Московский государственный университет имени М. В. Ломоносова

Механико-математический факультет

Кафедра Математической Теории Интеллектуальных Систем

Курсовая работа

по теме

«Транслирование изображений генеративно-состязательными сетями»

Выполнил
Студент 3 курса, группа 331
Селеев Артём Николаевич

Научные руководители:
Кандидат физико-математических наук
Иванов Илья Евгеньевич
Доктор физико-математических наук
Бабин Дмитрий Николаевич

Москва, 2025

Аннотация

В данной курсовой работе рассматривается модель условной генеративно-состязательной сети Pix2Pix, предназначенной для решения задач транслирования изображений. Рассматриваются теоретические основы GAN и cGAN, архитектура генератора и дискриминатора Pix2Pix, особенности функции потерь. Проведена практическая реализация модели[1] на фреймворке PyTorch и ее обучение на различных датасетах. Особое внимание уделяется расширениям Pix2Pix, включая Pix2PixHD, BicycleGAN и SPADE. Работа демонстрирует широкие возможности Pix2Pix для генерации и трансформации изображений при наличии парных обучающих данных.

[1] github.com/NSArt1/Pix2Pix

Оглавление

Введение **3**

1 Теоретические основы GAN, Conditional GAN, постановка задачи **5**

1.1 Генеративно-состязательные сети (GAN) 5

1.2 Условные генеративно-состязательные сети (cGAN) 6

1.3 Постановка задачи перевода изображений 7

2 Архитектура Pix2Pix **8**

2.1 Функция потерь Pix2Pix 8

2.2 Генератор на основе архитектуры U-Net 9

2.3 Дискриминатор PatchGAN 10

3 Особенности обучения и результаты **11**

3.1 Тренировочные датасеты 11

3.2 Выбор гиперпараметров и оптимизатора 11

3.3 Обучение модели и результаты 12

4 Современные модификации Pix2Pix **14**

4.1 Pix2PixHD 14

4.2 BicycleGAN 15

4.3 SPADE 16

4.4 Self-Attention 17

Заключение **19**

Введение

В последние годы в области глубинного обучения наблюдается стремительное развитие генеративных моделей, позволяющих сэмплировать новые данные на основе изученного распределения обучающих примеров. Особый интерес представляют генеративные модели изображений, которые научились создавать фотореалистичные изображения, сложно отличимые от настоящих. Одним из наиболее успешных подходов к генерации изображений являются генеративно-состязательные сети (Generative Adversarial Networks, GAN). Эти модели продемонстрировали выдающиеся результаты не только в задаче непосредственной генерации фотографий, но и в задачах преобразования изображений из одного вида в другой. Например, с помощью GAN удалось реализовать впечатляющие преобразования «стиль-на-стиль», такие как перевод фотографий лошадей в изображения зебр,повышение разрешения изображений super-resolution, генерация фотографий по картам сегментации или эскизам и т.д. В целом прослеживается тенденция к созданию универсальных методов, способных автоматизировать различные задачи компьютерного зрения, которые ранее решались отдельными специализированными алгоритмами.

На этом фоне актуальной является разработка и исследование моделей, выполняющих перевод изображения в изображение (image-to-image translation) на основе генеративных нейросетей. Одной из первых и наиболее известных таких моделей является Pix2Pix GAN — условный GAN, предлагающий общее решение для широкого класса задач преобразования изображений. Данный подход позволяет обучить нейросеть преобразованию одного типа изображения в другой, используя пары соответствующих тренировочных примеров. Преимущество Pix2Pix состоит в универсальности: вместо проектирования уникальных методов под каждую частную задачу (цветизация, восстановление изображений, перенос стиля и др.) применяется единая архитектура генератора и дискриминатора, обучаемых по специальной составной функции потерь. Благодаря этому подход Pix2Pix получил широкое распространение в задачах компьютерного зрения и графики, а его изучение продолжает оставаться актуальным как с теоретической, так и с практической точки зрения.

