UNIVERSITÀ
DEGLI STUDI
FIRENZE

Dipartimento di Ingegneria dell'Informazione (DINFO)

PhD Program in Smart Computing

Multi–stage generation for segmentation of medical images

Advisors:
Prof. Monica Bianchini
Prof. Franco Scarselli

Evaluation Committee:
Prof. Claudio Angione
Dr. Domenico Perrotta

Candidate:
Giorgio Ciano

Jury:
Prof. Battista Biggio
Prof. Stefano Cagnoni
Prof. Marco Maggini

XXXIV ciclo — 17th June 2022

PhD Program in Smart Computing
University of Florence, University of Pisa, University of Siena

Multi–stage generation for segmentation of medical images

Giorgio Ciano

Advisors:

Prof. Monica Bianchini
Prof. Franco Scarselli

Head of the PhD Program:
Prof. Stefano Berretti

Evaluation Committee:
Prof. Claudio Angione
Dr. Domenico Perrotta

Jury:
Prof. Battista Biggio
Prof. Stefano Cagnoni
Prof. Marco Maggini

XXXIV ciclo — 17th June 2022

Abstract

Recently, deep learning methods have had a tremendous impact on computer vision applications. The results obtained were unimaginable a few years ago. The problems of greatest interest are image classification, semantic segmentation, object detection, face recognition, and so on. All these tasks have in common the necessity of having a sufficient quantity of data to be able to train the model in a suitable manner. In fact, deep neural networks have a very high number of parameters, which imposes a fairly large dataset of supervised examples for their training. This problem is particularly important in the medical field, especially when the goal is the semantic segmentation of images, both due to the presence of privacy issues and the high cost of image tagging by medical experts. The main objective of this thesis is to study new methods for generating synthetic images along with their label–maps for segmentation purposes. The generated images can be used to augment real datasets. In the thesis, in order to achieve such a goal, new fully data–driven methods based on Generative Adversarial Networks are proposed. The main characteristic of these methods is that, differently from other approaches described in literature, they are multi–stage, namely they are composed of some steps. Indeed, by splitting the generation procedure in steps, the task is simplified and the employed networks require a smaller number of examples for learning. In particular, a first proposed method consists of a two–stage image generation procedure, where the semantic label–maps are produced first, and then the image is generated from the label–maps. This approach has been used to generate retinal images along with the corresponding vessel segmentation label–maps. With this method, learning the generator requires only a handful of samples. The method generates realistic high–resolution retinal images. Moreover, the generated images can be used to augment the training set of a segmentation algorithm. In this way, we achieved results that outperforms the state–of–the–art for the task of segmentation of retinal vessels. In the second part of the thesis, a three–stage approach is presented: the initial step consists in the generation of dots whose positions indicate the locations of the semantic objects represented in the image; then, in the second step, the dots are translated into semantic label–maps, which are, finally, transformed into the image. The method was evaluated on the segmentation of chest radiographic images. The experimental results are promising both from a qualitative and quantitative point of view.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

The advent of Machine Learning (ML) can be considered an evolution of computer science. In fact, while in the classical framework the program of an application is written by humans, in ML computers are able to automatically learn meaningful features, relationships and patterns from a series of examples and observations, so that at the end even the program is learnt. Like children, ML algorithms try to learn from their mistakes. The goal is to obtain a model along with its parameters, by minimizing an *ad hoc* loss function. The history of ML is not very recent, and it has alternated between periods of great popularity and periods of total darkness. Its rebirth and its definitive affirmation took place with the advent of Deep Learning (DL).

Deep Neural Networks (DNNs) are feedforward networks with more than one hidden layer. DNNs are also characterized by the presence of several types of layers, e.g. convolutional and pooling layers. These characteristics, which differentiate DNNs from classical neural networks, allowed to overcome important problems of ML algorithms and to implement sophisticated functions such as those encountered in computer vision and natural language understanding. In fact, the techniques used before DL relied on complex feature extraction methods. Instead, with DNNs, raw data can be fed directly into the network, which automatically extracts the representation needed to solve the assigned task. Another characteristic of modern DNNs is that they have become very large and complex, use a large number of parameters, and require a huge amount of data. Since nowadays it is increasingly easy to collect and share data, there has been the definitive outbreak of these networks. In addition, the availability of increasingly performing GPUs, which allow the use and faster training of complex networks, has also had a great influence.

Three types of frameworks characterize DL: supervised, unsupervised and reinforcement learning. Supervised learning exploits a labeled dataset, while unsupervised learning attempts to determine predetermined or unknown structures without the need for human intervention. Also reinforcement learning uses labeled datasets, but the labels are not immediately available for each pattern and its main goal is to find a compromise between unexplored territories (exploration) and current knowledge (exploitation). For each of these approaches, a fair amount of data is essential to obtain acceptable results, especially with regard to generalization, i.e. the ability of the network to make the right decision on new data. However, most of the success of deep architecture is based on supervised learning.

One of the main problems with this approach is the availability of data annotated by experts. There are many fields where collecting a huge amount of labeled data is difficult, one of which is the medical field. In recent years, the number of collaborations between machine learning experts and clinicians have grown exponentially, particularly for computer vision tasks. Classification, semantic segmentation, object detection are the main problems applied to Magnetic Resonance Imaging (MRI), Computed Tomography (CT) and X–Ray scans. In addition to the problem of sharing images due to the privacy of patients, who must give their consent, there is the difficulty in labeling the images. Consider the problem of semantic segmentation. Roughly speaking, the goal here is to classify every pixel in the image. In order to train a semantic segmentation network we need images and the corresponding label–maps, in which we associate each pixel with a class. To obtain these label–maps, a doctor or an expert will have to manually examine the image pixel by pixel. This work is very expensive both in terms of time (in some cases could take hours for a single image) and resources.

Several techniques have been used to overcome the lack of data over time. Classic data augmentation techniques include flips, rotations, scales, crops and Gaussian noise. Obviously, in this way the number of images is easily multiplied, but in some cases this is not enough. A possible alternative is to generate new images from real ones using ML techniques. Actually, machine learning algorithms are able to implicitly acquire a specific domain model from real data, which can be used to generate new data. A popular approach is the Generative Adversarial Network (GANs) (Goodfellow et al., 2014).

In this thesis, we propose new methods, based on GANs, to generate both realistic images and the corresponding label–maps. The main characteristics of those methods is that, differently from other approaches described in literature, they are multi–stage, namely they are composed of some steps. By splitting the generation procedure in steps, we simplify the generative task so that simpler GANs can be used and, more importantly, a smaller number of examples are required to train those GANs. The experimentation of the methods have confirmed that they can produce high quality images and they can be used to augment datasets for segmentation algorithms.

The thesis is focused on two tasks in medical field with different characteristics: the segmentation of retinal images and the segmentation of Chest X–Ray (CXR) images. In particular, in the former task, the goal is to identify blood vessels in retinal images while, in the latter, the goal is to determine which pixels belong to the lungs and the heart in CXR images.

For the former application, which is a binary classification problem, we propose a Two–Stage generation approach, while for CXR images, which involves the identification of multiple classes, a Three–Stage GAN is applied. The proposed Two–Stage method consists of two distinct phases. In the first stage, a GAN learns from data the typical distribution of blood vessels in the image, thus defining the semantic label–map. In the second phase, an image–to–image translation algorithm is trained to transform the blood vessels produced in the first phase into a synthetic image.

For CXR images, the proposed Three-Stage approach includes an extra step at the beginning of the generative procedure. In the first stage, the position of each anatomical part is generated and represented by a "dot" within the image; in the second stage, semantic labels are obtained from the dots; finally, the CXR image is generated. Interestingly, notice that, while in the above procedure the positions of the dots are automatically generated, in another possible use of the Three-Stage approach the dots may be manually positioned in order to have a tool that can generate objects at given positions.

The approaches described above, in addition to being able to generate high resolution images with a limited amount of data, allow us to improve the performance of Deep Learning models for segmentation when few data are available for training. In fact, a correct segmentation, in both types of images, is essential to obtain a correct diagnosis. In the case of retinal fundus imaging, we are mainly concerned with cardiovascular and ophtalmologic diseases. However, a visual analysis of the retinal fundus image is time–consuming. Therefore, it is necessary to develop automatic analysis tools for the retinal fundus images. Certain features of retinal blood vessels, such as width, tortuosity, and branching, are important symptoms of circulatory disease, so achieving good segmentation is the first step towards an accurate diagnosis. The correct automatic segmentation of the lungs and heart can also help physicians in detecting diseases and abnormalities. Similarly, the proper identification of the lungs can be used to extract clinically relevant features. Subsequently, these features can be used to train other networks that deal with classification tasks, making the diagnosis phase fully automatic and supporting the physician in the final decision.

The experimental results show that our methods can generate images that are true to reality. The images have been judged both from a qualitative and a quantitative point of view. For the former evaluation, the images have been visually assessed by experts. For the latter, some segmentation networks have been trained using the generated images and the results of the segmentation has been compared with the literature showing that we can reach the state-of-the-art and possibly outperform it, when the train set is small. In addition to comparing the results with those already present in the literature, we compared the Multi–Stage methods with a Single–Stage approach, in which images and label–maps are generated simultaneously. In particular, an improvement can be observed as the number of steps increases: in fact, the Two–Stage method outperforms the Single–Stage method just as the Three–Stage method outperforms the Two–Stage one.

1.1 Major contributions of the thesis

The main contributions of this thesis can be summarized as follows.

1. Development of two new general methods for generating images and the corresponding label–maps. Both methods allow to obtain a potentially infinite number of synthetic images from a dataset composed of a limited number of images.

   a) The first approach consists of a two–stage image generation procedure. In the first phase, the generation model learns to reproduce the semantic label–maps, while, an image–to–image translation algorithm is used to obtain the final synthetic image (based on (Andreini et al., 2022)).

   b) The second method extends a) by adding an initial step that generates dots corresponding to the objects in the image. Then image–to–image translation algorithms are used to translate first the dots into a label–map and, finally, the label–map into the synthetic image (based on (Ciano et al., 2021a)). With this approach, we can set objects within the image by simply setting the position of the related dot.

2. Application of the Multi–Stage generation methods in representative medical fields.

   a) Use of the Two–Stage method for generating high–resolution retinal images and the corresponding label–maps. The generated images have been used to train a semantic segmentation network, that improves the state–of–the–art (Andreini et al., 2022).

   b) Use of the Three–Stage method for generating high–resolution CXR images and the corresponding label–maps. A semantic segmentation network has been trained by exploiting generated images, obtaining the results discussed in (Ciano et al., 2021a).

1.2 Structure of the Thesis

The thesis is organized as follows.

Chapter 2 introduces Machine Learning and Deep Learning techniques applied to relevant computer vision tasks. In particular, we explain how the gradual transition from the single neuron models to modern networks, with millions of units, has occurred. Subsequently, we describe the techniques applied to computer vision tasks, so that we deal with classical problems concerning semantic segmentation, generation of synthetic images and image–to–image translation algorithms. Moreover, the literature related to the methods employed in this thesis are discussed.

In Chapter 3, we present a two–stage approach for the generation of both images and semantic label–maps. In this chapter, the method is explained in detail, highlighting its advantages from both a qualitative and quantitative point of view. Two–Stage GANs are capable of generating high–resolution synthetic images while being trained with few real images. The developed method was applied on retinal images, with the final goal of demonstrating the usefulness of the generated images for training a semantic segmentation network. Finally, we describe the results, comparing them with a single–stage generation method and with previous works in the literature.

Chapter 4 describes a three–stage method. It is explained how an additional step added to the image generation procedure can improve performance and how, starting from the generation of simple dots, corresponding to objects within the image, we are able to obtain a final high–resolution image. The method proposed in this chapter has been applied to a dataset of CXR images. Examples of generated images are presented together with the obtained quantitative results.

Chapter 5 briefly presents activities in which I have been involved during my PhD period, that are not strictly linked to my thesis.

Finally, Chapter 6 summarizes the contribution of the thesis and discusses matters for future research.

Chapter 2

From the single neuron to Deep Learning

2.1 Machine Learning Techniques

The beginning of the Artificial Intelligence (AI) goes back further than we might think. Since the creation of the "Turing Test" in 1950 (Turing and Ince, 1992), for many decades there have been alternating periods of great popularity of AI and related disciplines (Shannon, 1948; von Neuman et al., 1994) and periods of little interest from the scientific community. In recent years, this alternation has been interrupted. The availability of a huge amount of data and of increasingly performing GPUs — to train more and more complex models —, have sparked the interest in this sector, now present in every area of our life.

In this chapter, we will start from the simplest single–neuron models and then describe the more advanced Deep Learning techniques, in particular those used in Computer Vision.

2.1.1 Neuron model and architecture

An Artificial Neural Network (ANN) is a graph (oriented or not) $A = (V, E)$ whose vertices, $v \in V$, are called *neurons* or *units*, and whose arcs, $e \in E$, are called *connections* or *synapses*. Indeed, each connection, like the synapses in a biological brain, can transmit a signal to other neurons. The graph is labeled on both vertices and arcs. The labels on the arcs, called *weights*, are the *network parameters*, while the labels on the vertices are real values computed using an *activation function*. This function depends on both the input of the neuron and the values of the weights of each incoming connection.

The first ANN model — called *perceptron* and created by Frank Rosenblatt in 1958 (Rosenblatt, 1958; McCulloch and Pitts, 1943) — was a single–neuron architecture.

The perceptron is the basic computational unit for the creation of networks with different architectures and various levels of depth, which have led, time after time, to modern Deep Neural Networks (DNNs). A Multi Layer Perceptrons (MLPs) is an architecture that consists of layers of neurons, called input (I), hidden (H) and output (O) layers (see Figure 2.1). Each layer consists of one or more units, while the arcs connect neurons belonging to consecutive layers, with the information flowing from the input to the output layer. An MLP with several hidden layers is called a DNN. Regarding the activation functions, the Rectified Linear Unit (ReLU) (Krizhevsky et al., 2012) (and its variants) is mainly used in modern architectures while, in shallow MLPs, Threshold Logic Functions, Linear Functions, Sigmoids, Hyperbolic Tangent and Gaussian functions are the most common.

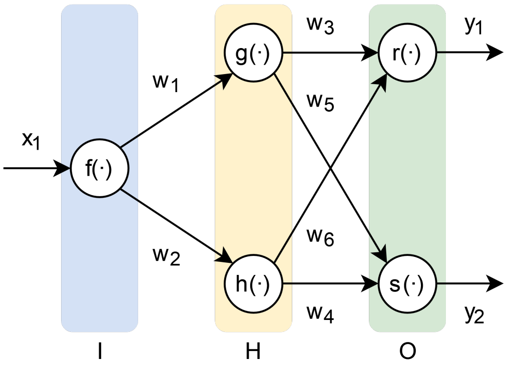


Figure 2.1: Example of an MLP with one hidden layer.

2.1.2 Dynamics

The dynamics of an MLP can be defined as the flow of information from the input to the output, a signal that propagates through the entire network. If the data to be processed are d–dimensional, $X = (x_1, x_2, \dots, x_d)$, the MLP has d input neurons ($|I| = d$). The neurons that belong to the layer I act as buffers, so their job is to pass the signal from the input to the neurons of the first hidden layer. Let i be a generic neuron. It receives as many signals, $o_{i_1}, o_{i_2}, \dots, o_{i_n}$ as the number of its incoming connections. Then, the activation function, $f(\cdot) : \mathbb{R} \to \mathbb{R}$, associated with unit i will produce a scalar, based on the weighted sum of all the input signals:

$$a_i = \sum_{j=1}^{n} w_{ij} \cdot o_{ij} \, , \tag{2.1}$$

where w_{ij} are the weights. Finally, we obtain the output of neuron i, $y_i = f(a_i)$ that will be propagated to the next layer. This mechanism will be repeated for each neuron in each layer up to the output neurons. Let k be an output neuron ($k \in O$), y_k

is assumed to be the k–th output of the network. If the MLP has m output neurons, its outputs are considered as an m–dimensional vector $Y = (y_1, y_2, \dots, y_m)$.

2.1.3 Learning

For the sake of simplicity, we initially consider a single neuron network. In the ANN, each input x_i is multiplied by the corresponding weight w_i, then a term b, called *bias*, is added and, finally, all the contributions are summed (see Figure 2.2). At this point, the weighted sum will be used as the input to the activation function, that will provide the output of the neuron.

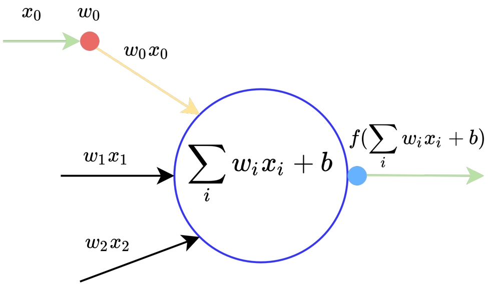


Figure 2.2: Perceptron.

As mentioned above, neurons can be grouped into one or more layers making an MLP. Let w_{ij} be the weight of the connection between the i–th unit of layer l and the j–th unit of layer $l-1$; the output of neuron i, y_i, can be calculated as:

$$y_i = f(a_i) = f(\sum_{j=1}^{n(l-1)} w_{ij} \cdot x_j) \tag{2.2}$$

where $n(l)$ indicates the dimension of layer l. If $f : \mathbb{R} \to \mathbb{R}$ is linear, then y_i can be rewritten as:

$$y_i = \sum_{j=1}^{n(l-1)} w_{ij} \cdot x_j = W^T \cdot X \tag{2.3}$$

where $W \in \mathbb{R}^{n(l-1),n(l)}$. Given a *training set* $\tau = \{(X_p, Y_p) : X_p \in \mathbb{R}^d, \hat{Y}_p \in \mathbb{R}^m,\ p = 1, \dots, P\}$, we need to determine an error measure over the training samples with an appropriate Loss Function (LF). There are several LFs in the literature: Mean Squared Error (MSE), Binary Cross Entropy, Categorical Cross Entropy, Hinge Loss, etc. Obviously, each of these LFs has its own characteristics, so that in the design phase of the network it is essential to choose the appropriate function to obtain optimal results. Let us consider, for simplicity, a network without hidden layers and the MSE loss. Therefore, we have a function $L(\tau, W)$, where $W = \{w_{ij} | i = 1, \dots, m; j = 1, \dots, d\}$. The LF can be defined as $L(\tau, W) = \frac{1}{2}\sum_p \sum_{i=1}^{m} (\hat{y}_i - y_i)^2$, with $p = 1, \dots, P$

and $P = |\tau|$. To train the network and thus minimize the LF, we use a gradient descent procedure. At each step t, the weights are updated as follows:

$$w_{ij}(t+1) = w_{ij}(t) + \Delta w_{ij}(t), \tag{2.4}$$

where $\Delta w_{ij}(t)$ depends on the partial derivative of the loss w.r.t. the weight w_{ij}:

$$\Delta w_{ij}(t) = -\mu \frac{\partial L}{\partial w_{ij}}. \tag{2.5}$$

The value μ is called *learning rate* and defines the step size at each iteration. There are two different learning approaches: *batch* and *on–line*. The main difference between these two approaches is that, in batch mode, the weights are updated after the presentation of all the training patterns to the network, whereas for on–line learning such a change is realized after each training sample. The repeated application of Eq. (2.4) implements the Gradient Descent iterative algorithm, which allows to find a local minimum in the loss function.

Let us now describe how the gradient of Eq. (2.5) can be calculated in networks with many layers. Thus, let $f_i(a_i)$ be the activation function associated with the i–th output unit, then for on–line weight updating, we can calculate $\frac{\partial L}{\partial w_{ij}}$ as follows:

$$\begin{aligned} \frac{\partial L}{\partial w_{ij}} &= \frac{\partial}{\partial w_{ij}} \left\{ \frac{1}{2} \sum_{k=1}^{m} (\hat{y}_k - y_k)^2 \right\} = \frac{1}{2} \sum_{k=1}^{m} \frac{\partial}{\partial w_{ij}} (\hat{y}_k - y_k)^2 \\ &= \frac{1}{2} \frac{\partial}{\partial w_{ij}} (\hat{y}_i - y_i)^2 = -(\hat{y}_i - y_i) \frac{\partial y_i}{\partial w_{ij}} \end{aligned} \tag{2.6}$$

From Eq. (2.5), we obtain:

$$\Delta w_{ij} = \mu (\hat{y}_i - y_i) \frac{\partial y_i}{\partial w_{ij}} \tag{2.7}$$

Let us proceed with the calculation of $\frac{\partial y_i}{\partial w_{ij}}$:

$$\begin{aligned} \frac{\partial y_i}{\partial w_{ij}} &= \frac{\partial f_i(a_i)}{\partial w_{ij}} = \frac{\partial f_i(a_i)}{\partial a_i} \frac{\partial a_i}{\partial w_{ij}} \\ &= f_i'(a_i) \frac{\partial}{\partial w_{ij}} \sum_{k=1}^{n(l)} w_{ik} x_k = f_i'(a_i) x_j. \end{aligned} \tag{2.8}$$

Finally, the so called *delta–rule* takes the form:

$$\Delta w_{ij} = \mu (\hat{y}_i - y_i) f_i'(a_i) x_j = \mu \delta_i x_j \tag{2.9}$$

having defined $\delta_i = (\hat{y}_i - y_i)f_i'(a_i)$. For any weight w_{jk} of any layer, the delta–rule described in Eq. 2.9 can be applied, based on the following definition for δ_j:

$$\delta_j = \begin{cases} (\hat{y}_i - y_i)f_j'(a_j), & \text{if } j \in L_l \\ (\sum_{i \in L_{k+1}} w_{ij}\delta_i)f_j'(a_j), & \text{if } j \in L_k \text{ with } k = l-1, \ldots, 0 \end{cases} \tag{2.10}$$

assuming a network with l layers, where L_0 denotes the input layer, $L_1, \ldots, L_{l-1}$ the hidden layers, and L_l the output layer. The *Back–Propagation* (BP) algorithm computes the gradient in this way, starting from the output layer and propagating the error signal back to the input layer.

Using the batch modality, the network training proceeds by *epochs*. An epoch consists in presenting all the patterns of the training set in the forward phase, back-propagating an error contribution which is the sum of all the errors accumulated during the epoch to update the weights.

Over time, additional modifications have been made to the BP algorithm, allowing, for example, to (partially) avoid local minima or to favor the generalization capacity of the network. Some example are as follows.

- *Weight–decay*: weigths are maintained numerically smaller which implies simpler solutions. The loss is defined as $L = \frac{1}{2}\sum_i(\hat{y}_i - y_i)^2 + \frac{\alpha}{2}\sum_{i,j}(w_{ij}^2)$, where the second term is the *regularization term*. For a generic w we obtain:

  $$\Delta w = -\mu\frac{\partial L}{\partial w} = -\mu\frac{\partial}{\partial w}\left[\frac{1}{2}\sum_i(\hat{y}_i - y_i)^2\right] - \mu\alpha w \tag{2.11}$$

  that is, when calculating the new value of w, in addition to the usual Δw due to the delta–rule, a $\mu\alpha$ portion of the same w is subtracted.

- In order to make learning more stable, an inertia or *momentum term* can be introduced into the delta–rule:

  $$\Delta w(t+1) = -\mu\frac{\partial L}{\partial w(t)} + \rho\Delta w(t), \quad \rho \in (0,1) \tag{2.12}$$

  The momentum term allows us to overcome the risk to be trapped in small local minima. Indeed, momentum helps in reducing the noise in gradient update term and thus helps to converge faster to the optimal (or near optimal) value.

2.2 Deep Learning

The term Deep Learning comes from the depth of the network used and thus the number of layers present in the designed architecture. An MLP with a single hidden layer is a universal approximator (Cybenko, 1989): under appropriate assumptions on the activation functions, an MLP can approximate any continuous function on a compact subset of $\mathbb{R}^n$. DNNs extend this propriety of MLPs and are able to approximate more complex functions (Bianchini and Scarselli, 2014). In this Section, we describe some DNN architectures that are particularly used in Computer Vision tasks.

2.2.1 Convolutional Neural Networks

Convolutional Neural Networks (CNNs) are one of the most famous and widely used DNNs, particularly in the field of computer vision. The term "convolutional" refers to the mathematical operator that is used instead of the simple matrix multiplication. This kind of architecture is very useful for processing data with a grid structure, such as 2D images. First, we need to see what a convolution is, from a mathematical point of view, and then how this operation is used in some layers of a CNN. A convolution between two functions f and g is indicated by $f * g$ and is defined as:

$$(f * g)(t) = \int f(\tau)g(t-\tau)d\tau\,. \tag{2.13}$$

Intuitively, the above value represents the area under the function $f(\tau)$ weighted by the function $g(-\tau)$ shifted by an amount t. Of course, we need to discretize the time t, since we process data on a computer. So we have:

$$s(t) = (x * w)(t) = \sum_{a=-\infty}^{+\infty} x(a)w(t-a), \tag{2.14}$$

where x and w are defined only on integer t, and are often referred to as the **input** and the **kernel**, respectively. In machine learning applications to computer vision, both the input and the kernel are multidimensional arrays, namely *tensors*. As we mentioned earlier, in computer vision, if we consider 2D images, we would also like to have a bidimensional kernel. Therefore, if we consider a bidimensional image I as input, we have:

$$S(i,j) = (I * K)(i,j) = \sum_m \sum_n I(m,n)K(i-m,j-n), \tag{2.15}$$

where (m,n) is the grid dimension, and for the commutative property of convolution we can write

$$S(i,j) = (K * I)(i,j) = \sum_m \sum_n I(i-m,j-n)K(m,n). \tag{2.16}$$

Finally, if we don't flip the kernel, we get a function implemented in many neural network libraries, called *cross–correlation*

$$S(i,j) = (K * I)(i,j) = \sum_m \sum_n I(i+m,j+n)K(m,n). \tag{2.17}$$

Just as we have established a correlation between a brain neuron and a single perceptron, we can observe a similarity between CNNs and our visual cortex. In particular, we can observe a similar connectivity pattern. Indeed, a single neuron responds to stimuli in a narrow region of the visual field, known as its *receptive field*, and a set of such overlapping fields covers the entire visual area. This is what, more or less, happens in CNNs.

A convolutional layer consists of three blocks. In the first block, a convolution operation is performed which produces linear activations. Subsequently, always taking the images as an example, some non–linear operations are calculated, using activation functions such as sigmoid, hyperbolic tangent or, as more often happens, the ReLU function. In the third block, a pooling function is used to further modify the output of the layer. The pooling function replaces the output of the convolutional layer by computing some statistics in a rectangular neighborhood of the output. Two common functions used in the pooling operation are: average pooling and max pooling. The former returns the average of a neighborhood of the output, the latter computes its maximum value. The pooling functions allow to reduce the spatial dimension of the representation, and consequently to reduce the computational cost of the operations to be performed later. Apart from their practical implementation, the most important properties that have made CNNs crucial in the Computer Vision field are: their capacity to produce **sparse interactions**, to realize **parameter sharing** and **equivariant representations**. Let us discuss these three fundamental aspects of CNNs.

