Introduction to generative modelling
Session 1: VAEs, NFs and GANs

Émile Mathieu, University of Cambridge

November 18, 2022

Outline of the course

- Goal of the course:
  - Introduce modern methods of generative modelling.
  - Present their strengths and limitations.
  - Draw connections between class of models.
- Outline of 1st session:
  - Variational Autoencoders (VAEs).
  - Normalising flows (NFs).
  - Generative Adversarial Networks (GANs).
- Outline of 2n session:
  - Energy-based models (EBMs).
  - Score matching (SM).
  - Discrete diffusion models.
  - Continuous diffusion models.

We acknowledge Valentin De Bortoli's for some material from his generative modelling course.

Introduction to generative modelling

Definition

- **Generative modelling**: Given a distribution $\pi \in \mathcal{P}(\mathbb{R}^d)$ how to obtain sample from π?
  - Access to $\hat{\pi} = (1/N)\sum_{k=1}^{N} \delta_{x^k}$, the **empirical distribution**.
  - $\{x^k\}_{k=1}^{N}$ are samples from π
- A general approach:
  - Start from an **easy-to-sample** distribution $\pi_0 \in \mathcal{P}(\mathbb{R}^p)$ $(p \neq d)$.
  - Choose a noise distribution π_{Z} on a space $(\mathsf{Z}, \mathcal{Z})$.
  - Define a mapping $g : \mathbb{R}^p \times \mathsf{Z} \to \mathbb{R}^d$ such that $g_{\#}(\pi_0, \pi_{\mathsf{Z}}) \approx \pi$.
- In other words:
  - Sample Z from π_{Z}, sample X_0 from π_0
  - Push with $g(X_0, Z) \to$ approximate sample from π.

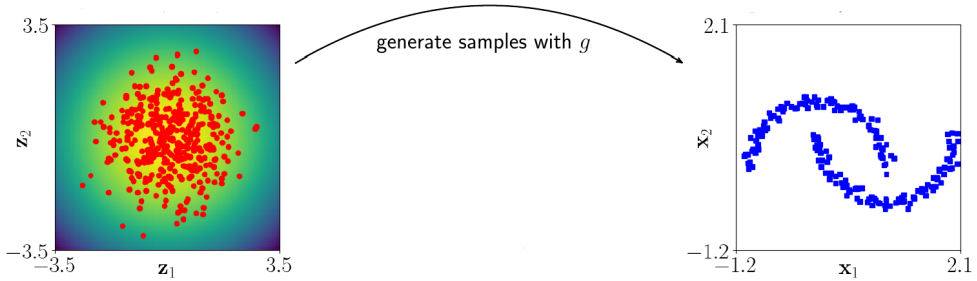


Figure 1: Image adapted from (**ruthotto2021introduction**).

Application (1/2): Nowcasting

- Application in meteorology: (ravuri2021skilful).
  - Prediction of rain in the next 2 hours: nowcasting.
  - Solving physical PDEs: planet scale predictions days ahead.
  - Struggle for high resolution predictions on short time ranges.
- Access to a lot of high quality data: conditional GAN.

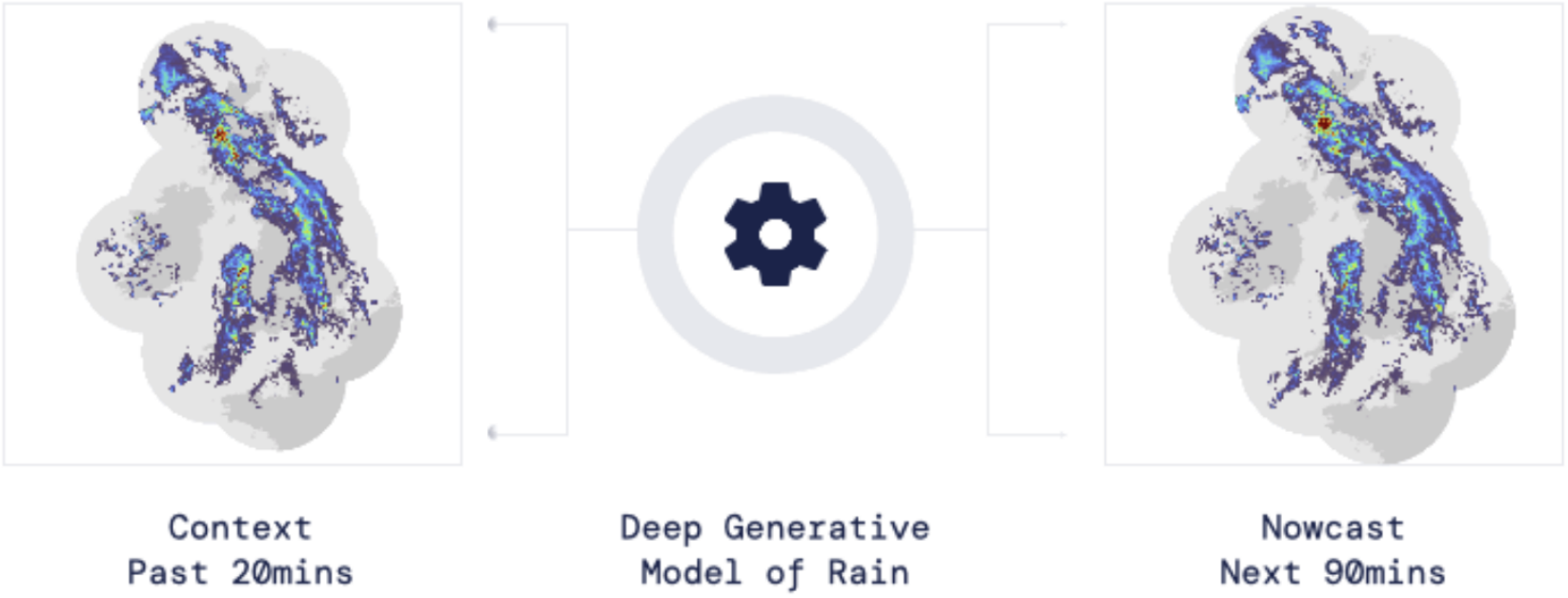


Figure 2: Image extracted from (ravuri2021skilful).

Application (2/2): Protein Folding

- Application in computational biology: (**senior2020improved**).
  - Amino-acid sequence to 3D structure.
  - Cryo-Electron Microscopy or crystallography = experimental techniques to determine the shape of the protein.
  - Crystallizing a protein is a real challenge (**avanzato2019structural**).
  - Competition to predict structure: Critical Assessment of protein Structure Prediction.
- Conditional generative modelling.

