МИНОБРНАУКИ РОССИИ
Федеральное государственное бюджетное образовательное учреждение
высшего образования
«САРАТОВСКИЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ ИМЕНИ Н.Г. ЧЕРНЫШЕВСКОГО»

Кафедра теоретических основ компьютерной безопасности и криптографии

Генеративно-состязательные нейронные сети

РЕФЕРАТ

студента 5 курса 531 группы
специальности 10.05.01 Компьютерная безопасность
факультета компьютерных наук и информационных технологий
Цыпина Андрея Алексеевича

Старший преподаватель
д.ф.-м.н., доцент _____ И. И. Слеповичев
подпись, дата

Саратов 2024

СОДЕРЖАНИЕ

ВВЕДЕНИЕ

В последние годы развитие искусственного интеллекта и методов глубокого обучения идет невероятными темпами, открывая перед исследователями и разработчиками новые возможности в обработке данных и решении сложных задач. Одним из ключевых прорывов в этой области стало создание генеративно-состязательных нейронных сетей (Generative Adversarial Networks, GAN), впервые представленных Иэном Гудфеллоу и его коллегами в 2014 году. Эти сети предложили принципиально новый подход к генерации данных, который позволяет моделировать процессы творчества и обучения, делая синтетические данные практически неотличимыми от реальных.

Сегодня GAN находят применение во множестве сфер, от обработки изображений и видео до создания текстов и моделирования медицинских данных. Например, с помощью GAN можно значительно улучшать качество фотографий, создавать изображения на основе текстовых описаний, генерировать синтетические медицинские данные для обучения алгоритмов диагностики и даже создавать реалистичные подделки человеческих лиц.

Однако использование GAN сопряжено и с определенными трудностями. Тем не менее, разработка новых архитектур, таких как DCGAN (Deep Convolutional GAN), WGAN (Wasserstein GAN), StyleGAN и их различных модификаций, помогает решать многие из этих вопросов.

Цель данной работы — всестороннее изучение генеративно-состязательных нейронных сетей, включая их архитектуру, принципы функционирования и примеры практического применения.

1 Основные понятия и история возникновения GAN

Генеративно-состязательные нейронные сети — это современный метод машинного обучения, который используется для генерации данных, максимально похожих на реальные.

Главной особенностью GAN является соревновательный процесс между двумя нейронными сетями:

- Генератор — сеть, которая создает новые данные на основе случайного шума.
- Дискриминатор — сеть, которая оценивает данные и пытается определить, являются ли они реальными (из обучающей выборки) или сгенерированными.

Принцип работы GAN можно сравнить с игрой между фальшивомонетчиком и экспертом:

Генератор (фальшивомонетчик) стремится создавать данные, которые будут неотличимы от реальных, а дискриминатор (эксперт) старается распознать подделки и отличить их от настоящих данных.

В процессе обучения обе сети соревнуются и совершенствуют свои способности: генератор учится создавать всё более реалистичные данные, а дискриминатор — всё точнее их различать. Эта соревновательная природа и дала название этому классу нейронных сетей — «состязательные».

Отцом генеративно-состязательных сетей (GAN) считается Ян Гудфеллоу, который разработал эту идею в 2014 году, когда обсуждал с друзьями способы обучения компьютеров рисованию. Вместо написания специальных алгоритмов он предложил использовать две нейросети, соревнующиеся друг с другом. Эта идея пришла к нему поздно ночью после визита в бар, где коллега отмечал защиту диплома.

Тем не менее, Гудфеллоу не был первым, кто высказал подобные идеи. В 1991 году Юрген Шмидхубер опубликовал концепцию генеративных и

состязательных нейросетей, которые конкурируют в игре с нулевой суммой. Одна сеть моделирует вероятностное распределение, а другая предсказывает реакции на результаты. Это взаимодействие назвали «искусственным любопытством». Также Олли Ньемитало в 2010 году предложил создать состязательные сети в своей работе, однако не реализовал эту идею [1].

2 Архитектура и модификации генеративно-состязательных сетей

2.1 Базовая архитектура GAN

Генеративно-состязательная сеть (GAN) состоит из двух основных частей: генератора и дискриминатора.

Ключевым элементом, ответственным за создание свежих и точных данных в генеративной состязательной сети (GAN), является модель генератора. Генератор принимает случайный шум в качестве входных данных и преобразует его в сложные выборки данных, такие как текст или изображения. Обычно это изображается как глубокая нейронная сеть.

Распределение исходных данных для обучения фиксируется слоями обучаемых параметров в процессе обучения. Генератор корректирует свои выходные данные, чтобы создавать образцы, максимально приближенные к реальным данным, по мере обучения с помощью обратного распространения ошибки для точной настройки параметров.

Успех генератора зависит от его способности генерировать высококачественные, разнообразные образцы, которые могут обмануть дискриминатор.

Задача генератора в GAN состоит в том, чтобы создавать синтетические образцы, достаточно реалистичные, чтобы обмануть дискриминатор. Генератор достигает этого, минимизируя свою функцию потерь J_G. Потери минимизируются, когда максимизируется логарифмическая вероятность, то есть, когда дискриминатор с высокой вероятностью классифицирует сгенерированные образцы как реальные. Ниже приведено следующее уравнение:

$$J_G = -\frac{1}{m}\sum_{i=1}^{m} logD(G(z_i)),$$

где $logD(G(z_i))$ представляет собой логарифмическую вероятность того, что дискриминатор будет правильно работать с сгенерированными образцами. Генератор стремится минимизировать эти потери, поощряя создание образцов, которые дискриминатор классифицирует как реальные ($logD(G(z_i))$ близкое к

1). Таким образом, J_G показывает, насколько хорошо генератор обманывает дискриминатор.

Дискриминатор используется в GAN для различения сгенерированных и фактических входных данных. Оценивая входные выборки и распределяя вероятность подлинности, дискриминатор функционирует как двоичный классификатор.

Со временем дискриминатор учится отличать реальные данные из набора данных от искусственных образцов, созданных генератором. Это позволяет ему постепенно совершенствовать свои параметры и повышать уровень мастерства.

Целью процедуры состязательного обучения является максимизация способности дискриминатора точно определять сгенерированные образцы как поддельные, а реальные образцы — как настоящие. В результате взаимодействия генератора и дискриминатора дискриминатор становится всё более точным, что помогает генеративно-состязательной сети генерировать чрезвычайно реалистичные синтетические данные в целом.

Дискриминатор снижает отрицательную логарифмическую вероятность правильной классификации как искусственных, так и реальных образцов. Эта потеря побуждает дискриминатор точно классифицировать сгенерированные образцы как поддельные и реальные с помощью следующего уравнения:

$$J_D = -\frac{1}{m}\sum_{i=1}^{m} logD(x_i) - -\frac{1}{m}\sum_{i=1}^{m} \log(1 - D(G(z_i))),$$

где $logD(x_i)$ — логарифмическая вероятность того, что дискриминатор точно классифицирует реальные данные, $\log(1 - D(G(z_i))$ — логарифмическая вероятность того, что дискриминатор правильно классифицирует сгенерированные образцы как поддельные. Дискриминатор стремится уменьшить эти потери, точно определяя искусственные и реальные образцы.

Цель обучения GAN — найти баланс между двумя моделями. Это достигается за счет использования состязательной функции потерь, которая измеряет разницу между сгенерированными образцами и реальными образцами.

Формула минимизации потерь выглядит следующим образом:

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

где фактические выборки данных, полученные из истинного распределения данных, p_{data} представлены в виде x, Случайный шум, взятый из предыдущего распределения $p_z(z)$ (обычно нормальное или равномерное распределение) представлен переменной z [2].

Наглядно архитектура базовой модели представлена на рисунке 1.

