with Orca dataset.

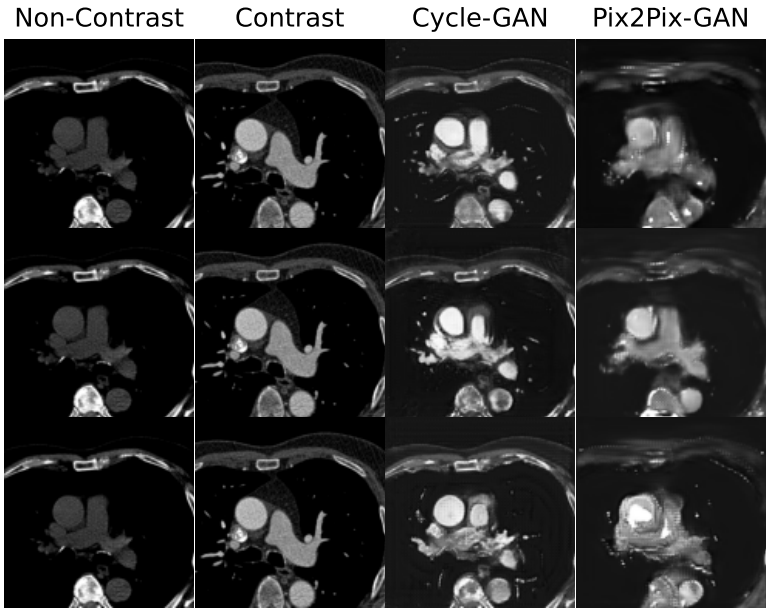


Figure 5.5: SkipResidual Generator Second Output Example trained and tested with Orca dataset.

the models with the Hospital images and testing with the test set from the same, the models achieved better performance. The DICE values are similar in all the models, making them consistent. This also happens for the MSE and PSNR. The SSIM values are higher in the Pix2Pix-GAN since it uses both the non-contrast image and contrast image in the discriminator for evaluation, forcing the generator to produce more realistic images. The FID continues higher in the Pix2Pix-GAN. This shows that the Cycle-GAN still produces a generated set more realistic

than the Pix2Pix-GAN.

Method	SSIM	PSNR	FID	DICE	MSE
Pix2Pix-GAN 2D	0,373	13,830	300,54	0,550	0,043
Pix2Pix-GAN 2.5D	0,381	13,735	309,775	0,562	0,044
Pix2Pix-GAN 3D	0,353	13,760	293,025	0,547	0,045
Cycle-GAN 2D	0,281	13,443	178,7	0,572	0,048
Cycle-GAN 2.5D	0,326	13,669	193,113	0,560	0,044
Cycle-GAN 3D	0,286	13,863	184,386	0,599	0,043

Table 5.3: Residual Generator results with the Hospital dataset.

In order to give a better visualization of the models performance we can see in Figure 5.6 the graph of each metrics with all the models used trained and tested with the hospital dataset.

The Figure 5.7 shows some output examples from this model. The rows follow the scheme from the previous examples. We can see that in these examples, the blur produced by the Pix2Pix-GAN is below the other output models. This happens because in this dataset, the images were retrieved only using one machine, and in the Orca dataset, multiple machines we different zooms were used. This made images different from each other, which introduced errors in the model training. The output for both models is now producing the right intensity in the correct areas, which is an improvement from previous models.

In Figure 5.8 we can see that both models are having more difficult producing images that have a great amount of contrast in them. It still produces in some zones the right intensity of contrast but in the generality of the image still has some flaws.

The Cycle-GAN 3D models present in all the different tests the best DICE values compared to 2.5D and 2D models. On the other hand, we can see that the feature distribution of the generated set (FID) is worse compared to the 2D. This also happens for the 2.5D compared to the 2D. This may be due to the fact that knowing the previous and posterior slices may hinder the training.

The Pix2Pix-GAN 3D and 2.5D models presented better results in the DICE metric compared to the 2D. Both these models resulted in similar values. In this model, the FID values were best in the 2.5D models. This may induce that knowing that there is a previous and posterior slices help the model during the training but using them hinder the training.

Both types of models of the Cycle-GAN and Pix2Pix-GAN produced similar MSE values that also resulted in similar PSNR values. The SSIM is similar in both 2D, 2.5D and 3D. However, the 2.5D for the Cycle-GAN produced slightly better results. For the Pix2Pix-GAN, the SSIM is always higher but hard to evaluate since it has blur. This blur is only low in the last models trained.

