3. 1+λ GP ENCODINGS ЕВОЛЮЦІЙНИЙ АЛГОРИТМ ДЛЯ БЕЗПЕЧНОЇ РОБОТИ З ДАНИМИ РІЗНИХ МОДАЛЬНОСТЕЙ

Олександр Яворський, аспірант
Дмитро Харь, магістр
Кафедра математичного моделювання і аналізу даних
Навчально-науковий Фізико-технічний інститут
Національний технічний університет України «Київський політехнічний інститут імені Ігоря Сікорського»

yaotianjiu@gmail.com, khar.dmytro@gmail.com

ВСТУП

Машинне навчання, зокрема з використанням нейронних мереж, активно впроваджується у великій кількості галузей, від медицини та фізики [1-4] до оптимізації показів реклами та інформаційної безпеки [5-8]. Однією з суттєвих відмінностей машинного навчання, що базується на використанні нейронних мереж, від так званих класичних алгоритмів та статистичного машинного навчання є проблема чорної скрині. Дана проблема описує ситуацію, коли зовнішній спостерігач не може дослідити кроки, які привели до отримання результату. А точніше, не може отримати їх формальне представлення, що, на практиці, унеможливлює пояснення того, яким чином було отримане те чи інше значення заданим алгоритмом [9-10]. Варто зазначити, що мова йде саме про етап навчання, оскільки отримання результату від навченої моделі добре зрозумілий і часто базується на використанні тих чи інших функцій схожості.

У свою чергу ця особливість нейронних мереж призводить до можливості атак даних алгоритмів під час навчання, знижуючи таким чином як фактичну безпеку рішень, що будуються на їх основі, так і впливаючи на загальний рівень довіри до таких рішень. Разом з цим, навіть ті мережі, що вже були навчені, можна піддавати атакам. Дійсно, як показано в [11-12], нейронні мережі можуть стабільно перевершувати людей-спеціалістів у сфері медицини за якістю своєї роботи, утім через відсутність вищезазначеної «прозорості» у їх роботі, використання даних підходів є порівняно обмеженим. Це ж можна сказати і про інші сфери, де безпека є домінуючим фактором, наприклад, в інформаційній безпеці, інженерії тощо.

Зважаючи на це, постає питання про дослідження у сфері пояснення роботи алгоритмів штучного інтелекту (англ., *Explainable AI*), а також про використання альтернативних методів, які, маючи достатню точність, надають дослідникам та користувачам можливість зрозуміти те, як саме було отримано те чи інше рішення. Разом з цим алгоритми, що будуються на основі нейронних мереж, мають певні структурні особливості, що уможливлюють атаки на дані алгоритми без порушення базових безпекових засад їх системної реалізації. Іншими словами, зловмисник може керувати результатом певних, або впливати одразу на велику кількість, відповідей, що продукуються такими моделями. Це, у комбінації з вищезгаданою відсутністю прозорості, створює суттєві перешкоди перед ефективним та безпечним використанням даних підходів. У даному розділі будуть розглянуті альтернативні методи, що мають кращу інтерпретабельність і при цьому мають меншу вразливість до певних видів атак.

3.1. 1+λ ЕВОЛЮЦІЙНИЙ АЛГОРИТМ З GP КОДУВАННЯМ

Однією з причин активного використання нейронних мереж є те, що алгоритми на їх основі можуть бути універсальними апроксиматорами [13], що є загальновідомим. Разом з тим, той факт, що нейронні мережі не є єдиними універсальними апроксиматорами, часто оминається. Це пов'язано в першу чергу з тим, що доведення універсальності не є тривіальним завданням і гарантує, здебільшого, теоретичний результат, а не практичний. В цьому підрозділі ми розглянемо один із таких універсальних апроксиматорів 1+λ еволюційний алгоритм з GP кодуванням («1+λ EA»), чия «універсальність» була нещодавно доведена [14].

Формально, еволюційний алгоритм EA можна представити як четвірку *EA*(*P*, *Off*, *i*, *Sel*, *Mut*), де *P* позначає батьків (початковий, часто випадковий, набір можливих розв'язків-індивідів), *Off* (англ., *offspring*) — нащадки-індивіди, *Sel* (англ., *selection*) — функція або алгоритм вибору, що визначає який індивід є оптимальним, здійснюючи тим самим процес навчання, *Mut* (англ. *mutation*), у свою чергу, позначає функцію мутації, що вносить випадкові або частково-випадкові зміни у можливі розв'язки, а ***i*** — відповідна ітерація алгоритму.

Що ж стосується GP кодування, то під ним, як і в [13], мається на увазі спеціальний тип представлення (англ., *encoding*),

суть якого полягає у використанні обмеженої кількості термінальних значень та операторів, якими може користуватись алгоритм для побудови розв'язку. Таким чином, на відміну від нейронних мереж, структурна варіативність розв'язків, що продукуються 1+λ EA з GP кодуванням є набагато меншою.

Це, в свою чергу, має специфічні наслідки для роботи алгоритму. З одного боку, глибокий контроль над можливими розв'язками дозволяє гарантувати високий рівень безпеки роботи алгоритму, з точки зору можливого "дрейфу" моделі в бік певних результатів під час донавчання та під час самої роботи, оскільки дозволяє математично оцінити "межі" розв'язків та відмінності між конкуруючими чи новоотриманими. У той же час, якщо наявна специфічна інформація про структуру розв'язку, як це буває у випадку використання фізично-інформованих нейронних мереж (англ., *PINNs*), це дозволяє отримати більш стабільний та швидкий процес навчання та мати певні "гарантії" щодо розв'язку. З іншого ж боку, така обмеженість може призводити до субоптимальних розв'язків для задач, про чиї розв'язки мало що відомо перед початком навчання.

