Міністерство освіти і науки України
Харківський національний університет радіоелектроніки

Факультет Комп'ютерних наук
(повна назва)

Кафедра Штучного інтелекту
(повна назва)

КВАЛІФІКАЦІЙНА РОБОТА
Пояснювальна записка

рівень вищої освіти другий (магістерський)

Дослідження дифузійних моделей для генерації зображень у порівнянні з генеративними змагальними мережами
(тема)

Виконав:
студент 2 курсу, групи СШІм-21-1
Овсієнко В.С.
(прізвище, ініціали)

Спеціальність 122 Комп'ютерні науки
(код і повна назва спеціальності)

Тип програми освітньо-наукова
(освітньо-професійна або освітньо-наукова)

Освітня програма Системи штучного інтелекту
(повна назва спеціалізації)

Керівник д.т.н., проф. Семенець В.В.
(посада, прізвище, ініціали)

Допускається до захисту

Зав. кафедри _____ В.О. Філатов
(підпис) (прізвище, ініціали)

2023 р.

Харківський національний університет радіоелектроніки

Факультет Комп'ютерних наук
(повна назва)

Кафедра Штучного інтелекту
(повна назва)

Рівень вищої освіти другий (магістерський)

Спеціальність 122 Комп'ютерні науки
(код і повна назва)

Тип програми освітньо-наукова
(освітньо-професійна або освітньо-наукова)

Освітня програма Системи штучного інтелекту (СШІ)
(повна назва)

ЗАТВЕРДЖУЮ:
Зав. кафедри _____
(підпис)
«_____» _____ 20 ___ р.

ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ РОБОТУ

студентові Овсієнко Владиславі Сергіївні
(прізвище, ім'я, по батькові)

1. Тема роботи Дослідження дифузійних моделей для генерації зображень у порівнянні з генеративними змагальними мережами

затверджена наказом університету від 31 березня 20 23 р. № 306Ст

2. Термін подання студентом роботи до екзаменаційної комісії 16 травня 20 23 р.

3. Вихідні дані до роботи науково-технічні публікації, дані Інтернет-джерел та відомих наукових проектів щодо розробки та дослідження генеративних моделей, набори даних зображень з підписами

4. Перелік питань, що потрібно опрацювати в роботі
1) Аналіз предметної галузі і постановка задачі дослідження
2), Огляд генеративних моделей,
3) Програмна реалізація і порівняння натренованих моделей

5. Перелік графічного матеріалу із зазначенням креслеників, схем, плакатів, комп'ютерних ілюстрацій (п.5 включається до завдання за рішенням випускової кафедри) Рисунок 1 – Опис типів завдань генерації зображень, Рисунок 2 – Архітектура генеративної змагальної мережі, Рисунок 3 – Архітектура автокодувальника, Рисунок 4 – Різниця між автокодувальником (детермінованим) і варіаційним автокодувальником (імовірнісним), Рисунок 5 – Архітектура генеративних моделей на основі потоку, Рисунок 6 – Концепт дифузійних моделей, Рисунок 7 - Хронологія створення генеративних моделей «текст-в-зображення», Рисунок 8 – Концепт генеративних змагальних мереж, Рисунок 9 – Концепт тренування дискримінатора, Рисунок 10 – Концепт тренування генератора, Рисунок 11 – Приклад архітектури моделі Conditional GAN, Рисунок 12 – Архітектура DC-GAN, Рисунок 13 – Архітектура StackGAN, Рисунок 14 – Архітектура AttnGAN

6. Консультанти розділів роботи (п.6 включається до завдання за наявності консультантів згідно з наказом, зазначеним у п.1)

Найменування розділу	Консультант (посада, прізвище, ім'я, по батькові)	Позначка консультанта про виконання розділу	
		підпис	дата

КАЛЕНДАРНИЙ ПЛАН

№	Назва етапів роботи	Терміни виконання етапів роботи	Примітка
1	Отримання завдання на кваліфікаційну роботу	03.04.23	виконано
2	Аналіз предметної області і постановка завдання	04.04.23-09.04.23	виконано
3	Вибір моделей і вхідних даних	10.04.23-11.04.23	виконано
4	Експериментальне моделювання та навчання моделей	12.04.23-25.04.03	виконано
5	Оцінка результатів	26.04.23-28.04.23	виконано
6	Написання пояснювальної записки	29.03.23-05.05.2023	виконано
7	Попередній захист	11.05.2023	виконано
8	Захист перед ЕК	16.05.2023	виконано

Дата видачі завдання 3 квітня 2023 р.

Студент _____
(підпис)

Керівник роботи _____ д.т.н., проф. Семенець В.В.
(підпис) (посада, прізвище, ініціали)

РЕФЕРАТ

Пояснювальна записка: 84 с., 34 рис., 4 табл., 2 дод., 41 джерело.

ГЕНЕРАТИВНІ ЗМАГАЛЬНІ МЕРЕЖІ, ГЕНЕРАЦІЯ ЗОБРАЖЕНЬ, ДИФУЗІЙНА МОДЕЛЬ, КОМП'ЮТЕРНИЙ ЗІР, СИНТЕТИЧНЕ ЗОБРАЖЕННЯ.

Об'єкт дослідження – процес генерації зображень дифузійними моделями, особливості і процес навчання.

Предмет дослідження – методи моделювання штучних нейронних мережі і їх навчання, порівняння, метрики оцінки якості генерацій.

Мета роботи – є вивчення та застосування дифузійних моделей для генерації зображень, а також порівняння їх ефективності з генеративними змагальними мережами.

Методами дослідження є аналіз технічної літератури з області генерації зображень, порівняльний аналіз досліджень.

У цій роботі розглядається процес генерації зображень за допомогою дифузійних моделей, що є альтернативою генеративним змагальним мережам. Застосування дифузійних моделей дає можливість генерувати якісні зображення з меншою кількістю параметрів моделі та більш стабільним навчанням.

ABSTRACT

Explanatory note: 84 p., 34 fig., 4 tables, 2 ann., 41 sources.

COMPUTER VISION, DIFFUSION MODEL, GENERATIVE ADVERSARIAL NETWORK, IMAGE GENERSTION, SYNTETIC IMAGE

The object of research is the process of image generation by diffusion models, features and training process.

The subject of the research is methods of modeling artificial neural networks and their training, comparison, and metrics for assessing the quality of generations.The purpose of the study is to study and apply diffusion models for image generation, as well as to compare their efficiency with generative adversarial networks.

Research methods include analysis of technical literature on image generation and comparative analysis of studies.

This paper discusses the process of image generation using diffusion models, which is an alternative to generative adversarial networks. The use of diffusion models makes it possible to generate high-quality images with fewer model parameters and more stable training.

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СИМВОЛІВ, ОДИНИЦЬ, СКОРОЧЕНЬ ТА ТЕРМІНІВ

AttnGAN – Attention Genertive Adversarial Network – генеративна змагальна мережа з увагою;

Bridge-FP – Bridge Feature Predictor – предиктор мостових ознак;

CLIP – Contrastive Language-Image Pre-Training – контрастне мова-зображення попереднє навчання;

DC-GAN – Deep Convolutional Generative Adversarial Network – генеративна змагальна мережа з глибокими згортками;

DDIM – неявна модель дифузії з усуненням шуму;

DFBlock – Deep text-image Fusion Block – блок глибокого злиття тексту і зображення;

DF-GAN – Deep Fusion Genertive Adversarial Network – генеративна змагальна мережа з глибоким синтезом;

DM – Diffusion Model – дифузійна модель;

DPM – Diffusion Probabilistic Model – імовірнісна модель дифузії;

DDPM – Denoising Diffusion Probabilistic Model – імовірнісна дифузійна модель з усуненням шуму;

FID – Frechet Inception Distance – Фреше Inception відстань;

GALIP – Generative Adversarial CLIP – генеративна змагальна CLIP;

GAN – Generative Adversarial Network – генеративна змагальна мережа;

GLIDE – Guided Language to Image Diffusion for Generation and Editing – керована мова для дифузії зображень для створення та редагування;

IS – Inception Score – Inception оцінка;

LDM – Latent Diffusion Model – латентна дифузійна модель;

SGM – Sequence Generation Model – модель генерації послідовностей;

VAE – Variational Autoencoder – варіаційний автокодувальник.

ВСТУП

За останні кілька років генеративні моделі набули здатності генерувати людиноподібну природну мову, нескінченні високоякісні синтетичні зображення, мову та музику. Ці моделі можуть бути використані різними способами, наприклад, для генерування зображень на основі текстових підказок або для вивчення корисних репрезентацій ознак. Хоча ці моделі вже здатні створювати реалістичні зображення і звук, все ще існує багато можливостей для вдосконалення поточного стану, і кращі генеративні моделі можуть мати широкий вплив на графічний дизайн, ігри, музичне виробництво та музику, дизайн, ігри та незліченну кількість інших галузей.

З розвитком глибинного навчання задача перетворення тексту в зображення стало одним з найбільш вражаючих застосувань комп'ютерного зору. Розробка автоматичної системи, яка генерує візуально реалістичні зображення з текстових описів є нетривіальним завданням і тому може розглядатися як важлива віха на шляху до людиноподібного або загального штучного інтелекту.

Перетворення тексту в зображення передбачає навчання моделі глибокого навчання на великому наборі зображень і відповідних текстових описів. Модель вчиться асоціювати певні слова та фрази з конкретними візуальними характеристиками, такими як форми, кольори та текстури. Під час виведення модель приймає текстовий опис як вхідні дані і генерує зображення, яке відповідає цьому опису. Однією з найпоширеніших архітектур є генеративна змагальна мережа (GAN). Хоча ці мережі досягли вражаючих результатів в генерації реалістичних зображеннях, їхні недоліки ускладнюють їхнє масштабування та застосування до нових доменів. Як наслідок, було зроблено багато роботи для досягнення якості вибірки, подібної до GAN, за допомогою ймовірнісної моделі – моделі дифузії.

Дифузійні моделі стають все більш популярними, оскільки вони забезпечують стабільність навчання, а також якісні результати генерації. Вони входять до класу ймовірнісних моделей, які створюють високоякісні зображення, пропонуючи при цьому такі бажані властивості, як покриття розподілу, стаціонарна навчальна задача та легка масштабованість.

У цій роботі буде розглянуто аналіз існуючих підходів до генерації зображень, розглянуто детально архітектури генеративних змагальних мереж і дифузійних моделей і порівняно їх на практиці.

1 АНАЛІЗ ПРЕДМЕТНОЇ ГАЛУЗІ ТА ПОСТАНОВКА ЗАДАЧІ ДОСЛІДЖЕННЯ

1.1 Опис завдання генерації зображень

З розвитком у різних галузях аналіз даних став справжнім викликом, особливо обробка візуальних даних. Крім того, з розвитком аналітичних технологій, таких як машинне та глибинне навчання, дослідник може знайти багато рішень для деяких проблем, які не можна було вирішити за допомогою традиційних та статистичних методів. Це дозволяє створювати багато напрямків досліджень, таких як генерація зображень з різних типів даних.

Різноманітні підходи до генерації зображень в основному поділяються на багато категорій, включаючи генерацію зображень, переклад зображення в зображення, генерацію ескізу в зображення, умовного зображення, тексту в зображення, декількох зображень, обличь, макета в зображення, зображень з урахуванням пози, відео, панорамних зображень та генерацію графів сцен. На рисунку 1.1 представлено опис цих категорій [1].

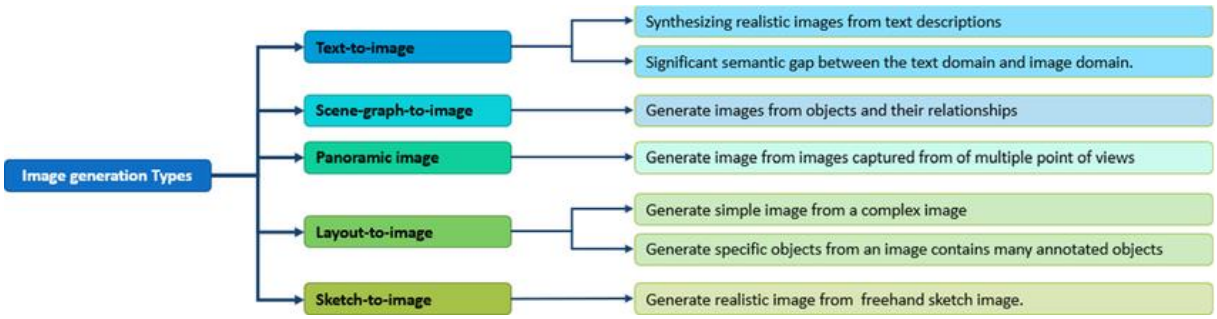

Рисунок 1.1 – Опис типів завдань генерації зображень

Задача перетворення тексту в зображення залишається унікальною і складною проблемою в галузі глибокого навчання. Існує кілька

особливостей, які відрізняють цю задачу від інших задач, пов'язаних із зображеннями, таких як класифікація зображень і виявлення об'єктів:

– неоднозначність і суб'єктивність текстових описів;

– висока розмірність;

– великі об’єми маркованих даних;

– якість і реалістичність зображень;

– обчислювальна складність.

Неоднозначність текстових описів викликана тим, що різні люди можуть по-різному інтерпретувати один і той самий текстовий опис, що може призвести до різних візуальних уявлень. Наприклад, опис «жовта квітка» може бути інтерпретований як ромашка, соняшник або будь-який інший тип жовтої квітки. Модель повинна бути навчена розуміти нюанси мови і точно інтерпретувати текстові описи. Текстові описи можуть бути дуже довгими, і існує багато різних способів описати один і той самий об'єкт або сцену. Це ускладнює засвоєння моделлю складних взаємозв'язків між текстовими описами та відповідними зображеннями.

На відміну від інших завдань, пов'язаних із зображеннями, для синтезу текст-зображення доступно невелика кількість маркованих наборів даних. Це пов'язано з тим, що вручну анотувати текстові описи з відповідними зображеннями складно і займає багато часу. Як наслідок, більшість моделей синтезу текст-зображення покладаються на неконтрольовані або напівконтрольовані підходи до навчання.

Згенеровані зображення повинні точно передавати деталі та характеристики об'єкта або сцени, описані в тексті, а також виглядати природно і реалістично. Для навчання вимагається потужне обладнання та тривалий час на навчання. Висока розмірність вхідного простору та складність генеративних моделей роблять це завдання особливо складним.

Незважаючи на ці виклики, синтез тексту в зображення має потенціал для революції в багатьох галузях, зокрема в іграх, електронній комерції, рекламі, медицині [2].

1.2 Аналіз існуючих рішень

За останні роки були розроблені різні підходи та моделі, що спираються на глибоке навчання для генерації, включаючи генеративні змагальні мережі, варіаційні автокодувальники (VAEs), потокові моделі (Flow-based models) та дифузійні моделі (DMs).

1.2.1 Генеративні змагальні мережі

Вперше генеративна змагальна мережа була представлена Яном Ґудфелоу у 2014 році [3], але Скотт Рід та ін. вперше застосували їх для перетворення тексту в зображення у 2016 році [4].

Генеративна змагальна мережа складається з двох моделей (рисунок 1.2). Дискримінатор D оцінює ймовірність того, що дана вибірка походить з реального набору даних. Він працює як критик і оптимізований для того, щоб відрізняти фальшиві зразки від справжніх. Генератор G виводить синтетичні вибірки на основі вхідної змінної шуму вносить потенційну різноманітність на виході). Генератор навчено вловлювати реальний розподіл даних так, щоб його генеративні зразки були максимально схожими на реальні, або могли «обдурити» дискримінатор, щоб запропонувати високу ймовірність. Ці дві моделі змагаються між собою під час навчання: генератор G намагається обдурити дискримінатор, в той час як модель критика D намагається не бути обдуреною. Це змагання з нульовою сумою між двома моделями мотивує обидві покращувати свої функціональні можливості [5].

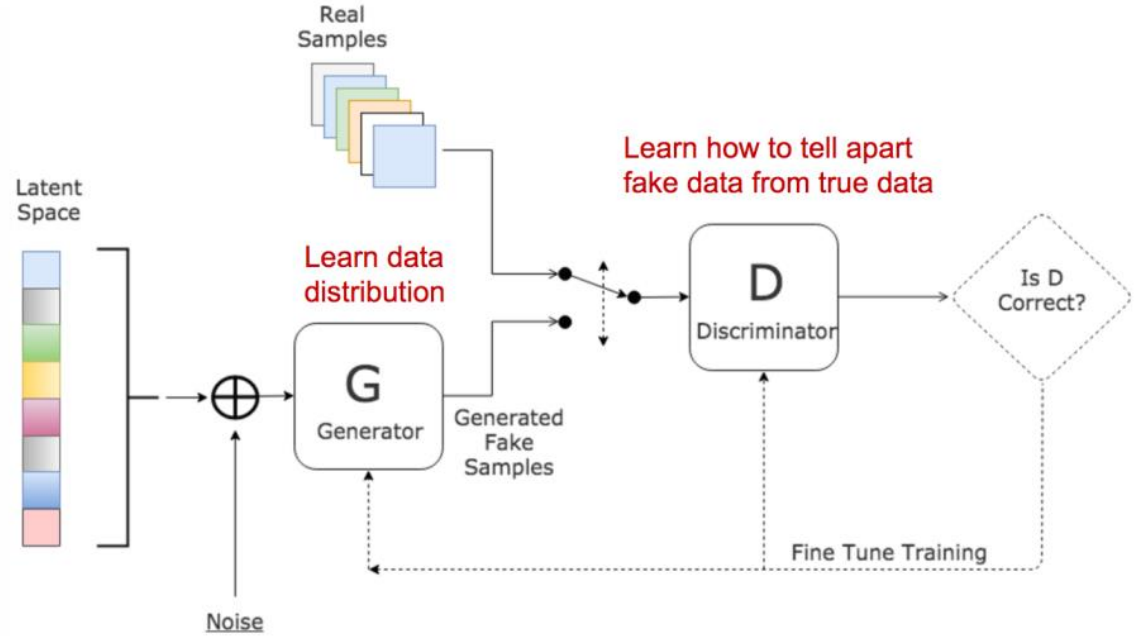

Рисунок 1.2 – Архітектура генеративної змагальної мережі

За останні роки з'явилося багато робіт, присвячених моделям з сімейства GANs. Навіть якщо якість вибірки загалом висока, вони страждають від колапсу мод (mode collapse), що призводить до недостатньої різноманітності вибірки [6]. Змагальний характер навчання робить його нестабільним і тому часто трудомістким в досягненні рівноваги [7].

1.2.2 Варіаційні автокодувальники

Автокодувальники (Autoencoder) – це нейронна мережа, призначена для навчання функції ідентичності в неконтрольований спосіб, щоб реконструювати оригінальні вхідні дані, стискаючи дані в процесі, щоб виявити більш ефективне і стиснене представлення. Ідея виникла у 1980-х роках, а згодом була розвинута у фундаментальній роботі Дж. Хінтона та Р. Салахутдінова [8].

Вона складається з двох мереж – кодувальник і декодувальник (рисунок 1.3). Мережа кодувальника перетворює оригінальний вхідний код високої розмірності у прихований код низької розмірності. Розмір вхідних даних більший за розмір вихідних даних. Декодувальник відновлює дані з коду, ймовірно, з більшими і більшими вихідними шарами. Мережа кодувальників забезпечує зменшення розмірності, так само як використовується аналіз головних компонент або матрична факторизація. Крім того, автокодувальник явно оптимізовано для відновлення даних з кодування. Хороше проміжне представлення не тільки може захопити латентні змінні, але й принести користь повному процесу декомпресії [9].

