

Глубокое обучение и вообще

Дмитрий Никулин

9 июня 2021 г.

Неделя 11: Идентификация объектов и обучение без учителя

Agenda

- Идентификация объектов
- Обучение без учителя, эмбеддинги
- Автокодировщики
- Генеративные нейронные сети

Идентификация объектов

Labeled Faces in the Wild

- Около 13 тысяч фотографий
- Около 6 тысяч человек



<http://vis-www.cs.umass.edu/lfw/>

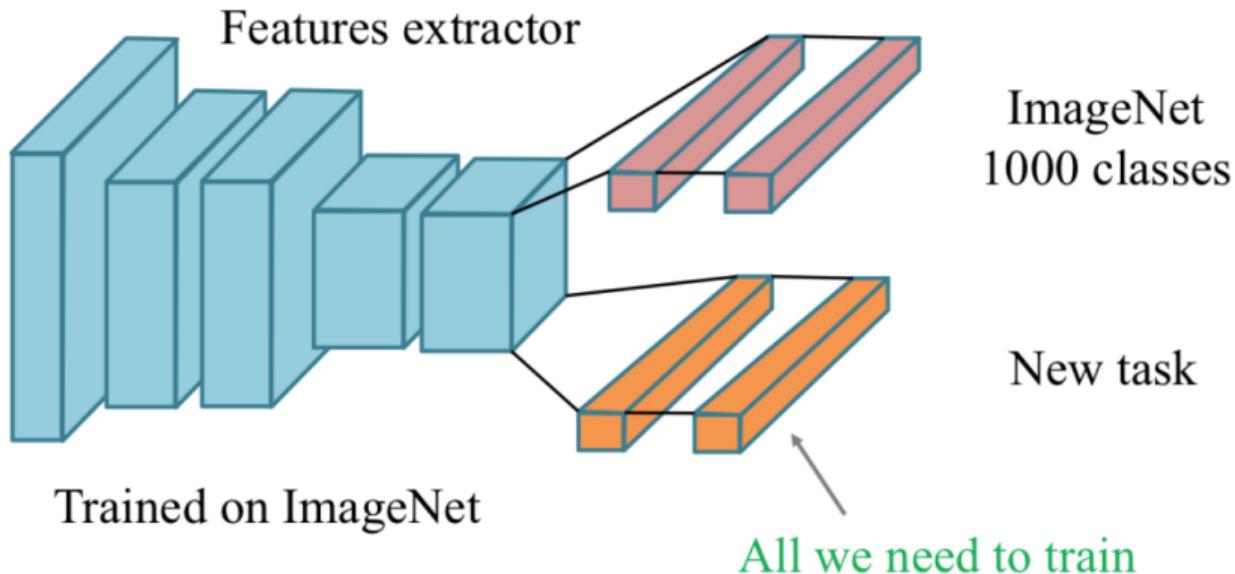
MegaFace

- 4.7 миллионов фотографий
- Около 700 тысяч человек
- В среднем 7 фото на человека



<http://megaface.cs.washington.edu/>

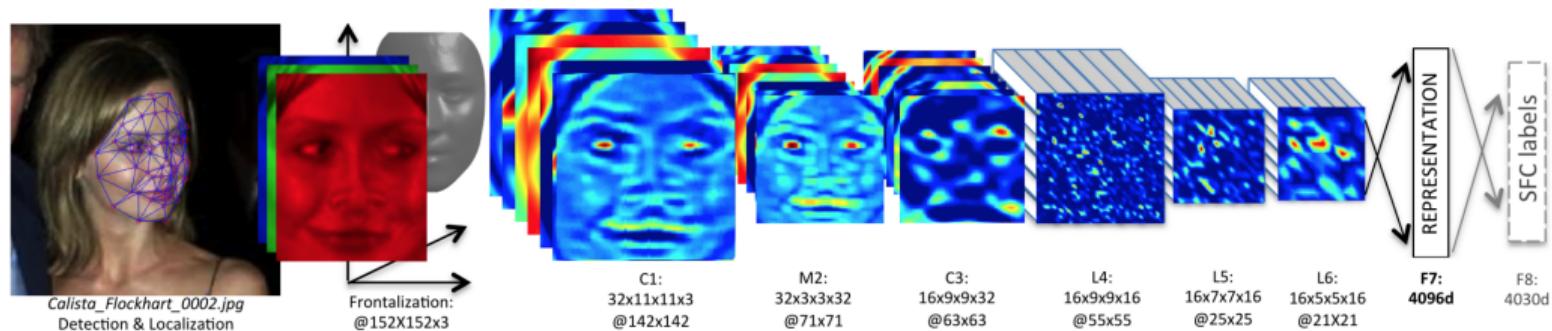
Дообучение (Transfer learning)



Дообучение (Transfer learning)

- Хорошо работает, даже если данных совсем мало
- Берём модель из другой задачи
- Заменяем последний слой на слой с нужным числом выходов
- Обучаем только его
- По сути, это обучение линейной модели на эмбеддинге картинки
- Иногда приходится дообучать часть экстрактора эмбеддингов

DeepFace (2014)



https://www.cs.toronto.edu/~ranzato/publications/taigman_cvpr14.pdf

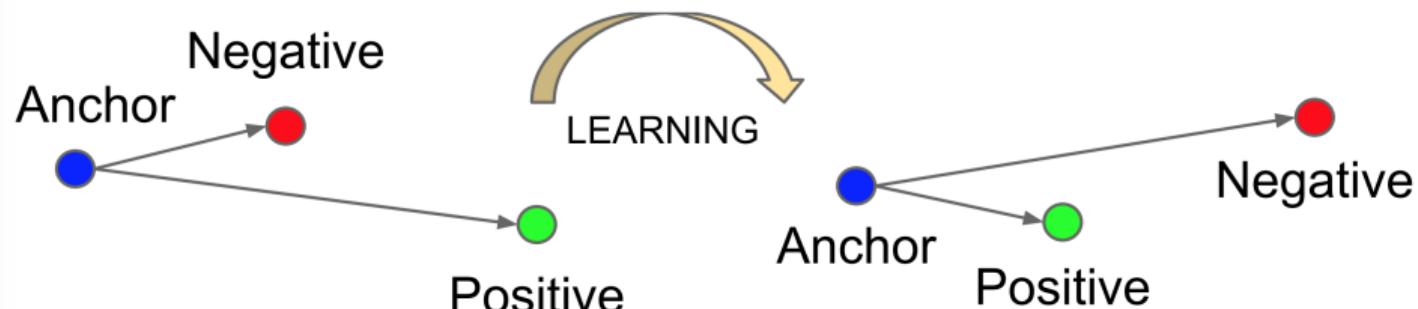
DeepFace (2014)

- Обучаем некоторую архитектуру для классификации (число классов = число людей в данных)
- Используем выходы предпоследнего слоя как признаковое описание изображения
- Считаем близость векторов по какой-нибудь метрике (например, можно взять скалярное произведение между отнормированными векторами)
- Можно сравнить расстояние с порогом, чтобы идентифицировать человека

https://www.cs.toronto.edu/~ranzato/publications/taigman_cvpr14.pdf

FaceNet

- Зачем делать так сложно? Почему бы сразу не обучать представления изображений так, чтобы фотографии одного человека имели близкие представления?



<https://arxiv.org/pdf/1503.03832.pdf>

Triplet loss

- Батчи формируем специально так, чтобы в них были и одинаковые, и разные люди
- Хотим, чтобы расстояние между эмбеддингами фотографий одного человека было по крайней мере на α меньше, чем расстояние между эмбеддингами разных людей

$$\|f(x_i^a) - f(x_i^p)\|_2^2 + \alpha \leq \|f(x_i^a) - f(x_i^n)\|_2^2$$

- Минимизируем лосс:

$$L = \sum_{i=1}^N \max \left(0, \|f(x_i^a) - f(x_i^p)\|_2^2 - \|f(x_i^a) - f(x_i^n)\|_2^2 + \alpha \right)$$

<https://arxiv.org/pdf/1503.03832.pdf>

Triplet loss

- Важно правильно выбирать триплеты
- x_i^a можно выбирать случайно
- Идея: выбирать $\operatorname{argmax}_{x_i^p} \|f(x_i^a) - f(x_i^p)\|_2^2$ (hard positive) и $\operatorname{argmin}_{x_i^n} \|f(x_i^a) - f(x_i^n)\|_2^2$ (hard negative)

