Internship report
M2 TSI SIBiom
2021-2022

Deep Learning for Microscope Enhancing

Gabriel Bon[1,2]

Supervised by :
Cedric Matthews[2]
Fabrice Daian[2]
Daniel Sapede[2]

Abstract
Microscopy techniques like confocal microscopy are evolving quickly thanks to the improvement of the detection chain but also thanks to the optical corrections.
Devices older than 5 years are still being used despite their lower image quality compared with newer generations microscopes. This quality loss is notably caused by the aging of the detectors and by the degradation over time of both chromatic and spherical corrections. It is a consequence of time itself but also of the accumulated running time of the system. For the instrument to be useful even with its age, it is needed to be upgraded to the performance level of more modern ones.
In order to assess the feasibility of such algorithmic microscope enhancing, I designed and built from scratch a dataset simulating older and newer device. It is made of different kind of biological tissues and metrology marbles acquired in both high and low quality configuration. Moreover, I have set up and designed different neural network architectures such as convolutional AutoEncoders and generative models (Pix2Pix, CycleGAN). I monitored their training through hyperparameters tuning. Finally, I assessed their quality using different image attribute metrics.
Learning processes have already been done in plenty of imaging domains to improve the efficiency of a given instrument. It is a completely different task to do it in the inter-generational context where we know the performance limits of the last generation whom we want to transfer to the previous ones. This frontier and original work will have an impact on the different moment of an instrument's life, such as:
- Temporary performance degradation, compensated algorithmically lying in wait of a fix-up from the after-sale service
- End of life for an instrument in case of technical degradation, which will be delayed with an improvement in data quality, to be taken as an algorithmic prosthesis.

Keywords
Deep Learning — Confocal microscopy — Biological imaging

[1] *Faculté des sciences de St-Jérôme, Aix-Marseille Université, Marseille, France*
[2] *Institut de Biologie du Développement de Marseille, Aix-Marseille Université, Marseille, France*
Corresponding author : gabriel.bon@etu.univ-amu.fr
Corresponding tutors : cedric.matthews@univ-amu.fr
fabrice.daian@univ-amu.fr
daniel.sapede@univ-amu.fr

Contents

Introduction

The IBDM (Marseille Developmental Biology Institute) is overseen by the CNRS (National Scientific Research Centre) and the AMU (Aix-Marseille University). Around twenty teams of researchers are being hosted there, with their research mainly oriented on how an embryo develops and in particular the pathologies than can occur while growing into an adult organism (neuropathy, cancer, ...).

I've got the opportunity to join as an intern the Optical Imaging and the Software Development facilities for 6 months during which I was able to get acquainted to some imaging systems and even manipulate some of them. My use of the microscopes has been mainly to build a robust dataset for the subsequent Deep Learning analysis. The task aimed with this internship is to set up and develop an analysis pipeline based on neural networks able to fix flawed images produced by older or temporarily degraded systems.

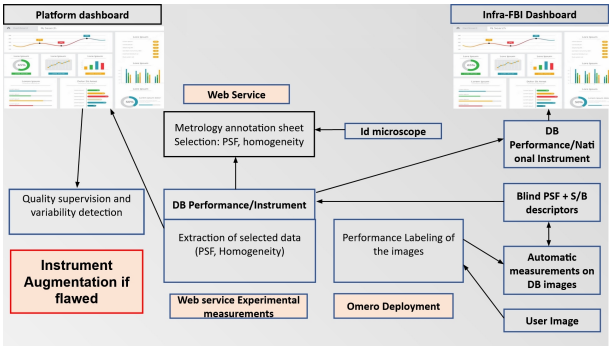
This report points out all the tools used and results this internship led to. The first part will be about confocal microscopy. I manipulated the acquisition systems myself. This allowed for a better understanding on how the images to be processed are created. At a later stage it also helps with some of the causes of any phenomenon appearing on the data to better process them afterward.

The second part will go into details about the different approaches, both AutoEncoder and generative neural network architectures used to find solutions in the image enhancing and denoising themes.

Finally I will display the different results achieved, comment them and discuss on the other approaches and methods I could have studied during this internship.

Context

The aim of this internship is inscribed into a larger and long-term project which is currently developed and supported by the IBDM Optical Microscope facility.



Organization of the global metrology project.The current work is marked with a red bordered frame in the lower left part of the diagram. This figure can be seen in larger in the annexes.

In order to improve data value and accessibility, the entire project aims at creating a metrology database with labeled data.

In this database, each piece of data would have its characteristics extracted. With this extraction, the data quality could be automatically evaluated. Since the data would be labeled with the name of the instrument it has been made with, we could know whether this same device is performing correctly or not throughout time.

In other words, this would indicate when an instrument could need to be enhanced with deep learning. This is the situation simulated in this project.

1. Material & Methods

This internship provided plenty of resources to work on. I've worked with several confocal microscopy setups. Moreover, I could set up and practice with various neural networks and deep learning materials.

1.1 Microscopy

In this project, the source of our images will be the signal emitted by the samples out of fluorescence. In order to produce this phenomenon, the sample needs to receive a specific lighting. Once this is done, some components of the sample can absorb the photons and emit it back a really short time later. While doing so a part of the energy will be wasted most likely into heat. Therefore, the emitted photon has a lower amount of energy. This translates into a color shift called Stokes' shift [Fig. 1].

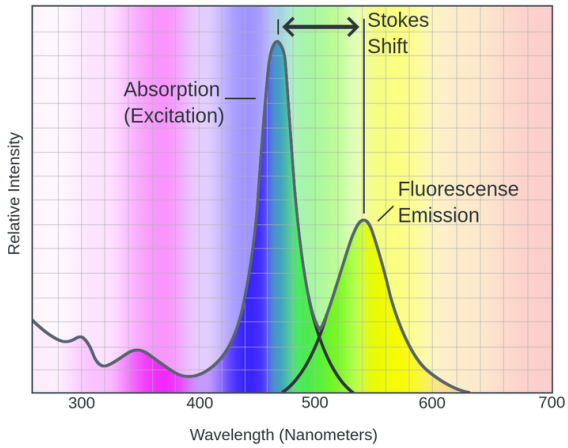


Figure 1. *Stokes' shift. Source : theory.labster.com*

It is explained by the Jablonski diagram[Fig. 2], where we see the fluorescent molecule reaching an excited state and losing energy to get back to a fundamental vibrational state before going back to the ground state.

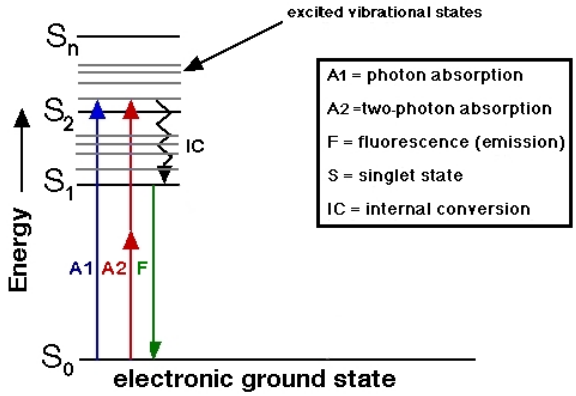


Figure 2. *Simplified Jablonski Diagram for single and two-photon excitation. Excitation wavelength for A1 is half the wavelength(more energetic) of the two photons of A2. Source : shsu.edu*

1.1.1 Confocal microscopy notions

Fluorescence microscopy Sometimes, widefield imaging illuminated by visible light doesn't permit the detection of certain structures. To bypass this restriction, some specific filter and lighting can be used to trigger a fluorescence reaction in a sample [Fig. 3]. It can be naturally sensitive to a given wavelength or be marked to be with some fluorochromes.

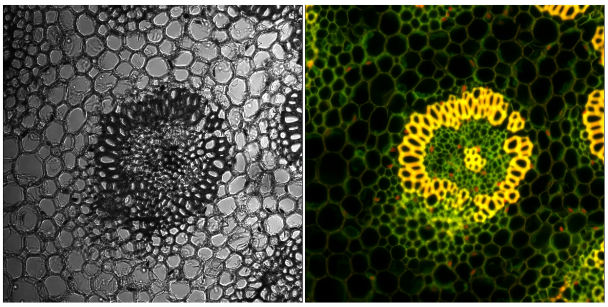


Figure 3. *LSM 510 acquisition of Convallaria rhizome. Transmission (left) and fluorescent (right) light imaging.*

For instance, the DAPI is known for its blue fluorescence, the entire Alexa Fluor series by MolecularProbes or GFP and RFP has got respectively Green and Red Fluorescent Protein which are biomarkers at the same time. Those fluorochromes aren't just matching a single color [Fig. 4]. During this kind of imaging, the light get it's wavelengths filtered along the way, before reaching the sample through excitation filter or on its way back from the sample thanks to an emission filter.

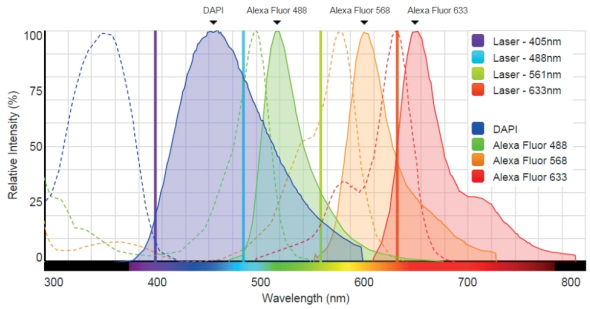


Figure 4. *Excitation and emission spectra for 4 different fluorescent dyes along with 4 typical excitation laser. Source : Zeiss*

Another possibility is to redirect only a part of the spectrum for it to reach a different detector and therefore imaging the sample on two (or more) different channels and detect separate structures in the sample that wouldn't react the same to the excitation light [Fig. 5]. These splitters are called dichroic mirror or dichroic beamsplitter.

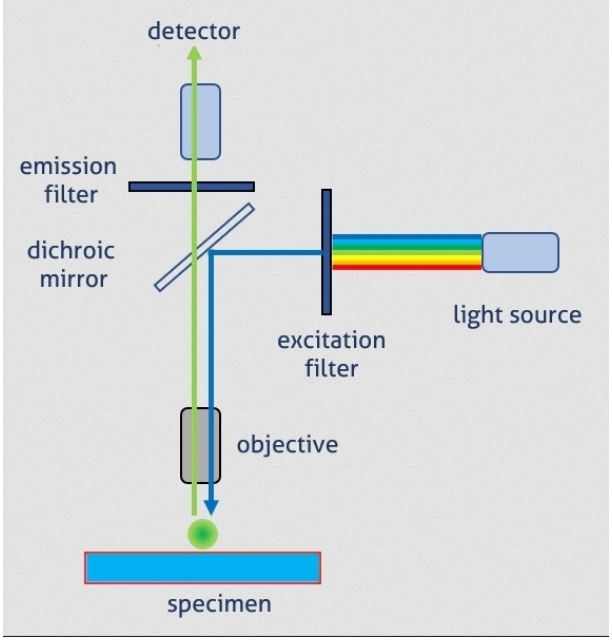


Figure 5. *Optical path for widefield fluorescence microscopy. Source : ptglab.com*

Confocal microscopy Confocal microscopy is another use of fluorescence microscopy. In the case of thicker samples for example, a widefield microscopy method could accumulate blur caused by all the dye molecules stimulated even when out of focus. Confocal technology uses a pinhole to get rid of the signals coming from the higher and lower planes [Fig. 6]. Thus, we keep only the signal of interest from the focal plane.

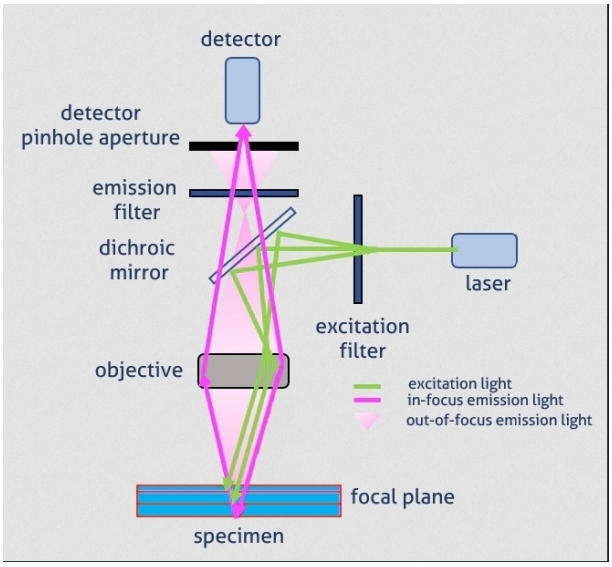


Figure 6. *Optical path for confocal microscopy. Source : ptglab.com*

The "blurry" contribution of an image typically comes from these higher and lower planes. The resulting confocal image is sharper [Fig. 7]. This same property permits volume imaging out of optical sectioning of the sample which means scanning the thick sample on different heights to generate a stack of images. This stack can be turned into a volume out of tri-dimensional reconstruction. Optical sectioning is also used to combine images acquired on different focal plans to get a picture sharp at every levels. Once reconstructed, the object will have a more realistic aspect for being sharp in every point and not blurry out of the focal plan as we could expect.

