Generative Adversarial Networks (GANs)
Synthetic report

Author: Sylvain Combettes
Department: Applied Mathematics
Track: Big Data & Data Science

sylvain.combettes [a t] mines-nancy.org
sylvain-combettes
sylvaincom

1 Introduction

Over the past decade, the explosion of the amount of available data – Big Data – the optimization of algorithms and the constant evolution of computing power have enabled **artificial intelligence** (AI) to perform more and more human tasks. In 2018, Google's CEO Sundar Pichai predicted that AI will have a more profound impact than electricity or fire.

If we claim that the purpose of AI is to simulate human intelligence, the main difficulty is creativity. In the field of AI, we talk about **generative models** and one of the most popular model nowadays is GANs (for "generative adversarial networks"). In a 2016 seminar, Yann LeCun has called GANs "the coolest idea in deep learning in the last 20 years."

In 2014, Ian Goodfellow invented GANs [1] and was then nicknamed "the GANfather". GANs introduce the concept of **adversarial learning**, as they lie in the rivalry between two neural networks. These techniques have enabled researchers to create realistic-looking but entirely computer-generated photos of people's faces. They have also allowed the creation of controversial "deepfake" videos. Actually, GANs can be used to imitate any data distribution (image, text, sound, etc.).

An example of GANs' results from 2018 is given figure 1: these images are fake but very realistic. The generation of these fictional celebrity portraits, from the database of real portraits CelebA-HQ composed of 30 000 images, took 19 days. The generated images have a size of 1024×1024 but have been compressed in this report.

Figure 1: Realistic but fictional portraits of celebrities generated from originals using GANs
Source: Nvidia [4]

2 How do GANs work?

Generative adversarial networks (GANs) are a generative model with implicit density estimation, part of unsupervised learning and are using two neural networks. Thus, we understand the terms "generative" and "networks" in "generative adversarial networks".

2.1 The principle: generator vs discriminator

The principle is a two-player game: a neural network called the generator and a neural network called the discriminator. The **generator** tries to fool the discriminator by generating real-looking images while the **discriminator** tries to distinguish between real images and fake images. Hence, we understand the term "adversarial" in "generative adversarial networks". See figure 2.

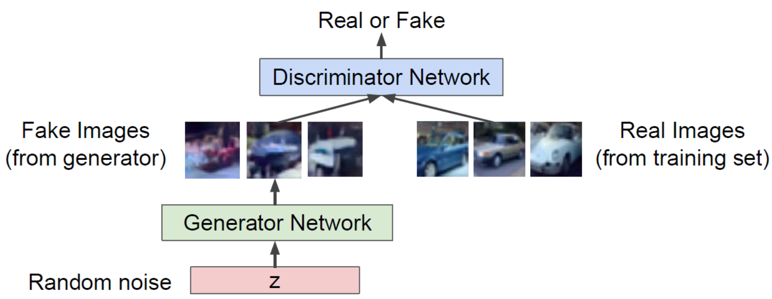

Figure 2: Roles of the generator and the discriminator
Source: Stanford CS231n [2]

At the bottom left of figure 2, we can see that our generator samples from a simple distribution: **random noise**.

The generator can be interpreted as an artist and the discriminator as an art critic. See figure 3.

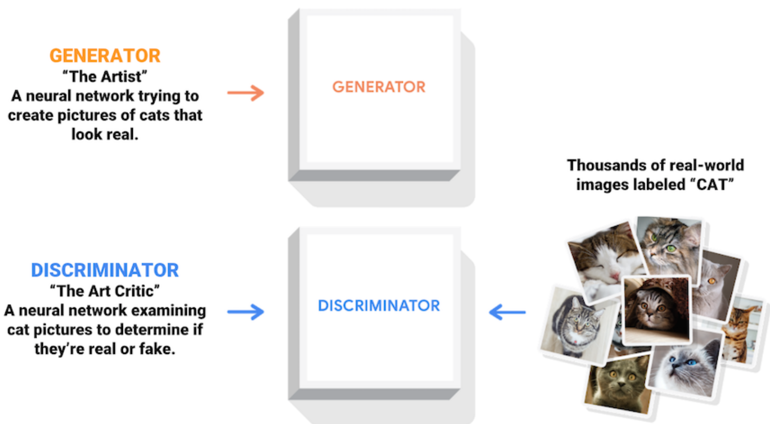

Figure 3: Interpretation: roles of the generator and the discriminator
Source: `https://www.tensorflow.org/beta/tutorials/generative/dcgan`

