

Прикладные задачи анализа данных

Лекция 3
Генеративно-состязательные сети

Михаил Гущин
mhushchyn@hse.ru

НИУ ВШЭ, 2021



НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ
УНИВЕРСИТЕТ

План

- ▶ Приложения
- ▶ Генеративно-состязательные сети (GANы)
- ▶ Проблемы GANов
- ▶ Wasserstein GAN
- ▶ Другие архитектуры

Приложения



This X Does Not Exist



This Person Does Not Exist

The site that started it all, with the name that says it all. Created using a style-based generative adversarial network (StyleGAN), this website had the tech community buzzing with excitement and intrigue and inspired many more sites.

Created by Phillip Wang.



This Cat Does Not Exist

These purr-fect GAN-made cats will freshen your feeline-gs and make you wish you could reach through your screen and cuddle them. Once in a while the cats have visual deformities due to imperfections in the model – beware, they can cause nightmares.

Created by Ryan Hoover.



This Rental Does Not Exist

Why bother trying to look for the perfect home when you can create one instead? Just find a listing you like, buy some land, build it, and then enjoy the rest of your life.

Created by Christopher Schmidt.

<https://thisxdoesnotexist.com>

Реалистичные фотографии



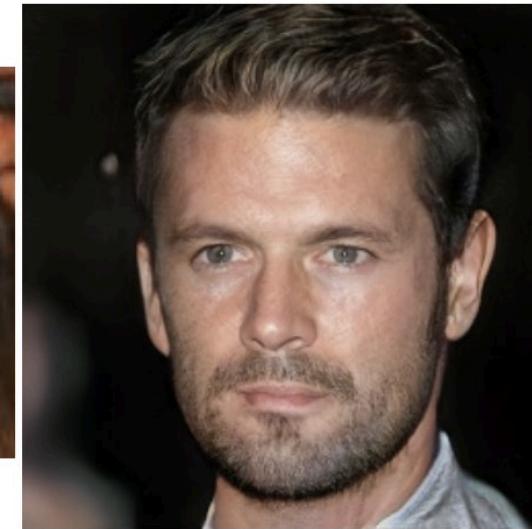
2014



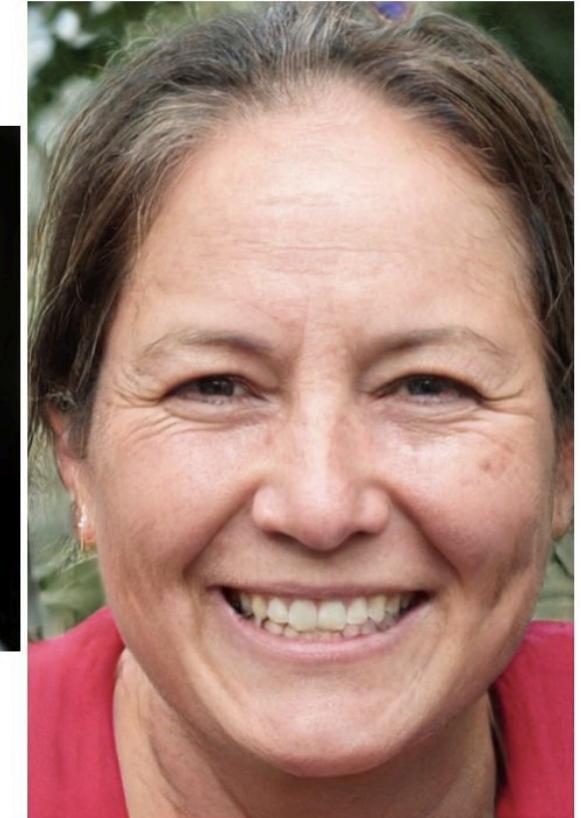
2015



2016



2017



2018

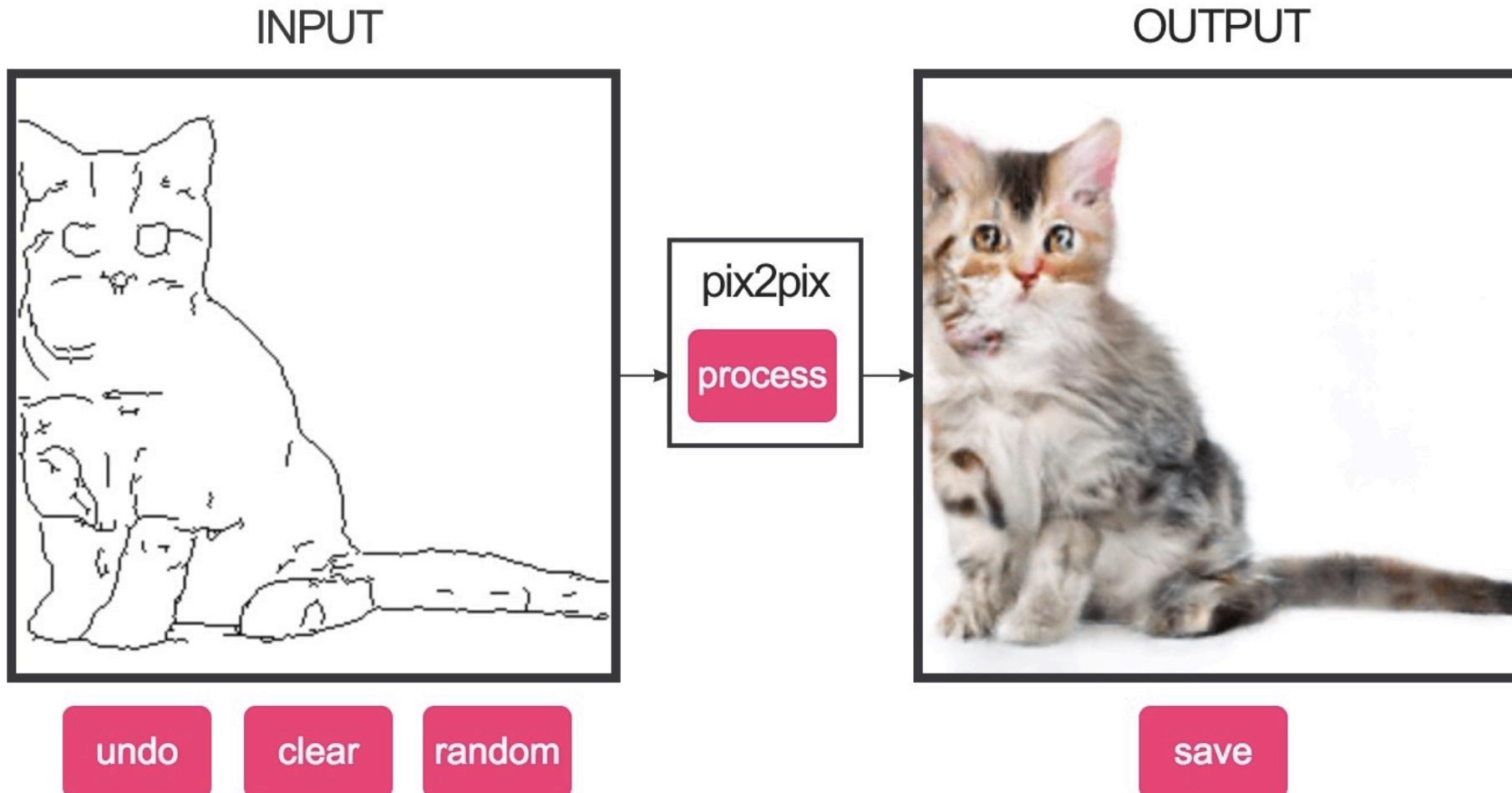
https://twitter.com/goodfellow_ian/status/1084973596236144640?s=20

Deepfake



<https://www.youtube.com/c/CtrlShiftFace>

Image-to-Image Translation



<https://affinelayer.com/pixsrv/>

Другие примеры

- ▶ Style Transfer
- ▶ Super Resolution
- ▶ Photo Inpainting
- ▶ 3D Object Generation
- ▶ Text-to-Image Translation
- ▶ и другие

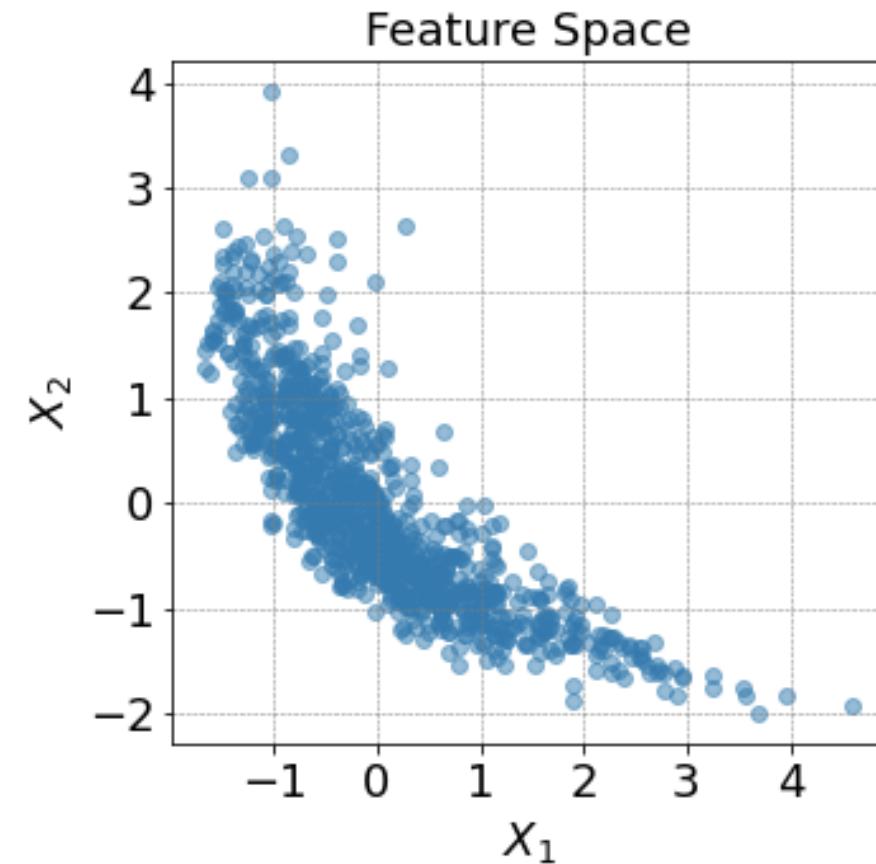
<https://machinelearningmastery.com/impressive-applications-of-generative-adversarial-networks/>

