

# CS294-158 Deep Unsupervised Learning

## Lecture 5 Implicit Models -- Generative Adversarial Networks (GANs)



Pieter Abbeel, Wilson Yan, Kevin Frans, Philipp Wu

# Outline

---

- Motivation
- Original GAN (Goodfellow et al, 2014)
- Evaluation: Parzen, Inception, Frechet
- Some Theory: Bayes-optimal Discriminator; Jensen-Shannon Divergence; Mode Collapse; Avoiding Saturation
- GAN Progression:
  - DC GAN (Radford et al, 2016)
  - Improved Training of GANs (Salimans et al, 2016)
  - WGAN, WGAN-GP, Progressive GAN, SN-GAN, SAGAN
  - BigGAN, BigGAN-Deep, StyleGAN, StyleGAN-v2, VIB-GAN
- Creative Conditional GANs
- GANs and Representations
- GANs as Energy Models
- GANs and Optimal Transport, Implicit Likelihood Models, Moment Matching
- Other uses of Adversarial Loss: Transfer Learning, Fairness
- GANs and Imitation Learning

# Outline

---

- **Motivation & Definition of Implicit Models**
- Original GAN (Goodfellow et al, 2014)
- Evaluation: Parzen, Inception, Frechet
- Some Theory: Bayes-optimal Discriminator; Jensen-Shannon Divergence; Mode Collapse; Avoiding Saturation
- GAN Progression:
  - DC GAN (Radford et al, 2016)
  - Improved Training of GANs (Salimans et al, 2016)
  - WGAN, WGAN-GP, Progressive GAN, SN-GAN, SAGAN
  - BigGAN, BigGAN-Deep, StyleGAN, StyleGAN-v2, VIB-GAN
- Creative Conditional GANs
- GANs and Representations
- GANs as Energy Models
- GANs and Optimal Transport, Implicit Likelihood Models, Moment Matching
- Other uses of Adversarial Loss: Transfer Learning, Fairness
- GANs and Imitation Learning

# Motivation: GAN Progress



Ian Goodfellow  
@goodfellow\_ian

4.5 years of GAN progress on face generation.

[arxiv.org/abs/1406.2661](https://arxiv.org/abs/1406.2661) [arxiv.org/abs/1511.06434](https://arxiv.org/abs/1511.06434)

[arxiv.org/abs/1606.07536](https://arxiv.org/abs/1606.07536) [arxiv.org/abs/1710.10196](https://arxiv.org/abs/1710.10196)

[arxiv.org/abs/1812.04948](https://arxiv.org/abs/1812.04948)



4:40 PM · Jan 14, 2019 · Twitter Web Client

1.4K Retweets 3.8K Likes

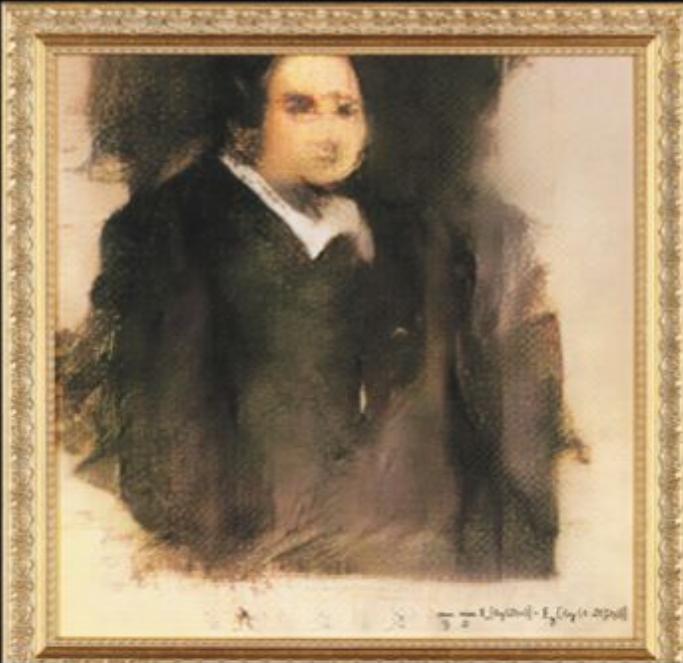
- Ian Goodfellow is first-author on the first GAN paper
- GAN is most prominent of Implicit Models

# Motivation: BigGAN



[BigGAN, Brock, Donahue, Simonyan, 2018]

# Motivation: GAN Art



$$\min_G \max_D \mathbb{E}_x[\log(D(x))] + \mathbb{E}_z[\log(1 - D(G(z)))]$$

10/25/2018

Sold at Christies for \$432,500

# So far...

---

- Autoregressive models
  - MADE, PixelRNN/CNN, Gated PixelCNN, PixelSNAIL
- Flow models
  - Autoregressive Flows, NICE, RealNVP, Glow, Flow++
- Latent Variable Models
  - VAE, IWAE, VQ-VAE
- **Common aspect:** Likelihood-based models
  - exact (autoregressive and flows)
  - approximate (VAE)

# Generative Models

---

- Sample
- Evaluate likelihood
- Train
- Representation

→ What if all we care about is sampling?

# Building a sampler

- How about this sampler?

```
import glob, cv2, numpy as np
files = glob.glob('*jpg')
def _sample():
    idx = np.random.randint(len(files))
    return cv2.imread(files[idx])
def sample(*, n_samples):
    samples = np.array([\_sample() for _ in range(n_samples)])
    return samples
```

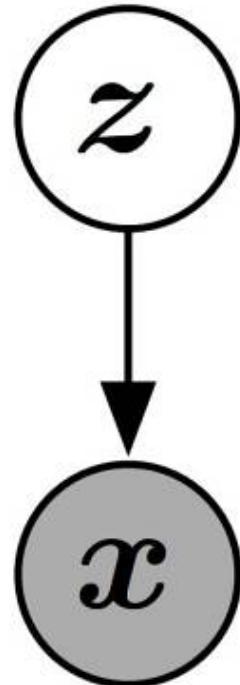
# Building a sampler

---

- You don't just want to sample the exact data points you have.
- You want to build a generative model that can understand the underlying distribution of data points and
  - smoothly interpolate across the training samples
  - output samples similar but not the same as training data samples
  - output samples representative of the underlying factors of variation in the training distribution.
  - Example: digits with unseen strokes, faces with unseen poses, etc.

# Implicit Models

- Sample  $z$  from a fixed noise source distribution (uniform or gaussian).
- Pass the noise through a deep neural network to obtain a sample  $x$ .
- Sounds familiar? Right:
  - Flow Models
  - VAE
- *What's going to be different here?*
  - *Learning the deep neural network **without** explicit density estimation*



# Implicit Models

---

- Given samples from data distribution  $p_{\text{data}} : x_1, x_2, \dots, x_n$
- Given a sampler  $q_\phi(z) = \text{DNN}(z; \phi)$  where  $z \sim p(z)$
- $x = q_\phi(z)$  induces a density function  $p_{\text{model}}$
- Do not have an explicit form for  $p_{\text{data}}$  or  $p_{\text{model}}$ ; can only draw samples
- Make  $p_{\text{model}}$  as close to  $p_{\text{data}}$  as possible by learning an appropriate  $\phi$

# Departure from maximum likelihood

---

- We need some measure of how far apart  $p_{\text{data}}$  and induced  $p_{\text{model}}$  are
- With density models, we used  $KL(p_{\text{data}} || p_{\text{model}})$  which gave us the objective  $\mathbb{E}_{x \sim p_{\text{data}}} [\log p_\theta(x)]$  (discarding the term independent of  $\theta$ ) where we explicitly modeled  $p_{\text{model}}$  as  $p_\theta(x)$
- Not having an explicit  $p_\theta(x)$  requires us to come up distance measures that potentially behave differently from maximum likelihood.

# Outline

---

- Motivation & Definition of Implicit Models
- **Original GAN (Goodfellow et al, 2014)**
- Evaluation: Parzen, Inception, Frechet
- Some Theory: Bayes-optimal Discriminator; Jensen-Shannon Divergence; Mode Collapse; Avoiding Saturation
- GAN Progression:
  - DC GAN (Radford et al, 2016)
  - Improved Training of GANs (Salimans et al, 2016)
  - WGAN, WGAN-GP, Progressive GAN, SN-GAN, SAGAN
  - BigGAN, BigGAN-Deep, StyleGAN, StyleGAN-v2, VIB-GAN
- Creative Conditional GANs
- GANs and Representations
- GANs as Energy Models
- GANs and Optimal Transport, Implicit Likelihood Models, Moment Matching
- Other uses of Adversarial Loss: Transfer Learning, Fairness
- GANs and Imitation Learning

# Generative Adversarial Networks

## Generative Adversarial Nets

Ian J. Goodfellow\*, Jean Pouget-Abadie†, Mehdi Mirza, Bing Xu, David Warde-Farley,  
Sherjil Ozair‡, Aaron Courville, Yoshua Bengio§

Département d'informatique et de recherche opérationnelle  
Université de Montréal  
Montréal, QC H3C 3J7



Ian Goodfellow

FOLLOW

DeepMind

Verified email at deepmind.com - [Homepage](#)

Deep Learning

TITLE	CITED BY	YEAR
<a href="#">Generative adversarial networks</a> I Goodfellow, J Pouget-Abadie, M Mirza, B Xu, D Warde-Farley, S Ozair, ... <i>Advances in neural information processing systems 27</i>	72471 *	2014

# Generative Adversarial Networks

$$\min_G \max_D \mathbb{E}_{x \sim p_{\text{data}}} [\log D(x)] + \mathbb{E}_{z \sim p(z)} [\log(1 - D(G(z)))]$$

- Two player minimax game between generator (G) and discriminator (D)
- (D) tries to maximize the log-likelihood for the binary classification problem
  - data: real (1)
  - generated: fake (0)
- (G) tries to minimize the log-probability of its samples being classified as “fake” by the discriminator (D)

# Generative Adversarial Networks

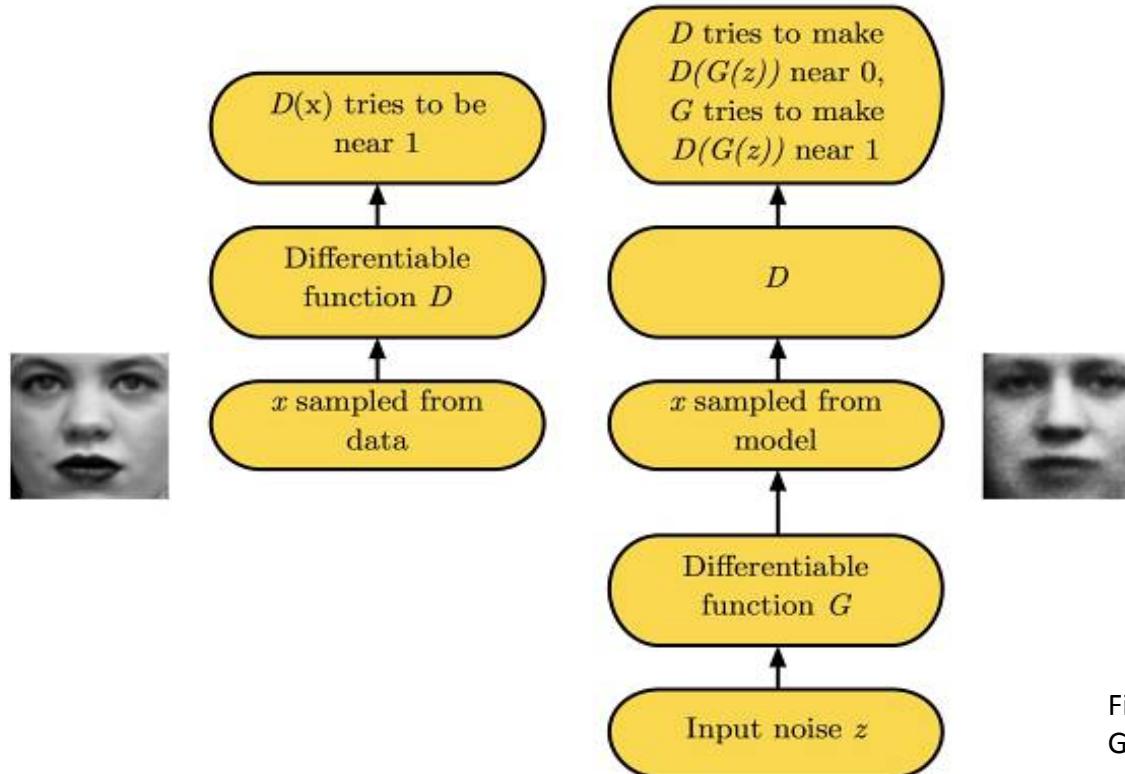


Figure from NeurIPS 2016  
GAN Tutorial (Goodfellow)

# GANs - Pseudocode

---

**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**

**for**  $k$  steps **do**

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

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

**end for**

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

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

**end for**

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

---

[Goodfellow et al 2014]

# GAN

---

See it in action: <https://poloclub.github.io/ganlab/>

# GAN samples from 2014



a)



b)



c)



d)

Figure from Goodfellow et al 2014

# How to evaluate?

---

- Evaluation for GANs is still an open problem
- Unlike density models, you cannot report explicit likelihood estimates on test sets.

# Parzen-Window density estimator

- Also known as Kernel Density Estimator (KDE)
- An estimator with kernel  $K$  and bandwidth  $h$ :

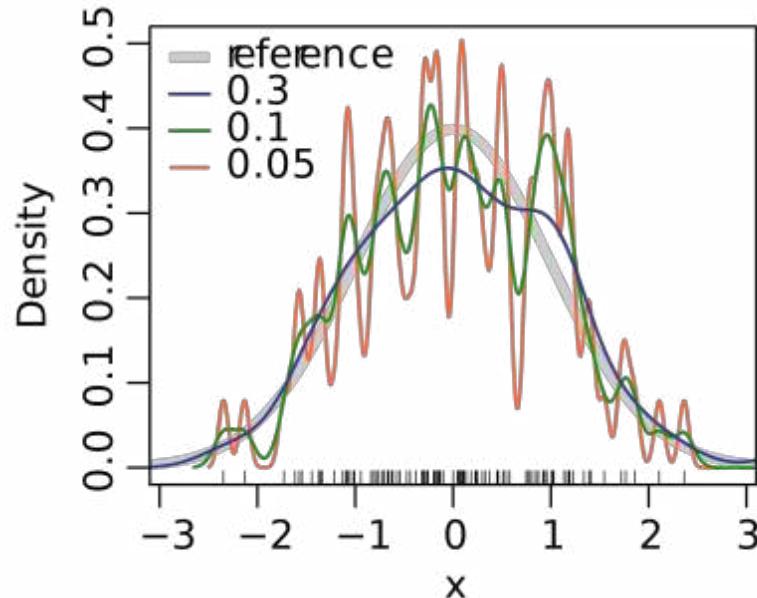
$$\hat{p}_h(x) = \frac{1}{nh} \sum_i K\left(\frac{x - x_i}{h}\right)$$

- In generative model evaluation,  $K$  is usually density function of standard Normal distribution

Bishop 2006

# Parzen-Window density estimator

- Bandwidth  $h$  matters
- Bandwidth  $h$  chosen according to validation set



Bishop 2006

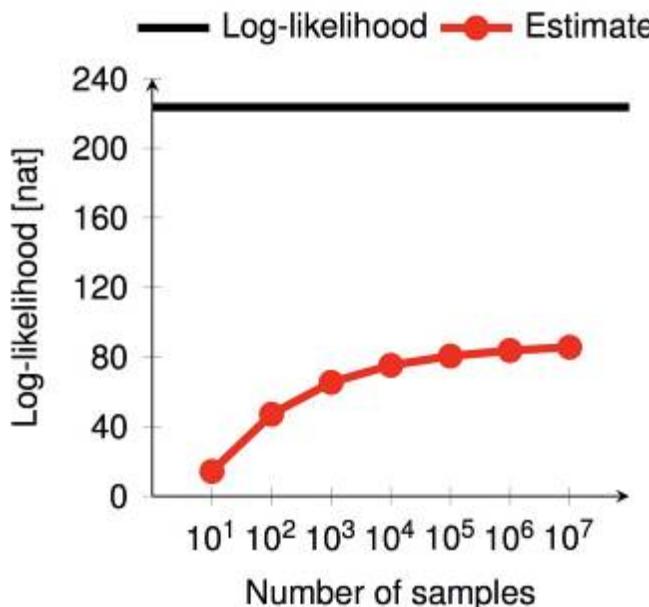
# Evaluation

Model	MNIST	TFD
DBN [3]	$138 \pm 2$	$1909 \pm 66$
Stacked CAE [3]	$121 \pm 1.6$	<b><math>2110 \pm 50</math></b>
Deep GSN [5]	$214 \pm 1.1$	$1890 \pm 29$
Adversarial nets	<b><math>225 \pm 2</math></b>	<b><math>2057 \pm 26</math></b>

Parzen Window density estimates (Goodfellow et al, 2014)

# Parzen-Window density estimator

- Parzen Window estimator can be unreliable



Model	Parzen est. [nat]
Stacked CAE	121
DBN	138
GMMN	147
Deep GSN	214
Diffusion	220
GAN	225
<b>True distribution</b>	<b>243</b>
GMMN + AE	282
<i>k</i> -means	313

[A note on the evaluation of generative models (Theis, Van den Oord, Bethge 2015)]

# Inception Score

---

- Can we side-step high-dim density estimation?
- One idea: good generators generate samples that are semantically diverse
- Semantics predictor: trained Inception Network v3
  - $p(y|x)$ ,  $y$  is one of the 1000 ImageNet classes
- Considerations:
  - each image  $x$  should have distinctly recognizable object ->  $p(y|x)$  should have low entropy
  - there should be as many classes generated as possible ->  $p(y)$  should have high entropy

# Inception Score

- Inception model:  $p(y|x)$
- Marginal label distribution:  $p(y) = \int_x p(y|x)p_g(x)$
- Inception Score:

$$\begin{aligned} \text{IS}(x) &= \exp(\mathbb{E}_{x \sim p_g} [D_{\text{KL}} [p(y|x) \parallel p(y)]]) \\ &= \exp(\mathbb{E}_{x \sim p_g, y \sim p(y|x)} [\log p(y|x) - \log p(y)]) \\ &= \exp(H(y) - H(y|x)) \end{aligned}$$

[Improved GAN: Salimans et al 2016]

# Inception Score

Samples				
Model	Real data	Our methods	-VBN+BN	-L+HA
Score $\pm$ std.	$11.24 \pm .12$	$8.09 \pm .07$	$7.54 \pm .07$	$6.86 \pm .06$

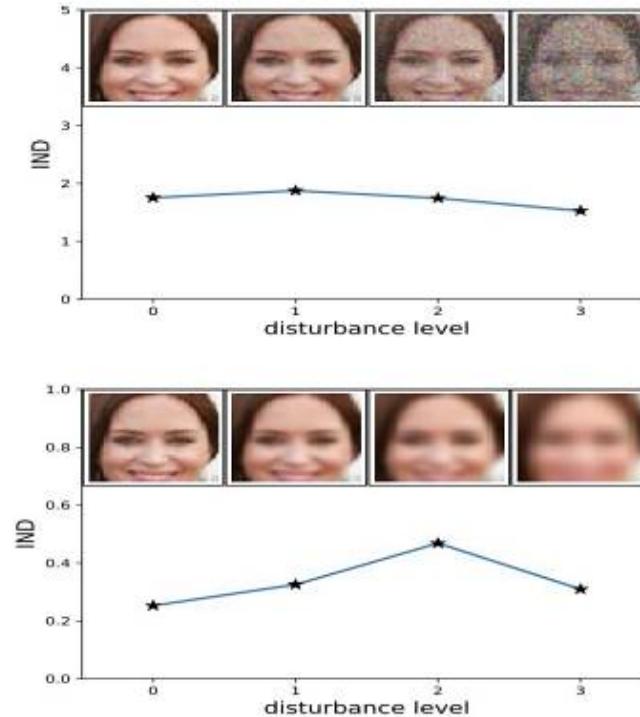
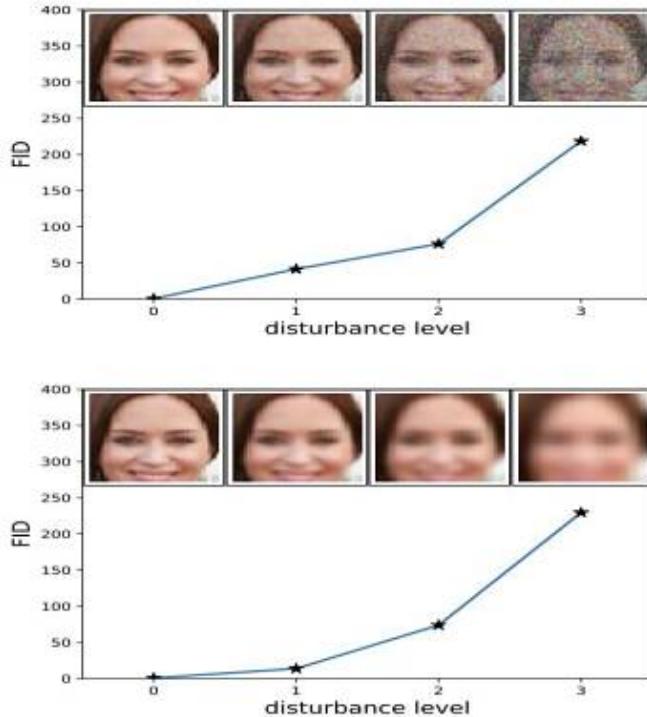
# Fréchet Inception Distance

- Inception Score doesn't sufficiently measure diversity: a list of 1000 images (one of each class) can obtain perfect Inception Score
- FID was proposed to capture more nuances
- Embed image  $x$  into some feature space (2048-dimensional activations of the Inception-v3 pool3 layer), then compare mean ( $m$ ) & covariance ( $C$ ) of those random features

$$d^2((\mathbf{m}, \mathbf{C}), (\mathbf{m}_w, \mathbf{C}_w)) = \|\mathbf{m} - \mathbf{m}_w\|_2^2 + \text{Tr}(\mathbf{C} + \mathbf{C}_w - 2(\mathbf{C}\mathbf{C}_w)^{1/2})$$

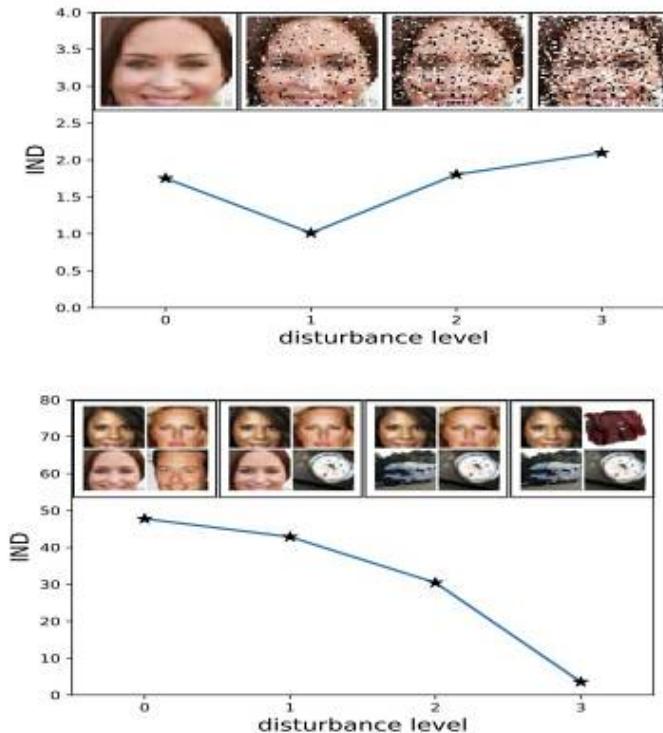
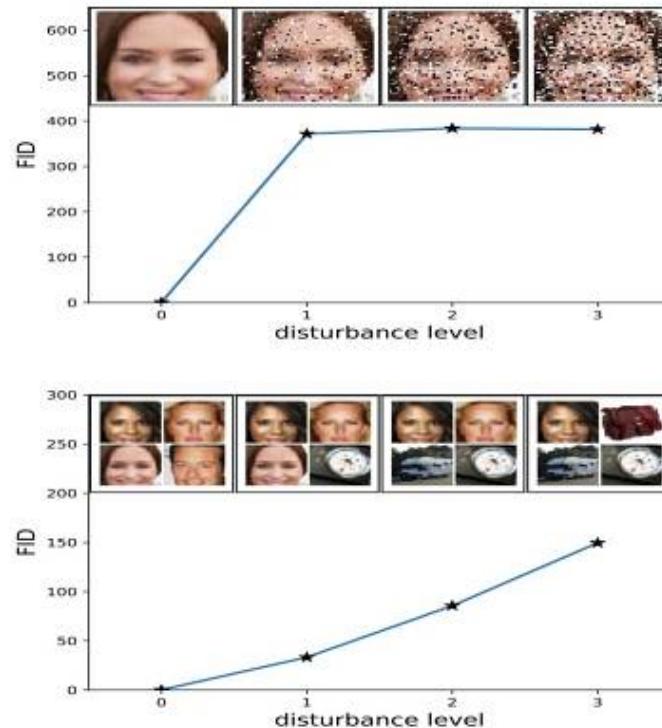
[Heusel et al, 2017]

# Fréchet Inception Distance



[Heusel et al, 2017]

# Fréchet Inception Distance



[Heusel et al, 2017]

# Generative Adversarial Networks

---

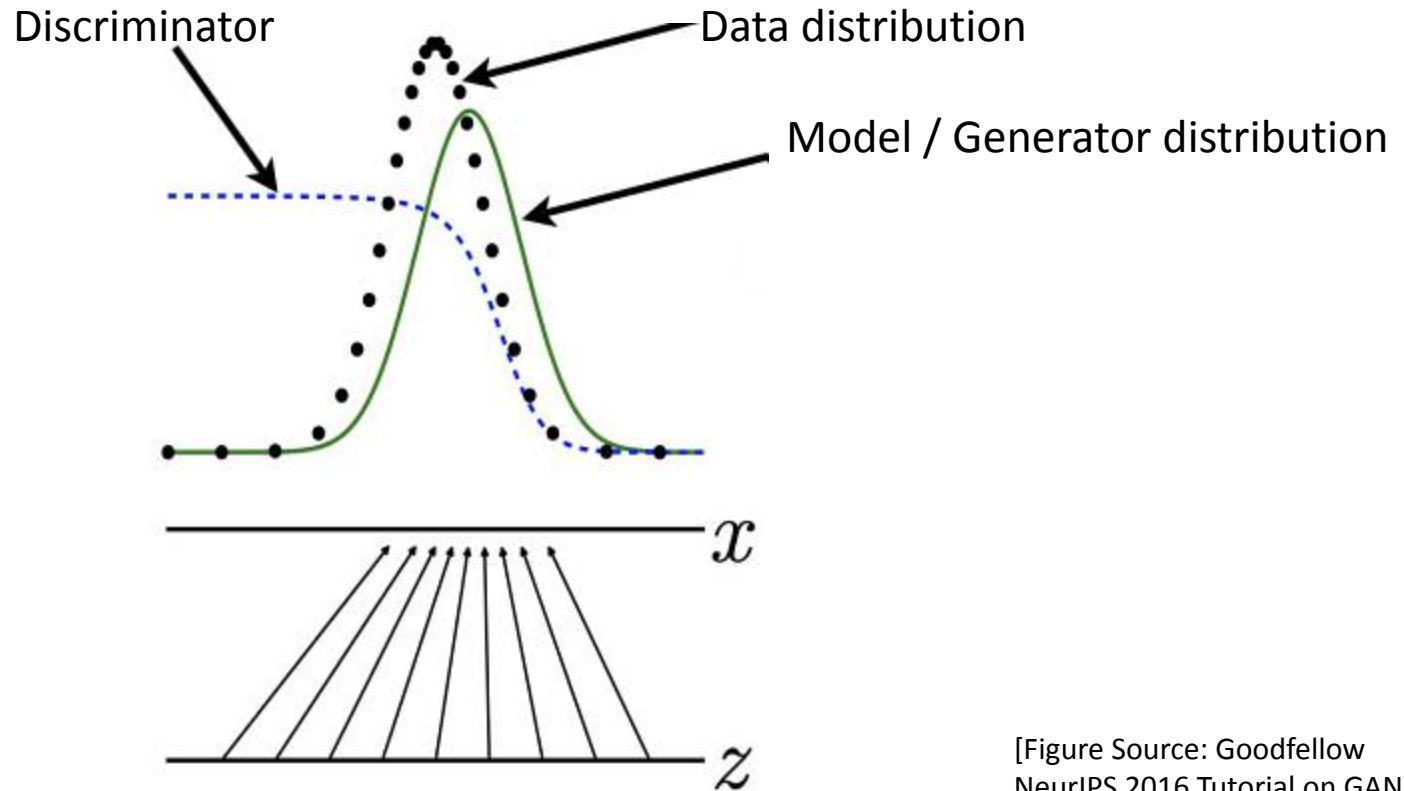
- Key pieces of GAN
  - Fast sampling
  - Notion of optimizing directly for what you care about - perceptual samples
  - No inference, i.e. no evaluation of  $p(\text{sample})$  or  $p(\text{data})$

# Outline

---

- Motivation & Definition of Implicit Models
- Original GAN (Goodfellow et al, 2014)
- Evaluation: Parzen, Inception, Frechet
- ***Some Theory: Bayes-optimal Discriminator; Jensen-Shannon Divergence; Mode Collapse; Avoiding Saturation***
- GAN Progression:
  - DC GAN (Radford et al, 2016)
  - Improved Training of GANs (Salimans et al, 2016)
  - WGAN, WGAN-GP, Progressive GAN, SN-GAN, SAGAN
  - BigGAN, BigGAN-Deep, StyleGAN, StyleGAN-v2, VIB-GAN
- Creative Conditional GANs
- GANs and Representations
- GANs as Energy Models
- GANs and Optimal Transport, Implicit Likelihood Models, Moment Matching
- Other uses of Adversarial Loss: Transfer Learning, Fairness
- GANs and Imitation Learning

# GAN: Bayes-Optimal Discriminator



[Figure Source: Goodfellow  
NeurIPS 2016 Tutorial on GANs]

# GAN: Bayes-Optimal Discriminator

- What's the optimal discriminator given generated and true distributions?

$$\begin{aligned} V(G, D) &= \mathbb{E}_{x \sim p_{\text{data}}} [\log D(x)] + \mathbb{E}_{z \sim p(z)} [\log(1 - D(G(z)))] \\ &= \int_x p_{\text{data}}(x) \log D(x) dx + \int_z p(z) \log(1 - D(G(z))) dz \\ &= \int_x p_{\text{data}}(x) \log D(x) dx + \int_x p_g(x) \log(1 - D(x)) dx \\ &= \int_x [p_{\text{data}}(x) \log D(x) + p_g(x) \log(1 - D(x))] dx \end{aligned}$$

$$\nabla_y [a \log y + b \log(1 - y)] = 0 \implies y^* = \frac{a}{a+b} \quad \forall \quad [a, b] \in \mathbb{R}^2 \setminus [0, 0]$$

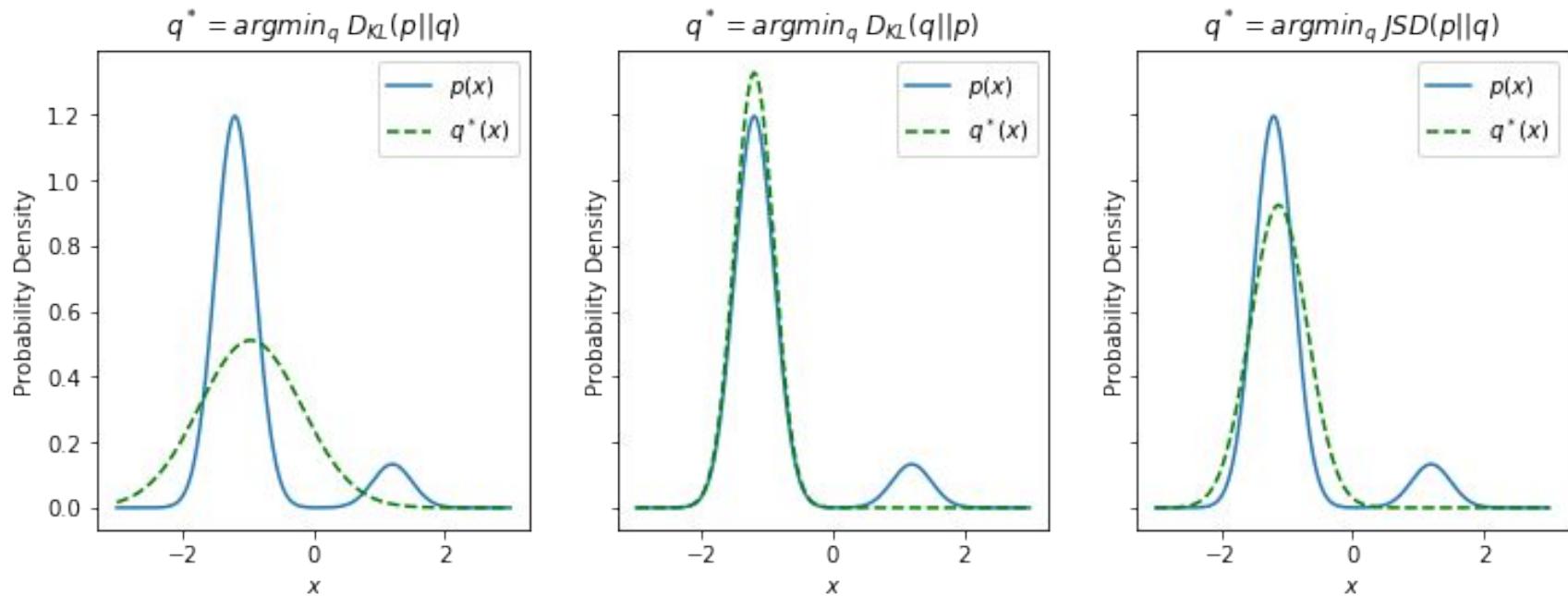
$$\implies D^*(x) = \frac{p_{\text{data}}(x)}{(p_{\text{data}}(x) + p_g(x))}$$

# GAN: Generator Objective under Bayes-Optimal Discriminator D\* ?

$$\begin{aligned} V(G, D^*) &= \mathbb{E}_{x \sim p_{\text{data}}} [\log D^*(x)] + \mathbb{E}_{x \sim p_g} [\log(1 - D^*(x))] \\ &= \mathbb{E}_{x \sim p_{\text{data}}} \left[ \log \frac{p_{\text{data}}(x)}{p_{\text{data}}(x) + p_g(x)} \right] + \mathbb{E}_{x \sim p_g} \left[ \log \frac{p_g(x)}{p_{\text{data}}(x) + p_g(x)} \right] \\ &= -\log(4) + \underbrace{KL \left( p_{\text{data}} \parallel \left( \frac{p_{\text{data}} + p_g}{2} \right) \right) + KL \left( p_g \parallel \left( \frac{p_{\text{data}} + p_g}{2} \right) \right)}_{(\text{Jensen-Shannon Divergence (JSD) of } p_{\text{data}} \text{ and } p_g) \geq 0} \end{aligned}$$

$$V(G^*, D^*) = -\log(4) \text{ when } p_g = p_{\text{data}}$$

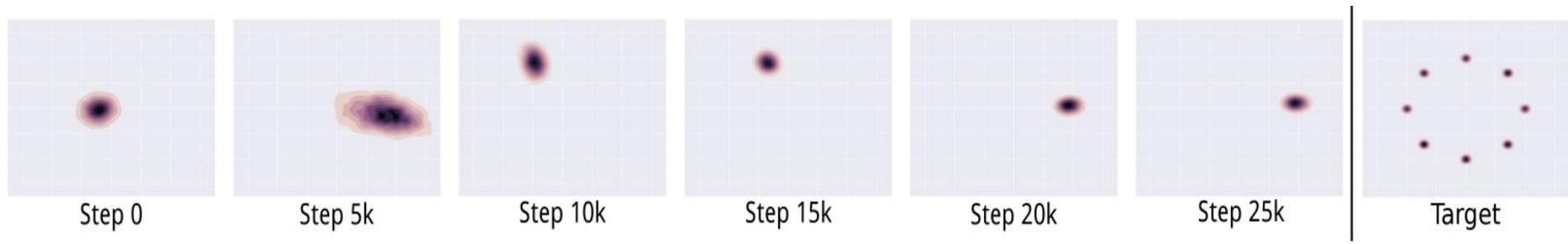
# KL, reverse KL, JSD



# Mode covering vs Mode seeking: Tradeoffs

- For compression, one would prefer to ensure all points in the data distribution are assigned probability mass.
- For generating good samples, blurring across modes spoils perceptual quality because regions outside the data manifold are assigned non-zero probability mass.
- Picking one mode without assigning probability mass on points outside can produce “better-looking” samples.
- **Caveat:** More expressive density models can place probability mass more accurately.  
Example: Using mixture of Gaussians as opposed to a single isotropic gaussian.

# Mode Collapse



Standard GAN training collapses when the true distribution is a mixture of gaussians (Figure from Metz et al 2016)

# Back to our GAN objective

## Recall

$$\min_G \max_D \underbrace{\mathbb{E}_{x \sim p_{\text{data}}} [\log D(x)] + \mathbb{E}_{z \sim p(z)} [\log(1 - D(G(z)))]}_{\text{Discriminator}}$$

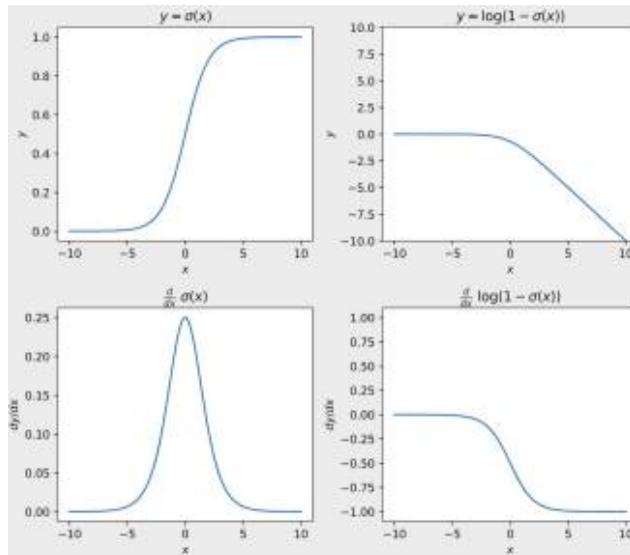
## Mini-Exercise

- Is it feasible to run the inner optimization to completion?
- For this specific objective, would it create problems if we were able to do so?

# Discriminator Saturation

- Generator samples confidently classified as fake by the discriminator receive no gradient for the generator update.

$$\nabla_{G(z)} \log(1 - D(G(z))) \text{ where } D(x) = \text{sigmoid}(x; \theta) = \sigma(x; \theta) \quad \nabla_x \sigma(x) = \sigma(x)(1 - \sigma(x))$$



# Avoiding Discriminator Saturation: (1) Alternating Optimization

- Alternate gradient steps on discriminator and generator objectives

$$L^{(D)}(\theta_D, \theta_G) = -\mathbb{E}_{x \sim p_{\text{data}}} [\log D(x; \theta_D)] - \mathbb{E}_{z \sim p(z)} [\log(1 - D(G(z; \theta_G), \theta_D))]$$

$$L^{(G)}(\theta_D, \theta_G) = \mathbb{E}_{z \sim p(z)} [\log(1 - D(G(z; \theta_G), \theta_D))]$$

$$\theta_D := \theta_D - \alpha^{(D)} \nabla_{\theta_D} L^{(D)}(\theta_D, \theta_G)$$

$$\theta_G := \theta_G - \beta^{(G)} \nabla_{\theta_G} L^{(G)}(\theta_D, \theta_G)$$

- Balancing these two updates is hard for the zero-sum game

## Avoiding Discriminator Saturation: (2) Non Saturating Formulation

$$L^{(D)} = -\mathbb{E}_{x \sim p_{\text{data}}} [\log D(x)] - \mathbb{E}_{z \sim p(z)} [\log(1 - D(G(z)))]$$

$$L^{(G)} = -L^D \equiv \min_G \mathbb{E}_{z \sim p(z)} \log(1 - D(G(z)))$$



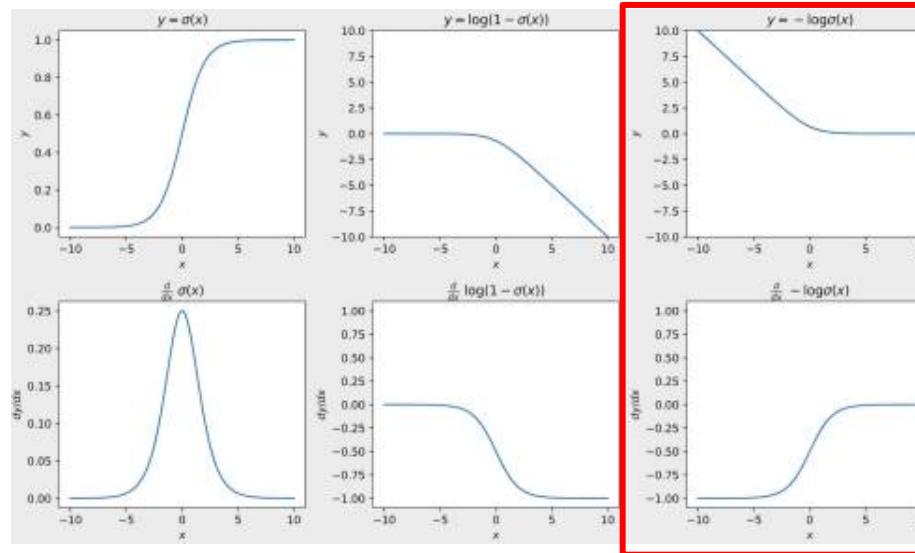
Not zero-sum

$$L^{(D)} = -\mathbb{E}_{x \sim p_{\text{data}}} [\log D(x)] - \mathbb{E}_{z \sim p(z)} [\log(1 - D(G(z)))]$$

$$L^{(G)} = -\mathbb{E}_{z \sim p(z)} \log(D(G(z))) \equiv \max_G \mathbb{E}_{z \sim p(z)} \log(D(G(z)))$$

# Avoiding Discriminator Saturation: (2) Non Saturating Formulation

- ORIGINAL ISSUE: Generator samples confidently classified as fake by the discriminator receive no gradient for the generator update.
- FIX: non-saturating loss for when discriminator confident about fake



# Outline

---

- Motivation & Definition of Implicit Models
- Original GAN (Goodfellow et al, 2014)
- Evaluation: Parzen, Inception, Frechet
- Some Theory: Bayes-optimal Discriminator; Jensen-Shannon Divergence; Mode Collapse; Avoiding Saturation
- **GAN Progression:**
  - *DC GAN (Radford et al, 2016)*
  - Improved Training of GANs (Salimans et al, 2016)
  - WGAN, WGAN-GP, Progressive GAN, SN-GAN, SAGAN
  - BigGAN, BigGAN-Deep, StyleGAN, StyleGAN-v2, VIB-GAN
- Creative Conditional GANs
- GANs and Representations
- GANs as Energy Models
- GANs and Optimal Transport, Implicit Likelihood Models, Moment Matching
- Other uses of Adversarial Loss: Transfer Learning, Fairness
- GANs and Imitation Learning

# Deep Convolutional GAN (DCGAN)

## UNSUPERVISED REPRESENTATION LEARNING WITH DEEP CONVOLUTIONAL GENERATIVE ADVERSARIAL NETWORKS

**Alec Radford & Luke Metz**

indico Research

Boston, MA

{alec,luke}@indico.io

**Soumith Chintala**

Facebook AI Research

New York, NY

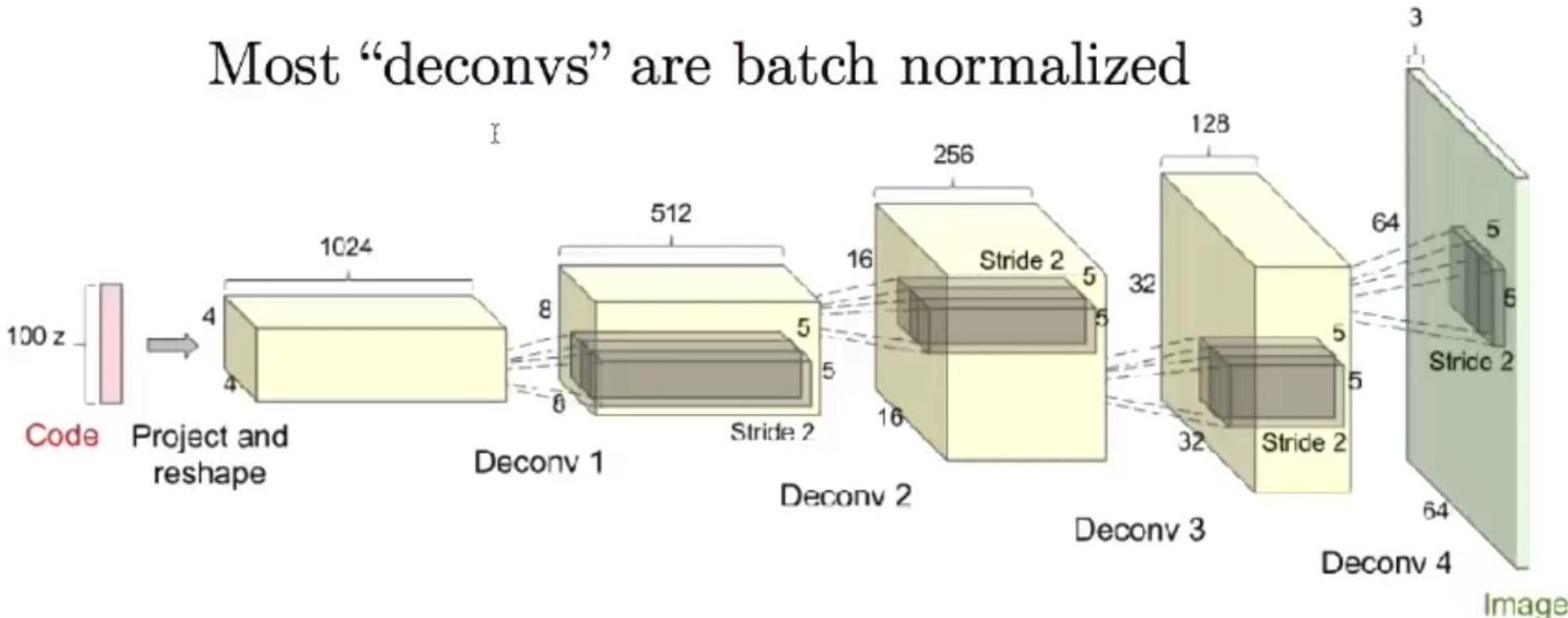
soumith@fb.com

### ABSTRACT

In recent years, supervised learning with convolutional networks (CNNs) has seen huge adoption in computer vision applications. Comparatively, unsupervised learning with CNNs has received less attention. In this work we hope to help bridge the gap between the success of CNNs for supervised learning and unsupervised learning. We introduce a class of CNNs called deep convolutional generative adversarial networks (DCGANs), that have certain architectural constraints, and demonstrate that they are a strong candidate for unsupervised learning. Training on various image datasets, we show convincing evidence that our deep convolutional adversarial pair learns a hierarchy of representations from object parts to scenes in both the generator and discriminator. Additionally, we use the learned features for novel tasks - demonstrating their applicability as general image representations.