- *Sparse interactions* – Using a smaller kernel than the input avoids the interaction of all output units with all input units. For this reason, this situation is also called sparse connectivity. In this way, not only a reduction in memory requirements is obtained, since there are less parameters to store, but also a significant decrease in the number of operations and, consequently, in the computational cost. By considering the example of images, we can immediately see the improvement we get by using this approach. An image can have thousands or millions of pixels but, with a kernel composed of a limited number of parameters, we can detect important image features, such as edges, at a reduced computational cost.

- *Parameter sharing* – In a traditional ANN, each element of the weight matrix is used once. In CNNs, the main assumption is the following, if at a certain location (x_1, y_1) a given weight is useful to compute a feature, then it should also be useful at another location (x_2, y_2). Thus, some neurons will be forced to share the same weights and biases.

- *Equivariant representations* – One consequence of parameter sharing is equivariance to translation. This means that if an input image is translated by a certain amount, the output feature map is translated by the same amount. This cannot be said for other transformations, such as rotations and scale changes. Therefore, to capture such a variability, we need to modify the input data with multiscale resize, flips and rotations, increasing the number of training images.

2.2.2 Semantic Segmentation

With the advent of DNNs and particularly CNNs, the number of studies performed on images has grown exponentially. Humans are able to classify images, detect objects and segment. This list of possible activities is sorted by difficulty. In fact, if in image classification, the goal is to return a label for each image, a further step is needed for the localization of the objects in an object detection task. Finally, with the segmentation, we try to classify every single pixel. There are two types of segmentation techniques:

- **Semantic Segmentation**: it consists of classifying each pixel assigning it a particular label, without making distinctions between different instances of the same object.
- **Instance Segmentation**: we try to assign a unique label to each instance of a particular object in the image. Therefore, if there are three cars in an image, each car will have its own label, while in the case of semantic segmentation each car would have the same label.

Before DNNs, techniques such as SVMs, K–Means Clustering, and Random Forest were used to solve image segmentation problems. However, the results obtained by DNNs have virtually eliminated competition from these classical techniques. Let us see which networks are most commonly used for the semantic the segmentation problem.

Fully Convolutional Network — The general architecture of CNNs consists of convolutional and pooling layers, followed by fully connected layers, to obtain the network output. Fully Convolutional Networks (FCNs) (see Figure 2.3) are used for classification tasks (e.g. AlexNet, VGGs and GoogLeNet), and they must be suitably modified to be transformed into semantic segmentation networks. The main difference concerns the last layer, as it is replaced by a 1×1 convolutional layer that covers the entire image. Indeed, it has been shown that exchanging the final dense layer with a convolutional layer yields the same or even better results. The main advantage of FCNs, however, is their ability to process any type of image, in terms of size, without the need of a predefined dimension. This removes the constraint on image size that is present in CNNs with final fully connected layers. In fact, when dealing

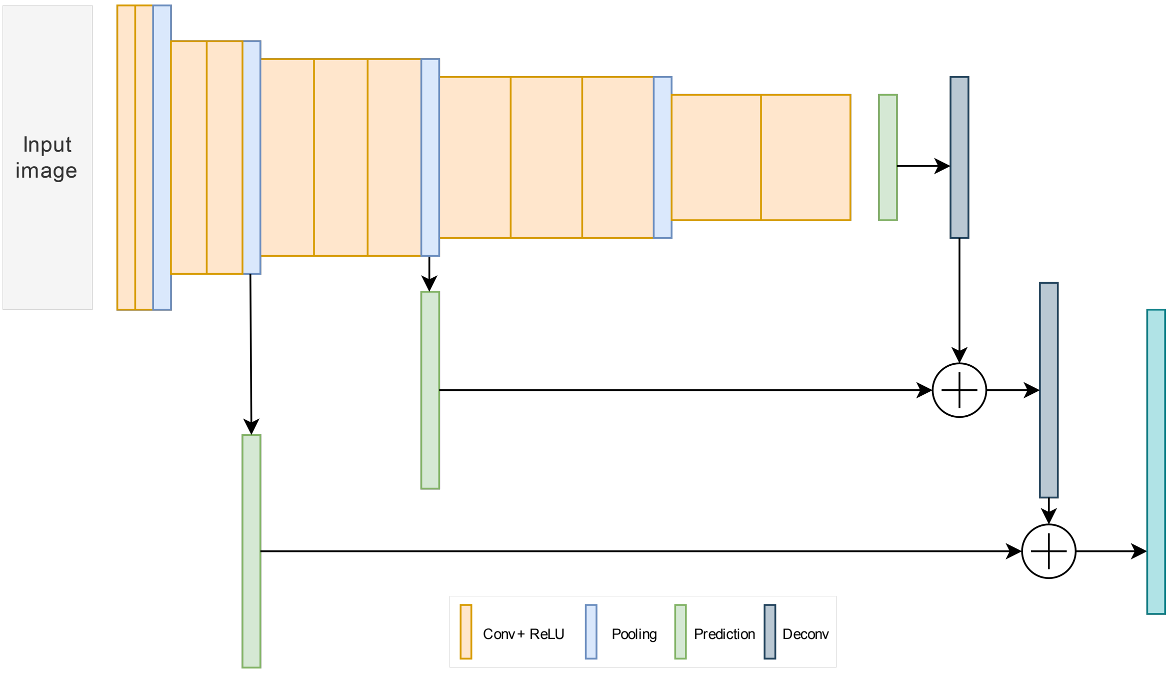


Figure 2.3: Scheme of the Fully Convolutional Network.

with dense layers, the size of the input is constrained and thus, when an input image has a different size, it must be resized. Even when a larger image is provided as input, the output produced will be a feature map. Moreover, the final feature map represents not just a class but a heatmap of the requested class. Obtaining the location of the objects with this heatmap is an useful information for the segmentation task. Convolutions in the final layer produce a down–sample (encoder), so that we need an up–sample (decoder). Interpolation techniques work well, but the authors of the FCN model argued that in–network upsampling is fast and effective for learning dense prediction. In this way, we are able to learn non–linear up–sampling as well. Going into more detail, the decoder employs transposed convolution for up–sampling and uses skip connections between layers at different resolution to recover details that have been lost due to the use of sub–sampling layers.

U–Net — The U–Net model (Ronneberger et al., 2015) is based on FCNs, so it contains a down–sample phase and an up–sample phase. These two phases of contraction and expansion form a "U" shape, from which the architecture takes its name (see Figure 2.4). The main advancement of the U–Net architecture consists in the inclusion of shortcut connections. To solve the problem of information loss during the down–sample phase, we concatenate the features of the decoder with the corresponding maps of the encoder. In particular, since the layers at the beginning of the encoder have more information, they would be able to support the up–sampling operation of the decoder by providing fine details corresponding to the input images, thus greatly improving the results. Another important contribution of this architec-

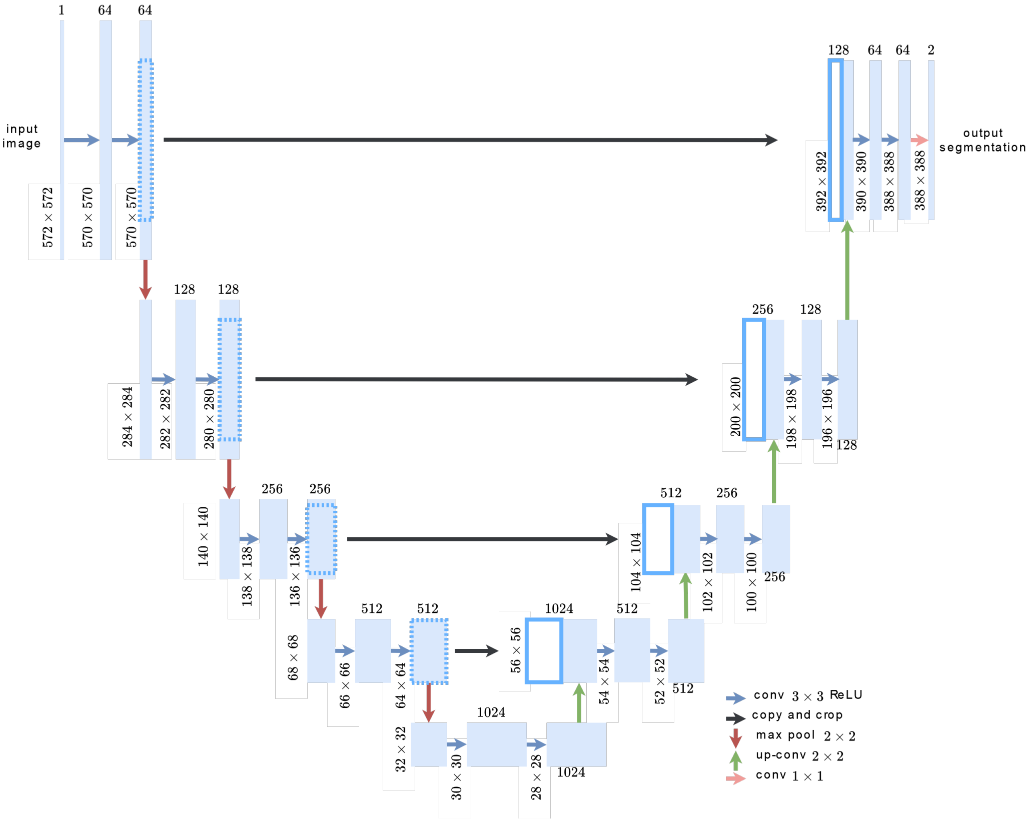


Figure 2.4: Scheme of the U–Net.

ture concerns the loss function used. It is a kind of "improved" Cross–Entropy, in which there is a weighted loss for each pixel in order to have higher weights on the edges of the segmented object.

DeepLab — A group of researchers from Google – DeepLab – proposed a set of techniques to improve semantic segmentation while trying to reduce the network complexity. The three main improvements of the DeepLab network are: atrous convolutions, Atrous Spatial Pyramidal Pooling (ASPP) and Conditional Random Fields (CRFs) (Chen et al., 2017a). As we mentioned earlier, the main problem with FCNs is the down–sampling phase, where we have a large loss of information. With atrous convolutions or dilated convolutions, we can process a large context using the same number of parameters. In simple terms, dilated convolutions increase the size of the filter by inserting "holes" in the filter, corresponding to zeros in the parameters. The dilation rate d indicates the number of zeros to be added between the parameters. Therefore, with $d = 1$ we have a classic convolution, with $d = 2$ we insert a zero between two parameters, then from a 3×3 filter we pass to a 5×5 filter, and so on. ASPP is an improvement of the Spatial Pyramidal Pooling (SPP) network and exploits atrous convolution. The SPP network permits to overcome the constraints on the fixed size input dimension of CNNs by adding an SPP layer on top of the last convolution layer. In this way, the features are grouped together and fixed–length outputs are generated, that are later used as input to fully connected layers. In ASPP, different dilation rates are used for the inputs and the outputs are fused together, obtaining information from different scales which allows to achieve better results. Finally, a Conditional Random Field (CRF) operates a post–processing step. With the CRF we obtain a more refined segmentation of object edges than that produced by a pooling operation. This is possible by exploiting not only the label of the pixel to be classified, but also that of the neighboring pixels.

Global Convolution Network — Semantic segmentation can be viewed as a competitive game between classification and location. Indeed, while classification networks are invariant with respect to rotations and translations, they do not give any importance to object positions, which contradicts the task of determining the object location for the final segmentation. However, most segmentation algorithms give more importance to location. Global Convolution Networks (GCNs) try to balance these two aspects using the classification part more, but without diminishing the contribution of the localization task. This is made possible by the introduction of GCN blocks. Each of these blocks employs a combination of $1 \times k + k \times 1$ and $k \times 1 + 1 \times k$ convolutions, which allows for dense connections within a large $k \times k$ region in the feature map. Thus, as the value of k increases, more and more context is captured. In (Peng et al., 2017), also based on (Zhou et al., 2014), it is argued that GCNs are able to obtain information from much smaller regions of the receptive field, which are called Valid Receptive Fields (VRFs). So, the kernel size of the convolutional structure should be as large as possible. By using symmetric and separable filters, the number of model parameters is reduced so as the computational cost.

Pyramid Scene Parsing Network — The Pyramid Scene Parsing Network (PSPNet) leverages both local and global information to make the final decision. A pretrained CNN with dilated convolution is used to extract the feature map and, on top of it, we have a Pyramid Pooling Module (PPM). A PPM solves a typical problem of CNNs: when the receptive field is larger than the input image, the empirical receptive field is smaller than the theoretical one, especially in the higher layers. PPM contains information with different scales which varies among different sub–regions, successfully incorporating the global scenery prior. This is possible by merging features on different pyramid scales, from the coarsest to the finest. Thus, the outputs of the different layers of the PPM represent features of different size and, to maintain the overall weight of the features, use a 1×1 convolution after each layer, to reduce the size of the context representation to $1/N$ from the original size (if the pyramid layer size is N).

SMANet — The Segmentation Multiscale Attention Network (SMANet) is a deep fully convolutional neural network with a ResNet backbone encoder. In the SMANet architecture, a convolutional decoder is employed to recover fine details, which are lost due to the presence of pooling and strided convolutions. A multiscale attention mechanism is also used to focus on the most informative part of the image. This topic will be dealt with in more detail later in the thesis.

2.2.3 Generative Adversarial Networks

A common limit of deep learning is the large amount of data needed to train a model. The number of parameters that the model must learn is usually large, so that also a large amount of data is necessary. For these reasons, we need either more data or a method that allows us to enlarge the training set. A few classic and widely used data augmentation techniques are:

- *Flip* – horizontally or vertically.
- *Rotation* – we can rotate the image by any angle, but taking care to preserve the image dimensions.
- *Scale* – the image can be scaled outward or inward.
- *Crop* – crop differs from scaling because in this case only a portion of the image is taken and then resized to the original image size. A crop is often cut randomly and, therefore, the related procedure is called random cropping.
- *Translation* – simply moves the image along the X and Y direction (or both). This method is very useful for forcing CNNs to look anywhere in the image, since objects can be anywhere.
- *Gaussian Noise* – adding the right amount of noise can improve the learning capability. Most of the time, overfitting may be due to patterns occurring too often. Gaussian noise distorts frequent patterns.

These classic techniques often work well, but the image distribution they generate is limited. Generative Adversarial Networks (GANs) provide an alternative solution to generate images similar to those available in a real dataset. A GAN (see Figure 2.5) (Goodfellow et al., 2014) is an architecture that uses two neural networks, a generator G and a discriminator D, which are trained one against the other, hence the term “adversarial”. G is trained to map a latent random variable $\mathbf{z} \in \mathbb{R}^Z$ into a fake image $\tilde{\mathbf{x}} = G(\mathbf{z})$, whereas D aims at distinguishing the fake samples from the real ones, $\mathbf{x} \in p_r(x)$. In other words, D and G play the following two–player minmax game with respect to the function $V(D, G)$:

$$\min_G \max_D V(D,G) = \mathbb{E}_{\mathbf{x}\sim p_r(x)}[\log D(\mathbf{x})] + \mathbb{E}_{\mathbf{z}\sim p_z(\mathbf{z})}[\log(1 - D(G(\mathbf{z})))] \tag{2.18}$$

The goal of this competitive game between G and D is to achieve the Nash equilibrium (Nash Jr, 1950). While the discriminator has to figure out if an image belongs to the training set or is a fake, the generator creates new synthetic images to try to fool the discriminator. The steps are the following: the generator returns an image from a random vector; this generated image is fed as input to the discriminator, along with an image stream taken from the current ground–truth dataset. Finally, the discriminator returns a probability value, where 1 represents a prediction of authenticity and 0 represent a fake image. Let us see which networks are most commonly used for synthetic image generation.

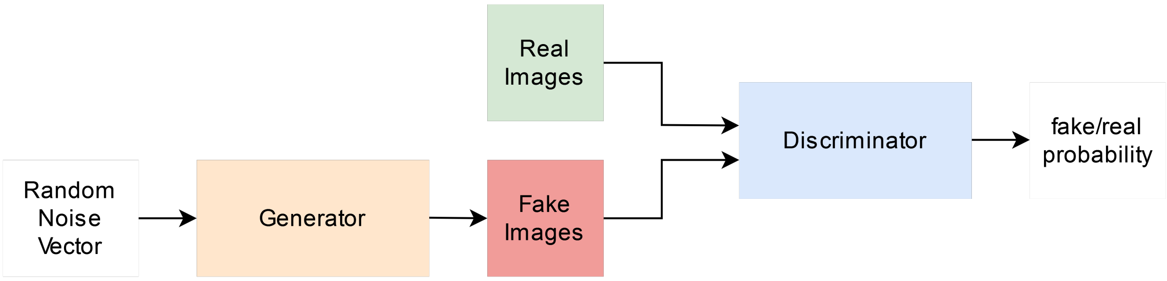


Figure 2.5: Scheme of a Generative Adversarial Network.

Progressive Growing of GAN — The Progressive Growing of GAN (PGGAN) (see Figure 2.6) (Karras et al., 2017) is an extension of the GAN architecture that allows a more stable training and is capable of producing HD images. This is possible by starting the training with very small images, adding blocks that increase the size of the generator output and the input size of the discriminator, until the desired image size is reached. Once the generator has reached a resolution of 16×16, a new layer to generate 32×32 images is added. An up–sample of the previously generated image (16×16 pixels) is performed and the two images are summed by weighting the two terms by α and $1 - \alpha$. When the new 32×32 output layer is added to the network, the output of the 16×16 layer is projected onto the 32×32 dimension with a simple nearest neighbor interpolation. More precisely, the projected layer is multiplied by $1 - \alpha$ and concatenated with the new output layer, multiplied by α, to form the new 32×32 generated image. The α parameter scales linearly from 0 to 1 and when reaches 1, the nearest neighbor interpolation from the 16×16 image is not taken into account. This smooth transition mechanism greatly stabilizes the training.

StyleGAN — The PGGAN architecture was successively modified obtaining two different versions of the StyleGAN. In the first version (Karras et al., 2019), the changes affect only the generator (see Figure 2.7), while the discriminator remains unchanged. The first change is the inclusion of a *Mapping Network* (MP). This network takes in input a latent vector z and produces another vector w that will be the input for the synthesis network. The goal of the MP is to create an intermediate

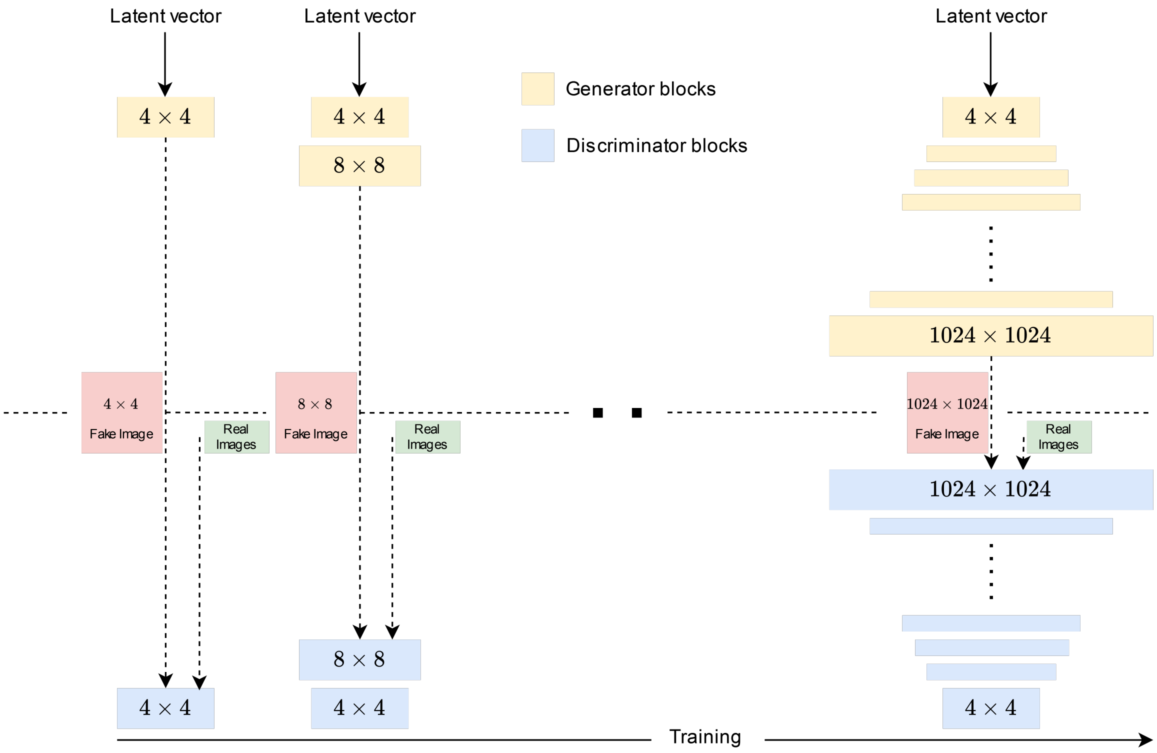


Figure 2.6: Scheme of the PGGAN.

vector whose different elements control different features. Then, also some Style Modules were introduced. After each convolutional block there is another convolutional block that performs an Adaptive Instance Normalization (AdaIN). The vector w, i.e., the output of the MP, is called a Style Vector and is given as input to each AdaIN block. Before doing this, however, it is further transformed through a Fully–Connected layer (called A). In addition to this transformed vector, the output of each normalized convolution is put as input in the AdaIn block. Compared to, for example, Batch Normalization, AdaIN has no parameters to learn. Thus, the vector w controls some parameters, called scaling and shifting parameters (Y_s and Y_b), of the feature normalization. Another difference between PGGAN and StyleGAN is represented by their input. Most of the models, as well as PGGAN, use a random input to create the initial image; instead, the StyleGAN uses a vector of constant values. This is because the features of the images are controlled by w and AdaIN, so the initial random input can be omitted and replaced by constant values. One final update introduced in the StyleGAN concerns the introduction of noise. In many cases, it is difficult to control the effect of noise due to feature entanglement, i.e., the model cannot map part of the input to the features. So, some noise was added to each channel before the AdaIN block, obtaining single–channel images with Gaussian noise (different for each layer). Then, some noise is also added after each convolutional layer, to have variations that are also seen in real life, such as the hair placement in

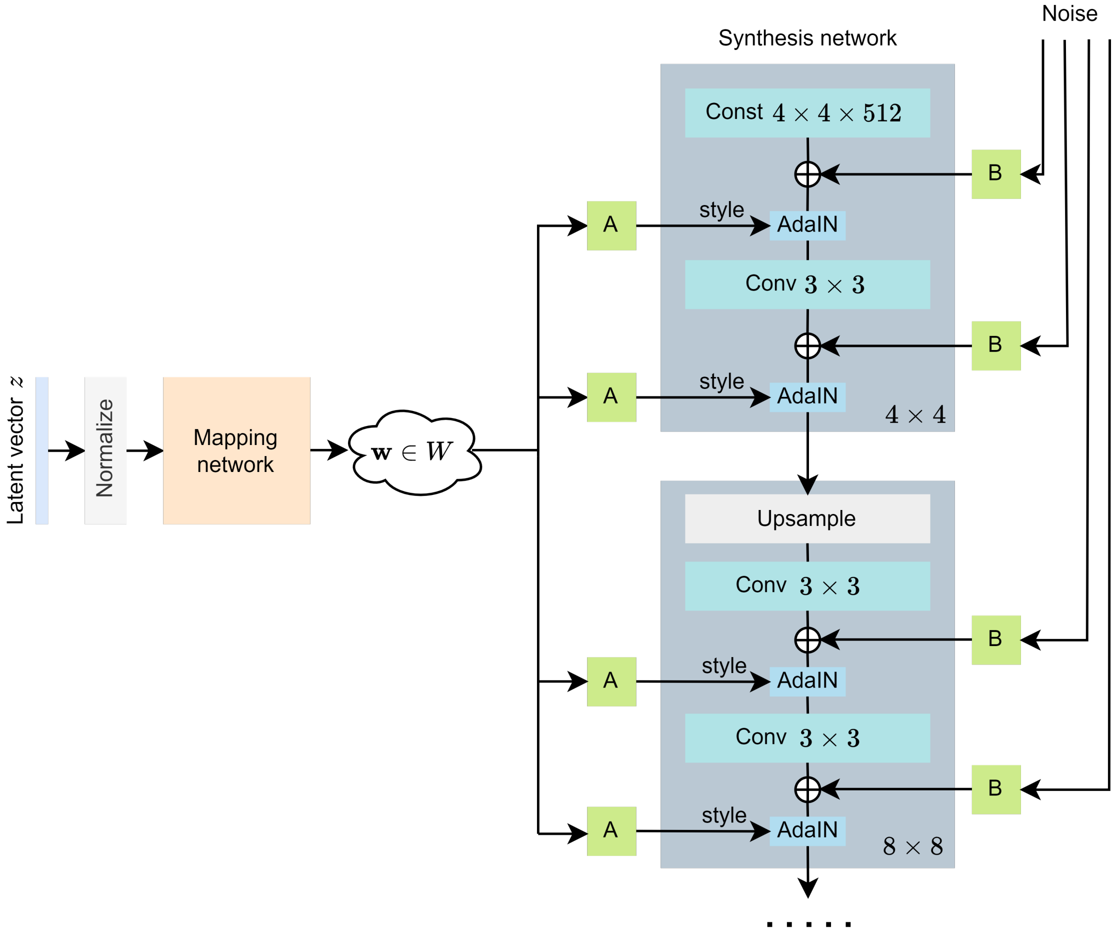
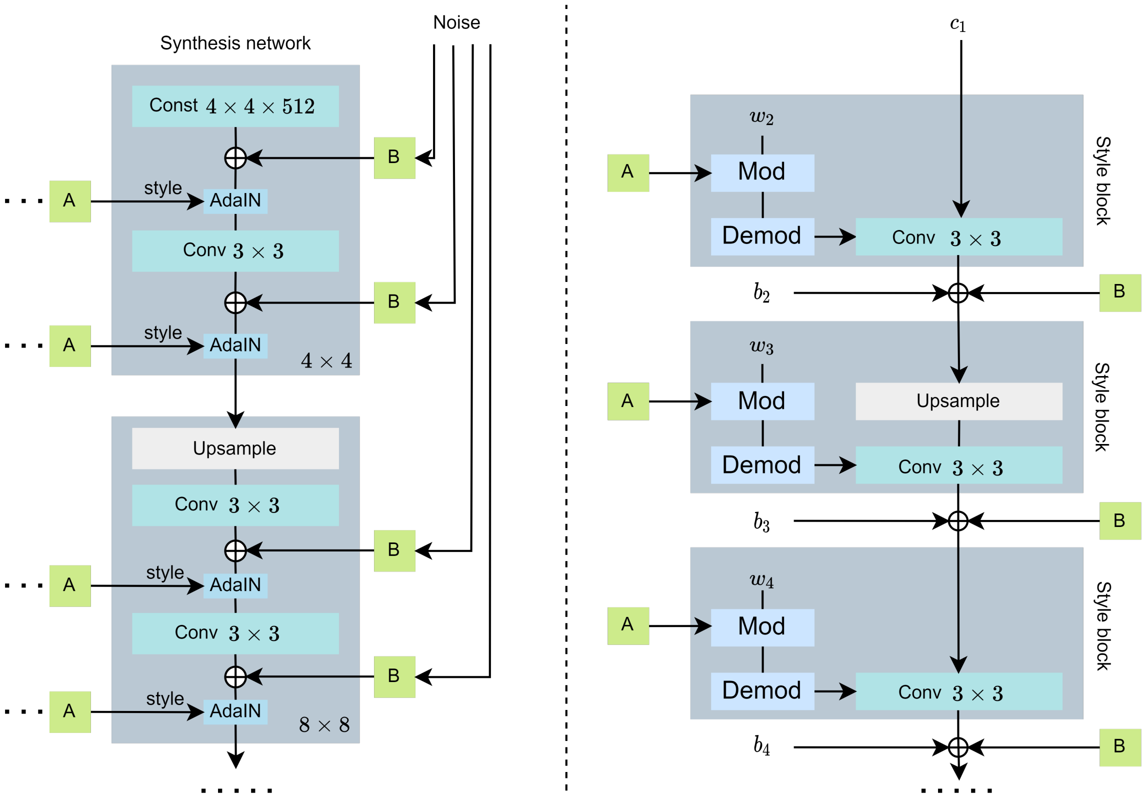


Figure 2.7: Scheme of the StyleGAN.

the generation of artificial faces. The main advantage of this architecture is that, by using different style vectors at different points in the synthesis network, it is possible to control the “style” of the resulting image, with various levels of detail. The results obtained are very good, but there are some problems, solved by the second version.

