Bayesian Methods in Generative Adversarial Networks

Alexey Umnov

November 17, 2017

Generative Adversarial Networks

Generative Adversarial Networks (GAN)

Goal:

$x \sim p^*(x)$ — data distribution.

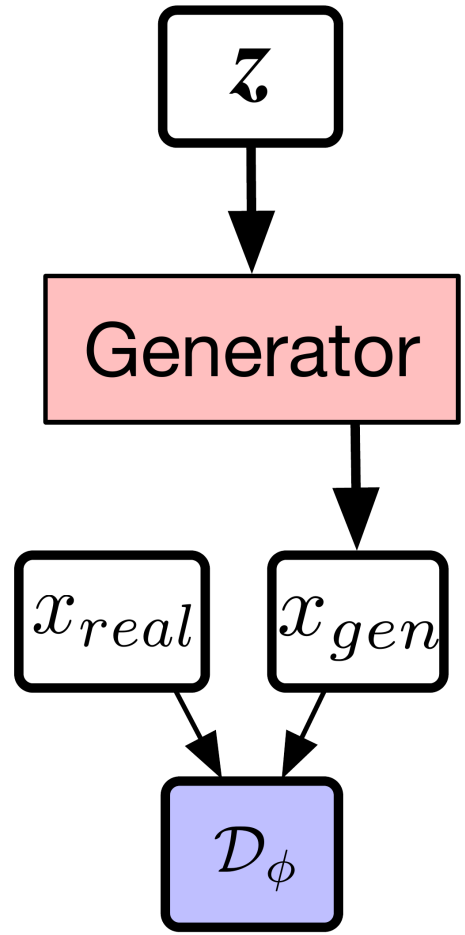
Learn to generate similar samples.

GAN:

$z \sim p(z)$ — latent variables.

$G_\theta(z)$ — generator.

$D_\phi(x)$ — discriminator.



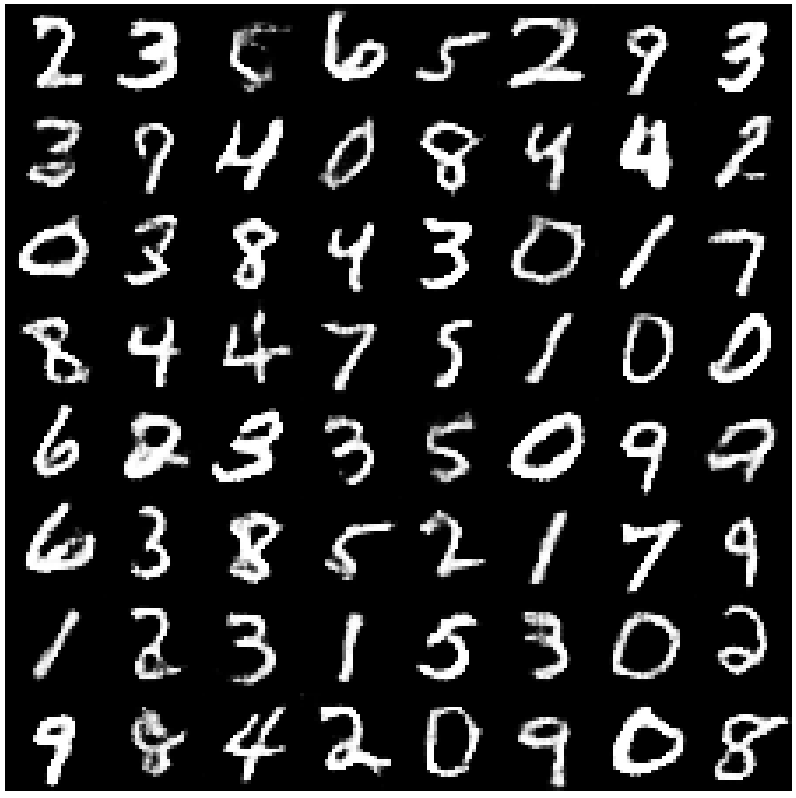
Generating ($p_\theta(x)$): $z \sim p(z),\ x = G_\theta(z)$.

Learning (alternating optimization steps):

$$\min_\phi \ \mathbb{E}_{p^*(x)}[-\log D_\phi(x)] + \mathbb{E}_{p_\theta(x)}[-\log(1 - D_\phi(x))].$$