# Deep Convolutional GAN (DCGAN)

Most “deconvs” are batch normalized



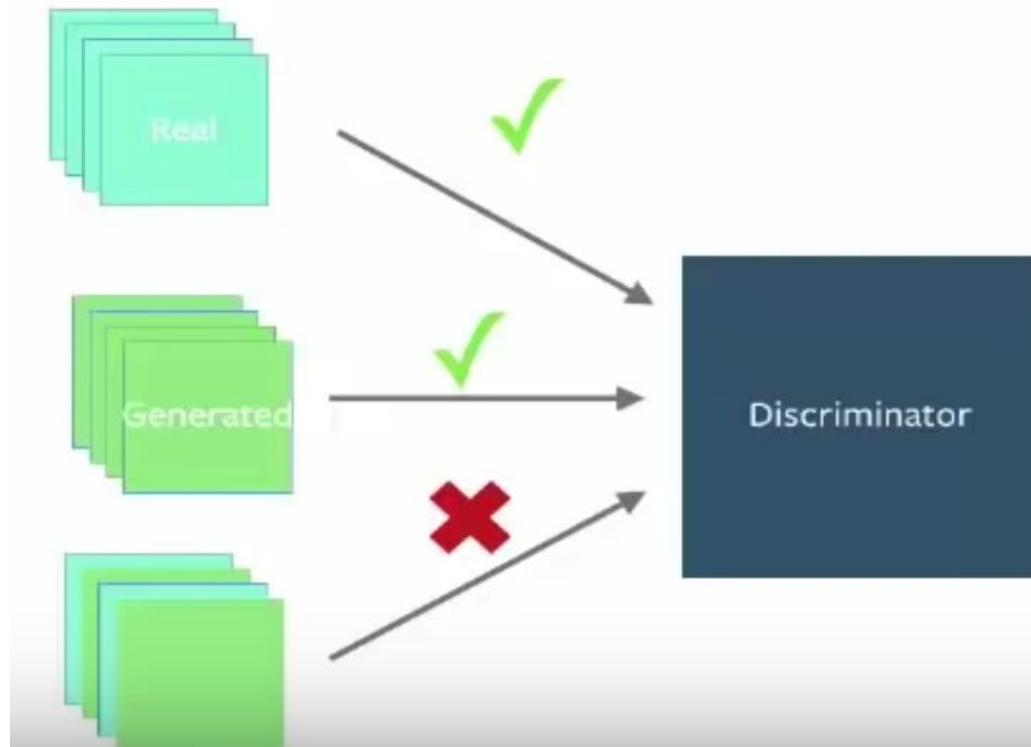
[Radford et al 2016]

# DCGAN - Architecture Design

- Supervised Learning CNNs not directly usable
  - Remove max-pooling and mean-pooling
  - Upsample using transposed convolutions in the generator
  - Downsample with strided convolutions and average pooling
  - Non-Linearity: ReLU for generator, Leaky-ReLU (0.2) for discriminator
  - Output Non-Linearity: tanh for Generator, sigmoid for discriminator
  - Batch Normalization used to prevent mode collapse
  - Batch Normalization is not applied at the output of G and input of D
- Optimization details
  - Adam: small LR - 2e-4; small momentum: 0.5, batch-size: 128

[Radford et al 2016]

# DCGAN Batch Norm



Chintala 2016

# DCGAN - Key Results

- Good samples on datasets with 3M images (Faces, Bedrooms) for the first time



[Radford et al 2016]

# DCGAN - Key Results



[Radford et al 2016]

# DCGAN - Key Results

- Smooth interpolations in high dimensions



[Radford et al 2016]

# DCGAN - Key Results

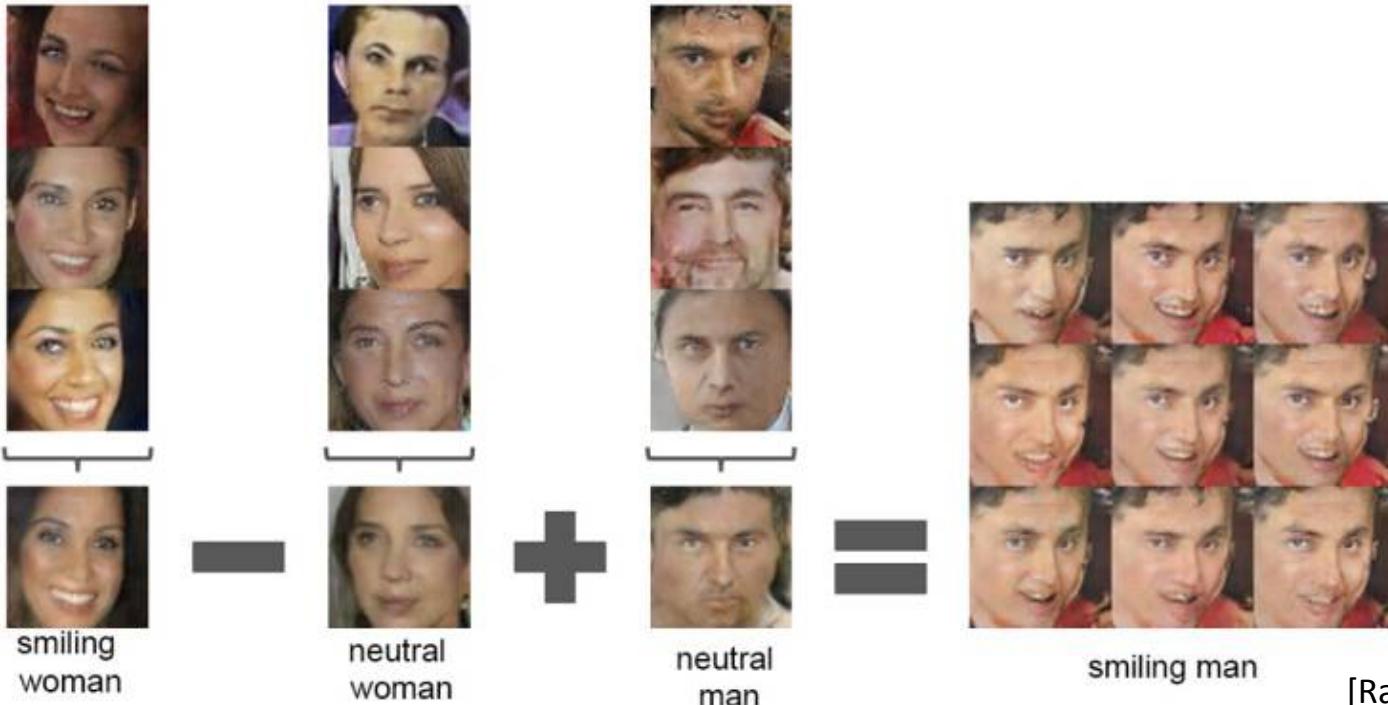
- Imagenet samples  
(32x32)



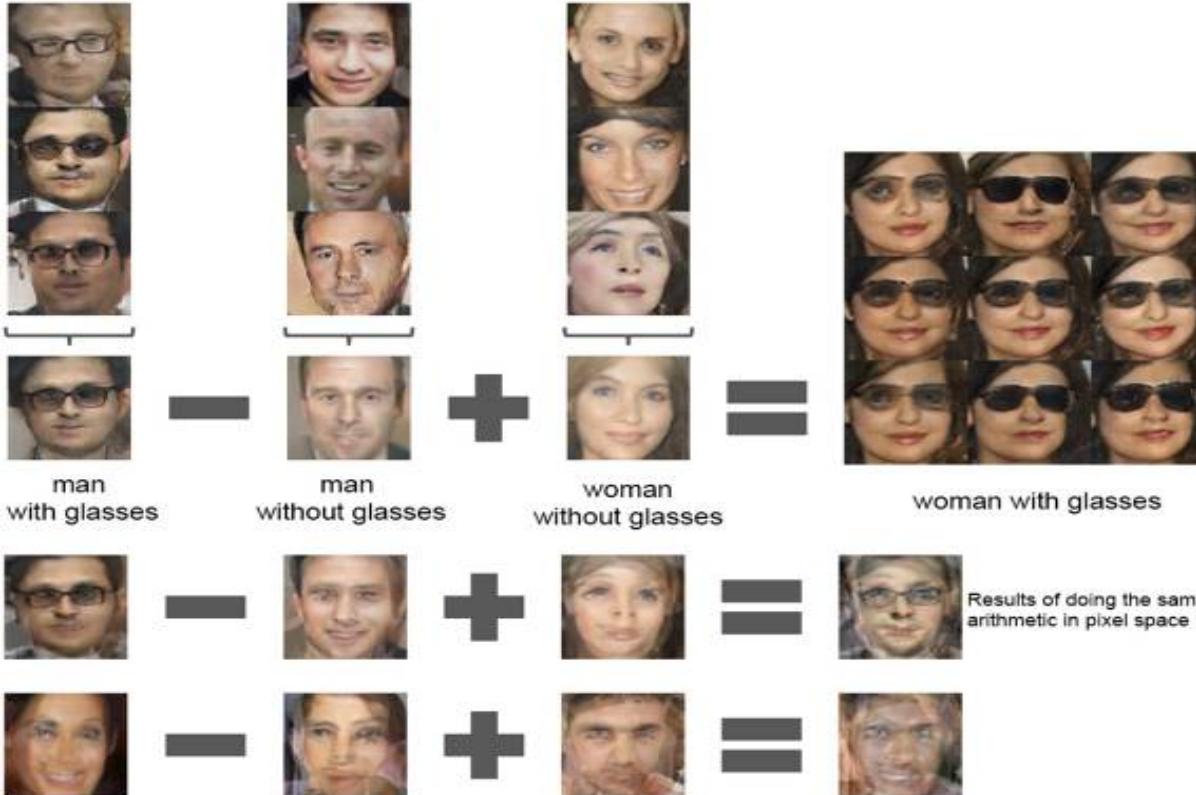
[Radford et al 2016]

# DCGAN - Key Results

## ■ Vector Arithmetic

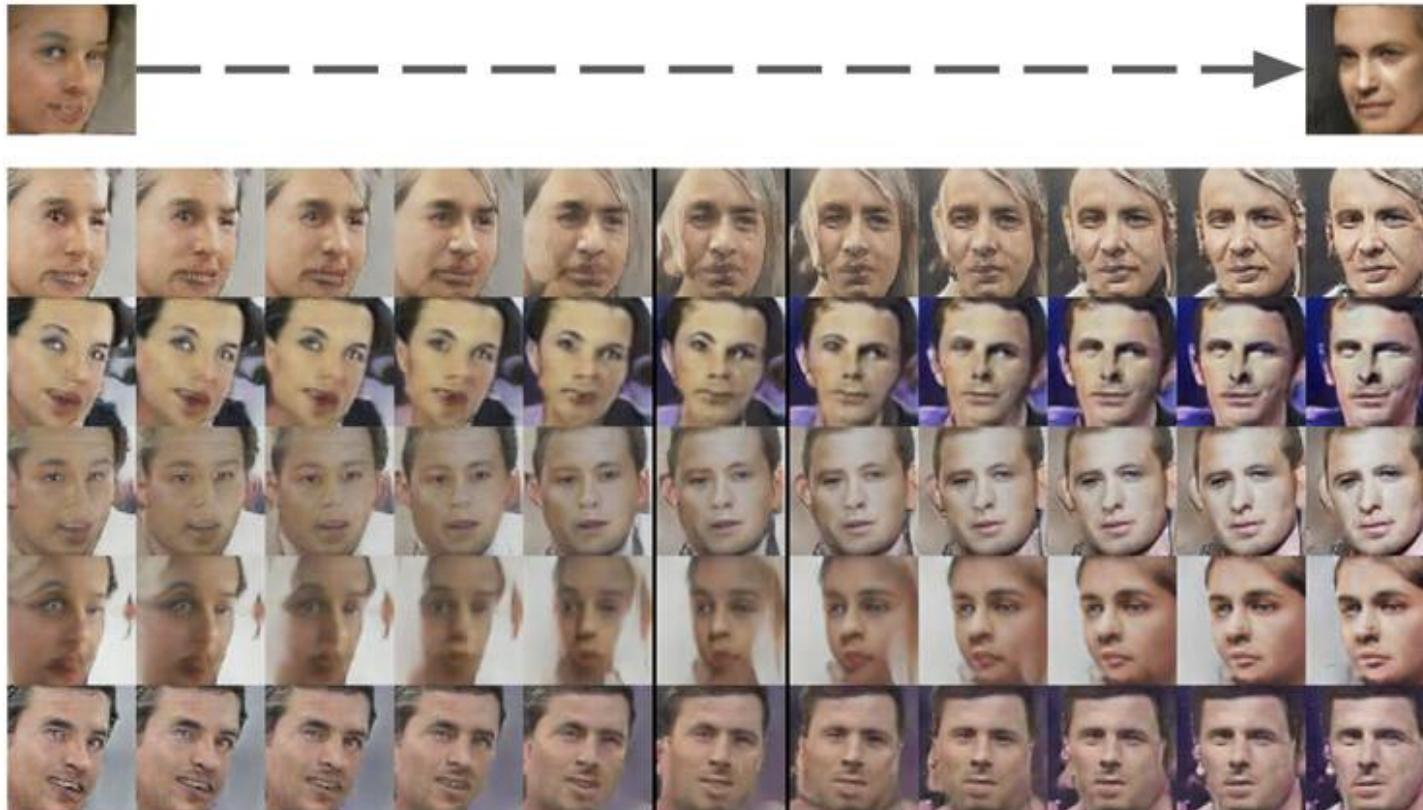


# DCGAN - Key Results



[Radford et al 2016]

# DCGAN - Key Results



[Radford et al 2016]

# DCGAN - Key Results

## Representation Learning

Model	Accuracy	Accuracy (400 per class)	max # of features units
1 Layer K-means	80.6%	63.7% ( $\pm 0.7\%$ )	4800
3 Layer K-means Learned RF	82.0%	70.7% ( $\pm 0.7\%$ )	3200
View Invariant K-means	81.9%	72.6% ( $\pm 0.7\%$ )	6400
Exemplar CNN	84.3%	77.4% ( $\pm 0.2\%$ )	1024
DCGAN (ours) + L2-SVM	82.8%	73.8% ( $\pm 0.4\%$ )	512

[Radford et al 2016]

# DCGAN - Conclusions

---

- Incredible samples for any generative model
- GANs could be made to work well with architecture details
- Perceptually good samples and interpolations
- Representation Learning
- **Problems to address:**
  - Unstable training
  - Brittle architecture / hyperparameters

# Outline

---

- Motivation & Definition of Implicit Models
- Original GAN (Goodfellow et al, 2014)
- Evaluation: Parzen, Inception, Frechet
- Some Theory: Bayes-optimal Discriminator; Jensen-Shannon Divergence; Mode Collapse; Avoiding Saturation
- **GAN Progression:**
  - DC GAN (Radford et al, 2016)
  - ***Improved Training of GANs (Salimans et al, 2016)***
  - WGAN, WGAN-GP, Progressive GAN, SN-GAN, SAGAN
  - BigGAN, BigGAN-Deep, StyleGAN, StyleGAN-v2, VIB-GAN
- Creative Conditional GANs
- GANs and Representations
- GANs as Energy Models
- GANs and Optimal Transport, Implicit Likelihood Models, Moment Matching
- Other uses of Adversarial Loss: Transfer Learning, Fairness
- GANs and Imitation Learning

# Improved training of GANs

- Minibatch discrimination
- Feature Matching
- Historical Averaging
- Virtual batch normalization
- One-sided label smoothing

---

## Improved Techniques for Training GANs

---

**Tim Salimans**

tim@openai.com

**Ian Goodfellow**

ian@openai.com

**Wojciech Zaremba**

woj@openai.com

**Vicki Cheung**

vicki@openai.com

**Alec Radford**

alec.radford@gmail.com

**Xi Chen**

peter@openai.com

Salimans 2016

# Improved training of GANs

## ■ Minibatch discrimination

$$\mathbf{f}(\mathbf{x}_i) \in \mathbb{R}^A \quad T \in \mathbb{R}^{A \times B \times C} \quad M_i \in \mathbb{R}^{B \times C}$$

$$c_b(\mathbf{x}_i, \mathbf{x}_j) = \exp(-\|M_{i,b} - M_{j,b}\|_{L_1}) \in \mathbb{R}$$

$$o(\mathbf{x}_i)_b = \sum_{j=1}^n c_b(\mathbf{x}_i, \mathbf{x}_j) \in \mathbb{R}$$

$$o(\mathbf{x}_i) = [o(\mathbf{x}_i)_1, o(\mathbf{x}_i)_2, \dots, o(\mathbf{x}_i)_B] \in \mathbb{R}^B$$

$$o(\mathbf{X}) \in \mathbb{R}^{n \times B}$$

Salimans 2016

Allows to incorporate side information from other samples and is superior to feature matching in the unconditional setting.  
Helps addressing mode collapse by allowing discriminator to detect if the generated samples are too close to each other.

# Improved training of GANs

## ■ Feature Matching

$$||\mathbb{E}_{x \sim p_{\text{data}}} f(x) - \mathbb{E}_{z \sim p(z)} f(G(z))||^2$$

Add this to Generator objective

Salimans 2016

# Improved training of GANs

## ■ Historical Averaging

$$||\theta - \frac{1}{t} \sum_{i=1}^t \theta[i]||^2$$

Salimans 2016

# Improved training of GANs

## ■ Virtual Batch Normalization

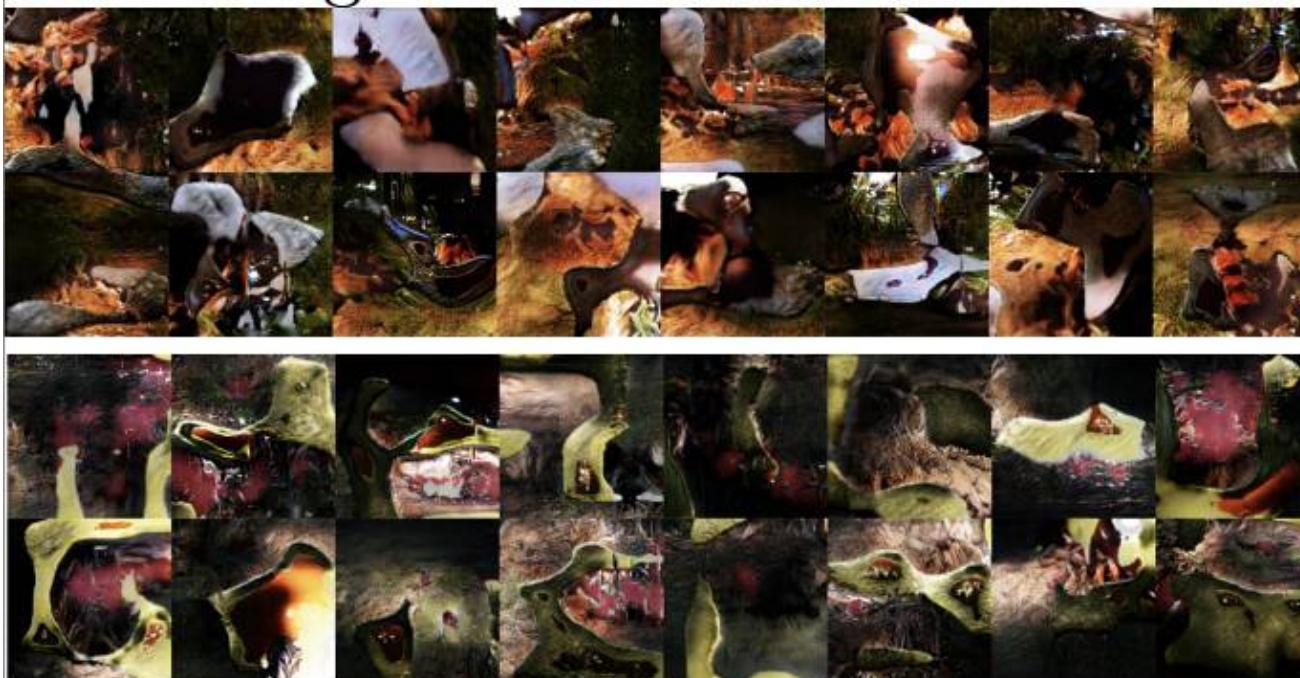


Figure source:  
NeurIPS tutorial  
Goodfellow 2016

# Improved training of GANs

## ■ Virtual Batch Normalization

- Use a reference batch (fixed) to compute normalization statistics
- Construct a batch containing the sample and reference batch

# Improved training of GANs

## ■ One-sided label smoothing

Default discriminator cost:

```
cross_entropy(1., discriminator(data))  
+ cross_entropy(0., discriminator(samples))
```



One-sided label smoothed cost (Salimans et al 2016):

```
cross_entropy(.9, discriminator(data))  
+ cross_entropy(0., discriminator(samples))
```

Figure source:  
NeurIPS tutorial  
Goodfellow 2016

# Improved training of GANs

## ■ Semi-Supervised Learning

- Predict labels in addition to fake/real in the discriminator
- Approximate way of modeling  $p(x,y)$
- Generator doesn't have to be made conditional  $p(x|y)$
- Use a deeper architecture for the discriminator compared to generator

$$\begin{aligned} L &= -\mathbb{E}_{\mathbf{x}, y \sim p_{\text{data}}(\mathbf{x}, y)} [\log p_{\text{model}}(y|\mathbf{x})] - \mathbb{E}_{\mathbf{x} \sim G} [\log p_{\text{model}}(y = K + 1|\mathbf{x})] \\ &= L_{\text{supervised}} + L_{\text{unsupervised}}, \text{ where} \end{aligned}$$

$$L_{\text{supervised}} = -\mathbb{E}_{\mathbf{x}, y \sim p_{\text{data}}(\mathbf{x}, y)} \log p_{\text{model}}(y|\mathbf{x}, y < K + 1)$$

$$L_{\text{unsupervised}} = -\{\mathbb{E}_{\mathbf{x} \sim p_{\text{data}}(\mathbf{x})} \log[1 - p_{\text{model}}(y = K + 1|\mathbf{x})] + \mathbb{E}_{\mathbf{x} \sim G} \log[p_{\text{model}}(y = K + 1|\mathbf{x})]\}$$

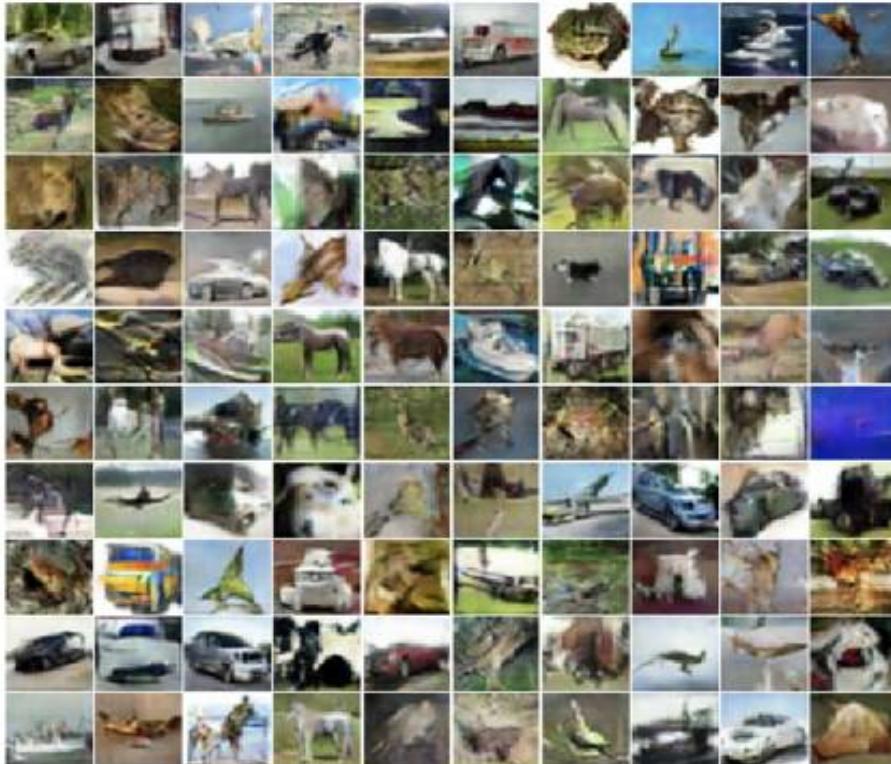
# Improved training of GANs

## ■ Inception Score

$$\exp(\mathbb{E}_{\mathbf{x}} \text{KL}(p(y|\mathbf{x}) || p(y)))$$

- Approximate way of modeling  $p(x,y)$
- Correlates with human judgement
- Captures some necessity for diversity

# Improved training of GANs



Salimans 2016

# fGAN - variational divergence

---

# Outline

---

- Motivation & Definition of Implicit Models
- Original GAN (Goodfellow et al, 2014)
- Evaluation: Parzen, Inception, Frechet
- Some Theory: Bayes-optimal Discriminator; Jensen-Shannon Divergence; Mode Collapse; Avoiding Saturation
- **GAN Progression:**
  - DC GAN (Radford et al, 2016)
  - Improved Training of GANs (Salimans et al, 2016)
  - **WGAN, WGAN-GP, Progressive GAN, SN-GAN, SAGAN**
  - BigGAN, BigGAN-Deep, StyleGAN, StyleGAN-v2, VIB-GAN
- Creative Conditional GANs
- GANs and Representations
- GANs as Energy Models
- GANs and Optimal Transport, Implicit Likelihood Models, Moment Matching
- Other uses of Adversarial Loss: Transfer Learning, Fairness
- GANs and Imitation Learning

# Wasserstein Distance

- We have seen  $KL(P_r \| P_g)$  and  $JSD(P_r, P_g) = KL(P_r \| (\frac{P_r + P_g}{2})) + KL(P_g \| (\frac{P_r + P_g}{2}))$
- Another distance measure inspired from Optimal Transport is the Earth Mover (EM) distance
- $W(P_r, P_g) = \inf_{\gamma \in \Pi(P_r, P_g)} \mathbb{E}_{(x,y) \sim \gamma} [| |x - y| |]$
- Goal: Design a GAN objective function such that the generator minimizes the Earth Mover / Wasserstein distance between data and generated distributions.

# Kantorovich Rubinstein Duality

- $W(P_r, P_g) = \inf_{\gamma \in \Pi(P_r, P_g)} \mathbb{E}_{(x,y) \sim \gamma} [| | x - y | |]$
- Intractable to estimate
- Kantorovich Rubinstein Duality:  $W(P_r, P_g) = \sup_{\|f_L\| \leq 1} \mathbb{E}_{x \sim P_r} [f(x)] - \mathbb{E}_{x \sim P_g} [f(x)]$
- Search over joint distributions is now a search over 1-Lipschitz functions

$f : X \rightarrow Y$  is K-Lipschitz if for distance functions  $d_X$  and  $d_Y$  on  $X$  and  $Y$   $d_Y(f(x_1), f(x_2)) \leq K d_X(x_1, x_2)$

# Wasserstein GAN

## Wasserstein GAN

Martin Arjovsky<sup>1</sup>, Soumith Chintala<sup>2</sup>, and Léon Bottou<sup>1,2</sup>

<sup>1</sup>Courant Institute of Mathematical Sciences

<sup>2</sup>Facebook AI Research

- Kantorovich Rubinstein Duality:  $W(P_r, P_g) = \sup_{\|f_L\| \leq 1} \mathbb{E}_{x \sim P_r} [f(x)] - \mathbb{E}_{x \sim P_g} [f(x)]$
- Supremum over linear (function space) expectations => search over K-Lipschitz gives you K times the Wasserstein distance.

# Wasserstein GAN - Pseudocode

---

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

---

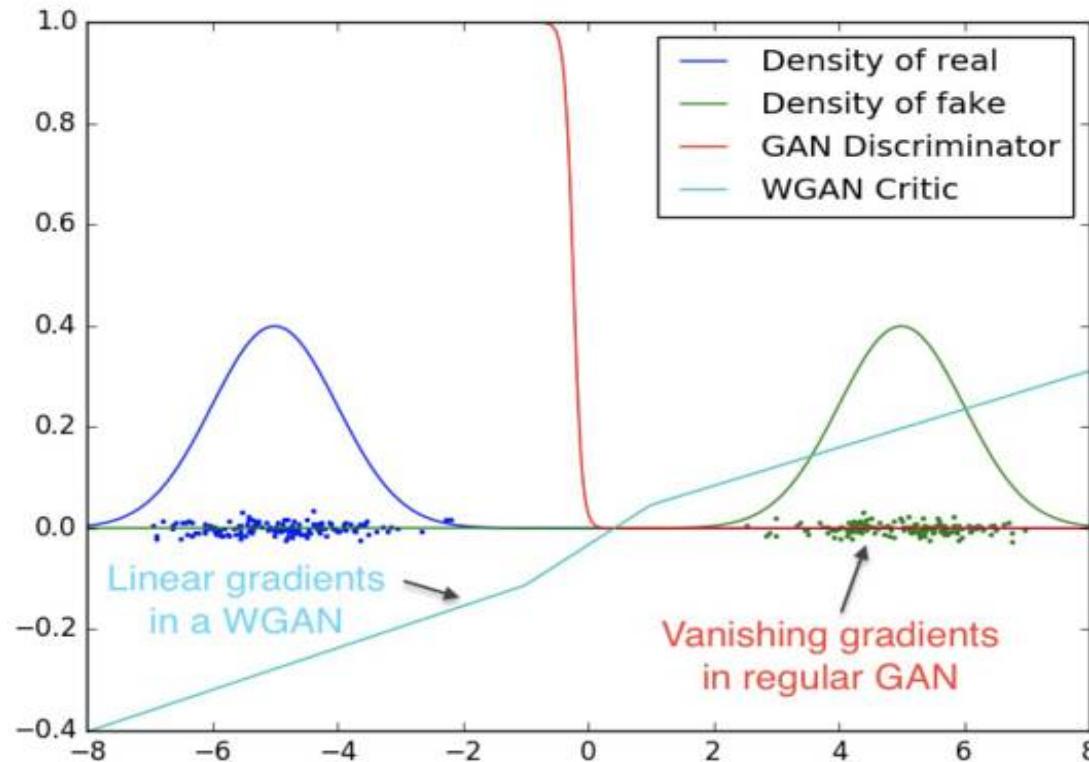
**Require:** :  $\alpha$ , the learning rate.  $c$ , the clipping parameter.  $m$ , the batch size.  $n_{\text{critic}}$ , the number of iterations of the critic per generator iteration.

**Require:** :  $w_0$ , initial critic parameters.  $\theta_0$ , initial generator's parameters.

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

(Arjovsky et al 2017)

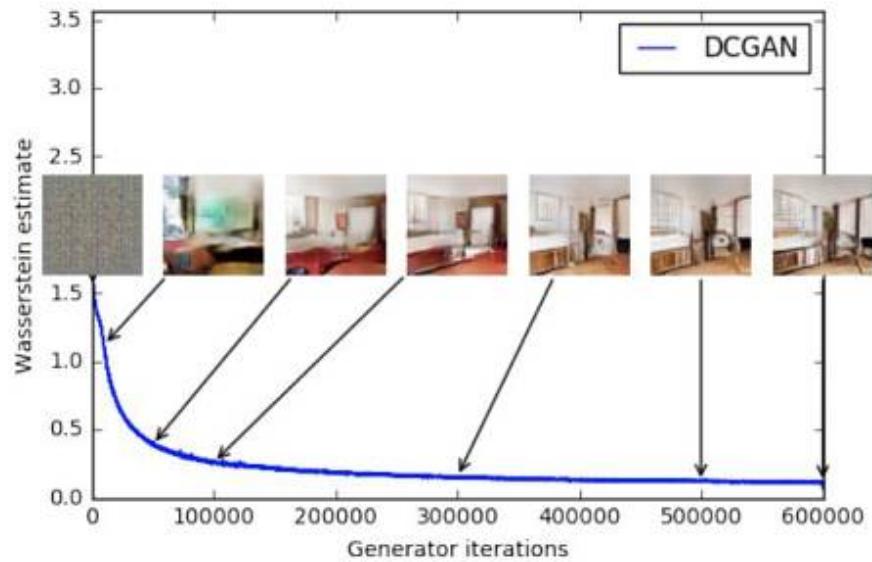
# Wasserstein GAN - Training critic to converge



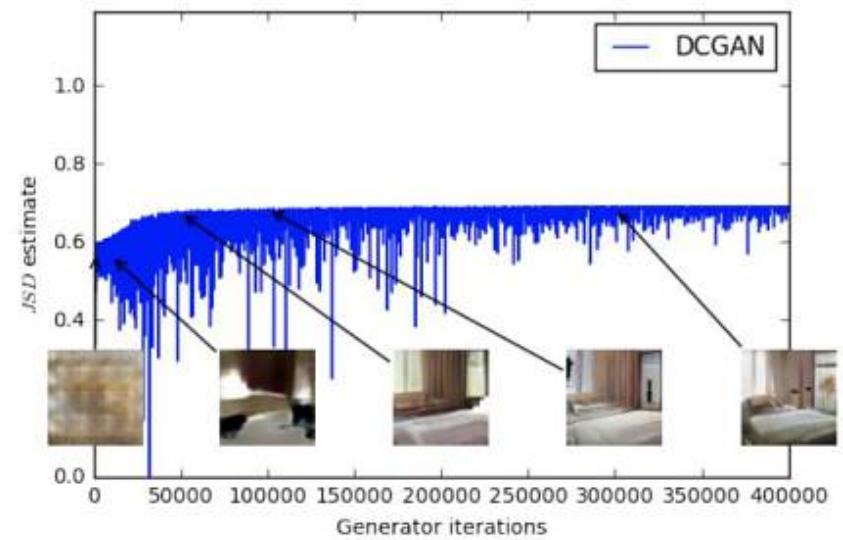
(Arjovsky et al 2017)

# Wasserstein distance correlates with sample quality

Wasserstein Estimate



JSD Estimate



(Arjovsky et al 2017)

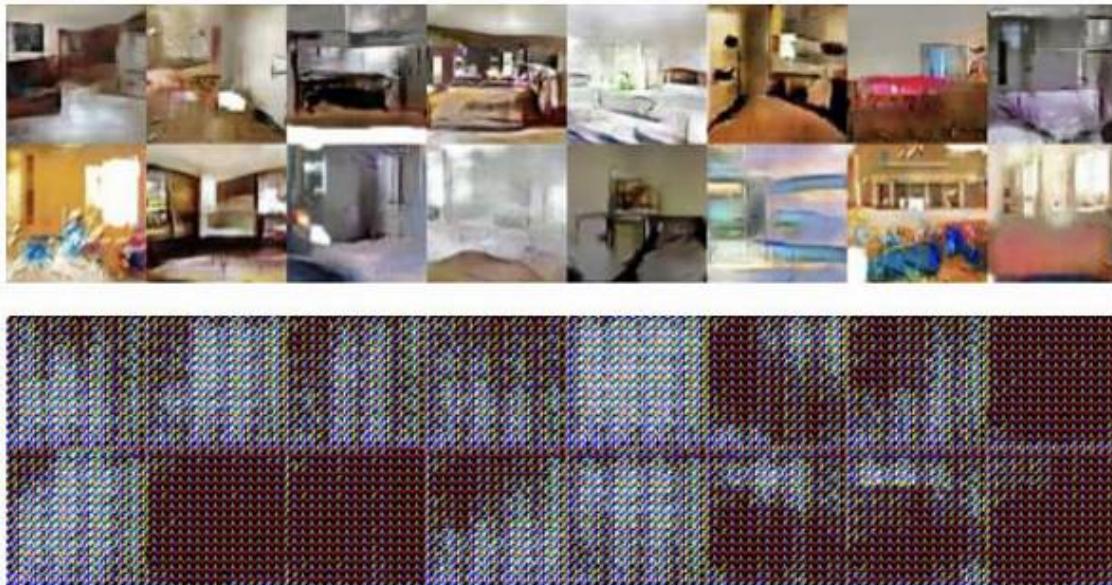
# WGAN Samples on par with DCGAN



(Arjovsky et al 2017)

*Top:* WGAN with the same DCGAN architecture. *Bottom:* DCGAN

# WGAN robust to architecture choices



*Top:* WGAN with DCGAN architecture, no batch norm. *Bottom:* DCGAN, no batch norm.

(Arjovsky et al 2017)

# WGAN robust to architecture choices



*Top:* WGAN with MLP architecture. *Bottom:* Standard GAN, same architecture. [Arjovsky et al 2017]

# WGAN Summary

Standard GAN

$$\min_G \max_D \mathbb{E}_{x \sim P_r} [\log D(x)] + \mathbb{E}_{\tilde{x} \sim P_g} [\log(1 - D(\tilde{x}))]$$



Wasserstein GAN

$$\min_G \max_{D \in \mathcal{D}} \mathbb{E}_{x \sim P_r} [D(x)] - \mathbb{E}_{\tilde{x} \sim P_g} [D(\tilde{x})]$$

(Arjovsky et al 2017)

# WGAN Summary

- New divergence measure for optimizing the generator
- Addresses instabilities with JSD version (sigmoid cross entropy)
- Robust to architectural choices
- Progress on mode collapse and stability of derivative wrt input
- Introduces the idea of using lipschitzness to stabilize GAN training
- **Negative:**

Weight clipping is a clearly terrible way to enforce a Lipschitz constraint. If the clipping parameter is large, then it can take a long time for any weights to reach their limit, thereby making it harder to train the critic till optimality. If the clipping is small, this can easily lead to vanishing gradients when the number of layers is big, or batch normalization is not used (such as in RNNs). We experimented with simple variants (such as projecting the weights to a sphere) with little difference, and we stuck with weight clipping due to its simplicity and already good performance. However, we do leave the topic of enforcing Lipschitz constraints in a neural network setting for further investigation, and we actively encourage interested researchers to improve on this method.

(Arjovsky et al 2017)

# Outline

---

- Motivation & Definition of Implicit Models
- Original GAN (Goodfellow et al, 2014)
- Evaluation: Parzen, Inception, Frechet
- Some Theory: Bayes-optimal Discriminator; Jensen-Shannon Divergence; Mode Collapse; Avoiding Saturation
- **GAN Progression:**
  - DC GAN (Radford et al, 2016)
  - Improved Training of GANs (Salimans et al, 2016)
  - WGAN, **WGAN-GP**, Progressive GAN, SN-GAN, SAGAN
  - BigGAN, BigGAN-Deep, StyleGAN, StyleGAN-v2, VIB-GAN
- Creative Conditional GANs
- GANs and Representations
- GANs as Energy Models
- GANs and Optimal Transport, Implicit Likelihood Models, Moment Matching
- Other uses of Adversarial Loss: Transfer Learning, Fairness
- GANs and Imitation Learning

# WGAN-GP: Gradient Penalty for Lipschitzness

---

## Improved Training of Wasserstein GANs

---

Ishaan Gulrajani<sup>1</sup>, Faruk Ahmed<sup>1</sup>, Martin Arjovsky<sup>2</sup>, Vincent Dumoulin<sup>1</sup>, Aaron Courville<sup>1,3</sup>

<sup>1</sup> Montreal Institute for Learning Algorithms

<sup>2</sup> Courant Institute of Mathematical Sciences

<sup>3</sup> CIFAR Fellow

igul222@gmail.com

{faruk.ahmed,vincent.dumoulin,aaron.courville}@umontreal.ca

ma4371@nyu.edu

### Abstract

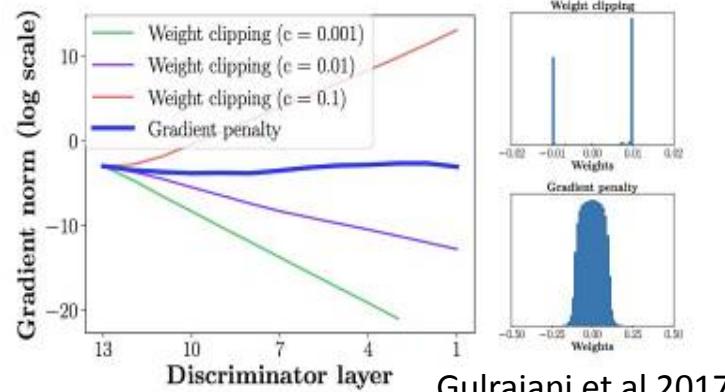
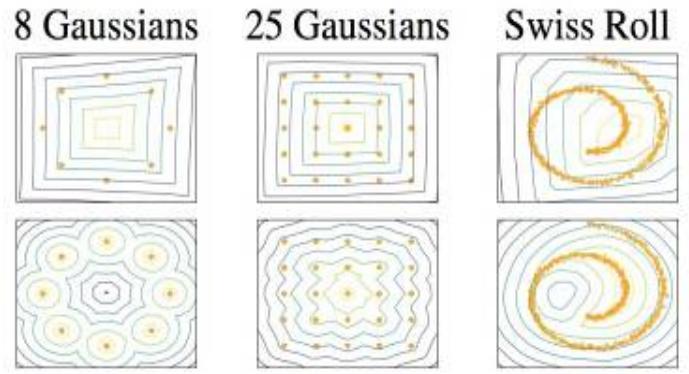
Generative Adversarial Networks (GANs) are powerful generative models, but suffer from training instability. The recently proposed Wasserstein GAN (WGAN) makes progress toward stable training of GANs, but sometimes can still generate only poor samples or fail to converge. We find that these problems are often due to the use of weight clipping in WGAN to enforce a Lipschitz constraint on the critic, which can lead to undesired behavior. We propose an alternative to clipping weights: penalize the norm of gradient of the critic with respect to its input. Our proposed method performs better than standard WGAN and enables stable training of a wide variety of GAN architectures with almost no hyperparameter tuning, including 101-layer ResNets and language models with continuous generators. We also achieve high quality generations on CIFAR-10 and LSUN bedrooms. <sup>†</sup>

Gulrajani et al 2017

# WGANGP: Gradient Penalty for Lipschitzness

$$\max_D \underbrace{\mathbb{E}_{x \sim P_r} [D(x)] - \mathbb{E}_{\tilde{x} \sim P_g} [D(\tilde{x})]}_{\text{Wasserstein critic objective}} + \lambda \underbrace{\mathbb{E}_{\hat{x} \sim P_{\hat{x}}} \left[ (\|\nabla_{\hat{x}} D(\hat{x})\|_2 - 1)^2 \right]}_{\text{Gradient Penalty for Lipschitzness}}$$

$$\hat{x} \leftarrow \epsilon x + (1 - \epsilon) \tilde{x}$$



Gulrajani et al 2017

# WGAN-GP: Pseudocode

---

**Algorithm 1** WGAN with gradient penalty. We use default values of  $\lambda = 10$ ,  $n_{\text{critic}} = 5$ ,  $\alpha = 0.0001$ ,  $\beta_1 = 0$ ,  $\beta_2 = 0.9$ .

---

**Require:** The gradient penalty coefficient  $\lambda$ , the number of critic iterations per generator iteration  $n_{\text{critic}}$ , the batch size  $m$ , Adam hyperparameters  $\alpha, \beta_1, \beta_2$ .

**Require:** initial critic parameters  $w_0$ , initial generator parameters  $\theta_0$ .

```
1: while  $\theta$  has not converged do
2:   for  $t = 1, \dots, n_{\text{critic}}$  do
3:     for  $i = 1, \dots, m$  do
4:       Sample real data  $\mathbf{x} \sim \mathbb{P}_r$ , latent variable  $\mathbf{z} \sim p(\mathbf{z})$ , a random number  $\epsilon \sim U[0, 1]$ .
5:        $\tilde{\mathbf{x}} \leftarrow G_\theta(\mathbf{z})$ 
6:        $\hat{\mathbf{x}} \leftarrow \epsilon \mathbf{x} + (1 - \epsilon) \tilde{\mathbf{x}}$ 
7:        $L^{(i)} \leftarrow D_w(\hat{\mathbf{x}}) - D_w(\mathbf{x}) + \lambda(\|\nabla_{\hat{\mathbf{x}}} D_w(\hat{\mathbf{x}})\|_2 - 1)^2$ 
8:     end for
9:      $w \leftarrow \text{Adam}(\nabla_w \frac{1}{m} \sum_{i=1}^m L^{(i)}, w, \alpha, \beta_1, \beta_2)$ 
10:   end for
11:   Sample a batch of latent variables  $\{\mathbf{z}^{(i)}\}_{i=1}^m \sim p(\mathbf{z})$ .
12:    $\theta \leftarrow \text{Adam}(\nabla_\theta \frac{1}{m} \sum_{i=1}^m -D_w(G_\theta(\mathbf{z})), \theta, \alpha, \beta_1, \beta_2)$ 
13: end while
```

Gulrajani et al 2017

# WGAN-GP: BatchNorm

**No critic batch normalization** Most prior GAN implementations [22, 23, 2] use batch normalization in both the generator and the discriminator to help stabilize training, but batch normalization changes the form of the discriminator's problem from mapping a single input to a single output to mapping from an entire batch of inputs to a batch of outputs [23]. Our penalized training objective is no longer valid in this setting, since we penalize the norm of the critic's gradient with respect to each input independently, and not the entire batch. To resolve this, we simply omit batch normalization in the critic in our models, finding that they perform well without it. Our method works with normalization schemes which don't introduce correlations between examples. In particular, we recommend layer normalization [3] as a drop-in replacement for batch normalization.

Gulrajani et al 2017

Since then no-one uses batchnorm in D anymore

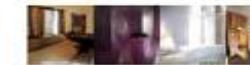
# WGAN-GP: Robustness to architectures

Nonlinearity ( $G$ )	[ReLU, LeakyReLU, $\frac{\text{softplus}(2x+2)}{2} - 1$ , tanh]
Nonlinearity ( $D$ )	[ReLU, LeakyReLU, $\frac{\text{softplus}(2x+2)}{2} - 1$ , tanh]
Depth ( $G$ )	[4, 8, 12, 20]
Depth ( $D$ )	[4, 8, 12, 20]
Batch norm ( $G$ )	[True, False]
Batch norm ( $D$ ; layer norm for WGAN-GP)	[True, False]
Base filter count ( $G$ )	[32, 64, 128]
Base filter count ( $D$ )	[32, 64, 128]

Min. score	Only GAN	Only WGAN-GP	Both succeeded	Both failed
1.0	0	8	192	0
3.0	1	88	110	1
5.0	0	147	42	11
7.0	1	104	5	90
9.0	0	0	0	200

Gulrajani et al 2017

# WGAN-GP: Robustness to architectures

DCGAN	LSGAN	WGAN (clipping)	WGAN-GP (ours)
Baseline ( $G$ : DCGAN, $D$ : DCGAN)			
			
$G$ : No BN and a constant number of filters, $D$ : DCGAN			
			
$G$ : 4-layer 512-dim ReLU MLP, $D$ : DCGAN			
			
No normalization in either $G$ or $D$			
			
Gated multiplicative nonlinearities everywhere in $G$ and $D$			
			
tanh nonlinearities everywhere in $G$ and $D$			
			
101-layer ResNet $G$ and $D$			
			

Gulrajani et al 2017

# WGAN-GP: High quality samples



Gulrajani et al 2017

# WGAN-GP: High quality samples

Table 3: Inception scores on CIFAR-10. Our unsupervised model achieves state-of-the-art performance, and our conditional model outperforms all others except SGAN.

Unsupervised		Supervised	
Method	Score	Method	Score
ALI [8] (in [27])	$5.34 \pm .05$	SteinGAN [26]	6.35
BEGAN [4]	5.62	DCGAN (with labels, in [26])	6.58
DCGAN [22] (in [11])	$6.16 \pm .07$	Improved GAN [23]	$8.09 \pm .07$
Improved GAN (-L+HA) [23]	$6.86 \pm .06$	AC-GAN [20]	$8.25 \pm .07$
EGAN-Ent-VI [7]	$7.07 \pm .10$	SGAN-no-joint [11]	$8.37 \pm .08$
DFM [27]	$7.72 \pm .13$	WGAN-GP ResNet (ours)	$8.42 \pm .10$
<b>WGAN-GP ResNet (ours)</b>	$7.86 \pm .07$	<b>SGAN [11]</b>	$8.59 \pm .12$

Gulrajani et al 2017

# WGAN-GP: Summary

---

- Robust to architectural choices
- Became a very popular GAN model - 10,000+ citations, has been used in NVIDIA's Progressive GANs, StyleGAN, etc - biggest GAN successes
- Residual architecture widely adopted
- Possible negative: slow wall clock time due to gradient penalty
- Gradient penalty applied on a heuristic distribution of samples from current generator. Could be unstable when learning rates are high

# Outline

---

- Motivation & Definition of Implicit Models
- Original GAN (Goodfellow et al, 2014)
- Evaluation: Parzen, Inception, Frechet
- Some Theory: Bayes-optimal Discriminator; Jensen-Shannon Divergence; Mode Collapse; Avoiding Saturation
- **GAN Progression:**
  - DC GAN (Radford et al, 2016)
  - Improved Training of GANs (Salimans et al, 2016)
  - WGAN, WGAN-GP, **Progressive GAN**, SN-GAN, SAGAN
  - BigGAN, BigGAN-Deep, StyleGAN, StyleGAN-v2, VIB-GAN
- Creative Conditional GANs
- GANs and Representations
- GANs as Energy Models
- GANs and Optimal Transport, Implicit Likelihood Models, Moment Matching
- Other uses of Adversarial Loss: Transfer Learning, Fairness
- GANs and Imitation Learning

# Progressive growing of GANs

## PROGRESSIVE GROWING OF GANs FOR IMPROVED QUALITY, STABILITY, AND VARIATION

**Tero Karras**

NVIDIA

{tkarras, taila, slaine, jlehtinen}@nvidia.com

**Timo Aila**

NVIDIA

**Samuli Laine**

NVIDIA

**Jaakko Lehtinen**

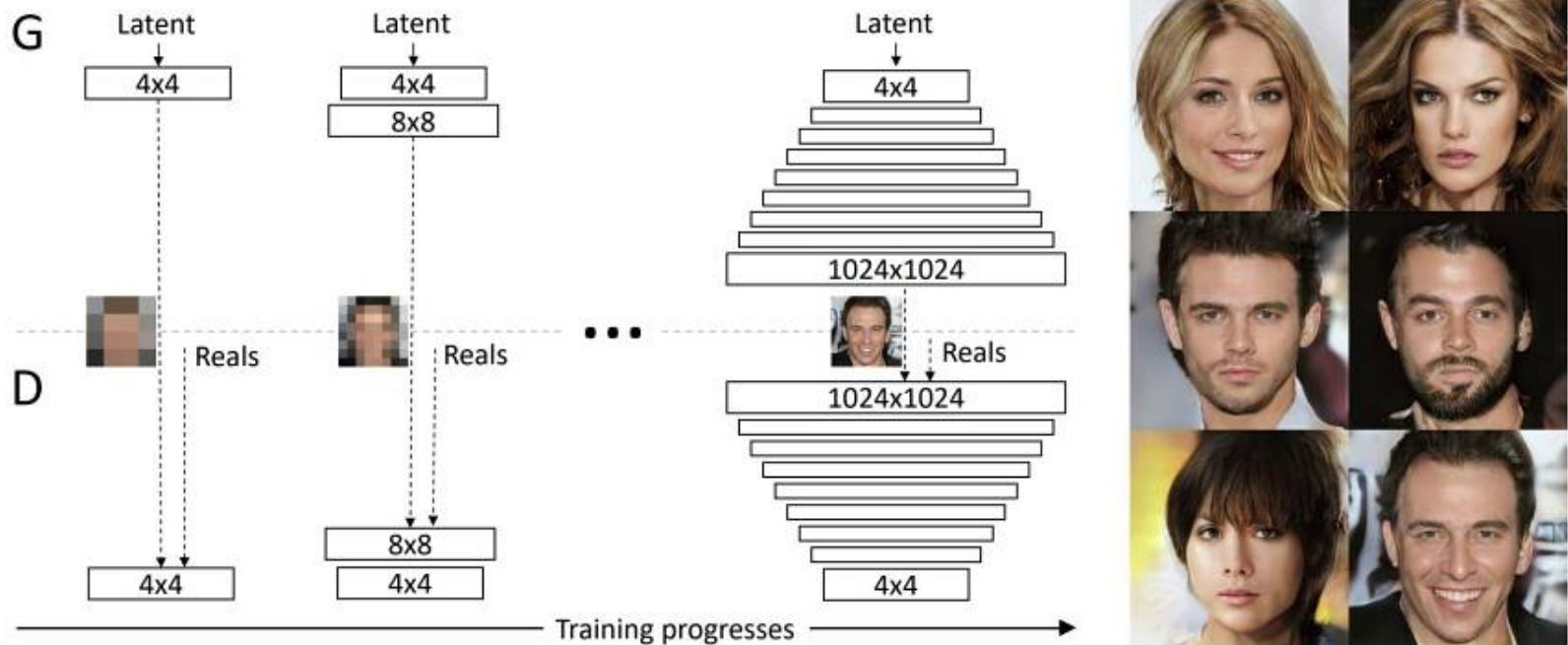
NVIDIA and Aalto University

### ABSTRACT

We describe a new training methodology for generative adversarial networks. The key idea is to grow both the generator and discriminator progressively: starting from a low resolution, we add new layers that model increasingly fine details as training progresses. This both speeds the training up and greatly stabilizes it, allowing us to produce images of unprecedented quality, e.g., CELEBA images at  $1024^2$ . We also propose a simple way to increase the variation in generated images, and achieve a record inception score of 8.80 in unsupervised CIFAR10. Additionally, we describe several implementation details that are important for discouraging unhealthy competition between the generator and discriminator. Finally, we suggest a new metric for evaluating GAN results, both in terms of image quality and variation. As an additional contribution, we construct a higher-quality version of the CELEBA dataset.

Karras et al 2017

# Progressive growing of GANs



First super-high quality samples from any generative models

Karras et al 2017

# Progressive growing of GANs



Karras et al 2017

# Progressive growing of GANs



Mao et al. (2016b) ( $128 \times 128$ )



Gulrajani et al. (2017) ( $128 \times 128$ )



Our ( $256 \times 256$ )

Karras et al 2017

# Progressive growing of GANs



K  
a  
r  
r  
a  
s  
e  
t  
a  
l  
2  
0  
1  
7

# Progressive growing of GANs



Karras et al 2017

# Outline

---

- Motivation & Definition of Implicit Models
- Original GAN (Goodfellow et al, 2014)
- Evaluation: Parzen, Inception, Frechet
- Some Theory: Bayes-optimal Discriminator; Jensen-Shannon Divergence; Mode Collapse; Avoiding Saturation
- **GAN Progression:**
  - DC GAN (Radford et al, 2016)
  - Improved Training of GANs (Salimans et al, 2016)
  - WGAN, WGAN-GP, Progressive GAN, **SN-GAN**, SAGAN
  - BigGAN, BigGAN-Deep, StyleGAN, StyleGAN-v2, VIB-GAN
- Creative Conditional GANs
- GANs and Representations
- GANs as Energy Models
- GANs and Optimal Transport, Implicit Likelihood Models, Moment Matching
- Other uses of Adversarial Loss: Transfer Learning, Fairness
- GANs and Imitation Learning

# Spectral Normalization GAN (SNGAN)

## SPECTRAL NORMALIZATION FOR GENERATIVE ADVERSARIAL NETWORKS

Takeru Miyato<sup>1</sup>, Toshiki Kataoka<sup>1</sup>, Masanori Koyama<sup>2</sup>, Yuichi Yoshida<sup>3</sup>

{miyato, kataoka}@preferred.jp

koyama.masanori@gmail.com

yyoshida@nii.ac.jp

<sup>1</sup>Preferred Networks, Inc. <sup>2</sup>Ritsumeikan University <sup>3</sup>National Institute of Informatics

### ABSTRACT

One of the challenges in the study of generative adversarial networks is the instability of its training. In this paper, we propose a novel weight normalization technique called spectral normalization to stabilize the training of the discriminator. Our new normalization technique is computationally light and easy to incorporate into existing implementations. We tested the efficacy of spectral normalization on CIFAR10, STL-10, and ILSVRC2012 dataset, and we experimentally confirmed that spectrally normalized GANs (SN-GANs) is capable of generating images of better or equal quality relative to the previous training stabilization techniques. The code with Chainer (Tokui et al., 2015), generated images and pretrained models are available at [https://github.com/pfnet-research/sngan\\_projection](https://github.com/pfnet-research/sngan_projection).

