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 перший (бакалаврський)

Застосування дифузних нейронних мереж для генерації зображень
(тема)

Виконав:
здобувач четвертого року навчання,
групи ІТШІ-21-4
Павло Чістяков
(власне ім'я, прізвище)
Спеціальність 122 Комп'ютерні науки
(код і повна назва спеціальності)
Тип програми освітньо-професійна
Освітня програма Штучний інтелект
(повна назва освітньої програми)
Керівник доц. Олег Золотухін
(посада, власне ім'я, прізвище)

Допускається до захисту

Завідувач кафедри ШІ _____ Олег ЗОЛОТУХІН
(підпис) (власне ім'я, прізвище)

2025 р.

Харківський національний університет радіоелектроніки

Факультет Комп'ютерних наук

Кафедра Штучного інтелекту

Рівень вищої освіти перший (бакалаврський)

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

Тип програми освітньо-професійна

Освітня програма Штучний інтелект
(повна назва)

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

ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ РОБОТУ

здобувачеві Чістякову Павлу Вадимовичу
(прізвище, ім'я, по батькові)

1. Тема роботи Застосування дифузних нейронних мереж для генерації зображень

затверджена наказом університету від 19 травня 20 25 р. № 378Ст
2. Термін подання студентом роботи до екзаменаційної комісії 18 червня 20 25 р.
3. Вихідні дані до роботи Python, PyTorch, Hugging Face Diffusers, LoRA, Stable Diffusion, AutoencoderKL, UNet2DConditionModel, DDPMScheduler, Accelerate, Transformers, torchvision, PIL, Matplotlib, Docker

4. Перелік питань, що потрібно опрацювати в роботі
1) Аналіз предметної галузі
2) Постановка задачі
3) Характеристика інструментів та технологій розробки
4) Розробка програмного забезпечення

КАЛЕНДАРНИЙ ПЛАН

№	Назва етапів роботи	Строк / терміни виконання етапів роботи	Примітка
1	Отримання завдання на кваліфікаційну роботу	19.05.2025	виконано
2	Аналіз предметної галузі	20.05–22.05	виконано
3	Постановка задачі	23.05–25.05	виконано
4	Обґрунтування вибору використання технологій	26.05–29.05	виконано
5	Програмна реалізація	30.05–04.06	виконано
6	Оформлення пояснювальної записки	05.06–13.06	виконано
7	Підготовка до захисту	14.06–17.06	виконано
8	Захист перед ЕК	18.06.2025	виконано

Дата видачі завдання 19 травня 2025 р.

Здобувач _____
(підпис)

Керівник роботи _____ доц. Олег Золотухін
(підпис) (посада, власне ім’я, прізвище)

РЕФЕРАТ

Пояснювальна записка: 66 с., 11 рис., 1 дод., 19 джерел.

ГЕНЕРАТИВНЕ МИСТЕЦТВО, МАШИННЕ НАВЧАННЯ, НЕЙРОННІ МЕРЕЖІ, ОБРОБКА ЗОБРАЖЕНЬ, СТАБІЛЬНА ДИФУЗІЯ, ТОНКЕ НАВЧАННЯ, GPU, LORA, PYTHON.

Об’єкт дослідження – процес тонкого донавчання моделей стабільної дифузії з урахуванням художніх стилів на основі малих вибірок зображень.

Предмет дослідження – методика використання Low-Rank Adaptation (LoRA) для створення унікальних стильових моделей на базі Stable Diffusion із застосуванням оптимізаційних технік та зменшенням вимог до пам’яті GPU.

Мета роботи – дослідити підходи до тонкого донавчання (fine-tuning) генеративної моделі Stable Diffusion методами LoRA, обґрунтувати вибір програмних інструментів і методів оптимізації для зменшення апаратних ресурсів, а також оцінити якість отриманих зображень у контексті візуальної відповідності заданому стилю.

Методи дослідження – аналіз сучасних публікацій у сфері генеративних нейронних мереж та перехідних шарів LoRA, експериментальне моделювання з використанням бібліотек PyTorch і Diffusers, оптимізація параметрів за допомогою зменшення точності (FP16) і градієнтної акумуляції, моделювання архітектури навчання та тестування на реальних прикладах.

У межах практики сформульовано функціональні вимоги до процесу тонкого донавчання моделі Stable Diffusion, обґрунтовано вибір мови Python та бібліотеки Diffusers для реалізації навчання, а також застосування LoRA для ефективного пристосування моделі до невеликих датасетів. Розроблено прототип, здатний генерувати зображення в індивідуальному стилі.

ABSTRACT

Bachelor’s thesis contains: 66 pp., 11 fig., 1 ann., 19 references.

FINE LEARNING, GENERATIVE ART, GPU, IMAGE PROCESSING, LORA, MACHINE LEARNING, NEURAL NETWORKS, PYTHON, STABLE DIFFUSION.

Object of research – the process of fine-tuning Stable Diffusion models with consideration of artistic styles based on small image samples.

Subject of research – the methodology for using Low-Rank Adaptation (LoRA) to create unique style models on the basis of Stable Diffusion, employing optimization techniques and reducing GPU memory requirements.

Aim of the work – to investigate approaches to fine-tuning the Stable Diffusion generative model using LoRA methods, justify the selection of software tools and optimization methods to reduce hardware resource demands, and evaluate the quality of the resulting images in terms of visual conformity to the specified style.

Research methods – analysis of recent publications in the field of generative neural networks and LoRA adapters, experimental modeling using the PyTorch and Diffusers libraries, parameter optimization through precision reduction (FP16) and gradient accumulation, modeling of the training architecture and testing on real-world examples.

Within the practical framework, functional requirements for the Stable Diffusion fine-tuning process were formulated, the choice of Python and the Diffusers library for implementation was justified, and the application of LoRA for efficient adaptation of the model to small datasets was substantiated. A prototype capable of generating images in an individual style was developed.

ЗМІСТ

Перелік умовних позначень, символів, одиниць, скорочень і термінів 8
Вступ 9
1 Аналіз предметної галузі 10
1.1 Вступ до предметної галузі 10
1.2 Актуальність 11
1.3 Існуючі підходи до тонкого донавчання Stable Diffusion та їхні обмеження 12
1.4 Методи машинного навчання та штучного інтелекту в побудові генеративних систем 15
1.5 Архітектурні особливості дифузійних моделей та їх еволюція 17
1.6 Принципи роботи Low-Rank Adaptation та математичні основи 19
1.7 Порівняльний аналіз методів генерації зображень 21
1.8 Текстово-керована генерація зображень та роль conditional generation 23
1.9 Оцінка якості генерованих зображень та метрики 26
1.10 Застосування генеративних моделей у творчих індустріях 28
1.11 Технічні виклики та обмеження сучасних дифузійних моделей 29
2 Постановка задачі 31
3 Теоретичні дослідження 36
3.1 Загальні положення 36
3.2 Вибір стеку технологій для розробки і впровадження оптимізованого коду 37
3.3 Особливості оптимізації латентного простору для тонкого донавчання 42
3.4 Методи регуляризації та стабілізації навчання LoRA-адаптера 43
3.5 Динамічне поєднання LoRA та умовного керування стилем 44
3.6 Розвиток генеративного штучного інтелекту 45
3.7 Принцип роботи Stable Diffusion 48
3.8 Візуалізація процесу дифузії та роль U-Net 50

3.9 Генеративні архітектури для генерації зображень 54
4 Розробка програмного забезпечення..... 56
4.1 Архітектура та основна реалізація 57
4.2 Оптимізації, генерація результатів та перспективи..... 59
Висновки 61
Перелік джерел посилання 63
Додаток А Відомість кваліфікаційної роботи 66

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СИМВОЛІВ, ОДИНИЦЬ, СКОРОЧЕНЬ І ТЕРМІНІВ

CLIP – Contrastive Language–Image Pretraining – контрастне попереднє навчання текст-зображення;

DALL·E – текст-до-зображення модель OpenAI;

DBSCAN – Density-Based Spatial Clustering of Applications with Noise – кластеризація з врахуванням шуму;

DDIM – Denoising Diffusion Implicit Models – детерміністичний семплер;

EWC – Elastic Weight Consolidation – еластична консолідація ваг;

FID – Fréchet Inception Distance – фрешева дистанція між розподілами;

FP16 – 16-бітна плаваюча точка з половинною точністю;

FP32 – 32-бітна плаваюча точка з одинарною точністю;

GAN – Generative Adversarial Network – генеративно-змагальна мережа;

IS – Inception Score – показник різноманітності та чіткості;

LoRA – Low-Rank Adaptation – адаптація низького рангу;

LPIPS – Learned Perceptual Image Patch Similarity – навчена перцептивна схожість патчів;

MSE – Mean Squared Error – середньоквадратична похибка;

PCA – Principal Component Analysis – аналіз головних компонент;

SD – Stable Diffusion – стабільна дифузія;

SSIM – Structural Similarity Index – індекс структурної подібності;

VAE – Variational Autoencoder (AutoencoderKL) – варіаційний автокодер;

VAE-G – гібрид β-VAE та GAN;

VAE-GAN – гібрид VAE та GAN;

UNet – UNet2DConditionModel – U-Net умовної моделі;

<artg> – унікальний токен стилю в текстовому промпті для LoRA.

ВСТУП

У сучасному цифровому середовищі все більшу популярність набувають генеративні нейронні мережі, які дозволяють створювати реалістичні та художньо виразні зображення. Модель Stable Diffusion, що репрезентує один із найсучасніших підходів до дифузійних процесів у комп’ютерному баченні, дає змогу якісно відтворювати візуальні сцени на основі текстових або стилістичних підказок. Проте стандартні моделі вимагають значних обчислювальних ресурсів і великих наборів даних, що ускладнює їхнє використання при створенні унікальних візуальних стилів для персональних чи вузькоспеціалізованих проєктів.

Одним із рішень цієї проблеми є використання підходу LoRA (Low-Rank Adaptation), який дає можливість зменшувати кількість параметрів при тонкому донавчанні моделі, зберігаючи при цьому якість одержуваних результатів. Завдяки цій методиці можна навчати модель на малих обсягах стилістичних даних, що робить процес тренування доступнішим для широкого кола дослідників та митців.

Метою даної роботи є проєктування та дослідження процесу тонкого донавчання Stable Diffusion із використанням LoRA на прикладі вузького набору зображень, що імітують унікальний художній стиль. У роботі розглянуто архітектуру моделі, оптимізаційні техніки для зниження навантаження на GPU, а також методи оцінювання отриманих результатів із позиції відповідності цільовому стилю. Розв’язання поставленої задачі сприятиме поліпшенню практичних аспектів у сфері генеративного мистецтва та формуванню підґрунтя для наступних досліджень у напрямі персоналізованих моделей зображень.

1 АНАЛІЗ ПРЕДМЕТНОЇ ГАЛУЗІ

1.1 Вступ до предметної галузі

Останнім часом у сфері генеративного машинного навчання спостерігається значне зростання інтересу до дифузійних моделей, що дають змогу створювати фотореалістичні або стилізовані зображення на основі текстових підказок чи прикладів. Технологія Stable Diffusion, яка вирізняється здатністю генерувати високоякісні візуальні результати, набуває дедалі більшого поширення серед митців, дослідників та представників креативної індустрії. Утім, базова реалізація подібних моделей зазвичай вимагає значних обчислювальних ресурсів і великих обсягів даних для первинного навчання.

На практиці часто постає потреба адаптувати потужну дифузійну модель до певного унікального стилю або специфічних вимог проєкту. При цьому доступні тільки невеликі датасети чи обмежена обчислювальна потужність. У такому випадку надзвичайно перспективним є підхід Low-Rank Adaptation (LoRA), що дозволяє скоротити обсяг параметрів, які потрібно змінювати під час донавчання. Завдяки цьому можна швидко та відносно недорого адаптувати вже наявну модель під потрібний візуальний стиль, уникаючи тривалого та ресурсоємного повторного навчання на всій вибірці.

Використання LoRA для тонкого донавчання Stable Diffusion відкриває нові можливості в галузі генеративного мистецтва, дизайнерських рішень та креативних експериментів. Ідеться про прискорення процесу налаштування моделі, економію апаратних ресурсів та одержання унікальних художніх результатів. Саме тому дослідження, спрямовані на вивчення й застосування подібних підходів, набувають особливої актуальності у сучасному світі цифрової візуалізації.

1.2 Актуальність

Актуальність розробки методології тонкого донавчання моделі Stable Diffusion за допомогою LoRA зумовлена інтенсивним розвитком генеративних нейронних мереж і зростанням потреби в унікальних та персоналізованих візуальних стилях. Сучасний ринок креативних технологій вимагає рішень, які забезпечують як високу якість отримуваних зображень, так і гнучкість налаштування під конкретні художні завдання. Водночас не всі проєкти можуть собі дозволити дороговартісне або багатомісячне перенавчання великих моделей на спеціалізованих, а іноді й досить малих вибірках даних.

Технологія LoRA пропонує дієвий механізм зниження обчислювальних витрат, що особливо важливо для незалежних дослідників, дизайнерів і малих креативних студій. Вона дає змогу швидко інтегрувати новий стиль у модель Stable Diffusion без потреби у високопродуктивних графічних процесорах або величезних датасетах. Такий підхід розширює можливості використання дифузійних моделей у комерційних і академічних контекстах, роблячи їх доступними для широкого кола користувачів. Результат генерації з використанням запиту (рисунок 1.1): «Draw a dark horse in the style of <artg> with lots of detail. A horse on a dark watercolor background.»

Рисунок 1.1 – Згенероване зображення з використанням запиту

Подібні розробки створюють основу для широкомасштабної інтеграції генеративного мистецтва в освітні, дослідницькі та промислові проєкти. Тонке донавчання дифузійних моделей, побудованих на ідеї LoRA, може слугувати інструментом для створення нових форм художнього самовираження, адаптації під різні дизайнерські запити та навіть експериментів у наукових дослідженнях. Усе це робить описаний підхід надзвичайно затребуваним і перспективним у сучасному інформаційному середовищі.

