(Big)GANs

Jeff Donahue
DeepMind

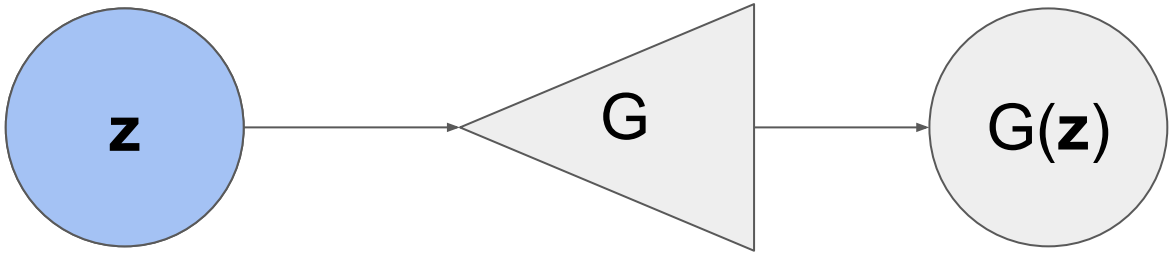
Generative Adversarial Networks (GAN)

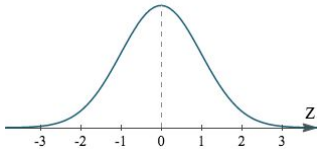
The Generator

latent (“noise”) vector
z ~ P(**z**)

generator G:
a deep neural network
(usually)

generated data
G(**z**)

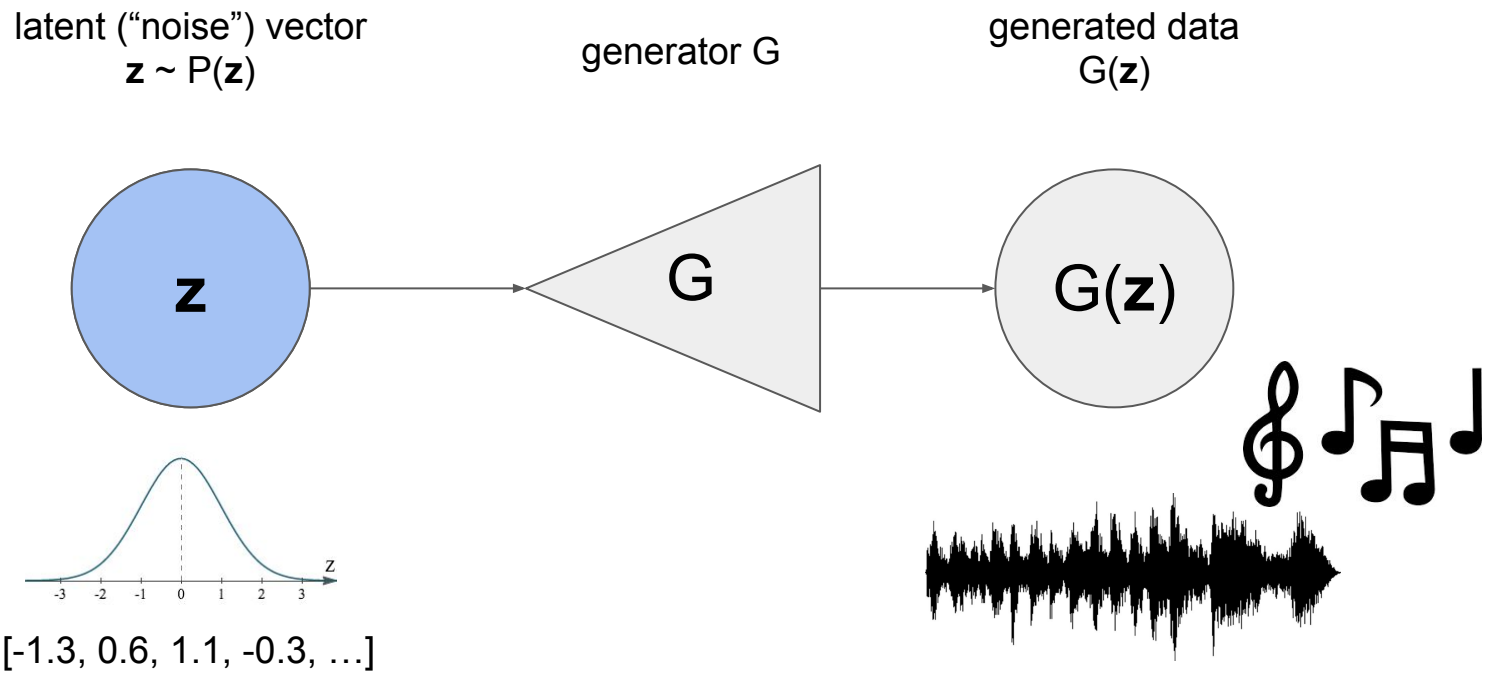




[-1.3, 0.6, 1.1, -0.3, …]

Generative Adversarial Networks (GAN)

The Generator



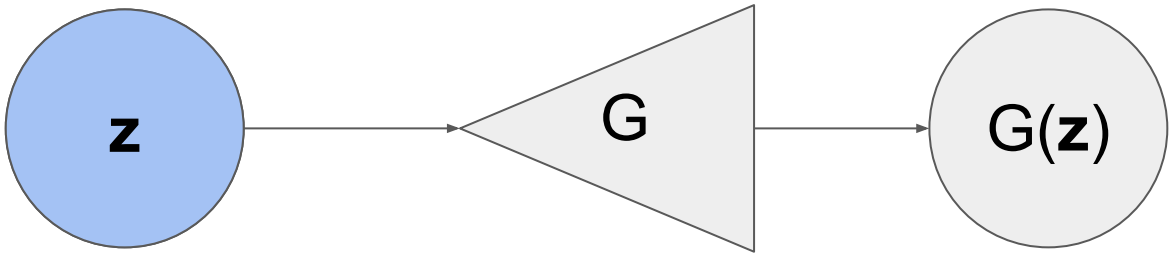
Generative Adversarial Networks (GAN)

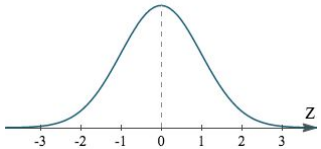
The Generator

latent (“noise”) vector
z ~ P(**z**)

generator G

generated data
G(**z**)





[-1.3, 0.6, 1.1, -0.3, …]

It was the best of times, it was the worst of times, it was the age of wisdom, it was the age of foolishness...

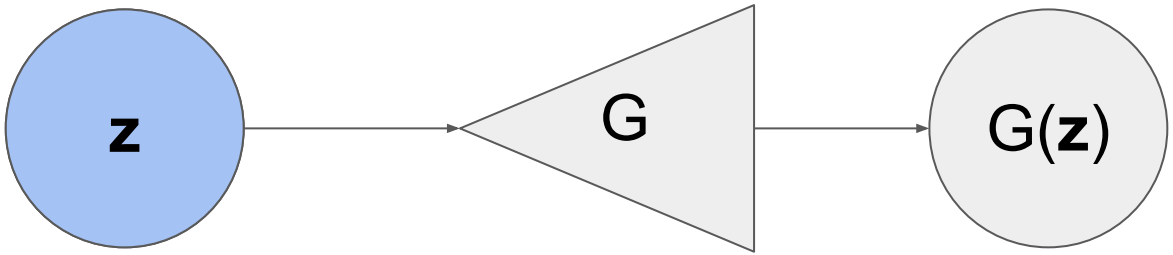
Generative Adversarial Networks (GAN)

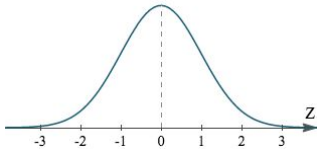
The Generator

latent (“noise”) vector
z ~ P(**z**)

generator G

generated data
G(**z**)





[-1.3, 0.6, 1.1, -0.3, …]

Any kind of data you want!