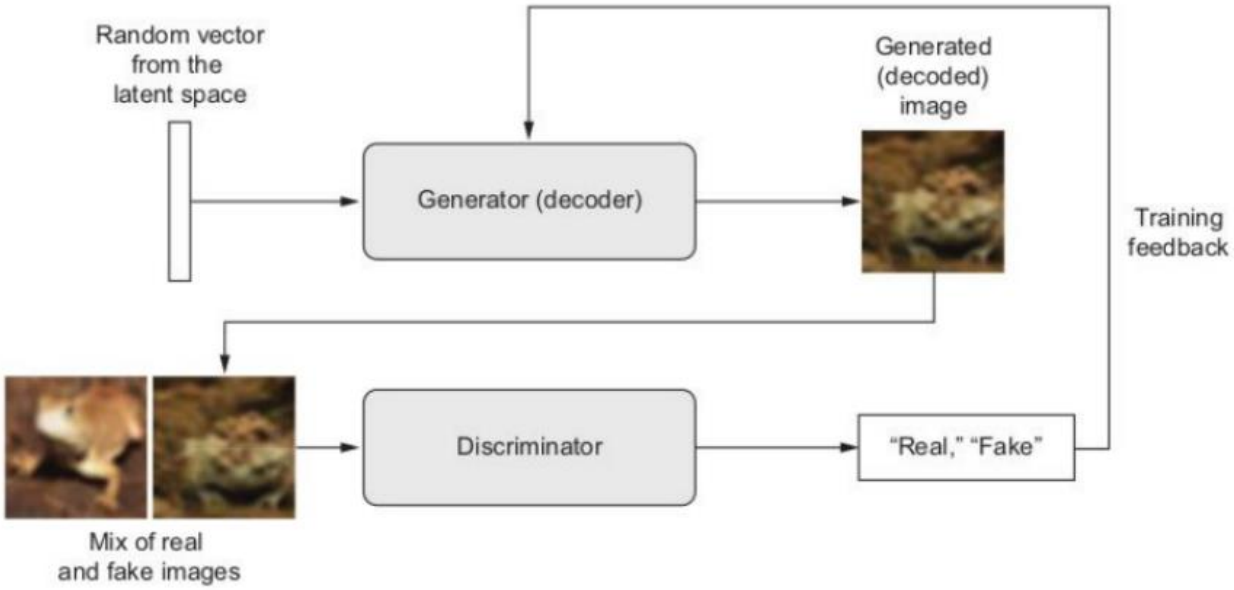

Рисунок 1 — Архитектура модели GAN

Основные недостатки базовой архитектуры:

- Коллапс мод (Mode Collapse) — генератор может начать создавать однотипные данные, полностью игнорируя разнообразие в распределении реальных данных.
- Обучение базовой GAN часто нестабильно, поскольку две сети (генератор и дискриминатор) могут не достигнуть равновесия. Это приводит к прерывистым улучшениям или полной остановке обучения.
- Проблема «разрыва градиентов» возникает, когда дискриминатор становится слишком сильным и практически всегда правильно классифицирует данные. Это оставляет генератор без полезного сигнала для обновления.

2.2 DCGAN (Deep Convolutional GAN) – применение свёрточных сетей

DCGAN, или «глубокая свёрточная генеративно-состязательная сеть», представляет собой модификацию базовой архитектуры GAN, специально адаптированную для генерации изображений. В этой модели как генератор, так и дискриминатор используют свёрточные нейронные сети (CNN), что делает её особенно эффективной для обработки визуальных данных. Рассмотрим подробнее обе части.

Сеть-генератор принимает на вход случайный шум и учится генерировать реалистичные изображения из этого шума. Её основная задача — отображать точки из скрытого пространства (обычно это случайный вектор) в пространство данных (пространство изображений). Генератор обычно состоит из нескольких свёрточных слоёв, за которыми следуют пакетная нормализация и функции активации ReLU. Он также может использовать транспонированные свёртки (также известные как слои повышения разрешения) для увеличения карт признаков, что позволяет генерировать изображения с более высоким разрешением [3].

Основными этапами в архитектуре генератора являются следующие:

1. Входные данные: вектор случайного шума или вектор скрытого пространства.
2. Свёрточные слои: эти слои принимают на вход вектор шума и увеличивают его разрешение, чтобы создать карты признаков с увеличивающимся пространственным разрешением и уменьшающейся глубиной.
3. Пакетная нормализация: нормализует активации в каждом слое, помогая стабилизировать процесс обучения.
4. Функция активации ReLU: функция активации ReLU (Rectified Linear Unit) используется после каждого слоя пакетной нормализации для введения нелинейности.

5. Выходной слой: в последнем слое обычно используется функция активации tanh для сжатия значений пикселей в диапазоне от -1 до 1, что позволяет получить итоговое синтетическое изображение.

Сеть-дискриминатор отвечает за различение реальных изображений из обучающего набора данных и поддельных изображений, созданных генератором. Она принимает изображение в качестве входных данных и учится классифицировать его как реальное (1) или поддельное (0). Как и генератор, дискриминатор также состоит из нескольких свёрточных слоёв, нормализации пакетов и функций активации.

Основными этапами в архитектуре дискриминатора являются следующие:

1. Входные данные: изображение (реальное или сгенерированное).

2. Свёрточные слои: эти слои обрабатывают входное изображение и понижают его разрешение, чтобы создать карты признаков с уменьшенными пространственными размерами и увеличенной глубиной.

3. Нормализация с помощью пакетов: как и генератор, нормализация с помощью пакетов используется для стабилизации обучения.

4. Нелинейная активация ReLU: в отличие от стандартного ReLU, нелинейная активация ReLU допускает небольшие отрицательные значения, предотвращая такие проблемы, как «умирающий ReLU».

5. Выходной слой: выходной слой представляет собой один нейрон с сигмоидальной функцией активации, который выдаёт оценку вероятности (от 0 до 1), указывающую, является ли входное изображение реальным или поддельным.

Процесс обучения DCGAN представлен на рисунке 2.

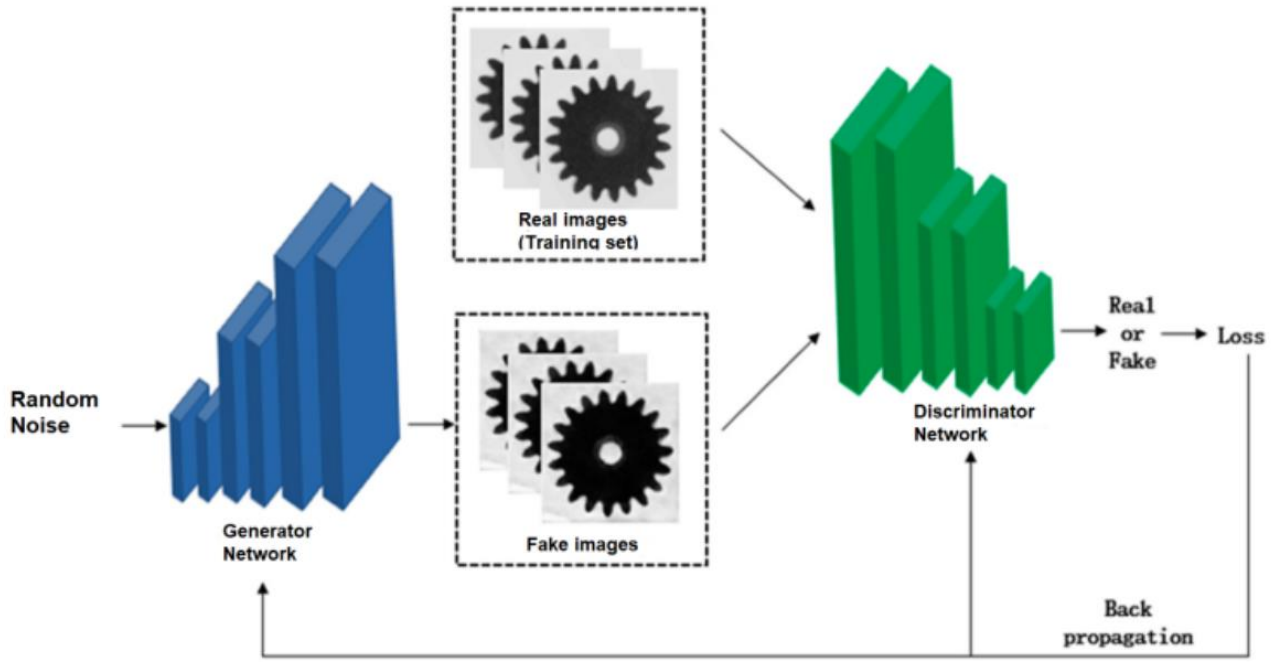

Рисунок 2 - Процесс обучения DCGAN

Важнейшим аспектом, способствующим успеху DCGAN, является тщательное чередование обучения генератора и дискриминатора. На протяжении всего процесса обучения в любой момент времени обновляются только веса одной сети, в то время как другая остаётся неизменной. Например, на этапе обучения генератора обновляются только веса генератора. Такое разделение позволяет предотвратить адаптацию дискриминатора к прогнозированию сгенерированных изображений как реальных, что может помешать прогрессу генератора. Вместо этого цель состоит в том, чтобы генератор создавал изображения, которые опытный распознаватель воспринимает как подлинные [4].

