СОДЕРЖАНИЕ

СПИСОК СОКРАЩЕНИЙ И УСЛОВНЫХ ОБОЗНАЧЕНИЙ 7

ТЕРМИНЫ И ОПРЕДЕЛЕНИЯ 8

ВВЕДЕНИЕ 9

1 ЛИТЕРАТУРНЫЙ ОБЗОР 13

1.1 Диффузионные модели 13

1.2 Диффузионные модели для генерации изображений 15

1.3 Персонализация в диффузионных моделях 18

2 ТЕОРЕТИЧЕСКИЙ ОБЗОР 23

2.1 Stable Difusion 23

2.2 PhotoMaker 27

3 ПРОЕКТИРОВАНИЕ 31

3.1 Высокоуровневая архитектура системы 31

3.1.1 Обзор компонентов системы 31

3.1.2 Диаграмма развертывания 32

3.2 Bot Handler 33

3.2.1 Функции бота 34

3.2.2 База данных ботов 35

3.3 Generate Image Engine 37

3.4 Redis-сервер 38

3.4.1 Ограничения вычислительных ресурсов при генерации изображений 38

3.4.2 Технические характеристики 39

4 РЕАЛИЗАЦИЯ 41

ЗАКЛЮЧЕНИЕ 45

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ 46

СПИСОК СОКРАЩЕНИЙ И УСЛОВНЫХ ОБОЗНАЧЕНИЙ

ИИ – Искусственный интеллект

VAE – Вариационный автоэнкодер

U-Net – Универсальная сеть

CLIP – Contrastive Language-Image Pre-training

MLP – Многослойный перцептрон

RQ – Redis Queue

GAN – Генеративные состязательные сети

LLM – Большие языковые модели

NCSN – Сети оценки с учетом шума

ТЕРМИНЫ И ОПРЕДЕЛЕНИЯ

Сервис — в микросервисной архитектуре, сервис представляет собой небольшую, независимую компоненту программного обеспечения, которая выполняет определенную функцию. Каждый сервис должен иметь чётко определенную границу, которая отделяет его от других сервисов. Это позволяет разрабатывать, тестировать и развертывать каждый сервис независимо от других компонентов приложения.

Контейнер (англ. container) — это лёгкий и автономный исполняемый пакет, который содержит все компоненты, необходимые для запуска приложения, включая код приложения, библиотеки и другие зависимости. Контейнеры обычно работают в изолированной среде, что обеспечивает надёжность и безопасность приложения.

ИИ (искусственный интеллект) — это область компьютерной науки, которая занимается созданием интеллектуальных систем, способных выполнять задачи, которые обычно требуют наличия человеческого интеллекта. ИИ может включать в себя различные технологии и методы, такие как машинное обучение, глубокое обучение, нейронные сети, обработку естественного языка, компьютерное зрение и другие.

Фреймворк (англ. framework) — набор соглашений, правил и шаблонов, который предоставляет разработчикам компоненты для решения типичных задач. Фреймворк определяет структуру приложения, предоставляет набор инструментов для работы с базами данных, управления пользовательским интерфейсом.

SQLite — это система управления базами данных, которая используется для хранения важной информации, такой как пути к загруженным изображениям, пользовательские настройки и данные, связанные с подсказками и рейтингами. SQLite предлагает простую настройку и мощные возможности обработки данных без затрат на более сложные системы баз данных.

RQ (Redis Queue) — это библиотека Python, которая упрощает постановку заданий в очередь и их обработку в фоновом режиме с помощью рабочих процессов. Она использует Redis для хранения заданий и их статусов, что делает его надежным выбором для асинхронного управления задачами.

Микросервисная архитектура (англ. microservice architecture) — подход к созданию приложений путем разбиения их на небольшие независимые компоненты, называемые микросервисами. Каждый микросервис отвечает за выполнение определенной функции и может разрабатываться, тестироваться и развертываться независимо от других, что обеспечивает гибкость и упрощает масштабирование приложений.

ВВЕДЕНИЕ

Значительные достижения в области искусственного интеллекта (ИИ) за последние десятилетия привели к кардинальным изменениям в различных глобальных секторах. От автоматизации рутинных задач до разработки новаторских решений — влияние ИИ распространяется на многие аспекты жизни современного общества. В области создания цифрового контента, в частности, создания изображений, искусственный интеллект совершил прорыв, который десять лет назад казался научной фантастикой. Такие методы, как нейронные сети, превратились в сложные инструменты, способные создавать детализированные, реалистичные и креативные изображения.

Рисунок 1 — Работа Джейсона Аллена «Théâtre D’opéra Spatial», созданная с помощью искусственного интеллекта

В 2022 году работа Джейсона М. Аллена «Théâtre D’opéra Spatial» (рисунок 1) заняла первое место на художественном конкурсе Ярмарки штата Колорадо, что стало поворотным моментом в эволюции художественного творчества. Это произведение было создано с использованием Midjourney [2], программы искусственного интеллекта, которая использует диффузионную модель для создания изображений, бросая вызов традиционным методам создания произведений искусства. Работа Аллена была воспринята неоднозначно: хотя она очаровала и зрителей, и судей, она также вызвала споры среди художников, которые считали использование им искусственного интеллекта

подрывающим художественную целостность. Тем не менее, это мероприятие подчеркнуло потенциал ИИ в создании изображений, которые не только визуально реалистичны, но и наполнены инновационными художественными особенностями, тем самым раздвигая границы цифрового творчества и расширяя возможности человеческого воображения.

Появление Midjourney и других достижений в области ИИ лишь поверхностно затрагивает возможности диффузионных моделей. Эти генеративные модели получили известность благодаря созданию высококачественных изображений, которые часто неотличимы от реальных фотографий. Этот метод значительно отличается от своих предшественников, таких как генеративные состязательные сети (GAN) [1], которые генерируют новые образцы данных посредством конкурентного процесса между двумя нейронными сетями.

Рисунок 2 — Изображения, сгенерированные с использованием модели Stable Diffusion, полученные из источника Stability.AI

Модели Stable Diffusion [3], основанные на диффузионных методах с открытым исходным кодом, быстро привлекли значительное внимание пользователей сообщества благодаря своей доступности для академических и практических целей. Они достигают почти идеального реализма и обладают способностью к созданию разнообразных стилевых выводов, как показано на различных примерах на рисунке 2. Их появление является своевременным, поскольку они отвечают растущему спросу на создание цифрового контента — важнейшего компонента в современном секторе коммуникаций, маркетинга и развлечений.

Платформы социальных сетей, насчитывающие миллиарды пользователей по всему миру, находятся на переднем крае этой контент-революции, выступая одновременно и площадками, и рынками сбыта, где индивиды и бренды демонстрируют свои сюжеты, продукты и услуги. Способность привлечь и удержать внимание аудитории среди бесконечного потока цифрового шума сейчас более важна, чем когда-либо. Создатели контента должны постоянно внедрять инновации, чтобы выделяться, что приводит к необходимости в способах, которые могут создавать не только высококачественный контент, но и высоко персонализированные и контекстуально релевантные результаты. Именно здесь в игру вступают технологии создания изображений с использованием ИИ, предлагающие возможности, которые когда-то были доступны только тем, кто обладает обширными навыками графического дизайна. Одним из подходов к достижению этой цели является использование PhotoMaker [4], конвейера, предназначенного для помощи диффузионным моделям в сохранении физических характеристик людей, запечатленных на эталонном изображении. Его возможности показаны на рисунке 3.

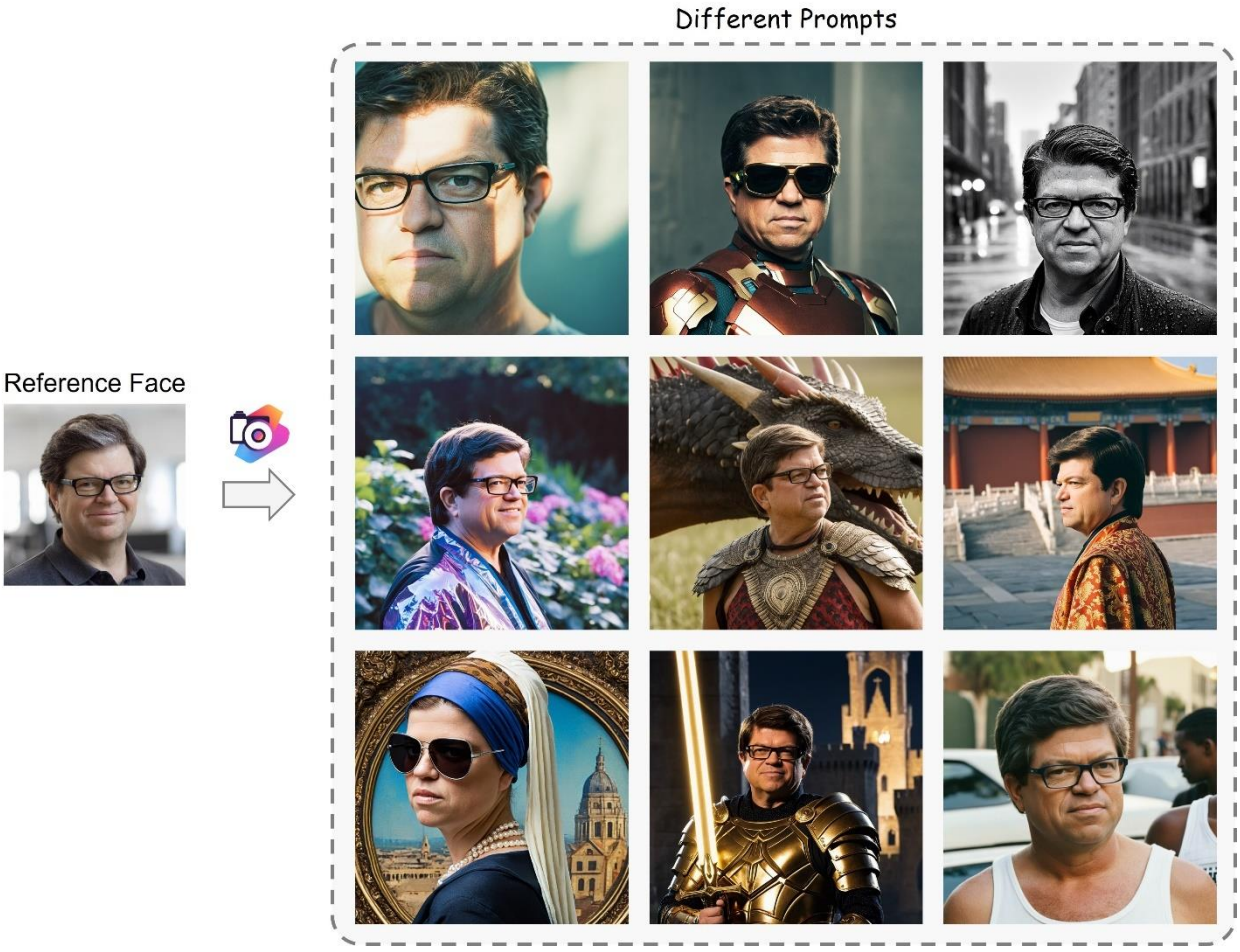

Рисунок 3 — Изображения, сгенерированные с использованием модели PhotoMaker

Несмотря на значительные достижения в области диффузионных моделей для создания изображений, существует несколько проблем, препятствующих их широкому внедрению среди людей, не являющихся инженерами. Эти проблемы в первую очередь связаны со сложностью настройки и использования этих моделей. Как правило, развертывание этих моделей требует глубокого понимания структур машинного обучения, значительных вычислительных ресурсов, а также способности манипулировать и настраивать многочисленные параметры, влияющие на процесс генерации. Для обычного человека без технического образования эти предпосылки представляют собой существенный барьер. Сложность установки и настройки таких моделей требует не только специальных знаний, но и доступа к оборудованию, которое зачастую находится за пределами досягаемости обычных пользователей. Эта сложность резко ограничивает практическую полезность диффузионных моделей для личного или повседневного использования.