Generative Adversarial Networks (GAN)

The Discriminator

real data **x** ~ P(**x**)

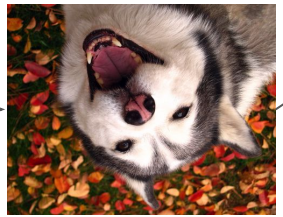
D

Predict 0-1:
real (1)
or
generated (0)?

generator G

G

generated data
G(**z**)



Generative Adversarial Networks (GAN)

The Discriminator

real data **x** ~ P(**x**)

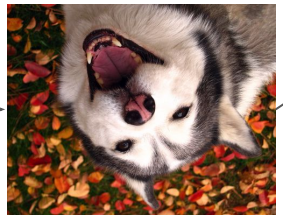
D

Predict 0-1:
real (1)
or
generated (0)?

generator G

G

generated data
G(**z**)



Discriminator's objective:
predict whether my inputs are real or generated as accurately as possible

Generative Adversarial Networks (GAN)

The Discriminator

Generator's objective: "fool" the discriminator into believing my generated data are real

real data **x** ~ P(**x**)

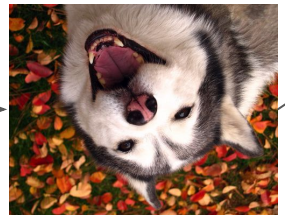
D

Predict 0-1:
real (1)
or
generated (0)?

generator G

G

generated data
G(**z**)



Discriminator's objective: predict whether my inputs are real or generated as accurately as possible

Generative Adversarial Networks (GAN)

The ~~Discriminator~~ **Teacher** (a less “adversarial” view)

Generator’s objective:
~~“fool” the discriminator~~
make the teacher happy
by making my generated data look real

real data **x** ~ P(**x**)

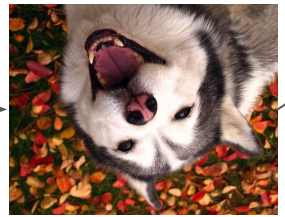
D

Predict 0-1:
real (1)
or
generated (0)?

generator G

G

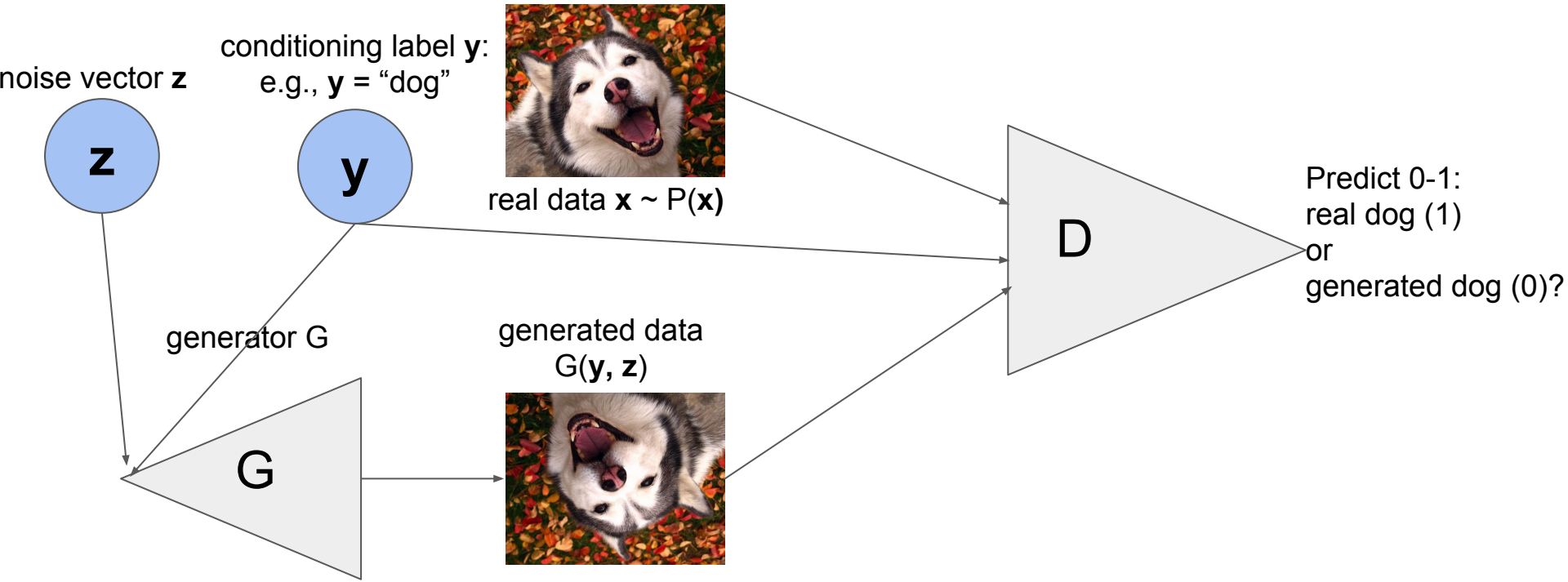
generated data
G(**z**)



Teacher’s objective:
predict whether my inputs are real or generated as accurately as possible, **so that I can tell the generator how to improve them**

Generative Adversarial Networks (GAN)

Conditional version



Generative Adversarial Networks (GAN)

The Game

$$\min_G \max_D V(D,G) = \underbrace{\mathbb{E}_{\boldsymbol{x}\sim p_{\text{data}}(\boldsymbol{x})}[\log D(\boldsymbol{x})]}_{\text{log-probability that D correctly predicts real data } \mathbf{x} \text{ are real}} + \underbrace{\mathbb{E}_{\boldsymbol{z}\sim p_{\boldsymbol{z}}(\boldsymbol{z})}[\log(1 - D(G(\boldsymbol{z})))]}_{\text{log-probability that D correctly predicts generated data G(z) are generated}}$$

discriminator's (D) goal: **maximize** prediction accuracy

generator's (G) goal: **minimize** D's prediction accuracy,
by **fooling** D into believing its outputs G(**z**) are real as often as possible

GANs. Goodfellow et al., 2014. https://arxiv.org/abs/1406.2661

Generative Adversarial Networks (GAN)

Training Algorithm

for number of training iterations **do**

  for k steps **do**

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

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

  end for

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

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

end for

GANs. Goodfellow et al., 2014. https://arxiv.org/abs/1406.2661

Generative Adversarial Networks (GAN)

What happens at convergence?

Theorem 1. *The global minimum of the virtual training criterion $C(G)$ is achieved if and only if $p_g = p_{data}$.*

In other words, the generator is **perfect** if the objective we train towards is perfectly satisfied: its outputs are **indistinguishable** from the dataset we trained it to mimic.

The discriminator simply predicts ½ probability of being real for all inputs, real or generated. **The generator "wins"**!

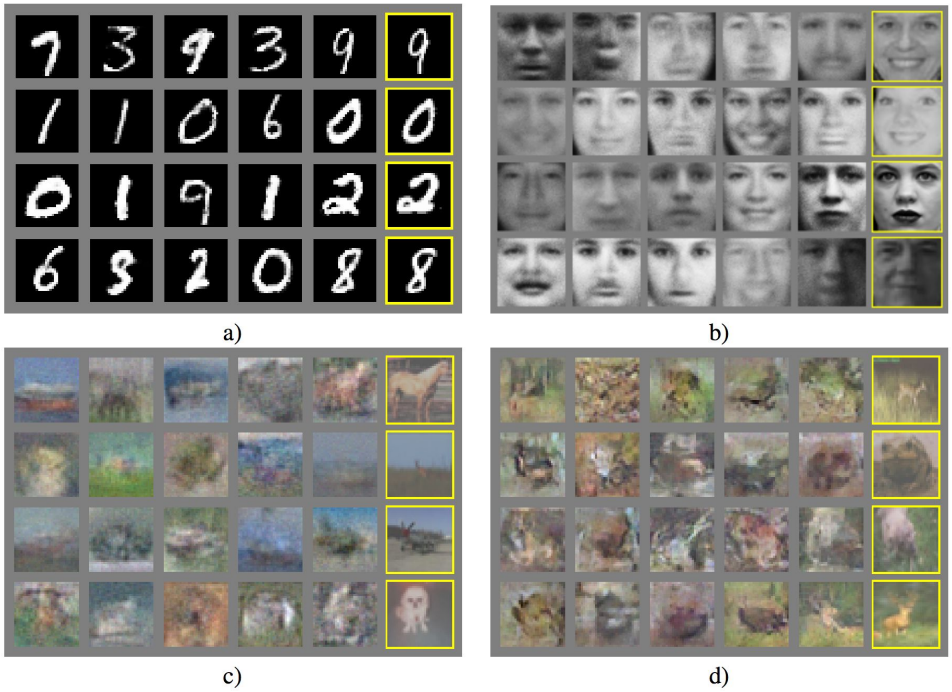
GANs. Goodfellow et al., 2014. https://arxiv.org/abs/1406.2661

Generative Adversarial Networks (GAN)

Evolution: from MNIST to ImageNet, and beyond

First results from original GAN paper:

- simple data (~32x32 images)
- simple models
- both G and D were (basically) ReLU networks
- In (a) & (b), G's output and D's input were both **flat vectors** (displayed as 2D images here)
  - ignored spatial structure: adjacent pixels treated the same as far away pixels



