Normandie Université

THÈSE

Pour obtenir le grade de Docteur de Normandie Université

Spécialité Informatique

l'École Doctorale Mathématiques, Information, Ingéniérie des Systèmes

Auxiliary Tasks for the Conditioning of Generative Adversarial Networks

Tâches auxilliaires pour le conditionnement des réseaux antagonistes génératifs

Présentée et soutenue par

Cyprien RUFFINO

Dirigée par Gilles GASSO et Romain HÉRAULT

Thèse soutenue publiquement le Wednesday 20th May, 2020
devant le jury composé de

Civilité / prénom NOM,	Grade / fonction / statut / lieu d'exercice	Rapporteur ou examinateur ou directeur de thèse ou codirecteur de thèse
Civilité / prénom NOM,	Grade / fonction / statut / lieu d'exercice	Rapporteur ou examinateur ou directeur de thèse ou codirecteur de thèse
Civilité / prénom NOM,	Grade / fonction / statut / lieu d'exercice	Rapporteur ou examinateur ou directeur de thèse ou codirecteur de thèse
Civilité / prénom NOM,	Grade / fonction / statut / lieu d'exercice	Rapporteur ou examinateur ou directeur de thèse ou codirecteur de thèse
Civilité / prénom NOM,	Grade / fonction / statut / lieu d'exercice	Rapporteur ou examinateur ou directeur de thèse ou codirecteur de thèse

INSA ROUEN NORMANDIE

MIΣ

litis

Abstract

Résumé

Remerciements

Contents

List of Figures

List of Tables

Acronyms

CGAN	Conditional Generative Adversarial Networks
CycleGAN	Cycle-Consistent Generative Adversarial Networks
ELBO	Evidence Lower Bound
FID	Fréchet Inception Distance
GAN	Generative Adversarial Networks
GMM	Gaussian Mixture Model
IS	Inception Score
JS	Jensen-Shannon (Divergence)
KL	Kullback-Leibler (Divergence)
MSE	Mean-Squared Error
ReLU	Rectified Linear Unit
SGD	Stochastic Gradient Descent
VAE	Variational Auto-Encoder

Introduction

Context

Generic deep learning introduction, generic introduction to generative modeling (image generation, whichfaceisreal.com, etc...)

Generative Adversarial Networks (GAN) Goodfellow et al., 2014 have been recently highlighted for their ability to generate photo-realistic images. By providing a simple framework for high-quality, high-dimensional generative modeling, they quickly found real-world applications such as the notorious "deepfakes" (Vaccari & Chadwick, 2020), face-aging (Antipov, Baccouche, & Dugelay, 2017), image super-resolution (Wang, Chen, & Hoi, 2020), map style tranfer (Kang, Gao, & Roth, 2019), video prediction (Vondrick, Pirsiavash, & Torralba, 2016) or 3D objects generation (Wu, Zhang, Xue, Freeman, & Tenenbaum, 2017).

Introduction to applied conditional generative modeling : examples others than geology

Motivations

Geostatistical application : introduction and needs

- Tuneable (quality vs enforcement of the constraints)
- Pixel-precise
- Keeping diversity

Polarimetry application : introduction and needs

- Designing custom-made constraints for the problem
- Non-euclidian
- Compatible with domain transfer

Contributions

Outline

Chapter 1

Generative Adversarial Networks : Principles, strengths and limitations

Chapter abstract

In this chapter, we first propose an introduction to the problem of generative modeling and some solutions to tackle this problem. We then propose an overview of the Generative Adversarial Networks (Goodfellow et al., 2014) framework, which is a recent method to train deep neural networks as generative models that is particularly adapted to the task of image generation. We will introduce some of its theoretical interpretations, as well as some of its variations and applications. We discuss the different limitations of this approach and expose a trilemma between the quality of the generated samples, their diversity and the conditioning of the model. We then discuss the recent advances that have been made to overcome some of these limitations and propose a taxonomy of these advances using the aforementioned trilemma. Finally, we discuss the evaluation of generative models and the difficulties of evaluating the intrinsic quality of a generated sample. We propose an overview of the different classical metrics and discuss their limitations.

Contents

1.1 Generative modeling

Generative modeling with deep neural networks has been a challenging task due to the stochastic nature of sampling, which prevents the computation of gradient, thus preventing the classical training of a deep model with gradient descent. However recent approaches such as variational autoencoders (VAEs) (Kingma & Welling, 2014), flow methods (Dinh, Sohl-Dickstein, & Bengio, 2017; Kingma & Dhariwal, 2018) and adversarial models (Goodfellow et al., 2014) managed to overcome this restriction. In this section, we first propose an introduction to generative modeling with a focus on latent variable models.

1.1.1 Generative modeling with maximum likelihood estimation

Generative modeling is the task of learning the underlying distribution of a dataset in order to generate more samples from that distribution. In other words, it describes how data is generated in terms of a probabilistic model, a distribution from which the whole dataset could have been sampled with a high likelihood.

Indeed, whereas a discriminative model tries to find decision boundaries by fitting a parametric model $p_\theta(y|x)$ to a conditional probability distribution $p(y|x)$ of data $x \in \mathcal{X}$ and labels $y \in \mathcal{Y}$ relatively to the data $x \sim p(x)$, a generative model aims to fit $p_\theta(x)$ to $p(x)$ the intrinsic distribution of the data and to provide a sampling mechanism on this distribution (see Figure. 1.1).

These two learning tasks, the discriminative (Equation. (1.1)) modeling and the generative modeling (Equation. (1.2)) can be formulated as a maximum log-likelihood estimation

$$\theta^* = \arg\max_\theta \mathop{\mathbb{E}}_{x,y\sim p(y|x)} \log p_\theta(y|x) \qquad (1.1) \qquad\qquad \theta^* = \arg\max_\theta \mathop{\mathbb{E}}_{x\sim p(x)} \log p_\theta(x) \qquad (1.2)$$

An simple example of generative model are Gaussian Mixture Models (GMM) . They consist in a sum of K Gaussian distributions $\mathcal{N}(\mu_k, \sigma_k^2), k \in 1..K$ which are all attributed a selection probability $p(z=k) = \pi_k$, with $z \in \mathcal{Z}$, so that $p(x|z=k) = \mathcal{N}(\mu_k, \sigma_k^2)$. The model is then formulated as

$$p_\theta(x) = \sum_z p(z)\, p_\theta(x|z) \ ,$$

with log-likelihood

$$\log \sum_{x\sim p(x)} p_\theta(x) = \sum_{x\sim p(x)} \log \sum_{k=1}^{k} \pi_k \mathcal{N}(x|\mu_k, \sigma_k^2) \ .$$

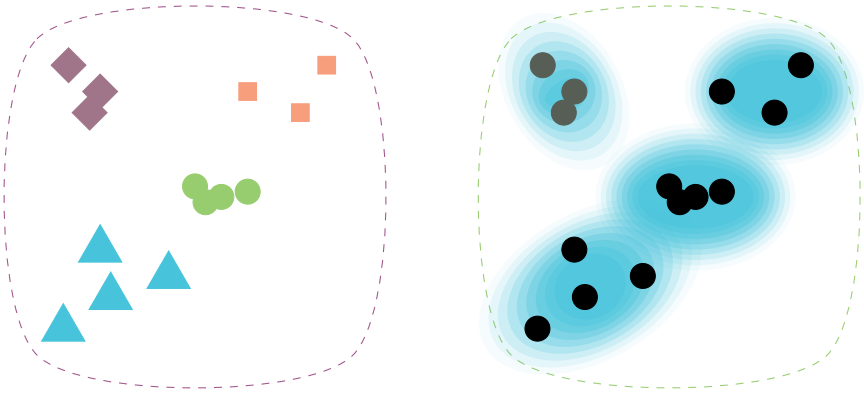


Figure 1.1: Left: Discriminative modeling, the model aims to assign a class to each data point. Right: Generative modeling, the model aims to learn the underlying probability distribution of the data points.

