TOMSK POLYTECHNIC UNIVERSITY ТОМСКИЙ ПОЛИТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ

Министерство науки и высшего образования Российской Федерации
федеральное государственное автономное образовательное учреждение высшего образования
«Национальный исследовательский Томский политехнический университет» (ТПУ)

Школа Инженерная школа информационных технологий и робототехники
Направление подготовки 09.03.04 Программная инженерия
ООП/ОПОП Разработка программно-информационных систем
Отделение школы (НОЦ) Отделение информационных технологий

ВЫПУСКНАЯ КВАЛИФИКАЦИОННАЯ РАБОТА БАКАЛАВРА

Тема работы
Разработка генеративно-состязательной нейросети для генерации КТ-изображений грудной клетки человека

УДК 004.032.26:004.853:616-073

Обучающийся

Группа	ФИО	Подпись	Дата
8К93	Кузнецов Илья Евгеньевич		

Руководитель ВКР

Должность	ФИО	Ученая степень, звание	Подпись	Дата
Доцент ОИТ ИШИТР	Аксенов Сергей Владимирович	к.т.н.		

КОНСУЛЬТАНТЫ ПО РАЗДЕЛАМ:

По разделу «Финансовый менеджмент, ресурсоэффективность, ресурсосбережение»

Должность	ФИО	Ученая степень, звание	Подпись	Дата
Профессор ОСГН	Гасанов Магеррам Али оглы	Доктор экономических наук		

По разделу «Социальная ответственность»

Должность	ФИО	Ученая степень, звание	Подпись	Дата
Старший преподаватель ООД ШБИП	Мезенцева Ирина Леонидовна			

ДОПУСТИТЬ К ЗАЩИТЕ:

Руководитель ООП/ОПОП, должность	ФИО	Ученая степень, звание	Подпись	Дата
Доцент ОИТ ИШИТР	Чердынцев Евгений Сергеевич	к.т.н		

Томск – 2023 г

ПЛАНИРУЕМЫЕ РЕЗУЛЬТАТЫ ОБУЧЕНИЯ ООП

Код компетенции	Наименование компетенции
УК(У)-1	Способен осуществлять поиск, критический анализ и синтез информации, применять системный подход для решения поставленных задач.
УК(У)-2	Способен определять круг задач в рамках поставленной цели и выбирать оптимальные способы их решения, исходя из действующих правовых норм, имеющихся ресурсов и ограничений.
УК(У)-3	Способен осуществлять социальное взаимодействие и реализовывать свою роль в команде.
УК(У)-4	Способен осуществлять деловую коммуникацию в устной и письменной формах на государственном языке Российской Федерации и иностранном(-ых) языке(-ах).
УК(У)-5	Способен воспринимать межкультурное разнообразие общества в социально-историческом, этическом и философском контекстах
УК(У)-6	Способен управлять своим временем, выстраивать и реализовывать траекторию саморазвития на основе принципов образования в течение всей жизни.
ОПК(У)-1	Способен применять естественнонаучные и общеинженерные знания, методы математического анализа и моделирования, теоретического и экспериментального исследования в профессиональной деятельности.
ОПК(У)-2	Способен использовать современные информационные технологии и программные средства, в том числе отечественного производства, при решении задач профессиональной деятельности.
ОПК(У)-3	Способен решать стандартные задачи профессиональной деятельности на основе информационной и библиографической культуры с применением информационно-коммуникационных технологий и с учетом основных требований информационной безопасности.
ОПК(У)-4	Способен участвовать в разработке стандартов, норм и правил, а также технической документации, связанной с профессиональной деятельностью.
ОПК(У)-5	Способен инсталлировать программное и аппаратное обеспечение для информационных и автоматизированных систем.
ПК(У)-1	Способен выполнять интеграцию программных модулей и компонент.
ПК(У)-2	Владение навыками моделирования, анализа и использования формальных методов конструирования программного обеспечения.

TOMSK POLYTECHNIC UNIVERSITY ТОМСКИЙ ПОЛИТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ

Министерство науки и высшего образования Российской Федерации
федеральное государственное автономное образовательное учреждение высшего образования
«Национальный исследовательский Томский политехнический университет» (ТПУ)

Школа Инженерная школа информационных технологий и робототехники
Направление подготовки (ООП/ОПОП) 09.03.04 Программная инженерия
Отделение школы (НОЦ) Отделение информационных технологий

УТВЕРЖДАЮ:
Руководитель ООП/ОПОП
_____ _____ Чердынцев Е.С.
(Подпись) (Дата) (ФИО)

ЗАДАНИЕ
на выполнение выпускной квалификационной работы

Обучающийся:

Группа	**ФИО**
8К93	Кузнецов Илья Евгеньевич

Тема работы:

Разработка генеративно-состязательной нейросети для генерации КТ-изображений грудной клетки человека	
Утверждена приказом директора (дата, номер)	

Срок сдачи обучающимся выполненной работы:	

ТЕХНИЧЕСКОЕ ЗАДАНИЕ:

Исходные данные к работе *(наименование объекта исследования или проектирования; производительность или нагрузка; режим работы (непрерывный, периодический, циклический и т. д.); вид сырья или материал изделия; требования к продукту, изделию или процессу; особые требования к функционированию (эксплуатации) объекта или изделия в плане безопасности эксплуатации, влияния на окружающую среду, энергозатратам; экономический анализ и т. д.)*	Объектом исследования является генеративно-состязательная нейросеть для генерации кт-изображений грудной клетки.

Перечень разделов пояснительной записки подлежащих исследованию, проектированию и разработке *(аналитический обзор литературных источников с целью выяснения достижений мировой науки техники в рассматриваемой области; постановка задачи исследования, проектирования, конструирования; содержание процедуры исследования, проектирования, конструирования; обсуждение результатов выполненной работы; наименование дополнительных разделов, подлежащих разработке; заключение по работе)*	1. Изучение различных архитектур генеративно-состязательной нейросети. 2. Реализация и обучение различных архитектур генеративно-состязательной нейросети. 3. Финансовый менеджмент, ресурсоэффективность и ресурсосбережение. 4. Социальная ответственность.
Перечень графического материала *(с точным указанием обязательных чертежей)*	1. Рисунки, демонстрирующие результаты работы.

Консультанты по разделам выпускной квалификационной работы *(с указанием разделов)*

Раздел	Консультант
Финансовый менеджмент, ресурсоэффективность, ресурсосбережение	Гасанов Магеррам Али оглы
Социальная ответственность	Мезенцева Ирина Леонидовна

Дата выдачи задания на выполнение выпускной квалификационной работы по линейному графику	

Задание выдал руководитель:

Должность	ФИО	Ученая степень, звание	Подпись	Дата
Доцент ОИТ ИШИТР	Аксенов Сергей Владимирович	к.т.н.		

Задание принял к исполнению обучающийся:

Группа	ФИО	Подпись	Дата
8К93	Кузнецов Илья Евгеньевич		

TOMSK POLYTECHNIC UNIVERSITY ТОМСКИЙ ПОЛИТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ

Министерство науки и высшего образования Российской Федерации
федеральное государственное автономное образовательное учреждение высшего образования
«Национальный исследовательский Томский политехнический университет» (ТПУ)

Школа Инженерная школа информационных технологий и робототехники
Направление подготовки 09.03.04 «Программная инженерия»
Уровень образования Бакалавриат
Отделение школы (НОЦ) Отделение информационных технологий
Период выполнения осенний / весенний семестр 2022 /2023 учебного года

КАЛЕНДАРНЫЙ РЕЙТИНГ-ПЛАН
выполнения выпускной квалификационной работы

Обучающийся:

Группа	ФИО
8К93	Кузнецов Илья Евгеньевич

Тема работы:

Разработка генеративно-состязательной нейросети для генерации КТ-изображений грудной клетки человека

Срок сдачи обучающимся выполненной работы:	

Дата контроля	Название раздела (модуля) / вид работы (исследования)	Максимальный балл раздела (модуля)
10.03.2023	Изучение различных архитектур генеративно-состязательной нейросети	20
20.05.2023	Реализация и обучение различных архитектур генеративно-состязательной нейросети	30
30.05.2023	Финансовый менеджмент, ресурсоэффективность, ресурсосбережение	25
22.05.2023	Социальная ответственность	25

СОСТАВИЛ:
Руководитель ВКР

Должность	ФИО	Ученая степень, звание	Подпись	Дата
Доцент ОИТ ИШИТР	Аксенов Сергей Владимирович	к.т.н.		

СОГЛАСОВАНО:
Руководитель ООП/ОПОП

Должность	ФИО	Ученая степень, звание	Подпись	Дата
Доцент ОИТ ИШИТР	Чердынцев Евгений Сергеевич	к.т.н.		

Обучающийся

Группа	ФИО	Подпись	Дата
8К93	Кузнецов Илья Евгеньевич		

РЕФЕРАТ

Выпускная квалификационная работа содержит 85 страниц, 30 рисунков, 17 таблиц, 11 источников, 1 приложение.

Ключевые слова: нейронная сеть, генеративно-состязательная нейросеть, GAN, DCGAN, ProGAN, генератор, дискриминатор, КТ-изображения грудной клетки, Pytnon, Pytorch, MONAI, графический пользовательский интерфейс, Tkinter.

Объект исследования: генеративно-состязательная нейросеть для генерации КТ-изображений грудной клетки.

Цель работы: разработать генеративно-состязательную нейросеть для генерации КТ-изображений грудной клетки.

Методы проведения работы: теоретические (изучение существующих архитектур состязательно генеративных нейросетей) и практические (реализация и обучение нейросетей DCGAN и ProGAN, разработка графического пользовательского интерфейса для взаимодействия с генератором изображений).

Область применения: наполнение датасетов для обучения нейронных сетей, работающих с КТ-изображениями грудной клетки.

В рамках развития проекта в будущем планируется:

– улучшения качества существующих моделей нейросетей, путем подбора оптимальных гипперпараметров обучения;

– изучения и реализация более продвинутых нейросетей для генерации изображений.

Содержание

Введение

Нейронная сеть — это алгоритм машинного обучения, который имитирует работу нервной системы человека. Она состоит из множества связанных между собой нейронов, которые обрабатывают информацию и передают ее дальше по сети. Она обучается на основе большого количества данных, которые помогают определить оптимальные веса связей между нейронами.

В настоящее время искусственные нейронные сети широко используются при решении самых разнообразных задач особенно там, где обычные алгоритмические решения оказываются неэффективными или вовсе невозможными. Одним из таких бурно развивающихся направлений применения нейросетей является генерация изображений с помощью архитектуры GAN [1].

Целью моей исследовательской работы являлось создание генеративно-состязательной нейросети для генераций изображений грудной клетки подобно тем, что получаются с помощью магнитно-резонансной томография.

Разработка генеративно-состязательной нейросети для генерации КТ-изображений грудной клетки является актуальной темой в области машинного обучения. В данный момент разрабатывается множеств нейронных сетей, задачей которых является классификация заболеваний или сегментация пораженных участков тела. Для обучения любой нейросети нужно много данных, и чем больше их, тем лучше будет качество натренированной модели.

КТ-изображения грудной клетки являются важным инструментом для диагностики заболеваний легких, сердца и других органов, расположенных в грудной клетке. Однако найти подобные изображения в свободном доступе довольно проблематично, поскольку данные о пациентах относятся к врачебной тайне. Самостоятельное же получение КТ-изображений требует значительных затрат времени и денег. Так стоимость одного сеанса на компьютерном томографе в Томске составляет 4000 рублей и в результат данного обследования получается в среднем лишь 150 изображений, а для

качественного обучения нейросети желательно иметь хотя бы 10000. Более того не каждый человек подойдет для снятия данных, необходимо еще найти пациентов с нужным типом патологии.

Разработка генеративно-состязательной нейросети может значительно упростить процесс получения КТ-изображений грудной клетки и снизить затраты на сбор данных. Полученные с помощью генератора изображения можно использовать для наполнения датасетов, используемых при обучении других нейросетей.

Для достижения поставленной цели были выполнены следующие задачи:

1. Изучение литературы, посвященной различным архитектурам GAN.
2. Создание датасета КТ-изображений грудной клетки для обучения нейронной сети
3. Реализация различных архитектур GAN для генерации КТ-изображений грудной клетки.
4. Обучение различных архитектур GAN на основе сформированного ранее датасета.
5. Оценка качества сгенерированных изображений.
6. Разработка графического пользовательского интерфейса для взаимодействия с нейронной сетью.

Определения, обозначения и сокращения

Нейросеть: Компьютерная модель, которая имитирует работу нервной системы человека.

Искусственный нейрон: Узел искусственной нейронной сети, являющийся упрощённой моделью естественного нейрона.

Сверточная нейросеть: Тип нейронной сети, который используется для обработки изображений и видео.

Сверточный слой: Слой искусственных нейронов, которые обнаруживают различные признаки в изображении.

Датасет: Набор данных для обучения и тестирования нейросети.

Батч: Подмножество датасета, которое используется для обработки нейросетью за один проход.

Эпоха: Один проход по всему датасету.

Графический пользовательский интерфейс (GUI): Способ взаимодействия пользователя с компьютерной программой, использующий графические элементы, такие как кнопки, поля ввода, меню и т.д.

ReLU (Rectified Linear Unit): Функция активации, которая используется в нейронных сетях. Она применяется к выходу каждого нейрона и возвращает значение 0, если входное значение отрицательное, и возвращает само входное значение, если оно положительное.

Обзор литературных источников

В статье Цаунит, А. Н. «Перспективы развития и применения нейронных сетей» [1] из журнала «Молодой ученый» рассказывается об основных областях применения и тенденциях развития нейронных сетей.

В блоге «Создаем GAN с помощью PyTorch» [2] объясняется основный принцип работы и процесс обучения генеративно-состязательной нейросети и приводится пример написания базовой архитектуры нейросети.

В научной статье «Unsupervised Representation Learning with Deep Convolutional Generative Adversarial Networks» [3] рассказывается о структуре и принципах разработки и обучения генеративно-состязательной нейросети глубокой свертки (DCGAN).

В блоге «ProGAN: Progressive Growing Generative Adversarial Networks» [4] рассказывается об основных особенностях архитектуры прогрессивной генеративно-состязательной нейросети (ProGAN).

В блоге «Conditional GAN (cGAN) in PyTorch and TensorFlow» [5] приводятся основные отличия условных генеративно-состязательной нейросетей от обычных.

1. Теоретическая часть

1.1. Генеративно-состязательная нейросеть (GAN)

Генеративно-состязательная нейросеть (Generative Adversarial Network, GAN) — это тип нейронных сетей, которые используются для генерации новых данных, которые похожи на обучающие данные. Она состоит из двух нейронных сетей: генератора и дискриминатора, настроенных на работу друг против друга [2].

Впервые идея GAN была предложена в 2014 году и заключалась она в том, чтобы использовать хорошо обученный классификатор, чтобы различать сгенерированное изображение и реальное изображение. При наличии такого классификатора, можно создать и обучить сеть-генератор, пока она не сможет производить изображения, которые могут полностью обмануть классификатор [2].

GAN является продуктом этой процедуры: она содержит генератор, который генерирует изображение на основе заданного набора данных, и дискриминатор (классификатор), чтобы различать, является ли изображение реальным или сгенерированным [2]. Абстрактная структура GAN изображена на рисунке 1.

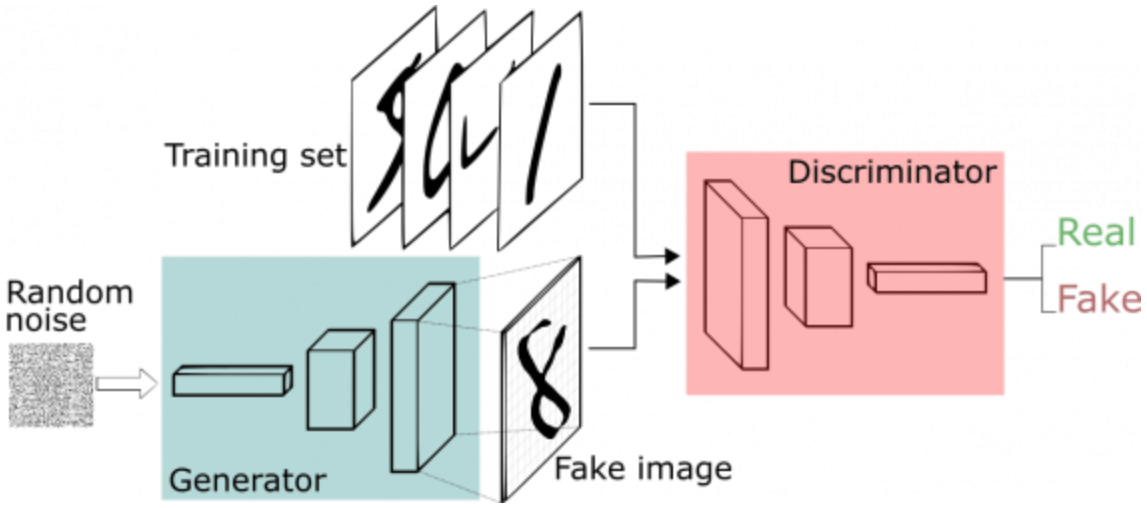

Рисунок 1 - Структура GAN

Самой простой версией GAN является архитектура, состоящая из полносвязных слоев. Однако данная нейросеть мало предназначена для

решения практических задач, поскольку у нее возникают трудности с генерацией даже простых изображений низкого разрешения.

1.2. Генеративно-состязательная нейросеть глубокой свертки (DCGAN)

1.2.1. Определение

DCGAN (Deep Convolutional Generative Adversarial Network)– это генеративно-состязательная нейросеть, в которой вместо полносвязных слоев применяются сверточные слои, в которых каждый фрагмент изображения умножается на матрицу (ядро) свёртки поэлементно, а результат суммируется и записывается в аналогичную позицию выходного изображения [3]. Работа свёрточной нейронной сети обычно интерпретируется как переход от конкретных особенностей изображения к более абстрактным деталям, и далее к ещё более абстрактным деталям вплоть до выделения понятий высокого уровня. При этом сеть самонастраивается и вырабатывает сама необходимую иерархию абстрактных признаков (последовательности карт признаков), фильтруя маловажные детали и выделяя существенное.

