# Generative Adversarial Networks : Report

Karl Hajjar, Kevin Riera

karl.hajjar@polytechnique.edu, kevin.riera@polytechnique.edu

## 1 Introduction

The aim of this report is to present the results we found during our final project on *Generative Adversarial Networks* (GANs) for the MVA class Object Recognition and Computer Vision.
Our first goal was to be able to reproduce results given in the original article on Cycle-Consistent Adversarial Networks [2] for the Monet to photo style transfer. Our second goal was to train our GANs for a totally different task: transforming images of roads during nighttime into daytime and vice-versa. Finally, we read [3] to investigate possible ways of making GANs work even better via a modification in the loss function and implement those changes. In particular, we tested the performances of GANS and Wassertein GANs (WGAN) on new photos taken from a Brazil trip. To achieve those three goals we used two Amazon Web Services (AWS) GPU instances.

## 2 Reproducing Results

### 2.1 Reading and Understanding the GAN plus CycleGAN articles and useful resources

After having set up and running AWS, the next step was going through and understanding articles [1] and [2] with the help of the resources found in [4], [5], [6], [7] and [8]. Through our reading, we tried to get a grasp of the main ideas used in the GANs and adversarial GANs. Here are a few key-points we retained from our reading : learning a probability distribution, then learning transfer from one distribution to another by learning both ways and enforcing cycling, using appropriate norms : norm $\ell_1$ for cycling and identity retrieval, using $\ell_2$ instead of the log-loss for discriminators. The resources previously cited helped us understand deconvolution, the use of residual networks or skip-connections in deep networks to ensure information propagation and ease learning of a mapping by learning a residual mapping coupled to the identity. We also understood why instance normalization is crucial for contrast smoothing and reduces content-specific contrast in the content image which makes it easier for the generator to learn to translate from input images. Finally we documented ourselves on the use of PatchGANs for the discriminator and on the Adam optimization method.

### 2.2 Trying out the algorithm

Once that theoretical work was done, we could start the trial phase using the code written in PyTorch by the authors of [2] which can be found here :
`https://github.com/junyanz/pytorch-CycleGAN-and-pix2pix`. We did several runs with different parameters. Just to get going, and to have an idea of the run-time of the algorithm, we started out by running **100 epochs** (75 epochs with constant learning rate, plus 25 with linearly decreasing learning rate) on a dataset of **4000 images** of size **128×128**. We then tried **200 epochs** (150 + 50) with a training set of **2000** images of the same size as previously. The additional training

length did not result in a major improvement in the quality of the results. As a general comment, we observed that the algorithm had much more trouble converting Monet paintings to real photographs but results where globally satisfactory in the overall.

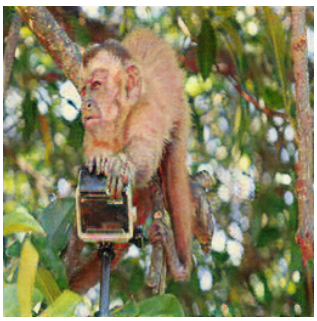
Our next run was a far more consequent run since we ran **100 epochs** (75 + 25) on **2000** images of size **256×256** (not forgetting this time to set the residual blocks of the generators to the number of 9 instead of 6 previously). While the results were already pretty good on the previous smaller images, the results of this run are truly impressive. Seeing the result on bigger images really shows the power of the algorithm and the ability to produce paintings from photos as well as to reconstruct photos from paintings is breathtaking. Some resultts are visible in the first two appendices **Appendix A**, **Appendix B**


Figure 1: DA, DB, GA and GB refer to the two discriminators and generators in the GAN. This plots presents their performance during the 100 epochs of the last training. The fact that generators performances increase while discriminators performances decrease is a good sign that shows that our GAN improve with epochs. Moreover, the best performances are obtained by GA, the Monet painting generator. This confirms our previous impression: it is easier to make a fake Monet than a fake photo.

## 2.3 Testing the learned generator on our own images

In this part, we are going to use the learned generator (from photo to Monet) during the previous training on $256 \times 256$ images to try transforming photos from our personal collection into Monet paintings and thus see if the algorithm truly generalizes well on images which share no similarities with the training images. The algorithm has been tested on various photos taken during a trip to Brazil. As we will see further, the results are not as good as before : many examples show that on those photos from Brazil, the algorithm is not able to reproduce Monet's style at all and we do not even recognize a painting in the generated image but we only see an image which is a slightly blurred, more colorful and brighter image.

