МИНИСТЕРСТВО ОБРАЗОВАНИЯ РЕСПУБЛИКИ БЕЛАРУСЬ
БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ
ФАКУЛЬТЕТ ПРИКЛАДНОЙ МАТЕМАТИКИ И ИНФОРМАТИКИ
Кафедра дискретной математики и алгоритмики

КАРПЕНКО
Дмитрий Сергеевич

СРАВНИТЕЛЬНЫЙ АНАЛИЗ МЕТОДОВ ГЕНЕРАЦИИ ИСКУССТВЕННЫХ МЕДИЦИНСКИХ ИЗОБРАЖЕНИЙ

Дипломная работа

Научный руководитель:
старший преподаватель
А.А. Буславский

Допущена к защите

«___» _____ 2024 г.

Заведующий кафедрой дискретной математики и алгоритмики,
доктор физико-математических наук, профессор В.М. Котов

Минск, 2024

РЕФЕРАТ

Дипломная работа, 52 страницы, 24 рисунка, 4 таблицы, 15 источников, 3 приложения.

Ключевые слова: ГЕНЕРАЦИЯ МЕДИЦИНСКИХ ИЗОБРАЖЕНИЙ, ИЗОБРАЖЕНИЯ, ГЕНЕРАТИВНО-СОСТЯЗАТЕЛЬНЫЕ МОДЕЛИ, GAN, DCGAN, SAGAN, CGAN, VAE, BIG-GAN, BIG-GAN-DEEP, SPADE, МЕХАНИЗМ ВНИМАНИЯ, РЕНТГЕНОВСКИЕ СНИМКИ, КТ-СНИМКИ, НЕЙРОННЫЕ СЕТИ.

Объект исследования: генеративно-состязательные нейронные сети.

Предмет исследования: реализация архитектур моделей генеративных нейронных сетей в процессе обработки медицинских изображений.

Цель исследования: теоретически изучить и опытно-экспериментально провести построение, обучение и оценку генеративных моделей в контексте задачи генерации искусственных медицинских изображений.

Методы исследования: системный подход, изучение соответствующей литературы и электронных источников, постановка задачи и её решение.

В результате исследования определены наиболее подходящие и полезные идеи, подходы и методы для генерации медицинских изображений.

Области применения. Полученные в ходе исследования данные могут быть использованы в медицине и бизнесе.

РЭФЕРАТ

Дыпломная работа, 52 старонкі, 24 відарысаў, 4 табліцы, 15 крыніц, 3 дадаткі.

Ключавыя словы: ГЕНЕРАЦЫЯ МЕДЫЦЫНСКІХ ВІДАРЫСАЎ, ВІДАРЫСЫ, ГЕНЕРАТЫЎНА-СПАБОРНІЦКІЯ МАДЭЛІ, GAN, DCGAN, SAGAN, CGAN, VAE, BIG-GAN, BIG-GAN-DEEP, SPADE, МЕХАНІЗМ УВАГІ, РЭНТГЕНАЎСКІЯ ЗДЫМКІ, КТ-ЗДЫМКІ, НЕЙРОННЫЯ СЕТКІ.

Аб'ект даследавання: генератыўна-спаборніцкія нейронныя сеткі.

Прадмет даследавання: рэалізацыя архітэктур мадэляў генератыўных нейронных сетак у працэсе апрацоўкі медыцынскіх відарысаў.

Мэта даследавання: тэарэтычна вывучыць і доследна-эксперыментальна правесці пабудову, навучанне і ацэнку генератыўных мадэляў у кантэксце задачы генерацыі штучных медыцынскіх відарысаў.

Метады даследавання: сістэмны падыход, вывучэнне адпаведнай літаратуры і электронных крыніц, фармуляванне задачы і яе рашэнне.

У выніку даследавання вызначаны найбольш прыдатныя і карысныя ідэі, падыходы і метады для генерацыі медыцынскіх відарысаў.

Вобласці прымянення. Атрыманыя ў ходзе даследавання даныя могуць быць выкарыстаны ў медыцыне і бізнэсе.

ABSTRACT

Diploma, 52 pages, 24 figures, 4 tables, 15 sources, 3 appendices.

Keywords: MEDICAL IMAGE GENERATION, IMAGES, GENERATIVE-ADVERSARIAL MODELS, GAN, DCGAN, SAGAN, SAGAN, CGAN, VAE, BIG-GAN, BIG-GAN-DEEP, SPADE, ATTENTION MECHANISM, X-RAY IMAGES, CT IMAGES, NEURAL NETWORKS.

Research Object: generative-adversarial neural networks.

Research Objective: implementation of generative neural network model architectures in medical image processing.

Research Methods: systematic approach, study of relevant literature and electronic sources, problem formulation and solution.

As a result of the research, identified the most appropriate and useful ideas, approaches and methods for medical image generation.

Areas of Application. The data obtained from the study can be used in medicine and business.

ОГЛАВЛЕНИЕ

ВВЕДЕНИЕ 7

ГЛАВА 1 ГЕНЕРАТИВНО-СОСТЯЗАТЕЛЬНЫЕ НЕЙРОННЫЕ СЕТИ 9

1.1 Введение в генеративно-состязательные нейронные сети 9

1.1.1 Общие сведения 9

1.1.2 Процесс обучения 10

1.1.3 Проблемы генеративно-состязательных моделей 11

1.2 Свёрточные генеративно-состязательные нейронные сети. Архитектура Deep Convolutional GAN 12

ГЛАВА 2 СВЁРТОЧНЫЕ ГЕНЕРАТИВНО-СОСТЯЗАТЕЛЬНЫЕ СЕТИ С БЛОКАМИ САМОВНИМАНИЯ 15

2.1 Архитектура Self-Attention GAN 16

2.2 Процесс обучения Self-Attention GAN 17

ГЛАВА 3 ДВУХЭТАПНАЯ УСЛОВНАЯ И БЕЗУСЛОВНАЯ ГЕНЕРАЦИЯ. CONDITIONAL GAN 19

3.1 Этапы генерации рентгеновского снимка 20

3.2 Генерация сегментированного изображения (маски) 21

3.2.1 Архитектура вариационного автокодировщика 21

3.2.2 Процесс обучения вариационного автокодирвщика 22

3.3 Восстановление изображения по результату его сегментации 23

3.3.1 Архитектура Conditional GAN 23

3.3.2 Процесс обучения Conditional GAN 25

ГЛАВА 4 АРХИТЕКТУРА BIG-GAN-DEEP ДЛЯ ГЕНЕРАЦИИ ИЗОБРАЖЕНИЙ ВЫСОКОЙ РАЗМЕРНОСТИ 27

4.1 Архитектура BIG-GAN-DEEP 27

4.2 Процесс обучения BIG-GAN-DEEP 28

4.3 Градиентная регуляризация для стабилизации обучения 29

ГЛАВА 5 АРХИТЕКТУРА SPADE-GAN ДЛЯ УСЛОВНОЙ ГЕНЕРАЦИИ ИЗОБРАЖЕНИЙ ВЫСОКОЙ РАЗМЕРНОСТИ 30

5.1 Архитектура SPADE-GAN 31

5.2 Процесс обучения SPADE-GAN 33
5.3 Применение SPADE-GAN 34

ГЛАВА 6 ЭКСПЕРИМЕНТЫ И РЕЗУЛЬТАТЫ 35
6.1 Применение архитектуры DC-GAN 37
6.2 Применение архитектуры SA-GAN 38
6.3 Применение архитектуры BIG-GAN-DEEP 40
6.4 Применение ансамбля генеративных моделей CGAN и VAE 43
6.5 Применение архитектуры SPADE-GAN 45
6.6 Выводы 49

ЗАКЛЮЧЕНИЕ 51

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ 53

ПРИЛОЖЕНИЯ 55
ПРИЛОЖЕНИЕ А 55
ПРИЛОЖЕНИЕ Б 56
ПРИЛОЖЕНИЕ В 73

ВВЕДЕНИЕ

Синтетические данные — это искусственно созданные данные, которые выглядят как данные из реального мира. В создании синтетических данных важную роль играют генеративные модели. Именно они являются основным способом создания поддельных данных, поскольку могут копировать статистические модели и характеристики реальных данных. В контексте медицинских изображений, часто требуется большое количество данных для обучения некоторых моделей машинного обучения. Часто собрать достаточное количество таких данных бывает трудно из-за вопросов конфиденциальности, редкости некоторых типов изображений, например, изображений с редкими патологиями, и сложности создания. Альтернативным способом устранения недостатка таких данных может послужить создание синтетических данных высокого качества с помощью генеративных нейронных сетей. Ниже перечислен ряд причин, по которым важно использовать генеративные модели для создания синтетических данных:

— **Конфиденциальность и защита данных.** Генеративные модели позволяют создавать синтетические наборы данных без персональной информации или конфиденциальных данных. Это делает наборы данных пригодными для исследований и разработок, при этом не нарушая конфиденциальность пользователей. Особенно важен этот пункт в контексте медицинских изображений.

— **Расширение (аугментация) данных.** С помощью генеративных моделей можно генерировать новые обучающие данные для пополнения реальных наборов данных. Это особенно полезно, когда данные редкие, их получение дорого или требует много времени.

— **Несбалансированные данные.** При наличии в данных несбалансированных классов, генеративные модели могут помочь, предоставляя синтетические примеры объектов недостающих классов.

— **Анонимизация.** Генеративные модели заменяют конфиденциальную информацию синтетическими, но статистически эквивалентными значениями. Это позволяет обмениваться данными для исследований или соблюдения нормативных требований без раскрытия конфиденциальной информации.

Медицинские изображения являются крайне чувствительными к любым изменениям данных. Они имеют сложную пространственно-геометрическую структуру, что связано с анатомией человека. Таким образом, достичь правдоподобия в генерации медицинских изображений — не простая задача.

В нашей работе мы будем изучать методы генерации изображений с помощью нейронных сетей, проводить сравнительный анализ моделей генерации по разным направлениям, оценивать качество генерируемых изображений.

Объект исследования: генеративно-состязательные нейронные сети.

Предмет исследования: реализация архитектур моделей генеративных нейронных сетей в процессе обработки медицинских изображений.

Цель дипломной работы — теоретически изучить и опытно-экспериментально провести построение, обучение и оценку генеративных моделей в контексте задачи генерации искусственных медицинских изображений.

Для реализации данной цели необходимо решить следующие **задачи**:

— Изучить теоретические модели и архитектуры генеративных нейронных сетей.

— Рассмотреть специфику и структуру работы свёрточных генеративно-состязательных нейронных сетей с блоками самовнимания.

— Раскрыть особенности ансамблирования генеративных моделей.

— Изучить модели для генерации искусственных медицинских изображений высокой размерности с анатомическими ограничениями.

— Осуществить реализацию архитектур моделей генеративных нейронных сетей, а именно Deep Convolutional GAN, Self-Attention GAN, Conditional GAN, VAE, BIG-GAN-DEEP, SPADE-GAN.

— Провести сравнительный анализ перечисленных выше архитектур генеративных моделей и сделать выводы.

Реализация поставленных задач позволит определить наиболее подходящие и выявить полезные идеи, подходы и методы для генерации медицинских изображений.

Методы исследования: системный подход, изучение соответствующей литературы и электронных источников, постановка задачи и её решение.

Работа состоит из введения, шести глав, девятнадцати параграфов, заключения, списка использованных источников и трёх приложений.

ГЛАВА 1

ГЕНЕРАТИВНО-СОСТЯЗАТЕЛЬНЫЕ НЕЙРОННЫЕ СЕТИ

1.1 Введение в генеративно-состязательные нейронные сети

1.1.1 Общие сведения

Глубокие генеративно-состязательные нейронные сети (ГСНС, Deep Generative adversarial Networks, GANs) были предложены и описаны Яном Гудфеллоу и коллегами в 2014 году в статье «Generative Adversarial Nets» [1]. ГСНС представляют собой мощный инструмент для создания новых данных, имитирующих наблюдаемые образцы. Они отличаются от других методов генерации данных тем, что обучаются на основе соревнования двух компонентов — генератора и дискриминатора.

Генератор в ГСНС является сетью, которая принимает на вход случайный вектор и генерирует данные, имитирующие распределение исходных данных. Например, в задаче генерации изображений, генератор может преобразовывать случайный вектор в реалистичные изображения, которые напоминают изображения из обучающего набора данных.

Дискриминатор, с другой стороны, является сетью, которая классифицирует данные как реальные или сгенерированные. Задача этой сети состоит в том, чтобы классифицировать те данные, которые были созданные генератором, как сгенерированные, а исходные данные как настоящие, реальные.

В процессе обучения генератор и дискриминатор взаимодействуют друг с другом. Генератор стремится создавать все более реалистичные данные, чтобы обмануть дискриминатор, в то время как дискриминатор старается стать более точным в определении реальных и сгенерированных данных. Это соревнование между генератором и дискриминатором приводит к улучшению качества генерируемых данных со временем. Такая структура соответствует минимаксной игре для двух игроков, как было показано в статье [1]. Равновесие этой игры достигается в ситуации, когда генератор создаёт идеальные подделки, которые выглядят так, будто они получены непосредственно из обучающих данных, а дискриминатору остается всегда угадывать с вероят-

ностью $\frac{1}{2}$, настоящими ли являются данные, полученные от генератора. Стоит отметить, что тип обучения генеративно-состязательных сетей — обучение без учителя.

1.1.2 Процесс обучения

Рассмотрим подробнее общий принцип обучения генеративно-состязательных моделей, определим функции потерь и опишем алгоритм.

Пусть x — объект данных, например, изображение. $D(x)$ — сеть-дискриминатор, принимающая на вход изображение и возвращающая скалярную величину — вероятность того, что изображение x из исходного распределения данных (из распределения тренировочной выборки), то есть является настоящим изображением. z — случайный вектор (как правило из многомерного стандартного или равномерного распределения). $G(z)$ — сеть-генератор, принимающая на вход случайный вектор и возвращающая объект из пространства исходных (тренировочных) данных. Вектор z называется *латентным*, скрытым представлением реальных данных. Генератор, иными словами, представляет собой отображение из латентного пространства в пространство реальных данных.

Дискриминатор решает задачу бинарной классификации, поэтому целесообразно использовать в качестве функции потерь бинарную кросс-энтропию, хотя это и не единственный вариант. Тогда для дискриминатора функция потерь будет представляться в виде суммы следующих величин:

$$\begin{aligned} Loss_{D_{real}} &= -y_{real} \cdot \log D(x) - (1 - y_{real}) \cdot \log(1 - D(x), \\ Loss_{D_{fake}} &= -y_{fake} \cdot \log D(G(z)) - (1 - y_{fake}) \cdot \log(1 - D(G(z))). \end{aligned}$$

где $y_{real} = 1$, $y_{fake} = 0$ — метки классов (реальное или сгенерированное соответственно). Тогда функция потерь дискриминатора равна:

$$Loss_D = -\log D(x) - \log(1 - D(G(z))).$$

При обучении генеративно-состязательных нейронных сетей дискриминатор является критерием качества генератора, поскольку генератор старается создать объект так, чтобы дискриминатор классифицировал его как реальный объект, то есть чтобы $D(G(z)) = 1$. Исходя из формулы для $Loss_{D_{fake}}$, положив в данном случае $y_{fake} = 1$ (к этому и стремиться генератор), получим функцию потерь для генератора:

$$Loss_G = -\log D(G(z)).$$

Алгоритм обучения генеративно-состязательных нейронных сетей состоит из двух этапов:

— Обучаем дискриминатор. Параметры генератора не обновляются. На вход дискриминатору подаются реальные и сгенерированные генератором данные. Вычисляем функцию потерь $Loss_D$. С помощью метода обратного распространения ошибки изменяем параметры дискриминатора.

— Обучаем генератор. Параметры дискриминатора не обновляются. Сгенерированные данные подаются на вход дискриминатору. Вычисляем функцию потерь $Loss_G$. С помощью метода обратного распространения ошибки изменяем параметры генератора.

1.1.3 Проблемы генеративно-состязательных моделей

Использование генеративно-состязательных нейронных сетей на практике бывает довольно сложно. Это связано со следующими проблемами, возникающими при обучении:

— **Коллапс модели** (**коллапс мод**). Реальные данные являются мультимодальными. Коллапс мод происходит, когда генератору удаётся создать некоторое количество данных, которые дискриминатор классифицирует неверно. Такое может произойти, например, когда темп обучения генератора выше, чем дискриминатора. В таком случае распределение сгенерированных изображений становится унимодальным, то есть все сгенерированные изображения становятся крайне похожи друг на друга. Как следствие — затухание градиентов дискриминатора.

— **Затухание и взрыв градиентов**. Может произойти так, что дискриминатор станет слишком хорошо отличать сгенерированные данные от настоящих. Тогда градиент генератора начнёт затухать и обучение перестанет происходить. Может возникнуть и обратная ситуация, когда генератор воспроизведёт очень качественные данные и градиент для дискриминатора может «взорваться».

— **Проблема стабильности обучения**. Задача обучения генератора и дискриминатора — это, в общем случае, не поиск минимума функции потерь, а поиск равновесия по Нэшу (поскольку имеем минимаксную игру двух игроков) [1]. В силу чего функции потерь зачастую ведут себя не монотонно. Значения могут постепенно стабилизироваться, но часто генеративно-состязательным моделям свойственны неконтролируемые

колебания значений функций потерь в широком диапазоне. Для решения этой проблемы предлагается использовать особые функции потерь, например, функцию потерь Вассерштейна [2] или Hinge Loss [3], а также некоторые другие эвристики.

— **Плохая интерпретируемость значений функций потерь**. В силу того, что функции потерь генератора и дискриминатора противоречат друг другу в том смысле, что они не могут быть минимизированы одновременно, значения функций потерь при стабильном обучении немонотонны и колеблются в некотором диапазоне. Из-за этого трудно сделать вывод о том, улучшается ли в процессе обучения качество модели в целом. Стоит отметить, что если одна из функций потерь находится около нуля, то модель перестаёт учиться и качество, скорее всего, будет только ухудшаться, поскольку это является признаком затухания или взрыва градиентов, коллапса мод. В таком случае значения функции потерь могут быть индикатором того, что продолжать обучение нецелесообразно.

— **Большая чувствительность к гиперпараметрам**. В генеративно - состязательных нейронных сетях используется большое количество гиперпараметров, таких как количество слоёв, параметры функций активации, размерность латентного пространства; кроме того, в цикле обучения присутствуют свои настраиваемые параметры, такие как темп обучения, размер пакетов (batch size), параметры алгоритма оптимизатора. Такие сети крайне чувствительны к небольшим изменениям каждого из этих параметров, поэтому необходимо глубоко понимать устройство данных моделей, чтобы грамотно корректировать гиперпараметры для решаемой задачи.

1.2 Свёрточные генеративно-состязательные нейронные сети. Архитектура Deep Convolutional GAN

Свёрточные генеративно-состязательные нейронные сети (Deep Convolutional Generative Adversarial Networks, DC-GAN) являются реализацией идеи GAN, в основном применяющиеся для генерации изображений. Модель строится с использованием свёрточных слоёв с постепенным изменением размерности вместо многослойного перцептрона, как в ориги-

нальной статье [1]. Данная архитектура была описана в статье «Unsupervised Representation Learning With Deep Convolutional Generative Adversarial Networks» [4].

Дискриминатор представляет из себя бинарный классификатор, принимающий на вход изображение и возвращающий скалярную величину. Модель состоит из нескольких блоков (слоёв), в каждом из которых есть слои свёртки (за счёт которых понижается размерность вплоть до единицы — единственного числа), batch-нормализации и активационных функций LeakyReLU. В самом конце в дискриминаторе находится функция активации сигмоида, которая переводит выход дискриминатора в величину из интервала $(0, 1)$.

Генератор отображает вектор из латентного пространства в пространство данных, в данном случае — в изображение из исходного распределения. Для этого применяются двумерные транспонированные свёрточные слои (вместо этого можно использовать обычные слои свёртки вместе со слоями, увеличивающими размерность в два раза на очередном шаге с помощью интерполяции), за каждым из которых следует batch-нормализация и активационная функция ReLU (рисунок 1.1). В конце генератора находится функция активации Tanh для того, чтобы результат был в интервале $(-1, 1)$, как и нормированное изображение.

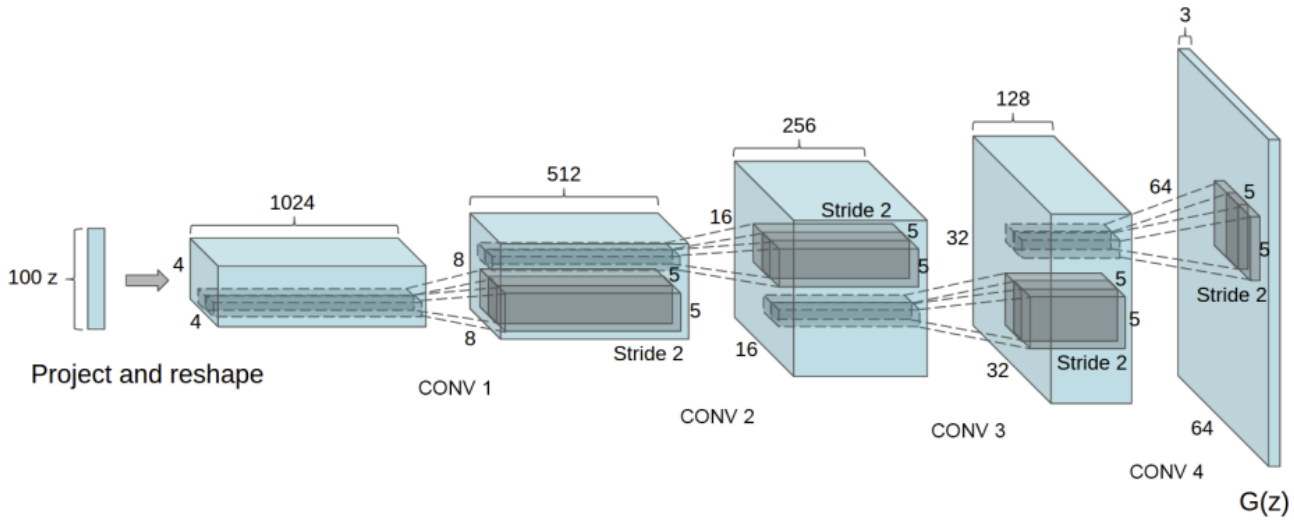


Рисунок 1.1 – Архитектура генератора

Для дополнительного обеспечения стабильности процесса тренировки, используется оптимизатор Adam (вместо SGD).

DC-GAN за счёт вышеперечисленных архитектурных изменений позволяет генерировать изображения сравнительно высокого качества, более того, обучение обладает большей стабильностью.

Существенным недостатком является то, что при большем разрешении исходных изображений коллапс мод становится более вероятным, поскольку

дискриминатору становится проще отличить реальные изображения от сгенерированных. Генерация изображений больших размерностей (выше 512×512) всё ещё остаётся сложной задачей для DC-GAN.

В данной главе рассмотрена идея генеративно-состязательных нейронных сетей, их алгоритм обучения, а также описаны следующие проблемы, возникающие при обучении таких сетей: коллапс мод, затухание и взрыв градиентов, проблема стабильности обучения, плохая интерпретируемость значений функций потерь, большая чувствительность к гиперпараметрам. Также была изучена и описана архитектура свёрточной генеративно-состязательной нейронной сети Deep Convolutional GAN, необходимая для построения модели и дальнейшего проведения экспериментов.

ГЛАВА 2

СВЁРТОЧНЫЕ ГЕНЕРАТИВНО-СОСТЯЗАТЕЛЬНЫЕ СЕТИ С БЛОКАМИ САМОВНИМАНИЯ

Свёрточные генеративно-состязательные сети были весьма успешными в задаче генерации изображений [4]. Однако, внимательно изучая образцы, сгенерированные такими моделями, можно убедиться в том, что свёрточным генеративно-состязательным сетям гораздо сложнее моделировать некоторые классы изображений. Например, такие модели отлично справляются с генерацией тех изображений, у которых есть лишь небольшие структурные ограничения (например, изображения неба, моря или ландшафта), иными словами, они хорошо генерируют те изображения, где большую часть занимает некоторая текстура. Однако этим моделям бывает трудно воспроизвести геометрические и структурные закономерности (например, сгенерированные изображения собак имеют весьма реалистичную текстуру шерсти, но неправильным количеством или расположением ног).

Одним из возможных объяснений этому является то, что свёрточные генеративно-состязательные модели в значительной степени опираются на свертку для моделирования зависимостей между различными областями изображения. Поскольку оператор свертки имеет локальную структуру, дальние зависимости (зависимости между отдалёнными участками изображения) могут быть обработаны только после прохождения через несколько свёрточных слоев.

Это может помешать изучению долгосрочных зависимостей по целому ряду причин: маленькая модель может не справиться с их представлением, алгоритмы оптимизации могут столкнуться с трудностями при поиске значений параметров, которые тщательно координируют работу нескольких слоев, чтобы уловить эти зависимости, и эти параметризации могут быть статистически хрупкими и склонными к сбоям при применении к ранее не виденным входным данным. Увеличение размера ядер свертки может повысить репрезентативную способность сети, но при этом теряется вычислительная и статистическая эффективность, достигаемая за счет использования локальной свёрточной структуры [5].

Блоки самовнимания, с другой стороны, демонстрирует лучший баланс между способностью моделировать дальние зависимости (зависимости между отдалёнными участками изображения) и вычислительной и статистической

эффективностью. Блоки самовнимания вычисляют реакцию в одной позиции как взвешенную сумму признаков во всех позициях, где веса, или векторы внимания, вычисляются с небольшими вычислительными затратами [5]. Таким образом, имеет смысл интегрировать блоки самовнимания в архитектуру DC-GAN.

2.1 Архитектура Self-Attention GAN

Опишем архитектуру свёрточной генеративно-состязательной нейронной сети с использованием блоков самовнимания (Self-Attention GAN).

Основным архитектурным изменением является добавление блоков самовнимания в архитектуру DC-GAN, причём как к генератору, так и к дискриминатору. Блоки самовнимания (рисунок 2.1) располагаются перед очередным свёрточным слоем, причём, как правило, во второй половине сети, то есть перед теми свёртками, которые детектируют высокоуровневые паттерны, то есть к картам активации высокой размерности. Это связано с тем, что блоки самовнимания моделируют зависимости между отдалёнными участками изображения (карт активации), поэтому имеет смысл применять их к картам активации высокой размерности, которые получаются на выходе свёрток во второй половине сети.

Блоки самовнимания состоят из одномерных свёрток. Карты активации с предыдущего скрытого слоя попадают в блок самовнимания и преобразуются в два пространства признаков f и g. Далее, применяя матричное произведение к полученным картам активации и функции softmax получим карты активации, отвечающими за внимание. Преобразовав исходные карты активации с помощью одномерных свёрток h, к ним, с помощью матричного произведения, добавляется информация о картах активации, при этом карты внимания умножаются на обучаемый параметр γ, который регулирует влияние карт внимания на исходные карты активации. В самом конце применяются ещё одни одномерные свёртки v и на выходе получаются карты активации блока самовнимания.

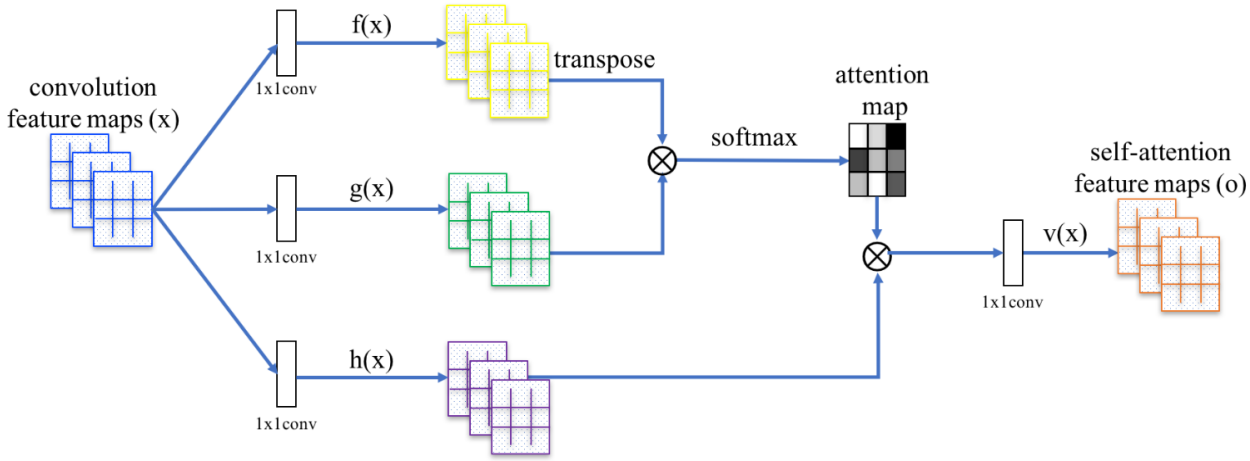


Рисунок 2.1 – Архитектура блока самовнимания

На рисунке 2.1 знак $\otimes$ означает матричное произведение. Функция softmax применяется к матрицам построчно.