Triplet loss

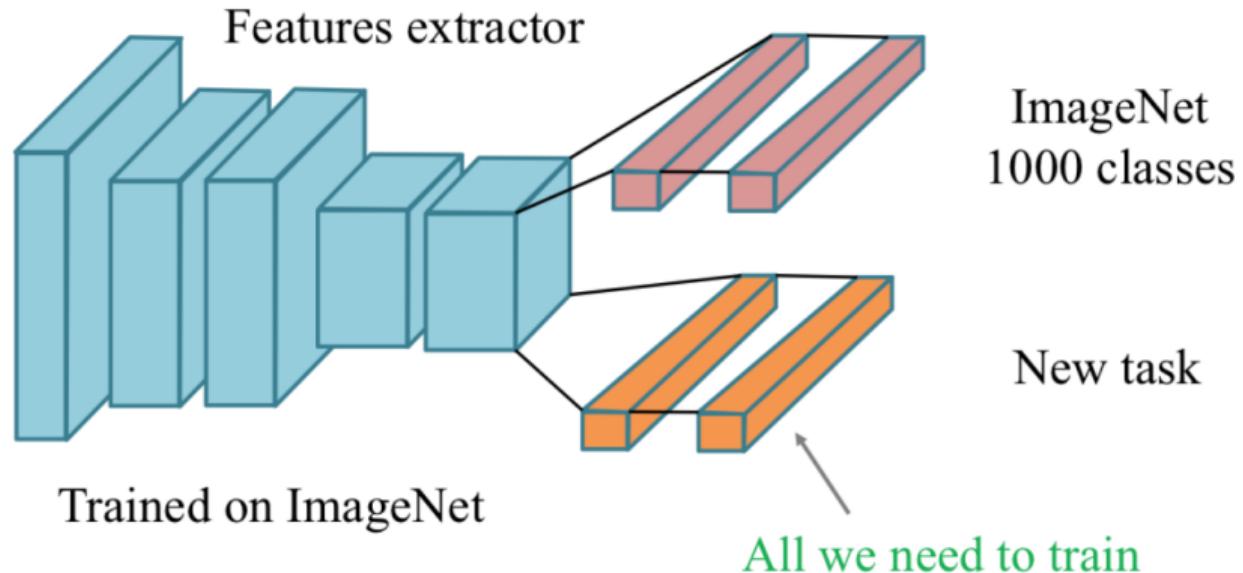
- Важно правильно выбирать триплеты
- x_i^a можно выбирать случайно
- Идея: выбирать $\operatorname{argmax}_{x_i^p} \|f(x_i^a) - f(x_i^p)\|_2^2$ (hard positive) и $\operatorname{argmin}_{x_i^n} \|f(x_i^a) - f(x_i^n)\|_2^2$ (hard negative)
- Работает не очень хорошо: модель может сколлапсировать в $f(x) \equiv 0$
- Другая идея: выбираем semi-hard negative:

$$\|f(x_i^a) - f(x_i^p)\|_2^2 < \|f(x_i^a) - f(x_i^n)\|_2^2 < \|f(x_i^a) - f(x_i^p)\|_2^2 + \alpha$$

<https://arxiv.org/pdf/1503.03832.pdf>

Обучение без учителя

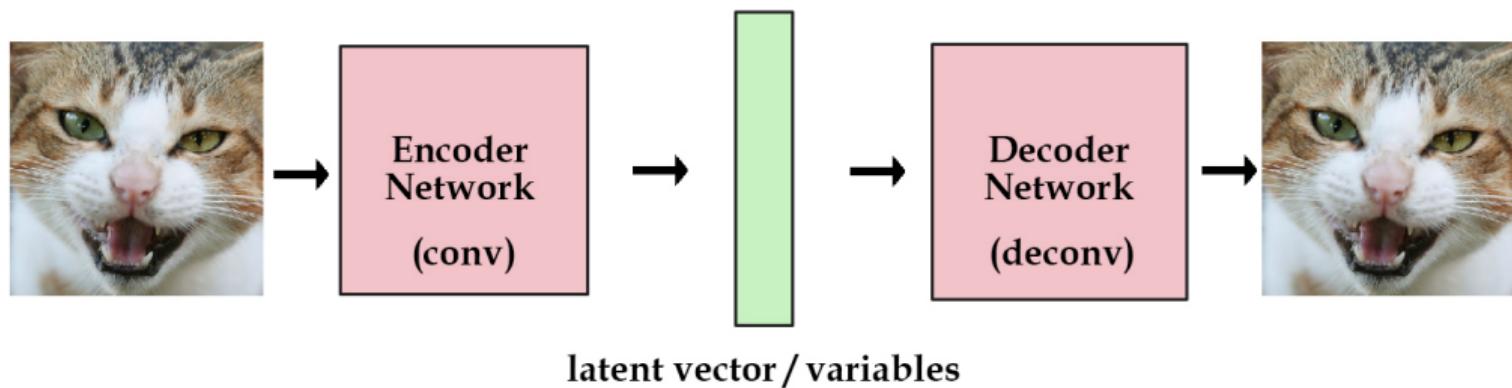
Представления изображений



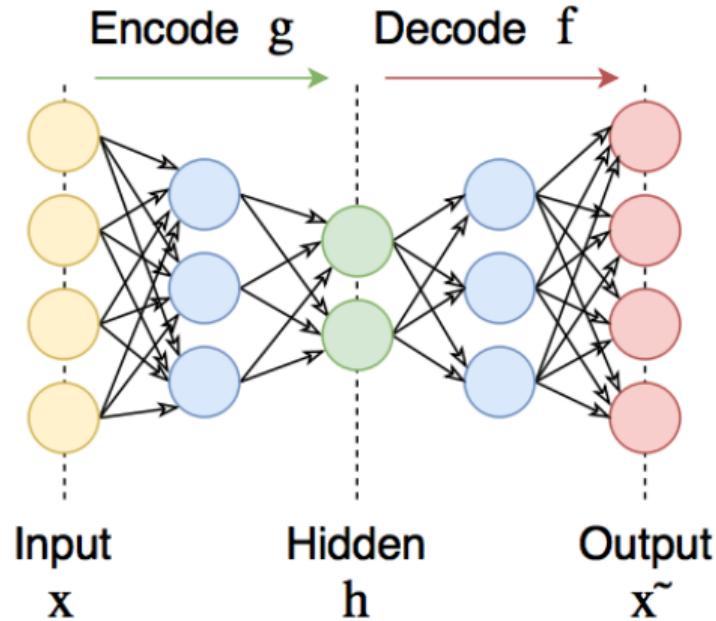
Представления изображений

- Выход предпоследнего полносвязного слоя — хорошее представления картинки
- Но для его обучения нужны изображения с разметкой
- В частности, для triplet loss надо знать, какой человек изображён на фотографии
- Может, получится строить такие представления совсем без разметки?