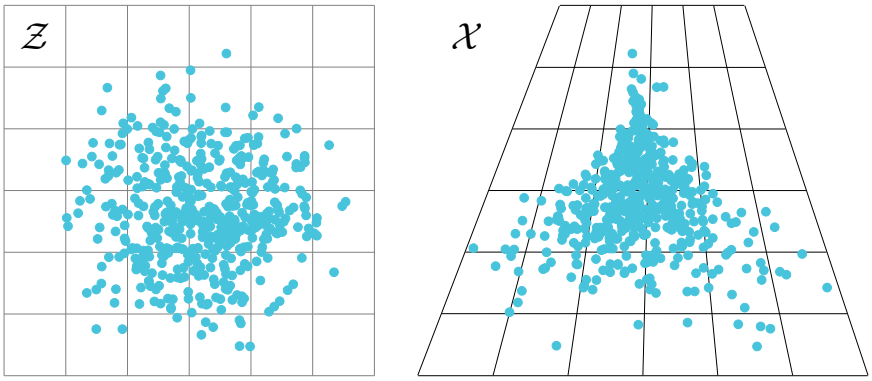


Figure 1.2: A mapping between a latent space $\mathcal{Z}$ and the space of a dataset $\mathcal{X}$.

In the case of the GMMs, the Expectation-Maximization (EM) algorithm (Dempster, Laird, & Rubin, 1977) can be used to find the parameters θ^* which, at convergence, maximizes the log-likelihood of the model. Once the model is trained, sampling a new data is done by picking a component k from the distribution $p(z)$, then drawing a sample from the Gaussian distribution $p(x|z=k) = \mathcal{N}(\mu_k^*, \sigma_k^{*2})$.

1.1.2 Latent variable models

Latent variable models and marginalization

For GMMs, sampling a new point consists in, once the Gaussian component has been selected, sampling a point on a normal distribution. This sampling can be done by using reparametrization: instead directly sampling $x \sim \mathcal{N}(\mu_k^*, \sigma_k^{*2})$, we can instead sample a latent variable $z \sim \mathcal{N}(0,1)$ and compute $x = G(z;\mu,\sigma) = \mu + z\sigma$. Such a model, that consists in a deterministic function $G : \mathcal{Z} \to \mathcal{X}$ with parameters θ applied to a random latent variable drawn from a fixed distribution $p(z)$ is a latent variable model.

Since more complex distributions does not necessarily provide a natural sampling mechanism, using a latent variable model allows to outsource the stochastic part of the sampling process from the learning process and only learn the function $G(z;\theta)$. More formally, instead of directly modeling $p(x)$, a latent variable model learns a deterministic mapping $p_G(x|z)$. From this mapping, the generative model can be obtain through marginalization

$$p_G(x) = \int_{\mathcal{Z}} p(z)\, p_G(x|z)\, dz = \int_{\mathcal{Z}} p(z)\, p(x|G(z;\theta))\, dz \ . \tag{1.3}$$

This marginalization allows for the use of an arbitrary flexible G. However, if G is non-linear, the actual evaluation of $p_G(x)$ is very likely to be intractable due to the integral over $\mathcal{Z}$, which prevents the training of such a model as is.

While the marginal distribution $p_G(x)$ cannot be explicitly computed for any function G, several solutions exist to overcome this problem and train deep generative models with latent variables anyways.

Variational auto-encoders

Variational Auto-Encoders (VAE) (Kingma & Welling, 2014) are deep latent variable models which tackle the marginalization problem by approximating the integral using a variational approach. To this end, they both learn the distribution of the latent model $p_G(x|z)$ as well as the distribution $q_F(z|x)$. This is done with with two different neural networks, a decoder network $G : \mathcal{Z} \to \mathcal{X}$ and an encoder network $F : \mathcal{X} \to \mathcal{Z}$ and allows to compute the distribution $p(x)$ as

$$\log p_G(x) - D_{KL}\Big(q_F(z|x)\Big\|p(z|x)\Big) = \mathop{\mathbb{E}}_{z\sim q_F(z|x)}\big[\log p_G(x|z)\big] - D_{KL}\Big(q_F(zx)\Big\|p(z)\Big) \ .$$

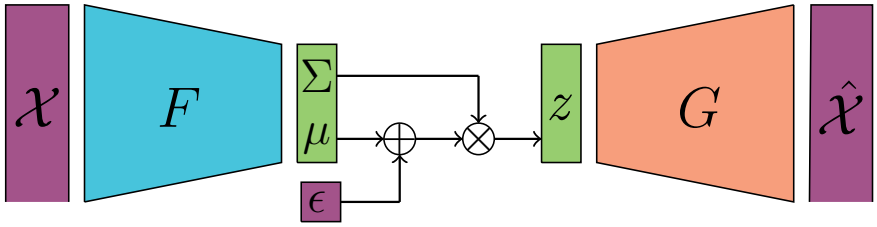


Figure 1.3: Variational auto-encoder framework

The KL terms evaluates the distance between the distribution $q_{\mathrm{F}}(z|x)$ learned by the encoder and real distribution $p(z|x)$, and since $p(z)$ is chosen Gaussian, this KL terms can be explicitly computed. The first term, is equivalent to the reconstruction error of an auto-encoder. Hence the model is trained by minimizing

$$\mathrm{L}_{\mathrm{VAE}}(\mathrm{F},\mathrm{G}) = \mathop{\mathbb{E}}_{z\sim q_{\mathrm{F}}(z|x)}\left[||x-\mathrm{G}(z)||_2^2\right] - \mathrm{D}_{\mathrm{KL}}\Big(q_{\mathrm{F}}(z|x)\Big|\Big|p(z)\Big)$$

However, since sampling $z\sim q_{\mathrm{F}}(z|x)$ is not differentiable, the VAE uses the so-called *reparametrization trick*, that is to have $\mathrm{F}(x)$ output the mean and the variance (μ_x,σ_x^2) of a normal distribution for a sample x, so that a $\epsilon\sim\mathcal{N}(0,1)$ is sampled outside of the model and given as a parameter, thus allowing to compute $z=\mu_x+\sigma_x\epsilon$, which is differentiable by considering ϵ a parameter.

Finally, generating a sample x with the trained model can be done by sampling a random vector $\epsilon\sim\mathcal{N}(0,1)$ and computing $x=\mathrm{G}(z)$.

Normalizing flows

Normalizing flow based techniques is a family of latent variable models that aim to tackle the marginalization problem by using the *change of variable formula*

$$p_{\mathrm{G}}(x) = p(z)\left|\det\left(\frac{\partial \mathrm{G}(z)}{\partial z^{\mathrm{T}}}\right)\right|^{-1} = p(\mathrm{G}^{-1}(x))\left|\det\left(\frac{\partial \mathrm{G}^{-1}(x)}{\partial x^{\mathrm{T}}}\right)\right| ,$$

with $z\sim p(z)$ a latent variable. This formulation has notable advantages such as explicitly allowing the computation of the exact inference. However, the model has to enforce some tough constraints: the input and output dimensions must be the same; G must be invertible; and computing the determinant of the Jacobian needs to be efficient and differentiable.

These constraints can be enforced through strong restrictions on the architecture of the model. By limiting the transformations to a set of invertible transformations with a tractable Jacobian determinant, the model remains invertible and the determinant of its Jacobian can be computed efficiently.

