Issue 15, 2024

https://doi.org/10.23939/sisn2024.15.015

УДК 004.054

АРХІТЕКТУРА ТА ФОРМАЛЬНО-МАТЕМАТИЧНЕ ОБҐРУНТУВАННЯ ГЕНЕРАТИВНИХ ЗМАГАЛЬНИХ МЕРЕЖ

Віталій Прозур

НТУУ "Київський політехнічний інститут імені Ігоря Сікорського",
кафедра штучного інтелекту, Київ, Україна
E-mail: vitaliy.prozur@gmail.com, ORCID: https://orcid.org/0009-0000-2996-483X

Мета статті – аналіз особливостей генеративних змагальних мереж. Об'єкт дослідження – процес алгоритмізації машинного навчання. Предмет дослідження – математичні методи, використовувані в генерації семантично пов'язаного тексту. У статті досліджено архітектуру та математичне обґрунтування такого виду генеративних моделей, як генеративні змагальні мережі. Генеративні змагальні мережі є потужним інструментом у галузі штучного інтелекту, який здатен створювати реалістичні дані, зокрема такі як фото, відео, звук тощо. Архітектура генеративних змагальних визначає її структуру, взаємодію компонентів та загальний опис процесу навчання. Математичне обґрунтування охоплює теоретичний аналіз принципів, алгоритмів та функцій, що є основою цих мереж.

У статті розглянуто загальну архітектуру генеративних змагальних мереж, кожен її компонент (а саме дві основні моделі-мережі – генератор та дискримінатор, їх вхідні та вихідні вектори даних) та його роль у роботі алгоритму. Визначено також математичні засади генеративних змагальних мереж, із зосередженням на теорії ігор та оптимізаційних методах (зокрема особливу увагу звернено на задачі мінімаксу та максиміну, гру із нульовою сумою, сідлові точки, рівновагу Неша), що використовують в їх навчанні. Описано функцію витрат та її виведення за допомогою рівноваги Неша в грі з нульовою сумою для генеративних змагальних мереж, а також розглянуто алгоритм навчання із використанням методу стохастичного градієнтного спуску та міні-пакетного підходу у вигляді псевдокоду, його ітерації, візуалізовано перебіг процесу навчання цієї архітектури мережі.

Зрештою, обґрунтовано висновок, що генеративні змагальні мережі – це ефективний інструмент для створення реалістичних та правдоподібних зразків даних, що ґрунтується на використанні елементів теорії ігор. Завдяки високій якості згенерованих даних генеративні змагальні мережі можна використовувати у різних сферах, зокрема й таких, як кібербезпека, медицина, торгівля, наука, мистецтво тощо.

Ключові слова: генеративні змагальні мережі; генератор; дискримінатор; мінімакс; гра із нульовою сумою; рівновага Неша.

Вступ

Генеративні змагальні мережі (GAN) – це архітектура для навчання генеративної моделі (тобто такої, що створює нові зразки даних). Сучасну ідею та опис загальної моделі під назвою GAN виклав у 1990-ті роки Юрген Шмідхубер у статтях [13–14]. У 2014 р. Ян Гудфеллоу ввів термін GAN і популяризував цей тип моделі після опублікування статті Generative Adversarial Nets [6]. Нині GAN використовуються у різних сферах і здатні на такі речі, як-от, наприклад, генерування фотографій людей, яких насправді немає, чи передбачення наступного кадру у відеоряді.

Генеративні змагальні мережі – це тип глибокої нейронної мережі, яка досягла багатьох найсучасніших результатів для генеративних завдань. GAN замінюють старі методи в деяких проблемах і встановлюють найсучасніші характеристики; GAN відкривають нові межі в проблемах, якими раніше не займалися; GAN можуть бути застосовані до різних масштабів; використовуються в різних проблемах і типах даних. Загалом GAN підтримує 26 унікальних доменів додатків, проте однією загальною проблемою у цій галузі нині є відсутність високоякісних наборів даних. Одержавши більше даних у майбутньому, GAN може дати ще кращі результати.

