UNIVERSITAT POLITÈCNI
BARCELONATECH
School of Professional & Exe

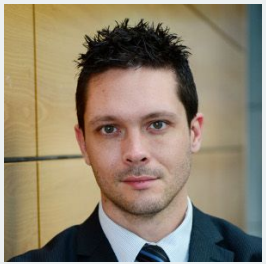
POSTGRADUATE COURSE

ARTIFICIAL INTELLIGENCE
WITH DEEP LEARNING

WWW.TALENT.UPC.EDU

DCU
Ollscoil Chathair
Bhaile Átha Cliath
Dublin City University

Generative Neural Models



Kevin McGuinness
kevin.mcguinness@dcu.ie

Assistant Professor
School of Electronic Engineering
Dublin City University

Overview

Generative models

Generative adversarial networks

Colab: MNIST digit generation with a GAN

Issues and important variants

Conditional GANs

Variational autoencoders (VAE)

Colab: convolutional VAE on MNIST

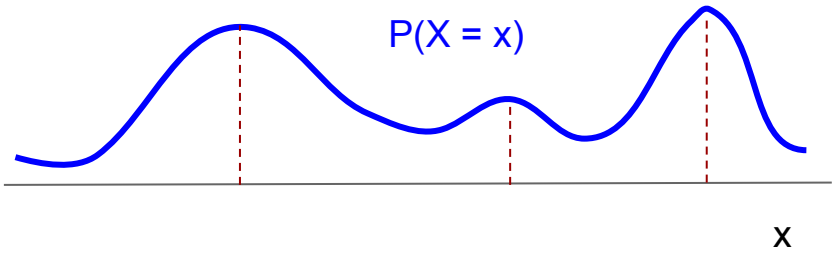
What is a generative model?

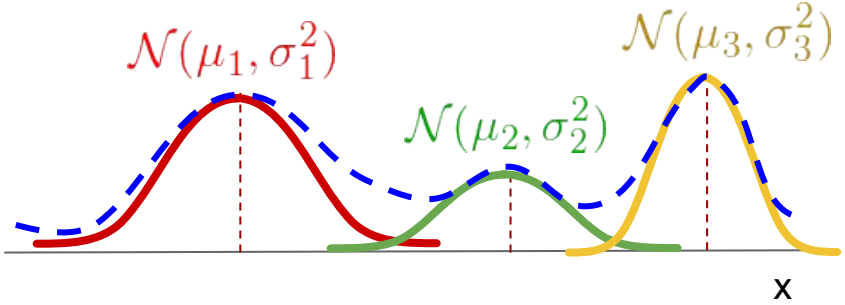
A model P(X; Θ) that we can draw samples from.

E.g. **A Gaussian Mixture Model**

- Fitting: EM algorithm
- Drawing samples:
  - Draw sample from categorical distribution to select Gaussian
  - Draw sample from Gaussian

GMMs are not generally complex enough to draw samples of images from.





$$P(X) = \lambda_1 \mathcal{N}(\mu_1, \sigma_1^2) + \lambda_2 \mathcal{N}(\mu_2, \sigma_2^2) + \ldots$$

Why are generative models important?

- Model the probability density of images
- Understanding P(X) may help us understand P(Y | X)
- Generate novel content
- Generate training data for discriminative networks
- Artistic applications
- Image completion
- Monte-carlo estimators

Overview

Generative models

Generative adversarial networks

Colab: MNIST digit generation with a GAN

Issues and important variants

Conditional GANs

Variational autoencoders (VAE)

Colab: convolutional VAE on MNIST

Generative adversarial networks

Novel method of training deep generative models invented by Ian Goodfellow et al. in 2014

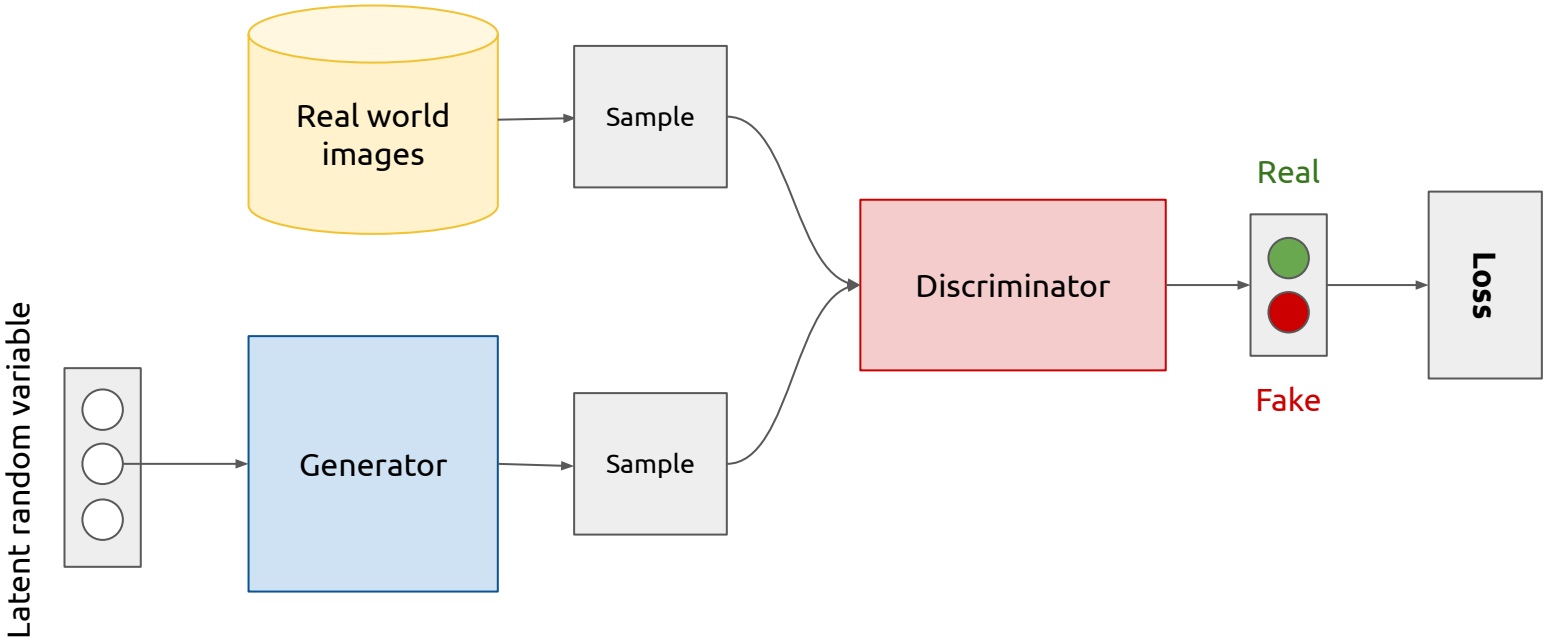
Idea: pit a generator and a discriminator against each other

- Generator tries to draw samples from P(X)
- Discriminator tries to tell if sample came from the generator or the real world

Both discriminator and generator are deep networks (differentiable functions)

Can train with backprop: train discriminator for a while, then train generator, then discriminator, ...

Generative adversarial networks (conceptual)

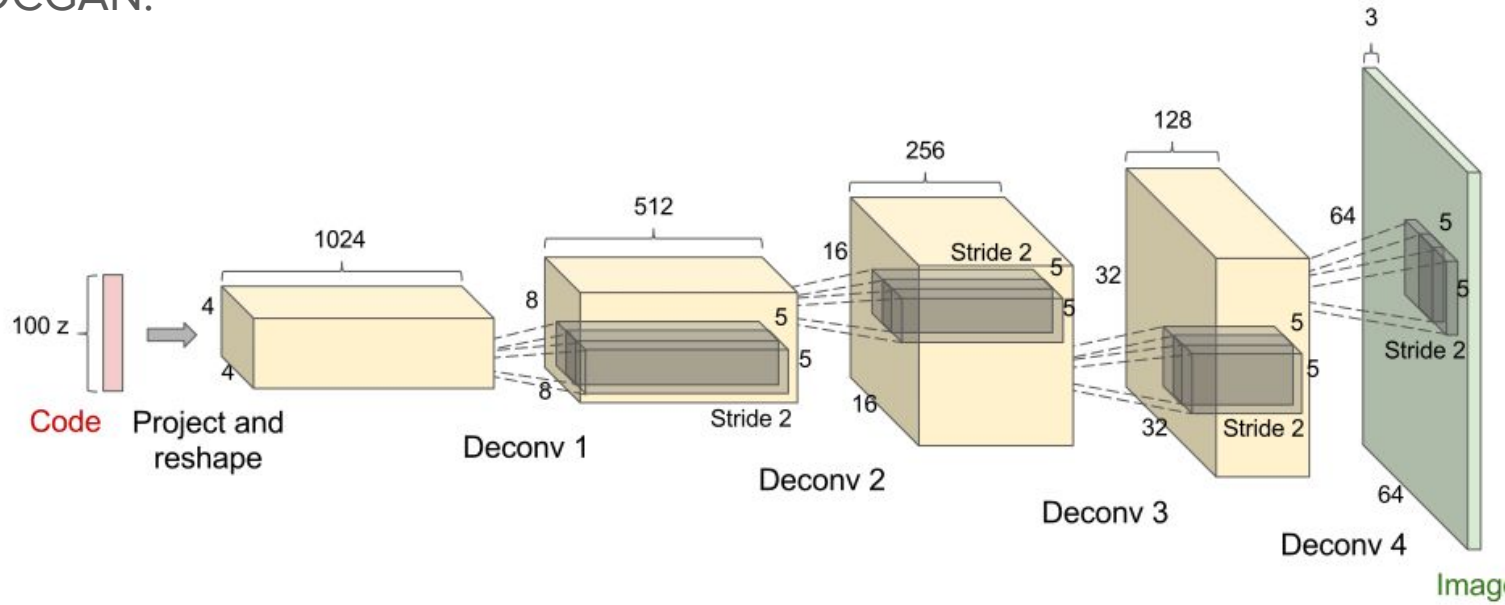


The generator

Deterministic mapping from a latent random vector to sample from q(x) ~ p(x)

Usually a deep neural network.

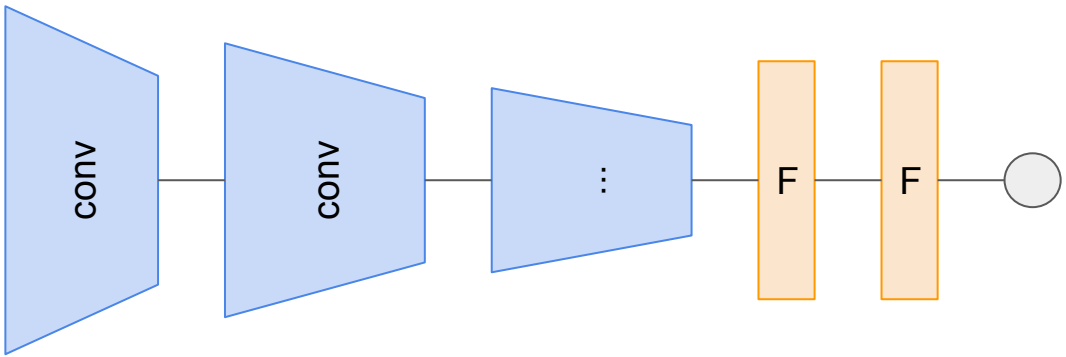
E.g. DCGAN:



The discriminator

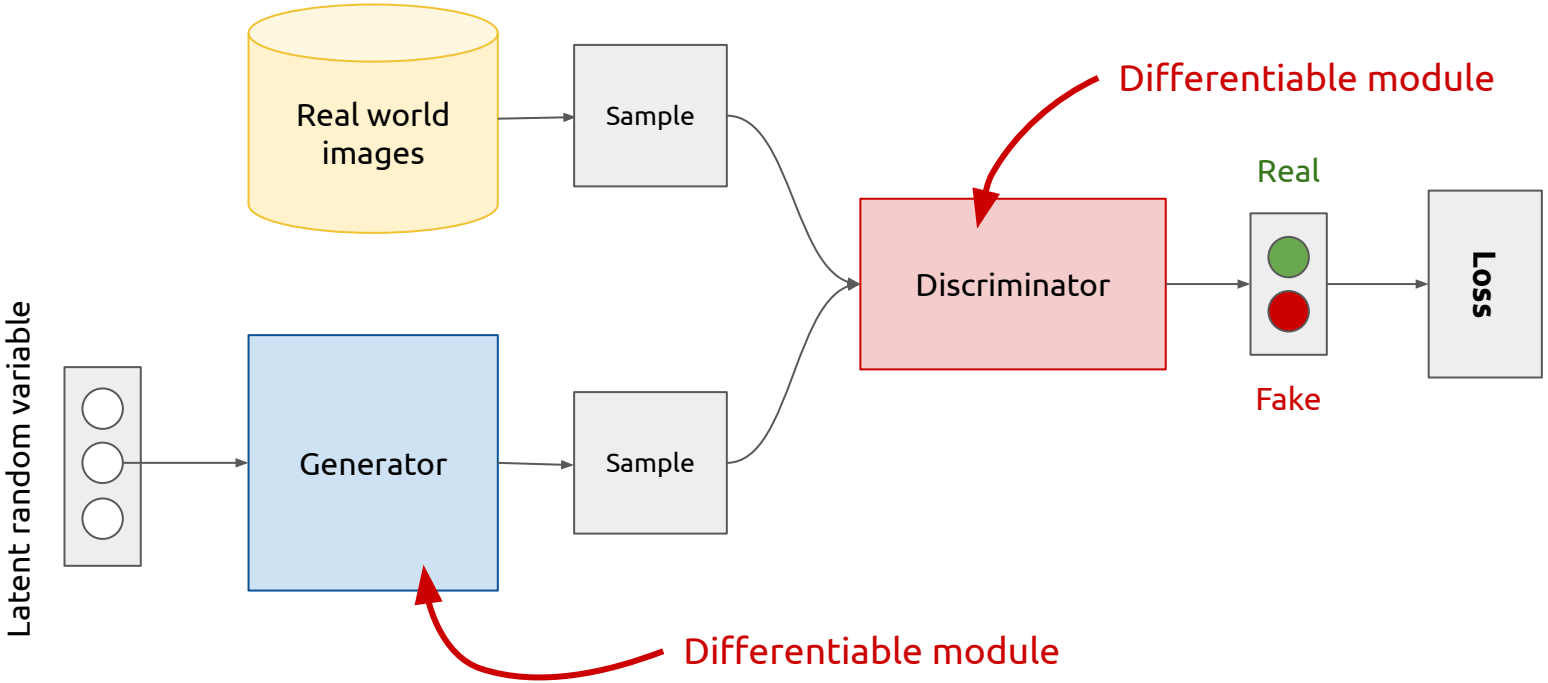
Parameterised function that tries to distinguish between samples from real images p(x) and generated ones q(x).

Usually a deep convolutional neural network.

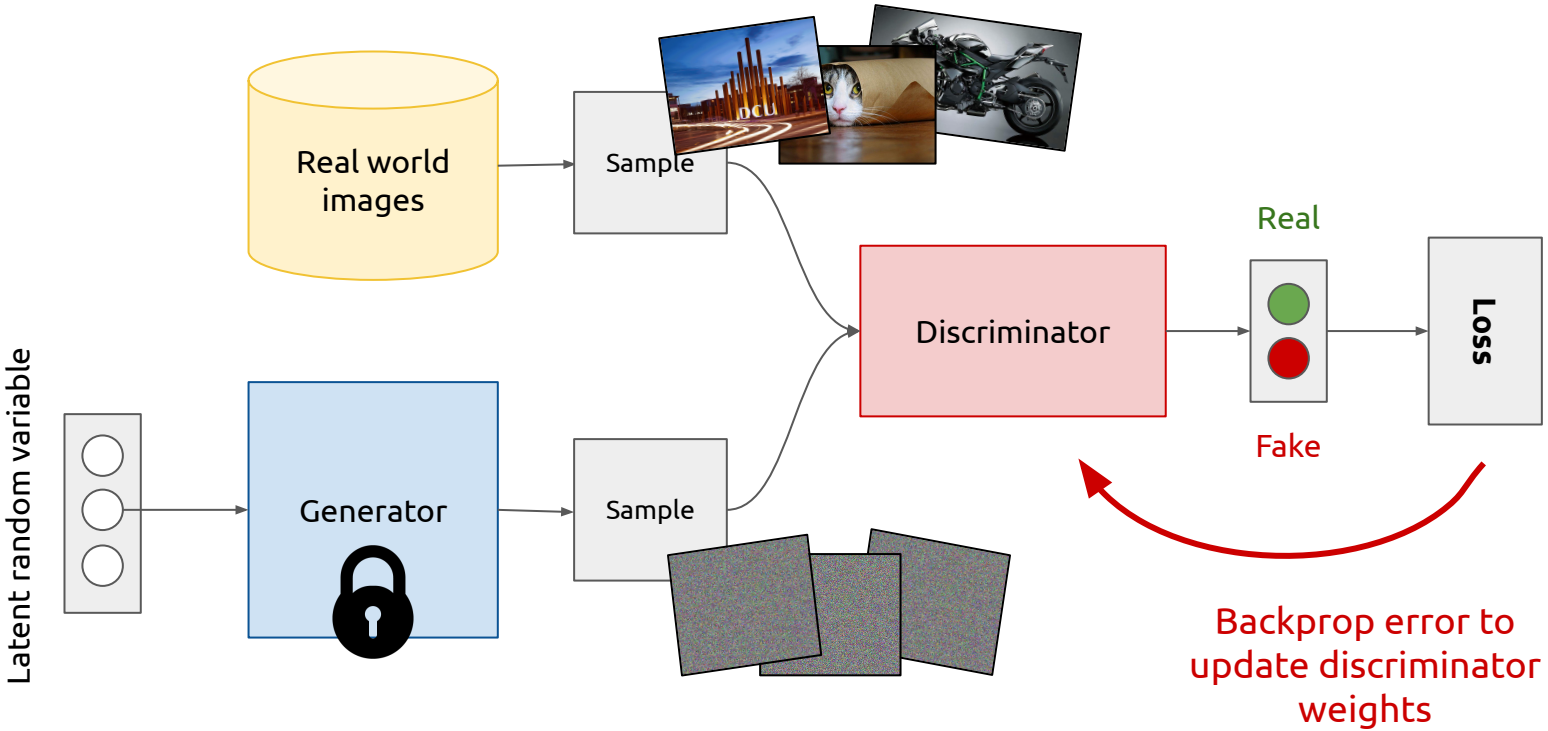


Training GANs

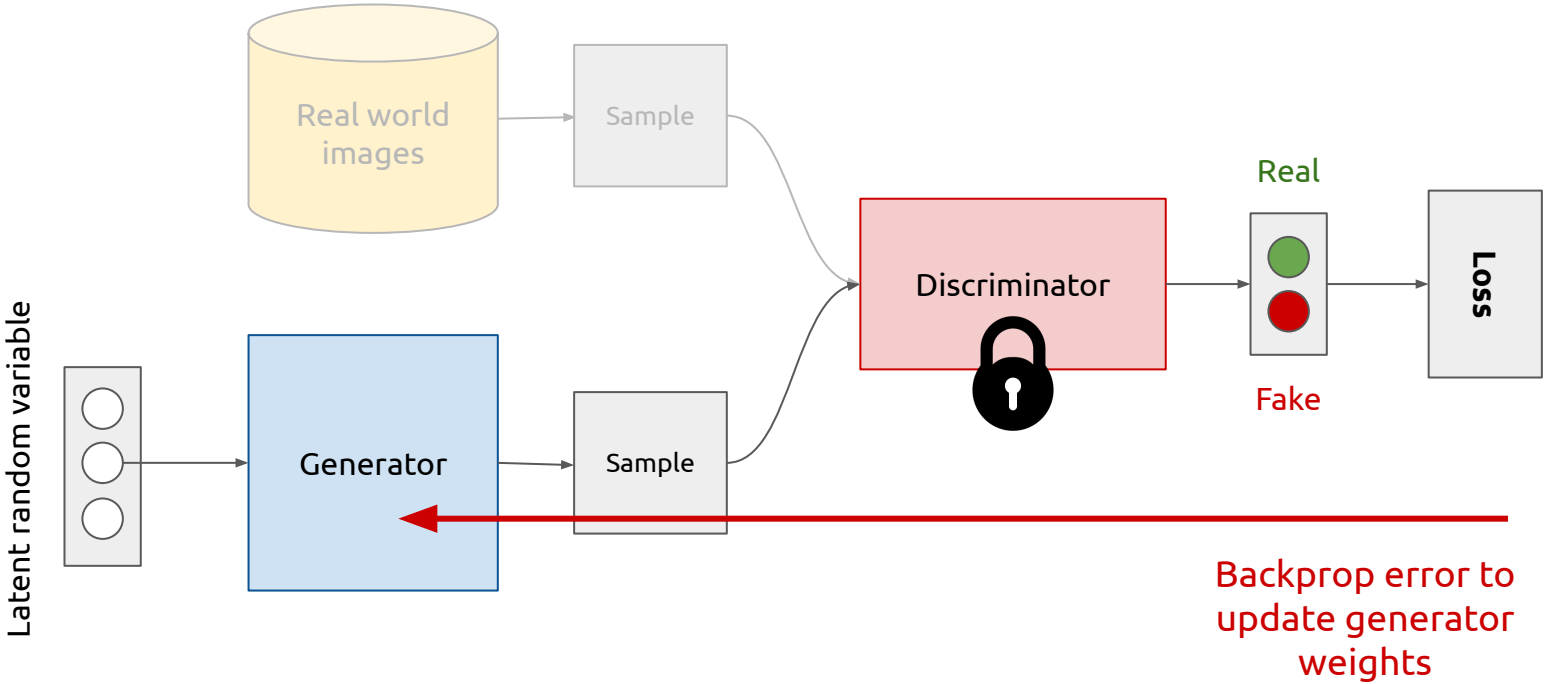
Alternate between training the discriminator and generator



1. Fix generator weights, draw samples from both real world and generated images
2. Train discriminator to distinguish between real world and generated images



1. Fix discriminator weights
2. Sample from generator
3. Backprop error through discriminator to update generator weights

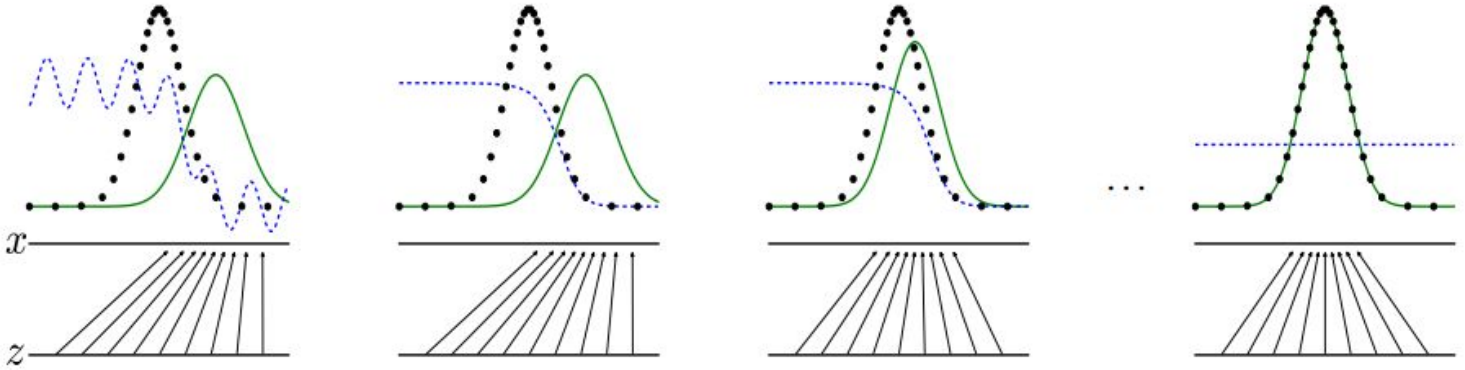


Training GANs

Iterate these two steps until convergence (which may not happen)

- Updating the discriminator should make it better at discriminating between real images and generated ones (**discriminator improves**)
- Updating the generator makes it better at fooling the current discriminator (**generator improves**)

Eventually (we hope) that the generator gets so good that it is impossible for the discriminator to tell the difference between real and generated images. Discriminator accuracy = 0.5



GAN objective

$$V(D, G) = \mathbb{E}_{x \sim p_x}[\log D(x)] + \mathbb{E}_{z \sim p_z}[\log(1 - D(G(z)))]$$

objective

$$\min_G \max_D V(D, G)$$

Algorithm 1 Minibatch stochastic gradient descent training of generative adversarial nets. The number of steps to apply to the discriminator, k, is a hyperparameter. We used $k = 1$, the least expensive option, in our experiments.

for number of training iterations **do**

Discriminator training

for k steps **do**

- Sample minibatch of m noise samples $\{\boldsymbol{z}^{(1)}, \ldots, \boldsymbol{z}^{(m)}\}$ from noise prior $p_g(\boldsymbol{z})$.
- Sample minibatch of m examples $\{\boldsymbol{x}^{(1)}, \ldots, \boldsymbol{x}^{(m)}\}$ from data generating distribution $p_{\text{data}}(\boldsymbol{x})$.
- Update the discriminator by ascending its stochastic gradient:

$$\nabla_{\theta_d} \frac{1}{m} \sum_{i=1}^{m} \left[\log D\left(\boldsymbol{x}^{(i)}\right) + \log\left(1 - D\left(G\left(\boldsymbol{z}^{(i)}\right)\right)\right) \right].$$

end for

Generator training

- Sample minibatch of m noise samples $\{\boldsymbol{z}^{(1)}, \ldots, \boldsymbol{z}^{(m)}\}$ from noise prior $p_g(\boldsymbol{z})$.
- Update the generator by descending its stochastic gradient:

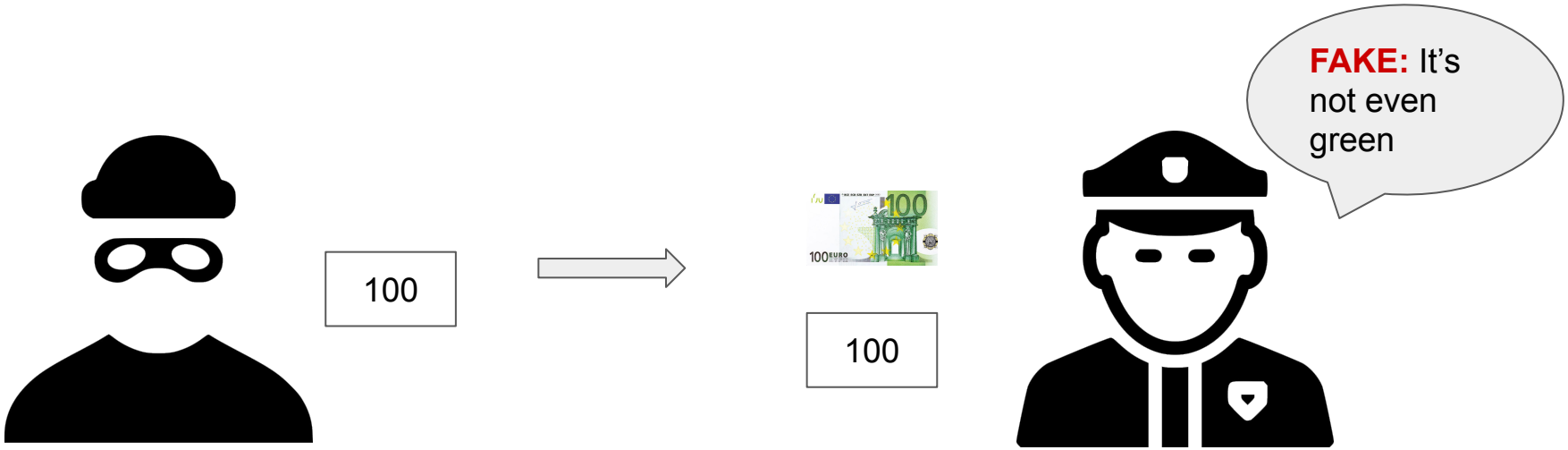
$$\nabla_{\theta_g} \frac{1}{m} \sum_{i=1}^{m} \log\left(1 - D\left(G\left(\boldsymbol{z}^{(i)}\right)\right)\right).$$

end for

The gradient-based updates can use any standard gradient-based learning rule. We used momentum in our experiments.

Adversarial training analogy: is it fake money?

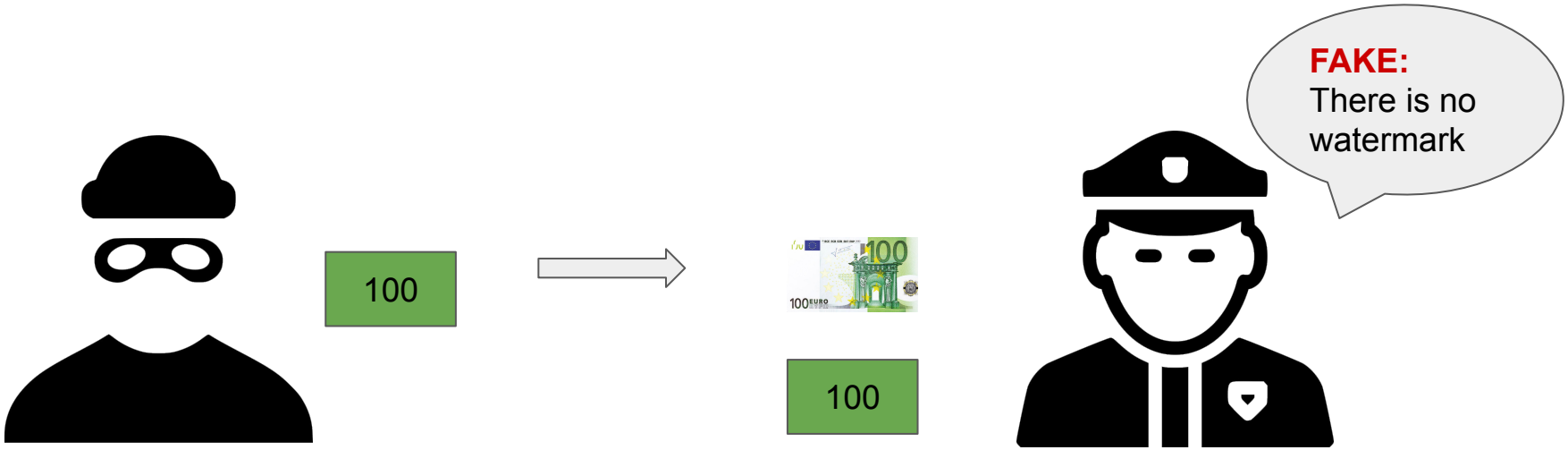
Imagine we have a counterfeiter (**G**) trying to make fake money, and the police (**D**) has to detect whether money is real or fake.



Slide credit: [Santiago Pascual de la Puente]

Adversarial training analogy: is it fake money?

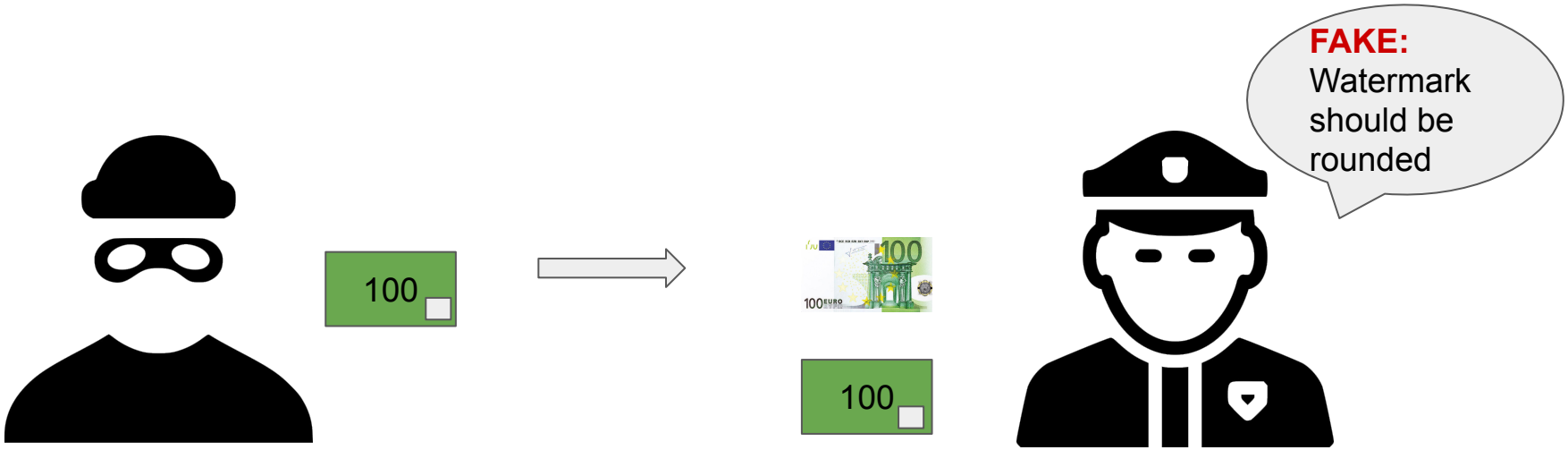
Imagine we have a counterfeiter (**G**) trying to make fake money, and the police (**D**) has to detect whether money is real or fake.



Slide credit: [Santiago Pascual de la Puente]

Adversarial training analogy: is it fake money?

Imagine we have a counterfeiter (**G**) trying to make fake money, and the police (**D**) has to detect whether money is real or fake.

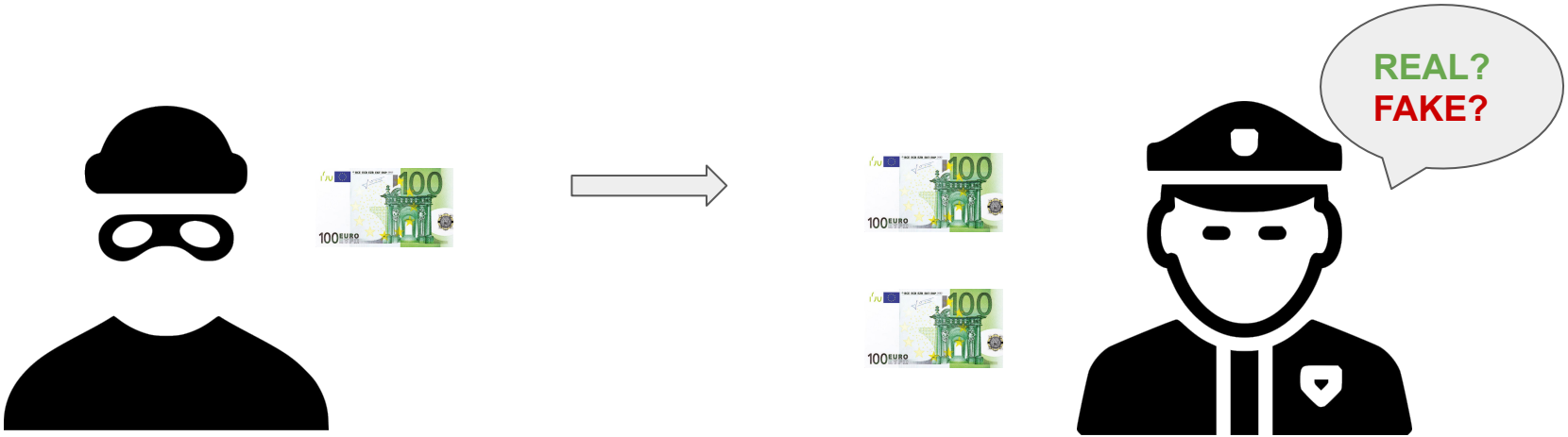


Slide credit: Santiago Pascual de la Puente

Adversarial training analogy: is it fake money?

Imagine we have a counterfeiter (**G**) trying to make fake money, and the police (**D**) has to detect whether money is real or fake.

After sufficient iterations, and if the counterfeiter is good enough, the police should be unable to tell the difference between real and fake money.



Overview

Generative models

Generative adversarial networks

Colab: MNIST digit generation with a GAN

Issues and important variants

Conditional GANs

Variational autoencoders (VAE)

Colab: convolutional VAE on MNIST

MNIST Digit Generation with a GAN

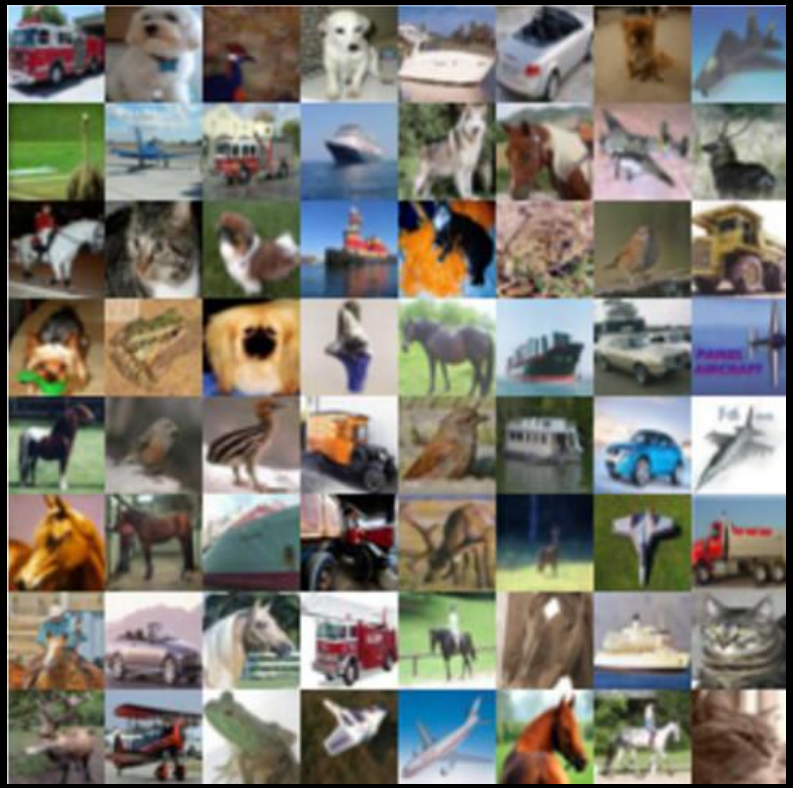
Some examples of generated images...