Генеративно-состязательные сети



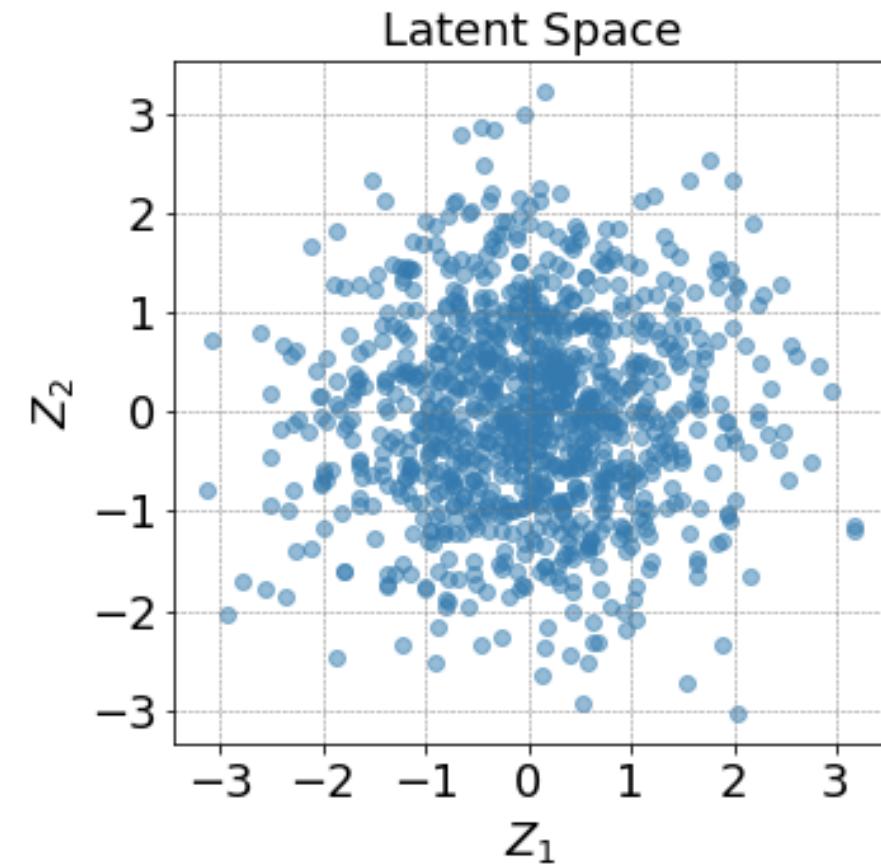
Постановка задачи

- ▶ **Дано:** выборка объектов $\mathbf{x}_i \sim p_x(x)$, где распределение $p_x(x)$ неизвестно.
- ▶ **Задача:** сгенерировать новые объекты $\hat{\mathbf{x}}_j \sim p_x(x)$ с таким же распределением, т.е. похожими на \mathbf{x}_i .

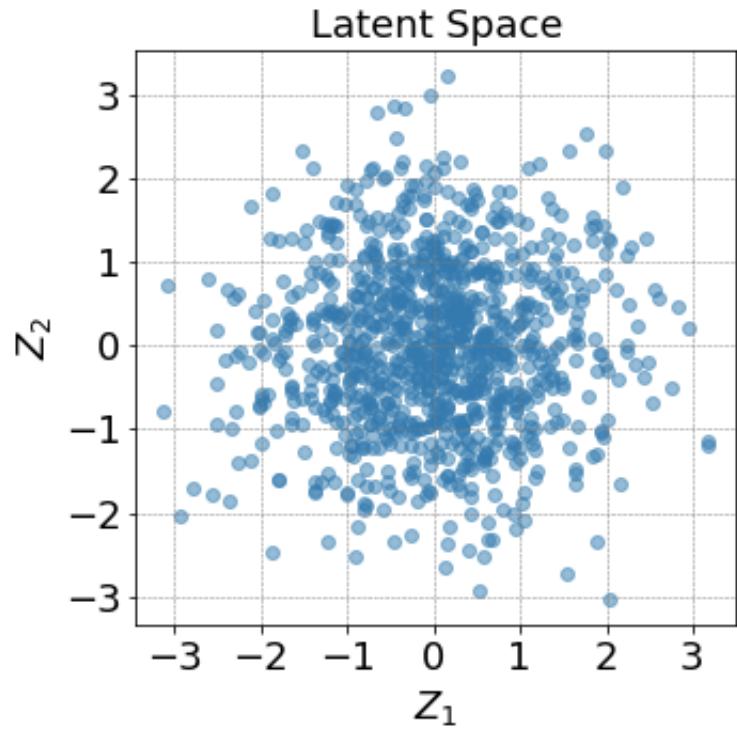


Известные распределения

- ▶ Мы умеем генерировать объекты из известных распределений.
- ▶ Сгенерируем выборку объектов $\mathbf{z}_i \sim p_z(\mathbf{z})$, где $p_z(z)$ - двумерное нормальное распределение.
- ▶ Пространство \mathbf{Z} называется **скрытым**.



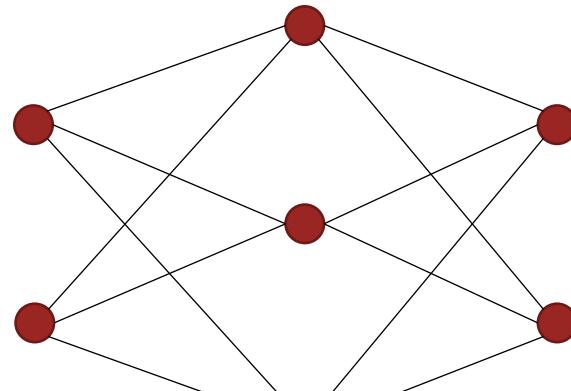
Генератор



Z

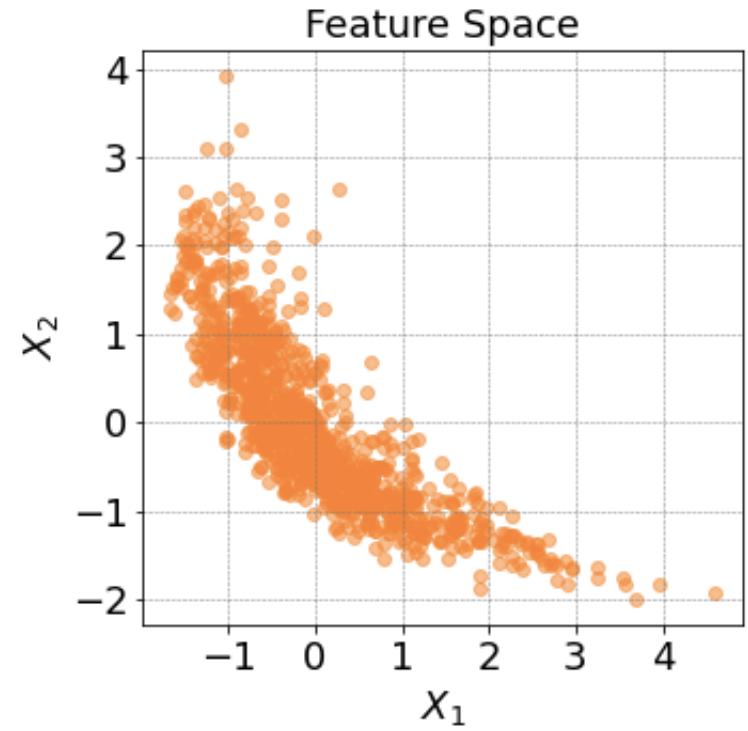
Случайный шум

Нейронная сеть



$G(Z)$

Генератор



$\hat{X} = G(Z)$

Сгенерированные
объекты

Генератор

- ▶ X – матрица реальных объектов.
- ▶ $\hat{X} = G(Z)$ – матрица сгенерированных объектов.
- ▶ Хотим, чтобы объекты из \hat{X} были похожи на объекты из X .
- ▶ Как посчитать сходство между \hat{X} и X ?
- ▶ Как обучить генератор $G(Z)$?