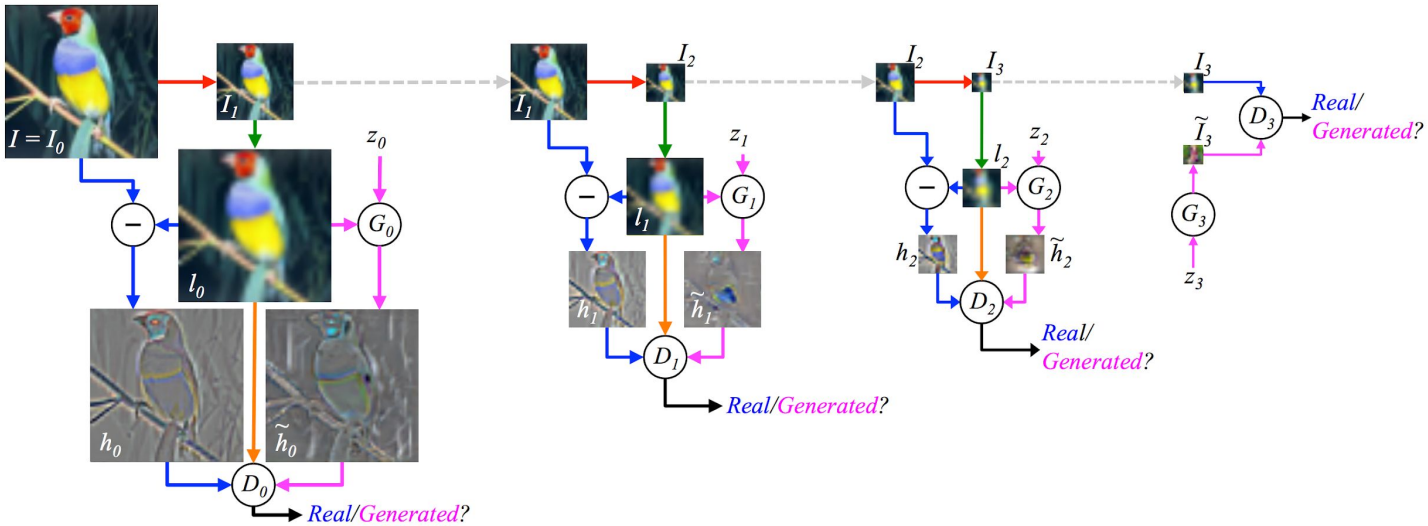
GANs. Goodfellow et al., 2014. https://arxiv.org/abs/1406.2661

Generative Adversarial Networks (GAN)

Evolution: from MNIST to ImageNet, and beyond

LAPGANs
(Laplacian GANs)

Starting from a tiny image, upsample to larger image and generate the difference (or Laplacian)



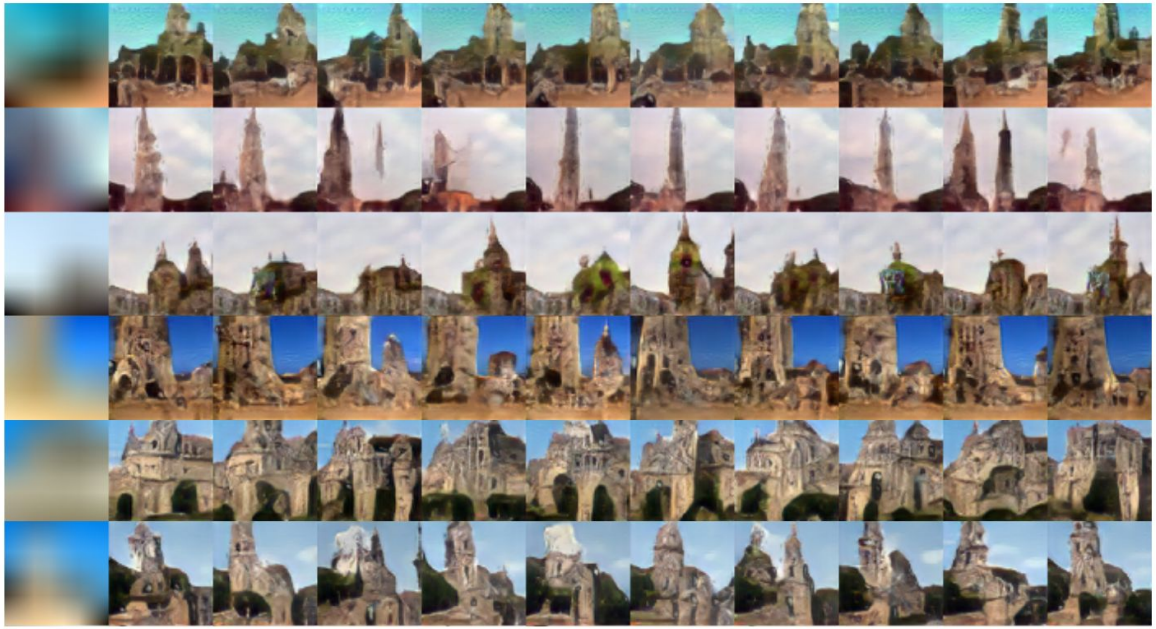
LAPGANs. Denton et al., 2015. https://arxiv.org/abs/1506.05751

Generative Adversarial Networks (GAN)

Evolution: from MNIST to ImageNet, and beyond

LAPGANs
(Laplacian GANs)

Starting from a tiny image, upsample to larger image and generate the difference (or Laplacian)



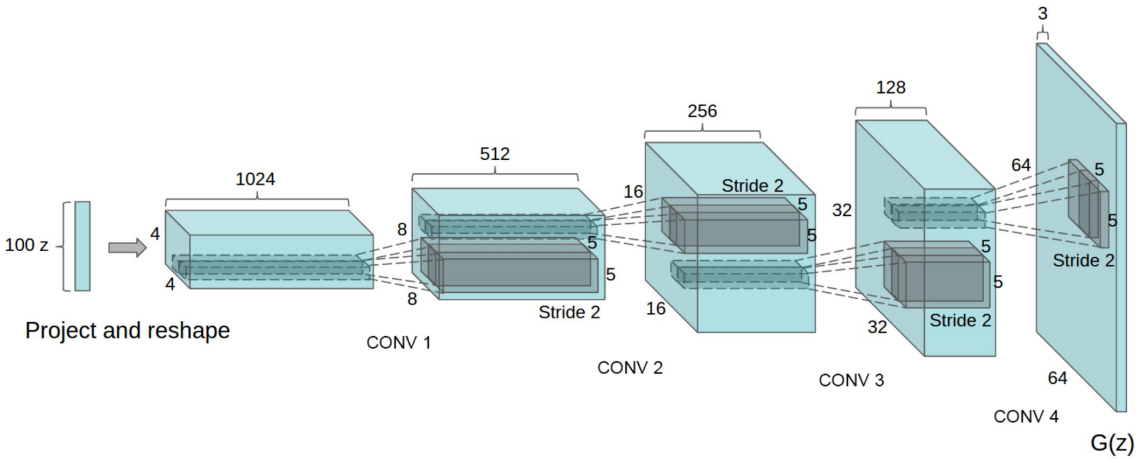
LAPGANs. Denton et al., 2015. https://arxiv.org/abs/1506.05751

Generative Adversarial Networks (GAN)

Evolution: from MNIST to ImageNet, and beyond

"**DCGAN**" (deep convolutional GANs)

- simply use deep convnets for G and D
- importantly, **batch normalization** (Ioffe and Szegedy, 2015) helps to stabilize notoriously difficult learning process



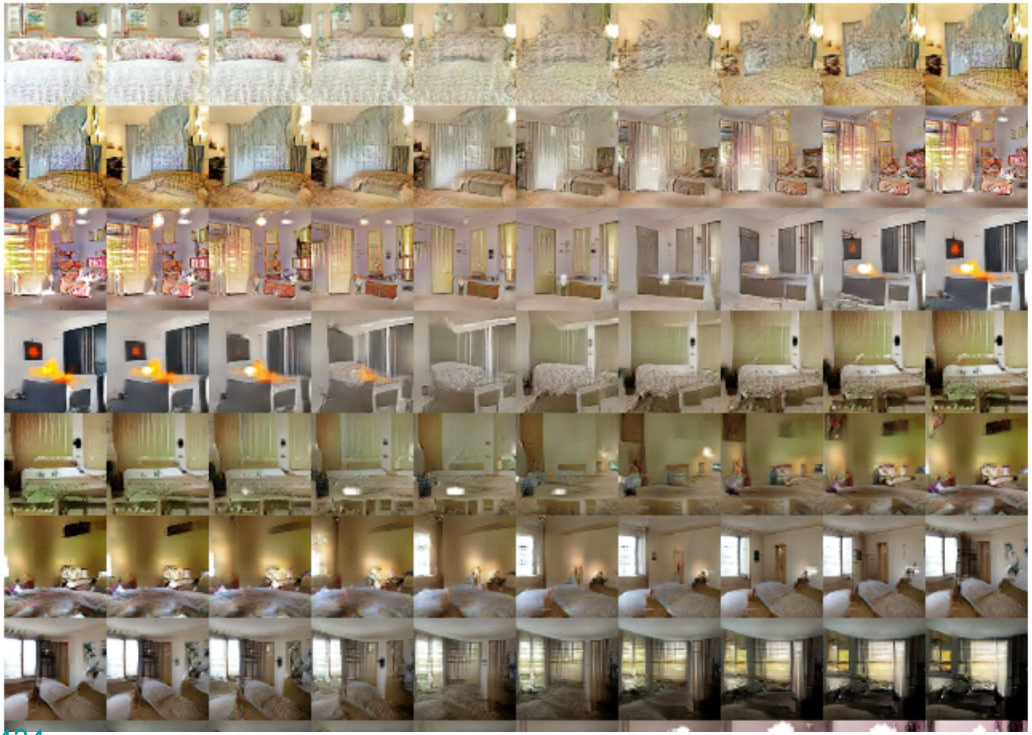
DCGAN. Radford et al., 2015. https://arxiv.org/abs/1511.06434