1.3 Існуючі підходи до тонкого донавчання Stable Diffusion та їхні обмеження

У сучасних дослідженнях, присвячених генеративним нейронним мережам, дедалі більшої популярності набувають дифузійні моделі, які дають змогу одержувати високоякісні зображення на основі послідовного внесення й усунення шуму в латентному просторі. Модель Stable Diffusion, що представляє один із найпоширеніших прикладів таких підходів, історично навчалася на масштабних вибірках і здатна відтворювати складні візуальні сцени з високим ступенем деталізації. Проте базове донавчання цієї моделі потребує значного обсягу обчислювальних ресурсів і чималих обсягів даних, що у багатьох випадках стає критичним обмеженням. Користувачі та дослідники часто прагнуть адаптувати Stable Diffusion під специфічні вимоги, зокрема під новий стиль або конкретні сюжетні завдання, але не завжди володіють необмеженою кількістю зображень чи достатньою GPU-пам’яттю.

Існують декілька поширених методик перенавчання дифузійних моделей, що пропонують варіанти зосередження саме на «стильовому» аспекті. Однією з ранніх технік є Textual Inversion, котра полягає у створенні додаткових «токенів», які в процесі навчання пов’язуються з певним візуальним стилем чи концептом [1]. Такий підхід дає змогу генеративній

системі «засвоїти» нові поняття або стилізовані елементи без повного перенавчання всіх ваг. Утім, Textual Inversion має суттєве обмеження у вигляді складнощів із відтворенням комплексних деталей чи структур, якщо задана вибірка зображень є надто малою або надто різнорідною. Інший відомий метод, DreamBooth, дає змогу «вшивати» конкретний образ чи персонажа у вже навчену модель, але також вимагає певного мінімального обсягу даних і ресурсоємного процесу: для переконливого відтворення складних стилів може знадобитися велика кількість ітерацій із суттєвим навантаженням на GPU. Під час такого навчання часто застосовується функція втрат, що порівнює згенероване зображення з оригіналом у різних просторах представлення (наприклад, у просторах векторів ознак або перцептивних відстаней). При цьому оцінка якості результату може бути формалізована через мінімізацію (1.1):

$$L(\theta) = 1/N \sum_{i=1}^{N} \left|\left|F(x_i;\theta) - y_i\right|\right|^2, \quad (1.1)$$

де x – вхідне стилізоване зображення або його латентне представлення;

y – цільова стилізована версія;

$F(x_i;\theta)$ – вихід моделі з параметрами θ.

Попри те, що вказані методи (Textual Inversion, DreamBooth тощо) можуть досить ефективно розширювати «словник» Stable Diffusion, вони часто стикаються з обмеженнями, пов’язаними з витратами часу, пам’яті та надійності результатів на різних вибірках [6].

При намаганні адаптувати модель до дуже специфічного або унікального стилю, який має лише кілька доступних зразків, традиційні підходи дають значні похибки або занадто узагальнений результат. Важливо й те, що через велику кількість параметрів мережі (сотні мільйонів і більше) повне або навіть часткове перенавчання спричиняє надмірне використання апаратних ресурсів [1].

На противагу цьому, методика LoRA (Low-Rank Adaptation) пропонує більш гнучкий і ресурсозберігальний спосіб перенавчання. Згідно з ідеєю розкладання великих матриць ваг у ранг-дефіцитні компоненти, оновленню під час навчання підлягають лише невеликі «вставки» рангу r, тоді як основна частина параметрів залишається незмінною. Такий підхід дає можливість зменшити не лише обсяг операцій, необхідних для перенавчання, але й ризик «забуття» оригінального функціоналу моделі. Формально, якщо припустити, що певні вагові матриці дифузійної моделі можна представити у вигляді

$$W = W0 + UV^T, \tag{1.2}$$

де W0 – незмінний блок базових параметрів, обчислювальна складність та потреба в пам’яті залежать переважно від розмірів матриць U і V, а не від усього W. Отже, LoRA дає змогу швидко й ефективно переналаштовувати Stable Diffusion для нових художніх стилів навіть за невеликих датасетів, що було б недоступно при повноцінному «перенавчанні з нуля» [2].

Таким чином, існуючі підходи до адаптації дифузійних моделей охоплюють як традиційні техніки (Textual Inversion, DreamBooth), так і новітні методи на кшталт LoRA, кожен із яких має свої переваги й недоліки. Потреба в узгодженні якості, часу навчання та обсягу необхідних обчислювальних ресурсів стимулює розвиток оптимізованих моделей, здатних навчатися на незначних вибірках і видавати якісні результати.

Однак без чіткого врахування обмежень і особливостей конкретного методу неможливо досягти балансу між точністю стилізації та ефективністю розгортання системи. Саме на розв’язання цієї проблеми й орієнтується запропонований у цій роботі підхід, що дозволяє отримати унікальний візуальний стиль через механізм часткового перенавчання ваг дифузійної моделі за допомогою LoRA та токенів стилю як <artg>.

1.4 Методи машинного навчання та штучного інтелекту в побудові генеративних систем

Сучасні підходи до створення генеративних моделей, зокрема для задач формування зображень, значною мірою базуються на глибокому машинному навчанні, яке передбачає використання багаторівневих штучних нейронних мереж. Такі методи дають змогу виявляти приховані структури у великих обсягах даних та будувати як складні регресійні, так і ймовірнісні залежності у високорозмірних просторах ознак. Основу генеративних моделей становлять ідеї, що дозволяють моделювати розподіли ймовірностей над об’єктами, як-от зображеннями чи аудіо, з метою подальшої вибірки (семплінгу) нових даних, які відтворюють статистичні властивості вихідних датасетів. Класичний підхід до опису генеративного процесу можна подати через сумісний розподіл

$$p_\theta(x, y) = p_\theta(y \mid x)p_\theta(x), \tag{1.3}$$

де x у багатьох випадках відповідає деякому шуму чи «контексту», а y – результату, тобто новому зображенню. У контексті дифузійних моделей часто розглядається неявне наближення цього розподілу за допомогою послідовного «зашумлення» та «дешумування», що має форму ітераційних рівнянь.

На практиці методи генеративного навчання можуть належати як до родини автокодировачів (VAE), так і включати механізми зворотніх переходів у просторі шуму (Diffusion, Score-based підходи), або ж використовувати змагальну схему (GAN). Для кожного напряму характерні власні сильні та слабкі сторони: VAE забезпечують безпосередній «стохастичний» простір латентних змінних, GAN здатні генерувати дуже реалістичні зображення, але вирізняються складністю налаштування, тоді як дифузійні моделі пропонують сталий і послідовний спосіб формувати

структуру зображення від високошумових станів до детальних результатів [4].

У процесі розробки та тонкого донавчання Stable Diffusion ключову роль відіграє ідея роботи з латентними просторами та застосування допоміжних перетворень для перетворення вхідної вибірки у вигляді шуму. Нехай *x0* є початковим зображенням, а x – його «зашумлена» версія на кроці t. Згідно з дифузійною моделлю, що описана розподілом

$$q(x_t \mid x_{t-1}) = N\left(\sqrt{(1-\beta_t)}x_{t-1}, \beta_t I\right). \quad (1.4)$$

мета полягає у відтворенні відсутньої інформації та покроковому наближенні до «чистого» зображення. Для цього будується апарат «оберненої» моделі, параметризованої нейронною мережею, що передбачає «знешумлені» версії, відзначені як:

$$p_\theta(x_{t-1} \mid x_t). \quad (1.5)$$

Застосування таких схем дозволяє при достатньому навчанні апроксимувати цільовий розподіл зображень і генерувати нові приклади. Водночас алгоритми оптимізації можуть орієнтуватися на мінімізацію середньоквадратичної похибки або інших функцій втрат між згенерованими та реальними прикладами.

Різниця між генеративними та дискримінативними алгоритмами стає особливо відчутною під час адаптації моделей до нових стилів. Для дифузійних мереж, що є типовим прикладом генеративної парадигми, важливо не просто класифікувати чи аналізувати вихідні дані, а навчитися відтворювати їхню внутрішню структуру. Фактичний процес донавчання можна розглядати як пошук оптимальних параметрів θ, які мінімізують похибку відтворення підмножини особливих зразків у загальному просторі ознак. З урахуванням LoRA, цей пошук обмежується ранг-дефіцитними

векторами, що забезпечує як швидкість навчання, так і більш точне «проникнення» у стиль [3].

Отже, у рамках машинного навчання й штучного інтелекту дифузійні моделі поєднують у собі елементи статистичної фізики (зашумлення), класичних ідей навчання на великих даних (пошук оптимального набору параметрів) та принципів відновлення сигналів (покрокове «очищення»). Використання LoRA при побудові та адаптації таких моделей дає змогу гнучко балансувати між ресурсами й бажаною якістю, а також ефективно враховувати невеликі обсяги унікальних даних. Це розширює горизонти застосувань генеративних систем у різноманітних галузях – від створення художніх об’єктів до наукових досліджень у зображальній аналітиці.

1.5 Архітектурні особливості дифузійних моделей та їх еволюція

Архітектура дифузійних моделей ґрунтується на концепції поступового додавання гаусівського шуму до даних з подальшим навчанням моделі відновлювати початкові дані з зашумлених версій. Цей підхід кардинально відрізняється від класичних генеративних архітектур, таких як варіаційні автокодувальники чи генеративно-змагальні мережі, оскільки передбачає моделювання зворотного процесу дифузії через багатокрокову процедуру денойзингу. Історично розвиток дифузійних моделей почався з робіт у сфері моделювання фізичних процесів, де дифузія розглядалася як поступове розсіювання концентрованої речовини в середовищі.

У контексті машинного навчання дифузійні моделі вперше були запропоновані як альтернативний спосіб генерації даних, що дозволяє уникнути проблем нестабільності навчання, характерних для генеративно-змагальних мереж. Основна ідея полягає в тому, що якщо ми знаємо, як поступово руйнувати структуру даних через додавання шуму, то можемо навчити модель виконувати зворотний процес і відновлювати структуру з

хаотичного стану. Це забезпечує більш стабільне та контрольоване генерування порівняно з іншими підходами.

Ключовою особливістю архітектури дифузійних моделей є використання U-Net архітектури з механізмами увагі, які дозволяють моделі фокусуватися на найбільш релевантних частинах зображення під час процесу денойзингу. U-Net складається з енкодерної частини, що поступово зменшує просторові розміри при збільшенні кількості каналів, та декодерної частини, яка відновлює початкові розміри. Пропускні з'єднання між відповідними рівнями енкодера та декодера забезпечують збереження деталей різних масштабів.

Механізми увагі в дифузійних моделях відіграють критично важливу роль, особливо при генерації зображень за текстовими описами. Cross-attention шари дозволяють моделі узгоджувати візуальну інформацію з текстовими embeddings, що забезпечує точне відтворення описаних об'єктів та сцен. Self-attention механізми допомагають моделі враховувати взаємозв'язки між різними частинами зображення, що є особливо важливим для створення когерентних та реалістичних результатів.

Темпоральне кодування також становить важливий компонент архітектури, оскільки модель повинна знати, на якому кроці процесу денойзингу вона знаходиться. Для цього використовуються sinusoidal position embeddings або навчені embedding векторі, які додаються до внутрішніх представлень мережі. Це дозволяє моделі адаптувати свою поведінку залежно від рівня шуму у вхідних даних.

Еволюція дифузійних моделей включає декілька ключових етапів розвитку. Перші моделі працювали безпосередньо в пікселному просторі, що вимагало значних обчислювальних ресурсів для високорозмірних зображень. Подальший розвиток привів до створення латентних дифузійних моделей, які виконують процес генерації в низькорозмірному латентному просторі, використовуючи попередньо навчений автокодувальник для перетворення між пікселним та латентним представленнями.

Латентні дифузійні моделі, до яких належить Stable Diffusion, демонструють значно кращу ефективність завдяки зменшенню розмірності робочого простору. Автокодувальник складається з енкодера, який стискає зображення до латентного представлення, та декодера, який відновлює зображення з латентного коду. Такий підхід дозволяє зберегти перцептивно важливу інформацію при значному зменшенні обчислювальних витрат.

Подальший розвиток дифузійних моделей включав удосконалення алгоритмів семплінгу, розробку детерміністичних методів генерації та створення технік прискорення. Зокрема, DDIM семплери дозволяють генерувати зображення за меншу кількість кроків без суттєвого погіршення якості, що робить процес генерації більш практичним для реального використання.

1.6 Принципи роботи Low-Rank Adaptation та математичні основи

Low-Rank Adaptation представляє інноваційний підхід до тонкого донавчання великих нейронних мереж, що базується на гіпотезі про те, що зміни ваг під час адаптації до нових завдань можуть бути представлені в низькоранговому підпросторі (рисунок 1.2). Цей принцип дозволяє drastично зменшити кількість параметрів, які потрібно оновлювати під час донавчання, зберігаючи при цьому ефективність адаптації.

