

1 ALMA MATER STUDIORUM · UNIVERSITY OF BOLOGNA
2

3

4 School of Science
5 Department of Physics and Astronomy
6 Master Degree in Physics

7

Dataset Generation for the Training of 8 Neural Networks Oriented toward 9 Histological Image Segmentation

10

Supervisor:
Dr. Enrico Giampieri

Co-supervisor:
Dr. Nico Curti

11

Submitted by:
Alessandro d'Agostino

12

Academic Year 2019/2020

¹⁴ Abstract

The project is inspired by an actual problem of timing ~~end-and~~ accessibility in the analysis of histological samples in the health-care system. In this project, I face the problem of synthetic histological image generation for the purpose of training Neural Networks for the segmentation of real histological images. The ~~collection of real histological human-labeled samples is a very time consuming and expensive process and often is not representative of healthy samples, for the intrinsic nature of the medical analysis.~~ The method I propose is based on the replication of the traditional specimen preparation technique in a virtual environment. The first step is the creation of a 3D virtual model of a region of the target human tissue. The model should encapture all the key features of the tissue, and the richer it is the better will be the yielded result. The second step is to perform a sampling of the model through a virtual ~~seetingioning tomography~~ process, which produces a first ~~completely labeled~~ image of the section, ~~which will act as a segmentation mask.~~ This image is then processed with different tools to achieve a histological-like aspect. The most significant ~~contribution-aesthetical adjustment~~ is given by the action of a style transfer neural network that implants the typical ~~visual texture of a histological sample onto histological visual texture on~~ the synthetic image. This procedure is presented in detail for two specific models of human tissue: one of pancreatic tissue and one of dermal tissue. The two resulting images compose a pair of images ~~and corresponding ground-truth image, which is~~ perfectly suitable for a supervised learning technique. The generation process is completely automatized and does not require the intervention of any human operator, hence it can be used to produce arbitrary large datasets. The synthetic images are inevitably less complex than the real samples and they offer an easier segmentation task to solve for the NN. However, the synthetic images are very abundant, and the training of a NN can take advantage of this feature, following the so-called curriculum learning strategy.

⁴⁰ Table of Contents

⁴¹	Introduction	6
⁴²	1 Theoretical Background	12
⁴³	1.1 Histological Images <u>Digitalization</u>	12
⁴⁴	1.1.1 Slides Preparation for Optic Microscopic Observation	15
⁴⁵	1.1.2 <u>Pancreas Microanatomy and Tumoral Evidences</u>	16
⁴⁶	1.1.3 <u>Skin Microanatomy and Tumoral Evidences</u>	18
⁴⁷	1.2 Introduction to Deep Learning	21
⁴⁸	1.2.1 Perceptrons and Multilayer Feedforward Architecture	21
⁴⁹	1.2.2 Training <u>of a NN - Error Back-Propagation Strategies in Deep Learning</u>	23
⁵⁰	1.2.3 <u>Training Algorithms - Error Back-Propagation</u>	25
⁵¹	1.3 Deep Learning-Based Segmentation Algorithms	29
⁵²	1.3.1 State of the Art on Deep Learning Segmentation	30
⁵³	1.3.2 Image Segmentation Datasets	35
⁵⁴	2 Technical Tools for Model Development	38
⁵⁵	2.1 Quaternions	39
⁵⁶	2.2 Parametric L-Systems	40
⁵⁷	2.3 Voronoi Tassellation	42
⁵⁸	2.4 Saltelli Algorithm - Randon Number Generation	44
⁵⁹	2.5 Planar Section of a Polyhedron	46
⁶⁰	2.6 Perlin Noise	48
⁶¹	2.7 Style-Transfer Neural Network	50
⁶²	2.8 Working Environment	53
⁶³	3 Tissue Models Development	55
⁶⁴	3.1 Pancreatic Tissue Model	55

65	3.1.1	2D Ramification	56
66	3.1.2	Expansion to 3D	57
67	3.1.3	Subdivision in Cells	58
68	3.1.4	Cells Identity Assignment	61
69	3.2	Dermal Tissue Model	62
70	3.3	Synthetic Images Production	66
71	3.3.1	Sectioning Process	66
72	3.3.2	Appearance Makeover	68
73	Conclusions		75
74	Bibliography		78

⁷⁶ Introduction

⁷⁷ In the last decades, the development of Machine Learning (ML) and Deep Learning (DL)
⁷⁸ techniques has contaminated every aspect of the scientific world, with interesting results
⁷⁹ in many different research fields. The biomedical field is no exception to this and a
⁸⁰ lot of promising applications are taking form, especially as Computer-Aided Detection
⁸¹ (CAD) systems which are tools for the support for physicians during the diagnostic
⁸² process. Medical doctors and the healthcare system in general collect a huge amount of
⁸³ data from patients during all the treatment, screening, and analysis activities in many
⁸⁴ different shapes, from anographical data to blood analysis to clinical images.

⁸⁵ In fact in medicine, the study of images is ubiquitous and countless diagnostic proce-
⁸⁶ dures rely on it, such as X-ray imaging (CAT), nuclear imaging (SPECT, PET), Magnetic
⁸⁷ resonance, and visual inspection of histological specimens after biopsies. The branch of
⁸⁸ artificial intelligence in the biomedical field that handles image analysis to assist physi-
⁸⁹ cians in their clinical decisions goes under the name of Digital Pathology Image Analysis
⁹⁰ (DPIA). In this thesis work, I want to focus on some of the beneficial aspects intro-
⁹¹ duced by DPIA in the histological images analysis and some particular issues in the
⁹² development of DL models able to handle this kind of procedure.

⁹³ Nowadays the great majority of analysis of histological specimens occurs through
⁹⁴ visual inspection, carried out by highly qualified experts. Some analysis, as cancer
⁹⁵ detection, requires the ability to distinguish if a region of tissue is healthy or not with high
⁹⁶ precision in very wide specimens. This kind of procedure is typically very complex and
⁹⁷ requires prolonged times of analysis besides substantial economic efforts. Furthermore,
⁹⁸ the designated personnel for this type of analysis is often limited, leading to delicate issues
⁹⁹ of priority assignment while scheduling analysis, based on the estimated patient's clinical
¹⁰⁰ development. Some sort of support to this analysis procedure is therefore necessary.

¹⁰¹ The problem of recognizing regions with different features within an image and de-
¹⁰² tect their borders is known in computer vision as the segmentation task, and it's quite
¹⁰³ widespread with countless different applications, allowing a sort of automatic image in-
¹⁰⁴ terpretation. In ML the segmentation problem is usually faced as a supervised task,
¹⁰⁵ hence the algorithm in order to be trained properly requires an appropriate quantity
¹⁰⁶ of pre-labeled images, from which learn the rules through which distinguish different
¹⁰⁷ regions. This means that the development of segmentation algorithms for a specific ap-

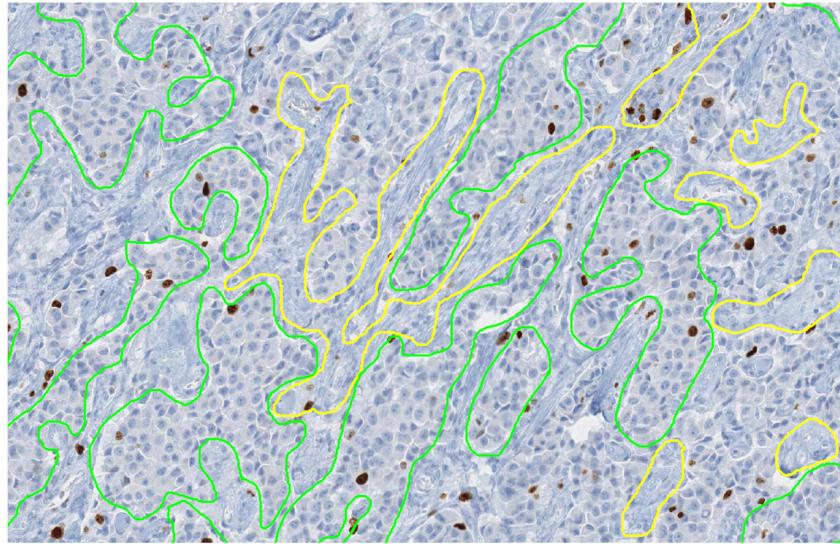


Figure 1: Interleaving of tumor (green annotation) and non-tumor (yellow annotation) regions [29].

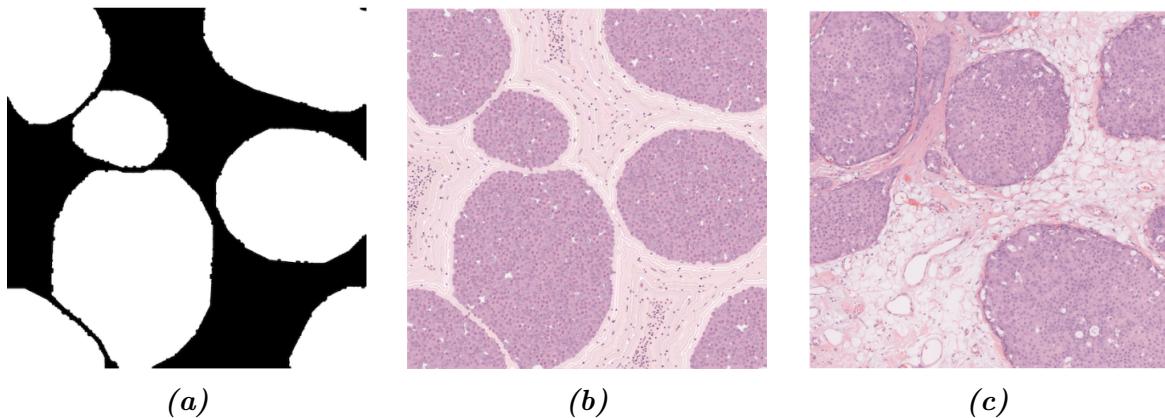
108 plication, as would be the one on histological images, would require a lot of starting
 109 material, previously analyzed from the same qualified expert encharged of the visual
 110 inspection mentioned before. A human operator thus is required to manually track the
 111 boundaries, for example, between healthy and tumoral regions within a sample of tissue
 112 and to label them with their identity, as in Figure 1. The more the algorithm to train
 113 is complex the more starting material is required to adjust the model’s parameters and
 114 reach the desired efficacy.

115 The latest developed segmentation algorithms are based on DL techniques, hence
 116 based on the implementation of intricated Neural Networks (NN) which process the
 117 input images and produce the corresponding segmentation. Those models are typically
 118 very complex, with millions of parameters to adjust and tune, therefore they need a
 119 huge amount of pre-labeled images to learn their segmentation rules. This need for data
 120 is exactly the main focus of my thesis work. The shortage of ground truth images is
 121 indeed one of the toughest hurdles to overcome during the development of DL-based
 122 algorithms. Another important aspect to bear in mind is the quality of the ground truth
 123 material. It’s impossible for humans to label boundaries of different regions with pixel-
 124 perfect precision, while for machines the more precise is the input the more effective is
 125 the resulting algorithm.

126 Different approaches have already been explored to overcome this problem, and they
 127 are mainly based on the generation of synthetic labeled data to use during the training
 128 phase. Some techniques achieve data augmentation manipulating already available
 129 images and then generating *new* images, but as we will see this approach suffers from

130 different issues. Here, I want to make an overview of some other interesting works on
131 the generation of synthetic histological images, which have followed completely different
132 paths and strategies from mine.

133 The first work I want to cite is a work from Ben Cheikh *et al.* from 2017 [9]. In
134 this work, they present a methodology for the generation of synthetic images of different
135 types of breast carcinomas. They propose a method completely based on two-dimensional
136 morphology operation, as successive image dilations and erosions. With the modulation
137 of a very restricted number of parameters, regulating the abundance of the objects, their
138 distribution in the image, and their shapes they are able to reconstruct realistic images
139 reflecting different histological situations. In Figure 2 is shown an example of generated
140 material besides a real histological H&E stained sample. Starting from a generated
141 segmentation mask which defines the *tumoral pattern* the production of the synthetic
142 image passes through successive steps, as the generation of characteristic collagen fibers
143 around the structure, the injection of all the immune system cells, and some general final
144 refinements.



145 **Figure 2:** Example of generated tumoral pattern (left), which acts as seg-
146 mentation mask, of generated image (center) and a real example of the tissue
147 to recreate, from [9].

148 The second work I want to mention is based on a DL-base technique, which ap-
149 proaches synthetic image generation using a specific cGAN architecture inspired to the
150 “U-net” [39] model, as will be described in Figure 1.15 in section 1.3.1. This model
151 works with Ki67 stained samples of breast cancer tissue, and it is able to generate high-
152 fidelity images starting from a given segmentation mask. Those starting segmentation
153 masks tough are obtained through the processing of other real histological samples, via
154 a nuclei-detection algorithm. The differences between real and synthetic samples are
imperceptible, and the material generated in this work has effectively fooled experts,
who qualified it has indistinguishable from the real one. In Figure 3 an example of a real
image, a generated one, and their corresponding segmentation mask.

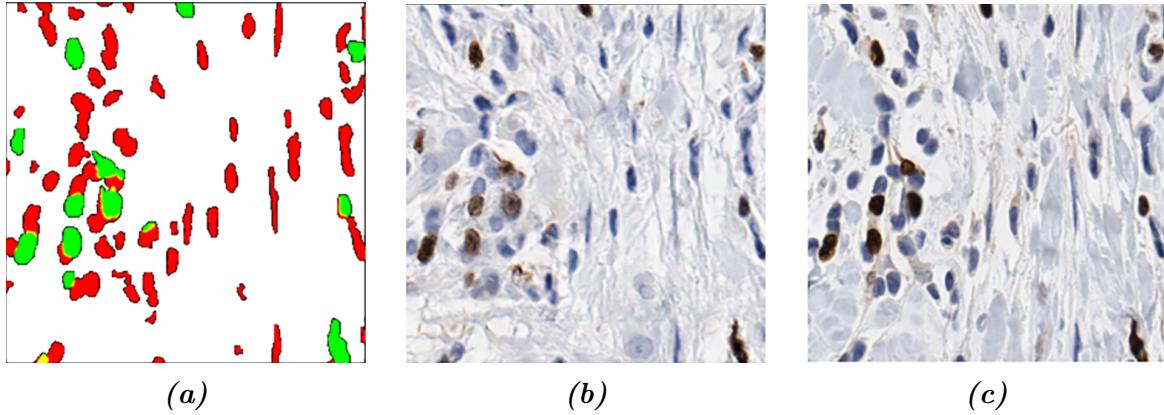


Figure 3: Example of generated tumoral pattern (left), which acts as segmentation mask, of generated image (center) and a real example of the tissue to recreate, from [39].

Both of the two before-mentioned strategies ~~produces~~ produce realistic (or even perfect) results, but they are based ~~onto~~ on considerations and analysis limited only to the aspect of the images. In the first work, the segmentation mask is produced in an almost full-random way, while in the second the segmentation mask is extracted starting from ~~an~~ actual real histological samples. The target of the present work instead lies in between those two approaches, and wants to produce randomized new images following a plausible modelization based on physical and histological considerations.

The technique I propose in this work follows a generation from scratch of entire datasets suitable for the training of new algorithms, based on the 3D modelization of a region of human tissue at the cellular level. The entire traditional sectioning process, which is made on real histological samples, is recreated virtually on this virtual model. This yields pairs of synthetic images with their corresponding ground-truth. Using this technique one would be able to collect sufficient material for the training (the entire phase or the preliminary part) of a model, overcoming the shortage of hand-labeled data. The 3D modeling of a region of particular human tissue is a very complex task, and it is almost impossible to capture all the physiological richness of a histological system. The models I implemented thus are inevitably less sophisticated respect the target biological structures. I'll show two models: one of pancreatic tissue and another of dermal tissue, besides all the tools I used and the choices I made during the designing phase.

Furthermore, since the image production passes through a wide and elaborated model, the resulting images contain a new level of semantic information that would not be capturable otherwise. From the modelization is possible to reflect on the segmentation mask image the relationship information between nuclei and their belonging cells, or the basins of blood irrigation corresponding to every blood vessel and many other pieces of information about the interrelationship of depicted elements. Moreover, achieving the

right mastery of the modelization it is possible to reproduce different physiological states of the tissue like the healthy *standard* configuration or different pathological situations, which reflect themselves as particular arrangements at the cellular level. This aspect is of great interest, in fact, there is a strong lack of real histological samples of healthy tissue samples, given the intrinsic nature of the medical analysis. Biopsies are usually invasive procedures and are typically performed only when there is a concern for a pathological situation. Unless an erroneous evaluation, they typically collect a sample of non-healthy tissue. This method thus allows us to collect an arbitrary number of samples in every interesting modeled condition, overcoming the problem of under-represented conditions.

In order to present organically all the steps of my work the thesis is organized in chapters as follows:

1. Theoretical Background .

In this chapter, I will describe how real histological images are obtained and their digitalization process works. Afterward, I will introduce the reader to the Deep Learning framework, explaining the key elements of this discipline and how they work. Finally, I will dedicate a section to the image segmentation problem, and the state of the art of segmentation DL-based algorithms, with particular attention to the applications in the bio-medical field.

2. Technical Tools for Model Development .

I will dedicate this chapter to the thorough description of every technical tool I needed during the designing phase of this project. The development has required the harmonization of many different technologies and mathematical tools, some of which not so popular like quaternions, quasi-random number generation, Voronoi decomposition, and style transfer neural networks. In this chapter, I will also describe my working environment ~~describe also the specialized algorithm I devised and implemented for the sectioning of an arbitrary polyhedron, which is the key element for the correct working of the virtual tomography technique described by this thesis work. As a conclusion for this chapter, I will describe the working environment I built for developing this project and I will mention all the code libraries I employed in my work.~~

3. Tissue Models Development .

This third chapter is the heart of the project. I will describe in detail all the steps necessary to create the two models, one of pancreatic tissue and the other of dermal tissue, and how I am able to produce the synthetic images. The first section is devoted to the modeling of the histological structures, while the second is entirely dedicated to the sectioning process and the subsequent refinements to the images.

²¹⁹ **Chapter 1**

²²⁰ **Theoretical Background**

²²¹ In this first chapter, I will depict the theoretical context of the work. Section 1.1 will
²²² be dedicated to histological images, and the different techniques used to prepare the
²²³ samples to analyze. Histological images recover a fundamental role in medicine and are
²²⁴ the pillar of many diagnosis techniques. This discipline borns traditionally from the
²²⁵ optical inspection of the tissue slides using a microscope, and it is gradually developing
²²⁶ and improving with the advent of computers and digital image processing. It is important
²²⁷ tough to understand how the samples are physically prepared, the final target of this
²²⁸ work is in fact the virtual reconstruction of this process. The section comes to an end
²²⁹ with an introduction to the anatomy of the two particular tissues under the attention
²³⁰ of this project, a brief introduction to the most common neoplasms that interest the
²³¹ pancreas and skin, and their shreds of evidence at the histological level. In section 1.2 I
²³² will introduce the Deep Learning framework and describe how a Neural Network works
²³³ and actually learns. The most advanced techniques for the automatic image processing
²³⁴ implement Deep Learning algorithm, and understanding the general rules behind this
²³⁵ discipline is crucial for a good comprehension of this work. In section 1.3 I will discuss in
²³⁶ particular the problem of image segmentation and how it is tackled with different Neural
²³⁷ Network architectures, showing what it is the state of the art of this research field.

²³⁸ **1.1 Histological Images Digitalization**

²³⁹ Modern histopathology is essentially based on the careful interpretation of microscopic
²⁴⁰ images, with the intention of correctly diagnose patients and to guide therapeutic de-
²⁴¹ cisions. In the last years, thanks to the quick development of scanning techniques and
²⁴² image processing, the discipline of histology have seen radical improvements: the main
²⁴³ of which undoubtedly is the passage from the microscope's oculars to the computer's
²⁴⁴ screen. This digitalization process has brought several advantages, that were previously
²⁴⁵ impossible in classical histology, like telepathology and remote assistance in diagnosis

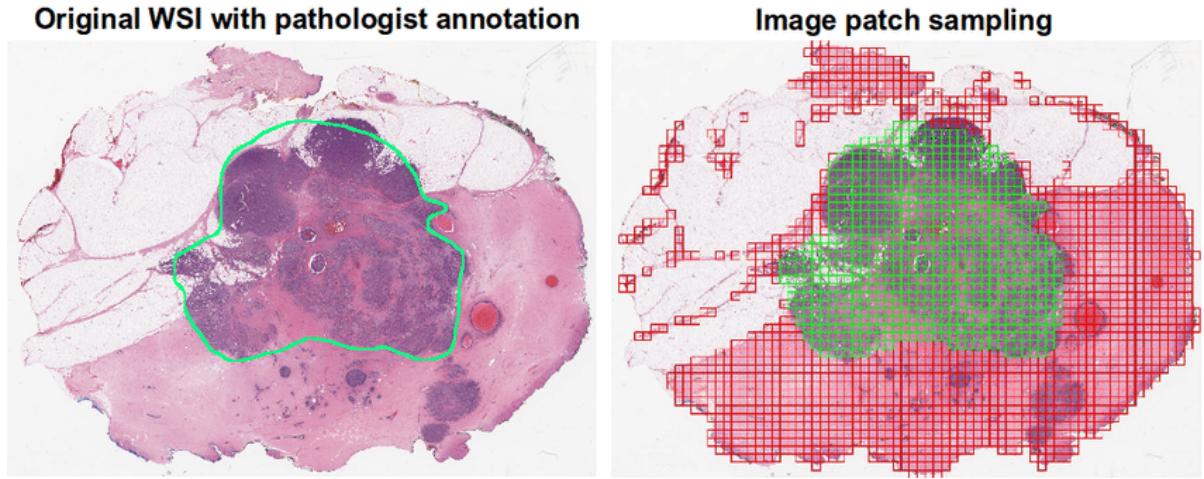


Figure 1.1: An example of whole slide image, with its grid decomposition in patches. It is visible the correspondence between a region of interest manually annotated and the patches that matches that region. From [13]

processes, the integration with other digitalized clinical workflows, and patients' history, and most importantly the opening to applications of artificial intelligence. The name Whole Slide Imaging (WSI) refers to the modern virtual microscopy discipline, which consists of scanning a complete microscope slide and creating a single high-resolution digital file. This is commonly achieved by capturing many small high-resolution image tiles or strips and then montaging them to create a full image of a histological section. The four key steps of this process are image acquisition (scansion), editing, and on-screen image visualization.

In the field of Digital Pathology (DP) an essential concept in image understanding is the magnification factor, which indicates the scale of representation of the image and allows dimension referencing. This factor is usually indicated as the magnification power of the microscope's lenses used during the analysis. After the digitalization process, this original magnification factor is prone to change, depending on the resolution of the visualization screen. Therefore, image resolution is measured in μm per pixel, and it is set by the different composition of the acquisition chain, as the optical sensor and the lenses. Histological scanner are usually equipped with $20\times$ or $40\times$ objectives, which correspond to 0.5 and 0.25 mm/pixels resolution values. Lenses with $20\times$ magnification factor are the most suitable for the great majority of histopathological evaluations, and it is the golden standard for scansions, for its good trade-off between image quality and time of acquisition. Scansions with $40\times$ magnification could increase four-fold acquisition and processing time, final file's dimension, and storage cost. A single WSI image, acquired with $20\times$ will occupy more than 600 MB alone.