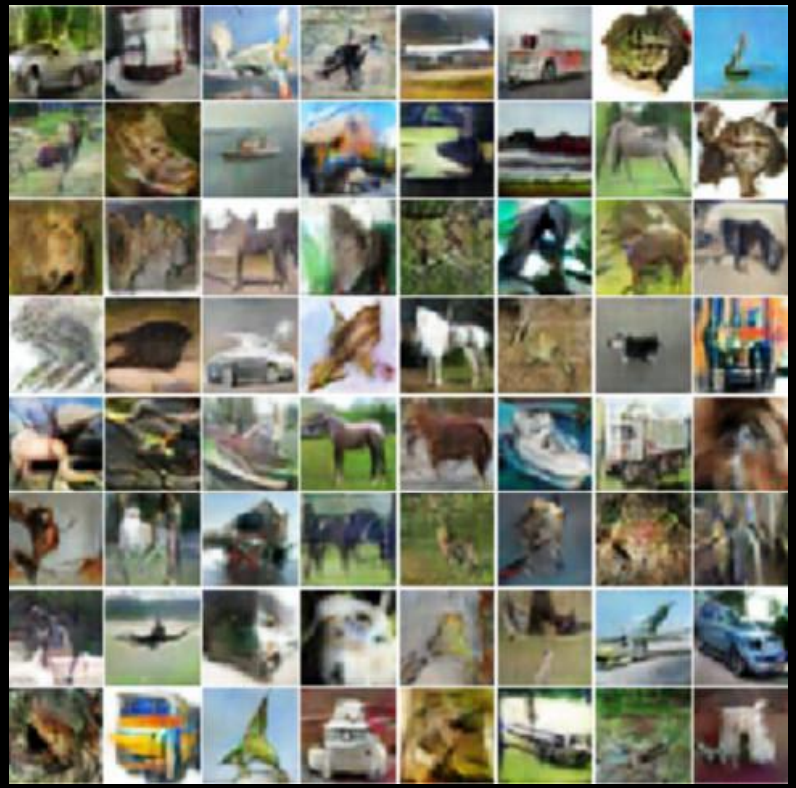
ImageNet

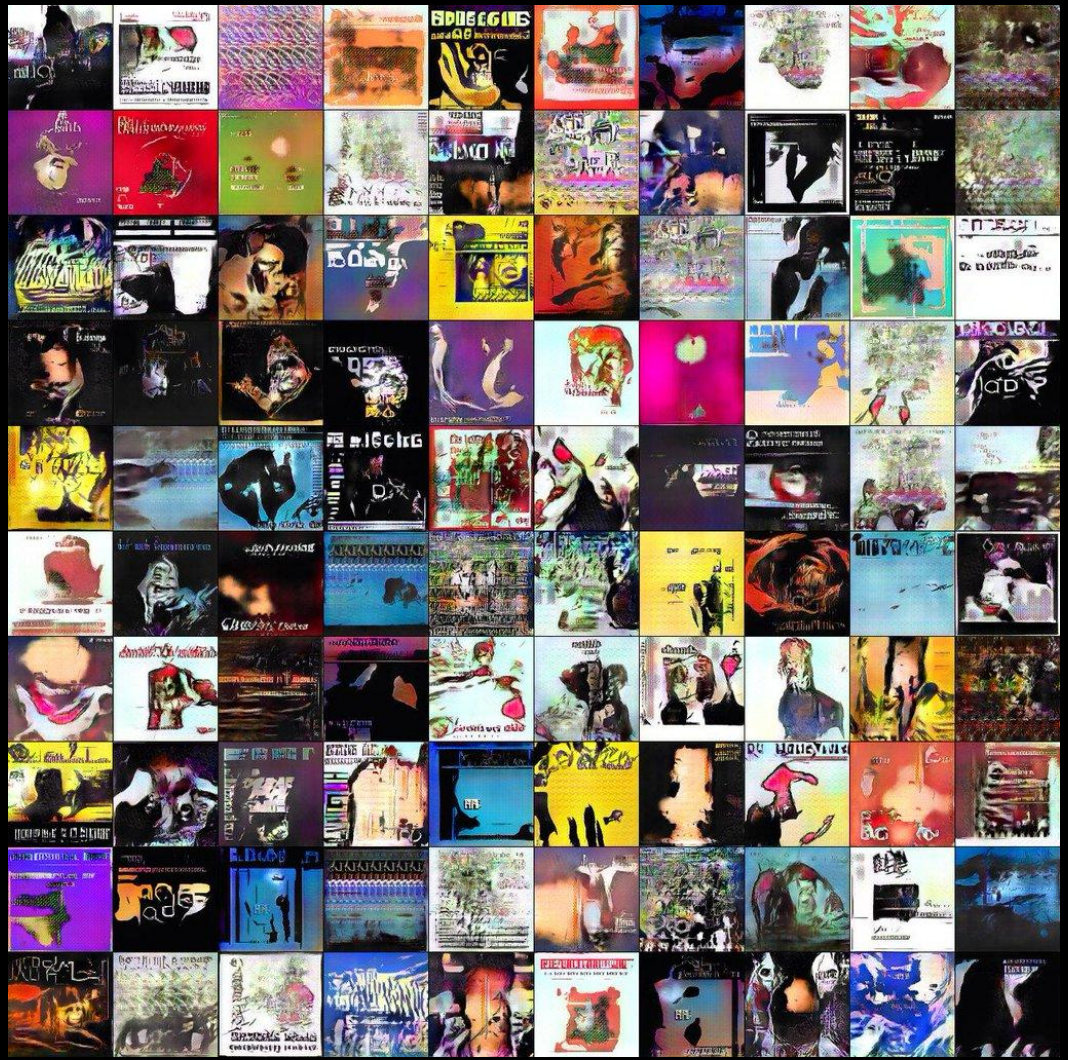
Source:
https://openai.com/blog/generative-models/

CIFAR-10

Source
https://openai.com/blog/generative-models/







Credit:
Alec Radford

Code on GitHub

Credit: Alec Radford Code on GitHub

Overview

Generative models

Generative adversarial networks

Colab: MNIST digit generation with a GAN

Issues and important variants

Conditional GANs

Variational autoencoders (VAE)

Colab: convolutional VAE on MNIST

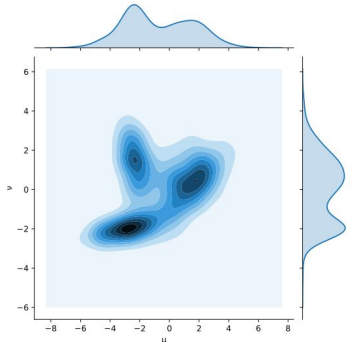
Issues

Known to be very difficult to train:

- Formulated as a “game” between two networks
- **Unstable** dynamics: hard to keep generator and discriminator in balance
- Optimization can **oscillate** between solutions
- **Mode collapse** in the generator

Difficult to evaluate results

Important variants



Wasserstein GAN (WGAN)

- MLE leads to a KL divergence loss.
- Numerical stability issues when estimated distribution and true distribution do not overlap significantly (loss blows up).
- WGAN idea is to use a coarse approximation of the Wasserstein distance (the Earth mover's distance).
- Weight clipping is needed to enforce Lipschitz constraint.

Overall effect is to make the GAN more stable. Discriminator can be trained more on each step without blowing up.

Can work well in practice, but clipping the weights to enforce Lipschitz slows training.

WGAN objective

$$\min_{G} \max_{f \in \mathcal{F}} \mathbb{E}_{x \sim p_x}[f(x)] - \mathbb{E}_{z \sim p_z}[f(G(z))]$$

set of all K-Lipschitz functions

$$\mathcal{W} = [-0.01, 0.01]^D$$

Important variants

Least squares GAN (LSGAN)

- Similar motivation to WGAN: want a loss that gives nice gradients and doesn't blow up.
- LSGAN Idea: just use squared error (L^2 distance)!
- Turns out this is the same as minimizing the Pearson χ^2 divergence.

$$\min_D V_{\mathrm{LSGAN}}(D) = \frac{1}{2}\mathbb{E}_{\boldsymbol{x}\sim p_{\mathrm{data}}(\boldsymbol{x})}\left[(D(\boldsymbol{x})-1)^2\right] + \frac{1}{2}\mathbb{E}_{\boldsymbol{z}\sim p_{\boldsymbol{z}}(\boldsymbol{z})}\left[(D(G(\boldsymbol{z}))+1)^2\right]$$

$$\min_G V_{\mathrm{LSGAN}}(G) = \frac{1}{2}\mathbb{E}_{\boldsymbol{z}\sim p_{\boldsymbol{z}}(\boldsymbol{z})}\left[(D(G(\boldsymbol{z})))^2\right].$$

(a) Church outdoor.

(b) Dining room.

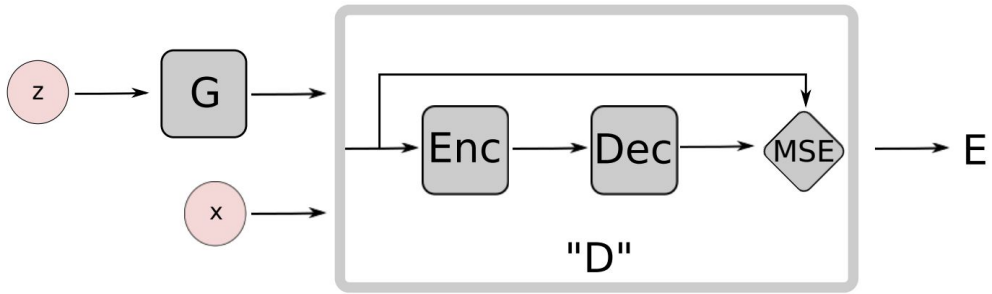
(c) Kitchen.

(d) Conference room.

Important variants

Energy-based GAN (EBGAN)

- Instead of using a binary classifier as the discriminator *D* use an energy-based model (an autoencoder)
- *D* models the image manifold since it is trained on real images
- Optimize to generate samples that have low energy
- Generator gets more *signal* from *D*



Important variants

Boundary Equilibrium GAN (BEGAN)

- Combines ideas from WGAN and EBGAN
- BEGAN idea: matching the distributions of the reconstruction losses can be a suitable proxy for matching the data distributions.
- Use Wasserstein distance approximation to do this
- Includes mechanism for automatically maintaining equilibrium

