УДК 004.22+004.89

В.М. ТЕРЕЩЕНКО
Київський національний університет імені Тараса Шевченка, Київ, Україна,
e-mail: *vtereshch@gmail.com*.

П.А. ЗАКАЛА
Київський національний університет імені Тараса Шевченка, Київ, Україна,
e-mail: *pzakala@gmail.com*.

ПОШУК БАЗОВОЇ МНОЖИНИ ДЛЯ ЗАДАЧ МАШИННОГО НАВЧАННЯ

Анотація. Розглянуто задачу пошуку базової множини та три способи її розв'язання: геометричний, із застосуванням генетичного алгоритму та на основі нейронних мереж. Проаналізовано ефективність кожного способу та зроблено висновки про шляхи їхнього використання. Особливу увагу приділено підходам на основі нейронних мереж. Проведено порівняльний аналіз різних підходів на основі нейронних мереж, описано їхні сильні та слабкі сторони, а також визначено подальші кроки для розв'язання задачі пошуку базової множини.

Ключові слова: базова множина, дистиляція даних, конденсація даних, геометрична базова множина, генетичні алгоритми.

ВСТУП

Протягом останніх 20-ти років підходи до роботи з даними зазнали суттєвих змін. Це пов'язано з тим, що ціни на зберігання інформації зменшились, а зберігання великих обсягів даних стало широкодоступним. Ще однією причиною цих змін став розвиток обчислювальних можливостей комп'ютерів. Як наслідок, виникли нові методи оброблення та аналізу даних, зокрема з використанням нейронних мереж.

Нейронні мережі зарекомендували себе як ефективний та універсальний метод. Проте їхньою основною проблемою залишається енергоефективність: навчання нейронних мереж потребує значних обчислювальних ресурсів. Одним зі способів розв'язання цієї проблеми є зменшення розміру вибірки, на якій навчають нейронну мережу. Це зменшить час навчання моделі і тим самим збереже ресурс.

Задача пошуку базової множини є методом зменшення обсягу даних для навчання. Базова множина — це вибірка із загального набору даних, яка максимізує інформацію, що зберігається в цих даних. У поєднанні з нейронними мережами цей підхід не лише пришвидшить процес навчання моделей, а й покращить якість самих даних. Аналізуючи базову множину, фахівець зможе краще зрозуміти природу даних, з якими він працює. Базова множина може стати потужним інструментом для аналізу даних, особливо складних, як-от зображення чи текстова інформація.

1. ПОСТАНОВКА ЗАДАЧІ

Базовою множиною (coreset) деякої множини D називають таку зважену множину C, що розв'язок, отриманий на цій множині, є порівнюваним (provably competitive) з розв'язком, отриманим на D. Для задач, що належать класу «навчання без учителя» (unsupervised learning), запропоновано таке означення базової множини [1].

Означення 1 (базова множина). Нехай задано набір даних D та деяку функцію втрат $cost(D, q)$, де $q \in Q$ — розв'язок. Зважену множину C називають

базовою множиною (*ε-coreset)* набору D, якщо для деякого $\varepsilon > 0$ і для всіх $q \in Q$ справджується така нерівність:

$$| cost(D, q) - cost(C, q) | \le \varepsilon \cdot cost(D, q) . \quad (1)$$

Можна виділити два основні критерії, яким повинна відповідати базова множина:

• базова множина має бути найкращою репрезентацією набору даних, на основі якого вона побудована;

• результат навчання моделі на базовій множині не повинен залежати від вибраної моделі. Оскільки множина є репрезентацією всього набору даних, якість навчання має залежати лише від якості моделі, а не від вибраної вибірки.

Слід також розуміти, що будь-який елемент даних містить ознаки, які описують як загальні патерни, що містяться в усьому наборі, так і часткові, притаманні тільки цьому елементу. Для коректної репрезентації даних потрібно, щоби базова множина містила ознаки як однієї, так і другої категорії.

Під час знаходження базового набору даних треба знайти баланс між такими двома параметрами:

• розмір базового набору даних;

• інформативність базового набору даних.

Знайшовши таку множину, яка мінімізуватиме перший параметр та максимізуватиме другий, можна найбільш ефективно виокремити інформацію з набору даних та використати її для навчання.

2. ПІДХОДИ НА ОСНОВІ МАШИННОГО НАВЧАННЯ

2.1. Пошук геометричної базової множини. Суть задачі є такою: маючи деякий набір з n точок $P \in R^d$, знайти таку підмножину $C \subset R^d$ цього набору, яка для заданої задачі Q дасть такий результат:

$$\mu(C) \ge (1 - \varepsilon)\mu(P), \quad (2)$$

де μ — верхня межа метрики якості, яку вдасться отримати, розв'язуючи задачу на наборі даних, $0 < \varepsilon < 1$ [1].

Головною особливістю цього підходу є представлення набору даних як множини геометричних об'єктів у просторі за допомогою точок. Підходи, які розробляють для пошуку геометричної базової множини, можна розділити на дві групи:

1) підхід до пошуку точної базової множини;

2) підхід до пошуку наближеної базової множини.

Нехай маємо деяку зважену множину точок $P' = (P, w)$ та простір запитів $\{P, w, X, f, loss\}$, де $P \in R^d$ — простір можливих точок, $w \in [0, 1]$ — ваги точок, X — набір можливих запитів, $f : P \times X \to [0, \infty]$ — функція вартості запиту в точці, *loss* — функція зваженої сумаризації вартостей. Зважену множину $C' = (C, w)$ називають точною базовою множиною, якщо для будь-якого $x \in X$ маємо таку рівність [2]:

$$loss((w(p) f(p, x))_{p \in P'}) = loss((w(c) f(c, x))_{c \in C'}) . \quad (3)$$

Точна базова множина — це вибірка з набору даних, яка не погіршує результат у разі її використання для навчання моделі. Ця множина не може бути універсальною, оскільки найкраща вибірка для однієї задачі може бути абсолютно непридатною для іншої.