Despite the WSI is a relatively mature discipline, it still struggles to integrate itself in

269 the standard primitive diagnosis phase in histopathological laboratories. This is primarily
270 due to some disadvantages, like images' resolution, image compression's artifacts, and
271 auto-focusing algorithms, which plays a key role in the specimen interpretation. Furthermore,
272 the scansion of histological samples is an additional step in the analysis which
273 takes time. Despite the technological improvements the average time for the acquisition
274 of a sample is around 5/10 minutes, depending on the number of slices in the slide, for
275 just a single level of magnification. While in traditional histology, the pathologist has
276 access to all the magnification levels at the same time. The real advantage, in fact, is in
277 the long term. Once the images have been acquired they can be archived and consulted
278 remotely almost instantaneously, helping clinical analysis and allowing remote assistance
279 (telemedicine). Furthermore, the images now can be processed by artificial intelligence
280 algorithms, allowing the application of technologies like Deep Learning which could rev-
281 olutionize the research field, as already has been on many different disciplines in the
282 scientific world.

283 In order to allow to automatically process, such big images as the ones obtained
284 through WSI, it is necessary to subdivide them in smaller patches. The dimension of
285 which should be big enough to allow interpretation and to preserve a certain degree of
286 representability of the original image. In Figure 1.1 is shown an example of a whole
287 slide image, with its grid decomposition in patches. If the patches are too small, it
288 should be over-specified for a particular region of tissue, loosing its general features.
289 This could lead the learning algorithm to misinterpretation. However, this is not an
290 exclusive limit of digital pathology, for a human pathologist would be impossible too to
291 make solid decisions on a too limited sample of tissue. After the subdivision in patches, a
292 typical process for biomedical images is the so-called *data augmentation* of images, that
293 is the process of creating re-newed images from the starting material through simple
294 geometrical transformations, like translation, rotation, reflection, zoom in/out.

295 The analysis of histological images usually consists in-of detecting the different compo-
296 nents in the samples and to recognise their arrangement as-an recognize their arrangement
297 as a healthy or pathological pattern. It is necessary to recognize every sign of the vitality
298 of the cells, evaluating the state of the nucleus. There are many addittionaladditional
299 indicators to consider like the presence of inflammatory cells or tumoral cells. Further-
300 more, samples which are taken from different part of the human body present completely
301 different characteristics, and this increase greatly the complexity of the analysis. A reli-
302 able esaminationexamination of a sample thus requirerequires a careful inspection made
303 by an high-a highly qualified expert. The automatization of this procedure would be
304 extremilyextremely helpful, giving an inereadibleincredible boost both in timing and
305 in-accessibility. However, this is not a simple task and in section 1.3.1 I will show some
306 actual model for biomedical image processing in detail.

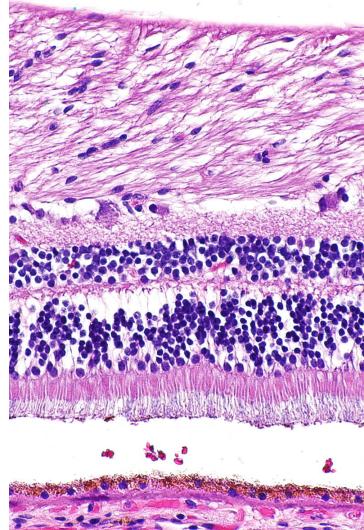


Figure 1.2: A sample of tissue from a retina (a part of the eye) stained with hematoxylin and eosin, cell nuclei stained blue-purple and extracellular material stained pink.

307 1.1.1 Slides Preparation for Optic Microscopic Observation

308 In modern, as in traditional, histology regardless ~~on of~~ the final support of the image the
309 slide has to be physically prepared, starting from the sample of tissue. The sample and
310 slide preparation is a crucial step for histological or cytological observation. It is essential
311 to highlight what needs to be observed and to *immobilize* the sample at a particular point
312 in time and with characteristics close to those of its living state. There are five key steps
313 for the preparation of samples [4]:

314 1) **Fixation** is carried out immediately after the removal of the sample to be observed.
315 It is used to immobilize and preserve the sample permanently in as life-like state
316 as possible. It can be performed immersing the biological material in a formalin
317 solution or by freezing, so immersing the sample in a tissue freezing medium which
318 is then cooled in liquid nitrogen.

319 2) **Embedding** if the sample has been ~~stabilized~~ stabilized in a fixative solution, this
320 is the subsequent step. It consists in hardening the sample in a paraffin embedding
321 medium, in order to be able to carry out the sectioning. It is necessary to dehy-
322 drate the sample beforehand, by replacing the water molecules in the sample with
323 ethanol.

324 3) **Sectioning** Sectioning is performed using microtomy or cryotomy. Sectioning is
325 an important step for the preparation of slides as it ensures a proper observa-
326 tion of the sample by microscopy. Paraffin-embedded samples are cut by ~~cross~~

327 ~~section~~cross-section, using a microtome, into thin slices of $5\ \mu m$. Frozen samples
328 are cut using a cryostat. The frozen sections are then placed on a glass slide for
329 storage at $-80^{\circ}C$. The choice of these preparation conditions is crucial in order
330 to minimize the artifacts. Paraffin embedding is favored for preserving tissues;
331 freezing is more suitable for preserving DNA and RNA and for the labeling of
332 water-soluble elements or of those sensitive to the fixation medium.

333 **4) Staining** Staining increases contrasts in order to recognize and differentiate the dif-
334 ferent components of the biological material. The sample is first deparaffinized and
335 rehydrated so that polar dyes can impregnate the tissues. The different dyes can
336 thus interact with the components to be stained according to their affinities. Once
337 staining is completed, the slide is rinsed and dehydrated for the mounting step.

338 Hematoxylin and eosin stain (H&E) is one of the principal tissue stains used in
339 histology [45], and it is the most widely used stain in medical diagnosis and is often
340 the gold standard [36]. H&E is the combination of two histological stains: hematoxylin
341 and eosin. The hematoxylin stains cell nuclei blue, and eosin stains the extracellular
342 matrix and cytoplasm pink, with other structures taking on different shades, hues, and
343 combinations of these colors. An example of H&E stained is shown in Figure 1.2, in
344 which we can see the typical ~~colour palette of an histological specimen~~color palette of a
345 histological specimen.

346 1.1.2 Pancreas Microanatomy and Tumoral Evidences

347 The Pancreas is an internal organ of the human body, part of both the digestive system
348 and the endocrine system. It acts as a gland with both endocrine and exocrine functions,
349 and it is located in the abdomen behind the stomach. Its main endocrine duty is the
350 secretion of hormones like insulin and glucagon which are responsible for the regulation
351 of sugar levels in the blood. As a part of the digestive system instead, it acts as an
352 exocrine gland secreting pancreatic juice, which has an essential role in the digestion of
353 many different nutritional compounds. The majority of pancreatic tissue has a digestive
354 role, and the cells with this role form clusters called *acini* around the small pancreatic
355 ducts. The acinus secrete inactive digestive enzymes called zymogens into the small
356 intercalated ducts which they surround, and then in the pancreatic blood vessels system
357 [27]. In Figure 1.3 is shown a picture of the pancreas, with its structure and its placement
358 in the human body. All the tissue is actually rich in other important elements as the
359 islets of Langerhans that have an important role in the endocrine action of the pancreas.
360 All over the structure is present a layer of connective tissue which are clearly visible in
361 the traditional histological specimens.

362 There are many tumoral diseases interest the pancreas, they represent one of the main
363 causes of decease for cancer in occidental countries, and its incidence rate in Europe
364 and the USA has risen significantly in the last decades. There are two main kinds

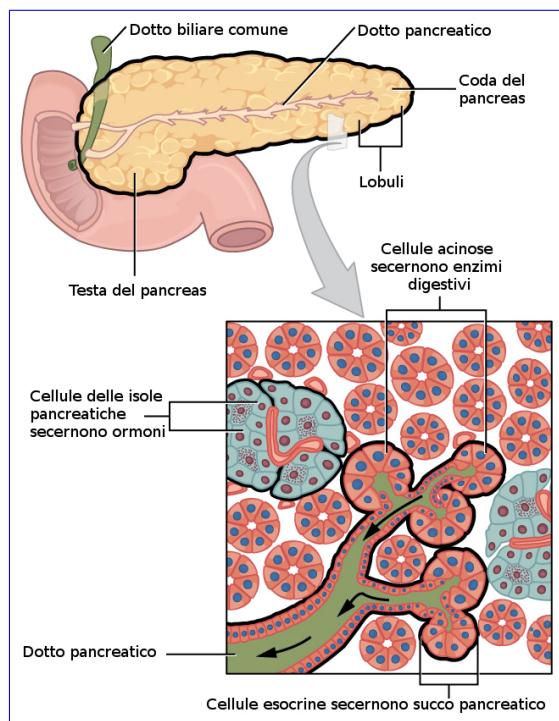


Figure 1.3: A picture of pancreas' structure in its physiological context. In this picture is clearly visible the macroscopic structure and the glandular organization at microscopic level.

365 of pancreatic neoplasms: endocrine and exocrine pancreatic tumors. The endocrine
366 pancreas neoplasms derive from the cells constituting the Langherans' islets and are
367 typically divided into functioning and non-functioning ones, depending on the capability
368 of the organs to secrete hormones. Those diseases might be benign or malignant and
369 they can have a different degree of aggressivity. Exocrine pancreatic tumors tough are
370 the most frequent ones. Among those, the great majority of episodes are of malignant
371 neoplasms, and in particular, the ductal adenocarcinoma is the most frequent form of
372 disease, responsible alone for the 95% of the cases. From a macroscopic point of view,
373 those tumors are characterized by an abundant fibrotic stroma¹, which can represent
374 over 50% of the tumor's mass and it is responsible for the hard-ligneous tumor's aspect.
375 This disease is frequently followed by pancreatitis episodes.

376 From a microscopical point of view, this tumor is characterized by the presence of
377 glandular structures, made by one or more layers of columnar or cuboidal epithelial cells
378 embedded in fibrotic parenchyma. The histological inspection of a sample of pancreatic
379 tissue allows us to grade the stadium of the disease. While analyzing a specimen the
380 operator looks for some specific markers like the glandular differentiation in tubular
381 and ductal structure, the degree of production of mucin, the percentage of cells in the
382 mitotic phase, and the degree of tissue nuclear atypia. The evaluation of those characters
383 allows us to assign a degree of development form I to III to the specific case of ductal
384 adenocarcinoma.

385 During the microscopical analysis of a histological sample, there are other interesting
386 markers to be evaluated, as a specific set of lesions which goes under the name of PanIN
387 (Pancreatic Intraepithelial Neoplasia) and could not be appreciated with other diagnostic
388 techniques. PanIN describes a wide variety of morphological modifications, differentiated
389 on the degree of cytological atypia and architectural alterations [20]. A careful analysis
390 of a histological sample of tissue after a biopsy is a fundamental step in the treatment
391 of a patient.

392 1.1.3 Skin Microanatomy and Tumoral Evidences

393 Skin is the layer of soft, flexible outer tissue covering the body of a vertebrate animal,
394 with the three main functions of protection, regulation, and sensation. Mammalian
395 skin is composed of two primary layers: the epidermis, which provides waterproofing
396 and serves as a barrier to infection, and the dermis, which serves as a location for the
397 appendages of skin. The epidermis is composed of the outermost layers of the skin. It
398 forms a protective barrier over the body's surface, responsible for keeping water in the
399 body and preventing pathogens from entering, and is a stratified squamous epithelium,
400 composed of proliferating basal and differentiated suprabasal keratinocytes. The dermis

¹Stroma is the part of a tissue or organ with a structural or connective role. It is made up of all the parts without specific functions of the organ like connective tissue, blood vessels, ducts, etc. The other part, the parenchyma, consists of the cells that perform the function of the tissue or organ.

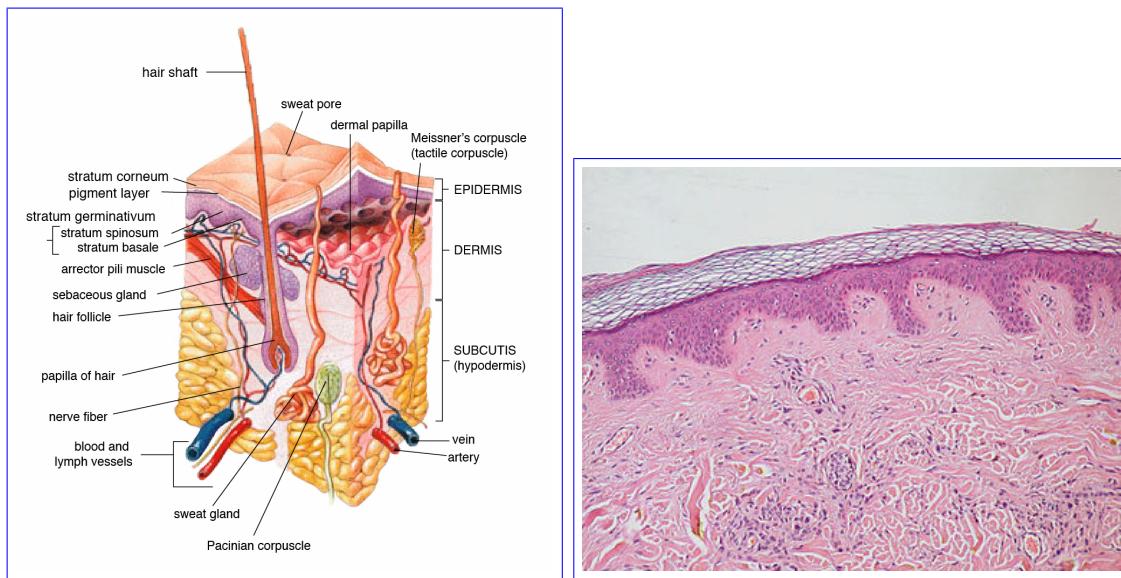


Figure 1.4: (left) Microanatomical description of a region of dermal tissue and all the interesting elements present in cutis, and subcutaneous layer. (right) An actual histological specimen from a sample of dermal tissue.

401 is the layer of skin beneath the epidermis that consists of connective tissue and cushions
 402 the body from stress and strain. The dermis provides tensile strength and elasticity to
 403 the skin through an extracellular matrix composed of collagen fibrils, microfibrils, and
 404 elastic fibers, embedded in hyaluronan and proteoglycans.

405 Melanoma is the most aggressive form of skin tumor. It is the second most frequent
 406 tumors in men under 50 years, and the third for women under 50 years, with over
 407 than 12.300 cases in Italy every year. Melanoma is considered nowadays a multifactorial
 408 pathology, which originates from the interaction between genetic susceptibility and environmental
 409 exposure. The most important environmental risk factor is the intermittent solar exposure,
 410 for the genotoxic effect of ultraviolet rays on the skin. Different studies show also a strong
 411 correlation between the total number of nevi on the skin and the incidence of melanoma.

412
 413 We can distinguish two separated phases in the growth of melanoma: the radial phase,
 414 in which the proliferation of malignant melanocytes is limited to the epidermis, and the
 415 vertical phase, where malignant melanocytes form nests or nodules in the dermis. The
 416 dermatological analysis should distinguish which specific type of melanoma is affecting
 417 the patient. There are many types of melanoma:

418 **Superficial Spreading Melanoma** : This is the most common subtype, and it represents
 419 alone more than the 75% of all melanoma cases. These neoplasms show relatively
 420 large malignant melanocytes, inflammatory cells (epithelioid), and an abundance

421 of cytoplasm.

422 **Lentigo Maligna Melanoma** : It typically arises in photodamaged skin regions. Neoplastic
423 melanocytes are of polygonal shape, with hyperchromatic nuclei, and they are
424 followed by a reduction in the cytoplasm.

425 **Acral lentiginous melanoma** : It typically arises on palmar, plantar, subungual, or
426 mucosal surfaces. Melanocytes are usually arranged along the dermal-epidermal
427 junction. The progression of this type of melanoma is characterized by the presence
428 of large junctional nests of atypical melanocytes, which are extended and hyperchromatic,
429 with a shortage of cytoplasm.

430 **Modular Melanoma** : By definition, this is the melanoma with a pure vertical growth.
431 It is followed by the presence of numerous little tumoral nests of neoplastic melanocytes,
432 with a high rate of mitotic state, arranged to form a single big nodule.

433 The careful examination of histological samples extracted from the tissue under
434 analysis is the most important step for the assessment of the actual form of neoplasms.

435 1.2 Introduction to Deep Learning

436 Deep Learning is part of the broader framework of Machine Learning and Artificial
437 Intelligence. Indeed all the problems typically faced using ML can also be addressed
438 with DL techniques, for instance, regression, classification, clustering, and segmentation
439 problems. We can think of DL as a universal methodology for iterative function ap-
440 proximation with a great level of complexity. In the last decades, this technology has
441 seen a frenetic diffusion and an incredible development, thanks to the always increasing
442 available computational power, and it has become a staple tool in all sorts of scientific
443 applications.

444 1.2.1 Perceptrons and Multilayer Feedforward Architecture

445 Like other artificial learning techniques, DL models aim to *learn* a relationship between
446 some sort of input and a specific kind of output. In other words, approximating nu-
447 merically the function that processes the input data and produces the desired response.
448 For example, one could be interested in clustering data in a multidimensional features
449 space, or the detection of objects in a picture, or text manipulation/generation. The
450 function is approximated employing a greatly complex network of simple linear and non-
451 linear mathematical operations arranged in a so-called Neural Network (typically with
452 millions of parameters). The seed idea behind this discipline is to recreate the function-
453 ing of actual neurons in the human brain: their entangled connection system and their
454 “ON/OFF” behavior [42].

455 The fundamental unit of a neural network is called perceptron, and it acts as a digital
456 counterpart of a human neuron. As shown in Figure 1.5 a perceptron collects in input a
457 series on n numerical signals $\vec{x} = 1, x_1, \dots, x_n$ and computes a linear weighted combination
458 with the weights vectors $\vec{w} = w_0, w_1, \dots, w_n$, where w_0 is a bias factor:

$$f(\vec{x}, \vec{w}) = \chi(\vec{x} \cdot \vec{w}). \quad (1.1)$$

459 The results of this linear combination are given as input to a non-linear function $\chi(x)$
460 called the activation function. Typical choices as activation function are any sigmoidal
461 function like $\text{sign}(x)$ and $\tanh(x)$, but in more advanced applications other functions
462 like ReLU [1] are used. The resulting function $f(\vec{x}, \vec{w})$ has then a simple non linear
463 behaviour. It produces a binary output: 1 if the weighted combination is high enough
464 and 0 if it is low enough, with a smooth modulation in-between the two values.

465 The most common architecture for a NN is the so-called *feed-forward* architecture,
466 where many individual perceptrons are arranged in chained layers, which take as input
467 the output of previous layers along with a straight information flux. More complex ar-
468 chitectures could implements also recursive connection, linking a layer to itself, but it
469 should be regarded as sophistication to the standard case. There are endless possibilities

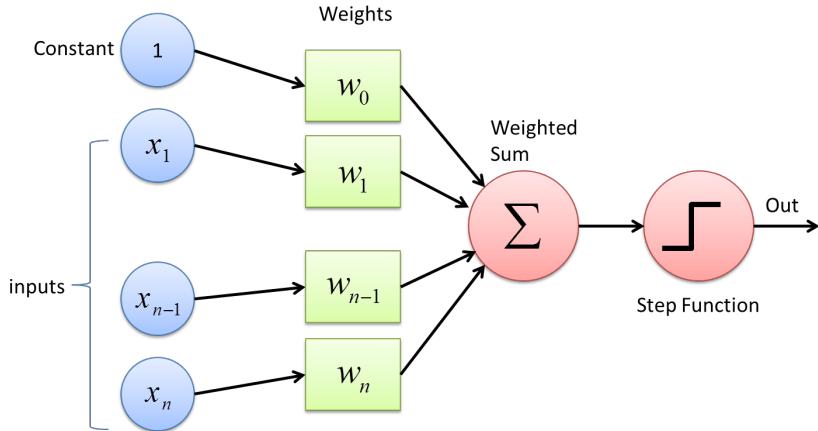


Figure 1.5: Schematic picture of a single layer perceptron. The input vector is linearly combined with the bias factor and sent to an activation function to produce the numerical "binary" output.

of combination and arrangement of neurons inside a NN's layer, but the most simple ones are known as *fully-connected* layers, where every neuron is linked with each other neuron of the following layer, as shown in Figure 1.6. Each connection has its weight, which contributes to modulate the overall combination of signals. The training of a NN consists then in the adjustment and fine-tuning of all the network's weights and parameters through iterative techniques until the desired precision in the output generation is reached.

Although a fully connected network represents the simplest linking choice, the insertion of each weight increases the number of overall parameters, and so the complexity of the model. Thus we want to create links between neurons smartly, rejecting the less useful ones. Depending on the type of data under analysis there are many different established typologies of layers. For example, in the image processing field, the most common choice is the convolutional layer, which implements a sort of discrete convolution on the input data, as shown in Figure 1.7. While processing images, the convolution operation confers to the perception of correlation between adjacent pixels of an image and their color channels, allowing a sort of spatial awareness. Furthermore, the majority of traditional computer vision techniques are based on the discrete convolution of images, and on the features extracted from them.

As a matter of principle a NN with just two successive layers, which is called a *shallow* network, and with an arbitrary number of neurons per layer, can approximate arbitrary well any kind of smooth enough function [33]. However, direct experience suggests that networks with multiple layers, called *deep* networks, can reach equivalent results exploiting a lower number of parameters overall. This is the reason why this discipline goes under the name of *deep* learning: it focuses on deep networks with up

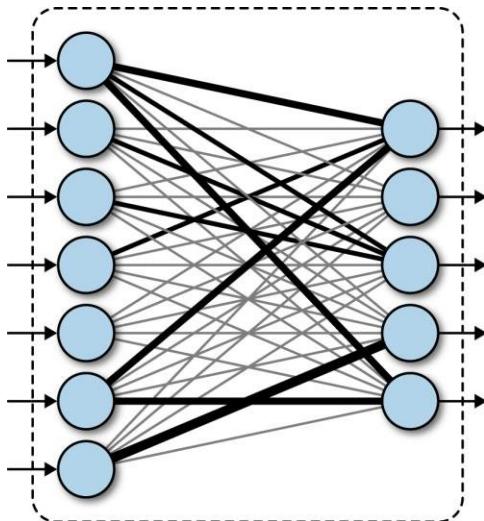


Figure 1.6: Schematic representation of a fully connected (or dense) layer. Every neuron from the first layer is connected with every output neuron. The link thickness represent the absolute value of the combination weight for that particular value.

494 to tens of hidden layers. Such deep structures allow the computation of what is called
 495 deep features, so features of the features of the input data, that allows the network to
 496 easily manage concepts that would be barely understandable for humans.