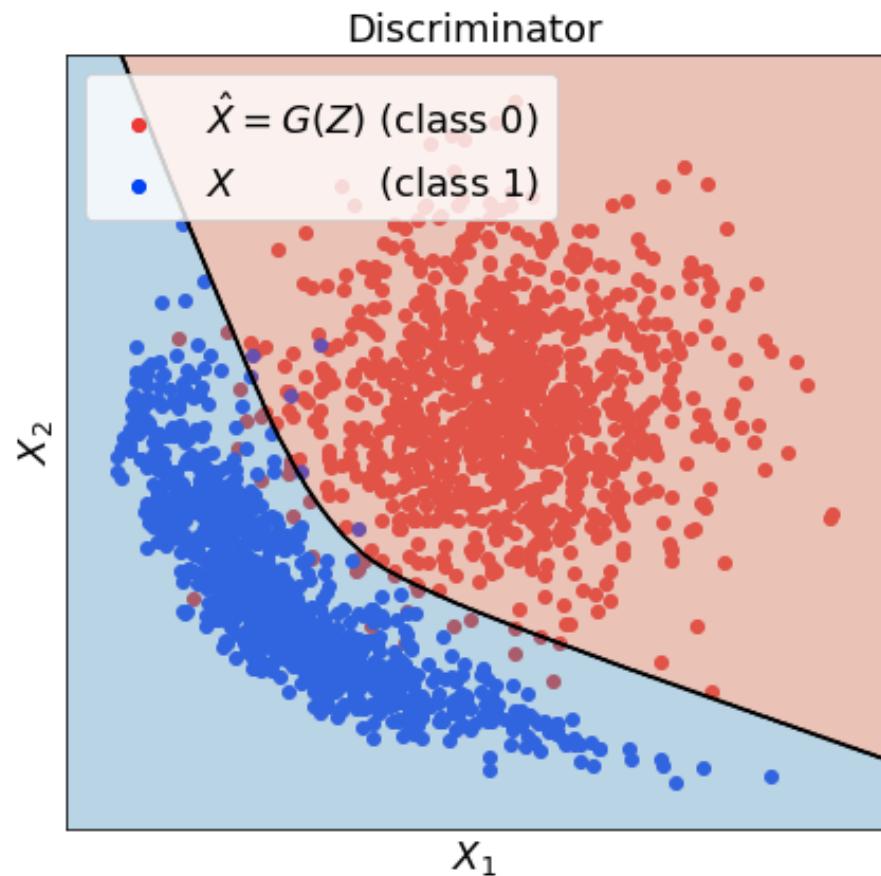
Дискриминатор

Давайте рассмотрим задачу **бинарной классификации**:

- ▶ пусть X – класс 1,
- ▶ пусть \hat{X} – класс 0.

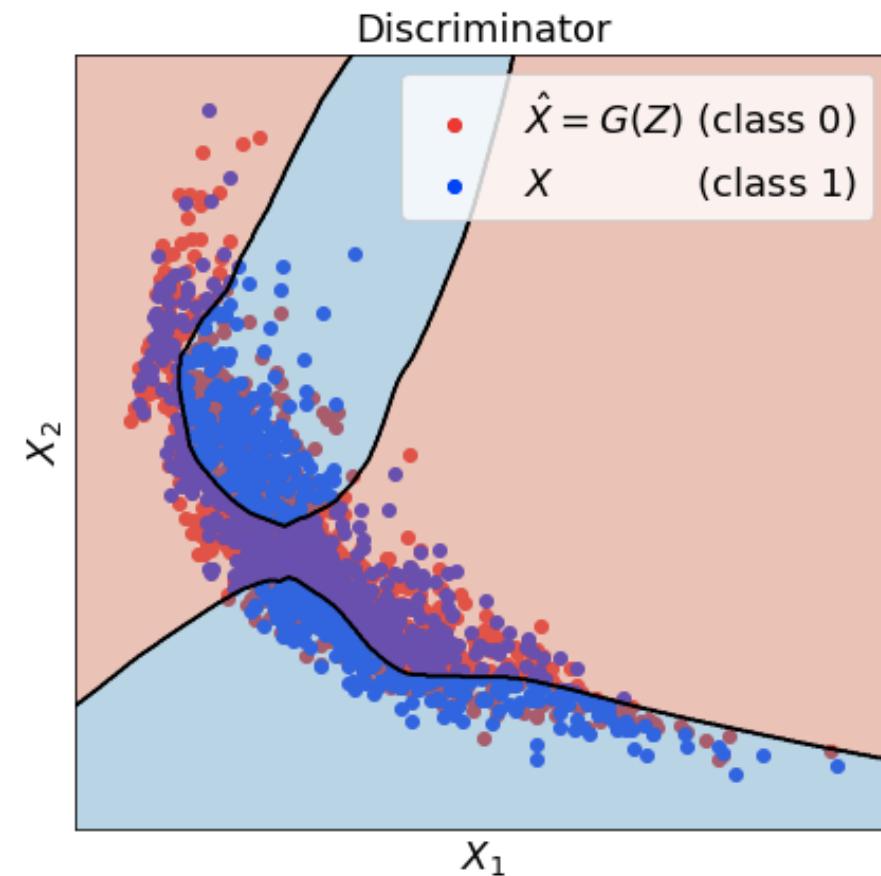
Задача классификатора – разделить два класса. Будем называть его **дискриминатором**.

Дискриминатор



ROC AUC = 0.995

Log Loss = 0.135



ROC AUC = 0.554

Log Loss = 0.689

Дискриминатор

В классификации используется **функция потерь L** :

$$L = -\frac{1}{n} \sum_{i=1}^n y_i \log D(x_i) + (1 - y_i) \log(1 - D(x_i))$$

где $D(x_i)$ – выход дискриминатора, $y_i \in \{0, 1\}$ – метка объекта.

Будем использовать значение этой функции как меру сходства объектов двух классов.

Дискриминатор

Перепишем функцию потерь:

$$L = -\frac{1}{n} \sum_{i:y_i=1} \log D(x_i) - \frac{1}{n} \sum_{i:y_i=0} \log(1 - D(x_i))$$

Здесь каждая сумма берется по своему классу.

Дискриминатор

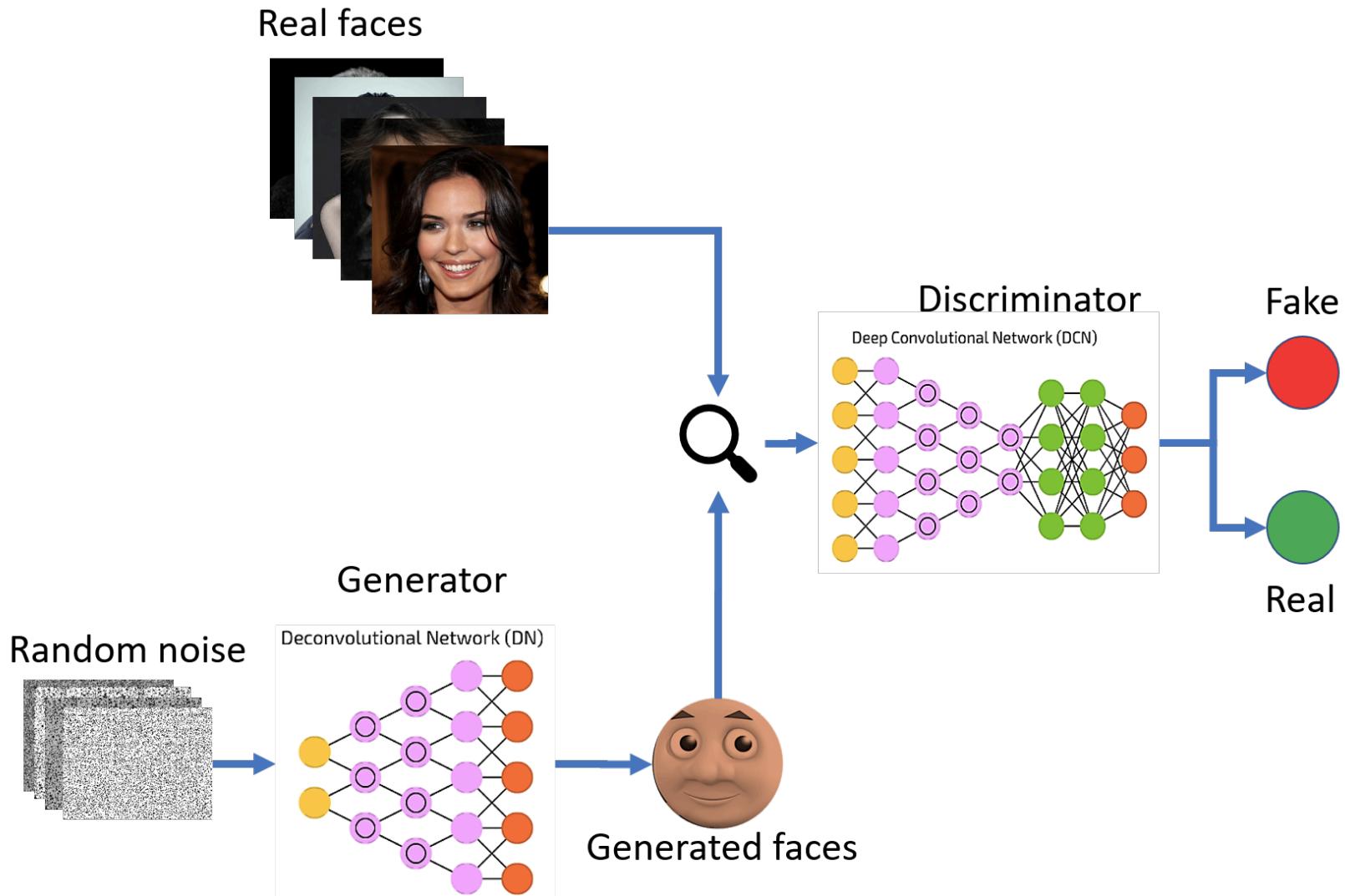
- ▶ Мы договорились, что X – класс 1, а $\hat{X} = G(Z)$ – класс 0.
- ▶ Тогда функция потерь имеет вид:

$$L(\mathbf{D}, \mathbf{G}) = -\frac{1}{n} \sum_{x_i \in X} \log \mathbf{D}(x_i) - \frac{1}{n} \sum_{z_i \in Z} \log(1 - \mathbf{D}(\mathbf{G}(z_i)))$$

- ▶ Обучение дискриминатора и генератора:

$$\max_{\mathbf{G}} \min_{\mathbf{D}} L(\mathbf{D}, \mathbf{G})$$

Алгоритм



Link: <https://www.spindox.it/en/blog/generative-adversarial-neural-networks/>

Алгоритм

Algorithm 1 Minibatch stochastic gradient descent training of generative adversarial nets. The number of steps to apply to the discriminator, k , is a hyperparameter. We used $k = 1$, the least expensive option, in our experiments.

