Deep Generative Models

Lecture 12

Roman Isachenko

Ozon Masters

Spring, 2022

Recap of previous lecture

$$f(\mathbf{x}, \phi) = \mathbf{W}_{K+1}\sigma_K(\mathbf{W}_K\sigma_{K-1}(\dots\sigma_1(\mathbf{W}_1\mathbf{x})\dots)).$$

- σ_k is a pointwise nonlinearities. We assume that $\|\sigma_k\|_L = 1$ (it holds for ReLU).
- $\mathbf{g}(\mathbf{x}) = \mathbf{W}\mathbf{x}$ is a linear transformation ($\nabla \mathbf{g}(\mathbf{x}) = \mathbf{W}$).

$$\|\mathbf{g}\|_L = \sup_{\mathbf{x}} \|\nabla \mathbf{g}(\mathbf{x})\|_2 = \|\mathbf{W}\|_2.$$

Critic spectral norm

$$\|f\|_L \leq \|\mathbf{W}_{K+1}\|_2 \cdot \prod_{k=1}^{K} \|\sigma_k\|_L \cdot \|\mathbf{W}_k\|_2 = \prod_{k=1}^{K+1} \|\mathbf{W}_k\|_2.$$

Spectral Normalization GAN

If we replace the weights in the critic $f(\mathbf{x}, \phi)$ by $\mathbf{W}_k^{SN} = \mathbf{W}_k / \|\mathbf{W}_k\|_2$, we will get $\|f\|_L \leq 1$.

Power iteration approximates the value of $\|\mathbf{W}\|_2$.

Miyato T. et al. Spectral Normalization for Generative Adversarial Networks, 2018

Recap of previous lecture

f-divergence minimization

$$D_f(\pi||p) = \mathbb{E}_{p(\mathbf{x})} f\left(\frac{\pi(\mathbf{x})}{p(\mathbf{x})}\right) \rightarrow \min_p .$$

Here $f : \mathbb{R}_+ \rightarrow \mathbb{R}$ is a convex, lower semicontinuous function satisfying $f(1) = 0$.

Variational divergence estimation

$$D_f(\pi||p) \geq \sup_{T \in \mathcal{T}} \left[\mathbb{E}_\pi T(\mathbf{x}) - \mathbb{E}_p f^*(T(\mathbf{x}))\right],$$

Fenchel conjugate

$$f^*(t) = \sup_{u \in \text{dom}_f} (ut - f(u)), \quad f(u) = \sup_{t \in \text{dom}_{f*}} (ut - f^*(t))$$

Note: To evaluate lower bound we only need samples from $\pi(\mathbf{x})$ and $p(\mathbf{x})$. Hence, we could fit implicit generative model.

Nowozin S., Cseke B., Tomioka R. f-GAN: Training Generative Neural Samplers using Variational Divergence Minimization, 2016

Recap of previous lecture

Let take some pretrained image classification model to get the conditional label distribution $p(y|\mathbf{x})$ (e.g. ImageNet classifier).

Evaluation of likelihood-free models

- Sharpness $\Rightarrow$ low $H(y|\mathbf{x}) = -\sum_y \int_{\mathbf{x}} p(y, \mathbf{x}) \log p(y|\mathbf{x}) d\mathbf{x}$.
- Diversity $\Rightarrow$ high $H(y) = -\sum_y p(y) \log p(y)$.

Inception Score

$$IS = \exp(H(y) - H(y|\mathbf{x})) = \exp\left(\mathbb{E}_{\mathbf{x}} KL(p(y|\mathbf{x})||p(y))\right)$$

Frechet Inception Distance

$$D^2(\pi, p) = \|\mathbf{m}_\pi - \mathbf{m}_p\|_2^2 + \mathrm{Tr}\left(\mathbf{\Sigma}_\pi + \mathbf{\Sigma}_p - 2\sqrt{\mathbf{\Sigma}_\pi \mathbf{\Sigma}_p}\right).$$

FID is related to moment matching.