497 1.2.2 Training ~~of a NN~~ Error Back-Propagation Strategies in 498 Deep Learning

499 Depending on the task the NN is designed for, it will have a different architecture and
 500 number of parameters. Those parameters are initialized to completely random values,
 501 tough. The training process is exactly the process of seeking iteratively the right values
 502 to assign to each parameter in the network in order to accomplish the task. The best
 503 start to understanding the training procedure is to look at how a supervised problem is
 504 solved. In supervised problems, we start with a series of examples of true connections
 505 between inputs and correspondent outputs and we try to generalize the rule behind those
 506 examples. After the rule has been picked up the final aim is to exploit it and to apply
 507 it to unknown data, so the new problem could be solved. In opposition to the concept
 508 of supervised problems, there are the *unsupervised* problems, where the algorithm does
 509 not try to learn a rule from a practical example but try to devise it from scratch. A task
 510 typically posed as unsupervised is clustering, when different data are separated in groups
 511 based on the values of their features in the feature space. Usually, only the number of
 512 groups is taken in input from the algorithm, and the subdivision is completely performed

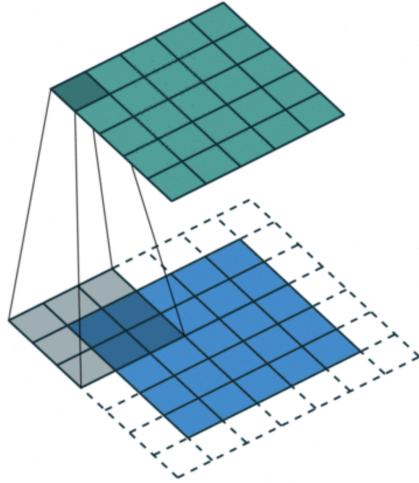


Figure 1.7: Schematic representation of a convolutional layer. The input data are processed by a window kernel that slides all over the image. This operation can recreate almost all the traditional computer vision techniques, and can overcome them, creating new operations, which would be unthinkable to hand-engineered.

513 by the machine. In the real world, by the way, there are many different and creative
 514 shapes between pure supervised and pure unsupervised learning, based on the actual
 515 availability of data and specific limitations to the individual task.

516 An interesting mention in this ~~regards~~-regard should be made about *semi-supervised*
 517 learning, which is typically used in ~~bio-medical~~-biomedical applications. This learn-
 518 ing technique combines a large quantity of unlabeled data during training with a lim-
 519 ited number of pre-labeled example. The blend between data can produce a consid-
 520 erable improvement in learning accuracy ~~-~~and leading to better results with respect
 521 to pure unsupervised techniques. The typical situation of usage of this technique is
 522 when the acquisition of labeled data requires a highly trained human agent (as an
 523 ~~anatomo-pathologist~~anatomopathologist) or a complex physical experiment. The actual
 524 cost of building entire and suitable fully labeled training sets in these situations would
 525 be unbearable, and semi-supervised ~~learnign~~-learning comes in great practical help.

526 Another important training technique which worth mentioning is the so-called *cur-
 527 riculum learning* [5]. This is a learning technique inspired by the typical learning curve
 528 human being and animal, that are used to face problems of always increasing difficulty
 529 while learning something new or a new skill. The example presented to the Neural Net-
 530 work are not randomly presented but organized in a meaningful order which illustrates
 531 gradually more concepts, and gradually more complex ones. The experiments show that
 532 through curriculum learning significant improvements in generalization can be achieved.

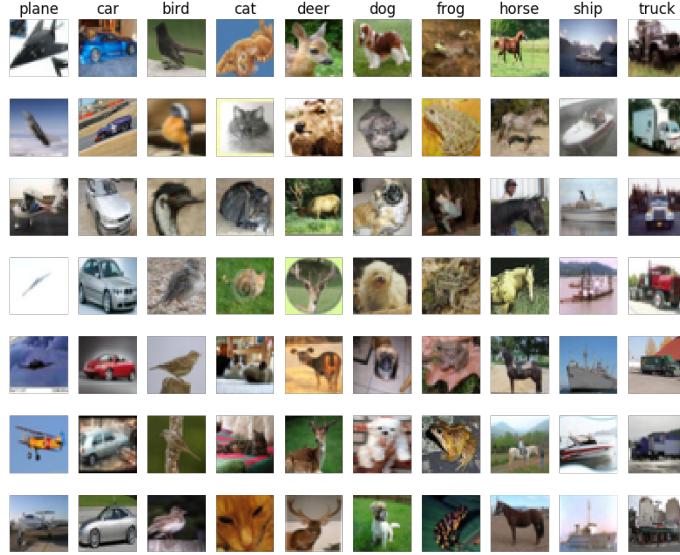


Figure 1.8: Sample grid of images from the CIFAR10 dataset. Each one of the 32×32 image is labeled with one of the ten classes of objects: plane, car, bird, cat, deer, dog, frog, horse, ship, truck.

533 This approach has both an effect on the speed of convergence of the training process to a
 534 minimum and, in the case of non-convex criteria, on the quality of the local minima ob-
 535 tained. This technique is of particular interest for this work: the generated images, which
 536 will be shown in section 3.3 , will offer a segmentation task much less complex respect to
 537 the analysis of real histological tissue. In the optical of training a DL-based model the
 538 abundantly produced images can be used for the preliminary phase of training, setting
 539 aside the more complex and more valuable hand-labeled images for the finalization of
 540 the training process.

541 1.2.3 Training Algorithms - Error Back-Propagation

542 A good example of supervised problems tough is the classification of images. Let's
 543 assume we have a whole dataset of pictures of different objects (as cats, dogs, cars, etc.)
 544 like the CIFAR10 [23] dataset. This famous dataset is made of over 60K labeled colored
 545 images 32×32 divided into 10 categories of objects as shown in Figure 1.8. We could be
 546 interested in the creation of a NN able to assign at every image its belonging class. This
 547 NN could be arbitrarily complex but it certainly will take as input a $32 \times 32 \times 3$ RGB
 548 image and the output will be the predicted class. A typical output for this problem
 549 would be a probability distribution over all the 10 classes like:

$$\vec{p} = (p_1, p_2, \dots, p_{10}), \quad (1.2)$$

$$\sum_{i=1}^{10} p_i = 1, \quad (1.3)$$

and it should be compared with the true label, that is represented just as a binary sequence \vec{t} with the bit correspondent to the belonging class set as 1, and all the others value set to 0:

$$\vec{t} = (0, 0, \dots, 1, \dots, 0, 0). \quad (1.4)$$

Every time an image is given to the model an estimate of the output is produced. Thus, we need to measure the *distance* between that prediction and the true value, to quantify the error made by the algorithm and try to improve the model's predictive power. The functions used for this purpose are called loss functions. The most common choice is the Mean Squared Error (MSE) function that is simply the averaged L^2 norm of the difference vector between \vec{p} and \vec{t} :

$$MSE = \frac{1}{n} \sum_{i=0}^n (t_i - p_i)^2. \quad (1.5)$$

Let's say the NN under training has L consecutive layers, each one with its activation function f^k and its weights vector \vec{w}^k , hence the prediction vector \vec{p} could be seen as the result of the consecutive, nested, application through all the layers:

$$\vec{p} = f^L(\vec{w}^L \cdot (f^{L-1}(\vec{w}^{L-1} \cdot \dots \cdot f^1(\vec{w}^1 \cdot \vec{x}))). \quad (1.6)$$

From both equations 1.5 and 1.6 it is clear that the loss function could be seen as a function of all the weights vectors of every layer of the network. So if we want to reduce the distance between the NN prediction and the true value we need to modify those weights to minimize the loss function. The most established algorithm to do so for a supervised task in a feed-forward network is the so-called *error back-propagation*.

The back-propagation method is an iterative technique that works essentially computing the gradient of the loss function with respect to the weights using the derivative chain rule and updating by a small amount the value of each parameter to lower the overall loss function at each step. Each weight is *moved* counter-gradient, and summing all the contribution to every parameter the loss function approaches its minimum. In equation 1.7 is represented the variation applied to the j^{th} weight in the i^{th} layer in a single step of the method:

$$\Delta w_{ij} = -\eta \frac{\partial E}{\partial w_{ij}}, \quad (1.7)$$

574 where E is the error function, and η is the *learning coefficient*, that modulate the effect of
575 learning through all the training process. This iterative procedure is applied completely
576 to each image in the training set several times, each time the whole dataset is reprocessed
577 is called an *epoch*. The great majority of the dataset is exploited in the training phase
578 to keep running this trial and error process and just a small portion is left out (typically
579 10% of the data) for a final performance test.

580 The loss function shall inevitably be differentiable, and its behavior heavily influences
581 the success of the training. If the loss function presents a gradient landscape rich of
582 local minima the gradient descent process would probably get stuck in one of them.
583 More sophisticated algorithms capable of avoiding this issue have been devised, with
584 the insertion of some degree of randomness in them, as the Stochastic Gradient Descent
585 algorithm, or the wide used *Adam* optimizer [22].

586 While Error-Back Propagation is the most established standard in DL applications,
587 it suffers from some problems. The most common one is the so-called vanishing or ex-
588 ploding gradient issue, which is due to the iterative chain derivation through all the
589 nested level of composition of the function. ~~Without~~ Without a careful choice of the
590 right activation function and the tuning of the learning hyper-parameters, it is very easy
591 to bump into this pitfall. Furthermore, the heavy use of derivation rises the inability to
592 handle non-differentiable components and hinders the possibility of parallel computation.
593 However, there are many alternative approaches to network learning ~~beside~~ besides EBP.
594 The Minimization with Auxiliary Variables (MAV) method builds upon previously pro-
595 posed methods that break the nested objective into easier-to-solve local subproblems via
596 inserting auxiliary variables corresponding to activations in each layer. Such a method
597 avoids gradient chain computation and the potential issues associated with it [10]. A fur-
598 ther alternative approach to train the network is the Local Error Signals (LES), which is
599 based on layer-wise loss functions. In [30], is shown that layer-wise training can approach
600 the state-of-the-art on a variety of image datasets. It is used a single-layer sub-networks
601 and two different supervised loss functions to generate local error signals for the hidden
602 layers, and it is shown that the combination of these losses helps with optimization in
603 the context of local learning.

604 The training phase is the pulsing heart of a DL model development and it could
605 take even weeks on top-level computers for the most complicated networks. In fact,
606 one of the great limits to the complexity of a network during the designing phase is
607 exactly the available computational power. There are many more further technical details
608 necessary for proper training, the adjustment of which can heavily impact the quality of
609 the algorithm. However, after the training phase, we need to test the performance of the
610 NN. This is usually done running the trained algorithm on never seen before inputs (the
611 test dataset) and comparing the prediction with the ground-truth value. A good way to
612 evaluate the quality of the results is to use the same function used as the loss function
613 during the training, but there is no technical restriction to the choice of this quality
614 metric. The average score on the whole test set is then used as a numerical score for

615 the network, and it allows straightforward comparison with other models' performances,
616 trained for the same task. All this training procedure is coherently customized to every
617 different application, depending on which the problem is posed as supervised or not and
618 depending on the more or less complex network's architecture. The leitmotif is always
619 finding a suitable loss function that quantifies how well the network does what it has
620 been designed to do and trying to minimize it, operating on the parameters that define
621 the network structure.



Figure 1.9: Example of the resulting segmentation mask of an image of an urban landscape. Every interesting object of the image is detected and a solid color region replaces it in the segmentation mask. Every color corresponds to a different class of objects, for example, persons are highlighted in magenta and scooters in blue. The shape and the boundaries of every region should match as precisely as possible the edges of the objects.

622 1.3 Deep Learning-Based Segmentation Algorithms

623 In digital image processing, image segmentation is the process of recognizing and sub-
 624 dividing an image into different regions of pixels that show similar features, like color,
 625 texture, or intensity. Typically, the task of segmentation is to recognize the edges and
 626 boundaries of the different objects in the image and assigning a different label to every
 627 detected region. The result of the segmentation process is an image with the same dimen-
 628 sions of the starting one made of solid color regions, representing the detected objects.
 629 This image is called *segmentation mask*. In Figure 1.9 is shown an example of segmenta-
 630 tion of a picture of an urban landscape: different colors are linked to different classes of
 631 objects like persons in magenta and scooters in blue. This technology has a significant
 632 role in a wide variety of application fields such as scene understanding, medical image
 633 analysis, augmented reality, etc.

634 A relatively easy segmentation problem, and one of the first to be tackled, could
 635 be distinguishing an object from the background in a grey-scale image, like in Figure
 636 1.10. The easiest technique to perform segmentation in this kind of problem is based on
 637 thresholding. Thresholding is a binarization technique based on the image's grey-level
 638 histogram: to every pixel with luminosity above that threshold is assigned the color
 639 *white*, and vice versa the color *black*. However, this is a very primitive and fallacious, yet
 640 very fast method, and it manages poorly complex images or images with un-uniformity
 641 in the background.

642 A lot of other traditional techniques improve this first segmentation method [11].
 643 Some are based on the object's edges recognition, exploiting the sharp change in lumi-
 644 nosity typically in correspondence of the boundary of a shape. Other techniques exploit
 645 instead a region-growing technology, according to which some *seed* region markers are



Figure 1.10: Example of the resulting segmentation mask of an image of a fingerprint obtained through a thresholding algorithm. The result is not extremely good, but this technique is very easy to implement and runs very quickly.

scattered on the image, and the regions corresponding to the objects in the image are grown to incorporate adjacent pixels with similar properties.

Every development of traditional computer vision or of Machine Learning-based segmentation algorithm suffers from the same, inevitable limitation. For every one of those techniques is the designing phase in which the operator should decide precisely which features to extract from the image, like different directional derivatives in the image plane or image entropy, and how to process them for the rest of the analysis. There is thus an intrinsic limitation in the human comprehension of those quantities and in the possible way to combine them. The choice is made on the previous experimental results in other image processing works, and on their theoretical interpretation. Neural Networks instead are relieved from this limitation, allowing themselves to learn which features are best suited for the task and how they should be processed during the training phase. The complexity is then moved on to the design of the DL model and on its learning phase rather than on the hand-engineering design of the features to extract. In Figure 1.11 an example high-level feature extracted from a DL model trained for the segmentation of nuclei in a histological sample. The model learns the typical pattern of arrangement of nuclei, which would have been impossible to describe equally in advance.

1.3.1 State of the Art on Deep Learning Segmentation

Similarly to many other traditional tasks, also for segmentation, there has been a thriving development lead by the diffusion of deep learning, that boosted the performances resulting in what many regards as a paradigm shift in the field [28].

In further detail, image segmentation can be formulated as a classification problem of pixels with semantic labels (semantic segmentation) or partitioning of individual objects (instance segmentation). Semantic segmentation performs pixel-level labeling with a set of object categories (e.g. boat, car, person, tree) for all the pixels in the image, hence it

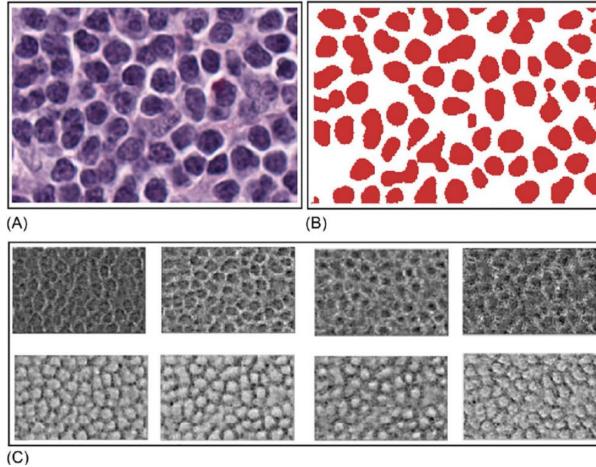


Figure 1.11: (A) An extract from an histological samples, used as input image for the model. (B) The exact segmentation mask. (C) Example of accentuated features during the training: (1-4) for back-ground recognition, (5-8) for nuclei detection. From [2].

is typically a harder task than image classification, which requires just a single label for the whole image. Instance segmentation extends semantic segmentation scope further by detecting and delineating each object of interest in the image (e.g. partitioning of individual nuclei in a histological image).

There are many prominent Neural Network architectures used in the computer vision community nowadays, based on very different ideas such as convolution, recursion, dimensionality reduction, and image generation. This section will provide an overview of the state of the art of this technology and will dwell briefly on the details behind some of those innovative architectures.

680 Recurrent Neural Networks (RNNs) and the LSTM

681 The typical application for RNN is processing sequential data, as written text,
 682 speech or video clips, or any other kind of time-series signal. In this kind of
 683 data, there is a strong dependency between values at a given time/position and
 684 values previously processed. Those models try to implement the concept of *memory*
 685 weaving connections, outside the main information flow of the network, with the
 686 previous NN's input. At each time-stamp, the model collects the input from the
 687 current time X_i and the hidden state from the previous step h_{i-1} and outputs a
 688 target value and a new hidden state (Figure 1.12). Typically RNN cannot manage
 689 easily long-term dependencies in long sequences of signals. There is no theoretical
 690 limitation in this direction, but often it arises vanishing (or exploding) gradient
 691 problematics during the training phase. A specific type of RNN has been designed
 692 to avoid this situation, the so-called Long Short Term Memory (LSTM) [19]. The

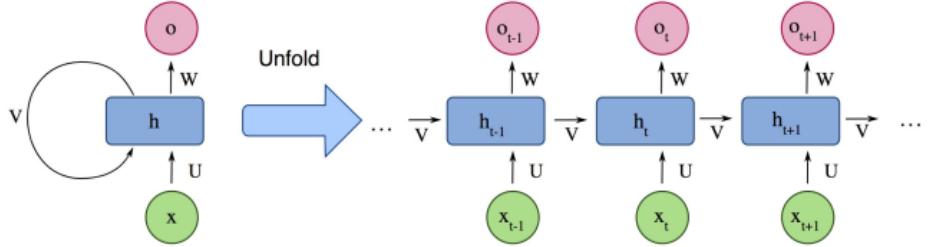


Figure 1.12: Example of the structure of a simple Recurrent Neural Network from [28].

693 LSTM architecture includes three gates (input gate, output gate, forget gate),
 694 which regulate the flow of information into and out from a memory cell, which
 695 stores values over arbitrary time intervals.

696 Encoder-Decoder and Auto-Encoder Models

697 Encoder-Decoder models try to learn the relation between an input and the corre-
 698 sponding output with a two steps process. The first step is the so-called *encoding*
 699 process, in which the input x is compressed in what is called the *latent-space* rep-
 700 resentation $z = f(x)$. The second step is the *decoding* process, where the NN
 701 predicts the output starting from the latent-space representation $y = g(z)$. The
 702 idea underneath this approach is to capture in the latent-space representation the
 703 underlying semantic information of the input that is useful for predicting the out-
 704 put. ED models are widely used in image-to-image problems (where both input
 705 and output are images) and for sequential-data processing (like Natural Language
 706 Processing, NLP). In Figure 1.13 is shown a schematic representation of this ar-
 707 chitecture. Usually, these model follow a supervised training, trying to reduce the
 708 reconstruction loss between the predicted output and the ground-truth output pro-
 709 vided while training. Typical applications for this technology are image-enhancing
 710 techniques like de-noising or super-resolution, where the output image is an im-
 711 proved version of the input image. Or image generation problems (e.g. plausible
 712 new human faces generation) in which all the properties which define the type of
 713 image under analysis should be learned in the representation latent space.

Generative Adversarial Networks (GANs)

The peculiarity of Generative Adversarial Network (GAN) lies in its structure. It is actually made of two distinct and independent modules: a generator and a discriminator, as shown in Figure 1.14. The first module G , responsible for the generation, typically learns to map a prior random distribution of input z to a target distribution y , as similar as possible to the target $G = z \rightarrow y$ (i.e. almost

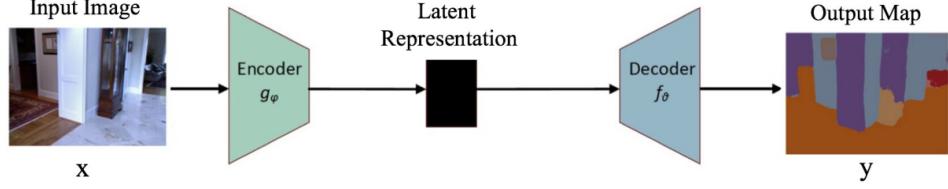


Figure 1.13: Example of the structure of a simple Encoder-Decoder Neural Network from [28].

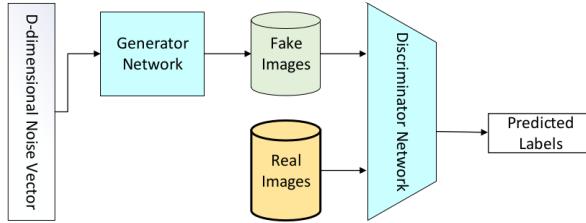


Figure 1.14: Schematical representation of a Generative Adversarial Networks, form [28].

any kind of image-to-image problem could be addressed with GANs, as in [21]). The second module, the discriminator D , instead is trained to distinguish between *real* and *fake* images of the target category. These two networks are trained alternately in the same training process. The generator tries to fool the discriminator and vice versa. The name adversarial is actually due to this *competition* within different parts of the network. The formal manner to set up this adversarial training lies in the accurate choice of a suitable loss function, that will look like:

$$L_{GAN} = \mathbb{E}_{x \sim p_{data}(x)}[\log D(x)] + \mathbb{E}_{z \sim p_z(z)}[\log(1 - D(G(z)))]$$

. The GAN is thus based on a min-max game between G and D . D aims to reduce the classification error in distinguishing fake samples from real ones, and as a consequence maximizing the L_{GAN} . On the other hand, G wants to maximize the D 's error, hence minimizing L_{GAN} . The result of the training process is the trained generator G^* , capable of produce an arbitrary number of new data (images, text, or whatever else):

$$G^* = \arg \min_G \max_D L_{GAN}$$

714 . This peculiar architecture has yielded several interesting results and it has been
 715 developed in many different directions, with influences and contaminations with
 716 other architectures [21].

717 Convolutional Neural Networks (CNNs)

718 As stated before CNNs are a staple choice in image processing DL applications.
719 They mainly consist of three types of layers:

- 720 i convolutional layers, where a kernel window of parameters is convolved with
721 the image pixels and produce numerical features maps.
- 722 ii nonlinear layers, which apply an activation function on feature maps (usually
723 element-wise). This step allows the network to introduce non-linear behavior
724 and then increasing its modeling capabilities.
- 725 iii pooling layers, which replace a small neighborhood of a feature map with some
726 statistical information (mean, max, etc.) about the neighborhood and reduce
727 the spatial resolution.

728 Given the arrangement of successive layers, each unit receives weighted inputs from
729 a small neighborhood, known as the receptive field, of units in the previous layer.
730 The stack of layers allows the NN to perceive different resolutions: the higher-
731 level layers learn features from increasingly wider receptive fields. The leading
732 computational advantage given by CNN architecture lies in the sharing of kernels'
733 weights within a convolutional layer. The result is a significantly smaller number
734 of parameters than fully-connected neural networks. In section 2.7 will be shown a
735 particular application of this architecture, known as *style-transfer* network, which
736 is a particular algorithm capable of implanting the visual texture of a *style* image
737 onto the content of a different image, producing interesting hybrid images. Some
738 of the most notorious CNN architectures include: AlexNet [24], VGGNet [41], and
739 U-Net [35].

