

Введение в искусственный интеллект.

Современное компьютерное зрение

Тема: Генеративные состязательные сети

Бабин Д.Н., Иванов И.Е., Петюшко А.А.

кафедра Математической Теории Интеллектуальных Систем



① Зачем нужна генеративная модель



План лекции

- ① Зачем нужна генеративная модель
- ② Дискриминативные и генеративные модели

План лекции

- ① Зачем нужна генеративная модель
- ② Дискриминативные и генеративные модели
- ③ GAN — генеративные состязательные сети

План лекции

- ① Зачем нужна генеративная модель
- ② Дискриминативные и генеративные модели
- ③ GAN — генеративные состязательные сети
- ④ DCGAN



- ① Зачем нужна генеративная модель
- ② Дискриминативные и генеративные модели
- ③ GAN — генеративные состязательные сети
- ④ DCGAN
- ⑤ Функции близости распределений

- ① Зачем нужна генеративная модель
- ② Дискриминативные и генеративные модели
- ③ GAN — генеративные состязательные сети
- ④ DCGAN
- ⑤ Функции близости распределений
- ⑥ Метрика Васерштейна

- ① Зачем нужна генеративная модель
- ② Дискриминативные и генеративные модели
- ③ GAN — генеративные состязательные сети
- ④ DCGAN
- ⑤ Функции близости распределений
- ⑥ Метрика Васерштейна
- ⑦ Условные GAN

Проблема1

Для обучения и надежного тестирования современных СНС нужны огромные объемы данных (и чистая разметка этих данных!). Если измерять в картинках, то это миллионы, в видео — десятки и сотни часов.



Проблема1

Для обучения и надежного тестирования современных СНС нужны огромные объемы данных (и чистая разметка этих данных!). Если измерять в картинках, то это миллионы, в видео — десятки и сотни часов.

- Для каждой новой задачи может потребоваться собирать новую БД (не бывает одной и универсальной), либо сделать разметку новых атрибутов на старых данных — все это стоит на требуемых объемах очень дорого (\$\$\$)



Проблема1

Для обучения и надежного тестирования современных СНС нужны огромные объемы данных (и чистая разметка этих данных!). Если измерять в картинках, то это миллионы, в видео — десятки и сотни часов.

- Для каждой новой задачи может потребоваться собирать новую БД (не бывает одной и универсальной), либо сделать разметку новых атрибутов на старых данных — все это стоит на требуемых объемах очень дорого (\$\$\$)
- Либо можно всего лишь быть компанией, уже имеющей такой объем данных благодаря своей модели бизнеса (Google, Facebook, Twitter и т.п.)



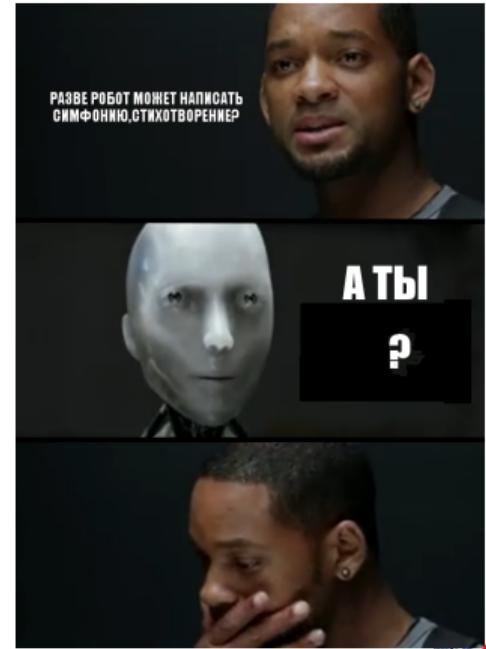
Проблема2

Почему роботы не сочиняют музыку, не пишут стихи, картины?



Проблема2

Почему роботы не сочиняют музыку, не пишут стихи, картины?



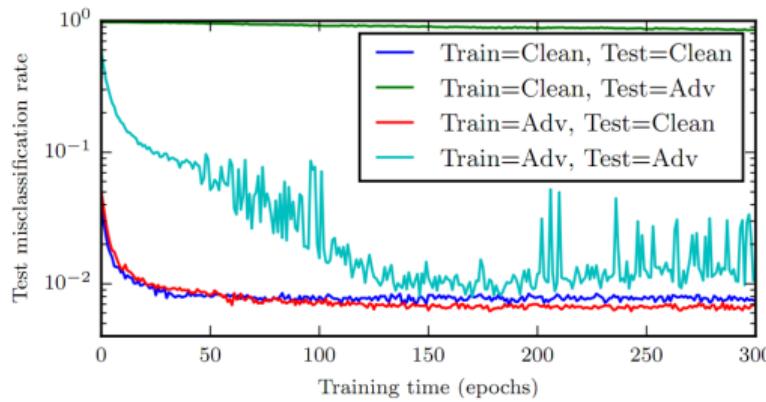
Желательный результат

- После решения **Проблемы1** хотелось бы, чтобы добавление в обучающую базу сгенерированных картинок дало лучший результат на тестовом множестве

Желательный результат

- После решения **Проблемы1** хотелось бы, чтобы добавление в обучающую базу сгенерированных картинок дало лучший результат на тестовом множестве

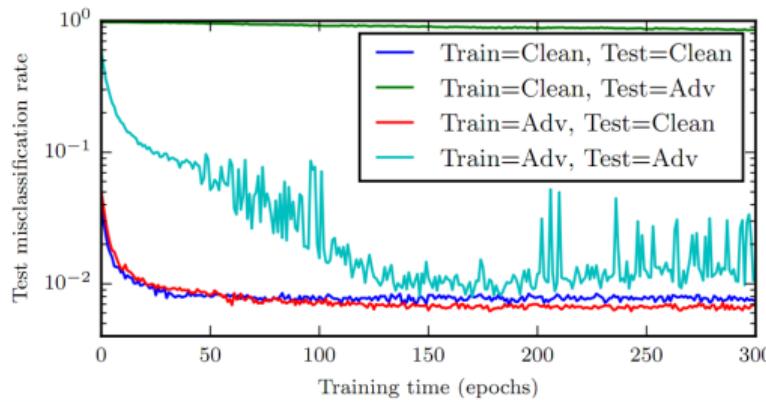
Training on Adversarial Examples



Желательный результат

- После решения **Проблемы1** хотелось бы, чтобы добавление в обучающую базу сгенерированных картинок дало лучший результат на тестовом множестве
- После решения **Проблемы2** хотелось бы, чтобы СНС рисовали картинки с изображениями, неотличимыми от фотографий

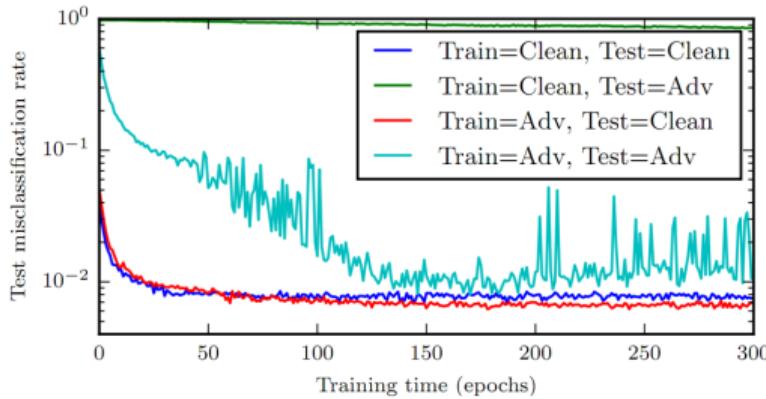
Training on Adversarial Examples



Желательный результат

- После решения **Проблемы1** хотелось бы, чтобы добавление в обучающую базу сгенерированных картинок дало лучший результат на тестовом множестве

Training on Adversarial Examples



- После решения **Проблемы2** хотелось бы, чтобы СНС рисовали картинки с изображениями, неотличимыми от фотографий



Допущение

Предположим, что у нас есть некий набор реальных данных из распределения p_{data} . Возможно, его недостаточно для обучения СНС (например, классификатора), но при этом мы полагаем, что этот набор в какой-то мере отражает то распределение данных, с которым мы хотим работать.



Допущение

Предположим, что у нас есть некий набор реальных данных из распределения p_{data} . Возможно, его недостаточно для обучения СНС (например, классификатора), но при этом мы полагаем, что этот набор в какой-то мере отражает то распределение данных, с которым мы хотим работать.

Предложение

Давайте будем использовать т.н. “генеративные модели” (generative models), которые бы генерировали (создавали) данные из распределения реальных данных p_{data} .



Важные замечания:

- ① Распределение картинок задано на *дискретном* носителе (если предполагать стандартное 8-битное кодирование цвета) **огромной мощности**, поэтому просто сэмплировать из распределения p_{data} не получится, как и делать это с помощью простых аналитических формул

О работе генеративных моделей

Важные замечания:

- ➊ Распределение картинок задано на *дискретном* носителе (если предполагать стандартное 8-битное кодирование цвета) **огромной мощности**, поэтому просто сэмплировать из распределения p_{data} не получится, как и делать это с помощью простых аналитических формул
- ➋ Разнообразие сгенерированных данных будет достигаться тем, что наша генеративная модель будет получать на вход вектор реализаций случайных значений из некоторого простого распределения (например, $z \sim N(0, 1)$)



- ➊ Дискриминативная модель — это модель условной вероятности метки (например, реальные данные или нет) у при наличии данных x (например, сгенерированных $x \sim p_{gen}$ и реальных $x \sim p_{data}$): $P(Y|X = x)$

- ① **Дискриминативная** модель — это модель условной вероятности метки (например, реальные данные или нет) у при наличии данных x (например, сгенерированных $x \sim p_{gen}$ и реальных $x \sim p_{data}$): $P(Y|X = x)$
- ② **Генеративная** модель — это модель совместной вероятности данных и их меток $P(X, Y)$

- ① **Дискриминативная** модель — это модель условной вероятности метки (например, реальные данные или нет) у при наличии данных x (например, сгенерированных $x \sim p_{gen}$ и реальных $x \sim p_{data}$): $P(Y|X = x)$
- ② **Генеративная** модель — это модель совместной вероятности данных и их меток $P(X, Y)$

Замечание. Применим формулу условной вероятности:

$$\arg \max_x P(X, Y = y) = \arg \max_x P(X|Y = y)P(Y = y) = \arg \max_x P(X|Y = y)$$

- ➊ **Дискриминативная** модель — это модель условной вероятности метки (например, реальные данные или нет) у при наличии данных x (например, сгенерированных $x \sim p_{gen}$ и реальных $x \sim p_{data}$): $P(Y|X = x)$
- ➋ **Генеративная** модель — это модель совместной вероятности данных и их меток $P(X, Y)$

Замечание. Применим формулу условной вероятности:

$$\arg \max_x P(X, Y = y) = \arg \max_x P(X|Y = y)P(Y = y) = \arg \max_x P(X|Y = y)$$

Значит, для генеративной модели эквивалентная модель (при условии фиксированных меток) — это $P(X|Y = y)$.

