

# Глубокое обучение и вообще

Ульянкин Филипп и Соловей Влад

8 марта 2020 г.

**Посиделка 9:** Генеративные нейронные сети

# GAN

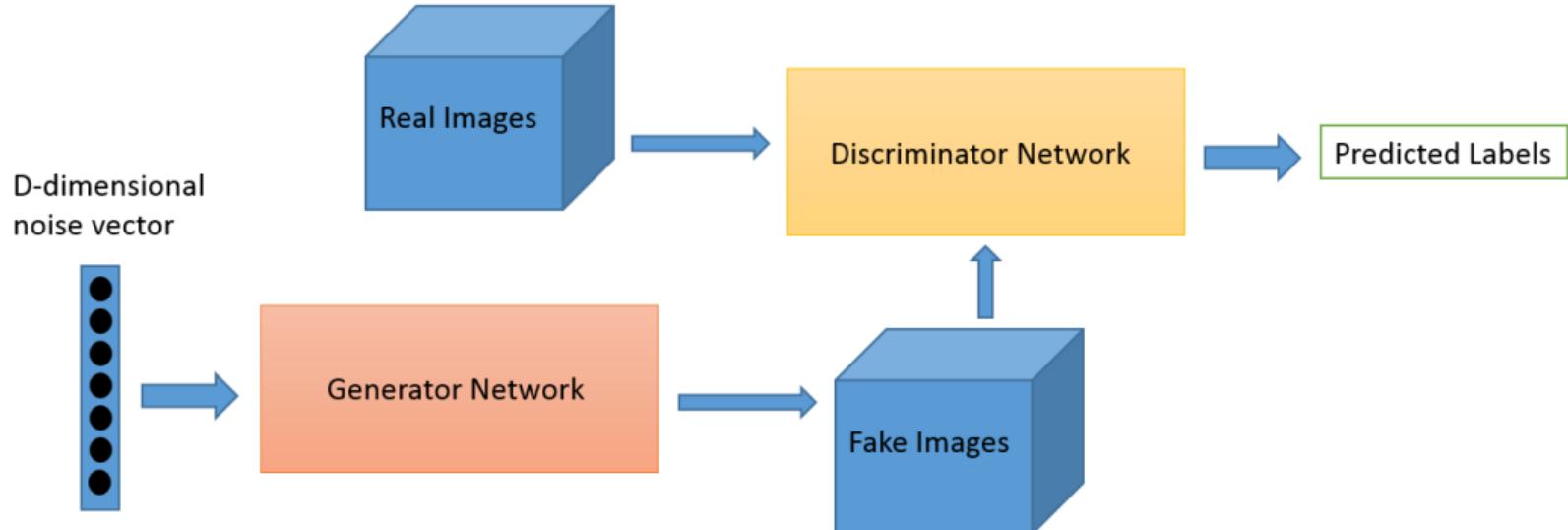


<https://thispersondoesnotexist.com>  
<https://arxiv.org/pdf/1812.04948.pdf>



<https://thiscatdoesnotexist.com>

# Генеративные сети

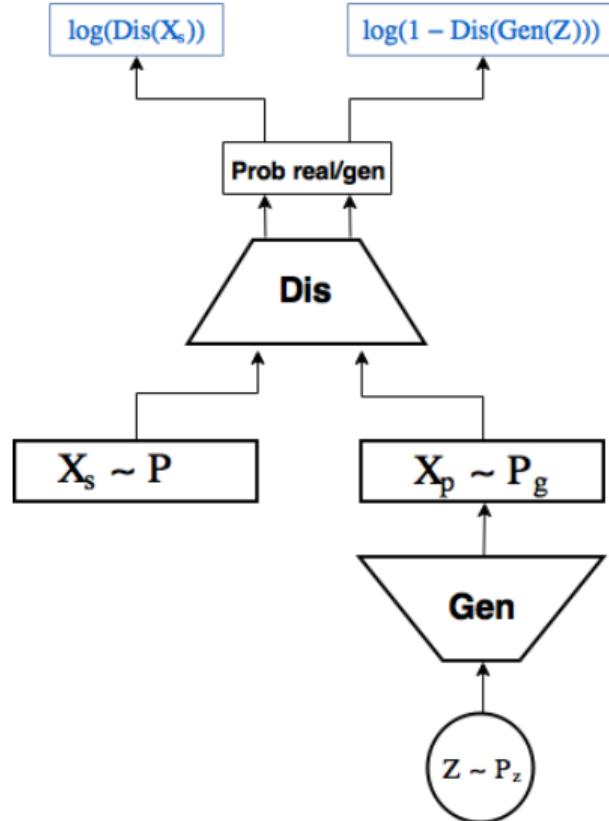


<https://arxiv.org/abs/1406.2661>

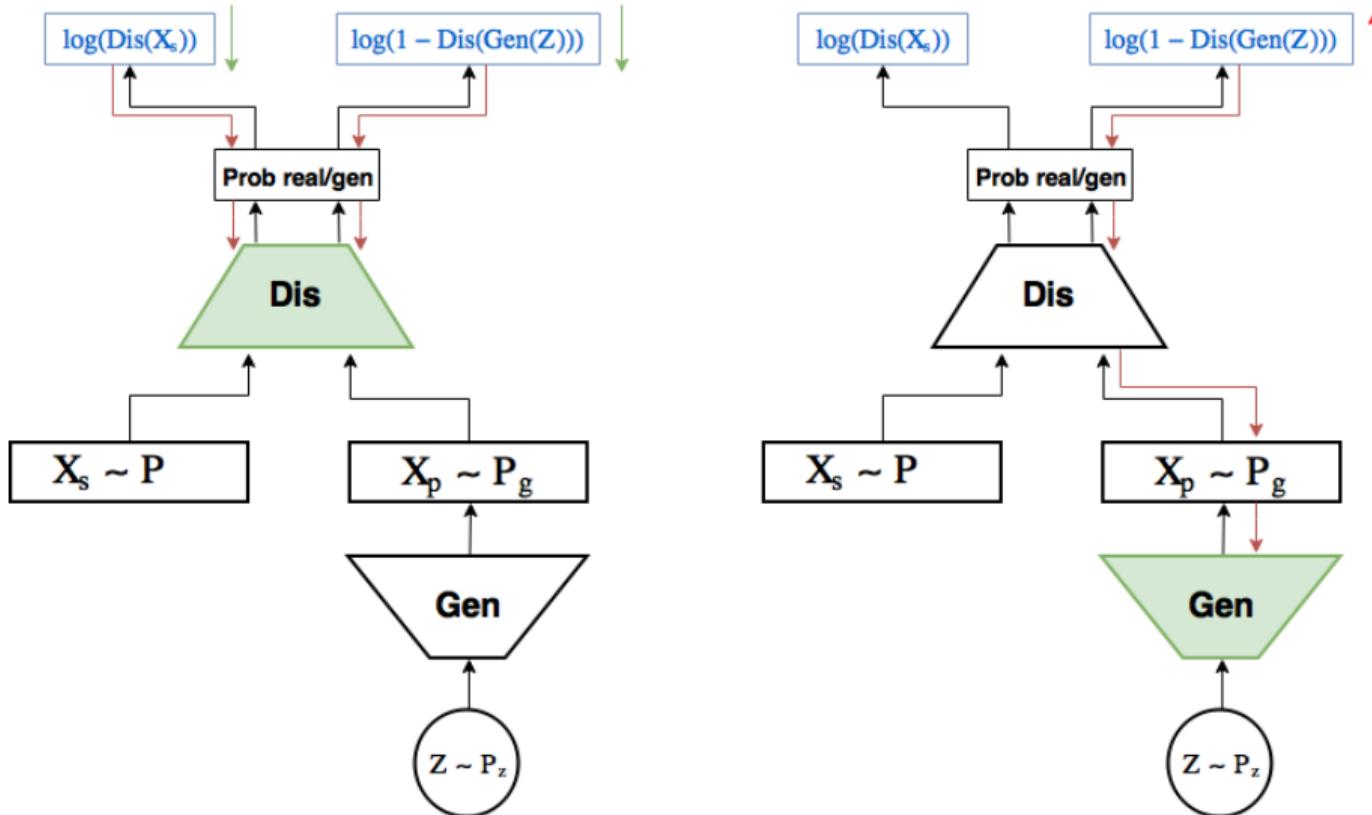
# Генеративные сети

- Две нейронных сетки
- **Генератор** сэмплит из какого-то случайного распределения вектора и пытается переработать их в картинки
