



LG Advanced Data Scientists Program

Deep Learning

[7: Generative Adversarial Networks]

Prof. Sungroh Yoon

Electrical & Computer Engineering | Seoul National University

© 2020 Sungroh Yoon. this material is for educational uses only. some contents are based on the material provided by other paper/book authors and may be copyrighted by them.

(last compiled at 22:51:00 on 2020/02/25)

Outline

Generative Adversarial Networks

Introduction

Training

Historical Developments

Challenges and Analyses

Summary

Appendix: Wasserstein GAN

References

- *Deep Learning* by Goodfellow, Bengio and Courville [▶ Link](#)
 - ▶ Chapter 20: Deep Generative Models
- survey:
 - ▶ *How GANs and Their Variants Work* [▶ Link](#)
- online resources:
 - ▶ *Stanford CS231n: CNN for Visual Recognition* [▶ Link](#)
 - ▶ *CVPR 2018 GAN Tutorial* [▶ Link](#)
 - ▶ *NIPS 2016 GAN Tutorial* [▶ Link](#)
 - ▶ *GAN Blog at Medium* [▶ Link](#)
- Wasserstein GAN
 - ▶ *ICML 2017 paper* [▶ Link](#)
 - ▶ *Read-through: Wasserstein GAN* [▶ Link](#)
 - ▶ *From GAN to WGAN* [▶ Link](#)

Outline

Generative Adversarial Networks

Introduction

Training

Historical Developments

Challenges and Analyses

Summary

Appendix: Wasserstein GAN

Comparison

- **autoregressive** models: define a tractable density function
 - ▶ optimize likelihood of training data:

$$p_{\theta}(\mathbf{x}) = \prod_{i=1}^n p_{\theta}(x_i | x_1, \dots, x_{i-1}) \quad (1)$$

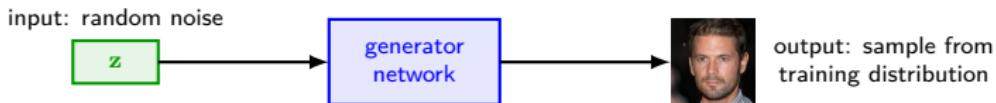
- **VAEs** define an intractable density function with latent \mathbf{z} :

$$p_{\theta}(\mathbf{x}) = \int p_{\theta}(\mathbf{z}) p_{\theta}(\mathbf{x} | \mathbf{z}) d\mathbf{z} \quad (2)$$

- ▶ cannot optimize (2) directly
- ⇒ derive and optimize a lower bound on the likelihood instead
- what if we give up on explicitly modeling density
 - ▶ and just want ability to _____?

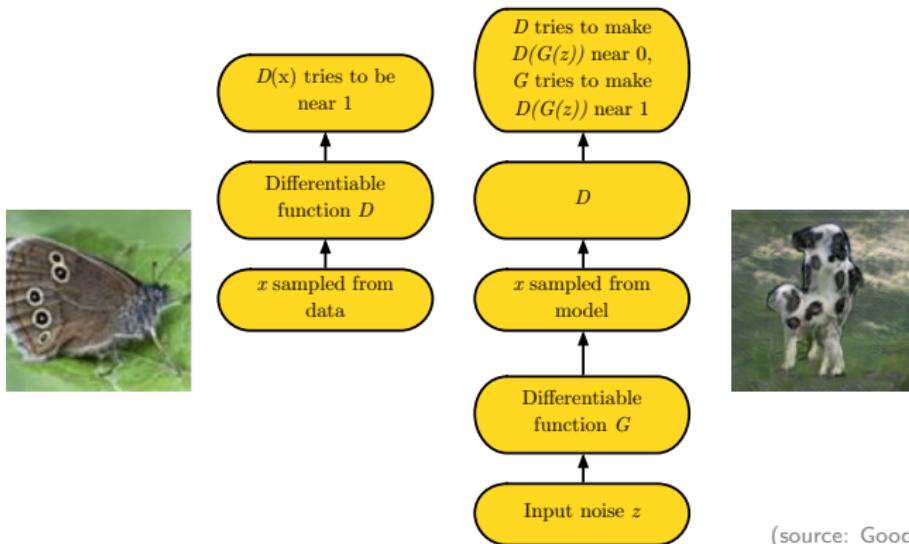
Main problem and solution

- problem:
 - ▶ want to sample from a complex/high-dimensional training distribution
 - ▶ directly doing this: very challenging (if not impossible)
- a solution: two-step approach
 1. sample from a _____ distribution (*e.g.* random noise)
 2. learn a transformation to the training distribution
 - ↑
how to represent this complex transformation?
 - ▶ use a neural net



Generative adversarial networks (GANs)

- do not work with any explicit density function
 - ▶ instead, take a _____ approach
 - ▶ learn to generate from training distribution through **2-player game**



(source: Goodfellow, 2018)

- GAN: based on a game-theoretic scenario
 - ▶ generator network must compete against an adversary (= discriminator)
- generator network
 - ▶ directly produces samples $\tilde{x} = G(z; \theta_g) = G_{\theta_g}(z)$
- its adversary: **discriminator network**
 - ▶ attempts to distinguish between
 - ▷ samples drawn from training data
 - and
 - ▷ samples drawn from _____
 - ▶ emits a probability value given by $\underbrace{D(x; \theta_d)}_{\uparrow} = D_{\theta_d}(x)$

↑
probability that x is a real training example
rather than a fake sample drawn from the model

Related idea

- self-play
 - ▶ **adversarial players** improve together through self-play training

1959: Arthur Samuel's checkers agent



(Silver et al, 2017)



(OpenAI, 2017)

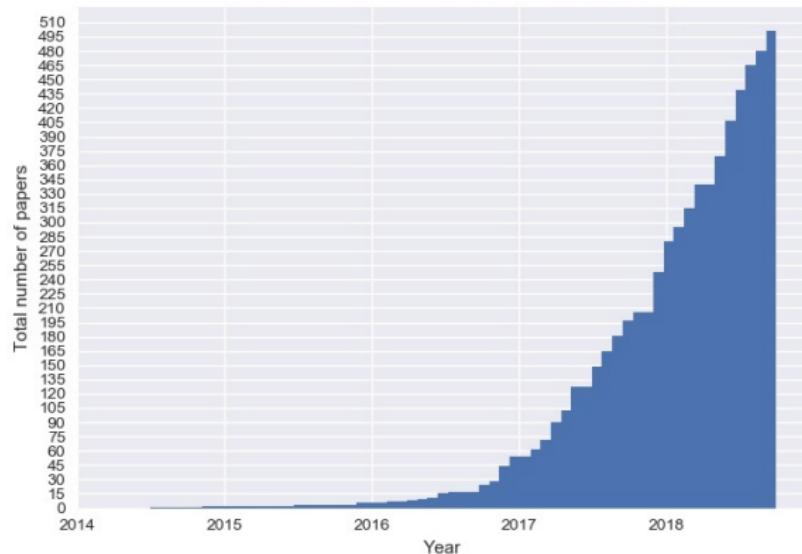


(Bansal et al, 2017)

(source: Goodfellow, 2018)

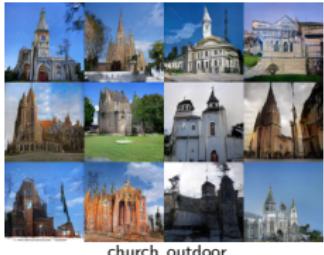
Explosion of GAN applications

- cumulative number of named GAN papers by month [Link](#)



(source: The GAN Zoo)