Следовательно, целью данной дипломной работы является разработка Телеграм-бота для стилизации изображений с использованием диффузионных моделей. Этот бот позволяет пользователям быстро и легко генерировать изображения по их запросам, сохраняя при этом характеристики их эталонных изображений.

Для достижения результатов исследования необходимо реализовать следующие этапы:

1) Проведение анализа предметной области и обзора существующих аналогов. Выбор наиболее подходящих методов для решения задачи;
2) Реализация модели диффузии и метода PhotoMaker с использованием библиотеки PyTorch и пакета Diffusers;
3) Разработка веб-приложения с пользовательским интерфейсом в виде бота в Телеграм, используя API Телеграм Bot и базу данных на SQLite;
4) Проведение тестирования и оптимизации производительности приложения;
5) Выгрузка исходного кода приложения в открытый репозиторий.

1 ЛИТЕРАТУРНЫЙ ОБЗОР

1.1 Диффузионные модели

Первоначальное внедрение диффузионных моделей в сообщество глубокого обучения приписывается Sohl-Dickstein и соавторам в их статье 2015 года под названием «Deep Unsupervised Learning using Nonequilibrium Thermodynamics» [5]. В этой статье обсуждался новый подход, основанный на использовании неравновесной термодинамики для систематического разрушения и последующего восстановления структуры в распределениях данных. Однако он не сразу привлек широкое внимание в исследовательском сообществе.

Последующий всплеск академического интереса, особенно к диффузионным моделям, связан с исследованием Сонга и соавторов, опубликованным в 2019 году под названием «Generative Modeling by Estimating Gradients of the Data Distribution» [6]. В этом документе представлена концепция «сети оценки с учетом шума» (NCSN), которая предоставляет новую методологию для обучения этих моделей путем прямой оценки градиентов распределения данных, что является значительным усовершенствованием в этой области.

Заметным поворотным моментом стала публикация в 2020 году «Denoising Diffusion Probabilistic Models» Хо и соавторов, обычно сокращенно обозначаемых как DDPM [7]. Это исследование обеспечило дальнейшие достижения, предложив усовершенствованные методы обучения диффузионным моделям, что привело к существенному прогрессу в разработке и внедрении этих генеративных моделей в различных приложениях. Эта публикация ознаменовала поворотный момент, после которого произошло заметное ускорение в исследованиях, разработке и совершенствовании основанных на диффузии генеративных моделей, что привело к существенным достижениям за удивительно короткий период.

После этих основополагающих работ многочисленные исследования расширили понимание и возможности диффузионных моделей, расширив границы их применения в различных областях, включая синтез изображений, молекулярный дизайн и, в последнее время, компьютерное зрение.

В области компьютерного зрения диффузионные модели были адаптированы для решения целого ряда задач — от генерации изображений до понимания сложных сцен. Эти модели продемонстрировали замечательную производительность при создании фотореалистичных изображений, превосходя традиционные генеративные модели, такие как генеративные состязательные сети (GAN), по качеству и разнообразию изображений.

Их успех в области компьютерного зрения можно объяснить их способностью моделировать распределение данных с высокой точностью.

Диффузионные модели являются генеративными моделями, что означает, что они используются для генерации данных, аналогичных тем, на основе которых они были обучены. По сути, диффузионные модели работают, уничтожая обучающие данные путем последовательного добавления гауссовского шума, а затем обучаясь восстанавливать данные, обращая вспять этот процесс создания шума. После обучения мы можем использовать диффузионную модель для генерации данных, просто пропуская случайно отобранные шумы через изученный процесс шумоподавления.

Процесс прямой диффузии начинается с выборки из базового, обычно гауссовского, распределения. Эта начальная простая выборка подвергается серии обратимых, постепенных модификаций, где каждый шаг вносит контролируемую степень сложности. Сложность постепенно нарастает, часто визуализируясь как добавление структурированного шума. Такое распространение исходных данных посредством последовательных преобразований позволяет модели улавливать и воспроизводить сложные закономерности и детали, присущие целевому распределению. Конечная цель процесса прямого диффузии состоит в том, чтобы превратить эти простые исходные данные в образцы, которые точно имитируют желаемое распределение сложных данных, демонстрируя, как, начиная с минимальной информации, можно получить богатые, подробные результаты. Процесс прямой диффузии представлен на рисунке 4.

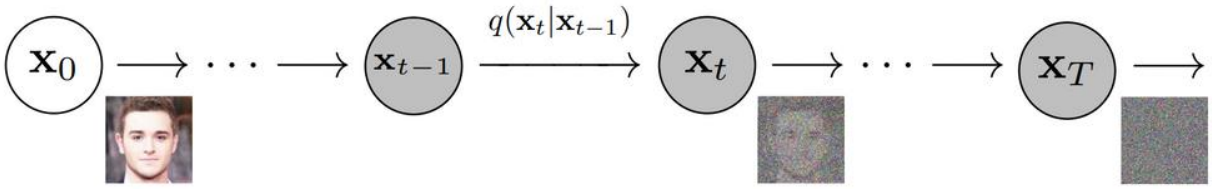

Рисунок 4 — Процесс прямой диффузии

Цель обучения диффузионной модели — изучить процесс обратной диффузии. На этом этапе диффузионные модели отличаются от других генеративных моделей, таких как GAN. Процесс обратной диффузии включает распознавание специфических шумовых паттернов, возникающих на каждом этапе, и соответствующее устранение шума в данных. Это непростой процесс, а скорее сложная реконструкция. Преобразование некоторого количества случайного шума в осмысленное изображение — сложная задача. Модель использует полученные знания для прогнозирования шума на каждом этапе, а затем тщательно устраняет его. Процесс обратной диффузии представлен на рисунке 5.

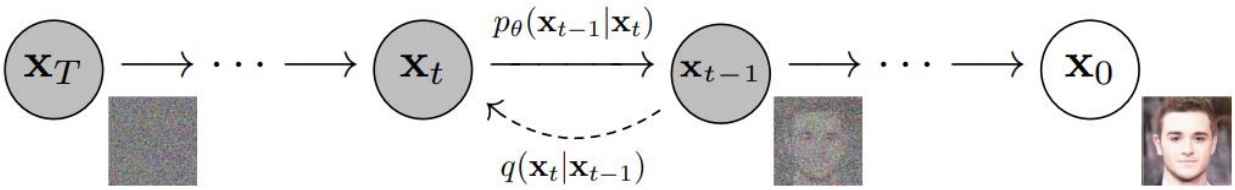

Рисунок 5 — Процесс обратной диффузии

1.2 Диффузионные модели для генерации изображений

Модели диффузии показали себя многообещающими в различных приложениях благодаря их способности моделировать сложные распределения данных и генерировать высококачественные образцы. В сфере компьютерного зрения генерация изображений, особенно с помощью моделей преобразования текста в изображение, значительно способствовала распространению диффузионных моделей. Яркие примеры включают DALL-E 2 от OpenAI [8], Imagen от Google Brain [9], Stable Diffusion от Stability AI [3] и Midjourney [2]. Эти модели используют расширенные возможности диффузионных моделей для преобразования текстовых описаний в подробные визуальные результаты, демонстрируя впечатляющее сближение понимания языка и визуального творчества. Обычно они объединяют языковую модель, которая преобразует входной текст в скрытое представление, и генеративную модель изображения, в частности диффузионную модель, которая затем генерирует изображение на основе этого представления.

DALL-E — DALL-E от OpenAI, представленный в январе 2021 года, ознаменовал важную веху как одна из первых моделей преобразования текста в изображение, захвативших воображение публики. Он использует систему на основе трансформера, способную преобразовывать текстовые описания в уникальные изображения. Его преемник, DALL-E 2, выпущенный в апреле 2022 года, расширил эти возможности, создавая более сложные и реалистичные изображения. DALL-E 2 использует контрастные модели, такие как CLIP, для изучения надежных представлений изображений, которые отражают семантику и стиль. Он имеет двухэтапную модель, состоящую из предыдущего этапа, который генерирует встраивание изображения CLIP на основе заданной текстовой подписи, и этапа декодера. Декодеры модели используют диффузию. Эти модели создают вариации изображения, которые сохраняют его семантику и стиль, изменяя при этом несущественные детали. Кроме того, интеграция CLIP в DALL-E 2 обеспечивает возможности нулевого выстрела, позволяя модели выполнять манипуляции с изображениями на основе текстовых описаний без непосредственного контроля. Это делает DALL-E 2 универсальным в создании изображений, которые точно соответствуют

приведенным описаниям, что повышает его полезность в различных приложениях. Это пример изображений, созданных с помощью модели DALL-E, DALL-E 2 (рисунок 6).

Рисунок 6 — Примеры изображений, созданных с помощью DALL-E, DALL-E 2

Imagen — Вслед за Dall-E 2, всего через месяц, в апреле 2022 года, Google выпустила свою систему алгоритмов генерации изображений на основе диффузии под названием Imagen (для генерации изображений). Модель построена на двух ключевых компонентах: больших предварительно обученных кодировщиках замороженного текста и диффузионных моделях. Используя возможности языковых моделей на основе трансформеров, таких как T5, Imagen демонстрирует замечательное мастерство в понимании текстовых описаний и эффективном их кодировании для синтеза изображений. Важным выводом исследования является важность масштабирования размера предварительно обученного кодировщика текста для задачи генерации изображений. Увеличение размера языковой модели в Imagen существенно положительно влияет как на точность сгенерированных образцов, так и на соответствие изображений и соответствующих текстовых описаний. Это подчеркивает эффективность больших языковых моделей (LLM) при кодировании осмысленных представлений текста, что существенно влияет на качество генерируемых изображений. В настоящее время Google Imagen недоступен для широкой публики и доступен только по приглашению. Это пример изображений, созданных с помощью модели Imagen (рисунок 7).

Рисунок 7 — Примеры изображений, созданных с помощью Imagen

Midjourney — это еще одна модель генерации изображений на основе диффузии, разработанная одноименной компанией. Midjourney стал доступен массам в марте 2020 года. Он быстро стал популярным благодаря своему характерному выразительному стилю, став доступным раньше своих современников, таких как DALL-E и Stable Diffusion. В настоящее время Midjourney работает с закрытым исходным кодом и не описан ни в одной опубликованной статье. Тем не менее, пользователи могут изучить возможности создания изображений через официального бота Discord. Это пример изображений, созданных с помощью модели Midjourney (рисунок 8).

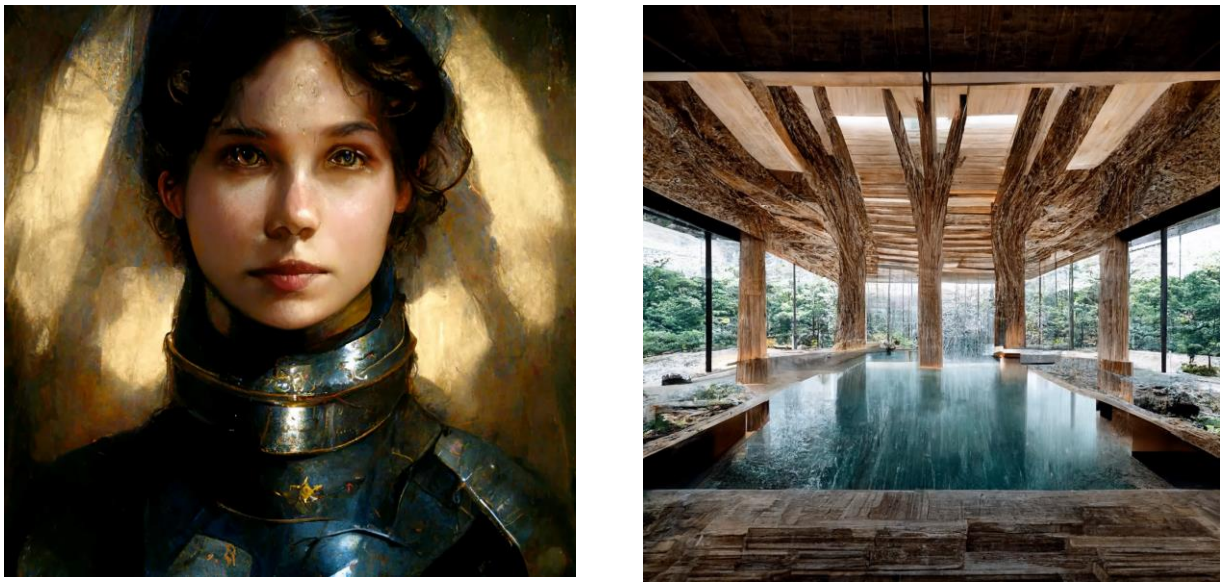

Рисунок 8 — Примеры изображений, созданных с помощью Midjourney

Stable Diffusion — Модель Stable Diffusion, созданная StabilityAI, основана на фундаментальной работе Ромбаха и др. Это модель генерации изображений на основе диффузии, исходный код которой полностью открыт. Его код и вес модели были

опубликованы публично, и он может работать на большинстве потребительских устройств, оснащенных графическим процессором. По этой причине сообщество открытого исходного кода было очень активным с момента его выпуска. За короткий промежуток времени сообщество выпустило несколько моделей Stable Diffusion с открытым исходным кодом, точно настроенных на различных художественных стилизованных наборах данных. Можно свободно использовать эти модели и создавать новые изображения с использованием этих стилей. Это может быть что угодно: от японского аниме и футуристических роботов до миров киберпанка. Вот несколько примеров из моделей Stable Diffusion (рисунок 9).

