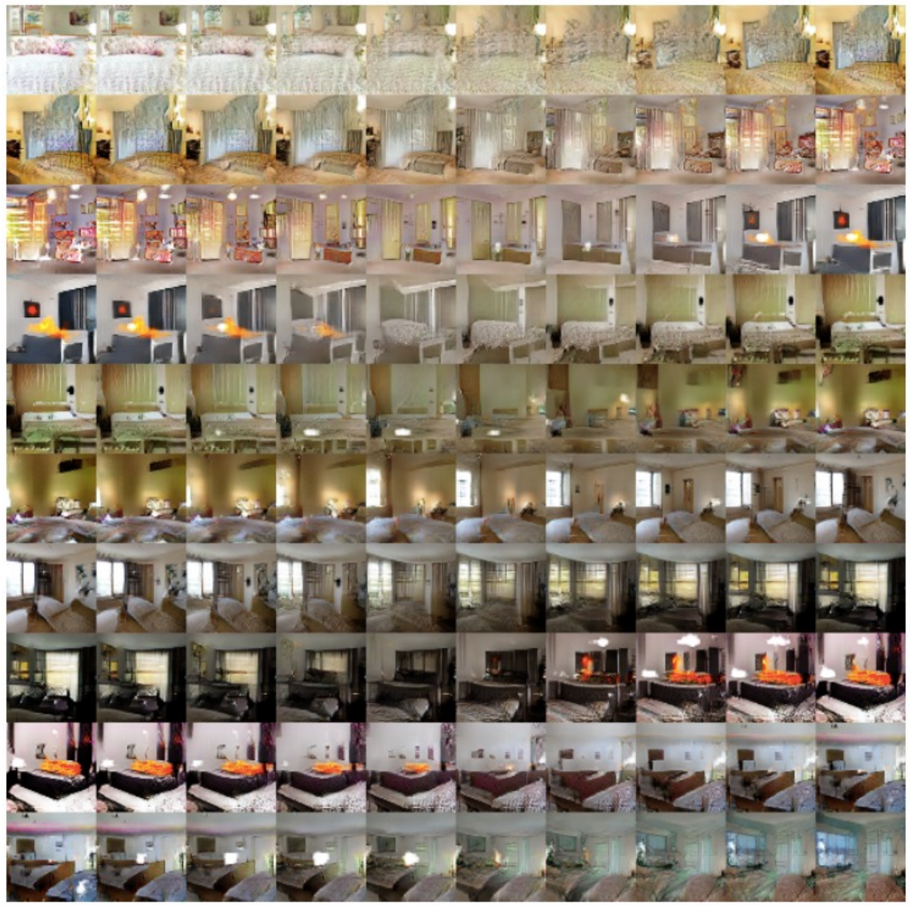
Generative adversarial networks: Introduction

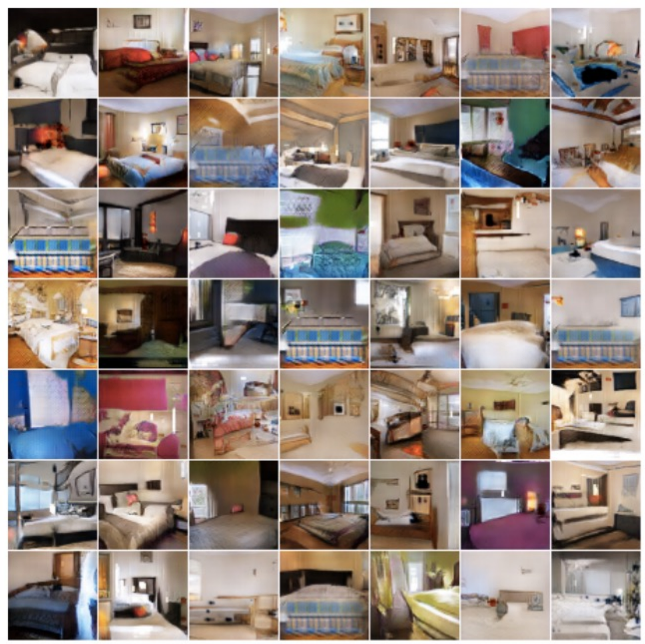


Outline

- Generative modeling tasks
- Original GAN formulation
- Alternative GAN objectives
- Evaluating GANs

Generative modeling tasks

- Generation: learn to sample from the distribution represented by the training set



Generative modeling tasks

- Generation conditioned on class label or text prompt

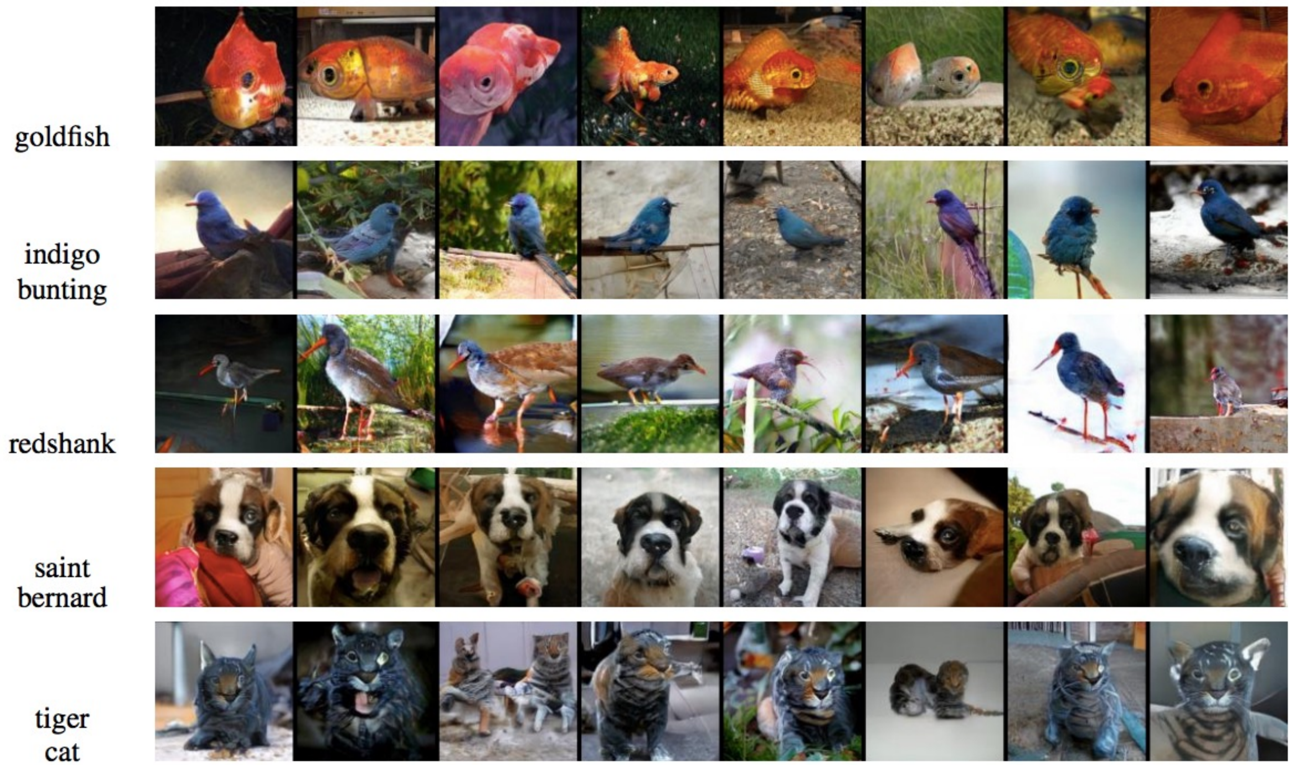
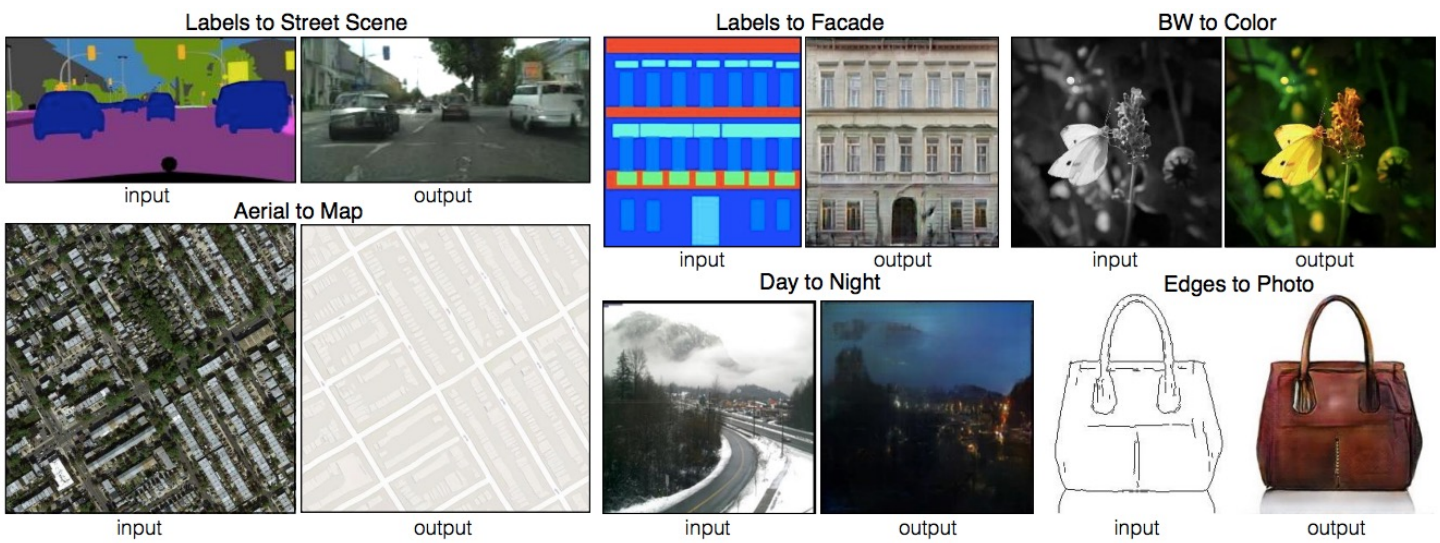


