УДК 004.8:338+629.33

М. В. Дратований; В. Б. Мокін, д. т. н., проф.

ІНТЕЛЕКТУАЛЬНИЙ МЕТОД З ПІДКРІПЛЕННЯМ СИНТЕЗУ ОПТИМАЛЬНОГО КОНВЕЄРУ ОПЕРАЦІЙ ПОПЕРЕДНЬОГО ОБРОБЛЕННЯ ДАНИХ У ЗАДАЧАХ МАШИННОГО НАВЧАННЯ

Стаття присвячена синтезу та оптимізації конвеєрів операцій попереднього оброблення у задачах побудови моделей машинного навчання. Зазначено, що важливо оптимізувати всю тріаду цих конвеєрів – вибрати оптимальну послідовність оптимальних операцій з оптимальними параметрами. І при цьому, зміна хоча б одного елемента одразу впливає на вибір усіх інших елементів та їх параметрів. У загальному випадку існує велика кількість допустимих варіантів таких конвеєрів для кожної моделі машинного навчання та вхідних даних (випадкових величин чи часових рядів) і, при цьому, як правило, не існує розмічених датасетів для тренування моделі для синтезу таких конвеєрів. Зроблено огляд відомих підходів до розв'язання таких задач та обґрунтовано висновок про те, що найкраще їх формалізувати як задачі машинного навчання з підкріпленням. Наведено типові підходи з формалізації та інтелектуальні методи розв'язання подібних задач.

Зазначено, що розв'язання задач з підкріпленням, як правило, ускладнено великою розмірністю можливих множин типів та підтипів операцій з різними параметрами і має значні проблеми зі збіжністю до дійсно оптимального значення за обмежений час. А тому запропоновано декілька удосконалень, які дозволять, за певних умов, вирішити цю проблему. По-перше, запропоновано виділяти змінну та незмінні ланки конвеєра операцій попереднього оброблення. Запропоновано для різних типів моделей машинного навчання які операції варто віднести до першої та останньої незмінних ланок, а які – до змінної ланки і тільки до неї запропоновано застосовувати навчання з підкріпленням. А по-друге, запропоновано алгоритм початкового налаштування параметрів RL-політики, залежно від певних статистичних та інших характеристик вхідних даних. Запропоноване удосконалення методу з підкріпленням синтезу оптимального конвеєра операцій може бути застосовано не тільки для операцій попереднього оброблення, а й до інших задач з подібною формалізацією даних та постановкою задачі.

***Ключові слова**: синтез оптимального конвеєру операцій, попереднього оброблення даних, ігрові алгоритми, машинне навчання з підкріпленням, інтелектуальна технологія.*

Вступ

Моделі машинного навчання отримали поширення практично в усіх сферах діяльності людства, але в багатьох задачах точність рішень залишається недостатньо високою. Як правило, це – задачі прогнозування процесів, які є нестаціонарними часовими рядами, наприклад, такі, які зазнають впливу метеорологічних факторів: концентрація у повітрі алергенного пилку рослин чи спор грибів, поширення вірусних інфекцій серед населення, зміна якості води у річках або якості атмосферного повітря, вміст вологості у зернових культурах тощо [1 – 3].

Дослідженню шляхів удосконалення моделей машинного навчання та їх ансамблів присвячена величезна кількість статей, але недостатня увага приділяється етапу попереднього оброблення даних. Адже, без ефективного фільтрування помилкових та аномальних даних важко досягти високої точності прогнозу основної статистичної сукупності даних. Наприклад вартість вживаного автомобіля у 2,5 млрд. доларів або у 0 доларів США [4] навряд чи дозволила б побудувати ефективну модель для прогнозування цієї вартості у США, якщо б такі дані не відфільтрували. Більшість моделей машинного навчання вимагають незалежність ознак та їх розподілення за нормальним законом, але на

практиці таке зустрічається рідко, особливо – друге. Іноді допомагає логарифмування даних, коли менші значення є більш частими. Одним із найпростіших способів забезпечити стаціонарність часового ряду (випадкова величина, яка змінюється у часі) це – перейти від моделювання його значень до моделювання їх приростів (різниць першого порядку) або – до приростів цих приростів (різниць другого порядку). Є й складніші способи. Як правило, послідовність операцій називають конвеєром (англ. «pipeline») [5].

На жаль, немає єдиного погодженого усіма комплексу правил послідовності операцій, щоб максимізувати точність моделі, яка потім їх оброблятиме. Перехід до першої різниці вносить зміни в майже усі закономірності, які треба аналізувати заново. Логарифмування даних чи інше нелінійне перетворення вимагає перегляду прийомів для забезпечення умов стаціонарності, оскільки змін можуть зазнати й усі різниці. Крім того, більшість операцій попереднього оброблення має параметри: для фільтрування аномалій варто знати поріг верхній (як правило, певний квантиль – P95, P90 чи квартиль Q3=P75 або ін.) і нижній (P05, P10 чи Q1=P25 або ін.), для взяття різниць – їх порядок, для нелінійних перетворень даних, якщо це – не просте логарифмування, теж варто знати певні параметри. Тобто треба ідентифікувати оптимальну тріаду для заданого конвеєра – знайти оптимальну послідовність оптимальних операцій з оптимальними параметрами, які оптимізують певний критерій оптимальності (метрику) машинної моделі, яка далі оброблятиме дані на виході цього конвеєра.

