Generative Neural Networks

Taxonomy of generative models

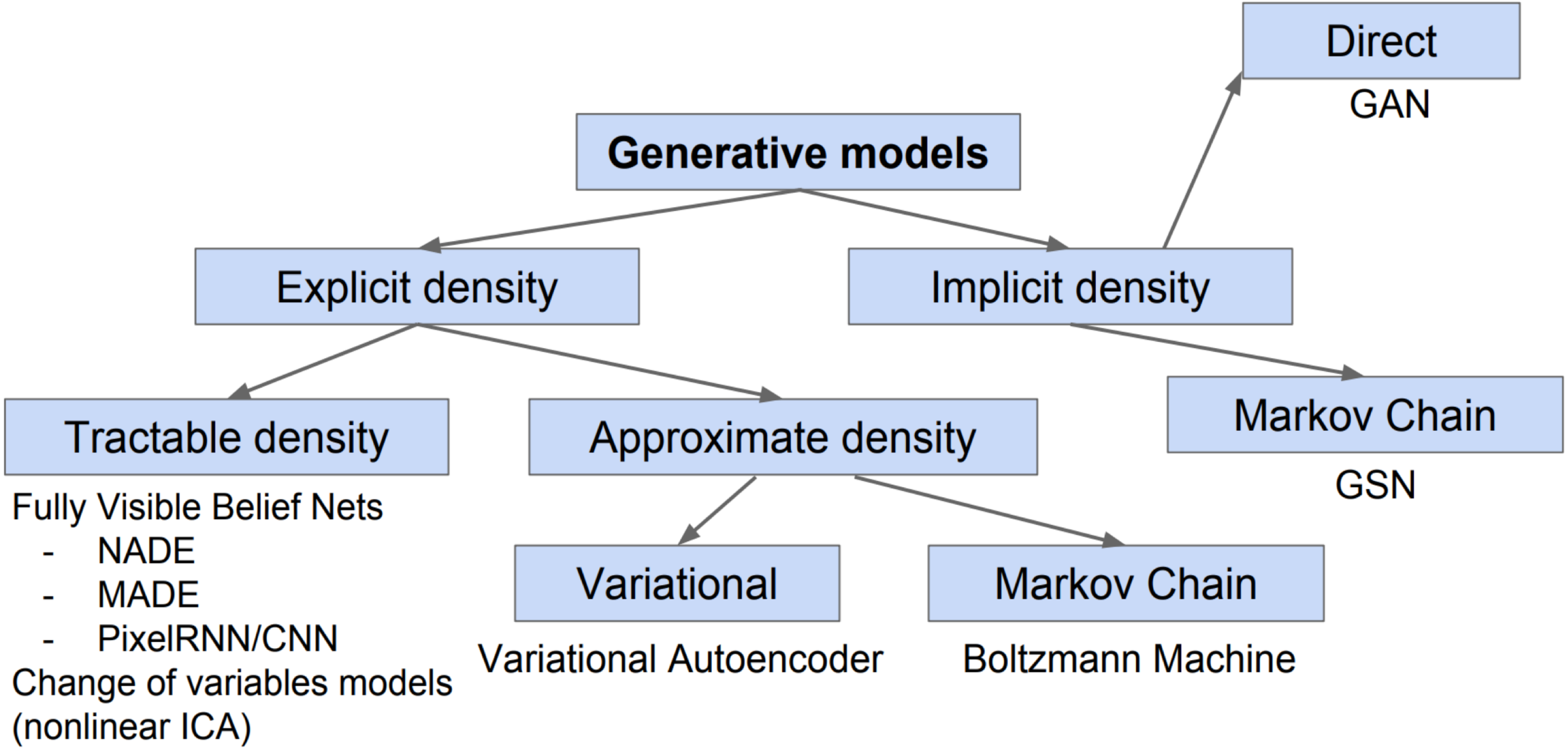


Figure from Ian Goodfellow, Tutorial on Generative Adversarial /networks, 2017

Generative Content Overview

- Generative Adversarial Networks (GANs)
  - Implicit densities

- Conditional GANs (cGANs)
  - Adding control

- Autoregressive Neural Networks
  - Explicit densities

- Neural Rendering: cutting edge-video generation / NVS

Taxonomy of generative models

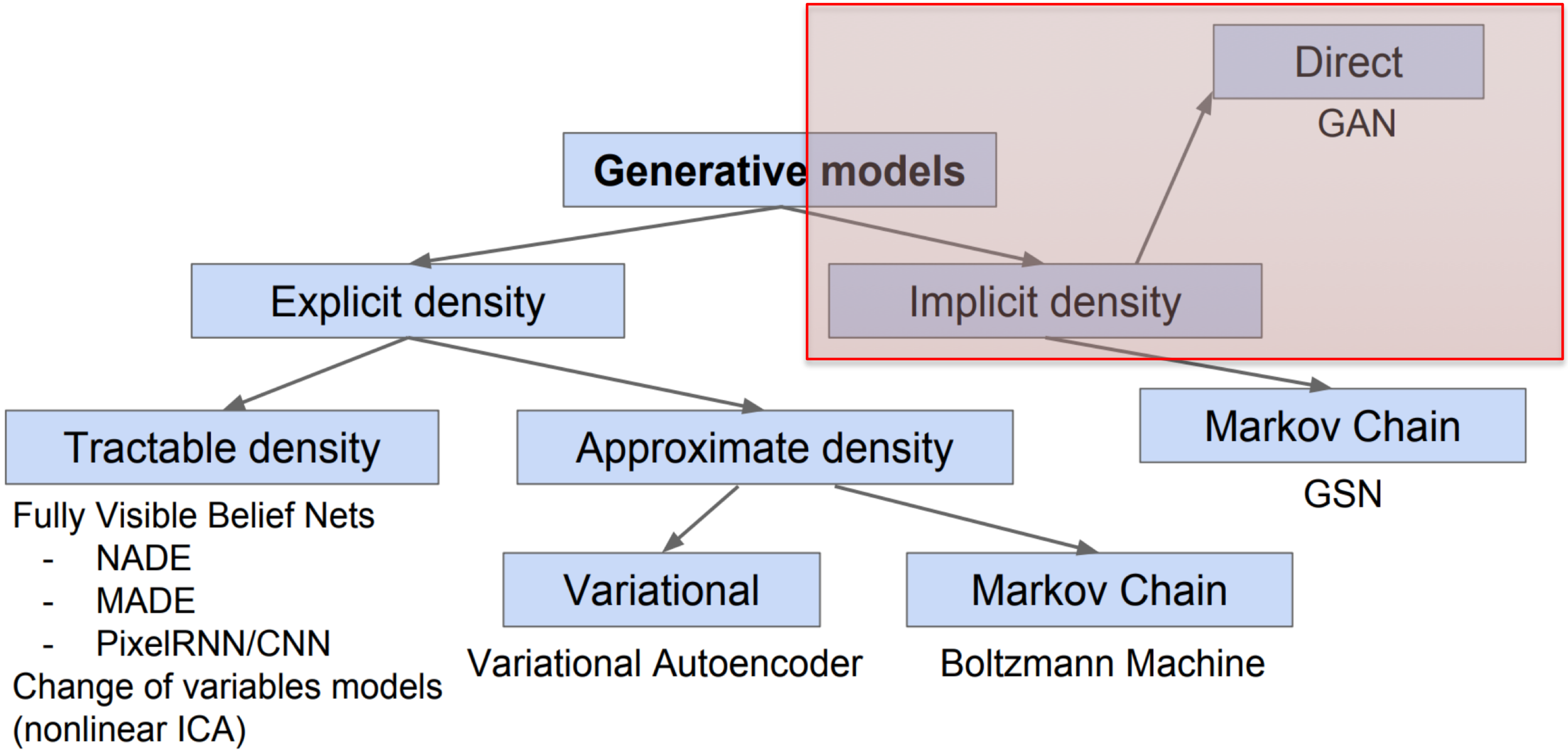
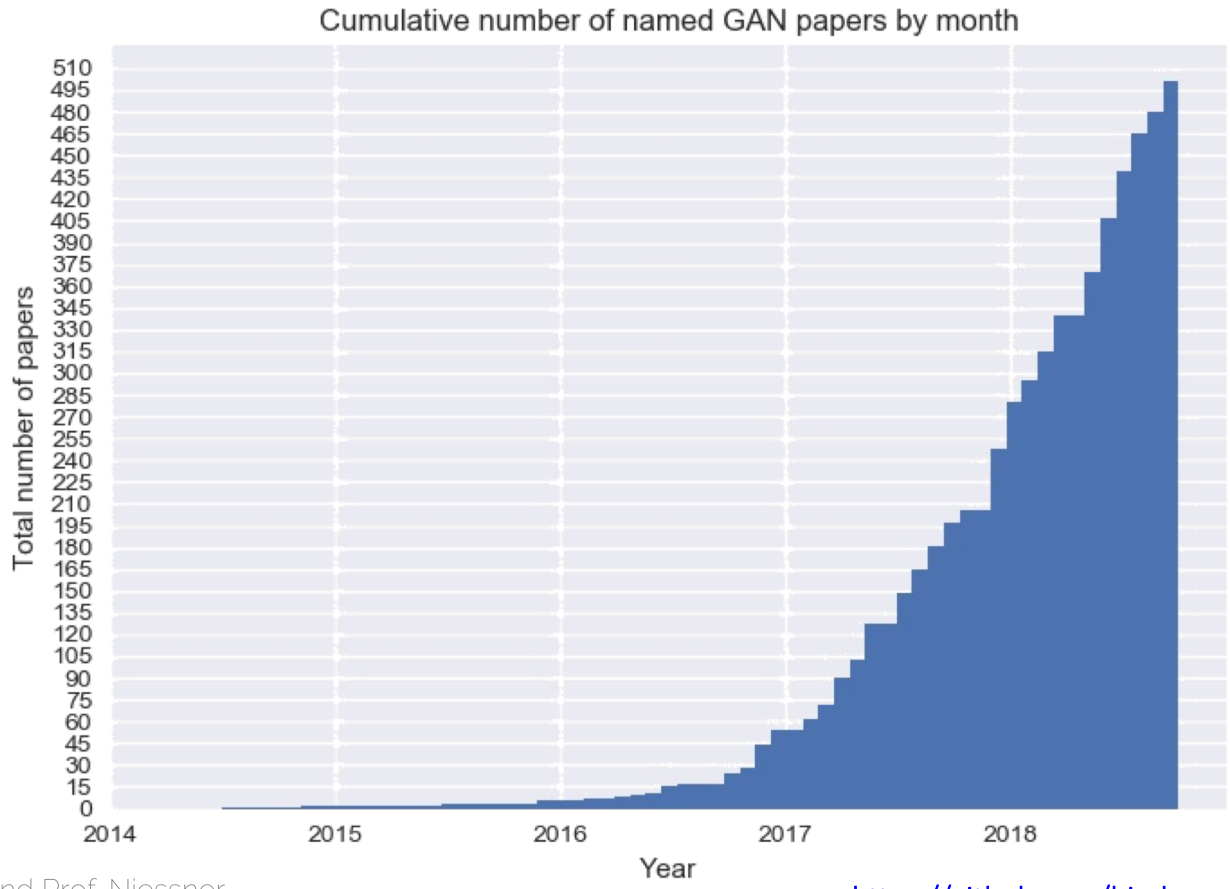


Figure from Ian Goodfellow, Tutorial on Generative Adversarial /networks, 2017

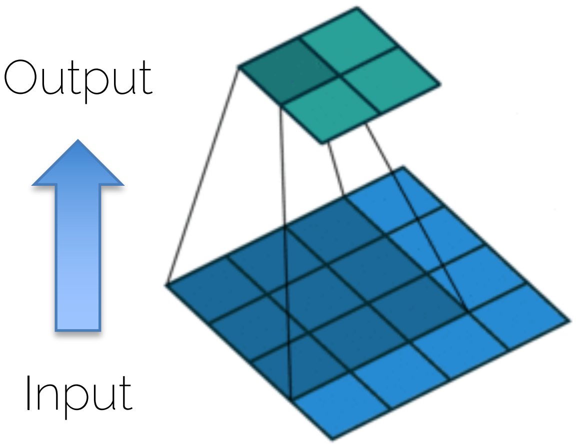
Generative Adversarial Networks (GANs)

Generative Adversarial Networks (GANs)