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:

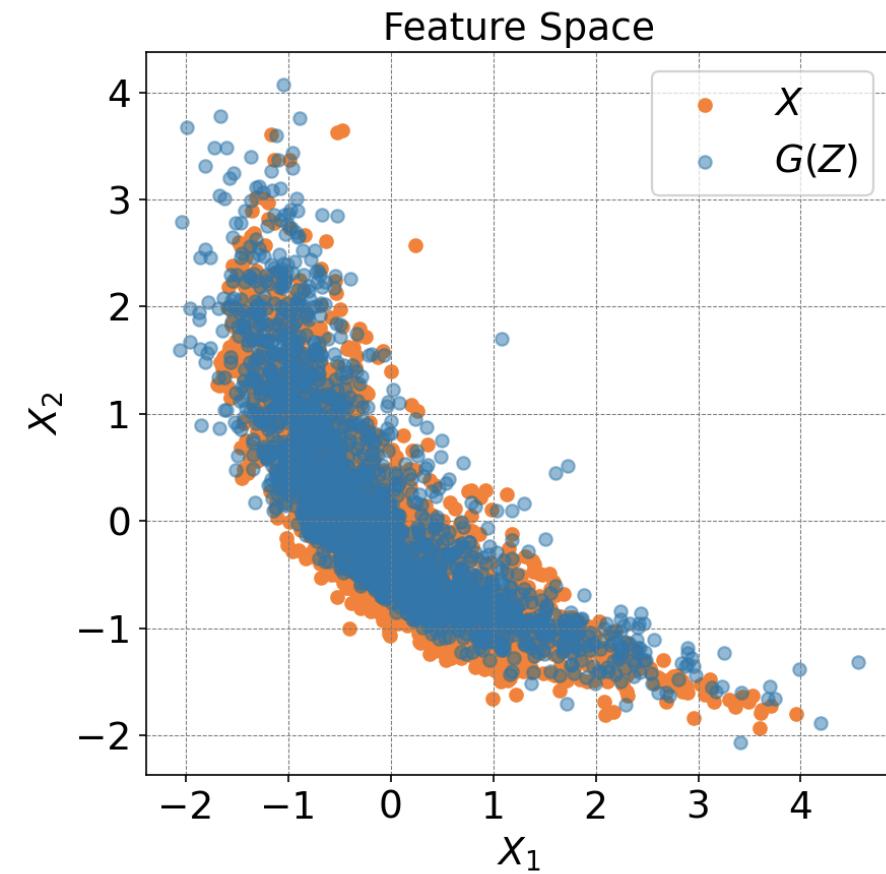
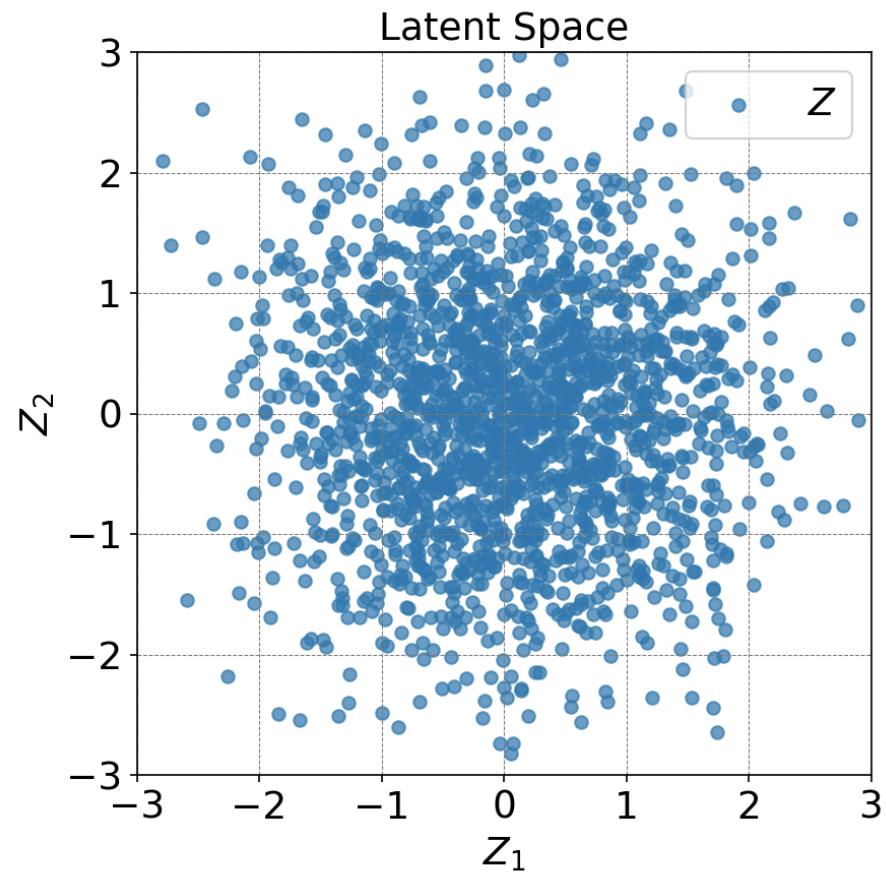
$$\nabla_{\theta_g} \frac{1}{m} \sum_{i=1}^m \log (1 - D(G(\mathbf{z}^{(i)}))).$$

end for

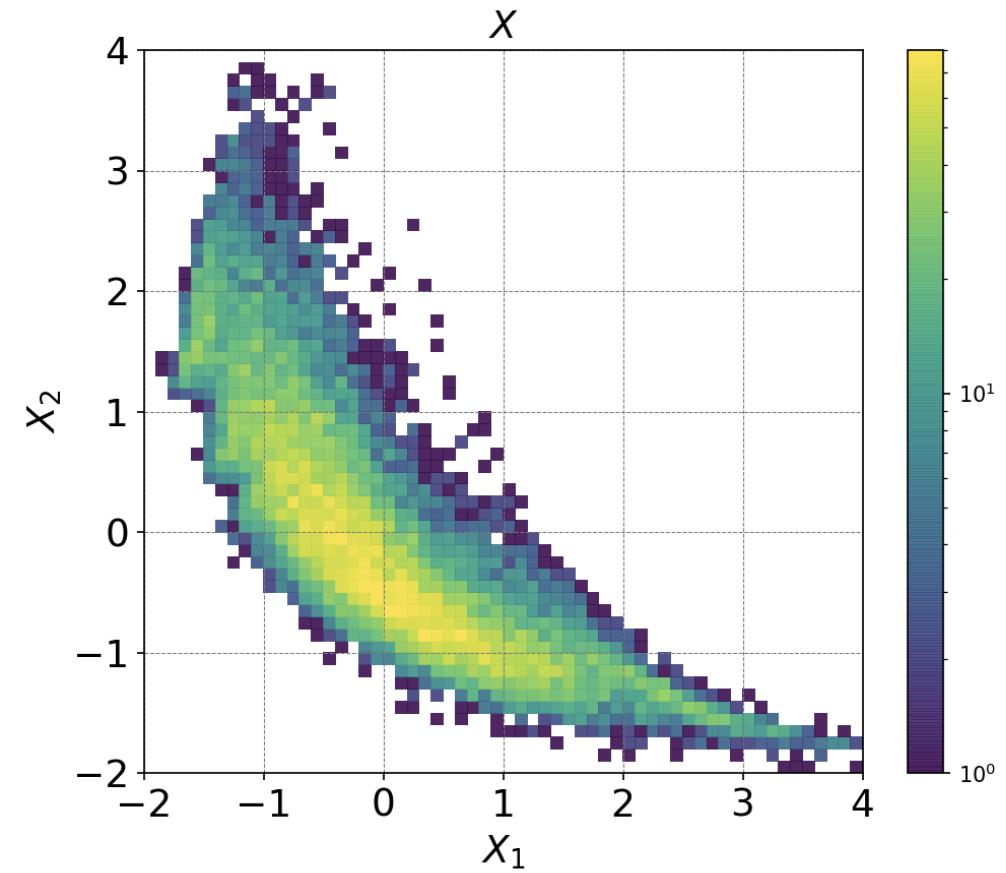
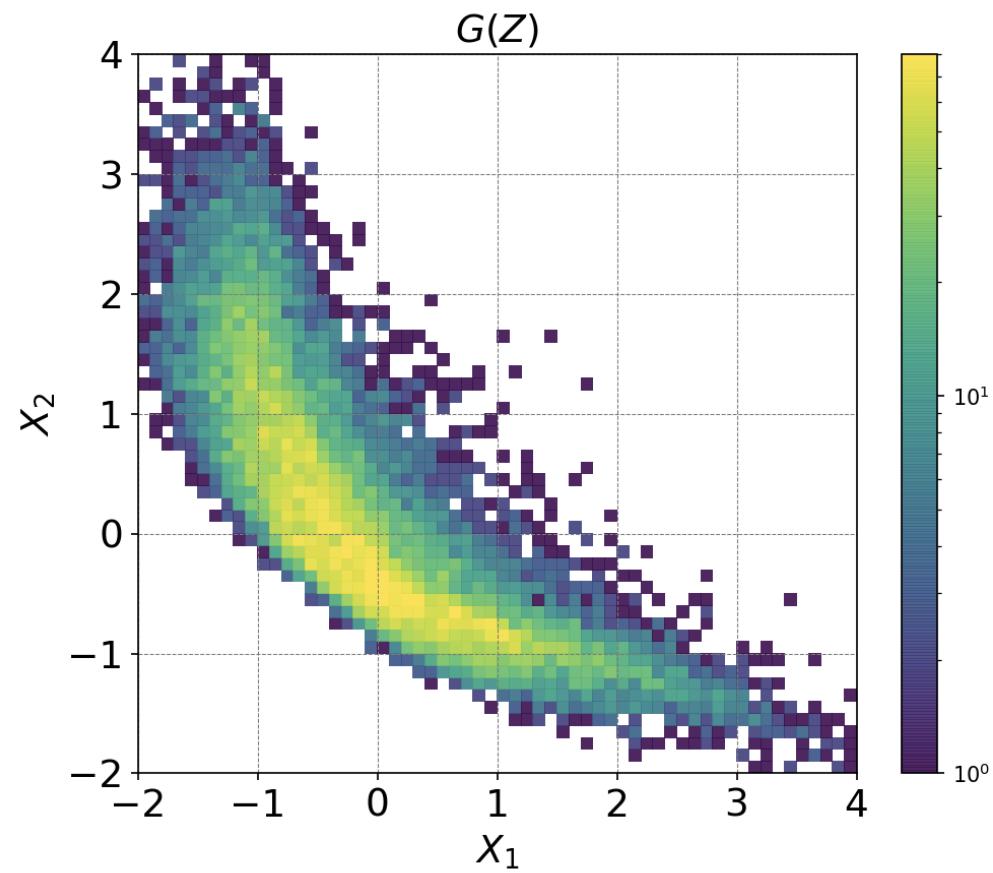
The gradient-based updates can use any standard gradient-based learning rule. We used momentum in our experiments.