Miyato et al 2017

# Spectral Normalization GAN (SNGAN)

$$f(\mathbf{x}, \theta) = W^{L+1} a_L(W^L(a_{L-1}(W^{L-1}(\dots a_1(W^1 \mathbf{x}) \dots))))$$

$$D(\mathbf{x}, \theta) = \mathcal{A}(f(\mathbf{x}, \theta))$$

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

$$\arg \max_{\|f\|_{\text{Lip}} \leq K} V(G, D)$$

Miyato et al 2017

# Spectral Normalization GAN (SNGAN)

- Key idea: Connecting Lipschitzness of discriminator to spectral norm of each layer.

$$g : \mathbf{h}_{in} \mapsto \mathbf{h}_{out}$$

$$\|g\|_{\text{Lip}} = \sup_{\mathbf{h}} \sigma(\nabla g(\mathbf{h}))$$

$$\sigma(A) := \max_{\mathbf{h}: \mathbf{h} \neq \mathbf{0}} \frac{\|A\mathbf{h}\|_2}{\|\mathbf{h}\|_2} = \max_{\|\mathbf{h}\|_2 \leq 1} \|A\mathbf{h}\|_2$$

Miyato et al 2017

# Spectral Normalization GAN (SNGAN)

$$\|g\|_{\text{Lip}} = \sup_{\mathbf{h}} \sigma(\nabla g(\mathbf{h})) = \sup_{\mathbf{h}} \sigma(W) = \sigma(W)$$

$$\|g_1 \circ g_2\|_{\text{Lip}} \leq \|g_1\|_{\text{Lip}} \cdot \|g_2\|_{\text{Lip}}$$

$$\begin{aligned} \|f\|_{\text{Lip}} &\leq \|(\mathbf{h}_L \mapsto W^{L+1}\mathbf{h}_L)\|_{\text{Lip}} \cdot \|a_L\|_{\text{Lip}} \cdot \|(\mathbf{h}_{L-1} \mapsto W^L\mathbf{h}_{L-1})\|_{\text{Lip}} \\ &\cdots \|a_1\|_{\text{Lip}} \cdot \|(\mathbf{h}_0 \mapsto W^1\mathbf{h}_0)\|_{\text{Lip}} = \prod_{l=1}^{L+1} \|(\mathbf{h}_{l-1} \mapsto W^l\mathbf{h}_{l-1})\|_{\text{Lip}} = \prod_{l=1}^{L+1} \sigma(W^l) \end{aligned}$$

$$\bar{W}_{\text{SN}}(W) := W/\sigma(W)$$

In practice: sigma = max of norm over all data passed through the layer

Miyato et al 2017

# Spectral Normalization GAN (SNGAN)

RGB image $x \in \mathbb{R}^{128 \times 128 \times 3}$	
ResBlock down 64	
ResBlock down 128	
ResBlock down 256	
Concat(Embed( $y$ ), $h$ )	
ResBlock down 512	
ResBlock down 1024	
ResBlock 1024	
ReLU	
Global sum pooling	
dense $\rightarrow 1$	
(c) Discriminator for conditional GANs. For computational ease, we embedded the integer label $y \in \{0, \dots, 1000\}$ into 128 dimension before concatenating the vector to the output of the intermediate layer.	(c) Discriminator for conditional GANs. For computational ease, we embedded the integer label $y \in \{0, \dots, 1000\}$ into 128 dimension before concatenating the vector to the output of the intermediate layer.
Generator	
$z \in \mathbb{R}^{128} \sim \mathcal{N}(0, I)$	RGB image $x \in \mathbb{R}^{128 \times 128 \times 3}$
dense, $4 \times 4 \times 1024$	ResBlock down 64
ResBlock up 1024	ResBlock down 128
ResBlock up 512	ResBlock down 256
ResBlock up 256	ResBlock down 512
ResBlock up 128	ResBlock down 1024
ResBlock up 64	ResBlock 1024
BN, ReLU, $3 \times 3$ conv 3	ReLU
Tanh	Global sum pooling
(a) Generator	dense $\rightarrow 1$
Discriminator for unconditional GANs.	
(b) Discriminator for unconditional GANs.	(b) Discriminator for unconditional GANs.

Miyato et al 2017

# Spectral Normalization GAN (SNGAN)

Welsh springer spaniel



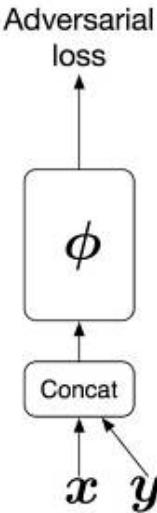
Miyato et al 2017

Pizza

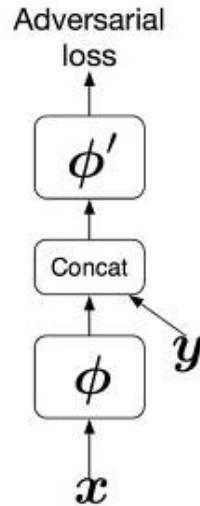


# Projection Discriminator

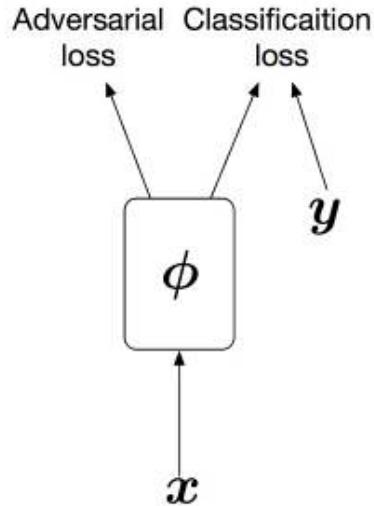
**(a) cGANs,  
input concat**  
(Mirza & Osindero, 2014)



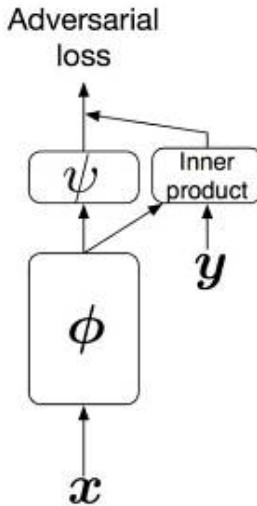
**(b) cGANs,  
hidden concat**  
(Reed et al., 2016)



**(c) AC-GANs**  
(Odena et al., 2017)



**(d) (ours) Projection**



# SNGAN: Summary

---

- High quality class conditional samples at Imagenet scale
- First GAN to work on full Imagenet (million image dataset)
- Computational benefits over WGAN-GP (single power iteration and no need of a backward pass)

# Outline

---

- Motivation & Definition of Implicit Models
- Original GAN (Goodfellow et al, 2014)
- Evaluation: Parzen, Inception, Frechet
- Some Theory: Bayes-optimal Discriminator; Jensen-Shannon Divergence; Mode Collapse; Avoiding Saturation
- **GAN Progression:**
  - DC GAN (Radford et al, 2016)
  - Improved Training of GANs (Salimans et al, 2016)
  - WGAN, WGAN-GP, Progressive GAN, SN-GAN, **SAGAN**
  - BigGAN, BigGAN-Deep, StyleGAN, StyleGAN-v2, VIB-GAN
- Creative Conditional GANs
- GANs and Representations
- GANs as Energy Models
- GANs and Optimal Transport, Implicit Likelihood Models, Moment Matching
- Other uses of Adversarial Loss: Transfer Learning, Fairness
- GANs and Imitation Learning

# Self Attention GAN (SAGAN)

---

## Self-Attention Generative Adversarial Networks

---

**Han Zhang\***

Rutgers University

**Ian Goodfellow**

Google Brain

**Dimitris Metaxas**

Rutgers University

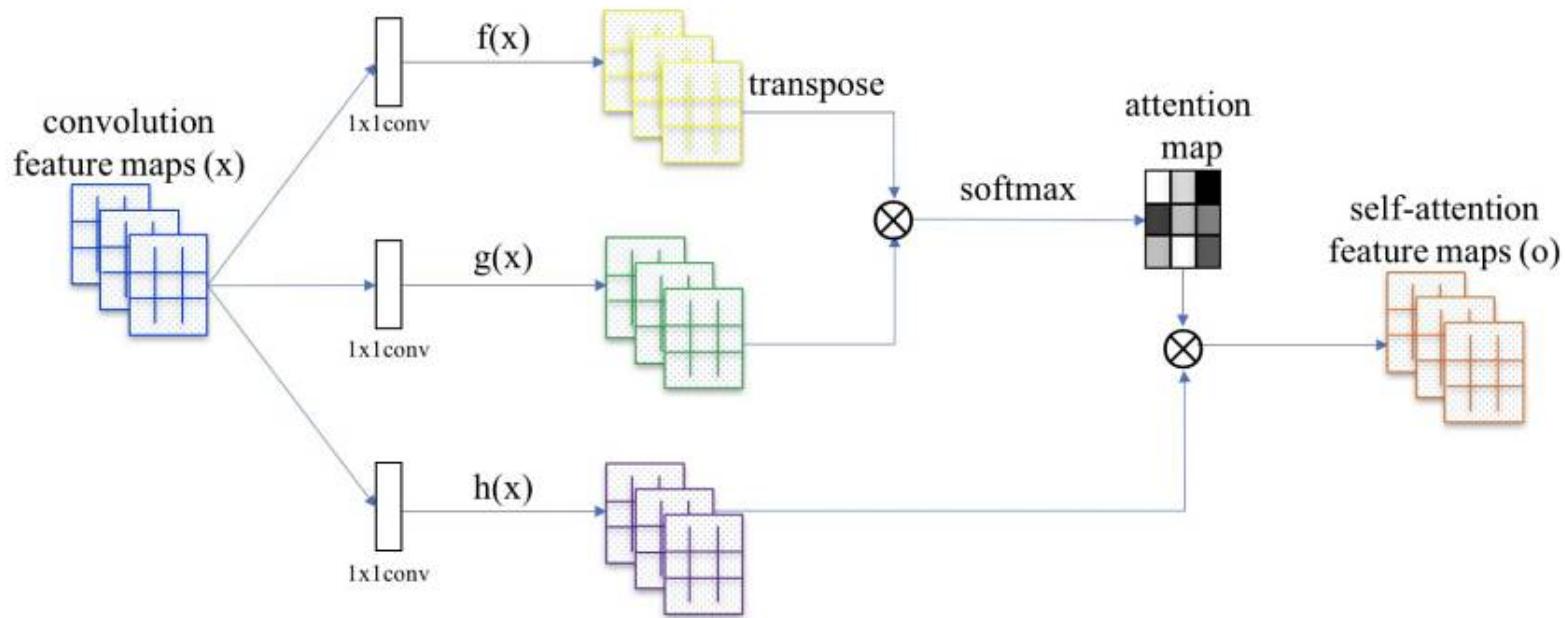
**Augustus Odena**

Google Brain

### Abstract

In this paper, we propose the Self-Attention Generative Adversarial Network (SAGAN) which allows attention-driven, long-range dependency modeling for image generation tasks. Traditional convolutional GANs generate high-resolution details as a function of only spatially local points in lower-resolution feature maps. In SAGAN, details can be generated using cues from all feature locations. Moreover, the discriminator can check that highly detailed features in distant portions of the image are consistent with each other. Furthermore, recent work has shown that generator conditioning affects GAN performance. Leveraging this insight, we apply spectral normalization to the GAN generator and find that this improves training dynamics. The proposed SAGAN achieves the state-of-the-art results, boosting the best published Inception score from 36.8 to 52.52 and reducing Fréchet Inception distance from 27.62 to 18.65 on the challenging ImageNet dataset. Visualization of the attention layers shows that the generator leverages neighborhoods that correspond to object shapes rather than local regions of fixed shape.

# Self Attention GAN (SAGAN)



Zhang et al 2018

# Self Attention GAN (SAGAN)

$$f(x) = W_f x, \ g(x) = W_g x$$

$$\beta_{j,i} = \frac{\exp(s_{ij})}{\sum_{i=1}^N \exp(s_{ij})}$$

$$s_{ij} = f(\mathbf{x}_i)^T g(\mathbf{x}_j)$$

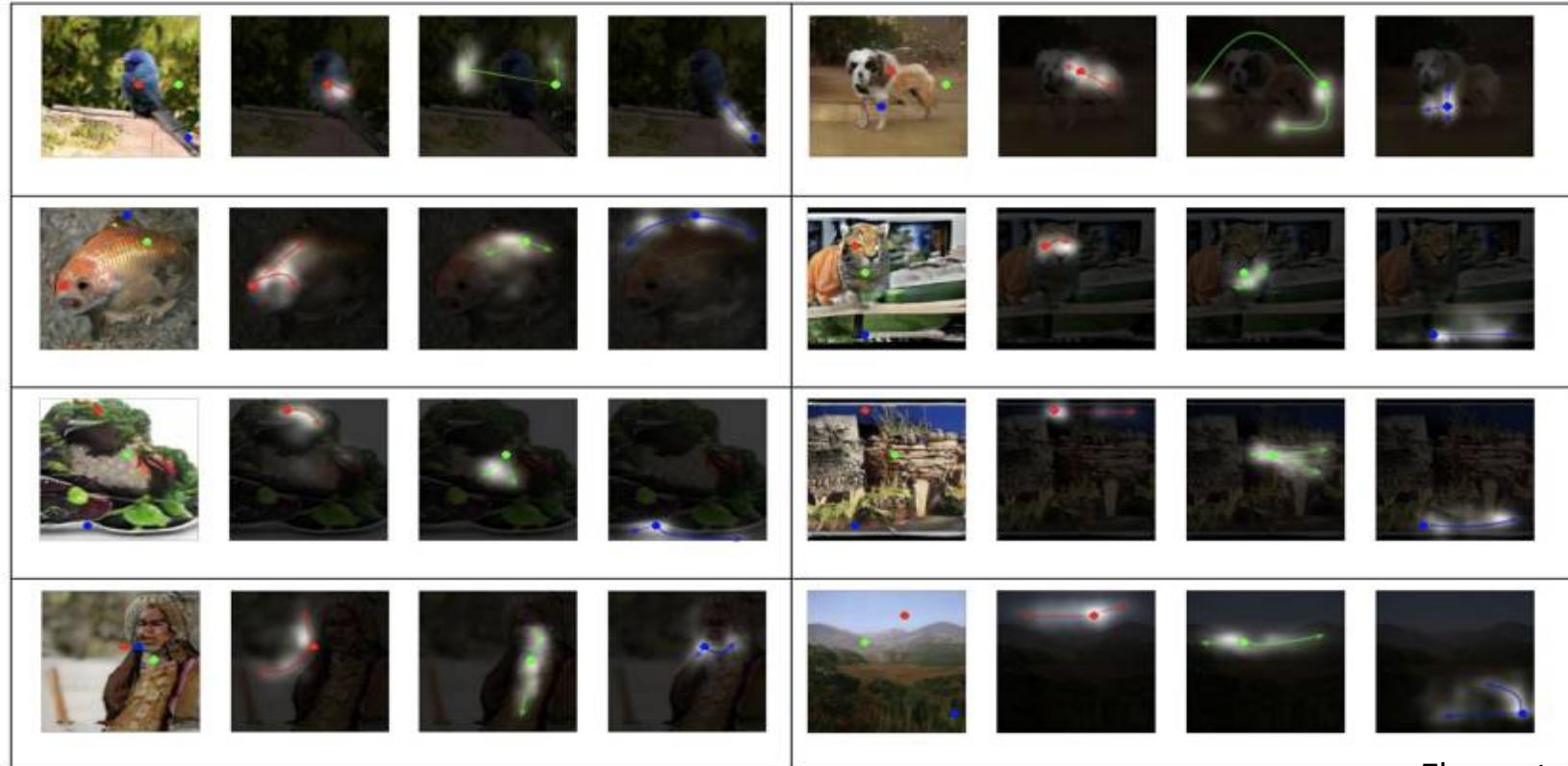
$$\mathbf{y}_i = \gamma \mathbf{o}_i + \mathbf{x}_i$$

Zhang et al 2018

# Self Attention GAN (SAGAN)

- Salient bits:
  - Applies spectral normalization to both the generator and discriminator weight matrices
    - This is counter-intuitive to popular belief that you only have to mathematically condition the discriminator
  - Uses self-attention in both the generator and discriminator
  - Hinge Loss
  - *First GAN to produce “good” unconditional full Imagenet samples*
  - Conditional models
    - Conditional BN for G, Projection Discriminator for D

# Self Attention GAN (SAGAN)



Zhang et al 2018

# Self Attention GAN (SAGAN)



Zhang et al 2018

# Self Attention GAN (SAGAN)

Model	Inception Score	FID
AC-GAN [31]	28.5	/
SNGAN-projection [17]	36.8	27.62*
SAGAN	<b>52.52</b>	<b>18.65</b>

Table 2: Comparison of the proposed SAGAN with state-of-the-art GAN models [19, 17] for class conditional image generation on ImageNet. FID of SNGAN-projection is calculated from officially released weights.

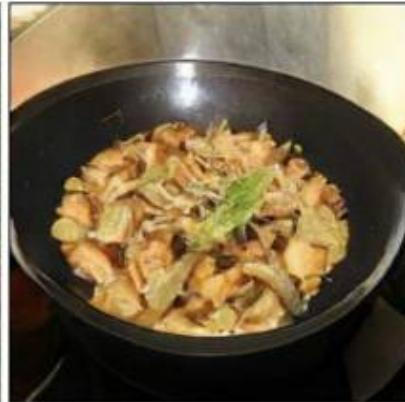
Zhang et al 2018

# Outline

---

- Motivation & Definition of Implicit Models
- Original GAN (Goodfellow et al, 2014)
- Evaluation: Parzen, Inception, Frechet
- Some Theory: Bayes-optimal Discriminator; Jensen-Shannon Divergence; Mode Collapse; Avoiding Saturation
- **GAN Progression:**
  - DC GAN (Radford et al, 2016)
  - Improved Training of GANs (Salimans et al, 2016)
  - WGAN, WGAN-GP, Progressive GAN, SN-GAN, SAGAN
  - ***BigGAN, BigGAN-Deep, StyleGAN, StyleGAN-v2, VIB-GAN***
- Creative Conditional GANs
- GANs and Representations
- GANs as Energy Models
- GANs and Optimal Transport, Implicit Likelihood Models, Moment Matching
- Other uses of Adversarial Loss: Transfer Learning, Fairness
- GANs and Imitation Learning

# BigGAN



# BigGAN



# BigGAN

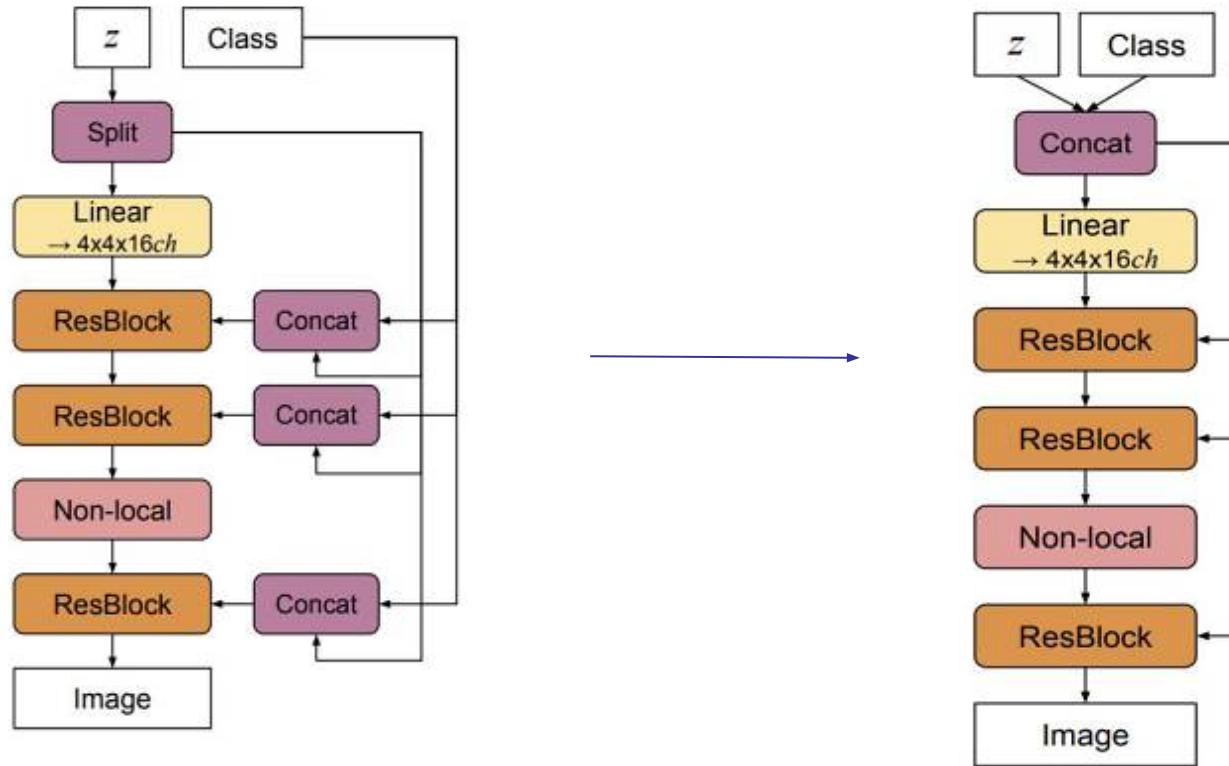
---

$$R_\beta(W) = \beta \|W^\top W - I\|_F^2$$

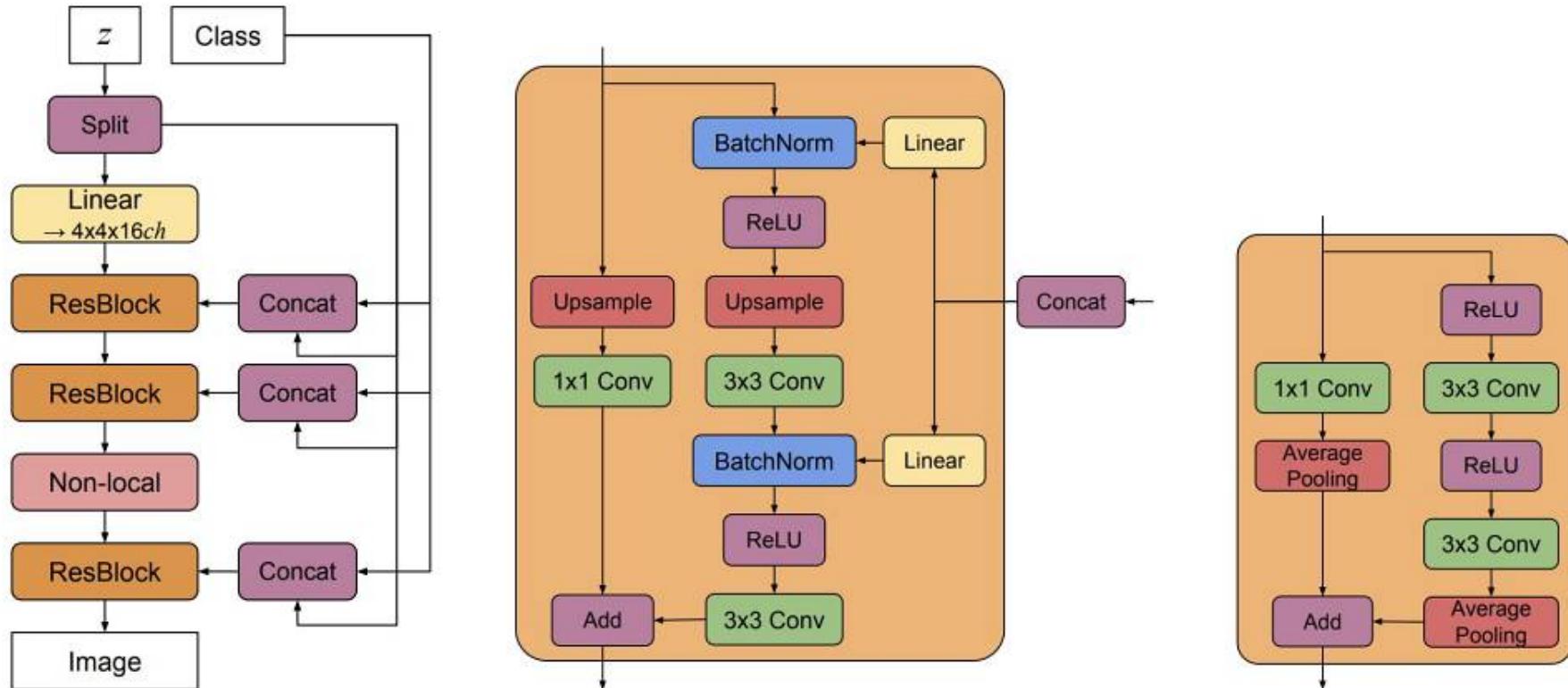


$$R_\beta(W) = \beta \|W^\top W \odot (1 - I)\|_F^2,$$

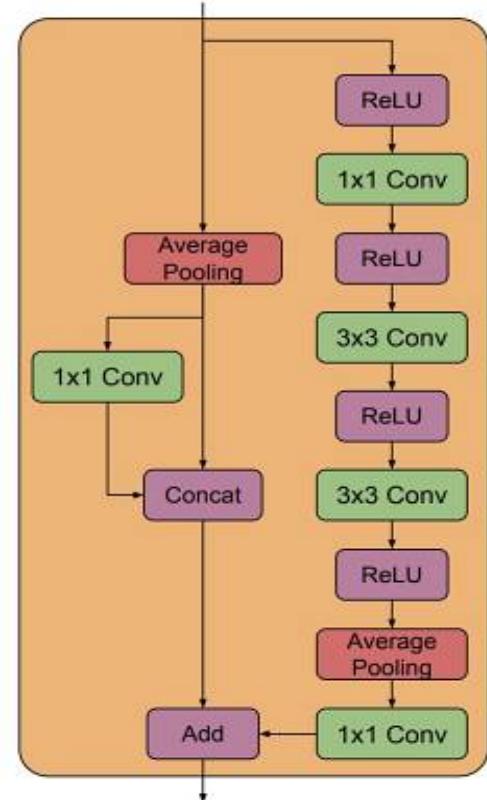
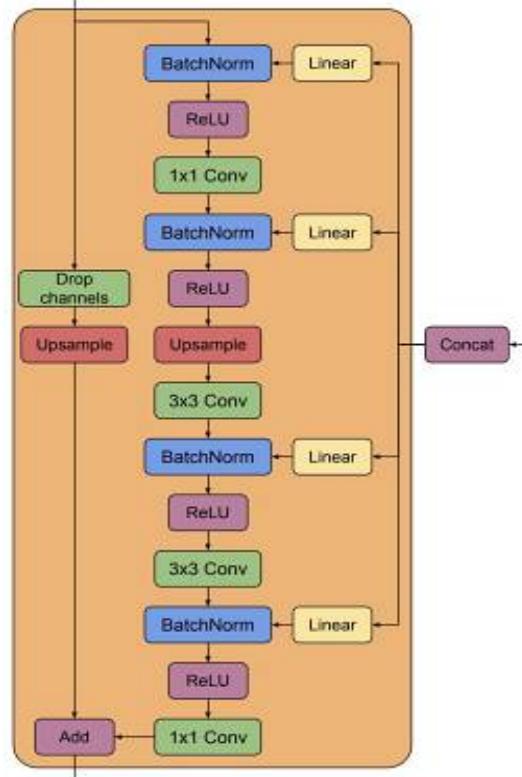
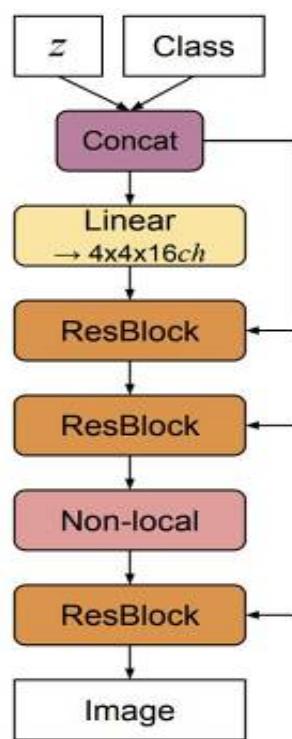
# BigGAN and BigGAN-deep



# BigGAN



# BigGAN-deep



# BigGAN

Table 6: BigGAN architecture for  $512 \times 512$  images. Relative to the  $256 \times 256$  architecture, we add an additional ResBlock at the  $512 \times 512$  resolution. Memory constraints force us to move the non-local block in both networks back to  $64 \times 64$  resolution as in the  $128 \times 128$  pixel setting.

$z \in \mathbb{R}^{160} \sim \mathcal{N}(0, I)$	$\text{RGB image } x \in \mathbb{R}^{512 \times 512 \times 3}$
Embed( $y$ ) $\in \mathbb{R}^{128}$	
Linear $(20 + 128) \rightarrow 4 \times 4 \times 16ch$	ResBlock down $ch \rightarrow ch$
ResBlock up $16ch \rightarrow 16ch$	ResBlock down $ch \rightarrow 2ch$
ResBlock up $16ch \rightarrow 8ch$	ResBlock down $2ch \rightarrow 4ch$
ResBlock up $8ch \rightarrow 8ch$	Non-Local Block $(64 \times 64)$
ResBlock up $8ch \rightarrow 4ch$	ResBlock down $4ch \rightarrow 8ch$
Non-Local Block $(64 \times 64)$	ResBlock down $8ch \rightarrow 8ch$
ResBlock up $4ch \rightarrow 2ch$	ResBlock down $8ch \rightarrow 16ch$
ResBlock up $2ch \rightarrow ch$	ResBlock down $16ch \rightarrow 16ch$
ResBlock up $ch \rightarrow ch$	ResBlock $16ch \rightarrow 16ch$
BN, ReLU, $3 \times 3$ Conv $ch \rightarrow 3$	ReLU, Global sum pooling
Tanh	Embed( $y$ ) $\cdot h + (\text{linear} \rightarrow 1)$
(a) Generator	(b) Discriminator

# BigGAN

---

- Salient bits

- Increase your batch size (as much as you can)
- Use Cross-Replica (Sync) Batch Norm
- Increase your model size
- Wider helps as much as deeper
- Fuse class information at all levels
- Hinge Loss
- Orthonormal regularization & Truncation Trick (on z)

# BigGAN

Batch	Ch.	Param (M)	Shared	Skip- $z$	Ortho.	Itr $\times 10^3$	FID	IS
256	64	81.5	SA-GAN Baseline			1000	18.65	52.52
512	64	81.5	✗	✗	✗	1000	15.30	58.77( $\pm 1.18$ )
1024	64	81.5	✗	✗	✗	1000	14.88	63.03( $\pm 1.42$ )
2048	64	81.5	✗	✗	✗	732	12.39	76.85( $\pm 3.83$ )
2048	96	173.5	✗	✗	✗	295( $\pm 18$ )	9.54( $\pm 0.62$ )	92.98( $\pm 4.27$ )
2048	96	160.6	✓	✗	✗	185( $\pm 11$ )	9.18( $\pm 0.13$ )	94.94( $\pm 1.32$ )
2048	96	158.3	✓	✓	✗	152( $\pm 7$ )	8.73( $\pm 0.45$ )	98.76( $\pm 2.84$ )
2048	96	158.3	✓	✓	✓	165( $\pm 13$ )	8.51( $\pm 0.32$ )	99.31( $\pm 2.10$ )
2048	64	71.3	✓	✓	✓	371( $\pm 7$ )	10.48( $\pm 0.10$ )	86.90( $\pm 0.61$ )

# BigGAN

Model	Res.	FID/IS	(min FID) / IS	FID / (valid IS)	FID / (max IS)
SN-GAN	128	27.62/36.80	N/A	N/A	N/A
SA-GAN	128	18.65/52.52	N/A	N/A	N/A
BigGAN	128	$8.7 \pm .6$ / $98.8 \pm 3$	$7.7 \pm .2$ / $126.5 \pm 0$	$9.6 \pm .4$ / $166.3 \pm 1$	$25 \pm 2$ / $206 \pm 2$
BigGAN	256	$8.7 \pm .1$ / $142.3 \pm 2$	$7.7 \pm .1$ / $178.0 \pm 5$	$9.3 \pm .3$ / $233.1 \pm 1$	$25 \pm 5$ / $291 \pm 4$
BigGAN	512	8.1/144.2	7.6/170.3	11.8/241.4	27.0/275
BigGAN-deep	128	$5.7 \pm .3$ / $124.5 \pm 2$	$6.3 \pm .3$ / $148.1 \pm 4$	$7.4 \pm .6$ / $166.5 \pm 1$	$25 \pm 2$ / $253 \pm 11$
BigGAN-deep	256	$6.9 \pm .2$ / $171.4 \pm 2$	$7.0 \pm .1$ / $202.6 \pm 2$	$8.1 \pm .1$ / $232.5 \pm 2$	$27 \pm 8$ / $317 \pm 6$
BigGAN-deep	512	7.5/152.8	7.7/181.4	11.5/241.5	39.7/298

# BigGAN - Truncation Trick



(a)

(b)

# BigGAN - Sampling

The default behavior with batch normalized classifier networks is to use a running average of the activation moments at test time. Previous works (Radford et al., 2016) have instead used batch statistics when sampling images. While this is not technically an invalid way to sample, it means that results are dependent on the test batch size (and how many devices it is split across), and further complicates reproducibility.

We find that this detail is extremely important, with changes in test batch size producing drastic changes in performance. This is further exacerbated when one uses exponential moving averages of  $\mathbf{G}$ 's weights for sampling, as the BatchNorm running averages are computed with non-averaged weights and are poor estimates of the activation statistics for the averaged weights.

To counteract both these issues, we employ “standing statistics,” where we compute activation statistics at sampling time by running the  $\mathbf{G}$  through multiple forward passes (typically 100) each with different batches of random noise, and storing means and variances aggregated across all forward passes. Analogous to using running statistics, this results in  $\mathbf{G}$ 's outputs becoming invariant to batch size and the number of devices, even when producing a single sample.

# BigGAN



(a)  $128 \times 128$



(b)  $256 \times 256$

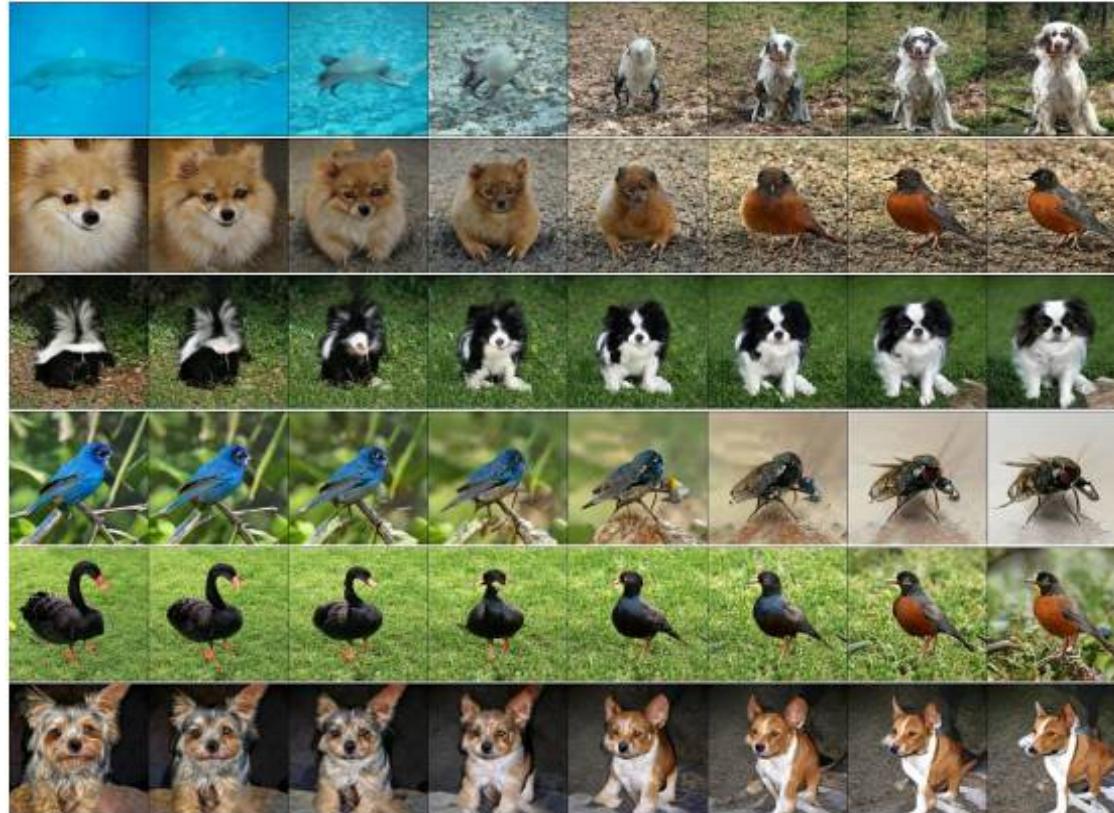


(c)  $512 \times 512$

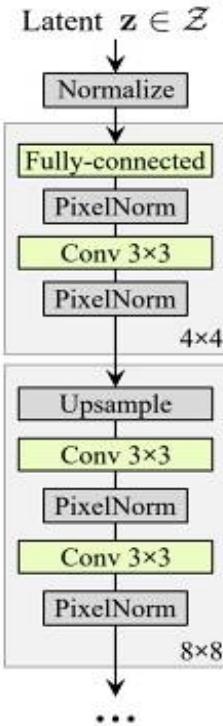


(d)

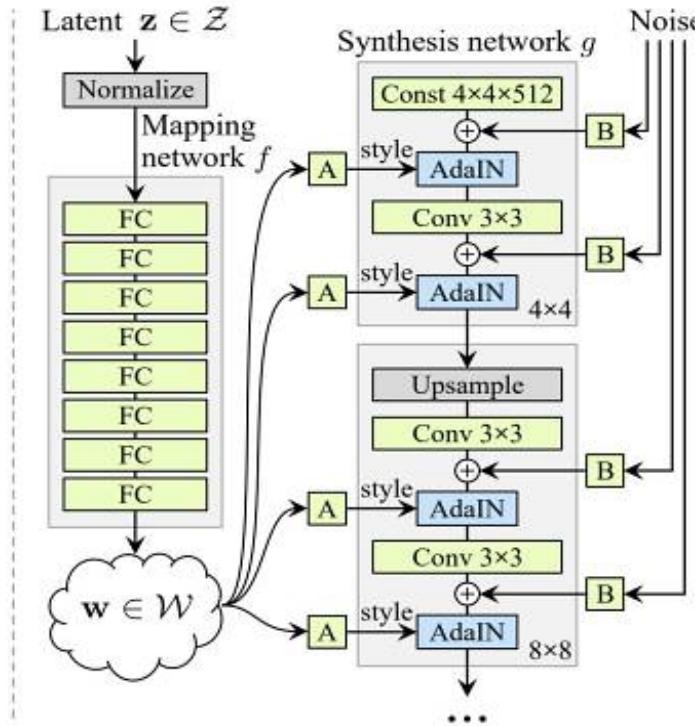
# BigGAN



# StyleGAN



(a) Traditional



(b) Style-based generator

# StyleGAN - Adaptive Instance Norm

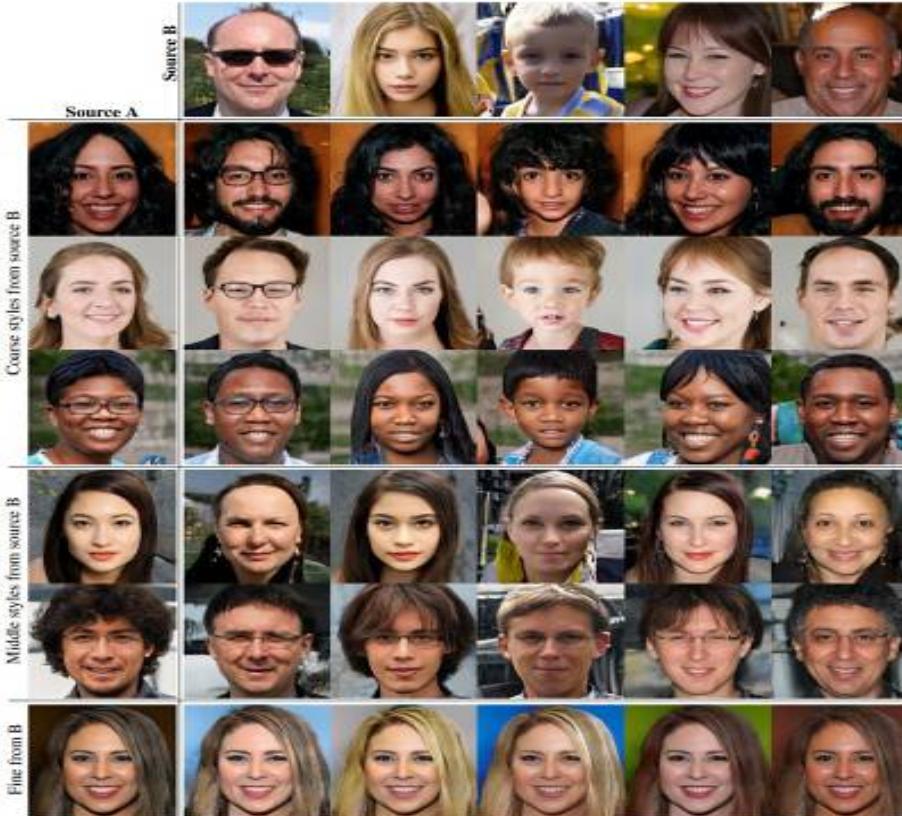
$$\text{AdaIN}(\mathbf{x}_i, \mathbf{y}) = \mathbf{y}_{s,i} \frac{\mathbf{x}_i - \mu(\mathbf{x}_i)}{\sigma(\mathbf{x}_i)} + \mathbf{y}_{b,i}$$

Instance normalization = normalize per channel per sample

Adaptive \_\_\_\_\_ = addition of learned  $\mathbf{y}$  inputs above

Note: layer normalization = normalize per sample

# StyleGAN - Style Transfer



# StyleGAN



# StyleGAN - Effect of adding noise

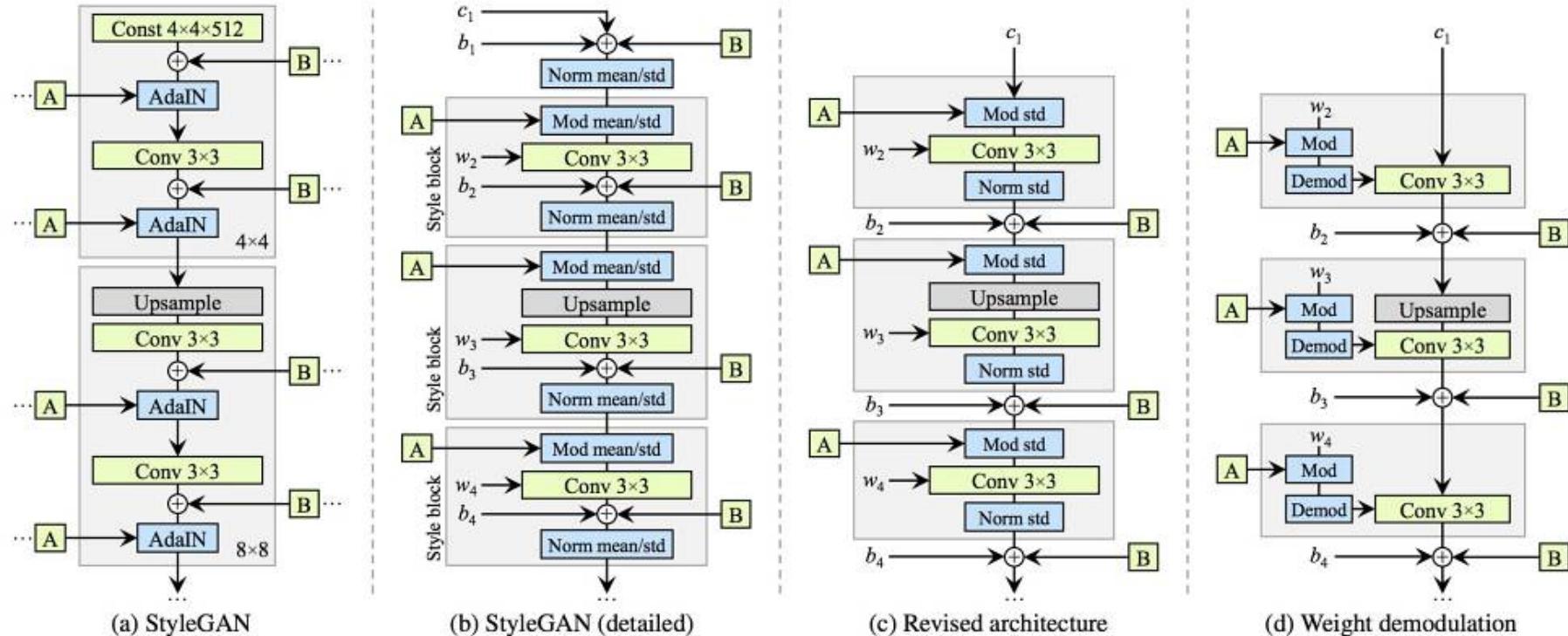


# StyleGAN-v2



Figure 1. Instance normalization causes water droplet -like artifacts in StyleGAN images. These are not always obvious in the generated images, but if we look at the activations inside the generator network, the problem is always there, in all feature maps starting from the 64x64 resolution. It is a systemic problem that plagues all StyleGAN images.