Існує ряд підходів до побудови таких конвеєрів. Як правило, використовують експертні інтуїтивні підходи, або оптимізацію на основі орієнтованого ациклічного графа, або з використанням різноманітних підходів нейромережевого проєктування. Загалом використовують методи навчання з учителем [6], мультиагентну оптимізацію [7] або просте Q-навчання з підкріпленням [8]. Однак усім методам властиві такі недоліки: 1) невелика кількість допустимих методів попереднього оброблення, що часто призводить до низької ефективності попереднього оброблення в цілому; 2) великі розмірності матриць суміжності для вибору оптимальної стратегії вибору методів попереднього оброблення даних, що призводить до значної тривалості роботи програми; 3) часто ці методи вимагають втручання операторів чи експертів, а відтак – не забезпечується універсальність технології або є ризик отримати лише квазіоптимальні рішення щодо підвищення точності передбачення даних (для часових рядів – точності прогнозування).

Отже, **метою** цього дослідження є удосконалення методів автоматичної ідентифікації оптимальних конвеєрів операцій попереднього оброблення даних шляхом удосконалення методів навчання з підкріпленням для підвищення точності та швидкості передбачення з використанням моделей машинного навчання.

Огляд операцій попереднього оброблення моделей машинного навчання

Різні дослідники виділяють багато операцій попереднього оброблення моделей машинного навчання. Як правило, їх розрізняють за типами [6 – 9] (будемо позначати як елементи множини A усіх можливих дій – англ. «actions»):

- Очищення (A_0):
  - Імпутинг (англ. «Imputing») (A_{00}):
    - середнє між сусідніми (A_{000});
    - методи факторизації матриць (MF) (A_{001});
    - багатофакторне імпутування за допомогою ланцюжкових рівнянь (MICE) (A_{002});
    - метод найближчого сусіда (англ. «k-Nearest Neighbors» (KNN)) (A_{003}) та ін.;
  - Усунення дублікатів (англ. «Deduplication») (A_{01}):

        - точне виявлення та видалення дублікатів (ED) (A_{010});
        - приблизне виявлення та видалення дублікатів записів (AD) (A_{011}) та ін.;
    - Виявлення та усунення невідповідностей та помилкових даних (англ. «Inconsistency detection») (A_{02}):
        - перевіряє, чи дані задовольняють обмеженням (CC) (A_{020});
        - перевіряє, чи дані відповідають шаблонам (PC) (A_{021}) та ін.;
    - Виявлення та усунення викидів (англ. «Outlier detection») – аномальних даних (A_1):
        - міжквартильний діапазон (англ. «Interquartile range» (IQR)) (A_{10});
        - по квантилях P01-P99 (A_{11});
        - фактор локального викиду (англ. «Local Outlier Factor» (LOF)) (A_{12});
        - виявляє викиди за допомогою Zscore, як функції медіани та середнього абсолютного відхилення (ZSB) (A_{13}) та ін.;
- Нормалізація (A_2):
    - мінімаксне масштабування (MM) (A_{20});
    - десяткове масштабування (DS) (A_{21}) та ін.;
- Приведення закону розподілу до нормального (A_3):
    - логарифмування (A_{30});
    - видобування квадратного кореня (A_{31});
    - розрахунок зворотних значень (A_{32});
    - степеневе перетворення Бокса-Кокса (A_{33});
- Забезпечення стаціонарності: взяття різниць n-го порядку (A_4).

Деякі дослідники, як наприклад у роботі [6], ще відносять до попереднього оброблення операції з видобування та фільтрування ознак (англ. «Feature engineering», скорочено – «FE»), але в цій статті пропонуємо поки цей етап відносити до етапу ідентифікації моделі і зосередитись тільки на етапі безпосередньо операцій попереднього оброблення даних для неї, оскільки етап FE має ряд специфічних особливостей і потребує окремого розгляду.

Формалізація задачі машинного навчання з підкріпленням

Як відомо, метод машинного навчання з підкріпленням оснований на формалізації усіх операцій на базі моделі марковського процесу. Однією з основних властивостей якого є те, що наступний стан залежить тільки від стану у попередній момент часу та вибраної дії. Використовується ігровий алгоритм, коли на кожному «ході» чи кроці алгоритму з тієї чи іншої допустимої множини варіантів дій з використанням певної «RL-політики» (англ. «policy») π вибирається один варіант (англ. «action») переходу системи з i-го стану S_i (англ. «state») в наступний S_{i+1} та обчислюється виграш R (англ. «reward»), який при цьому досягається, і перевіряється чи не досягнуто вже кінцевого виграшу, для якого, як правило, передбачено значно більший виграш (за винятком випадків, коли оптимальним варіантом є якомога триваліша «гра»). У разі вибору поганого чи недопустимого варіанту, встановлюється, як правило, від'ємне значення «виграшу», щоб стимулювати алгоритм більше його не вибирати. Задачею є ідентифікація послідовності допустимих операцій з певними параметрами та налаштування ймовірностей вибору варіантів «ходу» (напрямків на умовній «дошці» для «гри») RL-політики π [6 – 8].

