Automated Segmentation of Cell Images Using Cycle-Consistent Generative Adversarial Networks **CycleGAN**

Djahnine Aissam[1]

[1]Student Master || IMA - Sorbonne University/Télécom Paris

***Abstract*—Within the last few years, we have witnessed a strong increase in deep learning applications for medical images analysis that still remain challenging such as : *denoising, reconstruction, data simulation, detection, classification and segmentation*. Generative Adversarial Networks (GANs), with their fascinating ability to synthesize new samples at a *unprecedented level of realism* offers new ways to face famous medical image analysis challenges.**

In this report, we are going to give an overview on Generative adversarial networks and their different medical-based approaches.

***Index Terms*—Deep learning, Unsupervised learning, Generative Adversarial Networks, Cycle-Consistent Generative Adversarial Networks, Convolutional Neural Networks biomedical imaging.**

I. Objective of the work

The main objective of this research is twofold :

- Generate synthetic cell images that model the distribution of the input images for data augmentation. Use both of the synthetic and real cells images for training a context-aware CNN that can accurately segment these cells.
- We propose to employ a segmentation method based on cycle-consistent generative adversarial networks (Cycle-GANs) that can be trained even in absence of prepared image-mask pairs.

II. Introduction

The emergence of deep learning CNN Based approaches, as a powerful tool for segmentation has revolutionised the medical field (among others) and the secrets behind their success are : the *quality* and the *volume* of the available data.However,the availability of labeled-data (or paired image-segmentation data for our purpose) is a real obstacle in many medical field applications.Therefore, *Goodfellow et al* [1] introduced in 2014 a new *Unsupervised learning approach* that learns abstract representations of the given data called **Generative adversarial networks** *(GANs for short)*. This novel framework consists of a two player min-max game where the first player (called the generator) is tasked with transforming a random input to a specific distribution such that the second player (called the discriminator) can't distinguish between the true and *"fake"* distributions [2].

Generative Adversarial Networks are composed of two models Figure 6:

- The first model is called a Generator and it aims to generate new data similar to the expected one. The Generator could be assimilated to a human art forger, which creates fake works of art.
- The second model is named the Discriminator. This model's goal is to recognize if an input data is *real* i.e it belongs to the original dataset or if it is 'fake' — generated by a forger. In this scenario, a Discriminator is analogous to the police (or an art expert), which tries to detect artworks as truthful or fraud.

Figure 6 shows The generative adversarial network global architecture:

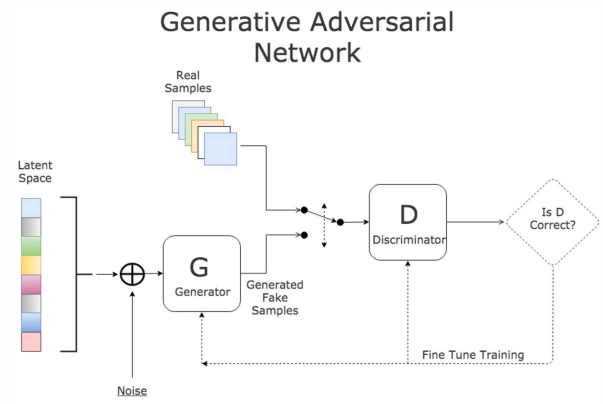


Figure 1: GAN global architecture - Source [1]

In the recent years, GANs models showed a *Fascinating* ability to generate realistic data. However, medical tasks are very sensitive to errors, and the generation of new synthetic data might be *critical* to constrain under these conditions. Still, Other approaches were proposed to overcome this limitation such as : *cGAN, Cycle-GAN* that we will be covering in future sections.

A central problem in biomedical imaging is the preparation of images for further quantitative analysis via automated image

segmentation. Recently, fully convolutional neural networks, such as the U-Net were applied successfully in a variety segmentation tasks. A downside of this approach is the requirement for a large amount of well-prepared training samples, consisting of image-ground truth mask pairs. Since these have to be prepared for each experiment by hand this task can be very costly and time-consuming. We present in future section a segmentation method based on cycle consistent generative adversarial networks, which can be trained even in absence of prepared image-mask pairs. We show that it successfully performs image segmentation tasks on two dataset chosen for this work.

III. Cell Images Generation

In this section, we first go through essential steps to generate synthetic data. We explain the mathematical framework using the notation denoted below, Neural network architecture, dataset used and some useful techniques that we are going to follow for our Beta Implementation.

IV. Notations

- We denote z as : input noise vector(latent space), x : input data(image), G : Generator, D : Discriminator, p_{data} : Real images distribution and p_z : Fake images distribution.

V. Mathematically Modeling a GAN

Though the **GANs** framework could be applied to any two models that perform the tasks described above, it is easier to understand when using universal approximators such as artificial neural networks. [1]

A neural network G(z, θ_1) is used to model the Generator mentioned above Figure 6. Its role is mapping input noise (latent) variables z to the desired data space x (say images). In the other hand, a second neural network D(x, θ_2) models the discriminator and outputs the probability that the data came from the real samples. In both cases, θ_i represents the weights or parameters that define each neural network.
As a result, the Discriminator is trained to correctly classify the input data as either real or fake. This means its weights are updated as to maximize the probability that any real data input x is classified as belonging to the *real* dataset, while minimizing the probability that any fake image is classified as belonging to the *real* dataset. In more technical terms, the loss/error function used maximizes the function $D(x)$, and it also minimizes $D(G(z))$ (as mentioned earlier , a min-max game).

Furthermore, the Generator is trained to fool the Discriminator by generating data as realistic as possible, which means that the Generator's weight's are optimized to maximize the probability that any fake image is classified as belonging to the real dataset. Formally this means that the loss/error function used for this network maximizes $D(G(z))$. *In practice*, the logarithm of the probability (e.g. log D(...)) is used in the loss functions instead of the raw probabilities, since using a log loss heavily penalises classifiers that are confident about an incorrect classification.

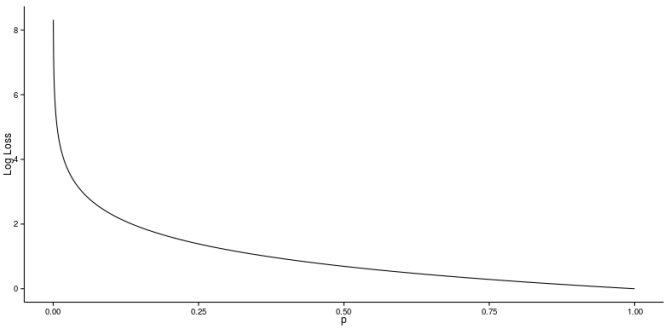


Figure 2: Log Loss Visualization

We notice in Figure 2 that Low probability values are highly penalized.