<https://arxiv.org/abs/1406.2661>

Пример



Пример

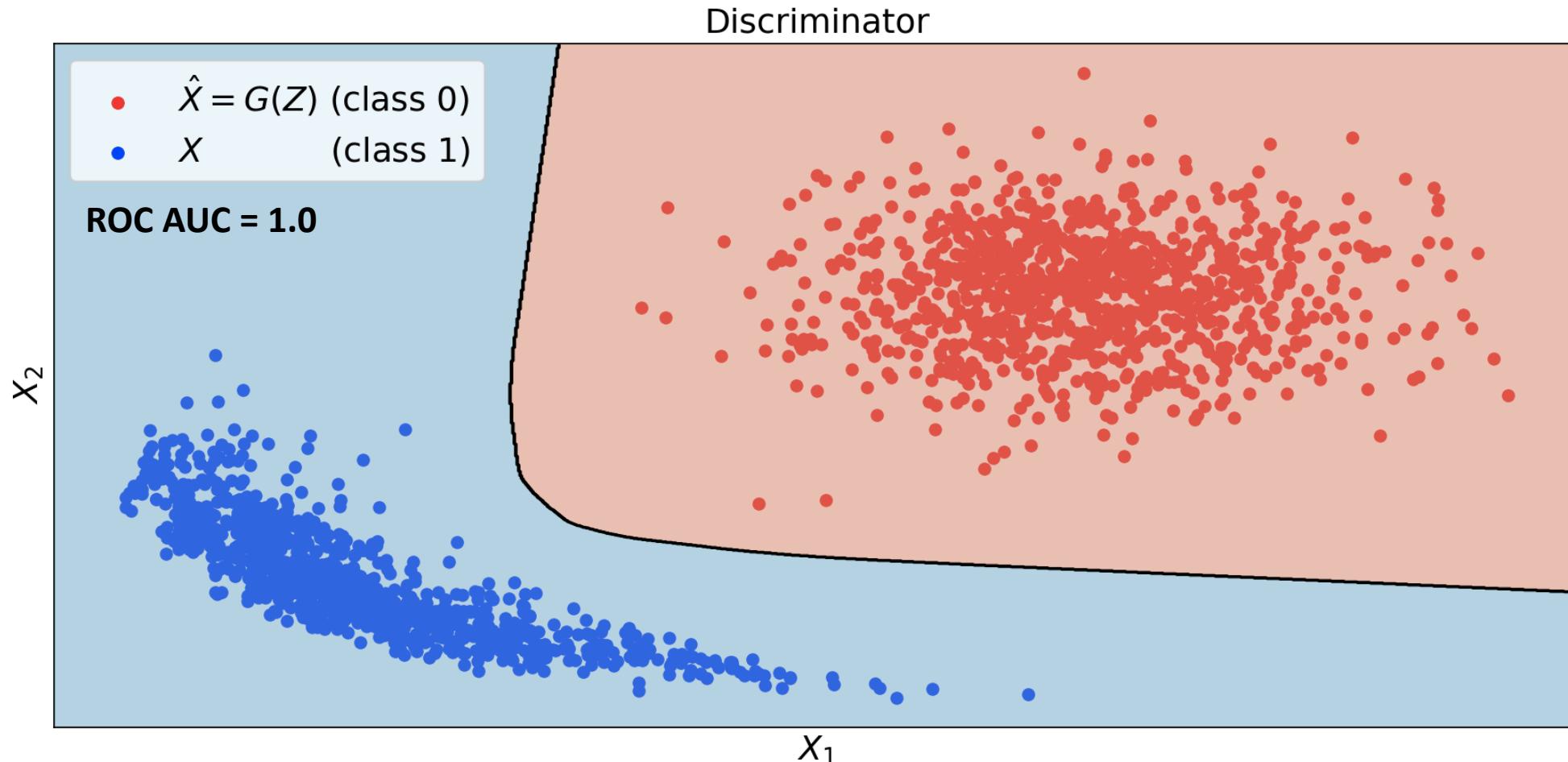


Проблемы GANов



Затухание градиентов

Какая-то итерация обучения GAN:



Затухание градиентов

- ▶ Классы идеально разделяются дискриминатором =>
 - ▶ $\mathbf{D}(x_i) = \mathbf{1}$ для любого x_i в X ;
 - ▶ $\mathbf{D}(\hat{x}_j) = \mathbf{D}(\mathbf{G}(z_j)) = \mathbf{0}$ для любого z_j в Z .
-
- ▶ Подставим эти значения в функцию потерь:

$$L(\mathbf{D}, \mathbf{G}) = -\frac{1}{n} \sum_{x_i \in X} \log \mathbf{D}(x_i) - \frac{1}{n} \sum_{z_i \in Z} \log(1 - \mathbf{D}(\mathbf{G}(z_i)))$$