Основные недостатки:

- Хотя DCGAN лучше справляется с коллапсом мод, эта проблема остаётся. Генератор может сосредоточиться на небольшом числе образцов, игнорируя другие моды распределения данных.
- Несмотря на использование свёрточных сетей, проблема дисбаланса между генератором и дискриминатором сохраняется. Слишком сильный дискриминатор приводит к затруднённому обучению генератора.

2.3 WGAN (Wasserstein GAN) и улучшение устойчивости обучения

Wasserstein GAN (WGAN) — это разновидность генеративно-состязательной сети, которая решает проблему коллапса мод и нестабильности обучения, часто встречающихся в традиционных GAN. WGAN решают эту проблему, используя в качестве функции потерь расстояние Вассерштейна. Такое изменение функции потерь приводит к более стабильному обучению, улучшенной сходимости и более качественному генерированию образцов.

Расстояние Вассерштейна — это мера несходства между двумя распределениями вероятностей. Оно определяется как минимальная стоимость переноса массы из одного распределения в другое с учётом расстояния между точками в пространстве. Расстояние Вассерштейна обладает желаемыми свойствами, такими как непрерывность и дифференцируемость, что делает его подходящим для использования в качестве функции потерь в GAN.

Архитектура WGAN похожа на традиционную GAN с генератором и дискриминатором. Однако есть несколько ключевых отличий:

1. Функция потерь: WGAN используют расстояние Вассерштейна в качестве функции потерь вместо традиционной функции потерь. Это позволяет получить более значимую и непрерывную оценку разницы между реальным и сгенерированным распределением данных.

2. Ограничение весовых коэффициентов: для того, чтобы оптимизация WGAN была корректной, расстояние Вассерштейна должно быть 1-Липшицевой функцией. Это означает, что градиенты дискриминатора не должны становиться слишком большими. Если функция критика не удовлетворяет условию Липшица, вычисленное расстояние Васерштейна перестанет быть корректной метрикой. Чтобы обеспечить выполнение условия Липшица для дискриминатора, весовые коэффициенты дискриминатора ограничиваются заданным диапазоном. Это гарантирует, что дискриминатор останется эффективным критиком для расстояния Вассерштейна.

3. Отсутствие логарифма в функции потерь: в отличие от традиционных GAN, WGAN не используют логарифм в функции потерь. Это помогает избежать исчезающих градиентов и повышает стабильность обучения.

4. Критик вместо дискриминатора: в WGAN дискриминатор часто называют критиком, поскольку его роль заключается в оценке расстояния Вассерштейна между реальным и сгенерированным распределениями данных, а не в классификации образцов как реальных или поддельных [5].

2.4 Conditional GAN (CGAN) – условные GAN

Условная генеративно-состязательная сеть (Conditional GAN, CGAN) — это модификация архитектуры DCGAN, которая позволяет лучше контролировать характеристики генерируемых изображений. В отличие от DCGAN, CGAN может выборочно изменять определённые характеристики генерируемого изображения, предоставляя генератору дополнительную информацию, например метки, во время обучения. Этот ввод позволяет генератору генерировать изображения, соответствующие информации о метках и конкретным характеристикам. Различия между CGAN и DCGAN заключаются в изменениях в методологии обучения. В CGAN метки направляют генератор во время обучения для перемещения в скрытое пространство — низкоразмерное пространство случайного шума, которое генератор использует для создания изображений. Эти метки дают CGAN больше контроля над тем, что он генерирует, позволяя ему более точно перемещаться в скрытом пространстве и ориентироваться на метки.

Общая архитектура CGAN схожа с архитектурой DCGAN, но с небольшими изменениями. Архитектура дискриминатора в CGAN схожа с архитектурой дискриминатора в DCGAN и состоит из нескольких свёрточных слоёв, слоёв пакетной нормализации и функций активации Leaky ReLU. Однако в CGAN дискриминатор получает дополнительный ввод (см. Рисунок 3): информацию об условиях, метки и сгенерированное изображение. Этот

дополнительный ввод позволяет дискриминатору учитывать как реалистичность изображения, так и согласованность информации об условиях при оценке сгенерированного изображения. Архитектура генератора в CGAN похожа на архитектуру DCGAN и состоит из нескольких транспонированных свёрточных слоёв, слоёв пакетной нормализации и функций активации ReLU. Однако в CGAN генератор получает дополнительный ввод — информацию об условиях и случайный шум, используемый в качестве скрытого кода. Этот ввод позволяет генератору создавать изображения, соответствующие информации об условиях.

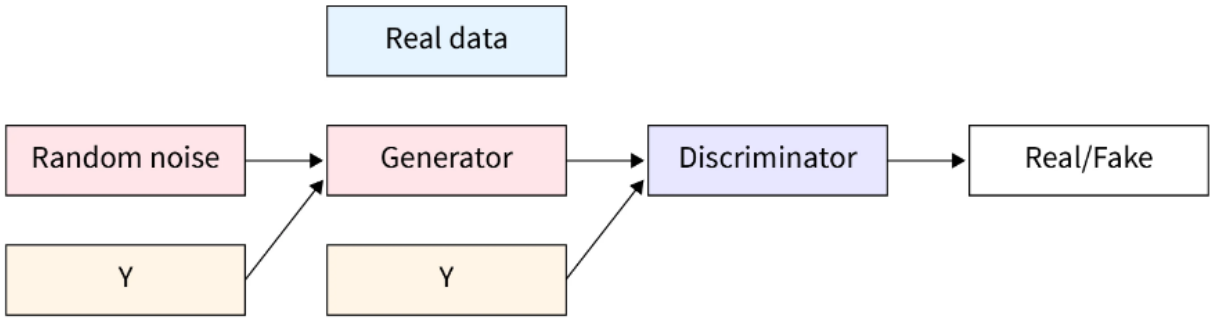

Рисунок 3 - Архитектура CGAN

Поскольку цель генератора состоит в том, чтобы постепенно создавать более качественные поддельные изображения, ему необходимо минимизировать разницу между прогнозируемым изображением и целевым. В этой архитектуре модель использует одну горячую кодированную метку, чтобы решить, какие признаки учитывать. Таким образом, функция потерь выглядит следующим образом.

$$J_G = -\frac{1}{m}\sum_{i=1}^{m} logD(G(z_i|y`)).$$

Поскольку задача дискриминатора состоит в том, чтобы маркировать сгенерированные изображения, его выходные данные представляют собой вероятность того, что изображение является истинным. Таким образом, функция потерь выглядит следующим образом.

$$J_D = -\frac{1}{m}\sum_{i=1}^{m} logD(x_i|y) - -\frac{1}{m}\sum_{i=1}^{m} \log(1 - D(G(z_i|y`))).$$

Обучение CGAN похоже на обучение любой другой GAN. Дискриминатор и генератор работают параллельно, создавая новые изображения и определяя,

насколько они реалистичны. Генератор сначала создаёт случайный шум и передаёт его дискриминатору. Дискриминатор в этом случае также получает метки и возвращает вероятность того, насколько реалистичным он считает сгенерированное изображение. Эта вероятность передаётся генератору, который постепенно обновляет свои веса, чтобы генерировать более качественные изображения. Этот цикл продолжается до тех пор, пока не будет достигнуто требуемое качество изображений [6].