Рисунок 9 — Примеры изображений, созданных с помощью Stable Diffusion

Учитывая ограничения доступности и финансовые последствия, связанные с облачными моделями, такими как DALL-E 2, Imagen и Midjourney, Stable Diffusion предлагает практическую альтернативу для разработчиков. Его модель с открытым исходным кодом гарантирует, что его можно адаптировать и эффективно развернуть, что делает его предпочтительным выбором для разработки бота Телеграм, нацеленного на персонализацию генерации изображений. Этот бот использует возможности Stable Diffusion для обработки запросов и создания изображений непосредственно в Телеграм, тем самым снижая барьеры для доступа к усовершенствованным изображениям, созданным с помощью искусственного интеллекта.

1.3 Персонализация в диффузионных моделях

На фоне развития мощных моделей генерации диффузных изображений были разработаны многочисленные алгоритмы на основе диффузии, чтобы удовлетворить растущий спрос пользователей на высококачественные, настраиваемые выходные данные.

Используя надежные генеративные возможности этих моделей, исследователи все больше внимания уделяют изучению методов персонализированной генерации. Эти усилия направлены на сохранение уникальных характеристик личности пользователей из их входных фотографий, гарантируя, что персонализированный контент останется верным исходным чертам, изображенным на изображениях. Это открывает потрясающие возможности для воссоздания контента. Благодаря индивидуальному характеру это приложение получило растущий общественный спрос, и многие исследователи и компании начали работать над этой конкретной задачей генерации.

Для достижения этой цели сначала применялись GAN для создания комбинации двух концепций в разных эталонных изображениях, например, наложение солнцезащитных очков на лицо цели. Однако этот подход демонстрировал неестественные результаты копирования и вставки, и процесс не поддерживал указания по условиям, такие как текстовая подсказка, что значительно ограничивало практическое использование. В последнее время модели диффузии сделали возможным гораздо более простое и осуществимое создание текстового контента, способствуя расцвету персонализации текстового контента.

В настоящее время основные методы персонализированного синтеза, основанные на диффузии, в основном делятся на два типа в зависимости от их стратегий обучения: методы, основанные на настройке, и методы без настройки.

Основной подход, основанный на настройке, в первую очередь представленный Textual Inversion [10] и DreamBooth [11], представленный в августе 2022 года, включает в себя точную настройку модели диффузии с использованием нескольких изображений конкретного объекта для установления уникального идентификатора класса для этого объекта. Этот процесс позволяет точно воссоздать объект на сгенерированных изображениях. Эти методы относятся к категории основанных на настройке, поскольку они корректируют определенные параметры модели для каждого нового введенного объекта.

Несмотря на их способность создавать высококачественные изображения, сохраняющие целостность лица субъекта, эти методы имеют заметные ограничения. Для эффективной настройки модели требуется значительное количество изображений одного и того же человека. Более того, процесс оптимизации изменяет исходные параметры модели, что приводит к высоким вычислительным требованиям и увеличению времени обработки. Эти ограничения затрудняют широкомасштабное применение этих методов в коммерческих условиях. Их производительность и визуальные результаты таких методов, основанных на настройке, можно наблюдать на рисунке 10.

Рисунок 10 — Примеры изображений, созданных с помощью методов DreamBooth

Чтобы решить эти проблемы, последние разработки в области методов без настройки направлены на упрощение и ускорение процесса создания индивидуальных изображений. Эти подходы используют обширные данные, специфичные для предметной области, и обучают кодировщик или гиперсеть преобразовывать входные идентификационные изображения во встраивания или веса LoRA [12] в модели. После обучения пользователи могут просто ввести идентификационное изображение для настройки, что позволяет генерировать персонализированное изображение за считанные секунды на этапе вывода.

FastComposer — FastComposer [13], запущенный в мае 2023 года исследователями из Массачусетского технологического института, представляет собой сложную модель, разработанную для обеспечения эффективного, персонализированного, многосубъектного преобразования текста в изображение без тщательной настройки. В этой модели используются механизмы локализованного внимания для нацеливания на определенные области изображения, что позволяет точно манипулировать и создавать несколько объектов в одной композиции. В отличие от традиционных моделей, которые могут применять единый подход ко всему изображению, FastComposer использует локализованное внимание для независимого управления различными областями изображения. Эта способность к быстрой адаптации обеспечивает точный контроль над несколькими объектами в рамках одной генеративной задачи, что делает FastComposer особенно ценным для задач, связанных с персонализацией изображений. Рисунок 11 иллюстрирует возможности этой модели.

Рисунок 11 — Примеры изображений, созданных с помощью FastComposer

IP-Adapter — Другой подход, IP-Adapter [14], представленный в августе 2023 года, направлен на улучшение реакции моделей диффузии текста в изображения на текстовые подсказки. Он улучшает эти модели, более точно согласовывая сгенерированные изображения с текстовыми описаниями пользователей. IP-Adapter обучает легкий, несвязанный модуль перекрестного внимания, в котором функции изображения и текста обрабатываются отдельно наряду с функциями запроса. Когда пользователь предоставляет текстовую подсказку, модуль адаптера уточняет этот ввод, чтобы углубить его семантическую интерпретацию. Это уточнение корректирует представление подсказки для лучшей синхронизации с генеративными возможностями модели. На рисунке 12 показано, чего может достичь IP-Adapter, включая эталонное изображение, текстовую подсказку и результирующее изображение, созданное IP-Adapter.

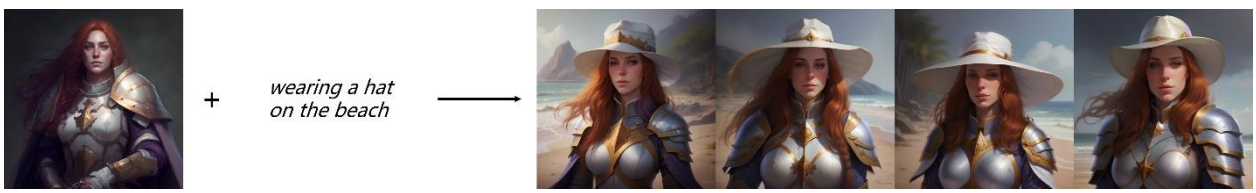

Рисунок 12 — Примеры изображений, созданных с помощью IP-Adapter

Однако результатам, полученным с помощью этих методов, не хватает как точности идентификации, так и разнообразия по сравнению с DreamBooth. Это связано с тем, что во

время обучения и целевое изображение, и изображение входного идентификатора выбираются из одного и того же изображения. В результате обученная модель имеет тенденцию запоминать характеристики, не связанные с идентификатором, такие как выражения и точки зрения, что приводит к ухудшению возможностей редактирования. Кроме того, использование только одного изображения идентификатора для настройки затрудняет точное различение характеристик идентификатора на основе ее внутренних знаний, что приводит к неудовлетворительной точности идентификатора.

В декабре 2023 года в статье под названием «**PhotoMaker**» [15] был представлен инновационный и в то же время простой фидфорвардный фреймворк для кастомизированной генерации изображений людей, способный обрабатывать несколько входных идентификаторов. Этот метод акцентирует внимание на создании портретов людей, интегрируя оба ранее упомянутых технических подхода. В частности, он не только создает персонализированную базу данных, ориентированную на идентификаторы, но и получает встраивание, представляющее идентификатор человека в семантическом пространстве. В отличие от предыдущих методов без настройки, PhotoMaker извлекает составное встраивание идентификатора из нескольких идентификаторов. Этот подход улучшает представление идентификатора, при этом сохраняя высокую эффективность, характерную для предыдущих методов без настройки. Рисунок 13 демонстрирует возможности PhotoMaker сохранять характеристики исходного изображения на сгенерированном изображении.

Рисунок 13 — Примеры изображений, созданных с помощью PhotoMaker

2 ТЕОРЕТИЧЕСКИЙ ОБЗОР

2.1 Stable Difusion

Stable Diffusion [3] является результатом проекта «Latent Diffusion» [16], совместной работы Университета Людвига-Максимилиана в Мюнхене и Гейдельбергского университета. Лицензией и дальнейшей разработкой этой модели занималась группа CompVis в Университете Людвига-Максимилиана под руководством Патрика Эссера из Runway и Робина Ромбаха из CompVis, которые сыграли решающую роль в создании структуры модели скрытой диффузии, лежащей в основе Stable Diffusion.

Модель Latent Diffusion отличается тем, что работает в сжатом скрытом пространстве, а не непосредственно в пространстве пикселей. Это ключевое различие между стандартной Diffusion и моделями Latent Diffusion: модель Latent Diffusion обучается генерировать скрытые (сжатые) представления изображений. Этот подход значительно снижает размерность данных, сокращая требования к вычислениям и памяти. Эта методология была подробно изложена в исследовательской работе «High-Resolution Image Synthesis with Latent Diffusion Models» исследовательской группы ИИ из Университета Людвига-Максимилиана [16]. Архитектура этой модели показана на рисунке 14.

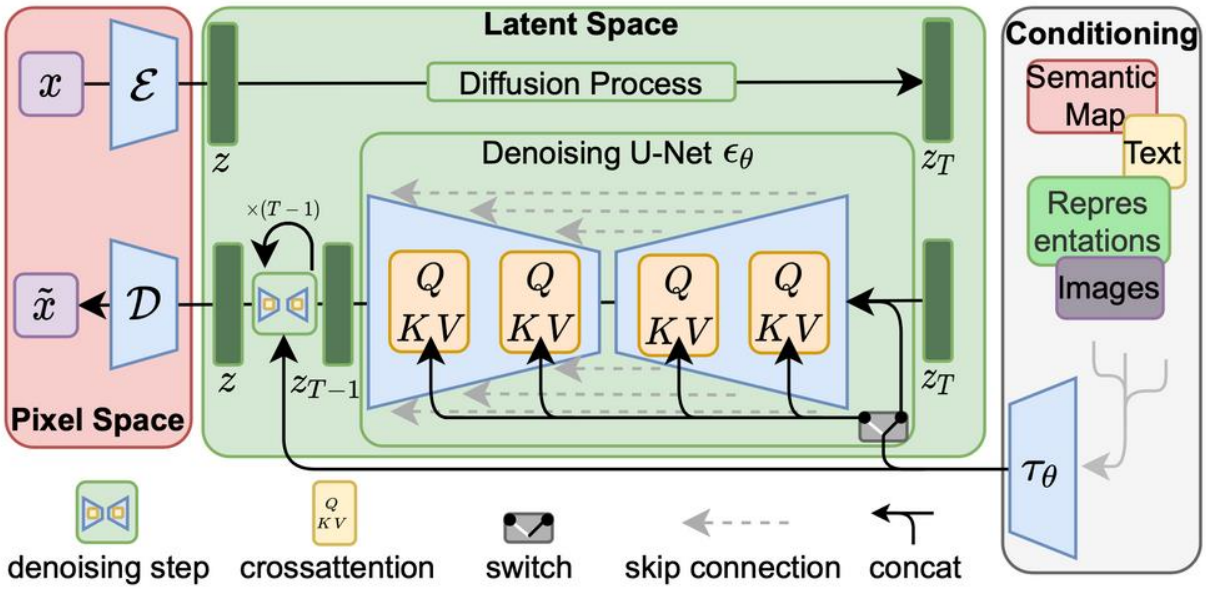

Рисунок 14 — Архитектура модели «Latent Diffusion»

Модель Stable Diffusion, выпущенная в 2022 году, стала продуктом совместных усилий CompVis Group и Runway при поддержке вычислительных ресурсов, предоставленных Stability. Он обучался на наборе данных LAION Aesthetics, тщательно отобранном из обширного набора данных LAION 5B, который включает 120 миллионов пар изображение-текст из почти 6 миллиардов доступных.