- better training and sample generation



LSGAN¹



WGAN², improved WGAN³



progressive GAN⁴

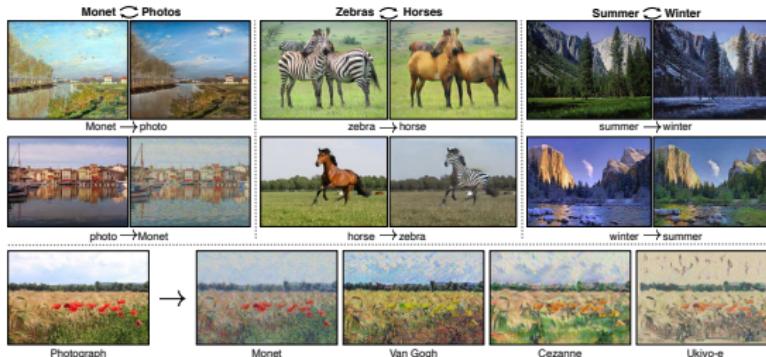
¹ Mao, X., Li, Q., Xie, H., Lau, R. Y., Wang, Z., and Smolley, S. P. (2017). Least squares generative adversarial networks. In *Computer Vision (ICCV), 2017 IEEE International Conference on*, pages 2813–2821. IEEE

² Arjovsky, M., Chintala, S., and Bottou, L. (2017). Wasserstein generative adversarial networks. In *International conference on machine learning*, pages 214–223

³ Gulrajani, I., Ahmed, F., Arjovsky, M., Dumoulin, V., and Courville, A. C. (2017). Improved training of wasserstein gans. In *Advances in neural information processing systems*, pages 5767–5777

⁴ Karras, T., Aila, T., Laine, S., and Lehtinen, J. (2017). Progressive growing of gans for improved quality, stability, and variation. *arXiv preprint arXiv:1710.10196*

- source → target domain transfer: CycleGAN⁵



- text → image synthesis⁶

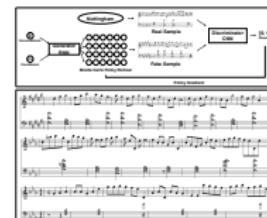
the flower has petals that are bright pinkish purple with white stigma



this white and yellow flower have thin white petals and a round yellow stamen



- music generation⁷



⁵ Zhu, J.-Y., Park, T., Isola, P., and Efros, A. A. (2017). [Unpaired image-to-image translation using cycle-consistent adversarial networks](#). *arXiv preprint arXiv:1703.10021*

⁶ Reed, S., Akata, Z., Yan, X., Logeswaran, L., Schiele, B., and Lee, H. (2016). [Generative adversarial text to image synthesis](#). *arXiv preprint arXiv:1605.05396*

⁷ Lee, S.-g., Hwang, U., Min, S., and Yoon, S. (2017). [A seqgan for polyphonic music generation](#). *arXiv preprint arXiv:1710.11418*

Outline

Generative Adversarial Networks

Introduction

Training

Historical Developments

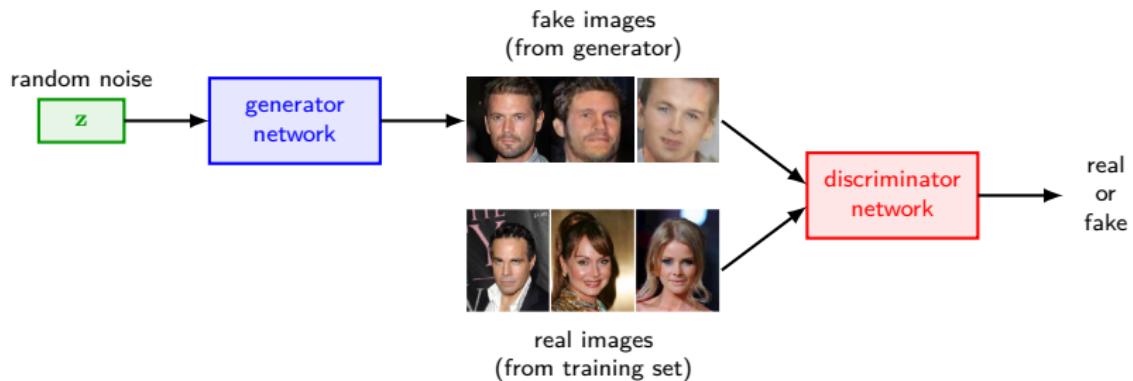
Challenges and Analyses

Summary

Appendix: Wasserstein GAN

Training GANs

- two-player game
 - ▶ **generator**: try to fool discriminator by generating real-looking images
 - ▶ **discriminator**: try to distinguish between real and ___ images



Training objectives

- assume:
 - discriminator D outputs the likelihood of input being ___ (range: [0, 1])
- **discriminator:** maximize J_D wrt θ_d

$$\begin{aligned} J_D &= \mathbb{E}_{x \sim p_{\text{data}}} \log \overbrace{D_{\theta_d}(x)}^{\text{how likely } x \text{ is real}} + \mathbb{E}_{z \sim p(z)} \overbrace{\log(1 - D_{\theta_d}(G_{\theta_g}(z)))}^{\text{how likely } G(z) \text{ is fake}} \\ &\approx \underbrace{\frac{1}{m} \sum_{i=1}^m \log D_{\theta_d}(x^{(i)})}_{m \text{ real samples}} + \underbrace{\frac{1}{m} \sum_{i=1}^m \log(1 - D_{\theta_d}(G_{\theta_g}(z^{(i)})))}_{m \text{ synthetic samples}} \end{aligned}$$

- **generator:** minimize J_G wrt θ_g

$$\begin{aligned} J_G &= \mathbb{E}_{z \sim p(z)} \log(1 - D_{\theta_d}(G_{\theta_g}(z))) \\ &\approx \underbrace{\frac{1}{m} \sum_{i=1}^m \log(1 - D_{\theta_d}(G_{\theta_g}(z^{(i)})))}_{m \text{ synthetic samples}} \end{aligned}$$

Minimax game

- generator/discriminator: trained jointly in a _____ **game**
 - minimax objective:

$$\min_{\theta_g} \max_{\theta_d} [\mathbb{E}_{x \sim p_{\text{data}}} \log \underbrace{D_{\theta_d}(x)}_{\substack{\text{discriminator output} \\ \text{for real data } x}} + \mathbb{E}_{z \sim p(z)} \log(1 - \underbrace{D_{\theta_d}(G_{\theta_g}(z))}_{\substack{\text{discriminator output for} \\ \text{generated fake data } G(z)}})]$$

- discriminator (θ_d) wants to **maximize** objective s.t.
 - $D(x)$ is close to 1 (real) and
 - $D(G(z))$ is close to 0 (fake)
- generator (θ_g) wants to **minimize** objective s.t.
 - $D(G(z))$ is close to 1

i.e. discriminator is fooled into thinking that generated $G(x)$ is real

remarks:

- **Nash equilibrium**⁸ of this minimax game: a parameter setting in which
 - ▶ distribution of points created by G = distribution of real data samples
 - ▶ $D = 0.5$ for all of its input
- important: for this approach to work well
 - ▶ D should be a _____ model and have access to a lot of data
- **jointly training** two networks: challenging/can be unstable
 - ▶ choosing objectives with better loss landscapes helps training
 - ⇒ active area of research
- e.g. Wasserstein GAN (WGAN):⁹ earth-mover distance bet'n p_{data} and p_{model}
 - ▶ based on the notion of optimal transport (see [Appendix](#))