- **Дискриминатор** получает на вход сгенерированные картинки и настоящие, пытается отличить реальные данные от подделки

# Генеративные сети



# Обучение сеток



# Обучение сеток

- Обучение сеток происходит поэтапно
- Сначала несколько шагов дискриминатор учится различать правду от лжи, минимизируется

$$L_D = -y_i \cdot \ln \hat{p}_i - (1 - y_i) \cdot \ln(1 - \hat{p}_i) \rightarrow \min_D$$

# Обучение сеток

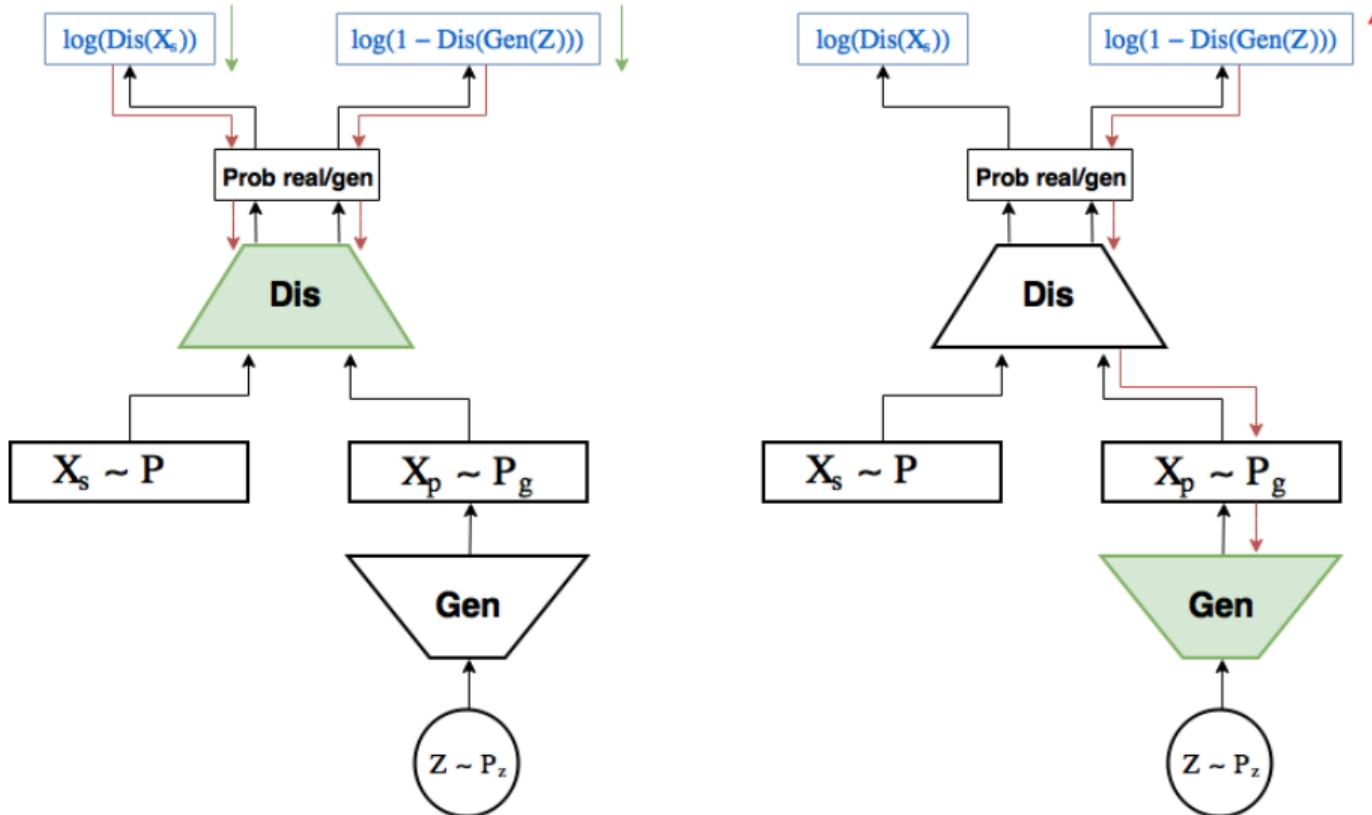
- Обучение сеток происходит поэтапно
- Сначала несколько шагов дискриминатор учится различать правду от лжи, минимизируется

$$L_D = -y_i \cdot \ln \hat{p}_i - (1 - y_i) \cdot \ln(1 - \hat{p}_i) \rightarrow \min_D$$