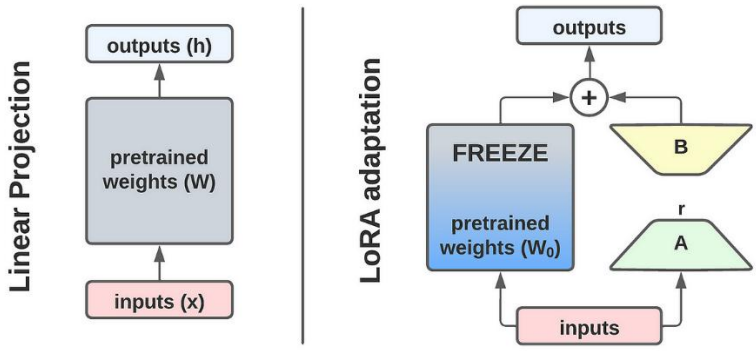

Рисунок 1.2 – Принципи роботи Low-Rank Adaptation

Математична основа LoRA полягає в декомпозиції матриці ваг на добуток двох матриць меншого рангу. Якщо початкова матриця ваг має розмірність d×k, то замість її повного оновлення створюються дві допоміжні матриці A розмірності d×r та B розмірності r×k, де r значно менше за min(d,k). Оновлена матриця ваг обчислюється як сума початкової матриці та добутку допоміжних матриць.

Ініціалізація матриць A та B відбувається по-різному: матриця A зазвичай ініціалізується випадковими значеннями з гаусівського розподілу, тоді як матриця B ініціалізується нулями. Такий підхід забезпечує, що на початку навчання LoRA адаптер не впливає на вихід моделі, дозволяючи поступове навчання без порушення початкової функціональності.

Ранг декомпозиції r є критично важливим гіперпараметром, який визначає баланс між якістю адаптації та ефективністю. Менший ранг призводить до більшої економії пам'яті та швидшого навчання, але може обмежити здатність моделі адаптуватися до складних паттернів. Більший ранг забезпечує кращу адаптацію, але зменшує переваги методу в плані ефективності.

Процес навчання LoRA адаптерів включає стандартну оптимізацію з використанням градієнтного спуску, але градієнти обчислюються тільки відносно параметрів матриць A та B, тоді як початкові ваги залишаються замороженими. Це дозволяє зберегти знання, набуті під час попереднього навчання, та запобіги катастрофічному забуванню.

Альфа-параметр у LoRA виконує роль масштабуючого коефіцієнта, який контролює силу впливу адаптера на початкову модель. Цей параметр зазвичай встановлюється пропорційно до рангу та дозволяє налаштовувати інтенсивність адаптації без зміни архітектури або переобчислення ваг.

Теоретичне обґрунтування ефективності LoRA базується на спостереженні, що в багатьох випадках оновлення ваг під час тонкого донавчання мають низький внутрішній ранг. Це означає, що хоча матриця

оновлень може мати великі розміри, вона може бути точно апроксимована добутком матриць меншого рангу без суттєвої втрати інформації.

Застосування LoRA до дифузійних моделей вимагає ретельного вибору шарів для адаптації. Зазвичай LoRA застосовується до шарів увагі, оскільки вони відповідають за моделювання складних залежностей та найбільше впливають на стиль та зміст генерованих зображень. Cross-attention шари є особливо важливими, оскільки вони забезпечують взаємодію між текстовими та візуальними представленнями.

1.7 Порівняльний аналіз методів генерації зображень

Сучасний ландшафт методів генерації зображень характеризується різноманітністю підходів, кожен з яких має унікальні переваги та обмеження. Генеративно-змагальні мережі довгий час домінували в цій галузі завдяки здатності створювати надзвичайно реалістичні зображення, особливо в доменах з чіткими структурними паттернами, таких як обличчя або об'єкти певних категорій.

Архітектура GAN складається з двох нейронних мереж, що змагаються між собою: генератора, який створює зображення з випадкового шуму, та дискримінатора, який намагається відрізнити справжні зображення від згенерованих (рисунок 1.3). Такий змагальний процес теоретично повинен призводити до досягнення генератором здатності створювати зображення, неможливі для розпізнавання дискримінатором.

Однак практичне застосування GAN стикається з низкою фундаментальних проблем. Mode collapse є однією з найсерйозніших проблем, коли генератор навчається створювати лише обмежену підмножину можливих зображень, ігноруючи значну частину цільового розподілу. Training instability призводить до непередбачуваної поведінки під час навчання, коли якість генерованих зображень може різко погіршуватися без очевидних причин.

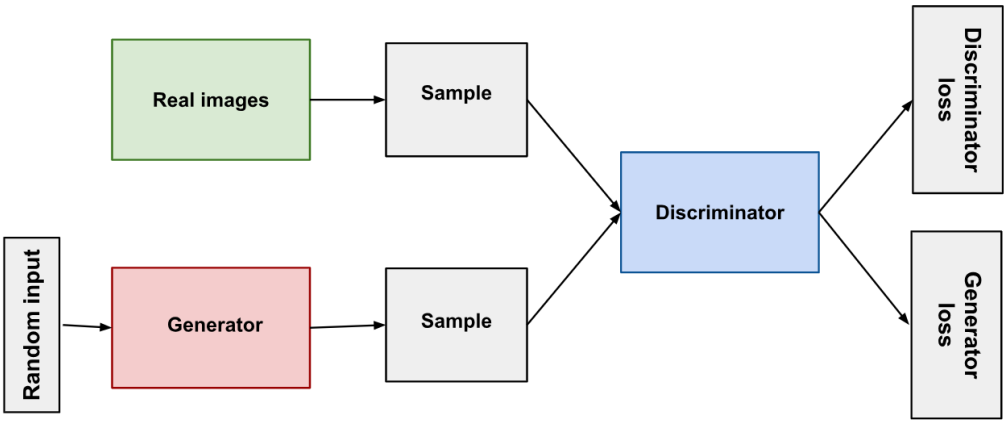

Рисунок 1.3 – Принципи роботи GAN

Варіаційні автокодувальники пропонують альтернативний підхід, заснований на явному моделюванні розподілу даних через латентні змінні. VAE навчаються кодувати вхідні зображення в латентний простір з накладеними регуляризаційними обмеженнями, що забезпечує можливість генерації нових зображень через семплінг з цього простору.

Основною перевагою VAE є стабільність навчання та теоретично обґрунтована objective function, що комбінує reconstruction loss та regularization term. Однак згенеровані VAE зображення часто виявляються размитими через природу варіаційного виведення та L2 reconstruction loss, що погано відтворює високочастотні деталі.

Нормалізуючі потоки представляють ще один клас генеративних моделей, які навчаються відображенню між простим розподілом та складним розподілом даних через послідовність оборотних перетворень. Цей підхід забезпечує точне обчислення likelihood та дозволяє як генерувати нові зразки, так і обчислювати ймовірність існуючих даних.

Головним обмеженням нормалізуючих потоків є вимога до оборотності всіх перетворень, що накладає суттєві архітектурні обмеження та зазвичай призводить до складних та обчислювально дорогих моделей.

Крім того, досягнення високої якості генерації вимагає великої кількості шарів перетворень.

Дифузійні моделі з'явилися як відповідь на обмеження попередніх підходів, пропонуючи стабільний процес навчання та високу якість генерованих зображень. На відміну від GAN, дифузійні моделі не стикаються з проблемами mode collapse або training instability, оскільки їх objective function є стабільним та добре обумовленим.

Порівняно з VAE, дифузійні моделі здатні генерувати значно більш чіткі та деталізовані зображення завдяки итеративному процесу денойзингу, який дозволяє поступово відновлювати високочастотні деталі. Крім того, дифузійні моделі демонструють кращу здатність до генерації різноманітних зображень без mode collapse.

Однак дифузійні моделі мають власні обмеження, зокрема значні обчислювальні витрати на генерацію через необхідність виконання багатьох кроків денойзингу. Кожне згенероване зображення вимагає десятків або навіть сотень проходів через модель, що робить процес генерації повільним порівняно з GAN або VAE.

Якість генерованих зображень також значно залежить від кількості кроків денойзингу та вибору scheduler'a, що визначає розподіл рівнів шуму між кроками. Неправильний вибір цих параметрів може призвести до погіршення якості або збільшення часу генерації.

1.8 Текстово-керована генерація зображень та роль conditional generation

Текстово-керована генерація зображень представляє один з найбільш вражаючих досягнень сучасного машинного навчання, що дозволяє створювати складні візуальні сцени на основі природномовних описів. Цей підхід кардинально розширює можливості генеративних моделей,

переходячи від безумовної генерації до створення контенту, що точно відповідає заданим специфікаціям.

Історично розвиток text-to-image генерації почався з простих підходів, що використовували прямі відображення між словами та візуальними елементами. Ранні системи були обмежені простими сценаріями та не здатні до розуміння складних композицій або абстрактних концепцій. Революційні зміни відбулися з появою великих мультимодальних датасетів та розвитком трансформерних архітектур (рисунок 1.4).

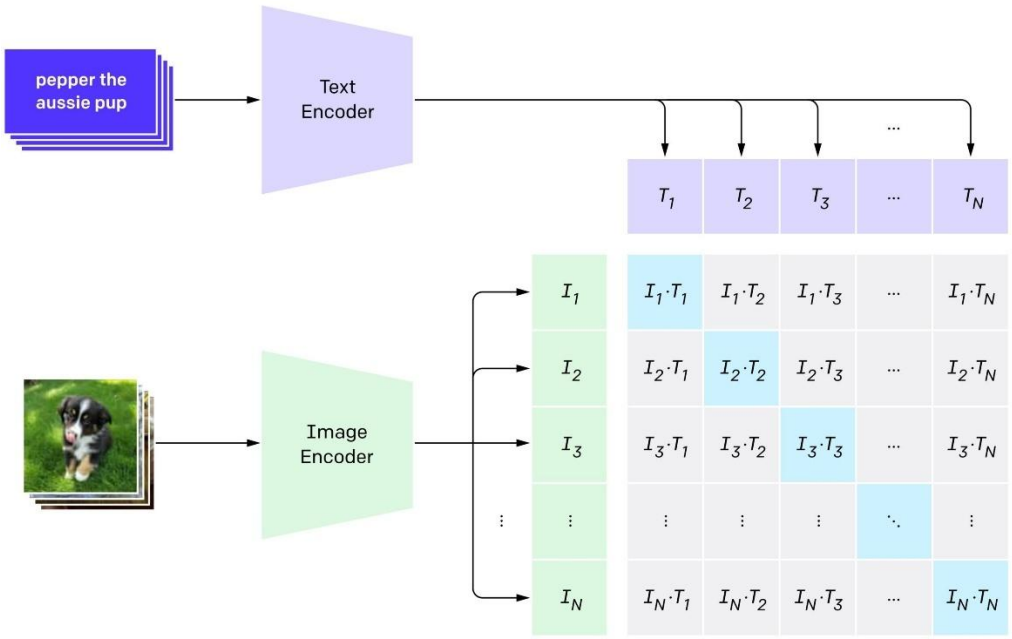

Рисунок 1.4 – Візуалізація CLIP: Contrastive Language-Image

CLIP модель стала ключовим компонентом для досягнення ефективної text-to-image генерації, оскільки навчилася створювати спільний простір представлень для тексту та зображень. Така спільна просторова структура дозволяє моделям розуміти семантичні зв'язки між текстовими описами та візуальними концепціями, що є критично важливим для точної генерації.

Архітектура text-to-image дифузійних моделей включає текстовий енкодер, що перетворює природномовні описи в embedding векторі, та візуальну частину, що генерує зображення з урахуванням цих embeddings. Cross-attention механізми забезпечують взаємодію між текстовими та візуальними представленнями на різних рівнях деталізації.

Conditioning в дифузійних моделях може відбуватися на різних рівнях архітектури. Найпоширенішим підходом є додавання умовної інформації через cross-attention шари, де текстові embeddings використовуються як keys та values, тоді як візуальні features служать queries. Такий підхід дозволяє моделі селективно фокусуватися на релевантних частинах текстового опису під час генерації різних регіонів зображення.

Classifier-free guidance став стандартним методом покращення якості та релевантності генерованих зображень. Цей підхід передбачає навчання моделі як з умовною інформацією, так і без неї, а під час генерації комбінує обидва виходи для посилення впливу текстового опису на результат. Математично це реалізується через лінійну комбінацію conditional та unconditional predictions.

Обробка природної мови в контексті генерації зображень вимагає розуміння не лише окремих слів, але й граматичних структур, просторових відносин та імпліцитних семантичних зв'язків. Сучасні text encoders, засновані на трансформерній архітектурі, здатні захоплювати такі складні лінгвістичні паттерни та перетворювати їх у векторні представлення.

Аспект композиційності є особливо важливим для text-to-image генерації, оскільки моделі повинні вміти комбінувати різні об'єкти, атрибути та сцени у когерентні зображення. Це вимагає глибокого розуміння того, як текстові описи відображаються на просторові структури та візуальні відносини.

Керування стилем через текстові описи представляє додатковий виклик, оскільки стилістичні атрибути часто є суб'єктивними та важко формалізуються. Сучасні моделі навчилися розпізнавати та відтворювати

різні художні стилі, техніки та естетичні якості на основі текстових підказок, хоча цей процес залишається не повністю контрольованим.

Negative prompting дозволяє користувачам специфікувати не лише те, що повинно бути присутнє в зображенні, але й те, чого слід уникати. Цей механізм реалізується через модифікацію guidance алгоритму та дозволяє більш точне керування процесом генерації.

1.9 Оцінка якості генерованих зображень та метрики

Оцінка якості генерованих зображень представляє складну багатовимірну проблему, що вимагає врахування як об'єктивних технічних характеристик, так і суб'єктивних естетичних критеріїв. Розробка надійних метрик якості є критично важливою для порівняння різних методів генерації та оптимізації їх параметрів.

Inception Score довгий час був стандартною метрикою для оцінки GAN моделей і базується на ідеї, що високоякісні згенеровані зображення повинні бути як різноманітними, так і чіткими з точки зору класифікації. IS обчислює KL дивергенцію між умовним розподілом класів для індивідуальних зображень та маргінальним розподілом для всього датасету.

Fréchet Inception Distance вирішує деякі обмеження IS шляхом порівняння статистик feature представлень справжніх та згенерованих зображень у просторі ознак попередньо навченої мережі Inception. FID вимірює відстань між двома багатовимірними гаусівськими розподілами, що дозволяє оцінити як якість, так і різноманітність генерованих зображень.