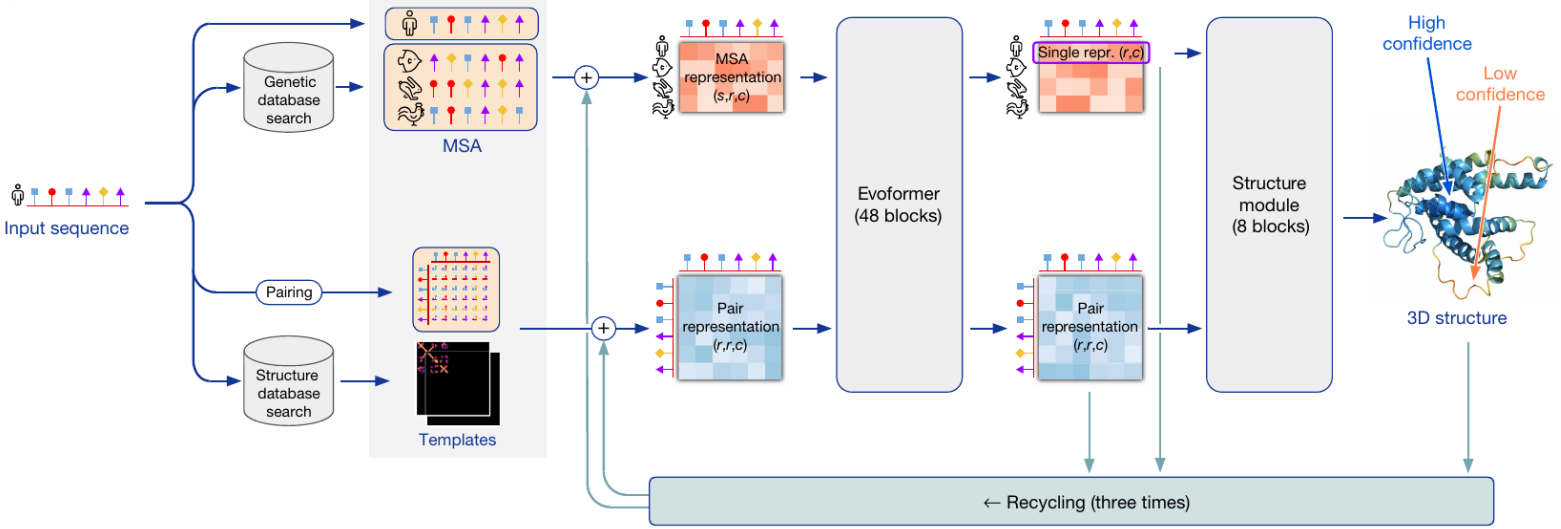


Figure 3: Image extracted from (**senior2020improved**).

Difference with statistical sampling

- Generative Modelling: sampling of target distribution π and we have access to $\hat{\pi} = (1/N)\sum_{k=1}^{N} \delta_{x^k}$, the empirical distribution.
- Different setting than statistical sampling.
  - Sampling from π with density (w.r.t. the Lebesgue measure on $\mathbb{R}^d$) proportional to $x \mapsto \exp[-U(x)]$.
  - $U : \mathbb{R}^d \to \mathbb{R}$ is called an (energy) potential.
  - Classical methods: Monte Carlo Markov Chains (**roberts1996exponential**; **durmus2017fast**; **dalalyan2017further**).
  - Applications in statistical physics and Bayesian statistics, see e.g. (**neal1992bayesian**).
- Interaction between statistical sampling and generative modelling:
  - Modification of GAN losses to design efficient Markov kernels with given invariant measure (**song2017nice**).
  - Use of Metropolis-Hastings rejection and discriminator step to improve generative modelling with GAN (**turner2019metropolis**).

Variational Autoencoders

Introduction of the ELBO

From log-likelihood to ELBO (1/2)

- Manifold hypothesis: the data distribution is supported on a space (submanifold) $\mathbb{R}^p \subsetneq \mathbb{R}^d$ with much lower dimension than $\mathbb{R}^d$.
  - We define a joint model on $\mathbb{R}^d \times \mathbb{R}^p$ and assume that $p_\theta(x) = \int_{\mathbb{R}^p} \underbrace{p_\theta(x,z)}_{\text{joint}} \mathrm{d}z = \int_{\mathbb{R}^p} \underbrace{p(z)}_{\text{prior}} \underbrace{p_\theta(x|z)}_{\text{decoder}} \mathrm{d}z$.
  - $p(z)$ is called the **prior** distribution (and does not depend on θ).
  - The distribution $p_\theta(x|z)$ decodes the latent vector z.
- Evaluating the log-likelihood is a marginalisation problem.

$$\log p_\theta(x) = \log\left(\int_{\mathbb{R}^p} p(z)\, p_\theta(x|z)\, \mathrm{d}z\right).$$

From log-likelihood to ELBO (2/2)

- Instead of directly maximising the log-likelihood we are going to consider a lower-bound.

$$
\begin{aligned}
\log(p_\theta(x)) &= \log\Big(\int_{\mathbb{R}^p} p_\theta(x|z)\; p(z)\, \mathrm{d}z\Big) \\
&= \log\Big(\int_{\mathbb{R}^p} p_\theta(x|z)\; \big(\, p(z) \,/\, q(z)\,\big)\; q(z)\, \mathrm{d}z\Big) \\
&\geq \int_{\mathbb{R}^p} \log\big(\, p_\theta(x|z)\; p(z) \,/\, q(z)\,\big)\; q(z)\, \mathrm{d}z \\
&\geq \int_{\mathbb{R}^p} \log\big(\, p_\theta(x|z)\,\big)\; q(z)\, \mathrm{d}z - \mathrm{KL}\big(\, q \,\big|\, p\,\big) \triangleq \mathcal{L}(\theta)\ .
\end{aligned}
$$

- Inequality obtained using the concavity of the logarithm.
- This last lower-bound is called the ELBO (Evidence Lower BOund) (mackay1992bayesian):
  - The first term controls the reconstruction.
  - The second term controls how close q is to the prior.
- The choice of the variational distribution q is crucial

Expectation-Maximisation (EM) Algorithm

- Before presenting the VAE setting we recall the basics of the Expectation-Maximisation (EM) algorithm.
- We begin with the same ELBO

$$\log(p_\theta(x)) \geq \int_{\mathbb{R}^p} \log(\, p_\theta(x|z)\,)\, q(z)\, \mathrm{d}z - \mathrm{KL}(\, q \,|\, p\,) \triangleq \mathcal{L}(\theta, q)\ .$$

- We consider the following procedure:
- Expectation: Compute posterior $q = \arg\max_q \mathcal{L}(\theta, q) = p_{\theta_k}(\cdot|x)$ and

$$\begin{aligned}\mathcal{L}(\theta, q = p_{\theta_k}(\cdot|x)) &= \int_{\mathbb{R}^p} \log(\, p_\theta(x|z)\,)\, p_{\theta_k}(z|x)\, \mathrm{d}z - \mathrm{KL}(p_{\theta_k}(\cdot|x)|\, p\,) \\ &= \int_{\mathbb{R}^p} \log(\, p_\theta(x,z)\,)\, p_{\theta_k}(z|x)\, \mathrm{d}z + \mathrm{H}(p_{\theta_k}(\cdot|x))\ .\end{aligned}$$

- Maximisation: $\theta^{k+1} = \arg\max_\theta \mathcal{L}(\theta, q = p_{\theta_k}(\cdot|x))$.
- Go back to the first step.
- Computing the expectation might be hard (useful for mixture of Gaussians models for instance).