Роботу 1+λ EA можна представити наступним чином. Першим кроком ініціалізується індивід, в нашому випадку індивідом буде дерево, у якого в якості внутрішніх вузлів виступають функції, які мають арність 2, а в якості листків — ознаки датасету або константи з набору: 1, 0, −1, e, π. Для цього індивіду розраховується фітнес-функція. Після того, як була розрахована фітнес-функція для створеного індивіду, він мутує λ разів, таким чином ми отримуємо λ нових індивідів. Мутація в даному випадку відбувається наступним чином: випадково обирається один вузол з усього індивіду і його значення замінюється на якесь інше випадкове, валідне значення (у випадку внутрішніх вузлів — значення замінюється на якусь іншу функцію, а у випадку листків на якусь іншу ознаку, або константу). Після цього розраховуються фітнес-функції для усіх новостворених індивідів і обирається індивід, який має найменше значення фітнес-функції. Далі цей індивід виступає в ролі батька на наступних ітераціях.

Еволюційні алгоритми та методи генетичного програмування є добре вивченими підходами для моделювання та побудови передбачень, вони, як і нейронні мережі, широко використовуються в різних галузях [15-19]. Алгоритм 1+λ є порівняно новим представником сімейства еволюційних підходів, але його варіації вже зарекомендували себе, наприклад, у роботі

перестановками [20-21]. Що ж стосується дослідження роботи самого алгоритму, то важливим є аналіз його ефективності для лінійних псевдо-Булевих функцій [22], особливості вибору розміру множини нащадків [23] та загальної популяції індивідів [24].

Разом з цим, більшість обчислювальних експериментів були скоріш “елементарними” [25] або технічними, як OneMinMax [26]. Це пов’язано в першу чергу з тим, що такі алгоритми мають гарну аналітичну інтерпретованість і їх швидкість сходження та низка інших важливих параметрів можуть бути оцінені у строгий спосіб. Утім, як відомо з історії, не дивлячись на формальну універсальність багатошарового персептрона, його практичне використання не є настільки ж безхмарним, що першочергово було оцінено шляхом проведення експериментів з так званими даними з реального світу. Що і мотивує дане дослідження.

Наразі, для роботи із зображеннями та різними типами табличних даних використовуються різноманітні алгоритми на основі нейронних мереж, як було зазначено в [1-12]. Серед, мабуть, найпопулярніших підходів на даний момент варто виділити конволюційні нейронні мережі (англ., *CNN*), ResNet та AlexNet та їх варіації для роботи з зображеннями. Для роботи з даними, що мають табличний вигляд, в залежності від задачі, популярним є або використання великих мовних моделей (англ., *LLM*) в комбінації з різними методами підготовки даних, що утворюють так звані системи розширеного пошуку та генерації (англ., *Retrieval Augmented Generation, RAG*), або ж більш класичні підходи, що базуються на багатошарових персептронах або ResNet. Разом з цим існують і змішані підходи, де алгоритм, що базується на нейронних мережах, поєднується з більш класичним, як, наприклад, у випадку алгоритмів нейронних лісів.

Як буде показано далі, 1+λ EA підхід може показувати високу ефективність, у порівнянні з багатошаровим персептроном (англ., *MLP*). Разом з тим, такий підхід може виявитися субоптимальним, коли мова йде про порівняно складні набори даних, наприклад, під час роботи з зображеннями. Це пов’язано з тим, що базовий підхід не має методів, які б підвищували більшу стабільність алгоритму та можливість самостійного виходу з локальних мінімумів.

3.2. ПОСТАНОВКА ЗАДАЧІ ТА ЗАПРОПОНОВАНИЙ ПІДХІД

Метою даної роботи є розгляд якісних результатів двох реалізацій 1+λ еволюційного алгоритму. Фактично, задачею алгоритму буде вивчення двох наборів даних для задач бінарної класифікації на табличних даних та багатокласової класифікції зображень. Обидва набори даних мають медичне спрямування.

Як було зазначено раніше, в межах даного розділу розглядається ситуація, коли використання алгоритмів на основі нейронних мереж може мати негативні безпекові наслідки. Простим прикладом такої ситуації може слугувати наступний випадок. Вважатимемо, що державна або приватна установа має на меті розробку нового, або апробацію вже реалізованого, алгоритму, що може в автоматичному режимі визначити наявність певного захворювання. Для цього автори алгоритму надали доступ до нього в мережі Інтернет і будь-яка особа може в ручному або автоматичному режимі за допомогою АРІ завантажити свої дані до моделі та отримати прогноз, чи має вона дане захворювання.

В такій ситуації, зловмисник може здійснити атаку на даний алгоритм та модифікувати результати його роботи. Важливим є те, що атака може бути здійснена без прямого порушення безпекових режимів установи, а лише шляхом завантаження даних, що мають певні математичні характеристики. Можливість атаки не залежить від того, чи знає зловмисник про модель, що була використана, чи ні [27-30].

Оскільки не можна гарантувати, чи був окремий запит або їх послідовність до алгоритму спланованою атакою, бажаним є або використання алгоритму, що легко інтерпретується, або складної системи запобіжників, яка не завжди зможе відрізнити “злоякісні” та “доброякісні” запити. Для цього і пропонується дослідження якості роботи 1+λ ЕА, як прикладу моделі, що легко інтерпретується, та зміни в якій легко відслідкувати, а отже, і запобігти небажаному чи просто підозрілому дрейфу моделі в бік певних результатів.