DCGAN была разработана для улучшения качества сгенерированных изображений и устранения проблем с размытыми и нечеткими изображениями, которые могут возникнуть при использовании обычных GAN.

1.2.2. Модель генератора и дискриминатора

Принцип работы генератора, структура которого изображена на рисунке 2, состоит в следующем: на вход нейросети подается 100-мерный вектор, который затем проходить через транспонированный сверточный слой, который увеличивает разрешение изображения, и слой batch-нормализации, который уменьшает величину, на которую смещаются значения узлов в скрытых слоях, что позволяет повысить производительность и стабилизировать работу искусственных нейронных сетей. После прохождения тензором подобных блоков несколько раз, на выходе генератора получается изображения с необходимыми нам числом каналов и разрешением [3].

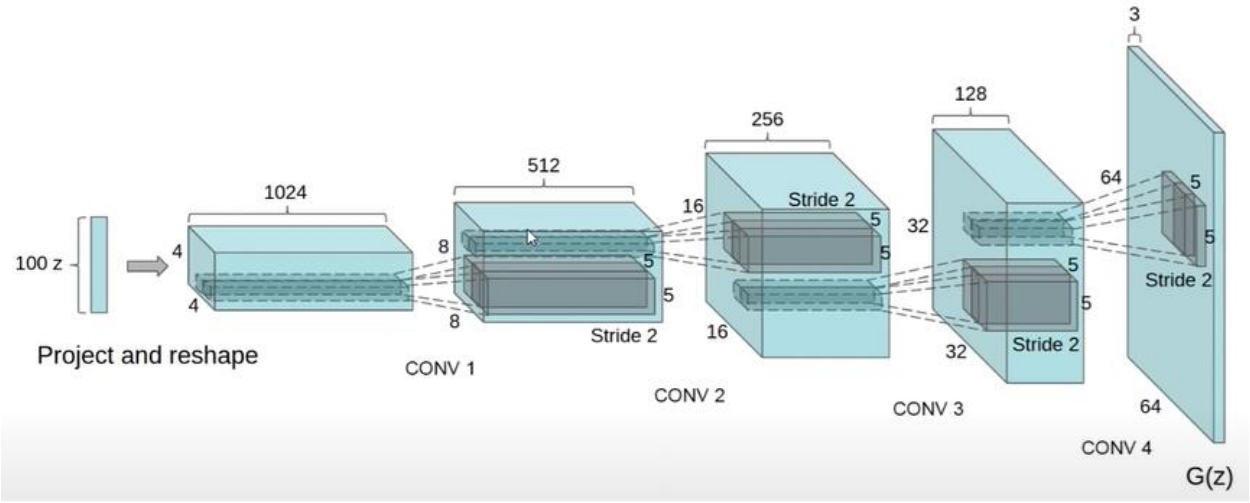

Рисунок 2 - Структура генератора DCGAN

Дискриминатор обладает обратной структурой, в которой сверточные слои понижают разрешение подаваемого на вход реального или сгенерированного изображения, пока на выходе не получается вектор, содержащий значения в пределах от 0 до 1, т.е. вероятность реальности пикселей изображения.

1.2.3. Метод обучения

Оптимизировать одновременно и генератор, и дискриминатор сложно, потому что две нейросети преследуют совершенно противоположные цели: генератор хочет создать что-то как можно более реалистичное, а дискриминатор хочет различать сгенерированные материалы.

При обучении DCGAN в качестве функции потерь используется бинарная кросс-энтропия. Пусть D(x) будет выходом дискриминатора, который представляет собой вероятность того, что x является реальным изображением, а G(z) будет выходом нашего генератора. Дискриминатор аналогичен бинарному классификатору, поэтому цель дискриминатора — максимизировать функцию потерь. С другой стороны, цель генератора минимизировать шансы дискриминатора сделать правильное определение, поэтому он старается минимизировать функцию потерь. Следовательно, окончательная функция потерь будет минимаксной игрой между двумя классификаторами, которую можно проиллюстрировать следующим образом:

$$\min_G \max_D L = \log\big(D(x)\big) + \log(1 - D(G(z)) \quad (1)$$

1.3. Прогрессивная генеративно-состязательная нейросеть (ProGAN)

1.3.1. Определение

ProGAN (Progressive Generative Adversarial Network) – это сверточная генеративно-состязательная нейросеть получившая свое название, благодаря своему основному ее принципу – прогрессивному или постепенному добавлению новых сверточных блоков и обучению генератора и критика на изображениях, увеличивающегося размера [4].

1.3.2. Основные особенности архитектуры

Задача сразу сгенерировать изображение большого разрешения для генератора может быть достаточно сложной, поэтому мы будем обучать его постепенно. Данный метод обучения называется прогрессивным ростом, и его принцип отражен на рисунке 3. Так для начала мы обучаем генератор создавать картинки 4x4 px, а критика распознавать их. Затем, когда генератор достаточно натренирован, мы учим его генерировать изображения 8x8 px и так далее вплоть до нужного разрешения, которое в моем случае составляло 256x256 px. При этом все слои остаются обучаемыми на время тренировки [4].

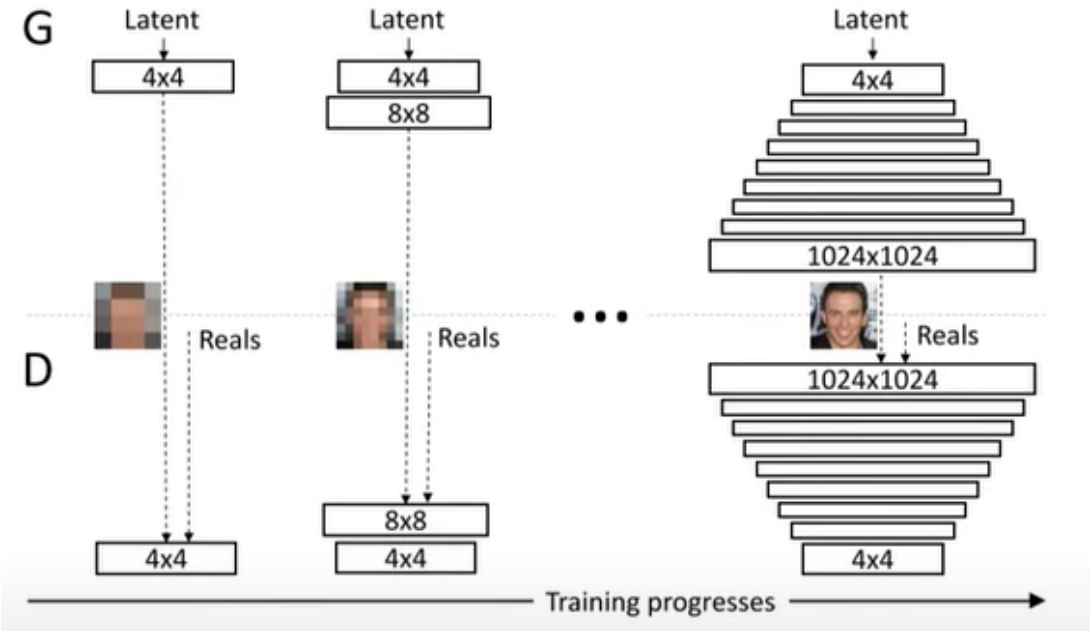

Рисунок 3 - Принцип прогрессивного роста

Одно из преимуществ такого метода это сокращенное время обучения. При прогрессивно растущих GAN большинство итераций выполняется с меньшим разрешением, и сопоставимое качество результата часто достигается в 2-6 раз быстрее, в зависимости от конечного разрешения [4].

При удвоении разрешения генератора и дискриминатора мы плавно встраиваем в структуру нейросетей новые сверточные слои. Так во время обучения нейросети при переходе от низкого разрешения к более высокому, мы постепенно замещаем слой, который просто удваивает число пикселей изображения, на тренирующийся сверточный слой.

Для выравнивания скорости обучения, мы вместо обычного сверточного слоя используем особый слой, в котором во время прямого прохода по нейросети значение весов узлов умножается на константу, вычисляемую следующим образом:

$$\text{scale} = \sqrt{\frac{2}{k*k*c}} \quad (2)$$

Где k*k – размер ядра свертки сверточного слоя

c – число каналов изображения

Чтобы исключить сценарий, в котором величины в генераторе и дискриминаторе выходят из-под контроля в результате конкуренции, в генераторе после каждого сверточного слоя мы нормализуем вектор признаков в каждом пикселе до общего масштаба. Для этого добавим слой PixelNorm, характеризующийся следующим выражением:

$$b_{x,y} = a_{x,y} / \sqrt{\frac{1}{N}\sum_{j=0}^{N-1}(a_{x,y}^{j})^2 + \epsilon} \quad (3)$$

Где $\epsilon = 10^{-8}$

N – Число каналов

$a_{x,y}$ и $b_{x,y}$ - исходный и нормализованный вектора признаков пикселя по координатам (x, y) соответственно

Еще одним большим отличием ProGAN от DCGAN является метод обучения моделей и вид функции потерь. Пусть Pg и Pr вероятностные распределения генерируемых и реальных изображений. Для генерации реалистичных изображений нужно чтобы они были как можно более близки. В таком случае в ходе обучения нейросети необходимо сдвигать Pg по направлению к Pr.

Новую функцию потерь можно представить следующим образом:

$$E_{x\sim Pr}[f(x)] - E_{x\sim Pg}[f(x)] \quad (4)$$

В левой части выражения x – это тензор, получаемый на выходе дискриминатора, который в ProGAN в официальной литературе называют критиком поскольку в конце его структуры отсутствует сигмойдная функций активации. В правой же части выражения x – тензор значений, который создает генератор. E означает нахождение среднего значения данных выходных сигналов.

В ходе обучения нейросети критик (еще одно название дискриминатора) стремится как можно больше разделить эти значения, чтобы суметь их различать, поэтому он стремится к максимизации данного выражения. В то же время задачей генератора наоборот является свести эти значения к одному

модулю, дабы обмануть критика, в связи с чем он стремится к минимизации данного выражения.

В этом принципе также заключается еще одно преимущество ProGAN перед предыдущей структурой, поскольку значение функции потерь позволяет понять успешность тренировки нейросетей. Чем это значение в конце тренировки ближе к 0, тем лучше генератор справляется со своей задачей.

1.4. Условная генеративно-состязательная нейросеть (CGan)

1.4.1. Определение

Обычная GAN обучается совершенно бесконтрольно, что означает, что в процессе обучения не задействованы никакие метки. Хотя генератор GAN может генерировать новые реалистичные образцы для определенного набора данных, мы не имеем никакого контроля над типом генерируемых изображений [5].

CGAN Conditional Generative Adversarial Network — это модификация GAN, которая позволяет управлять генерацией изображений путем добавления условий входных данных. В отличие от обычных GAN, где генератор создает изображения из случайного шума, в CGAN, структура которой изображена на рисунке 4 генератор получает дополнительную информацию в виде условий, которые определяют, какое изображение нужно сгенерировать [5].

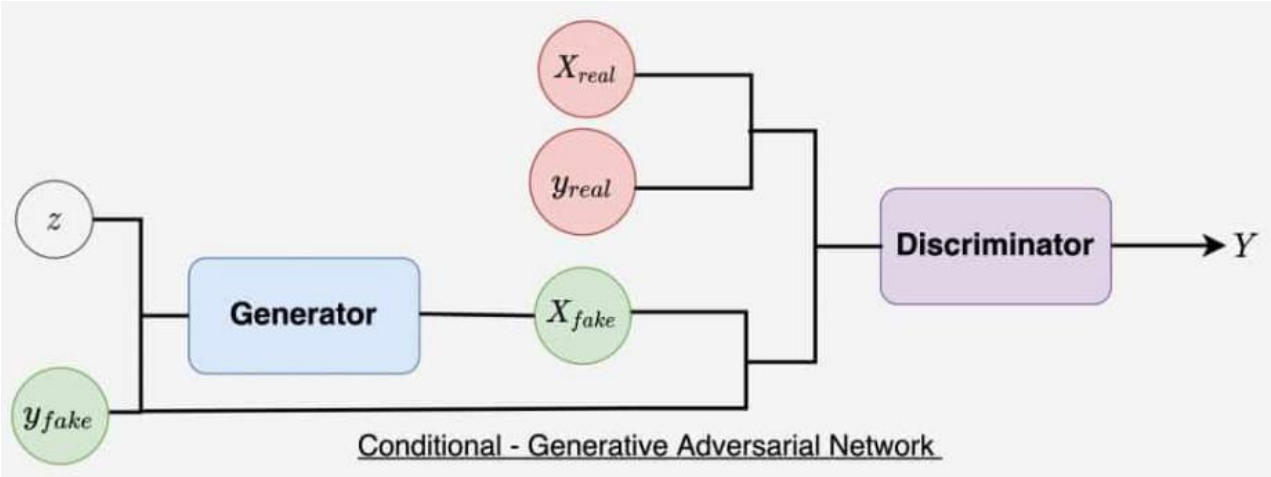

Рисунок 4 - Структура CGAN

1.4.2. Особенности моделей генератора и дискриминатора

Так при обучении CGAN:

- Генератор учится для создания реалистичных изображения для каждой метки в обучающем наборе данных
- Дискриминатор учится отличать поддельные изображения от настоящих, учитывая информацию о классе.
- Генератор и дискриминатор продолжают генерировать и классифицировать изображения точно так же, как и раньше, но с условной вспомогательной информацией.

Обычно генератору требуется лишь вектор шума для генерации изображений. Однако при условной генерации также требуется вспомогательная информация, которая сообщает генератору, выборку какого класса создавать [5].

Теперь генератору недостаточно создавать реалистично выглядящие картинки; не менее важно, чтобы сгенерированные примеры также соответствовали метке. Таким образом, условный генератор теперь несет гораздо большую ответственность за результат чем в обычной версии генеративно-состязательной нейросети [5].

Как только генератор будет полностью обучен ему можно указать изображения какого класса условный генератор должен создать, просто передав массив с нужными метками.

Дискриминатору передаются как реальные, так и поддельные примеры с метками. Он учится не только отличать реальные данные от поддельных, но выделять совпадающие пары. Пара считается совпадающей, когда изображению присвоена правильная метка. В конечном итоге дискриминатор выдает вероятность, указывающую на то, что входные данные являются реальными или поддельными.

Цель дискриминатора состоит в том, чтобы научиться:

- Одобрять все пары правильных меток с реальными изображениями
- Отклоните все пары поддельных изображений с метками
- Отклонять изображение с неподходящей меткой

2. Практическая часть

2.1. Выбор программных средств

Для предобработки данных, разработки и обучения различных архитектур GAN использовался Pytorch, библиотека машинного обучения с открытым исходным кодом, разработанная на языке программирования Python. На поздних этапах для предобработки реальных КТ-изображений использовалась библиотека MONAI, специально разработанная для обработки медицинских изображений.

В начале исследования разработка нейростей производилась в интерактивной среде Google Colaboratory (Colab). Данная среда позволяет заимствовать вычислительные мощности удаленных облачных серверов для обучения алгоритмов машинного обучения. Однако из-за серьезных ограничений на мощность и время использования GPU в бесплатной версии Colab, с которыми пришлось столкнуться при реализации архитектуры ProGAN, в дальнейшем разработка велась в локальном аналоге Colab Jupyter Notebook, запущенном в виртуальной среде, в которую останавливались необходимые библиотеки.

Для разработки графического пользовательского интерфейса для взаимодействия с обученным генератором, использовалась стандартная библиотека Python Tkinter, в которой имеется все необходимого для создания небольшого компьютерного приложения.

2.2. Разработка DCGAN

2.2.1. Обработка данных

Датасет с КТ-изображениями грудных клеток был получен от руководителя ВКР и содержал 6 подпапок, каждая из которых содержала изображения отдельной патологии дыхательной системы.

В имеющихся тренировочных данных были представлены следующие типы патологий:

Caseous Pneumonia (Казеозная пневмония) - тяжело протекающая самостоятельная клиническая форма туберкулеза легких, для которой характерно быстрое прогрессирование специфического воспаления, сопровождающееся разрушением лёгочной паренхимы и образованием каверн;

Disseminated Tuberculosis (Диссеминированный туберкулез легких) - клиническая форма туберкулезной инфекции, характеризующаяся формированием в легких многочисленных очагов специфического воспаления вследствие гематогенного или лимфогенного распространения микобактерий;

Fibroso Cavernous Tuberculosis (Кавернозный туберкулез легких) - деструктивная форма заболевания, отличительной чертой которой является наличие в легочной ткани изолированной полости распада (каверны);

Focal Tuberculosis (Очаговый туберкулез легких) - форма вторичного туберкулеза, протекающая с формированием в легких очагов специфического воспаления не более 10 мм в диаметре;

Infiltrative Tuberculosis (Инфильтративный туберкулез легких) - вторичная туберкулезная инфекция, характеризующаяся распространенным поражением легких с экссудативным типом воспалительной реакции и формированием очагов казеозного распада;

Tuberculoma (Туберкулома легкого) - осумкованный казеозный очаг в легочной ткани диаметром более 1 см, образующийся в исходе различных форм туберкулеза

Предоставленные изображения хранились в формате dcm (dycom), который используется для хранения медицинских изображений, таких как рентгеновские снимки, КТ и МРТ сканы. Помимо информации о пикселях в данном формате, также могут хранится данные о пациенте такие как пол, возраст, рост и т.д.

При работе над первыми версиями DCGAN и ProGAN, все изображения копировались в одна папку, а затем конвертировались в PNG-формат с помощью сторонней библиотеки.

Данное преобразование было необходимо, потому что Pytorch по умолчанию не может работать с картинками в формате dcm.