StyleGANv2 — Two main problems afflict StyleGANs. The first one is the production of spots that look like water drops in the image, which can appear anywhere. The second, again using the problem of synthetic face generation as a case study, is the position of the eyes and mouth, which remains almost always the same, or the teeth that do not follow the pose of the face. According to the authors, the water droplet problem is due to the AdaIN block. In fact, both the mean and variance of each feature map are normalized, potentially destroying any kind of information regarding the magnitude (order of magnitude, absolute value) of the features. To overcome this problem, the noise was moved out of the Style Block, while AdaIN was substituted by a different approach, called demodulation. A Style Block consists of modulation, convolution, and normalization (see Figure 2.8). After modulation and convolution, the outputs are scaled by the L_2 norm of the corresponding weighs. The goal of demodulation is to restore outputs with unit standard deviation. Another technical change is the addition of the Perceptual Path Length Regularization (PPLR) to the loss function of the generator, to enforce smoother latent space interpolation. The idea is that, if the latent vector is slightly changed, also a smooth change in the semantic of the generated images occurs. As mentioned above, StyleGAN images have a strong location preference for facial image features like noses and eyes, which is attributed to the progressive growing architecture. Inspired by Multi–Scale Gradient for Generative Adversarial Networks (MSG–GAN), a new architecture was designed to solve the problem. Better results are obtained but the total complexity of the architecture increases significantly.



(a) StyleGAN

(b) StyleGANv2

Figure 2.8: Comparison between the original StyleGAN and StyleGANv2.

2.2.4 Image–to–image Translation

Let us consider for a moment the problem of classifying landscapes within an image. As we know, a landscape can be composed of an enormous variety of plant species. What the neural network does not know is that some landscapes exist only under certain conditions, for example depending on seasons. Without this knowledge, a classifier might mistake the shores of a frozen lake for a glacier. To overcome this problem one could take pictures of the same landscape at different times of the year, but this requires a huge amount of work. In addition to the data augmentation techniques seen above, there are other GANs, suitably modified, that allow us to translate an image from one domain to another. These particular kind of GANs are called Conditional GANs (CGANs). For an example, Pix2Pix (Isola et al., 2017) is a CGAN that operates with supervision, and Pix2PixHD (Wang et al., 2018) employs a coarse–to–fine generator and discriminator, along with a feature–matching loss function, to translate images with higher resolution and quality.

2.3 Short review on image generation and segmentation

Since this thesis is focused on image generation and segmentation, in this section we will present a brief review of the literature on these topics.

2.3.1 Synthetic Image Generation

Methods for generating images can be classified into two main categories: model–based and learning–based approaches. The most conventional procedure is to formulate a model of the observed data and to render the images using a dedicated engine. This approach has been used, for example, to extend the available datasets of driving scenes in urban environments (Richter et al., 2016; Ros et al., 2016) or for object detection (Hodaň et al., 2019). In the field of medical image analysis, synthetic image generation has been extensively employed. For example, realistic digital brain–phantom has been synthesized in (Collins et al., 1998), while more recently, synthetic agar plate images have been generated for image segmentation (Andreini et al., 2018, 2020). The design of specialized engines for data generation requires an accurate model of the scene and a deep knowledge of the specific domain. For this reason, in recent years, the learning–based approach has attracted increasing research resources. In this context, machine learning techniques are used to capture the intrinsic spatial variability of a set of training images, so that the specific domain model is acquired implicitly from the data. Once the probability distribution that underlies the set of real images has been learned, the system can be used to generate new images that are likely to mimic the original ones.

If the synthetic images are close enough to the real ones, they can be used to enlarge existing datasets for training machine learning models. For example, GANs have been used in (Kugelman et al., 2021) to augment data for a patch–based approach to OCT chorio–retinal boundary segmentation. In (Waheed et al., 2020), synthetic chest X–ray (CXR) images are generated by developing an Auxiliary Classifier Generative Adversarial Network (ACGAN) model, called CovidGAN. Synthetic images produced by CovidGAN were used to improve the performance of a standard CNN for detecting COVID–19. In (Frid-Adar et al., 2018), different GANs have been used for the synthesis of each class of liver lesion (cysts, metastases and hemangiomas). In (Hu et al., 2018), Wasserstein GANs (WGANs) and InfoGANs have been combined to classify histopathological images, whereas in (Yi et al., 2018) WGAN and CatGAN generated images were used to improve the classification of dermoscopic images

In (Shin et al., 2018), synthetic abnormal MR images containing brain tumors are generated. An image–to–image translation algorithm is employed to construct semantic label–maps of real MR brain images, distortions are introduced on the generated segmentation (i.e., tumors are shrunk or enlarged, or their position is changed), and then the segmentation is translated back to images. Indeed, manually introducing distortions on the generated label–maps is not trivial because they can alter the image semantic — for instance, in the case of retinal image generation, enlarging or reducing blood vessels is not meaningful. We solve this issue directly by learning the semantic label–map distribution with a GAN.

2.3.2 Image–to–Image Translation

Recently, beside image generation, adversarial learning has also been extended to image–to–image translation, in which the goal is to translate an input image from one domain to another. Many computer vision tasks, such as image super–resolution (Ledig et al., 2017), image inpainting (Pathak et al., 2016), and style transfer (Gatys et al., 2015) can be casted into the image–to–image translation framework. Both unsupervised (Liu et al., 2017; Liu and Tuzel, 2016; Yi et al., 2017; Zhu et al., 2017) and supervised approaches (Isola et al., 2017; Karras et al., 2017; Chen and Koltun, 2017) can be used. For the proposed applications the unsupervised category is not relevant. Supervised training uses a set of pairs of corresponding images $\{(s_i, t_i)\}$, where s_i is an image of the source domain and t_i is the corresponding image in the target domain. In addition to the previously mentioned contributions, belong to the class of supervised approaches also the most recent BycicleGAN (Zhu et al., 2018), SIMS (Qi et al., 2018), and SPADE (Park et al., 2019) architectures.

2.3.3 Semantic Segmentation

The goal of semantic image segmentation is to infer the class of each pixel in an image. Many studies are aimed at semantic segmentation of natural scenes with fully convolutional deep neural networks (Long et al., 2015; Zhao et al., 2017; Chen et al., 2017a). Providing pixel–level oversight is difficult and expensive, however, relatively large datasets have been created for segmentation in natural images: for example, PASCAL VOC 2012 (Everingham et al., 2015) and MS-COCO (Lin et al., 2014), which collectively contain more than 100000 images with pixel–wise annotations. In medical imaging, the number of available samples is generally smaller, and the only viable alternative seems to be the use of small networks with a reduced number of parameters. In fact, one of the most successful deep learning methods in biomedical imaging is the U–Net architecture (Ronneberger et al., 2015), which uses a standard convolutional network, followed by an up–sampling part of up–convolutions combined with skip–connections.

Chapter 3

Two–stage image generation

This chapter presents the first approach to generating image data to be used for semantic segmentation. We show that by dividing the procedure into two steps, the generation task is considerably simplified, while it is possible to obtain good quality images with fewer examples. The approach is applied to the segmentation of retinal images, an important task in medicine with several possible applications, not limited to ophthalmology.

The retinal microvasculature is the only part of human circulation that can be directly and non–invasively visualized in vivo (Patton et al., 2006). Hence, it can be easily acquired and analyzed by automatic tools. As a result, retinal fundus images have a multitude of applications, including biometric identification, computer-assisted laser surgery, and the diagnosis of several disorders (Fraz et al., 2012a; Patil and Manza, 2016). One important processing step in such applications is the proper segmentation of retinal vessels. Image semantic segmentation aims to make dense predictions by inferring the object class for each pixel of an image and, indeed, the segmentation of digital retina images allows us to extract various quantitative vessel parameters and to obtain more objective and accurate medical diagnoses. In particular, the segmentation of retinal blood vessels can help the diagnosis, treatment, and monitoring of diseases such as diabetic retinopathy, hypertension, and arteriosclerosis (Kanski and Bowling, 2015; Abràmoff et al., 2010). Most of the leading approaches for semantic segmentation, in fact, rely on thousands of supervised images, while supervised public datasets for retinal vessel segmentation are very small (most datasets contain fewer than 30 images).

To face the scarcity of data, we propose a new approach for the generation of retinal images along with the corresponding semantic label–maps. Specifically, we propose a novel generation procedure based on two distinct phases. In the first phase, a generative adversarial network (GAN) (Goodfellow et al., 2014) generates the blood vessel structure (i.e., the vasculature). The GAN is trained to learn the typical semantic label–map distribution from a small set of training samples. To generate high–resolution label–maps, the Progressively Growing GAN (PGGAN) (Karras et al., 2017) approach has been employed. In a second, distinct phase, an image–to–image translation algorithm (Wang et al., 2018) is used to translate blood vessels structures into realistic retinal images.

The rationale behind this approach is that, in many applications, the semantic structure of an image can be learned regardless of its visual appearance. Once the semantic label–map has been generated, visual details can be incorporated using an image–to–image translation algorithm, thus obtaining realistic synthesized images. The benefits of using this generation strategy are listed below.

- **Reduced GPU memory requirement** – since the training is carried out in two separate stages, the computational demands of each individual step are reduced compared to the simultaneous generation of the label–map and the related image.

- **Reduced number of sample required** – by separating the whole process into two stages, the generation task is simplified.

- **Better control of the generation procedure** – each generation phase can be defined and fine–tuned independently, so that, for example, it is possible to use different architectures and different sets of hyperparameters.

- **Visually enhanced image quality** – the training is very effective and we obtained retinal images with unprecedented high resolution and quality, along with their semantic label–maps.

To assess the usefulness and correctness of the proposed approach, the generation procedure has been applied on two public datasets (i.e., DRIVE (Staal et al., 2004) and CHASE_DB1 (Fraz et al., 2012b)). Moreover, the two–step generation procedure has been compared with a single–stage generation, in which label–maps and retinal images have been generated simultaneously in two different channels. Indeed, in our experiments, the multi–stage approach allows us to significantly improve performance of vessels segmentation when used for data augmentation. In particular, the generated data have been used to train a Segmentation Multiscale Attention Network (SMANet) (Bonechi et al., 2020). Comparable results have been obtained by training the SMANet on the generated images in place of real data. It is interesting to note that, if the network is pre–trained on the synthesized data and then fine–tuned on real images, the segmentation results obtained on the DRIVE dataset come very close to those obtained by the best state–of–the–art approach (Sekou et al., 2019). If the same approach is applied to the CHASE_DB1 benchmark, the results overcome (to the best of our knowledge) those obtained by any other previously proposed method.

3.1 Retinal Images

3.1.1 Retinal Image Synthesis

One of the first applications of retinal image synthesis has been described in the seminal work (Sagar et al., 1994), in which an anatomic model of the eye and of the surrounding face has been implemented for surgical simulations. More recently, in (Fiorini et al., 2014), a large dictionary of small image patches containing no vessels, has been used to model the retinal background and fovea. A parametric intensity model, in which the parameters have been estimated from real images, is used to generate the optical disk. Complementary to (Fiorini et al., 2014), the contribution in (Menti et al., 2016) focuses on the generation of the vascular network, based on a parametric model, in which the parameters are learned from real vessel trees. While these methods give reasonable results, they are complex and heavily dependent on domain knowledge. To reduce the knowledge requirements, a completely learning–based approach has been proposed in (Costa et al., 2017a), where an image–to–image translation model has been employed to transform existing vessel networks into realistic retinal images. Vessel networks used for learning have been obtained using a suitable segmentation technique applied to a set of real retinal images. However, the quality of the generated images heavily depends on the segmentation module performance. In (Zhao et al., 2018), a generative adversarial approach, together with a style transfer algorithm, is used to reduce the need for annotated samples and to improve the representativeness (e.g., the variability) of synthesized images. The model still relies on pre–existing vessel networks (obtained manually or by a suitable segmentation technique). In (Costa et al., 2017b), an adversarial auto–encoder for retinal vessel synthesis has been adopted to avoid the dependence of the model on the availability of pre–existent vessel maps. Nevertheless, this approach is able to generate only low–resolution images, and the performance in vessel segmentation using synthesized data is far below the state–of–the–art. Higher–resolution retinal images, along with their segmentation label–maps, have been generated in (Beers et al., 2018), using Progressively Growing GANs (PG-GANs) (Karras et al., 2017). This method allows for the generation of images up to a resolution of 512×512 pixels. A set of 5550 images segmented by a pre–trained U–Net (Ronneberger et al., 2015) have been used during training. Unfortunately, the usefulness of the generation for image segmentation is not demonstrated.

The present thesis improves previous approaches generating synthetic images up to a resolution of 1024×1024 pixels. The generation is based on a very small set of pre–existing images (actually, 20 images with supervised segmentation maps). Both the retinal images and the corresponding semantic label–maps (the vasculature) are generated. Furthermore, we prove that combining real retinal images with synthesized ones for training a segmentation network improves the final segmentation performance.

3.1.2 Retinal Vessel Segmentation

During recent decades, several approaches for retinal vessel segmentation have been proposed, both supervised and unsupervised. Unsupervised methods depend heavily on prior knowledge on the vessel structure. For example, the so called vessel tracking techniques define an initial set of seed points and, thereafter, by chaining pixels that minimize a given cost function, iteratively extract the vasculature (Liu and Sun, 1993; Yin et al., 2012). In (Hoover et al., 2000), retinal images are convolved with a 2D filter to produce a Gaussian intensity profile of the blood vessels, that is subsequently thresholded to give the vessel map. Adaptive thresholding has been used in (Roychowdhury et al., 2015) and in (Neto et al., 2017). An active contour model that combines intensity and local phase information is used in (Zhao et al., 2015). In (Khan et al., 2020), a hybrid unsupervised approach was proposed. To obtain the vessel location map, the composition of two preprocessed images is fused with the enhanced image of B–COSFIRE filters followed by thresholding. Instead, an ensemble strategy automatically combining multiple segmentation results is presented in (Liu et al., 2019). Moreover, since the retinal blood vessels' diameter significantly changes based on the distance from the optic disc, multi–scale approaches can be particularly effective for the vessel segmentation (Khawaja et al., 2019; Shah et al., 2019). Supervised methods are currently the leading techniques in semantic segmentation. In this framework, true annotations are used to train a classifier aimed at distinguishing the vessels from the background. Various classification models have been employed for blood vessel segmentation based on a preliminary feature engineering stage (Niemeijer et al., 2004; Soares et al., 2006; Toptaş and Hanbay, 2021), which, however, has a fundamental impact on performance.

Conversely, deep learning methods automatically learn an increasingly complex hierarchy of features from input data, bypassing the need for problem–specific knowledge and generally providing better results. Indeed, a deep convolutional neural network (DCNN) for retinal image segmentation has been used in (Liskowski and Krawiec, 2016), while the training examples are subjected to various forms of preprocessing and augmented based on geometric transformations and gamma corrections. A neural network that can be efficiently used in real–time on embedded systems is proposed in (Hajabdollahi et al., 2018). In (Jiang et al., 2018), a fully convolutional network (Long et al., 2015) was described, with an AlexNet (Krizhevsky et al., 2012) encoder. Fully convolutional networks have also been used in (Dasgupta and Singh, 2017; Feng et al., 2017). In (Li et al., 2016), the segmentation task was remolded into a problem of cross–modality data transformation from retinal images to vessel maps. A modified U–Net (Ronneberger et al., 2015) was used in (Yan et al., 2018) to exploit a combination between segment–level loss and pixel–

level loss to deal with the unbalanced ratio between thick and thin vessels in fundus images. A Holistically Nested Edge Detection (HED) network (Xie and Tu, 2015) — originally designed for edge detection — followed by a conditional random field was employed for the retinal blood vessel segmentation in (Fu et al., 2016). Deep supervision was incorporated in some intermediate layers of a VGG network (Liu and Deng, 2015) in (Mo and Zhang, 2017; Maninis et al., 2016). In (Oliveira et al., 2018), a fully convolutional neural network used a stationary wavelet transform pre-processing step to improve the network performance. Finally, in (Sekou et al., 2019), a CNN was pre–trained on image patches and then fine–tuned at the image level.

In this thesis, we use the Segmentation Multiscale Attention Network (SMANet), which allows us to obtain excellent results, comparable with the state–of–the–art.

3.2 Single–stage method

In addition to comparing the results obtained with the state–of–the–art techniques, we also felt it necessary to consider a direct approach. With the Single–Stage (see Figure 3.1) method the label–map and the retinal image are generated simultaneously. Specifically, the retinal images and the label–maps, corresponding to the blood vessels, are stacked in two different channels and placed as input to the PG-GAN.

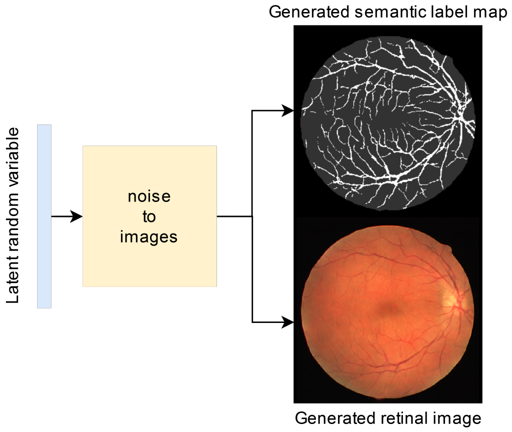


Figure 3.1: Single–Stage image generation scheme.

3.3 Two–stage method

The main goal of this work is to generate realistic retinal images and the corresponding semantic segmentation masks by using a very small number of training samples. The proposed generation procedure is composed of two steps (see Figure 3.2): the first one involves the generation of semantic label–maps of the vessels while, during the second, the synthesis of realistic images based on label–maps is carried out. The quality of the generated images was validated by an expert and their usefulness was verified by the performance obtained on two public benchmark datasets, using the synthesized images to train a segmentation network.

In Section 3.3.1, we give an overview of the approach used to generate the semantic label–maps, while Section 3.3.2 describes the image–to–image translation algorithm that synthesizes retinal images from the semantic label–maps.

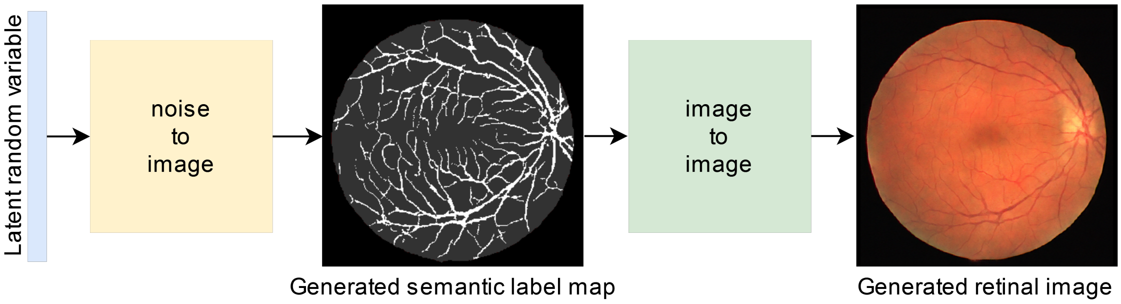


Figure 3.2: The proposed two–step image generation method.

3.3.1 Vessels Generation

The generation of the vessel structure is based on the use of PGGANs, which are capable of learning the distribution of the semantic label–maps. The label–maps are processed to encode both the retinal fundus and the vasculature (i.e., the vessel distribution). To reduce the risk related to the lack of an adequate descriptive power, due to the very limited number of available training samples, data augmentation was applied. Specifically, the semantic label–maps were slightly rotated ($\pm 15°$) and flipped in different ways (horizontal, vertical, and horizontal followed by vertical flips). The generation started at low resolution, and then, the resolution was progressively increased by adding new layers to the networks. The generator and the discriminator were symmetric and grew in sync. The transition from low–resolution image generation to high–resolution image generation followed the procedure described in (Karras et al., 2017), to avoid problems related to sudden transitions. The training started with both the generator and the discriminator having a spatial resolution of 4×4 pixels, progressively increasing until the final resolving power was reached. The Wasserstein loss, with a gradient penalty (Gulrajani et al., 2017), was used as the loss function for the discriminator. The learning procedure is illustrated in Figure 3.3.

It can be observed that the global structure of the vessel distribution was learned at the beginning of the training, whereas finer details were added as the resolution increased. The generation procedure allows us to obtain a virtually infinite number

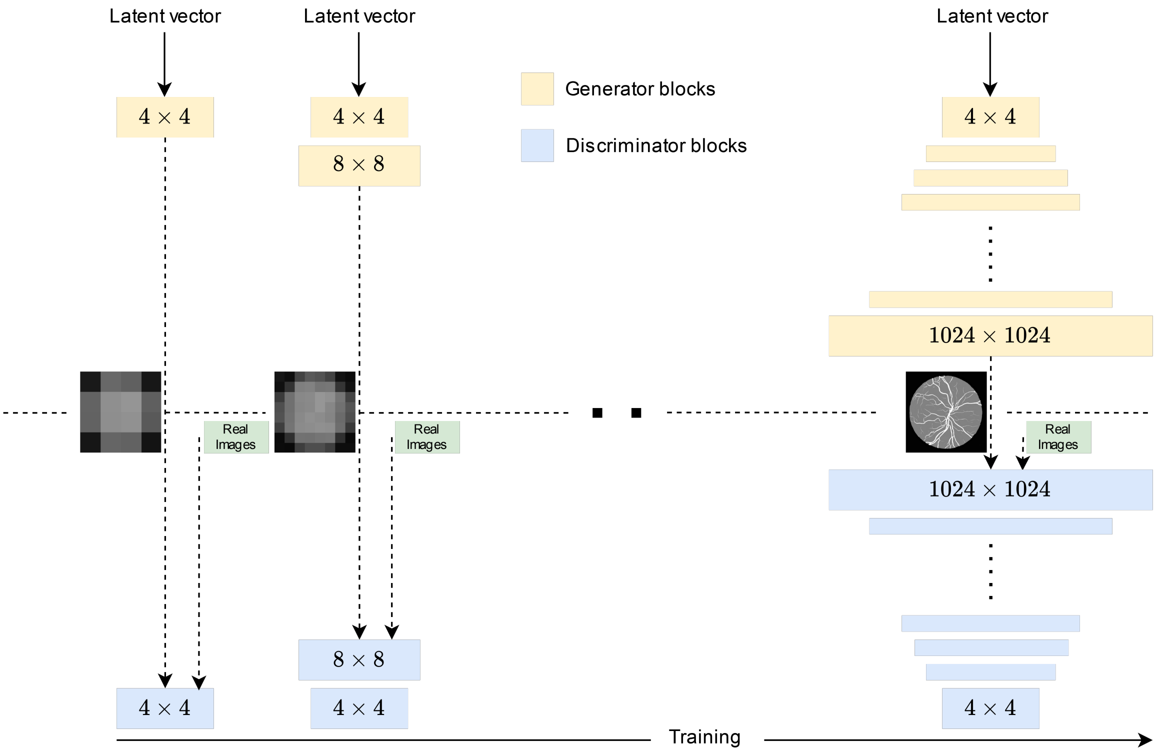


Figure 3.3: Training scheme for the generation of the semantic label–maps. The resolution of the generator (G) and the discriminator (D) was progressively increased until the final resolving power was reached.

of different vasculatures. To reduce the probability of introducing artifacts, a simple post–processing was carried out. Specifically, we improved the circularity of the retinal fundus by applying a morphological opening (Serra, 1983). Small holes were filled, and segments of small dimension were removed from the generated vessel structure.

3.3.2 From Vessels to retinal images

Once the vessel networks were obtained, they were transformed into realistic color retinal images. Our method is based on Pix2PixHD (Wang et al., 2018), a supervised image–to–image translation framework derived from Pix2Pix (Isola et al., 2017). In Pix2Pix, a conditional GAN learns to generate the output conditioned on the corresponding input image. The generator has an encoder–decoder structure, takes in input images belonging to a certain domain A, and generates images in a different domain B. The discriminator observes pairs of images, and the image of A is provided as input along with the corresponding image of B (real or generated). The discriminator aims to distinguish between real and fake (generated) pairs. Pix2PixHD improves upon Pix2Pix by introducing a coarse–to–fine generator composed of two subnetworks that operate at different resolution. A multiscale discriminator was also employed, with an adversarial loss that incorporates a feature–matching loss for training stabilization. In our setup, the semantic label–maps, previously generated, were fed into the generator, which is trained to generate retinal images. An overview of the proposed setup is given in Figure 3.4.