Stable Diffusion построен на трех основных компонентах: вариационном автоэнкодере (VAE), архитектуре U-Net и дополнительном текстовом кодировщике, обычно текстовом кодировщике CLIP.

Вариационный автоэнкодер (VAE) — этот компонент включает в себя как кодер, так и декодер. Кодер преобразует изображение в скрытое представление уменьшенного размера, которое служит входными данными для U-Net. И наоборот, декодер восстанавливает изображение из этого сжатого формата. На этапе обучения модели этот кодер фиксирует скрытые представления процесса прямой диффузии, которые постепенно вводят шум. Напротив, во время вывода модель в основном использует декодер для восстановления изображений из очищенных скрытых представлений.

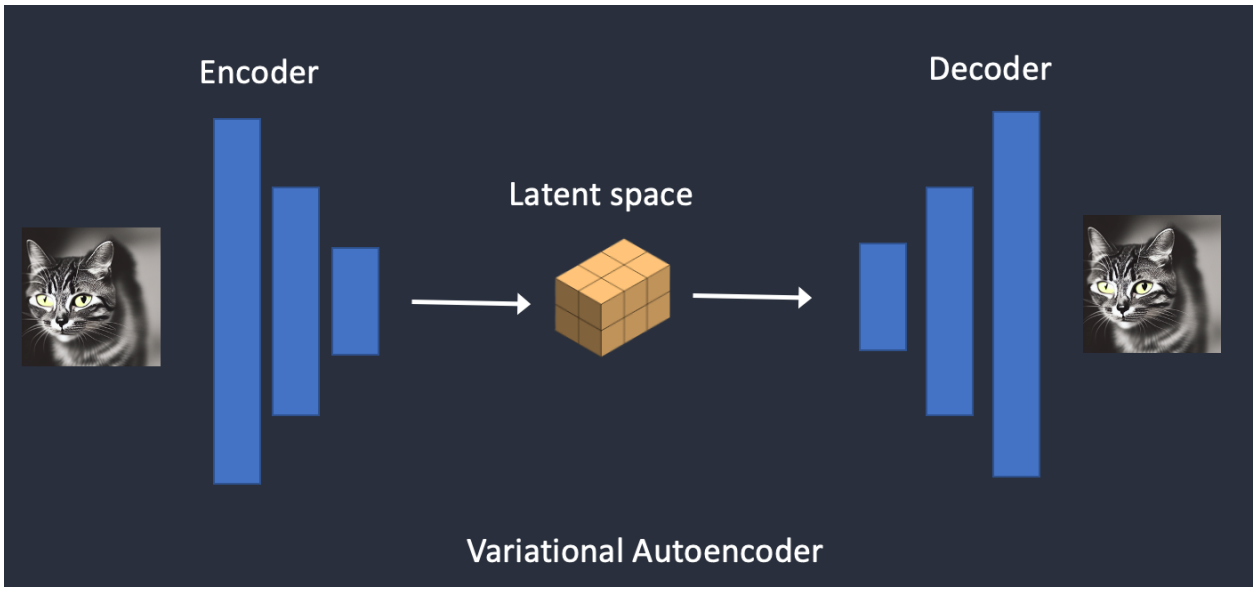

Рисунок 15 — Архитектура VAE

Платформа U-Net (рисунок 16) является неотъемлемой частью модели и состоит из блоков ResNet как в сегментах кодера, так и в декодере. Кодер уменьшает изображение до более низкого разрешения, а декодер реконструирует его в выходные данные с высоким разрешением. Короткие соединения между соответствующими блоками ResNet в кодере и декодере гарантируют сохранение важной информации в процессе уменьшения масштаба. Примечательно, что U-Net в Stable Diffusion включает в себя механизмы перекрестного внимания, которые позволяют ему обуславливать выходные данные встраиванием текста, способствуя более точному переводу текста в изображение.

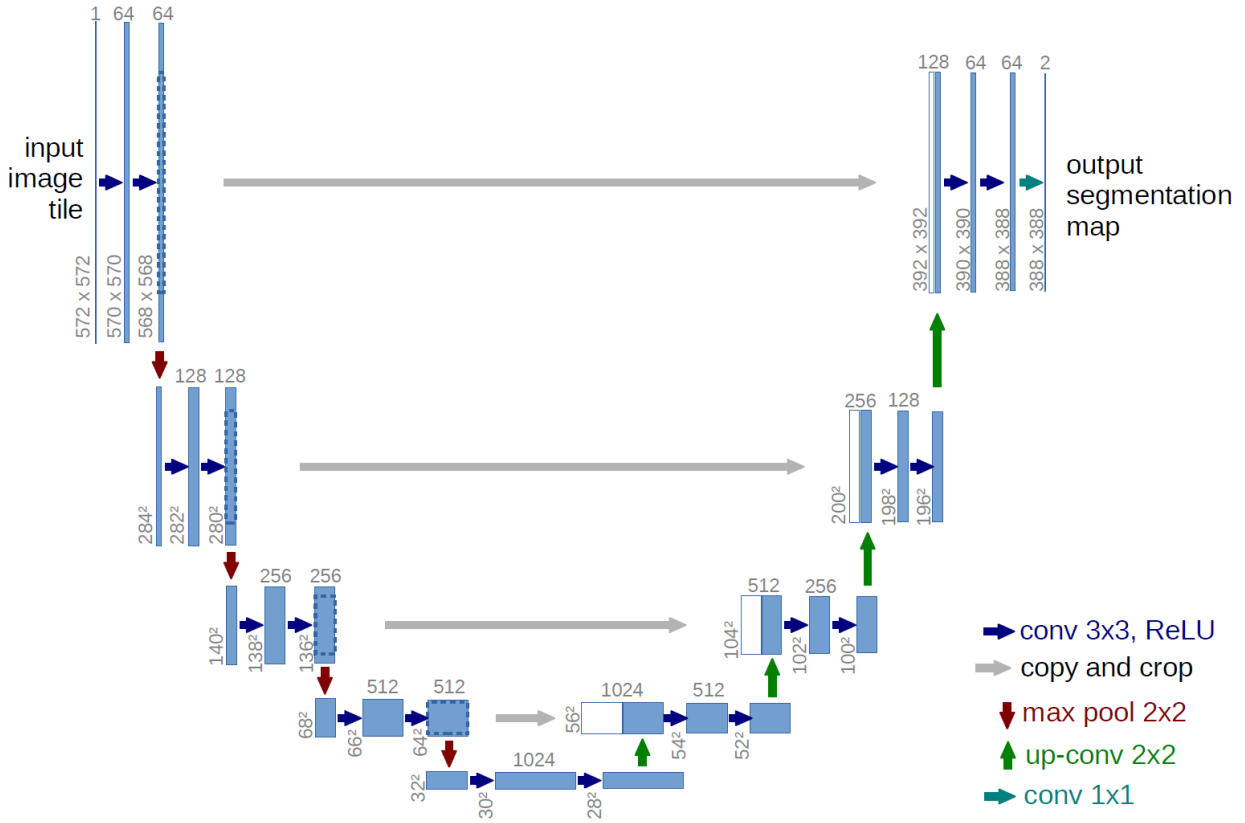

Рисунок 16 — Архитектура U-Net

Text Encoder преобразует текстовые подсказки в формат, понятный U-Net, используя архитектуру на основе преобразователей для сопоставления входных токенов со скрытыми встраиваниями текста. Следуя стратегии, используемой Imagen, Stable Diffusion использует предварительно обученный кодировщик текста CLIP, который не подвергается дальнейшему обучению во время обучения модели диффузии.

Чтобы проиллюстрировать полный рабочий процесс модели стабильной диффузии во время вывода, на Рисунке 17 представлено подробное схематическое изображение последовательных шагов.

Механизм вывода Stable Diffusion начинается с использования скрытого семени и текстовой подсказки в качестве входных данных. Latent seed инициализирует процесс, генерируя случайные представления скрытых изображений, каждое из которых имеет размер 64×64. Одновременно текстовое приглашение преобразуется с помощью текстового кодировщика CLIP, в результате чего получается встраивание текста размером 77×768. Эти внедрения управляют процессом создания изображения, обеспечивая контекстуальную релевантность синтезируемым функциям изображения.

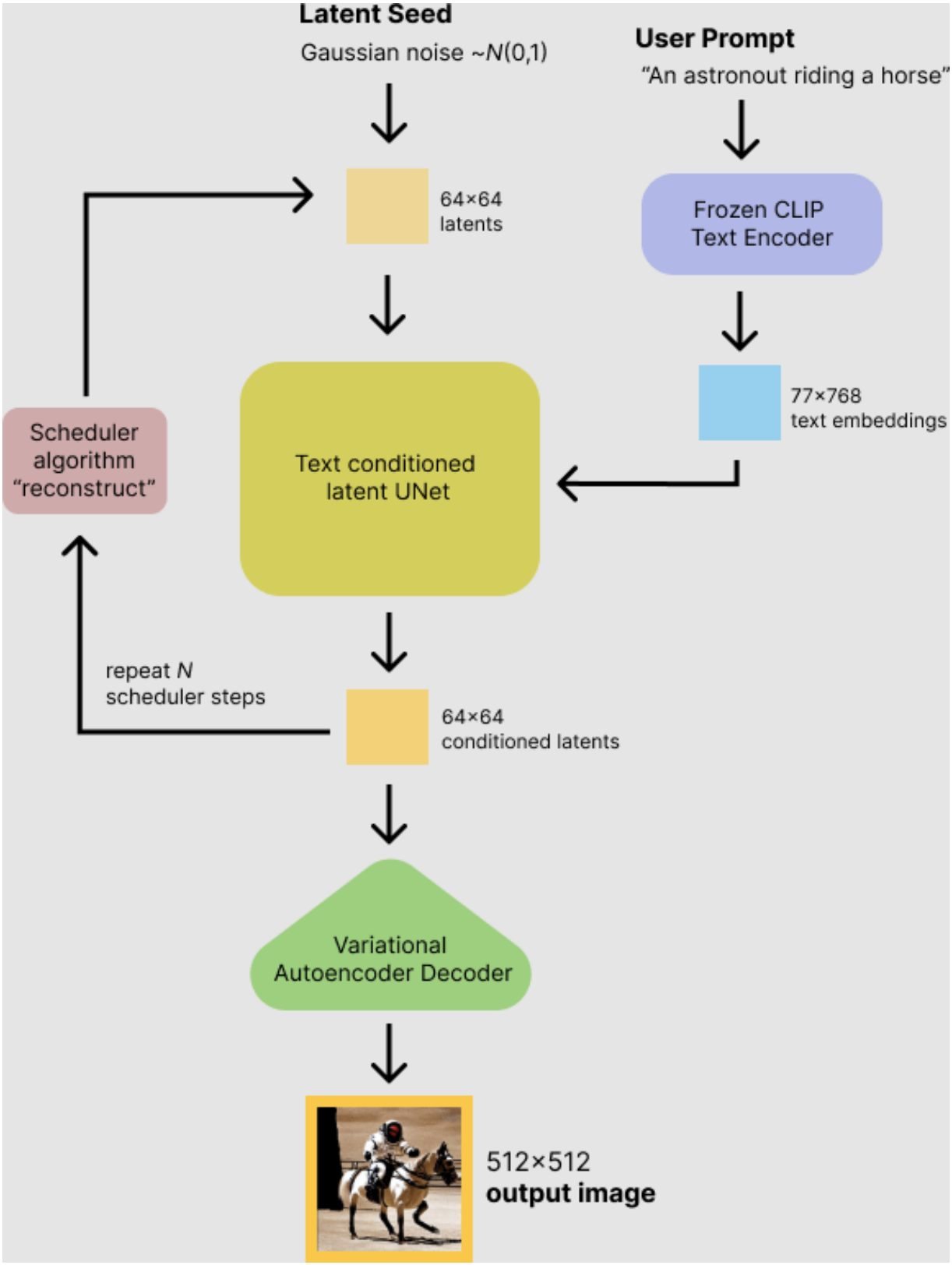

Рисунок 17 — Последовательная блок-схема механизма вывода «Stable Diffusion»

Ядро процесса вывода обрабатывается U-Net, который выполняет итеративное шумоподавление этих скрытых представлений изображений, при этом они обусловлены встраиванием текста. На этом этапе U-Net вычисляет остаточный шум, который является неотъемлемой частью уточнения представления скрытого изображения. Эти вычисления

управляются алгоритмом планировщика, который играет решающую роль на этапах шумоподавления. Доступны различные алгоритмы планировщика, каждый из которых имеет свои преимущества и недостатки, в первую очередь влияющие на эффективность и качество процесса шумоподавления.