После изображений загружались с помощью класса Dataset, предварительно проходя ряд трансформаций, отраженных на рисунке 5 и включающих в себя преобразование изображения в серую шкалу, изменение разрешения изображения нормализацию значений пикселей, а также преобразование в тензор.

```
from torchvision.transforms.transforms import Resize
transforms = transforms.Compose(
    [transforms.Grayscale(num_output_channels=1),
     transforms.Resize((IMAGE_SIZE, IMAGE_SIZE)),
     transforms.ToTensor(),
     transforms.Normalize([0.5 for _ in range(CHANNELS_IMG)], [0.5 for _ in range(CHANNELS_IMG)])]
)
```

Рисунок 5 - Список трансформаций при обучении DCGAN

2.2.2. Модели генератора и дискриминатора

Модели генератора и дискриминатора, приведенные на рисунках 6 и 7 соответственно, были реализованы на основе описанных в теоретической части принципах. Модели были построены из блоков, каждый из которых содержит сверточный слой, слой побатчевой нормализации и функцию активации ReLU.

```
class Generator(nn.Module):
    def __init__(self, z_dim, channels_img, features_g):
        super(Generator, self).__init__()
        #Input: N*z_dim*1*1
        self.gen = nn.Sequential(
            self.block(z_dim, features_g*32, 4, 1, 0),#4x4
            self.block(features_g*32, features_g*16, 4, 2, 1),#8x8
            self.block(features_g*16, features_g*8, 4, 2, 1),#16x16
            self.block(features_g*8, features_g*4, 4, 2, 1),#32x32
            self.block(features_g*4, features_g*2, 4, 2, 1),#64x64
            nn.ConvTranspose2d(features_g*2, channels_img, kernel_size=4, stride=2, padding=1),#128x128
            nn.Tanh()#[-1, 1]

        )

    def block(self, in_channels, out_channels, kernel_size, stride, padding):
        return nn.Sequential(
            nn.ConvTranspose2d(in_channels, out_channels, kernel_size, stride, padding, bias = False),
            nn.BatchNorm2d(out_channels),
            nn.ReLU()
        )

    def forward(self, x):
        return self.gen(x)
```

Рисунок 6 - Модель генератора DCGAN

```
class Discriminator(nn.Module):
    def __init__(self, channels_img, features_d):
        super(Discriminator, self).__init__()
        # Input: N*channels_img*128*128
        self.disc = nn.Sequential(
            nn.Conv2d(channels_img, features_d, kernel_size=4, stride=2, padding=1),#64x64
            nn.LeakyReLU(0.2),
            self.block(features_d, features_d*2, 4, 2, 1),#32x32
            self.block(features_d*2, features_d*4, 4, 2, 1),#16x16
            self.block(features_d*4, features_d*8, 4, 2, 1),#8x8
            self.block(features_d*8, features_d*16, 4, 2, 1),#4x4
            nn.Conv2d(features_d*16, 1, kernel_size=4, stride=2, padding=0),#1x1
            nn.Sigmoid()#[0, 1]
        )

    def block(self, in_channels, out_channels, kernel_size, stride, padding):
      return nn.Sequential(
          nn.Conv2d(in_channels, out_channels, kernel_size, stride, padding, bias = False),
          nn.BatchNorm2d(out_channels),
          nn.LeakyReLU(0.2)
      )

    def forward(self, x):
        return self.disc(x)
```

Рисунок 7 - Модель дискриминатора DCGAN

2.2.3. Процесс обучения DCGAN

Обучение нейросети происходит в уставленной разработчиком число эпох. Каждую эпоху мы с помощью батчей (небольших наборов данных) тренируем одновременно и дискриминатор, и генератор.

Сначала вычисляется значения функции потерь, бинарной-кросс энтропии, дискриминатора, на основе выходов нейросети после подачи реальных изображений, а затем находятся значения функции потерь, после прохождения через сеть сгенерированных изображений. Итоговая ошибка рассчитывается как среднее 2 из двух. Далее после вычисления градиента от функции потерь, с помощью оптимизатора корректируются веса сети дискриминатора.

Для тренировки Генератора, вновь находятся ошибки на выходе дискриминатора, после прохождения через сеть фальшивых изображений. Затем также с помощью оптимизатора подстраиваются веса генератора.

В начале каждой эпохи обучения происходит отображения ошибок по эпохам и сохранения фальшивых изображений в логи, которые можно посмотреть с помощью инструмента визуализации и отладки TensorBoard.

Листинг процесса обучения нейросети DCGAN приведен на рисунке 8.

```
for epoch in tqdm(range(NUM_EPOCHS)):
    for batch_idx, (real, _) in enumerate(loader):
        real = real.to(device)
        noise = torch.randn((BATCH_SIZE, Z_DIM, 1, 1)).to(device)
        ### Тренировка деёскриминатора: max log(D(x)) + log(1 - D(G(z)))
        disc_real = disc(real).view(-1)
        lossD_real = loss(disc_real, torch.ones_like(disc_real))
        fake = gen(noise)
        disc_fake = disc(fake).view(-1)
        lossD_fake = loss(disc_fake, torch.zeros_like(disc_fake))
        lossD = (lossD_real + lossD_fake) / 2
        disc.zero_grad()
        lossD.backward(retain_graph=True)
        opt_disc.step()

        ### Тренировка генератора: min log(1 - D(G(z))) <-> max log(D(G(z))
        output = disc(fake).view(-1)
        lossG = loss(output, torch.ones_like(output))
        gen.zero_grad()
        lossG.backward()
        opt_gen.step()

        if batch_idx == 0:
            print(
                f"\tEpoch [{epoch}/{NUM_EPOCHS}] Loss D: {lossD:.4f}, loss G: {lossG:.4f}"
            )

            with torch.no_grad():
                fake = gen(fixed_noise)
                img_grid_real = torchvision.utils.make_grid(real[:16], normalize=True)
                img_grid_fake = torchvision.utils.make_grid(fake[:16], normalize=True)

                writer_real.add_image(
                    "Реальные изображения КТ грудной клетки", img_grid_real, global_step=step
                )
                writer_fake.add_image(
                    "Сгенерированные изображения КТ грудной клетки", img_grid_fake, global_step=step
                )
                step += 1
```

Рисунок 8 - Процесс обучения нейросети DCGAN

2.2.4. Анализ полученных результатов

Обучение нейросети проходило в течении 30 эпох и длилось примерно 1.5 часа, за это время генератор научился создавать изображения разрешением 128x128 пикселей, пример которых представлен на рисунке 9.

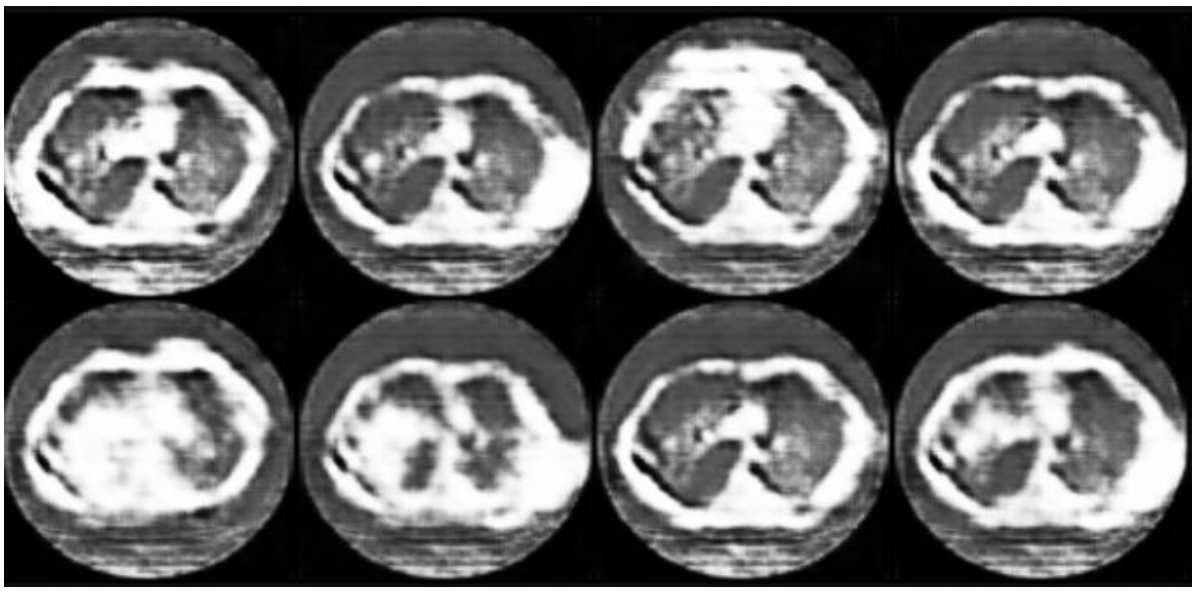

Рисунок 9 - Изображения, создаваемые генератором DCGAN

Как видно из рисунка выше изображения генерируемые DCGAN хоть и повторяют силуэт грудной клетки, но при этом не отображают мелкие детали и не имеют хорошей резкости. Для улучшения качества изображений была необходима более сложна архитектура нейросети, которой стала ProGAN.

2.3. Разработка ProGAN

2.3.1. Модели генератора и критика

Перед написанием моделей генератора и критика, сначала были реализованы новые слои, такие как взвешенный сверточный слой (WSCconv2d) и слой попиксельной нормализации (PixelNorm), принцип работы которых был описан в разделе 1.3.2. Реализация новых слоев приведена на рисунке 10.

Также был написан отдельный класс для блока свертки, листинг которого отображен на рисунке 11, включающий в себя два последовательных сверточных слоя WSCconv2d, функцию активации RelU и слой PixelNorm.

```python
class WSConv2d(nn.Module):

    def __init__(
        self, in_channels, out_channels, kernel_size=3, stride=1, padding=1, gain=2
    ):
        super(WSConv2d, self).__init__()
        self.conv = nn.Conv2d(in_channels, out_channels, kernel_size, stride, padding)
        self.scale = (gain / (in_channels * (kernel_size ** 2))) ** 0.5
        self.bias = self.conv.bias
        self.conv.bias = None

        # initialize conv layer
        nn.init.normal_(self.conv.weight)
        nn.init.zeros_(self.bias)

    def forward(self, x):
        return self.conv(x * self.scale) + self.bias.view(1, self.bias.shape[0], 1, 1)

class PixelNorm(nn.Module):
    def __init__(self):
        super(PixelNorm, self).__init__()
        self.epsilon = 1e-8

    def forward(self, x):
        return x / torch.sqrt(torch.mean(x ** 2, dim=1, keepdim=True) + self.epsilon)
```

Рисунок 10 - Реализация слоев WSConv2d и PixelNorm

```python
class ConvBlock(nn.Module):
    def __init__(self, in_channels, out_channels, use_pixelnorm=True):
        super(ConvBlock, self).__init__()
        self.use_pn = use_pixelnorm
        self.conv1 = WSConv2d(in_channels, out_channels)
        self.conv2 = WSConv2d(out_channels, out_channels)
        self.leaky = nn.LeakyReLU(0.2)
        self.pn = PixelNorm()

    def forward(self, x):
        x = self.leaky(self.conv1(x))
        x = self.pn(x) if self.use_pn else x
        x = self.leaky(self.conv2(x))
        x = self.pn(x) if self.use_pn else x
        return x
```

Рисунок 11 - Реализация блока свертки

Модели генератора и критика, показанные на рисунках 12 и 13 соответственно, претерпели серьезное изменение по сравнению с моделями в архитектуре DCGAN, поскольку в них были реализованы, описанные ранее принципы прогрессивного роста разрешения изображения и постепенного встраивания слоев.

```python
class Generator(nn.Module):
    def __init__(self, z_dim, in_channels, img_channels=3):
        super(Generator, self).__init__()
        # начальное преобразование 1x1 -> 4x4
        self.initial = nn.Sequential(
            PixelNorm(),
            nn.ConvTranspose2d(z_dim, in_channels, 4, 1, 0),
            nn.LeakyReLU(0.2),
            WSConv2d(in_channels, in_channels, kernel_size=3, stride=1, padding=1),
            nn.LeakyReLU(0.2),
            PixelNorm(),
        )
        self.initial_channels_change = WSConv2d(
            in_channels, img_channels, kernel_size=1, stride=1, padding=0
        )
        self.prog_blocks, self.channels_change_layers = (
            nn.ModuleList([]),
            nn.ModuleList([self.initial_channels_change]),
        )

        for i in range(len(factors) - 1):  # -1 чтобы предотвратить ошибку выхода индекса за пределы массива из-за factors[i+1]
            conv_in_c = int(in_channels * factors[i])
            conv_out_c = int(in_channels * factors[i + 1])
            self.prog_blocks.append(ConvBlock(conv_in_c, conv_out_c))
            self.channels_change_layers.append(
                WSConv2d(conv_out_c, img_channels, kernel_size=1, stride=1, padding=0)
            )
    def fade_in(self, alpha, upscaled, generated):
        # alpha должно быть скалярным в пределах [0, 1], и upscale.shape == generated.shape
        return torch.tanh(alpha * generated + (1 - alpha) * upscaled)

    def forward(self, x, alpha, steps):
        out = self.initial(x)

        if steps == 0:
            return self.initial_channels_change(out)

        for step in range(steps):
            upscaled = F.interpolate(out, scale_factor=2, mode="nearest")
            out = self.prog_blocks[step](upscaled)

        # Число каналов upscale Остаеться постоянным, в то время как при прохождение через prog_blocks оно может измениться.
        # Поэтому мы пропускаем их через разные channels_change_layers
        final_upscaled = self.channels_change_layers[steps - 1](upscaled)
        final_out = self.channels_change_layers[steps](out)
        return self.fade_in(alpha, final_upscaled, final_out)
```

Рисунок 12 - Модель генератора ProGAN

```
class Critic(nn.Module):
    def __init__(self, z_dim, in_channels, img_channels=3):
        super(Critic, self).__init__()
        self.prog_blocks, self.channels_change_layers = nn.ModuleList([]), nn.ModuleList([])
        self.leaky = nn.LeakyReLU(0.2)

        # Мы идем в обратном направлении, потому что критик должен зеркально отражать генератор
        for i in range(len(factors) - 1, 0, -1):
            conv_in = int(in_channels * factors[i])
            conv_out = int(in_channels * factors[i - 1])
            self.prog_blocks.append(ConvBlock(conv_in, conv_out, use_pixelnorm=False))
            self.channels_change_layers.append(
                WSConv2d(img_channels, conv_in, kernel_size=1, stride=1, padding=0)
            )


        self.initial_channels_change = WSConv2d(
            img_channels, in_channels, kernel_size=1, stride=1, padding=0
        )
        self.channels_change_layers.append(self.initial_channels_change)
        self.avg_pool = nn.AvgPool2d(kernel_size=2, stride=2)  # avg pool используется для уменьшения изображения вдвое

        # блок для изображения 4x4
        self.final_block = nn.Sequential(
            # +1 к in_channels поскольку мы брибавляем слой MiniBatch std
            WSConv2d(in_channels + 1, in_channels, kernel_size=3, padding=1),
            nn.LeakyReLU(0.2),
            WSConv2d(in_channels, in_channels, kernel_size=4, padding=0, stride=1),
            nn.LeakyReLU(0.2),
            WSConv2d(in_channels, 1, kernel_size=1, padding=0, stride=1)
        )

    def fade_in(self, alpha, downscaled, out):
        return alpha * out + (1 - alpha) * downscaled

    def minibatch_std(self, x):
        batch_statistics = (
            torch.std(x, dim=0).mean().repeat(x.shape[0], 1, x.shape[2], x.shape[3])
        )

        return torch.cat([x, batch_statistics], dim=1)

    def forward(self, x, alpha, steps):

        cur_step = len(self.prog_blocks) - steps


        out = self.leaky(self.channels_change_layers[cur_step](x))

        if steps == 0:  #image is 4x4
            out = self.minibatch_std(out)
            return self.final_block(out).view(out.shape[0], -1)


        downscaled = self.leaky(self.channels_change_layers[cur_step + 1](self.avg_pool(x)))
        out = self.avg_pool(self.prog_blocks[cur_step](out))

        # в отличии от генератора fade_in выполняется сначала между уменьшенным масштабом и входом
        out = self.fade_in(alpha, downscaled, out)

        for step in range(cur_step + 1, len(self.prog_blocks)):
            out = self.prog_blocks[step](out)
            out = self.avg_pool(out)

        out = self.minibatch_std(out)
        return self.final_block(out).view(out.shape[0], -1)
```

Рисунок 13 - Модель критика ProGAN

2.3.2. Процесс обучения ProGAN

Процесс обучения нейросети ProGAN сильно отличается, от метода обучения DCGAN. Во-первых, поскольку генератор и критик обучаются постепенно, та на каждое разрешение нейросеть тренируется определенное разработчиком число эпох. Во-вторых, как было сказано меняются принцип вычисления потерь. Вместо бинарной кросс-энтропии, для потерь критика мы вычисляем разность средних значений выходных сигналов критика после пропуска через него реального и фальшивого изображений плюс градиентный штраф.

Градиентный штраф используется для коррекции потерь критика. Для его реализации была написана отдельная функция, листинг которой приведен на рисунке 14, в которой находится градиент для значений функции потерь критика, через которого пропустили интерполированное случайным образам реальное и сгенерированное изображение. Сам градиентный штраф равен среднему значению квадрата разности нормы градиента и 1.

```
def gradient_penalty(critic, real, fake, alpha, train_step, device="cuda:0"):
    BATCH_SIZE, C, H, W = real.shape
    beta = torch.rand((BATCH_SIZE, 1, 1, 1)).repeat(1, C, H, W).to(device)
    interpolated_images = real * beta + fake.detach() * (1 - beta)
    interpolated_images.requires_grad_(True)

    mixed_scores = critic(interpolated_images, alpha, train_step)

    gradient = torch.autograd.grad(
        inputs=interpolated_images,
        outputs=mixed_scores,
        grad_outputs=torch.ones_like(mixed_scores),
        create_graph=True,
        retain_graph=True,
    )[0]
    gradient = gradient.view(gradient.shape[0], -1)
    gradient_norm = gradient.norm(2, dim=1)
    gradient_penalty = torch.mean((gradient_norm - 1) ** 2)
    return gradient_penalty
```

Рисунок 14 - Функция для вычисления градиентного штрафа

Еще одна особенность обучения нейросети заключается в том, что на каждый шаг обновления весов генератора, приходится несколько шагов

обновления параметров критика, что усложняет задачу генератора, в результате чего заставляя его создавать более правдоподобные изображения.