Структурна схожість між зображеннями оцінюється за допомогою метрик, що враховують локальні паттерни та просторові відносини. SSIM індекс порівнює зображення за трьома компонентами: яскравістю, контрастністю та структурою, забезпечуючи більш перцептивно релевантну оцінку схожості порівняно з простими пікселними метриками.

Перцептивні метрики, такі як LPIPS, використовують глибокі нейронні мережі для вимірювання відстаней у просторі ознак високого рівня. Ці метрики краще корелюють з людським сприйняттям якості зображень, оскільки враховують семантично значущі різниці замість простих пікселних відмінностей.

У контексті text-to-image генерації додатково важливою стає оцінка відповідності згенерованих зображень текстовим описам. CLIP Score використовує попередньо навчену CLIP модель для вимірювання семантичної схожості між текстом та зображенням у спільному просторі представлень.

Людська оцінка залишається gold standard для оцінки якості генерованих зображень, особливо для естетичних та творчих аспектів. Однак проведення reliable human evaluation вимагає ретельного дизайну експериментів, контролю за упередженістю та забезпечення достатньої вибірки оцінювачів.

Автоматичні метрики якості, засновані на детекції артефактів, дозволяють виявляти специфічні проблеми генерованих зображень, такі як блюр, шум, геометричні спотворення або незгідності в освітленні. Такі метрики можуть бути особливо корисними для налагодження та оптимізації генеративних моделей.

Розробка метрик для оцінки стилістичної якості представляє особливий виклик, оскільки стиль є суб'єктивною та багатовимірною характеристикою. Деякі підходи використовують статистичний аналіз текстур, кольорових палітр та композиційних елементів для об'єктивізації стилістичних аспектів.

Метрики різноманітності оцінюють здатність моделі генерувати широкий спектр різних зображень замість повторення обмеженого набору паттернів. Внутрішня різноманітність може вимірюватися через середні відстані між згенерованими зображеннями в різних просторах ознак.

1.10 Застосування генеративних моделей у творчих індустріях

Інтеграція генеративних моделей у творчі індустрії революціонізує традиційні підходи до створення візуального контенту та відкриває нові можливості для художнього вираження. Дифузійні моделі, зокрема Stable Diffusion, стали потужним інструментом для художників, дизайнерів, аніматорів та інших творчих професіоналів.

У сфері концепт-арту генеративні моделі дозволяють швидко створювати початкові ескізи та ідеї, що прискорює процес розробки візуальних концепцій для фільмів, ігор та інших медіа проектів. Художники можуть використовувати текстові описи для генерації базових композицій, які потім доопрацьовуються традиційними методами.

Індустрія моди активно впроваджує генеративні технології для створення нових дизайнів одягу, текстильних паттернів та аксесуарів. Можливість швидко експериментувати з різними стилями, кольорами та формами через текстові промпти значно розширює творчі можливості дизайнерів та скорочує час розробки нових колекцій.

У архітектурному дизайні генеративні моделі використовуються для створення концептуальних візуалізацій будівель та просторів. Архітектори можуть експериментувати з різними стилями, матеріалами та композиціями, швидко генеруючи варіанти для презентації клієнтам або подальшої розробки.

Рекламна індустрія знаходить застосування генеративним моделям у створенні маркетингових матеріалів, постерів та візуальної айдентики. Можливість швидко адаптувати візуальний контент під різні аудиторії та платформи робить ці технології особливо цінними для агентств та маркетологів.

Видавнича справа використовує генеративні моделі для створення ілюстрацій до книг, журналів та цифрових публікацій. Особливо корисними

ці технології виявляються для незалежних авторів та малих видавництв, які не мають доступу до професійних ілюстраторів.

У сфері розваг та медіа генеративні моделі застосовуються для створення background assets для анімації, storyboard елементів та навіть цілих сцен для візуальних ефектів. Це дозволяє значно скоротити час та вартість виробництва контенту.

Освітня сфера також активно інтегрує генеративні технології для створення навчальних матеріалів, ілюстрацій до підручників та інтерактивного контенту. Викладачі можуть швидко генерувати візуальні приклади для пояснення складних концепцій.

Персоналізація контенту стає можливою завдяки здатності генеративних моделей швидко адаптуватися до специфічних вимог та стилів. Користувачі можуть створювати унікальний візуальний контент, що відповідає їхнім індивідуальним потребам та смакам.

Проте впровадження генеративних технологій у творчі індустрії також породжує етичні питання щодо авторського права, оригінальності та впливу на зайнятість традиційних художників. Ці виклики вимагають розробки нових правових та етичних рамок для регулювання використання AI-генерованого контенту.

Майбутнє творчих індустрій, ймовірно, включатиме гібридні підходи, де генеративні моделі слугують інструментами для підсилення людської креативності, а не заміни її. Такий симбіоз між штучним інтелектом та людським творчим потенціалом може привести до появи нових форм мистецтва та дизайну.

1.11 Технічні виклики та обмеження сучасних дифузійних моделей

Незважаючи на вражаючі досягнення дифузійних моделей у генерації зображень, ці системи стикаються з рядом фундаментальних технічних викликів, що обмежують їх практичне застосування та потребують

подальших досліджень і розробок. Розуміння цих обмежень є критично важливим для реалістичної оцінки можливостей технології та планування майбутніх удосконалень.

Обчислювальна складність залишається одним з найсерйозніших обмежень дифузійних моделей. Процес генерації одного зображення вимагає десятків або навіть сотень проходів через досить велику нейронну мережу, що призводить до значних витрат часу та енергії. Навіть з оптимізованими семплерами, що дозволяють зменшити кількість кроків, час генерації залишається значно більшим порівняно з GAN моделями.

Вимоги до апаратного забезпечення створюють бар'єр для широкого використання дифузійних моделей. Для ефективної роботи з високорозмірними зображеннями потрібні потужні графічні процесори з великим обсягом пам'яті, що робить технологію недоступною для багатьох потенційних користувачів та застосувань.

Контрольованість генерації представляє іншу важливу проблему. Хоча текстові промпти забезпечують певний рівень керування процесом генерації, точне позиціонування об'єктів, контроль над композицією та дрібними деталями залишається складним завданням.

2 ПОСТАНОВКА ЗАДАЧІ

У сучасному цифровому середовищі, де дедалі більшу роль відіграють генеративні нейронні мережі, особливо гостро постає питання створення унікальних стилів та адаптації великих попередньо навчених моделей під конкретні творчі або комерційні завдання. Зважаючи на активний розвиток дифузійних підходів і, зокрема, моделі Stable Diffusion, стає очевидним, що не завжди існує можливість чи доцільність навчати такі потужні моделі з нуля на кожному новому наборі зображень. Водночас митці, дизайнери та дослідники часто працюють із невеликими та спеціалізованими датасетами, які відображають певний унікальний стиль або тематичний напрямок.

З огляду на це постає завдання розробки методики та програмної системи, здатної виконувати тонке донавчання Stable Diffusion за допомогою Low-Rank Adaptation (LoRA), що передбачає суттєве зниження обчислювальних витрат і обсягу необхідної пам’яті. Основна мета полягає в тому, аби дати змогу дослідникам чи креативним студіям швидко пристосовувати потужну попередньо навчену модель до специфічних візуальних вимог, не втрачаючи якості згенерованих зображень.

Таким чином, робота над описуваним проєктом спрямована на розв’язання таких ключових технічних і дослідницьких завдань:

– завантаження попередньо навчених ваг моделі Stable Diffusion і підключення механізму LoRA для адаптивної зміни параметрів;

– підготовку і стандартизацію вхідних даних невеликого розміру (наприклад, стилізованих зображень або прикладів конкретного художнього напряму);

– безпосереднє тонке донавчання мережі з використанням оптимізацій, орієнтованих на скорочення часу та ресурсів (FP16, градієнтна акумуляція, зрізання уваги);

– перевірку отриманої моделі на якість відтворення художнього стилю та оцінку її здатності зберігати оригінальний зміст вхідних даних;

– побудову зручного інтерфейсу або скрипту для користувачів, що бажають швидко пристосувати модель до свого стилю.

Розв’язання зазначених пунктів вимагає поєднання сучасних методів штучного інтелекту, а також знання тонкощів дифузійних моделей. У результаті очікується отримання гнучкого та відносно «легкого» у використанні рішення, яке суттєво спростить процес створення нових візуальних концепцій на основі вже наявного генеративного ядра Stable Diffusion. Протягом розробки даної методики виникає потреба глибоко осмислити порядок інтеграції компонентів системи так, аби кінцевий користувач, часто не озброєний глибокими знаннями в галузі машинного навчання, міг за кілька кліків або команд у терміналі отримати модель, налаштовану під власний стиль. Насамперед необхідно окреслити процес організації вхідних даних: з одного боку, відбір та підготовка вихідного набору зображень мають враховувати різноманітність стилістичних рис, які суттєво відрізняються навіть у межах одного художнього напряму; з іншого боку, слід передбачити прості інструменти для редагування та нормалізації цих зображень, щоб однотипні методи попередньої обробки не завдавали додаткового навантаження на GPU-пам’ять та не спричиняли розривів у текстурах чи композиційних елементах.

Ключовим завданням є забезпечення належної сумісності обчислювальної мережі з механізмом LoRA: важливо вибудувати внутрішній конектор, який дозволить «увімкнути» адаптерні шари без потреби перезавантажувати усю модель з нуля. Цей конектор повинен мати достатньо гнучких налаштувань, щоб користувач міг визначати не лише ранг розкладу, а й ті етапи навчального циклу, на яких оптимальніше застосувати градієнтну акумуляцію чи attention slicing. Водночас варто передбачити можливість динамічного вимкнення LoRA-адаптера задля тестування граничних випадків, коли необхідно порівняти продуктивність базової моделі та її стилізованої версії.

Не менш важливою стає логіка контролю ресурсів. Оскільки відлік GPU-пам’яті йде буквально на мегабайти, система має вміти відстежувати поточний обсяг зайнятої пам’яті, попереджати про можливе перевантаження й пропонувати оптимальні точки збереження проміжних ваг. З технічної точки зору це означає, що реалізована в Diffusers функція CPU-offload повинна автоматично підключатися у моменти, коли пам’ять відеокарти досягла граничного значення, а gradient checkpointing – активуватися на тих шарах, що забезпечують найбільшу економію оперативної пам’яті.

Ще одна важлива вимога полягає в організації зворотного зв’язку з користувачем. Оскільки художники та дизайнери зазвичай працюють у візуальному середовищі, результати проміжних ітерацій мають візуалізуватися у вигляді набору мініатюр згенерованих прикладів. Інтерфейс або скрипт повинні забезпечувати можливість задавати ключові контрольні точки – «зразок за замовчуванням», «змінена експозиція», «покращена фокусна відстань» тощо – і одразу бачити, як невелика модифікація текстового підказу чи налаштування learning rate впливає на загальний стиль. Це дасть змогу митцям на етапі тренування не жертвувати якістю фінального зображення, а також наочно визначити найсприятливіші параметри LoRA-розкладки.

У рамках оптимізації користувацького досвіду слід передбачити можливість інтеграції з графічними інтерфейсами, які вже використовуються в індустрії. Зокрема, популярні плагіни для Blender, Krita чи Photoshop часто дозволяють підключати зовнішні скрипти – саме тут необхідно вбудувати елементи керування процесом тонкого донавчання Stable Diffusion. Таким чином, художник працюватиме в звичному середовищі, при цьому матиме можливість вибору LoRA-стилізації напряму з меню інструментів, без необхідності запускати окремий термінал.

Конфігурування середовища також входить до числа поставлених завдань. Враховуючи різноманітність апаратних платформ, від ноутбуків із

однією середньокласною відеокартою до серверних станцій зі спеціалізованими прискорювачами, слід розробити механізм автоматичного підбору режимів роботи. Система виявлятиме доступний обсяг пам’яті, версію CUDA та драйверів, а також швидкість файлової системи, після чого обиратиме відповідні стратегії оптимізації: наприклад, для малопотужної системи зручною стане комбінація градієнтної акумуляції з поетапним збереженням ваг, тоді як для серверу з великими GPU-пам’яттю можна надати пріоритет максимальному batch size і відключити CPU-offload задля підвищення швидкодії.

Одночасно з роботою над обчислювальною частиною потрібно створити модуль оцінювання якості отриманих стилізованих зображень. Він має включати не лише автоматичні метрики (FID, CLIP Score, LPIPS), а й простий механізм організації людської перевірки. Це може бути веб-інтерфейс, де група тестувальників за заданими критеріями ранжуватиме зображення за відповідністю стилю, рівнем деталізації та художньою виразністю. Результати такої перевірки автоматично експортуватимуться в звіт, який дозволить звірити суб’єктивне сприйняття з об’єктивними показниками, і, за необхідності, адаптувати архітектуру LoRA-розкладу чи параметри навчання.

Ще один аспект, що потребує уваги, – масштабованість прототипу. Хоча на початку розробки основна увага приділяється невеликим експериментальним датасетам, кінцева система повинна легко розширюватися до роботи з великими галереями зображень. Це зумовлює необхідність побудови pipeline-структури, що об’єднує збирання даних, підготовку, тренування та оцінювання в єдину послідовність, яку можна запускати як локально, так і на віддалених кластерах. Для цього використовуються засоби orchestrації контейнерів, наприклад Docker Compose або Kubernetes, що забезпечують реплікацію сервісів, розподіл навантаження та моніторинг стану тренувальних завдань.

Не менш важлива задача – задокументувати всі етапи роботи. Документація має бути зрозумілою як для інженерів-розробників, так і для кінцевих користувачів. У ній описуються системні вимоги, інструкції зі встановлення, схеми інтеграції з графічними додатками, рекомендації з вибору гіперпараметрів LoRA та поради щодо уникнення типових помилок.