Figure source

Generative modeling tasks

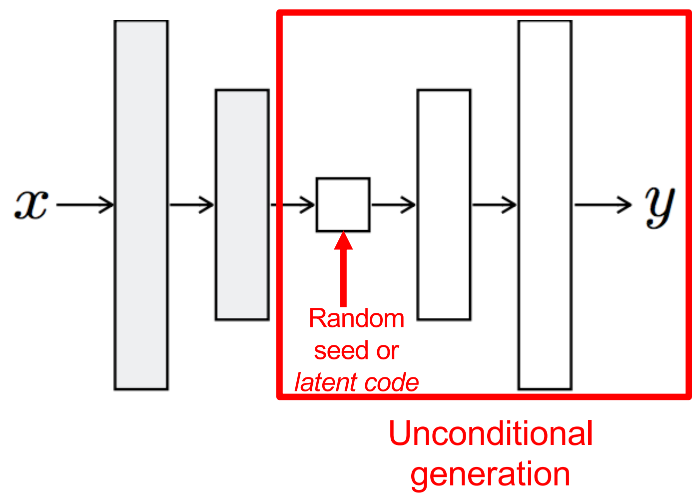
- Generation conditioned on image (*image-to-image translation*)



P. Isola, J.-Y. Zhu, T. Zhou, A. Efros, [Image-to-Image Translation with Conditional Adversarial Networks](), CVPR 2017

Designing a network for generative tasks

1. We need an architecture that can generate an image



Designing a network for generative tasks

1. We need an architecture that can generate an image

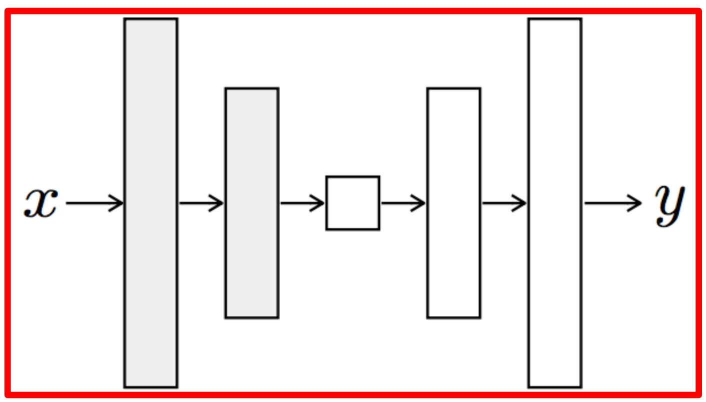
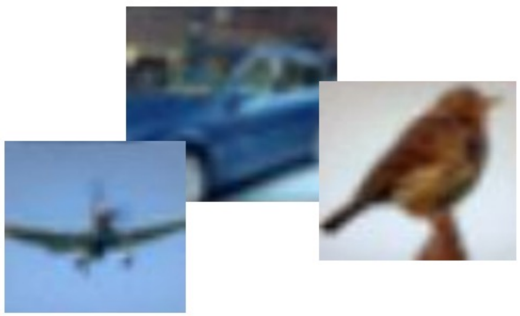


Image-to-image translation

Designing a network for generative tasks

1. We need an architecture that can generate an image
2. We need to design the right loss function and training framework

Learning to sample



Training data $x \sim p_{\text{data}}$

Generated samples $x \sim p_{\text{model}}$

We want to learn p_{model} that matches p_{data}

Adapted from [Stanford CS231n]

Generative adversarial networks

- Train two networks with opposing objectives:
  - **Generator:** learns to generate samples
  - **Discriminator:** learns to distinguish between generated and real samples

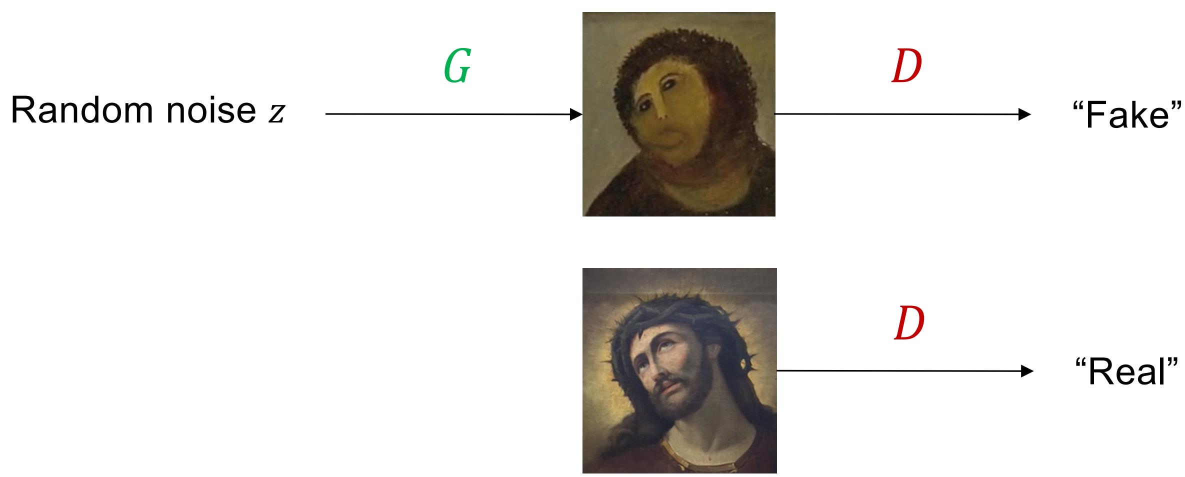


Figure adapted from F. Fleuret

I. Goodfellow et al. Generative adversarial nets. NeurIPS 2014

GAN objective

- The discriminator $D(x)$ should output the probability that the sample x is *real*
  - That is, we want $D(x)$ to be close to 1 for real data and close to 0 for fake
- According to the discriminator, the expected conditional log likelihood for real and generated data is given by

$$\mathbb{E}_{x\sim p_{\mathrm{data}}} \log D(x) \; + \; \mathbb{E}_{x\sim p_{\mathrm{gen}}} \log\big(1 - D(x)\big)$$

$$= \mathbb{E}_{x\sim p_{\mathrm{data}}} \log D(x) \; + \; \mathbb{E}_{z\sim p} \log(1 - D(G(z)))$$