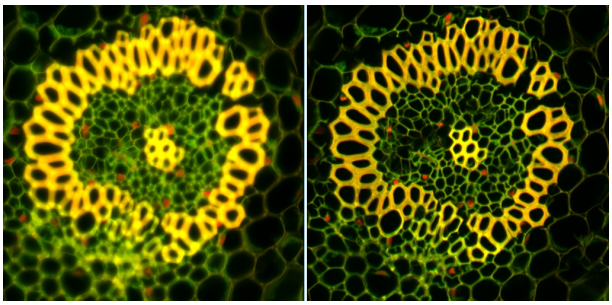


Figure 7. *LSM 510 acquisition, Widefield (left) and Confocal (right) microscopy.*

The other main difference is that confocal microscopy doesn't use the same light source for excitation. Each spot of the image is acquired by a laser scanning in XY plane to reconstruct a 2D image. The signal emitted back from those position is measured to put an intensity value to the corresponding pixel.

The way to capture the signal is the use of detectors. Thanks to a PMT [Fig. 8] the photons reaching for the detector will be amplified and transformed into photo-current, allowing it to be converted to a measure of intensity and therefore into pixel value and data. The sensitivity of the detector also affects the quality of the signal. The two different detector used, MultiAlkali and GaAsP, have their own characteristics. The one that will interest us the most here is the Quantum Efficiency. It indicates the amount of electron converted out of incident photons. The manufacturer Hamamatsu claims a quantum efficiency of 20 to 25 percent in the case of the MultiAlkali detector and a 40 percent quantum efficiency in the case of the GaAsP detector. In addition to that, the NDD-GaAsP detector used on the LSM 510 microscope at IBDM is cooled by a fan making it behave differently to the acquisition sessions (in opposition to the internal-MultiAlkali detector which is not cooled).

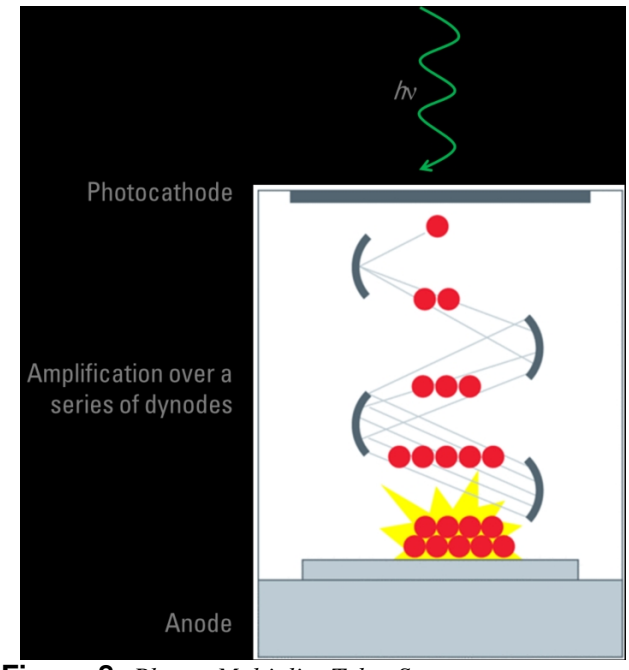


Figure 8. *Photon Multiplier Tube. Source : Leica-Microsystems*

The institute owns different systems and I had the opportunity to practice on some of them such as LSM 510, LSM 780 and LSM 880 from the brand Zeiss. I spent most of the time acquiring images on the LSM 510 system. Some other manufacturers could be cited such as Leica, Nikon or Olympus.

Even though a microscope model can be categorized by its name to tell how old it is (as in which generation it belongs to), it doesn't entirely define it. Its efficiency and usages can also depend on what has been installed on it. I was able to use two different LSM 510 and the difference between them I could observe is significant.

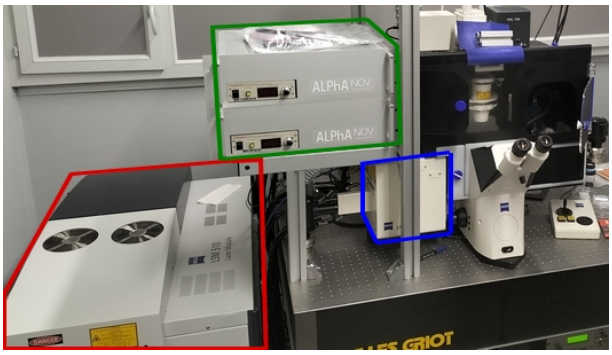


Figure 9. *LSM 510 microscope used to capture most of the data of this internship. The laser unit (red), the GaAsP detectors (green) or even the scan head containing the MultiAlkali detectors (blue).*

One of them has been upgraded with a specific laser and detector (visible on [Fig. 9], indicated as detector GaAsP and laser Mai Tai on [Fig. 10]) added to it, making it able to do two-photon excitation microscopy. The details behind two-photon technique are a bit complex and a detailed presentation would be beyond the scope of this project. A good simplification is to consider it as intrinsically confocal. Regardless to how it works, it gives images of similar quality and properties as a confocal, where the key property is optical sectioning.

For those interested in a more detailed explanation: the technology looks at first very similar to the confocal laser scanning microscope. A laser still scan the sample and that still is how the initial 2D image is created. However this is now a pulsed infrared laser (rather than continuous visible light laser). We still rely on fluorescence excitation, but only in the event of "multi-photon" (generally

two-photons) absorption. This phenomenon is outstandingly rare, and therefore only occurs at the focal point. Indeed, keep in mind we concentrate the light spatially but also in time (pulsed laser, rather than a continuous wave (CW) laser as used in confocal imaging), making the probability of the event likely only at this concentration peak. This allow for the strong (and verified) assumption that excitation only occurs at the focal plane, and that all fluorescent light emitted can only come from the focal plane. Because now excitation only occurs at the focal plane, we use the term "intrinsically" confocal, as a reference to the most famous technique for optical sectioning. In the end, the pinhole principle isn't necessary anymore, which is good because the way it cut out a lot of light (including the one scattered by the sample) is SNR-inefficient. "NDD" (non descanned detector) build up can be used instead. Removing the pinhole while keeping the optical sectioning is one among other advantages of this technique.

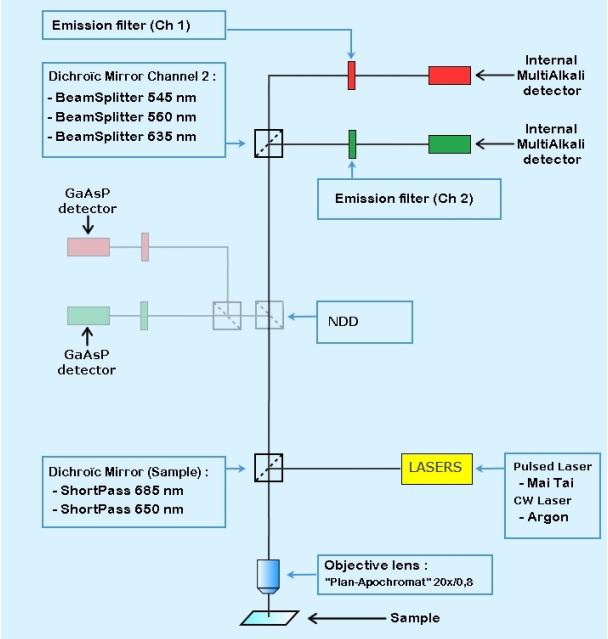


Figure 10. *510 optical path for internal detector. Details of filter here limited to the one used during the project.*

This conclude the presentation of the used instruments. I also would like to introduce two other techniques of interest, which were not used in the project.

First, the LSM 780 microscope contains the so-called "spectral detector". Instead of being sensitive to a predefined wavelength range, the detecting unit is made of 32-channel detectors. This allows a decomposition of the emission signal into its spectrum.

Second, the LSM 880 [Fig. 11] has a supplementary detector called AiryScan. It is another method from the manufacturer Zeiss to make the resolution once again better at the expense of some more time. Instead of having a pinhole to get rid of the information that would cause blur, the surrounding light is also collected by a specific configuration of detectors and processed to build a more precise and accurate image.

Neither spectral analysis or airyscan detectors were used in the presented work. However it is worth mentioning as potential and promising next steps of this project.

Figure 11. *LSM 880 microscope. The workstation next to it allow to pilot the microscope from the computer.*

1.1.2 Confocal microscopy practice

The quality of the microscope used is not only based on the quality of the material it is made of. Some might consider a perfect hardware pointless when it's not geared up with the correct set of tools. The software part of a system can make a whole different user experience from one microscope layout to another one.

Common settings During manipulation, the features I mainly needed to use are:

- to edit the laser power, the detector gain and the scan speed
- to pick the right dichroic mirrors, bandpass filter and detector wavelength

For the laser power, the value selected as a user is a set point, or requested power. It is expressed in percent, as "how much of the available power will be transmitted to the sample". The question discussed here lies in the details. A numerical value gives a feeling of precision, control and reproducibility. However a percentage rise a question: Is the initial value, from which we extract a percentage, also precise and reproducible itself ?

I used a power meter to verify this point. The device is placed at the same position as the sample and is subjected to a laser shot to measure the energy received. For various cause such as the warm up time needed, the room temperature or any other sources affecting the alignment of the laser, the laser power received may not be identical. From a session to another one, even though the laser power has been set to the exact same value on the software it might vary. The room has air conditioning to reduce any effect of the environment on the microscope. In the end, the impacts are generally minimized but not entirely prevented. According to my measures, the laser power received by the sample was indeed stable in our experimental conditions. The power meter indicated 150μw received from the confocal microscope's laser. Note that the two-photon microscopy needs much more power (1000x) in order to generate the two-photon phenomenon a decent amount of time.

This explanation to bring to the point that having a reproducible experiment is an every day struggle and need to be prepared before even booking the imaging system. Being able to reproduce an experiment means following a list of instructions leading to the same result every single time.

Mosaïc This option also called Tile-Scan is a very useful addition to a system. Usually, the microscope will acquire the current field and stop the scanning with, in my case, a 1024x1024 pixels image. Instead, a way larger field can be collected for example by instructing the instrument to collect 25 tiles in a 5 by 5 pattern all put together into a 5120x5120 picture [Fig. 12]. To do so, the microscope will still scan the same field size but it will move the stage once the current field has been scanned to proceed with the next one in the reading order.

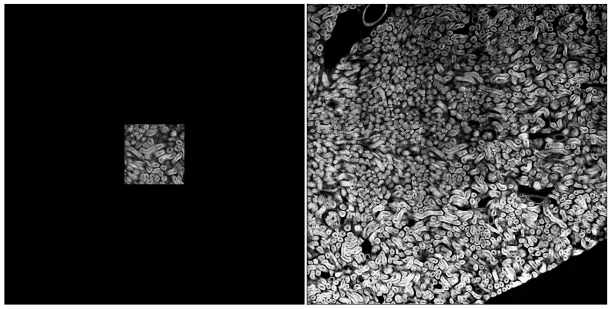


Figure 12. *A single field next to the tile-scan done around it. On the Tile scan, we can notice a great difference in intensity between the top and the bottom of the cumulated fields.*

This practice is great for the amount of data generated automatically but it also can bring a major problem. If the sample is not completely orthogonal to the optical axis, moving away from the area the focus have been made on will cause a change of focal plane. It implies that the signal and its quality will also drop significantly (See Figure 12). On some data, I could measure down to half the signal between the field in the center of the image and the one in the corner.

Z-Stack The optical sectioning evoked previously is able to compensate most slope the sample could follow. At the expense of as much acquisition time for each stack, the image can be collected on multiple z(height) levels forming a z-stack. It is quite costly time wise but it allows to select the slice that is the most interesting to our task. In our case, it was mostly about selecting the most intense of the stack as it translates in more signal emitted hence more quality (taking away the case of saturation).

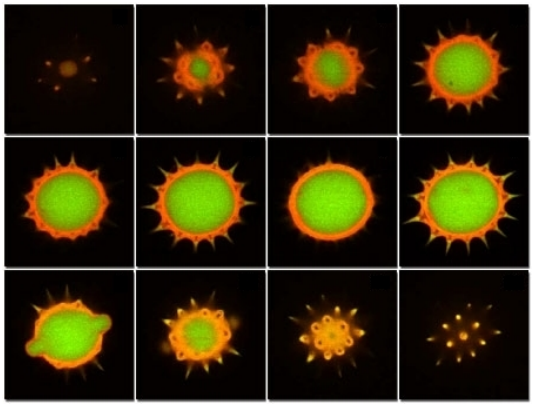


Figure 13. *Confocal Z-Stack of pollen grain*

Signal noise ratio The signal intensity is adjusted by two main factors. The first is the amount of light emitted by the sample out of laser excitation. The second is the detector gain, which amplifies the signal received by the detector.

Both of them has their side effect. When a fluorochrome is emitting a signal out of being activated by a laser, it will become weaker as if exhausted. After a certain amount of emission, it will fade to the point of not emitting anymore. This notion of a fluorochrome getting tired is called bleaching.

The detector gain will not have any effect on the sample (since it happens at the end of the process). It will simply increase all signal it receives. Nevertheless, it is amplifying both signal and noise. The said noise has multiple sources and doesn't follow the gain value linearly. That makes the experiment difficult to be reproducible for having a variating SNR value. Capturing the same field multiple times and making an average of it can reduce the amount of noise but it costs even more time.

1.2 Deep-learning

Deep learning is part of AI and machine learning. It is about guiding a computer on how to achieve a complex task automatically using a large amount of data, a powerful computation system and a neural network based architecture. In the field of image processing, the task deals mostly with image classification, segmentation, localization and generation.

This domain of computer science involves multiple needs such as a certain amount of data that sometimes might be compensated with specific methods or be a real problem when lacking for some given tasks. Finding the right network for the situation can also be challenging. Being bigger for a network can be good, but it can also cause more problems than it solves.

