ISSN 2786-9482

Ukrainian Journal of Information Systems and Data Science

since 2023

ISSN 2786-9482

Ukrainian Journal of Information Systems and Data Science

2024 Volume 2 Issue 2

УДК 004
Міжнародний науковий журнал ***Ukrainian Journal of Information Systems and Data Science*** виходить двічі на рік. Засновник та видавець – Донецький національний університету імені Василя Стуса. Журнал засновано в травні 2023 р. як електронне наукове видання. Журнал публікує статті українською або англійською мовами.
ISSN 2786-9482
Тематика журналу:

- алгоритми та структури даних;
- інформаційні ресурси;
- моделі прийняття рішень;
- мета-евристична оптимізація;
- моделювання та оптимізація мережевих систем;
- безпечність та надійність складних систем;
- людино-комп'ютерна взаємодія;
- інформаційні системи на основі інтернету речей;
- машинне навчання;
- інженерія знань;
- технології обробки природної мови.

Ukrainian Journal of Information Systems and Data Science: міжнародний науковий журнал / гол. ред. С. Штовба. Вінниця: Донецький національний університету імені Василя Стуса, 2024. №2. 123 с.

Ukrainian Journal of Information Systems and Data Science, №2 за 2024 р. видається за рішенням Вченої ради Донецького національного університету імені Василя Стуса, протокол №17 від 28 березня 2025 р.

Літературний редактор номера – **Ольга Солдатова**.
Номер зверстано редакційною колегією. Підписано до друку 08 травня 2025 р.

Адреса редакції:
Донецький національний університету
імені Василя Стуса, вул. 600-річчя, 21,
Вінниця, 21021, Україна
Тел.: +380-68-256-10-56.
https://jujisds.donnu.edu.ua

UDC 004
Ukrainian Journal of Information Systems and Data Science is an electronic international scientific journal with two issues per year. Vasyl' Stus Donetsk National University is the founder and publisher of the journal. The journal was founded in May 2023. Articles are published in Ukrainian and English.
ISSN 2786-9482
Topics of the journal include:

- Algorithms and Data Structures;
- Information Resources;
- Decision-Making Models;
- Metaheuristic Optimization;
- Modeling and Optimization of Networked Systems;
- Safety and Reliability of Complex Systems;
- Human-Computer Interaction;
- Internet of Things Systems;
- Machine Learning;
- Knowledge Engineering;
- Natural Language Processing Technologies.

Ukrainian Journal of Information Systems and Data Science: international scientific journal / editor-in-chief Serhiy Shtovba. Vinnytsia: Vasyl' Stus Donetsk National University, 2024, vol. 2, issue 2. 123 p.

Ukrainian Journal of Information Systems and Data Science, volume 2, issue 2 is published by the decision of the Academic Council of Vasyl' Stus Donetsk National University, minutes No 17 of March 28, 2025.

Language editor of the issue is **Olga Soldatova**.
The issue was designed by the editorial board. The issue was signed for publication on May 08, 2025.

Contacts:
Vasyl' Stus Donetsk National University,
600-richchia str., 21, Vinnytsia, 21021,
Ukraine
Phone: +380-68-256-10-56
https://jujisds.donnu.edu.ua

ЗМІСТ

CONTENTS

УДК 004.94 : 004.2

Параметризація граф-схем алгоритмів цифрових пристроїв керування

Роман Маркович Бабаков
доцент, д-р техн. наук
ORCID: 0000-0001-7196-0912
r.babakov@donnu.edu.ua

Донецький національний університет імені Василя Стуса

Олександр Олександрович Баркалов
професор, д-р техн. наук
ORCID: 0000-0002-4791-2088
a.barkalov@imei.uz.zgora.pl

Університет Зеленогурський

Ключові слова:
граф-схема алгоритму;
цифрові пристрої керування;
оптимізація схеми;
параметри;
псевдовипадкова генерація.

Розглядається науково-практична задача визначення множини параметрів граф-схем алгоритмів у контексті подальшої псевдовипадкової генерації граф-схем для дослідження ефективності методів синтезу і оптимізації цифрових пристроїв керування. Розглянуто структурні компоненти та визначено загальні параметри граф-схем алгоритмів, які традиційно використовуються для опису алгоритмів роботи цифрових пристроїв керування. Проаналізовано основні класи цифрових пристроїв керування, як-от мікропрограмний автомат (автомат з жорсткою логікою), мікропрограмний пристрій керування (автомат з програмувальною логікою) та композиційний мікропрограмний пристрій керування. Для зазначених класів пристроїв розглянуто основні методи оптимізації апаратурних витрат, серед яких кодування наборів мікрооперацій, заміна вхідних змінних, операційне перетворення кодів станів тощо. Для кожного з розглянутих класів пристроїв керування та методів оптимізації запропоновані набори параметрів граф-схеми алгоритму, які впливають на ефективність застосування відповідних структур і методів та характеризують як функцію переходів, так і функцію виходів пристрою керування. Для окремих параметрів визначено допустимий діапазон змін та співвідношення або взаємовиключність з іншими параметрами граф-схем. Наведені ілюстративні приклади визначення окремих параметрів за заданою граф-схемою. Надано рекомендації щодо використання запропонованих параметрів для псевдовипадкової генерації граф-схем алгоритмів. Визначено такі загальні вимоги щодо коректної псевдовипадкової генерації граф-схем алгоритмів: можливість досягнення кінцевої вершини з будь-якої іншої вершини; відсутність вершин, у яких вхід не зв’язаний з виходом іншої вершини; відсутність повторення логічних умов у послідовно розташованих вершинах; наявність хоча б однієї операторної вершини тощо.

DOI: https://doi.org/10.31558/2786-9482.2024.2.1

Вступ

Цифрові системи займають важливе місце в діяльності людства [1, 2]. Функціонування і взаємодія окремих компонентів цифрової системи здійснюється за допомогою пристрою керування (ПК), який реалізує алгоритм роботи системи [3, 4]. Способи організації ПК та підходи до їх проєктування розглядаються в теорії цифрових автоматів [5–7].

Ключові характеристики ПК стосуються швидкодії, апаратурних витрат, габаритів, енергоспоживання, надійності, вартості тощо. Ці характеристики значною мірою визначають характеристики цифрової системи загалом [8, 9]. Проблематика теорії цифрових автоматів спрямована в тому числі і на оптимізації за зазначеними характеристиками з урахуванням умов проєктування. До таких умов належать, зокрема, алгоритм керування, елементний базис, тип та характеристики об'єкта керування, наявні засоби автоматизованого проєктування тощо. Найбільш варіативним можна вважати алгоритм керування, який імплементується схемою ПК. Відома велика кількість методів оптимізації пристроїв керування, ефективність яких залежить саме від структури і параметрів імплементованого алгоритму керування [10–12].

Алгоритм керування можна представити граф-схемою алгоритму (ГСА) [12, 13]. ГСА містить інформацію про стани автомата, порядок аналізу вхідних сигналів (логічних умов) та залежність вихідних сигналів (мікрооперацій) від вхідних сигналів та стану автомата. Такий рівень абстракції зазвичай є достатнім для опису поведінки ПК та синтезу його схеми в заданому елементному базисі. Довільна ГСА має як структурні атрибути – лінійна, вертикальна, зі зворотними зв'язками, розпаралелена тощо, так і набір числових параметрів, які визначаються її структурою та наповненням вершин. Під час автоматизованого проєктування може застосовуватись представлення ГСА у вигляді текстового файла спеціального формату, наприклад, у форматі KISS2 [14, 15].

Для проєктувальника ПК актуальною задачею є вибір методу оптимізації схеми пристрою. В загальному випадку вибір може здійснюватись не людиною, а спеціалізованою САПР. У низці випадків такий вибір може бути зроблений за результатами аналізу лише параметрів ГСА. Для цього треба знати, які параметри ГСА визначають ефективність певного методу схемотехнічної оптимізації, а які, навпаки, вказують про недоцільність використання цього методу. Такі знання є неочевидними і можуть бути отримані шляхом дослідження ефективності застосування тих чи інших методів схемної оптимізації до ГСА з різною структурою.