На рисунке 15 представлена функция для обучения нейросети в рамках одного разрешения.

```
def train_fn(critic, gen,loader,dataset,step, alpha,opt_critic,opt_gen, tensorboard_step, writer,scaler_gen,scaler_critic,):
    loop = tqdm(loader, leave=True)
    for batch_idx, (real, _) in enumerate(loop):
        real = real.to(DEVICE)
        cur_batch_size = real.shape[0]

        # Train Critic: max E[critic(real)] - E[critic(fake)] <-> min -E[critic(real)] + E[critic(fake)]
        # which is equivalent to minimizing the negative of the expression
        noise = torch.randn(cur_batch_size, Z_DIM, 1, 1).to(DEVICE)

        with torch.cuda.amp.autocast():
            fake = gen(noise, alpha, step)
            critic_real = critic(real, alpha, step)
            critic_fake = critic(fake.detach(), alpha, step)
            gp = gradient_penalty(critic, real, fake, alpha, step, device=DEVICE)
            loss_critic = (
                -(torch.mean(critic_real) - torch.mean(critic_fake))
                + LAMBDA_GP * gp
                + (0.001 * torch.mean(critic_real ** 2))
            )

        opt_critic.zero_grad()
        scaler_critic.scale(loss_critic).backward()
        scaler_critic.step(opt_critic)
        scaler_critic.update()

        # Train Generator: max E[critic(gen_fake)] <-> min -E[critic(gen_fake)]
        with torch.cuda.amp.autocast():
            gen_fake = critic(fake, alpha, step)
            loss_gen = -torch.mean(gen_fake)

        opt_gen.zero_grad()
        scaler_gen.scale(loss_gen).backward()
        scaler_gen.step(opt_gen)
        scaler_gen.update()

        # Update alpha and ensure less than 1
        alpha += cur_batch_size / (
            (PROGRESSIVE_EPOCHS[step] * 0.5) * len(dataset)
        )
        alpha = min(alpha, 1)

        if batch_idx == 0:
          with torch.no_grad():
              fixed_fakes = gen(FIXED_NOISE, alpha, step) * 0.5 + 0.5
          plot_to_tensorboard(
              writer,
              loss_critic.item(),
              loss_gen.item(),
              real.detach(),
              fixed_fakes.detach(),
              tensorboard_step,
          )
          tensorboard_step += 1

        loop.set_postfix(
          gp=gp.item(),
          loss_critic=loss_critic.item(),
        )

    return tensorboard_step, alpha
```

Рисунок 15 - Функция обучения нейросети ProGAN

Функция train_fn представленная выше запускается несколько раз, в зависимости от того какое конечное разрешение генерируемых изображений нужно достичь.

2.3.3. Анализ полученных результатов

Нейросеть ProGAN обучалась в течении 9 часов, по 30 эпох на каждое разрешение. Результат обучения можно оценить по рисункам ниже. Так на рисунке 16 изображен график значений функции потерь в конце каждой эпохи, а на рисунке 17 пример создаваемых генератором изображений.

Рисунок 16 - График значений функции потерь нейросети ProGAN

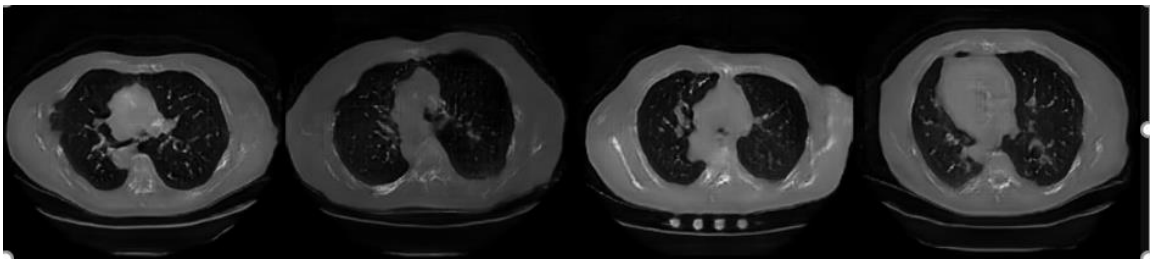

Рисунок 17 - Изображения, создаваемые генератором ProGAN

Как видно из рисунка 17 изменение архитектуры нейросети привело к значительному улучшению качества генерируемых изображений по

сравнению с DCGAN. Однако на получаемых таким образом изображениях все еще трудно различать различные ткани, поскольку изображения получаются однотонными. На это есть несколько причин. Как описывалось рпнее загрузчик данных библиотеки Pytorch, с помощью которой были реализованы как сами модели генератора и критика, так построен процесс обучения, не может напрямую работать с медицинским форматом изображений (dcm). Поэтому для обучения описанных ранее нейросетей приходилось предварительно с помощью сторонней библиотеки конвертировать изображения из обучающей в выборке в формат PNG. Такое преобразование могло приводить к потере качества. Во-вторых, в оригинальных трансформациях библиотеки Pytorch отсутствует возможность точной настройки контраста изображений, что как раз и приводит к неразличимости мелких деталей у изображений из обучающей выборке.

2.3.4. Использование MONAI для обработки изображений

Для решения описанных выше проблем была использованы библиотека MONAI. Данная библиотека была использована на этапе загрузки и предварительной трансформации изображений. Самой главной трансформацией являлось изменение контраста изображения, которое позволило сделать мелкие детали на изображении более различимыми. Разницу в предобработке изображений без использования библиотеки MONAI и с ее помощью можно понять по рисункам 18 и 19.

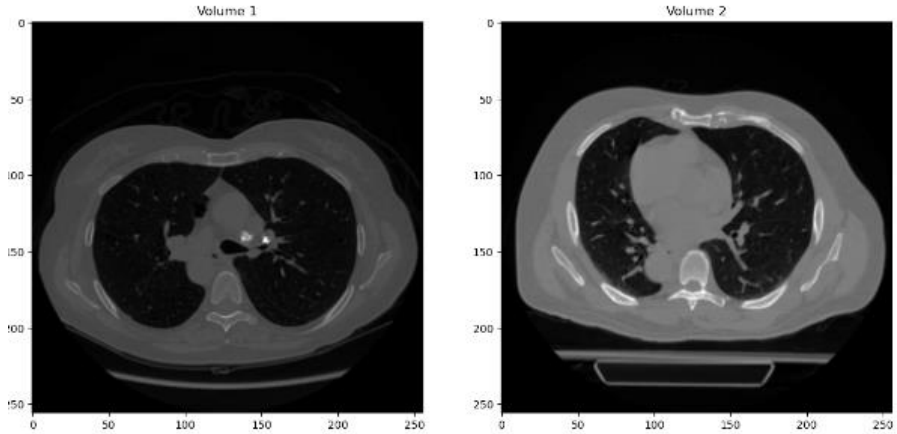

Рисунок 18 - Изображения, предобработанные без MONAI

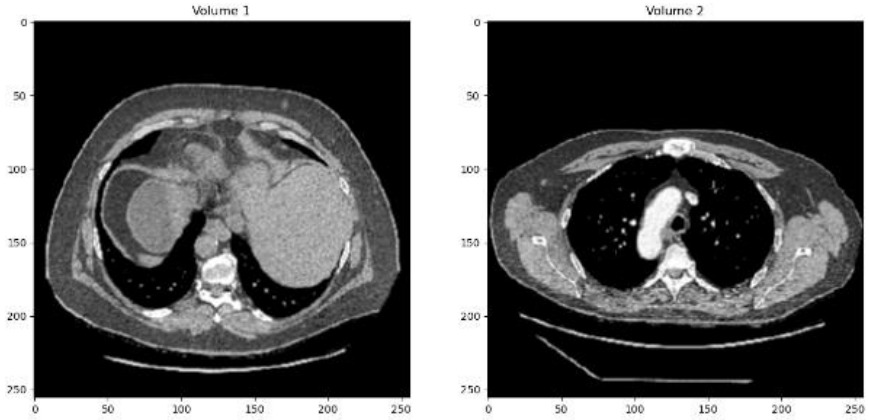

Рисунок 19 - Изображения, предобработанные с помощью MONAI

Структура моделей генератора и критика ProGAN остались прежними. Был изменен алгоритм обучения, а точнее добавлено приведение тензоров, полученных после трансформации к нужному формату.

Поскольку строение моделей генератора и критика остались неизменными, то сам процесс обучения нейросети так же занял 9 часов. Результаты обучения представлены на рисунках 20 и 21.

Рисунок 20 - График значений функции потерь нейросети ProGAN с MONAI

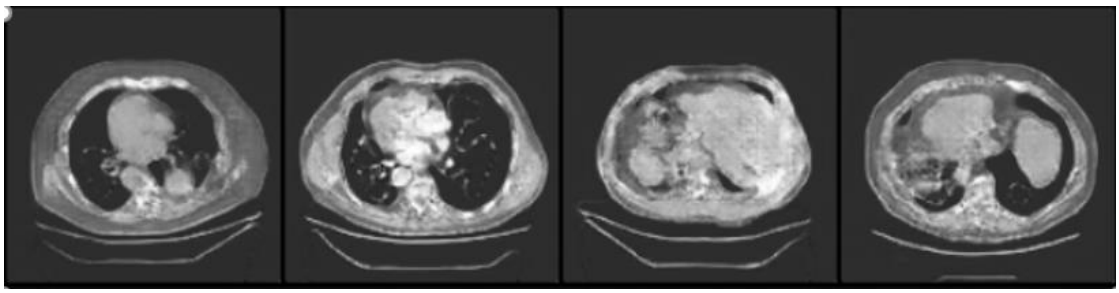

Рисунок 21 - Изображения, создаваемые генератором ProGAN с MONAI

При сравнении графиков потерь на рисунках 16 и 20 явно видно, что значения функции потерь нейросети ProGAN без использования MONAI больше, чем после применения библиотеки для трансформации изображений, а как было написано раннее, чем ближе значения графика к 0, тем лучше критик распознает изображения, а значит и генератор создает более реалистичные изображения. Также, по субъективной оценки изображения, создаваемые второй версией ProGAN получаются более детализированными по сравнению с прошлой версией.

2.4. Разработка CGAN

2.4.1. Модификации моделей и процесса обучения DCGAN

Поскольку в используемой для трансформации изображений библиотеке MONAI отсутствует класс датасета, автоматически определяющий числовую метку файла, по тому в какой подпапке он находится, мною был написан собственный класс датасета, показанный на рисунке рисунке 22. Метод getitem класса CTDataset возвращает кортеж из тензора и метки класса.

```python
class CTDataset(Dataset):
    """Кастомный класс для создания датасета из КТ-изображений грудной клетки и меток указывающих на тип патологии"""

    def __init__(self, image_files: dict, labels: list, transforms: transforms.Compose):
        self.image_files = image_files
        self.labels = labels
        self.transforms = transforms

    def __len__(self) -> int:
        return len(self.image_files)

    def __getitem__(self, index: int):
        img = self.image_files[index]
        img = self.transforms(img)
        return img["vol"], self.labels[index]
```

Рисунок 22 - Листинг класса CTDataset

Также была реализована функция, возвращающая список путей к тренировочным изображениям и список соответствующих классу изображений числовых меток.

С помощью описанных выше методов, типам патологий, перечисленным в теоретической части, были присвоены числовые метки от 0 до 5.

Сначала было решено модифицировать архитектуру DCGAN, дабы проверить возможность преобразования имеющийся архитектуры в условную генеративно-состязательную нейросеть.

В модель дискриминатора было внесено изменение, связанное с добавлением к тензору с данными о пикселях изображения дополнительного канала с метками, созданного с помощью слоя эмбеддинга.

Модификация генератора было также связано с добавлением слоя эмбеддинга, с помощью которого на этот раз закодированная информация о метке класса добавлялась к вектору случайных шумов.

Процесс обучения нейросети также был обновлен. Теперь из каждого батча из загрузчика берутся не только реальные изображения в виде тензоров, но и соответствующие им метки классов, которые далее передаются на вход дискриминатору и генератору.

2.4.2. Результаты обучения Conditional DCGAN

По итогу обучения условной DCGAN, генератор научился создавать КТ-изображения грудной клетки разрешением 256x256 пикселей с определенным типом патологии. Пример генерируемых изображений показан на рисунке 23

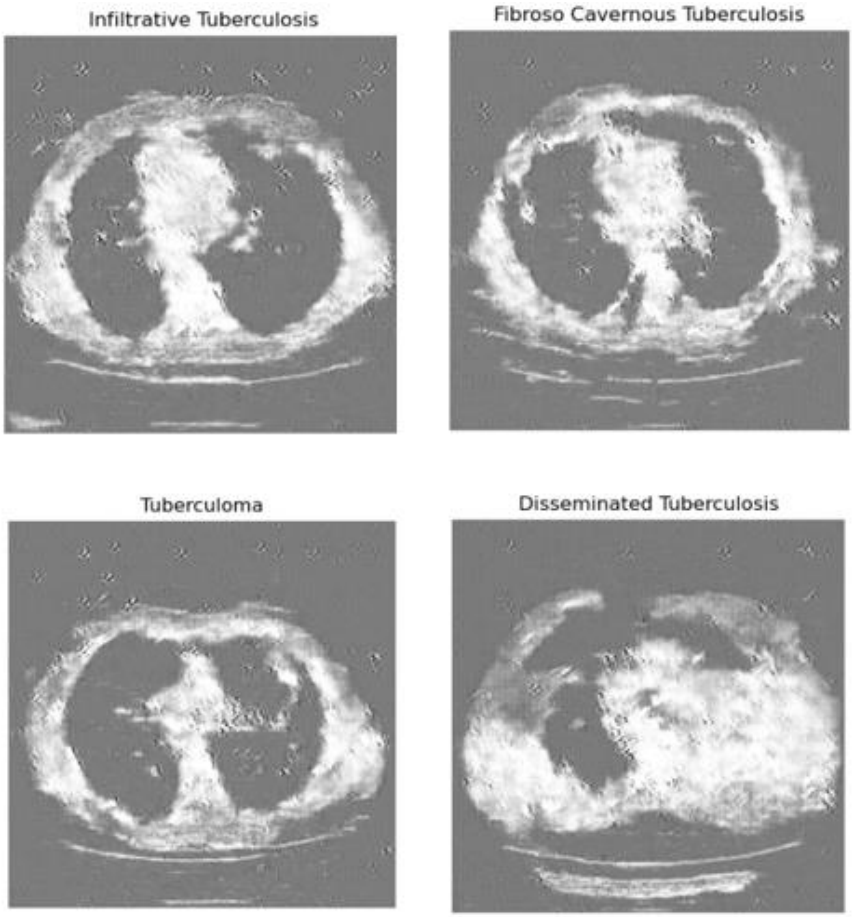

Рисунок 23 - Изображения, создаваемые генератором CDCGAN

Получаемые изображения имеют схожее с результатами генерации оригинальной архитектурой качество, которое является довольно низким. К тому же нейросеть легко переобучается. Так после тренировки в течении 30 эпох, генератор создает одни и те же изображения по одинаковым меткам, что является серьезным недостатком.

2.4.3. Модификации моделей и процесса обучения ProGAN

Модели генератора и критика также претерпели небольшие изменения при переходе нейросети к условной архитектуре.

Модификация генератора была аналогична той, что была использована при обновлении генератора DCGAN. Закодированные метки классов добавлялись новым каналом к вектору шумов.

Модификация же критика была более значительна. Поскольку в ходе обучение при увеличении разрешения изображений, подаваемых на вход критика, начальный сверточный слой постоянно заменялся, то новая матрица эмбеддинга, вычислялась при каждом новом переходе к более высокому размеру изображений.

Процесс обучения также претерпел изменения схожие с изменениями метода обучения Conditional DCGAN.

2.4.4. Результаты обучения Conditional ProGAN

CProGAN обучалась в течении 30 эпох на каждое разрешение. Время обучения составило почти 11 часов, что больше, чем у оригинальной архитектуры из-за того, что для каждого батча тратится время на кодирование меток классов. Особенно сильно это повлияло на время тренировки при низких разрешениях. По итогу обучения нейросеть научилась генерировать КТ-изображения грудной клетки, разрешением 256x256 пикселей с определенным типом патологии. Пример создаваемых изображений приведен на рисунке 24.

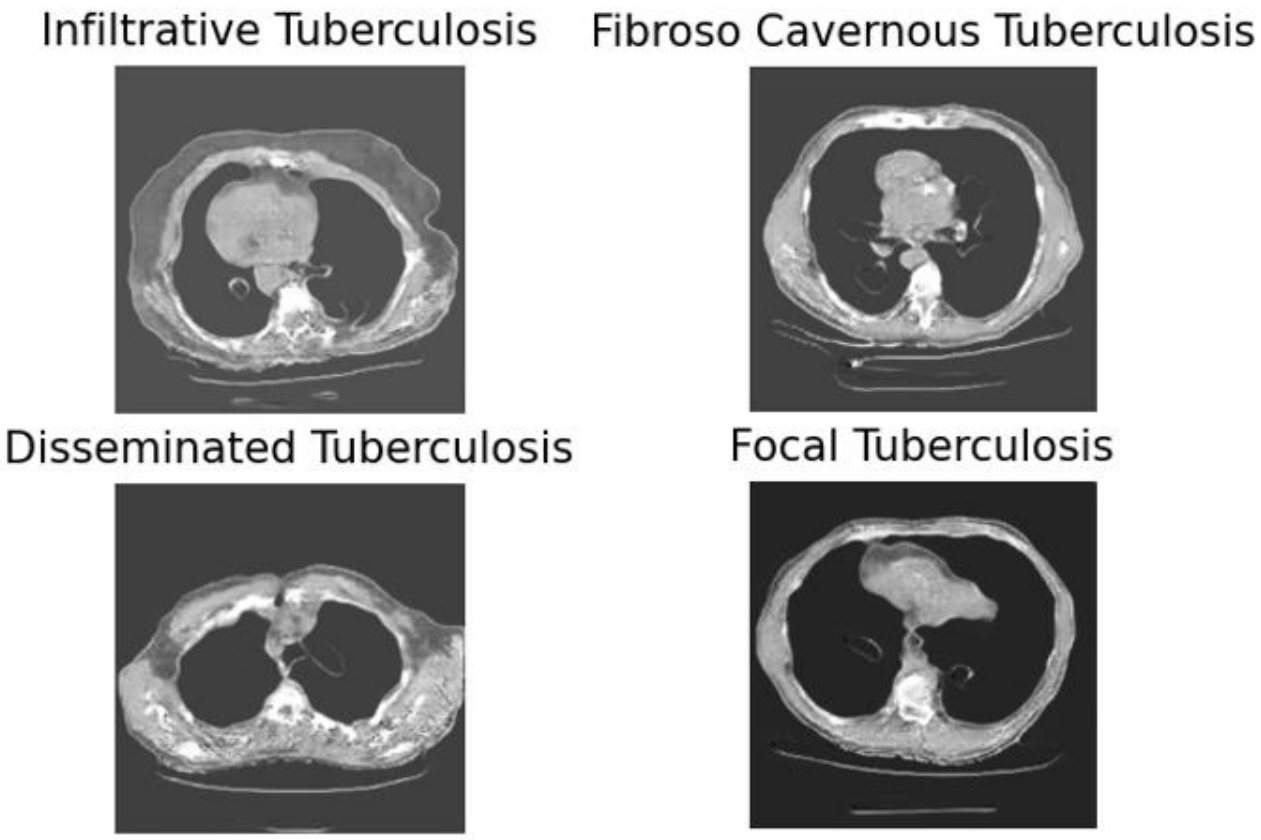

Рисунок 24 - Изображения, создаваемые генератором CProGAN

По сравнению с изображениями, генерируемыми CDGAN, изображения, получаемые с помощью генератора CProGAN имеют значительно лучшее качество. К тому же CProGAN не переобучается.

