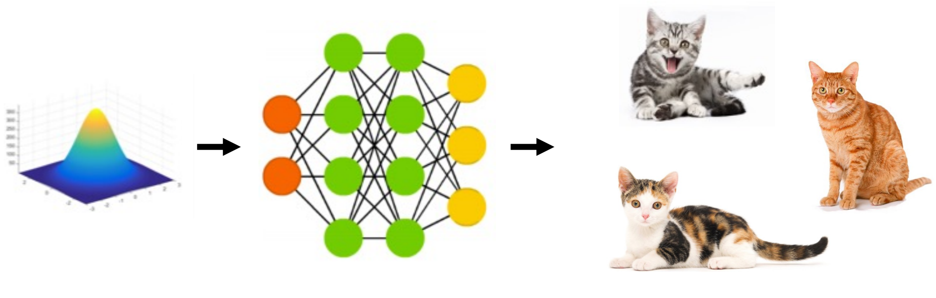
Generative Models

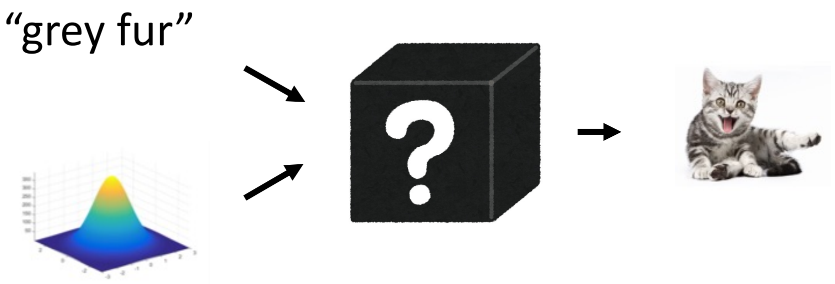
Advances & Applications

Yingzhen Li (yingzhen.li@imperial.ac.uk)

Conditional latent variable models



(unconditional) latent variable models

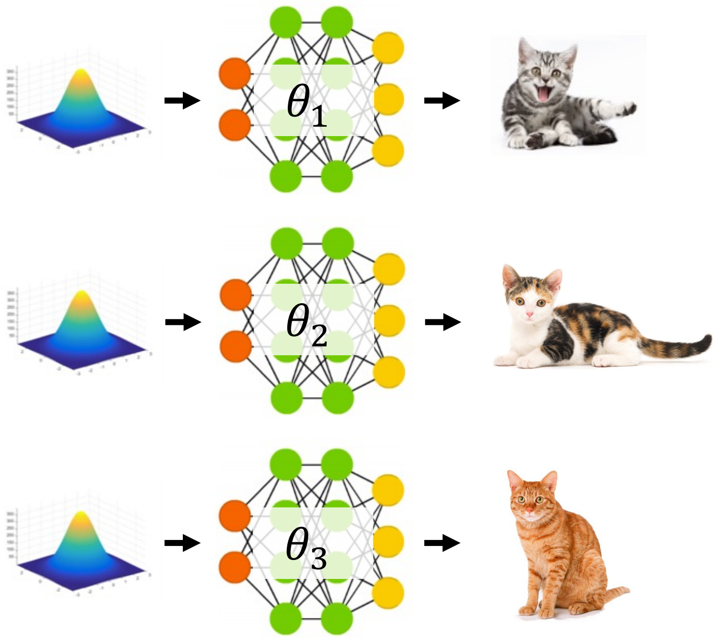


How to construct conditional LVMs?

Conditional latent variable models

- Goal: learn a generative model $p_\theta(x|y)$
  - x: data to be generated (e.g. an image)
  - y: label/info that the generation process is conditioned on (e.g. fur colour)

- Idea 1: if $y \in \{1, \dots, C\}$, train a set of models
  $p_\theta(x|y=c) = p_{\theta_c}(x) = \int p_{\theta_c}(x|z)p(z)dz$
  - Parameter inefficient: need to train C networks
  - Cannot generalise to continuous y



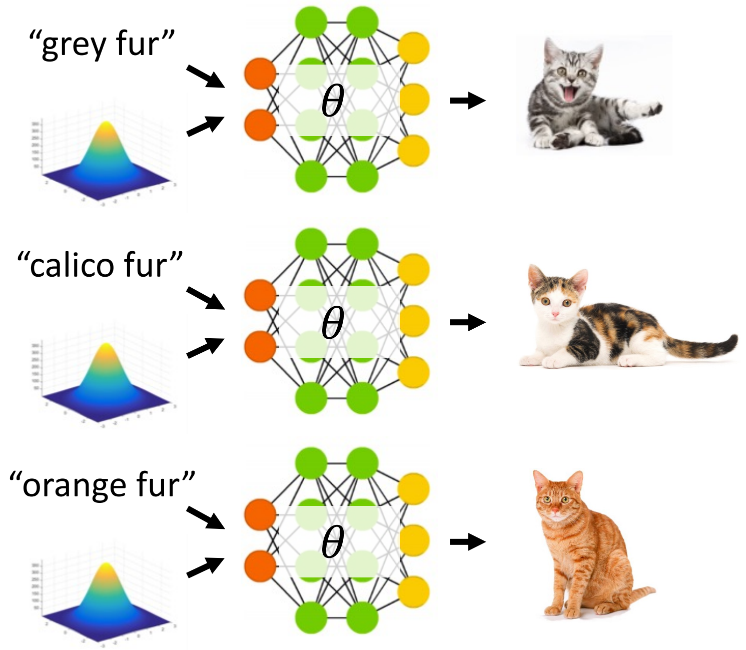
Conditional latent variable models

- Goal: learn a generative model $p_\theta(x|y)$
  - x: data to be generated (e.g. an image)
  - y: label/info that the generation process is conditioned on (e.g. fur colour)

- Idea 2: make (z, y) as the input of the network

  $$p_\theta(x|y=c) = \int p_\theta(x|z, y=c)p(z)dz$$

  - Parameter ~~inefficient~~ efficient
  - Can~~not~~ generalise to continuous y
  - Disentangled the learned representation z from the label info y



Conditional VAEs

- Training the conditional LVM:

model: $p_\theta(x|y) = \int p_\theta(x|z,y)p(z)dz,$ data: $\{(x_n, y_n)\}_{n=1}^N \sim p_{data}(x,y)$