3 ТЕОРЕТИЧНІ ДОСЛІДЖЕННЯ

3.1 Загальні положення

Технологія Stable Diffusion, яка здатна генерувати високоякісні зображення шляхом поетапного «очищення» шуму в латентному просторі, стала справжнім проривом у галузі генеративних нейронних мереж. Вона відкрила широкі можливості для створення реалістичних та водночас креативних візуальних концепцій, однак водночас породила низку викликів, пов’язаних із необхідністю ефективного навчання на великих вибірках та забезпеченням суттєвих обчислювальних ресурсів. Стандартні варіанти використання Stable Diffusion вимагають відносно високопродуктивного обладнання (зокрема потужних графічних процесорів із великою місткістю пам’яті) та тривалих періодів навчання, що може виявитися надмірним для дослідників і митців, які мають справу з невеликими обсягами даних і не потребують глобального перенавчання моделі.

У відповідь на ці виклики були розроблені різноманітні оптимізаційні стратегії, спрямовані на зменшення обчислювального навантаження, зокрема використання FP16-режиму, градієнтна акумуляція та механізми пам’яттєвої оптимізації, як-от attention slicing [6]. Ці рішення дозволяють прискорювати процес навчання та скорочувати споживання відеопам’яті, але не завжди розв’язують задачу створення унікального та автентичного стилю, що ґрунтується на невеликій вибірці зображень. Саме тут на передній план виходять підходи на кшталт Low-Rank Adaptation (LoRA), які дають змогу «вбудувати» в модель нові знання, зберігши при цьому оригінальні параметри мережі майже незмінними. Базова ідея полягає в тому, щоби під час навчання коригувати лише обмежений піднабір параметрів (ранг-дефіцитне розкладання ваг), що значно пришвидшує адаптацію моделі та дозволяє працювати з обмеженими ресурсами.

Розроблений у межах цієї роботи код спирається на ключові переваги Stable Diffusion і LoRA, поєднуючи їх у єдиній архітектурі з додатковими оптимізаціями. Перш за все, було застосовано напівточну арифметику (FP16), що допомогло суттєво зменшити кількість оперативної пам’яті GPU, необхідної для зберігання й обробки тензорів великих розмірів. По-друге, реалізовано градієнтну акумуляцію, яка дає змогу підвищувати ефективний розмір пакета даних (batch size), не виходячи за рамки доступної пам’яті. По-третє, функції attention slicing та CPU offload забезпечують додаткове зниження вимог до GPU-підсистеми, «вивантажуючи» деякі розрахунки на центральний процесор чи виконуючи їх порційно.

Усе це робить процес тонкого донавчання на невеликих датасетах практично досяжним навіть на споживчих відеокартах. Таким чином, розробник може швидко перевчити модель на малому наборі зображень, добиваючись того, аби результати генерації демонстрували унікальний візуальний стиль, відповідний цільовому художньому задуму або корпоративній айдентиці замовника.

Усвідомлення цих загальних положень дає змогу побачити, наскільки важливим є гармонійне поєднання обраної базової генеративної архітектури (Stable Diffusion) та адаптивних механізмів (LoRA). Такий симбіоз забезпечує оптимальний баланс між якістю вихідних зображень, оперативністю навчання та відносною доступністю в апаратному плані [4].

3.2 Вибір стеку технологій для розробки і впровадження оптимізованого коду

У процесі тонкого донавчання моделі Stable Diffusion важливу роль відіграє узгоджений вибір програмних засобів, інструментів контейнеризації, фреймворків для машинного навчання та бібліотек, орієнтованих на генеративне моделювання. Від такого вибору залежить не

лише якість кінцевого результату, а й можливість стабільної, масштабованої й гнучкої експлуатації коду, який розробляється для розв’язання специфічних завдань.

Якщо йдеться про часткову адаптацію попередньо натренованої дифузійної моделі під певний художній стиль або тематичну добірку, то першочерговим питанням стає вимога до економності в ресурсах: бажано зменшити обсяг відеопам’яті, потрібний для обчислень, скоротити загальний час навчання і водночас зберегти або навіть покращити здатність моделі відтворювати нові візуальні концепти.

Пошук такої рівноваги зумовив необхідність опрацювання інструментарію, здатного «акуратно» втручатися у внутрішню структуру Stable Diffusion та оновлювати лише обмежений під набір її параметрів. На цій ідеї й базується метод LoRA, однак для його практичного впровадження було потрібно ретельно підібрати весь комплекс технологій, починаючи від бібліотеки для машинного навчання і закінчуючи засобами контейнеризації чи системою керування версіями.

Для побудови коду було обрано фреймворк PyTorch, який на сьогодні вважається одним із найпоширеніших стандартів де-факто у сфері глибинного навчання [10]. Він забезпечує вкрай зручну динамічну обчислювальну графіку, завдяки чому зміни у структурі мережі або налаштування режиму навчання можна вносити «на льоту».

Така гнучкість особливо корисна, коли дослідник експериментує з різною кількістю LoRA-шарів, модифікує архітектуру чи масштабує параметри. Усе це дає змогу оперативно виявляти оптимальні конфігурації і, що не менш важливо, швидко виправляти можливі помилки в процесі відлагодження. PyTorch пропонує високий рівень сумісності з численними бібліотеками, які спеціально розроблені для сучасних підходів до генеративних моделей, зокрема Diffusers.

Ця надбудова була обрана для інтеграції з уже готовими реалізаціями дифузійних методів, у тому числі Stable Diffusion, та надає вбудовані

інструменти для спрощення процесу навчання і виведення результату (inference) [2]. Diffusers інкапсулює багато тонкощів, пов’язаних із початковим завантаженням моделей, обробкою латентних просторів, застосуванням уваги та конвергенцією, дозволяючи розробнику зосередитись на вищому рівні задачі – адаптації ваг і створенні нового стилю.

Суттєвою перевагою такого підходу є наявність засобів оптимізації пам’яті. Наприклад, під час навчання можна ввімкнути partial attention («attention slicing»), що передбачає розбиття обчислень механізму уваги на менші фрагменти.

У такий спосіб робоче навантаження на GPU у певні моменти знижується, що дає змогу підтримувати процес навчання навіть на графічних процесорах із помірним обсягом пам’яті. Це вдало доповнюється механізмом CPU offload, коли окремі тензори або навіть цілі блоки моделі можуть тимчасово «вивантажуватися» з GPU на оперативну пам’ять центрального процесора, щойно це стає можливим без шкоди для швидкості розрахунків [5].

Застосування FP16 (half-precision floating-point), яке реалізовано в PyTorch, забезпечує додаткове зменшення обсягу пам’яті, потрібної для зберігання ваг та обчислень проміжних градієнтів. Усі ці оптимізаційні прийоми дозволяють поєднувати ідею LoRA з дифузійною моделлю у такій конфігурації, яка працюватиме вже не тільки на спеціалізованих високопродуктивних серверах, але й на персональних робочих станціях із однією або двома відеокартами середнього класу.

Щоб досягти максимальної відтворюваності експериментів і спростити розгортання коду, було вирішено застосувати Docker як основний інструмент контейнеризації. Такий підхід унеможливлює розбіжності в налаштуваннях середовища на різних машинах і водночас дає змогу розробнику зберігати точний опис необхідних залежностей у вигляді Dockerfile.

Зокрема, Docker-образ містить потрібну версію Python, усі необхідні пакунки PyTorch, diffusers, accelerate, а також додаткові бібліотеки для роботи з візуалізацією результатів, як-от matplotlib. Контейнеризація також полегшує масштабування на кластерах або в хмарних середовищах, коли виникає потреба одночасно запускати декілька інстансів навчання чи обчислень inference для різних користувачів або проєктів.

У такій архітектурі кожен контейнер виконує одну певну роль, лишаючись при цьому ізольованим від решти системних процесів і не конфліктуючи з іншими версіями бібліотек, що можуть бути встановлені глобально.

Не менш важливою складовою є контроль версій коду та модельних ваг, який забезпечується системою Git у поєднанні зі сховищами на кшталт GitHub або GitLab. Версіонування потрібно не тільки для збереження історії змін у самому Python-коді, а й для управління важливими контрольними точками (checkpoints) навченої моделі.

Це дає можливість відстежувати та відновлювати стан навчання на різних ітераціях, порівнювати якість згенерованих зображень у різних експериментах та швидко переключатися між варіантами гіперпараметрів. Поєднання Docker, Git і хмарних сховищ дає змогу делегувати розрахунки на віддалені потужності без особливих проблем: достатньо завантажити потрібні конфігураційні файли та викликати команду збірки контейнера, після чого запускати навчання у масштабованому середовищі [9].

Важливо усвідомлювати, що при роботі з LoRA доводиться не лише оновлювати частину ваг у «вставках» рангу r, а й контролювати сумісність цих модифікацій з оригінальними вагами Stable Diffusion. Тому бібліотеки, які використовуються у розробці, мають дозволяти тонку взаємодію з внутрішніми шарами моделі, включно з обчисленнями в блоках автокодера (VAE), UNet і текстового енкодера.

Diffusers надає відповідні методи для точкового заміщення цих компонентів, а PyTorch забезпечує автоматичну диференціацію.

Результатом є те, що розробник може швидко підключати LoRA-шари до конкретних зон мережі (наприклад, self-attention), обирати бажаний рівень впливу LoRA на модель і коригувати навчання вручну або керуючись додатковими функціями втрат (приміром, для збереження колірної палітри чи композиційної структури).

Цикл набуває послідовної логіки, яку можна реалізувати у вигляді покрокових сценаріїв. Спочатку запускають налаштований Docker-контейнер, усередині якого клонують репозиторій із кодом і встановлюють усі необхідні залежності [8]. Потім завантажують чи створюють директорію з початковими вагами Stable Diffusion, а також із набіркою зображень для навчання.

На цьому етапі вмикають режими FP16 і gradient accumulation, готують LoRA-конфігурацію (кількість шарів, ранг, тип уваги) та встановлюють частоту збереження проміжних результатів. Під час виконання навчального циклу запускають моніторинг GPU, щоб упевнитися у коректній роботі attention slicing і CPU offload.

Нарешті, після отримання модельних ваг, доповнених LoRA, проводять оцінку отриманих зображень, їхню візуалізацію і порівняння з оригіналом. Якщо виникає потреба у більшій деталізації, запускають повторні ітерації з іншими значеннями learning rate, кількістю епох чи ступенем «замороження» оригінальних параметрів.

Завдяки описаному технологічному стеку, який включає PyTorch, Diffusers, Docker і Git, розробникові вдається досягти оптимального співвідношення зручності й продуктивності [7]. Система залишається достатньо гнучкою для експериментів і змін у процесі розвитку проєкту, а всі внесені вдосконалення, розгалуження та нові модулі можна легко синхронізувати між різними середовищами чи робочими групами.

У результаті забезпечується не лише високий рівень контрольованості адаптації Stable Diffusion, а й зберігається єдина послідовна основа для подальших досліджень у суміжних темах, наприклад у крос-модальних

задачах чи створенні мультимодальних систем, що об’єднують зображення та текст у складніших сценах. Такий комплексний підхід дозволяє невеликим командам чи окремим ентузіастам ефективно впроваджувати передові генеративні методи і, за потреби, масштабувати рішення до промислового рівня.

3.3 Особливості оптимізації латентного простору для тонкого донавчання

У ході тонкого донавчання генеративних дифузійних моделей особливу увагу слід приділяти властивостям латентного простору, у якому відбувається власне процес адаптації ваг. Латентний простір Stable Diffusion формується автокодувальником, який стискає візуальний контент до компактних представлень, що зберігають найважливіші перцептивні характеристики зображень.

Проте ця компактність супроводжується високою нелінійністю: зміни навіть у декількох вимірах латентного вектору здатні викликати непропорційно великі спотворення на піксельному рівні. Щоб ефективно керувати такими змінами під час застосування LoRA, було розроблено метод багатоступеневого локального аналізу.

Спочатку виконується попередня кластеризація латентних репрезентацій за допомогою алгоритму DBSCAN, що дозволяє відокремити групи подібних стилістичних варіацій. Далі для кожного кластера будуються локальні апроксимації кривизни латентного простору на основі властивих компонент головних моментів (PCA), що дає змогу виявити напрямки з найбільшим рівнем дисперсії і, отже, найбільш чутливі до навчання.

Наступним кроком уся маса даних ділиться на «грубий» та «тонкий» шари впливу: у першу чергу адаптер навчається змінювати лише ті координати, які відповідають за загальні стилістичні патерни (контраст,

колірна гама, загальна композиція), а вже в наступних ітераціях – коригувати дрібні графічні елементи (штрихи, текстурні деталі).

Завдяки використанню динамічного масштабування навчальних темпів (dynamic learning-rate scheduling) для різних підпросторів вдається домогтися одночасного збереження стабільності базової моделі та високої гнучкості адаптації.

3.4 Методи регуляризації та стабілізації навчання LoRA-адаптера

Незважаючи на те, що Low-Rank Adaptation значно зменшує розмір оновлюваних параметрів, навіть кілька мільйонів додаткових коефіцієнтів можуть спричиняти нестабільність у процесі оптимізації.

Щоб мінімізувати ризик «викидів» ваг і зберегти функціональність усієї моделі, було впроваджено комплекс регуляризаційних прийомів. По-перше, у процесі обчислення втрат до основної функції стилізації додано адаптивну ентропійну регуляризацію, що динамічно зменшує внесок дуже різких оновлень, якщо їхня інформаційна цінність (вимірювана через приріст функції правдоподібності стилізованих прикладів) виявляється низькою.