⁸ a proposed solution of a non-cooperative game involving two or more players in which each player is assumed to know the equilibrium strategies of the other players, and no player has anything to gain by changing only their own strategy (source: Wikipedia)

⁹ Arjovsky, M., Chintala, S., and Bottou, L. (2017). [Wasserstein generative adversarial networks](#). In *International conference on machine learning*, pages 214–223

Training details

- minimax objective:

$$\min_{\theta_g} \max_{\theta_d} [\mathbb{E}_{x \sim p_{\text{data}}} \log \overbrace{D_{\theta_d}(x)}^{\text{discriminator output for real data } x} + \mathbb{E}_{z \sim p(z)} \log(1 - \overbrace{D_{\theta_d}(G_{\theta_g}(z))}^{\text{discriminator output for generated fake data } G(z)})]$$

- alternate between:

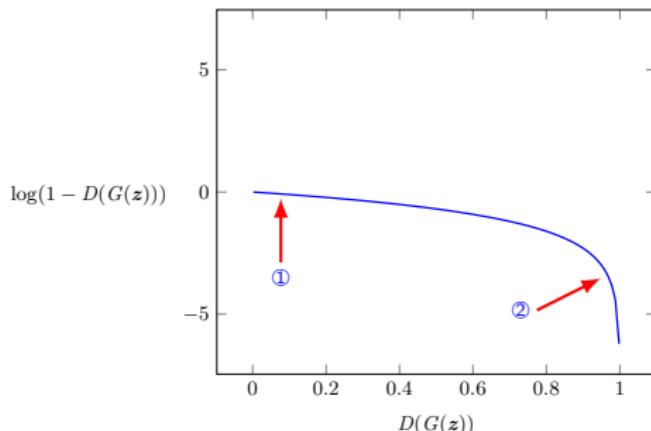
1. gradient _____ on discriminator

$$\max_{\theta_d} [\mathbb{E}_{x \sim p_{\text{data}}} \log D_{\theta_d}(x) + \mathbb{E}_{z \sim p(z)} \log(1 - D_{\theta_d}(G_{\theta_g}(z)))]$$

2. gradient descent on generator

$$\min_{\theta_g} \mathbb{E}_{z \sim p(z)} \log(1 - D_{\theta_d}(G_{\theta_g}(z)))$$

- in practice, optimizing this generator objective does not work well



- near ①: a sample is likely to be **fake** \Rightarrow want to learn from it to improve G
 - ▶ but gradient is relatively flat \Rightarrow little learning signal
- near ②: a sample is likely to be **real**
 - ▶ big gradient signal but of little use (the sample: already _____)

- minimax objective function:

$$\min_{\theta_g} \max_{\theta_d} [\mathbb{E}_{x \sim p_{\text{data}}} \log \overbrace{D_{\theta_d}(x)}^{\text{discriminator output for real data } x} + \mathbb{E}_{z \sim p(z)} \log(1 - \overbrace{D_{\theta_d}(G_{\theta_g}(z))}^{\text{discriminator output for generated fake data } G(z)})]$$

- alternate between:

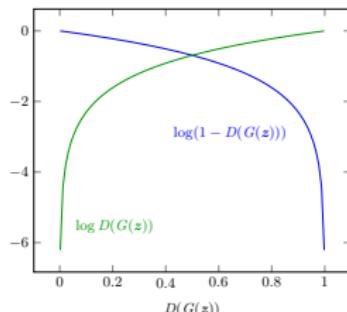
1. gradient ascent on discriminator

$$\max_{\theta_d} [\mathbb{E}_{x \sim p_{\text{data}}} \log D_{\theta_d}(x) + \mathbb{E}_{z \sim p(z)} \log(1 - D_{\theta_d}(G_{\theta_g}(z)))]$$

2. instead: gradient _____ on generator (different objective)

$$\max_{\theta_g} \mathbb{E}_{z \sim p(z)} \log(D_{\theta_d}(G_{\theta_g}(z)))$$

- * not minimizing likelihood of D being correct,
but maximize likelihood of D being wrong
 - * same objective of fooling D ,
but now higher gradient signal for bad samples
- ⇒ works much better! (standard in practice)



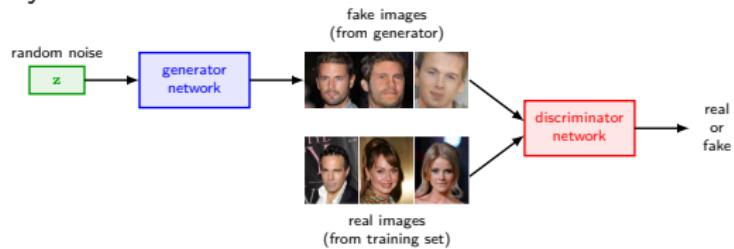
Each gradient update

1. **discriminator** update (_____ k times per iteration):

- ▶ prepare m synthetic samples (using generator) and m real samples
- ▶ feed these $2m$ samples to discriminator
- ▶ gradient ascent: backprop on discriminator network to update θ_d

2. **generator** update (once per iteration):

- ▶ prepare m synthetic samples using generator
- ▶ feed these m samples to discriminator
- ▶ gradient a/descent: backprop on whole network
 - ▷ backward gradient from discriminator output → generator
 - ▷ but update θ_g only



- putting it together¹⁰

Algorithm 1 Minibatch stochastic gradient descent training of generative adversarial nets. The number of steps to apply to the discriminator, k , is a hyperparameter.

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.

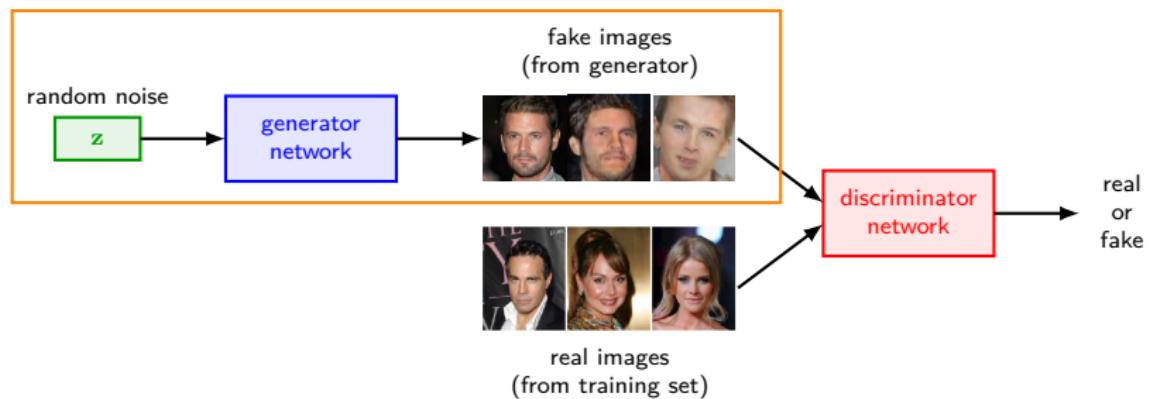
- no best rule for k (*i.e.* # steps to apply to discriminator D)

- ▶ some find $k = 1$ more stable; others use $k > 1$ (*e.g.* $k = 5$)
- ▶ recent work (*e.g.* Wasserstein GAN) alleviates this problem \Rightarrow better _____

¹⁰ Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., et al. (2014). Generative adversarial nets. In *Advances in neural information processing systems*, pages 2672–2680