We seed the generator with noise z drawn from a simple distribution p (Gaussian or uniform)

GAN objective

$$V(G, D) = \mathbb{E}_{x \sim p_{\text{data}}} \log D(x) + \mathbb{E}_{z \sim p} \log(1 - D(G(z)))$$

- The discriminator wants to correctly distinguish real and fake samples:

$$D^* = \arg\max_D V(G, D)$$

- The generator wants to fool the discriminator:

$$G^* = \arg\min_G V(G, D)$$

- We can try to train the generator and discriminator jointly in a *minimax game*

GAN objective: Theoretical properties

$$V(G, D) = \mathbb{E}_{x \sim p_{\text{data}}} \log D(x) + \mathbb{E}_{z \sim p} \log(1 - D(G(z)))$$

- Assuming unlimited capacity for generator and discriminator and unlimited training data:
  - The objective $\min_G \max_D V(G, D)$ is equivalent to *Jensen-Shannon divergence* between p_{data} and p_{gen} and global optimum (*Nash equilibrium*) is given by $p_{\text{data}} = p_{\text{gen}}$
  - If at each step, D is allowed to reach its optimum given G, and G is updated to decrease $V(G, D)$, then p_{gen} with eventually converge to p_{data}

Non-saturating GAN loss (NSGAN)

$$V(G, D) = \mathbb{E}_{x\sim p_{\text{data}}} \log D(x) + \mathbb{E}_{z\sim p} \log(1 - D(G(z)))$$

- Alternate between
  - *Gradient ascent* on discriminator:

$$D^* = \arg\max_D V(G, D)$$

  - *Gradient descent* on generator (minimize log-probability of generator samples being labeled “fake”):

$$G^* = \arg\min_G V(G, D)$$
$$= \arg\min_G \mathbb{E}_{z\sim p} \log(1 - D(G(z)))$$

  - In practice, do *gradient ascent* on generator (maximize log-probability of generator samples being labeled “real”):

$$G^* = \arg\max_G \mathbb{E}_{z\sim p} \log(D(G(z)))$$

Non-saturating GAN loss (NSGAN)

$$\min_{w_G} \mathbb{E}_{z\sim p} \log(1 - D(G(z))) \quad \text{vs.} \quad \max_{w_G} \mathbb{E}_{z\sim p} \log(D(G(z)))$$

Minimize log-probability of generator samples labeled “fake”

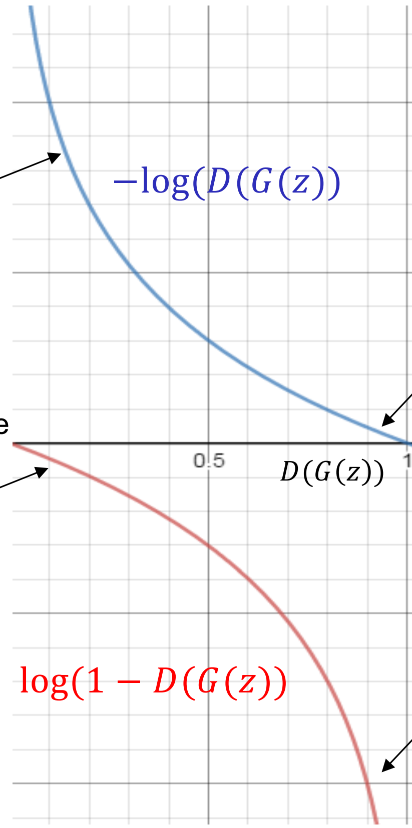
Maximize log-probability of generator samples labeled “real”

Non-saturating GAN loss (NSGAN)

$\min_{w_G} \mathbb{E}_{z \sim p} \log(1 - D(G(z)))$ vs. $\max_{w_G} \mathbb{E}_{z \sim p} \log(D(G(z)))$

Minimize log-probability of generator samples labeled “fake”

Maximize log-probability of generator samples labeled “real”



Large gradients for low-quality samples

Small gradients for high-quality samples

Low discriminator score (low-quality samples)

High discriminator score (high-quality samples)

Want to learn from confidently rejected sample but gradients here are small

These samples already fool the discriminator so we don’t need large gradients here

Figure source

GAN training in practice

- Update discriminator:
  - Repeat for k steps:
    - Sample mini-batch of noise samples $z_1, \dots, z_m$ and mini-batch of real samples $x_1, \dots, x_m$
    - Update parameters of D by stochastic gradient ascent on

$$\frac{1}{m}\sum_m [\log D(x_m) + \log(1 - D(G(z_m)))]$$

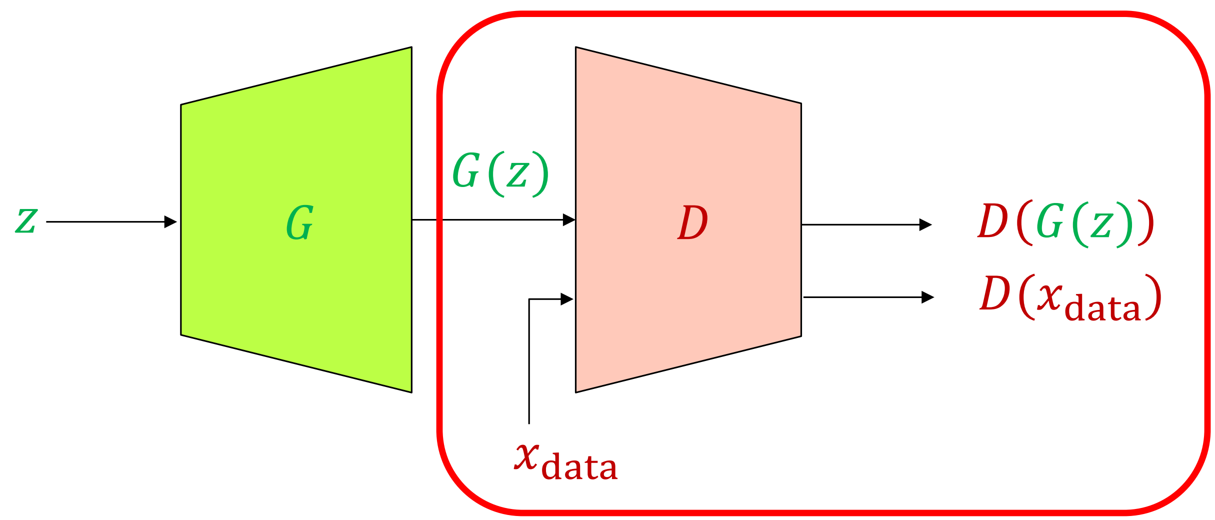
- Update generator:
  - Sample mini-batch of noise samples $z_1, \dots, z_m$
  - Update parameters of G by stochastic gradient ascent on