Generative Adversarial Networks (GAN)

Evolution: from MNIST to ImageNet, and beyond

DCGAN (deep convolutional GANs)

- simply use deep convnets for G and D
- importantly, **batch normalization** (Ioffe and Szegedy, 2015) helps to stabilize notoriously difficult learning process

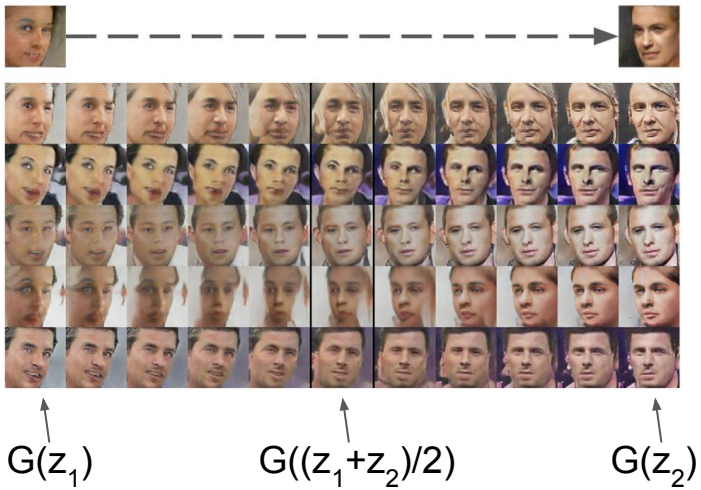


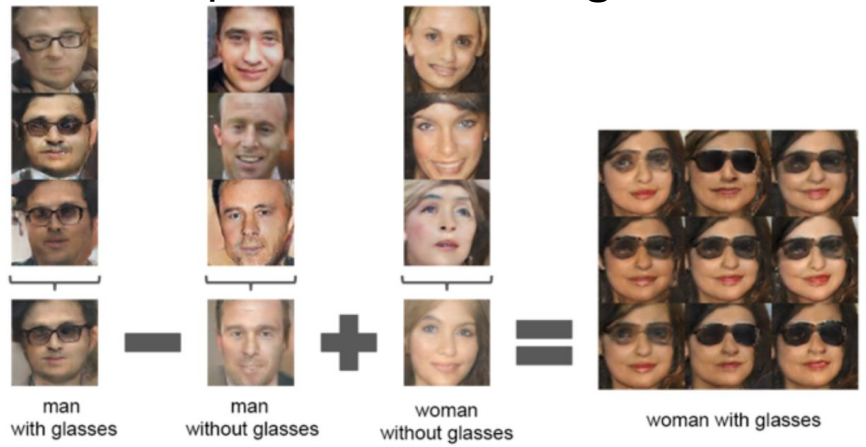
DCGAN. Radford et al., 2015. https://arxiv.org/abs/1511.06434

Generative Adversarial Networks (GAN)

Evolution: from MNIST to ImageNet, and beyond

DCGAN: observed that the generator's noise/latent space has **meaningful semantics**





DCGAN. Radford et al., 2015. https://arxiv.org/abs/1511.06434

Generative Adversarial Networks (GAN)
Evolution: from MNIST to ImageNet, and beyond

ACGAN: improved class-conditional discriminator with auxiliary classifier

$$L_S = E[\log P(S = real \mid X_{real})] + E[\log P(S = fake \mid X_{fake})]$$

$$L_C = E[\log P(C = c \mid X_{real})] + E[\log P(C = c \mid X_{fake})]$$

D: maximize $L_S + L_C$

G: maximize $L_C - L_S$

Trained on **ImageNet**:
1.28M images, 1000 categories

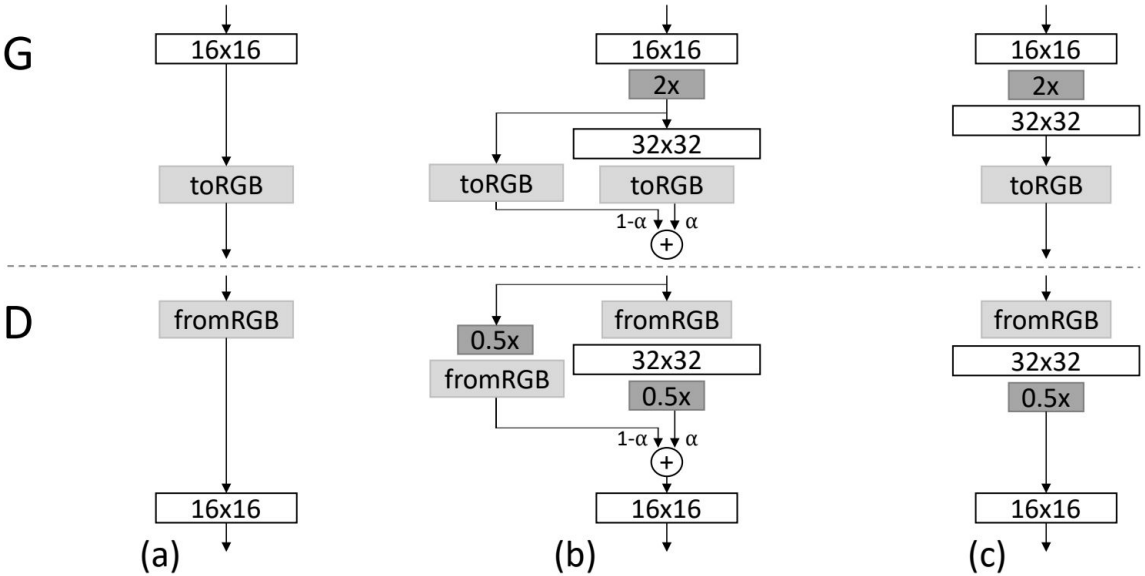
Odena et al., 2016. https://arxiv.org/abs/1610.09585

Generative Adversarial Networks (GAN)

Evolution: from MNIST to ImageNet, and beyond

Progressive GANs

- first, train a GAN to generate tiny (4x4) images
- after convergence, add a new layer to generate 8x8 resolution images, then 16x16, then 32x32, ..., repeat until 1024x1024
- compelling results in a restricted domain (faces)



Karras et al., 2018. https://arxiv.org/abs/1802.05637

Generative Adversarial Networks (GAN)
Evolution: from MNIST to ImageNet, and beyond

Progressive GANs

- first, train a GAN to generate tiny (4x4) images
- after convergence, add a new layer to generate 8x8 resolution images, then 16x16, then 32x32, ..., repeat until 1024x1024
- compelling results in a restricted domain (faces)

Karras et al., 2018. https://arxiv.org/abs/1802.05637

Generative Adversarial Networks (GAN)
Evolution: from MNIST to ImageNet, and beyond

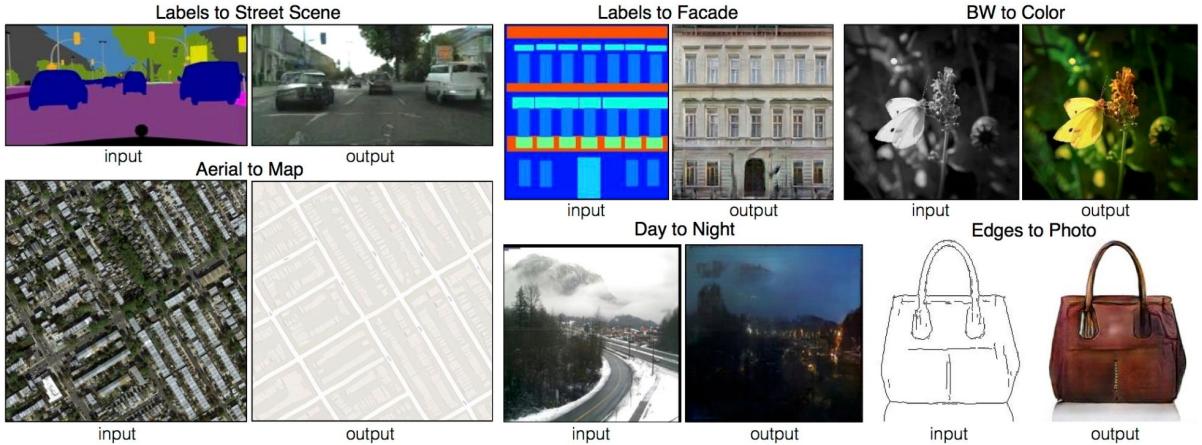
pix2pix

Given paired examples of images in two different domains, train a generator to translate between them

$$\mathcal{L}_{GAN}(G, D) = \mathbb{E}_y[\log D(y)] + \mathbb{E}_{x,z}[\log(1 - D(G(x, z))].$$

$$\mathcal{L}_{L1}(G) = \mathbb{E}_{x,y,z}[\|y - G(x, z)\|_1].$$

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



Example results on several image-to-image translation problems. In each case we use the same architecture and objective, simply training on different data.