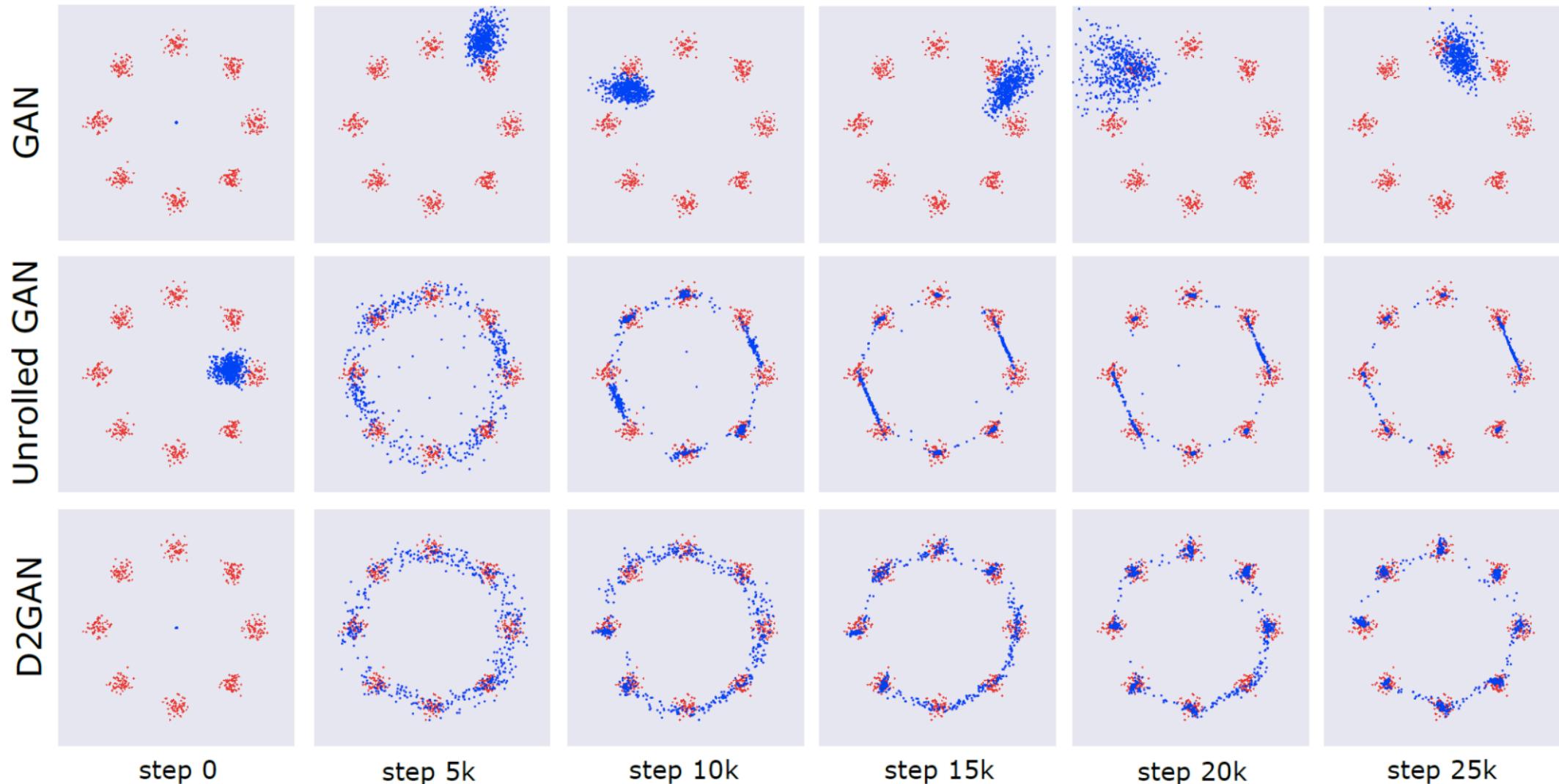
Затухание градиентов

- ▶ В результате получаем:

$$L(\mathbf{D}, \mathbf{G}) = -\frac{1}{n} \sum_{x_i \in X} \log \mathbf{1} - \frac{1}{n} \sum_{z_i \in Z} \log(1 - \mathbf{0}) = 0 = \text{const}$$

- ▶ $\nabla L(\mathbf{D}, \mathbf{G}) = 0 \Rightarrow$
- ▶ Дискриминатор и генератор больше **не обучаются!**

Схлопывание мод



Схлопывание мод

Причины:

- ▶ $D(x_i) = 1$ для x_i из некоторых мод =>
- ▶ $\nabla L(D, G) = 0$ в окрестности этих мод =>
- ▶ GAN не учится генерировать эти моды;
- ▶ Что-то еще, что все плохо понимают ☺

Wasserstein GAN

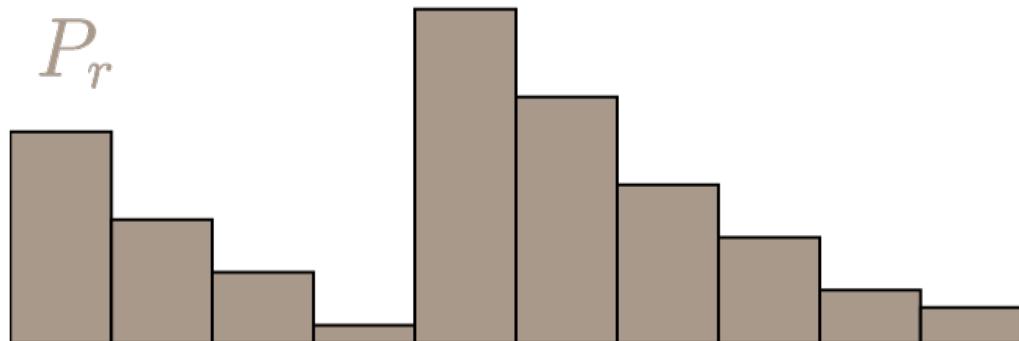


Мотивация

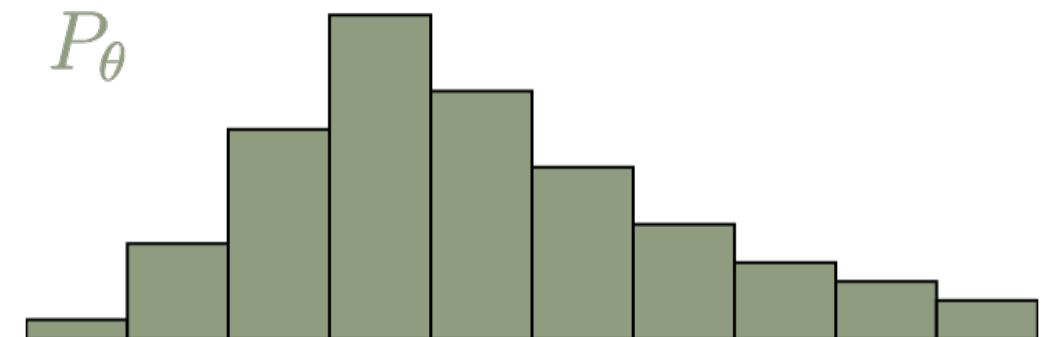
- ▶ Как решить проблему затухания градиентов?
- ▶ **Идея:** взять другую функцию потерь $L(\mathcal{D}, \mathcal{G})$!

Расстояние Вассерштейна

- ▶ Пусть даны две кучи земли P_r и P_θ ;
- ▶ Какую **минимальную работу** нужно совершить, чтобы перенести землю из одной кучи в другую?



Распределение
реальных объектов



Распределение
сгенерированных объектов

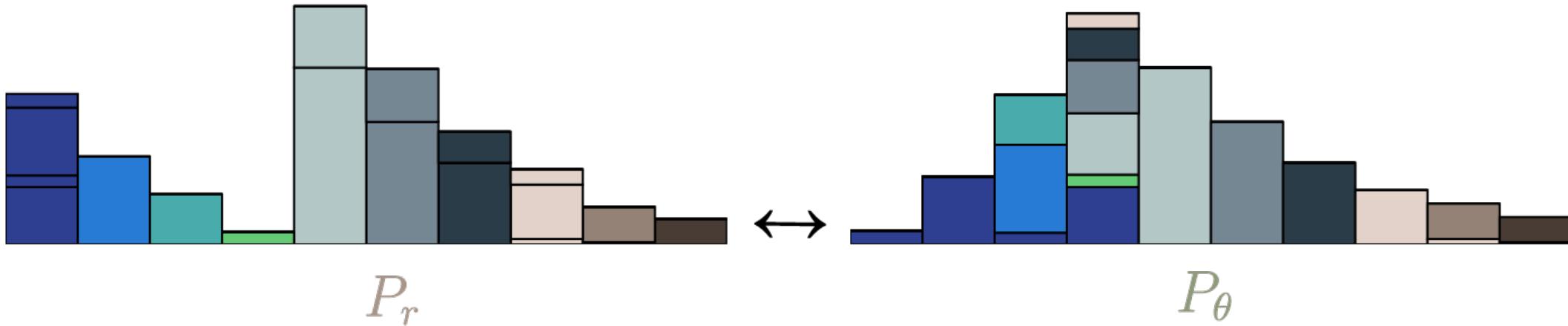
<https://vinctherrmann.github.io/blog/wasserstein/>

Расстояние Вассерштейна

- Определение (Earth Mover's Distance):

$$EMD(P_r, P_\theta) = \inf_{\gamma \in \Pi} \sum_{x,y} \|x - y\| \gamma(x, y)$$

План переноса



<https://vincentherrmann.github.io/blog/wasserstein/>

Расстояние Вассерштейна

- ▶ Дуальная форма:

$$EMD(P_r, P_\theta) = \sup_{\|\mathbf{f}\|_{L \leq 1}} \mathbb{E}_{x \sim P_r} \mathbf{f}(x) - \mathbb{E}_{x \sim P_\theta} \mathbf{f}(x)$$

- ▶ Где $\mathbf{f}(x)$ - некоторая функция, удовлетворяющая условию Липшица:

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

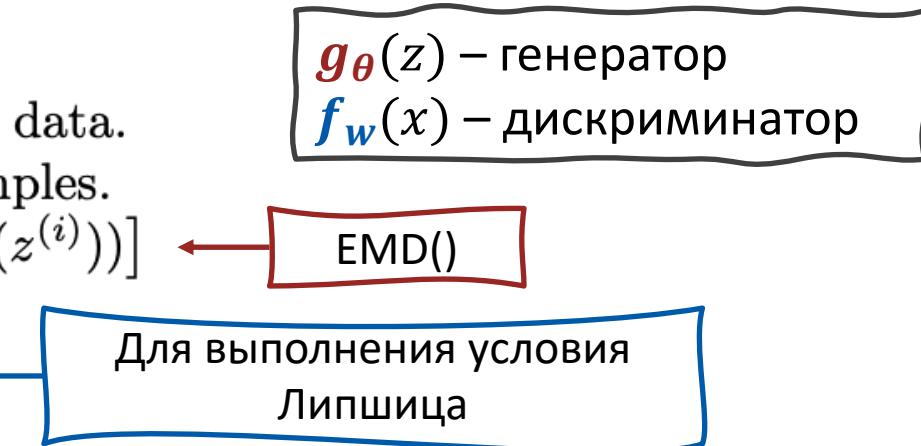
Wasserstein GAN

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.