Наприкінці попереднього розділу було зазначено, що стандартна реалізація 1+λ ЕА може мати порівняно низьку якість та, як буде висвітлено в наступних розділах, занадто довгу сходимість. Для розв’язання цих проблем нами було запропоновано модифікацію цього підходу, порівняння якої зі стандартним можна представлено нижче на рис. 1.

Серед основних змін варто виділити наступні.

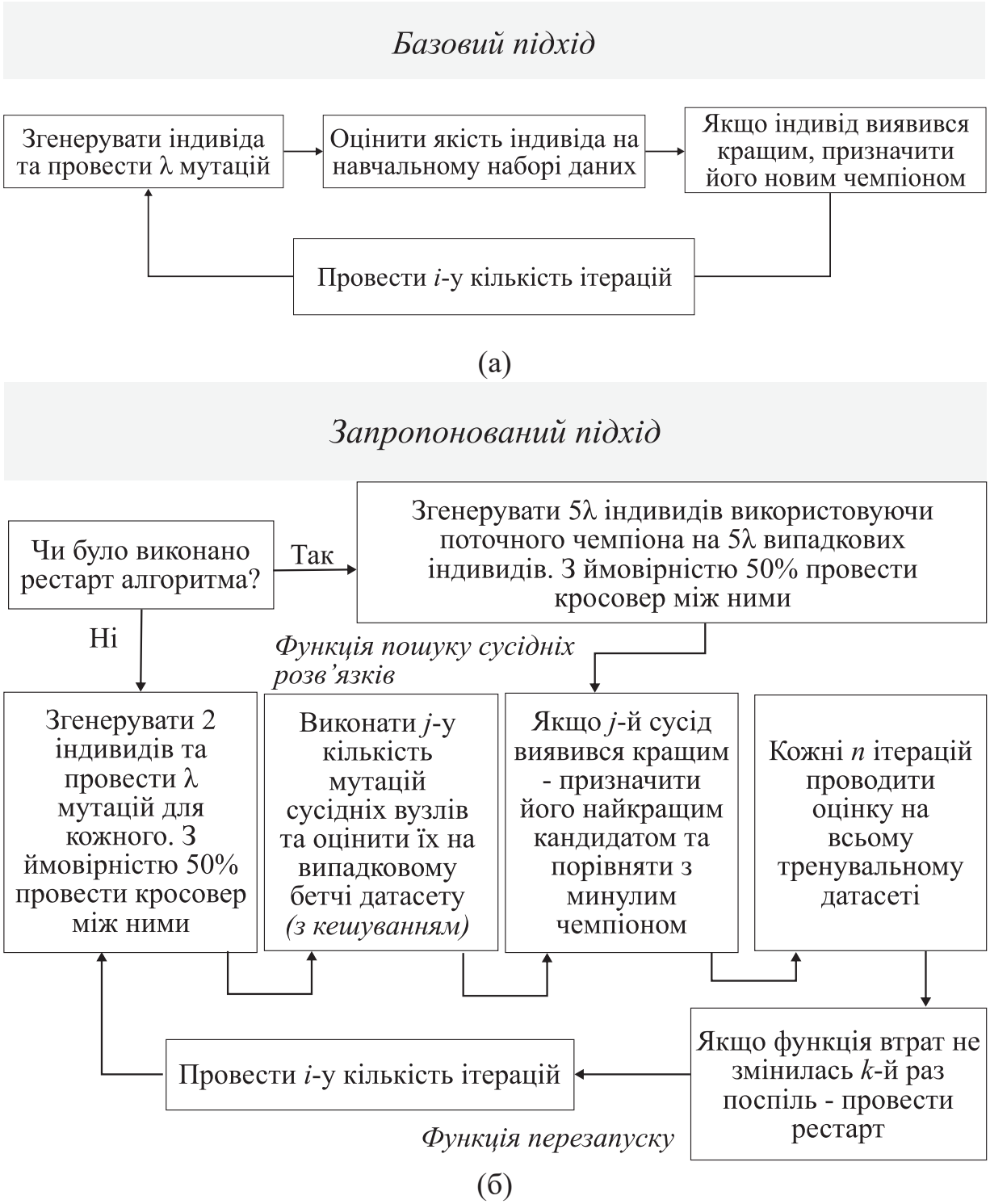

Рис. 1. Схеми роботи базового (а) та запропонованого (б) алгоритмів реалізації алгоритму 1+λ EA.

Додано *функцію перезапуску*, яка, у випадку стагнації швидкості навчання алгоритму використовує останнього найкращого індивіда та продукує 5λ його нащадків і таку саму кількість випадкових індивідів. Це дозволяє реініціалізувати

простір пошуку та згенерувати нових, більш оптимальних нащадків.

Зважаючи на випадковість генерації нащадків, було імплементовано *кешування індивідів*. Таким чином, якщо на *i*-й ітерації було створено індивід, що вже проходив оцінку, використовується попереднє її значення для аналізу якості розв'язку, що дозволило суттєво пришвидшити роботу алгоритму, зважаючи на велику кількість "копій" розв'язків, що генеруються.

Додано *функцію-кросовер*, що дозволяє комбінувати створені розв'язки-індивіди, що уможливлює створення більш різноманітних комбінацій генотипу (структури) індивідів. Це є особливо необхідним, оскільки, як зазначалося вище, алгоритм працює в доволі обмеженому просторі можливих представлень.

Крім цього, авторами було створено функцію пошуку сусідніх розв'язків, що дозволило як запобігти стагнації, так і покращити якість роботи алгоритму. Суть такого пошуку полягає в локалізованій мутації, що дозволяє швидко отримати схожі розв'язки. Для багатокласової класифікації у якості локалі розглядалися піддерева, тоді як для бінарної класифікації, мінімальним елементом слугували вершини дерева-розв'язку.