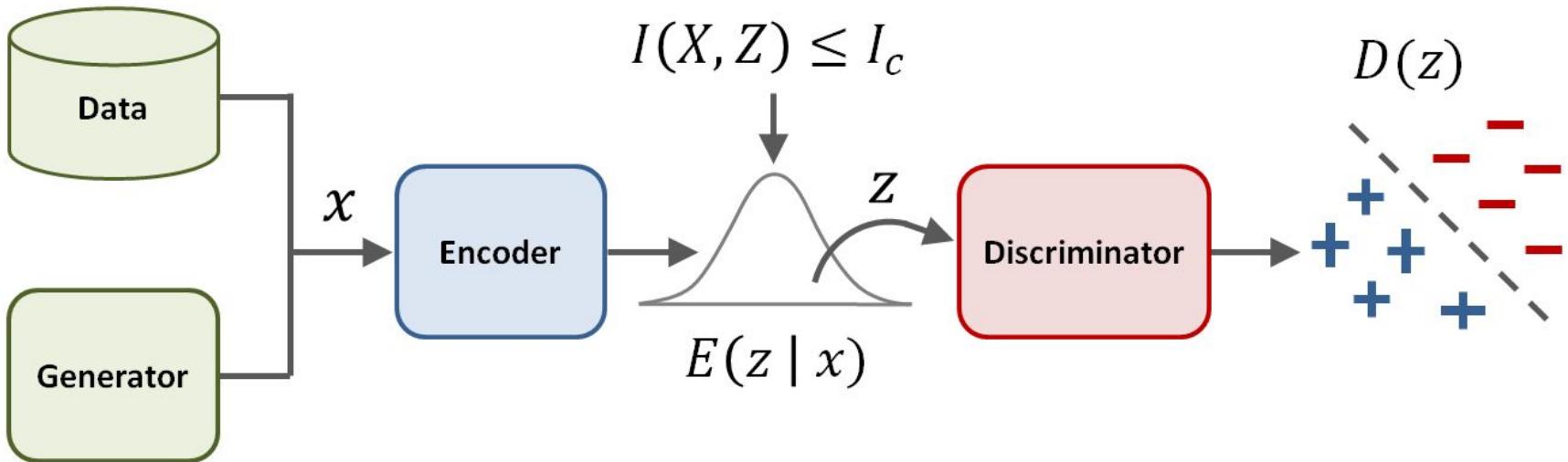
# StyleGAN-v2



# StyleGAN-v2



# Information Bottleneck



Variational Information Bottleneck [Alemi et al., 2016]

Variational Information Bottleneck GAN [Peng et al, 2019]

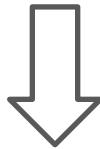
# Information Bottleneck

---

$$I(X, Z) \leq I_c$$

# Information Bottleneck

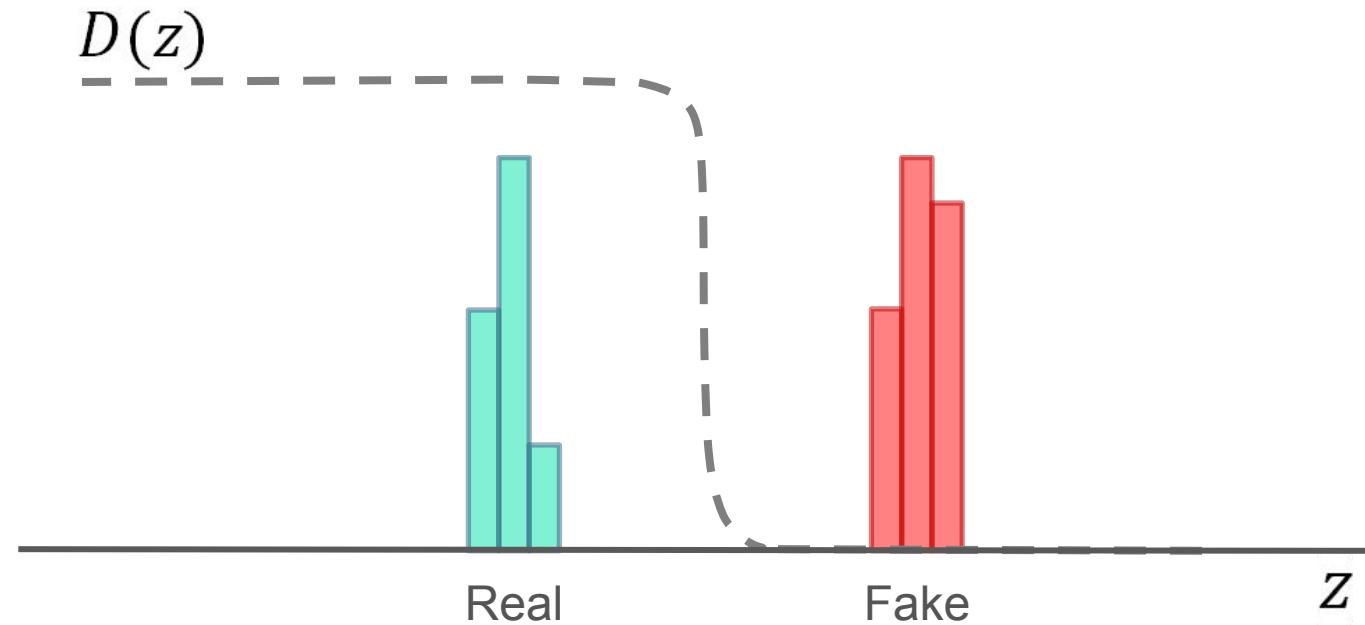
$$I(X, Z) \leq I_c$$



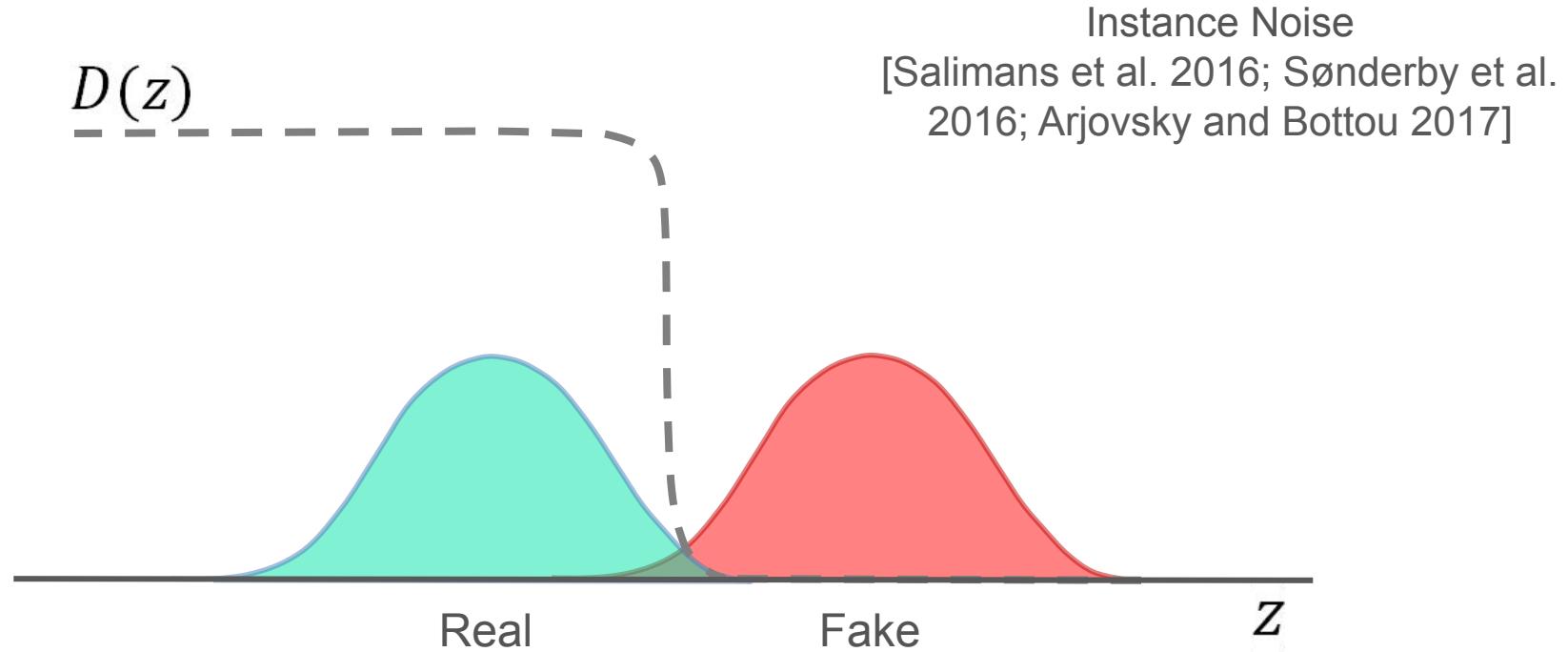
$$\mathbb{E}_{\mathbf{x} \sim \tilde{p}(\mathbf{x})} [\text{KL} [E(\mathbf{z}|\mathbf{x}) || r(\mathbf{z})]] \leq I_c$$

Variational Information Bottleneck (VIB)  
[Alemi et al., 2016]

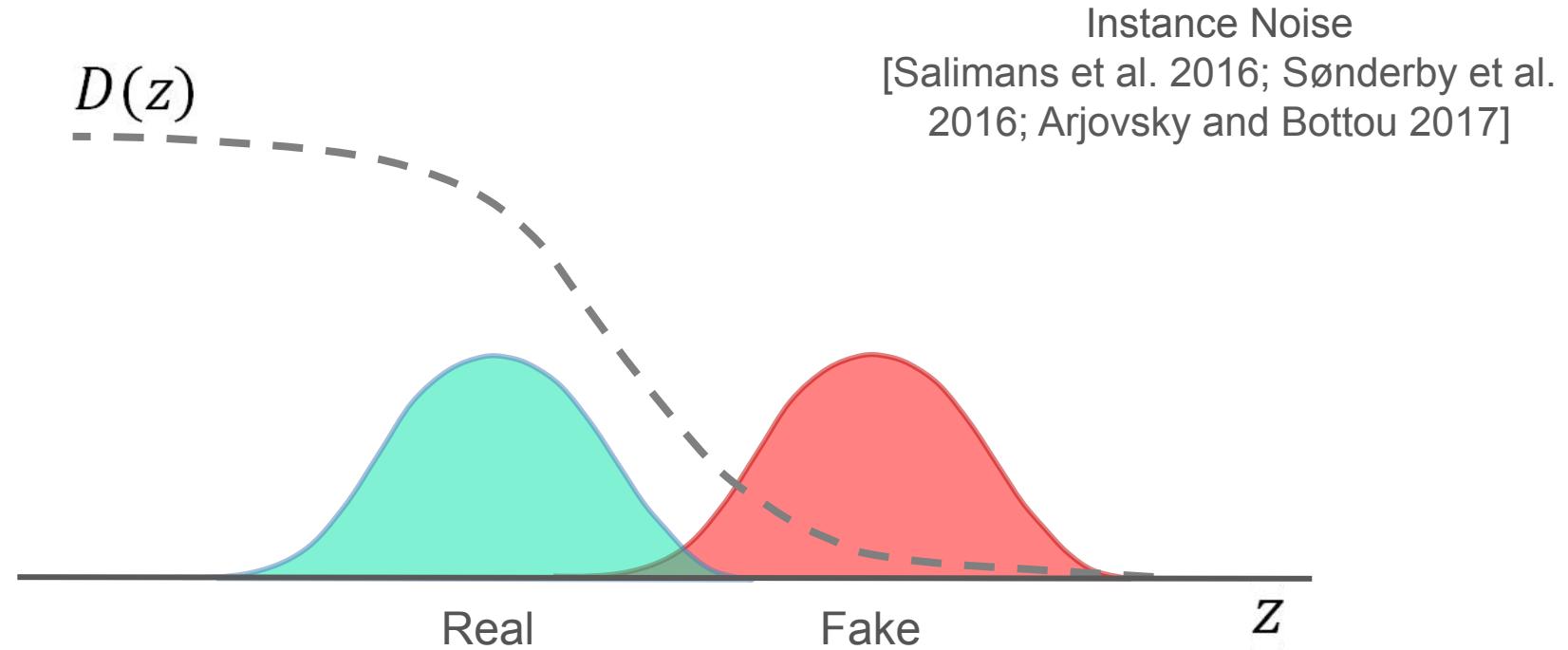
# Variational Information Bottleneck



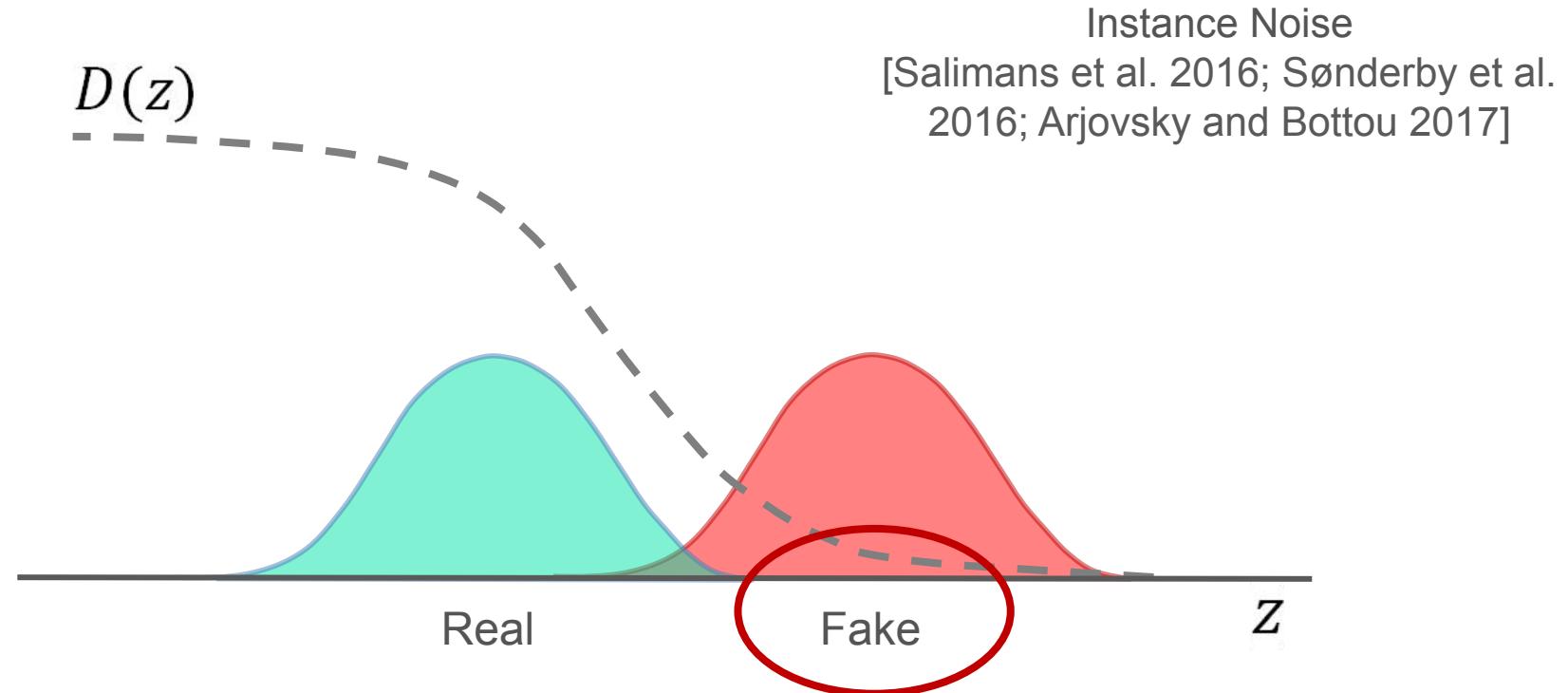
# Variational Information Bottleneck



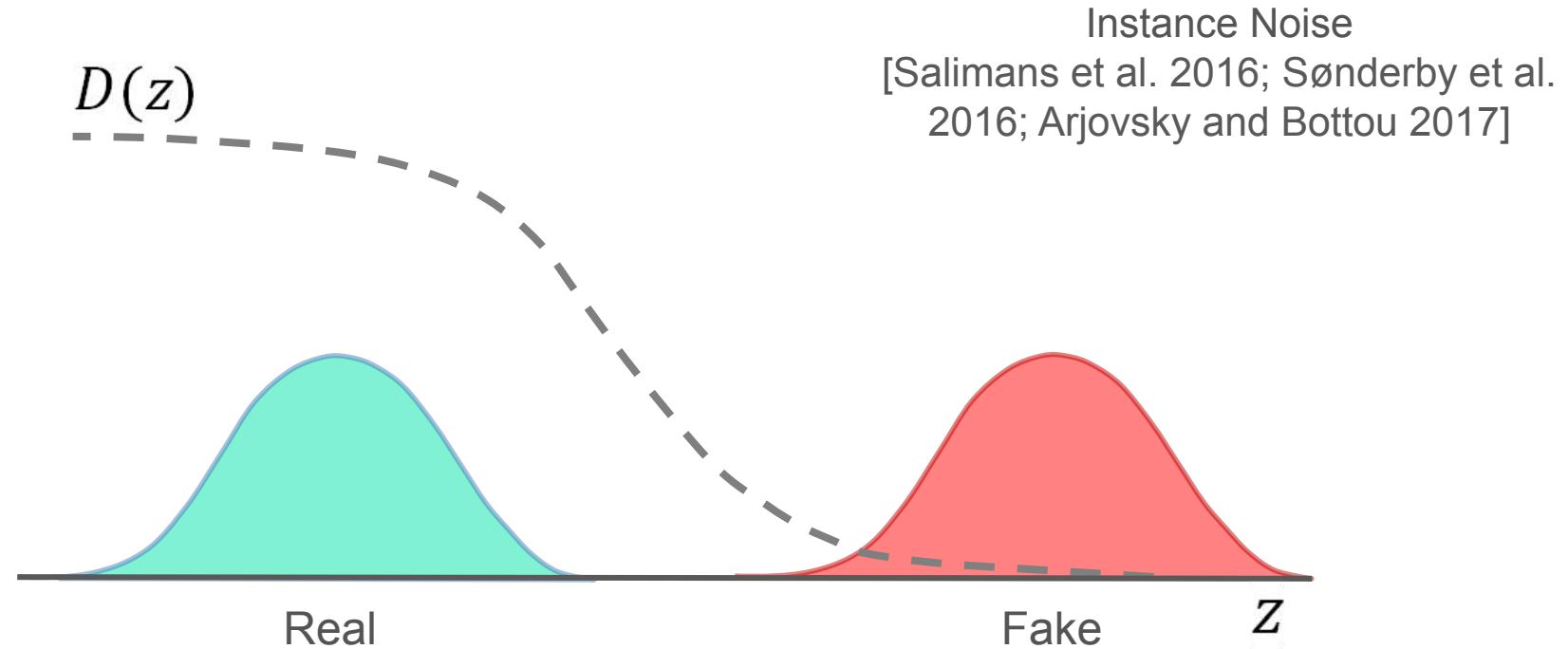
# Variational Information Bottleneck



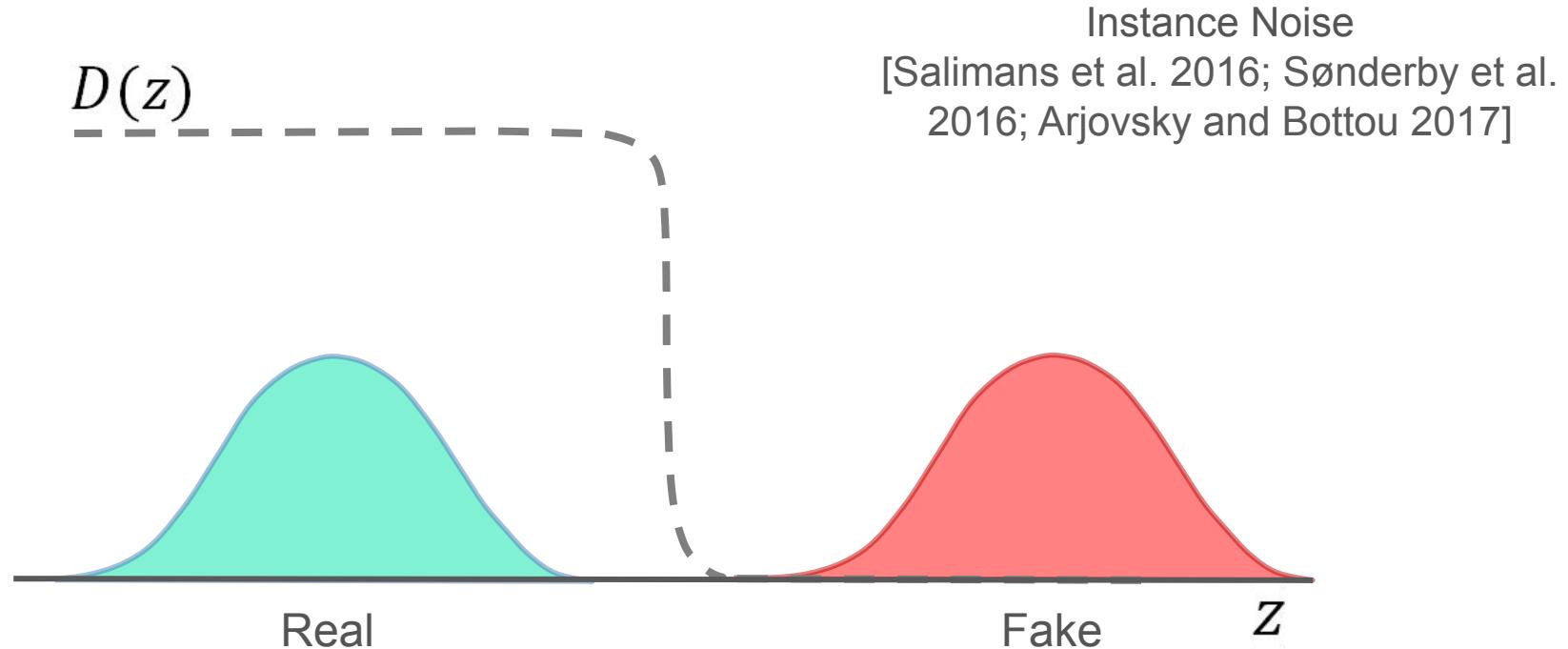
# Variational Information Bottleneck



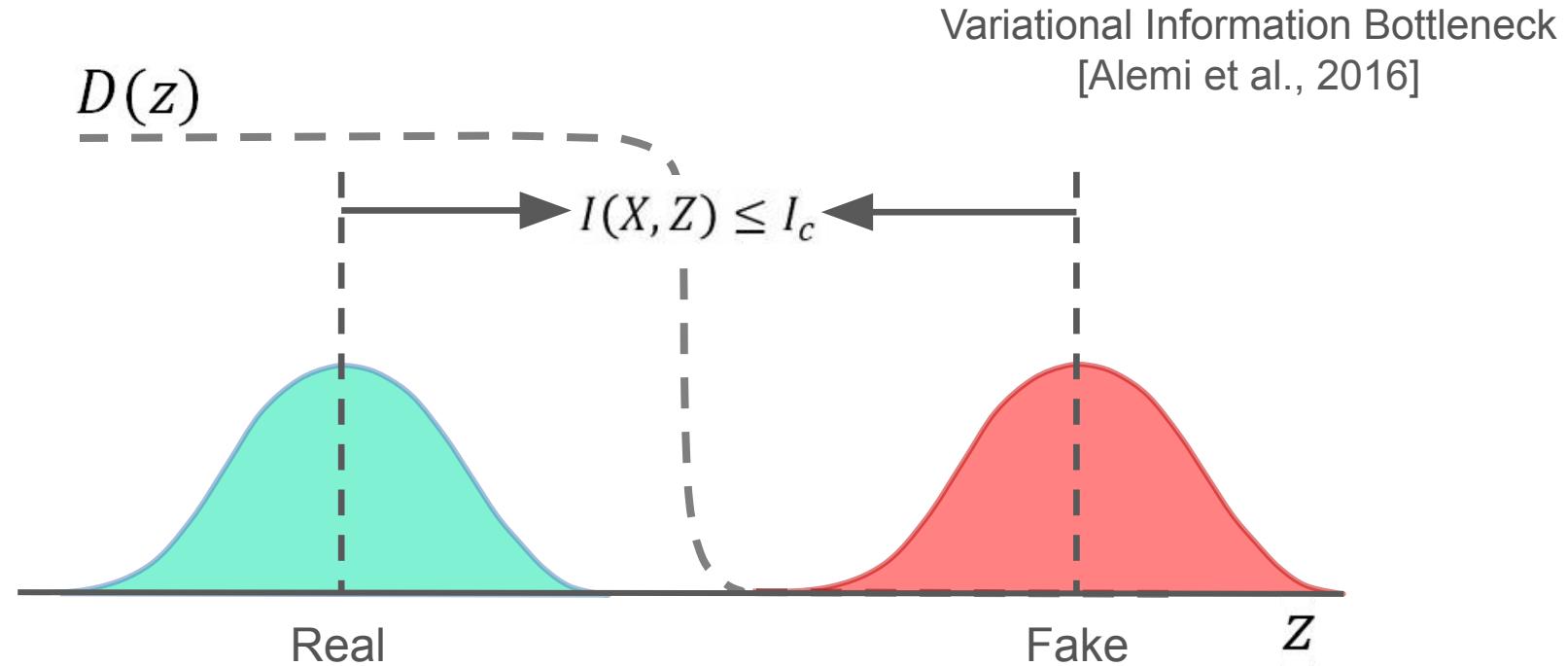
# Variational Information Bottleneck



# Variational Information Bottleneck

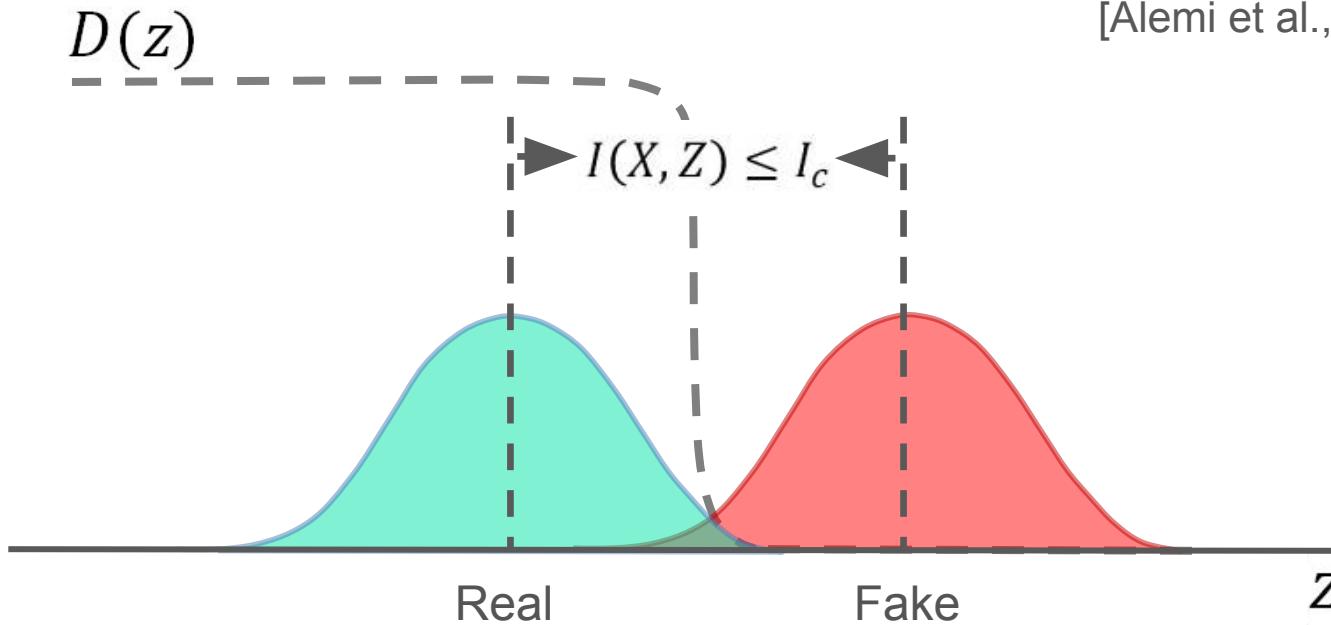


# Variational Information Bottleneck



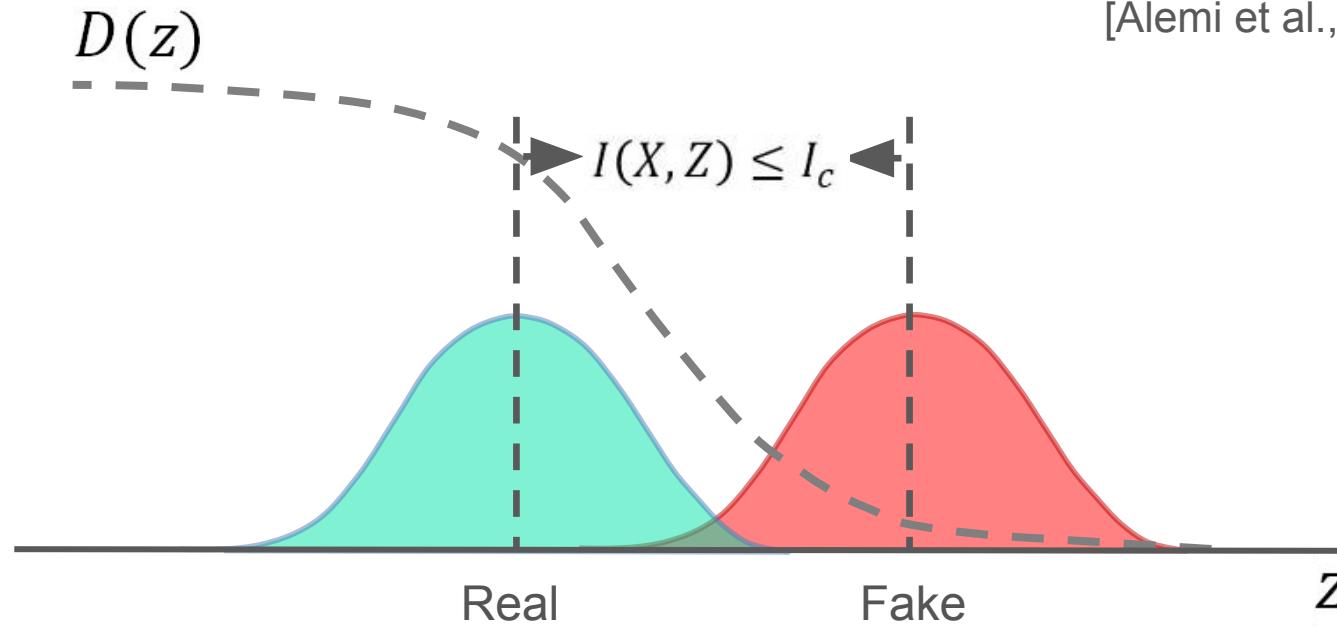
# Variational Information Bottleneck

Variational Information Bottleneck  
[Alemi et al., 2016]



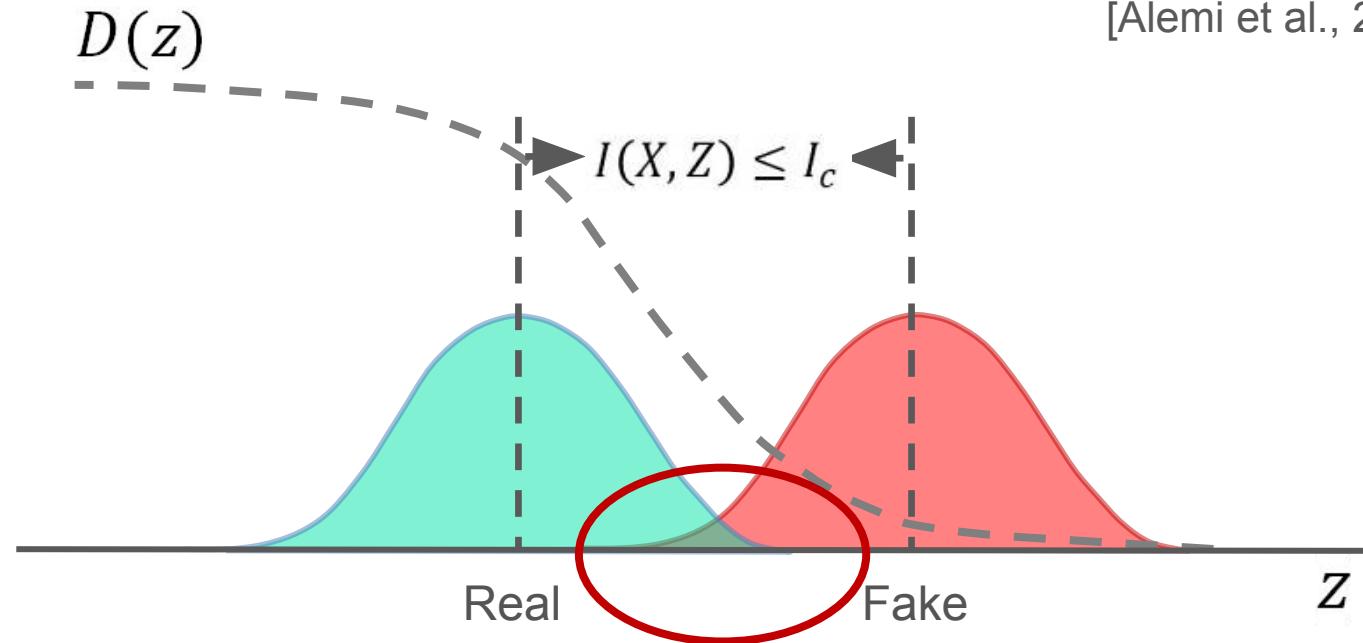
# Variational Information Bottleneck

Variational Information Bottleneck  
[Alemi et al., 2016]



# Variational Information Bottleneck

Variational Information Bottleneck  
[Alemi et al., 2016]

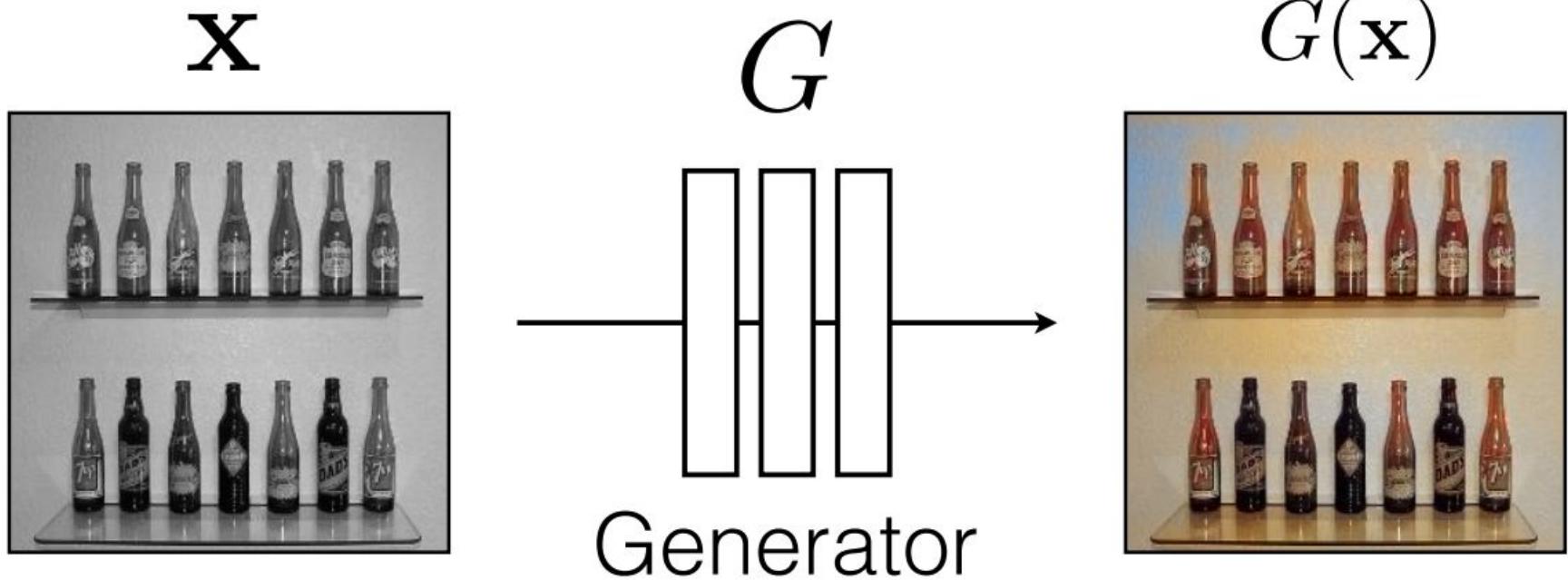


# Outline

---

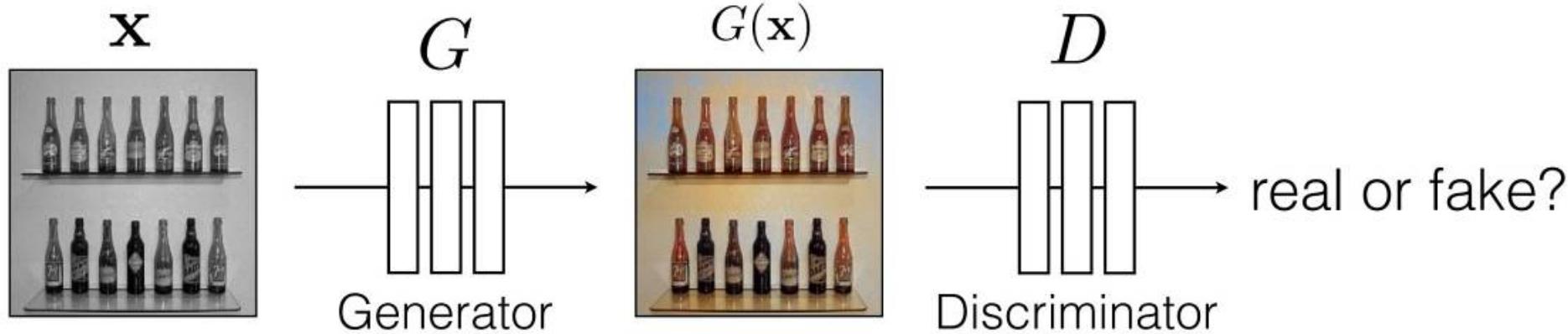
- Motivation & Definition of Implicit Models
- Original GAN (Goodfellow et al, 2014)
- Evaluation: Parzen, Inception, Frechet
- Some Theory: Bayes-optimal Discriminator; Jensen-Shannon Divergence; Mode Collapse; Avoiding Saturation
- GAN Progression:
  - DC GAN (Radford et al, 2016)
  - Improved Training of GANs (Salimans et al, 2016)
  - WGAN, WGAN-GP, Progressive GAN, SN-GAN, SAGAN
  - BigGAN, BigGAN-Deep, StyleGAN, StyleGAN-v2, VIB-GAN
- ***Creative Conditional GANs***
  - GANs and Representations
  - GANs as Energy Models
  - GANs and Optimal Transport, Implicit Likelihood Models, Moment Matching
  - Other uses of Adversarial Loss: Transfer Learning, Fairness
  - GANs and Imitation Learning

# Conditional GANs / pix2pix



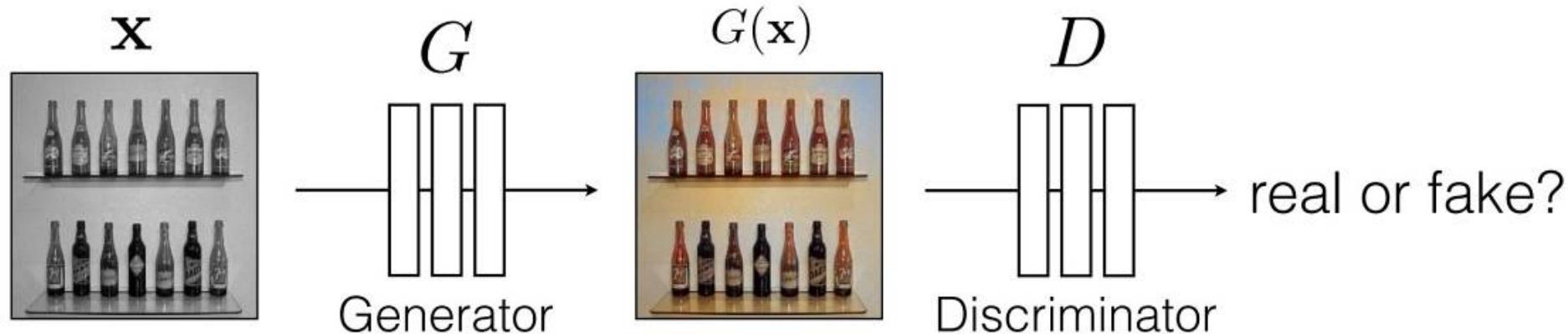
Slide: Phillip Isola

# Conditional GANs / pix2pix



Slide: Phillip Isola

# Conditional GANs / pix2pix

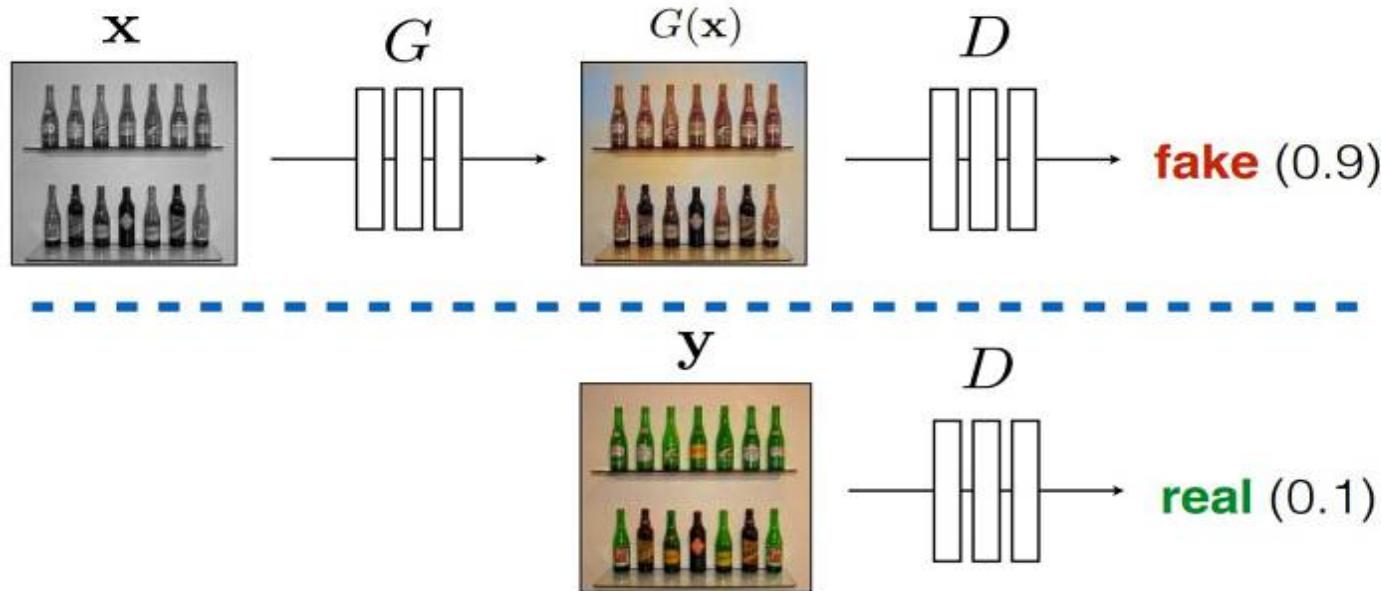


**G** tries to synthesize fake images that fool **D**

**D** tries to identify the fakes

Slide: Phillip Isola

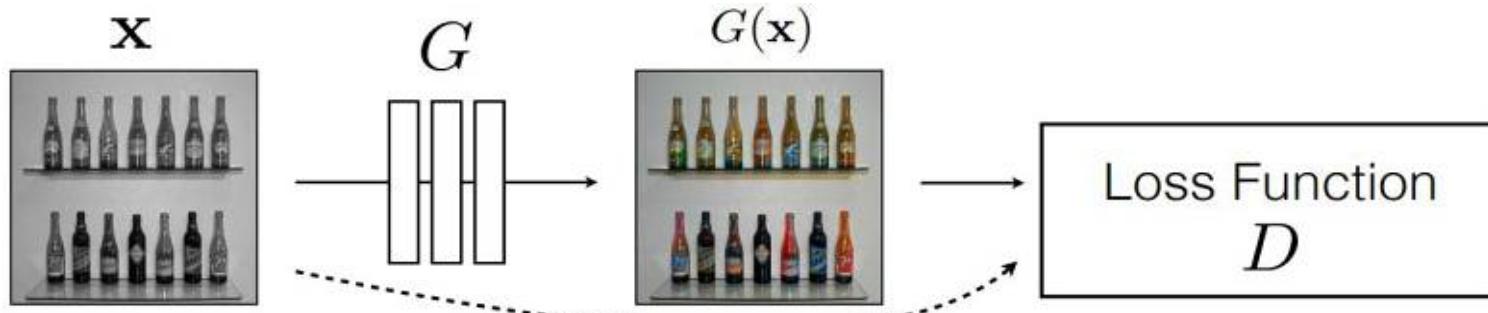
# Conditional GANs / pix2pix



$$\arg \max_D \mathbb{E}_{\mathbf{x}, \mathbf{y}} [ \boxed{\log D(G(\mathbf{x}))} + \boxed{\log(1 - D(\mathbf{y}))} ]$$

Slide: Phillip Isola

# Conditional GANs / pix2pix

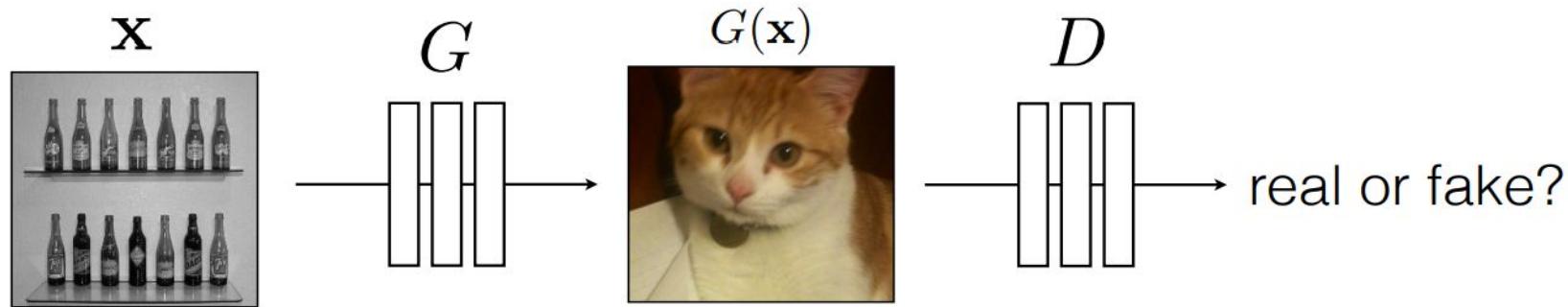


**G**'s perspective: **D** is a loss function.

Rather than being hand-designed, it is *learned*.

Slide: Phillip Isola

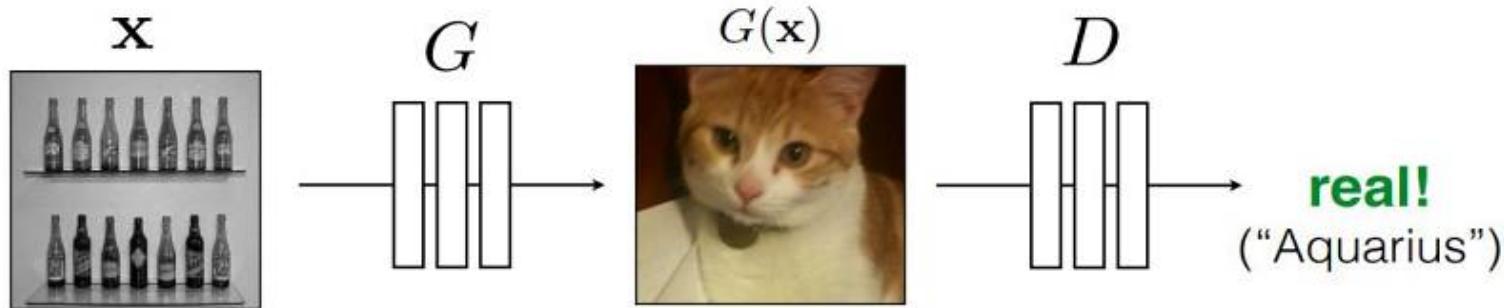
# Conditional GANs / pix2pix



$$\arg \min_G \max_D \mathbb{E}_{\mathbf{x}, \mathbf{y}} [ \log D(G(\mathbf{x})) + \log(1 - D(\mathbf{y})) ]$$

Slide: Phillip Isola

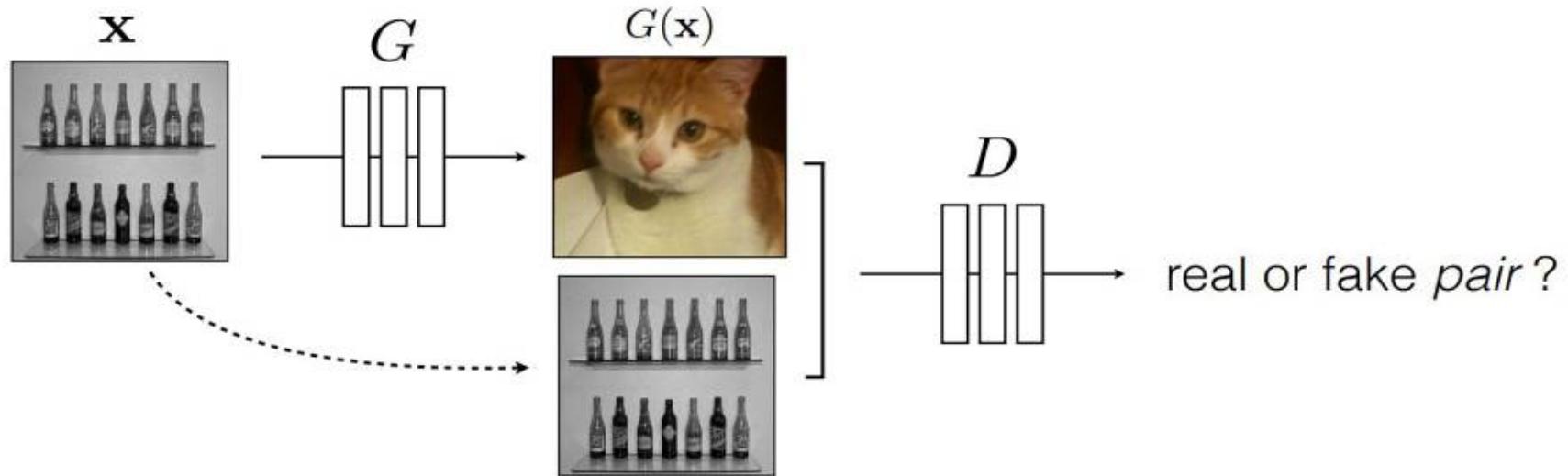
# Conditional GANs / pix2pix



$$\arg \min_G \max_D \mathbb{E}_{\mathbf{x}, \mathbf{y}} [ \log D(G(\mathbf{x})) + \log(1 - D(\mathbf{y})) ]$$

Slide: Phillip Isola

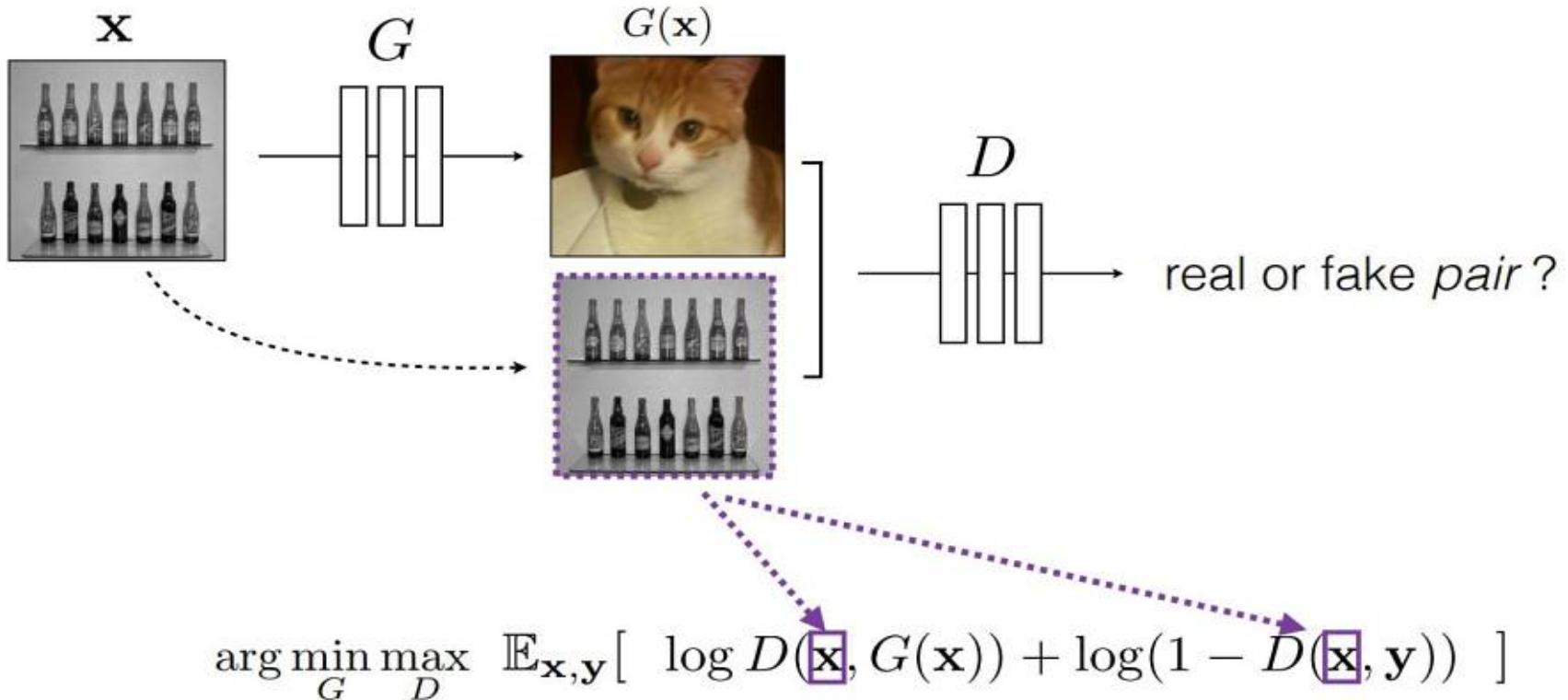
# Conditional GANs / pix2pix



$$\arg \min_G \max_D \mathbb{E}_{\mathbf{x}, \mathbf{y}} [ \log D(G(\mathbf{x})) + \log(1 - D(\mathbf{y})) ]$$

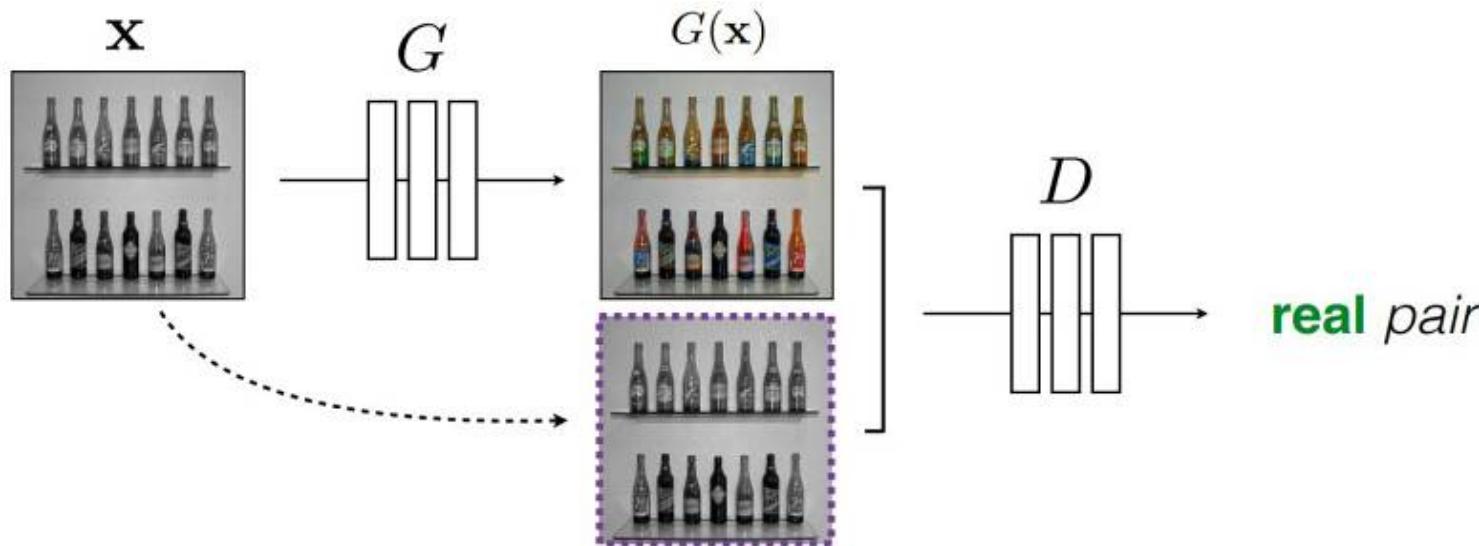
Slide: Phillip Isola

# Conditional GANs / pix2pix



Slide: Phillip Isola

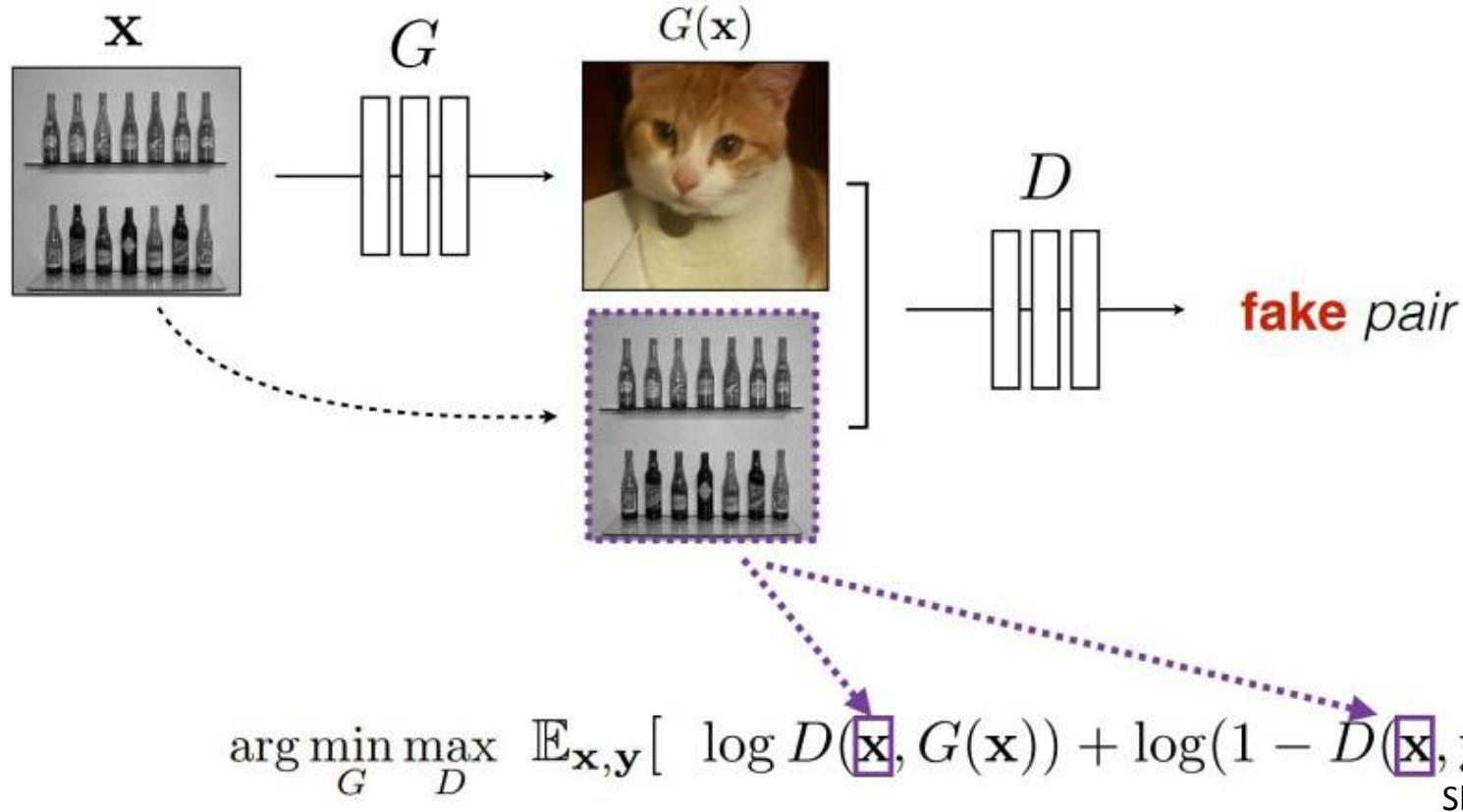
# Conditional GANs / pix2pix



$$\arg \min_G \max_D \mathbb{E}_{\mathbf{x}, \mathbf{y}} [ \log D(\mathbf{x}, G(\mathbf{x})) + \log(1 - D(\mathbf{x}, \mathbf{y})) ]$$