Основные виды генеративных моделей

Основные виды генеративных моделей (мы будем рассматривать последний):



Основные виды генеративных моделей

Основные виды генеративных моделей (мы будем рассматривать последний):

- 1 Авторегрессионная модель (марковское свойство)



Основные виды генеративных моделей

Основные виды генеративных моделей (мы будем рассматривать последний):

- ① Авторегрессионная модель (марковское свойство)
- ② Вариационный автоэнкодер



Основные виды генеративных моделей

Основные виды генеративных моделей (мы будем рассматривать последний):

- ① Авторегрессионная модель (марковское свойство)
- ② Вариационный автоэнкодер
- ③ Генеративные состязательные сети



Основные виды генеративных моделей (мы будем рассматривать последний):

- ① Авторегрессионная модель (марковское свойство)
- ② Вариационный автоэнкодер
- ③ Генеративные состязательные сети

Вариационный автоэнкодер

Сначала реальные данные отображаются в простое распределение (например, $z \sim N(\mu, \sigma^2)$) с помощью энкодера Enc , после чего отображаются обратно с помощью декодера Dec . Dec и Enc — нейросети, обучающиеся совместно. При генерации порождается $z \sim N(\mu, \sigma^2)$ и используется только $Dec(z)$.

Основная идея

- Не проверяется, из какого распределения z (главное, чтобы на тесте и обучении эти распределения совпадали)



Основная идея

- Не проверяется, из какого распределения z (главное, чтобы на teste и обучении эти распределения совпадали)
- Вводится дополнительный модуль **дискриминатора** D , который будет проверять, насколько поданные ему на вход данные похожи на реальные (при этом поданные на вход данные будут либо реальными, либо выходом декодера, который называется в этом подходе **генератором** $G(z)$)



Генеративные состязательные сети

Еще раз, “GAN” (generative adversarial networks) — это генеративная (**Generative**) модель, которая:

Еще раз, “GAN” (generative adversarial networks) — это генеративная (**G**enerative) модель, которая:

- Состоит из двух нейросетей (**n**eural **N**ets), имеющих противоположные цели (**A**dversarial)



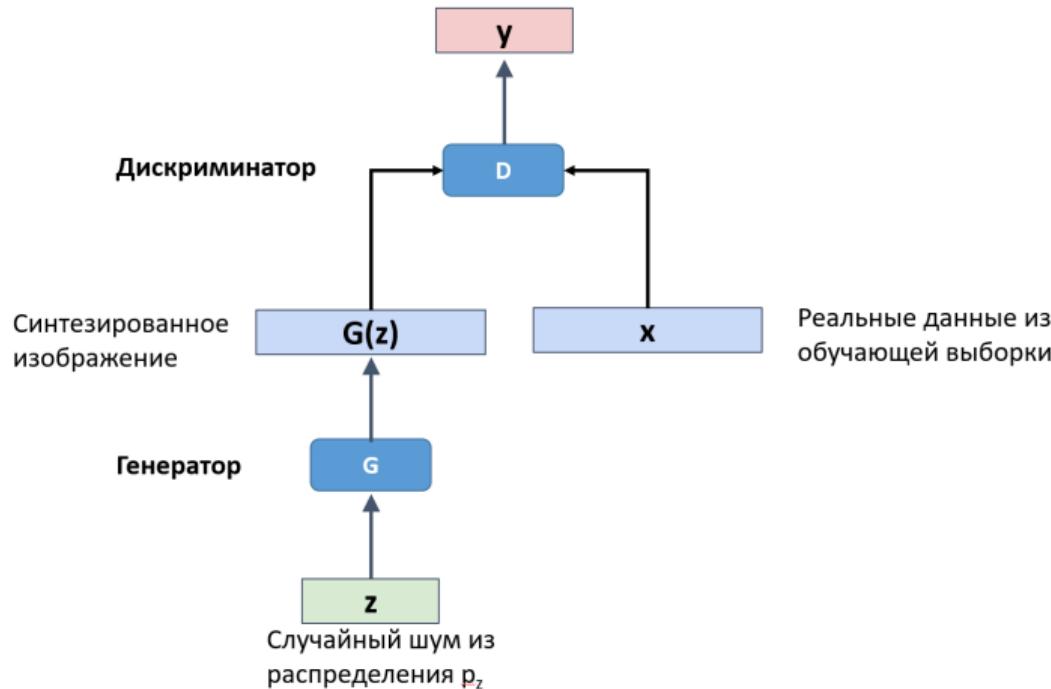
Еще раз, “GAN” (generative adversarial networks) — это генеративная (**Generative**) модель, которая:

- Состоит из двух нейросетей (**neural Nets**), имеющих противоположные цели (**Adversarial**)
- Дискриминатор D пытается определить, является ли его вход “реальным” (похожим на присутствующие в обучающей БД картинки) либо “ложным” (синтезированным)

Еще раз, “GAN” (generative adversarial networks) — это генеративная (**Generative**) модель, которая:

- Состоит из двух нейросетей (**neural Nets**), имеющих противоположные цели (**Adversarial**)
- Дискриминатор D пытается определить, является ли его вход “реальным” (похожим на присутствующие в обучающей БД картинки) либо “ложным” (синтезированным)
- Генератор G пытается синтезировать примеры, максимально близкие по распределению на картинки из обучающей БД

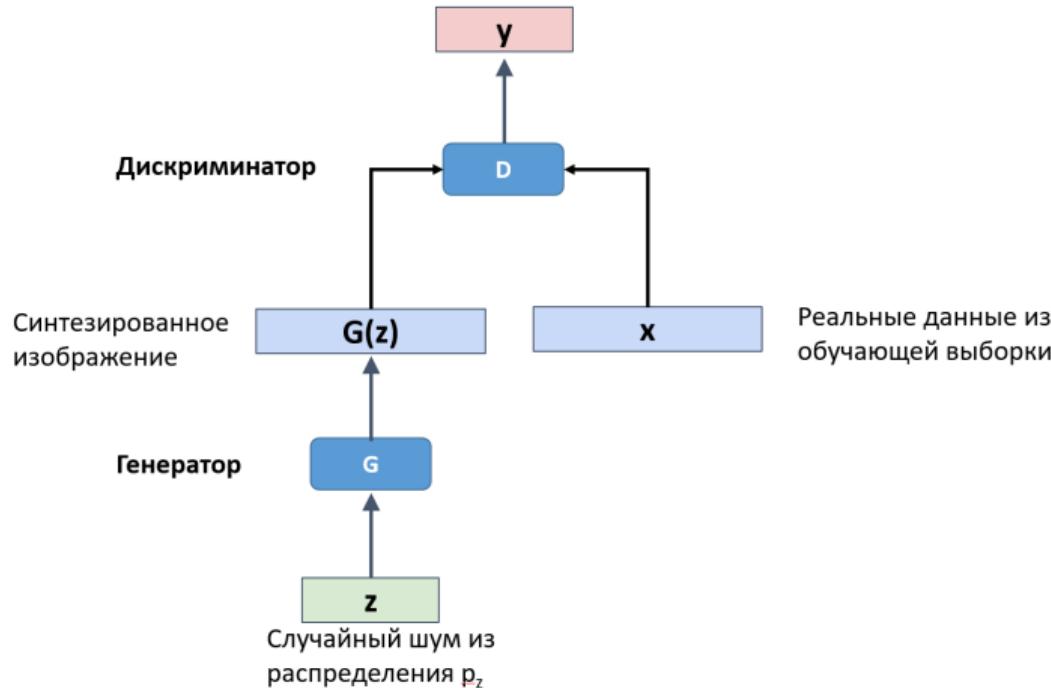
Схема GAN¹



¹I. Goodfellow et al. "Generative Adversarial Networks". 2014

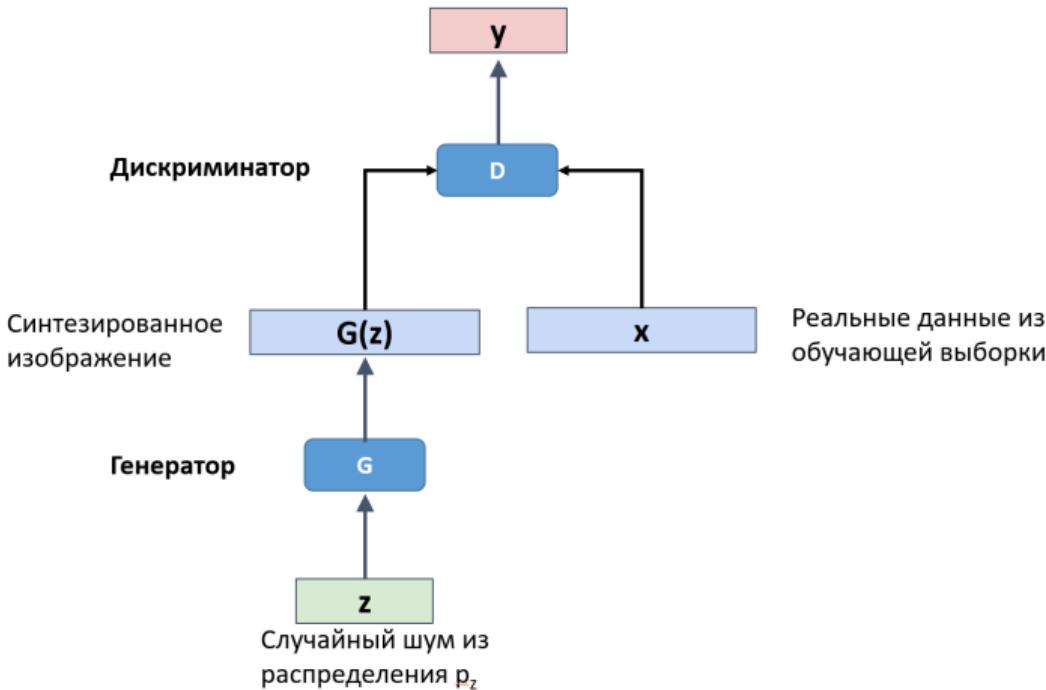
Схема GAN¹

Задача дискриминатора:



¹I. Goodfellow et al. "Generative Adversarial Networks". 2014

Схема GAN¹

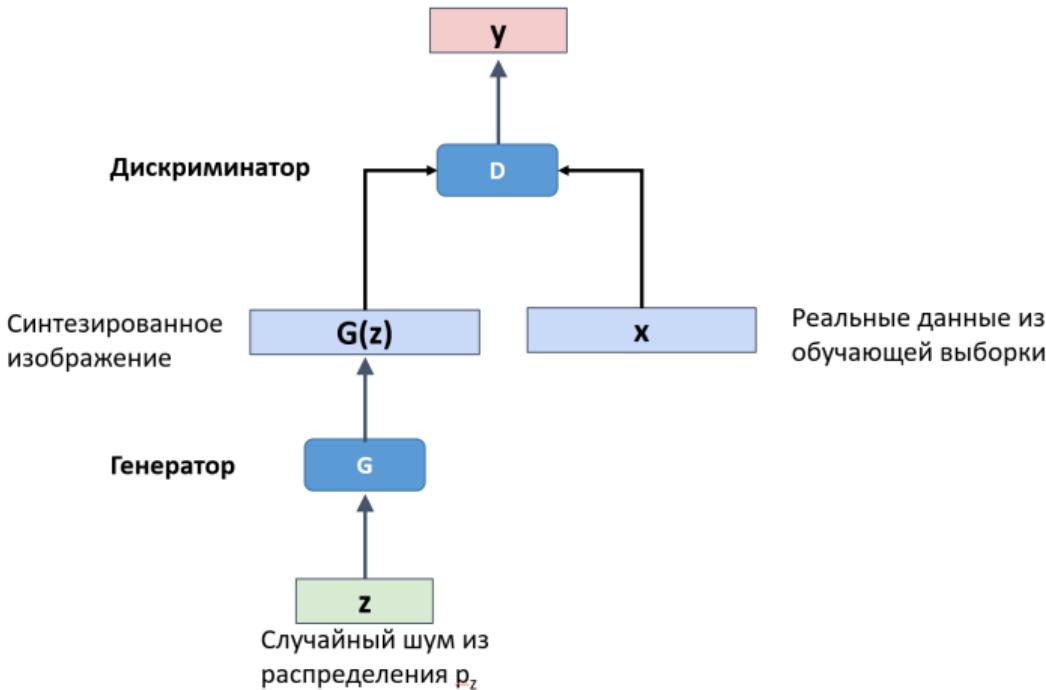


Задача дискриминатора:

- $D(x) \rightarrow 1$

¹I. Goodfellow et al. "Generative Adversarial Networks". 2014

Схема GAN¹

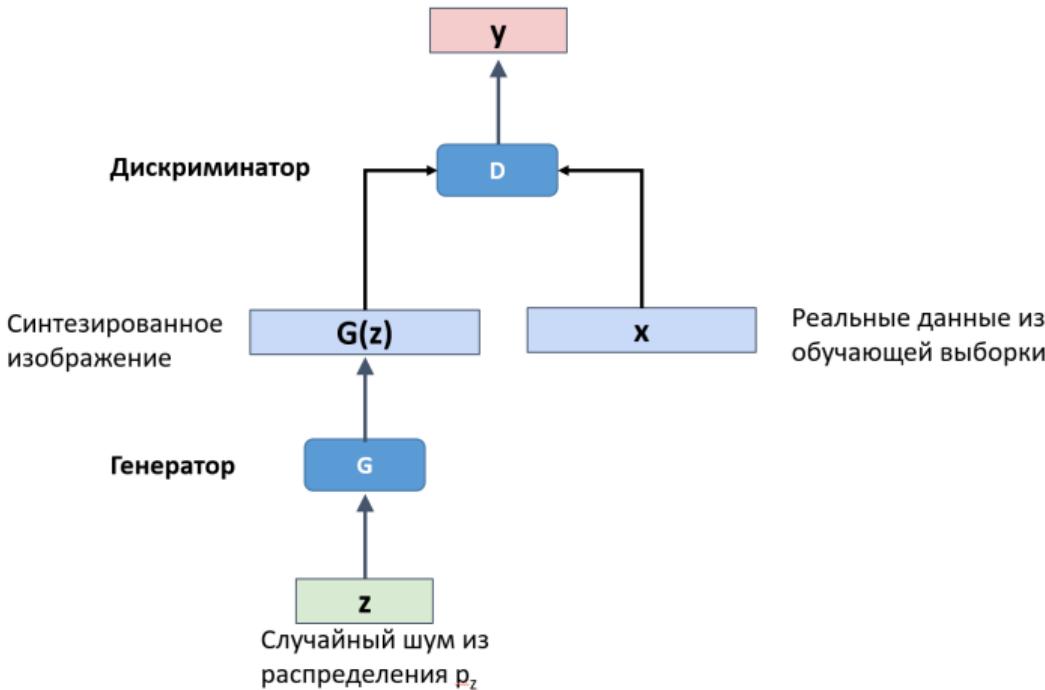


Задача дискриминатора:

- $D(x) \rightarrow 1$
- $D(G(z)) \rightarrow 0$

¹I. Goodfellow et al. "Generative Adversarial Networks". 2014

Схема GAN¹



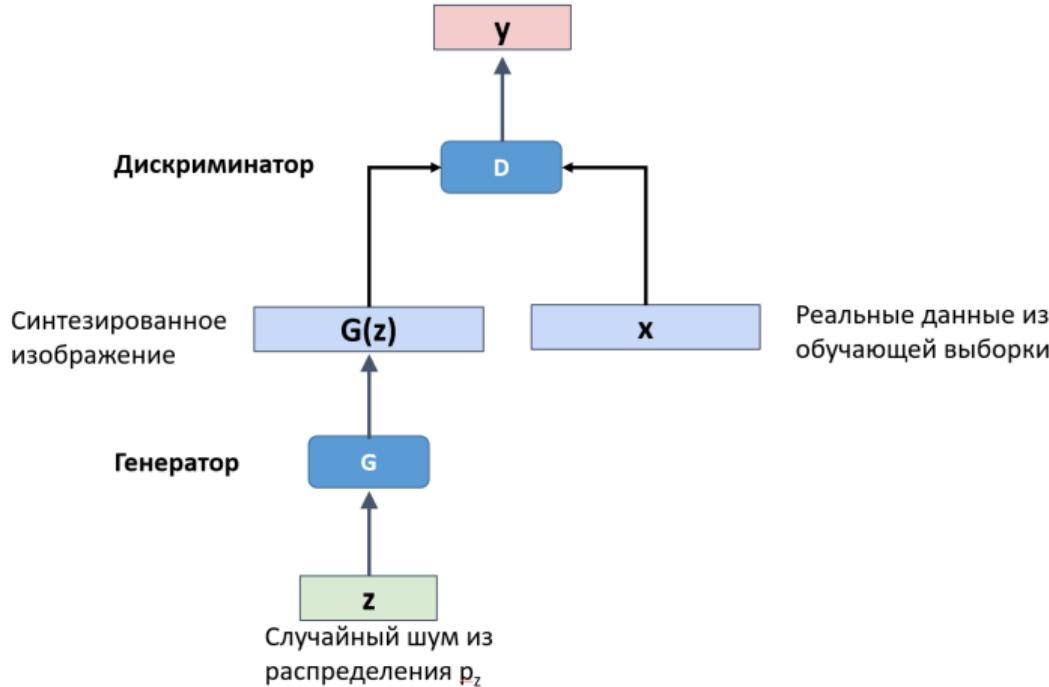
Задача дискриминатора:

- $D(x) \rightarrow 1$
- $D(G(z)) \rightarrow 0$

Задача генератора:

¹I. Goodfellow et al. "Generative Adversarial Networks". 2014

Схема GAN¹



Задача дискриминатора:

- $D(x) \rightarrow 1$
- $D(G(z)) \rightarrow 0$

Задача генератора:

- $D(G(z)) \rightarrow 1$

¹I. Goodfellow et al. "Generative Adversarial Networks". 2014

Дискриминатор в GAN

Дискриминатор D :



Дискриминатор в GAN

Дискриминатор D :

- Обычно за основу берется некоторая известная классификационная сеть (VGG, ResNet)

Дискриминатор D :

- Обычно за основу берется некоторая известная классификационная сеть (VGG, ResNet)
- Выход дискриминатора $y = D(x)$ — число из отрезка $[0, 1]$, которое показывает, насколько дискриминатор “уверен” в том, что вход взят из распределения реальных картинок, т.е.

Дискриминатор D :

- Обычно за основу берется некоторая известная классификационная сеть (VGG, ResNet)
- Выход дискриминатора $y = D(x)$ — число из отрезка $[0, 1]$, которое показывает, насколько дискриминатор “уверен” в том, что вход взят из распределения реальных картинок, т.е.
 - $D(x) = 1$ означает, что дискриминатор на 100% уверен в том, что x — реальное изображение

Дискриминатор D :

- Обычно за основу берется некоторая известная классификационная сеть (VGG, ResNet)
- Выход дискриминатора $y = D(x)$ — число из отрезка $[0, 1]$, которое показывает, насколько дискриминатор “уверен” в том, что вход взят из распределения реальных картинок, т.е.
 - $D(x) = 1$ означает, что дискриминатор на 100% уверен в том, что x — реальное изображение
 - $D(x) = 0$ означает, что дискриминатор на 100% уверен в том, что x — синтезированное изображение

Генератор в GAN

Генератор G :



Генератор G :

- Наиболее нетривиальная и интересная часть



Генератор G :

- Наиболее нетривиальная и интересная часть
- В случае с работой над распределением картинок G — это сверточная сеть:

Генератор G :

- Наиболее нетривиальная и интересная часть
- В случае с работой над распределением картинок G — это сверточная сеть:
 - Вход — это случайный вектор (даже не картинка!) $z \in R^N$ из некоторого простого распределения

Генератор G :

- Наиболее нетривиальная и интересная часть
- В случае с работой над распределением картинок G — это сверточная сеть:
 - Вход — это случайный вектор (даже не картинка!) $z \in R^N$ из некоторого простого распределения
 - Затем с помощью полносвязного слоя и reshape вход преобразуется в 2D-изображение гораздо меньшего размера, чем изображения из реального обучающего множества

Генератор G :

- Наиболее нетривиальная и интересная часть
- В случае с работой над распределением картинок G — это сверточная сеть:
 - Вход — это случайный вектор (даже не картинка!) $z \in R^N$ из некоторого простого распределения
 - Затем с помощью полносвязного слоя и reshape вход преобразуется в 2D-изображение гораздо меньшего размера, чем изображения из реального обучающего множества
 - После чего последовательно применяются сверточные слои, увеличивающие пространственный размер — транспонированные свертки

Генератор G :

- Наиболее нетривиальная и интересная часть
- В случае с работой над распределением картинок G — это сверточная сеть:
 - Вход — это случайный вектор (даже не картинка!) $z \in R^N$ из некоторого простого распределения
 - Затем с помощью полносвязного слоя и reshape вход преобразуется в 2D-изображение гораздо меньшего размера, чем изображения из реального обучающего множества
 - После чего последовательно применяются сверточные слои, увеличивающие пространственный размер — транспонированные свертки
 - Выход генератора $G(z)$ — тензор тех же размерностей, что и реальная картинка x

Функция потерь в GAN

Для обучения GAN используется минимаксная оптимизация для следующей функции:

$$V(D, G) = \log D(x) \cdot (1 - D(G(z)))$$



Функция потерь в GAN

Для обучения GAN используется минимаксная оптимизация для следующей функции:

$$V(D, G) = \log D(x) \cdot (1 - D(G(z)))$$

Почему минимакс?