The main resource though is the computational capability of the environment one is working with. When it comes to neural network training, computational units are rated good when they can achieve a greater number of operation at the same time. It is called parallel computing and is a key to deep-learning training. Central Processing Unit (CPU) and Graphic Processing Unit (GPU) are two different type of component a computer usually have. The CPU is made for low latency sequential computation. The GPU is better in our case for both its ability to compute plenty of operations simultaneously and for the CUDA (Compute Unified Device Architecture) cores composing it (in the case of Nvidia devices).

These cores in addition to working together are made for matrix computing, especially good for deep-learning which is made of tensor processing. For that reason, gaming graphic cards might be used even though some cards are designed especially for the kind of calculation done in back propagation with tensor core but at a cost. Since most office computer are not in need of such components, one wanting to get into deep learning could either get a computer designed for it or use a remote access to one through cloud computing (Amazon AWS, Google Cloud, ...).

1.2.1 Programming Environment and Deep-Learning notions

Google Colaboratory When somebody wants to practice programming at zero cost, Google Colaboratory (Colab) could be an option. Basically, it let users have access to a programming environment based on Jupyter notebook technology and using GPU calculation. Notebooks give the possibility to alternate between Markdown cells and code cells, which in my opinion has a great pedagogical value.

Now Colab is a service from Google that let you create or load notebooks and run them in remote access by logging in with a Google account. There are some good points to it. It lets you have access to your notebook from any place, has a Python environment already geared with the different deep learning frameworks (TensorFlow, Keras, PyTorch...), can load file from the Drive service from Google linked to the same account and allow the use of GPU. With the access to the different services such as the Microsoft Office equivalent (Docs, Tabs, Slides) anyone's Google account become a powerful tool with no perceivable cost.

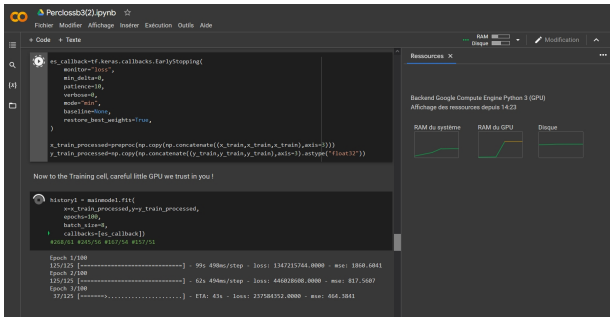


Figure 14. *Example of Colab Notebook with a GPU provided.*

There though are two major downsides that need to be taken in consideration. Whatever is on Google's services is usable by the company which is a major addition to their big data but a concern when the data are sensitive such as patient and clinical data. The second issue one could face is, in the case of Colab, a restricted access. Colab users are allowed a certain amount of time/calculation using a GPU and can get disconnected any time. Some premium offers give access to better GPU and longer sessions but it is still not providing a guaranteed access or a limitless session lifespan. When usually a neural network training could be done on a overnight session, here, the session can end at any moment dumping any progress that could have been made.

The amount of time allowed isn't really a problem since any supplementary Google account will give just as much time so potentially an unlimited quantity of free computation. The only trouble is not having a trustworthy saving method included in the code, method that potentially extend the running time needed for the same task. Nevertheless, it is still the best alternative to the locally available computer provided that you take the habits which are almost mandatory to work with.

1.2.2 Dataset

Since no ready-to-use dataset were available locally in the institute or on internet, I started to design a brand new one to fit our need. I chose to acquire different tissues from standard metrological plates available in the optical imaging facility. For each image, there is an internal-MultiAlkali detector part as input for the networks and a NDD-GaAsP detector part as ground truth. The NDD-GaAsP detector are made for two-photon system and will provide a quality image (ground truth) meanwhile the internal-MultiAlkali detector are not a correct use of the two-photon system which simulates a flawed image. Both were collected simultaneously using the LSM 510 microscope gifted with the two modalities. A great advantage of doing that is getting rid of the necessity for registration task (horizontal, vertical and deformation). Using two different microscopes would cause the sample to be manipulated and call for a registration procedure to properly gather the data. This approach also gave me the opportunity to add more flexibility toward the choice of neural network architectures and learning strategies (either supervised, semi-supervised or unsupervised).

In the beginning, I used samples of metrology marbles. When the network produced correct results in predicting the images, I could add to the dataset the bio-samples one made of a convallaria root slice and the other one of a mouse kidney slice. The point was to have different biological tissue shape and properties within the dataset in order to train the different networks to be more robust to the variability of biological images. Later, I also added some actins samples to the dataset. These extra sample were used to successfully improve the models quality. However, no detailed analysis will be shown on such data in the project. Most illustration shown here are about the kidney. Other predictions are in the annexes.

With the multiple data collecting session on the microscopes, I ended up with 7 images of 5120 x 5120 that I would each cut in 25 smaller fields of 1024 x 1024 and 10 images containing actin filaments of size 1024x1024. In fact, for each of these images I had a stack of 5 images piles up to be sure to capture a higher quality slice of the samples. Because of this, I integrated to the data loading part of my code a flattening part. A practice consists in projecting the maximum intensity among the 5 slices thus increasing the signal but also the noise. Instead, I decided to pick only the brightest slice indicating the slice with the highest amount of signal.

For a good training condition, the dataset has to be split into 3 smaller sets. The training set,

the validation set and the testing set. For the two first I took one image of 5120x5120 of metrology marbles, convallaria and kidney. After turning them into 1024x1024 images, I separated the set into 80% of testing images and 20% of validation images. The rest of the 5120x5120 images have been used as testing set to verify the efficiency of the networks on fields on which they couldn't overfit.

Dataset

Sample	Amount	Dimensions
Marbles	1	5120x5120
Kidney	3	5120x5120
Convallaria	3	5120x5120
Actins	10	1024x1024

Table 1. *As explained before, each image of dimension 5120x5120 is turned into 25 images with a size of 1024x1024.*

1.2.3 Neural Network Architectures

AutoEncoder

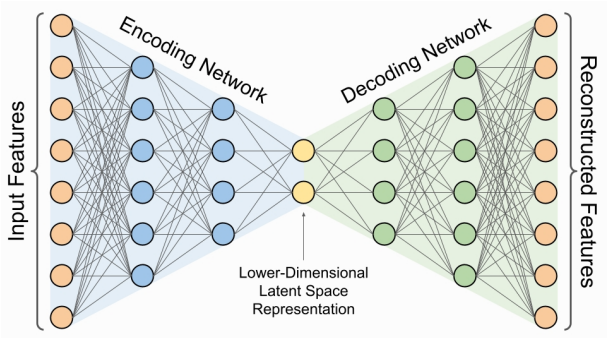


Figure 15. *Convolutional AutoEncoder network. Source : assemblyai.com*

Classical AutoEncoder When it comes to neural network, the neuron layers can be organized into structures. The AutoEncoder is made of two structures stuck back to back together [Fig. 15]. The first part, the encoder, is a sequence of layers proceeding to an encoding of the image being processed. Turning it into an encoding let the network learn efficient data representation (Latent space).

The second part called decoder is conceived to build an image back to the dimension of the original one from the encoding itself. Depending on the initial goal of the network, it will adjust the feature values going through it to produce the requested result. AutoEncoders are doing especially well with denoising. Autoencoder are a class of unsupervised learning architectures. Convolutional AutoEncoder are an extended version of AutoEncoder as they use convolutional neurons in the encoder part to produce the encoding and inverse convolution in decoder part to go back to the original image. For that reason, we use the term Convolutional Neural Network or CNN to talk about the architectures using convolutional neurons. The first networks used in this project are Convolutional AutoEncoders.

Variational AutoEncoder

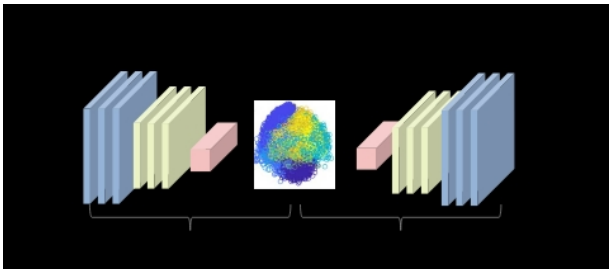


Figure 16. *Variational AutoEncoder network. Source : fr.mathworks.com*

Variational AutoEncoder (VAE) are another type of AutoEncoder family architecture [Fig. 16]. The main difference with AutoEncoders comes from the fact that the structure of the encoding provide a regularized latent space. The idea behind it is then being able to do algebraic operations in order to move from one type of object to another. For instance, the usage of the VAE is often illustrated on the MNIST dataset (a dataset made of written numerals). After training one, we could see the progressive transformation from any character to any other one [Fig. 17] by displaying the components of the latent space. The point in our situation would be to get the network to understand the transformation from a noisy image to a cleaner one.

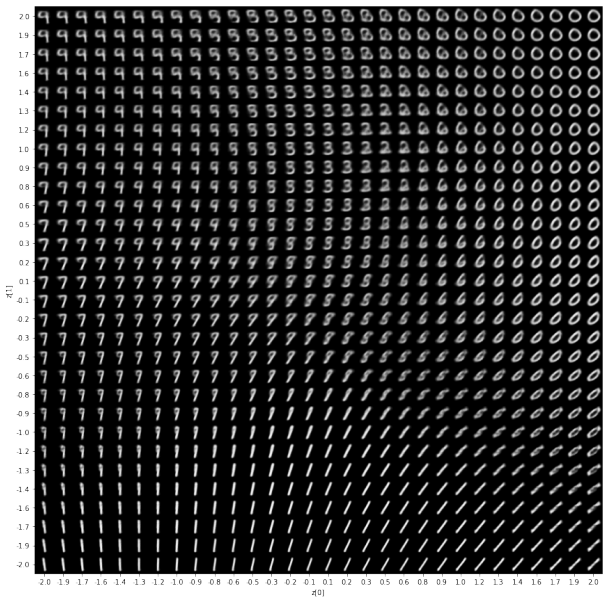


Figure 17. *Representation of the components of the latent space for the MNIST dataset. With the VAE having its latent space regularized, it is possible to project the space into a 2D plan. The image above is the display of the objects in this 2D projection. Here we can see that certain numerals are more represented than some other. More training of the used VAE could help in refining the elements of the latent space.*

This structure of neural network is roughly the one an AutoEncoder would have though it reduces the whole object from the input to a single vector. It then tries to create an output identical to the input out of this same vector. If this is achieved, the decoder part of the network is then able to change any vector into the corresponding object in the latent space. To note, this method doesn't need a ground truth for each image since the goal is the reconstruction of the image provided (unsupervised learning).

U-Net AutoEncoder

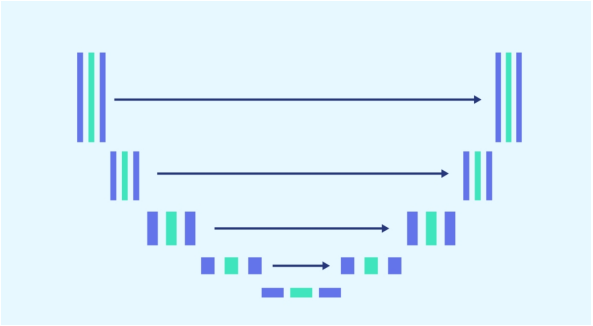


Figure 18. *Basic shape of the U-Net. We can identify the left encoder part and right decoder part of an AutoEncoder. The arrow symbolizing the skip connections is what makes the U-net more efficient than an AutoEncoder.Source: Datascientest.com*

This is one of the most famous architecture for image segmentation. The base of it is a simple AutoEncoder shape spiced up with a feature called skip connections. Those are some kind of bridges from the encoder to the decoder part at a few different level in them [Fig. 18]. It has been implemented to help the decoder in understanding the nature of the input and what it should look like at multiple scales. This greatly improves its efficiency. Though, once the structure is here, the basic blocks it is made of can be replaced with some more complex and tougher ones for the situations where the feature extractions become harder to do. In the current task, I made use of the efficientnet architecture to help the encoder part to store and extract more details contained in the images. These networks are said to be an evolution of the convolutional neural networks. They are optimized for higher accuracy and reduced amount of parameters [Fig. 19]. Given the complexity of setting up such network, I made use of the python library Segmentation-Models made by the GitHub user Qubvel that goes with the real name Pavel Iakubovskii. This also allows the use of transfer learning. This practice consists in loading a pre-trained network (on the huge ImageNet database) to transfer the pre-trained encoder weights into our custom model and then benefits of a powerful way to encode image feature to achieve our own task. The loaded model is likely to have its weights already adjusted near the desired value hence reducing the training time and imposing a strong network regularizer.

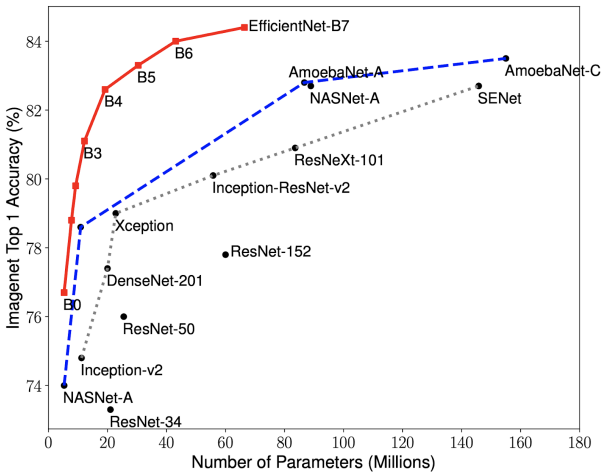


Figure 19. *Comparison of EfficientNet networks with other famous networks. The higher the accuracy is, the better the network is performing. On the other hand, the lower the number of parameters is, the cheaper it is regarding the training time and the resources required.*

It comes in different efficientnet encoder versions ranging from b0 to b7. It has an increasing parameter amount and efficiency as the number goes up. After attempts on using the different versions from b0 to b5, I used either the b0 or the b3 versions. The b5 tend to overfit and be too heavy both for our task and the memory demand on Colab. The b3 have shown good result on the Convolutional AutoEncoder, therefore I kept it for the Perceptual AutoEncoder. I also made use of the lightest, the version b0 as Encoder in the Generative Networks.