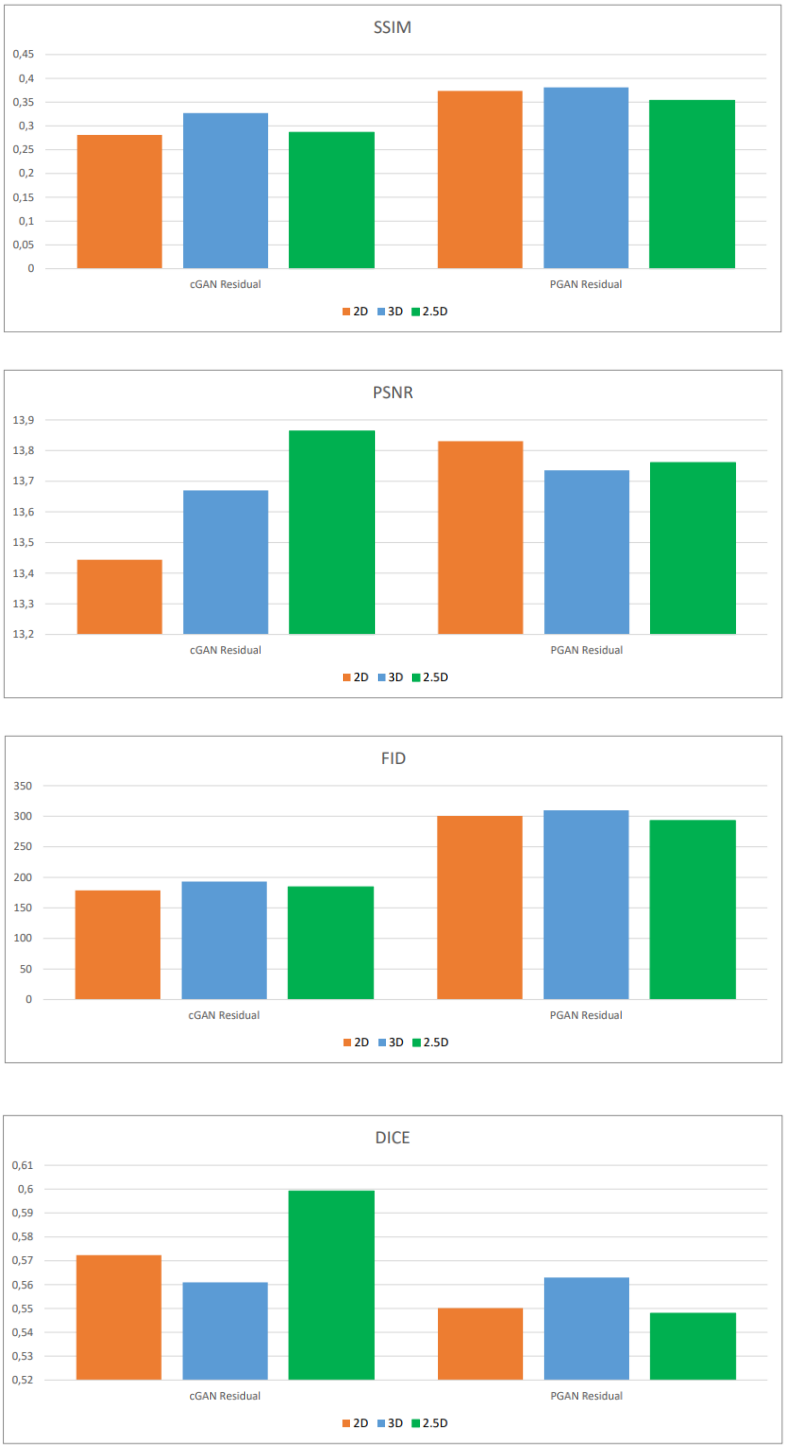


Figure 5.6: Graph performance for Residual generator only using hospital dataset.

5.3 Generalization experiment

We now consider the models performance obtained using the Hospital dataset for testing with the models previously trained over the Orca dataset. In Table 5.4, we can observe that the SSIM values are very low. The source of this low value is that the models are not prepared for this type