GAN Models are trained by solving the following optimisation problem that the discriminator is trying to maximize and the generator is trying to minimize.

$$\min_G \max_D V(D,G) = \mathbb{E}_{x\sim p_{data}(x)}[logD(x)] + \mathbb{E}_{z\sim p_z(z)}[log(1 - D(G(z)]$$

- Where G is the generator, D is the discriminator , $V(D,G)$ is the objective function (See subsection A) , p_{data} is the distribution of real samples and p_z is a distribution from which noise vectors are drawn (Spherical Gaussian for example).
The final layer of the discriminator network contains a *sigmoid* activation function so that D(x),D(G(z))$\in [0,1]$.
By maximizing the value function, the discriminator minimizes the error of its predictions with respect to target values 1 and 0 for real and fake sample respectively. Conversely, the generator tries to minimize the chance that the discriminator will predict a 0 for fake samples. The Loss of the generator depends directly on the performance of the discriminator [1].

- We define below two main concepts for the global GANs architecture :

A. Objective function

As mentioned in the original paper of $Goodfellow$ [1], the equilibrium point (Nash equilibrium) in training the GAN is unlikely reached i.e the training process is unstable. At the beginning of the training process the discriminator can easily distinguish p_{fake} samples among the p_{data}. The gradient for $log(1 - D(G(z))$ is close to zero, in consequence , the generator will fail to update its parameters and minimises its loss. Therefore,in practice (Used for the Beta-implementation) we use a different loss function for the generator :

$$\min_G V(D,G) = -\mathbb{E}_{z\sim p_z(z)}[logD(G(z))]$$

In the equation above, the objective for the generator is to **minimize** the probability that the discriminator identifies generated sample as fake.In figure (4) the objective is to **maximize** the probability that the discriminator identifies generated samples as real [3].

B. Latent Space

The generator G tries during the training process to map point from a low dimensional latent space (Noise vector z) to the high-dimensional sample space [3].

VI. Related Work

A. Applications in biomedical image analysis

Over years ,GANs were used to solve a list of problems in biomedical image analysis.We list below some of the main challenges and possible applications for these generative models [3].

- Sample Generation :
  - From a latent distribution, GANs can generate ***"new"*** samples of images data (medical images for purpose) that can potentially used ***to enlarge training sets for other discriminative models***.this task isn't easy at it seems it is because of the errors that can occur during the generation process that can lead to negative effects when dealing with medical images *(For example : Cancer prognosis and prediction)*
  - An Example of synthetic retinal images is shown in Figure 3.

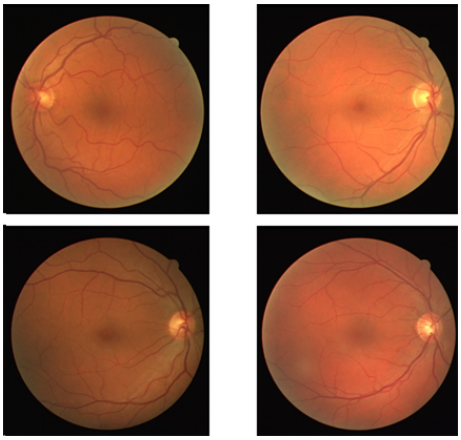


Figure 3: Example of synthetic retinal images.From left to right Real retinal image,the synthetic retinal image, Source[2]

  - You find In Section (6) a beta-implementation of generation 64 x 64 images of RPE human cells.

- **Image segmentation**:
  - The segmentation is considered as one of the important topics in image processing domain especially in medical imaging.CNNs architectures offers an effective tool for this task. Still, they don't produce perfect segmentation (holes and fragments) because of the use of a*voxel-wise unstructured loss* such as *the cross-entropy loss*. [4]
  - Many solutions were found to overcome this issue, correct segmentations can be imposed by an adversarial network model that evaluate and decide whether a segmentation, or a combination of segmentation and input image is possible. [2]
  - The second part of this report will treat different segmentation methods using : Cycle-GAN.
- **Image synthesis**:
  - We often deal with multiple medical images that might contain shared information between them.In order to lower the number of this redundant-data an accurate conversion can be useful to convert an imaging domain to another.

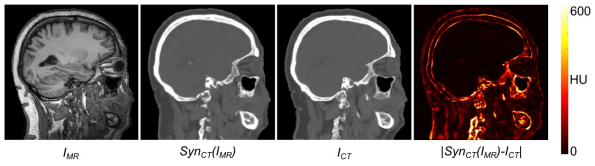


Figure 4: Example of converting a MR image into CT images.From left to right input MR image,the synthetic CT image , the reference CT image and the difference between the synthetic and reference CT image [2]

- **Image quality enhancement**:
  - The acquisition of medical images has its own constraints when it comes to Image-quality.It's often quite delicate to have good imaging results with reducing time,cost and patient discomfort.An example was cited in [2] where lower ionizing radiation dose levels in CT led to increased image noise levels,another example with MR image : reconstructing under-sampled image that could reduce scan time but lead to image artifacts.

VII. Segmentation Methods Using GANs

The second part of this report consist of segmenting Cell Images Using the GAN models that we presented in the previous sections. We tend to try one of these two approaches of segmentation methods described here [5], [6].
In [5] the main task consists of generating the samples using the GAN Model then train a U-net CNN-based segmentation model.The available data are passed through a GAN. The discriminator learns an unsupervised representation of the data.the weights of the of the discriminator will be transferred to the U-net encoder.

While in [6] the paper propose a simple unpaired mapping-based approach where cycle-consistency loss is used in a dual-GAN architecture to transfer between Masks and desired input images (the use of a cycle-GAN).

VIII. Beta-Implementation

In This Section , We are going to evaluate a beta-implementation of a GAN model that we build based on the architecture in [7] to attempt generating synthetic data.