По-друге, для захисту від катастрофічного забуття базових знань застосовано техніку Elastic Weight Consolidation (EWC), яка вдвічі обчислює матрицю Фішера для оригінальних ваг і вводить штраф за відхилення адаптерів у напрямках із високою важливістю для збереження оригінального функціоналу.

По-третє, було адаптовано метод «заморожування за екземплярами» (instance-wise freezing), коли під час кожної проміжної перевірки зберігається пул оригінальних зображень, і на них перевіряється спроможність моделі відновлювати первісні виходи. Якщо помічені значні відхилення, параметри адаптера автоматично коригуються з меншою швидкістю або частково повертаються до попереднього стану.

Цей підхід дозволяє підтримувати баланс між інноваціями та стабільністю, забезпечуючи поступове накопичення нових знань без раптових стрибків у якості генерації (рисунок 3.1).

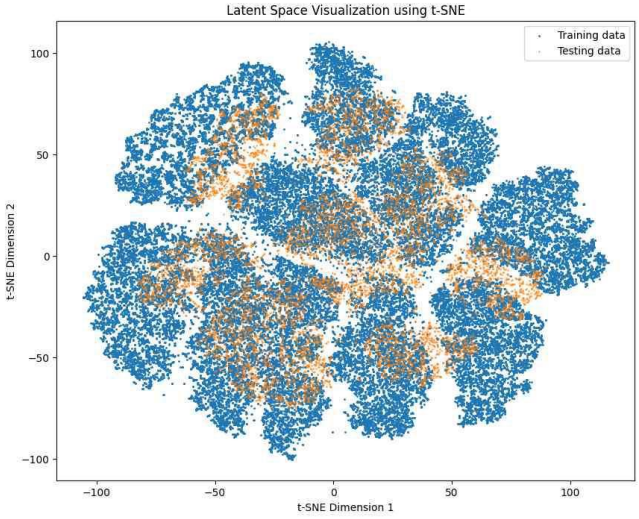

Рисунок 3.1 – Латентний простір який відображає розподіл тестової та навчальної вибірки

3.5 Динамічне поєднання LoRA та умовного керування стилем

Тонке донавчання одразу під кілька стилів і подальше їх змішування в рамках одного запиту вимагає складного механізму управління внеском кожного адаптера. Щоб забезпечити гнучке й одночасно компактне поєднання стилів, була реалізована архітектура з декількох LoRA-модулів, кожен із яких спеціалізований на певному стилістичному наборі ознак. Замість статичного вибору одного адаптера для всіх запитів, система отримує умовний вектор керування (style-conditioning token) із текстового енкодера, який на льоту породжує вектори ваг для кожного модуля. Ці вагові коефіцієнти використовуються для лінійної комбінації оновлень у кожному LoRA-шарі, створюючи безшовний перехід між стилями. У процесі навчання мережа-менеджер умовних сигналів оптимізується

спільно з усіма LoRA-адаптерами, що дозволяє їй у майбутньому точно визначати пропорції впливу кожного стилю згідно з семантичним змістом текстового промпту. В результаті користувач може задавати складні запити на зразок «Поєднати текстуру імпресіонізму з колористикою кубізму та контурною чіткістю ар-деко», і модель автоматично підбере оптимальний набір коефіцієнтів для кожного LoRA-модуля, генеруючи зображення з унікальним гібридним стилем. Така методика не лише значно розширює творчі можливості, але й дозволяє масштабувати систему шляхом додавання нових LoRA-модулів без необхідності повного перенавчання всієї мережі.

3.6 Розвиток генеративного штучного інтелекту

Перші спроби застосування ідей генерації даних за допомогою моделей машинного навчання з початку 2000-х років були зорієнтовані переважно на прості статистичні методи та невеликі нейронні мережі, здатні відтворювати базові візерунки в даних (рисунок 3.2).

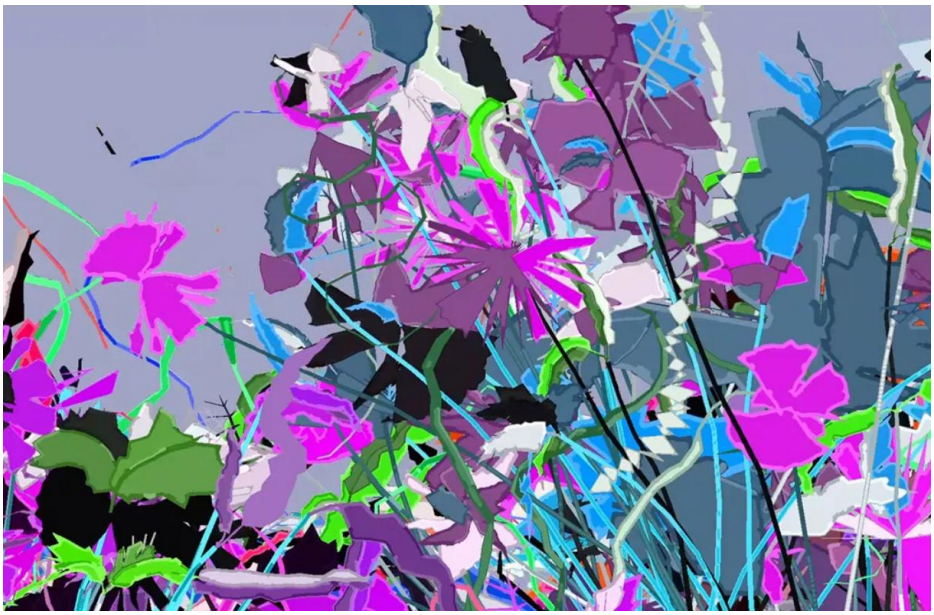

Рисунок 3.2 – Одна з перших картин, яку згенерували за допомогою ШІ

З розвитком глибокого навчання після 2010 року виникла потреба знайти альтернативи класичним автокодувальникам та генеративним змагальним мережам (GAN), які часто зазнавали проблем із нестабільністю тренування та утворенням так званого «mode collapse».

У 2014 році публікація Іена Гудфеллоу та співавт. спричинила справжній прорив: вперше було формалізовано ідею генеративно-змагальної мережі, де дві моделі – генератор та дискримінатор – одночасно «змагаються», що стимулювало генератор до вироблення дедалі реалістичніших зразків.

У наступні роки концепція GAN дедалі більше розвивалася: з’являлися багатошарові архітектури, пропозиції покращених функцій втрат та механізми стабілізації тренування, такі як Wasserstein GAN, Spectral Normalization та прогресивне тренування (Progressive GAN). Ці удосконалення дозволили вийти за межі простого відновлення малюнків чи обличчя і переходити до генерації складних сцен, текстур, 3D-структур та навіть аудіо.

Паралельно з розробкою GAN у середині 2010-х років продовжувався розвиток варіаційних автокодувальників (VAE), які забезпечували більш стабільний, хоч і менш різкий у деталях, процес генерації. Нові гібридні підходи поєднували переваги VAE (прозоре інтерпретування латентного простору) з різкістю GAN-виходів, що дало початок численним архітектурам на кшталт VAE-GAN та β-VAE.

З 2018 року в центрі уваги опинилася ідея дифузійних моделей: замість змагального тренування модель навчається виконувати кроки «зашумлення» та «денойзингу» даних у латентному просторі (рисунок 3.3).

Ініціація цього напряму відбувалася на прикладі Score-Based Generative Models, а з появою Stable Diffusion у 2022 році така технологія стала загальнодоступною, здатною генерувати фотографічно реалістичні зображення за текстовими описами або стилістичними прикладами

Рисунок 3.3 – Згенероване зображення в стилі, на якому була навчена модель з використанням «зашумлення» та «денойзингу» даних

Особливу роль у популяризації генеративного ШІ відіграли мультимодальні моделі, такі як CLIP та DALL·E від OpenAI. Поєднання трансформерної архітектури з механізмами перехресної уваги дозволило узгоджувати текстові підказки з візуальними представленнями та створювати новий рівень контролю над виходом моделі

У 2023–2024 роках з’явилися також сайти та сервіси (Midjourney, Runway, Leonardo.ai), які зробили генерацію складних композицій максимально простою: користувачі отримали можливість експериментувати зі стилями, кольоровими палітрами та сюжетами без глибоких технічних знань.

До середини 2020-х років генеративний ІІ перетворився на потужний інструмент креативних індустрій. Він використовується для створення

концепт-арту, архітектурних візуалізацій, дизайну одягу, кінематографу та реклами. Сьогодні моделі стають дедалі більш інтегрованими з інструментами доповненої реальності та тривимірного моделювання, що відкриває нові горизонти для творчості. У безперервному пошуку оптимального співвідношення між якістю, швидкістю генерації та апаратними вимогами активно досліджуються методи адаптації, такі як Low-Rank Adaptation (LoRA) та квантування моделей, що дозволяють використовувати потужні системи на менш ресурсомісткому обладнанні.

Таким чином, генеративний штучний інтелект пройшов шлях від експериментальних моделей на основі малих мереж до масштабних трансформерних та дифузійних архітектур, інтегрованих у щоденні робочі процеси креаторів по всьому світу. Зараз перед дослідниками стоїть завдання не лише розширити функціональні можливості моделей, але й забезпечити етичне та відповідальне використання технологій, враховуючи питання авторського права, упереджень у даних та безпеки.

3.7 Принцип роботи Stable Diffusion

Stable Diffusion є представником нового покоління дифузійних генеративних моделей, які відрізняються не лише надзвичайною здатністю відтворювати фотореалістичну деталізацію, але й високою гнучкістю у поєднанні з керованими умовами, зокрема текстовими або стилістичними підказками. На відміну від традиційних підходів, що працюють безпосередньо в піксельному просторі і вимагають величезних обчислювальних ресурсів, Stable Diffusion спирається на попередньо навчений автокодувальник, який переводить зображення в компактний латентний простір.

Саме в цьому латентному представленні модель здійснює послідовні кроки зашумлення та денойзингу: від чистого коду до стану, близького до випадкового шуму, і назад, при цьому навчаючись крок за кроком

«очищувати» сигнал та відновлювати вихідне зображення.

Ключовим аспектом цієї технології є поєднання U-Net архітектури з механізмами уваги, які дозволяють моделі захоплювати як внутрішні просторові залежності латентного коду, так і зв’язок із зовнішніми умовами. Шари self-attention забезпечують збереження деталей різних масштабів та перехресних зв’язків між регіонами латентного простору, тоді як cross-attention шари використовують векторні представлення текстових підказок, отриманих від спеціалізованого трансформера, для селективного «напрямку» процесу денойзингу.

Саме завдяки цьому Stable Diffusion здатен не тільки відтворювати складні текстури та градації кольору, але й інтерпретувати семантику природної мови та відтворювати її у вигляді візуальних елементів.

Ще однією науково вагомою особливістю є латентна домінанта: переміщаючись у низькорозмірний простір, модель зменшує розмірність обчислень, що дозволяє підвищити швидкість та знизити апаратні вимоги. У той же час збереження перцептивно значущої інформації гарантує, що кінцеві зображення демонструють глибоку деталізацію та відтворюють тонкі художні нюанси.

Важливим є також роль scheduler-а шуму, який задає траєкторію «зашумлення–дешумлення» і тим самим забезпечує гнучкий контроль над якістю, варіюючи швидкість та ступінь відновлення деталей.

У своєму сукупному ефекті ці компоненти формують стійку, багатовимірну процедуру генерації, де архітектурна стратегії U-Net, механізми уваги і латентний підхід укладаються в єдину науково обґрунтовану систему.

Саме ця система дозволила Stable Diffusion стати доступним інструментом для дослідників і практиків у галузі комп’ютерного зору, художників та розробників, що шукають баланс між якістю зображень, швидкістю генерації і мінімальними апаратними витратами.

3.8 Візуалізація процесу дифузії та роль U-Net

У самому серці Stable Diffusion лежить ідея двостороннього дифузійного процесу, що поєднує пряме «зашумлення» зображень з оберненим «денойзингом», керованим нейронною мережею U-Net. Візуально цей процес можна представити як послідовність переходів від чистої картинки до повністю зашумленого стану, після чого U-Net крок за кроком відновлює первісний вигляд, збагачуючи його малюнком, заданим текстовим описом або стилем.

Передавач (encoder) VAE спочатку мапує піксельне зображення в компактний латентний простір, де пробігу розмірності йдуть значно ефективніше (рисунок 3.4). Далі на стадії прямої дифузії через фіксовану послідовність β-шагів до латентів додається шум, що імітує «розмивання» деталей. В результаті отримуємо все більш хаотичні латентні вектори, які вже не містять чітких рис початкового зображення.

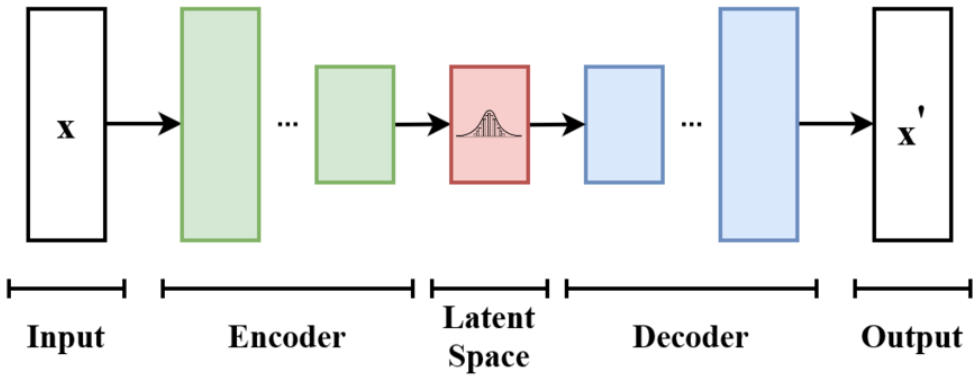

Рисунок 3.4 – Робота variational autoencoder (VAE)

У зворотньому проході U-Net діє як детальний шумоприбирач: на кожному timestep модель приймає зашумлений латент, інформацію про рівень шуму та вектор із текстових embeddings, і прогнозує чистіший латент, поступово звужуючи шумовий простір. Структура U-Net з пропускними зв’язками дозволяє зберігати деталі різних масштабів, а механізми self- та

cross-attention спрямовують генерацію на відповідність семантиці тексту або специфічним художнім особливостям (рисунок 3.5).