Цели

- рассмотреть классическую архитектуру генеративно-состязательной сети (GAN), включая роль генератора и дискриминатора;
- разработать и экспериментально проверить работу модели Pix2Pix на выбраннах задачах.
- рассмотреть современные модификации оригинального Pix2Pix, основыне особенности архитектуры, и их функции потерь

Методы

- анализ научных публикаций и открытых источников по тематике генеративных моделей
- Практическая реализация архитектуры Pix2Pix на Pytorch, ее обучение, и проведение экспериментов с датасетами

Глава 1

Теоретические основы GAN, Conditional GAN, постановка задачи

1.1 Генеративно-состязательные сети (GAN)

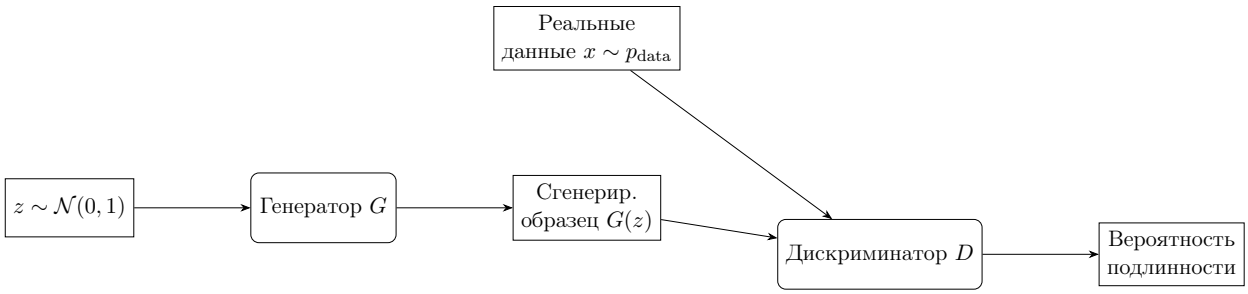

Рис. 1.1: Схема классического GAN

Классический GAN [1] состоит из двух блоков: генератора G и дискриминатора D.

Определение 1. *Генератор $G(z)$ - нейросеть, которая принимает на вход случайный шум $z \sim \mathcal{N}(0, I)$ и генерирует на его основе поддельный образец $x_{fake} = G(z)$.*

Цель генератора – научиться порождать сэмплы, распределение которых $q(x)$ максимально близко к распределению реальных данных $p_{\text{data}}(x)$.

Определение 2. *Дискриминатор $D(x)$ – это нейросеть-классификатор, которая принимает $x \sim X_{data}$, либо $x_{fake} = G(z) \sim X_{fake}$, выдаёт вероятность того, что входной пример принадлежит распределению реальных данных. $D(x) \in [0; 1]$*

Обучение GAN формулируется как процесс минимакс-оптимизации заданного функционала:

$$\mathbb{E}_{x \sim p_{data}(x)} \log D_\phi(x) + \mathbb{E}_{z \sim \mathcal{N}(0,I)} \left[1 - \log D_\phi(G_\theta(z))\right] \to \min_\theta \max_\phi$$

В процессе обучения параметры D и G обновляются поочерёдно: на каждом шаге сначала оптимизируются веса D (при фиксированном G) для повышения качества различения, затем – веса G (при фиксированном D) для уменьшения шанса обнаружения

генератора

Лемма 1. *Оптимальный дискриминатор D_{ϕ^*} при фиксированном генераторе принимает вид:*

$$D_{\phi^*} = \frac{p(x)}{p(x) + q(x)}$$

Теорема 1. *Пусть:*

$$\mathcal{D}_\theta = \max_\phi \mathcal{L}_{\theta,\phi} = \mathbb{E}_{x\sim p(x)} \left[\log D_{\phi^*}(x)\right] + \mathbb{E}_{x\sim q(x)} \log \left[1 - D_{\phi^*}(x)\right]$$

Тогда $\mathcal{D}_\theta \to \min_\theta \iff JSD(p \,\|q) \to \min_{q(x)}$

Таким образом, заданный функционал минимизирует дивергенцию Йенсена-Шенона, следовательно приближает распределение генератора к распределению исходных данных, а идеальное решение данной задачи является когда распределние генератора полностью совпадает с исходным распределением

1.2 Условные генеративно-состязательные сети (cGAN)

Классическая генеративно-состязательная сеть (GAN) моделирует безусловное распределение данных $p(y)$. Однако во многих задачах требуется моделирование *условного распределения* $p(y \mid x)$, где $x \in X$ — наблюдаемая переменная (условие). В этом случае применяется условный GAN (cGAN[2]), в котором и генератор G, и дискриминатор D зависят от условия x.