Generator variants: ProGAN

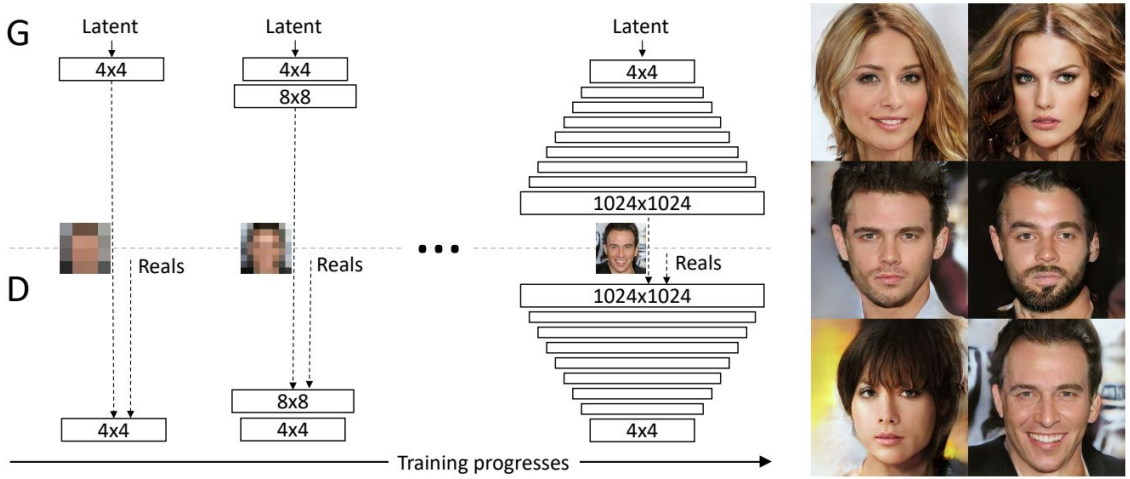


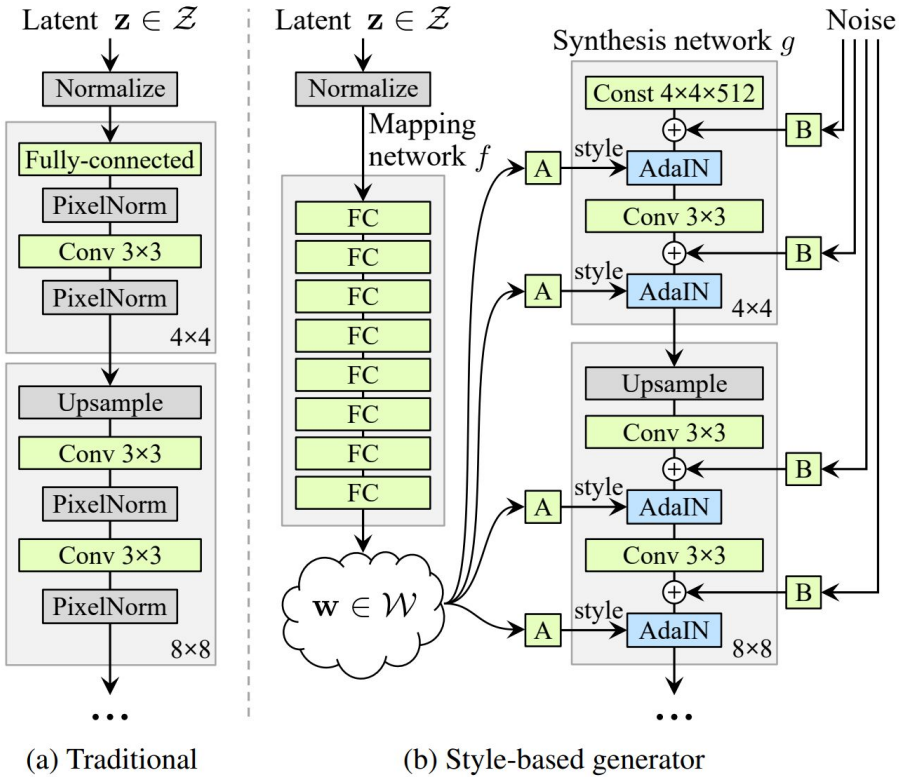
Figure 1: Our training starts with both the generator (G) and discriminator (D) having a low spatial resolution of 4×4 pixels. As the training advances, we incrementally add layers to G and D, thus increasing the spatial resolution of the generated images. All existing layers remain trainable throughout the process. Here $\boxed{N \times N}$ refers to convolutional layers operating on $N \times N$ spatial resolution. This allows stable synthesis in high resolutions and also speeds up training considerably. One the right we show six example images generated using progressive growing at 1024×1024.

Karras, Tero, et al. Progressive growing of gans for improved quality, stability, and variation. *ICLR* 2018.

Generator variants: StyleGAN

Condition each stage of the synthesis on a style embedding.

Add noise at each stage during synthesis.



Karras et al., A style-based generator architecture for generative adversarial networks. CVPR 2019

https://thispersondoesnotexist.com

Generated with StyleGAN

http://thiscatdoesnotexist.com

Overview

Generative models

Generative adversarial networks

Colab: MNIST digit generation with a GAN

Issues and important variants

Conditional GANs

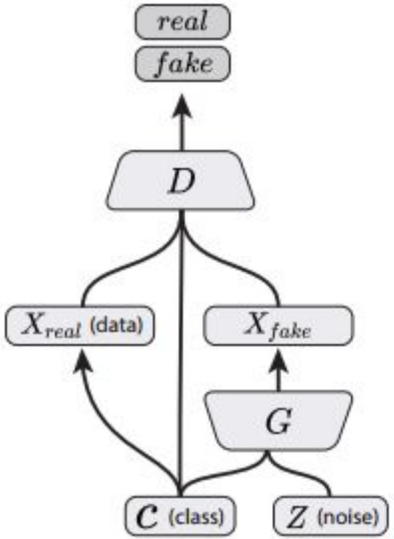
Variational autoencoders (VAE)

Colab: convolutional VAE on MNIST

Conditional GANs

GANs can be conditioned on other info: e.g. a label

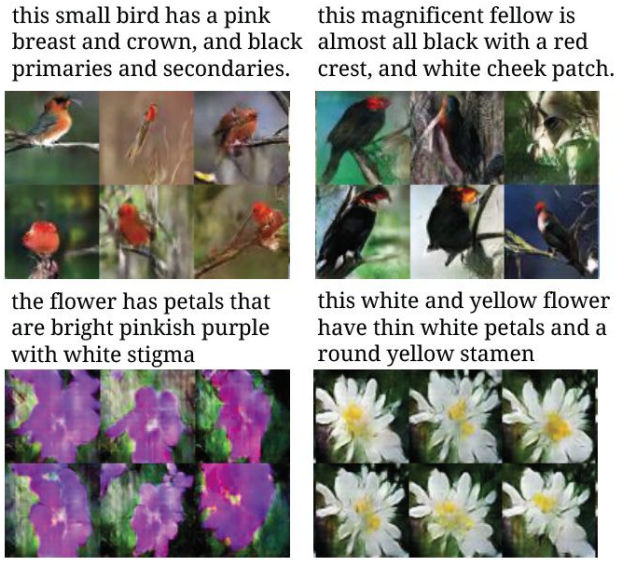
- z might capture random characteristics of the data, vari
  futures,
- c would condition the deterministic parts (label)



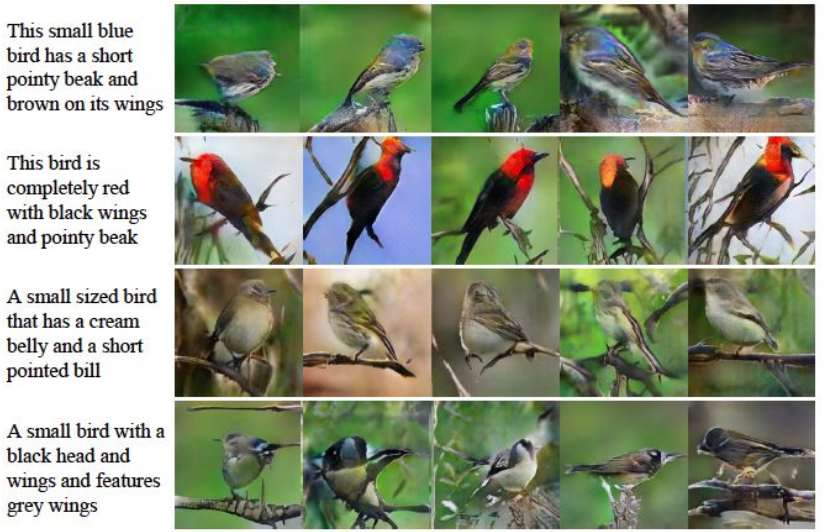
Conditional GAN
(Mirza & Osindero, 2014)

For details on ways to condition GANs: [Ways of Conditioning Generative Adversarial Networks (Wack et al.)]

Generating images conditioned on captions

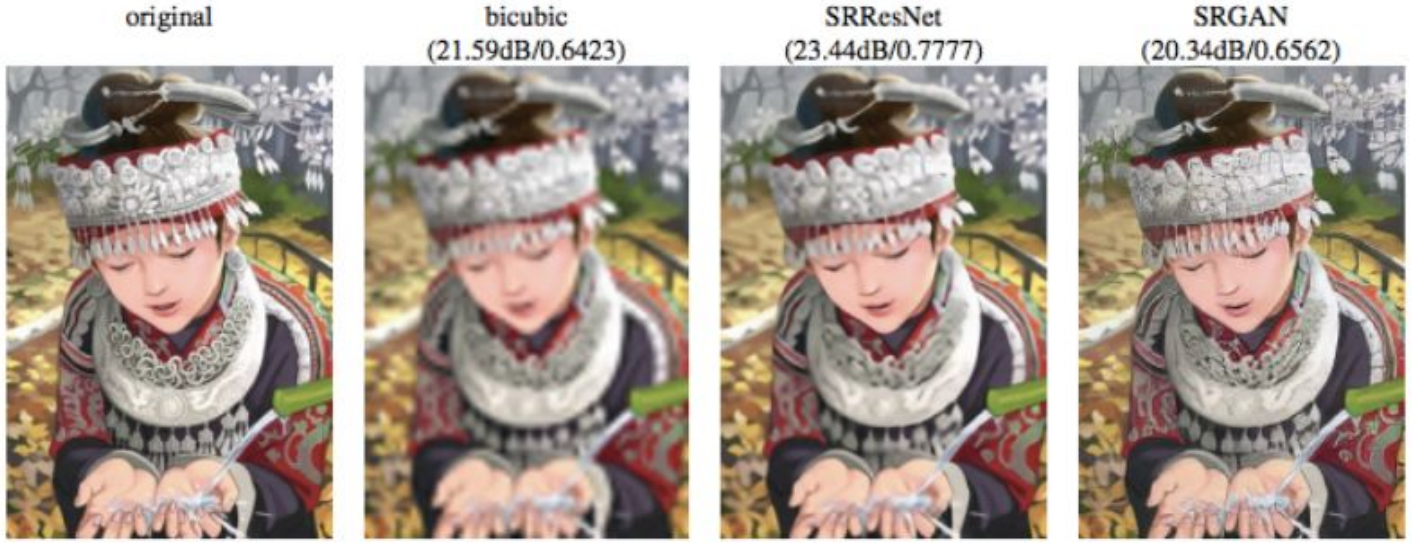


(Reed et al. 2016b)



(Zhang et al. 2016)

Image super-resolution



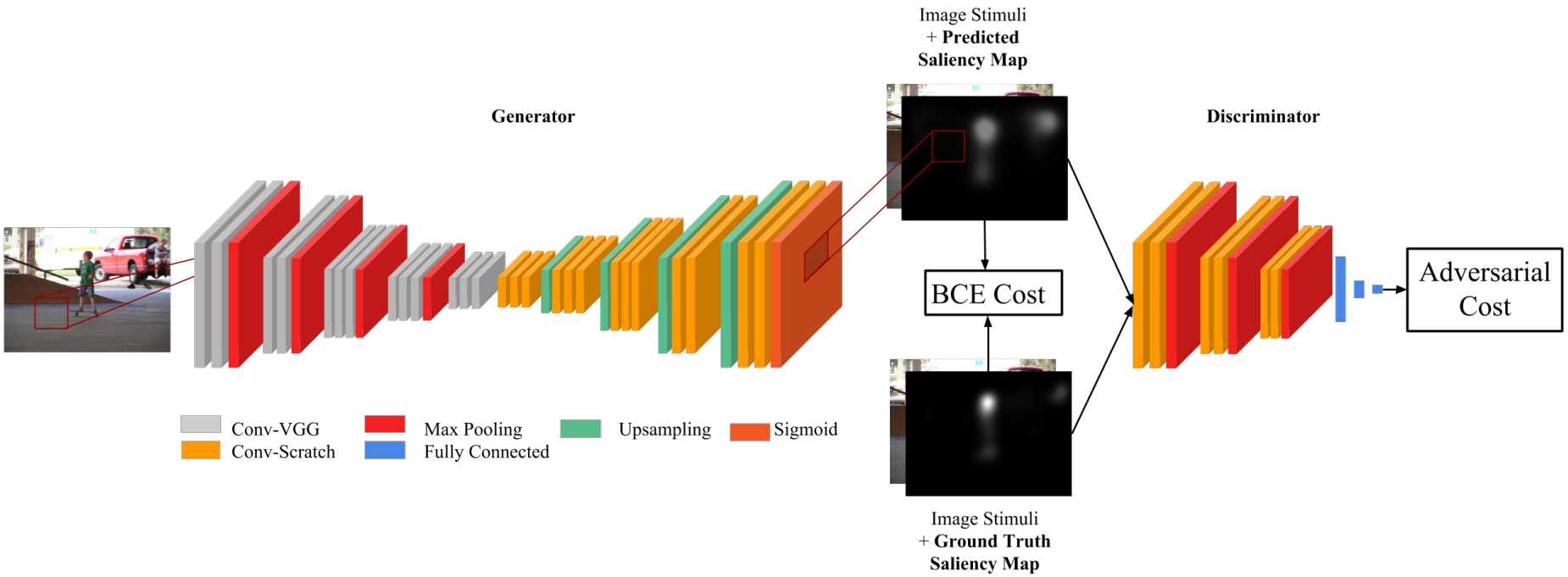
(Ledig et al. 2016)

Bicubic: not using data statistics. SRResNet: trained with MSE. SRGAN is able to understand that there are multiple correct answers, rather than averaging.