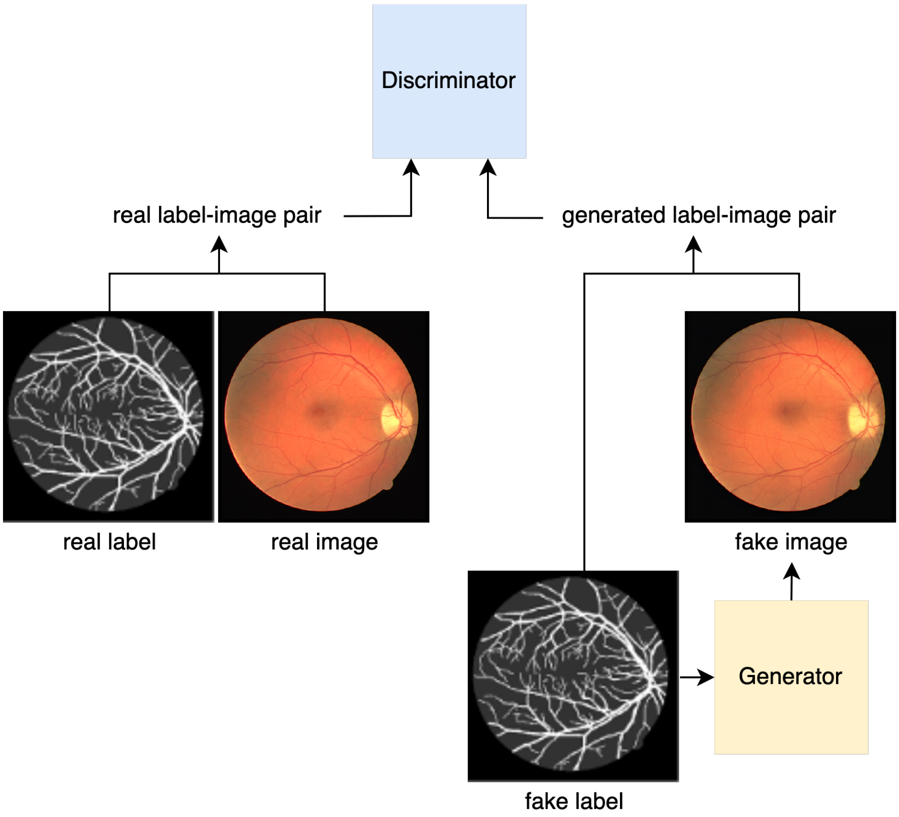


Figure 3.4: Scheme of the Pix2PixHD training framework employed to translate label–maps into retinal images.

3.4 Semantic segmentation network

The semantic segmentation network employed in this thesis is a Segmentation Multiscale Attention Network (SMANet) (Bonechi et al., 2020). The SMANet, originally proposed for scene text segmentation, comprises three main components: a ResNet encoder, a multi–scale attention module, and a convolutional decoder (see Figure 3.5).

The architecture is based on the PSPNet (Zhao et al., 2017), a deep fully convolutional neural network with a ResNet (He et al., 2016) encoder. In the PSPNet, to enlarge the receptive field of the neural network, a set of standard convolutions of the ResNet backbone has been replaced with dilated convolutions (i.e., atrous

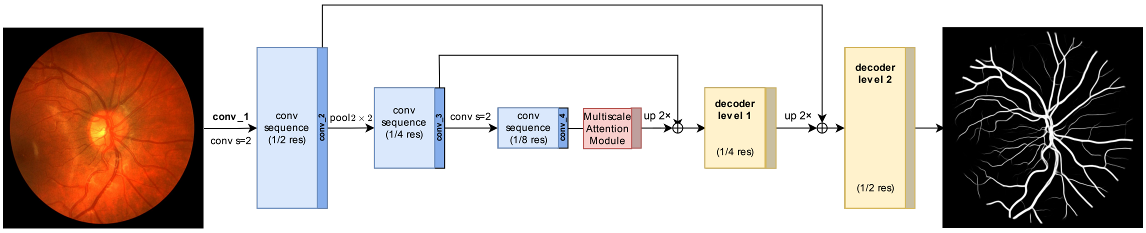


Figure 3.5: Scheme of the SMANet segmentation network.

convolutions (Papandreou et al., 2014)). Moreover, in the PSPNet, a pyramid of pooling layers, with different kernel size, has been employed to gather context information. The pooled feature maps are then up–sampled at the same resolution as the ResNet output, concatenated, and fed into a convolutional layer to obtain an encoded representation. In the original PSPNet, this representation is followed by a final convolutional layer that reduces the feature maps to the number of classes. The desired per–pixel prediction is obtained directly up–sampling to the original image resolution. In the SMANet, a multi–scale attention mechanism is adopted to focus on the relevant objects present in the image, while a two–level convolutional decoder is added to the architecture to better handle the presence of thin objects.

3.5 Training details

The SMANet, used in this work was implemented in TensorFlow. Random crops of 281 × 281 pixels were employed during training, whereas a sliding window of the same size was used for the evaluation. The Adam optimizer (Kingma and Ba, 2014), based on an initial learning rate of 10^{-4} and a mini–batch of 17 examples, was used to train the SMANet. Early stop was employed using a validation set of three images, randomly extracted from the real data training set. Additionally, the PGGAN was realized in TensorFlow, while Pix2PixHD was implemented in PyTorch. The PGGAN architecture is similar to that proposed in (Karras et al., 2017), but to speed up the computation and to reduce overfitting, the maximum number of feature maps for each layer was fixed to 128. Moreover, since the aim of the generator is to produce a semantic label–map, the output image has only one channel, instead of three. The PGGAN and Pix2PixHD hyperparameters were tuned by visually inspecting the quality of the generated samples. The images were resized to the nearest power–of–two resolution (i.e., the retinal images in the DRIVE dataset, which have a resolution of 565 × 584 pixels, were resized to 512 × 512 pixels, whereas the CHASE images that have a resolution of 999 × 960 pixels were resized to 1024 × 1024 pixels).

All experiments were conducted in a Linux environment on a single NVIDIA Tesla V100 SXM2 with 32 GB RAM.

3.6 Experiments and results

3.6.1 The benchmark datasets

- DRIVE dataset — The DRIVE dataset (Staal et al., 2004) includes 40 retinal fundus images of size 584 × 565 × 3 (20 images for training and 20 for test). The images were collected by a screening program for diabetic retinopathy in the Netherlands. Among the 40 photographs, 33 showed no diabetic retinopathy, while 7 showed mild early diabetic retinopathy. The segmentation ground–truth was provided both for the training and the test sets.

- CHASE_DB1 dataset — The CHASE_DB1 dataset (Fraz et al., 2012b) is composed by 28 fundus images of size 960 × 999 × 3, corresponding to the left and right eyes of 14 children. Each image is annotated by two independent human experts. An officially defined split between training and test is not provided for this dataset. In our experiments, we adopted the same strategy as (Li et al., 2016; Yan et al., 2018), selecting the first 20 images for training and the remaining 8 for testing.

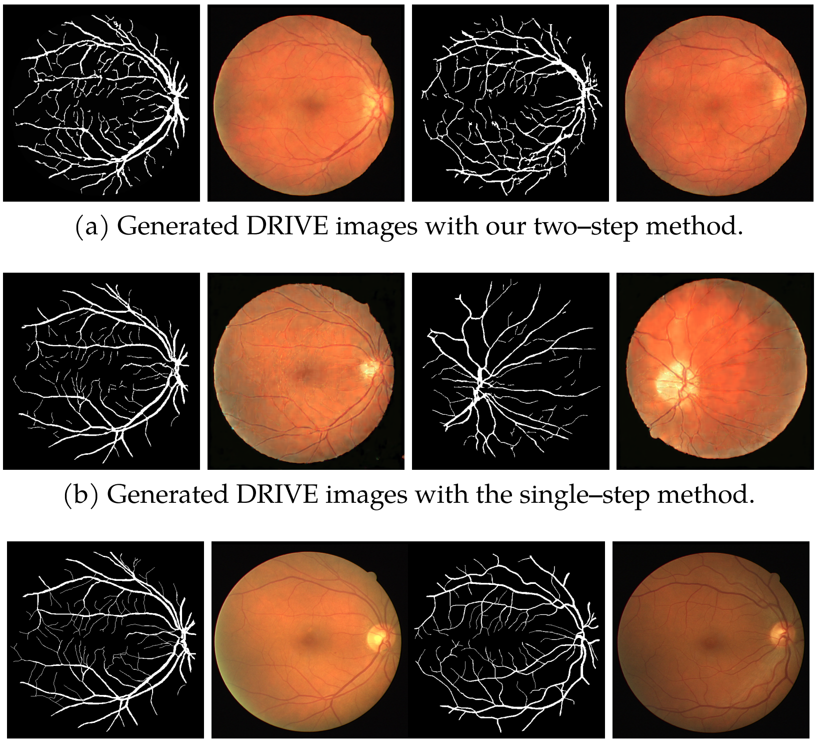
3.6.2 Experimental Results

We provide both qualitative and quantitative evaluations of the generated data. In particular, some qualitative results of the generated retinal images for the DRIVE and CHASE_DB1 datasets are given in Figures 3.6 and 3.7.

In Figure 3.8, a zoom on a random patch of a high–resolution generated image shows that the image–to–image translation allows us to effectively translate the generated vessel structures in retinal images by maintaining the semantic information provided by the semantic label–map. It is worth noting that, although most of the generated samples closely resemble real retinal fundus images, few examples are clearly sub–optimal (see Figure 3.9, which shows disconnected vessels and an unrealistic optical disc).

To further validate the quality of the generation process, a sub–sample of 100 synthetically generated retinal images were examined by an expert ophthalmologist. The evaluation showed that 35% of the images are of medium–high quality. The remaining 65% is visually appealing but contains small details that reveal an unnatural anatomy, such as an optical disc with feathered edges — which actually occur only in the case of specific diseases — or blood vessels that pass too close to the macula — while usually, except in the case of malformations, the macular region is avascular or at least paucivascular.

Table 3.1 compares the characteristics of the proposed method with respect to other learning–based approaches for retinal image generation found in the literature.

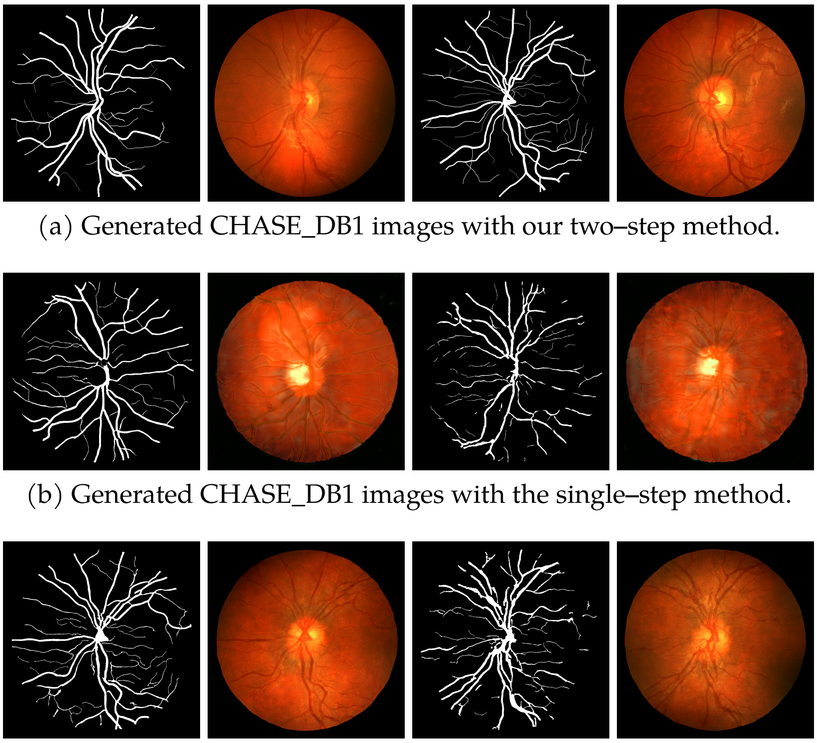


(a) Generated DRIVE images with our two–step method.

(b) Generated DRIVE images with the single–step method.

(c) Real DRIVE images.

Figure 3.6: Examples of real and generated DRIVE images.



(a) Generated CHASE_DB1 images with our two–step method.

(b) Generated CHASE_DB1 images with the single–step method.

(c) Real CHASE_DB1 images.

Figure 3.7: Examples of real and generated CHASE_DB1 images.

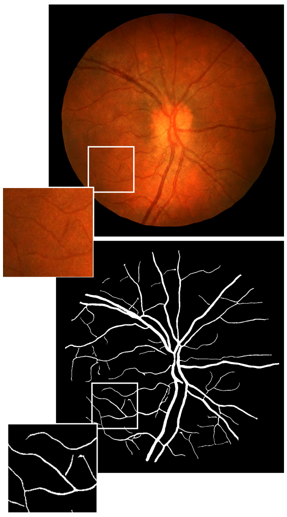


Figure 3.8: Example of a generated image (resolution 1024 × 1024) with the corresponding label–map from the CHASE_DB1 dataset.

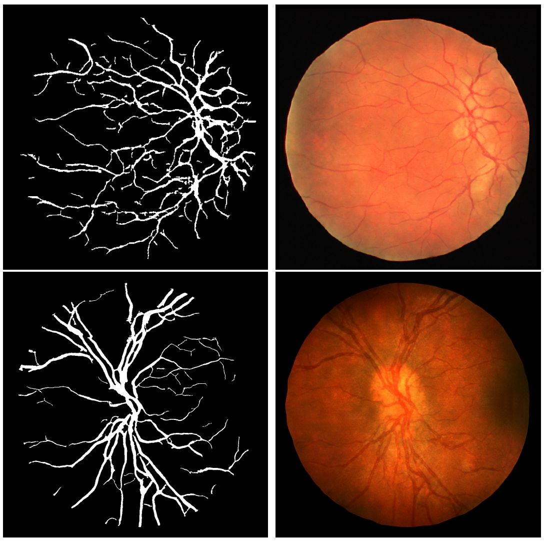


Figure 3.9: Examples of generated images with an unrealistic optical disc and vasculature from DRIVE (**top**) and CHASE _DB1 (**bottom**).

It can be observed that our approach is able to synthesize higher resolution images, with less training samples, with respect to methods that generate both the image and the corresponding segmentation. Moreover, for such methods, the usefulness of the inclusion of synthetic images in semantic segmentation was not assessed. Instead, in this thesis, we demonstrate that synthetic images can be effectively used for data augmentation, which indirectly guarantees the high quality of the generated data.

Indeed, the quantitative analysis consists of assessing the usefulness of the generated images for training a semantic segmentation network. This approach, similar

Methods	**Gen. Vessels**	**Max Res.**	**Samples**
(Costa et al., 2017a)	No	512×512	614
(Zhao et al., 2018)	No	2048×2048	10–20
(Costa et al., 2017b)	Yes	256×256	634
(Beers et al., 2018)	Yes	512×512	5550
Our	Yes	1024×1024	20

Table 3.1: Comparison with other generation approaches.

to (Shmelkov et al., 2018), is based on the assumption that the performance of a deep learning architecture can be directly related with the quality and variety of GAN–generated images. The generation procedure described in Section 3.3 was employed to generate 10,000 synthetic retinal images for both the DRIVE and the CHASE_DB1 datasets; the samples were generated in a single run without any selection strategy.

To evaluate the usefulness of the generated data for semantic segmentation, we employed the following experimental setup:

- SYNTH — the segmentation network was trained using only the 10,000 generated synthetic images;
- REAL — only real data were used to train the semantic segmentation network;
- SYNTH + REAL — synthetic data were used to pre–train the semantic segmentation network and real data were employed for fine–tuning.

Tables 3.2 and 3.3 report the results of the vessel segmentation for the DRIVE and CHASE_DB1 datasets, respectively.

Methods	**AUC**	**Acc**
SYNTH	98.5 %	96.88%
REAL	98.48%	96.87%
SYNTH + REAL	**98.65**%	**96.9**%

Table 3.2: Segmentation performance using the generated and real images from the DRIVE dataset.

It can be observed that the semantic segmentation network, trained on synthetic data, produces results very similar to those obtained by training on real data. This demonstrates that synthetic images effectively capture the training image distribution, so that they can be used to adequately train a deep neural network. Moreover, if fine–tuning with real data is applied after pre–training with synthetic data only, the results further improve with respect to the use of real data only. This fact indicates that the generated data can be effectively used to enlarge small training sets,

Methods	AUC	Acc
SYNTH	98.64%	97.49%
REAL	98.82%	97.5%
SYNTH + REAL	**99.16**%	**97.72**%

Table 3.3: Segmentation performance using the generated and real images from the CHASE_DB1 dataset.

such as DRIVE and CHASE_DB1. Specifically, the AUC is improved by 0.17% and 0.34% on the DRIVE and CHASE_DB1 datasets, respectively.

Another set of experiments was designed to compare the proposed two–stage generation procedure with a traditional single–step approach (described in Section 3.2). In particular, in the single–step method, the label–maps and the retinal images were generated simultaneously. The results of the single–step approach on the DRIVE and CHASE_DB datasets are shown in Tables 3.4 and 3.5.

Methods	AUC	Acc
SYNTH	93.49 %	91.01%
REAL	98.48%	96.87%
SYNTH + REAL	**98.57**%	**96.88**%

Table 3.4: Segmentation performance, using the single–step method, on the DRIVE dataset.

Methods	AUC	Acc
SYNTH	66.96%	92.62%
REAL	98.82%	97.5%
SYNTH + REAL	**98.87**%	**97.65**%

Table 3.5: Segmentation performance, using the single–step method, on the CHASE_DB1 dataset.

Tables 3.6 and 3.7 allows us to quickly visualize the differences between the two methods. It can be observed that better results are obtained in all the setups by employing the two–stage generation approach. In particular, if only synthetic data are used, the AUC increases by 5.01% (31.68%) with the two–stage method in the DRIVE (CHASE_DB1) dataset. As expected, the difference between the two methods is smaller if fine–tuning on real data is applied. Finally, we observe that the gap increases with higher image resolution. In the CHASE_DB1 dataset, in which the images have twice the resolution of the DRIVE dataset, the one–step generated images cannot be effectively used as data augmentation.

Methods	**AUC**	**Acc**
One–Step (S)	93.49 %	91.01%
Two–Step (S)	**98.5**%	**96.88**%
One–Step (S + R)	98.57 %	96.88%
Two–Step (S + R)	**98.65**%	**96.90**%

Table 3.6: A comparison of the vessel segmentation results on the DRIVE dataset between the one–step and the two–step methods.

Methods	**AUC**	**Acc**
One–Step (S)	66.96%	92.62%
Two–Step (S)	**98.64**%	**97.49**%
One–Step (S + R)	98.87%	97.65%
Two–Step (S + R)	**99.16**%	**97.72**%

Table 3.7: A comparison of the vessel segmentation results on the CHASE_DB1 dataset between the one–step and the two–step methods.

Finally, Tables 3.8 and 3.9 compare the proposed approach with other state–of–the–art techniques.

Methods	**AUC**	**Acc**
(Jiang et al., 2018)	96.80%	95.93%
(Li et al., 2016)	97.38%	95.27%
(Dasgupta and Singh, 2017)	97.44%	95.33%
(Yan et al., 2018)	97.52%	95.42%
(Mo and Zhang, 2017)	97.82%	95.21%
(Liskowski and Krawiec, 2016)	97.90%	95.35%
(Feng et al., 2017)	97.92%	95.60%
(Oliveira et al., 2018)	98.21%	95.76%
(Sekou et al., 2019)	**98.74**%	**96.90**%
Our	98.65%	**96.90**%

Table 3.8: A comparison with the state–of–the–art vessel segmentation methods on the DRIVE dataset.

The results show that the proposed approach reaches the state–of–the–art on the DRIVE dataset, where it is only outperfomed by (Sekou et al., 2019), based on the AUC, and outperforms all of the other methods on the CHASE_DB1 dataset. It is worth remembering that the experimental setups adopted by the previous approaches are varied and that a perfect comparison was impossible. For example, CHASE_DB1 does not provide an explicit training/test split, and in (Li et al., 2016;

Methods	**AUC**	**Acc**
(Jiang et al., 2018)	95.80%	95.91%
(Li et al., 2016)	97.16%	95.81%
(Yan et al., 2018)	97.81%	96.10%
(Mo and Zhang, 2017)	98.12%	95.99%
(Liskowski and Krawiec, 2016)	98.45%	95.77%
(Oliveira et al., 2018)	98.55%	96.53%
(Sekou et al., 2019)	98.78%	97.37%
Our	**99.16**%	**97.72**%

Table 3.9: A comparison with the state–of–the–art vessel segmentation on the CHASE_DB1 dataset.

Yan et al., 2018), the same split as in this paper was employed, while in (Sekou et al., 2019; Mo and Zhang, 2017; Oliveira et al., 2018) a fourfold cross–validation strategy was applied (in (Oliveira et al., 2018), where each fold included three images of one eye and four images of the other). Moreover, in (Liskowski and Krawiec, 2016), only patches that were fully inside the field of view were considered. However, even with those inevitable experimental limits, the results of Tables 3.8 and 3.9 suggest that the proposed method is promising and is at least as good as the best state–of–the–art techniques.

Chapter 4

Multi–stage image generation

The natural extension of the two–stage approach presented in Chapter 3 provides for the inclusion of an additional step in the image generation procedure. Indeed, in this chapter, we present a three–stage generation method in which the first additional step defines the positions of the semantic objects included in the image. The rest of the generation procedure is similar to the two–stage approach: first, the semantic labels are generated from the object positions and then the final image is produced. Such a three–stage approach has been applied to a multi–class medical image segmentation task, namely the segmentation Chest X–Ray images, in which the goal is to segment pixels belonging to the lungs and heart.

Chest X–ray (CXR) is one of the most used techniques worldwide for the diagnosis of various diseases, such as pneumonia, tuberculosis, infiltration, heart failure and lung cancer. Chest X–rays have enormous advantages: they are cheap, X–ray equipment is also available in the poorest areas of the world and, moreover, the interpretation/reporting of X–rays is less operator–dependent than the results of other more advanced techniques, such as computed tomography (CT) and magnetic resonance (RMI). Furthermore, undergoing this examination is very fast and minimally invasive (Mettler Jr et al., 2008). Recently, CXR images have gained even greater importance due to COVID–19, which mainly causes lung infection and, after healing, often leaves widespread signs of pulmonary fibrosis: the respiratory tissue affected by the infection loses its characteristics and its normal structure. Consequently, CXR images are often used for the diagnosis of COVID–19 and for the treatment of the after–effects of SARS–CoV–2 (Hussain et al., 2021; Ismael and Şengür, 2021; Nayak et al., 2021).

With the rapid growth in the number of CXRs performed per patient, there is an ever–increasing need for computer–aided diagnosis (CAD) systems to assist radiologists, since manual classification and annotation is time–consuming and subject to errors. Deep Learning (DL) has radically changed the perspective also in medical image processing, and deep neural networks (DNNs) have been applied to a variety of tasks, including organ segmentation, object and lesion classification (Bonechi et al., 2019a), image generation and registration (Van Ginneken et al., 2006). These DL methods constitute an important step towards the construction of CADs for medical images and, in particular, for CXRs.

Semantic segmentation of anatomical structures is the process of classifying each pixel of an image according to the structure to which it belongs. In CAD, segmentation plays a fundamental role. Indeed, segmentation of CXR images is usually necessary to obtain regions of interest and allows the extraction of size measurements of organs (e.g., cardiothoracic ratio quantification) and irregular shapes, which can provide meaningful information on important diseases, such as cardiomegaly, emphysema and lung nodules (Qin et al., 2018). Segmentation may also help to improve the performance of automatic classification: in (Teixeira et al., 2021), it is shown that, by exploiting segmentation, DL models focus their attention primarily on the lung, not taking into account unnecessary background information and noise.

Modern state–of–the–art segmentation algorithms are largely based on DNNs (Long et al., 2015; Chen et al., 2017a; Zhao et al., 2017). However, to achieve good results, DNNs need a fairly large amount of labeled data. Therefore, the main problem with segmentation by DNNs is the scarce availability of appropriate datasets to help solve a given task. This problem is even more evident in the medical field, where data availability is affected by privacy concerns and where a great deal of time and human resources are required to manually label each pixel of each image.

As mentioned above a common solution to cope with this problem is the generation of synthetic images, along with their semantic label–maps. In this thesis, we present a new model, based on GANs, to generate multi–organ segmentation of CXR images. Unlike other approaches, the main feature of the proposed method is that generation occurs in three stages. In the first stage, the position of each anatomical part is generated and represented by a "dot" within the image; in the second stage, semantic labels are obtained from the dots; finally, the chest X–ray image is generated. Each step is implemented by a GAN. More precisely, we adopt Progressively Growing GANs (PGGANs) (Karras et al., 2017), an extension of GANs that permits the generation of high–resolution images, and Pix2PixHD (Wang et al., 2018) for the translation steps. The intuitive idea underlying the approach is that generation benefits by the multi–stage procedure, since the GAN used in each single step faces a subproblem, and can be trained using fewer data. Actually, the generalization capability of neural networks, and more generally of deep learning approaches, has a solid mathematical foundation (see, e.g., the seminal work (Vapnik, 1998) and the more recent papers (Neyshabur et al., 2017; Kawaguchi et al., 2017)). The most general rule states that the simpler the model the better its generalization capability. In our approach, the simplification lies in that, in the three–stage method, the tasks to be solved in each of the three steps are simpler and require less effort.

In order to evaluate the performance of the proposed method, synthetic images were used to train a segmentation network (here, we use the SMANet (Bonechi et al., 2020), described in Chapter 3), subsequently applied to a popular benchmark for multi–organ chest segmentation, the Segmentation in Chest Radiographs (SCR) dataset (Van Ginneken et al., 2006). The results obtained are very promising and exceed, to the best of our knowledge, those obtained by existing methods. Moreover, the quality of the produced segmentation was confirmed by physicians. Finally, to demonstrate the capabilities of our approach, especially having little data available, we compared it to two other methods, using only 10% of the images in the dataset. In particular, the multi–stage approach was compared with a single–stage method — in which chest X–ray images and semantic label–maps are generated simultaneously — and with a two–stage method — where semantic label–maps are generated and then translated into X–ray images. The experimental results show that the proposed three–stage method outperforms the two–stage method, while the two–stage overcomes the single–stage approach, confirming that splitting the generation procedure can be advantageous, particularly when few training images are available.

4.1 Chest X–ray images

4.1.1 Chest X–ray Image Synthesis

Only in a few cases have GANs been used to generate chest radiographic images, as in (Madani et al., 2018), where images for cardiac abnormality classification were obtained with a semi–supervised architecture, or in (Srivastav et al., 2021), where GANs were used to generate low resolution (64×64) CXRs to diagnose pneumonia.

In this thesis, chest X–ray images were generated with the corresponding semantic label–maps, which correspond to different anatomical parts. We then used such images to train a segmentation network, with very promising results.