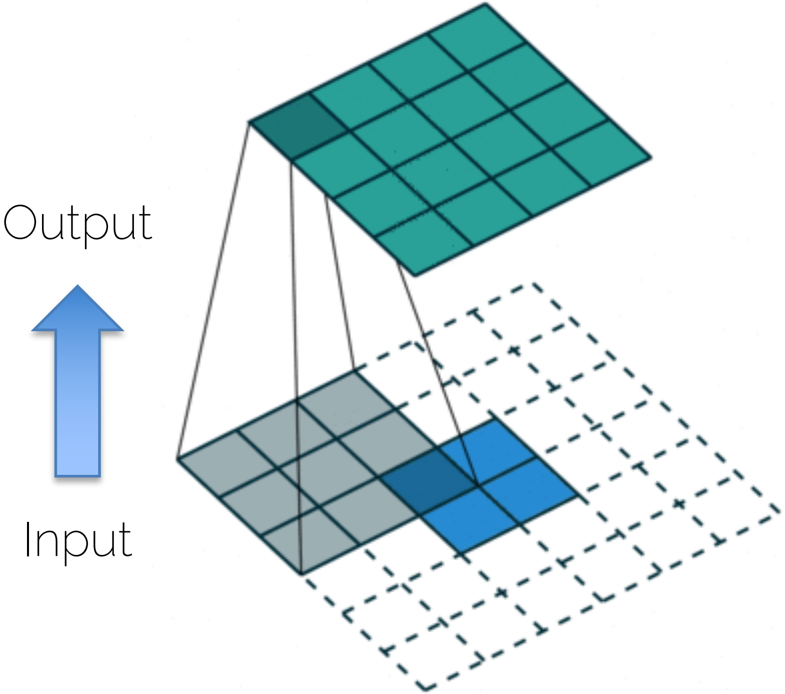


https://github.com/hindupuravinash/the-gan-zoo

Convolution and Deconvolution



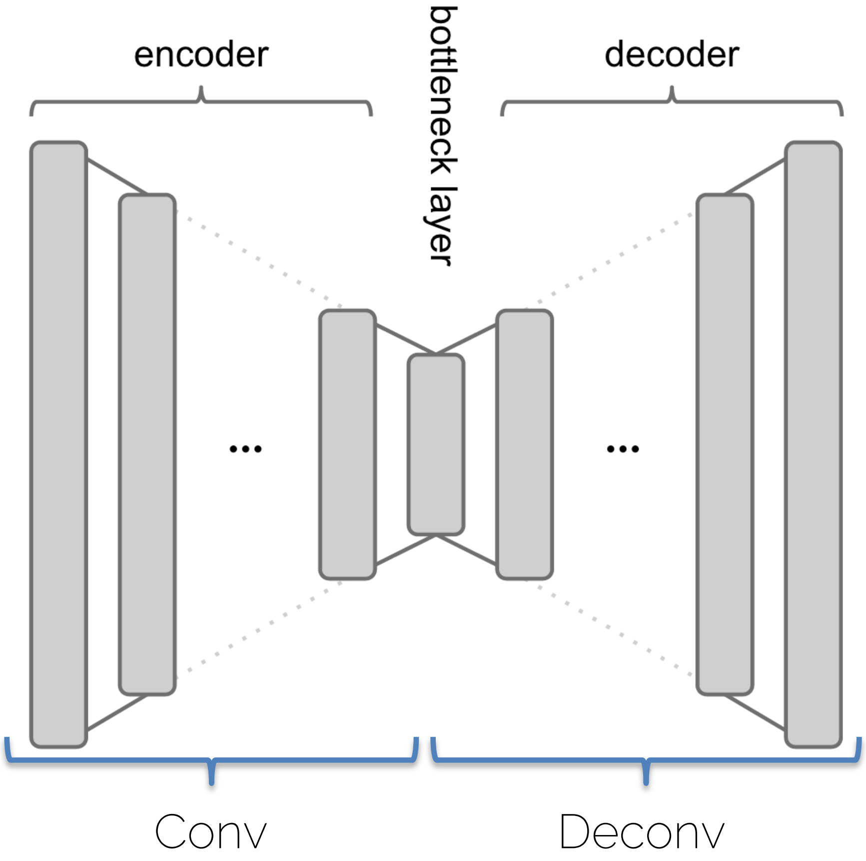
Convolution
no padding, no stride



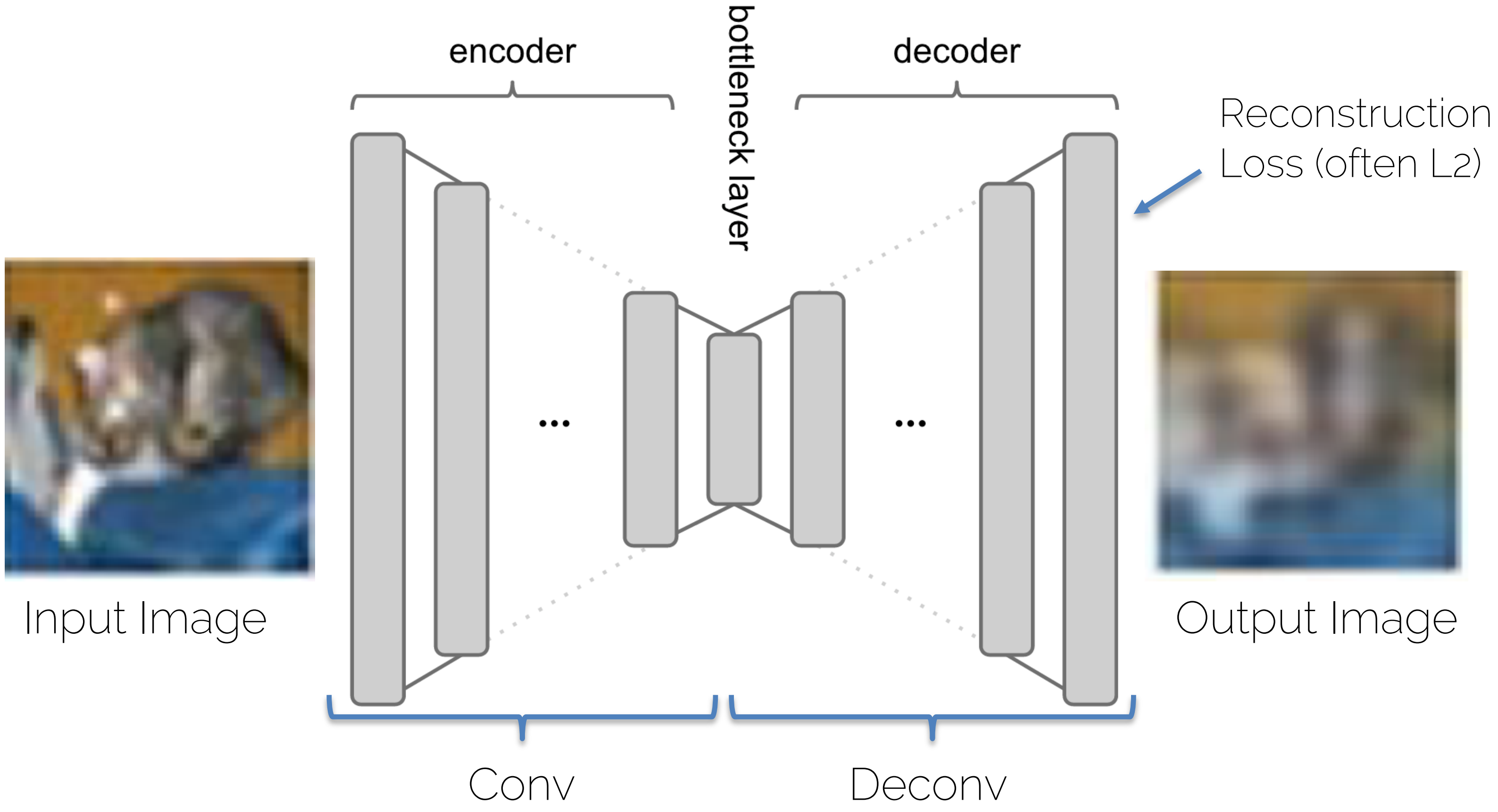
Transposed convolution
no padding, no stride