Функция потерь в GAN

Для обучения GAN используется минимаксная оптимизация для следующей функции:

$$V(D, G) = \log D(x) \cdot (1 - D(G(z)))$$

Почему минимакс?

- Мы хотим, чтобы параметры D были таковы, чтобы на реальных данных выход D был большим, а на синтетических — маленьким $\Rightarrow \max_D V$



Функция потерь в GAN

Для обучения GAN используется минимаксная оптимизация для следующей функции:

$$V(D, G) = \log D(x) \cdot (1 - D(G(z)))$$

Почему минимакс?

- Мы хотим, чтобы параметры D были таковы, чтобы на реальных данных выход D был большим, а на синтетических — маленьким $\Rightarrow \max_D V$
- Мы хотим, чтобы параметры G были таковы, что на синтетических данных выход D был большим $\Rightarrow \min_G V$



Функция потерь в GAN

Для обучения GAN используется минимаксная оптимизация для следующей функции:

$$V(D, G) = \log D(x) \cdot (1 - D(G(z)))$$

Почему минимакс?

- Мы хотим, чтобы параметры D были таковы, чтобы на реальных данных выход D был большим, а на синтетических — маленьким $\Rightarrow \max_D V$
- Мы хотим, чтобы параметры G были таковы, что на синтетических данных выход D был большим $\Rightarrow \min_G V$

Таким образом, приходим к следующей формальной записи:

$$\min_G \max_D V(D, G) = \min_G \max_D [\mathbb{E}_{x \sim p_{data}} \log D(x) + \mathbb{E}_{z \sim p_z} \log(1 - D(G(z)))]$$

Решение задачи минимакса

- Будем использовать поэтапное обучение: на одном шаге обновляем параметры D с помощью метода обратного распространения ошибки, на следующем шаге обновляем параметры G , затем опять D , потом снова G и т.д.

Решение задачи минимакса

- Будем использовать поэтапное обучение: на одном шаге обновляем параметры D с помощью метода обратного распространения ошибки, на следующем шаге обновляем параметры G , затем опять D , потом снова G и т.д.
- Для обновления дискриминатора нужны как реальные данные, так и синтетические. Также зачастую дискриминатор обновляют несколько итераций подряд

Решение задачи минимакса

- Будем использовать поэтапное обучение: на одном шаге обновляем параметры D с помощью метода обратного распространения ошибки, на следующем шаге обновляем параметры G , затем опять D , потом снова G и т.д.
- Для обновления дискриминатора нужны как реальные данные, так и синтетические. Также зачастую дискриминатор обновляют несколько итераций подряд
- Обновление дискриминатора — с помощью **градиентного подъема**



Решение задачи минимакса

- Будем использовать поэтапное обучение: на одном шаге обновляем параметры D с помощью метода обратного распространения ошибки, на следующем шаге обновляем параметры G , затем опять D , потом снова G и т.д.
- Для обновления дискриминатора нужны как реальные данные, так и синтетические. Также зачастую дискриминатор обновляют несколько итераций подряд
- Обновление дискриминатора — с помощью **градиентного подъема**
- Для обновления генератора нужны только синтетические данные, и функция потерь на этом шаге проще: $\min_G \mathbb{E}_{z \sim p_z} \log(1 - D(G(z)))$

Решение задачи минимакса

- Будем использовать поэтапное обучение: на одном шаге обновляем параметры D с помощью метода обратного распространения ошибки, на следующем шаге обновляем параметры G , затем опять D , потом снова G и т.д.
- Для обновления дискриминатора нужны как реальные данные, так и синтетические. Также зачастую дискриминатор обновляют несколько итераций подряд
- Обновление дискриминатора — с помощью **градиентного подъема**
- Для обновления генератора нужны только синтетические данные, и функция потерь на этом шаге проще: $\min_G \mathbb{E}_{z \sim p_z} \log(1 - D(G(z)))$
- Обновление генератора — с помощью **градиентного спуска**

Оригинальный алгоритм обучения GAN

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

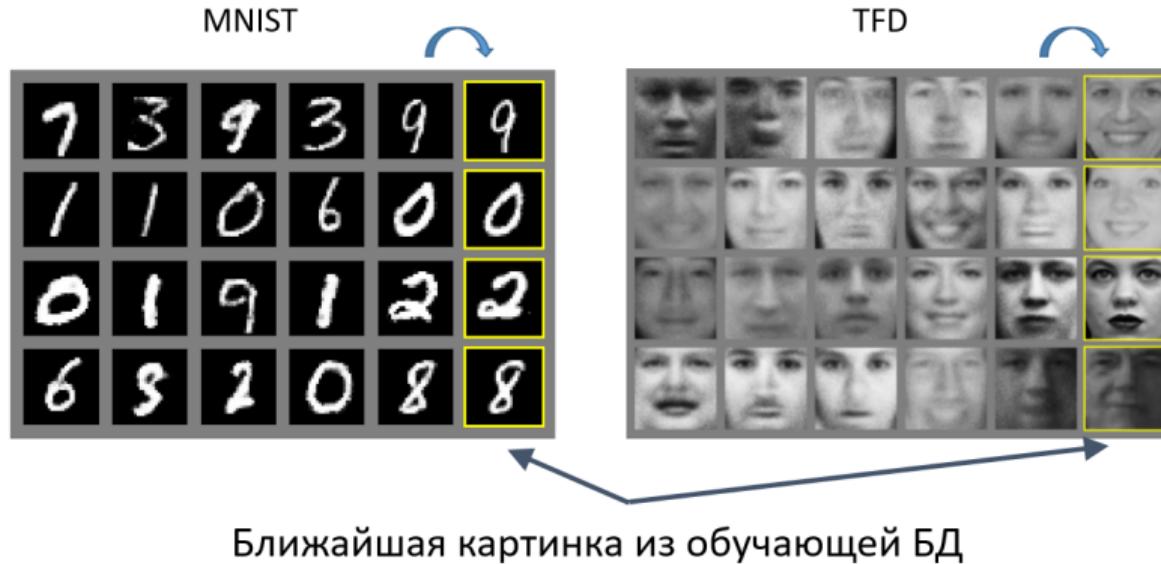
- Sample minibatch of m noise samples $\{\mathbf{z}^{(1)}, \dots, \mathbf{z}^{(m)}\}$ from noise prior $p_g(\mathbf{z})$.
- Update the generator by descending its stochastic gradient:

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

end for

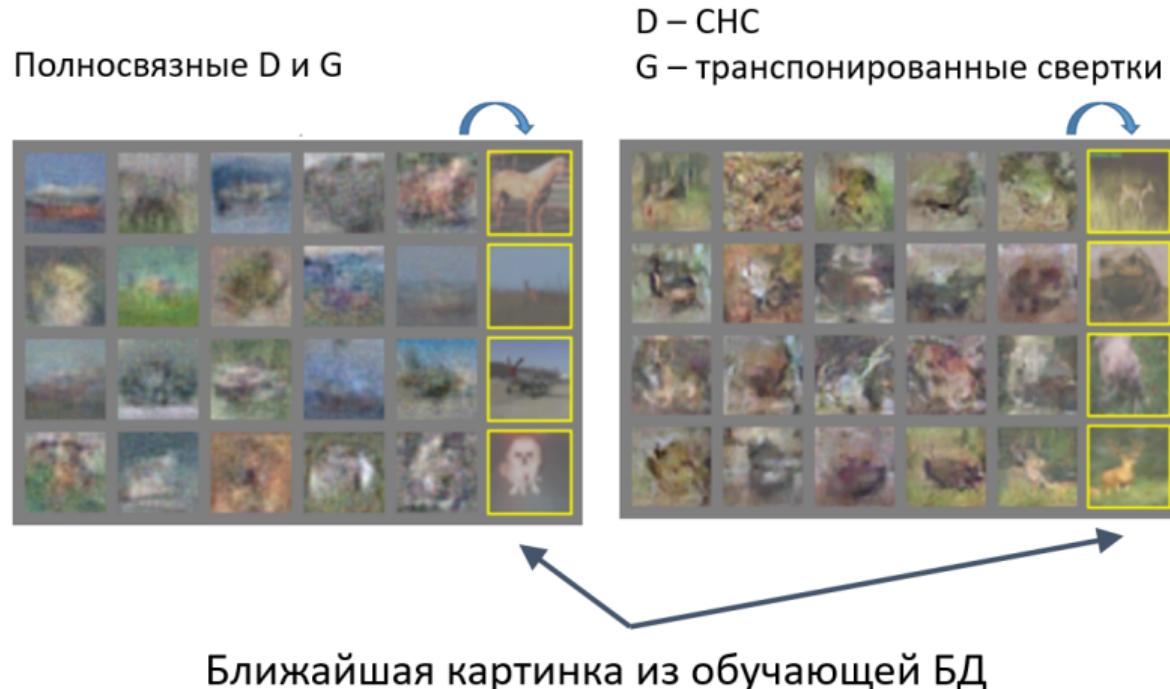
The gradient-based updates can use any standard gradient-based learning rule. We used momentum in our experiments.