4.1.2 Organ segmentation

X–rays are one of the most used techniques in medical diagnostics. The reasons are medical and economic, since they are cheap, noninvasive and fast examinations. Many diseases, such as pneumonia, tuberculosis, lung cancer, and heart failure are commonly diagnosed from CXR images. However, due to overlapping organs, low resolution and subtle anatomical shape and size variations, interpreting CXRs accurately remains challenging and requires highly qualified and trained personnel. Therefore, it is of a great clinical and scientific interest to develop computer–based systems that support the analysis of CXRs. In (Candemir et al., 2013), a lung boundary detection system was proposed, building an anatomical atlas to be used in combination with graph cut–based image region refinement (Boykov and Funka-Lea, 2006; Candemir and Akgül, 2011; Boykov and Jolly, 2001). A method for lung field segmentation, based on joint shape and appearance sparse learning, was proposed in (Shao et al., 2014), while a technique for landmark detection was presented in (Ibragimov et al., 2016). Haar–like features and a random forest classifier were combined for the appearance of landmarks. Furthermore, a Gaussian distribution augmented by shape–based random forest classifiers was adopted for learning spatial relationships between landmarks. InvertedNet, an architecture able to segment the heart, clavicles and lungs, was introduced in (Novikov et al., 2018). This network employs a loss function based on the Dice Coefficient, Exponential Linear Units (ELUs) activation functions, and a model architecture that aims at containing the number of parameters. Moreover, the U–Net (Ronneberger et al., 2015) architecture has been widely used for lung segmentation, as in (Wang, 2017; Oliveira and dos Santos, 2018; Islam and Zhang, 2018). In the Structure Correcting Adversarial Network (SCAN) (Dai et al., 2018) a segmentation network and a critic network were jointly trained with an adversarial mechanism for organ segmentation in chest X–rays.

4.2 Chest X–Ray Generation

The main goal of this study is to prove that by dividing the generation problem into multiple simpler stages, the quality of the generated images improves, so that they can be more effectively employed as a form of data augmentation. More specifically, we compare three different generation approaches. The first method, described in Section 4.2.1, consists of generating chest X–ray images and the corresponding label–maps in a single stage. In the second approach, presented in Section 4.2.2, the generation procedure is divided into two stages, where the label–maps are initially generated and then translated into images. The third method, reported in Section 4.2.3, consists of a three–stage approach, that starts by generating the position of the objects in the image, then the label–maps and, finally, the X–ray images. The images generated employing each of the three approaches are comparatively evaluated by training a segmentation network.

To increase the descriptive power of real images, especially with regards to the position of the various organs, standard data augmentation has been applied in advance. Therefore, the original X–ray images, along with their corresponding masks, were augmented by applying random rotations in the interval $[-2, 2]$ degrees, random horizontal, vertical and combined translations from -3% to $+3\%$ of the number of pixels, and adding a Gaussian noise, with zero mean and variance between 0.01 and 0.03×255. For the generation of images, we essentially used two networks well known in the literature, namely PGGANs (Karras et al., 2017) and Pix2PixHD (Wang et al., 2018).

4.2.1 Single–stage method

As in the case of retinas (Section 3.2), our baseline is a single–stage approach in which label–maps and CXR images are generated simultaneously (see Figure 4.1).

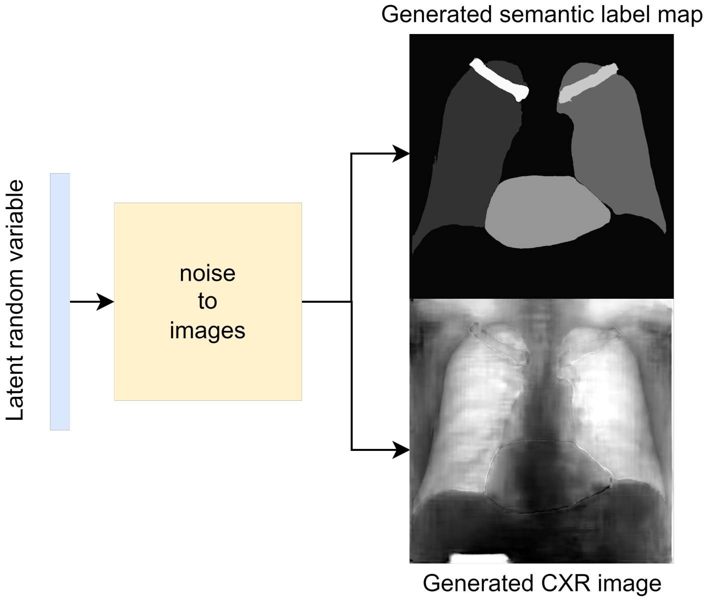


Figure 4.1: Single–stage image generation scheme.

4.2.2 Two–stage method

The two–step procedure used for retinal images (Section 3.3) was exploited for the label–map and CXR image generation (see Figure 4.2).

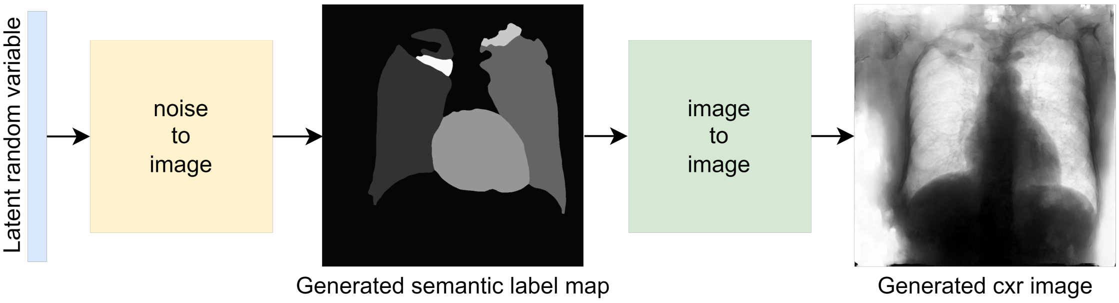


Figure 4.2: Two–stage image generation scheme.

4.2.3 Three–stage method

This section describes how the three–stage approach works. It comprises a further subdivision of the generation procedure, with a first phase consisting of generating the position and type of the objects that will be generated later, regardless of their shape or appearance. This is obtained by generating label–maps that contain "dots" in correspondence with different anatomical parts (lungs, heart, clavicles). The dots can be considered as "seeds", from which, through the subsequent steps, the complete label–maps are realized (second phase). Finally, in the last step, chest X–ray images are generated from the label–maps. The exact procedure is described in the following. Initially, label–maps containing "dots", with a specific value for each anatomical part, are created. The position of the "dot" center is given by the centroid of each labeled anatomical part. The label–maps generated in this phase have a low resolution (64 × 64), as a high level of detail is not necessary, because the exact object shapes are not defined — but only their centroid positions. It should be observed that this also allows a significant reduction in the computational burden of this stage and speeds up the computation. The generated label–maps must be subsequently resized to the original image resolution — required in the following stages of generation (a nearest neighbour interpolation was used to maintain the original label codes) — and translated into labels, which will be finally translated into images, using Pix2PixHD (see Figure 4.3).

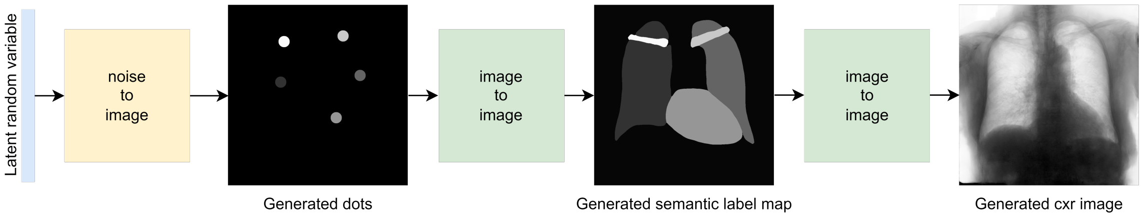


Figure 4.3: Three–stage image generation scheme.

4.3 Training details

The PGGAN architecture, proposed in (Karras et al., 2017), was employed for image generation; the number of parameters were modified to speed up learning and reduce overfitting. More specifically, the maximum number of feature maps for each layer was reduced to 64. Furthermore, since the PGGAN was used to generate seeds and labels, obtaining only the semantic label–maps in both cases, the output image has only one channel instead of three. The generation procedure (PGGAN and Pix2PixHD) was stopped by visually examining the generated samples during the training phase. The images, generated in the various steps for all the methods, have a resolution of 1024 × 1024, except in the case of the "dot" label maps, which, as mentioned before, are generated at a 64 × 64 resolution.

The SMANet is then used for image segmentation. Random crops of 377 × 377 pixels were employed during training, whereas a sliding window of the same size was used for testing. The Adam optimizer (Kingma and Ba, 2014), based on an initial learning rate of 10^{-4} and a mini batch of 17 examples, was used to train the SMANet. All the experiments were carried out in a Linux environment on a single NVIDIA Tesla V100 SXM2 with 32 GB RAM. The SMANet's goal is to produce the semantic segmentation of the lungs and heart. The network is trained by a supervised approach, both in the case of real and synthetic images. In particular, for the images generated by the three different methods, we are able to use this approach thanks to the generation of both the images and the label maps.

4.4 Experiments and results

In this section, after describing the dataset on which our new proposed method was tested, we evaluate the results obtained, both qualitatively — based on the judgment of three physicians — and quantitatively, comparing them with related approaches present in the literature.

4.4.1 Dataset

Chest X–ray images are available thanks to the Japanese Society of Radiological Technology (JSRT) (Shiraishi et al., 2000). The JSRT dataset consists of 247 chest X–ray images. The resolution of the images is 2048 × 2048 pixels, with a spatial resolution of 0.175 mm/pixel and 12 bit gray levels. Furthermore, segmentation supervisions for the JSRT database are available in the Segmentation in the Chest Radiographs (SCR) dataset (Van Ginneken et al., 2006). More precisely, this dataset provides chest X–ray supervisions which correspond with the pixel–level positions of the different anatomical parts. Such supervisions were produced by two observers who segmented five objects in each image: the two lungs, the heart and the two clavicles. The first observer was a medical student and his segmentation was used as the gold standard, while the second observer was a computer science student, specialized in medical imaging, and his segmentation was considered that of a human expert.

The SCR dataset comes with an official splitting, which is employed in this paper and consists of 124 images for learning and 123 for testing. We use two different experimental configurations. In the former, called FULL_DATASET, all the training images are exploited. More precisely, the PGGAN generation network is trained on the basis of 744 images, available in the SCR training set and obtained with the augmentation procedure described above. The SMANet is trained on 7500 synthetic images, generated by the PGGAN, and fine–tuned on the 744 images extracted from the SCR training set, while 2500 synthetic images are used for validation. For the second configuration, called TINY_DATASET, only 10% of the SCR training set is used and the PGGAN is trained on only 66 images (obtained both from SCR and with augmentation); furthermore, the SMANet is trained exactly as above, except for the fine–tuning, which is carried out on 66 images.

Generated images were employed to train a deep semantic segmentation network. The rationale behind the approach is that the performance of the network trained on the generated data reflects the data quality and variety. A good performance of the segmentation network indicates that the generated data successfully capture the true distribution of the real samples. To assess the segmentation results, some standard evaluation metrics were used. The Jaccard Index, J, also called Intersection Over Union (IOU), measures the similarity between two finite sample sets — the predicted segmentation and the target mask in this case — and is defined as the size of their intersection divided by the size of their union. For binary classification, the Jaccard index can be framed in the following formula:

$$J = \frac{TP}{TP + FP + FN}$$

where TP, FP and FN denote the number of true positives, false positives and false negatives, respectively. Furthermore, the Dice Score, DSC, is defined as:

$$DSC = \frac{2 \times TP}{2 \times TP + FP + FN}$$

DSC is a *quotient of similarity between sets* and ranges between 0 and 1.

The experiments can be divided into two phases: first, we evaluated the generation procedure described in Section 4.2.3 using the FULL_DATASET, then, we compared this approach with the other two methods described in Sections 4.2.1 and 4.2.2 using the TINY_DATASET. The purpose of this latter experiment was to evaluate whether multi–stage generation methods are actually more effective in producing data suitable for semantic segmentation with a limited amount of data. In particular, in the experimental setup based on the FULL_DATASET, for the three–stage method, the generation network was trained on all the SCR training images, to which the augmentation procedure described in Section 4.2 was applied. Then, 10,000 synthetic images were generated and used to train the semantic segmentation network. Moreover, we evaluated a fine–tuning of the network on the SCR real images after the pre–training on the generated images. The results, shown in Table 4.1, are compared with those obtained using only real images to train the semantic segmentation network, which can be considered as a baseline.

Next, the TINY_DATASET was used in order to evaluate the performance of the methods with a very small dataset. More precisely, the following experimental setups, the results of which are shown in Table 4.2, are considered:

		Real	**Three–Stage**	
			Synth 3	**Finetune**
J	Left Lung	96.10	95.30	**96.22**
	Heart	90.78	87.25	**91.11**
	Right Lung	**96.85**	96.15	96.79
	Average	94.58	92.90	**94.71**
DSC	Left Lung	98.01	97.6	**98.07**
	Heart	95.17	93.19	**95.35**
	Right Lung	**98.40**	98.04	98.37
	Average	97.19	96.28	**97.26**

Table 4.1: Evaluation of the proposed methods based on the FULL_DATASET, using 2500 generated images for the validation set. Real corresponds to the results obtained using the official training set; *Synth 3* corresponds to the results obtained using only the generated images, while in the *Finetune* column, real data are employed for fine–tuning.

- REAL — only real images are used for training the semantic segmentation network;
- SINGLE–STAGE — the segmentation network uses the images generated by the single–stage method (Synth 1 in the tables) for training while real images are employed for fine–tuning (Finetune in the tables);
- TWO–STAGES — the images generated with the two–stage method are used to pre–train the segmentation network (Synth 2) while real images are used for fine–tuning;
- THREE–STAGE — the images generated with the three–stage method are used for training the segmentation network (Synth 3), while real images are employed for fine–tuning.

In this case, the PGGAN was trained on 66 images, based on 11 images randomly chosen from the entire training set to which the augmentation described above was applied.

In general, we can see that the best results are obtained with the three–stage method followed by fine–tuning. From Table 4.1, we observe a small improvement in results using a fine–tune on a network previously trained with images generated using the three–stage method. Therefore, the three–stage method provides good synthetic data, but the advantage given by generated images is low when the training set is large.

		Real	Single–Stage		Two–Stage		Three–Stage	
			Synth 1	**Finetune**	**Synth 2**	**Finetune**	**Synth 3**	**Finetune**
J	Left Lung	93.70	55.59	74.11	94.91	94.4	94.96	**95.29**
	Heart	85.50	0.07	37.47	86.98	85.21	87.27	**87.47**
	Right Lung	93.70	52.78	79.99	95.90	95.44	95.90	**95.92**
	Average	90.97	36.15	63.86	92.60	91.68	92.71	**92.89**
DSC	Left Lung	96.75	71.46	85.13	97.39	97.12	97.42	**97.59**
	Heart	92.18	0.13	54.51	93.04	92.02	93.20	**93.32**
	Right Lung	96.74	69.09	88.89	97.91	97.66	97.90	**97.92**
	Average	95.22	46.89	76.18	96.11	95.60	96.17	**96.28**

Table 4.2: Evaluation of the proposed methods based on the TINY_DATASET, using 2500 generated images for the validation set. **Real** corresponds to the results obtained using the official training set; *Synth 1*, *Synth 2*, *Synth 3*, correspond to the results obtained using only the generated images, while in the *Finetune* columns, real data are employed for fine–tuning.

Conversely, when few training images are available, in the TINY_DATASET setup, multi–stage methods outperform the baseline (column REAL of Table 4.2) and this happens even without fine–tuning. Thus, in this case, the advantage provided by synthetic images is evident. Moreover, the three–stage method outperforms the two–stage approach, even with fine–tuning, which confirms our claim that splitting the generation procedure may provide a performance increase when few training images are available.

Finally, it is worth noting that fine–tuning improves the performance of the three–stage method, both in the FULL_DATASET and in the TINY_DATASET framework, which does not hold for the two–stage method. This behaviour may be explained by some complementary information that is captured from real images only with the three–stage method. Actually, we may argue that, in different phases of a multi–stage approach, different types of information can be captured: such a diversification seems to provide an advantage to the three–stage method, which develops some capability to model the data domain with more orthogonal information.

4.4.2 Comparison with Other Approaches

Table 4.3 shows our best results and the segmentation performance published by all recent methods, of which we are aware, on the SCR dataset. According to the results in the table, the three–stage method obtained the best performance score both for the lungs and the heart.

However, it is worth mentioning that Table 4.3 gives only a rough idea of the state–of–the–art, since a direct comparison between the proposed method and other approaches is not feasible, our primary focus being on image generation, in contrast with the comparative approaches that are mainly devoted to segmentation, and for which no results are reported on small image datasets. Moreover, the previous methods used different partitions of the SCR dataset to obtain the training and the test set, such as two–fold, three–fold, five–fold cross–validation or ad hoc splittings, which are often not publicly available, while, in our experiments, we preferred to use the original partition, provided with the SCR dataset (note that, compared to most of the other solutions used in comparative methods, the original subdivision has the disadvantage of producing a smaller training set, which is not in conflict, however, with the purpose of the present work). Finally, a variety of different image sizes have also been used, ranging from 256 × 256, to 400 × 400, and to 512 × 512 — the resolution used in this work.

Method	**Image Size**	**Augmentation**	**Evaluation Scheme**	**Lungs**		**Heart**	
				DSC	**J**	**DSC**	**J**
Human	2048 × 2048	No	–	–	94.6	–	87.8
U–Net	256 × 256	No	5–fold CV	–	95.9	–	89.9
InvertedNet	256 × 256	No	3–fold CV	97.4	95	93.7	88.2
SegNet	256 × 256	No	5–fold CV	97.9	95.5	94.4	89.6
FCN	256 × 256	No	5–fold CV	97.4	95	94.2	89.2
SCAN	400 × 400	No	(209/38)	97.3	94.7	92.7	86.6
Our	512 × 512	Yes	official split	**98.2**	**96.5**	**95.36**	**91.1**

Table 4.3: Comparison of segmentation results among different methods on the SCR dataset (CV stands for cross–validation). Human expert (Van Ginneken et al., 2006), U–Net (Wang, 2017), InvertedNet (Novikov et al., 2018), SegNet (Islam and Zhang, 2018), FCN (Islam and Zhang, 2018), SCAN (Novikov et al., 2018). The values in parentheses in the evaluation scheme of the SCAN method correspond to the split between training and testing.

4.4.3 Qualitative Results

In this section, some examples of images and corresponding segmentations, generated with the approaches described in Section 4.2, are qualitatively examined. We also report some comments from three physicians on the generated segmentations, to provide a medical assessment of the quality of our method.

Figures 4.4 and 4.5 display some examples — randomly chosen from all the generated images — of the label–maps and the corresponding chest X–ray images generated with the three methods described in Section 4.2, using the FULL_DATASET and the TINY_DATASET, respectively. We can observe that, with the single and two–stage methods, the images tend to be more similar to those belonging to the training set. For example, in most of the generated images there are white rectangles, which resemble those present in the training images, used to cover the names of both the patient and the hospital. Instead, the three–stage method does not produce such artifacts, suggesting that it is less prone to overfitting.

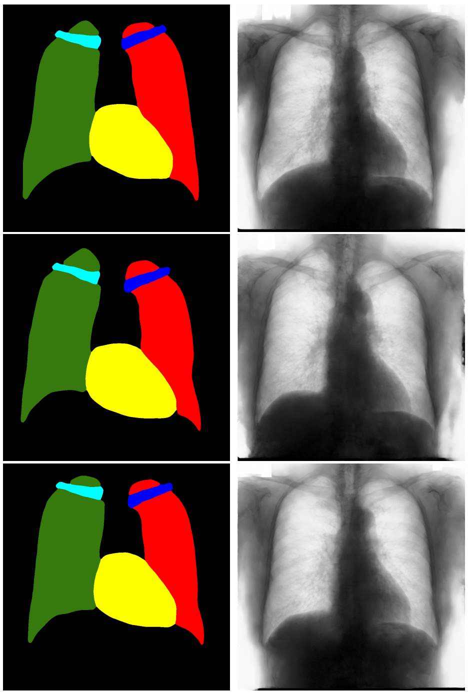
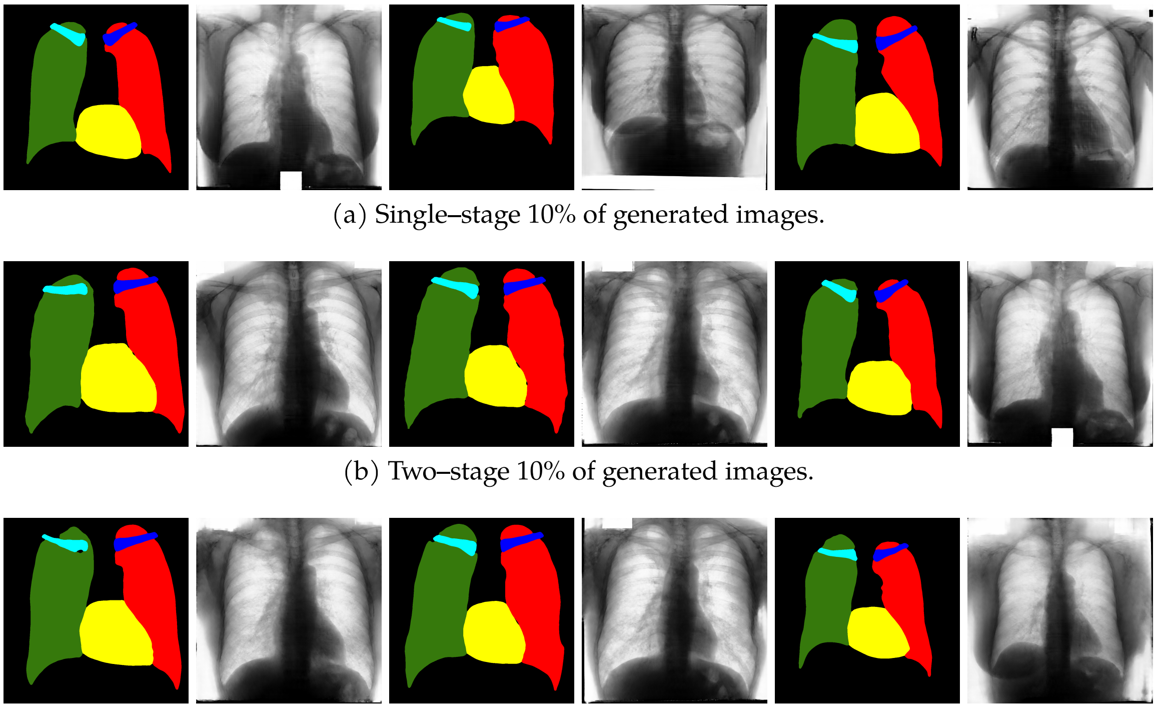


Figure 4.4: Examples of three–stage generated images based on the FULL_DATASET.

Moreover, in order to clarify the limits of the three–stage method, we assessed the quality of the segmentation results based on three human experts, who were asked to check 20 chest X–ray images, along with the corresponding supervision and the segmentation obtained by the SMANet. Such images were chosen among those that can be considered difficult, at least based on the high error obtained by the segmentation algorithm. Figures 4.6 and 4.7 show different examples of the images evaluated by the experts. The first column represents the chest X–ray image, while the second and the third columns, the order of which was randomly exchanged during the presentation to the experts, represent the target segmentation and our prediction, respectively. The three physicians were asked to choose the best segmentation and to comment about their choice. Apart from a general agreement of all the doctors on the good quality of both the target segmentation and the segmentation provided by the three–stage method, surprisingly, they often chose the second one. For the examples in Figure 4.6, for instance, all the experts shared the same opinion, preferring the segmentation obtained by the SMANet over the ground–truth segmentation. To report the results of the qualitative analysis, we numbered the target and predicted segmentation with numbers 1 and 2, respectively, while doctors were assigned unordered pairs to obtain an unbiased result. Then, with respect to Figure 4.6a, the comments reported by the experts were: (1) “In segmentation 1, a fairly



(a) Single–stage 10% of generated images.

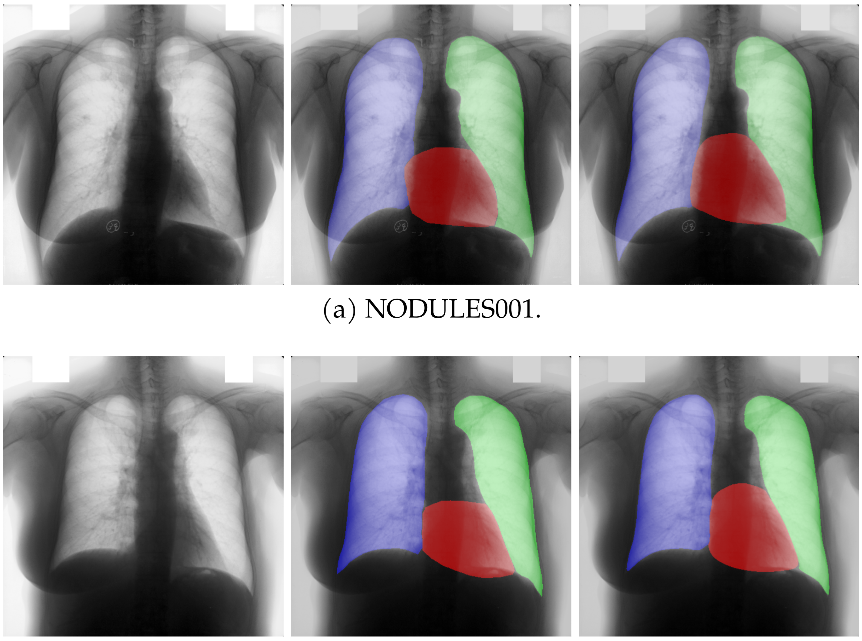
(b) Two–stage 10% of generated images.

(c) Three–stage 10% of generated images.

Figure 4.5: Examples of generated images based on the TINY_DATASET.

large part of the upper left ventricle is missing"; (2) "I choose the segmentation number 2 because the heart profile does not protrude to the left of the spine profile"; (3) "The best is number 2, the other leaves out a piece of the left free edge of the heart, in the cranial area". Furthermore, for Figure 4.6b, we obtained: (1) "The second image is the best for the cardiac profile. For lung profiles, the second image is always better. The only flaw is that it leaks a bit on the right and left costophrenic sinuses". (2) "Image 2 is the best, because the lower cardiac margin is lying down and does not protrude from the diaphragmatic dome. Image number 1 has a too flattened profile of the superior cardiac margin". (3) "Number 2, for the cardiac profile which is more faithful to the real contours".