Додатково, було також використано *бетчінг* (групування) та *k-fold оцінку*. Кількість елементів в групі, параметри *k*-fold оцінки та інші гіперпараметри моделі було оптимізовано шляхом комбінації жадібного пошуку (*англ.* greed search) та Байесовської оптимізації.

3.3. НАБОРИ ДАНИХ

Для проведення експериментів нами було обрано два набори даних з різними модальностями (табличні дані та зображення) і сформовано для них задачі бінарної та мультикласової класифікації, відповідно. Нижче наведено короткий опис використаних датасетів.

Chest X-Ray Images (Pneumonia) [31] — набір даних, який використано для задач бінарної та багатокласової класифікації зображень у медичних дослідженнях. Цей датасет містить рентгенівські знімки грудної клітини пацієнтів з пневмонією (вірусною або бактеріальною) та без неї. Набір даних складається з 5 840 зображень, які віднесені до навчальної (Train) та валідаційної (Test) вибірок. Кожна категорія містить зображення, позначені як "Pneumonia" або "Normal", або якщо розглядати

задачу, як багатокласову класифікацію, то «Pneumonia» розділяється на «Virus» та «Bacteria».

Pima Indians Diabetes Database [32] — набір даних, який використано для задач бінарної класифікації табличних даних в області біомедичних досліджень. Цей набір даних був зібраний Національним інститутом діабету, шлункових і ниркових захворювань США (*англ.* National Institute of Diabetes and Digestive and Kidney Diseases). Набір даних містить інформацію про жінок з племені Піма, що проживають в Арізоні, та включає показники здоров'я, які можуть впливати на розвиток діабету. Набір даних складається з 768 зразків, кожний з яких має 8 вхідних ознак і два вихідних класи, які вказують на наявність або відсутність діабету. В наборі представлено наступні дані: Pregnancies — кількість вагітностей у жінки; Glucose — рівень глюкози у плазмі крові через 2 години після навантажувального тесту; Blood Pressure — діастолічний артеріальний тиск; Skin Thickness — товщина шкірної складки трицепса; Insulin — рівень інсуліну у сироватці крові; BMI — індекс маси тіла; Diabetes Pedigree Function — функція родоводу діабету (враховує генетичну спадковість); Age — вік пацієнта; цільова змінна: Outcome — наявність діабету (0 - відсутній, 1 — наявний). Датасет має збалансовані класи.

Для Pima Indians Diabetes Database була проведена наступна передобробка даних. Спочатку, були замінені нульові значення, які зустрічаються у деяких ознаках (Pregnancies, Glucose, BloodPressure, SkinThickness, Insulin, BMI, DiabetesPedigreeFunction, Age) на відсутні значення (NaN). Далі для кожної ознаки, у якої були відсутні значення, було обчислено медіанні значення для груп з позитивним та негативним результатом по діабету (Outcome). Відсутні значення заповнювалися відповідно до медіанної величини для відповідної групи, оскільки це дозволяє знизити вплив значень, що знаходяться в хвостах розподілів ознак. На практиці, для медичних даних табличного формату використання статистичної імп'ютації не завжди є гарним вибором, адже медичні показники часто не мають явних кореляцій і можуть мати, в певному сенсі, «асиметрічні» значення, які і виступають маркерами належності до певного класу (напр., групи здорових пацієнтів та пацієнтів із захворюванням будуть мати певну асиметрію або декорельованість за певним списком параметрів, що й уможливлює їх віднесення до цих груп). Утім, використання більш складних і точних підходів до імп'ютації, зазвичай, стикається з проблемою взаємозалежних

передбачень. Іншими словами для того, щоб зробити передбачення щодо конкретного відсутнього значення в наборі даних, треба зробити загальне передбачення щодо класу (напр., здоровий чи хворий), і навпаки — значення класу залежить від окремих значень параметрів. Тому було обрано медіанний підхід, який, хоча і не дозволяє отримати ідеальні передбачення, допомагає підвищити обчислювальну стабільність на рівні алгоритму.

Після цього було проведено обробку змінної Insulin для видалення викидів. Викиди визначалися за допомогою методу Interquartile Range Technique [33]. Наступним кроком було виявлення та видалення викидів за допомогою методу Local Outlier Factor [34]. Цей метод використовує локальну щільність сусідів для визначення аномалій. Після розрахунку негативного фактору аномалії для кожного зразка, було визначено порогове значення, і видалено ті зразки, які мали значення нижче цього порогу.

Після видалення аномалій дані були розділені на ознаки та мітки. Вибірки було поділено на навчальний та тестовий набори даних у пропорції 80:20. Потім для навчального та тестового наборів було проведено стандартизацію ознак шляхом видалення середнього значення та масштабування до одиничної дисперсії. В результаті був отриманий оброблений датасет, який в подальшому використовуватиметься для порівняння моделей при розв’язанні задачі бінарної класифікації табличних даних.

Для Pima Indians Diabetes Database була проведена наступна підготовка даних. По-перше, дані були організовані у вигляді класів, де кожне зображення має відповідну мітку (0 — нормальний, 1 — бактеріальна пневмонія та 2 — вірусна пневмонія). Зображення були перетворені до розміру 224×224 пікселів і конвертовані в градації сірого для уніфікації формату. Для вилучення ознак було використано попередньо навчену модель ResNet-50 [35] без останнього повнозв’язного шару. Модель була завантажена зі збережених ваг (які були налаштовані з використанням цього набору даних) та використана для отримання векторів ознак зображень. Далі для обробки отриманих векторів ознак було застосовано стандартизацію ознак шляхом видалення середнього значення та масштабування до одиничної дисперсії. Для зменшення розмірності та збереження 99% варіативності даних було застосовано Метод головних компонент [36]. В результаті ми отримали оброблені вектори ознак, готові до подальшого використання у навчанні.