Розглянемо таку задачу [2]: нехай потрібно відкрити магазин, що повинен розташовуватися на заданій вулиці, паралельній деякій осі координат. Треба

знайти таку точку, яка буде знаходитись якомога ближче до всіх потенційних $n>1$ покупців. З огляду на це, функція втрат буде різницею відстані між вибраною точкою та найдальшим потенційним покупцем. Маємо такий простір запитів для цієї задачі:

$$P=\{p_1,\dots,p_n\}\subseteq\Re,\ w=1,\ X=\Re,\ f(p,x)=|p-x|,\ loss(\cdot)=||\cdot||_\infty. \quad (4)$$

Потрібно знайти базову множину C для заданої задачі у такий спосіб, щоб для будь-якого $x\in X$ виконувалася така рівність:

$$f_{loss}((C,1),x)=\max_{p\in C}|p-x|=\max_{p\in P}|p-x|=f_{loss}((P,1),x). \quad (5)$$

З огляду на постановку задачі та функцію вартості якість обраного місця $x\in X$ буде залежати тільки від максимальної відстані від покупця до магазину. Тоді для будь-якого x якнайдалі буде знаходитись розташований або найлівіше, або найправіше покупець (рис. 1, *а*). Тоді базова множина для цієї задачі матиме вигляд $C=\{p_{\min},p_{\max}\}\subseteq P$. Точність множини можна довести у такий спосіб:

$$f_{loss}((P,1),x)=\max_{p\in P}|p-x|=$$
$$=\max_{p_{\min},p_{\max}\in P}\{|p_{\min}-x|,|p_{\max}-x|\}=f_{loss}((C,1),x). \quad (6)$$

Якщо ускладнити умови задачі (зробити пряму вулицю кільцеподібною), знайдена вибірка не буде оптимальною, оскільки вона не міститиме розв'язку задачі, через що знайти його не вдасться. Це робить підхід з пошуком точної базової множини нестійким до зміни умови задачі.

Наближена базова множина дає змогу отримати приріст у швидкості розв'язання цільової задачі, але при цьому будуть втрати у точності. На відміну від попереднього підходу, цей підхід дає можливість знаходити більш універсальні вибірки, які можна використати для ширшого набору задач, при цьому є втрати у точності розв'язку. Прикладом застосування цього підходу є метод обчислення опуклої оболонки навколо множини точок.

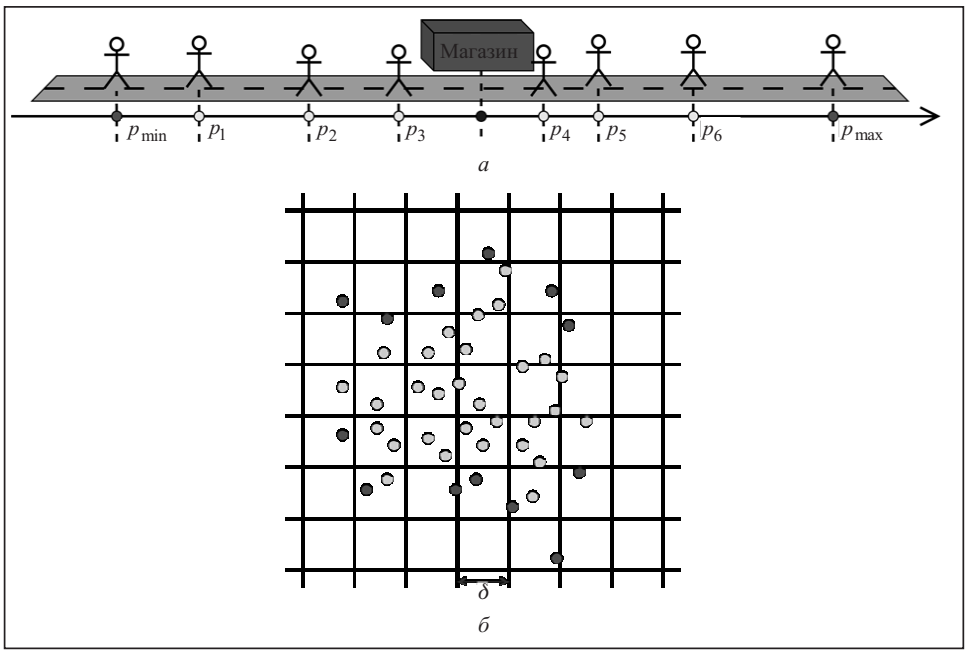

Рис. 1. Схема алгоритму пошуку: точної базової множини для задачі оптимального розташування магазину (*а*); наближеної опуклої оболонки (*б*)

Нехай S^{d-1} — деяка одинична сфера, розташована в центрі простору $\Re^d$. Для будь-якої множини точок P з $\Re^d$ [3] та деякого вектора $u \in S^{d-1}$ можна визначити ширину множини за напрямком у такий спосіб:

$$\omega(u, P) = \max_{p \in P} \langle u, p \rangle - \min_{p \in P} \langle u, p \rangle . \quad (7)$$

Підмножину $Q \subseteq P$ називають *ε-kernel*, якщо для будь якого $u \in S^{d-1}$

$$(1 - \varepsilon)\omega(u, P) \le \omega(u, Q). \quad (8)$$

Підмножина *ε-kernel* є опуклою оболонкою для набору точок P [1], тому алгоритми пошуку *ε-kernel* можна використати як апроксимацію опуклої оболонки для множини P.

Нехай $P \subseteq [-1, +1]^d$ є α-опуклою множиною. Визначимо $\delta \le (\varepsilon / \sqrt{d})$ та $1/\delta \in Z$. Тоді можна визначити таку d-вимірну гратку (рис. 1, *б*):

$$Z = \{(\delta_{i_1}, \ldots, \delta_{i_d}) \mid i_1, \ldots, i_d \in Z\}. \quad (9)$$

У кожній колонці вздовж осі x_d у найвищій та найнижчій непорожніх клітинках беремо випадкову точку та помістимо її у множину Q. Доведено [3], що множина, побудована у такий спосіб, є *ε-kernel*, причому обрахунок можна здійснити за час $O(n + 1/(\alpha\varepsilon)^{d-1})$. Розмір такого набору оцінюють як $O(1/(\alpha\varepsilon)^{d-1})$.

