https://doi.org/10.31649/1997-9266-2022-160-1-82-94

УДК 004.054:[004.032.26+004.85]

Я. О. Ісаєнков[1]
О. Б. Мокін[1]

АНАЛІЗ ГЕНЕРАТИВНИХ МОДЕЛЕЙ ГЛИБОКОГО НАВЧАННЯ ТА ОСОБЛИВОСТЕЙ ЇХ РЕАЛІЗАЦІЇ НА ПРИКЛАДІ WGAN

[1]Вінницький національний технічний університет

Представлено особливості будови, навчання та сфери застосування генеративних моделей глибокого навчання. До основних завдань таких модель відносяться генерування даних (зображень, музики, текстів, відео), перенесення стилів з одних даних на інші, поліпшення якості даних, їх кластеризація, пошук аномалій тощо. Зазначено, що результати роботи генеративних моделей, окрім поширених розважальних цілей, можуть використовуватися як: додаткові дані для навчання інших моделей машинного навчання, джерела нових ідей для творчих професій, інструменти анонімізації чутливих даних тощо. Проаналізовано переваги та недоліки таких базових видів генеративних моделей як автокодувальники, варіаційні автокодувальники, генеративні змагальні мережі (ГЗМ), ГЗМ Васерштейна (Wasserstein GAN, WGAN), StyleGAN, StyleGAN2 та BigGAN. Також описано покрокове дослідження імплементації генеративної моделі на прикладі WGAN, яке включає як реалізацію базової архітектури цієї моделі, так і застосування складніших елементів. Прикладами таких елементів є впровадження умовної генерації для можливості вибору потрібного класу та алгоритм білінійного підвищення дискретизації для вирішення проблеми так званого «ефекту шахової дошки». Фінальна модель, яка була створена в результаті дослідження та отримала назву CWGAN-GP_128, здатна генерувати реалістичні зображення кульбабок та чорнобривців у роздільній здатності 128×128 пікселів. Модель навчалася на авторському наборі даних, що складається з 900 фотографій (по 450 для кожного класу). У процесі навчання для аугментації зображень використовувалися такі афінні перетворення, як повороти та перевертання. Наголошено, що хоч результати роботи генеративних моделей часто легко оцінити візуально, проте разом з бурхливим розвитком ГЗМ зростає актуальність проблеми автоматизації процесу оцінювання якості згенерованих даних. Остаточна модель відкрита для публічного доступу, а з результатами її роботи можна ознайомитися на авторському вебсайті thisflowerdoesnotexist.herokuapp.com.

Ключові слова: генерування даних, генеративна змагальна мережа, автокодувальник, глибоке навчання, ГЗМ, GAN, WGAN.

1. Вступ

Генеративні моделі в останні роки отримали стрімкий розвиток за рахунок покращення можливостей для тренування більших та складніших моделей. Результати їх роботи стають популярнішими навіть за межами наукового середовища. Яскравим прикладом цього є система This Person Does Not Exist [1], що здатна генерувати великі (1024×1024 пікселів) фотореалістичні зображення облич неіснуючих людей (рис. 1). Іншим популярним прикладом є модель, що перетворює семантичну маску об’єктів у надзвичайні природні ландшафти [2], дозволяючи будь-кому, незалежно від художніх навичок, створювати мальовничі пейзажі (рис. 2).

Рис. 1. Приклади роботи сервісу This Person Does Not Exist

Рис. 2. Створення реалістичного ландшафту за ескізом (маскою об’єктів)

На сьогоднішній день генеративні моделі використовуються для розв’язання широкого спектра задач. Найвагоміші результати, на думку авторів, досягнуті у таких напрямках:

1. Генерування реалістичних зображень, зокрема з високою роздільною здатністю [3]. Такі зображення все частіше використовуються для доповнення даних (аугментації) під час тренування або налаштування моделей машинного навчання. Особливо це актуально для наборів даних з малою кількістю прикладів вихідних класів та/або з великим дисбалансом даних за цими класами. З чим часто можна зіштовхнутися в процесі роботи з тими ж медичними даними, наприклад, у задачі класифікації медичних кондицій за рентгенограмами грудної клітини [4]. Але, звісно, у роботі з медичними даними потрібно бути вкрай обережними, пам’ятаючи, що генеративні моделі, фактично, «вигадують» ці дані [5]. Також подібні згенеровані зображення можуть виступати у ролі самостійних креативних композицій або своєрідною підказкою для формування нових концепцій [6], наприклад, під час дизайну меблів, логотипів, інтер’єру приміщень, флористичних композицій тощо.

2. Покращення роздільної здатності зображень [7]. Наприклад, для відновлювання старих фотокарток та навіть покращення МРТ та рентгенівських знімків. Але, як уже зазначалося, цей підхід потрібно використовувати дуже обережно, оскільки багато деталей, особливо за суттєвої зміни роздільної здатності, генеративні моделі «додумують» з того, що вони бачили у процесі тренування.

3. Зміна атрибутів об’єкта. Наприклад, додавання окулярів на фото, зміна рис обличчя, одягу тощо. І хоча застосування цього інструмента більше направлене на розважальну та торговельну сфери [8], проте виникають і такі критичні випадки, коли генеративна модель дозволяє, до прикладу, приховати особистості свідків та інших осіб у гучній судовій справі [9]. На жаль, ця технологія використовується і зловмисниками для створення так званих «діпфейків» [10], коли, наприклад, створюються фіктивні компрометуючі зображення чи відео шляхом підміни облич.

4. Створення зображень за заданим текстовим описом [11], що також може використовуватись для створення аугментованих чи навіть нових даних.

5. Перетворення одного зображення в інше [12]. Наприклад, перетворення супутникового знімка в карту географічних/фізичних об’єктів, створення з чорно-білого зображення кольорового, перенесення особливостей однієї породи тварин на іншу тощо.

6. Генерація музики, мови, звуків [13].

7. Перетворення тексту в аудіо формат, що дозволяє контролювати процес створення нових прикладів для тренування, або створення аудіо-контенту (наприклад, аудіо-книжки) на основі звичайної текстової інформації [14].