- После происходит один шаг обучения генератора, он пытается заставить дискриминатор ошибаться и максимизирует значение ошибки на ложных картинках

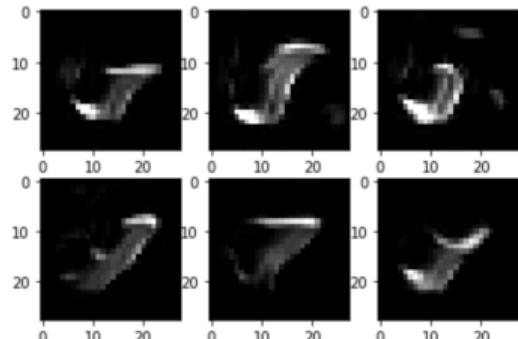
$$L_G = -\ln(1 - \hat{p}_i) \rightarrow \max_G$$

# Обучение сеток

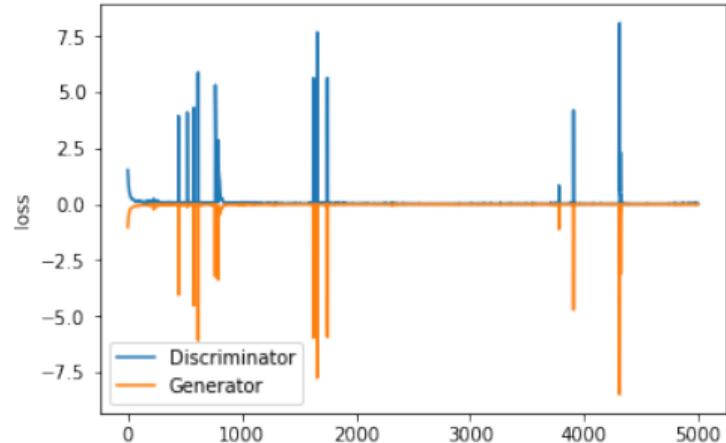
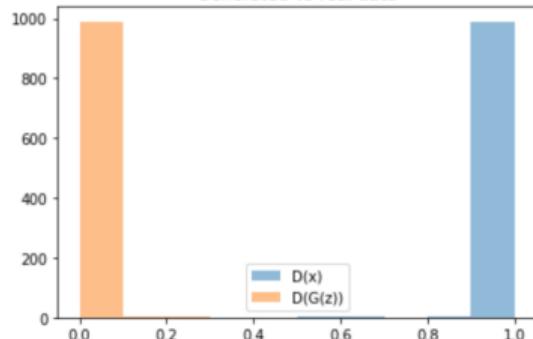


