

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



# <sup>14</sup> Abstract

<sup>15</sup> The project is inspired by an actual problem of timing and accessibility in the analysis of  
<sup>16</sup> histological samples in the health-care system. In this project, I face the problem of syn-  
<sup>17</sup> thetic histological image generation for the purpose of training Neural Networks for the  
<sup>18</sup> segmentation of real histological images. The collection of real histological human-labeled  
<sup>19</sup> samples is a very time consuming and expensive process and often is not representative of  
<sup>20</sup> healthy samples, for the intrinsic nature of the medical analysis. The method I propose  
<sup>21</sup> is based on the replication of the traditional specimen preparation technique in a virtual  
<sup>22</sup> environment. The first step is the creation of a 3D virtual model of a region of the target  
<sup>23</sup> human tissue. The model should encapture all the key features of the tissue, and the  
<sup>24</sup> richer it is the better will be the yielded result. The second step is to perform a sampling  
<sup>25</sup> of the model through a virtual tomography process, which produces a first completely  
<sup>26</sup> labeled image of the section. This image is then processed with different tools to achieve  
<sup>27</sup> a histological-like aspect. The most significant aesthetical adjustment is given by the  
<sup>28</sup> action of a style transfer neural network that implants the typical histological visual tex-  
<sup>29</sup> ture on the synthetic image. This procedure is presented in detail for two specific models  
<sup>30</sup> of human tissue: one of pancreatic tissue and one of dermal tissue. The two resulting  
<sup>31</sup> images compose a pair of images perfectly suitable for a supervised learning technique.  
<sup>32</sup> The generation process is completely automatized and does not require the intervention  
<sup>33</sup> of any human operator, hence it can be used to produce arbitrary large datasets. The  
<sup>34</sup> synthetic images are inevitably less complex than the real samples and they offer an  
<sup>35</sup> easier segmentation task to solve for the NN. However, the synthetic images are very  
<sup>36</sup> abundant, and the training of a NN can take advantage of this feature, following the  
<sup>37</sup> so-called curriculum learning strategy.

# <sup>38</sup> Table of Contents

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

63	3.1.1	2D Ramification . . . . .	54
64	3.1.2	Expansion to 3D . . . . .	56
65	3.1.3	Subdivision in Cells . . . . .	57
66	3.1.4	Cells Identity Assignment . . . . .	60
67	3.2	Dermal Tissue Model . . . . .	61
68	3.3	Synthetic Images Production . . . . .	64
69	3.3.1	Sectioning Process . . . . .	64
70	3.3.2	Appearance Makeover . . . . .	66
71	<b>Conclusions</b>		<b>73</b>
72	<b>Bibliography</b>		<b>76</b>



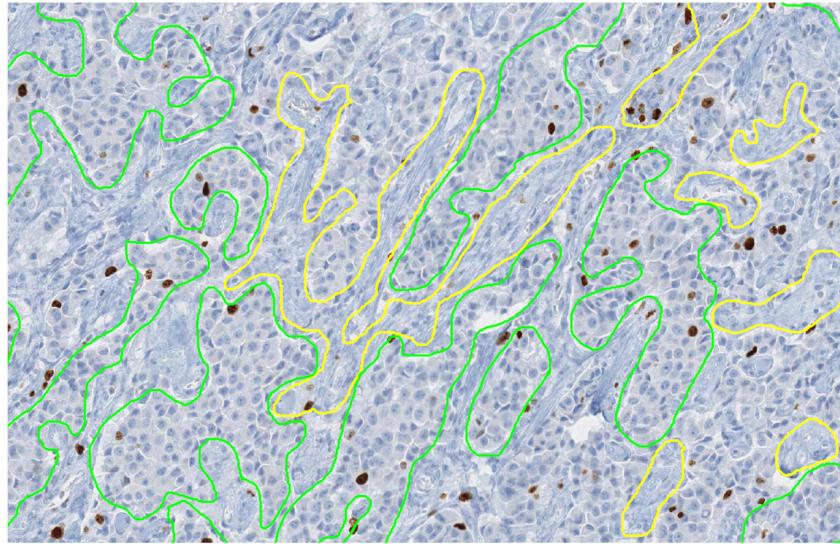
# <sup>74</sup> Introduction

<sup>75</sup> In the last decades, the development of Machine Learning (ML) and Deep Learning (DL)  
<sup>76</sup> techniques has contaminated every aspect of the scientific world, with interesting results  
<sup>77</sup> in many different research fields. The biomedical field is no exception to this and a  
<sup>78</sup> lot of promising applications are taking form, especially as Computer-Aided Detection  
<sup>79</sup> (CAD) systems which are tools for the support for physicians during the diagnostic  
<sup>80</sup> process. Medical doctors and the healthcare system in general collect a huge amount of  
<sup>81</sup> data from patients during all the treatment, screening, and analysis activities in many  
<sup>82</sup> different shapes, from anographical data to blood analysis to clinical images.

<sup>83</sup> In fact in medicine, the study of images is ubiquitous and countless diagnostic proce-  
<sup>84</sup> dures rely on it, such as X-ray imaging (CAT), nuclear imaging (SPECT, PET), Magnetic  
<sup>85</sup> resonance, and visual inspection of histological specimens after biopsies. The branch of  
<sup>86</sup> artificial intelligence in the biomedical field that handles image analysis to assist physi-  
<sup>87</sup> cians in their clinical decisions goes under the name of Digital Pathology Image Analysis  
<sup>88</sup> (DPIA). In this thesis work, I want to focus on some of the beneficial aspects intro-  
<sup>89</sup> duced by DPIA in the histological images analysis and some particular issues in the  
<sup>90</sup> development of DL models able to handle this kind of procedure.

<sup>91</sup> Nowadays the great majority of analysis of histological specimens occurs through  
<sup>92</sup> visual inspection, carried out by highly qualified experts. Some analysis, as cancer  
<sup>93</sup> detection, requires the ability to distinguish if a region of tissue is healthy or not with high  
<sup>94</sup> precision in very wide specimens. This kind of procedure is typically very complex and  
<sup>95</sup> requires prolonged times of analysis besides substantial economic efforts. Furthermore,  
<sup>96</sup> the designated personnel for this type of analysis is often limited, leading to delicate issues  
<sup>97</sup> of priority assignment while scheduling analysis, based on the estimated patient's clinical  
<sup>98</sup> development. Some sort of support to this analysis procedure is therefore necessary.

<sup>99</sup> The problem of recognizing regions with different features within an image and de-  
<sup>100</sup> tect their borders is known in computer vision as the segmentation task, and it's quite  
<sup>101</sup> widespread with countless different applications, allowing a sort of automatic image in-  
<sup>102</sup> terpretation. In ML the segmentation problem is usually faced as a supervised task,  
<sup>103</sup> hence the algorithm in order to be trained properly requires an appropriate quantity  
<sup>104</sup> of pre-labeled images, from which learn the rules through which distinguish different  
<sup>105</sup> 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].

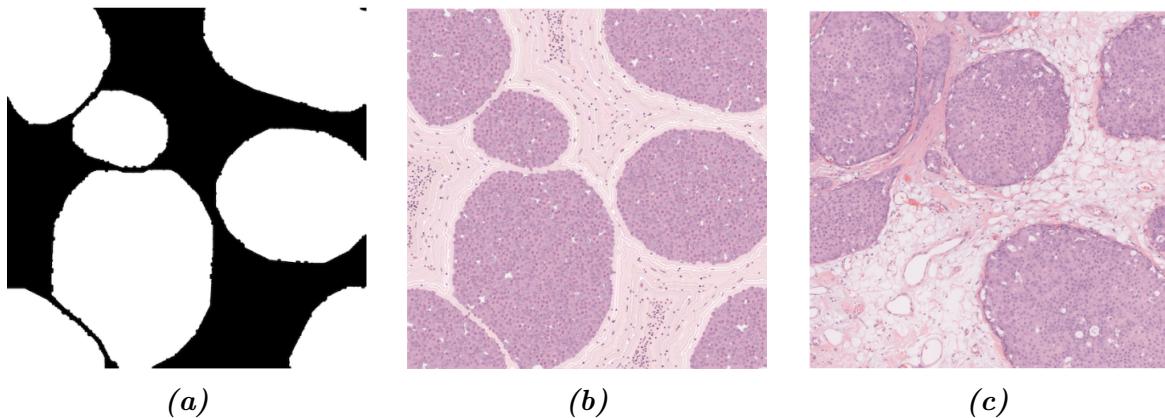
106 plication, as would be the one on histological images, would require a lot of starting  
107 material, previously analyzed from the same qualified expert encharged of the visual  
108 inspection mentioned before. A human operator thus is required to manually track the  
109 boundaries, for example,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

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

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

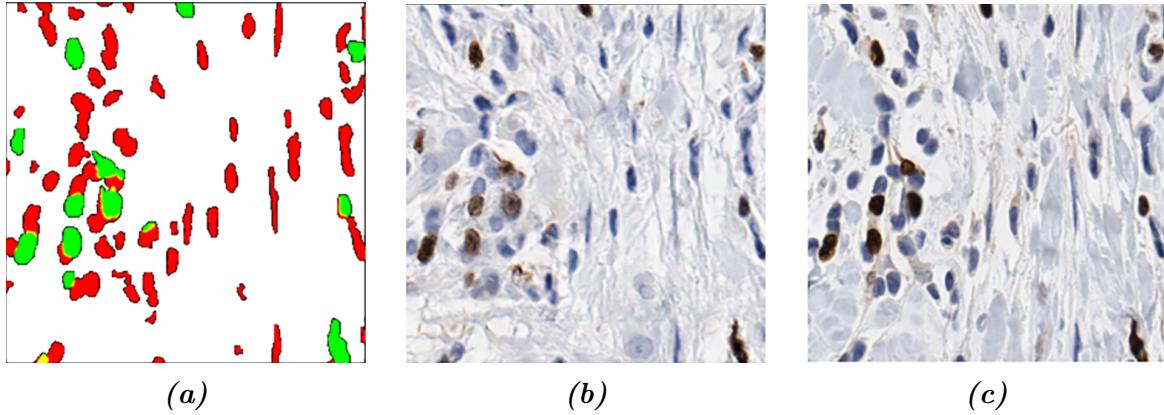
128 different issues. Here, I want to make an overview of some other interesting works on  
129 the generation of synthetic histological images, which have followed completely different  
130 paths and strategies from mine.

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



**Figure 2:** 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 [9].

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

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

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

### 189 1. Theoretical Background .

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

### 196 2. Technical Tools for Model Development .

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

### 207 3. Tissue Models Development .

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



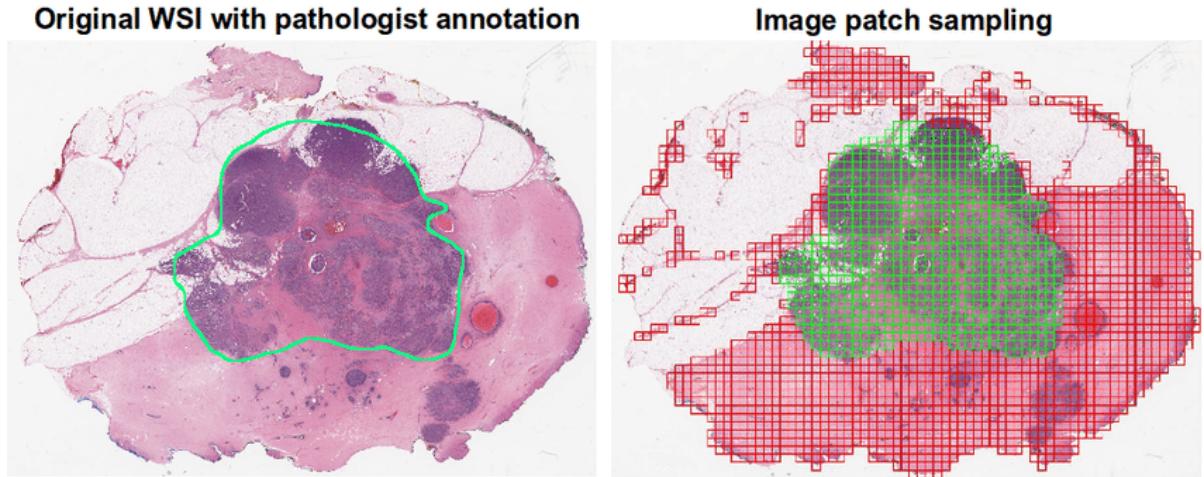
<sup>215</sup> **Chapter 1**

<sup>216</sup> **Theoretical Background**

<sup>217</sup> In this first chapter, I will depict the theoretical context of the work. Section 1.1 will  
<sup>218</sup> be dedicated to histological images, and the different techniques used to prepare the  
<sup>219</sup> samples to analyze. Histological images recover a fundamental role in medicine and are  
<sup>220</sup> the pillar of many diagnosis techniques. This discipline borns traditionally from the  
<sup>221</sup> optical inspection of the tissue slides using a microscope, and it is gradually developing  
<sup>222</sup> and improving with the advent of computers and digital image processing. It is important  
<sup>223</sup> tough to understand how the samples are physically prepared, the final target of this  
<sup>224</sup> work is in fact the virtual reconstruction of this process. The section comes to an end  
<sup>225</sup> with an introduction to the anatomy of the two particular tissues under the attention  
<sup>226</sup> of this project, a brief introduction to the most common neoplasms that interest the  
<sup>227</sup> pancreas and skin, and their shreds of evidence at the histological level. In section 1.2 I  
<sup>228</sup> will introduce the Deep Learning framework and describe how a Neural Network works  
<sup>229</sup> and actually learns. The most advanced techniques for the automatic image processing  
<sup>230</sup> implement Deep Learning algorithm, and understanding the general rules behind this  
<sup>231</sup> discipline is crucial for a good comprehension of this work. In section 1.3 I will discuss in  
<sup>232</sup> particular the problem of image segmentation and how it is tackled with different Neural  
<sup>233</sup> Network architectures, showing what it is the state of the art of this research field.

<sup>234</sup> **1.1 Histological Images Digitalization**

<sup>235</sup> Modern histopathology is essentially based on the careful interpretation of microscopic  
<sup>236</sup> images, with the intention of correctly diagnose patients and to guide therapeutic de-  
<sup>237</sup> cisions. In the last years, thanks to the quick development of scanning techniques and  
<sup>238</sup> image processing, the discipline of histology have seen radical improvements: the main  
<sup>239</sup> of which undoubtedly is the passage from the microscope's oculars to the computer's  
<sup>240</sup> screen. This digitalization process has brought several advantages, that were previously  
<sup>241</sup> 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.