$$\min_\theta \ \mathbb{E}_{p_\theta(x)}[-\log D_\phi(x)].$$

GAN advantage: image quality

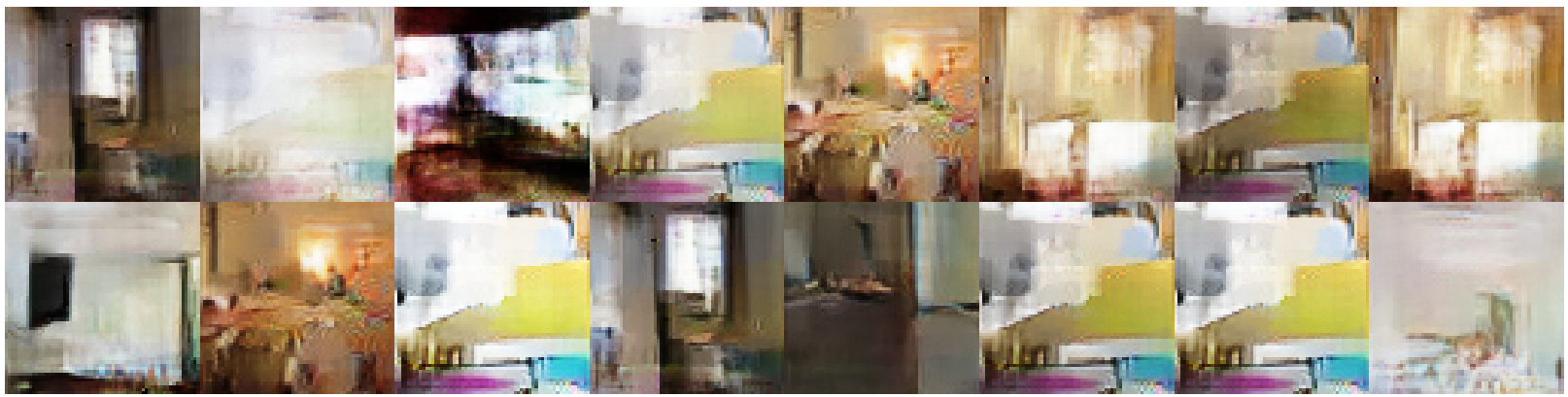


DCGAN

VAE

GAN problems (some of them)

- **Mode collapse.**
  Ignoring modes in the data distribution, generating samples with low diversity.



- **Hard to train.**
  Unstable training, hacks and tricks are necessary.

Dealing with problems: bayesian methods

α-GAN approach: Merge GAN and VAE architectures.

- Use more general likelihoods then in VAE.
- Fight intractability using discriminators.

Rosca M. et. al. Variational Approaches for Auto-Encoding Generative Adversarial Networks. 2017.

Bayesian GAN approach: Distributions over weights.

- Instead of learning one generative network learn a distribution over networks.
- To generate an example: draw random network, draw random sample.

Saatchi Y., Wilson A. Bayesian GAN. 2017.

α-GAN

Reminder: Variational Auto-Encoder

$x \sim p^*(x)$ — data distribution (unknown).
$z \sim p(z)$ — latent variables distribution (usually Gaussian).

$p_\theta(x \mid z)$ — decoder distribution.
$q_\eta(z \mid x)$ — encoder distribution.
θ, η — parameters that we want to learn.

Maximum likelihood:

$$\max \mathbb{E}_{p^*(x)} \log p_\theta(x) = \max \mathbb{E}_{p^*(x)} \log \int p_\theta(x \mid z) p(z) dz$$

Intractable $\Rightarrow$ using lower bound:

$$\log p_\theta(x) \geq \mathbb{E}_{q_\eta(z|x)} \log p_\theta(x \mid z) - \mathrm{KL}[q_\eta(z \mid x) \| p(z)].$$

Variational Auto-Encoder: Likelihood Choice

Lower bound optimization:

$$\max_{\theta,\phi} \mathbb{E}_{p^*(x)} \Big(\mathbb{E}_{q_\eta(z|x)} \log p_\theta(x \mid z) - \mathrm{KL}[q_\eta(z \mid x) \| p(z)] \Big).$$

Assumptions on likelihoods:

$E(z;\theta)$ — decoder network.
$p_\theta(x \mid z) = \mathcal{N}(E(z;\theta), \sigma^2 I)$ — decoder distribution.

$\mu(x;\eta), \Sigma(x;\eta)$ — encoder network.
$q_\eta(z \mid x) = \mathcal{N}(\mu(x;\eta), \Sigma(x;\eta))$ — encoder distribution.

- Good: lower bound is now tractable.
- Bad: assumptions may be too strong.

Density Ratio Trick

Consider GAN discriminator $D(x)$:

- Label $y = 1$: real data $x \sim p^*(x) = p(x \mid y = 1)$.
- Label $y = 0$: fake data $x \sim p_\theta(x) = p(x \mid y = 0)$.

Discriminator predicts probability of real data:

$D(x) = p(y = 1 \mid x)$.

Consider density ratio:

$$\frac{p^*(x)}{p_\theta(x)} = \frac{p(x \mid y = 1)}{p(x \mid y = 0)} = \frac{p(y = 1 \mid x)}{p(y = 0 \mid x)} \approx \frac{D(x)}{1 - D(x)}.$$

Synthetic Likelihoods

$$\max_{\theta,\eta} \mathbb{E}_{p^*(x)} \Big(\mathbb{E}_{q_\eta(z|x)} \log p_\theta(x \mid z) - \mathrm{KL}[q_\eta(z \mid x) \| p(z)] \Big).$$

First term:

$$\mathbb{E}_{q_\eta(z|x)} \log p_\theta(x \mid z) = \mathbb{E}_{q_\eta(z|x)} \left[\log \frac{p_\theta(x \mid z)}{p^*(x)} \right] + \mathbb{E}_{q_\eta(z|x)} \log p^*(x).$$

Density ratio trick:

$$\mathbb{E}_{q_\eta(z|x)} \left[\log \frac{p_\theta(x \mid z)}{p^*(x)} \right] \approx \mathbb{E}_{q_\eta(z|x)} \left[\frac{1 - D_\phi(G_\theta(z))}{D_\phi(G_\theta(z))} \right].$$

$\mathbb{E}_{q_\eta(z|x)} \log p^*(x)$ is independent of θ.