Генератор реализует отображение:

$$G : X \times Z \to Y, \quad G(x, z) \mapsto \hat{y},$$

где $z \sim p(z)$ — латентный шум. Дискриминатор получает пару (x, y) и вычисляет вероятность принадлежности y истинному условному распределению $p(y \mid x)$.

Функция потерь cGAN имеет вид:

$$\mathcal{L}_{\text{cGAN}}(G, D) = \mathbb{E}_{x,y\sim p_{\text{data}}(x,y)} \left[\log D(x, y)\right] + \mathbb{E}_{x\sim p(x), z\sim p(z)} \left[\log \left(1 - D(x, G(x, z))\right)\right]. \quad (1.1)$$

Оптимизация модели формулируется как задача минимакс:

$$G^* = \arg\min_G \max_D \; \mathcal{L}_{\text{cGAN}}(G, D).$$

Условие x может представлять собой категориальные метки, векторы признаков или изображения. Модель cGAN позволяет решать задачи условной генерации, в том числе image-to-image translation, управление стилем и синтез по описанию.

Таким образом, условные GAN обобщают стандартную модель GAN и позволяют решать широкий класс задач, включая генерацию по условию, преобразование изображений, управление стилем и содержанием.

1.3 Постановка задачи перевода изображений

Задача *image-to-image translation* формализуется как обучение отображения:

$$F : X \to Y$$

где X, Y — домены изображений, (x_i, y_i) элементы обучающей выборки, где y_i — желаемое преобразование x_i.
Цель — построить отображение $\hat{F}$ такое, что $\hat{F}(x) \approx y$ для новых примеров. К данной постановке сводятся многие задачи компьютерного зрения:

- колоризация изображений: x — ч/б фото, y — цветное;
- генерация по сегментации: x — карта классов, y — фотореалистичная сцена;
- структурный перевод: x — эскиз, y — изображение объекта;
- картографическая трансформация: x — спутниковый снимок, y — карта.

Ранее такие задачи решались специализированными методами. Однако условные генеративно-состязательные сети (cGAN) позволяют обучать универсальное отображение F, согласующее x и y в рамках единой архитектуры. Примером такой модели является Pix2Pix, рассмотренный в следующей главе.

Глава 2

Архитектура Pix2Pix

Модель Pix2Pix[3] состоит из двух основных компонентов: генератора, выполняющего преобразование изображения, и дискриминатора, оценивающего качество этого преобразования. На уровне архитектуры Pix2Pix представляет собой условный GAN, в котором генератор использует *U-Net*[4] подобную архитектуру, а дискриминатор – *PatchGAN*[5]-классификатор. Обучение осуществляется с использованием комбинированной функции потерь, состоящей из $\mathcal{L}$ и L_1 потерь.

2.1 Функция потерь Pix2Pix

Целевой функционал условной генеративно-состязательной сети (conditional GAN, cGAN) задаётся следующим образом:

$$\mathcal{L}_{\text{cGAN}}(G, D) = \mathbb{E}_{x,y}\left[\log D(x, y)\right] + \mathbb{E}_{x,z}\left[\log\left(1 - D(x, G(x, z))\right)\right], \tag{2.1}$$

где G — генератор, D — дискриминатор, x — входное изображение, y — целевое изображение, z — случайный шум. Задача обучения формулируется как:

$$G^* = \arg\min_G \max_D \mathcal{L}_{\text{cGAN}}(G, D). \tag{2.2}$$

Как показано в [3], полезно дополнять адверсариальную функцию традиционным критерием ошибки между $G(x, z)$ и y. В частности, используется L_1-ошибка:

$$\mathcal{L}_{L_1}(G) = \mathbb{E}_{x,y,z}\left[\|y - G(x, z)\|_1\right], \tag{2.3}$$

так как она, в отличие от L_2, способствует сохранению резких границ и уменьшает размытость изображений.

Итоговая функция потерь генератора принимает вид:

$$G^* = \arg\min_G \max_D \left(\mathcal{L}_{\text{cGAN}}(G, D) + \lambda \mathcal{L}_{L_1}(G)\right), \tag{2.4}$$

где $\lambda > 0$ — гиперпараметр, определяющий баланс между реализмом изображения и его близостью к целевому.