Постановка проблеми

Генеративні змагальні мережі є різновидом алгоритму штучного інтелекту, призначеного для вирішення проблеми генеративного моделювання. Генеративні змагальні мережі – це підхід до генеративного моделювання із використанням методів глибокого навчання, таких як згорткові нейронні мережі.

Генеративне моделювання – це створення спрощеної абстрактної копії реального об'єкта, системи або явища на підставі вирішення завдання навчання без вчителя в машинному навчанні, що передбачає автоматичне виявлення та вивчення закономірностей у заданих даних так, щоб модель можна було використовувати для створення або виведення нових прикладів, взятих із вихідного набору даних.

Метою генеративної моделі є вивчення колекції навчальних прикладів і визначення розподілу ймовірностей. Тоді генеративні змагальні мережі можуть генерувати більше прикладів із оціненого розподілу ймовірностей. Генеративні моделі, основані на глибокому навчанні, поширені, але GAN є одними з найуспішніших генеративних моделей (особливо з погляду їхньої здатності генерувати реалістичні зображення із високою роздільною здатністю). GAN були успішно застосовані для широкого спектра завдань (переважно в дослідницьких умовах), але продовжують створювати унікальні виклики та дослідницькі можливості, оскільки ґрунтуються на теорії ігор, тоді як більшість інших підходів до генеративного моделювання – на оптимізації.

Протягом тривалого часу однією з основних проблем алгоритмів машинного навчання з учителем (англ. "supervised learning") є проблема даних. Зокрема, для навчання нейромереж часто необхідні вибірки із мільйонів елементів. Якщо розглянути як приклад навчання розпізнавання зображень, то необхідно мільйони зображень належної якості, які повинні бути розмічені, що потребує ручної праці великої кількості людей. Тому непрямі витрати на навчання нейромережі з розпізнавання зображень колосальні. Виходом міг стати алгоритм, здатний без участі людини генерувати такі вибірки.

Аналіз останніх досліджень та публікацій

Розвиток глибокого навчання полягає у відкритті насичених ієрархічних моделей [2], які є розподілами ймовірностей за типами даних, які трапляються у додатках штучного інтелекту, таких як природні зображення, аудіосигнали. Найбільш вражає успіх у глибокому навчанні, зумовлений дискримінаційними моделями [7, 10]. Це пов'язано насамперед з алгоритмами зворотного поширення та виключення з використанням кусково-лінійних одиниць [3, 5, 8]. Глибокі генеративні моделі не набули поширення через труднощі апроксимації багатьох складних імовірнісних обчислень, які виникають під час оцінювання максимальної ймовірності та у пов'язаних стратегіях, а також через труднощі використання переваги кусково-лінійних одиниць у генеративному контексті.

Останні кілька років публікації з GAN все частіше стосуються використання міток класів. У роботі [11] запропоновано умовний варіант генеративно-змагальної мережі (conditional-class model), де мітку класу використано як на вході генератора, так і на виході дискримінатора. Таку мережу застосовують для вирішення завдання тегування зображень та генерації мультимодальних прикладів. Перша мультикласова стратегія виведення для GAN була розроблена в [16], де кількість виходів дискримінантного класифікатора дорівнює кількості класів, а навчання здійснюється як на немаркованих, так і на частково маркованих даних. Така мережа одержала назву категоріальної генеративно-змагальної мережі (CatGAN).

Виклад основного матеріалу

Ян Дж. Гудфеллоу представив генеративні змагальні мережі як алгоритм, що складається з двох моделей, які прийнято називати генератором та дискримінатором [6].

Генератор – модель, що створює (генерує) нові зразки даних, а дискримінатор – модель, яку використовують для допомоги в навчанні генератора, оцінюючи справжність згенерованих зразків даних. У "класичній" архітектурі генеративних змагальних мереж моделі генератора та дискримінатора були реалізовані як багатошарові перцептрони, проте з часом почали широко використовувати різні види реалізації, як-от згорткові нейронні мережі та інші.