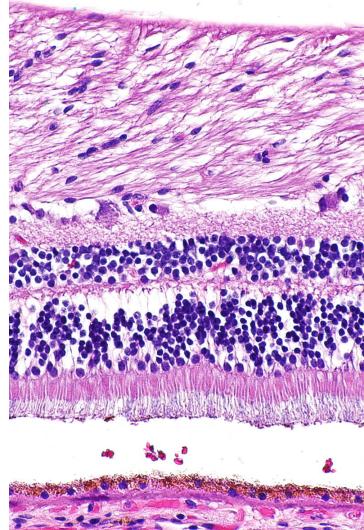
In the field of Digital Pathology (DP) an essential concept in image understanding is the magnification factor, which indicates the scale of representation of the image and allows dimension referencing. This factor is usually indicated as the magnification power of the microscope's lenses used during 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  $\mu m$  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

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

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

291 The analysis of histological images usually consists of detecting the different com-  
292 ponents in the samples and to recognize their arrangement as a healthy or pathological  
293 pattern. It is necessary to recognize every sign of the vitality of the cells, evaluating  
294 the state of the nucleus. There are many additional indicators to consider the pres-  
295 ence of inflammatory cells or tumoral cells. Furthermore, samples which are taken from  
296 different part of the human body present completely different characteristics, and this  
297 increase greatly the complexity of the analysis. A reliable examination of a sample thus  
298 requires a careful inspection made by a highly qualified expert. The automatization of  
299 this procedure would be extremely helpful, giving an incredible boost both in timing and  
300 accessibility. However, this is not a simple task and in section 1.3.1 I will show some  
301 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.

### 302    1.1.1 Slides Preparation for Optic Microscopic Observation

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

- 309    1) **Fixation** is carried out immediately after the removal of the sample to be observed.  
310       It is used to immobilize and preserve the sample permanently in as life-like state  
311       as possible. It can be performed immersing the biological material in a formalin  
312       solution or by freezing, so immersing the sample in a tissue freezing medium which  
313       is then cooled in liquid nitrogen.
- 314    2) **Embedding** if the sample has been stabilized in a fixative solution, this is the sub-  
315       sequent step. It consists in hardening the sample in a paraffin embedding medium,  
316       in order to be able to carry out the sectioning. It is necessary to dehydrate the  
317       sample beforehand, by replacing the water molecules in the sample with ethanol.
- 318    3) **Sectioning** Sectioning is performed using microtomy or cryotomy. Sectioning is an  
319       important step for the preparation of slides as it ensures a proper observation of the  
320       sample by microscopy. Paraffin-embedded samples are cut by cross-section, using a

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

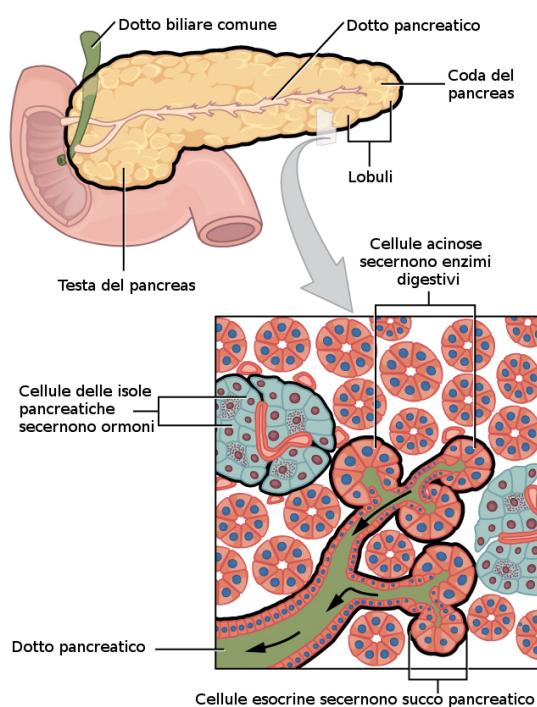
327 **4) Staining** Staining increases contrasts in order to recognize and differentiate the dif-  
328 ferent components of the biological material. The sample is first deparaffinized and  
329 rehydrated so that polar dyes can impregnate the tissues. The different dyes can  
330 thus interact with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

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

### 339 **1.1.2 Pancreas Microanatomy and Tumoral Evidences**

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

355 There are many tumoral diseases interest the pancreas, they represent one of the main  
356 causes of decease for cancer in occidental countries, and its incidence rate in Europe  
357 and the USA has risen significantly in the last decades. There are two main kinds  
358 of pancreatic neoplasms: endocrine and exocrine pancreatic tumors. The endocrine



**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.

359 pancreas neoplasms derive from the cells constituting the Langherans' islets and are  
360 typically divided into functioning and non-functioning ones, depending on the capability  
361 of the organs to secrete hormones. Those diseases might be benign or malignant and  
362 they can have a different degree of aggressivity. Exocrine pancreatic tumors tough are  
363 the most frequent ones. Among those, the great majority of episodes are of malignant  
364 neoplasms, and in particular, the ductal adenocarcinoma is the most frequent form of  
365 disease, responsible alone for the 95% of the cases. From a macroscopic point of view,  
366 those tumors are characterized by an abundant fibrotic stroma<sup>1</sup>, which can represent  
367 over 50% of the tumor's mass and it is responsible for the hard-ligneous tumor's aspect.  
368 This disease is frequently followed by pancreatitis episodes.

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

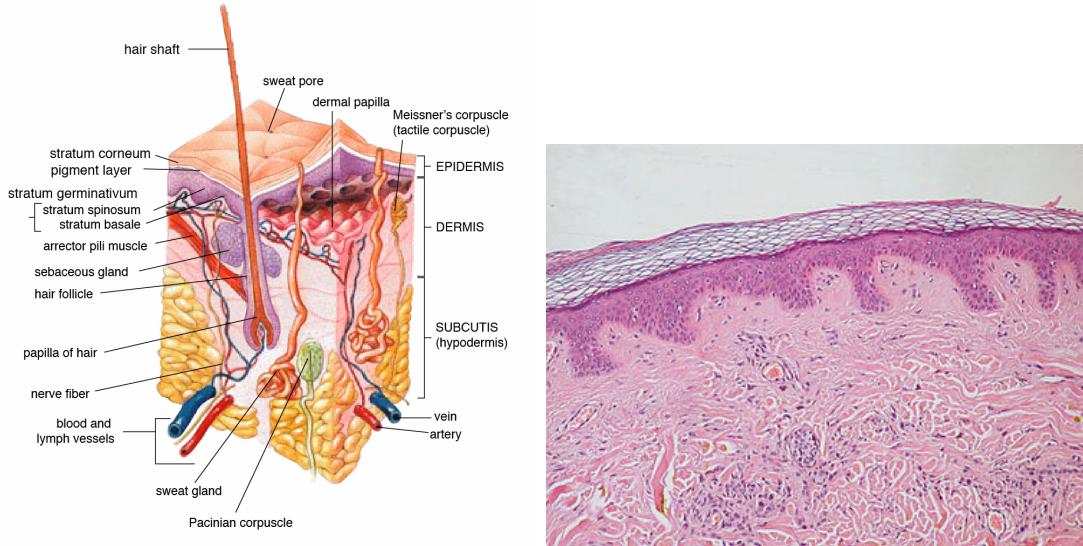
378 During the microscopical analysis of a histological sample, there are other interesting  
379 markers to be evaluated, as a specific set of lesions which goes under the name of PanIN  
380 (Pancreatic Intraepithelial Neoplasia) and could not be appreciated with other diagnostic  
381 techniques.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

### 385 1.1.3 Skin Microanatomy and Tumoral Evidences

386 Skin is the layer of soft, flexible outer tissue covering the body of a vertebrate animal,  
387 with the three main functions of protection, regulation, and sensation. Mammalian  
388 skin is composed of two primary layers: the epidermis, which provides waterproofing  
389 and serves as a barrier to infection, and the dermis, which serves as a location for the  
390 appendages of skin. The epidermis is composed of the outermost layers of the skin. It  
391 forms a protective barrier over the body's surface, responsible for keeping water in the  
392 body and preventing pathogens from entering, and is a stratified squamous epithelium,  
393 composed of proliferating basal and differentiated suprabasal keratinocytes. The dermis  
394 is the layer of skin beneath the epidermis that consists of connective tissue and cushions

---

<sup>1</sup>Stroma 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.

395 the body from stress and strain. The dermis provides tensile strength and elasticity to  
 396 the skin through an extracellular matrix composed of collagen fibrils, microfibrils, and  
 397 elastic fibers, embedded in hyaluronan and proteoglycans.

398 Melanoma is the most aggressive form of skin tumor. It is the second most fre-  
 399 quent tumors in men under 50 years, and the third for women under 50 years, with over  
 400 than 12.300 cases in Italy every year. Melanoma is considered nowadays a multifactorial  
 401 pathology, which originates from the interaction between genetic susceptibility and en-  
 402 vironmental exposure. The most important environmental risk factor is the intermittent  
 403 solar exposure, for the genotoxic effect of ultraviolet rays on the skin. Different studies  
 404 show also a strong correlation between the total number of nevi on the skin and the  
 405 incidence of melanoma.

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

411 **Superficial Spreading Melanoma :** This is the most common subtype, and it rep-  
 412 resents alone more than the 75% of all melanoma cases. These neoplasms show  
 413 relatively large malignant melanocytes, inflammatory cells (epithelioid), and an  
 414 abundance of cytoplasm.

415    **Lentigo Maligna Melanoma** : It typically arises in photodamaged skin regions. Ne-  
416       plasmatic melanocytes are of polygonal shape, with hyperchromatic nuclei, and  
417       they are followed by a reduction in the cytoplasm.

418    **Acral lentiginous melanoma** : It typically arises on palmar, plantar, subungual, or  
419       mucosal surfaces. Melanocytes are usually arranged along the dermal-epidermal  
420       junction. The progression of this type of melanoma is characterized by the pres-  
421       ence of large junctional nests of atypical melanocytes, which are extended and  
422       hyperchromatic, with a shortage of cytoplasm.

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

426       The careful examination of histological samples extracted from the tissue under anal-  
427       ysis is the most important step for the assessment of the actual form of neoplasms.

## 428 1.2 Introduction to Deep Learning

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

### 437 1.2.1 Perceptrons and Multilayer Feedforward Architecture

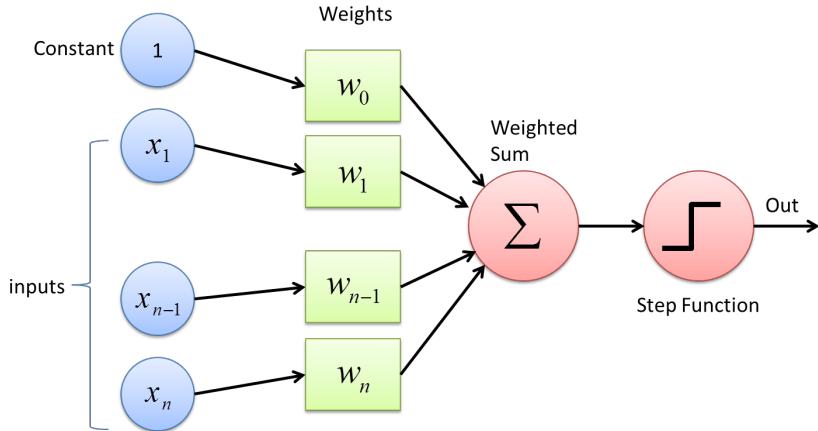
438 Like other artificial learning techniques, DL models aim to *learn* a relationship between  
439 some sort of input and a specific kind of output. In other words, approximating nu-  
440 mercially the function that processes the input data and produces the desired response.  
441 For example, one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

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

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

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

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

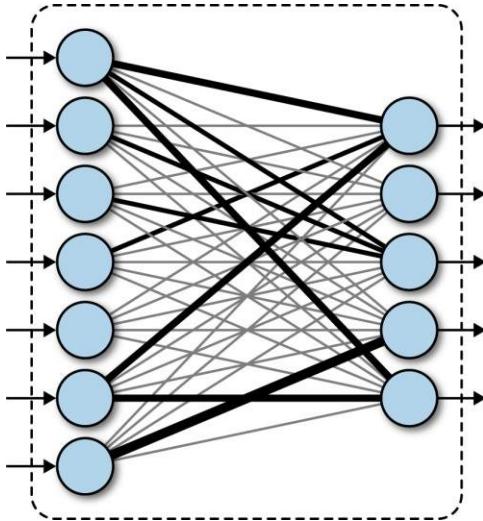


**Figure 1.5:** Schematic picture of a single layer perceptron. The input vector is linearly combined with the bias factor and sent to an activation function to produce the numerical "binary" output.

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

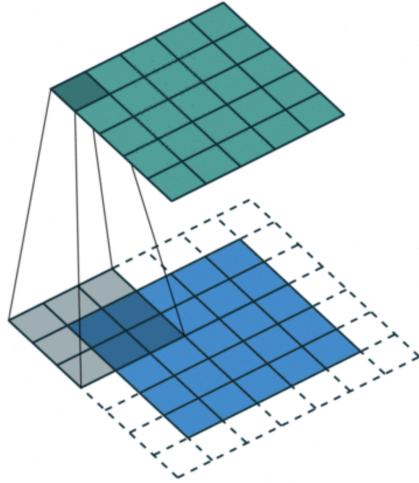


**Figure 1.6:** Schematic representation of a fully connected (or dense) layer. Every neuron from the first layer is connected with every output neuron. The link thickness represent the absolute value of the combination weight for that particular value.

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

#### 490 1.2.2 Training Strategies in Deep Learning

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

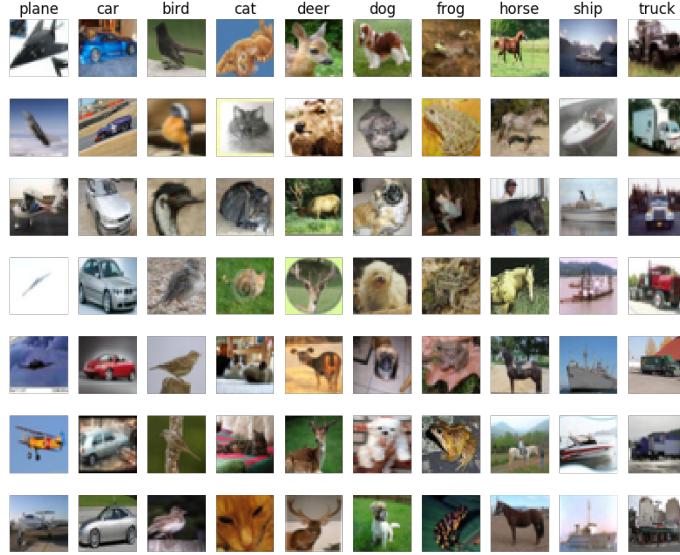


**Figure 1.7:** Schematic representation of a convolutional layer. The input data are processed by a window kernel that slides all over the image. This operation can recreate almost all the traditional computer vision techniques, and can overcome them, creating new operations, which would be unthinkable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

506 shapes between pure supervised and pure unsupervised learning, based on the actual  
507 availability of data and specific limitations to the individual task.

508 An interesting mention in this regard should be made about *semi-supervised* learning,  
509 which is typically used in biomedical applications. This learning technique combines a  
510 large quantity of unlabeled data during training with a limited number of pre-labeled  
511 example. The blend between data can produce a considerable improvement in learning  
512 accuracy and leading to better results with respect to pure unsupervised techniques.  
513 The typical situation of usage of this technique is when the acquisition of labeled data  
514 requires a highly trained human agent (as an anatomopathologist) or a complex physical  
515 experiment. The actual cost of building entire and suitable fully labeled training sets  
516 in these situations would be unbearable, and semi-supervised learning comes in great  
517 practical help.