Хотя в теории ввод случайного вектора z должен обеспечивать стохастичность, на практике генератор часто игнорирует его. Поэтому в Pix2Pix стохастичность вводится через механизм dropout, применяемый в нескольких слоях генератора как во время обучения, так и на стадии вывода. Несмотря на это, выход генератора остаётся практически детерминированным. Это указывает на ограниченность текущей модели в аппроксимации полной энтропии условного распределения $p(y \mid x)$.

2.2 Генератор на основе архитектуры U-Net

Генератор G в модели Pix2Pix реализован на основе архитектуры U-Net [4], изначально предложенной для задач биомедицинской сегментации. Эта архитектура представляет собой симметричную сеть энкодер-декодерного типа с пропускными соединениями (skip-connections) между слоями одинаковой глубины в энкодере и декодере.

Пусть входное изображение $x \in \mathbb{R}^{H \times W \times C}$ проходит через энкодер, состоящий из n сверточных блоков, каждый из которых уменьшает пространственное разрешение и увеличивает количество каналов признаков. Энкодер формирует сжатое представление (bottleneck), содержащее глобальную информацию о входном изображении. Далее декодер восстанавливает исходное разрешение, применяя транспонированные свёртки (deconvolutions).

Для сохранения мелких деталей и локальной информации, между слоями i в энкодере и слоями $n - i$ в декодере вводятся пропускные соединения, реализуемые через конкатенацию:

$$z^{(n-i)} = \text{Concat}(z^{(n-i)}, z^{(i)}), \tag{2.5}$$

где $z^{(i)}$ — выход i-го слоя энкодера, а $z^{(n-i)}$ — вход соответствующего слоя декодера. Эти соединения позволяют передавать низкоуровневые признаки напрямую, минуя узкое «бутылочное горлышко».

В архитектуре генератора используются следующие компоненты:

- **Сверточные слои** с ядрами 4×4 и шагом 2;
- **Функции активации**: Leaky ReLU в энкодере, ReLU в декодере;
- **Batch Normalization** после большинства свёрток (кроме первого слоя);
- **Dropout** в некоторых слоях декодера для регуляризации и повышения вариативности генерации;
- **Финальный слой**: сверточный слой с активацией tanh, преобразующий выход к диапазону $[-1, 1]$.

Архитектура U-Net обеспечивает эффективную передачу информации от входного изображения к выходному, позволяя генератору сохранять как общую структуру объектов, так и мелкие детали, что особенно важно в задачах преобразования изображений с точной пиксельной привязкой.

2.3 Дискриминатор PatchGAN

В модели Pix2Pix применяется дискриминатор D архитектуры *PatchGAN*[5], направленный на оценку реалистичности изображения на уровне локальных участков. В отличие от стандартного дискриминатора, возвращающего скаляр $D(x) \in [0, 1]$, PatchGAN моделирует распределение вероятностей $D(x, y) \in [0, 1]^{m \times m}$, где каждому элементу соответствует рецептивное поле фиксированного размера $N \times N$.

Формально, дискриминатор применяет свёрточную нейросеть D к объединению по каналам входного изображения x и реального или сгенерированного изображения y, и классифицирует каждый патч как подлинный или поддельный:

$$D : (x, y) \mapsto \{D_{i,j}(x, y)\}_{i,j=1}^{m},$$

где $m \times m$ — размеры выходной карты достоверности.

В оригинальной реализации используется 70-PatchGAN: рецептивное поле каждого выхода охватывает область 70×70 пикселей входного изображения. Усреднение по выходной карте:

$$\bar{D}(x, y) = \frac{1}{m^2} \sum_{i,j} D_{i,j}(x, y)$$

может быть использовано как итоговая мера подлинности.

Глава 3

Особенности обучения и результаты

3.1 Тренировочные датасеты

Чтобы исследовать обобщаю способность исследуемой архитектуры, были проведены эксперименты на следующих парных датасетах:

- Map $\leftrightarrow$ Aerial - состоит из 1096 тренировочных изображений карт участков города и их спутниковых снимков
- Labels $\rightarrow$ Facades - 400 изображений карт сегмантаций фасадов и реальных изображений фасадов
- Edges $\rightarrow$ Bags - 6000 изображений контуров сумок и реальных изображений
- Edges $\rightarrow$ Shoes - 10000 изображений контуров обуви и реальных изображений