Generative Networks

AutoEncoder were once believed to be a neural network structure that could do great at most problems the deep-learning could face. This statement has been greatly reevaluated with the arrival of Generative Adversarial Networks by Ian Goodfellow.

It is a kind of network combining two smaller networks working together for a better result. Playing the same game as cops and thieves, one network in charge of image generation will play the role of the thief by trying to produce fake images. The other network is in charge of detecting whether the image brought to it has been created out of nothing (fake) or if it is an original sample (real).

The main difference with the usual situation is that the discrimination network will provide all the information that helped it telling if it was a correct image or not. Out of it, the generating network will be able to adapt and produce images always closer to the reality or at least, the discriminator's reality. This way both networks will progress and get better through Nash Equilibrium. There is an exception if one of them is able to become too good too quickly.

If the gap between their efficiency widen, the one left behind will not be able to catch up and the training will be stuck. That can lead to a gradient vanishing or exploding or just the inability to leave the local minimum it reached.

The strength of this network is that the image is generated instead of being transformed like an AutoEncoder would. Thus the tasks achievable with the GAN networks can be more complex and still have higher quality.

Pix2Pix A GAN can also be trained so that the class label is a criterion for both the generator and discriminator models. It is called Conditional GAN or cGAN. In other words, the trained generator model can be used as a standalone model to produce images following the class label it has been trained to imitate. When this condition is an image, it is called a Pix2Pix.

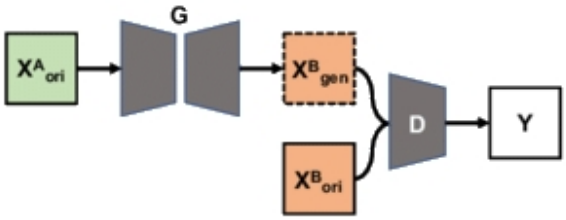


Figure 20. *X^A_{ori} is the input for noisy images , X^B_{ori} for the cleaner ones and X^B_{gen} the fake image we are looking for.*

The Pix2Pix architecture used here is an adaptation of an implementation provided by Jason Brownlee on the website MachineLearningMastery.com along with an explanation on how it works.

For example, this architecture is used for the generation of realistic images out of labels (see Labels to Street Scene or Labels to Facade), grayscale pictures coloration or even the translation of satellite into maps (see Aerial to Map).

Cycle Gan This architecture can also be used for image generation out of a given distribution but it

is benefiting from an additional feature. Just like regular GANs, it will transform the provided distribution into an image and check if it is correctly classified afterward. In addition to that the network will evaluate the inverse transformation, the ability to reproduce the distribution out of the generated image.

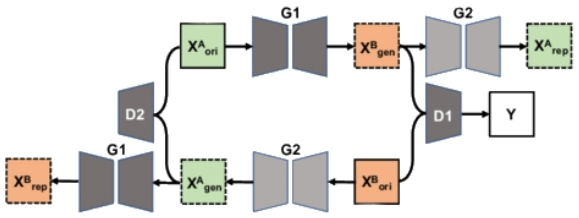


Figure 21. *X^A_{ori} is the input for noisy images , X^B_{ori} for the cleaner ones and X^B_{gen} the fake image we are looking for. In addition to those, there is the symmetrical of this configuration making it possible to train the network on the inverse operation. On this illustration, we can see the cycle giving its name to the network.*

This CycleGAN architecture also has been adapted from an implementation provided by Jason Brownlee on the website MachineLearningMastery.com.

1.2.4 Deep-Learning network regularization tools and hyperparameters

A good management of the hyperparameters and an addition of certain methods can help a neural network to perform better without changing anything about it's core structure. During a training, the perfect model will have low bias and variance error. However, the balance between those is a difficult task. If the network isn't trained enough, its bias error will be high because it will be missing important relations in the features of the dataset. That is also called underfitting. Training the network more will help lowering the bias error value and improve its ability to find patterns in the provided data. Nevertheless, there is a risk of becoming too good at detecting pattern which will cause an increase in the variance error value. A powerful network will find pattern where there shouldn't be, in random noise for example. That can also happend when the dataset isn't big enough. It is called overfitting. These errors can be measured during a training by comparing the efficiency of the network on both the training dataset and validation dataset. For instance, if the validation dataset is processed with difficulties and the training dataset is processed easily, there is high chances of an overfitting. This processing performance is measured differently depending on the task and with the usage of a loss function.

Losses When a network predicts an image, it needs an error metric in order to measure how far from the truth it landed. To do so there are plenty of loss function that need to be carefully picked depending on the task performed. With a segmentation task, the loss to use can be the Dice-Sorensen loss, the Binary cross-entropy (BCE), the Jaccard loss or even a combination of them. This last one, the Jaccard loss is a differentiable version of the IOU (Intersection Over Union) metric. It is made to compare the area of the prediction and the area of the ground truth, dividing how much they have in common by how much they cumulate if united.

For the variational AutoEncoder, I made use of the KL loss (Kullback-Leibler) which is a probabilistic approach of the image comparison and consider their respective distribution together to check for similarity.

The last loss I want to talk about is a classic one I mentioned before called MSE (Mean Square Error) made for pixel to pixel calculation, the square part is made to punish large differences. The same exist with absolute error instead of square to get every distances treated equally and some other one use logarithm not to give that much importance to large errors.

The BCE (Binary Cross Entropy) and any other metrics (such as PSNR or the fast SSIM) that are associated to a differentiable function can be used depending on the measure we are looking for.

Perceptual Loss In order to talk about perceptual loss, there is the neural network called VGG16 to introduce.

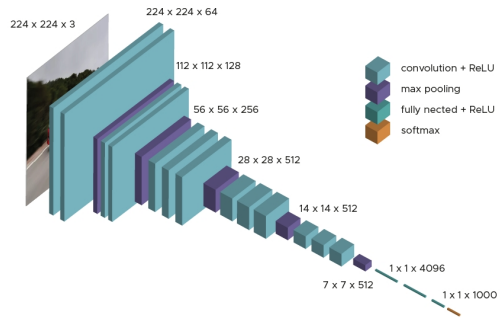


Figure 22. *VGG16 Network. Source: datascientest.com*

This architecture is also quite famous for the huge step forward that could occur in the domain of computer vision thanks to its creation. The idea of stacking multiple 3x3 convolution layers instead of a single higher dimension convolution made the VGG16 score second place in the contest right behind GoogleNet in the 2014 edition of an ImageNet classification challenge on large scale. From my point of view it seems daring to use plenty of small convolutions layers when at the time it was created, the amount of weighted layer limit was really short due to gradient vanishing and no usage/existence for the skip connection technique.

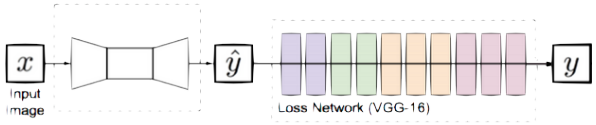


Figure 23. *The VGG-16 concatenated to the network for perceptual loss. Adapted from: deepai.org*

Since the network previously cited is doing great at detecting the structure inside an image for the classification, there is a procedure using it called perceptual loss. A standard loss function is the mean square error (MSE) that will compared the pixel value of the predicted image with the ground truth it is associated with. A problem could occur the moment someone tries to predict an image out of unregistered material.

The prediction will be rated bad for being shifted by a few pixel away from the ground truth and the only solution to lower the MSE loss will be to blur the image for the pixels to be more alike on average. Here comes the previously mentioned procedure consisting in comparing the prediction with the ground truth after passing them through a VGG16 network [Fig. 23], therefore extracting and comparing the structures in the image making the comparison between the image more about how it is perceived than about how it actually is.

ReLU outputs

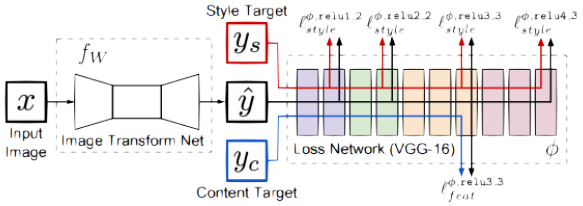


Figure 24. *The perceptual loss structure with multiple outputs. Source : deepai.org*

Instead of simply going through the VGG16 network, the tensor can be extracted at different points of the network. This gives the possibility to check how the network is performing on the perception at different scales. Since the network is shrinking the image down to its characteristics, the earlier the tensor is extracted, the smaller the details it perceived will be.

In order to get a multi scale perceptual loss, the tensor representing the images can be extracted all along the network at each ReLU layers [Fig. 24]. It might though be necessary to point out the heavy memory load of such tensor extraction. Not all computer configuration can convert each images in a batch to 64 times their initial size as is the tensor when arriving at the first ReLU layer.

Early Stopping Stopping the training and restoring the best weights can be an option when the loss is stagnating. Named Early Stopping, it consists in monitoring the validation loss as well as the training loss. When the loss value reaches near the minimum it is able to approach, it begins fluctuating. In the case of overfitting, the validation loss value can start increasing while the training loss value keeps going down [Fig.25].

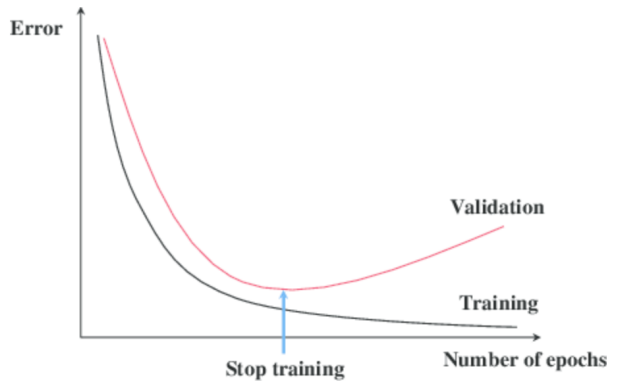


Figure 25. *Typical loss curve for Neural Network training. The blue arrow points at the minimum of the validation loss indicating the best moment to recover the network weights. Source: towardsdatascience.com*

That's the exact moment early stopping becomes useful for saving and restoring the best weight the network generated (when the validation loss value was the lowest) before the training goes wrong by overfitting. Some would though say that a perfectly constituted model wouldn't be able to overfit and that in theory it would just stop progressing when it arrived at it's lowest possible loss function value. On the other hand, some would say that no model is perfect enough and that it must end up overfitting. There is an early stopping callback on every network trained therefore such curve wasn't generated or visualized during this study.

Learning rate When progressing through the loss lowering with the help of gradient descent, the learning rate is the size of the step done following the slope. A larger step will help progressing toward the minimum value faster though a step being too large can go over and miss the result aimed. On the other hand, a step too small risk to progress too slowly. A slow progression can cause to get stuck in a local minimum. It becomes a problem when the local minimum is too far from the global minimum.

One could not bother and just wait for the temporary minimum to be reached on an important value for the learning rate and then lower it on a second session for it to become more accurate. Something similar can be done by creating a function to lower automatically the learning rate also called Learning Rate Scheduler. Either it lowers itself every certain number of epochs or something more pointy. For instance, waiting for the loss to be stagnating before lowering it automatically.

Optimizers When it comes to guiding the neurons weights toward the goal on minimizing the loss function, the gradient descent can follow multiple approaches, all of them being a tweaking of the original method. These tools called optimizers have different perspectives to achieve the same task [Fig. 26] and we can name SGD (stochastic Gradient Descent) or Adam (Adaptive Momentum) as the most famous and quite a list of other name like Adagrad, Adadelta or RMSprop.

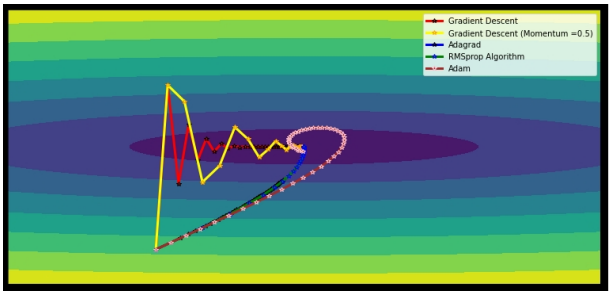


Figure 26. *Comparison of different optimizers. We can see some optimizers are progressing way faster than some others. The slower ones tend to do better at unstucking themselves from local minima and for that reason might be prefered.*

During the internship, my tutor indicated a few papers talking about AdamW. This implies the notion of weight decay correction because using Adam can cause troubles generalizing the models.

More data Every situation has its own difficulties and sometimes depending on the task and the network used for it, there is no other solution than getting more data for the network to be able to train properly. If the user expects the network to be able to generalize an image processing, it might need to see each cases at least once to be able to treat any similar samples in the future.