Оригинальный GAN — результаты на MNIST и лицах²



²Toronto Face Dataset

Оригинальный GAN — результаты на CIFAR-10



Замечания по оригинальному GAN

- Очень необычный минимаксный алгоритм



Замечания по оригинальному GAN

- Очень необычный минимаксный алгоритм
- Очень нестабильный и сложный в обучении

Замечания по оригинальному GAN

- Очень необычный минимаксный алгоритм
- Очень нестабильный и сложный в обучении
- Эксперименты показали, что GAN не просто запоминает картинки из обучающего множества

Замечания по оригинальному GAN

- Очень необычный минимаксный алгоритм
- Очень нестабильный и сложный в обучении
- Эксперименты показали, что GAN не просто запоминает картинки из обучающего множества
- Эксперименты показали, что для генерации картинок нужно использовать транспонированные свертки, а не полносвязные слои

DCGAN⁴

DCGAN (Deep Convolutional GAN) — одна из первых успешных реализаций³ GAN, являющаяся основой для множества проектов:

³<https://github.com/carpedm20/DCGAN-tensorflow>

⁴A. Radford et al. “Unsupervised Representation Learning with Deep Convolutional Generative Adversarial Networks”. 2015

DCGAN⁴

DCGAN (Deep Convolutional GAN) — одна из первых успешных реализаций³ GAN, являющаяся основой для множества проектов:

- Дискриминатор — сверточная сеть со свертками с шагом $s > 1$ (нет субдискретизации)

³<https://github.com/carpedm20/DCGAN-tensorflow>

⁴A. Radford et al. “Unsupervised Representation Learning with Deep Convolutional Generative Adversarial Networks”. 2015

DCGAN (Deep Convolutional GAN) — одна из первых успешных реализаций³ GAN, являющаяся основой для множества проектов:

- Дискриминатор — сверточная сеть со свертками с шагом $s > 1$ (нет субдискретизации)
- Генератор — сверточная сеть с транспонированными свертками

³<https://github.com/carpedm20/DCGAN-tensorflow>

⁴A. Radford et al. “Unsupervised Representation Learning with Deep Convolutional Generative Adversarial Networks”. 2015

DCGAN (Deep Convolutional GAN) — одна из первых успешных реализаций³ GAN, являющаяся основой для множества проектов:

- Дискриминатор — сверточная сеть со свертками с шагом $s > 1$ (нет субдискретизации)
- Генератор — сверточная сеть с транспонированными свертками
- Пакетная нормализация как в D , так и в G

³<https://github.com/carpedm20/DCGAN-tensorflow>

⁴A. Radford et al. “Unsupervised Representation Learning with Deep Convolutional Generative Adversarial Networks”. 2015

DCGAN (Deep Convolutional GAN) — одна из первых успешных реализаций³ GAN, являющаяся основой для множества проектов:

- Дискриминатор — сверточная сеть со свертками с шагом $s > 1$ (нет субдискретизации)
- Генератор — сверточная сеть с транспонированными свертками
- Пакетная нормализация как в D , так и в G
- В генераторе используется активация $ReLU$, а в дискриминаторе — $Leaky ReLU$

³<https://github.com/carpedm20/DCGAN-tensorflow>

⁴A. Radford et al. “Unsupervised Representation Learning with Deep Convolutional Generative Adversarial Networks”. 2015

DCGAN (Deep Convolutional GAN) — одна из первых успешных реализаций³ GAN, являющаяся основой для множества проектов:

- Дискриминатор — сверточная сеть со свертками с шагом $s > 1$ (нет субдискретизации)
- Генератор — сверточная сеть с транспонированными свертками
- Пакетная нормализация как в D , так и в G
- В генераторе используется активация $ReLU$, а в дискриминаторе — $Leaky ReLU$

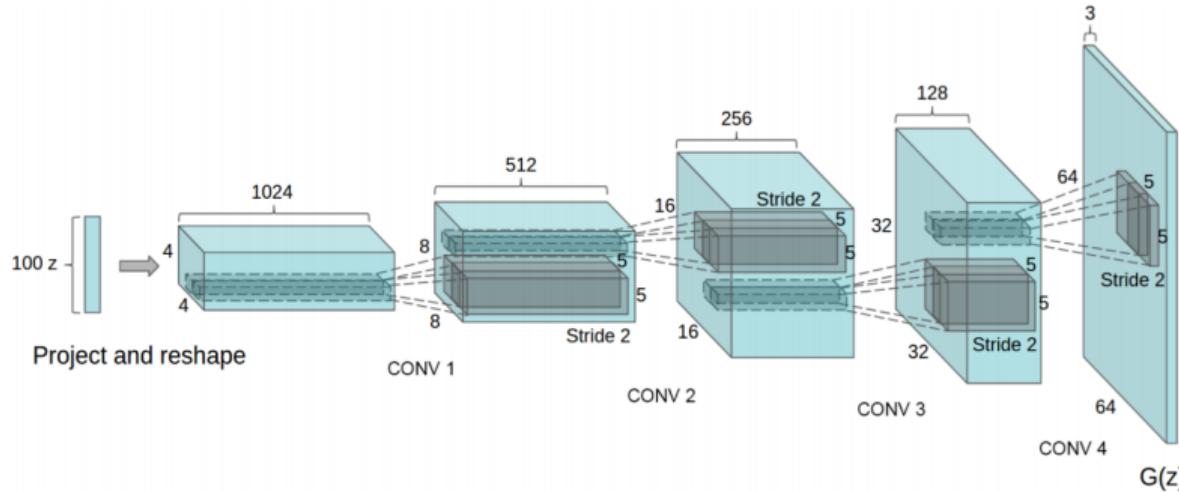
 carpedm20 / DCGAN-tensorflow

 Watch ▾ 245  Star 6.3k  Fork 2.5k

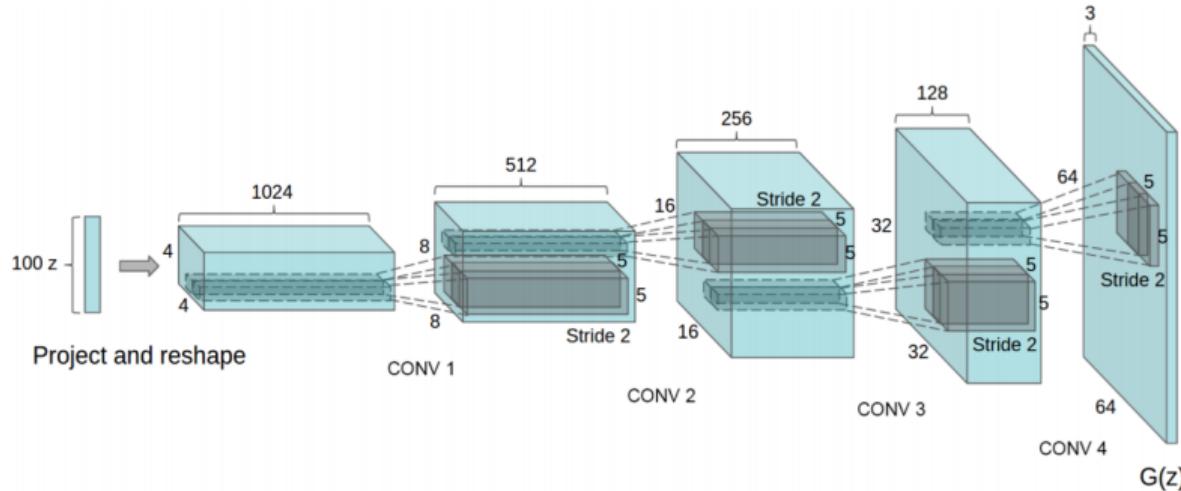
³<https://github.com/carpedm20/DCGAN-tensorflow>

⁴A. Radford et al. “Unsupervised Representation Learning with Deep Convolutional Generative Adversarial Networks”. 2015

Генератор в DCGAN

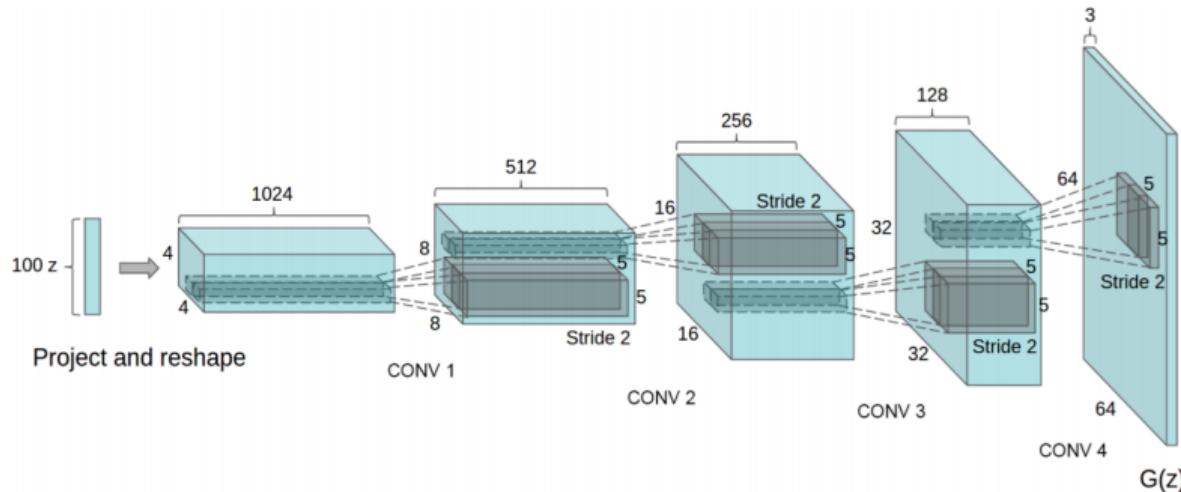


Генератор в DCGAN



- Вход — 100-мерный вектор шума $z \sim N(0, 1)$ (или $U[0, 1]$), выход — цветная картинка 64×64