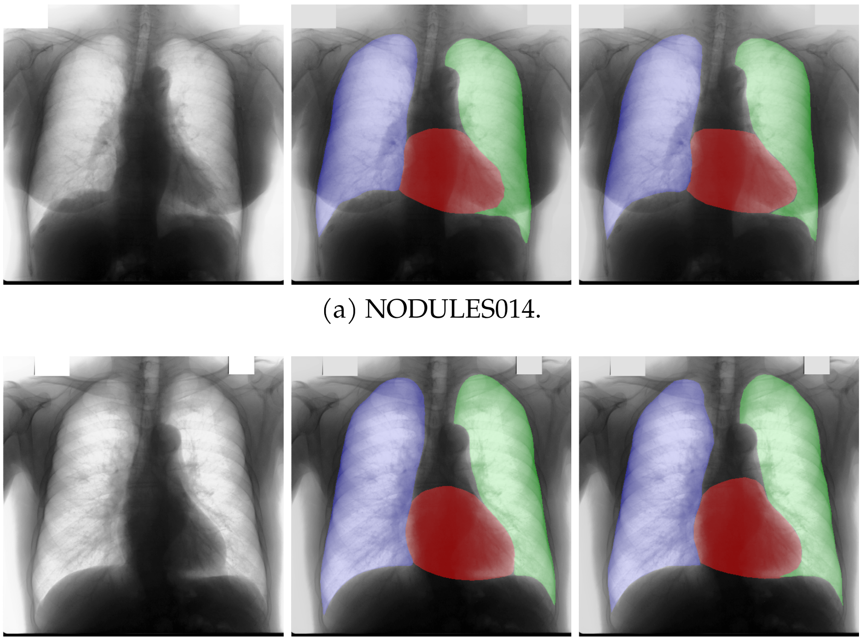
Furthermore, they reported conflicting opinions or decided not to give a preference with respect to the examples in Figure 4.7. When they agreed, they generally found different reasons for choosing one segmentation over the other. With respect to Figure 4.7a, the comments reported by the experts were: (1) I prefer not to indicate any options because the heart image is completely subverted; (2) Segmentation number 2 is better, even if it is complicated to read because there is a "bottle–shaped" heart. The only thing that can be improved in image 2 is that a small portion of the right side of the heart is lost; (3) Number 1 respects more what could be the real



(a) NODULES001.

(b) NODULES066.

Figure 4.6: Examples of segmented images for which doctors shared the same opinion. The first column represents the chest X–ray image, while the second and third columns are the target and our predicted segmentation, respectively.



(a) NODULES014.

(b) NODULES015.

Figure 4.7: Examples of segmented images for which doctors gave conflicting opinions. The first column represents the chest X–ray image, while the second and third columns are the target and our predicted segmentation, respectively.

contours of the heart image. Furthermore, for Figure 4.7b, we obtained: (1) I prefer number 2 because the tip of the heart is well placed on the diaphragm and does not let us see that small wedge–shaped image that incorrectly insinuates itself between heart and diaphragm in image 1 and which has no correspondence in the RX; (2) Both are good segmentations. Both have small problems, for example, in segmentation 1 a small portion of the tip (bottom right of the image) of the heart is missing, in segmentation 2 a part of the outflow cone (the "upper" part of the heart) is missing. It is difficult to choose, probably better number 1 because of the heart; (3) Number 2 because number 1 canal probably exceeds the real dimensions of the cardiac image, including part of the other mediastinal structures.

These different evaluations, albeit limited by the small number of examined images, confirm the difficulty of segmenting CXRs, a difficulty that is likely to be more evident in the case of the images selected for our quality analysis, which were chosen based on the large error produced by the segmentation algorithm.

Chapter 5

Related research topics

This chapter presents some research, conducted during the doctoral period, not included in the main stream of the thesis, but related to its contents. The contributions presented here can be divided into two main categories: theoretical properties and biological applications of Graph Neural Networks (GNNs), and deep learning techniques for image processing. In particular, in Section 5.1, GNNs are studied from a theoretical point of view, trying to understand how transductive learning can be advantageous in a graph processing framework, while Section 5.2 describes the application of GNNs to the problem of predicting protein–protein interfaces. In Section 5.3 and 5.4, instead, DL tecniques are explored for solving problems related to medical image processing and action recognition, respectively.

Finally, in Section 5.5, the project of an application usable by students with visual impairments is described, which was ranked among the top thirty (out of over 250 proposals) at the Rare Disease Hackathon 2020 (organized by Forum Sistema Salute).

5.1 On Inductive–Transductive Learning with Graph Neural Networks

In this section, we describe the work whose results has been published in (Ciano et al., 2022).

A graph G is defined as a pair $G = (V, E)$, where V represents a finite set of *nodes* and $E \subseteq V \times V$ denotes a set of *edges*. Both edges and nodes can be enriched by attributes that are collected into feature vectors. Graphs are powerful and versatile data structures and constitute a natural way of representing information coming, for instance, from social networks, cybersecurity, and computational biology. The main advantage of graphs is that they easily allow to represent entities (nodes) and interactions between them (edges), possibly attaching further information on the nature of the existing relationships. Generally speaking, graph data are widespread in most real–world applications, where existing relations between basic information entities cannot be ignored without affecting the very nature of the problem.

Graph Neural Networks (GNNs) are a class of connectionist models that can process input data encoded as general labeled graphs. This characteristic is shared with other machine learning approaches, such as support vector machines for graphs (Kondor and Lafferty, 2002; Gärtner et al., 2004; Ramon and Gärtner, 2003; Shervashidze et al., 2011; Costa and De Grave, 2010; Orsini et al., 2015), random fields (Lafferty et al., 2001), and Recursive Neural Netowrks (RNNs) (Frasconi et al., 1998; Sperduti and Starita, 1997).

The GNN is a supervised architecture able to face classification and regression tasks, where inputs are encoded as graphs (Scarselli et al., 2008b). The computation is driven by the input graph topology. To each node, a state vector is attached, which is updated as a function of the node label and of the informative contribution of its neighborhood based on an *information diffusion mechanism*. Indeed, GNNs are supposed to capture the local information relevant to the given task — which is stored into the node state — and, finally, thanks to the diffusion process, to collect the whole information attached to the input graph. Afterward, the state is used to compute the node output, f.i. its class or a target property.

More formally, let $x_n(t) \in \mathbb{R}^s$ and $o_n(t) \in \mathbb{R}^m$ be the state and the output of node n at time t, respectively. Then, the computation locally performed at each node during the diffusion process can be described by the following equations

$$x_n(t+1) = \sum_{v:(n,v)\in E} f_w(l_n, l_{(n,v)}, x_v(t), l_v)\,, \tag{5.1}$$

$$o_n(t+1) = g_w(x_n(t), l_n)\,, \tag{5.2}$$

where f_w is the state transition function, which drives the diffusion process, while g_w represents the output function. Moreover, $l_n \in \mathbb{R}^q$, and $l_{(n,v)} \in \mathbb{R}^p$ are the labels attached to n and (n, v), respectively. As previously stated, the computation considers the neighborhood of n, defined by its edges $(n, v) \in E$. In particular, for each node v adjacent to n, the state x_v and the label l_v are used in the state and (indirectly) in the output calculation. The summation in Eq. (5.1) allows us to deal with any number of neighbors without specifying a particular position for each of them.

Usually, both f_w and g_w are implemented by multilayer perceptrons, with a single hidden layer. Eq. (5.1) is replicated on all the nodes of the graph and defines a non–linear dynamic system that describes the unfolding of the *encoding network*. Actually, the encoding network is a recurrent network such that, for each node of the input graph, two modules exist: the f_w module, which is in charge of computing the node state; the g_w module, which calculates the node output. The connectivity in the encoding network, namely how the f_w modules exchange the node states, depends on the graph connectivity.

In order to guarantee the convergence to a steady state in Eqs. (5.1)–(5.2), the original GNN model forced the dynamics of the system to be contractive (Scarselli et al., 2008b). In this case, the Banach Theorem ensures the existence of a unique fixed point and the independence of such point from the initial state. During training, the network weights are adapted to reduce the error between the network outputs and the expected targets on a set of supervised nodes, namely nodes for which the desired output is available. The gradient computation is performed applying the BackPropagation Through Time algorithm on the unfolded encoding network, obtaining the so–called BackPropagation Through Structure (see (Bianchini and Maggini, 2013; Scarselli et al., 2008b) for more details). More recent implementations of GNNs, f.i. (Li et al., 2015; Rossi et al., 2018), relax the state convergence constraint, just computing the output after iterating Eq. (5.1) for a fixed number of steps. This demands more memory requirements for the gradient calculation with respect to (Scarselli et al., 2008b), though it removes the need to constrain parameters to ensure convergence.

Under weak assumptions, the GNN model can approximate in probability all the functions on graphs with any required precision (Scarselli et al., 2008a), showing a generalization capability similar to that of recurrent neural networks (Scarselli et al., 2018). A recent theoretical study on GNN properties can be found in (Xu et al., 2018).

Interestingly, GNNs can naturally exploit both inductive and transductive learning. In the inductive learning framework, a parametric model I_w is learnt by adjusting its weights w based on a training set. Then, the model can be applied to novel test patterns without further accessing the training set. With transductive learning instead, the training set patterns and their targets are used in conjunction with the test patterns. The decision on the test set is taken using a diffusion mechanism, e.g., exploiting the intuition that patterns with similar features are expected to be similar and belong to the same class.

The aim of this work is to study the properties, together with advantages and limitations, of the two learning frameworks applied to GNNs. To this end, we propose a mixed inductive–transductive model that can reproduce the peculiarities of both the frameworks at the same time. This paradigm allows us to use the training patterns both as the source of the transduction, in transductive learning, and to train the network parameters, in inductive learning.

To disentangle the contributions of the inductive and transductive parts of the model, we used an experimental methodology, based on the addition of noise on the node labels and on the repetition of experiments with different quantities of inductive–transductive patterns. The experiments were carried out using the original GNN model (Scarselli et al., 2008b) and the Graph Convolutional Network (GCN) model (Kipf and Welling, 2016), based on synthetic and real datasets. The results revealed interesting properties of the inductive–transductive model and have suggested that it could provide some advantages over the original inductive model. Furthermore, they evidenced interesting examples of conditions when one of the two parts, the inductive or the transductive component, is predominant on the other. These conditions may depend on the data characteristics, e.g., groups of nodes which are "clustered" within the graph, or on the problem peculiarity, e.g., its complexity.

Let us now try to better understand the difference between inductive and transductive learning. In the common inductive learning framework, a model I_w is learned by adjusting its weights, w, based on a set of supervised examples, collected in the training set. The overall learning procedure is aimed at minimizing a suitable loss function that induces the model to capture the statistical distribution of training data. After training, the model I_w can be applied to new patterns, never seen before, completely neglecting the training set, whose related information is collected into the learned parameters w. Conversely, in the transductive framework, learning may not be based on any form of parameter tuning but, instead, both training and test examples can be exploited at the same time, taking advantage of their mutual relationships, such as, for instance, some spatial regularization in the feature space (e.g., manifold regularization). Relationships between data can be exploited either in the learning or in the prediction phase, or in both of them. The prediction on the unsupervised data is obtained by propagating the information available on the neighboring examples, through a "diffusion mechanism" induced by the existing relations. For instance, if n is a test example, then the targets available on its neighboring patterns may be exploited as inputs — together with the local features of n — to compute its output. This approach is particularly useful and natural when only a small set of supervised data, which comes from an unknown stochastic process, is available.

It is worth noting that, in transductive learning, the information useful for processing a particular example is collected by exploring the examples related to it. For this reason, for plain data, the use of pattern relationships is often considered a distinctive feature of transductive learning. Nevertheless, this feature cannot be considered as distinctive when the input domain is constituted by graphs. For relational data, indeed, the difference between the two frameworks must be defined focusing on how the training set targets are used. Thus, we can adopt the following definition: in inductive learning, the targets are used for tuning the parameters, whereas in transductive learning, they are used for the information diffusion.

Modern neural network approaches to graph processing, including GNNs and the derived methods, are naturally prone to be used either for transductive and inductive learning. In order to understand, by simple experiments, advantages, disadvantages and peculiarities of the two learning approaches, and to clarify what happens when they are used in conjuction, we introduce a mixed inductive–transductive approach. In the proposed model, first the dataset is divided into training, validation and test sets. Then, for each of these sets, three disjoint sets of nodes are employed:

- The set of inductive nodes L, whose targets are used to compute the loss function and to adapt the parameters during the inductive network training;
- The set of transductive nodes T, whose targets are used in the transductive learning phase;
- The set of unsupervised nodes U, whose targets are not available.

The union of the sets of inductive and transductive nodes, $S = L \cup T$, constitutes the set of supervised nodes, for which a target is available.

In a pure inductive approach, only the inductive and the unsupervised sets of nodes exist. When graphs to be learnt are fed into the model, the targets of nodes in L are used only to learn the network parameters. The trained model can then be exploited to process both the original graph(s) in the learning set, or to compute the output for the unsupervised nodes in U. In other words, during learning, the model exploits only the graph topology and the information disseminated through the graph. Once the model has been trained, it can be used to generalize to unsupervised patterns; even in this phase, the prediction is based only on the node labels and on the graph topology, without any knowledge of the neighboring node targets.

Instead, in the mixed inductive–transductive learning framework, both the inductive and the transductive set of nodes are taken into account for learning. In particular, the labels of the nodes in T are enriched with their targets, to be explicitly exploited in the diffusion process, yielding a direct transductive contribution. Conversely, for the nodes in L and U, a special *null* target is attached (f.i. a vector of zeros). Formally, for a node n, the enriched label $\bar{l}_n$ will be defined as:

$$\bar{l}_n = \begin{cases} [l_n, t_n, 1] & \text{if } n \in T \\ [l_n, \mathbf{0}, \mathbf{0}] & \text{if } n \in U \cup L, \end{cases} \tag{5.3}$$

where the last scalar value of the label defines whether the node is transductive or not. Moreover, the targets of the inductive nodes in L are used to define the training loss, for tuning the network parameters. Thus, in the mixed framework, the model has to learn to diffuse the information provided by the targets of the transductive nodes, which must be combined with the information coming from the node labels and from the graph topology.

The experiments have been carried out on two synthetic benchmarks, here called *subgraph matching* (SGM) and *distance from the source* (DfS), and on four real–world datasets, namely *Web Spam*, *Cora*, *Citeseer* and *ogb–Arxiv*. In both synthetic benchmarks, data consist of randomly generated graphs. The goal of the subgraph matching problem is that of localizing a given subgraph inside a larger graph, while the objective of the distance from the source problem is that of computing the minimum distance between each node and a given target node, in terms of the number of arcs to traverse to go from one to the other. Instead, the Web Spam benchmark consists of a subset of the Web graph on which Web hosts have to be classified as spam or non–spam. Cora and Citeseer are datasets related to citation networks for scientific publications, widely used for testing graph based algorithms. Finally, the ogb–Arxiv dataset represents the citation network of all Computer Science (CS) Arxiv papers indexed by Microsoft Academic Graph (MAG) (Wang et al., 2020).

Intuitively, in order to study the properties of the proposed model, it is essential to distinguish between the inductive and the transductive contributions to its output. Notice that, actually, the mixed inductive–transductive paradigm aggregates two types of information, provided by the labels of the unsupervised nodes in U and by the targets of the transductive nodes in T, respectively. In the experiments, these two contributions are disentangled by two methods. On the one hand, several trials have been carried out using different percentages of unsupervised and transductive nodes. In this way, we can observe how the performance of the network changes when learning is transformed from pure inductive to increasingly transductive. On the other hand, experiments with an increasing amount of noise, added to the labels of the unsupervised nodes, have also been carried out. In fact, by adding noise to node labels, we corrupt the information available to the inductive algorithm. In the extreme case, when the noise is very large, the information in the labels is lost and the inductive part has no information to use (except for the graph connectivity). Therefore, in this way, we can switch on and off the contribution coming from the inductive part of the GNN.

The results obtained are really interesting, because in addition to demonstrating the effectiveness of the mixed inductive-transductive approach we are able to determine attractive properties for each used dataset. Regarding the subgraph matching problem, the experiments show how the transductive information allows to improve the GNN performance at each noise level, since the best accuracy is always achieved when the number of transductive nodes is maximal. Conversely, adding noise produces a general decrease in performance, for each percentage of transductive nodes, which suggests that a contribution from the inductive information also exists, rapidly deteriorated by noise. In summary, the obtained results suggest that the GNN is able to take into account and combine both types of information. On the contrary, the GCN does not take advantage from the transductive nodes and, in general, the performance of the GCN is lower than that of the GNN in all the cases. Perhaps, this is due to the fact that the GCN model suffers from some limitations in terms of the graphs that it can distinguish (Xu et al., 2018) — a problem that does not affect GNNs. In the case of the SM dataset, such a limitation is particularly important and cannot be alleviated by the transductive information.

The performance achieved by the mixed inductive–transductive framework for the DfS dataset show that inductive learning — for the chosen GNN and GCN configuration — suffers for the long–term dependency problem, since the error signal has to be propagated, on long paths, from the target node to the source node. Such a problem is alleviated in the inductive–transductive model, where the error signal can be propagated also to the transductive nodes. In fact, the results show how the pure inductive models perform poorly, which is also confirmed by the observation that their performance is not influenced by the noise. Finally, the performance of the inductive–transductive model is boosted by the introduction of more and more transductive nodes.

The results achieved on the Web Spam benchmark, using both spam and non–spam hosts as transductive nodes show that the performance increases when more transductive nodes are included, while it decreases in presence of increasingly noisy labels. Interestingly, a well–known peculiarity of the Web is that spam pages tend to refer each other, while it is rare for a non–spam page to have hyperlinks to some spam pages (Castillo et al., 2007). More generally, we expect that spam pages are more clustered than non–spam pages. In order to understand how such a peculiarity can influence the proposed model, two different experiments were performed, in which the set of transductive nodes T has been chosen to collect only spam hosts or only non–spam hosts, respectively. The results confirmed such an asymmetry and show that the proposed model can take advantage of transductive spam hosts, while the advantage provided by the non–spam hosts is low.

Also the results obtained on the datasets Cora and Citeseer confirm that both the models (i.e., GNN and GCN) can take advantage from the presence of transductive nodes. In particular, we can see that most of the times the highest scores for a certain level of noise are reached by the largest number of transductive nodes.

For the ogb–arXiv dataset we used two different approaches to define the inductive and transductive nodes. In the first approach, during testing, all the nodes in both the training and validation sets are used as transductive, whereas during training we use the same procedure as for the other datasets, where the nodes belonging to the training and validation sets are randomly split into transductive and inductive. The second approach is the one used also in the previous experiments, where even the test nodes are randomly split in transductive and inductive. The results also show that GNNs can take advantage of the transductive information in both the approaches.

In conclusion, the proposed analysis allowed to highlight interesting properties of the inductive–transductive model for graphs. These properties, together with the experimental method adopted, may be useful to design new models and algorithms.

A complete description of the proposed method and of the obtained results can be found in (Ciano et al., 2022).

5.2 Graph Neural Networks for the Prediction of Protein–Protein interfaces

In this section, we describe the work that has been published in (Pancino et al., 2020). Proteins are fundamental molecules for life. They are involved in any biological process that takes place in living beings, carrying out a huge variety of different tasks. In these molecules, functionality and structural conformation are strictly correlated (Hegyi and Gerstein, 1999). Therefore, analyzing structural features of proteins is often useful in understanding which biological processes they are involved in, which ligands they bind to and which molecular complexes they form.

The structure of a protein can be described at three different levels: the *primary* structure corresponds to the sequence of amino acids it is composed of; the *secondary* structure corresponds to the local conformation of the peptide chain, in the shape of *α-helices*, *β-sheets* or *coils*; the *tertiary* structure represents the three–dimensional configuration of the molecule. Often, two or more molecules bind together to form a protein complex, whose shape goes under the name of *quaternary* structure. *Dimers* are the simplest protein complexes, as they are composed of just two *monomers*. To form such complexes, *monomers* interact through specialized parts of their surface, called *binding sites* or *interfaces*. These interactions can be studied with the help of graph theory. Indeed, each monomer can be represented as a graph, with nodes corresponding to secondary structure elements (SSEs), while edges stand for spatial relationships between adjacent SSEs, which can be parallel, anti–parallel or mixed. Using graphs of two different monomers, a *correspondence graph* can be built, whose nodes describe all the possible couples of SSEs from the two different subunits (Grindley et al., 1993). Based on the correspondence graph, identifying binding sites on protein surfaces can be reformulated as a maximum clique search problem (Gardiner et al., 1997).

The maximum clique problem is known to be an NP–complete problem, meaning that, except for very small graphs, traditional operations research algorithms (Bomze et al., 1999) will employ a prohibitive amount of time before solving it. From this consideration stemmed the idea of using a machine learning method to solve the problem with reasonable computational costs. In particular, Graph Neural Networks (GNNs) (Scarselli et al., 2008b) look like the perfect model, with their ability to process graph–structured inputs. GNNs have seen many recent advances and have become a leading tool in graph–based applications (Kipf and Welling, 2016; Veličković et al., 2017; Li et al., 2015; Santoro et al., 2017; Battaglia et al., 2018).

The maximum clique problem consists in a binary classification between the nodes which belong to the maximum clique and those which do not. Clique detection was already addressed with GNNs in the seminal work (Gori et al., 2005), and, more recently, also in the transductive learning framework (Rossi et al., 2018).

Finally, this strategy was also further refined by exploiting the deeper version of GNNs, namely Layered Graph Neural Networks (LGNNs) (Bandinelli et al., 2010). In this model, each layer is a standalone GNN which is trained separately, using always the same target. The solution proposed by the previous layer — in the form of node states, outputs or both — is integrated to the input of each layer after the first, significantly addressing the long–term dependency issue.

We developed a binary GNN classifier for the detection of maximum cliques in the correspondence graphs, which addresses the problem as a node–focused classification task — which means that supervision is known on all nodes. The architecture of the MLP module dedicated to the output function was kept fixed, using a single level MLP and the softmax activation function. On the contrary, a 10–fold cross–validation was performed in order to determine the best hyperparameters for the MLP implementing the state transition function. According to the cross–validation results, the MLP architecture with better performance has got a single hidden layer with logistic sigmoid activation functions. This setup was used also to test a 5–layered GNN network, where each GNN layer shares the same architecture. In order to evaluate the performances of the LGNN, a 10–fold cross–validation was carried out again. The LGNN is composed of 5 GNN layers, with state dimension equal to 3. The state is calculated by a 1–layer MLP with logistic sigmoid activations, while the output is calculated with a 1–layer MLP with softmax activation. Since the negative/positive examples ratio is quite large, the weight of positive examples is fixed to the 10% of this ratio, against a weight of 1 for negative examples, in order to balance the learning procedure. The model is trained with the Adam optimizer (Kingma and Ba, 2014) and cross–entropy loss function.

The best performance is obtained with LGNNs integrating only the state in the node labels. There are slight improvements in precision and more tangible improvements in recall, which gains more than 10 percentage points in the second GNN level, and then continues to grow and stabilize in the following levels. This architecture is the only one in which we observe a significant increase of the F1–Score, getting more than 6 percentage points from nearly 35% of the first GNN level to more than 40% in the final GNN level. Contrariwise, integrating in the node labels only the output or both the state and the output, the F1–score decreases through the LGNN layers. The other parameters remain almost stable, except for recall, which slightly increases through the LGNN layers. However, the standard deviation of the recall tends to grow, suffering from a marked dependence on the initial conditions of the experiment. The results confirm the expectations based on biological data and show good performances in determining the interaction sites, recognizing on average about 60% of the interacting nodes.

In conclusion, our method, based on GNNs, can find the maximum clique in an affordable time. The performance of the model was measured in terms of F1–score and show that our approach is very promising, though it can be further improved. One key idea in this direction is that of using graphs in which the nodes correspond to single amino acids, rather than to SSEs. Although this latter approach would increase the complexity of the problem, it would avoid the loss of information we encounter in compressing amino acid features into SSE nodes. Moreover, predictions obtained in this setting would be more accurate, describing the binding site at the amino acid level.

A complete description of the proposed method and of the obtained results can be found in (Pancino et al., 2020).

5.3 Fusion of visual and anamnestic data for the classification of skin lesions with deep learning

In this section, we describe a work whose results has been published in (Bonechi et al., 2019b).

Recently, the results obtained by Deep Learning techniques, and in particular by Convolutional Neural Networks (CNNs), have had a vast impact on the field of image processing (He et al., 2016; Chen et al., 2017b). Many applications have been developed based on CNNs, ranging from automatic analysis reporting (Andreini et al., 2018), to age estimation based on brain NMRs (Rossi et al., 2019) and to skin lesion prognostic classification (Esteva et al., 2017; Yap et al., 2018).

In this study, we propose a new CNN–based tool, capable of classifying skin lesions, which can help dermatologists in the diagnosis of malignant pathologies. Skin cancer is one of the most common tumors in the world and its incidence is increasing worldwide. The main types of skin cancer are non–melanoma skin cancer (NMSC) and malignant melanoma (MM) (Leiter et al., 2014). NMSC includes basal cell carcinoma (BCC) and squamous cell carcinoma (SCC), which usually develop in the epidermis, the outermost layer of the skin. Both tumors, BCC and SCC, tend to occur in over–65 patients, on healthy skin or precancerous skin lesions. In contrast to melanoma, BCC and SCC have a low grade of malignancy and rarely spread to other parts of the body (Apalla et al., 2017). BCC clinically appears as ulcerations, nodules, reddish plaques or scars. Although BCC is locally invasive, it tends to grow slowly, and if diagnosed early and treated appropriately, in almost all cases, it is easily resolved (Paolino et al., 2017). SCC is the second most common skin cancer after BCC. SCC usually starts as a small nodule and grows until it becomes an ulcered lesion. It may present as papules or cutaneous horns. The metastasis incidence of SCC is estimated between 0.5–16% (Apalla et al., 2017). Unlike BCC and SCC, melanoma is an aggressive form of cancer, triggered by an uncontrolled proliferation of melanocytes, pigment–producing cells of neuroectodermal origin.