518 Another important training technique which worth mentioning is the so-called *cur-  
519 riculum learning* [5]. This is a learning technique inspired by the typical learning curve  
520 human being and animal, that are used to face problems of always increasing difficulty  
521 while learning something new or a new skill. The example presented to the Neural Net-  
522 work are not randomly presented but organized in a meaningful order which illustrates  
523 gradually more concepts, and gradually more complex ones. The experiments show that  
524 through curriculum learning significant improvements in generalization can be achieved.  
525 This approach has both an effect on the speed of convergence of the training process to a



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

526 minimum and, in the case of non-convex criteria, on the quality of the local minima ob-  
 527 tained. This technique is of particular interest for this work: the generated images, which  
 528 will be shown in section 3.3 , will offer a segmentation task much less complex respect to  
 529 the analysis of real histological tissue. In the optical of training a DL-based model the  
 530 abundantly produced images can be used for the preliminary phase of training, setting  
 531 aside the more complex and more valuable hand-labeled images for the finalization of  
 532 the training process.

### 533 1.2.3 Training Algorithms - Error Back-Propagation

534 A good example of supervised problems tough is the classification of images. Let's  
 535 assume we have a whole dataset of pictures of different objects (as cats, dogs, cars, etc.)  
 536 like the CIFAR10 [23] dataset. This famous dataset is made of over  $60K$  labeled colored  
 537 images  $32 \times 32$  divided into 10 categories of objects as shown in Figure 1.8. We could be  
 538 interested in the creation of a NN able to assign at every image its belonging class. This  
 539 NN could be arbitrarily complex but it certainly will take as input a  $32 \times 32 \times 3$  RGB  
 540 image and the output will be the predicted class. A typical output for this problem  
 541 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:

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

566 where  $E$  is the error function, and  $\eta$  is the *learning coefficient*, that modulate the effect of  
567 learning through all the training process. This iterative procedure is applied completely  
568 to each image in the training set several times, each time the whole dataset is reprocessed  
569 is called an *epoch*. The great majority of the dataset is exploited in the training phase  
570 to keep running this trial and error process and just a small portion is left out (typically  
571 10% of the data) for a final performance test.

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

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

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

607 the network, and it allows straightforward comparison with other models' performances,  
608 trained for the same task. All this training procedure is coherently customized to every  
609 different application, depending on which the problem is posed as supervised or not and  
610 depending on the more or less complex network's architecture. The leitmotif is always  
611 finding a suitable loss function that quantifies how well the network does what it has  
612 been designed to do and trying to minimize it, operating on the parameters that define  
613 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.

### 614 1.3 Deep Learning-Based Segmentation Algorithms

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

626 A relatively easy segmentation problem, and one of the first to be tackled, could  
 627 be distinguishing an object from the background in a grey-scale image, like in Figure  
 628 1.10. The easiest technique to perform segmentation in this kind of problem is based on  
 629 thresholding. Thresholding is a binarization technique based on the image's grey-level  
 630 histogram: to every pixel with luminosity above that threshold is assigned the color  
 631 *white*, and vice versa the color *black*. However, this is a very primitive and fallacious, yet  
 632 very fast method, and it manages poorly complex images or images with un-uniformity  
 633 in the background.

634 A lot of other traditional techniques improve this first segmentation method [11].  
 635 Some are based on the object's edges recognition, exploiting the sharp change in lumi-  
 636 nosity typically in correspondence of the boundary of a shape. Other techniques exploit  
 637 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.

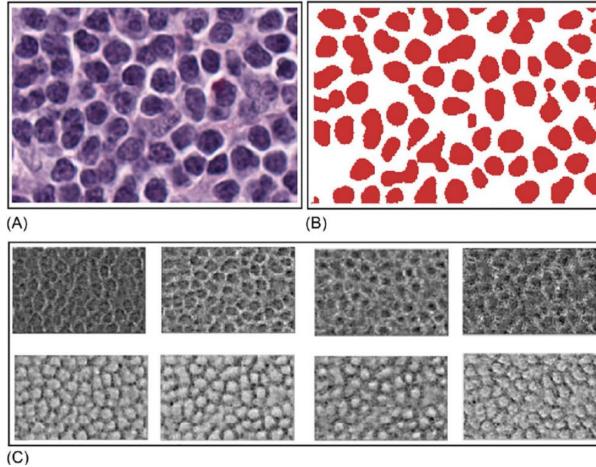
638 scattered on the image, and the regions corresponding to the objects in the image are  
639 grown to incorporate adjacent pixels with similar properties.

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

### 655 1.3.1 State of the Art on Deep Learning Segmentation

656 In a similar way to many other traditional tasks, also for segmentation, there has been  
657 a thriving development lead by the diffusion of deep learning, that boosted the perfor-  
658 mances resulting in what many regards as a paradigm shift in the field [28].

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



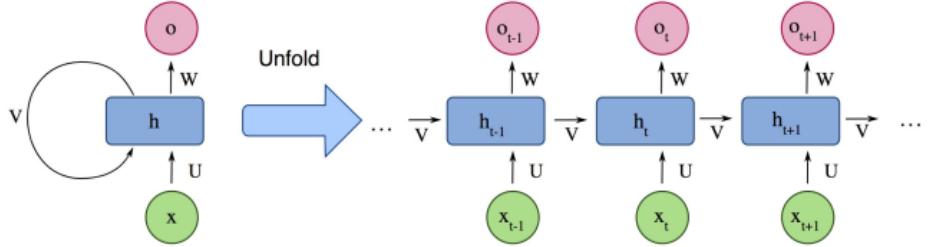
**Figure 1.11:** (A) An extract from an histological samples, used as input image for the model. (B) The exact segmentation mask. (C) Example of accentuated features during the training: (1-4) for back-ground recognition, (5-8) for nuclei detection. From [2].

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.

## Recurrent Neural Networks (RNNs) and the LSTM

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

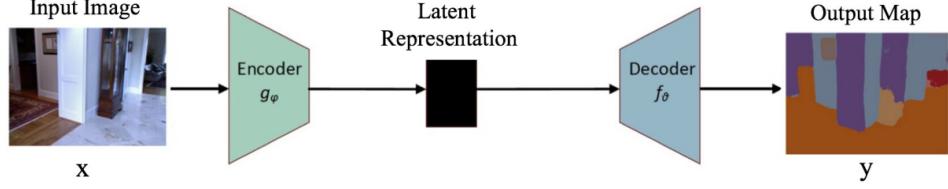
685 LSTM architecture includes three gates (input gate, output gate, forget gate),  
 686 which regulate the flow of information into and out from a memory cell, which  
 687 stores values over arbitrary time intervals.

#### 688 Encoder-Decoder and Auto-Encoder Models

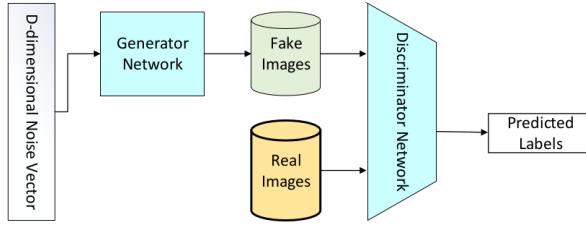
689 Encoder-Decoder models try to learn the relation between an input and the corre-  
 690 sponding output with a two steps process. The first step is the so-called *encoding*  
 691 process, in which the input  $x$  is compressed in what is called the *latent-space* rep-  
 692 resentation  $z = f(x)$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

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

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

709 **Convolutional Neural Networks (CNNs)**

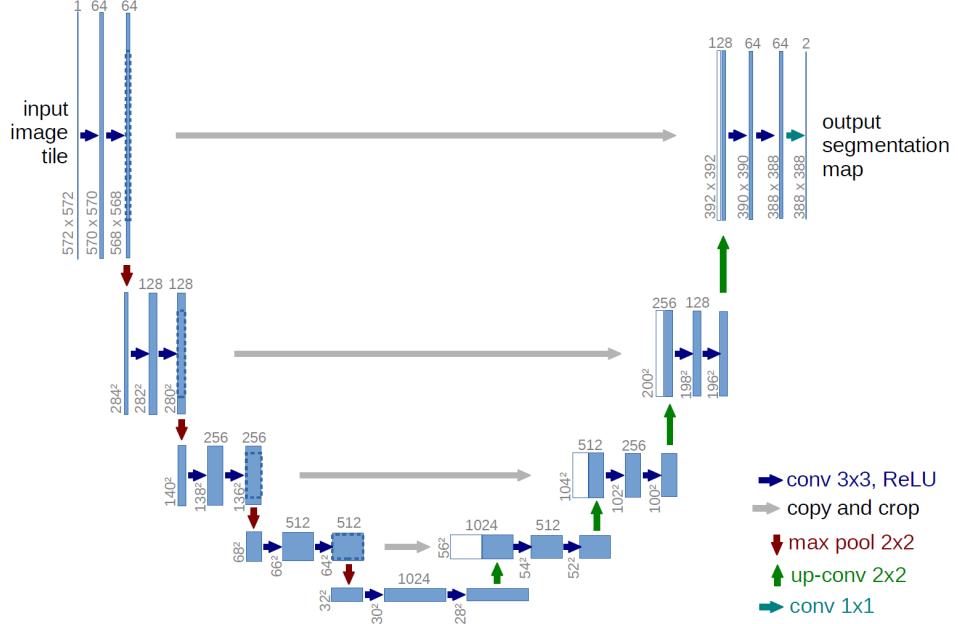
710 As stated before CNNs are a staple choice in image processing DL applications.  
711 They mainly consist of three types of layers:

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

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

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

746 A recent example of a practical application of a CNN to histological images could  
747 be found in [29]. In this work the Inception v3 is trained on hand-labeled samples of

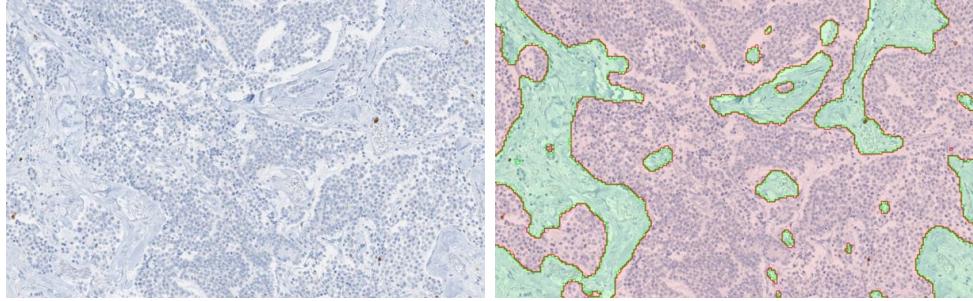


**Figure 1.15:** Scheme of the typical architecture of a U-net NN. This particular model was firstly proposed by Ronneberger et al. in [35].

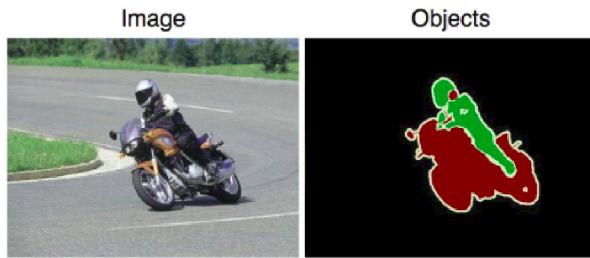
pancreatic tissue (like in Figure 1) to recognize tumoral regions from healthy ones in a pancreatic tissue specimen treated with Ki67 staining. The Inception v3 [34] network is a deep convolutional network developed by Google, trained for object detection and image classification on the ImageNet dataset [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 green the healthy ones. This work is based on a technique called **transfer learning**, which consists 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 only to image-to-image problems, like segmentation problems. There are a lot of widely used datasets, but I want to mention just a few of them to give the idea of their typical



**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].

766 characteristics.

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

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

781 As the last mention, I would report the ImageNet project [15], which is a large visual  
 782 database designed for use in visual object recognition software research. It consists of

783 more than 14 million images that have been hand-annotated by the project to indicate  
784 what objects are pictured and in at least one million of the images, bounding boxes are  
785 also provided. ImageNet contains more than 20,000 categories of objects. Since 2010,  
786 the ImageNet project runs an annual software contest, the ImageNet Large Scale Visual  
787 Recognition Challenge (ILSVRC), where software programs compete to correctly classify  
788 and detect objects and scenes. This kind of competition is very important for the research  
789 field, as it inspires and encourages the development of new models and architectures.

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

799 **Chapter 2**

800 **Technical Tools for Model  
801 Development**

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

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

822 **2.1 Quaternions**

823 Quaternions are, in mathematics, a number system that expands to four dimensions the  
824 complex numbers. They have been described for the first time by the famous mathemati-

cian William Rowan Hamilton in 1843. This number system define three independent *imaginary* units  $\mathbf{i}, \mathbf{j}, \mathbf{k}$  as in (2.1), which allows the general representation of a quaternion  $\mathbf{q}$  is (2.2) and its inverse  $\mathbf{q}^{-1}$  (2.3) where  $a, b, c, d$  are real numbers:

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

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

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

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

$$\begin{aligned} \mathbf{i} \cdot 1 &= 1 \cdot \mathbf{i} = \mathbf{i}, & \mathbf{j} \cdot 1 &= 1 \cdot \mathbf{j} = \mathbf{j}, & \mathbf{k} \cdot 1 &= 1 \cdot \mathbf{k} = \mathbf{k} \\ \mathbf{i} \cdot \mathbf{j} &= \mathbf{k}, & \mathbf{j} \cdot \mathbf{i} &= -\mathbf{k} \\ \mathbf{k} \cdot \mathbf{i} &= \mathbf{j}, & \mathbf{i} \cdot \mathbf{k} &= -\mathbf{j} \\ \mathbf{j} \cdot \mathbf{k} &= \mathbf{i}, & \mathbf{k} \cdot \mathbf{j} &= -\mathbf{i}. \end{aligned} \quad (2.4)$$

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

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

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

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

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

where in (2.6) we can clearly see a generalization of the Euler's formula for the exponential notation of complex numbers, which hold for quaternions. It can be shown

844 that the application of the rotation represented by  $\mathbf{q}$  on an arbitrary 3D vector  $\vec{v}$  should  
845 be easily expressed as:

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

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

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

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

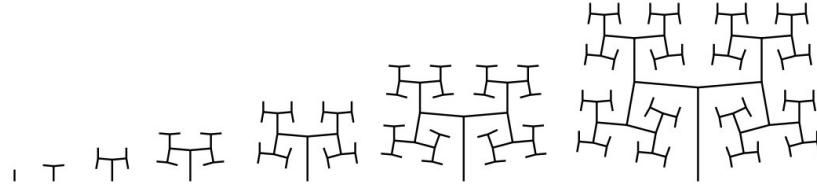
853 Given this property, quaternions are indeed widely used in all sorts of applications  
854 of digital 3D space design, as for simulations and videogame. The position of an object  
855 in the space in simulations is generally given by the application of several independent  
856 rotations, typically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

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

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

## 867 2.2 Parametric L-Systems

868 Lindenmayer systems, or simply L-systems, were conceived as a mathematical theory of  
869 plant development [26] in 1968 by Aristid Lindenmayer. Successively, a lot of geometri-  
870 cal interpretations of L-systems were proposed to make them a versatile instrument for  
871 modeling the morphology typical of plants and other organic structures. As a biologist,  
872 Lindenmayer studied different species of yeast and fungi and worked the growth patterns  
873 of various types of bacteria (e.g. as the *cyanobacteria Anabaena catenula*). The main  
874 purpose for which L-systems were devised was to allow a formal description of the devel-  
875 opment of simple multicellular living organisms. Subsequently, the potentiality of these  
876 systems was expanded to describe higher-order plants and complex branching structures.