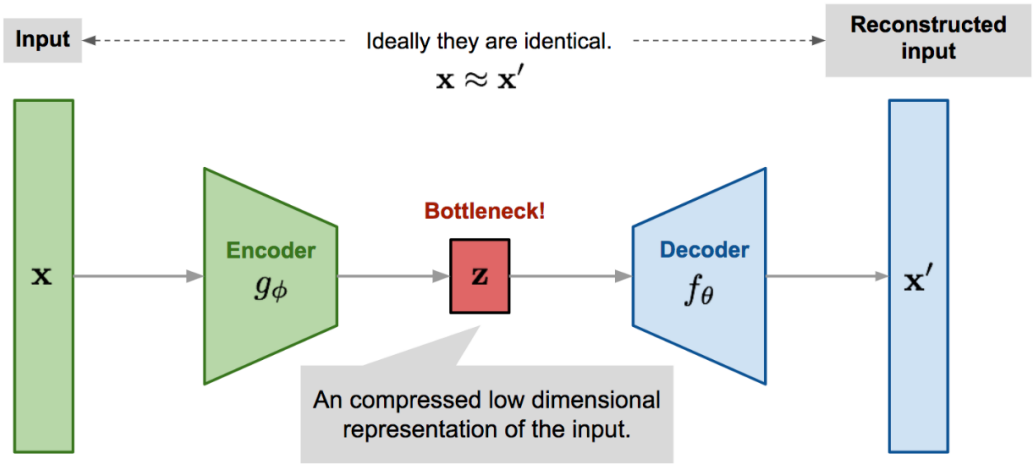

Рисунок 1.3 – Архітектура автокодувальника

Як і стандартний автокодувальник, варіаційний автокодувальник (VAE) – це архітектура, що складається з кодувальника та декодувальника і навчена мінімізувати похибку відновлення між закодованими та декодованими даними та початковими даними [10]. Однак відмінність в тому що замість того, щоб кодувати вхідні дані як одну точку, вони кодуються як розподіл у латентному просторі (рисунок 1.4).

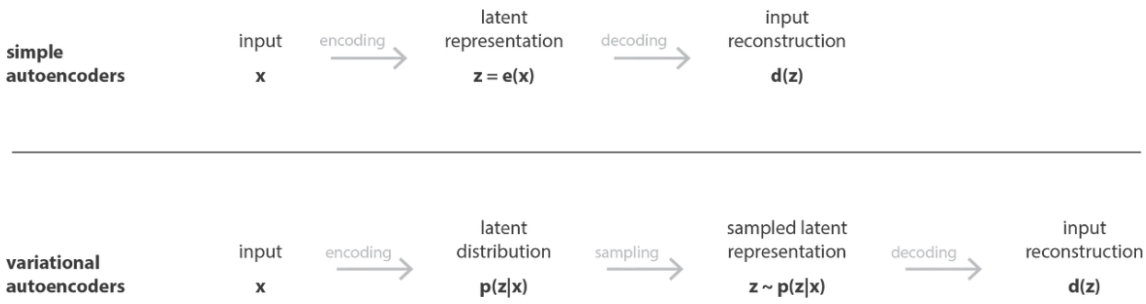

Рисунок 1.4 – Різниця між автокодувальником (детермінованим) і варіаційним автокодувальником (імовірнісним)

Модель навчається за наступними кроками:

– вхідні дані кодуються як розподіл у латентному просторі;

– з цього розподілу вибирається точка з латентного простору;

– вибіркова точка декодується і обчислюється похибка реконструкції;

– похибка реконструкції поширюється мережею.

На практиці кодовані розподіли вибирають нормальними, щоб кодувальник можна було навчити повертати середнє значення та матрицю коваріацій, які описують ці гаусіани. Причина, чому вхідні дані кодуються як розподіл з деякою дисперсією, а не як одна точка, полягає в тому, що це дозволяє дуже природно виразити латентну регуляризацію простору: розподіли, які повертає кодувальник, повинні бути близькими до стандартного нормального розподілу.

Обмеження на латентний простір дозволяють отримати вибірки, близькі до початкового розподілу даних, шляхом простої випадкової вибірки з латентного простору. VAE часто отримують високі значення логарифмічної правдоподібності (log-likelihood), але їм важко створювати нерозмиті вибірки високої якості [9].

1.2.3 Генеративні моделі на основі потоку

Генеративні моделі на основі потоку, загальна архітектура яких представлена на рисунку 1.5, це сімейство моделей, що складається із застосування послідовності інвертованих параметризованих функцій і представлені у роботах [11], [12].

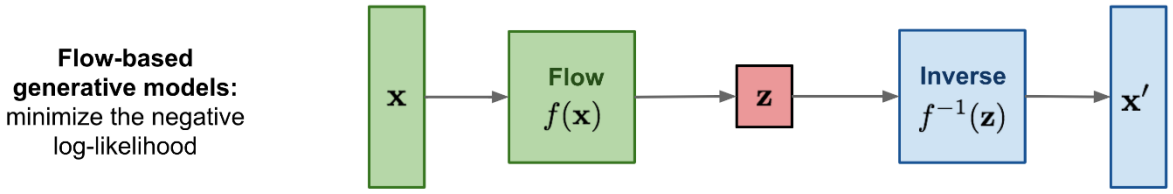

Рисунок 1.5 – Архітектура генеративних моделей на основі потоку

У загальних рисах, моделі на основі потоку застосовують стек інверсійних перетворень до вибірки з попередніх даних, щоб обчислити точну логарифмічну правдоподібність спостережень .

На відміну від GAN та VAE, генеративні моделі на основі потоку безпосередньо максимізують точну log-likelihood ймовірність. Тим не менш, ці моделі, як правило, перевершують інші з точки зору якості вибірки, в першу чергу через складність пошуку ефективних інвертованих архітектур.

Моделі на основі потоків мають дві великі категорії: моделі з нормалізуючими потоками та моделі з авторегресійними потоками, які намагаються підвищити продуктивність базової моделі. Моделі нормалізації потоку забезпечують надійне наближення розподілу. Вони перетворюють простий розподіл на складний, застосовуючи послідовність інверсійних функцій перетворення. Коли перетворення потоку в нормалізуючий потік подається у вигляді авторегресійної моделі, де кожен вимір векторної змінної залежить від попередніх вимірів, ця варіація моделі потоку називається авторегресійним потоком.

Потокові моделі є концептуально привабливими для моделювання складних розподілів, але вони обмежені проблемами оцінювання щільності порівняно з сучасними авторегресійними моделями. Крім того, хоча спочатку потокові моделі можуть замінити GAN, даючи якісні результати, між ними існує значний розрив у обчислювальних витратах на навчання: потокові моделі потребують у кілька разів більше часу, ніж GAN, щоб згенерувати зображення з однаковою роздільною здатністю [13].

1.2.4 Дифузійні моделі

Дифузійні моделі (DM), також широко відомі як дифузійні ймовірнісні моделі, є сімейством згенерованих моделей, які є ланцюгами Маркова навченими за допомогою варіаційного висновку. Метою навчання DM є резервування процесу зашумлення даних, тобто дифузії, для генерації вибірки.

Дифузійні моделі первинно базуються на підході до моделювання молекулярних систем, який називається динаміки Ланжевена. Вперше вони були представлені Дж. Соль-Дікштейном та іншими у 2015 році [15], і поступово стали серйозною альтернативою традиційним генеративним моделям, таким як GAN або VAE, отримавши видатні результати в перетворенні тексту в зображення (роботи [16], [17], [18]). Вони визначають марковський ланцюжок кроків дифузії для повільного додавання випадкового шуму до даних, а потім вчяться повертати процес дифузії назад, щоб побудувати бажані вибірки даних з шуму (рисунок 1.6).

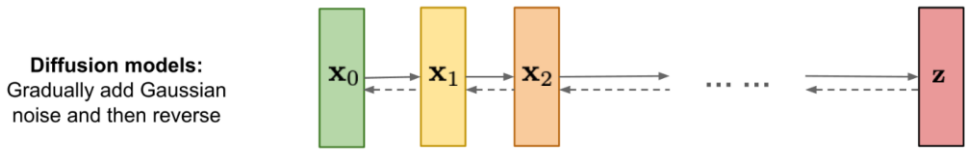

Рисунок 1.6 – Концепт дифузійних моделей

Знаковою роботою стала імовірнісна дифузійна модель з усуненням шуму (DDPM), яка була опублікована у 2020 році [14] і викликала експоненціально зростаючий інтерес у спільноті генеративних моделей. Поява DDPM в основному пов'язана з двома ранніми розробками: генеративними моделями на основі оцінок (SGM) [19], які досліджуються у 2019 році, та дифузійними імовірнісними моделями (DPM) [20], що з'явилися ще у 2015 році.

DPM є першою роботою, яка моделює розподіл ймовірностей шляхом оцінки реверсії дифузійного ланцюга Маркова, який відображає дані до простого розподілу. Зокрема, DPM визначає прямий процес (виведення), який перетворює складний розподіл даних на набагато простіший, а потім вивчає відображення шляхом зворотного процесу дифузії. Експериментальні результати на багатьох наборах даних показують ефективність DPM при оцінці складного розподілу даних.

Методи покращення генеративних моделей на основі оцінок також досліджувалися в роботах [19] та [21] пропонується збурювати дані випадковим гаусівським шумом різної величини.

Використовуючи градієнт логарифмічної густини ймовірності як функцію оцінки [19], SGM генерує вибірки в напрямку зменшення рівня шуму і навчає модель, оцінюючи функції оцінки для зашумленого розподілу даних. Незважаючи на різну мотивацію, SGM має схожу мету оптимізації з DDPM під час навчання, що також обговорюється в роботі [14], де показано, що DDPM за певної параметризації еквівалентний SGM під час навчання.

DDPM визначаються як параметризований ланцюг Маркова, який генерує зображення з шуму в межах скінченних переходів під час виведення. Під час навчання ядра переходів вивчаються у зворотному напрямку до збурення природних зображень шумом, де шум додається до даних на кожному кроці і оцінюється як ціль оптимізації.

Модель латентної дифузії (LDM) запускає процес дифузії в латентному просторі, а не в піксельному, що робить вартість навчання нижчою, а швидкість виведення швидшою [22].

1.3 Еволюція моделей генерацій зображень за текстовим описом

Ідея синтезувати зображення за описом виникла з ранньої роботи alignDRAW [23]. Однак якість зображень була дуже низькою, а згенеровані сцени та об'єкти ледве можна було розпізнати. За цим послідували п'ять років невеликих покращень (рисунок 1.7), спричинених розвитком GAN, з такими моделями, як Attentional Generative Adversarial Network (AttnGAN) [24], Dynamic Memory Generative Adversarial Networks (DM-GAN) [25] або Fusion Generative Adversarial Network (DF-GAN) [26]. Хоча частина змісту підписів почала відображатися, зображення все ще не були реалістичними, за винятком обмежених і простих наборів даних.

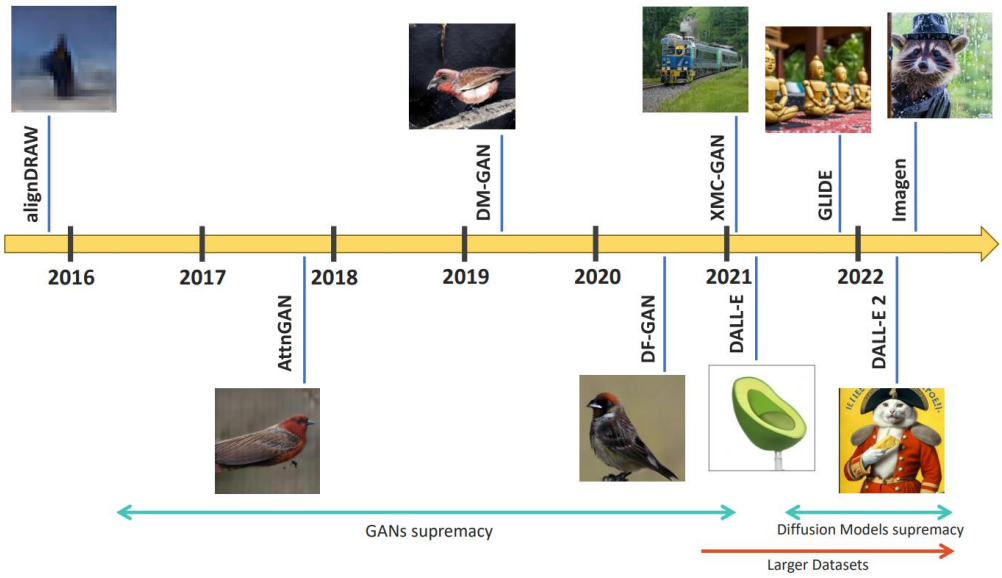

Рисунок 1.7 – Хронологія створення генеративних моделей «текст-в-зображення»

Інтеграція контрастного навчання в конвеєр і особливо збільшення розміру набору даних дозволило Cross-Modal Contrastive Generative Adversarial Network (XMC-GAN) [27] створювати кращі зображення з більш чіткими сценами. Аналогічно, автори DALL-E [28] показали, що подальше збільшення набору даних до 250 мільйонів пар зображення-текст може уможливити навчання з «нульового пострілу» (zero-shot learning). Таким чином, модель DALL-E здатна змішувати різні об'єкти, концепції та місця для створення незвичайних зображень, наприклад, крісло з авокадо. Однак DALL-E не використовує GAN, а натомість використовує Vector Quantised-Variational AutoEncoder (VQ-VAE) [29], а також два трансформери. Хоча результати DALL-E були вражаючими, нова категорія генеративних моделей продемонструвала, що це був лише початок.

Того ж року, коли була опублікована робота DALL-E, у роботі [30] показали, що дифузійні моделі можуть перевершити GAN у створенні зображень, обумовлених класом. Потім, в роботі [31] про Guided Language to Image Diffusion for Generation and Editing(GLIDE) запровадили дифузійні моделі для синтезу текст-зображення і отримали зображення вищої якості, ніж DALL-E, навчаючись на тому ж наборі даних. Автори GLIDE використовують мовну модель трансформеру для вбудовування текстових описів зображень, а потім дифузійні моделі за умови вбудовування для отримання зображень 256×256. Але під час розробки цієї дисертації з різницею в місяць вийшли дві великі роботи: DALL-E 2 [32] та Imagen [33].

DALL-E 2 дуже відрізняється від своєї попередньої версії. Подібно до GLIDE, він декодує вкладання (embeddings) для створення зображень. Однак вкладання надходять від Contrastive Language-Image Pre-Training (CLIP) [34], моделі мови зору, яка вивчає текст-зображення репрезентації. Два кодувальники CLIP створюють вкладання для зображень і текстів відповідно, де косинус подібністі між двома вбудовуваннями, що походять з однієї і тієї ж пари «зображення-підпис», вища, ніж між

некорельованими вбудовуваннями. DALL-E 2 використовує ці попередньо навчені кодувальники і вивчає попередню модель для перетворення текстових вбудовувань CLIP у вбудовування CLIP-зображень. Вони також використовують моделі каскадної дифузії [35] для підвищення дискретності зображень з 64 × 64 до 1024 × 1024.

Imagen, з іншого боку, більше схожий на GLIDE, але замість того, щоб навчати з нуля мовної моделі, автори повторно використовують велику заморожену модель трансформера, навчену на великому тільки текстовому корпусі, який називається моделлю T5 [36]. Вони демонструють, що збільшення розміру мовної моделі призводить до більших покращень, ніж збільшення розміру моделі дифузії. Автори стверджують, що їхня модель перевершує модель DALL-E 2, оскільки вони отримують нижчий критерію для оцінки продуктивності моделей, що генерують зображення, на валідаційному наборі даних Microsoft Common Objects in Context (MS-COCO) [37].

1.4 Постановка задачі

Завдання кваліфікаційної роботи – теоретично і практично порівняти генеративну змагальну мережу і дифузійні моделі для генерації зображень з текстового опису в різних аспектах, одним з яких є дослідження якості і можливостей цих моделей, особливо в умовах обмежених ресурсів.

Для успішного виконання цієї роботи необхідно:

– вивчити літературу та інтернет-джерела за темою генеративних моделей;

– описати математичні моделі;

– побудувати і навчити моделі на однаковому наборі даних;

– проаналізувати отримані результати для кожної моделі;

– зробити висновки про порівняльний аналіз моделей.

2 ОГЛЯД ГЕНЕРАТИВНИХ МОДЕЛЕЙ

2.1 Генеративні змагальні мережі

Концепція генеративних змагальних мереж (Generative Adversarial Networks, GAN) включає два конкуруючі компоненти, а саме генератор G і дискримінатор D, які беруть участь у мінімаксній грі для двох гравців. Завданням дискримінатора є розрізнення між реальними навчальними даними та синтетичними зображеннями, тоді як генератор має на меті обдурити дискримінатор, створюючи фальшиві зображення (рисунок 2.1).

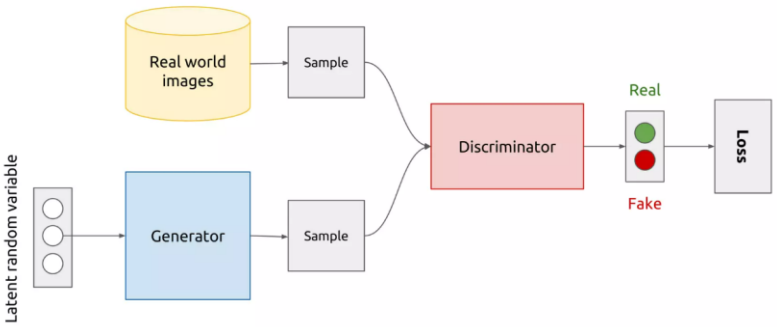

Рисунок 2.1 – Концепт генеративних змагальних мереж

Зокрема, дискримінатор і генератор грають у гру на $V(D,G)$ як наведено у формулі 2.1.

$$\min_G \max_G V(D,G) = \mathbb{E}_{x\sim p_{data}(x)}[\log D(x)] + \\ +\mathbb{E}_{z\sim p_z(z)}\left[\log\left(1 - D\left(G(z)\right)\right)\right], \tag{2.1}$$

де x – реальне зображення з навчального набору даних;

z –випадковий вектор шуму, що використовується генератором;

$D(x)$ – ймовірність того, що реальне зображення з навчального набору даних є реальним, згідно з дискримінатором;

$D\big(G(z)\big)$ – ймовірність того, що згенероване зображення є реальним, згідно з дискримінатором.

Генератор G неявно визначає розподіл ймовірностей p_g як розподіл вибірок $G(z)$, отриманих при $z \sim p_z$. Як видно з формули глобальний оптимум досягається, коли p_g дорівнює p_{data}. Більше того, якщо G і D мають достатню потужність, p_g збігається до p_{data} за умови виконання м'яких умов. Під час початкового навчання D має тенденцію відкидати погані зразки від G з високою впевненістю. Як практичне рішення, було помічено, що максимізація $log(D(G(z)))$ дає кращі результати для генератора порівняно з мінімізацією $log\left(1 - D\big(G(z)\big)\right)$[3].

Для навчання GAN необхідно виконати певну кількість навчальних ітерацій, під час яких по черзі навчається дискримінатор (рисунок 2.2), а потім генератор (рисунок 2.3).