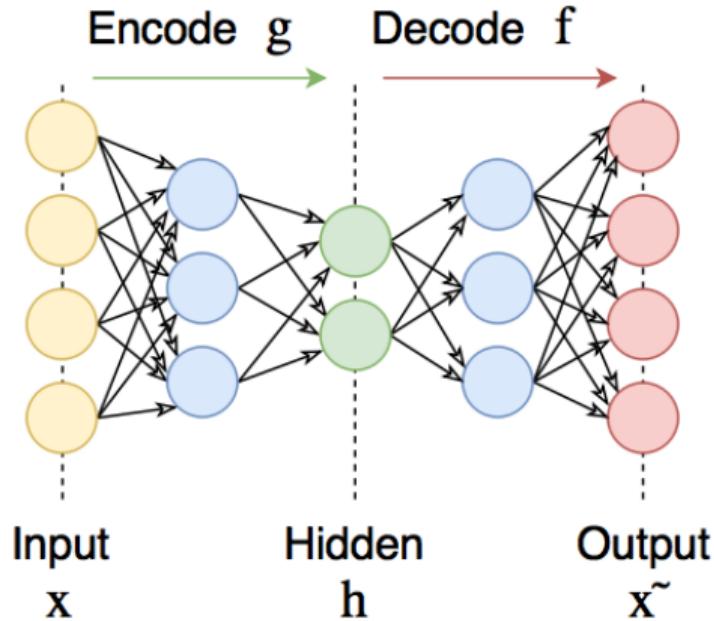
АВТОКОДИРОВЩИКИ



Автокодировщики

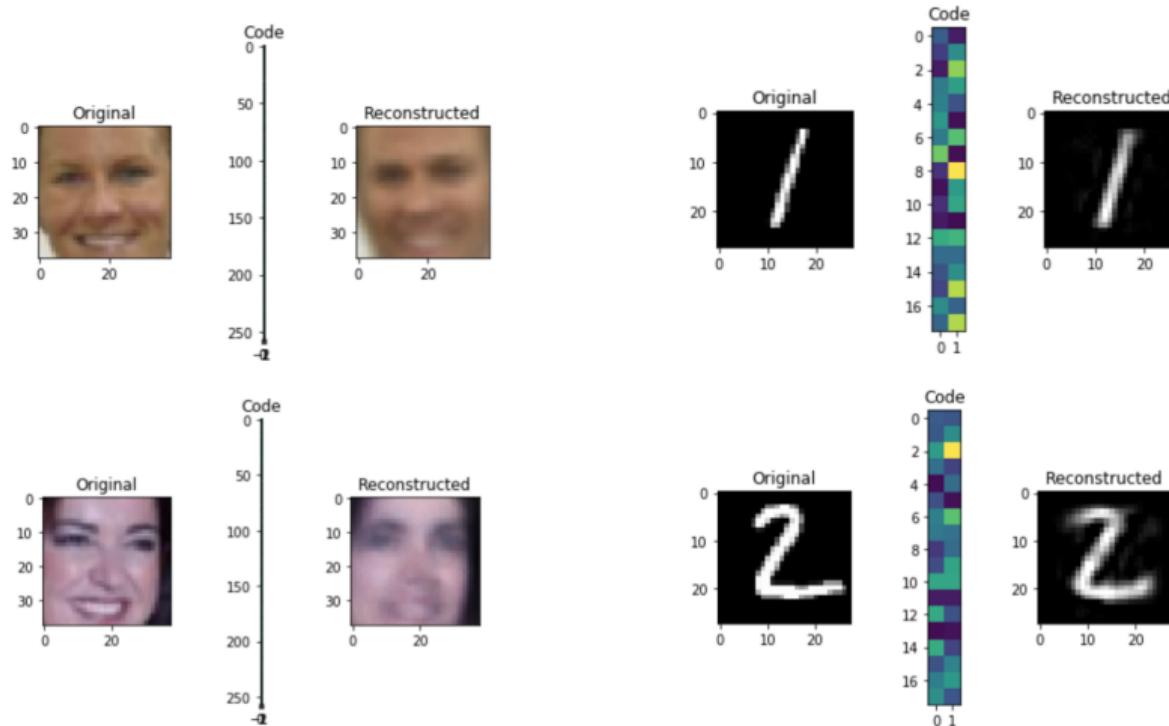


Автокодировщики



$$\frac{1}{n} \sum_{i=1}^n L(x_i, f(g(x_i))) \rightarrow \min$$

Пример сжатия



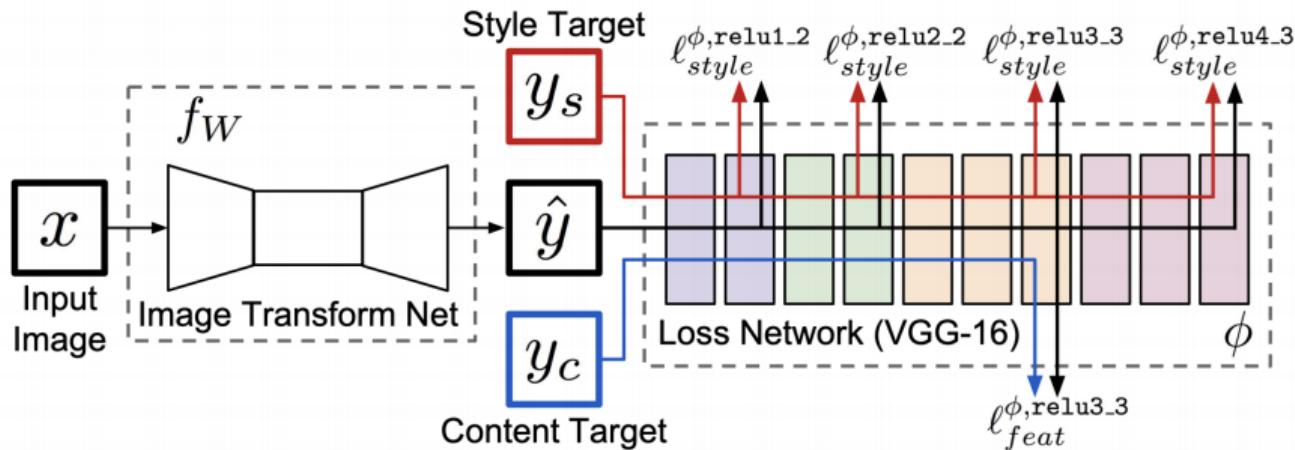
Зачем это всё?

- Поиск похожих изображений
- Трансформация изображений
- Генерация изображений
- Сжатие данных (нелинейный аналог РСА)
- Очистка картинки от шума

Автокодировщики

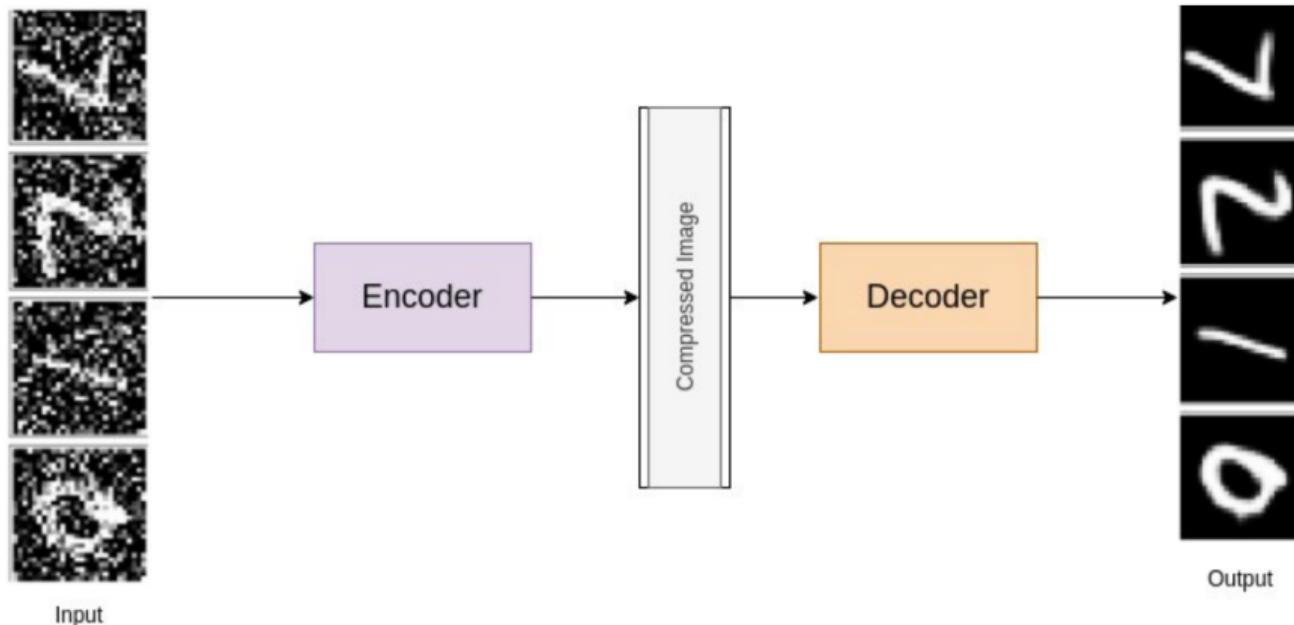
- Понижают размерность, исходная картинка восстанавливается с потерями
- Основная ценность — эмбеддинг из булочного горлышка, хочется чтобы он получился максимально качественным
- Наша архитектура сильно переобучается под выборку, для новых картинок работает хуже
- Нужна регуляризация, которая позволит извлекать смысл, а не восстанавливать пиксели в точности (фон тоже запоминается)
- Поэтому $L = \text{MSELoss}$ не идеален

Perceptual loss



<https://arxiv.org/pdf/1603.08155.pdf>

Denoising autoencoder



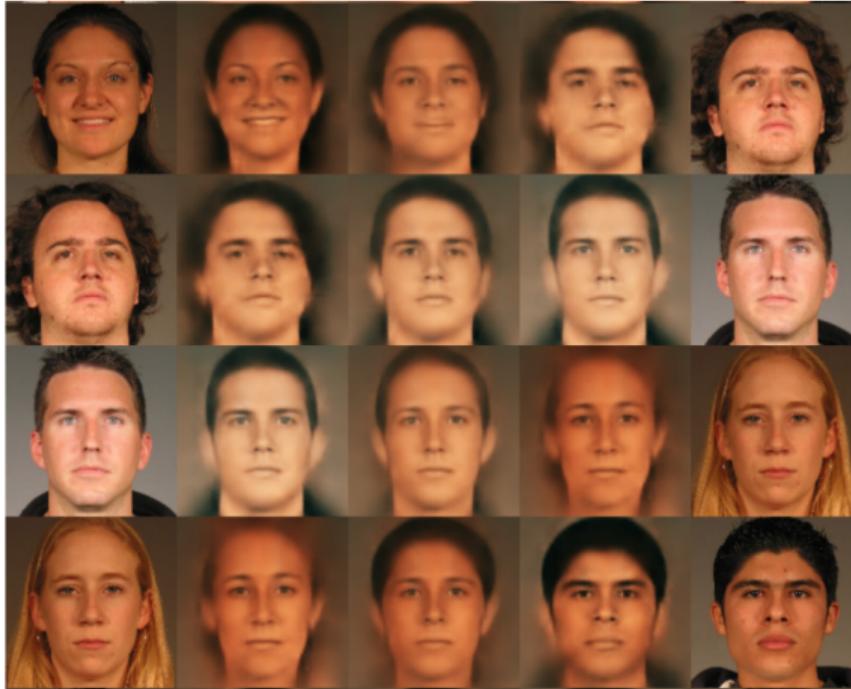
<https://medium.com/@garimanishad/>

reconstruct-corrupted-data-using-denoising-autoencoder-python-code-aeaff4

<https://www.cs.toronto.edu/~larocheh/publications/>

icml-2008-denoising-autoencoders.pdf

Morphing faces



http://essay.utwente.nl/81372/1/Heuver_MA_EEMCS.pdf

Morphing faces



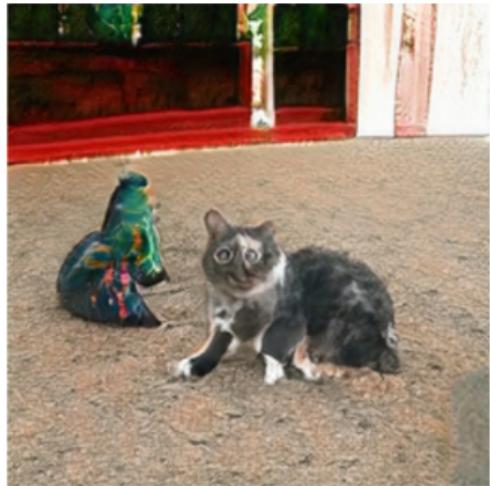
<https://www.echevarria.io/blog/lego-face-vae/index.html>