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

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

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

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

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

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

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

$$\begin{aligned}
& axiom : A & (2.10) \\
& rule_1 : A \rightarrow F(1)[+A][-A] \\
& rule_2 : F(s) \rightarrow F(s \cdot R)
\end{aligned}$$

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

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

## 2.3 Voronoi Tassellation

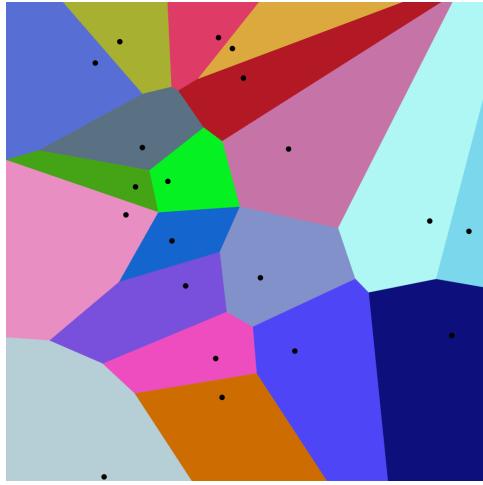
Voronoi diagrams, or Voronoi decompositions, are space-partitioning systems, which divides an  $n$ -dimensional Euclidian space into sub-regions depending on the proximity to a given set of objects. More precisely, given an  $n$ -dimensional space and  $m$  starting point  $p_1, \dots, p_m$  inside it,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

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

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

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 homogenously distributed the final distribution will appear as a simple regular lattice, while



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

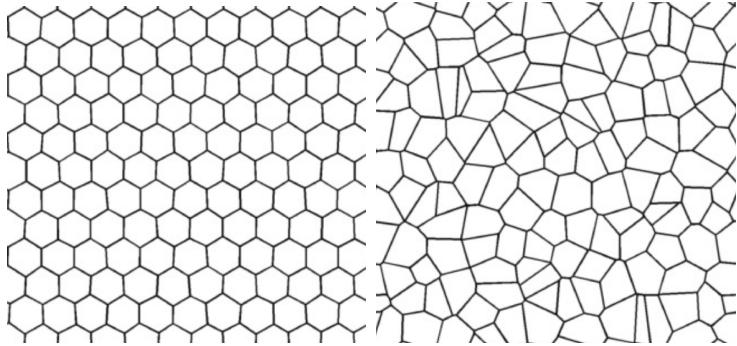
923 a completely random distribution of points in the space will provide a decomposition in  
 924 cells with very different shapes and volumes, as shown in Figure 2.3. Interesting results  
 925 concerning points from a semi-random distribution will be shown in section 3.1, which  
 926 leads to decomposition with a good richness in shapes but with the desired homogeneity  
 927 in volumes.

928 The Voronoi decomposition has been of great interest in this project for the division  
 929 of a 3D space in subregions, to recreate the spatial distribution of cells in a sample of  
 930 human tissue, as will be shown in section 3.1. The formal definition of Voronoi regions  
 931 (2.11) ensures the convexity of each decomposition's tassel, which in three-dimensional  
 932 space would be adjacent convex polyhedrons. Every tassel of the decomposition will  
 933 be represented by a bounded 3-dimensional convex hull<sup>1</sup>, with except for those most  
 934 external cells which are unbounded and requires special attention while using.

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

---

<sup>1</sup>See 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].

## 942 2.4 Saltelli Algorithm - Random Number Generation

943 As mentioned in section 2.3, in this project there was the need for quasi-random number  
 944 generation for the production of Voronoi tessellations. Quasi-random sequences (or low-  
 945 discrepancy sequences) are patterns of numbers that emulate the behavior of uniform  
 946 random distributions but have a more homogeneous and quick coverage of the sampling  
 947 domain, which provides an important advantage in applications as in quasi-Monte Carlo  
 948 integration techniques, as shown in Figure 2.4. In computer science there is not any  
 949 possibility of recreating *true* random sequences, hence any stochasticity is completely  
 950 deterministic in its essence even if produced by very chaotic processes<sup>2</sup>. Indeed, every al-  
 951 gorithm for random number generation is completely repeatable given its starting status.  
 952 Quasi-random sequences are completely deterministic too but implement more *regular*,  
 953 well-behaved algorithms.

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

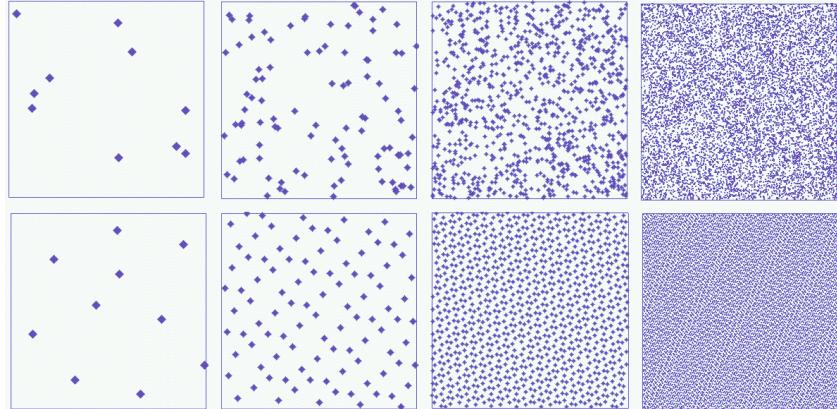
$$s_{n+1} = (s_n + \alpha) \bmod 1, \quad (2.12)$$

956 which for every seed element  $s_0$  and real parameter  $\alpha$  produced completely different  
 957 sequences.

958 In the bottom line of Figure 2.4 is clearly visible the good and homogeneous coverage  
 959 of the sampling domain, although it is strongly visible a regular pattern between points,  
 960 which does not convey an *organic* sensation at all. However, increasing the complexity of

---

<sup>2</sup>A 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.

961 our very simple starting model 2.12 it is possible to overcome this *artificial* appearance  
 962 of sampled points and to produce very good samples.

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

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

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

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

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

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

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

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

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

978 Given a non-negative integer  $t$ , a  $(t, s)$ -sequence in base  $b$  is an infinite sequence of  
979 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  
980 a  $(t, m, s)$ -net in base  $b$ .

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

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

## 988 2.5 Planar Section of a Polyhedron

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

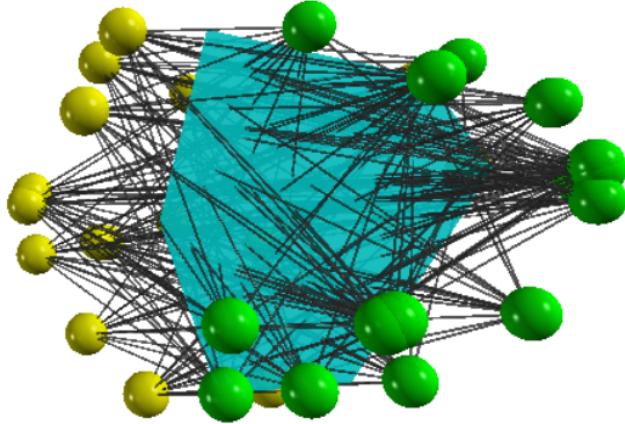
1001 Given a convex polyhedron with  $n$  vertices and a sectioning plane  $p$ , let  $V$  be the set  
1002 of all the vertices and  $f_p(\vec{x})$  the equation defining the plane. The algorithm is defined  
1003 by the following steps:

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:

$$\begin{aligned} A &= \{\vec{v} \in V \mid f_p(\vec{v}) \geq 0\} \\ B &= \{\vec{v} \in V \mid f_p(\vec{v}) \leq 0\} \end{aligned} \tag{2.16}$$

1004 If any of the two subsets turns out to be empty the plane  $p$  does not intersect the  
1005 polyhedron, and the section is empty.  $A$  and  $B$  are represented in different colors  
1006 in Figure 2.5.

2. Detect, and *draw*, any possible line that crosses two points respectively from  $A$  and  
 $B$ . If  $n_A$  and  $n_B$  are the numbers of points above and below the plane then there



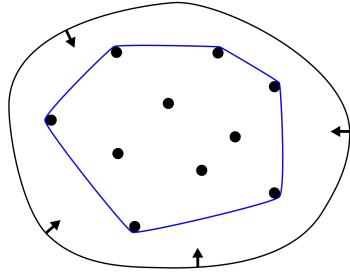
**Figure 2.5:** In this picture is shown an example of the application of the algorithm for the planar section on a polyhedron. The vertices are divided into two groups, with different colors yellow and green. All the possible lines between any couple of vertices picked from the two classes are drawn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

will be  $n_A \times n_B$  possible lines. In Figure 2.5 all the lines between the two classes of points are drawn in dark gray.

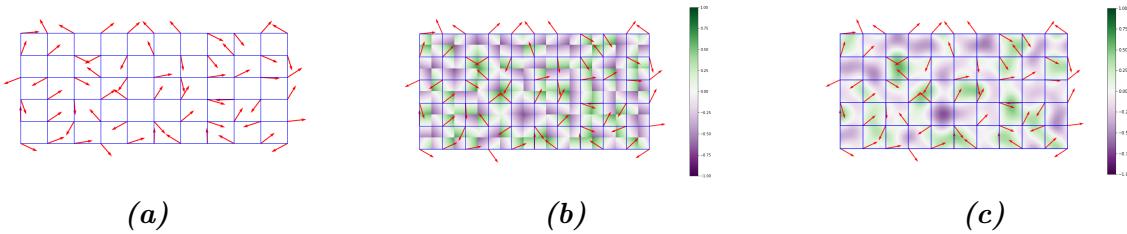
- 1011 3. Detect  $P$ , the set of all the points from the intersection between the  $n_A \times n_B$  lines
- 1012 from the previous step and the sectioning plane  $p$ . All these points will lie on the
- 1013 same plane, within the boundaries of the polygonal section.
- 1014 4. The final polygon is then yielded by computing the convex hull of the points in
- 1015  $P$ . The convexity of the starting polyhedron in fact ensures the convexity of any
- 1016 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 (In Figure 2.6 is shown the so-called “rubber band effect”), and in 3 dimensions it is the smallest convex polyhedron containing a set of points in the space.

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



**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.

## 1026 2.6 Perlin Noise

1027 Perlin noise is a widely used form of noise in computer graphics, which mimics very  
 1028 well natural and smooth fluctuations around a constant value. It has been developed by  
 1029 Ken Perlin in 1983, and it is now the staple tool for giving texture to object in virtual  
 1030 modelization, often considered the *salt* of computer graphics texturization. The Perlin  
 1031 noise is a gradient-based algorithm defined on grid discretization of a  $n$ -dimensional  
 1032 space. The algorithm involves three subsequent steps:

- 1033 1. The first step is to discretize the  $n$ -dimensional space in a regular lattice: the  
 1034 dimension of the grid will impact heavily on the scale of the noise. As in Figure  
 1035 2.7a at every node of the grid is assigned a randomly oriented  $n$ -dimensional unitary  
 1036 gradient vector. This is the preliminary setup which will allow the computation of  
 1037 the actual noise function in every point of the space.
- 1038 2. Given the candidate point  $\vec{x}$  between the grid nodes onto which evaluate the noise  
 1039 there are  $2^n$  nearest grid nodes. For each one of these  $2^n$  nodes, it is evaluated the

1040 distance vector from  $\vec{x}$  as the offset between the two points. Then it is computed  
1041 the dot product between every pair made of nearby gradient vectors and the offset  
1042 vector. This operation should be thought of as made on every point in the lattice,  
1043 as in Figure 2.7b, where at every point of the grid is represented just one of the  
1044  $2^2 = 4$  series of dot products.

- 1045 3. The final step is the interpolation between the  $2^n$  series of dot products. To per-  
1046 form the interpolation usually is used a function with vanishing first degree (and  
1047 preferably also second degree) derivative in correspondence of the  $2^n$  grid nodes <sup>3</sup>.  
1048 This means that the noise function will pass through zero at every node and have  
1049 a gradient equal to the pre-computed grid node gradient. These properties give  
1050 Perlin noise its characteristic spatial scale and smoothness.

1051 In general, the final result of the algorithm is a smooth function with a random-  
1052 like behavior that mimics really well an organic appearance, like in Figure 2.8, with  
1053 fluctuation around the value 0, with amplitude  $\in [0; 1]$ . The surface in Figure 2.8 has been  
1054 produced plotting in 3D the results of the function `pnoise2` from the library `noise`, which  
1055 offers a tool for the production of different type of noise. In order to put this module  
1056 in practical usage, some adjustments were required. The particular function `pnoise2`  
1057 simply yields the value of the Perlin noise surface in correspondence to a single  $(x, y)$  point  
1058 in the plane in a deterministic way. There was not the possibility to generate different  
1059 whole Perlin surfaces every run. To overcome this limitation I made a vectorized<sup>4</sup> version  
1060 of `pnoise2` able to evaluate the function over an arbitrary wide grid of points expressed as  
1061 the set of all the pairs of coordinates  $(x, y)$  of the grid's nodes. In this way a single call to  
1062 the function is able to produce the entire surface covering the grid, in the form of a single  
1063 NumPy 3D-array. Furthermore, to recover the possibility of generating always different  
1064 surfaces as in a random generation, I inserted an offset coordinate  $(x_O, y_O)$ , which moves  
1065 in the plane the origin of the surface generation. This pair of offset coordinates then acts  
1066 also as a *seed* in the generation, allowing to completely recreate previously generated  
1067 material in a controlled way.

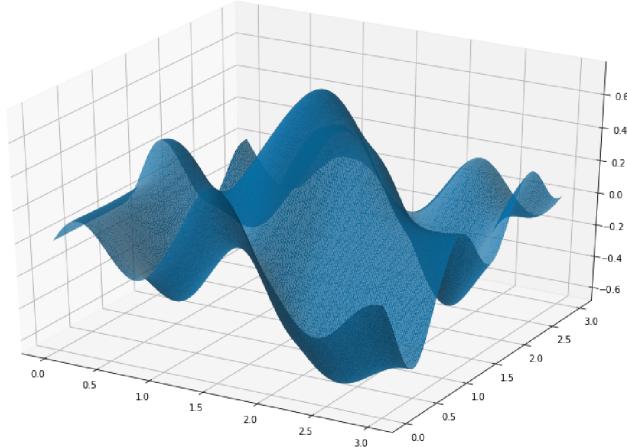
## 1068 2.7 Style-Transfer Neural Network

1069 Style-Transfer Neural Networks are common models, able to create new hybrid images  
1070 implanting the visual style from an image preserving the visual content of another image.