$$\frac{1}{m}\sum_m \log D(G(z_m))$$

- Repeat until happy with results

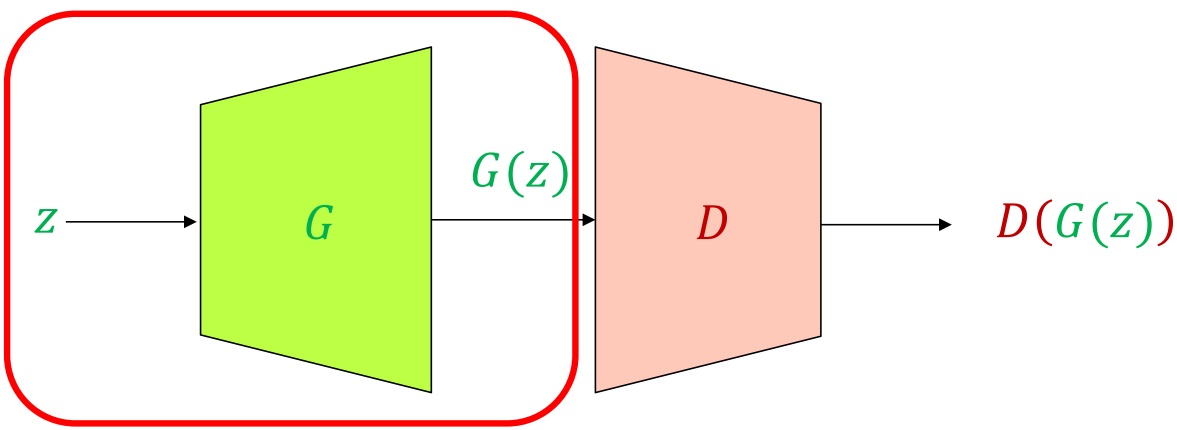
GAN: Schematic picture

- Update discriminator: push $D(x_{\text{data}})$ close to 1 and $D(G(z))$ close to 0
  - The generator is a "black box" to the discriminator



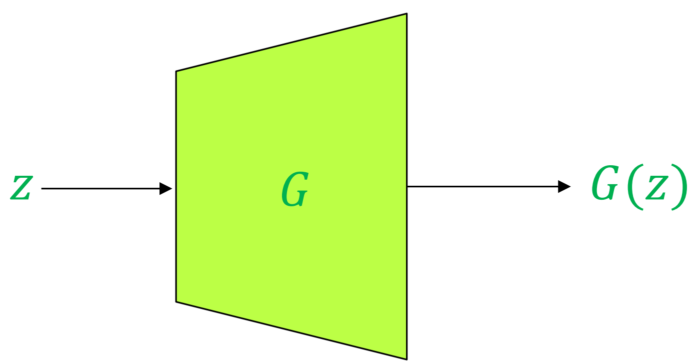
GAN: Schematic picture

- Update generator: increase $D(G(z))$
  - Requires back-propagating through the composed generator-discriminator network (i.e., the discriminator cannot be a black box)
  - The generator is exposed to real data only via the output of the discriminator *and its gradients*



GAN: Schematic picture

- Test time – the discriminator is discarded



Original GAN results

MNIST digits

Toronto Face Dataset

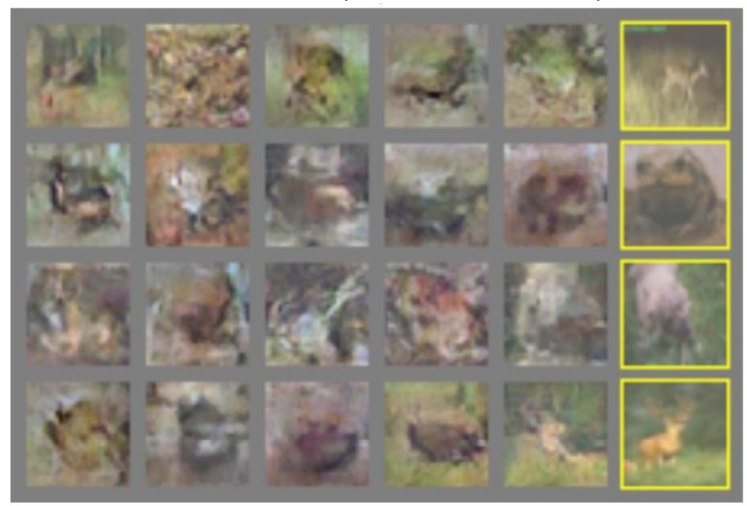
Nearest real image for
sample to the left

I. Goodfellow et al. Generative adversarial nets. NeurIPS 2014

Original GAN results

CIFAR-10 (FC networks)

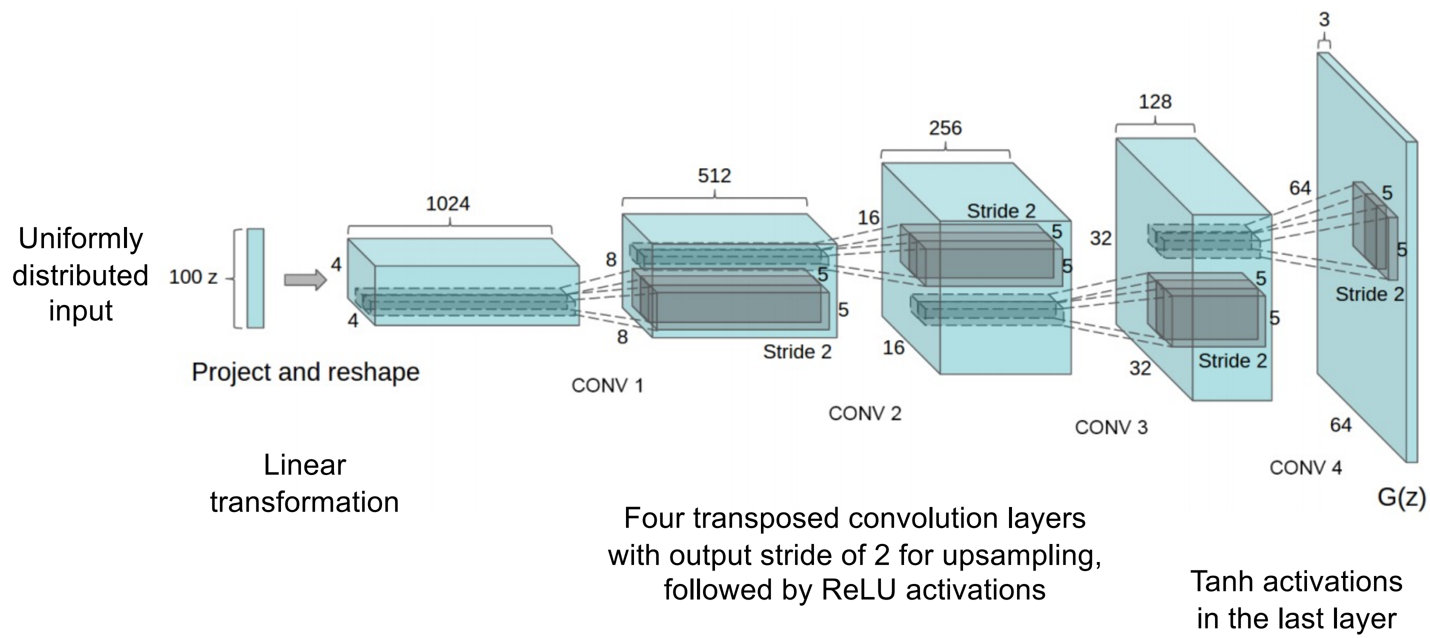
CIFAR-10 (conv networks)



I. Goodfellow et al. [Generative adversarial nets](). NeurIPS 2014

DCGAN

- Early, influential convolutional architecture for generator



A. Radford, L. Metz, S. Chintala, [Unsupervised representation learning with deep convolutional generative adversarial networks](), ICLR 2016