It is not surprising though that the algorithm trained on a specific dataset of photographs, and thus a specific data distribution, would not be able to generate proper paintings from photographs coming from a different data distribution. Photos that are not frequent or probable under the training distribution (that is to say photos which are not similar to many of the photo samples used during training, and on which the corresponding distribution does not put a lot of mass) will give the learned generator trouble to transform into Monet paintings. This is what happens with the photos from Brazil : most of the photos taken do not bear any resemblances with the photos from the training set and will thus not yield good paintings. On the other hand, among the photos taken in Brazil, those rare ones which feature similar elements to those of the training set yield paintings more likely resembling a Monet. This phenomena is illustraded in the following Figure.



(a) Poor painting transfer on a Brazil photo

(b) Correct painting transfer on a Brazil photo

Figure 2: The algorihtm is not able to transform the monkey into a painting but does it very well with a landscape. More examples and their original photos can be found on the first two columns of **Appendix G**

# 3 Trying out GANs for night to day transfer : the Alderly dataset

We chose a dashcam images set also known as the Alderley Dataset. It was captured in two different conditions for the same route: one on a sunny day and the other one during a rainy night. Both datasets were taken in the suburb of Alderley, Queensland. At night, weather conditions are extremely bad and this mixture of dazzling light, darkness and water makes it really difficult to distinguish anything in some photograhs. Yet, as we will see, the fact that all Alderley images are very similar together will help GANs in the learning step.

Our results have been obtained after 60 epochs of training on 1000 images. Some of them are presented in **Appendix C** and **Appendix D**. It seems that the networks learn some tricks. For the night to day translation, for example, they manage to detect the road, to change dark lateral areas into trees and replace vertical lights into pols and street lights.

More bugs are observed when the test images do not come from the Alderley dataset. Results for this step are presented in **Appendix E**. One can observe that the networks handle badly objects they have never seen before like humans or bridges. Moreover, the contrast and the illumination are sometimes badly choosen and not very natural.

# 4 Implementing the Wasserstein Loss

This part is based on our reading of the article [3] on the Wasserstein Distance. The Wasserstein distance is a distance between probability distributions, and as such is able to measure how close or how far two different probability distributions are. This is very relevant to the GAN framework since, in this case, the objective is to be able to learn to learn a probability distribution from a given one (Monet to photo or the other way around). This notion of distance has many advantages compared to other distances or divergences defined on probability distributions. We will not detail the differences, but let us state however that the Wasserstein distance has the good property of being weaker than the others.

In practice, this Wasserstein distance does not bring tremendous change to the shape of the loss used : it simply boils down to removing the log and keeping only the arguments of the log in the discriminator loss and in the generator loss (in the term which is neither the cycling term nor the identity term), and clamping the weights of the networks to keep them in a compact space.

Here, we have thus modified the PyTorch code and added a file **cycle_wgan_model.py** where we implemented the new loss. Examples of results with the Wasserstein GAN (WGAN) model are shown in **Appendix F**. We can observe that the images are more pixelated than before, but that

the photo to Monet transfer still seems to perform pretty well. On the other hand, this new WGAN is behaving pretty poorly when transforming paintings into photographs, precisely because the the rendering is very pixelated, so that the result of the transfer does not look at all like a photograph, which should on the contrary have pretty good resolution ! However, as we will see below, this WGAN seems to be generalizing better than the previous GAN since it appears to perform better on the photographs from Brazil.

In Figure 3we compare, for 2 different photographs from Brazil, the paintings generated by the GAN of part 2 and the new WGAN. It seems quite obvious that the result generated by the WGAN looks more like a real painting than that of the previous GAN. More comparisons like this are provided in **Appendix G**

(a) Photo (b) GAN painting (c) WGAN painting

Figure 3: Comparing painting generation on Brazil Images

# References


[1] Ian J. Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, Yoshua Bengio. *Generative Adversarial Nets*. NIPS 2014.

[2] Jun-Yan Zhu, Taesung Park, Phillip Isola, and Alexei A. Efros. *Unpaired Image-to-Image Translation using Cycle-Consistent Adversarial Networks*. ICCV 2017.

[3] Martin Arjovsky, Soumith Chintala, and Léon Bottou. *Wasserstein GAN*. ICML 2017.

[4] LISA lab. *Convolution arithmetic tutorial*.
`http://deeplearning.net/software/theano/tutorial/conv_arithmetic.html`, 2017.