Для дослідження ефективності оптимізації схем пристроїв керування можуть бути використані тестові ГСА, що мають абстрактний вміст і відповідають певним вимогам. Водночас ефективність деяких методів може проявлятись лише у випадку ГСА високої складності, що містять сотні станів, вхідних та вихідних сигналів. Виникає науково-практична задача створення великої кількості тестових абстрактних ГСА високої складності із заданими параметрами. Розв'язання цієї задачі можливе шляхом псевдовипадкової генерації ГСА. Перш ніж розробляти відповідні алгоритми, необхідно проаналізувати відомі методи оптимізації та визначити набір параметрів ГСА, які необхідно враховувати в процесі генерації. Ця наукова робота присвячена задачі

формування набору параметрів (параметризації) граф-схем алгоритмів та оцінці їх взаємного впливу з огляду на псевдовипадкову генерацію ГСА згідно з цими параметрами.

Постановка завдання дослідження

Науковим завданням цієї роботи є систематизація і формування множини параметрів ГСА з метою подальшої генерації абстрактних ГСА у псевдовипадковий спосіб відповідно до заданих критеріїв.

Для розв'язання задачі насамперед необхідно визначитися з поняттям і загальними властивостями ГСА. Будемо вважати, що ГСА – це орієнтований зв'язний граф, який може містити чотири типи вершин: початкову, кінцеву, операторну та умовну [4, 7]. Початкова вершина відповідає початку алгоритму і не має входу. Кінцева вершина відповідає кінцю алгоритму і не має виходу. Операторна вершина містить набір одночасно виконуваних мікрооперацій, які надходять від ПК в об'єкт керування. Умовна вершина містить єдиний елемент із множини логічних умов (вхідних сигналів ПК) і має два виходи, що відповідають нульовому та одиничному значенням цієї умови.

Якщо вершини ГСА позначати символами v_i, то множина V вершин ГСА утворюється множиною операторних вершин V_Y, множиною умовних вершин V_X, початковою та кінцевою вершинами v_s, v_e:

$$V = V_Y \bigcup V_X \bigcup \{v_s,\ v_e\}. \quad (1)$$

Як приклад розглянемо ГСА G_1 (рис. 1). Без прив'язки до методу синтезу ПК цю ГСА можна охарактеризувати низкою базових параметрів [12, 13]:

Параметр 1. Загальна кількість вершин N_V=11. Цей параметр дорівнює кількості елементів множини V, яка, згідно з (1), для наведеного прикладу дорівнює $V = \{v_s, v_1,\ ..., v_9, v_e\}$. Окремо можна визначати кількість операторних вершин N_{V_Y} та кількість умовних вершин N_{V_X}. Для ГСА G_1: $N_{V_Y} = 5$ та $N_{V_X} = 4$.

Параметр 2. Кількість різних мікрооперацій – N_Y. Дорівнює потужності множини мікрооперацій Y. У нашому прикладі $Y = \{y_1,\ y_2,\ ...,\ y_7\}$, тому $N_Y = 7$.

Параметр 3. Кількість різних логічних умов – N_X. Дорівнює кількості елементів множини логічних умов X. У нашому прикладі $X = \{x_1,\ x_2,\ x_3\}$, тому $N_X = 3$. Обов'язково $N_X \le N_{V_X}$.

В операторних вершинах вказуються лише ті мікрооперації, які мають одиничні значення. Наприклад, у вершині v_8 мікрооперації y_2 та y_5 формуються рівними одиниці, усі інші, а саме y_1, y_3, y_4, y_6 та y_7, формуються рівними нулю. Для довільної ГСА визначимо параметри 4–7.

Параметр 4. Мінімальна кількість мікрооперацій в операторній вершині – $N_{Y_{min}}$.

Параметр 5. Максимальна кількість мікрооперацій в операторній вершині – $N_{Y_{max}}$.

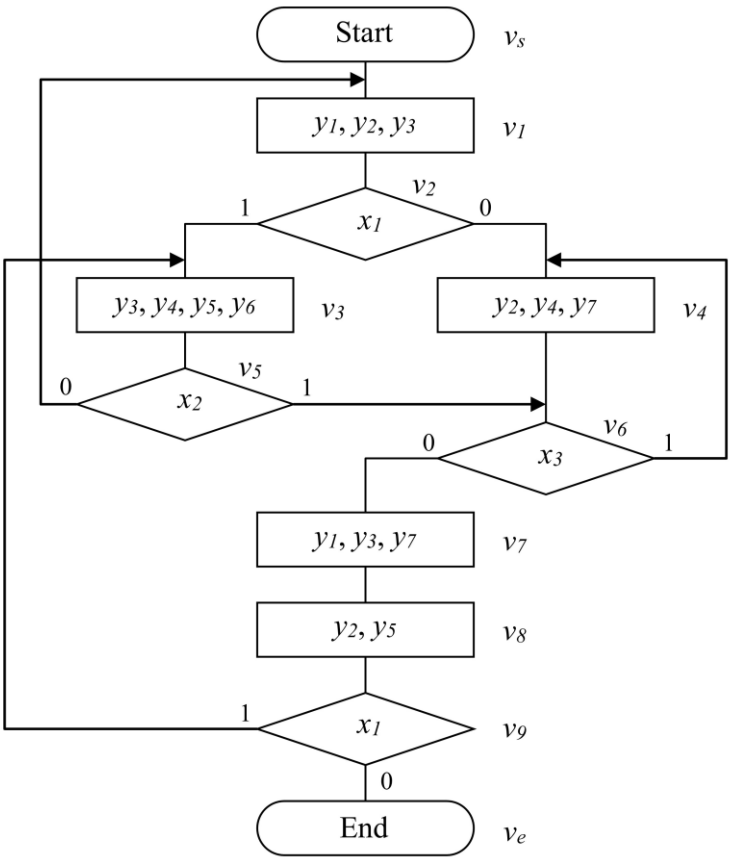

Рисунок 1. Граф-схема алгоритму G_1

Параметр 6. Середня кількість мікрооперацій в операторній вершині – $N_{Y_{mid}}$.

Параметр 7. Математичне очікування кількості мікрооперацій в операторній вершині – N_{Y_E} . N_{Y_E} є самостійним параметром і не пов'язаний безпосередньо із законом розподілу псевдовипадкових значень кількості мікрооперацій в операторних вершинах. Наприклад, якщо у згенерованої ГСА з параметрами $N_{Y_{min}} = 1$, $N_{Y_{max}} = 100$ N_{Y_E} дорівнює 10, то у більшості операторних вершин кількість мікрооперацій буде близька до N_{Y_E} .

Параметри 4–6 можуть залежати від використовуваного методу синтезу ПК, тому їх треба враховувати під час псевдовипадкового генерації ГСА. В загальному випадку можна вважати, що $N_{Y_{min}} > 0$, $N_{Y_{max}} \le N_Y$, $N_{Y_{min}} \le N_{Y_{mid}} \le N_{Y_{max}}$. Для ГСА G_1 $N_{Y_{min}} = 2$, $N_{Y_{max}} = 4$, $N_{Y_{mid}} = 3$.

Параметри 1–7 будемо розглядати як базові параметри ГСА, які не залежать від структури та методу синтезу ПК.

Параметризація ГСА відповідно до класу пристрою керування

Відомі наступні 3 класи ПК, що відрізняються способом імплементації алгоритму керування.

1. Мікропрограмний автомат (МПА) або автомат із жорсткою логікою [4, 7, 9, 13, 14, 16]. Імплементований алгоритм керування реалізується апаратно у вигляді електронної схеми за принципом кінцевого автомата.