Synthetic Likelihoods

$$\max_{\theta,\eta} \mathbb{E}_{p^*(x)} \Big(\mathbb{E}_{q_\eta(z|x)} \log p_\theta(x \mid z) - \mathrm{KL}[q_\eta(z \mid x) \| p(z)] \Big).$$

Second term:

$$-\mathrm{KL}[q_\eta(z \mid x) \| p(z)] = \mathbb{E}_{q_\eta(z|x)} \left[\log \frac{p(z)}{q_\eta(z \mid x)} \right] \approx$$

$$\approx \mathbb{E}_{q_\eta(z|x)} \left[\log \frac{C_\omega(z)}{1 - C_\omega(z)} \right].$$

Likelihood choice

Likelihood choice 1 (Laplace distribution):

$$\mathbb{E}_{q_\eta(z|x)} \log p_\theta(x \mid z) = \mathbb{E}_{q_\eta(z|x)} \big[-\lambda \|x - G_\theta(z)\|_1\big].$$

VAE-like objective, no mode collapse.

Likelihood choice 2 (density ratio trick):

$$\mathbb{E}_{q_\eta(z|x)} \log p_\theta(x \mid z) \approx \mathbb{E}_{q_\eta(z|x)} \left[\log \frac{1 - D_\phi(G_\theta(z))}{D_\phi(G_\theta(z))}\right].$$

GAN-like objective, good samples quality.

Final approach: combine both.

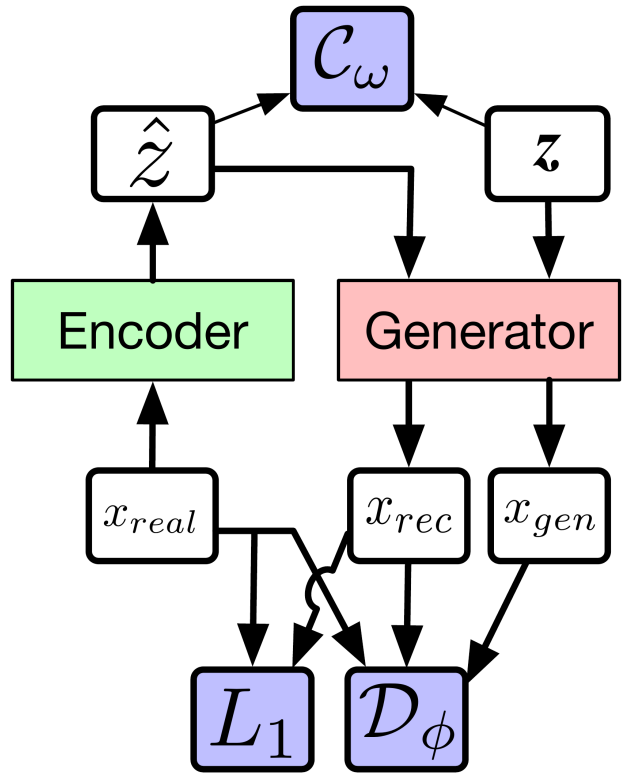
Full architecture

$G_\theta(z)$ — generator (decoder).

$E_\eta(x)$ — encoder.

$D_\phi(x)$ — data discriminator.

$C_\omega(z)$ — latent discriminator.

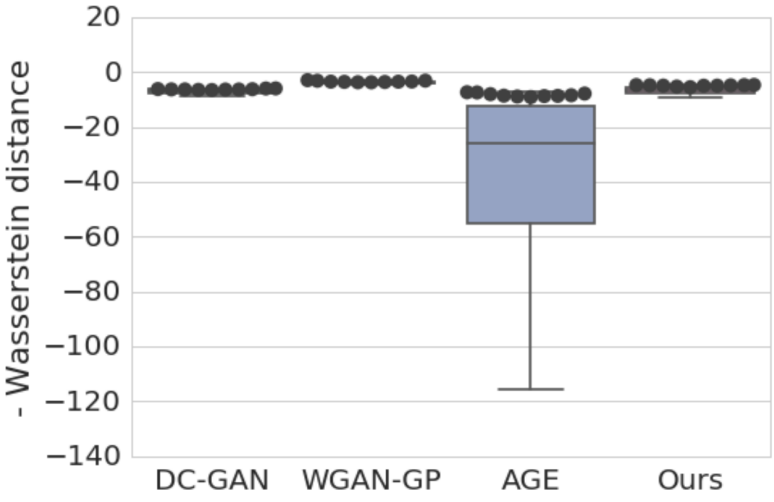


Full hybrid loss[1]:

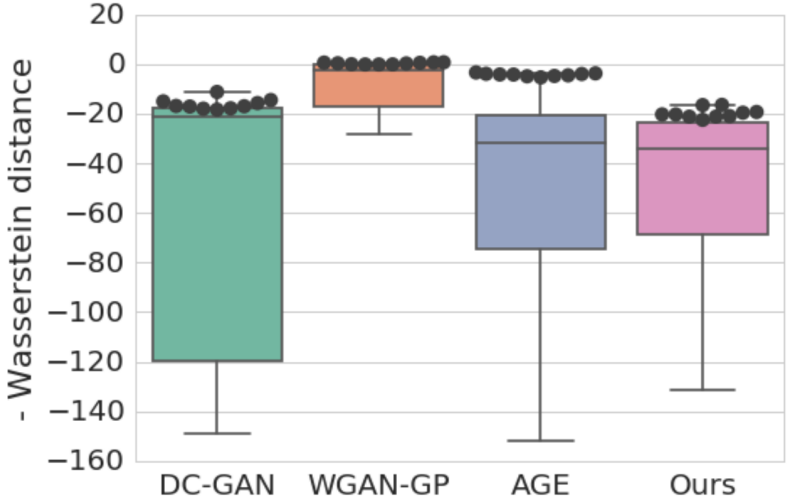
$$\mathbb{E}_{q_\eta(z|x)}\left[-\lambda\|x-G_\theta(z)\|_1+\log\frac{1-D_\phi(G_\theta(z))}{D_\phi(G_\theta(z))}+\log\frac{C_\omega(z)}{1-C_\omega(z)}\right].$$