[5] Michael Dietz. *Understand Deep Residual Networks - a simple, modular learning framework that has redefined state-of-the-art*.
`https://blog.waya.ai/deep-residual-learning-9610bb62c355`, 2017.

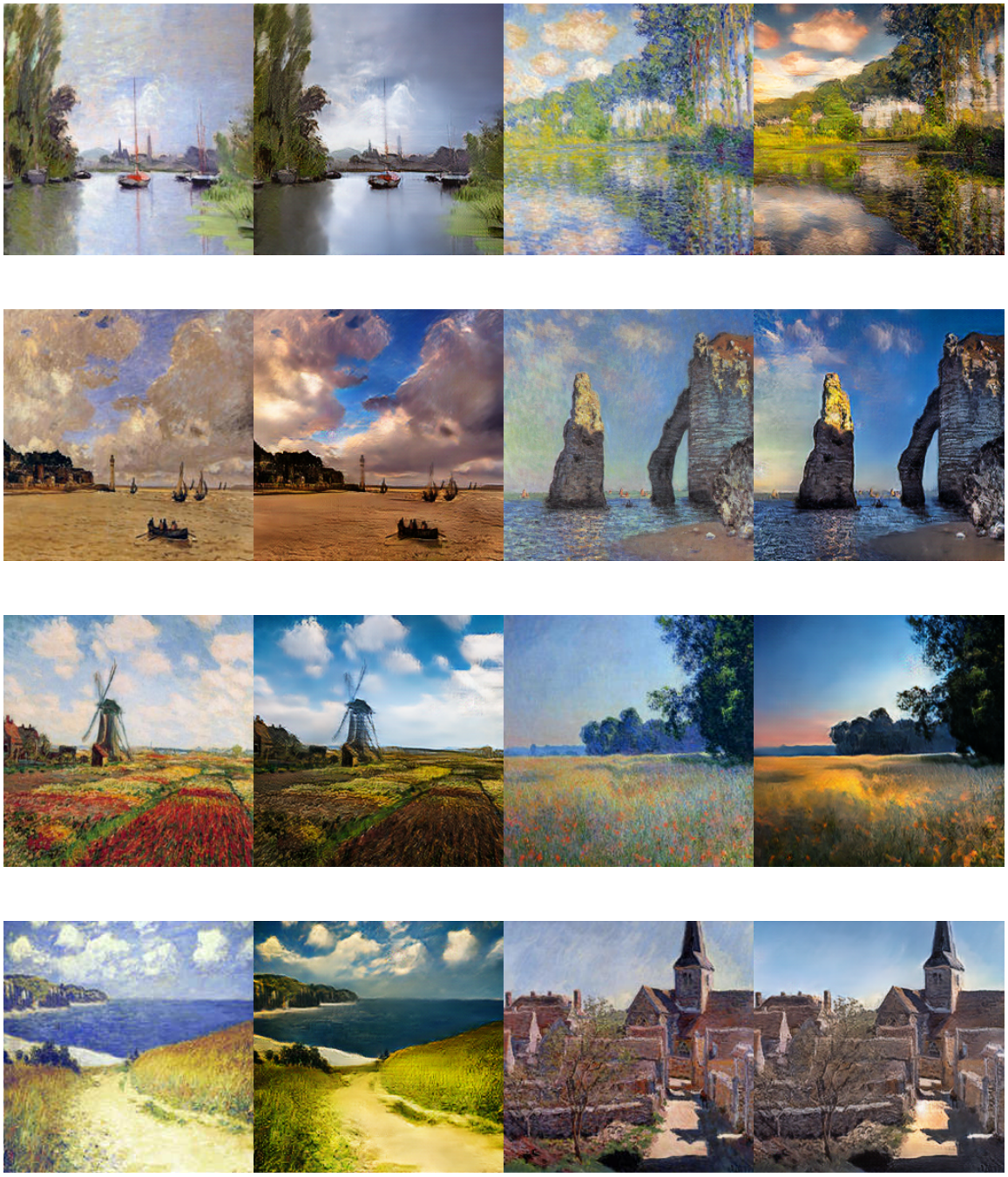
[6] Dmitry Ulyanov, Andrea Vedaldi, Victor Lempitsky. *Instance Normalization: The Missing Ingredient for Fast Stylization*, 2017.