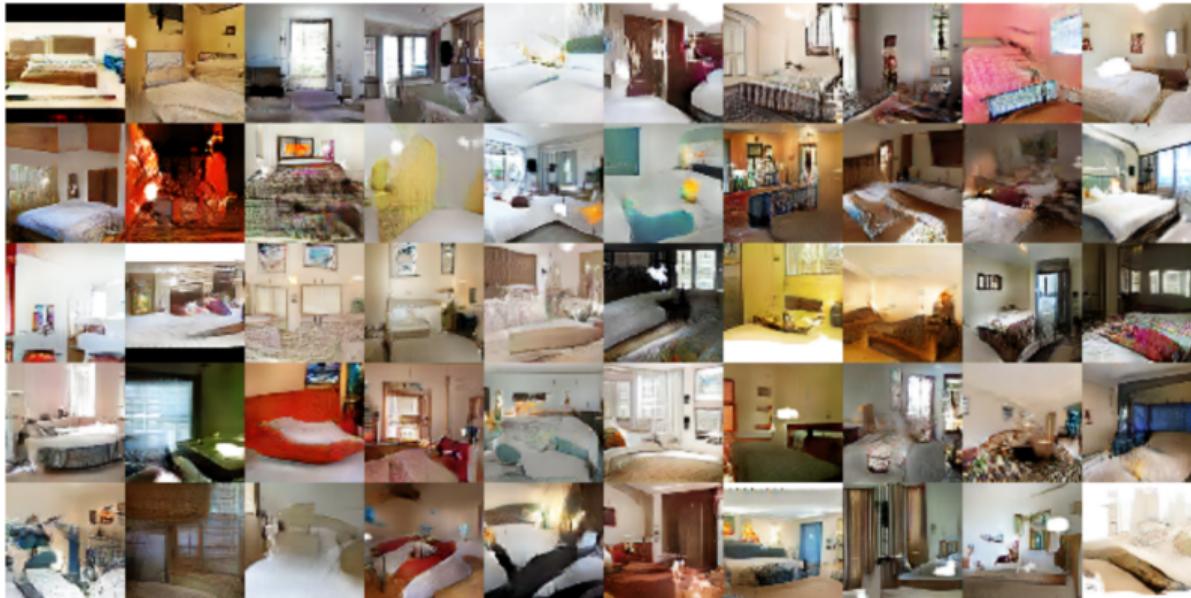
Генератор в DCGAN



- Вход — 100-мерный вектор шума $z \sim N(0, 1)$ (или $U[0, 1]$), выход — цветная картинка 64×64
- Правило, разновидность которого затем будут часто использовать во многих подходах (не только в GAN): при увеличении пространственной размерности в k раз во столько же уменьшают количество карт признаков

Результаты на DCGAN

Результаты обучения на LSUN⁵



⁵<https://www.yf.io/p/lsun>

Интерполяция на скрытом (латентном) представлении

Линейная интерполяция (слева направо сверху вниз)



$$t \in [0, 1]$$



Арифметика на скрытом (латентном) представлении



О функции близости распределений

Для сравнения распределений зачастую применяют т.н. функцию близости распределений, которая не должна являться в общем случае метрикой (функцией расстояния).

О функции близости распределений

Для сравнения распределений зачастую применяют т.н. функцию близости распределений, которая не должна являться в общем случае метрикой (функцией расстояния).

Определение

Функция близости (divergence) двух распределений, заданных на одном пространстве функций распределения S — это функция $D(\cdot||\cdot) : S \times S \rightarrow \mathbb{R}$, такая что:

- ① $D(P||Q) \geq 0$ для всех $P, Q \in S$
- ② $D(P||Q) = 0 \Leftrightarrow P = Q$



О функции близости распределений

Для сравнения распределений зачастую применяют т.н. функцию близости распределений, которая не должна являться в общем случае метрикой (функцией расстояния).

Определение

Функция близости (divergence) двух распределений, заданных на одном пространстве функций распределения S — это функция $D(\cdot||\cdot) : S \times S \rightarrow \mathbb{R}$, такая что:

- ① $D(P||Q) \geq 0$ для всех $P, Q \in S$
- ② $D(P||Q) = 0 \Leftrightarrow P = Q$

Замечание. Как видно, по сравнению с метрикой опущены условия симметричности и неравенства треугольника.

Функция близости Кульбака-Лейблера

Функция близости Кульбака-Лейблера (Kullback-Leibler divergence) — самая применяемая функция близости в машинном обучении.

Функция близости Кульбака-Лейблера⁶

$$D_{KL}(P||Q) = \sum_x p(x) \log \frac{p(x)}{q(x)}$$

⁶https://en.wikipedia.org/wiki/Kullback-Leibler_divergence

Функция близости Кульбака-Лейблера

Функция близости Кульбака-Лейблера (Kullback-Leibler divergence) — самая применяемая функция близости в машинном обучении.

Функция близости Кульбака-Лейблера⁶

$$D_{KL}(P||Q) = \sum_x p(x) \log \frac{p(x)}{q(x)}$$

Замечание. $D_{KL}(P||Q) = -\sum_x p(x) \log q(x) + \sum_x p(x) \log p(x) = H(P, Q) - H(P)$, где $H(P, Q)$ — кросс-энтропия распределений P и Q , $H(P)$ — энтропия P .

⁶https://en.wikipedia.org/wiki/Kullback-Leibler_divergence

Функция близости Кульбака-Лейблера

Функция близости Кульбака-Лейблера (Kullback-Leibler divergence) — самая применяемая функция близости в машинном обучении.

Функция близости Кульбака-Лейблера⁶

$$D_{KL}(P||Q) = \sum_x p(x) \log \frac{p(x)}{q(x)}$$

Замечание. $D_{KL}(P||Q) = -\sum_x p(x) \log q(x) + \sum_x p(x) \log p(x) = H(P, Q) - H(P)$, где $H(P, Q)$ — кросс-энтропия распределений P и Q , $H(P)$ — энтропия P .

Упражнение. Доказать, что D_{KL} — это действительно функция близости распределений (указание: применить **неравенство Гиббса**).

⁶https://en.wikipedia.org/wiki/Kullback-Leibler_divergence

Функция близости Кульбака-Лейблера

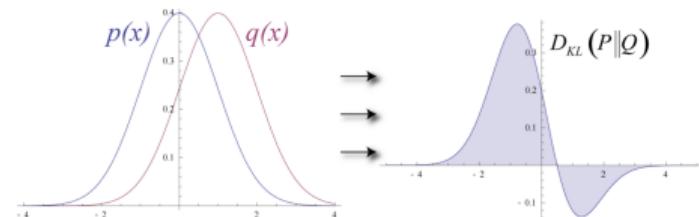
Функция близости Кульбака-Лейблера (Kullback-Leibler divergence) — самая применяемая функция близости в машинном обучении.

Функция близости Кульбака-Лейблера⁶

$$D_{KL}(P||Q) = \sum_x p(x) \log \frac{p(x)}{q(x)}$$

Замечание. $D_{KL}(P||Q) = -\sum_x p(x) \log q(x) + \sum_x p(x) \log p(x) = H(P, Q) - H(P)$, где $H(P, Q)$ — кросс-энтропия распределений P и Q , $H(P)$ — энтропия P .

Упражнение. Доказать, что D_{KL} — это действительно функция близости распределений (указание: применить **неравенство Гиббса**).



⁶https://en.wikipedia.org/wiki/Kullback-Leibler_divergence

Почему для классификации используется кросс-энтропия

- Пусть p — это унитарное кодирование правильного класса, а q — это выход SoftMax-слоя нейросети

Почему для классификации используется кросс-энтропия

- Пусть p — это унитарное кодирование правильного класса, а q — это выход SoftMax-слоя нейросети
- Кросс-энтропия

$$H(p, q) = - \sum_x p(x) \log q(x) = H(p) + D_{KL}(p||q)$$

Почему для классификации используется кросс-энтропия

- Пусть p — это унитарное кодирование правильного класса, а q — это выход SoftMax-слоя нейросети
- Кросс-энтропия

$$H(p, q) = - \sum_x p(x) \log q(x) = H(p) + D_{KL}(p||q)$$

- Для унитарного кодирования правильного класса энтропия всегда равна нулю (доказать!)

Почему для классификации используется кросс-энтропия

- Пусть p — это унитарное кодирование правильного класса, а q — это выход SoftMax-слоя нейросети
- Кросс-энтропия

$$H(p, q) = - \sum_x p(x) \log q(x) = H(p) + D_{KL}(p||q)$$

- Для унитарного кодирования правильного класса энтропия всегда равна нулю (доказать!)
- Значит, в нашем случае $H(p, q) = D_{KL}(p||q) \geq 0$, при этом $H(p, q) = 0 \Leftrightarrow p = q$

Почему для классификации используется кросс-энтропия

- Пусть p — это унитарное кодирование правильного класса, а q — это выход SoftMax-слоя нейросети
- Кросс-энтропия

$$H(p, q) = - \sum_x p(x) \log q(x) = H(p) + D_{KL}(p||q)$$

- Для унитарного кодирования правильного класса энтропия всегда равна нулю (доказать!)
- Значит, в нашем случае $H(p, q) = D_{KL}(p||q) \geq 0$, при этом $H(p, q) = 0 \Leftrightarrow p = q$
- Именно поэтому мы минимизируем кросс-энтропию

Функция близости Йенсена-Шеннона и GAN

У функции близости Кульбака-Лейблера есть симметричная версия — функция близости Йенсена-Шеннона (Jensen-Shannon divergence).

Функция близости Йенсена-Шеннона

$$D_{JS}(P||Q) = \frac{1}{2}D_{KL}(P||M) + \frac{1}{2}D_{KL}(Q||M), \text{ где } M = \frac{1}{2}(P + Q).$$



Функция близости Йенсена-Шеннона и GAN

У функции близости Кульбака-Лейблера есть симметричная версия — функция близости Йенсена-Шеннона (Jensen-Shannon divergence).

Функция близости Йенсена-Шеннона

$$D_{JS}(P||Q) = \frac{1}{2}D_{KL}(P||M) + \frac{1}{2}D_{KL}(Q||M), \text{ где } M = \frac{1}{2}(P + Q).$$

Упражнение 1. Показать, что $\sqrt{D_{JS}}$ — метрика.

Функция близости Йенсена-Шеннона и GAN

У функции близости Кульбака-Лейблера есть симметричная версия — функция близости Йенсена-Шеннона (Jensen-Shannon divergence).

Функция близости Йенсена-Шеннона

$$D_{JS}(P||Q) = \frac{1}{2}D_{KL}(P||M) + \frac{1}{2}D_{KL}(Q||M), \text{ где } M = \frac{1}{2}(P + Q).$$

Упражнение 1. Показать, что $\sqrt{D_{JS}}$ — метрика.