A. Neural Network Architecture

Historical attempts to scale up GANs using CNNs to model images have been unsuccessful (Generate high-resolution images). After extensive model exploration *Goodfellow* in his original paper [1] identified a family of architectures that results in stable training across a range of datasets[3] and allowed for training higher resolution and deeper generative models.*DCGAN* [4] is one of the popular and successful network design for GAN. It mainly composes of convolution layers without max pooling or fully connected layers.The architecture guidelines for stable Deep Convolutional GANs found in [7] :

- Replace any pooling layers with *strided convolutions* (discriminator) and fractional-strided convolutions (generator).
- Use batchNorm in both the generator and the discriminator.
- Remove fully connected hidden layers for deeper architecture.
- Use ReLU activation in generator for all layers except for the output, which uses Tanh.
- Use LeakyReLU activation in the discriminator for all layers.

B. Dataset

1) Human Retinal Pigment Epithelium (RPE) cells: The *Dataset* used here consisted of 1032 256x256 absorbance microscopy images of human Retinal Pigment Epithelium (RPE) cell images [5].You can browse and download the data using the URL[5] below.

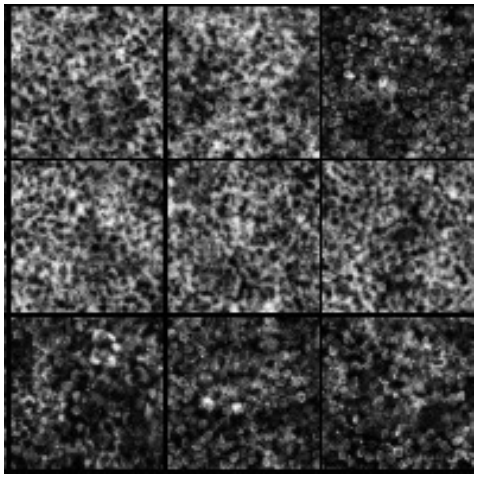


Figure 5: Training Data Example

C. Training details

The training code was implemented using *Pytorch*. Since both the generator and discriminator are being modeled with neural networks, a gradient-based optimization algorithm(Adam Solver) was used to train the GAN.We used tuning tips from the original paper [7] :

> " All models were trained with mini-batch stochastic gradient descent (SGD) with a mini-batch size of 128. All weights were initialized from a zero-centered Normal distribution with standard deviation 0.02. In the LeakyReLU, the slope of the leak was set to 0.2 in all models. While previous GAN work [1] has used momentum to accelerate training, we used the Adam optimizer with tuned *hyperparameters*. We found the suggested learning rate of 0.001, to be too high, using 0.0002 instead. Additionally, we found leaving the momentum term 1 at the suggested value of 0.9 resulted in training oscillation and instability while reducing it to 0.5 helped stabilize training".

We Trained the model for 300 epochs using also some useful *GANHACKs* tricks proven to be useful for training GANs, found here [6]

Algorithm 1 Minibatch stochastic gradient descent training of generative adversarial nets. The number of steps to apply to the discriminator, k, is a hyperparameter. We used $k = 1$, the least expensive option, in our experiments.

for number of training iterations **do**

for k steps **do**

- Sample minibatch of m noise samples $\{\boldsymbol{z}^{(1)}, \ldots, \boldsymbol{z}^{(m)}\}$ from noise prior $p_g(\boldsymbol{z})$.
- Sample minibatch of m examples $\{\boldsymbol{x}^{(1)}, \ldots, \boldsymbol{x}^{(m)}\}$ from data generating distribution $p_{\text{data}}(\boldsymbol{x})$.
- Update the discriminator by ascending its stochastic gradient:

$$\nabla_{\theta_d} \frac{1}{m} \sum_{i=1}^{m} \left[\log D\left(\boldsymbol{x}^{(i)}\right) + \log\left(1 - D\left(G\left(\boldsymbol{z}^{(i)}\right)\right)\right)\right].$$

end for

- Sample minibatch of m noise samples $\{\boldsymbol{z}^{(1)}, \ldots, \boldsymbol{z}^{(m)}\}$ from noise prior $p_g(\boldsymbol{z})$.
- Update the generator by descending its stochastic gradient:

$$\nabla_{\theta_g} \frac{1}{m} \sum_{i=1}^{m} \log\left(1 - D\left(G\left(\boldsymbol{z}^{(i)}\right)\right)\right).$$

end for

The gradient-based updates can use any standard gradient-based learning rule. We used momentum in our experiments.

Figure 6: GAN Training algorithm

IX. Results

Here in Table I, You find some results of the implemented DCGAN model.

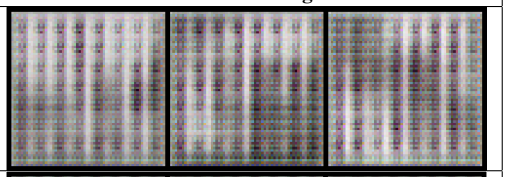
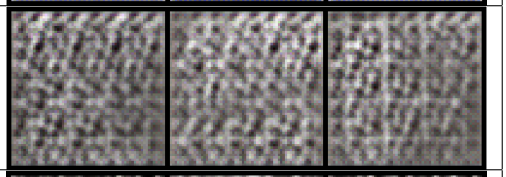
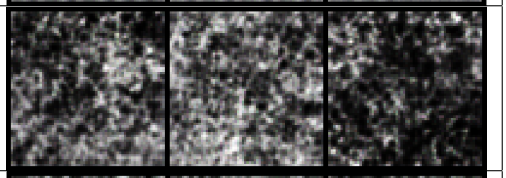
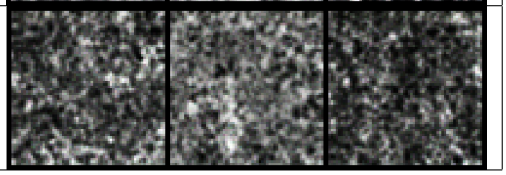
[3]MNIST,Fashion-MNIST, Cifar10

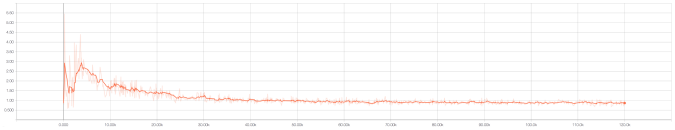
[4]Deep Conventional Generative adversarial Network