Ещё одним важным усовершенствованием архитектуры является применение помимо *batch*-нормализации спектральной нормализации. Такая нормализация стабилизирует обучение за счёт контролирования константы Липшица (после спектральной нормализации константа Липшица становится равна 1) и не имеет настраиваемых гиперпараметров, более того, не требует много вычислительных мощностей [6]. Спектральная нормализация заключается в том, что каждая карта активации на очередном слое, представляющая собой матрицу, делится на спектральную норму этой матрицы. Спектральная норма матрицы равна наибольшему собственному значению этой матрицы [6].

На практике, для подсчёта наибольшего собственного значения используется степенной метод, причём одна итерация, поскольку такой точности бывает вполне достаточно. Таким образом, спектральная нормализация не является вычислительно сложной относительно всего процесса обучения сети. В генераторе также используется спектральная нормализация, поскольку она позволяет предотвратить рост значений параметров и избежать необычных градиентов. Стоит отметить, что эмпирически было обнаружено, что спектральная нормализация генератора и дискриминатора позволяет использовать меньшее количество обновлений дискриминатора на одно обновление генератора, что значительно снижает вычислительные затраты на обучение [5].

2.2 Процесс обучения Self-Attention GAN

Алгоритм обучения Self-Attention GAN схож с алгоритмом обучения DCGAN. Однако предлагается заменить функцию потерь на hinge-loss. Теперь

функции потерь для дискриминатора и генератора соответственно выглядят так:

$$Loss_D = -\min(0, -1 + D(x)) - \min(0, -1 - D(G(z))),$$
$$Loss_G = -D(G(z)).$$

Также предлагается изменить темп обучения сетей, а именно воспользоваться правилом TTUR (Two time-scale update rule) [14]. Регуляризация дискриминатора часто замедляет процесс обучения GAN. На практике методы, использующие регуляризацию дискриминатора, обычно требуют нескольких шагов обновления дискриминатора на каждый шаг обновления генератора в процессе обучения. TTUR предлагает использовать отдельные темпы обучения для генератора и дискриминатора для компенсации проблемы медленного обучения при регуляризации дискриминатора, что позволяет использовать меньшее количество шагов дискриминатора на шаг генератора. Используя этот подход можно получить лучшие результаты при том же времени работы [5].

В данной главе рассмотрена идея и архитектура блока самовнимания. Изучив свёрточные генеративно-состязательные сети, их недостатки в генерации изображений было выяснено, что целесообразно интегрировать блоки самовнимания в архитектуру DC-GAN. Описана архитектура и процесс обучения свёрточной генеративно-состязательной нейронной сети с блоками самовнимания Self-Attention GAN. В алгоритме обучения Self-Attention GAN, схожим с алгоритмом обучения DC-GAN, предлагается заменить функцию потерь на Hinge-loss и изменить темп обучения сетей, а именно: воспользоваться правилом TTUR (Two time-scale update rule). В результате, основным архитектурным изменением становится добавление блоков самовнимания в архитектуру DC-GAN как к генератору, так и к дискриминатору.

ГЛАВА 3

ДВУХЭТАПНАЯ УСЛОВНАЯ И БЕЗУСЛОВНАЯ ГЕНЕРАЦИЯ. CONDITIONAL GAN

У генеративно-состязательных моделей, использующих свёрточные слои, зачастую возникают проблемы при генерации изображений со сложной пространственно-геометрической структурой, которая исключительно важна при генерации медицинских изображений, поскольку натуральность изображения определяется в том числе анатомически верным расположением органов и тканей. В то же время архитектуры, подобные DC-GAN, хорошо воспроизводят локальную структуру (текстуры) изображения. Исходя из этого, можно попытаться улучшить DC-GAN-подобные архитектуры, используя информацию о пространственно-геометрической структуре генерируемого изображения.

Предлагается разделить генерацию на два этапа:

— Первый этап. Преобразовать случайный (латентный) вектор не в готовое изображение, а в некоторую осмысленную пространственно-геометрическую информацию (она может быть, вообще говоря, в любом виде), которая уже на втором этапе будет преобразована в готовое изображение.

— Второй этап. Используя сгенерированную на предыдущем этапе пространственно-геометрическую информацию, создать готовое изображение.

Стоит отметить, что для каждого этапа может использоваться отдельная модель, наиболее подходящая для конкретной задачи. Более того, для определённого вида пространственно-геометрической информации можно подобрать наиболее подходящую модель. Для второго этапа можно использовать DC-GAN-подобную модель, поскольку, обладая достаточным количеством пространственно-геометрической информации, останется лишь правильно воспроизвести локальную структуру (текстуру) изображения.

3.1 Этапы генерации рентгеновского снимка

Первый этап генерации — сгенерировать маску, или же сегментацию, рентгена лёгких. Формально, это задача генерации изображения. При этом данные имеют весьма специфическую, в некотором смысле простую структуру: маска — бинарное изображение, а лёгкие представляют из себя два белых пятна определённой формы на чёрном фоне (рисунок 3.1). Здесь нет сложных пространственных зависимостей и ощутимой разницы между изображениями, чего уже нельзя сказать о рентгеновских снимках.

Рисунок 3.1 – Примеры исходных масок лёгких

Предполагается, что с генерацией таких несложных с точки зрения пространственных зависимостей и текстур могут справиться и не самые сложные модели.

Второй этап генерации — преобразование маски лёгких в соответствующий итоговый рентгеновский снимок (рисунок 3.2). Формально, это задача преобразования одного изображения в другое (известная также как задача pix2pix).

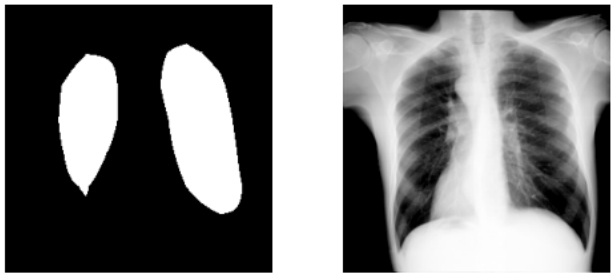


Рисунок 3.2 – Пример преобразования маски лёгких в рентгеновский снимок

Также заметим, что данная задача — обратная задаче сегментации изображения. На этом этапе необходимо на основе данной маски, являющейся изображением, которую можно интерпретировать как высокоуровневую пространственную информацию в виде матрицы, сгенерировать итоговое изображение путём наложения соответствующих текстур в нужные участки изображений. Отметим, что имея такую информацию, задача становится более

подходящей для DC-GAN-подобных архитектур, которые хорошо воссоздают локальные зависимости (текстуры).

3.2 Генерация сегментированного изображения (маски)

Первый этап генерации — создание маски лёгких. Как уже отмечалось ранее, изображение сегментации лёгких имеет довольно простую пространственно-геометрическую структуру, а значит с этой задачей может справиться даже не самая сложная модель.

Для осуществления генерации сегментации лёгких используем вариационный автокодировщик. Такая модель имеет более простую архитектуру, как правило, с меньшим числом параметров, нежели модели GAN, быстрее и стабильнее обучаются. Конечно, такие модели слабее DC-GAN-подобных архитектур в задачах генерации изображений, однако, в виду того, что изображения имеют простую структуру, предполагается, что вариационный автокодировщик хорошо справиться с этой задачей.

3.2.1 Архитектура вариационного автокодировщика

Вариационный автокодировщик (Variational Autoencoder, VAE) использует вариационный вывод для обучения модели. Это статистический подход, который позволяет оценить апостериорное распределение скрытых переменных, основываясь на наблюдаемых данных. Вариационный вывод позволяет эффективно обучать модель и справляться с высокоразмерными данными.

Вариационный автокодировщик состоит из двух блоков: кодировщик и декодировщик. Кодировщик преобразует исходное изображение в два вектора, соответствующие среднему и стандартному отклонению. Далее происходит «трюк репараметризации» (Reparameterization trick), благодаря которому формируется латентное представление входных данных. Принцип трюка состоит в том, что, приняв выход кодировщика за среднее и стандартное отклонение, можно взять латентный вектор из нормального распределения с этими параметрами.

Пусть μ, σ — выходы кодировщика, соответствующие среднему и нормальному отклонению некоторого нормального распределения. Тогда латентный вектор z можно получить по следующей формуле [7]:

$$z = \mu + \sigma \odot \varepsilon, \quad \varepsilon \sim \mathcal{N}(0, \mathbf{I})$$

И в кодировщике, и в декодировщике используются блоки, состоящие из двумерных свёрток (транспонированных для декодировщика), пакетной нормализации и функци активации ReLU. Перед получением параметров μ и σ и после получения латентного вектора z используется полносвязный линейный слой. В конце декодировщика стоит функция активации Tanh, нормализующая итоговое изображение в пределах от -1 до 1

Задача декодировщика состоит в том, чтобы из латентного представления z восстановить исходное изображение. В общем, вариационный автокодировщик стремится кодировать и декодировать изображение с минимальной потерей информации, то есть итоговое (восстановленное) изображение должно быть максимально схожим с входным.

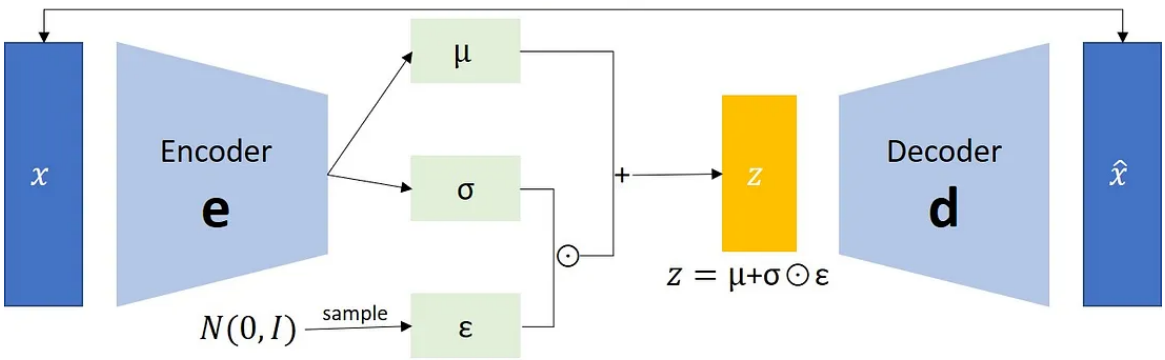


Рисунок 3.3 – Архитектура вариационного автокодировщика

3.2.2 Процесс обучения вариационного автокодирвщика

Алгоритм обучения весьма стандартный: сеть обучается посредством метода обратного распространения ошибки на каждом пакете данных. Функция потерь состоит из двух частей:

— KL-дивиргенция, или расхождение Кульбака-Лейблера. Эта часть функции потерь отвечает за близость распределения латентного вектора z к стандартному нормальному распределению. Это необходимо для последующего применения модели для генерации изображений. Так, подав на вход декодировщику вектор из нормального стандартного распределения, мы получим изображение из распределения тренировочных данных, поскольку модель была обучена восстанавливать исходные изображения из таких веткоров. Вычисляется следующим образом:

$$KL = -\frac{1}{2}\sum_{i=1}^{dimZ}(1 + \log(\sigma_i^2) - \mu_i^2 - \sigma_i^2),$$

где μ, σ — среднее и стандартное отклонение соответственно, получаемые на выходе кодировщика.

— Логарифм правдоподобия, вычисляемый как бинарная кросс-энтропия между исходным и полученным (реконструированным) изображением. Отвечает за качество восстановления (реконструкции) изображения. Вычисляется следующим образом:

$$Loss_{BCE} = y \log x + (1 - y) \log(1 - x),$$

где x — восстановленное изображение, y — исходное.

Общая функция потерь выглядит следующим образом:

$$Loss_{VAE} = Loss_{BCE} + KL.$$

3.3 Восстановление изображения по результату его сегментации

Второй этап генерации — восстановить изображение рентгена лёгких, исходя из информации о его пространственно-геометрической структуре в виде маски, иными словами — решение задачи преобразования одного изображения в другое. Имея информацию о зависимости отдалённых друг от друга участков изображения, применение генеративных моделей, использующих свёрточные слои, оправдано и целесообразно, поскольку необходимо восстановить лишь локальную структурную зависимость.

Для решения этой задачи используем условную генеративно-состязательную сеть (Conditional GAN). Задачу второго этапа генерации можно переформулировать как генерация по заданному условию в виде изображения сегментации. Эта модель хорошо показывает себя в задачах преобразования одного изображения в другое, достигая таких результатов за счёт архитектурных нововведений и модифицированного процесса обучения [8].

3.3.1 Архитектура Conditional GAN

Условная генеративно-состязательная сеть состоит из генератора и дискриминатора. Генератор имеет U-net-подобную архитектуру (рисунок 3.4). Такая архитектура имеет структуру кодировщик-декодировщик (сначала

изображение «сжимается», уменьшается размерность и общее количество элементов, а затем «разжимается» в новое изображение с теми же размерами, что и исходное) с использованием пропускающих соединений (skip-connections). Пропускающие соединения расположены между каждым слоем i и $n - i$, где n — общее количество слоёв. Такие соединения имеют ряд преимуществ:

— Решают проблему затухающих градиентов. Если сеть довольно глубока, значение градиентов по мере обратного распространения ошибки может сильно приблизиться к нулю, а значит начальные слои будут незначительно обновляться на очередной итерации, то есть слабо обучаться. При наличии таких соединений, градиент может быть посчитан по ним и за один шаг добраться из соответствующего слоя декодировщика в слой кодировщика. Решение этой проблемы улучшает процесс обучения.

— Доставляет дополнительную информацию об изначальной структуре изображения (до очередного шага кодирования), что существенно улучшает качество генерации.

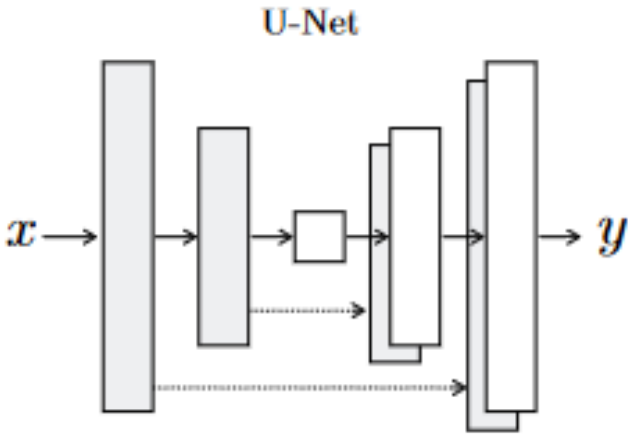


Рисунок 3.4 – Схема U-Net архитектуры

Каждый блок кодировщика состоит из слоя двумерных свёрток, пакетной нормализации и функции активации LeakyReLU. Каждый блок декодировщика аналогичен соответствующим блокам кодировщика, однако имеет транспонированные двумерные свёртки и функцию активации ReLU. В конце к выходу генератора применяется функция активации Tanh.

В силу того, что мы уже обладаем пространственно-геометрической информацией об итоговом изображении, стоит сконцентрироваться на моделировании локальной структуры изображения. Поэтому предлагается использовать архитектуру дискриминатора PatchGAN [8]. Такой дискриминатор на

выходе выдаёт не одно число, как в классических GAN архитектурах, а матрицу, каждый элемент которой — вердикт дискриминатора о том, является ли соответствующая часть изображения правдоподобной (реальной) или нет (сгенерированной). Он состоит из нескольких блоков, состоящих из двумерных свёрток, пакетной нормализации и функции активации LeakyReLU. В конце к итоговой матрице применяется функция активации Softmax, таким образом каждый элемент матрицы соответствует уверенности дискриминатора в том, насколько соответствующая часть изображения реальная. Такой дискриминатор эффективно моделирует изображение как марковское случайное поле, предполагая независимость между пикселями, разделёнными более чем заданным диаметром участка изображения. Таким образом, PatchGAN можно воспринимать как форму текстурной, стилевой функции потерь [8].

Отметим, что на вход CGAN подаётся только маска изображения, при этом случайный вектор в данной реализации отсутствует, поскольку модели VAE и CGAN работают в ансамбле, «элемент случайности» есть только в VAE, а задача CGAN — просто восстановить изображение по некоторой сгенерированной маске.

3.3.2 Процесс обучения Conditional GAN

Процесс обучения CGAN такой же, как процесс обучения классической GAN модели. Функция потерь для генератора состоит из двух слагаемых:

— Бинарная кросс-энтропия между выходом дискриминатора по сгенерированным изображениям и желаемыми метками класса (класс реальных изображений имеет метку 1, а цель генератора состоит в том, чтобы дискриминатор считал сгенерированные изображения реальными). Эта функция потерь показывает, насколько успешен генератор в обмане дискриминатора. Формула была описана ранее.

— L1-функция потерь, представляющая из себя L1-расстояние между сгенерированным и реальным изображением:

$$Loss_{L1} = |x - y|,$$

где x — сгенерированное изображение, а y — реальное. Это заставляет модель генерировать правдоподобные и близкие к реальным изображения в смысле L1-расстояния. Использование L1-расстояния повышает чёткость, резкость изображения [8].

Таким образом, функция потерь генератора выглядит следующим образом:

$$Loss_G = Loss_{BCE} + \lambda Loss_{L1},$$

где λ — гиперпараметр, регулирующий влияние L1-функции потерь.

Функция потерь дискриминатора представляет из себя бинарную кросс-энтропию, определяемую так же, как и ранее для генеративно-состязательных сетей.

В данной главе предложена идея разделения процесса генерации изображения на два этапа и ансамблирования генеративных моделей для условной и безусловной генерации изображений. Опираясь на выявленные недостатки свёрточных слоев, мы предлагаем разделить процесс генерации изображения на два этапа: преобразовать случайный (латентный) вектор не в готовое изображение, а в некоторую осмысленную пространственно-геометрическую информацию, а затем, используя сгенерированную на предыдущем этапе пространственно-геометрическую информацию, создать готовое изображение. Также нами был проведен анализ ансамблирования генеративных моделей для условной и безусловной генерации изображений. Описана архитектура и процесс обучения вариационного автокодировщика и Conditional GAN при их ансамблировании.

ГЛАВА 4

АРХИТЕКТУРА BIG-GAN-DEEP ДЛЯ ГЕНЕРАЦИИ ИЗОБРАЖЕНИЙ ВЫСОКОЙ РАЗМЕРНОСТИ

Часто подходы, которые хорошо справляются с генерацией изображений невысокой размерности, плохо справляются с генерацией изображений высокого разрешения. В частности, использование одинаковых конфигураций гиперпараметров и архитектур нейронных генеративно-состязательных сетей для одних и тех же данных разной размерности не гарантирует сходимость метода: для данных низкой размерности обучение может быть стабильным, но стоит повысить размерность, как модель становится подверженной коллапсу мод, взрыву и затуханию градиентов в процессе обучения. Для решения проблемы генерации изображений высокого разрешения была создана архитектура BIG-GAN и BIG-GAN-DEEP [9].

4.1 Архитектура BIG-GAN-DEEP

Будем использовать более глубокую версию архитектуры BIG-GAN — BIG-GAN-DEEP. Для последующих экспериментов будем использовать безусловную архитектуру (наши данные не будут разделены на классы, поэтому в модели не будет классового условия).

Архитектура BIG-GAN-DEEP представляет собой классическую свёрточную генеративно-состязательную нейронную сеть с генератором и дискриминатором, имеющая лишь архитектурные отличия в строении. Опишем архитектуру для генерации изображений размера 256×256 более подробно в виде таблиц (для генератора (a) и дискриминатора (b), таблица 4.1). «Non-Local Block» — то же самое, что и Attention-Block. Применяется как в генераторе, так и в дискриминаторе, после карт активации размерности 64×64.

Таблица 4.1 – Архитектура генератора (a) и дискриминатора (b)

$z \sim \mathcal{N}^n(0,1)$
Linear (128) $\to 4 \times 4 \times 16ch$
ResBlock $16ch \to 16ch$
ResBlock up $16ch \to 16ch$
ResBlock $16ch \to 16ch$
ResBlock up $16ch \to 8ch$
ResBlock $8ch \to 8ch$
ResBlock up $8ch \to 8ch$
ResBlock $8ch \to 8ch$
ResBlock up $8ch \to 4ch$
Non-Local Block (64×64)
ResBlock $4ch \to 4ch$
ResBlock up $4ch \to 2ch$
ResBlock $2ch \to 2ch$
ResBlock up $2ch \to 1ch$
BN, ReLU, 3×3 Conv $1ch \to 1$
Tanh

(a)

image 256×256
3×3 Conv $1 \to 1ch$
ResBlock down $ch \to 2ch$
ResBlock $2ch \to 2ch$
ResBlock down $2ch \to 4ch$
ResBlock $4ch \to 4ch$
Non-Local Block (64×64)
ResBlock down $4ch \to 8ch$
ResBlock $8ch \to 8ch$
ResBlock down $8ch \to 16ch$
ResBlock $16 \to 16$
ResBlock down $16ch \to 16ch$
ResBlock $16ch \to 16ch$
ReLU, Global sum pooling
Linear $16ch \to 1$

(b)

В этой архитектуре последовательно применяются ResBlock — residual-блоки с «бутылочным горлышком» (применяется последовательность свёрток $1 \times 1,\ 3 \times 3,\ 3 \times 3,\ 1 \times 1$), которые включают в себя две дополнительные свертки 1×1: первая сокращает количество каналов в 4 раза перед более дорогостоящей сверткой 3×3; вторая позволяет получить требуемое количество выходных каналов [9]. Пометка «up» или «down» означает блоки, которые повышают и понижают соответственно размерности карт активации в два раза. В блоке «ResBlock up» используется интерполяция по ближайшим соседям для повышения размерности, а в блоке «ResBlock down» используется Average-pooling для понижения. Стоит отметить, что всюду после свёрток используется спектральная нормализация.

4.2 Процесс обучения BIG-GAN-DEEP

Процесс обучения BIG-GAN-DEEP такой же, как и для модели Self-Attention GAN. Предлагается использовать hinge-loss, TTUR.

4.3 Градиентная регуляризация для стабилизации обучения

Для стабилизации обучения генеративно-состязательных сетей существует множество подходов. Применим один из лучших методов стабилизации обучения — градиентная регуляризация дискриминатора [10,11]. Предлагается к функции потерь дискриминатора добавить новое слагаемое, тем самым наложить условие на градиенты выходов дискриминатора:

$$\frac{\gamma}{2}\text{mean}\left((1-\sigma(\varphi))^2||\Delta\varphi||^2+\sigma^2(\psi)||\Delta\psi||^2\right),$$

где φ — выходы дискриминатора от реальных изображений, ψ — выходы дискриминатора от сгенерированных изображений, σ — сигмоида. Данная регуляризация позволяет минимизировать дивергенцию между исходным и аппроксимированным с помощью генератора распределениями, за счёт чего и стабилизируется обучение [11].

В данной главе рассмотрена проблема генерации изображений высокого разрешения. Для решения этой проблемы нами была выбрана и рассмотрена архитектура BIG-GAN и BIG-GAN-DEEP. Для стабилизации обучения генеративно-состязательных сетей определен и описан один из лучших методов стабилизации обучения — градиентная регуляризация дискриминатора.

ГЛАВА 5

АРХИТЕКТУРА SPADE-GAN ДЛЯ УСЛОВНОЙ ГЕНЕРАЦИИ ИЗОБРАЖЕНИЙ ВЫСОКОЙ РАЗМЕРНОСТИ

У генеративно-состязательных моделей, использующих свёрточные слои, зачастую возникают проблемы при генерации изображений со сложной пространственно-геометрической структурой, которая исключительно важна при генерации медицинских изображений, поскольку натуральность изображения определяется в том числе анатомически верным расположением органов и тканей. В то же время архитектуры, подобные DC-GAN, хорошо воспроизводят локальную структуру (текстуры) изображения. Исходя из этого, можно попытаться улучшить DC-GAN-подобные архитектуры, используя каким-то образом полученную информацию о пространственно-геометрической структуре генерируемого изображения.

Одним из способов улучшить пространственно-геометрическую структуру генерируемого изображения является передача информации о расположении и формах объектов в сеть в явном виде. Сделать это можно, например, используя семантическую маску изображения или любое другое изображение, содержащее в себе пространственную информацию (например, границы объектов); также можно использовать сразу несколько изображений, содержащих в себе разнородную и разноуровневую пространственную информацию. Тогда задача сети будет состоять в том, чтобы правильно заполнить некоторыми текстурами определённые в семантической маске или контурами границ объектов области. Такую задачу можно трактовать как преобразование одного или группы изображений, содержащих в себе пространственную информацию, в другое, или как условную генерацию, где в качестве условия выступает пространственно-геометрическая информация.

Одной из лучших архитектур, созданных для решения этой задачи, является SPADE-GAN [12]. SPADE-GAN является улучшенной версией архитектуры pix2pixHD [13] во многом за счёт использования SPADE-блоков (spatially-adaptive denormalization блоков), которые учитывают пространственную информацию подаваемую в виде, например, семантической маски.

5.1 Архитектура SPADE-GAN

SPADE-GAN состоит из трёх основных частей: генератор, энкодер и дискриминатор. Энкодер и генератор формируют структуру вариационного автокодировщика (VAE).

Дискриминатор включается в себя три дискриминатора типа PatchGAN [8], каждый из которых работает с изображениями разных размерностей. Изображение, подаваемое на вход дискриминатору, уменьшается в два и четыре раза, таким образом получим три изображения разного размера, и каждое из этих изображений подаётся в соответствующий дискриминатор PatchGAN. PatchGAN дискриминатор на выходе выдаёт не одно число, означающее уверенность в том, что изображение настоящее, как в классических GAN архитектурах, а матрицу, каждый элемент которой — вердикт дискриминатора о том, является ли соответствующая часть изображения (патч) правдоподобной или нет. Он состоит из нескольких блоков, состоящих из двумерных свёрток, поканальной нормализации Instance Norm и функции активации LeakyReLU. В конце к итоговой матрице применяется функция активации Softmax, таким образом каждый элемент матрицы соответствует уверенности дискриминатора в том, насколько соответствующая часть изображения правдоподобна. Такой дискриминатор эффективно моделирует изображение как марковское случайное поле, предполагая независимость между пикселями, разделёнными более чем заданным диаметром участка изображения. Таким образом, PatchGAN можно воспринимать как форму текстурной, стилевой функции потерь [8].

Энкодер представляет собой несколько слоёв двумерных свёрток с нормализацией Instance Norm. Далее следует два независимых (параллельных) линейных полносвязных слоя, которые, в соответствии с архитектурой вариационного автокодировщика, возвращают среднее и стандартное отклонение соответственно, с помощью которых получается латентный вектор путём параметризации нормального распределения полученными параметрами. Далее этот вектор отправляется на вход генератору.

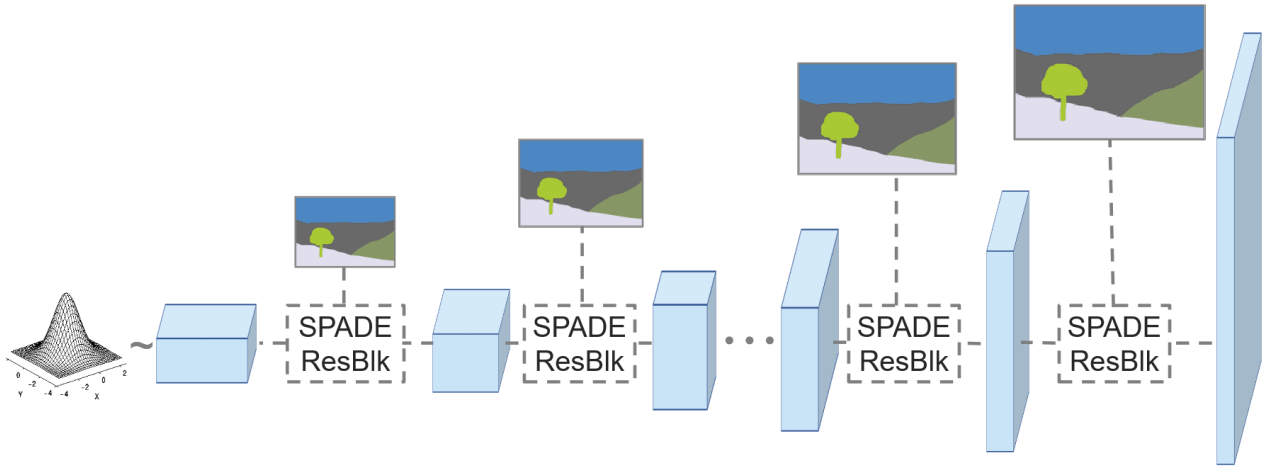


Рисунок 5.1 – Схема архитектуры генератора

Генератор (рисунок 5.1) состоит из нескольких подряд идущих SPADE ResBlk блоков с «пропущенными соединениями» (skip connections) (рисунок 5.2), в каждом из которых дважды применяется SPADE блок, активация (ReLU) и двумерная свёртка. Отметим, что в таких блоках не применяется какая-либо нормализация.

