МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Західноукраїнський національний університет
Факультет комп'ютерних інформаційних технологій
Кафедра інформаційно-обчислювальних систем і управління

ЛИСЮК Роман Олександрович

Метод генерації коміксів за допомогою генеративних змагальних мереж / A Method for Generating Comics Using Generative Adversarial Networks

спеціальність: 122 - Комп'ютерні науки
освітньо-професійна програма - Комп'ютерні науки

Кваліфікаційна робота

Виконав студент групи КНм-21
Р.О. Лисюк
_____

Науковий керівник:
к.т.н. Д.І. Загородня
_____

Кваліфікаційну роботу
допущено до захисту:
«___» _____ 20___ р.
Завідувач кафедри
_____ М.П. Комар

ТЕРНОПІЛЬ - 2023

Факультет комп'ютерних інформаційних технологій
Кафедра інформаційно-обчислювальних систем і управління
Освітній ступінь «магістр»
спеціальність: 122 – Комп'ютерні науки
освітньо-професійна програма – Комп'ютерні науки

ЗАТВЕРДЖУЮ
Завідувач кафедри
_____ М.П. Комар
«____» _____20__р.

ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ РОБОТУ СТУДЕНТУ
Лисюк Роман Олександрович
(прізвище, ім'я, по батькові)

1. Тема кваліфікаційної роботи
Метод генерації коміксів за допомогою генеративних змагальних мереж / A Method for Generating Comics Using Generative Adversarial Networks
керівник роботи к.т.н., Д.І. Загородня
затверджені наказом по університету від 8 грудня 2022 року № 491.

2. Строк подання студентом закінченої кваліфікаційної роботи 1 грудня 2023 р.

3. Вихідні дані до кваліфікаційної роботи: завдання на кваліфікаційну роботу студента, наукові статті, технічна література.

4. Основні питання, які потрібно розробити
- огляд предметної області;
- аналіз існуючих технік GAN;
- вивчення різних архітектур GAN;
- підбір або розробка набору даних для навчання;
- експерименти з генерацією коміксів;
- навчання та тестування GAN на підготовленому датасеті з метою генерації коміксових зображень і сцен;
- оцінка якості генерованих зображень;
- аналіз та оцінка реалістичності;
- дослідження можливостей застосування.

5. Перелік графічного матеріалу у роботі
- схема архітектури мережі GAN;
- приклади згенерованх зображень коміксів

6. Консультанти розділів кваліфікаційної роботи

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		Завдання видав	Завдання прийняв

7. Дата видачі завдання 8 грудня 2022 р.

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів кваліфікаційної роботи	Строк виконання етапів кваліфікаційної роботи	Примітка
1	Огляд предметної області і постановка задачі	12.2022 р. – 03.2023 р.	
2	Методи і підходи для генерації коміксів	03.2023 р. – 05.2023 р.	
3	Реалізація та аналіз результатів	05.2023 р. – 11.2023 р.	
4	Повне завершення та представлення кваліфікаційної роботи на кафедру	01.12.2023 р.	

Студент _____ Р.О. Лисюк
підпис

Керівник роботи _____ к.т.н. Д.І. Загородня
підпис

РЕЗЮМЕ

Кваліфікаційна робота на тему «Метод генерації коміксів за допомогою генеративних змагальних мереж» на здобуття освітнього ступеня «Магістр» зі спеціальності 122 «Комп’ютерні науки» освітньої програми «Комп’ютерні науки» написана обсягом в 88 сторінок і містить 36 ілюстрацій, 2 додатки та 60 використаних джерел.

Метою даної кваліфікаційної роботи є розробка та аналіз методу генерації коміксів з використанням генеративних змагальних мереж для створення високоякісних та унікальних візуальних коміксів.

Методи досліджень: теорія генеративних змагальних мереж, обробка зображень, комп'ютерне зору, машинне навчання, нейронні мережі.

Результати дослідження: розроблено та впроваджено модель на основі StyleGAN2-ADA, яка демонструє високу якість генерованих коміксів з точним відтворенням стилістичних особливостей та можливість генерації нових унікальних візуальних рішень.

Результати роботи можуть бути використані у видавничій сфері, графічному дизайні, розвагах та медіа, де потрібне створення візуального контенту в стилі коміксів.

Ключові слова: ГЕНЕРАТИВНІ ЗМАГАЛЬНІ МЕРЕЖІ, СТИЛЕГАН, КОМІКСИ, ГЕНЕРАЦІЯ ЗОБРАЖЕНЬ, НЕЙРОННІ МЕРЕЖІ.

ABSTRACT

Qualification work on the topic "Method of Generating Comics Using Generative Adversarial Networks" for the Master's degree in specialty 122 "Computer Science" of the educational program "Computer Science" is written on 88 pages and contains 36 illustrations, 2 annexes, and 60 sources.

The purpose of this qualification work is to develop and analyze a method for generating comics using generative adversarial networks to create high-quality and unique visual comics.

Research methods: theory of generative adversarial networks, image processing, computer vision, machine learning, neural networks.

Research results: developed and implemented a model based on StyleGAN2-ADA, which demonstrates high quality of generated comics with accurate reproduction of stylistic features and the ability to generate new unique visual solutions.

The results can be used in the publishing field, graphic design, entertainment, and media where the creation of visual content in the style of comics is required.

Keywords: GENERATIVE ADVERSARIAL NETWORKS, STYLEGAN, COMICS, IMAGE GENERATION, NEURAL NETWORKS.

ЗМІСТ

ВСТУП

Актуальність теми. Генеративні змагальні мережі (GAN) у останні роки зробили революційний прорив у галузі обробки зображень, відкривши нові можливості для творчості та інновацій. Серед численних застосувань GAN, генерація коміксів є особливо перспективною областю, яка поєднує арт-технології та розважальну індустрію. Використання GAN для створення коміксів дозволяє не лише автоматизувати та прискорити процес творчості, але й відкриває двері для генерації нових, унікальних візуальних стилів та сюжетів.

Тема GAN у контексті генерації коміксів також важлива з точки зору розвитку штучного інтелекту та машинного навчання. Комікси, як унікальна форма візуального мистецтва, вимагають особливого підходу до зображення персонажів, сцен і емоцій. Розвиток моделей GAN, здатних розпізнавати та відтворювати такі складні аспекти, є значним кроком у напрямку підвищення інтелектуальних здібностей штучного інтелекту. Такі дослідження сприяють не лише технічному прогресу, але й розумінню того, як машини можуть інтерпретувати та візуалізувати людську креативність.

Крім того, GAN відіграють важливу роль у розробці нових методів інтерактивного оповідання та ігрового дизайну. Можливість швидко генерувати різноманітні візуальні елементи та сцени може радикально змінити спосіб створення цифрових ігор та інтерактивних додатків. Це не тільки відкриває нові горизонти для розробників та дизайнерів, але й створює нові можливості для залучення та занурення користувачів у віртуальні світи.

Вивчення GAN у контексті створення коміксів також має велике значення для освітнього сектору. Інструменти, засновані на GAN, можуть бути використані для навчальних цілей, наприклад, для створення наочних посібників та навчальних матеріалів. Вони можуть сприяти більш ефективному та захоплюючому процесу навчання, особливо в областях, де важливе візуальне представлення інформації.

Нарешті, розвиток GAN у генерації коміксів також має потенціал для впливу на соціальні та культурні аспекти суспільства. Вони можуть стати мостом

між технологіями та мистецтвом, збагачуючи культурне життя та сприяючи розумінню та прийняттю штучного інтелекту в масовій культурі. Отже, дослідження в цій області має важливе значення не лише з технічної точки зору, але й як засіб розширення меж людської креативності та взаємодії з технологіями.

Метою даної магістерської роботи є розробка та аналіз методу генерації коміксів за допомогою генеративних змагальних мереж. Це включає розробку моделі, здатної автоматично генерувати візуально привабливі та стилістично відповідні зображення коміксів, а також оцінку ефективності та реалістичності отриманих результатів.

Об'єктом дослідження є процес генерації зображень коміксів за допомогою генеративних змагальних мереж. Це включає вивчення та аналіз алгоритмів генерації, а також оцінку їх здатності до створення візуально привабливих та реалістичних зображень коміксів.

Предметом дослідження є алгоритми та методики, що використовуються в генеративних змагальних мережах для створення зображень коміксів. Це включає аналіз структури та принципів роботи GAN, а також дослідження різних архітектурних рішень, які можуть бути застосовані для досягнення найкращих результатів у генерації коміксів.

Методи дослідження. У дослідженні використовуються такі методи, як експериментальне моделювання, аналіз даних та комп'ютерне бачення. Основна увага приділяється розробці та тренуванню GAN моделей, аналізу їх вихідних даних, та порівнянню ефективності різних архітектур. Також використовуються методи оцінки якості згенерованих зображень, зокрема, за допомогою метрики Frechet Inception Distance (FID).

Наукова новизна одержаних результатів. Це дослідження вносить вклад у розвиток сучасних технологій у галузі комп'ютерного зору та штучного інтелекту, зокрема у створенні нових методів для генерації візуального контенту, що має велике значення як для наукових, так і для комерційних застосувань.

Практичне значення одержаних результатів має значне практичне значення у різних сферах. Нові можливості для галузі розваг, зокрема у створенні

коміксів та ілюстрацій. Використання GAN для автоматизації процесу створення мистецтва може суттєво знизити час та витрати на виробництво, а також сприяти створенню унікальних та інноваційних візуальних стилів.

Апробація результатів дослідження. Основні теоретичні положення роботи й практичні результати дослідження доповідалися й обговорювалися:

«Огляд архітектури мережі StyleGAN2-ADA для генерації коміксів» прийняті до публікації в збірнику за матеріалами V Міжнародна студентська конференція «Наука сьогодення: від досліджень до стратегічних рішень» (22.12.2023, м. Чернівці, Україна).

«Огляд наборів даних для тренування StyleGAN2-ADA для генерації коміксів» прийняті до публікації в збірнику за матеріалами V Міжнародна студентська конференція «Наука сьогодення: від досліджень до стратегічних рішень» (22.12.2023, м. Чернівці, Україна).

Кваліфікаційна робота складається із вступу, трьох розділів, висновків, списку використаних джерел та додатків.

1. ОГЛЯД ПРЕДМЕТНОЇ ОБЛАСТІ І ПОСТАНОВКА ЗАДАЧІ

1.1. Вступ до генеративних змагальних мереж (GAN)

Генеративні змагальні мережі (GAN) були вперше запропоновані Яном Гудфеллоу та його командою в 2014 році. Ця концепція відразу привернула увагу наукового співтовариства через свою новаторськість та потенціал у генерації реалістичних зображень. Відтоді GAN зазнали значного розвитку, знаходячи застосування у різноманітних сферах, від створення мистецтва до поліпшення якості медичних зображень.

У перші роки, GAN часто страждали від так званого "mode collapse", коли генератор вчився виробляти лише обмежену кількість варіацій зображень, але з часом були розроблені нові архітектури та методи навчання, які значно покращили їхню якість та різноманітність.

Давайте спочатку розглянемо, що таке генеративні моделі і чим вони відрізняються від дискримінантних моделей. Скажімо, у вас є вхідні дані x і відповідні вихідні мітки y. Дискримінантна модель намагається безпосередньо вивчити умовний розподіл ймовірностей $P(y|x)$. З іншого боку, генеративна модель намагається вивчити спільний розподіл ймовірностей $P(x,y)$. Він може бути перетворений на $P(y|x)$ за допомогою правила Байєса. Однак, на відміну від дискримінаційних моделей, генеративні моделі можуть використовувати спільний розподіл $P(x,y)$ для генерації ймовірних (x,y) вибірок.

Можна ставити собі питання що такого особливого в тому, щоб просто генерувати більше даних, тим більше, що їх вже є так багато. Але насправді це може мати кілька застосувань. Наприклад, можна подати текст, написаний певним почерком, на вхід генеративної моделі, щоб отримати більше тексту тим самим почерком. Генеративні моделі, зокрема GAN, також можуть бути використані для дослідження в навчанні з підкріпленням, де вони можуть бути використані для створення штучних середовищ. Серед інших застосувань - перетворення ескізів на зображення, згладжування зображень, перетворення зображень низької роздільної здатності на зображення високої роздільної здатності, створення творів мистецтва, перетворення супутникових знімків на

мапи тощо. Окрім широкого спектру застосувань, генеративні моделі особливо корисні, коли відсутня більшість міток, завдяки їхній здатності до напівкерованого навчання.

Тепер, коли ми з'ясували, що таке генеративне моделювання і чому воно корисне, давайте розглянемо різні підходи до генеративного моделювання. З метою порівняння моделей, всі вони можуть бути описані так, щоб забезпечити максимальну правдоподібність.

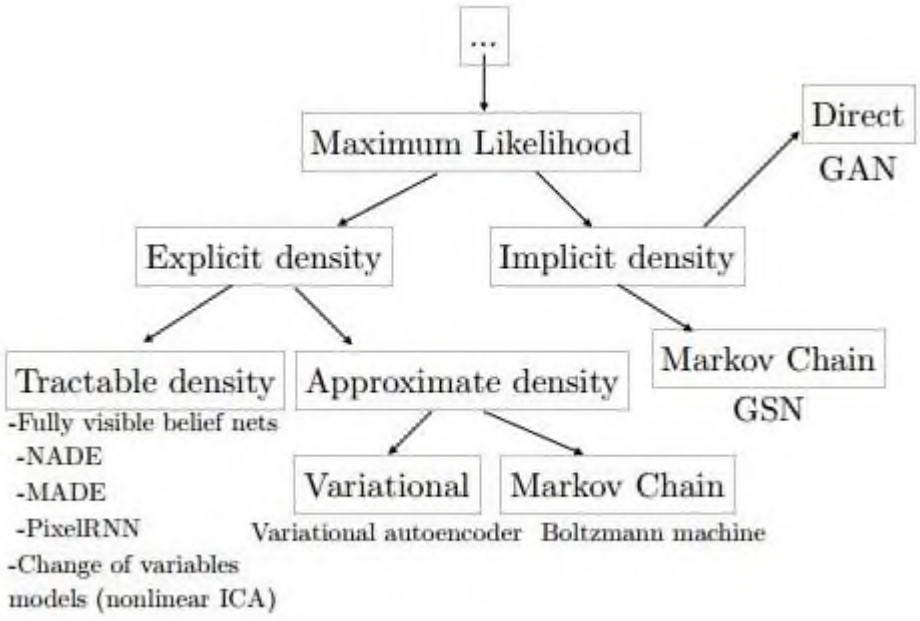

Рисунок 1.1. - Таксономія генеративних моделей [1]

На наведеному вище рисунку 1.1. зображено різні сімейства генеративних моделей, описані Яном Гудфеллоу в його навчальному посібнику з NIPS. Тепер давайте розглянемо переваги та недоліки деяких популярних підходів у сімействах моделей, згаданих вище.

Повністю видимі мережі переконань (FVBN). Вони використовують правило ланцюгових ймовірностей для розкладання розподілу ймовірності розподілу ймовірності по вектору на добуток по кожному з членів вектора.

$$P_{model}(x) = \prod_{i=1}^{n} p_{model}\left(x_i | x_i, \dots, x_{i-1}\right) \qquad 1.1$$

Найпопулярнішою моделлю в цьому сімействі є PixelCNN.

Рисунок 1.2. - Зображення, згенеровані PixelCNN [2]

Найбільшим недоліком FVBN є те, що швидкість генерації зразків дуже повільна. Щоразу, коли ви хочете згенерувати новий зразок, вам доведеться запускати модель знову. Це неможливо зробити паралельно.

Моделі, засновані на зміні змінних (не-лінійні ICA). Такі моделі починаються з простого розподілу, наприклад, гаусівського, і використовують нелінійну функцію для перетворення розподілу в інший простір. Основним недоліком такого підходу є те, що перетворення має бути інверсійним, а латентні змінні повинні мати ту саму розмірність, що й дані. Тобто, якщо ви хочете згенерувати 5000 пікселів, вам потрібно мати 5000 латентних змінних.

Варіаційний автокодер (VAE). Принцип роботи варіаційних автокодерів полягає у виділенні випадкової величини z з функції щільності $log\ p(x)$. Оскільки це важко вирішити, використовується варіаційна апроксимація. Модель прагне максимізувати нижню межу логарифмічної ймовірності даних.

Рисунок 1.3. Зображення обличь, схожих на знаменитостей, згенеровані за допомогою VAE[3]

Основним недоліком є те, що модель є асимптотично узгодженою, якщо розподіл q є досконалим. В іншому випадку буде розрив між нижньою межею та фактичною щільністю даних. Іншим недоліком є те, що згенеровані вибірки мають відносно низьку якість.

Машина Больцмана може бути визначена функцією енергії, а ймовірність певного стану пропорційна кожному значенню енергії. Щоб перетворити це на фактичний розподіл ймовірностей, виконується перенормування шляхом ділення суми на різні стани. Ця сума є нерозв'язною, що вимагає апроксимації за допомогою методів Монте-Карло. Недоліком є те, що ці методи, особливо методи ланцюгового Монте-Карло Маркова, погано працюють у просторах високої розмірності. Тому, хоча вони можуть добре працювати на таких зображеннях, як MNIST, ви не отримаєте подібної продуктивності на зображеннях з ImageNet.

GAN були розроблені для подолання багатьох недоліків, зазначених у вищезгаданих моделях. На відміну від повністю видимих мереж переконань, GAN використовують прихований код і можуть генерувати зразки паралельно. На відміну від варіаційних автокодерів, GAN є асимптотично узгодженими. Крім

того, GAN не потребують ланцюгів Маркова, що є перевагою над машинами Больцмана. Нарешті, GAN часто вважають, що вони генерують найкращі зразки, хоча це дуже суб'єктивна оцінка, яка наразі є предметом дискусій, оскільки з ними конкурують такі моделі, як PixelCNN.

Рисунок 1.4. - Зображення, згенеровані GAN з використанням бази даних обличь Торонто [4]

Тепер давайте розберемо як працює GAN. GAN складаються з двох основних компонентів: генератора та дискримінатора. Генератор відповідає за створення зображень, які намагаються імітувати реальні дані. Дискримінатор, у свою чергу, намагається відрізнити справжні зображення від тих, що створені генератором.

Основна ідея полягає у тому, що генератор та дискримінатор змагаються один з одним: генератор намагається створювати все більш переконливі зображення, а дискримінатор — навчатися краще розрізняти справжні зображення від штучних. Цей процес часто описують як "гру" між генератором та дискримінатором, де кожен намагається перехитрити іншого.

Процес навчання GAN є ітеративним. Спочатку генератор створює зображення, яке подається дискримінатору разом з реальними зображеннями.

Дискримінатор оцінює кожне зображення, намагаючись визначити, чи є воно реальним чи синтетичним. Інформація про правильність або неправильність оцінки передається назад до генератора, який використовує її для покращення своєї здатності створювати реалістичні зображення.

З часом, якщо навчання просувається успішно, генератор стає здатним створювати все більш переконливі зображення, а дискримінатор — краще відрізняти справжні зображення від синтетичних. Це веде до створення високоякісних, реалістичних зображень, які можуть бути важко відрізнити від справжніх.

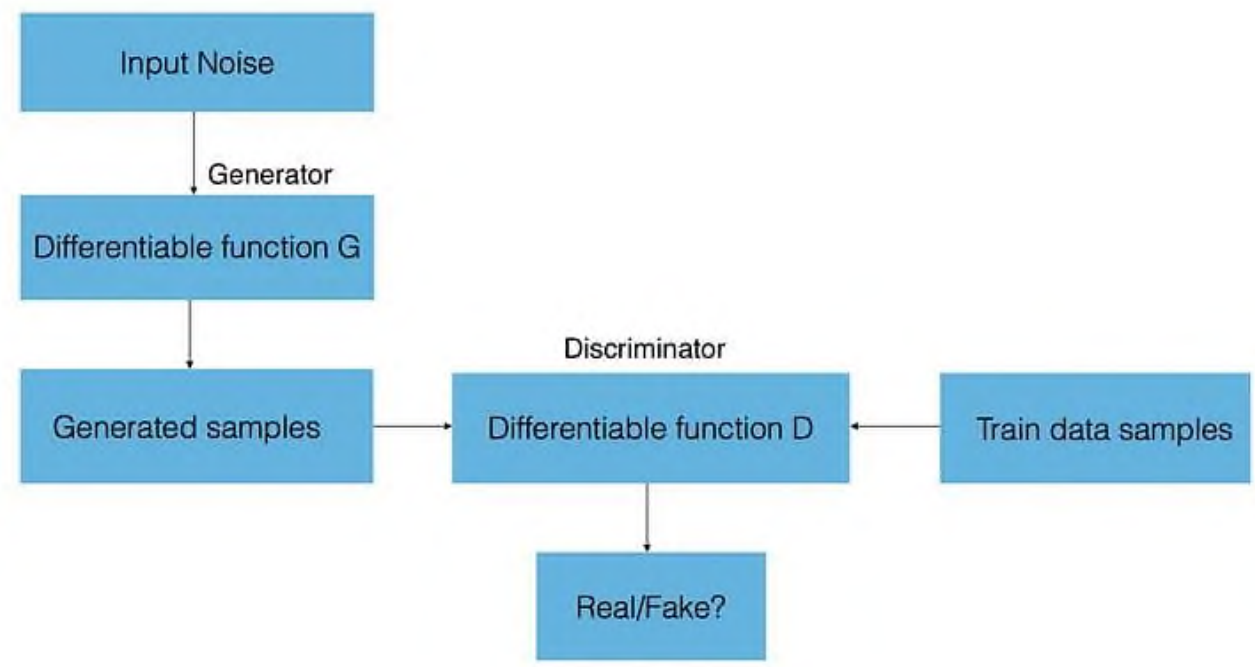

Рисунок 1.5. – Огляд структури мережі GAN. [3]

Розглянемо це більше з математичної сторони процес створення GAN. Генератор є диференційованою функцією G, яка має параметри, що можуть бути визначені за допомогою градієнтного спуску. Вхідні дані для G отримуються шляхом вибірки латентного вектора z з деякого попереднього розподілу над латентними змінними. Отже, по суті, z - це вектор неструктурованого шуму. G застосовується до z, щоб отримати вибірку x з моделі, яка в ідеалі має бути подібною до фактичних даних з навчальної вибірки. Як і генератор,

дискримінатор також є диференційованою функцією *D*, яка має параметри, що можуть бути визначені за допомогою градієнтного спуску. Функція *D*, застосована до вибірки *x*, отриманої з *G(z),* в ідеалі повинна виводити значення, близьке до нуля, що вказує на те, що вибірка є фальшивою. Коли на вхід *D* подається справжня вибірка з даних, вона повинна виводити значення, близьке до одиниці.