[1] plus some tricks for better training

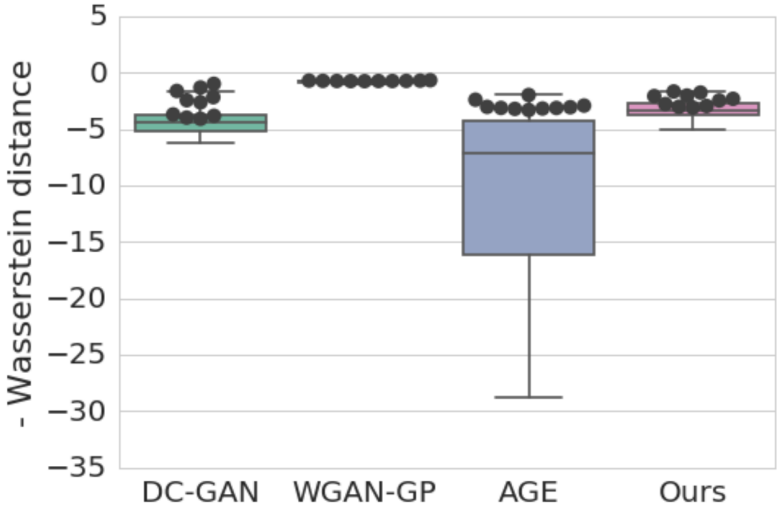
Results: Wasserstein distance



ColorMNIST

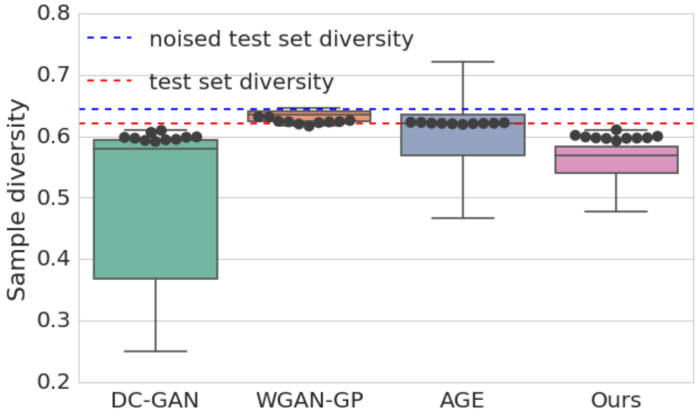


CelebA

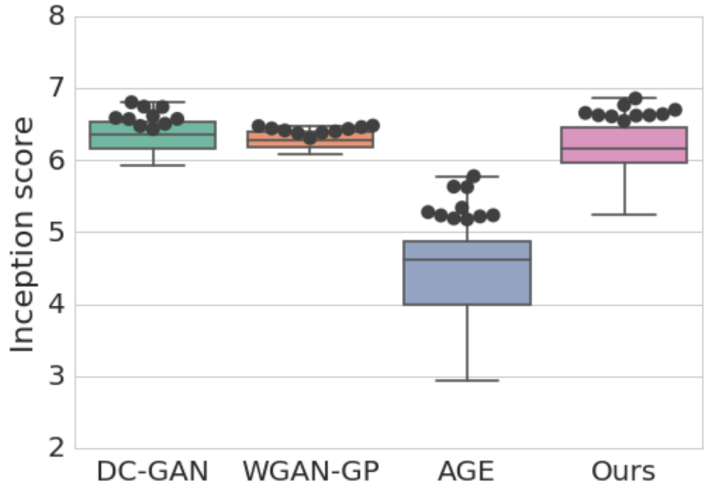


CIFAR-10

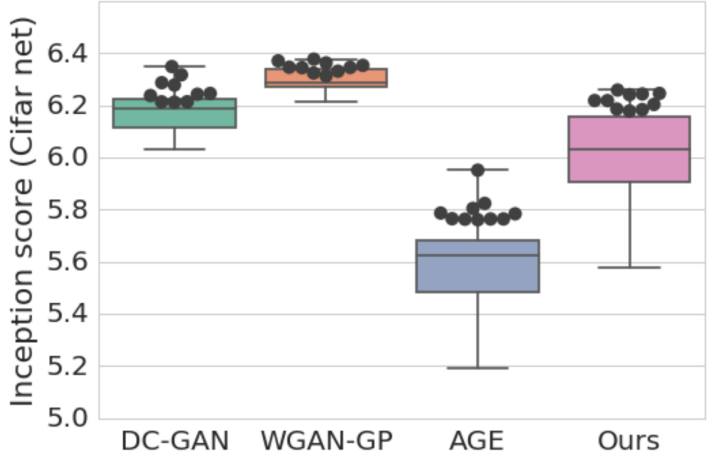
Samples diversity



ColorMNIST



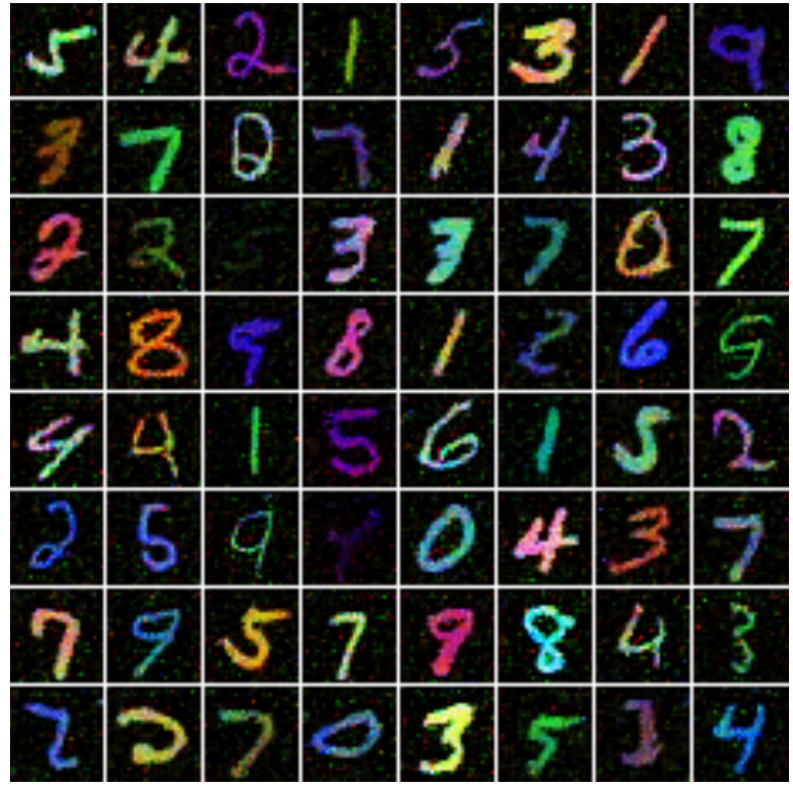
CelebA



CIFAR-10

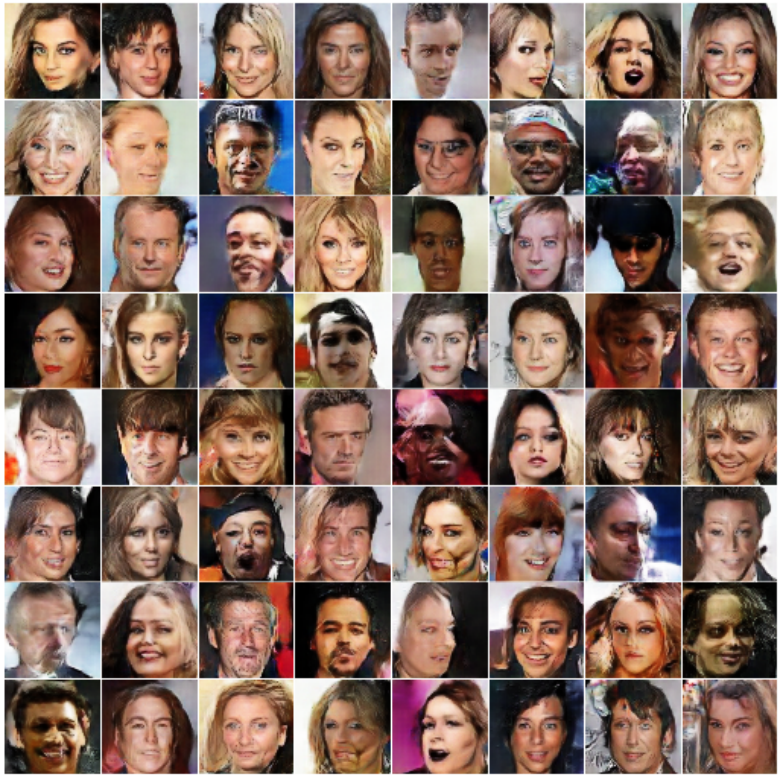
Examples for ColorMNIST

DCGAN

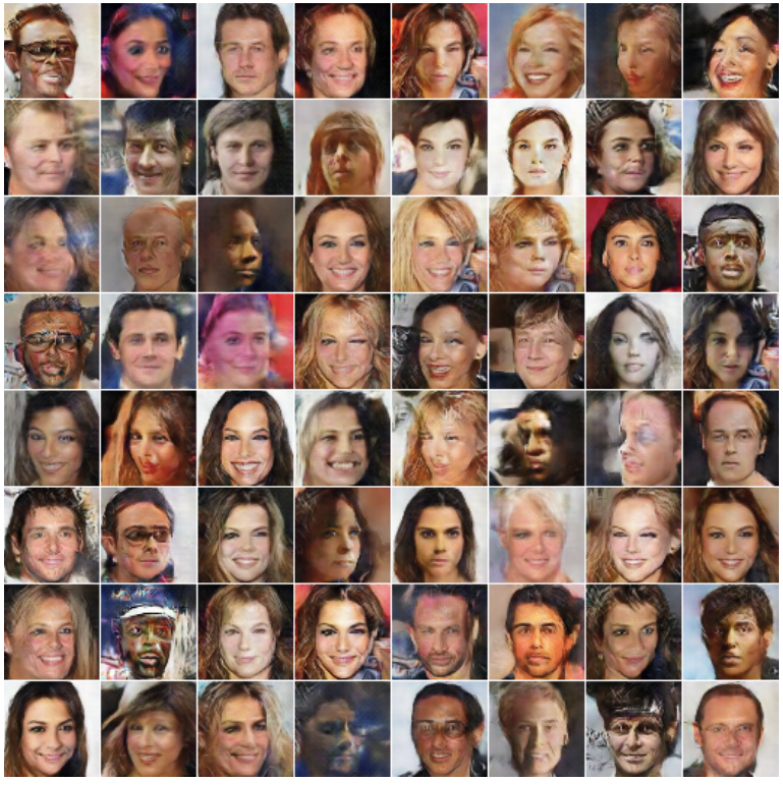


α-GAN

Examples CelebA



AGE



α-GAN

α-GAN: bottom-line

- GAN architecture with good performance.