3.2 Выбор гиперпараметров и оптимизатора

Обучения модели проходило с использованием Adam оптимизатора с парматерами $\alpha = 2 \cdot 10^{-4}$, $\beta_1 = 0.5$, $\beta_2 = 0.999$. Adam обеспечивает быструю и стабильную сходимость даже при высокоразмерных параметрах сети.
Основные гиперпараметры модели включают:

- λ — весовой коэффициент при L_1-регуляризаторе в функции потерь;
- размер батча: 1 или 16 (в зависимости от объёма датасета);
- количество эпох: от 15 до 200 в зависимости от объема исходного датасета;
- Глубина U-net генератора: в реализуемой нами модели была выбрана глубина 5

Рис. 3.1: Пример работы Pix2Pix

3.3 Обучение модели и результаты

Обучение проводилось на GPU с использованием PyTorch. Потери $L1$ и L_{adv} постепенно уменьшались, а визуальные результаты подтверждали генерацию изображений, приближённых к ground truth по структуре и стилю. Результаты модели можно проиллюстрировать примерами на приведенных рисунках.
Для оценки качества сгенерированных изображений используется PSNR-метрика:

$$\text{PSNR} = 10 \cdot \log_{10} \left(\frac{L^2}{\text{MSE}} \right), \tag{3.1}$$

где L - макисмальное значение пикселя (в нашем случае L = 2 для диапазона [-1,1])

Датасет	**Значение PSNR**	**Значение LPIPS**
aerial→ maps	25.3	0.19
maps → aerial	16.7	0.23
edges → bags	16.24	0.31
edges → shoes	18,6	0.33
cityscapes → segments	20,9	0.21

Таблица 3.1: Результаты PSNR и LPIPS на изучаемых датасетах

Также будем пользоваться LPIPS[6] метрикой:

$$\mathrm{LPIPS}(x, y) = \sum_{l} \frac{1}{H_l W_l} \sum_{h=1}^{H_l} \sum_{w=1}^{W_l} \left\| \hat{f}_l^x(h, w) - \hat{f}_l^y(h, w) \right\|_2^2, \tag{3.2}$$

где:

- $x, y \in \mathbb{R}^{3 \times H \times W}$ — сравниваемые изображения, нормализованные в диапазоне $[-1, 1]$;
- $f_l^x, f_l^y \in \mathbb{R}^{C_l \times H_l \times W_l}$ — активации слоя l предобученной сверточной сети при подаче изображений x и y;
- $\hat{f}_l = f_l / \|f_l\|$ — активации, нормализованные по каналам (channel-wise unit norm);
- H_l, W_l — пространственные размеры карты признаков на слое l (VGG-19);

Рис. 3.2: Пример Pix2Pix

Глава 4

Современные модификации Pix2Pix

4.1 Pix2PixHD

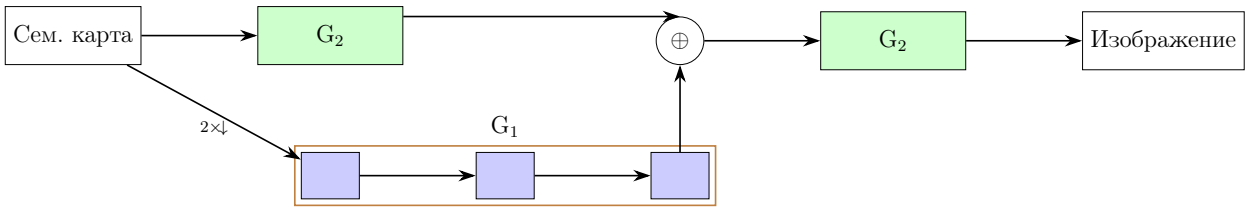

Рис. 4.1: Схема каскадного генератора Pix2PixHD

Pix2PixHD[7] – расширение Pix2Pix для генерации изображений высокого разрешения (например 2048×1024). Основные усовершенствования Pix2PixHD по сравнению с оригинальной Pix2Pix следующие:

- Каскадный генератор:

$$G_1 : X \times Z \to Y_L, \quad G_2 : Y_L \to Y_H,$$

  где Y_L и Y_H — низкое и высокое разрешение.