Этот цикл шумоподавления повторяется примерно 50 раз, постепенно повышая четкость и детализацию представления скрытого изображения. С каждой итерацией U-Net постепенно снижает уровень шума, обеспечивая более четкое и точное представление изображения.

По завершении этих циклов на последнем этапе задействуется компонент декодера вариационного автоэнкодера. Декодер берет уточненное представление скрытого изображения и реконструирует его в окончательное изображение с высоким разрешением. Этот шаг имеет решающее значение, поскольку он переводит обработанные скрытые представления обратно в визуальный формат, завершаясь созданием подробного и контекстуально точного изображения на основе исходной текстовой подсказки.

Версия SD XL использует ту же архитектуру, за исключением большего размера: более крупная магистральная сеть U-Net, больший контекст перекрестного внимания, два текстовых кодировщика вместо одного и обучение на нескольких соотношениях сторон (а не только на квадратном соотношении сторон, как в предыдущих версиях).

2.2 PhotoMaker

В сфере цифровых изображений создание высокореалистичных фотографий человека всегда представляло серьезные проблемы, в первую очередь из-за сложностей, связанных с достоверным захватом и воспроизведением человеческих черт. В документе под названием «PhotoMaker» [15] эти проблемы решаются с помощью инновационного подхода. В этом исследовании представлена система PhotoMaker, в которой используется метод, известный как «Stacked ID Embedding». Этот метод объединяет несколько уровней внедрения идентификации для создания высокореалистичных и настраиваемых изображений человека.

В этом методе особое внимание уделяется созданию человеческих портретов путем объединения обоих ранее упомянутых технических подходов. В частности, он не только создает набор данных персонализации, ориентированный на идентификатор, но также получает встраивание, представляющее идентификатор человека в семантическом пространстве. В отличие от более ранних методов, не требующих настройки, PhotoMaker извлекает составной идентификатор из нескольких изображений идентификатора. Этот

подход улучшает представление идентификатора, сохраняя при этом высокую эффективность, характерную для предыдущих методов без настройки.

Учитывая набор изображений идентификаторов, которые необходимо настроить, цель PhotoMaker — создать новое фотореалистичное изображение человека, которое сохраняет характеристики входных идентификаторов, изменяя при этом содержимое или атрибуты сгенерированного идентификатора на основе текстовой подсказки. Подобно DreamBooth, PhotoMaker использует несколько идентификационных изображений для настройки, но сохраняет эффективность методов без настройки, обеспечивая настройку за один проход. Это обеспечивает высокую точность идентификации и управляемость текста. Кроме того, PhotoMaker может смешивать несколько входных идентификаторов, создавая изображение, которое точно сохраняет характеристики различных идентификаторов, тем самым расширяя возможности его применения. Эти возможности во многом обусловлены предлагаемым простым, но эффективным внедрением многоуровневых идентификаторов, которое обеспечивает унифицированное представление входных идентификаторов. На рисунке 18 представлен обзор предлагаемой системы PhotoMaker.

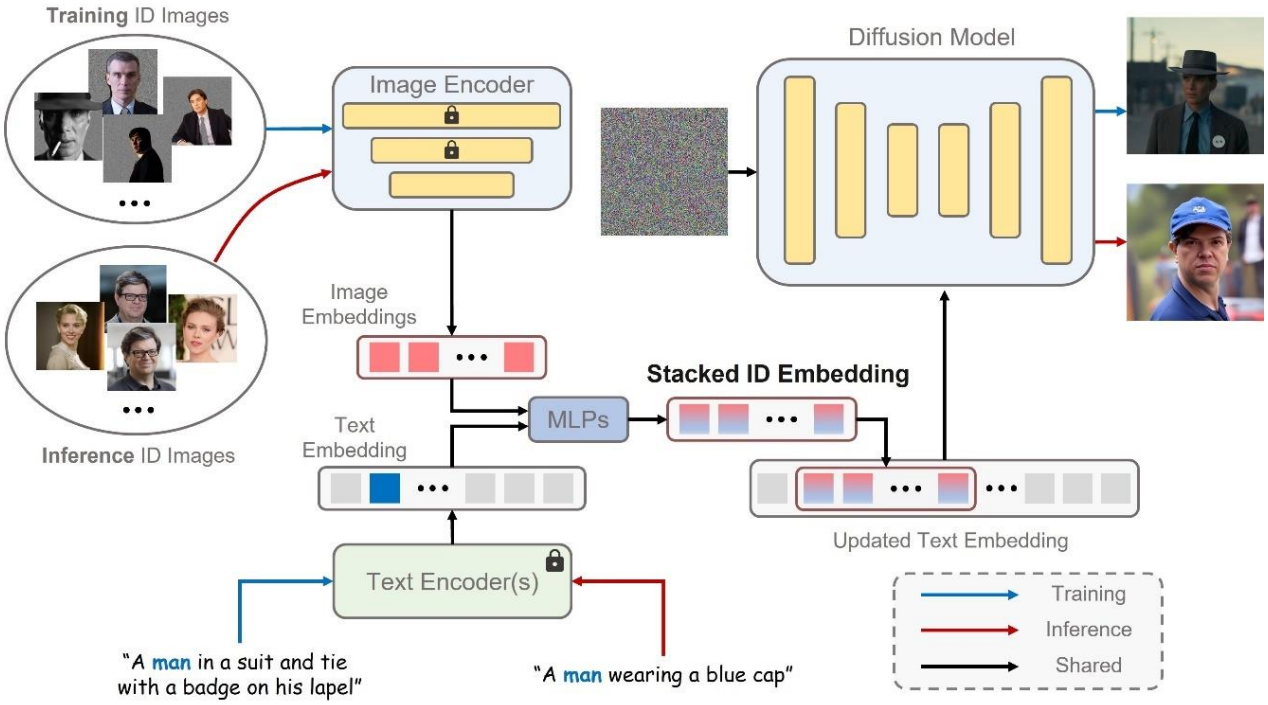

Рисунок 18 — Обзор системы PhotoMaker

В предлагаемом PhotoMaker процесс начинается с получения вложений текста и изображений из кодировщиков текста и кодировщика изображений соответственно. Следующий шаг включает в себя извлечение объединенного внедрения путем объединения соответствующего внедрения класса (например, «man» или «woman») с каждым внедрением изображения. Эти объединенные внедрения затем объединяются по

размеру длины, чтобы сформировать составное внедрение идентификаторов. Это сложенное внедрение идентификаторов впоследствии передается на все уровни перекрестного внимания для адаптивного объединения содержимого идентификаторов в модели распространения. Важно отметить, что хотя во время обучения используются изображения одного и того же идентификатора с замаскированным фоном, система может напрямую вводить изображения разных идентификаторов без искажения фона для генерации нового идентификатора во время вывода.

Энкодеры: В системе PhotoMaker кодировщик изображений CLIP используется для извлечения внедренных изображений, обеспечивая их соответствие пространству текстового представления, используемому в диффузионных моделях. Перед обработкой области каждого входного изображения, не относящиеся к телу, заполняются случайным шумом, чтобы предотвратить помехи от других идентификаторов и фона. Поскольку исходный кодировщик изображений CLIP был обучен на естественных изображениях, они настраивают некоторые слои преобразователя, чтобы улучшить извлечение специфичных для идентификаторов вложений из замаскированных изображений. Кроме того, они вводят обучаемые проекционные слои, соответствующие размерам встроенных изображений и текста. В результате этого процесса создаются внедрения, которые инкапсулируют идентификационную информацию входных изображений, в то время как внедренные тексты получаются из предварительно обученного текстового кодировщика CLIP на основе предоставленного текстового приглашения.

Стекинг: Недавние исследования показали, что модели преобразования текста в изображение могут представлять персонализированную идентификационную информацию персонажа с использованием уникальных токенов. Подход PhotoMaker использует аналогичную стратегию для лучшего сбора идентификационной информации входных изображений человека. В частности, они выделяют слово класса (например, мужчина или женщина) во входном заголовке и извлекают вектор признаков в его позиции во встраивании текста. Затем этот вектор объединяется с каждым внедрением изображения с использованием двух слоев MLP, в результате чего образуются объединенные внедрения, которые более полно представляют изображения входного идентификатора.

Объединение: Авторы PhotoMaker используют механизм перекрестного внимания, присущий моделям диффузии, для адаптивного объединения идентификационной информации, содержащейся в многоуровневом внедрении идентификаторов. Первоначально они заменяют вектор признаков, соответствующий слову класса в исходном текстовом внедрении, на многоуровневое внедрение идентификатора, что приводит к обновленному внедрению текста. Затем применяется операция перекрестного

внимания, при которой обновленное встраивание текста взаимодействует со скрытыми входными данными, закодированными шумоподавителем UNet через матрицы проекции. Эта операция позволяет модели эффективно объединять идентификационную информацию.

3 ПРОЕКТИРОВАНИЕ

В этом разделе подробно рассматривается архитектура системы и методологии, использованные при разработке Телеграм-бота для стилизации изображений с использованием диффузионных моделей.

3.1 Высокоуровневая архитектура системы

В этом разделе представлена высокоуровневая архитектура Телеграм-бота, дающая общее представление о приложении путем определения его ключевых компонентов, их взаимодействия и модульной структуры. Архитектура высокого уровня учитывает как функциональные, так и нефункциональные требования, формируя основу для дальнейшей разработки архитектурных деталей и приложений. Она включает описание архитектурных компонентов, таких как серверы, клиенты и базы данных, а также выбор технологий для их реализации, включая платформы, языки программирования и инструменты. Архитектура высокого уровня, обычно разрабатываемая на ранних стадиях жизненного цикла разработки программного обеспечения, на этапе анализа и проектирования, может быть представлена в виде диаграмм, документации или других форматов, которые помогают детализировать и реализовать архитектуру.

Это приложение разработано на основе микросервисной архитектуры — подхода к созданию приложений путем разбиения их на небольшие независимые компоненты, называемые микросервисами. Каждый микросервис отвечает за выполнение определенной функции и может разрабатываться, тестироваться и развертываться независимо от других, что обеспечивает гибкость и упрощает масштабирование приложений.