К основным недостаткам данного метода можно отнести следующее:

- CGAN сильно зависит от качества и полноты меток данных. Ошибочные или неполные метки приводят к значительному снижению качества генерации.
- CGAN плохо масштабируется к задачам с большим числом категорий. Генератору сложно одновременно обучиться воспроизводить разнообразие между всеми классами.
- Проблемы базовой GAN, такие как нестабильность обучения и коллапс мод, сохраняются и в CGAN.

2.5 StyleGAN и StyleGAN2 – генерация реалистичных изображений

В StyleGAN применяется глубокая сеть, называемая картографической сетью, для преобразования вектора шумов z в промежуточное скрытое пространство w. Цель этой картографической сети - создать распутанные объекты, которые легко визуализировать с помощью генератора, и избежать комбинаций функций, которых нет в обучающем наборе данных.

StyleGAN представляет сеть отображения f для преобразования z в это промежуточное скрытое пространство с использованием восьми полностью связанных слоев (см Рисунок 4). w можно рассматривать как новый z (z`). Через эту сеть скрытое пространство размером 512 Dz преобразуется в промежуточное скрытое пространство размером 512 Dw.

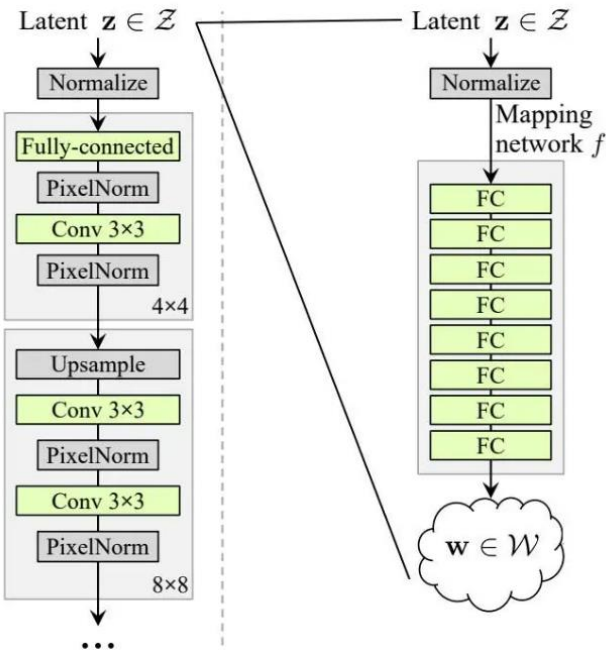

Рисунок 4 – сравнительная архитектура базовой GAN и StyleGAN

В обычном GAN z используется в качестве входных данных только для первого глубокого сетевого уровня. Можно утверждать, что его роль уменьшается по мере того, как мы углубляемся в сеть. В StyleGAN применяют отдельно изученную аффинную операцию A для преобразования w в каждом слое (см. Рисунок 5). Этот преобразованный w действует как информация о стиле, применяемая к пространственным данным.

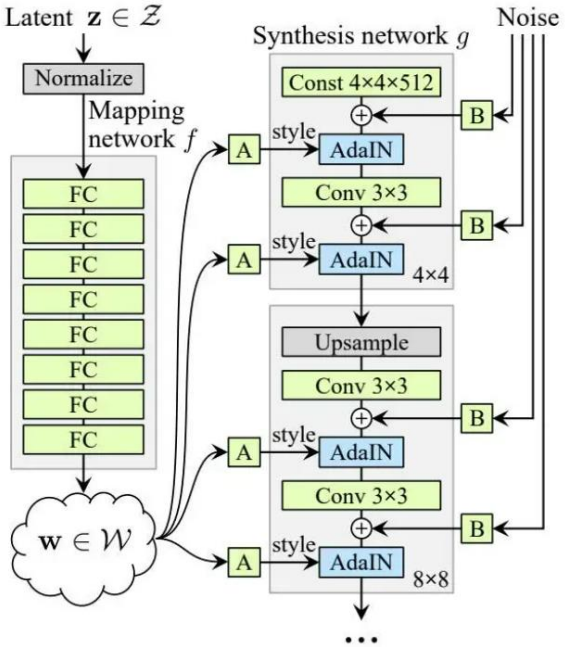

Рисунок 5 – Архитектура StyleGAN

Рассмотрим улучшения в StyleGAN относительно базовой архитектуры.

Одним из улучшений будет добавление картографической сети и стиля. Для последнего ADaIN (адаптивная нормализация экземпляров) заменит PixelNorm при применении стиля к пространственным данным (см. Рисунок 6).

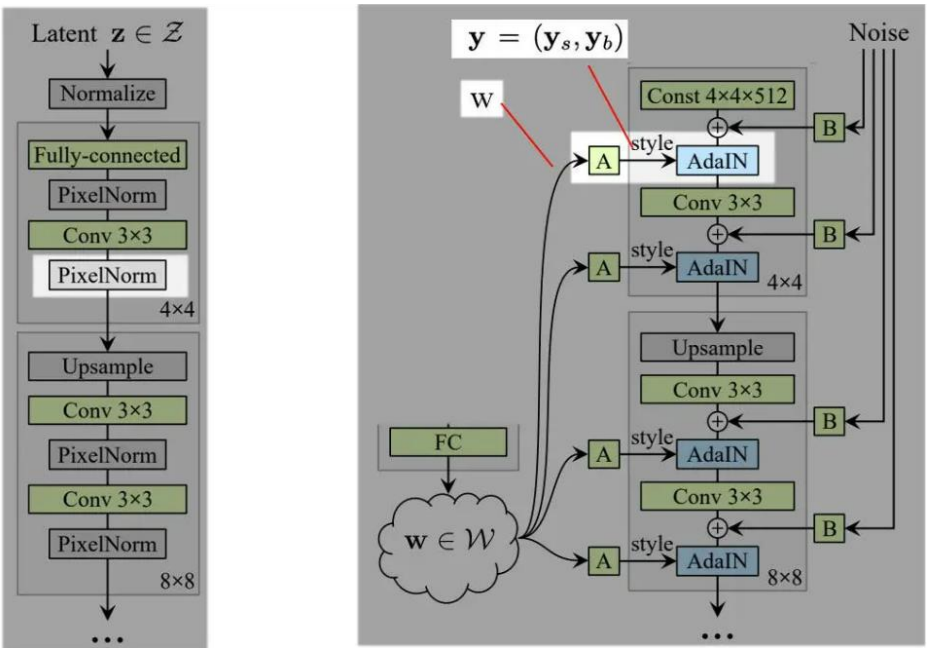

Рисунок 6 - AdaIN

AdaIN определяется как:

$$AdaIN(x_i, y) = y_{s,i} \frac{x_i - \mu(x_i)}{\sigma(x_i)} + y_{b,i},$$

где x_i - входной тензор, $\mu(x_i)$ и $\sigma(x_i)$ - среднее значение и стандартное отклонение признаков для x_i, $y_{s,i}$ и $y_{b,i}$ - параметры, задающие стиль: они масштабируют и сдвигают нормализованные признаки, добавляя стиль к изображению. Таким образом, AdaIN позволяет смешивать содержимое, описываемое признаками x_i, и стиль, задаваемый параметрами $y_{s,i}$ и $y_{b,i}$.

В базовом GAN входом для первого уровня является z. Эмпирические результаты показали, что добавление входной переменной в первый уровень StyleGAN не дает никаких преимуществ, и поэтому она заменяется постоянным входом. Для улучшения входные данные первого уровня заменяются постоянной матрицей с размером 4 × 4 × 512.