[5]https://isg.nist.gov/deepzoomweb/data/RPEimplants

[6]https://github.com/soumith/ganhacks

Table I: Implementation results

Epochs	Generated Images
Epoch 10	
Epoch 50	
Epoch 200	
Epoch 300	



(a) Generator loss



(b) Discriminator loss

Figure 7: Traditional GAN loss

A. *Comments*

- We can clearly notice In the beginning of the training process that the generated images are pure noise.
- As we can see in the Figure 7, the discriminator error is very high in the beginning, as it does not know how to classify correctly images as being either real or fake. As the discriminator becomes better and its error decreases to about 0.5 at step 5k, the generator error increases, proving that the discriminator outperforms the generator and it can correctly classify the fake samples. As time passes and training continues, the generator error lowers, implying that the images it generates are better and better. While the generator improves, the discriminator's error increases, because the synthetic images are becoming more realistic each time.
- The results got better within the epoch but still noisy(due to the low image resolution 128 x 128. They miss the structural details of the input data and since the output images size is set to 64 x 64, that cost us a loss in term of resolution.
- The lack of *enough* training data had an impact on the results. The model was trained on 1032 256 x 256 images which is not enough to train such hard-training models.
- We can modify the model in order to generate 128 x 128 output images but it costly from computation resource point of view.

X. Conclusion

- In machine learning, the generative models try to generate data from a given (complex) probability distribution.
- GANs are hard to train, and they're often evaluated qualitatively rather than the use of usual metrics in deep learning field.
- We can say that the idea of adversarial training suggested by Ian Goodfellow and its co-authors is really a great one. This way to twist the loss function to go from a direct comparison to an indirect one is really something that can be very inspiring for further works in the deep learning area.

XI. CycleGAN

A. Overview

CycleGAN was introduced in the now well-known 2017 paper out of Berkeley, Unpaired Image-to-Image Translation using Cycle-Consistent Adversarial Networks. It was interesting because it did not require paired training data — while an x and y set of images are still required, they do not need to directly correspond to each other.
Since paired data is harder to find in most domains, and not even possible in some, the unsupervised training capabilities of CycleGAN are quite useful.

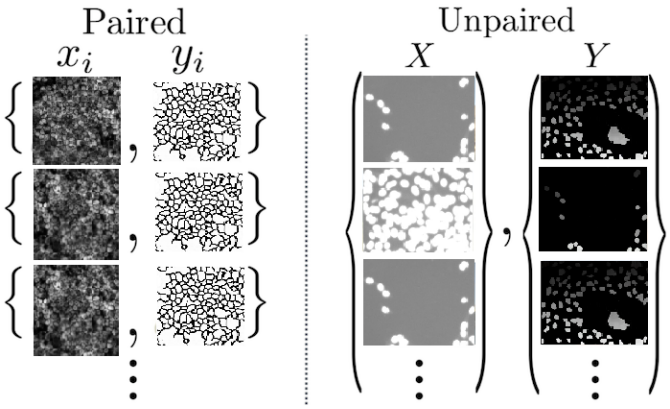


Figure 8: Paired Vs Unpaired approaches [8]

- We propose to employ a segmentation method based on cycle-consistent generative adversarial networks (CycleGANs) that can be trained even in absence of prepared image-mask pairs. We show that this model performs competitively on standard segmentation tasks even when trained on just a few target samples.

B. How does it work?

Image-to-image translation is a class of vision and graphics problems where the goal is to learn the mapping between an input image and an output image using a training set of aligned image pairs. However, for many tasks, paired training data will not be available. We present an approach for learning to translate an image from a source domain X to a target domain Y in the absence of paired examples. Our goal is to learn a mapping $G : X ß Y$ such that the distribution of images from $G(X)$ is indistinguishable from the distribution Y using an adversarial loss. [8] Because this mapping is highly under-constrained, we couple it with an inverse mapping $F : Y ß X$ and introduce a cycle consistency loss to enforce $F(G(X))X$ (and vice versa).

C. Problem formulation

1) ***A Tale of Two Generators****:* Our goal is to learn mapping functions between two domains X and Y given training samples $(x_i)_{i=1}^N$ where $x_i \in X$ and $(y_j)_{j=1}^M$ where $y_j \in Y$. We denote the data distribution as x ~$p_{data}(x)$ and y ~$p_{data}(y)$. Our model includes two mappings $G : X ß Y$ and $F : Y ß X$. In addition, we introduce two adversarial discriminators D_X and D_Y , where D_X aims to distinguish between images x and translated images $F(y)$; in the same way, D_Y aims to discriminate between y and $G(x)$. Our objective contains two types of terms: adversarial losses for matching the distribution of generated images to the data distribution in the target domain; and cycle consistency losses to prevent the learned mappings G and F from contradicting each other. [8]

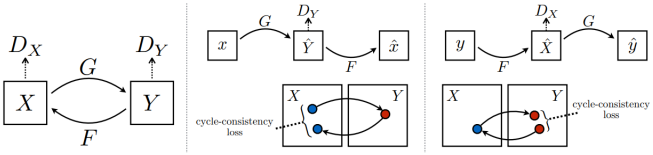


Figure 9: Cycle consistency constraint

D. The Objective Function

1) ***Adversarial Loss****:* We apply adversarial losses to both mapping functions. For the mapping function $G : X ß Y$ and its discriminator D_Y , we express the objective as:

$$\mathrm{L}_{GAN}(G, D_Y, X, Y) = \mathbb{E}_{y \sim pdata(y)}[logD_Y(y)] + \mathbb{E}_{x \sim pdata(x)}[log(1 - \ D_Y(G(x))]$$

Where G tries to generate images $G(x)$ that look similar to images from domain Y , while D_Y aims to distinguish between translated samples $G(x)$ and real samples y. G aims to minimize this objective against an adversary D that tries to maximize it, i.e, $min_G max_{D_Y} L_{GAN}(G, DY, X, Y)$ We introduce a similar adversarial loss for the mapping function F : Y → X and its discriminator $\mathrm{D}_X aswell$: *i.e.*, $\mathrm{min}_F max_{D_X} L_{GAN}(F, D_X, Y, X)$.