3.4. ОСОБЛИВОСТІ НАВЧАННЯ

Як зазначалося раніше, алгоритм 1+λ EA розглядається нами як альтернатива MLP, тому порівняльний аналіз було використано для цієї пари моделей. Алгоритм MLP було реалізовано на основі фреймворку Torch. Нижче представлено гіперпараметри для обох задач та моделей (табл. 1, табл. 2). Слід зазначити, що в табл. 2 представлені гіперпараметри для запропонованої, а не базової, варіації 1+λ EA.

Таблиця 1 Гіперпараметри MLP для відповідних задач під час тренування

Гіперпараметри	**Бінарна класифікація**	**Мультикласова класифікація**
hidden_layer_sizes	(10, 15, 10)	(10, 10)
activation	tanh	tanh
solver	sgd	sgd
alpha	0.0009	0.0001
learning_rate_init	0.004	0.001
learning_rate	adaptive	invscaling
batch_size	32	128
tol	0.00002	0.00003

Таблиця 2 Гіперпараметри 1+λ EA для відповідних задач під час тренування

Гіперпараметри	**Бінарна класифікація**	**Мультикласова класифікація**
lambda	2	4
restart_basic	75	75
restart_quick	15	15
restart_lambda_mult	5	5
batch_size	256	1024
tree_depth	2 or 3	2 or 3
nearest_solutions	5	10
full_batch_frequency	15	15
crossover_prob	0.5	0.5
crossover_parents	2	2

Деякі гіперпараметри алгоритму 1+λ ЕА має сенс розглянути більш детально. В табл. 2 *restart_basic* позначає, як часто відбувається реініціалізація простору індивідів, якщо відбувається стагнація навчання. У базовій варіації, значення *tree_depth* та *lambda* дорівнювали 3, для бінарної класифікації та *tree_depth* = 8 і *lambda* = 5 для багатокласової.

У свою чергу, *restart_quick* позначає, як швидко відбудеться перший рестарт алгоритму під час стагнації. Оскільки перша ініціалізація простору є випадковою, стагнація на порівняно високих значеннях функції втрат є не тільки ймовірною а й неоптимальною (тобто не позначає сходимість алгоритму).

Аналогічно, *restart_basic* регулює, як часто буде відбуватися перезапуск алгоритма з оновленим простором індивідів після першої стагнації.

restart_lambda_mult є множником, який використовується у функції перезапуску для підвищення різноманітності індивідів. Занадто великі значення можуть призводити до субоптимальних генерацій, в яких алгоритму важко знайти ефективні розв'язки, тоді як відсутність мультиплікатора (для невеликих значень lambda) означає маленьку вибірку нового простору, що теж негативно впливає на якість роботи алгоритму.

Параметр *tree_depth* приймає значення 2 або 3 випадковим чином для кожного дерева, регулюючи його максимальну глибину. Цікаво, що під час підбору оптимальних гіперпараметрів було виявлено, що більші значення глибини дерев не обов'язково призводять до швидкого перенавчання, хоча майже гарантовано негативно впливають на точність алгоритму. Це можна пояснити тим, що наявна конфігурація алгоритму не передбачає пустих значень вузлів дерев, тобто таких, які позначають відсутність перетворення (умовно, як множення на 1 чи додавання 0), у такому випадку неглибоке, оптимальне дерево пригнічується обов'язковими субоптимальними значеннями сусідніх вузлів або дерев.

Значення *nearest_solutions* впливає на те, скільки мутацій для дерева або вузла, обране під час певної ітерації, буде зроблено в якості пошуку більш оптимальної альтернативи. Великі значення цього параметра будуть призводити до перенавчання для наборів даних з великою варіативністю, але можуть бути гарним рішенням для низьковаріативних наборів.

full_batch_frequency регулює частоту оцінки розв'язку на всій множині тренувальних даних.

Параметри *crossover_prob* та *crossover_parents* відповідають за ймовірність кросоверу та кількість батьків, що прийматимуть в ньому участь, тоді як *full_batch_frequency* позначає як часто відбувається оцінка індивіда-чемпіона для більшої частини тренувальної вибірки.

Для оцінки якості навчання нами для MLP алгоритму було використано BCELoss (Binary Cross Entropy) та Cross Entropy для бінарної та багатокласової задач класифікації, відповідно. BCELoss, або Критерій бінарної перехресної ентропії, визначається наступним чином для цільового значення та передбачення моделі, де останнє є певною ймовірністю значення класа для вхідного значення моделі:

$$l(x,y) = \{l_1,\dots,l_N\}^T, \text{ де } y \in (0,1);$$
$$l_n = -w_n[y_n * logx_n + (1-y_n) * log(1-x_n)]. \quad (1)$$

Перехресна ентропія у свою чергу, визначається для двох розподілів у дискретному випадку як:

$$H = -\sum_{x\in X} p(x) log\, q(x). \quad (2)$$

Обидві варіації 1+λ еволюційних алгоритмів покладалися на значення Log Loss (сума всіх елементів з $\{l_1,\dots,l_N\}^T$), яке обраховувалося після використання сігмоїди та softmax перетворень для бінарної та мультикласових задач в аналогічних випадках. Ми використовували класичні варіації зазначених функцій, а саме:

$$sigmoid(x) = 1/(1+e^{-x});\ softmax(x)_i = e^{x_i} / \sum_{j=1}^{K} e^{x_j}. \quad (3)$$

3.5. АНАЛІЗ РЕЗУЛЬТАТІВ ЕКСПЕРИМЕНТІВ

Спочатку розглянемо базову та запропоновану реалізації 1+λ еволюційного алгоритму.