Cutaneous melanoma is the 20th most common cancer worldwide. It occurs most frequently in adults aged between 40 and 60, while it is rarely observed before puberty (Rastrelli et al., 2014). It is slightly more common in men than in women. Although cutaneous melanoma comprises less than 5% of all skin tumor cases, it causes the majority (75%) of skin cancer deaths.
The worldwide incidence of this pathology has risen sharply over the last decades (Schadendorf and Hauschild, 2014). Globally, 287,723 new cases of cutaneous melanoma have been reported in 2018, and 466,914 new cases are expected to occur until 2040, according to the estimates of Globocan (`https://gco.iarc.fr/tomorrow/en`). Furthermore, the incidence trends vary significantly across different geographic locations and ethnic groups (Matthews et al., 2017). According to data from Globocan, the highest incidence rate worldwide is recorded in Australia and New Zealand, where melanoma is the third most common form of cancer. In Europe the highest incidence rates occur in Norway and Denmark, with 29.6 and 27.6 cases per 100,000 people per year, respectively. Regular clinical screenings and head–to–toe self–examinations are recommended to detect melanoma in its earlier stages, when the lesion is smaller than 2 *mm* and can be easily removed with surgery. If melanoma is diagnosed in a more advanced stage, in which the cancer has already spread to lymph nodes, the excision is insufficient. To treat these cases, surgery must be combined with radiotherapy, immunotherapy or targeted therapy (Domingues et al., 2018). The ABCDE rule is a common screening tool used to distinguish malignant melanoma from a benign mole. The characteristics of a lesion which can help in classifying it as a melanoma include Asymmetry, Border irregularity, Color variegation, a Diameter longer than 6 *mm* and the Evolution of its shape. The development of cutaneous melanoma is a complex phenomenon. It is based on a series of interactions between environmental and endogenous factors, including phototype, number of nevi, presence of atypical nevi, genetic alterations and UV exposure, which is thought to be the major risk factor for this pathology (Gandini et al., 2005). In the diagnosis of melanoma, the dermatologist's expertise is a key element to recognize all the typical elements of a malignant lesion and put them together to set up a correct care path.

Recently, the results obtained by deep learning techniques and, in particular, by Convolutional Neural Networks (CNNs), in the field of image processing, have pushed the use of these methods to develop medical decision support tools. Therefore, our proposal is to implement a CNN–based tool capable of classifying lesion images, which can help dermatologists in diagnosing melanoma. More specifically, this study aims at improving the efficiency in the early detection of skin cancers, developing a classifier capable of integrating the information coming from both dermoscopic images and anamnestic data. Experimental tests were carried out on the freely downloadable International Skin Imaging Collaboration1 (ISIC) Archive

(Codella et al., 2018), showing the importance of the *exogenous* patient data for the correct classification of lesions.

Using the anamnestic data of the patient together with the visual inspection of the skin lesion is the standard procedure in dermatological diagnostics. In fact, it has proven to be fundamental even in the case of the automatic analysis of dermoscopic images with CNNs. Actually, the proposed modular architecture was trained separately with respect to the two types of data — making each module act as an informed feature extractor — whose responses can be properly merged to define the prognosis. The impact of using also clinical data is clearly evidenced by our preliminary experimental results, which show a significant improvement in performance.

A complete description of the proposed method and of the obtained results can be found in (Bonechi et al., 2019a).

5.4 Deep Learning Techniques for Dragonfly Action Recognition

In this section, we propose the work whose results has been published in (Monaci et al., 2020).

Odonata are an order of medium/large hemimetabolous insects, composed of more than 5000 species which differ in color and size. Odonata are morphologically divided into two main infraorders: Zygoptera and Anisoptera. Commonly, Anisoptera are also referred to as “Dragonflies”. They live mainly in freshwater environments, such as ponds, rivers and lakes. They are characterized by a long and thin body, two large multifaceted eyes — made up of thousands of elementary eyes called ommatidia —, two pairs of transparent wings and six legs. They can move the four wings in a fully independent way. This feature, unique in the world of insects, allows them to reach speeds of up to 50 km/h and to obtain formidable performance in flight and hunting, where they can perform backward movements, very narrow turns of death and stops in mid–air. Although the biology of dragonflies has been widely surveyed, there are still very few studies on the kinematic analysis of these insects. In 1975, the Swedish biologist Norberg was the first to study their flight by filming a dragonfly in the open field (Norberg, 1975). He measured parameters such as the width and frequency of the wing flapping, revealing that dragonflies keep their body in an almost horizontal position during flight. A decade later, Azuma et al., using a more advanced video camera, showed that the flaps of the dragonfly’s wings follow a trajectory which can be well represented by a sinusoidal function (Azuma et al., 1985), thus confirming the vortex theory, postulated since 1979. Moreover, by collecting both morphological and kinematic data, they were able to define the first mathematical expression of the wing speed. Since then, numerous experiments have been carried out in this particular research context. New technologies (Wang et al., 2003) were exploited, for instance, to analyze the muscle movements during flight (Faller and Luttges, 1990, 1991), in order to produce prototypes of robotic drones capable of accurately simulating the flight of a dragonfly (Couceiro et al., 2010). These simulations, however, failed to fully reproduce the dexterity, capacity, flexibility and freedom of maneuver of dragonflies (Hu et al., 2009).

This work aims at creating an action recognition model capable of distinguishing the different phases of the dragonfly flight, using deep learning techniques. Given the wide interest of the biological research community in the study of dragonflies, we assume that it is quite useful to develop a reliable system for recognizing dragonfly actions. Indeed, research in different fields could benefit from the recognition system to test hypotheses on dragonfly anatomy, flight dynamics and predatory behaviors. In this project, we propose a dragonfly action recognition system capable of classifying video frames in five classes: take–off, flight, landing, stationary and absent (frames in which the dragonfly is not present). Deep learning requires a huge set of fully annotated data, but, unfortunately, we are not aware of a publicly available labeled dataset of dragonfly images. To train a deep learning architecture, we first collected a suitable number of samples from online videos, which were appropriately preprocessed and labeled frame by frame. Then, different classifier networks for action recognition were compared. First of all, a standard Convolutional Neural Network was tested: this model elaborates one frame at a time, discarding the information of previous frames. To correctly identify the action, the information contained in the previous frame could be fundamental. Therefore, we also trained an LSTM model, which is capable of elaborating frame sequences.

In conclusion, apart from the collection of a large labeled dataset, some guidelines for the calibration, design and implementation of deep models to face this task have been provided. It will be a matter of future research to improve the classification performance of the proposed models, for instance by collecting a larger dataset — in particular providing more frames for the take–off/landing classes — or employing data augmentation techniques in order to extend the available data. A further improvement could be brought by the introduction of more pre–processing operations, compatible with the data type, in order to reduce the disturbing elements in the images and to facilitate the classification task.

A complete description of the proposed method and of the obtained results can be found in (Monaci et al., 2020).

5.5 SlAIde2Voice: A new educational tool for students with visual disabilities

In this section, we propose the work that has been published in (Ciano et al., 2021b). Due to the Covid–19 outbreak, the role played by technology in teaching and learning activities is becoming more and more predominant. School closures has required the implementation of distance learning solutions. All around the world, millions of students experimented for the first time with this new way of attending school. This way of participating lessons involves not only students, but also their families, providing extra difficulties for some types of disabilities, such as visual impairments.

The impact of COVID–19 on students with disabilities could be manifold. In fact such students might suffer due to the lack of accessible software, teaching materials and tutor direct support (WHO, 2020b). The World Health Organization (WHO) has estimated that globally 285 million people of all ages are visually impaired, of which 39 million are blind. Furthermore, almost 18.9 million children below the age of 16 have visual disabilities (WHO, 2020a). Generally, blind or visually impaired students need teaching material presented through other channel, by means of a tactile sign language interpreter to facilitate communication and learning in the school environment.

This work addresses the problem of replacing the visual channel in remote lessons, since vision is fundamental to fully comprehend online presentation, even more considering that a tutor could not be available due to Covid restrictions. According to (Bustamante, 2020), a large portion of students feels Sars–CoV–2 influenced the decision to continue their educational path (11% students decided to not enroll in college and 24% of students claimed they were likely to change their minds about what college to attend) and the majority of college students (63%) think online classes are less effective with respect to traditional in–person lectures. Moreover, parents pointed out the difficulty to reach the teaching staff and to be assisted by it. Finally, it is important to emphasise that distance education courses have been largely used also in pre–pandemic years and may increase in the next years (Bustamante, 2020). It is clear, therefore, that allowing visually impaired students to have the same material of the rest of the class is necessary, especially when dealing with distance learning.

This is not a problem that affects only people at young age, since videoconferencing programs, and remote presentations, are largely used also in work environments. Therefore, the ability of access completely the material provided by the teacher, or by colleagues in a company, could be game–changing for people around the world who are visually impaired, counting both blindness and low vision.

We proposed SlAIde2Voice, a new framework to improve the fruition level of online lectures for visually impaired students. Thanks to precise design choices, our tool is completely independent from the conference platform, making the environment extendable to every person affected by visual disabilities who would like to join an online meeting. Moreover, it allows to reproduce slides in an offline way, something particularly useful for self–study or reasoning after the presentation. Finally, the use of open–source slide making software avoids the problems related to licensed suite, indeed reducing the realization costs to the Braille tablet purchase only. We believe that SlAIde2Voice could fill the gap in learning tools for visually impaired people, especially in conditions in which real lectures and conferences are not possible, as in the case of Covid–19. Our proposed software can increase the self–independence of the user and therefore reduce the need of help from parents or tutors. Moreover, thanks to the acquaintance of self–independence, national money devoted to instantiate special dispositions for visually impaired patients can be saved. Costs, could be even more reduced with the availability of cheaper Braille tablets in the market. We think that our system could be very useful, also when Covid–19 pandemic will be over, for normal lecturing or already existing online university courses (i.e Coursera, Udemy[1]). The approach used to develop the SlAIde2Voice architecture can easily be extended to support learning with other disabilities. For instance, the integration of a simple speech recognition system make the software useful for hearing impaired people.

A complete description of the proposed method can be found in (Ciano et al., 2021b).

[1] `https://www.coursera.org`, `https://www.udemy.com`

Chapter 6

Conclusions and future perspective

In this thesis, we investigated the use of GAN–based deep learning techniques for synthetic image generation along with the corresponding label–maps for segmentation purposes. In the proposed procedure, the generation is split into several steps. The main idea supporting the proposal is that if the difficulty of the problem increases, the generation can benefit from this division into simpler problems. Thus, compared to other generation methods, we can generate the label–maps and images with a simpler network and a smaller number of examples. The generated images can be used to augment the training set of semantic segmentation networks. To demonstrate the effectiveness of the multi–stage method, we applied the method on two important applications in medical image analysis. In the first case, we used a two–step approach for the generation of retinal images. In this case, the segmentation network has to decide whether a pixel belongs to a retinal vessel or to the background. The second application is a multi–class task, aimed at creating CXR images, which is more complex and which is faced based on a three–stage generation procedure.

More detailed conclusions for both these applications are presented below.

Generation and segmentation of retinal images — In Chapter 3, we proposed a two–stage procedure to generate synthetic retinal images. During the first stage, the semantic label masks, which correspond to the retinal vessels, were generated by a Progressively Growing GAN. Then, an image–to–image translation approach was employed to obtain the retinal images from the label masks. The proposed approach allowed us to generate images with unprecedented high resolution and realism. The reported experiments demonstrate the usefulness of synthetic images, which can be effectively used to train a deep segmentation network. Moreover, if fine–tuning based on real images is applied, after a preliminary learning phase based only on synthetic images, the performance of the segmentation network further improves, reaching the performance of or even outperforming the best methods in literature.

Generation and segmentation of Chest X–Ray images — In Chapter 4, we proposed a multi–stage method based on GANs to generate multi–organ segmentation of chest X–ray images. Unlike existing image generation algorithms, in the proposed approach, generation occurs in three stages, starting with “dots”, which represent anatomical parts, and initially involves low–resolution images. After the first step, the resolution is increased to translate “dots” into label–maps. We performed this step with Pix2PixHD, thus making the information grows and obtaining the labels for each considered anatomical part. Finally, Pix2PixHD is also used for translating the label–maps into the corresponding chest X–ray images. The usefulness of our method was experimentally demonstrated, especially when there were few images in the training set, making such a problem affordable thanks to the multi–stage nature of the approach.

Future perspective

The approach proposed in this thesis is general and can be applied to different tasks, not only to the medical field. Indeed, it is future matter of research the experimentation of the multi–stage methods on different application fields.

In fact, both the two–stage and the three–stage method can be applied to other domains, where the the latter approach is likely more suitable for more complex tasks. Another advantage of the three–stage method is its use of "seeds", which can be employed to generate objects in given positions in any type of image. For example, regarding the generation of the Chest X–Ray images, we can generate cancer nodules, in addition to the anatomical parts. The images generated in this way can be used to expand the dataset and to train a segmentation network, the aim of which is to work out whether there is a nodule in a chest X–ray image or not and, possibly, localizing the nodule. Moreover, since dots can be posed in any position inside the image, CXRs with nodules in rare locations can be generated. It is worth mentioning that, currently, we already started experimenting nodule generation for CXR images both based on random and manually located nodules.

Finally, as we said before, the proposed methods are general. Thus, we can apply them to natural images. Currently, we are running experiments on the Cityscapes Dataset (Cordts et al., 2016). The number of images in this dataset is high, but the possibility of moving objects in a natural scene is an interesting and demanding task.

Appendix A

Publications

Journal papers

1. P. Andreini, **Giorgio Ciano**, S. Bonechi, C. Graziani, V. Lachi, A. Mecocci, A. Sodi, F. Scarselli, M. Bianchini, "A Two–Stage GAN for High–Resolution Retinal Image Generation and Segmentation", *Electronics, Special Issue: Image and Video Analysis and Understanding*, vol. 11, num. 1, 2022.

   Candidate's contributions: conceptualization, model and algorithms design, software implementation, experimental setup, original manuscript draft, manuscript reviewing and editing.

2. **Giorgio Ciano**, P. Andreini, T. Mazzierli, M. Bianchini, F. Scarselli, "A Multi–Stage GAN for Multi–Organ Chest X–Ray Image Generation and Segmentation", *Mathematics, Special Issue: Mathematical Modelling and Machine Learning Methods for Bioinformatics and Data Science Applications*, vol. 9, num. 22, 2021.

   Candidate's contributions: conceptualization, model and algorithms design, software implementation, experimental setup, original manuscript draft, manuscript reviewing and editing.

3. **Giorgio Ciano**, A. Rossi, M. Bianchini, F. Scarselli, "On Inductive–Transductive Learning with Graph Neural Networks", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 44, num. 2, pages: 758–769, 2022.

   Candidate's contributions: model and algorithms design, software implementation, experimental setup, original manuscript draft, manuscript reviewing and editing.

Peer reviewed conference papers

1. **Giorgio Ciano**, G.M. Dimitri, A. Rossi, G. Giacomini, S. Bonechi, P. Andreini, E. Messori, "SlAIde2Voice: A new Educational tool for students with visual

disabilities", *CEUR Workshop Proceedings, 1st Workshop on Technology Enhanced Learning Environments for Blended Education*, vol. 2817, 2021.

Candidate's contributions: model design, carried out theoretical analyses, original manuscript draft, manuscript reviewing and editing.

2. N. Pancino, A. Rossi, **Giorgio Ciano**, G. Giacomini, S. Bonechi, P. Andreini, F. Scarselli, M. Bianchini, P. Bongini, "Graph Neural Networks for the Prediction of Protein–Protein Interfaces", *European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning (ESANN)*, pages: 127–132, 2020.

   Candidate's contributions: algorithm design, original manuscript draft, manuscript reviewing and editing.

3. M. Monaci, N. Pancino, P. Andreini, S. Bonechi, P. Bongini, A. Rossi, **Giorgio Ciano**, G. Giacomini, F. Scarselli, M. Bianchini, "Deep Learning Techniques for Dragonfly Action Recognition", *Proceedings of the 9th International Conference on Pattern Recognition Applications and Methods (ICPRAM)*, pages: 562–569, 2020.

   Candidate's contributions: algorithm design, original manuscript draft, manuscript reviewing and editing.

4. S. Bonechi, M. Bianchini, P. Bongini, **Giorgio Ciano**, G. Giacomini, R. Rosai, L. Tognetti, A. Rossi, P. Andreini, "Fusion of visual anamnestic data for the classification of skin lesions with deep learning", *International Conference on Image Analysis and Processing (ICIAP)*, vol. 11808, pages: 211–219, 2019.

   Candidate's contributions: algorithm design, original manuscript draft, manuscript reviewing and editing.

Bibliography

Abràmoff, M. D., Garvin, M. K., and Sonka, M. (2010). Retinal imaging and image analysis. IEEE reviews in biomedical engineering, 3:169–208.

Andreini, P., Bonechi, S., Bianchini, M., Mecocci, A., and Scarselli, F. (2018). A deep learning approach to bacterial colony segmentation. In International Conference on Artificial Neural Networks, pages 522–533. Springer.

Andreini, P., Bonechi, S., Bianchini, M., Mecocci, A., and Scarselli, F. (2020). Image generation by gan and style transfer for agar plate image segmentation. Computer methods and programs in biomedicine, 184:105268.

Andreini, P., Ciano, G., Bonechi, S., Graziani, C., Lachi, V., Mecocci, A., Sodi, A., Scarselli, F., and Bianchini, M. (2022). A two-stage gan for high-resolution retinal image generation and segmentation. Electronics, 11(1).

Apalla, Z., Nashan, D., Weller, R. B., and Castellsagué, X. (2017). Skin cancer: epidemiology, disease burden, pathophysiology, diagnosis, and therapeutic approaches. Dermatology and therapy, 7(1):5–19.

Azuma, A., Azuma, S., Watanabe, I., and Furuta, T. (1985). Flight mechanics of a dragonfly. Journal of experimental biology, 116(1):79–107.

Bandinelli, N., Bianchini, M., and Scarselli, F. (2010). Learning long-term dependencies using layered graph neural networks. In The 2010 International Joint Conference on Neural Networks (IJCNN), pages 1–8. IEEE.

Battaglia, P. W., Hamrick, J. B., Bapst, V., Sanchez-Gonzalez, A., Zambaldi, V., Malinowski, M., Tacchetti, A., Raposo, D., Santoro, A., Faulkner, R., et al. (2018). Relational inductive biases, deep learning, and graph networks. arXiv preprint arXiv:1806.01261.

Beers, A., Brown, J., Chang, K., Campbell, J. P., Ostmo, S., Chiang, M. F., and Kalpathy-Cramer, J. (2018). High-resolution medical image synthesis using progressively grown generative adversarial networks. arXiv preprint arXiv:1805.03144.

Bianchini, M. and Maggini, M. (2013). Supervised neural network models for processing graphs. In Handbook on Neural Information Processing, pages 67–96. Springer.

Bianchini, M. and Scarselli, F. (2014). On the complexity of neural network classifiers: A comparison between shallow and deep architectures. IEEE Transactions on Neural Networks and Learning Systems, 25(8):1553–1565.

Bomze, I. M., Budinich, M., Pardalos, P. M., and Pelillo, M. (1999). The maximum clique problem. In Handbook of combinatorial optimization, pages 1–74. Springer.

Bonechi, S., Bianchini, M., Bongini, P., Ciano, G., Giacomini, G., Rosai, R., Tognetti, L., Rossi, A., and Andreini, P. (2019a). Fusion of visual and anamnestic data for the classification of skin lesions with deep learning. In Cristani, M., Prati, A., Lanz, O., Messelodi, S., and Sebe, N., editors, New Trends in Image Analysis and Processing - ICIAP 2019 - ICIAP International Workshops, BioFor, PatReCH, e-BADLE, DeepRetail, and Industrial Session, Trento, Italy, September 9-10, 2019, Revised Selected Papers, volume 11808 of Lecture Notes in Computer Science, pages 211–219. Springer.

Bonechi, S., Bianchini, M., Bongini, P., Ciano, G., Giacomini, G., Rosai, R., Tognetti, L., Rossi, A., and Andreini, P. (2019b). Fusion of visual and anamnestic data for the classification of skin lesions with deep learning. In International Conference on Image Analysis and Processing, pages 211–219. Springer.

Bonechi, S., Bianchini, M., Scarselli, F., and Andreini, P. (2020). Weak supervision for generating pixel–level annotations in scene text segmentation. Pattern Recognition Letters, 138:1–7.

Boykov, Y. and Funka-Lea, G. (2006). Graph cuts and efficient nd image segmentation. International journal of computer vision, 70(2):109–131.

Boykov, Y. and Jolly, M.-P. (2001). Interactive graph cuts for optimal boundary amp; region segmentation of objects in n-d images. In Proceedings Eighth IEEE International Conference on Computer Vision. ICCV 2001, volume 1, pages 105–112 vol.1.

Bustamante, J. (2020). Online education statistics.

Candemir, S. and Akgül, Y. S. (2011). Statistical significance based graph cut regularization for medical image segmentation. Turkish Journal of Electrical Engineering and Computer Science, 19(6):957–972.

Candemir, S., Jaeger, S., Palaniappan, K., Musco, J. P., Singh, R. K., Xue, Z., Karargyris, A., Antani, S., Thoma, G., and McDonald, C. J. (2013). Lung segmentation in chest radiographs using anatomical atlases with nonrigid registration. IEEE transactions on medical imaging, 33(2):577–590.

Castillo, C., Donato, D., Gionis, A., Murdock, V., and Silvestri, F. (2007). Know your neighbors: Web spam detection using the web topology. In Proceedings of the 30th annual international ACM SIGIR conference on Research and development in information retrieval, pages 423–430.

Chen, L.-C., Papandreou, G., Kokkinos, I., Murphy, K., and Yuille, A. L. (2017a). Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs. IEEE transactions on pattern analysis and machine intelligence, 40(4):834–848.

Chen, L.-C., Papandreou, G., Schroff, F., and Adam, H. (2017b). Rethinking atrous convolution for semantic image segmentation. arXiv preprint arXiv:1706.05587.

Chen, Q. and Koltun, V. (2017). Photographic image synthesis with cascaded refinement networks. In Proceedings of the IEEE international conference on computer vision, pages 1511–1520.

Ciano, G., Andreini, P., Mazzierli, T., Bianchini, M., and Scarselli, F. (2021a). A multi-stage gan for multi-organ chest x-ray image generation and segmentation. Mathematics, 9(22).

Ciano, G., Dimitri, G. M., Rossi, A., Giacomini, G., Bonechi, S., Andreini, P., and Messori, E. (2021b). Slaide2voice: A new educational tool for students with visual disabilities. In CEUR Workshop Proc.

Ciano, G., Rossi, A., Bianchini, M., and Scarselli, F. (2022). On inductive–transductive learning with graph neural networks. IEEE Transactions on Pattern Analysis and Machine Intelligence, 44(2):758–769.

Codella, N. C., Gutman, D., Celebi, M. E., Helba, B., Marchetti, M. A., Dusza, S. W., Kalloo, A., Liopyris, K., Mishra, N., Kittler, H., et al. (2018). Skin lesion analysis toward melanoma detection: A challenge at the 2017 international symposium on biomedical imaging (isbi), hosted by the international skin imaging collaboration (isic). In 2018 IEEE 15th international symposium on biomedical imaging (ISBI 2018), pages 168–172. IEEE.

Collins, D., Zijdenbos, A., Kollokian, V., Sled, J., Kabani, N., Holmes, C., and Evans, A. (1998). Design and construction of a realistic digital brain phantom. IEEE Transactions on Medical Imaging, 17(3):463–468.

Cordts, M., Omran, M., Ramos, S., Rehfeld, T., Enzweiler, M., Benenson, R., Franke, U., Roth, S., and Schiele, B. (2016). The cityscapes dataset for semantic urban scene understanding. In Proceedings of the IEEE conference on computer vision and pattern recognition, pages 3213–3223.

Costa, F. and De Grave, K. (2010). Fast neighborhood subgraph pairwise distance kernel. In ICML.

Costa, P., Galdran, A., Meyer, M. I., Abramoff, M. D., Niemeijer, M., Mendonça, A. M., and Campilho, A. (2017a). Towards adversarial retinal image synthesis. arXiv preprint arXiv:1701.08974.

Costa, P., Galdran, A., Meyer, M. I., Niemeijer, M., Abràmoff, M., Mendonça, A. M., and Campilho, A. (2017b). End-to-end adversarial retinal image synthesis. IEEE transactions on medical imaging, 37(3):781–791.

Couceiro, M. S., Ferreira, N., and Tenreiro Machado, J. (2010). Modeling and control of a dragonfly-like robot. Journal of Control Science and Engineering, 2010.

Cybenko, G. (1989). Approximation by superpositions of a sigmoidal function. Mathematics of control, signals and systems, 2(4):303–314.

Dai, W., Dong, N., Wang, Z., Liang, X., Zhang, H., and Xing, E. P. (2018). Scan: Structure correcting adversarial network for organ segmentation in chest x-rays. In Deep learning in medical image analysis and multimodal learning for clinical decision support, pages 263–273. Springer.

Dasgupta, A. and Singh, S. (2017). A fully convolutional neural network based structured prediction approach towards the retinal vessel segmentation. In 2017 IEEE 14th International Symposium on Biomedical Imaging (ISBI 2017), pages 248–251.

Domingues, B., Lopes, J. M., Soares, P., and Pópulo, H. (2018). Melanoma treatment in review. ImmunoTargets and therapy, 7:35.

Esteva, A., Kuprel, B., Novoa, R. A., Ko, J., Swetter, S. M., Blau, H. M., and Thrun, S. (2017). Dermatologist-level classification of skin cancer with deep neural networks. nature, 542(7639):115–118.

Everingham, M., Eslami, S. A., Van Gool, L., Williams, C. K., Winn, J., and Zisserman, A. (2015). The pascal visual object classes challenge: A retrospective. International journal of computer vision, 111(1):98–136.

Faller, W. and Luttges, M. (1990). Flight control in the dragonfly: a neurobiological simulation. Advances in neural information processing systems, 3:514–520.

Faller, W. E. and Luttges, M. W. (1991). Recording of simultaneous single-unit activity in the dragonfly ganglia. Journal of neuroscience methods, 37(1):55–69.

Feng, Z., Yang, J., and Yao, L. (2017). Patch-based fully convolutional neural network with skip connections for retinal blood vessel segmentation. In 2017 IEEE International Conference on Image Processing (ICIP), pages 1742–1746.

Fiorini, S., Ballerini, L., Trucco, E., and Ruggeri, A. (2014). Automatic generation of synthetic retinal fundus images. In MIUA, pages 7–12.

Frasconi, P., Gori, M., and Sperduti, A. (1998). A general framework for adaptive processing of data structures. IEEE transactions on Neural Networks, 9(5):768–786.

Fraz, M. M., Remagnino, P., Hoppe, A., Uyyanonvara, B., Rudnicka, A. R., Owen, C. G., and Barman, S. A. (2012a). Blood vessel segmentation methodologies in retinal images–a survey. Computer methods and programs in biomedicine, 108(1):407–433.

Fraz, M. M., Remagnino, P., Hoppe, A., Uyyanonvara, B., Rudnicka, A. R., Owen, C. G., and Barman, S. A. (2012b). An ensemble classification-based approach applied to retinal blood vessel segmentation. IEEE Transactions on Biomedical Engineering, 59(9):2538–2548.