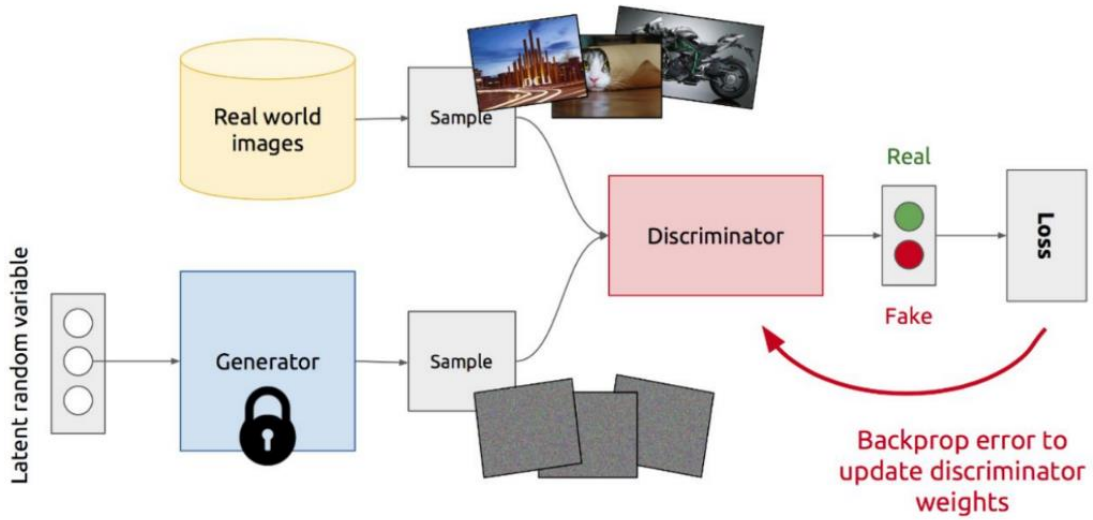

Рисунок 2.2 – Концепт тренування дискримінатора

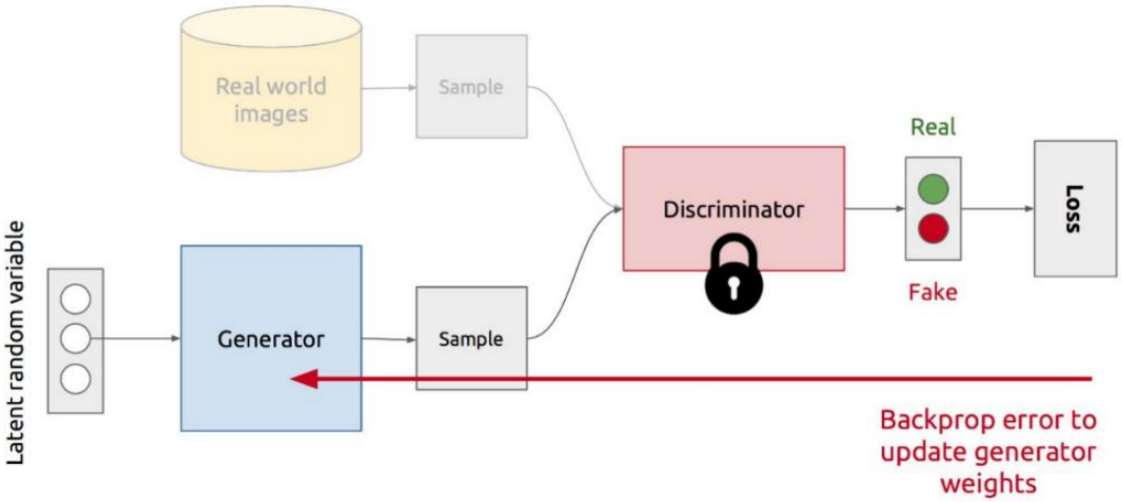

Рисунок 2.3 – Концепт тренування генератора

На кожному кроці з попереднього шуму $p_g(z)$ вибирається пакет з m зразків шуму $\{z(1),\ldots,z(m)\}$, а інший міні-пакет з m прикладів $\{x(1),\ldots,x(m)\}$ з розподілу, що генерує дані $p_{data}(x)$. Дискримінатор оновлюється шляхом зростання його стохастичного градієнта, що передбачає обчислення градієнта функції втрат відносно його параметрів θ_d. Параметри оновлюються по градієнту (висхідним шляхом) у формулі 2.2.

$$\nabla_{\theta_d} \frac{1}{m} \sum_{i=1}^{m} \left[\log D\left(x^{(i)}\right) + \log\left(1 - D\left(G(z^{(i)})\right)\right) \right]. \quad (2.2)$$

Перший доданок є логарифмом виходу дискримінатора на реальних навчальних прикладах, а другий є виходом дискримінатора на синтетичних зображеннях, згенерованих генератором. Після оновлення дискримінатора вибирається ще одна партія з m відліків шуму, і генератор оновлюється за спаданням стохастичного градієнта у формулі 2.3.

$$\nabla_{\theta_g} \frac{1}{m} \sum_{i=1}^{m} \left[\log\left(1 - D\left(G(z^{(i)})\right)\right) \right]. \quad (2.3)$$

2.1.1 Додання умови до генерації в GAN

GAN можна розширити, щоб генерувати вихідні дані умовно, коли генеративна модель навчається генерувати нові приклади з вхідної області, маючи додаткові вхідні дані (наприклад, текст чи клас), які обумовлюють випадковий вектор з латентного простору. В умовному GAN (Conditional GAN) дискримінатор також є умовним, тобто йому надається як вхід зображення, яке є реальним або фальшивим, так і додатковий вхідний сигнал (умова). Приклад архітектури зображено на рисунку 2.4. За допомогою такої умови дискримінатор може навчитися класифікувати вхідні дані відповідно до свого класу, що дає генератору вказівку створювати зразки з цього класу, щоб обдурити дискримінатор [38].

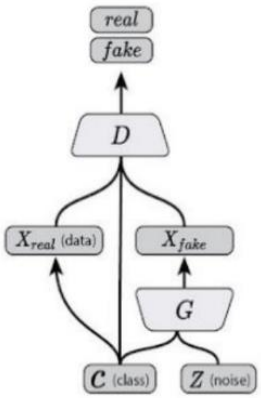

Рисунок 2.4 – Приклад архітектури моделі Conditional GAN

Можна використовувати GAN для перекладу з тексту на зображення або з зображення на текст, навчаючи його на прикладах з домену, наприклад, на фотографіях. Це має широкий спектр застосувань, включно з передачею стилю, додаванням кольору до зображень і зміною пори року або часу доби на фотографіях.

2.1.2 Генеративна змагальна мережа з глибокими згортками (DC-GAN)

GAN з глибокими згортками (DC-GAN) навчає мережу, використовуючи текстову інформацію, що зберігається на рівні символів за допомогою гібридної згортково-рекурентної нейронної мережі. Це означає, що прямий висновок здійснюється генератором і дискримінатором, обидва з яких покладаються на текстові властивості.

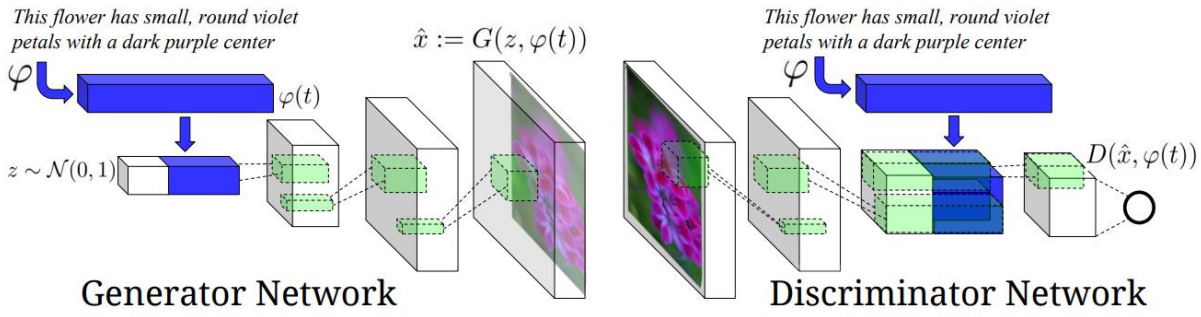

Рисунок 2.5 – Архітектура DC-GAN

Для того, щоб створити зображення, генераторна мережа G спочатку вибирає вектор шуму z з нормального розподілу із середнім значенням 0 і дисперсією 1, тобто $z \in R^z \sim N(0,1)$. Потім текстовий запит t кодується за допомогою текстового кодувальника φ, який стискається до меншої розмірності 128 за допомогою повністю зв'язаного шару з активацією leaky ReLU. Вектор шуму z і стиснений текстове вбудовування t об'єднуються перед подачею на генератор. Синтетичне зображення x', отримане за рівнянням, що є виходом генератора, генерується за рівнянням 2.4

$$x' \leftarrow G\big(z, \varphi(t)\big). \tag{2.4}$$

Дискримінаторна мережа D виконує згортку stride2 з просторовою пакетною нормалізацією, після чого відбувається активація leaky ReLU. При подальшій ректифікації розмірність вкладеного опису *t* зменшується в окремому повністю зв'язаному шарі.

Вбудовування опису просторово повторюється і об’єднується за глибиною, коли просторовий розмір дискримінатора становить 4 на 4. Потім обчислюється оцінка дискримінатора шляхом виконання згортки 1×1, ректифікація і згортки 4×4. На кожному шарі згортки виконується пакетна нормалізація [2].

2.1.3 Стекова генеративна змагальна мережа (StackGAN)

Оскільки розподіл даних реальних зображень і розподіл моделі можуть не перетинатися у високорозмірному піксельному просторі, навчити мережу генерувати зображення високої роздільної здатності на основі текстового опису за допомогою стандартної архітектури GAN досить складно.

Щоб вирішити цю проблему, Стекова генеративна змагальна мережа (StackGAN) використовує Stage-I GAN для створення зображень з низькою роздільною здатністю, а потім Stage-II GAN, яка спирається на Stage-I GAN для створення зображень з високою роздільною здатністю (наприклад, 256x256) на основі результатів Stage-I GAN і текстових описів.

На основі вхідного текстового опису на першому кроці створюється зображення з низькою роздільною здатністю шляхом малювання простої форми об'єкта і розфарбовування його основними кольорами. Вектор шуму використовується для випадкового створення макета фону. На другому етапі виправляються недоліки зображення низької роздільної здатності та заповнюються особливості об'єкта, знову ж таки, на основі текстового опису. Архітектура такого рішення зображена на рисунку 2.6

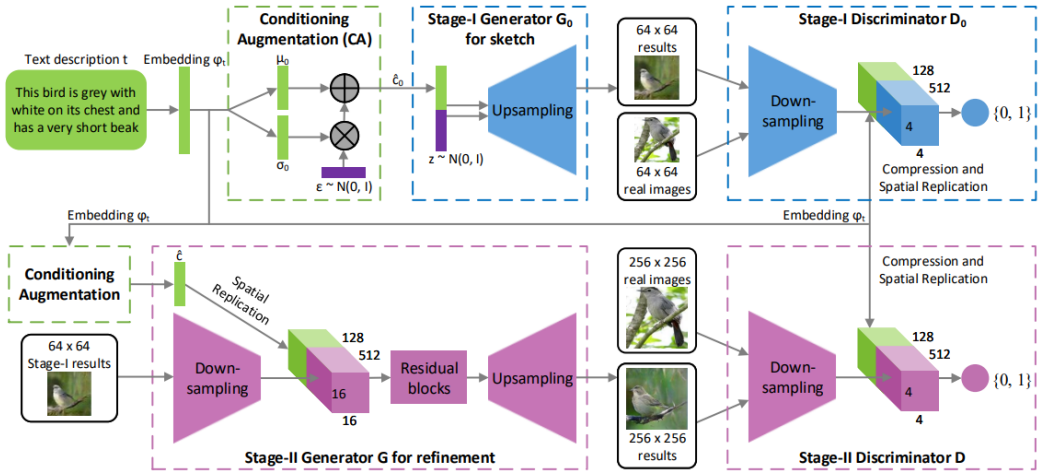

Рисунок 2.6 – Архітектура StackGAN

На першому етапі StackGAN навчаються дискримінатор D_0 (максимізуючи $\mathcal{L}_{D_0}$ в формулі 2.5) та генератор G_0 (мінімізуючи $\mathcal{L}_{G_0}$ в формулі 2.6):

$$\mathcal{L}_{D_0} = \mathbb{E}_{(I_0,t)\sim p_{\text{data}}}[\log D_0(I_0, \varphi_t)] + \\ + \mathbb{E}_{z\sim p_z, t\sim p_{\text{data}}}\left[\log\left(1 - D_0(G_0(z, \hat{c}_0), \varphi_t)\right)\right], \tag{2.5}$$

$$\mathcal{L}_{G_0} = \mathbb{E}_{z\sim p_z, t\sim p_{\text{data}}}\left[\log\left(1 - D_0(G_0(z, \hat{c}_0), \varphi_t)\right)\right] + \\ + \lambda D_{KL}\left(\mathcal{N}\left(\mu_0(\varphi_t), \Sigma_0(\varphi_t)\right) \parallel \mathcal{N}(0, I)\right), \tag{2.6}$$

де I_0 і t – реальне зображення, текстовий опис з справжнього розподілу даних p_{data};

φ_t – вбудовування текстового опису, згенерований попередньо навченим кодувальником;

z – вектор шуму, випадково обраний з p_z;

$\hat{c}_0$ – Гаусова змінна обумовлення для вбудовування тексту, обрана з $\mathcal{N}\left(\mu_0(\varphi_t), \Sigma_0(\varphi_t)\right)$ для того, щоб зафіксувати значення φ_t з варіаціями;

λ – параметр регуляризації.

В Stage-II GAN дискримінатор D і генератор G навчаються шляхом альтернативної максимізації $\mathcal{L}_D$ в рівнянні та мінімізації $\mathcal{L}_G$ в формулах 2.7 і 2.8 відповідно [39]:

$$\mathcal{L}_{D_0} = \mathbb{E}_{(I,t)\sim p_{\text{data}}}[\log D(I_0, \varphi_t)] + \\ + \mathbb{E}_{s_0\sim p_{G_0}, t\sim p_{\text{data}}}\left[\log\left(1 - D(G(s_0, \hat{c}), \varphi_t)\right)\right], \tag{2.7}$$

$$\mathcal{L}_G = \&\mathbb{E}_{s_0\sim p_{G_0}, t\sim p_{\text{data}}}\left[\log\left(1 - D(G(s_0, \hat{c}), \varphi_t)\right)\right] + \\ + \lambda D_{KL}\left(\mathcal{N}\left(\mu(\varphi_t), \Sigma(\varphi_t)\right) \| \mathcal{N}(0, I)\right), \tag{2.8}$$

де $s_0 = G_0(z, \hat{c}_0)$ – обумовлення на результат з низькою роздільною здатністю;

$\hat{c}$ – гаусова латентна змінна з того самого навченого кодувальника що й $\hat{c}_0$.

В результаті це дало кращі результати ніж стандартний GAN (рисунок 2.7)

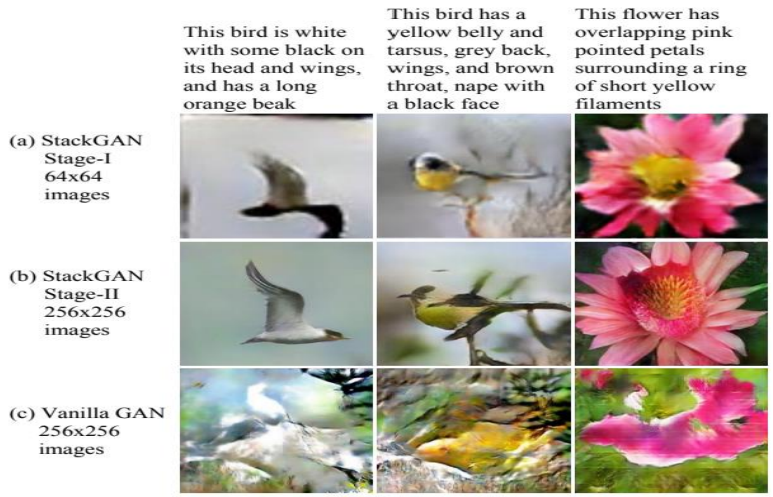

Рисунок 2.7 – Порівняння генерації зображень роздільності 256×256 стандартним GAN та StackGAN

2.1.4 Генеративні змагальні мережі з увагою (AttnGan)

Для того, щоб покращити якість згенерованих зображень та їхню кореляцію з підписом до зображення, AttnGAN включає в себе крос-модальну увагу та спеціальні додаткові втрати – Deep Attentional Multimodal Similarity Model (DAMSM). Модель складається з трьох дискримінаторів і генератора (рисунок 2.8). На різних етапах створення зображення дискримінатори забезпечують зворотний зв'язок для стабілізації навчання. За допомогою механізму уваги різним словам у підписі надається різний рівень важливості для різних частин згенерованого зображення.

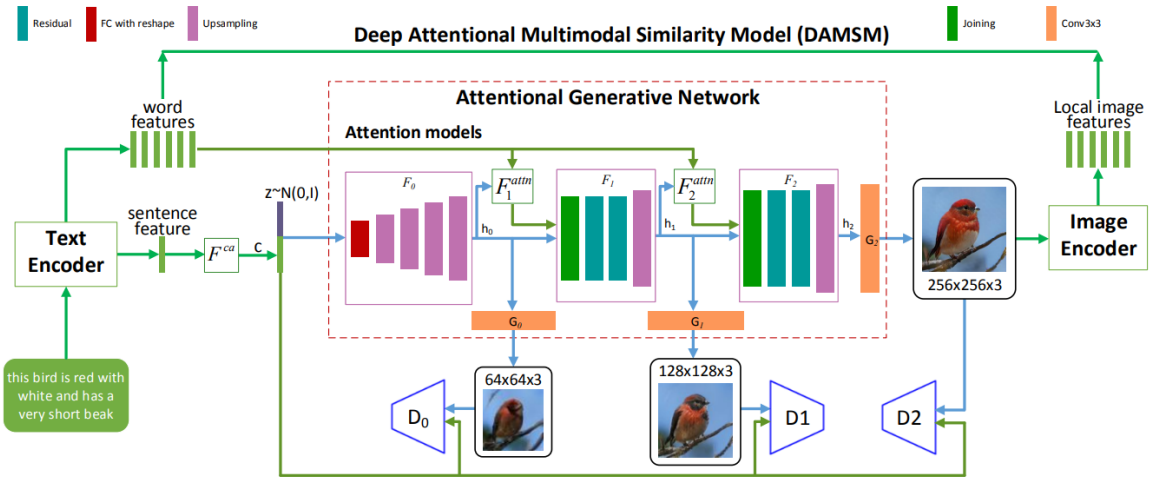

Рисунок 2.8 – Архітектура AttnGAN

Як видно з рисунка AttnGAN має m генераторів $(G_0, G_1, \ldots, G_{m-1})$ на вхід до яких подаються приховані стани $(h_0, h_1, \ldots, h_{m-1})$ і генерують образи від малого до великого масштабу $(\widehat{x_0}, \widehat{x_1}, \ldots, \widehat{x_{m-1}})$ в формулах 2.9, 2.10, 2.11.

$$h_0 = F_0(z, F^{ca}(\bar{e})), \tag{2.9}$$

$$h_i = F_i\left(h_{i-1}, F_i^{attn}(e, h_{i-1})\right), \text{ для } i = 1,2,\ldots,m-1, \tag{2.10}$$

$$\hat{x}_i = G_i(h_i), \tag{2.11}$$

де z – вектор шуму обраний з нормального розподілу;

$\bar{e}$ – глобальний вектор речення;

e – матриця векторів слів;

F^{ca} – метод Conditioning Augmentation, що перетворює вектор $\bar{e}$ на вектор обумовлення;

F_i^{attn} – модель уваги на етапі i.

Функція втрат мережі описана у формулі 2.12.

$$\mathcal{L} = \mathcal{L}_G + \lambda \mathcal{L}_{DAMSM}, \tag{2.12}$$

де $\mathcal{L}_G = \sum_{i=0}^{m-1} \mathcal{L}_{G_i}$;

λ – параметр регуляризації.

Генератор G_i в AttnGAN пов'язаний з дискримінатором D_i на і-му етапі. Функцію втрат для G_i можна виразити як в формулі 2.13.

$$\mathcal{L}_{G_i} = \underbrace{-\frac{1}{2}\mathbb{E}_{\hat{x}_i \sim p_{G_i}}[\log(D_i(\hat{x}_i)]}_{\text{unconditional loss}} \underbrace{-\frac{1}{2}\mathbb{E}_{\hat{x}_i \sim p_{G_i}}[\log(D_i(\hat{x}_i, \bar{e})]}_{\text{conditional loss}}. \tag{2.13}$$

А для D_i вона має такий вигляд як в формулі 2.14.

$$\begin{aligned}\mathcal{L}_{D_i} = &\underbrace{-\frac{1}{2}\mathbb{E}_{x_i \sim p_{\text{data}\ i}}[\log D_i(x_i)] - \frac{1}{2}\mathbb{E}_{\hat{x}_i \sim p_{G_i}}[\log(1 - D_i(\hat{x}_i)]}_{\text{unconditional loss}} - \\ &\underbrace{-\frac{1}{2}\mathbb{E}_{x_i \sim p_{\text{data}\ i}}[\log D_i(x_i, \bar{e})] - \frac{1}{2}\mathbb{E}_{\hat{x}_i \sim p_{G_i}}[\log(1 - D_i(\hat{x}_i, \bar{e})]}_{\text{conditional loss}}.\end{aligned} \tag{2.14}$$

DAMSM створена для вивчення спільного простору вбудовування зображень і тексту, які вирівняні і порівнюються безпосередньо. Багатомасштабна втрата співставлення (multi-scale matching loss) та подвійна втрата уваги (dual attention loss) – це кмопоненти функції втрат. Перша порівнює, наскільки схожими є зображення і текстові ознаки на різних рівнях абстракції, від базових візуальних елементів до складних семантичних ідей. Втрати визначаються шляхом усередненого на всіх рівнях абстракції від'ємного косинуса схожості між зіставленими частинами зображення і тексту. Другий використовується для того, щоб переконати модель враховувати і картинку, і текст при визначенні вбудовувань.