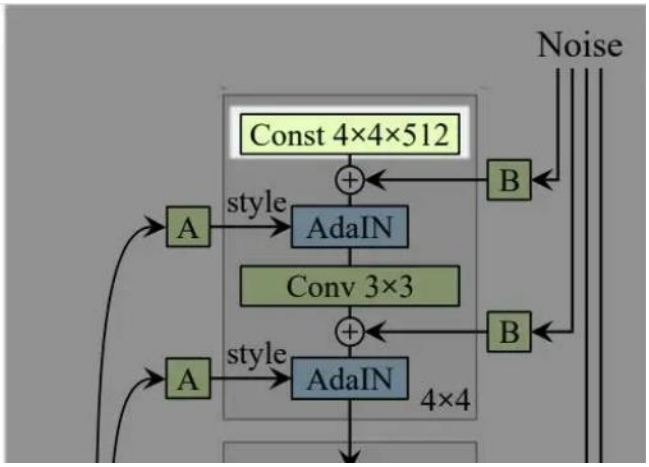

Рисунок 7 - изменения в StyleGAN

Еще одним улучшением StyleGAN относительно базовой модели является внесение шума на каждом уровне итерации для создания реалистичных мелких деталей (см. Рисунок 8)

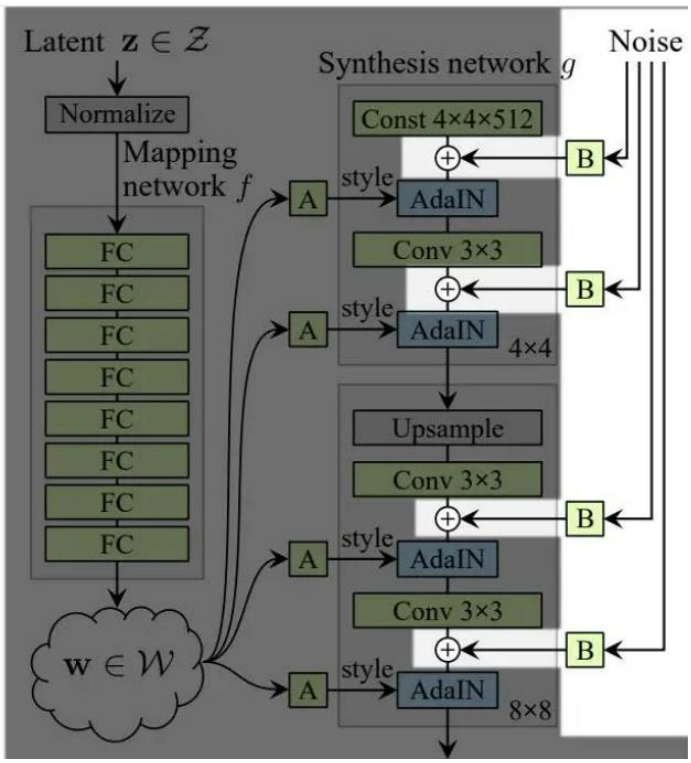

Рисунок 8 - добавление шума в StyleGAN.

Например, для пространственного слоя 8 × 8 создается матрица 8 × 8 с элементами, содержащими некоррелированный гауссов шум. Эта матрица будет использоваться всеми картами функций. Но StyleGAN изучит отдельный коэффициент масштабирования для каждой карты функций и умножит его на

матрицу шума перед добавлением его к выходным данным предыдущего слоя [7].

Однако модель StyleGAN имела свой недостаток. На изображениях, сгенерированных StyleGAN, наблюдаются артефакты, похожие на капли (похожие на капли воды), которые более очевидны при изучении промежуточных карт функций внутри сети генератора (см. Рисунок 9).

Рисунок 9 - Артефакты на изображениях StyleGAN

Данную проблему связывали с операцией AdaIN, которая нормализует среднее значение и дисперсию каждой карты признаков по отдельности, тем самым потенциально уничтожая любую информацию, найденную в величинах признаков относительно друг друга.

StyleGAN2 предложил решить проблему следующим образом.

Прежде всего, представим архитектуру StyleGAN в несколько измененном виде, где модуль AdaIN разделен на два (см. Рисунок 10).

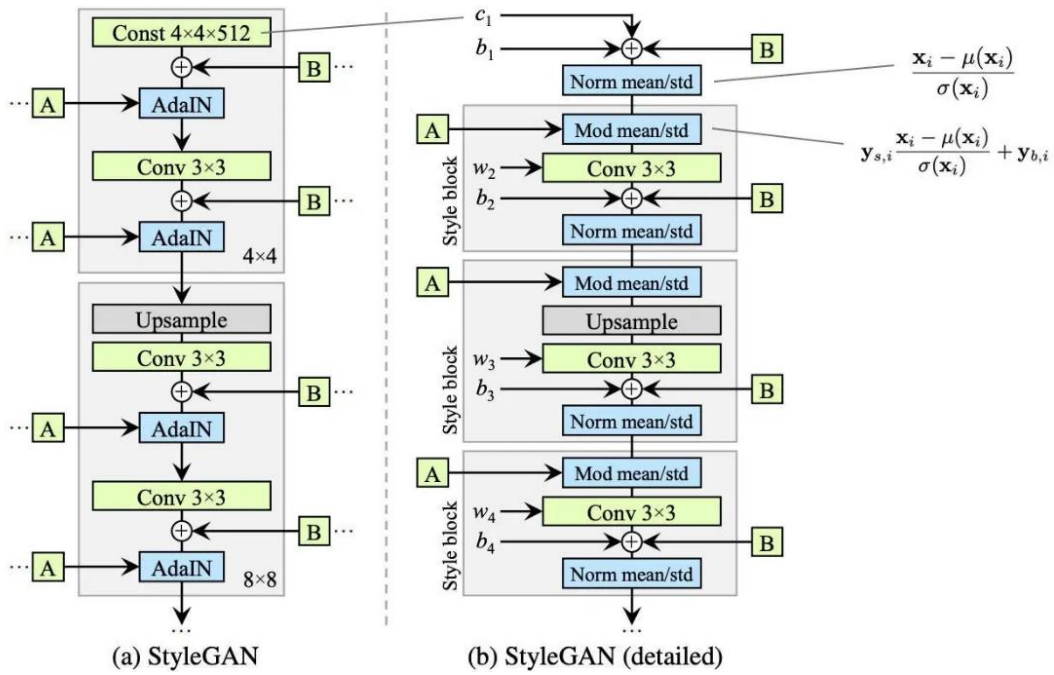

Рисунок 10 - Детализация схемы StyleGAN

Таким образом, в StyleGAN2 были внесены следующие изменения (см Рисунок 11 и 12):

1. В StyleGAN2 фиксированный стартовый тензор заменён на сгенерированное представление, которое формируется через один из первых свёрточных слоев. Таким образом, начальные карты признаков стали обучаемыми, а их форма — более гибкой.

2. Каждый свёрточный слой сначала масштабирует веса в соответствии с параметрами стиля (модуляция), а затем приводит их к единому масштабу (демодуляция). Это снижает риск возникновения артефактов.

3. Шум стал добавляться через отдельный обучаемый слой (т.е. его влияние теперь регулируется весами). Это позволило более точно контролировать, какие именно текстурные детали добавляются на разных уровнях.