Мета генеративних змагальних мереж – навчитися створювати дублікати даних, максимально схожі на відомі вхідні дані. Основна ідея полягає в тому, щоб взяти дві нейронні мережі (генератор та дискримінатор) та поставити їм протилежну ціль. Для генератора – намагатися створити ідеальні "фальшиві" дані, а для дискримінатора – спробувати знайти дефект у даних, створених генератором. Це приводить до того, що генератор підвищує точність і створює все кращі зразки даних. Власне, таку ситуацію можна змоделювати в теорії ігор як мінімаксну гру.

Весь процес побудований на тому, що генератор намагається обдурити дискримінатор, тоді як дискримінатор намагається уникнути обману. У міру навчання моделей обидва методи вдосконалюються до моменту, коли штучно створені зображення не можна буде відрізнити від справжніх. Це і є кінцева мета навчання. Архітектуру GAN наведено на рис. 1.

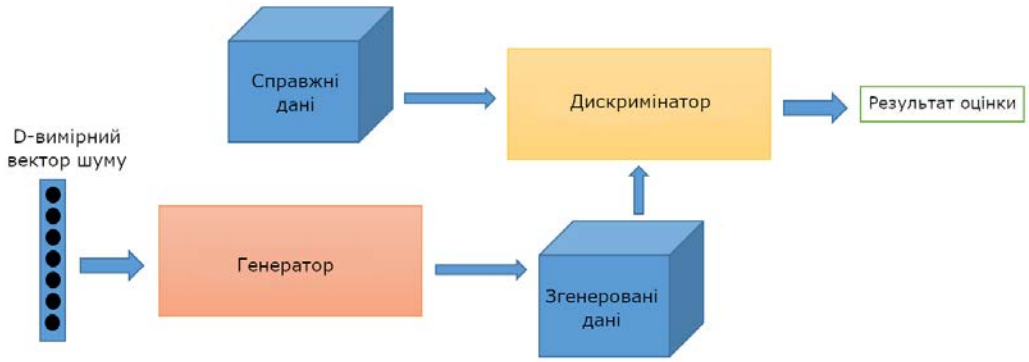

Рис. 1. Архітектура генеративних змагальних мереж

Джерело: [15].

Головна ідея генеративних змагальних мереж зосереджена навколо конфлікту двох компонентів системи, саме тому в обґрунтуванні роботи генеративних змагальних мереж велику роль виконує теорія ігор, оскільки вона вивчає математичні моделі для прийняття оптимальних рішень в умовах конфлікту. Власне, увесь процес можна змоделювати як мінімаксну гру, що детальніше висвітлено далі.

Задачі мінімаксу та максиміну

В іграх з декількома учасниками кожен із них прагне максимізувати власну вигоду та збільшити шанси на виграш. Якщо ми розглядаємо гру з N учасниками, то оптимальна стратегія *i*-го гравця полягає в тому, щоб отримати максимальний виграш, за умови, що інші гравці грають так, щоб йому завадити. Стратегія гравця *i*, за якої його мінімальний виграш є найбільшим з усіх можливих, незалежно від дій всіх інших гравців, називається максимінною стратегією, а найбільший з мінімальних виграшів – максимінним значенням. Якщо гравець *i* дотримується максимінної стратегії, його виграш буде не меншим від максимінного значення. Отже, максимінну стратегію s_i* і максимінне значення L_i* можна записати як:

$$s_i^* = \operatorname{argmax} L_i(s_i, s_{-i}), \quad (1)$$

$$L_i^* = \operatorname{maxmin} L_i(s_i, s_{-i}), \quad (2)$$

Максимінну стратегію гравця *i* легше інтерпретувати, якщо вважати, що йому відомі ходи противника і він знає, що противник намагатиметься мінізувати його найбільший виграш на кожному ході. За цього припущення гравець *i* повинен робити хід, що принесе максимальний з усіх мінімальних виграшів.

Відповідно до мінімаксної стратегії гравець *i* повинен припускати, що його суперники, позначені як *-i*, дозволятимуть йому отримати лише мінімальний виграш з усіх максимальних на кожному ході. За мінімаксної стратегії виграш гравця записують так:

$$L_i^* = \text{minmax}\, L_i(s_i, s_{-i}), \tag{3}$$

Варто зазначити, що кінцеві виграші гравця можуть відрізнятися від мінімаксного чи максимінного значення.