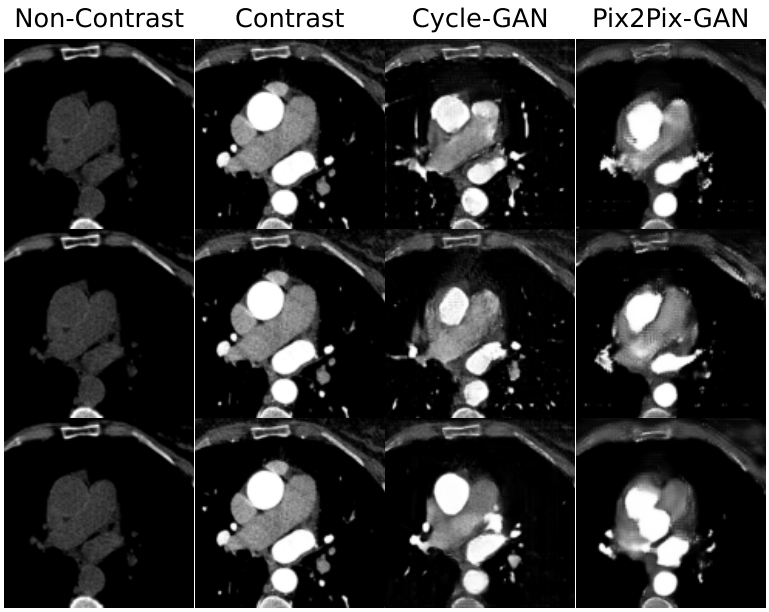


Figure 5.7: Residual Generator Output Example trained and tested with Hospital dataset.

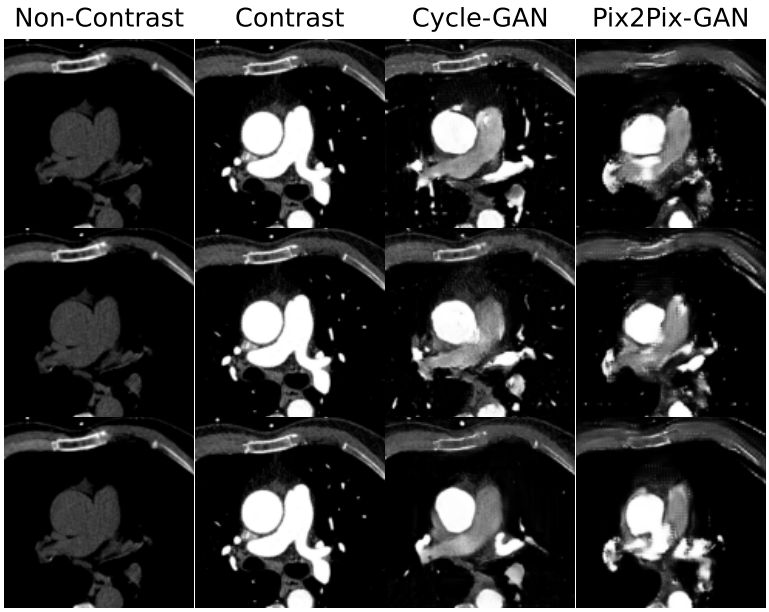


Figure 5.8: Residual Generator Second Output Example trained and tested with Hospital dataset.

of image. Since the model is working with a new dataset, we can see the FID as huge values that tell us the distribution of the features got worse in both models. The DICE values are not that distant from the values obtained using the Orca dataset for testing. This tells us the contrast is being generated in a similar way. The MSE increased by double in most cases which also led to a decrease in the PSNR.

Looking for the results is not the only thing we should be doing, so for that, we analyzed some examples manually to see how the model was working. In Figure 5.9, we can see some examples.

Method	SSIM	PSNR	FID	DICE	MSE
Pix2Pix-GAN 2D	0,109	13,157	388,806	0,290	0,053
Pix2Pix-GAN 2.5D	0,115	13,255	355,067	0,409	0,052
Pix2Pix-GAN 3D	0,101	12,935	385,258	0,369	0,055
Cycle-GAN 2D	0,107	13,726	225,693	0,418	0,045
Cycle-GAN 2.5D	0,120	13,069	251,84	0,301	0,054
Cycle-GAN 3D	0,099	13,604	246,652	0,453	0,046

Table 5.4: Residual Generator results trained with Orca and tested with Hospital dataset.

Each row represents the type of the model used, being this 2D, 2.5D and 3D, respectively. The Pix2Pix-GAN outputs have a lot of noise, making them hard to understand, resulting in low-quality performance. The Cycle-GAN outputs generated the right contrast in some areas making these promising results.

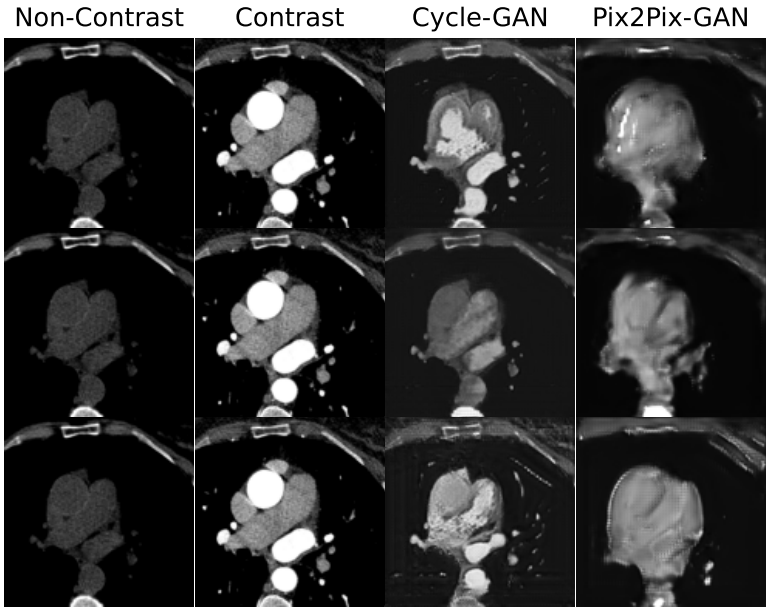


Figure 5.9: Residual Generator Output Example trained with Orca and tested with Hospital dataset.