Именно генератор CProGAN вошел в основу скрипта для генерации изображений с помощью графического пользовательского интерфейса.

2.5. Разработка графического пользовательского интерфейса

Разработанный с помощью библиотеки Tkinter пользовательский интерфейс для взаимодействия с генератором CProGAN, представляет из себя окно, продемонстрированное на рисунке 25, содержащие поля для выбора параметров генерации и сохранения изображений, а также кнопку для запуска скрипта по получению изображений.

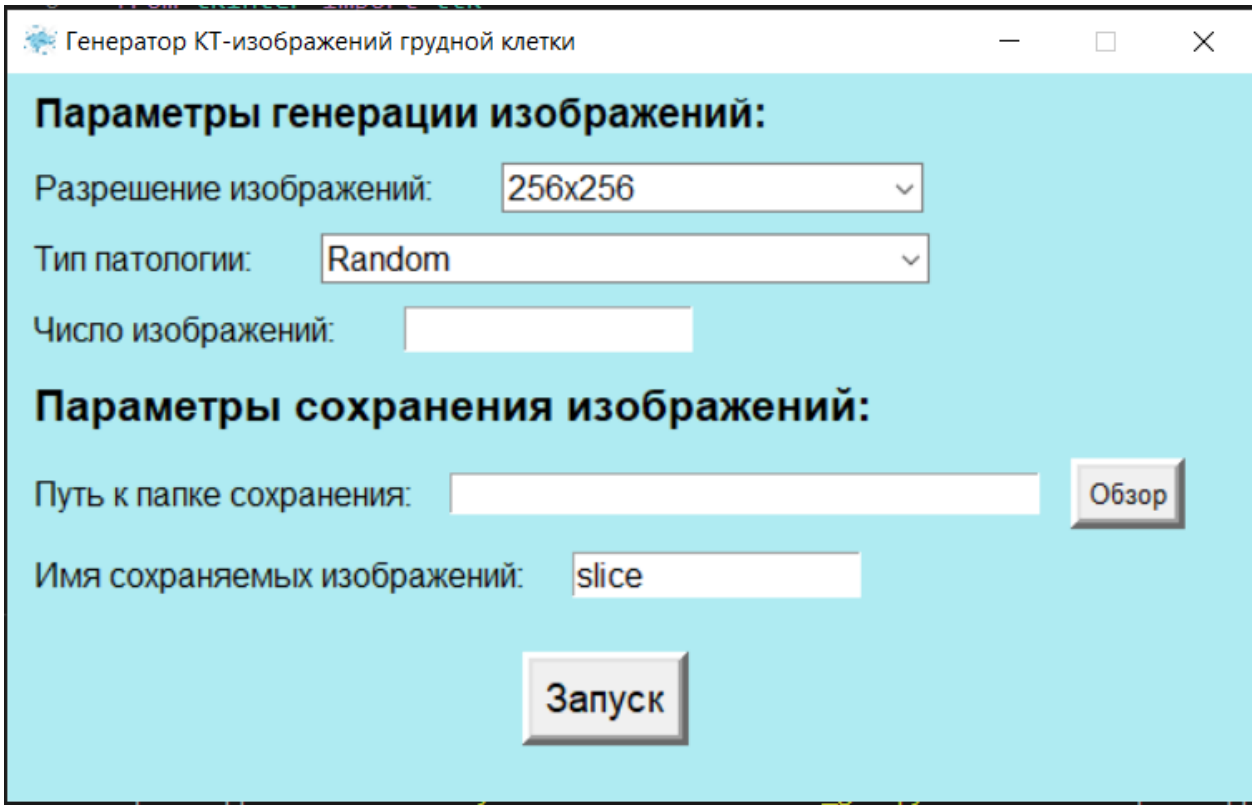

Рисунок 25 - Графический пользовательский интерфейс

Поля для выбора разрешения генерируемых изображений и типа патологии, отображаемых на них, представляют из себя комбинированные списки. В первом из них пользователь может выбрать разрешение либо 256x256 пикселей, либо 128x128. В втором пользователь может выбрать одну из 6 патологий, либо случайную генерацию.

В поле число изображений, задается соответственно количество изображений, которое генератор должен создать. Если занести в это поле значение, не являющиеся натуральным числом, то при запуске генерации на экран будет выведено сообщение об ошибке, показанное на рисунке 26.

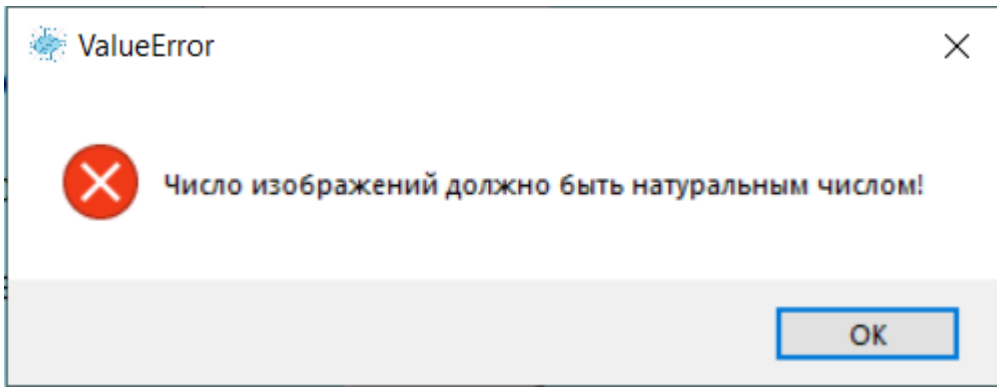

Рисунок 26 - Сообщение о неверном значении числа изображений

В поле путь к папке сохранения, можно самостоятельно вписать абсолютный путь к папке, в которую будут сохранятся сгенерированные изображения. Также при нажатии на кнопку обзор открывается диалоговое окно, как на рисунке 26, для выбора папки сохранения.

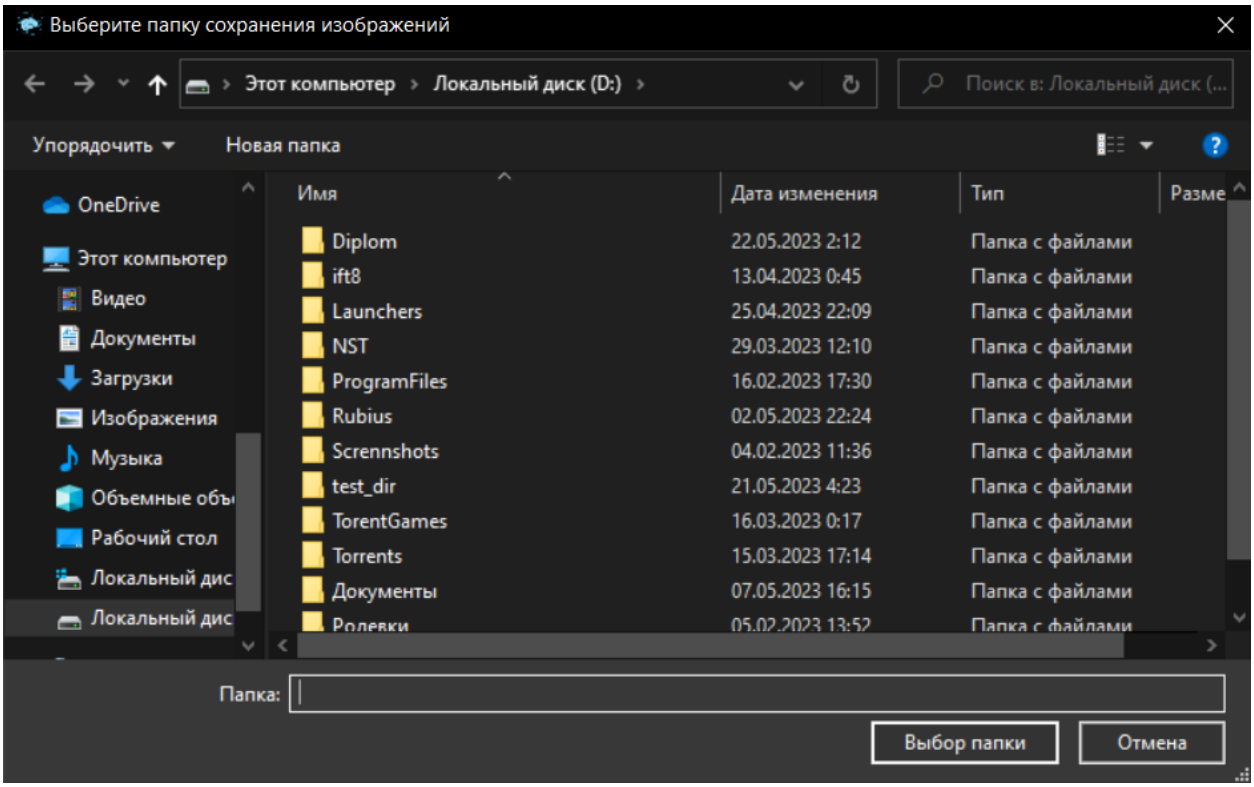

Рисунок 27 - Диалоговое окно для выбора папки сохранения изображений

Если папки по введенному пути не существует, то на экран пользователя будет выведено сообщение об ошибке, такое как на рисунке 28.

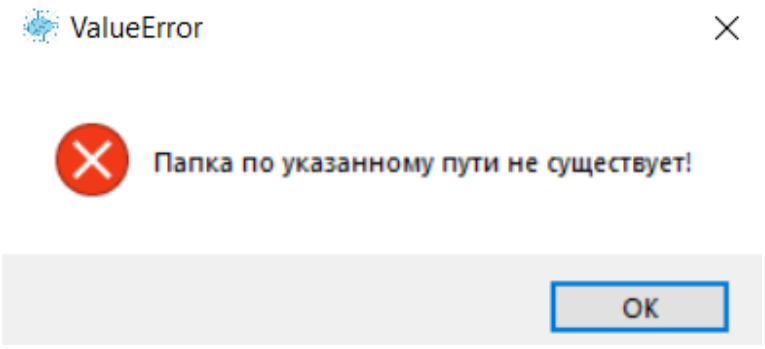

Рисунок 28 - Сообщение об отсутствии папки

В последнем поле задается общие имя для генерируемых изображений.

При нажатии на кнопку “Запуск”, введённые параметры генерации и сохранения передаются на вход скрипту, который создает изображения на основе переданных характеристик с помощью обученного генератора нейросети CProGAN и локально сохраняет изображения в формате dcm. Пример сгенерированных изображений, сохраненных в выбранную ранее папку показан на рисунке 29.

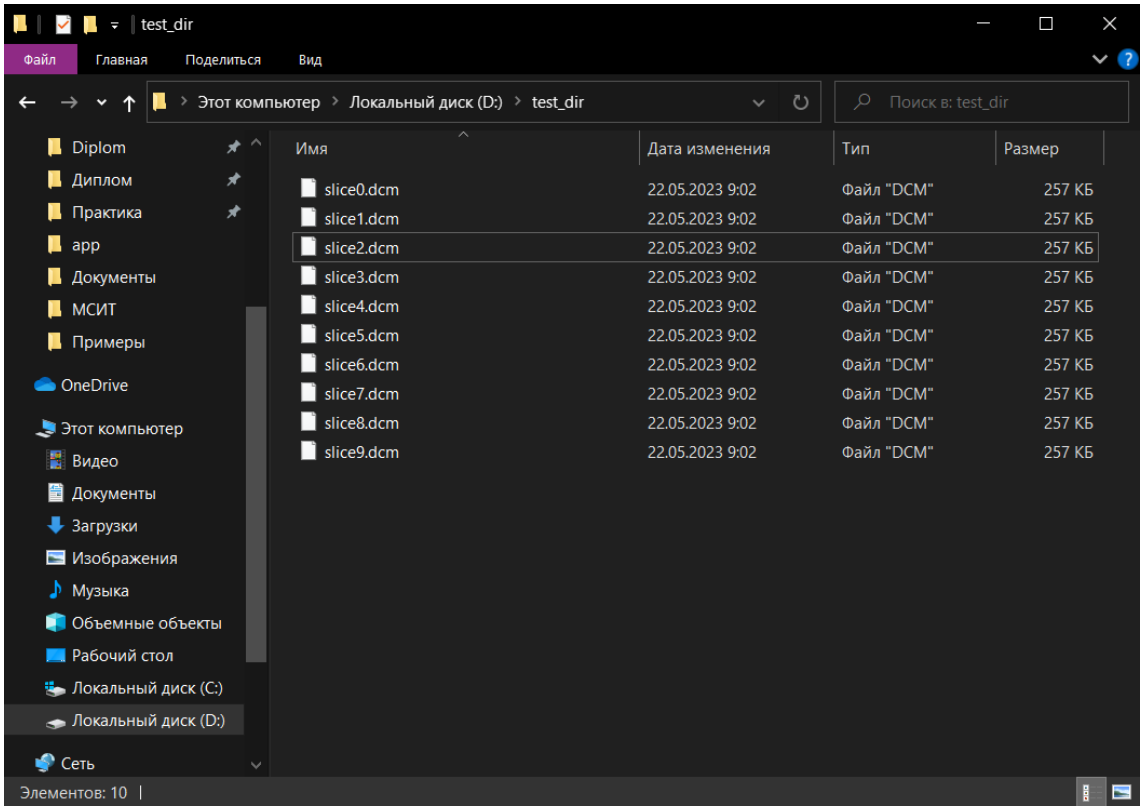

Рисунок 29 - Результат работы приложения

ЗАДАНИЕ К РАЗДЕЛУ
«ФИНАНСОВЫЙ МЕНЕДЖМЕНТ, РЕСУРСОЭФФЕКТИВНОСТЬ И РЕСУРСОСБЕРЕЖЕНИЕ»

Обучающемуся:

Группа	ФИО
8К93	Кузнецову Илье Евгеньевичу

Школа	ИШИТР	**Отделение школы (НОЦ)**	ОИТ
Уровень образования	Бакалавриат	**Направление/специальность**	09.03.04 Программная инженерия

Исходные данные к разделу «Финансовый менеджмент, ресурсоэффективность и ресурсосбережение»:	
1. Стоимость ресурсов научного исследования (НИ): материально-технических, энергетических, финансовых, информационных и человеческих	Стоимость материалов и оборудования учитывается в соответствии с рыночными ценами оценивается в 75000 руб. Оклад студента-разработчика в соответствии с МРОТ по Томской области составит 16424 руб. Оклад научного руководителя составит 50000 руб.
2. Нормы и нормативы расходования ресурсов	Премиальный коэффициент руководителя 35%; Премиальный коэффициент инженера 25%; Доплаты и надбавки руководителя 20%; Дополнительная заработная плата 10%; Накладные расходы 5%; Районный коэффициент Томска равный 1.3
3. Используемая система налогообложения, ставки налогов, отчислений, дисконтирования и кредитования	Коэффициент отчислений на уплату во внебюджетные фонды равный 30 %
Перечень вопросов, подлежащих исследованию, проектированию и разработке:	
1. Оценка коммерческого потенциала, перспективности и альтернатив проведения НИ с позиции ресурсоэффективности и ресурсосбережения	Определение конечного потребителя. Проведение SWOT-анализа реализации проекта.
2. Планирование и составление бюджета проекта	Определение целей и ожидаемых результатов, требований проекта. Расчет трудоемкости выполнения работ. Подсчет бюджета проекта.
3. Определение ресурсной (ресурсосберегающей), финансовой, бюджетной и экономической эффективности	Расчет показателей финансовой эффективности, ресурсоэффективности и эффективности исполнения.
Перечень графического материала (с точным указанием обязательных чертежей):	
1. Оценка конкурентоспособности технических решений 2. Матрица SWOT 3. Альтернативы проведения НИ 4. График проведения и бюджет НИ 5. Оценка ресурсной, финансовой и экономической эффективности НИ	

Дата выдачи задания к разделу в соответствии с календарным учебным графиком	

Задание выдал консультант по разделу «Финансовый менеджмент, ресурсоэффективность и ресурсосбережение»:

Должность	ФИО	Ученая степень, звание	Подпись	Дата
Профессор ОСГН	Гасанов Магеррам Али оглы	Доктор экономических наук		

Задание принял к исполнению обучающийся:

Группа	ФИО	Подпись	Дата
8К9З	Кузнецов Илья Евгеньевич		

3. Финансовый менеджмент, ресурсоэффективность и ресурсосбережение

3.1. Введение

Разработкой НИ занимаются студент-инженер и научный руководитель.

Цель данной выпускной квалификационной работы заключается в проектировании и разработке генеративно-состязательной нейросети для генерации КТ-изображений грудной клетки и графического пользовательского интерфейса к ней.

Целью раздела «Финансовый менеджмент, ресурсоэффективность и ресурсосбережение» является определение перспективности и успешности НИ, оценка его эффективности, уровня возможных рисков, разработка механизма управления и сопровождения конкретных проектных решений на этапе реализации.

Для достижения поставленной цели необходимо выполнить следующие задачи:

1. Проанализировать альтернативные варианты реализации проекта;
2. Оценить коммерческий потенциал и перспективность разработки проекта;
3. Провести оценку научно-технического уровня исследования и оценку рисков;
4. Составить план работ по реализации проекта;
5. Рассчитать бюджет проекта.