740 For this work, U-net architecture is particularly interesting. The U-net model was
741 initially developed for biomedical image segmentation, and in its structure reflects char-
742 acteristics of both CNN and Encoder-Decoder models. Ronneberger et al.[35] proposed
743 this model for segmenting biological microscopy images in 2015. The U-Net architecture
744 is made of two branches, a contracting path to capture context, and a symmetric expand-
745 ing path (see Figure 1.15). The down-sampling flow is made of a Fully Convolutional
746 Network (FCN)-like architecture that computes features with 3×3 kernel convolutions.
747 On the other hand, the up-sampling branch exploits up-convolution operations (or de-
748 convolution), reducing the number of feature maps while increasing their dimensions.
749 Another characteristic of this architecture is the presence of direct connections between
750 layers of a similar level of compression in compressing and decompressing branches.
751 Those links allow the NN to preserve spatial and pattern information. The Network
752 flow eventually ends with a 1×1 convolution layer responsible for the generation of the
753 segmentation mask of the input image.

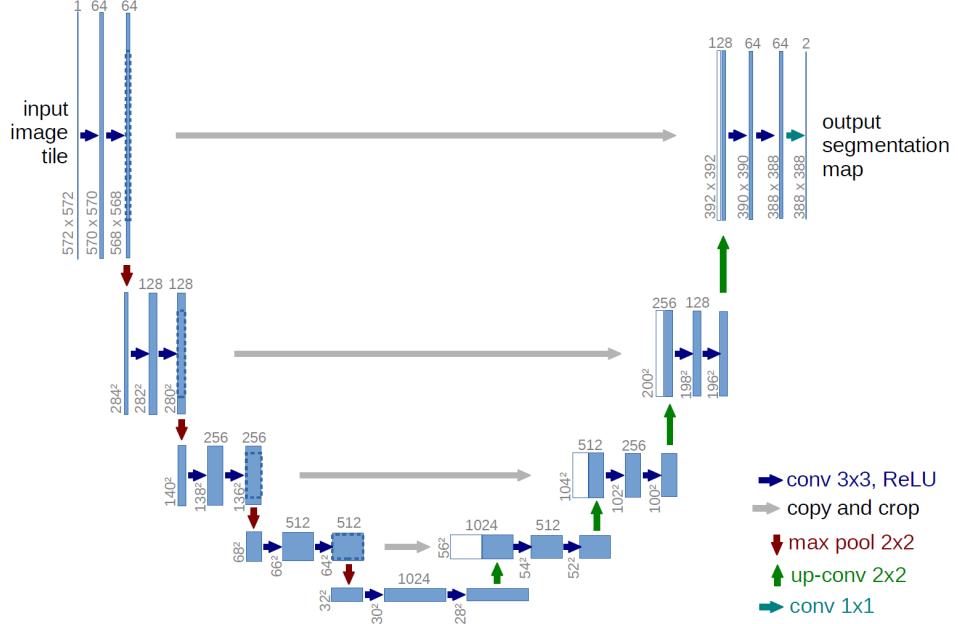


Figure 1.15: Scheme of the typical architecture of a U-net NN. This particular model was firstly proposed by Ronneberger et al. in [35].

A recent example of a practical application of a CNN to histological images could be found in [29]. In this work the Inception v3 is trained on hand-labeled samples of pancreatic tissue (like in Figure 1) to recognise recognize tumoral regions from healthy healthy ones in a pancreatic tissue specimen treated with Ki67 staining. The Inception v3 [34] network is a deep convolutional network developed by Google, trained for object detection and image classification classification on the ImageNet dataset [15]. Recognition of the tumoral region of Ki67 stained pancreatic tissue samples is based on the detection and counting of some specific marker cells. In Figure 1.16 is shown a pair of the original image and the computed segmentation mask, which label in red tumoral regions and in green the healthy ones. This work is based on a technique called transfer learning, which consists in-of the customization and specialization of a pre-trained NN previously trained for similar, but essentially different, tasks. The final part of the training of this version of Inception v3 has been performed of a dataset of 33 whole slide images of Ki67 stained neuroendocrine tumor biopsies acquired from 33 different patients, digitized with a 20 \times magnification factor and successively divided in 64 \times 64 patches.

1.3.2 Image Segmentation Datasets

Besides the choice of suitable architecture the most important aspect while developing a NN is the dataset on which perform the training process. Let's confine the discussion

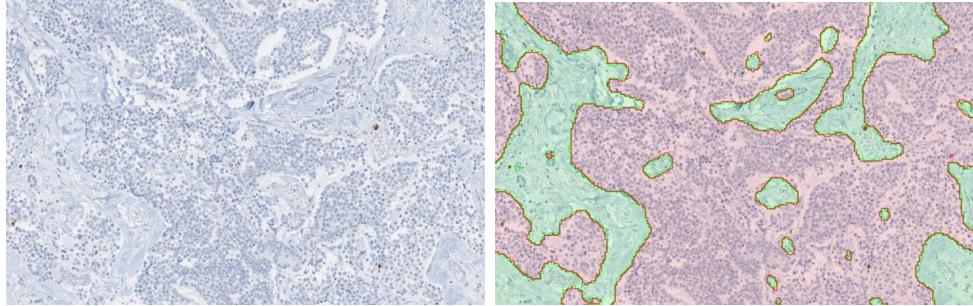


Figure 1.16: (left) Original image of a Ki67 stained pancreatic tissue sample, (right) the corresponding segmentation mask, which label in red tumoral regions and in green the healthy ones. From [29].

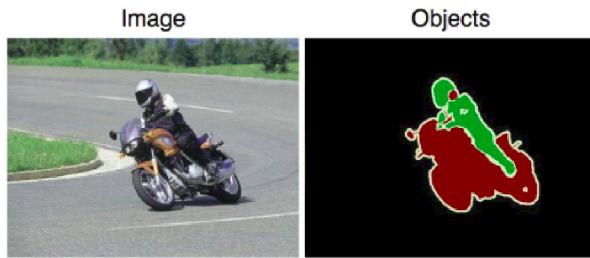


Figure 1.17: An example image from the PASCAL dataset and its corresponding segmentation mask [17].

772 only to image-to-image problems, like segmentation problems. There are a lot of widely
 773 used datasets, but I want to mention just a few of them to give the idea of their typical
 774 characteristics.

775 A good example of segmentation is the Cityscapes dataset [12], which is a large-scale
 776 database with a focus on semantic understanding of urban street scenes. The dataset
 777 is made of video sequences from the point of view of a car in the road traffic, from 50
 778 different cities in the world. The clips are made of 5K frames, labeled with extremely high
 779 quality at pixel-level and an additional set of 20K weakly-annotated frames. Each pixel
 780 in the segmentation mask contains the semantic classification, among over 30 classes of
 781 objects. An example of an image from this dataset is shown in Figure 1.9.

782 The PASCAL Visual Object Classes (VOC) [17] is another of the most popular
 783 datasets in computer vision. This dataset is designed to support the training of algo-
 784 rithms for 5 different tasks: segmentation, classification, detection, person layout, and
 785 action recognition. In particular, for segmentation, there are over 20 classes of labeled
 786 objects (e.g. planes, bus, car, sofa, TV, dogs, person, etc.). The dataset comes divided
 787 into two portions: training and validation, with 1,464 and 1,449 images, respectively. In
 788 Figure 1.17 is shown an example of an image and its corresponding segmentation mask.

789 As ~~last mention~~the last mention, I would report ~~t he~~the ImageNet project [15], which
790 is a large visual database designed for use in visual object recognition software research.
791 It consists of more than 14 million images that have been hand-annotated by the project
792 to indicate what objects are pictured and in at least one million of the images, bounding
793 boxes are also provided. ImageNet contains more than 20,000 categories of objects. Since
794 2010, the ImageNet project runs an annual software contest, the ImageNet Large Scale
795 Visual Recognition Challenge (ILSVRC), where software programs compete to correctly
796 classify and detect objects and scenes. This kind of ~~competitions~~competition is very
797 important for the research field, as it ~~inspire and encourage~~inspires and encourages the
798 development of new models and architectures.

799 It is worth mentioning that in the medical image processing domain typically the
800 available dataset is definitely not that rich and vast (that is actually the seed of this
801 work) and thus many techniques of data augmentation have been devised, to get the
802 best out of the restricted amount of material. Generally, data augmentation manipu-
803 lates the starting material applying a set of transformation to create new material, like
804 rotation, reflection, scaling, cropping and shifting, etc. Data augmentation has been
805 proven to improve the efficacy of the training, making the model less prone to over-
806 fitting, increasing the generalization power of the model, and helping the convergence to
807 a stable solution during the training process.

808 **Chapter 2**

809 **Technical Tools for Model
810 Development**

811 As mentioned in the introduction, this project wants to produce synthetic histological
812 images paired with their corresponding segmentation mask, to train Neural Networks
813 for the automatization of real histological images analysis. The production of artificial
814 images passes through the processing of a ~~three-dimensional~~three-dimensional, virtual
815 model of a histological structure, which is the heart of this thesis work. The detailed
816 description of the development of the two proposed histological models will follow the
817 present chapter and will occupy all the chapter 3. Here I will dwell, instead, on every
818 less common tool employed during the models' designing phase. From the practical
819 point of view, this project is quite articulated and the development has required the
820 harmonization of many different technologies, tools, and code libraries. The current
821 chapter should be seen as a theoretical complement for chapter 3, and its reading is
822 suggested to the reader for any theoretical gap or for any further technical deepening.
823 The reader already familiar with those technical tools should freely jump to the models'
824 description.

825 All the code necessary for the work has been written in a pure **Python** environment,
826 using several already established libraries and writing by ~~my self~~myself the missing code
827 for some specific applications. I decided to code in **Python** given the thriving variety of
828 available libraries geared toward scientific computation, image processing, data analysis,
829 and last but not least for its ease of use (compared to other programming languages).
830 In each one of the following subsections, I will mention the specific code libraries which
831 have been employed in this project for every technical necessity.

832 2.1 Quaternions

833 Quaternions are, in mathematics, a number system that expands to four dimensions the
 834 complex numbers. They have been described for the first time by the famous mathematician
 835 William Rowan Hamilton in 1843. This number system define three independent
 836 *imaginary* units $\mathbf{i}, \mathbf{j}, \mathbf{k}$ as in (2.1), which allows the general representation of a quaternion
 837 \mathbf{q} is (2.2) and its inverse \mathbf{q}^{-1} (2.3) where a, b, c, d are real numbers:

$$i^2 = j^2 = k^2 = ijk = -1, \quad (2.1)$$

$$\mathbf{q} = a + bi + cj + dk, \quad (2.2)$$

$$\mathbf{q}^{-1} = (a + bi + cj + dk)^{-1} = \frac{1}{a^2 + b^2 + c^2 + d^2} (a - bi - cj - dk). \quad (2.3)$$

838 Furthermore, the multiplication operation between quaternionn does not benefit from
 839 commutativity, hence the product between basis elements will behave as follows:

$$i \cdot 1 = 1 \cdot i = i, \quad j \cdot 1 = 1 \cdot j = j, \quad k \cdot 1 = 1 \cdot k = k \quad (2.4)$$

$$i \cdot j = k, \quad j \cdot i = -k$$

$$k \cdot i = j, \quad i \cdot k = -j$$

$$j \cdot k = i, \quad k \cdot j = -i.$$

840 This number system has plenty of peculiar properties and applications, but for this
 841 project, quaternions are important for their ability to represent, in a very convenient way,
 842 rotations in three dimensions. The particular subset of quaternions with vanishing real
 843 part ($a = 0$) has a useful, yet redundant, correspondence with the group of rotations in
 844 tridimensional space $\text{SO}(3)$. Every 3D rotation of an object can be represented by a 3D
 845 vector \vec{u} : the vector's direction indicates the axis of rotation and the vector magnitude $|\vec{u}|$
 846 express the angular extent of rotation. However, the matrix operation which expresses
 847 the rotation around an arbitrary vector \vec{u} it is quite complex and does not scale easily
 848 for multiple rotations [7], which brings to very heavy and entangled computations.

849 Using quaternions for expressing rotations in space, instead, it is very convinient.
 850 Given the unit rotation vector \vec{u} and the rotation angle θ , the corresponding rotation
 851 quaternion \mathbf{q} becomes (2.6):

$$\vec{u} = (u_x, u_y, u_z) = u_x \mathbf{i} + u_y \mathbf{j} + u_z \mathbf{k}, \quad (2.5)$$

$$\mathbf{q} = e^{\frac{\theta}{2}(u_x \mathbf{i} + u_y \mathbf{j} + u_z \mathbf{k})} = \cos \frac{\theta}{2} + (u_x \mathbf{i} + u_y \mathbf{j} + u_z \mathbf{k}) \sin \frac{\theta}{2}, \quad (2.6)$$

$$\mathbf{q}^{-1} = \cos \frac{\theta}{2} - (u_x \mathbf{i} + u_y \mathbf{j} + u_z \mathbf{k}) \sin \frac{\theta}{2}, \quad (2.7)$$

852 where in (2.6) we can clearly see a generalization of the Euler's formula for the
 853 exponential notation of complex numbers, which hold for quaternions. It can be shown
 854 that the application of the rotation represented by \mathbf{q} on an arbitrary 3D vector \vec{v} should
 855 be easily expressed as:

$$\vec{v}' = \mathbf{q}\vec{v}\mathbf{q}^{-1}, \quad (2.8)$$

856 using the Hamilton product defined on quaternions (2.4). This rule raises a very con-
 857 vinient and an extremely scalable way to compute consecutive rotations in space. Given
 858 two independent and consecutive rotations represented by the two quaternions \mathbf{q} and \mathbf{p}
 859 applied on the vector \vec{v} the resulting rotated vector \vec{v}' is simply yielded as:

$$\vec{v}' = \mathbf{p}(\mathbf{q}\vec{v}\mathbf{q}^{-1})\mathbf{p}^{-1} = (\mathbf{pq})\vec{v}(\mathbf{qp})^{-1}, \quad (2.9)$$

860 which essentially is the application of the rotation $\mathbf{r} = \mathbf{qp}$ on the vector \vec{v} . This repre-
 861 sentation is completely coherent with the algebra of 3D rotations, which does not benefit
 862 from commutativity in turn.

863 Given this property, quaternions are indeed widely used in all sorts of applications
 864 of digital 3D space design, as for simulations and videogame. The position of an object
 865 in the space in simulations is generally given by the application of several independent
 866 rotations, typically in the order of a tenth of rotations, which with quaternions is given
 867 easily by the product of simple objects. Every other alternative method would imply the
 868 use of matrix representation of rotations or other rotation systems as Euler's angles and
 869 would eventually make the computation prohibitive.

870 The use of quaternions in this work will be justified in section 3.1, while speaking of
 871 parametric L-systems in 3D space, used to build the backbone of the ramified structure
 872 of blood vessels in the reconstruction of a sample of pancreatic tissue.

873 I was able to find many Python libraries for computation with quaternions, but the
 874 one I appreciated the most for its interface and ease of use was the `pyquaternion`. With
 875 this library, it's immediate the definition of a quaternion by its correspondent rotation
 876 vector, and the multiplication between quaternions is straightforward.

877 2.2 Parametric L-Systems

878 Lindenmayer systems, or simply L-systems, were conceived as a mathematical theory of
 879 plant development [26] in 1968 by Aristid Lindenmayer. Successively, a lot of geometri-
 880 cal interpretations of L-systems were proposed to make them a versatile instrument for
 881 modeling the morphology typical of plants and other organic structures. As a biologist,
 882 Lindenmayer studied different species of yeast and fungi and worked the growth patterns
 883 of various types of bacteria (e.g. as the *cyanobacteria Anabaena catenula*). The main

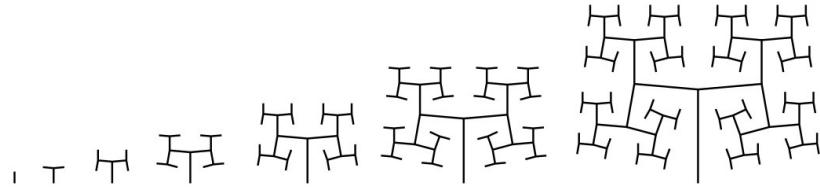


Figure 2.1: Growth pattern for the space-filling fractal-like system, used to mimic the blood vessel bifurcations in sec 3.1.

purpose for which L-systems were devised was to allow a formal description of the development of simple multicellular living organisms. Subsequently, the potentiality of these systems was expanded to describe higher-order plants and complex branching structures.

An L-system is in general defined by an *axiom* sequence and some development *rules*, which are recursively applied to the sequence and lead its development. The original proposed L-system was fairly simple and shows really well the idea underneath:

$$\begin{aligned} \textit{axiom} &: A \\ \textit{rules} &: (A \rightarrow AB), \quad (B \rightarrow A) \end{aligned}$$

where A and B could be any two different patterns in the morphology of an algae, or could be different bifurcations in a ramified structure. The iterative application of the rules to the axiom sequence, let's say for 7 times, will produce the following sequence:

$$\begin{aligned} n = 0 & : A \\ n = 1 & : AB \\ n = 2 & : ABA \\ n = 3 & : ABAAB \\ n = 4 & : ABAABABA \\ n = 5 & : ABAABABAABAAB \\ n = 6 & : ABAABABAABAABABAABABA \\ n = 7 & : ABAABABAABAABABAABAABABAABABAAB . \end{aligned}$$

This kind of tool, as will be shown also in 3.1, is particularly suited for the creation of structures with fractal behavior, and it has been used in this work to create the backbone of the entangled bifurcation in blood vessels in the modelization of pancreatic tissue. In particular, there was the need for a fractal-like space-filling ramification as the one shown in Figures 2.1.

898 The system in Figure 2.1 represent the successive ramification of a structure which
 899 grows adding segments gradually shorter, by a lenght ratio parameter R and inclined
 900 of $\delta = \pm 85^\circ$ respect the previous branch. The axiom and the rules that produce this
 901 structure are the following:

$$\begin{aligned} & \text{axiom : } A & (2.10) \\ & \text{rule}_1 : A \rightarrow F(1)[+A][-A] \\ & \text{rule}_2 : F(s) \rightarrow F(s \cdot R) \end{aligned}$$

902 where A represent the start of a new branch and $F(s)$ represent a branch of lenght s .
 903 The presence of a rule which acts differently depending on the target object, is an further
 904 sophistication respect to the standard L-system. For this reason these systems are called
 905 *parametric* L-systems.

906 The use of standard L-systems turned out to be widespread, and there were a lot of
 907 different Python libraries at my disposal for coding. By the way, parametric L-systems
 908 were not just as popular, and I was not able to find a reliable library on which to build
 909 my work. I decided then to code a parametric branching system able to recreate the
 910 structure with rules (2.10) at any desired level of iteration. Having created the tool I
 911 needed on my own I was able to add all the optional features I would have needed during
 912 the development, like an adjustable degree of angular noise in the branch generation.

913 2.3 Voronoi Tassellation

914 Voronoi diagrams, or Voronoi decompositions, are space-partitioning systems, which
 915 divides an n -dimensional Euclidian space into sub-regions depending on the proximity
 916 to a given set of objects. More precisely, given an n -dimensional space and m starting
 917 point p_1, \dots, p_m inside it, the whole space will be subdivided in m adjacent regions. Every
 918 point of the space is assigned to the region correspondent to the nearest starting point.
 919 In Figure 2.2 is shown a practical example of a Voronoi decomposition of a plane into
 920 20 regions corresponding to the 20 starting points. Informal use of Voronoi diagrams
 921 can be traced back to Descartes in 1644, and many other mathematicians after him.
 922 But, Voronoi diagrams are named after Georgy Feodosievych Voronoy who defined and
 923 studied the general n-dimensional case in 1908 [47].

924 More precisely, let X be a metric space and d the distance defined on it. Let K be
 925 the set of indices and let $(P_k)_{k \in K}$ be the tuple of sites in the space X . The k^{th} Voronoi
 926 cell R_k , associated with the site P_k is the set of all the points in X whose distance to P_k
 927 is smaller than the distance to any other site P_j , with $j \neq k$, or in other words:

$$R_k = \{x \in X \mid d(x, P_k) \leq d(x, P_j) \forall j \in K, j \neq k\}, \quad (2.11)$$

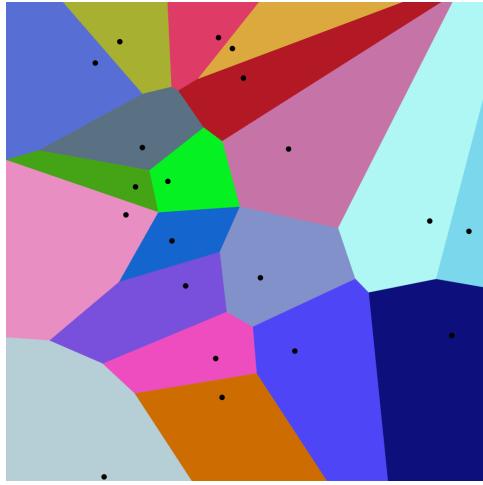


Figure 2.2: Example of a Voronoi decomposition of a plane into 20 regions corresponding to 20 starting points.

depending on the notion of distance defined on the space X the final redistribution in subregions will look very differently.

In addition to the choice of the distance function, another fundamental factor is the distribution of sites in the space to be divided. If the points are chosen equally and homogeneously distributed the final distribution will appear as a simple regular lattice, while a completely random distribution of points in the space will provide a decomposition in cells with very different shapes and volumes, as shown in Figure 2.3. Interesting results concerning points from a semi-random distribution will be shown in section 3.1, which leads to decomposition with a good richness in shapes but with the desired homogeneity in volumes.

The Voronoi decomposition has been of great interest in this project for the division of a 3D space in subregions, to recreate the spatial distribution of cells in a sample of human tissue, as will be shown in section 3.1. The formal definition of Voronoi regions (2.11) ensures the convexity of each decomposition's tassel, which in three-dimensional space would be adjacent convex polyhedrons. Every tassel of the decomposition will be represented by a bounded 3-dimensional convex hull¹, with except for those most external cells which are unbounded and requires special attention while using.

The most widespread tool for the computation of Voronoi decompositions in Python is contained in the `spatial` submodule of the famous library `SciPy` [46], which is a staple tool for an incredible variety of scientific algorithms. The `Voronoi` object from `Scipy` library offers a very efficient algorithm for space-partitioning, and it has been one of the pillars for the modelization of tissues. Unluckily this module does not easily allow to perform Voronoi decomposition with different definitions of distance functions d other

¹See section 2.5 for further details.

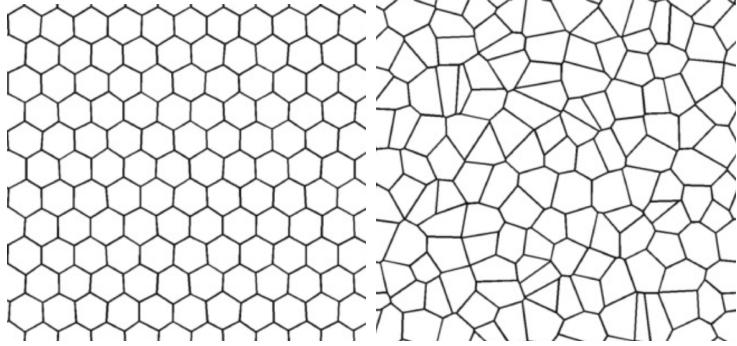


Figure 2.3: On the right an example of 2D Voronoi decomposition resulting from homogeneously distributed points in the plane. On the left the resulting decomposition obtained from randomly distributed points in the plane, from [3].