Нехай $\boldsymbol{\theta}(D)$ та $\boldsymbol{\theta}(G)$ є параметрами *D* та *G* відповідно. Дискримінатор хоче мінімізувати свою вартість $J(D)(\boldsymbol{\theta}(D),\boldsymbol{\theta}(G))$, але не має контролю над $\boldsymbol{\theta}(G)$, тоді як генератор хоче мінімізувати $J(G)(\boldsymbol{\theta}(D),\boldsymbol{\theta}(G))$ без контролю над $\boldsymbol{\theta}(D)$. Отже, ми хочемо знайти значення рівноваги Неша для $(\boldsymbol{\theta}(D),\boldsymbol{\theta}(G))$ таким чином, щоб $J(D)$ було мінімальним відносно $\boldsymbol{\theta}(D)$ і $J(G)$ було мінімальним відносно $\boldsymbol{\theta}(G)$.

Далі буде показано розбір процедури навчання GAN. Процедура навчання полягає у наступному вибрати алгоритм оптимізації, наприклад, Адам, і застосувати його одночасно до двох міні-батчів даних, один з яких є власне навчальними даними, а інший - зразками, згенерованими *G*. Крім того, ви можете оновлювати дані одного гравця частіше, ніж іншого гравця.

$$J^{(D)} = -\frac{1}{2}E_{x\sim p_{data}} logD(x) - \frac{1}{2}E_z \log\Big(1 - D\big(G(z)\big)\Big) \qquad 1.2$$

$$J^{(G)} = -J^{(D)} \qquad 1.3$$

Перший доданок в $J(D)$ представляє фактичні дані, які подаються на дискримінатор, і дискримінатор хоче максимізувати логарифмічну ймовірність передбачення одиниці, що вказує на те, що дані є справжніми. Другий доданок представляє вибірки, згенеровані *G*. Тут дискримінатор хоче максимізувати лог ймовірність передбачення нуля, що вказує на те, що дані є фальшивими. Генератор, з іншого боку, намагається мінімізувати логарифмічну ймовірність того, що дискримінатор правильний. Рішенням цієї проблеми є точка рівноваги гри, яка є сідловою точкою втрати дискримінатора.

Основна проблема цієї мінімаксної гри полягає в тому, що коли дискримінатор стає все більш розумним, градієнт для генератора зникає. Один із

способів виправити це - змінити порядок аргументів у функції перехресної ентропії замість того, щоб просто змінити знак вартості дискримінатора.

$$J^{(G)} = -\frac{1}{2}E_z \log D(G(z)) \qquad 1.4$$

Тепер генератор хотів би максимізувати логарифмічну ймовірність того, що дискримінатор помиляється. Тепер рівновага не може бути описана єдиною функцією вартості, і мотивація саме цієї вартості є набагато більш евристичною.

Початкові GAN не дуже добре масштабуються для великих додатків. Щоб подолати цю проблему, Радфорт та ін. представили архітектуру глибокої згортки (Deep Convolution GAN). Хоча спочатку GAN вже були глибокими і згортковими, DCGAN наголошують на тому, щоб мати більше згорткових шарів, і додатково використовують такі методи, як пакетна нормалізація. Пакетна нормалізація застосовується до кожного шару, крім останнього шару генератора, щоб зробити процес навчання більш стабільним.

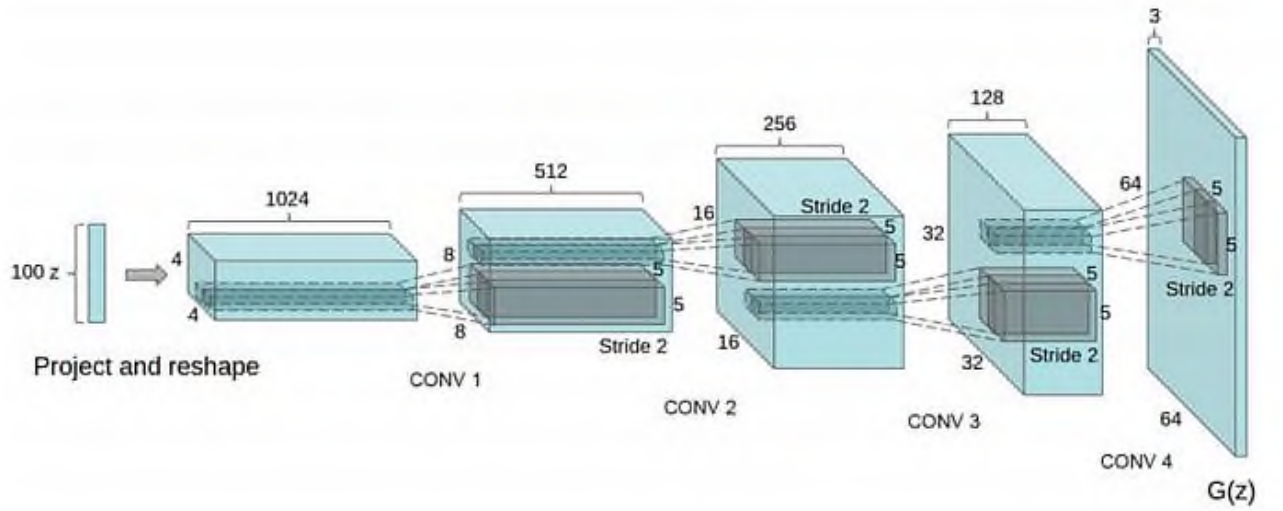

Рисунок 1.6. - Архітектура DCGAN [4]

Це не міф, що GAN штурмом захопили світ штучного інтелекту, і вони справді залишилися тут, щоб залишитися. Перш ніж завершити цей розділ, давайте подивимось на деякі з найцікавіших застосувань GAN сьогодні.

Одне з популярних застосувань полягає у створенні зображення високої роздільної здатності із зображення низької роздільної здатності за допомогою методів надвисокої роздільної здатності.

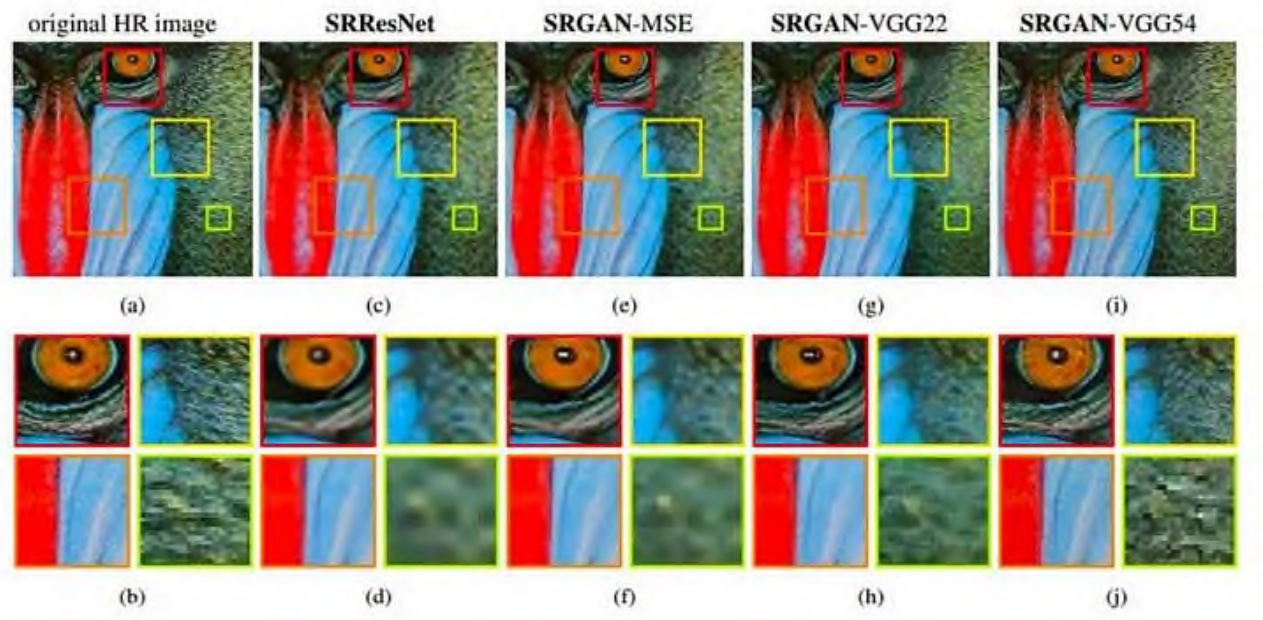

Рисунок 1.7. - SRGAN використовується для отримання зображень з високою роздільною здатністю.[5]

Оригінальне зображення HR спочатку дискретизується для отримання зображення з низькою роздільною здатністю, а потім використовуються різні методи для відновлення оригінального зображення HR.

Рисунок 1.8. - Генеративний змагальний синтез тексту та зображень [6]

Як було описано раніше, GAN також мають багато застосувань у навчанні з підкріпленням. Вони також застосовуються для згладжування зображень і створення творів мистецтва, якщо назвати декілька прикладів.

1.2. Комікси як форма мистецтва

Комікси в США, манга в Японії та Bande Dessinee «мальована стрічка» у Франції та Бельгії - це графічні романи, що мають світову аудиторію. Вони є важливою частиною американської, японської та франкомовної культур. Їх часто розглядають як м'яку силу цих країн, особливо манґи для Японії [7,3]. У Франції bande dessinee розглядається як мистецтво, і зазвичай називається "дев'ятим мистецтвом" [4] (у порівнянні з кіно, яке є сьомим мистецтвом). Однак ще кілька років тому це було не так. Комікси вважали "дитячою літературою" або "підлітературою", оскільки вони містять суміш зображень і тексту. Однак останнім часом комікси викликають великий інтерес, коли люди визнають їх складною формою графічного вираження, яка може передавати глибокі ідеї та глибоку естетику [5].

З економічної точки зору ринок коміксів є великим. Згідно зі звітом, опублікованим у лютому 2017 року "Все японською асоціацією видавців та редакторів журналів і книг" (AJPEA), продаж манґи в Японії у 2016 році становив 445,4 мільярда єн (приблизно 4 мільярди доларів) [6]. У цьому звіті ми бачимо, що ринок є стабільним у період між 2015 та 2014 роками. Однак цифровий ринок значно зріс: з 2014 по 2016 рік він майже подвоївся. Цифровий формат має кілька переваг для читачів: його можна відображати на смартфонах чи планшетах і читати будь-де і будь-коли. Для редакторів вартість публікації та розповсюдження набагато нижча порівняно з паперовою версією.

Однак, навіть якщо формат змінився з паперового на екранний, споживачеві не було запропоновано жодної додаткової цінності. Було визначено що демократизація цифрового формату є гарною можливістю для дослідників з усіх галузей комп'ютерних наук запропонувати нові послуги. А саме в створенні більшої кількості коміксів за допомогою GAN.

Дослідження коміксів є досить складним через їхню природу. Комікси містять суміш малюнків і тексту. Щоб повністю проаналізувати і зрозуміти зміст коміксів, ми повинні розглянути обробку природної мови, щоб зрозуміти історію і діалоги; і комп'ютерний зір, щоб зрозуміти лінійні малюнки, персонажів, локації, дії і т.д. Високорівневий аналіз також необхідний для розуміння подій, емоцій, розповіді, стосунків між персонажами тощо. Було проведено багато відповідних досліджень для охоплення подібних аспектів у випадку природних зображень (тобто, фотографічних зображень) і відео за допомогою класичного комп'ютерного зору. Однак велика різноманітність малюнків і дефіцит маркованих даних роблять завдання складнішим, ніж для природних зображень.

Термін комікс - це спосіб передачі інформації, подібний до телебачення, радіо тощо. Ми також можемо говорити про комікс у цьому випадку мається на увазі екземпляр носія, наприклад, комікс або комікс-сторінку.

Як і в будь-якому мистецтві, у створенні коміксів немає суворих правил. Автори можуть малювати що завгодно і як завгодно. Однак деякі класичні макети або патерни зазвичай використовуються автором, оскільки він хоче розповісти історію, передати почуття та емоції, привернути увагу читачів [7]. Автору потрібен досвід і знання, щоб плавно керувати увагою читачів за допомогою коміксів [8]. Крім того, макет коміксів з часом еволюціонує [9], відходячи від звичайних сіток до більш декоративних і динамічних способів.

Комікси зазвичай друкуються на книжках і можуть бути одно- або двосторінковими. Коли читач відкриває книгу, він бачить обидві сторінки. Таким чином, деякі автори використовують цей фізичний макет як частину історії: деякі малюнки можуть бути розміщені на двох сторінках, і коли читач перегортає одну сторінку, щось може статися на наступній сторінці. рисунок 1 ілюструє класичний зміст коміксів.

Рисунок 1.9. - Приклад подвійної сторінки коміксу

Сторінка зазвичай складається з набору панелей, що визначають певну дію або ситуацію. Панелі можуть бути укладені в рамку і розділені білим простором, який називається "жолоб" (gutter). Порядок читання панелей залежить від мови. Наприклад, в японській мові порядок читання зазвичай справа наліво і зверху вниз. Мовленнєві кульки та підписи включені в панель для опису розмов або розповіді історії. Діалогові кульки мають певний порядок читання, який зазвичай збігається з порядком читання панелей. Деякі звукові ефекти або ономатопеї включені, щоб дати читачеві більше відчуттів, таких як запах або звук. Японські комікси часто містять "манпу" рисунок 1.10 - графічні символи, які використовуються для візуалізації почуттів і відчуттів персонажів, наприклад, сліди поту на голові, щоб показати, що він відчуває дискомфорт, навіть якщо він насправді не пітніє.

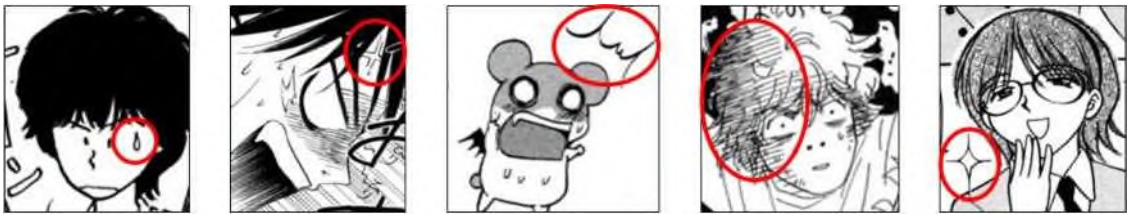

Рисунок 1.10. - Деякі приклади "манпу"

Незважаючи на те, що з'являється дедалі більше оцифрованих версій друкованих видань, лише деякі комікси створюються в цифровому форматі з використанням переваг нової технології.

Рисунок 1.1. - Цифровий комікс "Протанопія"

На рисунку 1.11. показано приклад цифрового коміксу, що використовує переваги функцій планшета: зображення безперервно анімовані, а користувач може нахиляти планшет, щоб керувати кутом камери. Цей комікс створений Андре Бергсом [12] і знаходиться у вільному доступі в App Store та Google Play. Ми уявляємо, що в майбутньому можна буде створювати такі інтерактивні комікси автоматично завдяки алгоритмам комп'ютерних наук.

1.3. Застосування генерації контенту в створенні коміксів

Застосування генерування контенту у створенні коміксів є відносно новим, але стрімко розвивається напрямком у сфері штучного інтелекту та комп'ютерної графіки. Останнім часом було проведено низку досліджень та експериментів, які демонструють потенціал у цій області.

Метою створення або збагачення контенту є використання коміксів для створення нового контенту на основі коміксів або інших медіа.

Векторизація. Оскільки більшість коміксів створюються не в цифровому вигляді, векторизація - це спосіб перетворення відсканованих коміксів у векторне представлення для рендерингу в реальному часі з довільною роздільною здатністю [44]. Створення векторизованих коміксів необхідне для їх гарної візуалізації в оцифрованому середовищі. Це також важливий крок для редагування змісту коміксів і один з основних етапів збагачення коміксів [45].

Колоризація. Було запропоновано кілька методів для автоматичного розфарбовування [46-50] та реконструкції кольорів [51], оскільки комікси з кольорами можуть бути більш привабливими для деяких читачів. Колоризація є досить складною проблемою, оскільки різні частини персонажа, такі як руки, кисті, пальці, обличчя, волосся, одяг і т.д., повинні бути знайдені, щоб правильно розфарбувати кожну частину. Крім того, пози персонажа можуть сильно відрізнятися одна від одної: деякі частини можуть з'являтися, зникати або деформуватися. Приклад розфарбовування показано на рисунку 1.12.

Рисунок 1.12. - Приклад процесу розфарбовування на основі style2paints.

Нещодавно підхід, заснований на глибокому навчанні, був використаний для створення кольорових версій манґа-книг, які розповсюджуються професійними компаніями в Японії [53].

Створення коміксів та персонажів. Однією з проблем при створенні коміксів є створення макету та розміщення різних компонентів, таких як персонажі, текстові кульки тощо, у правильному положенні для забезпечення комфортного читання. Цао та ін. запропонували метод автоматичного створення

стилістичного макета [54], а також метод розміщення та організації елементів на панелі відповідно до високорівневих специфікацій користувача [8].

Зв'язок між реальним середовищем і середовищем, представленим у коміксах, може бути використаний для створення або доповнення коміксів. Ву та Айзава запропонували метод створення коміксів безпосередньо з фотографії [55].

Наприкінці 2017 року Jin та ін. [56] представили метод автоматичної генерації персонажів коміксів. Приклад згенерованого персонажа за їхньою онлайн-демонстрацією [57] показано на рисунку 1.13. Результат генерації не завжди є візуально досконалим, але це все одно потужний інструмент, оскільки можна згенерувати необмежену кількість персонажів.

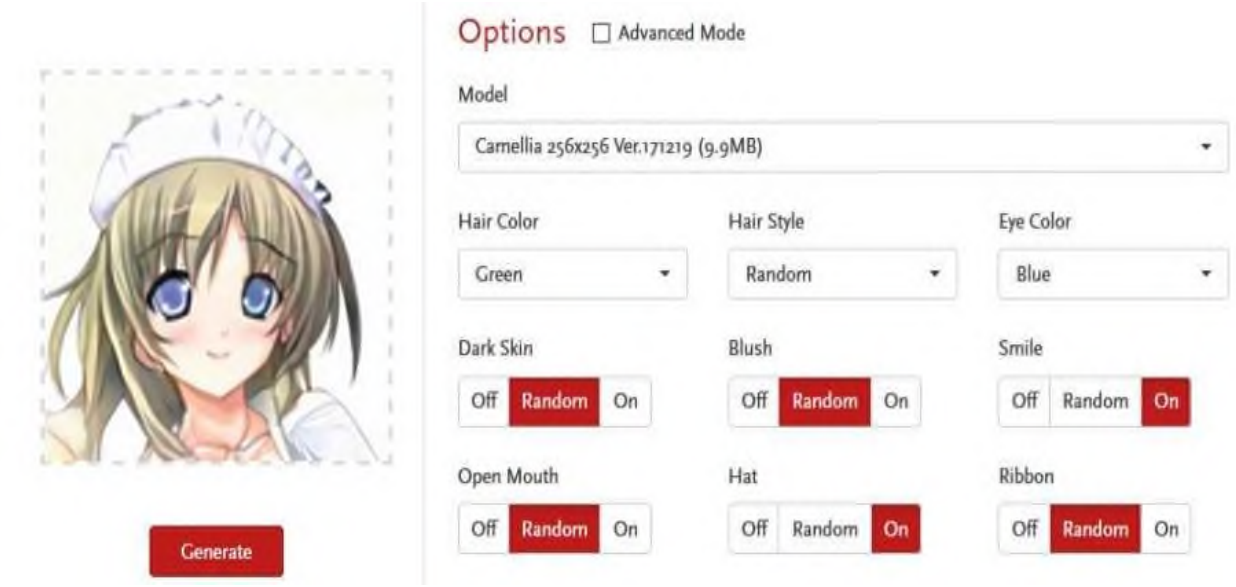

Рисунок 1.13. Приклад генерації випадкових символів

Анімація. Оскільки комікси є нерухомими зображеннями, одним із способів покращення візуалізації коміксів є створення анімації. Нещодавно деякі дослідники запропонували спосіб анімації нерухомих зображень коміксів за допомогою рухів камери [58,59]. Кілька анімаційних фільмів і серіалів були адаптовані в комікси і навпаки. Перспектива може полягати у створенні анімаційного фільму з паперових коміксів або паперових коміксів з анімаційного фільму.

Для природних зображень було запропоновано кілька методів оживлення облич людей за допомогою латентних просторових інтерполяцій. Як показано на рисунку 9, латентні вектори можна обчислити для нейтрального та усміхненого обличчя, щоб згенерувати анімацію усмішки [60].