Figure 5.10 shows an example of the Cycle-GAN producing a image closer to the original one. This happens more in the 3D model (last row), this model produce contrast in almost every zone it was supposed but still missing the intensity. This intensity error is due to the fact that this model was trained with another dataset that may not contain images with this amount/intensity of contrast. The Pix2Pix-GAN still produces blur making it almost non-readable.

In Table 5.5, we can see the SkipResidual generator also tested with the Hospital dataset. The SSIM, PSNR, FID and MSE are almost the same performance from the Residual generator. The Cycle-GAN had a good performance with the DICE metric, meaning that the contrast is

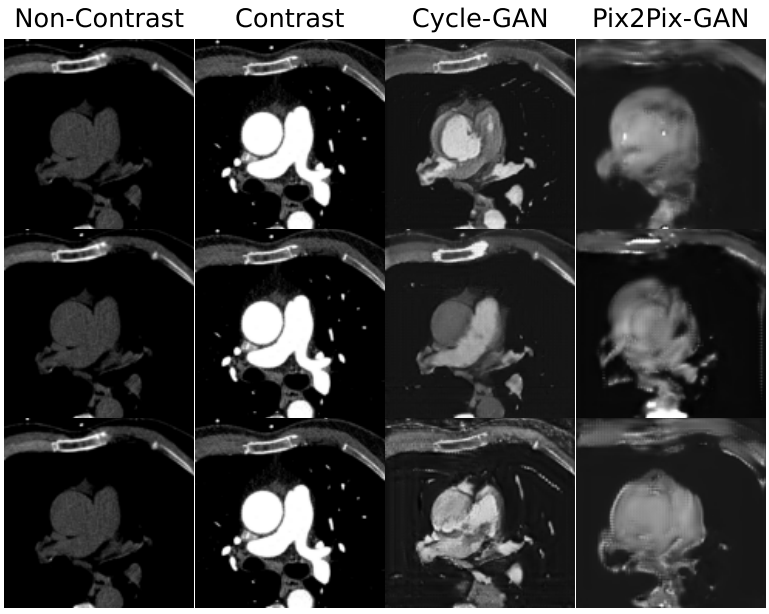


Figure 5.10: Residual Generator Second Output Example trained with Orca and tested with Hospital dataset.

being better generated.

Method	SSIM	PSNR	FID	DICE	MSE
Pix2Pix-GAN 2D	0,111	12,976	379,508	0,154	0,056
Pix2Pix-GAN 2.5D	0,114	13,118	360,821	0,339	0,053
Pix2Pix-GAN 3D	0,103	13,181	373,797	0,367	0,052
Cycle-GAN 2D	0,115	13,905	210,704	0,462	0,044
Cycle-GAN 2.5D	0,117	13,635	229,066	0,455	0,046
Cycle-GAN 3D	0,111	13,799	272,664	0,495	0,044

Table 5.5: SkipResidual Generator results trained with Orca and tested with Hospital dataset.

In order to give a better visualization of the models performance we can see in Figure 5.11 the graph of each metrics with all the models used trained with the orca dataset and tested with the hospital dataset.