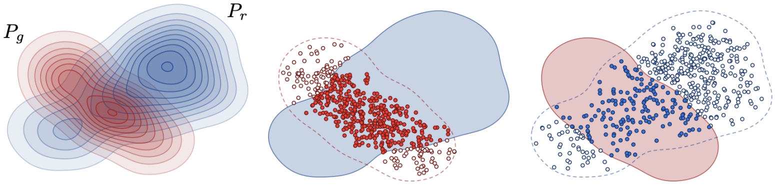
Salimans T. et al. Improved Techniques for Training GANs, 2016
Heusel M. et al. GANs Trained by a Two Time-Scale Update Rule Converge to a Local Nash Equilibrium, 2017

Outline

Precision-Recall for Generative Models

What do we want from samples

- **Sharpness**: generated samples should be of high quality.
- **Diversity**: their variation should match that observed in the training set.



(a) Example distributions (b) Precision (c) Recall

- **Precision** denotes the fraction of generated images that are realistic.
- **Recall** measures the fraction of the training data manifold covered by the generator.

Kynkäänniemi T. et al. Improved precision and recall metric for assessing generative models, 2019

Precision-Recall for Generative Models

- $\mathcal{S}_\pi = \{\mathbf{x}_i\}_{i=1}^n \sim \pi(\mathbf{x})$ – real samples;
- $\mathcal{S}_p = \{\mathbf{x}_i\}_{i=1}^n \sim p(\mathbf{x}|\boldsymbol{\theta})$ – generated samples.

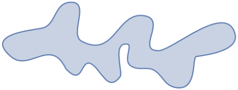
Embed samples using pretrained classifier network (as previously):

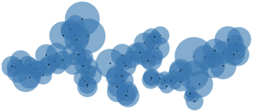
$$\mathcal{G}_\pi = \{\mathbf{g}_i\}_{i=1}^n, \quad \mathcal{G}_p = \{\mathbf{g}_i\}_{i=1}^n.$$

Define binary function:

$$f(\mathbf{g}, \mathcal{G}) = \begin{cases} 1, \text{if exists } \mathbf{g}' \in \mathcal{G} : \|\mathbf{g} - \mathbf{g}'\|_2 \leq \|\mathbf{g}' - \text{NN}_k(\mathbf{g}', \mathcal{G})\|_2; \\ 0, \text{otherwise.} \end{cases}$$

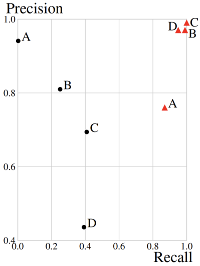
$$\text{Precision}(\mathcal{G}_\pi, \mathcal{G}_p) = \frac{1}{n} \sum_{\mathbf{g} \in \mathcal{G}_p} f(\mathbf{g}, \mathcal{G}_\pi); \quad \text{Recall}(\mathcal{G}_\pi, \mathcal{G}_p) = \frac{1}{n} \sum_{\mathbf{g} \in \mathcal{G}_\pi} f(\mathbf{g}, \mathcal{G}_p).$$


(a) True manifold


(b) Approx. manifold

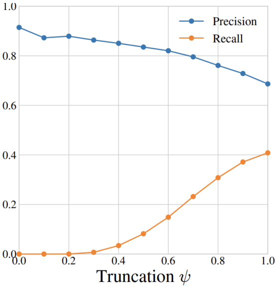
Kynkäänniemi T. et al. Improved precision and recall metric for assessing generative models, 2019

Precision-Recall for Generative Models



A (FID = 91.7) **B** (FID = 16.9) **C** (FID = 4.5) **D** (FID = 16.7)

$\psi = 0.0$ $\psi = 0.3$ $\psi = 0.7$ $\psi = 1.0$



Kynkäänniemi T. et al. Improved precision and recall metric for assessing generative models, 2019