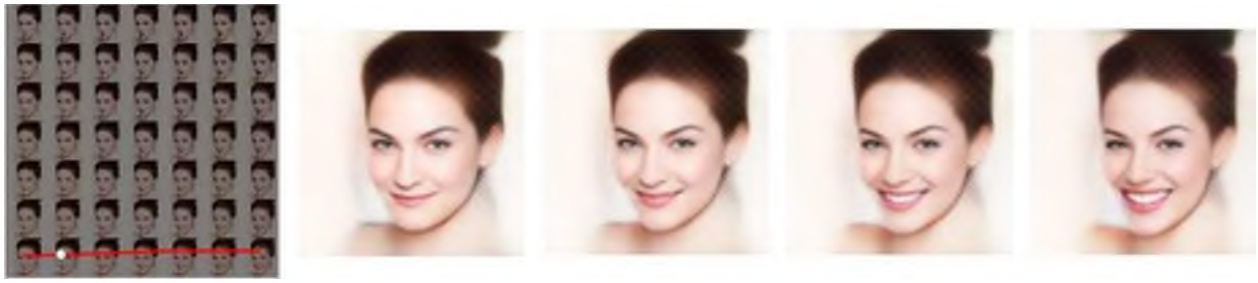

Рисунок 1.14. - Приклад анімації усмішки в концептуальному просторі

Інше застосування полягає у вилученні ключових точок обличчя та використанні іншого джерела (тексту, промови або обличчя) для анімування рота персонажа. Це було використано для створення фотореалістичного відео промови Обами на основі текстового введення [62].

Перетворення медіа. У більш загальному плані можна уявити собі перетворення тексту, відео або будь-якого контенту на комікси і навпаки. Цю проблему можна розглядати як конвертацію медіа. Наприклад, Jing та ін. запропонували систему для перетворення відео на комікси [63]. Для успішного перетворення існує багато викликів: узагальнення відео, стилізація зображень, генерація макету коміксів та позицій текстових кульок тощо.

Застосування, яке не було зроблено для коміксів, але для природних відео, полягає в додаванні згенерованого звуку до відео [64]. Для коміксів такого застосування не було, але цей метод можна застосувати для створення звукових ефектів (мечі, що б'ються один об одного, ревіння вихлопної труби тощо) або звуків атмосфери (село, сільська місцевість, натовп і т.д.).

Створення описового тексту на основі коміксів або генерування коміксів на основі описового тексту може стати можливим у майбутньому, як це було зроблено для природних зображень. Рід та ін. [65] запропонували метод автоматичного синтезу реалістичних природних зображень з тексту.

Адаптація вмісту. Зміст коміксів також може бути змінений, щоб краще відповідати пристрою. Наприклад, Арай і Толле запропонували систему для аналізу макета манги, а потім її сегментації з метою адаптації до мобільних пристроїв, таких як КПК або смартфони [32].

У тій же статті Арай і Толле також запропонували метод виявлення тексту на зображенні та його перекладу іншою мовою для адаптації контенту до мови користувача.

1.4. Постановка задачі

Давайте проведемо визначення цілей та завдань дослідження. Основною метою даного дослідження є розробка та аналіз методу генерації коміксів за допомогою генеративних змагальних мереж (GAN). Для досягнення цієї мети визначено наступні основні завдання:

- Аналіз існуючих технік GAN: Вивчення різних архітектур GAN, що застосовуються для генерації зображень, з метою вибору найбільш підходящої моделі для створення коміксів.
- Підбір або розробка набору даних для навчання: Створення або адаптація існуючого набору даних, який містить різноманітні зразки коміксів, для ефективного тренування GAN.
- Експерименти з генерацією коміксів: Навчання та тестування GAN на підготовленому датасеті з метою генерації коміксових зображень і сцен.
- Оцінка якості генерованих зображень: Аналіз та оцінка реалістичності, оригінальності, та візуальної привабливості згенерованих коміксів.
- Дослідження можливостей застосування: Розгляд потенційних сфер застосування методу, включаючи комерційне використання, освітні цілі, та розваги.

Мотивація для цього дослідження випливає зі зростаючого інтересу до автоматизації творчих процесів за допомогою штучного інтелекту. Застосування GAN для створення коміксів може значно збільшити швидкість та обсяг

виробництва в цій сфері, а також дозволити митцям та сценаристам експериментувати з новими ідеями без значних витрат часу та ресурсів.

Потенційний вплив дослідження є значним. У разі успіху, методи, розроблені в рамках цього проекту, можуть відкрити нові можливості для індустрії коміксів, дозволяючи швидко створювати високоякісний контент. Це також може привести до розвитку нових форм наративу та візуального стилю в коміксах, збільшивши творчі горизонти для митців та письменників. Крім того, дослідження може мати вплив на інші галузі, де використовуються візуальні наративи, включаючи рекламу, кіно, та відеоігри.

Висновок до розділу 1

1. В підрозділі 1.1. демонструє, як GAN стали значущою частиною сучасних досліджень у галузі штучного інтелекту. Від їх винайдення у 2014 році, GAN зазнали значного розвитку та вдосконалення, подолавши ранні проблеми, такі як "mode collapse". Їх здатність генерувати реалістичні зображення і застосування в різноманітних сферах, включаючи мистецтво та медичні зображення, відкриває нові перспективи для досліджень і інновацій. Порівняння з дискримінативними моделями і детальний огляд різних підходів до генеративного моделювання подає зрозуміле уявлення про унікальність та потенціал GAN.

2. Також було розкрито важливість коміксів як культурного та художнього феномену у різних країнах. Вона підкреслює їхню трансформацію з "дитячої літератури" до визнаної форми графічного вираження, що вміло поєднує зображення та текст для передачі глибоких ідей та емоцій. Економічний аспект, особливо зосередження на манзі в Японії, вказує на величезний потенціал і вплив цієї індустрії, що продовжує розвиватися у цифрову еру.

3. Було розкрито інноваційні підходи до створення коміксів за допомогою технологій, зокрема GAN. Вона охоплює широкий спектр аспектів, включаючи векторизацію, колоризацію, створення персонажів, анімацію та адаптацію вмісту. Цей розділ підкреслює потенціал застосування штучного інтелекту для революціонізації способу створення та презентації коміксів, відкриваючи можливості для більшої творчості та інтерактивності.

4. Ввизначено конкретні цілі та завдання для дослідження використання GAN у створенні коміксів. Завдання включають аналіз існуючих технік GAN, підготовку даних, експерименти з генерацією коміксів, та оцінку якості згенерованих зображень. Цілі дослідження відображають прагнення інтегрувати новітні технології в традиційне мистецтво створення коміксів, вказуючи на значний потенціал впливу на індустрію коміксів і творчість загалом.

2. МЕТОДИ І ПІДХОДИ ДЛЯ ГЕНЕРАЦІЇ КОМІКСІВ

2.1. Архітектури GAN для генерації зображень

Генеративні змагальні мережі (GAN) мають кілька основних архітектур, кожна з яких має свої унікальні особливості та способи застосування. Вибір конкретної архітектури залежить від специфіки задачі, наявності даних та бажаних результатів. Важливо також враховувати обмеження у вигляді обчислювальних ресурсів, оскільки деякі архітектури вимагають значно більше обчислювальної потужності, ніж інші. Нижче наведено декілька з найбільш відомих та ефективних архітектур GAN:

1. Deep Convolutional GAN (DCGAN): Ця архітектура вносить зміни в структуру генератора та дискримінатора, використовуючи конволюційні нейронні мережі. DCGAN відома покращенням якості згенерованих зображень та стабільністю навчання.

2. Архітектура WGAN включає кілька ключових особливостей. WGAN складається з двох частин -генератора та критика (замість дискримінатора). Критик оцінює зображення, що генерує генератор, але замість прогнозування ймовірності (справжнє чи штучне), він надає скалярну оцінку реалістичності зображення

3. StyleGAN: Останнім часом StyleGAN стала однією з найпопулярніших архітектур, особливо для створення реалістичних облич. Вона вводить поняття стилів, які можуть контролювати різні аспекти згенерованого зображення.

4. StyleGAN2, являє собою удосконалену версію оригінальної архітектури StyleGAN, запропоновану командою Nvidia, яка забезпечує покращену якість зображень та більшу стабільність тренування. Основні нововведення StyleGAN2 включають виправлення проблем з нереалістичними артефактами та впровадження змін у механізм мапінгу стилів, що дозволяє досягати більш точного контролю над візуальними аспектами згенерованих зображень.

5. StyleGAN2-ADA (Adaptive Discriminator Augmentation) - це розширення архітектури StyleGAN2, розроблене для покращення тренування генеративних змагальних мереж на невеликих датасетах. Основна ідея полягає у динамічному

застосуванні аугментації до зображень лише в тому випадку, коли дискримінатор починає перевчитися, що допомагає уникнути звуження розподілу даних та підтримує ефективність тренування. Вибір конкретної архітектури залежить від специфіки задачі, наявності даних та бажаних результатів. Важливо також враховувати обмеження у вигляді обчислювальних ресурсів, оскільки деякі архітектури вимагають значно більше обчислювальної потужності, ніж інші.

Для визначення найбільш підходящої моделі, розглянемо кожну з моделей, зазначених вище, детальніше. Можливо, здасться неочікуваним включення в дослідження моделі, як DCGAN, розробленої ще у 2015 році. У світі штучного інтелекту, де технології розвиваються дуже швидко, вісім років може здатися довгим періодом. Проте DCGAN залишається актуальною через свою простоту, стабільність тренування та здатність досягати задовільних результатів. Особливо цінним є той факт, що DCGAN можна ефективно навчати на стандартних комп'ютерах без необхідності використання величезних датасетів на потужних обчислювальних системах. Як зазначається у джерелі [8], багато популярних та успішних розробок GAN надихнуті прогресом у галузі комп'ютерного зору, особливо конволюційними нейронними мережами (CNN). Як приклад, DCGAN, який значно підвищує якість зображень завдяки своїй архітектурі, використовує набір конволюційних операцій, включаючи техніку просторового масштабування для вдосконалення генератора. Відомо, що DCGAN покращує якість згенерованих зразків, що робить його стандартом для порівняння з іншими моделями GAN.

У цьому дослідженні використовується генератор G, який починається з 100-вимірного випадкового вектора z. Через чотири шари конволюцій з дробовими стрибками, згенеровано зображення з роздільною здатністю 64x64. Дискримінатор D має схожу, але обернену структуру, завершуючись одним значенням ймовірності. В оригінальному дослідженні було доведено, що GAN оптимізують дивергенцію Дженсена-Шеннона (JSD).

$$JSD(P_r||P_g) = \frac{1}{2}KL(P_r||P_A) + \frac{1}{2}KL(P_g||P_A) \qquad 2.1$$

де *KL* позначає дивергенцію Кульбака-Лейблера, а P_A визначається як "середній" розподіл $P_A = \frac{P_r+P_g}{2}$. Під час тренування DCGAN ми маємо інформацію лише про втрати генератора G та дискримінатора D, які, в ідеальному випадку, повинні прагнути до значення 0,5, сподіваючись, що це забезпечить мінімізацію розбіжності між модельним та реальним розподілами даних. Проте сама архітектура DCGAN не зорієнтована безпосередньо на мінімізацію кількісної метрики цієї розбіжності, що стає основною метою WGAN.

Проблема полягає в тому, що як реальний розподіл даних, так і модельний розподіл знаходяться на низьковимірних многовидах у вищевимірному просторі, і малоймовірно, що вони матимуть значний перетин. Це означає, що відстань Кульбака-Лейблера не є визначеною або є нескінченною. У статті, на яку посилаються, надано доказ того, що відстань Вассерштейна має кращі характеристики, ніж відстань Єнсена-Шеннона або відстань Кульбака-Лейблера. Відстань Вассерштейна, також відома як відстань Earth-Mover, визначається як мінімальна кількість "роботи", необхідна для перетворення одного розподілу в інший, що робить її більш придатною для оцінки реалістичності згенерованих моделлю даних.

$$W(P_r, P_g) = \inf_{\gamma \in \prod(P_r, P_g)} E_{(x,y)\sim\gamma}[||x - y||] \qquad 2.2$$

де $\prod(P_r, P_g)$позначає множину всіх спільних розподілів $\gamma(\mathrm{x},\mathrm{y})$, граничні значення яких відповідно дорівнюють P_r та P_g . Інтуїтивно, $\gamma(\mathrm{x},\mathrm{y})$ вказує, скільки "маси" потрібно перевезти з *x* в *y*, щоб перетворити розподіл P_r у розподіл P_g. Тоді відстань ЕМ є "вартістю" оптимального плану перевезень. Однак, інфімум дуже складним для розв'язання", і за допомогою дуальності Канторовича-Рубінштейна, яка використовує функцію *K*- Ліпшиця, в роботі запропоновано простий, але грубий спосіб наближення цієї відстані Вассерштейна, шляхом відсікання модельної ваги. У цьому алгоритмі *D*

навчається до оптимальності на кожному кроці. Враховуючи що [18] аргумент простий: чим більше ми тренуємо критика, тим краще, тим надійніший градієнт Вассерштейна ми отримаємо і можливо, ще важливіше те, що той факт, що ми можемо навчати критика до оптимальності, унеможливлює колапс режимів, коли ми це робимо. Оскільки функція втрат є оцінкою відстані EM, на відміну від DCGAN, ми тепер маємо значущу метрику втрат як показник якості згенерованих зразків під час навчання. Втрати повинні послідовно зменшуватися під час навчання, в той час як якість зразків повинна зростати.

На опису WGAN, варто згадати зауваження, зроблене в статті [14] варіанти GAN з функцією втрат, хоча часто добре обґрунтовані, такі як WGAN, показують, що їх результати також можуть бути досягнуті шляхом відповідної гіперпараметричної оптимізації архітектури варіантів GAN, таких як DCGAN. Всебічне порівняння мереж GAN продемонструвало невелике поліпшення загальних показників у сучасних мережах GAN з втратами порівняно з такими архітектурами, як DCGAN, і що доцільніше зосередитися на налаштуванні гіперпараметрів, а не на впровадженні нових методів.

Коли було запропоновано StyleGAN [23], це стало великим стрибком уперед у якості зображень. Автори статті стверджували, що "генератори продовжують працювати як чорні скриньки, і, незважаючи на нещодавні зусилля, розуміння різних аспектів процесу синтезу зображень, наприклад, походження стохастичних особливостей, все ще бракує". Вони запропонували повністю переробити модель. Було внесено дві основні зміни: виокремлення латентного простору та пряме керування особливостями зображення на різних масштабах.

StyleGAN відходить від ідеї випадкового вектора *z*, який генерує зображення після того, як воно пройшло через мережу прямого поширення. Цього разу використовуються дві мережі: мережа відображення *f* та мережа синтезу *g* (див. рисунок 2.1). На вхід мережі синтезу g подається постійний тензор. Він поширюється по мережі і створює зображення. Для того, щоб внести зміни до згенерованих зображень , характеристики зображення налаштовуються на різних рівнях мережі. Це робиться через мережу відображення f . І саме тут знову з'являється випадковий вектор z. Вектор z береться випадковим чином з

латентного простору Z і тепер спочатку поширюється через нелінійну мережу відображення f для створення нового вектора w в латентному просторі W. Після цього w використовується для управління адаптивною нормалізацією екземплярів на кожному шарі згортки.

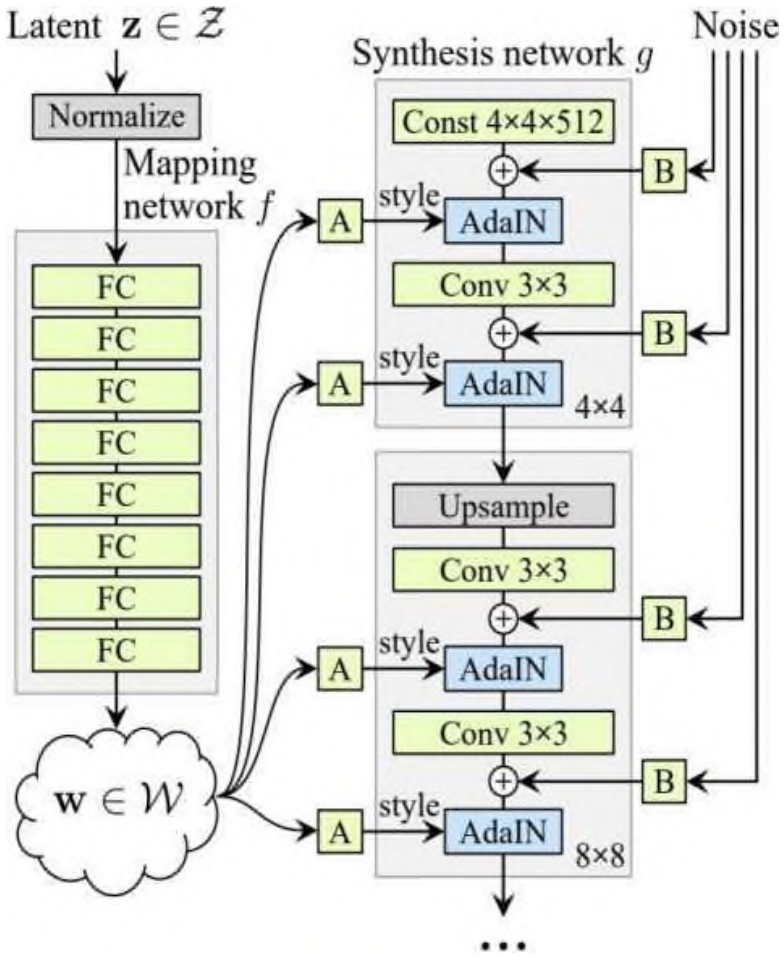

Рисунок 2.1. - StyleGAN

Пояснюючи [] для простоти ми встановили розмірність обох просторів 512, а відображення f реалізовано за допомогою 8-шарового MLP. Вивчені афінні перетворення потім спеціалізують *w* на стилі $y=(y_s,y_b)$, які керують операціями адаптивної нормалізації екземплярів (AdaIN) після кожного шару згортки мережі синтезу *g*. Операція AdaIN визначається як

$$AdaIN(x_i, y) = y_{s,i}\frac{x_i - \mu(x_i)}{\sigma(x_i)} + y_{b,i} \qquad 2.3$$

де кожна карта ознак xi нормалізується окремо, а потім масштабується та зміщується за допомогою відповідних скалярних компонент зі стилю *y*. Таким чином, розмірність *y* - це вдвічі більше, ніж кількість карт об'єктів на цьому шарі".

Причина використання картографічної мережі пояснюється в статті на прикладі набору даних обличь. У наш час чоловіки з довгим волоссям зустрічаються досить рідко. Це яскравий приклад переплетення двох стильових ознак (статі та довжини волосся). У латентному просторі *Z* моделі, навченої на наборі обличь, є регіони, для яких немає або дуже мало прикладів довговолосих чоловіків. Але випадковий вектор *z* вибирається з нормального розподілу. Отже, класична мережа *G* повинна буде вивчити деформацію *Z* у своєму відображенні, щоб уникнути цих заборонених зон. У StyleGAN це вирішується завдяки мережі відображення f . Спочатку латентний простір *Z* перетворюється на латентний простір *W* тієї ж розмірності 512, так що значної частини цього викривлення можна уникнути в генераторі.

На кожному рівні вводиться додатковий шум, який був попередньо помножений на вивчений коефіцієнт масштабування. Пояснення цьому таке [] можна припустити, що в будь-якій точці генератора існує тиск з метою якнайшвидшого введення нового контенту, і найпростіший спосіб для нашої мережі створити стохастичну варіацію - це покластися на наданий шум. Для кожного шару доступний свіжий набір шуму, а отже, немає стимулу генерувати стохастичні ефекти від попередніх активацій, що призводить до локального ефекту.

Функціями втрат є WGAN-GP та насичуванні втрати з регуляризацією R_1 Автори також уникають вибірки з областей *W* з меншою ймовірністю за допомогою так званого трюку усічення, але тільки для нижчих роздільних здатностей згенерованого зображення. Така вибірка з усіченого латентного простору покращує якість зображення, але також обмежує кількість варіацій.

Інші методи стабілізації не використовуються. Як зазначено в статті [] жодним чином не модифікується дискримінатор або функцію втрат, і роботи з

цією мережою є ортогональною до поточної дискусії про функції втрат, регуляризацію та гіперпараметри GAN.

StyleGAN2 [23] було впроваджено для вирішення деяких артефактів на зображеннях, згенерованих StyleGAN. Результати дійсно значно кращі. Основні зміни полягають у перегляді нормалізації екземплярів, редизайні нормалізації, що використовується в генераторі, та відмові від прогресивної структури GAN. Архітектура StyleGAN змінена наступним чином. По-перше, модуляцію та нормалізацію вилучено з потоку екземплярів, які безпосередньо впливають на ваги у згортці. Як пояснюється в статті: "Основна ідея полягає в тому, щоб базувати нормалізацію на очікуваній статистиці вхідних зображень, але без явного примусу. [...] Модуляція масштабує кожну вхідну карту ознак згортки на основі вхідного стилю, що може бути альтернативно реалізовано шляхом масштабування ваг згортки:

$$w'_{ijk} = s_i * w_{ijk} \tag{2.4}$$

де w та w' - вихідна та модульована ваги відповідно, s_i - масштаб, що відповідає і-й вхідній карті ознак, а j та k - вихідні карти ознак та просторовий слід згортки відповідно. Мета нормалізації екземплярів полягає в тому, щоб суттєво усунути вплив s зі статистики вихідних карт ознак згортки. Ми бачимо, що цієї мети можна досягти більш прямим шляхом".

Нормалізація зводиться до того:

$$w''_{ijk} = \frac{w'_{ijk}}{\sqrt{\sum_{i,k} {w'_{ijk}}^2 + \epsilon}} \tag{2.5}$$

де ϵ - невелика константа, щоб уникнути числових проблем. На рисунку 2.2 показано переглянуту архітектуру.