951 than the Euclidian distance, which would have allowed interesting studies.

952 2.4 Saltelli Algorithm - Random Number Generation

953 As mentioned in section 2.3, in this project there was the need for quasi-random number
 954 generation for the production of Voronoi tessellations. Quasi-random sequences (or low-
 955 discrepancy sequences) are patterns of numbers that emulate the behavior of uniform
 956 random distributions but have a more homogeneous and quick coverage of the sampling
 957 domain, which provides an important advantage in applications as in quasi-Monte Carlo
 958 integration techniques, as shown in Figure 2.4. In computer science there is not any
 959 possibility of recreating *true* random sequences, hence any stochasticity is completely
 960 deterministic in its essence even if produced by very chaotic processes². Indeed, every
 961 algorithm for random number generation is completely repeatable given its starting sta-
 962 tus. Quasi-random sequences are completely deterministic too, ~~but implements but~~
 963 implement more *regular*, well-behaved algorithms.

964 A first good example to understand the concept of quasi-random generation could be
 965 an additive recurrence, as the following:

$$s_{n+1} = (s_n + \alpha) \bmod 1, \quad (2.12)$$

966 which for every seed element s_0 and real parameter α produced completely different
 967 sequences.

²A chaotic process is a deterministic process which has an extremely sensible dependence on its starting conditions. This property mimics very effectively the behavior of true random processes, which are intrinsically forbidden in computer science.

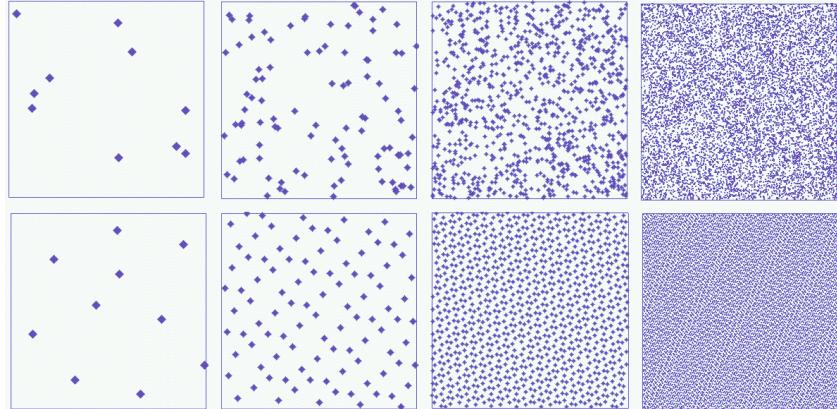


Figure 2.4: Coverage of the unit square with an additive quasirandom numbers sequence as in 2.12 (up) and for uniformly sampled random numbers (bottom). From left to right: 10, 100, 1000, 10000 points.

968 In the bottom line of Figure 2.4 is clearly visible the good and homogeneous coverage
 969 of the sampling domain, although it is strongly visible a regular pattern between points,
 970 which does not convey an *organic* sensation at all. However, increasing the complexity of
 971 our very simple starting model 2.12 it is possible to overcome this *artificial* appearance
 972 of sampled points and to produce very good samples.

973 A notorious algorithm for quasi-random number generation is the Sobol sequence,
 974 introduced by the russian mathematician Ilya M. Sobol in 1967 [44]. In its work, Sobol
 975 wanted to construct a sequence x_n of points in the s -dimensional unitary hypercube
 976 $I^s = [0, 1]^s$ such as for any integrable function f :

$$\lim_{n \rightarrow \infty} \frac{1}{n} \sum_{i=1}^n f(x_i) = \int_{I_s} f. \quad (2.13)$$

977 Sobol wanted to minimize the *holes* in the sampled domain (which it could be shown
 978 to be a property that helps the convergence of the sequence) and minimize as well
 979 the *holes* in every lower-dimension projection of the sampled points. The particularly
 980 good distributions that fulfill those requirements are known as (t, m, s) -nets and (t, s) -
 981 sequences in base b .

982 To better understand them we need first to define the concept of s -interval in base b ,
 983 which is a subset of I_s such as:

$$E_s^b = \prod_{j=1}^s \left[\frac{a_j}{b^{d_j}}, \frac{a_j + 1}{b^{d_j}} \right), \quad (2.14)$$

984 where a_j and d_j are non-negative integers, and $a_j < b^{d_j}$ for all j in $\{1, \dots, s\}$.

985 Let be t and m two integers such as $0 \leq t \leq m$. A (t, m, s) -net in base b is defined
 986 as a sequence x_n of b^m points of I_s such that:

$$\text{Card } \mathbf{P} \cap \{x_1, \dots, x_n\} = b^t \quad (2.15)$$

987 for all the elementary interval \mathbf{P} in base b of hypervolume $\lambda(\mathbf{P}) = b^{t-m}$.

988 Given a non-negative integer t , a (t, s) -sequence in base b is an infinite sequence of
 989 points x_n such that for all integers $k \geq 0$, $m \geq t$ the sequence $\{x_{kb^m}, \dots, x_{(k+1)b^m-1}\}$ is
 990 a (t, m, s) -net in base b .

991 Sobol in his article described in particular (t, m, s) -net and (t, s) -sequence in base 2.
 992 A more thorough description of all the formal properties of those particular sequences
 993 could be found in [43].

994 In order to perform the actual sampling during the modelization, it has been used the
 995 `saltelli` module from the `SALib` library, which performs sampling in an s -dimensional
 996 space following the Saltelli algorithm, which is a specific improved version of the Sobol
 997 algorithms oriented toward the parameter sensitivity analysis [37], [38].

998 2.5 Planar Section of a Polyhedron

999 As will be shown in section 3.1 a fundamental step for the functioning of the modelization
 1000 is the planar section of a three-dimensional polyhedron. It turned out that there is
 1001 no general rule to perform a planar section of a convex polyhedron with an arbitrary
 1002 number of faces, respect to an arbitrary sectioning plane. Hence, I devised an algorithm
 1003 to handle this task. In the case of a full intersection, the result of the sectioning process
 1004 of a polyhedron is a polygonal surface, otherwise, it could be an empty set of points or
 1005 a segment in case of particular tangency, but those two cases are not of interest to the
 1006 model. This tool has been created from scratch by me in the `preliminar-preliminary`
 1007 design phase and also the implemented algorithm has been devised by me and chosen
 1008 as the best option among other possibilities. It is not `the` fruit of an extended and
 1009 thorough formal analysis, hence it has not the `prententious-pretense` to be an extremely
 1010 optimized `algortihm-algorithm`. Nevertheless, this tool passed all the `test-tests` I required
 1011 and yielded the expected results.

1012 Given a convex polyhedron with n vertices and a sectioning plane p , let V be the set
 1013 of all the vertices and $f_p(\vec{x})$ the equation defining the plane. The algorithm is defined
 1014 by the following steps:

1. Divide V in two subsets: A made of those vertices which lie above and B , made of
 those which lie below the sectioning plane. Like in 2.16:

$$A = \{\vec{v} \in V \mid f_p(\vec{v}) \geq 0\} \quad (2.16)$$

$$B = \{\vec{v} \in V \mid f_p(\vec{v}) \leq 0\}$$

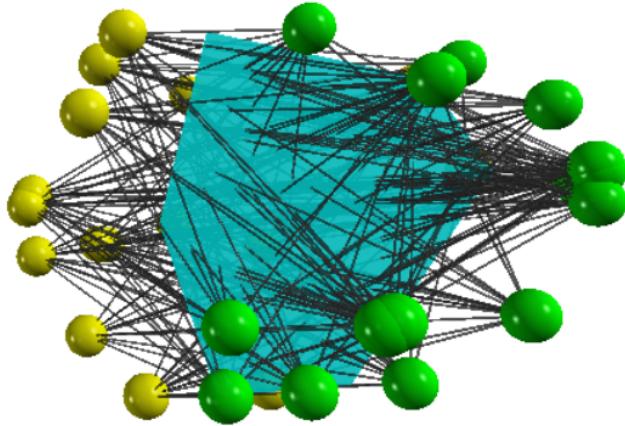


Figure 2.5: In this picture is shown an example of the application of the algorithm for the planar section on a polyhedron. The vertices are divided into two groups, with different colors yellow and green. All the possible lines between any couple of vertices picked from the two classes are drawn in dark gray. In Turquoise the resulting planar section, obtained as the convex hull containing all the intersections between the lines and the plane.

If any of the two subsets turns out to be empty the plane p does not intersect the polyhedron, and the section is empty. A and B are represented in different colors in Figure 2.5.

- 1015 If any of the two subsets turns out to be empty the plane p does not intersect the
- 1016 polyhedron, and the section is empty. A and B are represented in different colors
- 1017 in Figure 2.5.
- 1018 2. Detect, and *draw*, any possible line that crosses two points respectively from A and
- 1019 B . If n_A and n_B are the numbers of points above and below the plane then there
- 1020 will be $n_A \times n_B$ possible lines. In Figure 2.5 all the lines between the two classes
- 1021 of points are drawn in dark gray.
- 1022 3. Detect P , the set of all the points from the intersection between the $n_A \times n_B$ lines
- 1023 from the previous step and the sectioning plane p . All these points will lie on the
- 1024 same plane, within the boundaries of the polygonal section.
- 1025 4. The final polygon is then yielded by computing the convex hull of the points in
- 1026 P . The convexity of the starting polyhedron in fact ensures the convexity of any
- 1027 section of the solid.

The result of the algorithm is, as just stated, a convex hull, which in geometry is defined as the smallest convex envelope or convex closure of a set of points. In 2 dimensions is the smallest convex polygon containing a certain set of points in a plane

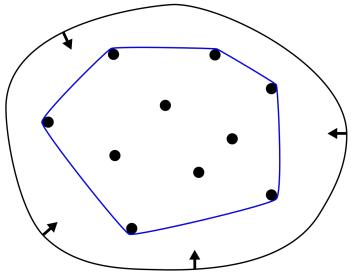


Figure 2.6: Representation of the convex hull of a bounded planar set of points. This particular enclosure goes under the name of "rubber band effect".

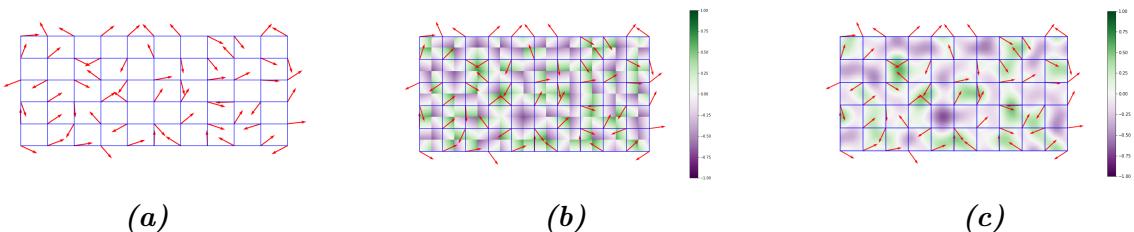


Figure 2.7: The three main steps of the algorithm to produce Perlin noise. In 2.7a the plane discretization and the assignment of a gradient vector to every node of the grid. In 2.7b the computation of the dot product with all the points inside the discretization and in 2.7c the interpolation of the values to create the final function.

1031 (In Figure 2.6 is shown the so-called "rubber band effect"), and in 3 dimensions it is the
 1032 smallest convex polyhedron containing a set of points in the space.

1033 In Python, the most convenient way to work with convex hulls was to use the sub-
 1034 module `spatial.ConvexHull` from the `SciPy` library [46]. This module allows also a
 1035 convenient way for plotting images with `Matplotlib`, which is the point of reference for
 1036 plotting and image formation in Python.

1037 2.6 Perlin Noise

1038 Perlin noise is a widely used form of noise in computer graphics, which mimics very
 1039 well natural and smooth fluctuations around a constant value. It has been developed by
 1040 Ken Perlin in 1983, and it is now the staple tool for giving texture to object in virtual
 1041 modelization, often considered the *salt* of computer graphics texturization. The Perlin
 1042 noise is a gradient-based algorithm defined on grid discretization of a n -dimensional
 1043 space. The algorithm involves three subsequent steps:

- 1044 1. The first step is to discretize the n -dimensional space in a regular lattice: the

1045 dimension of the grid will impact heavily on the scale of the noise. As in Figure
1046 2.7a at every node of the grid is assigned a randomly oriented n -dimensional unitary
1047 gradient vector. This is the preliminary setup which will allow the computation of
1048 the actual noise function in every point of the space.

- 1049 2. Given the candidate point \vec{x} between the grid nodes onto which evaluate the noise
1050 there are 2^n nearest grid nodes. For each one of these 2^n nodes, it is evaluated the
1051 distance vector from \vec{x} as the offset between the two points. Then it is computed
1052 the dot product between every pair made of nearby gradient vectors and the offset
1053 vector. This operation should be thought of as made on every point in the lattice,
1054 as in Figure 2.7b, where at every point of the grid is represented just one of the
1055 $2^2 = 4$ series of dot products.
- 1056 3. The final step is the interpolation between the 2^n series of dot products. To per-
1057 form the interpolation usually is used a function with vanishing first degree (and
1058 preferably also second degree) derivative in correspondence of the 2^n grid nodes ³.
1059 This means that the noise function will pass through zero at every node and have
1060 a gradient equal to the pre-computed grid node gradient. These properties give
1061 Perlin noise its characteristic spatial scale and smoothness.

1062 In general, the final result of the algorithm is a smooth function with a random-
1063 like behavior that mimics really well an organic appearance, like in Figure 2.8, with
1064 fluctuation around the value 0, with amplitude $\in [0; 1]$. The surface in Figure 2.8 has
1065 been produced plotting in 3D the results of the function `pnoise2` from the library `noise`,
1066 which offers a tool for the production of different type of noise. In order to put ~~in practical~~
1067 ~~use this module some adjustment~~ this module in practical usage, some adjustments were
1068 required. The particular function `pnoise2` simply yields the value of the Perlin noise
1069 surface in correspondence to a single (x, y) point in the plane in a deterministic way.
1070 There was not the possibility to generate different whole Perlin surfaces every run. To
1071 overcome this limitation I made a vectorized⁴ version of `pnoise2` able to evaluate the
1072 function over an arbitrary wide grid of points expressed as the set of all the pairs of
1073 coordinates (x, y) of the grid's nodes. In this way a single call to the function is able
1074 to produce the entire surface covering the grid, in the form of a single NumPy 3D-array.
1075 Furthermore, to recover the possibility of generating always different surfaces as in a
1076 random generation, I inserted an offset ~~e~~ coordinate coordinate (x_O, y_O) , which moves in

³Usually are used functions with a sigmoidal behavior, like any smoothstep function, which is a family of very common items in computer graphics.

⁴In Python the staple tool for scientific, number-cruncher computation is the NumPy library, which allows a fast, complete and efficient way to perform computation between number structure. The operation of ~~trasnforming~~ transforming a function which acts on a single value (or pair of values) to a function able to perform on a suitable ~~datastructure~~ data structure is called *vectorization*, and its the ~~recomended recommended~~ way to proceed when handling numerical functions in Python.

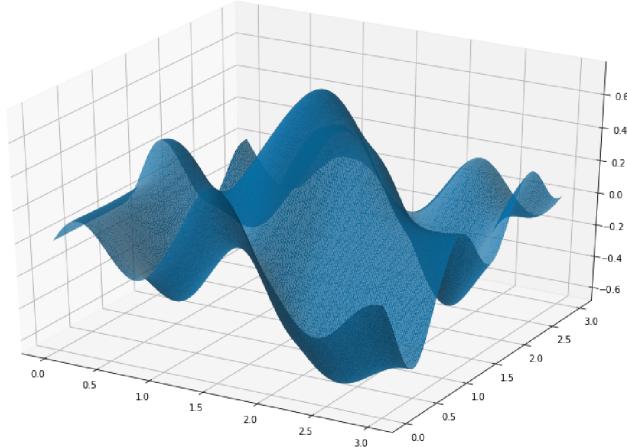


Figure 2.8: Example of Perlin noise 2D function produced while working on this project. This surface offers a smooth variation around the value 0 with amplitude $\in [0; 1]$.

1077 the plane the origin of the surface generation. This pair of offset coordinates then acts
 1078 also as a *seed* in the generation, allowing to completely recreate previously generated
 1079 material in a controlled way.

1080 2.7 Style-Transfer Neural Network

1081 Style-Transfer Neural Networks are common models, able ~~of creating to create~~ new hybrid
 1082 images implanting the visual style from an image preserving the visual content of another
 1083 image. The two images necessary for the algorithm are called *style* image S and *content*
 1084 image C , and the resulting *styled* picture X , as in Figure 2.9.

1085 There are many different tested and comparable architectures to compute this kind
 1086 of algorithm. In my work I decided to use in particular the procedure described in [18],
 1087 using the PyTorch ecosystem to implement the necessary code.

1088 The backbone of the architecture is the VGG-19 network, which is a convolutional
 1089 neural network 19 layers deep, as in Figure 2.10. This huge model has been pre-trained
 1090 on over a million images from the ImageNet database [16], for the classification into
 1091 over than 1000 classes of objects. As a result, the network has learned rich feature
 1092 representations for a wide range of images. The best (and conceptually the only) way to
 1093 load a pre-trained model is to load the ordered set of weights that define the network and
 1094 to initialize an empty module with those values. This is the perfect start for creating a
 1095 style transfer network, which requires a further and briefer training phase, to completely
 1096 customize the network.

1097 The key ingredient for finalizing the model is to insert some little but fundamental



Figure 2.9: Different examples of application of a style-transfer NN on the same content image, with different style images, from [18]. The original picture depicts the Neckarfront in Tbingen, Germany (TOP-LEFT). The painting used as style image shown in the bottom left corner of each panel are in clockwise order: • The Shipwreck of the Minotaur by J.M.W. Turner, 1805 • Femme nue assise by Pablo Picasso, 1910 • Composition VII by Wassily Kandinsky, 1913 • Der Schrei by Edvard Munch, 1893 • The Starry Night by Vincent van Gogh, 1889.

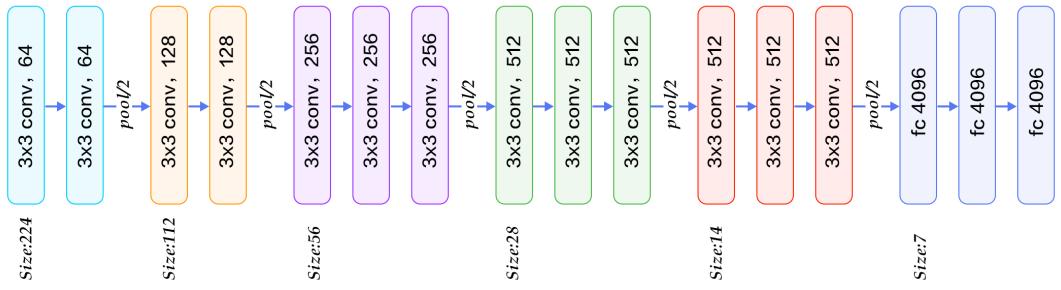


Figure 2.10: The structure of the VGG 19 network. It is for its most convolutional NN with 224×224 input size and with some downsampling layers which reduce the first two dimensions of the tensors along with the information flux of the network. At the very end of the architecture, there are three subsequent fully connected layers, responsible for the actual classification based on the features extracted from the previous layers.

1098 modifications and to extend the training on the pair of input images. This final training
 1099 should be aware of the *concepts* of the visual style and visual content of the image, and
 1100 the operation should try to preserve them both. This is usually done minimizing two
 1101 new loss function, computed between the staring image and the produced image:

1102 Content Loss

1103 The content loss is a function that represents a weighted version of the content
 1104 distance for an individual layer. The most commonly used function to evaluate the
 1105 preservation of content between two images is the simple Mean Squared Error as
 1106 in equation (1.5). It can be computed between any couple of same-sized object,
 1107 hence also between the results of the same feature maps on the images X and C
 1108 at the same layer L :

$$L_{Content} = \|F_{XL}F_{CL}\|^2. \quad (2.17)$$

1109 In order to evaluate this content loss, it is necessary to insert a custom transparent⁵
 1110 layer directly after the convolution layer(s) that are being used to compute the
 1111 content distance.

1112 Style Loss

1113 The concept of *style* loss function is the true novelty introduced by [18]. This loss
 1114 function is implemented similarly to the content loss module, as it will act as a
 1115 transparent layer in the network. The computation of the style loss requires in
 1116 ~~advanced~~advance the evaluation of the Gram matrix G_{XL} at a certain layer L . A
 1117 Gram matrix is a result of multiplying a given matrix by its transposed matrix. In
 1118 this case, the matrix to multiplicate is a reshaped version of the feature maps F_{XL} :
 1119 \hat{F}_{XL} , a $K \times N$ matrix, where K is the number of feature maps at layer L and N
 1120 is the length of any vectorized feature map F_{XL}^k . Furthermore, the Gram matrix
 1121 must be normalized by dividing each element by the total number of elements in the
 1122 matrix. The style distance is now computed using the mean square error between
 1123 G_{XL} and G_{SL} :

$$L_{Style} = \|G_{XL}G_{SL}\|^2. \quad (2.18)$$

1124 After the appropriate insertion of the loss-function evaluator layers, one last piece for
 1125 finalizing the model is the right choice of the gradient descent optimizer. As in [18] and
 1126 according to the Deep Learning community the optimizer which suite best this role is the
 1127 Limited Memory-BFGS [8],[40]. L-BFGS is an iterative algorithm in the family of quasi-
 1128 Newton methods that approximates the Broyden-Fletcher-Goldfarb-Shanno algorithm
 1129 (BFGS) using a limited amount of computer memory, and it is a popular choice when
 1130 estimating parameters of a non-linear differentiable scalar function.

⁵A transparent layer is a layer that performs some operations, like evaluating a function on its input, but returns as output an unchanged copy of its input.

1131 The final phase of the training process thus makes run the optimizer for hundreds of
1132 epochs on X , C , and S , and reduce the two loss functions values acting on the network's
1133 parameters. After the fine-tuning of the weights, the hybrid image is produced, as in
1134 Figure 2.9.

1135 2.8 Working Environment

1136 In this section, I will briefly ~~describe~~describe the machine I used to develop my project
1137 and the working environment I built. All the work has been done on my personal
1138 computer, mounting a `GNU/linux` operating system, in particular ~~the~~the 18.04 LTS Ubuntu
1139 version. The computer mounts an Intel i7 core, 8 Gb of RAM beside ~~an-a~~a 2Gb NVidia
1140 940MX GPU. All the Python libraries have been installed and harmonized in a virtual
1141 environment mounting `Python 3.7.6`. All the code produced during the development,
1142 the images, and the data produced have been collected in a devoted repository on GitHub
1143 [14], which is freely available.