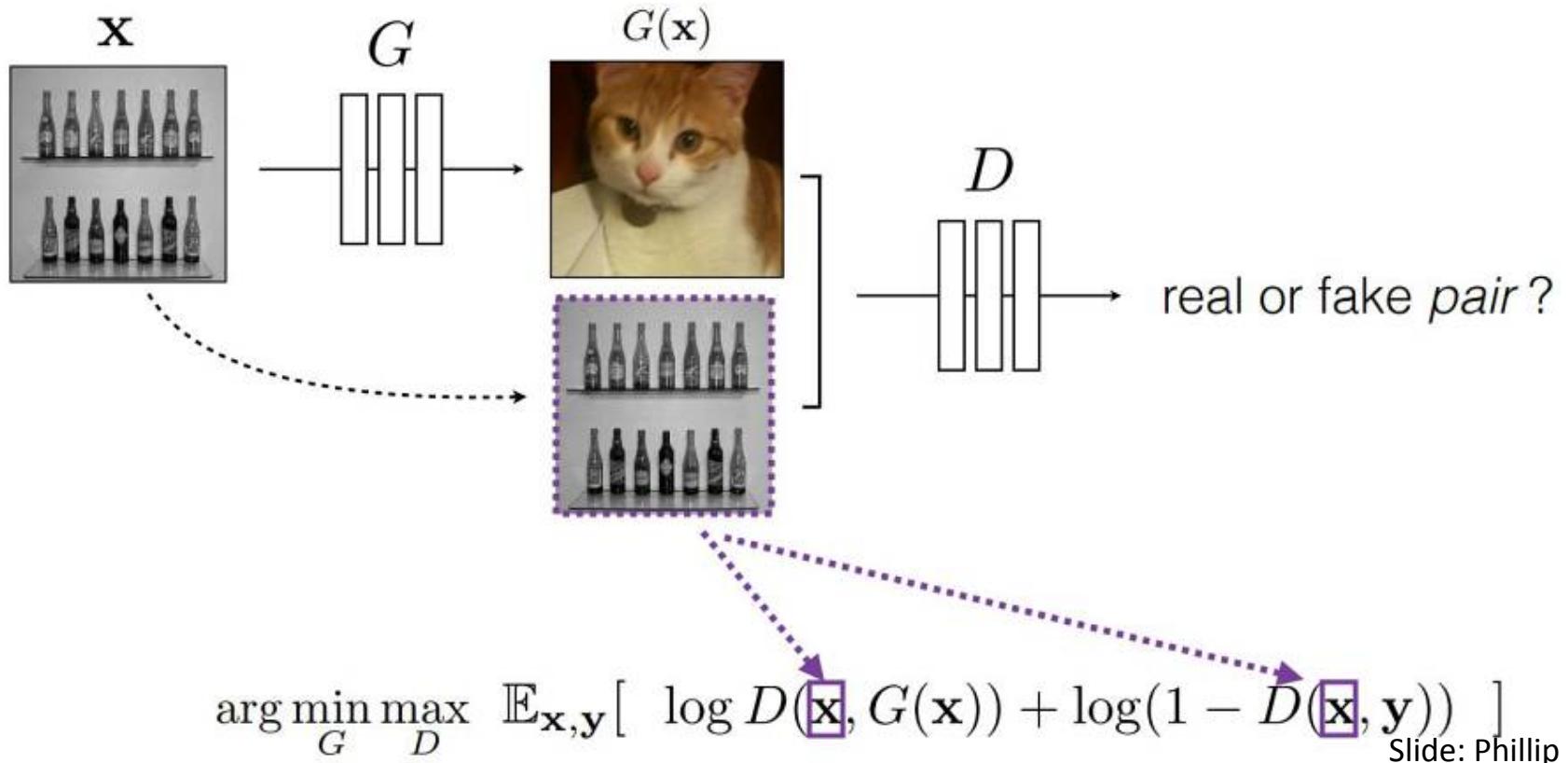
Slide: Phillip Isola

# Conditional GANs / pix2pix

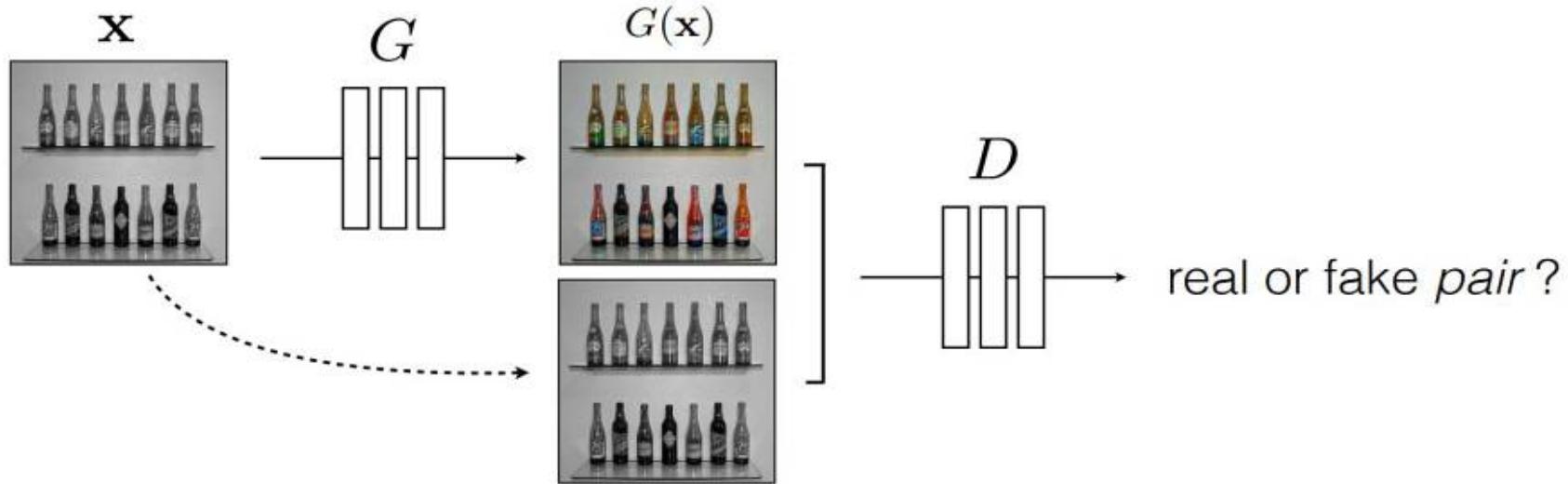


Slide: Phillip Isola

# Conditional GANs / pix2pix



# Conditional GANs / pix2pix



$$\arg \min_G \max_D \mathbb{E}_{\mathbf{x}, \mathbf{y}} [ \log D(\mathbf{x}, G(\mathbf{x})) + \log(1 - D(\mathbf{x}, \mathbf{y})) ]$$

Slide: Phillip Isola

# Conditional GANs / pix2pix

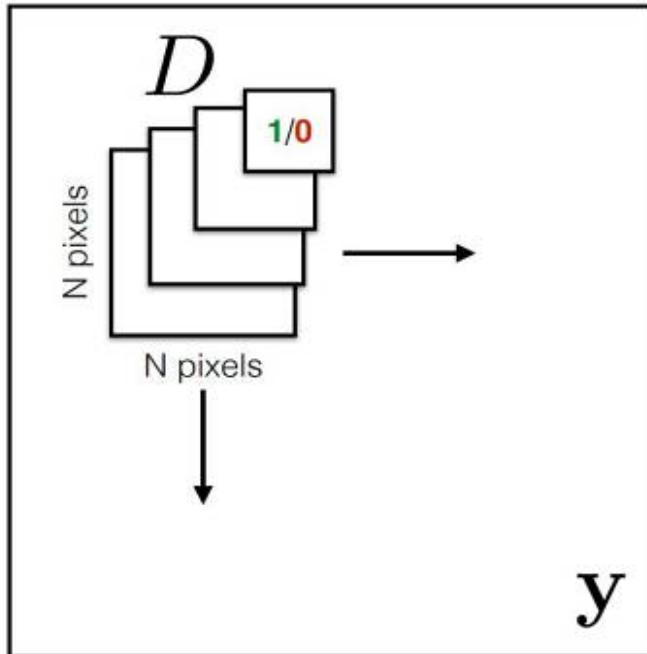
---

$$G^* = \arg \min_G \max_D \mathcal{L}_{cGAN}(G, D) + \lambda \mathcal{L}_{L1}(G)$$

Slide: Phillip Isola

# Conditional GANs / pix2pix

Shrinking the capacity: Patch Discriminator

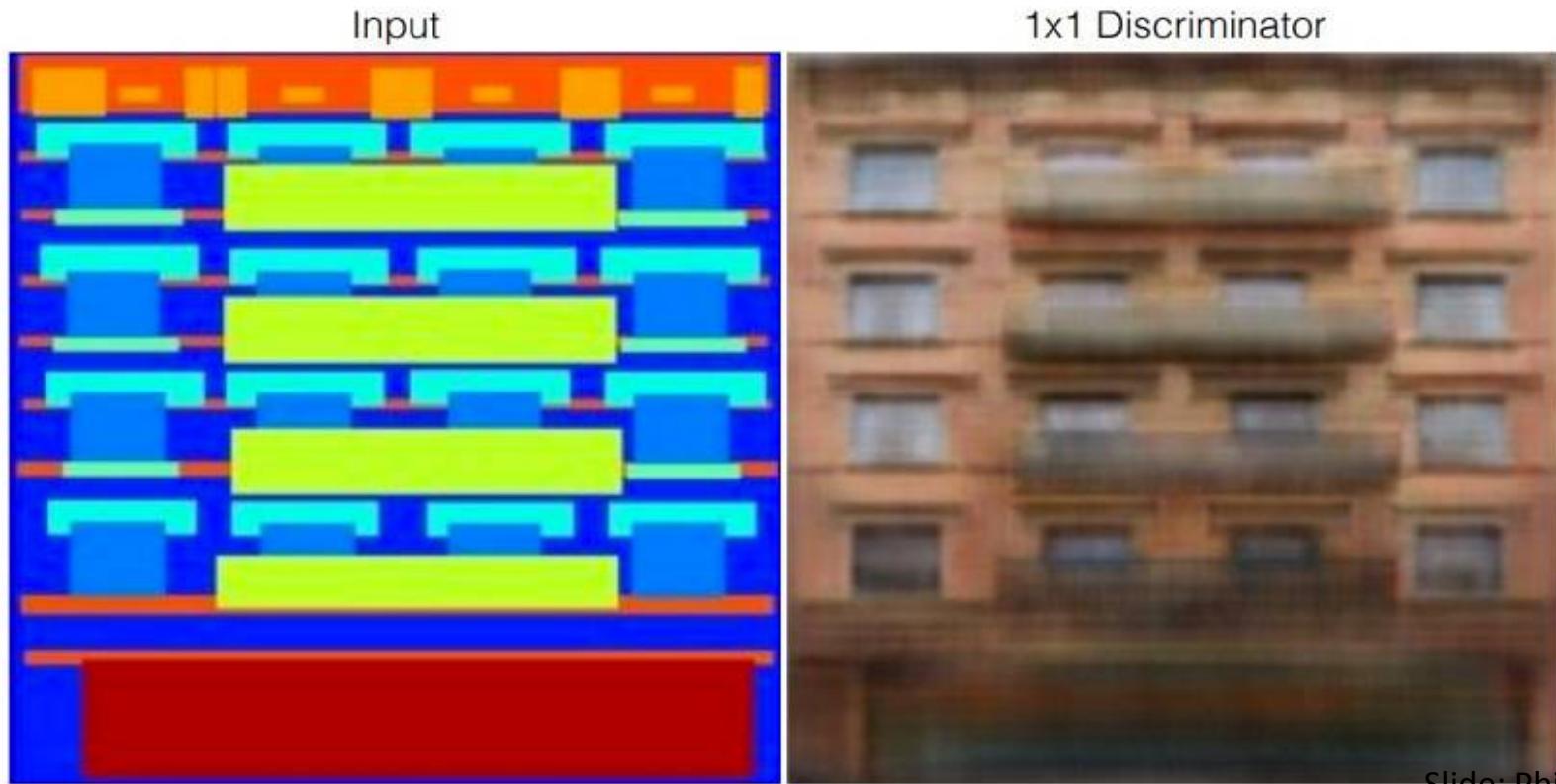


Rather than penalizing if output *image* looks fake, penalize if each overlapping *patch* in output looks fake

[Li & Wand 2016]  
[Shrivastava et al. 2017]  
[Isola et al. 2017]

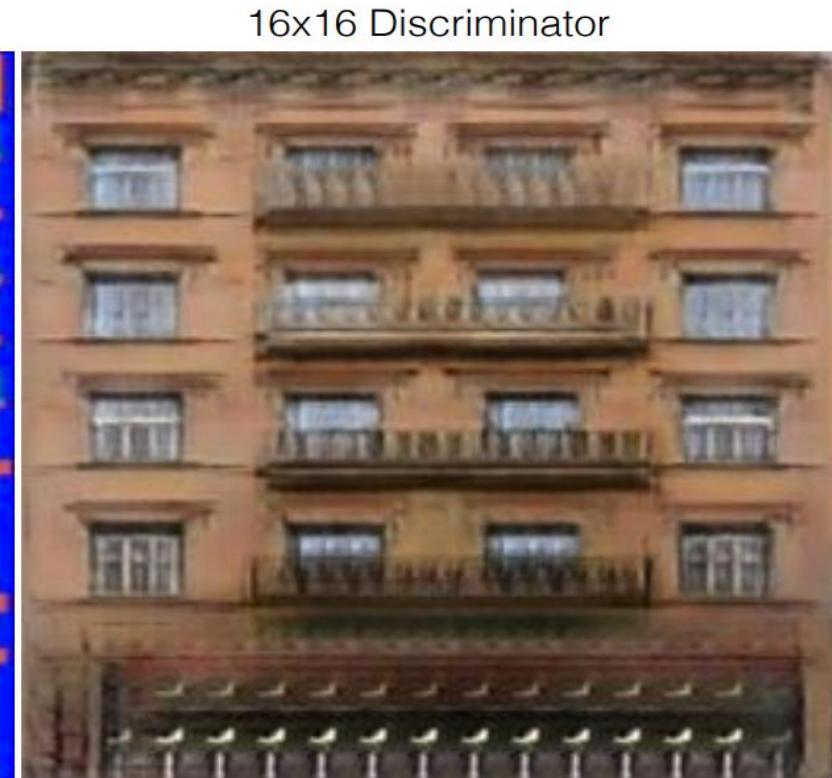
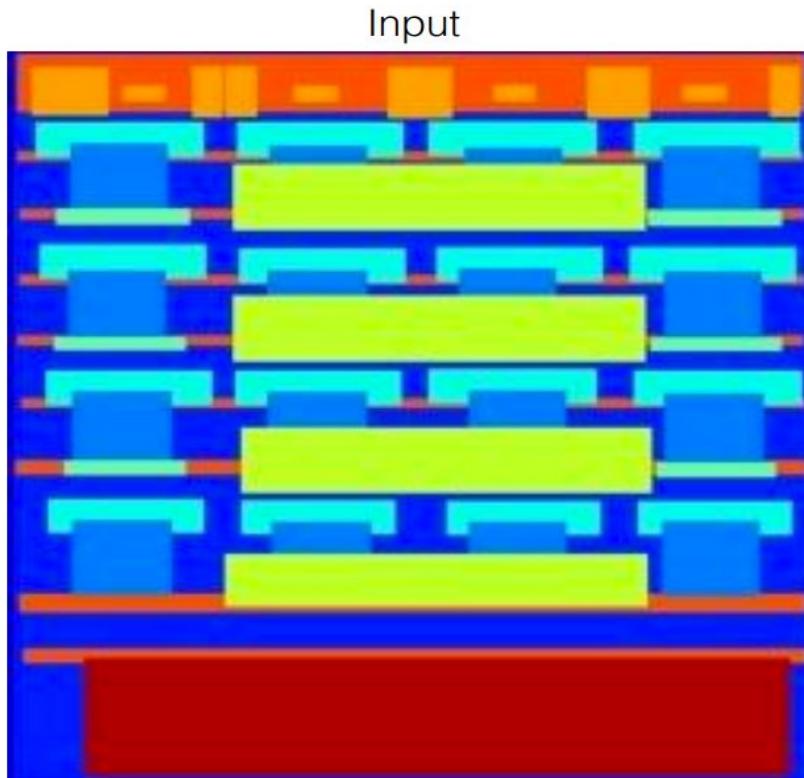
Slide: Phillip Isola

# Conditional GANs / pix2pix



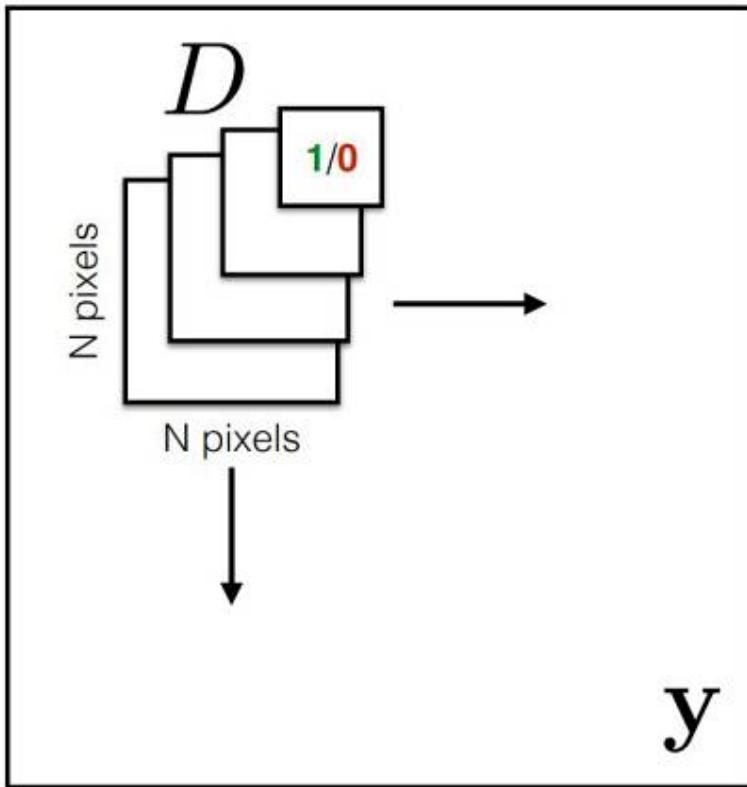
Slide: Phillip Isola

# Conditional GANs / pix2pix



Slide: Phillip Isola

# Conditional GANs / pix2pix

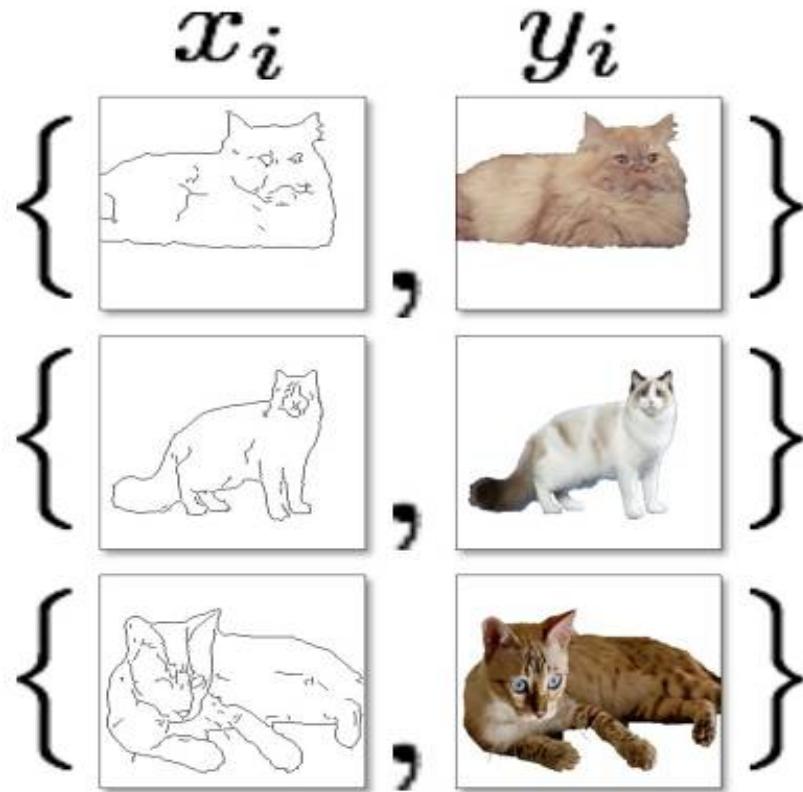


Rather than penalizing if output *image* looks fake, penalize if each overlapping *patch* in output looks fake

- Faster, fewer parameters
- More supervised observations
- Applies to arbitrarily large images

Slide: Phillip Isola

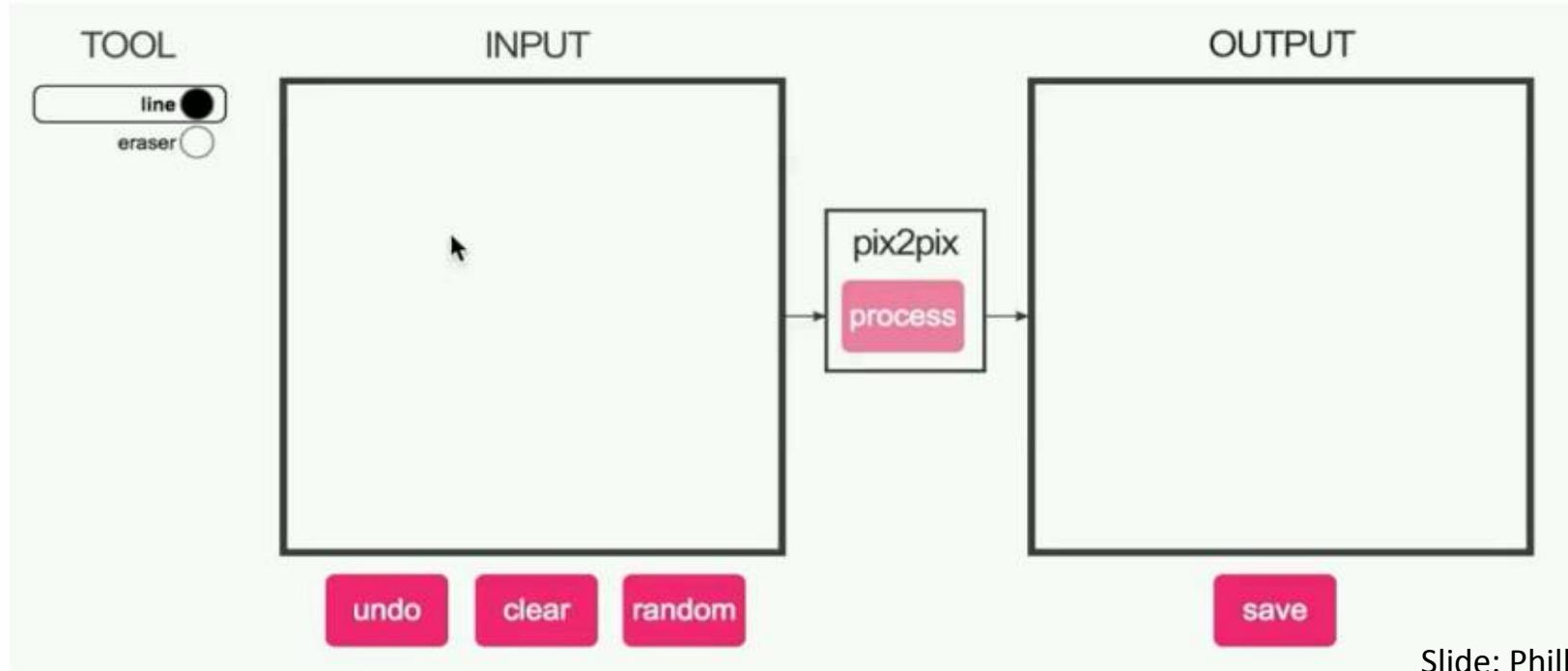
# Conditional GANs / pix2pix



Slide: Phillip Isola

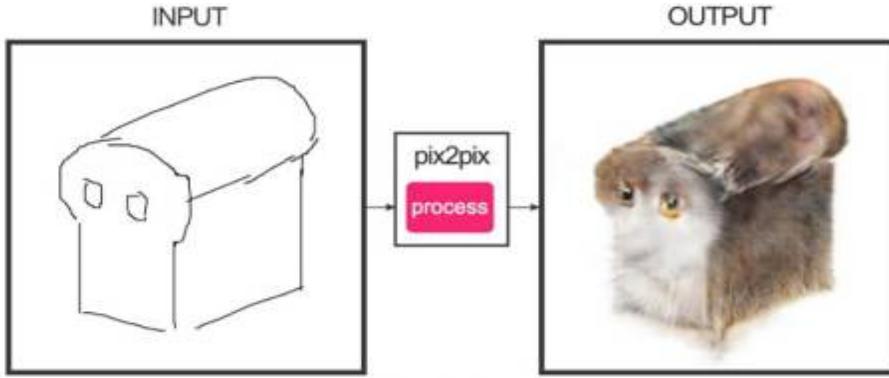
# Conditional GANs / pix2pix

#edges2cats [Chris Hesse]



Slide: Phillip Isola

# Conditional GANs / pix2pix



Ivy Tasi @ivymyt



Vitaly Vidmirov @vvid

Slide: Phillip Isola

# Conditional GANs / pix2pix

BW → Color

Input



Output



Input



Output



Input



Output



Slide: Phillip Isola

# Video2Video (NVIDIA)



# Everybody Dance Now



# NVIDIA GauGAN: sketch->photorealistic image



# Learning to paint (GANs + RL)

---

<https://learning-to-paint.github.io/>

# Outline

---

- Motivation & Definition of Implicit Models
- Original GAN (Goodfellow et al, 2014)
- Evaluation: Parzen, Inception, Frechet
- Some Theory: Bayes-optimal Discriminator; Jensen-Shannon Divergence; Mode Collapse; Avoiding Saturation
- GAN Progression:
  - DC GAN (Radford et al, 2016)
  - Improved Training of GANs (Salimans et al, 2016)
  - WGAN, WGAN-GP, Progressive GAN, SN-GAN, SAGAN
  - BigGAN, BigGAN-Deep, StyleGAN, StyleGAN-v2, VIB-GAN
- Creative Conditional GANs
- **GANs and Representations**
- GANs as Energy Models
- GANs and Optimal Transport, Implicit Likelihood Models, Moment Matching
- Other uses of Adversarial Loss: Transfer Learning, Fairness
- GANs and Imitation Learning

# GANs for unsupervised feature learning

---

- InfoGAN (Information Maximizing GAN)
- BiGAN (Bidirectional Generative Adversarial Networks)
  - BigBiGAN (Big Bidirectional Generative Adversarial Networks)

# InfoGAN



```
array([[151, 157, 250, ..., 20, 0, 0],  
       [148, 161, 242, ..., 15, 0, 0],  
       [235, 228, 255, ..., 3, 0, 0],  
       ...,  
       [252, 254, 176, ..., 240, 253, 253],  
       [253, 253, 253, ..., 253, 200, 200],  
       [253, 253, 253, ..., 253, 200, 200]]  
     , dtype=uint8)
```

# InfoGAN

```
array([[151, 157, 250, ..., 20, 0, 0],  
       [148, 161, 242, ..., 15, 0, 0],  
       [235, 228, 255, ..., 3, 0, 0],  
       ...,  
       [252, 254, 176, ..., 240, 253, 253],  
       [253, 253, 253, ..., 253, 200, 200],  
       [253, 253, 253, ..., 253, 200, 200]]  
, dtype=uint8)
```

Data: x

**Simple factors interact to create complex observations.**

Digit type: “5”

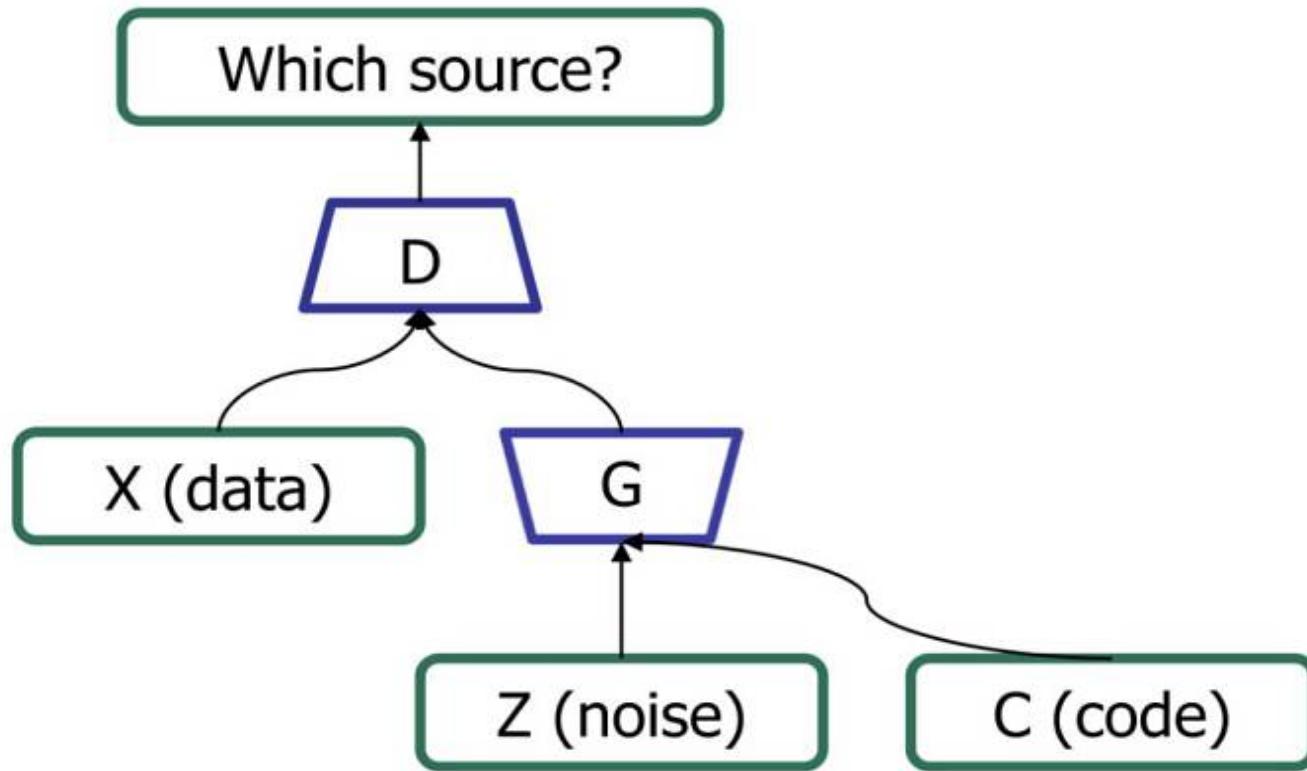
Rotation: Tilting to the right

Width: Medium

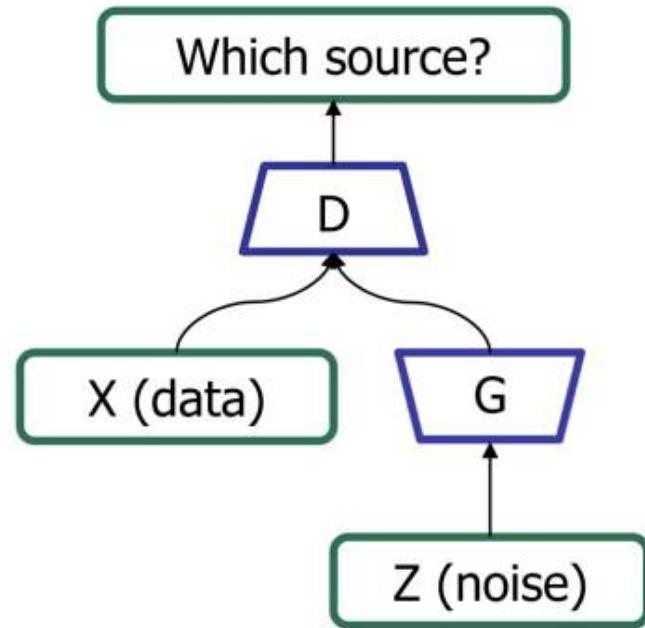
.....

Latent Code: c

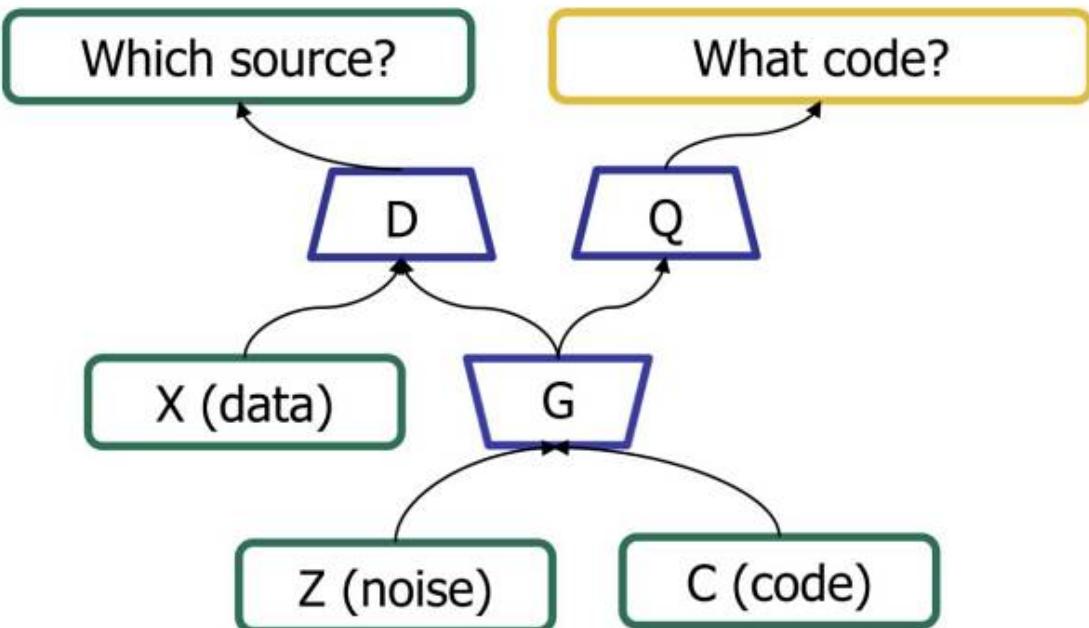
# InfoGAN



# InfoGAN

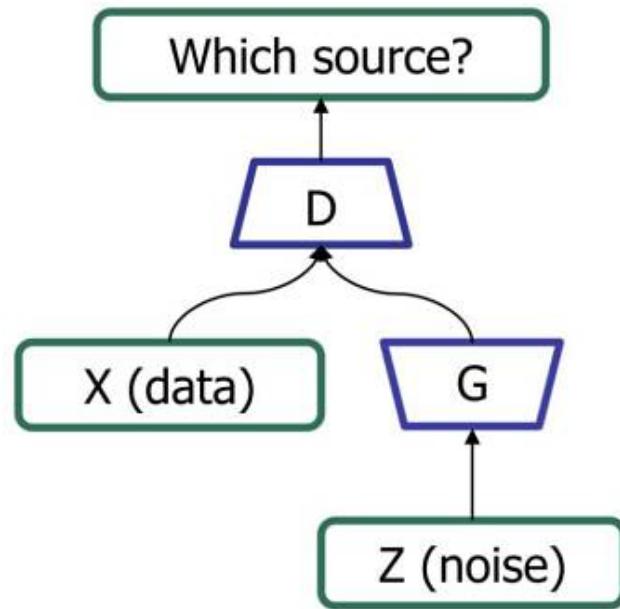


GAN

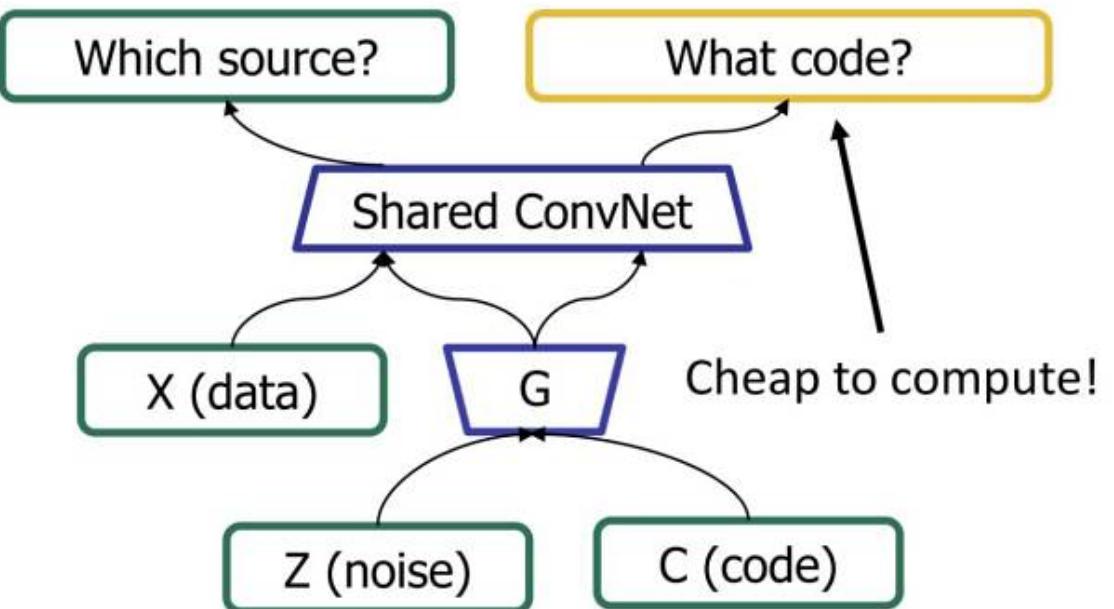


InfoGAN

# InfoGAN



GAN



InfoGAN

# InfoGAN

**Simple idea: Independent factors in latent code should maximally explain variations in generated images.**

Formally: We want to maximize the mutual information between latent code and generated images:

$$\begin{aligned}\max_G I(c; x) &= H(x) - H(x|c) \\ &= H(c) - H(c|x)\end{aligned}$$

where  $x = G(z, c)$

# InfoGAN

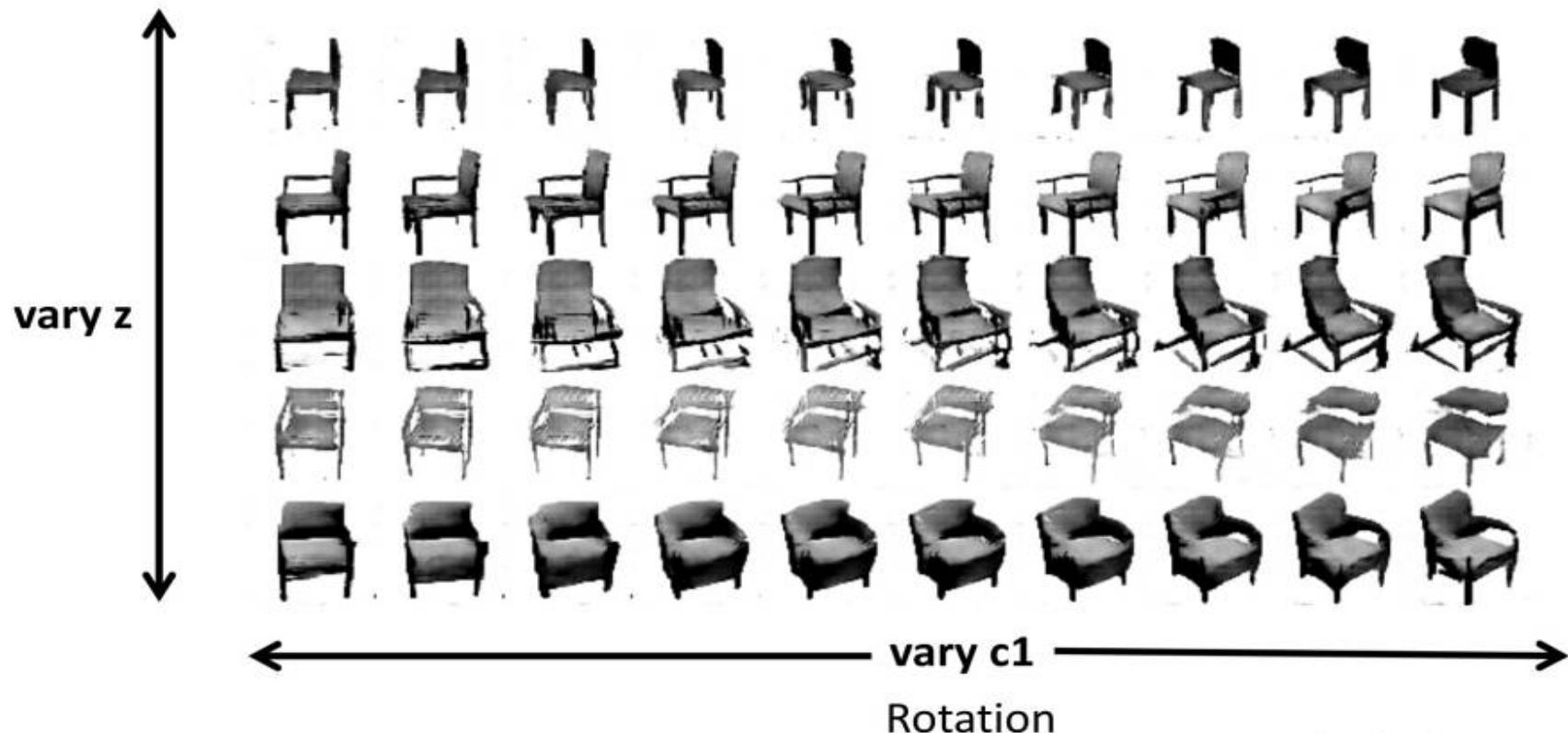
**Mutual information can be maximized easily with a variational lower bound:**

$$\begin{aligned} I(c; G(z, c)) &= H(c) - H(c|G(z, c)) \\ &= H(c) + \mathbb{E}_{x \sim G(z, c)} [\mathbb{E}_{c' \sim P(c|x)} [\log P(c'|x)]] \\ &= H(c) + \mathbb{E}_{x \sim G(z, c)} [\mathbb{E}_{c' \sim P(c|x)} [\log Q(c'|x)]] + \underbrace{D_{\text{KL}}(P(\cdot|x) \parallel Q(\cdot|x))}_{\geq 0} \\ &\geq H(c) + \mathbb{E}_{x \sim G(z, c)} [\mathbb{E}_{c' \sim P(c|x)} [\log Q(c'|x)]] \end{aligned}$$

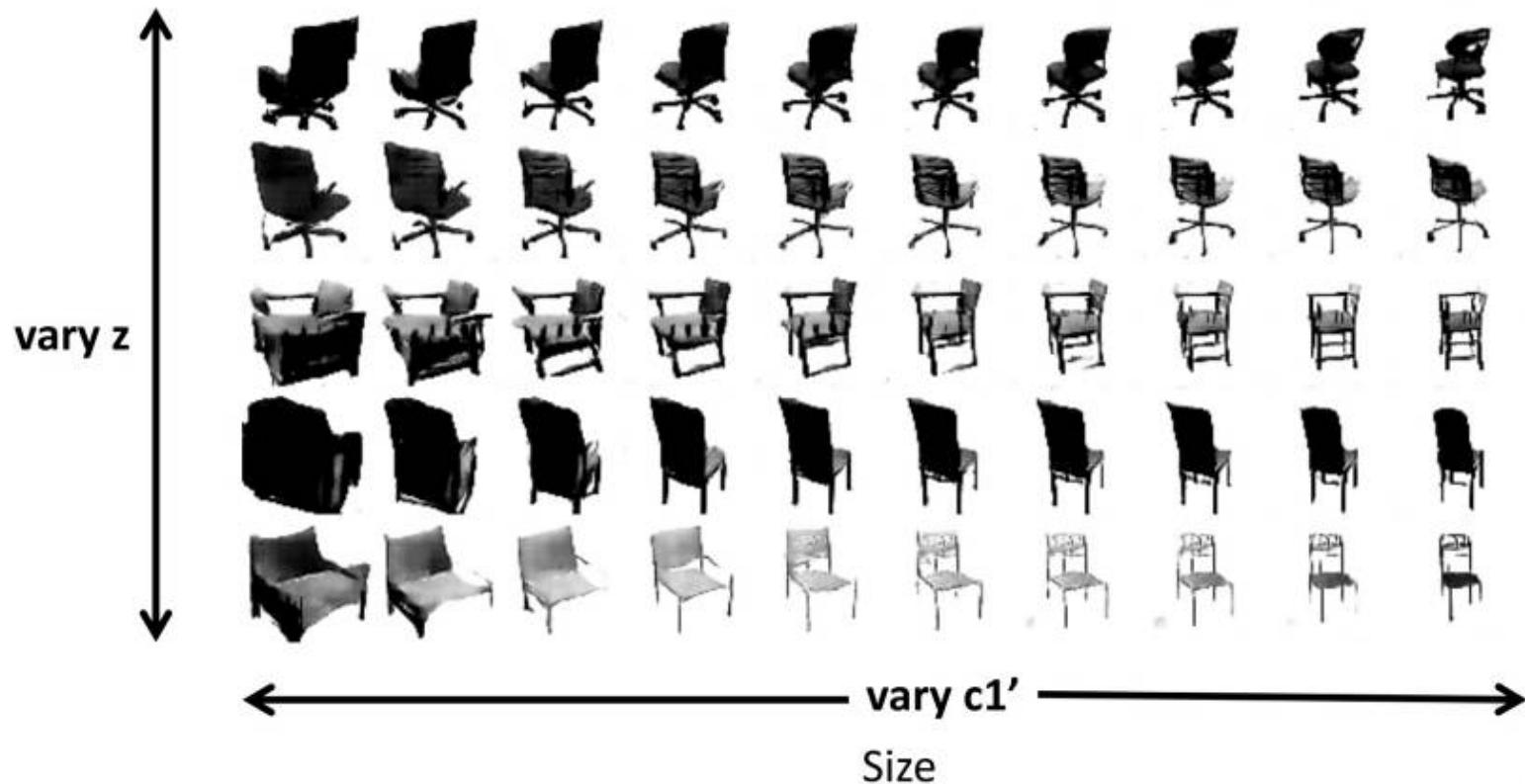


Simply MLE for a classifier/regressor

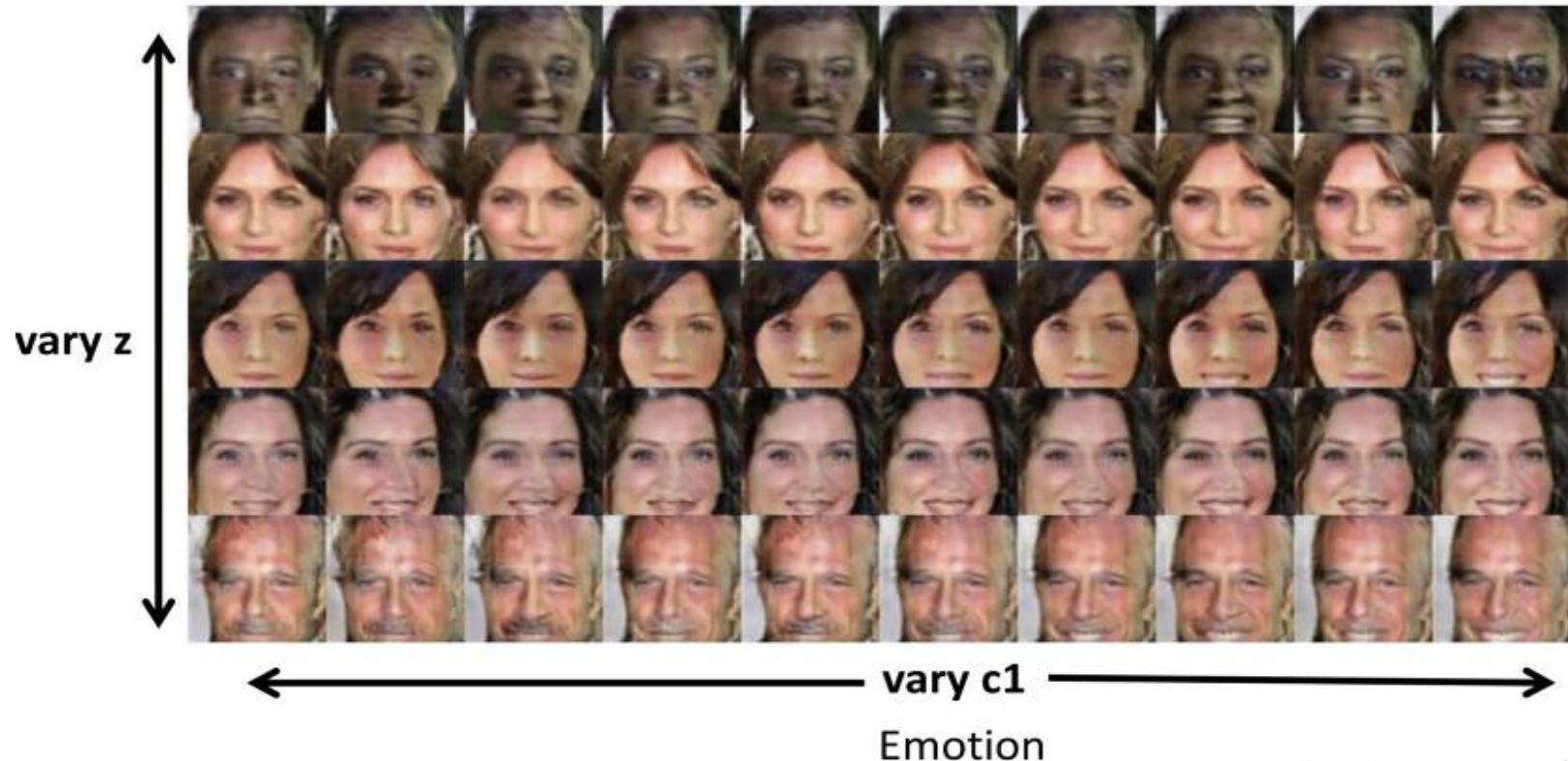
# InfoGAN



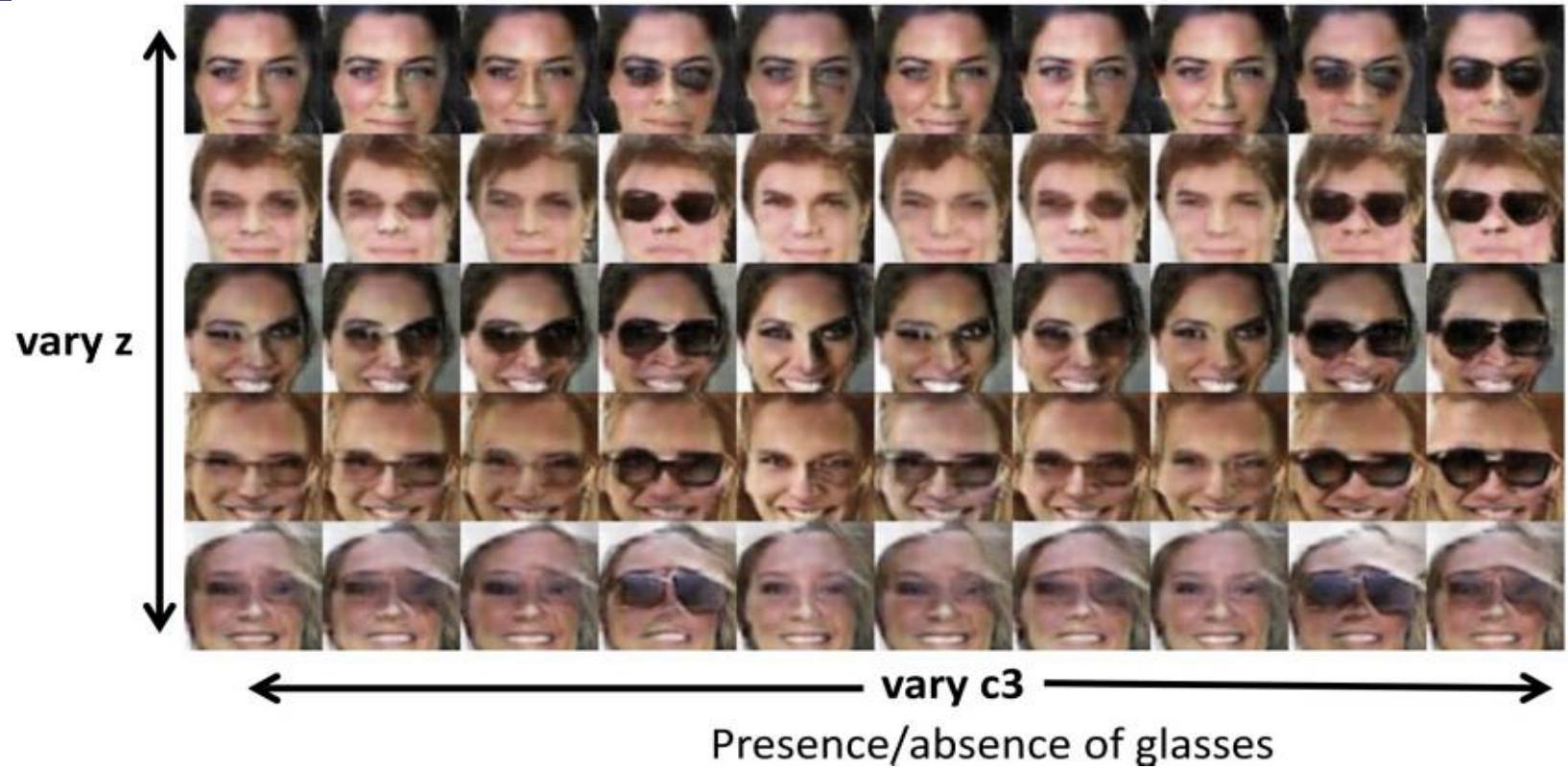
# InfoGAN



# InfoGAN



# InfoGAN



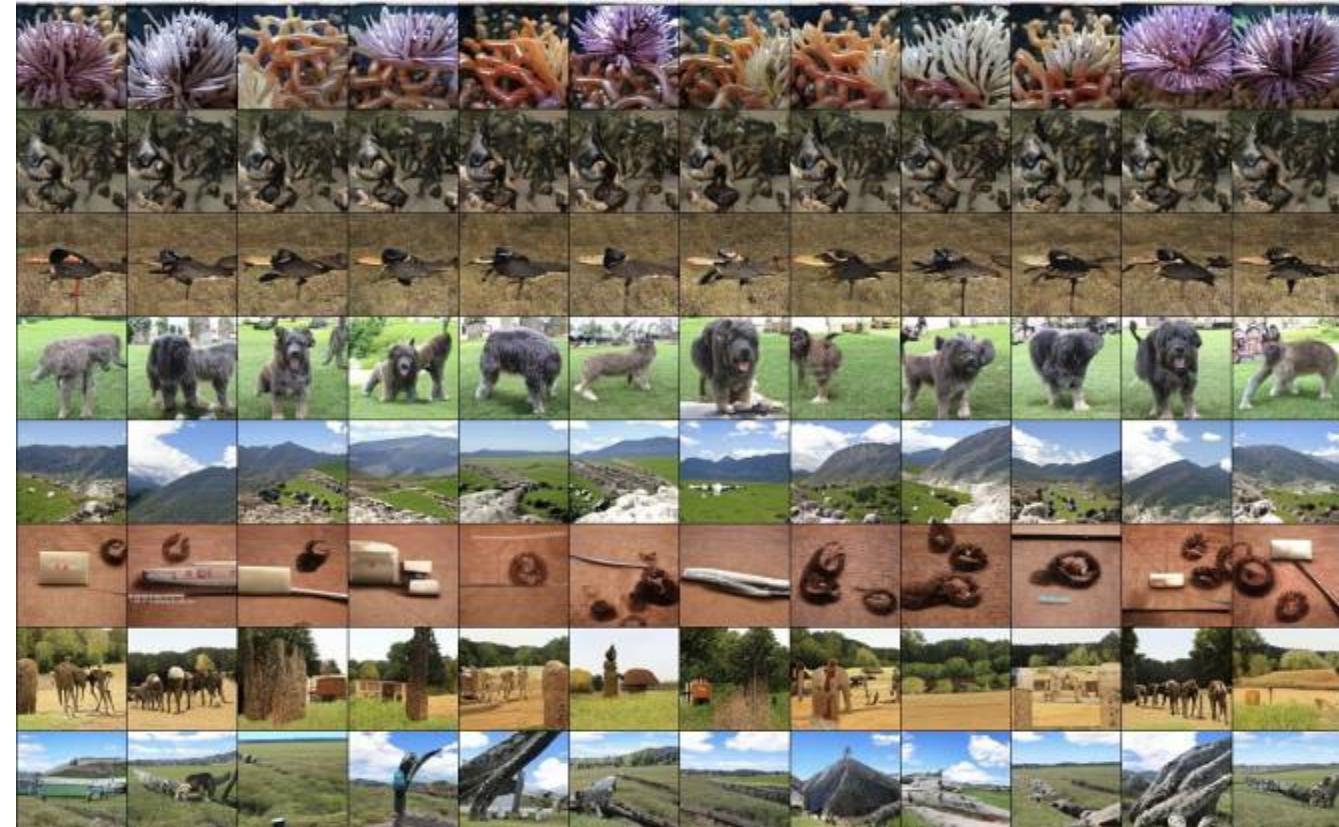
# Unsupervised Category Discovery - BigGAN