Друга важлива зміна пов'язана з усвідомленням того, що довжина шляху сприйняття (PPL) є метрикою, пов'язаною з якістю зображення. Як зазначено в []: "Ми спостерігаємо кореляцію між якістю сприйманого зображення і

довжиною шляху сприйняття (PPL), метрикою, яка спочатку була введена для кількісної оцінки гладкості відображення з латентного простору до вихідного зображення шляхом вимірювання середніх LPIPS-розбіжностей між згенерованими зображеннями при невеликих збуреннях в латентному просторі". LPIPS розшифровується як Learned Perceptual Image Patch Similarity і "обчислюється як зважена різниця між двома вбудовуваннями VGG16"[]. Хоча авторам не одразу зрозуміло, чому PPL має корелювати з якістю зображення, вони вбудовують регуляризатор у генератор, щоб спонукати його отримувати добре обумовлене відображення з латентного простору W у простір зображень. []: "На практиці ми помічаємо, що регуляризація довжини шляху призводить до більш надійних і стабільних моделей, що полегшує дослідження архітектури". Однак це призводить до компромісу між PPL та FID у погано структурованих наборах даних. Таку регуляризацію довжини шляху також не можна налаштувати в реалізації PyTorch StyleGAN2, яку я використовував.

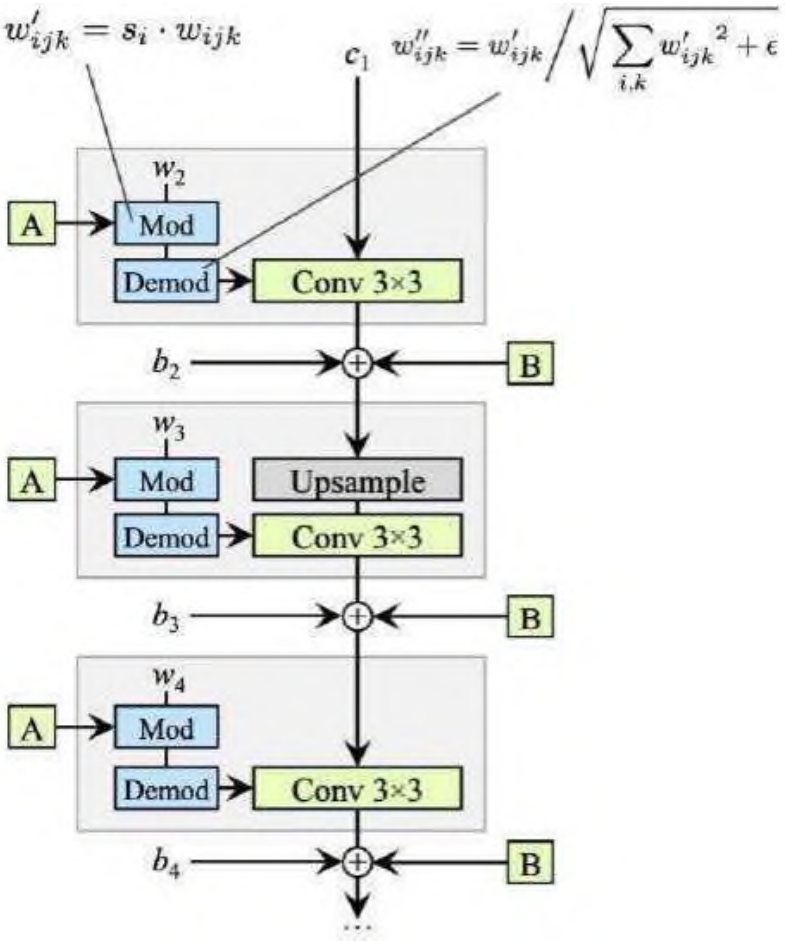

Рисунок 2.2. StyleGAN2

StyleGAN був заснований на ідеях прогресивно зростаючої GAN [34], але StyleGAN2 відходить від цієї архітектури. Експерименти показали, що G складається з пропущених зв'язків, а D - із залишкових мереж. Стаття доводить, що філософія прогресивного зростання не була втрачена. Замість того, щоб нав'язувати

Якщо робити це вручну, генератор, здається, спочатку зосереджується на низькій роздільній здатності, щоб потім приділяти більше уваги високій роздільній здатності.

Як результат, виявилося, що проекції зображень у латентному просторі також легше виконувати. Це техніка, в якій шукається оптимальний латентний вектор w для наданого зображення. Якщо зображення попередньо згенероване, це вдається майже ідеально. однак є явні відхилення, коли це реальне зображення, якого не було в набору даних.

Перенесення стилю також є однією з можливостей. Це відбувається шляхом об'єднання векторів *w* двох спроектованих зображень в один новий вектор *w*. Вбудовування я відбувається в розширеній версії в латентному просторі *W+* . *"W+* - це конкатенація 18 різних 512-вимірних векторів w, по одному для кожного рівня архітектури StyleGAN, які можуть отримувати вхідні дані через AdaIn"[]. Нижня половина першого вектора впливає на грубі риси зображення, а верхня половина другого вектора визначає дрібні риси.

Як згадувалося раніше, навчанню часто перешкоджає надмірне налаштування дискримінатора. Коли *D* стає надмірно самовпевненим, він більше не дає цікавих градієнтів, з яких *G* може навчитися чомусь корисному. [35] є цікавим дослідженням ефектів широкого спектру доповнень до зображень, що призводить до кращої збіжності моделі, без витоку доповнень у згенеровані зображення. У дуже простому підході доповнення застосовуються як до фальшивих, так і до справжніх зображень, перш ніж вони будуть оцінені генератором. Пояснюється за допомогою чіткої метафори в [36]: "Доповнення дискримінатора відповідає надяганню на дискримінатор спотворюючих, можливо, навіть руйнівних окулярів і проханню до генератора видавати зразки,

які неможливо відрізнити від навчальної вибірки, якщо дивитися на них крізь окуляри". Єдина умова полягає в тому, що перетворення є інвертованими. Це не означає, що окремі маніпуляції із зображенням мають бути незворотними. Це означає, що все ще має бути можливість міркувати про початковий розподіл. Як проілюстровано в [37]: "[...] випадкові обертання, вибрані рівномірно з {0°, 90°, 180°, 270°} не є оберненими: неможливо розрізнити відмінності між орієнтації після доповнення. Ситуація змінюється, якщо це обертання виконується лише з ймовірністю $p<1$: це збільшує відносну частоту появи 0°, і тепер доповнені розподіли можуть співпасти лише за умови, що згенеровані зображення мають правильну орієнтацію". Експерименти привели до максимального значення цієї ймовірності p = 0,8, при якому витоки є малоймовірними. Запропоновано конвеєр аугментації, який складається з блітінгу пікселів3, геометричних і колірних перетворень. Певною мірою він залишається пошуковим, оскільки [38] "[...] оптимальна сила аугментації сильно залежить від кількості навчальних даних, і не всі категорії аугментації однаково корисні на практиці". ADA, що розшифровується як Adaptive Discriminator Augmentation, має навіть вбудоване налаштування аугментації міцність, щоб оптимально уникнути надмірного припасування.

Для коміксів не повинно бути жодного витоку аугментації. Це не є реальною проблемою у випадку горизонтального перевертання або ізотропного масштабування. Це трансформації, які часто використовуються в процесі створення. Але про корекцію кольору або розвороти на 90° не може бути й мови. У коміксах часто використовується фіксована кольорова палітра, і ви не хочете, щоб персонажі на панелях виглядали перевернутими, якщо тільки це не було зроблено навмисно.

2.2. Архітектура мережі StyleGAN2-ADA

StyleGAN2-ADA (Adaptive Discriminator Augmentation) є розширенням архітектури StyleGAN2, яке вводить адаптивну аугментацію для дискримінатора, спеціально розроблену для покращення тренування моделей на невеликих

датасетах. Ця архітектура створена командою дослідників з Nvidia і вирішує кілька ключових викликів, які стоять перед попередніми версіями StyleGAN. Нижче наведено детальний опис основних аспектів StyleGAN2-ADA:

Динамічне Застосування Аугментацій: Основна ідея ADA полягає у використанні аугментацій (таких як зміни масштабу, обертання, відображення тощо) способом, який адаптується до ступеня перенавчання дискримінатора. Це означає, що чим більше дискримінатор схильний до перенавчання, тим інтенсивніші аугментації застосовуються.

Регулювання Інтенсивності Аугментації: У StyleGAN2-ADA інтенсивність аугментації регулюється автоматично на основі моніторингу показника перенавчання, дозволяючи моделі ефективно тренуватися навіть на обмежених датасетах без втрати якості генерованих зображень.

Виправлення Артефактів: StyleGAN2-ADA вирішує деякі проблеми з артефактами, які спостерігалися у попередніх версіях, такі як небажані візуальні відмінності та помилки в текстурах.

Ефективніше Використання Ресурсів: Оптимізована архітектура забезпечує більш ефективне використання обчислювальних ресурсів, поліпшуючи якість згенерованих зображень при меншому обчислювальному навантаженні.

Тренування на Малих Датасетах: Однією з ключових переваг StyleGAN2-ADA є її здатність ефективно тренуватися на значно менших датасетах, ніж це потрібно для традиційних моделей GAN, зменшуючи ризик перенавчання.

Різноманітність та Якість Зображень: Завдяки адаптивній аугментації та вдосконаленій архітектурі, StyleGAN2-ADA здатна генерувати зображення високої якості з більшою різноманітністю та меншою кількістю артефактів.

StyleGAN2-ADA є значним досягненням у розвитку генеративних змагальних мереж, забезпечуючи високу якість зображень навіть при обмежених датасетах, що робить її ідеальним вибором для широкого спектру застосувань у галузі генерації зображень.

Мета StyleGAN2-ADA полягає у створенні передового підходу для тренування генеративних змагальних мереж (GAN) у ситуаціях, коли доступ до

великих датасетів є обмеженим або неможливим. Акронім ADA в цьому контексті означає "адаптивне доповнення дискримінатора" (Adaptive Discriminator Augmentation), що вказує на ключову інновацію цієї архітектури. Ця інновація полягає в тому, що процес аугментації (або доповнення) даних для дискримінатора в моделі GAN є адаптивним, тобто він налаштовується відповідно до поточного стану навчання мережі.

Адаптивна аугментація в StyleGAN2-ADA призначена для збалансування тренувального процесу, особливо у випадках, коли доступні лише обмежені обсяги даних. У звичайних умовах, тренування GAN на невеликих датасетах часто призводить до перенавчання дискримінатора, що в свою чергу може негативно позначитися на якості генерації зображень. StyleGAN2-ADA вирішує цю проблему, використовуючи методику стохастичного доповнення, яка дозволяє дискримінатору отримувати більш різноманітні дані в процесі навчання. Це допомагає уникнути перенавчання, забезпечуючи при цьому ефективне навчання генератора.

Суть адаптивного доповнення полягає в динамічному зміні рівня і інтенсивності аугментації на основі поточного стану моделі. Якщо модель показує ознаки перенавчання, аугментація стає більш інтенсивною, додаючи різноманітності до тренувальних даних і тим самим підвищуючи здатність дискримінатора до генералізації. Цей процес адаптації відбувається безперервно в ході тренування, що робить StyleGAN2-ADA особливо ефективною для сценаріїв із обмеженими даними.

Завдяки цій методиці, StyleGAN2-ADA відкриває нові можливості для дослідників та розробників, які працюють у сфері генерації зображень, особливо там, де великі датасети недоступні або їх збір є непрактичним. Ця модель не лише підвищує якість генерації зображень, але й робить процес більш гнучким та доступним.

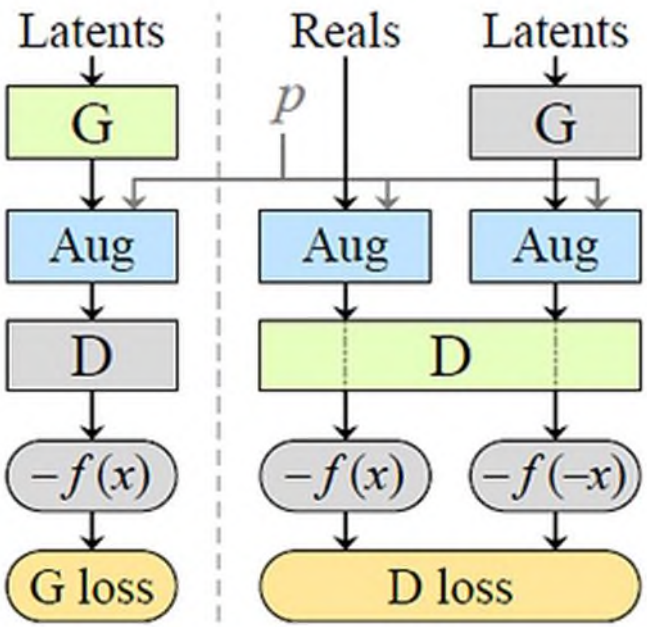

Рисунок 2.3. - Доповнення стохастичного дискримінатора

У контексті архітектури StyleGAN2-ADA, G та D використовуються для позначення двох ключових компонентів моделі: G є генератором, а D - дискримінатором. Генератор (G) відповідає за створення зображень, які намагаються імітувати реальні дані, в той час як дискримінатор (D) аналізує як згенеровані, так і реальні зображення, намагаючись розрізнити їх.

Однією з ключових особливостей StyleGAN2-ADA є використання адаптивної аугментації, що регулюється параметром p, який лежить в діапазоні від 0 до 1. Цей параметр p визначає ймовірність застосування аугментації до кожного зображення, що подається на вхід дискримінатора D. Інтуїтивно, p відображає ступінь інтенсивності аугментації: нижчі значення p вказують на меншу ймовірність застосування аугментації, тоді як вищі значення p сприяють більш інтенсивному застосуванню аугментаційних технік.

Цей механізм аугментації є важливим для підтримки балансу тренування в StyleGAN2-ADA, особливо коли працюємо з обмеженими датасетами. Адаптація інтенсивності аугментації на основі показника p дозволяє ефективно управляти тренуванням дискримінатора, зменшуючи ризик перенавчання і підвищуючи якість згенерованих зображень. Цей підхід забезпечує, що дискримінатор

отримує достатньо різноманітні дані навіть з обмеженого набору зразків, сприяючи ефективному та збалансованому навчанню моделі.

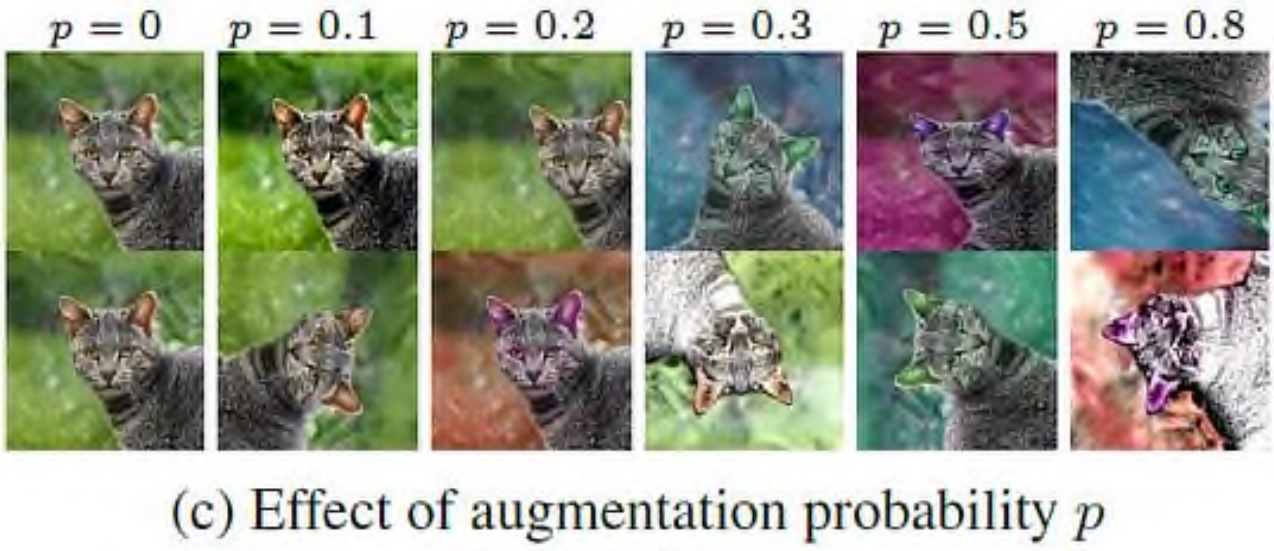

Рисунок 2.4. - Сила аугментації

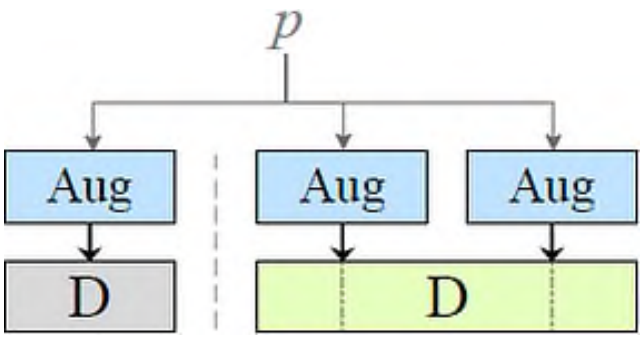

Рисунок 2.5. - Аугментація перед подачею в дискримінатор

Дискримінатор D у архітектурі StyleGAN2-ADA рідко має можливість аналізувати чисті, незмінені зображення, і ось чому:

1. Наявність Різноманітних Аугментацій: У архітектурі передбачено використання різноманітних методів аугментації у процесі тренування. Це означає, що більшість зображень, які надходять до дискримінатора, піддаються різним видам обробки, таким як зміни кольору, обертання, масштабування або відображення. Ця стратегія допомагає у боротьбі з перенавчанням, особливо коли використовуються обмежені датасети.

2. Високе Значення Ймовірності Аугментації (p): У StyleGAN2-ADA, ймовірність застосування аугментації (означена як p) часто встановлюється на високому рівні, близько 0.8. Це означає, що близько 80% зображень, які аналізуються дискримінатором, піддані аугментаціям. Такий підхід забезпечує, що дискримінатор тренується на дуже різноманітному наборі даних, що допомагає уникнути перенавчання та підвищити якість генерації зображень.

Завдяки цим особливостям, дискримінатор D у StyleGAN2-ADA ефективно адаптується до різноманітних умов, що покращує здатність моделі генерувати високоякісні зображення, навіть коли використовується відносно малий обсяг тренувальних даних. Висока ймовірність аугментації забезпечує, що дискримінатор навчається розпізнавати реалістичні зображення навіть серед різноманітно змінених варіантів, що значно підвищує загальну ефективність моделі.

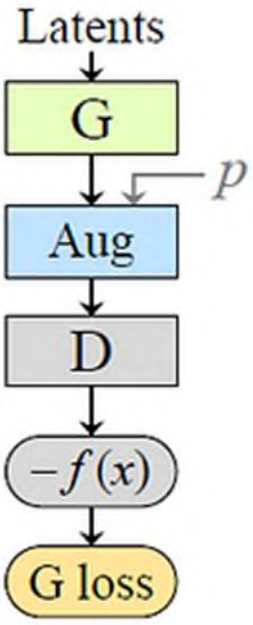

Рисунок 2.6. - Навчання генератора

Під час тренування моделі StyleGAN2-ADA, згенеровані генератором G зображення піддаються аугментації перед їх оцінкою дискримінатором D. Важливим аспектом цього процесу є те, що операції аугментації (Aug) відбуваються після того, як генератор створив зображення. Це означає, що

генератор фокусується виключно на створенні чистих, незмінених зображень, тоді як уся варіативність і різноманітність вносяться на етапі аугментації, що відбувається після генерації.

У ході експериментів з моделлю StyleGAN2-ADA було виявлено, що наступні три типи аугментацій є найбільш ефективними для досягнення оптимальних результатів:

1. Викреслювання пікселів: Ця категорія аугментацій включає перевертання зображень по осях, повороти на 90 градусів, а також цілочисельні переведення. Такі зміни додають варіативність у просторовому розташуванні та орієнтації об'єктів на зображенні, змушуючи дискримінатор D краще адаптуватися та розпізнавати суттєві характеристики зображень.

2. Геометричні аугментації: Ці аугментації включають зміни форми та розміру об'єктів на зображенні, наприклад, масштабування або зміни перспективи. Геометричні зміни сприяють розвитку здатності дискримінатора розрізняти реалістичні зображення з різними геометричними характеристиками.

3. Кольорові перетворення: Ці аугментації включають зміни кольорової палітри, яскравості, контрастності та насиченості зображення. Такі кольорові модифікації допомагають дискримінатору вчитися розпізнавати реалістичні зображення незалежно від їхніх кольорових особливостей.

Ці аугментації, застосовані після генерації зображень генератором G і перед їх оцінкою дискримінатором D, забезпечують, що дискримінатор отримує різноманітні дані для аналізу, що важливо для запобігання його перенавчанню. Такий підхід дозволяє StyleGAN2-ADA ефективно працювати з обмеженими датасетами, підвищуючи якість та реалістичність генерованих зображень.

У попередньому налаштуванні доповнення стохастичного дискримінатора ми використовували однакове значення p для всіх перетворень. Однак хотілось б уникнути ручного налаштування сили доповнення p і натомість контролювати її динамічно на основі ступеня перенавчання.