8. Перетворення стилю аудіозаписів [15], як, наприклад, перетворення гітарного виконання на фортепіанне звучання або перетворення одного голосу в інший, що потенційно може допомогти у перекладі фільмів, при цьому зберігаючи оригінальні голоси акторів [16].

9. Більшість прикладів, зазначених вище, у комплексі чи окремо можуть бути застосовані і до відео.

10. Для текстів за допомогою генеративних моделей можна також розв’язувати задачі генерування [17] та реферування [18].

11. Окрім задач, пов’язаних безпосередньо з генеруванням певних об’єктів, існують також підходи використання генеративних моделей для задач фільтрації аномальних даних [19] та кластеризації [20].

Метою дослідження є огляд основних генеративних моделей, включно з їхніми перевагами та недоліками, з акцентом на найпоширеніші та найперспективніші з них, а також аналіз особливостей покрокової реалізації та удосконалення однієї з ключових архітектур — генеративної змагальної мережі Васерштейна — на реальному прикладі.

У наступному розділі наведено аналіз основних видів генеративних моделей.

2. Види генеративних моделей

Автокодувальники. Цей тип моделей складається з двох штучних нейронних мереж: кодувальника та декодувальника (рис. 3). Посередині знаходиться код (який часто також називають bottleneck, з англ. «вузьке місце») — це шар нейронів невеликої розмірності (у порівнянні з іншими шарами). Основна задача моделі — це зменшення розмірності даних, тобто кодування вхідних даних у вектор ознак меншої розмірності (код) ніж вони були спочатку та подальше відтворення (декодування) початкових даних з цього вектора [21]. Навчання автокодувальників відбувається за допомогою методу зворотного поширення помилки, а в основі функції втрат використовується різниця між оригінальними даними (наприклад, зображеннями), що подаються на вхід кодувальника, та відновленими декодувальником.

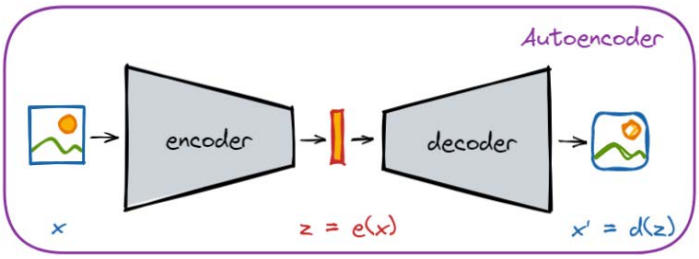

Рис. 3. Спрощена структура звичайного авто кодувальника

З погляду генерації даних найбільшим недоліком класичних автокодувальників є те, що, мінімізуючи функцію втрат, автокодувальник може різко змінювати прихований простір (навіть незважаючи на регуляризаційні механізми) [22]. В результаті точки, що знаходяться поруч, можуть відповідати за вкрай несхожі між собою за ознаками дані, або можуть існувати такі області в просторі між двома точками даних, які не дозволять відтворити щось схоже з реальним розподілом даних. Цей недолік має суттєвий вплив на вибір звичайних автокодувальників як моделей генерування даних.

Варіаційні автокодувальники [22]. Цей тип генеративних моделей не має вищеописаного недоліку, притаманного звичайним автокодувальникам. Замість визначення для кожного прикладу вхідних даних фіксованих значень коду варіаційний автокодувальник вивчає параметри його розподілу, стискаючи таким чином вхідну інформацію до обмеженого ймовірнісного латентного простору. А на вхід декодувальника надходить вектор, що випадково вибирається з цього розподілу.

У варіаційних автокодувальниках використовується так званий трюк перепараметрування, який під час тренування моделі уможливлює використання методу зворотного поширення помилки, попри випадковий характер вибору значень вектора кодування з розподілу. Суть трюку перепараметрування полягає у відокремленні стохастичності вибору значень цього вектора (він фактично стає окремим входом) від основного шляху поширення сигналів у системі, що дозволяє виключити цю складову з процесу зворотного поширення помилки (рис. 4).

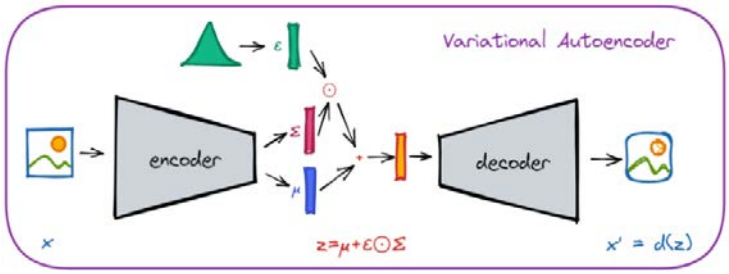

Рис. 4. Спрощена структура варіаційного авто кодувальника

Але може виникнути ситуація, коли кодувальник може створювати розподіли з нульовою дисперсією, або дуже віддаленими середніми значеннями, що є еквівалентним у різних масштабах, і таким чином може призвести до виникнення зазначеної вище проблеми, притаманної звичайному автокодувальнику. Тому у функцію втрат як регуляризаційний параметр включається міра розходження Кульбака–Лейблера між згенерованими розподілами та стандартним нормальним розподілом. Завдяки цьому всі дані групуються якомога ближче до стандартного нормального розподілу, що дозволяє брати з нього випадкові вектори і генерувати з них нові дані, а також знаходити проміжні вектори між двома вибраними за допомогою інтерполяції.

Генеративні змагальні мережі (ГЗМ). Винайдена у 2014 році [23], ця модель швидко завоювала популярність. Основними складовими ГЗМ зазвичай є дві окремі штучні нейронні мережі – генератор та дискримінатор (у деяких видах ГЗМ дискримінатор називають критиком), які показано на рис. 5. Метою генератора є відтворення (імітація) розподілу реальних даних та створення псевдореальних даних, які ще називають фейковими (від англ. «fake», яке перекладається як «підробка»). Мета дискримінатора визначити чи дані, що подаються йому на вхід, взяті з реального набору чи створені генератором. Навчання цих моделей можна описати мінімаксною грою двох

гравців: генератор намагається максимізувати вірогідність помилки дискримінатора.