Втрати обчислюються як від'ємна сума перехресних ентропійних втрат між вагами уваги та істинними анотаціями як для зображення, так і для тексту [24].

2.1.5 Генеративна змагальна мережа з глибоким синтезом (DF-GAN)

В цій архітектурі пропонується (рисунок 2.9):

– новий одноетапний блок перетворення тексту в зображення, який створює зображення з високою роздільною здатністю безпосередньо, без зв'язків між кількома генераторами;

– новий дискримінатор, який покращує семантичну узгодженість між текстом і зображенням без додавання додаткових мереж і складається з штрафу за усвідомлений градієнт (Matching Aware Gradient Penalty, MA-GP) та одностороннього виводу (One-Way Output);

– новий блок глибокого злиття тексту і зображення (Deep text-image Fusion Block, DFBlock), який більш повно поєднує текстові та візуальні функції, повністю інтегрує текстові та візуальні характеристики, інтенсифікує процес злиття.

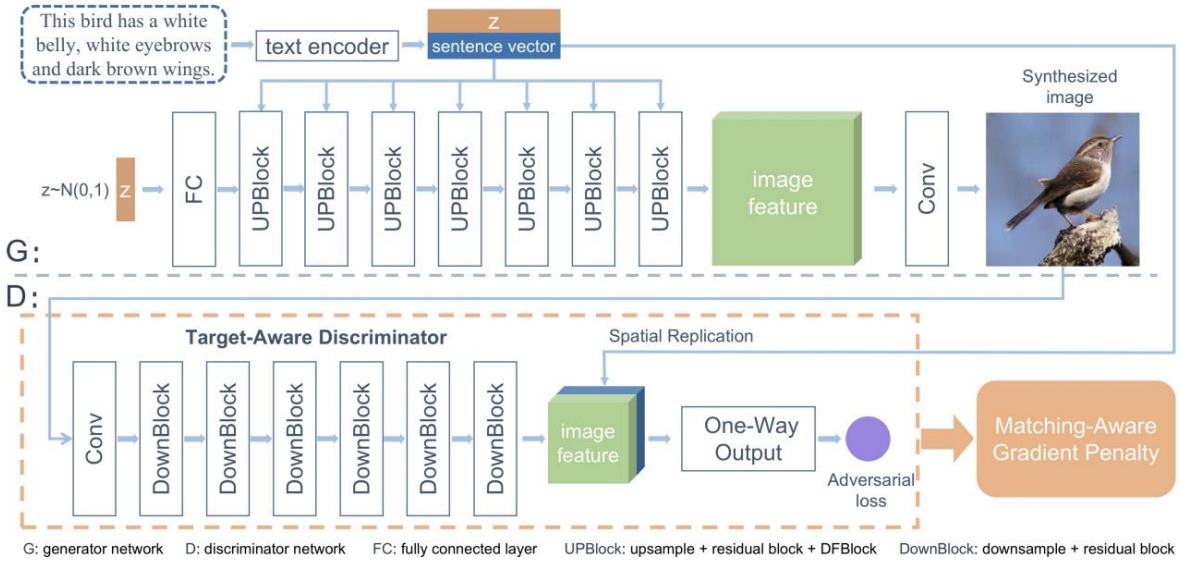

Рисунок 2.9 – Архітектура DF-GAN

Як показано на рисунку 2.9, DF-GAN складається з генератора, дискримінатора та попередньо навченого текстового кодувальника. Щоб забезпечити різноманітність створених зображень, генератор потребує два входи: вектор фраз, закодований текстовим кодувальником, і вектор шуму, вибраний з гауссівського розподілу. Спочатку повністю зв'язний шар отримує вектор шуму і змінює його форму. Для того, щоб підвищити дискретизацію характеристик зображення, застосовується послідовність UPB-блоків (UPBlocks). Щоб об'єднати текстові та графічні елементи при створенні зображення, UPBlock складається з шару підвищення дискретизації, залишкового блоку та DFBlocks. DFBlock складається з двох афінних шарів, двох активаційних шарів ReLU та шару згортки. Шар згортки перетворює характеристики зображення на власне зображення для фінішної обробки.

За допомогою серії DownBlocks дискримінатор перетворює зображення в ознаки зображення. Потім вектор фраз копіюється, а характеристики зображень об'єднуються з ним. Щоб оцінити візуальну реалістичність та семантичну зв'язність вхідних даних, передбачається змагальна втрата. Дискримінатор спонукає генератор створювати

зображення з кращою якістю та семантичною зв'язністю між текстом і зображенням, відокремлюючи створені зображення від справжніх зразків.

Кодувальник тексту являє собою двонаправлену мережу довготривалої короткочасної пам'яті, яка витягує семантичні вектори з текстового опису.

One-Way Output є методом, коли дискримінатор об'єднує вектор фраз та ознаку зображення перед тим, а потім виводить ліше одну втрута через 2 шари згортки. Використовуючи цей підхід, можна зробити так, щоб єдиний градієнт одразу спрямовувався до цільових точок даних (фактичних і співпадаючих), що оптимізує і прискорює збіжність генератора.

Формулювання функцій втрат для моделі з MA-GP описано в формулах 2.15 і 2.16 [26].

$$\begin{aligned} L_D = &-\mathbb{E}_{x\sim\mathbb{P}_r}\left[\min\left(0,-1+D(x,e)\right)\right]- \\ &-(1/2)\mathbb{E}_{G(z)\sim\mathbb{P}_g}\left[min\left(0,-1-D(G(z),e)\right)\right]- \\ &-(1/2)\mathbb{E}_{x\sim\mathbb{P}_{mis}}\left[min\left(0,-1-D(x,e)\right)\right]+ \\ &+k\mathbb{E}_{x\sim\mathbb{P}_r}[(\|\nabla_x D(x,e)\|+\|\nabla_e D(x,e)\|)^p], \end{aligned} \tag{2.15}$$

$$L_G = -\mathbb{E}_{G(z)\sim\mathbb{P}_g}[D(G(z),e)], \tag{2.16}$$

де z – вектор шуму, отриманий з гаусівського розподілу;

e – вектор речень;

k, p – гіперпараметри градієнтного штрафу;

$\mathbb{P}_r$ – розподіл реальних даних;

$\mathbb{P}_g$ – розподіл згенерованих даних;

$\mathbb{P}_{mis}$ – розподіл невідповідності даних.

2.1.6 Генеративна змагальна CLIP (GALIP)

Архітектура генеративної змагальної CLIP (GALIP), включає модель CLIP як в дискримінатор, так і в генератор. Кодувальник тексту CLIP, дискримінатор на основі CLIP і генератор з підтримкою CLIP утворюють систему GALIP, яка показана на рисунку 2.10.

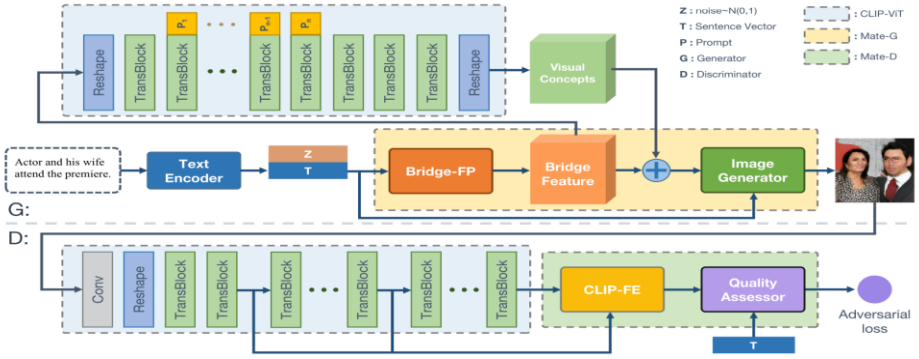

Рисунок 2.10 – Архітектура GALIP

Попередньо навчений текстовий кодувальник CLIP спочатку обробляє опис тексту, створюючи глобальний вектор речень *T*. У системі GAN генератор з підтримкою CLIP і дискримінатор на основі CLIP використовують цей вектор після цього. Предиктор ознак Bridge-FP (Bredge Feature Predictor), предиктор підказок і генератор зображень - це три основні модулі, з яких складається генератор Mate-G, що є частиною генератора на основі CLIP. Вектор фрази *T* і вектор шуму z, вибраний з гаусівського розподілум, є входами генератора.

Bridge-FP оброблює текст і вектор шуму для прогнозування функції моста для CLIP-ViT. Для адаптації завдання, до блоків трансформаторів (TransBlock) у CLIP-ViT додається багато текстових підказок. Використовуючи задані візуальні концепції, характеристики моста, слова і вектори шуму, генератор зображень створює зображення.

Заморожений CLIP-ViT і дискримінатор Mate-D утворюють дискримінатор на основі CLIP. CLIP-ViT використовує шар згортки і блоки трансформації для перетворення зображення в ознаки зображення, тоді як екстрактор ознак CLIP (CLIP-FE) в Mate-D збирає ознаки зображення з різних шарів у CLIP-ViT. Після цього оцінювач якості Mate-D витягує корисні візуальні ознаки із зібраних CLIP-ознак, щоб спрогнозувати втрати супротивника на основі ознак і векторів речень.

Дискримінатор розрізняє синтетичні та реальні зображення, що мотивує генератор створювати більш ефективні зразки. Для стабілізації навчального процесу змагального навчання, використовується втрату петлі та односторонній дискримінатор, тому цільова функція має вигляд як зображено у формулі 2.17 і 2.18

$$\begin{aligned} L_D = &-\mathbb{E}_{x\sim\mathbb{P}_r}\big[min\big(0,-1+D(C(x),e)\big)\big]- \\ &-(1/2)\mathbb{E}_{G(z,e)\sim\mathbb{P}_g}\Big[\min\Big(0,-1-D\big(C\big(G(z,e)\big),e\big)\Big)\Big] \\ &-(1/2)\mathbb{E}_{x\sim\mathbb{P}_{mis}}[min(0,-1-D(C(x),e))]+ \\ &+k\mathbb{E}_{x\sim\mathbb{P}_r}\big[\big(\|\nabla_{C(x)}D(C(x),e)\|+\|\nabla_e D(C(x),e)\|\big)^p\big], \end{aligned} \tag{2.17}$$

$$L_G = -\mathbb{E}_{G(z,e)\sim\mathbb{P}_g}[D(C(G(z,e)),e)]-\lambda\mathbb{E}_{G(z,e)\sim\mathbb{P}_g}[S(G(z,e),e)], \tag{2.18}$$

де z – вектор шуму, отриманий з гаусівського розподілу;

G – генератор розширений моделлю CLIP;

C – заморожений CLIP-ViT в дискримінатор на основі CLIP;

D – дискримінатор Mate-D;

S – косинус подібності між закодованими візуальними та текстовими ознаками CLIP;

k, p – гіперпараметри градієнтного штрафу;

λ – коефіцієнт схожості між текстом і зображенням;

$\mathbb{P}_r$ – розподіл реальних даних;

$\mathbb{P}_g$ – розподіл згенерованих даних;

$\mathbb{P}_{mis}$ – розподіл невідповідності даних.

2.1.7 Переваги і недоліки GAN

Поява генеративних змагальних мереж викликала значний інтерес і призвела до значних дослідницьких зусиль у галузі синтезу зображень. Це зробило можливим навчання генеративних моделей для зображень як без нагляду (unsupervised), так і з наглядом (supervised).

GANs можуть генерувати високоякісні зображення, які непогано відповідають текстовим описам, наданим на вході. Їх можна навчати на різних текстових джерелах, таких як підписи до зображень, описи природною мовою і навіть структуровані дані. GAN можна навчити генерувати вказані стилі та типи зображень певної області, що дає змогу створювати різноманітні зображення. Вони не вимагають явного маркування навчальних даних, що може бути корисним, коли маркованих даних мало або їх важко отримати.

Іншими ефективними використанням цих мереж є:

– генерація даних, які схожі, але не ідентичні навчальним даним, що може бути корисно в таких додатках, як доповнення даних, де метою є збільшення розміру набору навчальних даних;

– неконтрольоване навчання, коли метою є виявлення основних закономірностей у даних без попереднього знання міток;

– передавальне навчання (transfer learning), коли попередньо навчений GAN може бути точно налаштований на новому наборі даних для конкретного завдання;

Хоча вони і показали перспективні результати у створенні високоякісних зображень з текстових описів, але вони також мають свої обмеження. Серед недоліків слід зазначити наступні:

– навчання GAN може бути складним і трудомістким, оскільки генератор і дискримінатор постійно конкурують з іншими, що може зробити навчання нестабільним і повільним;

– мережі часто потребують великої кількості навчальних даних і значних обчислювальних ресурсів для отримання хороших результатів;

– генерування зображень зі складних текстових описів може бути складним завданням, а отримані зображення не завжди можуть відповідати оригінальному текстовому вводу;

– можуть створювати зображення з артефактами або візуальними невідповідностями, особливо в регіонах, які недостатньо чітко визначені у вхідному тексті;

– виникнення проблеми заплутування (disentanglement problem) – виявлення кореляції в ознаках, які не пов'язані (слабко пов'язані) в реальному світі;

– мережі можуть страждати від колапс мод (mode collapse), коли генератор навчається створювати обмежений набір зображень, які не покривають адекватно діапазон можливостей.

Mode collapse виникає коли під час навчання генератор переходить в стан, коли він постійно генерує одні й ті самі зображення, що зводить простір згенерованих зображень до меншої розмірності, ніж вихідний простір. Головна причина цього полягає у тому, що генератор навчається обманювати дискримінатор, а не відтворювати вихідний розподіл. Якщо генератор починає постійно генерувати однакові зображення, що є найбільш ймовірними для поточного дискримінатора, то залежність від вектору шуму z поступово зникає, і градієнт $G(z)$ прямує до нуля. На рисунку 2.11 ілюструється проблема mode collapse і процес під час якого генератор «переходить» між різними модами, не наближаючись до цільового розподілу.

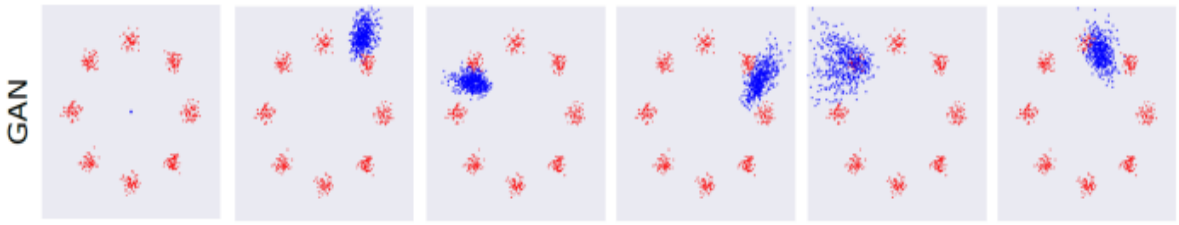

Рисунок 2.11 – Проблема mode collapse у GAN

На даний момент mode collapse є однією з головних проблем GAN, ефективне рішення якої ще шукається. Було запропоновано багато вирішень, щоб знизити ризик виникнення цієї проблеми:

– функція втрат Вассерштейна (Wasserstein loss), що спрямована на більш безпосередню мінімізацію відстані між двома розподілами ймовірностей. Це важливо, оскільки виграш генератора в грі не обов'язково призводить до зменшення відстані між згенерованим і реальним розподілами ймовірностей, що може призвести до таких проблем, як колапс режиму;

– пакування (Packing) – модифікація дискримінатора, яка дозволяє йому приймати рішення на основі декількох зразків, що належать до одного класу (реального або штучного). Дивлячись на групу зразків одночасно, дискримінатор має більше шансів ідентифікувати штучні групи, якщо вони не є різноманітними. Цей метод може допомогти вирішити проблеми з надмірною залежністю від кількох домінуючих мод на виході генератора;

– розгортання (Unrolling) – це метод, який передбачає оновлення функції втрат генератора для зворотного розповсюдження через k кроків оновлення градієнта дискримінатора. Це дозволяє генератору «бачити» на k кроків у «майбутнє» і спонукає його створювати більш різноманітні та реалістичні вибірки.

2.2 Дифузійні моделі

Дифузійні моделі – це клас генеративних моделей призначених для моделювання складних розподілів ймовірностей. Вони є ланцюгами Маркова, навчені за допомогою варіаційного висновку і працюють шляхом ітераційного застосування певної кількості кроків дифузії до даних (наприклад, зображення).

Дифузійні моделі можна поділити мінімум на три категорії. До першої підкатегорії належать імовірнісні моделі дифузії з усуненням шуму (DDPM), підґрунтям яких була теорія нерівноважної термодинаміки. Розподіл ймовірностей оцінюється за допомогою латентних змінних у DDPM, тому ці моделі можна розглядати як особливий тип варіаційних автокодувальників, де процес прямої дифузії – це процес кодування всередині VAE, а зворотної дифузії – декодування.

До другої підкатегорії відносяться мережі оцінок з умовою шуму (NCSN). В цій моделі вибірки генеруються динамікою Ланжевена з використанням градієнтів розподілу даних, що оцінюються зіставленням оцінок. Коли дані знаходяться на низьковимірних маніфолдах, вони збурюються гаусівським шумом різного рівня і спільно оцінюються відповідні оцінки – векторні поля градієнтів розподілу збурених даних для всіх рівнів шуму. Це обумовлено можливістю градієнтів бути нечітко визначеними і важко оціненими.

Третя підгрупа моделей дифузії – стохастичні диференціальні рівняння (SDE), пропонує інший підхід до моделювання дифузії Використовується континуум розподілів, які змінюються з часом відповідно до процесу дифузії, замість додавання шуму до даних зі скінченною кількістю розподілів шуму. Завдяки заданому стохастичному диференціальному рівнянню що не має параметрів, що навчаються і не залежить від даних моделюється цей процес. Створювати нові зразки можна

перевертаючи процедуру. Можна вважати, що SDE є узагальненням для DDPM і NCSN.

Далі будуть розглянуть імовірнісні моделі дифузії з усуненням шуму DDPM, їх покращену версію DDIM, LDM, фреймворки для генерації зображення за текстом в піксельному (GLIDE, Imagen) та латентному (VQ-Diffusion, Stable Diffusion) просторах.

2.2.1 Ймовірнісна дифузійна модель з усуненням шуму (DDPM)

DDPM, як і інші дифузійні моделі, складаються з двох процесів: прямої дифузії та параметризованої зворотної (рисунок 2.12).