Вспомним оригинальную функцию потерь:

$$\min_G \max_D V(D, G) = \min_G \max_D [\mathbb{E}_{x \sim p_{data}} \log D(x) + \mathbb{E}_{z \sim p_z} \log(1 - D(G(z)))]$$



Функция близости Йенсена-Шеннона и GAN

У функции близости Кульбака-Лейблера есть симметричная версия — функция близости Йенсена-Шеннона (Jensen-Shannon divergence).

Функция близости Йенсена-Шеннона

$$D_{JS}(P||Q) = \frac{1}{2}D_{KL}(P||M) + \frac{1}{2}D_{KL}(Q||M), \text{ где } M = \frac{1}{2}(P + Q).$$

Упражнение 1. Показать, что $\sqrt{D_{JS}}$ — метрика.

Вспомним оригинальную функцию потерь:

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

Если обозначить $C(G) = \max_D V(D, G)$, p_g — распределение $G(z)$, то $C(G) = -\log 4 + 2D_{JS}(p_{\text{data}}||p_g)$ для оптимального дискриминатора $D_G = \frac{p_{\text{data}}}{p_{\text{data}}+p_g}$, и при этом минимум $C(G)$ будет достигаться при $p_{\text{data}} = p_g$.

Функция близости Йенсена-Шеннона и GAN

У функции близости Кульбака-Лейблера есть симметричная версия — функция близости Йенсена-Шеннона (Jensen-Shannon divergence).

Функция близости Йенсена-Шеннона

$$D_{JS}(P||Q) = \frac{1}{2}D_{KL}(P||M) + \frac{1}{2}D_{KL}(Q||M), \text{ где } M = \frac{1}{2}(P + Q).$$

Упражнение1. Показать, что $\sqrt{D_{JS}}$ — метрика.

Вспомним оригинальную функцию потерь:

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

Если обозначить $C(G) = \max_D V(D, G)$, p_g — распределение $G(z)$, то $C(G) = -\log 4 + 2D_{JS}(p_{\text{data}}||p_g)$ для оптимального дискриминатора $D_G = \frac{p_{\text{data}}}{p_{\text{data}}+p_g}$, и при этом минимум $C(G)$ будет достигаться при $p_{\text{data}} = p_g$.

Упражнение2. Доказать.

О метрике Васерштейна

- Несмотря на некоторые теоретические обоснования процедуры оптимизации оригинальных GAN, они отличались крайне нестабильной процедурой обучения

О метрике Васерштейна

- Несмотря на некоторые теоретические обоснования процедуры оптимизации оригинальных GAN, они отличались крайне нестабильной процедурой обучения
- Предложение — использовать т.н. Васерштейн-1 **метрику** (также известную как Earth Mover Distance) для работы с распределениями на разных носителях:

$$W_1(P, Q) = \inf_{\gamma \in \Pi(P, Q)} \mathbb{E}_{(x, y) \sim \gamma} [| | x - y | |_1]$$

где $\Pi(P, Q)$ — множество всех таких совместных распределений $\gamma(x, y)$, что их частные распределения — это P и Q соответственно

О метрике Васерштейна

- Несмотря на некоторые теоретические обоснования процедуры оптимизации оригинальных GAN, они отличались крайне нестабильной процедурой обучения
- Предложение — использовать т.н. Васерштейн-1 **метрику** (также известную как Earth Mover Distance) для работы с распределениями на разных носителях:

$$W_1(P, Q) = \inf_{\gamma \in \Pi(P, Q)} \mathbb{E}_{(x, y) \sim \gamma} [| |x - y| |_1]$$

где $\Pi(P, Q)$ — множество всех таких совместных распределений $\gamma(x, y)$, что их частные распределения — это P и Q соответственно

- Однако в таком определении $W_1(P, Q)$ неразрешима. Но недавно была доказана⁷ двойственная теорема Канторовича-Рубинштейна:

$$W_1(P, Q) = \sup_{\|f\|_L \leq 1} (\mathbb{E}_{x \sim P}[f(x)] - \mathbb{E}_{x \sim Q}[f(x)])$$

⁷Cedric Villani. Optimal Transport: Old and New. 2009

О липшицевых функциях

- Условие $\|f\|_L \leq 1$ означает, что f — это 1-Липшицева функция

О липшицевых функциях

- Условие $\|f\|_L \leq 1$ означает, что f — это 1-Липшицева функция
- **1-Липшицева функция**, или т.н. **короткое отображение** (metric map) — это непрерывная функция отображения между метрическими пространствами, которая не увеличивает расстояния

- Условие $\|f\|_L \leq 1$ означает, что f — это 1-Липшицева функция
- **1-Липшицева функция**, или т.н. **короткое отображение** (metric map) — это непрерывная функция отображения между метрическими пространствами, которая не увеличивает расстояния

Короткое отображение

Если X, Y — метрические пространства, $f : X \rightarrow Y$ — функция отображения, $d_X(\cdot, \cdot)$, $d_Y(\cdot, \cdot)$ — функции расстояния на X и Y соответственно, то f — **короткое отображение**, если для любых точек $a, b \in X$ выполняется $d_Y(f(a), f(b)) \leq d_X(a, b)$.



Метрика Васерштейна и GAN

$$W_1(P, Q) = \sup_{\|f\|_L \leq 1} (\mathbb{E}_{x \sim P}[f(x)] - \mathbb{E}_{x \sim Q}[f(x)])$$

⁸M. Arjovsky et al. "Wasserstein GAN". 2017



Метрика Васерштейна и GAN

$$W_1(P, Q) = \sup_{\|f\|_L \leq 1} (\mathbb{E}_{x \sim P}[f(x)] - \mathbb{E}_{x \sim Q}[f(x)])$$

- Давайте внедрим метрику Васерштейна в оригинальные GAN⁸:

⁸M. Arjovsky et al. “Wasserstein GAN”. 2017

Метрика Васерштейна и GAN

$$W_1(P, Q) = \sup_{\|f\|_L \leq 1} (\mathbb{E}_{x \sim P}[f(x)] - \mathbb{E}_{x \sim Q}[f(x)])$$

- Давайте внедрим метрику Васерштейна в оригинальные GAN⁸:
 - В качестве функции f возьмем дискриминатор D

⁸M. Arjovsky et al. “Wasserstein GAN”. 2017



Метрика Васерштейна и GAN

$$W_1(P, Q) = \sup_{\|f\|_L \leq 1} (\mathbb{E}_{x \sim P}[f(x)] - \mathbb{E}_{x \sim Q}[f(x)])$$

- Давайте внедрим метрику Васерштейна в оригинальные GAN⁸:
 - В качестве функции f возьмем дискриминатор D
 - В качестве распределения P — распределение реальных данных p_{data}

⁸M. Arjovsky et al. “Wasserstein GAN”. 2017



Метрика Васерштейна и GAN

$$W_1(P, Q) = \sup_{\|f\|_L \leq 1} (\mathbb{E}_{x \sim P}[f(x)] - \mathbb{E}_{x \sim Q}[f(x)])$$

- Давайте внедрим метрику Васерштейна в оригинальные GAN⁸:
 - В качестве функции f возьмем дискриминатор D
 - В качестве распределения P — распределение реальных данных p_{data}
 - В качестве распределения Q — распределение сгенерированных данных p_g (распределение $G(z)$)

⁸M. Arjovsky et al. “Wasserstein GAN”. 2017



Метрика Васерштейна и GAN

$$W_1(P, Q) = \sup_{\|f\|_L \leq 1} (\mathbb{E}_{x \sim P}[f(x)] - \mathbb{E}_{x \sim Q}[f(x)])$$

- Давайте внедрим метрику Васерштейна в оригинальные GAN⁸:
 - В качестве функции f возьмем дискриминатор D
 - В качестве распределения P — распределение реальных данных p_{data}
 - В качестве распределения Q — распределение сгенерированных данных p_g (распределение $G(z)$)
- Тогда функционал примет вид:

$$\min_G W_1(p_{data}, p_g) = \min_G \max_{\|D\|_L \leq 1} (\mathbb{E}_{x \sim p_{data}}[D(x)] - \mathbb{E}_{z \sim p_z}[D(G(z))])$$

⁸M. Arjovsky et al. “Wasserstein GAN”. 2017

Алгоритм обучения WGAN

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 priors.
- 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: $w \leftarrow w + \alpha \cdot \text{RMSProp}(w, g_w)$
- 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: $\theta \leftarrow \theta - \alpha \cdot \text{RMSProp}(\theta, g_\theta)$
- 12: **end while**

Алгоритм обучения WGAN

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 priors.
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:      $w \leftarrow w + \alpha \cdot \text{RMSProp}(w, g_w)$ 
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:   $\theta \leftarrow \theta - \alpha \cdot \text{RMSProp}(\theta, g_\theta)$ 
12: end while
```

- Условие короткого отображения для дискриминатора в данном случае обходится с помощью сатурации его весов: $w_L = \min(\max(-c, w), c)$, где $c \in \mathbb{R}_+$ — некоторое маленькое число

Алгоритм обучения WGAN

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 priors.
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:      $w \leftarrow w + \alpha \cdot \text{RMSProp}(w, g_w)$ 
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:   $\theta \leftarrow \theta - \alpha \cdot \text{RMSProp}(\theta, g_\theta)$ 
12: end while
```

- Условие короткого отображения для дискриминатора в данном случае обходится с помощью сатурации его весов: $w_L = \min(\max(-c, w), c)$, где $c \in \mathbb{R}_+$ — некоторое маленькое число
- Дискриминатор называется **критиком** (critic)

Алгоритм обучения WGAN

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 priors.
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:      $w \leftarrow w + \alpha \cdot \text{RMSProp}(w, g_w)$ 
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:   $\theta \leftarrow \theta - \alpha \cdot \text{RMSProp}(\theta, g_\theta)$ 
12: end while
```

- Условие короткого отображения для дискриминатора в данном случае обходится с помощью сатурации его весов: $w_L = \min(\max(-c, w), c)$, где $c \in \mathbb{R}_+$ — некоторое маленькое число
- Дискриминатор называется **критиком** (critic)
- Нет логарифмов для функции потерь, используется выход критика (например, глобальное усреднение)



Алгоритм обучения WGAN

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 priors.
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:      $w \leftarrow w + \alpha \cdot \text{RMSProp}(w, g_w)$ 
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:   $\theta \leftarrow \theta - \alpha \cdot \text{RMSProp}(\theta, g_\theta)$ 
12: end while
```

- Условие короткого отображения для дискриминатора в данном случае обходится с помощью сатурации его весов: $w_L = \min(\max(-c, w), c)$, где $c \in \mathbb{R}_+$ — некоторое маленькое число
- Дискриминатор называется **критиком** (critic)
- Нет логарифмов для функции потерь, используется выход критика (например, глобальное усреднение)
- На каждой итерации веса критика сатурируются отрезком $[-c, c]$



DCGAN vs WGAN



Генерация

DCGAN vs WGAN



- Слева DCGAN, справа — WGAN

DCGAN vs WGAN



- Слева DCGAN, справа — WGAN
- WGAN обучается примерно на 25% быстрее DCGAN

DCGAN vs WGAN



- Слева DCGAN, справа — WGAN
- WGAN обучается примерно на 25% быстрее DCGAN
- Качество сопоставимое

WGAN-GP⁹: дальнейшее развитие WGAN

- Перепишем условие $\|f\|_L \leq 1$ как:

⁹Gulrajani I. et al. "Improved training of wasserstein gans". 2017.