Trained with no labels!

```
z = concat([  
    (a)  $[N(0, I)]^{120}$ ,  
    (b) UniformCateg(1024)  
])
```

Each row is one value  
of the categorical (b);  
columns are Gaussian  
samples (a)

Slide: Jeff Donahue



# Unsupervised Category Discovery - BigGAN

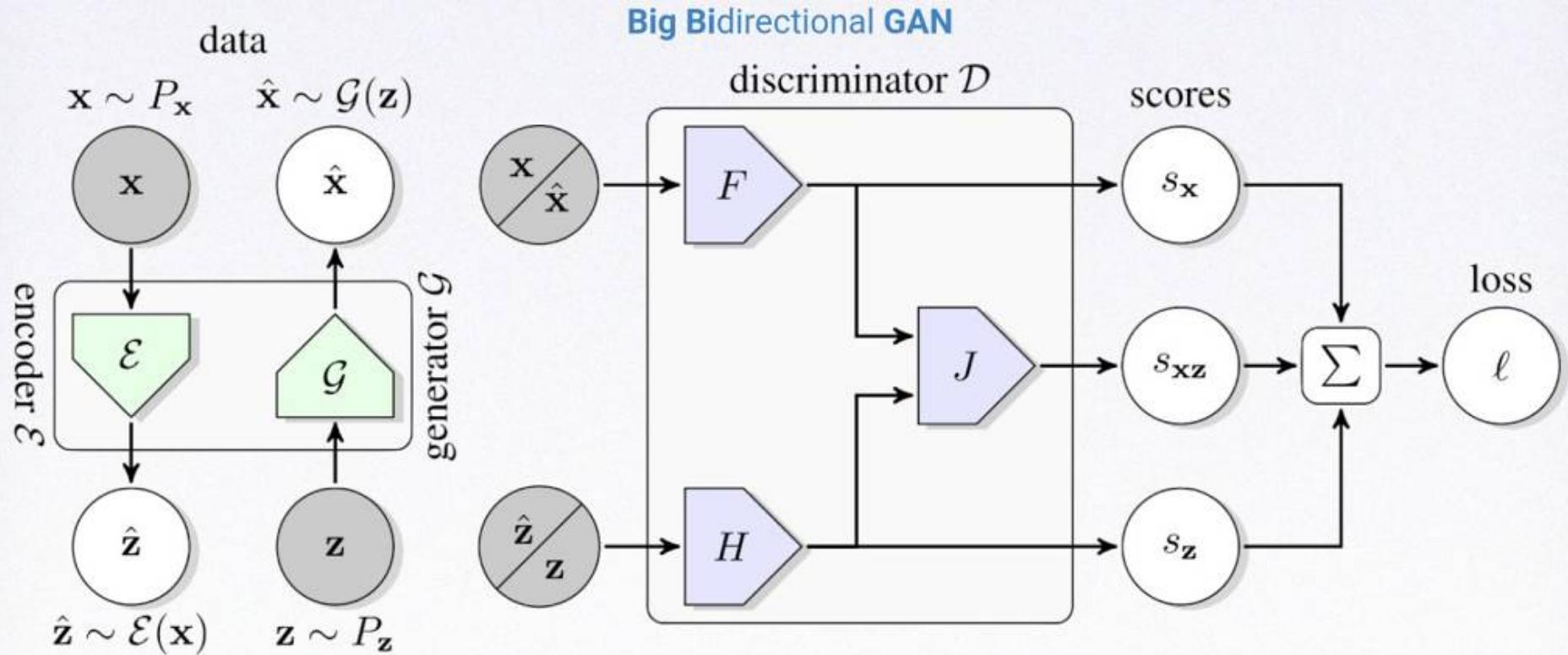
Trained with no labels!

$\mathbf{z} = \text{concat}([$   
    (a)  $[\mathcal{N}(0, I)]^{120}$ ,  
    (b)  $\text{UniformCateg}(1024)$   
])

Each row is one value  
of the categorical (b);  
columns are Gaussian  
samples (a)

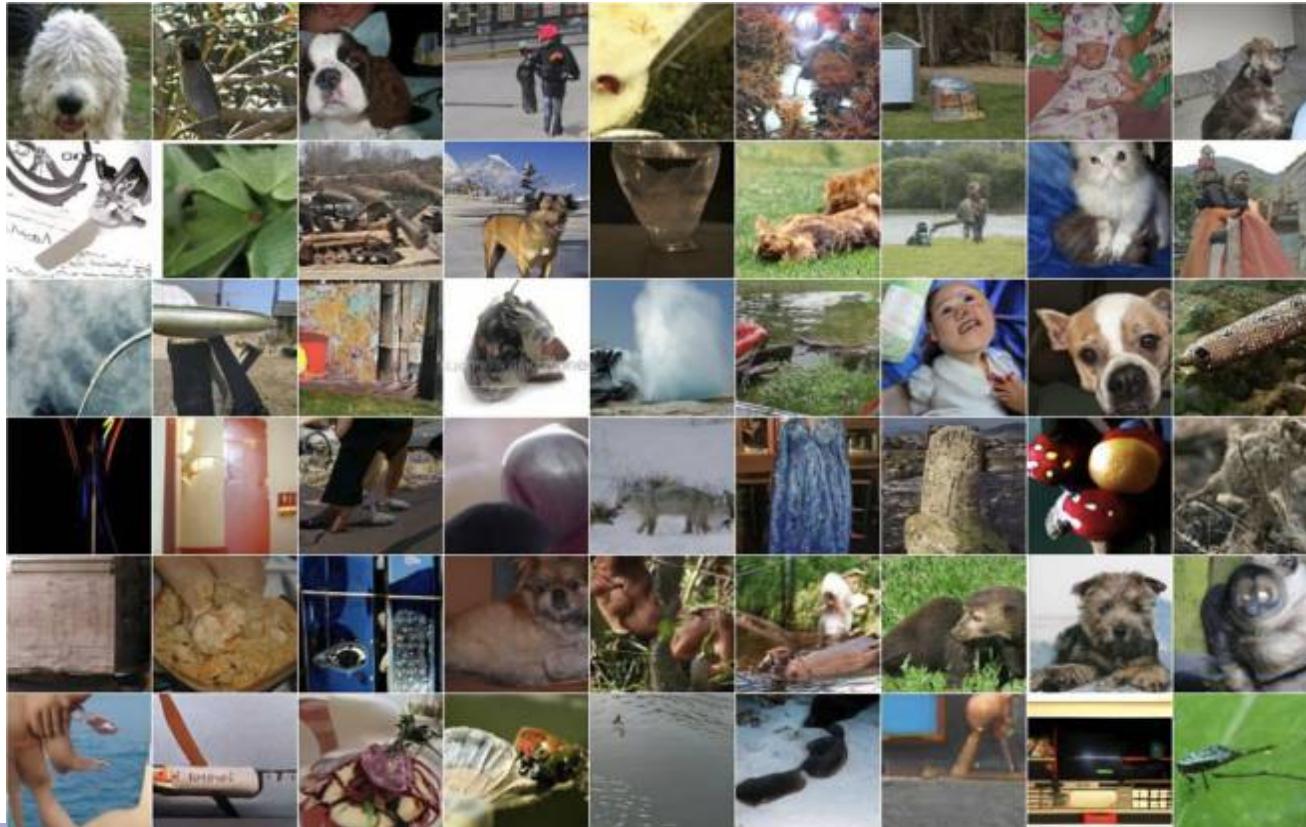


# Big Bidirectional GAN



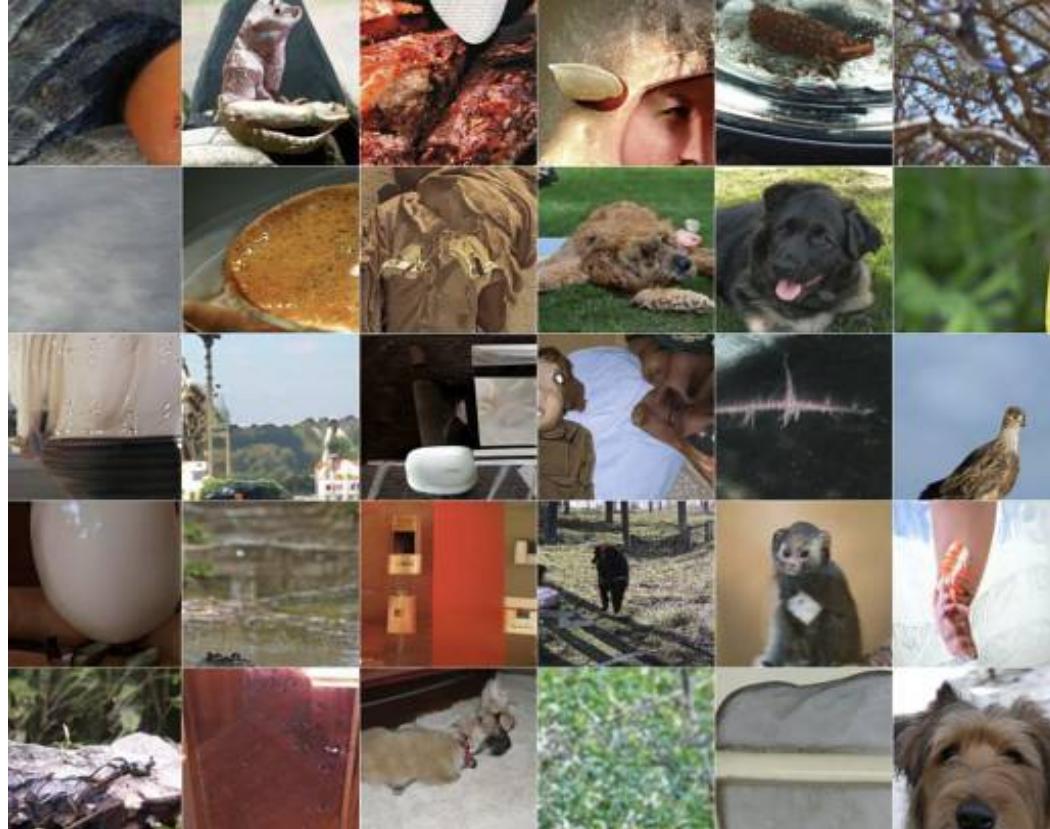
Slide: Jeff Donahue

# BigBiGAN:Unconditional Image Generation



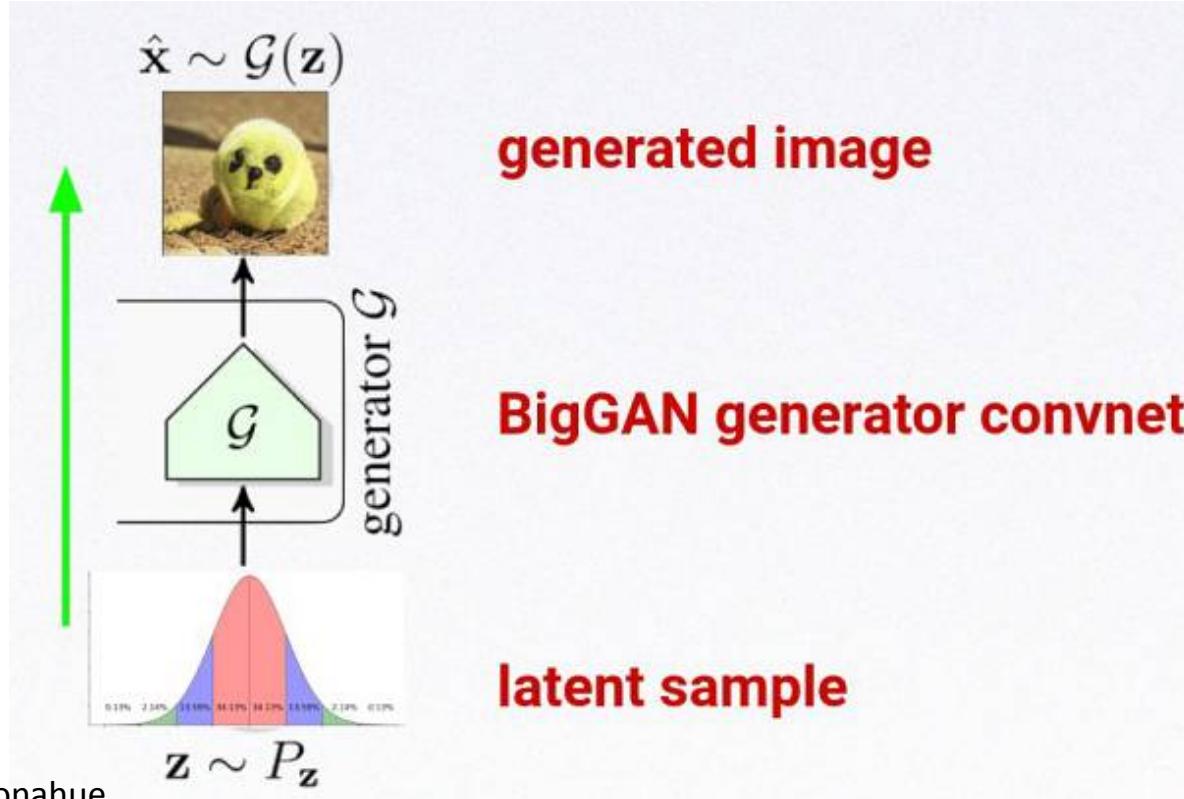
Slide: Jeff Donahue

# BigBiGAN - Unconditional Image Generation



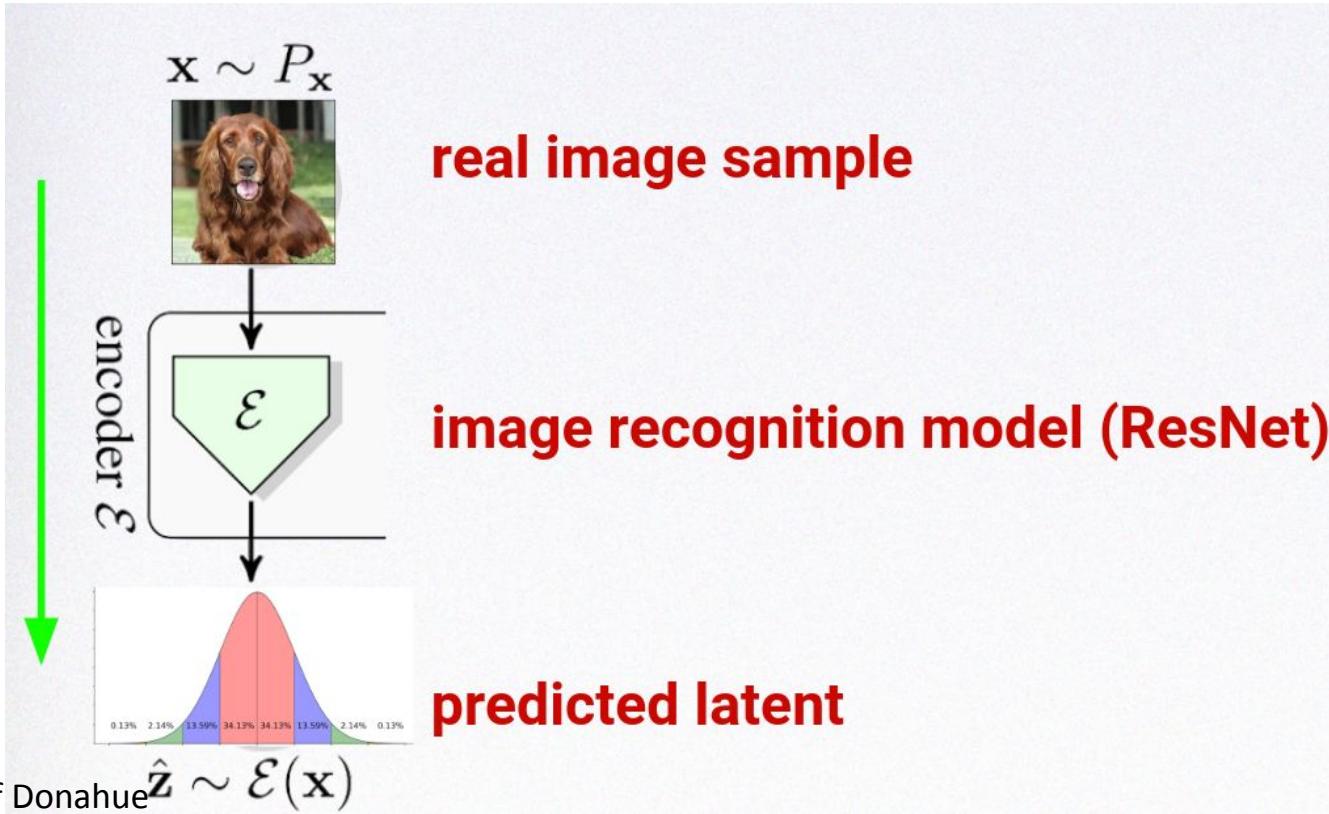
Slide: Jeff Donahue

# BigBiGAN



Slide: Jeff Donahue

# BigBiGAN



# BigBiGAN

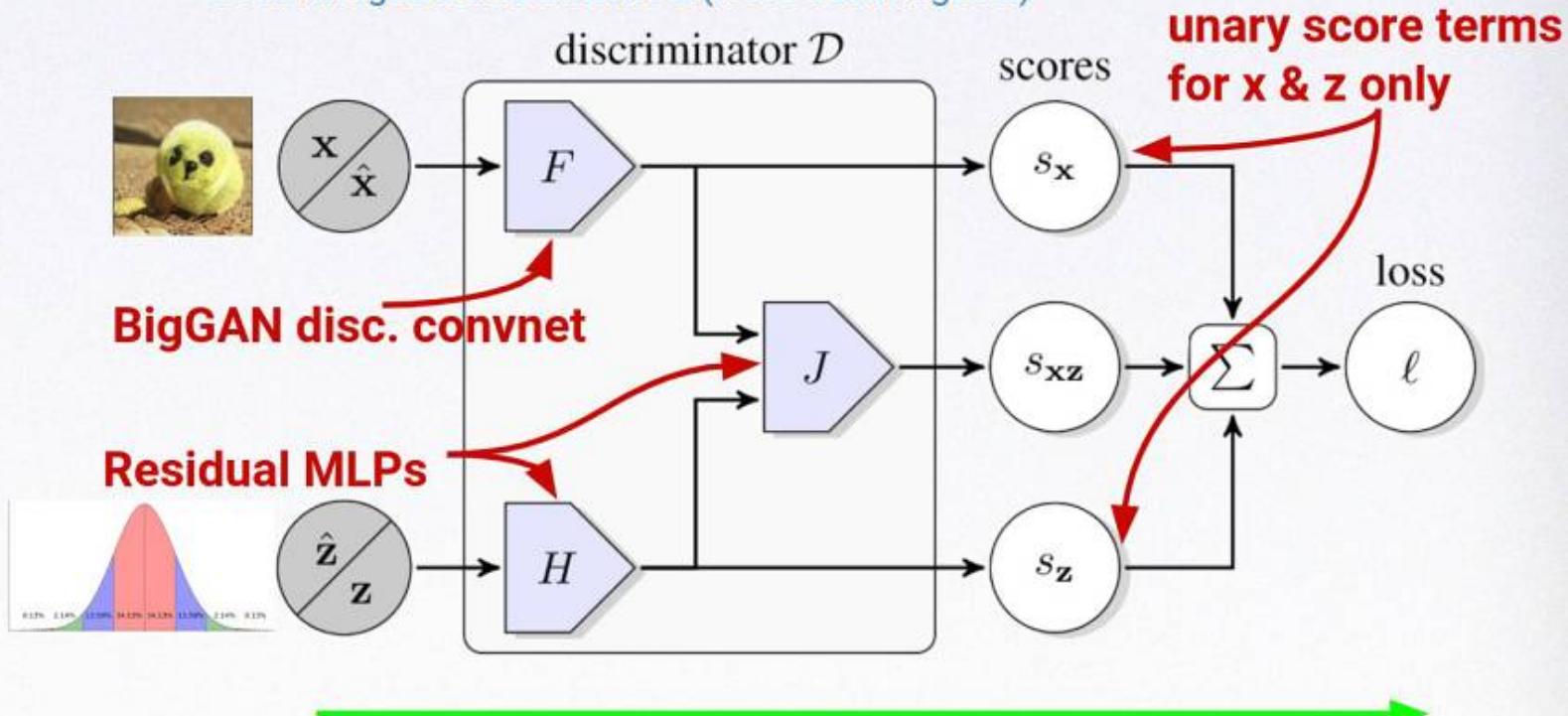
Sees Images  $x$  and Latents  $z$  (Not Just Images  $x$ )

Discriminates between input pairs:

Encoder pair  
 $(x, z' = E(x))$

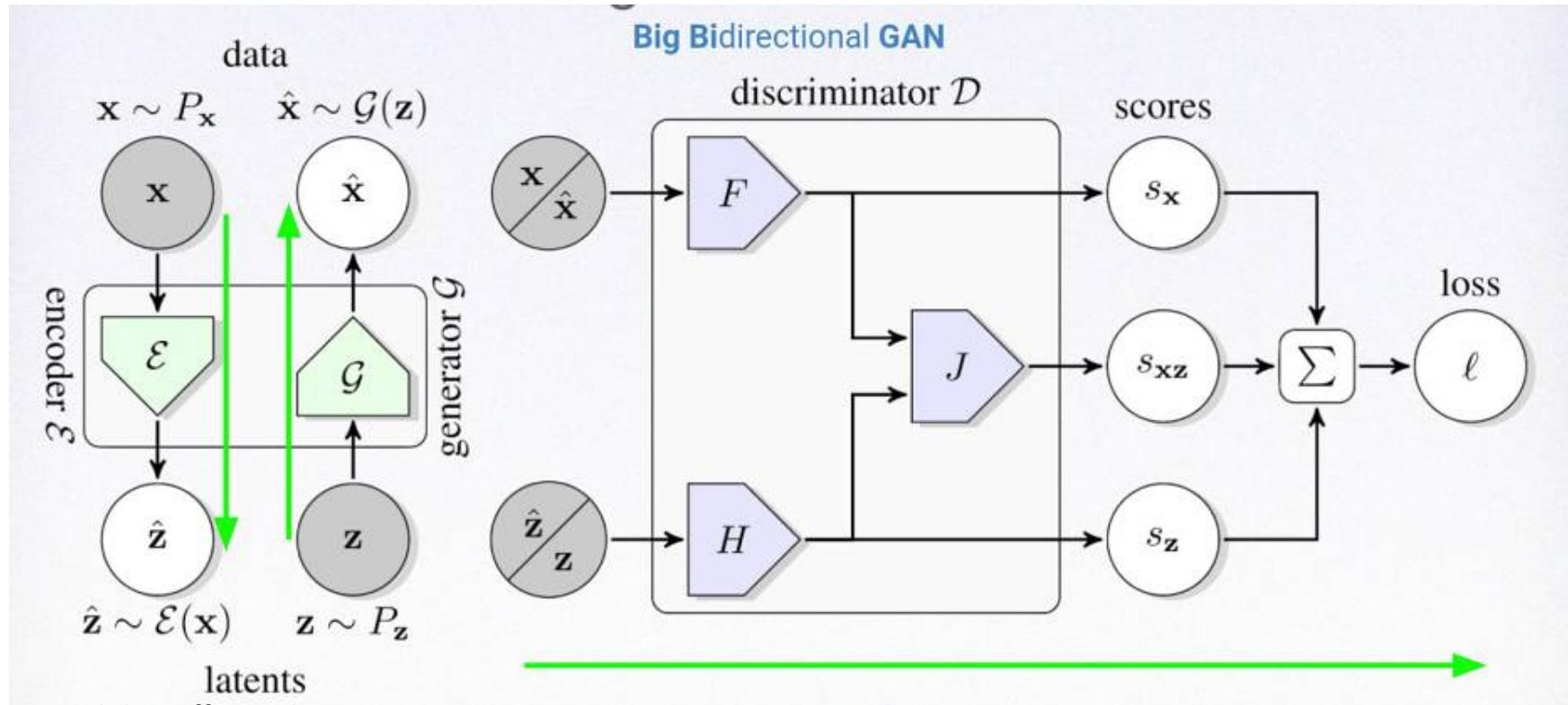
vs.

Generator pair  
 $(x' = G(z), z)$



Slide: Jeff Donahue

# BigBiGAN



Slide: Jeff Donahue

# BigBiGAN: Representation Learning

Method	Architecture	Feature	Top-1	Top-5
BiGAN [7, 42]	AlexNet	Conv3	31.0	-
SS-GAN [4]	ResNet-19	Block6	38.3	-
Motion Segmentation (MS) [30, 6]	ResNet-101	AvePool	27.6	48.3
Exemplar (Ex) [8, 6]	ResNet-101	AvePool	31.5	53.1
Relative Position (RP) [5, 6]	ResNet-101	AvePool	36.2	59.2
Colorization (Col) [41, 6]	ResNet-101	AvePool	39.6	62.5
Combination of MS+Ex+RP+Col [6]	ResNet-101	AvePool	-	69.3
CPC [39]	ResNet-101	AvePool	48.7	73.6
Rotation [11, 24]	RevNet-50 $\times$ 4	AvePool	55.4	-
Efficient CPC [17]	ResNet-170	AvePool	61.0	83.0
BigBiGAN (ours)	ResNet-50	AvePool	55.4	77.4
	ResNet-50	BN+CReLU	56.6	78.6
	RevNet-50 $\times$ 4	AvePool	60.8	81.4
	RevNet-50 $\times$ 4	BN+CReLU	61.3	81.9

Slide: Jeff Donahue

# BigBiGAN: Latent Space NNs



Slide: Jeff Donahue

# BigBiGAN Reconstructions

Computing a reconstruction  $\mathbf{x}' = G(E(\mathbf{x}))$ :

- (1) Sample a real image  $\mathbf{x} \sim P_x$
- (2) Encoder predicts latents  $\mathbf{z}' = E(\mathbf{x})$
- (3) Generator predicts reconstruction  $\mathbf{x}' = G(\mathbf{z}')$

**(Big)BiGAN is not directly trained for reconstruction!**

Arises out of the objective: approx. reconstruction  $\mathbf{x}' \cong G(E(\mathbf{x}))$  optimally confuses the joint data-latent discriminator.

Reconstructions give insight into the semantics modeled.

real images  $\mathbf{x}$



Slide: Jeff Donahue

# BigBiGAN Reconstructions



Slide: Jeff Donahue

# Outline

---

- Motivation & Definition of Implicit Models
- Original GAN (Goodfellow et al, 2014)
- Evaluation: Parzen, Inception, Frechet
- Some Theory: Bayes-optimal Discriminator; Jensen-Shannon Divergence; Mode Collapse; Avoiding Saturation
- GAN Progression:
  - DC GAN (Radford et al, 2016)
  - Improved Training of GANs (Salimans et al, 2016)
  - WGAN, WGAN-GP, Progressive GAN, SN-GAN, SAGAN
  - BigGAN, BigGAN-Deep, StyleGAN, StyleGAN-v2, VIB-GAN
- Creative Conditional GANs
- GANs and Representations
- **GANs as Energy Models**
- GANs and Optimal Transport, Implicit Likelihood Models, Moment Matching
- Other uses of Adversarial Loss: Transfer Learning, Fairness
- GANs and Imitation Learning

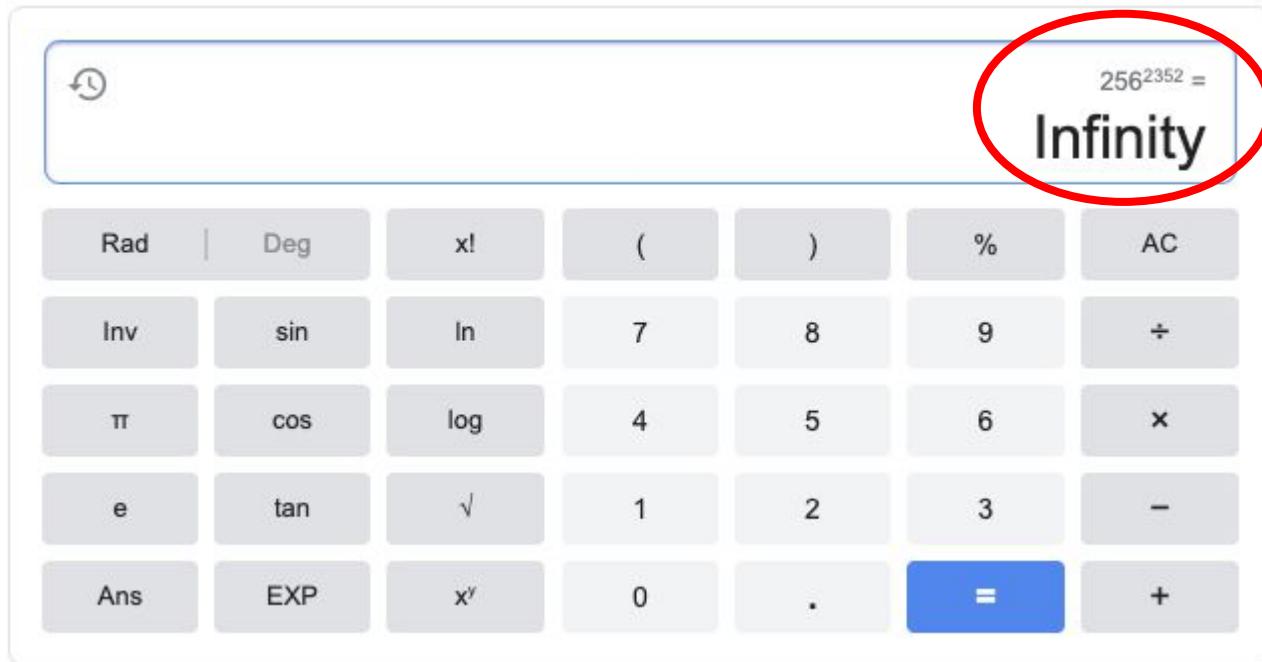
# Energy Models

- Assign energy  $E(x)$  to every possible  $x$

$$p(x) = \frac{1}{Z} e^{-E(x)} \quad Z = \sum_{x \in \mathcal{X}} e^{-E(x)}$$

- Low energy makes for high probability  $x$
- High energy makes for low probability  $x$
- Practical challenge: domain of  $x$  usually very large
  - E.g. all possible 28x28x3 images =  $256^{(28*28*3)} \sim 10^{16464}$

# Energy Models



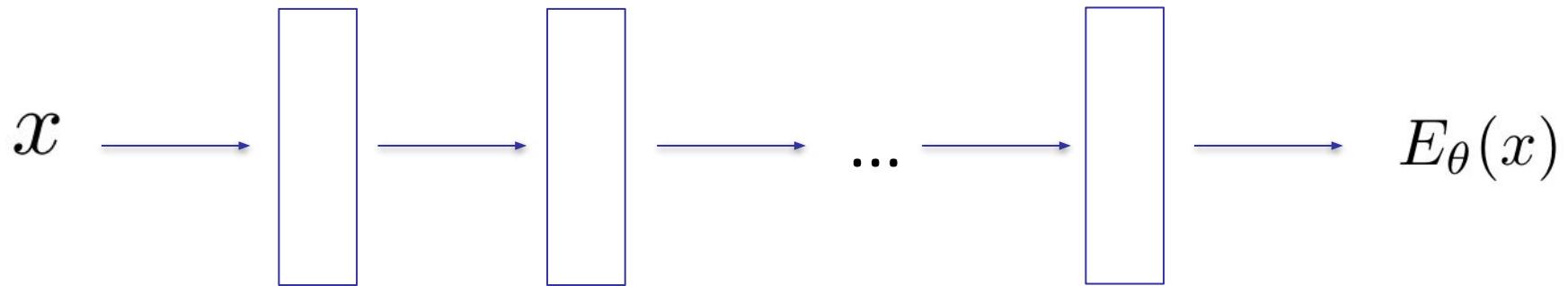
# Energy Models -- Definition

- Assign energy  $E(x)$  to every possible  $x$

$$p(x) = \frac{1}{Z} e^{-E(x)} \quad Z = \sum_{x \in \mathcal{X}} e^{-E(x)}$$

- Low energy makes for high probability  $x$
- High energy makes for low probability  $x$
- Practical challenge: domain of  $x$  usually very large
  - E.g. all possible 28x28x3 images =  $256^{(28*28*3)} \sim 10^{16464}$
  - **Z impractical to compute**

# Energy Models -- Maximum Likelihood



$$\max_{\theta} \mathbb{E}_{x \sim p_{\text{data}}} [\log p_{\theta}(x)] = \max_{\theta} \mathbb{E}_{x \sim p_{\text{data}}} \log \left( \frac{e^{-E_{\theta}(x)}}{Z(\theta)} \right) = \max_{\theta} \mathbb{E}_{x \sim p_{\text{data}}} [-E_{\theta}(x)] - \log Z(\theta)$$

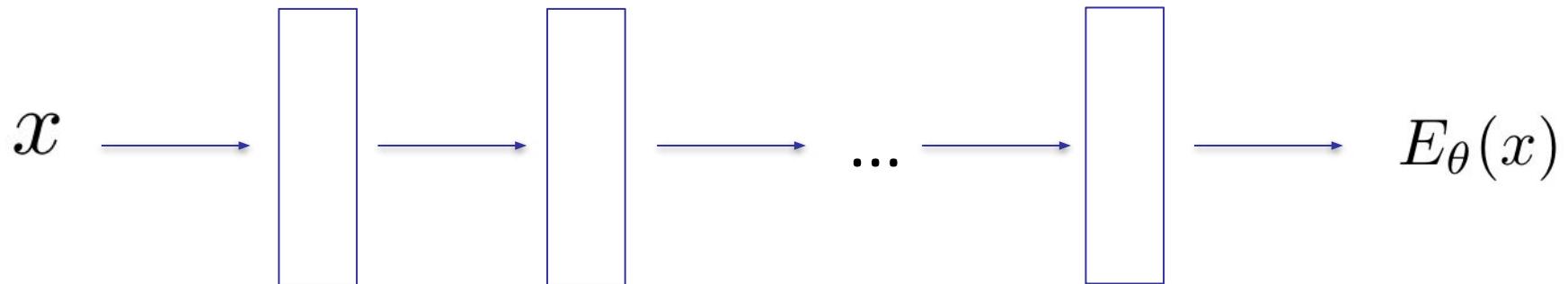
Can we just ignore  $Z$  and maximize first part of the objective?

# Variational Lower Bound for $\log Z$

$$\begin{aligned}\log Z &= \log \sum_{x \in \mathcal{X}} e^{-E_\theta(x)} \\&= \log \sum_{x \in \mathcal{X}} \frac{q_\phi(x)}{q_\phi(x)} e^{-E_\theta(x)} \\&= \log \mathbb{E}_{x \sim q_\phi} \left[ \frac{e^{-E_\theta(x)}}{q_\phi(x)} \right] \\&\geq \max_{q_\phi} \mathbb{E}_{x \sim q_\phi} \left[ \log \frac{e^{-E_\theta(x)}}{q_\phi(x)} \right] \\&= \max_{q_\phi} \mathbb{E}_{x \sim q_\phi} [-E_\theta(x)] + \mathcal{H}(q_\phi)\end{aligned}$$

$$-\log Z \leq \min_{q_\phi} \mathbb{E}_{x \sim q_\phi} [E_\theta(x)] - \mathcal{H}(q_\phi)$$

# Energy Models -- Maximum Likelihood



$$\max_{\theta} \mathbb{E}_{x \sim p_{\text{data}}} [\log p_{\theta}(x)] = \max_{\theta} \mathbb{E}_{x \sim p_{\text{data}}} \log \left( \frac{e^{-E_{\theta}(x)}}{Z(\theta)} \right) = \max_{\theta} \mathbb{E}_{x \sim p_{\text{data}}} [-E_{\theta}(x)] - \log Z(\theta)$$

$$= \max_{\theta} \mathbb{E}_{x \sim p_{\text{data}}} [-E_{\theta}(x)] + \min_{q_{\phi}} \mathbb{E}_{x \sim q_{\phi}} [E_{\theta}(x)] - \mathcal{H}(q_{\phi})$$

= entropy regularized W-GAN objective

$$\min_G \max_{D \in \mathcal{D}} \mathbb{E}_{x \sim P_r} [D(x)] - \mathbb{E}_{\tilde{x} \sim P_g} [D(\tilde{x})]$$

# GANs and Energy Models

$$\max_{\theta} \min_{q_\phi} \mathbb{E}_{x \sim p_{\text{data}}} [-E_\theta(x)] + \mathbb{E}_{x \sim q_\phi} [E_\theta(x)] - \mathcal{H}(q_\phi)$$

Wasserstein GAN

$$\min_G \max_{D \in \mathcal{D}} \mathbb{E}_{x \sim P_r} [D(x)] - \mathbb{E}_{\tilde{x} \sim P_g} [D(\tilde{x})]$$

- Energy Model = W-GAN with Entropy Regularization
  - Discriminator D = - energy E
  - Generator G = q\_phi
- Note: W-GAN also had regularizer (lipschitz) on D (= -E)
- Is Entropy easy to compute?
  - Generally, no... but can be done for some models, worth investigating!
    - E.g. q\_phi auto-regressive or VAE or flow (why not just log-likelihood objective for those?)
    - Could in principle also add entropy regularization to those models when training w/ log-likelihood

# GANs and Energy Models

Inspiring, clean comprehensive mathematical write-up

- John Schulman  
Notes On GANs, Energy-Based Models, and Saddle Points  
<http://joschu.net/docs/gan-notes.pdf> (2016)

Early papers:

- Taesup Kim and Yoshua Bengio  
Deep Directed Generative Models with Energy-Based Probability Estimation  
<https://arxiv.org/abs/1606.03439>
- Junbo Zhao, Michael Mathieu, Yann LeCun  
Energy-based Generative Adversarial Network  
<https://arxiv.org/abs/1609.03126>

Recent related papers:

- Adji B. Dieng, Francisco J. R. Ruiz, David M. Blei, Michalis K. Titsias  
Prescribed Generative Adversarial Networks  
<https://arxiv.org/abs/1910.04302>
- Aditya Grover, Manik Dhar, Stefano Ermon  
Flow-GAN: Combining Maximum Likelihood and Adversarial Learning in Generative Models  
<https://arxiv.org/abs/1705.08868>

# Outline

---

- Motivation & Definition of Implicit Models
- Original GAN (Goodfellow et al, 2014)
- Evaluation: Parzen, Inception, Frechet
- Some Theory: Bayes-optimal Discriminator; Jensen-Shannon Divergence; Mode Collapse; Avoiding Saturation
- GAN Progression:
  - DC GAN (Radford et al, 2016)
  - Improved Training of GANs (Salimans et al, 2016)
  - WGAN, WGAN-GP, Progressive GAN, SN-GAN, SAGAN
  - BigGAN, BigGAN-Deep, StyleGAN, StyleGAN-v2, VIB-GAN
- Creative Conditional GANs
- GANs and Representations
- GANs as Energy Models
- ***GANs and Optimal Transport, Implicit Likelihood Models, Moment Matching***
- Other uses of Adversarial Loss: Transfer Learning, Fairness
- GANs and Imitation Learning

# Recall: Wasserstein Distance

- Earth Mover (EM) distance

$$W(P_r, P_g) = \inf_{\gamma \in \Pi(P_r, P_g)} \mathbb{E}_{(x,y) \sim \gamma} [| | x - y | |]$$

- W-GAN optimizes the dual
- How about optimizing the primal directly?
  - Optimal Transport GANs

# GANs and Optimal Transport

---

- Improving GANs using Optimal Transport

Tim Salimans, Han Zhang, Alec Radford, Dimitris Metaxas

ICLR 2018

<https://arxiv.org/abs/1803.05573>

- Sinkhorn AutoDiff GAN: Learning Generative Models with Sinkhorn Divergences

Aude Genevay, Gabriel Peyre, Marco Cuturi

AISTATS 2018

<https://arxiv.org/abs/1706.00292>

- Cramer GAN: The Cramer Distance as a Solution to Biased Wasserstein Gradients

Marc Bellemare, Ivo Danihelka, W Dabney, S Mohamed, B Laskhiminarayanan, S Hoyer, R Munos

<https://arxiv.org/abs/1705.10743>

# Implicit Likelihood Models

---

**Algorithm 1** Implicit maximum likelihood estimation (IMLE) procedure

---

**Require:** The dataset  $D = \{\mathbf{x}_i\}_{i=1}^n$  and a sampling mechanism for the implicit model  $P_\theta$

Initialize  $\theta$  to a random vector

**for**  $k = 1$  **to**  $K$  **do**

- Draw i.i.d. samples  $\tilde{\mathbf{x}}_1^\theta, \dots, \tilde{\mathbf{x}}_m^\theta$  from  $P_\theta$
- Pick a random batch  $S \subseteq \{1, \dots, n\}$
- $\sigma(i) \leftarrow \arg \min_j \|\mathbf{x}_i - \tilde{\mathbf{x}}_j^\theta\|_2^2 \quad \forall i \in S$
- for**  $l = 1$  **to**  $L$  **do**

  - Pick a random mini-batch  $\tilde{S} \subseteq S$
  - $\theta \leftarrow \theta - \eta \nabla_\theta \left( \frac{n}{|\tilde{S}|} \sum_{i \in \tilde{S}} \|\mathbf{x}_i - \tilde{\mathbf{x}}_{\sigma(i)}^\theta\|_2^2 \right)$

- end for**

**end for**

**return**  $\theta$

---

→ every training data point needs to have a closeby generated neighbor

- Implicit Maximum Likelihood Estimation  
Ke Li, Jitendra Malik  
<https://arxiv.org/abs/1809.09087>
- Diverse Image Synthesis from Semantic Layouts via Conditional IMLE  
Ke Li, Tianhao Zhang, Jitendra Malik  
<https://arxiv.org/abs/1811.12373>

# Moment Matching

- Key idea: Match the moments of the data and model distributions to bring them closer
- Called the two-sample test in hypothesis testing

Given samples  $X = \{x_i\}_{i=1}^N$  from  $P_X$ ,  $Y = \{y_j\}_{j=1}^M$  from  $P_Y$ , how do we compare  $P_X$  and  $P_Y$ ?