Encoding families and reparameterisation trick

Encoding variational family

- The variational distribution $q(z) = p_\theta(z|x)$ is optimal.

$$\log(p_\theta(x)) - \mathcal{L}(\theta) = \log(p_\theta(x)) - \int_{\mathbb{R}^d} \log\left(p_\theta(x,z) \,/\, q(z) \right) q(z)\, \mathrm{d}z$$
$$= -\int_{\mathbb{R}^d} \log\left(p_\theta(z|x) \,/\, q(z) \right) q(z)\, \mathrm{d}z = \mathrm{KL}(p_\theta(\cdot|x) \,|\, q) \,.$$

- Hence choosing the posterior closes the variational gap.
- Unfortunately the posterior can be hard to compute.
- In VAEs we consider a variational family of distribution $z \mapsto q_\phi(z|x)$:
  - ϕ is a parameter of q_ϕ (parametric family).
  - q_ϕ transforms a data point into a latent representation.
  - The ELBO can be written as follows

$$\boxed{\mathcal{L}(\theta,\phi) = \int_{\mathbb{R}^p} \log\left(p_\theta(x|z) \right) q_\phi(z|x)\, \mathrm{d}z - \mathrm{KL}\left(q_\phi(\cdot|x) \,|\, p \right) .}$$

Computing the gradient

- The ELBO can be written as follows

$$\mathcal{L}(\theta, \phi) = \int_{\mathbb{R}^p} \log(\, p_\theta(x|z)\,)\, q_\phi(z|x)\, \mathrm{d}z - \mathrm{KL}(\, q_\phi(\cdot|x)\, |p)\, .$$

- The goal is to optimise jointly w.r.t. θ and ϕ.

- Taking the gradient w.r.t. θ

$$\nabla_\theta \mathcal{L}(\theta, \phi) = \int_{\mathbb{R}^p} \nabla_\theta \log(\, p_\theta(x|z)\,)\, q_\phi(z|x)\, \mathrm{d}z\, .$$

- Taking the gradient w.r.t. ϕ

$$\nabla_\phi \mathcal{L}(\theta, \phi) = -\int_{\mathbb{R}^p} \nabla_\phi \log(\, q_\phi(z|x)\,)\, q_\phi(z|x)\, \mathrm{d}z$$
$$+ \int_{\mathbb{R}^d} \log(\, p_\theta(x, z) \,/\, q_\phi(z|x)\,) \nabla_\phi \log(\, q_\phi(z|x)\,)\, q_\phi(z|x)\, \mathrm{d}z\, .$$

- Using Monte Carlo estimators we can approximate these integrals.
  - With the reparameterisation trick (**kingma2013auto**) we will see a simpler way to compute these quantities.

Reparameterisation trick

- Recall the ELBO

$$\mathcal{L}(\theta, \phi) = \int_{\mathbb{R}^p} \log(\, p_\theta(x|z)\,)\, q_\phi(z|x)\, \mathrm{d}z - \mathrm{KL}(\, q_\phi(\cdot|x) \,|\, p\,)\,.$$

- The reparameterisation trick (**kingma2013auto**) consists into decoupling the randomness and the parameters.
  - Sampling from $Z \sim q_\phi(\cdot|x)$ as $Z = g_\phi(x, \varepsilon)$ where $\varepsilon \sim q$.
  - q does not depend on any parameter θ or ϕ.
  - In the Gaussian setting $Z = m_\phi(z) + \Sigma_\phi^{1/2}(z)\varepsilon$ with $\varepsilon \sim \mathrm{N}(0, \mathrm{Id})$.
- We can rewrite the ELBO as follows

$$\begin{aligned}\mathcal{L}(\theta, \phi) &= \int_{\mathbb{R}^p} \log(\, p_\theta(x|z)\,)\, q_\phi(z|x)\, \mathrm{d}z - \mathrm{KL}(\, q_\phi(\cdot|x) \,|\, p\,) \\ &= \mathbb{E}_{\varepsilon\sim q}\left[\log(\, p_\theta(x|g_\phi(x,\varepsilon))\,) - \log(\, q_\phi(g_\phi(x,\varepsilon)\,|x)/\, p(g_\phi(x,\varepsilon))\,)\right]\,.\end{aligned}$$

  - Change of variable $z = g_\varphi(x, \varepsilon)$.
- We can compute the integrand and differentiate them w.r.t. θ and ϕ.
- Note that the integrals *do not* depend on the parameters.

Gradient estimators

- Rewriting the ELBO:

$$\mathcal{L}(\theta,\phi) = \int_{\mathbb{R}^p} \{\log(\, p_\theta(g_\phi(x,\varepsilon),x)\,) - \log\, q_\phi(g_\phi(x,\varepsilon)|x)\,\} q(\varepsilon)\mathrm{d}\varepsilon\ .$$

- Different gradient estimators of the ELBO (**roeder2017sticking**).

$$\nabla_\phi \mathcal{L}(\theta,\phi) = \int_{\mathbb{R}^p} \nabla_z \{\log(\, p_\theta(g_\phi(x,\varepsilon),x)\,) - \log\, q_\phi(g_\phi(x,\varepsilon)|x)\,\} \nabla_\phi g_\phi(x,\varepsilon) q(\varepsilon$$

$$- \int_{\mathbb{R}^p} \nabla_\phi \log\, q_\phi(g_\phi(x,\varepsilon)|x)\, q(\varepsilon)\mathrm{d}\varepsilon\ .$$

- Note that $\int_{\mathbb{R}^p} \nabla_\phi \log\, q_\phi(g_\phi(x,\varepsilon)|x)\, q(\varepsilon)\mathrm{d}\varepsilon = 0$.
- $\{\varepsilon^k\}_{k=1}^N$ i.i.d. samples from q. Two unbiased estimators (and link with control variates):
  - Path derivative estimator

$$\hat{\nabla}_\phi^{\mathrm{PD}} \mathcal{L}(\theta,\phi) = \sum_k^N \nabla_z \{\log(\, p_\theta(g_\phi(x,\varepsilon^k),x)\,) - \log\, q_\phi(g_\phi(x,\varepsilon^k)|x)\,\} \nabla_\phi g_\phi(x$$

  - Total derivative estimator

$$\hat{\nabla}_\phi^{\mathrm{TD}} \mathcal{L}(\theta,\phi) = \hat{\nabla}_\phi^{\mathrm{PD}} \mathcal{L}(\theta,\phi) - \sum_{k=1}^N \nabla_\phi \log\, q_\phi(g_\phi(x,\varepsilon^k)|x)\ .$$

Gaussian case and training

Interpretation in the Gaussian case

- The ELBO is given by

$$\mathcal{L}(\theta,\phi) = \int_{\mathbb{R}^p} \log(\, p_\theta(x|g_\phi(x,\varepsilon))\,) q(\varepsilon)\mathrm{d}\varepsilon$$
$$- \int_{\mathbb{R}^p} \log(\, q_\phi(g_\phi(x,\varepsilon)|x) \,/\, p(g_\phi(x,\varepsilon))\,) q(\varepsilon)\mathrm{d}\varepsilon\ .$$

- Recall that in practice, we restrict ourselves to the Gaussian case:
  - $p_\theta(x|z) = \mathrm{N}(m_\theta(z), \Sigma_\theta^{1/2}(z))$.
  - $q_\phi(z|x) = \mathrm{N}(m_\phi(x), \Sigma_\phi^{1/2}(x))$.
- For simplicity assume that $\Sigma_\theta^{1/2} = \Sigma_\phi^{1/2} = \mathrm{Id}$.

$$\mathcal{L}(\theta,\phi) = -(1/2)\int_{\mathbb{R}^p} \|x - m_\theta(m_\phi(x)+\varepsilon)\|^2 q(\varepsilon)\mathrm{d}\varepsilon$$
$$-(1/2)\int_{\mathbb{R}^p} \|m_\phi(x)+\varepsilon\|^2 q(\varepsilon)\mathrm{d}\varepsilon + C\ .$$

  - C is a constant independent of the parameters .
  - The first term is the reconstruction loss.
  - The second term is the regularisation term.