https://github.com/vdumoulin/conv_arithmetic

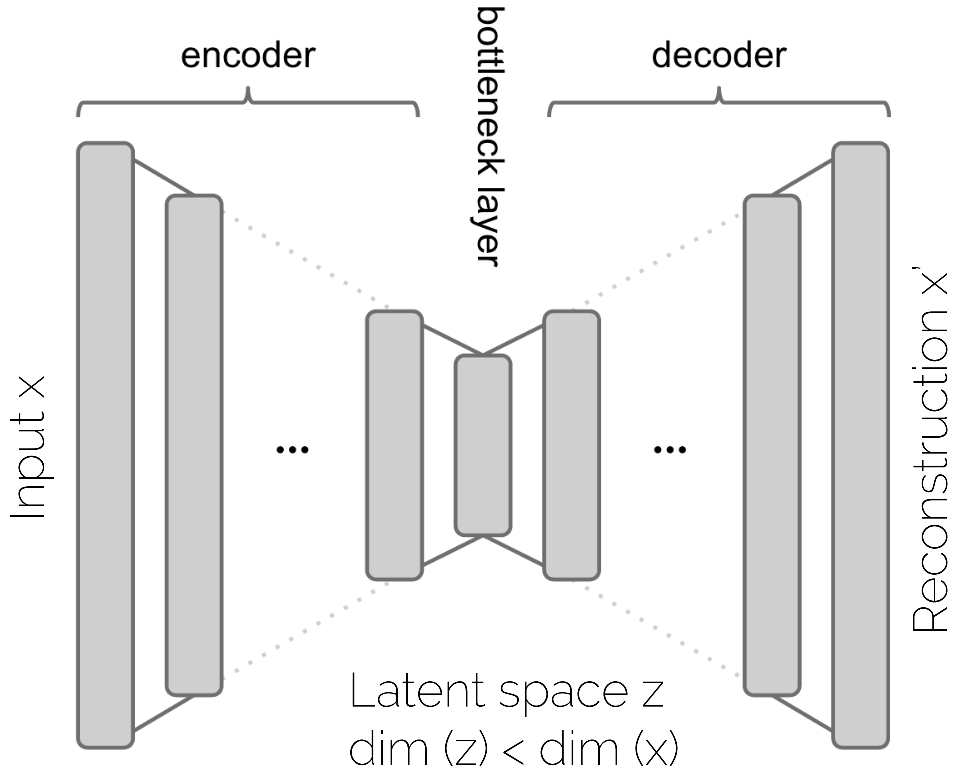
Autoencoder



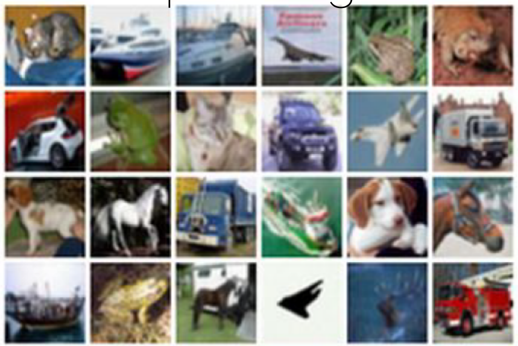
Reconstruction: Autoencoder



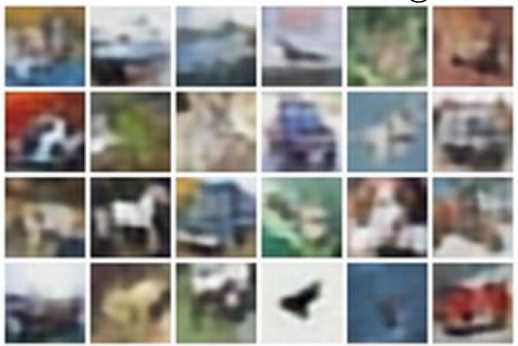
Training Autoencoders



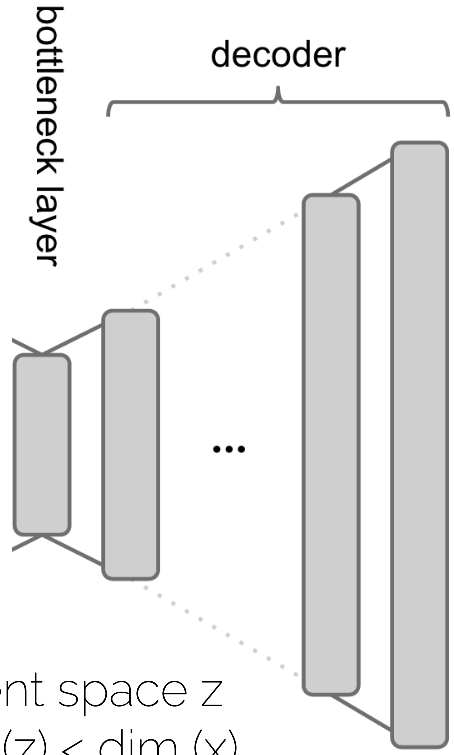
Input images



Reconstructed images



Decoder as Generative Model



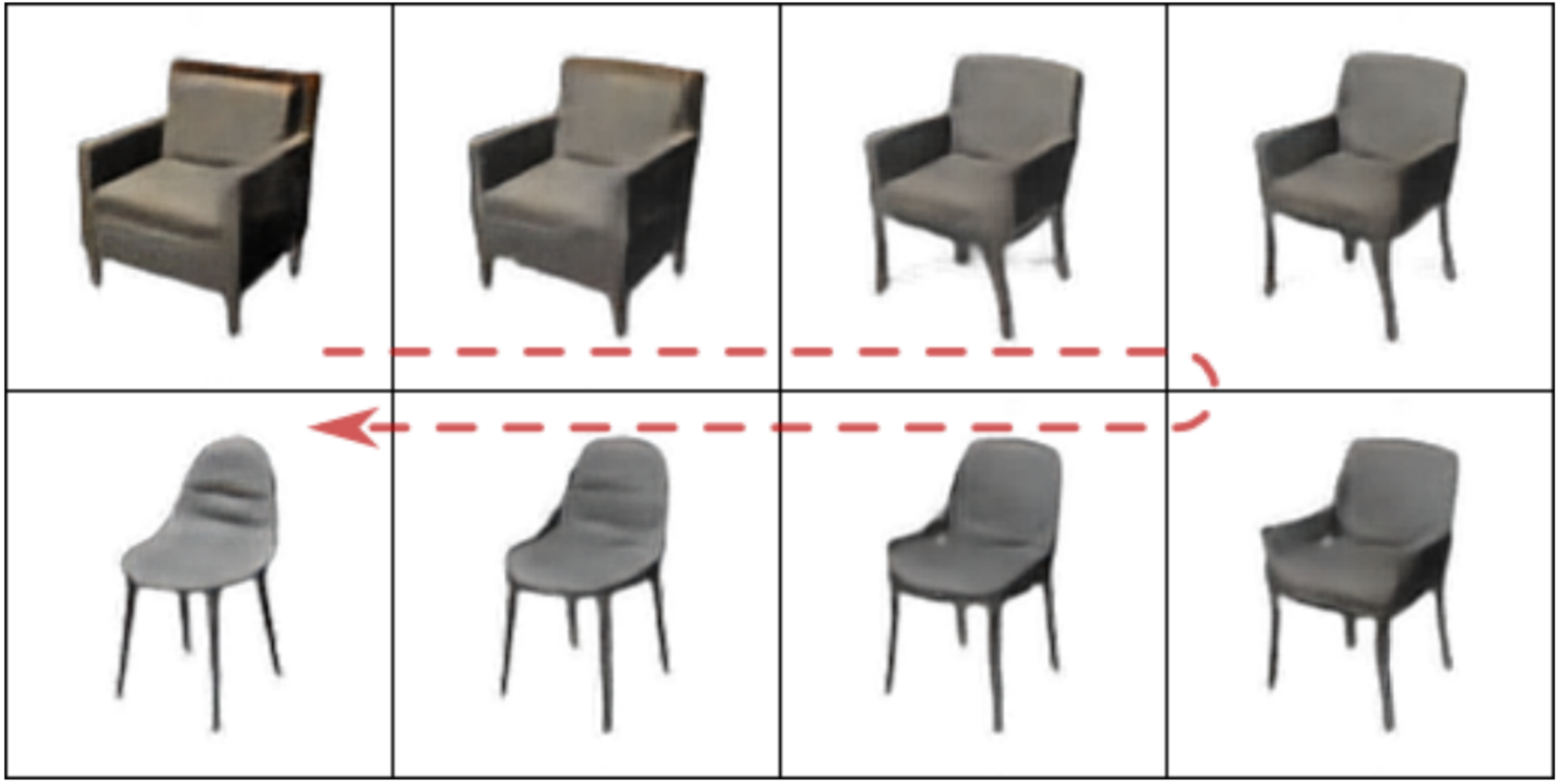
Reconstruction Loss (often L2)

Output Image

"Test time":
-> reconstruction from 'random' vector

Latent space z
dim (z) < dim (x)

Decoder as Generative Model



Interpolation between two chair models

[Dosovitsky et al. 14] Learning to Generate Chairs

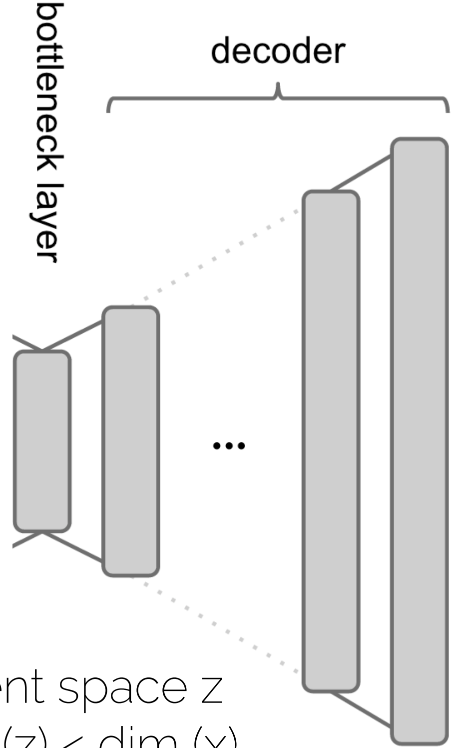
Decoder as Generative Model

Morphing between chair models

[Dosovitsky et al. 14] Learning to Generate Chairs

Decoder as Generative Model

"Test time":
-> reconstruction from
'random' vector

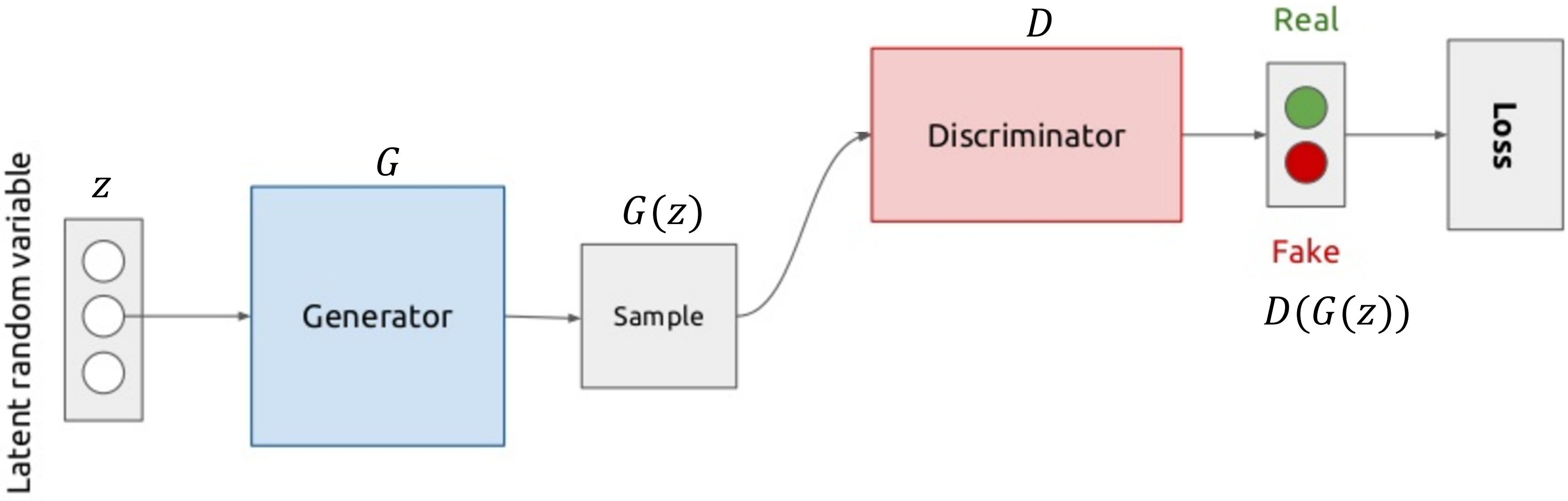


Latent space z
dim (z) < dim (x)

Reconstruction Loss
Often L2, i.e., sum of squared dist.
-> L2 distributes error equally
-> mean is opt.
-> res. Is blurry

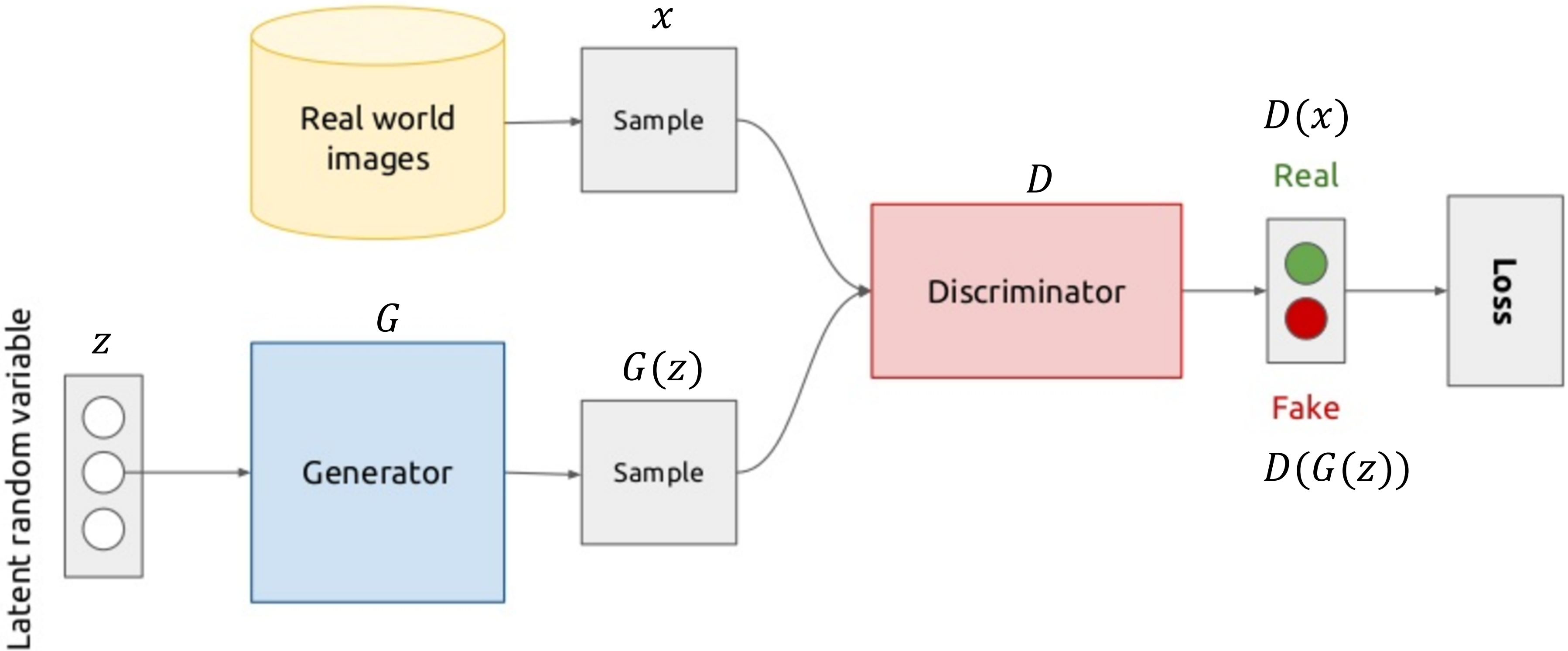
Instead of L2, can we "learn"a loss function?

Generative Adversarial Networks (GANs)



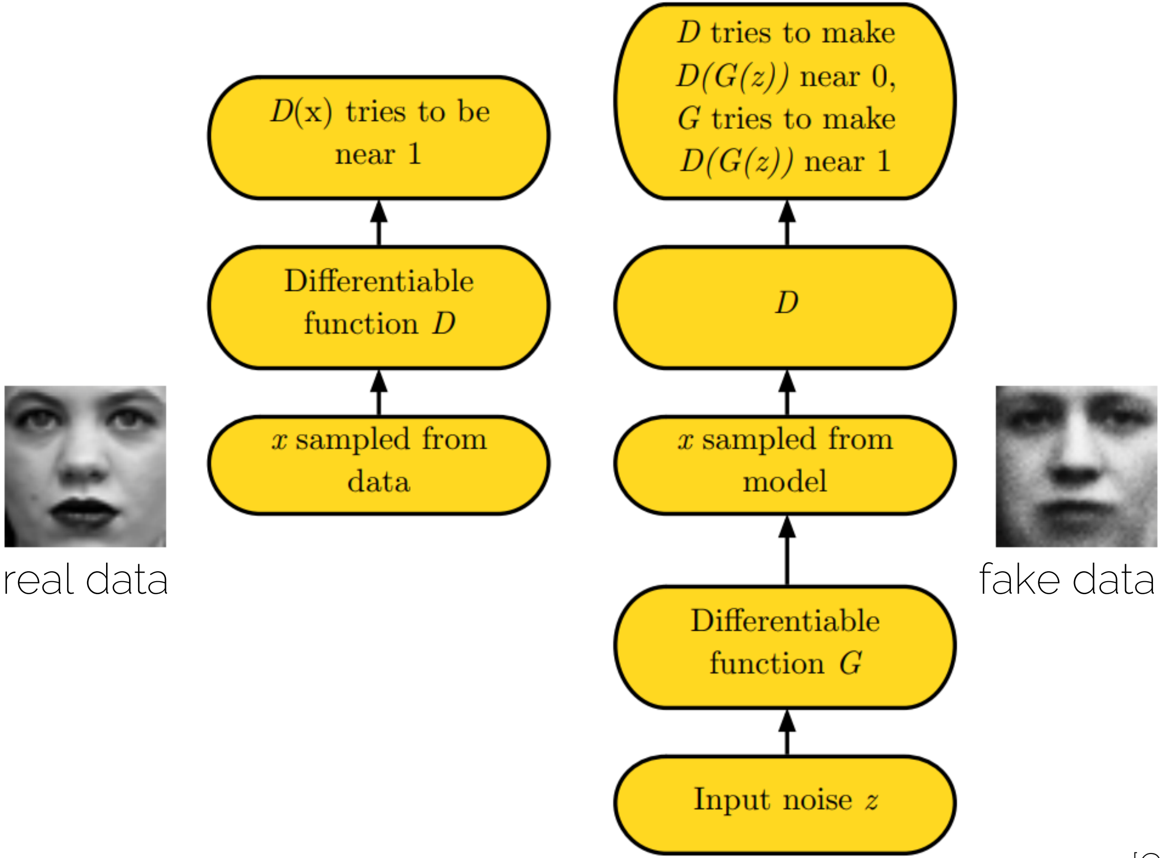
[Goodfellow et al. 14] GANs (slide McGuinness)