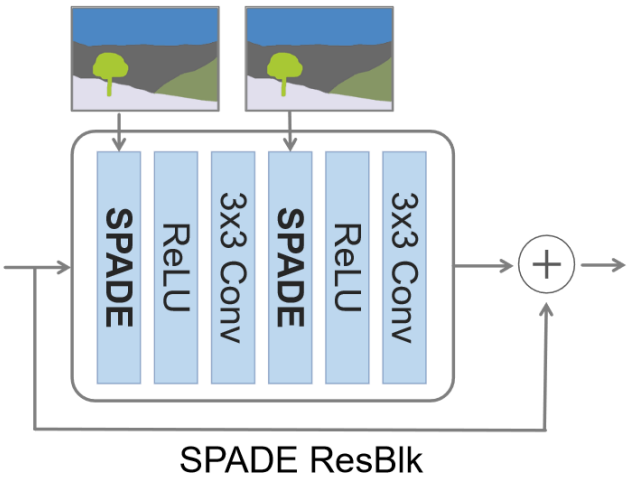


Рисунок 5.2 – Схема архитектуры блока SPADE ResBlk

SPADE блок (рисунок 5.3) представляет из себя денормализационный блок, который применяется после некоторого нормировочного слоя (например, после пакетной нормализации) для того, чтобы денормализовать его выход. На вход этому блоку подаётся семантическая маска (или любая другая пространственная информация в виде изображения), которая пропускается через несколько свёрточных слоёв, в результате чего получаются два параметра: смещение (β) и разброс (γ). Далее происходит денормализация: карты активации после выхода очередного слоя умножаются на γ («растягиваются» или «сужаются») и складываются с β (смещаются). Эти параметры

получаются после обработки изображения, содержащего пространственную информацию, и эта информация поступает в каждый блок генератора, что позволяет генератору учесть и «не забыть» как расположены и какую форму имеют объекты на генерируемом изображении.

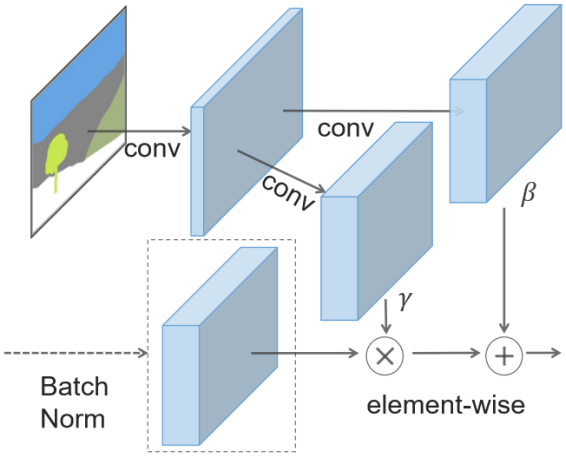


Рисунок 5.3 – Схема архитектуры блока SPADE

SPADE сохраняет семантическую и пространственную информацию об объектах на изображении, в то время как слои нормализации Batch Norm или Instance Norm имеют свойство стирать такую информацию [12].

5.2 Процесс обучения SPADE-GAN

Функция потерь генератора состоит из трёх слагаемых. Первым слагаемым является hinge loss [3]. Вторым является KL-дивергенция, чтобы выходы энкодера соответствовали параметрам стандартного распределения. Третьим слагаемым является L1-расстояние между выходами каждого из слоёв каждого из дискриминаторов PatchGAN на исходных и сгенерированных изображениях соответственно. В процессе обучения энкодеру подаётся на вход реальное изображение, далее в результате работы энкодера формируется латентный вектор, соответствующий этому изображению и подаётся на вход генератору вместе с семантической маской (для SPADE блоков). Таким образом, задачей генератора в процессе обучения является восстановление исходного изображения по его латентному вектору и семантической маске.

Функцией потерь дискриминатора является hinge loss [3]. Задачей дискриминатора, как и в любом GAN, является успешная классификация реальных и сгенерированных изображений.

5.3 Применение SPADE-GAN

Описанная выше архитектура позволяет генерировать изображение двумя способами. Первый способ заключается в генерации только по семантической маске, то есть на вход генератору подаётся латентый вектор (из стандартного многомерного распределния) и некоторая семантическая маска, которой должно соответствовать сгенерированное изображение. Второй способ — генерация по некоторому исходному изображению (исходное изображение, на которое должно быть похоже сгенерированное) и некоторой семантической маске (которой должно соответствовать сгенерированное изображение с точки зрения расположения и формы объектов).

Основным преимуществом данной модели, помимо явного учёта пространственно-геометрической информации, а также о типе (классе) объектов, является контролируемая генерация. В частности, такая модель способна генерировать изображения с заранее заданным числом объектов, их классом и расположении. Основным применением данной модели может быть классовая аугментация, когда число классов не сбалансировано. При этом не придётся генерировать неопределённое количество случайных изображений, а сгенерировать конкретное число изображений каждого класса.

В данной главе изложена архитектура и процесс обучения модели SPADE-GAN для условной генерации изображений высокой размерности. Описана архитектура денормализационного блока SPADE (spatially-adaptive denotmalization блок), который учитывает пространственную информацию, подаваемую в виде семантической маски. Преимуществом данной модели, является контролируемая генерация. Такая модель способна генерировать изображения с заранее заданным числом объектов, их классом и расположением. Это дает возможность применять данную модель для расширения (аугментации) данных в условиях несбалансированных классов. При этом достаточно сгенерировать конкретное необходимое число изображений нужного класса.

ГЛАВА 6

ЭКСПЕРИМЕНТЫ И РЕЗУЛЬТАТЫ

Прежде чем проводить сравнительный анализ вышеперечисленных архитектур генеративных нейронных сетей, стоит отметить, что в нашей работе архитектуры DC-GAN, SA-GAN и BIG-GAN-DEEP являются безусловными генеративными моделями, то есть наложить некоторое условие или ограничение на результат генерации нельзя; SPADE-GAN является условной, при этом условием выступает некоторое изображение, содержащее пространственную информацию, с учётом которой нужно произвести генерацию. Отдельно стоит отметить ансамбль генеративных моделей VAE и CGAN, ведь вариационный автокодировщик способен как сгенерировать маску из случайного вектора, так и преобразовать данную маску, то есть ансамбль способен генерировать как условно, так и безусловно. При этом, генерация по заданной маске не будет иметь случайный характер, то есть выход модели для фиксированной маски будет фиксированным, поэтому генерация с условием в данном случае является не генерацией, а восстановлением изображения по некоторой маске. Безусловным генеративным моделям не нужны размеченные данные, но нужны условным, где в качестве разметки будет выступать сегментация некоторых объектов на изображении. Поэтому, для проведения сравнительного анализа было выбрано два набора данных.

Первый набор данных представляет собой порядка 32.000 изображений рентгеновских снимков людей разных полов и возрастов с сегментированными лёгкими (рисунок 6.1). Изображения являются одноканальными с разрешением 512×512, сегментация лёгких представляет собой бинарное одноканальное изображение, маску, где 1 означает лёгкое и 0 — всё остальное. Каждое из изображений сжималось до разрешения 256×256 с помощью бикубической интерполяции, таким образом все модели использовались для генерации изображений с разрешением 256×256. Безусловные генеративные модели обучались на этом наборе без использования масок, а условные, соответственно, с их использованием.

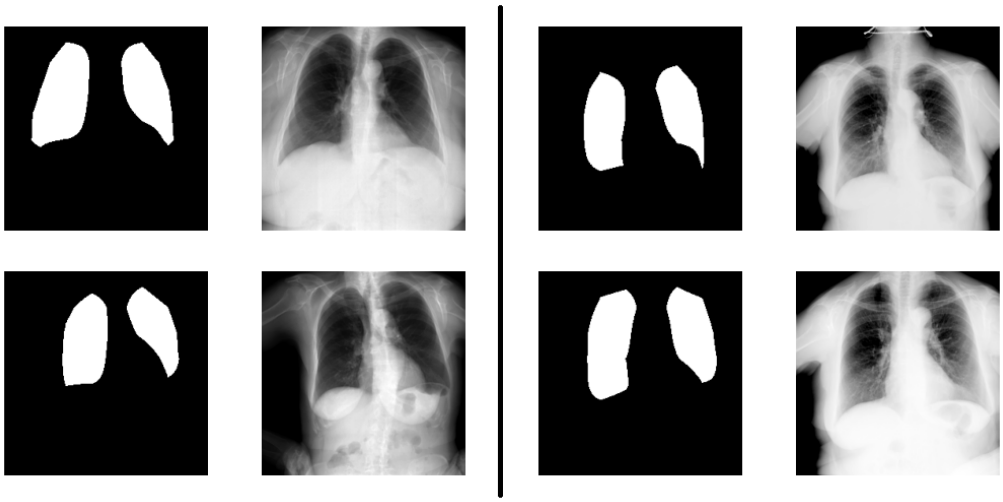


Рисунок 6.1 – Пример исходных изображений набора рентгеновских снимков

Второй набор данных представляет собой 60.000 изображений КТ-срезов людей разных полов и возрастов (рисунок 6.2). Изображения также являются одноканальными с разрешением 512×512, которые сжимаются до разрешения 256×256 с помощью бикубической интерполяции. В силу того, что этот набор данных не размечен, он использовался для проведения сравнительного анализа только безусловных архитектур.

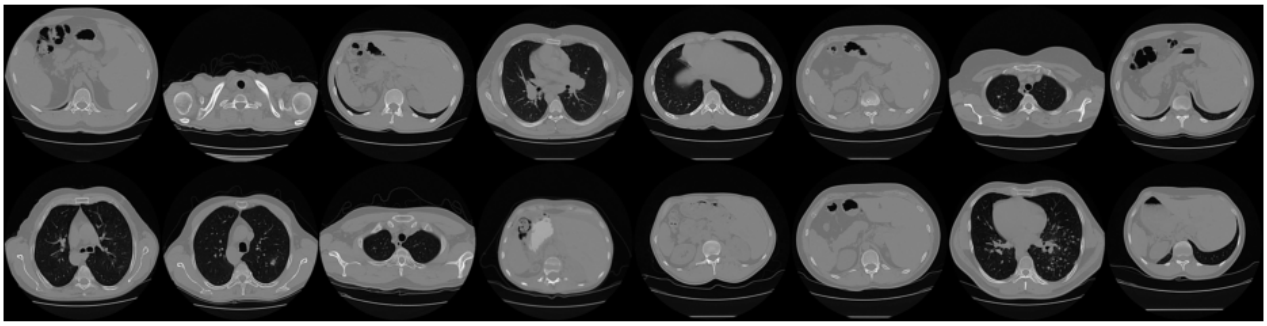


Рисунок 6.2 – Пример исходных изображений набора КТ-снимков

Оценка качества сгенерированных изображений количественно оценивалась с помощью метрики FID [14]. Чем меньше её значение, тем ближе сгенерированные изображения к исходным в смысле их распределений. Метрика FID рассчитывалась между всеми исходными изображениями и таким же количеством сгенерированных.

Каждая из моделей обучалась с использованием оптимизационного алгоритма Adam [15].

Ссылка на репозиторий github с исходным кодом реализации вышеописанных архитектур и циклов обучения находится в Приложении А.

6.1 Применение архитектуры DC-GAN

Использование модели DC-GAN с транспонированными свёртками не приводило к достойному результату. Модель либо не обучалась вовсе, из-за непропорциональной силы дискриминатора относительно генератора, либо крайне быстро коллапсировала. Стоит отметить, что такая модель имеет в несколько раз большее число параметров, нежели модель без использования таких свёрток, в силу чего её обучение гораздо более вычислительно затратное. Далее описываются эксперименты и результаты модели DC-GAN с использованием обычных свёрток с ядром размера 3×3 вместе со слоями, увеличивающими размерность в два раза на очередном шаге с помощью интерполяции.

Модель DC-GAN для генерации изображений разрешения 256×256 имеет 26.805.633 обучаемых параметров весом 102,26 MB в генераторе и 15.729.921 обучаемых параметров весом 60 MB в генераторе. Подсчёт градиентов для одного изображения требует 110,50 MB для дискриминатора и 148,25 MB для дискриминатора. Вычисления производились на видеокарте GPU Tesla P100 16GB.

На наборе изображений рентгеновских снимков модель обучалась на протяжении 70 эпох, на графике функции потерь (рисунок 6.3) видно, что обучение происходило нестабильно, видны частые и резкие скачки значений функции потерь как генератора, так и дискриминатора.

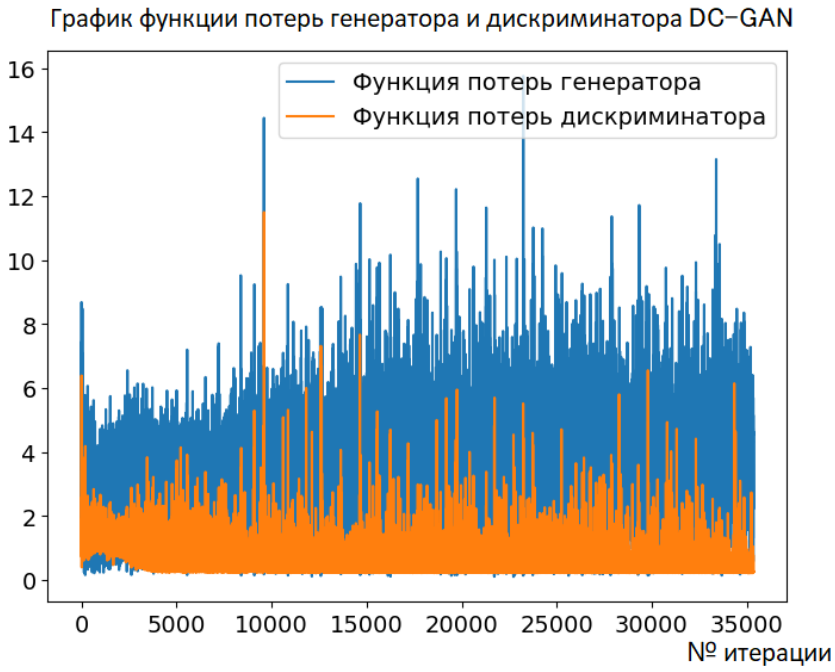


Рисунок 6.3 – График функции потерь DC-GAN

Лучший результат по метрике FID для набора изображений рентгеновских снимков модель показала после 50 эпох, FID составил 46,7. Полное время обучения составило 10 часов 40 минут. Примеры генерации изображены на рисунке 6.4.

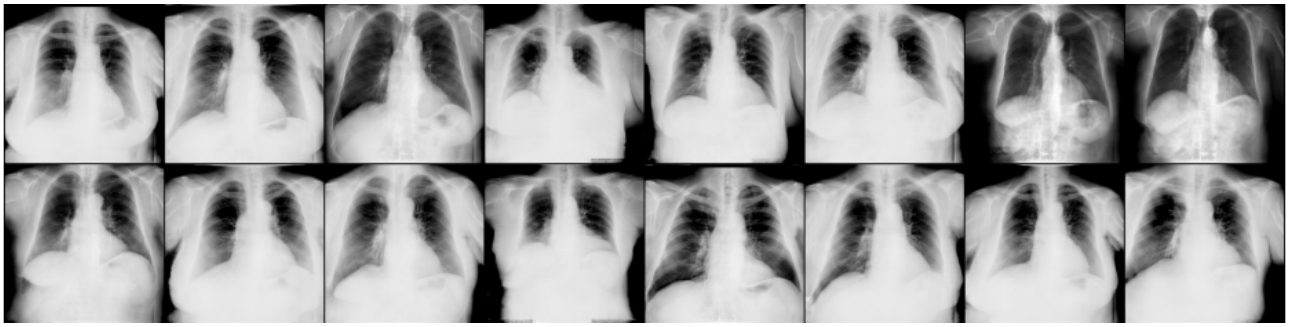


Рисунок 6.4 – Пример генерации с помощью модели DC-GAN

Стоит отметить, что из-за нестабильного обучения довольно трудно оценить, когда функция потерь попадает в локальный минимум, в силу чего трудно сказать, когда стоит остановить обучение.

На наборе с изображениями КТ-срезов модель была склонна к коллапсу, и удовлетворительных результатов не показала. В частности, значение метрики FID превышало 100. Связано это может быть с почти в два раза большим размером этого набора изображений (60.000) по сравнению с размером набора данных с изображениями рентгеновских снимков (32.000).

Дополнительные графики функций потерь и выходов дискриминатора находятся в Приложении Б. Дополнительные примеры генерации находятся в Приложении В.

6.2 Применение архитектуры SA-GAN

Модель SA-GAN для генерации изображений разрешения 256×256 имеет 38.452.561 обучаемых параметров весом 146,68 MB в дискриминаторе и 66.247.969 обучаемых параметров весом 252,72 MB в генераторе. При этом, из-за большого количества одномерных свёрток в блоках самовнимания, которых в модели несколько, подсчёт градиентов для одного изображения требует 180 MB для дискриминатора и 423,38 MB для генератора, что больше, чем занимают веса модели. Вычисления производились на видеокарте GPU Tesla P100 16GB.

На обоих наборах данных данная модель не смогла обучиться до приемлемого качества и была подвержена коллапсу мод.Варьирование гиперпара-

метров, имеющих отношение к регуляризации дискриминатора, не приводили к приемлемому результату. На наборе данных с КТ-срезами модель коллапсировала практически сразу, хоть при некоторых гиперпараметрах и удавалось ненадолго отсрочить этот момент. После коллапса, часто, генератор начинает производить шум (рисунок 6.5).

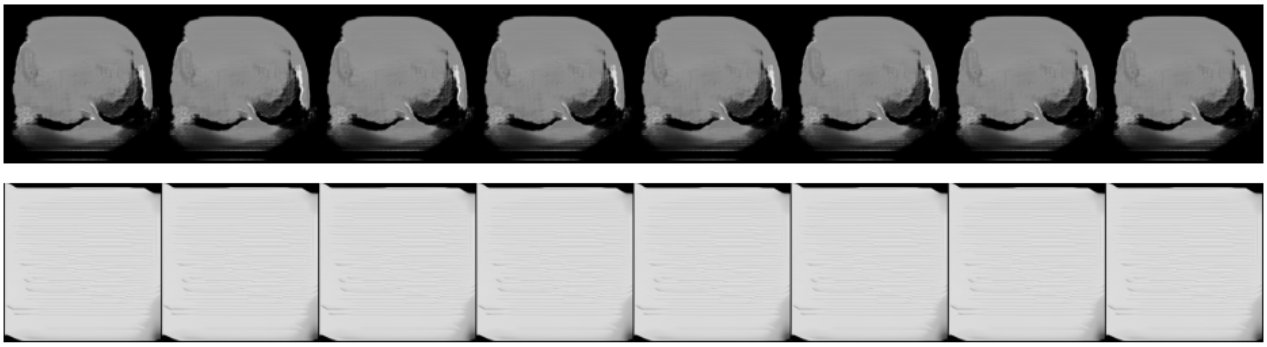


Рисунок 6.5 – Коллапс мод и следующий за ним шум

Коллапс мод, как правило, происходит из-за взрыва градиентов [9]. В нашем случае, связано это с тем, что дискриминатор непропорционально «силён» по отношению к генератору, то есть слишком хорошо классифицирует реальные и сгенерированные изображения. При этом, как только генератору случайно удаётся создать изображение, которое генератор классифицирует плохо, градиенты резко меняются и увеличиваются, то есть происходит их «взрыв», и генератор начинает создавать исключительно такие изображения, которые смогли «обмануть» дискриминатор. В таком случае генератор воспроизводит только одно какое-то изображение, как правило, слабо похожее на что-то осмысленное, часто это изображение представляет из себя шум.

На графиках значений функции потерь генератора и дискриминатора (рисунок 6.6), а также на графике значений выходов дискриминатора на реальных и сгенерированных изображениях отчётливо виден момент, когда генератор полностью «ломается» и начинает воспроизводить шум, при этом заметно, что коллапс происходит практически с самого начала обучения и на первой половине графика функции потерь видно, что значения резко изменяются. Резкий скачок на графике выходов дискриминатора и практически полная стабилизация на графике функции потерь соответствует моменту, когда генератор начинает воспроизводить шум и перестаёт учиться, при этом можно заметить, что значения функции потерь дискриминатора очень близки к нулю.

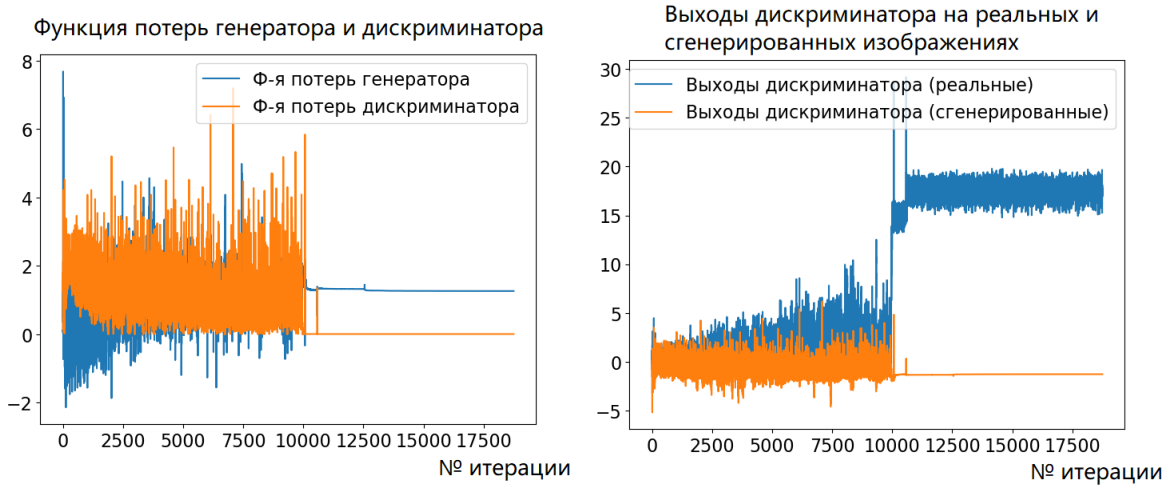


Рисунок 6.6 – Графики функции потерь и выходов дискриминатора

Похожие результаты были получены и на наборе данных с рентгеновскими снимками. При этом, поскольку рентгеновских снимков почти в два раза меньше, чем КТ-срезов, коллапс мод на этом наборе данных происходил ещё раньше.

Стоит отметить, что при понижении разрешения изображения до 128×128 и 64×64 модель не коллапсирует и способна генерировать правдопобные изображения. При этом процесс обучения даже на таком низком разрешении вычислительно затратный.

Дополнительные графики функций потерь и выходов дискриминатора представлены в Приложении Б. Дополнительные примеры генерации представлены в Приложении В.

6.3 Применение архитектуры BIG-GAN-DEEP

Модель BIG-GAN-DEEP для генерации изображений разрешения 256×256 имеет 2.268.001 обучаемых параметров весом 8,65 MB в дискриминаторе и 3.019.009 обучаемых параметров весом 11,52 MB в генераторе. При этом, из-за наличия блока самовнимания в генераторе и дискриминаторе, а также из-за использования Residual-блоков, подсчёт градиентов для одного изображения требует 387,48 MB для дискриминатора и 453,75 MB для генератора, что в десятки раз больше, чем занимают веса модели. Вычисления производились на двух видеокартах GPU Tesla V100 16GB параллельно, при этом подсчёт статистик при пакетной нормализации синхронизировался на обоих устройствах.

Как и любая генеративно-состязательная сеть, эта модель подвержена

коллапсу мод и обучение не всегда стабильно. Требуется подбирать гиперпараметры и проводить серию экспериментов, чтобы выяснить, какая их комбинация даст лучший результат.

Мы варьировали темпы обучения (lr) для генератора и дискриминатора с использованием TTUR. Каждая модель обучалась на протяжении пятидесяти эпох. Стоит отметить, что параметр γ изменялся экспоненциально от значения $\gamma = 0{,}1$ (на первой эпохе) до $\gamma = 0{,}001$ (на последней эпохе). Последний эксперимент был проведён с неизменяющимся параметром $\gamma = 0{,}1$. В таблице 6.1 приведены лучшие результаты по метрике FID.

Таблица 6.1 – Результаты экспериментов для архитектуры BIG-GAN-DEEP

Градиентная регуляризация	lr дискриминатора	lr генератора	Лучший FID
Нет	0,00005	0,000025	61,75 (15-я эпоха)
Нет	0,0001	0,000025	66,6 (45-я эпоха)
Да	0,000025	0,000025	49,5 (20-я эпоха)
Да	0,00005	0,000025	55,77 (45-я эпоха)
Да	0,0001	0,000025	82,23 (10-я эпоха)
Да	0,000025	0,00005	39,03 (43-я эпоха)
Да, $\gamma = 0{,}1 = \text{const}$	0,000025	0,000025	**28,61** (50-я эпоха)

Градиентная регуляризация в самом деле стабилизирует обучение, ускоряет сходимость и улучшает качество генерации. Лучший результат (FID = 28,61) был достигнут при использовании градиентной регуляризации с фиксированным параметром γ и одинаковом темпе обучения для генератора и дискриминатора, равному 0,000025 (пример генерации на рисунке 6.7). Полное время обучения одной модели при заданных параметрах составило 69 часов.

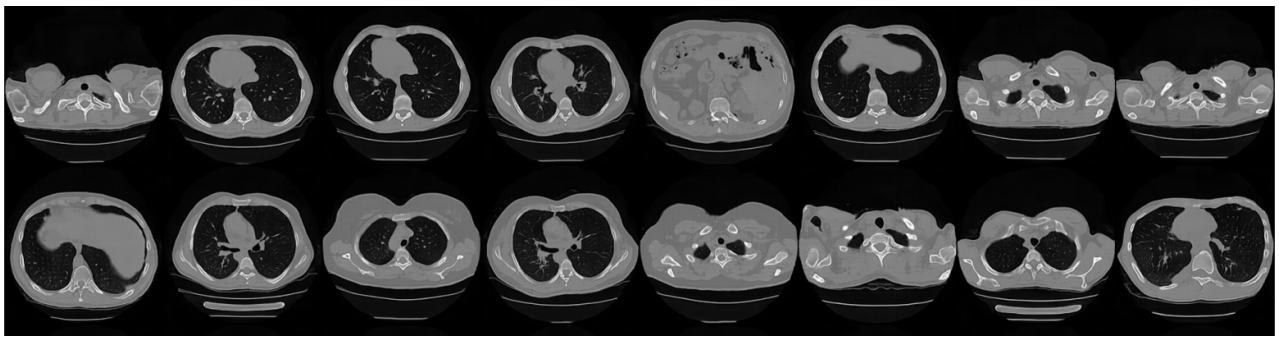


Рисунок 6.7 – Пример генерации (FID = 28,61)

Стоит отметить, что использование TTUR (разный темп обучения) в совокупности с градиентной регуляризацией не улучшает ни стабильность, ни

качество. Лучший результат был достигнут при одинаковом темпе обучения, но при фиксированном параметре γ, а значит градиентная регуляризация способна самостоятельно стабилизировать обучение, за счёт чего в том числе улучшается качество.

Без градиентной регуляризации модель обучается менее стабильно, достигает худшего качества и мы можем наблюдать коллапс мод. В частности, при отсутствии градиентной регуляризации, при темпе обучения 0,000025 и 0,0001 для генератора и дискриминатора соответственно, после 45-й эпохи качество изображений начинает резко падать и, на 50-й эпохе, происходит коллапс мод (рисунок 6.8).

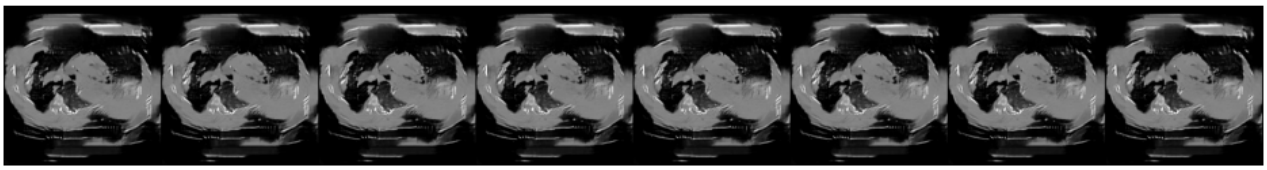


Рисунок 6.8 – Пример генерации при коллапсе мод

При этом, на графике функции потерь отчётливо виден момент, на котором произошёл коллапс мод. Связано это с резким скачком значений градиентов («взрыв градиентов»), который, в свою очередь, связан со скачком функции потерь, и, как следствие, с резким изменением значения функции потерь. На рисунке 6.9 (слева) представлен график значений функции потерь на каждом пакете данных, начиная с 45-й эпохи и до 55-й. На нём отчётливо виден резкий скачок, после которого значения функции потерь начинают сильно и хаотично изменяться.

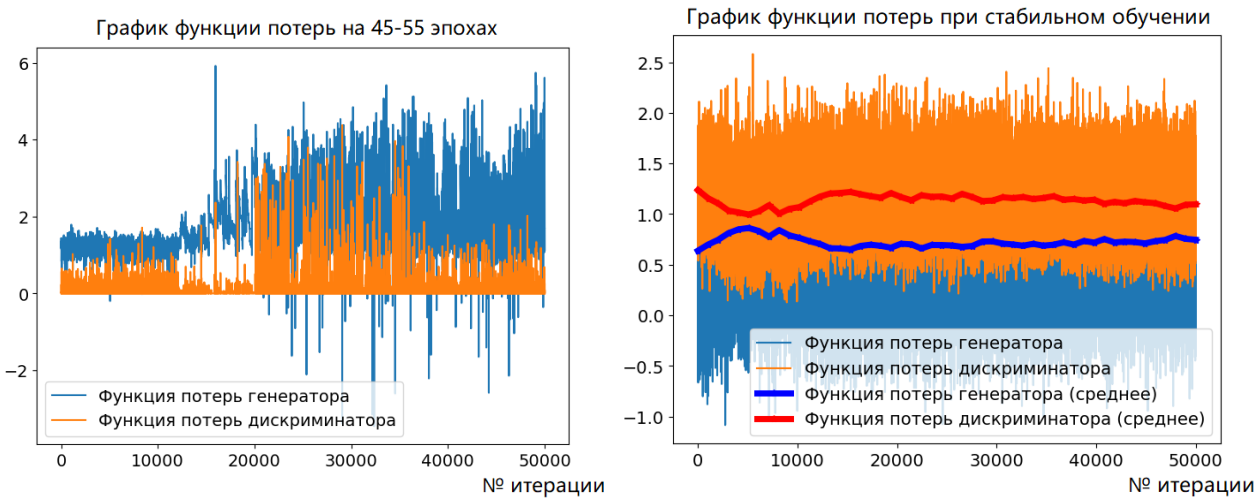


Рисунок 6.9 – График функции потерь в окрестности коллапса (слева) и график функции потерь при стабильном обучении (справа)

Для сравнения, на рисунке 6.9 (справа) изображён график функции по-

терь при стабильном обучении для модели с лучшей метрикой FID. Можно отметить, что, по сравнению с графиком около коллапса, график при стабильном обучении не имеет таких резких и больших скачков.