---

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

<sup>4</sup>In 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 transforming a function which acts on a single value (or pair of values) to a function able to perform on a suitable data structure is called *vectorization*, and its the 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  $\in [0; 1]$ .

1071 The two images necessary for the algorithm are called *style* image  $S$  and *content* image  
 1072  $C$ , and the resulting *styled* picture  $X$ , as in Figure 2.9.

1073 There are many different tested and comparable architectures to compute this kind  
 1074 of algorithm. In my work I decided to use in particular the procedure described in [18],  
 1075 using the PyTorch ecosystem to implement the necessary code.

1076 The backbone of the architecture is the VGG-19 network, which is a convolutional  
 1077 neural network 19 layers deep, as in Figure 2.10. This huge model has been pre-trained  
 1078 on over a million images from the ImageNet database [16], for the classification into  
 1079 over than 1000 classes of objects. As a result, the network has learned rich feature  
 1080 representations for a wide range of images. The best (and conceptually the only) way to  
 1081 load a pre-trained model is to load the ordered set of weights that define the network and  
 1082 to initialize an empty module with those values. This is the perfect start for creating a  
 1083 style transfer network, which requires a further and briefer training phase, to completely  
 1084 customize the network.

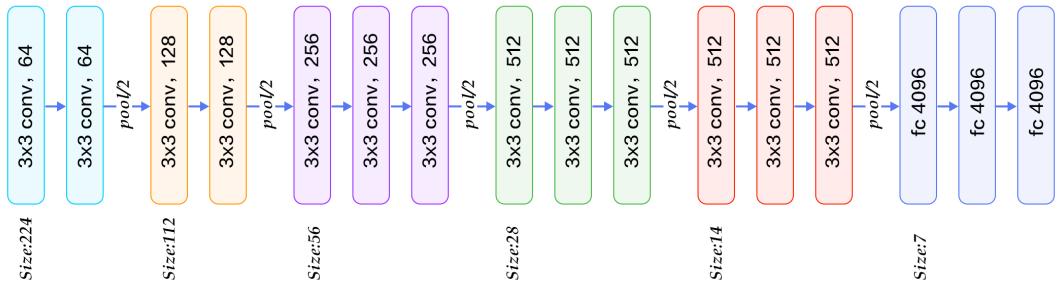
1085 The key ingredient for finalizing the model is to insert some little but fundamental  
 1086 modifications and to extend the training on the pair of input images. This final training  
 1087 should be aware of the *concepts* of the visual style and visual content of the image, and  
 1088 the operation should try to preserve them both. This is usually done minimizing two  
 1089 new loss function, computed between the staring image and the produced image:

#### 1090 Content Loss

1091 The content loss is a function that represents a weighted version of the content  
 1092 distance for an individual layer. The most commonly used function to evaluate the  
 1093 preservation of content between two images is the simple Mean Squared Error as



**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  $224 \times 224$  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.

1094 in equation (1.5). It can be computed between any couple of same-sized object,  
 1095 hence also between the results of the same feature maps on the images  $X$  and  $C$   
 1096 at the same layer  $L$ :

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

1097 In order to evaluate this content loss, it is necessary to insert a custom transparent<sup>5</sup>  
 1098 layer directly after the convolution layer(s) that are being used to compute the  
 1099 content distance.

### 1100 Style Loss

1101 The concept of *style* loss function is the true novelty introduced by [18]. This loss  
 1102 function is implemented similarly to the content loss module, as it will act as a  
 1103 transparent layer in the network. The computation of the style loss requires in  
 1104 advance the evaluation of the Gram matrix  $G_{XL}$  at a certain layer  $L$ . A Gram  
 1105 matrix is a result of multiplying a given matrix by its transposed matrix. In this  
 1106 case, the matrix to multiplicate is a reshaped version of the feature maps  $F_{XL}$ :  
 1107  $\hat{F}_{XL}$ , a  $K \times N$  matrix, where  $K$  is the number of feature maps at layer  $L$  and  $N$   
 1108 is the length of any vectorized feature map  $F_{XL}^k$ . Furthermore, the Gram matrix  
 1109 must be normalized by dividing each element by the total number of elements in the  
 1110 matrix. The style distance is now computed using the mean square error between  
 1111  $G_{XL}$  and  $G_{SL}$ :

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

1112 After the appropriate insertion of the loss-function evaluator layers, one last piece for  
 1113 finalizing the model is the right choice of the gradient descent optimizer. As in [18] and  
 1114 according to the Deep Learning community the optimizer which suite best this role is the  
 1115 Limited Memory-BFGS [8],[40]. L-BFGS is an iterative algorithm in the family of quasi-  
 1116 Newton methods that approximates the Broyden-Fletcher-Goldfarb-Shanno algorithm  
 1117 (BFGS) using a limited amount of computer memory, and it is a popular choice when  
 1118 estimating parameters of a non-linear differentiable scalar function.

1119 The final phase of the training process thus makes run the optimizer for hundreds of  
 1120 epochs on  $X$ ,  $C$ , and  $S$ , and reduce the two loss functions values acting on the network's  
 1121 parameters. After the fine-tuning of the weights, the hybrid image is produced, as in  
 1122 Figure 2.9.

## 1123 2.8 Working Environment

1124 In this section, I will briefly describe the machine I used to develop my project and the  
 1125 working environment I built. All the work has been done on my personal computer,

---

<sup>5</sup>A 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.

1126 mounting a **GNU/linux** operating system, in particular the 18.04 LTS **Ubuntu** version.  
1127 The computer mounts an Intel i7 core, 8 Gb of RAM beside a 2Gb NVidia 940MX GPU.  
1128 All the Python libraries have been installed and harmonized in a virtual environment  
1129 mounting **Python 3.7.6**. All the code produced during the development, the images,  
1130 and the data produced have been collected in a devoted repository on GitHub [14], which  
1131 is freely available.

1132 As a conclusion for this chapter I will recollect all the references to the different  
1133 Python libraries I used during the development of this work:

1134 **NumPy:** NumPy is the pillar of every scientific computation-oriented library. Is the most  
1135 spread library for heavy multidimensional numerical computation, and it offers a  
1136 broad variety of tools like random number generators, and pre-implemented linear  
1137 algebra utilities [31].

1138 **SciPy:** The SciPy library is one of the core packages that make up the SciPy stack.  
1139 It provides many user-friendly and efficient numerical routines, such as routines  
1140 for numerical integration, interpolation, optimization, linear algebra, and statistics  
1141 [46]. Two modules in particular form this libraries have covered an essential  
1142 role in this project: the `SciPy.spatial.Voronoi` module for the computation  
1143 of the 3D Voronoi decomposition, as mentioned in section 2.3, and the  
1144 `SciPy.spatial.ConvexHull` module for the computation of 3D and 2D convex  
1145 hulls (section 2.5).

1146 **PyTorch:** PyTorch is a rich ecosystem of tools and libraries geared toward Machine  
1147 Learning and Deep Learning. The application of the style-transfer NN described  
1148 in section 2.7 has required the use of this framework [32].

1149 **SALib:** The SALib is a library which collects many tools for the Sensitivity Analysis of  
1150 parameters. In particular, the `SALib.saltelli` submodel was used for the quasi-  
1151 random numerical sampling in a three-dimensional box, and it has been described  
1152 in section 2.4.

1153 **pnoise2:** pnoise2 contains many tools for the production of specific types of noise.  
1154 The module `noise` was tweaked for the production of two-dimensional Perlin noise  
1155 surfaces, and it has been introduced in section 2.6.

1156 **pyquaternion:** The `pyquaternion` library provides a framework for handling quaternions.  
1157 It has been widely used in section 2.1 for the design of three-dimensional  
1158 ramifications for handling multiple spatial rotations.

<sub>1159</sub> **Chapter 3**

<sub>1160</sub> **Tissue Models Development**

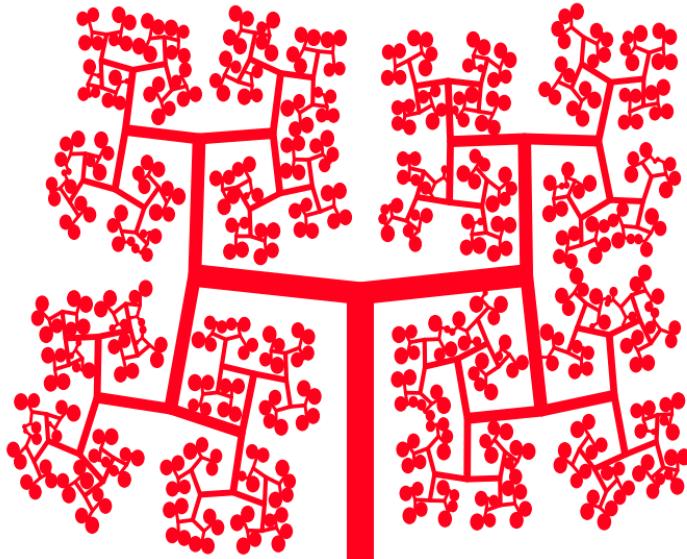
<sub>1161</sub> The main goal of the present work, as stated before, is to recreate a three-dimensional  
<sub>1162</sub> virtual model of histological tissue as faithfully as possible and then, to perform planar  
<sub>1163</sub> sectioning on it to emulate virtually the traditional histological specimen preparation  
<sub>1164</sub> procedure. The creation of a model of such complex structures is definitely a high-  
<sub>1165</sub> level problem, and it has required a careful designing, made of subsequent stages of  
<sub>1166</sub> improvements. In section 3.1, I will describe all the necessary steps to create the model  
<sub>1167</sub> of a small region of pancreatic tissue, while in section 3.2 I will expose the steps I  
<sub>1168</sub> followed to build a model of dermal tissue. In section 3.3, instead, I will show the  
<sub>1169</sub> resulting synthetic images from the sectioning process performed on both the models  
<sub>1170</sub> and all the enrichments and processing necessary to give them the most realistic look I  
<sub>1171</sub> was able to recreate.

<sub>1172</sub> **3.1 Pancreatic Tissue Model**

<sub>1173</sub> In this first attempt of modelization from scratch the main focus was put on reflecting  
<sub>1174</sub> only the main structural features on the virtual specimens. Given the pancreatic tissue's  
<sub>1175</sub> organization, described back in section 1.1.2, the first features I decided to emphasize on  
<sub>1176</sub> were: 1) The iterative (with a fractal-like behavior) ramification of blood vessels for the  
<sub>1177</sub> irrigation of glandular acinus, 2) The space-filling distribution of acinus in the tissue, in  
<sub>1178</sub> fact, we expect a homogeneous density in the organ and to not see *holes* at all inside it.  
<sub>1179</sub> In this section I will describe step by step all the process I followed to create the model  
<sub>1180</sub> of a portion of pancreatic tissue, and all the interesting pitfalls I overcame.

<sub>1181</sub> **3.1.1 2D Ramification**

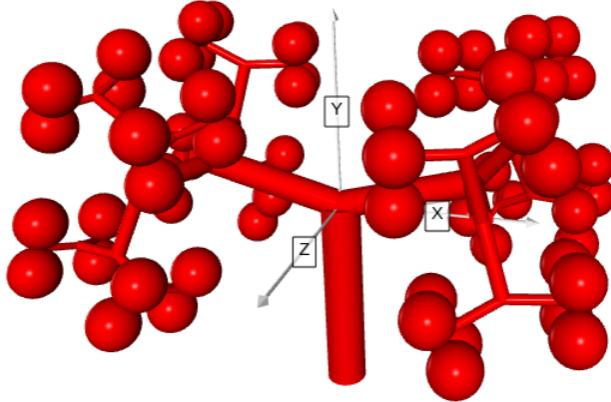
<sub>1182</sub> The first step was taken in two dimensions, and it was the choice of the right *structure*  
<sub>1183</sub> to emulate the ramification of blood vessels in pancreatic tissue. The choice fell on a



**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.

particular parametric L-system, as the one shown in Figure 2.1, in section 2.2. This structure is made of an iterative bifurcation of gradually shorter segments, with an angle of  $\pm 85^\circ$  respect the main direction. For a start I added some features to give a more realistic look to the structure, which are all well represented in Figure 3.1:

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



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

1201 To produce the specific image in Figure 3.1 I used a particular setting of the tool  
 1202 described in section 2.2, which has a greatly wider range of customization and could be  
 1203 used to create many other different structures to the need.

### 1204 3.1.2 Expansion to 3D

1205 The successive step I followed was to expand this structure in three dimensions and fill  
 1206 the space in each of the three directions. The idea to evolve the structure in Figure 3.1 is  
 1207 simply to twist of  $90^\circ$  the ramification at every junction point, in such a way to exit the  
 1208 previous belonging plane. However, putting into practice this development has not been  
 1209 easy. The organization of the structure in a 3D space requires an appropriate system  
 1210 of reference for handling subsequent rotations in three dimensions. The best option for  
 1211 handling relative 3D rotations, often used in computer graphics and every kind of 3D  
 1212 modelization, are quaternions, as shown in section 2.1.

1213 In this new structure, segments are replaced with cylinders, and circles are replaced  
 1214 with spheres. At every bifurcation to every cylinder are applied the following transfor-  
 1215 mations:

- 1216 • a contraction in its extensions, regulated by an adjustable parameter  $R$ .
- 1217 • the usual deviation of  $\pm 85^\circ$  respect to the direction of the parent branch.
- 1218 • a  $90^\circ$  specific rotation along the axis of its parent branch.

1219 The result of this procedure is a 3D ramification like the one in Figure 3.2, in which  
 1220 we can recognize a good coverage of the space defined by the structure's boundaries and

immediate relation with the 2D structure in Figure 3.1. It should be noted that, in the further refinements of the model from now on, there won't be present the progressive angular indetermination on the direction of branches. Although it is a feature already implemented and working, it requires efficient control to avoid reciprocal overlapping between elements to produce a realistic structure. This second element has not been already developed and it would certainly enrich the representative power of the model.