DCGAN

- Early, influential convolutional architecture for generator
- Discriminator architecture (empirically determined to give best training stability):
  - Don’t use pooling, only strided convolutions
  - Use Leaky ReLU activations (sparse gradients cause problems for training)
  - Use only one FC layer before the softmax output
  - Use batch normalization after most layers (in the generator also)

A. Radford, L. Metz, S. Chintala, [Unsupervised representation learning with deep convolutional generative adversarial networks](), ICLR 2016

DCGAN results

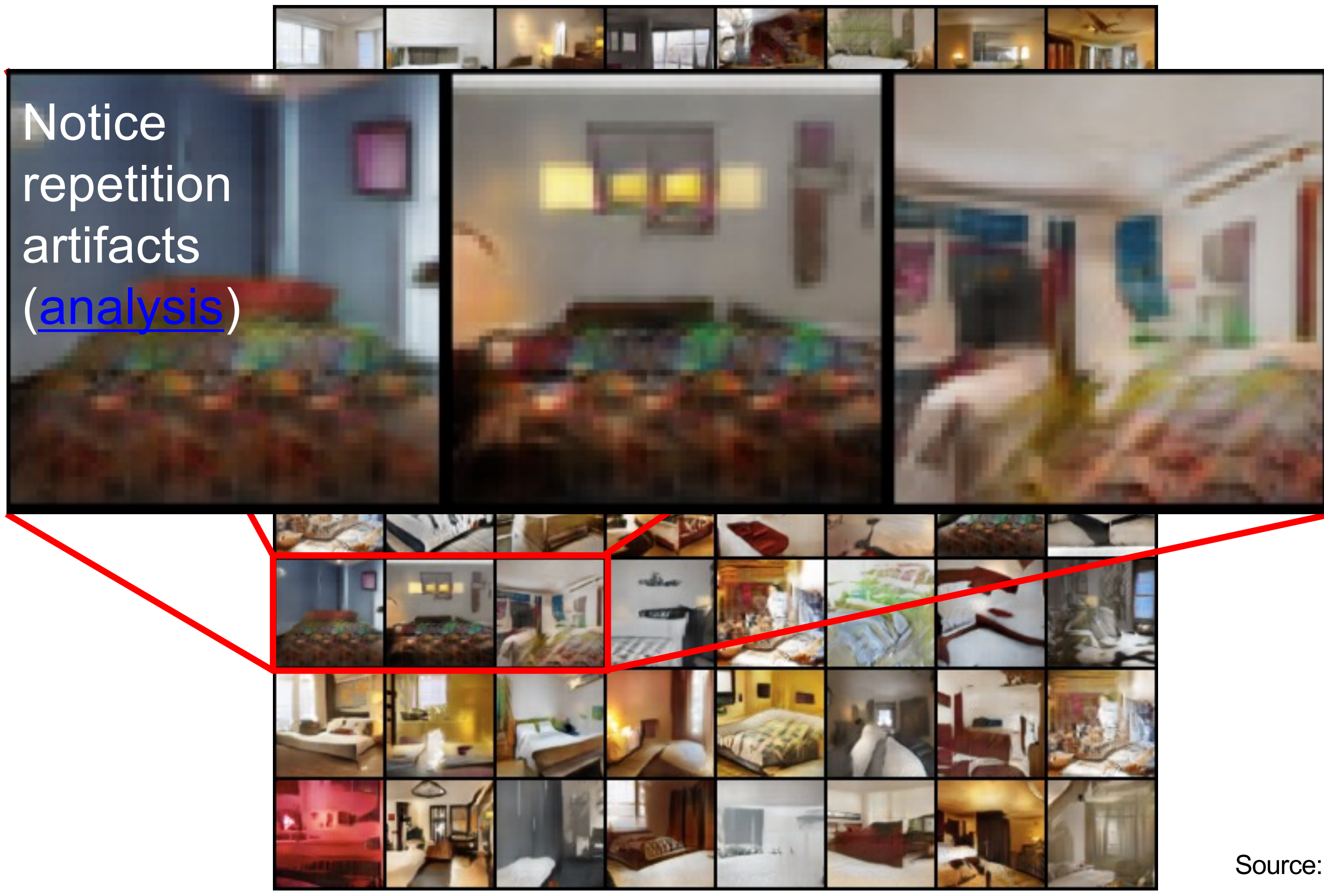
Generated bedrooms after one epoch

DCGAN results

Generated bedrooms after five epochs

DCGAN results

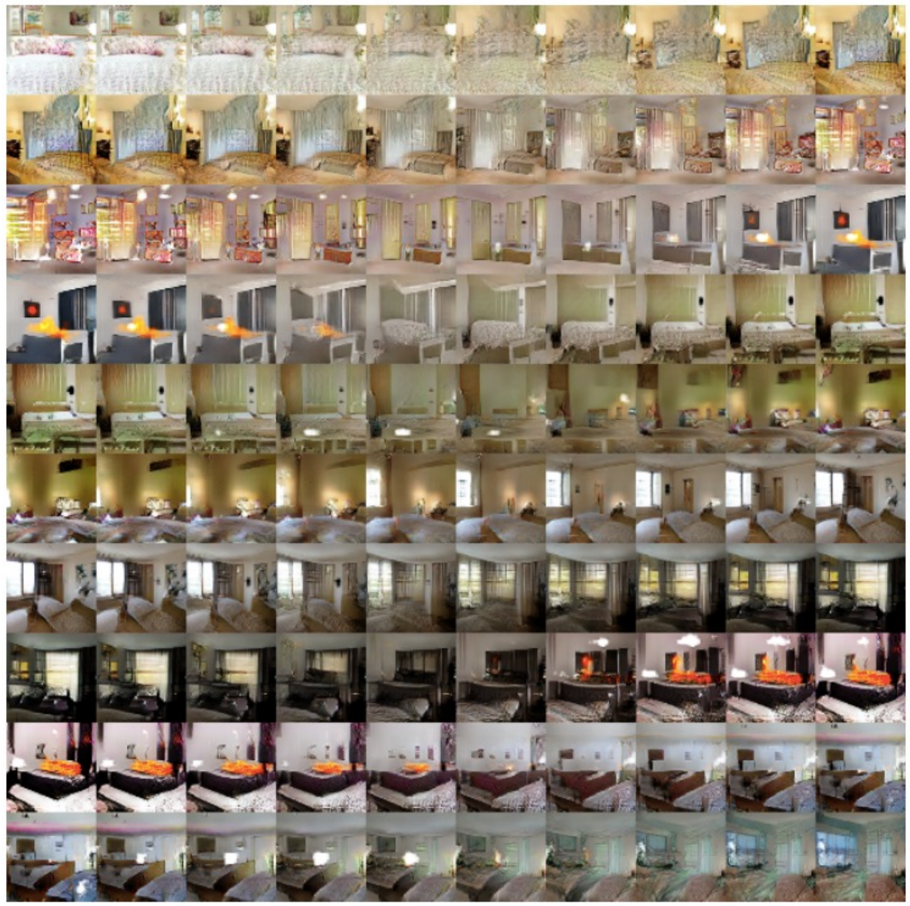
More bedrooms



Source: F. Fleuret

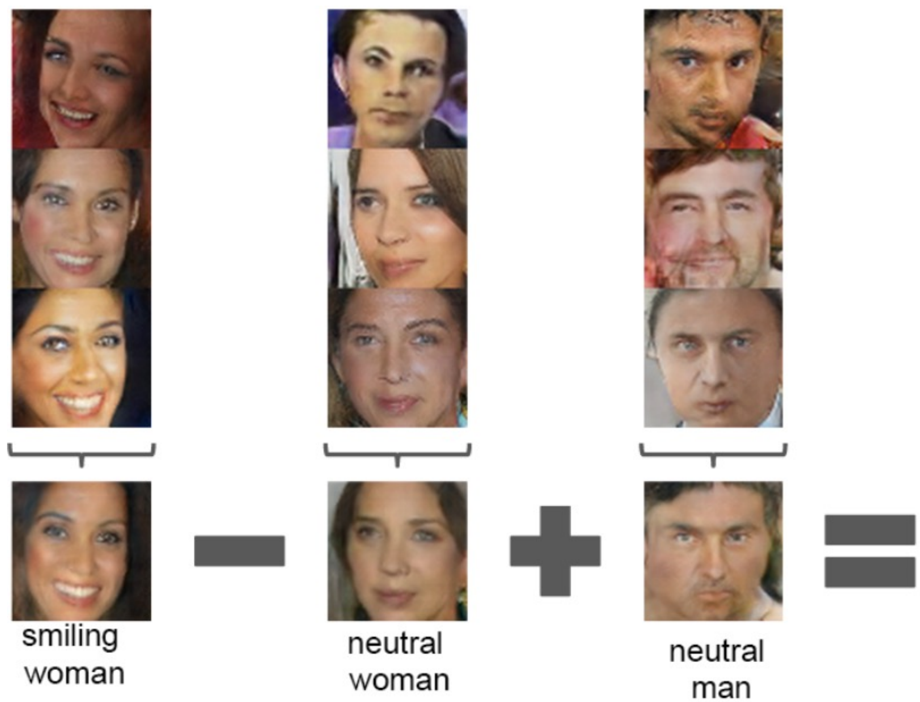
DCGAN results

Interpolation between different points in the z space



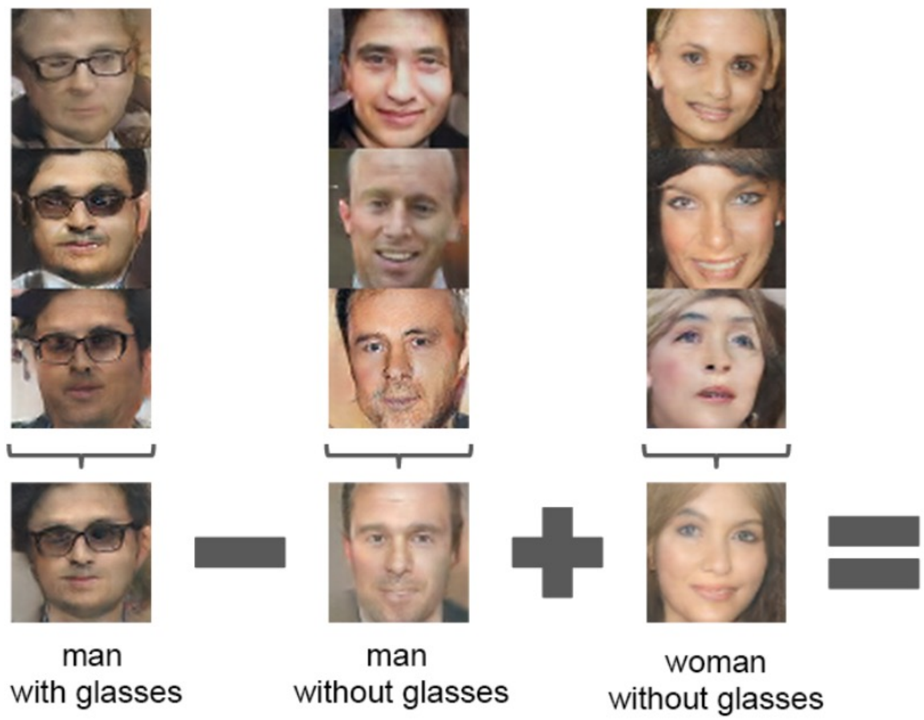
DCGAN results

- Vector arithmetic in the z space



DCGAN results

- Vector arithmetic in the z space



DCGAN results