- Мультимасштабные дискриминаторы $\{D_k\}$ на масштабах k:

$$\mathcal{L}_{\text{cGAN}} = \sum_k \mathbb{E}_{x,y}\big[\log D_k(x,y)\big] + \mathbb{E}_{x,z}\big[\log(1 - D_k(x, G(x,z)))\big].$$

- Feature matching loss:

$$\mathcal{L}_{\text{FM}} = \sum_k^K \sum_l^{L_k} \left\| D_k^{(l_k)}(y) - D_k^{(l_k)}(G(x,z)) \right\|_1.$$

  где k - масштаб, l_k - слой D_k

4.2 BicycleGAN

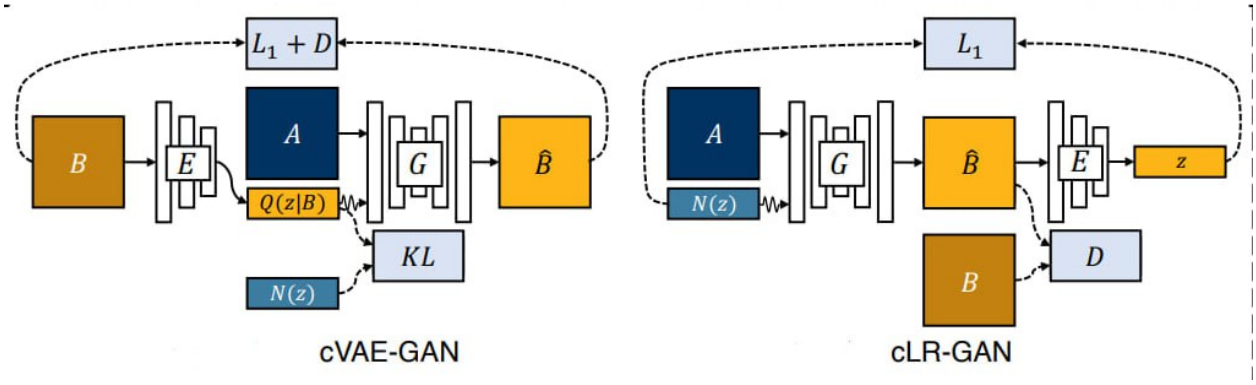

Рис. 4.2: cVAE-GAN, cLR-GAN

BicycleGAN[8] направлен на решение проблемы неоднозначности в задачах преобразования изображений, где одному входному изображению может соответствовать множество реалистичных выходов. Модель использует два типа согласованности: между изображениями и между латентным пространством и выходом, что позволяет генерировать разнообразные и реалистичные результаты. Это достигается путем объединения условного вариационного автоэнкодера (cVAE) и условного латентного регрессора (cLR) в единую архитектуру.

cVAE-GAN ($B \to z \to \hat{B}$):

Целью cVAE-GAN оптимизации является:

$$G^*, E^* = \arg\min_{G,E} \max_{D} \; \mathcal{L}_{\text{GAN}}^{\text{VAE}}(G, D, E) \; + \; \lambda \, \mathcal{L}_1^{\text{VAE}}(G, E) \; + \; \lambda_{\text{KL}} \, \mathcal{L}_{\text{KL}}(E),$$

где:

$$\mathcal{L}_{\text{GAN}}^{\text{VAE}}(G, D, E) = \mathbb{E}_{A,B\sim p(A,B)}\big[\log D(A, B)\big] + \mathbb{E}_{A,B\sim p(A,B),\, z\sim E(B)}\big[\log\big(1 - D\big(A, G(A, z)\big)\big)\big]$$

$$\mathcal{L}_{\text{KL}}(E) = \mathbb{E}_{B\sim p(B)}\big[D_{\text{KL}}\big(E(B) \,\big\|\, \mathcal{N}(0, I)\big)\big].$$

cLR-GAN ($z \to \hat{B} \to \hat{z}$)

Целью cLR оптимизации является:

$$G^*, E^* = \arg\min_{G,E} \max_{D} \; \mathcal{L}_{\text{GAN}}(G, D) \; + \; \lambda_{\text{latent}} \, \mathcal{L}_1^{\text{latent}}(G, E)$$

где:

$$\mathcal{L}_1^{\text{latent}}(G, E) = \mathbb{E}_{A\sim p(A),\, z\sim p(z)}\big\| z \; - \; E\big(G(A, z)\big)\big\|_1$$