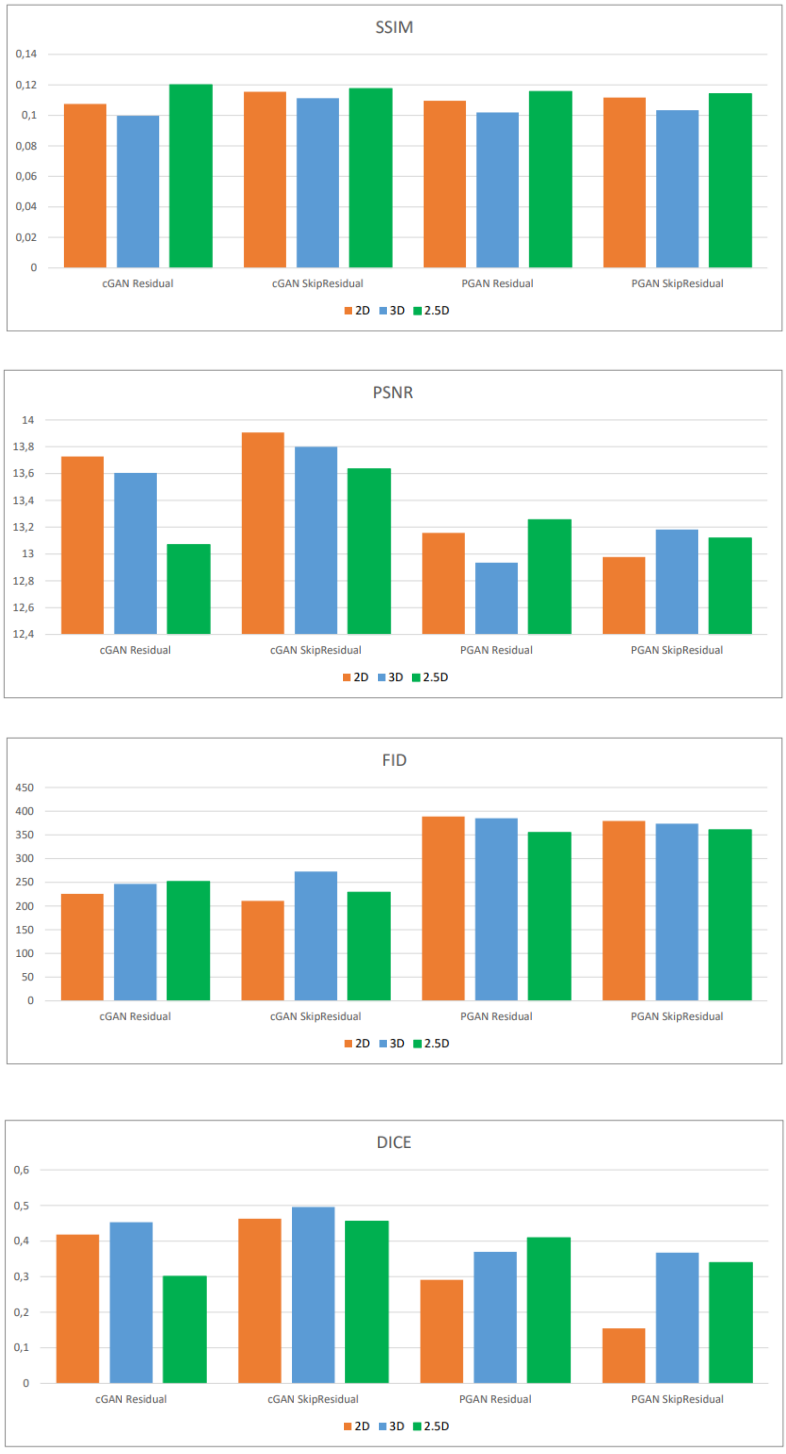


Figure 5.11: Graph performance for both generators using orca dataset for train and hospital dataset for test.

Figure 5.12 shows some examples from this models outputs. The Pix2Pix-GAN is still with a huge amount of blur, maintaining the bad performance generating the image. The Cycle-GAN is trying already to generate contrast with the same intensity as the original image. In the row, we can see that the Cycle-GAN is generating the contrast in the right places but still misses some

intensity values.

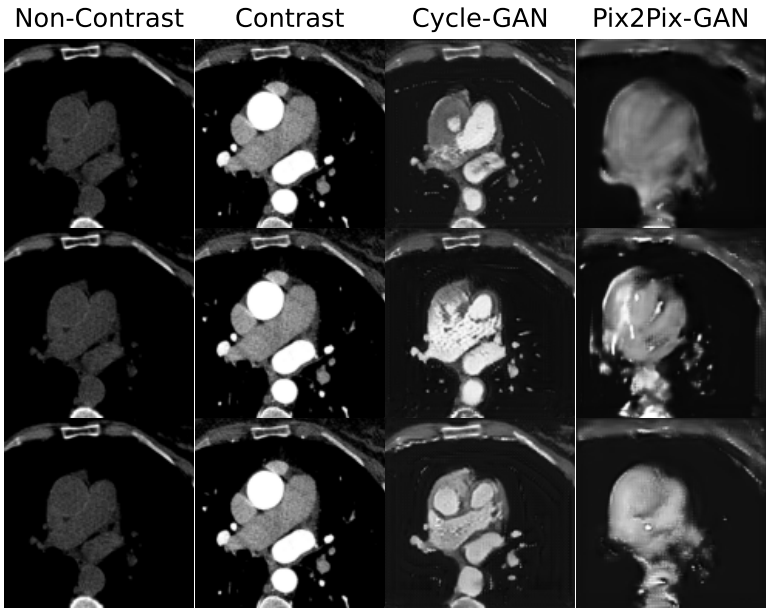


Figure 5.12: SkipResidual Generator Output Example trained with Orca and tested with Hospital dataset.