After training

- use generator network to generate new images
 - ▶ _____ discriminator network



Performance evaluation

- recall: entropy
 - ▶ high entropy = high randomness (less predictable)
 - ▶ low entropy = low randomness (more predictable)
 - desideratum #1: **quality**
 - ▶ low entropy (highly predictable) conditional label distribution $p(y | \mathbf{x})$
 - i.e. given an image \mathbf{x} , should know its label y easily
 - ▶ use neural net to classify generated images \Rightarrow gives distribution $p(y | \mathbf{x})$
 - desideratum #2: **diversity**
 - ▶ high entropy (less predictable) $p(y)$
$$p(y) = \int p(y | \mathbf{x} = G(\mathbf{z})) d\mathbf{z}$$

i.e. if generated images are diverse, $p(y)$ should be _____
- * there is often trade-off between **quality** and **diversity**¹¹

¹¹Karras, T., Aila, T., Laine, S., and Lehtinen, J. (2017). Progressive growing of gans for improved quality, stability, and variation. *arXiv preprint arXiv:1710.10196*

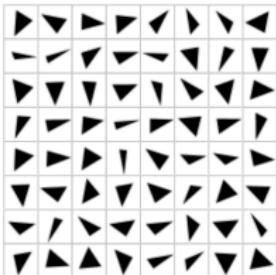
- Inception score (IS)
 - ▶ combines the two desiderata into one score:¹²
$$\text{IS}(G) = \exp(\mathbb{E}_{x \sim G}[D_{\text{KL}}(p(y | x) || p(y))])$$
 - ▶ used _____ for image classification; $\exp(\cdot)$: for easier comparison
 - ▶ the higher the better
 - ⚠ misrepresents performance if G only generates one image per class¹³
- Fréchet Inception distance (FID)¹⁴
 - ▶ actually compares statistics of generated samples to real samples
$$\text{FID}(\mathbf{x}_r, \mathbf{x}_g) = \|\boldsymbol{\mu}_r - \boldsymbol{\mu}_g\|_2^2 + \text{Tr}(\boldsymbol{\Sigma}_r + \boldsymbol{\Sigma}_g - 2(\boldsymbol{\Sigma}_r \boldsymbol{\Sigma}_g)^{1/2})$$
 - ▷ $\mathbf{x}_r \sim \mathcal{N}(\boldsymbol{\mu}_r, \boldsymbol{\Sigma}_r)$ and $\mathbf{x}_g \sim \mathcal{N}(\boldsymbol{\mu}_g, \boldsymbol{\Sigma}_g)$: 2048-dim activations of **Inception v3 pool3 layer** for real and generated samples, respectively
 - ▶ the lower the better

¹² Salimans, T., Goodfellow, I., Zaremba, W., Cheung, V., Radford, A., and Chen, X. (2016). Improved techniques for training gans. In *Advances in Neural Information Processing Systems*, pages 2234–2242

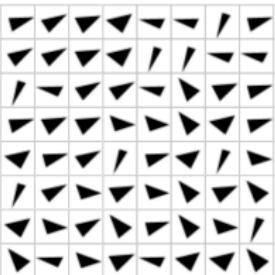
¹³ $p(y)$ will still be uniform despite low diversity

¹⁴ Heusel, M., Ramsauer, H., Unterthiner, T., et al. (2017). Gans trained by a two time-scale update rule converge to a local nash equilibrium. In *Advances in Neural Information Processing Systems*, pages 6626–6637

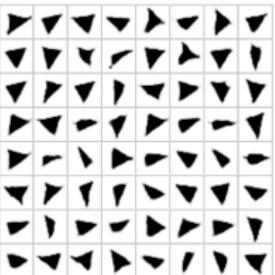
- precision, recall, and F_1 score
 - precision:** high when generated images look similar to real on average
 - recall:** high when generator can generate any sample in training data
 - F_1 score:** _____ mean of precision and recall



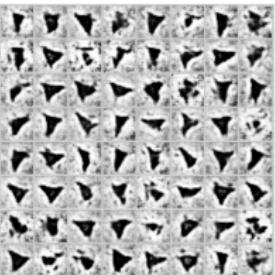
(a) High precision, high recall



(b) High precision, low recall



(c) Low precision, high recall

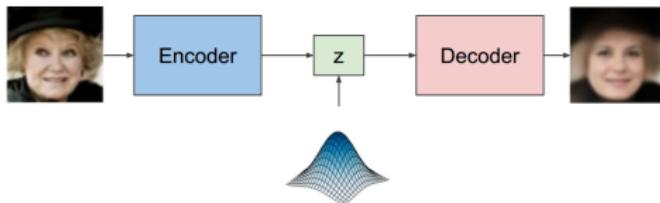


(d) Low precision, low recall

(source: [Lucic et al., 2017]¹⁵)

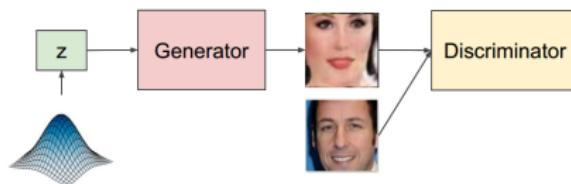
¹⁵Lucic, M., Kurach, K., Michalski, M., Gelly, S., and Bousquet, O. (2017). Are gans created equal? a large-scale study. *arXiv preprint arXiv:1711.10337*

VAE vs GAN



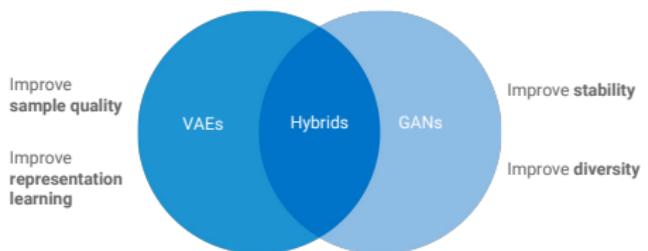
VAE

- ⊕ given x , easy to find z
- ⊕ interpretable probability $p(x)$
- ⊖ usually outputs blurry images



GAN

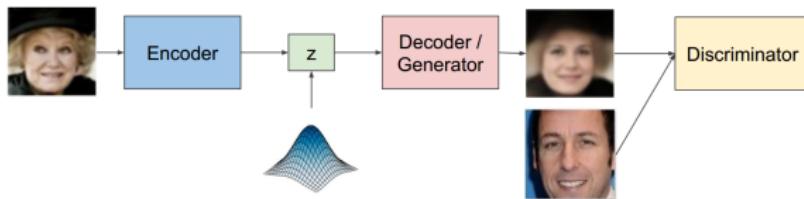
- ⊕ very sharp images
- ⊖ given x , difficult to find z
- ⊕/⊖ no explicit $p(x)$



(source: [Larsen et al., 2015]¹⁶, http://slazebni.cs.illinois.edu/spring17/lec12_vae.pdf, Rosca 2018)

¹⁶Larsen, A. B. L., Sønderby, S. K., Larochelle, H., and Winther, O. (2015). Autoencoding beyond pixels using a learned similarity metric. *arXiv preprint arXiv:1512.09300*

VAE + GAN



- training objective:

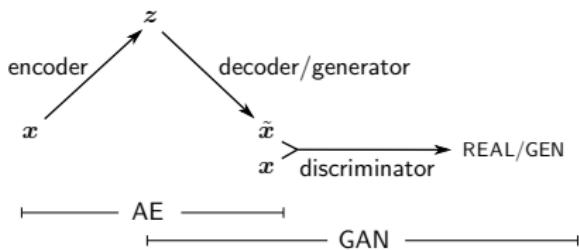
$$\mathcal{L} = \mathcal{L}_{\text{prior}} + \mathcal{L}_{\text{like}}^D + \mathcal{L}_{\text{GAN}}$$

where

$$\mathcal{L}_{\text{prior}} = D_{\text{KL}}[q(z|x) || p(z)]$$

$$\mathcal{L}_{\text{like}}^D = -\mathbb{E}_{q(z|x)}[\log p(D(x)|z)]$$

$$\mathcal{L}_{\text{GAN}} = \log(D(x)) + \log(1 - D(G(z)))$$



(source: [Larsen et al., 2015]¹⁷, http://slazebni.cs.illinois.edu/spring17/lec12_vae.pdf)

- adversarial AE, adversarial variational Bayes, VEEGAN, ALI/BiGAN, AlphaGAN, ...