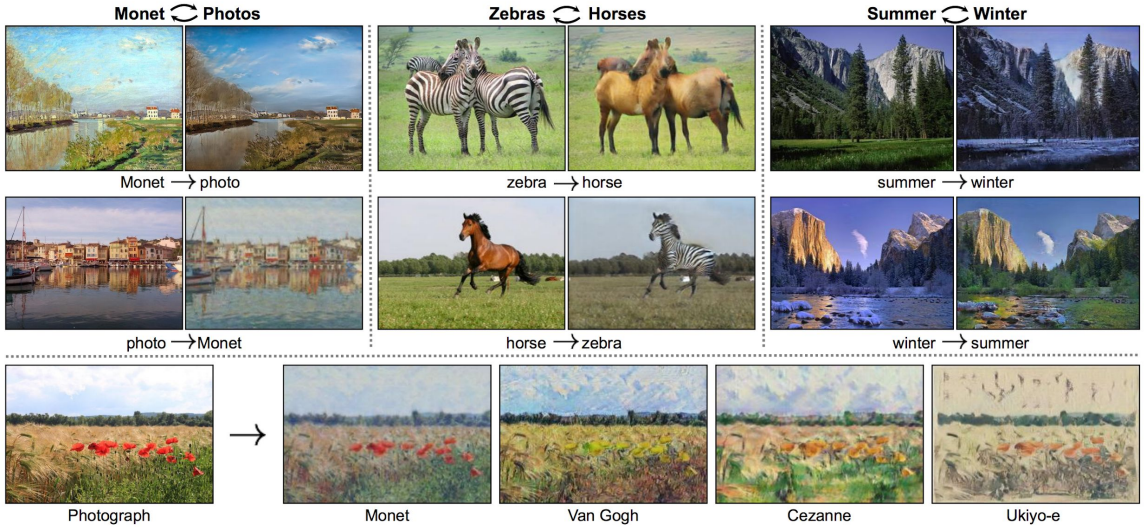
Isola et al., 2016. https://arxiv.org/abs/1611.07004

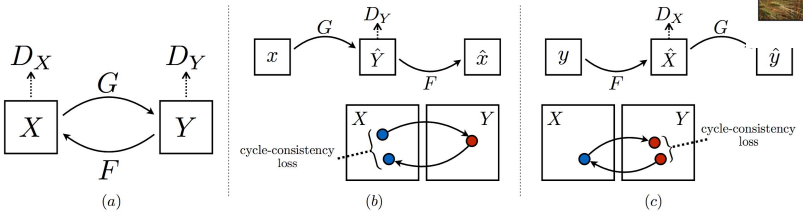
Generative Adversarial Networks (GAN)

Evolution: from MNIST to ImageNet, and beyond

CycleGAN

Given **unpaired** examples of images in two different domains, train a generator to translate between them





Zhu et al., 2017. https://arxiv.org/abs/1703.10593

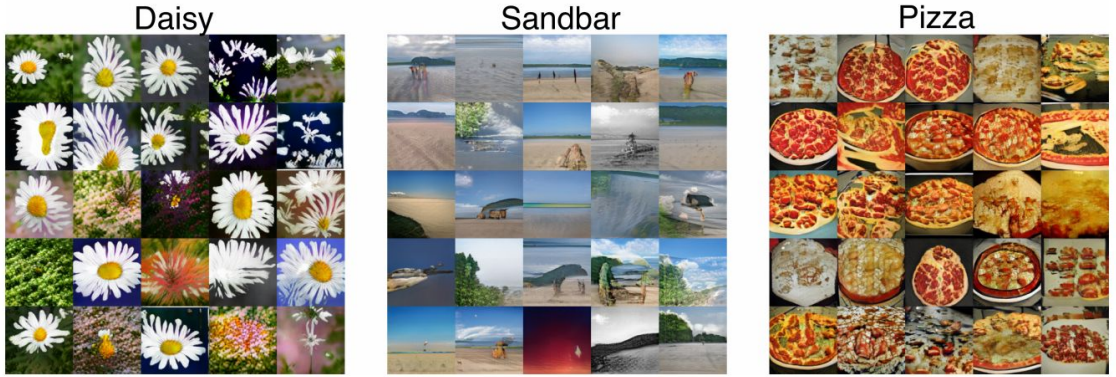
Generative Adversarial Networks (GAN)

Evolution: from MNIST to ImageNet, and beyond

SNGAN: stabilized training by clamping the singular values of D's weights to 1

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

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



Miyato et al., 2018. https://arxiv.org/abs/1802.05957

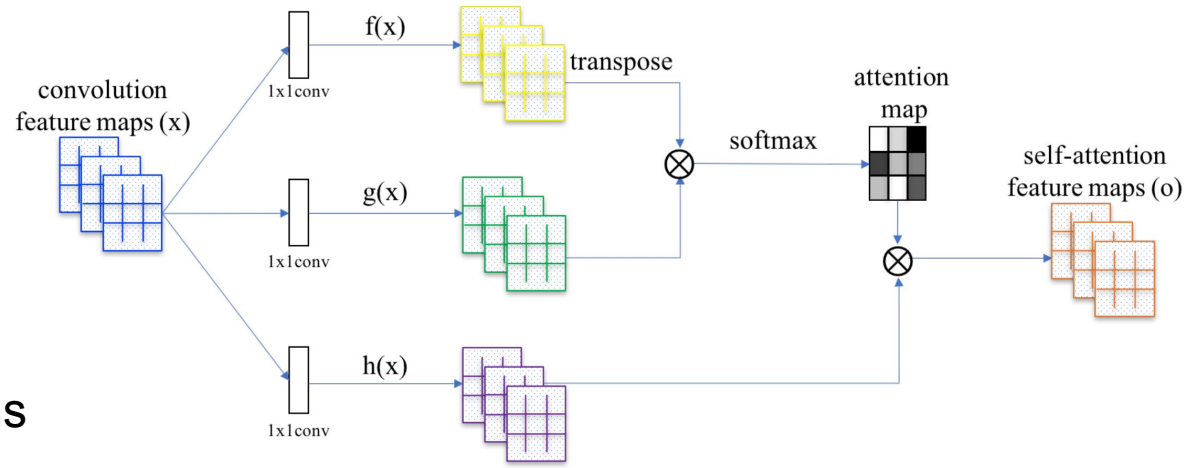
Generative Adversarial Networks (GAN)

Evolution: from MNIST to ImageNet, and beyond

Self-attention GAN (SAGAN)

- added **self-attention** to give images better **global structure** and coherence

- self-attention is a recent advance that has made a bit impact in many domains
  - especially language modeling, translation



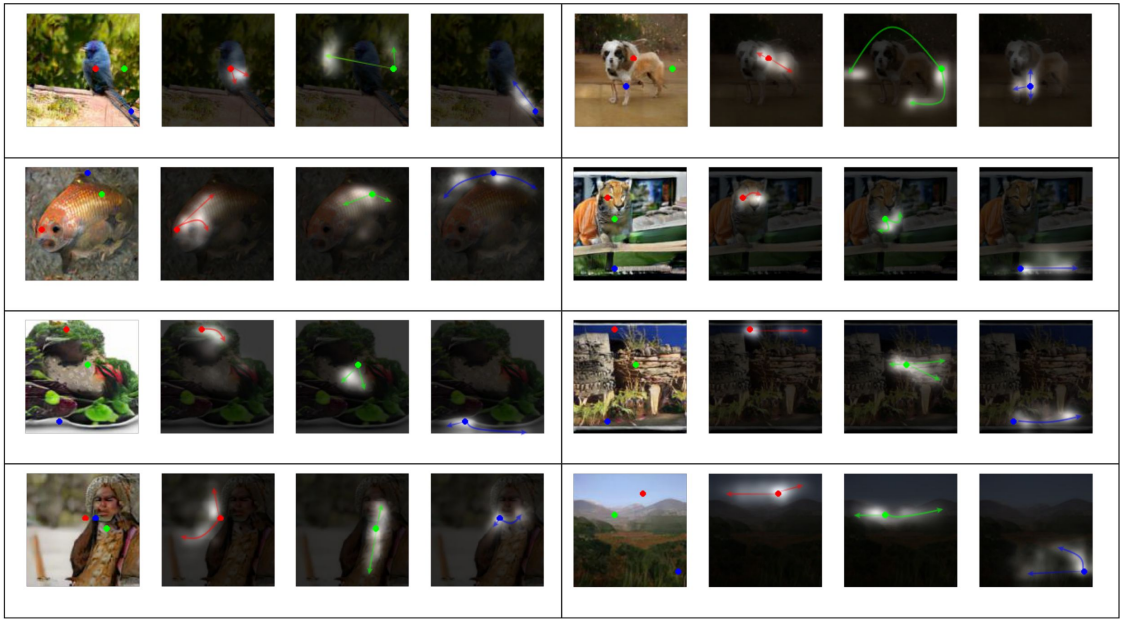
Zhang et al., 2018. https://arxiv.org/abs/1805.08318