During training, the generator progressively becomes better at creating images that look real, while the discriminator becomes better at telling them apart. The process reaches equilibrium when the discriminator can no longer distinguish real images from fake. See figure 4. Thus, if the discriminator is well trained and the generator manages to generate real-looking images that fool the discriminator, then we have a good generative model: we are generating images that look like the training set.

After this training phase, we only need the generator to sample new (false) realistic data. We no longer need the discriminator. Note that the random noise z guarantees that the generator does not always produce the same image (which can deceive the discriminator).

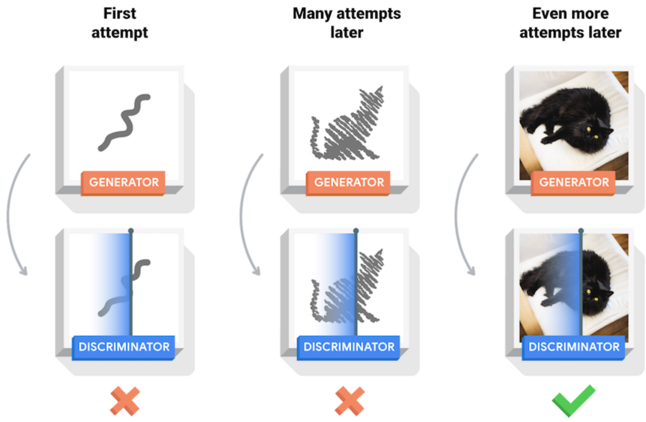

Figure 4: Generator and discriminator training
Source: `https://www.tensorflow.org/beta/tutorials/generative/dcgan`

Note that at the beginning of the training in figure 4, the generator only generates a random noise that does not resemble the training data.

2.2 Mathematically: the two-player minimax game

The generator G and the discriminator D are jointly trained in a **two-player minimax game** formulation. The minimax objective function is:

$$\min_{\theta_g} \max_{\theta_d} \left[\mathbb{E}_{x\sim p_{\text{data}}} \log D_{\theta_d}(x) + \mathbb{E}_{z\sim p(z)} \log \left(1 - D_{\theta_d}\left(G_{\theta_g}(z) \right) \right) \right] \tag{1}$$

where θ_g is the parameters of G and θ_d is the parameters of D.

In the following, we simply refer to D_{θ_d} as D and G_{θ_g} as G.

By definition, D outputs the likelihood of real image in interval $[0,1]$:

- $D(x)$ equals 1 (or is close to 1) if D considers that x is a real data
- $D(x)$ equals 0 (or is close to 0) if D considers that x is a fake data (e.g. a generated data).

We can prove that, at the equilibrium, D outputs $1/2$ everywhere because D can not distinguish fake generated data from real data.

Because $\mathrm{x} \sim p_{\text{data}}$, x is a real data. By definition of G, $G(z)$ is a fake generated data. For example, x would be a real-life image of a cat and $G(z)$ would be a fake generated image of a cat. Thus, $D(x)$ is the output of the discriminator for a real input x and $D(G(z))$ is the output of the discriminator for a fake generated data $G(z)$.

GOODFELLOW wrote the two-player minimax game (1) such that θ_g and θ_d evolve so that the following points from subsection 2.1 are true:

- The discriminator D tries to distinguish between real data x and fake data $G(z)$.
  More precisely, the discriminator D plays with θ_d (θ_g being fixed) to maximize the objective function such that $D(x)$ is close to 1 (x being real data) and such that $D(G(z))$ is close to 0 (a generated data is detected as false).