- Interesting approach for combining VAE and GAN.
- Still necessary to use tricks for learning.

Bayesian GAN

Architecture

$x \sim p^*(x)$ — data distribution.
$z \sim p(z)$ — latent variables distribution.

$G(z; \theta_g)$ — generator with parameters θ_g.
$D(x; \theta_d)$ — discriminator with parameters θ_d.
$p(\theta_g), p(\theta_d)$ — distributions we want to learn.

Generation procedure:

- $\theta_g \sim p(\theta_g)$;
- $z \sim p(z)$;
- $x_{fake} = G(z; \theta_g)$.

Posterior distributions

$$p(\theta_g \mid \mathbf{z}, \theta_d) \propto \left(\prod_i D(G(z^{(i)}; \theta_g); \theta_d) \right) \times p_0(\theta_g).$$

Intuition: push up parameters that fool discriminator.

$$p(\theta_d \mid \mathbf{z}, \mathbf{X}, \theta_g) \propto \left(\prod_i D(x^{(i)}; \theta_d) \right) \times \\ \times \left(\prod_i (1 - D(G(z^{(i)}; \theta_g); \theta_d)) \right) \times p_0(\theta_d).$$

Intuition: push up parameters that better distinguish real from fake.

$\mathbf{z}, \mathbf{X}$ — latent codes and data batches.
$p_0(\theta_g), p_0(\theta_d)$ — prior distributions.

Learning

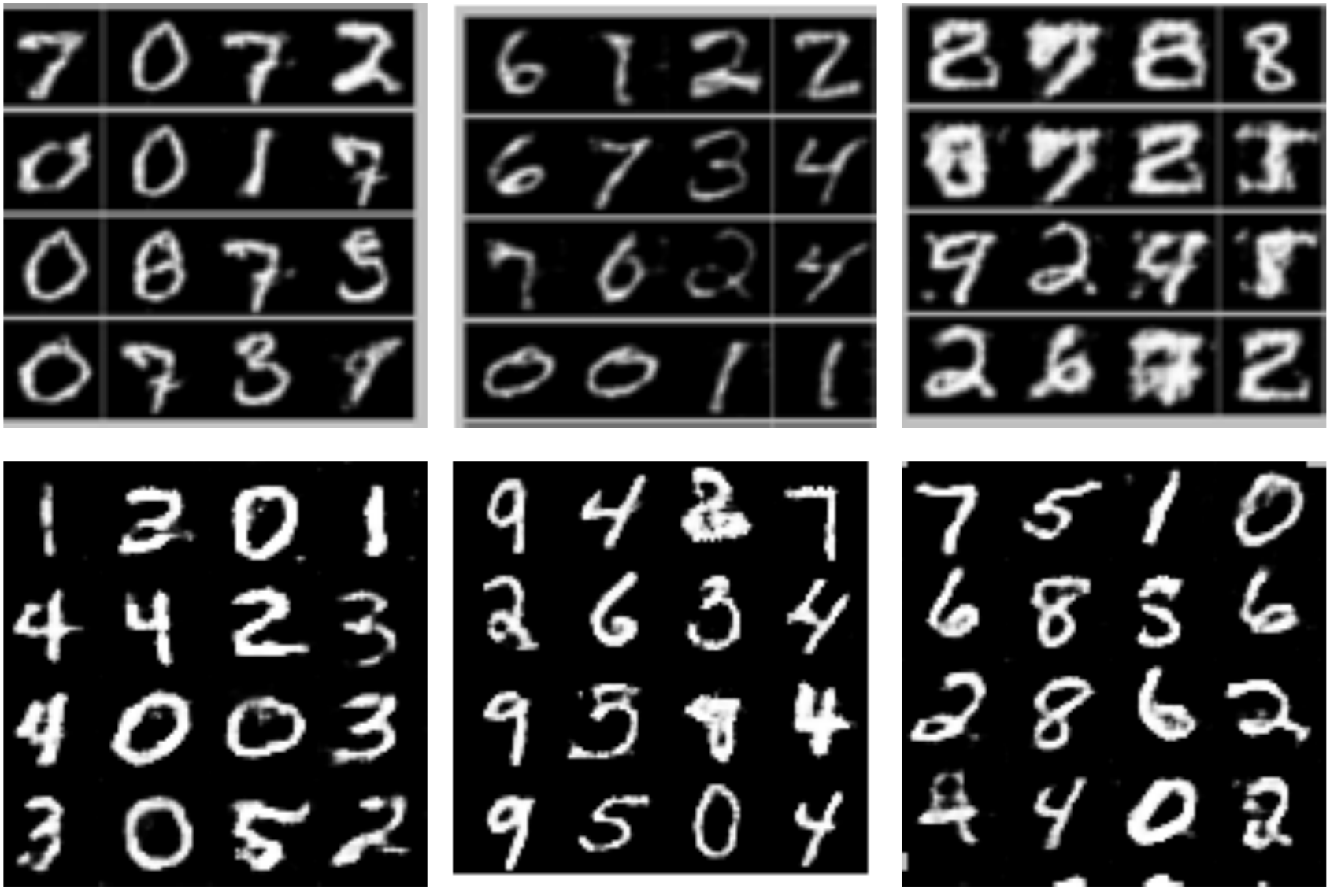
Algorithm:

- Start from some random θ_d^0.
- Repeat for N iterations:
  - Sample $\theta_g^k \sim p(\theta_g \mid \mathbf{z}, \theta_d^{k-1})$.
    ($\mathbf{z}$ — latent code minibatch).
  - Sample $\theta_d^k \sim p(\theta_d \mid \mathbf{z}, \mathbf{X}, \theta_g^{k-1})$.
    ($\mathbf{z}$ — latent code minibatch, $\mathbf{X}$ — data minibatch).
- Output $\{\theta_g^k\}_{k=1}^N, \{\theta_d^k\}_{k=1}^N$.

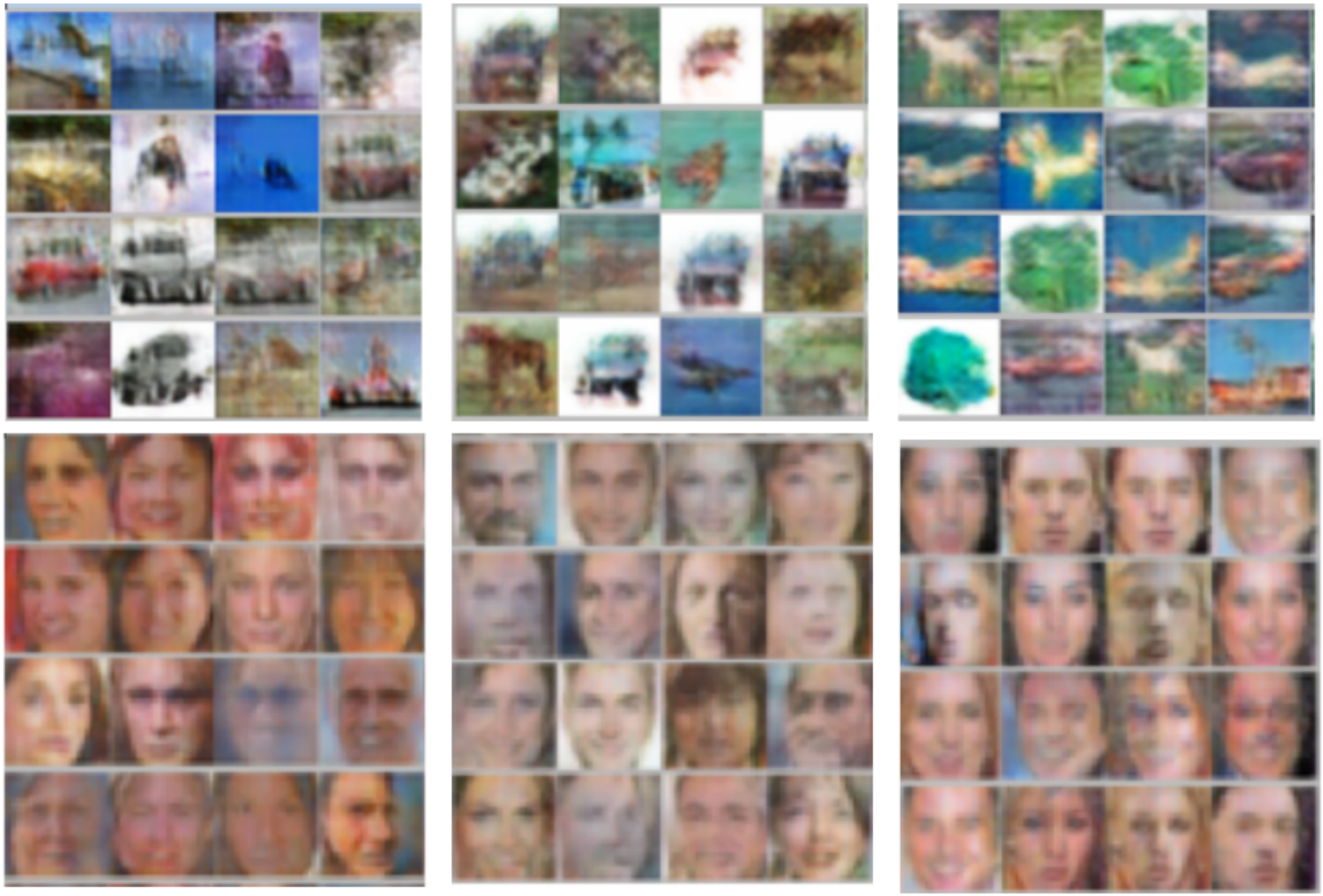
Note:

- $p(\theta_g), p(\theta_d)$ are represented with $\{\theta_g^k\}_{k=1}^N, \{\theta_d^k\}_{k=1}^N$.
- Sampling is done by several iterations of SGHMC.