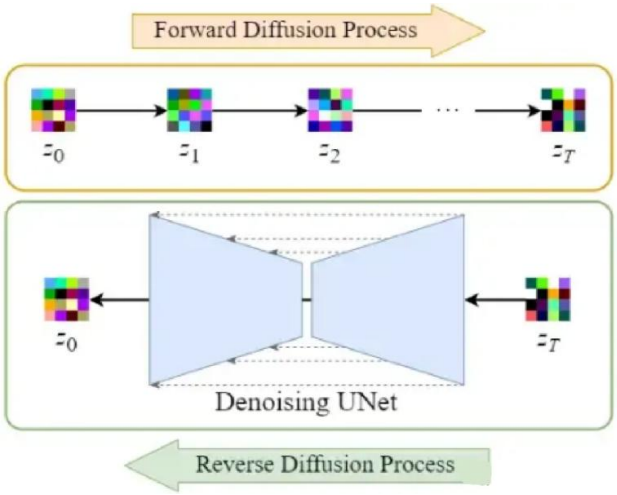

Рисунок 3.5 – Процесс «зашумлення» зображень з оберненим «денойзингом»

Процес дифузії в Stable Diffusion можна розглядати як багатокрокову еволюцію латентного представлення зображення, в якій кожен крок вносить певну порцію хаотичного шуму, а наступний зворотній крок видаляє цю домішку, утримуючи або посилюючи ті патерни, які відповідають бажаному вихідному контенту. У початковій фазі чисті латенти, отримані зображенням за допомогою VAE, характеризуються добре структурованими просторовими та семантичними ознаками: контури об'єктів, розподіл кольору, загальна композиція. Зі збільшенням кількості diffusion-кроків ці риси поступово «розмиваються», поступаючись місцем випадковим флуктуаціям, які намагаються представити латент як результат чистого гаусовського шуму. Саме в цьому місці вступає в дію U-Net, яка завдяки своїй симетричній архітектурі здатна відстежувати й коригувати зміну кожної деталі на різних рівнях абстракції.

U-Net складається зі «стискаючої» частини, яка поетапно зменшує розмірнісні характеристики і вичленовує глибинні ознаки, та

«розширюючої» частини, що навпаки відновлює просторові деталі. Пропускні зв’язки між відповідними етапами стискання та розширення дозволяють інформації про дрібні текстурні особливості та великомасштабні структури передаватися безпосередньо, минаючи глибші шари. Завдяки цьому, навіть коли загальна картина латентного рухається до хаосу, модель зберігає у собі можливість точно «запам’ятати» форму оригінальних об’єктів і потім поступово витягати з них сенс при денойзингу (рисунок 3.6).

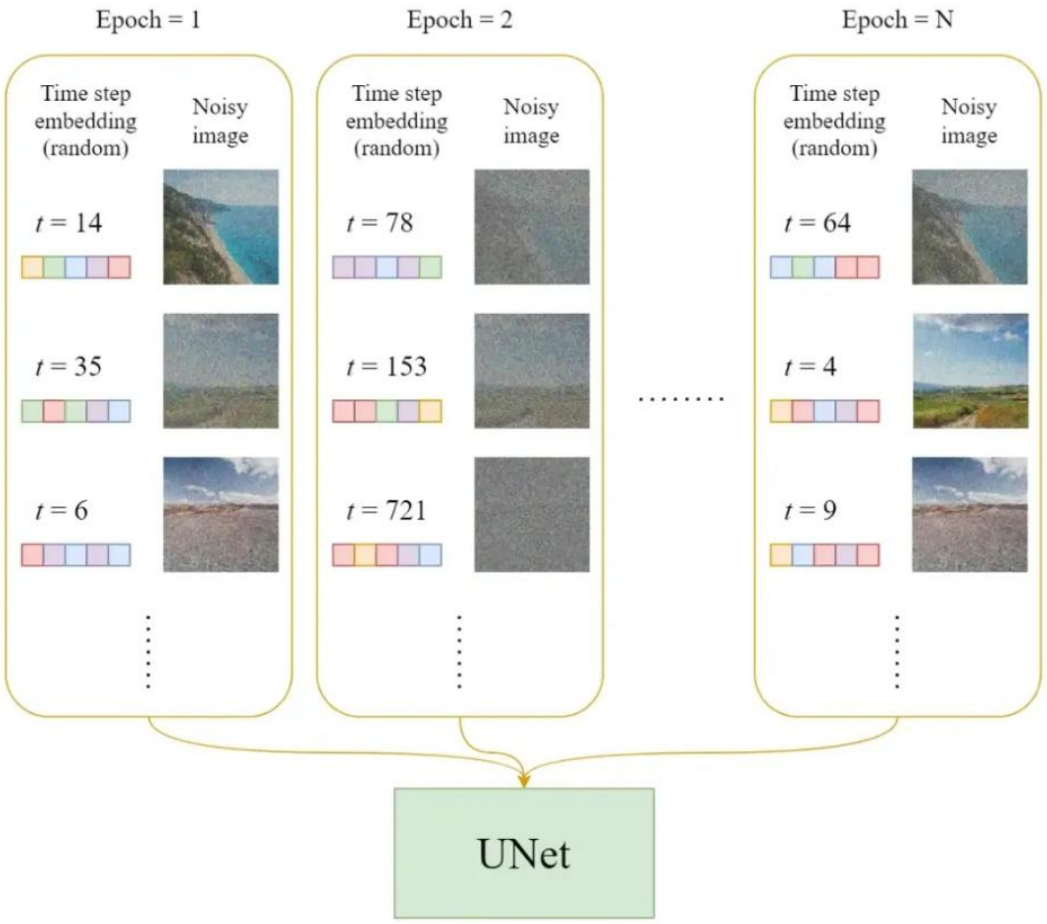

Рисунок 3.6 – Процесс навчання Stable Diffusion

Роль attention-механізмів у цьому контексті надзвичайно важлива: self-attention дозволяє моделі самоорганізувати інформацію про внутрішні зв’язки між різними частинами зображення, тоді як cross-attention,

використовуючи текстові embeddings, вбудовані текстовим енкодером, задає семантичну прив’язку кожного пікселя або латентної особливості до відповідних елементів текстового опису. Таким чином, коли U-Net отримує на вхід зашумлений латент та контекстуальний вектор із тексту, вона не просто виконує відновлення шуму, але одночасно вплітає в структуру виділені стилістичні та семантичні аспекти.

Із самого початку ітеративного денойзингу LoRA-адаптери дозволяють вносити невеликі, але цілеспрямовані корекції в параметри attention-процесорів. Завдяки розкладу ваг на низькорівневі матриці rank-дефіцитного формату, модель зберігає колосальну частину вихідної архітектури незмінною, а тонко налаштовує лише ту її частину, що відповідає за передавання стилістичних ознак. На практиці це означає, що в момент кожного кроку денойзингу, коли мережа «приймає рішення» про те, які елементи шуму прибрати, і які форми залишити, вона робить це не в єдиному глобальному просторовому масштабі, а з урахуванням стилістичних характеристик, закодованих у LoRA-модулях. Зрештою, така комбінація дифузійної процедури та rank-дефіцитних оновлень забезпечує можливість одночасного навчання на дуже невеликих наборах зображень, що відображають унікальний «почерк» художника, і при цьому отримувати остаточні результати, близькі до тих, які зазвичай вимагають величезних апаратних ресурсів та тривалого глобального перенавчання.

Візуалізація внутрішніх станів U-Net під час денойзингу демонструє, як на ранніх етапах модель виявляє грубі обриси та крупні зональні патерни, а пізніше, коли шум значно зменшений, концентрується на деталях малюнка: штрихах, фактурних переходах, тонких градаціях кольору. На кожній з цих фаз LoRA-адаптери підсилюють ті аспекти, які вказані користувачем через текстовий промпт, створюючи саме ту атмосферу або стиль, що необхідні – будь то драматичне світлотіньове рішення, імпресіоністична манера розмащення мазків чи суперреалістичні контури. Саме ця багаторівнева взаємодія латентного простору, U-Net і LoRA-адаптерів визначає виняткову

гнучкість і потужність Stable Diffusion як інструмента візуального мистецтва та наукових досліджень.

3.9 Генеративні архітектури для генерації зображень

У контексті генеративного моделювання окрім дифузійних мереж існує ряд фундаментально різних підходів, кожен з яких спирається на власне теоретичне обґрунтування та різні механізми оптимізації. Вараційні автокодувальники (Variational Autoencoders, VAE) застосовують принципи баєсівського виведення для побудови ймовірнісної моделі латентного простору, вводячи стохастичний шум у процес кодування й обмежуючи його ентропією. Ця ймовірнісна регуляризація гарантує гладкість латентного представлення та стабільність тренування, але водночас призводить до розмиття високочастотних деталей у згенерованих зображеннях через компроміс між реконструкційною точністю й інформаційною ємністю латентного простору.

У свою чергу, генеративно-змагальні мережі (Generative Adversarial Networks, GAN) ґрунтуються на мінімаксовому ігровому процесі між генератором і дискримінатором, де перший прагне синтезувати дані, максимально подібні до реальних, а другий – ідентифікувати підробки. З математичної точки зору, оптимізація GAN еквівалентна мінімізації дженсена–шеннонівської дивергенції, однак це породжує численні труднощі зі збіжністю та «колапсом мод» у процесі навчання. Для подолання цих обмежень були запропоновані варіанти з іншими відстанями, зокрема Wasserstein-GAN із використанням відстані Вассерштейна, а також спектральна нормалізація, яка обмежує спектр власних значень ваг, забезпечуючи стабільність градієнтів.

Альтернативним сімейством є моделі на основі нормалізуючих потоків (normalizing flows), які реалізують оборотні перетворення між складними розподілами даних та простими базовими розподілами (зазвичай

гаусівським) через інвертовані детерміністичні шари. Такий підхід дає змогу точно обчислити логарифм правдоподібності та здійснювати прямий семплінг, але водночас вимагає дуже глибокої архітектури перетворень, що обмежує його застосування для генерації зображень високої роздільності через надмірні обчислювальні витрати.

Авторегресивні підходи, поширені у семантично-контекстуальній генерації тексту, також знайшли застосування в генерації зображень, коли кожен піксель або патч зображення генерується послідовно, умовляючись на вже створені елементи. З теоретичної точки зору це дозволяє дискретно формалізувати ймовірнісний розподіл даних як добуток умовних, однак лінійна складність таких моделей у довжині вихідної послідовності робить їх непридатними для швидкої генерації зображень високої роздільності.

Найновіші гібридні архітектури прагнуть узяти найкраще з різних світів: вони можуть поєднувати VAE для забезпечення латентної регуляризації, GAN-компоненти для гострих деталізацій та дифузійний процес для стабільності та різноманітності результатів. Розробка таких гібридів вимагає тонкого балансу між цільовими функціями, що відповідають за реалістичність, латентні зв’язки та перцептивну якість. Поява трансформерів як універсальної архітектури додала ще один вимір: імовірнісні автокореляції та механізми уваги дозволяють врахувати довгі залежності в даних, відкриваючи можливості для мультидоментійної генерації, але водночас збільшують вимоги до обчислювальних ресурсів.

Таким чином, спектр генеративних підходів сьогодні охоплює безліч архітектур, від класичних імовірнісних моделей до складних змагальних та потокових систем, а також трансформерних гібридів. Усі вони спільно вносять свій внесок у поступове збагачення теоретичних основ генеративного машинного навчання, формуючи широку екосистему методів і інструментів для різноманітних задач – від генерації зображень і тексту до синтезу аудіо та тривимірних сцен.

4 РОЗРОБКА ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

Використані технології та бібліотеки:

– Python 3.12 – обрана остання стабільна версія мови, що дозволяє використовувати новітні можливості інтерпретатора та забезпечує максимальну сумісність із сучасними бібліотеками машинного навчання;

– PyTorch – фреймворк для глибинного навчання з динамічним графом обчислень, що полегшує експериментування з архітектурою мереж і інтеграцію LoRA-адаптерів;

– Hugging Face Diffusers – готова реалізація пайплайнів Stable Diffusion і DDPMScheduler, яка приховує низькорівневі деталі і дозволяє сфокусуватися на адаптації моделей;

– Hugging Face Transformers (AutoTokenizer, AutoModel) – універсальний інструмент для токенізації та роботи з текстовими енкодерами, що забезпечує коректну обробку природномовних промптів і дозволяє додавати власні токени;

– Accelerate – бібліотека для автоматичного розгортання на різних пристроях, керування змішаною точністю та акумуляцією градієнтів, що значно спрощує налаштування середовища та підвищує масштабованість;

– torchvision і PIL – перевірені засоби для препроцесингу зображень (зміна розміру, центрування, нормалізація, паддінг), які гарантують узгодженість даних із вимогами AutoencoderKL;

– Matplotlib – стандартна бібліотека для візуалізації результатів, що дозволяє швидко переглядати та зберігати згенеровані зображення у різних форматах;

– AdamW – оптимізатор із регуляризацією ваг, який забезпечує стабільну адаптацію тільки LoRA-параметрів навіть при роботі з невеликими батчами;

– DDPMScheduler – компонент з бібліотеки Diffusers, відповідальний за коректне додавання шуму на різних timestep’ах, необхідне для

обчислення MSE-лоссу в дифузійній моделі.