Generative Adversarial Networks (GANs)



[Goodfellow et al. 14] GANs (slide McGuinness)

Generative Adversarial Networks (GANs)



[Goodfellow et al. 14/16] GANs

GANs: Loss Functions

Discriminator loss

$$J^{(D)} = \underbrace{-\frac{1}{2}\mathbb{E}_{\boldsymbol{x}\sim p_{\text{data}}} \log D(\boldsymbol{x}) - \frac{1}{2}\mathbb{E}_{\boldsymbol{z}} \log\left(1 - D\left(G(\boldsymbol{z})\right)\right)}_{\text{binary cross entropy}}$$

Generator loss

$$J^{(G)} = -J^{(D)}$$

- Minimax Game:
  - G minimizes probability that D is correct
  - Equilibrium is saddle point of discriminator loss

-> D provides supervision (i.e., gradients) for G

[Goodfellow et al. 14/16] GANs

GANs: Loss Functions

Discriminator loss

$$J^{(D)} = -\frac{1}{2}\mathbb{E}_{\boldsymbol{x}\sim p_{\text{data}}} \log D(\boldsymbol{x}) - \frac{1}{2}\mathbb{E}_{\boldsymbol{z}} \log\left(1 - D\left(G(\boldsymbol{z})\right)\right)$$

Generator loss

$$J^{(G)} = -\frac{1}{2}\mathbb{E}_{\boldsymbol{z}} \log D\left(G(\boldsymbol{z})\right)$$

- Heuristic Method (often used in practice)
  - G maximizes the log-probability of D being mistaken
  - G can still learn even when D rejects all generator samples

Alternating Gradient Updates

- Step 1: Fix G, and perform gradient step to

$$J^{(D)} = -\frac{1}{2}\mathbb{E}_{\boldsymbol{x}\sim p_{\text{data}}} \log D(\boldsymbol{x}) - \frac{1}{2}\mathbb{E}_{\boldsymbol{z}} \log\left(1 - D\left(G(\boldsymbol{z})\right)\right)$$

- Step 2: Fix D, and perform gradient step to

$$J^{(G)} = -\frac{1}{2}\mathbb{E}_{\boldsymbol{z}} \log D\left(G(\boldsymbol{z})\right)$$

Vanilla GAN

for number of training iterations **do**

  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:

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

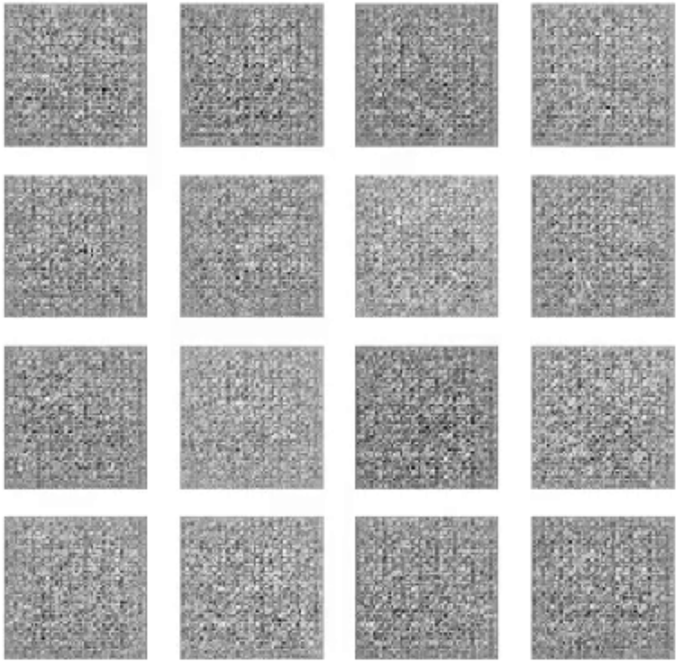
  end for

  - 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

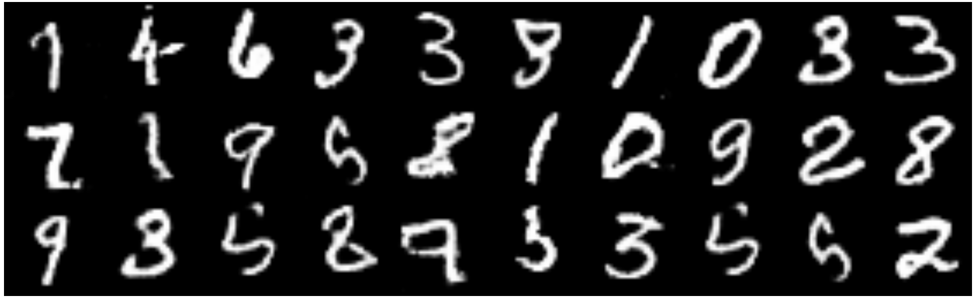
Training a GAN

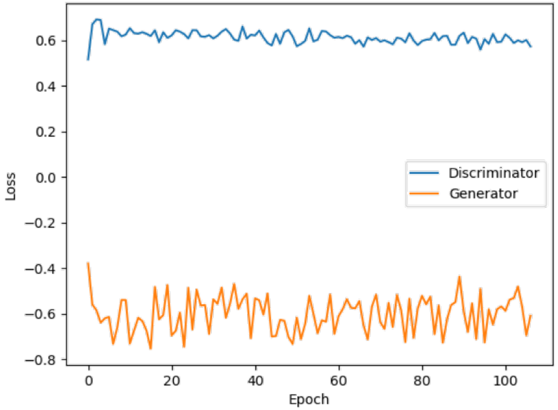


https://medium.com/ai-society/gans-from-scratch-1-a-deep-introduction-with-code-in-pytorch-and-tensorflow-cb03cdcdba0f

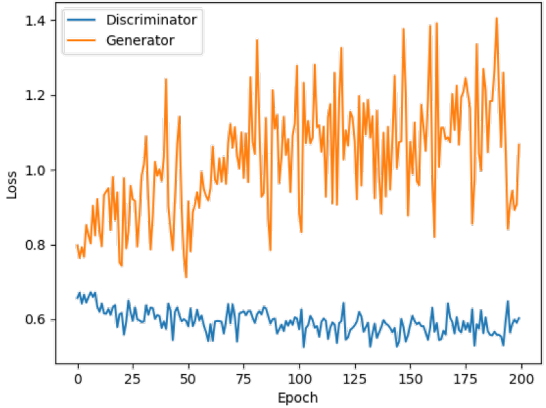
GANs: Loss Functions

Minimax



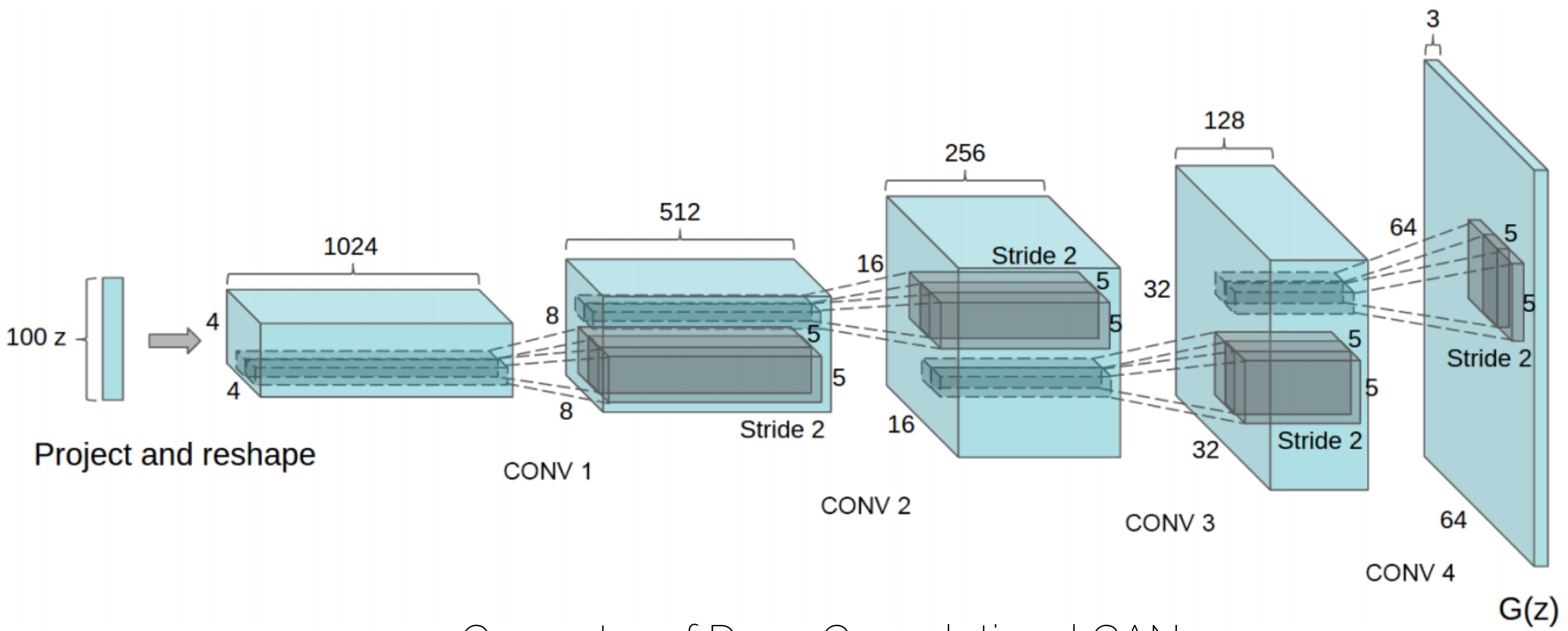


Heuristic



[Goodfellow et al. 14/16] GANs

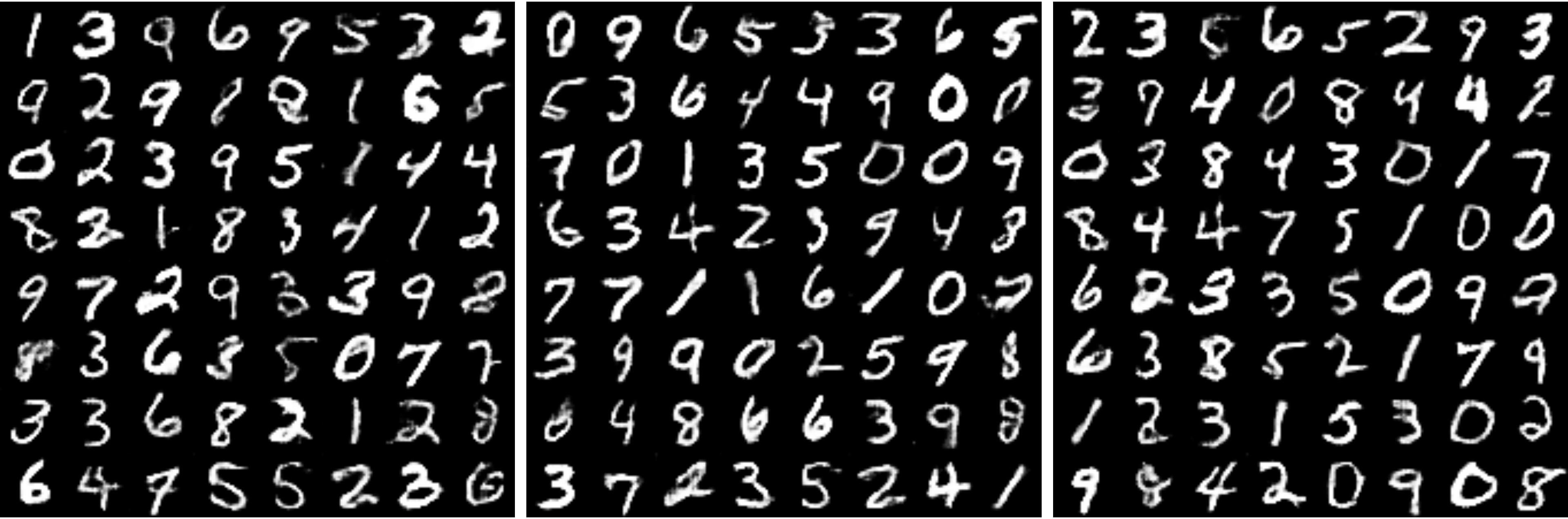
DCGAN: Generator



Generator of Deep Convolutional GANs

DCGAN: https://github.com/carpedm20/DCGAN-tensorflow

DCGAN: Results



Results on MNIST

DCGAN: https://github.com/carpedm20/DCGAN-tensorflow

DCGAN: Results

Results on CelebA (200k relatively well aligned portrait photos)