Influence of loss terms

- MNIST reconstruction with VAE (10 digits give 10 classes).
  - Minimising only the reconstruction loss does not yield meaningful interpolation (sampling is hard).
  - Minimising only the regularisation loss does not yield meaningful encoding.
- The latent space is two dimensional here.

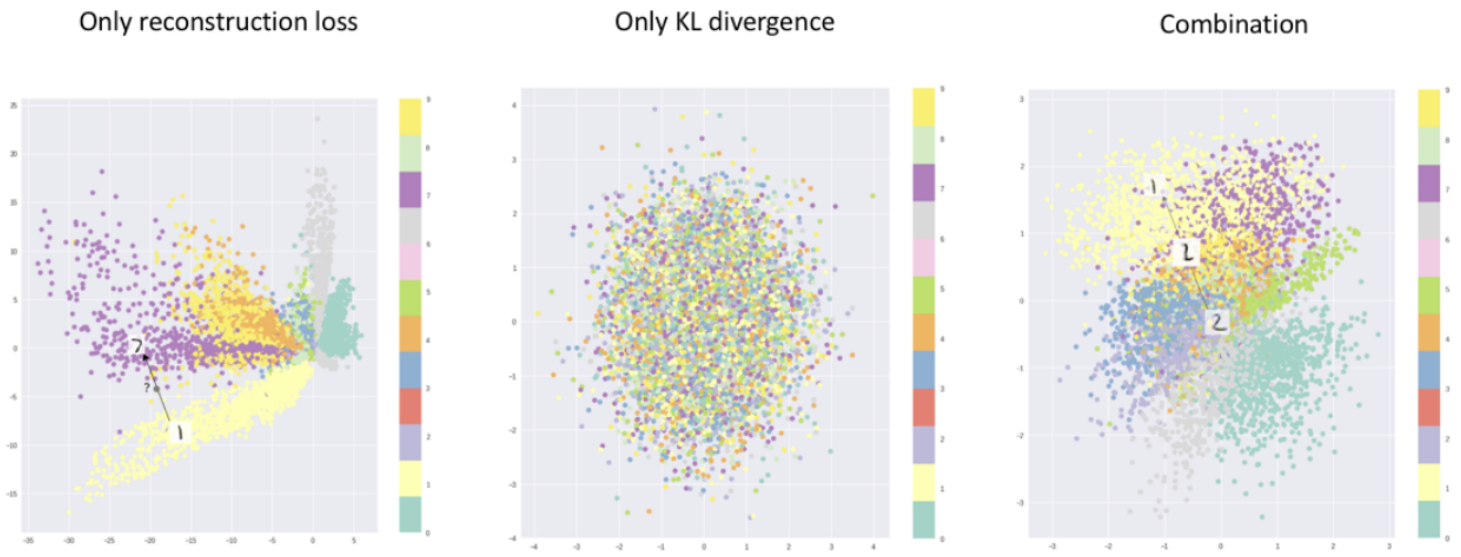


Figure 4: Image extracted from an online tutorial.

Interpolation in the latent space

- By travelling in the latent space we can interpolate in the dataset in a "meaningful" manner.

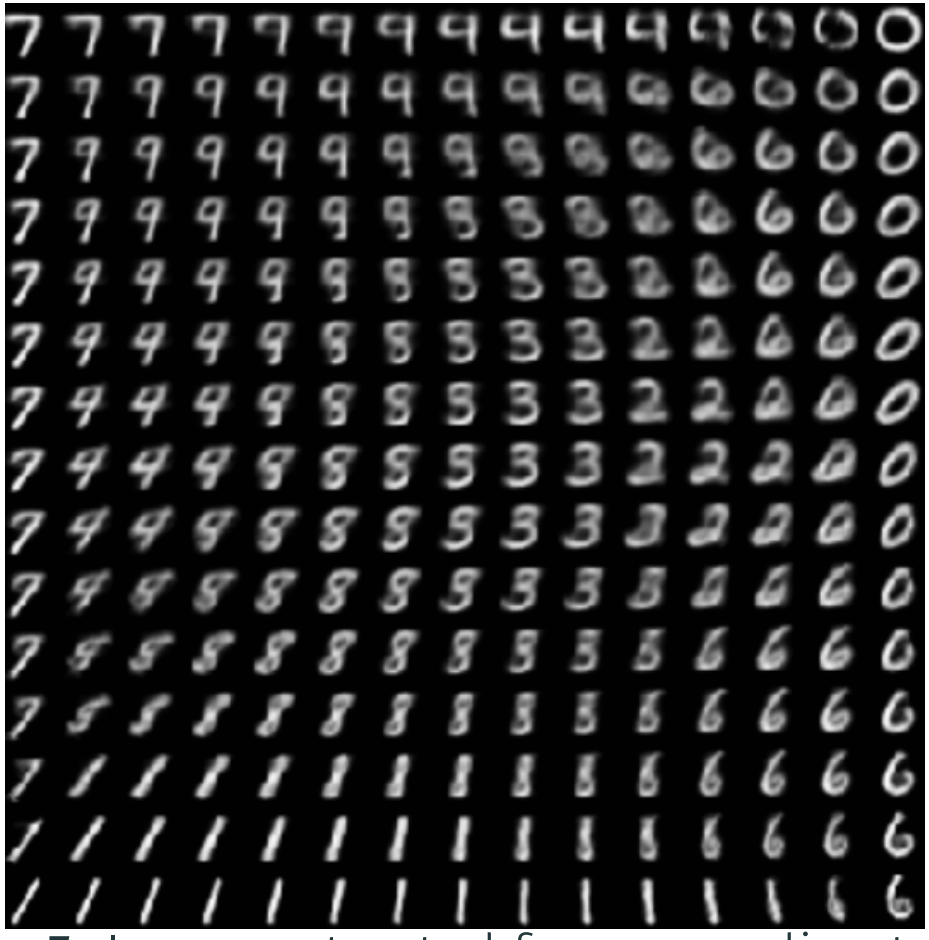


Figure 5: Image extracted from an online tutorial.

VAE training algorithm

Algorithm 1 Training of VAE

Input: n_{iter}, K, $\hat{\pi}$, q, estimator of $\nabla\mathcal{L}$, N_{batch}, δ_θ, δ_ϕ θ_0, ϕ_0.
for $n = 0$ to $n_{\text{iter}} - 1$ **do**
    Sample $X_n^{1:N_{\text{batch}}} = \{X_n^k\}_{k=1}^{N_{\text{batch}}}$ i.i.d. from $\hat{\pi}$.
    $Z_n^{1:N_{\text{batch}}} = \{Z_n^k\}_{k=1}^{N_{\text{batch}}}$ i.i.d. from q.
    Compute estimator of the gradient of the ELBO w.r.t. θ, $\hat{\nabla}_\theta\mathcal{L}(\theta_n, \phi_n)$
    Compute estimator of the gradient of the ELBO w.r.t. ϕ, $\hat{\nabla}_\phi\mathcal{L}(\theta_n, \phi_n)$
    $\theta_{n+1} = \theta_n + \delta_\theta\hat{\nabla}_\theta\mathcal{L}(\theta_n, \phi_n)$
    $\phi_{n+1} = \phi_n + \delta_\phi\hat{\nabla}_\phi\mathcal{L}(\theta_n, \phi_n)$

- Different choices of estimators (path derivative, total derivative).
- Stochastic gradient descent can be replaced by other algorithms (such as ADAM (**kingma2014adam**)).

