**Панчак Дмитро Вікторович,**
*магістр спеціальності 122 "Комп'ютерні науки"*
*Західноукраїнський національний університет*

*Науковий керівник:*
***Коран Володимир Володимирович,***
*кандидат технічних наук, професор*
*Західноукраїнський національний університет*

## Метод генерування зображень за допомогою GAN-мереж

Алгоритми GAN виникли в 2014 році [1] і з тих пір були виділені як потенційні альтернативи для збільшення даних і відсутніх проблем з даними, серед іншого, завдяки їх видатним можливостям генерувати реалістичні екземпляри даних, особливо зображення.

На сьогоднішній день багато дослідників продовжують вивчати свої можливості і розширювати свої області потенціального застосування, що дає позитивний погляд на цю технологію. Зокрема, до цього проєкту призвело порушене питання, в якому йдеться про доцільність використання систем GAN для генерації підроблених даних, не обов'язково зображень, що імітують атрибути приватного набору даних. Якщо можливо, ця згенерована машина буде дуже корисним інструментом, оскільки вона дозволить необмежену кількість подібних до вихідних даних без шкоди для конфіденційності оригінальних елементів, уникаючи будь-якого потенційного ризику витоку конфіденційної інформації. Застосування цього інструменту, як це буде видно пізніше, може варіюватися від освітніх цілей до наукових симуляцій та досліджень, оскільки конфіденційні дані з будь-якої галузі можуть бути доступні без ризику витоку приватних даних.

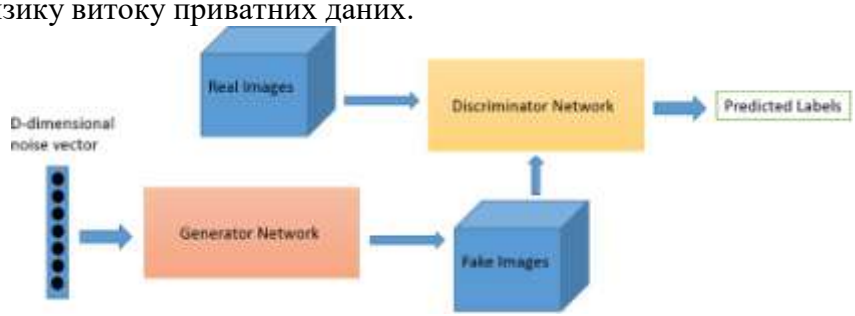


Рис. 1: Базова архітектура GAN.

Для того щоб відповісти на питання, виставлене в попередніх абзацах, в першу чергу необхідно було б зрозуміти, як працює GAN. Далі буде здійснено ретельне перерахування загальних і конкретних цілей проєкту.

Генеративні змагальні мережі (GAN) - це системи, засновані на стратегії min-max, де стикаються два алгоритми: один алгоритм генерує дані (генератор), а інший дискримінує підроблені та реальні дані (дискримінатор).

Мета генератора полягає в тому, щоб максимізувати помилку дискримінатора, тоді як дискримінатор хоче її мінімізувати.

### СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley,Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In Z. Ghahramani, M. Welling, C. Cortes, N. D. Lawrence, and K. Q. Weinberger, editors, Advances in Neural Information Processing Systems 27, pages 2672– 2680. Curran Associates, Inc., 2014. URL http://papers.nips.cc/paper/5423-generative-adversarial-nets.pdf.
2. Patrick L. Paper review on generative adversarial network(gan) part 1. Medium,2017. URL https://medium.com/@patrickhk/paper-review-on-generativeadversarial-network-gan-part-1-48597bcc96df.

Це ітераційний процес, який закінчується, коли помилка дискримінатора становить 0,5, що означає, що вона не вдається 50% разів, базова помилка в бікласифікації. Ми можемо думати про GAN як про «Гру в кішки та мишки» між поліцейським та грошовим фальшивомонетником [2], де фальшивомонетник (генератор) намагається обдурити поліцейського (дискримінатора), створюючи нескінченну петлю, де обидва гравці продовжують вдосконалюватися чистою конкуренцією. На рисунку 1 представлена базова система GAN.

Як описано, для того, щоб генерувати фальшиве зображення, нам завжди потрібне джерело «творчості», яке в даному випадку походить від випадкового вектора шуму (seed). З іншого боку, для того, щоб мати можливість розрізняти реальні та підроблені зображення (щоб модель дискримінатора могла надсилати до моделі генератора, що робить не так), необхідна база даних реальних зображень.