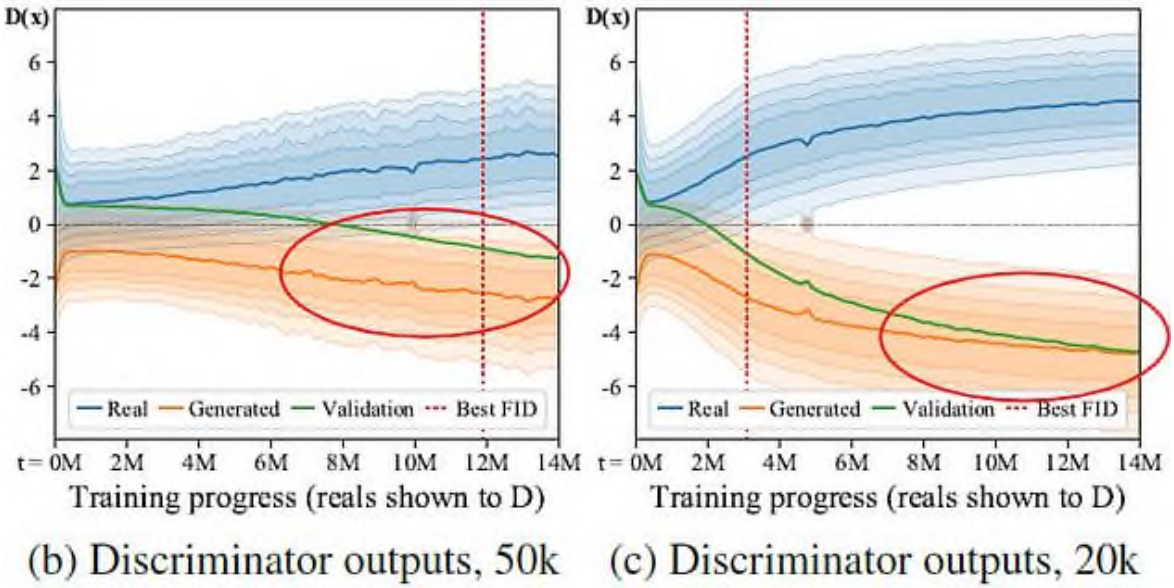

Рисунок 2.7. - Виходи дискримінатора на 50k та 20k навчальних даних.

Коли D_validation наближається до D_generated, відбувається перенавчання.

Виходи дискримінатора на 50k та 20k навчальних даних. Коли D_validation наближається до D_generated, відбувається перенавчання

Позначимо D_train, D_validation, D_generated і D_real як виходи дискримінатора для навчального набору, валідаційного набору, згенерованих зображень і тестового набору відповідно.

Дослідники розробили наступні евристики для кількісної оцінки перенавчання.

Require validation set

$$r_v = \frac{\mathbb{E}[D_{\text{train}}] - \mathbb{E}[D_{\text{validation}}]}{\mathbb{E}[D_{\text{train}}] - \mathbb{E}[D_{\text{generated}}]}$$

Not require validation set

$$r_t = \mathbb{E}[\text{sign}(D_{\text{train}})]$$

Рисунок 2.8. - Евристика надмірної пристосованості

Евристичне правило r_v Оскільки навчальна та перевірочна множини є просто розділеними множинами з реальних зображень, то в ідеалі D_validation має бути ближчою до D_real/D_train. Однак, коли вона перенастроюється, D_validation (зелена) наближається до D_generated (помаранчева).

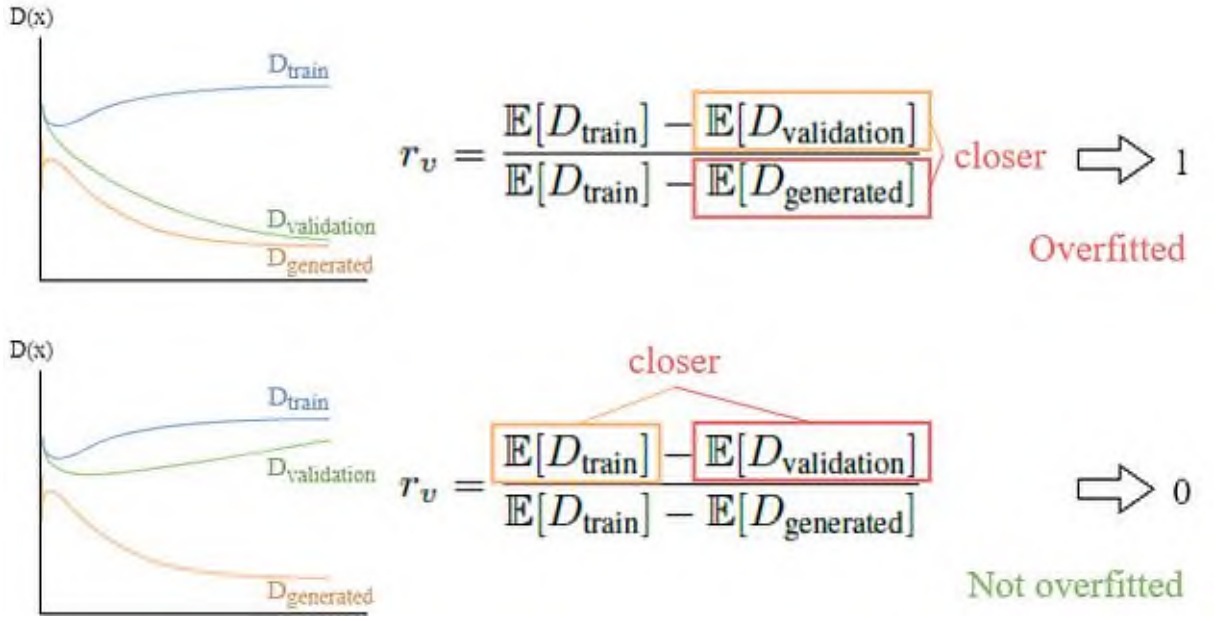

Рисунок 2.9. - Перше евристичне правило

Недоліком цієї евристики є те, що вона вимагає перевірочного набору даних. Оскільки у нас і так мало даних, ми не хочемо ще більше розбивати набір даних. Тому ми включаємо його в основному як метод порівняння.

Евристичне правило r_t. На тренінгу ми будемо використовувати другу евристику r_t. D_real/D_train та D_generated симетрично розходяться навколо нуля, коли ситуація погіршується (перенавчання).

$$r_t = \mathbb{E}[\text{sign}(D_{\text{train}})]$$

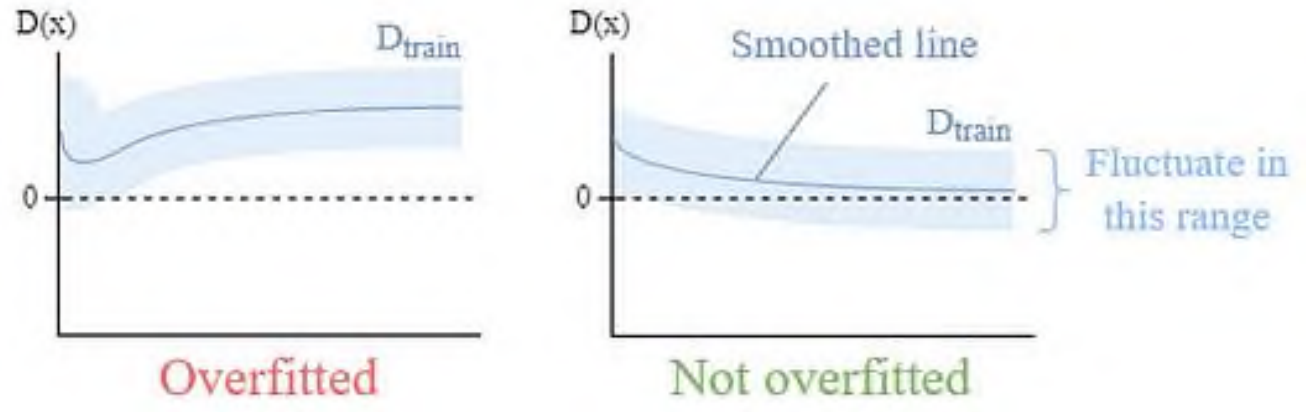

Рисунок 2.10. - Друге евристичне правило

r_t оцінює частину D_train, що набуває додатного значення. Якщо D_train відхиляється від нуля, то це перенавчання, інакше це не перенавчання.

Відрегулюємо p за допомогою r_t

1. Ініціалізуємо p = 0
2. Регулюйте p кожні 4 міні-партії, виходячи з наступних умов:

- Якщо r_t велике → надмірна підгонка → збільште p (додайте більше)
- Якщо r_t низький → не надмірно → зменшити p (зменшити дозу)

Далі розглянемо деякі результати навчання StyleGAN2-ADA.

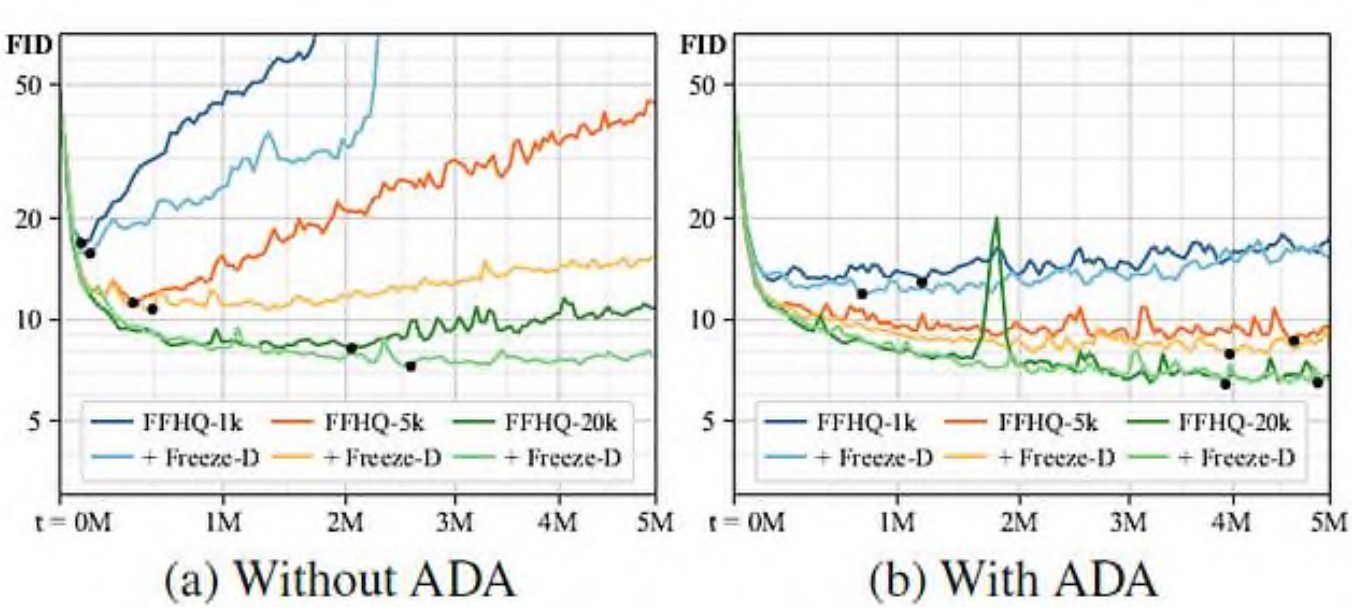

Рисунок 2.11. - Показники FID без використання ADA та з використанням ADA (чим нижче, тим краще)

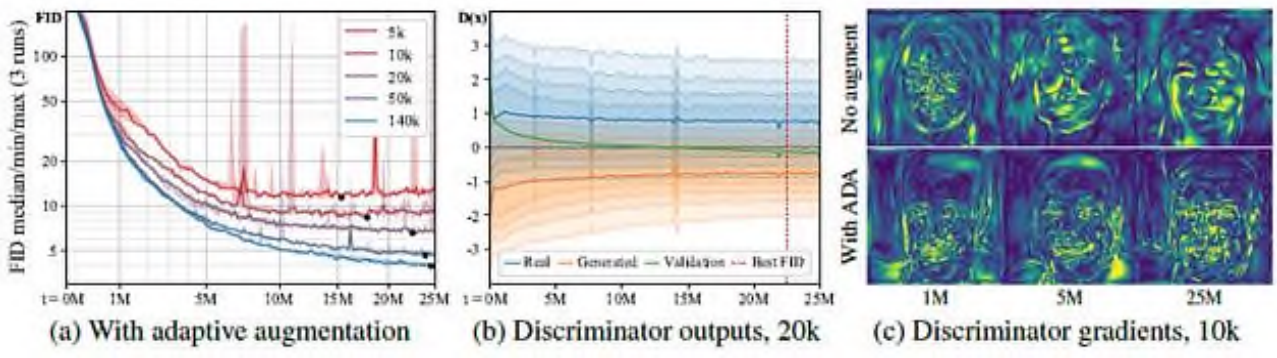

Рисунок 2.12. – Збіжність навчального процесу

Збіжність тепер досягається незалежно від розміру навчальної вибірки, а перенавчання більше не відбувається Навчальні криві для FFHQ з різними розмірами навчальної вибірки з використанням адаптивного доповнення. Опори

реальних згенерованих зображень продовжують перекриватися.Приклади величин градієнтів, які генератор отримує від дискримінатора під час навчання.

2.3. Вибір набору даних для навчання

Вибір набору даних для тренування StyleGAN2-ADA є вирішальним кроком, який значно впливає на ефективність та якість кінцевих результатів. В контексті генеративних змагальних мереж, і зокрема StyleGAN2-ADA, якість та різноманітність набору даних безпосередньо визначають, наскільки реалістичними та переконливими будуть згенеровані зображення. Ось чому важливо розуміти та враховувати ключові фактори при виборі відповідних даних для тренування.

Можна виділити кілька ключових аспектів для вибору набору даних а саме це різноманітність, якість та баланс з репрезентативністю. Далі розглянемо їх детальніше.

Різноманітність Зображень: Один з основних аспектів ефективного набору даних - це його різноманітність. StyleGAN2-ADA використовує аугментації для боротьби з перенавчанням, і наявність широкого спектру зображень різних стилів, сценаріїв та контекстів забезпечує більш ефективне навчання та уникнення упереджень.

Якість Даних: Висока якість зображень у датасеті є критичною. Чіткі, деталізовані зображення дозволяють генератору краще "вчитися" та відтворювати реалістичні деталі та текстури в генерованих зображеннях.

Баланс та Репрезентативність: Датасет повинен бути збалансованим та репрезентативним, щоб уникнути ситуації, коли модель буде перекошена в бік певного типу зображень. Збалансовані датасети сприяють більшій загальній здатності моделі до генералізації.

Правильний вибір датасету для тренування StyleGAN2-ADA має безпосередній вплив на кінцеву якість згенерованих зображень. Недостатньо різноманітний або низькоякісний датасет може призвести до зниження реалістичності, виникнення артефактів, або обмеженого діапазону зображень, які

здатна створювати модель. З іншого боку, добре підібраний датасет, що відповідає зазначеним вище критеріям, значно підвищує шанси на успішне тренування моделі, що здатна генерувати високоякісні, різноманітні та реалістичні зображення.

Різноманітність даних є одним з ключових елементів, що забезпечують успіх тренування генеративних змагальних мереж, таких як StyleGAN2-ADA. Ця важливість випливає з кількох основних причин:

- Різноманітність Зображень

Підвищення Здатності до Генералізації: Включення широкого спектру зображень у тренувальний датасет забезпечує, що модель зможе вчитися на різноманітних прикладах та покращувати свої загальні здібності до генерації зображень. Це допомагає уникнути упереджень у тренуванні та розвиває здатність моделі до генерації більш різноманітних і реалістичних результатів.

Уникнення Перенавчання: Різноманітність у датасеті знижує ризик перенавчання, оскільки модель не буде обмежена невеликою кількістю повторюваних патернів або характеристик.

- Вплив Обмежених Датасетів

Проблема Обмеженої Інформації: Навчання на обмежених датасетах може призвести до того, що модель не зможе відтворювати широкий діапазон варіацій у реальному світі. Це може обмежити здатність моделі розпізнавати та генерувати реалістичні зображення.

Вирішення Проблеми в StyleGAN2-ADA: StyleGAN2-ADA впроваджує адаптивну аугментацію, що дозволяє ефективно використовувати обмежені датасети, забезпечуючи різноманітність у тренувальних даних. Це досягається шляхом зміни зображень, що входять у датасет, збільшуючи їх різноманітність та уникнувши перенавчання.

- Якість та Розмір Зображень

Висока Роздільна Здатність: Зображення високої роздільної здатності надають більше деталей та текстур, що є необхідним для генерації реалістичних та деталізованих візуальних результатів. StyleGAN2-ADA оптимізована для роботи з детальними зображеннями, тому вибір датасету з високоякісними

зображеннями є важливим. Чіткість Зображень: Чіткі та ясні зображення сприяють кращому навчанню, оскільки забезпечують чіткі та недвозначні приклади для моделі.

Тепер давайте обговоримо детальніше джерела наборів даних для навчання нейронних мереж. Є публічні набори даних та створи власний набір даних. Коротко про ці два способи.

Публічні Датасети: Існує багато відкритих джерел даних, таких як ImageNet, CIFAR та інші, які пропонують різноманітні зображення, які можна використовувати для тренування. Перевага публічних датасетів полягає в їхній широкій доступності та часто великому обсязі даних.

Збір Власних Даних: Можна також зібрати унікальний датасет, використовуючи власні або спеціалізовані джерела. Це може бути особливо корисним, якщо ви хочете тренувати модель на специфічних типах зображень, яких немає у загальнодоступних дата сетах.

Проаналізувавши критерії та особливості наборів даних можна приступити до пошуку набору даних. Давайте розглянемо кілька наборів даних які нам підходять по тематиці. Наш погляд впав на такі набори даних як Danbooru2020, e621, Derpibooru та Safebooru. Так тепер розглянемо кожен набір даних детальніше.

Danbooru2020 є великомасштабною базою даних зображень в стилі аніме, яка містить понад 4.2 мільйона зображень, анотованих понад 130 мільйонами текстових тегів, що детально описують зміст зображень. Цей датасет може бути корисним для різноманітних цілей машинного навчання, включаючи розпізнавання зображень та генерацію. Danbooru20xx щорічно оновлюється зображеннями та поліпшеннями метаданих попередніх років. Попередні версії включають Danbooru2017, Danbooru2018 та Danbooru2019. Така широка колекція робить Danbooru2020 цінним ресурсом для тренування генеративних моделей, як-от StyleGAN2-ADA

Рисунок 2.13. - Зразок з набору даних Danbooru2020

Набір даних з e621.net складається з приблизно 55 000 зображень SFW (Safe For Work). Ця колекція може бути цінним ресурсом для навчання генеративних моделей, таких як StyleGAN2-ADA, особливо в контекстах, де основна увага приділяється різноманітному і безпечному для роботи контенту. Специфічні характеристики набору даних, такі як якість зображень, різноманітність і діапазон охоплених тем, відіграватимуть вирішальну роль у його ефективності для навчальних цілей.

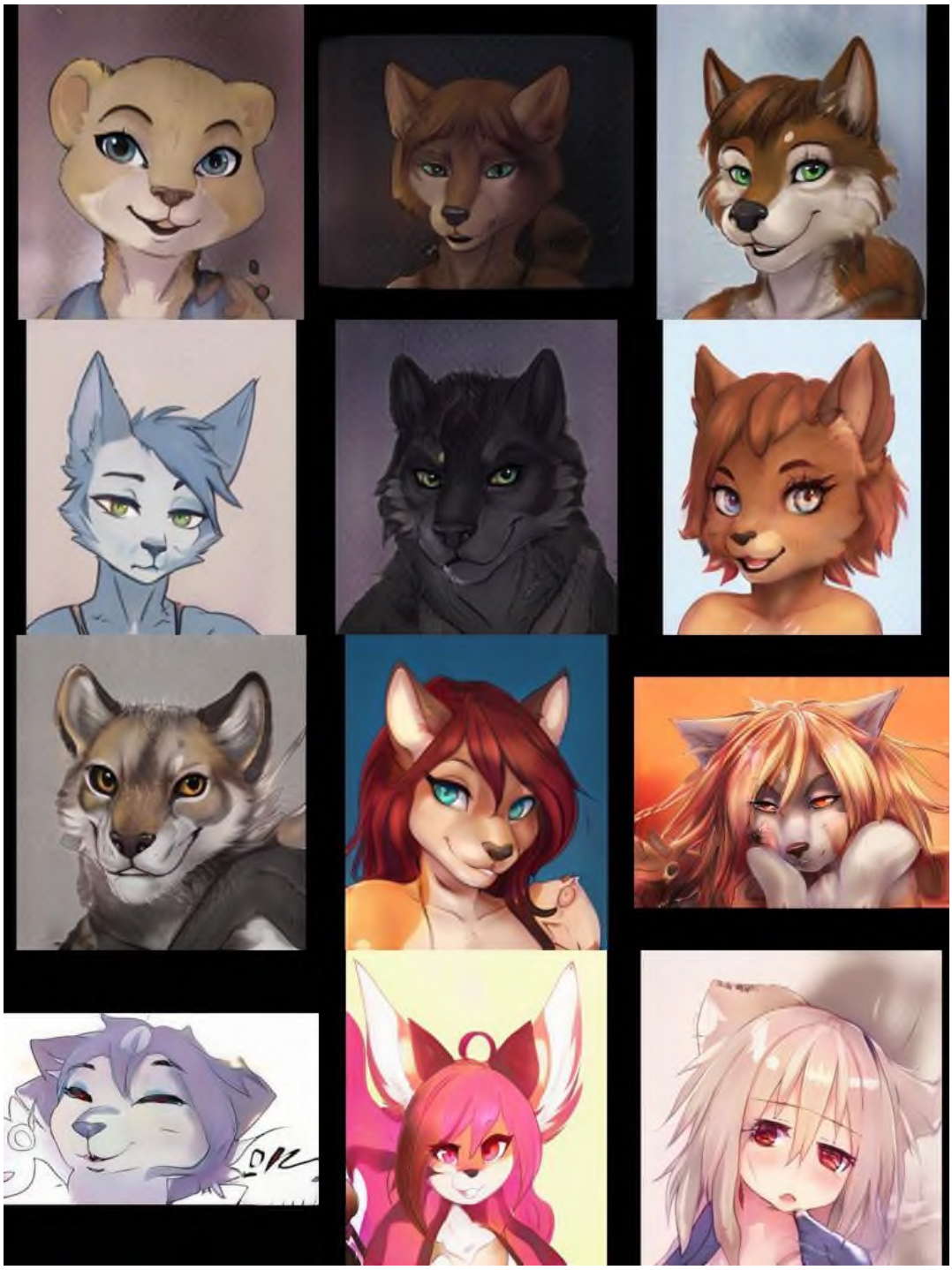

Рисунок 2.14. - Зразок з набору даних e621.net

Derpibooru - це всеосяжне сховище зображень, яке фокусується на контенті, пов'язаному з "My Little Pony: Friendship is Magic". Він пропонує величезну колекцію зображень, що робить його потенційно багатим набором даних для різних програм машинного навчання, зокрема для навчання StyleGAN2-ADA. Однак, при визначенні придатності Derpibooru для конкретного проєкту важливо враховувати специфічний фокус і тематичний контент, який він містить. Сильна

тематична спрямованість набору даних може обмежити його застосовність, якщо потрібен більш різноманітний або інший тематичний набір даних.