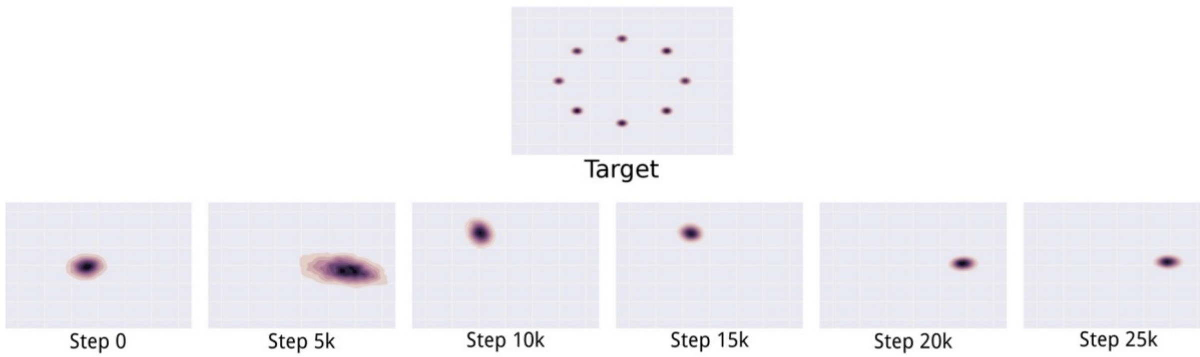
- Pose transformation by adding a “turn” vector

Problems with GAN training

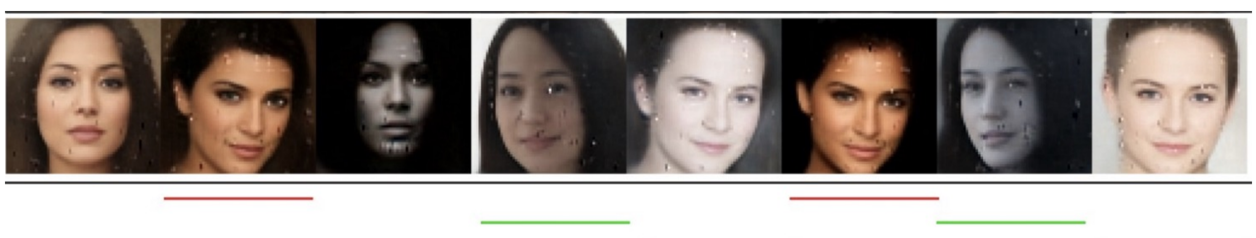
- Stability
  - Parameters can oscillate or diverge, generator loss does not correlate with sample quality
  - Behavior very sensitive to hyperparameter selection

Problems with GAN training

- Mode collapse
  - Generator ends up modeling only a small subset of the training data



Source



Source

Outline

- Generative modeling tasks
- Original GAN formulation
- Alternative GAN objectives

Wasserstein GAN (WGAN)

- Motivated by *Wasserstein or Earth mover's distance*, which is an alternative to JS divergence for comparing distributions
  - In practice, use linear activation instead of sigmoid in the discriminator and drop the logs from the objective:

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

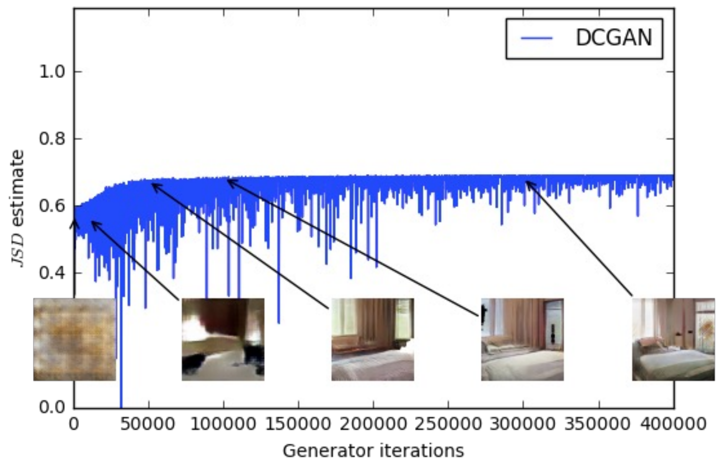
  - Due to theoretical considerations, important to ensure smoothness of discriminator
  - This paper's suggested method is clipping weights to fixed range $[-c, c]$