3.1.1 Обзор компонентов системы

Эта системная архитектура в основном состоит из трех основных компонентов: **Bot_Handler**, **Image_Generate_Engine** и **Redis-сервер**. Каждый из этих компонентов играет решающую роль в обеспечении эффективной работы Телеграм-бота. Ниже приведено краткое описание каждого компонента:

1) **Bot_Handler:** управляет взаимодействием с пользователем, пользовательскими данными и обрабатывает команды. Он также координирует постановку в очередь и обработку задач создания изображений;

2) **Image_Generate_Engine:** отвечает за создание стилизованных изображений на основе вводимых пользователем данных;

3) **Redis-сервер:** управляет и организует очередь заданий, обеспечивая эффективное распределение и обработку задач между Bot_Handler и Image_Generate_Engine.

3.1.2 Диаграмма развертывания

Чтобы обеспечить четкое и систематическое представление об архитектуре системы Телеграм-бота, диаграмма развертывания UML представлена на рисунке 19. Эта диаграмма иллюстрирует блок-схему и взаимодействие между различными компонентами, которые в совокупности обеспечивают функциональность бота по стилизации изображений Телеграм. Она визуально отображает распределение компонентов системы в различных аппаратных и программных средах, обеспечивая ясность при донесении конструкции системы до заинтересованных сторон.

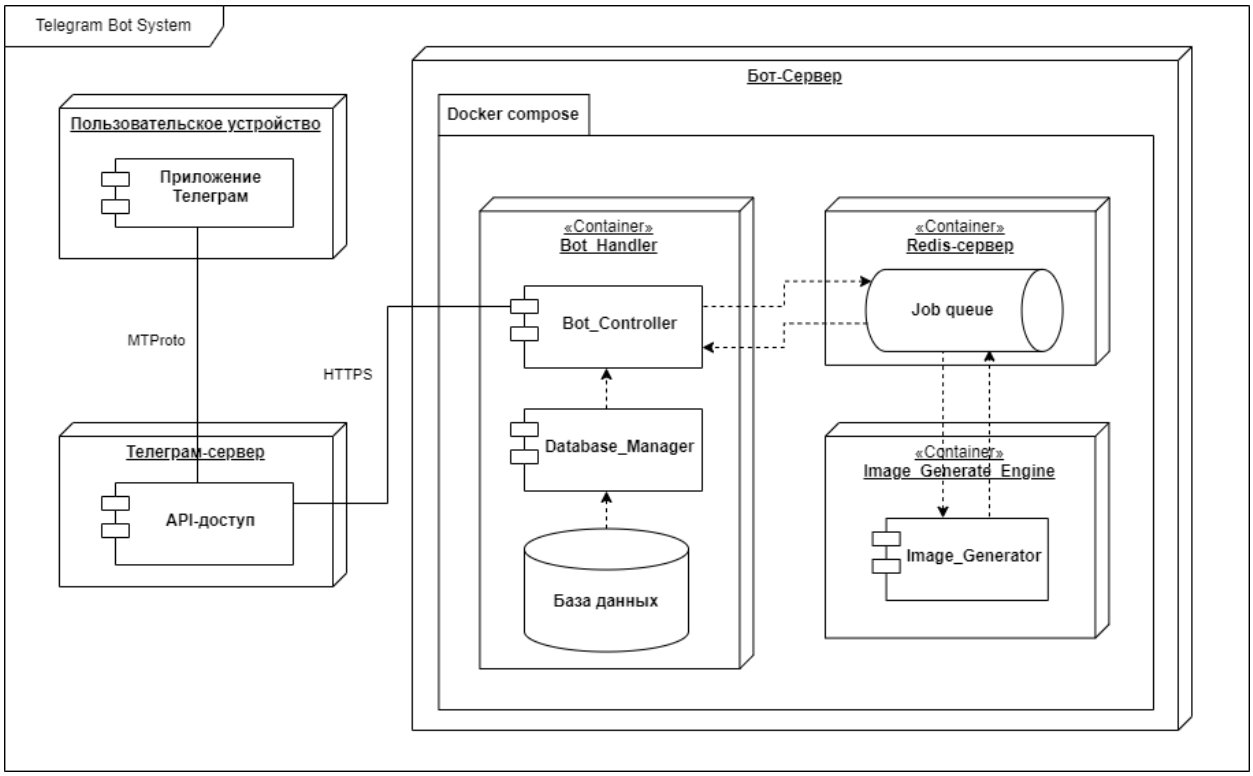

Рисунок 19 — Диаграмма развертывания системы

Диаграмма развертывания показывает физическое расположение аппаратных и программных компонентов системы с подробным описанием их взаимодействия друг с другом. В этой конфигурации весь код развертывается на одном сервере с помощью Docker Compose, при этом каждый компонент инкапсулируется в собственный контейнер Docker.

Docker Compose — это инструмент, который помогает определять и управлять многоконтейнерными приложениями Docker. Используя файл YAML, Docker Compose позволяет настраивать службы, сети и тома вашего приложения. С помощью одной команды Docker Compose может запускать, останавливать и перестраивать службы, что упрощает управление сложными приложениями.

Контейнеры Docker — это легкие, портативные и самодостаточные модули, которые включают в себя все необходимое для запуска программного обеспечения, включая код, среду выполнения, системные инструменты, библиотеки и настройки. Контейнеры обеспечивают согласованную работу программного обеспечения независимо от среды.

Используя Docker Compose для управления развертыванием этих компонентов в их собственных контейнерах, архитектура системы обеспечивает независимое масштабирование и управляемость каждого компонента. Такой подход, ориентированный на микросервисы, создает надежное и гибкое приложение, позволяющее легко обновлять, обслуживать и масштабировать для удовлетворения растущих потребностей пользователей.

3.2 Bot Handler

Bot Handler — это ключевой компонент системной архитектуры, служащий основным интерфейсом между пользователем и серверными службами. Он обрабатывает все взаимодействия с пользователями, организует очередь задач для обработки изображений и управляет потоком данных между базой данных и пользовательским интерфейсом. Рабочий процесс Bot Handler тщательно продуман для обеспечения точности и эффективности обработки запросов пользователей:

1) **Прием входных данных:** входные данные, будь то текстовые или мультимедийные, принимаются через API Телеграм;

2) **Проверка ввода:** каждый ввод тщательно проверяется на предмет правильности формата и соответствия требованиям безопасности;

3) **Интерпретация команд:** входные данные анализируются для определения необходимых действий, команд и предполагаемых операций;

4) **Реализация действия:** в зависимости от команды инициируется соответствующее действие;

5) **Формулировка ответа:** после завершения действия система формулирует соответствующий ответ, который может варьироваться от сгенерированных изображений до текстовых подтверждений или уведомлений об ошибках;

6) **Отправка ответа:** этот последний шаг включает отправку созданного ответа обратно пользователю через API Телеграм.

3.2.1 Функции бота

Функции бота являются важным аспектом системы, поскольку они определяют основные взаимодействия, которые пользователи могут осуществлять с ботом Телеграм. Учитывая, что это чат-бот, эти команды представляют собой основные функции, которые пользователи могут использовать для взаимодействия с системой. Эти функции играют центральную роль в работе бота и должны быть тщательно разработаны, чтобы обеспечить удобство работы с пользователем. Каждая команда предназначена для выполнения определенной задачи, облегчая взаимодействие с пользователем и улучшая общую функциональность бота. В следующих разделах представлен подробный обзор каждой функции бота с описанием того, как они влияют на возможности бота и взаимодействие с пользователем:

– **/start:** эта команда инициирует взаимодействие с ботом. Когда пользователь отправляет эту команду, бот отвечает приветственным сообщением и краткими инструкциями по использованию его функций. Обычно это первая команда, которую будет использовать новый пользователь;

– **/help:** предоставляет подробные инструкции и информацию о том, как использовать различные команды и функции бота. Это полезно для пользователей, которым нужна помощь или есть вопросы о возможностях бота;

– **/menu:** предлагает подробное меню, в котором перечислены все доступные команды и функции. Это помогает пользователям понять, какие действия они могут выполнять, и обеспечивает структурированный способ доступа к функциям бота;

– **/upload:** подготавливает бота к получению предстоящего изображения от пользователя. Эта команда сигнализирует о том, что следующий загруженный файл должен быть добавлен в список изображений пользователя, позволяя боту корректно обрабатывать входящие данные;

– **/show_uploads:** отображает галерею или список всех изображений, загруженных пользователем. Это позволяет пользователям просматривать ранее загруженный контент и управлять им, что упрощает выбор изображений для дальнейшей обработки или удаления;

– **/delete:** позволяет пользователям удалять определенное изображение из загруженной коллекции. Пользователь предоставляет идентификатор изображения, и бот удаляет соответствующее изображение из базы данных;

– **/generate:** указывает, что следующее сообщение пользователя будет рассматриваться как запрос на создание нового изображения. Эта команда предлагает

пользователю предоставить необходимые параметры, такие как стиль или подсказка, для создания персонализированного изображения;

– **/popular_prompts:** отображает список подсказок, которые пользователи используют чаще всего. Эта функция помогает пользователям обнаруживать часто используемые подсказки и тенденции в сообществе ботов;

– **/top_rated_prompts:** показывает подсказки пользователей с самым высоким рейтингом, выделяя высококачественные или особенно успешные подсказки. Это может помочь пользователям выбрать эффективные подсказки для запросов на создание изображений.

Каждая команда предназначена для оптимизации взаимодействия с пользователем, делая процесс интуитивно понятным и эффективным. Такой подход гарантирует, что пользователи смогут легко перемещаться по функциям бота, максимально используя его возможности.

В дополнение к основным функциям бот призван реагировать на ошибки пользователя и направлять пользователей по правильному использованию. Обработка ошибок является важнейшим аспектом конструкции бота Телеграм, гарантируя, что пользователи получат бесперебойную и положительную работу, даже если что-то пойдет не так. Бот оснащен механизмами для эффективного обнаружения ошибок, управления ими и реагирования на них. Бот постоянно отслеживает вводимые пользователем данные, чтобы убедиться, что они соответствуют требуемому формату и стандартам безопасности. Сюда входит проверка текстовых команд, проверка правильности мультимедийных файлов и обеспечение соответствия всех входных данных предопределенным правилам.

При обнаружении ошибки бот дает четкие инструкции, как исправить ввод. Например, если пользователь загружает файл неподдерживаемого типа, бот укажет приемлемые форматы. В случае временных проблем, таких как временная недоступность сервера, бот предлагает пользователям повторить запрос после небольшой задержки.

3.2.2 База данных ботов

Для выполнения функций, описанных в предыдущих разделах, база данных ботов была разработана с упором на эффективность, масштабируемость и удобство обслуживания. Структура базы данных обеспечивает эффективное управление пользовательскими данными, загруженными изображениями и подсказками, обеспечивая удобство взаимодействия пользователей с ботом. Конструкция соответствует модели реляционной базы данных для обеспечения целостности и доступности данных.

Рисунок 20 представляет собой визуальное представление схемы базы данных, иллюстрирующее, как таблицы связаны между собой посредством отношений первичного и внешнего ключей.