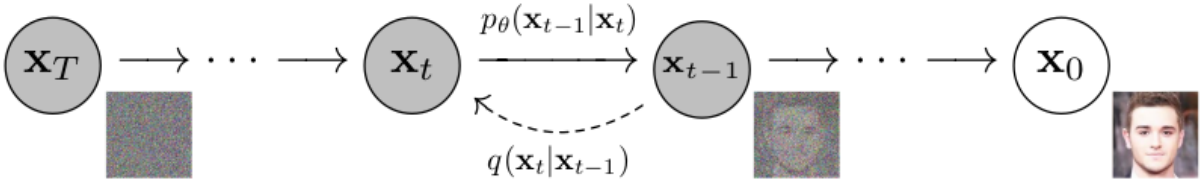

Рисунок 2.12 – Процеси прямої і зворотної дифузії

Дифузійний (прямий) процес (справа наліво на рисунку 2.11) є стохастичним процесом, який починається з вибірки даних і ітеративно генерує більш зашумлені вибірки за допомогою простого ядра гаусової дифузії. Це ланцюг Маркова, адже в момент часу *t* зображення залежить не від усіх попередніх зображень, а від зображення $t-1$. На кожному кроці x_t вибирається відповідно до наступної ймовірності переходу (формула 2.19).

$$q(\mathrm{x}_t \mid \mathrm{x}_{t-1}) := \mathcal{N}\left(\mathrm{x}_t; \sqrt{1-\beta_t}\mathrm{x}_{t-1}, \beta_t \mathrm{I}\right), \forall t \in \{1, \ldots, T\}, \quad (2.19)$$

де $N(x;\ \mu, \sigma)$ – нормальний розподіл, що видає x з середнім значенням μ і дисперсією σ;

β_t – параметри, визначають дисперсійний графік і можуть бути гіперпараметрами або вивчені;

I – матриця тотожності, що має розміри як у вхідному зображенні x_t.

Важливою властивістю прямого процесу є те, що він допускає дискретизацію x_t у довільний момент часу t, тому використовуючи нові позначення маємо формулу 2.20.

$$q(\mathrm{x}_t \mid \mathrm{x}_0) := \mathcal{N}\left(\mathrm{x}_t; \sqrt{\bar{\alpha}_t}\,\mathrm{x}_0, (1 - \bar{\alpha}_t)\mathrm{I}\right), \tag{2.20}$$

де $\alpha_t := 1 - \beta_t$;

$\bar{\alpha}_t := \prod_{s=1}^{t} \alpha_s$.

За цією формулою шляхом вибірки з гаусівського розподілу можна отримати зображення в усіх можливих t.

Параметризований зворотний процес виконує ітеративне зашумлення, який скасовує пряму дифузію. Використовуючи властивості прямого проходу, необхідно згенерувати $p(\mathrm{x}_0)$ від $\mathrm{x}_t \sim \mathcal{N}(0, \mathrm{I})$ за допомоги зворотних шагів (формула 2.21).

$$p(\mathrm{x}_{t-1} \mid \mathrm{x}_t) := \mathcal{N}(\mathrm{x}_{t-1};\ \mu(\mathrm{x}_t, t), \sum(\mathrm{x}_t, t)). \tag{2.21}$$

Для цього необхідно натренувати нейронну мережу, де входом є зашумлена картинка x_t і вбудовування на часовому кроці t, а параметрами, що прогнозуються є середнє $\mu_\theta(\mathrm{x}_t, t)$ і дисперсія $\sum_\theta(\mathrm{x}_t, t)$ (формула 2.22).

$$p_\theta(\mathrm{x}_{t-1} \mid \mathrm{x}_t) := \mathcal{N}\left(\mathrm{x}_{t-1}; \mu_\theta(\mathrm{x}_t, t), \sum_\theta(\mathrm{x}_t, t)\right). \tag{2.22}$$

Використовуючи перепараметризацію з прямого процесу маємо формулу 2.23.

$$\mathrm{x}_t(\mathrm{x}_0, \epsilon) = \sqrt{\bar{\alpha}_t}\mathrm{x}_0 + \sqrt{1-\bar{\alpha}_t}\epsilon \text{ для } \epsilon \sim \mathcal{N}(0, \mathrm{I}). \quad (2.23)$$

Процес прямої і зворотної дифузії наглядно можна побачити на рисунку 2.13.

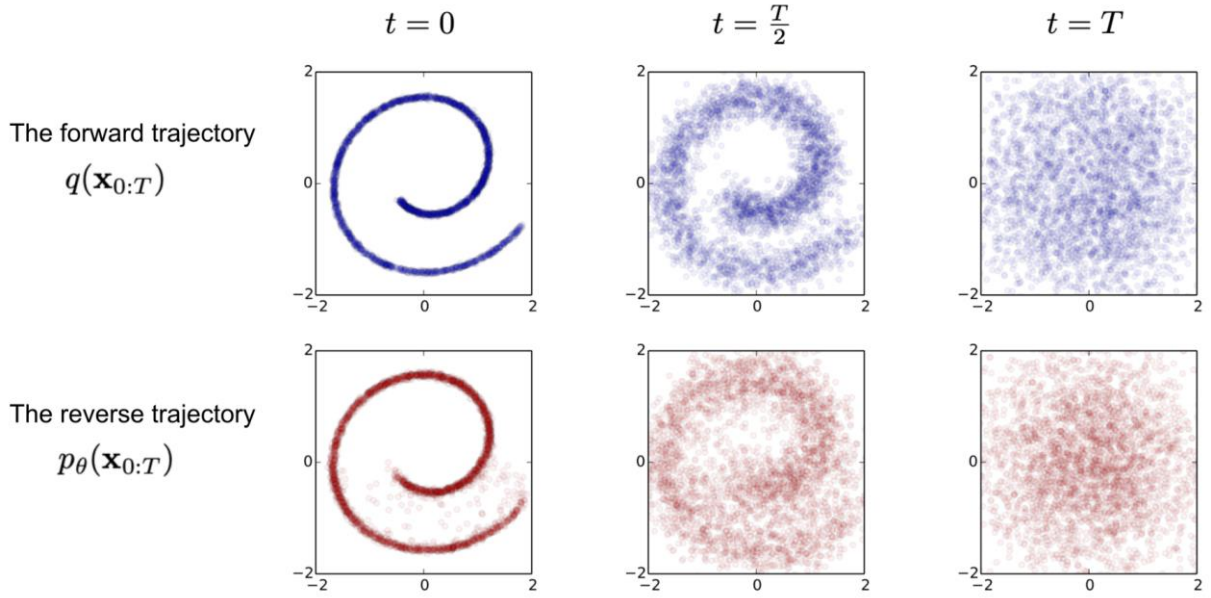

Рисунок 2.13 – Приклад навчання дифузійної моделі для моделювання 2D даних

В ідеальному випадку для навчання нейронної мережі слід було б використовувати ціль максимальної правдоподібності, щоб переконатися, що ймовірність кожного навчального прикладу була якомога вищою. Однак, оскільки для обчислення $p(\mathrm{x}_0)$ необхідно маргіналізувати всі потенційні зворотні траєкторії, це нерозв'язна задача. Щоб вирішити це можна мінімізувати фіксовану нижню межу від'ємної логарифмічної ймовірності. А якщо дисперсію зробити константним значенням, а середнє сформулювати як функцію шуму (формула 2.24), то можна отримати більш спрощену форму цільової функції, яка вимірює розділення між реальним

шумом та оцінкою шуму моделі для випадкового часового кроку t прямого процесу (формула 2.25).

$$\mu_\theta(\mathrm{x}_t, t) = \tilde{\mu}_t\left(\mathrm{x}_t, \frac{1}{\sqrt{\bar{\alpha}_t}}\left(\mathrm{x}_t - \sqrt{1-\bar{\alpha}_t}\epsilon_\theta(\mathrm{x}_t)\right)\right) = \frac{1}{\sqrt{\alpha_t}}\left(\mathrm{x}_t - \right.$$
$$\left. - \frac{\beta_t}{\sqrt{1-\bar{\alpha}_t}}\epsilon_\theta(\mathrm{x}_t, t)\right), \tag{2.24}$$

$$L_{DM} = \mathbb{E}_{x,\epsilon\sim\mathcal{N}(0,1),t}\left[\|\epsilon - \epsilon_\theta(x_t, t)\|_2^2\right]. \tag{2.25}$$

де ϵ_θ –автокодувальник, що зашумлює (варіант обмежений у часі U-Net).

На рисунку 2.14 представлено алгоритм тренування і методу відбору. В реальності тренування відбувається на пакетах даних (а не на одному зразку), оскільки для оптимізації нейронних мереж використовується стохастичний градієнтний спуск [14].

Algorithm 1 Training

1: **repeat**
2: $\mathbf{x}_0 \sim q(\mathbf{x}_0)$
3: $t \sim \text{Uniform}(\{1, \dots, T\})$
4: $\boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$
5: Take gradient descent step on
$\nabla_\theta \left\|\boldsymbol{\epsilon} - \boldsymbol{\epsilon}_\theta(\sqrt{\bar{\alpha}_t}\mathbf{x}_0 + \sqrt{1-\bar{\alpha}_t}\boldsymbol{\epsilon}, t)\right\|^2$
6: **until** converged

Algorithm 2 Sampling

1: $\mathbf{x}_T \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$
2: **for** $t = T, \dots, 1$ **do**
3: $\mathbf{z} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$ if $t > 1$, else $\mathbf{z} = \mathbf{0}$
4: $\mathbf{x}_{t-1} = \frac{1}{\sqrt{\alpha_t}}\left(\mathbf{x}_t - \frac{1-\alpha_t}{\sqrt{1-\bar{\alpha}_t}}\boldsymbol{\epsilon}_\theta(\mathbf{x}_t, t)\right) + \sigma_t\mathbf{z}$
5: **end for**
6: **return** $\mathbf{x}_0$

Рисунок 2.14 – Алгоритм тренування і відбору зразків в DDPM

2.2.2 Неявна модель дифузії з усуненням шуму (DDIM)

Як прямий, так і зворотний процеси часто використовують тисячі кроків, що робить процес створення зразка з DDPM дуже повільним.

У неявних моделях дифузії з усуненням шуму (DDIM) марковський прямий процес замінено на немарковський. Завдяки модифікації процесу генерації модель прогнозує нормальну вибірку перед тим, як використовувати її для оцінки наступного кроку в ланцюжку. Не впливаючи на якість отриманих вибірок, коригування прискорює процес формування вибірки. Також DDIM можна використовувати після навчання моделі, що дозволяє легко трансформувати DDPM-моделі в DDIM без необхідності навчати нову модель.

Було перевизначено процес зворотної дифузії для одного кроку в формулі 2.26.

$$x_{t-1} = \sqrt{\bar{\alpha}_{t-1}}\left(\frac{x_t - \sqrt{1-\bar{\alpha}_t}\epsilon_\theta^{(t)}(x_t)}{\sqrt{\bar{\alpha}_t}}\right) + \sqrt{1-\bar{\alpha}_{t-1}-\sigma_t^2}\cdot\epsilon_\theta^{(t)} + \sigma_t\epsilon_t. \quad (2.26)$$

При $\sigma = 0$ отримуємо формулу 2.27.

$$x_{t-1} = \sqrt{\bar{\alpha}_{t-1}}\left(\frac{x_t - \sqrt{1-\bar{\alpha}_t}\epsilon_\theta^{(t)}(x_t)}{\sqrt{\bar{\alpha}_t}}\right) + \sqrt{1-\bar{\alpha}_{t-1}}\cdot\epsilon_\theta^{(t)}. \quad (2.27)$$

З формули вище видно, що дані не містять шуму. У цьому полягає особливість DDIM. Оскільки більше ніякого шуму не вноситься протягом процесу згладжування, єдиним присутнім шумом є початковий шум при x_0, коли $\sigma = 0$. Як результат, процес згладжування є повністю детермінованим. Тому не має сенсу використання ланцюгів Маркова, адже вони використовуються для імовірнісних процесів. Використовуються немарковська процедура, яка дозволяє пропускати кроки, що зображена на рисунку 2.15.

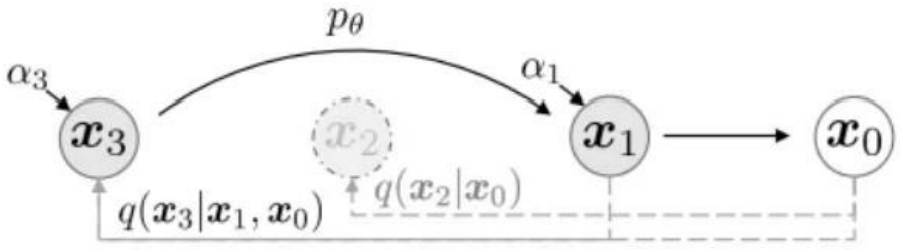

Рисунок 2.15 – Немарковський зворотний і прямий процес, де $\tau = [1, 3]$

На цьому рисунку крок x_2 пропускається при переході від x_3 до x_1.

Новий процес дифузії моделюється як підпослідовність τ, яка є підмножиною оригінальної дифузійної послідовності. Наприклад, можна отримати вибірку кожного наступного кроку дифузії, щоб отримати підпослідовність $\tau = [0, 2, 4, \ldots, T-2, T]$.

Як інтерполяція між DDIM і DDPM дисперсія визначається за формулою 2.28.

$$\sigma_t = \eta \sqrt{\frac{1-\bar{\alpha}_{t-1}}{1-\bar{\alpha}_t}} \sqrt{1-\frac{\bar{\alpha}_t}{\bar{\alpha}_{t-1}}} = \eta \sqrt{\tilde{\beta}_t}. \tag{2.28}$$

Дифузійна модель є DDIM, коли $\eta = 0$ так, як шум відсутній, і оригінальною DDPM, коли $\eta = 1$. Будь-яке значення η між 0 і 1 є інтерполяцією між DDIM і DDPM [41].

У порівнянні з DDPM, DDIM може:

– створювати вибірки вищої якості за значно меншу кількість кроків;

– мати властивість узгодженості, тобто вибірки однаковими латентними змінними матимуть схожі характеристики однієї, оскільки процес генерації є детермінованим;

– виконувати семантично значущу інтерполяцію латентної змінної.

2.2.3 Додання умов в дифузійні моделі. Фреймворки GLIDE і Imagen

У стандартній дифузійній моделі для генерації змінні випадковим чином вибираються. Додання умови (клас чи текст) до процесу генерації може значно покращить якість згенерованих зображень, тому як і для GAN (підрозділ 2.1.1) вводяться моделі обумовлені класом (class conditional diffusion models). Ці моделі додатково включають мітку класу як вхідну змінну, що дає їм середнє значення $\mu_\theta(x_t|y)$ та дисперсію $\Sigma_\theta(x_t|y)$.

Крім того, можна впливати на прогноз моделі під час вибірки, щоб направити зразок до мітки. Також можна використовувати мітку лише для керування відбору зразка. Існує кілька методів керованого відбору зразків. В керуванні на основі класифікатора (classfier-based guidance) градієнт логарифму ймовірності цільового класу y, передбаченого класифікатором, зміщує середнє значення (формула 2.29).

$$\widehat{\mu}_\theta(\mathrm{x}_t \mid y) = \mu_\theta(\mathrm{x}_t \mid y) + s \cdot \Sigma_\theta(\mathrm{x}_t \mid y)\nabla_{\mathrm{x}_t}\log\, p_\phi(y \mid \mathrm{x}_t). \qquad (2.29)$$

де s – шкала наведення (guidance scale).

Вхід x_t подається не з кінцевого зображення, а з випадкового проміжку часу.

Стратегія керування без класифікатора (classifier-free guidance) пропонується використовувати аби уникнути необхідності навчання додаткової моделі. Тут припускається, що прогноз моделі є $\epsilon_\theta(x_t|y)$. Умовна модель $\epsilon_\theta(x_t|y)$ і безумовна $\epsilon_\theta(x_t)$ навчаються спільно. Таким чином, різниця між прогнозами умовної та безумовної моделей прямо пропорційна збуренню (формула 2.30) [41].

$$\widehat{\epsilon}_\theta(x_t) = \epsilon_\theta(x_t) + s \cdot \left(\epsilon_\theta(\mathrm{x}_t|y) - \epsilon_\theta(x_t)\right). \qquad (2.30)$$

Першою роботою DM для генерації зображень за текстовим описом (T2I) був GLIDE. Для цього замінити мітку в зумовленій класом DM на текст, щоб зумовити генерацію зразка текстом. Мотивовано покращенням якості зображень керованою дифузією і полегшенням обробки підказок (prompt) у довільній формі в варіанті керування без класифікатора, GLIDE реалізував безкласифікаторне керування/наведення в T2I, замінивши оригінальної мітки класу на текст. Архітектура являє собою модель U-Net з блоками залишкового шару згортки (residual convolutional blocks) та блоками уваги, об'єднаними разом.

На вхід подаються підпис c, зображення x_t та часовий інтервал t. Створюється гармонійне вбудовування з часового інтервалу, а потім проектується за допомогою лінійних шарів і може додаватись до виходів залишкових блоків. Текст обробляється створенням послідовності K токенів з c і включенням в модель трансформера. До вбудовування часового інтервалу додається фінальне вбудовування токенів.

Крім того, шляхом проектування, а потім конкатенації з входами уваги до ключа і значення, кінцевий шар з K вбудовувань токенів, вводиться до кожного шару уваги моделі U-Net.

Модель видає зображення формату 64 x 64, тому для отримання 256 x 256 кінцевих виходів тренується модель, яка також залежить від текстових вхідних даних.

Як і GLIDE, Imagen використовує безкласифікаторне керування для створення зображень. Як показано на рисунку 2.16, текстовий кодувальник, який використовує GLIDE та Imagen, є ключовою відмінністю між ними. Зокрема, GLIDE використовує парні дані зображення-текст для навчання текстового кодувальника і дифузії, а Imagen використовує попередньо навчену і заморожену велику мовну модель. Заморожування навчання кодувальника значно зменшує обчислювальне навантаження на онлайн-навчання попереднього дифузійного навчання [41].

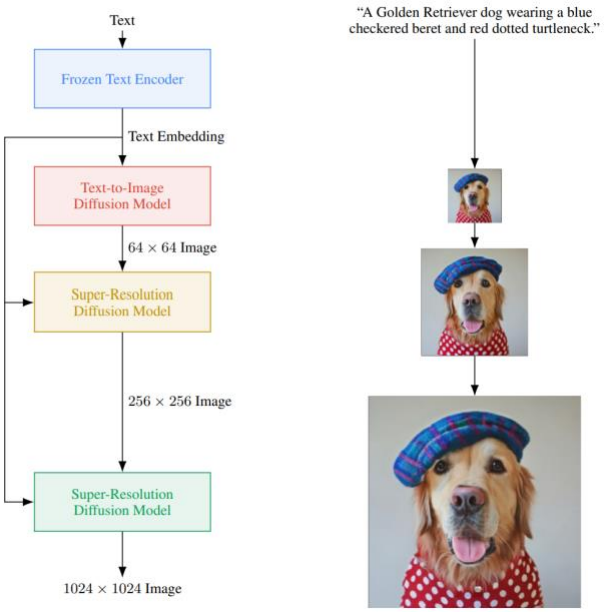

Рисунок 2.16 –Діаграма моделі Imagen

Крім того, текстовий кодувальник може бути попередньо навчений як на корпусах лише тексту, так і на корпусах зображення-текст (наприклад, CLIP). Оскільки корпус лише тексту набагато більший, ніж парні дані зображення-текст, ці великі мовні моделі піддаються впливу тексту, який є багатим і різноманітним. Для підвищення ефективності в цій моделі створено архітектуру Efficient U-Net, який є простішим, швидше сходиться та ефективніше використовує пам'ять [41].

2.2.4 Латентні дифузійні моделі (LDM). Фреймворки Stable Diffusion, VQ-Diffusion

Моделі латентної дифузії виконують процес дифузії в латентному просторі, а не в піксельному, що знижує витрати на навчання і пришвидшує генерацію. Натхненням для цього стало відкриття того, що навіть після екстремального стиснення семантична та концептуальна композиція зображення все ще існує. Спочатку усуваючи надлишковість на рівні пікселів за допомогою автокодувальника, а потім генеруючи семантичні

концепції за допомогою процесу дифузії на вивчених латентних представленях, LDM вільно декомпозує перцептивне і семантичне стиснення за допомогою генеративного моделювання навчання.

Було запропоновано кодувати вхідні дані в латентне представлення за допомогою мережі кодувальників. Обґрунтуванням цього вибору є обробка вхідних даних у просторі зі зменшеною розмірністю, що зменшить обчислювальні вимоги для навчання дифузійних моделей. Потім типова дифузійна модель (U-Net) використовується для створення нових даних, які декодувальна мережа аналізує.