In Figure 5.13 we can see the the Pix2Pix-GAN still produces blur because of being trained with different dataset and not being prepared for this one. The Cycle-GAN is still missing the intensity of the contrast, this is because of the previous data not having this intensity and even this quantity of contrast making hard to predict correctly this input.

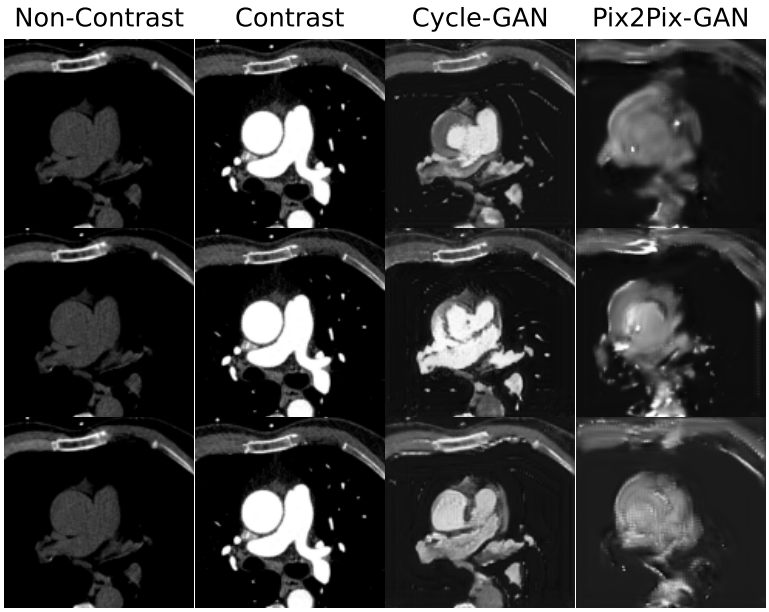


Figure 5.13: SkipResidual Generator Second Output Example trained with Orca and tested with Hospital dataset.

In Figure 5.14 we can see the graphs that describe the performance for the models only tested with the hospital dataset and the models trained and tested with this dataset. As expected, the models trained with the hospital dataset produced better results since they knew the dataset and were prepared for the shapes and intensity in the images.