DCGAN: https://github.com/carpedm20/DCGAN-tensorflow

DCGAN: Results

Asian face dataset

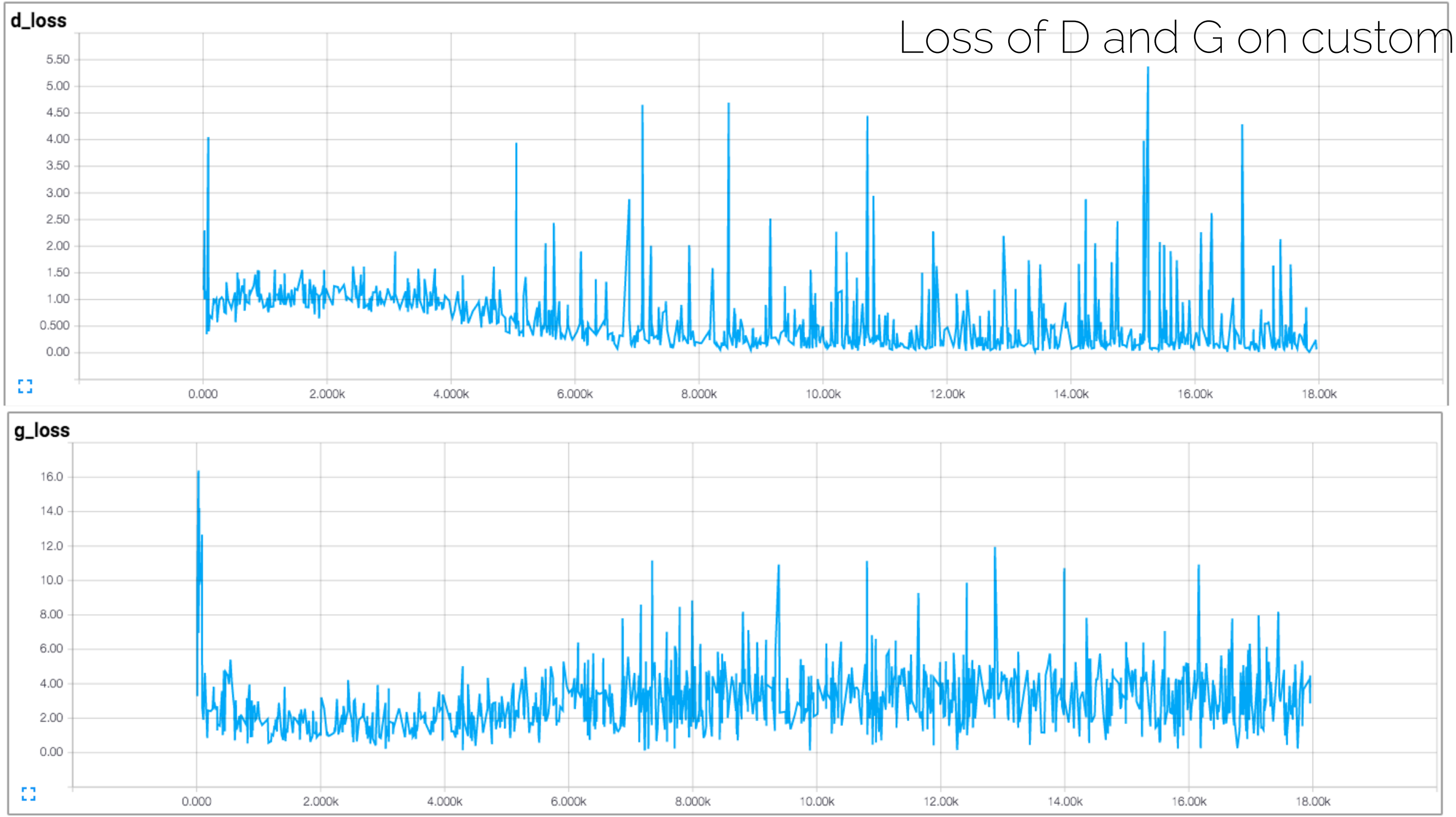
DCGAN: https://github.com/carpedm20/DCGAN-tensorflow

DCGAN: Results

DCGAN: https://github.com/carpedm20/DCGAN-tensorflow

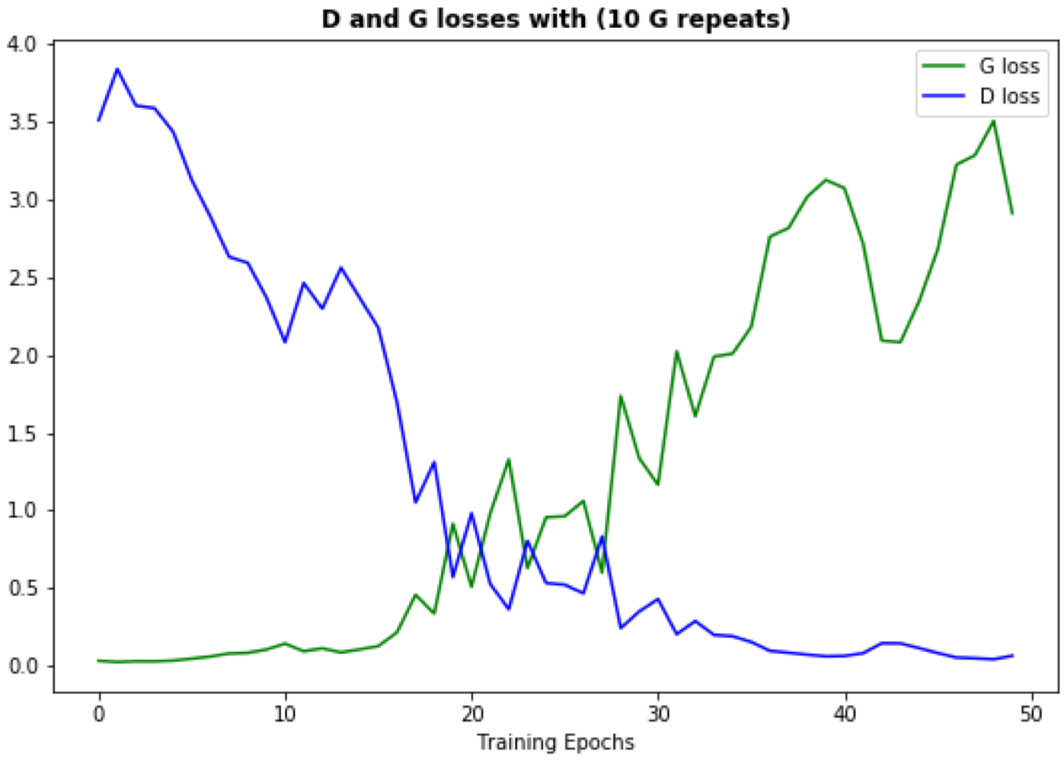
DCGAN: Results

Loss of D and G on custom dataset



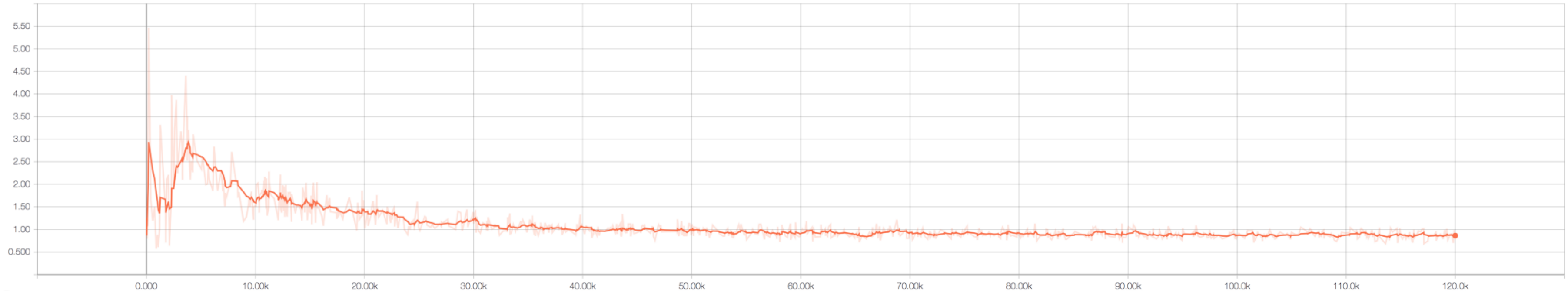
DCGAN: https://github.com/carpedm20/DCGAN-tensorflow

“Bad” Training Curves

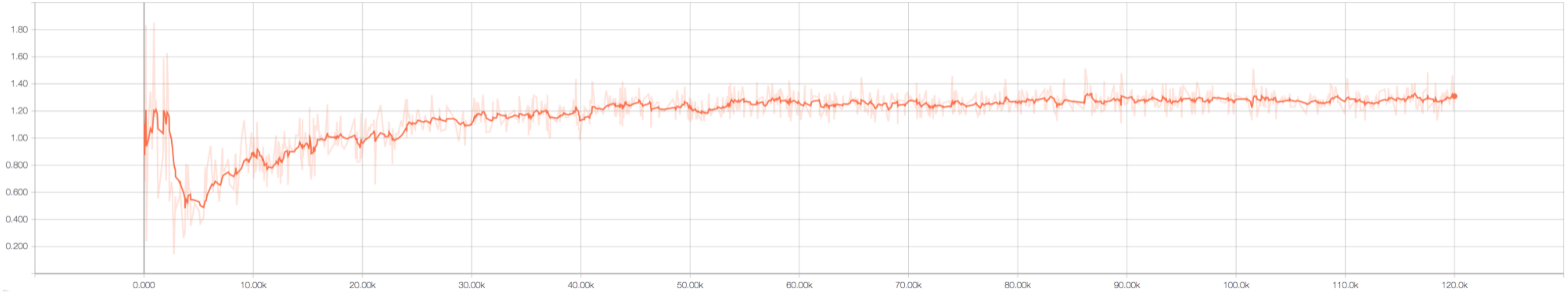


https://stackoverflow.com/questions/44313306/dcgans-discriminator-getting-too-strong-too-quickly-to-allow-generator-to-learn

“Good” Training Curves



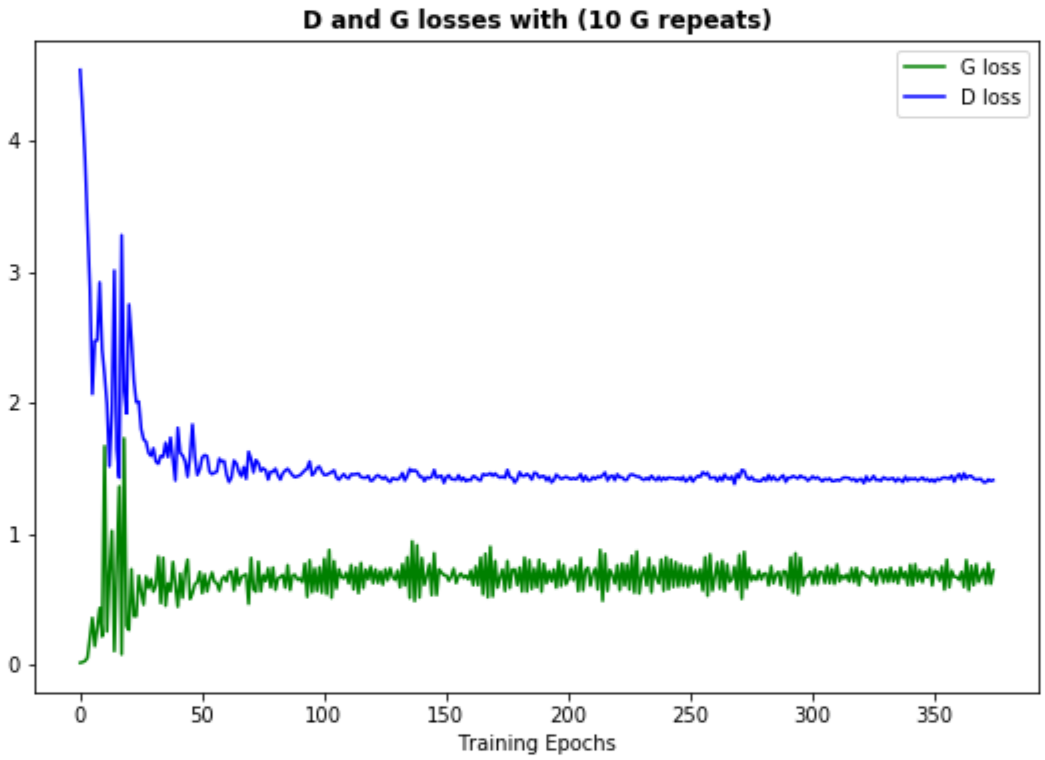
Generator’s Error through Time



Discriminator’s Error through Time

https://medium.com/ai-society/gans-from-scratch-1-a-deep-introduction-with-code-in-pytorch-and-tensorflow-cb03cdcdba0f

"Good" Training Curves



https://stackoverflow.com/questions/42690721/how-to-interpret-the-discriminators-loss-and-the-generators-loss-in-generative

Training Schedules

- Adaptive schedules

- For instance:

```
while loss_discriminator > t_d:
    train discriminator
while loss_generator > t_g:
    train generator
```

Weak vs Strong Discriminator

Need balance ☺

- Discriminator too weak?
  - No good gradients (cannot get better than teacher…)
- Generator too weak?
  - Discriminator will always be right