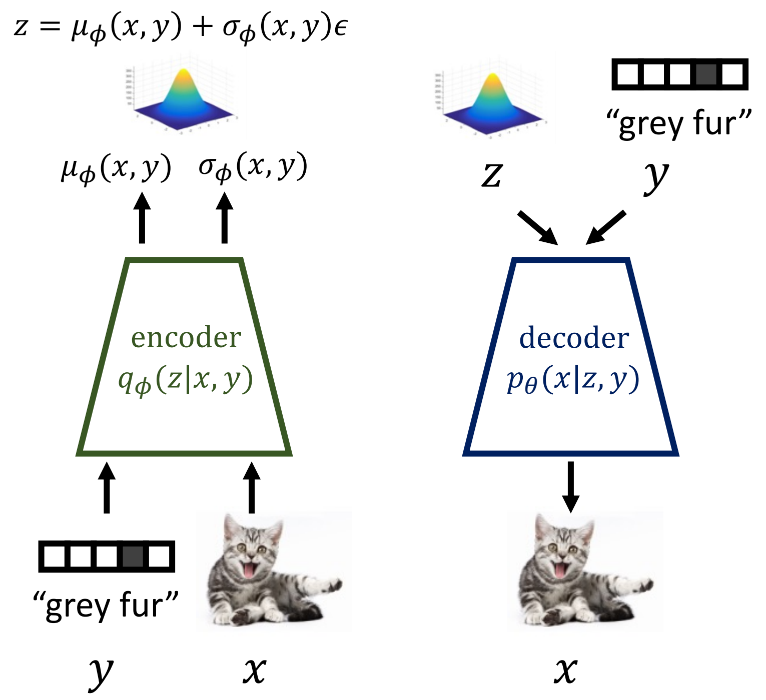
  - Maximum Likelihood training (MLE):

$$\max_\theta E_{p_{data}(x,y)}[\log p_\theta(x|y)]$$

  - (conditional) variational lower-bound:

$$\log p_\theta(x|y) \geq E_{q_\phi(z|x,y)}[\log p_\theta(x|z,y)] - KL[q_\phi(z|x,y)\|p(z)]$$
$$:= L(x,y,\phi,\theta)$$

$\Rightarrow$ maximise $E_{p_{data}(x,y)}[L(x,y,\phi,\theta)]$ w.r.t. ϕ, θ



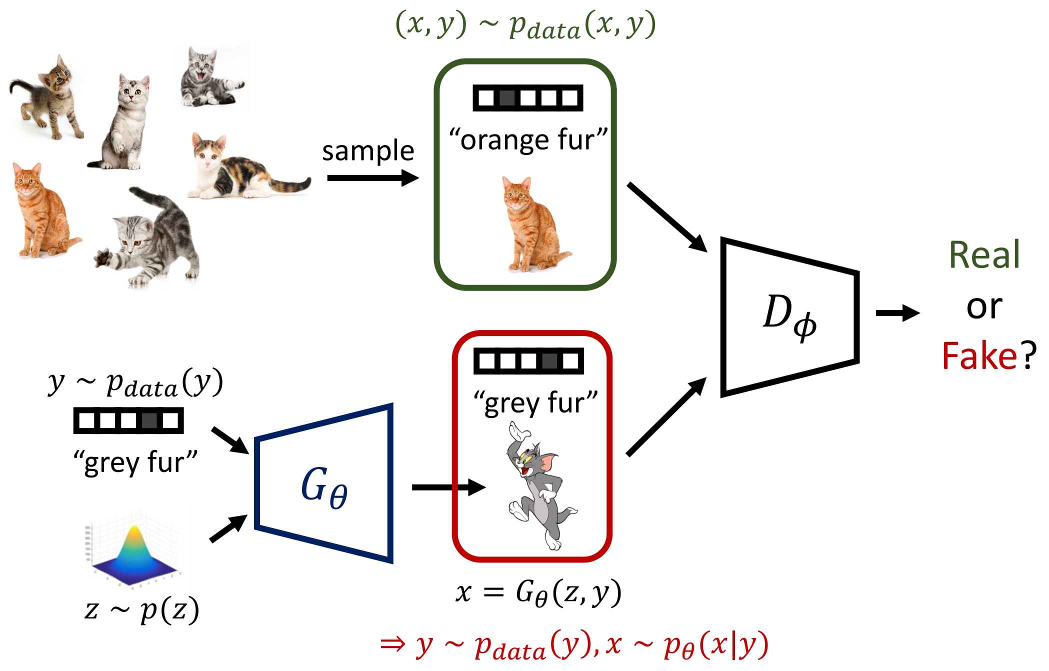
Conditional GANs

- Training the conditional LVM:

$$\text{model: } p_\theta(x|y) = \int p_\theta(x|z,y)p(z)dz, \qquad \text{data: } \{(x_n, y_n)\}_{n=1}^N \sim p_{data}(x,y)$$

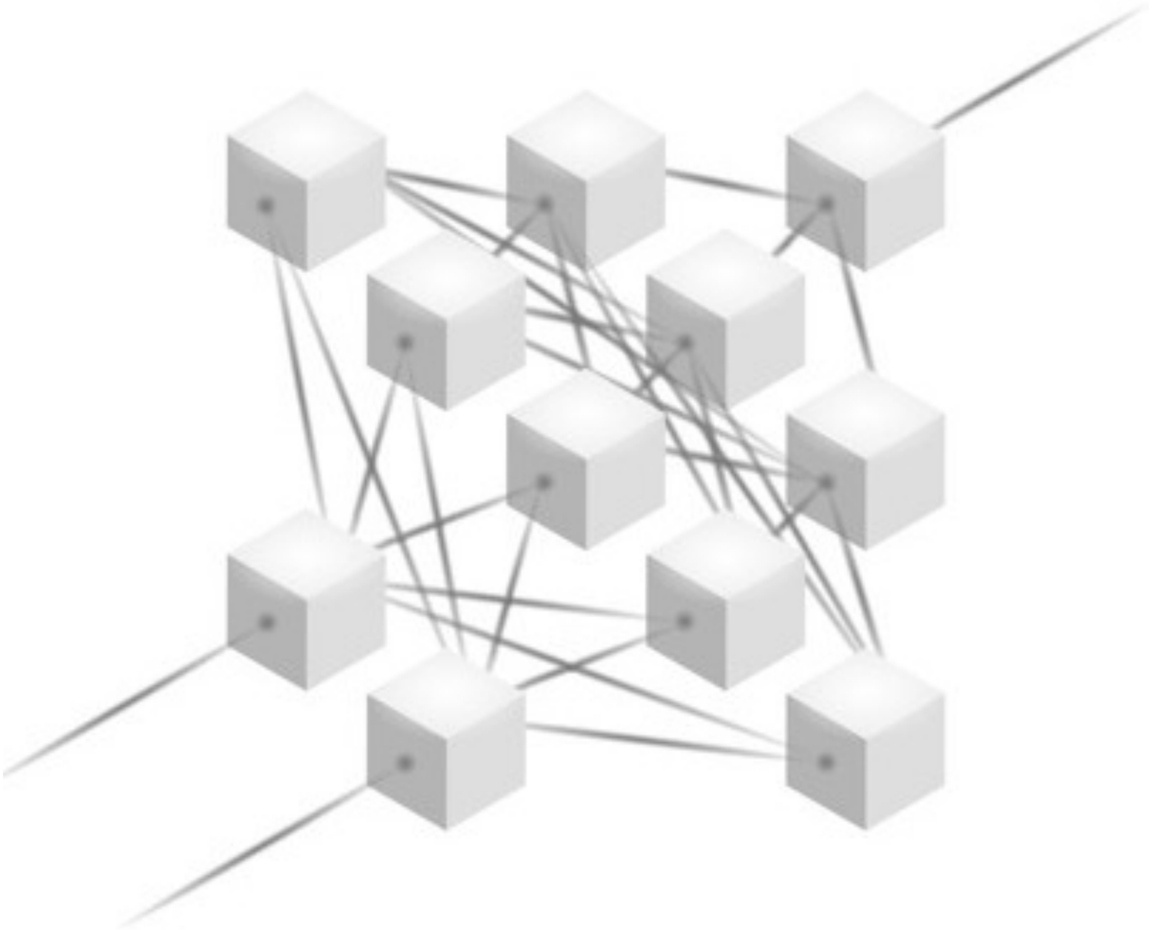
  - Adversarial training:
    - Label $(x_n, y_n) \sim p_{data}(x,y)$ as "real"
    - Label $(G_\theta(z,y), y), z \sim p(z)$ as "fake"
    - For fake data, sample label $y \sim p_{data}(y)$

$$\min_\theta \max_\phi \underbrace{E_{p_{data}(x,y)}[\log D_\phi(x,y)]}_{\text{"real"}} + \underbrace{E_{p(z)p_{data}(y)}[\log(1 - D_\phi(G_\theta(z,y),y))]}_{\text{"fake"}}$$