Generative Adversarial Networks (GAN)
Evolution: from MNIST to ImageNet, and beyond

Self-attention GAN (SAGAN)

- added **self-attention** to give images better **global structure** and coherence

- self-attention is a recent advance that has made a bit impact in many domains
  - especially language modeling, translation



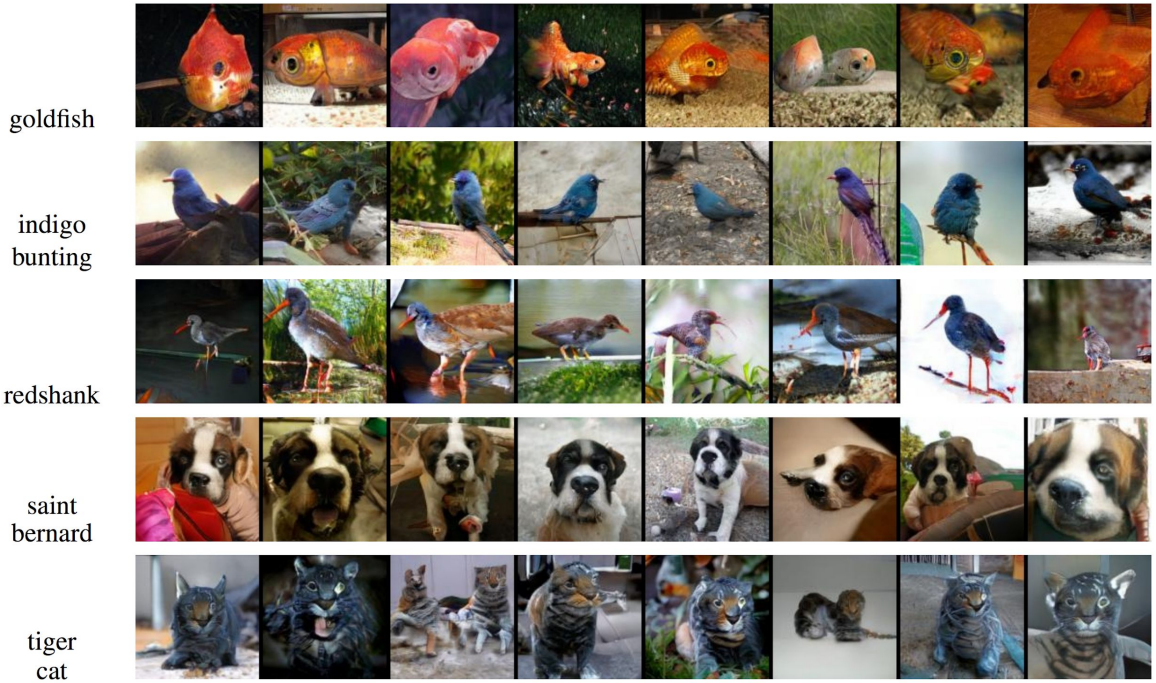
Zhang et al., 2018. https://arxiv.org/abs/1805.08318

Generative Adversarial Networks (GAN)
Evolution: from MNIST to ImageNet, and beyond

Self-attention GAN (SAGAN)

- added **self-attention** to give images better **global structure** and coherence

- self-attention is a recent advance that has made a bit impact in many domains
  - especially language modeling, translation



Zhang et al., 2018. https://arxiv.org/abs/1805.08318

BigGAN

Main idea: make GANs really, really big

- big **batches** (2048 images -- 8-32x larger than typical previous models)
- big **models**: first layer features in G are 4x4x1536 = ~24K
- big **datasets**:
  - ImageNet (1.28M images)
  - JFT (internal Google dataset, 300M images)
- big **resolution**: up to 512x512

Brock, Donahue, & Simonyan, 2018.
https://arxiv.org/abs/1809.11096

BigGAN

How did we get there?

- explored just about every trick from prior work, and kept the ones that worked
  - hinge loss
  - spectral normalization
  - self-attention
  - projection discriminator
- and added a few new ones
  - orthogonal regularization
  - “skip connections” from noise
  - shared class label embedding

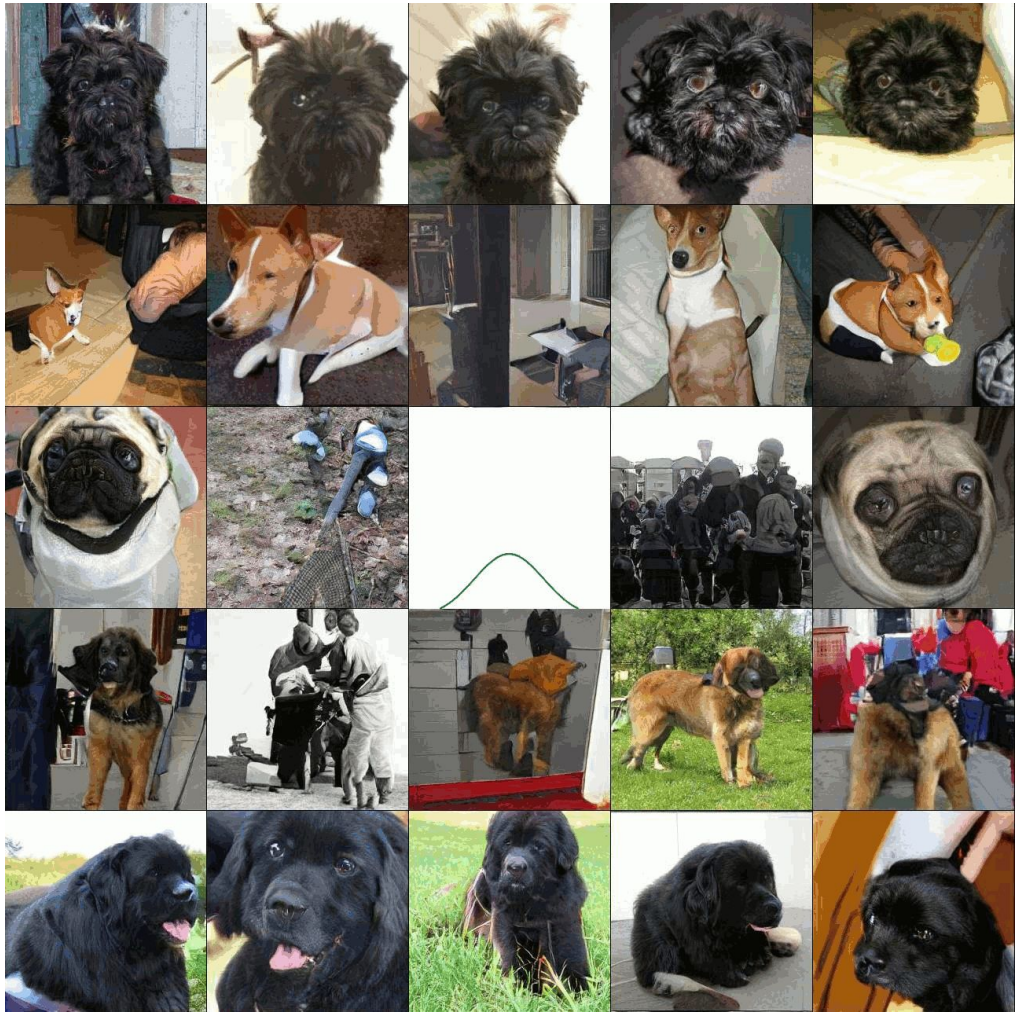



Brock, Donahue, & Simonyan, 2018.
https://arxiv.org/abs/1809.11096

BigGAN

The “Truncation Trick”

- Change the scale of the noise **z** input to the generator
- Make the noise **smaller** (truncate) to increase image **fidelity**
  - Generates prototypical examples of each class
- Make the noise **larger** to increase **variety**
  - Generates the full class distribution