Навчання ГЗМ здійснюється за допомогою зворотного поширення помилки та складається з двох кроків:

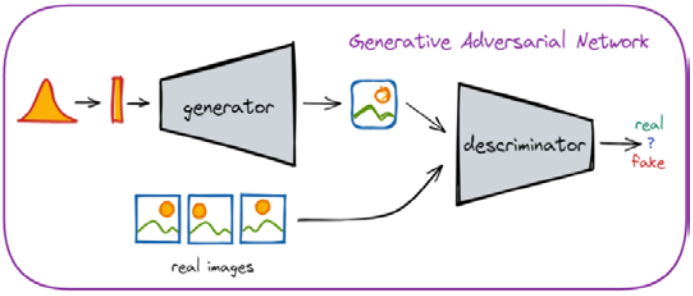

Рис. 5. Типова структура ГЗМ

1. На першому кроці відбувається навчання дискримінатора. Для цього виконуються такі дії: генератор створює набір фейкових даних, до них додається набір реальних, і далі вони разом передаються до дискримінатора. Дискримінатор намагається визначити, які дані є реальними, а які згенерованими. Відповіді дискримінатора порівнюються з реальними мітками за допомогою функції втрат (наприклад, бінарної перехресної ентропії), що дозволяє дискримінатору оновити свої ваги.

2. На другому кроці відбувається навчання генератора. Для цього останній створює набір фейкових даних, і цей набір одразу відправляється на оцінку до дискримінатора (на другому етапі реальні дані не використовуються). За допомогою функції втрат результати роботи дискримінатора порівнюються з тимчасовими мітками, які вказують, що ці зображення є справжніми (адже мета генератора зробити так, щоб дискримінатор вважав фейкові дані реальними), відповідний зворотний зв'язок надходить до генератора, і його ваги оновлюються. Варто зазначити, що хоча градієнт під час зворотного поширення помилки проходить також і через дискримінатор, його ваги на цьому етапі не оновлюються.

Для того, щоби генератор кожний раз створював унікальні дані йому на вхід подається вектор випадкового шуму з нормального розподілу (див. рис. 5), а розмір цього вектора є гіперпараметром моделі. У задачах умовної генерації, коли потрібно створювати дані конкретного класу (зі списку відомих класів), до цього вектора додається унітарний код, де значення «1» стоїть у цільовому для генерування класі. Ця інформація зазвичай подається і до дискримінатора. Оскільки у задачах генерування зображень на вхід до дискримінатора подається зображення, що має висоту H, ширину W та кількість каналів C (зазвичай $C = 3$), то унітарний код подається у вигляді додаткових каналів. При цьому кількість цих додаткових каналів дорівнює кількості класів, а кожний клас має вигляд матриці розмірності $H \times W$, повністю заповненої нулями або одиницями. Тобто загальна кількість каналів після додавання інформації про клас буде дорівнювати $C + K$, де K — це кількість класів. У такому випадку задача генератора полягає не лише у розділенні даних на реальні та фейкові, але й у перевірці чи мають фейкові дані потрібний клас.

На сьогодні існує багато різновидів ГЗМ: за типом моделей, що використовуються як генератори та дискримінатори; за видом функції втрат, що використовується для оцінки компонентів, та іншими особливостями.

Якщо основним завданням автокодувальників є стиснення інформації та відновлення вхідних даних, то завданням ГЗМ є створення з шуму реалістичних даних. Процесом навчання автокодувальників є мінімізація різниці вхідних даних та вихідних, водночас один з основних компонентів ГЗМ намагається відтворити розподіл реальних даних.

Далі наведено опис найвідоміших ГЗМ для генерування зображень.

3. Найвідоміші ГЗМ

Deep Convolutional Generative Adversarial Networks (DCGAN) [24]. Цей тип моделей просто поєднує переваги згорткових нейронних мереж, що є одними з фаворитів основних задач комп'ютерного зору (класифікації зображень, сегментації, детектування), та базових ГЗМ, які побудовані на основі звичайних повноз'єднаних шарів.

Wasserstein GAN (WGAN) [25]. У ГЗМ Васерштейна вводиться поняття критика замість класичного дискримінатора. Мета критика дати деяку числову оцінку реальним та фейковим зображенням і якомога краще розділити реальні та фейкові дані, максимізуючи дистанцію Васерштейна, тоді як задача генератора — мінімізувати цю дистанцію. Заміна функції втрат пов'язана з тим, що у класичній функції втрат (бінарній перехресній ентропії) може виникати проблема затухання градієнта — коли дискримінатор дуже впевнено знаходить фейкові зображення та градієнт функції активації виходить на плато. У WGAN ця проблема відсутня (рис. 6).

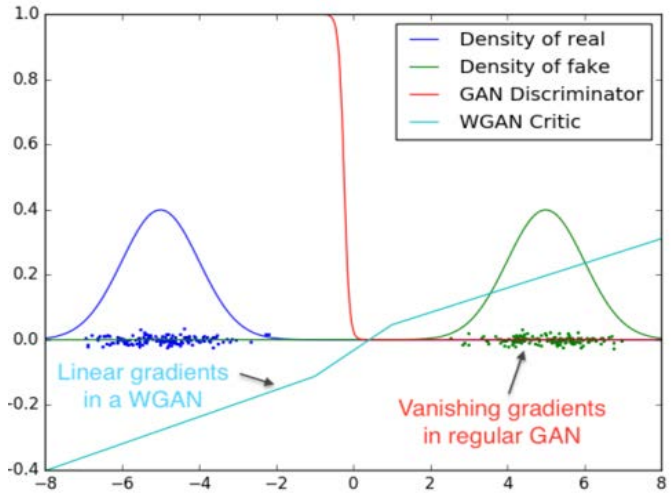

Рис. 6. Різниця між градієнтами у класичних ГЗМ та WGAN

Авторами WGAN [25] показано, що для стабілізації навчання критик повинен мати обмеження 1-Ліпшиц (норма градієнта повинна бути не більше одиниці у кожній точці). У класичній реалізації запропоновано використання відсікання ваги (weight clipping), але цей спосіб має низку суттєвих недоліків. Якщо параметр відсікання великий, то може знадобитися багато часу, щоб будь-які ваги досягли своєї межі, що ускладнює підготовку критика до оптимального стану. Якщо відсікання маленьке, це може легко призвести до зникнення градієнтів, коли кількість шарів велика, або не використовується пакетна нормалізація. Використання цього методу обумовлене його простотою та досягненням прийнятних результатів авторами моделі.