Summary of VAEs

- Vanilla autoencoders consist in the reconstruction loss only.
- Variational autoencoders are better generative models.
- Advantages:
  - VAEs are easy to train with clear estimators of the ELBO.
  - They provide interesting latent representations.
- Problems:
  - VAEs with Gaussian priors are not competitive in generative modelling.
  - The choice of the latent space dimension is arbitrary.
  - The choice of the prior is arbitrary.
  - No easy likelihood evaluation.
- Links with other methods
  - VAE can be combined with normalising flows (**kingma2016improved**; **vahdat2020nvae**).
  - Score-based generative models can be seen as autoencoders (**huang2021variational**; **dieleman2022diffusion**; **ho2020denoising**).

Normalising flows

Principles of normalising flows

- Normalising flows can be seen as reparameterisation trick.
- We still aim at maximising the likelihood $\log(p_\theta(x))$.
  - Model p_θ flexible enough to approximate the data distribution.
  - Sampling from p_θ must be easy.
- The principles of normalising flows:
  - Start from a distribution π_0 with density p which is easy to sample.
  - Define $\pi_\theta = (g_\theta)_\# \pi_0$ and its density p_θ.
  - Maximise the log-likelihood $\log(p_\theta(x))$.

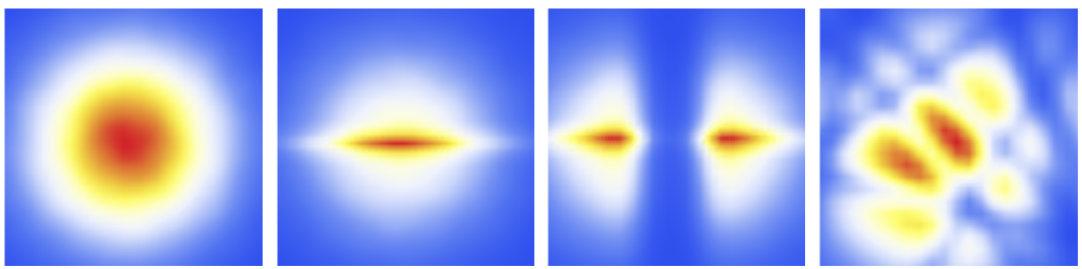


Figure 6: Several transformations of a $\mathrm{N}(0, \mathrm{Id})$ density. Image extracted from (**rezende2015variational**).

- In what follows:
  - First normalising flows and GLOW.
  - Autoregressive models and IAFVAE.

First normalising flows and GLOW

Invertible transformations

- The density of $(g_\theta)_{\#}\pi_0$ is given by a change of variable.
  - We assume that g_θ is a diffeomorphism (not necessary, one can use the co-area/area formula (caterini2021rectangular)).
  - Using the d-dimensional change of variable we have for any $f \in \mathrm{C}_c(\mathbb{R}^d, \mathbb{R})$

$$\mathbb{E}[f(X)] = \mathbb{E}[f(g_\theta(Z))] = \int_{\mathbb{R}^d} f(x) p(g_\theta^{-1}(x)) |J_\theta(g_\theta^{-1}(x))| \mathrm{d}z \ .$$

  - Hence, maximising the log-likelihood is equivalent to maximising

$$\boxed{\ell(\theta) = \log(p(g_\theta^{-1}(x))) + \log(|J_\theta(g_\theta^{-1}(x))|) \ .}$$

- Composition of transformation: $g_\theta = g_\theta^0 \circ g_\theta^1 \circ \cdots \circ g_\theta^K$.
- Conditions on the transformations:
  - g_θ and g_θ^{-1} are easy to compute and differentiate.
  - The Jacobian J_θ is easy to compute and differentiate.

Different types of flows

- Planar and radial flows (**rezende2015variational**).
  - Planar flow: $g : x \mapsto x + \mathbf{v}h(\mathbf{w}^\top x + b)$
  - Sylvester flow: $g : x \mapsto x + \mathbf{V}h(\mathbf{W}^\top x + \mathbf{b})$ (**vandenberg2018Sylvester**)
- Two other very efficient flows (**dinh2016density**; **dinh2014nice**):
  - Affine coupling layer.
  - Invertible 1x1 convolution.
- How does the affine coupling layer work?
  - We split $x \in \mathbb{R}^d$ in $x = (x_0, x_1)$ with $x_0 \in \mathbb{R}^{d_0}$, $x_1 \in \mathbb{R}^{d_1}$.
  - Forward transform $g_\theta(x) = (x_0, \exp[s_\theta(x_0)] \odot x_1 + t_\theta(x_0))$.
  - Reverse transform $g_\theta^{-1}(x) = (x_0, (x_1 - t_\theta(x_0)) \oslash \exp[s_\theta(x_0)])$.
  - Log-Jacobian: $\log(|J_\theta(x)|) = \sum_{i=1}^{d_1} s_\theta(x_0)_i$.
- How does the invertible 1x1 convolution work?
  - Matrix $\mathbf{W}_\theta \in \mathbb{R}^{C \times C}$ (number of channels), $x \in \mathbb{R}^{H \times W \times C}$.
  - Forward transform $g_\theta(x)_{i,j} = \mathbf{W}_\theta x_{i,j}$.
  - Reverse transform $g_\theta^{-1}(x)_{i,j} = \mathbf{W}_\theta^{-1} x_{i,j}$.
  - Log-Jacobian $\log(|J_\theta(x)|) = H \times W \times \log(|\mathbf{W}_\theta|)$.

Generative Flow (GLOW)

- Results obtained by (**kingma2018glow**).

- Combining actnorm, invertible convolution and affine coupling layers (multiple times).
- The "actnorm" layer is simply an affine layer.
- High quality results and interpolation.

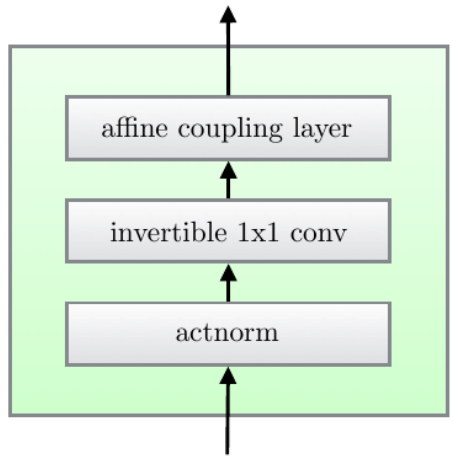


Figure 7: One step of GLOW. Image extracted from (**kingma2018glow**).

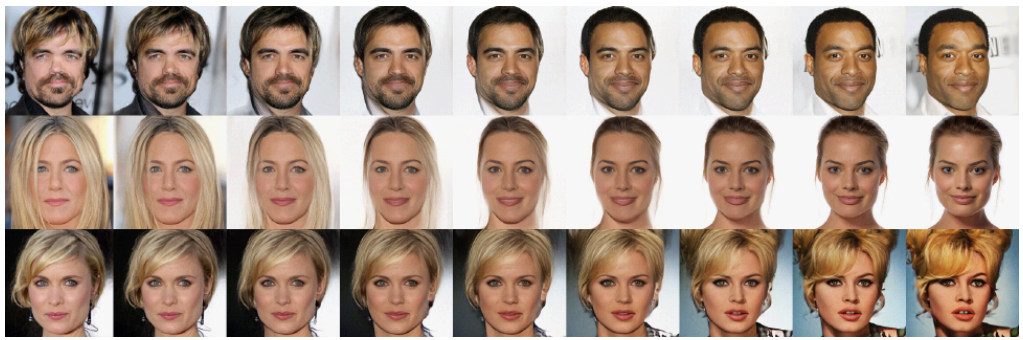


Figure 8: GLOW results. Image extracted from (**kingma2018glow**).