Собрал, запустил, не работает :(

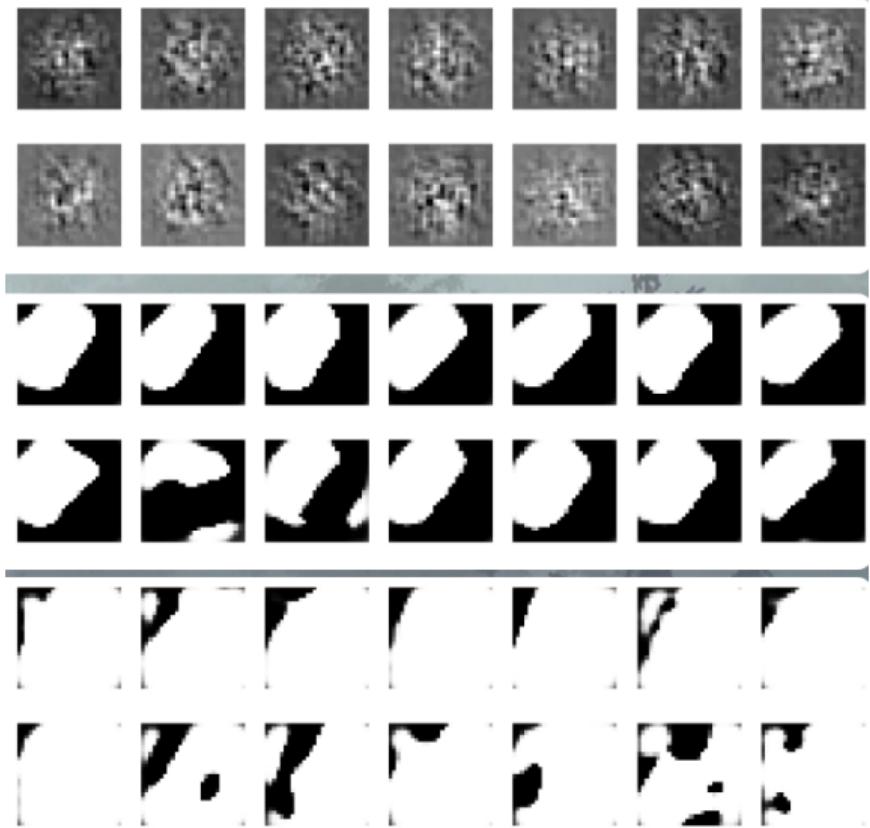
# Кекс 1



Generated vs real data



## Кекс 2





Soumith Chintala

@soumithchintala

Following

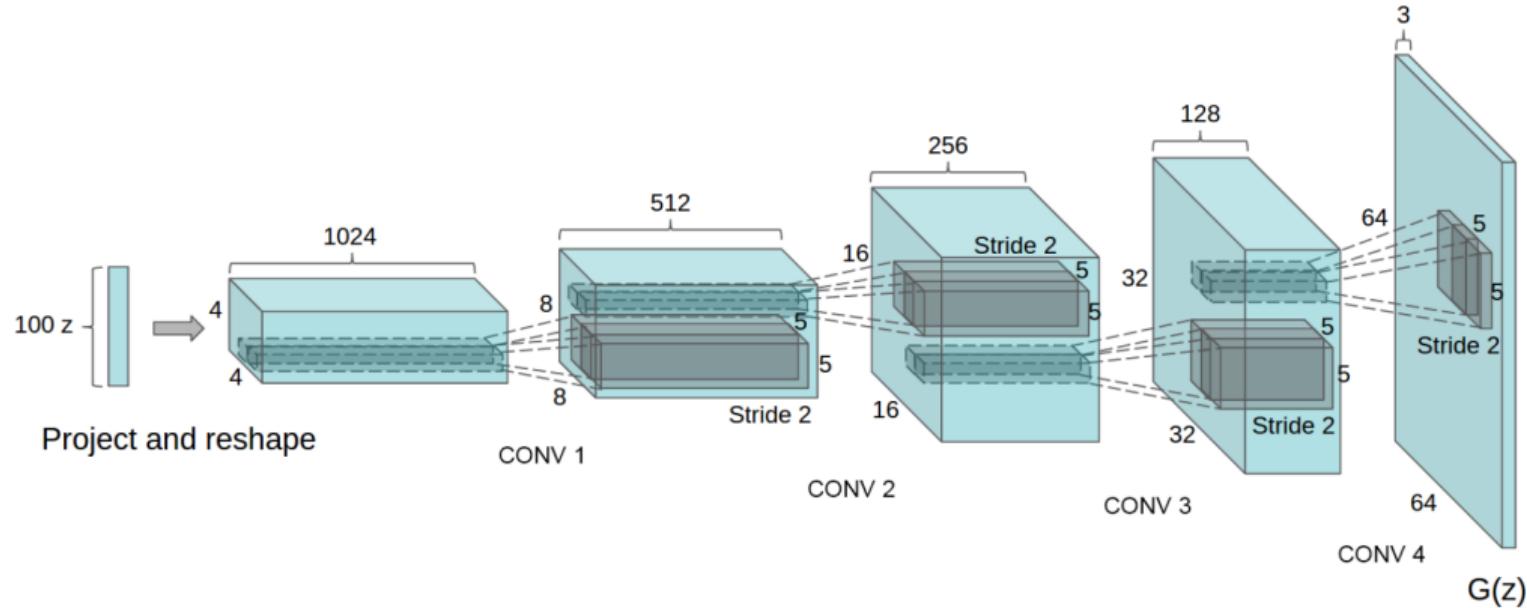
How to Train a GAN? I'm as clueless as you.  
Best I'll do is summarize the tricks, hacks  
and untold dark rituals that we've all used so  
far!



10:08 AM - 3 Dec 2016

# Свёрточные ганы

# Deep Convolutional GAN



## Советы для хороших хозяшек [1]

- Нормализуйте входы на  $[-1, 1]$  и на последний слой генератора добавляйте Tanh
- Сэмплируйте из нормального распределения, а не из равномерного
- Используйте разные BatchNorm в обеих сетках
- Можно попробовать использовать разный BatchNorm для фейковых и реальных данных
- Полносвязные слои не очень хорошая мысль, свёрточные — хорошая

## Советы для хороших хозяшек [2]

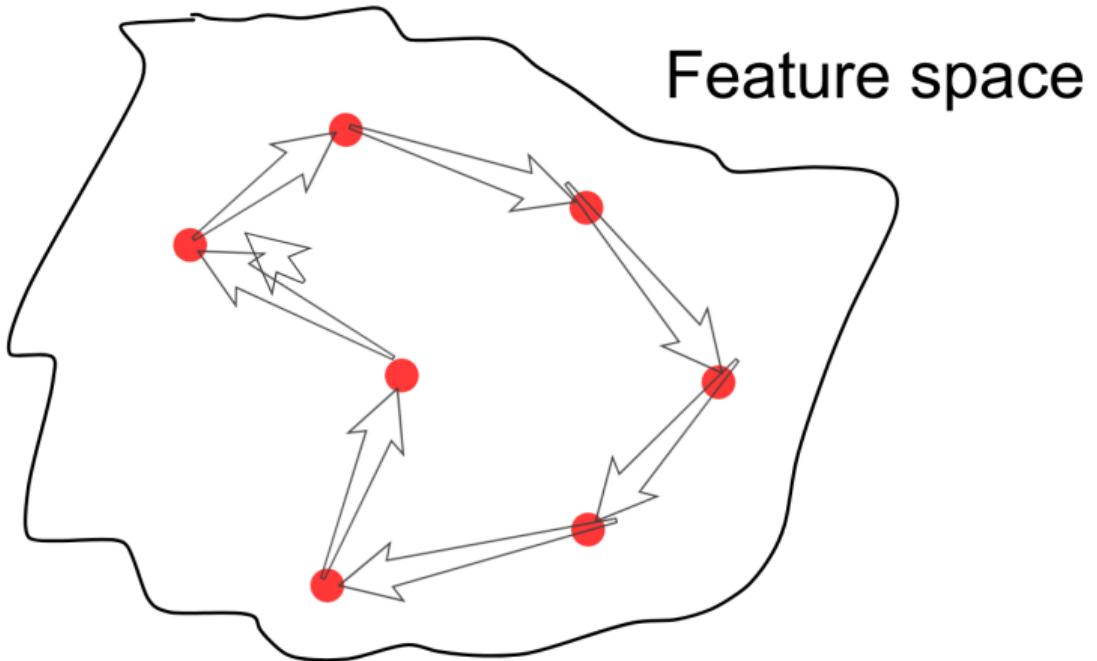
- Лучше избегать пулинга и делать stridge внутри свёрток
- Если градиенты разряжены, GAN ведёт себя стабильнее, ReLU и её модификации — хорошая идея
- Зашумляйте реальные данные, чтобы не возникало переобучения
- Если в самом начале дискриминатор сильно вырвется вперёд, обучение может заглохнуть