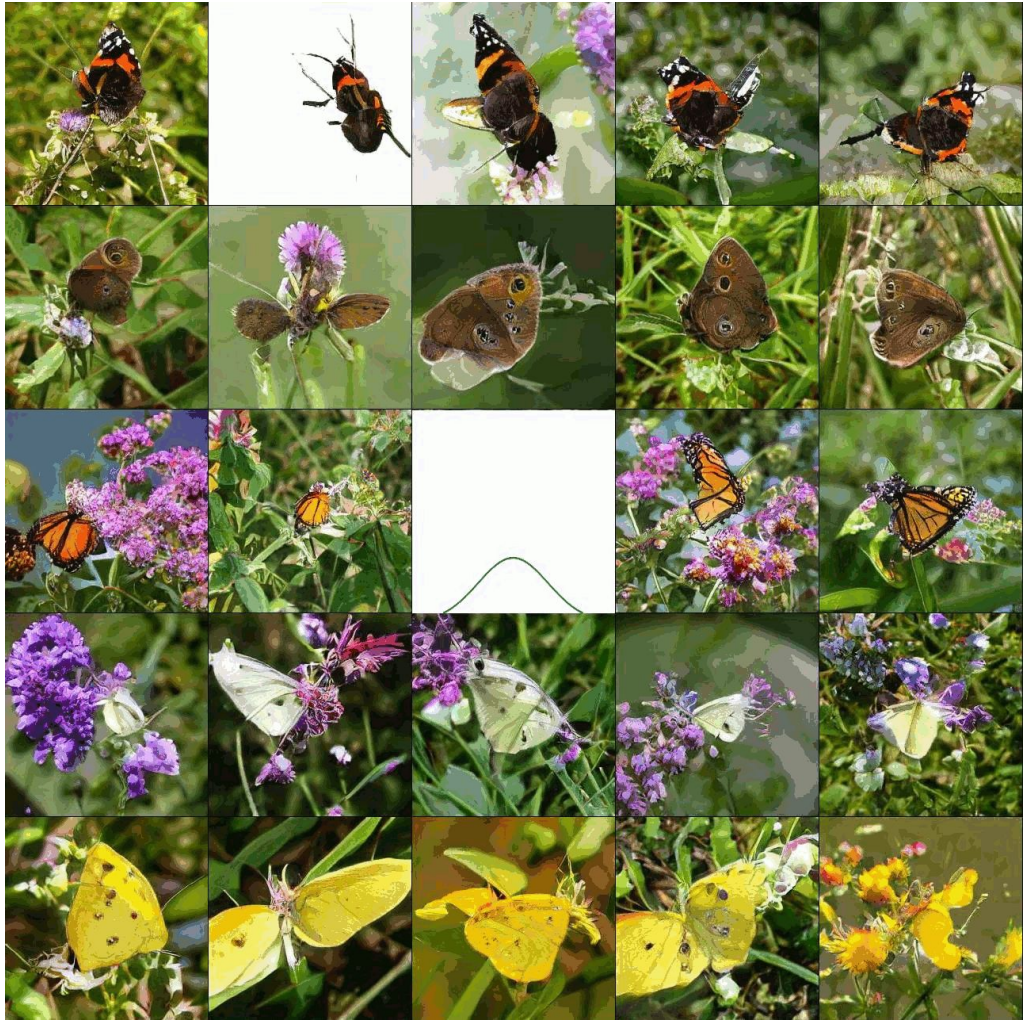


Brock, Donahue, & Simonyan, 2018.
https://arxiv.org/abs/1809.11096

BigGAN

The “Truncation Trick”

- Change the scale of the noise **z** input to the generator
- Make the noise **smaller** (truncate) to increase image **fidelity**
  - Generates prototypical examples of each class
- Make the noise **larger** to increase **variety**
  - Generates the full class distribution

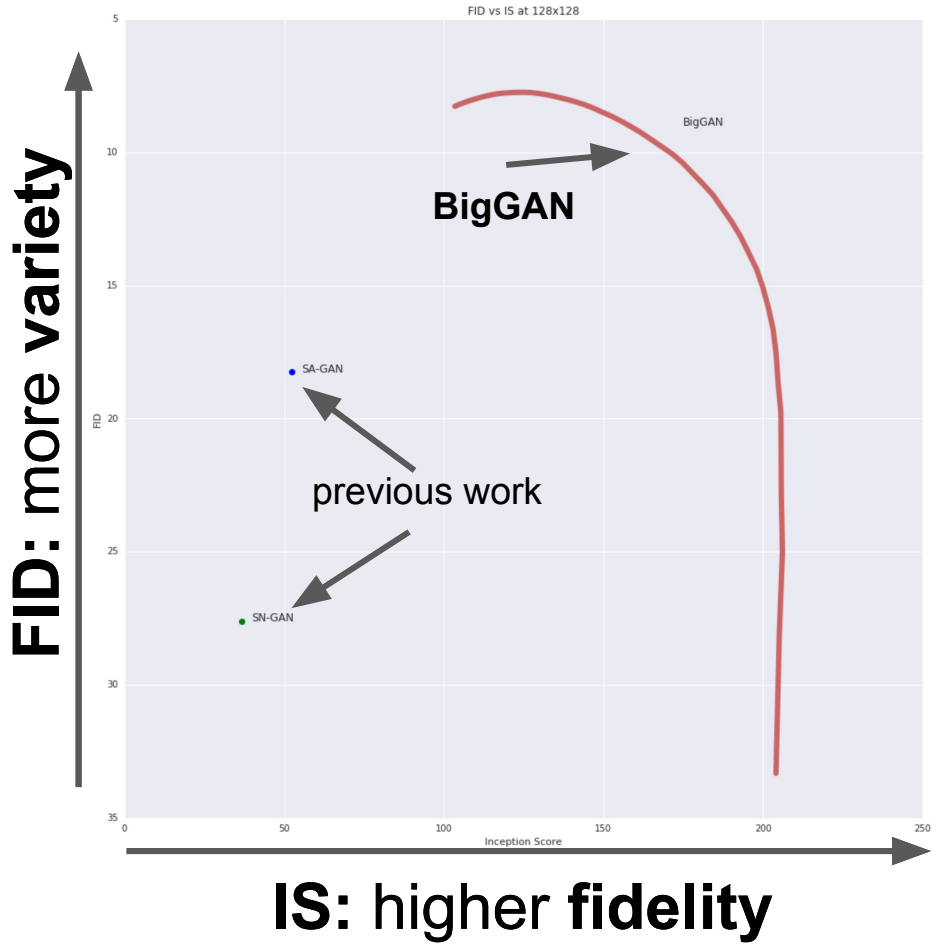


Brock, Donahue, & Simonyan, 2018.
https://arxiv.org/abs/1809.11096

BigGAN

Metrics: how do we know it works?

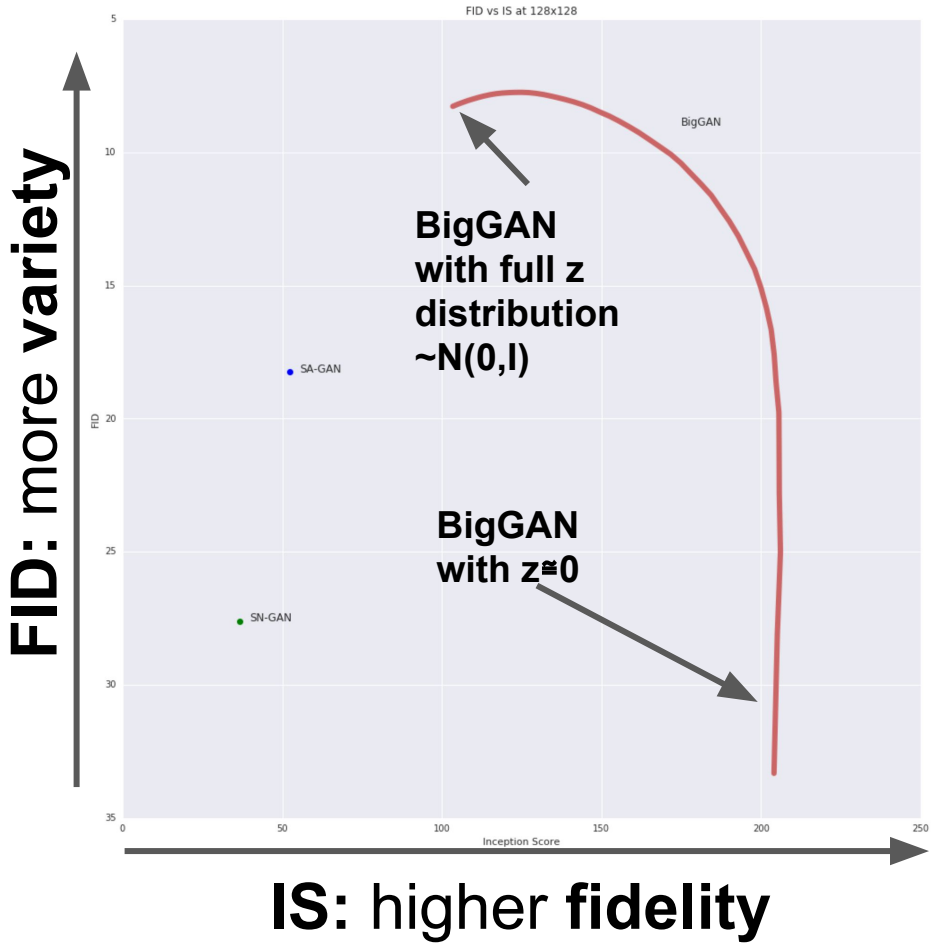
- GANs are difficult to evaluate
- Standard metrics use a **classifier** pretrained for image classification
- Inception Score (Salimans et al., 2016)
  - how **confident** is the classifier in its predictions for the generated images?
- Fréchet Inception Distance (FID) (Heusel et al., 2017)
  - how similar is the overall **distribution** of generated images to that of real images?