WGAN-GP⁹: дальнейшее развитие WGAN

- Перепишем условие $\|f\|_L \leq 1$ как:
- $|f(x) - f(y)| \leq \|x - y\|$

⁹Gulrajani I. et al. "Improved training of wasserstein gans". 2017.

WGAN-GP⁹: дальнейшее развитие WGAN

- Перепишем условие $\|f\|_L \leq 1$ как:
- $|f(x) - f(y)| \leq \|x - y\|$
- $\frac{|f(x) - f(y)|}{\|x - y\|} \leq 1$

⁹Gulrajani I. et al. "Improved training of wasserstein gans". 2017.



WGAN-GP⁹: дальнейшее развитие WGAN

- Перепишем условие $\|f\|_L \leq 1$ как:
- $|f(x) - f(y)| \leq \|x - y\|$
- $\frac{|f(x) - f(y)|}{\|x - y\|} \leq 1$
- Что при $x \rightarrow y$ дает: $\|\nabla_x f(x)\| \leq 1$

⁹Gulrajani I. et al. "Improved training of wasserstein gans". 2017.

WGAN-GP⁹: дальнейшее развитие WGAN

- Перепишем условие $\|f\|_L \leq 1$ как:
- $|f(x) - f(y)| \leq \|x - y\|$
- $\frac{|f(x) - f(y)|}{\|x - y\|} \leq 1$
- Что при $x \rightarrow y$ дает: $\|\nabla_x f(x)\| \leq 1$

Таким образом, можно внедрить т.н. gradient penalty (GP):

WGAN-GP

Пусть $x \sim p_{data}$, $y = G(z) \sim p_g$, $s = tx + (1 - t)y$, $t \in [0, 1]$. Функционал для оптимизации:

$$\max_G \min_D \left(\mathbb{E}_{z \sim p_z} [D(G(z))] - \mathbb{E}_{x \sim p_{data}} [D(x)] + \lambda \mathbb{E}_{s \sim p_s} [(\|\nabla_s D(s)\| - 1)^2] \right)$$

⁹Gulrajani I. et al. "Improved training of wasserstein gans". 2017.

WGAN-GP⁹: дальнейшее развитие WGAN

- Перепишем условие $\|f\|_L \leq 1$ как:
- $|f(x) - f(y)| \leq \|x - y\|$
- $\frac{|f(x) - f(y)|}{\|x - y\|} \leq 1$
- Что при $x \rightarrow y$ дает: $\|\nabla_x f(x)\| \leq 1$

Таким образом, можно внедрить т.н. gradient penalty (GP):

WGAN-GP

Пусть $x \sim p_{data}$, $y = G(z) \sim p_g$, $s = tx + (1 - t)y$, $t \in [0, 1]$. Функционал для оптимизации:

$$\max_G \min_D \left(\mathbb{E}_{z \sim p_z} [D(G(z))] - \mathbb{E}_{x \sim p_{data}} [D(x)] + \lambda \mathbb{E}_{s \sim p_s} [(\|\nabla_s D(s)\| - 1)^2] \right)$$

Замечание. Необходимость использования s вместо x доказывается в соотв. статье
(иdea: на линии $s = tx + (1 - t)y$ градиент имеет единичную норму).

⁹Gulrajani I. et al. "Improved training of wasserstein gans". 2017.

Условные GAN

Предположим, что мы хотим налагать условие на генерацию^{10,11} — например, класс объекта y . Тогда функция потерь для cGAN (conditional GAN):

$$\min_G \max_D [\mathbb{E}_{x \sim p_{data}, y \sim p_y} \log D(x, y) + \mathbb{E}_{z \sim p_z, y \sim p_y} \log(1 - D(G(z, y), y))]$$

¹⁰Mirza, Mehdi, and Simon Osindero. "Conditional generative adversarial nets." 2014

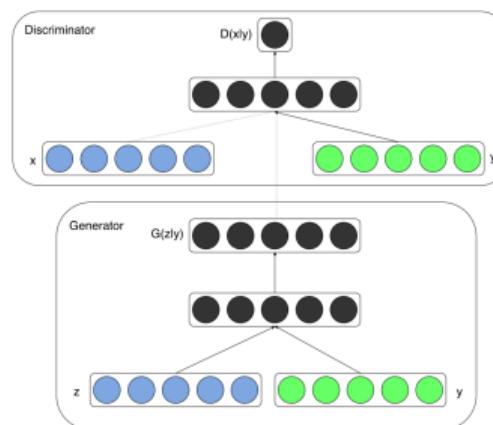
¹¹Isola, Phillip, et al. "Image-to-image translation with conditional adversarial networks" 2016



Условные GAN

Предположим, что мы хотим налагать условие на генерацию^{10,11} — например, класс объекта y . Тогда функция потерь для cGAN (conditional GAN):

$$\min_G \max_D [\mathbb{E}_{x \sim p_{data}, y \sim p_y} \log D(x, y) + \mathbb{E}_{z \sim p_z, y \sim p_y} \log(1 - D(G(z, y), y))]$$



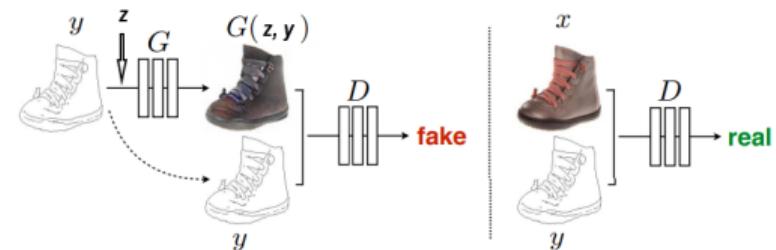
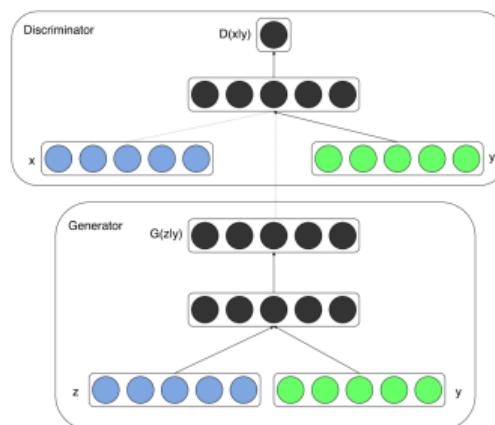
¹⁰Mirza, Mehdi, and Simon Osindero. "Conditional generative adversarial nets." 2014

¹¹Isola, Phillip, et al. "Image-to-image translation with conditional adversarial networks" 2016

Условные GAN

Предположим, что мы хотим налагать условие на генерацию^{10,11} — например, класс объекта y . Тогда функция потерь для cGAN (conditional GAN):

$$\min_G \max_D [\mathbb{E}_{x \sim p_{data}, y \sim p_y} \log D(x, y) + \mathbb{E}_{z \sim p_z, y \sim p_y} \log(1 - D(G(z, y), y))]$$



¹⁰Mirza, Mehdi, and Simon Osindero. "Conditional generative adversarial nets." 2014

¹¹Isola, Phillip, et al. "Image-to-image translation with conditional adversarial networks" 2016

Заключительные выводы

- Наиболее популярная генеративная модель для синтеза изображений — GAN

¹²Gulrajani, Ishaan, et al. “Improved training of wasserstein gans.” 2017.

¹³<https://deepfakedetectionchallenge.ai/>

Заключительные выводы

- Наиболее популярная генеративная модель для синтеза изображений — GAN
- Обучение GAN на основе минимакса крайне нестабильное

¹²Gulrajani, Ishaan, et al. “Improved training of wasserstein gans.” 2017.

¹³<https://deepfakedetectionchallenge.ai/>

Заключительные выводы

- Наиболее популярная генеративная модель для синтеза изображений — GAN
- Обучение GAN на основе минимакса крайне нестабильное
- Основные улучшения процесса обучения GAN связаны с метрикой Васерштейна¹²

¹²Gulrajani, Ishaan, et al. “Improved training of wasserstein gans.” 2017.

¹³<https://deepfakedetectionchallenge.ai/>

Заключительные выводы

- Наиболее популярная генеративная модель для синтеза изображений — GAN
- Обучение GAN на основе минимакса крайне нестабильное
- Основные улучшения процесса обучения GAN связаны с метрикой Васерштейна¹²
- Огромное число публикаций по GAN демонстрирует их полезность

¹²Gulrajani, Ishaan, et al. “Improved training of wasserstein gans.” 2017.

¹³<https://deepfakedetectionchallenge.ai/>

Заключительные выводы

- Наиболее популярная генеративная модель для синтеза изображений — GAN
- Обучение GAN на основе минимакса крайне нестабильное
- Основные улучшения процесса обучения GAN связаны с метрикой Васерштейна¹²
- Огромное число публикаций по GAN демонстрирует их полезность
- С каждым годом качество синтетических данных становится лучше

¹²Gulrajani, Ishaan, et al. “Improved training of wasserstein gans.” 2017.

¹³<https://deepfakedetectionchallenge.ai/>

Заключительные выводы

- Наиболее популярная генеративная модель для синтеза изображений — GAN
- Обучение GAN на основе минимакса крайне нестабильное
- Основные улучшения процесса обучения GAN связаны с метрикой Васерштейна¹²
- Огромное число публикаций по GAN демонстрирует их полезность
- С каждым годом качество синтетических данных становится лучше
- Мы входим в эру DeepFake¹³

¹²Gulrajani, Ishaan, et al. “Improved training of wasserstein gans.” 2017.

¹³<https://deepfakedetectionchallenge.ai/>

Время для вопросов



Спасибо за внимание!