2) ***Cycle Consistency Loss****:* Adversarial training can, in theory, learn mappings G and F that produce outputs identically distributed as target domains Y and X respectively. However, with large enough capacity, a network can map the same set of input images to any random permutation of images in the target domain, where any of the learned mappings can induce an output distribution that matches the target distribution. Thus, adversarial losses alone cannot guarantee that the learned function can map an individual input x_i to a desired output y_i [9]. To further reduce the space of possible mapping functions, we argue that the learned mapping functions should be cycle-consistent: as shown in Figure 9, for each image x from domain X, the image translation cycle should be able to bring x back to the original image,i.e., $x \rightarrow G(x) \rightarrow F(G(x)) \approx$x. We call this forward cycle consistency. Similarly, as illustrated in Figure 9, for each image y from domain Y , G and F should also satisfy backward cycle consistency: $y \rightarrow F(y) \rightarrow G(F(y)) \approx$y. To respect this behavior we a cycle consistency loss:

$$L_{cycle}(G, F) = \mathbb{E}_{x \sim pdata(x)} [|| F(G(x)) - x ||_1] + \mathbb{E}_{y \sim pdata(y)} [|| G(F(y)) - y ||_1]$$

3) Full objective: The full objective is :

$$L(G, F, D_X, D_Y) = L_{GAN}(G, D_Y, X, Y) + L_{GAN}(F, D_X, Y, X) + \lambda L_{cycle}(G, F)$$

where λ controls the relative importance of the two objectives. We aim to solve:

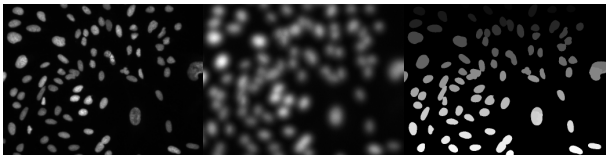
$$G^*, F^* = \underset{G,F}{\operatorname{argmin}} \max_{DY, DY} L(G, F, D_X, D_Y)$$

XII. Dataset

For experiments, we used two public datasets: **Human U2OS cells** and **Human Retinal Pigment Epithelium (RPE) cells**

1) Human U2OS cells:

- This benchmark is available on the official website of Broad institute (Broad Bio-image Benchmark Collection)[Link to browse and download] [7]. It contains more than 55 k images. Unfortunately only 768 image-mask pairs were available for training and testing. This image set provides examples of in- and out-of-focus HCS [8] images.



(a) In-Focus (b) Out-Focus (c) Mask

Figure 10: example of in- and out-of-focus HCS images

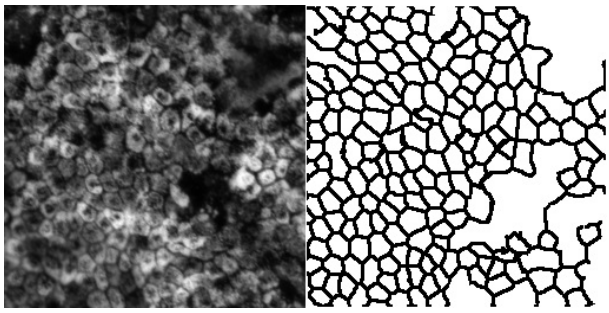
- Images were acquired from one 384-well microplate containing U2OS cells. The optimal focus was found using laser auto-focusing to find the well bottom. The automated microscope was then programmed to collect a z-stack of 32 image sets (z = 16 at the optimal focal plane, 15 images above the focal plane, 16 below) with 2 m between slices. Each image is 696 x 520 pixels in 16-bit TIF format. Each image filename includes either 'w1' to denote Hoechst images or 'w2' to denote phalloidin images.
- The constraints about this dataset is that the images were taken with different focal planes. The segmentation task still however an easy task for this kind of images. The goal here is to use a generative-based method.
- The images were manually examined by an expert to classify which focal planes corresponded to in- and out-focus images. For in-focus images, the pixels belonging to the nuclei were automatically identified as foreground and segmented

[7]https://data.broadinstitute.org/bbbc/BBBC006/

[8]High-content screening

A. Human Retinal Pigment Epithelium (RPE) cells

The *Dataset* consisted of 1032 256x256 absorbance microscopy images of human Retinal Pigment Epithelium (RPE) cell images [5].You can browse and download the data using the URL[9] below.



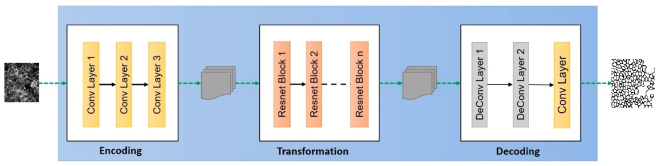
(a) Image-mask pair

Figure 11: Training Data Example -RPE cells-

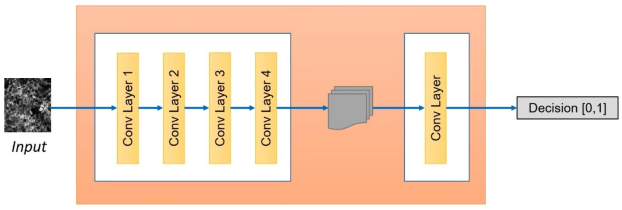
XIII. Implementation

*1) **Network Architecture**:* Each *CycleGAN* generator has three sections: an encoder, a transformer, and a decoder. The input image is fed directly into the encoder, which shrinks the representation size while increasing the number of channels. The encoder is composed of a set of convolution layers. The resulting activation is then passed to the transformer, a series of residual blocks. It is then expanded again by the decoder, which uses several transpose convolutions to enlarge the representation size, and one output layer to produce the final image. For the discriminator networks we use 70 × 70 *PatchGANs*, which aim to classify whether 70 × 70 overlapping image patches are real or fake. Such a patch-level discriminator architecture has fewer parameters than a full-image discriminator and can work on arbitrarily-sized images in a fully convolutional fashion. Figure 12 summarises the network architecture.

[9]https://isg.nist.gov/deepzoomweb/data/RPEimplants



(a) Generator



(b) Discriminator

Figure 12: Network Architecture

*2) **Training details**:* The training approach was fairly typical for an image-to-image translation task. The Adam optimizer, a common variant of gradient descent, was used to make training more stable and efficient. The learning rate was set to 0.0002 for the first half of training, and then linearly reduced to zero over the remaining iterations. The batch size was set to 1, which is why we refer to instance normalization, rather than batch normalization.

*3) **Reducing Model Oscillation**:* To prevent the model from changing drastically from iteration to iteration, the discriminators were fed a history of generated images, rather than just the ones produced by the latest versions of the generators. To do this, we keep a pool to store the 50 most recently generated images. This technique of reducing model oscillation was pioneered by Shrivastava et al.

XIV. Experiments

We decided to go through the Unpaired data for RPE cells dataset while working with Paired approach for U2OS. We present in the next section some evaluation metrics.Finally, we show experimental results, with discussion and suggested solution to improve the performance of the global approach.