Mode Collapse

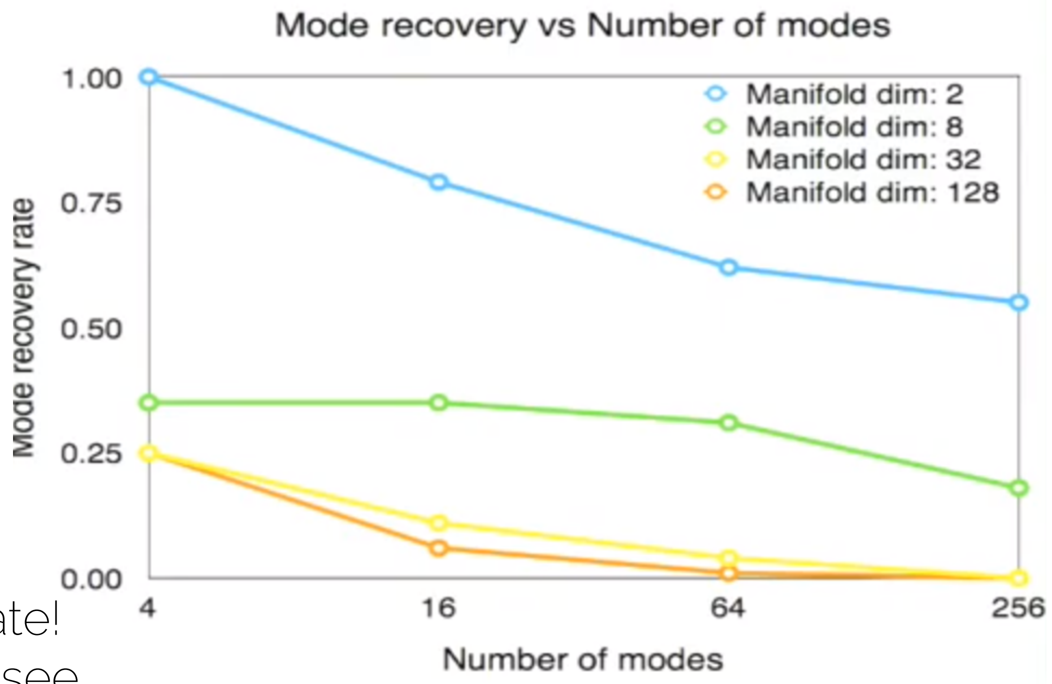
- $\min_G \max_D V(G,D) \neq \max_D \min_G V(G,D)$

- D in inner loop -> convergence to correct dist.
- G in inner loop -> easy to convergence to one sample

[Metz et al. 16]

Mode Collapse

- Data dim. Fixed (512)
- Performance correlates with # of modes



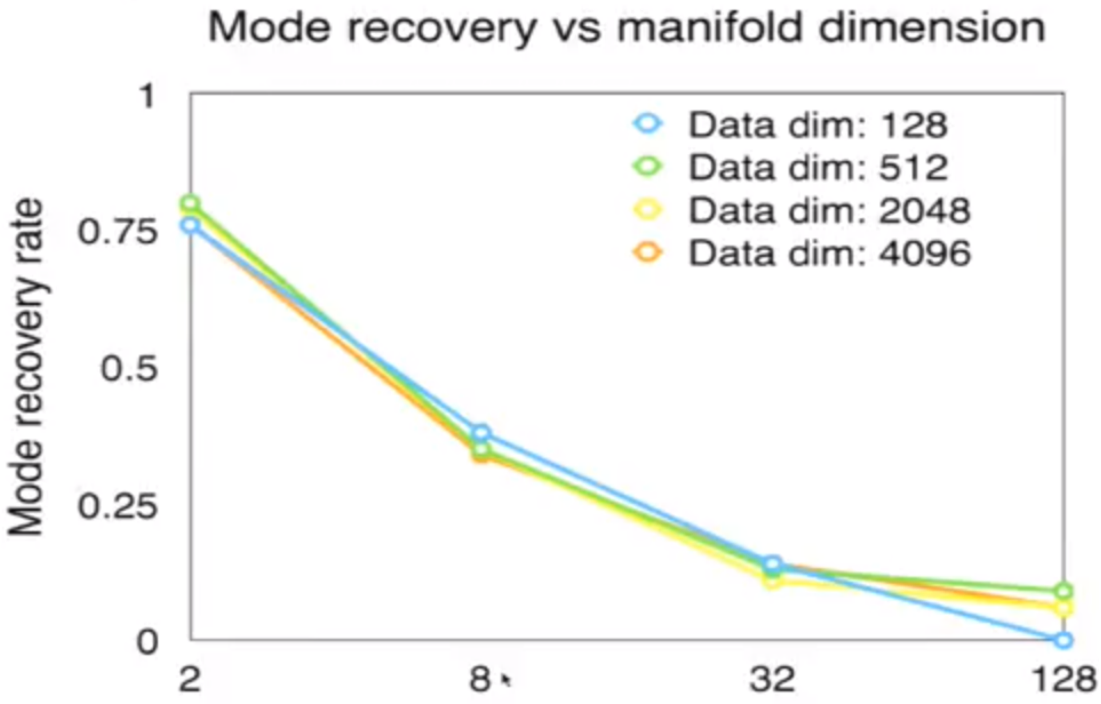
-> More modes, smaller recovery rate!
-> part of the reason, why we often see GAN-results on specific domains (e.g., faces)

Slide credit Ming-Yu Liu

Mode Collapse

- Performance correlates with dim of manifold

-> Larger latent space, more mode collapse

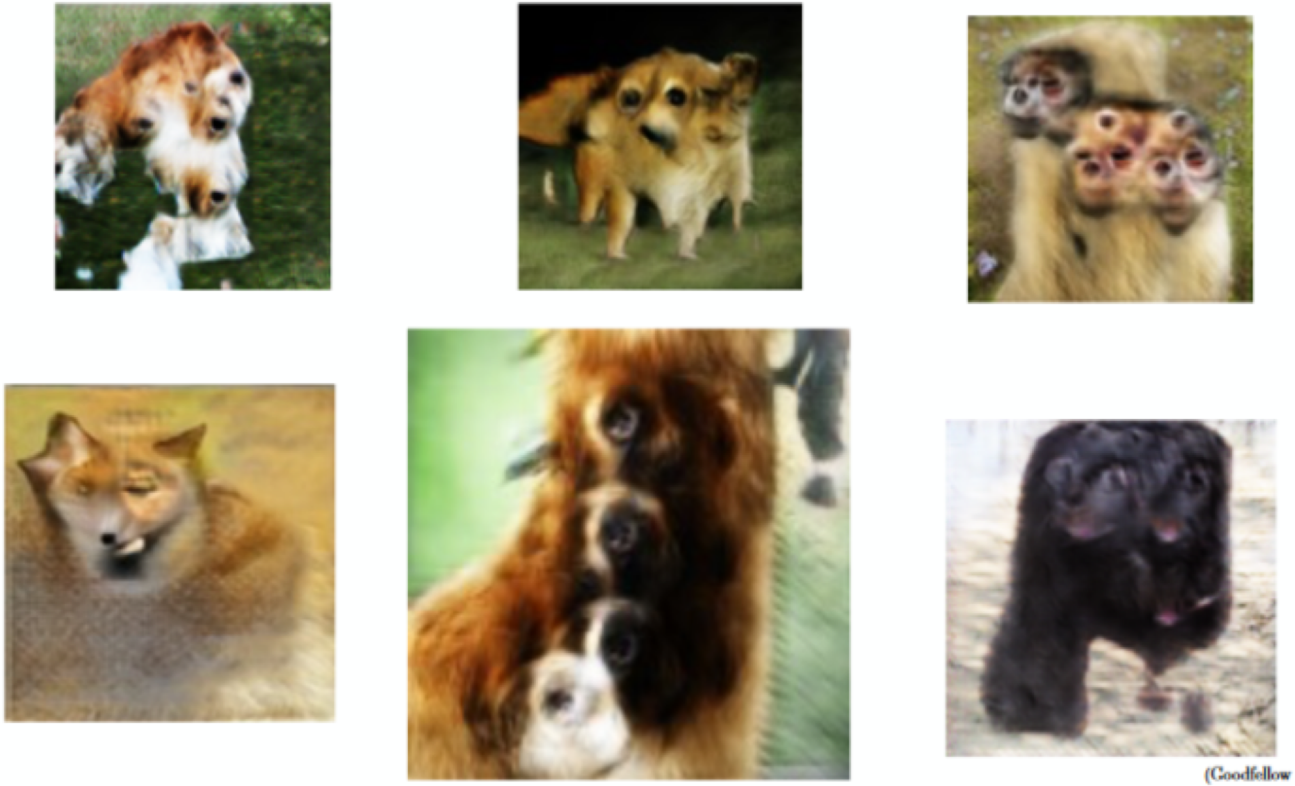


Slide credit Ming-Yu Liu

Problems with Global Structure

(Goodfellow 2016)

Problems with Counting



(Goodfellow 2016)

Evaluation of GAN Performance

Evaluation of GAN Performance

- Main difficulty of GANs: we don't know how good they are
- People cherry pick results in papers -> some of them will always look good, but how to quantify?
- Do we only memorize or do we generalize?
- GANs are difficult to evaluate! [This et al., ICLR 2016]

Evaluation of GAN Performance

Human evaluation:

- Every n updates, show a series of predictions
- Check train curves
- What does 'look good' mean at the beginning?
  - Need variety!
  - But don't have 'realistic' predictions yet…
- If it doesn't look good? Go back, try different hyperparameters…

Evaluation of GAN Performance

Inception Score (IS)

- Measures saliency and diversity

- Train an accurate classifier
- Train a image generation model (conditional)
- Check how accurate the classifier can recognize the generated images
- Makes some assumptions about data distributions…

Evaluation of GAN Performance

Inception Score (IS)

- Saliency: check whether the generated images can be classified with high confidence (i.e., high scores only on a single class)

- Diversity: check whether we obtain samples from all classes

<u>What if we only have one good image per class?</u>

Evaluation of GAN Performance

- Could also look at discriminator
  - If we end up with a strong discriminator, then generator must also be good
  - Use D features, for classification network
  - Only fine-tune last layer
  - If high class accuracy -> we have a good D and G

Caveat: not sure if people do this... Couldn't find paper

Next: Making GANs Work in Practice

- Training / Hyperparameters (most important)
- Choice of loss function
- Choice of architecture

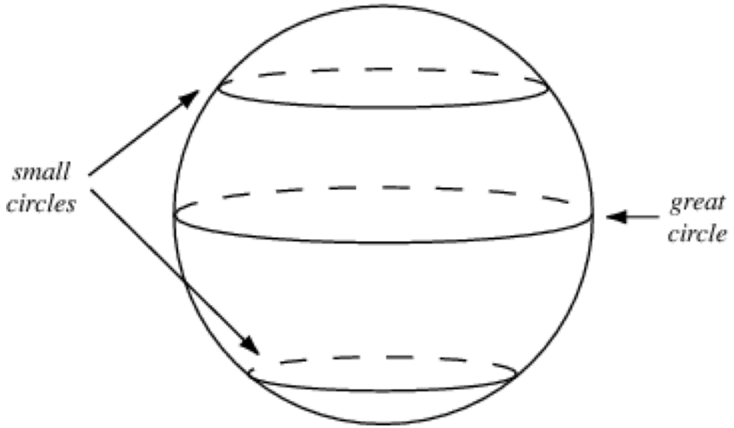
GAN Hacks: Normalize Inputs

- Normalize the inputs between -1 and 1
- Tanh as the last layer of the generator output
- No-brainer ☺

https://github.com/soumith/ganhacks

GAN Hacks: Sampling

- Use a spherical z
- Don't sample from a uniform distribution
- Sample from a Gaussian Distribution

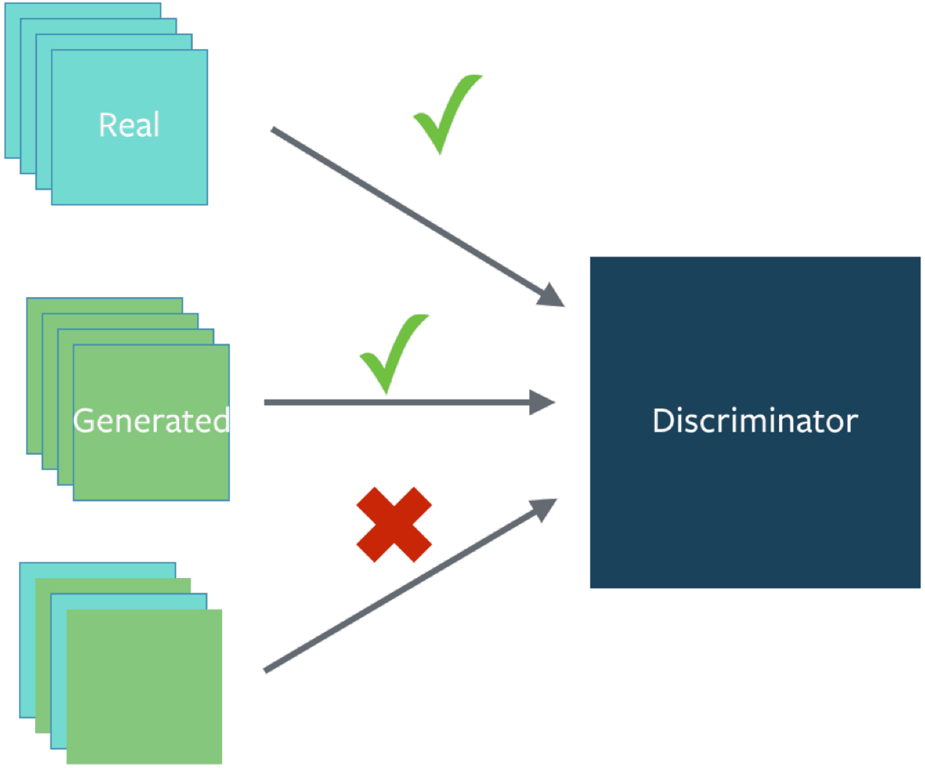


- When doing interpolations, do the interpolation via a great circle, rather than a straight line from point A to point B

- Tom White's Sampling Generative Networks ref code https://github.com/dribnet/plat has more details

GAN Hacks: BatchNorm

- Use Batch Norm
- Construct different mini-batches for real and fake, i.e. each mini-batch needs to contain only all real images or all generated images.