Real-valued non-volume preserving (RealNVP) normalizing flows (Dinh, Sohl-Dickstein, & Bengio, 2017) uses affine coupling transformations, which transforms a variable by adding and scaling it by a non-linear transformation of itself, usually computed with deep neural networks. These transformations can be inverted by substracting and downscaling by the same transformed variables and their Jacobian is triangular, therefore computing its determinant can be done efficiently by computing the product of its diagonal terms. *Glow* (Kingma & Dhariwal, 2018) extended this set of transformations to 1×1 invertible convolutions as well as a variant of batch normalization that allows for more expressiveness in the model.

1.2 Adversarial models

In this section, we will focus on the Generative Adversarial Networks (Goodfellow et al., 2014) framework, their training process and some of their variants, especially their conditional and domain-transfer variants.

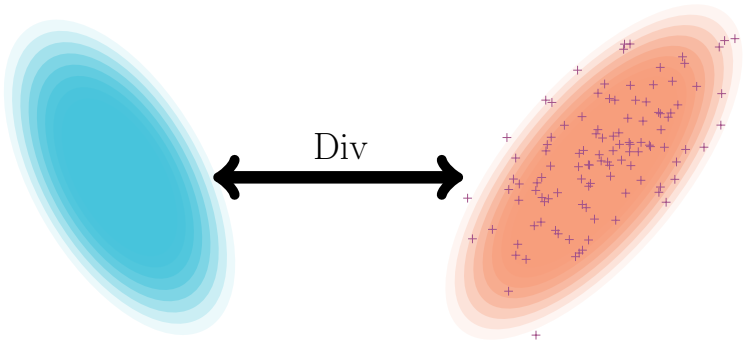


Figure 1.4: A divergence can capture the distance between a (parametric) distribution and the estimated distribution of a set of data points $\mathscr{X}$

We will then outline some limitations of this framework and propose a formulation of these limitations in the form of a trilemma between the intrinsic quality of the generated samples, their diversity and the quality of the conditioning of the model. With this tool, we propose a taxonomy of the recent GAN approaches and identify trends in these approaches.

1.2.1 Generative Adversarial Networks

In the same fashion as the generative models mentioned in Subsection. 1.1.2, Generative Adversarial Networks (GANs) (Goodfellow et al., 2014) aims to learn a parameterized mapping $p_{\mathrm{G}}(x|z)$ between a simple distribution $p(z)$ (usually normal or uniform) to the real data distribution $p(x)$. However, instead of relying on the likelihood and trying to estimate the distribution through marginalization, it aims to minimize an estimation of a divergence between $p(x)$ and the mapped distribution $p_{\mathrm{G}}(x)$. Therefore, GANs are often qualified as *likelihood-free* generative models.

Since a divergence $\mathrm{Div}(p(x)||q(x))$ between two distributions $p(x)$ and $q(x)$ is analogous to a distance between these distributions (see Figure. 1.4), minimizing such a divergence allows for a parametric distribution $p_\theta(x)$ to fit a target distribution $p(x)$. When this divergence is both tractable (or estimable) and differentiable w.r.t the parameters θ, it can be directly optimized, allowing for the training of a generative model.

However in practice, such divergences usually intractable in the case of generic distributions. GANs aim to estimate this divergence by relying on a second learned function that will act as a surrogate to the divergence, the discriminator model D. This discriminator is a binary classifier that aims to predict the probability that a sample x was sampled on the real distribution $p(x)$ or was generated from $z \sim p(z)$ and is trained with binary cross-entropy

$$\mathrm{L_D}(\mathrm{D},\mathrm{G}) = \mathop{\mathbb{E}}_{x\sim p(x)}[\log \mathrm{D}(x)] + \mathop{\mathbb{E}}_{x\sim p_{\mathrm{G}}(x)}[1-\log \mathrm{D}(x)] \ .$$

The intuition behind this model is that once the discriminator is trained, maximizing its error on generated samples $x \sim p_{\mathrm{G}}(x)$ w.r.t the parameters of G should push $p_{\mathrm{G}}(x)$ towards $p(x)$.

The minimum of $f(x) = a\log(x) + b\log(1-x)$ is $\frac{a}{a+b}$, so the discriminator that maximizes $\mathrm{L_D}(\mathrm{D},\mathrm{G})$ for a fixed G is given by

$$\mathrm{D}^*_{\mathrm{G}}(x) = \frac{p(x)}{p(x)+p_{\mathrm{G}}(x)} \ .$$

By plugging this optimal into the discriminator cost, we get

$$\min_{\mathrm{D}} \mathrm{L_D}(\mathrm{D},\mathrm{G}) = \mathop{\mathbb{E}}_{x\sim p(x)}\left[\log \frac{p(x)}{p(x)+p_{\mathrm{G}}(x)}\right] + \mathop{\mathbb{E}}_{x\sim p_{\mathrm{G}}(x)}\left[1-\log \frac{p_{\mathrm{G}}(x)}{p(x)+p_{\mathrm{G}}(x)}\right] \ .$$

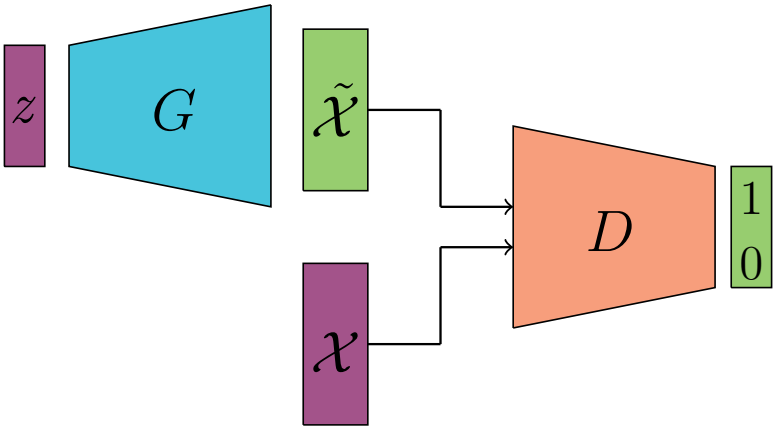


Figure 1.5: Generative Adversarial Networks framework

As said previously, the objective of the generator model G will be to maximize the error of the discriminator D. Thus, we can formulate a criterion $L_G(G)$ as $L_G(G) = \min_D L_D(D,G)$. Up to additive and multiplicative constants, the criterion $L_G(G)$ can be reformulated as

$$L_G(G) = D_{KL}\left(p(x)\middle\|\frac{p(x)+p_G(x)}{2}\right) + D_{KL}\left(p_G(x)\middle\|\frac{p(x)+p_G(x)}{2}\right) = 2\cdot D_{JS}\left(p(x)\middle\|p_G(x)\right) \ .$$

When the discriminator is trained to convergence, minimizing the criterion $L_G(G) = L_{GAN}(D^*,G)$ is equivalent to minimizing the Jensen-Shannon (JS) divergence between $p(x)$ and $p_G(x)$. This training process is summed up as a mini-max game in Equation. (1.4)

$$\arg\min_G \max_D L_{GAN} = \arg\min_G \max_D \mathop{\mathbb{E}}_{x\sim p(x)}[\log D(x)] + \mathop{\mathbb{E}}_{z\sim p(z)}[1-\log D(G(z))] \ . \tag{1.4}$$

As shown above, this mini-max game has, assuming infinite capacity for both G and D, a global optimum for $p(x) = p_G(x)$. The GAN training algorithm then consists in alternatively updating the discriminator and the generator via gradient ascent/descent. A summary of this process is presented in Algorithm. 1.