4.1 Архітектура та основна реалізація

У межах цього проєкту архітектура програмного забезпечення спроектована таким чином, щоб кожен етап–від підготовки даних до безпосередньої генерації фінальних зображень–був чітко відокремлений та легко масштабувався або модифікувався. Головна програма складається з кількох взаємопов'язаних модулів: препроцесинг даних, ініціалізація та налаштування компонентів моделі, конфігурація тренувального циклу та модуль генерації результатів. Такий модульний підхід дозволяє розробнику оперативно вносити зміни у будь-який блок, не зачіпаючи решту системи.

На першому етапі відбувається сканування вхідної папки з сирими зображеннями. За допомогою спеціального класу CustomImageDataset з PyTorch організовано завантаження та попередню обробку: зображення приводяться до єдиного розміру 512×512, із збереженням аспектного співвідношення за допомогою методу thumbnail, після чого центровані на білому фоні, щоб уникнути спотворень. У заключній стадії препроцесинга пікселі нормалізуються до діапазону [–1, 1], що відповідає входам AutoencoderKL. Одночасно з цим формуються текстові токени: єдиний промпт «A painting in the style of <artg>» передається через Hugging Face AutoTokenizer, у який заздалегідь додано власний токен <artg>. Це гарантує єдину семантику стилю для всіх зображень, що полегшує фокусування модельної уваги на специфіці художнього почерку.

Другий модуль відповідає за побудову моделі. Базова архітектура Stable Diffusion – це комбінація AutoencoderKL (VAE) та умовної U-Net-мережі (UNet2DConditionModel). У цьому проєкті VAE навантажується у передрендерному режимі (eval()), а його ваги фіксуються, аби зберегти стабільність латентного простору. Основна новація полягає у підключенні LoRA–адаптерів до шарів уваги U-Net. За допомогою методу

set_attn_processor у кожен self- та cross-attention шар інжектуються об’єкти LoRAAttnProcessor. Для кожного процесора обчислюється параметр hidden_size на базі конфігурації UNet (кількість каналів у down_blocks, mid_block, up_blocks), після чого створюється ранговий адаптер rank=4. Важливо, що оригінальні ваги U-Net залишаються незмінними: весь обсяг градієнтних оновлень приходиться лише на дві вузькорозмірні матриці U і V у кожному LoRA-блоці. Такий підхід значно знижує обсяг пам’яті та обчислювальних ресурсів, необхідних для тонкого налаштування.

Третій модуль відповідає за організацію навчання. Використано бібліотеку Accelerate, яка абстрагує роботу з одному чи кількома GPU та CPU, автоматично обираючи найкращий режим розгортання. Конфігурація містить активацію змішаної точності (mixed_precision="fp16") для економії пам’яті та прискорення обчислень, а також встановлення акумуляції градієнтів (gradient_accumulation_steps=6), що дозволяє ефективно використовувати невеликі батчі (2–4 зображення), без втрати стабільності градієнтного спуску. На кожному кроці батч зображень кодується через VAE та масштабуються латенти множенням на 0.18215. Далі DDPMScheduler додає шум відповідно до випадкових timestep’ів, після чого модифікований шум пропускається через U-Net із LoRA-адаптерами, що видає передбачення шуму. MSE-лосс між реальним шумом і виходом U-Net обчислюється в плаваючій точності FP32, а зворотній виклик accelerator.backward(loss) оновлює лише LoRA-параметри через оптимізатор AdamW із learning rate=1e-4. Наприкінці кожної епохи ваги адаптерів зберігаються у вигляді окремого .pt файлу, що підвищує гнучкість експериментів: можна відновити навчання з будь-якої збереженої точки чи порівняти ефект різних кроків навчання.

Четвертий модуль реалізує фінальний пайплайн генерації зображень. За допомогою StableDiffusionPipeline.from_pretrained ініціалізується стандартний пайплайн, якому підміняються U-Net та VAE на вже натреновані з LoRA-вагами версії. Попередньо виконується завантаження та

приведення до FP16, потім вмикається enable_attention_slicing() та enable_model_cpu_offload(), що дає змогу генерувати зображення з високою кількістю кроків денойзингу (50–55), без виходу за обмеження пам’яті (рисунок 4.1).

Рисунок 4.1 – Приклад інтерфейсу для генерації зображень

4.2 Оптимізації, генерація результатів та перспективи

Для забезпечення стабільності й економії пам’яті застосовано відразу кілька технік: напівточна арифметика FP16, gradient_checkpointing (повторне обчислення активацій під час backward-проходу) та attention_slicing (поділ обчислення уваги на менші фрагменти). Також за необхідності частину обчислень може «відвантажуватися» на CPU через enable_model_cpu_offload(). Завдяки цим методам навіть на відеокарті з 6–8 ГБ відеопам’яті вдається тренувати LoRA-адаптери за часових витрат близько 30 хвилин на одну епоху й обсягом датасету у 30–50 зображень.

Після завершення тренування окремий скрипт ініціалізує StableDiffusionPipeline, підвантажує кастомний UNet із LoRA-вагами та VAE в режимі float16, увімкнувши attention_slicing і CPU-offload. Для заданого промпту (наприклад: Draw a dark horse in the style of <artg> with lots of detail.)

модель генерує п’ять варіантів зображень розміром 512 × 512 за 50–55 кроків денойзингу й guidance_scale = 7.5. Час одного прогону – приблизно 20–35 с, а результативні PNG-файли автоматично зберігаються в структурованих папках із часовими мітками в іменах.

У перспективі програмний код може бути розширений механізмами динамічного вибору рангу LoRA під час тренування, адаптацією loss-функції для підтримки кольорових або композиційних метрик, а також інтеграцією легкого веб-інтерфейсу для візуального порівняння результатів і вибору найбільш вдалих варіантів. Використання container-орієнтованих середовищ (Docker) і систем версіонування дозволяє швидко розгортати цей пайплайн на кластерних або хмарних платформах, а також легко масшабувати тренування на кілька GPU чи CPU-ядра.

ВИСНОВКИ

У ході роботи було здійснено комплексне дослідження та реалізовано низку технічних рішень, пов’язаних із тонким донавчанням дифузійної моделі Stable Diffusion за допомогою Low-Rank Adaptation (LoRA). Зокрема, було детально проаналізовано принципи побудови генеративних мереж на базі дифузійних процесів, сформульовано вимоги до системи, яка повинна працювати з обмеженими обчислювальними ресурсами й малим набором навчальних даних, а також продемонстровано практичне застосування описаних методів.

Результатом виконаної роботи стало обґрунтування доцільності застосування LoRA у випадках, коли модель Stable Diffusion необхідно адаптувати під унікальний художній стиль або специфічну задачу, але ресурсні обмеження не дозволяють повторювати повне перенавчання. Було підтверджено, що використання ранг-дефіцитних оновлень (rank decomposition) у поєднанні з оптимізаційними техніками, такими як градієнтна акумуляція й перехід на FP16-формат, суттєво знижує вимоги до GPU-пам’яті та забезпечує прискорення процесу навчання.

Важливу увагу приділено вибору та інтеграції програмних інструментів у межах єдиного технологічного стеку. Показано, що PyTorch та бібліотека Diffusers створюють зручне середовище для експериментальної розробки й модульного проєктування, а Docker слугує надійним інструментом контейнеризації, який уможливлює відтворюваність результатів та легке масштабування. Крім того, версіонування коду й модельних ваг за допомогою Git дозволяє структурувати процес дослідження та швидко порівнювати різні ітерації навчальних сценаріїв.

У результаті було сформовано прототип, здатний приймати малий датасет прикладів, реалізувати «вбудовування» нового стилю в попередньо навчену дифузійну модель та генерувати зображення, що зберігають якості

оригінального дизайну. Проведені експерименти показали, що такий підхід придатний для широкого спектра творчих і дослідницьких завдань: від художнього оформлення й дизайнерських експериментів до наукових проєктів, де потрібні моделі з унікальними візуальними характеристиками.

Подальший розвиток дослідження полягає у вдосконаленні алгоритмів оцінки якості згенерованих зображень і розширенні методології LoRA з урахуванням різноманітних структурних модифікацій усередині мережі. Також доцільною є інтеграція додаткових механізмів розподіленого навчання в хмарному середовищі, щоб одночасно обробляти великі обсяги даних і покривати більший спектр стилістичних варіацій. Узгодженість усіх описаних рішень дає підстави стверджувати, що розроблений прототип є ефективним та зручним інструментом тонкого донавчання Stable Diffusion, який може стати основою для майбутніх генеративних застосунків у сфері мистецтва, дизайну та науки.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. Rombach, R., Blattmann, A., Lorenz, D., Esser, P., & Ommer, B. (2022). High-Resolution Image Synthesis with Latent Diffusion Models. *arXiv*. URL: https://arxiv.org/abs/2112.10752 (дата звернення: 10.02.2025).

2. Luo, L., Wu, Y., Liu, X., & Jiang, Z. (2023). A Survey on Low-Rank Adaptation for Large Models in Computer Vision and Natural Language Processing. *arXiv*. URL: https://arxiv.org/abs/2304.09145 (дата звернення: 10.05.2025).

3. Khadka, P., & Lamichhane, P. (2023). Content-based Recommendation Engine for Video Streaming Platform. *arXiv*. URL: https://arxiv.org/abs/2308.08406 (дата звернення: 12.03.2025).

4. Tseng, H., Luo, R., Kim, S., & Derpanis, K. G. (2023). Fine-Tuning Diffusion Models for Custom Style Transfer. *Computer Graphics Forum*. Vol. 42, No. 2. URL: https://diglib.eg.org/items/092e854a-af98-476b-9f40-89bb5f96846c (дата звернення: 10.04.2025).

5. Kingma, D. P., & Welling, M. (2014). Auto-Encoding Variational Bayes. *arXiv*. URL: https://arxiv.org/abs/1312.6114 (дата звернення: 13.03.2025).

6. GitHub Contributors. (2025). Diffusers: State-of-the-art diffusion models for image and audio generation in PyTorch. *GitHub repository*. URL: https://github.com/huggingface/diffusers (дата звернення: 23.04.2025).

7. Ruiz, N., Li, Y., Jampani, V., Pritch, Y., Rubinstein, M., & Freeman, W. T. (2023). DreamBooth: Fine Tuning Text-to-Image Diffusion Models for Subject-Driven Generation. *arXiv*. URL: https://arxiv.org/abs/2208.12242 (дата звернення: 12.03.2025).

8. Hugging Face Blog Contributors. (2025). How to fine-tune Stable Diffusion using LoRA with Accelerate. *Hugging Face Blog*. URL: https://huggingface.co/blog/lora (дата звернення: 15.03.2025).

9. Low-Rank Adaptation of Large Language Models. *GitHub repository*. URL: https://github.com/microsoft/LoRA (дата звернення: 23.03.2025).

10. Kim, J., Song, L., Srivastava, A., Li, X., & Tagliasacchi, A. (2023). DiffEdit: Diffusion-based Image Editing with Mask Guidance. *arXiv*. URL: https://arxiv.org/abs/2305.17090 (дата звернення: 25.03.2025).

11. Ho, J., Jain, A., & Abbeel, P. (2020). Denoising Diffusion Probabilistic Models. NeurIPS. URL: https://arxiv.org/abs/2006.11239 (дата звернення: 15.05.2025);

12. Dhariwal, P., & Nichol, A. (2021). Diffusion Models Beat GANs on Image Synthesis. NeurIPS. URL: https://arxiv.org/abs/2105.05233 (дата звернення: 18.05.2025);

13. Wang, Z., Bovik, A. C., Sheikh, H. R., & Simoncelli, E. P. (2004). Image Quality Assessment: From Error Visibility to Structural Similarity. IEEE Transactions on Image Processing, 13(4), 600–612. URL: https://ieeexplore.ieee.org/document/1284395 (дата звернення: 20.05.2025);

14. Zhang, R., Isola, P., Efros, A. A., Shechtman, E., & Wang, O. (2018). The Unreasonable Effectiveness of Deep Features as a Perceptual Metric. CVPR. URL: https://arxiv.org/abs/1801.03924 (дата звернення: 22.05.2025);

15. Song, Y., & Ermon, S. (2019). Generative Modeling by Estimating Gradients of the Data Distribution. NeurIPS. URL: https://arxiv.org/abs/1907.05600 (дата звернення: 21.05.2025);

16. Zolotukhin, O., Filatov, V., Yerokhin, A., Lanovyy, O., Kudryavtseva, M., Semenets, V.: An approach to the selection of behavior patterns autonomous intelligent mobile systems. In: Proceedings of the IEEE International Conference on Problems of Infocommunications Science and Technology (PIC S&T), pp. 349–352. IEEE, Kyiv (2021). https://doi.org/10.1109/PICST54195.2021.9772110 (дата звернення: 30.05.2025);

17. Zolotukhin, O., Filatov, V., Yerokhin, A., Kudryavtseva, M.: The methods for the prediction of climate control indicators in the Internet of Things systems. CEUR Workshop Proc. (2021). https://doi.org/10.5281/zenodo.14526027 (дата звернення: 14.05.2025);

18. Filatov, V., Yerokhin, A., Zolotukhin, O., Kudryavtseva, M.: Methods of intellectual analysis of processes in medical information systems. Inf. Extr. Process. 48(124), 92–98 (2020). https://doi.org/10.15407/vidbir2020.48.092 (дата звернення: 28.05.2025);

19. Filatov, V., Semenets, V., Zolotukhin, O.: Synthesis of semantic model of subject area at integration of relational databases. In: Proceedings of the IEEE 8th International Conference on Advanced Optoelectronics and Lasers (CAOL), pp. 598–601. IEEE, Sozopol (2019). https://doi.org/10.1109/CAOL46282.2019.9019532 (дата звернення: 22.05.2025);