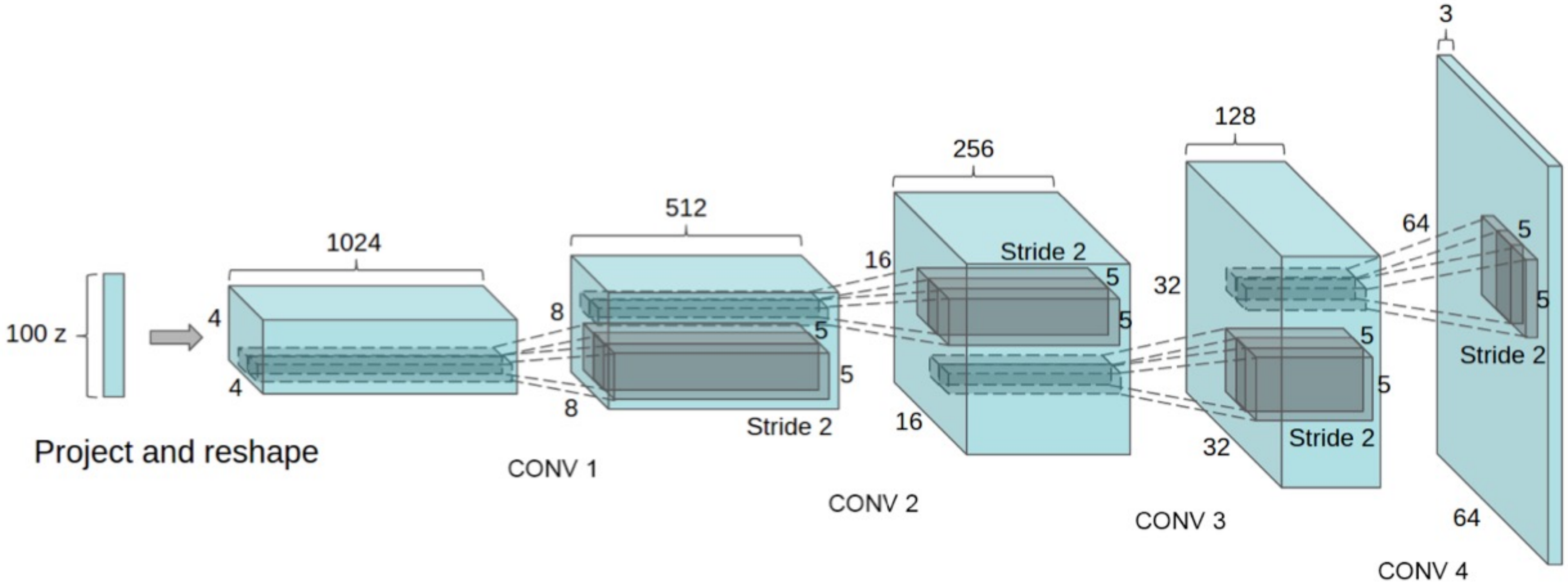


Mirza and Osindero. Conditional Generative Adversarial Networks. arXiv:1411.1784

Generative Model Architecture Design



DCGAN

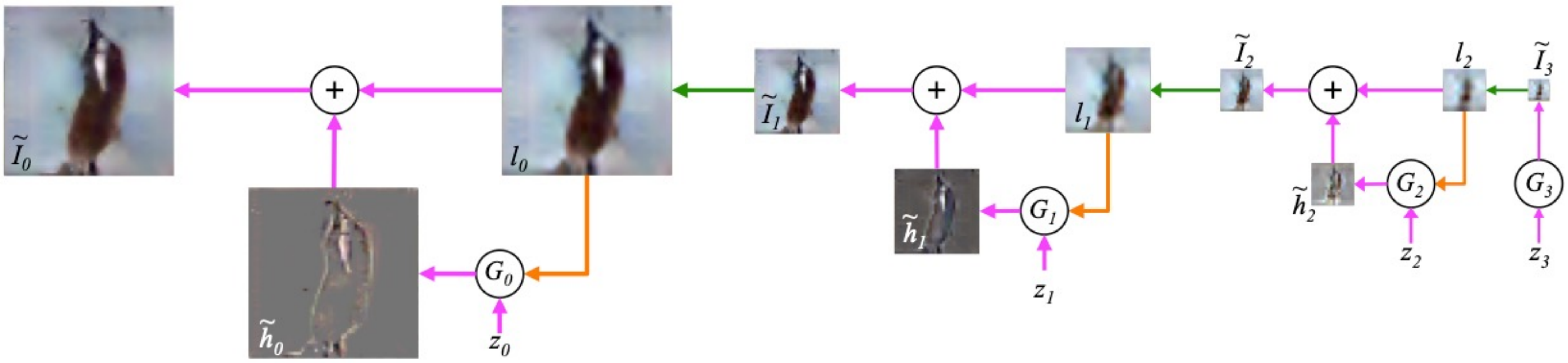


Tricks used in the DCGAN architecture & training:

- Replace pooling layers with strided convolutions (discriminator) and fractional-strided convolutions (generator).
- Use batchnorm.
- Remove fully connected hidden layers for deeper architectures.
- Use LeakyReLU activation in the discriminator for all layers.

Radford et al. Unsupervised Representation Learning with Deep Convolutional Generative Adversarial Networks. ICLR 2016

LAPGAN



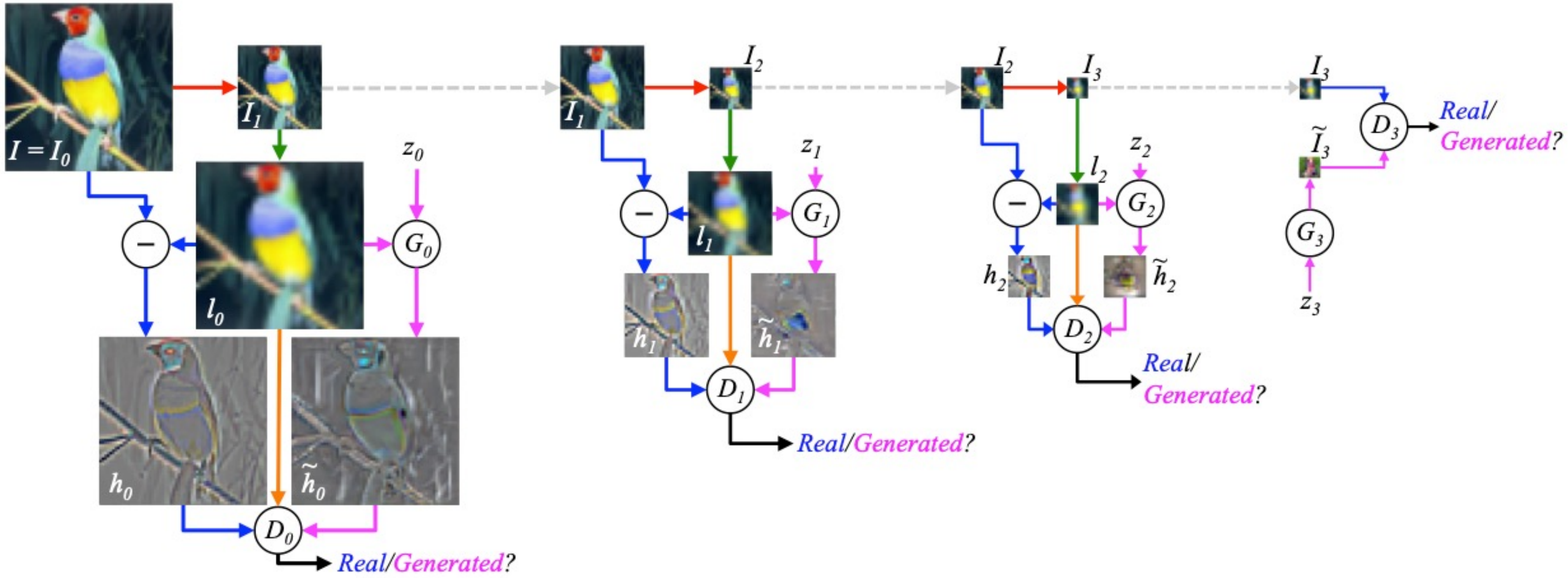
Generator's multi-scale architecture:

- Start generation for low-resolution images: $\tilde{I}_3 = G_3(z_3)$
- Generate higher-resolution images conditioned on the lower-resolution ones:

$$l_i = upscale(\tilde{I}_{i+1}), \qquad \tilde{I}_i = l_i + G_i(z_i, l_i)$$

Denton et al. Deep Generative Image Models using a Laplacian Pyramid of Adversarial Networks. NeurIPS 2015

LAPGAN



LAPGAN's discriminator design:

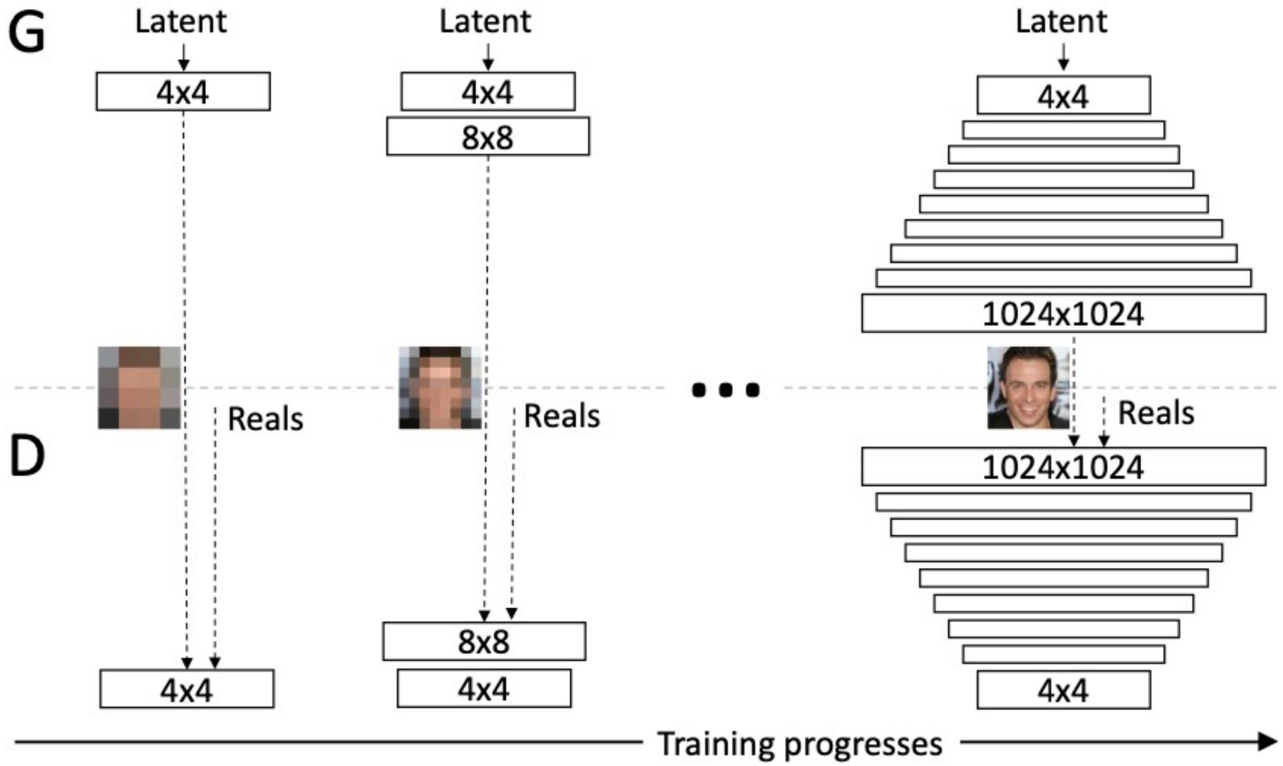
- Multiple discriminators in use, paired with generators at different resolutions
- At each resolution, check whether the "fine details" generated by G_i matches the real ones:
    - "real" input: $h_i = I_i - l_i, l_i = upscale(I_{i+1}), I_{i+1} = downscale(I_i)$
    - "fake" input: $\tilde{h}_i = G_i(z_i, l_i)$

Denton et al. Deep Generative Image Models using a Laplacian Pyramid of Adversarial Networks. NeurIPS 2015

Progressive GAN

Progressively building GAN generator and discriminator:

- High-res images downscaled to get training data of low resolutions
- Train a GAN starting from 4x4 images
- Add new layers into generator and discriminator
- Adapt old & new layers by GAN training with 8x8
- Continue with 16x16, 32x32…



Karras et al. Progressive Growing of GANs for Improved Quality, Stability, and Variation. ICLR 2018

StyleGAN

Disentangling different sources of randomness:

- Latent variable z is transformed to "style" representation w
- This "style" w controls generation at every resolutions
- Fine details generated with noise at different scales