Отже, маючи кодувальник $\mathcal{E}$ та латентне представлення z, втрата для моделі латентної дифузії (LDM) має вигляд як у формулі 2.31.

$$L_{LDM} = \mathbb{E}_{\mathcal{E}(x),\epsilon\sim\mathcal{N}(0,1),t}\left[\|\epsilon - \epsilon_\theta(z_t, t)\|_2^2\right]. \quad (2.31)$$

Stable Diffusion, розширена версія моделі латентної дифузії (LDM), є репрезентативним фреймворком, який навчає моделі дифузії на латентному просторі. Використовуючи VQ-GAN для латентного представлення на першому кроці, Stable Diffusion наслідує Dall-E, який використовує VQ-VAE для вивчення візуального коду. Слід зазначити, що VQ-GAN покращує VQ-VAE шляхом включення конкурентної мети для підвищення автентичності синтезованих зображень. Stable Diffusion з попередньо навченим VAE змінює процес прямої дифузії, який створює шум у латентному просторі. Підхід також вводить перехресну увагу як універсальну умову для різних сигналів стану, таких як текст. Згідно з експериментальними даними, моделювання дифузії в латентному просторі значно перевершує моделювання дифузії в піксельному просторі з точки зору зменшення складності та збереження ознак [22].

Архітектуру цієї мережі зображено на рисунку 2.17.

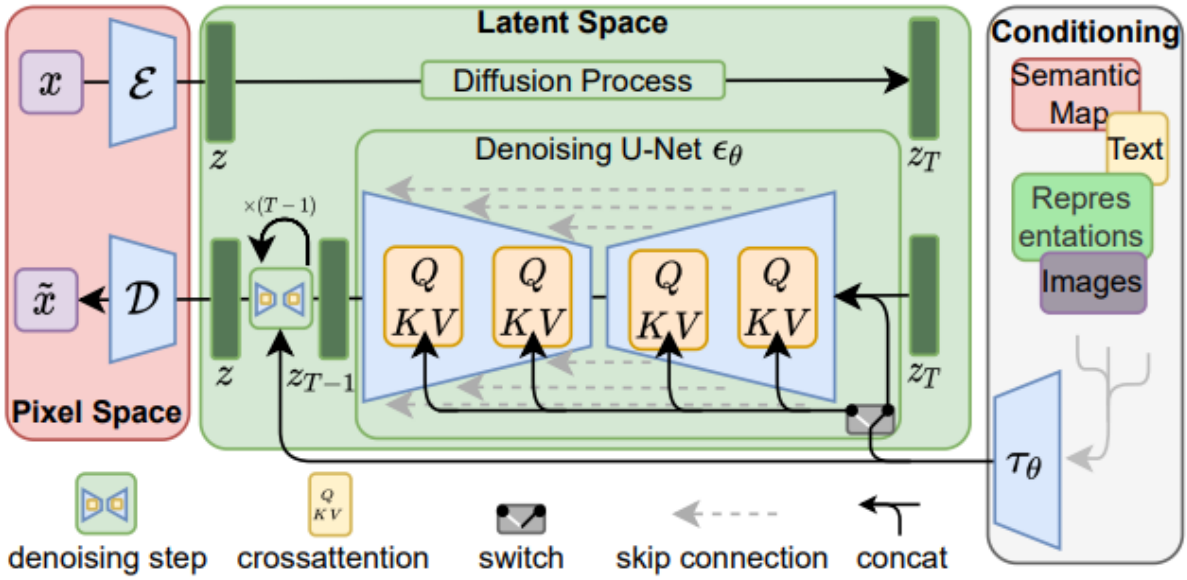

Рисунок 2.17 – Архітектура латентної дифузійної моделі і Stable Diffusion

Цей підхід можна поділити на чотири основні етапи. На першому за допомогою кодувальника Ɛ, зображення спочатку стискається у зменшену репрезентацію. Прихована дифузія працює в прихованому просторі кодувальника, а не в просторі пікселів.

Далі як частина процесу дифузії, що рухається від z до z_T до представлення додається гаусівський шум. Після цього через U-Net проходить представлення z_t. За прогнозування z_{t-1} відповідає U-Net, і ця процедура виконується T разів поспіль до отримання z, яке потім переводиться з латентного простору в піксельний простір декодувальником D.

Останнім етапом є йде обумовлення шляхом зіставлення іншої модальності, наприклад текст. Вхідна модальність y перетворюється за допомогою спеціалізованого кодувальника τ_θ, а потім відображається на проміжні шари U-Net за допомогою того ж шару перехресної уваги, що і в трансформерах.

Для навчання автокодувальника для стиснення зображень, використовується підхід як в VQGAN (рисунок 2.18)

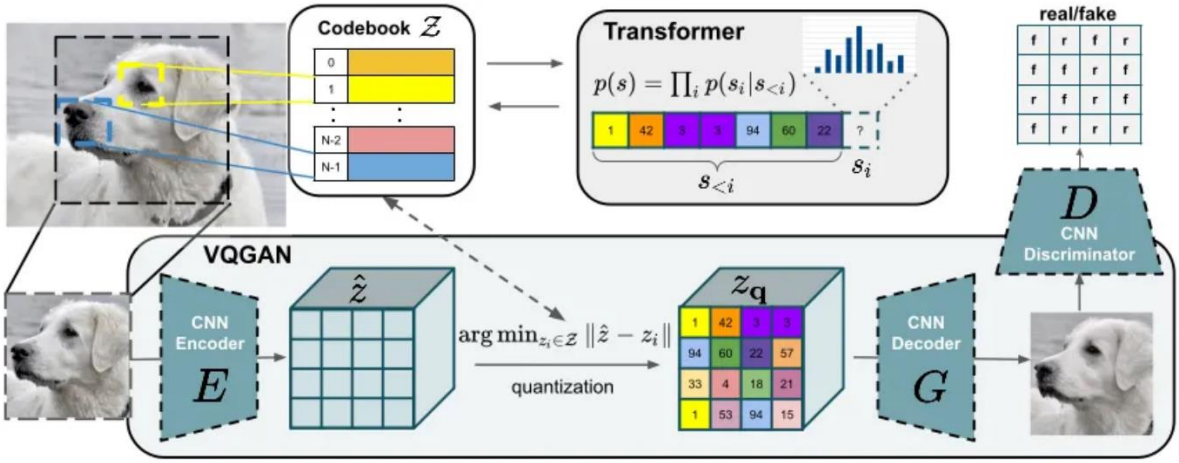

Рисунок 2.18 – Згортковий VQGAN для вивчення кодової книги контекстно-багатих візуальних частин

Цільова функція моделі автокодувальника (E, D) для навчання описана у формулі 2.32.

$$L_{\text{Autoencoder}} = \min_{\mathcal{E},\mathcal{D}} \max_{\psi} (L_{rec}\left(x, \mathcal{D}(\mathcal{E}(x))\right) - L_{adv}\left(\mathcal{D}(\mathcal{E}(x))\right) + \\ + \log D_{\psi}(x) + L_{reg}(x; \mathcal{E}, \mathcal{D})), \tag{2.32}$$

де L_{rec} – втрата реконструкції (квадратична помилка між x і реконструйованим $\hat{x}$ $(\hat{x} = \mathcal{D}(\mathcal{E}(x)))$;

L_{adv} – змагальна втрата (adversarial loss) і дорівнює $\log(1 - D_{\psi}(\hat{x}))$;

D_{ψ} – дискримінатор на основі патчів, оптимізований для розрізнення оригінальних зображень з реконструкцій $\hat{x}$;

L_{reg} – регуляризаційна втрата.

Для регуляризації можна використовувати або регуляризацію Кульбака-Лейблера або шар векторної квантизації (VQ). В першому підході ідея в тому, щоб наблизити розподіл латентної змінної до стандартного нормального розподілу. Це впорядкує і сконцентрує латентний простір так,

що якщо z_1 і z_2 знаходяться близько, то і $D(z_1)$ з $D(z_2)$ в чомусь будуть схожі. В другому випадку, як видно з рисунка 2.14, кожна просторову точка виходу кодувальника замінюється на найближчий вектор з «кодової книги». В результаті, на вхід декодувальнику тільки комбінації векторів кодової книги – дискретизація або квантування латентного простору можуть бути використані декодувальником як вхідні дані під час виводу.

В цьому підхід обумовлення дифузійної моделі текстом передбачається інтеграцією в проміжні шари моделі U-Net з використанням перехресної уваги (що схоже на архітектуру трансформера). Більш конкретно, використовуючи специфічний для домену кодувальник, відомий як (приклад буде наведено пізніше), вхідні дані (такі як текст) спочатку перетворюються на проміжне представлення. Це представлення потім додається до проміжних рівнів U-Net після проходження через шар перехресної уваги (формула 2.33).

$$Attention(Q, K, V) = softmax\left(\frac{QK^T}{\sqrt{d}}\right) \cdot V, \tag{2.33}$$

де $Q = W_Q^{(i)} \cdot \varphi(z_i)$, $W_Q^{(i)} \in R^{d \times d_\varepsilon^i}$;

$K = W_K^{(i)} \cdot \tau_\theta(y)$, $W_K^{(i)} \in R^{d \times d_\tau}$;

$V = W_V^{(i)} \cdot \tau_\theta(y)$, $W_V^{(i)} \in R^{d \times d_\tau}$;

W_Q, W_K, W_V – проекційні матриці, що навчаються;

$\varphi(z_i) \in R^{N \times d_\varepsilon^i}$ є сплющеним проміжним представленням U-Net;

τ_θ – специфічний для домену кодувальник, що проектує y на проміжне представлення $\tau_\theta(y) \in R^{M \times d_\tau}$.

Також важливо зазначити, що вихід шару перехресної уваги додається до вихідного шару U-Net.

Така ж стратегія дифузії також була досліджена у VQ-дифузії з використанням схожої методології. Наведення без класифікатора також

значно покращує моделі дифузії текст-зображення в латентному просторі, підтверджуючи результати, отримані за допомогою методу піксельного простору.

Модель VQ-Diffusion, метод синтезу тексту в зображення, який позбавлений односпрямованого упередження попередніх методів. На рисунку 2.19 зображено архітектуру цієї мережі.

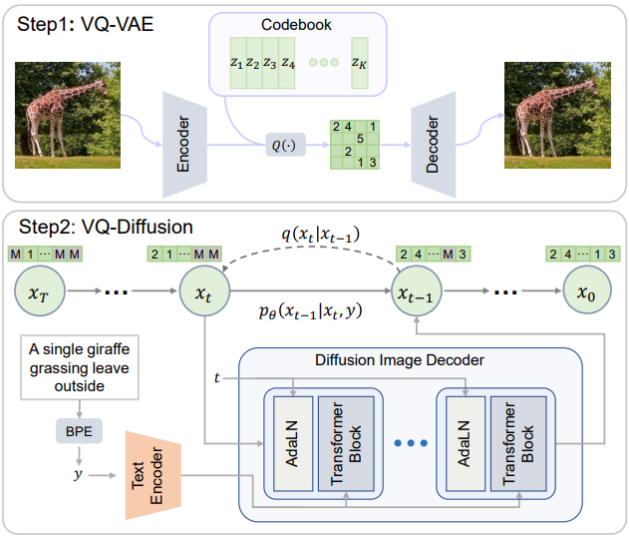

Рисунок 2.19 – Архітектура VQ-Diffusion

Запропоноване рішення запобігає накопиченню помилок під час виведення завдяки своїй стратегії маскування. Модель складається з двох етапів, перший з яких базується на VQ-VAE, що навчається представляти зображення за допомогою дискретних токенів, а другий – дискретна дифузійна модель, що маніпулює дискретним латентним простором VQ-VAE. Навчання DM обумовлено текстовими вбудовуваннями. Деякі токени замінюються на маски.

Текстовий кодувальник створює умовну послідовність ознак з текстових токенів y. Розподіл токенів без шуму $p_\theta(\widetilde{\mathrm{x}_0} \mid \mathrm{x}_t, y)$ створюється дифузійним декодувальником зображень, використовуючи токен зображення x_t і часовий інтервал t. До складу декодувальника входить шар

softmax і багато блоків трансформеру . Вони включають блок мережі прямого поширення, перехресну увагу для інтеграції текстової інформації та повну увагу. Поточний момент часу t вводиться в мережу за допомогою адаптивної нормалізації шару (ADALN) описаної в формулі 2.34.

$$AdaLN(h,t) = a_t LayerNorm(h) + b_t, \tag{2.34}$$

де h – проміжні активації;

a_t і b_t отримуються за допомогою лінійної проекції вбудовування часового кроку.

Для пришвидшення процесу виводу результату пропускається декілька шагів використовуючи перепараметризацію. Вибірки робиться в послідовності $x_T, x_{T-\Delta_t}, x_{T-2\Delta_t} \dots x_0$ (де Δt – часовий проміжок), замість $x_T, x_{T-1}, x_{T-2} \dots x_0$, зі зворотним розподілом переходу:

$$p_\theta\left(x_{t-\Delta_t} \mid x_t, y\right) = \sum_{\tilde{x}_0=1}^{K} q\left(x_{t-\Delta_t} \mid x_t, \tilde{x}_0\right) p_\theta(\tilde{x}_0 \mid x_t, y). \tag{2.35}$$

При такому пришвидшенні отриманні вибірки якість зображення лише несуттєво погіршується [41].

2.2.6 Переваги і недоліки DM

Дифузійні моделі досягають найсучасніших результатів синтезу зображень і не тільки. Загалом їх процес навчання та генерації є більш стабільним, ніж у GAN, оскільки процес дифузії є ітеративним. Цей ітеративний характер також дозволяє нам проводити навчання під наглядом на кожному етапі, що є перевагою. Також на відміну від GAN, де для

навчання потрібні дві моделі, дифузії потрібна лише одна модель. Також ці моделі є адаптивними та аналітично розв'язуваними.

Дифузійні моделі мають великий потенціал, але вони також мають недоліки:

– вони можуть бути досить дорогими в часі генерації і ресурсах, оскільки базуються на довгому марковському ланцюгу етапів дифузії для створення зразків. Хоча були запропоновані нові підходи для значного прискорення процедури, вибірка все ще повільніша, ніж GAN;

– необхідність у великих обсягах якісних навчальних даних, бо модель повинна правильно засвоїти відображення між простором текстових описів і латентним простором мережі генераторів зображень;

– DM використовують випадковий процес для вибірки, що може призвести до неможливості генерування певних типів зображень.

3 ПРОГРАМНА РЕАЛІЗАЦІЯ

3.1 Обґрунтування обраних програмних засобів

Було обрано мову програмування Python для реалізації архітектури мереж, їх тренування і оцінювання. Вона має широкий набір бібліотек та інструментів, які дозволяють швидко та ефективно створювати та тестувати моделі глибокого навчання.

Для побудови моделей використано бібліотеку PyTorch. Вона має всі необхідні реалізовані компоненти, інтерфейси потоків для побудови моделей та зручні інструменти обробки даних. PyTorch забезпечує підтримку GPU-прискорення, що може значно прискорити навчання моделей глибокого навчання. Також фреймворк забезпечує автоматичну диференціацію, що дозволяє розробнику обчислювати градієнти автоматично, без необхідності виводити та застосовувати градієнти вручну. З бібліотеки diffusers було отримано AutoEncoderKL для латентної дифузійної моделі. Крім того, використано бібліотеки numpy, pandas, matplotlib, scipy для маніпулювання і візуалізації даних.

Середовищем розробки було обрано Google Colab Pro, так як він надає доступ до більш потужних графічних процесорів (V100, A100). Також зчитування та збереження даних з накопичувача в Colab є доволі швидким процесом.

3.2 Огляд датасету

Для тренування моделей було обрано частину набору даних LAION Aesthetic, а саме 300 тис. зображень. LAION Aesthetic – це підмножина набору даних LAION 5B, яка була оцінена моделлю, навченою на основі вбудовувань CLIP моделі, як естетична. Цільове використання цього набору даних – створення зображень. Усі набори даних зображень LAION створені

на основі Common Crawl, некомерційної організації, яка щомісяця сканує мільярди веб-сторінок і публікує їх у вигляді величезних наборів даних. LAION зібрав усі HTML-теги зображень, які мали атрибути alt-тексту, класифікував отримані 5 мільярдів пар зображень на основі їхньої мови, а потім відфільтрував результати в окремі набори даних, використовуючи їхню роздільну здатність, прогнозовану ймовірність наявності водяного знаку та прогнозовану «естетичну» оцінку (тобто суб'єктивну візуальну якість). Приклади зображень продемонстровані на рисунку 3.1.

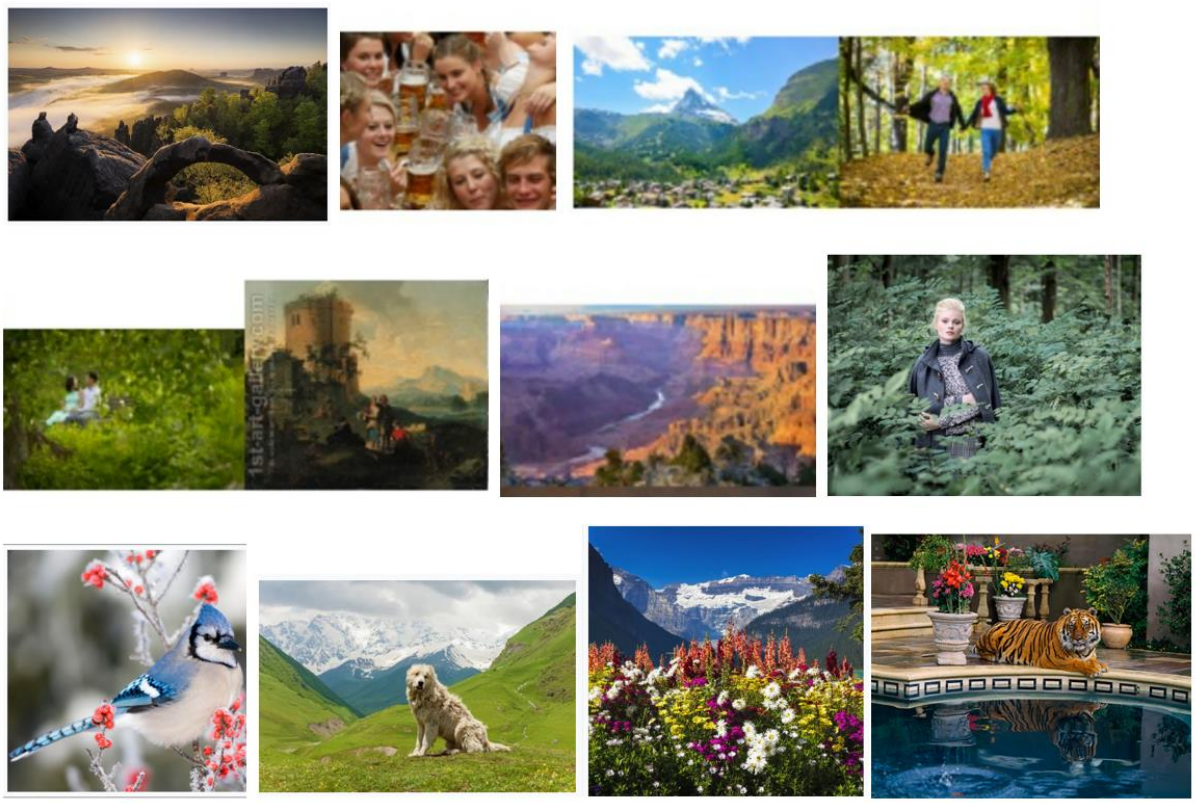

Рисунок 3.1 – Приклади даних з набору даних LAION Aesthetic

Оцінка якості натренованих моделей проводилась на наборі даних COCO, адже він є одним із стандартних для перевірки якості генерації зображень. Існують вже виміряні метрики для різних класів моделей, включаючи DMs і GANs, на цих даних, що дає порівняти не тільки натреновані моделі в обмежених ресурсах, а й найкращі моделі.

Набір даних MS COCO – це широко використовуваний набір даних для завдань комп'ютерного зору, зокрема для генерації зображень (image generation). Набір даних COCO містить понад 330 тисяч зображень з 80 категоріями об'єктів, що зустрічаються у повсякденному житті. Кожне зображення містить анотації з більш ніж 200 000 об'єктів, що робить цей набір даних незвичайно багатим і різноманітним. На рисунку 3.2 представлені приклади зображень з цього набору даних.