Wasserstein GAN with gradient penalty (WGAN-GP) [26]. Замість застосування відсікання ваги WGAN-GP дає штраф моделі, якщо норма градієнта відходить від її граничної цільової норми, що дорівнює одиниці.

BigGAN [27]. До особливостей, що сформували цей тип ГЗМ, можна віднести:

– використання якомога більшого розміру пакету даних, що за результатами навчання дозволило як покращити значення метрик Inception Score (IS) і Frechet Inception Distance (FID), так і запобігти проблемі втрати модальності (mode collapse);

– збільшення кількості параметрів моделі як у ширину, так і у довжину;

– використання функції завісних втрат (hinge loss), що зазвичай застосовується для методу опорних векторів;

– інформація про клас передається до генератора завдяки умовній класовій пакетній нормалізації;

– використання ортогональної регуляризації, завдяки чому до генератора можна ефективно застосовувати трюк відсікання для контролювання компромісу правильності та різноманітності, тобто якості генерування та різноманітності даних (трюк відсікання — це обрізка вектора випадкового шуму шляхом відсікання значень з величиною вищою за обране порогове значення: чим нижче значення, тим якіснішими та менш різноманітними будуть дані).

StyleGAN [28]. Однією з ключових особливостей цієї моделі є використання прогресивного зростання (progressive growing) – спочатку мережа вчиться створювати маленькі зображення (наприклад, 4×4), а потім поступово збільшує їх роздільну здатність до цільового розміру (наприклад, 1024×1024). Це забезпечується за рахунок додавання додаткових шарів. Такий підхід дозволяє спочатку вивчити великомасштабну структуру зображення, а потім переключити увагу на дрібніші деталі, замість того, щоб вивчати все одночасно.

Наступною важливою особливістю такого типу ГЗМ є мережа відображення шуму, яка є повнозв’єднаним перцептроном, що створює з вектора випадкового шуму новий вектор стилю, котрий подається в кожний з блоків основної частини моделі (у випадку звичайних ГЗМ вектор подається тільки у перший шар).

Також варто відмітити використання адаптивної нормалізації екземплярів (AdaIN), що забезпечує передачу стильової інформації з вектора стилю, отриманого мережею відображення шуму, до зображення, яке створюється. На ранніх етапах, коли розміри зображення малі, застосовуються глобальні риси, а на кінцевих етапах вже додаються детальні риси.

Останнім важливим компонентом є змішування стилів, коли для одного зображення генерується не один вектор стилю, а декілька, кожен з яких подається у різні блоки генератора, додаючи у мережу більше варіативності.

StyleGAN2 [29]. Ця модель, як зрозуміло з назви, побудована на базі StyleGAN, але деякі вищезгадані особливості зазнали суттєвих змін. Адаптивна нормалізація екземплярів може призводити до краплеподібних артефактів, приклад яких показано на рис. 7. Ця техніка перероблена з використанням демодуляції ваг.

Рис. 7. Приклад краплеподібних артефактів

Прогресивне зростання тепер використовує механізм пропуску з'єднання, що дозволяє прогресивно навчати мережу, використовуючи всі роздільні здатності одразу, але надаючи спочатку перевагу (більший коефіцієнт) маленьким зображенням з плавним переходом на великі (без зміни топології мережі).

До особливостей StyleGAN2 також можна віднести використання:

– регуляризації довжини шляху, як додаткової частини функції втрат, що дозволяє забезпечити гладкішу інтерполяцію латентного простору, тим самим поліпшуючи якість зображень та створюючи якісні проміжні кадри при інтерполяції двох зображень;

– так званої лінивої регуляризації (lazy regularization), коли регуляризаційна частина функції втрат застосовується окремо від основної функції (та менш часто).

Далі перейдемо до особливостей практичної реалізації ГЗМ на прикладі однієї з найпопулярніших її варіацій, а саме ГЗМ Васерштейна, особливості якої описано вище.

4. Особливості практичної реалізації WGAN

Тепер детальніше зупинимося на задачі генерації зображень за допомогою ГЗМ для двох типів всім добре відомих в України квітів: кульбабок та чорнобривців. Для розв'язання цієї задачі зібрано набір даних з 900 реальних фотографій квітів — 450 прикладів на кожний з двох класів, де приклад — це кольорове триканальне RGB зображення розміром 512×512 пікселів у форматі PNG (рис. 8).

Рис. 8. Приклади реальних зображень класів, вибраних для генерації

У всіх описаних далі експериментах для цих зображень використовувалася така аугментація даних:

– транспонування;

– горизонтальне перевертання;

– вертикальне перевертання;

– поворот на випадковий кут в діапазоні значень від –90° до 90°.

Під час розв'язання поставленої задачі реалізовано та проаналізовано різні архітектурні рішення щодо побудови генераторів та критиків ГЗМ.

За основу взято одну з найпопулярніших та найефективніших моделей WGAN-GP.

Для першого експерименту вибрано невеликий вихідний розмір зображень (28×28 пікселів) для перевірки того, наскільки WGAN підходить для цієї задачі генерації вибраного типу зображень. При цьому задача розділення на два класи (чорнобривці та кульбабки) у цьому експерименті не розв'язувалась, тобто задачею генератора було просто створити будь-яку квітку, схожу на реальну.

Як генератор сформовано нейрону мережу з чотирьох блоків, перші три з яких мають такий вид:

– шар транспонованої згортки (формує корисні ознаки та збільшує розмір);
– пакетна нормалізація;
– функція активації — зрізаний лінійний вузол (ReLU).