¹⁷ Larsen, A. B. L., Sønderby, S. K., Larochelle, H., and Winther, O. (2015). Autoencoding beyond pixels using a learned similarity metric. *arXiv preprint arXiv:1512.09300*

Outline

Generative Adversarial Networks

Introduction

Training

Historical Developments

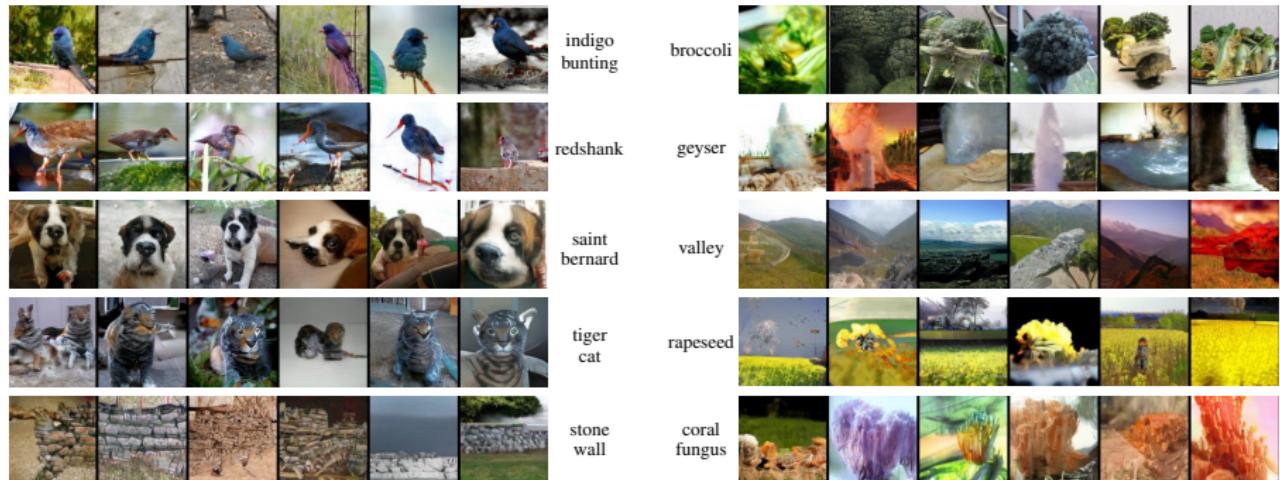
Challenges and Analyses

Summary

Appendix: Wasserstein GAN

Self-attention GAN (SAGAN)¹⁸

- state-of-the-art on ImageNet
 - ▶ 1000 categories, 128×128 pixels



(source: [Zhang et al., 2018])

¹⁸ Zhang, H., Goodfellow, I., Metaxas, D., and Odena, A. (2018). Self-attention generative adversarial networks. *arXiv preprint arXiv:1805.08318*

From GAN to SAGAN

- great ideas developed
 1. depth and convolution
 2. class-conditional generation
 3. spectral normalization
 4. two-timescale update rule
 5. self-attention



GAN [Goodfellow et al., 2014]¹⁹



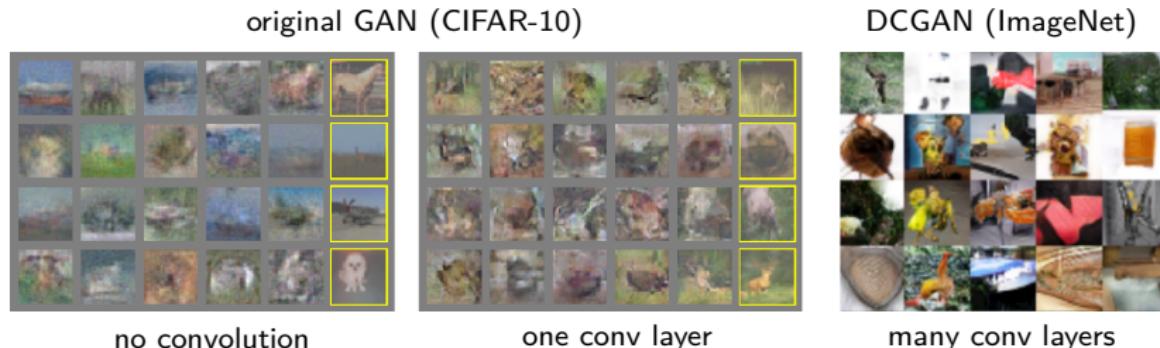
SAGAN [Zhang et al., 2018]²⁰

¹⁹ Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., et al. (2014). Generative adversarial nets. In *Advances in neural information processing systems*, pages 2672–2680

²⁰ Zhang, H., Goodfellow, I., Metaxas, D., and Odena, A. (2018). Self-attention generative adversarial networks. *arXiv preprint arXiv:1805.08318*

1. Depth and convolution

- difficulty matters
 - ▶ simple tasks: no convolution needed
 - ▶ harder tasks: depth and convolution needed

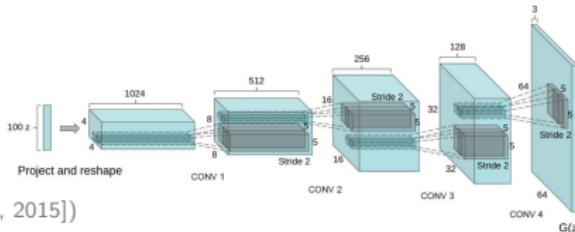


(source: [Goodfellow et al., 2014]²¹, [Radford et al., 2015]²²)

²¹ Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., et al. (2014). Generative adversarial nets. In *Advances in neural information processing systems*, pages 2672–2680

²² Radford, A., Metz, L., and Chintala, S. (2015). Unsupervised representation learning with deep convolutional generative adversarial networks. *arXiv preprint arXiv:1511.06434*

- deep convolutional GAN (DCGAN)²³
 - ▶ **generator:** an upsampling network with fractionally-strided convolutions



(source: [Radford et al., 2015])

- architectural guidelines for stable DCGANs
 - replace any pooling layers with
 - ▷ strided conv (discriminator) + fraction-strided conv (generator)
 - use **batch norm** in both generator and discriminator
 - remove **fully connected** hidden layers for deeper architectures
 - use _____ in generator for all layers except for the output, which uses tanh
 - use LeakyReLU in discriminator for all layers

²³ Radford, A., Metz, L., and Chintala, S. (2015). [Unsupervised representation learning with deep convolutional generative adversarial networks](#). *arXiv preprint arXiv:1511.06434*

2. Class-conditional generation

- even after developing idea 1
 - ▶ ImageNet samples did not yet have _____ objects
 - key idea to generate recognizable objects
 - ▶ let generative model learn joint distribution over pixels (x) and classes (y)

$$p(x, y) = p(y) \underbrace{p(x | y)}_{\text{conditional GAN learns this}}$$

- first work on conditional GAN:²⁴
 - ▶ discriminator gets to see both class label and pixel input, saying
 - ▷ whether pixels match requested class
 - ▷ not whether they are realistic
 - ▶ could generate 10 MNIST digits