XV. Evaluation Metrics

The performance of automatic segmentation methods is usually evaluated by computing some common objective criteria. We use for our application two of these such as the **Dice similarity coefficient (DSC)** and **Hausdorff Distance (HD)**.

*1) **Hausdorff Distance (HD)**:* HD(X,Y) is the longest distance one has to travel from a point in one of the two sets to its closest point in the other set. In image segmentation, HD is computed between boundaries of the estimated and ground-truth segmentations, which consist of curves in 2D and surfaces in 3D.Among all criteria, HD is one of the most informative and useful criteria because it is an indicator of the largest segmentation error. For two point sets X and Y , the one-sided HD from X to Y is defined as :

$$hd(X,Y) = \max_{x \in X} \min_{y \in Y} \| \text{x-y} \|_2$$

and similarly for hd(Y, X):

$$hd(Y,X) = \max_{y \in Y} \min_{x \in X} \| \text{x-y} \|_2$$

The bidirectional HD between these two sets is then:

$$HD(X,Y) = max(hd(X,Y), hd(Y,X))$$

*2) **Dice similarity coefficient (DSC)**:* The Dice coefficient, also called the overlap index, is the most used metric in validating medical volume segmentations. In our application we use this metric for a direct comparison between automatic and ground truth segmentations. This coefficient is calculated using the formula defined by:

$$Dice(A,B) = \frac{2|A \cdot B|}{|A| + |B|} = \frac{2TP}{2TP + FP + FN}$$

Where TP: "True positive", FP: "False positive" and FN: "False negative".

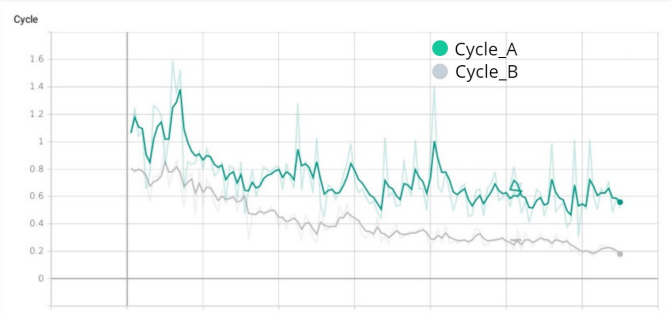
XVI. Results

In this section, we show results on our both paired datasets where ground truth input-mask pairs are available for evaluation. We then visualize and discuss the evolution of both the adversarial loss and the cycle consistency loss. Full results can be found in the Appendix A and B.
The results shown in the Appendix are visually compelling results in many cases (especially for the second dataset).In order to qualitatively evaluate our approach and using the metrics defined above, We compute the segmentation accuracy w.r.t to ground truth masks provided.

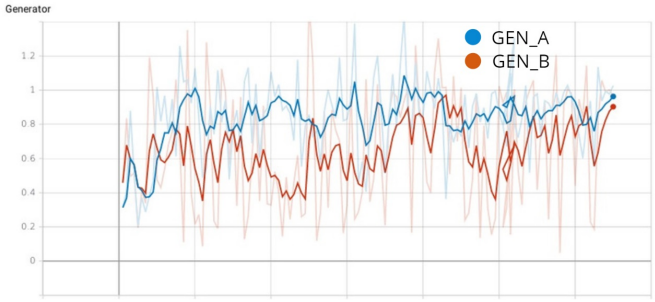
Table II: Evaluation Metrics

Dataset	HD	DSC
Human RPE	9.8871	0.7155
Human U2OS	3.1026	0.9316

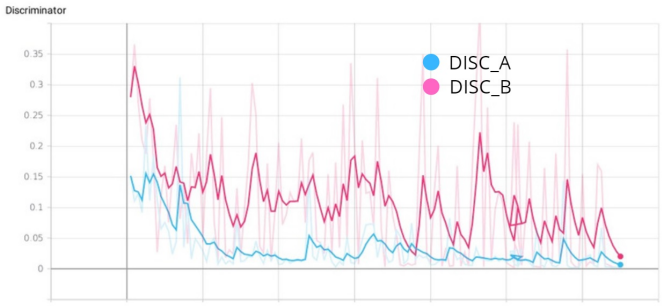
Table II shows the evaluation results on both test sets(20 images for RPE, 45 for U2OS), we computed the mean of DSC (respectively HD). U2OS shows better results, it's due to the simplicity of cells images to segment.