Фінальний блок містить:

– шар транспонованої згортки (формує корисні ознаки та збільшує розмір);
– функція активації — гіперболічний тангенс (tanh).

Критик складається з двох блоків виду:

– шар згортки (формує корисні ознаки та зменшує розмір);
– пакетна нормалізація;
– функція активації — нещільний зрізаний лінійний вузол (Leaky ReLU);

Останній згортковий шар критику повертає лише одне значення (має один канал з розміром 1×1), яке служить оцінкою критика — чим воно більше, тим критик впевненіший, що зображення реальне.

Результат генерування після першого етапу показано на рис. 9, з якого видно, що модель створює досить цікаві приклади, які поки мають невелику роздільну здатність. Також добре видно, що згенеровані зображення не мають чіткого розділення квітів за класами, що ілюструє приклад з першого рядка та четвертої колонки рис. 9 — половина квітки має червоний колір, а інша половина — жовтий.

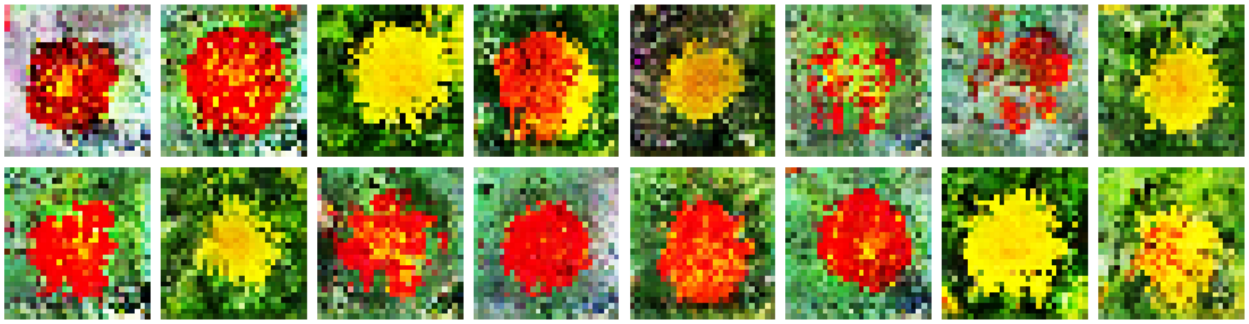

Рис. 9. Результат роботи WGAN-GP_28 на першому етапі експерименту

На другому етапі перейдемо до розділення результатів генерування за класами. Для цього використаємо так звану умовну генерацію, коли генератор, окрім вектора шуму, буде отримувати ще й вектор унітарного коду, розмір якого має дорівнювати кількості класів. Всі елементи вектора будуть нульовими, а потрібний для генерації клас позначається одиницею. В задачі генерації квітів цей вектор має тільки два елементи (наші два види квітів), а отже після конкатенації з вектором шуму, розмір якого дорівнює 64, вхідний вектор буде мати 66 елементів. У свою чергу, критик на вході отримає на два канали унітарного коду більше, щоб також мати орієнтир, якого саме класу зображення він бачить. Оскільки до цього він отримував зображення з трьох каналів (RGB), то тепер йому на вхід надходитиме матриця розмірністю 28×28×5.

Результати другого етапу генерування (умовної ГЗМ) показано на рис. 10, на якому добре видно, що, окрім вирішення проблеми з розділенням класів та можливістю генерування потрібного класу, якість зображень в цілому також стала кращою.

Рис. 10. Результат роботи CWGAN-GP_28 на другому етапі

На наступному (третьому) кроці покращимо роздільну здатність зображення до 64×64 пікселів. Для цього також використаємо вищеописані блоки, тобто для генератора це буде чотири основних блоки та один кінцевий, але з однією суттєвою відмінністю — у шарах транспонованої згортки використаємо фільтри (ядра), рівні трьом (замість чотирьох), а крок залишимо без змін (рівний двом). Це і дозволить отримати вихідний розмір зображень 64×64 пікселя. Для критика використаємо ті ж чотири основних блоки та кінцевий шар для перетворення виходу в одне значення.

Через те, що розмір ядра не ділиться без залишку на розмір кроку, виникла проблема — деякі пікселі отримують більше уваги ніж інші, викликаючи так званий «ефект шахової дошки», який можна легко помітити на результатах генерування, показаних на рис. 11. Майже у кожної картинки в лівому нижньому кутку присутній патерн, що виглядає як клітинки шахової дошки.

Рис. 11. Результат роботи CWGAN-GP_64 на третьому етапі дослідження
з яскраво вираженим ефектом шахової дошки

Щоб позбутися цього ефекту, на четвертому етапі експерименту вирішено повернути розмір фільтрів до чотирьох, а вихідного розміру (64×64) досягнути за рахунок іншого гіперпараметра — доповнення (padding) згортки. Проте це не дозволило повністю позбутися ефекту шахової дошки, хоча він і став менш помітним. На рис. 12 видно, що текстура зображення має не гладку поверхню, як повинно бути на реальних фотографіях, а повторювану з певним кроком сітку, що також нагадує шахову дошку.

Рис. 12. Результат роботи CWGAN-GP_64 на четвертому етапі експерименту
з іншим проявом ефекту шахової дошки

Для подолання і цієї варіації ефекту шахової дошки (на п’ятому етапі експерименту) змінено архітектуру блоків генератора. Вирішено відмовитись від шарів транспонованої згортки, яка відповідала і за збільшення зображення, і за його адекватність. Тепер основні блоки отримали вигляд:

– шар звичайної згортки (для покращення якості зображень та формування важливих ознак; розмір ядра — 3, розмір кроку — 1);
– пакетна нормалізація;
– функція активації — зрізаний лінійний вузол.

Останній блок генератора також змінився:

– шар звичайної згортки;
– функція активації — гіперболічний тангенс.

А після кожного (або двох) основних блоків використовується алгоритм білінійного підвищення дискретизації — для збільшення зображення у два рази, що раніше робила транспонована згортка [30]. Така трансформація забезпечила гладкіші текстурні перетворення.

Фінальна архітектура генератора показана на рис. 13.