Saliency prediction

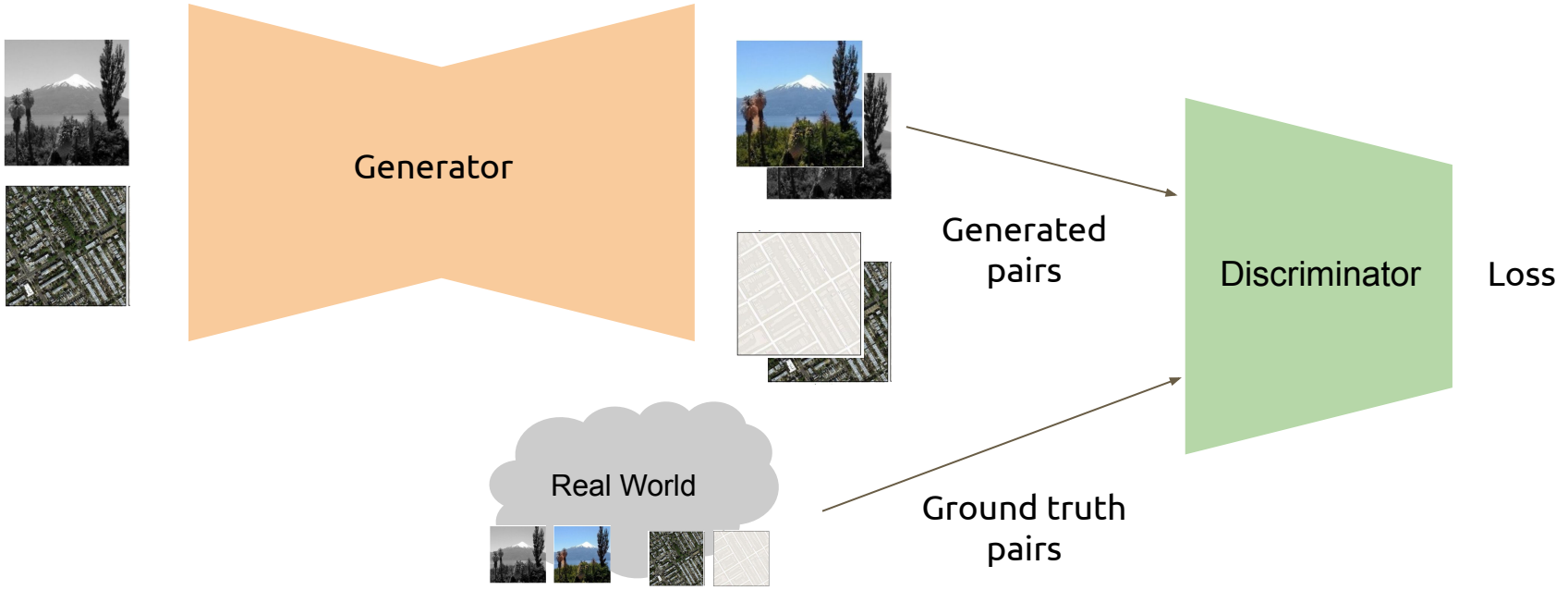
Dala loss Adversarial loss

$$\alpha \cdot \mathcal{L}_{BCE} - \log D(I, \hat{S}),$$



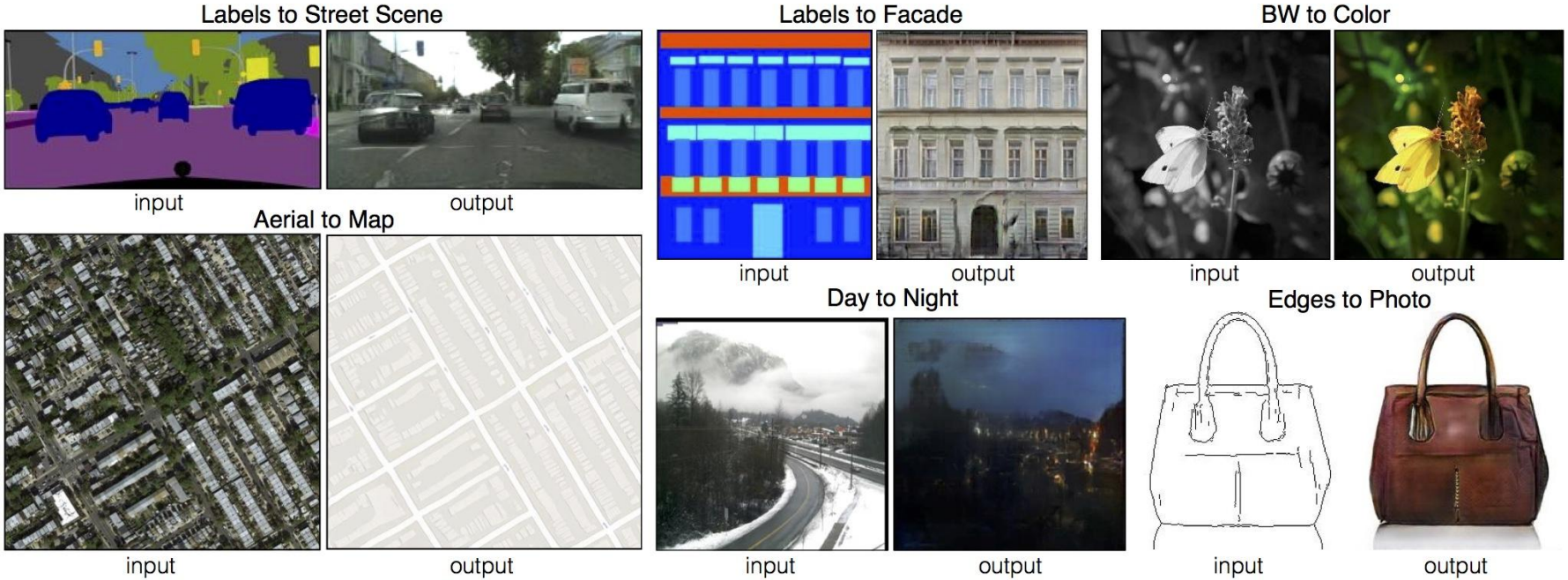
Junting Pan, Cristian Canton, Kevin McGuinness, Noel E. O'Connor, Jordi Torres, Elisa Sayrol and Xavier Giro-i-Nieto. "SalGAN: Visual Saliency Prediction with Generative Adversarial Networks." arXiv. 2017.

Image-to-Image translation (pix2pix)



Isola, Phillip, Jun-Yan Zhu, Tinghui Zhou, and Alexei A. Efros. "Image-to-image translation with conditional adversarial networks." arXiv:1611.07004 (2016).

Image-to-Image translation (pix2pix)



Isola, Phillip, Jun-Yan Zhu, Tinghui Zhou, and Alexei A. Efros. "Image-to-image translation with conditional adversarial networks." arXiv:1611.07004 (2016).

https://affinelayer.com/pixsrv/

Image-to-Image Demo

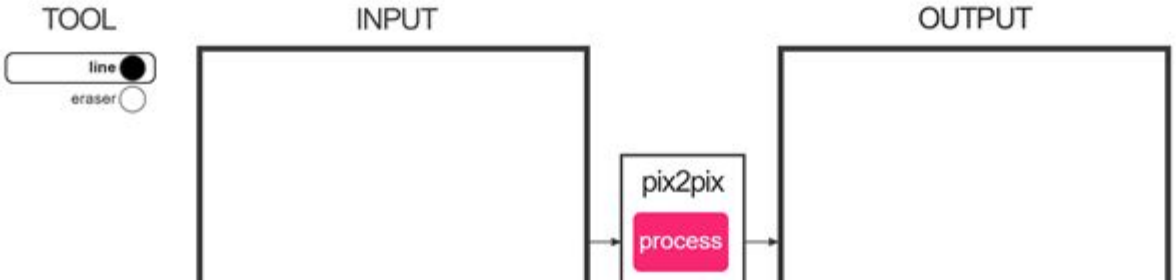
Interactive Image Translation with pix2pix-tensorflow

Written by Christopher Hesse — February 19th, 2017

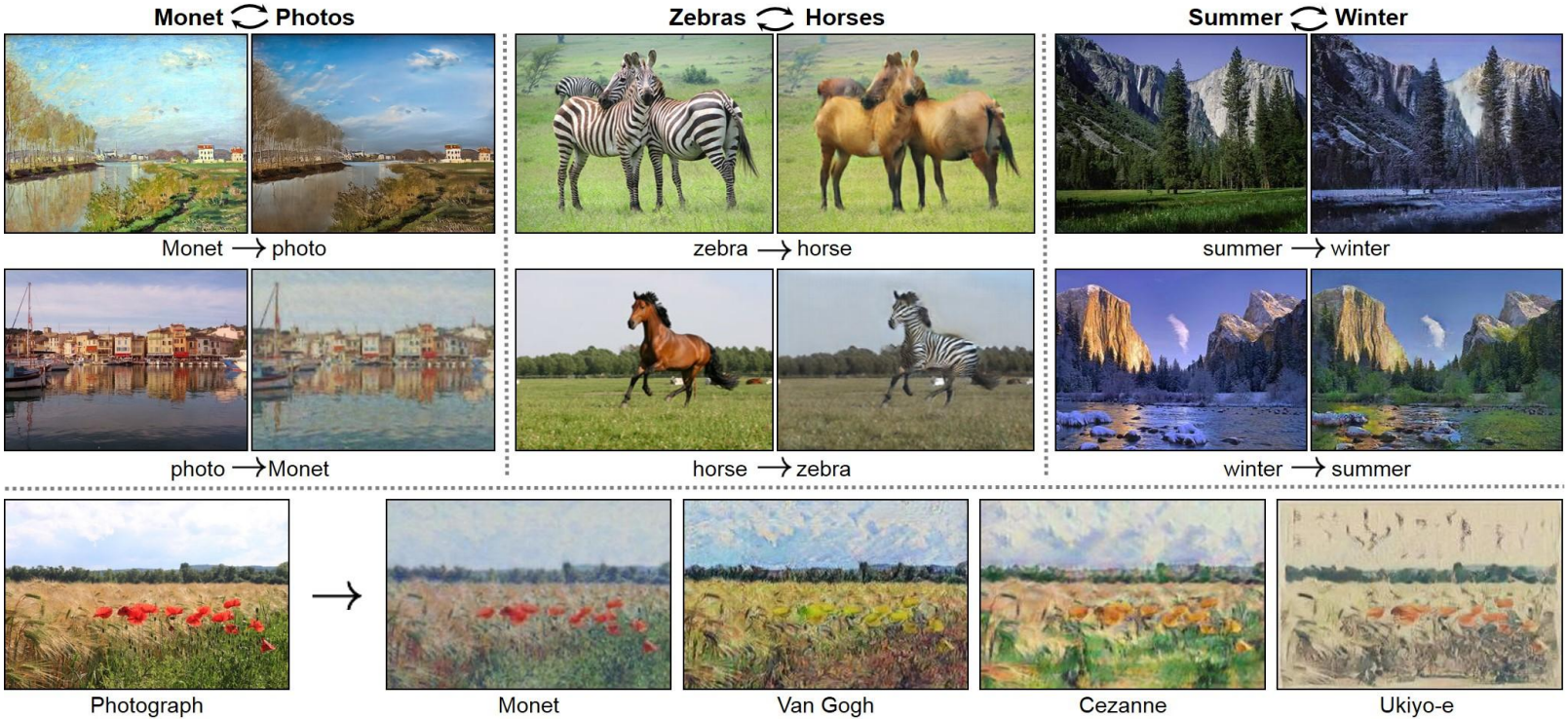
Recently, I made a Tensorflow port of pix2pix by Isola et al., covered in the article Image-to-Image Translation in Tensorflow. I've taken a few pre-trained models and made an interactive web thing for trying them out. Chrome is recommended.