Frid-Adar, M., Diamant, I., Klang, E., Amitai, M., Goldberger, J., and Greenspan, H. (2018). Gan-based synthetic medical image augmentation for increased cnn performance in liver lesion classification. Neurocomputing, 321:321–331.

Fu, H., Xu, Y., Wong, D. W. K., and Liu, J. (2016). Retinal vessel segmentation via deep learning network and fully-connected conditional random fields. In 2016 IEEE 13th International Symposium on Biomedical Imaging (ISBI), pages 698–701.

Gandini, S., Sera, F., Cattaruzza, M. S., Pasquini, P., Picconi, O., Boyle, P., and Melchi, C. F. (2005). Meta-analysis of risk factors for cutaneous melanoma: Ii. sun exposure. European journal of cancer, 41(1):45–60.

Gardiner, E. J., Artymiuk, P. J., and Willett, P. (1997). Clique-detection algorithms for matching three-dimensional molecular structures. Journal of Molecular Graphics and Modelling, 15(4):245–253.

Gärtner, T., Lloyd, J. W., and Flach, P. A. (2004). Kernels and distances for structured data. Machine Learning, 57(3):205–232.

Gatys, L. A., Ecker, A. S., and Bethge, M. (2015). A neural algorithm of artistic style. arXiv preprint arXiv:1508.06576.

Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., and Bengio, Y. (2014). Generative adversarial nets. Advances in neural information processing systems, 27.

Gori, M., Monfardini, G., and Scarselli, F. (2005). A new model for learning in graph domains. In Proceedings. 2005 IEEE International Joint Conference on Neural Networks, 2005., volume 2, pages 729–734. IEEE.

Grindley, H. M., Artymiuk, P. J., Rice, D. W., and Willett, P. (1993). Identification of tertiary structure resemblance in proteins using a maximal common subgraph isomorphism algorithm. Journal of molecular biology, 229(3):707–721.

Gulrajani, I., Ahmed, F., Arjovsky, M., Dumoulin, V., and Courville, A. (2017). Improved training of wasserstein gans. In Proceedings of the 31st International Conference on Neural Information Processing Systems, NIPS'17, page 5769–5779, Red Hook, NY, USA. Curran Associates Inc.

Hajabdollahi, M., Esfandiarpoor, R., Najarian, K., Karimi, N., Samavi, S., and Reza-Soroushmeh, S. (2018). Low complexity convolutional neural network for vessel segmentation in portable retinal diagnostic devices. In 2018 25th IEEE International Conference on Image Processing (ICIP), pages 2785–2789. IEEE.

He, K., Zhang, X., Ren, S., and Sun, J. (2016). Deep residual learning for image recognition. In Proceedings of the IEEE conference on computer vision and pattern recognition, pages 770–778.

Hegyi, H. and Gerstein, M. (1999). The relationship between protein structure and function: a comprehensive survey with application to the yeast genome. Journal of molecular biology, 288(1):147–164.

Hodaň, T., Vineet, V., Gal, R., Shalev, E., Hanzelka, J., Connell, T., Urbina, P., Sinha, S. N., and Guenter, B. (2019). Photorealistic image synthesis for object instance detection. In 2019 IEEE International Conference on Image Processing (ICIP), pages 66–70. IEEE.

Hoover, A., Kouznetsova, V., and Goldbaum, M. (2000). Locating blood vessels in retinal images by piecewise threshold probing of a matched filter response. IEEE Transactions on Medical Imaging, 19(3):203–210.

Hu, B., Tang, Y., Eric, I., Chang, C., Fan, Y., Lai, M., and Xu, Y. (2018). Unsupervised learning for cell-level visual representation in histopathology images with generative adversarial networks. IEEE journal of biomedical and health informatics, 23(3):1316–1328.

Hu, Z., McCauley, R., Schaeffer, S., and Deng, X. (2009). Aerodynamics of dragonfly flight and robotic design. In 2009 IEEE international conference on robotics and automation, pages 3061–3066. IEEE.

Hussain, E., Hasan, M., Rahman, M. A., Lee, I., Tamanna, T., and Parvez, M. Z. (2021). Corodet: A deep learning based classification for covid-19 detection using chest x-ray images. Chaos, Solitons & Fractals, 142:110495.

Ibragimov, B., Likar, B., Pernuš, F., and Vrtovec, T. (2016). Accurate landmark-based segmentation by incorporating landmark misdetections. In 2016 IEEE 13th International Symposium on Biomedical Imaging (ISBI), pages 1072–1075.

Islam, J. and Zhang, Y. (2018). Towards robust lung segmentation in chest radiographs with deep learning. arXiv preprint arXiv:1811.12638.

Ismael, A. M. and Şengür, A. (2021). Deep learning approaches for covid-19 detection based on chest x-ray images. Expert Systems with Applications, 164:114054.

Isola, P., Zhu, J.-Y., Zhou, T., and Efros, A. A. (2017). Image-to-image translation with conditional adversarial networks. In Proceedings of the IEEE conference on computer vision and pattern recognition, pages 1125–1134.

Jiang, Z., Zhang, H., Wang, Y., and Ko, S.-B. (2018). Retinal blood vessel segmentation using fully convolutional network with transfer learning. Computerized Medical Imaging and Graphics, 68:1–15.

Kanski, J. J. and Bowling, B. (2015). Kanski's clinical ophthalmology e-book: a systematic approach. Elsevier Health Sciences.

Karras, T., Aila, T., Laine, S., and Lehtinen, J. (2017). Progressive growing of gans for improved quality, stability, and variation. arXiv preprint arXiv:1710.10196.

Karras, T., Laine, S., and Aila, T. (2019). A style-based generator architecture for generative adversarial networks. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pages 4401–4410.

Kawaguchi, K., Kaelbling, L. P., and Bengio, Y. (2017). Generalization in deep learning. arXiv preprint arXiv:1710.05468.

Khan, K. B., Siddique, M. S., Ahmad, M., and Mazzara, M. (2020). A hybrid unsupervised approach for retinal vessel segmentation. BioMed Research International, 2020.

Khawaja, A., Khan, T. M., Khan, M. A., and Nawaz, S. J. (2019). A multi-scale directional line detector for retinal vessel segmentation. Sensors, 19(22):4949.

Kingma, D. P. and Ba, J. (2014). Adam: A method for stochastic optimization. arXiv preprint arXiv:1412.6980.

Kipf, T. N. and Welling, M. (2016). Semi-supervised classification with graph convolutional networks. arXiv preprint arXiv:1609.02907.

Kondor, R. I. and Lafferty, J. (2002). Diffusion kernels on graphs and other discrete structures. In Proceedings of the 19th international conference on machine learning, volume 2002, pages 315–322.

Krizhevsky, A., Sutskever, I., and Hinton, G. E. (2012). Imagenet classification with deep convolutional neural networks. Advances in neural information processing systems, 25:1097–1105.

Kugelman, J., Alonso-Caneiro, D., Read, S. A., Vincent, S. J., Chen, F. K., and Collins, M. J. (2021). Data augmentation for patch-based oct chorio-retinal segmentation using generative adversarial networks. Neural Computing and Applications, pages 1–16.

Lafferty, J., McCallum, A., and Pereira, F. C. (2001). Conditional random fields: Probabilistic models for segmenting and labeling sequence data.

Ledig, C., Theis, L., Huszár, F., Caballero, J., Cunningham, A., Acosta, A., Aitken, A., Tejani, A., Totz, J., Wang, Z., et al. (2017). Photo-realistic single image super-resolution using a generative adversarial network. In Proceedings of the IEEE conference on computer vision and pattern recognition, pages 4681–4690.

Leiter, U., Eigentler, T., and Garbe, C. (2014). Epidemiology of skin cancer. Sunlight, vitamin D and skin cancer, pages 120–140.

Li, Q., Feng, B., Xie, L., Liang, P., Zhang, H., and Wang, T. (2016). A cross-modality learning approach for vessel segmentation in retinal images. IEEE Transactions on Medical Imaging, 35(1):109–118.

Li, Y., Tarlow, D., Brockschmidt, M., and Zemel, R. (2015). Gated graph sequence neural networks. arXiv preprint arXiv:1511.05493.

Lin, T.-Y., Maire, M., Belongie, S., Hays, J., Perona, P., Ramanan, D., Dollár, P., and Zitnick, C. L. (2014). Microsoft coco: Common objects in context. In European conference on computer vision, pages 740–755. Springer.

Liskowski, P. and Krawiec, K. (2016). Segmenting retinal blood vessels with deep neural networks. IEEE transactions on medical imaging, 35(11):2369–2380.

Liu, B., Gu, L., and Lu, F. (2019). Unsupervised ensemble strategy for retinal vessel segmentation. In International Conference on Medical Image Computing and Computer-Assisted Intervention, pages 111–119. Springer.

Liu, I. and Sun, Y. (1993). Recursive tracking of vascular networks in angiograms based on the detection-deletion scheme. IEEE Transactions on medical imaging, 12(2):334–341.

Liu, M.-Y., Breuel, T., and Kautz, J. (2017). Unsupervised image-to-image translation networks. In Advances in neural information processing systems, pages 700–708.

Liu, M.-Y. and Tuzel, O. (2016). Coupled generative adversarial networks. Advances in neural information processing systems, 29:469–477.

Liu, S. and Deng, W. (2015). Very deep convolutional neural network based image classification using small training sample size. In 2015 3rd IAPR Asian Conference on Pattern Recognition (ACPR), pages 730–734.

Long, J., Shelhamer, E., and Darrell, T. (2015). Fully convolutional networks for semantic segmentation. In Proceedings of the IEEE conference on computer vision and pattern recognition, pages 3431–3440.

Madani, A., Moradi, M., Karargyris, A., and Syeda-Mahmood, T. (2018). Semi-supervised learning with generative adversarial networks for chest x-ray classification with ability of data domain adaptation. In 2018 IEEE 15th International symposium on biomedical imaging (ISBI 2018), pages 1038–1042. IEEE.

Maninis, K.-K., Pont-Tuset, J., Arbeláez, P., and Van Gool, L. (2016). Deep retinal image understanding. In Ourselin, S., Joskowicz, L., Sabuncu, M. R., Unal, G., and Wells, W., editors, Medical Image Computing and Computer-Assisted Intervention – MICCAI 2016, pages 140–148, Cham. Springer International Publishing.

Matthews, N. H., Li, W.-Q., Qureshi, A. A., Weinstock, M. A., and Cho, E. (2017). Epidemiology of melanoma. Exon Publications, pages 3–22.

McCulloch, W. S. and Pitts, W. (1943). A logical calculus of the ideas immanent in nervous activity. The bulletin of mathematical biophysics, 5(4):115–133.

Menti, E., Bonaldi, L., Ballerini, L., Ruggeri, A., and Trucco, E. (2016). Automatic generation of synthetic retinal fundus images: Vascular network. In International Workshop on Simulation and Synthesis in Medical Imaging, pages 167–176. Springer.

Mettler Jr, F. A., Huda, W., Yoshizumi, T. T., and Mahesh, M. (2008). Effective doses in radiology and diagnostic nuclear medicine: a catalog. Radiology, 248(1):254–263.

Mo, J. and Zhang, L. (2017). Multi-level deep supervised networks for retinal vessel segmentation. International journal of computer assisted radiology and surgery, 12(12):2181–2193.

Monaci, M., Pancino, N., Andreini, P., Bonechi, S., Bongini, P., Rossi, A., Ciano, G., Giacomini, G., Scarselli, F., and Bianchini, M. (2020). Deep learning techniques for dragonfly action recognition. In ICPRAM, pages 562–569.

Nash Jr, J. F. (1950). Equilibrium points in n-person games. Proceedings of the national academy of sciences, 36(1):48–49.

Nayak, S. R., Nayak, D. R., Sinha, U., Arora, V., and Pachori, R. B. (2021). Application of deep learning techniques for detection of covid-19 cases using chest x-ray images: A comprehensive study. Biomedical Signal Processing and Control, 64:102365.

Neto, L. C., Ramalho, G. L., Neto, J. F. R., Veras, R. M., and Medeiros, F. N. (2017). An unsupervised coarse-to-fine algorithm for blood vessel segmentation in fundus images. Expert Systems with Applications, 78:182–192.

Neyshabur, B., Bhojanapalli, S., McAllester, D., and Srebro, N. (2017). Exploring generalization in deep learning. arXiv preprint arXiv:1706.08947.

Niemeijer, M., Staal, J., van Ginneken, B., Loog, M., and Abramoff, M. D. (2004). Comparative study of retinal vessel segmentation methods on a new publicly available database. In Medical imaging 2004: image processing, volume 5370, pages 648–656. International Society for Optics and Photonics.

Norberg, R. Å. (1975). Hovering flight of the dragonfly aeschna juncea l., kinematics and aerodynamics. In Swimming and flying in nature, pages 763–781. Springer.

Novikov, A. A., Lenis, D., Major, D., Hladůvka, J., Wimmer, M., and Bühler, K. (2018). Fully convolutional architectures for multiclass segmentation in chest radiographs. IEEE transactions on medical imaging, 37(8):1865–1876.

Oliveira, A., Pereira, S., and Silva, C. A. (2018). Retinal vessel segmentation based on fully convolutional neural networks. Expert Systems with Applications, 112:229–242.

Oliveira, H. and dos Santos, J. (2018). Deep transfer learning for segmentation of anatomical structures in chest radiographs. In 2018 31st SIBGRAPI Conference on Graphics, Patterns and Images (SIBGRAPI), pages 204–211. IEEE.

Orsini, F., Frasconi, P., and De Raedt, L. (2015). Graph invariant kernels. In Twenty-Fourth International Joint Conference on Artificial Intelligence.

Pancino, N., Rossi, A., Ciano, G., Giacomini, G., Bonechi, S., Andreini, P., Scarselli, F., Bianchini, M., and Bongini, P. (2020). Graph neural networks for the prediction of protein-protein interfaces. In ESANN, pages 127–132.

Paolino, G., Donati, M., Didona, D., Mercuri, S. R., and Cantisani, C. (2017). Histology of non-melanoma skin cancers: an update. Biomedicines, 5(4):71.

Papandreou, G., Kokkinos, I., and Savalle, P.-A. (2014). Untangling local and global deformations in deep convolutional networks for image classification and sliding window detection. arXiv preprint arXiv:1412.0296.

Park, T., Liu, M.-Y., Wang, T.-C., and Zhu, J.-Y. (2019). Semantic image synthesis with spatially-adaptive normalization. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pages 2337–2346.

Pathak, D., Krahenbuhl, P., Donahue, J., Darrell, T., and Efros, A. A. (2016). Context encoders: Feature learning by inpainting. In Proceedings of the IEEE conference on computer vision and pattern recognition, pages 2536–2544.

Patil, D. D. and Manza, R. R. (2016). Design new algorithm for early detection of primary open angle glaucoma using retinal optic cup to disc ratio. In 2016 International Conference on Electrical, Electronics, and Optimization Techniques (ICEEOT), pages 148–151.

Patton, N., Aslam, T. M., MacGillivray, T., Deary, I. J., Dhillon, B., Eikelboom, R. H., Yogesan, K., and Constable, I. J. (2006). Retinal image analysis: concepts, applications and potential. Progress in retinal and eye research, 25(1):99–127.

Peng, C., Zhang, X., Yu, G., Luo, G., and Sun, J. (2017). Large kernel matters – improve semantic segmentation by global convolutional network. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR).

Qi, X., Chen, Q., Jia, J., and Koltun, V. (2018). Semi-parametric image synthesis. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pages 8808–8816.

Qin, C., Yao, D., Shi, Y., and Song, Z. (2018). Computer-aided detection in chest radiography based on artificial intelligence: a survey. Biomedical engineering online, 17(1):1–23.

Ramon, J. and Gärtner, T. (2003). Expressivity versus efficiency of graph kernels. In Proceedings of the first international workshop on mining graphs, trees and sequences, pages 65–74.

Rastrelli, M., Tropea, S., Rossi, C. R., and Alaibac, M. (2014). Melanoma: epidemiology, risk factors, pathogenesis, diagnosis and classification. In vivo, 28(6):1005–1011.

Richter, S. R., Vineet, V., Roth, S., and Koltun, V. (2016). Playing for data: Ground truth from computer games. In European conference on computer vision, pages 102–118. Springer.

Ronneberger, O., Fischer, P., and Brox, T. (2015). U-net: Convolutional networks for biomedical image segmentation. In International Conference on Medical image computing and computer-assisted intervention, pages 234–241. Springer.

Ros, G., Sellart, L., Materzynska, J., Vazquez, D., and Lopez, A. M. (2016). The synthia dataset: A large collection of synthetic images for semantic segmentation of urban scenes. In Proceedings of the IEEE conference on computer vision and pattern recognition, pages 3234–3243.

Rosenblatt, F. (1958). The perceptron: a probabilistic model for information storage and organization in the brain. Psychological review, 65(6):386.

Rossi, A., Tiezzi, M., Dimitri, G. M., Bianchini, M., Maggini, M., and Scarselli, F. (2018). Inductive–transductive learning with graph neural networks. In IAPR Workshop on Artificial Neural Networks in Pattern Recognition, pages 201–212. Springer.

Rossi, A., Vannuccini, G., Andreini, P., Bonechi, S., Giacomini, G., Scarselli, F., and Bianchini, M. (2019). Analysis of brain nmr images for age estimation with deep learning. Procedia Computer Science, 159:981–989.

Roychowdhury, S., Koozekanani, D. D., and Parhi, K. K. (2015). Iterative vessel segmentation of fundus images. IEEE Transactions on Biomedical Engineering, 62(7):1738–1749.

Sagar, M. A., Bullivant, D., Mallinson, G. D., and Hunter, P. J. (1994). A virtual environment and model of the eye for surgical simulation. In Proceedings of the 21st annual conference on Computer graphics and interactive techniques, pages 205–212.

Santoro, A., Raposo, D., Barrett, D. G., Malinowski, M., Pascanu, R., Battaglia, P., and Lillicrap, T. (2017). A simple neural network module for relational reasoning. arXiv preprint arXiv:1706.01427.

Scarselli, F., Gori, M., Tsoi, A. C., Hagenbuchner, M., and Monfardini, G. (2008a). Computational capabilities of graph neural networks. IEEE Transactions on Neural Networks, 20(1):81–102.

Scarselli, F., Gori, M., Tsoi, A. C., Hagenbuchner, M., and Monfardini, G. (2008b). The graph neural network model. IEEE transactions on neural networks, 20(1):61–80.

Scarselli, F., Tsoi, A. C., and Hagenbuchner, M. (2018). The vapnik–chervonenkis dimension of graph and recursive neural networks. Neural Networks, 108:248–259.

Schadendorf, D. and Hauschild, A. (2014). Melanoma—the run of success continues. Nature reviews Clinical oncology, 11(2):75–76.

Sekou, T. B., Hidane, M., Olivier, J., and Cardot, H. (2019). From patch to image segmentation using fully convolutional networks–application to retinal images. arXiv preprint arXiv:1904.03892.

Serra, J. P. F. (1983). Image analysis and mathematical morphology.

Shah, S. A. A., Shahzad, A., Khan, M. A., Lu, C.-K., and Tang, T. B. (2019). Unsupervised method for retinal vessel segmentation based on gabor wavelet and multiscale line detector. IEEE Access, 7:167221–167228.

Shannon, C. E. (1948). A mathematical theory of communication. The Bell system technical journal, 27(3):379–423.

Shao, Y., Gao, Y., Guo, Y., Shi, Y., Yang, X., and Shen, D. (2014). Hierarchical lung field segmentation with joint shape and appearance sparse learning. IEEE transactions on medical imaging, 33(9):1761–1780.

Shervashidze, N., Schweitzer, P., Van Leeuwen, E. J., Mehlhorn, K., and Borgwardt, K. M. (2011). Weisfeiler-lehman graph kernels. Journal of Machine Learning Research, 12(9).

Shin, H.-C., Tenenholtz, N. A., Rogers, J. K., Schwarz, C. G., Senjem, M. L., Gunter, J. L., Andriole, K. P., and Michalski, M. (2018). Medical image synthesis for data augmentation and anonymization using generative adversarial networks. In International workshop on simulation and synthesis in medical imaging, pages 1–11. Springer.

Shiraishi, J., Katsuragawa, S., Ikezoe, J., Matsumoto, T., Kobayashi, T., Komatsu, K.-i., Matsui, M., Fujita, H., Kodera, Y., and Doi, K. (2000). Development of a digital image database for chest radiographs with and without a lung nodule: receiver operating characteristic analysis of radiologists' detection of pulmonary nodules. American Journal of Roentgenology, 174(1):71–74.

Shmelkov, K., Schmid, C., and Alahari, K. (2018). How good is my gan? In Proceedings of the European Conference on Computer Vision (ECCV), pages 213–229.

Soares, J. V., Leandro, J. J., Cesar, R. M., Jelinek, H. F., and Cree, M. J. (2006). Retinal vessel segmentation using the 2-d gabor wavelet and supervised classification. IEEE Transactions on medical Imaging, 25(9):1214–1222.

Sperduti, A. and Starita, A. (1997). Supervised neural networks for the classification of structures. IEEE Transactions on Neural Networks, 8(3):714–735.

Srivastav, D., Bajpai, A., and Srivastava, P. (2021). Improved classification for pneumonia detection using transfer learning with gan based synthetic image augmentation. In 2021 11th International Conference on Cloud Computing, Data Science & Engineering (Confluence), pages 433–437. IEEE.

Staal, J., Abramoff, M., Niemeijer, M., Viergever, M., and van Ginneken, B. (2004). Ridge-based vessel segmentation in color images of the retina. IEEE Transactions on Medical Imaging, 23(4):501–509.

Teixeira, L. O., Pereira, R. M., Bertolini, D., Oliveira, L. S., Nanni, L., Cavalcanti, G. D. C., and Costa, Y. M. G. (2021). Impact of lung segmentation on the diagnosis and explanation of covid-19 in chest x-ray images.

Toptaş, B. and Hanbay, D. (2021). Retinal blood vessel segmentation using pixel-based feature vector. Biomedical Signal Processing and Control, 70:103053.

Turing, A. M. and Ince, D. (1992). Collected works of alan turing, mechanical intelligence.

Van Ginneken, B., Stegmann, M. B., and Loog, M. (2006). Segmentation of anatomical structures in chest radiographs using supervised methods: a comparative study on a public database. Medical image analysis, 10(1):19–40.

Vapnik, V. N. (1998). Statistical Learning Theory. Wiley-Interscience.

Veličković, P., Cucurull, G., Casanova, A., Romero, A., Lio, P., and Bengio, Y. (2017). Graph attention networks. arXiv preprint arXiv:1710.10903.

von Neuman, J., Burks, A., Gardner, M., Wolfram, S., Wolfram, S., Sipper, M., Wolfram, S., Wolfram, S., Wolfram, S., Wolfram, S., Baldwin, J. T., et al. (1994). The theory of self-reproducing automata. IEEE Transactions on Neural Networks, 5(1):3–14.

Waheed, A., Goyal, M., Gupta, D., Khanna, A., Al-Turjman, F., and Pinheiro, P. R. (2020). Covidgan: data augmentation using auxiliary classifier gan for improved covid-19 detection. Ieee Access, 8:91916–91923.

Wang, C. (2017). Segmentation of multiple structures in chest radiographs using multi-task fully convolutional networks. In Scandinavian Conference on Image Analysis, pages 282–289. Springer.

Wang, H., Zeng, L., Liu, H., and Yin, C. (2003). Measuring wing kinematics, flight trajectory and body attitude during forward flight and turning maneuvers in dragonflies. Journal of Experimental Biology, 206(4):745–757.

Wang, K., Shen, Z., Huang, C., Wu, C.-H., Dong, Y., and Kanakia, A. (2020). Microsoft academic graph: When experts are not enough. Quantitative Science Studies, 1(1):396–413.

Wang, T.-C., Liu, M.-Y., Zhu, J.-Y., Tao, A., Kautz, J., and Catanzaro, B. (2018). High-resolution image synthesis and semantic manipulation with conditional gans. In Proceedings of the IEEE conference on computer vision and pattern recognition, pages 8798–8807.

WHO (2020a). Blindness and vision impairment.

WHO (2020b). Policy brief: The impact of covid-19 on children.

Xie, S. and Tu, Z. (2015). Holistically-nested edge detection. In 2015 IEEE International Conference on Computer Vision (ICCV), pages 1395–1403.

Xu, K., Hu, W., Leskovec, J., and Jegelka, S. (2018). How powerful are graph neural networks? arXiv preprint arXiv:1810.00826.

Yan, Z., Yang, X., and Cheng, K.-T. (2018). Joint segment-level and pixel-wise losses for deep learning based retinal vessel segmentation. IEEE Transactions on Biomedical Engineering, 65(9):1912–1923.

Yap, J., Yolland, W., and Tschandl, P. (2018). Multimodal skin lesion classification using deep learning. Experimental dermatology, 27(11):1261–1267.

Yi, X., Walia, E., and Babyn, P. (2018). Unsupervised and semi-supervised learning with categorical generative adversarial networks assisted by wasserstein distance for dermoscopy image classification. arXiv preprint arXiv:1804.03700.

Yi, Z., Zhang, H., Tan, P., and Gong, M. (2017). Dualgan: Unsupervised dual learning for image-to-image translation. In Proceedings of the IEEE international conference on computer vision, pages 2849–2857.

Yin, Y., Adel, M., and Bourennane, S. (2012). Retinal vessel segmentation using a probabilistic tracking method. Pattern Recognition, 45(4):1235–1244.

Zhao, H., Li, H., Maurer-Stroh, S., and Cheng, L. (2018). Synthesizing retinal and neuronal images with generative adversarial nets. Medical image analysis, 49:14–26.

Zhao, H., Shi, J., Qi, X., Wang, X., and Jia, J. (2017). Pyramid scene parsing network. In Proceedings of the IEEE conference on computer vision and pattern recognition, pages 2881–2890.

Zhao, Y., Rada, L., Chen, K., Harding, S. P., and Zheng, Y. (2015). Automated vessel segmentation using infinite perimeter active contour model with hybrid region information with application to retinal images. IEEE Transactions on Medical Imaging, 34(9):1797–1807.

Zhou, B., Khosla, A., Lapedriza, A., Oliva, A., and Torralba, A. (2014). Object detectors emerge in deep scene cnns. arXiv preprint arXiv:1412.6856.

Zhu, J.-Y., Park, T., Isola, P., and Efros, A. A. (2017). Unpaired image-to-image translation using cycle-consistent adversarial networks. In Proceedings of the IEEE international conference on computer vision, pages 2223–2232.

Zhu, J.-Y., Zhang, R., Pathak, D., Darrell, T., Efros, A. A., Wang, O., and Shechtman, E. (2018). Toward multimodal image-to-image translation.