Ещё советы: <https://github.com/soumith/ganhacks>

# Примеры драконов



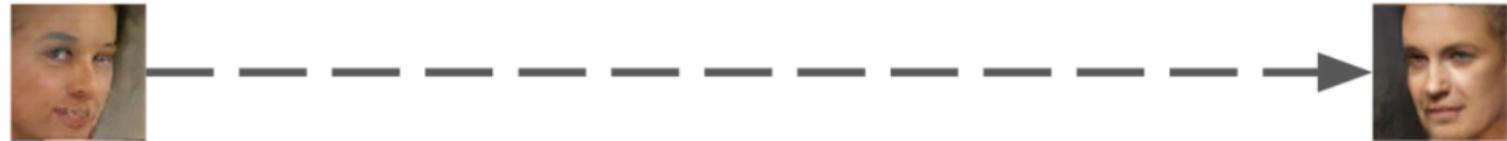
# Интерполяция



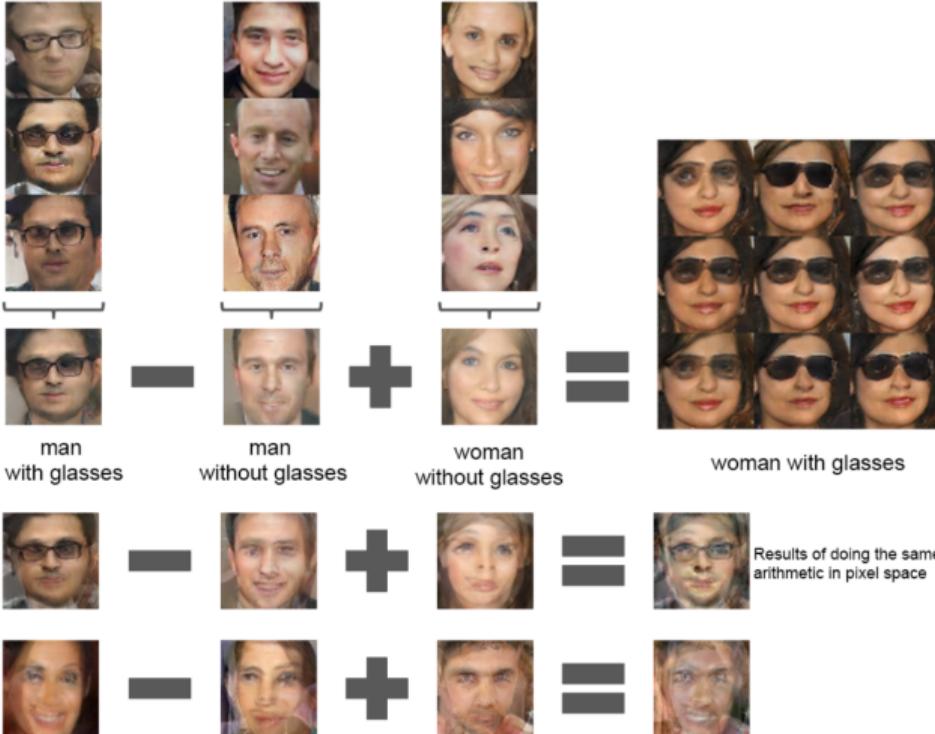
# Интерполяция



# Интерполяция



# Арифметика



# Зоопарк из ганов

# Зоопарк из ганов

## The GAN Zoo

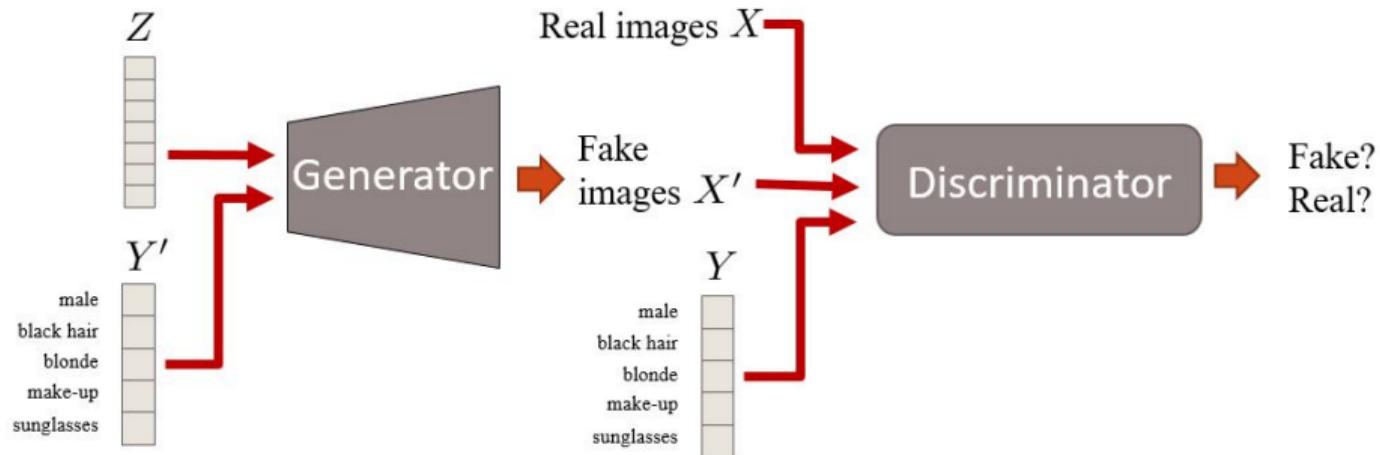
---



Every week, new GAN papers are coming out and it's hard to keep track of them all, not to mention the incredibly creative ways in which researchers are naming these GANs! So, here's a list of what started as a fun activity compiling all named GANs!