Brock, Donahue, & Simonyan, 2018.
https://arxiv.org/abs/1809.11096

BigGAN

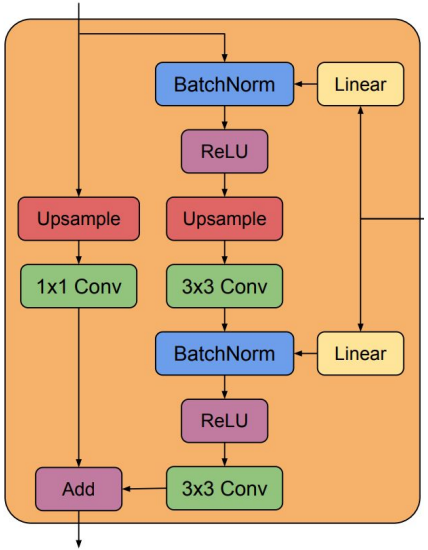
The **truncation trick** lets us trade-off between more fidelity (better IS) and more variety (better FID)



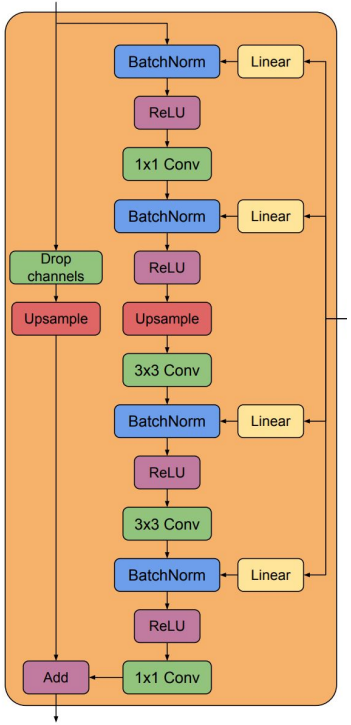
BigGAN-deep: latest results

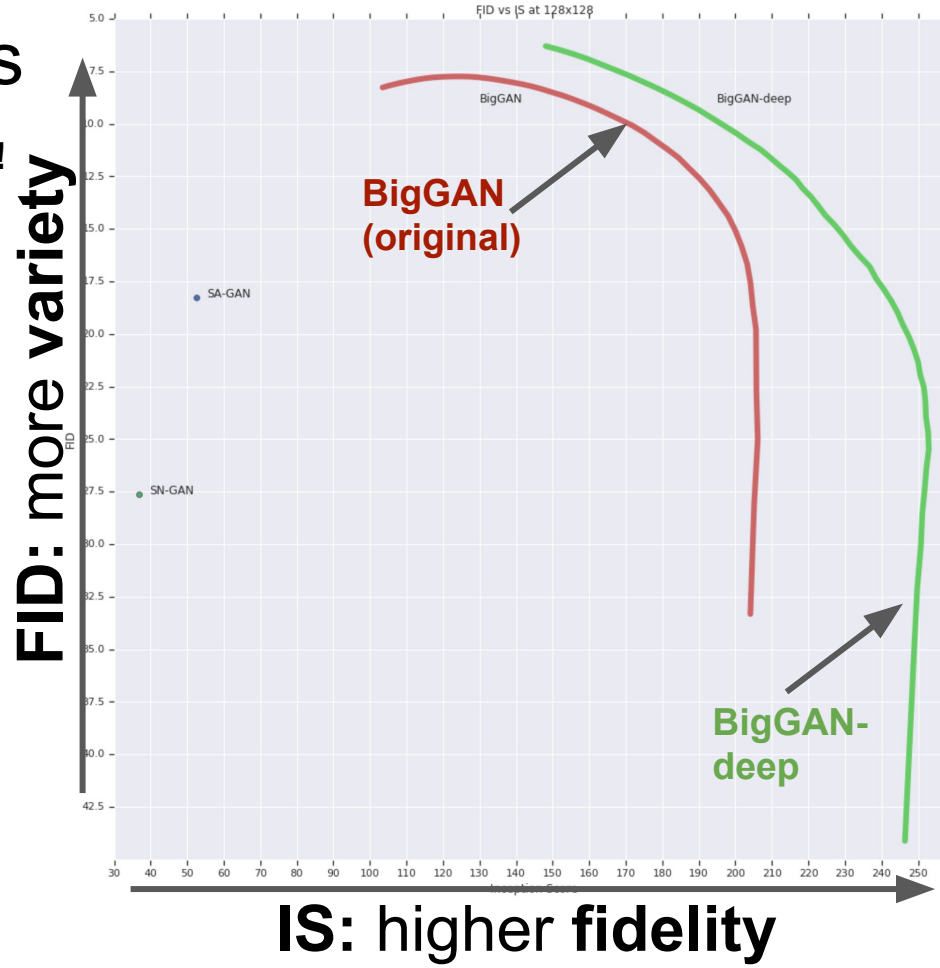
4x deeper, but more efficient!

BigGAN (original) Block



BigGAN-deep Block





Brock, Donahue, & Simonyan, 2018.
https://arxiv.org/abs/1809.11096

BigGAN: memorization?

Is it just memorizing the dataset? No!

For a given BigGAN generated image, we check the nearest neighbors -- the most similar images -- from the dataset it was trained to mimic.

We rarely/never see near-duplicates of images from the training set.

Brock, Donahue, & Simonyan, 2018.
https://arxiv.org/abs/1809.11096

BigGAN: memorization?

Is it just memorizing the dataset? No!

For a given BigGAN generated image, we check the nearest neighbors -- the most similar images -- from the dataset it was trained to mimic.

We rarely/never see near-duplicates of images from the training set.

BigGAN generated image

Nearest neighbors: most similar real images

Brock, Donahue, & Simonyan, 2018.
https://arxiv.org/abs/1809.11096

BigGAN: failure modes

Dogball

An example of “class leakage”

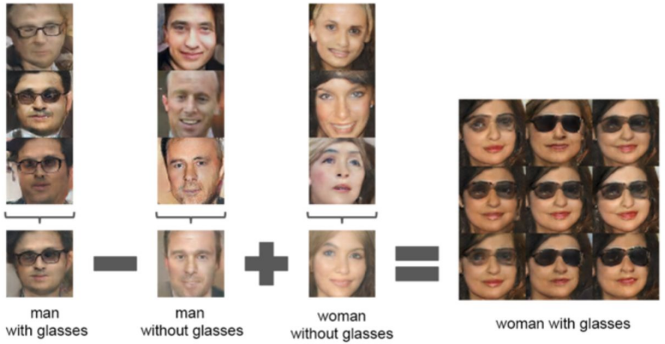
BigGAN: failure modes

Difficult classes:

- Complex structure
- Complex scenes
- Under-represented in dataset

Why train GANs? (Or any generative models?)

- Because it’s fun! (Reason enough?)
- If you can generate it, you understand it
- Planning for agents
  - GANs are particularly efficient samplers
  - Use to “imagine” the outcomes of different decisions and act accordingly
- Unsupervised learning
  - Latent space of a GAN encodes interesting semantic attributes with no labels
  - Learn from huge unlabeled databases?

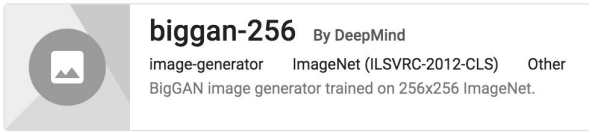


BigGAN: Colab demo (time permitting)

(1) Go to tfhub.dev

(2) Search for **biggan**

(3) Click on any of the 3 results (e.g., **biggan-256**)



(4) Click  (right side of the page)