GAN Hacks: Use ADAM

- See Adam usage [Radford et al. 15]
- SGD for discriminator
- ADAM for generator

GAN Hacks: One-sided Label Smoothing

- Prevent discriminator from giving too large gradient signal to generator:

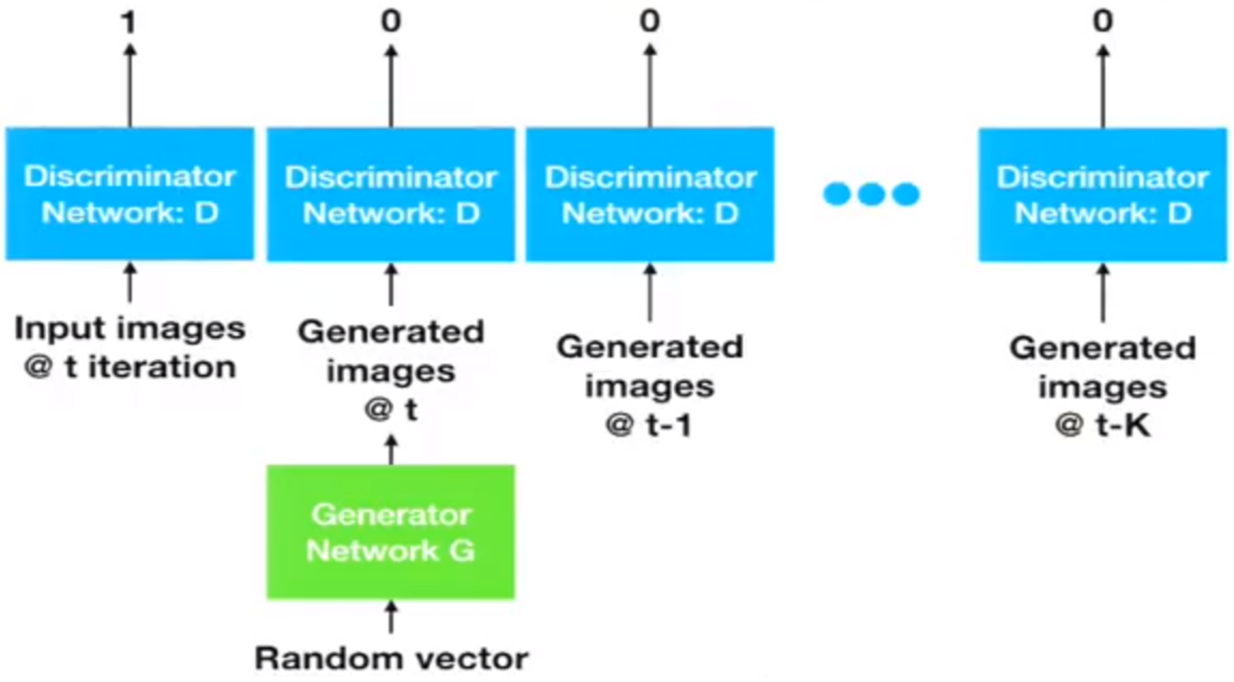
$$J^{(D)} = -\frac{1}{2}\mathbb{E}_{\boldsymbol{x}\sim p_{\text{data}}}\lambda \log D(\boldsymbol{x}) - \frac{1}{2}\mathbb{E}_{\boldsymbol{z}} \log\left(1 - D\left(G(\boldsymbol{z})\right)\right)$$

Some value smaller than 1; e.g.,0.9

-> reduces confidence; i.e., makes disc. 'weaker'
-> encourages 'extreme samples' (prevents extrapolating)

Salimans et al. 17 "Improved Techniques for Training GANs"

GAN Hacks: Historical Generator Batches

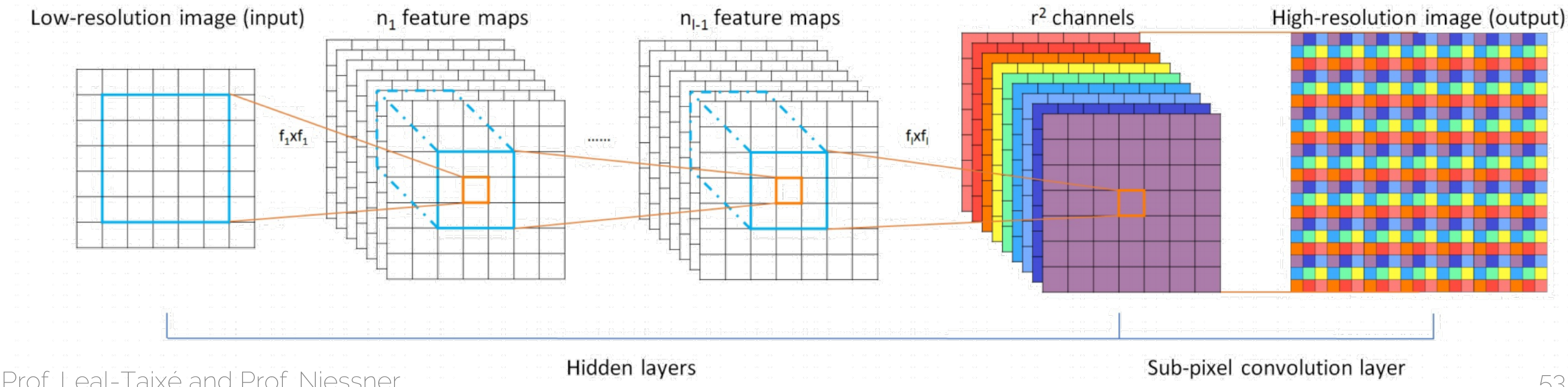


Help stabilize discriminator training in early stage

Srivastava et al. 17 “Learning from Simulated and Unsupervised Images through Adversarial Training”

GAN Hacks: Avoid Sparse Gradients

- Stability of GAN game suffers if gradients are sparse
- LeakyReLU -> good in both G and D
- Downsample -> use average pool, conv+stride
- Upsample -> deconv+stride, PixelShuffle



[Shi et al. 16] https://arxiv.org/pdf/1609.05158.pdf

Exponential Averaging of Weights

- Problem: discriminator is noisy due to SGD

- Rather than taking final result of a GAN, would be biased on last latest iterations (i.e., latest training samples),

  -> exponential average of weights

  -> keep second 'vector' of weights that are averaged
  -> almost no cost, average of weights from last n iters

New Objective Functions

New Objective Functions

"heuristic is standard…"

EBGAN: "Energy-based Generative Adversarial Networks"

BEGAN: "Boundary Equilibrium GAN"

WGAN: "Wasserstein Generative Adversarial Networks"

LSGAN: "Least Squares Generative Adversarial Networks"

….

The loss function alone will not make it suddenly work!

GAN Losses: EBGAN

- Discriminator is AE (Energy-based GAN)
- a good autoencoder: we want the reconstruction cost D(x) for real images to be low.
- a good critic: we want to penalize the discriminator if the reconstruction error for generated images drops below a value m.

$$\mathcal{L}_D(x, z) = D(x) + [m - D(G(z))]^+$$

$$\mathcal{L}_G(z) = D(G(z))$$

$$D(x) = ||Dec(Enc(x)) - x||$$

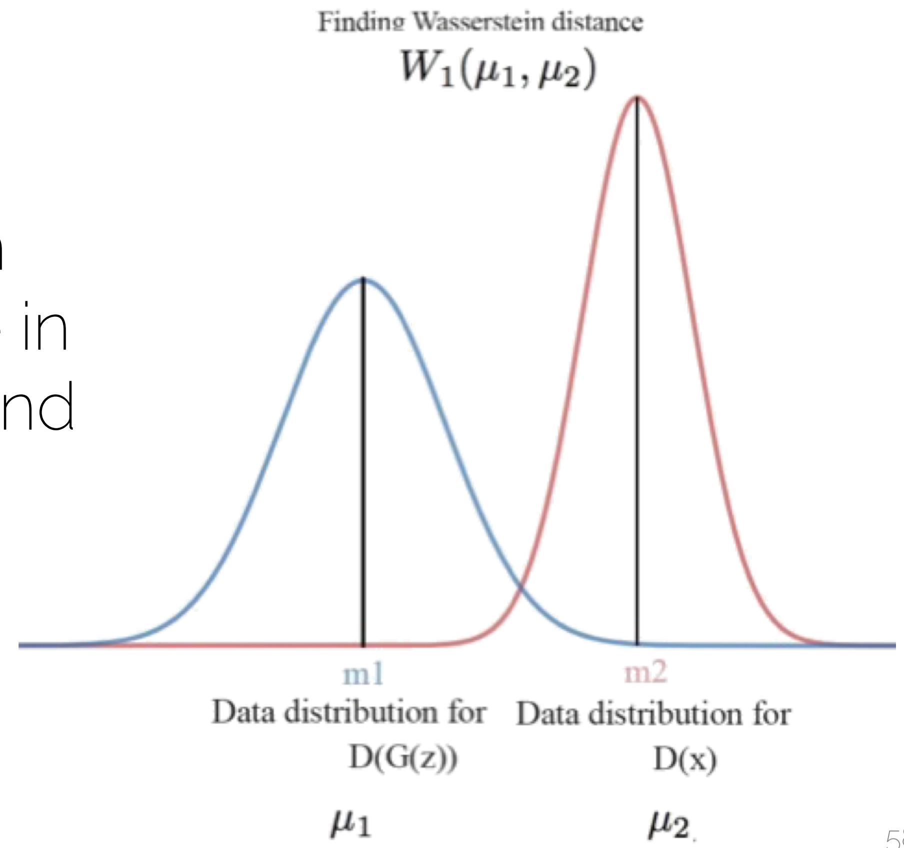
where $[\,\mathrm{u}\,]^+ = max(0, \mathrm{u}\,)$

https://medium.com/@jonathan_hui/gan-energy-based-gan-ebgan-boundary-equilibrium-gan-began-4662cceb7824

GAN Losses: BEGAN

- Similar to EBGAN

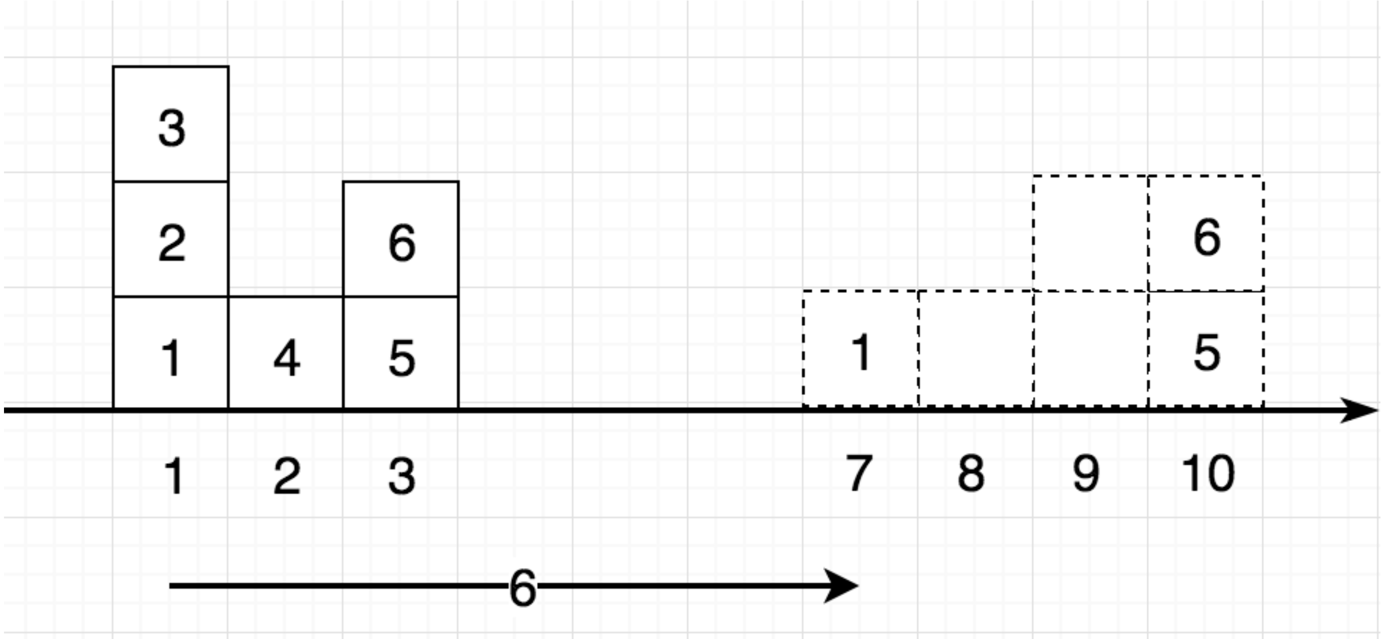
- Instead of reconstruction loss, measure difference in data distribution of real and generated images



https://medium.com/@jonathan_hui/gan-energy-based-gan-ebgan-boundary-equilibrium-gan-began-4662cceb7824