На наборе изображений рентгеновских снимков был обучен BIG-GAN-DEEP с использованием градиентной регуляризации с фиксированным параметром $\gamma = 0{,}1$ и одинаковыми темпами обучения для генератора и дискриминатора, равными 0,000025. Модель обучалась на протяжении 80 эпох, и лучший результат по метрике FID был достигнут после 65 эпох обучения, FID составил 40,19, при этом начиная с 65 эпохи метрика FID не улучшалась, в частности, после 80 эпохи FID составил 40,3, что сопоставимо с результатом после 65 эпох обучения. При изменении темпа обучения на 0,000025 для генератора и 0,00005 для дискриминатора, модель достигла лучших результатов. Так, после 50 эпох обучения, FID составил 29,12. Отметим, что для каждого набора данных, то есть для каждой задачи, требуется отдельная настройка гиперпараметров. Обучение происходило так же стабильно, как и на наборе данных с КТ-срезами. Графики функции потерь выглядят схожим образом.

Дополнительные графики функций потерь и выходов дискриминатора находятся в Приложении Б. Дополнительные примеры генерации находятся в Приложении В.

6.4 Применение ансамбля генеративных моделей CGAN и VAE

Обучение ансамбля генеративных моделей происходило последовательно. Сперва обучался вариационный автокодировщик на масках лёгких, и только после этого обучался CGAN. При этом, прежде чем подать маску на вход CGAN, она сначала пропускалась через VAE, и уже выход VAE подавался на вход CGAN. Связано это с тем, что генерируемые VAE маски, во-первых, не являются бинарными (выходы VAE — дробные числа, которые, в случае генерации бинарной маски, либо близки к нулю, либо к единице), а во-вторых, VAE не идеально восстанавливает маски. Таким образом, ансамбль получается согласованным.

Вариационный автокодировщик для генерации изображений разрешения 256×256 имеет 802.645 обучаемых параметров весом 3,06 MB. Подсчёт градиентов для одного изображения требует 20,80 MB. Вычисления производились на видеокарте GPU Tesla P100 16GB.

Модель обучалась на протяжении 600 эпох, при этом темп обучения на

первых 400 эпохах был выбран 0,0001 и каждое слагаемое в функции потерь имело коэффициент 1. На последних 200 эпохах темп обучения был снижен до 0,00005, часть функции потерь бинарная кросс-энтропия суммировалась с весом 0,1, KL-диверегенция с весом 1. Сделано это для того, чтобы ещё сильнее приблизить выходы кодировщика к параметрам стандартного нормального распределения, поскольку по прошествии 400 эпох было обнаружено, что модель весьма неплохо восстанавливает изображения, но среди сгенерированных изображений встречаются аномальные (рисунок 6.10, слева), что свидетельствовало о несоответствии статистик, воспроизводимых кодировщиком, параметрам нужного распределения. После обучения качество генерируемых масок удовлетворительное, примеры на рисунке 6.10, справа. Полное время обучения вариационного автокодировщика составило 6 часов.

Рисунок 6.10 – Результаты генерации VAE после 400 (слева) и 600 эпох (справа)

CGAN для восстановления изображений по некоторой маске разрешения 256×256 имеет 54.407.809 обучаемых параметров весом 207,55 MB в генераторе и 2.764.609 обучаемых параметров весом 10,55 MB в дискриминаторе. Подсчёт градиентов для одного изображения требует 45,28 MB для дискриминатора и 131,46 MB для генератора. Вычисления производились на видеокарте GPU Tesla P100 16GB.

Модель обучалась на протяжении 100 эпох, при этом обучение происходило весьма стабильно, что можно наблюдать на графике функции потерь на рисунке 6.11.

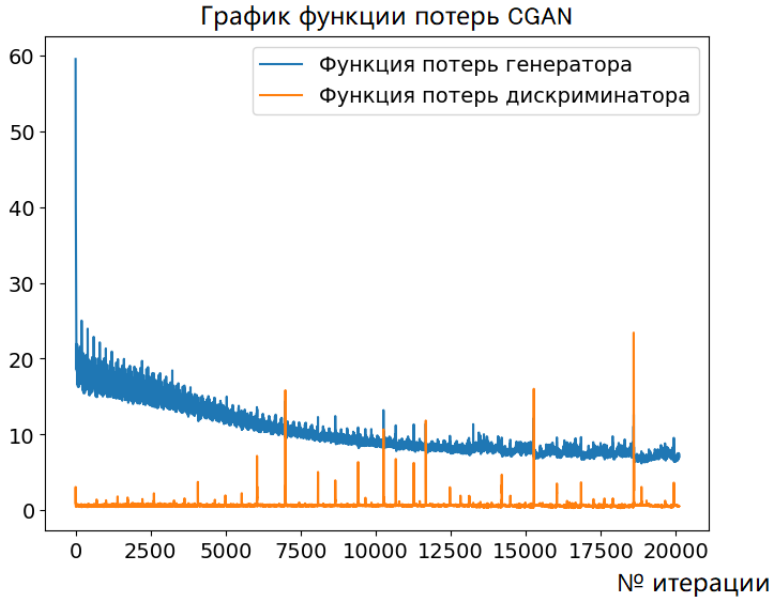


Рисунок 6.11 – График функции потерь CGAN

В результате был обучен ансамбль генеративных моделей. Метрика FID после полного обучения моделей составила 47,37 и измерялась по случайно сгенерированным изображениям, то есть вариационному автокодировщику на вход подавался случайный вектор. Примеры сгенерированных изображений приведены на рисисунке 6.12. Полное время обучения модели CGAN составило 18 часов.

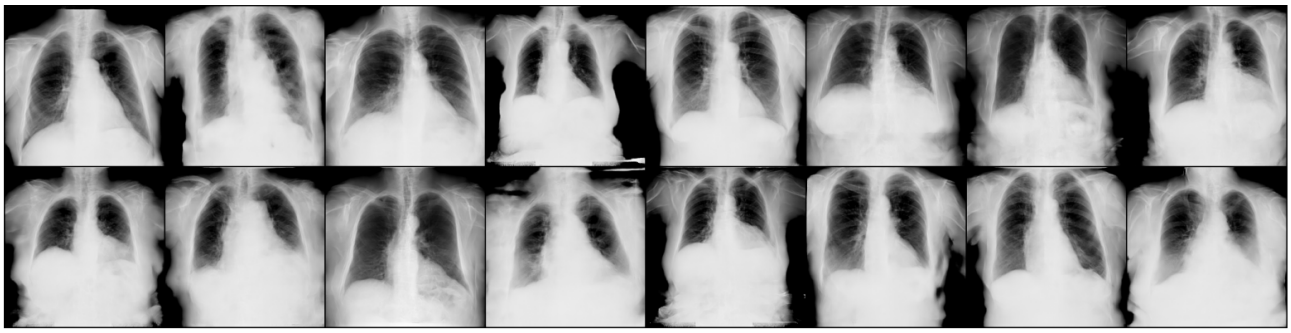


Рисунок 6.12 – Примеры сгенерированных ансамблем изображений

Дополнительные графики функций потерь и выходов дискриминатора представлены в Приложении Б. Дополнительные примеры генерации представлены в Приложении В.

6.5 Применение архитектуры SPADE-GAN

Модель SPADE-GAN для генерации изображений разрешения 256 × 256 имеет 96.198.977 обучаемых параметров весом 366,97 MB в генерато-

ре, 8.291.139 обучаемых параметров весом 31,63 MB в дискриминаторе и 10.464.256 обучаемых параметров весом 39,92 MB в энкодере. При этом, из-за архитектурных особенностей генератора, подсчёт градиентов для одного изображения требует 2321,06 MB для генератора, 59,11 MB для дискриминатора и 45,94 MB для энкодера, что суммарно больше примерно в 5,5 раз, чем занимают веса модели. Вычисления производились на видеокарте GPU Tesla V100 16GB.

На наборе изображений рентгеновских снимков модель обучалась на протяжении 80 эпох. Общее время обучения составило 69 часов. На графике значений функции потерь генератора и дискриминатора (рисунок 6.13) видно, что обучение происходило не стабильно, видны частые скачки значений функций потерь.

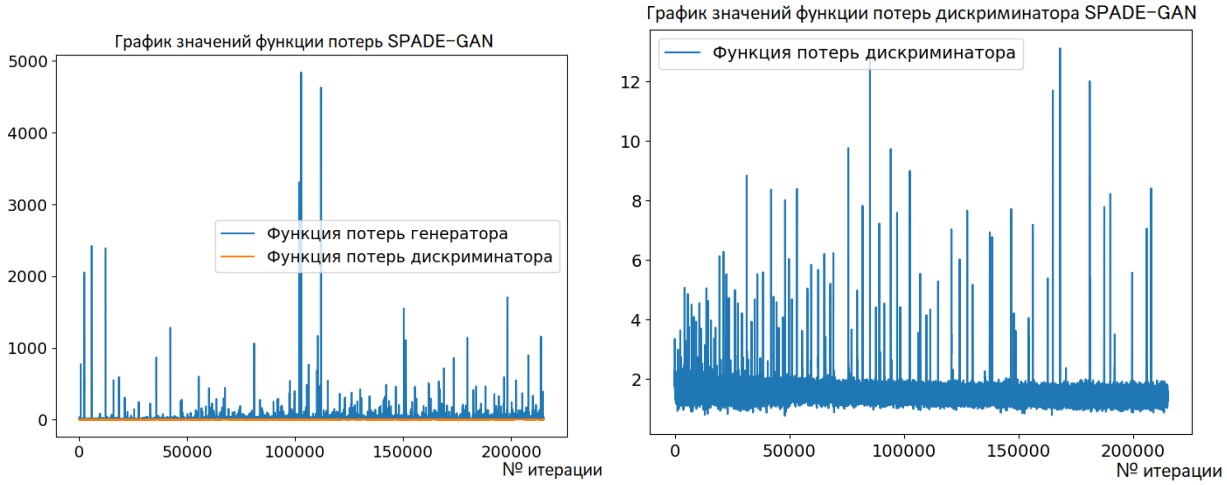


Рисунок 6.13 – График значений фукнции потерь генератора и дискриминатора (слева) и график значений функции потерь дискриминатора (справа)

Несмотря на нестабильное поведение значений функции потерь, до 50 эпохи значение метрики FID улучшалось, и по прошествии 80 эпох достигла своего минимума. Поскольку модель способна генерировать изображения двумя способами, а именно только по некоторой маске и по маске с некоторым исходным изображением (на котороже должно быть в некотором смысле похоже генерируемое изобраежние), метрика FID измерялась по обоим способам генерации. Для первого способа, генерация происходила для каждой маски из исходного набора изображений. Для второго способа все маски и исходные изображения случайно перемешивались (чтобы маска не соответствовала исходному изображению) и генерация осуществлялась по таким парам. Результаты по метрике FID для каждого из способов генерации по всем эпохам представлены в таблице 6.2.

Таблица 6.2 – Значения метрики FID при обучении SPADE-GAN

Прошло эпох	FID для генерации только по маске	FID для генерации по маске и исходному изображению
10	33,71	33.14
20	16,42	15,27
30	12,41	12,75
40	10,14	9,69
50	9,01	8,48
60	9,3	9,2
70	9,28	9,19
80	**7,27**	**7,4**

В отличие от ансамбля моделей CGAN и VAE, модель SPADE-GAN для одной и той же маски генерирует не идентичные изображения. Вариативность сгенерированных изображений для одной и той же маски представлена на рисунке 6.14. Стоит отметить, что изображения кажутся практически одинаковыми на первый взгляд, однако при детальном рассмотрении заметны отличия в расположении некоторых костей, общей формы тела.

Рисунок 6.14 – Пример сгенерированных изображений для фиксированной маски лёгких

На рисунке 6.15 представлены изображения, сгенерированные соответственно с помощью маски из первой строки и исходного изображения из первого столбца. Здесь (при подсчёте слева направо для масок и сверху вниз для исходных изображений) i-му исходному изображению соответствует i-я маска лёгких. В строках расположены результаты генерации для соответству-

ющих масок и фиксированного исходного изображения (первого в строке), в столбцах расположены результаты генерации для фиксированной маски и соответствующих исходных изображений.

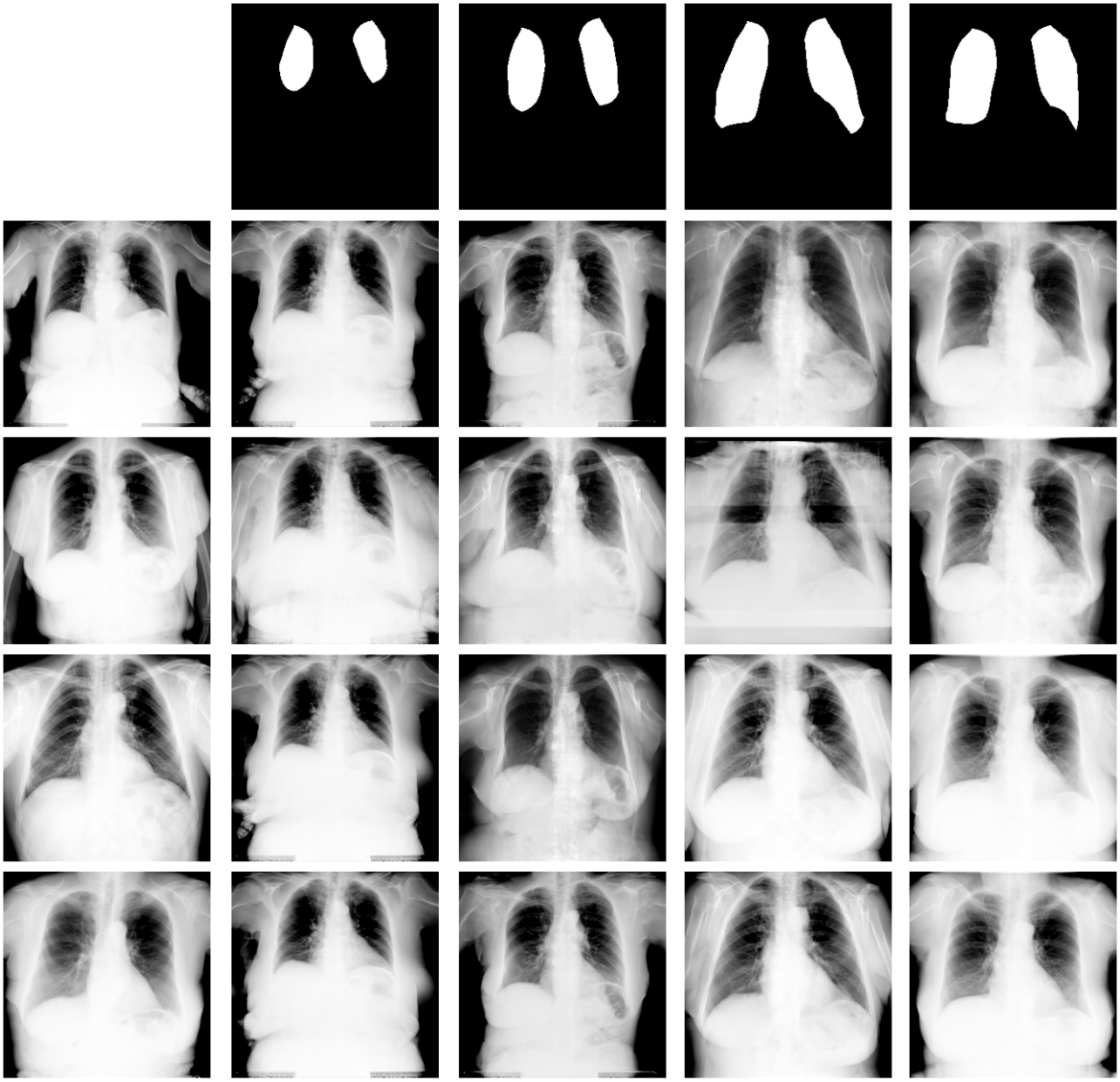


Рисунок 6.15 – Пример сгенерированных изображений с помощью масок (первая строка) и исходных изображений (первый столбец)

Можно заметить, что маска всё же доминирует при генерации и снижает разнообразие получаемых изображений, однако видно, что дополнительное условие в виде некоторого исходного изображения всё же меняет итог генерации, иногда существенно. Важно, что сгенерированные изображения соответствуют исходным анатомическим ограничениям в виде заданных масок лёгких.

Дополнительные графики функций потерь и выходов дискриминатора

находятся в Приложении Б. Дополнительные примеры генерации находятся в Приложении В.

6.6 Выводы

В таблице 6.3 для сравнения представлены лучшие результаты по метрике FID на разных наборах данных для каждой модели, а также число параметров модели и время обучения.

Таблица 6.3 – Значения метрики FID при обучении SPADE-GAN

Модель	Общее число параметров	Общее время обучения	Лучший FID (рентгеновские снимки)	Лучший FID (КТ-срезы)
DCGAN	42.535.554	10,67 ч	46,7	>100
SAGAN	104.700.530	—	Коллапс	Коллапс
BIG-GAN-DEEP	5.287.010	69 ч	29,12	**28,61**
Ансамбль (CGAN+VAE)	57.975.063	24 ч	47,37	—
SPADE-GAN	114.954.372	55 ч	**7,27** (**7,4**)	—

Лучший результат на наборе изображений КТ-срезов показала модель BIG-GAN-DEEP, которая имеет наименьшее число параметров среди всех моделей, но обучалась дольше всех из-за архитектурных особенностей и применённой при обучении градиентной регуляризации. Такое сравнительно небольшое число параметров является большим плюсом относительно других моделей, ведь чем меньше число параметров, тем меньше модель занимает памяти и быстрее генерирует изображения.

Стоит отметить, что благодаря градиентной регуляризации модель BIG-GAN-DEEP обучалась стабильно. При этом, несмотря на некоторые скачки в графике функции потерь дискриминатора CGAN (рисунок 6.11), ансамбль генеративных моделей также обучался стабильно, без признаков коллапса.

SPADE-GAN показал наилучший результат по метрике FID на наборе изображений рентгеновских снимков. Дополнительное преимущество относительно других моделей — контролируемая генерация, возможность задать анатомическое ограничение для генерируемого изображения, а также возможность получить разнообразные изображения для одного и того же огра-

ничения (маски лёгких). Вместе с этим данная модель имеет наибольшее число параметров и является одной из самых долгообучаемых. При этом стоит отметить, что такое высокое качество может быть связно, в том числе с тем, что модель, в силу большого числа параметров, хорошо «запомнила» тренировочный набор данных.

Таким образом, можно сделать следующие выводы:

— Исходя из проведённых экспериментов видно, что обучение генеративно-состязательных моделей является весьма трудной задачей и требует тщательного подбора гиперпараметров, чтобы модель показала свой лучший результат. Те гиперпараметры для одной и той же модели, которые давали наилучший результат на одном наборе данных, не обязательно дадут наилучший результат на другом.

— В результате применения градиентной регуляризации дискриминатора для модели BIG-GAN-DEEP обучение стабилизировалось, коллапс мод был предотвращён и был достигнут лучший результат генерации.

— Использование архитектурных нововведений улучшает качество модели, при этом не всегда большее число параметров значит лучшее качество. В частности, архитектура DCGAN имеет в 8 раз большее число параметров, чем архитектура BIG-GAN-DEEP, однако уступает в качестве генерируемых изображений на обоих наборах данных.

— В результате применения архитектуры SPADE-GAN для условной генерации изображений видно, что использование дополнительной пространственно-геометрической информации положительно влияет на качество генерируемых изображений. Модель SPADE-GAN демонстрирует большой прирост качества не только из-за большого числа параметров, но и благодаря архитектурным решениям, с помощью которых модель в полной мере использует анатомические условия генерируемого изображения в виде маски лёгких.

ЗАКЛЮЧЕНИЕ

В результате проведенного исследования цель дипломной работы — теоретически изучить и опытно-экспериментально провести построение, обучение и оценку генеративных моделей достигнута. Для реализации данной цели нами была проведена следующая работа и решены такие **задачи**:

— Изучены теоретические модели и архитектуры генеративно-состязательных моделей DCGAN, SAGAN, ансамбля генеративных моделей, состоящего из вариационного автокодировщика и CGAN, BIG-GAN-DEEP, SPADE-GAN.

— Расмотрены специфика и структура работы свёрточных генеративно-состя- зательных нейронных сетей с блоками самовнимания.

— Раскрыты особенности ансамблирования генеративных моделей.

— Изучены способы генерации искусственных медицинских изображений с анатомическими ограничениями.

— Реализованы вышеперечисленные архитектуры, цикл обучения.

— Исследовано явления коллапса мод у генеративно-состязательных моделей.

— Проведен сравнительный анализ генеративных моделей и сделаны выводы:

  – Используя метрику FID на разных наборах данных для каждой модели, а также учитывая число параметров модели и время обучения было определено, что лучший результат на наборе изображений КТ-срезов показала модель BIG-GAN-DEEP, которая имеет наименьшее число параметров среди всех моделей. Однако обучалась данная модель дольше всех из-за архитектурных особенностей и использованной при обучении градиентной регуляризации.

  – Модель BIG-GAN-DEEP обучалась стабильно без признаков коллапса благодаря градиентной регуляризации дискриминатора.

  – Использование архитектурных нововведений улучшает качество модели, при этом не всегда большее число параметров значит лучшее качество. В частности, архитектура DCGAN имеет в 8 раз

большее число параметров, чем архитектура BIG-GAN-DEEP, однако уступает в качестве генерируемых изображений на обоих наборах данных.

– Наилучший результат на наборе изображений рентгеновских снимков по метрике FID показала модель SPADE-GAN. Такое преимущество данная модель имеет за счет контролируемой генерации, возможности задать анатомическое ограничение для генерируемого изображения, а также возможности получить разнообразные изображения для одного и того же ограничения (маски лёгких). Вместе с этим данная модель имеет наибольшее число параметров и является одной из самых долгообучаемых.

– Проведённые эксперименты подтверждают, что обучение генеративно-состязательных моделей требует тщательного подбора гиперпараметров. Применение различных архитектур на разных наборах данных требует отдельного подбора гиперпараметров для каждого из них.

Отметим также возможные пути дальнейшего исследования:

— Реализация новых методов и архитектур для генерации медицинских изображений, в том числе изображений более высокого разрешения и 3D изображений.

— Усовершенствование рассмотренных ранее подходов.

— Продолжение изучения метода ансамблирования генеративных моделей.

— Изучение проблемы точного воспроизведения исходных данных нейронными сетями в контексте проблемы приватности данных.

Такое многовекторное исследование позволит повысить качество генерируемых медицинских изображений и разработать наиболее подходящие архитектуры генеративных моделей для поставленной задачи.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1 Goodfellow I. et al. Generative adversarial nets //Advances in neural information processing systems. — 2014. — Т. 27.

2 Arjovsky M., Chintala S., Bottou L. Wasserstein generative adversarial networks // International conference on machine learning. — PMLR, 2017. — С. 214-223.

3 Geometric GAN / Jae Hyun Lim, Jong Chul Ye — Режим доступа: https://arxiv.org/pdf/1705.02894v2. — Дата доступа: 15.01.2024.

4 Unsupervised Representation Learning With Deep Convolutional Generative Adversarial Networks / Alec Radford, Luke Metz — Режим доступа: https://arxiv.org/pdf/1511.06434.pdf. — Дата доступа: 13.01.2024.

5 Zhang H. et al. Self-attention generative adversarial networks //International conference on machine learning. – PMLR, 2019. – С. 7354-7363.

6 Spectral Normalization For Generative Adversarial Networks / Takeru Miyato, Toshiki Kataoka, Masanori Koyama, Yuichi Yoshida — Режим доступа: https://arxiv.org/pdf/1802.05957.pdf. — Дата доступа: 15.01.2024.

7 Auto-Encoding Variational Bayes / Diederik P. Kingma, Max Welling — Режим доступа: https://arxiv.org/pdf/1312.6114.pdf. — Дата доступа: 27.10.2023.

8 Image-to-image translation with conditional adversarial networks / Isola P. et al. //Proceedings of the IEEE conference on computer vision and pattern recognition. – 2017. – С. 1125-1134.

9 Large Scale GAN Training for High Fidelity Natural Image Synthesis / Andrew Brock, Jeff Donahue, Karen Simonyan — Режим доступа: https://arxiv.org/pdf/1809.11096.pdf. — Дата доступа: 10.02.2024

10 Which training methods for gans do actually converge? / Mescheder L., Geiger A., Nowozin S. //International conference on machine learning. – PMLR, 2018. – С. 3481-3490.

11 Stabilizing Training of Generative Adversarial Networks through Regularization / Kevin Roth, Aurelien Lucchi, Sebastian Nowozin, Thomas

Hofmann — Режим доступа: https://arxiv.org/pdf/1705.09367.pdf. — Дата доступа 24.02.2024

12 Semantic image synthesis with spatially-adaptive normalization / Park T. et al. //Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. – 2019. – С. 2337-2346.

13 High-resolution image synthesis and semantic manipulation with conditional gans / Wang T. C. et al. //Proceedings of the IEEE conference on computer vision and pattern recognition. – 2018. – С. 8798-8807.

14 Gans trained by a two time-scale update rule converge to a local nash equilibrium / Heusel M. et al. //Advances in neural information processing systems. – 2017. – Т. 30.

15 Adam: A Method For Stochastic Optimization / Diederik P. Kingma, Jimmy Lei Ba — Режим доступа: https://arxiv.org/pdf/1412.6980.pdf. — Дата доступа: 07.01.2024.