Autoregressive models and IAFVAE

A detour by autoregressive models

- Another generative modeling approach: autoregressive models
  - Masked Autoencoder for Distribution Estimation (autoregressive autoencoder), (**germain2015made**).
  - PixelRNN (autoregressive LSTM), (**van2016pixel**).
  - Both models are trained by maximising the log-likelihood.

- Both models assume the following raster-scan decomposition.

$$p_\theta(x) = \prod_{i=1}^{d} p_\theta(x_i | x_{1:i-1}) \ .$$

- Problems:
  - As many predicitions as the dimension.
  - Can be parallelized for training but not for sampling.

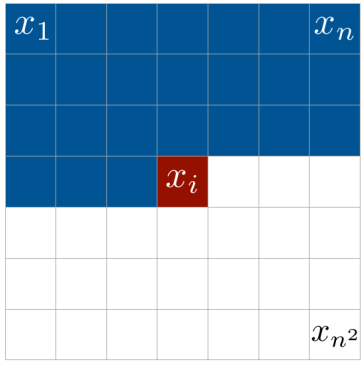


Figure 9: Raster scan order. Image extracted from (**van2016pixel**).

The autoregressive layer

- ▶ These ideas can however be reapplied to define a normalising flow layer.
- ▶ (kingma2016improved) introduces the autoregressive layer:

- $x = \{x_i\}_{i=1}^d$
- $\sigma_\theta^i(x_{1:i-1}) = \text{sigmoid}(s_\theta^i(x_{1:i-1}))$
- Forward transform $g_\theta(x)_i = \sigma_\theta^i(x_{1:i-1})x_i + (1 - \sigma_\theta^i(x_{1:i-1}))t_\theta^i(x_{1:i-1})$.
- Reverse transform $g_\theta^{-1}(x)_i = (x_i - (1 - \sigma_\theta^i(x_{1:i-1}))t_\theta^i(x_{1:i-1}))/\sigma_\theta^i(x_{1:i-1})$.
- Log-Jacobian $\sum_{i=1}^d \log(\sigma_\theta^i(x_{1:i-1}))$.
  - ▶ The Jacobian is triangular (easy computation of the determinant).
  - ▶ Parameterization with the sigmoid is numerically stable (inspired by LSTM (hochreiter1997long)).
  - ▶ Between each autoregressive layer the ordering is reversed.
  - ▶ More involved autoregressive models in practice:
- Masked autoencoders (germain2015made).
- Convolutional autoregressive models (van2016pixel).

Inverse Autoregressive Flow VAE

- Problem: flows tend to not be flexible enough for generative modelling.
  - (**kingma2016improved**): VAE with normalising flow prior.
  - The model is called Inverse Autoregressive Flow VAE.
- The "only" change compared to a classical VAE is the choice of prior:
  - Gaussian assumption in classical VAE.
  - Normalising flows in IAFVAE.
- The training is still done by maximising the ELBO. This is possible because one can compute $\log(q(z|x))$ when parameterized with normalising flows.

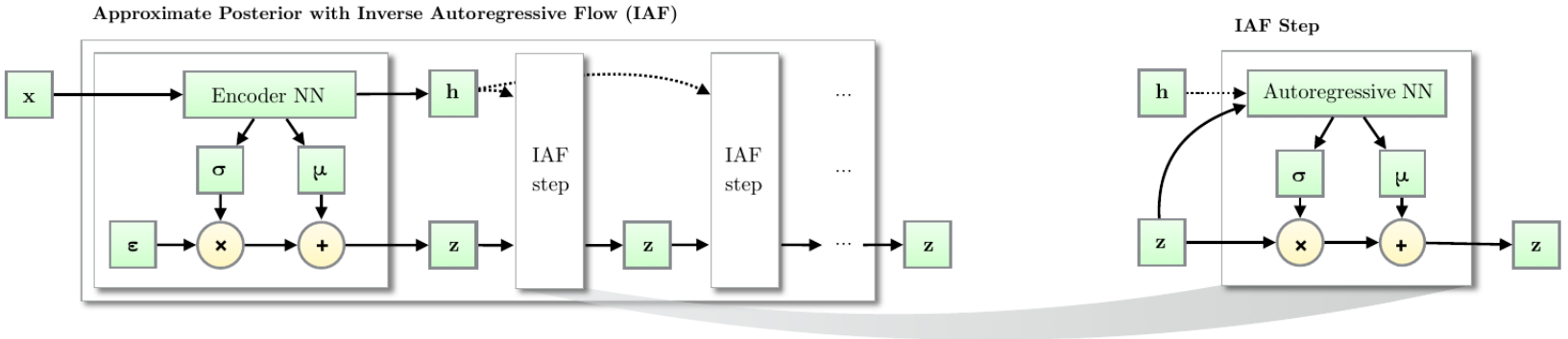


Figure 10: IAF worflow. Image from (**kingma2016improved**).

Rewriting the ELBO

- We recall that the ELBO is given by

$$\boxed{\mathcal{L}(\theta, \phi) = \int_{\mathbb{R}^p} \log(p_\theta(x|z)) q_\phi(z|x) \mathrm{d}z - \mathrm{KL}(q_\phi(\cdot|x)|p) \ .}$$

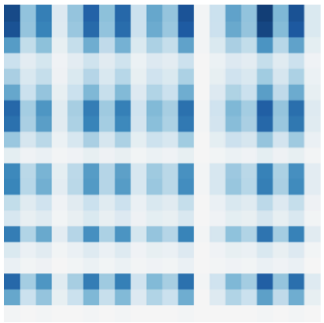
- Usually, $p(z)$ is a Gaussian prior.
- More complicated prior p_Ψ (parametric form).

$$\boxed{\mathcal{L}(\theta, \phi) = \int_{\mathbb{R}^p} \log(p_\theta(x|z)) q_\phi(z|x) \mathrm{d}z - \mathrm{KL}(q_\phi(\cdot|x)|p_\Psi) \ .}$$

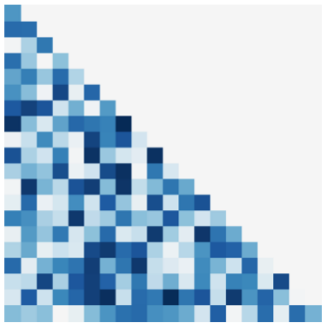
- Some interesting cases:

- Cascade of Gaussian models (as in **sonderby2016ladder**).
- Normalising flow (as in IAF-VAE **kingma2016improved**; **chen2016variational**).
- Diffusion model (as in **vahdat2021score**; **wehenkel2021diffusion**).
- In the case of diffusion model, we need to derive another ELBO.

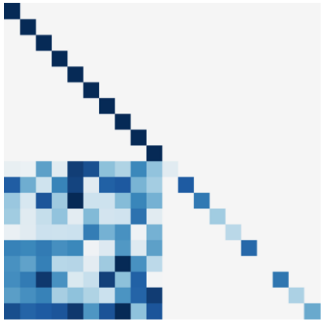
Summary of Normalising Flows



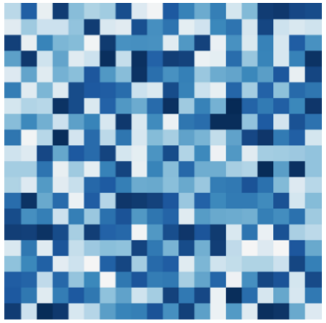
(a) Det. Identities
(Low Rank)



(b) Autoregressive
(Lower Triangular)



(c) Coupling
(Structured Sparsity)



(d) **Unbiased Est.**
(Free-form)

Figure 11: Different normalising flows and their Jacobian structure. Image from (**chen2019Residual**).