Важливим є те, що застосування RL-моделі повинно проходити в умовах, якомога більше наближених до тих умов, на яких вона тренувалась. Однією з найбільш цінних можливостей цієї технології є те, що вона може працювати за відсутності реальних даних – по суті, генерувати випадкові послідовності за заданих правил та вимог і тренувати сама себе. А коли надходять реальні дані, то вона адаптується до них і корегує свої першочергові налаштування. Звідси – й назва: «навчання з підкріпленням».

Метою задачі є пошук оптимального конвеєру операцій C, який забезпечить максимум виграшу за результатами роботи методу навчання з підкріпленням, та його застосування для отримання максимально ефективно оброблених даних. Цільовою функцією цього методу у

час t на заданій епосі (ітерації алгоритму) є функція G_t [10]:

$$G_t = \sum_{k=0}^{\infty} \gamma^k R_{t+k+1}, \tag{1}$$

де R_t – функція виграшу в момент часу t, γ – коефіцієнт дисконтування, що знаходиться в діапазоні [0, 1] і визначає через скільки епох слід «забувати» вплив попередніх значень. Якщо модель, для якої здійснюється попереднього оброблення, та метод її ідентифікації вже відомі, тоді, як альтернатива, в (1) може враховуватись й виграш (як правило, точність) тільки на останньому кроці, а не – середній чи сумарний виграш на усіх елементах конвеєра.

З формули (1) отримаємо функцію $V(s)$ значення стану s для оцінювання очікуваного стану S_t в момент часу t [10]:

$$V(s) = E[G_t | S_t = s] = E[\sum_{k=0}^{\infty} \gamma^k R_{t+k+1} | S_t = s], \tag{2}$$

де E – операція взяття математичного очікування.

Аналогічно здійснюється формалізація функції значення «стан-дія» $Q(s,a)$ [10]:

$$Q(s,a) = E[G_t | S_t = s, A_t = a] = E[\sum_{k=0}^{\infty} \gamma^k R_{t+k+1} | S_t = s, A_t = a], \tag{3}$$

де A_t – обрана дія в момент часу t, на вибір якої впливає RL-політика π методу на поточній епосі:

$$A_t = \pi(S_t), \tag{4}$$

яка після кожної епохи чи певної кількості таких епох зазнає оновлення до π'. Існують різні методи оновлення RL-політики π', наприклад, у роботі [10] їх наведено 6, є й інші – пропонуємо їх узагальнити таким виразом:

$$\pi' = \mathrm{F}(\mathrm{G}, \mathrm{V}(\mathrm{S}), \mathrm{Q}(\mathrm{S}, \mathrm{A})), \tag{5}$$

тобто для оновлення може використовуватись варіант, який забезпечує максимальний сумарний виграш G (це – найбільш поширений варіант), але може бути й інша функція виграшу, в якій використовуються значення функцій $V(S)$ та $Q(S,A)$, обчислені для різних значень функцій стану та дій, що вибирались на різних етапах, за виразами (2) та (3), відповідно.

Схема RL-методу з урахуванням (1 – 5) наведена на рис. 1.