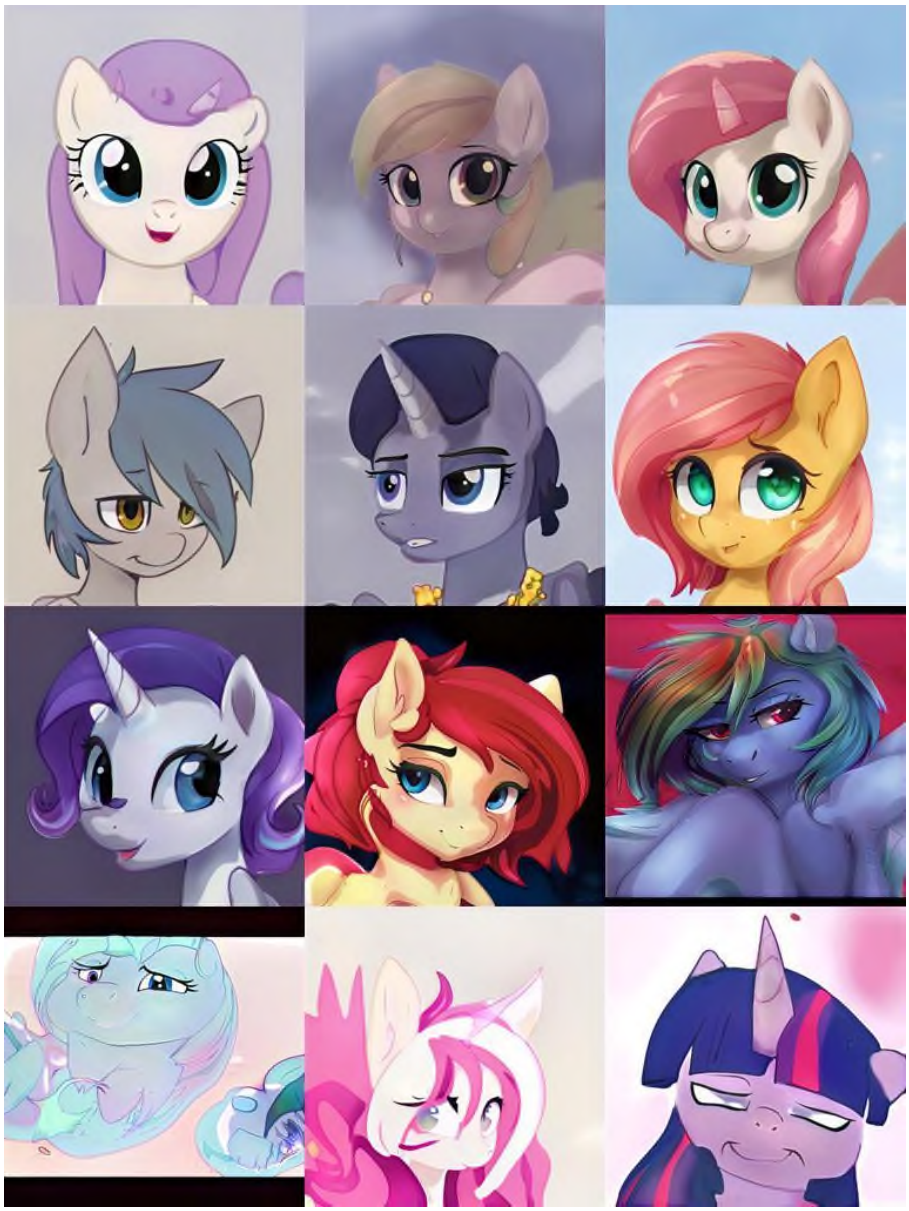

Рисунок 2.15. - Зразок з набору даних Derpibooru

Safebooru - це набір даних зображень аніме, який містить значну колекцію ілюстрацій у стилі аніме. Хоча в джерелах, до яких я звертався, не було конкретної інформації про точну кількість зображень та їхні характеристики в наборі даних Safebooru, його вважають підмножиною великого набору даних Danbooru, який відомий своєю масштабною та різноманітною колекцією аніме-ілюстрацій. Safebooru, зосереджуючись на безпечному для роботи (SFW)

контенті, пропонує різноманітні аніме-зображення, придатні для різних програм машинного навчання, включаючи навчальні моделі на кшталт StyleGAN2-ADA.

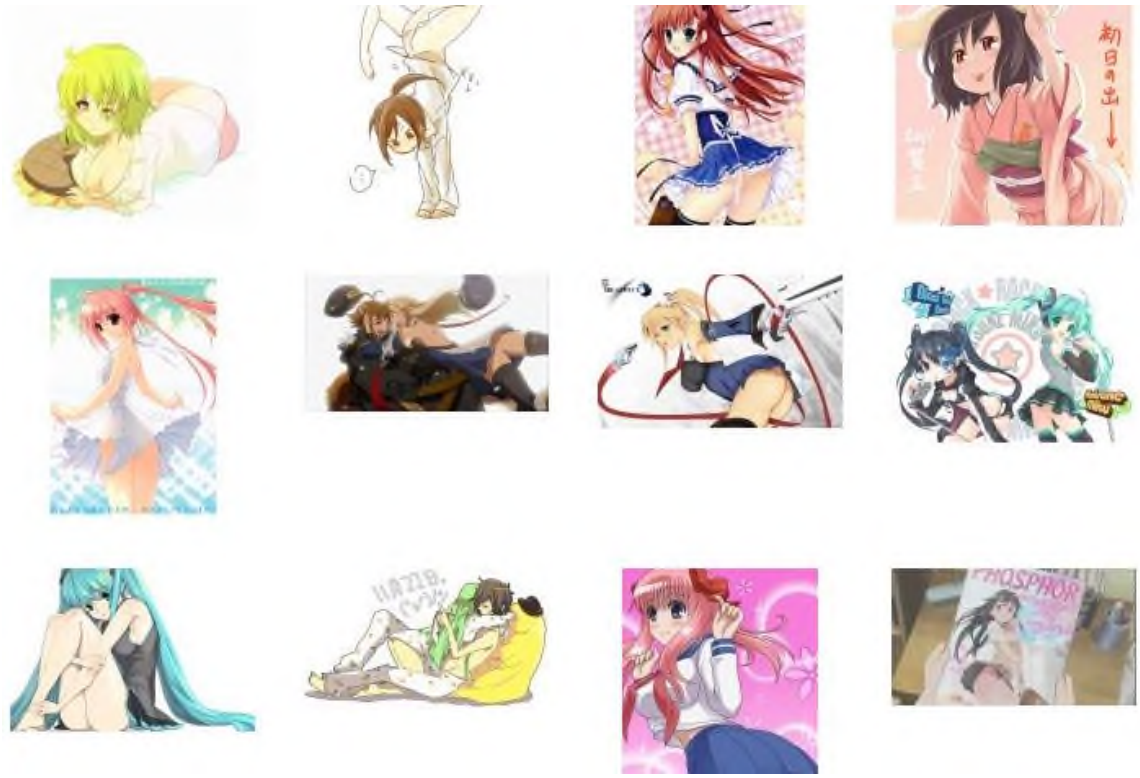

Рисунок 2.16. - Зразок з набору даних Safebooru

Набір даних Flickr-Faces-HQ (FFHQ) - це високоякісний набір зображень, що містить 70 000 зображень облич з високою роздільною здатністю. Завдяки значним варіаціям віку, етнічної приналежності, походження та аксесуарів, FFHQ надає різноманітний спектр зображень людських облич. Це робить його чудовим ресурсом для навчання сучасних генеративних моделей, таких як StyleGAN2-ADA, особливо для задач, що фокусуються на рисах людського обличчя. Зображення з високою роздільною здатністю особливо корисні для створення детальних і реалістичних результатів у різних програмах машинного навчання та комп'ютерного зору.

Рисунок 2.17. - Зразок з набору даних Flickr-Faces-HQ

Важливо зазначити, що для ефективного навчання обраний набір даних має бути не лише великим за обсягом, але й різноманітним за типами зображень, які в ньому містяться. Така різноманітність допомагає в навчанні надійної моделі, здатної генерувати широкий спектр зображень у стилі коміксів. Крім того, використання передових стратегій доповнення даних, таких як ScoreMix, може ще більше підвищити продуктивність моделі, особливо при роботі з обмеженими даними. Цей метод генерує доповнені вибірки, які урізноманітнюють навчальні дані та допомагають зменшити проблеми перенавчання.

Висновок до розділу 2

1. Висвітлюється різноманіття архітектур GAN, що використовуються для генерації зображень, кожна з яких має свої унікальні характеристики та призначення. Deep Convolutional GAN (DCGAN) відзначається простотою та ефективністю навчання, WGAN пропонує краще визначення відстані між розподілами за допомогою відстані Вассерштейна, а StyleGAN та його наступні версії (StyleGAN2, StyleGAN2-ADA) вносять інновації у візуалізацію та контроль над згенерованими зображеннями. Кожна модель має свої переваги, залежно від конкретних цілей та доступних обчислювальних ресурсів, що

відкриває широкі можливості для їх застосування в генерації коміксів і інших областях.

2. Архітектура StyleGAN2-ADA представляє собою значне вдосконалення у сфері генеративних змагальних мереж, зокрема у контексті тренування на невеликих датасетах. Ця модель вводить адаптивну аугментацію для дискримінатора, що дозволяє збалансувати тренування і запобігти перенавчанню, важливій проблемі, яка часто виникає при роботі з обмеженими даними. Аугментації, що динамічно регулюються залежно від стану перенавчання дискримінатора, забезпечують більшу різноманітність та якість згенерованих зображень. Крім того, StyleGAN2-ADA здатний вирішувати проблеми з артефактами, притаманні попереднім версіям StyleGAN, поліпшуючи якість зображень і забезпечуючи ефективніше використання обчислювальних ресурсів. Важливою особливістю цієї моделі є її гнучкість та здатність адаптуватися до різних умов навчання, роблячи її ідеальним інструментом для широкого спектра застосувань у генерації зображень.

3. Проведено вибір набору даних для тренування StyleGAN2-ADA є критичним елементом, який значно впливає на якість та реалістичність згенерованих зображень. Різноманітність, якість і баланс в датасеті є ключовими факторами, що забезпечують успішне навчання і здатність моделі до генералізації. Використання деталізованих та репрезентативних даних, таких як Danbooru2020, e621, Derpibooru, Safebooru та FFHQ, сприяє розробці здатності моделі генерувати різноманітні та якісні візуальні вихідні дані. Цей етап вибору є вирішальним для забезпечення того, що кінцевий продукт буде не тільки технічно продуктивним, але й візуально привабливим і різноманітним, здатним відтворювати широкий спектр стилів і тем.

3. РЕАЛІЗАЦІЯ І АНАЛІЗ РЕЗУЛЬТАТІВ

3.1. Розробка та реалізація моделі

Для розробки та реалізації моделі було обрано Google Colab. Google Colab, також відомий як Colaboratory, є інноваційним хмарним сервісом від Google, спеціально розробленим для полегшення роботи в області машинного навчання та штучного інтелекту. Ця платформа заснована на концепції Jupyter Notebooks і дозволяє користувачам писати та виконувати код Python безпосередньо у веб-браузері, що робить її надзвичайно зручною та доступною для широкого кола користувачів. Важливою особливістю Google Colab є його здатність надавати користувачам доступ до високопродуктивних обчислювальних ресурсів, таких як GPU та TPU, безкоштовно, що є значною перевагою для виконання обчислювально вимогливих задач.

Однією з ключових переваг Google Colab є його спрощена співпраця та інтеграція. Користувачі можуть легко ділитися своїми блокнотами з іншими, а також працювати над ними спільно в режимі реального часу, подібно до Google Docs. Це робить Colab ідеальним інструментом для спільних наукових досліджень та освітніх проектів. Крім того, інтеграція з Google Drive спрощує зберігання, доступ та управління робочими файлами.

Платформа також активно використовується для навчання та тестування моделей машинного навчання. Завдяки наявності вже встановлених бібліотек для машинного навчання та аналізу даних, таких як NumPy, Pandas, Matplotlib, користувачі можуть швидко розпочати роботу над своїми проектами.

Однак, необхідно зазначити, що попри численні переваги, Google Colab має певні обмеження, такі як залежність від стабільного інтернет-з'єднання та обмежений час сесій у безкоштовній версії. Крім того, питання конфіденційності даних та безпеки коду варто розглядати серйозно, оскільки всі блокноти зберігаються на серверах Google.

Незважаючи на ці обмеження, Google Colab залишається потужним інструментом для дослідників, студентів та розробників, що працюють у сфері

машинного навчання та штучного інтелекту, та може бути надзвичайно корисним ресурсом для магістерської роботи.

Рисунок 3.1. - Середовище Google Colab

Для реалізації методу генерування коміксів використав код, офіціно наданий з репозиторію git компанії .Nvidia. Це офіційна реалізація PyTorch

Функція втрат для D еквівалентна змагальним втратам в оригінальній ро боті GAN, з додатковою регуляризацією R_1.

Було обрано почати зі значення Y, рівного 50, тому що так рекомендують. Однак цей параметр слід точно налаштувати методом спроб і помилок. Більше значення Y робить модель більш стабільною, що є корисним у випадку колапсу режиму. Але тоді результати будуть менш різноманітними, тому згенеровані зображення не покриватимуть різноманіття реального набору даних. оскільки підібрані набори даних характеризуються великою різноманітністю, це буде помітно. невелике значення *y* дає більше різноманітності, але може зробити модель нестійкою тому в ідеалі тренування повторюється багато разів, зі збільшеними або зменшеними значеннями *Y* щоб знайти оптимальні результати.

Функція втрат для *G* є ненасичуваною логістичною втратою *f(x)=log(sigmoid(x))* з регуляризацією довжини шляху.оскільки функції втрат суттєво не змінилися порівняно з DCGAN, еволюція цих параметрів під час навчання також нічого не говорить нам про розбіжність між реальним і фальшивим розподілом. однак реалізація PyTorch дозволяє мати уявлення про ступінь збіжності завдяки показнику FID (Frechet Inception Distance), введеному. FID - це метод вимірювання якості згенерованих зразків зображень. він робить це , порівнюючи статистику згенерованих зразків з реальними зразками на рівні

характеристик зображення. Тому він використовує вихід останнього шару об'єднання попередньо навченої моделі Inception v3 для 50 тис. згенерованих зображень і всіх зображень у навчальному наборі даних. обчислення цієї метрики займає значну кількість часу, але зазвичай вона підтверджує те, що ви спостерігаєте в послідовних результатах, а саме, що модель більше не еволюціонує після достатньої кількості ітерацій. Саме тоді досягається збіжність.

Оптимізатор є адамівським оптимізатором з $\beta_1=0$, $\beta_2=0.99$ та $\epsilon=10^{-8}$.

3.2. Навчання моделі

StyleGAN2 є настільки складною мережею, що навчання з нуля займає неймовірно багато часу. У статті повідомляється []: "Загальний час навчання склав 9 днів для FFHQ і 13 днів для LSUN CAR". Це на NVIDIA DGX-1 з 8 графічними процесорами Tesla V100. Для цієї роботи було використано тільки можливості тільки безплатної версії Google Colab детальніше описано в попередньому підрозділі.

Тому доречно почати з попередньо навченої моделі. на основі початкового тестування на попередньо навченій на зображеннях My Little Pony, оскільки стиль малювання був найближчим до того що потрібно було генерувати а саме фрейми коміксів. Навчання до повної збіжності зайняло близько тижня при роздільній здатності 1282. Другим набір даних, це фотографії персонажів Kinky & Cosy.

Еволюція оцінок D-втрат, G-втрат, D для реальних зображень і FID показана для обох наборів даних на рисунках 3.2 і 3.3. обидва експерименти призводять до зменшення значення FID. Хоча загальний набір даних досягає збіжності FID швидше (після 1 500 kimg), ніж набір даних, сфокусований на Kinky & Cosy (після 4 000 kimg).

Перший набір даних збігається при FID = 46, а другий - при FID = 25. Ця різниця, безсумнівно, є результатом більшої внутрішньої різноманітності першого набору даних порівняно з другим. Також можна відзначити, що оцінка D для зображень з навчального набору демонструє зростаючу еволюцію.

незважаючи на вплив адаптивного доповнення дискримінатора, легкої форми перенавчання все одно не уникнути. Навчання

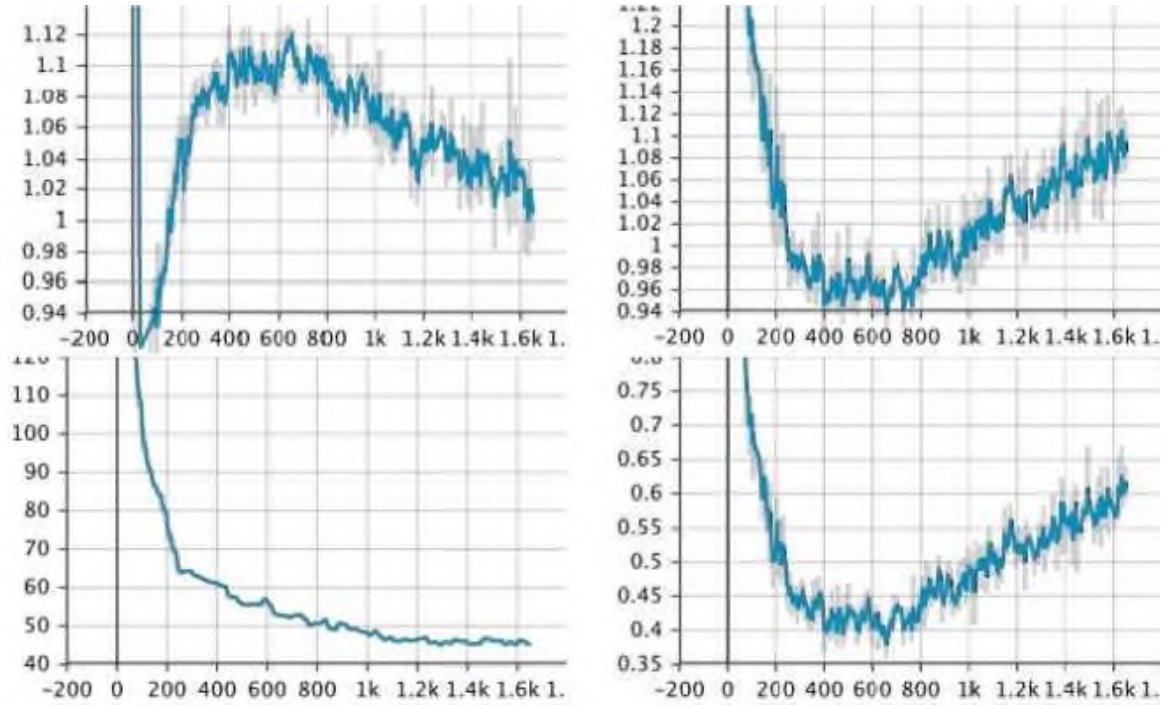

Рисунок 3.2. - За годинниковою стрілкою: D-втрати, G-втрати, оцінка D для реальних зображень і FID для набору загальних зображень.

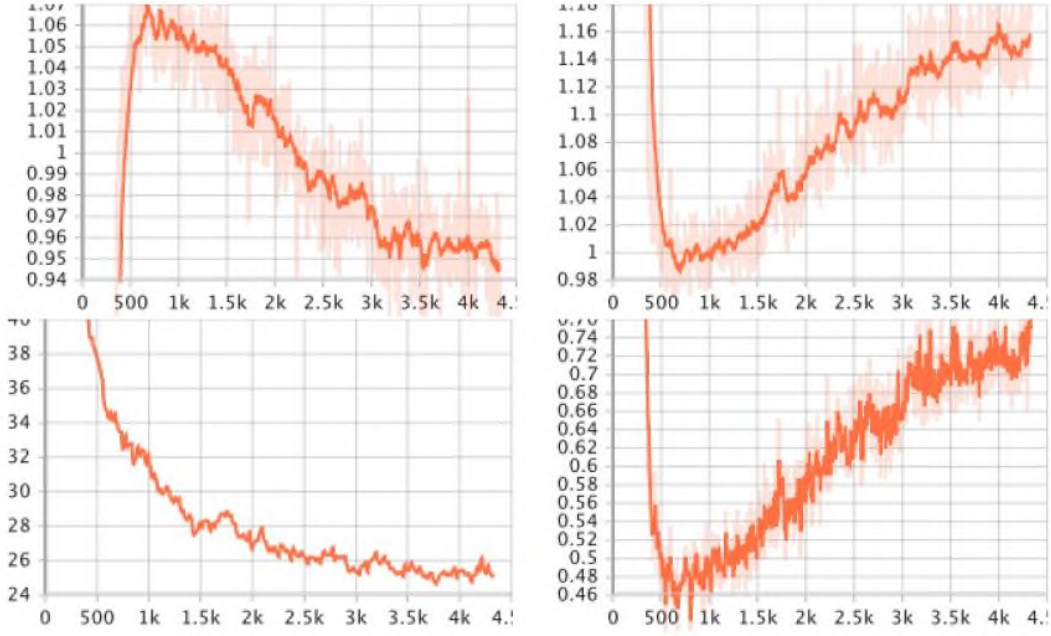

Рисунок 3.3. - За годинниковою стрілкою: D-втрати, G-втрати, D-бали для реальних зображень та FID для набору зображень,
сфокусованих на Kinky & Cosy.