1144 As a conclusion for this chapter I will ~~re-collect~~recollect all the references to the
1145 different Python libraries I used during the development of this work:

1146 **NumPy:** NumPy is the pillar of every scientific computation-oriented library. Is the most
1147 spread library for heavy multidimensional numerical computation, and it offers a
1148 broad variety of tools like random ~~numebr~~number generators, and pre-implemented
1149 linear algebra utilities [31].

1150 **SciPy:** The SciPy library is one of the core packages that make up the SciPy stack.
1151 It provides many user-friendly and efficient numerical routines, such as routines
1152 for numerical integration, interpolation, optimization, linear algebra, and statistics [46]. Two modules in particular form this libraries have covered an essential
1153 role in this project: the `SciPy.spatial.Voronoi` module for the computation of the 3D Voronoi decomposition, as mentioned in section 2.3, and the
1154 `SciPy.spatial.ConvexHull` module for the computation of 3D and 2D convex
1155 hulls (section 2.5).

1158 **PyTorch:** PyTorch is a rich ecosystem of tools and libraries geared toward Machine
1159 Learning and Deep Learning. The application of the style-transfer NN described
1160 in section 2.7 has required the use of this framework [32].

1161 **SALib:** The SALib is a library which collects many tools for the ~~Sensitiviy~~Sensitivity
1162 Analysis of parameters. In particular, the `SALib.saltelli` submodel was used for
1163 the quasi-random numerical sampling in a three-dimensional box, and it has been
1164 described in section 2.4.

1165 **pnoise2**: `pnoise2` contains many tools for the production of specific types of noise.
1166 The module `noise` was tweaked for the production of two-dimensional Perlin noise
1167 surfaces, and it has been introduced in section 2.6.

1168 **pyquaternion**: The `pyquaternion` library provides a framework for handling quater-
1169 nions. It has been widely used in section 2.1 for the design of three-dimensional
1170 ramifications for handling multiple spatial rotations.

₁₁₇₁ **Chapter 3**

₁₁₇₂ **Tissue Models Development**

₁₁₇₃ The main goal of the present work, as stated before, is to recreate a three-dimensional
₁₁₇₄ virtual model of histological tissue as faithfully as possible and then, to perform planar
₁₁₇₅ sectioning on it to emulate virtually the traditional histological specimen preparation
₁₁₇₆ procedure. The creation of a model of such complex structures is definitely a high-
₁₁₇₇ level problem, and it has required a careful designing, made of subsequent stages of
₁₁₇₈ improvements. In section 3.1, I will describe all the necessary steps to create the model
₁₁₇₉ of a small region of pancreatic tissue, while in section 3.2 I will expose the steps I
₁₁₈₀ followed to build a model of dermal tissue. In section 3.3, instead, I will show the
₁₁₈₁ resulting synthetic images from the sectioning process performed on both the models
₁₁₈₂ and all the enrichments and processing necessary to give them the most realistic look I
₁₁₈₃ was able to recreate.

₁₁₈₄ **3.1 Pancreatic Tissue Model**

₁₁₈₅ ~~The Panereas is an internal organ of the human body, part of both the digestive system~~
₁₁₈₆ ~~and the endocrine system. It acts as a gland with both endocrine and exocrine functions,~~
₁₁₈₇ ~~and it is located in the abdomen behind the stomach. Its main endocrine duty is the~~
₁₁₈₈ ~~secretion of hormones like insulin and glucagon which are responsables for the regulation of~~
₁₁₈₉ ~~sugar levels in the blood. As a part of the digestive system instead it acts as an exocrine~~
₁₁₉₀ ~~gland secreting pancreatic juice. The majority of pancreatic tissue has a digestive role,~~
₁₁₉₁ ~~and the cells with this role form clusters (*acini*) around the small pancreatic duets. The~~
₁₁₉₂ ~~acinus secrete inactive digestive enzymes called zymogens into the small intercalated~~
₁₁₉₃ ~~duets which they surround, and then in the pancreatic blood vessels system [27]. In~~
₁₁₉₄ ~~Figure 1.3 is shown a picture of the pancreas, with its structure and its placement in the~~
₁₁₉₅ ~~human body.~~

₁₁₉₆ ~~(left) A picture of pancreas' structure in its phisiologiael context. In this picture is~~
₁₁₉₇ ~~clearly visible the macroscopic structure and the galndular organization at microscopic~~

1198 level, and how it reflects in the histological sample. (right) A real pancreatic tissue
1199 sample with H&E staining.

1200 All the tissue is actually rich in other important elements as the islets of Langerhans,
1201 and sporadic connective tissue all over the structure, which are clearly visible in the
1202 traditional histological specimens. In this first attempt of modelization from scratch
1203 this second layer of complexity has not been already considered, and the main focus has
1204 been to reflect was put on reflecting only the main structural features on the virtual
1205 specimens. Given the pancreatic tissue's organization, described back in section 1.1.2,
1206 the first features I decided to put emphasis emphasize on were: 1) The iterative (with a
1207 fractal-like behavior) ramification of blood vessels for the irrigation of glandular acinus,
1208 2) The space-filling distribution of acinus in the tissue, in fact, we expect a homogeneous
1209 density in the organ and to not see holes at all inside it. In this section I will describe
1210 step by step all the process I followed to create the model of a portion of pancreatic
1211 tissue, and all the interesting pitfalls I overcame.

1212 3.1.1 2D Ramification

1213 The first step was took taken in two dimensions, and it was the choice of the right
1214 structure to emulate the ramification of blood vessels in pancreatic tissue. The choice
1215 fell on a particular parametric L-system, as the one shown in Figure 2.1, in section 2.2.
1216 This structure is made of an iterative bifurcation of gradually shorter segments, with an
1217 angle of $\pm 85^\circ$ respect the main direction. For a start I added some features to give a
1218 more realistic look to the structure, which are all well represented in Figure 3.1:

- 1219 • A progressive thickness of the bifurcation's segments, starting from a thick main
1220 branch that dwindle every junction. The idea is that the main blood vessel be-
1221 comes gradually smaller becoming capillaries for single-cell irrigation.
- 1222 • A progressive randomness in the angular deflection at every fork. Perfectly repeated
1223 angles are almost nonexistent in nature, so I decided to introduce an increasing
1224 indetermination in the angle of bifurcation from the main branch to the free ends
1225 of the structures' branches.
- 1226 • Spheres at the ends of each branch, which acts as glandular acini. The maximum
1227 radius is comparable to the length of the final segments.
- 1228 • A mechanism to avoid self-superimposition between branches and spheres. After
1229 the insertion of noise, the cumulative effect on the final segments might lead to
1230 different branches to intersect. This is clearly a paradoxical situation, as real
1231 tissues while growing naturally occupy the space in a gradual way.

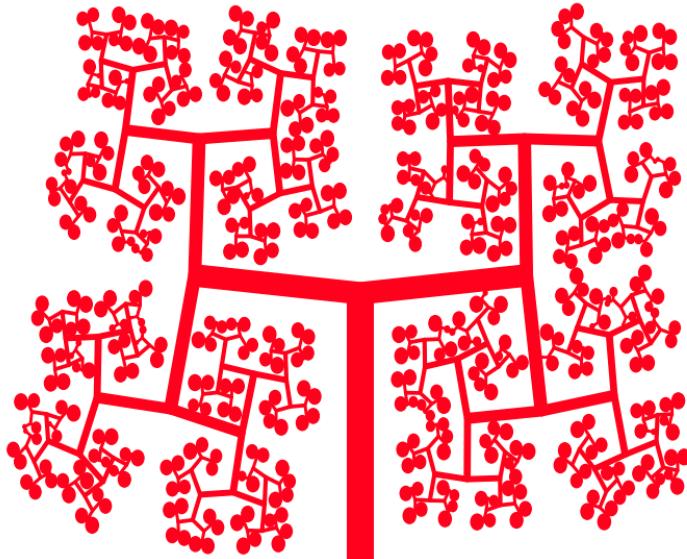


Figure 3.1: The development of the simple ramification in Figure 2.1, with some features to give it a more realistic look, like progressive thickness, angular noise in bifurcation, and spheres at free ends of the ramification. The image is made using the tools exposed in section 2.2.

1232 To produce the specific image in Figure 3.1 I used a particular setting of the tool
 1233 described in section 2.2, which have has a greatly wider range of customization and could
 1234 be used to create many other different structures to the need.

1235 3.1.2 Expansion to 3D

1236 The successive step I followed was to expand this structure in three dimensions and fill
 1237 the space in each of the three directions. The idea to evolve the structure in Figure 3.1 is
 1238 simply to twist of 90° the ramification at every junction point, in such a way to exit the
 1239 previous belonging plane. However, putting into practice this development has not been
 1240 easy. The organization of the structure in a 3D space requires an appropriate system
 1241 of reference for handling subsequent rotations in three dimensions. The best option for
 1242 handling relative 3D rotations, often used in computer graphics and every kind of 3D
 1243 modelization, are quaternions, as shown in section 2.1.

1244 In this new structure, segments are replaced with cylinders, and circles are replaced
 1245 with spheres. At every bifurcation to every cylinder are applied the following transfor-
 1246 mations:

- 1247 • a contraction in its extensions, regulated by an adjustable parameter R .
- 1248 • the usual deviation of $\pm 85^\circ$ respect to the direction of the parent branch.

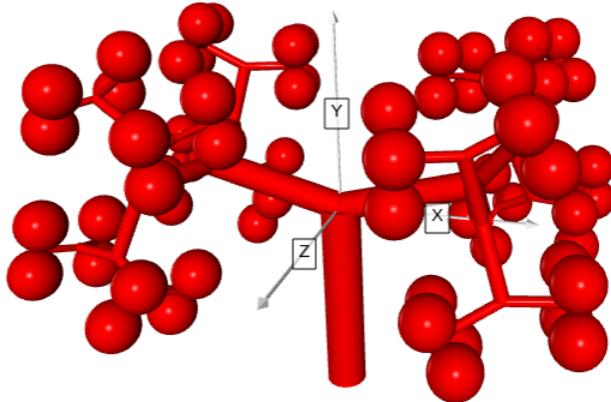


Figure 3.2: The three-dimensional expansion of the 2D ramification in Figure 3.1.

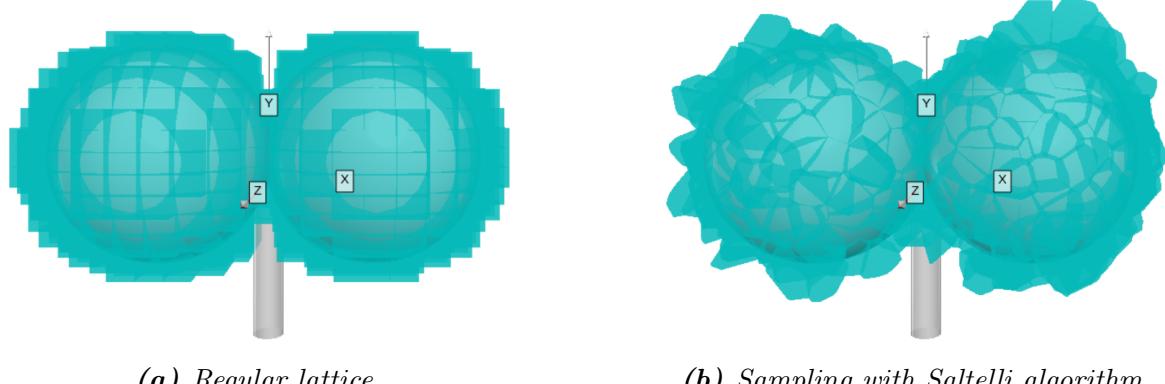
- 1249 • a 90° specific rotation along the axis of its parent branch.

1250 The result of this procedure is a 3D ramification like the one in Figure 3.2, in which
 1251 we can recognize a good coverage of the space defined by the structure's boundaries and
 1252 immediate relation with the 2D structure in Figure 3.1. It should be noted that, in the
 1253 further refinements of the model from now on, there won't be present the progressive
 1254 angular indetermination on the direction of branches. Although it is a feature already
 1255 implemented and working, it requires efficient control to avoid reciprocal overlapping
 1256 between elements to produce a realistic structure. This second element has not been
 1257 already developed and it would certainly enrich the representative power of the model.

1258 As for the 2D ramification the production of this structure has required the imple-
 1259 mentation of a tool for the 3D generation with a greatly wider power, able to produce
 1260 almost any type of three-dimensional iterative structure after the right adjustment, and
 1261 with a high degree of customization. It is necessary to mention the fundamental tool
 1262 which allowed me to accomplish this step of the development, which is the `Python` library
 1263 `VPython`: a library for 3D graphics visualization. This library allows a convenient and
 1264 powerful interface to draw many types of objects and to move them around in space,
 1265 which has been priceless to orient my self in three dimensions while developing the model
 1266 and to produce all the 3D images visible in this work.

1267 3.1.3 Subdivision in Cells

1268 Once the 3D backbone of the pancreatic tissue blood vessels ramification system has
 1269 taken shape, the next step was to embed all this structure in a spatial partitioning
 1270 process, to create the subdivision into single cells. To perform this important task I



(a) Regular lattice.

(b) Sampling with Saltelli algorithm.

Figure 3.3: Comparison between two Voronoi decompositions. The first (left) is created from a regular lattice of starting points, and every piece is exactly equal to all the others, creating a regular subdivision of the space. The second (right) is created instead from a sampling made following the Saltelli quasi-random algorithm. The pieces are all different in shape, but they all have similar sizes and volumes. In this pictures in particular have been shown only the pieces of the tessellation which lie in correspondence to the boundaries of the spheres underneath. While watching this picture one should immagine the decomposition extended similarly in all the space around the ramification, within certain boundaries, which loosely contains the structure. This limitation was necessary to enhance the interpretability of the image.

1271 used a 3D Voronoi decomposition, as shown in section 2.3. Depending on the choice
 1272 of the starting points, the Voronoi tessellation could be an excellent item to recreate
 1273 individual cells because it could guarantee some important properties: all the regions
 1274 are convex, adjacent, with similar size and volume, with different shapes, and without
 1275 holes. These have been chosen as the most significant properties to be reflected in the
 1276 first modelization of cells.

1277 As shown in section 2.3, the Voronoi decomposition strongly depends on the choice of
 1278 the starting point. Points spread uniformly on a 3D regular lattice will produce a series
 1279 of parallelepipeds repeated in the space. An example of uniform tessellation is shown in
 1280 Figure 3.3a. On the other hand, a decomposition based on a quasi-random generated
 1281 point can present all the good properties we mentioned before, including the diversity in
 1282 shapes. In Figure 3.3b is shown an example of a Voronoi decomposition based on points
 1283 sampled in a 3D with the Saltelli algorithm, in reference to section 2.4. Regardless of
 1284 the points sampling technique, the boundaries of the sampling 3D box have been chosen
 1285 to loosely contain the ramification.

1286 There are tough some delicate considerations to be highlighted about the decomposi-
 1287 tion procedure. The first regards the most external pieces of the decomposition. Whilst

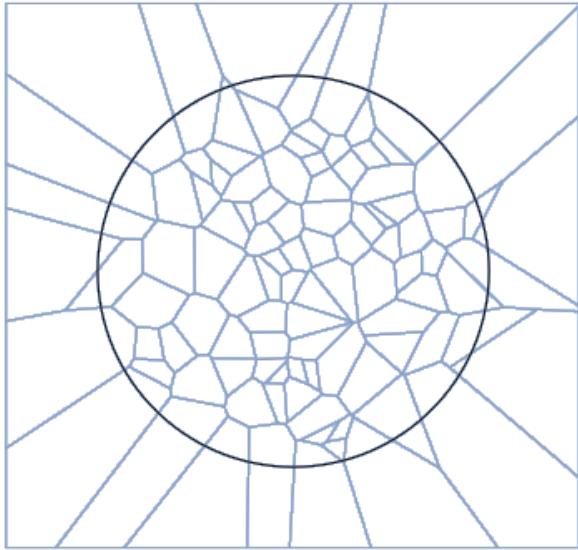


Figure 3.4: Example of circular cropping in a 2D Voronoi decomposition: all the regions which intersect the circumference have to be resized.

the internal pieces are neatly bounded and defined, the most external layer instead is made on unbounded regions, which extend themselves to infinity. Those regions have clearly to be rejected, as it would be absurd for a cell to have an infinity volume. Typically those unbounded regions are resized in order to adhere to some limiting boundaries, with an operation known as *cropping*. In Figure 3.4 is shown an example of circular cropping in a 2D Voronoi decomposition: all the regions which intersect the circumference have to be resized.

The cropping operation in 3D is extremely complex, tough. Thus, a more simple and efficient, yet less elegant, technique has been used. Instead of resizing the regions which lie on the boundaries of the sampling region, those regions have directly been rejected. This process is really fast and it does not lead to any danger of representativity loss if the boundaries are loose enough and if the density of sampling is not too low.

The other important consideration regards the density of sampling points. Increasing the number of points to be extracted from the same volume automatically the number of cells in the box will rise, and in contrast, their relative dimension will decrease. This is a key element of the model: a too rarified decomposition would not be able to reflect the complexity of the structure underneath, but a too crowd decomposition on the other side would lead to an unrealistic dimension of the cells in the tissue. Furthermore, this parameter has a huge influence on the computing time necessary to generate the model and to process it for the sectioning as will be shown in section 3.3. In almost all the applications so far, the density parameter has been tuned by eye, with a trial and error procedure. Although, a more rigorous way to adjust this parameter would be to consider

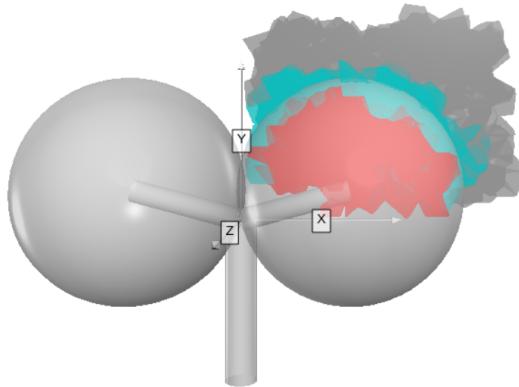


Figure 3.5: Portion of the complete Voronoi decomposition, showing the three different classes of cell in three different colors: the internal cells in red, those on the boundaries in turquoise and the external cells in gray.

the average dimension of the cells and make some microanatomical considerations to define the correct relative dimensions. The measure of the volume¹ of the decomposition's regions is an accessible parameter, thus an easy way to estimate the average linear dimension of the cells can be to approximate all the cells to cuboid seeing their volumes as $V \approx L^3$. Averaging all the L measures an estimate \hat{L} can be done. This average length may be compared to the length of the blood vessel ramification, allowing a good reference tool.

3.1.4 Cells Identity Assignment

The great power of creating all the models virtually is to know exactly the identity of every point in the structure. Although, This identity has to be reflected at the cellular level, assigning to every region a label. Imagining the Voronoi decomposition represented in Figure 3.3b extended to the entire box containing the ramification, good discrimination would distinguish three classes of cells: those which lie completely inside a sphere, those which lie completely outside a sphere, and those which lie on the boundaries of a sphere. In Figure 3.5 is shown a portion of the complete decomposition where the three classes of cells are reported with different colors: the internal cells in red, those on the boundaries in turquoise, and the external cells in gray.

In this particular case to find the relative position between every sphere in the structure and each cell it has been used a test on the proximity between the spheres' centers and the vertices of every polyhedral cell. If all the vertices of a region lie within a distance lower than the radius from the center of the same sphere then that region can be

¹The volume is expressed in the same arbitrary length unit of measures used during the ramification structure. This allows a coherent reference tool.

1331 said to be an internal one. If none of the vertices lie within the radius distance from
1332 any center then that region is said to be external. In any other case, the region is said
1333 to be on the boundaries of some sphere, and this third label is assigned to it. As could
1334 be imagined the number of cells inside the volume can grow very quickly, and in the
1335 more rich ramifications also the number of spheres could be high. If we think that any
1336 polyhedron has a number of vertices of the order of 20/30 then it is clear that the number
1337 of distance evaluations could grow very quickly, requiring some relevant computational
1338 power in the more extended simulations. In order to optimize this computation, I decided
1339 to use a python implementation of a K-dimensional Tree, which is a space-partitioning
1340 data structure especially suited for fast and optimized computation of distances [6]. A
1341 K-d Tree is an algorithm that iteratively binary splits the space: every node of the three
1342 could be thought as a splitting $(k - 1)$ -hyperplane dividing the space into two semi-
1343 hyperspace. The result is an optimized algorithm for repeated distance evaluations. As
1344 for many other tools, in my code I used a pre-implemented module `KDTree` from the
1345 `Scipy` library.

1346 This procedure of labeling the regions is completely customizable, and it should be
1347 adapted to the specific application. By the way, the principle will always be to perform
1348 some sort of spatial consideration respect to the primary structure and assign all the
1349 interesting labels accordingly to the cells in the volume.

1350 After labeling the cells in the decomposition the model is considered complete. Every
1351 enrichment to the structure should be reflected in some type of label for the cells, which
1352 are chosen as the fundamental unit in the model. As we will see in section 3.3 during
1353 the sectioning process in the produced image will be printed mainly the identity of the
1354 cells, hence any detail on a finer scale in the model would not be conveyed properly on
1355 the final image.

1356 3.2 Dermal Tissue Model

1357 Skin is the layer of soft, flexible outer tissue covering the body of a vertebrate animal,
1358 with the three main functions of protection, regulation, and sensation. Mammalian
1359 skin is composed of two primary layers: the epidermis, which provides waterproofing
1360 and serves as a barrier to infection, and the dermis, which serves as a location for the
1361 appendages of skin. The epidermis is composed of the outermost layers of the skin. It
1362 forms a protective barrier over the body's surface, responsible for keeping water in the
1363 body and preventing pathogens from entering, and is a stratified squamous epithelium,
1364 composed of proliferating basal and differentiated suprabasal keratinocytes. The dermis
1365 is the layer of skin beneath the epidermis that consists of connective tissue and cushions
1366 the body from stress and strain. The dermis provides tensile strength and elasticity to
1367 the skin through an extracellular matrix composed of collagen fibrils, microfibrils, and
1368 elastic fibers, embedded in hyaluronan and proteoglycans.

1369 (left) Microanatomical description of a region of dermal tissue and all the interesting
1370 elements present in cutis, and subcutaneous layer. (right) An actual histological specimen
1371 from a sample of dermal tissue.

1372 The modeling of the dermal tissue has followed analogous schedule respect to the
1373 previous model, hence many procedures and considerations have been repeated. The
1374 main target for this model was to recreate the stratification of different specific tissues
1375 in the section of a dermal sample. As clearly visible in Figure ??-1.4 in a dermal
1376 histological specimen one can distinguish the lighter and wider region of proper *dermal*
1377 tissue, underneath a more shallow and darker region of *epidermal* tissue. It is very
1378 interesting the boundary between those two regions, which can be seen as an irregular
1379 and smooth surface, populated of dermal lobes. On the other side of the epidermal layer
1380 lies a layer of *keratin*, with a smooth and regular boundary surface. Keratin is a family
1381 of fibrous structural proteins known as scleroproteins, a key structural material for hair,
1382 nails, and the outer layer of skin. The upper white region in Figure ??-1.4 instead is the
1383 support for the samples to perform optical analysis, and has no histological meaning.