<https://github.com/hindupuravinash/the-gan-zoo>

# Conditional GAN

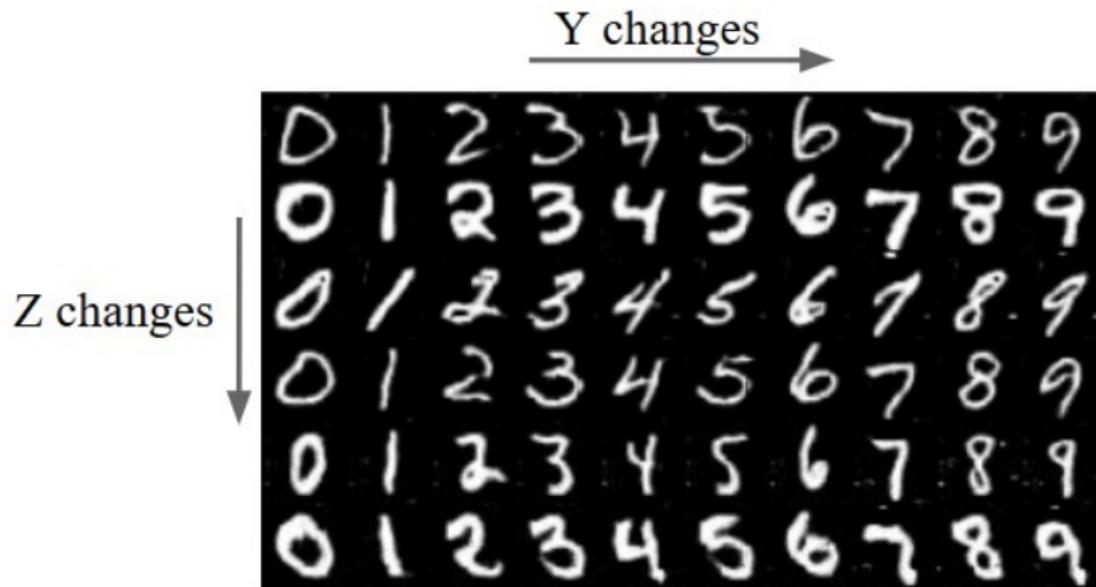


<https://arxiv.org/pdf/1411.1784.pdf>

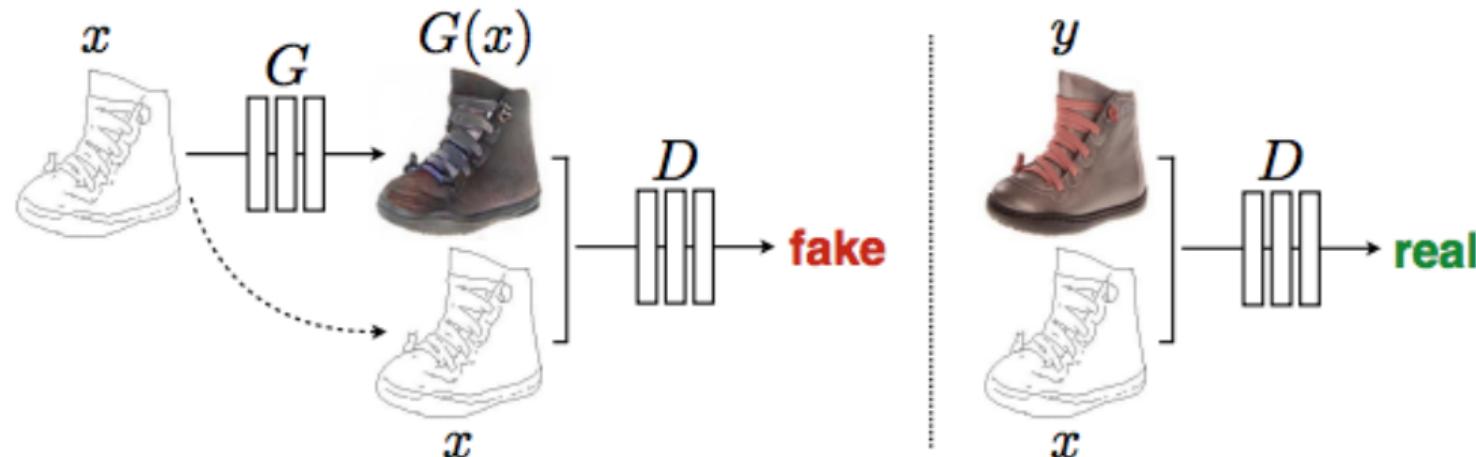
# Conditional GAN

- CGAN использует информацию из меток, если они есть
- Это позволяет генерировать не рандомные объекты, а конкретные
- В качестве меток  $u$  можно пытаться использовать и разную другую информацию
- Например, в случае лиц, можно использовать цвет волос, наличие очков, эмоции на лице и тп
- Можно даже завести несколько разных меток