Отже, цільовою функцією повної мережі є наступне:

$$\min_G \max_D V(D,G) = E_{x\sim P_{data}(x)}[\log D(x)] + E_{z\sim P_z(z)}[\log(1 - D(G(z)))] \qquad (1.1)$$

Цей вираз представляє значення (V ), яке є функцією як дискримінатора D, так і генератора G. Мета полягає в тому, щоб максимізувати втрати дискримінатора (D) і мінімізувати втрати генератора (G). Значення V - це сума очікуваної ймовірності журналу для реальних і згенерованих даних. Імовірності (ймовірності) є дискримінаційними виходами для реальних або породжених ім- віків. Зверніть увагу, що вихід дискримінатора для згенерованого зображення віднімається від 1 перед тим, як взяти журнал. Максимізація отриманих значень призводить до оптимізації параметрів дискримінатора таким чином, щоб він навчився правильно ідентифікувати як реальні, так і підроблені дані.

Також необхідно пояснити, що:

Pdata: представляє розподіл реальних даних.

Pz: Представляє розподіл шуму (зазвичай це розподіл Гауса), з якого ми можемо створити підроблене зображення.

x і z: представляють зразки з кожного відповідного простору.

Ex та Ez: Представляють очікувану ймовірність журналу з різних виходів як реальних, так і згенерованих зображень.

Функція D виводить дійсне число, коливається між 0 і 1, що представляє імовірність для даних, що є реальними (1) або підробленими (0). З іншого боку, функція G виводить згенерований зразок або екземпляр.

Крім того, для того, щоб навчити генератора та дискримінатора, помилки на їх виходах поширюються назад у моделі. Ці помилки поширюються як градієнти наступних функцій втрати.

Правило оновлення для дискримінатора:

$$\nabla_{\theta_d}\frac{1}{m}\sum_{i=1}^{m}[\log D(x^{(i)}) + \log(1 - D(G(z^{(i)})))] \qquad (1.2)$$

Правило оновлення для генератора

$$\nabla_{\theta_g}\frac{1}{m}\sum_{i=1}^{m}\log(1 - D(G(z^{(i)}))) \qquad (1.3)$$

де m представляє загальну кількість зразків, протестованих в партії перед оновленням обох моделей, а θ d і θg - ваги кожної моделі. Варто зазначити, що в наступних розділах цього проекту ми будемо посилатися на ці помилки, які ми розраховуємо градієнт як втрати.

Оскільки GAN - це всього лише системна структура, вибір елементів для складання цієї системи (генератора і дискримінатора) залежить від користувача. У цьому випадку буде використовуватися найпопулярніший варіант: використання згорткової нейронної мережі (CNN) для дискримінатора і транспонованого CNN для генератора. Для того, щоб цей проект був зосереджений на структурі GAN, CNN не будуть детально пояснюватися.

Тепер, коли було описано призначення, елементи та цільову функцію, будуть перераховані етапи тренувального процесу. Кожна петля цього тренувального процесу називається епохою. Усередині цієї епохи всі реальні зразки, як правило, доступні зображення, будуть оброблятися партіями: кожна партія є випадковою підмножиною фіксованого розміру (називається розміром партії) з набору даних реальних зразків. Після кожної партії помилки від невідповідності між виходами дискримінатора та генератора та їх очікуваними значеннями знову поширюються на моделі і, отже, використовуються для навчання моделей. Кроки всередині кожної партії:

Пакетні реальні образи підживлюють їх у дискримінатор.

Помилки з виводу дискримінатора розраховуються, знаючи, що в ідеалі ці виходи повинні бути всі 1, оскільки всі ці зображення реальні.

Пакетні підроблені зображення генеруються з пакетними векторами шуму.

Пакетні підроблені зображення подаються в дискримінатор.

Помилки з виводу дискримінатора розраховуються, знаючи, що в ідеалі ці виходи повинні бути всі 0, оскільки всі ці зображення є підробленими.

Обидві помилки, розраховані у дискримінатора, тепер поширюються на його модель.

Кроки 3 і 4 повторюються з новими векторами шуму.

Цього разу помилки розраховуються для генератора, знаючи, що в ідеалі всі виходи з дискримінатора повинні бути 1, оскільки ці зображення повинні ідеально імітувати реальні .

Ці помилки тепер поширюються на модель генератора.

Почніть знову з чергової партії з цієї епохи.

Як буде показано в наступному розділі, ці системи можуть збільшити свою складність, додавши нові елементи і функції в основну структуру. Найбільш поширеним додатковим елементом енкодер (зазвичай інший CNN) для живлення генератора не випадковим вектором шуму, а іншим джерелом інформації, що дозволяє системі генерувати реальні зображення, наприклад, з описового тексту або іншого зображення.

Ще одним фактором, який додає складності системі, є необхідність тонкої настройки конфігурації для різних елементів GAN з метою полегшення процесу її навчання та оптимізації.