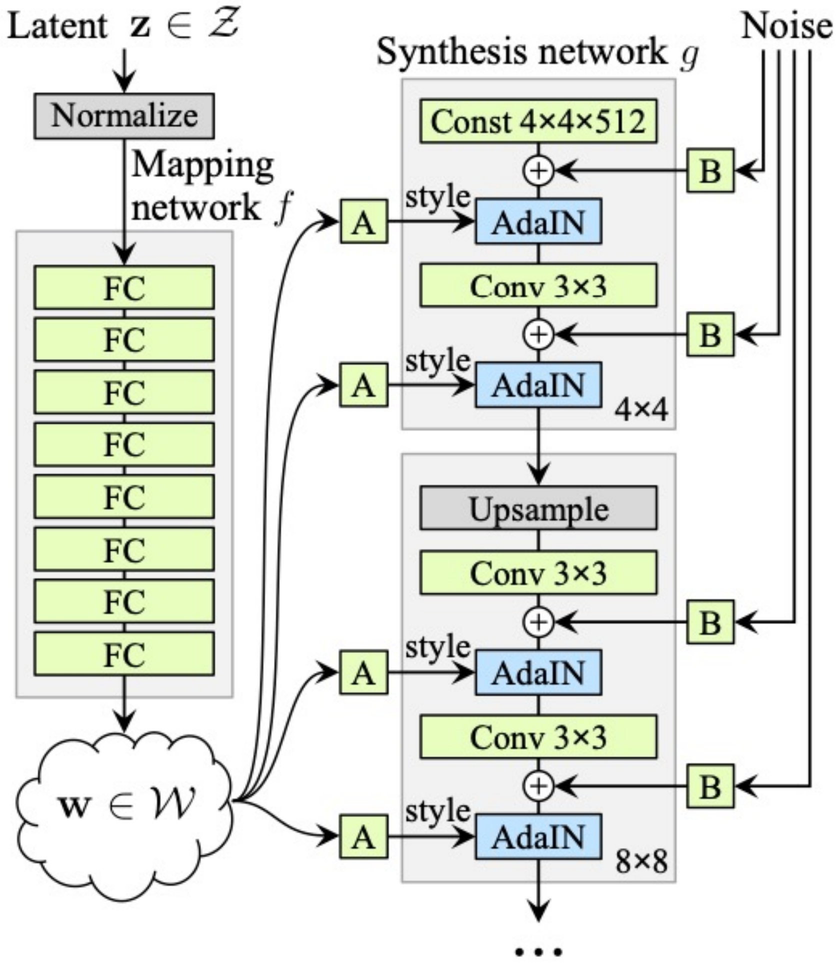
$$y = (y_s, y_b) = A(w)$$

$$x = \text{"upscaled last block output"} + B(\epsilon)$$

$$AdaIN(x_i, y) = y_{s,i} \boxed{\frac{x_i - \mu(x_i)}{\sigma(x_i)}} + y_{b,i}$$

Channel index

normalised feature map for each channel



Karras et al. A Style-Based Generator Architecture for Generative Adversarial Networks. CVPR 2019

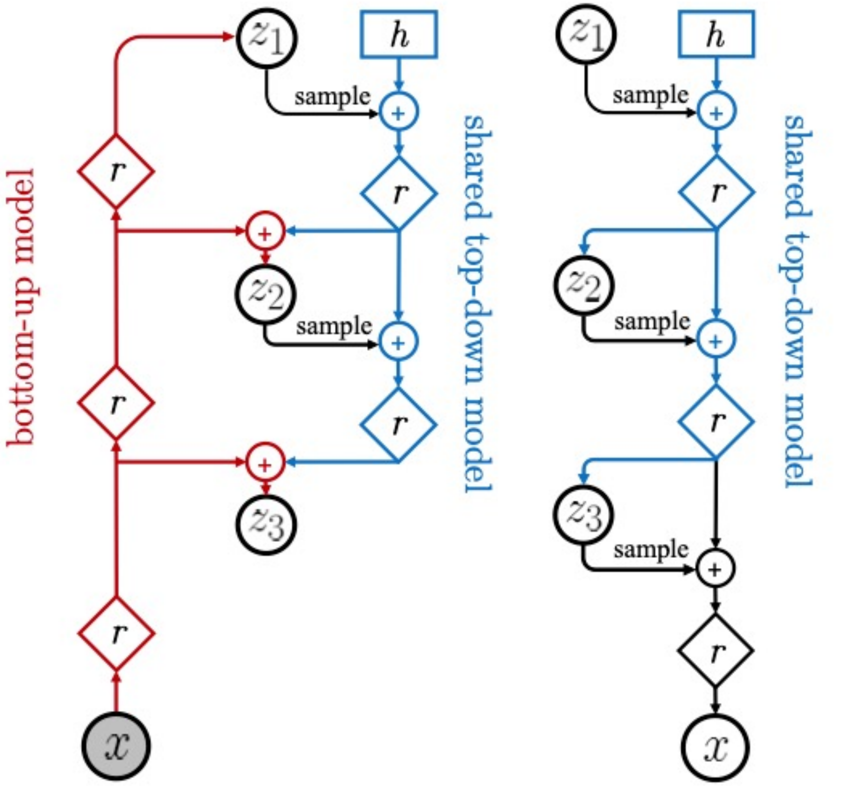
StyleGAN

Karras et al. A Style-Based Generator Architecture for Generative Adversarial Networks. CVPR 2019

NVAE – improved VAE image generation

State-of-the-art VAE for image generation (2020):

- Hierarchical LVM with multi-scale architecture
- Using residual networks for the r blocks
- BatchNorm in usage
- Improved q distribution design to control the $KL[q(z|x)||p(z)]$ term

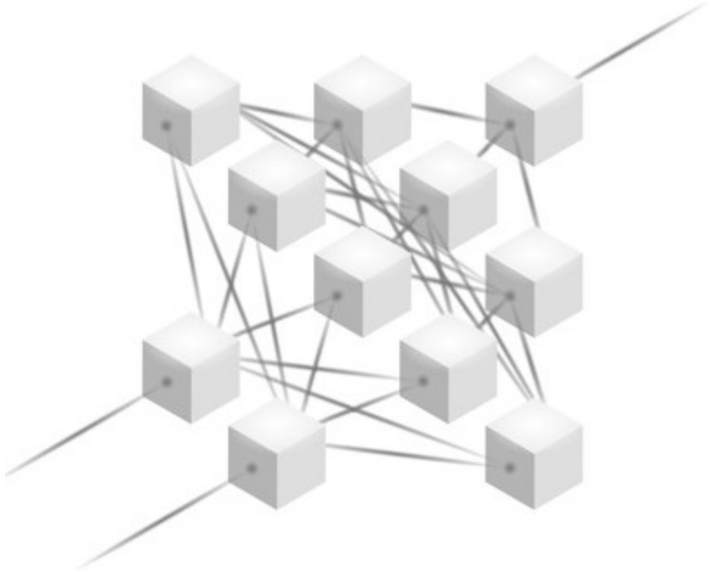


(a) Bidirectional Encoder (b) Generative Model

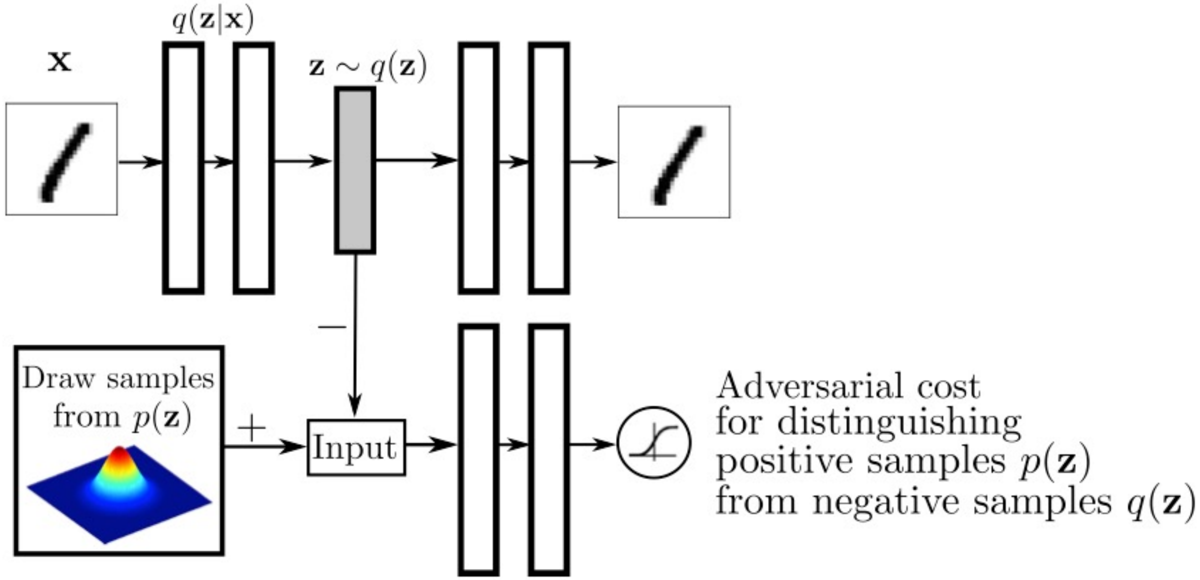
Vahdat and Kautz. NVAE: A Deep Hierarchical Variational Autoencoder. NeurIPS 2020