The pix2pix model works by training on pairs of images such as building facade labels to building facades, and then attempts to generate the corresponding output image from any input image you give it. The idea is straight from the pix2pix paper, which is a good read.

edges2cats

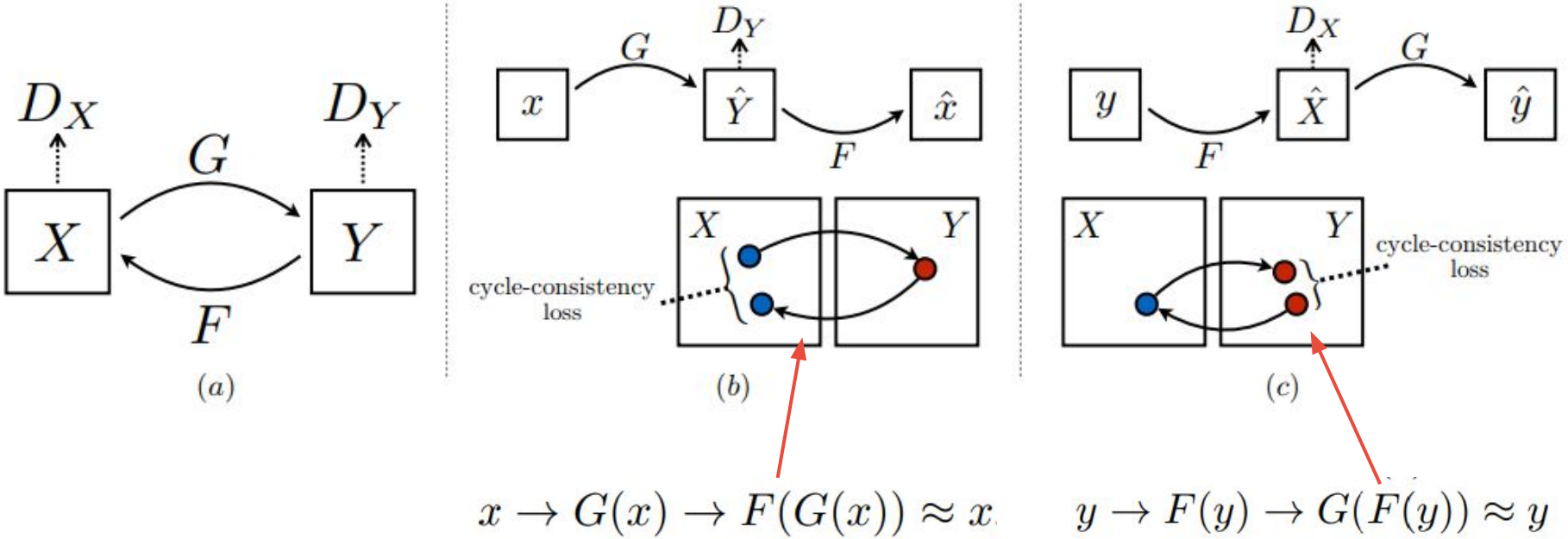


CycleGAN: Unpaired Image-to-Image Translation



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

CycleGAN: Unpaired Image-to-Image Translation



Overview

Generative models

Generative adversarial networks

Colab: MNIST digit generation with a GAN

Issues and important variants

Conditional GANs

Variational autoencoders (VAE)

Colab: convolutional VAE on MNIST

Variational Autoencoders (VAE)

Totally different idea than GAN for a generative model

Idea: modify an autoencoder to that the **representation in the bottleneck** is forced to have a **known distribution** that is easy to draw samples from (often a Gaussian)

Kingma and Welling, Auto-Encoding Variational Bayes, ICLR 2014

Variational Autoencoders (VAE)

Recall: autoencoders

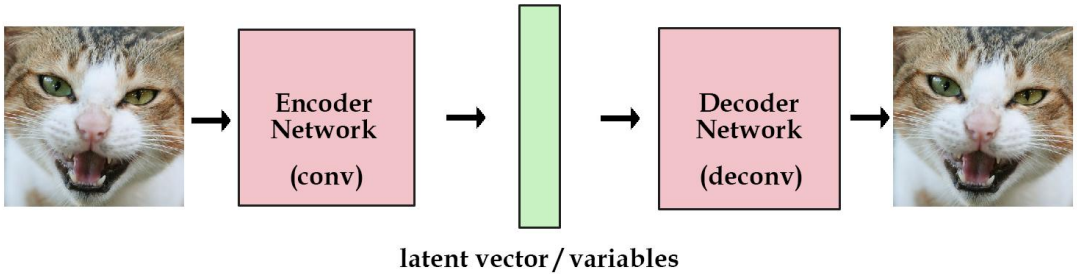
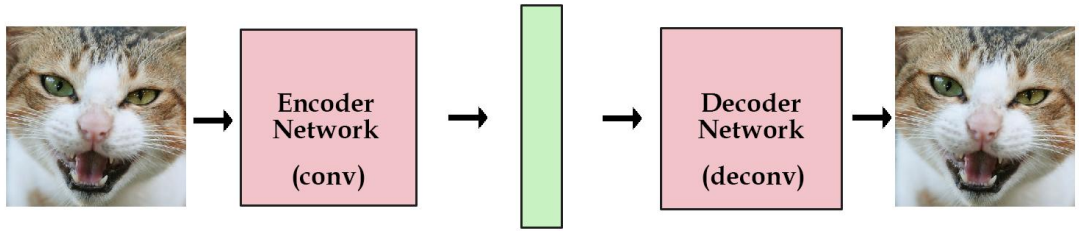


Image credit: Kevin Frans

Variational Autoencoders (VAE)

Recall: autoencoders



Question: why can't we sample from a trained autoencoder by just inserting random values into the bottleneck and running the decoder?

Image credit: Kevin Frans

Variational Autoencoders (VAE)

Recall: autoencoders

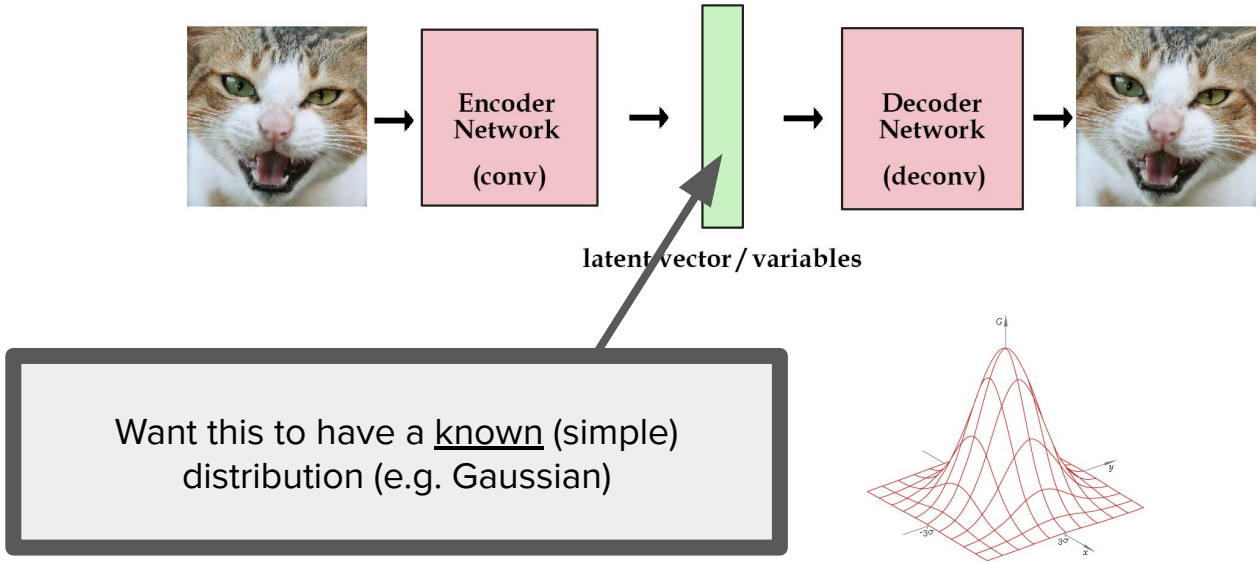


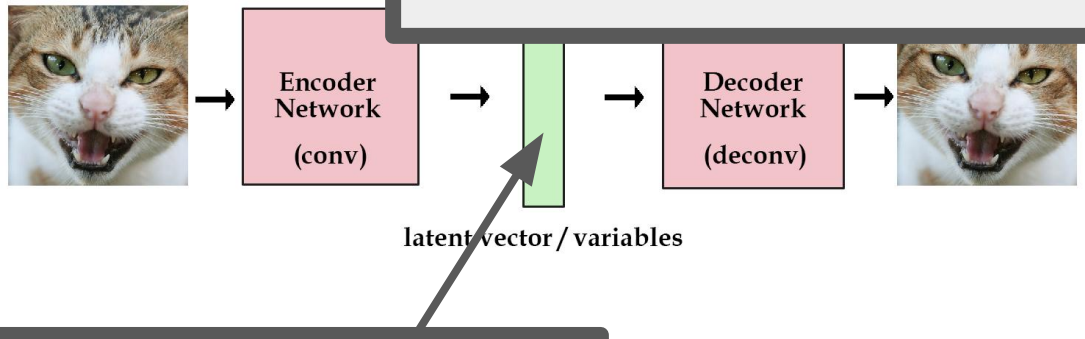
Image credit: Kevin Frans

Variational Autoencod

Recall: autoencoders

Why? If a simple distribution, then we can **easily sample** from it.

Then use decoder network to generate realistic samples from the distribution we care about (cat pictures!)



Want this to have a known (simple) distribution (e.g. Gaussian)

Image credit: Kevin Frans

Variational Autoencoders (VAE)

Recall: autoencoders

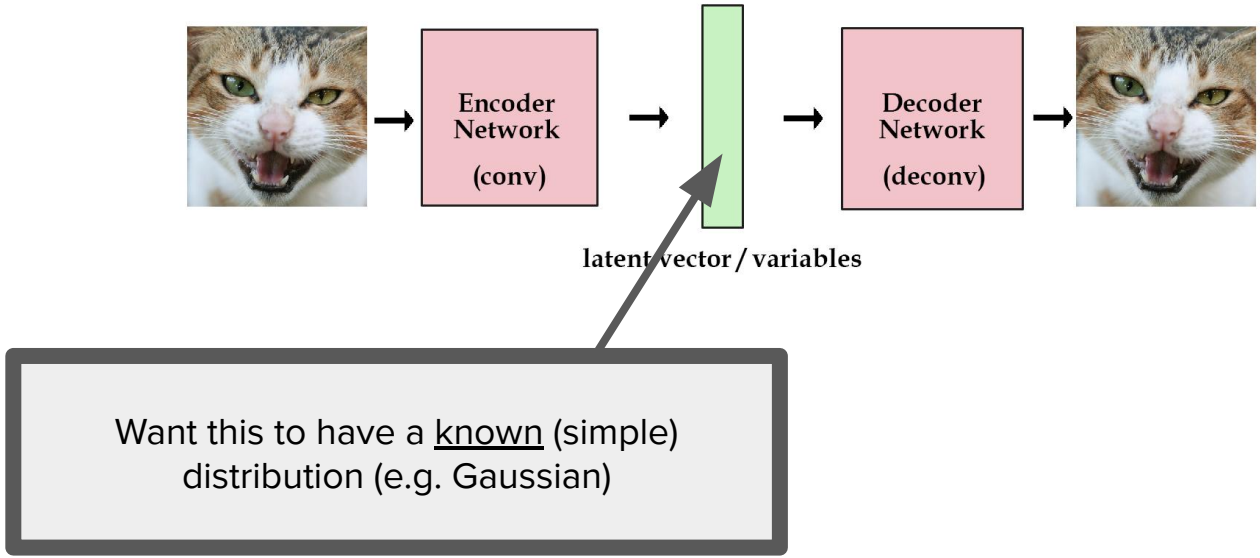


Image credit: Kevin Frans

Variational Autoencoders (VAE)

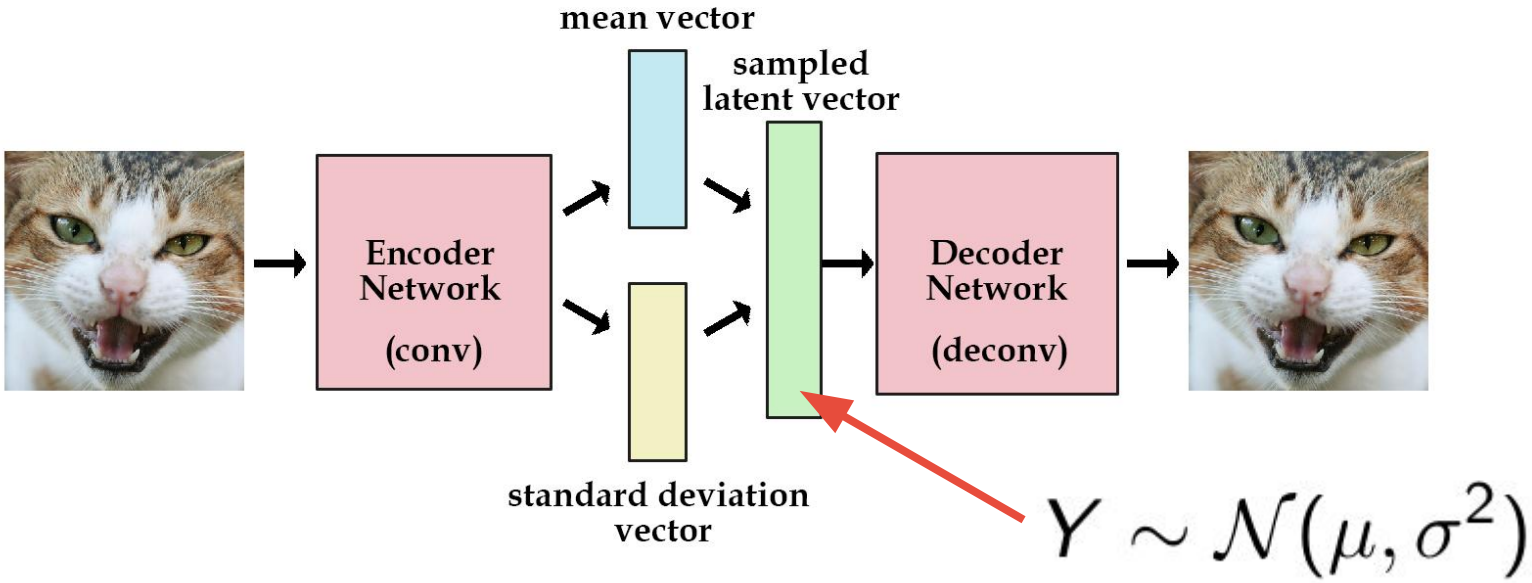


Image credit: Kevin Frans

Variational Autoencoders (VAE)