2. Мікропрограмний пристрій керування (МПК) [12, 13, 17]. Алгоритм керування реалізується у вигляді набору мікрокоманд, що являють собою двійкові вектори і зберігаються у спеціальній керуючій пам'яті. Для доступу до керуючої пам'яті використовуються різні способи адресації мікрокоманд, що породжує різні структурні модифікації МПК та методи їх синтезу.

3. Композиційний МПК [12, 17, 18]. Структурно являє собою композицію мікропрограмного автомата і автомата з програмувальною логікою, в такий спосіб переваги обох цих класів ПК.

Розглянемо ці 3 класи ПК в контексті параметризації ГСА.

Мікропрограмний автомат

Під час синтезу ПК у вигляді мікропрограмного автомата (finite state machine, FSM) одним з етапів є розмітка ГСА відповідно до станів кінцевого автомата. Можливе позначення ГСА станами автомата Мура або станами автомата Мілі [4, 13, 14, 16]. У випадку автомата Мура відповідність його станів вершинам графу є такою:

– початкова і кінцева вершини позначаються однаковим станом a_0;
– кожна операторна вершина позначається окремим станом, починаючи з a_1.

У випадку автомата Мілі стани відповідають дугам ГСА так:

– вихід початкової і вхід кінцевої вершин позначаються однаковим станом b_0;
– вихід кожної операторної вершини позначається окремим станом, починаючи з b_1;
– якщо виходи декількох операторних вершин поєднуються, то вони позначаються єдиним станом – через це автомат Мілі може мати меншу кількість станів, ніж еквівалентний автомат Мура.

Для перевірки відповідності ГСА станами автомата Мілі треба переконатись, що під час переходу між будь-якими двома станами ми обов'язково проходимо через одну операторну вершину. Для ілюстрації на рис. 2 наведена ГСА G_1 з рис. 1 з розміткою станами автомата Мура (стани $a_0 - a_5$) та станами автомата Мілі (стани $b_0 - b_5$). Спосіб розмітки залежить від того, до якого типу кінцевих автоматів належить синтезований ПК.

Кількість станів автомата Мура або автомата Мілі не може бути обчислена аналітично на основі кількості вершин N_V, N_{V_Y} та N_{V_X}, або на основі інших розглянутих параметрів, оскільки потребує знання структури ГСА. Отже, кількість станів мікропрограмного автомата визначимо додатковим параметром.

Параметр 8. Кількість станів кінцевого автомата *M*.

Під час синтезу МПА його стани позначаються унікальними двійковими векторами (кодами станів). Розрядність кодів усіх станів у межах ГСА однакова і визначається так:

$$R = \log_2(M). \quad (2)$$

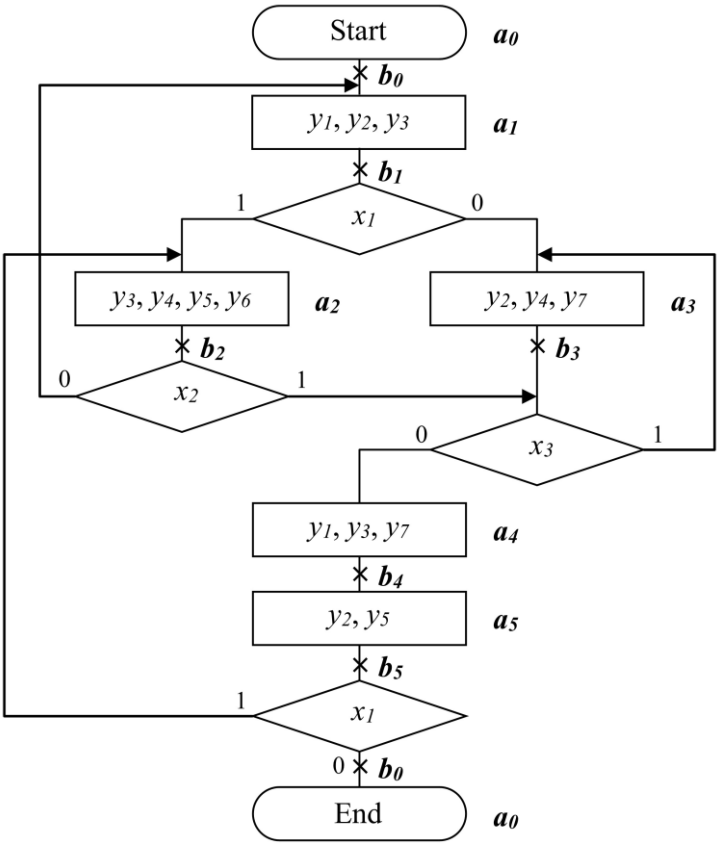

Рисунок 2. ГСА G_1 з розміткою станів автоматів Мілі і Мура

Якщо ГСА задана, значення R визначається значенням M. Для псевдовипадкової генерації ГСА значення R можна задавати і генерувати на його основі значення M. Наприклад, за $R=5$ кількість станів M генерованої ГСА може бути від 1 до 32, але з практичних міркувань M доцільно обмежити діапазоном від 17 до 32. Отже, будемо розглядати R як окремий параметр, хоча його використання за псевдовипадкової генерації ГСА не є обов'язковим і залежить від алгоритму генерації.

Параметр 9. Розрядність двійкових кодів станів автомата R.

Мікропрограмний пристрій керування

Цей клас ПК має регулярну структуру та відносно прості алгоритми синтезу. Метод синтезу залежить від використовуваного способу адресації мікрокоманд (примусова адресація, природна адресація, комбінована адресація та сторінкова адресація) [12, 17]. Замість станів МПА використовуються мікрокоманди, кожна з яких є кортежем булевих векторів певного формату.

В імплементованій ГСА можуть існувати переходи, які залежать від кількох логічних умов. На рис. 2 перехід зі стану a_2 в стан a_4 можливий за одночасного виконання умов $x_2=1,\ x_3=0$, тобто потребує аналізу двох логічних умов одночасно. Як наслідок, перехід зі стану a_2 можливий у більше ніж два стани, а саме у стани $a_1,\ a_3,\ a_4$. Такі переходи називаються багатоспрямованими (на відміну від однопрямованих та двоспрямованих) і в загальному випадку можуть залежати від будь-якої кількості логічних умов. Схема МПА

дає змогу реалізовувати будь-які багатоспрямовані переходи за один такт роботи пристрою. Однак формат мікро-команд МПК дає змогу в одній мікрокоманді вказувати код тільки однієї логічної умови. Залежно від значення логічної умови МПК виконує перехід по одній з двох можливих адрес, що задаються явно або неявно у відповідних полях мікрокоманди. Отже, МПК не здатен виконувати багатоспрямовані переходи за один такт своєї роботи.

Якщо задана ГСА містить послідовності умовних вершин, на першому етапі синтезу МПК необхідно виконати перетворення ГСА, позбавившись багатоспрямованих мікропрограмних переходів. Це робиться шляхом додавання порожньої операторної вершини в кожну дугу, що з'єднує дві умовні вершини. Порожня операторна вершина не містить жодної мікрооперації, однак відповідає окремій мікрокоманді в пам'яті МПК. Внаслідок цього перетворена ГСА буде містити більшу кількість вершин, ніж початкова ГСА, а кількість мікрокоманд МПК буде більшою, ніж кількість станів еквівалентного МПА Мура. Як наслідок, у випадку МПК виконання алгоритму керування за заданою ГСА може займати більшу кількість тактів, ніж у випадку МПА. Наприклад, якщо в ГСА є ділянки з десятьма послідовними умовними вершинами, МПА буде виконувати відповідні переходи за один такт, а МПК – за 10. Якщо такі ділянки в процесі роботи алгоритму повторюються циклічно, схема МПК може суттєво програвати у швидкодії схемі МПА. Відповідно до цих міркувань, введемо 3 додаткові параметри, які можуть бути використані для псевдовипадковій генерації ГСА з орієнтуванням на синтез ПК у вигляді МПК.