Evolution of GANs


2014


2015


2016

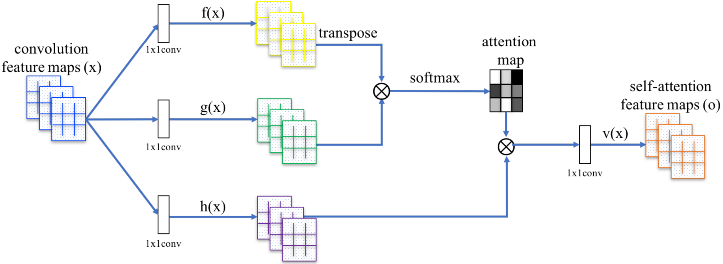

2017


2018

- **Standard GAN** https://arxiv.org/abs/1406.2661
- **DCGAN** https://arxiv.org/abs/1511.06434
- **CoGAN** https://arxiv.org/abs/1606.07536
- **ProGAN** https://arxiv.org/abs/1710.10196
- **StyleGAN** https://arxiv.org/abs/1812.04948

Self-Attention GAN

Convolutional layers process the information in a local neighborhood $\Rightarrow$ inefficient for modeling long-range dependencies in images.



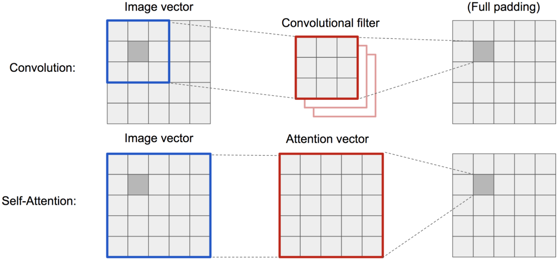
$$\mathbf{f}(\mathbf{x}) = \mathbf{W}_f\mathbf{x}, \quad \mathbf{g}(\mathbf{x}) = \mathbf{W}_g\mathbf{x}, \quad \mathbf{h}(\mathbf{x}) = \mathbf{W}_h\mathbf{x}, \quad \mathbf{v}(\mathbf{x}) = \mathbf{W}_v\mathbf{x}$$

$$s_{ij} = \mathbf{f}(\mathbf{x}_i)^T\mathbf{g}(\mathbf{x}_j), \quad a_{ij} = \frac{\exp s_{ij}}{\sum_{i=1}^N \exp s_{ij}}, \quad \mathbf{o}_j = \mathbf{v}\left(\sum_{i=1}^N a_{ij}\mathbf{h}(\mathbf{x}_i)\right)$$

Zhang H. et al. Self-Attention Generative Adversarial Networks, 2018

Self-Attention GAN

Convolution vs Attention



Visualization of attention maps

image credit: https://lilianweng.github.io/lil-log/2018/06/24/attention-attention.html
Zhang H. et al. Self-Attention Generative Adversarial Networks, 2018

BigGAN

Batch-size is matter

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)

Samples (512x512)

Brock A., Donahue J., Simonyan K. Large Scale GAN Training for High Fidelity Natural Image Synthesis, 2018

Progressive Growing GAN

Problems with HR image generation

- Disjoint manifolds $\Rightarrow$ gradient problem.
- Small minibatch $\Rightarrow$ training instability.

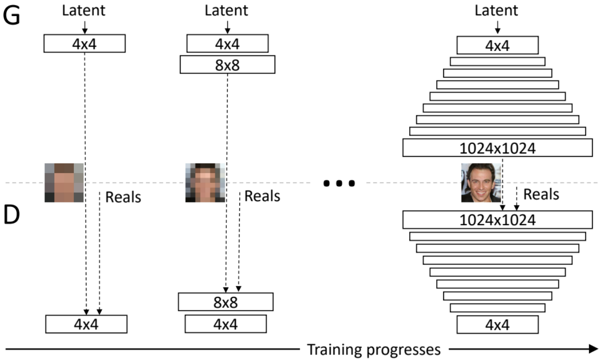
Samples (1024x1024)

Karras T. et al. Progressive Growing of GANs for Improved Quality, Stability, and Variation, 2017

Progressive Growing GAN

Grow both the generator and discriminator progressively, new layers will introduce higher-resolution details as the training progresses.

- Train GAN which generate 4x4 images (2 convs for G and D).
- Add upsampling layers to G, downsampling layers to D.
- Train GAN which generate 8x8 images.
- etc.