M. Arjovsky, S. Chintala, L. Bottou, Wasserstein generative adversarial networks, ICML 2017

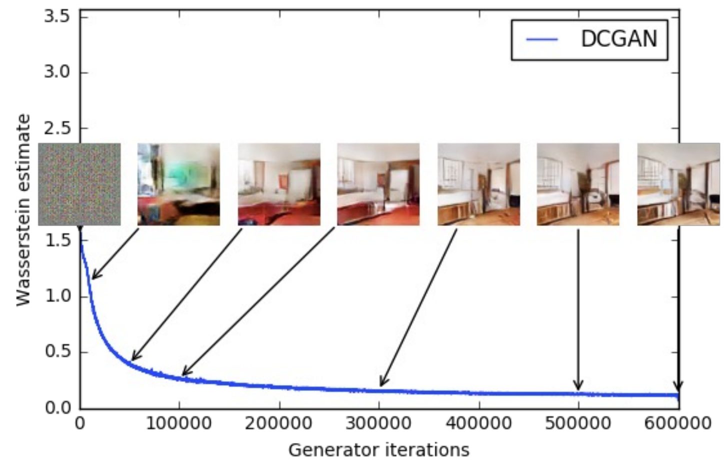
Wasserstein GAN (WGAN)

- Benefits (claimed)
  - Better gradients, more stable training
  - Objective function value is more meaningfully related to quality of generator output

Original GAN divergence



WGAN divergence



M. Arjovsky, S. Chintala, L. Bottou, [Wasserstein generative adversarial networks](), ICML 2017

Improved Wasserstein GAN (WGAN-GP)

- Weight clipping leads to problems with discriminator training
- Improved Wasserstein discriminator loss:

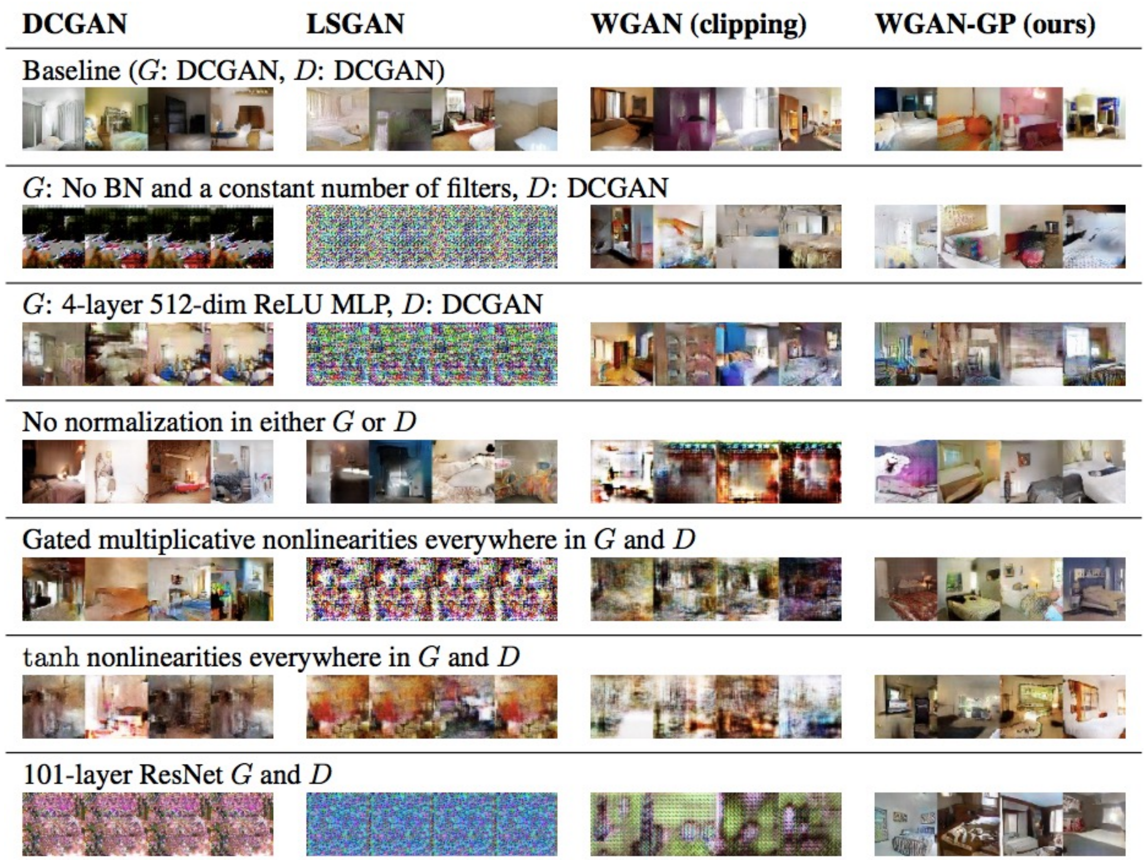
$$\mathbb{E}_{\tilde{x}\sim p_{\text{gen}}} D(\tilde{x}) - \mathbb{E}_{x\sim p_{\text{real}}} D(x)$$

$$+ \lambda\, \mathbb{E}_{\hat{x}\sim p_{\hat{x}}}\left[\left(\|\nabla_{\hat{x}} D(\hat{x})\|_2 - 1\right)^2\right]$$

Unit norm gradient penalty on points $\hat{x}$ obtained by interpolating real and generated samples

I. Gulrajani et al. Improved training of Wasserstein GANs. NeurIPS 2017

Improved Wasserstein GAN: Results



I. Gulrajani et al. Improved training of Wasserstein GANs. NeurIPS 2017

Least Squares GAN (LSGAN)

- Use least squares cost for generator and discriminator
  - Equivalent to minimizing Pearson χ^2 divergence

$$L_D = \mathbb{E}_{x\sim p_{\text{data}}}(D(x) - 1)^2 + \mathbb{E}_{z\sim p}(D(G(z)))^2$$

Push discrim. response on real data close to 1

Push response on generated data close to 0

$$L_G = \mathbb{E}_{z\sim p}(D\big(G(z)\big) - 1)^2$$

Push response on generated data close to 1

X. Mao et al. [Least squares generative adversarial networks](). ICCV 2017

Least Squares GAN (LSGAN)

- Benefits (claimed)
  - Higher-quality images

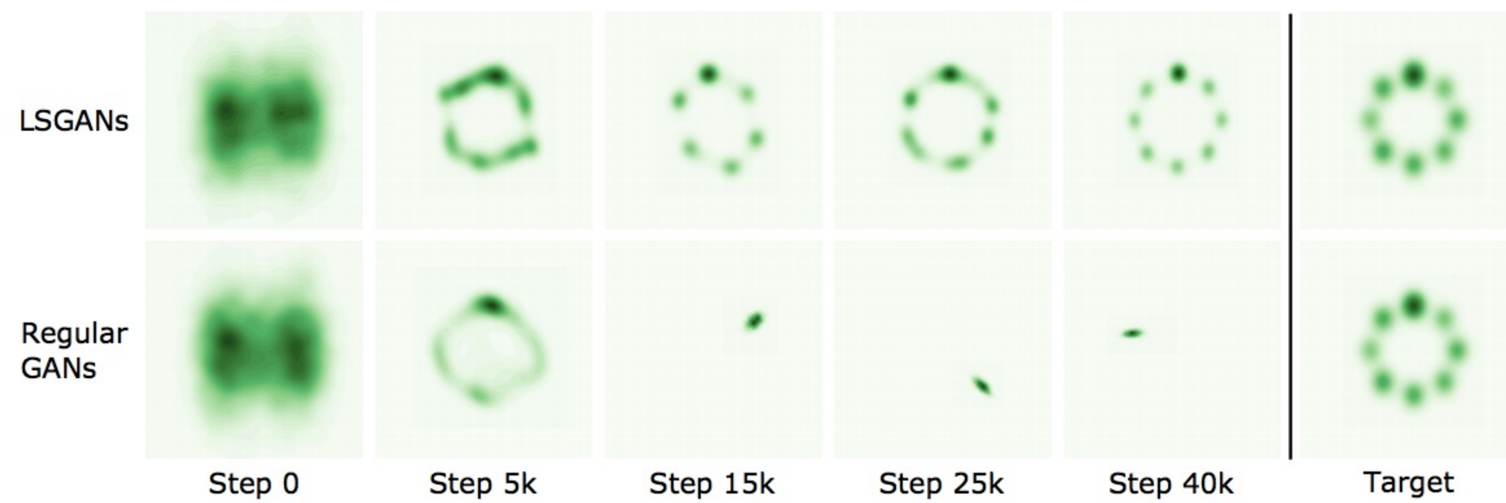
(a) Generated images (112 × 112) by LSGANs.