Гра з нульовою сумою, мінімакс та сідлові точки

В теорії ігор гра з нульовою сумою (*zero-sum game*) – це математичне формулювання ситуації, у якій виграші чи програші одного учасника дорівнюють програшам чи виграшам іншого. Тому для системи загалом чистий виграш чи програш її учасників дорівнює нулю.

Для мінімаксних задач в іграх з нульовою сумою, до яких залучені два гравці А та В, виграш $U(x, y)$ для гравця A можна записати як:

$$\widehat{U} = minmax\ U(x, y), \tag{4}$$

де x – хід гравця А; y – хід гравця В.

Крім того значення x і y, що відповідають $\widehat{U}$, являють собою рівнозважені стратегії гравців А та В відповідно, тобто гравці не змінять ходи, якщо продовжать дотримуватися мінімаксної чи максимінної стратегії. Для гри з нульовою сумою і двома учасниками мінімаксна і максимінна стратегії будуть давати однакові результати, а отже, ця рівновага правильна, якщо гравці використовують мінімаксну чи максимінну стратегії. І оскільки мінімаксні та максимінні значення однакові, то порядок визначення мінімаксної чи максимінної стратегії не має значення. Можна з однаковим успіхом дозволити гравцям А та В незалежно вибирати свої найкращі стратегії для кожної стратегії противника, і тоді ми б побачили, що для гри з нульовою сумою одна з комбінацій стратегій перекриватиметься. Ця умова перекривання – найкраща стратегія для обох гравців, ідентична мінімаксній стратегії. Цю умову також називають рівновагою Неша.

Нехай далі x – хід гравця А, а y – хід гравця В – неперервні змінні, а $f(x, y)$ – функція вигоди гравця. Нам треба знайти точку рівноваги, яка буде мінімаксом чи максиміном функції вигоди для будь-якого гравця. Оскільки для гри з нульовою сумою для двох учасників мінімакс і максимін збігаються, то порядок значення не має, тобто

$$\text{minmax}\, f(x, y) = \text{maxmin}\, f(x, y), \tag{5}$$

Для неперервної функції це можливо лише в тому випадку, якщо розв'язок попередньої буде сідловою точкою. Сідлова точка – це точка, в якій градієнт за кожною змінною дорівнює нулю. Проте вона не є локальним мінімумом чи максимумом. Натомість вона представляє локальний мінімум в одних напрямках вхідного вектора і локальний максимум у других. Для багатовимірної функції $f(x)$ *c* $\forall x \in R^{nx1}$ можемо визначити сідлову точку за допомогою тесту.

Підраховуємо градієнт $f(x)$ за вектором x і прирівнюємо його до нуля.

Обчислимо гесіан $\nabla_x^2 f(x)$ функції $f(x)$, тобто матрицю похідних другого порядку в кожній з точок, в яких вектор градієнта $\nabla_x f(x)$ дорівнює нулю. Якщо гесіан має як додатні, так і від'ємні значення в оцінюваній точці, то ця точка – сідлова.

Повертаючись до функції вигоди двох змінних $f(x, y)$, для гравця А визначимо її, щоб навести приклад:

$$f(x, y) = x^2 - y^2. \tag{6}$$

Відповідно функція вигоди для гравця В буде:

$$f(x, y) = -x^2 + y^2, \tag{7}$$

Перевіримо, чи забезпечує ця функція вигоди рівновагу, якщо обидва гравці дотримуються мінімаксної чи максимінної стратегії в грі з нульовою сумою. У грі буде рівновага, поза якою гравці

не зможуть покращити свої виграші, оскільки їхні стратегії оптимальні, якщо функція $f(x, y)$, має сідлову точку. Умова рівноваги – це рівновага Неша для гри.