Як можна помітити, незважаючи на значну втрату даних, утворена вибірка все ще здатна репрезентувати форму оригінального набору, через що її можна використати і для інших задач, окрім пошуку опуклої оболонки.

Підходи до пошуку геометричної базової множини є швидкими, що є їхньою найбільшою перевагою, проте основною їхньою проблемою є неможливість працювати зі складними даними. Якщо набір даних являє собою набір зображень або текстову інформацію, методи геометричної базової множини не зможуть розпізнати високорівневі зв'язки у даних та виділити коректну вибірку. Специфіка представлення даних у вигляді геометричних об'єктів обмежує цю можливість.

2.2. Підхід на основі генетичного алгоритму. Генетичний алгоритм є універсальним методом, який застосовують до задач оптимізації. Прикладом застосування цього підходу до задачі пошуку базової множини є EvoCore [4, 5].

Цей підхід дає змогу розв'язати задачу пошуку базової підмножини з використанням алгоритму NSGA-II [6]. Основна ідея є такою: знайти таку підмножину із множини усього набору даних, щоб можна було максимізувати метрики навчання моделі та мінімізувати розмір підмножини.

На вхід алгоритму подають набір даних D, для якого потрібно знайти базову множину, та модель M, на основі якої здійснюють навчання. Задля мінімізації розміру множини, встановлено верхню границю її розміру. Пошук базової множини виконують за допомогою генетичного алгоритму. Як геном вибрано бінарну стрічку, кожне значення якої означає, чи входить елемент $s_i \in D$ в базову множину, чи ні.

Автори підходу, наведеного у [4, 5], протестували алгоритм, розв'язуючи задачу класифікації на таких наборах даних: Blobs dataset [7], Circles dataset [8], Moons dataset [9], Iris dataset [10] та MNIST dataset. Як моделі для оптимізації вибрано Ridge [11], SVC (Support Vector Classification), Bagging класифікатор, Random Forest. Навчання здійснено на повному наборі даних та на знайденій ба-

Таблиця 1. Результати застосування алгоритму EvoCore [4]

Набори даних для тестування	Результат навчання моделей за допомогою алгоритму EvoCore			
	Ridge	SVC	Bagging	Random Forset
Blobs dataset	0.8963 0.9185	0.9407 0.9111	0.8963 0.9407	0.9185 0.9333
Circles dataset	0.5000 0.6343	0.9851 0.9776	0.9478 0.9552	0.9552 0.9627
Moons dataset	0.8134 0.8209	0.9179 0.9403	0.9254 0.9478	0.9328 0.9403
Iris dataset	0.8134 0.9412	0.9179 0.9412	0.9254 0.9608	0.9328 0.9412
MNIST	0.9447 0.8466	—	—	—

зовій множині. Результати навчання наведено для порівняння у табл. 1 (перше число: результат навчання на повному наборі даних; друге число: результат навчання на знайденій базовій множині). Зауважимо, що у більшості випадків вдалося покращити результат навчання моделі порівняно з навчанням на повному наборі даних. Проте результат намагань отримати базову множину на наборі даних MNIST виявився значно гіршим, аніж для повного набору даних. Причиною цього є недостатньо великий розмір вибірки, знайденої алгоритмом.

Ще однією проблемою цього підходу є час, який витрачає алгоритм. За даними авторів [4, 5] пошук базової множини для набору даних MNIST у разі застосування моделі Ridge займає близько трьох годин. Якщо ускладнити модель, а набір даних замінити на більший та складніший, час пошуку зросте у декілька разів, що не є вигідним у реальних умовах.

Алгоритм EvoCore демонструє універсальний підхід для пошуку базової множини для будь-яких даних та моделей. Проте він також наводить на думку, що неможливо виділити єдину базову множину, придатну для будь-якої моделі навчання. Варто шукати підходи, що є універсальними для деякого кластера задач та працюють з певними типами даних.

Можна дійти висновку, що для роботи зі складними даними, які потребують застосування важких моделей (моделей з більшою кількістю вагів для навчання), цей підхід не є практично виправданим, адже вигода від знайденої базової множини не буде покривати витрат на її пошук.

3. ПОШУК БАЗОВОЇ МНОЖИНИ ДЛЯ ЗАДАЧ НЕЙРОННИХ МЕРЕЖ

Нейронні мережі добре зарекомендували себе в обробленні даних високого рівня, зокрема зображень та текстів, де високорівневі зв'язки грають провідну роль в інтерпретації даних. Нагадаємо, що методи знаходження геометричної базової множини не здатні обробляти високорівневі зв'язки у даних, тому для роботи з таким типом інформації важливу роль можуть відіграти саме нейронні мережі.

Аналізуючи задачу пошуку базової множини, автори цієї статті дійшли висновку, що цю задачу можна розв'язати в один з двох способів. Перший — це знаходження евристичного підходу до відбору даних з вхідного набору, або ж створення метрики, за допомогою якої буде виконуватись відбір. Цей підхід є ефективним, адже здатний за час, наближений до лінійного обробити вхідний потік даних та видати результат. Другим підходом є створення нового набору даних на основі вхідного, так званого синтетичного набору. Головна ідея: виок-

ремити потрібну інформацію з набору даних та на її основі згенерувати новий набір меншого розміру. Цей підхід у перспективі дасть можливість працювати з наборами даних на рівні інформації, що міститься в них, що сприятиме оптимізації та покращенню роботи інженера.