Час виконання та кількість ітерацій для алгоритмів склала 50 ітерацій за 0.7326 секунд для базової варіації та 2500 ітерацій та 3146.1100 секунд для запропонованої. Як можна побачити, не дивлячись на значні зміни в логіці роботі алгоритму, базова версія має значну перевагу за часом та дещо перевершує за точністю.

Разом з цим, ми додатково проаналізували випадок використання 20 розділень для крос-валідації (замість базових 5 для всіх алгоритмів) та навчання протягом лише 100 ітерацій.

Такі результати можна пояснити тим, що невелика кількість параметрів набору даних, які необхідно вивчити алгоритмові та порівняно велика глибина базового підходу дозволяють швидко знайти гарний оптимум. У той же час оновлений підхід використовує меншу глибину дерев та більш складний процес підбору оптимальних індивідів, що вимагає більше часу, утім не має позитивного ефекту на генералізації алгоритму.

Порівнюючи MLP та базовий варіант алгоритму 1+λ EA можна побачити, що еволюційний підхід має суттєво кращу точність (як і запропонований підхід у порівнянні з MLP) і програє лише в значеннях повноти (англ., recall). В той же час отримані результати ілюструють, що для проміжної, 100-ї, ітерації значення повноти знаходиться в межах середньоквадратичної похибки, яка розрахована для даної моделі як 0.0513. Навчання такої моделі зайняло лише 89.1413 секунди (табл. 3, табл. 4).

Таблиця 3 Порівняння базового та запропонованого 1+λ алгоритмів для задачі бінарної класифікації табличних даних

Метрика / Модель	**Basic 1+λ EA**	**Adjusted 1+λ EA 100 epochs**	**Adjusted 1+λ EA 2500 epochs**
Accuracy	**0.8882**	0.8592	0.8724
Precision	**0.8148**	0.7580	0.8005
Recall	0.8627	**0.8896**	0.8552
F1-score	**0.8381**	0.8153	0.8232

Таблиця 4 Порівняння базового 1+λ EA та MLP для задачі бінарної класифікації табличних даних

Метрика / Модель	**Basic 1+λ EA**	**Adjusted 1+λ EA 100 epochs**	**MLP**
Accuracy	**0.8882**	0.8592	0.8421
Precision	**0.8148**	0.7580	0.7121
Recall	0.8627	0.8896	**0.9038**
F1-score	**0.8381**	0.8153	0.7966

Тепер порівняймо роботу базового та запропонованого алгоритмів для задачі багатокласової класифікації, де вхідними

даними є зображення. Як видно з табл. 5, запропонований підхід має суттєво кращий результат за всіма метриками. Разом з цим, варто зазначити, що на роботу базової версії було витрачено 43179.4566 секунд (близько 12 годин), що є суттєвим обмеженням при практичній реалізації такого алгоритму, особливо у сферах, що вимагають щоденного або навіть щотижневого перенавчання моделей. За цей час алгоритм здійснив 2382 ітерацій. Оновлений 1+λ алгоритм навчався лише 795 секунд або приблизно 13 хвилин.

Порівнюючи запропонований підхід із багатошаровим перцептроном на табл. 6, можна побачити, що MLP класифікатор впорався з поставленим завданням краще, при цьому витративши лише 2 ітерації, які тривали біля 1 секунди.

Таблиця 5 Порівняння базового та запропонованого 1+λ EA для задачі багатокласової класифікації зображень

Метрика / Модель	**Basic 1+λ EA**	**Adjusted 1+λ EA**
Accuracy	0.6955	**0.7227**
Precision	0.7195	**0.7282**
Recall	0.6634	**0.7212**
F1-score	0.6903	**0.7167**

Таблиця 6 Порівняння запропонованого алгоритму 1+λ EA та MLP для задачі багатокласової класифікації зображень

Метрика / Модель	**Adjusted 1+λ EA**	**MLP**
Accuracy	0.7227	**0.7885**
Precision	0.7282	**0.7997**
Recall	0.7212	**0.7885**
F1-score	0.7167	**0.7847**

Таку різницю в результатах можна пояснити, в першу чергу, підходами в навчанні, а саме у використанні стохастичного градієнтного спуску. Оскільки MLP оцінює «напрямок» змін, що продукуються вагами моделі, алгоритму набагато простіше підібрати коректні ваги. У той самий час, алгоритм 1+λ EA покладається виключно на жадібний пошук, що призводить до генерації вже обчислених індивідів. Наприклад, під час навчання для роботи із зображеннями, більш ніж 60% індивідів були обраховані з кешу навчання, що означає, що більш ніж половину

часу, яку алгоритм витратив на створення та підбір кандидатів, було витрачено впусту, оскільки результатом стали не нові кандидати, а копії старих.

3.6. АНАЛІЗ РЕЗУЛЬТАТІВ

В даному розділі було представлено аналіз базової реалізації 1+λ еволюційного алгоритма з GP кодуванням, його оновленої версії та MLP. Для порівняння було використано два набори даних з медичної сфери різних модальностей, а саме табличні дані та зображення.