Karras T. et al. Progressive Growing of GANs for Improved Quality, Stability, and Variation, 2017

StyleGAN

- Generating of HR images is hard.
- Progressive growing greatly simplifies the task.
- The ability to control specific features of the generated image is very limited.

Face image features

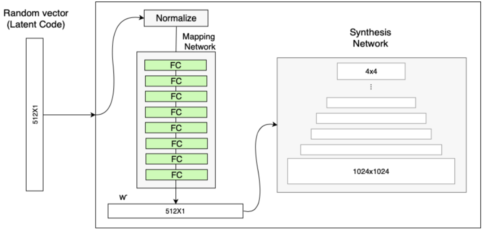
- Coarse (pose, general hair style, face shape). Resolution $4^2 - 8^2$.
- Middle (finer facial features, hair style, eyes open/closed). Resolution $16^2 - 32^2$.
- Fine (color scheme (eye, hair and skin) and micro features). Resolution $64^2 - 1024^2$.

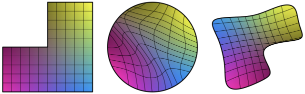
Karras T., Laine S., Aila T. A Style-Based Generator Architecture for Generative Adversarial Networks, 2018

StyleGAN

Mapping Network

- Generator input is likely to be **disentangled**. Each component of input vector $\mathbf{z}$ should be responsible for one generative factor.
- Mapping network $f : \mathcal{Z} \to \mathcal{W}$ is used to reduce correlations between components of $\mathbf{z}$.





(a) Distribution of features in training set (b) Mapping from $\mathcal{Z}$ to features (c) Mapping from $\mathcal{W}$ to features

Karras T., Laine S., Aila T. A Style-Based Generator Architecture for Generative Adversarial Networks, 2018

Truncation trick

BigGAN: truncated normal sampling

$$p(\mathbf{z}|b) = \mathcal{N}(\mathbf{z}|0,1) / \int_{-\infty}^{b} \mathcal{N}(\mathbf{z}|0,1) d\mathbf{z}$$

Components of $\mathbf{z} \sim \mathcal{N}(0, \mathbf{I})$ which fall outside a predefined range are resampled.

StyleGAN

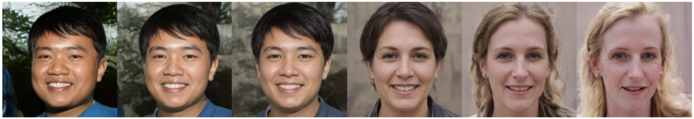
$$\mathbf{w}' = \hat{\mathbf{w}} + \psi \cdot (\mathbf{w} - \hat{\mathbf{w}}), \quad \hat{\mathbf{w}} = \mathbb{E}_{\mathbf{z}} p(f(\mathbf{z}))$$

- Constant ψ is a tradeoff between diversity and fidelity.
- $\psi = 0.7$ is used for most of the results.
- Truncation is done only at the low-resolution layers.

Karras T., Laine S., Aila T. A Style-Based Generator Architecture for Generative Adversarial Networks, 2018

StyleGAN

Truncation trick

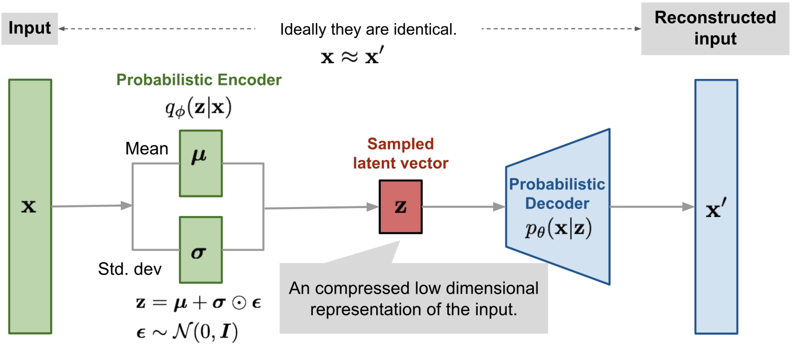


$\psi = 1$ $\psi = 0.7$ $\psi = 0.5$ $\psi = 0$ $\psi = -0.5$ $\psi = -1$

Samples (1024x1024)