(source: [Mirza and Osindero, 2014])

²⁴ Mirza, M. and Osindero, S. (2014). Conditional generative adversarial nets. *arXiv preprint arXiv:1411.1784*

- auxiliary classifier GAN (AC-GAN)²⁵
 - ▶ could generate class-conditional ImageNet samples
- trick: an _____ of specialized classifiers
 - ▶ a group of 100 different generators each specialized in 10 classes
- generated images: recognizable, coming from the desired class
- limitations
 - ▶ not yet **high-quality** samples
 - ▶ not yet a single generator for all 100 classes



monarch butterfly



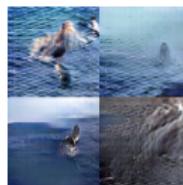
goldfinch



daisy



redshank



grey whale

(source: [Odena et al., 2016])

²⁵ Odena, A., Olah, C., and Shlens, J. (2016). *Conditional image synthesis with auxiliary classifier gans*. *arXiv preprint arXiv:1610.09585*

- spectrally normalized GAN (SN-GAN)²⁶
 - ▶ a **single generator** can make recognizable samples from every class
 - ▶ generated images: recognizable and somewhat high quality



(source: [Miyato et al., 2018])

- the idea of _____ normalization
 - ▶ important beyond just the context of class-conditional generation

²⁶Miyato, T., Kataoka, T., Koyama, M., and Yoshida, Y. (2018). Spectral normalization for generative adversarial networks. *arXiv preprint arXiv:1802.05957*

3. Spectral normalization (SN)²⁷

- main idea:
 - ▶ _____ the eigen/singular value spectrum of each weight matrix
- how it works:
 - ▶ for every layer of the net (that is represented by a linear mapping W)
 - ▶ estimate the **largest singular value** of that mapping and divide by it
 - ⇒ each of the weight matrices has a bounded spectral radius

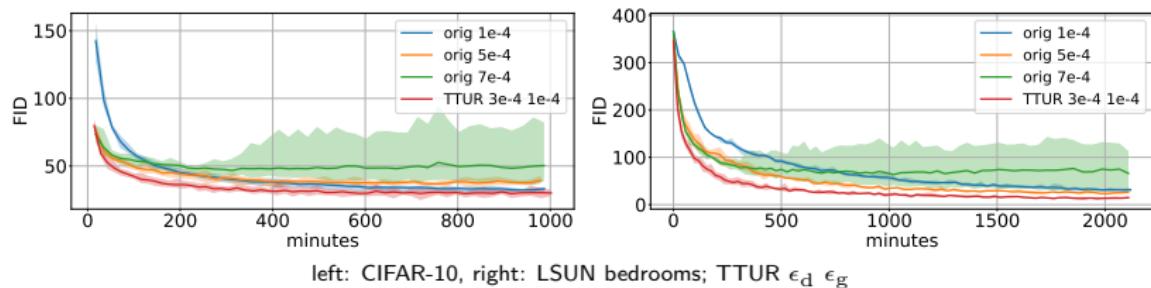
$$\bar{W}_{\text{SN}} \triangleq W / \underbrace{\sigma(W)}_{\substack{\text{spectral norm of } W \\ = \text{largest singular value}}}$$

- effects:
 - ▶ makes learning process much **more stable**
 - ▷ despite no direct connection to game theory
 - ▶ makes it more possible to use a lot of **computation** with GANs
 - ⇒ complex GAN training runs for weeks without diverging ⇒ beneficial

²⁷ Miyato, T., Kataoka, T., Koyama, M., and Yoshida, Y. (2018). Spectral normalization for generative adversarial networks. *arXiv preprint arXiv:1802.05957*

4. Two-timescale update rule (TTUR)²⁸

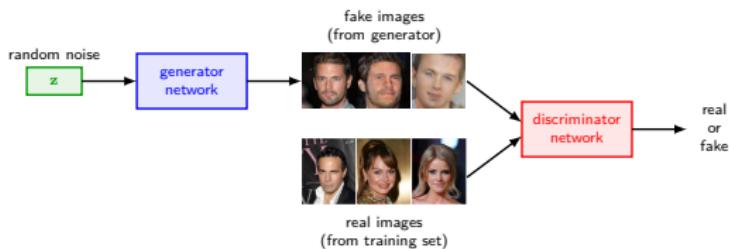
- key idea:
 - ▶ use **different learning rates** (ϵ_g , ϵ_d) for generator and discriminator
- in theory: we can prove
 - ▶ this helps with convergence if you decay ϵ_g _____ than ϵ_d
- in practice: no need for decaying ϵ_g
 - ▶ just using two different learning rates: sufficient



(source: [Heusel et al., 2017])

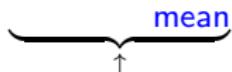
²⁸ Heusel, M., Ramsauer, H., Unterthiner, T., et al. (2017). *Gans trained by a two time-scale update rule converge to a local nash equilibrium*. In *Advances in Neural Information Processing Systems*, pages 6626–6637

- known tip for getting better GAN performance:
 - ▶ run D for several steps before running G (*e.g.* $k = 5$)
 - ▶ reason: D generates signals G uses to learn
- unfortunate consequence: to calculate a D step, we need
 - ▶ go forward through $G \rightarrow D$; then go backward through D
 - ⇒ much **computation wasted** if you do multiple D and G steps



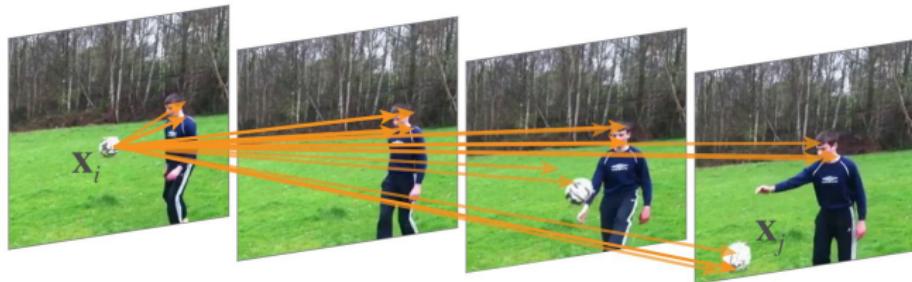
- TTUR benefits
 - ▶ allows a much _____ learning rate for D and for G
 - ▶ makes both players doing simultaneous steps and keeps the learning stable

5. Self-attention

- self-attention (= intra-attention)²⁹
 - ▶ relates different positions of a single sequence (*e.g.* a sentence)
 - ▷ in order to compute a representation of the sequence
 - ▶ computes the response at a position in a sequence by
 - ▷ attending to all positions and
 - ▷ taking their weighted average in an embedding space
 - ▶ can be viewed as a form of the  a weighted mean of all pixels in an image

²⁹Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., et al. (2017). Attention is all you need. In *Advances in Neural Information Processing Systems*, pages 5998–6008

- non-local neural nets³⁰
 - ▶ bridges self-attention to more general class of non-local filtering ops
 - ↑ for machine translation
 - ↑ for image/video problems



a space-time non-local operation [Wang et al., 2017]

³⁰Wang, X., Girshick, R., Gupta, A., and He, K. (2017). Non-local neural networks. *arXiv preprint arXiv:1711.07971*, 10

- self-attention GAN (SAGAN)³¹

- ▶ uses layers from the non-local neural nets
- ▶ main idea: during generation process
 - ▷ when generating each pixel, we can look around the image
 - ▷ and see what the other pixels look like