Проведені експерименти показали, що базова реалізація 1+λ алгоритму здатна перевершувати MLP алгоритм за точністю, принципово не програючи у швидкості (0.58 та 0.60 секунд було витрачено на навчання для MLP та 1+λ алгоритмів, відповідно, для задачі бінарної класифікації табличних даних), даючи приріст в точності у 4.61%. Разом з цим, оновлений підхід впорався з бінарною класифікацією гірше, втративши 1.58% в точності, та витративши суттєво більше часу (89.14 та 3146.11 секунд для 100 та 2500 ітерацій та 20 і 5 розділень для крос-валідації, відповідно).

Залежність повноти від кількості розділень можна пояснити порівняно невеликим розміром набору даних, що може викликати небажане перенавчання та знижувати якість роботи на тестовій вибірці. Тоді як збільшення розділень допомагає отримати більш робастне узагальнення.

З іншого боку, запропонована варіація алгоритму може мати кращу точність за базовий підхід та досягати аналогічних до MLP значень повноти, що є важливим у сферах, що чутливі до типу статистичних помилок, наприклад, у медицині, де хибно-позитивний результат під час визначення чи є пацієнт хворим є кращим, аніж хибно-негативний.

Разом з цим, оновлений 1+λ алгоритм показав кращі результати на задачі багатокласової класифікації, випередивши базову реалізацію на 2.72% в абсолютних значеннях. Швидкість роботи алгоритму для цієї задачі, у порівнянні з базовою реалізацією, також підвищилась у десятки разів. Також важливо зазначити, що запропонована варіація алгоритму показала на 5.78% краще значення повноти для розв’язуваної задачі.

Утім, запропонований 1+λ алгоритм все ж виявився гіршим за підхід на основі MLP як за швидкістю, так і за точністю, що

пов'язано з використанням більш оптимального, з точки зору якості навчання, алгоритму оптимізації.

ВИСНОВКИ

В якості підсумку можна констатувати, що варіації еволюційного алгоритму типу 1+λ з GP кодуванням можуть вдало працювати з різними модальностями для задач багатокласової та бінарної класифікацій.

Принциповою є можливість не тільки отримання адекватних результатів на тестових вибірках, але і можливість запропонованих алгоритмів перевершувати MLP для задачі бінарної класифікації на табличних даних.

Таким чином, можна говорити про доцільність використання даних підходів для розв'язання задач бінарної та багатокласової класифікації у сферах, де збереження «прозорості» та прямої інтерпретованості результатів є принциповим.

Важливо також зазначити, що слабкість еволюційних алгоритмів, що були розглянуті, є і їх сильною стороною, коли мова йде про безпекові аспекти використання методів машинного навчання. Це пов'язано, в першу чергу, з тим, що суттєва кількість підходів до атак на алгоритми, що базуються на нейронних мережах, такі як MLP, використовують градієнтний спуск у якості своєї цілі. Саме цей метод оптимізації (та інші суміжні методи похідних) дозволяє проводити настільки ефективне навчання, утім, залишаючи складність роз'яснення результатів дослідникам та користувачам з одного боку, а з іншого — дозволяючи використання методу Fast Gradient Sign (FSGM), який призначений саме для генерації синтетичних даних, які можуть призводити до вибіркової або невибіркової помилкової класифікації даних під час навчання, що призводить до того, що зацікавлена сторона може контролювати поведінку алгоритму під час інференсу.

Оскільки запропоновані еволюційні алгоритми не використовують даний метод оптимізації, FSGM та схожі методи не мають простору для використання. Разом з цим, можливість прослідкувати за процесом навчання та прийняття рішень алгоритмом від початку до кінця дозволяє захистити користувачів від інших методів втручання в роботу алгоритмів.

Наступними важливими кроками в розвитку цих підходів ми вбачаємо використання 1+λ EA для інших типів задач (наприклад,

замість повнозв’язних нейронних мереж в трансформерах для обробки текстів), оптимізація швидкості роботи алгоритмів та дослідження можливості їх використання замість або разом з фізично-інформованими нейронними мережами, адже обидва підходи дозволяють легку інтеграцію параметрів, що можуть обмежувати процес оптимізації, дозволяючи направити його у необхідному напрямку, що пришвидшує навчання та сприяє якісній генералізації.

ПЕРЕЛІК ПОСИЛАНЬ

1. Belis, Vasilis, et al. "Machine learning for anomaly detection in particle physics." Reviews in Physics, vol. 12, 2024, 100091.

2. Guo, Shenghan, et al. "Machine learning for metal additive manufacturing: Towards a physics-informed data-driven paradigm." Journal of Manufacturing Systems, vol. 62, Jan. 2022, pp. 145-163.

3. Malbranke, Cyril, et al. "Machine learning for evolutionary-based and physics-inspired protein design: Current and future synergies." Current Opinion in Structural Biology, vol. 80, June 2023, 102571.

4. Nilius, Henning, et al. "Machine learning applications in precision medicine: Overcoming challenges and unlocking potential." TrAC Trends in Analytical Chemistry, vol. 179, Oct. 2024, 117872.

5. Munde, Anjali, and Jasmandeep Kaur. "Predictive Modelling of Customer Sustainable Jewelry Purchases Using Machine Learning Algorithms." Procedia Computer Science, vol. 235, 2024, pp. 683-700.

6. Berghout, Tarek, Mohamed Benbouzid, and S.M. Muyeen. "Machine learning for cybersecurity in smart grids: A comprehensive review-based study on methods, solutions, and prospects." International Journal of Critical Infrastructure Protection, vol. 38, Sept. 2022, 100547.

7. Pritee, Zinniya Taffannum, et al. "Machine learning and deep learning for user authentication and authorization in cybersecurity: A state-of-the-art review." Computers & Security, vol. 140, May 2024, 103747.