Другий підхід має низку вагомих переваг у контексті задач комп'ютерного зору. Однією з переваг є можливість зважувати інформацію у вхідному наборі та перегенеровувати вже зважені дані. Другою перевагою є можливість контролювати інформативність набору даних. Маючи механізм генерації даних, можна контролювати обсяг згенерованої інформації залежно від кількості згенерованих значень. Третьою перевагою є те, що цей метод дасть змогу змішувати різні набори даних, оминаючи проблему несумісності різних наборів. На основі ідей процесу Knowledge distillation, де навчання моделі-учня здійснюють за допомогою групи моделей-учителів, можна генерувати базову множину на основі декількох наборів даних.

Методика Dataset Distillation або Dataset Condensation дає змогу здійснити екстракцію інформації з набору даних та генерацію нового, значно меншого за розміром, набору даних на основі цієї інформації. Нехай маємо набір даних $T = (X_t, Y_t)$, де $X_t \in R^{N \times d}$, N — розмір реальних даних, d — розмірність вхідного елемента, $Y_t \in R^{N \times C}$, C — розмірність вихідного елемента. Потрібно знайти набір синтетичних даних $S = (X_S, Y_S)$, $X_t \in R^{M \times d}$, $Y_t \in R^{N \times C}$, де M — розмір синтетичних даних [12]. Формально задачу можна описати у такий спосіб:

$$S = \arg\min L(S, T), \tag{10}$$

де L — деяка цільова функція для дистиляції набору даних.

3.1. Огляд наявних підходів. У літературі запропоновано три підходи до розв'язання задачі пошуку базової множини шляхом дистиляції даних:

1) пошук шляхом зіставлення результатів навчання (Performance Matching);
2) пошук шляхом порівняння градієнтів (Gradient Matching);
3) пошук шляхом порівняння розподілів (Distribution Matching).

Пошук шляхом зіставлення результатів навчання (Performance Matching) є одним з найочевидніших способів розв'язання даної задачі. Підхід полягає у тому, щоб на кожному наступному кроці навчати модель на синтетичному наборі даних та перевіряти модель на реальному наборі, а далі на основі результатів перевірки на реальному наборі оптимізувати синтетичний набір (рис. 2). Для реалізації цього механізму використано алгоритм дворівневої оптимізації [13]. На внутрішньому рівні оптимізації модель цільової задачі навчають на синтетичному наборі даних; у цей момент відбувається кешування всього градієнта. На зовнішньому рівні здійснюють валідацію моделі на реальних даних, а на основі збереженого градієнта оптимізують генератор синтетичних даних.

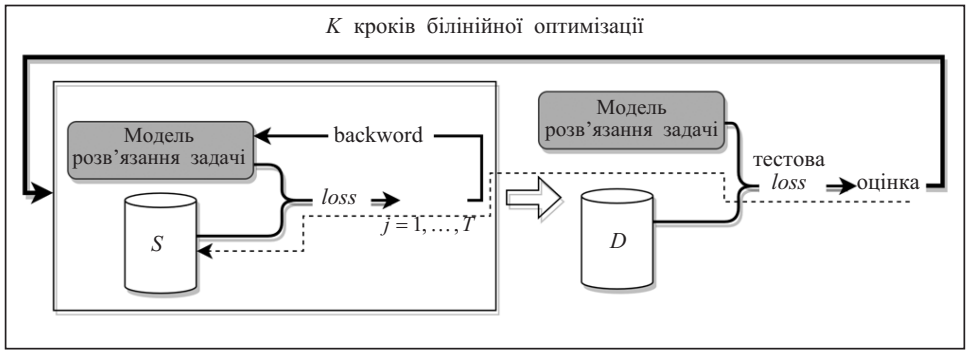

Рис. 2. Схема роботи алгоритмів сім'ї Performance matching

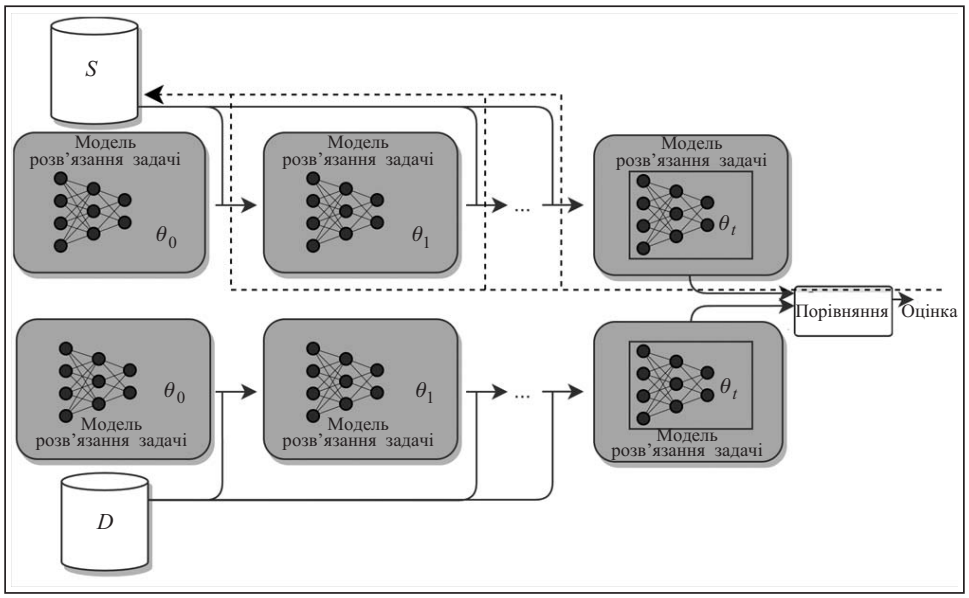

Рис. 3. Схема роботи алгоритмів сім'ї Gradient Matching

Знаковою роботою за цією тематикою є [14]. Її автори спромоглися не лише отримати прийнятний результат, а й дослідити різні аспекти цього підходу. В основу дослідження покладено ідею Knowledge distillation [15], яка полягає в отриманні інформації з важкої моделі або групи моделей (модель-учитель) та навчанні на її основі меншої за розміром моделі (модель-учень). У такий спосіб модель-учень отримує концентровану інформацію про цільову задачу, що дає можливість ефективно вивчити проблему. Ще однією важливою ідеєю, яку привнесли автори, є використання методів оптимізації гіперпараметрів на основі градієнтів [13]. Суть полягає в тому, щоб навчати цільову модель з допомогою гіперпараметрів і водночас зберігати граф градієнтів, після чого на етапі валідації моделі використовувати цей граф для оптимізації вже не моделі, а гіперпараметрів. Отже, вдалося звести синтетичний набір даних до множини гіперпараметрів, для якої потрібно знайти оптимальне значення.