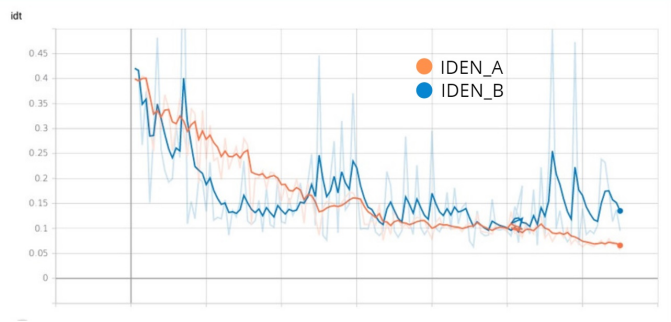
(a) Cycle-Consistency Loss



(b) Generator Loss



(c) Discriminator Loss

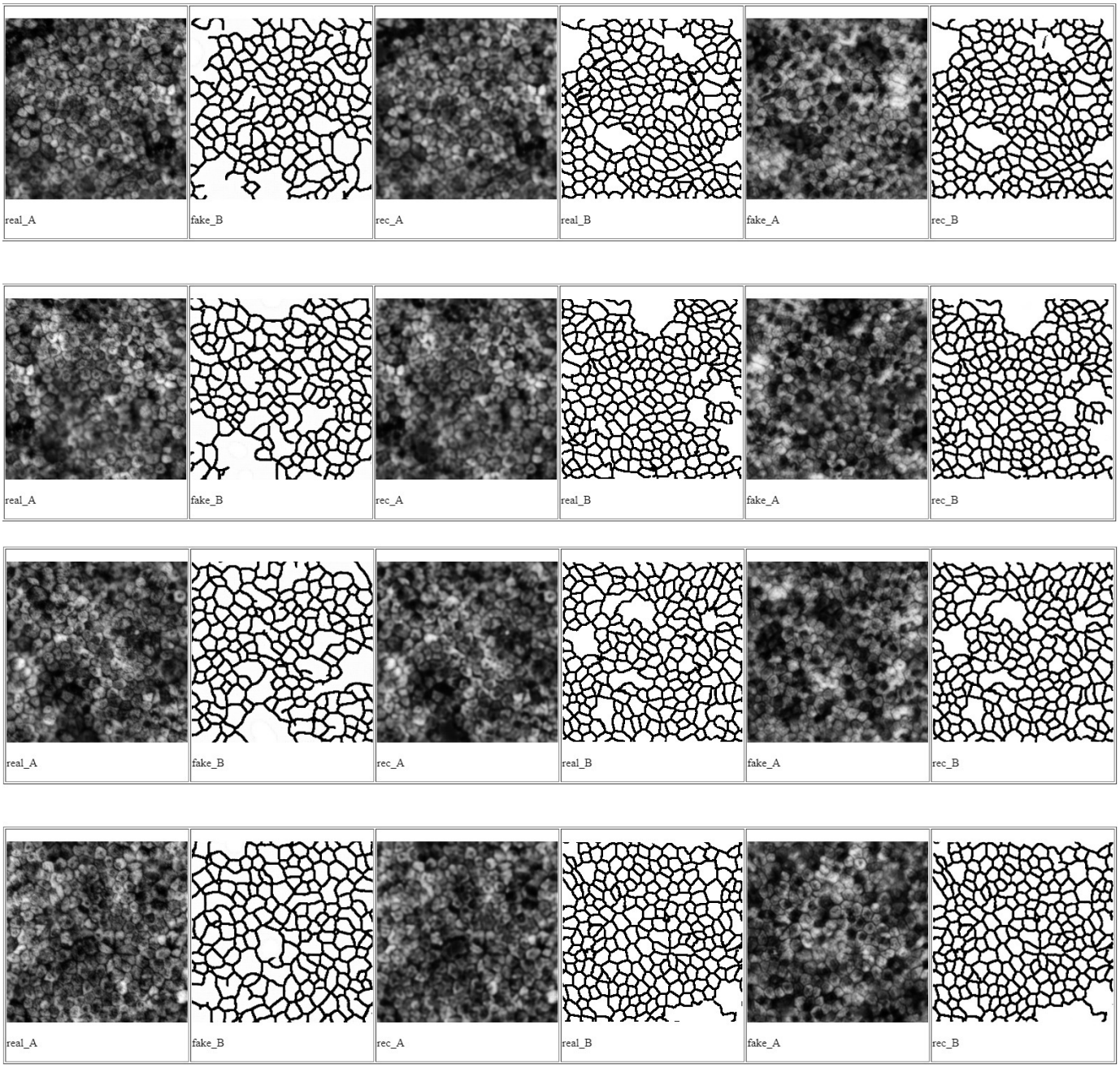


(d) Identity Loss

Figure 13: Loss curves

XVII. APPENDIX A

- We denote, A : " input images domain", B : "Segmentation masks domain", rec : "Identity mapping in both directions".
- $Real_B$, $Fake_B$ are : Ground truth and our result respectively.
- Figure 14 show the results on **Human Retinal Pigment Epithelium (RPE) cells** dataset.

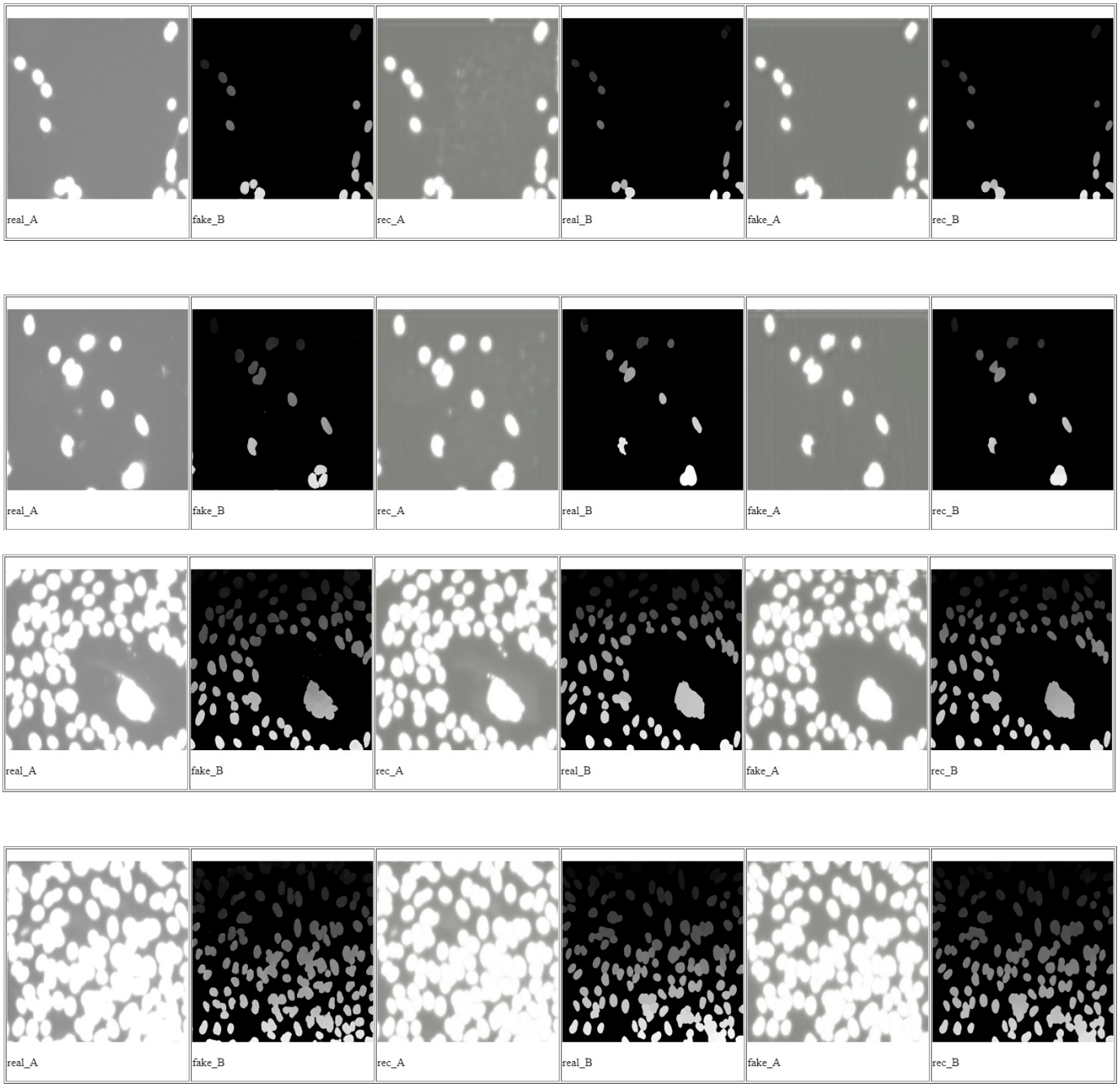


(b) "real A" : input image , "fake B" : CycleGan Segmentation, "real B" : Ground-Truth