- The generator G tries to fool the discriminator D into thinking that its fake generated data is real.
  More precisely, the generator G plays with θ_g (θ_d being fixed) to minimize the objective function such that $D(G(z))$ is close to 1 (a false generated data is detected as true by the discriminator).

Although we are in unsupervised learning (the data is not labeled), we choose that the data generated by G has a 0 label for false (regardless of what the discriminator returns) and the real learning data has a 1 label for true. We can thus define a loss function.

GOODFELLOW's first paper on GANs [1] proves that the minimax game has a global (and unique) optimum for $p_g = p_{\text{data}}$ where p_g is the generative distribution and p_{data} the real data distribution. However, in practice, having p_g to converge towards p_{data} is not easy.

3 Why are GANs so interesting?

Generative models have several very useful applications: colorization, super-resolution, generation of artworks, etc. In general, the advantage of using a simulated model over the real model is that the computation can be faster.

Many interesting examples are given in GOODFELLOW's tutorial [3] and Stanford's lecture [2]. In particular, examples given by GOODFELLOW in the conference « Generative Adversarial Networks (NIPS 2016 tutorial) », from 4:15 to 12:33, are impressive. The link to this video is the following: `https://www.youtube.com/watch?v=HGYYEUSm-0Q`.

One example is given figure 5. These real images are transposed into realistic fictional images - or vice versa - with the CycleGan developed by researchers at the University of Berkeley. The concept, called **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. There exists a tutorial on CycleGAN for Tensorflow: `https://www.tensorflow.org/beta/tutorials/generative/cyclegan`.

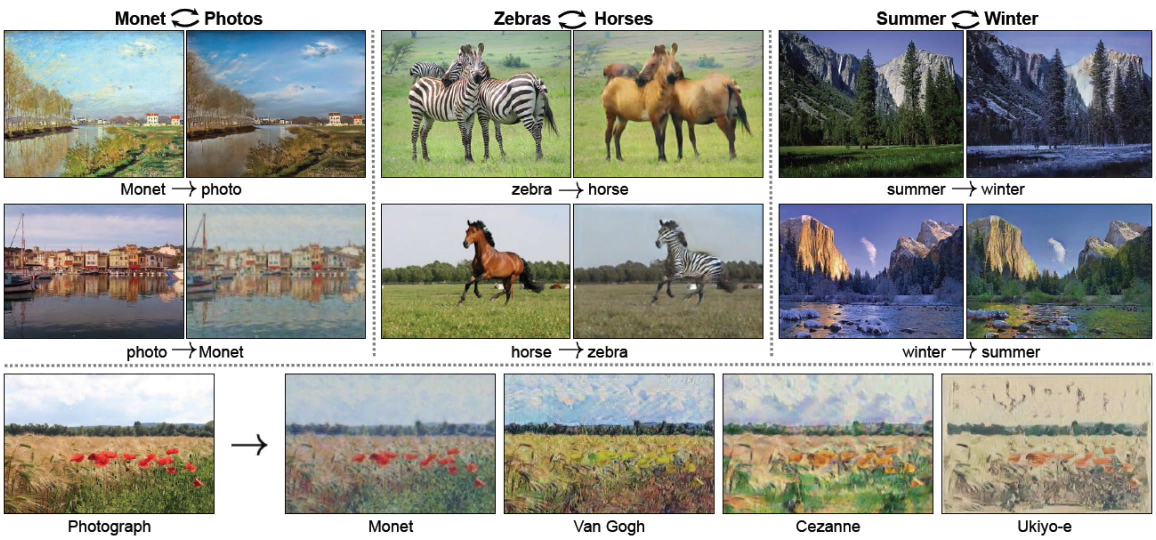

Figure 5: CycleGAN: real images transposed into realistic fictional images using GANs
Source: Berkeley AI Research (BAIR) laboratory [5]

A second example is shown in figure 6. For example, the aerial to map feature can be very useful to Google Maps or similar applications and the edges to photo feature can help designers.