GAN



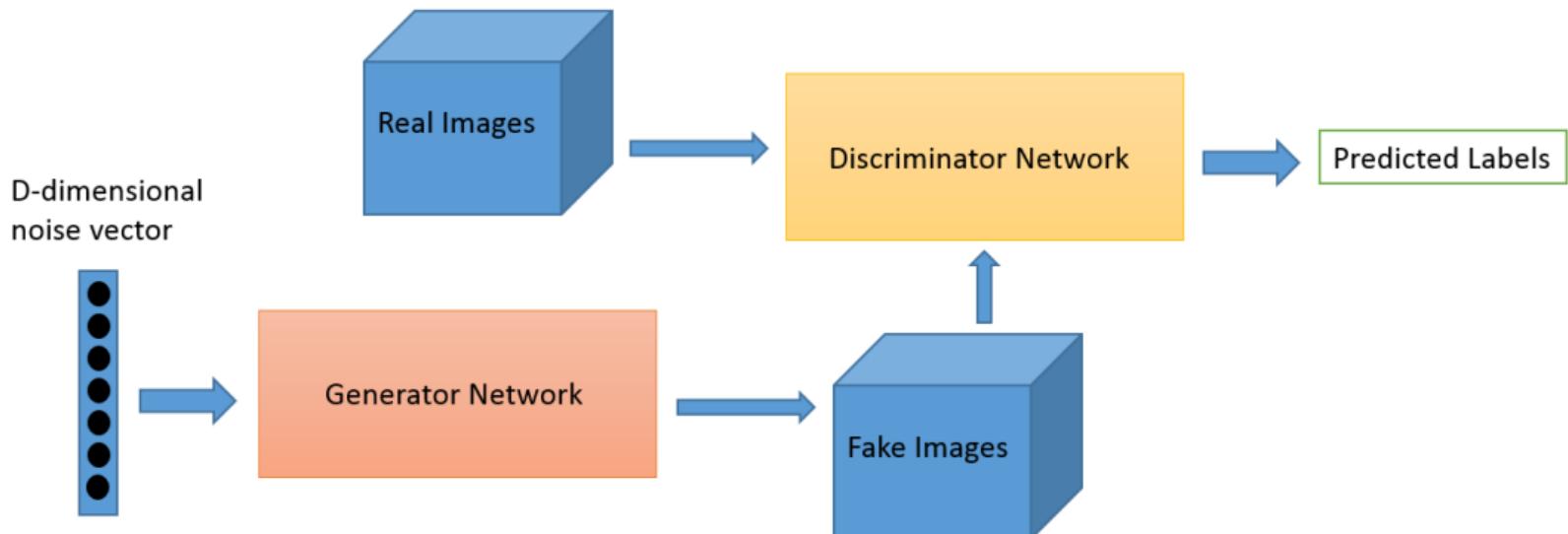
<https://thispersondoesnotexist.com>

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



<https://thiscatdoesnotexist.com>

Генеративные сети

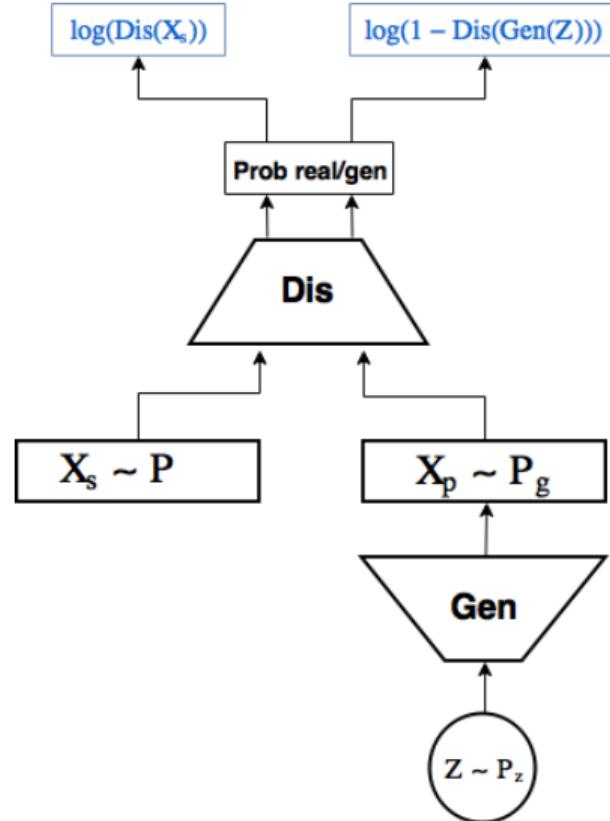


<https://arxiv.org/abs/1406.2661>

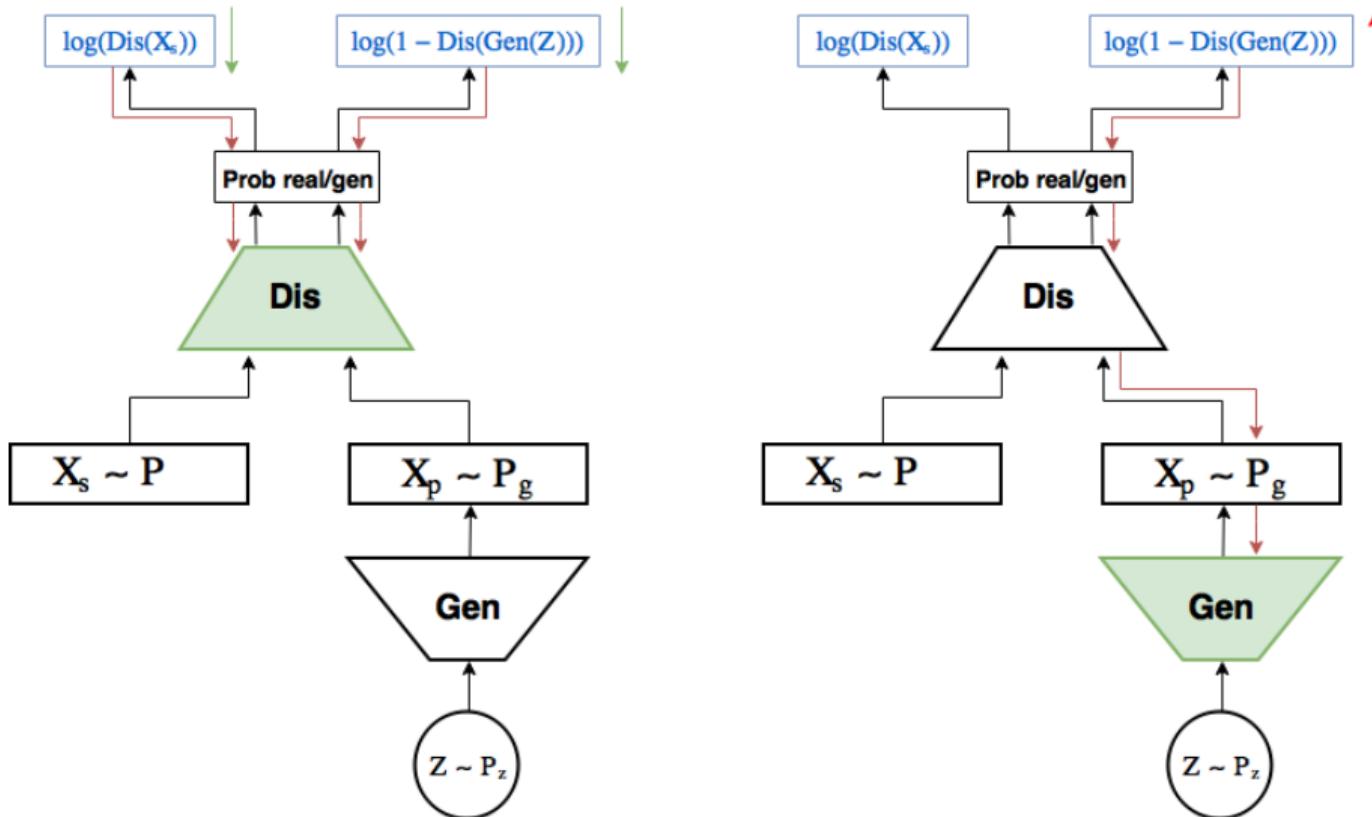
Генеративные сети

- 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

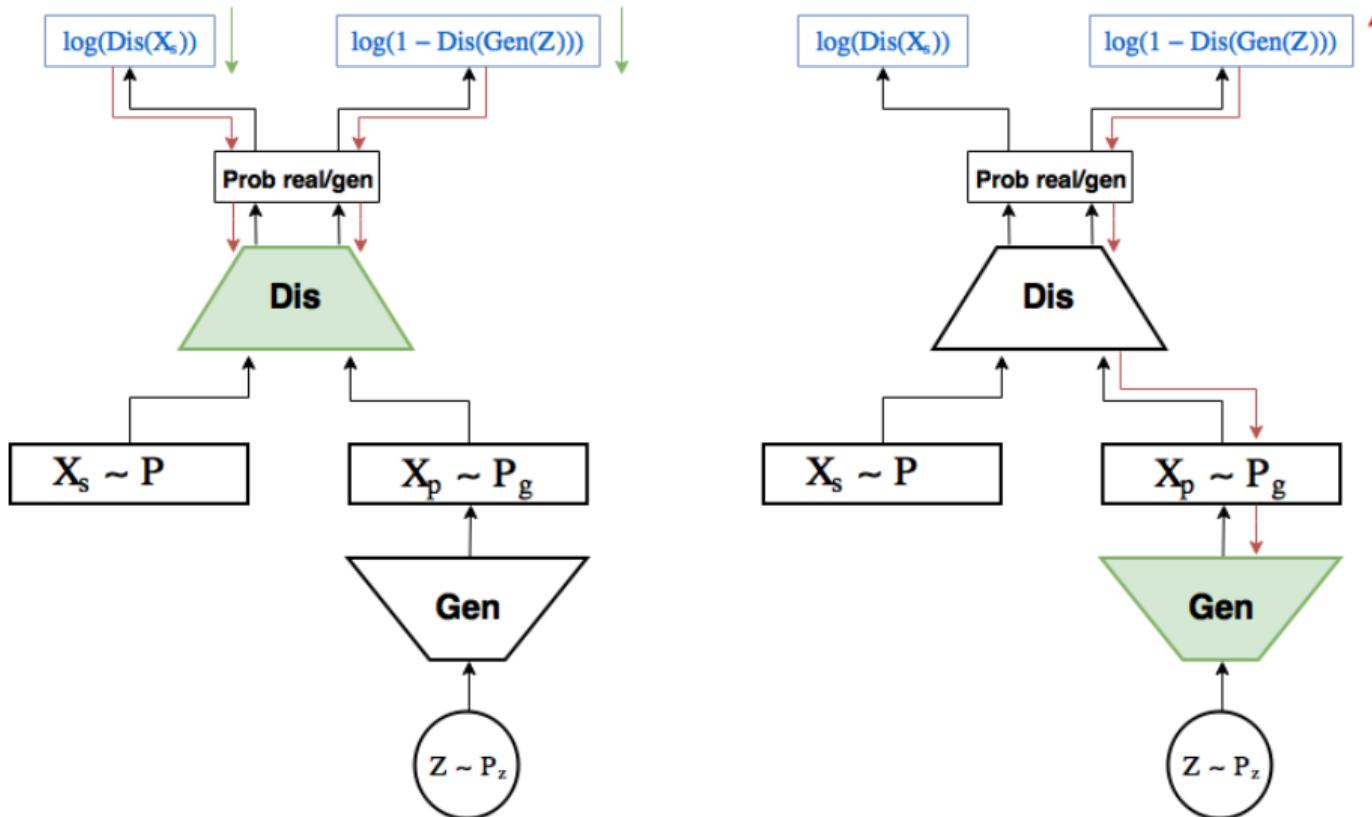
- Обучение сеток происходит поэтапно
- Сначала несколько шагов дискриминатор учится различать правду от лжи, минимизируется

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