8. Bahassi, Hanan, et al. "Toward an exhaustive review on Machine Learning for Cybersecurity." Procedia Computer Science, vol. 203, 2022, pp. 583-587.

9. Burrell, J. "How the machine ‘thinks’: Understanding opacity in machine learning algorithms." Big Data & Society, vol. 3, no. 1, 2016, 205395171562251.

10. Ribeiro, M. T., S. Singh, and C. Guestrin. "“Why Should I Trust You?”: Explaining the Predictions of Any Classifier." ArXiv, 1602.04938v3, 2016.

11. Richens, Jonathan G., Ciaran M. Lee, and Saurabh Johri. "Improving the accuracy of medical diagnosis with causal machine learning." Nature Communications, vol. 11, no. 3923, 2020.

12. Artificial Intelligence and Machine Learning in Software as a Medical Device. Url: https://www.fda.gov/medical-devices/software-medical-device-samd/artificial-intelligence-and-machine-learning-software-medical-device

13. Hornik, Kurt, Maxwell Stinchcombe, and Halbert White. "Multilayer feedforward networks are universal approximators." Neural Networks, vol. 2, no. 5, 1989, pp. 359-366.

14. Meyerson, Elliot, Xin Qiu, and Risto Miikkulainen. "Simple Genetic Operators are Universal Approximators of Probability Distributions (and other Advantages of Expressive Encodings)." ArXiv, 2202.09679, 2022. Url: https://arxiv.org/pdf/2202.09679

15. He, Chunlin, et al. "A review of surrogate-assisted evolutionary algorithms for expensive optimization problems." Expert Systems with Applications, vol. 217, 2023.

16. Gobeyn, S., et al. "Evolutionary algorithms for species distribution modelling: A review in the context of machine learning." Ecological Modelling, vol. 392, 2019, pp. 179-195.

17. Thakkar, Ankit, et al. "Applicability of genetic algorithms for stock market prediction: A systematic survey of the last decade." Computer Science Review, vol. 53, 2024.

18. Kuptametee, Chanin, et al. "A review of efficient applications of genetic algorithms to improve particle filtering optimization problems." Measurement, vol. 224, 2024.

19. Wang, ZhenZhou, et al. "A comparative review between Genetic Algorithm use in composite optimisation and the state-of-the-art in evolutionary computation." Composite Structures, vol. 233, 2020.

20. Bassin, Anton, et al. "The (1 + (λ, λ)) genetic algorithm for permutations." GECCO '20: Proceedings of the 2020 Genetic and Evolutionary Computation Conference Companion, pp. 1669-1677, 2020.

21. Laessig, Joerg, and Dirk Sudholt. "Analysis of speedups in parallel evolutionary algorithms and (1+λ) EAs for combinatorial optimization." Theoretical Computer Science, vol. 551, 2014, pp. 66-83.

22. Doerr, Benjamin, and Marvin Кьnnemann. "Optimizing linear functions with the evolutionary (1+λ) algorithm—Different asymptotic runtimes for different instances." Theoretical Computer Science, vol. 561, Part A, 2015, pp. 3-23.

23. Rowe, Jonathan E., and Dirk Sudholt. "The choice of the offspring population size in the (1+λ) evolutionary algorithm." Theoretical Computer Science, vol. 545, 2014, pp. 20-38.

24. Kaufmann, Marc, et al. "Self-adjusting population sizes for the (1,λ)-EA on monotone functions." Theoretical Computer Science, vol. 979, 2023.

25. Doerr, Benjamin, et al. "The (1+λ) Evolutionary Algorithm with Self-Adjusting Mutation Rate." ArXiv, 1704.02191, 2018. URL: https://arxiv.org/pdf/1704.02191

26. Doerr, Benjamin, et al. "The (1 + (λ, λ)) Global SEMO Algorithm." ArXiv, 2210.03618, 2022. URL: https://arxiv.org/pdf/2210.03618

27. Liu, Jia, et al. "A comprehensive survey of robust deep learning in computer vision." Journal of Automation and Intelligence, vol. 2, Issue 4, 2023, pp. 175-195.

28. Xu, H., et al. "Adversarial Attacks and Defenses in Images, Graphs and Text: A Review." International Journal of Automation and Computing, vol. 17, 2020, pp. 151-178.

29. Dong, Huoyuan, et al. "Transferable adversarial distribution learning: Query-efficient adversarial attack against large language models." Computers & Security, vol. 135, 2023.

30. Qiu, Shilin, et al. "Adversarial attack and defense technologies in natural language processing: A survey." Neurocomputing, vol. 492, 2022, pp. 278-307.

31. Kermany, Daniel, Kang Zhang, and Michael Goldbaum. "Labeled Optical Coherence Tomography (OCT) and Chest X-Ray Images for Classification." 2018.

32. "Pima Indians Diabetes Database." Kaggle. Url: https://www.kaggle.com/datasets/uciml/pima-indians-diabetes-database/data.

33. Vinutha, H. P., B. Poornima, and B. Sagar. "Detection of Outliers Using Interquartile Range Technique from Intrusion Dataset." 2018, pp. 511-518.

34. Breunig, Markus, et al. "LOF: Identifying Density-Based Local Outliers." Proceedings of the ACM SIGMOD International Conference on Management of Data, vol. 29, 2000, pp. 93-104.

35. He, Kaiming, et al. "Deep residual learning for image recognition." Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2016, pp. 770-778.

36. Maćkiewicz, Andrzej, and Waldemar Ratajczak. "Principal components analysis (PCA)." Computers and Geosciences, vol. 19, no. 3, 1993, pp. 303-342.