Параметр 10. Мінімальна кількість послідовно розташованих умовних вершин $N_{X_{min}}$.

Параметр 11. Максимальна кількість послідовно розташованих умовних вершин $N_{X_{max}}$.

Параметр 12. Середня кількість послідовно розташованих умовних вершин $N_{X_{mid}}$.

У загальному випадку для параметрів 10–12 справедливе таке відношення:

$$N_{X_{min}} \le N_{X_{mid}} \le N_{X_{max}} \le N_X . \quad (3)$$

Вираз (3) вказує, зокрема, на те, що у послідовно розташованих умовних вершинах не можуть зустрічатись однакові логічні умови. Якщо в ГСА міститься найдовший можливий ланцюжок умовних вершин довжиною N_X, усі логічні умови в цьому ланцюжку мають бути різними. На рис. 3 показано фрагмент абстрактної ГСА G_2, в якому у трьох послідовно розташованих умовних вершинах логічна умова x_1 зустрічається двічі. Це є помилкою, оскільки не дає змогу однозначно виконати перехід зі стану a_5 за умовою $x_1 = 0$.

Якщо необхідно, щоб у генерованій ГСА кількість послідовно розташованих умовних вершин групувалась навколо певного значення, можна за аналогією з параметром N_{Y_E}, ввести додатковий параметр 13.

Параметр 13. Математичне очікування кількості послідовно розташованих умовних вершин N_{X_E}.

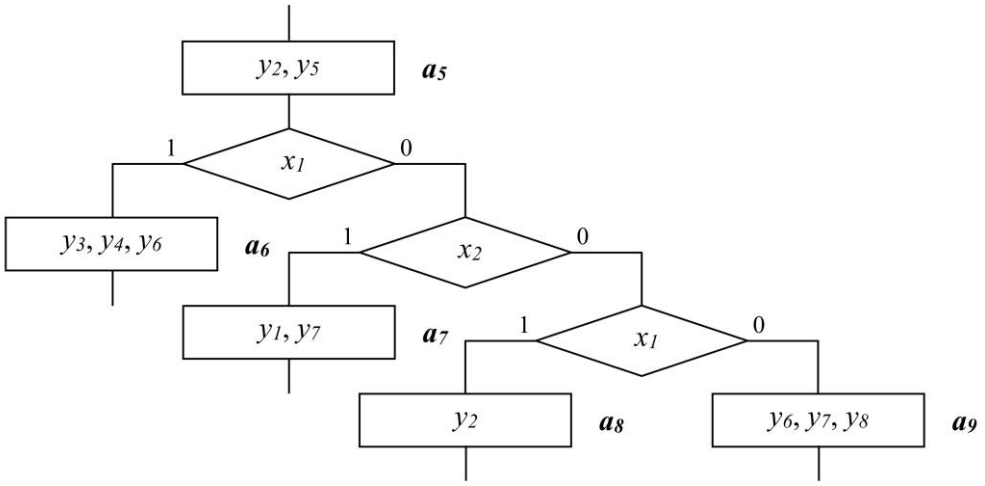

Рисунок 3. Фрагмент ГСА G_2, в якому у послідовно розташованих умовних вершинах двічі зустрічається логічна умова x_1

Композиційний мікропрограмний пристрій керування

Синтез композиційного МПК передбачає виявлення у заданій ГСА так званих операторних лінійних ланцюгів (ОЛЛ, operational linear chain) [18]. Із поняттям ОЛЛ пов'язані наступні чотири визначення.

1. Операторним лінійним ланцюгом у ГСА називається така кінцева послідовність операторних вершин, що для будь-якої пари вершин v_i та v_{i+1} існує дуга, що виходить з v_i і входить у v_{i+1}.

2. Входом ОЛЛ називається операторна вершина цієї ОЛЛ, вхід якої зв'язаний з початковою або умовною вершиною, або з вершинами, що входять в інші ОЛЛ.

3. Вхід ОЛЛ називається головним входом, якщо відсутній зв'язок цього входу з виходами операторних вершин.

4. Виходом ОЛЛ називається вершина, вихід якої зв'язаний зі входами кінцевої або умовної вершини, або зі входом операторної вершини, що входить в іншу ОЛЛ.

Також існує поняття простого операторного лінійного ланцюга. Простим ОЛЛ називається послідовність операторних вершин, між якими немає умовних вершин, а вхід кожної вершини (окрім початкової) зв'язаний тільки з виходом попередньої вершини цього ОЛЛ. Входом простого ОЛЛ вважається операторна вершина, вхід якої зв'язаний з виходом початкової або умовної вершини, або з виходами декількох вершин (операторних чи умовних). Виходом простого ОЛЛ вважаться операторна вершина, вихід якої зв'язаний з кінцевою або умовною вершиною, або з операторною вершиною, що є входом іншої ОЛЛ.

На рис. 4 зображена абстрактна ГСА G_3, в якій операторні лінійні ланцюги окреслені пунктиром. ГСА містить чотири ОЛЛ $O_1 - O_4$. ОЛЛ O_2 та O_4 є простими, оскільки містять одну вхідну вершину. ОЛЛ O_1 містить дві вхідні вершини v_1 та v_2, ОЛЛ O_3 також містить дві вхідні вершини v_8 та v_{10}.

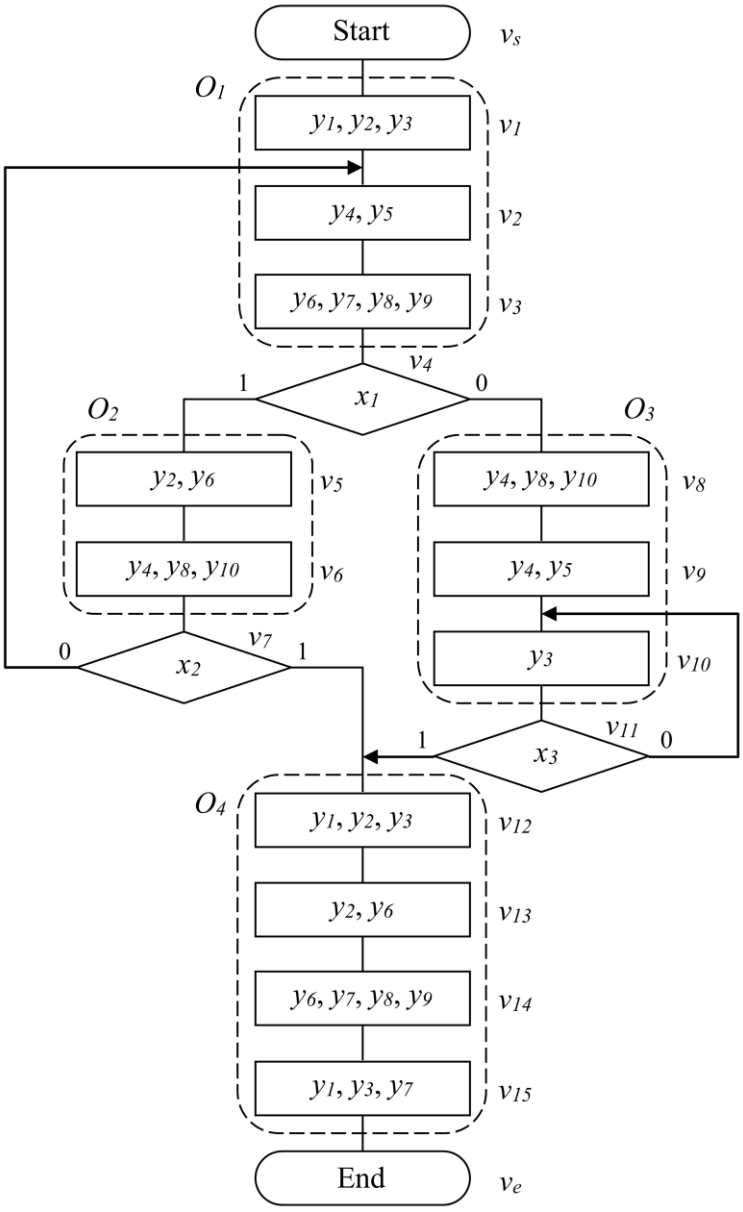

Рисунок 4. ГСА G_3 з виділеними операторними лінійними ланцюгами

ГСА G_1 з рис. 1 також містить 4 ОЛЛ, хоча й меншої довжини. В алгоритмі генерації псевдовипадкової ГСА, спрямованої на синтез КМПК, пропонується враховувати такі параметри.