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

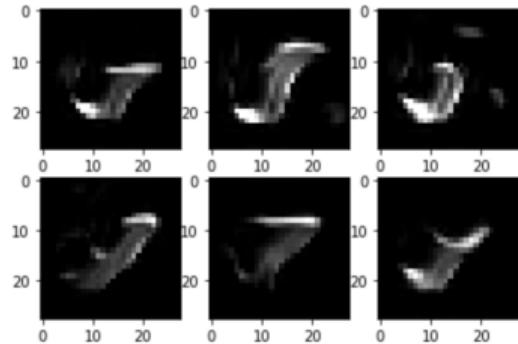
$$L_G = -\ln(1 - \hat{p}_i) \rightarrow \max_G$$

Обучение сеток

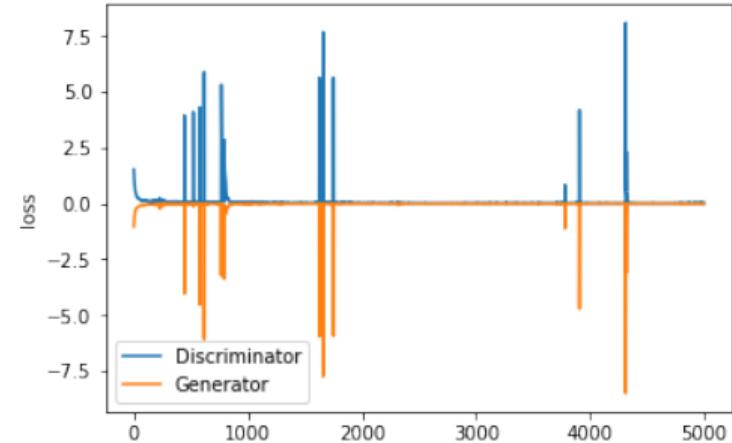
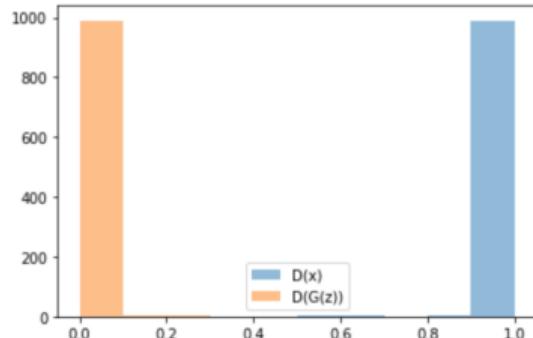


Собрал, запустил, не работает :(

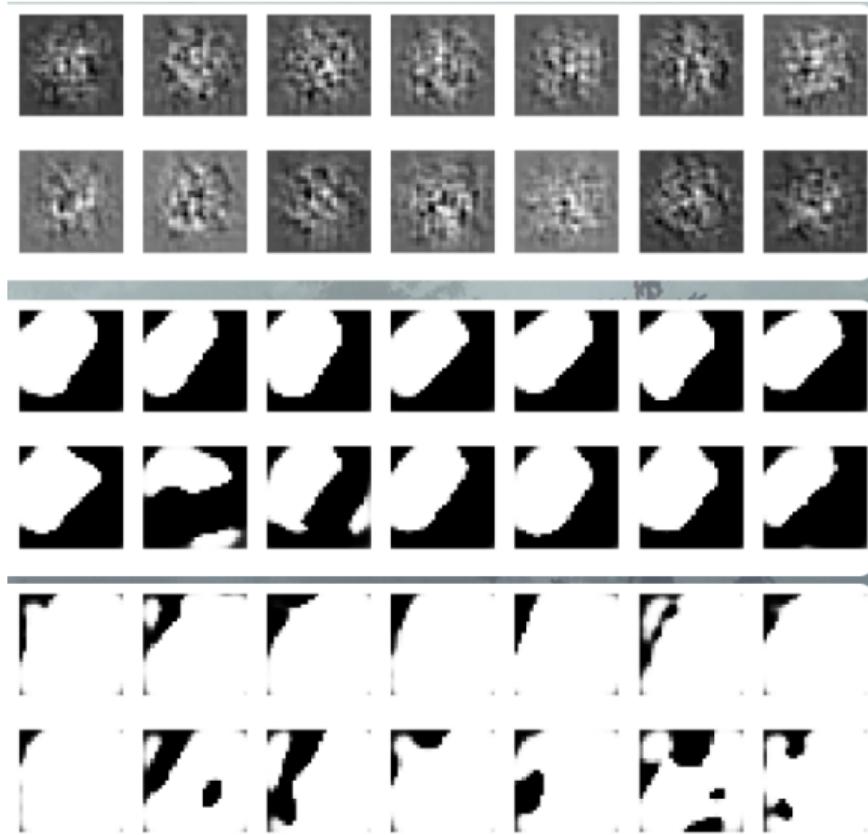
Кекс 1



Generated vs real data



Кекс 2





Soumith Chintala 
@soumithchintala

Following

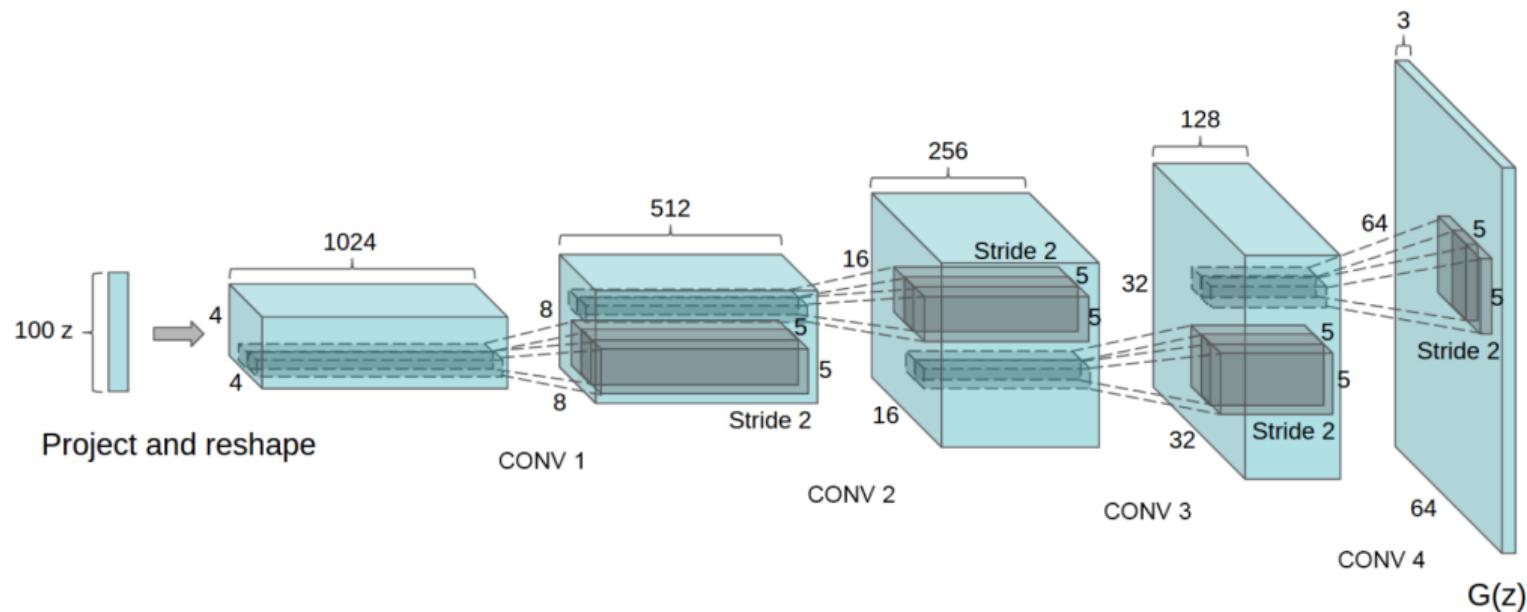
How to Train a GAN? I'm as clueless as you.
Best I'll do is summarize the tricks, hacks
and untold dark rituals that we've all used so
far!