ПРИЛОЖЕНИЯ

ПРИЛОЖЕНИЕ А

Ссылка и QR-код на репозиторий github с исходным кодом

Исходный код реализации архитектур генеративно-состязательных нейронных сетей, а также циклы обучения находятся в репозитории github по ссылке:

https://github.com/DmitryKarpenk0/GANs_architecture_and_training

ПРИЛОЖЕНИЕ Б

Дополнительные графики функций потерь и выходов дискриминатора в процессе обучения генеративно-состязательных моделей

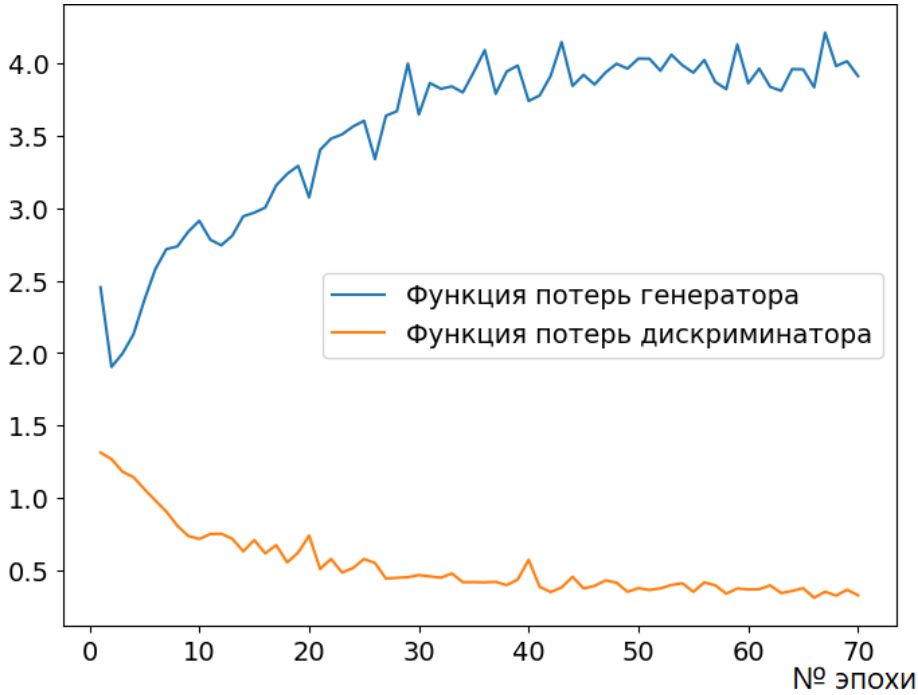


График функции потерь DC-GAN на наборе изображений рентгеновских снимков по эпохам

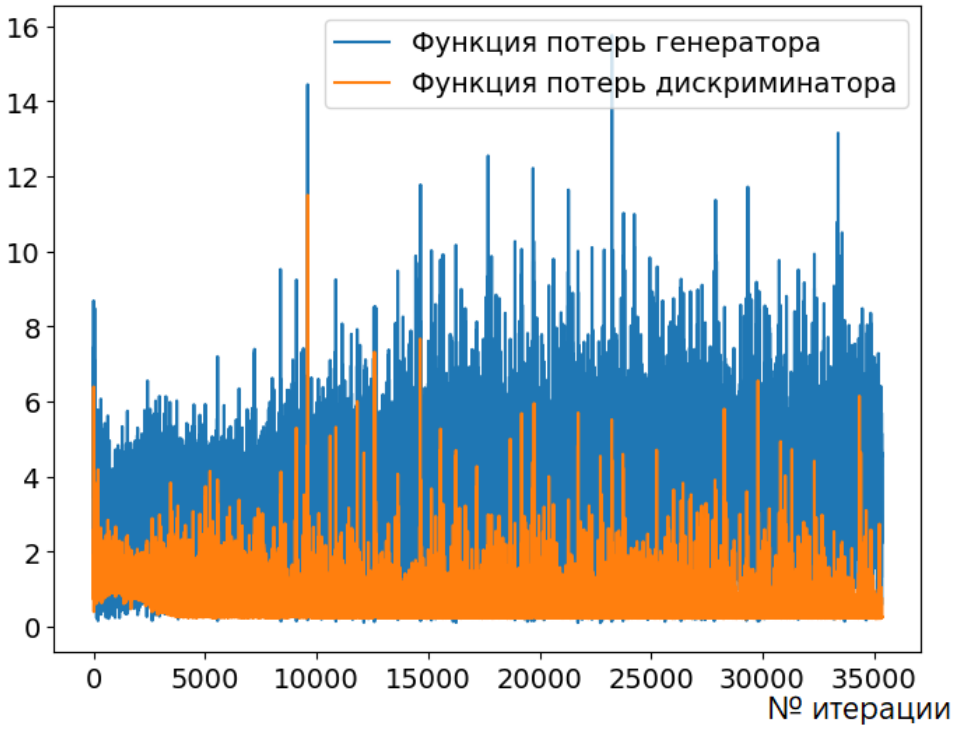


График функции потерь DC-GAN на наборе изображений рентгеновских снимков на каждой итерации обучения

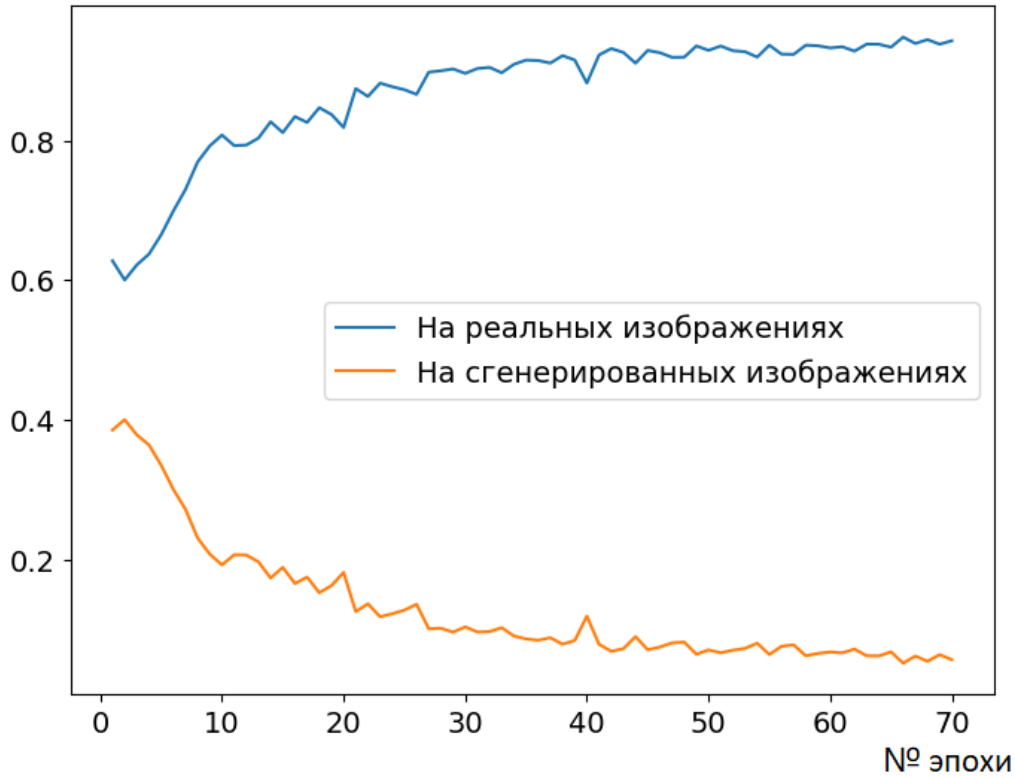


График усреднённых значений выходов дискриминатора DC-GAN на реальных и сгенерированных изображениях на наборе изображений рентгеновских снимков по эпохам

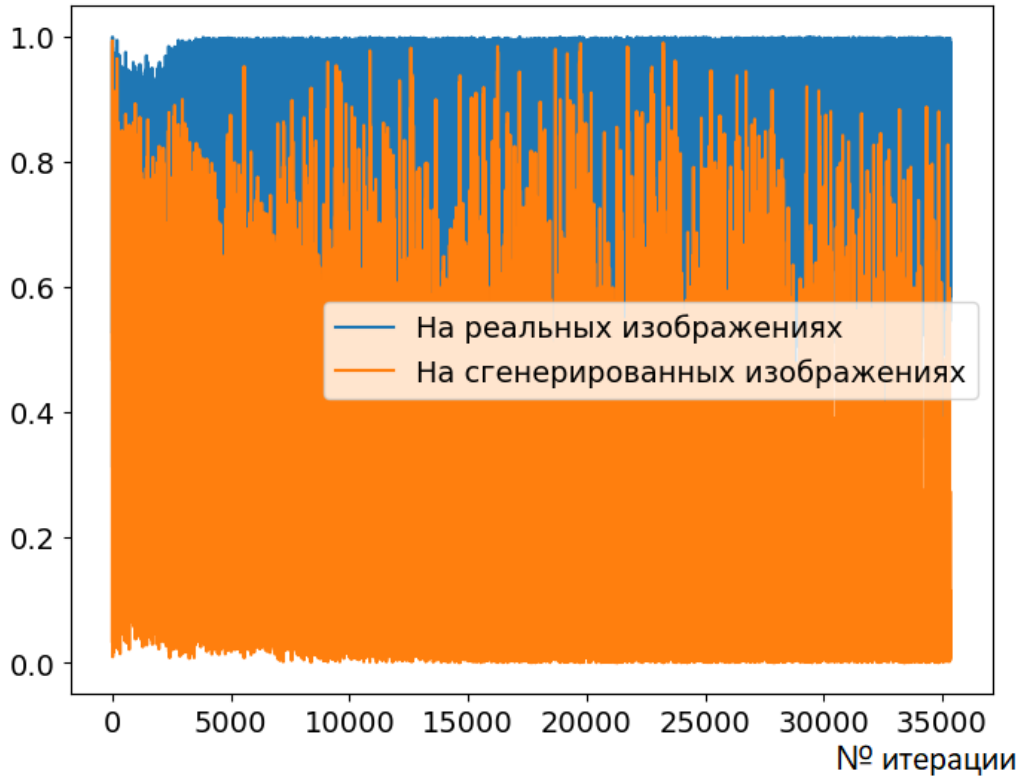


График усреднённых значений выходов дискриминатора DC-GAN на реальных и сгенерированных изображениях на наборе изображений рентгеновских снимков на каждой итерации обучения

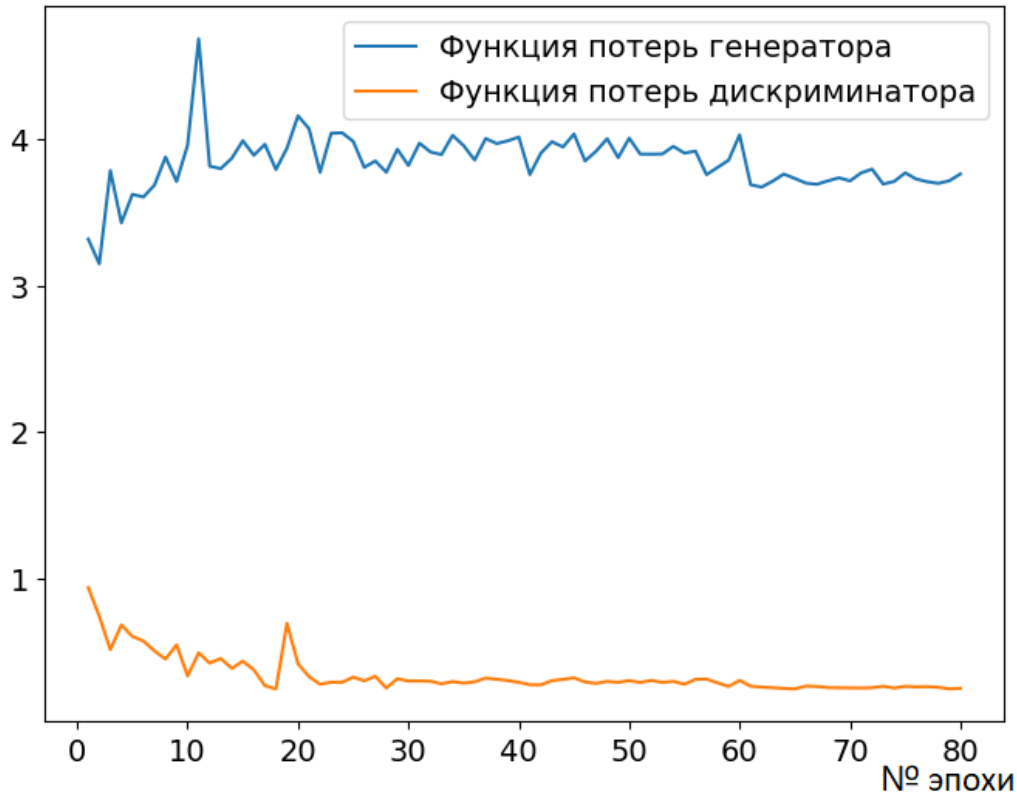


График функции потерь DC-GAN на наборе изображений КТ-срезов по эпохам

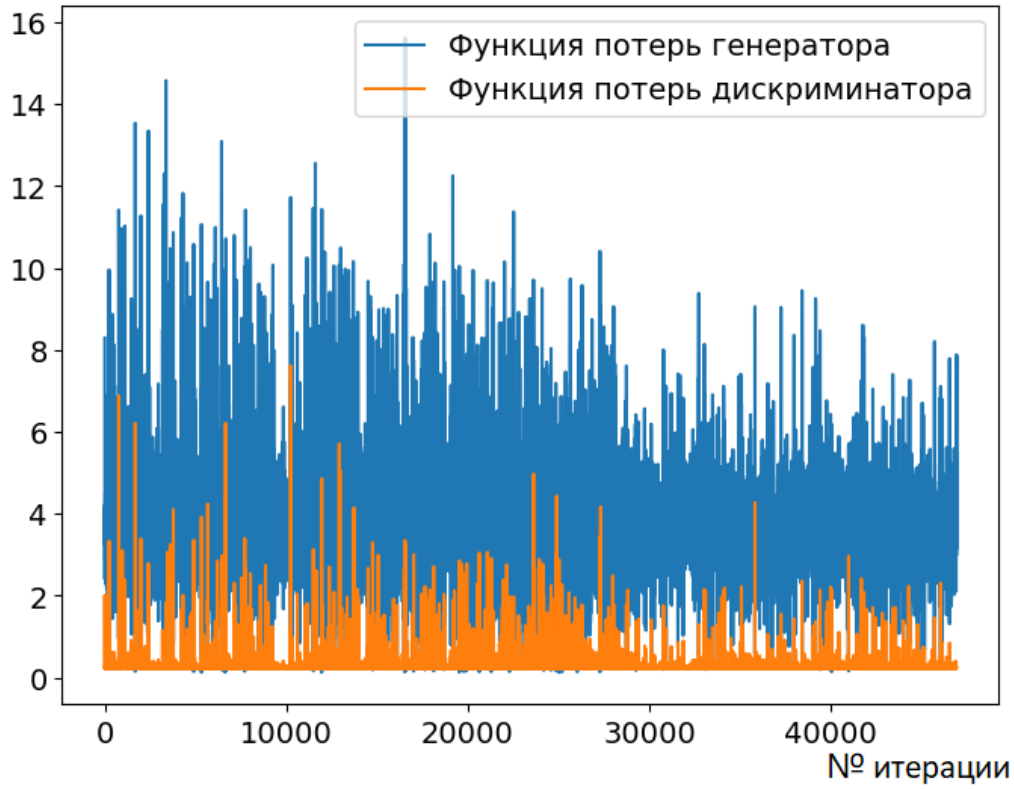


График функции потерь DC-GAN на наборе изображений КТ-срезов на каждой итерации обучения

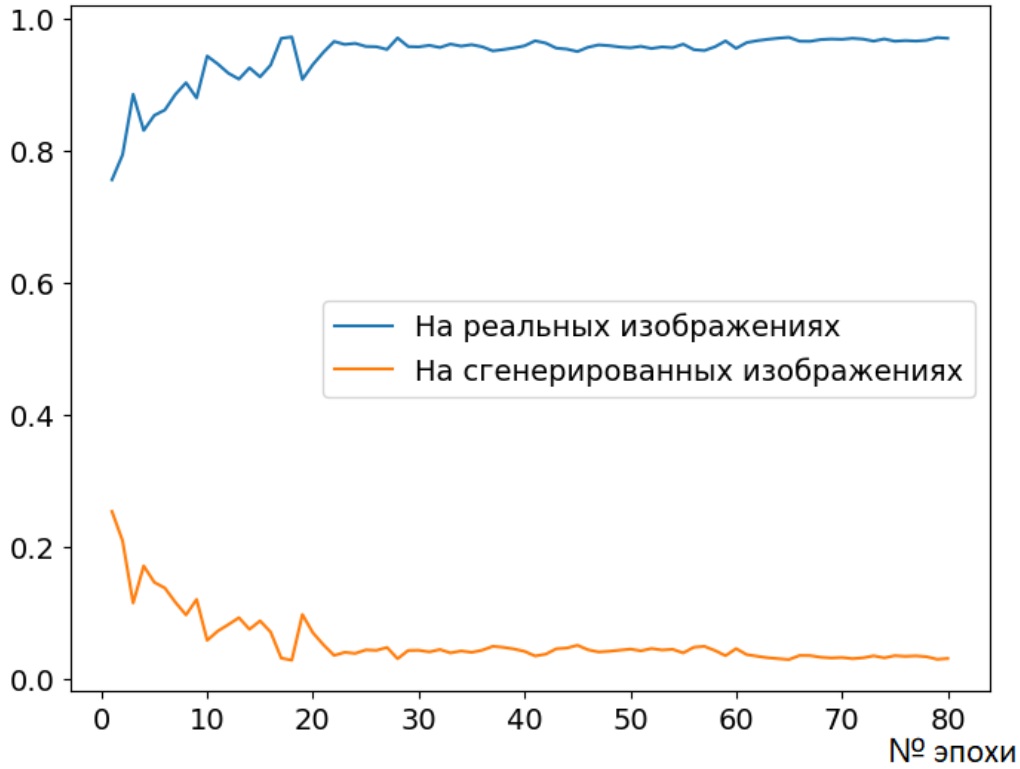


График усреднённых значений выходов дискриминатора DC-GAN на реальных и сгенерированных изображениях на наборе изображений КТ-срезов по эпохам

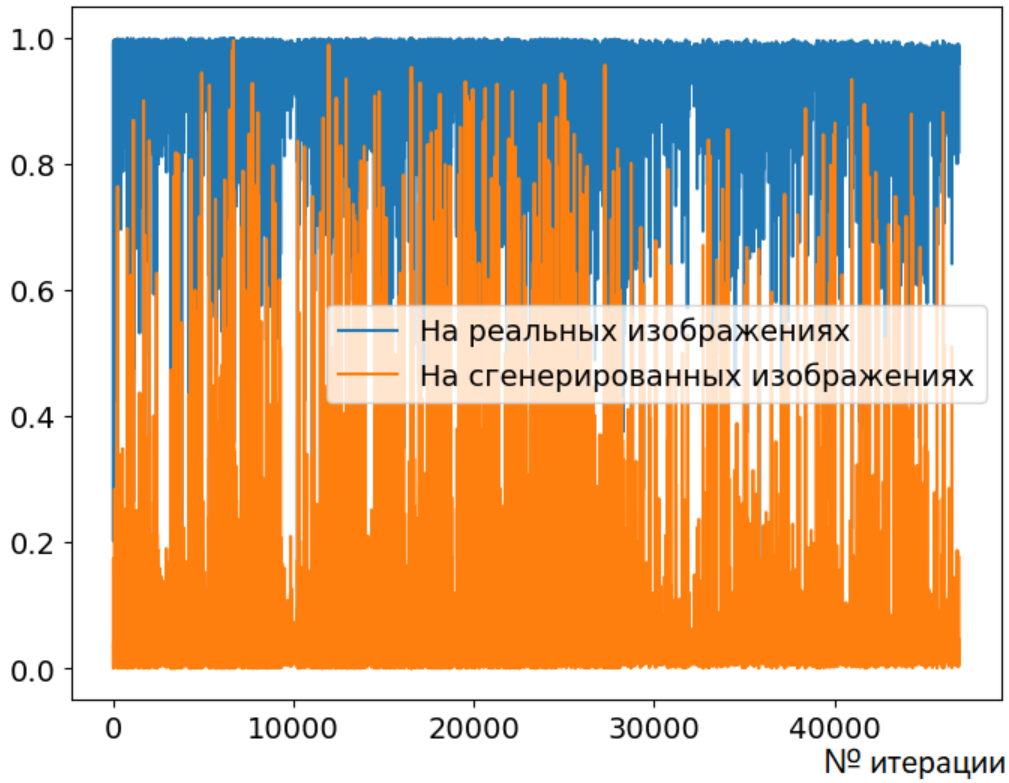


График усреднённых значений выходов дискриминатора DC-GAN на реальных и сгенерированных изображениях на наборе изображений КТ-срезов на каждой итерации обучения

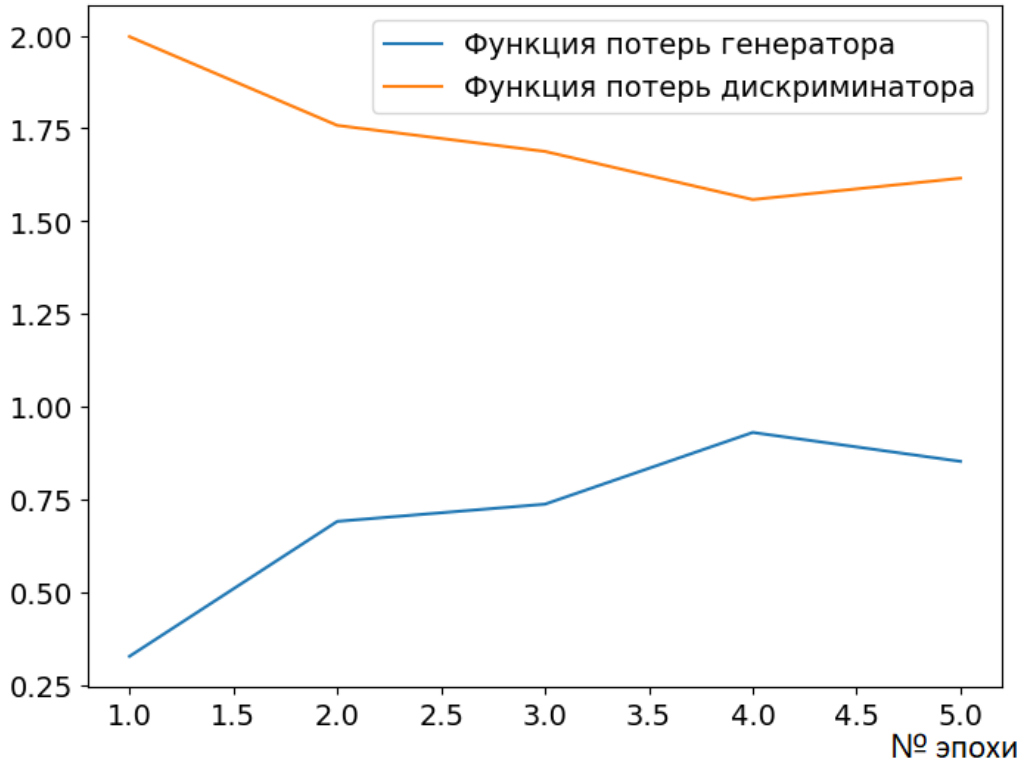


График функции потерь SA-GAN на наборе изображений рентгеновских снимков по эпохам

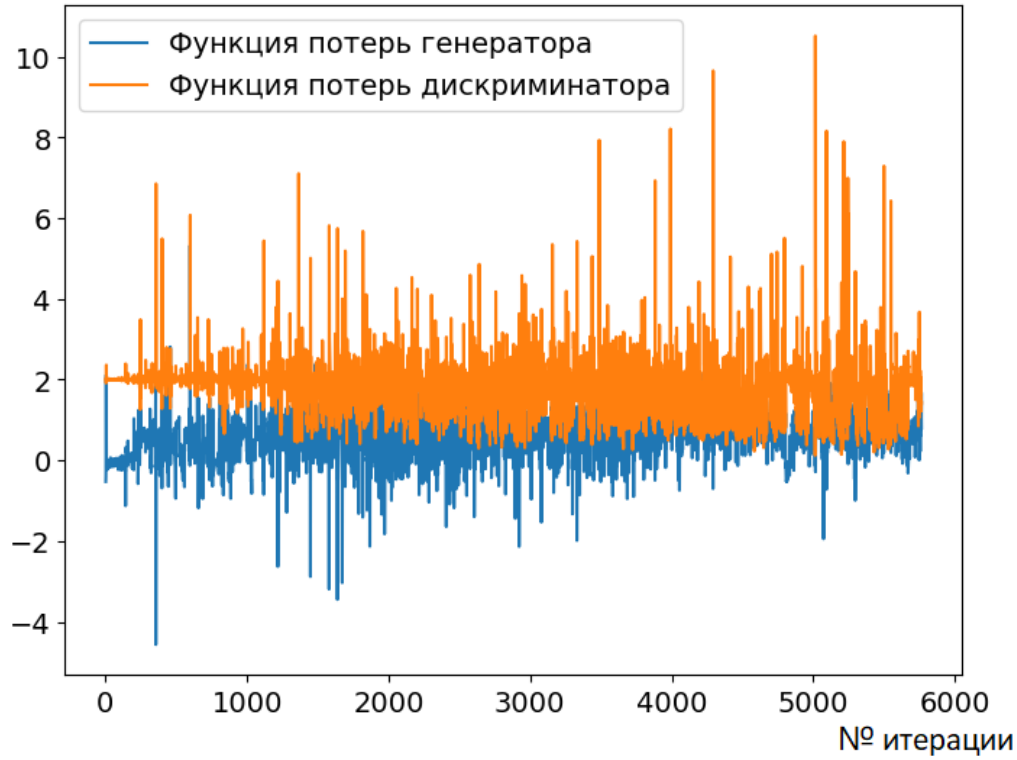


График функции потерь SA-GAN на наборе изображений рентгеновских снимков на каждой итерации обучения

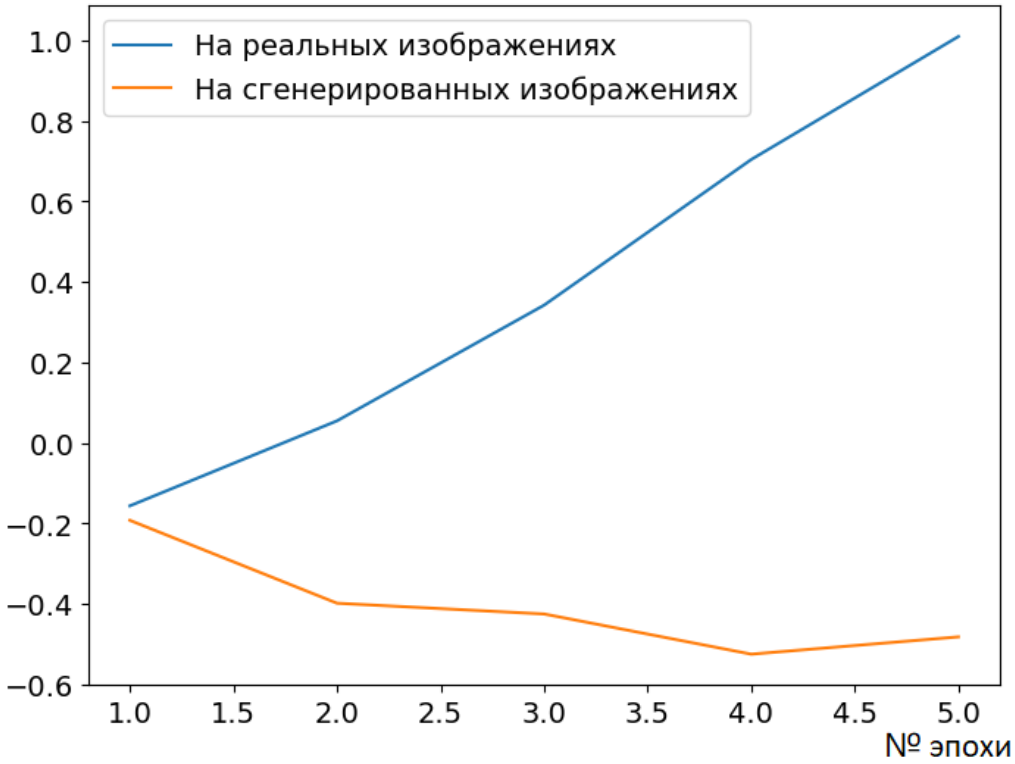


График усреднённых значений выходов дискриминатора SA-GAN на реальных и сгенерированных изображениях на наборе изображений рентгеновских снимков по эпохам

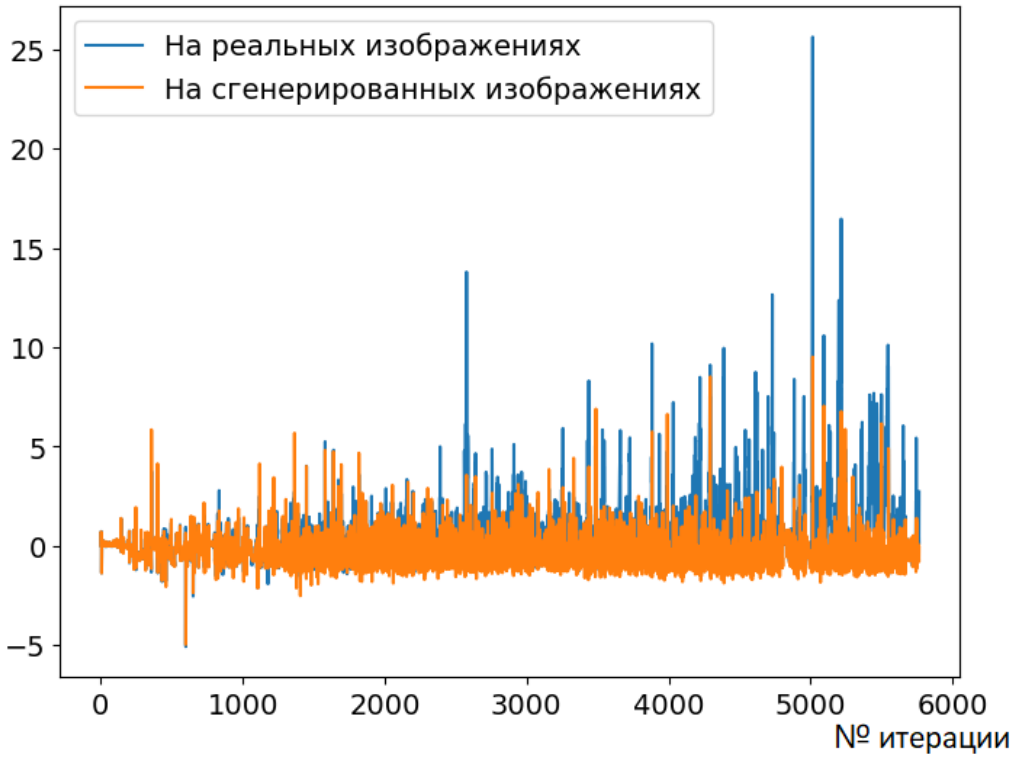


График усреднённых значений выходов дискриминатора SA-GAN на реальных и сгенерированных изображениях на наборе изображений рентгеновских снимков на каждой итерации обучения