Можна припустити, що набір даних з ще меншим розмаїттям міг би досягти нижчого показника FID. Скільки саме часу займає навчання, підлягає подальшому дослідженню. Як зазначено на веб-сайті []: "У типових випадках для досягнення конвергенції потрібно 25 000 kimg або більше, але результати вже є цілком прийнятними в межах 5.000 kimg. 1.000 kimg кілограмів часто достатньо для трансферного навчання, яке має тенденцію до зближення значно швидше". Для візуального покращення результатів бажано було б використати менш різноманітний набір даних.

3.3. Оцінка моделі

На рисунку 3.4 показано результати першого набору даних. найяскравіше враження - це величезне покращення порівняно з попередньо протестованими моделями. StyleGAN2 чітко "зрозумів" стилістичні особливості коміксів Kinky & Cosy. незважаючи на обмежену роздільну здатність, навколо символів є чітка чорна лінія, а кольори, як правило, рівні на окреслених поверхнях, як і повинно бути, за винятком деяких артефактів у вигляді коротких горизонтальних чорних смуг. крім того, кольори правильні і не надто бліді, як це було раніше.

Рисунок 3.4. - Згенеровані зображення за допомогою StyleGAN2-ADA, навченого на загальних обрізаних фотографіях (розмір 128 × 128 пікселів)

Однак головним недоліком є те, що жоден з малюнків не є правильним. Персонажі спотворені, так само як і тло. кінцівки відсутні, голови де- дубльовані, персонажі іноді мають більше двох очей, і що найбільш вражає, кінкі та козі, очевидно, є трійнею, а не близнюками див. рисунок 3.4 зображення в четвертому рядку та третьому стовпчику.

На рисунку 3.5. показано синтезовані зображення після навчання на другому наборі даних. Ці результати ще кращі, що дає надію на досягнення мети дослідження: створення придатних для використання малюнків на основі

існуючих коміксів. На попередньому наборі даних були отримані зображення, на яких можна було розрізнити персонажів. У цьому випадку їх не лише можна розрізнити, але й ідентифікувати. У деяких випадках вони бездоганні. Кольори правильні і добре вирівняні в межах окреслених областей, а чорний контур чіткий. Модель передала концепцію, що ці дві дівчини ідентичні, за винятком окулярів.

Рисунок 3.5. - Згенеровані зображення за допомогою StyleGAN2-ADA, навченого на зображеннях з фокусом на Kinky & Cosy (розмір 128 × 128 пікселів)

Показник FID (Frechet Inception Distance) у даній моделі виявився кращим – 25, у порівнянні з 100 та 89,2, які показали аналогічні моделі. Проте, як зазначено в дослідженні Lucic et al. у 2017 році, пряме порівняння показників FID

між моделями не дає повної картини їхньої ефективності. Автори пропонують замість цього аналізувати розподіли мінімально досягнутих значень FID в межах фіксованого обчислювального бюджету. Такий підхід показує, що алгоритмічні відмінності в сучасних GAN (Generative Adversarial Networks) стають менш важливими, коли збільшується обчислювальний бюджет. Але при обмеженому бюджеті, наприклад, з обмеженням в один місяць обчислювального часу, "кращий" алгоритм може поступитися "гіршому".

Недоліком, однак, є те, що в 95% випадків малюнки все ще містять помилки. Деякі кінцівки відсутні, обличчя іноді деформовані, а тло подекуди зливається з персонажами. Отже, як такі, ці малюнки ще не підходять для публікації в книзі чи газеті.

Основним недоліком є недостатня різноманітність у згенерованих зображеннях, яка відрізняється від різноманітності в датасеті. Для порівняння, візьмемо StyleGAN2, навчений на FFHQ (згадано в дослідженні Karras et al., 2019), який зміг створювати реалістичні зображення облич, відрізняючись від реальних фотографій. В даному експерименті, однак, StyleGAN2 не зміг відтворити настільки ж різноманітні сцени. Наприклад, у датасеті були малюнки Kinky & Cosy верхи на верблюді в пустелі та Kinky & Cosy на велосипеді в місті. Однак, зображення Kinky & Cosy на велосипеді в пустелі, яке б було цікавим новим синтезом, не було створено. Це стає очевиднішим при генерації інтерполяцій у латентному просторі. Можливі причини включають перенастроювання дискримінатора та надмірну регуляризацію генератора. Рішенням для першої проблеми може бути збільшення аугментації, а для другої - зменшення значень R1.

Іншим недоліком є надмірна кількість пішохідних п та велосипедів. Причиною, як згадувалося раніше, є слабкість набору даних. Ці анімовані цикли створюють велику концентрацію майже ідентичних зображень. Як наслідок, розподіл даних демонструє занадто багато піків у кількох обмежених місцях. Вирішення цієї проблеми полягає в прагненні до нормального розподілу даних. Цього можна досягти, очистивши набір даних так, щоб у ньому з'являлися лише абсолютно різні зображення.

Інша проблема полягає у надлишковій кількості зображень пішоходів та велосипедів у наборі даних. Як зазначалося раніше, це є слабкістю самого набору даних. Через ці повторювані анімаційні цикли у наборі з'являється велика кількість майже ідентичних зображень, що призводить до концентрації даних у декількох обмежених областях і створює нерівномірний розподіл з вираженими піками. Щоб вирішити цю проблему, потрібно прагнути до більш рівномірного, нормального розподілу даних. Це можна зробити шляхом очищення набору даних таким чином, щоб у ньому залишилися тільки унікальні, різноманітні зображення, зменшуючи кількість повторень.

Висновки до розділу 3

1. У розділі 3 було представлено детальний опис процесу розробки та реалізації моделі за допомогою Google Colab, використовуючи код від Nvidia та PyTorch. Аналіз результатів тренування моделі StyleGAN2-ADA виявив значне поліпшення у генерації зображень порівняно з попередніми моделями. Зокрема, модель демонструє високу здатність до відтворення стилістичних особливостей коміксів, з високою чіткістю кольорів та ліній.

2. Значення FID, як показник якості зображень, підтверджує ефективність моделі, особливо при навчанні на спеціалізованих датасетах. Проте, виявлені проблеми, такі як спотворення персонажів та обмеження у різноманітності генерованих сцен, вказують на необхідність подальшого удосконалення моделі та оптимізації навчальних датасетів.

3. Результати показують, що при належній настройці та з врахуванням особливостей наборів даних, StyleGAN2-ADA має потенціал для створення високоякісних зображень коміксів, що може бути корисним у майбутніх дослідженнях та застосуваннях у цій галузі.

ВИСНОВКИ

В рамках магістерської роботи «Метод генерації коміксів за допомогою генеративних змагальних мереж» було отримано наступні висновки:

1. В підрозділі 1.1. демонструє, як GAN стали значущою частиною сучасних досліджень у галузі штучного інтелекту. Від їх винайдення у 2014 році, GAN зазнали значного розвитку та вдосконалення, подолавши ранні проблеми, такі як "mode collapse". Їх здатність генерувати реалістичні зображення і застосування в різноманітних сферах, включаючи мистецтво та медичні зображення, відкриває нові перспективи для досліджень і інновацій. Порівняння з дискримінативними моделями і детальний огляд різних підходів до генеративного моделювання подає зрозуміле уявлення про унікальність та потенціал GAN.

2. Також було розкрито важливість коміксів як культурного та художнього феномену у різних країнах. Вона підкреслює їхню трансформацію з "дитячої літератури" до визнаної форми графічного вираження, що вміло поєднує зображення та текст для передачі глибоких ідей та емоцій. Економічний аспект, особливо зосередження на манзі в Японії, вказує на величезний потенціал і вплив цієї індустрії, що продовжує розвиватися у цифрову еру.

3. Було розкрито інноваційні підходи до створення коміксів за допомогою технологій, зокрема GAN. Вона охоплює широкий спектр аспектів, включаючи векторизацію, колоризацію, створення персонажів, анімацію та адаптацію вмісту. Цей розділ підкреслює потенціал застосування штучного інтелекту для революціонізації способу створення та презентації коміксів, відкриваючи можливості для більшої творчості та інтерактивності.

4. Ввизначено конкретні цілі та завдання для дослідження використання GAN у створенні коміксів. Завдання включають аналіз існуючих технік GAN, підготовку даних, експерименти з генерацією коміксів, та оцінку якості згенерованих зображень. Цілі дослідження відображають прагнення інтегрувати новітні технології в традиційне мистецтво створення коміксів, вказуючи на значний потенціал впливу на індустрію коміксів і творчість загалом.

5. Висвітлюється різноманіття архітектур GAN, що використовуються для генерації зображень, кожна з яких має свої унікальні характеристики та призначення. Deep Convolutional GAN (DCGAN) відзначається простотою та ефективністю навчання, WGAN пропонує краще визначення відстані між розподілами за допомогою відстані Вассерштейна, а StyleGAN та його наступні версії (StyleGAN2, StyleGAN2-ADA) вносять інновації у візуалізацію та контроль над згенерованими зображеннями. Кожна модель має свої переваги, залежно від конкретних цілей та доступних обчислювальних ресурсів, що відкриває широкі можливості для їх застосування в генерації коміксів і інших областях.

6. Архітектура StyleGAN2-ADA представляє собою значне вдосконалення у сфері генеративних змагальних мереж, зокрема у контексті тренування на невеликих датасетах. Ця модель вводить адаптивну аугментацію для дискримінатора, що дозволяє збалансувати тренування і запобігти перенавчанню, важливій проблемі, яка часто виникає при роботі з обмеженими даними. Аугментації, що динамічно регулюються залежно від стану перенавчання дискримінатора, забезпечують більшу різноманітність та якість згенерованих зображень. Крім того, StyleGAN2-ADA здатний вирішувати проблеми з артефактами, притаманні попереднім версіям StyleGAN, поліпшуючи якість зображень і забезпечуючи ефективніше використання обчислювальних ресурсів. Важливою особливістю цієї моделі є її гнучкість та здатність адаптуватися до різних умов навчання, роблячи її ідеальним інструментом для широкого спектра застосувань у генерації зображень.

7. Проведено вибір набору даних для тренування StyleGAN2-ADA є критичним елементом, який значно впливає на якість та реалістичність згенерованих зображень. Різноманітність, якість і баланс в датасеті є ключовими факторами, що забезпечують успішне навчання і здатність моделі до генералізації. Використання деталізованих та репрезентативних даних, таких як Danbooru2020, e621, Derpibooru, Safebooru та FFHQ, сприяє розробці здатності моделі генерувати різноманітні та якісні візуальні вихідні дані. Цей етап вибору є вирішальним для забезпечення того, що кінцевий продукт буде не тільки

технічно продуктивним, але й візуально привабливим і різноманітним, здатним відтворювати широкий спектр стилів і тем.

8. У розділі 3 було представлено детальний опис процесу розробки та реалізації моделі за допомогою Google Colab, використовуючи код від Nvidia та PyTorch. Аналіз результатів тренування моделі StyleGAN2-ADA виявив значне поліпшення у генерації зображень порівняно з попередніми моделями. Зокрема, модель демонструє високу здатність до відтворення стилістичних особливостей коміксів, з високою чіткістю кольорів та ліній.

9. Значення FID, як показник якості зображень, підтверджує ефективність моделі, особливо при навчанні на спеціалізованих датасетах. Проте, виявлені проблеми, такі як спотворення персонажів та обмеження у різноманітності генерованих сцен, вказують на необхідність подальшого удосконалення моделі та оптимізації навчальних датасетів.

10. Результати показують, що при належній настройці та з врахуванням особливостей наборів даних, StyleGAN2-ADA має потенціал для створення високоякісних зображень коміксів, що може бути корисним у майбутніх дослідженнях та застосуваннях у цій галузі.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Augereau, O.; Iwata, M.; Kise, K. An Overview of Comics Research in Computer Science. In Proceedings of the 2017 14th IAPR International Conference on Document Analysis and Recognition (ICDAR), Kyoto, Japan, 9-15 November 2017; pp. 54-59.

2. Lam, P.E. Japan's quest for "soft power": Attraction and limitation. East Asia 2007, 24, 349-363.

3. Hall, I.; Smith, F. The struggle for soft power in Asia: Public diplomacy and regional competition. Asian Secur. 2013, 9, 1-18.

4. Screech, M. Masters of the Ninth Art: Bandes Dessinees and Franco-Belgian Identity; Liverpool University Press: Liverpool, UK, 2005; Volume 3.

5. Christiansen, H.C. Comics & Culture: Analytical and Theoretical Approaches to Comics; Museum Tusculanum Press: K0benhavn, Denmark, 2000.

6. AJPEA. Manga Market in Japan. 2017. Available online: http://www.ajpea.or.jp/information/20170224/ index.html

7. Jain, E.; Sheikh, Y.; Hodgins, J. Inferring artistic intention in comic art through viewer gaze. In Proceedings of the ACM Symposium on Applied Perception, Los Angeles, CA, USA, 3-4 August 2012; pp. 55-62.

8. Cao, Y.; Lau, R.W.; Chan, A.B. Look over here: Attention-directing composition of manga elements. ACM Trans. Graph. 2014, 33, 94.

9. Pederson, K.; Cohn, N. The changing pages of comics: Page layouts across eight decades of American superhero comics. Stud. Comics 2016, 7, 7-28.

10. Fujimoto, A.; Ogawa, T.; Yamamoto, K.; Matsui, Y.; Yamasaki, T.; Aizawa, K. Manga109 dataset and creation of metadata. In Proceedings of the 1st International Workshop on coMics ANalysis, Processing and Understanding, Cancun, Mexico, 4 December 2016; p. 2.

11. Wilber, M.J.; Fang, C.; Jin, H.; Hertzmann, A.; Collomosse, J.; Belongie, S. BAM! the behance artistic media dataset for recognition beyond photography. arXiv 2017, arXiv:1704.08614.

12. Bergs, A. Protanopia, a Revolutionary Digital Comic for Iphone and Ipad. Available online: http:// andrebergs.com/protanopia

13. Ito, K.; Matsui, Y.; Yamasaki, T.; Aizawa, K. Separation of Manga Line Drawings and Screentones. In Proceedings of the Eurographics, Zurich, Switzerland, 4-8 May 2015; pp. 73-76.

14. Arai, K.; Tolle, H. Method for real time text extraction of digital manga comic. Int. J. Image Process. 2011, 4, 669-676.

15. Chu, W.T.; Cheng, W.C. Manga-specific features and latent style model for manga style analysis. In Proceedings of the 2016 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), Shanghai, China, 20-25 March 2016; pp. 1332-1336.

16. Daiku, Y.; Augereau, O.; Iwata, M.; Kise, K. Comic Story Analysis Based on Genre Classification. In Proceedings of the 2017 14th IAPR International Conference on Document Analysis and Recognition (ICDAR), Kyoto, Japan, 9-15 November 2017; pp. 60-65.

17. Liu, X.; Li, C.; Wong, T.T. Boundary-aware texture region segmentation from manga. Comput. Vis. Media 2017, 3, 61-71.

18. Li, C.; Liu, X.; Wong, T.T. Deep extraction of manga structural lines. ACM Trans. Graph. 2017, 36, 117.

19. Rigaud, C.; Tsopze, N.; Burie, J.C.; Ogier, J.M. Robust frame and text extraction from comic books. In Graphics Recognition. New Trends and Challenges; Springer: Heidelberg/Berlin, Germany, 2013; pp. 129-138.

20. Aramaki, Y.; Matsui, Y.; Yamasaki, T.; Aizawa, K. Text detection in manga by combining connected- component-based and region-based classifications. In Proceedings of the 2016 IEEE International Conference on Image Processing (ICIP), Phoenix, AZ, USA, 25-28 September 2016; pp. 2901-2905.

21. Rigaud, C.; Burie, J.C.; Ogier, J.M. Segmentation-Free Speech Text Recognition for Comic Books. In Proceedings of the 14th IAPR International Conference on Document Analysis and Recognition (ICDAR), Kyoto, Japan, 9-15 November 2017; pp. 29-34.

22. Hiroe, S.; Hotta, S. Histogram of Exclamation Marks and Its Application for Comics Analysis. In Proceedings of the 2017 14th IAPR International Conference on Document Analysis and Recognition (ICDAR), Kyoto, Japan, 9-15 November 2017; pp. 66-71.

23. Sun, W.; Burie, J.C.; Ogier, J.M.; Kise, K. Specific comic character detection using local feature matching. In Proceedings of the 2013 12th International Conference on Document Analysis and Recognition (ICDAR), Washington, DC, USA, 25-28 August 2013; pp. 275-279.

24. Chu, W.T.; Li, W.W. Manga FaceNet: Face Detection in Manga based on Deep Neural Network. In Proceedings of the 2017 ACM on International Conference on Multimedia Retrieval, Bucharest, Romania, 6-9 June 2017; pp. 412-415.

25. Qin, X.; Zhou, Y.; He, Z.; Wang, Y.; Tang, Z. A Faster R-CNN Based Method for Comic Characters Face Detection. In Proceedings of the 2017 14th IAPR International Conference on Document Analysis and Recognition (ICDAR), Bucharest, Romania, 6-9 June 2017; pp. 1074-1080.

26. Nguyen, N.V.; Rigaud, C.; Burie, J.C. Comic Characters Detection Using Deep Learning. In Proceedings of the 2017 14th IAPR International Conference on Document Analysis and Recognition (ICDAR), Kyoto, Japan, 9-15 November 2017; pp. 41-46.

27. Cao, Z.; Simon, T.; Wei, S.E.; Sheikh, Y. Realtime multi-person 2d pose estimation using part affinity fields. In Proceedings of the 2017 IEEE Conference on Computer Vision and Pattern Recognition, Honolulu, HI, USA, 21-26 July 2017.

28. Correia, J.M.; Gomes, A.J. Balloon extraction from complex comic books using edge detection and histogram scoring. Multimed. Tools Appl. 2016, 75, 11367-11390.

29. Rigaud, C.; Le Thanh, N.; Burie, J.C.; Ogier, J.M.; Iwata, M.; Imazu, E.; Kise, K. Speech balloon and speaker association for comics and manga understanding. In Proceedings of the 2015 13th International Conference on Document Analysis and Recognition (ICDAR), Tunis, Tunisia, 23-26 August 2015; pp. 351-355.

30. Yamanishi, R.; Tanaka, H.; Nishihara, Y.; Fukumoto, J. Speech-balloon Shapes Estimation for Emotional Text Communication. Inf. Eng. Express 2017, 3, 1-10.

31. Tanaka, T.; Shoji, K.; Toyama, F.; Miyamichi, J. Layout Analysis of Tree-Structured Scene Frames in Comic Images. In Proceedings of the 20th International Joint Conference on Artifical intelligence, Hyderabad, India, 6-12 January 2007; pp. 2885-2890.

32. Arai, K.; Tolle, H. Automatic e-comic content adaptation. Int. J. Ubiquit. Comput. 2010, 1, 1-11.

33. Pang, X.; Cao, Y.; Lau, R.W.; Chan, A.B. A robust panel extraction method for manga. In Proceedings of the 22nd ACM International Conference on Multimedia, Orlando, Florida, USA, 3-7 November 2014; pp. 1125-1128.

34. Iyyer, M.; Manjunatha, V.; Guha, A.; Vyas, Y.; Boyd-Graber, J.; Daume III, H.; Davis, L. The Amazing Mysteries of the Gutter: Drawing Inferences Between Panels in Comic Book Narratives. In Proceedings of the 2017 IEEE Conference on Computer Vision and Pattern Recognition, Honolulu, HI, USA, 21-26 July 2017.

35. Rigaud, C.; Guerin, C.; Karatzas, D.; Burie, J.C.; Ogier, J.M. Knowledge-driven understanding of images in comic books. Int. J. Doc. Anal. Recognit. 2015, 18, 199-221.

36. McCloud, S. Understanding Comics: The Invisible Art; HarperCollins Publishers: New York, NY, USA, 1993.

37. Mohammad, S.M. Sentiment analysis: Detecting valence, emotions, and other affectual states from text. In Emotion Measurement; Elsevier: New York, NY, USA, 2016; pp. 201-237.

38. Cohn, N. Visual narrative structure. Cognit. Sci. 2013, 37, 413-452.

39. Matsui, Y.; Ito, K.; Aramaki, Y.; Fujimoto, A.; Ogawa, T.; Yamasaki, T.; Aizawa, K. Sketch-based manga retrieval using manga109 dataset. Multimed. Tools Appl. 2017, 76, 21811-21838.

40. Narita, R.; Tsubota, K.; Yamasaki, T.; Aizawa, K. Sketch-Based Manga Retrieval Using Deep Features. In Proceedings of the 2017 14th IAPR International

Conference on Document Analysis and Recognition (ICDAR), Kyoto, Japan, 9-15 November 2017; pp. 49-53.

41. Le, T.N.; Luqman, M.M.; Burie, J.C.; Ogier, J.M. Retrieval of comic book images using context relevance information. In Proceedings of the 1st International Workshop on coMics ANalysis, Processing and Understanding, Cancun, Mexico, 4 December 2016; p. 12.

42. Saito, M.; Matsui, Y. Illustration2vec: a semantic vector representation of illustrations. In Proceedings of the SIGGRAPH Asia 2015 Technical Briefs, Kobe, Japan, 2-6 November 2015; p. 5.

43. Vie, J.J.; Yger, F.; Lahfa, R.; Clement, B.; Cocchi, K.; Chalumeau, T.; Kashima, H. Using Posters to Recommend Anime and Mangas in a Cold-Start Scenario. arXiv 2017, arXiv:1709.01584.