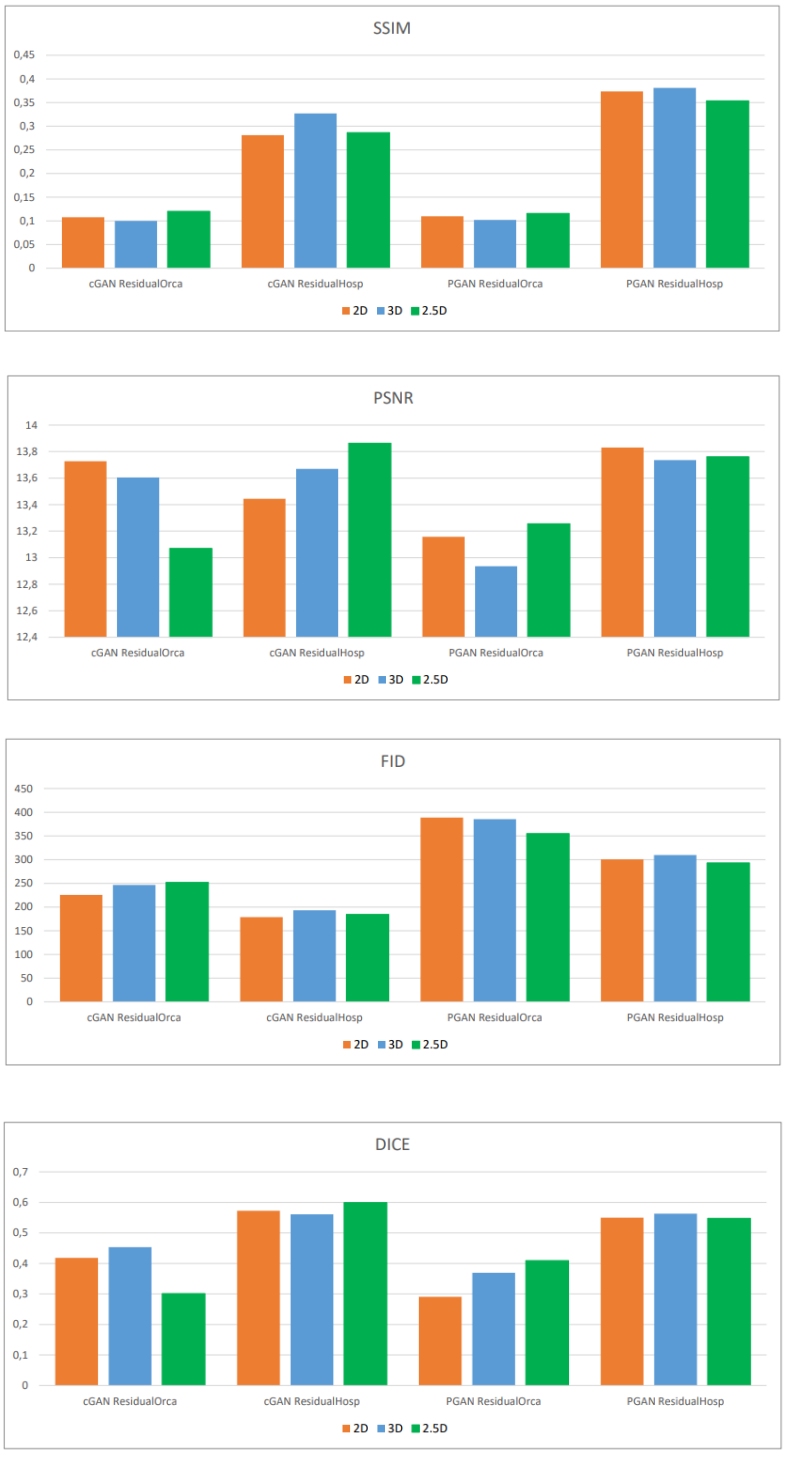


Figure 5.14: Graph performance for models trained with both datasets and tested with hospital dataset.

Chapter 6

Conclusion

This dissertation described the implementation of two different types of DL algorithms to generate contrast in contrastless CT. The algorithms used are defined as GAN, which are commonly used in image generation. The two variants of this algorithm implemented were the Cycle-GAN and Pix2Pix-GAN. The fundamentals behind the CAD, DL, GAN and neural networks were described, and an overview of the most recent and pertinent literature on the implementations already made in this context was made, namely the use of GAN to solve this problem.

The results presented show that the Cycle-GAN seems more promising than the Pix2Pix-GAN. This is due its robustness against image blur. This happens because the Cycle-GAN does not require perfectly aligned pairs of images to train, while the Pix2Pix is trying to generate the supposed shape of the output and generated the contrast in that same shape.

We also explored the use of different types of data formats: 2D, 2.5D and 3D. The last two types of data show that having information from neighbour slices helps generate a more consistent set of contrast images. For the Orca dataset, the 2D produced images with more significant similarity, while the Hospital dataset got better results with the 2.5D. This happened because, in the Orca dataset image alignment was obtained via manual selection and some slices got removed because there was no corresponding image, while in the Hospital dataset we applied a simple registration algorithm in order to align the images, thus reducing the loss of slices. The evaluation of contrast generation using the DICE metric shows that, for the Cycle-GAN, the 3D produced better results, and for the Pix2Pix, the 2.5D worked better. This shows us that information from a neighbour slice always helps but for Pix2Pix-GAN, using it induces an error. The impact of the blur produced in image generation can be captured by the evaluation of the FID, that represents a fundamental performance metric that is usually not considered by related works in the literature. All the results obtained were satisfactory in general since they helped us to understand the limitations of each model and of each type of data.

For future work, using a larger and well-distributed dataset will improve the performance of the models. This may be done by obtaining more data from the same source or even trying to combine datasets from different sources and see how the model works with that. Also, with the

results obtained, the focus now can be the modification of both model architectures with the type of data that best suits them in order to improve their performance. Since the results from some metrics do not represent the true performance of the models for future work, it would be good to involve clinical experts in the area to help evaluate the results obtained.

Bibliography

[1] Coronary artery disease: Causes, symptoms, diagnosis & treatments.

[2] Saugat Bhattarai. What is activation functions in neural network (nn)?, Jun 2018.

[3] Anirudh Chandrashekar, N Shivakumar, P Lapolla, A Handa, Vicente Grau, and Regent Lee. A deep learning approach to generate contrast-enhanced computerised tomography angiograms without the use of intravenous contrast agents. *European Heart Journal*, 41 (Supplement_2):ehaa946–0156, 2020.

[4] Xiaojiao Xiao, Yan Qiang, Juanjuan Zhao, Xingyu Yang, and Xiaotang Yang. Segmentation of liver lesions without contrast agents with radiomics-guided densely unet-nested gan. *IEEE Access*, 9:2864–2878, 2020.

[5] Peter Libby and Pierre Theroux. Pathophysiology of coronary artery disease. *Circulation*, 111(25):3481–3488, 2005.

[6] Karen Okrainec, Devi Banerjee, and Mark Eisenberg. Coronary artery disease in the developing world. *American heart journal*, 148:7–15, 07 2004. doi:10.1016/j.ahj.2003.11.027.