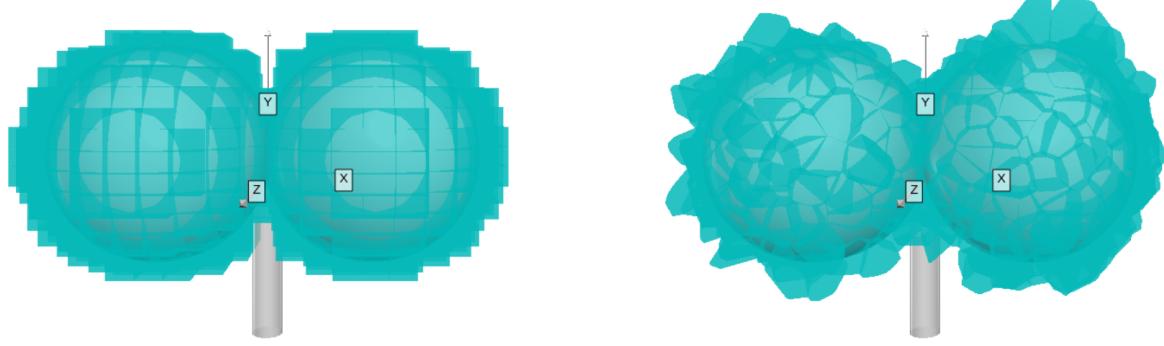
As for the 2D ramification the production of this structure has required the implementation of a tool for the 3D generation with a greatly wider power, able to produce almost any type of three-dimensional iterative structure after the right adjustment, and with a high degree of customization. It is necessary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

### 3.1.3 Subdivision in Cells

Once the 3D backbone of the pancreatic tissue blood vessels ramification system has taken shape, the next step was to embed all this structure in a spatial partitioning process, to create the subdivision into single cells. To perform this important task I used a 3D Voronoi decomposition, as shown in section 2.3. Depending on the choice of the starting points, the Voronoi tessellation could be an excellent item to recreate individual cells because it could guarantee some important properties: all the regions are convex, adjacent, with similar size and volume, with different shapes, and without holes. These have been chosen as the most significant properties to be reflected in the first modelization of cells.

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

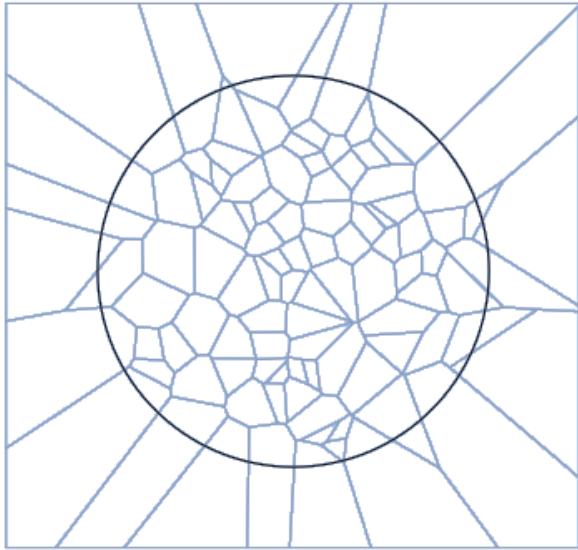
There are tough some delicate considerations to be highlighted about the decomposition procedure. The first regards the most external pieces of the decomposition. Whilst the internal pieces are neatly bounded and defined, the most external layer instead is made on unbounded regions, which extend themselves to infinity. Those regions have clearly to be rejected, as it would be absurd for a cell to have an infinity volume. Typi-



(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.

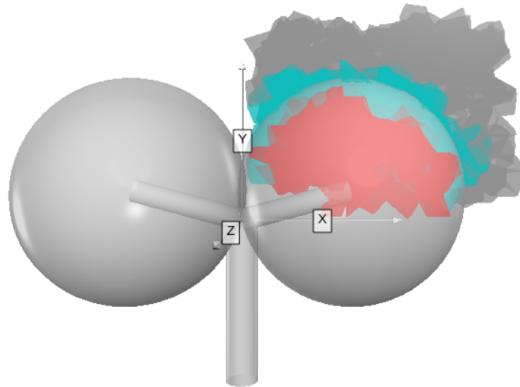


**Figure 3.4:** Example of circular cropping in a 2D Voronoi decomposition: all the regions which intersect the circumference have to be resized.

1260 cally those unbounded regions are resized in order to adhere to some limiting boundaries,  
 1261 with an operation known as *cropping*. In Figure 3.4 is shown an example of circular crop-  
 1262 ping in a 2D Voronoi decomposition: all the regions which intersect the circumference  
 1263 have to be resized.

1264 The cropping operation in 3D is extremely complex, tough. Thus, a more simple and  
 1265 efficient, yet less elegant, technique has been used. Instead of resizing the regions which  
 1266 lie on the boundaries of the sampling region, those regions have directly been rejected.  
 1267 This process is really fast and it does not lead to any danger of representativity loss if  
 1268 the boundaries are loose enough and if the density of sampling is not too low.

1269 The other important consideration regards the density of sampling points. Increasing  
 1270 the number of points to be extracted from the same volume automatically the number  
 1271 of cells in the box will rise, and in contrast, their relative dimension will decrease. This  
 1272 is a key element of the model: a too rarified decomposition would not be able to reflect  
 1273 the complexity of the structure underneath, but a too crowd decomposition on the other  
 1274 side would lead to an unrealistic dimension of the cells in the tissue. Furthermore, this  
 1275 parameter has a huge influence on the computing time necessary to generate the model  
 1276 and to process it for the sectioning as will be shown in section 3.3. In almost all the  
 1277 applications so far, the density parameter has been tuned by eye, with a trial and error  
 1278 procedure. Although, a more rigorous way to adjust this parameter would be to consider  
 1279 the average dimension of the cells and make some microanatomical considerations to



**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.

1280 define the correct relative dimensions. The measure of the volume<sup>1</sup> of the decomposition's  
 1281 regions is an accessible parameter, thus an easy way to estimate the average linear  
 1282 dimension of the cells can be to approximate all the cells to cuboid seeing their volumes  
 1283 as  $V \approx L^3$ . Averaging all the  $L$  measures an estimate  $\hat{L}$  can be done. This average  
 1284 length may be compared to the length of the blood vessel ramification, allowing a good  
 1285 reference tool.

### 1286 3.1.4 Cells Identity Assignment

1287 The great power of creating all the models virtually is to know exactly the identity of  
 1288 every point in the structure. Although, This identity has to be reflected at the cellular  
 1289 level, assigning to every region a label. Imagining the Voronoi decomposition represented  
 1290 in Figure 3.3b extended to the entire box containing the ramification, good discrimination  
 1291 would distinguish three classes of cells: those which lie completely inside a sphere, those  
 1292 which lie completely outside a sphere, and those which lie on the boundaries of a sphere.  
 1293 In Figure 3.5 is shown a portion of the complete decomposition where the three classes of  
 1294 cells are reported with different colors: the internal cells in red, those on the boundaries  
 1295 in turquoise, and the external cells in gray.

1296 In this particular case to find the relative position between every sphere in the struc-  
 1297 ture and each cell it has been used a test on the proximity between the spheres' centers  
 1298 and the vertices of every polyhedral cell. If all the vertices of a region lie within a dis-  
 1299 tance lower than the radius from the center of the same sphere then that region can be  
 1300 said to be an internal one. If none of the vertices lie within the radius distance from

---

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

any center then that region is said to be external. In any other case, the region is said to be on the boundaries of some sphere, and this third label is assigned to it. As could be imagined the number of cells inside the volume can grow very quickly, and in the more rich ramifications also the number of spheres could be high. If we think that any polyhedron has a number of vertices of the order of 20/30 then it is clear that the number of distance evaluations could grow very quickly, requiring some relevant computational power in the more extended simulations. In order to optimize this computation, I decided to use a python implementation of a K-dimensional Tree, which is a space-partitioning data structure especially suited for fast and optimized computation of distances [6]. A K-d Tree is an algorithm that iteratively binary splits the space: every node of the three could be thought as a splitting  $(k - 1)$ -hyperplane dividing the space into two semi-hyperspace. The result is an optimized algorithm for repeated distance evaluations. As for many other tools, in my code I used a pre-implemented module `KDTree` from the `Scipy` library.

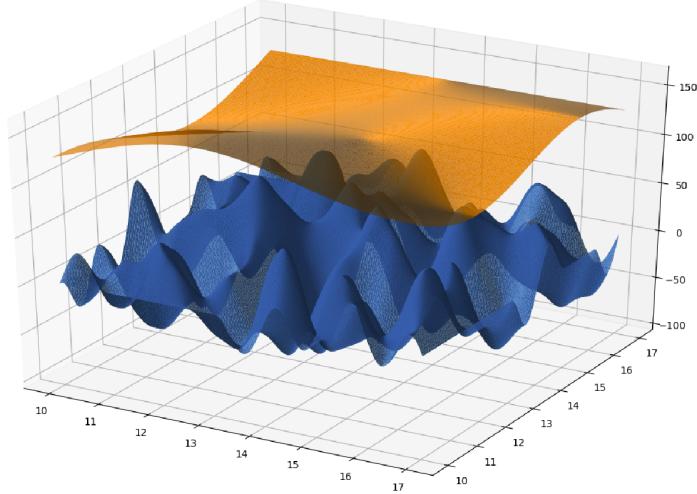
This procedure of labeling the regions is completely customizable, and it should be adapted to the specific application. By the way, the principle will always be to perform some sort of spatial consideration respect to the primary structure and assign all the interesting labels accordingly to the cells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

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

## 3.2 Dermal Tissue Model

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

### 1) Stratified Structure



**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.

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

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

## 2) Subdivision in Cells

1360 The subdivision in cells of the volume containing the structure has followed the  
1361 exact same steps described in section 3.1, hence in this paragraph I will shortly  
1362 resume the process. The first step is the definition of a suitable volume containing  
1363 the structure. Then is the time for the generation of the decomposition's starting  
1364 points according to a quasi-random number generation technique, as described in  
1365 section 2.4. Afterward, the points are used as a base for the decomposition, and  
1366 all the cells with undefined boundaries are rejected

### 1367 3) Cells Identity Assignment

1368 The identity assignment procedure instead has been customized for this particular  
1369 application. In this model there are no *boundary* cells like the one lying on the  
1370 spherical surfaces in the pancreatic model, thus there is no need for a test on the  
1371 position of each vertex of every cell. In this model, the starting point for every  
1372 Voronoi region has been used as a reference, and its relative position respect to the  
1373 boundary surfaces was the discriminating factor for assessing the identity. In order  
1374 to perform a coherent test on the relative position between the regions' center and  
1375 the boundaries surfaces the positions of all the centers had to be discretized on  
1376 the same grid onto which were defined the surfaces. The comparison with the flat  
1377 horizontal plane defining the boundary between epidermal tissue and the keratine  
1378 layer instead was simply a test on the  $z$  coordinate of the point:  $z = f_{(x,y)} \leq \hat{z}_{plane}$ .  
1379 The result of this procedure is that every region is assigned a label corresponding  
1380 to the belonging tissue layer:  $D$  for dermal,  $E$  for epidermal,  $K$  for keratine, and  
1381  $V$ , which stands for *void* for the white empty space above the sample.

1382 In this model, as in section 3.1, after the assignment of the identity to all the cells in  
1383 the volume the modelization process is considered complete. Any sort of improvement  
1384 and enrichment should be inserted during the structure designing phase and should be  
1385 linked to an identity label.

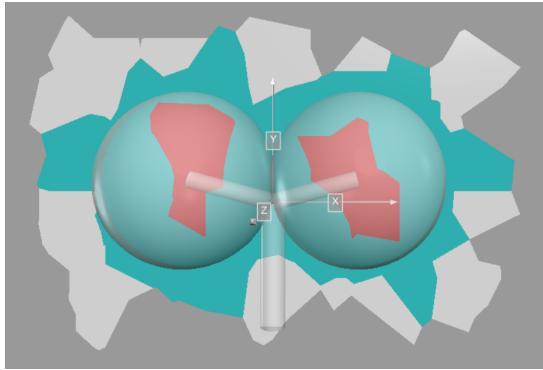
1386 **3.3 Synthetic Images Production**

1387 After the model is complete we have a three-dimensional representation of the tissue  
1388 under study. The aim of the work is though to produce synthetic images from that  
1389 structure, more precisely we want pairs of images composed of the synthetic-histological  
1390 images and its related segmentation mask. The transition from 3D structure to a 2D  
1391 image is the last step in the process, and it is inspired by the actual traditional technique  
1392 for the preparation of the histological specimen, as described back in section 1.1.1. As  
1393 the biopsy sample is treated and then sectioned with the microtome, the virtual model  
1394 is sectioned in a random direction, producing an image representing the slice. This  
1395 first image contains all the information of the section but its appearance is completely  
1396 arbitrary and its look has nothing to share with a realistic sample. The original slice  
1397 then acts perfectly as a segmentation mask, but some careful and dramatic makeover is  
1398 needed to produce the final realistic looking image. In this section I will describe the  
1399 general procedure to produce virtual slices from the two 3D virtual models described  
1400 before in section 3 and the technique used to edit the images and give them the desired  
1401 appearance.

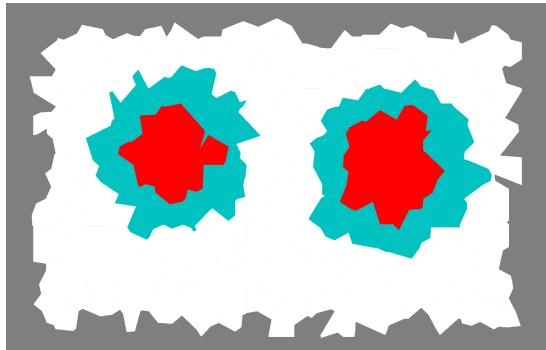
1402 **3.3.1 Sectioning Process**

1403 For any model created following the general procedure described in 3, even more so for  
1404 the two particular models of pancreatic and dermal tissue, the sectioning process will  
1405 be almost the same, and it will rely mainly on the algorithm for the general section of  
1406 polyhedron described back in section 2.5. As stated before the models are essentially  
1407 composed by labeled polyhedron spatially organized in a 3D volume. The ordered section  
1408 of each polyhedron will yield all the polygons that shall be assembled in the final section.  
1409 In Figure 3.7a is shown the three-dimensional representation of the section of a simple  
1410 ramification, as the one in Figure 3.5. All the polygons that compose the section are  
1411 drawn with the color correspondent to their label, following the same idea of Figure 3.5.  
1412 In Figure 3.7b is printed the final result of the sectioning algorithm applied to the model,  
1413 which will be the segmentation mask in the single pair of synthetic images. The colors  
1414 in the produced slice match the colors used for the different identities in the 3D model.

1415 The simplest, yet over-abundant, way to proceed is to create the model in its entirety  
1416 and subsequently choosing the sectioning plane. Afterward, it is necessary to select  
1417 only the regions that intersect the plane and section them all. Actually, the test on the  
1418 intersection passes through the check on the relative position of the polyhedron's vertices  
1419 respect to the sectioning plane: if all the vertices lie on the same semi-space then the  
1420 intersection would be null and the polyhedron is not of interest for that particular section.  
1421 This procedure is exactly the first step of the algorithm in 2.5, thus the filtering on the  
1422 regions is actually made during construction for optimization. The alternative method  
1423 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.

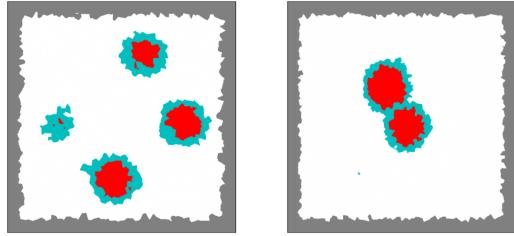


(b) The final section image, produced directly on a planar picture, skipping completely the 3D representation.