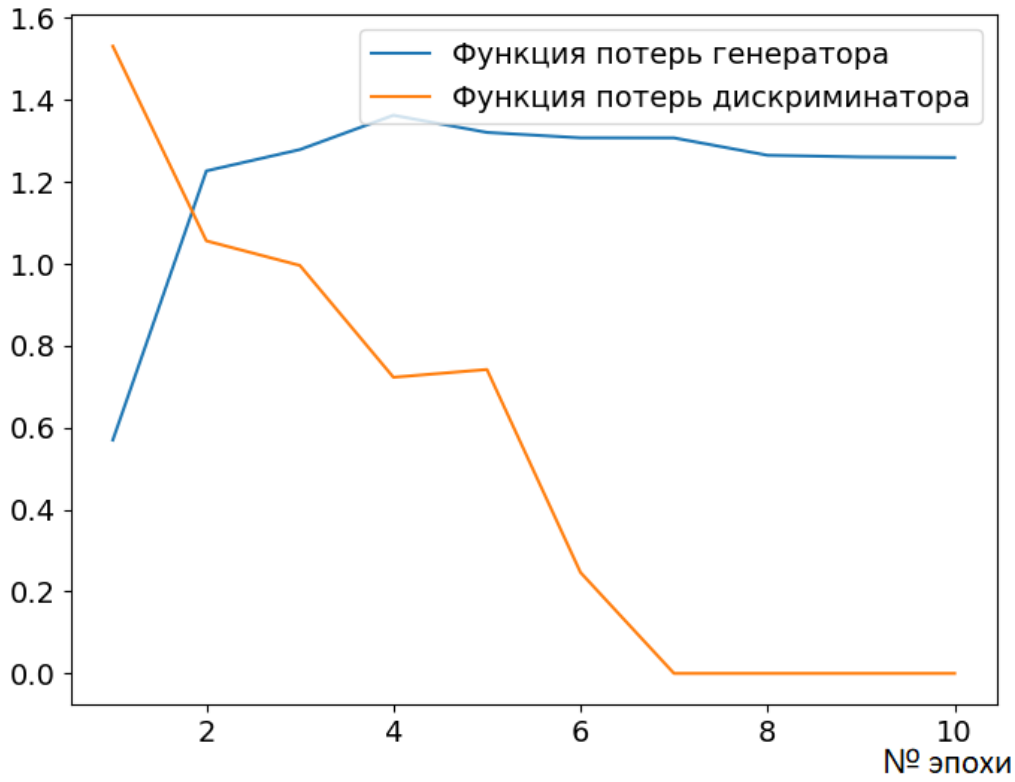


График функции потерь SA-GAN на наборе изображений КТ-срезов по эпохам

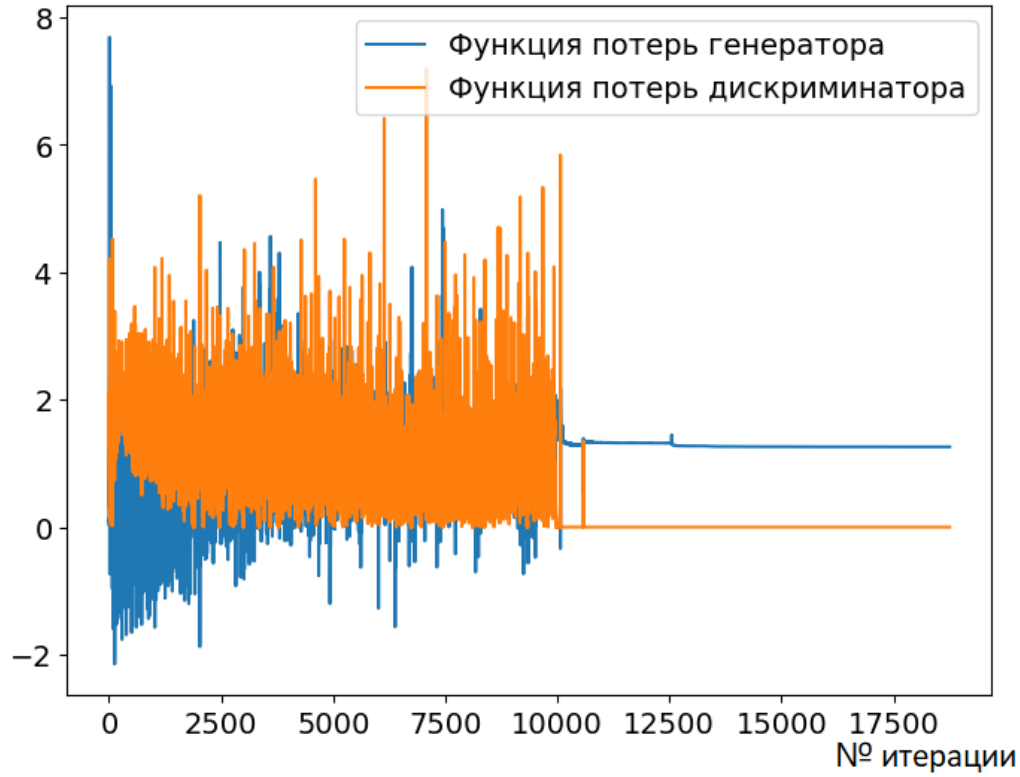


График функции потерь SA-GAN на наборе изображений КТ-срезов на каждой итерации обучения

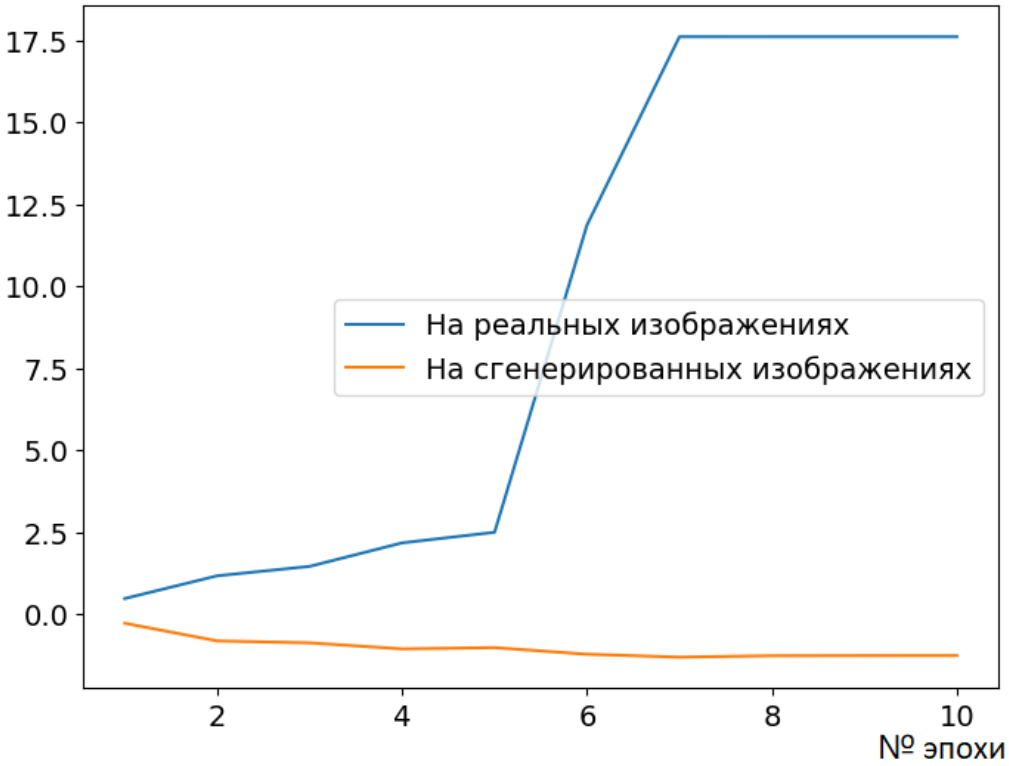


График усреднённых значений выходов дискриминатора SA-GAN на реальных и сгенерированных изображениях на наборе изображений КТ-срезов по эпохам

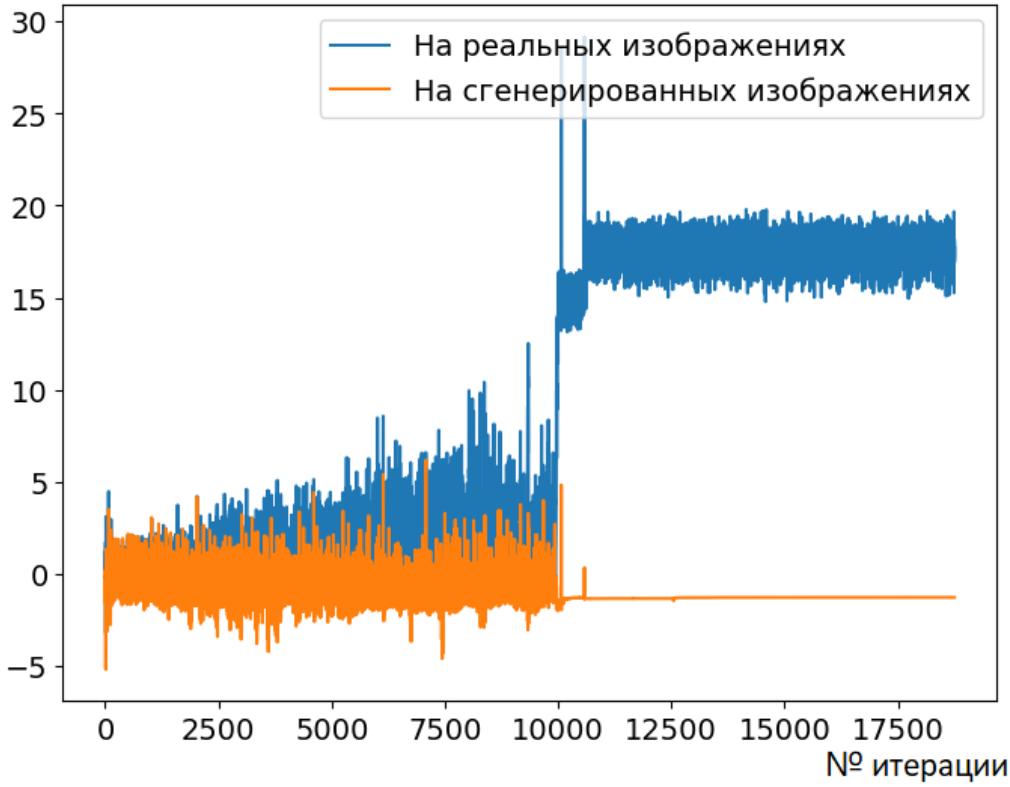


График усреднённых значений выходов дискриминатора SA-GAN на реальных и сгенерированных изображениях на наборе изображений КТ-срезов на каждой итерации обучения

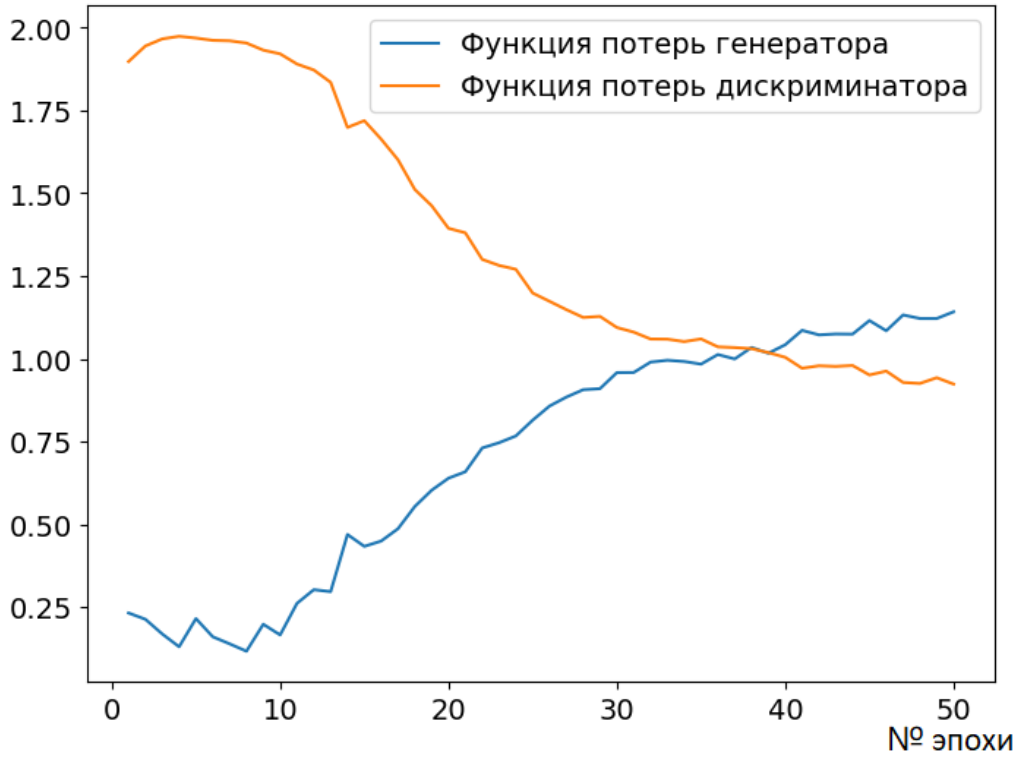


График функции потерь BIG-GAN-DEEP на наборе изображений рентгеновских снимков по эпохам

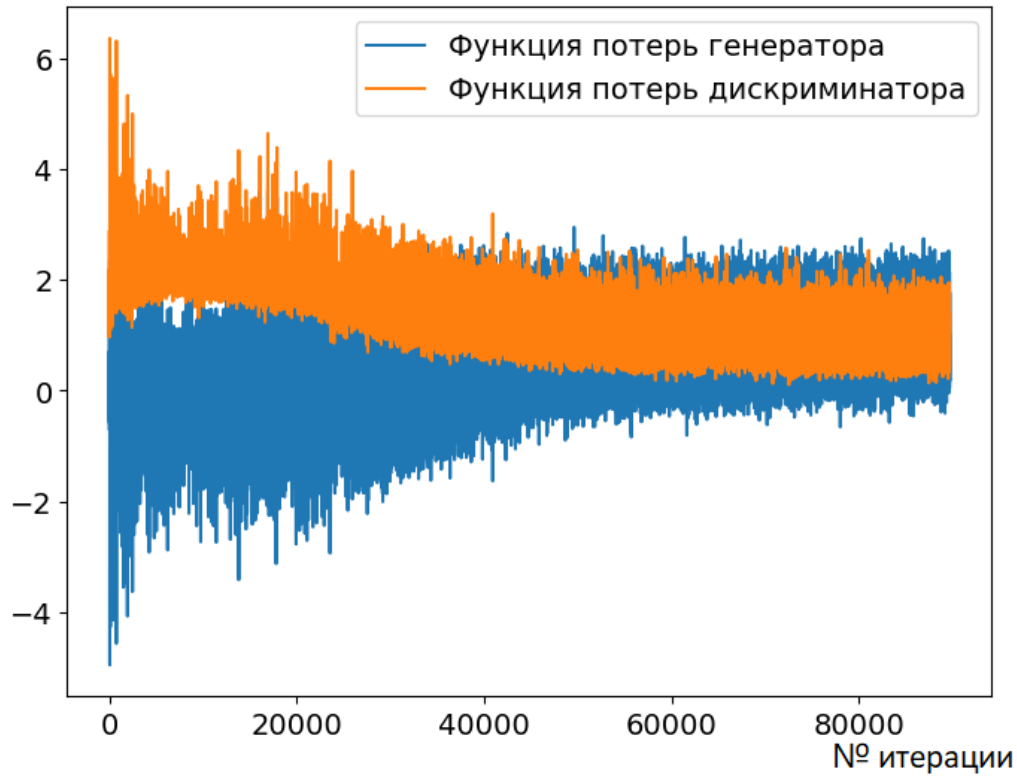


График функции потерь BIG-GAN-DEEP на наборе изображений рентгеновских снимков на каждой итерации обучения

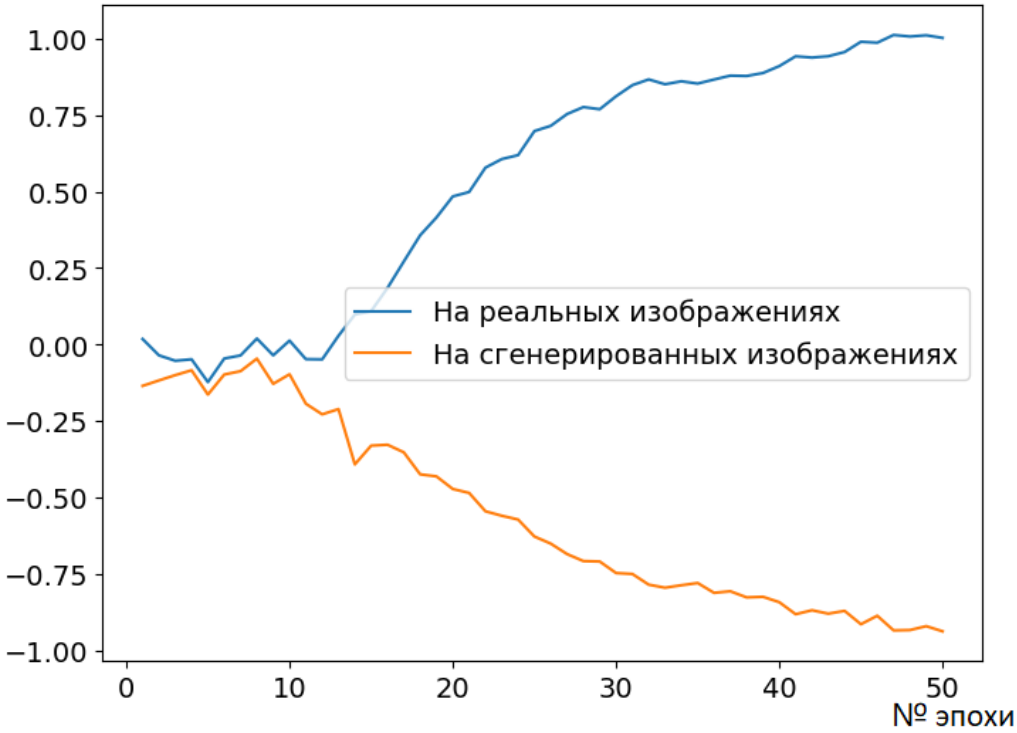


График усреднённых значений выходов дискриминатора BIG-GAN-DEEP на реальных и сгенерированных изображениях на наборе изображений рентгеновских снимков по эпохам

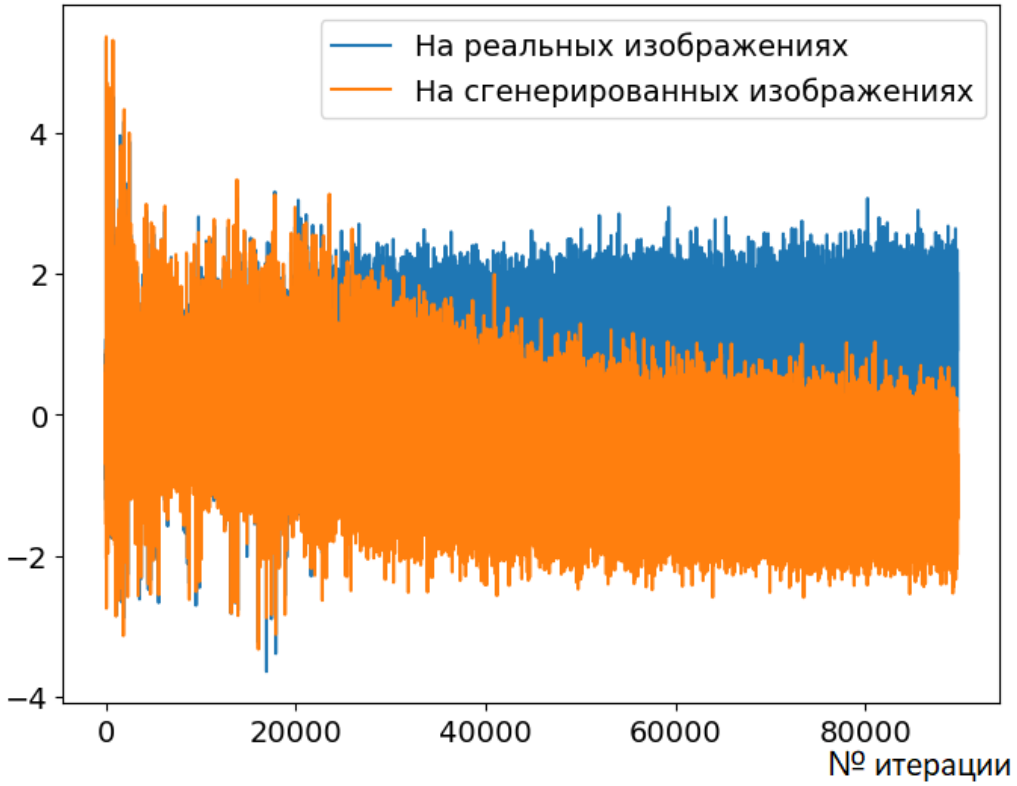


График усреднённых значений выходов дискриминатора BIG-GAN-DEEP на реальных и сгенерированных изображениях на наборе изображений рентгеновских снимков на каждой итерации обучения

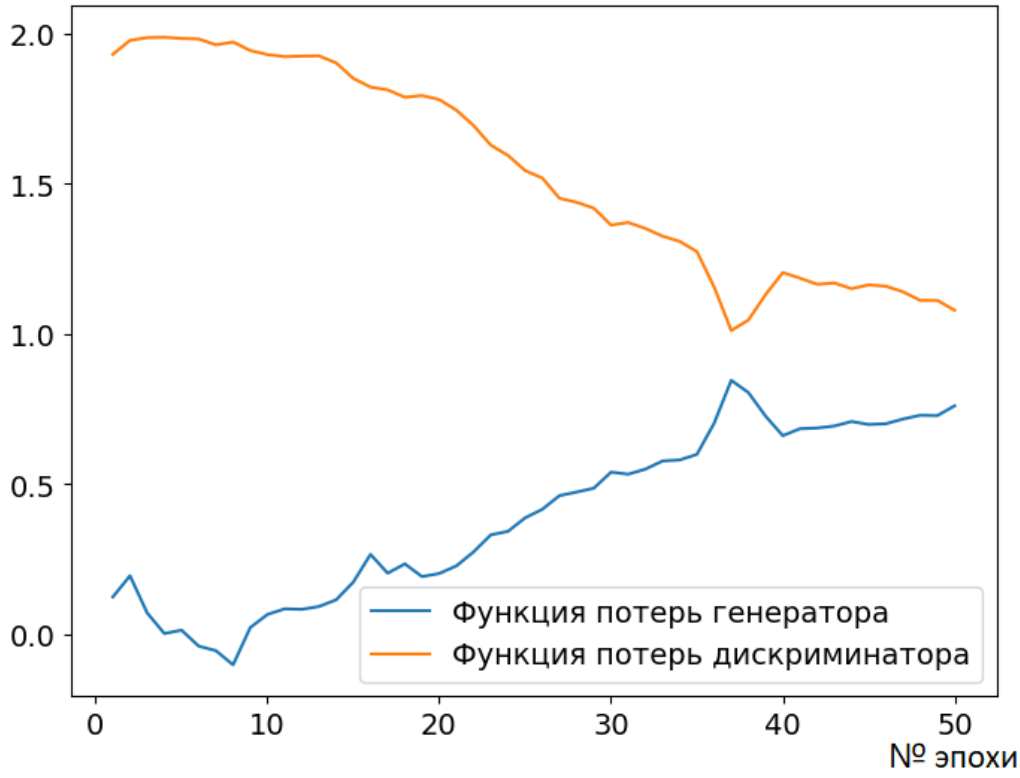


График функции потерь BIG-GAN-DEEP на наборе изображений КТ-срезов по эпохам

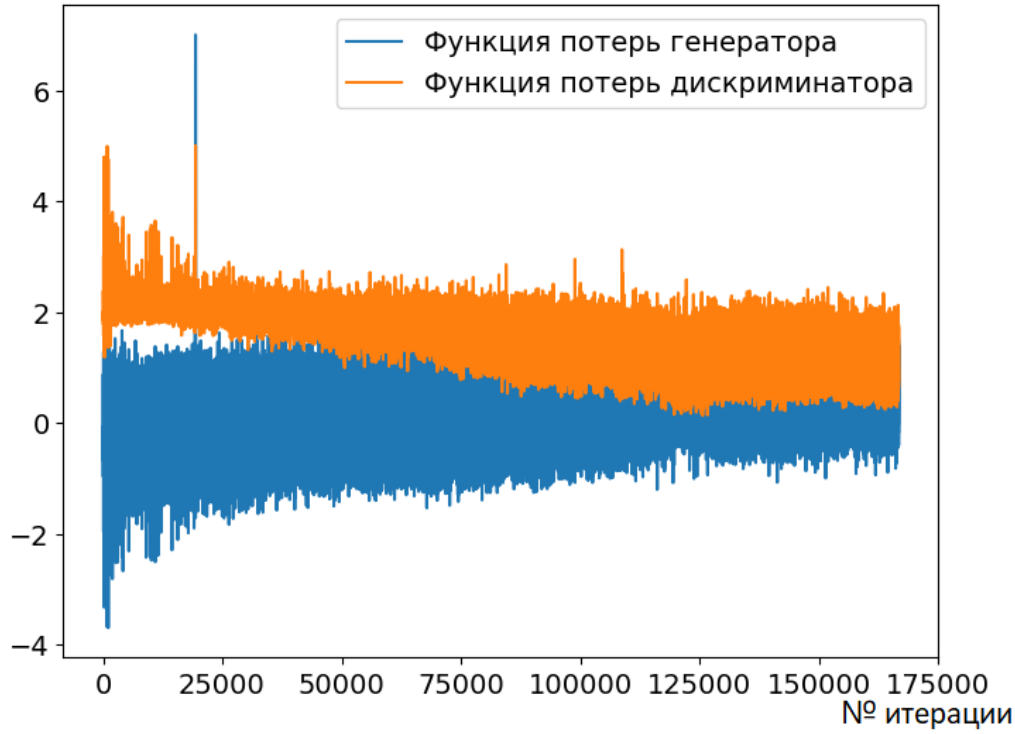


График функции потерь BIG-GAN-DEEP на наборе изображений КТ-срезов на каждой итерации обучения

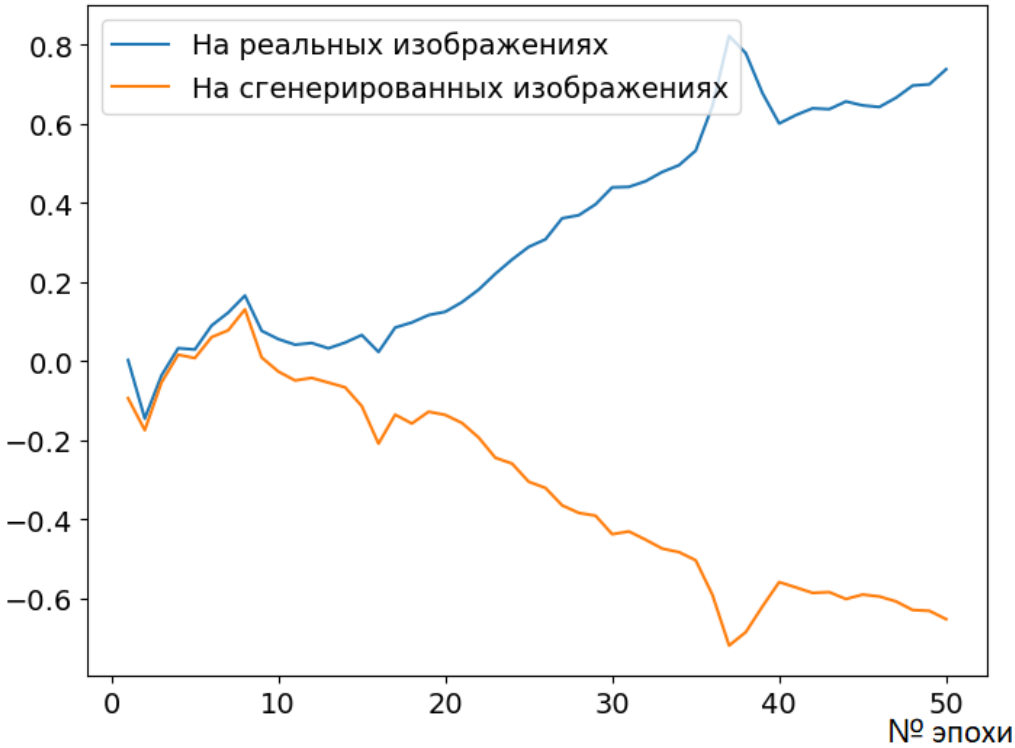


График усреднённых значений выходов дискриминатора BIG-GAN-DEEP на реальных и сгенерированных изображениях на наборе изображений КТ-срезов по эпохам