Параметр 14. Кількість ОЛЛ в ГСА N_{OLC}. В загальному випадку $N_{OLC} \le N_{V_Y}$.

Параметр 15. Мінімальна довжина ОЛЛ $N_{OLC_{min}}$.

Параметр 16. Максимальна довжина ОЛЛ $N_{OLC_{max}}$.

Параметр 17. Середня довжина ОЛЛ $N_{OLC_{mid}}$.

Параметр 18. Математичне очікування довжини ОЛЛ N_{OLC_E}.

Параметр 19. Мінімальна кількість входів ОЛЛ $N_{ENT_{min}}$.

Параметр 20. Максимальна кількість входів ОЛЛ $N_{ENT_{max}}$.

У загальному випадку $1 \le N_{ENT_{min}} \le N_{ENT_{max}}$, $N_{OLC_{min}} \le N_{ENT_{max}} \le N_{OLC_{max}}$. За умови $N_{ENT_{max}} = 1$ усі ОЛЛ в ГСА стають простими.

Параметризація ГСА для різних методів оптимізації

Більшість методів оптимізації пристроїв керування спрямована на зменшення їх апаратурних витрат, що дає змогу знизити вартість і фізичні розміри схеми, підвищити її енергоефективність, надійність та швидкодію. Дослідження ефективності того чи іншого методу оптимізації проводяться зазвичай в умовах використання елементного базису, який є актуальним на момент розробки методу. У разі зміни елементного базису ефективність методів оптимізації схеми ПК треба перевіряти повторно. Також майже не дослідженою залишаються можливість одночасного застосування різних методів оптимізації та їх взаємний вплив. Такі дослідження можуть бути проведені з використанням множини ГСА, згенерованих у псевдовипадковий спосіб відповідно до заданих параметрів.

Розглянемо методи оптимізації, які спрямованими на мінімізацію апаратурних витрат у схемі ПК, зокрема: кодування наборів мікрооперацій, заміна логічних умов, кодування автоматних переходів, вертикалізація ГСА, формування класів псевдоеквівалентних станів, використання лічильника кодів станів та операційне перетворення кодів станів.

Кодування наборів мікрооперацій

Набором мікрооперацій називається множина мікрооперацій, що записана в одній операторній вершині ГСА [12, 16]. За своєю суттю набір мікрооперацій поступає в об’єкт управління і впливає одночасно на різні його елементи з метою виконання складної операції. Якщо об’єктом управління ПК виступає операційний автомат або арифметико-логічний пристрій, кожен набір мікрооперацій може ініціювати одну операцію над даними, як-от додавання, віднімання, кон’юнкція тощо. Протягом виконання свого алгоритму пристрій керування може неодноразово ініціювати виконання однієї і тієї ж дії об’єктом керування. На рівні ГСА це проявляється в тому, що декілька операторних вершин містять один і той самий набір мікрооперацій.

На рис. 4 ГСА G_3 містить 7 різних наборів мікрооперацій: $Y_1 = \{y_1, y_2, y_3\}$, $Y_2 = \{y_4, y_5\}$, $Y_3 = \{y_6, y_6, y_8, y_9\}$, $Y_4 = \{y_2, y_6\}$, $Y_5 = \{y_4, y_8, y_{10}\}$, $Y_6 = \{y_3\}$, $Y_7 = \{y_1, y_3, y_7, y_4\}$. Для кодування семи наборів достатньо трьох двійкових розрядів. Внаслідок цього схема ПК формує 3 розряди коду набору мікрооперацій замість 10 окремих мікрооперацій $y_1 - y_{10}$, після чого проводить дешифрування кодів наборів та формує окремі мікрооперації. Це за певних умов дає змогу зменшити складність схеми, що реалізує функцію виходів ПК.

Псевдовипадкова генерація наборів мікрооперацій потребує таких параметрів:

Параметр 21. Кількість наборів мікрооперацій у ГСА N_{SET}^{Y}. Цей параметр не може перебільшувати 2^{N_Y} (потужність булеану для множини мікрооперацій).

Параметр 22. Мінімальна кількість мікрооперацій в наборі $N^Y_{SET_{min}}$.

Параметр 23. Максимальна кількість мікрооперацій в наборі $N^Y_{SET_{max}}$.

Параметр 24. Середня кількість мікрооперацій в наборі $N^Y_{SET_{mid}}$.

Параметр 25. Математичне очікування кількості мікрооперацій в наборі $N^Y_{SET_E}$.

У загальному випадку $0 \le N^Y_{SET_{min}} \le N^Y_{SET_{max}}$, $N^Y_{SET_{min}} \le N^Y_{SET_{max}} \le N_Y$.

Заміна логічних умов

Заміна логічних умов застосовується в тих випадках, коли в ГСА аналізується велика кількість різних логічних умов, однак одночасно (в одному переході) аналізується лише декілька із них [12, 16–18]. Наприклад, у будівлі є 100 кімнат, в кожній з яких встановлені 3 датчики: датчик руху, датчик вібрації та датчик тепла. Пристрій керування сигналізацією циклічно перевіряє усі датчики в кожній кімнаті по черзі. Цикл перевірки кімнат повторюється нескінченну кількість разів. У разі спрацювання якогось датчика ПК реагує у певний спосіб. ГСА, що реалізує подібний алгоритм, буде мати 300 вхідних сигналів $x_1 - x_{300}$, однак одночасно будуть перевірятись лише 3, що відповідають датчикам окремої кімнати. Метод заміни вхідних змінних дає змогу використати в ГСА замість 300 логічних умов 3 псевдологічні умові p_1, p_2, p_3, які можуть перевірятись одночасно в одному мікропрограмному переході. Під час цього до структури ПК вводиться додаткова схема, яка перетворює 3 логічні умови, потрібні для аналізу на даному кроці, у сигнали p_1, p_2, p_3. Внаслідок цього замість 300 сигналів від логічних умов у схему ПК подаються 3 сигнали, що знижує складність схеми ПК. Кількість сигналів p_i визначається максимальною кількістю логічних умов, що аналізуються одночасно в межах ГСА. У ГСА G_1 на рис. 1 одночасно аналізуються максимум 2 логічні умови, а в ГСА G_3 на рис. 4 – лише одна логічна умова.

У ГСА кількість одночасно аналізованих логічних умов дорівнює кількості послідовно розташованих умовних вершин. Однак ми вже визначали подібні параметри під час розгляду мікропрограмних пристроїв керування. Отже, під час генерації ГСА зі спрямуванням на метод заміни вхідних змінних, можна використовувати раніше визначені параметри 10–13 та враховувати співвідношення (3). Потреби у введенні окремих параметрів ГСА для цього метода немає.

Кодування переходів автомата

Під час синтезу мікропрограмного автомата будується пряма структурна таблиця, кожен рядок якої відповідає окремому переходу автомата з одного стану в інший. Рядки прямої структурної таблиці кодуються унікальними двійковими кодами, на основі яких спеціальна схема формує відповідний код стану переходу та набір мікрооперацій. Такий підхід дає змогу спростити схему адресації і за певних умов зменшити апаратурні витрати на реалізацію схеми автомата [12, 16].