10:08 AM - 3 Dec 2016

Свёрточные ганы

Deep Convolutional GAN



<https://arxiv.org/pdf/1511.06434.pdf>

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]

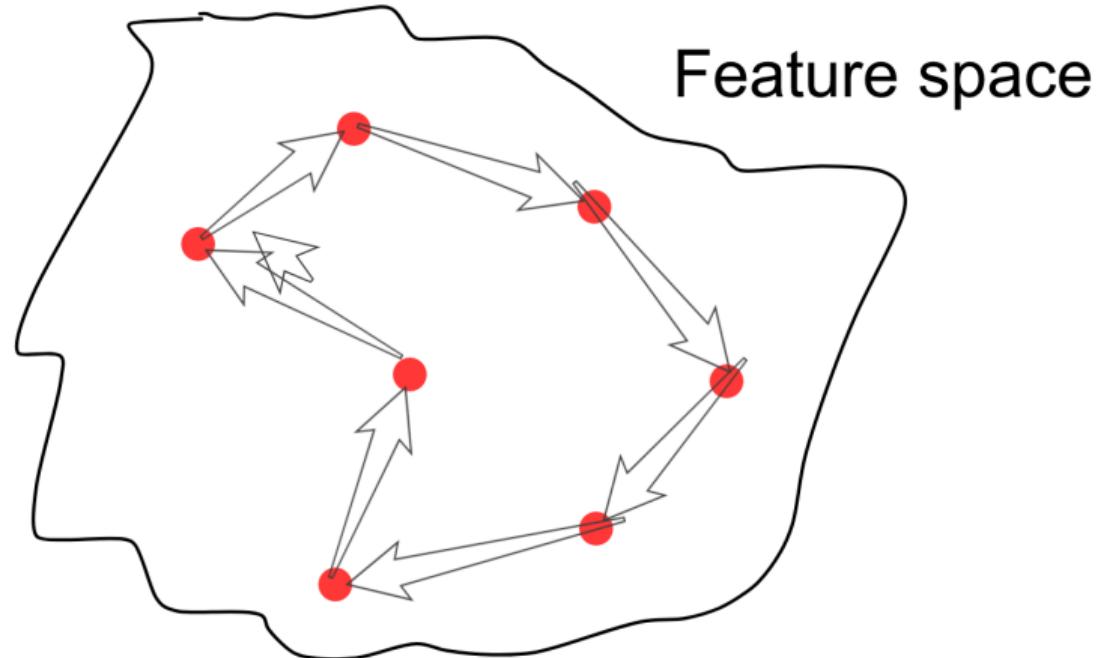
- Лучше избегать пулинга и делать stride внутри свёрток
- Если градиенты разрежены, GAN ведёт себя стабильнее, ReLU и её модификации — хорошая идея
- Зашумляйте реальные данные, чтобы не возникало переобучения
- Если в самом начале дискриминатор сильно вырвется вперёд, обучение может заглохнуть