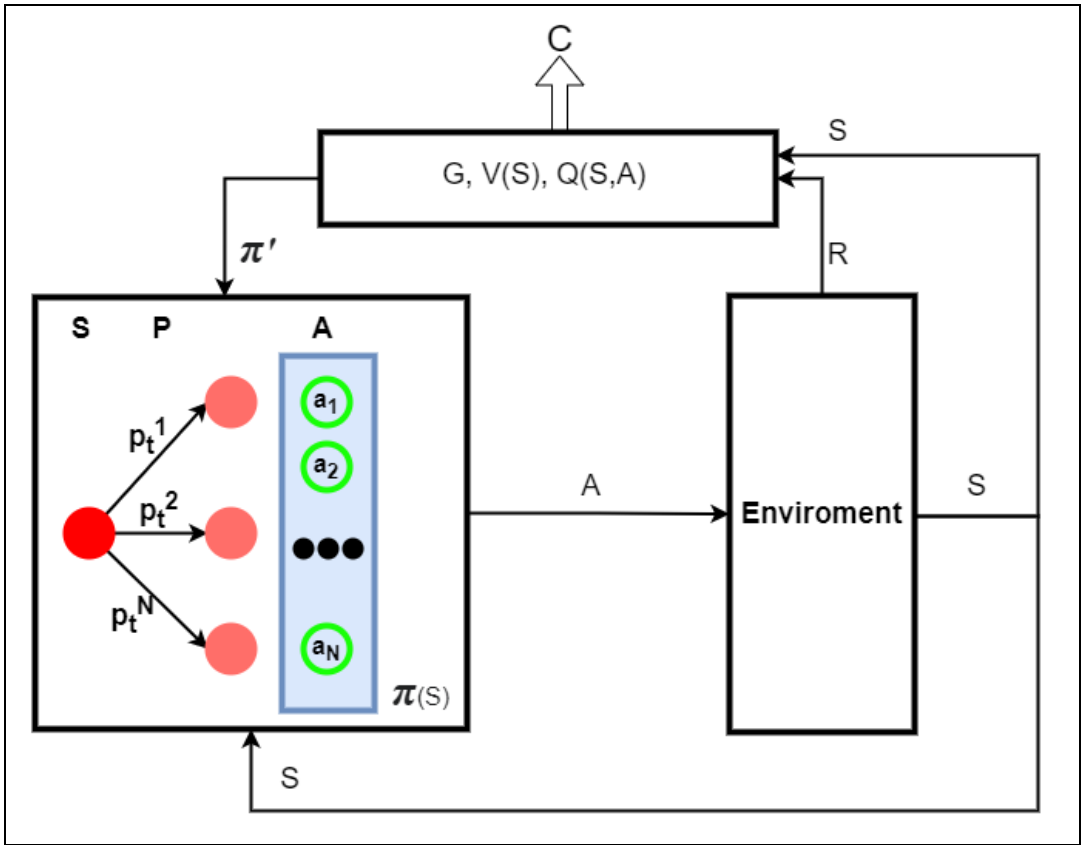

Рис. 1. Схема методу машинного навчання з підкріпленням

Однією з ключових складових RL-методів є алгоритм визначення виграшу – саме це забезпечує збіжність, оптимальність та швидкість отримання результату. Традиційно вибирають якісь фіксовані числа, наприклад: 1 для коректного проміжного ходу, -1 чи -10 для некоректного, 100 – для фінального, який завершує гру. Інший варіант – якась проста функція, яка є однаковою для усіх ходів. Це дійсно спрощує збіжність алгоритму, але суттєво зменшує його функціональність.

Автори пробували використовувати для обчислення проміжного виграшу таку метрику R_i:

$$R_i = \frac{1}{S_{AUCi}}, \tag{6}$$

де S_{AUCi} – безрозмірна величина, яка показує у скільки разів зменшується відсоток площі під кривою після *i*-ої операції попереднього оброблення (англ. «AUC» – «Area Under Curve» – від назви використаної Python-команди бібліотеки sklearn).

Але експерименти з моделювання якості атмосферного повітря у м. Вінниця показали (рис. 2), що для деяких показників, значення яких є більшими за число e = 2,718…, наприклад після логарифмування (операція «Ln» на рис. 2), можуть досягатись дуже великі значення такого виграшу «R3» (рис. 2а). А це суттєво ускладнює визначення сумарного виграшу, оскільки для коректності співставлення конвеєрів треба ще обчислювати ваги виграшів, отриманих на різних ходах. Але, в загальному випадку, конвеєри можуть мати різні послідовності операцій і тоді важко буде їх співставляти.

Psmall	R1	Pbig	R2	Ln	R3
0.03	1.03	0.98	1.02	1	8545.04
0.03	1.03	0.98	1.02	1	8545.04
0.03	1.03	0.98	1.02	1	8545.04
0.03	1.03	0.98	1.02	1	8545.04
0.03	1.03	0.98	1.02	1	8545.04
0.03	1.03	0.98	1.02	1	8545.04
0.02	1.02	0.98	1.02	1	8543.47
0.02	1.02	0.98	1.02	1	8543.47
0.02	1.02	0.98	1.02	1	8543.47
0.03	1.03	0.98	1.02	1	8545.04
0.03	1.03	0.98	1.02	1	8545.04
0.03	1.03	0.98	1.02	1	8545.04

а)

Psmall	R1	Pbig	R2	Ln	R3
0.02	1.0	0.97	1.04	1	1.06
0.02	1.0	0.97	1.04	1	1.06
0.02	1.0	0.97	1.04	1	1.06
0.02	1.0	0.97	1.04	1	1.06
0.02	1.0	0.97	1.04	1	1.06
0.02	1.0	0.97	1.04	1	1.06
0.02	1.0	0.97	1.04	1	1.06
0.02	1.0	0.97	1.04	1	1.06
0.02	1.0	0.97	1.04	1	1.06
0.02	1.0	0.98	1.03	1	1.06
0.02	1.0	0.98	1.03	1	1.06
0.02	1.0	0.98	1.03	1	1.06

б)

Рис. 2. Результати застосування операцій попереднього оброблення до усереднених за кожні 4 години даних показників якості атмосферного повітря у пункті № 1315 мережі EcoCity, розташованого на території ВНТУ, виміряних протягом 11.11.2021 р. – 11.02.2022 р.: а) концентрація озону O_3, б) концентрація оксиду азоту NO_2

А тому пропонуємо не використовувати проміжний виграш по гіпотетичних формулах, а одразу застосовувати модель машинного навчання із заданою метрикою. Найкраще для цього використовувати певну спрощену модель, яка обчислюється досить швидко, а не – моделі, які обчислюються, на багатьох графічних процесорах GPU, протягом декількох годин чи діб. Крім того, краще брати метрики, найкращим значенням яких є 1, а менші значення є гіршими, наприклад в бібліотеці sklearn для регресійних задач це – «r2_score», а для класифікаційних – «accuracy_score», але можливі й інші варіанти. А ще можна брати як виграш не саме значення такої метрики, а – його квадрат чи куб, щоб підсилити вагомість значень, більш близьких до 1.