Переходи можуть бути безумовними (прямими) і умовними. Безумовний перехід не передбачає перевірки логічних умов і в ГСА виглядає як дуга між двома операторними

вершинами або між початковою і операторною вершинами чи між операторною і кінцевою вершинами. Умовний перехід – це послідовність із мінімум трьох послідовно зв'язаних вершин, у якій:

- першою вершиною є початкова або операторна вершина;
- останньою вершиною є операторна або кінцева вершина;
- між першою і останньою вершинами розташовані одна або декілька умовних вершин.

Очевидним параметром ГСА, що враховується методом кодування переходів, є загальна кількість переходів з одного стану в інший. Також можна окремо розглядати кількість або частку умовних і безумовних переходів, що опишемо такими трьома параметрами:

Параметр 26. Кількість переходів N_T.

Параметр 27. Кількість безумовних переходів N_{T_D}.

Параметр 28. Кількість умовних переходів N_{T_C}.

Вертикалізація ГСА

Суть вертикалізації полягає у приведенні ГСА до такого вигляду, що в кожній операторній вершині міститься не більше однієї мікрооперації [12, 16]. Така ГСА називається вертикальною і може бути отримана з початкової ГСА шляхом розщеплення операторних вершин із декількома мікрооперaціями на відповідну кількість послідовно розташованих операторних вершин з однією мікрооперацією в кожній. Далі усі мікрооперації у ГСА кодуються унікальними двійковими кодами. Ці коди формуються спеціальним блоком у схемі ПК, після чого здійснюється їх декодування і отримання сигналів мікрооперацій, що надходять в об'єкт керування.

Для формування вертикальної ГСА можна скористатись раніше введеними параметрами 4–7, взявши їх усі рівними одиниці. З іншого боку, можна не генерувати вже вертикалізовану ГСА, залишивши процес вертикалізації відповідному алгоритму синтезу автомата. Отже, додаткових параметрів ГСА у випадку зазначеного методу вводити не потрібно.

Формування класів псевдоеквівалентних станів

Формування класів псевдоеквівалентних станів застосовується тільки для МПА Мура і дає змогу замінити стани автомата класами, кількість яких менша за кількість станів автомата Мура і дорівнює кількості станів автомата Мілі [12, 13, 16]. Зменшення апаратурних витрат у схемі автомата можливе тоді, коли для кодування класів псевдоеквівалентних станів потрібна менша кількість розрядів, ніж для кодування станів автомата Мура.

В автоматі Мура псевдоеквівалентними вважаються стани, що відповідають операторним вершинам, виходи яких поєднані зі входом однієї вершини (операторної, умовної або кінцевої). На рис. 5 наведена ГСА G_4, в якій можуть бути виділені чотири класи псевдоеквівалентних станів: $B_1 = \{a_0\}$, $B_2 = \{a_1, a_2\}$, $B_3 = \{a_3, a_4, a_5\}$, $B_4 = \{a_6\}$. Для кодування чотирьох класів $B_1 - B_4$ достатньо двох двійкових розрядів, тоді як для

кодування семи станів $a_0 - a_6$ потрібно 3 розряди. Менша кількість розрядів у загальному випадку сприяє спрощенню схеми адресації МПА.

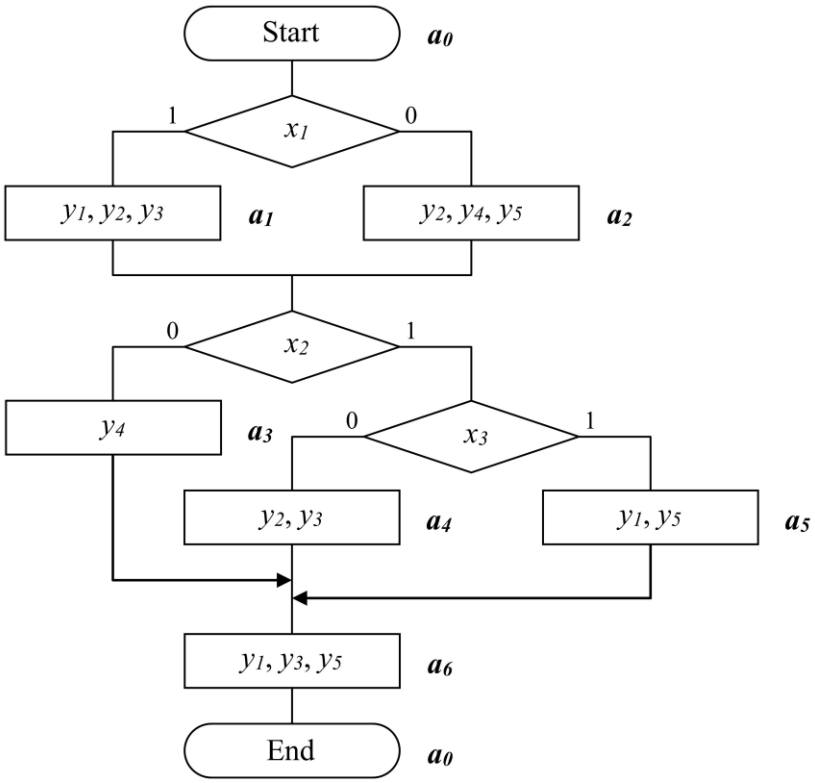

Рисунок 5. Граф-схема алгоритму G_4

Введемо параметри, які характеризують ГСА з погляду об'єднання виходів операторних вершин.

Параметр 29. Мінімальна кількість операторних вершин з об'єднаними виходами $N_{B_{min}}$.

Параметр 30. Максимальна кількість операторних вершин з об'єднаними виходами $N_{B_{max}}$.

Параметр 31. Середня кількість операторних вершин з об'єднаними виходами $N_{B_{mid}}$.

Параметр 32. Математичне очікування кількості операторних вершин з об'єднаними виходами N_{B_E}.

Використання лічильника кодів станів

Структура композиційного мікропрограмного пристрою керування містить у своєму складі лічильник мікрокоманд [12, 16]. Він реалізує природну адресацію мікрокоманд у межах операторних лінійних ланцюгів. Подібний підхід може бути застосований до мікропрограмних автоматів і породжує клас автоматів на лічильнику, в яких лічильник використовується замість звичайного регістра пам'яті [16, 18].

За аналогією з операторними лінійними ланцюгами в автоматах на лічильнику визначаються лінійні послідовності станів (ЛПС). ЛПС відрізняється від ОЛЛ тим, що в межах ЛПС допустиме існування як безумовних переходів, так і умовних переходів за одиничним значенням логічної умови. Такі переходи в автоматах на лічильнику виконуються за допомогою інкремента. На рис. 4 перехід із вершини v_3 у вершину v_4 здійснюється за умови $x_1 = 1$. Так само виконується перехід із вершини v_6 у вершину v_{12}. Якщо ГСА G_3 позначити станами автомата Мура, то стани, відповідні до вершин v_1, v_2, v_3, v_5, v_6, v_{12}, v_{13}, v_{14}, v_{15}, утворюють ЛПС і для коректної роботи автомата мають бути закодовані послідовними кодами. Друга ЛПС ГСА G_3 буде містити стани автомата Мура, еквівалентні вершинам v_8, v_9, v_{10}. Ці стани також мають бути закодовані послідовними кодами, відмінними від кодів станів першої ЛПС. Отже, в ГСА G_3 можуть бути виділені 4 операторні лінійні ланцюги та 2 лінійні послідовності станів. Ці структурні складники не пов'язані між собою і використовуються в різних методах синтезу.