Augmentation

Figure 27. *Augmentation on a RBG image. Source : Albumentation*

When the data are not really lacking but you still are not sure about the toughness of the operation the network learned, there is a method to reinforce the dataset by altering it with all kind of transformations [Fig 27]. Applying something like a 180 degrees rotation to an image will result in a brand new image from the network sight. This whole labor also has the benefit to partly prevent the overfitting to happen too early. Nevertheless, it is not always a free addition to the training and need to be done carefully.

A common example given is a 180 degrees rotation on the characters of the MNIST dataset (made of the figures from 0 to 9). The entire learning can be messed up when applied to the 6 or the 9 turned into each others. This would drop the success rate on those by roughly 50%. It is also a problem for other characters since a 7 or a 4 seen upside down have no meaning and are not likely to be seen anywhere by the network afterward. This is just adding a work load to the training for no benefits. In this case the solution sounds easy, limiting the rotation to plus or minus 20 degrees, but it is not that obvious for the long list of augmentation applicable to the data.

It still is a near unlimited source of image once a combination of the right augmentations are done. While generating the augmented image, each transformation has a probability to be done. It permits different possible mixes for the same image therefore more image variability from the neural network point of view. To help with that, there is a library called Albumentation implementing plenty of transformations that can be simple image shifts to more complex ones with random rain or snow (useful to create a network for weather correction for example or something like classification of road signs despite any kind of weather).

In this project, the dataset can be augmented with various augmentations. A rotation, optical deformation, translation or such are the kind of things that can occur by mistake during an acquisition which means these augmentations are a good thing to add. However, an augmentation like channel swapping for RGB images or inverting the pixel value can be dangerous and shouldn't be used in our case.

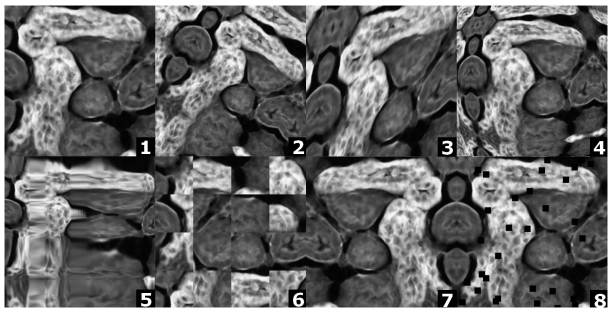


Figure 28. *Augmentation used in this project on a kidney image from the dataset. The transformations applied are listed right after the picture.*

1. *No Transformation*
2. *Shift-Scale-Rotate*
3. *Elastic Transform*
4. *Optical Distortion*
5. *Grid Distortion*
6. *Random Grid Shuffle*
7. *Horizontal Flip*
8. *Cutout*

Tiling The following practice is good on multiple levels. It consists in cutting the image into pieces before proceeding with them. When the memory capacity could allow only one image at a time, the gradient descent within a single batch wouldn't even exist and the evolution of the loss over time could become way more erratic.

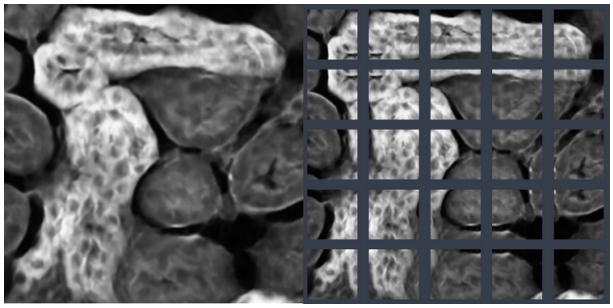


Figure 29. *Kidney sample cut into smaller images through the use of a tiling procedure to help on the resource management. We can notice the sides of each tile to be identical to the tiles around it thanks to the overlap feature.*

After a tiling procedure, the same image could be turned into 16 smaller images [Fig. 29], making it possible to smooth the loss value lowering. It is also possible with tiling to set up an overlapping strategy between consecutive tiles. In fact, convolutional networks can struggle to get a correct edge rendering. The overlap on two neighbor tiles could allow giving up on a part of each tile to get rid of this poor edge quality and get a clean assembled image.

Test Time Augmentation On certain situations, the network will have trained better in certain region, shades or scales. To take advantage of this, the once trained network can predict numerous versions of the same image to which augmentations have been applied.

Once all these predictions have been generated, the results are put together with help from a mean or a threshold function to determine with more accuracy the result expected. These augmentations need to have an inverse transformation to later properly turn it back to the original image.

2. Results and Discussion

2.1 Results

During the whole internship, I could learn and practice various methods and set up different prototype in order to have all kind of approaches to reach the main goal. Doing so led to multiple results.

2.1.1 Segmentation

On the first place, I had little to no notions of programming with Tensorflow/Keras framework. The lesson and practice classes that stood during the first semester were oriented toward PyTorch framework. To get to acknowledge the usages and practices coming with deep learning programming using TensorFlow, I got challenged to do segmentation on mitochondria dataset [Fig. 30].

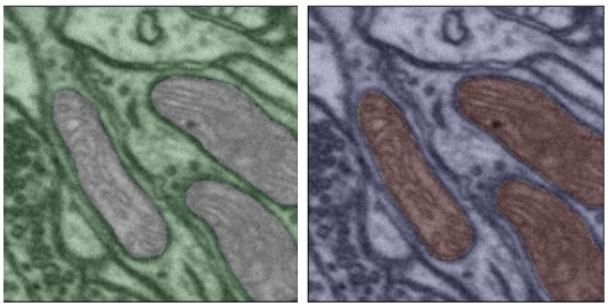


Figure 30. *Mitochondria segmentation with U-Net, Ground Truth(left) and prediction(right). Mitochondria Dataset from www.epfl.ch*

After encouraging results, I felt a need to go through the project previously offered during first semester and to do it again properly. The objects to segment on this task was a prostate [Fig 31].

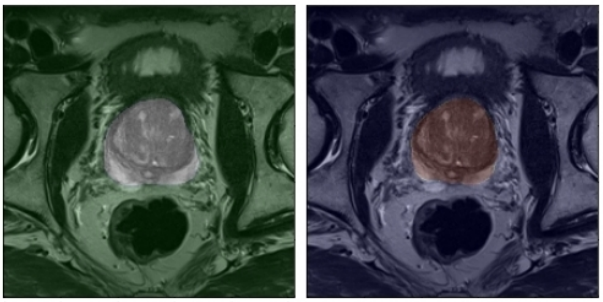


Figure 31. *Prostate segmentation with U-Net, Ground Truth(left) and prediction(right). MRI Prostate Dataset with segmentation from medicaldecathlon.com*

VAE After quite some time and more experience, I worked on getting a variational AutoEncoder to

train and generate images properly. I succeed on getting some significant progress with the marbles becoming rounder over time [Fig 32]. However if the reconstructed image is not good enough, the quality of the components of the latent space will be affected. Because that's the point of we decided to move on to another approach using AutoEncoders. By making use of a classical Convolutional AutoEncoders networks, the reconstructed images will look better than the ones produced with the VAE approach. This though is not a comparison to do. Despite its lower visual quality, the point of using the VAE was to get a feature extraction of the noise and therefore being able to do the transition from the very noisy pictures to some with better signal/noise ratio. Even if there is a latent space in both case, the one from the VAE network is regularized and allow a wandering along vector's value (which implies it is not possible to do the same with an AE).

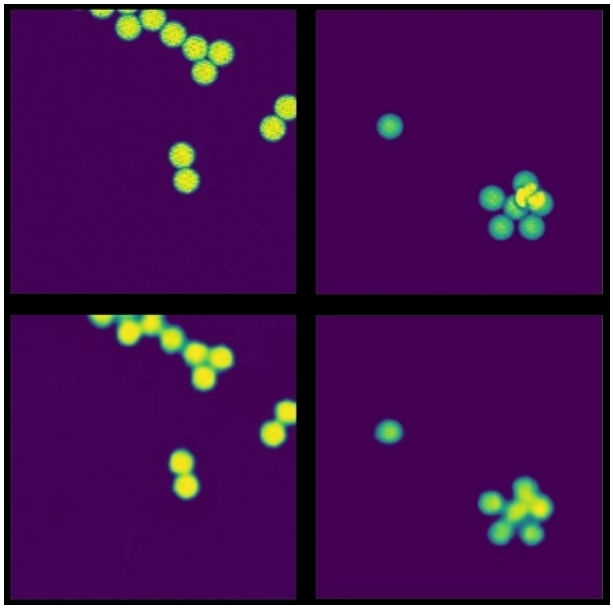


Figure 32. *Confocal image of* $4\mu m$ *beads (top) and their predictions with the VAE (bottom). Illustration on two kind of images, one noisier (left) and one cleaner (right). It is quite noticeable how differently the network reconstructed them differently.*

2.1.2 Measurements

There is plenty to say with metrics and ways to measure any image. The major distinction is if there is a reference to compare the image with or not. With a single image, you can determine its average and standard deviation (STD) to have a rough insight on the content of the image. A large STD value tends to indicate a noisy image when the average value could tell the amount of information the image holds.

This though is not an absolute rule for the simple reason that a "perfect" standard deviation value would mean zero variation in an image and a "perfect" average value would be the highest so the ideal image with these measures is made of 100% pure white pixels. I had the idea to detect the contours in the image with a gradient module detection out of a bidirectional sobel filter.

The amount of contours could indicate the sharpness contained in the images. After doing so, two conclusion could be given. A blurry image will have a really low sharpness value, making the metric useful. On the other hand, there is a negative note due to noise. A noise like salt&pepper will have a huge impact on the measure. It will create local peaks inflating the gradient module amount. Despite that, I created two metrics out of this gradient module. The first is the comparison pixel to pixel of the gradient module images through MSE (MSEMG). The second one is sharpness which is the sum of the gradient modules in an image thus a low value indicate a blurry image.

The other idea consisting in comparing with an ideal image permits usage of metrics such as Peak Signal-Noise-Ratio (PSNR), Structure Similarity (SSIM) or even Mean Square Error (MSE).

We could discuss over whether a measure such as MSE is a good idea or not since an identical image will score very few points if shifted in a direction by any number of pixels. The major concern is obtaining an ideal image usable for comparison (ground truth).

In my case, the image used is the one with higher SNR acquired through NDD-GaAsP detector usage thanks to their better quantum efficiency (40%). It for sure helps improving the lower quality ones captured thanks to internal-MultiAlkali detector (25% quantum efficiency). Our questioning is that it could refrain the network from progressing beyond the best quality available. This same ground truth quality concern raises another question about

the metric evaluation. A perfectly accurate image will be very different pixel to pixel with the NDD image and have a low score in MSE for instance.

- The formula for the MSE metric is:
  $MSE(x,y) = \frac{1}{n}\sum_{i=1}^{n}(x_i - y_i)^2$
- The formula for the SSIM metric is:
  $\mathrm{SSIM}(x,y) = l(x,y)^{\alpha} \cdot c(x,y)^{\beta} \cdot s(x,y)^{\gamma}$
  Its components are:
  Luminance: $l(x,y) = \frac{2\mu_x\mu_y + c_1}{\mu_x^2 + \mu_y^2 + c_1}$
  Contrast: $c(x,y) = \frac{2\sigma_x\sigma_y + c_2}{\sigma_x^2 + \sigma_y^2 + c_2}$
  Structure: $s(x,y) = \frac{\sigma_{xy} + c_3}{\sigma_x\sigma_y + c_3}$
  Where μ_x is the average of the image x and σ_x is the standard deviation. And these components are weighted depending on the value of the α, β and γ.
- The formula for the PSNR metric is:
  $PSNR(x,y) = 10 \cdot \log_{10}\left(\frac{MAX(x,y)^2}{MSE(x,y)}\right)$
- The formula for the CV metric is:
  $c_{\mathrm{v}}(x) = \frac{\sigma_x}{\mu_x}$
  Where μ_x is the average of the image and σ_x is its standard deviation.
- The formula for the MSEMG metric is:
  $MSEMG(x,y) = MSE(MG(x), MG(y))$
  with the gradient module:
  $\mathbf{MG}(\mathbf{I}) = \sqrt{\mathbf{G}_x{}^2 + \mathbf{G}_y{}^2}(\mathbf{I})$
  Gradient determined with the following convolutions:
  $\mathbf{G}_x(\mathbf{I}) = \begin{bmatrix} +1 & 0 & -1 \\ +2 & 0 & -2 \\ +1 & 0 & -1 \end{bmatrix} * \mathbf{I}$
  $\mathbf{G}_y(\mathbf{I}) = \begin{bmatrix} +1 & +2 & +1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{bmatrix} * \mathbf{I}$
- The formula for the Sharpness metric is:
  $Sharpness(x) = \frac{1}{n}\sum_{i=1}^{n}(MG(x))$

NDD GaAsP Image vs Itself

	(1)	(2)	(3)	(4)
MSE	0.000	0.000	0.000	0.000
MSEMG	0.000	0.000	0.000	0.000
SSIM	1.000	1.000	1.000	1.000
PSNR	∞	∞	∞	∞
NDD GaAsP Image				
CV	0.686	0.711	2.492	2.902
Sharpness	113.63	109.29	31.668	38.797

Internal-MultiAlkali vs NDD-GaAsP Images

	(1)	(2)	(3)	(4)
MSE	97.056	97.473	34.795	31.435
MSEMG	4.198	4.387	3.819	3.909
SSIM	0.254	0.196	0.523	0.398
PSNR	28.261	28.242	32.716	33.157
Internal-MultiAlkali Image				
CV	0.774	0.747	2.255	2.627
Sharpness	162.91	199.74	68.813	73.415