Algorithm 1 The GAN training algorithm

Require: $\mathcal{D}_X$ the real dataset, G the generator model, and D the discriminator model
  repeat
    sample a mini-batch $\{x_i\}_{i=1}^m$ from $\mathcal{D}_X$
    sample a mini-batch $\{z_i\}_{i=1}^m$ from $p(z)$
    update D by stochastic gradient ascent of
      $\sum_{i=1}^m \log(D(x_i)) + \log(1-D(G(z_i)))$
    sample a a mini-batch $\{z_j\}_{j=1}^n$ from distribution $p(z)$;
    update G by stochastic gradient descent of
      $\sum_{j=1}^n \log(1-D(G(z_j)))$
  until a stopping condition is met

1.2.2 Conditional modeling with CGANs

While classical generative models such as GANs try to unconditionally approximate the real-data distribution $p(x)$, a conditional generative model aim to learn a model of the conditional distribution $p(x|y)$, where $y \in \mathcal{Y}$ is a label of any kind.

Several extensions of the GAN framework allow for conditional modeling. First introduced, Conditional GANs (CGANs)(Goodfellow et al., 2014; Mirza & Osindero, 2014) simply adds the label

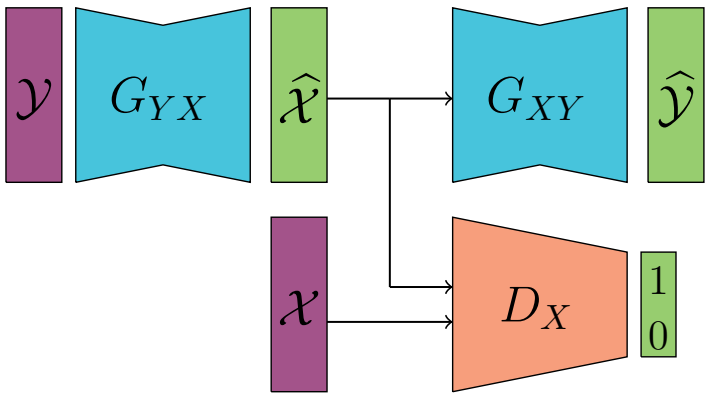


Figure 1.6: The CycleGAN approach. Half of the training setup is illustrated, the other half consisting in the same setup but with inverted X and Y

y as an input for both the discriminator and the generator. The new optimization problem that results from this change is summed-up in Equation. (1.5) as follows

$$\arg\min_{\mathrm{G}}\max_{\mathrm{D}} \mathrm{L}_{\mathrm{CGAN}} = \arg\min_{\mathrm{G}}\max_{\mathrm{D}} \mathop{\mathbb{E}}_{x,y\sim p(x,y)} [\log \mathrm{D}(x,y)] + \mathop{\mathbb{E}}_{\substack{y\sim p(y)\\ z\sim p(z)}} [1-\log \mathrm{D}(\mathrm{G}(y,z),y)] \tag{1.5}$$

While this approach is trivially simple to implement, it relies entirely on the discriminator to use the label. Other approaches try to learn the conditional distribution by adding an explicit loss term to the optimization problem, such as Auxillary Classifier GAN (ACGAN) (Odena, Olah, & Shlens, 2016). This approach aims to learn a conditional generative model with discrete labels by adding another output to the discriminator that acts as a classifier. The model is then trained by having both the generator and the discriminator minimize the categorical cross-entropy between the real and predicted labels.

1.2.3 Domain-transfer with GANs

Domain-transfer is the task of learning a mapping $p(x|y)$ between two high-dimensional distributions $p(x)$ and $p(y)$ that maintains semantic information, for example changing the color palette of an image while keeping the same objects at the same position. CGANs already learn to model the conditional distribution $p(x|y)$, and adding a way to enforce the consistency of the semantic information enables domain-transfer.

Pix2Pix (Isola, Zhu, Zhou, & Efros, 2016) implemented this approach explicitly by using paired samples $(x,y) \sim p(x|y)$ forcing the generator to minimize the ℓ_1 reconstruction term between x and $\mathrm{G}(y,z)$

$$\arg\min_{\mathrm{G}}\max_{\mathrm{D}} \mathrm{L}_{p2p} = \arg\min_{\mathrm{G}}\max_{\mathrm{D}} \mathrm{L}_{\mathrm{CGAN}}(\mathrm{D},\mathrm{G}) + \lambda \mathop{\mathbb{E}}_{\substack{x\sim p(x)\\ y\sim p(y)\\ z\sim p(z)}} ||x-\mathrm{G}(y,z)||_1 \ .$$

However, this kind of approaches rely on paired data which can be very hard to obtain, especially in the case of natural images. When trying for example to transfer images of zebras to images of horses, you need a dataset of very similar zebras and horses in the exact same position for the ℓ_1 term to work.

This problem of paired data was solved by CycleGAN (Zhu, Liu, Qin, & Li, 2017) using cycle-consistency. Instead of training a single model G with reconstruction between x and $\mathrm{G}(y,z)$, the CycleGAN approach train two domain-transfer models simultaneously: $\mathrm{G_{YX}}$ and $\mathrm{G_{XY}}$ that map samples from $p(y)$ onto $p(x)$ and $p(x)$ onto $p(y)$, respectively (see Figure. 1.6). This allows to compute the ℓ_1 reconstruction errors $||x-\mathrm{G_{YX}}(\mathrm{G_{XY}}(x))||_1$ and $||y-\mathrm{G_{XY}}(\mathrm{G_{YX}}(y))||_1$, thus completely removing the need for paired data (x,y). The training of the two models in done an adversarial setup,

with two discriminators D_X and D_Y, and is summed up as an optimization problem in Equation. (1.6)

$$\begin{aligned}\arg \min_{G_{XY},G_{YX}} \max_{D_X,D_Y} L_{CycGAN} = \arg \min_{G_{XY},G_{YX}} \max_{D_X,D_Y} & L_{GAN}(D_X, G_{YX}) + L_{GAN}(D_Y, G_{XY}) \\ & + \lambda \mathop{\mathbb{E}}_{x\sim p(x)} ||x - G_{YX}(G_{XY}(x)||_1 + \lambda \mathop{\mathbb{E}}_{y\sim p(y)} ||y - G_{XY}(G_{YX}(y)||_1 \ . \end{aligned} \tag{1.6}$$

The CycleGAN training process then consists in alternatively updating the two discriminator and the two generators via gradient ascent/descent. A summary of this process is presented in Algorithm. 2.

Algorithm 2 CycleGAN training algorithm

Require: $\mathcal{X}$ and $\mathcal{Y}$ two unpaired datasets, G_{XY} and G_{YX} the mapping networks, D_X and D_Y the discrimination models, m the mini-batch size

repeat

sample a mini-batch $\{x_i\}_{i=1}^m$ from $\mathcal{X}$

sample a mini-batch $\{y_i\}_{i=1}^m$ from $\mathcal{Y}$

update D_X by stochastic gradient descent of

$\sum_{i=1}^m (D_X(x_i) - 1)^2 + (D_X(G_{YX}(y_i)))^2$

update D_Y by stochastic gradient descent of

$\sum_{i=1}^m (D_Y(y_i) - 1)^2 + (D_Y(G_{XY}(x_i)))^2$

sample a mini-batch $\{x_i\}_{i=1}^m$ from X

sample a mini-batch $\{y_i\}_{i=1}^m$ from Y

update G_{XY} by stochastic gradient descent of

$\sum_{i=1}^n (D_Y(G_{XY}(x_i)) - 1)^2 + \lambda(||x_i - G_{YX}(G_{XY}(x_i))||_1$

$+||y_i - G_{XY}(G_{YX}(y_i))||_1)$

update G_{YX} by stochastic gradient descent of

$\sum_{i=1}^n (D_X(G_{YX}(y_i)) - 1)^2 + \lambda(||x_i - G_{YX}(G_{XY}(x_i))||_1$

$+||y_i - G_{XY}(G_{YX}(y_i))||_1)$

until a stopping condition is met

1.2.4 Limitations

GANs have shown strong advantages over the classical generative modeling methods, such as generating sharper samples than VAEs or taking significantly less time to train and to generate a sample than normalizing flows. They however bear limitations, namely: the instability of the training process; the loss of diversity in the generated samples (*mode-collapse*); and finally the problem of black-box conditioning.