[7] Phillip Isola Jun-Yan Zhu Tinghui Zhou Alexei A. Efros. *Image-to-Image Translation with Conditional Adversarial Networks*

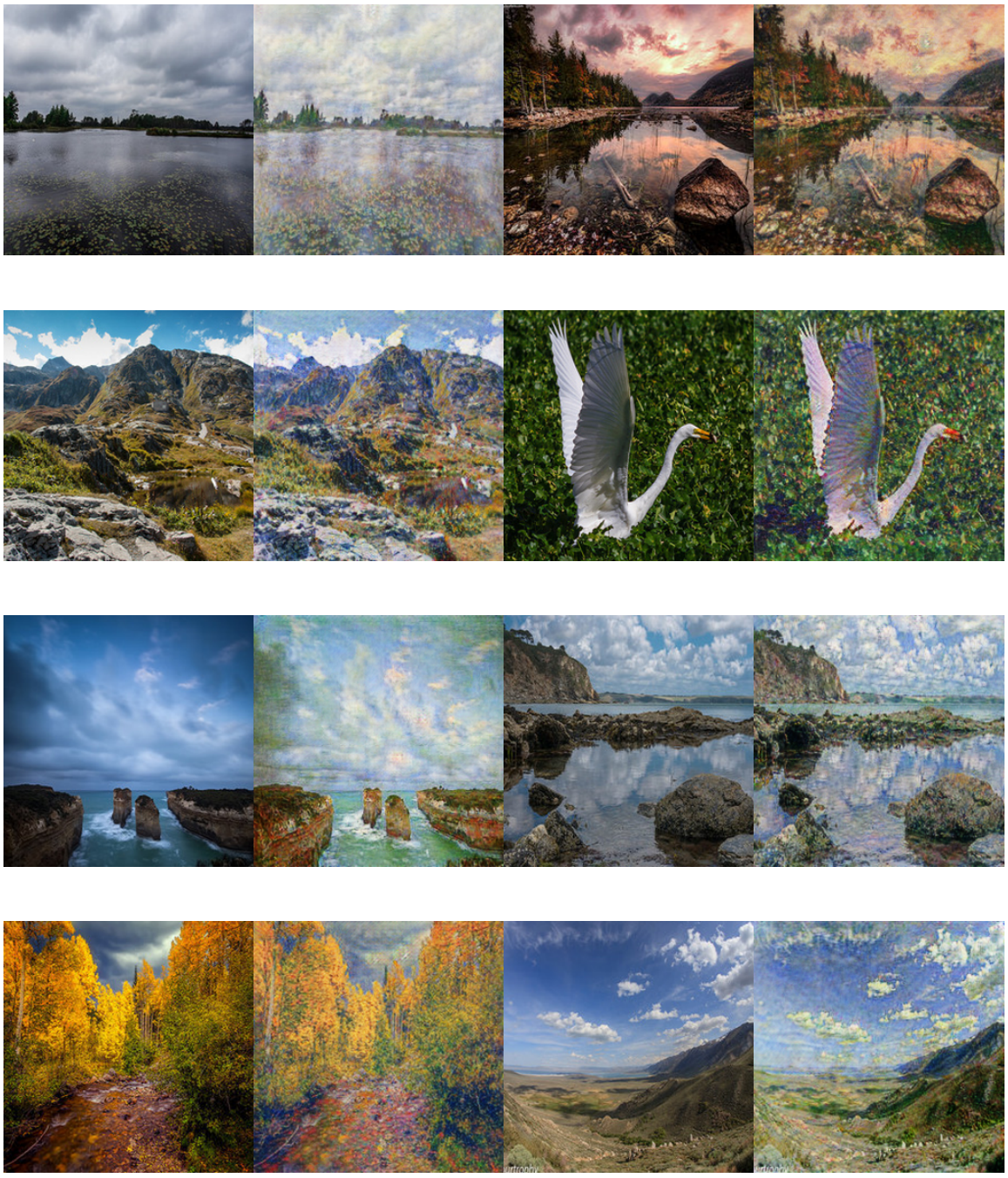
[8] Jason Brownlee. *Gentle Introduction to the Adam Optimization Algorithm for Deep Learning*
`https://machinelearningmastery.com/adam-optimization-algorithm-for-deep-learning/`

# Appendices

## A Monet to Photo



## B Photo to Monet



## C Alderley Day to Rainy Night

## D Alderley Rainy Night to Day

## E Testing Alderley Day to Night on random road images

## F Wasserstein GANs

## G Original / GAN / WGAN tested on Brazil trip photos