**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.

1424 place to generate the model's decomposition only in the volume adjacent to that plane.  
 1425 This method enables the sparing of a good amount of computation, without any negative  
 1426 impact on the final result. The only delicate step is the choice of a wide enough region  
 1427 of space around the plane, which doesn't compromise the representation.

1428 As a guarantee for richness and diversification among the images, there is the need  
 1429 for some degree of controlled randomness in the sectioning process, for example in the  
 1430 determination of the sectioning plane direction. All the sectioning process is then based  
 1431 on a single starting *seed*, which determines the direction of the sectioning plane in a  
 1432 deterministic way, and all the rest of the model is generated as a consequence. In this  
 1433 way, all the possible angulations are equally probable and will be sampled in view of  
 1434 multiple applications of this process. In Figure 3.8 are shown two different sections,  
 1435 along two random planes on two simple ramified structures with four spheres.



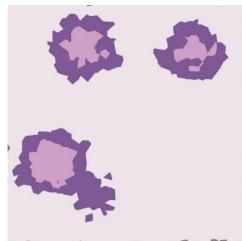
**Figure 3.8:** Two different section, along two random planes on two simple ramified structure with four spheres.

### 1436 3.3.2 Appearance Makeover

1437 After the application of the sectioning algorithm of the previous section, the image  
 1438 which will act as a segmentation mask is ready. The last and most complex task that  
 1439 remains is to transform the image and to give it a realistic look. I tried many different  
 1440 transformations, more or less complicated, and there was not a final decision on which is  
 1441 the best blend of them. In this section, I will describe them as an arsenal of possibilities  
 1442 and show their impact on the images.

#### 1443 Color Palette

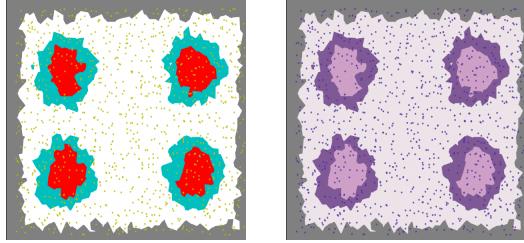
1444 The first correction to do to the images will inevitably be a change in the colors  
 1445 of the image. Gray, Turquoise, and Red are the perfect choice for label-colors but  
 1446 act poorly as physiological colors. In Figure 3.9 is shown an example of an image  
 1447 produced re-mapping the colors to a new palette, inspired to the coloring of the  
 1448 real specimen in Figure 1.3 and 1.2, given by the traditional hematoxylin and eosin  
 1449 staining process.



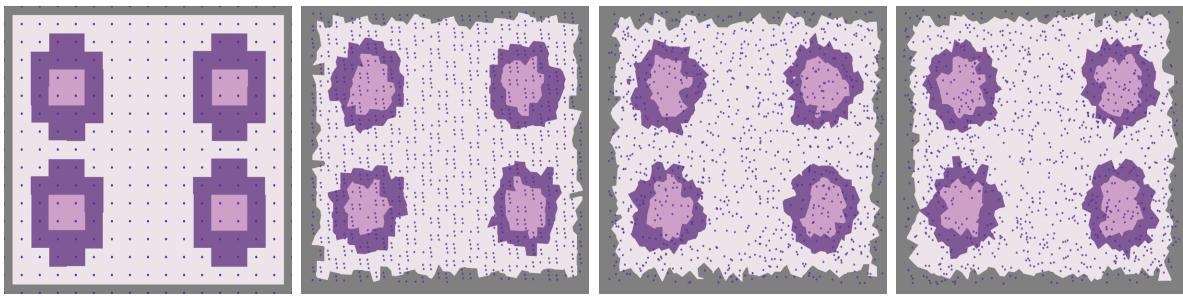
**Figure 3.9:** Example of images produced re-mapping the colors with a color palette inspired to a real H&E stained histological sample.

### 1450 Nuclei Projection

1451 Another fundamental processing needed was the projection of cells' nuclei on the  
 1452 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.

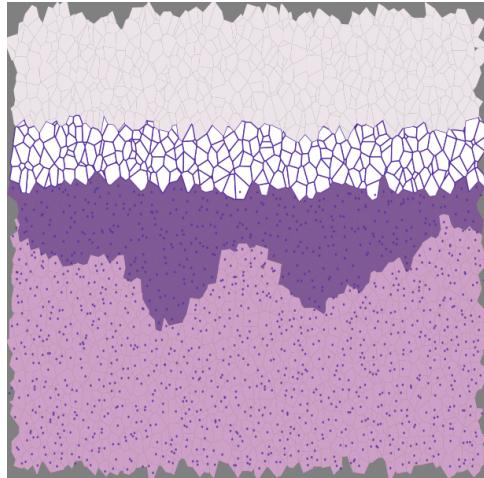
analysis allowing to detect individual cells in the specimen. As a reference for the nucleus position the starting point of every polyhedral region has been used and projected on the sectioning plane as a little dark circle. The diameter of those circles as been chosen to be a submultiple (10%) of the linear estimated dimension  $\hat{L}$  of the cells in the decomposition<sup>2</sup>.

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

---

<sup>2</sup>Following 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.

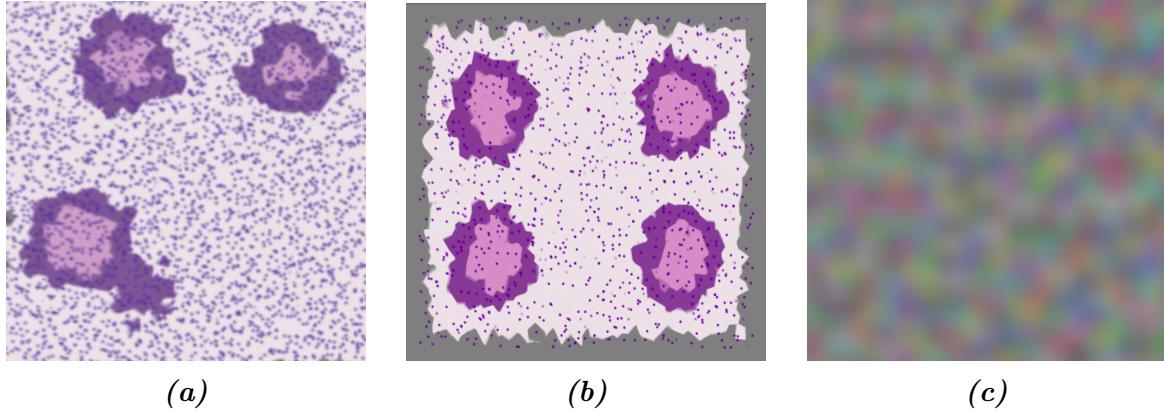
On the same wave of the previous tool, another operation that can help the appearance of an image is the projection of the boundaries of each (or just a part) of the polygonal sections. The drawing can be clearly tuned and customized depending on the specific necessities. In Figure 3.12 is shown an example 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.

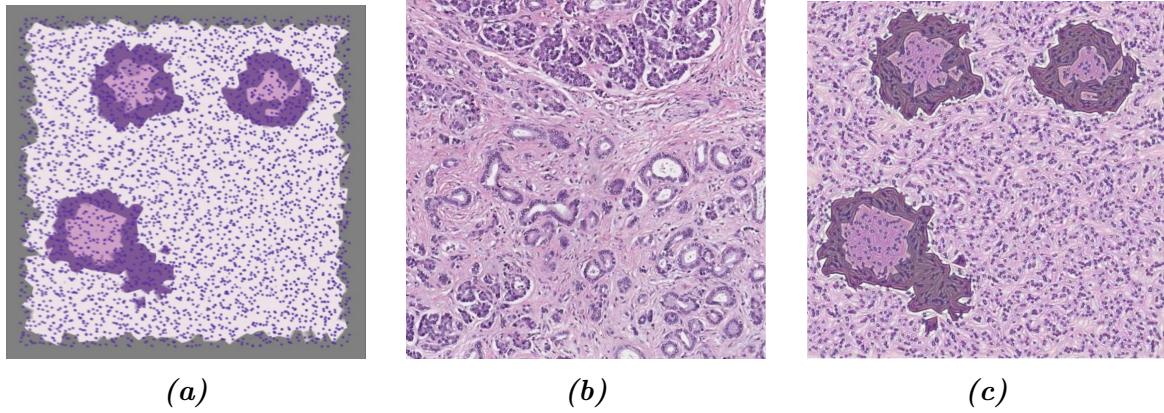
Perlin noise, described in section 2.6. The idea is to create some fluctuation among the color channels of the image around the sharp values of the image produced by the sectioning algorithm. From a practical point of view, I created three different and independent Perlin noise surfaces, one for every 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



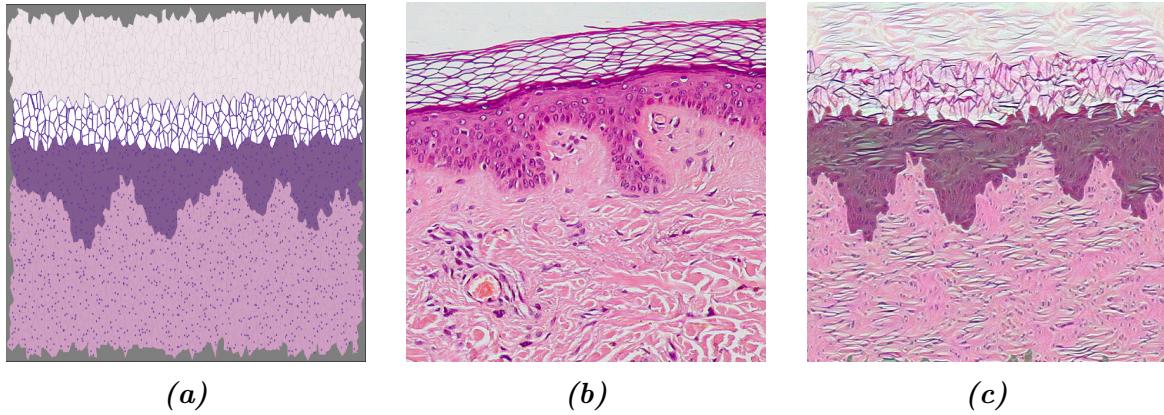
**Figure 3.14:** Application of the style-transfer NN on a section of the pancreatic tissue model: 3.14a the content image, 3.14b the style image, 3.14c the hybrid resulting image.

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.

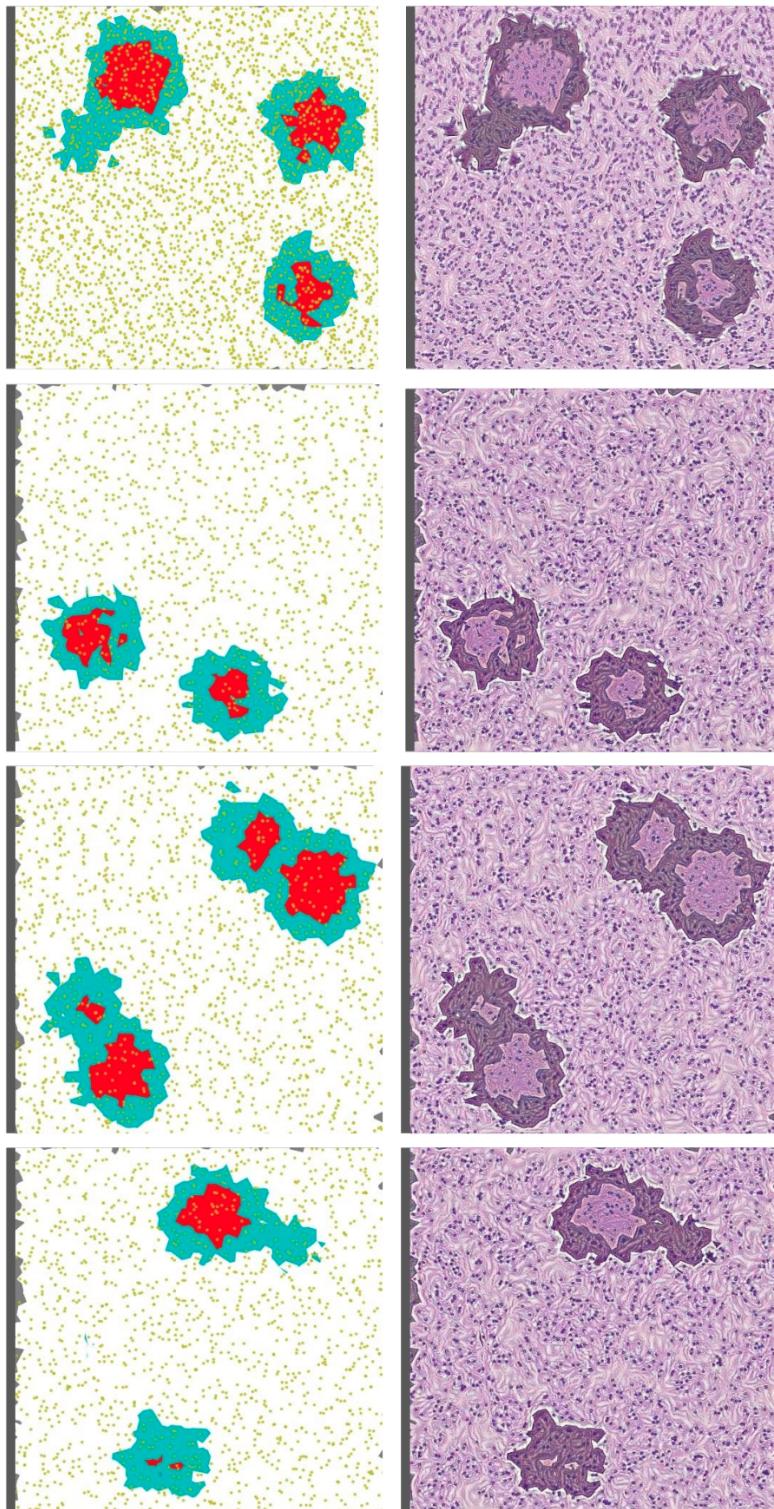
One single complete application of the process consists then in the generation of a tissue model, in the sectioning along a random section plane, and in the processing of the image, in order to produce the pair of ground-truth image and the synthetic histological image. The target is to apply over and 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.

# <sup>1549</sup> Conclusions

<sup>1550</sup> In this project, I face the problem of synthetic histological image generation for the  
<sup>1551</sup> purpose of training Neural Networks for the segmentation of real histological images. The  
<sup>1552</sup> manual analysis of histological specimens is a complex, time-consuming, and expensive  
<sup>1553</sup> task and nevertheless, it is a pillar of countless diagnostic techniques. Any form of  
<sup>1554</sup> support for this procedure hence is welcome and endorsed by the health-care system. In  
<sup>1555</sup> particular, in this work, I focus on the problem of histological specimens segmentation.  
<sup>1556</sup> The most advanced algorithms for image segmentation are based on Deep Learning  
<sup>1557</sup> and requires the training of extensive and complex neural networks. One of the toughest  
<sup>1558</sup> hurdles to overcome for the training of those NN is the abundance and the quality of pre-  
<sup>1559</sup> labeled examples of segmentation on real histological samples. The collection of hundreds  
<sup>1560</sup> of hand-labeled histological samples, with pixel-level precision, is virtually impossible.  
<sup>1561</sup> This work thus proposes a methodology to generate, in a completely automatic way,  
<sup>1562</sup> synthetic pre-labeled histological-like images, that can be used as training material for  
<sup>1563</sup> a NN.