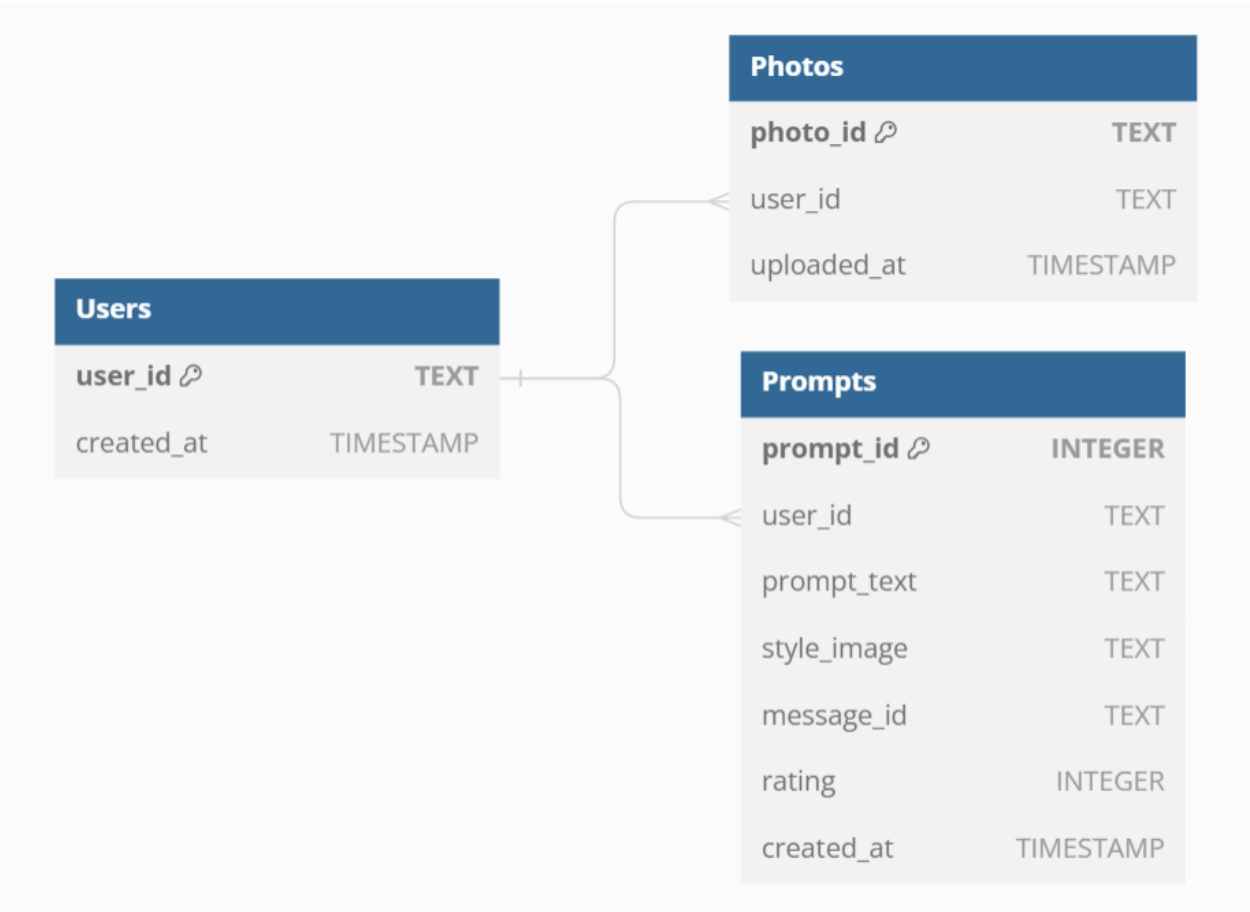

Рисунок 20 — Схемы базы данных Телеграм-бота

База данных бота состоит из трех основных таблиц: «Пользователи» (Users), «Фотографии» (Photos) и «Подсказки» (Prompts). Каждая таблица служит определенной цели и связана с другими посредством внешних ключей, обеспечивая согласованность данных и позволяя выполнять эффективные запросы.

– **Таблица «Пользователи»** — хранит информацию о пользователях, взаимодействующих с ботом;

– **Таблица «Фотографии»** — управляет изображениями, загруженными пользователями;

– **Таблица «Подсказки»** — хранит информацию о подсказках, предоставленных пользователями для создания изображений, и их рейтинге.

Отношения между этими таблицами проиллюстрированы на схеме базы данных (рисунок 21). Таблица «Пользователи» связана с таблицами «Фотографии» и «Подсказки» через поле user_id. Эта реляционная структура гарантирует, что данные каждого

пользователя, включая загруженные фотографии и отправленные запросы, управляются последовательно и эффективно.

Разработав базу данных с учетом этих принципов, бот Телеграм может эффективно обрабатывать взаимодействие с пользователями, управлять потоком данных и предоставлять надежную платформу для создания персонализированных изображений. Кроме того, такая конструкция позволяет эффективно выполнять запросы, например, выявлять наиболее часто используемые подсказки и самые популярные, что позволяет боту предоставлять пользователям подробные рекомендации и анализ тенденций.

3.3 Generate Image Engine

Generate_Image_Engine — это основной компонент Телеграм-бота, отвечающий за создание персонализированных изображений на основе пользовательского ввода. В этом разделе описываются методологии, используемые для реализации этой важной функции.

В движке используются две основные технологии: библиотека **Diffusers [17]** для интеграции модели **Stable Diffusion XL** и **PyTorch [18]** для реализации модели **PhotoMaker**. Эти технологии работают вместе, чтобы эффективно предоставлять пользователям высококачественные персонализированные изображения.

Поскольку это модели глубокого обучения, их обучение на большом наборе данных потребует значительного времени и вычислительных ресурсов. Поэтому в этой диссертации Телеграм-бот использует предварительно обученные веса моделей для обеих моделей. Эти предварительно обученные веса получены из Hugging Face Hub, который обеспечивает доступ к высококачественным современным моделям искусственного интеллекта для создания изображений.

Hugging Face Hub [19] — это платформа, на которой размещена обширная коллекция предварительно обученных моделей, созданных различными исследователями искусственного интеллекта. Он предлагает централизованный репозиторий, где разработчики могут получать доступ к моделям, обмениваться ими и совместно работать над ними, что облегчает интеграцию расширенных возможностей искусственного интеллекта в приложения.

Библиотека Diffusers была выбрана из-за ее прямой поддержки моделей Hugging Face и удобного интерфейса для развертывания моделей диффузии. Она обеспечивает бесшовную интеграцию с моделью Stable Diffusion XL, позволяя эффективно генерировать изображения. Stable Diffusion XL, интегрированный в систему через библиотеку Diffusers, обеспечивает плавную загрузку, выполнение и взаимодействие моделей. Эта интеграция

позволяет эффективно обрабатывать запросы пользователей и генерировать динамические изображения на основе введенных пользователем данных.

Библиотека PyTorch предлагает обширную поддержку и гибкость настройки модели, что делает её идеальной для настройки, необходимой для архитектуры PhotoMaker. Её надежные вычислительные возможности обеспечивают эффективную обработку задач глубокого обучения. PhotoMaker, реализованный с использованием PyTorch, предоставляет необходимые возможности вычислений и управления моделями. Эта настройка включает в себя настройку модели для получения входных данных с сервера Redis, обработки этих входных данных для создания персонализированных изображений и вывода результатов обратно в систему для доставки пользователю.

Рабочий процесс Generate Image Engine предназначен для обеспечения эффективного и точного создания изображений с использованием передовых моделей искусственного интеллекта и оптимизированной обработки данных. Ниже приведен рабочий процесс этого компонента:

1) **Получение входных данных**: Generate_Image_Engine получает запросы пользователя и данные изображения с сервера Redis;

2) **Обработка**: движок использует модель Stable Diffusion XL и модель PhotoMaker для обработки входных данных и создания персонализированных изображений;

3) **Результат ответа**: сгенерированные изображения отправляются обратно в систему через сервер Redis, который затем доставляет их пользователю через интерфейс Телеграм-бота.

3.4 Redis-сервер

Сервер Redis составляет основу управления задачами в Телеграм-бота, обеспечивая эффективную обработку и масштабирование запросов пользователей на создание изображений. В этом разделе подробно описаны конфигурация, функциональность и интеграция очередей заданий Redis и RQ (Redis Queue) [20] для облегчения выполнения нескольких запросов на создание изображений.

3.4.1 Ограничения вычислительных ресурсов при генерации изображений

Развертывание передовых технологий генерации изображений, таких как Stable Diffusion XL и PhotoMaker, в приложениях реального времени требует четкого понимания их конкретных вычислительных требований. Эти модели, хотя и различаются по архитектурным основам и функциональным задачам, имеют общую характеристику —

высокую нагрузку на ресурсы графического процессора, особенно по отношению к памяти видеокарты.

Как Stable Diffusion XL, так и PhotoMaker демонстрируют более широкую тенденцию в технологиях создания изображений на основе искусственного интеллекта к увеличению вычислительных потребностей. Эти требования в первую очередь обусловлены необходимостью обработки больших объемов данных и быстрого и надежного выполнения сложных вычислений. Высокий уровень использования видеопамяти является критическим фактором, поскольку он напрямую влияет на способность моделей генерировать изображения без ущерба для скорости или качества. Это особенно актуально в приложениях реального времени, таких как Telegram-бот, где задержки или проблемы с производительностью могут существенно повлиять на работу пользователей.

Интенсивное использование ресурсов графического процессора для одной задачи может привести к возникновению узких мест при одновременном выполнении нескольких запросов. Это критическая проблема в приложениях реального времени, таких как Телеграм-бот, где несколько пользователей могут одновременно отправлять запросы на создание изображений. Эффективное управление этими запросами без длительного ожидания или сбоев системы требует стратегического подхода к распределению ресурсов.

Чтобы эффективно управлять этими вычислительными потребностями, крайне важно реализовать стратегии, которые оптимизируют распределение ресурсов и улучшают масштабируемость системы. Благодаря интеграции механизма очереди задач с использованием Redis Server и RQ (Redis Queue) система может обрабатывать запросы на создание изображений последовательно. Такой подход гарантирует, что каждая задача получит необходимые ресурсы графического процессора, не перегружая систему. Следовательно, эта настройка максимизирует эффективность доступных ресурсов графического процессора, сохраняя стабильность и производительность системы даже при высоких нагрузках.

3.4.2 Технические характеристики

Redis-сервер: Сервер Redis действует как централизованное хранилище структур данных, управляя состоянием и очередностью задач в системе. Конфигурация обеспечивает высокую доступность и быстрый доступ, что крайне важно для приложений реального времени, таких как Телеграм-бот.

Реализация очереди заданий RQ: Очередь заданий RQ (*англ:* Redis Queue) — это библиотека Python, которая упрощает постановку заданий в очередь и их обработку в

фоновом режиме с помощью рабочих процессов. Она использует Redis для хранения заданий и их статусов, что делает его надежным выбором для асинхронного управления задачами:

– **Отправка задачи:** когда пользователь отправляет запрос через бота, Bot_Handler обрабатывает запрос, упаковывает необходимые данные и ставит задание в очередь generate_tasks;

– **Обработка задач:** Generate_Image_Engine управляет обработчиками RQ, которые отслеживают очередь generate_tasks. Эти рабочие процессы выполняют функцию создания изображения, когда задачи выводятся из очереди;

– **Обработка результатов:** после создания изображения результат передается обратно в Bot_Handler, который затем доставляет изображение пользователю.

4 РЕАЛИЗАЦИЯ

Разработка бота Телеграм включает в себя несколько ключевых этапов: от регистрации бота до реализации его функционала с помощью **«Telegram Bot API»** [21] и интеграции с базой данных. В этом разделе описывается первоначальная настройка и процесс разработки для создания функционального и интерактивного бота Телеграм.

Первым шагом в разработке Телеграм-бота является его регистрация в BotFather. BotFather — это бот Телеграма, который помогает разработчикам создавать своих ботов и управлять ими. Настройте имя бота, приветственный баннер, список команд, информацию и описание через BotFather. Эта конфигурация будет определять, как пользователи взаимодействуют с ботом и какую информацию они видят, когда начинают его использовать. После выполнения этих шагов у бота появится экран приветствия, аналогичный рис. 6, на котором отображается имя бота и краткое введение.

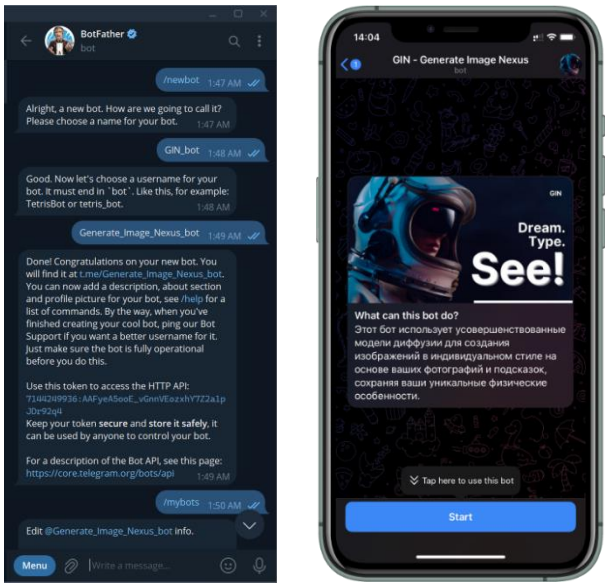

Рисунок 21 — Архитектура модели стабильной диффузии

Разработка бота упрощается с помощью **Telegram Bot API**, который предоставляет набор методов и инструментов для взаимодействия с пользователями Телеграм и обработки сообщений. **Telegram Bot API** — это мощный и гибкий интерфейс, который позволяет разработчикам создавать сложных ботов, способных выполнять широкий спектр задач.