(source: [Zhang et al., 2018])

e.g. when generating the eye of a bear

- ▶ look around the other eye and make sure that they are _____
- ▶ we can focus attention in very unusually shaped regions too
 - e.g. narrow tail of a bird (square/circle-focused attention would miss)

³¹Zhang, H., Goodfellow, I., Metaxas, D., and Odena, A. (2018). Self-attention generative adversarial networks. *arXiv preprint arXiv:1805.08318*

Outline

Generative Adversarial Networks

Introduction

Training

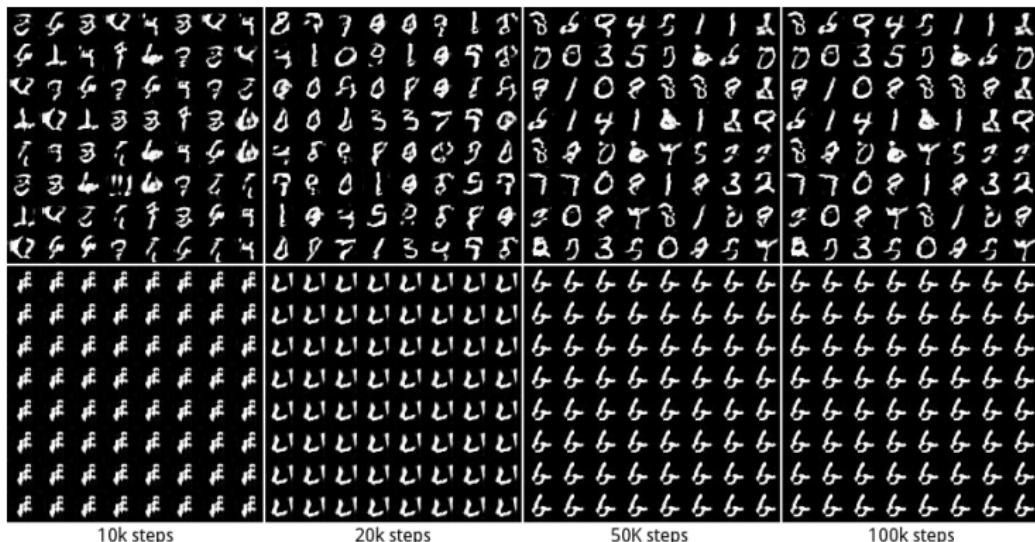
Historical Developments

Challenges and Analyses

Summary

Appendix: Wasserstein GAN

Mode collapse



(source: [Metz et al., 2016]³²)

- ▶ **top:** no mode collapsing (all digits appear)
- ▶ **bottom:** mode collapsing (to digit 6)

³² Metz, L., Poole, B., Pfau, D., and Sohl-Dickstein, J. (2016). Unrolled generative adversarial networks. *arXiv preprint arXiv:1611.02163*

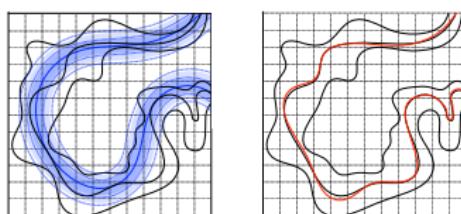
Sample sharpness

- GAN produces _____ samples
 - ▶ image generation



(source: [Larsen et al., 2015]³³)

- ▶ manifold estimation (left: VAE, right: GAN)



(source: Courville, 2017)

³³Larsen, A. B. L., Sønderby, S. K., Larochelle, H., and Winther, O. (2015). Autoencoding beyond pixels using a learned similarity metric. *arXiv preprint arXiv:1512.09300*

GAN training challenges

1. non-convergence
 - ▷ model parameters oscillate/destabilize/never converge
2. mode collapse
 - ▷ produce limited varieties of samples
3. diminished gradient
 - ▷ D too successful $\Rightarrow G$ gradient vanishes/learns nothing
4. generator/discriminator unbalance
 - ▷ causes _____
5. hyperparameter sensitivity
 - ▷ difficult to train robustly

Outline

Generative Adversarial Networks

Summary

Appendix: Wasserstein GAN

Summary

- GANs: do not work with an explicit density function
 - ▶ take a game-theoretic approach
 - ▶ learn to generate from training distribution through 2-player game
- advantages
 - ▶ beautiful, state-of-the-art samples
 - ▶ can parallelize sampling of generated data
 - ▶ no need to approximate likelihood by introducing lower bound
- disadvantages
 - ▶ trickier/more unstable to train (*e.g.* non-convergence, mode collapse)
 - ▶ cannot solve inference queries such as $p(x)$, $p(z|x)$
- areas of research
 - ▶ better loss function, more stable training (Wasserstein GAN, LSGAN, ...)
 - ▶ handling sample quality-variance trade-off
 - ▶ novel applications of GAN

Outline

Generative Adversarial Networks

Summary

Appendix: Wasserstein GAN

Wasserstein GAN (WGAN)³⁴

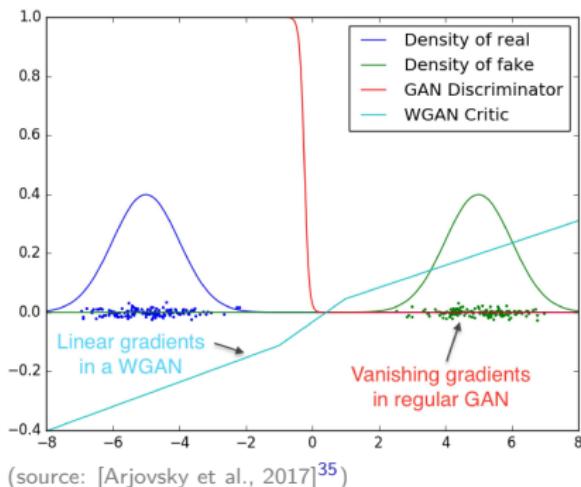
- proposes a **new cost function** using **Wasserstein distance**
 - ▶ has **smoother gradient** everywhere
 - ▶ improves stability of training
 - ▶ correlates with quality of generated images
- name (and role) change: discriminator → critic
 - ▶ GAN discriminator: predict the probability of generated images being real
 - ▶ WGAN critic: scores the realness/fakeness of a given image
- additional properties
 - ▶ learns no matter generator is performing or not
 - ▶ less sensitive to model architecture/hyperparameters

³⁴ Arjovsky, M., Chintala, S., and Bottou, L. (2017). Wasserstein generative adversarial networks. In *International conference on machine learning*, pages 214–223

WGAN advantages

- **improved training stability**

- ▶ suffer less from mode collapsing
- ▶ generator can still learn even when critic performs well



- optimal discriminator and critic

- ▶ when learning to differentiate two Gaussians

- traditional GAN discriminator

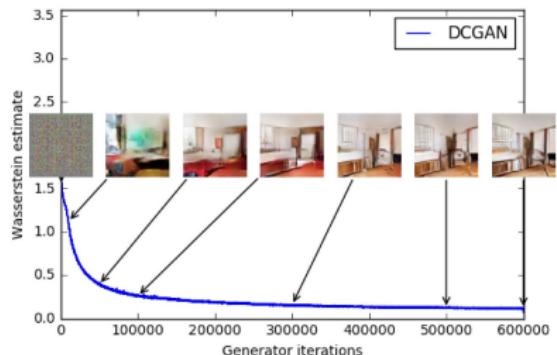
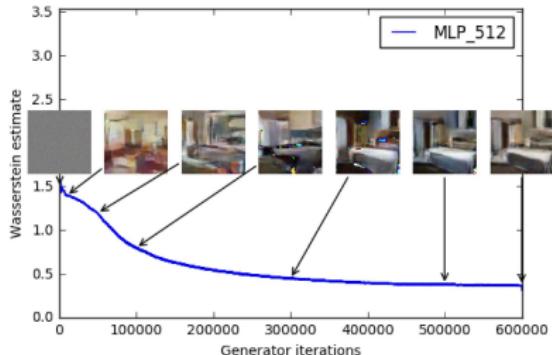
- ▶ saturates and results in vanishing gradients