<sup>1564</sup> The method I propose consists of the recreation of the traditional histological speci-  
<sup>1565</sup> mens' preparation, and it is based on the sectioning of a 3D virtual model of a region of  
<sup>1566</sup> histological tissue. The virtual 3D model of a region of a particular type of human tissue  
<sup>1567</sup> is built after physical and physiological considerations. The model is then subject to a  
<sup>1568</sup> virtual sectioning operation, which yields the synthetic sampling of the virtual tissue in  
<sup>1569</sup> which the histological identity of every pixel is perfectly known. This first image will act  
<sup>1570</sup> as a segmentation mask for a second, realistic image. In fact, on top of this first image  
<sup>1571</sup> are applied several aesthetical processing and refinements and the final product is the  
<sup>1572</sup> synthetic histological-like image. The pair made of the two images is perfectly suitable  
<sup>1573</sup> for the supervised learning of a NN oriented toward the segmentation of histological  
<sup>1574</sup> images. The production of each pair of images is completely automatic and it does not  
<sup>1575</sup> require the intervention of any human operator, it is thus a scalable process that can  
<sup>1576</sup> produce a great abundance of images. The quality of the images is directly connected  
<sup>1577</sup> to the richness and the quality of the model. The perfect modelization of a region of  
<sup>1578</sup> tissue, let's say human pancreatic tissue, is by far out of reach for this, hence the rich-  
<sup>1579</sup> ness and the fidelity of the produced images are inevitably lower than the real sample.  
<sup>1580</sup> Nevertheless, the quality of the produced images is sufficient to perform the preliminary

1581 phase of the training of a NN following a training strategy known as curriculum learning.  
1582 This learning process consists of giving the NN a copious quantity of lower complexity  
1583 level example in the first instance, reserving the few and sophisticated real hand-labeled  
1584 histological samples for the finalization of the training.

1585 The first chapter of this thesis is devoted to the contextualization of the present  
1586 work. It is offered a description of how the histological samples are obtained after a  
1587 tissue biopsy and how the digitalization process of the images works.

1588 The second chapter collects all the details of every less common technical tool I used  
1589 during the design of this project. A brief theoretical introduction is proposed for every  
1590 item besides the thorough description of its practical use. In this chapter, a section is  
1591 devoted to the description of a general methodology for computing the 2D section of an  
1592 arbitrary three-dimensional polyhedron. The algorithm here described has been devised  
1593 and implemented all by my self, and then inserted in the workflow of the project. This  
1594 is one of the key pieces for the automatization of the virtual tomography process, and it  
1595 allows us to connect the three-dimensional model to the two-dimensional representation  
1596 of a sampled section.

1597 The third chapter is the center of this work, and it contains the description of all  
1598 the design choices, and the steps I followed for the development of the two human tissue  
1599 models I propose: the first of pancreatic tissue and the second of dermal tissue. The  
1600 first and second sections are dedicated to the description of the two proposed 3D virtual  
1601 tissue models, which consists of different steps. The third section instead contains a  
1602 thorough description of the method to perform the sectioning onto a virtual model and  
1603 how to process the resulting images. The development has required the harmonization  
1604 of many different technical aspects and mathematical tools and it results in a general  
1605 methodology for the generation of synthetic histological images. The process passes first  
1606 through the building of the target tissue's structure in a virtual environment. This struc-  
1607 ture is then embedded in a three-dimensional space decomposition that subdivides the  
1608 volume into individual cells. Those cells are labeled in correspondence to their role in  
1609 the model, and their identity is then perfectly encaptured by the virtual tomography  
1610 procedure. The sectioning process is responsible for the production of synthetic images,  
1611 which are then conveyed toward an aesthetical enrichment pipeline specialized for the  
1612 particular target tissue. The product of any application of this process is a pair made of  
1613 a segmentation mask and the corresponding synthetic histological-like image. This com-  
1614 pletely automatized procedure allows building arbitrary large datasets for the training  
1615 of NN, without the intervention of any human operator.

1616 The method for the generation of datasets of synthetic images I propose in my thesis  
1617 work is a self-supporting project and it is formally consistent. By the way, there are  
1618 many possibilities for improvement and enrichment for the project. One first aspect to  
1619 strengthen could be the richness of the models: adding more elements in the structure,  
1620 and refining their representation of the tissue at the cellular level. This would lead  
1621 to a better quality of the synthetic images, that would assist the training of NN in

more and different applications. Another aspect that lends itself to improvements in the development of a dedicated style transfer NN targeting the histological texture transfer, which could lead to interesting signs of progress in the quality of image generation. There is also the intention to perform an actual attempt of NN training on the images produced with this process. This would complete conceptually the idea underneath the project and would be an excellent opportunity to detect weaknesses and to draw up possible lines of development. The repeated application of the generation method would allow the building of entire datasets suitable for the training of DL-based models.

<sub>1630</sub>

# Bibliography

- <sub>1631</sub> [1] Abien Fred Agarap. Deep learning using rectified linear units (relu), 2018.
- <sub>1632</sub> [2] Salah Alheejawi, Mrinal Mandal, Hongming Xu, Cheng Lu, Richard Berendt, and Naresh Jha. Deep learning-based histopathological image analysis for automated detection and staging of melanoma. In *Deep Learning Techniques for Biomedical and Health Informatics*, pages 237–265. Elsevier, 2020.
- <sub>1633</sub> [3] J. Alsayednoor and P. Harrison. Evaluating the performance of microstructure generation algorithms for 2-d foam-like representative volume elements. *Mechanics of Materials*, 98:44 – 58, 2016.
- <sub>1634</sub> [4] Hani A Alturkistani, Faris M Tashkandi, and Zuhair M Mohammedsaleh. Histological stains: A literature review and case study. *Global Journal of Health Science*, 8(3):72, June 2015.
- <sub>1635</sub> [5] Yoshua Bengio, Jérôme Louradour, Ronan Collobert, and Jason Weston. Curriculum learning. In *Proceedings of the 26th Annual International Conference on Machine Learning*, ICML '09, page 4148, New York, NY, USA, 2009. Association for Computing Machinery.
- <sub>1636</sub> [6] Jon Louis Bentley. Multidimensional binary search trees used for associative searching. *Commun. ACM*, 18(9):509517, September 1975.
- <sub>1637</sub> [7] R. W. Brockett. Robotic manipulators and the product of exponentials formula. In P. A. Fuhrmann, editor, *Mathematical Theory of Networks and Systems*, pages 120–129, Berlin, Heidelberg, 1984. Springer Berlin Heidelberg.
- <sub>1638</sub> [8] C. G. BROYDEN. The Convergence of a Class of Double-rank Minimization Algorithms 1. General Considerations. *IMA Journal of Applied Mathematics*, 6(1):76–90, 03 1970.
- <sub>1639</sub> [9] Bassem Ben Cheikh, Catherine Bor-Angelier, and Daniel Racoceanu. A model of tumor architecture and spatial interactions with tumor microenvironment in breast carcinoma. In Metin N. Gurcan and John E. Tomaszewski, editors, *Medical Imaging*

- 1657        2017: *Digital Pathology*, volume 10140, pages 73 – 80. International Society for  
 1658        Optics and Photonics, SPIE, 2017.
- 1659 [10] Anna Choromanska, Benjamin Cowen, Sadhana Kumaravel, Ronny Luss, Mattia  
 1660        Rigotti, Irina Rish, Brian Kingsbury, Paolo DiAchille, Viatcheslav Gurev, Ravi  
 1661        Tejwani, and Djallel Bouneffouf. Beyond backprop: Online alternating minimization  
 1662        with auxiliary variables, 2018.
- 1663 [11] Siddharth Singh Chouhan, Ajay Kaul, and Uday Pratap Singh. Image segmentation  
 1664        using computational intelligence techniques: Review. *Archives of Computational  
 1665        Methods in Engineering*, 26(3):533–596, February 2018.
- 1666 [12] Marius Cordts, Mohamed Omran, Sebastian Ramos, Timo Rehfeld, Markus En-  
 1667        zweiler, Rodrigo Benenson, Uwe Franke, Stefan Roth, and Bernt Schiele. The  
 1668        cityscapes dataset for semantic urban scene understanding. *CoRR*, abs/1604.01685,  
 1669        2016.
- 1670 [13] Angel Cruz-Roa, Ajay Basavanhally, Fabio Gonzlez, Hannah Gilmore, Michael Feld-  
 1671        man, Shridar Ganesan, Natalie Shih, John Tomaszewski, and Anant Madabhushi.  
 1672        Automatic detection of invasive ductal carcinoma in whole slide images with convo-  
 1673        lutional neural networks. *Progress in Biomedical Optics and Imaging - Proceedings  
 1674        of SPIE*, 9041, 02 2014.
- 1675 [14] Alessandro d'Agostino. Dataset generation for the training of neural networks ori-  
 1676        ented toward histological image segmentation, april 2020.
- 1677 [15] J. Deng, W. Dong, R. Socher, L. Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale  
 1678        hierarchical image database. In *2009 IEEE Conference on Computer Vision and  
 1679        Pattern Recognition*, pages 248–255, 2009.
- 1680 [16] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei. ImageNet: A Large-  
 1681        Scale Hierarchical Image Database. In *CVPR09*, 2009.
- 1682 [17] Mark Everingham, Luc Van Gool, Christopher KI Williams, John Winn, and An-  
 1683        drew Zisserman. The pascal visual object classes (voc) challenge. *International  
 1684        journal of computer vision*, 88(2):303–338, 2010.
- 1685 [18] Leon A. Gatys, Alexander S. Ecker, and Matthias Bethge. A neural algorithm of  
 1686        artistic style, 2015.
- 1687 [19] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural com-  
 1688        putation*, 9(8):1735–1780, 1997.
- 1689 [20] R. H. Hruban, A. Maitra, and M. Goggins. Update on pancreatic intraepithelial  
 1690        neoplasia. *Int J Clin Exp Pathol*, 1(4):306–316, Jan 2008.

- 1691 [21] Phillip Isola, Jun-Yan Zhu, Tinghui Zhou, and Alexei A. Efros. Image-to-image  
1692 translation with conditional adversarial networks, 2016.
- 1693 [22] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization,  
1694 2014.
- 1695 [23] Alex Krizhevsky. Learning multiple layers of features from tiny images. Technical  
1696 report, 2009.
- 1697 [24] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with  
1698 deep convolutional neural networks. In *Advances in neural information processing*  
1699 *systems*, pages 1097–1105, 2012.
- 1700 [25] Johannes Kster and Sven Rahmann. Snakemakea scalable bioinformatics workflow  
1701 engine. *Bioinformatics*, 28(19):2520–2522, 08 2012.
- 1702 [26] Aristid Lindenmayer. Mathematical models for cellular interactions in development  
1703 ii. simple and branching filaments with two-sided inputs. *Journal of theoretical*  
1704 *biology*, 18(3):300–315, 1968.
- 1705 [27] Daniel. Longnecker. Anatomy and histology of the pancreas, 2014.
- 1706 [28] Shervin Minaee, Yuri Boykov, Fatih Porikli, Antonio Plaza, Nasser Kehtarnavaz,  
1707 and Demetri Terzopoulos. Image segmentation using deep learning: A survey, 2020.
- 1708 [29] Muhammad Niazi, Thomas Tavolara, Vidya Arole, Douglas Hartman, Liron Pan-  
1709 tanowitz, and Metin Gurcan. Identifying tumor in pancreatic neuroendocrine neo-  
1710 plasms from ki67 images using transfer learning. *PLOS ONE*, 13:e0195621, 04 2018.
- 1711 [30] Arild Nkland and Lars Hiller Eidnes. Training neural networks with local error  
1712 signals, 2019.
- 1713 [31] Travis E Oliphant. *A guide to NumPy*, volume 1. Trelgol Publishing USA, 2006.
- 1714 [32] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gre-  
1715 gory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban  
1716 Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan  
1717 Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith  
1718 Chintala. Pytorch: An imperative style, high-performance deep learning library,  
1719 2019.
- 1720 [33] Allan Pinkus. Approximation theory of the mlp model in neural networks. *Acta*  
1721 *Numerica*, 8:143195, 1999.

- 1722 [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.
- 1725 [35] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. *CoRR*, abs/1505.04597, 2015.
- 1727 [36] Juan Rosai. Why microscopy will remain a cornerstone of surgical pathology. *Laboratory Investigation*, 87(5):403–408, April 2007.
- 1729 [37] Andrea Saltelli. Making best use of model evaluations to compute sensitivity indices. *Computer Physics Communications*, 145(2):280 – 297, 2002.
- 1731 [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.
- 1735 [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.
- 1739 [40] David F Shanno. Conditioning of quasi-newton methods for function minimization. *Mathematics of computation*, 24(111):647–656, 1970.
- 1741 [41] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition, 2014.
- 1743 [42] Sandro Skansi. *Introduction to Deep Learning: From Logical Calculus to Artificial Intelligence*. Springer Publishing Company, Incorporated, 1st edition, 2018.
- 1745 [43] I.M. Sobol. Uniformly distributed sequences with an additional uniform property. *USSR Computational Mathematics and Mathematical Physics*, 16(5):236 – 242, 1976.
- 1748 [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.
- 1751 [45] M Titford. The long history of hematoxylin. *Biotechnic & Histochemistry*, 80(2):73– 78, 2005.

- 1753 [46] Pauli Virtanen, Ralf Gommers, Travis E. Oliphant, Matt Haberland, Tyler Reddy,  
1754 David Cournapeau, Evgeni Burovski, Pearu Peterson, Warren Weckesser, Jonathan  
1755 Bright, Stéfan J. van der Walt, Matthew Brett, Joshua Wilson, K. Jarrod Millman,  
1756 Nikolay Mayorov, Andrew R. J. Nelson, Eric Jones, Robert Kern, Eric Larson,  
1757 CJ Carey, İlhan Polat, Yu Feng, Eric W. Moore, Jake VanderPlas, Denis Laxalde,  
1758 Josef Perktold, Robert Cimrman, Ian Henriksen, E. A. Quintero, Charles R Harris,  
1759 Anne M. Archibald, Antônio H. Ribeiro, Fabian Pedregosa, Paul van Mulbregt,  
1760 and SciPy 1.0 Contributors. SciPy 1.0: Fundamental Algorithms for Scientific  
1761 Computing in Python. *Nature Methods*, 17:261–272, 2020.
- 1762 [47] Georges Voronoi. Nouvelles applications des paramètres continus à la théorie des  
1763 formes quadratiques. premier mémoire. sur quelques propriétés des formes quadra-  
1764 tiques positives parfaites. *Journal für die reine und angewandte Mathematik (Crelles  
1765 Journal)*, 1908:102 – 97.