Прирівнюючи градієнт $f(x, y)$ до нуля, отримуємо

$$\nabla f(x,y) = \left[\frac{\partial f}{\partial x}\ \frac{\partial f}{\partial y}\right] = [2x\ -2y\ ] = 0\ \Rightarrow (0,0). \quad (8)$$

Підрахуємо гесіан для цієї функції:

$$\nabla^2 f(x,y) = \left[\frac{\partial^2 f}{\partial x^2}\ \frac{\partial^2 f}{\partial x \partial y}\ \frac{\partial^2 f}{\partial y x}\ \frac{\partial^2 f}{\partial y^2}\right] = [2,0,0,-2\ ]. \quad (9)$$

Гесіан функції дорівнює $[2,0,0,-2\ ]$ для будь якого значення (x, y), ураховуючи значення $(x, y) = (0,0)$. Оскільки гесіан має як додатні, так і від'ємні значення (2 і –2), точка (x, y) = (0,0) – сідлова. В умовах рівноваги стратегія для гравця А повинна полягати в тому, щоб встановити значення $x = 0$, а для В – у тому, щоб встановити $y = 0$ в мінімаксній чи максимінній грі з нульовою сумою.

Функція вартості та тренування мережі GAN

В GAN мережі генератора і дискримінатора намагаються перемогти в мінімаксній грі з нульовою сумою. В цьому випадку ходи – це параметри, які вибирає мережа. Нехай G – параметри генератора, а D – параметри дискримінатора. Розглянемо функції вигоди. Дискримінатор буде намагатись якомога коректніше класифікувати як реальні, так і згенеровані дані, тобто намагатиметься максимізувати функцію вигоди:

$$U(D,G) = E_{x\sim P_x(x)}[log\ D(x)\ ] + E_{z\sim P_z(z)}[log\ (1 - D(G(z))\ ], \quad (10)$$

де x – реальні дані, з розподілу $P_x(x)$, а z – шум, з апріорно зашумленого розподілу $P_z(z)$.

Крім того, дискримінатор намагається вивести 1 для реального зразка даних x і 0 для даних, створених генератором. Отже, дискримінатор хоче дотримуватися стратегії, яка наближає $D(x)$ до 1. Що ближче $D(x)$ до 1, то менша вигода, яку отримає дискримінатор. Точно так само дискримінатор хоче досягти 0 ймовірності для згенерованих даних, тобто $D(G(x))$ буде якнайближче до 0. Вплив генератора на дискримінатор бачимо в другому члені рівняння, тобто $G(z)$, тому генератор буде намагатись мінімізувати виграш дискримінатора. Що більший виграш у дискримінатора, то гірша ситуація для генератора. Тому можемо вважати, що генератор має ту саму функцію виграшу, тільки з від'ємним знаком, що перетворює цю гру на гру з нульовою сумою. Функція виграшу генератора:

$$V(D,G) = -E_{x\sim P_x(x)}[logD(x)] - E_{z\ \sim\ P_z(z)}[log(1 - D(G(z))]. \quad (11)$$

Генератор буде намагатись максимізувати свою функцію вигоди $V(D,G)$, інакше кажучи, намагатиметься генерувати такі зразки даних, які одурений дискримінатор сприйме за справжні, призначивши їм високу ймовірність. Високі значення $D(G(z))$ приведуть до того, що $log(1 - D(G(z)))$ матиме велике від'ємне значення, тобто вираз $-E_{z\ \sim\ P_z(z)}[log(1 - D(G(z)))]$ матиме велике додатне значення, тим самим збільшуючи виграш генератора. Генератор не може впливати на перший член рівняння, оскільки він містить тільки реальні дані.

Моделі генератора та дискримінатора тренують, надаючи їм можливість дотримуватися мінімаксної стратегії в грі з нульовою сумою. Дискримінатор намагатиметься максимізувати свою функцію вигоди і досягти її мінімаксного значення:

$$u^* = \max\min E_{x\sim P_x(x)}[\log D(x)\ ] + E_{z\sim P_z(z)}[\log\ (1 - D(G(z))\ ], \quad (12)$$

А генератор дотримуватиметься тієї самої стратегії:

$$v^* = \max\min -E_{x\sim P_x(x)}[\log D(x)] - E_{z\ \sim\ P_z(z)}\left[\log\left(1 - D\left(G(z)\right)\right)\right], \quad (13)$$