Цей підхід має свої недоліки, найголовніший з яких — це ресурс, витрачений на пошук синтетичного набору даних. Щоб зберігати граф градієнтів на кожному кроці навчання моделі, потрібні великі обсяги пам'яті. Якщо застосувати цей підхід до важких моделей, ресурсу може не вистачити взагалі. Ще однією проблемою є час пошуку синтетичних даних.

Альтернативою цьому підходу є метод «Пошук шляхом порівняння градієнтів» (Gradient Matching) (GM). Його суть полягає у порівнянні параметрів моделей, що навчаються на синтетичному та оригінальному наборах даних (рис. 3). Отже, маємо таку гіпотезу: модель, навчена на базовій множині за допомогою алгоритму градієнтного спуску (gradient descent), має таку саму динаміку сходження, як і модель, навчена на оригінальному набору даних, за умови, що обидві моделі характеризуються однаковими параметрами на початку навчання. Цей підхід є більш ефективним, адже є можливість контролювати кількість ітерацій навчання моделі, після здійснення яких виконують оптимізацію синтетичного набору.

Вперше цей підхід запропоновано у роботі [16] і далі розвинуто у роботах [17, 18]. За основу взято ідею дворівневої оптимізації та ініціалізації синтетичного набору з [13]. Формулювання задачі є таким:

$$\min_{S}\left[\sum_{t=0}^{K-1} dist(\theta_t^S, \theta_t^D)\right], \tag{11}$$

$$\theta_{t+1}^S(S) = opt - alg_\theta(L^S(\theta_t^S), \varsigma^S), \tag{12}$$

$$\theta_{t+1}^D(S) = opt - alg_\theta(L^D(\theta_t^D), \varsigma^D). \tag{13}$$

Цей підхід характеризується тим, що пошук синтетичного набору даних для задачі класифікації здійснюють для кожного класу в реальному наборі даних окремо. У такий спосіб можна отримати одразу збалансований набір даних, що є необхідною умовою для коректного навчання моделі. Проте недоліком цього набору є втрата міжкласових зв'язків, які існували в оригінальному наборі. Це зумовлює виникнення ситуацій, в яких звичайна випадкова вибірка з оригінального набору даних дає кращий результат навчання, ніж синтетичні дані. Розв'язати цю проблему можна методом Dataset Condensation with Contrastive Signal (DCC) [19].

Аналізуючи попередні два підходи, можна помітити, що алгоритм дворівневої оптимізації, який використовують для пошуку базової множини, потребує надзвичайно великого обчислювального ресурсу для розрахунку та збереження графу градієнтів. Цю проблему розв'язують з використанням підходу «Пошук шляхом порівняння розподілів» (Distribution Matching) (DM) (рис. 4). Його суть полягає в тому, щоб використати модель-encoder для переведення значень реального та синтетичного наборів даних у простір ознак, та, порівнюючи розподіли синтетичних та реальних даних у просторі ознак, приводити синтетичні дані до вигляду, аналогічного реальному.

Вперше цю ідею запропоновано в роботі [20]. Як модель-encoder вибрано декілька класичних згорткових нейронних мереж (LeNet, AlexNet, VGG-11, ResNet-18), ініціалізованих випадковими значеннями. Процес пошуку синтетичних даних містив K кроків, у межах яких здійснювали ініціалізацію моделі-encoder випадковими значеннями, генерацію просторових ознак на основі підмножини з реальних та синтетичних даних, порівняння розподілів двох наборів ознак та оптимізацію синтетичних даних на основі різниці розподілів. Як метрику для визначення різниці розподілів вибрано Maximum Mean Discrepancy (MMD) [21].