BicycleGAN: В BicycleGAN объединяют лоссы обоих моделей:

$$G^*, E^* = \arg\min_{G,E} \max_{D} \mathcal{L}_{\text{GAN}}^{\text{VAE}}(G, D, E) + \lambda \mathcal{L}_1^{\text{VAE}}(G, E) + \mathcal{L}_{\text{GAN}}(G, D) + \lambda_{\text{l}} \mathcal{L}_1^{\text{l}}(G, E) + \lambda_{\text{KL}} \mathcal{L}_{\text{KL}}(E)$$

Таким образом VAE-цикл не дает G игнорировать z: если генератор не слушал шум, кодировщик E не сможет восстановить тот же z. LR - цикл заставляет точно воспроизводить y, когда z получен из y.
Вместе оба цикла создают биективное отображение $z \longleftrightarrow y \ \forall x \in A$

4.3 SPADE

Одним из ограничений оригинального Pix2Pix, было, то что при генерации по семантической разметке детали могли размываться из-за глобальной нормализации признаков. Метод SPADE(Spattialy-adaptive normalization)[9] направлен на решение данной проблемы. Ключевая идея - подача карты сегментации на каждый слой.

Для активаций $h^i \in \mathbb{R}^{C^i \times H^i \times W^i}$ и семантической карты m вводятся пространственно-адаптивные параметры:

$$h^i_{n,c,y,x,\text{out}} = \gamma^i_{c,y,x}(m) \, \frac{h^i_{n,c,y,x} - \mu^i_c}{\sigma^i_c} \ + \ \beta^i_{c,y,x}(m) \tag{4.1}$$

где γ, β получаются свёртками по m.

$$\mu^i_c = \frac{1}{N \, H^i \, W^i} \sum_{n=1}^{N} \sum_{y=1}^{H^i} \sum_{x=1}^{W^i} h^i_{n,c,y,x} \tag{4.2}$$

$$\sigma^i_c = \sqrt{\frac{1}{N \, H^i \, W^i} \sum_{n=1}^{N} \sum_{y=1}^{H^i} \sum_{x=1}^{W^i} \left(h^i_{n,c,y,x} - \mu^i_c \right)^2} \tag{4.3}$$

Следует отметить, что SPADE - не отдельная задача, а модуль встраиваемый в архитектуру генератора. Его можно комбинировать с другими улучшениями.Авторы SPADE в своей реализации использовали мультимасштабные дискриминаторы и Feature matching loss.

4.4 Self-Attention

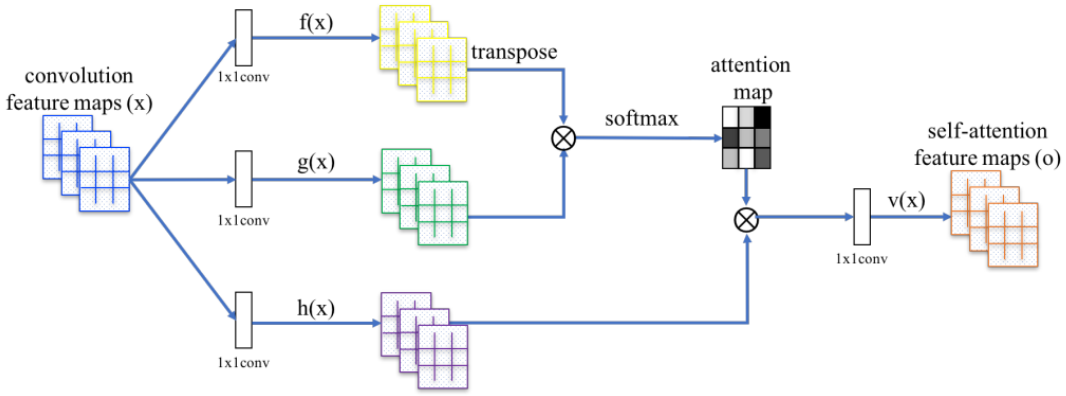

Рис. 4.3: self-attention GAN

Важное направление улучшений GAN – внедрение механизмов само-внимания (self-attention[10]) для учёта дальних связей в изображении, как в генераторе так и в дискриминаторе. Классическая свёрточная сеть ограничена локальным рецептивным полем: каждый пиксель выходного изображения зависит в основном от небольшого соседства входа. Это затрудняет генератору моделирование глобальных зависимостей.