- WGAN critic

- ▶ provides very clean gradients on all parts of the space

³⁵ Arjovsky, M., Chintala, S., and Bottou, L. (2017). Wasserstein generative adversarial networks. In *International conference on machine learning*, pages 214–223

- Wasserstein loss: correlates well with image quality
 - ▶ loss curve over time along with generated images



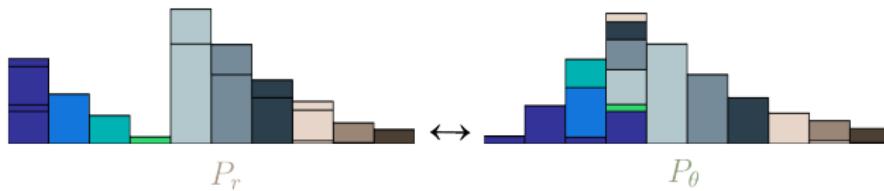
(source: [Arjovsky et al., 2017]³⁶)

- ▶ left: MLP generator + DCGAN critic
- ▶ right: DCGAN generator + DCGAN critic

³⁶ Arjovsky, M., Chintala, S., and Bottou, L. (2017). Wasserstein generative adversarial networks. In *International conference on machine learning*, pages 214–223

Wasserstein distance

- a measure of the distance between probability distributions
aka earth mover's distance (EM distance)
- intuition: EM distance = **min energy cost** of moving/transforming
 - ▶ a pile of dirt in the shape of one distribution to the shape of the other
 - ▶ the cost = (amount of dirt moved) \times (moving distance)



(source: <https://vincentherrmann.github.io/blog/wasserstein/>)

- definition:

$$W(P_r, P_g) = \inf_{\gamma \in \Pi(P_r, P_g)} \mathbb{E}_{(x,y) \sim \gamma} [\|x - y\|] \quad (3)$$

- ▶ $\Pi(P_r, P_g)$: set of all joint distributions $\gamma(x, y)$ ³⁷
- ▶ $\gamma(x, y)$: how much “mass” must be transported from x to y in order to transform distribution P_r into distribution P_g

- note:

- ▶ infimum in (3): highly intractable³⁸

³⁷ whose marginals are respectively P_r and P_g

³⁸ Arjovsky, M., Chintala, S., and Bottou, L. (2017). Wasserstein generative adversarial networks. In *International conference on machine learning*, pages 214–223

Recall: Lipschitz continuity

- the function f in Wasserstein metric should satisfy $\|f\|_L \leq K$
i.e. f should be *K-Lipschitz continuous*
- a real-valued function $f : \mathbb{R} \mapsto \mathbb{R}$ is called *K-Lipschitz continuous* if
 - ▶ there exists a real constant $K \geq 0$ s.t. for all $x_1, x_2 \in \mathbb{R}$,

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

- ▶ K : Lipschitz constant for f
- ▶ everywhere continuously differentiable $\not\Rightarrow$ Lipschitz continuous

e.g. $f(x) = |x|$

Duality and parametrization

- by Kantorovich-Rubinstein duality, W is equivalent to

$$W(P_r, P_\theta) = \sup_{\|f\|_L \leq 1} \mathbb{E}_{x \sim P_r}[f(x)] - \mathbb{E}_{x \sim P_\theta}[f(x)] \quad (4)$$

- ▶ $\|f\|_L \leq 1$ indicates f should be *1-Lipschitz continuous*
- ▶ supremum in (4): still intractable but easier to approximate [▶ Link](#)
- suppose a parameterized function family $\{f_w\}_{w \in \mathcal{W}}$ (w : weights)
 - ▶ assuming these functions are all K -Lipschitz for some K gives
$$\begin{aligned} \max_{w \in \mathcal{W}} \mathbb{E}_{x \sim P_r}[f_w(x)] - \mathbb{E}_{x \sim P_\theta}[f_w(x)] &\leq \sup_{\|f\|_L \leq K} \mathbb{E}_{x \sim P_r}[f(x)] - \mathbb{E}_{x \sim P_\theta}[f(x)] \\ &= K \cdot W(P_r, P_\theta) \end{aligned}$$
- in Wasserstein GAN
 - ▶ we will learn f_w (now called “**critic**” instead of discriminator)
 - ▶ no need to know exact K (absorbed into learning rate tuning) [▶ Link](#)

Wasserstein GAN

- want to train $P_\theta = g_\theta(Z)$ to match P_r
 - ▶ Z : random variable with distribution $p(z)$
 - ▶ g_θ : parametric function ($\mathcal{Z} \mapsto \mathcal{X}$; typically a neural net)
- Wasserstein distance and its derivative:

$$W(P_r, P_\theta) = \max_{w \in \mathcal{W}} \mathbb{E}_{x \sim P_r} [f_w(x)] - \mathbb{E}_{z \sim p(z)} [f_w(g_\theta(z))]$$

$$\nabla_\theta W(P_r, P_\theta) = -\mathbb{E}_{z \sim p(z)} [\nabla_\theta f_w(g_\theta(z))]$$

- training procedure:³⁹

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

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

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

- 1: **while** θ 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: $g_w \leftarrow \nabla_w [\frac{1}{m} \sum_{i=1}^m f_w(x^{(i)}) - \frac{1}{m} \sum_{i=1}^m f_w(g_\theta(z^{(i)}))]$
- 6: $w \leftarrow w + \alpha \cdot \text{RMSPProp}(w, g_w)$
- 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: $\theta \leftarrow \theta - \alpha \cdot \text{RMSPProp}(\theta, g_\theta)$
- 12: **end while**

³⁹ Arjovsky, M., Chintala, S., and Bottou, L. (2017). Wasserstein generative adversarial networks. In *International conference on machine learning*, pages 214–223

Enforcing K -Lipschitz continuity

how to enforce K -Lipschitz continuity of f_w during training:

- **original WGAN**: simple but very practical trick
 - ▶ after every gradient update, clamp w to a small window (*e.g.* [-0.01,0.01])
⇒ results in a compact parameter space \mathcal{W}
- **WGAN-GP** (WGAN with gradient penalty)⁴⁰
 - ▶ penalizes gradient norm moving away from its target value 1⁴¹

$$L = \underbrace{\mathbb{E}_{\tilde{x} \sim P_g}[D(\tilde{x})] - \mathbb{E}_{x \sim P_r}[D(x)]}_{\text{original critic loss}} + \lambda \underbrace{\mathbb{E}_{\hat{x} \sim P_{\hat{x}}}[(\|\nabla_{\hat{x}} D(\hat{x})\|_2 - 1)^2]}_{\text{gradient penalty}}$$

where \hat{x} is randomly sampled from \tilde{x} and x

$$\hat{x} = t\tilde{x} + (1-t)x \quad \text{with } 0 \leq t \leq 1$$

⁴⁰ Gulrajani, I., Ahmed, F., Arjovsky, M., Dumoulin, V., and Courville, A. C. (2017). Improved training of wasserstein gans. In *Advances in neural information processing systems*, pages 5767–5777

⁴¹ a differentiable function is 1-Lipschitz iff it has gradients with norm at most 1 everywhere