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

Примеры драконов



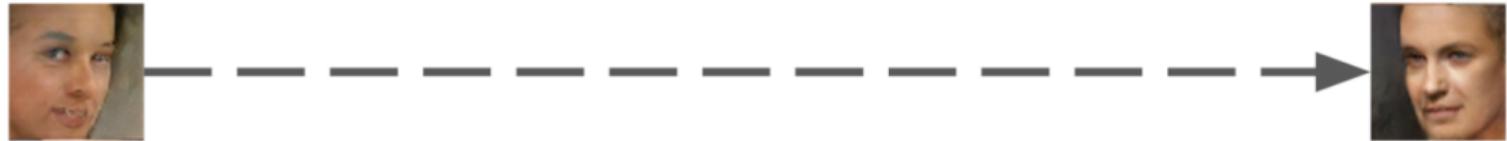
Интерполяция



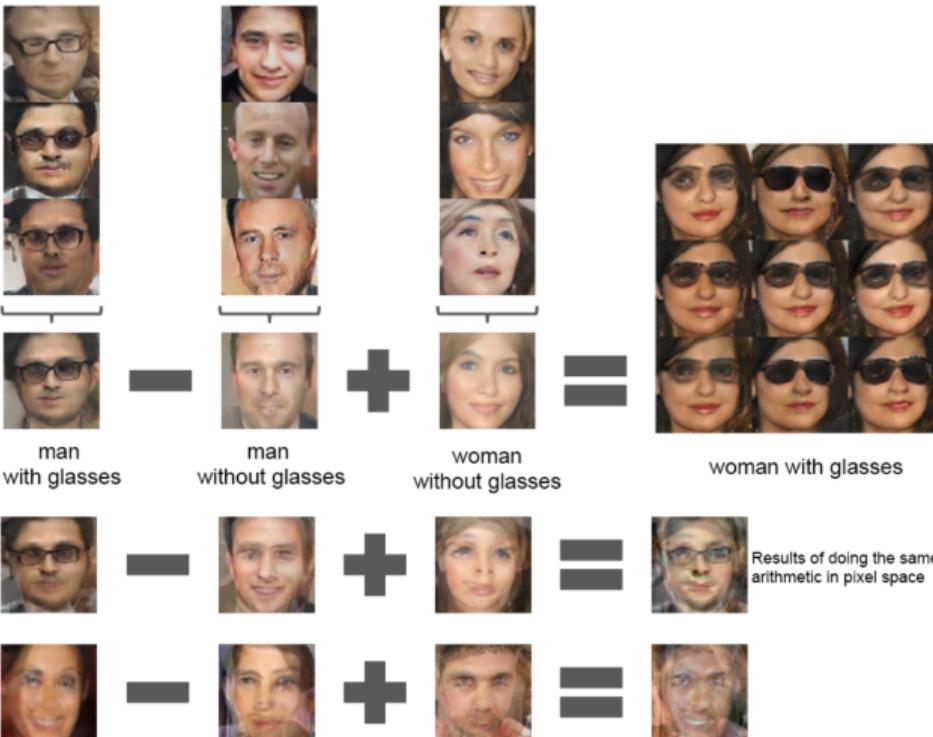
Интерполяция



Интерполяция



Арифметика



Зоопарк из ганов

Зоопарк из ганов

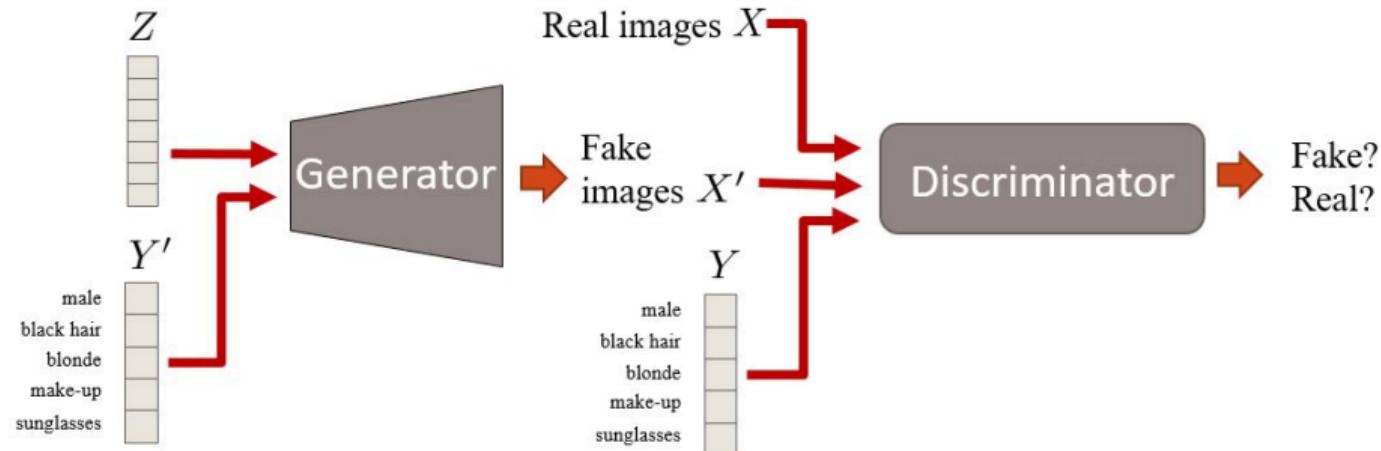
The GAN Zoo



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

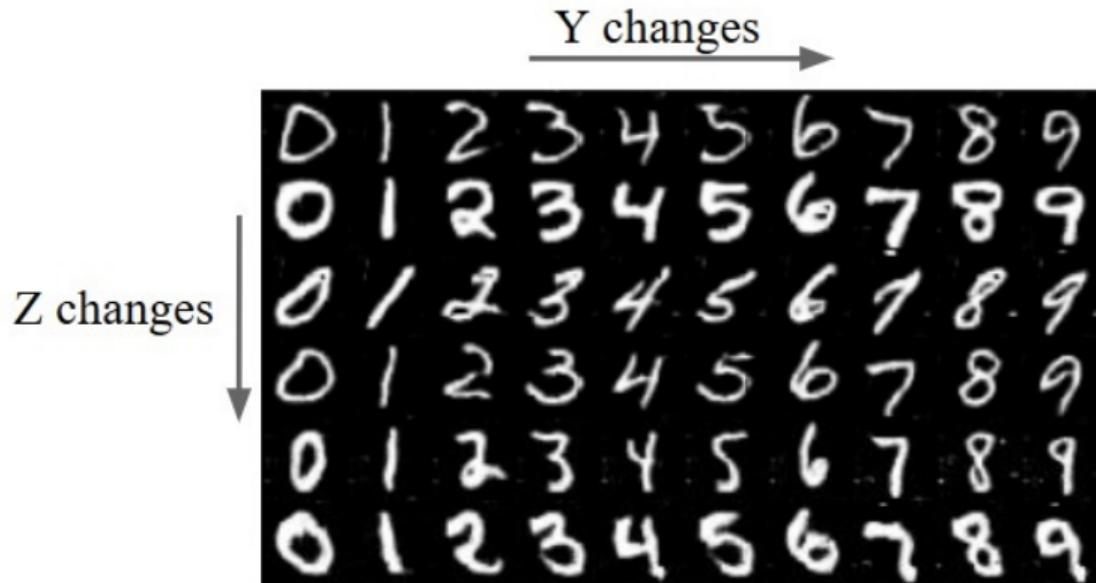


<https://arxiv.org/pdf/1411.1784.pdf>

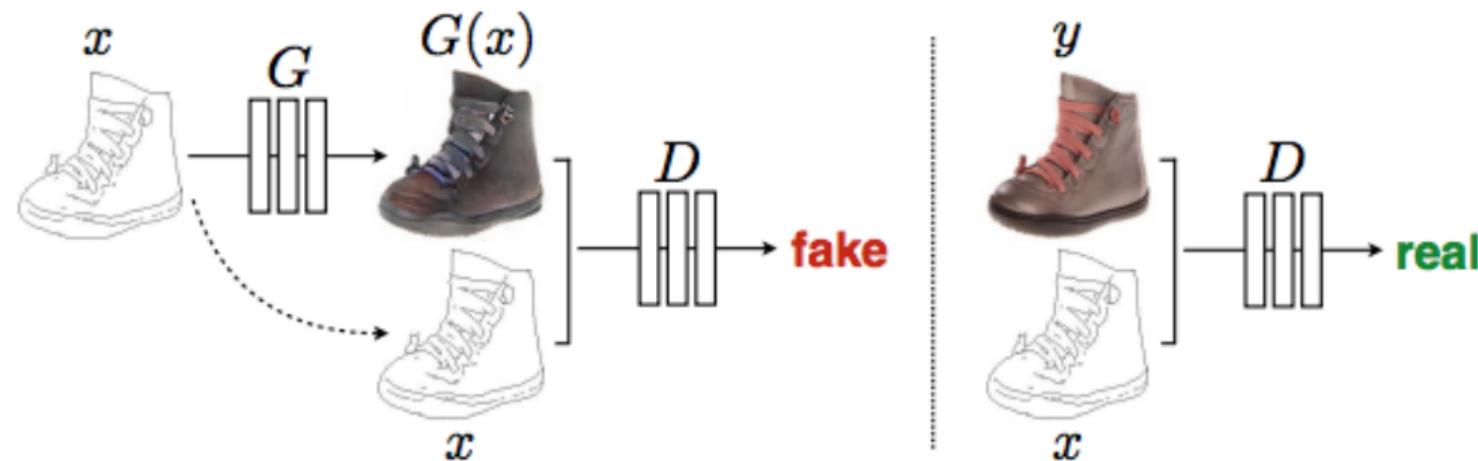
Conditional GAN

- CGAN использует информацию из меток, если они есть
- Это позволяет генерировать не рандомные объекты, а конкретные
- В качестве меток y 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



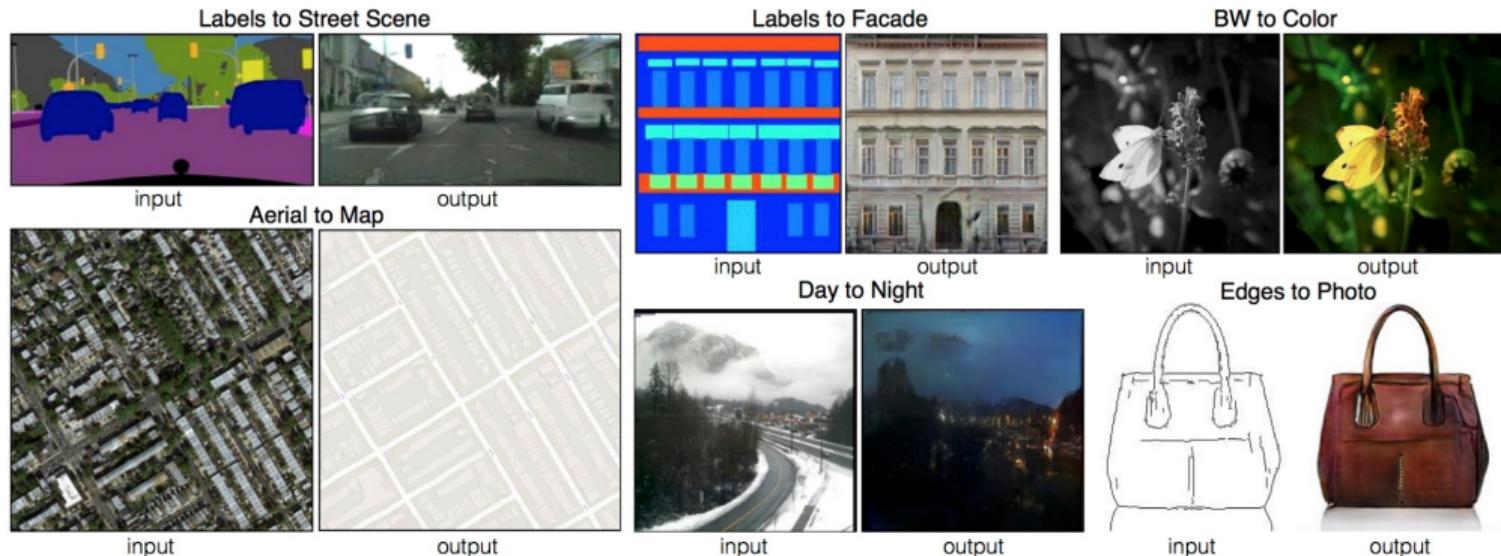
Pix2Pix GAN



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

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

Pix2Pix GAN

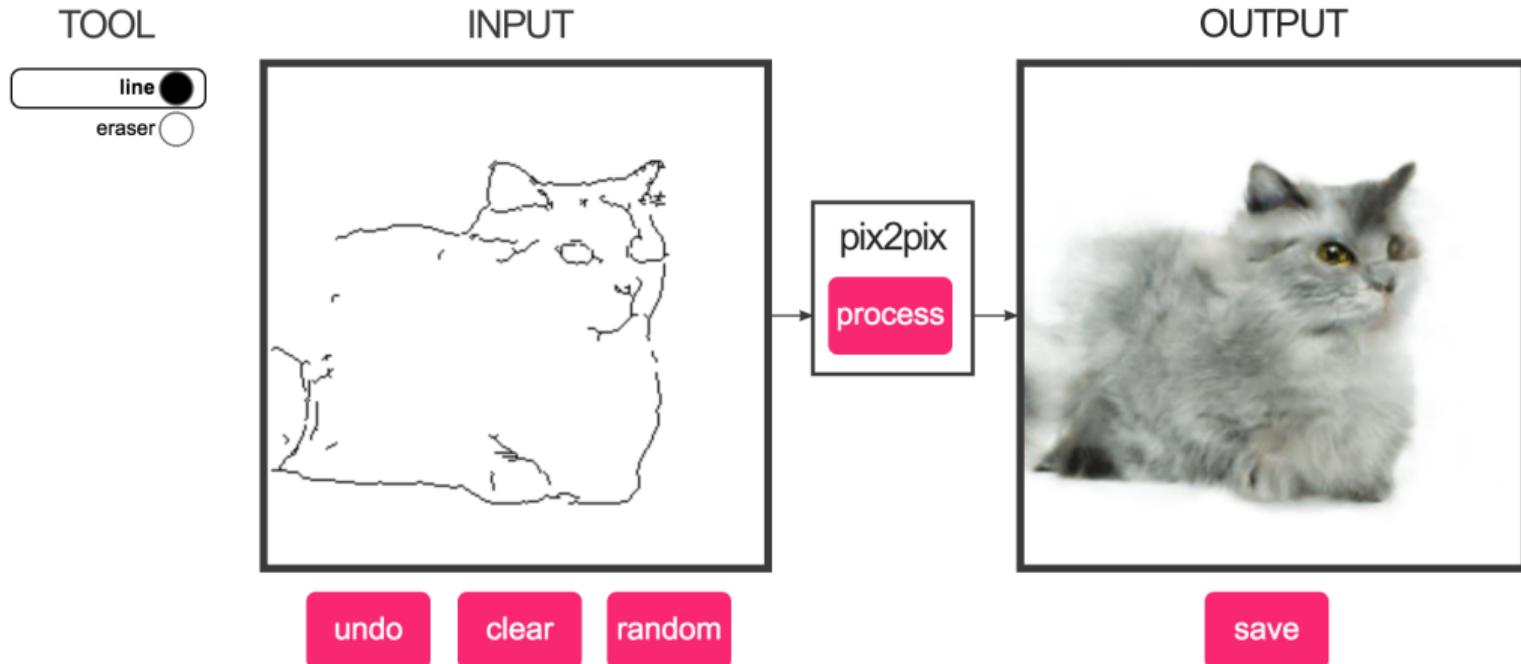


Example results on several image-to-image translation problems. In each case we use the same architecture and objective, simply training on different data.