Існує декілька методів оптимізації RL-моделі [10 – 14]: Q-навчання, DQN, PPO, A2C та їх різні варіації. У найбільш простому, а тому – найбільш поширеному, методі Q-навчання використовується наперед задана таблиця виграшів для усіх комбінацій станів та дій. Як правило, задача суттєво спрощується, щоб мінімізувати розмір такої таблиці. Яскравим прикладом розв'язання цим методом поставленої задачі наведено у роботі [8] (рис. 3). Недоліками є фіксована матриця виграшів, надто мала розмірність такої матриці, за якої буде забезпечуватись розв'язок за обмежений час, та те, що процес вибору оптимальних операцій формується з кінця, тобто задається бажана модель машинного навчання і вже під неї підбираються методи попереднього оброблення, що не завжди може дати правильний ефект, враховуючи, що краще моделі машинного навчання налаштовувати (тюнити) до даних, а не підганяти дані під них.

DQN – це метод, оснований на ідентифікації RL-політики у вигляді глибокої нейромережі з використанням принципів методу звичайного Q-навчання. Оскільки агент Q-навчання не може оцінити ті стани, які не бачив, двовимірну сітку виграшів для кожної комбінації станів та дій замінили на нейронну мережу. Тобто метод DQN використовує нейронну мережу для оцінювання значень Q-функції. На вхід мережі подаються дані, а на виході отримуємо значення Q для кожної можливої дії [8, 13]. Основним недоліком такого методу реалізації навчання з підкріпленням є те, що дискретний простір дій, навіть за невеликого розміру простору дій, збільшується експоненціально, а тому надзвичайно складно отримати збіжність до оптимального рішення за обмежений час.

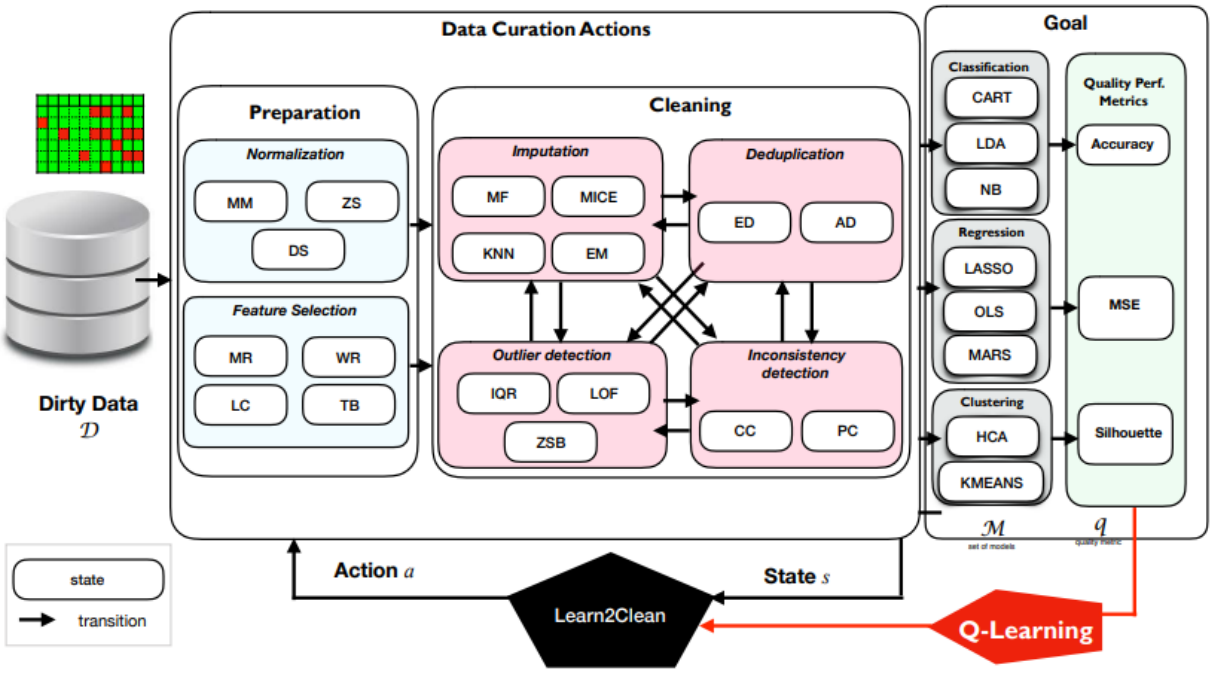

Рис. 3. Алгоритм роботи методу з використанням Q-навчання [8]

PPO – це метод з квазіоптимальною політикою, який полягає в тому, щоб покращити стабільність політики навчання, обмеживши зміни, які відбуваються під час кожної епохи навчання, шляхом уникнення надто різких оновлень політики [10, 14].