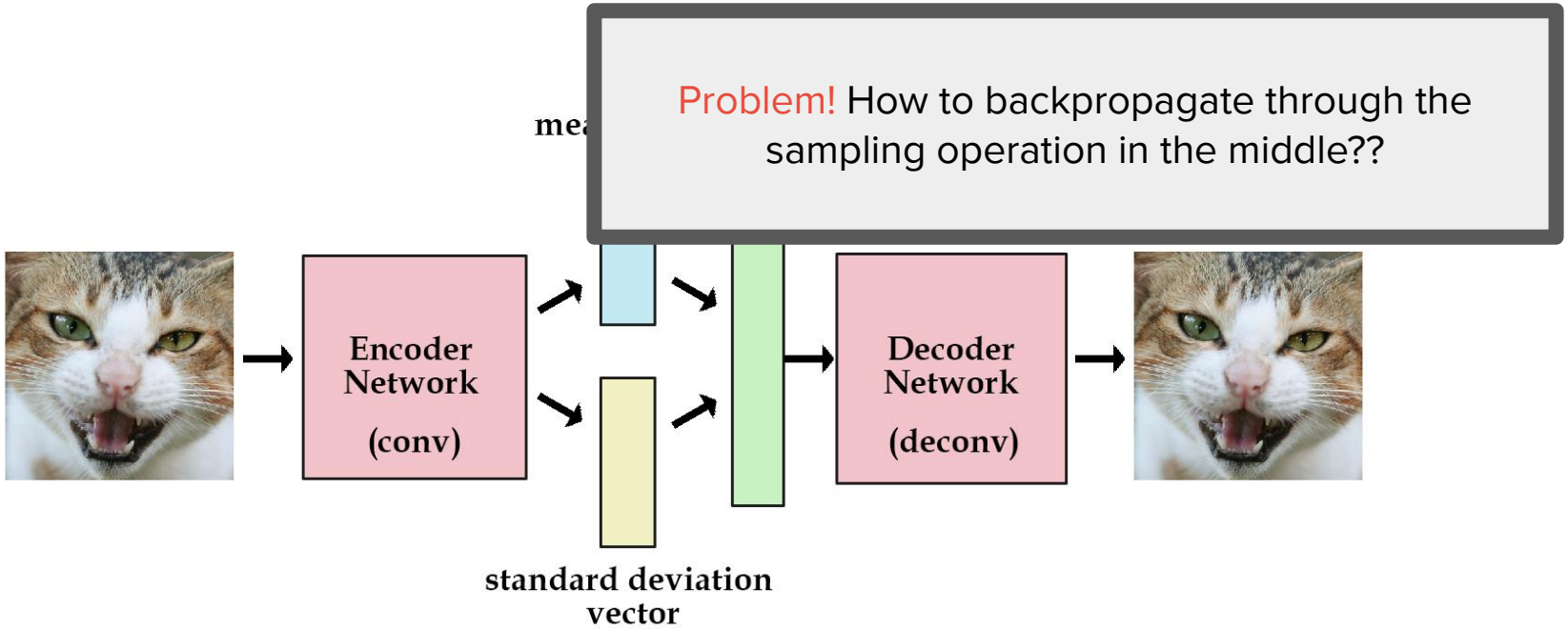


Image credit: Kevin Frans

VAE: reparameterization trick

Univariate case

$$X \sim \mathcal{N}(0, 1)$$

$$Y = \sigma X + \mu$$

$$Y \sim \mathcal{N}(\mu, \sigma^2)$$

Multivariate case

$$\mathbf{x} \sim \mathcal{N}_D(0, I)$$

$$\mathbf{y} = \Sigma^{\frac{1}{2}}\mathbf{x} + \mu$$

$$\mathbf{y} \sim \mathcal{N}_D(\mu, \Sigma)$$

VAE: reparameterization trick

Implication: to draw from a normal distribution

$$\mathcal{N}(\mu, \sigma^2)$$

We can just draw from a zero mean unit variance Gaussian, multiply by sigma and add mu

Gives us a way to backpropagate!

$$X \sim \mathcal{N}(0, 1)$$

$$Y = \sigma X + \mu$$

$$Y \sim \mathcal{N}(\mu, \sigma^2)$$

VAE: reparameterization trick

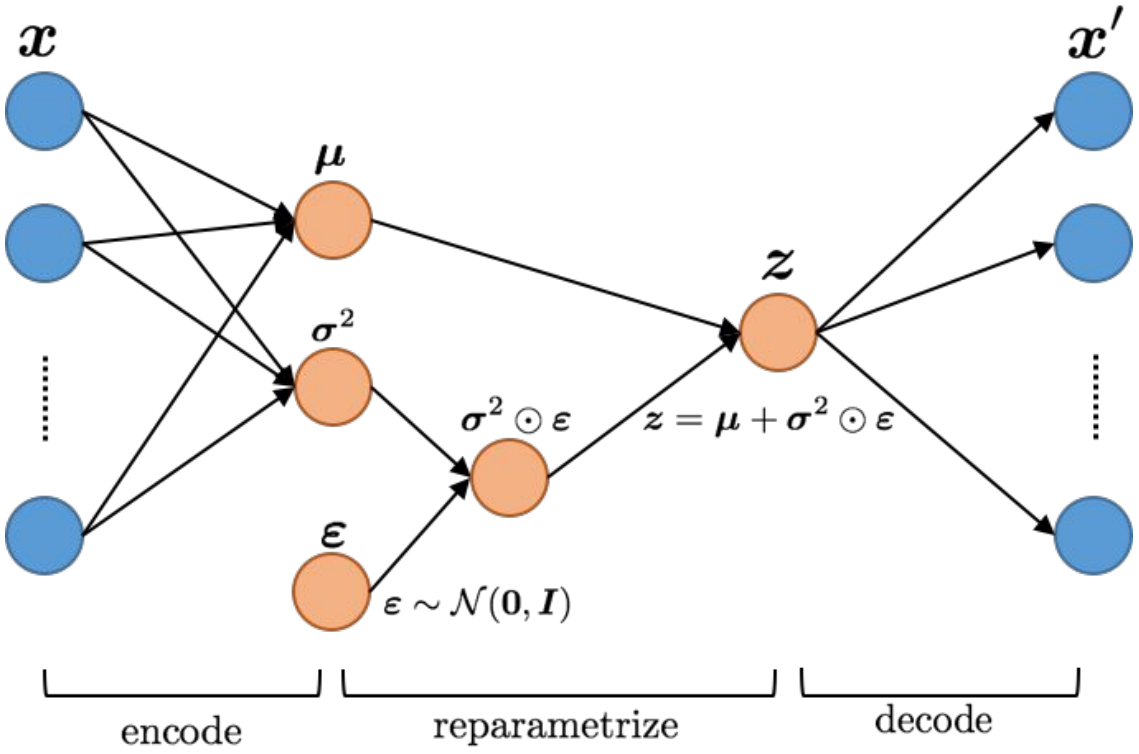


Image Source

Overview

Generative models

Generative adversarial networks

Colab: MNIST digit generation with a GAN

Issues and important variants

Conditional GANs

Variational autoencoders (VAE)

Colab: convolutional VAE on MNIST

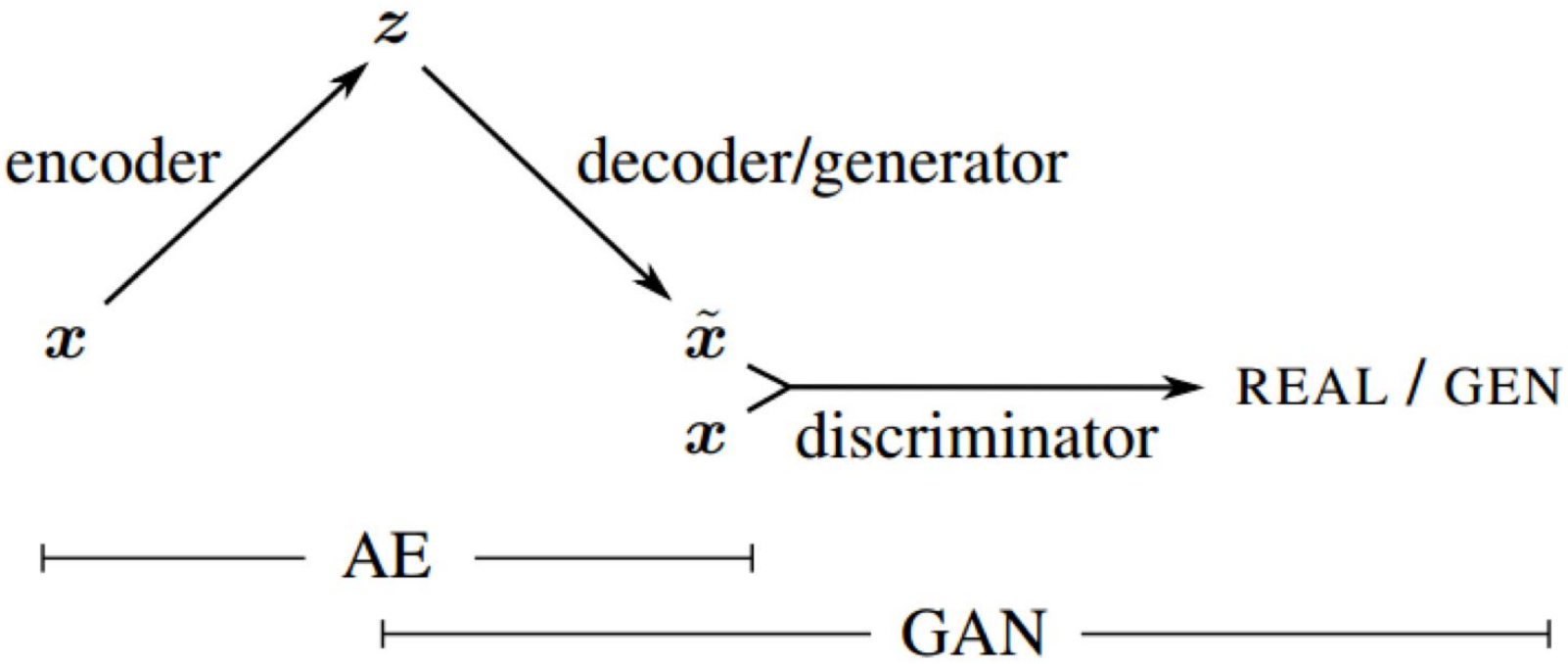
Convolutional VAE on MNIST

VAE in practice

- VAEs are **easier to train** than GAN
- VAEs have a tractable likelihood function
- GANs often suffer from mode collapse. But VAEs spread probability mass too widely
- GAN produces higher fidelity (sharper) samples (VAE samples are blurrier)

Image Credit: Alec Radford

VAE GAN

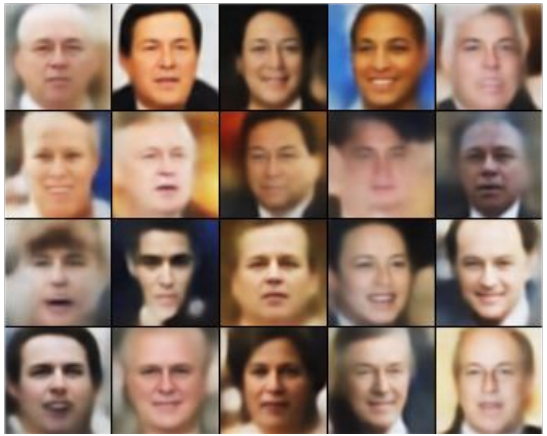


Larsen et al. Autoencoding beyond pixels using a learned similarity metric. ICML 2016

VAE GAN

Input

VAE reconstruction



VAE/GAN reconstruction

Larsen et al. Autoencoding beyond pixels using a learned similarity metric. ICML 2016

Summary

GAN: pit a **generator** (*G*) against a **discriminator** (*D*) and **alternately train** each to improve both.

Attempt to reach a **saddle point** in the loss surface (maximum likelihood for *G*, minimum error for *D*).

Can be **difficult to train** in practice! Generator and discriminator must stay in **equilibrium**. Lots of proposed **variants**.

Conditional GAN: generate conditioned on other content (e.g. captions)

VAE: output is **blurrier** but likelihood function is **tractable**. **Easier to train**.

Questions?