Оскільки генератор ніяк не впливає на перший член рівняння, то:

$$v^* = \max\min -E_{z\ \sim\ P_z(z)}\left[\log\left(1 - D\left(G(z)\right)\right)\right], \quad (14)$$

Розглядаючи формулювання мінімаксної стратегії дискримінатора, отримуємо для значення виграшу дискримінатора за рівноваги Неша:

$$u^* = \min\max E_{x\sim P_x(x)}[\log D(x)\ ] + E_{z\sim P_z(z)}[\log\ (1 - D(G(z)))\ ], \quad (15)$$

Значення $\hat{G}$ та $\hat{D}$ за рівноважного значення u^* – оптимальне значення параметрів для обох мереж, у разі зміни яких жодна із них не збільшить свій виграш. Крім того, це також сідлова точка для функції вигоди дискримінатора.

Попередні формулювання також можна спростити, розділивши оптимізацію на дві частини, тобто дозволити *D* максимізувати свою функцію вигоди за своїми параметрами, а *G* – мінімізувати функцію вигоди *D* за своїми параметрами.

Кожен з них, оптимізуючи власну функцію вигоди, розглядає ходи противника як фіксовані. Такий ітеративний спосіб оптимізації є не чим іншим, як методом градієнтного спуску для обчислення сідлової точки. Для зручності роботи з більшістю програмних пакетів, які заточені під мінімізацію функцій, варто помножити цільову функцію дискримінатора на –1, і мінізувати її, замість максимізації. Нижче наведено псевдокод реалізації GAN із застосуванням мініпакетного підходу:

For N #кількість епох

For K #кроки

Отримати *m* зразків із зашумленого розподілу $z \sim P_z(z)$.

Одержати *m* зразків із розподілу реальних даних $x \sim P_x(x)$.

Оновити параметри дискримінатора, використовуючи стохастичний градієнтний спуск. Якщо позначити параметри дискримінатора як *θd*, то *θd* оновлюються за формулою:

$$\theta_D \rightarrow \theta_D - \theta_{V_D}\left[-\frac{1}{m}\left(\log D\left(x^{(i)}\right)\right) + \log\left(1 - D\left(G\left(z^{(i)}\right)\right)\right)\right], \tag{16}$$

End

Отримати *m* зразків із зашумленого розподілу $z \sim P_z(z)$.

Оновити генератор методом стохастичного градієнта:

$$\theta_G \rightarrow \theta_G - \nabla_{\theta_G}\left[-\frac{1}{m}\sum_{i=1}^{m}\theta \log\left(1 - D\left(G\left(z^{(i)}\right)\right)\right)\right], \tag{17}$$

End

Нижче на графіку наведено приклад, на якому наочно видно, як проходить навчання. Дискримінатор доволі швидко вчиться розрізняти зображення, проте потім його крива починає коливатися, оскільки генератор починає створювати значно кращі зразки даних.

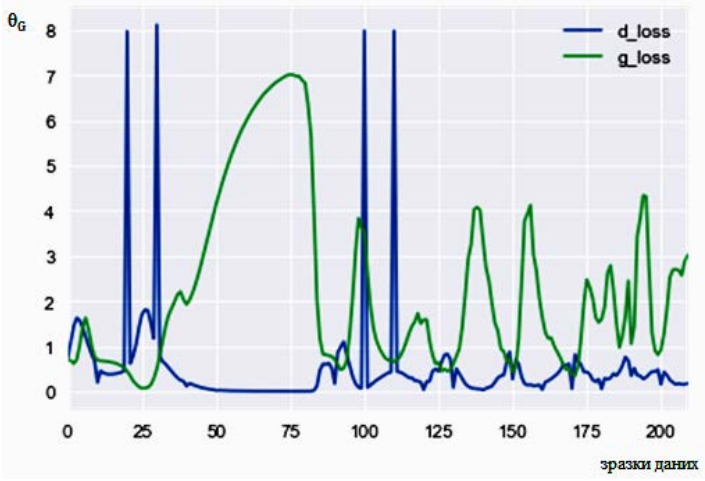

Рис. 2. Процес навчання генеративної змагальної мережі

Джерело: [15].