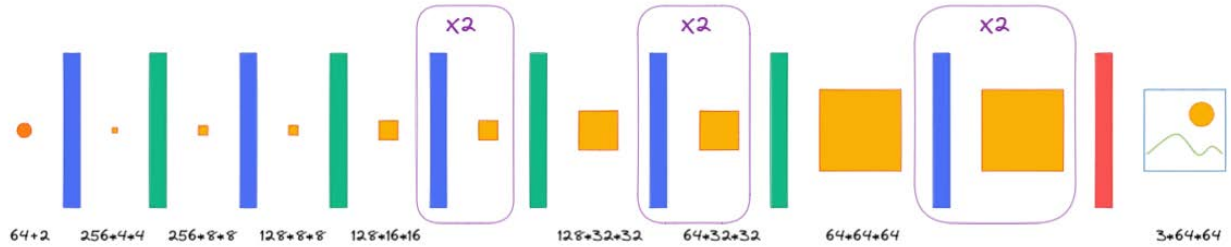

Рис. 13. Фінальна архітектура CWGAN-GP_64 (п’ятий етап експерименту):
сині блоки відповідають за формування ознак (features); зелені блоки — за підвищення дискретизації;
червоний (останній) блок — за формування вихідного зображення

З використанням показаного вище підходу вдалося суттєво покращити правильність зображень (рис. 14). Для зручності аналізу на рис. 15 наведено трохи збільшені згенеровані зображення. Очевидно, що вони мають більш гладку текстуру та не містять артефактів, притаманних результатам роботи моделей, у яких використовувалася транспонована згортка.

Рис. 14. Результат роботи моделі CWGAN-GP_64 на п’ятому етапі

Рис. 15. Демонстрація відсутності ефекту шахової дошки у згенерованих зображеннях розміром 64×64 пікселі

На останньому шостому етапі вирішено ще збільшити (вдвічі) розмір вихідного зображення. Для цього до описаної вище структури додано ще один шар підвищення дискретизації та два блоки зі згорткою (рис. 16), що дозволило отримати деталізованіші зображення розміром 128×128 пікселів (рис. 17).

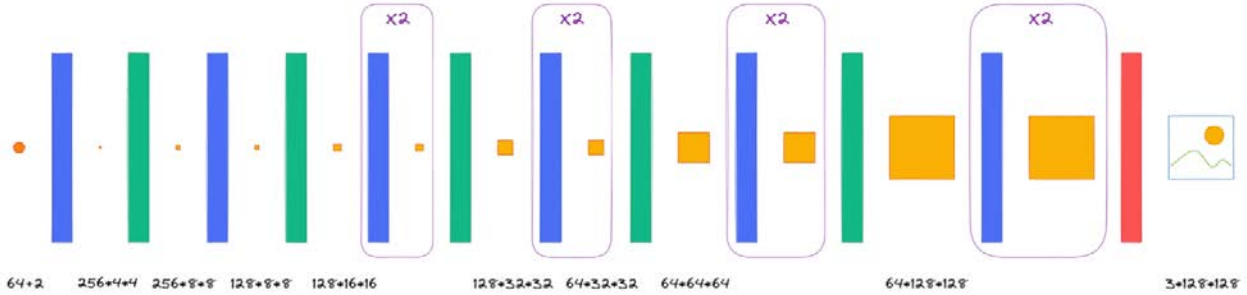

Рис. 16. Архітектура моделі CWGAN-GP_128 на шостому етапі експерименту

Рис. 17. Результат генерування моделі CWGAN-GP_128 на шостому етапі

На рис. 18 наведено збільшені зображення, отримані за допомогою моделі CWGAN-GP_128, з яких добре видно, що вони мають гладку текстуру та не містять небажаних патернів.

Рис. 18. Збільшені приклади зображень розміром 128×128 пікселів

На рис. 19 показана зміна вихідного зображення генератора (зліва – початкове, праворуч – кінцеве) з кроком 500 епох для кожного з класів при однакових вхідних даних (однаковий вектор шуму та унітарний код) протягом навчання фінальної ГЗМ моделі. Добре видно, як генератор постійно покращує якість та деталі зображень, частково зберігаючи такі особливості як розташування, розмір і напрям росту квіток, а також тло.

Рис. 19. Зміна результатів генерування протягом навчання фінальної ГЗМ

На рис. 20 подано приклади застосування трюку відсікання до фінальної генеративної моделі, який дозволяє знайти певний компроміс між правильністю та різноманітністю згенерованих зображень.

Рис. 20. Використання так званого трюку відсікання шуму для фінальної моделі:
перший рядок — відсікання 0,01; другий рядок — відсікання 1,5; третій рядок — без відсікання

У першому рядку застосовано сильне відсіканням, що дозволяє досягти високої правильності зображень: всі квітки розташовані у центрі зображення, їх межі чіткі, а також можна помітити багато інших реалістичних деталей, до прикладу, властиві відповідній рослині пелюстки. Але, на жаль, відсутня різноманітність зображень — всі вони майже повні копії одне одного.

У третьому рядку відсікання шуму повністю відсутнє, тому деякі приклади не мають чітких меж та деталей, їх розташування не завжди центроване, проте тепер вони всі різні та мають багато цікавих особливостей, наприклад, незвичний рожевий фон у третьому прикладі.

А ось у другому рядку підібрано середнє (компромісне) відсікання, коли приклади достатньо правильні та мають значну різноманітність.

Детальніше з іншими результатами та можливостями роботи фінальної моделі CWGAN-GP_128 можна ознайомитися на авторському вебсайті This Flower Does Not Exist [30].

Попри велику кількість переваг ГЗМ як генеративних моделей, результати їх роботи часто потребують ретельної перевірки, оскільки навіть за високої якості та реалістичності вони можуть містити серйозні неточності (наприклад, зображення собаки з п’ятьма лапами). Тому не менш актуальним за генерування є напрямок автоматизації перевірки та оцінки результатів роботи ГЗМ, який і буде досліджений в одній з подальших робіт.