Table 2. *Measurements for the original images acquired on the microscopes and used for the predictions afterward. The datasets (1) and (2) are kidney samples while (3) and (4) are convallaria rhizome samples. The images display afterward to illustrate the kind of image studied with the metrics are a small part of the dataset (1). The metrics analyzed the entire testing dataset. About the individual metrics: As mentioned before, the noisy image scored high in sharpness. The other metric provided, Coefficient of Variation (CV), is a SNR indicator calculated as Standard Deviation divided by the average of the image. The higher the CV is, the noisier the image gets.About the comparative metrics: The values from both Tables will be used as scale for the values measured on the network predictions. The NDD-GaAsP image table correspond to the measure on ground truth (the ideal ones) and the Internal-MultiAlkali image table to the reference measurements (the ones to improve). There is also in the annexes the measurements done on conventional filters (i.e. gaussian, mean and median).*

Convolutional AutoEncoder

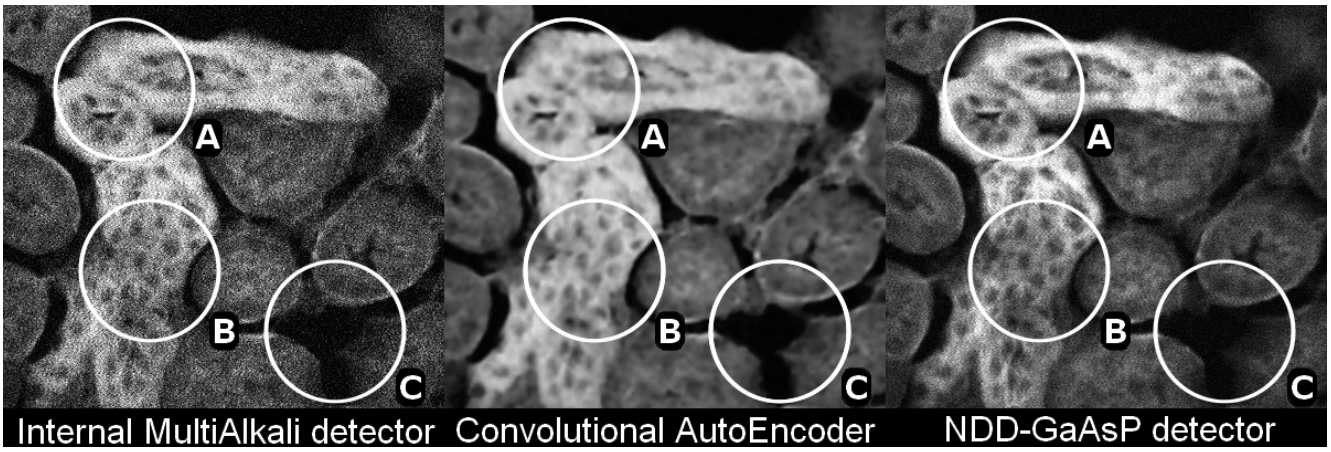


Figure 33. *Acquisition of kidney by internal-MultiAlkali detector (left), prediction of Convolutional AutoEncoder (middle) trying to mimic the same acquisition by NDD-GaAsP detector (right, ground truth). Zoom on a particular area of interest (AOI). This area allow to observe high intensity patterns (A), gradient of middle intensity pattern (B), and low intensity, highly noisy pattern (C). The same AOI will be used for further illustration of different models.*

The ability of the model to preserve or on the contrary to modify this specific patterns is of as much interest as the overall analysis provided by the metrics (Table 3.). One can for instance observe how in the circle C area, the prediction create sharp edges between the black and low signals, which are absent from both the raw data and the ground truth. In a similar way, the circle B area seems to provide an up to down gradient (high to less high intensity) in raw and ground truth data, which disappear or at least fainted in the prediction. In the area A circle, the model tends to flat out the signal and set it to saturation. All these phenomena illustrate unwanted data modification.

This first network gave promising results such as the reproduction of large structures. The major good point is the removal of the noise. The Gaussian noise seems to have completely been smoothed. Another good point: The edges between the background and the different component of the image are sharp. There is however a global blur on the image within the texture of the visible objects.

CAE Prediction Image vs NDD Image

	(1)	(2)	(3)	(4)
MSE	84.556	86.334	-	22.026
MSEMG	3.959	3.892	-	3.320
SSIM	0.478	0.471	-	0.813
PSNR	28.859	28.769	-	34.701
CAE Prediction Image				
CV	0.644	0.592	-	2.725
Sharpness	51.646	58.310	-	26.917

Table 3. *Refer to the Tab.2 except this is the metrics of the prediction of the Convolutional AutoEncoder. In this particular case, the convallaria sample image (3) has be removed from the comparative table. It was used as training set in this network's training and would cause biased results. The metrics indicate an overall improvement compared with the same measurements on the internal detector image except for the sharpness line. The green numbers indicate performing better than any other network on this specific evaluation. On the opposite, the red values indicate scoring the worst score among the neural network predictions. See page 25 for a summary of all the comparative tables.*

Perceptual AutoEncoder

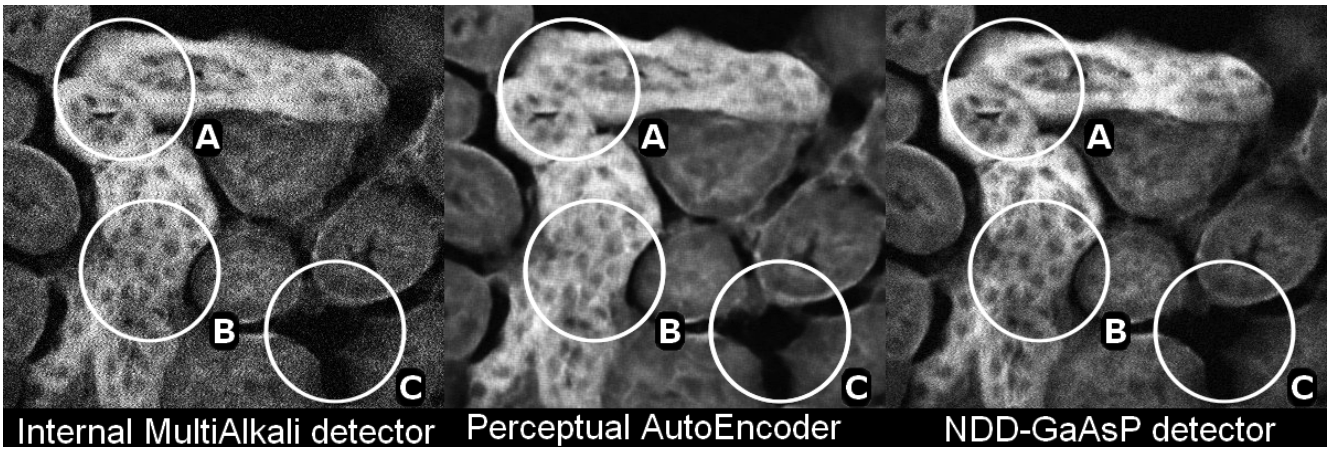


Figure 34. *Refer to Fig.31, with the difference of showing prediction of AutoEncoder with perceptual loss (middle). Note how on the previously highlighted patterns, the sharp edges (C) or the high intensity profile (A) is now closer in the prediction to both the row data and ground truth. The evolution on the B area however is still not satisfying.*

This approach with perceptual loss intends to reconstruct the structures of the image with more accuracy.In order to help the network in detecting and comparing the structures within the predicted image with the ones in the picture fixed as goal, there is a loss function that involves another pre-trained neural network. It consists in comparing the images but not directly pixel wise like a MSE loss would do but by checking the difference between the vector they have been encoded in by a classifier network. Here it helped getting more details in the image. The network seem to have tried to imitate the noise with regular patterns that can be seen everywhere on the textures of the image. Note that this network would certainly be able to learn a lot better than the CAE without perceptual loss on a dataset that hasn't been perfectly registered.

Perceptual AE Prediction Image vs NDD Image

	(1)	(2)	(3)	(4)
MSE	98.053	81.617	44.164	48.303
MSEMG	4.002	3.904	3.778	3.942
SSIM	0.410	0.439	0.439	0.417
PSNR	28.216	29.013	31.680	31.291
Perceptual AE Prediction Image				
CV	0.663	0.602	2.349	2.014
Sharpness	46.553	52.245	24.986	27.215

Table 4. *Refer to the Tab.2 except this is the metrics of the prediction of the AutoEncoder with Perceptual Loss. There is no metric showing this network as better than the others. Even if the visual quality is with no doubt improved, this network trained both on MSE and Perceptual Loss with the usage of a VGG16 network. This implies that the pixel to pixel error increased in order to land on a compromise with the perceptual interpretation from the VGG16.*

Pix2Pix

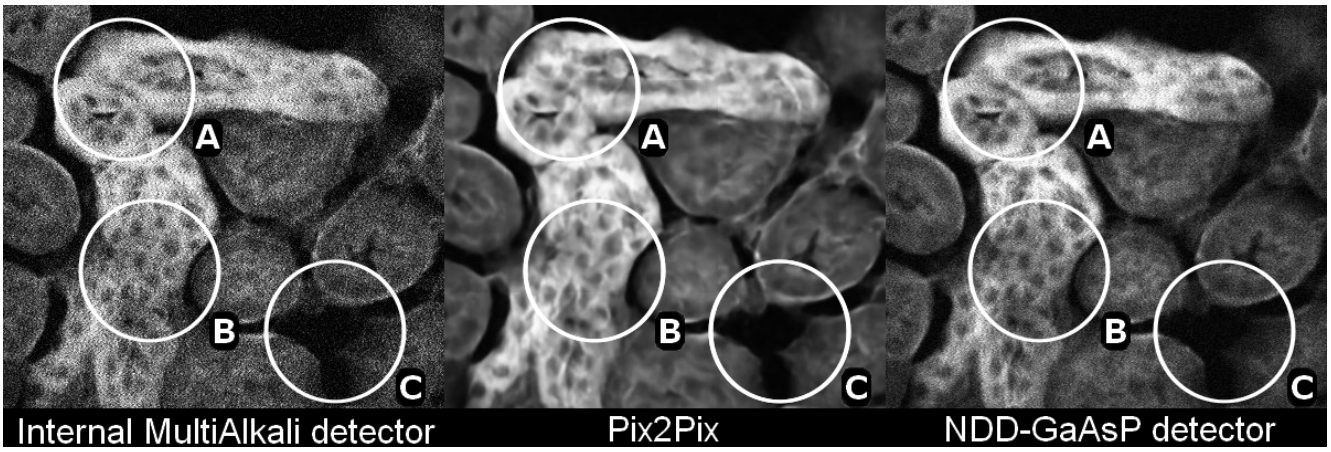


Figure 35. *Refer to Fig.31, with the difference of showing prediction of the Pix2Pix (middle). Note how on the previously highlighted patterns, the sharp edges (C) or the high intensity profile (A) is now closer in the prediction to both the row data and ground truth.*

The evolution on the B area seem nearer from the ground truth even if not perfect yet. The prediction picture overall seems extremely nice looking (not noisy). The very refined patterns are, according to the specialists, perfectly realistic and comparable to what could be obtained when using an SNR optimized acquisition or well-known data processing techniques (deconvolution) (not shown). Such an end result is very promising and satisfying, but one should also keep in mind that the ability of a model to do "better than the ground truth" is also a strong warning flag in deep-learning modeling.

The GAN approach is the most trending. The structures like AutoEncoders are becoming quite old in the history of Deep-Learning and they tend to become component of larger network. Moreover, AutoEncoder are know to have several drawbacks dealing mainly with the tendency to blur output image. This problems are now known to be solved by using such GAN networks.

In this Pix2Pix, the generator is a convolutional AutoEncoder (U-Net). Even though there is a lack of texture similar to the CAE, this could have a better justification here. The noise is again missing on this prediction, however, it doesn't seem to have cleaned any necessary details from the original image.