Висновки

Генеративні змагальні мережі є потужним інструментом у галузі штучного інтелекту, який здатен створювати реалістичні дані, зокрема такі як фото, відео, звук тощо. Архітектура генеративних змагальних мереж визначає її структуру, взаємодію компонентів та загальний опис процесу

навчання. Математичне обґрунтування, своєю чергою, охоплювати теоретичний аналіз принципів, алгоритмів та функцій, покладених в основу цих мереж.

Відомий набір даних є вихідною інформацією для навчання дискримінатора. Його навчання передбачає надання йому зразків із набору навчальних даних доти, доки він досягне прийнятної точності. Генератор тренується залежно від того, чи вдалося обдурити дискримінатор. Зазвичай генератор заповнюють вхідними рандомізованими даними, які вибирають із заздалегідь визначеного прихованого простору (наприклад, багатовимірного нормального розподілу). Після цього кандидатів, синтезованих генератором, оцінює дискримінатор. До обох мереж застосовують незалежні процедури зворотного поширення, так що генератор робить якісніші вибірки, тоді як дискримінатор стає досвідченішим у маркуванні синтетичних вибірок. Під час генерації зображення генератор зазвичай є деконволюційною нейронною мережею, а дискримінатор – крутною нейронною мережею. Мережі GAN часто страждають від "колапсу режиму", коли вони не можуть правильно узагальнити, пропускаючи цілі режими у вхідних даних.

Отже, генеративні змагальні мережі – це дієвий інструмент для створення правдоподібних зразків даних, що ґрунтується на поєднанні нейронних мереж та теорії ігор. За рахунок високої якості згенерованих даних генеративні змагальні мережі можна використовувати в різних галузях, ураховуючи навіть науку та мистецтво, кібербезпеку, медицину, торгівлю.

Список літератури

1. Arjovsky, M., Chintala, S., & Bottou, L. (2017). Wasserstein generative adversarial networks. *In International conference on machine learning (ICML). Proceedings of the 34th International Conference on Machine Learning*, PMLR, 70: 214–223. DOI: https://doi.org/10.48550/arXiv.1701.07875

2. Bengio, Y. (2009). Learning deep architectures for AI. Now Publishers. *Foundations and Trends® in Machine Learning*, 2: 1, 1–127. DOI: http://dx.doi.org/10.1561/2200000006

3. Glorot, X., Bordes, A., and Bengio, Y. (2011). Deep sparse rectifier neural networks. In *Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics*, PMLR, 15: 315–323.

4. Goodfellow, I. et al. (2020). Generative adversarial networks *Communications of the ACM*, November, 63(11), 139–144. DOI: https://doi.org/10.1145/3422622

5. Goodfellow, I. J. *et al*. (2013). Maxout networks. *In ICML'2013 JMLR WCP*, 28 (3): 1319–1327. DOI: .https://doi.org/10.48550/arXiv.1302.4389

6. Goodfellow, I. J. *et al*. (2014) Generative Adversarial Nets. *Proceedings of the 27th International Conference on Neural Information Processing Systems*, 2, 2672–2680 URL: https://arxiv.org/pdf/1406.2661.pdf. DOI: https://doi.org/10.48550/arXiv.1406.2661

7. Hinton, G. et al. (2012). Deep Neural Networks for Acoustic Modeling in Speech Recognition: The Shared Views of Four Research Groups, in *IEEE Signal Processing Magazine*, 29 (6), 82–97, Nov. 2012. DOI: 10.1109/MSP.2012.2205597

8. Jarrett, K. et al. (2009). What is the best multi-stage architecture for object recognition? *IEEE 12th International Conference on Computer Vision*, Kyoto, Japan, 2146–2153. DOI: 10.1109/ICCV.2009.5459469.