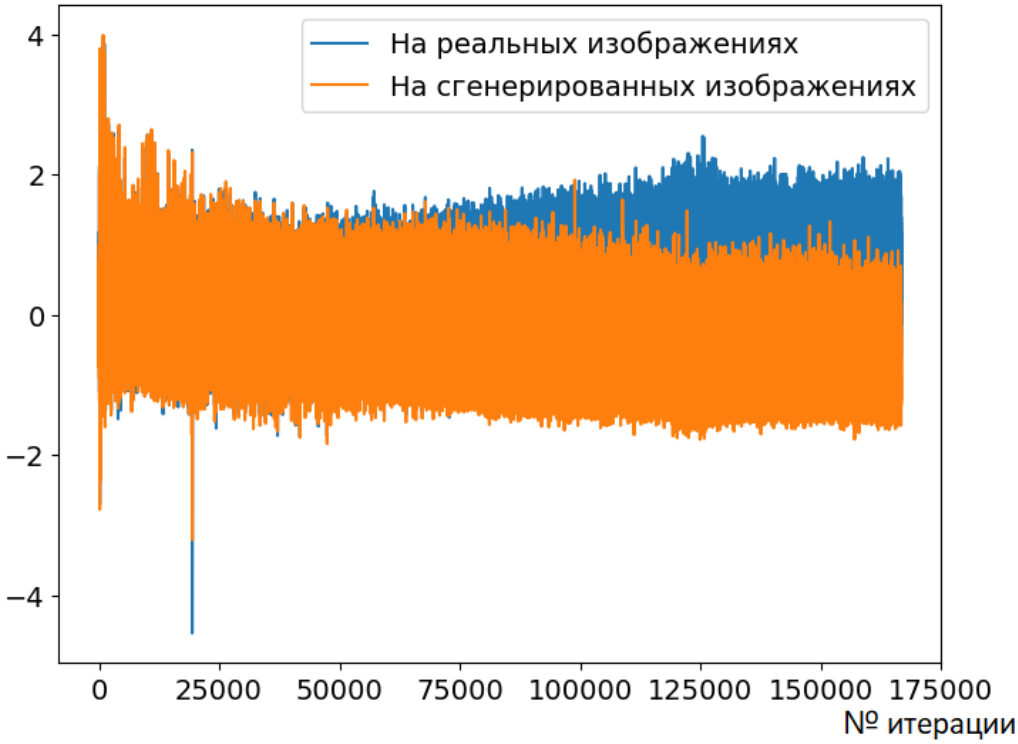


График усреднённых значений выходов дискриминатора BIG-GAN-DEEP на реальных и сгенерированных изображениях на наборе изображений КТ-срезов на каждой итерации обучения

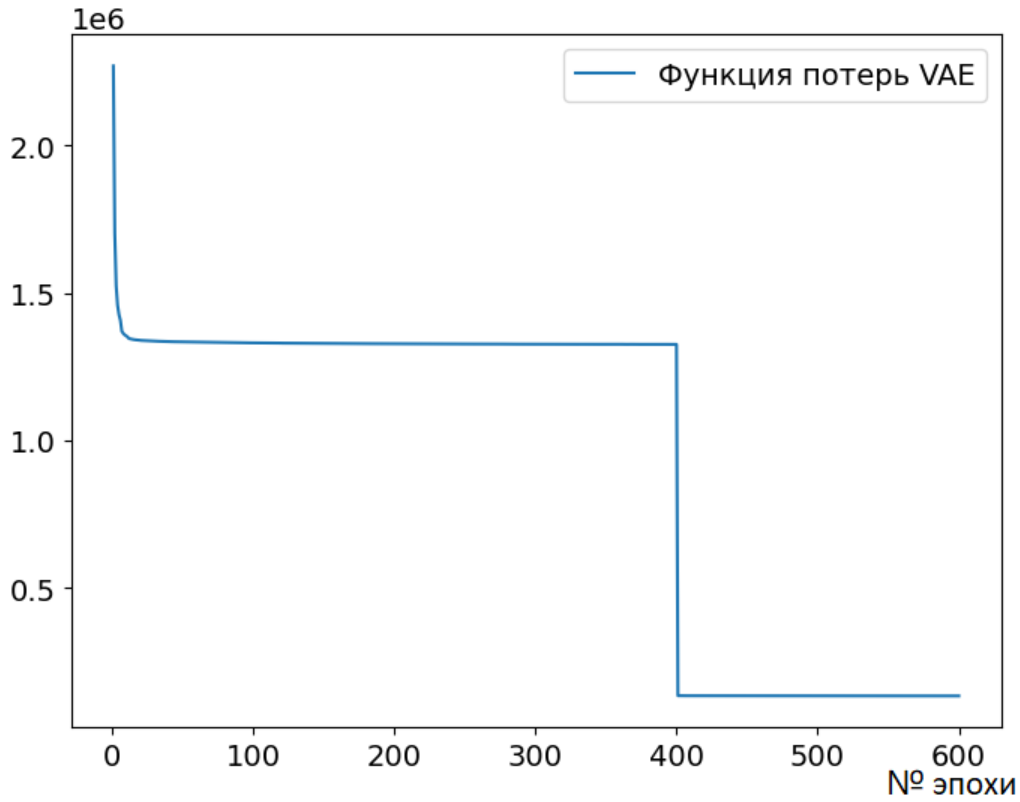


График функции потерь VAE на наборе изображений масок лёгких рентгеновских снимков по эпохам

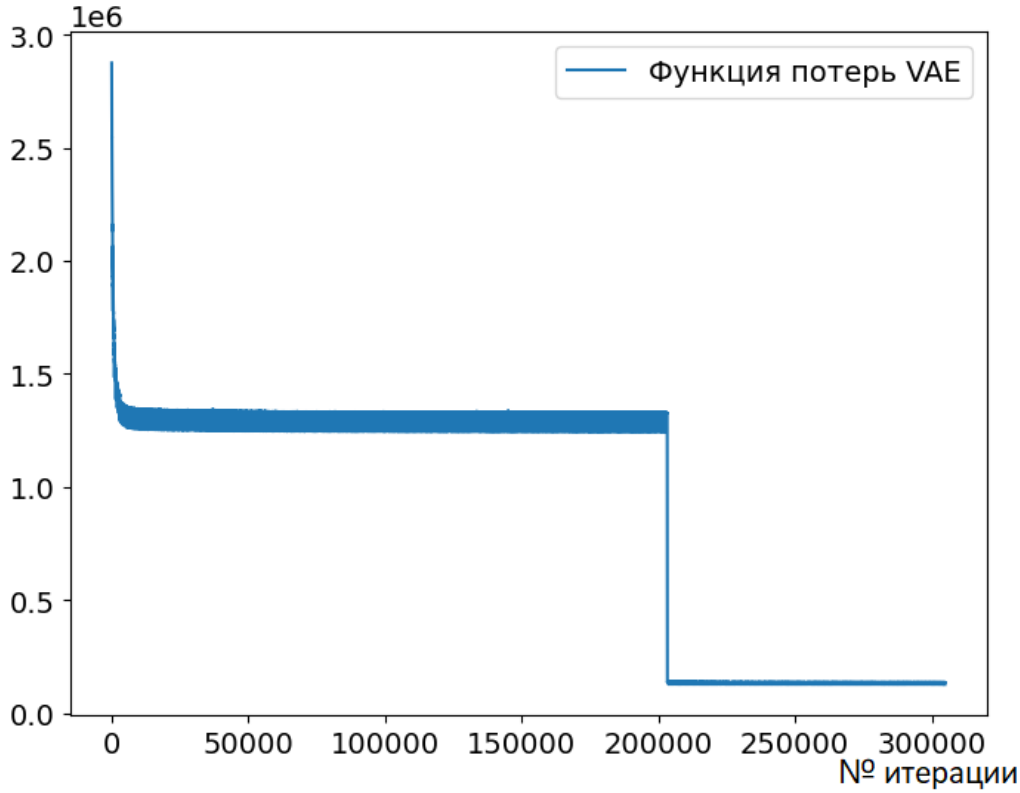


График функции потерь VAE на наборе изображений масок лёгких рентгеновских снимков на каждой итерации обучения

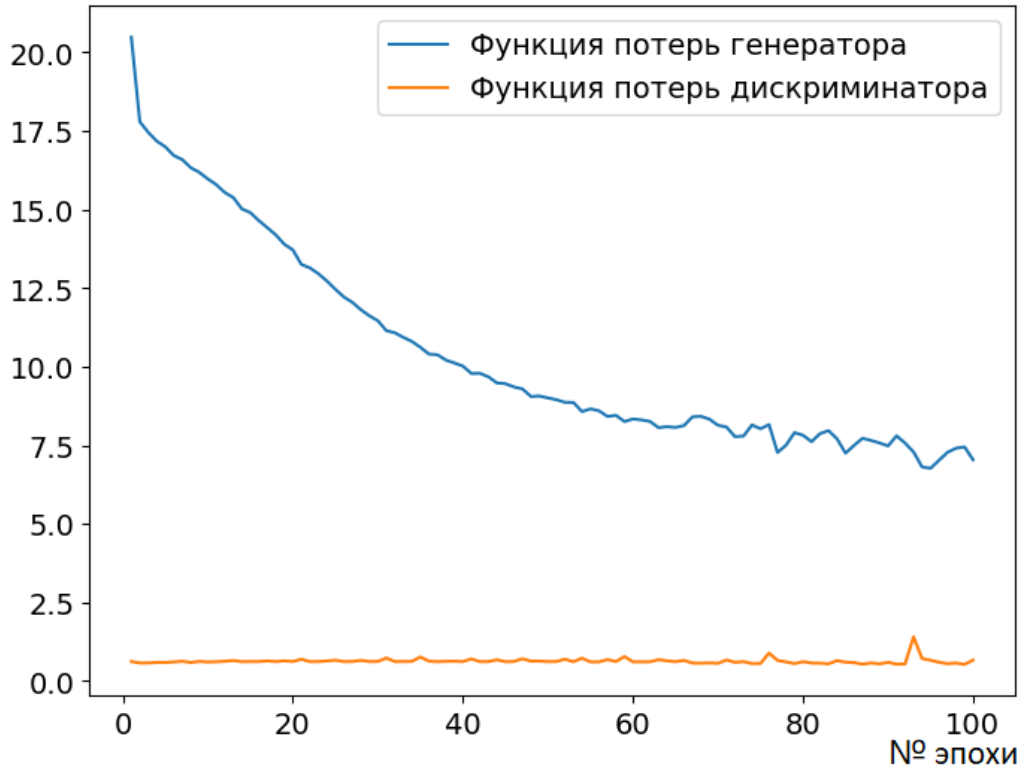


График функции потерь CGAN на наборе изображений рентгеновских снимков по эпохам

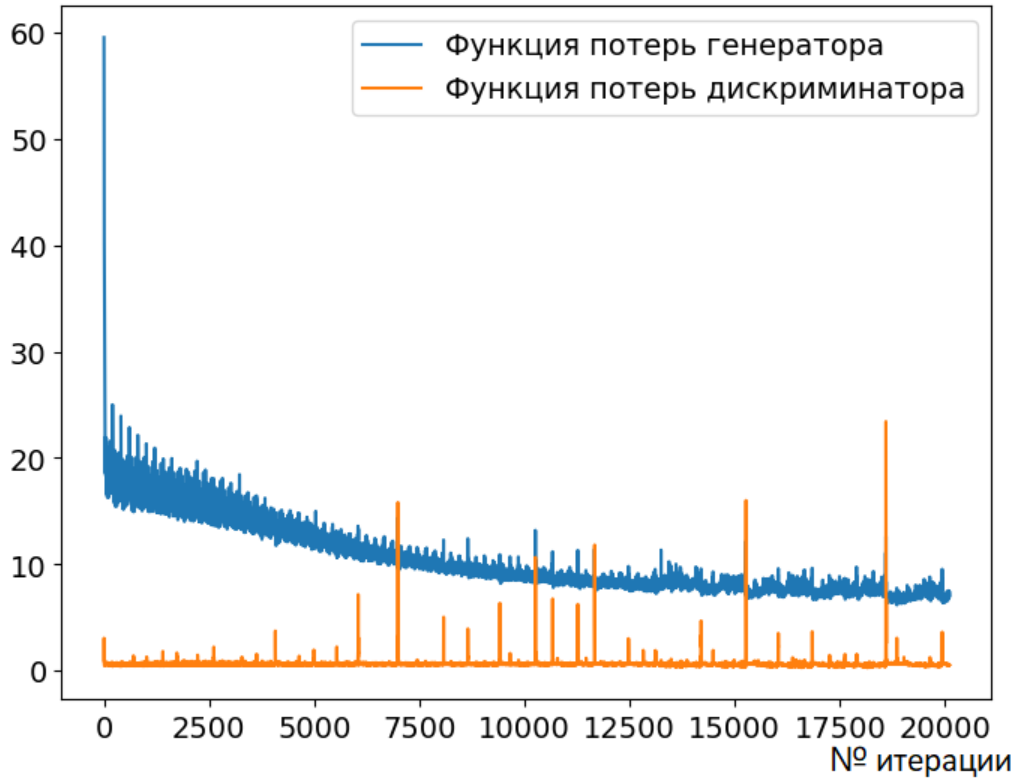


График функции потерь CGAN на наборе изображений рентгеновских снимков на каждой итерации обучения

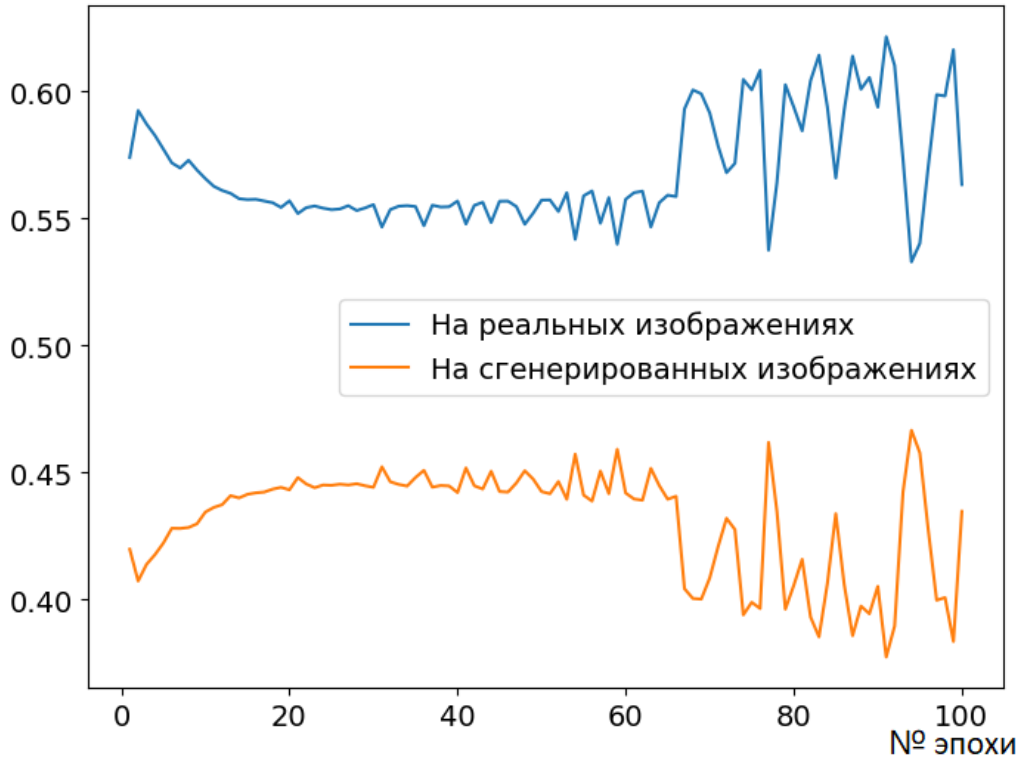


График усреднённых значений выходов дискриминатора CGAN на реальных и сгенерированных изображениях на наборе изображений рентгеновских снимков по эпохам

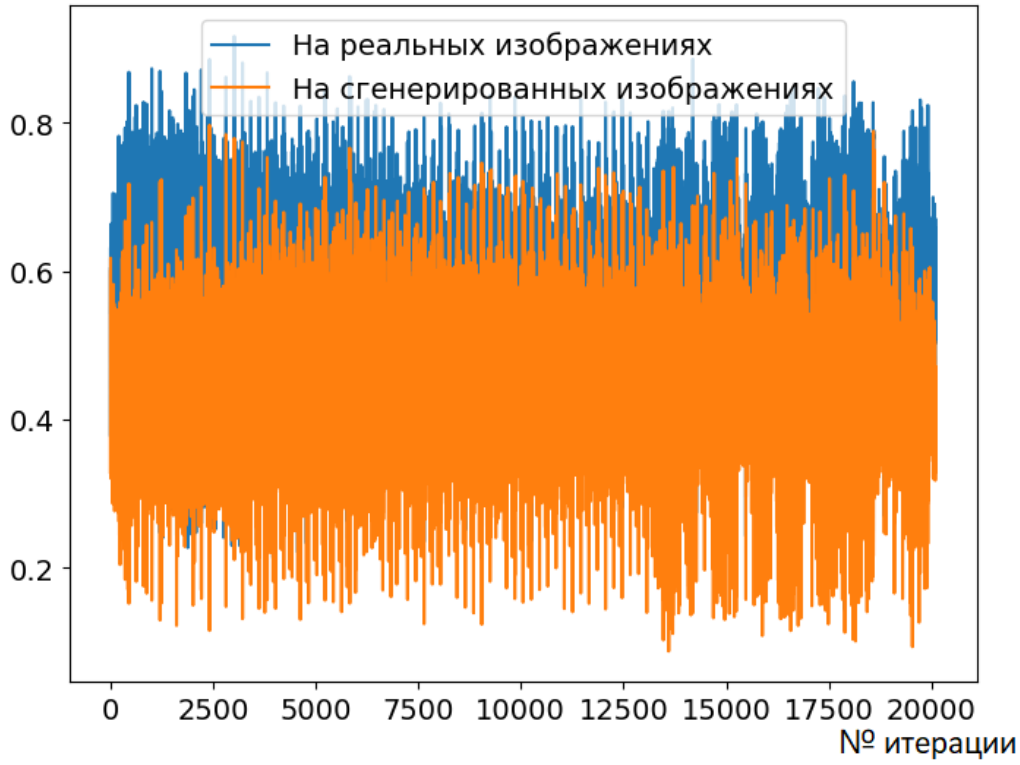


График усреднённых значений выходов дискриминатора CGAN на реальных и сгенерированных изображениях на наборе изображений рентгеновских снимков на каждой итерации обучения

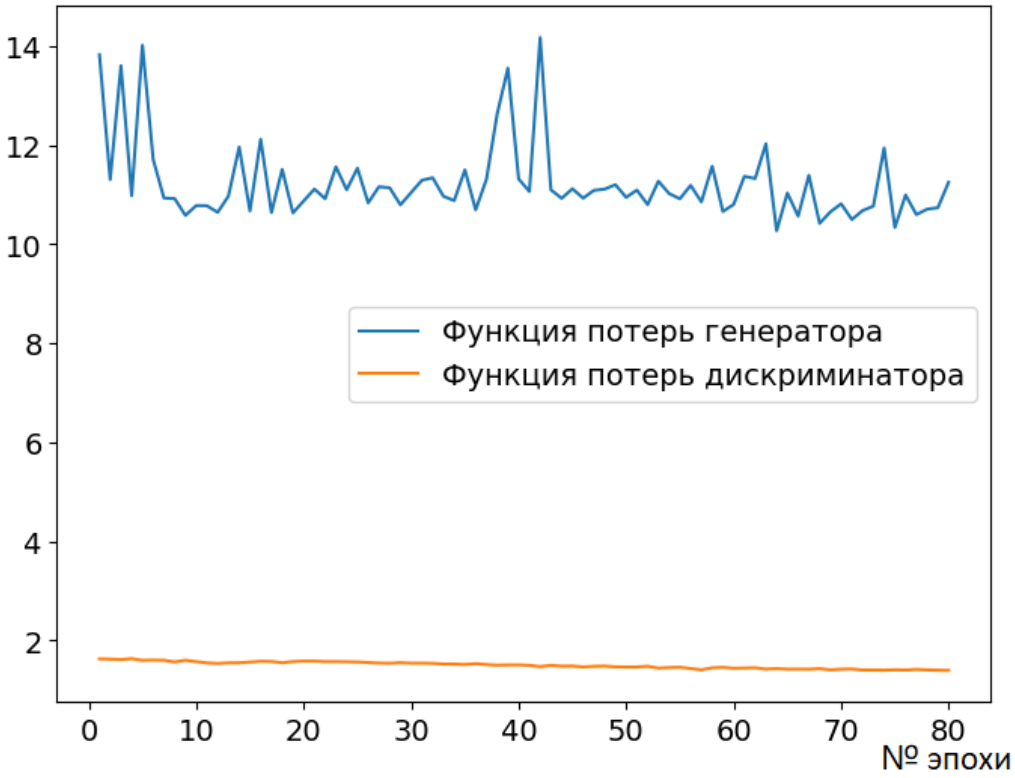


График функции потерь SPADE-GAN на наборе изображений рентгеновских снимков по эпохам

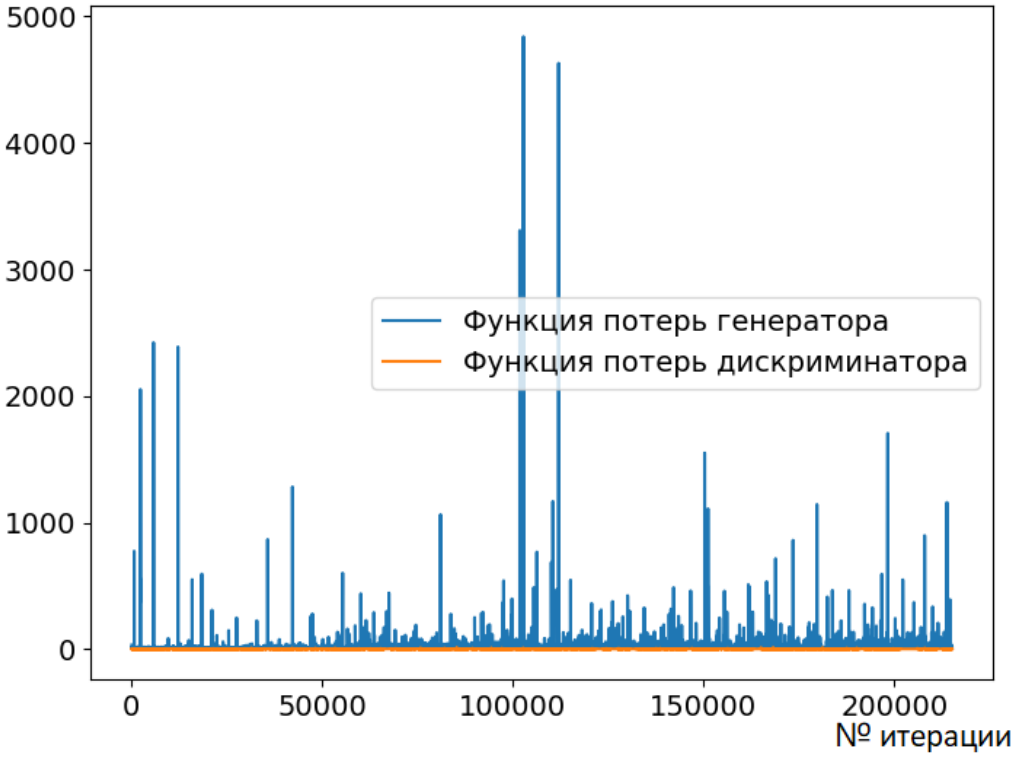


График функции потерь SPADE-GAN на наборе изображений рентгеновских снимков на каждой итерации обучения

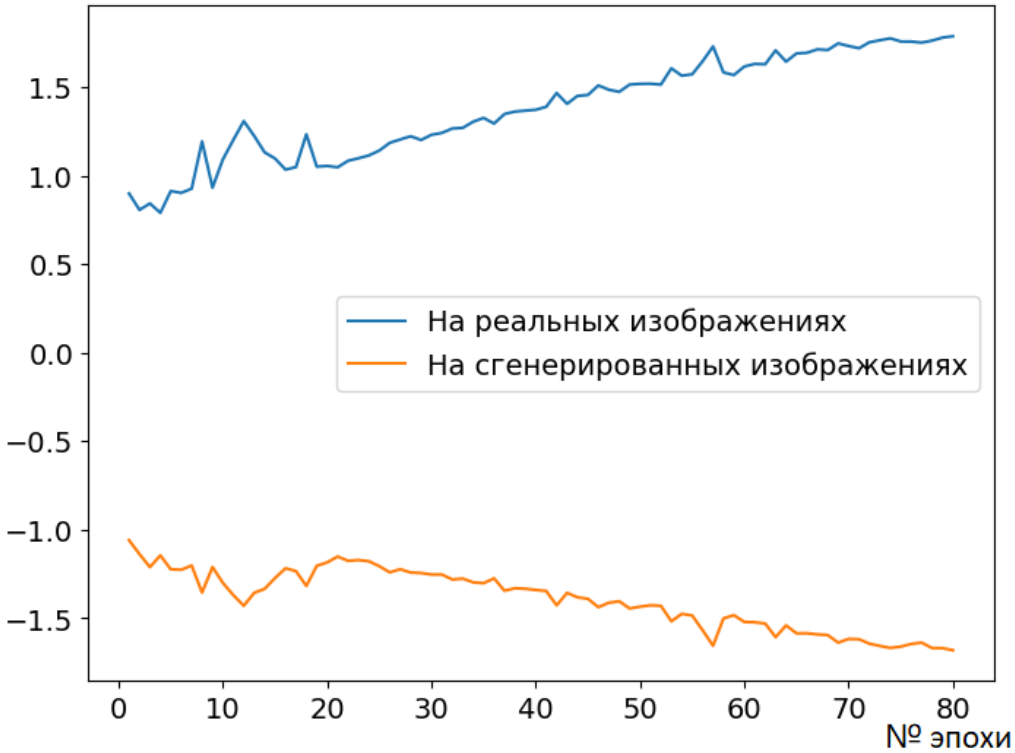


График усреднённых значений выходов дискриминатора SPADE-GAN на реальных и сгенерированных изображениях на наборе изображений рентгеновских снимков по эпохам

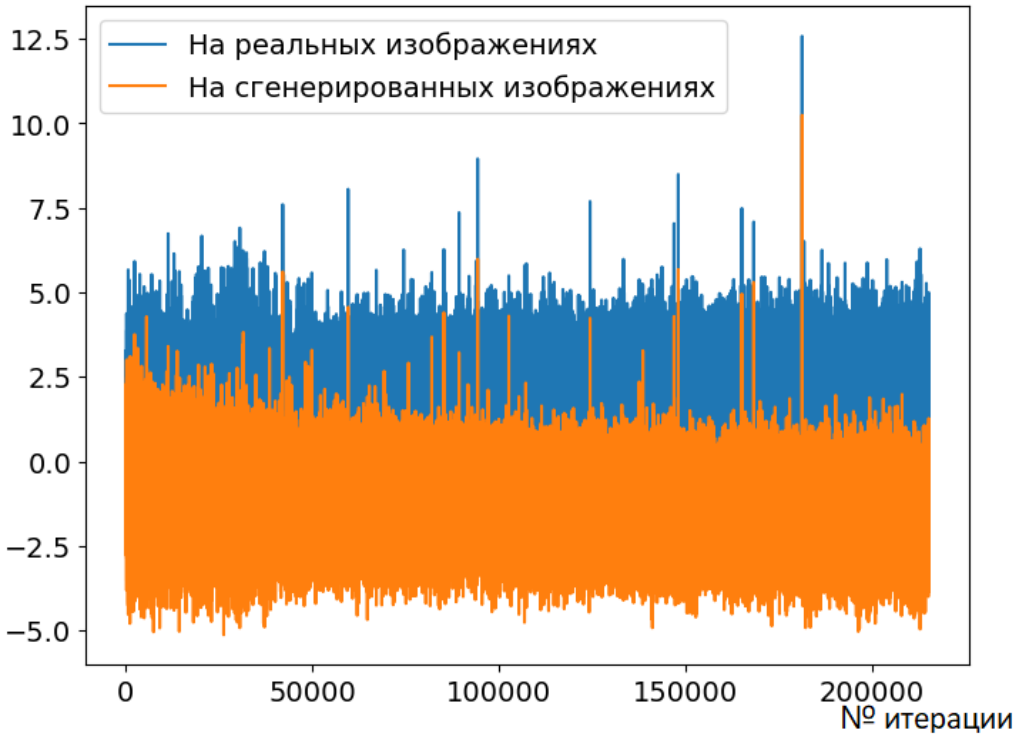
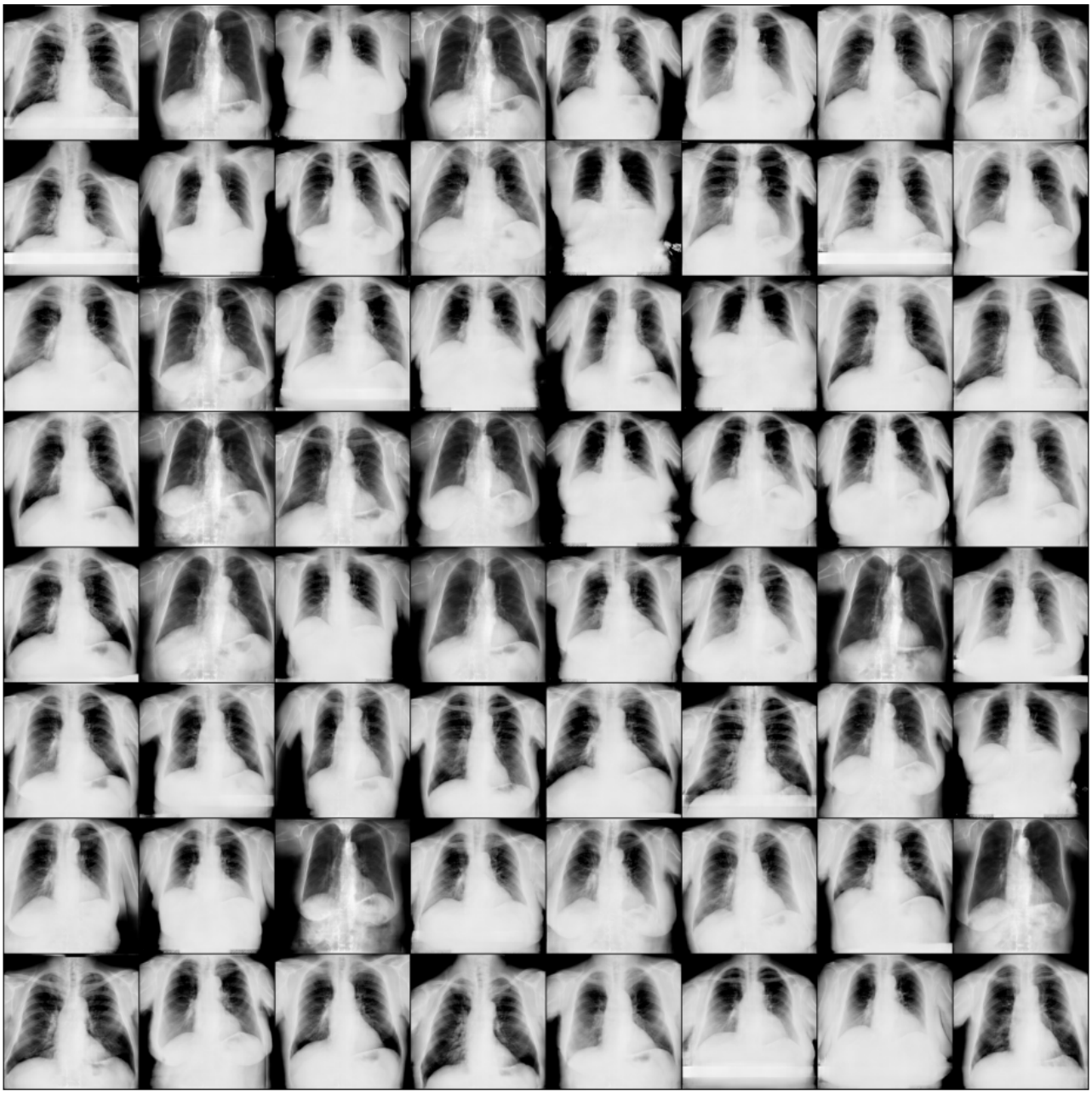