Examples: MNIST (BayesianGAN / DCGAN)



Examples: CIFAR-10, CelebA (BayesianGAN)



Semi-supervised learning

Problem: K-class classification.

Learning:

Unlabeled observations $\{x^{(i)}\}_{i=1}^{n}$.

Labeled observations $\{x_s^{(i)}, y_s^{(i)}\}_{i=1}^{n_s}$.

Typically $n \gg n_s$.

Goal: Use unlabeled observations to learn x structure and improve classification.

Semi-supervised learning using GAN

Discriminator $D(x) \to \{0, \ldots, K\}$.

- Class 0: fake samples (from generator);
- Classes $1, \ldots, K$: real classes.

Generator loss:

$$\mathbb{E}_z \log p_D(G(z) = 0).$$

Discriminator loss:

$$\mathbb{E}_x \log p_D(x = 0) + \mathbb{E}_{x_s, y_s} \log p_D(x_s \neq y_s) + \mathbb{E}_z \log p_D(G(z) \neq 0).$$

Semi-supervised posterior distributions

$$p(\theta_g \mid \mathbf{z}, \theta_d) \propto \left(\prod_i \sum_{y=1}^{K} D(G(z^{(i)}; \theta_g) = y; \theta_d) \right) \times p_0(\theta_g).$$

Intuition: push up parameters that fool discriminator.

$$p(\theta_d \mid \mathbf{z}, \mathbf{X}, \mathbf{x}_s, \mathbf{y}_s, \theta_g) \propto \left(\prod_i \sum_{y=1}^{K} D(x^{(i)} = y; \theta_d) \right) \times$$

$$\times \left(\prod_i D(G(z^{(i)}; \theta_g) = 0; \theta_d) \right) \times \left(\prod_i D(x_s^{(i)} = y_s^{(i)}; \theta_d) \right) \times p_0(\theta_d).$$

Intuition: push up parameters that

- Assign any non-fake class to unlabeled data;
- Assign fake class to generated data;
- Assign correct class to labeled data.

Semi-supervised learning: results

n_s	No. of misclassifications for MNIST. Test error rate for others.				
	Supervised	DCGAN	W-DCGAN	DCGAN-10	BayesGAN
MNIST	N=60k, $D = (28, 28)$	14	15	114	153
20	—	1823 ± 412	1687 ± 387	1087 ± 564	$\mathbf{997 \pm 348}$
50	—	453 ± 110	490 ± 170	189 ± 103	$\mathbf{154 \pm 89}$
100	2134 ± 525	128 ± 11	156 ± 17	97 ± 8.2	$\mathbf{89 \pm 3.4}$
200	1389 ± 438	95 ± 3.2	91 ± 5.2	$\mathbf{78 \pm 2.8}$	79 ± 1.4
CIFAR10	N=50k, $D = (32, 32, 3)$	18	19	146	205
1000	63.4 ± 2.6	58.2 ± 2.8	57.1 ± 2.4	31.1 ± 2.5	$\mathbf{29.7 \pm 3.2}$
2000	56.1 ± 2.1	47.5 ± 4.1	49.8 ± 3.1	29.2 ± 1.2	$\mathbf{25.4 \pm 4.3}$
4000	51.4 ± 2.9	40.1 ± 3.3	38.1 ± 2.9	27.4 ± 3.2	$\mathbf{22.8 \pm 2.4}$
8000	47.2 ± 2.2	29.3 ± 2.8	27.4 ± 2.5	25.5 ± 2.4	$\mathbf{20.9 \pm 1.8}$
SVHN	N=75k, $D = (32, 32, 3)$	29	31	217	322
500	53.5 ± 2.5	31.2 ± 1.8	29.4 ± 1.8	27.1 ± 2.2	$\mathbf{26.7 \pm 2.8}$
1000	37.3 ± 3.1	25.5 ± 3.3	25.1 ± 2.6	18.3 ± 1.7	$\mathbf{14.1 \pm 2.3}$
2000	26.3 ± 2.1	22.4 ± 1.8	23.3 ± 1.2	16.7 ± 1.8	$\mathbf{14.0 \pm 1.4}$
4000	20.8 ± 1.8	20.4 ± 1.2	19.4 ± 0.9	14.0 ± 1.4	$\mathbf{9.8 \pm 1.1}$
CelebA	N=100k, $D = (50, 50, 3)$	103	98	649	1134
1000	53.8 ± 4.2	52.3 ± 4.2	51.2 ± 5.4	47.3 ± 3.5	$\mathbf{43.9 \pm 2.9}$
2000	36.7 ± 3.2	37.8 ± 3.4	39.6 ± 3.5	31.2 ± 1.8	$\mathbf{30.8 \pm 2.3}$
4000	34.3 ± 3.8	31.5 ± 3.2	30.1 ± 2.8	29.3 ± 1.5	$\mathbf{25.4 \pm 1.3}$
8000	31.1 ± 4.2	29.5 ± 2.8	27.6 ± 4.2	26.4 ± 1.1	$\mathbf{22.3 \pm 1.8}$

BayesianGAN: bottom-line

- Good approach for fighting mode collapse and instability.
- Good performance for semi-supervised learning.
- Low image quality (for GANs).
- Slow learning.