9. Knudson, K. C. et al. (2014). *Advances in Neural Information Processing Systems 27*, eds. Z. Ghahramani, M. Welling, C. Cortes, N. D. Lawrence, and K. Q. Weinberger, 1215–1223. URL: https://pillowlab.princeton.edu/pubs/Knudson_COMP_NIPS14.pdf

10. Krizhevsky, A., Sutskever, I., and Hinton, G. (2012). ImageNet classification with deep convolutional neural networks. In *Advances in Neural Information Processing Systems, 25 (NIPS 2012)* Communications of the ACM, (6), 84–90. DOI: https://doi.org/10.1145/3065386

11. Krizhevsky, A., Sutskever, I., and Hinton, G. (2012). ImageNet classification with deep convolutional neural networks. In *Advances in Neural Information Processing Systems, 25 (NIPS 2012)* Communications of the ACM, (6), 84–90. DOI: https://doi.org/10.1145/3065386

12. Mirza, M., Osindero, S. (2014). Conditional generative adversarial Nets Computing Research Repository. URL: https://arxiv.org/abs/1411.1784

13. Radford, A., Metz, L., & Chintala, S. (2015). Unsupervised representation learning with deep convolutional generative adversarial networks. arXiv preprint arXiv:1511.06434. *International Conference on Learning Representations.* DOI: https://doi.org/10.48550/arXiv.1511.06434

14. Schmidhuber, J. (1991). Adaptive confidence and adaptive curiosity. Technical Report FKI-149-91, Inst. f. Informatik, Tech. Univ. Munich, April. URL: https://people.idsia.ch/~juergen/FKI-149-91ocr.pdf

15. Schmidhuber, J. (1992). Learning Factorial Codes by Predictability Minimization. Neural Comput.; 4 (6): 863–879. DOI: https://doi.org/10.1162/neco.1992.4.6.863

16. Shatri, E. A review on Generative Adversarial Networks. How did the GANs change the way machine learning works? URL: HTTPS://TOWARDSDATASCIENCE.COM/A-REVIEW-OF-GENERATIVE-ADVERSARIAL-NETWORKS-9AF21E94BDA4

17. Springenberg, J. T. (2016). Unsupervised and Semi-supervised Learning with Categorical Generative Adversarial Networks. URL: https://arxiv.org/abs/1511.06390

18. Zhu, J. Y. *et al*. (2017). Unpaired image-to-image translation using cycle-consistent adversarial networks. In *Proceedings of the IEEE international conference on computer vision (ICCV)*. DOI: 10.1109/ICCV.2017.244.

ARCHITECTURE AND FORMAL-MATHEMATICAL JUSTIFICATION OF GENERATIVE ADVERSARIAL NETWORKS

Vitalii Prozur

National Technical University of Ukraine “Igor Sikorsky Kyiv Polytechnic Institute”,
Department of Artificial Intelligence, Kyiv, Ukraine
E-mail: vitaliy.prozur@gmail.com, ORCID: https://orcid.org/0009-0000-2996-483X

The purpose of the work is to analyze the features of generative adversarial networks. The object of research is the process of machine learning algorithmization. The subject of the research is mathematical methods used in the generation of semantically related text. This article explores the architecture and mathematical justification of such a type of generative models as generative adversarial networks. Generative adversarial networks are a powerful tool in the field of artificial intelligence, capable of generating realistic data, including photos, videos, sounds, etc. The architecture of generative competition defines its structure, the interaction of components and a general description of the learning process. Mathematical justification, in turn, includes a theoretical analysis of the principles, algorithms and functions underlying these networks.

The article examines the general architecture of generative adversarial networks, examines each of its components (namely, the two main network models – generator and discriminator, their input and output data vectors) and its role in the operation of the algorithm. The author also defined the mathematical principles of generative adversarial networks, focusing on game theory and optimization methods (in particular, special attention is paid to minimax and maximin problems, zero-sum game, saddle points, Nash equilibrium) used in their study. The cost function and the process of deriving it using the Nash equilibrium in a zero-sum game for generative adversarial networks are described, and the learning algorithm using the method of stochastic gradient descent and the mini-batch approach in the form of a pseudocode, its iterations, is visualized network architecture.

Finally, the conclusion that generative adversarial networks is an effective tool for creating realistic and believable data samples based on the use of elements of game theory is substantiated. Due to the high quality of generated data, generative adversarial networks can be used in various fields, including: cyber security, medicine, commerce, science, art, etc.

Key words: Generative Adversarial Networks; generator; discriminator; minimax; zero-sum game; Nash equilibrium.