3.2. Потенциальные потребители результатов исследования

В качестве потенциальных потребителей рассматриваются специалисты по машинному обучению граждане Российской Федерации, а также дружественных стран, занимающиеся разработкой нейросетей, обучающихся на КТ-изображениях грудной клетки.

Для обучения нейросетей требуются тысячи разнообразных изображений, однако найти достаточное количество необходимых кт-

изображений в свободном доступе может быть очень сложно, поскольку данные изображения относятся к медицинской тайне и редко выходят за пределы лечебных учреждений. Получение же новых КТ-изображений требует значительных затрат времени и денег, а также может быть связано с некоторыми рисками для пациента. Так, например в Томске стоимость одного обследования на компьютерном томографе может достигать 4000 рублей.

Еще одними потенциальными пользователями приложения могут стать студенты-медики, которые будут использовать сгенерированные изображения в практике диагностирования различных патологий.

3.3. Анализ конкурентных технических решений

Прямых конкурентов разрабатываемой генеративно-состязательной нейросети на российском рынке нет, поэтому анализ конкурентных технических решений с позиции ресурсоэффективности и ресурсосбережения будет основываться на сравнении разрабатываемой системы и двух других популярных нейросетей, используемых для генерации различных изображений:

- Midjourney — это система на базе искусственного интеллекта, которая создает изображения из подсказок пользователя. Нейросеть разработала одноименная американская компания, основанная ученым и бывшим сотрудником Института Макса Планка Дэвидом Хольцем.
- StableDifusion — это программное обеспечение, создающее изображения по текстовым описаниям, с открытым исходным кодом. Разработано группой компаний CompVis в Мюнхенском университете.

Анализ конкурентных технических решений был проведен с помощью оценочной карты, представленной в таблице 1.

Где Бр – баллы нейросети ProGAN (текущая разработка);

Бm - баллы нейросети Midjourney;

Бs - баллы нейросети StableDifusion.

Таблица 1 – оценочная карта сравнения конкурентных технических решений

№	Критерии оценки	Вес критерия	Баллы			Конкурентоспособность		
			Бр	Бm	Бs	Кр	Кm	Кs
Технические критерии оценки ресурсоэффективности								
1	Потребность в ресурсах памяти	0,05	0,05	1	3	2	0,05	0,15
2	Функциональность	0,1	0,1	3	2	3	0,3	0,2
3	Быстродействие	0,2	0,2	1	3	2	0,2	0,6
4	Простота эксплуатации	0,1	0,1	4	3	2	0,4	0,3
5	Качество пользовательского интерфейса	0,2	0,2	2	1	1	0,4	0,2
Экономические критерии оценки эффективности								
1	Стоимость разработки	0,1	4	2	3	0,4	0,5	0,2
2	Стоимость поддержки системы	0,05	3	1	2	0,05	0,15	0,05
3	Рентабельность	0,1	1	3	1	0,2	0,1	0,1
4	Риски, связанные с выходом на рынок	0,1	1	3	2	0,1	0,1	0,1
	Итого	1	20	21	18	2,1	2,3	1,65

Анализ конкурентных решений определяется по формуле 1:

$$K = \Sigma Bi \times Бi, \quad (5)$$

где К – конкурентоспособность научной разработки или конкурента;

Bi – вес показателя (в долях единицы);

Бi – балл i-го показателя.

Согласно полученной оценочной карте генеративно-состязательная нейросеть слегка уступает по конкурентоспособности нейросети Midjourney, однако это компенсируется тем, что из-за своей узкой специализации ProGAN способна генерировать более качественные КТ-изображения по сравнению с универсальной Midjourney. Более того, НИ по началу будет распространяется

по свободной лицензии, что делает ее более доступной для потенциального потребителя по сравнению с Midjourney, для пользования которой необходимо оплачивать ежемесячную подписку.

3.4. SWOT-анализ

Также произведем в данном разделе SWOT-анализ НИ, позволяющий оценить факторы и явления, способствующие или препятствующие продвижению проекта на рынок.

Сильные стороны – это факторы, которые положительно сказываются на развитии проекта. Сюда обычно включают все, что превращает функционирование в успешную и конкурентную работу.

Слабые стороны – это недостаток, упущение или ограниченность научно-исследовательского проекта, которые препятствуют достижению его целей. Это то, что плохо получается в рамках проекта или где он располагает недостаточными возможностями или ресурсами по сравнению с конкурентами.

Возможности включают в себя любую предпочтительную ситуацию в настоящем или будущем, возникающую в условиях окружающей среды проекта: тенденцию, изменение или предполагаемую потребность, которая поддерживает спрос на результаты проекта и позволяет руководству проекта улучшить свою конкурентную позицию.

Угроза представляет собой любую нежелательную ситуацию, тенденцию или изменение в условиях окружающей среды проекта, которые имеют разрушительный или угрожающий характер для его конкурентоспособности в настоящем или будущем. В качестве угрозы может выступать барьер, ограничение или что-либо еще, что может повлечь за собой проблемы, разрушения, вред или ущерб, наносимый проекту.

Процесс анализа можно разделить на два этапа. Начальный выявление сильных и слабых сторон решения, возможностей, а также угроз, завершающий – определение соответствий утверждений, определенных на

предыдущем этапе, между собой. Результаты выполненной в ходе данного этапа работы приведены в таблице 2:

Таблица 2 – SWOT анализ

	Сильные стороны: С1. Отсутствие идентичных сервисов; С2. Качество генерируемых изображений может улучшаться с течением времени; С3. Интуитивно понятный пользовательский интерфейс.	Слабые стороны: Сл1. Отсутствие опыта разработки генеративных нейросетей; Сл2. Большие временные затраты.
Возможности: В1. Возможность улучшения качества генерируемых изображений и расширения выбора параметров генерации; В2. Использование актуальных и популярных инструментов разработки.	Возможность улучшения качества генерируемых изображений и расширения выбора параметров генерации позволит улучшить качество сервиса и обслуживания пользователей, что приведет к приросту числа этих самых пользователей.	Соотнося возможности и слабые стороны проекта будет уместным сделать вывод, что система может быть не разработана вовремя.
Угрозы: У1. Нейросеть может не получить одобрения со стороны регуляторных органов и медицинских сообществ, если ее эффективность и безопасность не будут доказаны; У2. Нехватка специалистов для дальнейшей поддержки проекта;	Соотношение сильных стороны с угрозами проекта говорят о том, что несмотря на отсутствие похожего сервиса, проект может быть выведен на реальный рынок, только если его эффективность будет доказана.	Слабые стороны и угрозы сигнализируют о том, что система может не иметь спрос у пользователей на длинном промежутке времени.

Из результатов проведенного SWOT-анализа можно сделать вывод, что несмотря на угрозы и слабые стороны, планируемое решение обладает сильными сторонами и возможностями для развития, что делает его выгодным для разработки и выведения на рынок.

3.5. Определение возможных альтернатив проведения научных исследований

В таблице 3 представлены возможные варианты реализации разработки.

Таблица 3 – Морфологическая таблица

	1	2
А. Тип разрабатываемой нейросети	GAN	StableDifusion
Б. Основной фреймоворк разработки	Pytorch	TensorFlow
В. Среда разработки	Google Colaboratory	Jupyter Notebook
Г. Тип пользовательского интерфейса	Сайт	Настольное приложение
Д. Методология разработки	Kanban	Agile

Основным вариантом решения является А1Б1В2Г2Д1. Выбор данного варианта решения обусловлен такими факторами, как наличие опыта разработки с использованием данных технологий, а также относительно большим объемом информации, которую можно найти по выбранной теме.

Альтернативными вариантами решения являются:

- А1Б1В1Г2Д2 – вариант, который первоначально прорабатывался на начальных этапах выпускной работы, однако от него пришлось отказаться, по причине сильных ограничений бесплатной версии среды разработки;
- А2Б2В1Г1Д1 – вариант, который возможного мог привести, к получению нейросети генерирующей более качественные изображения и более удобного сервиса. Не был выбран по причине отсутствия у студента опыта в разработке веб-приложений и малого объема информация об архитектуре нейросети StableDifusion.

3.6. Планирование работ по научно-техническому исследованию

3.6.1. Структура работ в рамках научного исследования

Планирование комплекса предполагаемых работ осуществляется в следующем порядке:

1. Определение структуры работ в рамках научного исследования.
2. Определение участников каждой работы.
3. Установление продолжительности работ.
4. Построение графика проведения научных исследований.

Для выполнения научных исследований формируется рабочая группа, в состав которой могут входить научные сотрудники и преподаватели, инженеры, техники и лаборанты, численность групп может варьироваться. По каждому виду запланированных работ устанавливается соответствующая должность исполнителей. Перечень этапов и работ, распределение исполнителей по данным видам работ приведен в таблице 4.

Таблица 4 – Перечень этапов, работ и распределение исполнителей

Основные этапы	№ раб	Содержание работ	Должность исполнителя
Выбор направления исследований	1	Составление и утверждение темы бакалаврской работы	Руководитель Студент
Содержание проекта	2	Определение содержания проекта	Руководитель Студент
Техническое задание	3	Постановка требований к программному обеспечению	Руководитель Студент
	4	Разработка бюджета проекта	Студент
	5	Создание календарного плана-графика	Руководитель Студент
Проектирование программного обеспечения	6	Проектирование моделей нейростей	Студент
	7	Проектирование процесса обучения нейросетей	Студент
	8	Проектирование пользовательского интерфейса	Студент
Разработка программного обеспечения	9	Разработка различных архитектур нейростей	Студент
	10	Разработка пользовательского интерфейса	Студент
Тестирование	11	Тестирование обученных нейросетей	Студент

Документация	12	Подготовка документации	Студент
Оформление отчета по НИР	13	Разработка плана оформления ПЗ	Студент
	14	Оформление ПЗ	

3.6.2. Структура работ в рамках научного исследования

Трудовые затраты в большинстве случаев образуют основную часть стоимости разработки, поэтому важным моментом является определение трудоемкости работ каждого из участников научного исследования.

Трудоемкость выполнения научного исследования оценивается экспертным путем в человеко-днях и носит вероятностный характер, который зависит от множества трудно учитываемых факторов. Для определения, ожидаемого (среднего) значения трудоемкости $t_{\text{ожi}}$ используется следующая формула:

$$t_{\text{ожi}} = \frac{3t_{\min i}+2t_{\max i}}{5}, \quad (6)$$

где $t_{\text{ожi}}$ – ожидаемая трудоемкость выполнения i-ой работы чел.-дн.;

$t_{\min i}$ – минимально возможная трудоемкость выполнения заданной i-ой работы, чел.-дн.;

$t_{\max i}$ – максимально возможная трудоемкость выполнения заданной -iой работы, чел.-дн.

Исходя из ожидаемой трудоемкости работ, определяется продолжительность каждой работы в рабочих днях $T_{\text{р}}$, учитывающая параллельность выполнения работ по нескольким исполнителям:

$$T_{pi} = \frac{t_{\text{ож}\,i}}{\text{Ч}_i}, \quad (7)$$

где T_{pi} – продолжительность одной работы, раб.дн.;

$t_{\text{ож}\,i}$– ожидаемая трудоемкость выполнения одной работы, чел.-дн.;

Ч_i– численность исполнителей, выполняющих одновременно одну и ту же работу на данном этапе, чел.

3.6.3. Разработка графика проведения научного исследования

Наиболее удобным и наглядным представлением проведения научных работ является построение ленточного графика в форме диаграммы Ганта.

Диаграмма Ганта – горизонтальный ленточный график, на котором работы по теме представляются протяженными во времени отрезками, характеризующимися датами начала и окончания выполнения данных работ.

Для удобства построение графика, длительность каждого из этапов работ из рабочих дней следует перевести в календарные дни. Для этого необходимо воспользоваться следующей формулой:

$$T_{ki} = T_{pi} \times k_{\text{кал}}\,, \tag{8}$$

где T_{ki} – продолжительность выполнения i-й работы в календарных днях;

T_{pi} – продолжительность выполнения i-й работы в рабочих днях;

$k_{\text{кал}}$ – коэффициент календарности.

Коэффициент календарности определяется по следующей формуле:

$$k_{\text{кал}} = \frac{T_{\text{кал}}}{T_{\text{кал}} - (T_{\text{вых}} + T_{\text{пр}})} = 1{,}48 \tag{9}$$

Таблица 4.5 – Временные показатели проведения научного исследования

Название работы	Трудоёмкость работ									Исполнители	Длительность работ в рабочих днях T_{pi}			Длительность работ в календарных днях T_{ki}		
	T_{min}, чел–дни			T_{max}, чел–дни			$T_{ожi}$, чел– дни									
	Исп.1	Исп.2	Исп.3	Исп.1	Исп.2	Исп.3	Исп.1	Исп.2	Исп.3		Исп.1	Исп.2	Исп.3	Исп.1	Исп.2	Исп.3
Составление и утверждение темы бакалаврской работы	1	1	1	2	2	2	1,4	1,4	1,4	Студент, научный руководитель	1	1	1	1	1	1
Определение содержания проекта	1	1	1	2	2	2	1,4	1,4	1,4	Студент, научный руководитель	1	1	1	1	1	1
Постановка требований к программному обеспечению	2	2	3	4	4	5	2,8	2,8	3,8	Студент, научный руководитель	1	1	1	1	1	1
Разработка бюджета проекта	5	6	7	8	8	9	6,2	6,8	7,8	Студент	6	6	7	8	8	10

Создание календарного плана-графика	2	2	2	3	3	4	2,4	2,4	2,8	Студент, научный руководитель	1	1	1	1	1	1
Проектирование моделей нейростей	10	12	14	16	18	20	12,4	14,4	16,4	Студент	12	14	16	17	20	23
Проектирование процесса обучения нейросетей	7	9	9	10	12	12	8,2	10,2	10,2	Студент	8	10	10	11	14	14
Проектирование пользовательского интерфейса	14	14	18	19	19	20	16	16	18,8	Студент	16	16	18	23	23	26
Разработка различных архитектур нейростей	36	38	40	44	46	48	39,2	41,2	43,2	Студент	39	41	43	57	60	63
Разработка пользовательского интерфейса	3	4	4	5	5	6	3,8	4,4	4,8	Студент	3	4	4	4	5	5
Тестирование обученных нейросетей	2	3	3	4	5	5	3,2	3,8	3,8	Студент	3	3	3	4	4	4
Подготовка документации	5	7	7	8	8	10	6,2	7,4	8,2	Студент	6	7	8	8	10	11
Разработка плана оформления ПЗ	1	2	2	3	3	4	1,8	2,4	2,8	Студент	1	2	2	1	2	2
Оформление ПЗ	5	7	7	8	9	10	6,2	7,8	8,2	Студент	6	7	8	8	10	11

Составлен план научного исследования, в котором разработан календарный план выполнения работ. Для построения таблицы временных показателей проведения НИ был рассчитан коэффициент календарности. С помощью показателей в табл. 6 был разработан календарный план-график проведения НИ по теме. Для иллюстрации календарного плана была использована диаграмма Ганта, указывающая на целесообразность проведения данного исследования.

Таблица 6 – Календарный план-график проведения научного исследования

№	Название работы	Исполнители	Продолжительность выполнения работ								
			Сентябрь	Октябрь	Ноябрь	Декабрь	Январь	Февраль	Март	Апрель	Май
1	Составление и утверждение темы бакалаврской работы	Студент. НР									
2	Определение содержания проекта	Студент НР									
3	Постановка требований к программному обеспечению	Студент НР									
4	Разработка бюджета проекта	Студент									
5	Составление календарного плана-графика	Студент НР									
6	Проектирование моделей нейростей	Студент									
7	Проектирование процесса обучения нейросетей	Студент									

8	Проектирование пользовательского интерфейса	Студент									
9	Разработка различных архитектур нейростей	Студент									
10	Разработка пользовательского интерфейса	Студент									
11	Тестирование обученных нейросетей	Студент									
12	Подготовка документации	Студент									
13	Разработка плана оформления ПЗ	Студент									
14	Оформление ПЗ	Студент									

3.7. Бюджет научно-технического исследования (НТИ)

3.7.1. Расчет материальных затрат НТИ

Материальные затраты учитываются в стоимости оборудования и накладным расходам. Так как заранее не известно, какое количество вторичных ресурсов понадобится, поэтому берется условная сумма материальных затрат равная 1500 рублей.

3.7.2. Расчет затрат на специальное оборудование для научных работ

Для обучения нейросетей после их реализации был приобретён мощный ноутбук с рыночной стоимостью 75000 рублей.

3.7.3. Основная заработная плата исполнителей

Данная статья включает в себя расчет основной заработной платы исполнителей: студента и научного руководителя. Расчет основной заработной платы приводится в таблице 7.

Таблица 7 – Расчет основной заработной платы

№	Наименование этапов	Исполнители	Трудоемкость, чел.-дн.			Заработная плата, приходящаяся на один чел.-дн., тыс. руб.	Всего заработная плата по тарифу (окладам), тыс. руб.		
			Исп. 1	Исп. 2	Исп. 3		Исп. 1	Исп. 2	Исп. 3
1	Составление и утверждение темы бакалаврской работы	Руководитель	3	3	3	3,23	9,69	9,69	9,69
		Студент	1	1	1	1,96	1,96	1,96	1,96
2	Определение содержания проекта	Руководитель	3	3	3	3,23	9,69	9,69	9,69
		Студент	5	5	5	1,96	9,8	9,8	9,8
3	Постановка требований к программному обеспечению	Руководитель	1	1	1	3,23	3,23	3,23	3,23
		Студент	3	3	3	1,96	5,88	5,88	5,88
4	Разработка бюджета проекта	Студент	4	4	4	1,96	7,84	7,84	7,84
5	Составление календарного плана-графика	Руководитель	3	3	3	3,23	9,69	9,69	9,69
		Студент	4	4	4	1,96	7,84	7,84	7,84
6	Проектирование моделей нейростей	Студент	12	12	15	1,96	23,52	23,52	29,4
7	Проектирование процесса обучения нейросетей	Студент	8	8	14	1,96	15,68	15,68	27,44
8	Проектирование пользовательского интерфейса	Студент	4	8	8	1,96	7,84	15,68	15,68
9	Разработка различных архитектур нейростей	Студент	50	50	45	1,96	98	98	88,2
10	Разработка пользовательского интерфейса	Студент	8	16	16	1,96	15,68	31,36	31,36
11	Тестирование обученных нейросетей	Студент	12	12	10	1,96	23,52	23,52	19,6
12	Подготовка документации	Студент	3	3	4	1,96	5,88	5,88	7,84
13	Разработка плана оформления ПЗ	Студент	3	3	4	1,96	5,88	5,88	7,84
14	Оформление ПЗ	Студент	7	8	10	1,96	13,72	15,68	19,6
						Итог	**275,34**	**300,82**	**312,58**

Статья включает основную заработную плату работников, непосредственно занятых выполнением проекта, (включая премии, доплаты) и дополнительную заработную плату и рассчитывается по формуле:

$$З_{зп} = З_{осн} + З_{доп} \quad (10)$$

где $З_{осн}$ – основная заработная плата;

$З_{доп}$ – дополнительная заработная плата.