5. Висновки

У роботі розглянуто особливості будови, навчання, сфери застосування та завдання, які можуть вирішувати генеративні моделі. Зазначено, що однією з головних задач є створення додаткових даних для доповнення існуючих або створення нових наборів даних. Показано, що генеративні моделі знаходять своє застосування у дедалі більшій кількості завдань, наприклад, знаходження аномальних даних або кластеризації. Описано переваги та недоліки генеративних змагальних мереж та їх попередників — автокодувальників. Проаналізовано найкращі та найпопулярніші види генеративних змагальних мереж для створення зображень, а саме: DCGAN, WGAN, WGAN-GP, BigGAN, StyleGAN, StyleGAN2. Наведено архітектурні особливості цих моделей та підходів до їхнього тренування.

Представлено особливості імплементації ГЗМ моделі на прикладі WGAN з використанням авторського набору даних, що містить два класи квітів: кульбабки та чорнобривці. Продемонстрована покрокова побудова та налаштування моделі для задачі генерації реалістичних зображень квітів. До основного підходу навчання додано механізм умовної генерації для забезпечення поділу на класи даних, що генеруються, а також проаналізовано ефективність використання шарів білінійного підвищення дискретизації для уникнення небажаних патернів під час навчання. В результаті дослідження отримано модель, що отримала назву CWGAN-GP_128, завдяки якій можна контрольовано створювати неіснуючі зображення розміром 128×128 пікселів двох класів квітів. За такої роздільної здатності більшість згенерованих даних візуально дуже схожі на реальні і не завжди вдається їх чітко розрізнити — згенеровані квіти мають багато рис, притаманних реальним, наприклад, наявність пелюсток та відповідність їх параметрів, форми та зміни градієнта кольору включно. Тому можна зробити висновок, що мета роботи досягнута і реалізація отриманої моделі є коректною.

СПИСОК ВИКОРИСТАНОЇ ЛІТЕРАТУРИ

[1] *This person does not exist*. [Online]. Available: https://thispersondoesnotexist.com/. Accessed on: February 2, 2022.

[2] *GauGAN2*. [Online]. Available: http://gaugan.org/gaugan2/. Accessed on: February 2, 2022.

[3] T. Karras, S. Laine, M. Aittala, J. Hellsten, J. Lehtinen, and T. Aila, “Analyzing and Improving the Image Quality of StyleGAN,” in *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020, pp. 8107-8116. https://doi.org/10.1109/CVPR42600.2020.00813 .

[4] S. Sundaram, and N. Hulkund, “GAN-based Data Augmentation for Chest X-ray Classification,” in *arXiv e-prints*, 2021. [Online]. Available: https://arxiv.org/pdf/2107.02970.pdf . Accessed on: February 2, 2022.

[5] S.-Y. Wang, O. Wang, R. Zhang, A. Owens, and A. A. Efros, “CNN-Generated Images Are Surprisingly Easy to Spot… for Now,” *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020, pp. 8692-8701. https://doi.org/10.1109/CVPR42600.2020.00872 .

[6] *This vessel does not exist*. [Online]. Available: https://thisvesseldoesnotexist.com/ . Accessed on: February 2, 2022.

[7] J. Ho, C. Saharia, W. Chan, D. J. Fleet, M. Norouzi, and T. Salimans, “Cascaded Diffusion Models for High Fidelity Image Generation,” in *arXiv e-prints*, 2021. [Online]. Available: https://arxiv.org/pdf/2106.15282.pdf . Accessed on: February 2, 2022.

[8] *Augmented Reality for Jewelry*. [Online]. Available: https://tryon.jewelry/main . Accessed on: February 2, 2022.

[9] Z. Cai, Z. Xiong, H. Xu, P. Wang, W. Li, and Y. Pan., “Generative Adversarial Networks: A Survey Toward Private and Secure Applications,” *ACM Computing Surveys*, vol. 54, no. 6, pp. 1-38, July, 2022. https://doi.org/10.1145/3459992 .

[10] R. Tolosana, R. Vera-Rodriguez, J. Fierrez, A. Morales, and J. Ortega-Garcia, “Deepfakes and beyond: A survey of face manipulation and fake detection,” in *Information Fusion*, vol 64, pp. 131-148, December, 2020. https://doi.org/10.1016/j.inffus.2020.06.014.

[11] S. Reed, Z. Akata, X. Yan, L. Logeswaran, B. Schiele, and H. Lee, “Generative Adversarial Text to Image Synthesis,” in Proceedings of The 33rd International Conference on Machine Learning, 2016, pp. 1060-1069.

[12] P. Isola, J. Zhu, T. Zhou, and A. A. Efros, "Image-to-Image Translation with Conditional Adversarial Networks," *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017, pp. 5967-5976. https://doi.org/10.1109/CVPR.2017.632 .

[13] J.-Y. Liu, Y.-H. Chen, Y.-C. Yeh, and Y.-H. Yang, “Unconditional Audio Generation with Generative Adversarial Networks and Cycle Regularization,” in *arXiv e-prints*, 2020. [Online]. Available: https://arxiv.org/pdf/2005.08526.pdf . Accessed on: February 2, 2022.

[14] M. Bińkowski et al., “High Fidelity Speech Synthesis with Adversarial Networks,” in *arXiv e-prints*, 2019. [Online]. Available: https://arxiv.org/pdf/1909.11646v2.pdf . Accessed on: February 3, 2022.

[15] M. Pasini, “MelGAN-VC: Voice Conversion and Audio Style Transfer on arbitrarily long samples using Spectrograms,” in *arXiv e-prints*, 2019. [Online]. Available: https://arxiv.org/pdf/1910.03713.pdf . Accessed on: February 2, 2022.

[16] *Voice Cloning for Content Creators*. [Online]. Available: https://www.respeecher.com/. Accessed on: February 2, 2022.

[17] D. Croce, G. Castellucci, and R. Basili, “GAN-BERT: Generative Adversarial Learning for Robust Text Classification with a Bunch of Labeled Examples,” in *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, 2020, pp. 2114–2119. https://doi.org/10.18653/v1/2020.acl-main.191 .