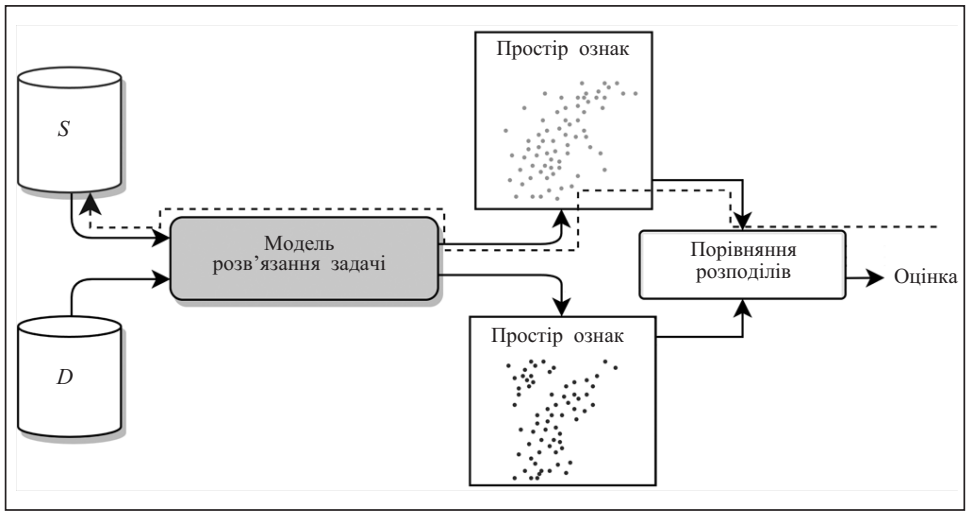

Рис. 4. Схема роботи алгоритмів сім'ї Distribution Matching

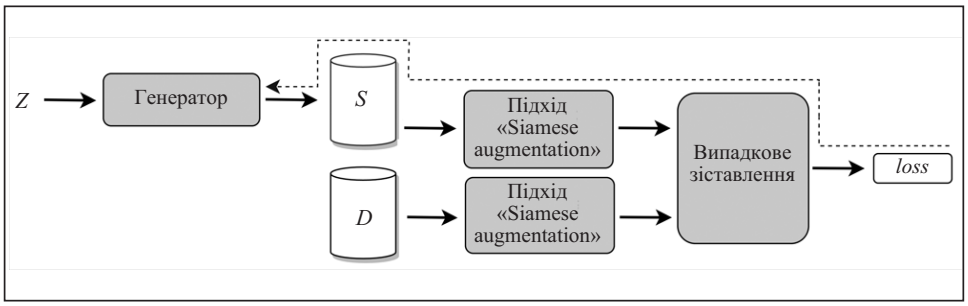

Рис. 5. Схема роботи алгоритму IT-GAN [23]

Ще одним підходом, який дає змогу знайти базовий набір даних шляхом порівняння розподілу з реальним набором, є CAFE [22]. Він ґрунтується на підході Gradient Matching та об'єднує підхід дворівневої оптимізації з додаванням функції втрат, що поелементно порівнює набори векторів ознак на кожному рівні згорткової нейронної мережі.

Підходи, напрацьовані в галузі вивчення Dataset Condensation, демонструють цікавий спосіб екстракції інформації з великого набору даних та конденсації цієї інформації у заданий формат. Аналогічні властивості мають генеративні нейронні мережі (GAN), що здатні, ґрунтуючись на деякому наборі даних, генерувати інформацію за певним патерном та видавати нову, раніше невідому інформацію, яка за стилем схожа на оригінальний набір даних. Аналогічне припущення зроблено авторами методу IT-GAN [23]. Взявши за основу напрацювання в галузі Dataset Condensation [20, 14, 16], вони запропонували спосіб навчання генеративної нейронної мережі на основі реального набору даних. Як основну ідею для навчання моделі взято підхід Distribution Matching. По суті розподіл синтетичних даних у просторі ознак повинен збігатися з розподілом реальних даних у тому самому просторі (рис. 5). Як наслідок, вдалося перевершити результат застосування підходу Distribution Matching.

3.2. Порівняльний аналіз наявних підходів. На основі аналізу описаних підходів можна дійти таких висновків.

- Підходи на основі Performance Matching та Gradient Matching використовують конкретну архітектуру нейронної мережі для знаходження синтетичних даних, що зменшує універсальність отриманих даних.
- Підхід Performance Matching використовує якість навченої на синтетичних даних моделі як цільове значення для оптимізації. Це може зумовити дещо більші значення метрик якості підходу порівняно з іншими алгоритмами.
- Усі розглянуті алгоритми аналізують лише побічну інформацію про набори даних, не заглиблюючись при цьому в інформаційний вміст зображень. У разі застосування таких підходів до більш інформативних наборів результат, отриманий на синтетичних даних, буде значно гіршим.
- Розміри синтетичних наборів, які були згенеровані цими методами, не відповідають інформативності повних наборів даних.

У табл. 2 для порівняння наведено результати застосування алгоритмів DD (Dataset Distillation), GM (Gradient Matching), DM (Distribution Matching) та CAFE на різних наборах даних [20]. Жирним шрифтом виділено значення метрики серед зазначених підходів з найкращим результатом дистиляції на відповідному наборі даних та кількість зображень, згенерованих для одного класу з набору. Можна помітити, що підхід на основі Performance Matching показує значно кращий результат порівняно з іншими підходами, що збігається з висловленим раніше припущенням.

Таблиця 2. Результати застосування алгоритмів DD (Dataset Distillation), GM (Gradient Matching), DM (Distribution Matching) та CAFE на різних наборах даних [20]

Набори даних для тестування	Кількість зображень, згенерованих для одного класу з набору даних	Результат навчання згорткової нейронної мережі на базовій множині, згенерованій одним з алгоритмів			
		DD	GM	DM	CAFE
MNIST	1	**95.2±0.3**	91.7±0.5	89.9±0.8	93.1±0.3
	10	**98.0±0.1**	97.4±0.2	97.6±0.1	97.2±0.2
	50	**98.8±0.1**	98.8±0.2	98.6±0.1	98.6±0.2
CIFAR-10	1	**46.6±0.6**	28.3±0.5	26.5±0.4	30.3±1.1
	10	**60.2±0.4**	44.9±0.5	48.9±0.6	46.3±0.6
	50	**65.3±0.4**	53.9±0.5	63.0±0.4	55.5±0.6
CIFAR-100	1	**19.6±0.4**	12.6±0.4	11.4±0.3	12.9±0.3
	10	**32.7±0.4**	25.4±0.3	29.7±0.3	27.8±0.3
	50	**35.0±0.3**	29.7±0.3	43.6±0.4	37.9±0.3
Fashion-MNIST	1	**83.4±0.3**	70.5±0.6	71.5±0.5	77.1±0.9
	10	**87.6±0.4**	82.3±0.4	83.6±0.2	83.0±0.4
	50	**87.7±0.3**	83.6±0.4	88.2±0.1	84.8±0.4

Також можна помітити, що метод Distribution Matching показує кращу динаміку метрики якості навчання моделі на синтетичних наборах даних залежно від розміру згенерованих даних, ніж метод Gradient Matching (табл. 2). У тому разі, коли генерується одне зображення для одного класу з набору даних, результат застосування Gradient Matching є найкращим. Проте зі збільшенням кількості зображень для одного класу результат змінюється і покращення показує Distribution Matching. Це пов'язано з підходом, що порівнює розподіли реальних та синтетичних даних. Для коректної роботи цього підходу потрібна достатня кількість даних для порівняння, чого неможливо досягнути, маючи лише одне зображення для класу.