Пусть на входе слоя лежат активации

$$F \in \mathbb{R}^{C\times H\times W},$$

где C — число каналов, $H \times W = N$ — число позиций. Развёртываем их в матрицу

$$F = \left[f_1, f_2, \ldots, f_N\right] \in \mathbb{R}^{C\times N}.$$

1. Запросы, ключи и значения

Три линейных проекции (1×1-свёртки) порождают матрицы

$$Q = W_q\, F, \quad K = W_k\, F, \quad V = W_v\, F,$$

где $W_q, W_k, W_v \in \mathbb{R}^{d\times C}$, а $Q, K, V \in \mathbb{R}^{d\times N}$ — «запросы», «ключи» и «значения».

2. Вычисление внимания

Для каждой пары позиций i, j вычисляем скалярное произведение

$$s_{ij} = q_i^{\mathsf{T}} k_j,$$

и масштабируем:

$$\tilde{s}_{ij} = \frac{s_{ij}}{\sqrt{d}}.$$

Нормируем по j с помощью softmax:

$$a_{ij} = \frac{\exp(\tilde{s}_{ij})}{\sum_{j'=1}^{N} \exp(\tilde{s}_{ij'})}, \quad A = [a_{ij}] \ \in [0,1]^{N\times N}.$$

3. Агрегация значений

Собираем новую карту признаков как взвешенную сумму значений:

$$o_i = \sum_{j=1}^{N} a_{ij}\, v_j, \quad O = [o_1, \ldots, o_N] = V\, A^{\mathsf{T}} \ \in \ \mathbb{R}^{d\times N}.$$

4. Проекция и residual-связь

Проецируем обратно в C каналов и добавляем исходный F:

$$F' = W_o\, O \ + \ F, \quad W_o \in \mathbb{R}^{C\times d}.$$

—

Таким образом self-attention обеспечивает:

- сочетание глобального контекста через матрицу A
- *локального сохранения* через остаточную связь $+F$, что позволяет GAN-сетям учитывать дальние зависимости и синхронизировать детали по всему изображению.

Заключение

В данной работе рассмотрена модель Pix2Pix и её основные расширения (Pix2PixHD, BicycleGAN, SPADE и self-attention GAN). Показано, как условные генеративно-состязательные сети позволяют решать широкий класс задач преобразования изображений при наличии парных данных, а предложенные архитектурные модификации и дополнительные потери (feature matching, циклические и латентные потери, адаптивная нормализация) обеспечивают высокое качество, фотореализм и разнообразие генерируемых изображений. Практическая реализация на базе PyTorch продемонстрировала применение данной модели к разным наборам данных: фасады, спутниковые карты, эскизы и сегментационные маски. Анализ метрик PSNR, LPIPS подтвердил эффективность предложенных подхода.

В перспективе дальнейшие исследования могут быть направлены на:

- интеграцию механизмов self-attention с диффузионными моделями для объединения преимуществ обоих подходов;
- разработку более стабильных схем генерации мультимодальных результатов без потери качества;
- применение предобученных энкодеров (например, CLIP) для семантического контроля и повышения гибкости генерации;
- оптимизацию архитектуры для работы в реальном времени и на устройствах с ограниченными ресурсами.

Литература

[1] Goodfellow I., Pouget-Abadie J., Mirza M., Xu B., Warde-Farley D., Ozair S., Courville A., Bengio Y. Generative Adversarial Networks

[2] Mirza M., Osindero S. Conditional Generative Adversarial Nets

[3] Isola P., Zhu J.-Y., Zhou T., Efros A. A. Image-to-Image Translation with Conditional Adversarial Networks

[4] Ronneberger O., Fischer P., Brox T. U-Net: Convolutional Networks for Biomedical Image Segmentation

[5] C. Li and M. Wand. Precomputed real-time texture synthesis with markovian generative adversarial networks

[6] Zhang R., Isola P., Efros A.A. et al. The Unreasonable Effectiveness of Deep Features as a Perceptual Metric

[7] Wang T.-C., Liu M.-Y., Zhu J.-Y., et al. High-Resolution Image Synthesis and Semantic Manipulation with Conditional GANs

[8] Juan-Yan Zhu, Richard Zhang, Deepak Pathak, Trevor Darell. Toward Multimodal Image-to-Image Translation

[9] Taesung Park, Ming-Yu Liu, Ting-Chun Wang, Jun-Yan Zhu Semantic Image Synthesis with Spatially-Adaptive Normalization

[10] Zhang H., Goodfellow I., Metaxas D., Odena A. Self-Attention Generative Adversarial Networks