На відміну від DQN та PPO, A2C – це метод «актор-критик», який має гібридну архітектуру, що поєднує методи, засновані і на діях, і на зміні політики, зокрема:

- «актор» контролює поведінку агента (метод на основі політики);
- «критик» визначає наскільки була ефективною дія чи операція (метод на основі дій).

Під час навчання «актор» вивчає найкращу дію, використовуючи відгуки «критиків» (замість використання винагороди безпосередньо). У той же час, «критик» вивчає функцію цінності з винагород, щоб він міг належним чином критикувати «актора». Загалом ці агенти можуть обробляти як дискретні, так і безперервні простори дій. Як правило, ідея методу полягає у накопиченні вибраних траєкторій (послідовностей, конвеєрів, комбінацій) можливо оптимальних дій і періодичному їх аналізі з метою корегування політики з метою покращення правил вибору цих оптимальних дій. Пропонуємо використовувати саме цей метод для розв'язання поставленої задачі, але з деякими удосконаленнями.

Оптимізація відомих методів з підкріпленням для побудови конвеєру прийомів попереднього оброблення даних

Для підвищення точності та швидкості застосування методу A2C або його модифікацій до поставленої задачі пропонується, по-перше, виділяти змінну та незмінні ланки загального конвеєра Ω операцій попереднього оброблення:

$$\Omega = \beta(C(\alpha(D))), \tag{7}$$

де α – незмінна ланка послідовних операцій, доцільність застосування яких перевіряється першою для заданих даних D за простим послідовним алгоритмом, C – змінна ланка з оптимальним конвеєром операцій, який застосовується до результату застосування операцій α та ідентифікується з використанням методів навчання з підкріпленням, β – незмінна ланка послідовних операцій, доцільність застосування яких перевіряється останньою для

результату застосування операцій *A* за простим послідовним алгоритмом. Критерієм змінності і незмінності ланок є їх довжина. Якщо у α і β вона – фіксована, то, в загальному випадку, ланка конвеєру *A* може бути різної довжини.

Як відомо, варто розрізняти операції, доцільність яких визначити відносно легко за простим алгоритмом, та операції щодо визначення доцільності, послідовності і параметрів яких дійсно потрібні RL-алгоритми. Крім того, різні моделі вимагають різних підходів щодо попереднього оброблення даних для їх застосування.

Зокрема:

- для усіх моделей потрібно виявляти та усувати дублікати даних;
- усі моделі потребують перевірки на наявність та вилучення із датасетів помилкових та невідповідних даних;
- моделі бібліотеки інструментарію вченого з оброблення даних (англ.: «Science Toolkit for Machine Learning» – скорочено «scikit-learn» або «sklearn»), як правило, вимагають обов'язкову стандартизацію даних середньоквадратичного відхилення з центруванням навколо середнього значення даних (операція «StandardScaler»);
- моделі часових рядів ARIMA, як правило вимагають дані з однаковим часовим кроком без жодних пропусків, а отже, слід робити обов'язкову перевірку даних на відсутність пропущених та їх інтерполяцію, в інших моделях (Facebook Prophet та ін.) більше практикують видалення даних (рядків таблиць), де можуть бути й пропущені, оскільки будь-яка інтерполяція спотворює оригінальні дані і може вносити помилкові закономірності;
- у разі застосування неспецифічних для часових рядів моделей машинного навчання (дерев рішень чи нерекурентних нейронних мереж) до прогнозування періодичних часових часто необхідно ідентифікувати та вилучити періодичний адитивний чи мультиплікативний тренд (специфічні моделі ARIMA, Facebook Prophet тощо мають вбудовані операції для цього);
- масштабування даних більше потрібне для побудови графіків в одній системі координат, а на роботу моделей машинного навчання це, як правило, не впливає.

Причому, операція усунення дублікатів має бути першою, вилучення помилкових та невідповідних – другою, інтерполяція, за необхідності – передостанньою, стандартизація, за необхідності – останньою, щоб мати можливість виконання зручного зворотного перетворення, якщо в цьому є потреба:

$$\alpha = A_{02}(A_{01}(D)) \ , \ \beta = A_4(A_{00}(C(\alpha(D)))) \tag{8}$$

А між цими операціями повинні бути більш складні операції змінної ланки *C*, послідовність, наявність та параметри яких й необхідно ідентифікувати з використанням RL-методів як для випадкових величин, так і для часових рядів:

1. Фільтрування аномалій у більших значеннях, наприклад шляхом вибору фільтру по значеннях квантилів P01-P25 чи P75-P100 (26 варіантів, у т. ч. варіант P100, який означає відсутність фільтрування).

2. Фільтрування аномалій у менших значеннях, наприклад шляхом вибору фільтру по значеннях квантилів P00-P26 (26 варіантів, у т. ч. варіант P00, який означає відсутність фільтрування).

3. Приведення закону розподілу до нормального (логарифмування, взяття зворотних значень або кореня тощо – можлива різна кількість варіантів).