# Conditional GAN



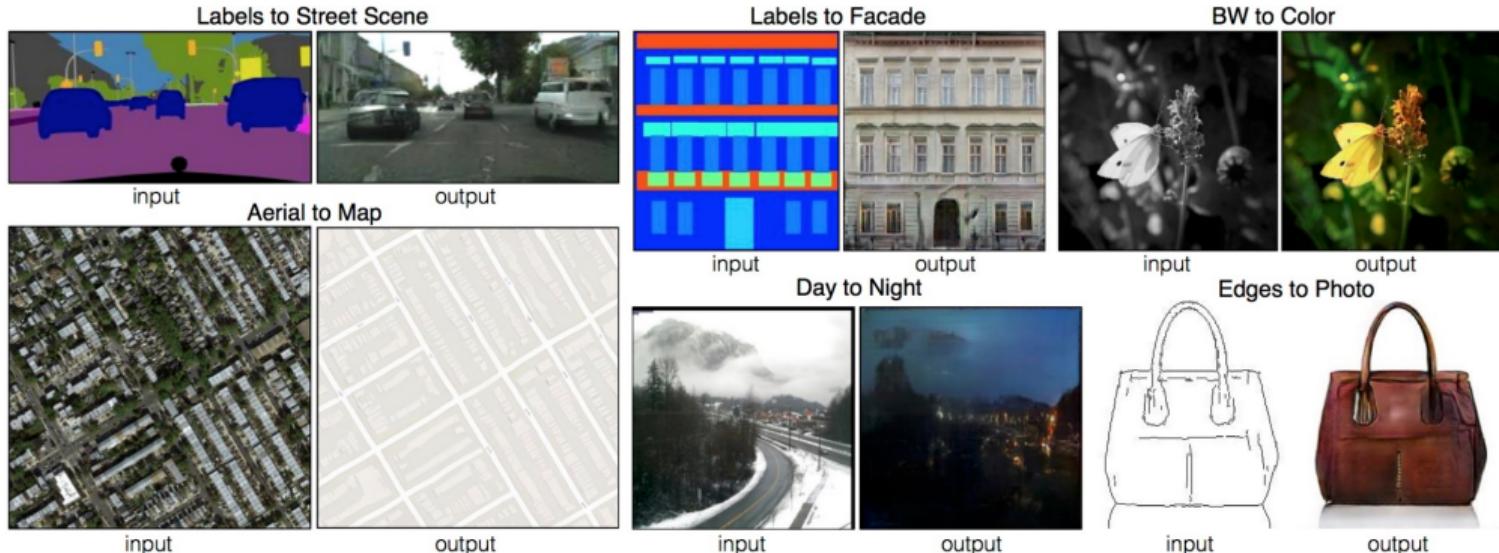
# Pix2Pix GAN



<https://arxiv.org/pdf/1808.06601.pdf>

<https://www.tensorflow.org/tutorials/generative/cyclegan>

# Pix2Pix GAN

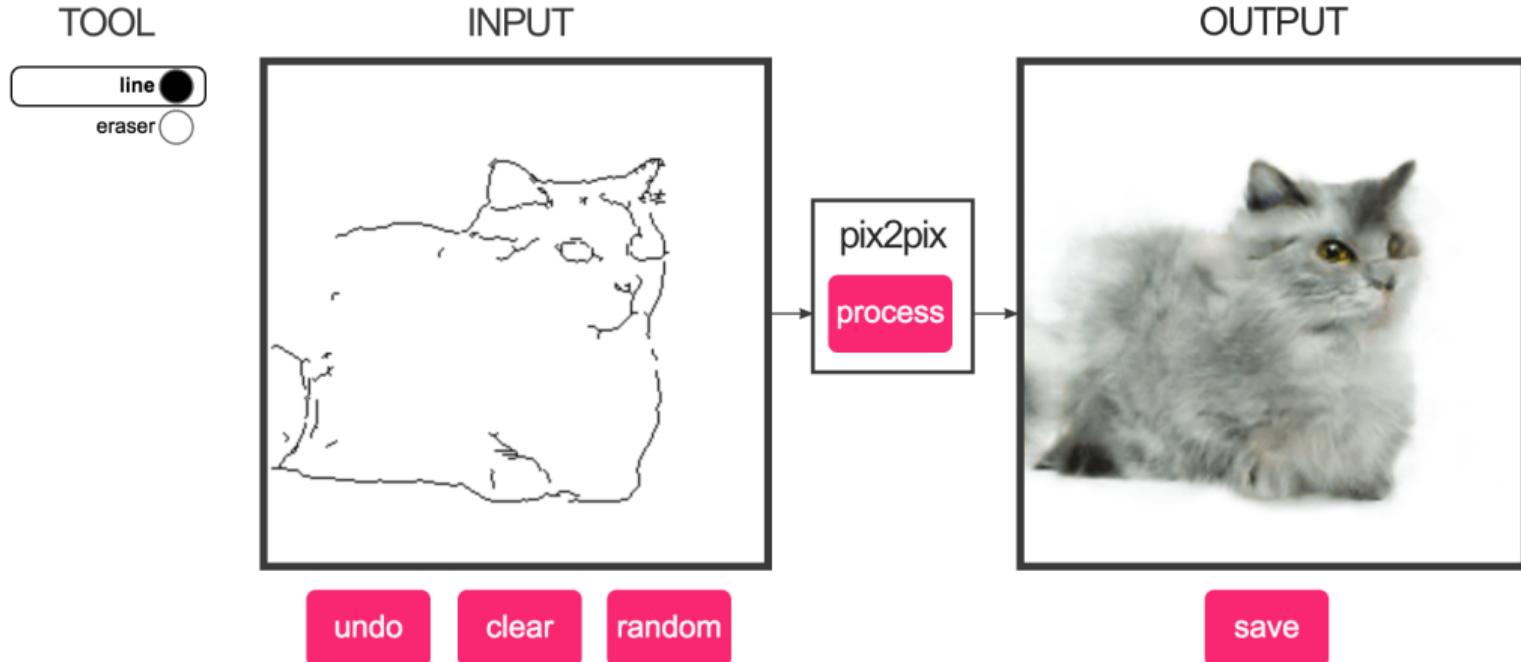


*Example results on several image-to-image translation problems. In each case we use the same architecture and objective, simply training on different data.*