Instability

As we have seen in the previous section, training GANs consist in solving a minimax problem. While the alternate gradient descent algorithm is a straightforward method for solving such a problem, the alternating updates can cause significant instabilities during the training process. This can result in oscillating values of the loss function which prevents convergence (Mescheder, Geiger, & Nowozin, 2018), which makes it difficult to determine a when to stop training. In the end, this behavior can be harmful in terms of performance.

CR: Figure loss GAN

The instability of the GAN training process has first been conjectured to be caused by the bad quality of the gradients obtained when G generates bad samples, which makes D strongly reject these samples and therefore saturating the loss. The first solution proposed (Goodfellow et al., 2014) was to slightly change the generator's loss function from $\log(1 - D(G(z)))$ to $-\log(D(G(z)))$,

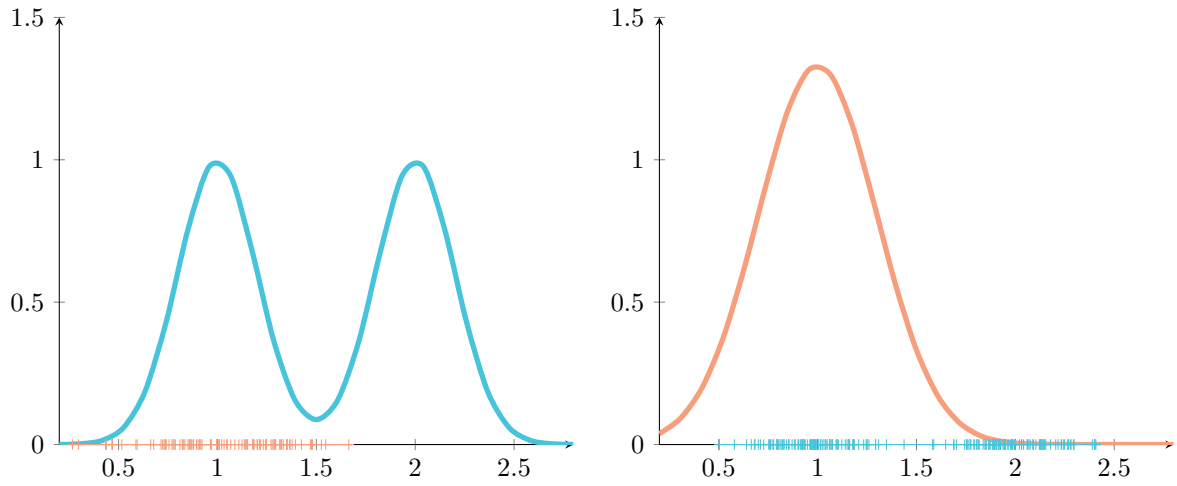


Figure 1.7: Reverse KL (left) and KL (right) divergence between the true blue distribution and the mode-collapsed orange distribution . The distance is lower in the case of the reverse KL, even if a missing mode is clearly visible.

which helped considerably in avoiding failures of the training process and was then widely used (Radford, Metz, & Chintala, 2015) CR: Plus de citations.

While this loss term converges to the same minimum as the original loss term, minimizing it no longer correspond to minimizing a JS divergence but the non-symmetric reverse KL divergence (minus a JS term) (Arjovsky & Bottou, 2017). More formally,

$$\mathop{\mathbb{E}}_{z\sim p(z)}\left[\nabla_G \log D^*(G(z))\right] = \nabla_G\left[D_{KL}\left(p_G(x)\middle\|p(x)\right) - 2D_{JS}\left(p_G(x)\middle\|p(x)\right)\right] .$$

However, albeit an empirical reduction of the instability, this loss substitution has been proved to not solve the instability problem (Arjovsky & Bottou, 2017). This is mainly due to an unstable behavior of these divergences when the real distribution and the learned one does not share the same support.

A lot of similar tricks can be applied to the training process in order to avoid this pitfall (Salimans et al., 2016; Sønderby, Caballero, Theis, Shi, & Huszár, 2017; Heusel, Ramsauer, Unterthiner, Nessler, & Hochreiter, 2017), and while more recent approaches seemed to help alleviate this issue (which will be more detailed in the next section), instability can still be observed in the most recent approaches (Brock, Donahue, & Simonyan, 2018). Even though several theory-backed techniques aimed to solve this issue (Arjovsky, Chintala, & Bottou, 2017; Nowozin, Cseke, & Tomioka, 2016; Li, Chang, Cheng, Yang, & Póczos, 2017), there are still, at the time of writing this thesis, neither clear consensus on the theoretical causes of this instability nor completely efficient solutions.

Mode collapse

Although the aforementioned change of loss can help solving the instability issues, using the reverse KL divergence is conjectured to be one of the causes of another issue: the *mode collapse* problem: different z_1, z_2 are mapped to samples $G(z_1)$ and $G(z_2)$ that are very close; and *mode dropping*: only a localized subset of the distribution can actually be mapped to, leading to missing modes in the generated samples.

Indeed, the reverse KL divergence does not penalize "missing" parts of the learned distribution $p_G(x)$, which are some points in the support of $p(x)$ that have zero (or near-zero) probability on $p_G(x)$ (see Figure. 1.7).

Another conjectured cause is raised by the alternate gradient descent, in that it does not clearly prioritize the minimax formulation $G^* = \min_G \max_D L_{GAN}$ over the maximin formulation $G^* = \max_D \min_G L_{GAN}$, which does not behave in the fashion as it pushes the generator towards mapping every z to the single most probable x, evaluated by the generator (Goodfellow, 2016).

In the same fashion as the instability problem, there is at the time of writing this thesis no fundamental explanation to this issue. However, it still raise a first trade-off: since using the original GAN creates instability which lead to a drop of visual quality, and using the non-saturating variant creates a lack of diversity. This extends to more recent approaches in which higher visual quality induces a loss of diversity (Brock, Donahue, & Simonyan, 2018).

In the most extreme cases, this loss of diversity can result in a complete collapsing of the sampling mechanism, making it completely impossible to draw diverse samples. However this is not as much of an issue for conditional tasks that consists in mapping an input to one of many acceptable outputs, with one example of such a task being the aforementioned domain-transfer (see Section. 1.2.3).

Black-box conditioning

CR:

Instability, catastrophic forgetting and the mode collapse problem

Trade-off image quality/diversity : Explanation through the loss term and distribution coverage

Black-box approach to conditioning, no tuning possible, no interpretability

1.2.5 A note on the evaluation of generative models

Unlike discriminative models, evaluating and comparing GAN approaches is a non-trivial task. Two approaches can be envisioned: evaluating the *intrinsic* quality of generated samples with ad-hoc criterions or directly evaluating the likelihood of the generated samples. However, unlike VAEs and flow-based models, GANs offer no explicit way to evaluate or approximate the likelihood of the generated samples. Thus, a significant part of the GAN literature resorted to a subjective visual evaluation of the generated samples.