[7] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. Deep learning. *nature*, 521(7553): 436–444, 2015.

[8] Sijie Yang, Fei Zhu, Xinghong Ling, Quan Liu, and Peiyao Zhao. Intelligent health care: Applications of deep learning in computational medicine. *Frontiers in Genetics*, 12, 2021. ISSN: 1664-8021. doi:10.3389/fgene.2021.607471.

[9] Xian-Da Zhang. Machine learning. In *A Matrix Algebra Approach to Artificial Intelligence*, pages 223–440. Springer, 2020.

[10] Leslie Pack Kaelbling, Michael L Littman, and Andrew W Moore. Reinforcement learning: A survey. *Journal of artificial intelligence research*, 4:237–285, 1996.

[11] Pramila P Shinde and Seema Shah. A review of machine learning and deep learning applications. In *2018 Fourth international conference on computing communication control and automation (ICCUBEA)*, pages 1–6. IEEE, 2018.

[12] Jiuxiang Gu, Zhenhua Wang, Jason Kuen, Lianyang Ma, Amir Shahroudy, Bing Shuai, Ting Liu, Xingxing Wang, Gang Wang, Jianfei Cai, and Tsuhan Chen. Recent advances in convolutional neural networks. *Pattern Recognition*, 77:354–377, 2018. ISSN: 0031-3203. doi:https://doi.org/10.1016/j.patcog.2017.10.013.

[13] Antonia Creswell, Tom White, Vincent Dumoulin, Kai Arulkumaran, Biswa Sengupta, and Anil A. Bharath. Generative adversarial networks: An overview. *IEEE Signal Processing Magazine*, 35(1):53–65, 2018. doi:10.1109/MSP.2017.2765202.

[14] Phillip Isola, Jun-Yan Zhu, Tinghui Zhou, and Alexei A. Efros. Image-to-image translation with conditional adversarial networks. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5967–5976, 2017. doi:10.1109/CVPR.2017.632.

[15] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. *CoRR*, abs/1505.04597, 2015.

[16] Joyce Henry, Terry Natalie, and Den Madsen. Pix2pix gan for image-to-image translation.

[17] Jun-Yan Zhu, Taesung Park, Phillip Isola, and Alexei A. Efros. Unpaired image-to-image translation using cycle-consistent adversarial networks. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, Oct 2017.

[18] Xudong Mao, Qing Li, Haoran Xie, Raymond Y.K. Lau, Zhen Wang, and Stephen Paul Smolley. Least squares generative adversarial networks. In *2017 IEEE International Conference on Computer Vision (ICCV)*, pages 2813–2821, 2017. doi:10.1109/ICCV.2017.304.

[19] Jae Won Choi, Yeon Jin Cho, Ji Young Ha, Seul Bi Lee, Seunghyun Lee, Young Hun Choi, Jung-Eun Cheon, and Woo Sun Kim. Generating synthetic contrast enhancement from non-contrast chest computed tomography using a generative adversarial network. *Scientific reports*, 11(1):1–10, 2021.

[20] Wook Kim, Sang-Keun Woo, Jisu Park, Heesoon Sheen, Ilhan Lim, Sang Moo Lim, and Byung Hyun Byun. Contrast ct image generation model using ct image of pet/ct. In *2018 IEEE Nuclear Science Symposium and Medical Imaging Conference Proceedings (NSS/MIC)*, pages 1–3. IEEE, 2018.

[21] Sang-Keun Woo. Generation of contrast enhanced computed tomography image using deep learning network. *Journal of The Korea Society of Computer and Information*, 24(3):41–47, 2019.

[22] Jelmer M. Wolterink PhD. Bob D. de Vos PhD. Tim Leiner MD PhD. Max A. Viergever PhD. Ivana Išgum PhD. Orca score. https://orcascore.grand-challenge.org/, January 2022.

[23] 3d slicer image computing platform.

[24] Geometric image transformations.

[25] Jun-Yan Zhu, Taesung Park, Phillip Isola, and Alexei A. Efros. Unpaired image-to-image translation using cycle-consistent adversarial networks, 2017. doi:10.48550/ARXIV.1703.10593.

[26] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift, 2015. doi:10.48550/ARXIV.1502.03167.

[27] Dmitry Ulyanov, Andrea Vedaldi, and Victor Lempitsky. Instance normalization: The missing ingredient for fast stylization, 2016. doi:10.48550/ARXIV.1607.08022.

[28] Yuxin Wu and Kaiming He. Group normalization, 2018. doi:10.48550/ARXIV.1803.08494.

[29] Alec Radford, Luke Metz, and Soumith Chintala. Unsupervised representation learning with deep convolutional generative adversarial networks, 2015. doi:10.48550/ARXIV.1511.06434.