44. Yao, C.Y.; Hung, S.H.; Li, G.W.; Chen, I.Y.; Adhitya, R.; Lai, Y.C. Manga Vectorization and Manipulation with Procedural Simple Screentone. IEEE Trans. Vis. Comput. Graph. 2017, 23, 1070-1084.

45. Zhang, S.H.; Chen, T.; Zhang, Y.F.; Hu, S.M.; Martin, R.R. Vectorizing cartoon animations. IEEE Trans. Vis. Comput. Graph. 2009, 15, 618-629.

46. Qu, Y.; Wong, T.T.; Heng, P.A. Manga colorization. ACM Trans. Graph. 2006, 25, 1214-1220.

47. Sato, K.; Matsui, Y.; Yamasaki, T.; Aizawa, K. Reference-based manga colorization by graph correspondence using quadratic programming. In Proceedings of the SIGGRAPH Asia 2014 Technical Briefs, Shenzhen, China, 3-6 December 2014; p.

48. Cinarel, C.; Zhang, B. Into the Colorful World of Webtoons: Through the Lens of Neural Networks. In Proceedings of the 2017 14th IAPR International Conference on Document Analysis and Recognition (ICDAR), Kyoto, Japan, 9-15 November 2017; pp. 35-40.

49. Furusawa, C.; Hiroshiba, K.; Ogaki, K.; Odagiri, Y. Comicolorization: semi-automatic manga colorization. In Proceedings of the SIGGRAPH Asia 2017 Technical Briefs, Bangkok, Thailand, 27-30 November 2017; p. 12.

50. Zhang, L.; Ji, Y.; Lin, X. Style transfer for anime sketches with enhanced residual u-net and auxiliary classifier GAN. arXiv 2017, arXiv:1706.03319.

51. Kopf, J.; Lischinski, D. Digital reconstruction of halftoned color comics. ACM Trans. Graph. 2012, 31, 140.

52. lllyasviel. Style2paints Github. Available online: https://github.com/lllyasviel/style2paints

53. Preferred Networks. Hakusensha and Hakuhodo DY Digital Announces the Launch of Colorized Manga Products Using PaintsChainer. 2018. Available online: https://www.preferred-networks.jp/en/news/ pr20180206

54. Cao, Y.; Chan, A.B.; Lau, R.W. Automatic stylistic manga layout. ACM Trans. Graph. 2012, 31, 141.

55. Wu, Z.; Aizawa, K. MangaWall: Generating manga pages for real-time applications. In Proceedings of the Acoustics, Speech and Signal Processing (ICASSP), Florence, Italy, 4-9 May 2014; pp. 679-683.

56. Jin, Y.; Zhang, J.; Li, M.; Tian, Y.; Zhu, H.; Fang, Z. Towards the Automatic Anime Characters Creation with Generative Adversarial Networks. arXiv 2017, arXiv:1708.05509.

57. Jin, Y.; Zhang, J.; Li, M.; Tian, Y.; Zhu, H.; Fang, Z. MakeGirlsMoe. Available online: http://make.girls.moe/#/

58. Cao, Y.; Pang, X.; Chan, A.B.; Lau, R.W. Dynamic Manga: Animating Still Manga via Camera Movement. IEEE Trans. Multimed. 2017, 19, 160-172.

59. Jain, E.; Sheikh, Y.; Hodgins, J. Predicting Moves-on-Stills for Comic Art Using Viewer Gaze Data. IEEE Comput. Graph. Appl. 2016, 36, 34-45.

60. Тернопіль: ТНЕУ, 2018. 67 с. 75. Комар М.П., Саченко А.О., Васильків Н.М. Методичні рекомендації до виконання кваліфікаційної роботи з освітньо-професійної програми «Комп'ютерні науки» спеціальності 122 «Комп'ютерні науки» за другим (магістерським) рівнем вищої освіти. Тернопіль: ЗУНУ, 2021. 32 с.

Додаток А

Код програмної реалізації StyleGAN2-ADA

```
# -*- coding: utf-8 -*-
"""SG2-ADA-PyTorch.ipynb

Automatically generated by Colaboratory.

Original file is located at
    https://colab.research.google.com/github/dvschultz/stylegan2-ada-pytorch/blob/main/SG2_ADA_PyTorch.ipynb

# StyleGAN2-ADA-PyTorch

## Setup
"""

!nvidia-smi -L

from google.colab import drive
drive.mount('/content/drive')

"""## Install repo

"""

# Commented out IPython magic to ensure Python compatibility.
import os
!pip install gdown --upgrade

if os.path.isdir("/content/drive/MyDrive/colab-sg2-ada-pytorch"):
```

```
#     %cd "/content/drive/MyDrive/colab-sg2-ada-pytorch/stylegan2-ada-pytorch"
elif os.path.isdir("/content/drive/"):
    #install script
#     %cd "/content/drive/MyDrive/"
    !mkdir colab-sg2-ada-pytorch
#     %cd colab-sg2-ada-pytorch
    !git clone https://github.com/dvschultz/stylegan2-ada-pytorch
#     %cd stylegan2-ada-pytorch
    !mkdir downloads
    !mkdir datasets
    !mkdir pretrained
    !gdown --id 1-5xZkD8ajXw1DdopTkH_rAoCsD72LhKU -O
/content/drive/MyDrive/colab-sg2-ada-pytorch/stylegan2-ada-
pytorch/pretrained/wikiart.pkl
else:
    !git clone https://github.com/dvschultz/stylegan2-ada-pytorch
#     %cd stylegan2-ada-pytorch
    !mkdir downloads
    !mkdir datasets
    !mkdir pretrained
#     %cd pretrained
    !gdown --id 1-5xZkD8ajXw1DdopTkH_rAoCsD72LhKU
#     %cd ../


#Uninstall new JAX
!pip uninstall jax jaxlib -y
#GPU frontend
!pip install "jax[cuda11_cudnn805]==0.3.10" -f https://storage.googleapis.com/jax-
releases/jax_cuda_releases.html
#CPU frontend
```

```
#!pip install jax[cpu]==0.3.10

#Downgrade Pytorch

!pip uninstall torch torchvision -y

!pip install torch==1.9.0+cu111 torchvision==0.10.0+cu111 -f
https://download.pytorch.org/whl/torch_stable.html

!pip install timm==0.4.12 ftfy==6.1.1 ninja==1.10.2 opensimplex


"""You probably don’t need to run this, but this will update your repo to the latest and
greatest."""


# Commented out IPython magic to ensure Python compatibility.

# %cd "/content/drive/My Drive/colab-sg2-ada-pytorch/stylegan2-ada-pytorch"

!git config --global user.name "test"

!git config --global user.email "test@test.com"

!git fetch origin

!git pull

!git stash

!git checkout origin/main -- train.py generate.py legacy.py
closed_form_factorization.py flesh_digression.py apply_factor.py README.md
calc_metrics.py training/stylegan2_multi.py training/training_loop.py util/utilgan.py


"""## Dataset Preparation


Завантажте файл .zip з квадратними зображеннями до теки `datasets`. Раніше
вам потрібно було конвертувати вашу модель у .tfrecords. Це більше не потрібно


## Train model


* `dataset_path`: це шлях до вашого .zip-файлу

* `resume_from`: якщо ви починаєте з нового набору даних, я рекомендую
`ffhq1024` або `'./pretrained/wikiart.pkl``.
```

* `mirror_x` та `mirror_y`: Дозволяють набору даних використовувати горизонтальне або вертикальне віддзеркалення.

"""

#required: definitely edit these!

dataset_path = '/content/drive/MyDrive/stylegan_xl/data/unsplash-landscapes-1024.zip'

resume_from = './pretrained/wikiart.pkl'

aug_strength = 0.0

train_count = 0

mirror_x = True

#mirror_y = False

#optional: you might not need to edit these

gamma_value = 50.0

augs = 'bg'

config = '11gb-gpu'

snapshot_count = 4

!python train.py --gpus=1 --cfg=$config --metrics=None --outdir=./results --data=$dataset_path --snap=$snapshot_count --resume=$resume_from --augpipe=$augs --initstrength=$aug_strength --gamma=$gamma_value --mirror=$mirror_x --mirrory=False --nkimg=$train_count

"""### Відновлення навчання

Після завершення роботи Colab вам потрібно буде відновити тренування. Скиньте значення наведених вище змінних, зокрема параметрів `resume_from` і `aug_strength`.

1. Вкажіть `resume_from` на останній .pkl, який ви тренували (ви знайдете його у теці `results`).

2. Оновіть `aug_strength`, щоб він відповідав значенню доповнення останнього pkl-файлу. Часто ви бачите це у консолі, але вам може знадобитися зазирнути до `log.txt`. Оновлення цього файлу гарантує, що навчання залишатиметься максимально стабільним.

3. Можливо, ви захочете оновити `train_count`, щоб відстежувати прогрес вашого тренування.

Після того, як все це буде скинуто, запустіть цю комірку зі змінними і наступну за нею комірку з тренувальною командою.

Конвертація застарілої моделі

Якщо ви використовуєте стару версію моделі (StyleGAN на основі Tensorflow або завантажений з Runway .pkl файл), вам потрібно конвертувати її в найновішу версію. Якщо ви навчалися у цьому зошиті, вам **не** потрібно використовувати цю комірку.

`--source`: шлях до моделі, яку ви хочете конвертувати

`--dest`: шлях та ім'я файлу для перетворення.
"""

!python legacy.py --source=/content/drive/MyDrive/runway.pkl --dest=/content/drive/MyDrive/colab-sg2-ada-pytorch/stylegan2-ada-pytorch/runway.pkl

"""## Тестування/Висновки

Також відоме як "Висновок", "Оцінка" або "Тестування" моделі. Це процес використання навченої моделі для створення нового матеріалу, зазвичай зображень або відео.

Згенерувати окремі зображення

`--network`: Переконайтеся, що аргумент `--network` вказує на ваш файл .pkl. (Я вважаю за краще клацнути правою кнопкою миші на файлі у панелі Files ліворуч і вибрати `Copy Path`, а потім вставити цей шлях до аргументу після знака `=`).

`--seeds`: Дозволяє вибрати випадкові насіння з моделі. Пам'ятайте, що наші вхідні дані у StyleGAN - це 512-вимірний масив. Ці зерна згенерують ці 512 значень. Кожне значення буде генерувати інший, випадковий масив. Одне і те ж значення також завжди генерує один і той же випадковий масив, тому пізніше ми можемо використовувати його для інших цілей, наприклад, для інтерполяції.

Усічення: Усічення, ну, усікає латентний простір. Це може мати незначний або значний вплив на ваші зображення, залежно від значення, яке ви використовуєте. Чим менше число, тим реалістичніше виглядатимуть ваші зображення, але це також вплине на різноманітність. Більшість людей обирають між 0,5 і 1,0, але технічно цей діапазон нескінченний.

"""

!python generate.py --outdir=/content/out/images/ --trunc=0.8 --seeds=0 --network=/content/drive/MyDrive/ladiescrop-network-snapshot-012885.pkl

!python generate.py --outdir=/content/out/images/ --trunc=0.7 --size=1820-1024 --scale-type=symm --seeds=0-499 --network=/content/crystal.pkl

!python generate.py --process="truncation" --outdir=/content/out/trunc-trav-3/ --start=-0.8 --stop=2.8 --increment=0.02 --seeds=470 --network=/content/drive/MyDrive/stylegan2-transfer-models/mixed6k-network-snapshot-016470.pkl

"""### Інтерполяції

Інтерполяція - це процес генерування дуже малих змін у векторі для того, щоб він виглядав анімованим від кадру до кадру.

Нижче ми розглянемо різні приклади інтерполяції.

Параметри

`--network`: шлях до вашого .pkl-файлу

`--interpolation`: Тип кроку визначає тип інтерполяції, який ви бажаєте. У деяких випадках він також може вказувати, чи потрібен вам z-простір, чи w-простір.

`--frames`: Кількість кадрів, які ви хочете створити. Використовуйте цей параметр для керування довжиною відео.

`--trunc`: значення усічення

Linear Interpolation
"""

!python generate.py --outdir=/content/out/video1-w-0.5/ --space="z" --trunc=0.5 --process="interpolation" --seeds=463,470 --network=/content/drive/MyDrive/stylegan2-transfer-models/mixed6k-network-snapshot-016470.pkl

!python generate.py --outdir=out/video1-w/ --space="w" --trunc=1 --process="interpolation" --seeds=85,265,297,849 --network=/content/stylegan2-ada-pytorch/pretrained/wikiart.pkl

"""#### Інтерполяція Slerp

Це трохи запаморочливо, але технічно лінійна інтерполяція не є найкращою у високорозмірних ГПС. [Це посилання на github] (https://github.com/soumith/dcgan.torch/issues/14) є одним з найпопулярніших пояснень, що обговорюються в рекламі.

Насправді я не бачу великої різниці між лінійною та сферичною інтерполяціями (у багатьох випадках достатньо різниці у z- та w-просторі), але я реалізував slerp тут для тих, хто цікавиться.

"""

!python generate.py --outdir=out/slerp-z/ --space="z" --trunc=1 --process="interpolation" --interpolation="slerp" --seeds=85,265,297,849 --network=/content/stylegan2-ada-pytorch/pretrained/wikiart.pkl --frames=24

!python generate.py --outdir=out/slerp-w/ --space="w" --trunc=1 --process="interpolation" --interpolation="slerp" --seeds=85,265,297,849 --network=/content/stylegan2-ada-pytorch/pretrained/wikiart.pkl --frames=12

"""#### Шумова петля

Якщо ви хочете зробити випадкову, але цікаву інтерполяцію вашої моделі, шумова петля - саме те, що треба. Він створює випадковий шлях у просторі z, щоб показати вам різноманітний набір зображень.

`--interpolation="noiseloop"`: встановлює використання функції шумової петлі

`--diameter`: Визначає "ширину" петлі. Зробіть його меншим, щоб показати менш різноманітний діапазон вибірок. Зробіть його більшим, щоб покрити велику кількість семплів. Цей параметр, а також `--frames` можуть допомогти визначити, наскільки швидким буде відео.

`--random_seed`: цей параметр дозволяє вам змінювати початкове місце у просторі z. Примітка: це значення не має нічого спільного з насінням, яке ви використовуєте для генерації зображень. Воно просто дозволяє вам випадковим чином вибрати початкову точку (і якщо ви захочете повернутися до неї, ви можете використати той самий семпл декілька разів).

Шумові петлі наразі працюють лише у просторі z.

"""

!python generate.py --outdir=out/video-noiseloop-0.9d/ --trunc=0.8 --process="interpolation" --interpolation="noiseloop" --diameter=0.9 --random_seed=100 --network=/content/stylegan2-ada-pytorch/pretrained/wikiart.pkl

"""#### Circular Loop

The noise loop is, well, noisy. This circular loop will feel much more even, while still providing a random loop.

I recommend using a higher `--diameter` value than you do with noise loops. Something between `50.0` and `500.0` alongside `--frames` can help control speed and diversity.
"""

!python generate.py --outdir=out/video-circularloop/ --trunc=1 --process="interpolation" --interpolation="circularloop" --diameter=800.00 --frames=720 --random_seed=90 --network=/content/stylegan2-ada-pytorch/pretrained/wikiart.pkl

"""## Projection

Basic Projector

* `--target`: this is a path to the image file that you want to "find" in your model. This image must be the exact same size as your model.

* `--num-steps`: how many iterations the projctor should run for. Lower will mean less steps and less likelihood of a good projection. Higher will take longer but will likely produce better images.
"""

!python projector.py --help

```
!python projector.py --network=/content/drive/MyDrive/colab-sg2-ada-pytorch/stylegan2-ada-pytorch/results/00023-chin-morris-mirror-11gb-gpu-gamma50-bg-resumecustom/network-snapshot-000304.pkl --outdir=/content/projector/ --target=/content/img005421_0.png --num-steps=200 --seed=0


"""### Peter Baylies' Projector"""


!python /content/stylegan2-ada-pytorch/pbaylies_projector.py --help


!python /content/stylegan2-ada-pytorch/pbaylies_projector.py --network=/content/ladiesblack.pkl --outdir=/content/projector-no-clip-006265-4-inv-3k/ --target-image=/content/img006265-4-inv.png --num-steps=3000 --use-clip=False --use-center=False --seed=99


"""## Combine NPZ files together"""


!python combine_npz.py --outdir=/content/npz --npzs='/content/projector-no-clip-006264-1-inv-3k/projector-no-clip-006264-1-inv-3k.npz,/content/projector-no-clip-006265-1-inv-3k/projector-no-clip-006265-1-inv-3k.npz,/content/projector-no-clip-006264-5-inv-3k/projector-no-clip-006264-5-inv-3k.npz,/content/projector-no-clip-006265-3-inv-3k/projector-no-clip-006265-3-inv-3k.npz,/content/projector-no-clip-006265-4-inv-3k/projector-no-clip-006265-4-inv-3k.npz,/content/projector-no-clip-006264-1-inv-3k/projector-no-clip-006264-1-inv-3k.npz'


!python generate.py --help


!python generate.py --process=interpolation --interpolation=linear --easing=easeInOutQuad --space=w --network=/content/ladiesblack.pkl --outdir=/content/combined-proj/ --projected-w=/content/npz/combined.npz --frames=120


"""## Feature Extraction using Closed Form Factorization
```

Feature Extraction is the process of finding “human readable” vectors in a StyleGAN model. For example, let’s say you wanted to find a vector that could open or close a mouth in a face model.

The feature extractor tries to automate the procss of finding important vectors in your model.

`--ckpt`: This is the path to your .pkl file. In other places its called `--network` (It’s a long story for why its name changed here)

`--out`: path to save your output feature vector file. The file name must end in `.pt`!

"""

!python closed_form_factorization.py --out=/content/ladiesblack-cff.pt --ckpt=/content/ladiesblack.pkl

"""Once this cell is finished you’ll want to save that `.pt` file somewhere for reuse.

This process just created the vctor values, but we need to test it on some seed values to determine what each vector actually changes. The `apply_factor.py` script does this.

Arguments to try:

* `-i`: This stands for index. By default, the cell above will produce 512 vectors, so `-i` can be any value from 0 to 511. I recommend starting with a higher value.

* `-d`: This stands for degrees. This means how much change you want to see along th vector. I recommend a value between 5 and 10 to start with.

* `--seeds`: You know what these are by now right? :)

* `--ckpt`: path to your .pkl file

* `--video`: adding this to your argument will produce a video that animates your seeds along the vector path. I find it much easier to figure out what’s changing with an animation.

* `--output`: where to save the images/video

* `--space`: By default this will use the w space to reduce entanglement

Lastly you need to add the path to the `.pt` file you made in th above cell. It’s weird, but you don’t need to add any arguments bfore it, just make sure its after `apply_factor.pt`

"""

!python apply_factor.py -i 0 -d 10 --seeds 5,10 --ckpt /content/ladiesblack.pkl /content/ladiesblack-cff.pt --output /content/cff-vid/ --video

"""That just produced images or video for a single vector, but there are 511 more! To generate every vector, you can uuse the cell below. Update any arguments you want, but don’t touch the `-i {i}` part.

Warning: This takes a long time, especially if you have more than one seed value (pro tip: don’t usee more than one seed value)! Also, this will take up a good amount of space in Google Drive. You’ve been warned!
"""

for i in range(512):

  !python apply_factor.py -i {i} -d 10 --seeds 177 --ckpt /content/drive/MyDrive/network-snapshot-008720.pkl /content/ladies-black-cff.pt --output /content/drive/MyDrive/ladiesblack-cff-17/ --video #--out_prefix 'ladiesblack-factor-{i}'

"""# Layer Manipulations

The following scripts allow you to modify various resolution layers of the StyleGAN model.

"""

```
!python flesh_digression.py --pkl /content/stylegan2-ada-
pytorch/pretrained/wikiart.pkl --psi 0.5 --seed 9999


!python blend_models.py --lower_res_pkl /content/ffhq-pt.pkl --split_res 64 --
higher_res_pkl /content/bone-bone-pt.pkl --output_path /content/ffhq-bonebone-
split64.pkl
```

Додаток Б

Апробація отриманих результатів

ОГЛЯД АРХІТЕКТУРИ МЕРЕЖІ STYLEGAN2-ADA ДЛЯ ГЕНЕРАЦІЇ КОМІКСІВ

Лисюк Роман Олександрович
здобувач вищої освіти факультету комп'ютерних інформаційних технологій
Західноукраїнський національний університет, Україна
Науковий керівник: Заргородня Діана Іванівна
к.т.н, кафедри Інформаційно
обчислювальних систем і управління
Західноукраїнський національний університет, Україна

StyleGAN2-ADA (Adaptive Discriminator Augmentation) є розширенням архітектури StyleGAN2, яке вводить адаптивну аугментацію для дискримінатора, спеціально розроблену для покращення тренування моделей на невеликих датасетах. Ця архітектура створена командою дослідників з Nvidia і вирішує кілька ключових викликів, які стоять перед попередніми версіями StyleGAN. Нижче наведено детальний опис основних аспектів StyleGAN2-ADA:

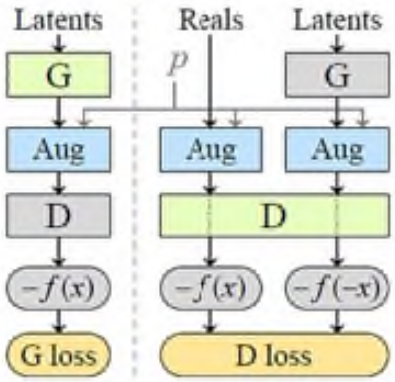

Рис. 1. **Доповнення стохастичного дискримінатора**

У контексті архітектури StyleGAN2-ADA, G та D використовуються для позначення двох ключових компонентів моделі: G є генератором, а D -