Pix2Pix Prediction Image vs NDD Image

	(1)	(2)	(3)	(4)
MSE	87.697	81.398	26.964	23.089
MSEMG	3.932	3.872	3.132	3.269
SSIM	0.479	0.478	0.830	0.815
PSNR	28.701	29.027	33.823	34.497
Pix2Pix Prediction Image				
CV	0.702	0.616	2.217	2.663
Sharpness	54.012	55.820	36.398	31.272

Table 5. *Refer to the Tab.2 except this is the metrics of the prediction of the Pix2Pix GAN. Most metric indicate this network as greater than the others. Despite all the good ratings when compared with the NDD image, it has be defined as the one with the highest CV on the kidney (1) image. This is making the CV a questionable metric to measure the noise since no other image seem visually cleaner. Nevertheless, it is still and improvement of the initial image even in this measurement.*

CycleGAN

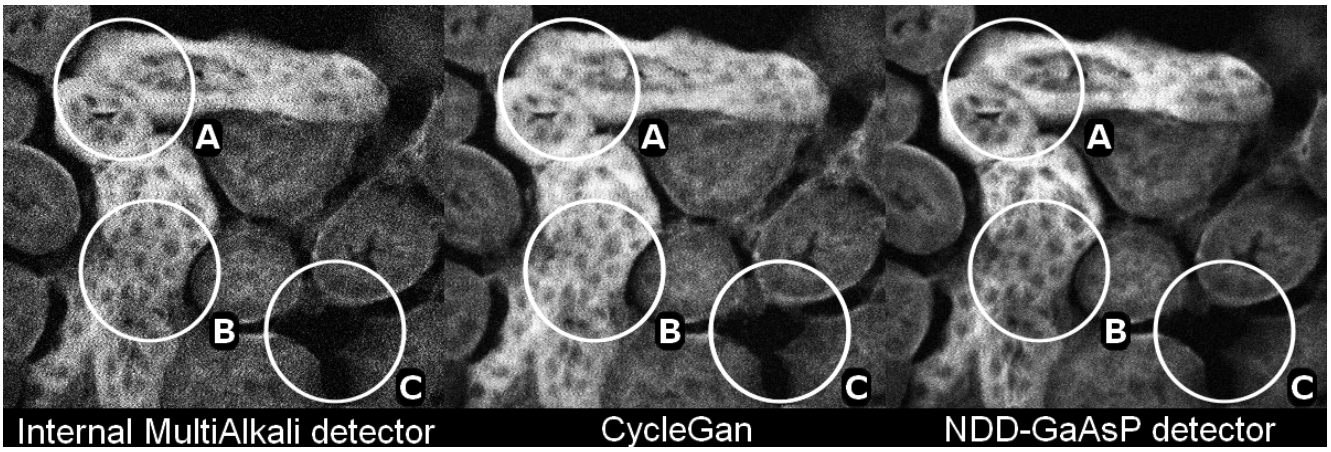


Figure 36. *Refer to Fig.31, with the difference of showing prediction of the CycleGAN (middle). The textures of the image here seem to be correctly restored. However this network seem be generating a more saturated image which is quite noticeable on the (A) area. On the (B) and (C) area, the model tend to overestimate and artificially amplify the lower intensity part of the patterns. This is also visible on the metrics [Tab.6] through the sharpness measurement.*

In this case, I restricted the training set to a single type of image. Since the CycleGAN is basing its training on images that are not the exact ground truth of the input image, it needs to compare the nature of both category of images. For that reason, I kept only the kidney data for the task to be more realistic. That also bring up the fact that it is semi-supervised: It needs the flawed image dataset, the ground truth for it but not registered as it will compare the nature of images and not actually the content. This network was the only one able to entirely reproduce a realistic noise. Even with debatable image quality, this is still a really interesting result for being completely different from the predictions generated before hence able to achieve different tasks.

CycleGAN Prediction Image vs NDD Image

	(1)	(2)	(3)	(4)
MSE	91.610	90.780	47.094	49.261
MSEMG	3.916	4.003	3.444	3.599
SSIM	0.361	0.301	0.412	0.317
PSNR	28.511	28.551	31.401	31.206
CycleGAN Prediction Image				
CV	0.694	0.651	1.690	1.976
Sharpness	111.74	135.60	49.548	52.531

Table 6. *Refer to the Tab.2 except this is the metrics of the prediction of the CycleGAN. Most metric indicate this network as worse than others. In spite of that, because of the nature of the image, it is to note that it was able to create a realistic noise while still improving the original image. This same noise is what make it score so high in the sharpness metric (by adding a large amount of edges).*

Recap of predictions for comparison

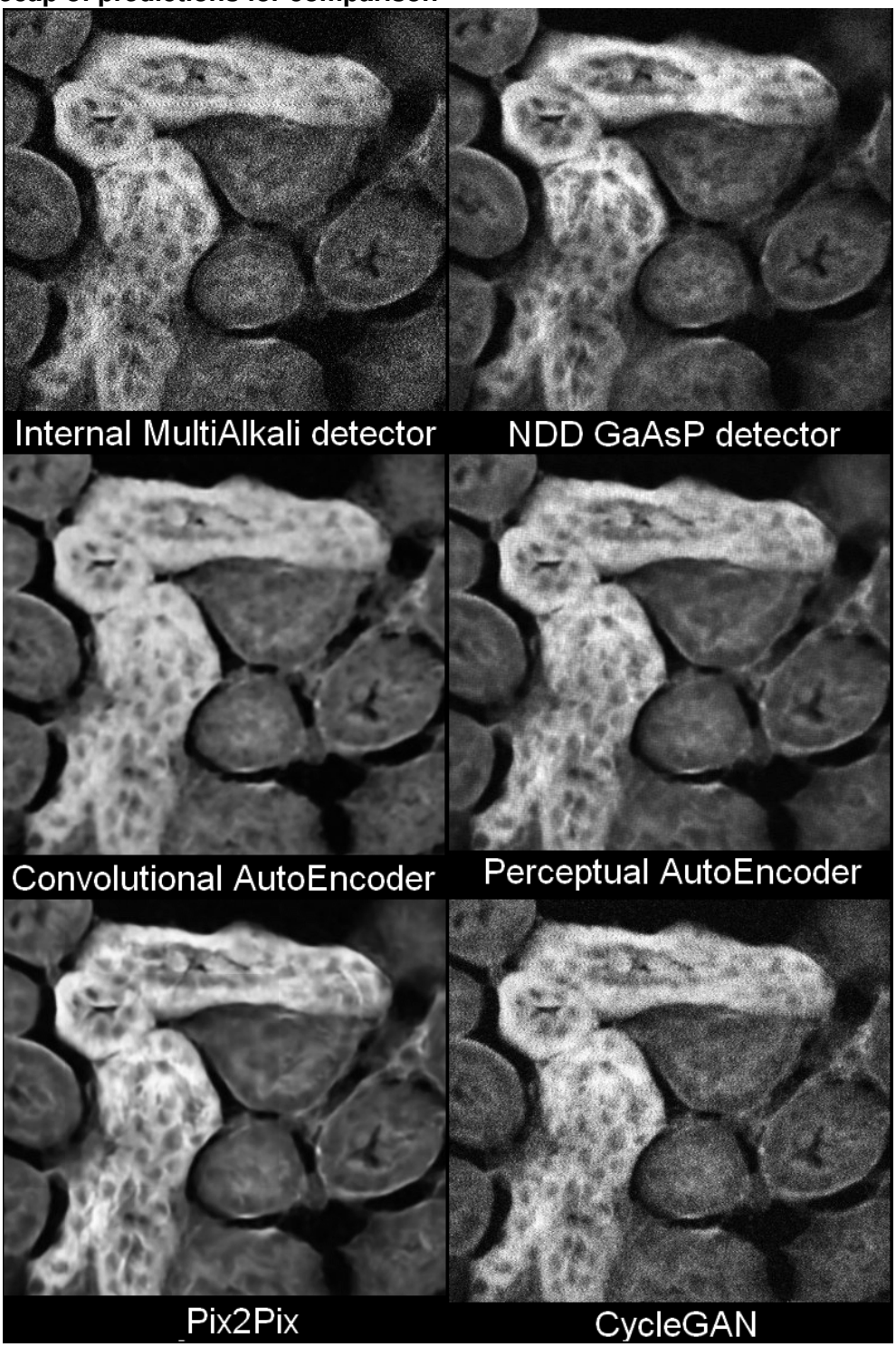


Figure 37. *Summary of the predictions. These are the same pictures as Figures [33-36] but concatenated to simplify the model comparison.*

Recap of measurements for comparison

Internal Detector Image vs NDD Image

	(1)	(2)	(3)	(4)
MSE	97.056	97.473	34.795	31.435
MSEMG	4.198	4.387	3.819	3.909
SSIM	0.254	0.196	0.523	0.398
PSNR	28.261	28.242	32.716	33.157
Internal Detector Image				
CV	0.774	0.747	2.255	2.627
Sharpness	162.91	199.74	68.813	73.415

NDD Image vs Itself

	(1)	(2)	(3)	(4)
MSE	0.000	0.000	0.000	0.000
MSEMG	0.000	0.000	0.000	0.000
SSIM	1.000	1.000	1.000	1.000
PSNR	∞	∞	∞	∞
NDD Image				
CV	0.686	0.711	2.492	2.902
Sharpness	113.63	109.29	31.668	38.797

CAE Prediction Image vs NDD Image

	(1)	(2)	(3)	(4)
MSE	84.556	86.334	-	22.026
MSEMG	3.959	3.892	-	3.320
SSIM	0.478	0.471	-	0.813
PSNR	28.859	28.769	-	34.701
CAE Prediction Image				
CV	0.644	0.592	-	2.725
Sharpness	51.646	58.310	-	26.917

Perceptual AE Prediction Image vs NDD Image

	(1)	(2)	(3)	(4)
MSE	98.053	81.617	44.164	48.303
MSEMG	4.002	3.904	3.778	3.942
SSIM	0.410	0.439	0.439	0.417
PSNR	28.216	29.013	31.680	31.291
Perceptual AE Prediction Image				
CV	0.663	0.602	2.349	2.014
Sharpness	46.553	52.245	24.986	27.215

Pix2Pix Prediction Image vs NDD Image

	(1)	(2)	(3)	(4)
MSE	87.697	81.398	26.964	23.089
MSEMG	3.932	3.872	3.132	3.269
SSIM	0.479	0.478	0.830	0.815
PSNR	28.701	29.027	33.823	34.497
Pix2Pix Prediction Image				
CV	0.702	0.616	2.217	2.663
Sharpness	54.012	55.820	36.398	31.272

CycleGAN Prediction Image vs NDD Image

	(1)	(2)	(3)	(4)
MSE	91.610	90.780	47.094	49.261
MSEMG	3.916	4.003	3.444	3.599
SSIM	0.361	0.301	0.412	0.317
PSNR	28.511	28.551	31.401	31.206
CycleGAN Prediction Image				
CV	0.694	0.651	1.690	1.976
Sharpness	111.74	135.60	49.548	52.531

Table 7. *Summary of the metrics for every models. All networks show improvement according to these metrics, the Pix2Pix stands out both visually and on measurements.*

2.1.3 Instrument defects

In order to evaluate the capability of the trained neural network, we decided to voluntarily trigger common misadjustment on the microscope. Indeed, users might prepare their acquisition with settings that will corrupt the image generated.
In a first time, when scanning the scanning speed can be nearly doubled by putting the scanning direction into bidirectional. In certain cases, that can generate a shearing artifact on the image that we exaggerated here to display and see the limits of the networks. There is a second artifact that is created by acquiring only one line on two (or more) called linestep. The resulting image in automatically interpolated by the microscope before generating the image. However, the interpolation corrected most of the problem and the difference with the image without any artifact isn't blatant. Finally, we put oil immersion liquid in place of the air gap between the front lens of an objective that is not made for immersion. Indeed, several kind of lenses exist, with or without immersion media. Putting the wrong media on a lens is a typical user mistake. The expected result is a blurry image because of the higher refractive property of oil.

The measurements for the comparison with the degraded images are in the annexes. Those could be used for a comparison with a future version of the networks with a training including these new types of images to improve. The example shown below is the most visual, bidirectional scanning, and the architecture used is the one of the Pix2Pix network which performed best on regular predictions. All other examples can be found in the annexes.

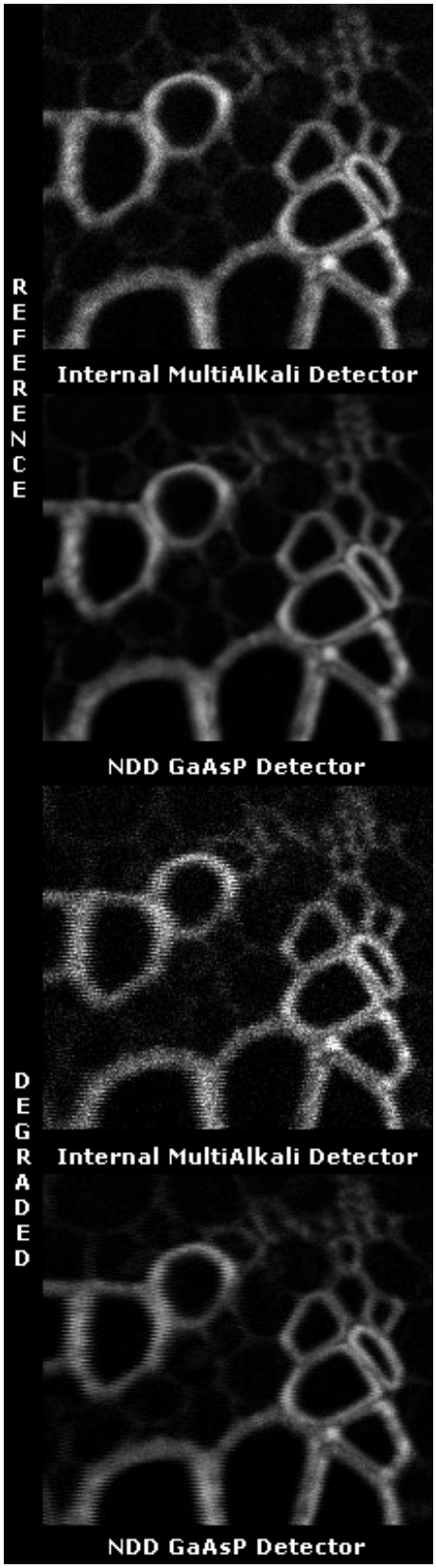


Figure 38. *LSM 510 Convallaria sample acquired with a typical misadjustment. Regular scanning method (top) and exaggerated bidirectional scanning artifact (bottom).*