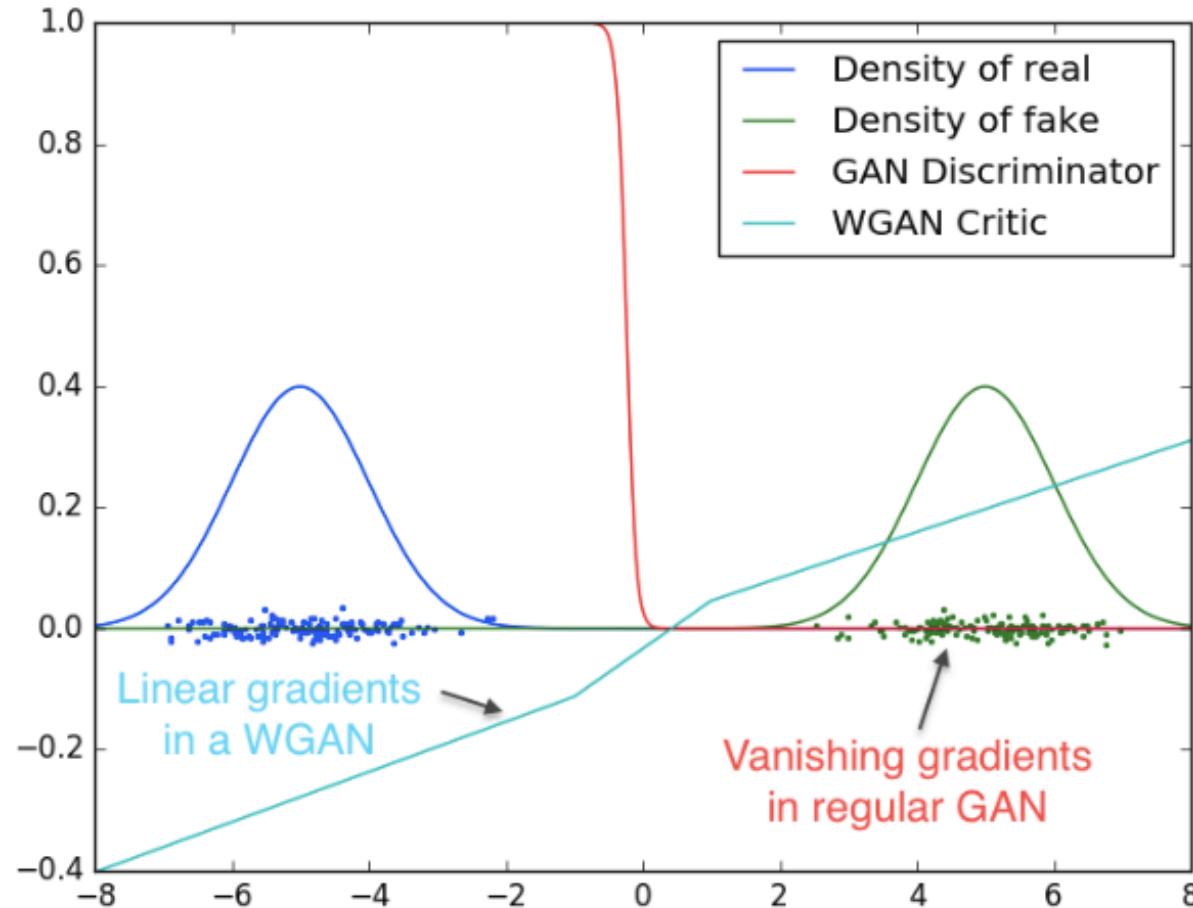
```
1: while  $\theta$  has not converged do
2:   for  $t = 0, \dots, n_{\text{critic}}$  do
3:     Sample  $\{x^{(i)}\}_{i=1}^m \sim \mathbb{P}_r$  a batch from the real data.
4:     Sample  $\{z^{(i)}\}_{i=1}^m \sim p(z)$  a batch of prior samples.
5:      $g_w \leftarrow \nabla_w \left[ \frac{1}{m} \sum_{i=1}^m f_w(x^{(i)}) - \frac{1}{m} \sum_{i=1}^m f_w(g_\theta(z^{(i)})) \right]$  ← EMD()
6:      $w \leftarrow w + \alpha \cdot \text{RMSProp}(w, g_w)$ 
7:      $w \leftarrow \text{clip}(w, -c, c)$  ← Для выполнения условия
                                         Липшица
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:   $\theta \leftarrow \theta - \alpha \cdot \text{RMSProp}(\theta, g_\theta)$ 
12: end while
```



$g_\theta(z)$ – генератор

$f_w(x)$ – дискриминатор

Пример



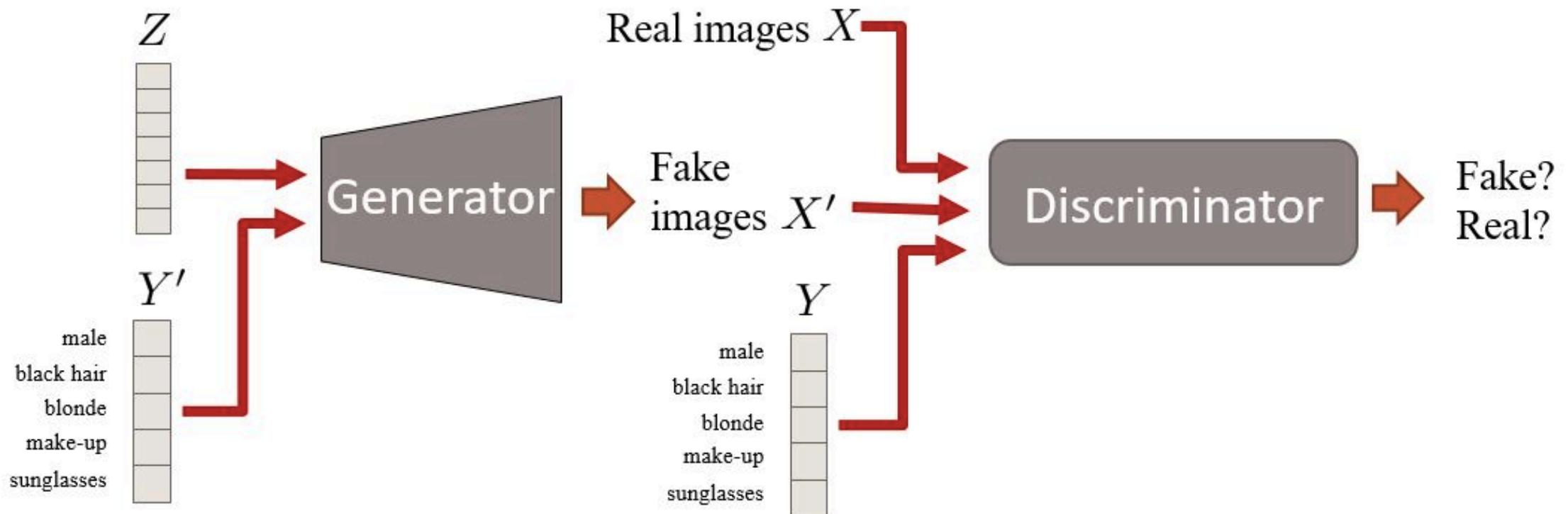
Преимущества WGAN

- ▶ Нет проблемы затухания градиентов
- ▶ Стабильное обучение
- ▶ Частично решает проблему схлопывания мод
- ▶ Одна из наиболее популярных архитектур

Другие архитектуры



Conditional GAN



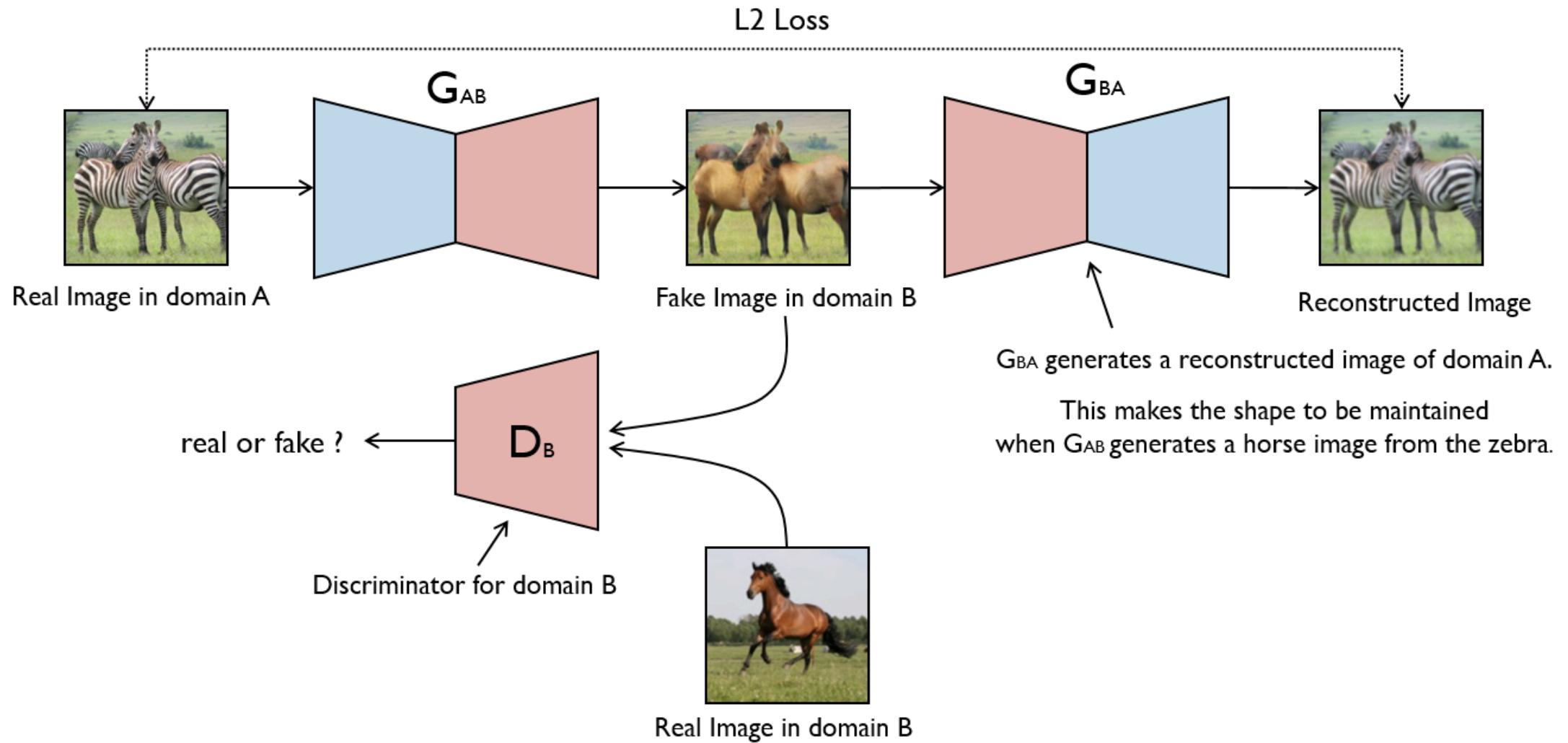
<https://guimperarnau.com/blog/2017/03/Fantastic-GANs-and-where-to-find-them>