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

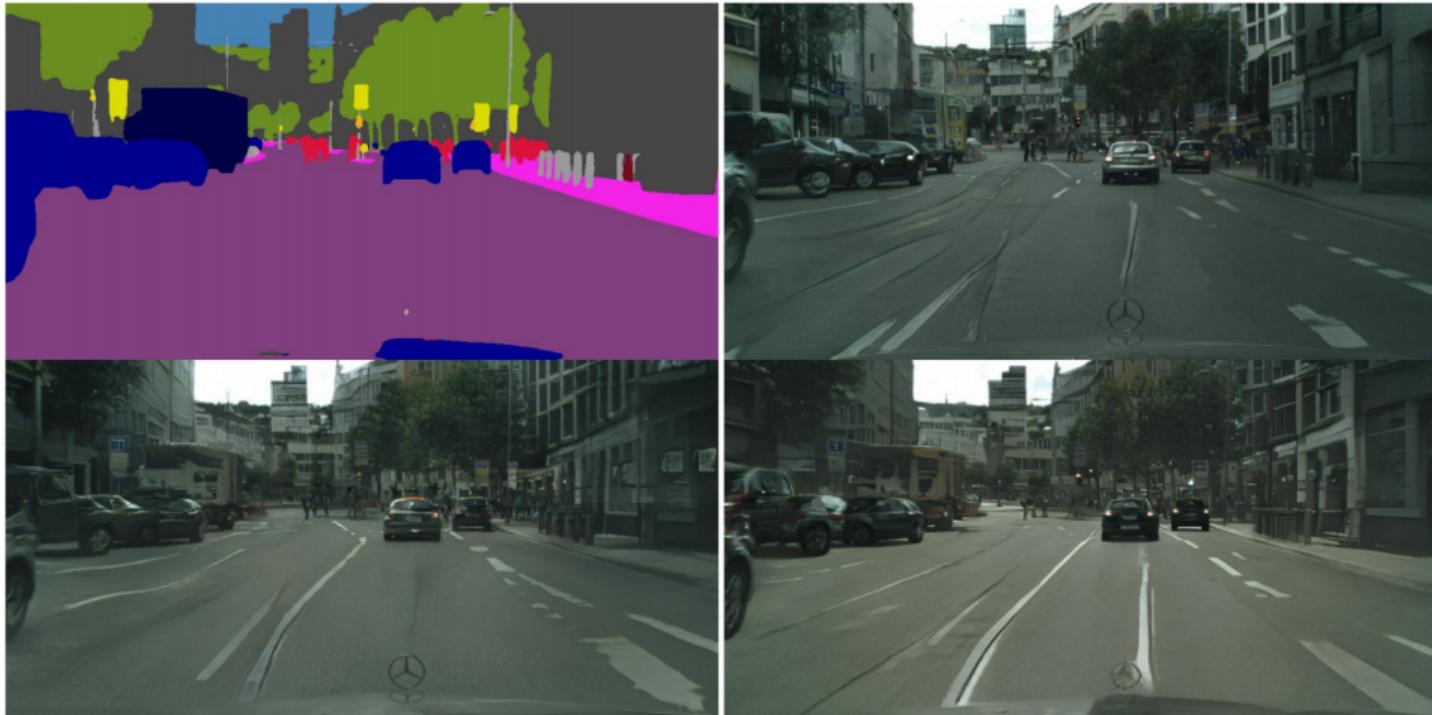
<https://www.tensorflow.org/tutorials/generative/pix2pix>

Pix2Pix GAN



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

Pix2Pix GAN



CycleGAN

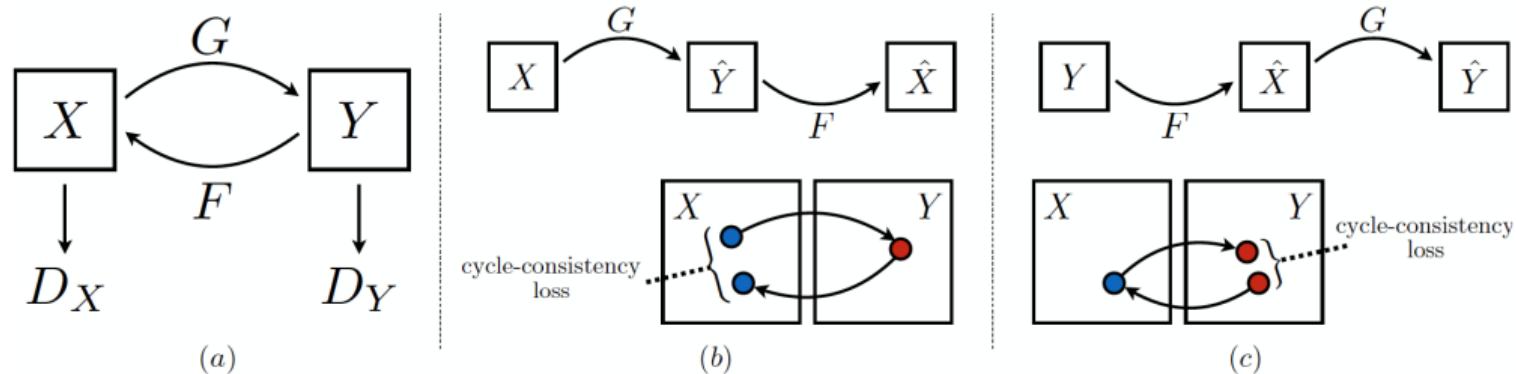


Figure 3: (a) Our model contains two mapping functions $G : X \rightarrow Y$ and $F : Y \rightarrow X$, and associated adversarial discriminators D_Y and D_X . D_Y encourages G to translate X into outputs indistinguishable from domain Y , and vice versa for D_X , F , and X . To further regularize the 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: $y \rightarrow F(y) \rightarrow G(F(y)) \approx y$

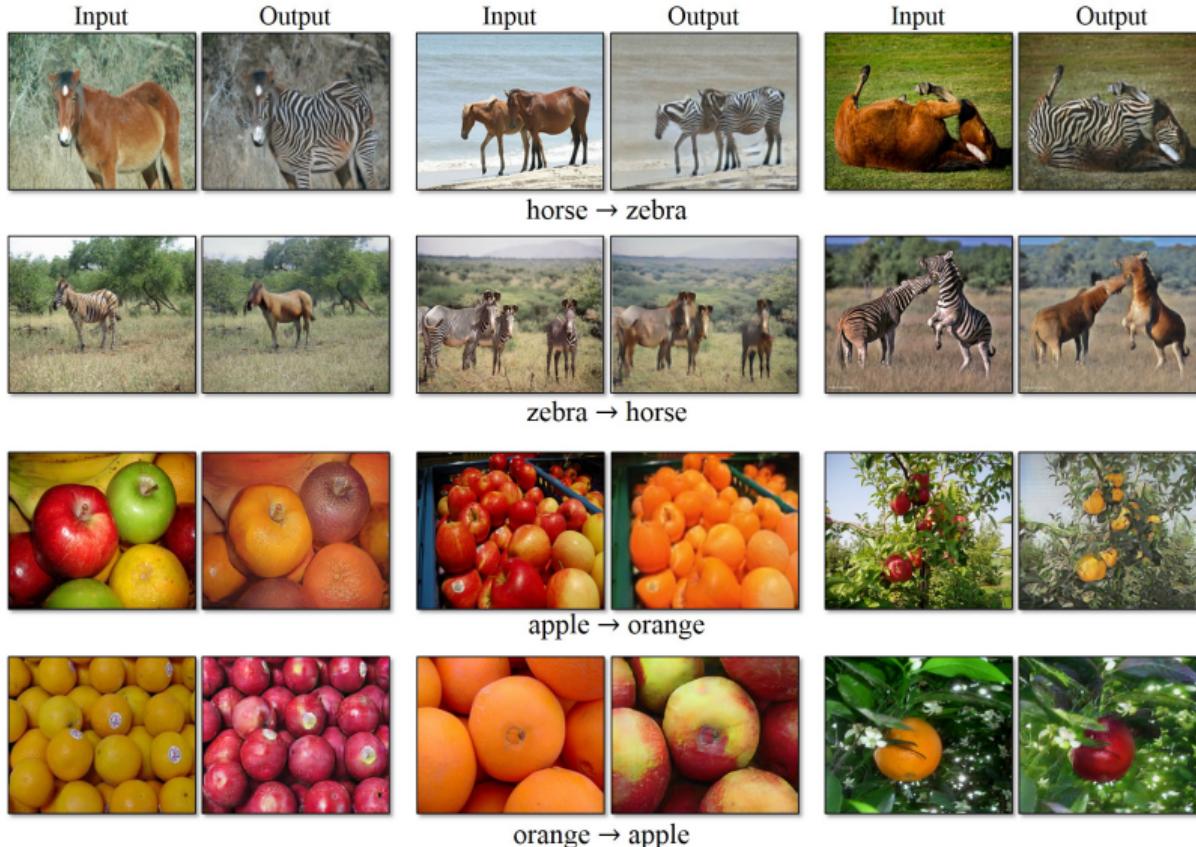
<https://arxiv.org/abs/1703.10593>

<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
- Генератор хорроров от MIT