Figure 14: Segmentation Results

XVIII. Appendix B

We denote, A : " input images domain", B : "Segmentation masks domain", rec : "Identity mapping in both directions". $Real_B$, $Fake_B$ are : Ground truth and our result respectively.
Figure 15 show the results on **Human U2OS cells** dataset.



(b) "real A" : input image , "fake B" : CycleGAN Segmentation, "real B" : Ground-Truth

Figure 15: Segmentation Results

XIX. Discussions

- Analogously with the report notation, the A represents the domain X,B domain Y. In Figure 13 we display 4 losses uses to train the cycleGAN model. $Cycle_A$ for cycle-consistency loss, $Gen_{A,B}$ for generators loss, $Disc_{A,B}$ for discriminators and $Iden_{A,B}$ for Identity loss.

- Figure 13(a) shows cycle-consistency loss, at the beginning of training the values for Generator A,B (analogue to X,Y) are between [1, 1.4]. they start to descend within 150 epochs (0.6 for $Cycle_A$, 0.2 for $Cycle_B$. This is a good sign although we have some oscillations around.

- From both Figure 13(b)(c), we see a noisy unstable loss, however, for discriminator it decreases within epochs approaching to zero it follow the cycle-consistency evolution while it increases for generator. From our experiences in GANs, we rarely get stable usual loss curves, we proceed to test and results to validates the outputs of the model.
- The "Identity" loss(Optional) that we added for training helped with training, it decreases to about 0.1 for both generators.
- For Human U2OS cells dataset, the task of segmentation was kind of easy, the cycle-consistency loss converges to 0 after less than 60 epochs(same case for identity).
- From Figure 14, the segmentation result is quite satisfying. However, we can clearly see some regions that were not well segmented. This is due to the complexity of the input images. Suggested alternatives is to use a larger dataset for training, and proceed with paired data approach rather unpaired one.
- Some failure cases are caused by the distribution characteristics of the training datasets(case of RPE dataset). The method may fail in cases that involve color and texture changes.

XX. Conclusion

- The main takeaways of this report are:

- GANs, may be one of the greatest ideas in the last 20 years since it provides interesting results in different tasks : generation, segmentation etc.
- Working with such architectures help us going beyond traditional deep learning approaches.
- We can treat real world problems where paired data is not available with Cycle-GANs that proved to be powerful for hard tasks such as segmentation.
- Training a GAN is not easy, and the EQUILIBRIUM point is rarely reached. However, different tricks, methods are used to improve results(depends on the datasets, the domain etc).
- Automated segmentation using Cycle-GAN competes Traditional approaches, with close performances.

XXI. Acknowledgments

We thank Mrs *Meas-Yedid Vannary* for her helpful comments, feedback and advice during the project.

References

[1] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative Adversarial Nets," in *Advances in Neural Information Processing Systems 27*, Z. Ghahramani, M. Welling, C. Cortes, N. D. Lawrence, and K. Q. Weinberger, Eds. Curran Associates, Inc., 2014, pp. 2672–2680. [Online]. Available: http://papers.nips.cc/paper/5423-generative-adversarial-nets.pdf

[2] S. Kazeminia, C. Baur, A. Kuijper, B. van Ginneken, N. Navab, S. Albarqouni, and A. Mukhopadhyay, "GANs for Medical Image Analysis," *arXiv:1809.06222 [cs, stat]*, Oct. 2019, arXiv: 1809.06222. [Online]. Available: http://arxiv.org/abs/1809.06222

[3] J. M. Wolterink, A. M. Dinkla, M. H. F. Savenije, P. R. Seevinck, C. A. T. v. d. Berg, and I. Isgum, "Deep MR to CT Synthesis using Unpaired Data," *arXiv:1708.01155 [cs]*, Aug. 2017, arXiv: 1708.01155. [Online]. Available: http://arxiv.org/abs/1708.01155

[4] M. Rezaei, H. Yang, and C. Meinel, "Conditional Generative Refinement Adversarial Networks for Unbalanced Medical Image Semantic Segmentation," *arXiv:1810.03871 [cs]*, Oct. 2018, arXiv: 1810.03871. [Online]. Available: http://arxiv.org/abs/1810.03871

[5] M. Majurski, P. Manescu, S. Padi, N. Schaub, N. Hotaling, C. S. Jr, and P. Bajcsy, "Cell Image Segmentation Using Generative Adversarial Networks, Transfer Learning, and Augmentations," p. 9.

[6] F. Mahmood, D. Borders, R. Chen, G. N. McKay, K. J. Salimian, A. Baras, and N. J. Durr, "Deep Adversarial Training for Multi-Organ Nuclei Segmentation in Histopathology Images," *arXiv:1810.00236 [cs]*, Oct. 2018, arXiv: 1810.00236. [Online]. Available: http://arxiv.org/abs/1810.00236

[7] A. Radford, L. Metz, and S. Chintala, "UNSUPERVISED REPRESENTATION LEARNING WITH DEEP CONVOLUTIONAL GENERATIVE ADVERSARIAL NETWORKS," p. 16, 2016.

[8] J.-Y. Zhu, T. Park, P. Isola, and A. A. Efros, "Unpaired Image-to-Image Translation using Cycle-Consistent Adversarial Networks," *arXiv:1703.10593 [cs]*, Nov. 2018, arXiv: 1703.10593. [Online]. Available: http://arxiv.org/abs/1703.10593

[9] M. Häring, J. Großhans, F. Wolf, and S. Eule, "Automated Segmentation of Epithelial Tissue Using Cycle-Consistent Generative Adversarial Networks," Bioinformatics, preprint, Apr. 2018. [Online]. Available: http://biorxiv.org/lookup/doi/10.1101/311373