Постає питання: чи є потреба у введенні додаткових параметрів, що характеризують ГСА в контексті ЛПС, чи запропонованих для ОЛЛ параметрів 14–20 достатньо? ОЛЛ і ЛПС – різні поняття, і їх генерація має відбуватися за різними алгоритмами. ОЛЛ і ЛПС використовуються в різних класах пристроїв керування. Генерація ГСА, в якій одночасно будуть *X* ОЛЛ та *Y* ЛПС, не завжди є можливою і не має практичної цінності. Якщо потрібно порівняти ефективності КМПК і автомата на лічильнику, достатньо згенерувати ГСА із заданою кількістю ОЛЛ або із заданою кількістю ЛПС, після чого синтезувати по цій ГСА обидва типи ПК і порівняти характеристики отриманих схем. Отже, для генерування ГСА для автоматів на лічильнику можна використовувати параметри 14–20, розуміючи під ОЛЛ саме лінійні послідовності станів. Впровадження додаткових параметрів ГСА в цьому випадку є зайвим.

Операційне перетворення кодів станів

Операційне перетворення кодів станів використовується в мікропрограмних автоматах. Переходи між станами автомата здійснюються шляхом виконання арифметико-логічних операцій над кодом поточного стану [19, 20]. Сьогодні не визначено параметри ГСА, які сприяють ефективності автоматів з операційним перетворенням кодів станів. Результатом синтезу таких пристроїв є зіставлення кожному автоматному переходу однієї арифметико-логічної операції із заданої множини операцій відповідно до певних вимог. Можна стверджувати, що чим більше переходів містить ГСА, тим складнішим є синтез такого МПА. Отже, саме кількість переходів ГСА постає визначальним критерієм. Це дає змогу спиратись під час псевдовипадкової генерації ГСА на раніше введені параметри 26–28 і не вводити додаткові параметри.

Обговорення отриманих результатів

Запропоновані 32 параметри граф-схем алгоритмів дають змогу описати ГСА в контексті різних класів пристроїв керування та методів оптимізації їх схем. Ці параметри можуть бути використані як початкові дані для програм псевдовипадкового генерування

граф-схем алгоритмів. Перелік параметрів не є остаточним і може доповнюватися з урахуванням інших структур пристроїв керування та методів їх синтезу.

Однак потрібно враховувати, що деякі з розглянутих параметрів можуть бути взаємовиключними. Наприклад, неможливо сформувати $N_{SET}^{Y} = 100$ наборів мікрооперацій, якщо загальна кількість різних мікрооперацій в ГСА $N_Y = 5$. Якщо користувач зазначив саме такі значення параметрів, програма, що генерує ГСА, повинна повідомити про помилку або запропонувати найбільш близькі значення параметрів, що не викликають конфлікту.

Генерація псевдовипадкових ГСА за заданими параметрами не є тривіальною задачею. Окрім відповідності параметрам, згенерована ГСА повинна мати такі загальні ознаки коректності, наприклад:

- не містити вершин, із яких неможливо досягти кінцевої вершини;
- не містити переходів у початкову вершину (замість них треба використовувати переходи у кінцеву вершину);
- не містити вершин, вхід або вихід яких не зв’язані з іншими вершинами або зв’язані з неіснуючими вершинами;
- не містити ланцюгів умовних вершин, у яких хоча б одна логічна умова зустрічається більше одного разу;
- містити рівно одну початкову та кінцеву вершини та хоча б одну операторну вершину.

Подібні перевірки рекомендується робити відразу після генерації ГСА. У разі виявлення помилок бажано мати можливість їх автоматичного виправлення. Також корисною функцією може бути візуалізація згенерованої ГСА з висвітленням тих чи інших особливостей генерації (наприклад, із виділенням операторних лінійних ланцюгів). Візуалізація ГСА може виконуватись окремим програмним модулем, початковими даними для якого виступатиме текстовий файл з описом згенерованої ГСА.

Дослідження методів синтезу цифрових пристроїв керування зазвичай потребують великої кількості тестових ГСА. Автоматизація методів синтезу дає змогу синтезувати ПК відповідно до ГСА, що містять сотні і тисячі станів або мікрокоманд. Псевдовипадкова генерація ГСА за заданими параметрами дасть можливість швидкого створення великої кількості різних ГСА зі схожими характеристиками, що зекономить час та підвищить якість і достовірність отриманих результатів.

Висновки

Сформовано множини уніфікованих параметрів граф-схем алгоритмів із метою використання їх у процесі псевдовипадкової генерації ГСА. Вибрані параметри враховують як базові структурні властивості ГСА, так і особливості різних класів ПК та методів оптимізації схем пристроїв.

Практична цінність роботи полягає у закладанні підґрунтя для створення алгоритмів і програмних засобів псевдовипадкової генерації ГСА, що дають змогу дослідити ефективність відомих та нових методів синтезу й оптимізації схем пристроїв керування.

Подальший напрям досліджень автори вбачають у розробці і дослідженні алгоритмів псевдовипадкової генерації ГСА відповідно за запропонованими в цій роботі множині параметрів.

Література

1. Bailliul, J., Samad, T. (2015). *Encyclopedia of Systems and Control*. Springer: London, UK, 1554 p. https://doi.org/10.1007/978-1-4471-5058-9
2. Suh, S. C., Tanik, U. J., & Carbone, J. N. (2013). *Applied Cyber-Physical Systems*. New York, USA: Springer, 253 p. https://doi.org/10.1007/978-1-4614-7336-7
3. Kam, T., Villa, T., Brayton, R., & Sangiovanni-Vincentelli, A. (1997). *A Synthesis of Finite State Machines: Functional Optimization*. Boston, MA, USA: Springer, 282 p. https://doi.org/10.1007/978-1-4757-2622-0
4. Baranov, S. (1994). *Logic Synthesis for Control Automata*. Dordrecht: Kluwer Academic Publishers, 312 p.
5. Hartmanis, J., Stearns, R. E. (1966). *Algebraic Structure Theory of Sequential Machines*. New Jersey: Prentice-Hall, 211 p.
6. Czerwinski, R., Kania, D. (2013). *Finite State Machine Logic Synthesis for Complex Programmable Logic Devices*. Berlin: Springer, 172 p. https://doi.org/10.1007/978-3-642-36166-1
7. Sklyarova, I., Sklyarov, V., & Sudnitson, A. (2012). *Design of FPGA-based Circuits using Hierarchical Finite State Machines*. Tallin: TUT Press, 240 p.
8. Nelson, V., Nagle, H., Irwin, J., & Carroll, B. (1995). *Digital logic circuit analysis and design*. New Jersey: Prentice Hall, 842 p.
9. DeMicheli, G. (1994). *Synthesis and Optimization of Digital Circuits*. NY: McGraw-Hill, 576 p.
10. Baranov, S. (2008). *Logic and System Design of Digital Systems*. TUTPress: Tallinn, Estonia, 267 p.
11. Scholl, C. (2001). *Functional Decomposition with Application to FPGA Synthesis*. Boston, MA, USA: Kluwer Academic, 264 p. https://doi.org/10.1007/978-1-4757-3393-8
12. Barkalov, A. A., Titarenko, L. A., Babakov, R. M., & Baiev, A. V. (2016). *Logic synthesis for FPGA-based finite state machines: monograph*. Vinnytsia: DonNU Vasyl' Stus, 195 p.
13. Baranov, S. (2018). *Finite State Machines and Algorithmic State Machines*. Seattle, WA, USA: Amazon, 185 p.
14. Yang, S. (1991). *Logic synthesis and optimization benchmarks User Guide Version 3.0.* Tech. rep. Microelectronics Center of North Carolina, P. O. Box 12889, Research Triangle Park, NC 27709.
15. Minns, P., Elliot, I. (2008). *FSM-Based Digital Design Using Verilog HDL*. New Jersey: J. Wiley & Sons, 408 p.
16. Barkalov, A., Titerenko, L. (2009). *Logic synthesis for FSM-based control units*. Berlin: Springer, 233 p.