У табл. 3 для порівняння наведено результати застосування алгоритмів Gradient Matching (GM) та Dataset Condensation with Contrastive Signal (DCC) на різних наборах даних. Жирним виділено значення метрики серед зазначених підходів з найкращим результатом дистиляції на відповідному наборі даних та кількість зображень, згенерованих для одного класу з набору.

Цікавий результат був продемонстрований в роботі IT-GAN [23]. У табл. 4 наведено результати застосування алгоритмів Gradient Matching (GM) та IT-GAN на різних наборах даних. Для порівняння було згенеровано синтетичні набори даних у розмірі 25 % від усього набору. Жирним шрифтом виділено значення метрики серед зазначених підходів з найкращим результатом дистиляції на відповідному наборі даних та кількість зображень, згенерованих для одного класу з набору.

За допомогою підходів Gradient Matching та IT-GAN автори отримали набір синтетичних даних розміром 25 % від усього набору. Далі автори порівняли результати навчання реальної моделі на синтетичних даних з результатом, отриманим під час навчання моделі на повному наборі даних, демонструючи покращення якості класифікації. Це може свідчити про потенційно кращу здатність генеративних нейронних мереж до екстракції інформації з набору даних.

Таблиця 3. Результати застосування алгоритмів Gradient Matching (GM) та Dataset Condensation with Contrastive Signal (DCC) на різних наборах даних [19]

Набори даних для тестування	Кількість зображень, згенерованих для одного класу з набору даних	Результат навчання згорткової нейронної мережі на базовій множині, згенерованій одним з алгоритмів		Результат навчання на повному наборі даних
		GM	DCC	
SVHN	1	**34.6±2.0**	34.3±1.6	92.1±0.2
	10	**76.2±0.6**	76.2±0.8	
	50	82.7±0.3	**83.3±0.2**	
CIFAR-10	1	28.2±0.7	**32.9±0.8**	81.6±0.3
	10	44.7±0.6	**49.4±0.5**	
	50	54.8±0.5	**61.6±0.4**	
CIFAR-100	1	12.8±0.3	**13.3±0.3**	52.5±0.3
	10	26.6±0.3	**30.6±0.4**	
	50	32.1±0.3	**40.0±0.3**	

Таблиця 4. Результати застосування алгоритмів Gradient Matching (GM) та IT-GAN на різних наборах даних

Набори даних для тестування	Результат навчання згорткової нейронної мережі на базовій множині, згенерованій одним з алгоритмів		Результат навчання моделі на повному наборі даних
	GM	IT-GAN	
CIFAR-10	85.1±0.3	**85.7±0.4**	93.4±0.2
CIFAR-100	56.7±0.5	**60.1±0.2**	74.1±0.2

Можна помітити також відмінність результату у разі застосування розглянутих методів до наборів даних різної складності (табл. 2, 3). Так, у разі пошуку базової множини на наборах CIFAR-10 та CIFAR-100 отримані під час навчання на цих наборах базові множини суттєво відрізняються за своєю інформативністю. Це пов'язано з тим, що набір CIFAR-100 є складним порівняно з набором CIFAR-10. Оскільки інформативність CIFAR-100 є значно більшою, гірший результат за однакової кількості згенерованих зображень на один клас є очікуваним.

З огляду на зазначене, можна виділити такі аспекти розвитку цієї тематики.

- Необхідно додати крок, який аналізуватиме інформацію, зашифровану в зображенні, та генеруватиме дані, максимально різні за інформативністю відносно один одного.
- Підхід на основі генеративних нейронних мереж дає змогу добути інформацію та зберегти її в гнучкій формі (у вигляді вагів нейронної мережі), тоді як інші підходи екстрагують результат своєї роботи в набір зображень, який вже не можна змінити.

ВИСНОВКИ

За результатами аналізу задачі пошуку базової множини для набору даних можна дійти висновку про різноманітність підходів, напрацьованих для розв'язання цієї задачі. Розглянуті геометричні підходи орієнтовані на роботу з низькорівневими даними, а саме геометричними об'єктами у просторі. Пере-

вагою цих підходів є їхня швидкодія, необхідна під час роботи з великими обсягами вхідних даних. Проте ці підходи не можна безпосередньо застосовувати до високорівневих даних (як-от зображення чи текст), а потрібно шукати перехідні кроки, які адаптують високорівневі дані до формату, прийнятного для цих алгоритмів.

Підхід на основі генетичних алгоритмів є універсальним, адже його можна застосувати до будь-якого набору даних і будь-якої задачі. Генетичний алгоритм дає змогу знайти оптимальне значення заданої функції. Проте простір пошуку для відповідної задачі є надзвичайно великим, що ускладнює роботу алгоритму, особливо у разі наборів даних високого рівня, наприклад зображень. Цей підхід працює довше порівняно з іншими, а отриманий результат при цьому значно гірший, адже не завжди вдається знайти оптимальне значення на великому просторі даних (embedding space).

Підходи на основі нейронних мереж показують найкращий результат знаходження базової множини для високорівневих даних. Це досягається за рахунок здатності нейронних мереж екстрагувати інформацію з великих обсягів даних та концентрувати її в потрібному форматі. Проте, незважаючи на отриманий результат, розглянуті підходи мають низку недоліків:

- набір даних розглядається загалом, при цьому ігноруються низькорівневі ознаки, що можуть міститися в кожній окремій одиниці даних;
- відсутній гнучкий формат запису синтетичних даних;
- ресурсовитратність цих методів.

З огляду на це галузь Dataset Distillation є перспективною для її подальшого дослідження. Потрібно сконцентрувати увагу саме на способах аналізу інформації, що міститься в зображеннях, та на генерації синтетичних даних, які міститимуть якомога більше унікальної для навчання інформації. Також перспективним напрямком є розробки в галузі генеративних нейронних мереж. Зазначений спосіб кодування набору даних є гнучким інструментом для роботи з великими обсягами даних, адже дає змогу створювати нові дані без прив'язки до обчислювальних ресурсів.