In order to provide a more precise evaluation of the visual quality of generated samples, two ad-hoc methods Inception Score (IS) (Salimans et al., 2016) and Fréchet Inception Distance (FID) (Heusel, Ramsauer, Unterthiner, Nessler, & Hochreiter, 2017) were proposed, which both make use of a pre-trained Inception v3 model (Szegedy, Vanhoucke, Ioffe, Shlens, & Wojna, 2016), a deep classifier trained on the ImageNet dataset (Deng et al., 2009).

Inception Score (IS)) (Salimans et al., 2016) is based on the evaluation of the entropy of the labels y predicted by the Inception classifier of generated data. High-fidelity samples should be easier to classify and therefore have a conditional label distribution $p_{\mathrm{G}}(y|x)$ with low entropy. In addition to the high quality, the samples should be diverse, therefore the marginal distribution $p_{\mathrm{G}}(y) = \int_{\mathcal{Z}} p_{\mathrm{G}}(y|x = \mathrm{G}(z))dz$ should have a high entropy. By combining these two requirements, the IS is formulated as

$$\mathrm{IS}(y) = \exp\left[\mathop{\mathbb{E}}_{x\sim p_{\mathrm{G}}(x)} \mathrm{D}_{\mathrm{KL}}\Big(p_{\mathrm{G}}(y|x)\Big|\Big|p_{\mathrm{G}}(y)\Big)\right] \ .$$

Although it has been widely used, IS has shown major issues (Barratt & Sharma, 2018) that raise from the use of the conditional label distribution. Most notably, examples that are correctly classified are not necessarily of the highest quality and the pre-determined label classes can skew the estimation of the marginal distribution $p_{\mathrm{G}}(y)$.

The **Fréchet Inception Distance (FID)** (Heusel, Ramsauer, Unterthiner, Nessler, & Hochreiter, 2017) differs from IS since it evaluates a distance between the distributions of visual features computed on real and generated data, instead of relying on the labels. These features are extracted at the penultimate layer of the Inception classifier. The distributions of these features are assumed Gaussian, so that the Fréchet distance (or Wasserstein-2 distance) can be computed as

$$\mathrm{FID} = ||\mu - \mu_{\mathrm{G}}||^2 + \mathrm{Tr}(\Sigma + \Sigma_{\mathrm{G}} - 2\sqrt{\Sigma * \Sigma_{\mathrm{G}}}) \ ,$$

where $\mathcal{N}(\mu, \Sigma)$ and $\mathcal{N}(\mu_G, \Sigma_G)$ are the distributions of the extracted features of the real and generated data, respectively. FID is considered more robust than IS and has been either completing or replacing the use of IS in recent works.

However, while these two metrics are considered to be the standard method for evaluating GANs, their reliance on the pre-trained Inception model can prove to be an issue. Indeed, they behave well when used to compare models learned on natural images datasets such as ImageNet, but they cannot directly be applied to other datasets. A solution to consider can be the training of another classifier network on a more adapted dataset, but this solution cannot be applied when no labeled data is available.

For completeness, we can also refer to less used metrics for evaluating visual quality: The Parzen window (or kernel density estimation) (Parzen, 1962) aim to estimate the likelihood of the generated samples; the Sliced Wasserstein Distance (Julien et al., 2011) is an efficient approximation of the Earth-Mover (or Wasserstein) distance; the Kernel Inception Distance (Bińkowski, Sutherland, Arbel, & Gretton, 2018) is a recent metric that evaluates the maximum mean discrepancy between Inception features with a polynomial kernel.

Finally it is to note that for conditioned models, evaluating the aforementioned metrics does not inform about the quality of the conditioning. However, since the conditioning usually requires either labels or prior information, these can often be evaluated by, for example, predicting the labels of generated samples with a pre-trained classifier and computing the error between the predicted label and the original one.

1.3 The GAN Zoo

1.3.1 A taxonomy of GANs

Enorme nombre de variantes de GANs

Taxonomie des approches GANs (pour s'éviter une liste des différents GANs)

Schéma pour définir les grandes familles de GAN (évoquer les AmbientGAN / UNIR)

1.3.2 Architecture variants

1.3.3 Divergence variants

Table des loss alternatives (f-divergences + transport optimal)