(b) Generated images (112 × 112) by DCGANs.

X. Mao et al. [Least squares generative adversarial networks.](#) ICCV 2017

Least Squares GAN (LSGAN)

- Benefits (claimed)
  - Higher-quality images
  - More stable and resistant to mode collapse



X. Mao et al. [Least squares generative adversarial networks.](#) ICCV 2017

GAN with hinge loss

- Discriminator: Drive discriminator score on real data above 1, on generated data below -1

$$L_D = -\mathbb{E}_{x\sim p_{\text{data}}}[\min(0, D(x) - 1)]$$
$$-\mathbb{E}_{z\sim p}\left[\min(0, -D\big(G(z)\big) - 1)\right]$$

- Generator: maximize discriminator score on generated data

$$L_G = -\mathbb{E}_{z\sim p} D(G(z))$$

T. Miyato et al. Spectral normalization for generative adversarial networks. ICLR 2018

Outline

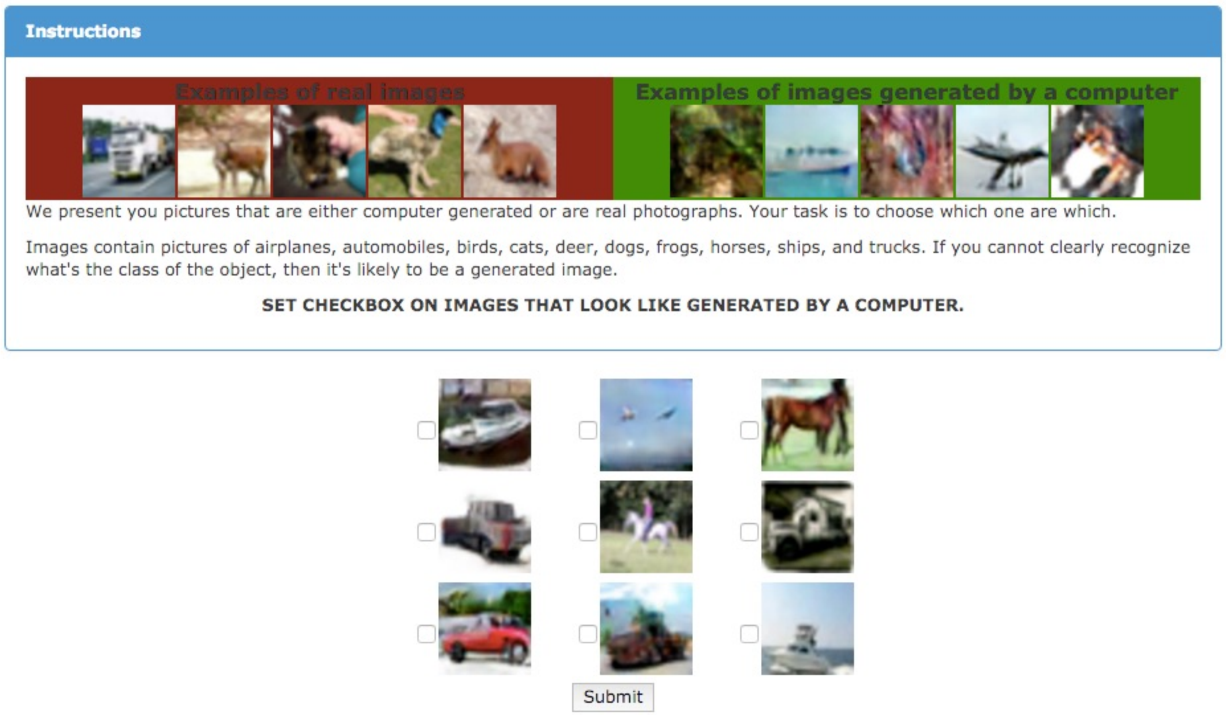
- Generative modeling tasks
- Original GAN formulations
- Alternative GAN objectives
- Evaluating GANs

How to evaluate GANs?

- Showing pictures of samples is not enough, especially for simpler datasets like MNIST, CIFAR, faces, bedrooms, etc.
- We cannot directly compute the likelihoods of high-dimensional samples (real or generated), or compare their distributions
- Many GAN approaches claim mainly to improve stability, which is hard to evaluate

GAN evaluation: Human studies

- Example: Turing test



T. Salimans et al. [Improved techniques for training GANs](). NeurIPS 2016

GAN evaluation: Inception score (IS)

- Key idea: generators should produce images with a variety of recognizable object classes
  - Pass generated samples x through an image classifier (InceptionNet), compute posterior class distributions $P(y|x)$ and marginal distribution $P(y)$
  - Compute Inception score as

$$IS(G) = \exp[\mathbb{E}_{x \sim G}\, KL(P(y|x) \parallel P(y))].$$

- IS should be high when:
  - Samples x contain recognizable objects, so entropy of $P(y|x)$ is *low*
  - The predicted labels of samples are diverse, so the entropy of $P(y)$ is *high*

T. Salimans et al. [Improved techniques for training GANs](). NeurIPS 2016

GAN evaluation: Inception score (IS)

- Disadvantages
  - A GAN that simply memorizes the training data (overfitting) or outputs a single image per class (mode dropping) could still score well
  - Is sensitive to network weights, not necessarily valid for generative models not trained on ImageNet, can be gamed (Barratt & Sharma 2018)

Figure 1. Sample of generated images achieving an Inception Score of 900.15. The maximum achievable Inception Score is 1000, and the highest achieved in the literature is on the order of 10.

GAN evaluation: Fréchet Inception Distance (FID)

- Key idea: fit simple distributions (Gaussians) to statistics of feature activations for real and generated data; estimate divergence parametrically
  - Pass generated samples through a network (InceptionNet), compute activations for a chosen layer
  - Estimate multivariate mean and covariance of activations, compute *Fréchet distance* to those of real data
- Advantages: correlated with visual quality of samples and human judgment, can detect mode dropping (unlike IS)
- Disadvantages: cannot detect overfitting (like IS), can be sensitive to resampling and compression (Parmar et al. 2021)

M. Heusel et al. GANs trained by a two time-scale update rule converge to a local Nash equilibrium, NeurIPS 2017

Are GANs created equal?

- From the abstract:

  *"We find that most models can reach similar scores with enough hyperparameter optimization and random restarts. This suggests that improvements can arise from a higher computational budget and tuning more than fundamental algorithmic changes … **We did not find evidence that any of the tested algorithms consistently outperforms the non-saturating GAN introduced in Goodfellow et al. (2014)**"*

M. Lucic et al. Are GANs created equal? A large-scale study. NeurIPS 2018