Karras T., Laine S., Aila T. A Style-Based Generator Architecture for Generative Adversarial Networks, 2018

VAE recap



- Encoder $q(\mathbf{z}|\mathbf{x}, \phi) = \mathcal{N}(\mathbf{z}|\boldsymbol{\mu}_\phi(\mathbf{x}), \boldsymbol{\sigma}_\phi(\mathbf{x}))$.
- Variational posterior $q(\mathbf{z}|\mathbf{x}, \phi)$ originally approximates the true posterior $p(\mathbf{z}|\mathbf{x}, \boldsymbol{\theta})$.
- Which methods are you already familiar with to make the posterior is more flexible?

image credit: https://lilianweng.github.io/lil-log/2018/08/12/from-autoencoder-to-beta-vae.html

Adversarial Variational Bayes

ELBO objective

$$\mathcal{L}(\phi, \boldsymbol{\theta}) = \mathbb{E}_{q(\mathbf{z}|\mathbf{x},\phi)} \left[\log p(\mathbf{x}|\mathbf{z}, \boldsymbol{\theta}) + \log p(\mathbf{z}) - \log q(\mathbf{z}|\mathbf{x}, \phi)\right] \rightarrow \max_{\boldsymbol{\phi}, \boldsymbol{\theta}}.$$

What is the problem to make the variational posterior model an implicit model?

- The first term is reconstruction loss that needs only samples from $q(\mathbf{z}|\mathbf{x}, \phi)$ to evaluate.
- Reparametrization trick allows to get gradients of reconstruction loss

$$\nabla_\phi \int q(\mathbf{z}|\mathbf{x}, \phi) f(\mathbf{z}) d\mathbf{z} = \nabla_\phi \int r(\epsilon) f(\mathbf{z}) d\epsilon$$
$$= \int r(\epsilon) \nabla_\phi f(g(\mathbf{x}, \epsilon, \phi)) d\epsilon \approx \nabla_\phi f(g(\mathbf{x}, \epsilon^*, \phi)),$$

where $\epsilon^* \sim r(\epsilon)$, $\quad \mathbf{z} = g(\mathbf{x}, \epsilon, \phi)$, $\quad \mathbf{z} \sim q(\mathbf{z}|\mathbf{x}, \phi)$.

Mescheder L., Nowozin S., Geiger A. Adversarial variational bayes: Unifying variational autoencoders and generative adversarial networks, 2017

Adversarial Variational Bayes

ELBO objective

$$\mathcal{L}(\phi, \boldsymbol{\theta}) = \mathbb{E}_{q(\mathbf{z}|\mathbf{x}, \phi)} \left[\log p(\mathbf{x}|\mathbf{z}, \boldsymbol{\theta}) + \log p(\mathbf{z}) - \log q(\mathbf{z}|\mathbf{x}, \phi)\right] \rightarrow \max_{\boldsymbol{\phi}, \boldsymbol{\theta}}.$$

What is the problem to make the variational posterior model an implicit model?

- The third term requires the explicit the value of $q(\mathbf{z}|\mathbf{x}, \phi)$.
- We could join second and third terms:

$$\mathbb{E}_{q(\mathbf{z}|\mathbf{x}, \phi)} \log \frac{p(\mathbf{z})}{q(\mathbf{z}|\mathbf{x}, \phi)} = \mathbb{E}_{q(\mathbf{z}|\mathbf{x}, \phi)} \log \frac{p(\mathbf{z})\pi(\mathbf{x})}{q(\mathbf{z}|\mathbf{x}, \phi)\pi(\mathbf{x})}.$$

- We have to estimate density ratio

$$r(\mathbf{x}, \mathbf{z}) = \frac{q_1(\mathbf{x}, \mathbf{z})}{q_2(\mathbf{x}, \mathbf{z})} = \frac{p(\mathbf{z})\pi(\mathbf{x})}{q(\mathbf{z}|\mathbf{x}, \phi)\pi(\mathbf{x})}.$$