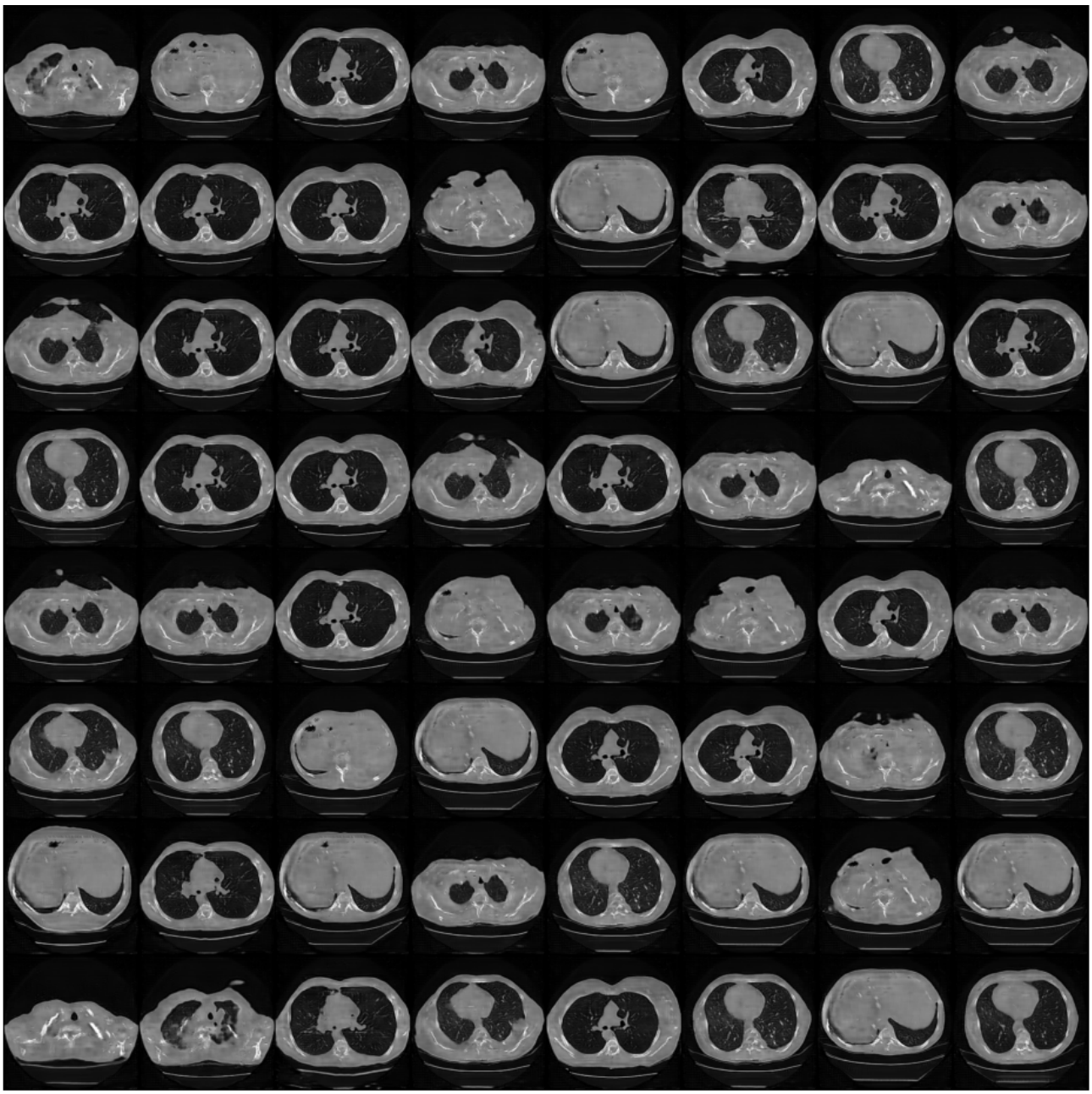
График усреднённых значений выходов дискриминатора SPADE-GAN на реальных и сгенерированных изображениях на наборе изображений рентгеновских снимков на каждой итерации обучения

ПРИЛОЖЕНИЕ В

Примеры изображений, сгенерированных генеративно-состязательными моделями

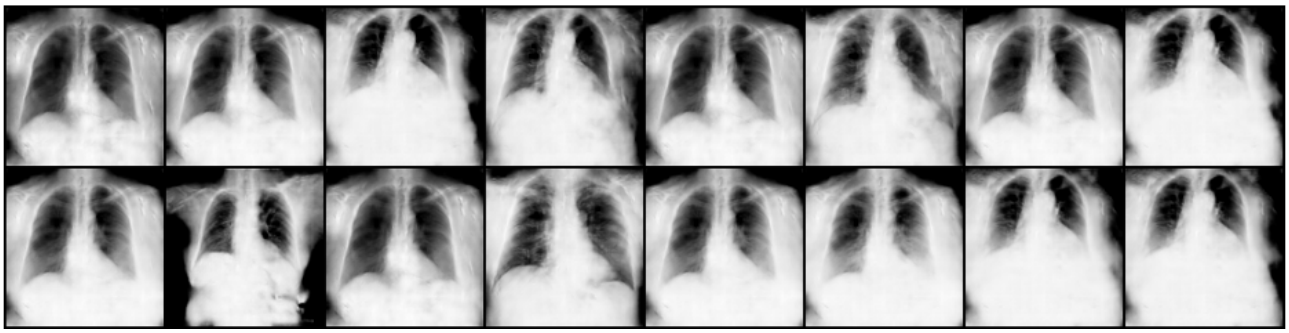


Результат генерации модели DC-GAN после обучения на наборе изображений рентгеновских снимков

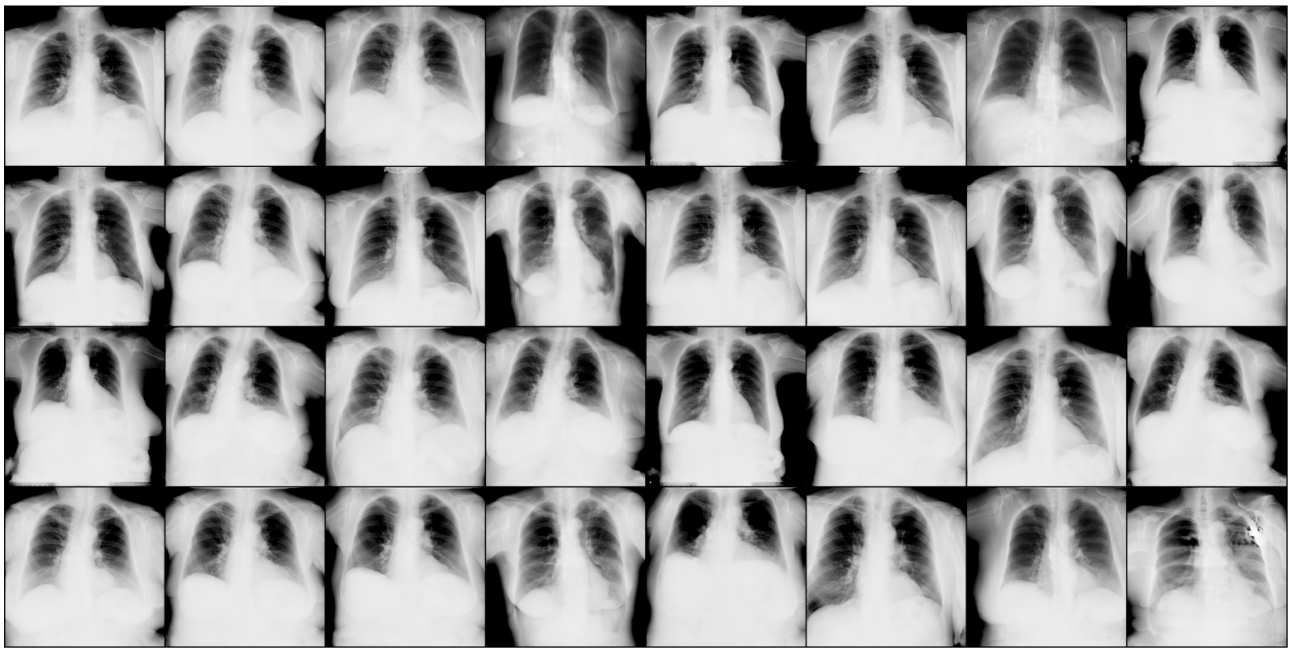


Результат генерации модели DC-GAN после обучения на наборе изображений КТ-срезов

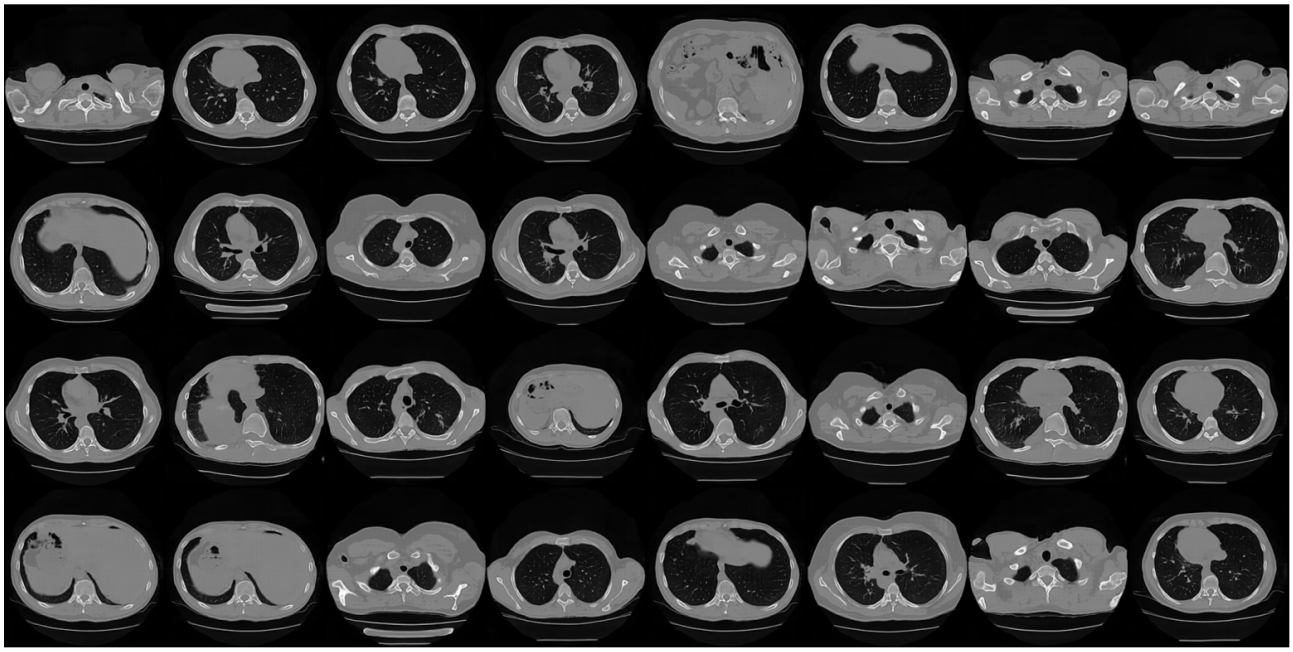
Результат генерации модели SA-GAN после обучения на наборе изображений рентгеновских снимков разрешения 64×64



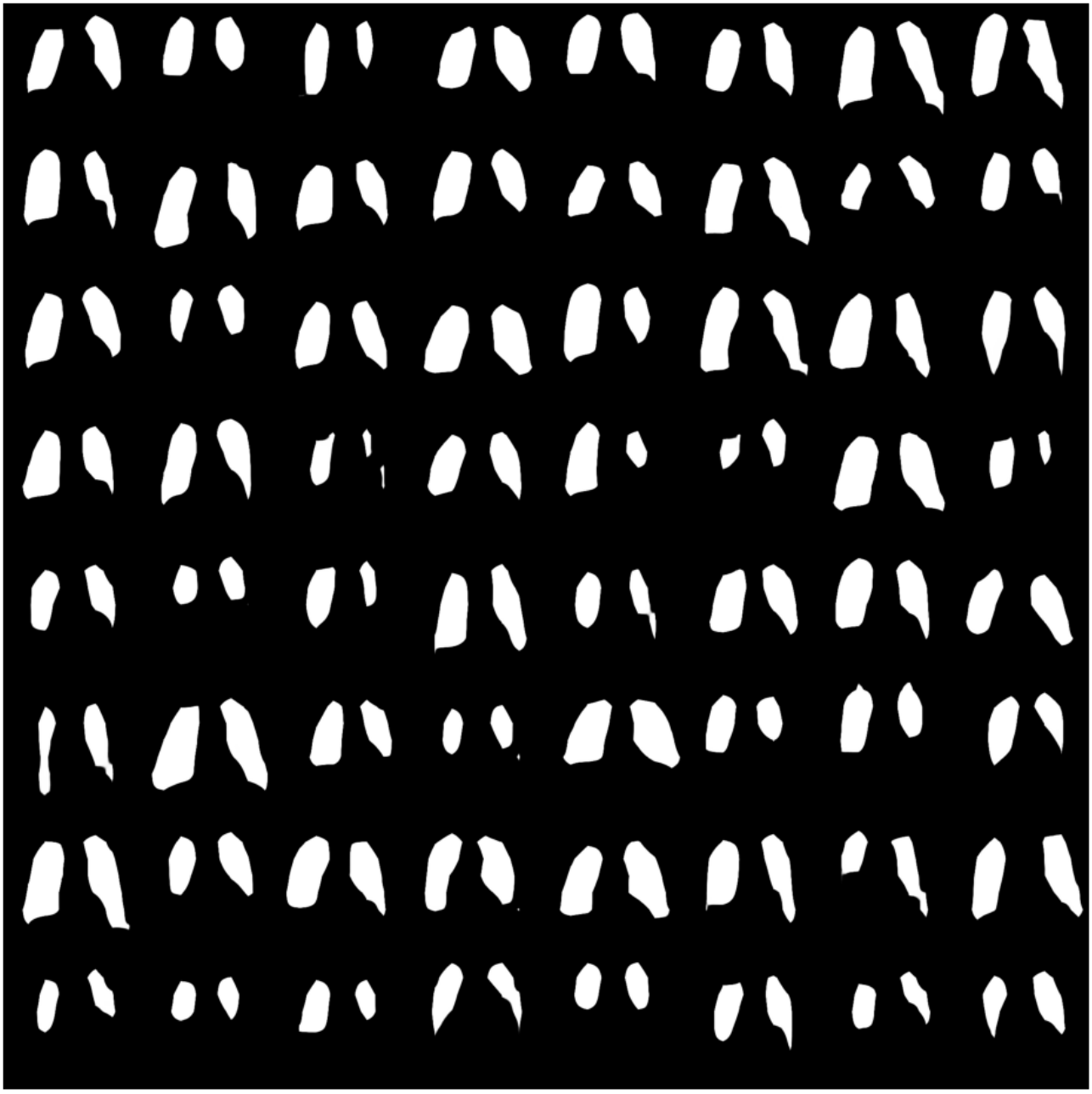
Результат генерации модели SA-GAN после обучения на наборе изображений рентгеновских снимков разрешения 128×128



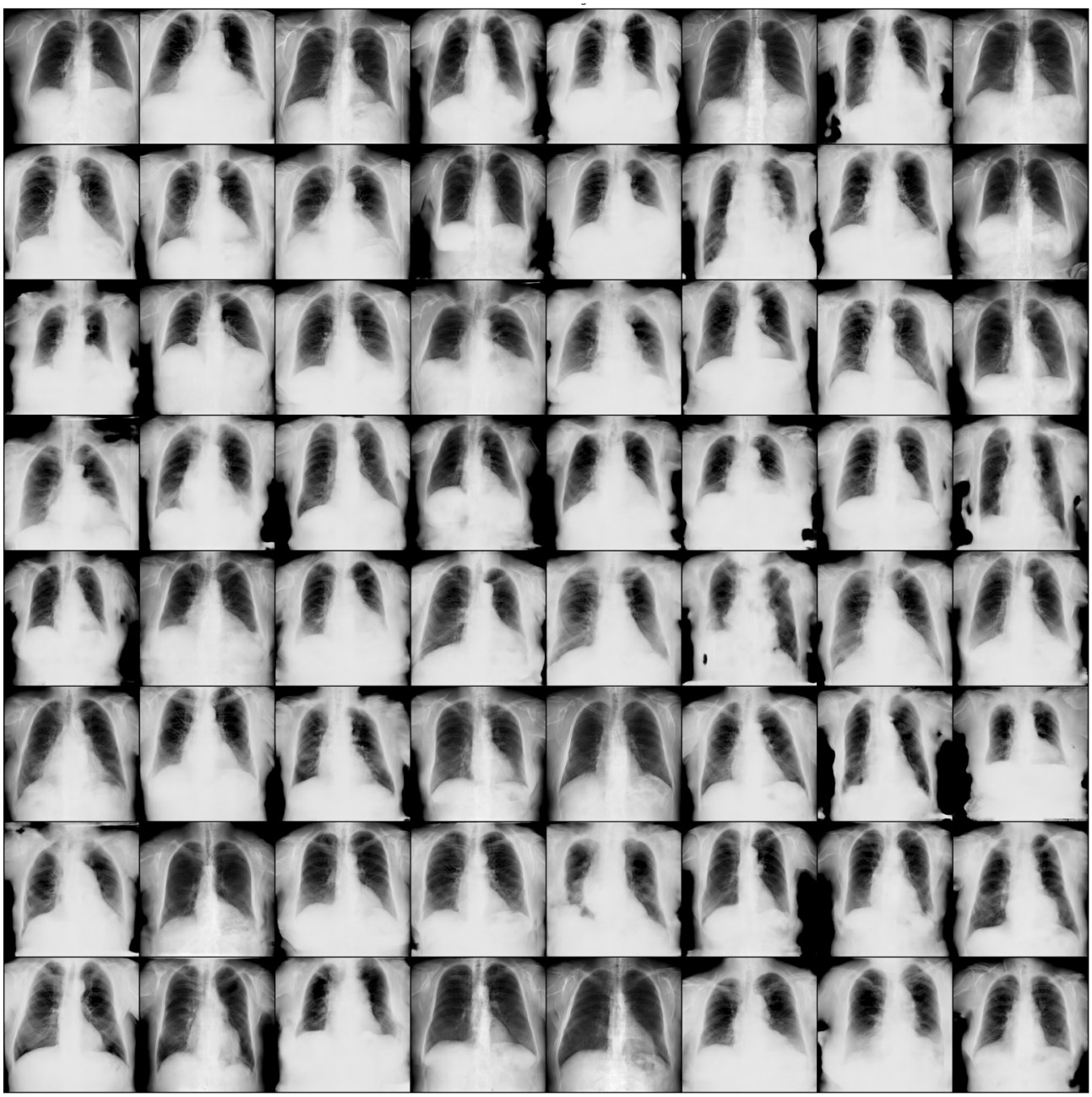
Результат генерации модели BIG-GAN-DEEP после обучения на наборе изображений рентгеновских снимков



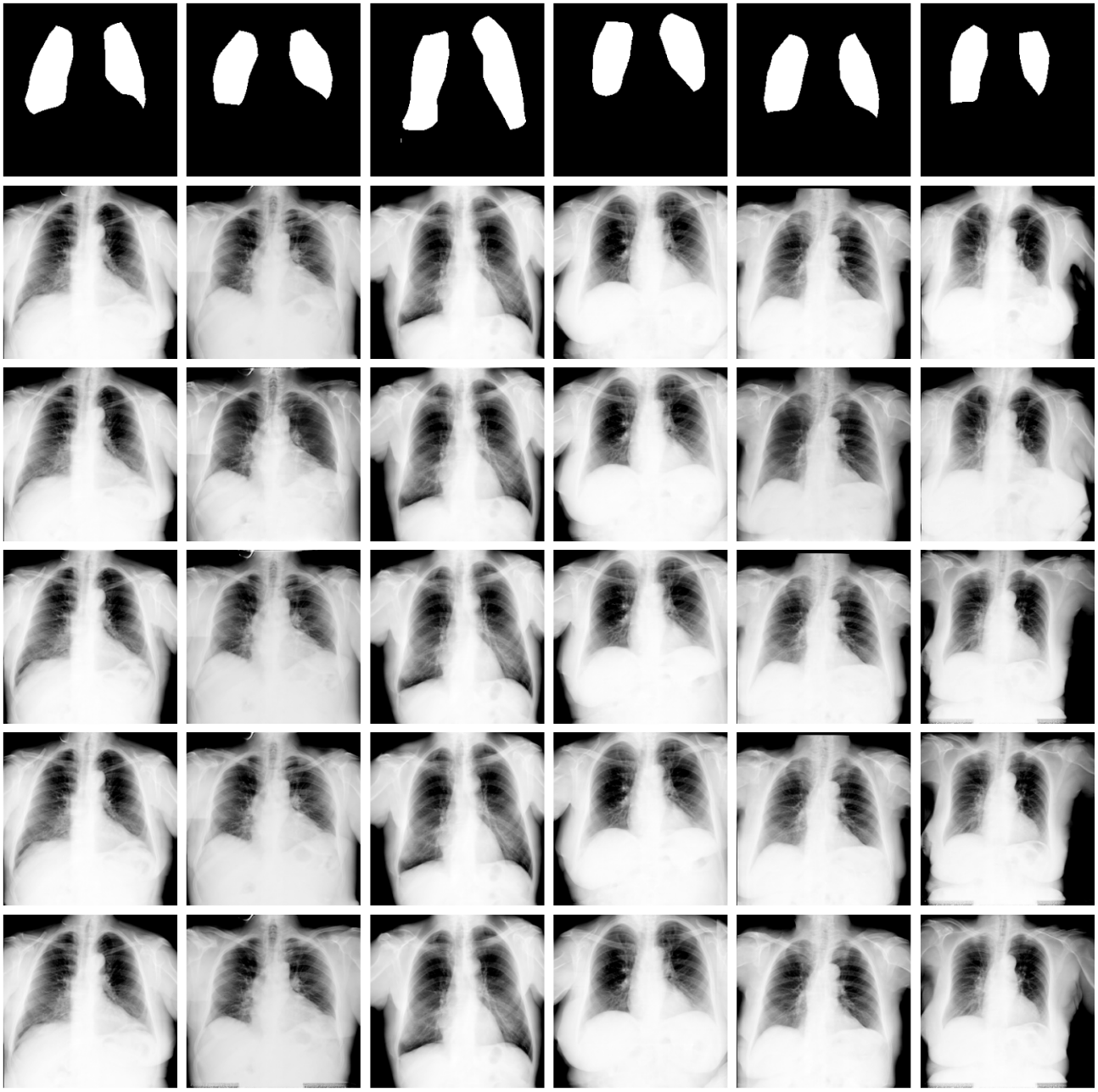
Результат генерации модели BIG-GAN-DEEP после обучения на наборе изображений КТ-срезов



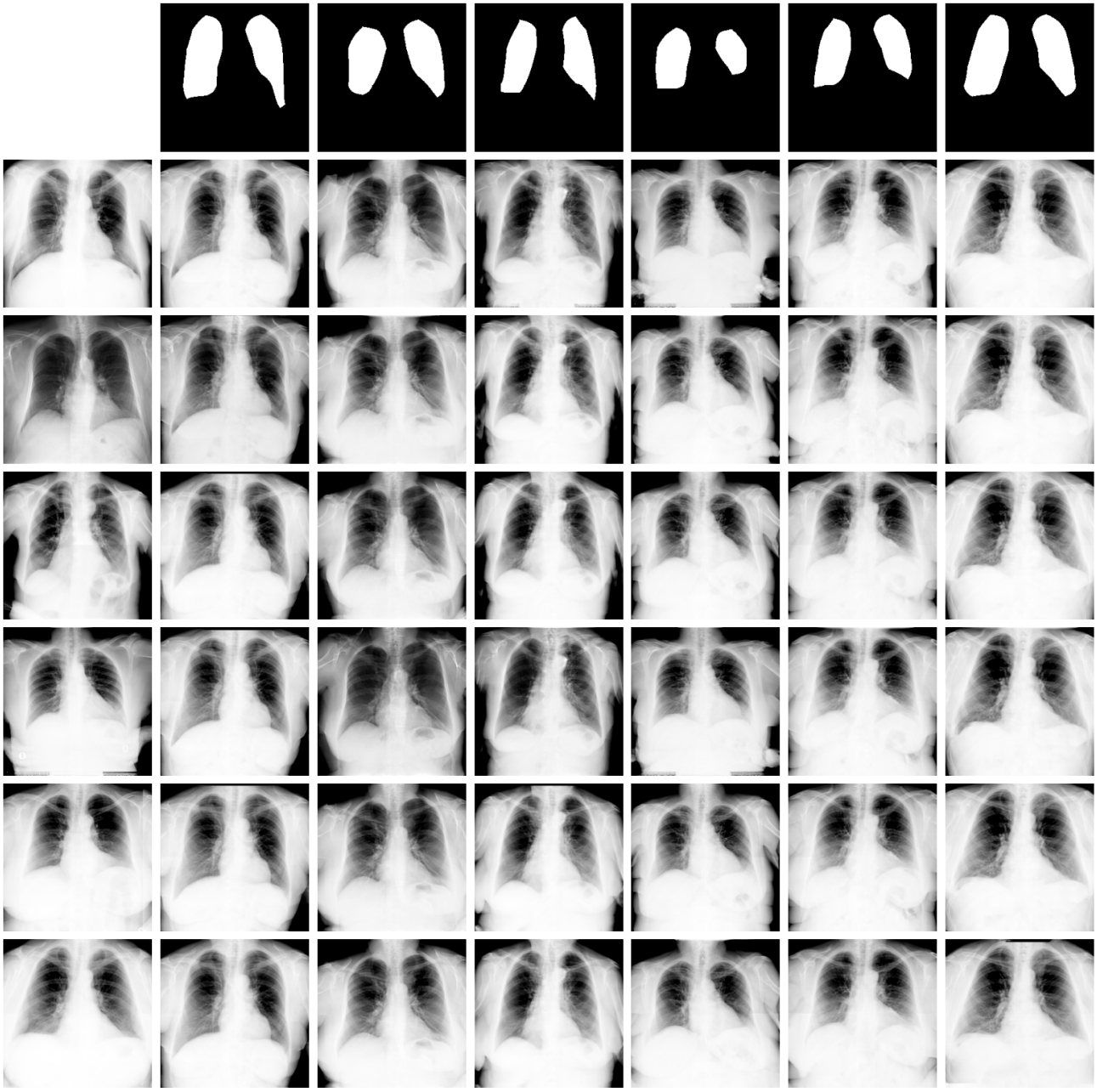
Результат генерации модели VAE после обучения на наборе изображений масок лёгких рентгеновских снимков



Результат генерации ансамбля генеративных моделей VAE и CGAN после обучения на наборе изображений рентгеновских снимков



Результат генерации модели SPADE-GAN после обучения на наборе изображений рентгеновских снимков. Способ генерации: только маска



Результат генерации модели SPADE-GAN после обучения на наборе изображений рентгеновских снимков. Способ генерации: маска и исходное изображение. В первом столбце расположены исходные изображения, в первой строке расположены исходные маски. На пересечении стоят результаты генерации для соответствующих исходных масок и изображений