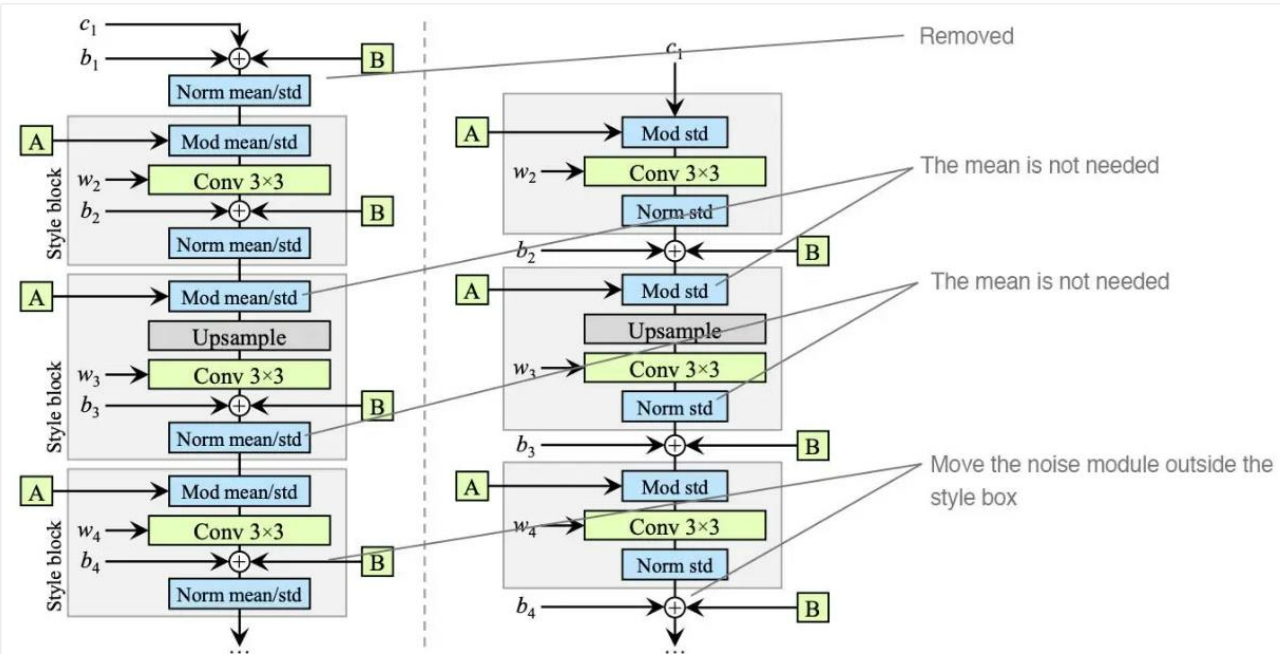

Рисунок 11 - пересмотренная архитектура относительно детализованной схемы StyleGAN

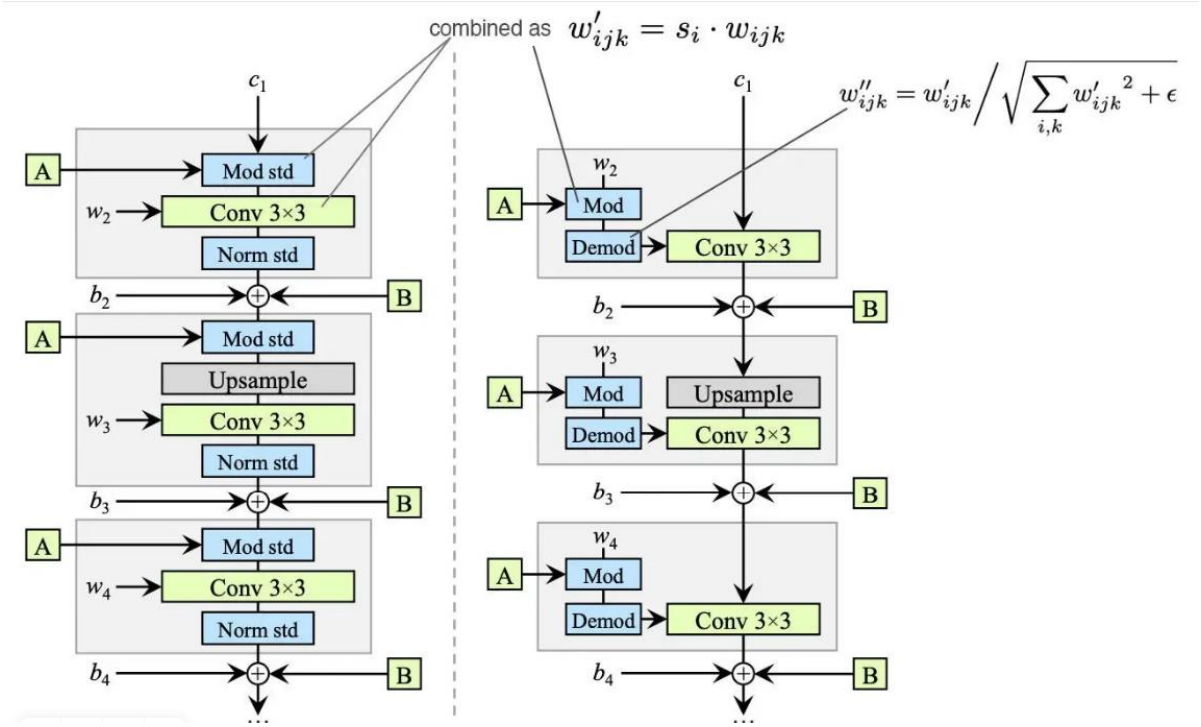

Рисунок 12 - промежуточные изменения архитектуры (слева) и архитектура StyleGAN2 (справа)

Благодаря всем изменениям, StyleGAN2 стал эволюционным шагом вперёд, устранив слабые места оригинального StyleGAN и сделав генерацию изображений более реалистичной, гибкой и качественной. Улучшения, такие как

отказ от AdaIN, использование демодуляции весов и обучаемого начального тензора, сделали StyleGAN2 эталоном в генерации изображений [8].

3 Исследования применения GAN в различных областях

3.1 Медицина

Первой областью применения GAN рассмотрим медицину. В данной сфере проводились исследования по генерации биомедицинских изображений для аугментации данный с помощью GAN [9].

Для исследования возможности генерации биомедицинских изображений из ограниченного набора данных с целью аугментации был использован набор снимков МРТ головного мозга с опухолями, который предоставлялся в рамках соревнования Multimodal Brain Tumor Segmentation Challenge 2020. Датасет содержал 369 снимков в формате NIFTI с разрешением 240 x 240 x 155 вокселей. Каждому изображению сопоставлялся набор аннотаций, размеченных вручную экспертами, следуя протоколам, результаты их работы одобрены сертифицированной экспертной комиссией нейрорадиологов (см. Рисунок 13).

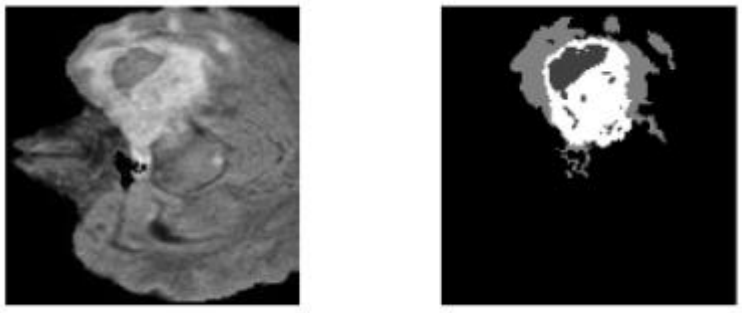

Рисунок 13 - снимок МРТ головного мозга и соответствующая ему аннотация

Из снимков данного датасета был составлен собственный в размере 30 изображений, взятых случайным образом. Затем было изменено разрешение каждого изображения на 256 x 256 пикселей, а также для удобства изменен формат снимков на TIFF.

После обучения нейронной сети, а также сдвигов, поворотов, переворотов изображений по осям удалось увеличить число изображений в наборе данных с 30 до 300 изображений. Примеры сгенерированных снимков МРТ головного мозга представлены на рисунке 14.

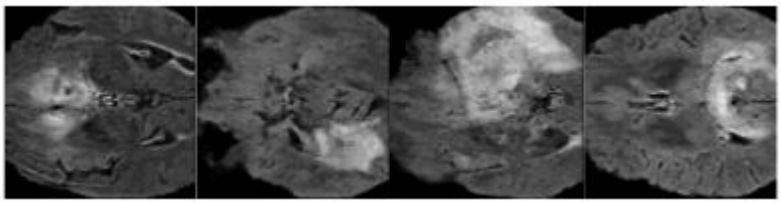

Рисунок 14 - сгенерированные снимки МРТ головного мозга

По окончанию обучения нейронной сети была проведена оценка качества модели. Модель продемонстрировала свою эффективность в задаче сегментации опухоли для случаев с увеличением данных и без нее, достигая значений метрик 0,71 против 0,27 Dice Score, 0,69 против 0,43 Mean IoU на обучающем наборе данных, 0,60 против 0,25 Dice Score, 0,58 против 0,37 Mean IoU на проверочном наборе данных и 0,66 против 0,25 Dice Score, 0,59 против 0,38 Mean IoU на тестовом наборе данных. Был сделан вывод, что благодаря использованию генеративно-состязательных сетей возможно создание реалистичных биомедицинских изображений с целью увеличения конкретного набора данных.