Summary of Normalising Flows

- Advantages:
  - Normalising Flows are flexible.
- Problems:
  - There is no latent representation.
  - Vanilla normalising flows are not competitive in generative modeling.
  - The class of flows is restricted in the discrete-time setting.
  - The training can be complicated in the continuous-time setting.
- Links with other methods
  - VAE can be combined with normalising flows (**kingma2016improved**; **vahdat2020nvae**).
  - Score-based generative models can be seen as normalising flows (**song2021maximum**).

Generative Adversarial Networks

Principles of Vanilla GAN

- In Generative Adversarial Network models we *do not* optimise the log-likelihood or a lower-bound on the log-likelihood.
- Instead we rely on a minimax game.
- We train two competing network.
  - A generative network which synthesizes data (fake data).
  - A discriminative network which tells which data is fake or real.
- This is still related to a divergence on probability measures.

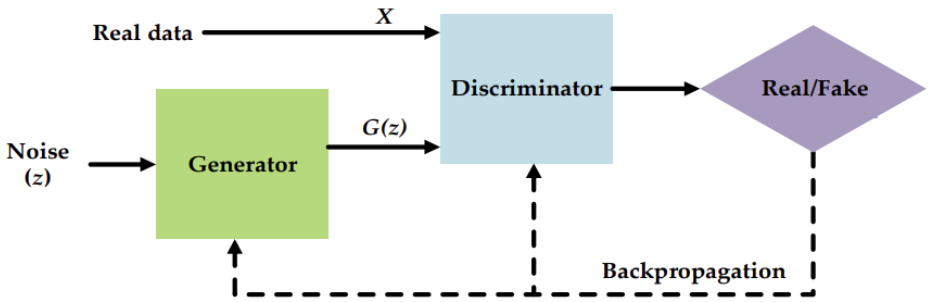


Figure 12: Original GAN model. Image extracted from (feng2020generative).

- In what follows:
  - Vanilla and Least-Square GANs.
  - IPM and WPGAN.

Vanilla and Least-Square GANs

Loss function and Jensen-Shannon divergence

- We consider a generator $g : \mathbb{R}^p \to \mathbb{R}^d$ and a discriminator $d : \mathbb{R}^d \to [0,1]$ (networks) which optimise the loss

$$\ell(g, d) = -\int_{\mathbb{R}^d} \log(d(x))\mathrm{d}\pi(x) - \int_{\mathbb{R}^p} \log(1 - d(g(z)))\mathrm{d}\pi_0(z) \ .$$

  - π is the data distribution, π_0 is an easy-to-sample distribution.
  - We denote p_g the density of $g_{\#}\pi_0$ and p the one of π.
  - In practice we parameterise the generator and discriminator.
- For a fixed generator, the optimal discriminator is given by $d^\star$ s.t.

$$d^\star(x) = p(x)/(p(x) + p_g(x)) \ .$$

- Plugging this optimal discriminator into ℓ we get

$$\ell(g, d^\star) = \log 4 - \int_{\mathbb{R}^d} \log(p(x)/p_{\mathrm{mid}}(x))\mathrm{d}p(x) - \int_{\mathbb{R}^d} \log(p_g(x)/p_{\mathrm{mid}}(x))\mathrm{d}p_g($$

  - Hence, $\ell(g, d^\star) = \log(4) - \mathrm{JS}(\pi, g_{\#}\pi)$, where JS is the Jensen-Shannon divergence.

A link with regression

- The discriminator tries to classify the data
  - $d(x) = 1$ if the data is from the original dataset.
  - $d(x) = 0$ if the data is from the generated dataset.
- Consider Y a Bernoulli r.v. and $Y = 1$ with probability $d_\theta(X)$.
- We have $p(Y|x,\theta) = p(Y=1|x,\theta)^Y p(Y=0|x,\theta)^{1-Y}$. Hence, we get

$$\begin{aligned}
&\int_{\mathbb{R}^d\times\{0,1\}} \log(p(y|x,\theta))\mathrm{d}\bar{\pi}(x,y) \\
&= \int_{\mathbb{R}^d\times\{0,1\}} \{y \log p(y=1|x,\theta) + (1-y)\log p(y=0|x,\theta)\}\bar{p}(x)\mathrm{d}\bar{\pi}(x,y) \\
&= \int_{\mathbb{R}^d\times\{0,1\}} \{y \log(d_\theta(x)) + (1-y)\log(1-d_\theta(x))\}\mathrm{d}\bar{\pi}(x,y)\ .
\end{aligned}$$

- $\bar{\pi}$ is the such that $\bar{\pi}(\mathbb{R}^d, \cdot) = \mathrm{Ber}(1/2)$.
  - $(X,Y) \sim \bar{\pi}$ is such that $X \sim \pi$ (data distribution) if $Y = 1$ and $X \sim g_\# \pi_0$ (generated distribution) if $Y = 0$.

$$\begin{aligned}
&\int_{\mathbb{R}^d\times\{0,1\}} \log(p(y|x,\theta))\mathrm{d}\bar{\pi}(x,y) \\
&\quad = (1/2)\int_{\mathbb{R}^d} \log(d_\theta(x))\mathrm{d}\pi(x) + (1/2)\int_{\mathbb{R}^p} \log(1-d_\theta(g(z)))\mathrm{d}\pi_0(z)\ .
\end{aligned}$$

- We recover the cross-entropy loss. This is the same as optimising $\mathrm{KL}(\bar{\pi}|p(\cdot|\theta))$ (maximum likelihood).

Least-square GAN

- Another flavor of GAN: Least Square GAN (mao2017least).
- The discriminator is a classifier.
  - In vanilla GAN we consider the cross-entropy loss (goodfellow2014generative).
  - In LSGAN we consider the square loss (mao2017least).
- The (coupled) losses are given by

$$\ell_g(d) = \int_{\mathbb{R}^d}(d(x)-1)^2 \mathrm{d}\pi(x) + \int_{\mathbb{R}^p}(d(g(z))+1)^2 \mathrm{d}\pi_0(z)\ ,$$

$$\ell_d(g) = \int_{\mathbb{R}^p}(d(g(z))-1)^2 \mathrm{d}\pi_0(z)\ .$$

- Thus $\ell_{d^\star}(g) = \chi^2((\pi + g_\# \pi_0)/2 | g_\# \pi_0)$ (χ^2 the Pearson divergence).

Figure 13: LSGAN results on LSUN. Image extracted from (mao2017least).