Основная заработная плата руководителя рассчитывается по следующей формуле:

$$З_{осн} = З_{дн} \cdot Т_р \quad (11)$$

где $З_{осн}$– основная заработная плата одного работника;

$Т_р$ – продолжительность работ, выполняемых научно-техническим работником, раб. дн.;

$З_{дн}$ – среднедневная заработная плата работника, руб.

Среднедневная заработная плата рассчитывается по формуле:

$$З_{дн} = \frac{З_м \cdot М}{F_д} \quad (12)$$

где $З_м$ – месячный должностной оклад работника, руб.;

М – количество месяцев работы без отпуска в течение года:

при отпуске в 24 раб. дня М =11,2 месяца, 5–дневная неделя;

при отпуске в 48 раб. дней М=10,4 месяца, 6–дневная неделя;

$F_д$ – действительный годовой фонд рабочего времени научно–технического персонала, раб. дн.

Месячный должностной оклад работника (руководителя):

$$З_м = З_{тс} \cdot (1 + k_{пр} + k_д) \cdot k_р \quad (13)$$

где $З_{тс}$ – заработная плата по тарифной ставке, руб.;

$k_{пр}$ – премиальный коэффициент, равный 0,35;

$k_д$ – коэффициент доплат и надбавок составляет 0,2;

$k_р$ – районный коэффициент, равный 1,3 (для Томска).

Тарифный коэффициент для НР = 1,866; для С = 1,407.

Расчет основной заработной платы представлен в таблице 8

Таблица 8 – Расчет основной заработной платы при условии распределения рабочих дней для исп. 1

Исполнители	Разряд	$k_т$	$З_{тс}$, руб.	$k_{пр}$	$k_д$	$k_р$	$З_м$, руб.	$З_{дн}$, руб.	$Т_р$, раб. дн.	$З_{осн}$, руб.
Научный руководитель	Доцент	1,866	50000	0,35	0,2	1,3	77500	3229	13	41977
Студент	Инженер	1,407	16424	0,35	0,2	1,3	25457	979	105	102795
Итого										144772

3.7.4. Расчет дополнительной заработной платы

Дополнительная заработная плата учитывает величину предусмотренных Трудовым кодексом РФ доплат за отклонение от нормальных условий труда, а также выплат, связанных с обеспечением гарантий и компенсаций.

Расчет дополнительной заработной платы рассчитывается по формуле:

$$З_{доп} = k_{доп} \cdot З_{осн}\ , \qquad (14)$$

где $k_{доп}$ – коэффициент дополнительной заработной платы, принятый на стадии проектирования за 0,1.

Тем самым дополнительная заработная плата составит 4197 руб. для научного руководителя и 10279,5 рублей для студента.

3.7.5. Отчисления во внебюджетные фонды

В данной статье расходов отражаются обязательные отчисления по установленным законодательством Российской Федерации нормам органам государственного социального страхования (ФСС), пенсионного фонда (ПФ) и медицинского страхования (ФФОМС) от затрат на оплату труда работников.

Величина отчислений во внебюджетные фонды определяется исходя из формулы:

$$З_{внеб} = k_{внеб} \cdot (З_{осн} + З_{доп}) \qquad (15)$$

где $k_{внеб}$ – коэффициент отчислений на уплату во внебюджетные фонды (пенсионный фонд, фонд обязательного медицинского страхования и пр.).

В соответствии с Федеральным законом от 24.07.2009 №212-ФЗ установлен размер страховых взносов равный 30 %.

Расчет отчислений во внебюджетные фонды представлены в таблице 9.

Таблица 9 – Расчет отчислений во внебюджетные фонды

Исполнитель	**Основная заработная плата, руб.**			**Дополнительная заработная плата, руб.**		
	Исп.1	**Исп.2**	**Исп.3**	**Исп.1**	**Исп.2**	**Исп.3**
Руководитель проекта	41977	41977	41977	4197	4197	4197
Студент	102795	109648	116501	10279,5	10964,8	11650,1
Коэффициент отчислений во внебюджетные фонды	0,3					
Итого						
Отчисления	**Исп.1, руб.**		**Исп.2, руб.**		**Исп.3, руб.**	
	47774,76		50036,25		52297,74	

3.7.6. Накладные расходы

Накладные расходы учитывают прочие затраты организации, не попавшие в предыдущие статьи расходов, для таких проектов они обычно не превышают 7% из-за количества людей и объемов работы. Их величина определяется по формуле:

$$З_{накл} = (\sum статей) \cdot k_{нр} \qquad (16)$$

где $k_{нр}$ – коэффициент, учитывающий накладные расходы, равный 5%.

Расчет накладных расходов для каждого исполнения представлен в таблице 10

Таблица 10 – Расчет накладных расходов

Наименование статьи затрат	Сумма, руб.		
	Исп. 1	Исп. 2	Исп. 3
Материальные затраты	1500	1500	1500
Затраты на специальное оборудование	75000	75000	75000

Основная заработная плата	144772	151625	158478
Дополнительная заработная плата	14477,2	15162,5	15847,8
Отчисления во внебюджетные фонды	47774,76	50036,25	52297,74
Сумма статей	283524	293324	303124
Накладные расходы	**14176**	**14666**	**15156**

3.7.7. Формирование бюджета затрат научно-исследовательского проекта

Рассчитанная величина затрат научно–исследовательской работы является основой для формирования бюджета затрат проекта. Определение бюджета затрат на научно–исследовательский проект приведено в таблице 11.

Таблица 11 –Расчет бюджета затрат НТИ

Наименование статьи	Сумма, руб.			Примечание
	Исп.1	Исп.2	Исп.3	
1. Материальные затраты	1500	1500	1500	Пункт 4.7.1
2. Затраты на специальное оборудование	75000	75000	75000	Пункт 4.7.2
3. Основная заработная плата	144772	151625	158478	Пункт 4.7.3
4. Дополнительная заработная плата	14477,2	15162,5	15847,8	Пункт 4.5.4
5. Отчисления во внебюджетные фонды	47774,76	50036,25	52297,74	Пункт 4.7.5
6. Затраты на научные и производственные командировки	-	-	-	Отсутствуют
7. Контрагентские расходы	-	-	-	Отсутствуют
8. Накладные расходы	14176	14666	15156	Пункт 4.7.6
9. Бюджет затрат НТИ	297700	307990	318280	

3.8. Определение ресурсной (ресурсосберегающей), финансовой, бюджетной, социальной и экономической эффективности исследования

Определение эффективности происходит на основе расчета интегрального показателя эффективности научного исследования. Его нахождение связано с определением двух средневзвешенных величин: финансовой эффективности и ресурсоэффективности.

Интегральный показатель финансовой эффективности научного исследования определяется как:

$$I_{\text{фин.р}}^{\text{исп.}i} = \frac{\Phi_{pi}}{\Phi_{max}} \quad (17)$$

где $I_{\text{фин.р}}^{\text{исп.}i}$ – интегральный финансовый показатель разработки;

Φ_{pi} – стоимость i-го варианта исполнения;

Φ_{max} – максимальная стоимость исполнения научно-исследовательского проекта.

$$I_{\text{фин.р}}^{\text{исп1}} = \frac{297700}{318280} = 0{,}94;$$

$$I_{\text{фин.р}}^{\text{исп2}} = \frac{307990}{318280} = 0{,}97;$$

$$I_{\text{фин.р}}^{\text{исп3}} = \frac{318280}{318280} = 1$$

Интегральный показатель ресурсоэффективности вариантов исполнения объекта исследования можно определить следующим образом:

$$I_{pi} = \sum_{i=1}^{n} a_i \times b_i \quad (18)$$

где I_{pi} – интегральный показатель ресурсоэффективности для i-го варианта исполнения разработки;

a_i – весовой коэффициент i-го варианта исполнения разработки;

b_i– бальная оценка i-го варианта исполнения разработки, устанавливается экспертным путем по выбранной шкале оценивания;

n – число параметров сравнения.

Расчет интегрального показателя ресурсоэффективности представлен в таблице 12.

Таблица 12 – Сравнительная оценка характеристик вариантов исполнения проекта

Объект исследования / **Критерии**	**Весовой коэффициент параметра**	**Исп.1**	**Исп.2**	**Исп.3**
1. Надежность	0,15	5	5	4
2. Скорость работы	0,2	4	4	5
3. Удобство в использовании	0,2	4	3	3
4. Масштабируемость	0,1	4	4	5
5. Доступность	0,15	4	3	3
6. Ограничения функциональности	0,2	3	4	4
Итого	1	4,15	3,8	3,95

Интегральный показатель эффективности вариантов исполнения разработки ($I_{\text{испi}}$) определяется на основании интегрального показателя ресурсоэффективности и интегрального финансового показателя по формуле:

$$I_{\text{исп1}} = \frac{I_{\text{рисп1}}}{I_{\text{фин.р}}^{\text{исп1}}} = \frac{4{,}15}{0{,}94} = 4{,}41$$

$$I_{\text{исп2}} = \frac{I_{\text{рисп2}}}{I_{\text{фин.р}}^{\text{исп2}}} = \frac{3{,}8}{0{,}97} = 3{,}92;$$

$$I_{\text{исп3}} = \frac{I_{\text{рисп3}}}{I_{\text{фин.р}}^{\text{исп3}}} = \frac{3{,}95}{1} = 3{,}95.$$

Сравнение интегрального показателя эффективности вариантов исполнения разработки позволяет определить сравнительную эффективность проекта и выбрать наиболее целесообразный вариант из предложенных по формуле:

$$\text{Э}_{\text{ср}} = \frac{I_{\text{исп2}}}{I_{\text{исп1}}} \quad (19)$$

Расчет сравнительной эффективности проекта представлен в таблице 13.

Таблица 13 – Расчет сравнительной эффективность разработки

№ п/п	Показатели	Исп.1	Исп.2	Исп.3
1	Интегральный финансовый показатель разработки	0,94	0,97	1
2	Интегральный показатель ресурсоэффективности разработки	4,15	3,8	3,95
3	Интегральный показатель эффективности	4,41	3,92	3,95
4	Сравнительная эффективность вариантов использования	1	0,889	0,896

Сравнив значения интегральных показателей эффективности, можно сделать вывод, что реализация технологии в первом исполнении является наиболее эффективным вариантом решения задачи, поставленной в данной работе с позиции финансовой и ресурсной эффективности.

3.9. Вывод по разделу

В результате проделанной по разделу «Финансовый менеджмент, ресурсоэффективность и ресурсосбережение» работы, был исследован с экономической точки зрения проект, выполняемый в рамках научно-исследовательской работы, выделены слабые и сильные стороны проекта благодаря SWOT-матрице, рынок потребителей, проанализированы конкурентные технические решения, были рассчитаны временные показатели проекта и построен календарный график работ в виде диаграммы Ганта.

Полагаясь на результаты проведенной работы, разработка генеративно-состязательной характеризуется как конкурентоспособная и перспективная. Рассчитанный бюджет составил 297700 рублей.

Сравнение интегральных показателей эффективности вариантов исполнения показало, что наиболее выгодным с точки зрения ресурсоэффективности является 1 вариант исполнения, который и был реализован.

ЗАДАНИЕ К РАЗДЕЛУ «СОЦИАЛЬНАЯ ОТВЕТСТВЕННОСТЬ»

Обучающемуся:

Группа	ФИО
8К93	Кузнецов Илья Евгеньевич

Школа	ИШИТР	Отделение (НОЦ)	ОИТ
Уровень образования	Бакалавриат	Направление/ специальность	09.03.04 Программная инженерия

Исходные данные к разделу «Социальная ответственность»:	
Введение – Характеристика объекта исследования (вещество, материал, прибор, алгоритм, методика) и области его применения. – Описание рабочей зоны (рабочего места) при разработке проектного решения/при эксплуатации	*Объект исследования:* генеративно-состязательная нейросети с графическим пользовательским интерфейсом для генерации кт-изображений грудной клетки. *Область применения:* машинное обучение *Рабочая зона:* офисное помещение *Размеры помещения:* 15 м^2. *Количество и наименование оборудования рабочей зоны:* 1 персональный компьютер. *Рабочие процессы, связанные с объектом исследования, осуществляющиеся в рабочей зоне:* задание параметров генератора нейросети, получение кт-изображений грудной клетки, проверка сгенерированных изображений
Перечень вопросов, подлежащих исследованию, проектированию и разработке:	
1. Правовые и организационные вопросы обеспечения безопасности при разработке проектного решения: – специальные (характерные при эксплуатации объекта исследования, проектируемой рабочей зоны) правовые нормы трудового законодательства; – организационные мероприятия при компоновке рабочей зоны.	"Трудовой кодекс Российской Федерации" от 30.12.2001 N 197-ФЗ (ред. от 19.12.2022, с изм. от 11.04.2023); ТК РФ Статья 91. Понятие рабочего времени. Нормальная продолжительность рабочего времени; ГОСТ 12.2.032-78. Система стандартов безопасности труда. Рабочее место при выполнении работ сидя.
2. Производственная безопасность при разработке проектного решения: 2.1. Анализ выявленных вредных и опасных факторов 2.2. Обоснование мероприятий по снижению воздействия	**Вредные факторы:** 1. Отсутствие или недостаток необходимого искусственного освещения; 2. Повышенный уровень шума; 3. Производственные факторы, связанные с аномальными микроклиматическими параметрами воздушной среды на местонахождении работающего; 4. Психологические нагрузки, вызванные монотонной работой. **Опасные факторы:** 1. Факторы, связанные с электрическим током, вызываемым разницей электрических потенциалов, под действие которого попадает рабочий. **Требуемые средства коллективной защиты от выявленных факторов:** системы естественного освещения, приборы искусственного освещения, изоляционные средства, предохранительные устройства.

3. Экологическая безопасность при разработке проектного решения:	**Воздействие на селитебную зону:** не выявлено. **Воздействие на литосферу:** загрязнение среды из-за неверного способа утилизации рабочей техники. **Воздействие на гидросферу:** загрязнение среды из-за неверного способа утилизации рабочей техники. **Воздействие на атмосферу:** загрязнение среды из-за неверного способа утилизации рабочей техники.
4. Безопасность в чрезвычайных ситуациях при разработке проектного решения	**Возможные ЧС:** Природные (землетрясения); Техногенные (транспортные аварии, пожары, аварии с выбросом химически/радиоактивно опасных веществ и т.д.); Социальные (терроризм); Биологические (пандемия). **Наиболее типичная ЧС**: пожар.

Дата выдачи задания к разделу в соответствии с календарным учебным графиком	

Задание выдал консультант по разделу «Социальная ответственность»:

Должность	ФИО	Ученая степень, звание	Подпись	Дата
Старший преподаватель	Мезенцева Ирина Леонидовна			

Задание принял к исполнению обучающийся:

Группа	ФИО	Подпись	Дата
8К93	Кузнецов Илья Евгеньевич		

4. Социальная ответственность

4.1. Введение

Данный раздел выпускной квалификационной работы направлен на выявление опасных и вредных факторов, которые могут присутствовать при разработке и использования нейронное сети на рабочем месте оператора ПЭВМ. Также рассмотрены меры по снижению и предупреждению вредных воздействий на окружающую среду, исследованы правовые и организационные вопросы обеспечения безопасности при чрезвычайных ситуациях.

Разработка и тестирование программного обеспечения в рамках выпускной квалификационной работы осуществляется в рабочей зоне, которая представляет из себя офисное помещенье площадью 15 м², оборудованное персональным компьютером.

4.2. Правовые и организационные вопросы обеспечения безопасности

4.2.1. Специальные правовые нормы трудового законодательства

Правовые вопросы организации труда работника, обеспечение его безопасности регулируется Трудовым кодексом Российской Федерации от 30.12.2001 N 197-ФЗ.

В соответствии со ст.100 Трудового кодекса РФ режим рабочего времени устанавливается правилами внутреннего трудового распорядка, которые, в свою очередь, утверждаются работодателем с учетом мнения представительного органа работников. Согласно ст. 91 Трудового кодекса РФ продолжительность рабочего времени, которая не должна быть меньше указанного времени в трудовом договоре, но, в свою очередь, не должна превышать 40 часов в неделю. Также в ст.108 Трудового кодекса РФ говорится, что в течение рабочего дня работнику должен быть предоставлен перерыв для отдыха и питания продолжительностью не более двух часов и не менее 30 минут, который в рабочее время не включается.

Согласно ФЗ «Об обязательном социальном страховании от несчастных случаев на производстве» установлены правовые основы обязательного социального страхования и определен порядок возмещения вреда, причиненного жизни и здоровью работника при исполнении им обязанностей по трудовому договору.

В качестве поощрения работника за выполнение им трудовых обязанностей предусмотрена система оплаты труда, включающая размеры окладов, доплат и надбавок компенсационного характера и стимулирующего характера, системы премирования, установленные трудовыми договорами и соглашениями в соответствии с трудовым законодательством и иными нормативными правовыми актами.

4.2.2. Основные эргономические требования к правильному расположению и компоновке рабочей зоны

Рабочее место оператора ПК должно быть организовано с учетом требований ГОСТ 12.2.032–78, устанавливающим общие эргономические требования к рабочим местам при выполнении работ в положении сидя, а именно:

1. Экран видеомонитора должен находиться от глаз пользователя на расстоянии (600–700) мм, но не ближе 500 мм с учетом размеров алфавитно-цифровых знаков и символов.