<https://phillipi.github.io/pix2pix/>

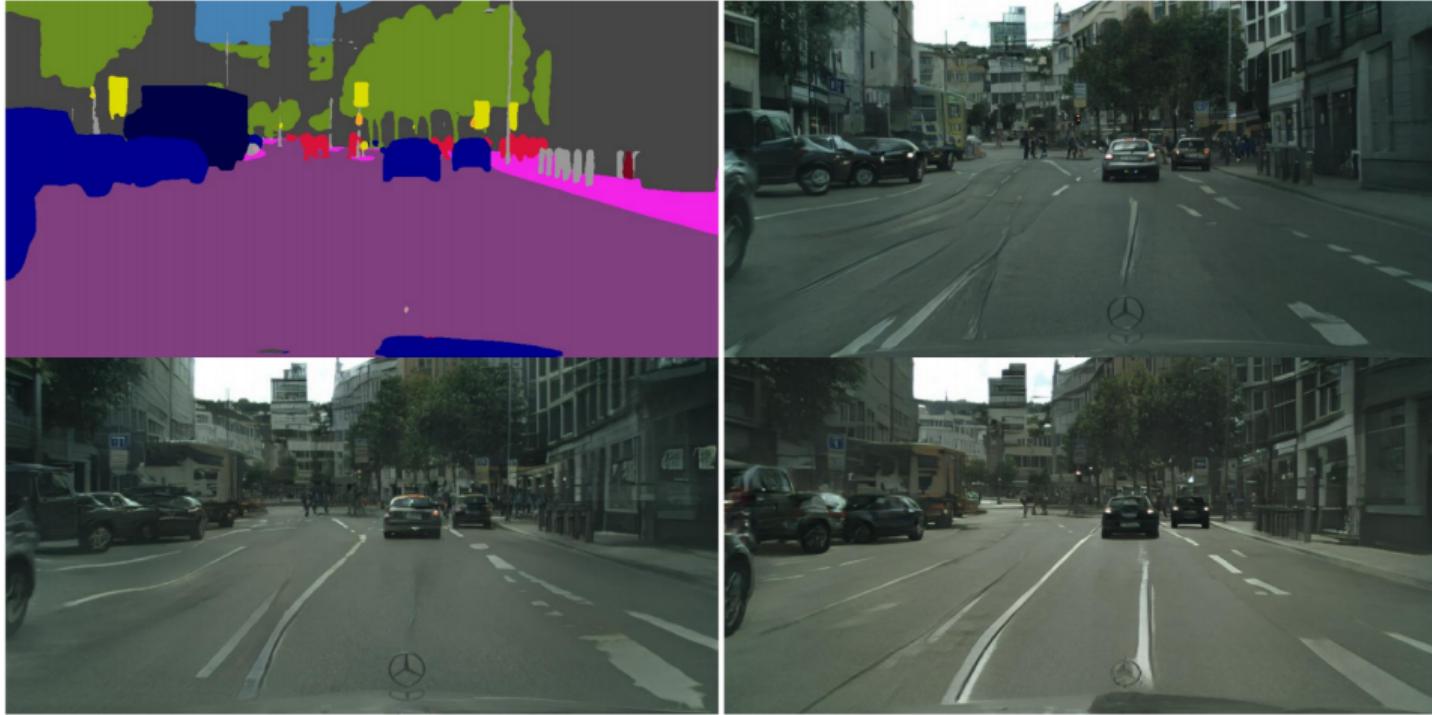
<https://www.tensorflow.org/tutorials/generative/pix2pix>

# Pix2Pix GAN



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

# Pix2Pix GAN



# CycleGAN

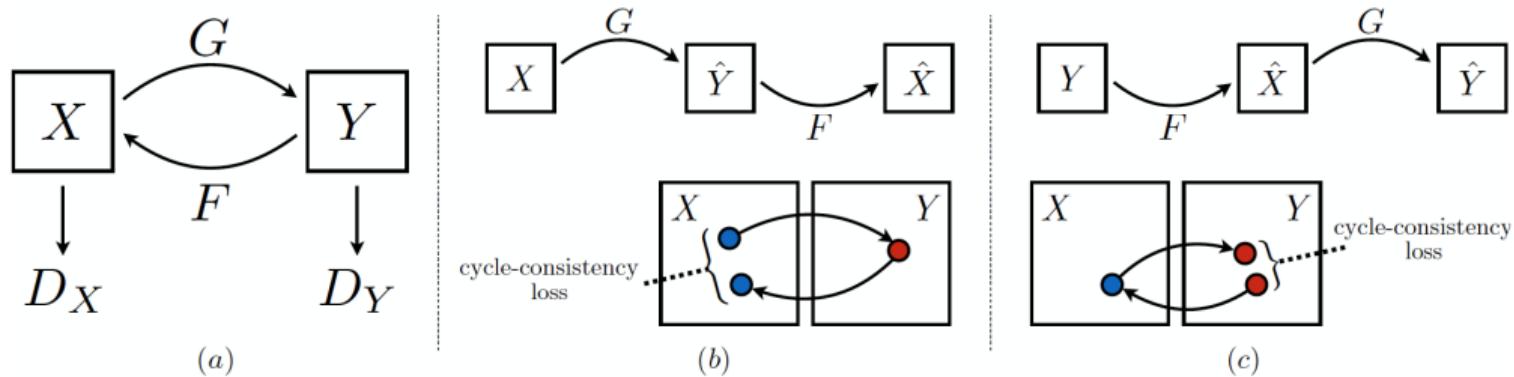


Figure 3: (a) Our model contains two mapping functions  $G : X \rightarrow Y$  and  $F : Y \rightarrow X$ , and associated adversarial discriminators  $D_Y$  and  $D_X$ .  $D_Y$  encourages  $G$  to translate  $X$  into outputs indistinguishable from domain  $Y$ , and vice versa for  $D_X$ ,  $F$ , and  $X$ . To further regularize the mappings, we introduce two “cycle consistency losses” that capture the intuition that if we translate from one domain to the other and back again we should arrive where we started: (b) forward cycle-consistency loss:  $x \rightarrow G(x) \rightarrow F(G(x)) \approx x$ , and (c) backward cycle-consistency loss:  $y \rightarrow F(y) \rightarrow G(F(y)) \approx y$

<https://arxiv.org/abs/1703.10593>

<https://www.tensorflow.org/tutorials/generative/cyclegan>

# Conditional GAN

- $X$  – объекты первого типа,  $Y$  – объекты второго типа
- Два генератора, у каждого свой дискриминатор
- Дискриминатор пытается понять правильный  $X$  пришёл на вход или он сделан из  $Y$
- Сетки генераторы сделаем обратными друг к другу функциями

# CycleGAN



# Почиташки

- Отличный обзор разных статеек про GAN на хабре, от оригинальной статьи до продвинутых идей
- Хорошая серия статей про ганы и автокодировщики
- Отлично написанная глава про GAN из уже ставшей классической книги про нейросетки от Николенко  
**ОСТОРОЖНО! ПИРАТСТВО!**
- Генератор хороров от MIT