IPM and WPGAN

Integral Probability Metrics

- The concept of generator/discriminator can be recovered using Integral Probability Metrics.
- An IPM is defined by a class of functions $\mathsf{F} \subset \mathcal{F}(\mathbb{R}^d)$ (measurable function from $\mathbb{R}^d$ to $\mathbb{R}$).
- We define $\mathcal{P}_\mathsf{F} = \{\pi \in \mathcal{P}(\mathbb{R}^d) \ : \ \sup_{f \in \mathsf{F}_0} \pi[|f|] < +\infty\}$ (with $f \in \mathsf{F}_0$ if $f(0) = 0$ and $f \in \mathsf{F}$) and d_F such that for any $\pi_1, \pi_2 \in \mathcal{P}_\mathsf{F}$

$$\boxed{d_\mathsf{F}(\pi_1, \pi_2) = \sup\{\pi_1[f] - \pi_2[f] \ : \ f \in \mathsf{F}\} \ .}$$

  - Symmetric and non-negative if $\mathsf{F} = -\mathsf{F}$.
  - Defines a distance on $\mathcal{P}_\mathsf{F}$ if F separates $\mathcal{P}_\mathsf{F}$ in the following sense: for any $\pi_1, \pi_2 \in \mathcal{P}_\mathsf{F}$ there exists $f \in \mathsf{F}$ such that $\pi_1[f] - \pi_2[f] \neq 0$.
  - $f \in \mathsf{F}$ can be seen as a discriminator between two probability measures.
- Let F be the set of 1-Lipschitz functions.
  - F is separating.
  - The associated IPM is called the Wasserstein distance of order 1 and is denoted $\mathbf{W}_1$.

Basics on Wasserstein distances

- We have define the Wasserstein distance of order one as

$$\boxed{\mathbf{W}_1(\pi_1, \pi_2) = \sup\{\pi_1[f] - \pi_2[f] \ : \ f \in \mathrm{Lip}_1(\mathbb{R}^d)\} \ .}$$

- This is the dual formulation of the following definition

$$\boxed{\mathbf{W}_1(\pi_1, \pi_2) = \inf\{\textstyle\int_{\mathbb{R}^d \times \mathbb{R}^d} \|x - y\| \mathrm{d}\Pi(x, y) \ : \ \Pi \in \Lambda(\pi_1, \pi_2)\} \ .}$$

  - $\Lambda(\pi_1, \pi_2)$ is the set of couplings between π_1 and π_2
  - For any $\mathsf{A} \in \mathcal{P}(\mathbb{R}^d \times \mathbb{R}^d)$, $\Pi(\mathsf{A} \times \mathbb{R}^d) = \pi_1(\mathsf{A})$ and $\Pi(\mathbb{R}^d \times \mathsf{A}) = \pi_2(\mathsf{A})$.
- By changing $\|x - y\|$ into $\|x - y\|^p$ we can define Wasserstein cost of order p with $p > 0$.
  - This is a distance on $\mathcal{P}_p(\mathbb{R}^d) = \{\pi \in \mathcal{P}(\mathbb{R}^d) \ : \ \int_{\mathbb{R}^d} \|x\|^p \, \mathrm{d}\pi(x) < +\infty\}$ if $p \geq 1$.
  - This distance is stronger than the weak convergence (equivalent on compact sets).
  - Wasserstein costs are IPM only for $p = 1$.

Wasserstein GAN

- Different divergences yield different losses (vanilla GAN, LSGAN).
- Each IPM can be turned into a GAN.

$$\boxed{\inf\{d_{\mathsf{F}}(\pi, g_{\#}\pi_0) \ : \ g \in \mathsf{G}\} = \inf \sup\{\pi[f] - \pi_0[f \circ g] \ : \ f \in \mathsf{F}, g \in \mathsf{G}\} \ ,}$$

  - F is the space of test functions (discriminators).
  - F is the space of generators.
- (**arjovsky2017wasserstein**) uses the Wasserstein distance of order one.
  - Lipschitz condition is enforced by clipping of the parameters.

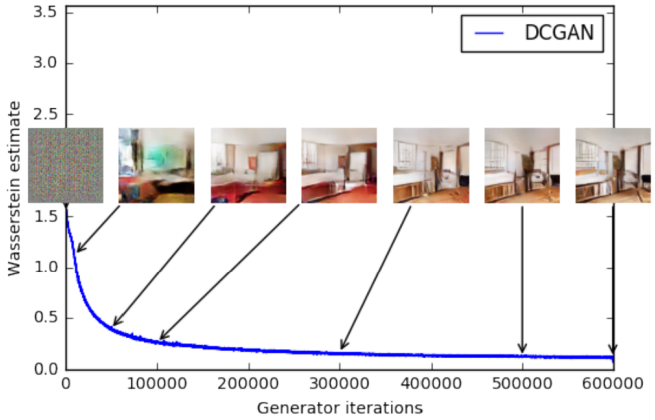


Figure 14: Influence of training on Wasserstein GANs. Image extracted from (**arjovsky2017wasserstein**).

Stability with gradient penalty

- Gradient clipping can lead to undesired behavior.
- (**gulrajani2017improved**) proposes to change the loss function of the GAN.

$$\boxed{\ell(f,g) = \pi[f] - \pi_0[f \circ g] + \lambda \pi_0[(\|\nabla f\| \circ g - 1)^2] \, .}$$

- $\lambda > 0$ is a regularization parameter.
- The last term is a gradient penalty.

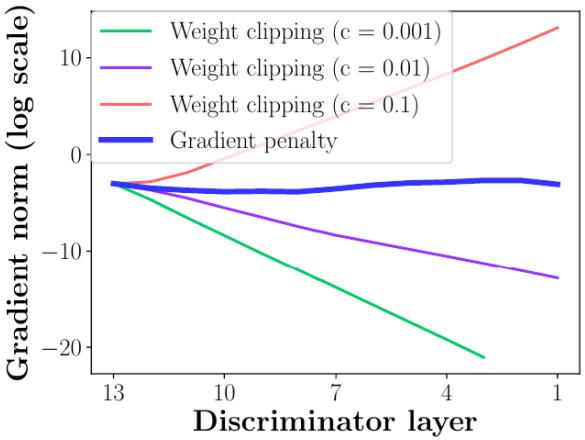


Figure 15: Influence of the regularisation. Image extracted from (**gulrajani2017improved**).

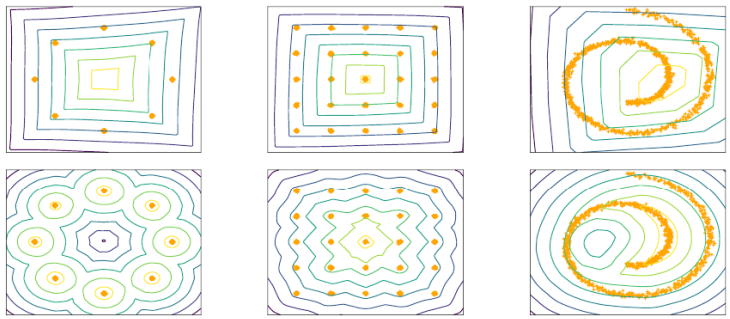


Figure 16: Influence of the gradient penalty. Image extracted from (**gulrajani2017improved**).

Summary of GANs

- Advantages:
  - GANs provide state-of-the-art results
  - They provide interesting latent representations.
  - They allows flexible losses and formulations.
- Problems:
  - it is very hard to train (collapse during training).
  - Diversity is a problem (mode collapse).
  - Theoretical analysis is hard (biau2020some).
- Links with other methods
  - GANs can be combined with score-based models (xiao2021tackling).

Conclusion

Conclusion

- Generative modelling has many different flavors:
  - Variational AutoEncoders.
  - Normalizing Flows (and Autoregressive models).
  - Generative Adversarial Networks.
- Depending on the application architecture matters.
- Until recently GANs were the state-of-the-art in terms of visual results.
- In the next sessions: score-based generative modelling:
  - New contender with state-of-the-art results.
  - Theoretical analysis is possible.
  - Links with stochastic control and optimal control.