Mescheder L., Nowozin S., Geiger A. Adversarial variational bayes: Unifying variational autoencoders and generative adversarial networks, 2017

Density ratio trick

Consider two distributions $q_1(\mathbf{x})$, $q_2(\mathbf{x})$ and probabilistic model

$$p(\mathbf{x}|y) = \begin{cases} q_1(\mathbf{x}), & \text{if } y = 1, \\ q_2(\mathbf{x}), & \text{if } y = 0, \end{cases} \qquad y \sim \text{Bern}(0.5).$$

Density ratio

$$\frac{q_1(\mathbf{x})}{q_2(\mathbf{x})} = \frac{p(\mathbf{x}|y=1)}{p(\mathbf{x}|y=0)} = \frac{p(y=1|\mathbf{x})p(\mathbf{x})}{p(y=1)} \Bigg/ \frac{p(y=0|\mathbf{x})p(\mathbf{x})}{p(y=0)} =$$
$$= \frac{p(y=1|\mathbf{x})}{p(y=0|\mathbf{x})} = \frac{p(y=1|\mathbf{x})}{1 - p(y=1|\mathbf{x})} = \frac{D(\mathbf{x})}{1 - D(\mathbf{x})}$$

Here $D(\mathbf{x})$ is a discriminator model the output of which is a probability that $\mathbf{x}$ is a sample from $q_1(\mathbf{x})$ rather than from $q_2(\mathbf{x})$.

Adversarial Variational Bayes

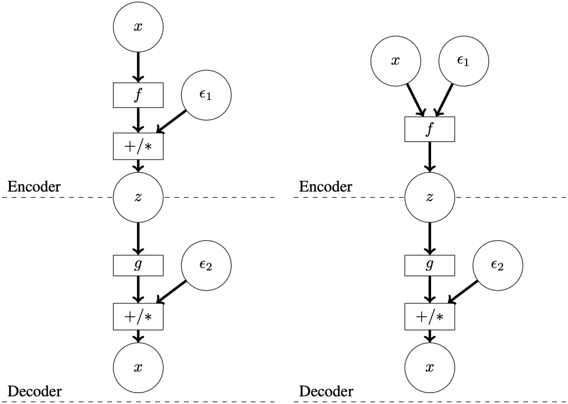
$$\max_D \left[\mathbb{E}_{\pi(\mathbf{x})}\mathbb{E}_{q(\mathbf{z}|\mathbf{x},\phi)} \log D(\mathbf{x}, \mathbf{z}) + \mathbb{E}_{\pi(\mathbf{x})}\mathbb{E}_{p(\mathbf{z})} \log(1 - D(\mathbf{x}, \mathbf{z}))\right]$$

Mescheder L., Nowozin S., Geiger A. Adversarial variational bayes: Unifying variational autoencoders and generative adversarial networks, 2017

Adversarial Variational Bayes

ELBO objective

$$\mathcal{L}(\phi, \boldsymbol{\theta}) = \mathbb{E}_{q(\mathbf{z}|\mathbf{x},\phi)} \left[\log p(\mathbf{x}|\mathbf{z}, \boldsymbol{\theta}) + \log \frac{p(\mathbf{z})}{q(\mathbf{z}|\mathbf{x}, \phi)} \right] \to \max_{\phi, \boldsymbol{\theta}}.$$



Mescheder L., Nowozin S., Geiger A. Adversarial variational bayes: Unifying variational autoencoders and generative adversarial networks, 2017

Summary

- Precision-recall allows to select model with compromise with sample quality and sample diversity.
- Self-Attention GAN allows to make huge receptive field and reduce convolution inductive bias.
- BigGAN shows that large batch size increase model quality gradually.
- Progressive growing for GAN learning allows to make training more stable.
- StyleGAN introduces mapping network to get more disentangled latent representation.
- Adversarial Variational Bayes uses density ratio trick to get more powerful variational posterior.