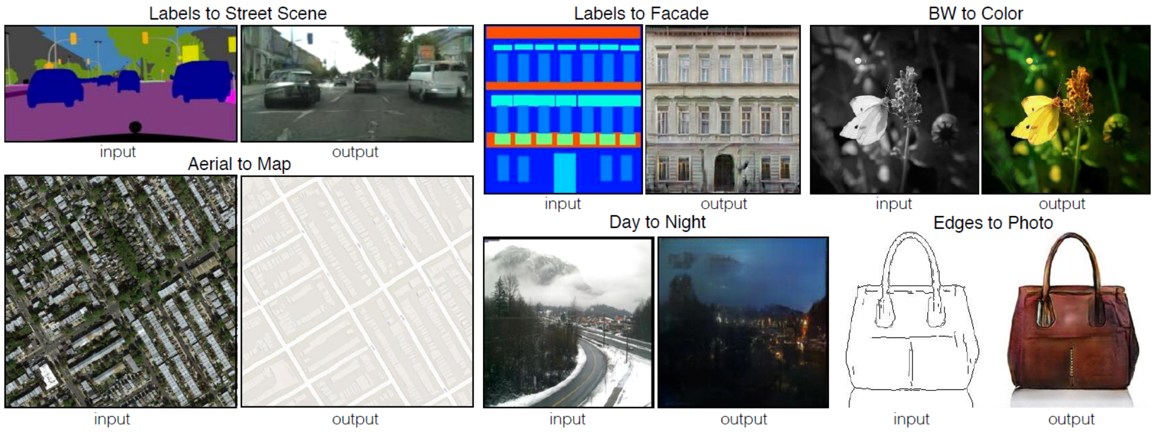

Figure 6: Several types of image transformations using GANs
Source: Berkeley AI Research (BAIR) Laboratory [6]

4 Conclusion

GANs' applications have increased rapidly, in particular for images. I believe that GANs can be very interesting for companies, in particular for Servier. For example, GANs can generate realistic images of new medical images and image-to-image translation can help designers draw and be more creative. Moreover, GANs can be used for data augmentation when we only have one hundred images and we wish to have more. However, as Francis Bach explained in his talk at DataJob 2018, as of today, only a human eye can say that the generated images actually look good: there is no rigorous performance measure.

GANs have also recently been developed for binary outputs (sick or not) or discrete outputs (rounded blood pressure, rounded weight...). Servier can benefit a lot from this new research. For example, instead of sending confidential data from Excel sheets, they can send fake realistic data (that keeps the correlation between the variables) to their partners.

References

[1] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozairy, A. Courville, Y. Bengio. Generative Adversarial Nets. *arXiv:1406.2661v18*, 2014.

[2] F.-F. Li, J. Johnson, S. Yeung. CS231n: Convolutional Neural Networks for Visual Recognition. Lecture 13 | Generative Models. Spring 2017. `http://cs231n.stanford.edu/`

[3] I. Goodfellow. NIPS 2016 Tutorial: Generative Adversarial Networks. *arXiv:1701.00160v4*, 2017.

[4] T. Karras, T. Aila, S. Laine, J. Lehtinen. Progressive Growing of GANs for Improved Quality, Stability, and Variation. NVIDIA, *arXiv:1710.10196v3*, 2018.

[5] J.-Y. Zhu, T. Park, P. Isola, A.Efros Unpaired Image-to-Image Translation using Cycle-Consistent Adversarial Networks. Berkeley AI Research (BAIR) laboratory, UC Berkeley. *arXiv:1703.10593v6*, 2018.

[6] P. Isola, J.-. Zhu, T. Zhou, A. Efros. Image-to-Image Translation with Conditional Adversarial Networks. Berkeley AI Research (BAIR) Laboratory, UC Berkeley *arXiv:1703.10593v6*, 2018