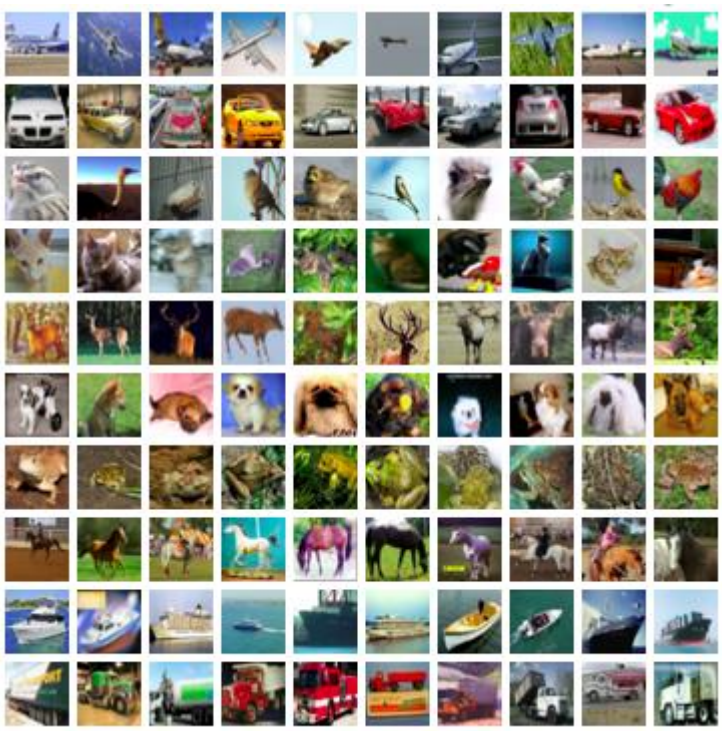

Рисунок 3.2 – Приклади даних з набору даних MS COCO

3.3 Огляд метрик для оцінки моделей текст-зображення

Оцінка генеративних моделей є суб'єктивною за своєю природою. Якісна оцінка таких моделей може бути схильна до помилок і може неправильно вплинути на рішення. Однак кількісні показники не обов'язково відповідають якості зображення. Тому, зазвичай, поєднання

якісних та кількісних оцінок дає сильніший сигнал при виборі однієї моделі над іншою.

Якісне оцінювання зазвичай передбачає людську оцінку створених зображень. Якість вимірюється за такими аспектами, як композиційність, відповідність між зображенням і текстом та просторові відносини.

До кількісних показників відносяться Inception Score (IS) і Frechet Inception Distance (FID). Вони надають числове значення, яке вимірює схожість між розподілами згенерованих зображень і реальних зображень у наборі даних. Тому їх можна використовувати для порівняння різних генеративних моделей.

Метрика FID є однією з найбільш популярних метрик для оцінки якості генеративних моделей. Вона базується на порівнянні двох розподілів зображень: розподілу зображень з навчального набору та розподілу зображень, що були згенеровані моделлю. Використовуючи нейромережу Inception (Inception V3), яка навчена класифікувати зображення, FID обчислює різницю між статистичними характеристиками цих двох розподілів. Ця різниця представляє собою евклідову відстань між двома наборами чисел, що описують характеристики зображень з розподілів.

Формула для обчислення FID наведено у формулі 3.1

$$FID = ||\mu_1 - \mu_2||^2 + Tr(\mathrm{C1} + \mathrm{C2} - 2 * \sqrt{\mathrm{C1} * \mathrm{C2}}), \quad (3.1)$$

де μ_1, μ_2 – вектори середніх значень Inception-векторів для навчального набору та набору згенерованих зображень відповідно;

$\mathrm{C1}$, $\mathrm{C2}$ – коваріаційні матриці для навчального набору та набору згенерованих зображень відповідно;

Tr – слід матриці.

На рисунку 3.3 представлено приклади оцінок FID для різних можливих спотворень згенерованих зображень.

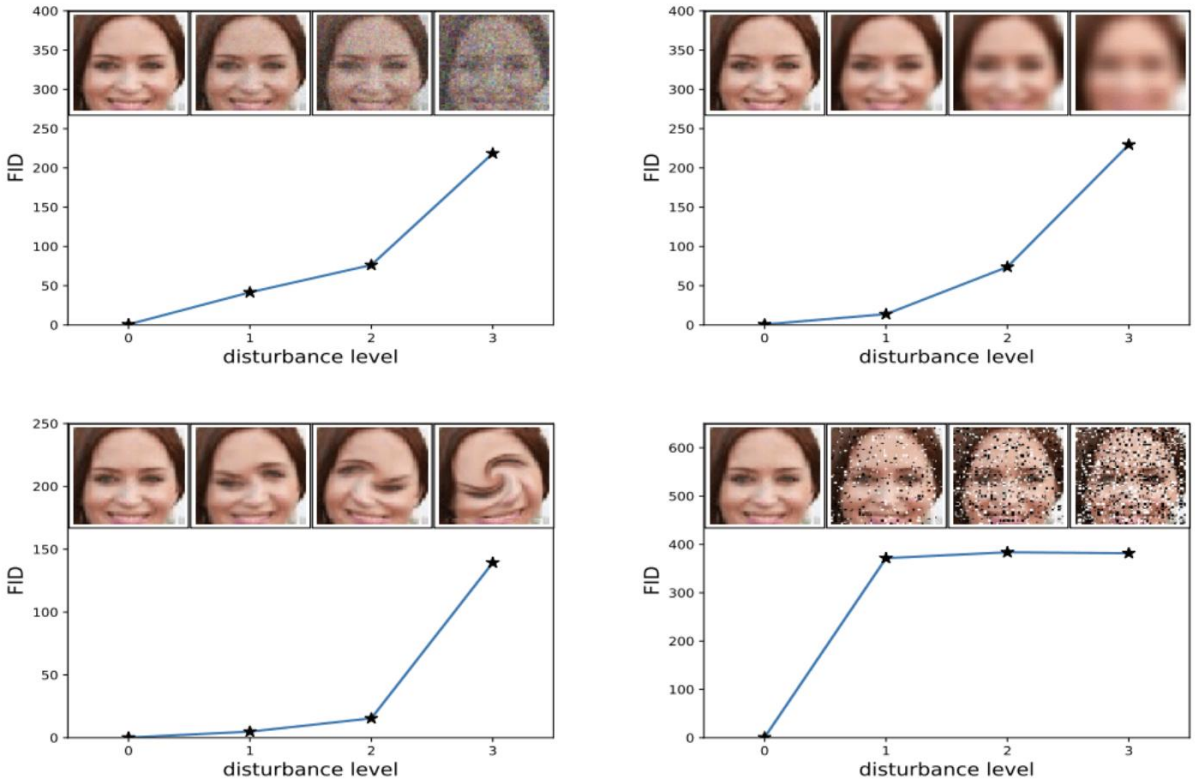

Рисунок 3.3 – Приклади кореляції спотвореного зображення з показником FID

Отже, чим менше значення FID, тим більш схожі розподіли зображень, тобто тим краща якість генерації моделлю. Зазвичай, значення FID менше 50 вважається дуже хорошим, а значення більше 100 – дуже поганим.

В цій роботі як кількісні показники будуть використані IS і FID) в поєднанні з візуальним оглядом і людською оцінкою для прийняття обґрунтованих рішень щодо продуктивності генеративної моделі.

Метрика IS вимірює різноманітність та якість згенерованих зображень, обчислюючи середнє значення KL розбіжності між умовним розподілом класів (обчисленим за допомогою softmax виходу мережі Inception) та граничним розподілом класів згенерованих зображень. Формула IS представлена в формулі 3.2.

$$IS = exp(E[KL(p(y|x) || p(y))]), \quad (3.2)$$

де $p(y|x)$ – умовний розподіл класів згенерованих зображень x, обчислений за допомогою softmax виходу мережі Inception;

$p(y)$ – граничний розподіл класів реальних зображень у наборі даних, обчислений як частота кожного класу в навчальній вибірці4

KL – розбіжність Кульбака-Лейблера між двома розподілами;

E – математичне очікуване значення на згенерованих зображеннях x.

Тобто оцінюється те, наскільки впевнений класифікатор в класифікації згенерованих зображень. Як правило, вищий показник IS вказує на те, що згенеровані зображення є більш різноманітними та репрезентативними для цільових класів, тоді як нижчий показник IS вказує на те, що згенеровані зображення є менш різноманітними та менш репрезентативними.

3.4 Обґрунтування вибору моделей

Для порівняння дифузійних моделей з генеративними змагальними мережами на практиці було обрано реалізувати латенту дифузійну модель подібну до Stable Diffusion (підрозділ 2.2.4) і GALIP (підрозділ 2.1.6).

Інтегровані в CLIP генератор і дискримінатор підвищують ефективність навчання, і в результаті модель потребує менше даних ніж зазвичай в цьому класі моделей. Додання CLIP покращує виокремлення інформативних, значимих візуальних ознак узгоджуючи текстовий опис і

забезпечує здатність до узагальнення домену. Модель порівняно з іншими має меншу кількість навчальних параметрів, має один генератор і дискримінатор, що скоротить час на навчання.

Модель латентної дифузії може працювати з низьковимірним латентним простором, що може бути більш ефективним в обчислювальному

плані порівняно з роботою безпосередньо з високорозмірним піксельним простором. Також може краще контролюватися процес генерації, що дозволяє більш тонко маніпулювати згенерованими зображеннями. Модель латентної дифузії може генерувати зображення вищої якості порівняно з моделями в піксельному просторі.

Отже в умовах обмежених ресурсів вони є кращою альтернативою для проведення експериментів в цій роботі.

3.5 Реалізація і тренування моделей

Реалізована модель GALIP складається з текстового кодувальника CLIP, дискримінатора і генератора на основі CLIP.

Попередньо навчений кодувальник тексту CLIP на вхід приймає текст за яким необхідно згенерувати зображення і повертає глобальний вектор речення T . Було обрано ViT-B/32 версію CLIP.

Дискримінатор на основі CLIP складається із замороженого CLIP-ViT і дискримінатора (Mate-D) зображеного на рисунку 3.4.

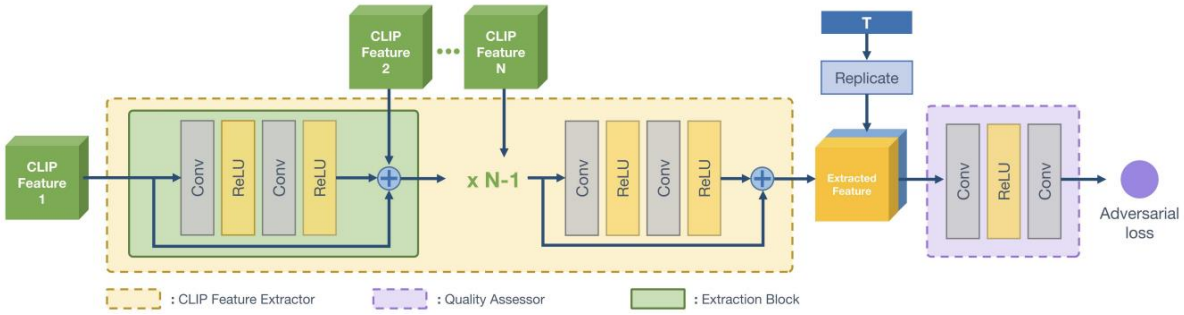

Рисунок 3.4 –Архітектура Mate-D

CLIP-ViT складається з трансформерних блоків та шару згортки і тим самим перетворює зображення на ознаки зображення. Далі ознаки зображення передаються з різних шарів у CLIP-ViT в Mate-D за допомогою

витягувача ознак CLIP (CLIP FE), що на рисунку зображений як блок CLIP Feature Extractor. Кожен блок CLIP FE містить два шари згортки та дві активні функції ReLU і пропуски з'єднань (skip connection). Результат блоку об'єднується з вхідом в блок і наступним елементом CLIP. Всього таких блоків 2, адже використовуються ознаки з 2, 5 і 9 шару CLIP. Для останньої ознаки CLIP додаються два шари згортки без додавання властивості CLIP позаду.

Отримані ознаки зображення після CLIP FE конкатенуються з вектором речень *T*. Ця комбінація передається на вхід до мережі інспектора якості (на рисунку Quality Assessor), що складається з шарів згортки і витягує додаткові корисні візуальні ознаки. Quality Assessor прогнозує оцінку якості зображення. Для стабілізації навчання Mate-D застосовується градієнтний штраф з урахуванням збігу (MAGP) до зібраних ознак CLIP і відповідних текстових ознак.

Генератор є комбінацією замороженого CLIP-ViT та дискримінатора Mate-G. Mate-G складається з предиктор особливостей Bridge Feature Predictor (Bridge-FP), предиктор підказок (Prompt predictor), замороженого CLIP-ViT і генератора зображення (Image Generator) (рисунок 3.5).

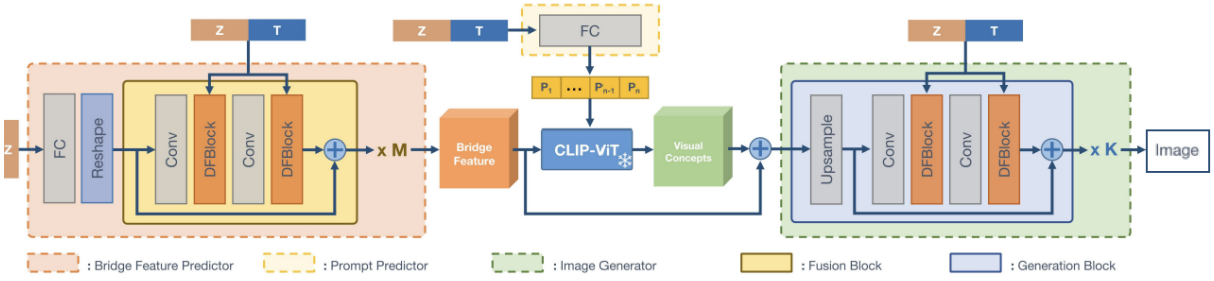

Рисунок 3.5 – Архітектура Mate-G

Bridge-FP складається з повнозв'язного шару (FC) та 4 блоків злиття (F-BLK). На FC подається вектор шуму z з гаусівського розподілу і перетворюється на розмірність (7, 7, 64) як початкова ознака в Bridge-FP.

F-BLK містить 2 шари згортки і 2 блоків глибокого злиття тексту і зображення (DFBlock), на який окремо подаються *z* і *T*. Таким чином, вектори тексту і шуму будуть перетворені у ознаки, які налаштовані для створення змістовних візуальних концепцій за допомогою CLIP-ViT.

Prompt Predictor створений, щоб зменшити складність перекладу мостових ознак з текстових, використовує шар FC для генерування підказок за допомогою *T* та *z*. У CLIP-ViT обмумовлені текстом підказки додаються до вбудовувань візуальних патчів (8 підказок для TransBlocks від 2 до 10 в CLIP-ViT).

Image Generator містить 6 блоків генерації (G-BLKs). На вхід передаються підсумовані візуальні ознаки та за допомогою DFBlocks об’єднуються вектор речення і вектор шуму в кожному G-BLK. Шари з підвищеною вибіркою (Upsample на рисунку 3.5) збільшують проміжні характеристики зображення в процесі створення. І в кінці створюються RGB-зображення розміром 128 x 128.

Для навчання використовувався оптимізатор Adam. В таблиці 3.1 зображені значення для гіперпараметрів навчання моделі.

Таблиця 3.1 – Гіперпараметри навчання моделі

Параметр	Значення
гіперпараметр градієнтного штрафу k	2
гіперпараметр градієнтного штрафу p	6
коефіцієнт схожості між текстом і зображенням, λ	4
крок навчання генератора	0.0001
крок навчання дискримінатора	0.0004

Модель латентної дифузійної моделі (підрозділ 2.2.4) реалізована з наступних компонентів:

– Варіаційний автокодувальник VAE;

– Кодувальник тексту – CLIP;

– Denoising UNet.

В цій роботі навчається лише UNet з нуля і використовуються попередньо навчені моделі для текстового кодувальника та VAE .

Модель CLIP, яка приймає текстову підказку (текстовий опис) генерує вбудовування розмірністю 512. Далі ці вбудовування передаються на вхід до UNet.

Було обрано натренований VAE з втратою Кульбака-Лейблера. Зображення кодуються за допомогою цієї моделі в низьковимірний латентний простір, де відбувається дифузія. Це виправдано тим, що природні зображення зазвичай можна досить сильно стиснути без втрати якості сприйняття, що робить модель дифузії в просторі зображення (пікселів) марнотратною. Автокодувальник бере зображення розміром 128*128*3 і кодує їх у простір розміром 16*16*4. Це дозволяє моделі латентної дифузії швидше сприймати ключові ознаки і дозволить декодувальник впоратися з більш тонкими деталями.

Denoising UNET за структурою схожий на автокодувальник (теж стискає до меншої розмірності, а потім розтискає в більшу), але також маж додаткові особливості:

– більша глибина мережі;

– skip connections;

– шари самоуваги та перехресної уваги;

– більша глибина;

– синусоїдальні вбудовування зашумленого рівня.

Процес навчання в реалізованій моделі виглядає наступним чинном:

– кодується зображення у латентний простір автокодувальником;

– кодується текстовий опис текстовим кодувальником;

– тренується модель U-NET, на вході якої 3 типи вхідних даних: рівень шуму (вибірка від 0 до 1 з конфігурацією пріоритету концентрації

значень до 0), вбудовування зображення, пошкоджене рівнем випадкового шуму та текстове вбудовування.

Модель UNET навчається для мінімізації абсолютної похибки між прогнозом і реальним прихованим зображенням. Для оптимізації було обрано Adam з навчальним кроком 0.0003. Процес генерації зображень зображено на рисунку 3.6.

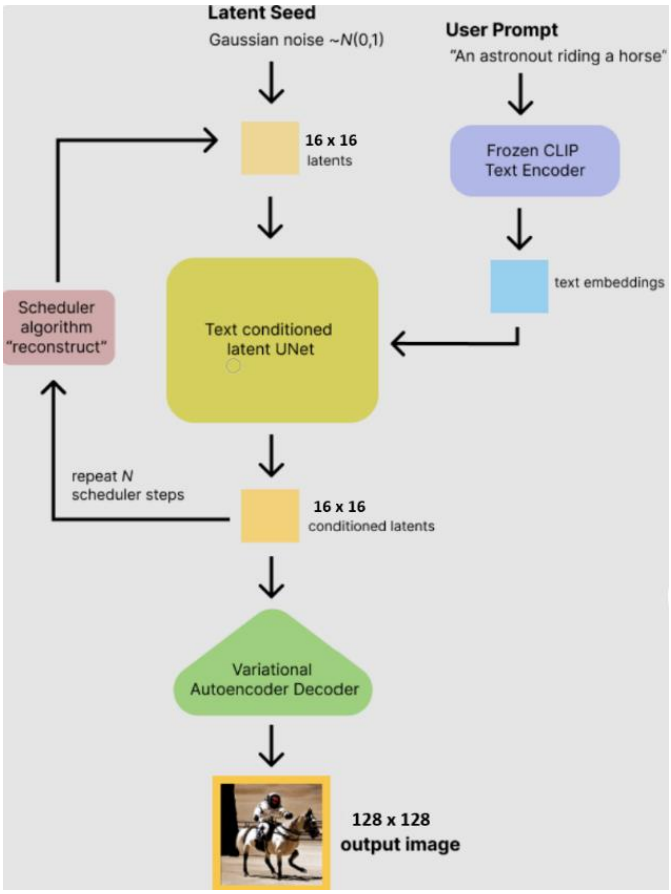

Рисунок 3.6 – Логічна послідовність генерації зображення в реалізованій LDM

Порівняння навчання реалізованих моделей зображена в таблиці 3.2 , а швидкості і важкості реалізованих моделей в таблиці 3.3.