Bidirectional

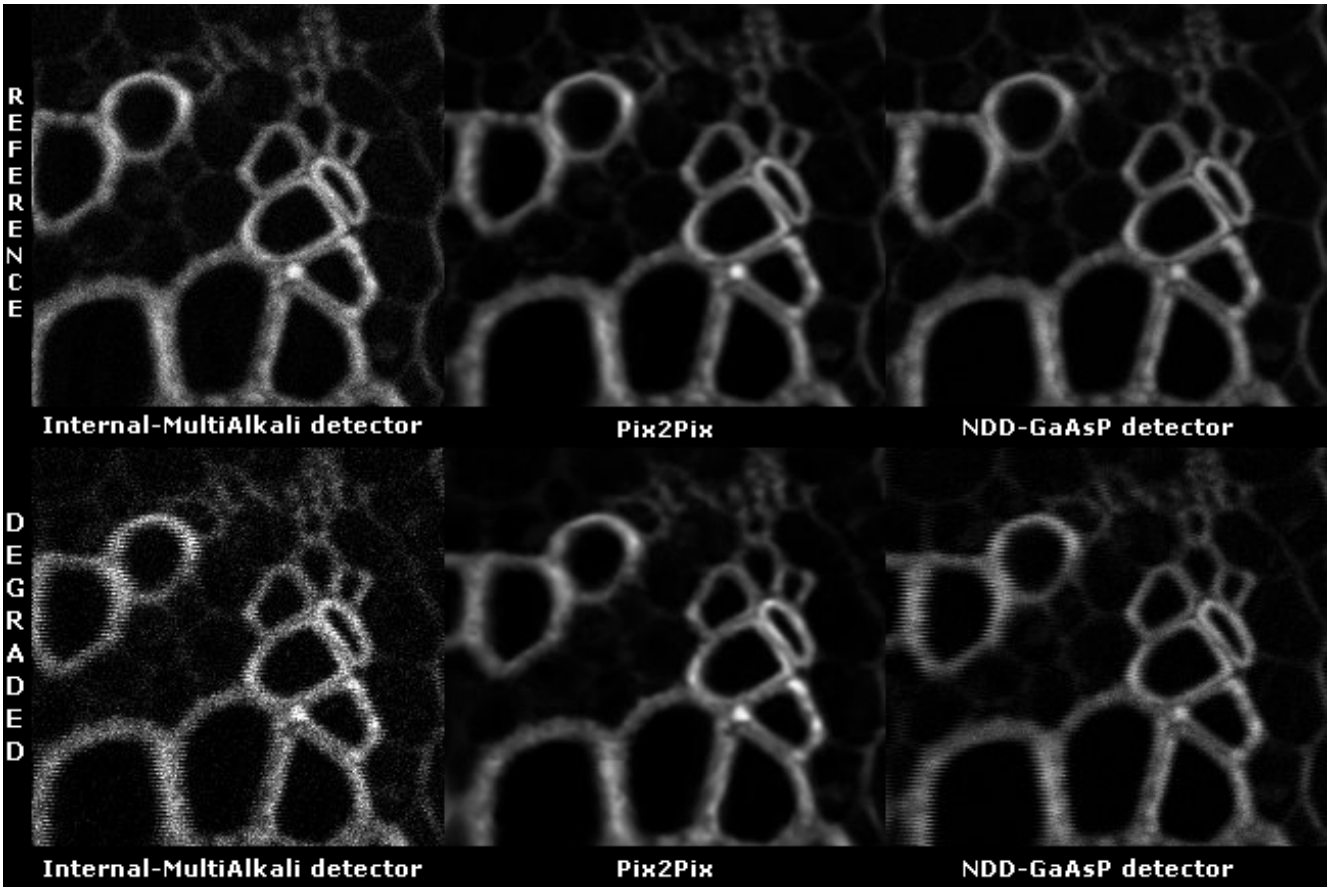


Figure 39. *Pix2Pix prediction in standard and bidirectional scanning.*

The prediction of the degraded image by the neural network shows little difference with the prediction of the reference image. They both imitate with little error the reference image of the NDD-GaAsP detector. That indicates a certain robustness of the Pix2Pix network and displays its ability to reconstruct images that are necessarily with the same nature as the objects it trained on (i.e. without defects).

2.2 Perspective

Working on this subject led me to learn and read about plenty of other methods and leads in how the project could either evolve or be adapted to similar topics.

2.2.1 Microscope Enhancing

Super Resolution An option we could imagine to make this project progress further is to get another kind of dataset along with a generative network. Instead of a dataset enhancing the content of an image, we could get a larger resolution for the ground truth and put in place a super-resolution network to transform low quality images into higher quality ones [Fig. 40], this however would mean acquiring plenty more of data and this on a high resolution system which implies longer acquisition time on a demanded system.

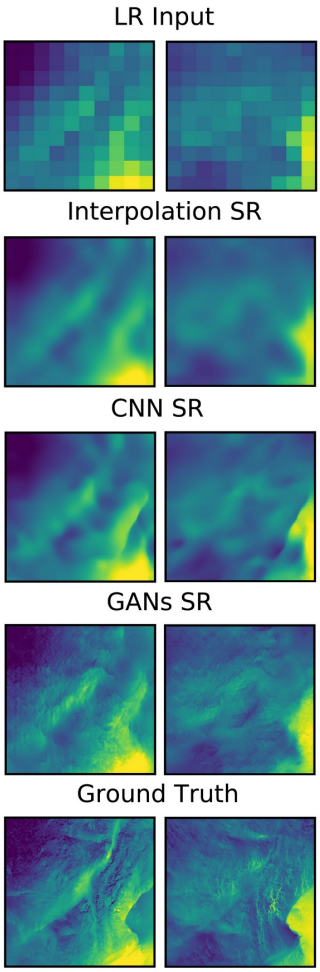


Figure 40. *An example of the possibilities with Super Resolution. Source: pnas.org*

Other instruments The LSM 780 and LSM 880 microscopes mentioned before had different detectors that could allow further tasks. As said before, the LSM 780 allow spectral imaging for a decomposition of the emission signal into its spectrum. The airyscan detector included on the LSM 880 configuration permits higher quality image. By constituting a dataset to enhance the quality of images that are detected on the same wavelength on the two systems, the quality of the LSM 780 microscope could be increased enough to have a spectrum decomposition of the emission signal with the quality of the airyscan imaging. This however implies spending a lot of time on both systems constituting the dataset as well as registration concerns to pair identical acquired areas in the case of supervised learning.

2.2.2 Deep-learning approaches

Another classifier for perceptual loss The VGG16 neural network is used for its ability to detect contents in images and classify them. Since it's been created, classification task has been done better by some other networks. We could imagine a perceptual loss using more modern classification models like inceptionV2 or NASNet. This last kind of network cited has a specifity, it has been build automatically through Neural Architecture Search, explaining its name.

Pushing the CycleGAN further As seen previously, the CycleGAN is a generative network that will pay attention to the nature of the image and produces results that are quite different from other neural network architectures. There is a major interest in using this network for its ability to perform semi-supervised learning. This implies the possibility to generate a dataset without the registration required in the case of supervised learning. Knowing that, the CycleGAN could be used on any acquisition of a sample done on two different instruments. It could achieve image transformations of the first instrument to the images of the second instrument without further indication. This allows an expansion of the current project to a much larger field.

Latent space reworking The first approach with latent space modeling using VAE is quite old and might not be good enough to compete with the generative networks. The latent space approach however might not be to exclude. It is possible to regularize the latent space of a GAN network like the Pix2Pix which is showing promising results. Doing latent space algebra on a more efficient network could allow the characterization of the noise and structures present on the dataset images and therefore the noise removal and the structure adaptation [Fig. 41].

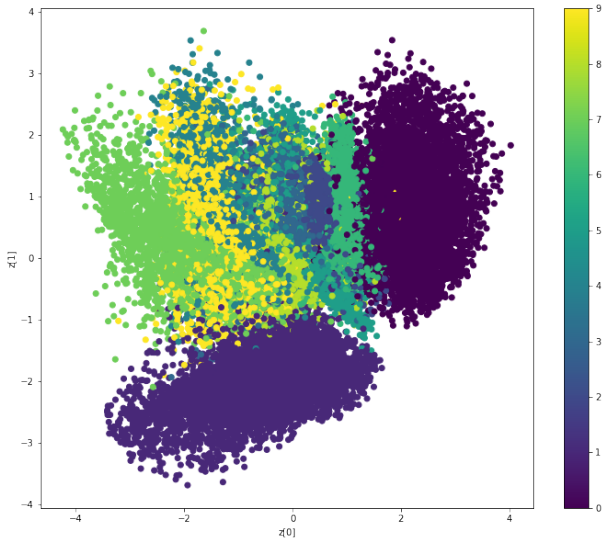


Figure 41. *The latent space organization.*

2.2.3 Dataset improvement

Fresh restart The data accumulated up to today might not be of the best quality especially the oldest ones. That is why I believe even with the current network, training it again being more careful on the data content (especially the ground truth used) could make a significant difference.

Pseudo Labeling I've been introduced to this practice that consists in injecting other image nature in the network, check how good it predicted it and if it did well add it to the training dataset. That is a possible enlargement of the dataset and network abilities to predict image natures right. For that reason, it is also an augmentation method. I've also been warned that this is not a guaranteed success and could instead waste time and energy if the nature is too far from what's already known by the neural network [Fig. 42].

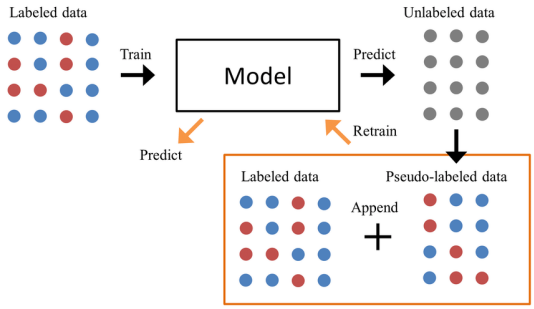


Figure 42. *Use of pseudo-labeling for dataset enlargement. Source : Research Gate*

2.2.4 Metric explorations

Deep taylor visual interpretability After the gradient module calculation and the metrics settling to tell whether an image is good looking or not, I've done some researches and came across Visual Interpretability methods. It seems to be a data processing that let see what is important in the image. Among them, Deep Taylor seem to have impressive results. This could be used to compare the areas marked as interesting by the procedure both in the predicted image and ground truth image [Fig. 43].

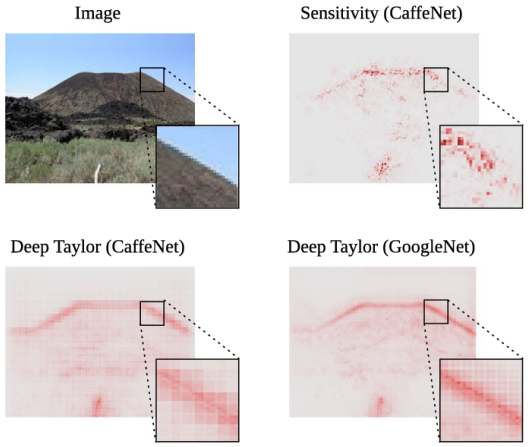


Figure 43. *Heatmaps of Networks place of interest. Source : Semantic Scholar*

GLCM measures Another metric telling about the homogeneity of the image could be used to assess the image quality. The gray-level co-occurence matrix (GLCM) allows an evaluation of the texture of the objects in the image. Comparing the homogeneity of the NDD-GaAsP image and the predicted images would be an additional indication to the networks efficiency.

Acknowledgments

Even though I am the main actor of this internship, it has only been possible thanks to many other people and entities.
Among them, there first is Aix-Marseille University hosting the Signal and Image processing master's degree (Master TSI). Thanks to Mr. Bellemare responsible of the master, all the teachers belonging to the teaching team and my classmates, the master's degree has been really enjoyable while full of instruction.
I am also thankful to GDR-ImaBio and IBDM for co-funding the internship thus making it possible for it to take place within the CNRS.
Last but not the least, I am grateful to all people in the IBDM I spent time with could it be for a lunch or for any project-related activity. I want to thank them for their kindness and the amount of their knowledge they gave me access to. In particular, Cedric Matthews has been my main tutor, guided the whole project and took some of his precious time to guide me on the direction to take and for a few microscope sessions. My second tutor, Fabrice Daian, taught me most practices and knowledge regarding Deep-Learning and guided me through the different approaches about the programming in this project. My third tutor, Daniel Sapede, conducted me through the usage of confocal microscope now permitting me to acquire data by myself whenever I need. For their help, guidance and availability, I could never thank enough my three tutors.

References

- Jablonski diagram:
  https://www.shsu.edu/7Echm_tgc/sounds/HTML5/Jablonski.HTML5.html
- Mitochondria dataset:
  https://www.epfl.ch/labs/cvlab/data/data-em/
- List of optimizers for deep learning:
  https://ruder.io/optimizing-gradient-descent/index.html#fn20
- Adam convergence discussion:
  https://arxiv.org/pdf/1904.09237.pdf
- Method for Adam correction:
  https://openreview.net/pdf?id=Bkg6RiCqY7
- CycleGAN presentation:
  https://datalchemy.net/blog/cyclegan
- CycleGAN implementation:
  https://machinelearningmastery.com/cyclegan-tutorial-with-keras/
- Pix2Pix Article:
  https://arxiv.org/abs/1611.07004
- Segmentation Decathlon Article:
  https://arxiv.org/abs/2106.05735
- Explanation and paid solution for plenty of networks:
  https://developers.arcgis.com/python/
- Usages of Pix2Pix:
  https://phillipi.github.io/pix2pix/
- Segmentation-Models' author's github:
  https://github.com/qubvel
- Latent space regularization for Pix2Pix:
  https://machinelearningmastery.com/how-to-interpolate-and-perform-vector-arithmetic-with-faces-using-a-generative-adversarial-network/
- Article for generative networks:
  https://www.researchgate.net/publication/333259964