2. Конструкция рабочей мебели должна обеспечивать возможность индивидуальной регулировки соответственно росту пользователя и создавать удобную позу для работы.

3. Конструкцией рабочего места должно быть обеспечено выполнение трудовых операций в пределах зоны досягаемости моторного поля.

4. Рабочие столы следует размещать таким образом, чтобы видеодисплейные терминалы были ориентированы боковой стороной к световым проемам, чтобы естественный свет падал преимущественно слева.

5. Часто используемые средства отображения информации, требующие менее точного и быстрого считывания показаний, допускается располагать в

вертикальной плоскости под углом ±30° от нормальной линии взгляда и в горизонтальной плоскости под углом ±30° от сагиттальной плоскости.

При выполнении выпускной квалификационной работы правовых и организационных нарушений по указанным требованиям не было выявлено, рабочее место было оборудовано согласно всем нормам и правилам.

4.3. Производственная безопасность

ГОСТ 12.0.003–2015 устанавливает вредные и опасные факторы, которые могут воздействовать на сотрудника. В таблице 1 перечислены факторы, которые могут возникнуть при работах по проектированию, разработке и тестированию веб-приложения.

Таблица 14 – Возможные опасные и вредные производственные факторы на рабочем месте инженера-программиста

Факторы	Нормативные документы
Отсутствие или недостаток необходимого искусственного освещения;	СанПиН 1.2.3685–21 Гигиенические нормативы и требования к обеспечению безопасности и (или) безвредности для человека факторов среды обитания
Повышенный уровень шума	СП 51.13330.2011 Защита от шума
Производственные факторы, связанные с аномальными микроклиматическими параметрами воздушной среды на местонахождении работающего	СанПиН 1.2.3685–21 Гигиенические нормативы и требования к обеспечению безопасности и (или) безвредности для человека факторов среды обитания
Психологические нагрузки, вызванные монотонной работой	СанПиН 1.2.3685–21 Гигиенические нормативы и требования к обеспечению безопасности и (или) безвредности для человека факторов среды обитания
Факторы, связанные с электрическим током, вызываемым разницей электрических потенциалов, под действие которого попадает рабочий	ГОСТ 12.1.019-2017 ССБТ. Электробезопасность. Общие требования и номенклатура видов защиты.

4.3.1. Отсутствие или недостаток необходимого искусственного освещения

В помещениях, где происходит работа за ПК используется естественное и искусственное освещение. Естественное освещение предполагает проникновение внутрь зданий солнечного света через окна и различного типа светопроемы. Нормы естественного освещения для разных зданий и помещений разрабатываются с учетом их назначения. В помещениях общественных зданий применяют систему комбинированного освещения в помещениях общественных зданий, где выполняется напряженная зрительная работа. Общее освещение в рабочих помещениях поддерживают равномерным.

Согласно требованиям СанПиН 1.2.3685–21 необходимо при проведении испытаний соблюдать определенные правила:

- применять комбинированную освещенность;
- естественный свет преимущественно должен падать слева;
- освещенность на поверхности стола в зоне размещения рабочего документа должна быть 300 – 500 лк;
- освещенность поверхности экрана не должна быть более 300 лк;
- в качестве источников света при искусственном освещении следует применять преимущественно люминесцентные лампы либо компактные светодиодные лампы;

Безопасность и здоровье условия труда в большой степени зависят от освещенности рабочих мест и помещений. Отсутствие или недостаток необходимого искусственного освещения может привести к ухудшению зрения, головным болям, нарушению работы сердечно-сосудистой и нервной системы, усталости и снижению работоспособности. Неудовлетворительное освещение утомляет не только зрение, но и вызывает утомление организма в целом. Неправильное освещение может быть причиной травматизма: плохо освещенные опасные зоны, слепящие лампы, резкие тени ухудшают или вызывают полную потерю зрения, ориентации. Правильное освещение

уменьшает количество несчастных случаев, повышает производительность труда.

4.3.2. Повышенный уровень шума

Повышенный шум влияет на нервную и сердечно-сосудистую системы, репродуктивную функцию человека, вызывает раздражение, нарушение сна, утомление, агрессивность, способствует психическим заболеваниям. При постоянном нахождение в помещение с высоким уровнем шума могут наблюдаться нарушения слуха.

Источники шума в офисном помещении могут быть различными. Некоторые из них могут быть связаны с работой самого офиса, например, звуки телефонов, компьютеров, принтеров, сканеров и другого офисного оборудования.

Уровень шума на рабочих местах разработчика-программиста не должен превышать значений, которые указаны в СП 51.13330.2011. Согласно пункту 6.3 уровень шума в офисном помещении не должен превышать значение в 65 дБА.

К мероприятиям по защите от шума относятся:

- Использование звукопоглощающих материалов для отделки помещения: использование ковровых покрытий, тканевых обоев, штор и других материалов, которые поглощают звук;
- Использование мебели и оборудования, которые не создают лишнего шума: выбор тихих компьютеров, принтеров, сканеров и другой оргтехники;
- Размещение рабочих мест в соответствии с принципами эргономики: установка перегородок между рабочими местами, размещение рабочих столов и кресел на определенном расстоянии друг от друга;

4.3.3. Производственные факторы, связанные с аномальными микроклиматическими параметрами воздушной среды на местонахождении работающего

Микроклимат – это метеорологические условия внутренней среды, определяемые действующими на организм человека сочетаниями температуры, относительно влажности и скорости движения воздуха. Причиной отклонения показателей от установленных норм зачастую является некорректная работа системы вентиляции офисного помещения, которая одновременно влияет и на температуру окружающего воздуха в помещении, на влажность, и на скорость его движения.

Высокий уровень влажности воздушных потоков внутри жилого или производственного помещения способствует развитию заболеваний дыхательной и мочевыделительной системы. Если температура воздуха повышена или понижена, это может нанести вред организму, вызвав его перегрев или переохлаждение.

Работа программиста относится к категории Iа тяжести труда, поскольку работы выполняются сидя и сопровождаются незначительным физическим напряжение с интенсивностью энергозатрат до 139 Вт/час. На рабочих местах пользователей ПК должны обеспечиваться оптимальные параметры микроклимата в соответствии с СанПиНом 1.2.3685–21 для категории тяжести работ Iа (таблица 15).

Таблица 15 – Допустимые величины параметров микроклимата

Период года	Температура воздуха, °С	Температура поверхностей, °С	Относительная влажность, %	Скорость движения воздуха, м/с
Холодный	18-24	21-25	30-60	0,3
Тёплый	18-28	19-26	30-65	0,25

В производственных помещениях, в которых допустимые нормативные величины показателей микроклимата невозможно установить, условия микроклимата рассматривают как вредные и опасные. В целях профилактики

неблагоприятного воздействия микроклимата используют системы местного кондиционирования воздуха, компенсацию неблагоприятного воздействия одного параметра микроклимата изменением другого, спецодежду и другие средства индивидуальной защиты, помещения для отдыха и обогревания, регламентацию времени работы, в частности, перерывы в работе, сокращение рабочего дня, увеличение продолжительности отпуска, уменьшение стажа работы и др.

4.3.4. Психологические нагрузки, вызванные монотонной работой

Работа за компьютером может негативно сказаться на человека, из-за большого однообразного потока информации, который программисту необходимо обрабатывать каждый день.

Монотонная работа может привести к снижению активности симпатической нервной системы, ухудшению иммунитета и потери остроты зрения.

Согласно СанПиН 1.2.3685–21, длительность сосредоточенного наблюдения должна составлять от 26 до 50% от времени смены, то есть не более 4-х часов при 8-часовом рабочем дне.

Для снижения психофизиологических нагрузок на работника необходимо соблюдать требования к режиму труда и отдыха, а также уделить пристальное внимание организации рабочего места.

4.3.5. Факторы, связанные с электрическим током, вызываемым разницей электрических потенциалов, под действие которого попадает рабочий

Электрическая безопасность включает в себя правовые, социально-экономические, организационно-технические, лечебно-профилактические, реабилитационные и иные мероприятия. Электрические установки, к которым относится практически все оборудование ЭВМ, представляют для человека потенциальную опасность, так как в процессе эксплуатации или проведения профилактических работ может произойти случайный контакт человека с

электрически неизолированными либо незаземленными частями ПК. Электрический ток оказывает на организм человека термическое, электролитическое и биологическое действия, приводящие к ожогам тела, нарушению физико-химического состава крови, а также к раздражению и возбуждению живых тканей организма, сопровождающиеся непроизвольными судорожными сокращениями мышц сердца, лёгких и т.д.

Вопросы требований к защите от поражения электрическим током освещены в ГОСТ 12.1.019-2017 ССБТ.

Мероприятия, направленные на предотвращение возможности поражения электрическим током, включают в себя следующее:

- при выполнении монтажных работ необходимо использовать только исправно работающий инструмент;
- запрет на выполнение работ на задней панели при включенном сетевом напряжении;
- выполнение работ по устранению неисправностей должно производиться компетентными людьми;
- нужно постоянно наблюдать за исправностью электропроводки и в случае обнаружения неисправностей незамедлительно принимать действия по их устранению.

4.4. Экологическая безопасность

Разработка проектного решения может оказывать косвенное влияние на природную среду, в частности на литосферу, гидросферу и атмосферу в связи с утилизацией компьютерной и офисной техники. Компьютер и другая оргтехника в своем составе содержит токсичные вещества. При завершении срока службы такого оборудования, его можно классифицировать, как отходы электронной промышленности. Неправильная утилизация этих отходов может произойти загрязнению почвы, подземных вод и атмосферного воздуха. Так при сжигании комплектующих электронно-вычислительных машин выделяются токсичные газы, такие как диоксид серы, оксиды азота и углерода (таблица 4), которые являются источником загрязнения атмосферы.

Таблица 16 – Предельно допустимые концентрации в атмосферном воздухе городских и сельских поселений

Наименования вещества	Формула	Предельно допустимые концентрации, мг/м3			Класс опасности
		максимальная разовая	среднесуточная	среднегодовая	
Диоксид серы	SO2	0,5	0,05	-	3
Оксид углерода	CO	20	30	10	2
Оксид азота	NO	0,4	-	0,06	3

Таким образом рабочее помещение инженера-программиста относится к IV категории объектов, оказывающих негативное воздействие на окружающую среду.

Утилизация, как электронно-вычислительных машин, так и другой оргтехники включает в себя работы по: погрузке, транспортировке, разгрузке, демонтажу и извлечению различных материалов из исписанных технических средств, а также сдачу на материалы специализированным организациям для дальнейшей переработки.

Для снижения опасности для окружающей среды, исходящей от компьютерной техники, необходимо:

- применять оборудование, соответствующее санитарным нормам и стандартам экологической безопасности;
- применять расходные материалы с высоким коэффициентом использования и возможностью их полной или частичной регенерации;
- использовать экономные режимы работы оборудования.

4.5. Безопасность в чрезвычайных ситуациях

Чрезвычайная ситуация — это состояние, при котором в результате возникновения источника ЧС на объекте нарушаются нормальные условия жизни и деятельности людей, возникает угроза их жизни и здоровью, наносится ущерб имуществу населения, народному хозяйству и природной среде.

Наиболее распространенными источниками возникновения чрезвычайных ситуаций техногенного характера в офисном помещении являются пожары. Пожары на предприятиях могут возникать в результате повреждения электропроводки и электрооборудования, находящегося под напряжением, нарушение правил эксплуатации электрического оборудования, эксплуатация его в неисправном состоянии, перегрузка электрических сетей, применение неисправных осветительных приборов.

Рабочее помещение программиста является пожароопасным и относится к категории В, так как содержит твердые горючие и трудногорючие вещества и материалы. Согласно классификации пожаров по виду горючего материала, установленной статьей 8 Федерального закона от 22.07.2008 N 123-ФЗ, возможный пожар может относится к классу А (пожары твердых горючих веществ и материалов), так и классу Е (пожары горючих веществ и материалов электроустановок, находящихся под напряжением).

В число превентивных мероприятий могут быть включены мероприятия, направленные на устранение причин, которые могут вызвать пожар. Офисное помещение оснащаются системами автоматической пожарной защиты. Они быстро обнаруживают очаг загорания, автоматически отключают электропитание ПК, локализируют и тушат пожар. В помещении должен быть установлен углекислотный огнетушитель типа ОУ-5 для тушения пожаров. В случае угрозы возникновения ЧС необходимо отключить электропитание, вызвать по телефону пожарную команду, эвакуировать людей из помещения согласно плану эвакуации. Каждый сотрудник обязан соблюдать меры пожарной безопасности на предприятии и следить за их соблюдением другими.

В целях профилактики пожара предлагается не использовать открытые обогревательные приборы в офисном помещении. Также в целях уменьшения вероятности возникновения пожара вследствие короткого замыкания необходимо, чтобы электропроводка была скрытой. Еще одним фактором

возникновения пожара может стать курение в помещении. Поэтому курение в рабочем помещении категорически запрещено.

4.6. Вывод по разделу

В результате работы над данным разделом были выявлены опасные и вредные факторы, являющиеся потенциально опасными для программиста, работающего над разработкой нейросети. Возможные вредные и опасные факторы соответствуют нормативным значениям.

Согласно правилам установок электроустановок, рабочее помещение инженера-программиста относится к помещениям без повышенной опасности поражения электрическим током.

Рабочее помещение оборудовано в соответствии с требованиями электро- и пожарной безопасности. Работа программиста относится к категории тяжести труда Iа и требует 1 группы по электробезопасности.

Рабочее помещение является пожароопасным и относится к категории В, а возможный пожар может относится к классу А, так и классу Е.

В отношении негативного воздействия на окружающую среду, рабочее помещение инженера-программиста относится к IV категории объектов.

Используемое помещение для разработки проекта выпускной квалификационной работы удовлетворяет всем требованиям безопасности и охраны труда.

Заключение

В рамках данной выпускной квалификационной работы была изучена структура и особенности различных генеративно-состязательных нейростей. Были разработаны архитектуры DCGAN и ProGAN, а также их условные модификации. Результаты обучения всех версий генератино-состязательной нейросети представлены в таблице 17.

Таблица 17 – Сравнительная таблица архитектур GAN

№	Наименование архитектуры	Время обучения, ч	Результаты генерации
1	DCGAN	1,5	Генерируемые изображения лишь частично повторяют силуэт грудной клетки и при этом получаются расплывчатыми.
2	ProGAN	9	Генерируемые изображения полностью повторяют структуру грудной клетки, имеют хорошую резкость и частично отображают мелкие детали, такие как трахея и сосуды.
3	ProGAN с MONAI	9	По сравнению с предыдущей версией генерируемые изображения обладают большей контрастностью, что позволяет различать различные ткани человеческого тела.
4	CDCGAN	2	Обладает всеми недостатками оригинальной архитектуры и при этом склонна к переобучению, так как на каждый тип патологии генерируется одно и тоже изображение.
5	CProGAN	11	Мелкие детали на генерируемых изображениях стали более различимыми. Не переобучается и

			генерирует правдоподобные изображения с патологиями дыхательной системы 6 типов.

Также был разработан графический пользовательский интерфейс для взаимодействия с генератором CProGAN для получения КТ-изображений грудной клетки.

Материалы исследовательской работы были представлены на XX Международной научно-практической конференции студентов, аспирантов и молодых ученых «Молодёжь и современные информационные технологии». В результате выступления в секции «Цифровизация, поддержка принятия решений, медицинские ИС» был получен диплом 2 степени, копия которого представлена в приложении А.

Список использованных источников

1. Цаунит, А. Н. Перспективы развития и применения нейронных сетей / А. Н. Цаунит. — Текст : непосредственный // Молодой ученый. — 2021. — № 23 (365). — С. 114-117.
2. Создаем GAN с помощью PyTorch. [Электронный ресурс]. – URL: https://habr.com/ru/company/otus/blog/569858/ (дата обращения 10.05.2023)
3. Unsupervised Representation Learning with Deep Convolutional Generative Adversarial Networks. [Электронный ресурс]. – URL: https://arxiv.org/abs/1511.06434 (дата обращения 15.05.2023)
4. ProGAN: Progressive Growing Generative Adversarial Networks. [Электронный ресурс]. – URL: https://blog.paperspace.com/progan/ (дата обращения 16.05.2023)
5. Conditional GAN (cGAN) in PyTorch and TensorFlow. [Электронный ресурс]. – URL: https://learnopencv.com/conditional-gan-cgan-in-pytorch-and-tensorflow/ (дата обращения 18.05.2023)
6. "Трудовой кодекс Российской Федерации" от 30.12.2001 N 197-ФЗ (ред. от 19.12.2022, с изм. от 11.04.2023) (с изм. и доп., вступ. в силу с 01.03.2023)
7. ГОСТ 12.2.032-78. Система стандартов безопасности труда (ССБТ). Рабочее место при выполнении работ сидя. Общие эргономические требования
8. ГОСТ 12.0.003-2015 Система стандартов безопасности труда (ССБТ). Опасные и вредные производственные факторы. Классификация (с Поправкой)
9. СанПиН 1.2.3685–21 Гигиенические нормативы и требования к обеспечению безопасности и (или) безвредности для человека факторов среды обитания
10. СП 51.13330.2011 Защита от шума. Актуализированная редакция СНиП 23-03-2003

11.ГОСТ 12.1.019-2017 ССБТ. Электробезопасность. Общие требования и номенклатура видов

ПРИЛОЖЕНИЕ А

ТОМСКИЙ ПОЛИТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ

ДИПЛОМ

II степени

ВРУЧАЕТСЯ

КУЗНЕЦОВУ ИЛЬЕ ЕВГЕНЬЕВИЧУ

за доклад, представленный на

XX Международной научно-практической конференции
студентов, аспирантов и молодых ученых
«Молодёжь и современные информационные технологии»

и оцененный в **85** баллов,

«РАЗРАБОТКА ГЕНЕРАТИВНО-СОСТЯЗАТЕЛЬНОЙ НЕЙРОСЕТИ ДЛЯ ГЕНЕРАЦИИ КТ-ИЗОБРАЖЕНИЙ ГРУДНОЙ КЛЕТКИ»

20-22 марта 2023 г.

Проректор по цифровизации,
директор ИШИТР А.С. Фадеев

ТОМСК

Рисунок А.1 – Диплом 2 степени за доклад на конференции МСИТ