Таблиця 3.2 – Характеристика тренувань реалізованих моделей

	LDM	GALIP
Тривалість однієї епохи	~15 хвилин	~ 50 хвилин
Кількість епох	400	100
Розмір даних (батчу)	256	256

Таблиця 3.3 – Швидкість і кількість параметрів моделей

	LDM	GALIP
Загальна кількість параметрів всіх компонентів	~89M	~320M
Кількість параметрів, що навчалась	~24 M	~ 160 M
Час генерації зображення	5с	1с

Отже, реалізована модель LDM навчалась швидше, адже має меншу кількість параметрів для навчання і навчає лише одну мережу, а не дві як в GALIP, але генерація зображення на 4 секунди довша. Обидві моделі можна далі навчати збільшуючи тренувальний набір і т. д. для покращення якості результату, але через обмежені ресурси для тренування було залишено на найбільш оптимальному етапі.

3.6 Оцінка якості результатів

Щоб оцінити і порівняти результати генерації обох моделей було згенеровані зображення за однаковим текстовим описом по 8 зразків (повні результати винесені в Додаток А). Для генерації текстові описи підбирались більш загальні, тобто не мало багато конкретики аби переглянути різноманітність згенерованих зображень. На рисунках 3.7 і 3.8 зображені результати генерації на текстових описах «червоні і жовті квіти», «дівчина з собакою».

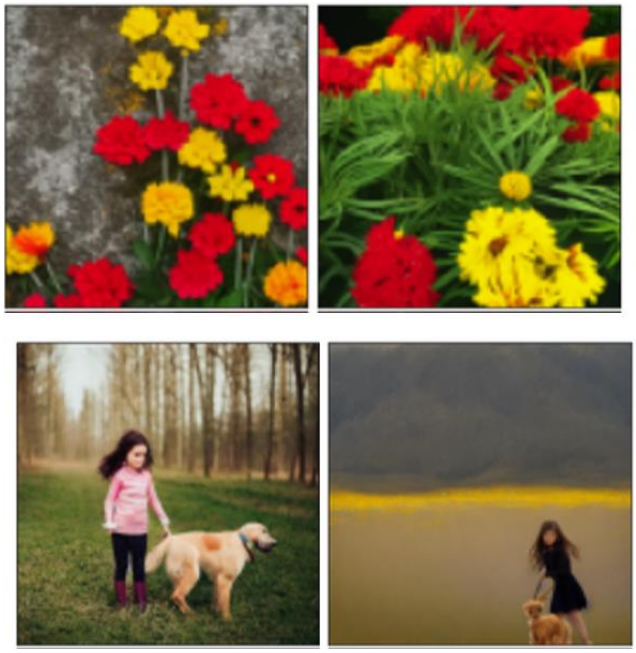

Рисунок 3.7 – Результати генерування за текстовими описами в LDM

Рисунок 3.8 – Результати генерування за текстовими описами в GALIP

Як видно з рисунків вище і рисунків з додатку А LDM генерує більш чіткі і різноманітні зображення у порівнянні з GALIP. У натренованої моделі GALIP виникла проблема mode collapse, яка була описана в

підрозділі 2.1.7. Всі зображення за одним текстовим описом схожі одне на одного.

Для кількісного порівняння було пораховано метрики FID і IS (таблиця 3.4) описані в підрозділі 3.3. Було згенеровано 5 тисяч зображень за текстовими описами з набору даних COCO і для прорахування FID використовувались реальні дані з цього ж набору за цими описами.

Таблиця 3.4 – Кількісна оцінка якості моделі

Метрика	LDM	GALIP	Реальні зображення
Frechet Inception Distance	157.49	170.84	6.09
Inception score	2.42	1.97	34.88

Отже, кількісні метрики теж показали що реалізована LDM дає кращі результати ніж GALIP.

3.7 Перспективи розвитку дифузійних моделей

Останніми роками дифузійна модель (DM) стала найсучаснішою парадигмою для перетворення тексту в зображення. І так як активні розробки і дослідження відбувалась в останні два-три роки, вони ще можливо не розкрили свій потенціал до кінця.

На оригінальній DDPM базується більшість вдосконалень і алгоритмів застосування. Але варто приймати дифузійні моделі як клас а не розгалуження від DDPM і подальші значні дослідження проводити у вивченні попереднього розподілу, перехідного ядра, методу вибірки та принципу дифузії.

Однією з можливих сфер інновацій є розробка складніших архітектур дифузійних моделей. Наприклад, можна дослідити використання більш

досконалих механізмів уваги, таких як самоувага, щоб покращити здатність моделі фіксувати довгострокові залежності в зображеннях.

За останні кілька років глибинне навчання значно просунулося вперед у ряді галузей, включаючи маскований автокодувальник у самоконтрольованому навчанні та сучасний ChatGPT у сфері обробки природної мови. Цікавим напрямком буде дослідити їх поєднавши з моделлю дифузії в сучасних дослідницьких галузях.

Оскільки основною функцією перетворення тексту в зображення є створення зображень з тексту, його можна вважати підмножиною мультимодального навчання. Більшість досліджень зосереджуються на одному завданні – створенні зображень з тексту, однак об'єднання кількох завдань в одній моделі може бути корисним.

ВИСНОВКИ

У кваліфікаційні роботі було проведено аналіз галузі генерації зображень і існуючі моделі для вирішення цієї задачі. Проведено аналіз генеративних моделей, а саме генеративних змагальних моделей, варіаційних автокодувальників, моделей на базі потоку, дифузійних моделей. Аналіз включає опис архітектур цих мереж їх переваги і недоліки, порівняння з іншими. Поставлені межі цієї роботи і план задач успішно дотримано і виконано.

Більш детально було описано різні архітектурні рішення і модифікації дифузійних моделей, які набувають все більшого значення в широкому спектрі прикладних галузей. Розглянуто DDPM, DDIM, LDM і фреймворки в піксельному і латентному просторах.

Було побудовано і навчено латентну дифузійну модель і GAN на основі CLIP. В результаті порівняння отримано, що якісні і кількісні оцінки (FID і IS) кращі в LDM, але вона довше генерує зображення, що може бути проблемою в онлайн генераціях. Описано перспективи і напрямки розвитку дифузійних моделей.

Таким чином, виконана кваліфікаційна робота дозволяє теоретично і практично проаналізувати переваги і недоліки дифузійних моделі порівняно з генеративними змагальними мережами для генерації зображень з текстового опису, оцінити практичні можливості цих моделей в умовах обмежених ресурсів.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. Elasri M., Elharrouss O., Al-Maadeed S., Tairi H. Image Generation: A Review. *Neural Processing Letters 54*. 2022. P. 4609–4646.

2. Zhou R., Jiang C., Xu Q. A survey on generative adversarial network-based text-to-image synthesis. Neurocomputing. 2021. Vol. 451. P. 316–336.

3. Generative adversarial networks / I. Goodfellow et al. *NIPS*. 2014. P. 2672–2680

4. Generative adversarial text to image synthesis / S. Reed et al. In Proceedings of The 33rd International Conference on Machine Learning. 2016. P. 1060-1069.

5. Weng L. From GAN to WGAN. 2019. URL: https://arxiv.org/abs/1904.08994 (Last accessed: 23.03.2023).

6. Bias and generalization in deep generative models: An empirical study / S. Zhao et al. Advances in Neural Information Processing Systems 31: Annual Conference on Neural Information Processing Systems 2018 Montreal, Canada. 2018. P. 10815–10824.

7. Roth K., Lucchi A., S. Nowozin, and T. Hofmann. Stabilizing training of generative adversarial networks through regularization. Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, Long Beach, CA, USA, December 4– 9. 2017. P. 2018–2028.

8. Hinton G. E. Reducing the dimensionality of data with neural networks. Science. 2006. Vol. 313, no. 5786. P. 504–507.

9. Kingma D. P., Welling M. Introduction to Variational Autoencoders. Now Publishers, 2019.

10. Kingma D. P., Welling M. Auto-Encoding Variational Bayes.URL: https://arxiv.org/abs/1312.6114 (Last accessed: 23.03.2023).

11. Dinh L., Krueger D., Bengio Y. Nice: Non-linear independent components estimation, 2014. URL: https://arxiv.org/abs/1410.8516 (Last accessed: 23.03.2023).

12. D. J. Rezende and S. Mohamed. Variational inference with normalizing flows. In Proceedings of the 32nd International Conference on Machine Learning, 52 ICML 2015, Lille, France. 2015. P. 1530–1538.

13. Tomczak J. M. Flow-Based Models. Deep Generative Modeling. Cham, 2021. P. 27–56.

14. Ho J., Jain A., Abbeel P. Denoising diffusion probabilistic models. Advances in Neural Information Processing Systems. 2020. P. 6840–6851.

15. Sohl-Dickstein J., Weiss E. A., Maheswaranathan N., Ganguli S. Deep unsupervised learning using nonequilibrium thermodynamics. In Proceedings of the 32nd International Conference on Machine Learning. 2015. P. 2256–2265.

16. Glide: Towards photorealistic image generation and editing with text-guided diffusion models / A. Nichol et al. 2021. URL: https://arxiv.org/abs/2112.10741 (Last accessed: 23.03.2023).

17. Hierarchical text-conditional image generation with clip latents / A. Ramesh. 2022. URL: https://arxiv.org/abs/2204.06125 (Last accessed: 23.03.2023).

18. Photorealistic text-to-image diffusion models with deep language understanding / C. Saharia et al. 2022. URL: https://arxiv.org/abs/2205.11487 (Last accessed: 23.03.2023).

19. Song Y., Ermon S. Generative modeling by estimating gradients of the data distribution. 2019. URL: https://arxiv.org/abs/1907.05600 (Last accessed: 23.03.2023).

20. Sohl-Dickstein J., Weiss E., Maheswaranathan N., Ganguli S. Deep unsupervised learning using nonequilibrium thermodynamics. 2015. P. 2256–2265.

21. Song Y., Ermon S. Improved techniques for training score-based generative models. Advances in neural information processing systems. 2020. P. 12 438–448. URL: https://arxiv.org/abs/2006.09011 (Last accessed: 23.03.2023).

22. Rombach R., Blattmann A., Ommer B. Text-Guided Synthesis of Artistic Images with Retrieval-Augmented Diffusion Models. 2022. URL: https://arxiv.org/abs/2207.13038 (Last accessed: 23.03.2023).

23. Mansimov E., Parisotto E., Ba L. J., Salakhutdinov R. Generating images from captions with attention. In 4th International Conference on Learning Representations, ICLR 2016, San Juan, Puerto Rico. 2016.

24. Attngan: Fine-grained text to image generation with attentional generative adversarial networks / T. Xu et al. In 2018 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2018, Salt Lake City, UT, USA. 2018. P. 1316–1324.

25. Zhu M., Pan P., Chen W., Yang Y. DM-GAN: dynamic memory generative adversarial networks for text-to-image synthesis. In IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2019, Long Beach, CA, USA. 2019. P. 5802–5810.

26. A simple and effective baseline for text-to-image synthesis / M. Tao et al. 2020. URL: https://arxiv.org/abs/2008.05865 (Last accessed: 23.03.2023).

27. Cross-modal contrastive learning for text-to-image generation / H. Zhang et al. 2021. URL: https://arxiv.org/abs/2101.04702 (Last accessed: 23.03.2023).

28. Zero-shot text-to-image generation / A. Ramesh et al. 2021. URL: https://arxiv.org/abs/2102.12092 (Last accessed: 23.03.2023).

29. Oord A. V. D., Vinyals O., Kavukcuoglu K. Neural discrete representation learning. In Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, Long Beach, CA, USA. 2017. P. 6306–6315.

30. Dhariwal P., Nichol A. Diffusion models beat gans on image synthesis. Advances in Neural Information Processing Systems. 2021.

31. Glide: Towards photorealistic image generation and editing with text-guided diffusion models / A. Nichol et al. 2021. URL: https://arxiv.org/abs/2112.10741 (Last accessed: 23.03.2023).

32. Hierarchical text-conditional image generation with clip latents / A. Ramesh et al. 2022. URL: https://arxiv.org/abs/2204.06125 (Last accessed: 23.03.2023).

33. Photorealistic text-to-image diffusion models with deep language understanding / C. Saharia et al. 2022. URL: https://arxiv.org/abs/2205.11487 (Last accessed: 23.03.2023).

34. Learning transferable visual models from natural language supervision / A. Radford et al. 2021. URL: https://arxiv.org/abs/2103.00020 (Last accessed: 23.03.2023).

35. Cascaded diffusion models for high fidelity image generation / J. Ho. 2021. URL: https://arxiv.org/abs/2106.15282 (Last accessed: 23.03.2023)

36. Exploring the limits of transfer learning with a unified text-to-text transformer / C. Raffel et al. 2019. URL: https://arxiv.org/abs/1910.10683 (Last accessed: 23.03.2023)

37. Microsoft COCO: Common Objects in Context / T.-Y. Lin et al. Computer Vision – ECCV 2014. Cham, 2014. P. 740–755. URL: https://doi.org/10.1007/978-3-319-10602-1_48 (Last accessed: 23.03.2023).

38. Adversarial text-to-image synthesis: A review / S. Frolov et al. Neural Networks. 2021. Vol. 144. P. 187–209.

39. Dhivya K., Sharfaras Navas N. Text to Realistic Image Generation Using Stackgan. 2020 7th International Conference on Smart Structures and Systems (ICSSS), Chennai, India, 23–24 July 2020. 2020

40. GALIP: Generative Adversarial CLIPs for Text-to-Image Synthesis / M. Tao et al. 2023. URL: https://arxiv.org/pdf/2301.12959.pdf (Last accessed: 23.03.2023).

41. Diffusion Models in Vision: A Survey / F.-A. Croitoru et al. IEEE Transactions on Pattern Analysis and Machine Intelligence. 2023. P. 1–20.

ДОДАТОК А
Результати генерування натренованих моделей

А.1 Результати генерації зображень натренованою LDM

Рисунок А.1 – Результати генерації для текстового опису «a village in the mountains in spring»

Рисунок А.2 – Результати генерації для текстового опису «red and yellow flowers»

Рисунок А.3 – Результати генерації для текстового опису «a large number of cows in the field»

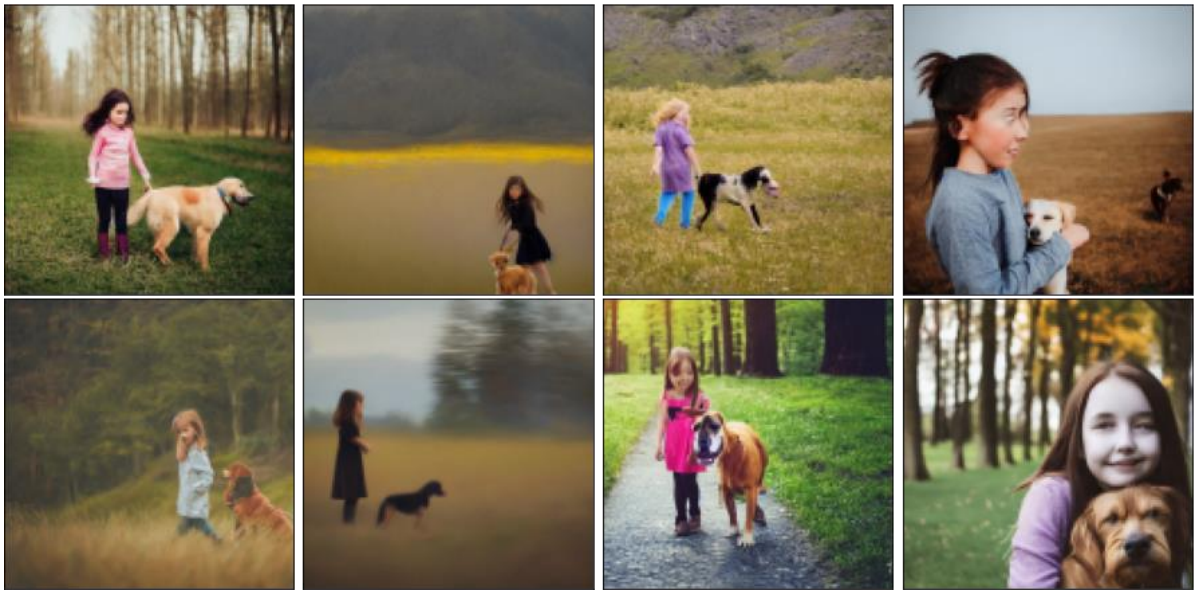

Рисунок А.4 – Результати генерації для текстового опису «girl with dog»

Рисунок А.5 – Результати генерації для текстового опису «only one horse in the forest»

Рисунок А.6 – Результати генерації для текстового опису «small blue bird»

А.2 Результати генерації зображень натренованою GAN

Рисунок А.7 – Результати генерації для текстового опису «a village in the mountains in spring»

Рисунок А.8 – Результати генерації для текстового опису «red and yellow flowers»

Рисунок А.9 – Результати генерації для текстового опису «a large number of cows in the field»

Рисунок А.10 – Результати генерації для текстового опису «girl with dog»

Рисунок А.11 – Результати генерації для текстового опису «only one horse in the forest»

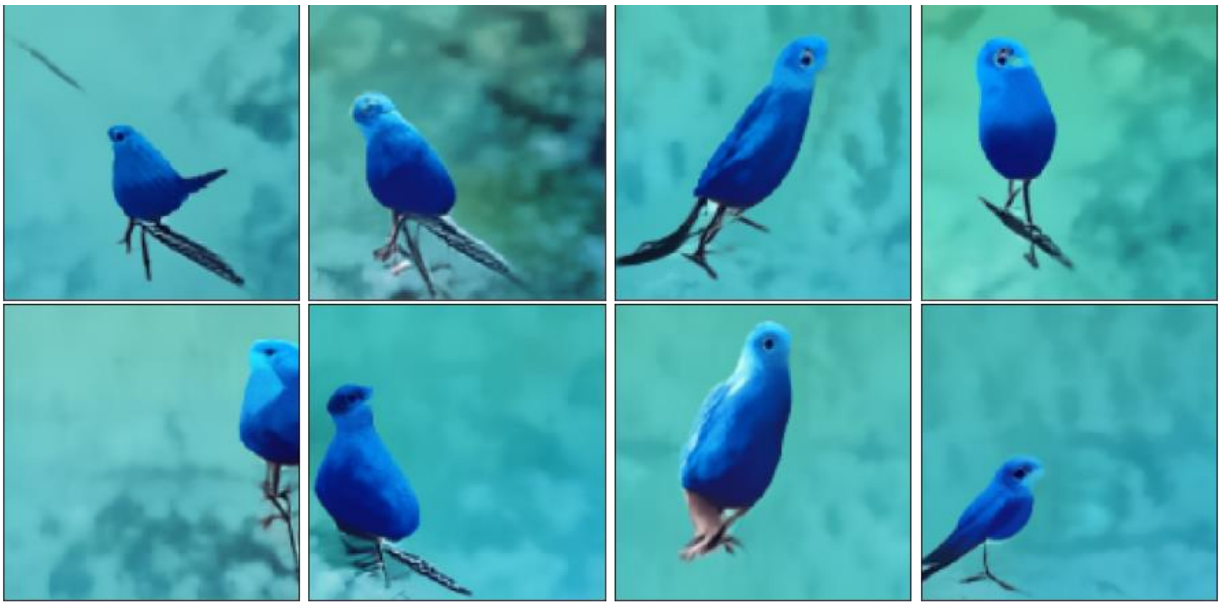

Рисунок А.12 – Результати генерації для текстового опису «small blue bird»

ДОДАТОК Б

Відомість кваліфікаційної роботи

Позначення	Найменування	Дод. відомості
	Текстові документи	
1.	Пояснювальна записка	84 с.
	Інші документи	
2.	Презентаційні матеріали	32 плакатів

.	.	Прізвище та ініціали.	Підп.	Дата			
Розробив		Овсієнко В.С.			Дослідження дифузійних моделей для генерації зображень у порівнянні з генеративними змагальними мережами	Шифр групи	Код напр./спец.
Перевірив		Семенець В.В.				СШІм-21-1	122
Н.контр.		Малєєва І.А.				ХНУРЕ кафедра ШІ	
Затв.		Філатов В.О.					