1384 1) Stratified Structure

1385 In order to represent faithfully the stratification of different tissue layers, I decided
1386 to use one flat plane to separate epidermal tissue and the keratin layer and two
1387 boundary surfaces modulated by a Perlin noise function on different scales for the
1388 other two separations, as shown in Figure 3.6. As stated in section 2.6, after some
1389 easy customization the generation of different Perlin noise surfaces is easy and
1390 straightforward. By the way, to achieve the regularity and the smoothness of the
1391 orange surface in Figure 3.6 it was needed a horizontal stretching.

1392 Following the scale of the image, the standard blue surface is created in a 7×7
1393 square, while the orange surface has primarily been generated in a 1×1 square,
1394 and then has been stretched to cover the same 7×7 square, multiplying the values
1395 in its grid points respectively for the stretching factors $R_x = R_y = \frac{1}{7}$. In this
1396 primary structure, there are many important parameters defining the surfaces, like
1397 the distance between the two ~~surfae~~surfaces' average values and the amplitudes of
1398 the peaks and the valleys of the surfaces. In its standard version the Perlin noise
1399 covers the $[-1; 1]$ range, but with a simple multiplication for an amplitude factor
1400 A_S the values can be adjusted. Those particular values have been adjusted after
1401 some tries to recreate the proportions typical of a real specimen. To sum up, each
1402 one of the two surfaces is stored as a discretized three-dimensional array, or better
1403 as an array of 3-tuples in the form $(x, y, f_{(x,y)})$, one tuple for every (x, y) node of
1404 the grid, while the discretization grid was the same for both the surfaces.

1405 2) Subdivision in Cells

1406 The subdivision in cells of the volume containing the structure has followed the
1407 exact same steps described in section 3.1, hence in this paragraph I will shortly

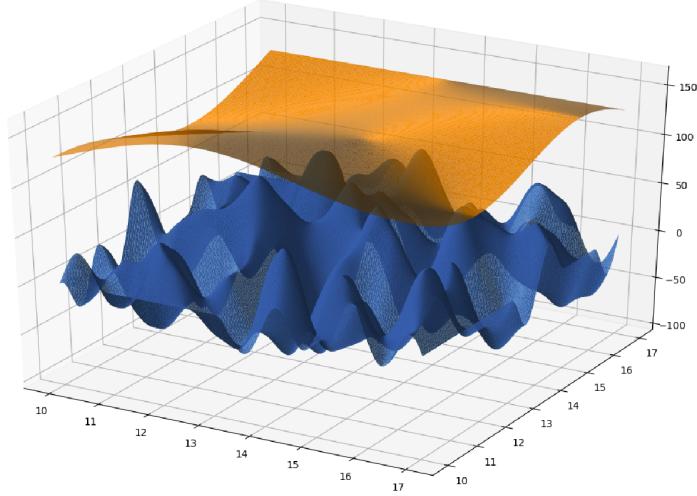


Figure 3.6: Pictures of two different Perlin noise surfaces used to separate dermal from epidermal tissue (blue) and epidermal from keratine layer (orange). The two surface are made by the same Perlin noise function, but the latter is stretched and compressed in order to have a more regular behavior.

resume the process. The first step is the definition of a suitable volume containing the structure. Then is the time for the generation of the decomposition's starting points according to a quasi-random number generation technique, as described in section 2.4. Afterward, the points are used as a base for the decomposition, and all the cells with undefined boundaries are rejected

3) Cells Identity Assignment

The identity assignment procedure instead has been customized for this particular application. In this model there are no *boundary* cells as-like the one lying on the spherical surfaces in the pancreatic model, thus there is no need for a test on the position of each vertex of every cell. In this model, the starting point for every Voronoi region has been used as a reference, and its relative position respect to the boundary surfaces was the discriminating factor for assessing the identity. In order to perform a coherent test on the relative position between the regions' center and the boundaries surfaces the positions of all the centers had to be discretized on the same grid onto which were defined the surfaces. The comparison with the flat horizontal plane defining the boundary between epidermal tissue and the keratine layer instead was simply a test on the z coordinate of the point: $z = f_{(x,y)} \leq \hat{z}_{plane}$. The result of this procedure is that every region is assigned a label corresponding to the belonging tissue layer: D for dermal, E for epidermal, K for keratine, and V , which stands for *void* for the white empty space above the sample.

₁₄₂₈ In this model, as in section 3.1, after the assignment of the identity to all the cells in
₁₄₂₉ the volume the modelization process is considered complete. Any sort of improvement
₁₄₃₀ and enrichment should be inserted during the structure designing phase and should be
₁₄₃₁ linked to an identity label.

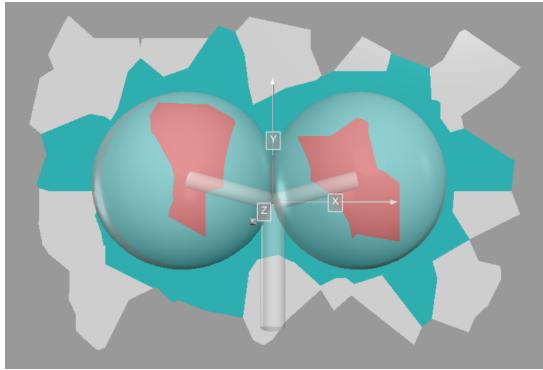
3.3 Synthetic Images Production

After the model is complete we have a three-dimensional representation of the tissue under study. The aim of the work is though to produce synthetic images from that structure, more precisely we want pairs of images composed of the synthetic-histological images and its related segmentation mask. The transition from 3D structure to a 2D image is the last step in the process, and it is inspired by the actual traditional technique for the preparation of the histological specimen, as described back in section 1.1.1. As the biopsy sample is treated and then sectioned with the microtome, the virtual model is sectioned in a random direction, producing an image representing the slice. This first image contains all the information of the section but its appearance is completely arbitrary and its look has nothing to share with a realistic sample. The original slice then acts perfectly as a segmentation mask, but some careful and dramatic makeover is needed to produce the final realistic looking image. In this section I will describe the general procedure to produce virtual slices from the two 3D virtual models described before in section 3 and the technique used to edit the images and give them the desired appearance.

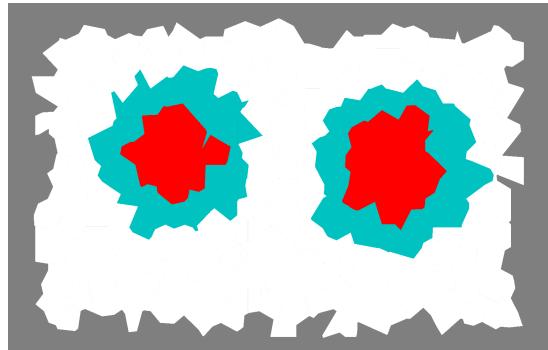
3.3.1 Sectioning Process

For any model created following the general procedure described in 3, even more so for the two particular models of pancreatic and dermal tissue, the sectioning process will be almost the same, and it will rely mainly on the algorithm for the general section of polyhedron described back in section 2.5. As stated before the models are essentially composed by labeled polyhedron spatially organized in a 3D volume. The ordered section of each polyhedron will yield all the polygons that shall be assembled in the final section. In Figure 3.7a is shown the three-dimensional representation of the section of a simple ramification, as the one in Figure 3.5. All the polygons that compose the section are drawn with the color correspondent to their label, following the same idea of Figure 3.5. In Figure 3.7b is printed the final result of the sectioning algorithm applied to the model, which will be the segmentation mask in the single pair of synthetic images. The colors in the produced slice match the colors used for the different identities in the 3D model.

The simplest, yet over-abundant, way to proceed is to create the model in its entirety and subsequently choosing the sectioning plane. Afterward, it is necessary to select only the regions that intersect the plane and section them all. Actually, the test on the intersection passes through the check on the relative position of the polyhedron's vertices respect to the sectioning plane: if all the vertices lie on the same semi-space then the intersection would be null and the polyhedron is not of interest for that particular section. This procedure is exactly the first step of the algorithm in 2.5, thus the filtering on the regions is actually made during construction for optimization. The alternative method could be to choose in the first place the direction of the sectioning plane, and in second



(a) 2D polygonal sections represented in a 3D environment.



(b) The final section image, produced directly on a planar picture, skipping completely the 3D representation.

Figure 3.7: In this Figure is shown the idea of the correspondence between the section of a simple structure in the space and the correspondent section. The correspondence is not perfect for representation requirements, in fact the two images even if very similar are produced with two completely different methods. At the left an image of 2D polygonal section embeded in a 3D space, made using 3D visualization tools. At the right a simple image produced printing the polygons in a planar picture. Printing 2D polygons in a 3D space is much more complicated than one would think using the same tool used to produce the other images like Figure 3.5 and 3.3. This choice has been done for the sake of the overall homogeneity in pictures style.

1470 place to generate the model's decomposition only in the volume adjacent to that plane.
 1471 This method enables the sparing of a good amount of computation, without any negative
 1472 impact on the final result. The only delicate step is the choice of a wide enough region
 1473 of space around the plane, which doesn't compromise the representation.

1474 As a guarantee for richness and diversification among the images, there is the need
 1475 for some degree of controlled randomness in the sectioning process, for example in the
 1476 determination of the sectioning plane direction. All the sectioning process is then based
 1477 on a single starting *seed*, which determines the direction of the sectioning plane in a
 1478 deterministic way, and all the rest of the model is generated as a consequence. In this
 1479 way, all the possible angulations are equally probable and will be sampled in view of
 1480 multiple applications of this process. In Figure 3.8 are shown two different sections,
 1481 along two random planes on two simple ramified structures with four spheres.

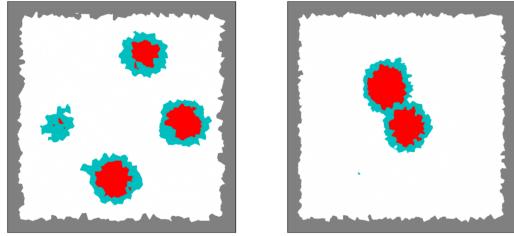


Figure 3.8: Two different section, along two random planes on two simple ramified structure with four spheres.

1482 3.3.2 Appearance Makeover

1483 After the application of the sectioning algorithm of the previous section, the image
 1484 which will act as a segmentation mask is ready. The last and most complex task that
 1485 remains is to transform the image and to give it a realistic look. I tried many different
 1486 transformations, more or less complicated, and there was not a final decision on which is
 1487 the best blend of them. In this section, I will describe them as an arsenal of possibilities
 1488 and show their impact on the images.

1489 Color Palette

1490 The first correction to do to the images will inevitably be a change in the colors
 1491 of the image. Gray, Turquoise, and Red are the perfect choice for label-colors but
 1492 act poorly as physiological colors. In Figure 3.9 is shown an example of an image
 1493 produced re-mapping the colors to a new palette, inspired to the coloring of the
 1494 real specimen in Figure 1.3 and 1.2, given by the traditional hematoxylin and eosin
 1495 staining process.

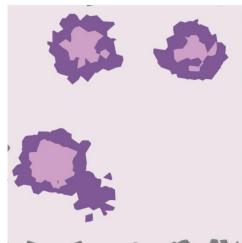


Figure 3.9: Example of images produced re-mapping the colors with a color palette inspired to a real H&E stained histological sample.

1496 Nuclei Projection

1497 Another fundamental processing needed was the projection of cells' nuclei on the
 1498 image. Usually, nuclei are clearly visible in histological samples and guide the

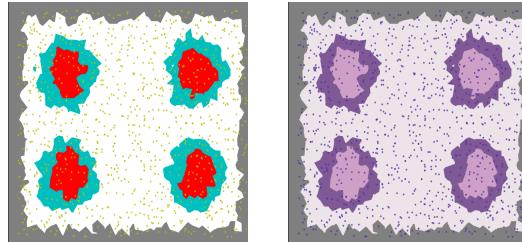


Figure 3.10: Nuclei projection on the image: (left) in yellow in the segmentation mask and (right) in purple in the image under makeover.

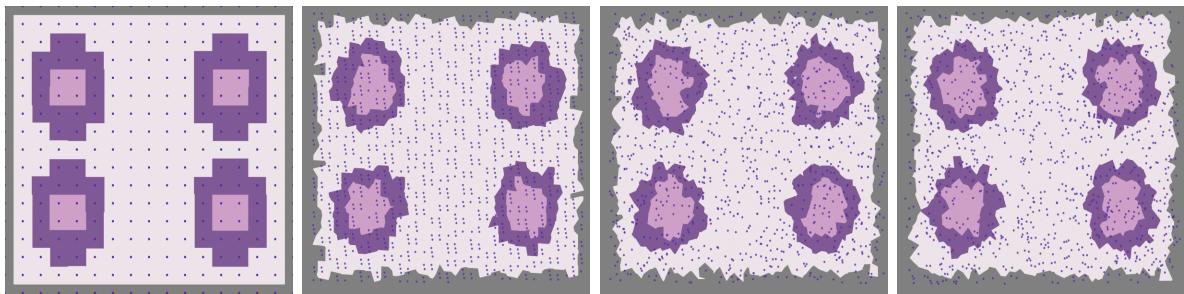


Figure 3.11: Four different sections produced with the same density on cells but with four different method for the sampling of starting decomposition's ponits (from left to right): • points sampled on a regular lattice, • sampling following a simple recursion sequence as the one in equation (2.12), • following the saltelli algorithm, • following a fully-random distribution.

analysis allowing to detect individual cells in the specimen. As a reference for the nucleus position the starting point of every polyhedral region has been used and projected on the sectioning plane as a little dark circle. The diameter of those circles as been chosen to be a submultiple (10%) of the linear estimated dimension \hat{L} of the cells in the decomposition².

Nuclei projection, among the other things, is an excellent tool to perceive the different effects obtained with different choices of quasi-random distributions or fully-random distributions (with reference to section 2.4). The different impact on the overall image is huge, and it really changes the overall sense of the image. In Figure 3.11 are reported four different sections, produced with the same density on cells but with four different methods for the sampling of starting decomposition's points.

Boundaries Projection

²Following the same logic of step 3 of section 3.1.

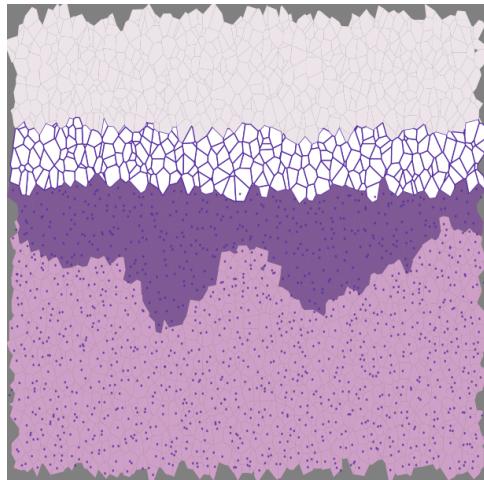


Figure 3.12: Example of images produced re-mapping the colors with a color palette inspired to a real H&E stained histological sample.

On the same wave of the previous tool, another operation that can help the appearance of an image is the projection of the boundaries of each (or just a part) of the polygonal sections. The drawing can be clearly tuned and customized depending on the specific necessities. In Figure 3.12 is shown an example of a section on the dermal tissue model in 3.2, in which the boundaries of all the cells have been lightly marked, and the boundaries of the cells in the keratine layer have been heavily marked instead.

Blurring Effects

In all the images produced so far the boundaries between polygonal sections are perfectly sharp and without any smudge. To give a more realistic feeling to those pictures I tried different forms of blurring. As a first try, I applied a Gaussian blurring filter, which is an extremely common blurring operation in computer vision, which consists of a simple discrete convolution with a 2D Gaussian kernel. The effect is a regular and diffuse blur all over the image, as in Figure 3.13a. The second blurring effect I implemented was based instead on the averaging of parallel and adjacent slices on the same model. This method is inspired by the real sectioning technique (section 1.1.1), in which every slice is not an infinitesimal layer of matter, but a finite sample, which suffers from mechanical dragging during the process. The idea is that the average between three or more slices equally spaced above and below the *main* slice should recreate a realistic blurring effect. An example of an image produced with this process is shown in Figure 3.13b.

Perlin RGB Noise

A further attempt to give visual texture to the image was done using again the

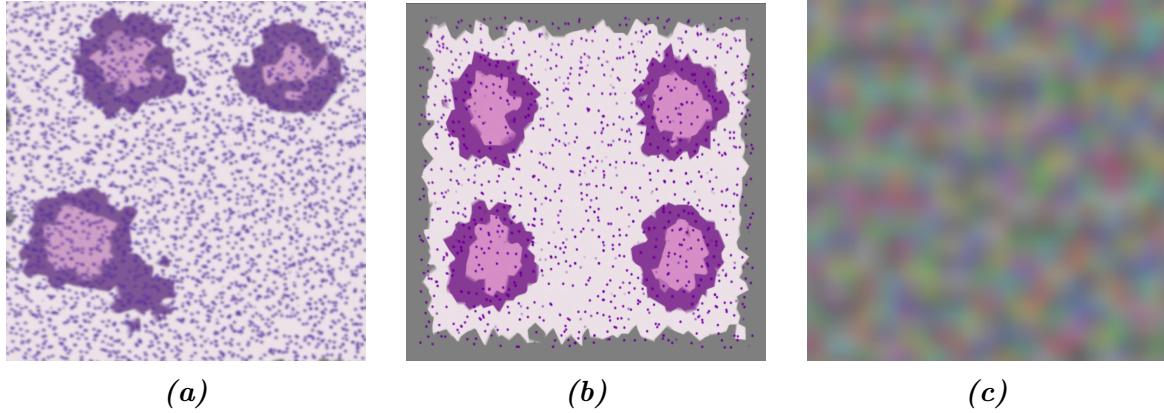


Figure 3.13: The two blurring effects used in this work: (left) A standard Gaussian-filter blur and (center) a specific blur introduced averaging adjacent parallel slices on the same image, (right) an example of RGB color noise built joining three different Perlin noise surfaces, one for each color channel.

Perlin noise, described in section 2.6. The idea is to create some fluctuation among the color channels of the image around the sharp values of the image produced by the sectioning algorithm. From a practical point of view, I created three different and independent Perlin noise surfaces, one for every color channel (Red, Green, and Blue), and added them to create an RGB noise on the image. An example of the resulting image is shown in Figure 3.13c.

Style Transfer

This last tool I will describe is the most sophisticated so far. It consists of the application of a style-transfer neural network (STNN) on the image obtained through the sectioning process, for the implantation of the visual texture from a real sample of the corresponding tissue. Style-transfer NNs, and their functioning, have been described in detail in section 2.7, and here I will cover just the particular applications on the two type of section produced.

The first manipulation I report is the one on a section from the pancreatic tissue model. The image of which to conserve the visual content is a section with some simple pre-processing picked from the ones described before (Figure 3.14a): a more accurate color palette, the projection of nuclei, and the average on five adjacent slices. The image from which to pick the style, thus the visual texture, is a portion of an actual histological sample of the pancreas, and it is shown in Figure 3.14b. The application of the STNN yields a hybrid image, shown in Figure 3.14c. The second application was made on a section on dermal tissue. The three content, style and styled images are shown respectively in Figure 3.15a, 3.15b, and 3.15c.

In Figures 3.14, and 3.15 are reported the best results among all the different tests

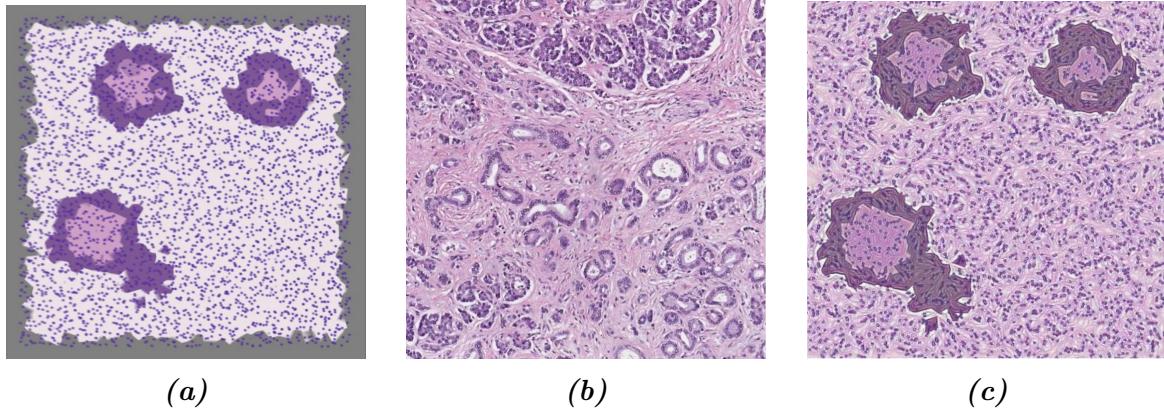


Figure 3.14: Application of the style-transfer NN on a section of the pancreatic tissue model: 3.14a the content image, 3.14b the style image, 3.14c the hybrid resulting image.

made on the sections. I made different tries on the same image with different processing before the manipulation with the STNN, to see the impact of the different adjustment on the resulting styled image. It turned out that the presence of nuclei is essential to give a homogeneous texture to the image and avoid unrealistic artifacts. On the other hand, the choice of the color palette has a way lighter effect than what one would think: the model yields almost the same result with a grey-levels image or with any other palette.

It is interesting to notice the timing cost of this style transfer operation. While all the other manipulation described in this chapter requires a very short time (seconds) to be applied, and are in practice *instantaneous*, the transfer of style is a way more robust operation, which implies the finalization of the training of a pre-trained neural network. On a computer with the technical specification described in section 2.8 this operation instead took minutes, which is a time two full orders of magnitude greater.

It should be noted that the presented results are obtained from the application of a pre-trained STNN model. The development of a specialized model for histological texture transfer could improve extremely the ability to produce realistic images, way further the present results.

One single complete application of the process consists then in the generation of a tissue model, in the sectioning along a random section plane, and in the processing of the image, in order to produce the pair of ground-truth image and the synthetic histological image. The target is to apply over and over this process to collect the necessary amount of images and constitute an entire dataset. An important feature to have for the process is thus a complete automatization, in order to scale up the

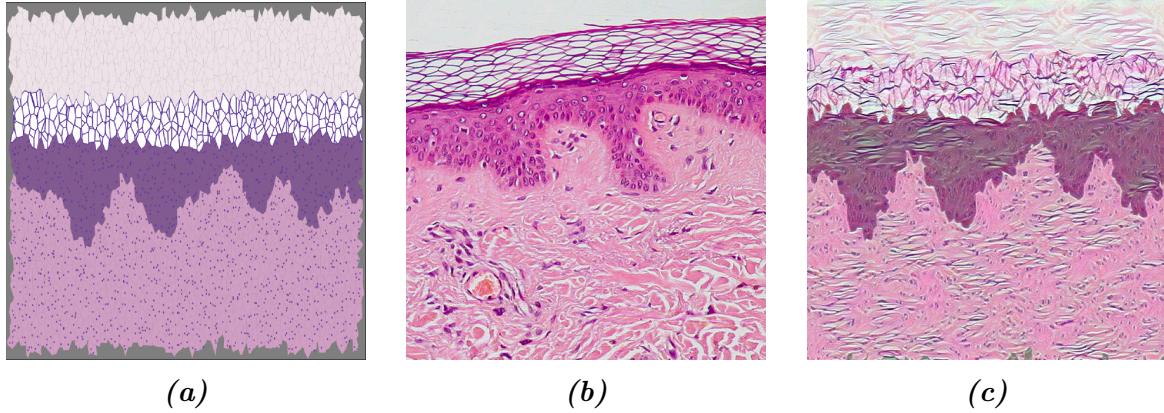


Figure 3.15: Application of the style-transfer NN on a section of the dermal tissue model: 3.15a the content image, 3.15b the style image, 3.15c the hybrid resulting image.

generation of images, possibly even in parallel computation. For this purpose, I created a pipeline workflow interface for the image generation, with an automatized harmonization of every piece of the process. The generation now requires just to fill a configuration file in which writes all the specific characteristics of the images: the type of structure, its features, the desired processing on the images, and eventually the random seeds for a supervised generation. In Figure 3.16 is reported a small scale example of a dataset produced with multiple automatized applications of the generation tool on a ramified structure inspired to a pancreatic tissue model. It is clear the correspondence between segmentation mask and synthetic histological images, and the diversification given by the supervised randomness on the generation.