3.2 Создание аудио

На конференции ICLR 2019 разработчики Google Magenta представили новый подход к синтезу аудио с помощью генеративно-состязательной нейронной сети. GANSynth позволяет генерировать музыку в 50 000 раз быстрее методов, основанных на авторегрессии, таких как WaveNet от DeepMind. В работе исследователи адаптировали модель WaveGAN и предложили несколько новых архитектур [10].

Вместо последовательной генерации звука, GANSynth генерирует всю последовательность параллельно, синтезируя аудио в 50000 быстрее, чем WaveNet. В отличие от автоэнкодеров оригинальной WaveNet в которой используется скрытый код с распределением по времени, GANSynth генерирует все аудио из одного латентного вектора, что упрощает разделение глобальных функций, таких как высота и тембр и позволяет контролировать их независимо.

Для обучения нейросети исследователи использовали набор данных NSynth, состоящий из 300000 отдельных нот 1000 музыкальных инструментов. Много примеров сгенерированной музыки можно послушать здесь. На них показано, как тембр интерполируется по ходу произведения.

Во время экспериментов ученые разработчики обнаружили, что метод IF-GAN, который генерирует мгновенные частоты (instantaneous frequencies) для фазового компонента превосходит другие модели в когерентности сигнала. На рисунке 15 показаны типы сигналов. WaveGAN и PhaseGAN имеют много фазовых неровностей, в то время как результат IF-GAN гораздо более последовательный, с небольшими вариациями от цикла к циклу.

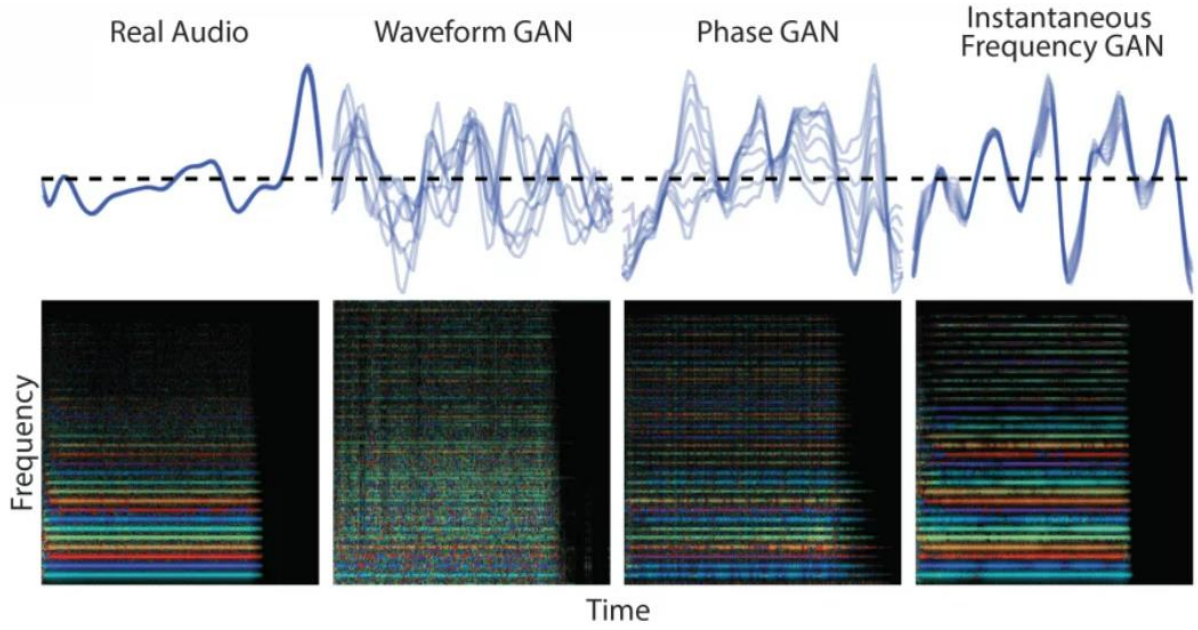

Рисунок 15 - Сгенерированная форма волны

На рисунке 16 показаны результаты тестов прослушивания, в ходе которых пользователям проигрывали аудиопримеры, созданные двумя разными методами, и спрашивали, какой из них они предпочитают:

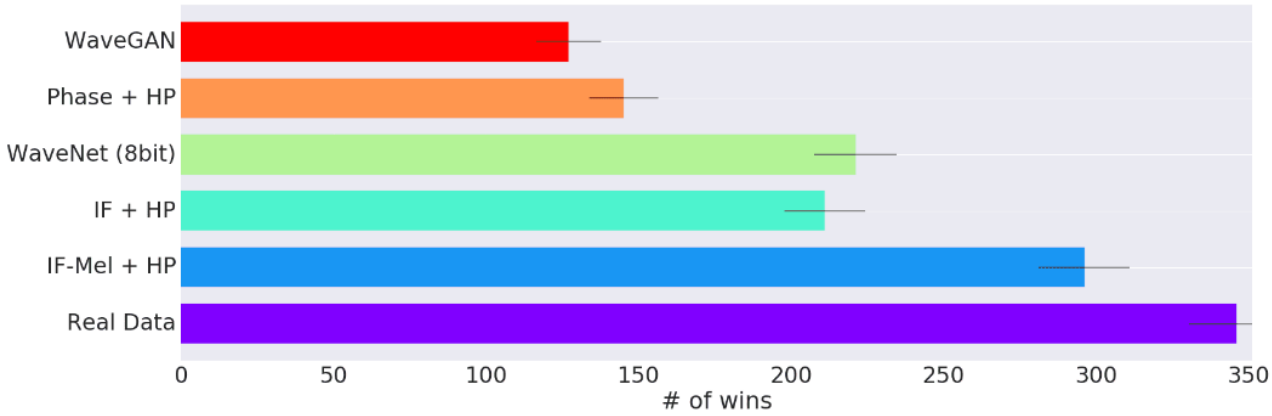

Рисунок 16 - Результаты исследований GANSynth

По графику можно видеть, что слушатели отдавали больше предпочтения IF-GAN в сравнении с другими моделями.

3.3 Модификация изображений

Другим не менее важным исследованием возможностей GAN является модификация (вставка или удаление) объектов на изображении. Метод использует традиционные предобученные GAN для перевода изображений, такие как CycleGAN или Pix2Pix GAN, которые дообучаются на ограниченном наборе данных эталонных изображений, связанных с семантическими картами. В работе [11] поднимается вопрос качественной и количественной эффективности техники, при применении ее в области подделки и редактирования изображений.

В работе предлагается следующая структура манипуляции изображениями: используется генеративная модель (с архитектурой, подобной CycleGAN или Pix2PixHD GAN, предварительно обученной на стандартных больших наборах данных таким образом, чтобы она могла генерировать изображения на основе их семантических масок), вместе с небольшой коллекцией изображений, помеченных их семантическими масками. Будем называть этот набор данных smalldata. После дообучается предварительно обученный GAN, на smalldata, и настраивается модель так, чтобы GAN генерировал изображение, которое близко похоже на оригинальное изображение

из семантической маски, на которой он был обучен. Другими словами, от GAN требуется запомнить соответствие между семантической картой и изображением. Как только GAN обучен, генерируется поддельная семантическая маска, например, удаляя объект из оригинальной маски, M, которая является образцом в smalldata. Подделка семантической карты может быть легко выполнена с использованием программного обеспечения с открытым исходным кодом, такого как Photoshop или GIMP. Используя эту поддельную карту, GAN генерирует изображение, которое соответствует этой поддельной карте, что в данном случае включает генерацию изображения, похожего на оригинальное изображение, которое было связано с M, но с удаленным объектом (см. Рисунок 17).