- Not feasible to compute higher order moments in high dimensions

# Moment Matching

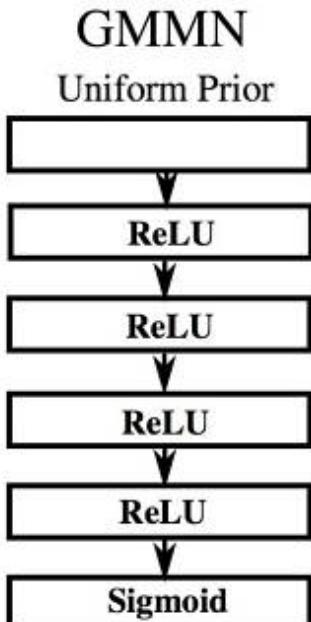
- Kernel trick

$$\begin{aligned} L_{MMD} &= \left\| \frac{1}{N} \sum_{i=1}^N \phi(x_i) - \frac{1}{M} \sum_{j=1}^M \phi(y_j) \right\|^2 \\ &= \frac{1}{N^2} \sum_{i=1}^N \sum_{i'=1}^N \phi(x_i)^T \phi(x_{i'}) + \frac{1}{M^2} \sum_{j=1}^M \sum_{j'=1}^M \phi(y_j)^T \phi(y_{j'}) - \frac{2}{MN} \sum_{i=1}^N \sum_{j=1}^M \phi(x_i)^T \phi(y_j) \\ &= \frac{1}{N^2} \sum_{i=1}^N \sum_{i'=1}^N k(x_i, x_{i'}) - \frac{2}{MN} \sum_{i=1}^N \sum_{j=1}^M k(x_i, y_j) + \frac{1}{M^2} \sum_{j=1}^M \sum_{j'=1}^M k(y_j, y_{j'}) \end{aligned}$$

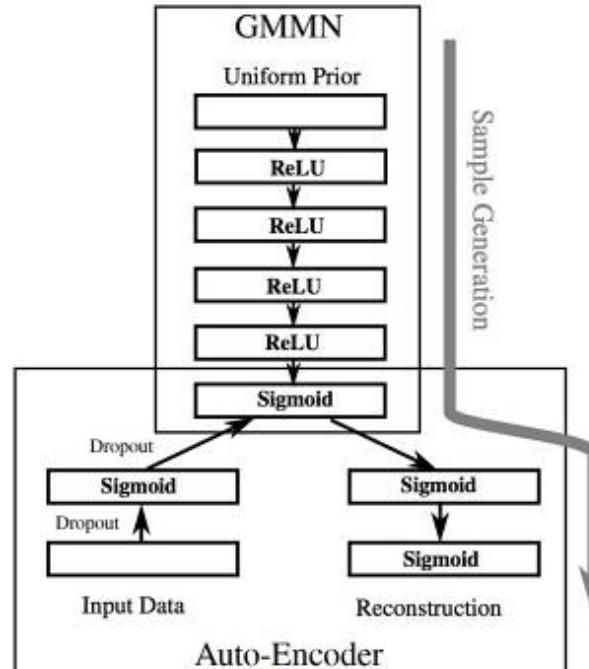
# Moment Matching

- Identity function  $\phi$  is equivalent to matching sample means
- Kernel trick allows you to optimize mean discrepancy in higher dimensional spaces.
- Example  $k(x, x') = \exp\left(\frac{-||x - x'||^2}{2\sigma}\right)$
- Refer to Gretton et al 2007, 2012 - Mathematical Treatment

# Generative Moment Matching Networks



(a) GMMN



(b) GMMN+AE

Li, Swersky, Zemel  
(2015)

# Generative Moment Matching Networks

**Input** : Dataset  $\{\mathbf{x}_1^d, \dots, \mathbf{x}_N^d\}$ , prior  $p(\mathbf{h})$ , network  $f(\mathbf{h}; \mathbf{w})$  with initial parameter  $\mathbf{w}^{(0)}$

**Output:** Learned parameter  $\mathbf{w}^*$

```
1 while Stopping criterion not met do
2     Get a minibatch of data  $\mathbf{X}^d \leftarrow \{\mathbf{x}_{i_1}^d, \dots, \mathbf{x}_{i_b}^d\}$ 
3     Get a new set of samples  $\mathbf{X}^s \leftarrow \{\mathbf{x}_1^s, \dots, \mathbf{x}_b^s\}$ 
4     Compute gradient  $\frac{\partial \mathcal{L}_{\text{MMD}}}{\partial \mathbf{w}}$  on  $\mathbf{X}^d$  and  $\mathbf{X}^s$ 
5     Take a gradient step to update  $\mathbf{w}$ 
6 end
```

Li, Swersky, Zemel  
(2015)

1	7	1	5	3	1
4	8	6	9	9	8
2	5	9	4	8	2
7	3	4	5	1	7
5	1	2	9	8	2

(a) GMMN MNIST samples



(b) GMMN TFD samples

4	5	6	4	0	0
1	9	4	6	1	2
6	6	5	0	4	5
7	2	9	5	0	6
2	3	1	2	6	5

(c) GMMN+AE MNIST samples



(d) GMMN+AE TFD samples

# Generative Moment Matching Networks

- Need a good kernel for the mean discrepancy measure

For universal kernels like the Gaussian kernel, defined as  $k(x, x') = \exp(-\frac{1}{2\sigma}|x - x'|^2)$ , where  $\sigma$  is the bandwidth parameter, we can use a Taylor expansion to get an explicit feature map  $\phi$  that contains an infinite number of terms and covers all orders of statistics. Minimizing MMD under this feature expansion is then equivalent to minimizing a distance between *all* moments of the two distributions.
- Not shown to scale well beyond MNIST and TFD (and some variants on CIFAR 10 later) - needs autoencoding, large minibatch and mixture of kernels with different bandwidths

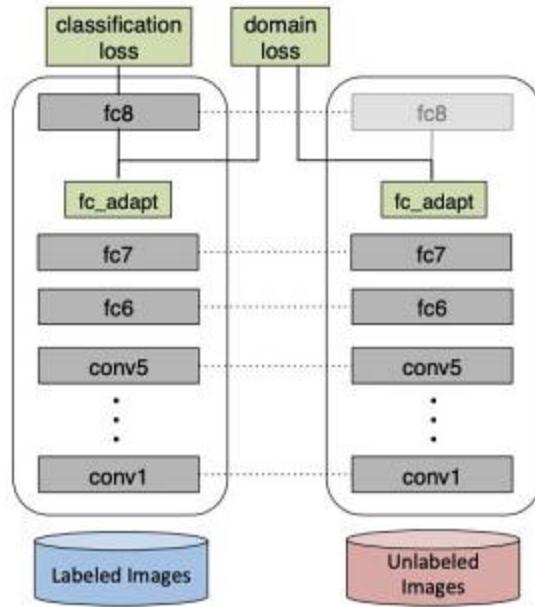
# Outline

---

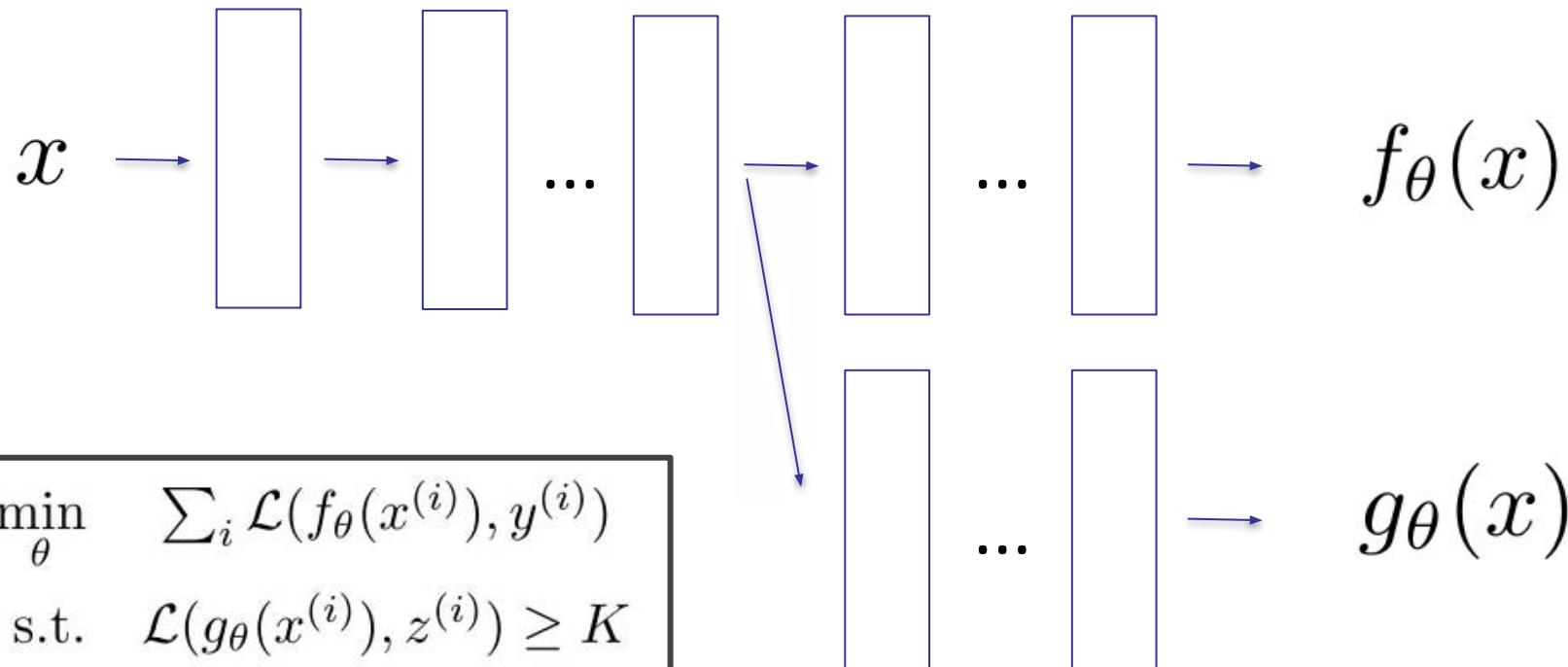
- Motivation & Definition of Implicit Models
- Original GAN (Goodfellow et al, 2014)
- Evaluation: Parzen, Inception, Frechet
- Some Theory: Bayes-optimal Discriminator; Jensen-Shannon Divergence; Mode Collapse; Avoiding Saturation
- GAN Progression:
  - DC GAN (Radford et al, 2016)
  - Improved Training of GANs (Salimans et al, 2016)
  - WGAN, WGAN-GP, Progressive GAN, SN-GAN, SAGAN
  - BigGAN, BigGAN-Deep, StyleGAN, StyleGAN-v2, VIB-GAN
- Creative Conditional GANs
- GANs and Representations
- GANs as Energy Models
- GANs and Optimal Transport, Implicit Likelihood Models, Moment Matching
- **Other uses of Adversarial Loss: Transfer Learning, Fairness**
- GANs and Imitation Learning

# Other Uses of Adversarial Loss -- Transfer

- Deep Domain Confusion: Maximizing fc  
Eric Tzeng, Judy Hoffman, Ning Zhang, et al.  
<https://arxiv.org/abs/1412.3474>



# Other Uses of Adversarial Loss -- Fairness



# Outline

---

- Motivation & Definition of Implicit Models
- Original GAN (Goodfellow et al, 2014)
- Evaluation: Parzen, Inception, Frechet
- Some Theory: Bayes-optimal Discriminator; Jensen-Shannon Divergence; Mode Collapse; Avoiding Saturation
- GAN Progression:
  - DC GAN (Radford et al, 2016)
  - Improved Training of GANs (Salimans et al, 2016)
  - WGAN, WGAN-GP, Progressive GAN, SN-GAN, SAGAN
  - BigGAN, BigGAN-Deep, StyleGAN, StyleGAN-v2, VIB-GAN
- Creative Conditional GANs
- GANs and Representations
- GANs as Energy Models
- GANs and Optimal Transport, Implicit Likelihood Models, Moment Matching
- Other uses of Adversarial Loss: Transfer Learning, Fairness
- **GANs and Imitation Learning**

# GANs and Imitation Learning: GAIL

Imitation learning as a GAN problem:

$$\mathbb{E}_\pi[\log(D(s, a))] + \mathbb{E}_{\pi_E}[\log(1 - D(s, a))] - \lambda H(\pi)$$

- Discriminator tries to distinguish trajectories (s,a) from demonstrator vs. from learned imitation policy  $\pi$
- Learned policy  $\pi$  tries to make itself indistinguishable from demonstrator
- Note: matches Energy-Based Model GAN formulation

GAIL: Generative Adversarial Imitation Learning  
Jonathan Ho, Stefano Ermon  
NeurIPS 2016  
<https://arxiv.org/pdf/1606.03476.pdf>

# GANs and Imitation Learning: GAIL

---

**Algorithm 1** Generative adversarial imitation learning

---

- 1: **Input:** Expert trajectories  $\tau_E \sim \pi_E$ , initial policy and discriminator parameters  $\theta_0, w_0$
- 2: **for**  $i = 0, 1, 2, \dots$  **do**
- 3:   Sample trajectories  $\tau_i \sim \pi_{\theta_i}$
- 4:   Update the discriminator parameters from  $w_i$  to  $w_{i+1}$  with the gradient

$$\hat{\mathbb{E}}_{\tau_i} [\nabla_w \log(D_w(s, a))] + \hat{\mathbb{E}}_{\tau_E} [\nabla_w \log(1 - D_w(s, a))] \quad (17)$$

- 5:   Take a policy step from  $\theta_i$  to  $\theta_{i+1}$ , using the TRPO rule with cost function  $\log(D_{w_{i+1}}(s, a))$ . Specifically, take a KL-constrained natural gradient step with

$$\begin{aligned} & \hat{\mathbb{E}}_{\tau_i} [\nabla_\theta \log \pi_\theta(a|s) Q(s, a)] - \lambda \nabla_\theta H(\pi_\theta), \\ & \text{where } Q(\bar{s}, \bar{a}) = \hat{\mathbb{E}}_{\tau_i} [\log(D_{w_{i+1}}(s, a)) \mid s_0 = \bar{s}, a_0 = \bar{a}] \end{aligned} \quad (18)$$

- 6: **end for**
-

# GANs and Imitation Learning: GAIL

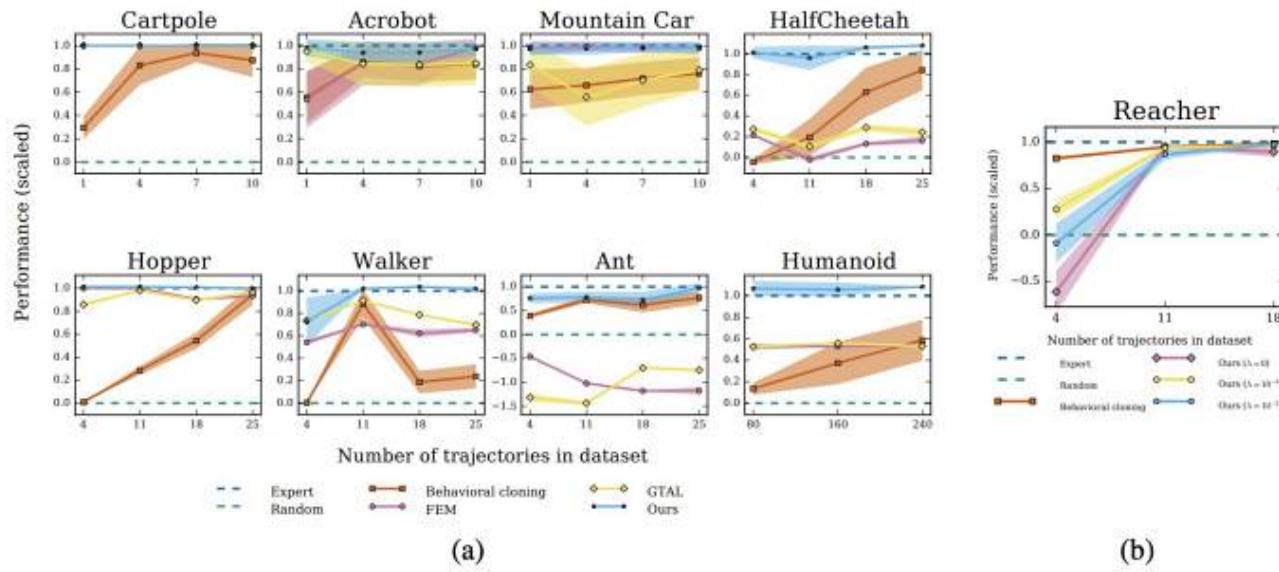
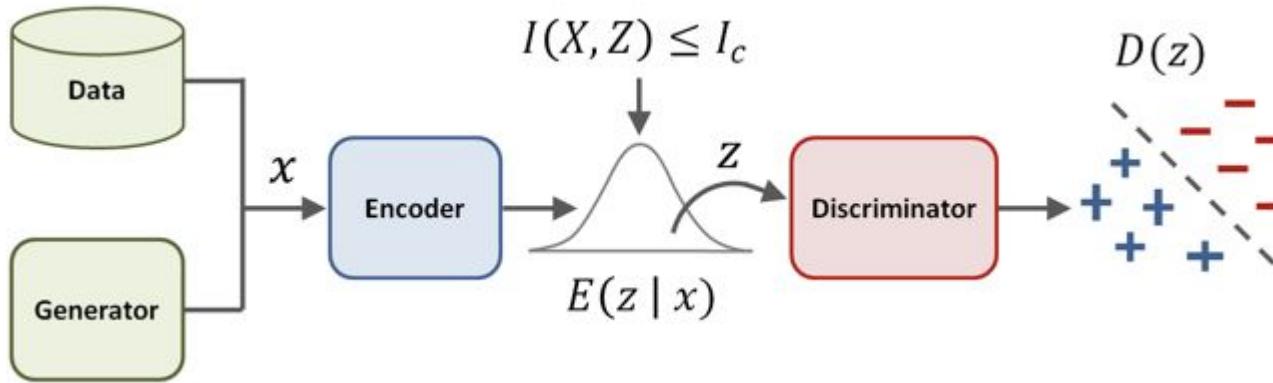


Figure 1: (a) Performance of learned policies. The  $y$ -axis is negative cost, scaled so that the expert achieves 1 and a random policy achieves 0. (b) Causal entropy regularization  $\lambda$  on Reacher.

GAIL: Generative Adversarial Imitation Learning  
Jonathan Ho, Stefano Ermon  
NeurIPS 2016  
<https://arxiv.org/pdf/1606.03476.pdf>

# GANs and Imitation Learning: VAIL

- Recall: Variational Discriminator Bottleneck GAN



Variational Discriminator Bottleneck: Improving Imitation Learning, Inverse RL, and GANs by Constraining Information Flow

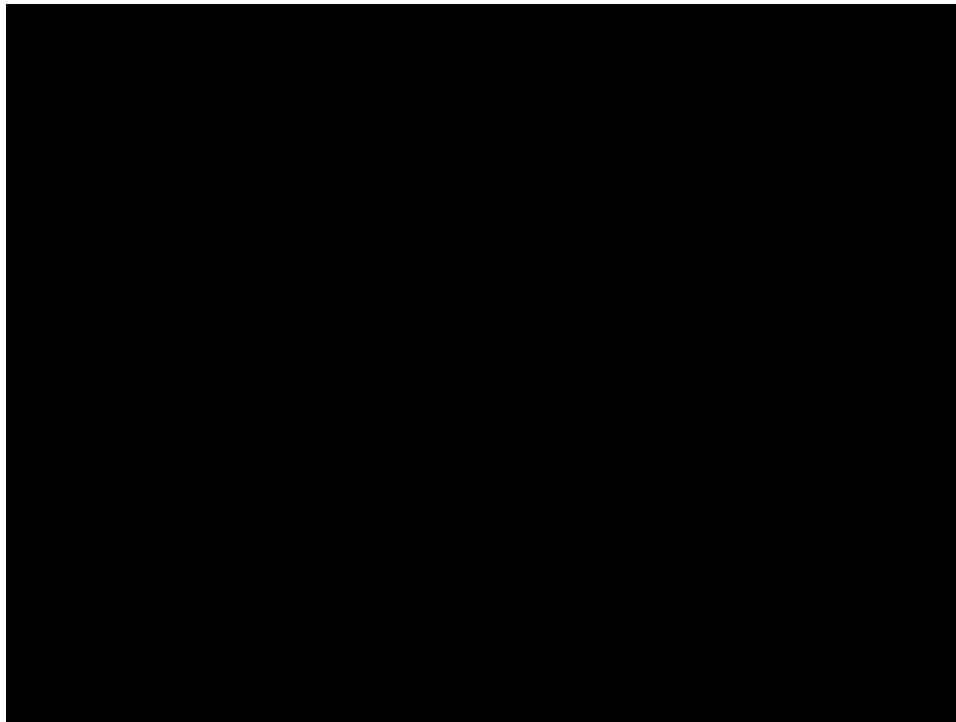
Xue Bin Peng, Angjoo Kanazawa, Sam Toyer, Pieter Abbeel, Sergey Levine

ICLR 2019

<https://arxiv.org/pdf/1810.00821.pdf>

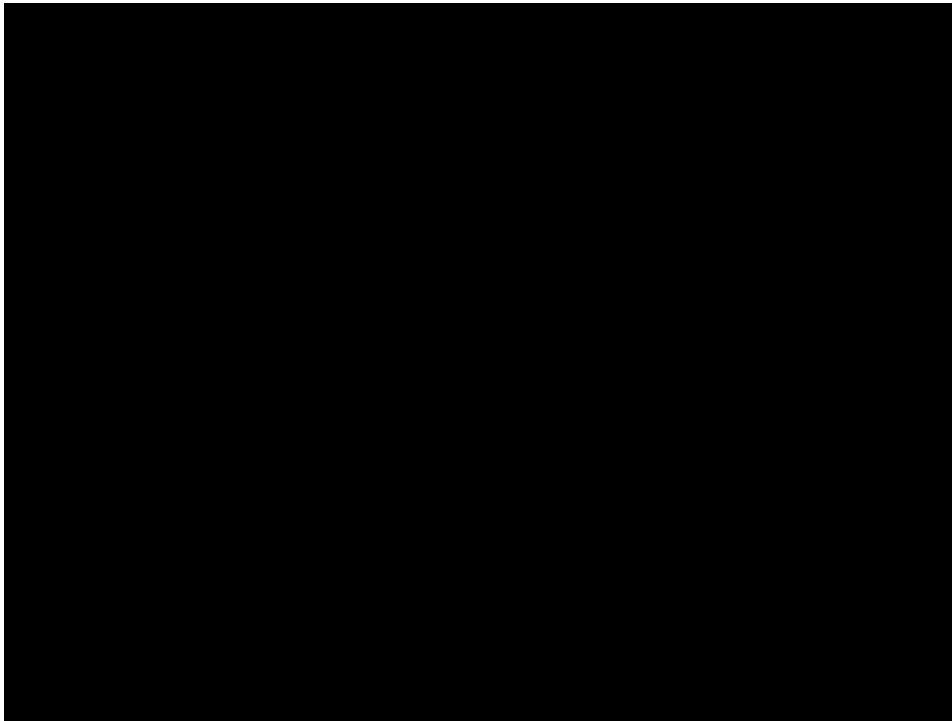
# VAIL

---

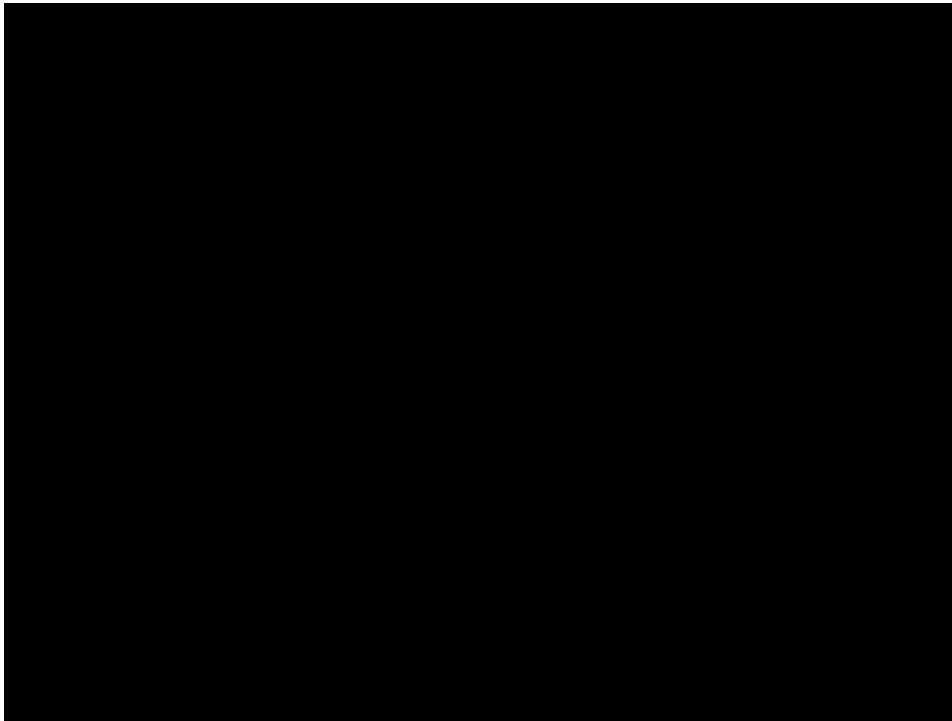


# VAIL

---

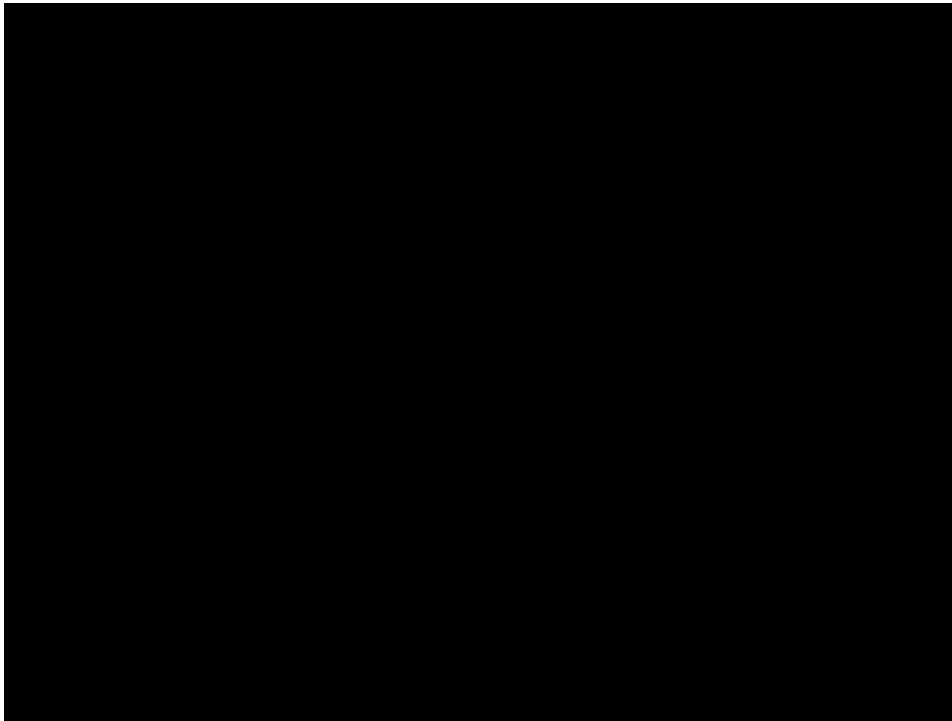


# VAIL

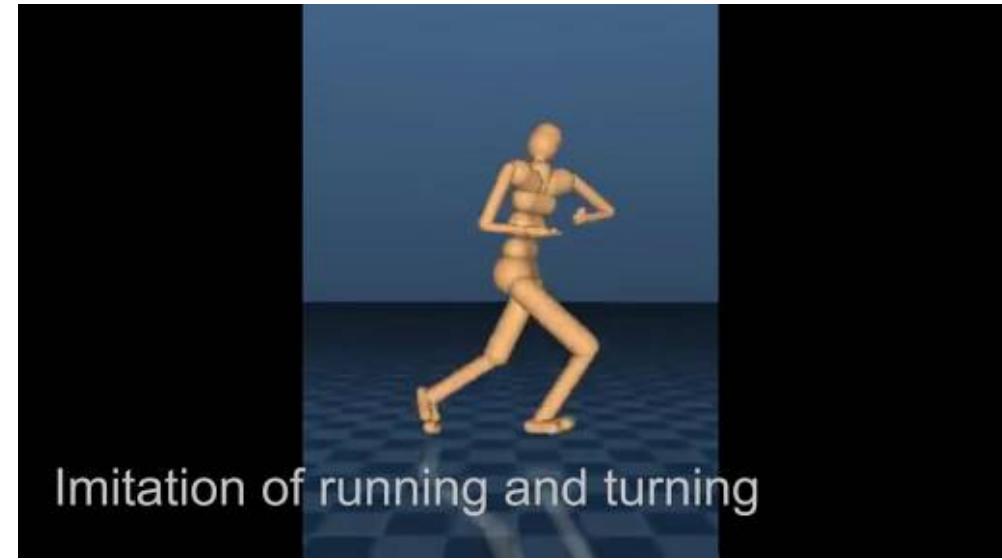


# VAIL

---



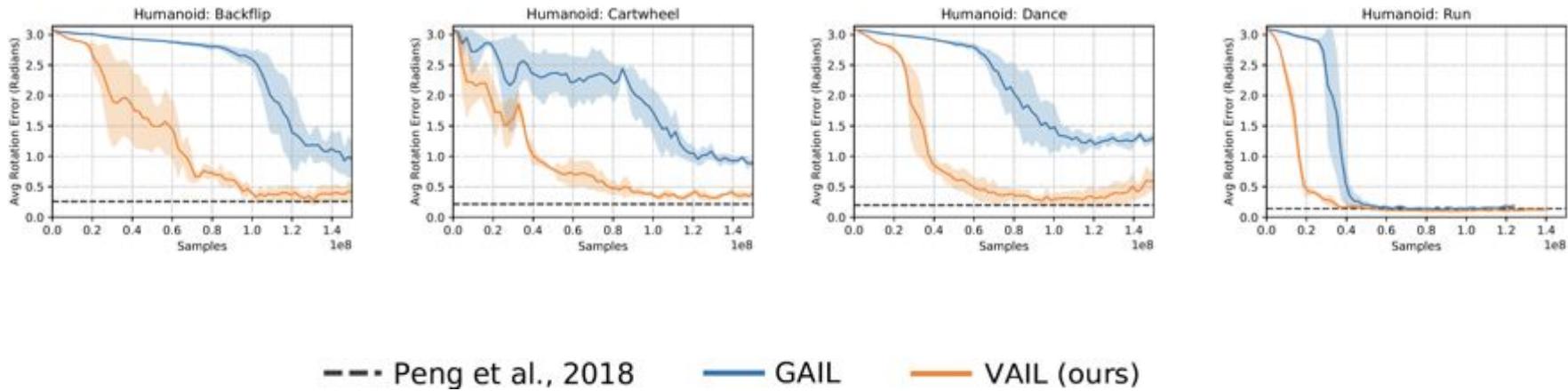
# VAIL



VAIL

[Merel et al 2017]

# VAIL



Variational Discriminator Bottleneck: Improving Imitation Learning, Inverse RL, and GANs by Constraining Information Flow

Xue Bin Peng, Angjoo Kanazawa, Sam Toyer, Pieter Abbeel, Sergey Levine

ICLR 2019

<https://arxiv.org/pdf/1810.00821.pdf>

# Summary

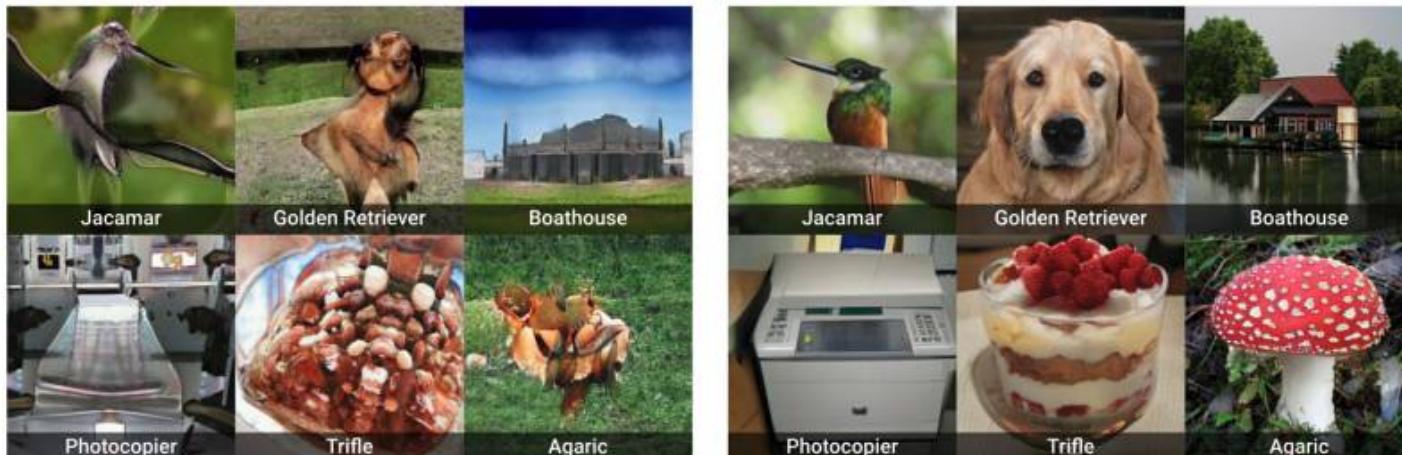
---

- Motivation & Definition of Implicit Models
- Original GAN (Goodfellow et al, 2014)
- Evaluation: Parzen, Inception, Frechet
- Some Theory: Bayes-optimal Discriminator; Jensen-Shannon Divergence; Mode Collapse; Avoiding Saturation
- GAN Progression:
  - DC GAN (Radford et al, 2016)
  - Improved Training of GANs (Salimans et al, 2016)
  - WGAN, WGAN-GP, Progressive GAN, SN-GAN, SAGAN
  - BigGAN, StyleGAN
  - More GANs: BigGAN-Deep, StyleGAN-v2, VIB-GAN, LOGAN
- Creative Conditional GANs
- GANs and Representations
- GANs as Energy Models
- GANs and Optimal Transport
- Implicit Likelihood Models
- Moment Matching
- Other uses of Adversarial Loss: Transfer Learning, Fairness
- GANs and Imitation Learning
- **Scaling up: GigaGAN, StyleGAN-XL**
- GAN losses for improved quantized tokenization: VQGAN, ViT-VQGAN

# StyleGAN-XL

StyleGAN struggles when scaling to large diverse datasets and higher resolutions (ie ImageNet)

First to generate quality images at 1024x1024x scale



StyleGAN3

StyleGAN-XL

# StyleGAN-XL

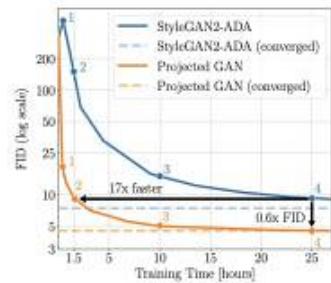
---

- StyleGAN3 base
- Leverage Projected GAN ideas, which enabled more stable training
- Design a progressive growing strategy
- Leverage classifier guidance for additional class conditioning
- Enables 3x parameters of StyleGAN3
- Matches prior SOTA

# StyleGAN-XL

## Projected GANs Converge Faster

Axel Sauer<sup>1,2</sup> Kashyap Chitta<sup>1,2</sup> Jens Müller<sup>3</sup> Andreas Geiger<sup>1,2</sup>  
<sup>1</sup>University of Tübingen   <sup>2</sup>Max Planck Institute for Intelligent Systems, Tübingen  
<sup>3</sup>Computer Vision and Learning Lab, University Heidelberg  
2{firstname.lastname}@tue.mpg.de    3{firstname.lastname}@iwr.uni-heidelberg.de



$$\min_G \max_{\{D_l\}} \sum_{l \in \mathcal{L}} \left( \mathbb{E}_x [\log D_l(P_l(x))] + \mathbb{E}_z [\log(1 - D_l(P_l(G(z))))] \right)$$

## Projected GAN does not work well with StyleGAN

- Reducing latent space helps 512-64 -> enables lower FID and stable training

# StyleGAN-XL

- Pretrained class embeddings
  - Extract mean of lowest resolution feature per class from an Efficientnet-lite0
- When growing, remove 2 layers, then add 7
  - Reduce aliasing
  - Discriminator fixed (upsampled to match the max resolution)
  - Lower resolution layers are fixed

	<b>Configuration</b>	<b>FID ↓</b>	<b>IS ↑</b>
<b>A</b>	StyleGAN3	53.57	15.30
<b>B</b>	+ Projected GAN & small $z$	22.98	57.62
<b>C</b>	+ Pretrained embeddings	20.91	35.79
<b>D</b>	+ Progressive growing	19.51	35.74
<b>E</b>	+ ViT & CNN as $F_{1,2}$	12.43	56.72
<b>F</b>	+ CLF guidance (StyleGAN-XL)	<b>12.24</b>	<b>86.21</b>

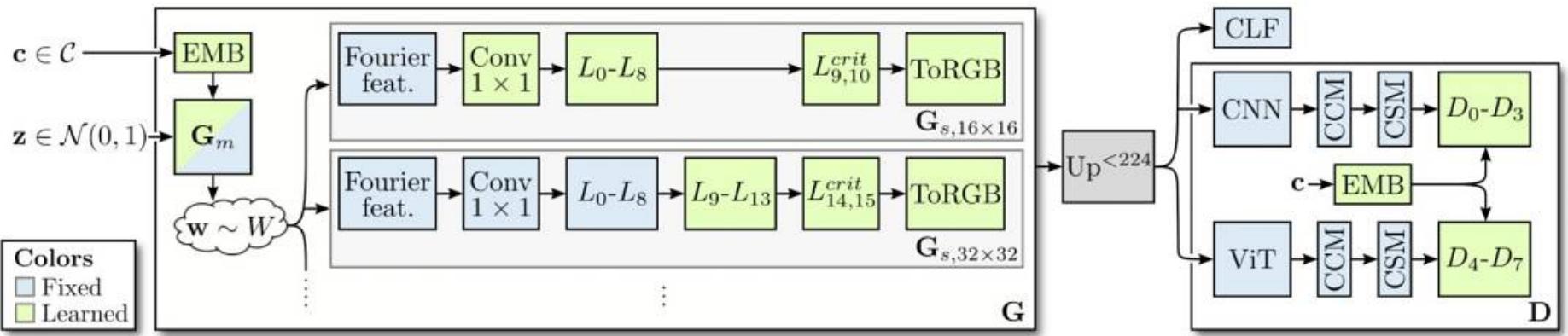
# StyleGAN-XL

- Pretrained class embeddings
  - Extract mean of lowest resolution feature per class from an Efficientnet-lite0
- When growing, remove 2 layers, then add 7
  - Reduce aliasing
  - Discriminator fixed (upsampled to match the max resolution)
  - Lower resolution layers are fixed
- Combine multiple different feature extractors in Projection GAN loss

Configuration		FID ↓	IS ↑
A	StyleGAN3	53.57	15.30
B	+ Projected GAN & small z	22.98	57.62
C	+ Pretrained embeddings	20.91	35.79
D	+ Progressive growing	19.51	35.74
E	+ ViT & CNN as $F_{1,2}$	12.43	56.72
F	+ CLF guidance (StyleGAN-XL)	<b>12.24</b>	<b>86.21</b>

Model		Type		Objective		FID ↓	IS ↑
F <sub>1</sub>	F <sub>2</sub>	F <sub>1</sub>	F <sub>2</sub>	F <sub>1</sub>	F <sub>2</sub>		
EffNet		CNN		Class		19.51	35.74
EffNet	ResNet50	CNN	CNN	Class	Class	16.16	49.13
EffNet	ResNet50	CNN	CNN	Class	Self	18.53	38.26
EffNet	DeiT-M	CNN	ViT	Class	Class	<b>12.43</b>	<b>56.72</b>

# StyleGAN-XL



# StyleGAN-XL

## ImageNet Synthesis

<b>Model</b>	<b>FID ↓</b>	<b>sFID ↓</b>	<b>rFID ↓</b>	<b>IS ↑</b>	<b>Pr ↑</b>	<b>Rec ↑</b>
<b>Resolution 128<sup>2</sup></b>						
BigGAN	6.02	7.18	6.09	145.83	<b>0.86</b>	0.35
CDM	3.52	128.80		128.80		
ADM	5.91	5.09	13.29	93.31	0.70	<b>0.65</b>
ADM-G	2.97	5.09	3.80	141.37	0.78	0.59
StyleGAN-XL	<b>1.81</b>	<b>3.82</b>	<b>1.82</b>	<b>200.55</b>	0.77	0.55

# GigaGAN

What:

- Scale GANs to 1B parameters
- Text to Image GAN with Lower FID than Stable Diffusion
- Fast Generation (0.13s for 512px)
- Disentangled Prompt Mixing and Interpolation
- High Res Upscaling

**Scaling up GANs for Text-to-Image Synthesis**

Minguk Kang<sup>1,3</sup> Jun-Yan Zhu<sup>2</sup> Richard Zhang<sup>3</sup>  
Jaesik Park<sup>1</sup> Eli Shechtman<sup>3</sup> Sylvain Paris<sup>3</sup> Taesung Park<sup>3</sup>  
<sup>1</sup>POSTECH      <sup>2</sup>Carnegie Mellon University      <sup>3</sup>Adobe Research

# GigaGAN



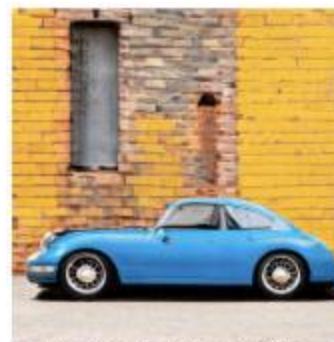
A portrait of a human growing colorful flowers from her hair. Hyperrealistic oil painting. Intricate details.



A golden luxury motorcycle parked at the King's palace, 35mm f/4.5.



a cute magical flying maltipoo at light speed, fantasy concept art, bokeh, wide sky.



a blue Porsche 356 parked in front of a yellow brick wall.



A hot air balloon in shape of a heart. Grand Canyon



Eiffel Tower, landscape photography

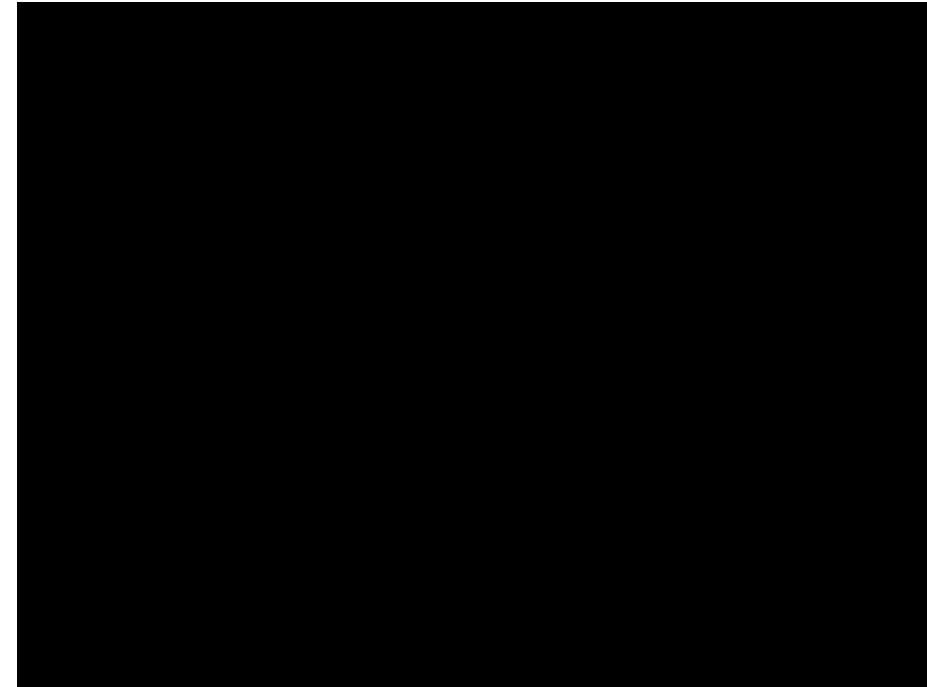


low poly bunny with cute eyes

# GigaGAN



# GigaGAN



# GigaGAN

What:

- Scale GANs to 1B parameters
- Text to Image GAN with Lower FID than Stable Diffusion
- Fast Generation (0.13s for 512px)
- Disentangled Prompt Mixing and Interpolation
- High Res Upscaling

How:

- Sample-adaptive kernel selection
- Self-attention layers
- Multi-scale generator and discriminator
- Additional loss

# GigaGAN - Generator

## Sample-adaptive kernel selection

Goal: increase model capacity

- Bank of N input features

$$\{\mathbf{K}_i \in R^{C_{in} \times C_{out} \times K \times K}\}_{i=1}^N$$

- Style vector + affine layer

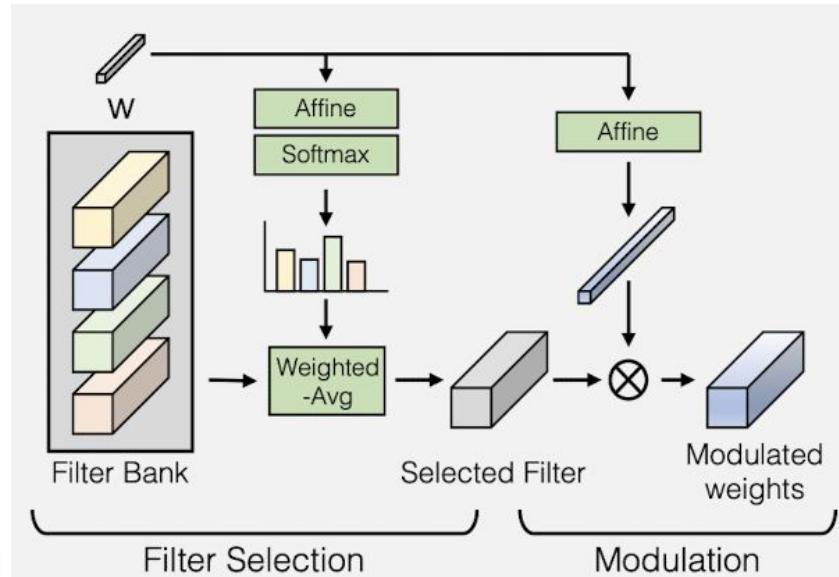
$$\mathbf{w} \in R^d \quad [W_{filter}, b_{filter}] \in R^{(d+1) \times N}$$

- Aggregated filter

$$\mathbf{K} = \sum_{i=1}^N \mathbf{K}_i \cdot \text{softmax}(W_{filter}^\top \mathbf{w} + b_{filter})_i \quad \mathbf{K} \in R^{C_{in} \times C_{out} \times K \times K}$$

- Then StyleGAN2 weight modulation conv

$$g_{\text{adaconv}}(\mathbf{f}, \mathbf{w}) = ((W_{mod}^\top \mathbf{w} + b_{mod}) \otimes \mathbf{K}) * \mathbf{f}$$



# GigaGAN - Generator

## Attention Layers

- Directly adding attention layers results in training collapse
- Dot-product self attention is NOT lipschitz
- Use L2-distance instead of dot-product distance
- Querry and Key weights have to be tied (sufficient condition)

---

### The Lipschitz Constant of Self-Attention

---

Hyunjik Kim<sup>1</sup> George Papamakarios<sup>1</sup> Andriy Mnih<sup>1</sup>

Standard dot product attention

$$\begin{aligned} DP(X) &:= \text{softmax} \left( \frac{XW^Q(XW^K)^\top}{\sqrt{D/H}} \right) XW^V \\ &= PXW^V, \end{aligned}$$

Use L2 distances instead

$$P_{ij} \propto \exp(L_{ij}) := \exp \left( -\frac{\|\mathbf{x}_i^\top W^Q - \mathbf{x}_j^\top W^K\|_2^2}{\sqrt{D/H}} \right)$$
$$W^Q = W^K$$

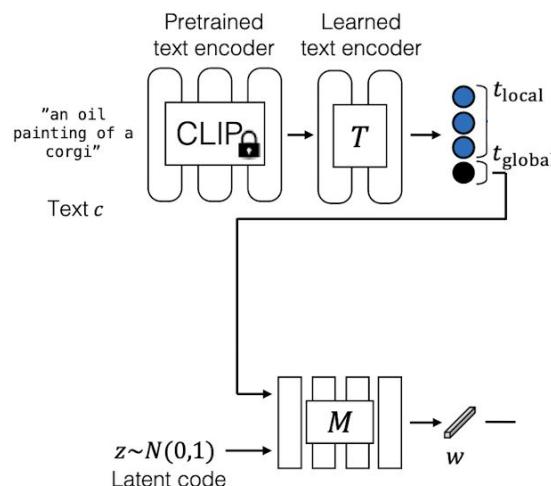
# GigaGAN - Generator

## Text-Conditioning

$$\mathbf{w} = M(\mathbf{z}, \mathbf{t}_{\text{global}})$$

$$\mathbf{t} = (\mathbf{t}_{\text{local}}, \mathbf{t}_{\text{global}}) = T(\mathcal{E}_{\text{txt}}(\mathbf{c})) \in \mathbb{R}^{C \times 768}$$

$$\mathbf{t}_{\text{local}} = \mathbf{t}_{\{1:C\} \setminus \text{EOT}} \in \mathbb{R}^{(C-1) \times 768} \quad \mathbf{t}_{\text{global}} \in \mathbb{R}^{768}$$



# GigaGAN - Generator

## Text-Conditioning

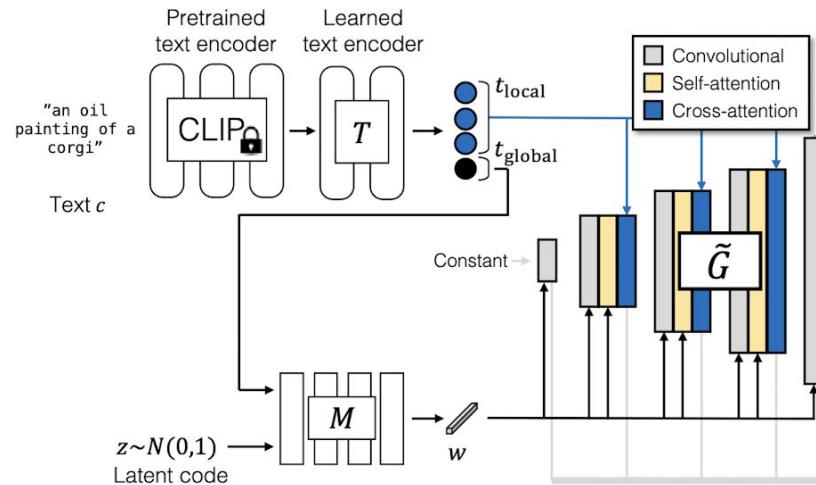
$$\mathbf{t} = (\mathbf{t}_{\text{local}}, \mathbf{t}_{\text{global}}) = T(\mathcal{E}_{\text{txt}}(\mathbf{c})) \in \mathbb{R}^{C \times 768}$$

$$\mathbf{t}_{\text{local}} = \mathbf{t}_{\{1:C\} \setminus \text{EOT}} \in \mathbb{R}^{(C-1) \times 768} \quad \mathbf{t}_{\text{global}} \in \mathbb{R}^{768}$$

$$\mathbf{w} = M(\mathbf{z}, \mathbf{t}_{\text{global}})$$

$\mathbf{t}_{\text{local}}$  Used in cross attention layers

$\mathbf{t}_{\text{global}}$  Used in computation of the style vector  $\mathbf{w}$  for adaptive kernel convs and an additional token in self attention



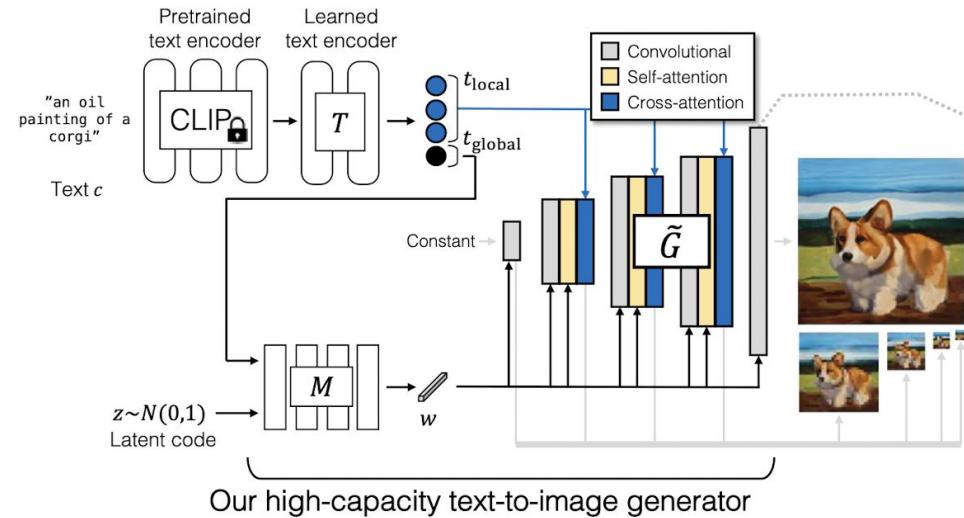
# GigaGAN - Generator

## Multi scale image generation

- Predict images at multiple resolutions
- Images at each level

$$\{\mathbf{x}_i\}_{i=0}^{L-1} = \{\mathbf{x}_0, \mathbf{x}_1, \dots, \mathbf{x}_4\}$$

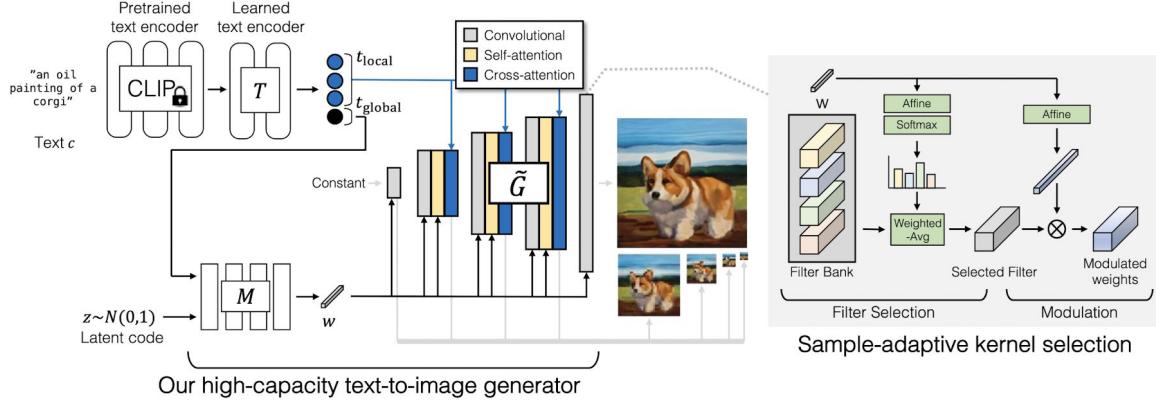
$$\{S_i\}_{i=0}^{L-1} = \{64, 32, 16, 8, 4\}$$



# GigaGAN - Generator

## Summary

- StyleGANv2
- Base architecture
- Text condition the style vector  $\mathbf{w}$
- Introduce adaptive kernels based on  $\mathbf{w}$
- L2 Lipschitz attention layers
- Apply cross attention to the local text features
- Predict images at multiple resolutions  $\{S_i\}_{i=0}^{L-1} = \{64, 32, 16, 8, 4\}$   $\{\mathbf{x}_i\}_{i=0}^{L-1} = \{\mathbf{x}_0, \mathbf{x}_1, \dots, \mathbf{x}_4\}$
- Important to keep other design details from StyleGANv2
- Turn off style mixing and path regularization as in StyleGAN-XL



# GigaGAN - Discriminator

Multiscale feature extractor  $\phi_{i \rightarrow j}$

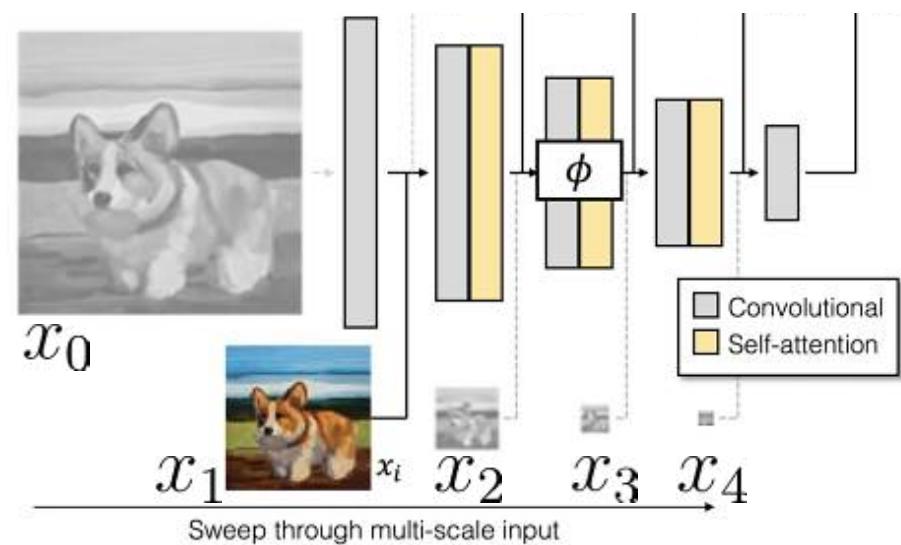
$$\{X_j^D\} = \{32, 16, 8, 4, 1\}$$

$$\phi_{i \rightarrow j} : \mathbb{R}^{X_i \times X_i \times 3} \rightarrow \mathbb{R}^{X_j^D \times X_j^D \times C_j}$$

$\phi_{i \rightarrow j}$  Is a subset of  $\phi \triangleq \phi_{0 \rightarrow L}$

$i > 0$  indicating late entry

$j < L$  indicating early exit.