Progress in architecture design

- GAN progression:
  - DCGAN – fully convolutional neural networks
  - LAPGAN & Progressive GAN – multi-scale architectures
  - StyleGAN – disentangling sources of randomness
- VAE progression:
  - Hierarchical LVMs
  - Tuning the KL regulariser
  - Deep learning tricks applied
  - Incorporate design ideas from GAN networks



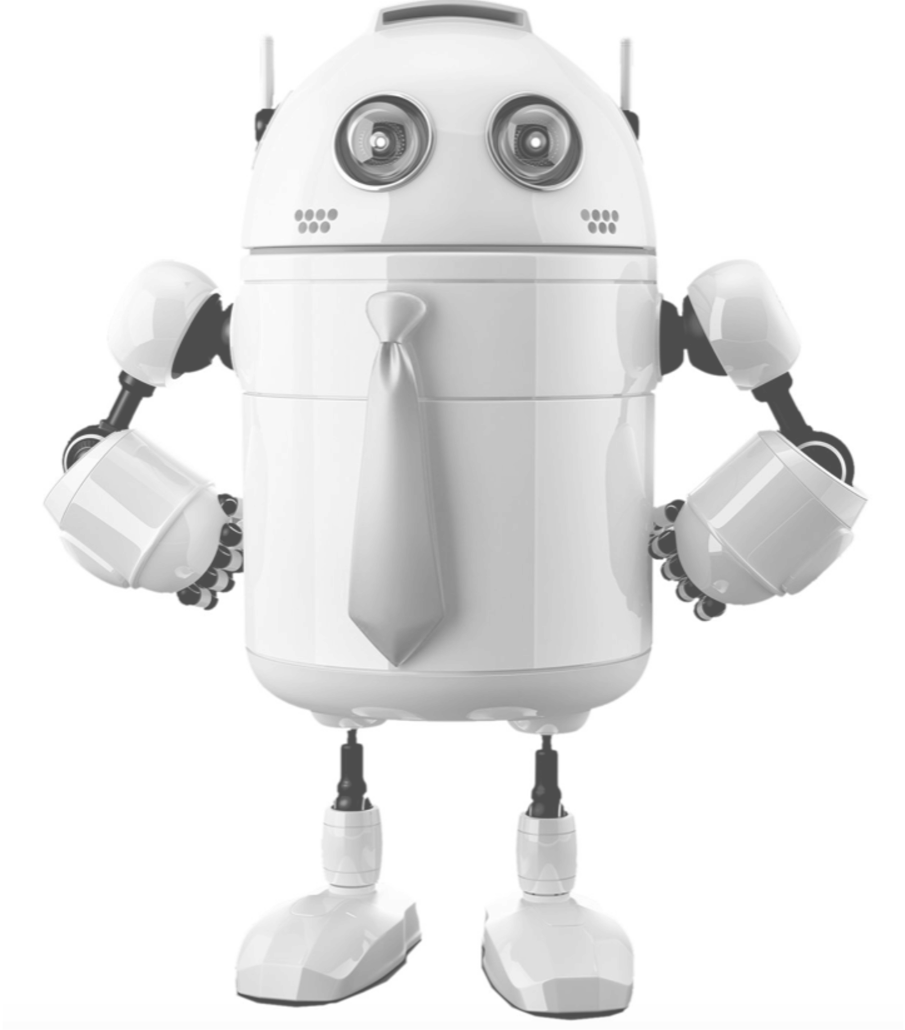
Combining VAEs & GANs



Adversarial auto-encoders:

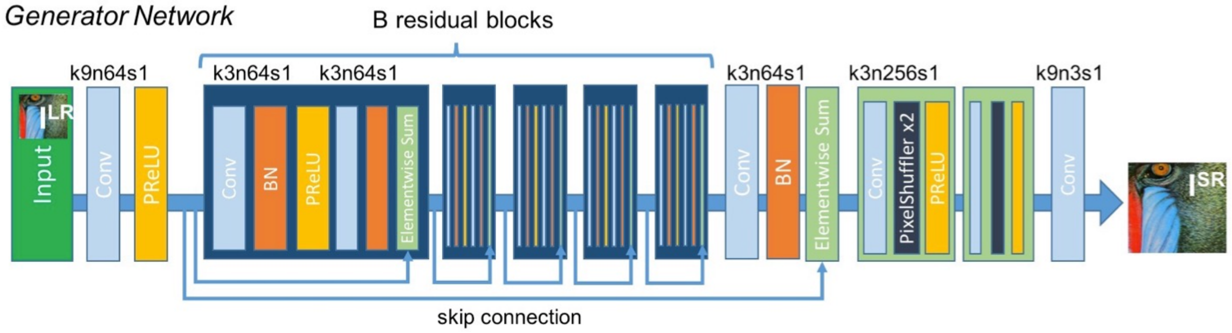
- Reconstruction loss in x space (similar to VAEs)
- Adversarial loss in z space (similar to GANs)

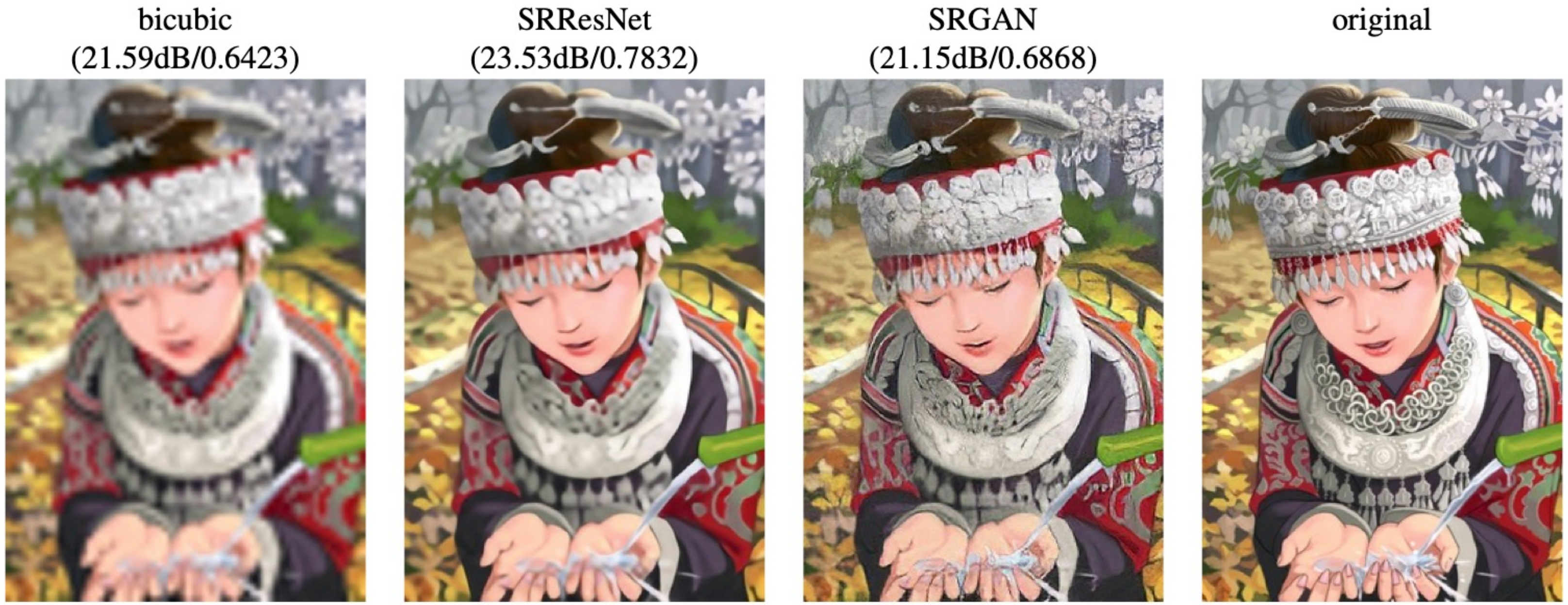
Makhzani et al. Adversarial Autoencoders. arXiv:1511.05644