СПИСОК ЛІТЕРАТУРИ

1. Pankaj K. Agarwal S. Har-Peled, K. R. Varadarajan. Geometric approximation via coresets. In: Combinatorial and Computational Geometry. Goodman J.E., Pach J., Welzl E. (Eds.). New York: Cambridge University Press, 2005. P. 1–30.
2. Jubran I., Maalouf A., Feldman D. Introduction to coresets: accurate coresets. arXiv:1910.08707v1 [cs.LG] 19 Oct 2019. https://doi.org/10.48550/arXiv.1910.08707.
3. Agarwal P.K., Har-Peled S., Varadarajan K.R. Approximating extent measures of points. *Journal of the ACM*. 2004. Vol. 51, Iss. 4. P. 606–635. https://doi.org/10.1145/1008731.1008736.
4. Barbiero P., Squillero G., Tonda A. Evolutionary discovery of coresets for classification. *Proc. Genetic and Evolutionary Computation Conference Companion* (13–17 July 2019, Prague, Czech Republic). Prague, 2019. P. 1747–1754. https://doi.org/10.1145/3319619.3326846.
5. Barbiero P., Squillero G., Tonda A. Uncovering coresets for classification with multi-objective evolutionary algorithms. arXiv:2002.08645v1 [cs.LG] 20 Feb 2020. https://doi.org/10.48550/arXiv.2002.08645.
6. Deb K., Pratap A., Agarwal S., Meyarivan T. A fast and elitist multiobjective genetic algorithm: NSGA-II. *IEEE Transactions on Evolutionary Computation.* 2002. Vol. 6, Iss. 2, P. 182–197. https://doi.org/10.1109/4235.996017.
7. Sklearn datasets: Blobs dataset. URL: https://scikit-learn.org/stable/modules/generated/sklearn.datasets.make_blobs.

8. Sklearn datasets: Circles dataset. URL: https://scikit-learn.org/stable/modules/generated/sklearn.datasets.make_circles.
9. Sklearn datasets: Moons dataset. URL: https://scikit-learn.org/stable/modules/generated/sklearn.datasets.make sub moons.
10. Iris Data Set. URL: https://archive.ics.uci.edu/ml/datasets/iris.
11. Hoerl A.E., Kennard R.W. Ridge regression: biased estimation for nonorthogonal problems. *Technometrics*. 2000. Vol. 42, Iss. 1. P. 80–86. https://doi.org/10.2307/1271436.
12. Yu R., Liu S., Wang X. Dataset distillation: a comprehensive review. arXiv:2301.07014v2 [cs.LG] 21 Jan 2023. https://doi.org/10.48550/arXiv.2301.07014.
13. Maclaurin D., Duvenaud D., Adams R.P. Gradient-based hyperparameter optimization through reversible learning. arXiv:1502.03492v3 [stat.ML] 2 Apr 2015. https://doi.org/10.48550/arXiv.1502.03492.
14. Wang T., Zhu J.-Y., Torralba A., Efros A.A. Dataset distillation. arXiv:1811.10959v3 [cs.LG] 24 Feb 2018. https://doi.org/10.48550/arXiv.1811.10959.
15. Hinton G., Vinyals O., Dean J. Distilling the knowledge in a neural network. arXiv:1503.02531v1 [stat.ML] 9 Mar 2015. https://doi.org/10.48550/arXiv.1503.02531.
16. Zhao B., Mopuri K.R., Bilen H. Dataset condensation with gradient matching. arXiv:2006.05929v3 [cs.CV] 8 Mar 2021. https://doi.org/10.48550/arXiv.2006.05929.
17. Cazenavette G., Wang T., Torralba A., Efros A.A., Zhu J.-Y. Dataset distillation by matching training trajectories. arXiv:2203.11932v1 [cs.CV] 22 Mar 2022. https://doi.org/10.48550/arXiv.2203.11932.
18. Jiang Z., Gu J., Liu M., Pan D.Z. Delving into effective gradient matching for dataset condensation. arXiv:2208.00311v1 [cs.LG] 30 Jul 2022. https://doi.org/10.48550/arXiv.2208.00311.
19. Lee S., Chun S., Jung S., Yun S., Yoon S. Dataset condensation with contrastive signals. arXiv:2202.02916v3 [cs.CV] 16 Jun 2022. https://doi.org/10.48550/arXiv.2202.02916.
20. Zhao B., Bilen H. Dataset condensation with distribution matching. arXiv:2110.04181v3 [cs.LG] 22 Dec 2021. https://doi.org/10.48550/arXiv.2110.04181.
21. Gretton A., Borgwardt K.M, Rasch M., Scholkopf B., Smola A. A kernel two-sample test. *The Journal of Machine Learning Research.* 2012. N 13. P. 723–773.
22. Wang K., Zhao B., Peng X., Zhu Z., Yang S., Wang S., Huang G., Bilen H., Wang X., You Y. CAFE: learning to condense dataset by aligning features. arXiv:2203.01531v2 [cs.CV] 27 Mar 2022. https://doi.org/10.48550/arXiv.2203.01531.
23. Zhao B., Bilen H. Synthesizing informative training samples with GAN. arXiv:2204.07513v2 [cs.LG] 21 Dec 2022. https://doi.org/10.48550/arXiv.2204.07513.

V.M. Tereshchenko, P.A. Zakala

CORESET DISCOVERY FOR MACHINE LEARNING PROBLEMS

Abstract. The coreset discovery problem is reviewed as well as the following three main methods to solve it: geometric coreset estimation, coreset discovery using the genetic algorithm, and coreset discovery using neural networks. We analyze each of these methods and find the cases where they show the best results. The focus of the paper is on neural network-based approaches and their ability to solve the coreset discovery problem. We perform a comparative analysis of several neural network-based approaches, describe their pros and cons, and determine the next steps in solving the coreset discovery problem.

Keywords: coreset, dataset distillation, dataset condensation, geometry coreset, genetic algorithm.

Надійшла до редакції 26.07.2023