4. Для часових рядів: забезпечення стаціонарності часового ряду шляхом взяття різниці *n*-го порядку, де $n = 0, 1, 2, 3$ (більші значення *n*, як правило, не використовуються), а $n = 0$ відповідає відсутності доцільності використання цього перетворення (4 варіанти).

5. Для часових рядів: нелінійні перетворення тренду, у т. ч. періодичного, і вилучення

його з ряду значень (можлива різна кількість варіантів).

Можливі й інші операції. Але перші 3, а особливо, перші 2, є найбільш популярними та ефективними.

Така оптимізація алгоритму з урахуванням (8) дозволяє оптимізувати швидкодію та залучати більшу кількість операцій на кожному кроці із множин їх допустимих варіантів.

Ще однієї оптимізації можна досягти, якщо врахувати попередній досвід та результати розвідувального аналізу даних. Один із традиційних підходів до роботи RL-методів полягає у відсутності будь-якої попередньої інформації про вхідні дані. Але наявність такої інформації дозволяє часто суттєво прискорити збіжність алгоритму до оптимального рішення за рахунок налаштування ймовірностей у політиці π не випадково, а – за певним алгоритмом:

1. Фільтрування аномалій у більших значеннях («зверху»). Пропонується ймовірності вибирати пропорційно значенням квантилів, масштабованих у діапазон [0, 1] так, щоб максимальні значення (як правило, це – максимум, який умовно можна позначити як P100) вибирались частіше, а P75, при якому втрачається найбільше даних – рідко. Інший варіант – задати ймовірності кривою Гауса таким чином, щоб P75 та максимум вибирались більш рідко, а P87 та сусідні – найбільш часто.
2. Фільтрування аномалій у менших значеннях («знизу»). Аналогічно до п. 1, але дзеркально: ймовірності вибирати зворотно пропорційно значенням квантилів, масштабованих у діапазон [0, 1] так, щоб мінімальні значення (як правило, це – мінімум, який умовно можна позначити як P00) вибирались більш часто, а P25, при якому втрачається найбільше даних – рідко. Інший варіант – задати ймовірності кривою Гауса таким чином, щоб P25 та мінімум вибирались більш рідко, а P12 та сусідні – найбільш часто.
3. Для забезпечення стаціонарності ряду варто враховувати, що оптимальним є варіант, який забезпечує p-value (або p) менше 0,05, але найбільш близько до цього значення. Тобто, якщо перевірка гіпотези про те, що самі значення ряду розподілені не за нормальним законом, дає p-value, що дорівнює 0,1, а щодо першої різниці – 0,04, а другої – 0,02, то це означає, що варто брати першу різницю, а не – другу, оскільки більший порядок означає більшу втрату інформації. А тому варто для кожного варіанту порахувати критерій J_s близькості p-value до 0,05:

$$J_s = \begin{cases} e^{-(p-0.5)^2}, p \le 0.5, \\ 0, p > 0.5, \end{cases} \tag{9}$$

або брати ймовірності $1/n$, $n = 1, 2, 3$, тобто – зворотно пропорційно їх порядку, масштабовані у діапазон [0, 1], а для $n = 0$ брати, наприклад, значення 2, оскільки трактувати результати моделювання для самого ряду часто зручніше, ніж для його різниці.

Аналогічно можна налаштувати усі ймовірності політики π для кожної множини дій на кожному ході (ітерації). Рекомендується найбільшу частоту чи ймовірність задавати для варіанту, коли операція не застосовується, оскільки будь-яка операція вносить зміни у первинний датасет, що не є надто бажаним.

А ще важливо після роботи алгоритму з певною періодичністю оновлювати апріорні частоти (ймовірності) політики π для кожної множини дій вже за результатами моделювання, тобто – апостеріорно.

Висновки

Таким чином, задачу синтезу оптимального конвеєру операцій попереднього оброблення для заданих моделей варто здійснювати з використанням методів та технологій машинного навчання з підкріпленням (англ. «Reinforcement Learning» – RL), оскільки інші методи не гарантують збіжність до оптимального варіанту, якщо брати до уваги відносно велику

кількість операцій попереднього оброблення з різною послідовністю та параметрами.

Здійснено формалізовану постановку задачі. Обґрунтовано вибір методу «A2C» як найкращий для розв'язання цієї задачі з одночасною оптимізацією дій та станів у різний спосіб. Зазначено, що й RL-методи теж мають проблеми зі збіжністю та швидкодією за умов великої кількості можливих операцій. А тому запропоновано декілька удосконалень. Варто здійснювати декомпозицію конвеєру операцій на послідовність незмінних за довжиною ланок та однією середньою ланкою потенційно змінної довжини. Запропоновано прості алгоритми для незмінних ланок, а для змінної ланки варто застосовувати вже складні RL-алгоритми – це дозволяє суттєво зменшити розмірність задачі. Крім того, запропоновано підходи до вибору початкової політики або алгоритму для відбору дій на першій епосі (ітерації) алгоритму з урахуванням статистичних показників та інших закономірностей вхідних даних. Все це дає можливість підвищити шанси знайти дійсно оптимальний розв'язок за прийнятний час.