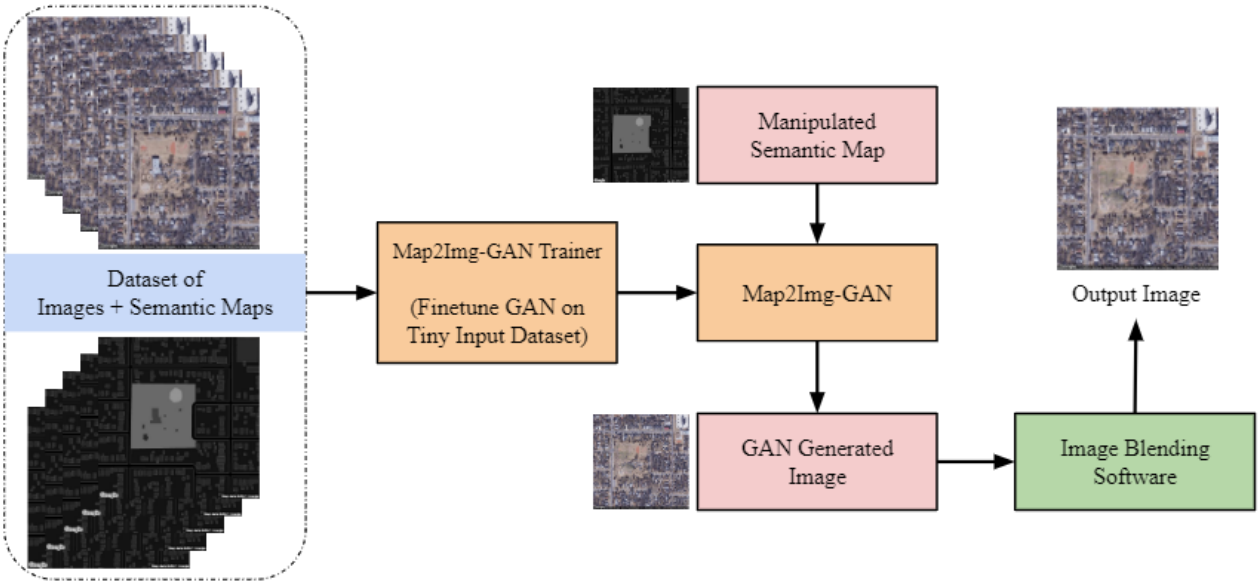

Рисунок 17 - Обзор предлагаемой системы манипулирования изображениями

Некоторые результаты предложенного метода можно видеть на рисунках 18 и 19:

Рисунок 18 - результаты тестирования

Рисунок 19 - результаты тестирования

Данная работа показывает, что большинство современных типичных техник судебной экспертизы на основе глубокого обучения испытывают трудности с нахождением изображений, сгенерированных GAN, которые были изменены после их создания. Представленная методология сохраняет подавляющее большинство пикселей оригинального изображения и может генерировать подделки, которые трудно визуально идентифицировать для человека.

ЗАКЛЮЧЕНИЕ

Генеративно-состязательные нейронные сети (GAN) представляют собой одно из самых значительных достижений в области машинного обучения и искусственного интеллекта. Их уникальная архитектура, основанная на взаимодействии двух сетей — генератора и дискриминатора, позволила открыть новые горизонты в решении сложных задач, связанных с генерацией, обработкой и анализом данных. GAN находят широкое применение в различных областях человеческой деятельности.

С момента появления базовой архитектуры GAN в 2014 году, эти технологии прошли огромный путь, эволюционировав в более сложные и специализированные модели, такие как DCGAN, WGAN, CGAN и StyleGAN. Каждая из них внесла вклад в развитие генеративных технологий, решая проблемы своих предшественников и открывая новые возможности.

На сегодняшний день GAN все еще не идеальны, у них по-прежнему остаются недостатки и большая область для развития. Так, перспективы развития можно разделить на два основных типа:

1. Улучшение качества генерируемых данных
2. Снижение предвзятости нейросетей

Таким образом, генеративно-состязательные сети являются не только важным инструментом в современных исследованиях, но и одной из ключевых технологий будущего. Их дальнейшее развитие и интеграция в научные и прикладные области продолжат изменять нашу жизнь, повышая эффективность работы и создавая новые перспективы в самых разных сферах.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Инь и ян машинного обучения: в чем сила двойственной природы GAN [Электронный ресурс] // SberAI : [сайт]. — URL: https://ai.sber.ru/post/in-i-yan-mashinnogo-obucheniya:-v-chem-sila-dvojstvennoj-prirody-gan (дата обращения: 20.12.2024). – Загл. с экрана. – Яз рус.

2. Ian J. Goodfellow, Generative Adversarial Networks [Электронный ресурс] / Ian J. Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, Yoshua Bengio // arXiv : [Электронный ресурс]. — URL: https://arxiv.org/pdf/1406.2661 (дата обращения: 20.12.2024). – Загл. с экрана. – Яз англ.

3. Alec Radford, Unsupervised Representation Learning with Deep Convolutional Generative Adversarial Networks [Электронный ресурс] / Alec Radford, Luke Metz, Soumith Chintala // arXiv : [Электронный ресурс]. — URL: https://arxiv.org/pdf/1511.06434 (дата обращения: 20.12.2024). – Загл. с экрана. – Яз англ.

4. Ajitesh Kumar DCGAN Architecture Concepts, Real-world Examples [Электронный ресурс] / Ajitesh Kumar // Analytics Yogi : [Электронный ресурс]. — URL: https://vitalflux.com/dcgan-architecture-concepts-real-world-examples/ (дата обращения: 20.12.2024). – Загл. с экрана. – Яз англ.

5. Aadhithya Sankar Demystified: Wasserstein GANs (WGAN) [Электронный ресурс] / Aadhithya Sankar // Medium : [Электронный ресурс]. — URL: https://towardsdatascience.com/demystified-wasserstein-gans-wgan-f835324899f4 (дата обращения: 20.12.2024). – Загл. с экрана. – Яз англ.

6. Subhaditya Mukherjee Introduction to Conditional GANs [Электронный ресурс] / Subhaditya Mukherjee // Scaler Topics : [Электронный ресурс]. — URL: https://www.scaler.com/topics/cgan/ (дата обращения: 20.12.2024). – Загл. с экрана. – Яз англ.

7. Tero Karras, A Style-Based Generator Architecture for Generative Adversarial Networks [Электронный ресурс] / Tero Karras, Samuli Laine, Timo Aila

// arXiv : [Электронный ресурс]. — URL: https://arxiv.org/pdf/1812.04948 (дата обращения: 20.12.2024). – Загл. с экрана. – Яз англ.

8. Tero Karras, Analyzing and Improving the Image Quality of StyleGAN [Электронный ресурс] / Tero Karras, Samuli Laine, Miika Aittala, Janne Hellsten, Jaakko Lehtinen, Timo Aila // arXiv : [Электронный ресурс]. — URL: https://arxiv.org/pdf/1912.04958 (дата обращения: 20.12.2024). – Загл. с экрана. – Яз англ.

9. Новоселов И.Э. Генерация биомедицинских изображений для аугментации данных с помощью генеративно-состязательных сетей / И.Э. Новоселов, А.А. Смирнов // Международный научно-исследовательский журнал. — 2024. — №5 (143) S. - URL: https://research-journal.org/archive/5-143-2024-may/10.60797/IRJ.2024.143.158 (дата обращения: 20.12.2024). – Загл. с экрана. – Яз рус.

10. Jesse Engel GANSynth: Making music with GANs [Электронный ресурс] / Jesse Engel // magenta : [Электронный ресурс]. — URL: https://magenta.tensorflow.org/gansynth (дата обращения: 20.12.2024). – Загл. с экрана. – Яз англ.

11. Chandrakanth Gudavalli, CIMGEN: Controlled Image Manipulation by Finetuning Pretrained Generative Models on Limited Data [Электронный ресурс] / Chandrakanth Gudavalli, Erik Rosten, Lakshmanan Nataraj, Shivkumar Chandrasekaran, B. S. Manjunath // arXiv : [Электронный ресурс]. — URL: https://arxiv.org/pdf/2401.13006 (дата обращения: 20.12.2024). – Загл. с экрана. – Яз англ.