[18] L. Liu, Y. Lu, M. Yang, Q. Qu, J. Zhu, and H. Li, “Generative Adversarial Network for Abstractive Text Summarization,” in *arXiv e-prints*, 2017. [Online]. Available: https://arxiv.org/pdf/1711.09357.pdf . Accessed on: February 3, 2022.

[19] H. Zenati, C. S. Foo, B. Lecouat, G. Manek, and V. R. Chandrasekhar, “Efficient GAN-Based Anomaly Detection,” in *arXiv e-prints*, 2018. [Online]. Available: https://arxiv.org/pdf/1802.06222.pdf . Accessed on: February 3, 2022.

[20] S. Mukherjee, H. Asnani, E. Lin, and S. Kannan, “ClusterGAN: Latent Space Clustering in Generative Adversarial Networks,” in *arXiv e-prints*, 2018. [Online]. Available: https://arxiv.org/pdf/1809.03627.pdf . Accessed on: February 3, 2022.

[21] M. Kramer, “Nonlinear principal component analysis using autoassociative neural networks,” in *AIChE*, vol.37, no. 2, pp. 233-243, February, 1991. https://doi.org/10.1002/aic.690370209 .

[22] D. Kingma, and M. Welling, “Auto-Encoding Variational Bayes,” in *arXiv e-prints*, 2013. [Online]. Available: https://arxiv.org/pdf/1312.6114.pdf . Accessed on: February 2, 2022.

[23] I. Goodfellow et al., “Generative adversarial networks,” in *Communications of the ACM*, vol. 63, no. 11, pp. 139-144, November. 2020. https://doi.org/10.1145/3422622.

[24] A. Radford, L. Metz, and S. Chintala, “Unsupervised Representation Learning with Deep Convolutional Generative Adversarial Networks,” in *arXiv e-prints*, 2015. [Online]. Available: https://arxiv.org/pdf/1511.06434.pdf . Accessed on: February 2, 2022.

[25] M. Arjovsky, S. Chintala; and L. Bottou, “Wasserstein Generative Adversarial Networks,” in *Proceedings of the 34th International Conference on Machine Learning*, 2017, pp. 214-223.

[26] I. Gulrajani, F. Ahmed, M. Arjovsky, V. Dumoulin, and A. Courville, “Improved training of wasserstein GANs,” in *Proceedings of the 31st International Conference on Neural Information Processing Systems (NIPS’17)*, 2017, pp. 5769-5779.

[27] A. Brock, J. Donahue, and K. Simonyan, “Large Scale GAN Training for High Fidelity Natural Image Synthesis,” in *arXiv e-prints*, 2018. [Online]. Available: https://arxiv.org/pdf/1809.11096.pdf . Accessed on: February 2, 2022.

[28] T. Karras, S. Laine, and T. Aila, “A Style-Based Generator Architecture for Generative Adversarial Networks,” *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019, pp. 4396-4405. http://doi.org/10.1109/CVPR.2019.00453 .

[29] T. Karras, S. Laine, M. Aittala, J. Hellsten, J. Lehtinen, and T. Aila, “Analyzing and Improving the Image Quality of StyleGAN,” *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020, pp. 8107-8116. http://doi.org/10.1109/CVPR42600.2020.00813 .

[30] A. Odena, V. Dumoulin, and C. Olah, “Deconvolution and Checkerboard Artifacts,” *Distill*, October, 2016. http://doi.org/10.23915/distill.00003 .

[31] *This Flower Does Not Exist*. [Online]. Available: https://thisflowerdoesnotexist.herokuapp.com/ . Accessed on: February 2, 2022.

Рекомендована кафедрою кафедри системного аналізу та інформаційних технологій ВНТУ

Стаття надійшла 16.02.2022

Ісаєнков Ярослав Олександрович — аспірант кафедри системного аналізу та інформаційних технологій, e-mail: oiuygl@gmail.com ;

Мокін Олександр Борисович — д-р техн. наук, професор, професор кафедри системного аналізу та інформаційних технологій, e-mail: abmokin@gmail.com .

Вінницький національний технічний університет, Вінниця

Ya. O. Isaienkov[1]
O. B. Mokin[1]

Analysis of Generative Deep Learning Models and Features of Their Implementation on the Example of WGAN

[1]Vinnytsia National Technical University

The paper presents architecture features, the learning process, and the scope of generative deep learning models. The main tasks of such models include data generation (images, music, texts, videos), transferring styles from one data to another, improving data quality, data clustering, anomaly detection, etc. It is noted that the results of generative models are commonly used for entertainment purposes. In addition, they can be used as data for learning other machine learning models, sources of new ideas for creative professions, tools for anonymization of sensitive data, etc. The article analyzes the advantages and disadvantages of basic generative models like autoencoders, variational autoencoders, generative adversarial networks (GAN), Wasserstein GAN (WGAN), StyleGAN, StyleGAN2, and BigGAN. The paper also describes a step-by-step study of the generative model implementation on the example of WGAN, which includes the basic architecture implementation and more complex elements. Examples of such elements are the introduction of conditional generation to add the ability to select the desired class and the algorithm of bilinear sampling to solve the problem of the so-called 'checkerboard effect'. The final model, created as a result of the study and named CWGAN-GP_128, is capable of generating realistic images of dandelions and marigolds at a resolution of 128x128 pixels. The model learned on the authors' data set consists of 900 photos (450 for each class). The learning process includes affine transformations such as rotations and inversions to augment the images. It is emphasized that although the results of generative models are often easy to evaluate visually, along with the rapid progress of GAN, the problem of automating the process of checking the quality of generated data is growing. The final model is open for public access, and the results are accessible on the authors' website thisflowerdoesnotexist.herokuapp.com.

Keywords: data generation, generative adversarial network, autoencoder, deep learning, GAN, WGAN.

Isaienkov Yaroslav O. — Post-Graduate Student of the Chair of System Analysis and Information Technologies, e-mail: oiuygl@gmail.com ;

***Mokin Oleksandr** B.* — Dr. Sc. (Eng.), Professor, Professor of the Chair of System Analysis and Information Technologies, e-mail: abmokin@gmail.com