Applications of Generative Models

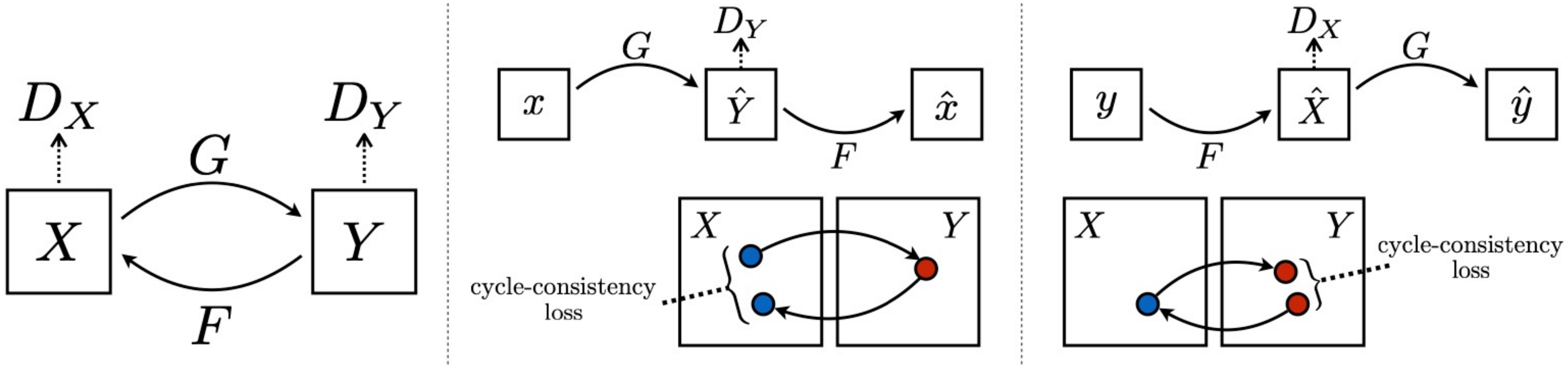
Super Resolution





Ledig et al. Photo-Realistic Single Image Super-Resolution Using a Generative Adversarial Network. CVPR 2017

Image-to-Image Translation



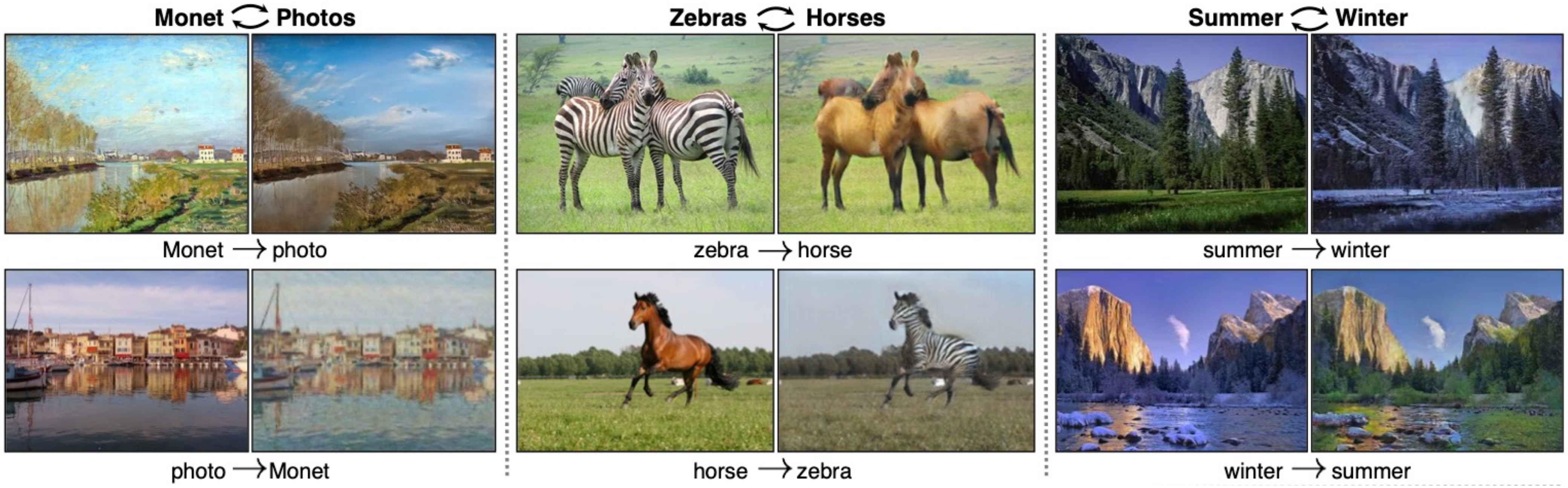
Translation between images in two domains X and Y:

- GAN training applied to generators $y = G(x)$ and $x = F(y)$
- Consistency loss applied in both direction: enforcing

$$x \approx F(G(x)), \qquad y \approx G(F(y))$$

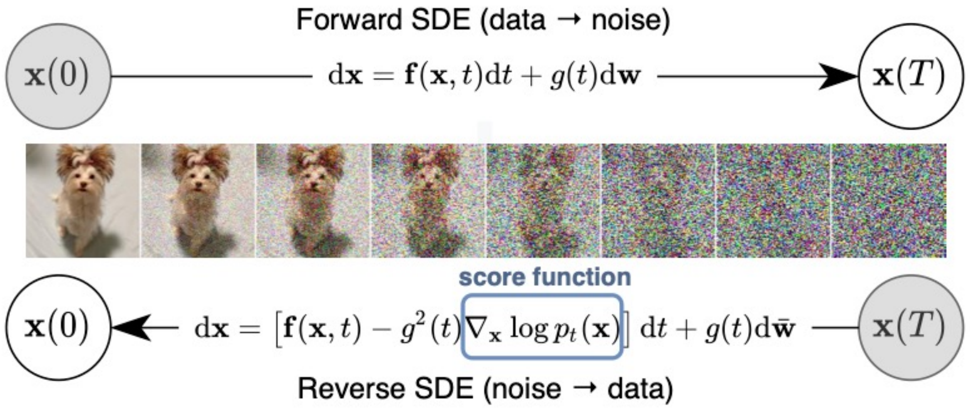
Zhu et al. Unpaired Image-to-Image Translation using Cycle-Consistent Adversarial Networks. ICCV 2017