1.3.4 Task-specific losses

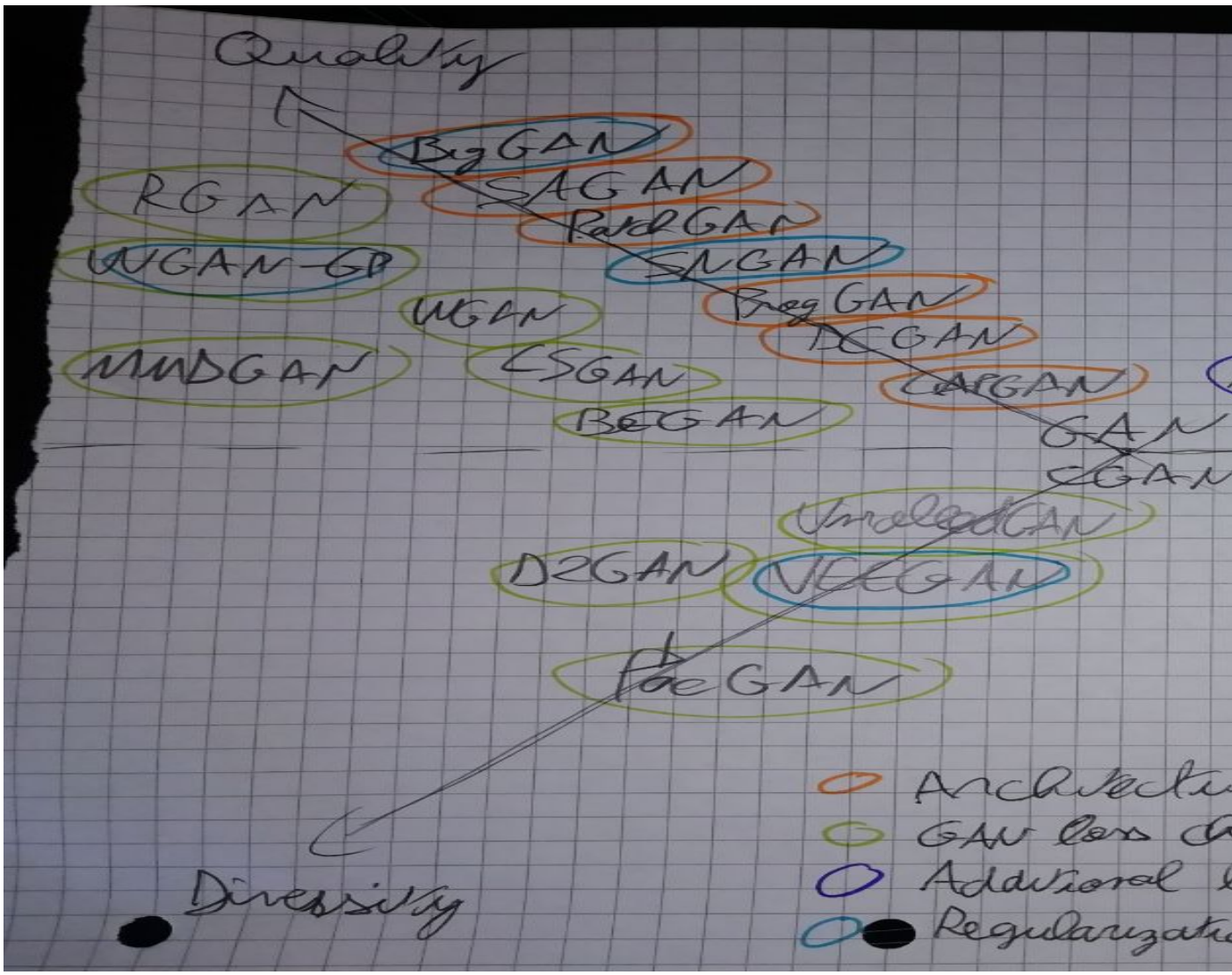


Figure 1.8: Classifications of some advances in GANs on the trilemma

Chapter 2

Reconstruction as an Auxiliary Task for Generative Modeling

Chapter abstract

In this chapter, we propose an approach for conditioning a GAN model to reconstuct images from a very sparse set of randomly-positioned pixels known beforehand. This approach, based on a Maximum A Posteriori estimation, takes the form of an explicit auxiliary reconstruction task which adds to the GAN objective as an additional loss term. Complemented with the PacGAN variant for training GANs, this approach enables the generation of diverse samples from a sparse pixel map. As opposed to the more classical Conditional GAN approach, this auxiliary task is interpretable and a hyperparameter allows to control the importance of the conditioning in the learning process. We evaluate our approach on the classical MNIST, FashionMNIST and CIFAR10 datasets, as well as a custom-made texture dataset. Finally, we apply this approach to a task of geostatistical simulation.

Contents

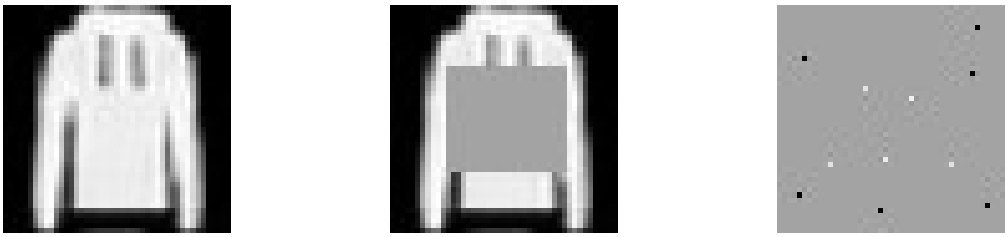
2.1 Image Reconstruction with Generative Models

Image reconstruction is the task of completing an image from a very small subset of the pixels. Such source data can usually be found in domains where the measurement process is very noisy or where measurements are expensive. This task differs from image inpainting since the source data is usually unstructured and very scarce, as in this chapter we will consider randomly scattered measurements of less than a percent of the image. While our discussion focus on image reconstruction, it is noteworthy to mention that this applies to other kinds of signals.

The task of image reconstruction is challenging since very few information is available for use. To overcome this lack of information, generative models such as GANs leverage on existing datasets to learn the distribution of the real images. By conditioning the learned distribution, a GAN could learn to generate an image while enforcing the constraint that the pixels known beforehand must remain similar in the generated image.

Similarly as in the GAN setup, we denote by $\mathrm{X} \in \mathscr{X}$ a random variable and x its realization. Let p_{X} be the distribution of X over $\mathscr{X}$ and $p_{\mathrm{X}}(x)$ be its evaluation at x. Similarly $p_{\mathrm{X}|\mathrm{Y}}$ represents the distribution of X conditioned on the random variable $\mathrm{Y} \in \mathscr{Y}$.

We denote by $x \in \mathscr{X} = [-1,1]^{n \times p \times c}$ (see Figure 2.1a) an image sampled from an unknown distribution p_{X} and a sparse matrix $y \in \mathscr{Y} = [-1,1]^{n \times p \times c}$ (Figure 2.1c) as the given constrained pixels.



(a) Original Image (b) Inpainting Input (c) Constraint Map

Figure 2.1: Difference between regular inpainting (2.1b) and the problem undertaken in this work (2.1c) on a real sample (2.1a).

(a) Original Image (b) Constraints (c) Generated Image (d) Satisfied Consts.

Figure 2.2: Generation of a sample during training. We first sample an image from a training set (2.2a) and we sample the constraints (2.2b) from it. Then our GAN generates a sample (2.2c). The constraints with squared error smaller than $\epsilon = 0.1$ are deemed satisfied and shown by green pixels in (2.2d) while the red pixels are unsatisfied.

The problem then consists in finding a generative model G with inputs z (a random vector sampled from a known distribution p_Z over the space $\mathcal{Z}$) and constrained pixel values $y \in [-1,1]^{n \times p \times c}$ that maps the distribution p_Z onto the conditional distribution $p_{X|Y}$ of the real images given the constraints y (see Figure 2.2).

CR: Related works : CGAN, AmbientGAN, UNIR, Compressed Sensing with Meta-Learning
Limitations of these models

2.2 Conditional generation as a Maximum A Posteriori estimation

Approche de l'article NeuCom :

Formulation as a Maximum A Posteriori Estimation, assumptions (normal error)
Construction of the loss term using bayes rule and least squares
PacGAN for keeping the diversity

2.3 Experimental evaluation and application to underground soil generation

Datasets : MNIST/FashionMNIST/CelebA/Texture

Evaluation : MSE/FID; Epoch selection criterion
Architectures : Appendix ? DCGAN + SGAN (encoder-decoder)
Results : visible trade-off, good fidelity overall
Application to hydro-geology : subsurface dataset
Evaluation : MSE/HOG+LBP

2.4 Conclusion

Objective reached : tuneable loss, pixel-wise, keeping diversity

Applications in hydro-geology : papier Eric

Future works : other distributions (modelling error using Laplacian, beta or Poisson distributions)

Chapter 3

Conditioning generation with multiple task-specific constraints

Chapter abstract

content...

Contents

3.1 Introduction

Formulation as a constrained optimization problem
- Reformulation of CycleGAN as a constrained optimization problem
- Relaxation of the constraints
- Ici, expérimenter sur des datasets artificiels ?

3.2 Proximal method for non-Euclidean output space

Travail sur le proximal ?
- Envelope theorem application

3.3 Application to RGB to Polarimetric domain transfer

Introduction to polarimetry-specific physical constraints (briefly, no need to write a physics essay)
- Reformulation as constraints on the output space
- Relaxations : L_2 term + rectified term
- Dataset, Evaluation
- Experiments and results

3.4 Conclusion

Relaxation of the constraints works even when a lot of constraints are applied
- The application to the polarimetric dataset works
- Future works : using adapted metrics for the non-euclidean outspace

Chapter 4

Conclusion and Perspectives

Bibliography

Antipov, Grigory, Moez Baccouche, and Jean-Luc Dugelay (May 30, 2017). "Face Aging With Conditional Generative Adversarial Networks". In: arXiv: 1702.01983 [cs]. URL: http://arxiv.org/abs/1702.01983 (visited on 05/19/2020) (cit. on p. 1).

Arjovsky, Martin and Léon Bottou (2017). "Towards Principled Methods for Training Generative Adversarial Networks". In: URL: https://arxiv.org/pdf/1701.04862.pdf (cit. on p. 11).

Arjovsky, Martin, Soumith Chintala, and Léon Bottou (2017). "Wasserstein GAN". In: URL: https://arxiv.org/pdf/1701.07875.pdf (cit. on p. 11).

Barratt, Shane and Rishi Sharma (2018). *A Note on the Inception Score*. URL: https://github.com/ (cit. on p. 12).

Bińkowski, Miko\laj et al. (Jan. 2018). "Demystifying MMD GANs". In: URL: http://arxiv.org/abs/1801.01401 (cit. on p. 13).