GAN Losses: WGAN

- Earth Mover Distance / Wasserstein Distance



Minimum amount of work to move earth from p(x) to q(x)

https://medium.com/@jonathan_hui/gan-wasserstein-gan-wgan-gp-6a1a2aa1b490

GAN Losses: WGAN

- Formulate EMD via it's dual:

$$W(\mathbb{P}_r, \mathbb{P}_\theta) = \sup_{\|f\|_L \leq 1} \mathbb{E}_{x \sim \mathbb{P}_r}[f(x)] - \mathbb{E}_{x \sim \mathbb{P}_\theta}[f(x)]$$

$$|f(x_1) - f(x_2)| \leq |x_1 - x_2|.$$

1-Lipschitz function: upper bound between densities

https://medium.com/@jonathan_hui/gan-wasserstein-gan-wgan-gp-6a1a2aa1b490

GAN Losses: WGAN

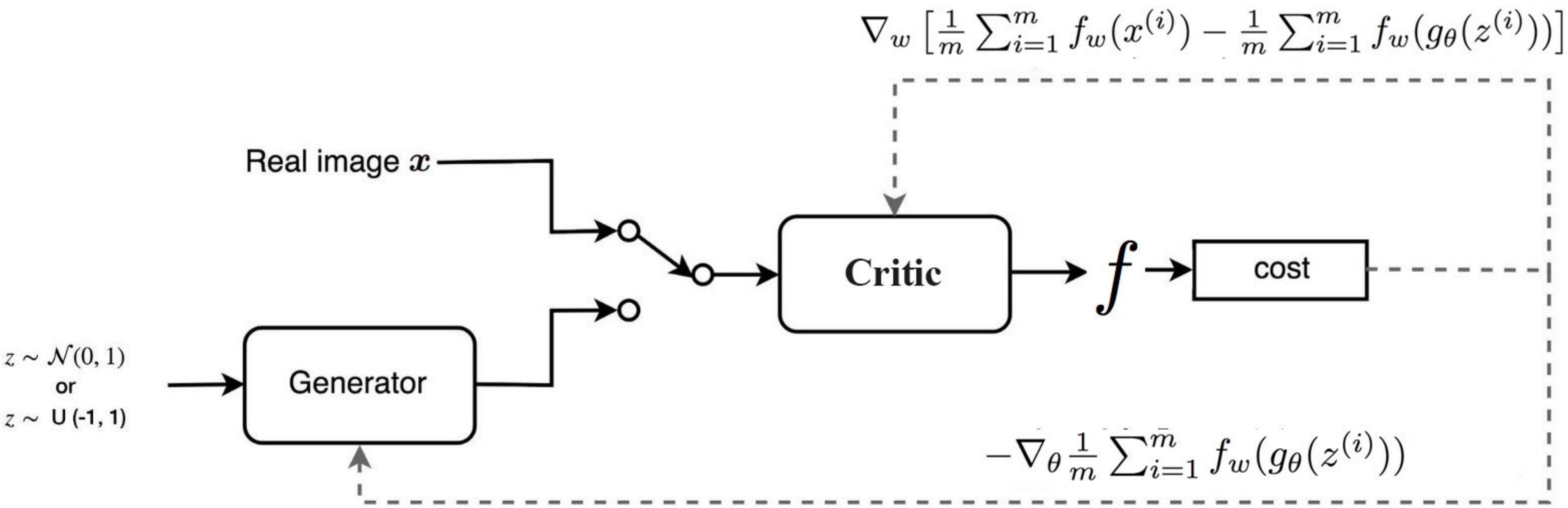
$$|f(x_1) - f(x_2)| \leq |x_1 - x_2|.$$

f is a critic function, defined by a neural network
-> f needs to be 1-Lipschitz; WGAN restricts max weight value in f; weights of the discriminator must be within a certain range controlled by hyperparameters c

$$w \leftarrow w + \alpha \cdot \mathrm{RMSProp}(w, g_w)$$
$$w \leftarrow \mathrm{clip}(w, -c, c)$$

https://medium.com/@jonathan_hui/gan-wasserstein-gan-wgan-gp-6a1a2aa1b490

GAN Losses: WGAN



GAN Losses: WGAN

	Discriminator/Critic	Generator
GAN	$\nabla_{\theta_d} \frac{1}{m} \sum_{i=1}^{m} \left[\log D\left(\boldsymbol{x}^{(i)}\right) + \log\left(1 - D\left(G\left(\boldsymbol{z}^{(i)}\right)\right)\right)\right]$	$\nabla_{\theta_g} \frac{1}{m} \sum_{i=1}^{m} -\log\left(D\left(G\left(\boldsymbol{z}^{(i)}\right)\right)\right)$
WGAN	$\nabla_w \frac{1}{m} \sum_{i=1}^{m} \left[f\left(x^{(i)}\right) - f\left(G\left(z^{(i)}\right)\right)\right]$	$\nabla_\theta \frac{1}{m} \sum_{i=1}^{m} -f\left(G\left(z^{(i)}\right)\right)$

https://medium.com/@jonathan_hui/gan-wasserstein-gan-wgan-gp-6a1a2aa1b490

GAN Losses: WGAN

Algorithm 1 WGAN, our proposed algorithm. All experiments in the paper used the default values $\alpha = 0.00005$, $c = 0.01$, $m = 64$, $n_{\text{critic}} = 5$.

Require: : α, the learning rate. c, the clipping parameter. m, the batch size. n_{critic}, the number of iterations of the critic per generator iteration.

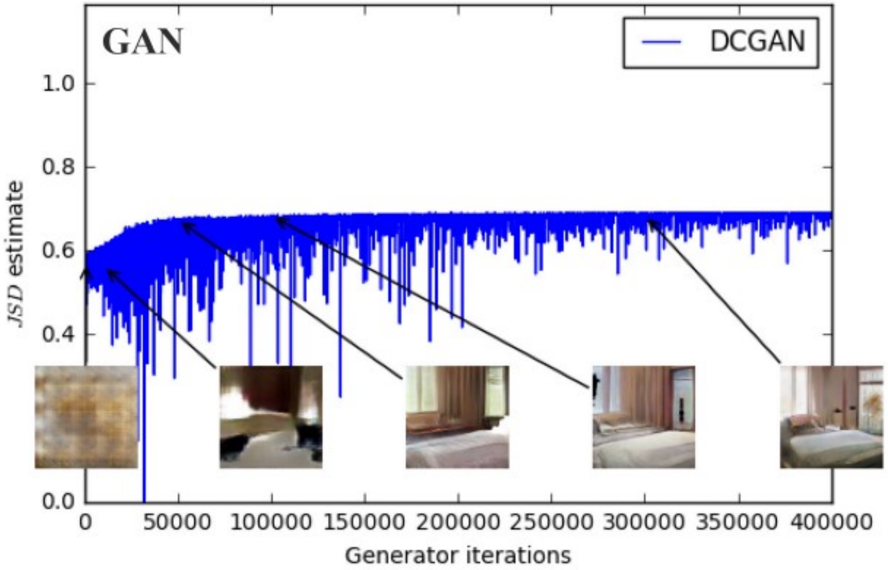
Require: : w_0, initial critic parameters. θ_0, initial generator's parameters.

while θ has not converged **do**
- **for** $t = 0, ..., n_{\text{critic}}$ **do**
  - Sample $\{x^{(i)}\}_{i=1}^m \sim \mathbb{P}_r$ a batch from the real data.
  - Sample $\{z^{(i)}\}_{i=1}^m \sim p(z)$ a batch of prior samples.
  - $g_w \leftarrow \nabla_w \left[\frac{1}{m}\sum_{i=1}^m f_w(x^{(i)}) - \frac{1}{m}\sum_{i=1}^m f_w(g_\theta(z^{(i)}))\right]$
  - $w \leftarrow w + \alpha \cdot \text{RMSProp}(w, g_w)$
  - $w \leftarrow \text{clip}(w, -c, c)$
- **end for**
- Sample $\{z^{(i)}\}_{i=1}^m \sim p(z)$ a batch of prior samples.
- $g_\theta \leftarrow -\nabla_\theta \frac{1}{m}\sum_{i=1}^m f_w(g_\theta(z^{(i)}))$
- $\theta \leftarrow \theta - \alpha \cdot \text{RMSProp}(\theta, g_\theta)$

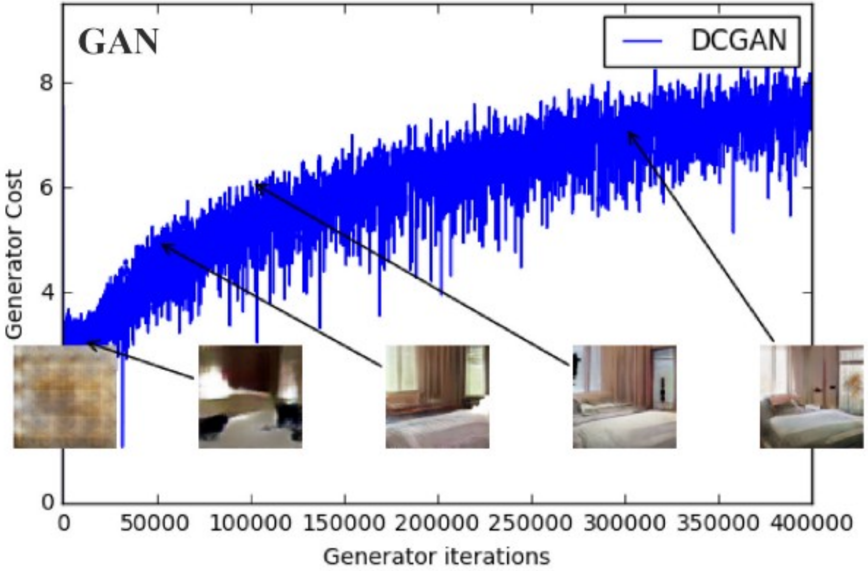
end while

https://medium.com/@jonathan_hui/gan-wasserstein-gan-wgan-gp-6a1a2aa1b490

GAN Losses: WGAN



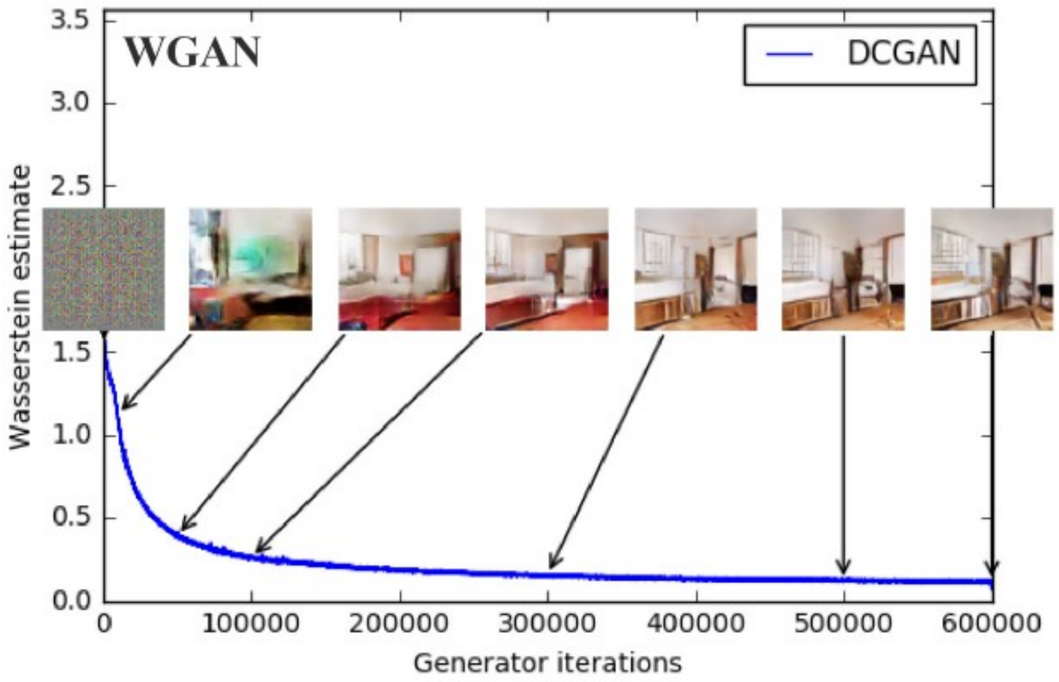
$$\frac{1}{m}\sum_{i=1}^{m}\log\left(1-D\left(G\left(z^{(i)}\right)\right)\right)$$



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

https://medium.com/@jonathan_hui/gan-wasserstein-gan-wgan-gp-6a1a2aa1b490

GAN Losses: WGAN



$$\frac{1}{m}\sum_{i=1}^{m} f_w(x^{(i)}) - \frac{1}{m}\sum_{i=1}^{m} f_w(g_\theta(z^{(i)}))$$

https://medium.com/@jonathan_hui/gan-wasserstein-gan-wgan-gp-6a1a2aa1b490

GAN Losses: WGAN

+ mitigates mode collapse
+ generator still learns when critic performs well
+ actual convergence

- Enforcing Lipschitz constraint is difficult
- Weight clipping is “terrible”
  -> too high: takes long time to reach limit; slow training
  -> too small: vanishing gradients when layers are big

GAN Losses

- Many more variations!!!
- High-level understanding: “loss” is a meta loss to train the actual loss (i.e., D) to provide gradients for G
- Always start simple: if things don’t converge, don’t randomly shuffle loss around; always try easy things first (AE, VAE, ‘simple heuristic’ GAN)

Next Lectures

- Next Lectures: more on Generative models
  - Conditional GANs (cGANs)!
  - Neural Rendering

- Keep working on the projects!

Thanks ☺