Image-to-Image Translation

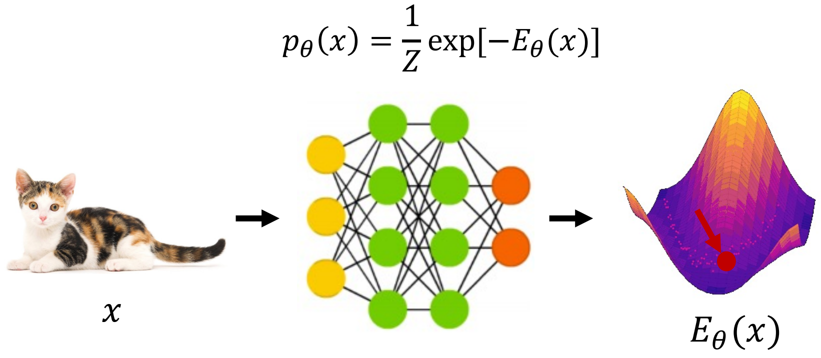


Zhu et al. Unpaired Image-to-Image Translation using Cycle-Consistent Adversarial Networks. ICCV 2017

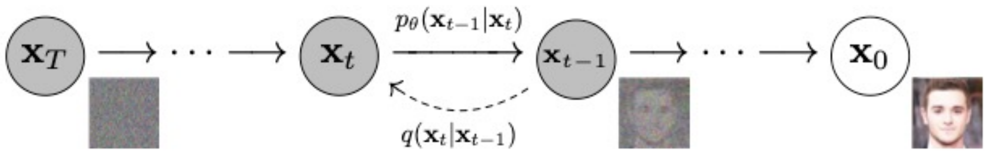
Other types of generative models



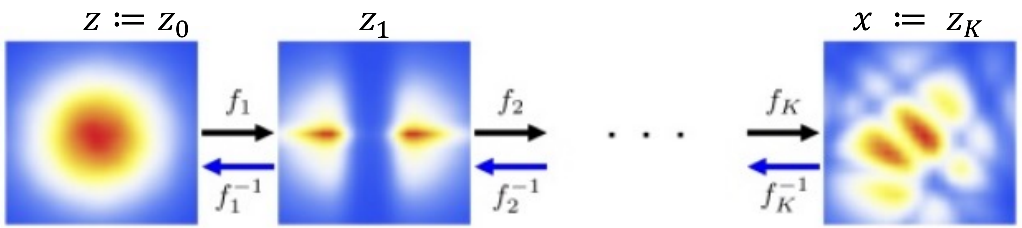
Score-based generative models



Energy-based models



Denoising diffusion probabilistic models
(as a hierarchical latent variable model)



(discrete-time) normalising flow

Song et al. Score-based Generative Modeling Through Stochastic Differential Equations. ICLR 2021
Ho et al. Denoising Diffusion Probabilistic Models. NeurIPS 2020
Du and Mordatch. Implicit Generation and Generalization with Energy Based Models. NeurIPS 2019
Rezende and Mohamed. Variational Inference with Normalizing Flows. ICML 2015

Model design in practice

- “Algorithms/paradigms” vs “network architecture”
  - VAE/GAN/flow/EBM/SGM as “algorithms/modelling paradigms”
  - MLP/CNN/Transformer as “network architecture”

Model design in practice

- On choosing a modelling paradigm (e.g. VAE or GAN or EBM…)
  - Depending on the specific application
    - GAN is often preferred for better visual quality (VAEs & others are catching up)
    - VAE, flow, etc. preferred for applications that need good likelihood estimates
      - E.g. neural data compression
  - Practical solutions are often a mix of many paradigms

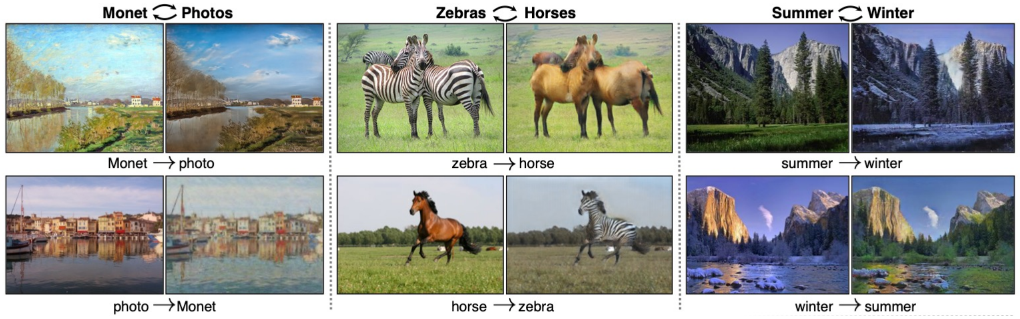
Latest generation results from diffusion models

learned compression

JPEG2000 compression

Dhariwal and Nichol. Diffusion Models Beat GANs on Image Synthesis. NeurIPS 2021
Mentzer et al. Conditional Probability Models for Deep Image Compression. CVPR 2018

Applications of Generative Models

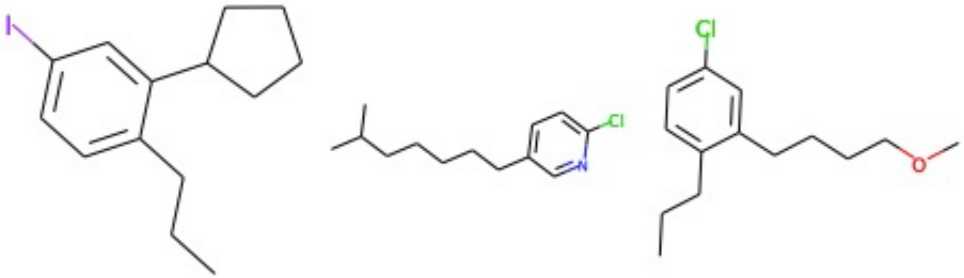


computer vision

speech synthesis

Human: Hey, how are you doing?
AI: I'm good! What would you like to chat about?
Human: Have you done any coding with Python?
AI: Yep! I've done a lot of Python coding as it is my favorite language.
Human: Nice, it's my favorite too! What have you done with it?
AI: Oh, a lot. I have done everything from creating Python modules to creating micro-websites with it.
Human: What web framework do you use?
AI: I used to use Django, but it was too difficult for me. I have found Flask is a better option as it's lighter and simpler.
Human: Wow, I agree 100%. Have you used WebSocket at all?
AI: Yes, I actually have written a chatbot using it!
Human:

natural language generation



molecule structure generation

Deep Fake

Concerns emerge as generative models improve:

- Generated visuals are getting very photo-realistic
- Has been used in fraud and scam videos
- Regulations and detection techniques needed

Channel 4's deep fake queen speech, Dec 25th, 2020

Disclaimer: I do NOT endorse this deep fake broadcast.