Conditional GAN

Условием Y может являться:

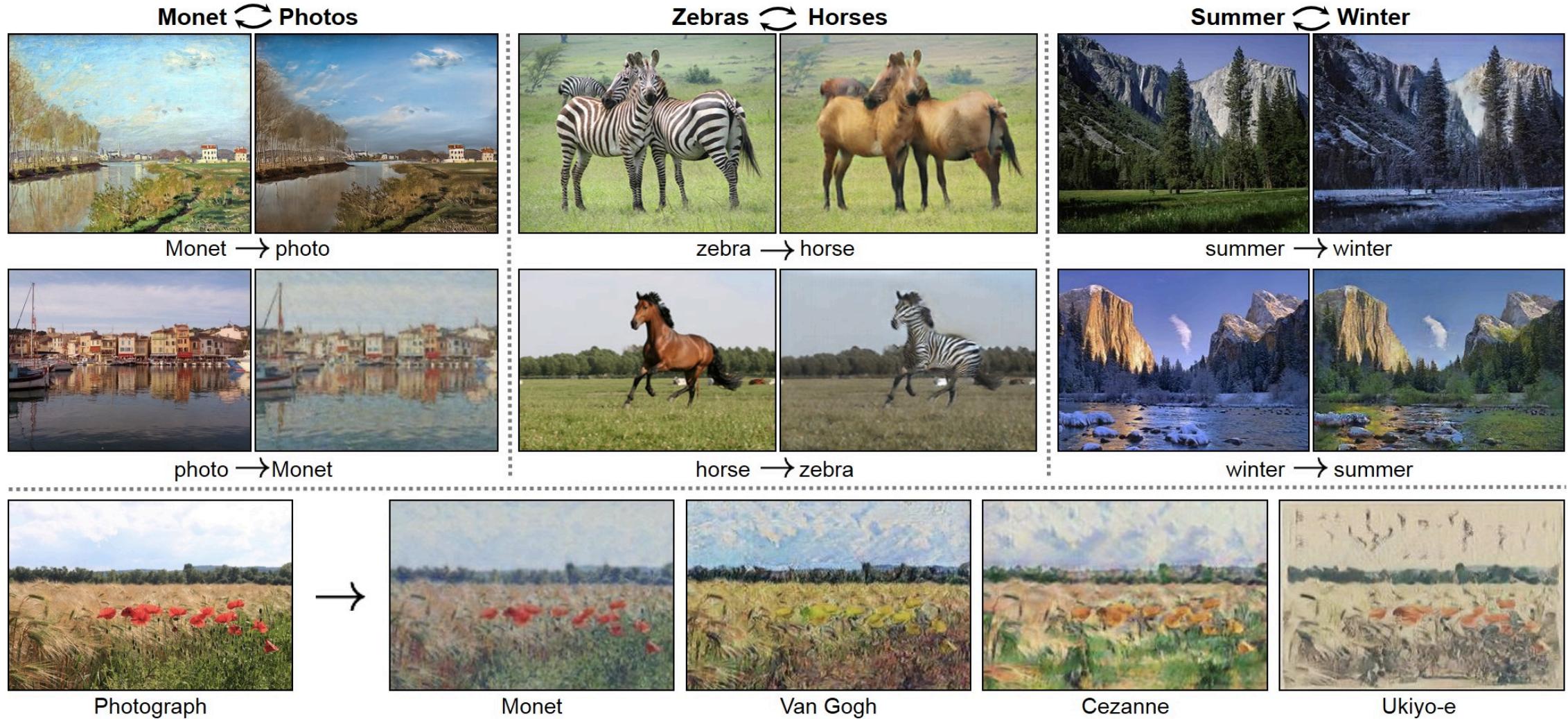
- ▶ Метка класса. Например, «котик» или «собака».
- ▶ Дополнительные признаки. Например: возраст, рост, освещенность.
- ▶ Другая картинка. Например, эскиз рисунка.

Cycle GAN



<https://towardsdatascience.com/image-to-image-translation-using-cyclegan-model-d58cff04755>

Cycle GAN



<https://junyanz.github.io/CycleGAN/>

Bidirectional GAN

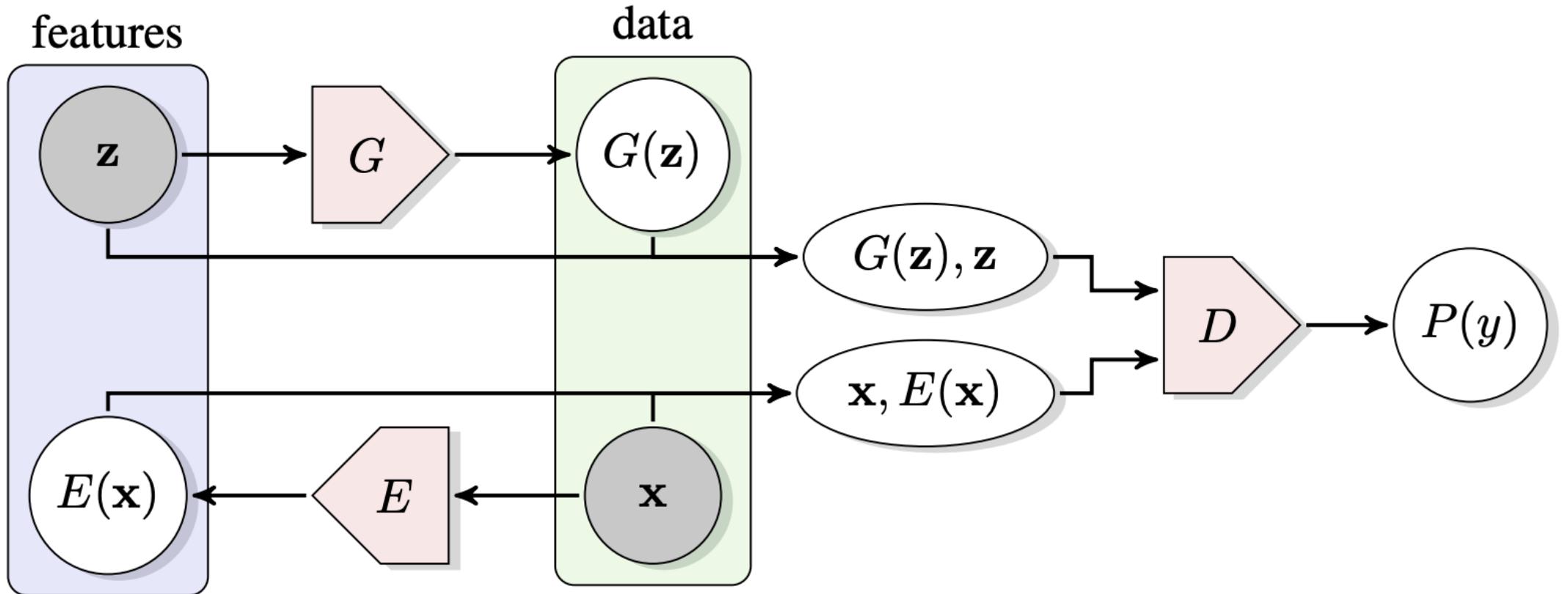


Figure 1: The structure of Bidirectional Generative Adversarial Networks (BiGAN).

<https://arxiv.org/abs/1605.09782>

Заключение



Заключение

- ▶ Генеративно-состязательные модели
 - GAN
 - Wasserstein GAN
 - Conditional GAN
 - Cycle GAN и Bidirectional GAN
- ▶ Приложения
 - Генерация текстов, фото и видео
 - Изменение стиля фотографий
 - Дорисовка фото