The actual tool used for the set up of a working pipeline was the **Snakemake** [25] workflow management system, which is a python-based tool to create reproducible and scalable data analyses.

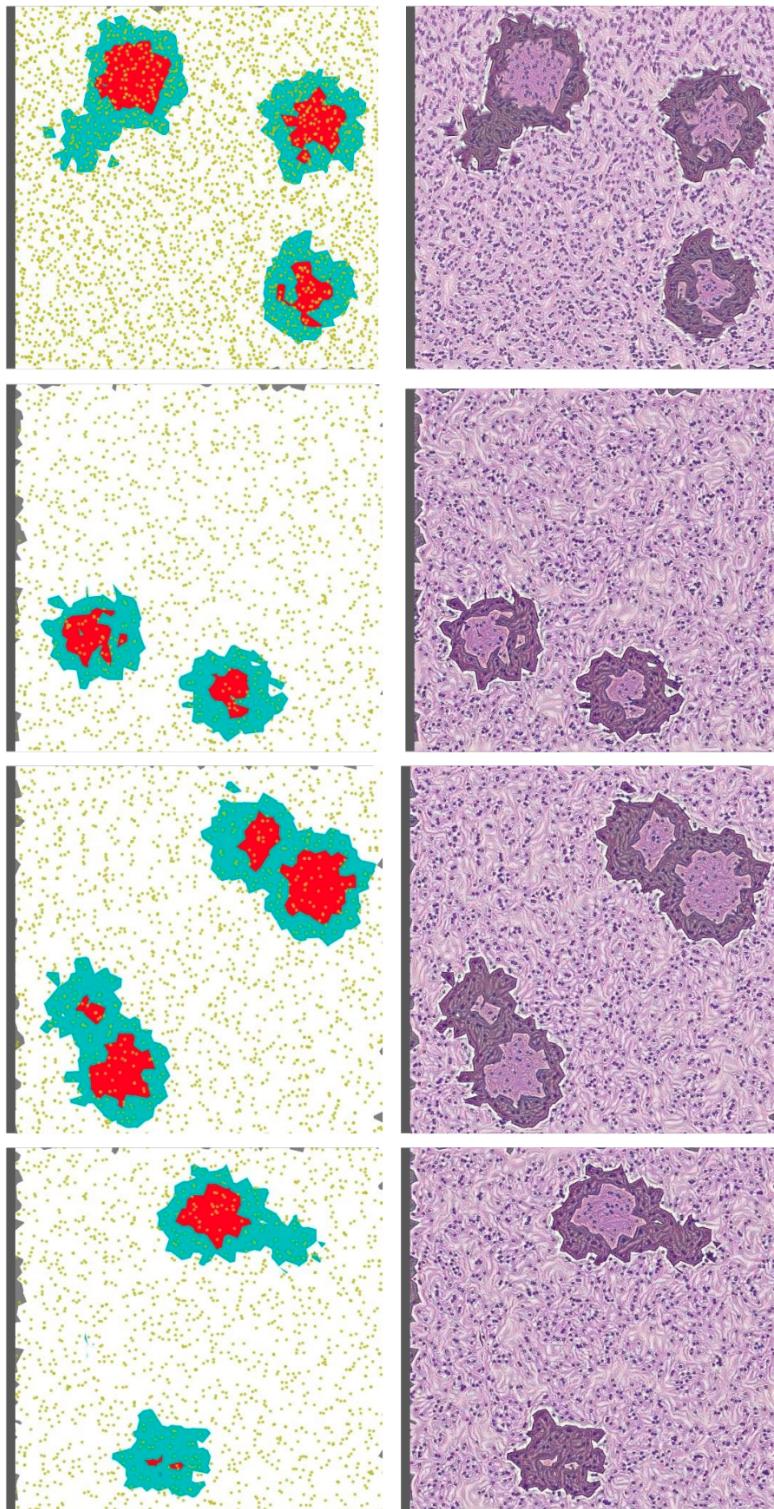


Figure 3.16: Small scale example of dataset produced with multiple automated applications of the generation tool on a ramified structure inspired to a pancreatic tissue model.

1595 Conclusions

1596 In this project, I face the problem of synthetic histological image generation for the pur-
1597 pose of training Neural Networks for the segmentation of real histological images. The
1598 manual analysis of histological ~~specimen~~-specimens is a complex, ~~time consuming~~time-consuming,
1599 and expensive task and nevertheless, it is a pillar of countless diagnostic techniques. Any
1600 form of support for this procedure hence is welcome and endorsed by the health-care
1601 system. In particular, in this work, I focus on the problem of histological specimens
1602 segmentation. The most advanced algorithms for image segmentation are based on Deep
1603 Learning and requires the training of extensive and complex neural networks. One of
1604 the toughest hurdles to overcome for the training of those NN is the abundance and the
1605 ~~quantity-quality~~ of pre-labeled examples of segmentation on real histological samples.
1606 The collection of hundreds of hand-labeled histological samples, with pixel-level preci-
1607 sion, is virtually impossible. This work thus proposes a methodology to generate, in
1608 a completely automatic way, synthetic pre-labeled histological-like images, that can be
1609 used as training material for a NN.

1610 The method I propose consists of the recreation of the traditional histological speci-
1611 mens' preparation, and it is based on the sectioning of a 3D virtual model of a region of
1612 histological tissue. The virtual 3D model of a region of a particular type of human tissue
1613 is built after physical and physiological considerations. The model is then subject to a
1614 virtual sectioning operation, which yields the synthetic sampling of the virtual tissue in
1615 which the histological identity of every pixel is perfectly known. This first image will
1616 act as a segmentation mask for a second, realistic image. In fact, on top of this first
1617 image are applied several aesthetical processing and refinements and the final product
1618 is the synthetic histological-like image. The pair made of the two images is ~~then perfect~~
1619 perfectly suitable for the supervised learning of a NN oriented toward the segmentation
1620 of histological images. The production of each pair of images is completely automatic
1621 and it does not require the intervention of any human operator, it is thus a scalable pro-
1622 cess that can produce a great abundance of images. The quality of the images is directly
1623 connected to the richness and the quality of the model. The perfect modelization of a
1624 region of tissue, let's say human pancreatic tissue, is by far out of reach for this, hence
1625 the richness and the fidelity of the produced images are inevitably lower than the real
1626 sample. Nevertheless, the quality of the produced images is sufficient to perform the

1627 preliminary phase of the training of a NN following a training strategy known as curriculum learning. This learning process consists of giving the NN a copious quantity of
1628 lower complexity level example in the first instance, reserving the few and sophisticated
1629 real hand-labeled histological samples for the finalization of the training.
1630

1631 The first chapter of this thesis is devoted to the contextualization of the present
1632 work. It is offered a description of how the histological samples are obtained after
1633 a tissue biopsy and how the digitalization process of the images works. ~~The reader
1634 is then introduced to the framework of Deep Learning, its fundamental aspect, and
1635 components. The concept of neural network is exposed and it is given the general idea
1636 underneath the training of a DL-based model. The chapter comes to an end with a
1637 general introduction to the segmentation task in computer vision, and how it is currently
1638 tackled with state-of-the-art algorithms.~~

1639 The second chapter collects all the details of every less common technical tool I used
1640 during the design of this project. A brief theoretical introduction is proposed for ev-
1641 ery item besides the thorough description of its practical use. ~~Some of the arguments
1642 touched by this chapter are quaternions, quasi-random number generation, Voronoi
1643 decomposition, style transfer neural networks.~~ In this chapter, a section is devoted
1644 to the description of a general methodology for computing the 2D section of an arbi-
1645 trary three-dimensional polyhedron. The algorithm here described has been devised and
1646 implemented all by my self, and then inserted in the workflow of the project. ~~As a
1647 conclusion for this chapter, I describe the working environment I built for developing
1648 this project and I mention all the code libraries I employed in my work~~ This is one of the
1649 key pieces for the automatization of the virtual tomography process, and it allows us to
1650 connect the three-dimensional model to the two-dimensional representation of a sampled
1651 section.

1652 The third chapter is the center of this work, and it contains the description of all
1653 the design choices, and the steps I followed for the development of the two human tissue
1654 models I propose: the first of pancreatic tissue and the second of dermal tissue. The
1655 ~~development has required the harmonization of many different technical aspects and
1656 mathematical tools. The first section is then first and second sections are~~ dedicated to
1657 the description of the ~~pancreatic tissue model, which passes through different steps: from
1658 a two-dimensional ramification taken as inspiration for the behavior of blood vessels to
1659 the complete three-dimensional model, with its subdivision in labeled cells. The second
1660 section is occupied by the description of the dermal tissue model, following the same
1661 spirit~~ two proposed 3D virtual tissue models, which consists of different steps. The third
1662 section instead contains a thorough description of the method to perform the sectioning
1663 onto a virtual model and how to process the resulting images. ~~It is necessary to perform
1664 several alternative processing and adjustment to achieve the desired aspect both for the
1665 segmentation mask image and for the~~ The development has required the harmonization
1666 of many different technical aspects and mathematical tools and it results in a general
1667 methodology for the generation of synthetic histological images. The process passes

1668 first through the building of the target tissue's structure in a virtual environment. This
1669 structure is then embedded in a three-dimensional space decomposition that subdivides
1670 the volume into individual cells. Those cells are labeled in correspondence to their role
1671 in the model, and their identity is then perfectly encaptured by the virtual tomography
1672 procedure. The sectioning process is responsible for the production of synthetic images,
1673 which are then conveyed toward an aesthetical enrichment pipeline specialized for the
1674 particular target tissue. The product of any application of this process is a pair made
1675 of a segmentation mask and the corresponding synthetic histological-like image **after the**
1676 **sectioning process.** This completely automatized procedure allows building arbitrary
1677 large datasets for the training of NN, without the intervention of any human operator.

1678 The method for the generation of datasets of synthetic images I propose in my thesis
1679 work is a self-supporting project and it is formally consistent. By the way, there are
1680 many possibilities for improvement and enrichment for the project. One first aspect to
1681 strengthen could be the richness of the models: adding more elements in the structure,
1682 and refining their representation of the tissue at the cellular level. This would lead
1683 to a better quality of the synthetic images, that would assist the training of NN in
1684 more and different applications. Another aspect that lends itself to improvements in the
1685 development of a dedicated style transfer NN targeting the histological texture transfer,
1686 which could lead to interesting signs of progress in the quality of image generation. There
1687 is also the intention to perform an actual attempt of NN training on the images produced
1688 with this process. This would complete conceptually the idea underneath the project
1689 and would be an excellent opportunity to detect weaknesses and to draw up possible
1690 lines of development. The repeated application of the generation method would allow
1691 the building of entire datasets suitable for the training of DL-based models.

₁₆₉₂

Bibliography

- ₁₆₉₃ [1] Abien Fred Agarap. Deep learning using rectified linear units (relu), 2018.
- ₁₆₉₄ [2] Salah Alheejawi, Mrinal Mandal, Hongming Xu, Cheng Lu, Richard Berendt, and
₁₆₉₅ Naresh Jha. Deep learning-based histopathological image analysis for automated
₁₆₉₆ detection and staging of melanoma. In *Deep Learning Techniques for Biomedical*
₁₆₉₇ and *Health Informatics*, pages 237–265. Elsevier, 2020.
- ₁₆₉₈ [3] J. Alsayednoor and P. Harrison. Evaluating the performance of microstructure
₁₆₉₉ generation algorithms for 2-d foam-like representative volume elements. *Mechanics*
₁₇₀₀ of *Materials*, 98:44 – 58, 2016.
- ₁₇₀₁ [4] Hani A Alturkistani, Faris M Tashkandi, and Zuhair M Mohammedsaleh. Histolog-
₁₇₀₂ ical stains: A literature review and case study. *Global Journal of Health Science*,
₁₇₀₃ 8(3):72, June 2015.
- ₁₇₀₄ [5] Yoshua Bengio, Jérôme Louradour, Ronan Collobert, and Jason Weston. Curricu-
₁₇₀₅ lum learning. In *Proceedings of the 26th Annual International Conference on Ma-*
₁₇₀₆ *chine Learning*, ICML '09, page 4148, New York, NY, USA, 2009. Association for
₁₇₀₇ Computing Machinery.
- ₁₇₀₈ [6] Jon Louis Bentley. Multidimensional binary search trees used for associative search-
₁₇₀₉ ing. *Commun. ACM*, 18(9):509517, September 1975.
- ₁₇₁₀ [7] R. W. Brockett. Robotic manipulators and the product of exponentials formula.
₁₇₁₁ In P. A. Fuhrmann, editor, *Mathematical Theory of Networks and Systems*, pages
₁₇₁₂ 120–129, Berlin, Heidelberg, 1984. Springer Berlin Heidelberg.
- ₁₇₁₃ [8] C. G. BROYDEN. The Convergence of a Class of Double-rank Minimization Algo-
₁₇₁₄ rithms 1. General Considerations. *IMA Journal of Applied Mathematics*, 6(1):76–90,
₁₇₁₅ 03 1970.
- ₁₇₁₆ [9] Bassem Ben Cheikh, Catherine Bor-Angelier, and Daniel Racoceanu. A model of
₁₇₁₇ tumor architecture and spatial interactions with tumor microenvironment in breast
₁₇₁₈ carcinoma. In Metin N. Gurcan and John E. Tomaszewski, editors, *Medical Imaging*

- 1719 2017: *Digital Pathology*, volume 10140, pages 73 – 80. International Society for
1720 Optics and Photonics, SPIE, 2017.
- 1721 [10] Anna Choromanska, Benjamin Cowen, Sadhana Kumaravel, Ronny Luss, Mattia
1722 Rigotti, Irina Rish, Brian Kingsbury, Paolo DiAchille, Viatcheslav Gurev, Ravi
1723 Tejwani, and Djallel Bouneffouf. Beyond backprop: Online alternating minimization
1724 with auxiliary variables, 2018.
- 1725 [11] Siddharth Singh Chouhan, Ajay Kaul, and Uday Pratap Singh. Image segmentation
1726 using computational intelligence techniques: Review. *Archives of Computational
1727 Methods in Engineering*, 26(3):533–596, February 2018.
- 1728 [12] Marius Cordts, Mohamed Omran, Sebastian Ramos, Timo Rehfeld, Markus En-
1729 zweiler, Rodrigo Benenson, Uwe Franke, Stefan Roth, and Bernt Schiele. The
1730 cityscapes dataset for semantic urban scene understanding. *CoRR*, abs/1604.01685,
1731 2016.
- 1732 [13] Angel Cruz-Roa, Ajay Basavanhally, Fabio Gonzlez, Hannah Gilmore, Michael Feld-
1733 man, Shridar Ganesan, Natalie Shih, John Tomaszewski, and Anant Madabhushi.
1734 Automatic detection of invasive ductal carcinoma in whole slide images with convo-
1735 lutional neural networks. *Progress in Biomedical Optics and Imaging - Proceedings
1736 of SPIE*, 9041, 02 2014.
- 1737 [14] Alessandro d'Agostino. Dataset generation for the training of neural networks ori-
1738 ented toward histological image segmentation, april 2020.
- 1739 [15] J. Deng, W. Dong, R. Socher, L. Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale
1740 hierarchical image database. In *2009 IEEE Conference on Computer Vision and
1741 Pattern Recognition*, pages 248–255, 2009.
- 1742 [16] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei. ImageNet: A Large-
1743 Scale Hierarchical Image Database. In *CVPR09*, 2009.
- 1744 [17] Mark Everingham, Luc Van Gool, Christopher KI Williams, John Winn, and An-
1745 drew Zisserman. The pascal visual object classes (voc) challenge. *International
1746 journal of computer vision*, 88(2):303–338, 2010.
- 1747 [18] Leon A. Gatys, Alexander S. Ecker, and Matthias Bethge. A neural algorithm of
1748 artistic style, 2015.
- 1749 [19] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural com-
1750 putation*, 9(8):1735–1780, 1997.
- 1751 [20] R. H. Hruban, A. Maitra, and M. Goggins. Update on pancreatic intraepithelial
1752 neoplasia. *Int J Clin Exp Pathol*, 1(4):306–316, Jan 2008.

- 1753 [21] Phillip Isola, Jun-Yan Zhu, Tinghui Zhou, and Alexei A. Efros. Image-to-image
1754 translation with conditional adversarial networks, 2016.
- 1755 [22] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization,
1756 2014.
- 1757 [23] Alex Krizhevsky. Learning multiple layers of features from tiny images. Technical
1758 report, 2009.
- 1759 [24] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with
1760 deep convolutional neural networks. In *Advances in neural information processing*
1761 *systems*, pages 1097–1105, 2012.
- 1762 [25] Johannes Kster and Sven Rahmann. Snakemakea scalable bioinformatics workflow
1763 engine. *Bioinformatics*, 28(19):2520–2522, 08 2012.
- 1764 [26] Aristid Lindenmayer. Mathematical models for cellular interactions in development
1765 ii. simple and branching filaments with two-sided inputs. *Journal of theoretical*
1766 *biology*, 18(3):300–315, 1968.
- 1767 [27] Daniel. Longnecker. Anatomy and histology of the pancreas, 2014.
- 1768 [28] Shervin Minaee, Yuri Boykov, Fatih Porikli, Antonio Plaza, Nasser Kehtarnavaz,
1769 and Demetri Terzopoulos. Image segmentation using deep learning: A survey, 2020.
- 1770 [29] Muhammad Niazi, Thomas Tavolara, Vidya Arole, Douglas Hartman, Liron Pan-
1771 tanowitz, and Metin Gurcan. Identifying tumor in pancreatic neuroendocrine neo-
1772 plasms from ki67 images using transfer learning. *PLOS ONE*, 13:e0195621, 04 2018.
- 1773 [30] Arild Nkland and Lars Hiller Eidnes. Training neural networks with local error
1774 signals, 2019.
- 1775 [31] Travis E Oliphant. *A guide to NumPy*, volume 1. Trelgol Publishing USA, 2006.
- 1776 [32] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gre-
1777 gory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban
1778 Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan
1779 Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith
1780 Chintala. Pytorch: An imperative style, high-performance deep learning library,
1781 2019.
- 1782 [33] Allan Pinkus. Approximation theory of the mlp model in neural networks. *Acta*
1783 *Numerica*, 8:143195, 1999.

- 1784 [34] Prabu Ravindran, Adriana Costa, Richard Soares, and Alex C Wiedenhoeft. Classification of cites-listed and other neotropical meliaceae wood images using convolutional neural networks. *Plant methods*, 14(1):1–10, 2018.
- 1785
- 1786
- 1787 [35] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. *CoRR*, abs/1505.04597, 2015.
- 1788
- 1789 [36] Juan Rosai. Why microscopy will remain a cornerstone of surgical pathology. *Laboratory Investigation*, 87(5):403–408, April 2007.
- 1790
- 1791 [37] Andrea Saltelli. Making best use of model evaluations to compute sensitivity indices. *Computer Physics Communications*, 145(2):280 – 297, 2002.
- 1792
- 1793 [38] Andrea Saltelli, Paola Annoni, Ivano Azzini, Francesca Campolongo, Marco Ratto, and Stefano Tarantola. Variance based sensitivity analysis of model output. design and estimator for the total sensitivity index. *Computer Physics Communications*, 181(2):259 – 270, 2010.
- 1794
- 1795
- 1796
- 1797 [39] Caglar Senaras, Muhammad Khalid Khan Niazi, Berkman Sahiner, Michael P. Pennell, Gary Tozbikian, Gerard Lozanski, and Metin N. Gurcan. Optimized generation of high-resolution phantom images using cGAN: Application to quantification of ki67 breast cancer images. *PLOS ONE*, 13(5):e0196846, May 2018.
- 1798
- 1799
- 1800
- 1801 [40] David F Shanno. Conditioning of quasi-newton methods for function minimization. *Mathematics of computation*, 24(111):647–656, 1970.
- 1802
- 1803 [41] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition, 2014.
- 1804
- 1805 [42] Sandro Skansi. *Introduction to Deep Learning: From Logical Calculus to Artificial Intelligence*. Springer Publishing Company, Incorporated, 1st edition, 2018.
- 1806
- 1807 [43] I.M. Sobol. Uniformly distributed sequences with an additional uniform property. *USSR Computational Mathematics and Mathematical Physics*, 16(5):236 – 242, 1976.
- 1808
- 1809
- 1810 [44] I.M Sobol. Global sensitivity indices for nonlinear mathematical models and their monte carlo estimates. *Mathematics and Computers in Simulation*, 55(1):271 – 280, 2001. The Second IMACS Seminar on Monte Carlo Methods.
- 1811
- 1812
- 1813 [45] M Titford. The long history of hematoxylin. *Biotechnic & Histochemistry*, 80(2):73– 78, 2005.
- 1814

- 1815 [46] Pauli Virtanen, Ralf Gommers, Travis E. Oliphant, Matt Haberland, Tyler Reddy,
1816 David Cournapeau, Evgeni Burovski, Pearu Peterson, Warren Weckesser, Jonathan
1817 Bright, Stéfan J. van der Walt, Matthew Brett, Joshua Wilson, K. Jarrod Millman,
1818 Nikolay Mayorov, Andrew R. J. Nelson, Eric Jones, Robert Kern, Eric Larson,
1819 CJ Carey, İlhan Polat, Yu Feng, Eric W. Moore, Jake VanderPlas, Denis Laxalde,
1820 Josef Perktold, Robert Cimrman, Ian Henriksen, E. A. Quintero, Charles R Harris,
1821 Anne M. Archibald, Antônio H. Ribeiro, Fabian Pedregosa, Paul van Mulbregt,
1822 and SciPy 1.0 Contributors. SciPy 1.0: Fundamental Algorithms for Scientific
1823 Computing in Python. *Nature Methods*, 17:261–272, 2020.
- 1824 [47] Georges Voronoi. Nouvelles applications des paramètres continus à la théorie des
1825 formes quadratiques. premier mémoire. sur quelques propriétés des formes quadra-
1826 tiques positives parfaites. *Journal für die reine und angewandte Mathematik (Crelles
1827 Journal)*, 1908:102 – 97.