# GigaGAN - Discriminator

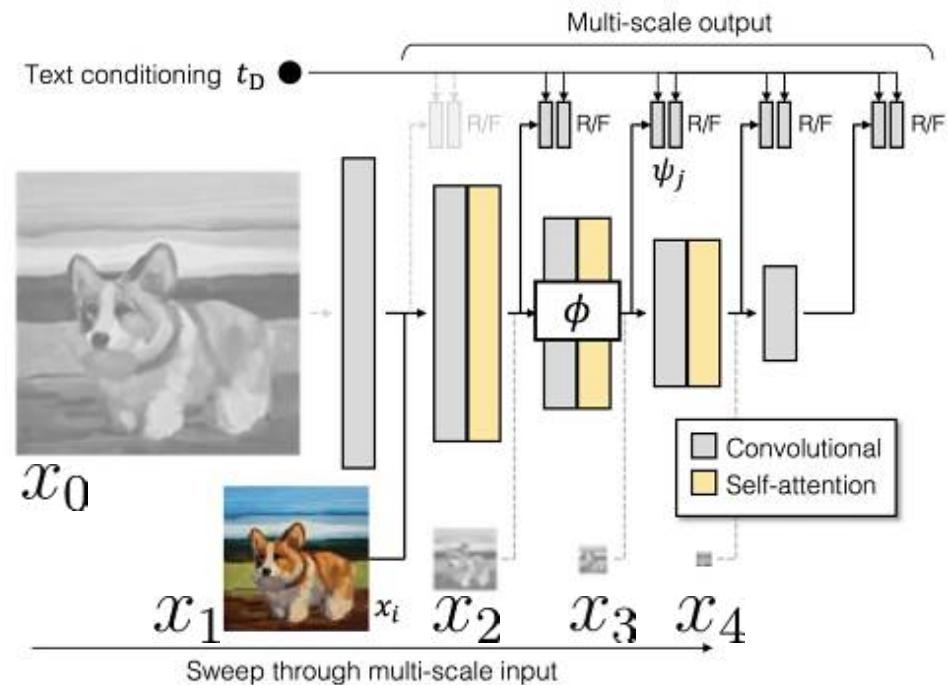
Multiscale feature extractor  $\phi_{i \rightarrow j}$

$$\phi_{i \rightarrow j} : \mathbb{R}^{X_i \times X_i \times 3} \rightarrow \mathbb{R}^{X_j^D \times X_j^D \times C_j}$$

Multiscale input / output Discriminator

$$D_{ij}(\mathbf{x}, \mathbf{c}) = \psi_j(\phi_{i \rightarrow j}(\mathbf{x}_i), \mathbf{t}_D) + \text{Conv}_{1 \times 1}(\phi_{i \rightarrow j}(\mathbf{x}_i))$$

$\psi_j$ : 4 layer 1x1 conv modulated by  $t_D$

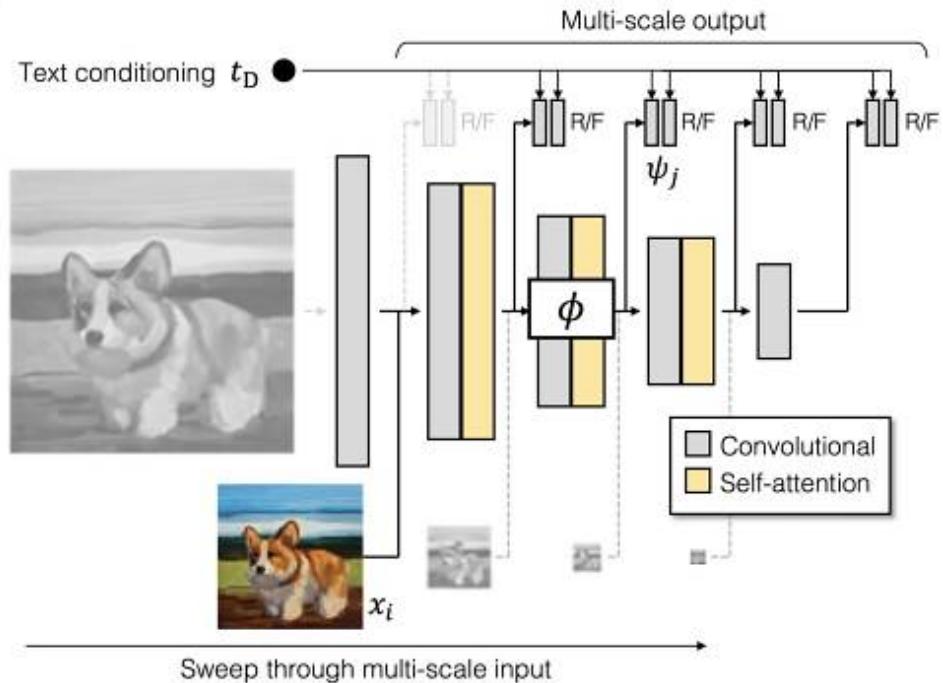


# GigaGAN - Discriminator

$$D_{ij}(\mathbf{x}, \mathbf{c}) = \psi_j(\phi_{i \rightarrow j}(\mathbf{x}_i), \mathbf{t}_D) + \text{Conv}_{1 \times 1}(\phi_{i \rightarrow j}(\mathbf{x}_i))$$

Loss:

$$\mathcal{V}_{\text{MS-VO}}(G, D) = \sum_{i=0}^{L-1} \sum_{j=i+1}^L \underbrace{\mathcal{V}_{\text{GAN}}(G_i, D_{ij}) + \mathcal{V}_{\text{match}}(G_i, D_{ij})}_{\text{Original non saturating GAN loss}}$$



# GigaGAN - Discriminator

$$D_{ij}(\mathbf{x}, \mathbf{c}) = \psi_j(\phi_{i \rightarrow j}(\mathbf{x}_i), \mathbf{t}_D) + \text{Conv}_{1 \times 1}(\phi_{i \rightarrow j}(\mathbf{x}_i))$$

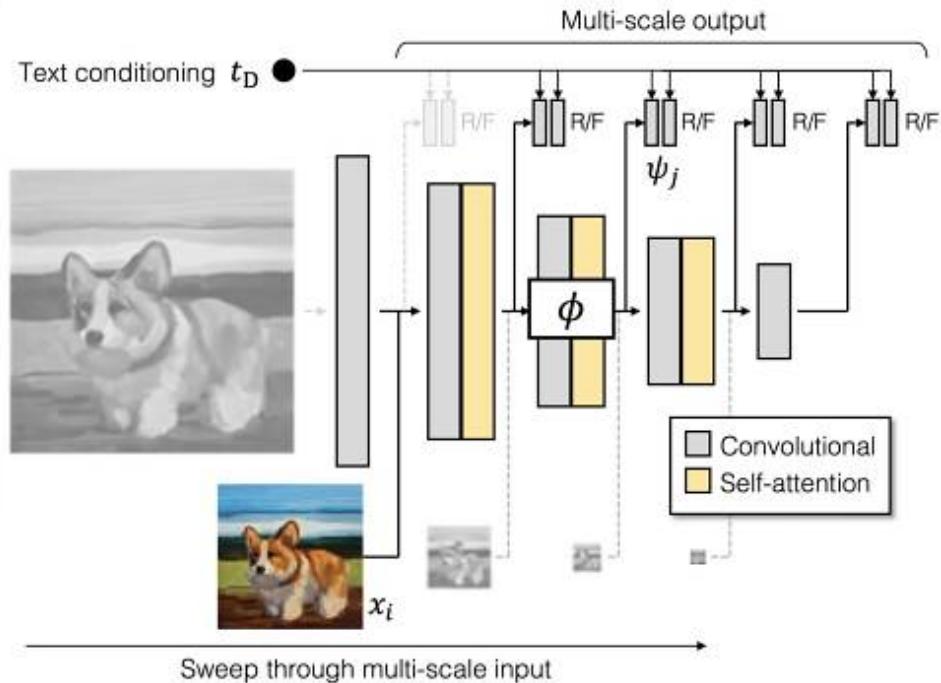
Loss:

$$\mathcal{V}_{\text{MS-VO}}(G, D) = \sum_{i=0}^{L-1} \sum_{j=i+1}^L \underbrace{\mathcal{V}_{\text{GAN}}(G_i, D_{ij}) + \mathcal{V}_{\text{match}}(G_i, D_{ij})}_{\text{Original non saturating GAN loss}}$$

Matching-aware loss

$$\begin{aligned} \mathcal{V}_{\text{match}} = \mathbb{E}_{\mathbf{x}, \mathbf{c}, \hat{\mathbf{c}}} & [ \log(1 + \exp(D(\mathbf{x}, \hat{\mathbf{c}}))) \\ & + \log(1 + \exp(D(G(\mathbf{c}), \hat{\mathbf{c}}))) ] \end{aligned}$$

Enforce discriminator to incorporate conditioning on  $\hat{\mathbf{c}}$  (a randomly sampled caption from the dataset)



# GigaGAN - Additional Losses

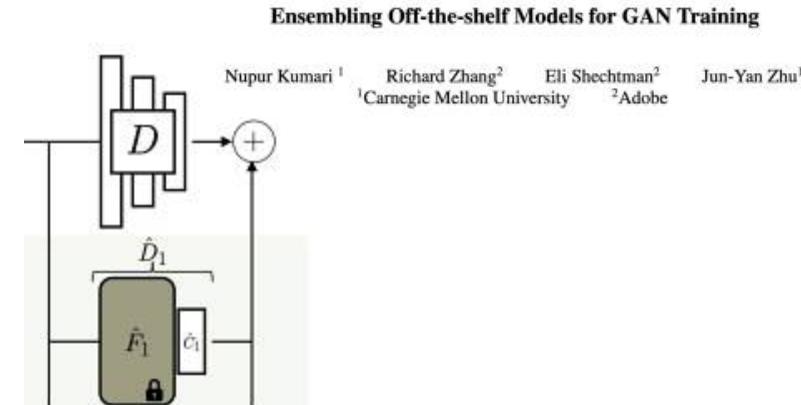
$$\mathcal{V}(G, D) = \mathcal{V}_{\text{MS-IO}}(G, D) + \mathcal{L}_{\text{CLIP}}(G) + \mathcal{L}_{\text{Vision}}(G)$$

- CLIP contrastive loss

- $\mathcal{E}_{\text{img}}$  and  $\mathcal{E}_{\text{txt}}$  are frozen CLIP models

$$\mathcal{L}_{\text{CLIP}} = \mathbb{E}_{\{\mathbf{c}_n\}} \left[ -\log \frac{\exp(\mathcal{E}_{\text{img}}(G(\mathbf{c}_0))^{\top} \mathcal{E}_{\text{txt}}(\mathbf{c}_0))}{\sum_n \exp(\mathcal{E}_{\text{img}}(G(\mathbf{c}_0))^{\top} \mathcal{E}_{\text{txt}}(\mathbf{c}_n))} \right]$$

- Vision aided GAN:  $\hat{D}_k = \hat{C}_k \circ \hat{F}_k$
- Freeze CLIP image encoder
  - Extract intermediate features and learn a simple discriminator
  - Text condition through modulation



# GigaGAN Experiments

Model	FID-10k ↓	CLIP Score ↑	# Param.
StyleGAN2	29.91	0.222	27.8M
+ Larger (5.7×)	34.07	0.223	158.9M
+ Tuned	28.11	0.228	26.2M
+ Attention	23.87	0.235	59.0M
+ Matching-aware D	27.29	0.250	59.0M
+ Matching-aware G and D	21.66	0.254	59.0M
+ Adaptive convolution	19.97	0.261	80.2M
+ Deeper	19.18	0.263	161.9M
+ CLIP loss	14.88	0.280	161.9M
+ Multi-scale training	14.92	0.300	164.0M
+ Vision-aided GAN	13.67	0.287	164.0M
+ Scale-up ( <b>GigaGAN</b> )	9.18	0.307	652.5M

# GigaGAN Experiments

Model	Type	# Param.	# Images	FID-30k ↓	Inf. time
GLIDE [63]	Diff	5.0B	5.94B	12.24	15.0s
LDM [79]	Diff	1.5B	0.27B	12.63	9.4s
DALL-E 2 [74]	Diff	5.5B	5.63B	10.39	-
Imagen [80]	Diff	3.0B	15.36B	7.27	9.1s
eDiff-I [5]	Diff	9.1B	11.47B	6.95	32.0s
256 DALL-E [75]	AR	12.0B	1.54B	27.50	-
Parti-750M [101]	AR	750M	3.69B	10.71	-
Parti-3B [101]	AR	3.0B	3.69B	8.10	6.4s
Parti-20B [101]	AR	20.0B	3.69B	7.23	-
LAFITE [108]	GAN	75M	-	26.94	0.02s
SD-v1.5* [78]	Diff	0.9B	3.16B	9.62	2.9s
512 Muse-3B [10]	AR	3.0B	0.51B	7.88	1.3s
GigaGAN	GAN	1.0B	0.98B	9.09	0.13s

Model	Steps	FID-5k ↓	CLIP ↑	Inf. time
SD-distilled-2 [59]	2	37.3	0.27	0.23s
SD-distilled-4 [59]	4	26.0	0.30	0.33s
SD-distilled-8 [59]	8	26.9	0.30	0.52s
SD-distilled-16 [59]	16	28.8	0.30	0.88s
<b>GigaGAN</b>	1	21.1	0.32	0.13s

# GigaGAN Experiments

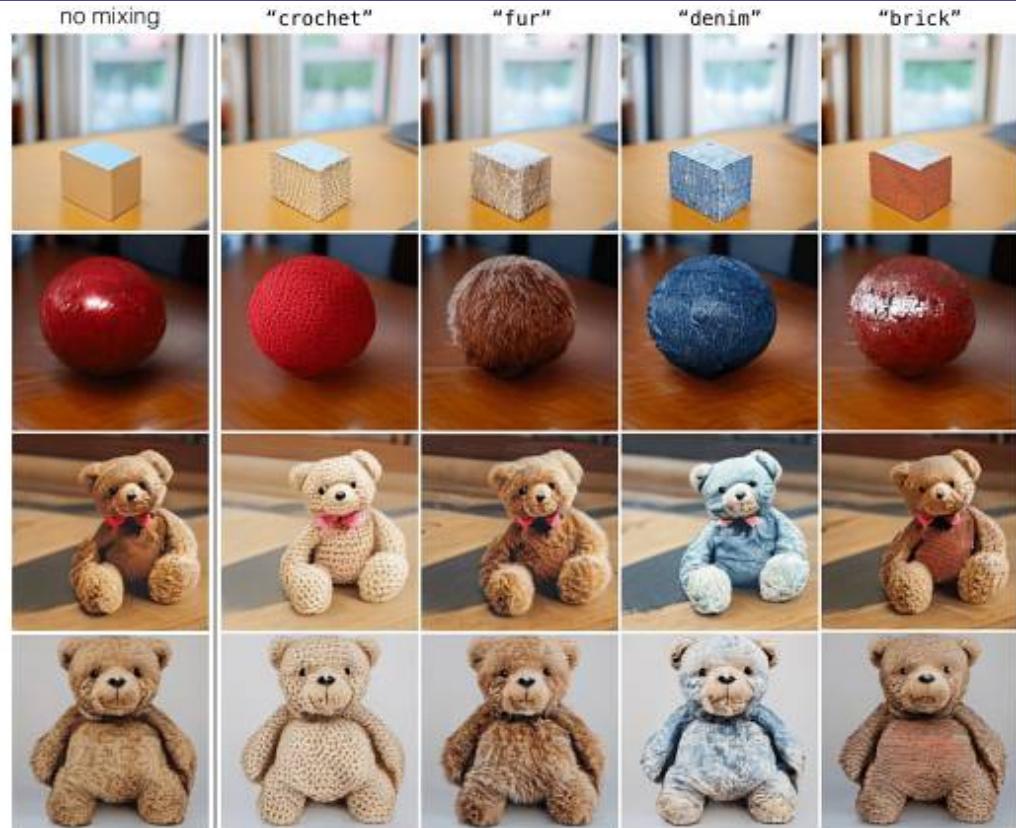
- Disentangled latent space
- Start with: “a X on tabletop”, and generate style w
- Re compute style w and t for second half with prompt “a X with the texture of Y on tabletop”

“a cube  
on tabletop”

“a ball  
on tabletop”

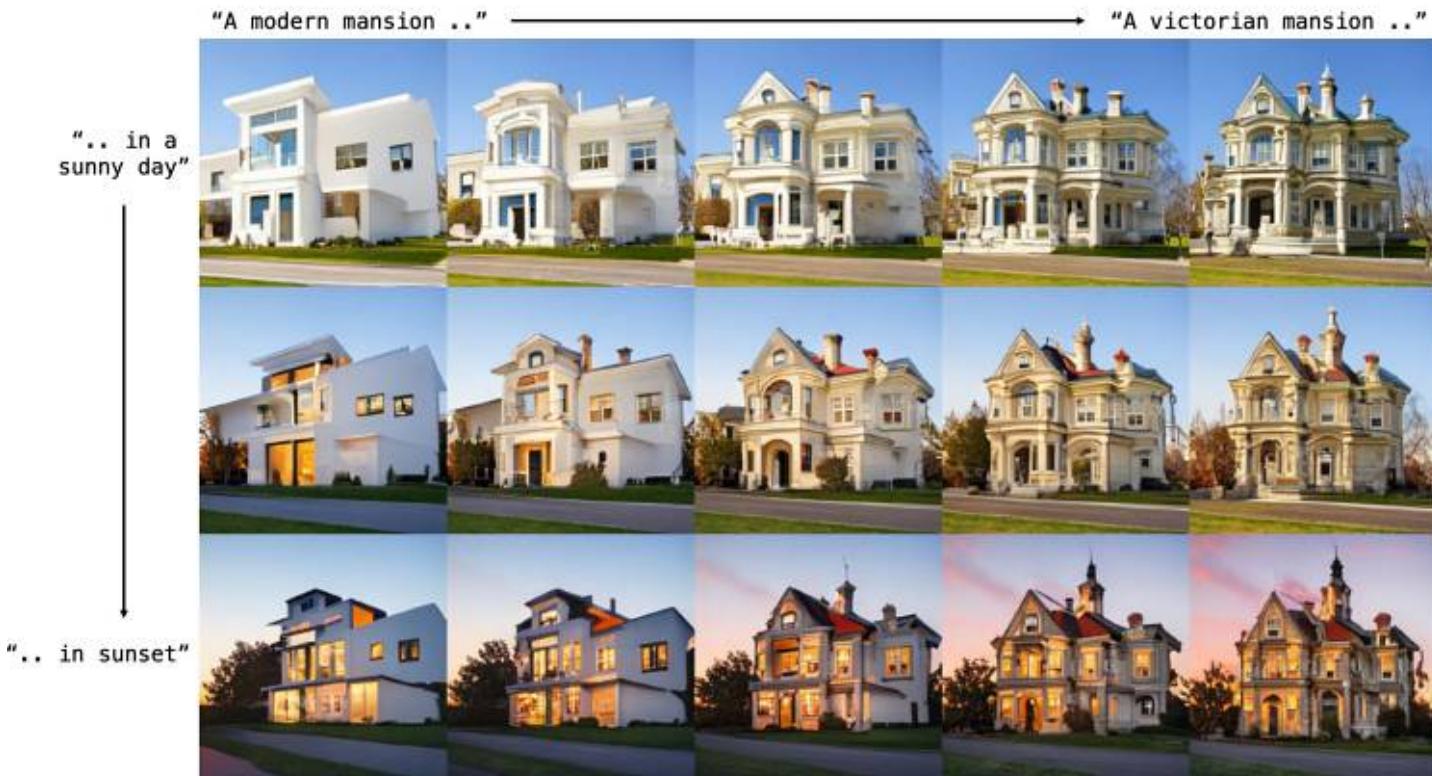
“a teddy bear  
on tabletop”

“a teddy bear  
on tabletop”



# GigaGAN Experiments

- Prompt interpolation
- Fix noise vector Z
- Change the caption c



# GigaGAN Experiments

## Failure modes

"A teddy bear on a skateboard in times square."

Ours  
(0.13s / img)



Stable Diffusion  
(2.9s / img)



DALL·E 2



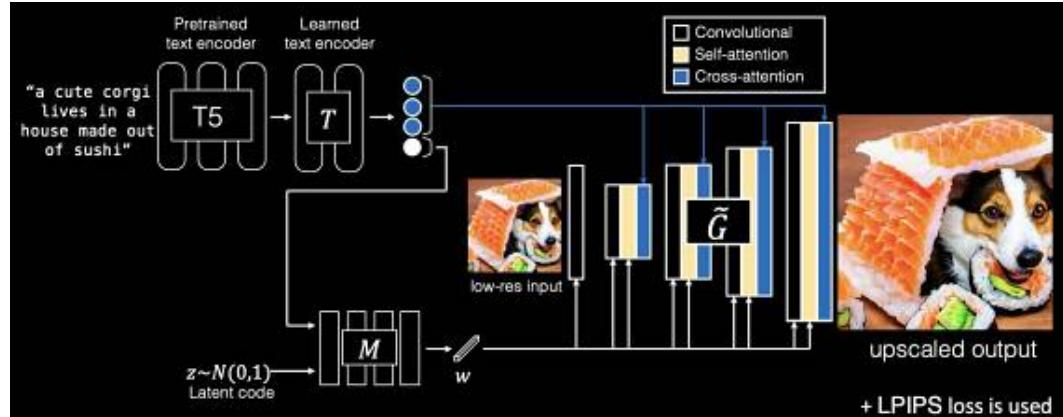
# GigaGAN - Upsampler

Trains a text conditioned superresolution model

Generator becomes an asymmetric U-Net

- Input 64x64 image
- 3 downsampling blocks, 6 upsampling blocks
- Remove Vision-aided GAN loss
- Add LPIPS Perceptual Loss
- All other losses are the same

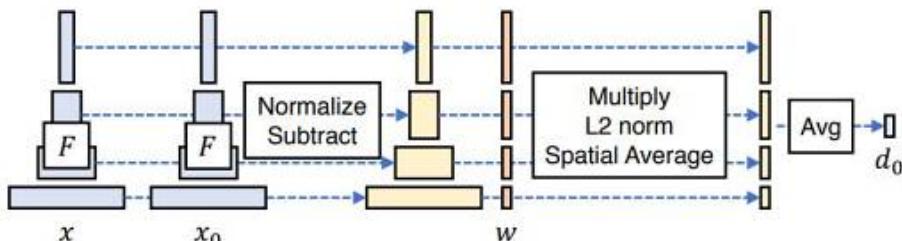
$$d(x, x_0) = \sum_l \frac{1}{H_l W_l} \sum_{h,w} \|w_l \odot (\hat{y}_{hw}^l - \hat{y}_{0hw}^l)\|_2^2$$



The Unreasonable Effectiveness of Deep Features as a Perceptual Metric

Richard Zhang<sup>1</sup> Phillip Isola<sup>1,2</sup> Alexei A. Efros<sup>1</sup>  
<sup>1</sup>UC Berkeley <sup>2</sup>OpenAI  
`{rich.zhang, isola, efros}@eecs.berkeley.edu`

Eli Shechtman<sup>3</sup> Oliver Wang<sup>3</sup>  
<sup>3</sup>Adobe Research  
`{elishe, owang}@adobe.com`



# Summary

---

- Motivation & Definition of Implicit Models
- Original GAN (Goodfellow et al, 2014)
- Evaluation: Parzen, Inception, Frechet
- Some Theory: Bayes-optimal Discriminator; Jensen-Shannon Divergence; Mode Collapse; Avoiding Saturation
- GAN Progression:
  - DC GAN (Radford et al, 2016)
  - Improved Training of GANs (Salimans et al, 2016)
  - WGAN, WGAN-GP, Progressive GAN, SN-GAN, SAGAN
  - BigGAN, StyleGAN
  - More GANs: BigGAN-Deep, StyleGAN-v2, VIB-GAN, LOGAN
- Creative Conditional GANs
- GANs and Representations
- GANs as Energy Models
- GANs and Optimal Transport
- Implicit Likelihood Models
- Moment Matching
- Other uses of Adversarial Loss: Transfer Learning, Fairness
- GANs and Imitation Learning
- Scaling up: GigaGAN, StyleGAN-XL
- **GAN losses for improved quantized tokenization: VQGAN, ViT-VQGAN**

# VQGAN

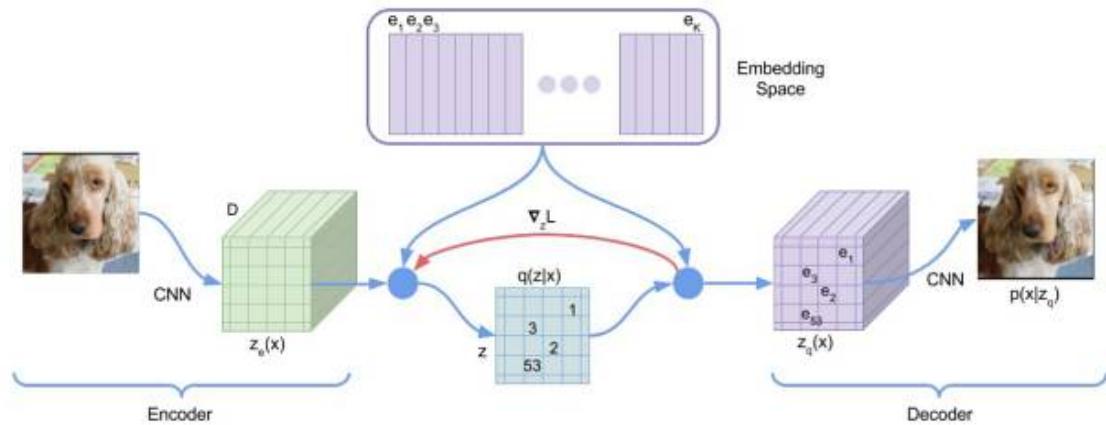
- Recall VQ-VAE
- VAE with a discrete latent code
  - Initially assumed uniform prior over  $z$
  - After encoder is trained, train an autoregressive to learn the prior (ie. pixel CNN)

## Neural Discrete Representation Learning

Aaron van den Oord  
DeepMind  
avdnoord@google.com

Oriol Vinyals  
DeepMind  
vinyals@google.com

Koray Kavukcuoglu  
DeepMind  
korayk@google.com



$$L = \log p(x|z_q(x)) + \|\text{sg}[z_e(x)] - e\|_2^2 + \beta \|z_e(x) - \text{sg}[e]\|_2^2$$

# VQGAN

## Taming Transformers for High-Resolution Image Synthesis

Patrick Esser\*    Robin Rombach\*    Björn Ommer

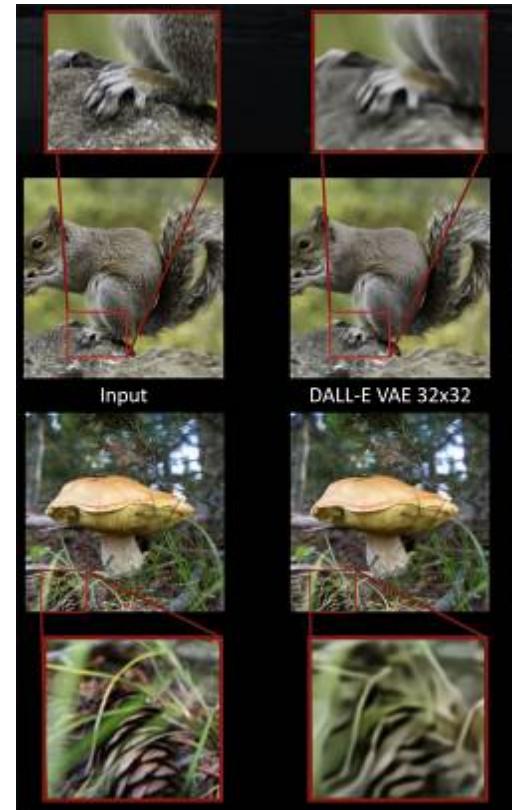
Heidelberg Collaboratory for Image Processing, IWR, Heidelberg University, Germany

\*Both authors contributed equally to this work

- VQVAE has blurry smoothed out details

$$\log p(x|z_q(x))$$

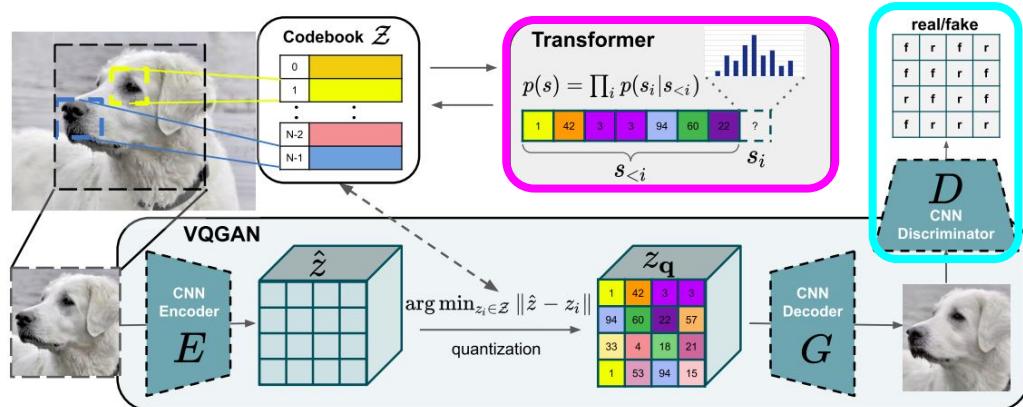
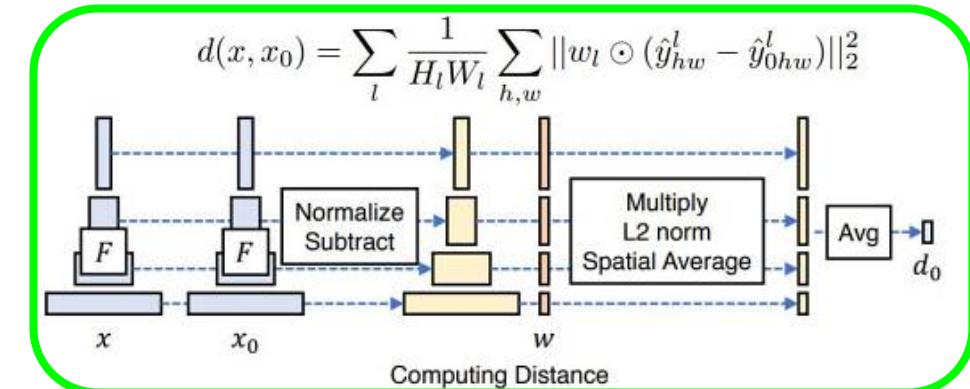
- GANs have been successful at achieving crisp and realistic looking images



# VQGAN

3 main innovations

- Add **perceptual loss (LPIPS)**
- Add **patch GAN loss**
- **Transformer** to learn the prior over latent codes



# VQGAN

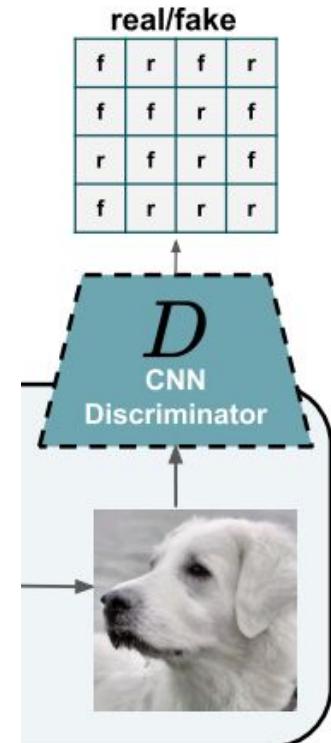
Loss function:

$$\mathcal{Q}^* = \arg \min_{E, G, \mathcal{Z}} \max_D \mathbb{E}_{x \sim p(x)} \left[ \mathcal{L}_{\text{VQ}}(E, G, \mathcal{Z}) + \lambda \mathcal{L}_{\text{GAN}}(\{E, G, \mathcal{Z}\}, D) \right]$$

$$\mathcal{L}_{\text{GAN}}(\{E, G, \mathcal{Z}\}, D) = [\log D(x) + \log(1 - D(\hat{x}))]$$

Adaptive weights:  $\lambda = \frac{\nabla_{G_L} [\mathcal{L}_{\text{rec}}]}{\nabla_{G_L} [\mathcal{L}_{\text{GAN}}] + \delta}$

Patch GAN



# VQGAN

Encoder	Decoder
$x \in \mathbb{R}^{H \times W \times C}$	$z_q \in \mathbb{R}^{h \times w \times n_z}$
$\text{Conv2D} \rightarrow \mathbb{R}^{H \times W \times C'}$	$\text{Conv2D} \rightarrow \mathbb{R}^{h \times w \times C''}$
$m \times \{ \text{Residual Block, Downsample Block} \} \rightarrow \mathbb{R}^{h \times w \times C''}$	$\text{Residual Block} \rightarrow \mathbb{R}^{h \times w \times C''}$
$\text{Residual Block} \rightarrow \mathbb{R}^{h \times w \times C''}$	$\text{Non-Local Block} \rightarrow \mathbb{R}^{h \times w \times C''}$
$\text{Non-Local Block} \rightarrow \mathbb{R}^{h \times w \times C''}$	$\text{Residual Block} \rightarrow \mathbb{R}^{h \times w \times C''}$
$\text{Residual Block} \rightarrow \mathbb{R}^{h \times w \times C''}$	$m \times \{ \text{Residual Block, Upsample Block} \} \rightarrow \mathbb{R}^{H \times W \times C'}$
$\text{GroupNorm, Swish, Conv2D} \rightarrow \mathbb{R}^{h \times w \times n_z}$	$\text{GroupNorm, Swish, Conv2D} \rightarrow \mathbb{R}^{H \times W \times C}$

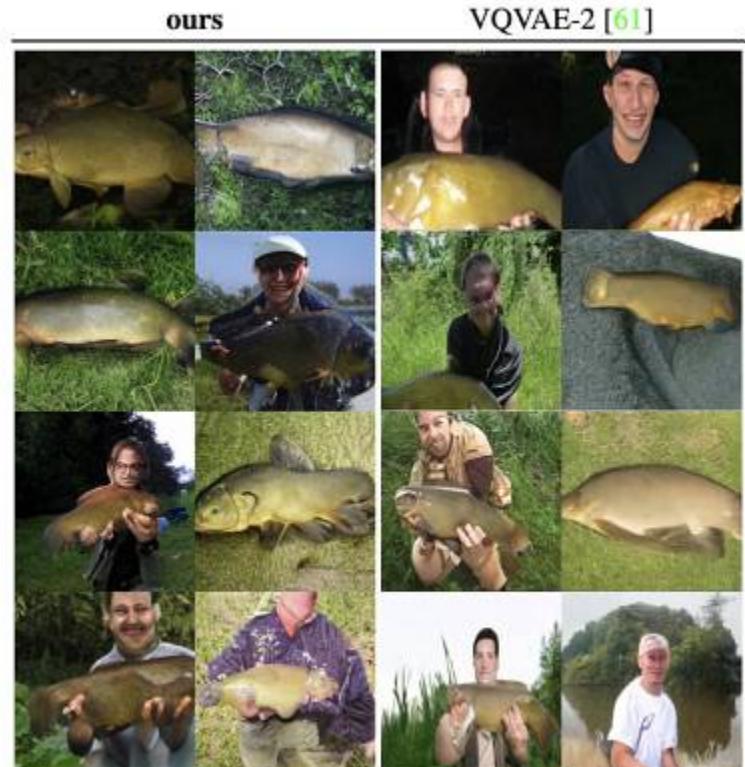
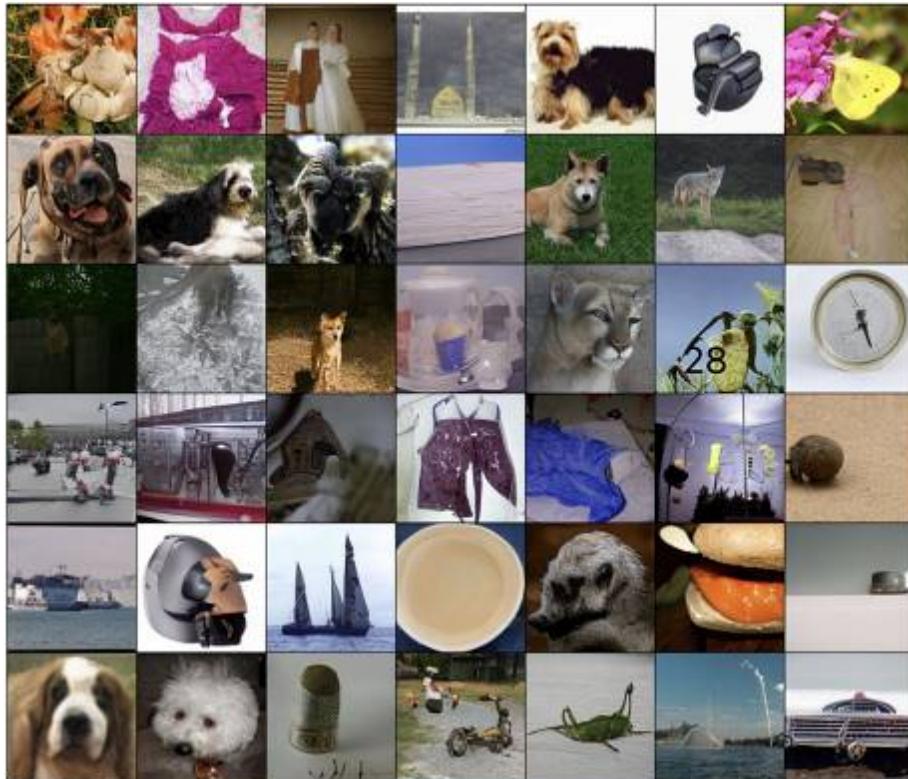
# VQGAN

- Preserves high frequency details and textures
- Enables higher compression ratios
- Less tokens in the latent space makes it easier to learn a prior with a transformer

Model	Codebook Size	dim $\mathcal{Z}$	FID/val	FID/train
VQVAE-2	$64 \times 64 \text{ & } 32 \times 32$	512	n/a	$\sim 10$
DALL-E [59]	$32 \times 32$	8192	32.01	33.88
VQGAN	$16 \times 16$	1024	7.94	10.54
VQGAN	$16 \times 16$	16384	4.98	7.41
VQGAN*	$32 \times 32$	8192	1.49	3.24
VQGAN	$64 \times 64 \text{ & } 32 \times 32$	512	1.45	2.78



# VQGAN



# ViT-VQGAN

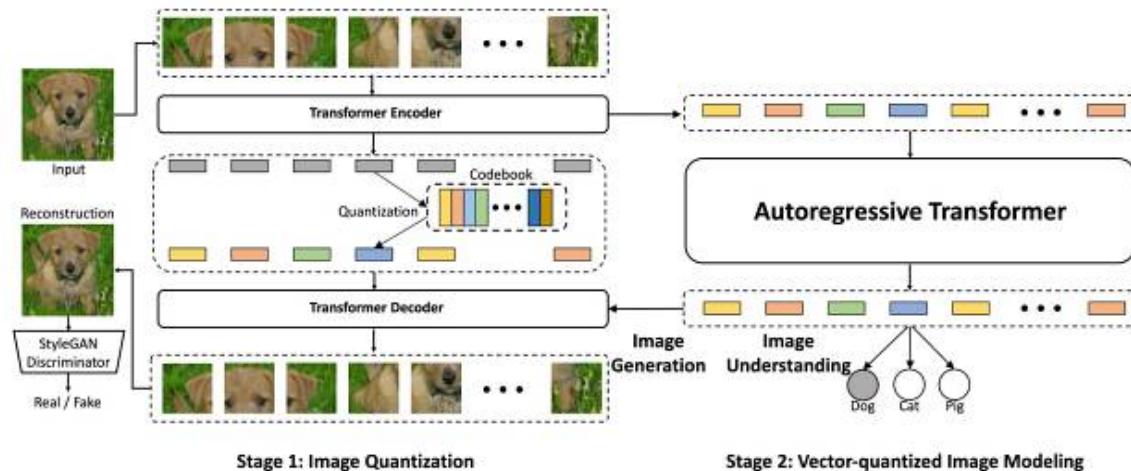
## VECTOR-QUANTIZED IMAGE MODELING WITH IMPROVED VQGAN

Improved image quantization process

- Transformer encoder / decoder
- Adversarial loss
- Logit laplace loss
- Factorized Codebook

Jiahui Yu   Xin Li   Jing Yu Koh   Han Zhang   Ruoming Pang   James Qin  
Alexander Ku   Yuanzhong Xu   Jason Baldridge   Yonghui Wu

Google Research



# ViT-VQGAN

$$L = L_{\text{VQ}} + 0.1 L_{\text{Adv}} + 0.1 L_{\text{Perceptual}} + 0.1 L_{\text{Logit-laplace}} + 1.0 L_2$$

$$L_{\text{VQ}} = \|\text{sg}[z_e(x)] - e\|_2^2 + \beta \|z_e(x) - \text{sg}[e]\|_2^2$$

- Use StyleGANv2 Discriminator instead of PatchGAN

# ViT-VQGAN

$$L = L_{\text{VQ}} + 0.1 L_{\text{Adv}} + 0.1 L_{\text{Perceptual}} + 0.1 L_{\text{Logit-laplace}} + 1.0 L_2$$

$$L_{\text{VQ}} = \|\text{sg}[z_e(x)] - e\|_2^2 + \beta \|z_e(x) - \text{sg}[e]\|_2^2$$

- Use StyleGANv2 Discriminator instead of PatchGAN
- Logit-Laplace loss - consider modeling a pdf over (0, 1)

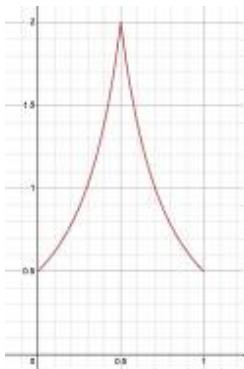
$$f(x | \mu, b) = \frac{1}{2bx(1-x)} \exp\left(-\frac{|\text{logit}(x) - \mu|}{b}\right)$$

# ViT-VQGAN

$$L = L_{\text{VQ}} + 0.1 L_{\text{Adv}} + 0.1 L_{\text{Perceptual}} + 0.1 L_{\text{Logit-laplace}} + 1.0 L_2$$

$$L_{\text{VQ}} = \|\text{sg}[z_e(x)] - e\|_2^2 + \beta \|z_e(x) - \text{sg}[e]\|_2^2$$

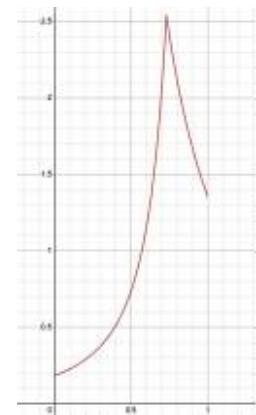
- Use StyleGANv2 Discriminator instead of PatchGAN
- Logit-Laplace loss - consider modeling a pdf over (0, 1)



$$\mu = 0, b=1$$

$$f(x | \mu, b) = \frac{1}{2bx(1-x)} \exp\left(-\frac{|\text{logit}(x) - \mu|}{b}\right)$$

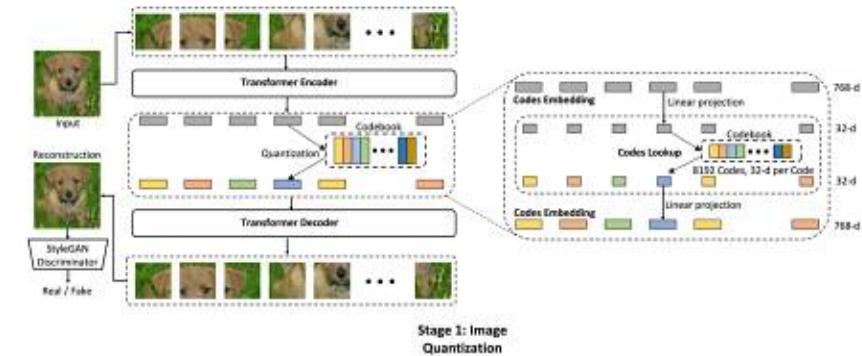
$$\mu = 1, b=1$$



# ViT-VQGAN

## Factorized Codebook

- Vanilla VQVAE results in many codes being rarely used or unused
- Introduce a linear projection from the feature to the code index
- Apply L2 normalization on encoded latents and codebook latents
  - Maps latents onto a sphere - improves stability and reconstruction quality



$$\|\ell_2(z_e(x)) - \ell_2(e_j)\|_2^2$$

# ViT-VQGAN

Model	Size	#Params	#Blocks	#Heads	Model Dim	Hidden Dim	Dropout	#Tokens
ViT-VQGAN	Small	32M	8	8	512	2048	0.0	1024
ViT-VQGAN	Base	91M	12	12	768	3072	0.0	1024
ViT-VQGAN	Large	599M	32	16	1280	5120	0.0	1024
Architecture		Model Size (encoder-decoder)		Throughput $\uparrow$ (imgs/sec)	$\ell_2$ loss $\downarrow$ (1e-2)	Logit-Laplace loss $\downarrow$	FID $\downarrow$	IS $\uparrow$
ViT-VQGAN	Small-Small	<b>1520</b>		946	3.34	-2.44	1.99	184.4
CNN-VQGAN	Channels $\times$ 1	946		960	3.81	-2.36	2.26	178.7
ViT-VQGAN	Base-Base	<b>3.09</b>		3.09	<b>-2.54</b>	<b>1.55</b>	<b>190.2</b>	
CNN-VQGAN	Channels $\times$ 2	400		400	3.44	-2.46	1.91	183.4
ViT-VQGAN	Small-Large	384		384	<b>2.88</b>	<b>-2.58</b>	<b>1.28</b>	<b>192.3</b>

Model	Dataset	Latent Size	dim $\mathcal{Z}$	FID on Validation
DALL-E dVAE	Web data	32 $\times$ 32	8192	32.00
VQGAN	ImageNet	16 $\times$ 16	1024	7.94
VQGAN	ImageNet	16 $\times$ 16	16384	4.98
VQGAN*	ImageNet	32 $\times$ 32	8192	1.49
VQGAN**	ImageNet	64 $\times$ 64 & 32 $\times$ 32	512	1.45
ViT-VQGAN (Ours)	ImageNet	32 $\times$ 32	8192	1.28

# ViT-VQGAN

Ablation on	Encoder Size		Decoder Size		Architecture		Discriminator		Latent dim	$\ell_2$ -normalized	$\ell_1$ (1e-2) $\downarrow$	$\ell_2$ (1e-3) $\downarrow$	Logit-Laplace $\downarrow$	IS $\uparrow$	FID $\downarrow$	Codebook Usage $\downarrow$	Throughput $\downarrow$
	Base	Base	ViT	StyleGAN	32	✓	3.06	3.09	-2.54	190.2	1.55	96%	960				
Model Size	Small	Small	ViT	StyleGAN	32	✓	3.22	3.34	-2.44	184.4	1.99	95%	1520				
	Small	Large	ViT	StyleGAN	32	✓	2.93	2.88	-2.58	192.3	1.28	95%	384				
Architecture	-	-	CNN	StyleGAN	32	✓	3.45	3.81	-2.36	178.7	2.26	63%	946				
	Base	Base	ViT	PatchGAN	32	✓	2.82	2.58	-2.62	165.6	3.88	89%	1227				
Codebook Learning	Base	Base	ViT	StyleGAN	256	✓	3.60	4.28	-2.38	160.1	3.68	4%	954				
	Base	Base	ViT	StyleGAN	128	✓	3.41	3.93	-2.44	173.9	2.77	14%	960				
	Base	Base	ViT	StyleGAN	64	✓	3.18	3.37	-2.49	179.5	2.50	37%	960				
	Base	Base	ViT	StyleGAN	16	✓	3.00	2.96	-2.54	191.2	1.50	95%	960				
	Base	Base	ViT	StyleGAN	8	✓	2.98	2.92	-2.55	189.5	1.52	96%	960				
	Base	Base	ViT	StyleGAN	4	✓	3.55	4.18	-2.37	143.8	3.68	96%	960				
	Base	Base	ViT	StyleGAN	32	✗	4.13	5.41	-2.20	123.6	5.44	2%	960				

# ViT-VQGAN

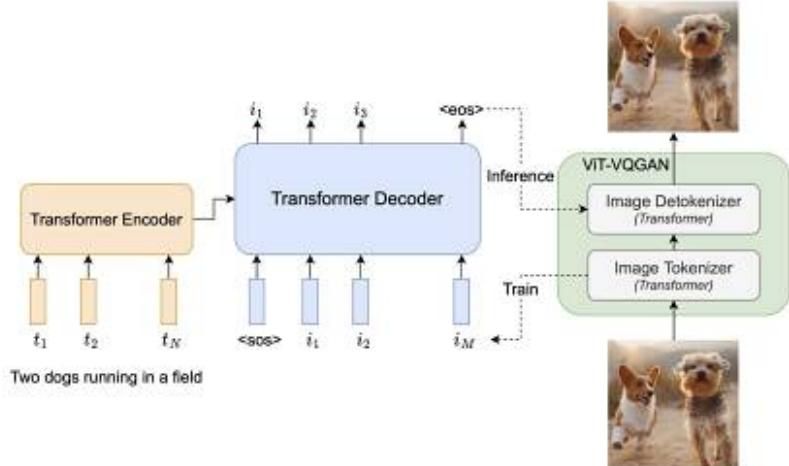
---

- a better image quantizer with respect to both computational efficiency and reconstruction quality
- faster quantizer quantizer can speed up stage 2 training (where augmentations are applied)
- Better quantizer can reduce information loss, important for image understanding

# Applications

# Parti

Pathways Autoregressive Text-to-Image Model



Stable Diffusion