Brock, Andrew, Jeff Donahue, and Karen Simonyan (Sept. 2018). "Large Scale GAN Training for High Fidelity Natural Image Synthesis". In: URL: http://arxiv.org/abs/1809.11096 (cit. on pp. 11, 12).

Dempster, A. P., N. M. Laird, and D. B. Rubin (Sept. 1977). "Maximum Likelihood from Incomplete Data Via the EM Algorithm". In: *Journal of the Royal Statistical Society: Series B (Methodological)* 39.1, pp. 1–22. DOI: 10.1111/j.2517-6161.1977.tb01600.x. URL: http://doi.wiley.com/10.1111/j.2517-6161.1977.tb01600.x (cit. on p. 5).

Deng, Jia et al. (2009). *ImageNet: A Large-Scale Hierarchical Image Database*. URL: http://www.image-net.org. (cit. on p. 12).

Dinh, Laurent, Jascha Sohl-Dickstein, and Samy Bengio (Feb. 27, 2017). "Density Estimation Using Real NVP". In: arXiv: 1605.08803 [cs, stat]. URL: http://arxiv.org/abs/1605.08803 (visited on 05/11/2020) (cit. on pp. 4, 6).

Goodfellow, Ian (2016). "NIPS 2016 Tutorial: Generative Adversarial Networks". In: URL: https://arxiv.org/pdf/1701.00160.pdf (cit. on p. 11).

Goodfellow, Ian J et al. (2014). "Generative Adversarial Nets". In: URL: https://arxiv.org/pdf/1406.2661.pdf (cit. on pp. 1, 3, 4, 6–8, 10).

Heusel, Martin et al. (2017). "GANs Trained by a Two Time-Scale Update Rule Converge to a Local Nash Equilibrium". In: URL: https://arxiv.org/pdf/1706.08500.pdf (cit. on pp. 11, 12).

Isola, Phillip et al. (2016). "Image-to-Image Translation with Conditional Adversarial Networks". In: URL: https://arxiv.org/pdf/1611.07004v1.pdf (cit. on p. 9).

Julien, Rabin et al. (2011). *Wasserstein Barycenter and Its Application to Texture Mixing*, pp. 435–446. URL: https://hal.archives-ouvertes.fr/hal-00476064 (cit. on p. 13).

Kang, Yuhao, Song Gao, and Robert E. Roth (May 4, 2019). "Transferring Multiscale Map Styles Using Generative Adversarial Networks". In: *International Journal of Cartography* 5.2-3, pp. 115–141. ISSN: 2372-9333, 2372-9341. DOI: 10.1080/23729333.2019.1615729. URL: https://www.tandfonline.com/doi/full/10.1080/23729333.2019.1615729 (visited on 05/19/2020) (cit. on p. 1).

Kingma, Diederik P and Max Welling (2014). "Auto-Encoding Variational Bayes". In: URL: https://arxiv.org/pdf/1312.6114.pdf (cit. on pp. 4, 5).

Kingma, Durk P. and Prafulla Dhariwal (2018). *Glow: Generative Flow with Invertible 1x1 Convolutions*. 10236–10245. URL: http://papers.nips.cc/paper/8224-glow-generative-flow-with-invertible-1x1-convolutions (cit. on pp. 4, 6).

Li, Chun-Liang et al. (Nov. 27, 2017). "MMD GAN: Towards Deeper Understanding of Moment Matching Network". In: arXiv: 1705.08584 [cs, stat]. URL: http://arxiv.org/abs/1705.08584 (visited on 05/19/2020) (cit. on p. 11).

Mescheder, Lars, Andreas Geiger, and Sebastian Nowozin (2018). "Which Training Methods for GANs Do Actually Converge?" In: URL: http://proceedings.mlr.press/v80/mescheder18a/mescheder18a.pdf (cit. on p. 10).

Mirza, Mehdi and Simon Osindero (2014). "Conditional Generative Adversarial Nets". In: URL: https://arxiv.org/pdf/1411.1784.pdf (cit. on p. 8).

Nowozin, Sebastian, Botond Cseke, and Ryota Tomioka (2016). *F-GAN: Training Generative Neural Samplers Using Variational Divergence Minimization*. URL: https://arxiv.org/pdf/1606.00709.pdf (cit. on p. 11).

Odena, Augustus, Christopher Olah, and Jonathon Shlens (2016). "Conditional Image Synthesis with Auxiliary Classifier GANs". In: URL: https://arxiv.org/pdf/1610.09585.pdf (cit. on p. 9).

Parzen, Emanuel (1962). "On Estimation of a Probability Density Function and Mode". In: *Annals of Mathematical Statistics* 33.3, pp. 1065–1076. ISSN: 0003-4851. DOI: 10.1214/AOMS/1177704472 (cit. on p. 13).

Radford, Alec, Luke Metz, and Soumith Chintala (Nov. 2015). "Unsupervised Representation Learning with Deep Convolutional Generative Adversarial Networks". In: URL: http://arxiv.org/abs/1511.06434 (cit. on p. 11).

Salimans, Tim et al. (June 2016). "Improved Techniques for Training GANs". In: URL: http://papers.nips.cc/paper/6125-improved-techniques-for-training-gans.pdf%20http://arxiv.org/abs/1606.03498 (cit. on pp. 11, 12).

Sønderby, Casper Kaae et al. (Feb. 21, 2017). "Amortised MAP Inference for Image Super-Resolution". In: arXiv: 1610.04490 [cs, stat]. URL: http://arxiv.org/abs/1610.04490 (visited on 05/19/2020) (cit. on p. 11).

Szegedy, Christian et al. (Dec. 2016). "Rethinking the Inception Architecture for Computer Vision". In: *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*. Vol. 2016-Decem. _eprint: 1512.00567. IEEE Computer Society, pp. 2818–2826. ISBN: 978-1-4673-8850-4. DOI: 10.1109/CVPR.2016.308. URL: http://arxiv.org/abs/1512.00567 (cit. on p. 12).

Vaccari, Cristian and Andrew Chadwick (2020). "Deepfakes and Disinformation: Exploring the Impact of Synthetic Political Video on Deception, Uncertainty, and Trust in News". In: *Social Media and Society* 6.1. ISSN: 20563051. DOI: 10.1177/2056305120903408 (cit. on p. 1).

Vondrick, Carl, Hamed Pirsiavash, and Antonio Torralba (Oct. 26, 2016). "Generating Videos with Scene Dynamics". In: arXiv: 1609.02612 [cs]. URL: http://arxiv.org/abs/1609.02612 (visited on 05/19/2020) (cit. on p. 1).

Wang, Zhihao, Jian Chen, and Steven C.H. Hoi (2020). "Deep Learning for Image Super-Resolution: A Survey". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pp. 1–1. ISSN: 1939-3539. DOI: 10.1109/TPAMI.2020.2982166 (cit. on p. 1).

Wu, Jiajun et al. (Jan. 4, 2017). "Learning a Probabilistic Latent Space of Object Shapes via 3D Generative-Adversarial Modeling". In: arXiv: 1610.07584 [cs]. URL: http://arxiv.org/abs/1610.07584 (visited on 05/19/2020) (cit. on p. 1).

Zhu, Xinyue et al. (2017). *Emotion Classification with Data Augmentation Using Generative Adversarial Networks*. URL: https://arxiv.org/pdf/1711.00648.pdf (cit. on p. 9).

Appendix A

Publications

Appendix B

Experiment details for the Pixel-Wise Conditionned GAN

Appendix C

Experiment details for the Polarimetric CycleGAN