Для упрощения работы с **Telegram Bot API** используется библиотека **pyTelegramBotAPI [22]**. Эта библиотека способствует упрощению взаимодействия с API Телеграм Bot. Она абстрагирует сложности прямых вызовов API, позволяя разработчикам сосредоточиться на реализации функциональности, а не на управлении издержками связи. Библиотека поддерживает как синхронные, так и асинхронные операции, обеспечивая масштабируемый подход к одновременной обработке нескольких пользовательских

взаимодействий. Это важно для поддержания производительности и оперативности при различных нагрузках на запросы пользователей.

SQLite [23] используется для хранения важной информации, такой как пути к загруженным изображениям, пользовательские настройки и данные, связанные с подсказками и рейтингами. SQLite предлагает двойные преимущества: простую настройку и мощные возможности обработки данных без затрат на более сложные системы баз данных. Это делает его идеальным выбором для приложений, требующих высокой надежности и скорости при умеренном объеме и сложности данных.

На рисунке 22 показаны интерфейсы взаимодействия с пользователем бота Телеграм при различных входных командах.

(а) (б) (в) (г)

Рисунок 22 — Интерфейсы взаимодействия пользователя Телеграм-бота для команд (а) /start, (б) /help, (в) /menu, (г) /popular_prompts

Telegram Bot API предоставляет обширный набор инструментов, которые разработчики могут использовать для создания очень гибких и адаптированных пользовательских интерфейсов. Например, в случае с этим ботом были включены интерактивные кнопки, которые значительно повышают удобство пользователя. Эти кнопки позволяют пользователям легко получить доступ к списку загруженных изображений простым касанием, устраняя необходимость вводить определенные команды. Эта функция не только упрощает процесс взаимодействия, но и обогащает пользовательский опыт, делая навигацию по различным функциям бота более интуитивно понятной. На рисунке 23 показана эта конкретная функция, иллюстрирующая, как пользователи могут переключаться между загрузками изображений, нажимая кнопки со стрелками в интерфейсе бота.

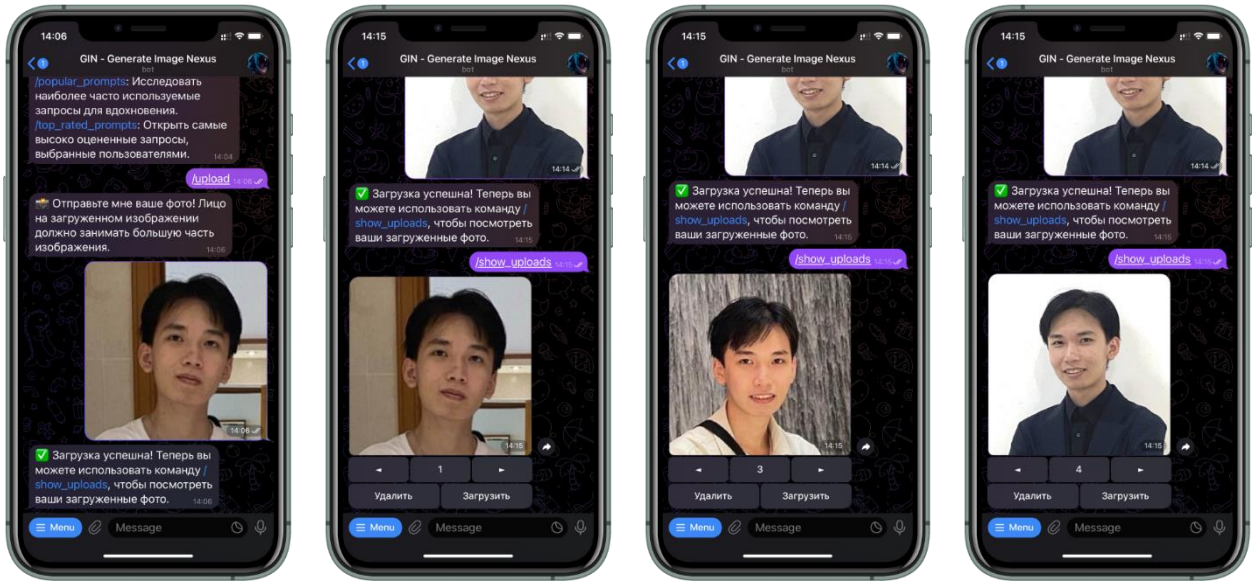

Рисунок 23 — Взаимодействие пользователя с галереей загрузки изображений в Телеграм-боте

Модель Stable Diffusion XL, интегрированная в этого Телеграм-бота, предлагает пользователям возможность создавать изображения в различных стилях. Такая универсальность возможна благодаря расширенным возможностям модели в интерпретации и рендеринге художественных деталей на основе текстовых подсказок. Однако, чтобы в полной мере использовать эти возможности, пользователи должны иметь возможность создавать подсказки, которые точно отражают их творческие идеи. Понимая, что это может быть сложной задачей, особенно для тех, кто не имеет опыта работы с такой технологией, Телеграм-бот упрощает процесс, предоставляя выбор заранее выбранных стилей.

Когда пользователь выбирает один из этих стилей, бот автоматически дополняет подсказку пользователя конкретными описательными словами, соответствующими выбранному стилю. Это усовершенствование оптимизирует входные данные для Stable Diffusion XL, гарантируя, что сгенерированное изображение точно соответствует видению пользователя. Интегрируя эти функции, бот не только делает расширенную генерацию изображений более доступной, но и улучшает общий пользовательский опыт. На рисунке 24 показаны эти стили вместе с процессом создания пользовательского изображения.

Рисунок 24 — Процесс генерации изображений в Телеграм-боте

ЗАКЛЮЧЕНИЕ

В данной дипломной работе была разработана и реализована система в виде Telegram-бота, использующего диффузионные модели для стилизации изображений, что позволяет пользователям создавать персонализированные изображения, сохраняя их индивидуальные черты лица. Этот бот представляет собой шаг на пути к тому, чтобы сделать сложные модели генерации изображений доступными и простыми в использовании для всех.

Работа включала всесторонний обзор существующих моделей диффузии и методов их применения для генерации изображений с упором на персонализацию. Были рассмотрены основные достижения в этой области, в том числе модели Stable Diffusion и PhotoMaker. Эти модели демонстрируют значительные возможности по созданию фотореалистичных и художественных изображений на основе текстовых подсказок.

Разработка Телеграм-бота включала в себя несколько этапов: от анализа предметной области и выбора подходящих технологий до реализации и интеграции моделей с использованием библиотек PyTorch и Diffusers. Архитектура системы построена с использованием микросервисного подхода, что обеспечило гибкость и масштабируемость приложения. Интеграция сервера Redis для управления очередями задач позволила эффективно распределять вычислительные ресурсы и обрабатывать несколько пользовательских запросов одновременно, что критично для реальных приложений с высокой нагрузкой.

В заключение, данная работа демонстрирует успешное применение диффузионных моделей для создания персонализированных изображений и предлагает перспективное решение для интеграции сложных алгоритмов генерации изображений в повседневные пользовательские приложения. Дальнейшие исследования могут быть направлены на улучшение качества и скорости генерации изображений, а также на расширение функциональности бота для поддержки большего количества стилей и параметров персонализации.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Goodfellow I., Pouget-Abadie J., Mirza M., Xu B., Warde-Farley D., Ozair S., Courville A., Bengio Y. Generative Adversarial Nets // Advances in Neural Information Processing Systems – 2014. – Vol. 27. P. 2672 – 2680.
2. Midjourney [Электронный ресурс]. – URL: https://www.midjourney.com/
3. Stability AI. Stable Diffusion [Электронный ресурс]. – URL: https://stability.ai/
4. Abdal R., Zhu P., Mitra T., Wonka P. PhotoMaker: Fine-Tuning Diffusion Models with Stacked ID Embeddings for Photorealistic Portrait Generation // arXiv preprint arXiv:2312.00201 – 2023.
5. Sohl-Dickstein J., Weiss E., Maheswaranathan N., Ganguli S. Deep Unsupervised Learning using Nonequilibrium Thermodynamics // arXiv preprint arXiv:1503.03585 – 2015.
6. Song Y., Ermon S. Generative Modeling by Estimating Gradients of the Data Distribution // Advances in Neural Information Processing Systems – 2019. – Vol. 32. P. 11895 – 11907.
7. Ho J., Jain A., Abbeel P. Denoising Diffusion Probabilistic Models // Advances in Neural Information Processing Systems – 2020. – Vol. 33. P. 6840 – 6851.
8. Ramesh A., Pavlov M., Goh G., Gray S., Voss C., Radford A., Chen M., Sutskever I. Zero-shot Text-to-Image Generation // arXiv preprint arXiv:2102.12092 – 2021.
9. Saharia C., Chan W., Saxena S., Li L., Whang J., Denton E., Ghasemipour S., Ayan B.K., Mahdavi S.S., Lopes R.G., Salimans T., Ho J., Fleet D.J., Norouzi M. Photorealistic Text-to-Image Diffusion Models with Deep Language Understanding // arXiv preprint arXiv:2205.11487 – 2022.
10. Gal R., Alaluf Y., Atzmon Y., Patashnik O., Bermano A.H., Cohen-Or D. Textual Inversion: Enabling Personalized Text-to-Image Generation // arXiv preprint arXiv:2208.01618 – 2022.
11. Ruiz N., Li T., Grigorev N., Belongie S. DreamBooth: Fine Tuning Text-to-Image Diffusion Models for Subject-Driven Generation // arXiv preprint arXiv:2208.12242 – 2022.
12. Hu E., Shen Y., Wallis P., Allen-Zhu Z., Li Y., Wang L., Chen W., Wang K. LoRA: Low-Rank Adaptation of Large Language Models // arXiv preprint arXiv:2106.09685 – 2021.
13. Jiang Z., Liang Y., Fei-Fei L., Li J. FastComposer: Tuning-Free Text-to-Image Composition // arXiv preprint arXiv:2305.18252 – 2023.
14. Liu Z., Zhang Y., Wei Z., Zhang Y., Li J. IP-Adapter: Text-to-Image Diffusion Models with Efficient Pathway to User Intent // arXiv preprint arXiv:2308.06387 – 2023.
15. Abdal R., Zhu P., Mitra T., Wonka P. PhotoMaker: Fine-Tuning Diffusion Models with Stacked ID Embeddings for Photorealistic Portrait Generation // arXiv preprint arXiv:2312.00201 – 2023.

16. Esser P., Rombach R., Ommer B. Taming Transformers for High-Resolution Image Synthesis // Proceedings of the IEEE/CVF International Conference on Computer Vision – 2021. P. 12873 – 12883.
17. Diffusers [Электронный ресурс]. – URL: https://huggingface.co/docs/diffusers/index
18. PyTorch [Электронный ресурс]. – URL: https://pytorch.org/docs/stable/index.html
19. Hugging Face [Электронный ресурс]. – URL: https://huggingface.co/
20. Redis Queue [Электронный ресурс]. – URL: https://python-rq.org/
21. Telegram Bot API [Электронный ресурс]. – URL: https://core.telegram.org/bots/api
22. pyTelegramBotAPI [Электронный ресурс]. – URL: https://github.com/eternnoir/pyTelegramBotAPI
23. SQLite [Электронный ресурс]. – URL: https://www.sqlite.org/index.html