Запропоноване удосконалення методу з підкріпленням синтезу оптимального конвеєра операцій може бути застосовано не тільки для операцій попереднього оброблення в задачах машинного навчання, а й до інших задач з подібною формалізацією даних та постановкою задачі.

СПИСОК ЛІТЕРАТУРИ

1. Мокін В. Б. Аналіз експериментальних даних, необхідних для синтезу математичної моделі прогнозування розповсюдження алергенних спор грибів Alternaria / В. Б. Мокін, В. В. Родінкова, М. В. Дратований // Вісник Вінницького політехнічного інституту. – 2019. – № 3. – С. 50 – 58.

2. Environmental Factors Which Increase Alternaria spores in Central Ukraine / V. Rodinkova, V. Mokin, O. Bilous, [et al.] // Journal of Allergy and Clinical Immunology. – 2018. – № 141 (2). – P. AB30.

3. Прогнозування хвиль коронавірусу на основі відновленої когнітивної карти міжрегіонального впливу / В. Б. Мокін, М. В. Дратований, А. В. Лосенко [та ін.] // Інформаційні технології та комп'ютерна інженерія. – 2021. – № 3. – С. 86 – 94.

4. Інтелектуальна технологія аналізу та передбачення цін на вживані автомобілі / В. Б. Мокін, А. В. Лосенко, М. В. Дратований [та ін.] // Вісник Вінницького політехнічного інституту. – 2019. – № 6. – С. 62 – 72.

5. Automatically Debugging Auto ML Pipelines Using Maro : ML Automated Remediation Oracle (Extended Version). [Електронний ресурс] / J. Dolby, J. Tsay, M. Hirzel. – 2022. – Режим доступу : https://arxiv.org/pdf/2205.01311.pdf.

6. Reinforcement learning for data cleaning and data preparation [Електронний ресурс] / L. Berti-Equille // Proc. ACM SIGMOD Int. Conf. Manag. – 2019. – Режим доступу : https://laureberti.github.io/website/pub/HILDA2019_Laure.pdf.

7. Automated Data Preprocessing for Machine Learning Based Analyses [Електронний ресурс] A. Paranjape, P. Katta, M. Ohlenforst // COLLA 2022 : The Twelfth International Conference on Advanced Collaborative Networks, Systems and Applications. – 2022. – Режим доступу : https://www.researchgate.net/profile/Akshay-Paranjape/publication/361026018_Automated_Data_Preprocessing_for_Machine_Learning_Based_Analyses/links/6298a13c6886635d5cb84dee/Automated-Data-Preprocessing-for-Machine-Learning-Based-Analyses.pdf.

8. Scenario-based automated data preprocessing to predict severity of construction accidents [Електронний ресурс] / K. Koc, A. P. Gurgun // Automation in Construction. – 2022. – Volume 140. – P. 104351. – Режим доступу : https://www.sciencedirect.com/science/article/abs/pii/S0926580522002242?via%3Dihub.

9. Мокін Б. І. Математичні методи ідентифікації динамічних систем : навчальний посібник / Б. І. Мокін, В. Б. Мокін, О. Б. Мокін. – Вінниця : ВНТУ, 2010. – 260 с.

10. Benchmark study of reinforcement learning in controlling and optimizing batch processes [Електронний ресурс] / W. Zhu, I. Castillo, Z. Wang [et al.] // Journal of Advanced Manufacturing and Processing. – 2022. – № 4 (2). – Режим доступу : https://doi.org/10.1002/amp2.10113.

11. Reinforcement learning for data preparation with active reward learning [Електронний ресурс] / L. Berti-Equille // In Internet Science : 6th International Conference, INSCI 2019, Perpignan, France, December 2 – 5, 2019. – Proceedings 6. – P. 121 – 132. – Springer International Publishing. – Режим доступу : https://doi.org/10.1007/978-3-030-34770-3_10.

12. Papoulis A. Brownian Movement and Markoff Processes / A. Papoulis // Ch. 15 in Probability, Random Variables, and Stochastic Processes, 2nd ed. – New York: McGraw-Hill, – 1984. – P. 515 – 553,

13. Competitive reinforcement learning in Atari games / M. McKenzie, P. Loxley, W. Billingsley [et al.]

// Australasian Joint Conference on Artificial Intelligence. – 2017. – P. 14 – 26.

14. Learning from learners : Adapting reinforcement learning agents to be competitive in a card game / P. Barros, A. Tanevska, A. Sciutti // In 2020 25th International Conference on Pattern Recognition (ICPR). – 2021. – P. 2716 – 2723.

Стаття надійшла до редакції 24.12.2022.
Стаття пройшла рецензування 28.12.2022.

Мокін Віталій Борисович – д. т. н., професор, завідувач кафедри системного аналізу та інформаційних технологій, e-mail: vbmokin@gmail.com.

Дратований Михайло Володимирович – асистент кафедри системного аналізу та інформаційних технологій, e-mail: mishadratovany@gmail.com.

Вінницький національний технічний університет.