17. Barkalov, A., Wegrzyn, M. (2006). *Design of control units with programmable logic*. Zielona Gora: University of Zielona Gyra Press, 150 p.
18. Barkalov, A., Titerenko, L. (2008). *Logic synthesis for compositional microprogram control units*. Berlin: Springer, 272 p.
19. Babakov, R. M. (2019). *Development and Hardware Optimization of Control Units for IOT Devices in Industry Systems*. Internet of Things for Industry and Human Applications. Vol. 1. Assessment and Implementation / V. S. Kharchenko (ed.). Ministry of Education and Science of Ukraine, National Aerospace University KhAI, p. 834–865.
20. Barkalov, A. A., Titarenko, L. A., & Babakov, R. M. (2023). *Synthesis of VHDL-Model of a Finite State Machine with Datapath of Transitions*. Radio Electronics, Computer Science, Control. Volume 4, p. 135–147. https://doi.org/10.15588/1607-3274-2023-4-13

Рукопис отримано – 04/03/2025 р.; прийнято до публікації – 25/03/2025 р.

Parametrization of graph-schemes of algorithms for digital control units

Roman Babakov, Alexander Barkalov

Abstract

The paper considers the scientific and practical problem of determining the set of parameters of graph-schemes of algorithms in the context of further pseudo-random generation of graph-schemes in order to study the effectiveness of methods for synthesizing and optimization of digital control units. Structural components are considered and general parameters of graph-schemes of algorithms are determined, which are traditionally used to describe algorithms for digital control units. The main classes of digital control units are analyzed, such as microprogram finite state machine (automated state machine with hardware logic implementation), microprogram control unit (automated state machine with programmable logic) and compositional microprogram control unit. For these classes, the main methods of optimizing hardware expenses are considered, including encoding sets of microoperations, replacing input variables, operational transformation of state codes, etc. For each of the considered classes of control units and optimization methods, sets of graph-schemes of algorithms parameters are proposed, which affect the effectiveness of applying the corresponding structures and methods and characterize both the transition function and the function of the outputs of control unit. For individual parameters, the permissible range of changes, correlation or mutual exclusivity with other parameters of graph-schemes are determined. Illustrative examples of determining individual parameters for a given graph-scheme are given. Recommendations are given on the use of the proposed parameters for pseudo-random generation of graph-schemes of algorithms. General requirements for correct pseudo-random generation of graph-schemes of algorithms are determined. Such requirements are: the possibility of reaching the final node from any other node; the absence of nodes whose input is not connected with the output of another node; the absence of repetition of logical conditions in consecutive conditional nodes; the presence of at least one operational node, etc.

Keywords: graph-scheme of algorithm; digital control units; circuit optimization; parameters; pseudo-random generation.

УДК 001.02

Експрес-ідентифікація реальних функцій українських наукових фахових видань на основі аналізу статистичних розподілів

Сергій Дмитрович Штовба
професор, д-р техн. наук
ORCID: 0000-0003-1302-4899
s.shtovba@donnu.edu.ua

Донецький національний університет імені Василя Стуса

Микола Володимирович Петричко
доктор філософії
ORCID: 0000-0001-6836-7843
mpetrychko@vntu.edu.ua

Вінницький національний технічний університет

Ключові слова:
наукові журнали;
редакційна колегія;
PhD-дисертація;
спеціальності;
датасет;
статистичний розподіл;
нерівномірність;
аномалії;
кореляція;
індекс Чекановського;
Україна.

В Україні налічується майже 1 700 наукових фахових видань, приблизно 10% з яких належать до категорії А. Наукові журнали, окрім базової функції з поширення нових знань та фіксації пріоритету наукового результату, виконують і обліково-залікові та комерційні функції. Під час управління академічною діяльністю виникають питання, чи потрібна саме така кількість національних наукових журналів, чи повною мірою вони виконують усі свої функції і наскільки ефективно. Для відповіді на ці питання необхідно знати не лише загальну кількість національних наукових журналів, але і їх спектр – належність до тих чи інших галузей та спеціальностей, а також їх зв'язок з іншими показниками академічної діяльності. Нами зібрано експериментальні дані і побудовано за ними статистичні розподіли кількості вітчизняних фахових видань, нормованої кількості вітчизняних фахових видань, кількості експертів Національного агентства із забезпечення якості вищої освіти та кількості захистів PhD-дисертації. Нормована кількість вітчизняних фахових видань розглядається як груба оцінка кількості фахових публікацій у розрізі спеціальностей, статистичні дані за якими нам недоступні. Усі розподіли виявилися нерівномірними з квартильними коефіцієнтами від 14.8 до 81.7. Схожість розподілів оцінено за індексом Чекановського. Виявлено високу схожість розподілу кількості фахових журналів, а відповідно – і дотичного до нього розподілу членів редколегій, з розподілом кількості експертів Національного агентства із забезпечення якості вищої освіти, що дає змогу висунути гіпотезу про подібність мотивів цих двох академічних спільнот. Імовірно, головним мотивом є набуття додаткової статусності, відчуття деякої престижності від входження в ці спільноти. Також виявлено високу схожість розподілів нормованої кількості фахових журналів та кількості захищених PhD-дисертацій, що дає змогу висунути гіпотезу про те, що фахові видання позиціонуються саме як майданчик для аспірантських статей.

DOI: https://doi.org/10.31558/2786-9482.2024.2.2

Вступ

Відповідно до «Порядку формування Переліку наукових фахових видань України» вітчизняні журнали та збірники наукових праць проходять експертизу в Атестаційній колегії МОНУ і отримують статус наукового фахового видання категорії А або Б. Наукові журнали є основним комунікаційним інструментом між вченими. Журнали ведуть свою історію з далекого січня 1665 р., коли у Франції вийшов перший номер літературно-наукового журналу *Journal des sçavans*. Майже відразу після цієї події в Англії з березня того самого року розпочали щомісячно видавати науковий журнал – *Philosophical Transactions of the Royal Society*. Англійський журнал видається і зараз, але вже не щомісяця, а щодватижні, і в двох серіях – A та B. За понад ніж 350-річнну історію наукових журналів склалися загальні принципи та функції такої комунікації, але час від часу з'являються нові задачі, нові цілі та способи їх досягнення [1].

Наукові журнали, окрім базової функції з поширення нових знань та фіксації пріоритету наукового результату, виконують і обліково-залікові та комерційні функції. Обліково-залікові функції вітчизняних фахових журналів пов'язані з тим, що за публікацію п'яти статей у таких журналах науково-педагогічному працівнику зараховується один із пунктів ліцензійних умов провадження освітньої діяльності. Для захисту PhD-дисертації потрібно опублікувати 3 статті у фахових виданнях або у журналах із баз Scopus чи Web of Science, для захисту докторської дисертації – 20 таких статей. На багатьох конкурсах кількість публікацій у фахових журналах враховується під час оцінювання проєктних та грантових заявок. Вимоги можуть бути і є неявними, наприклад, під час акредитації освітньої програми доцільно мати кілька статей за тематикою дисципліни, щоб у експертів не виникало сумнівів щодо профпридатності викладача. Афілійованість із фаховим журналом також надає деякі переваги членам редколегії – це, окрім престижу, ще й виконання одного із пунктів ліцензійних умов та додаткові бали в університетському рейтингу. Сам же університет за кожен фаховий журнал отримує бали під час атестації наукового напряму.

Описані обліково-залікові функції не є чимось специфічним, які притаманні виключно Україні. Наприклад, Міністерство науки і вищої освіти Польщі у 2004 р. запровадило перелік наукових журналів, за кожну статтю в яких нараховується від 20 до 200 балів. Список оновлюється щороку. У нього входять як польські національні видання, так і закордонні. Список доволі широкий – він займає 970 сторінок. Журнали в ньому згруповано за науковими напрямами, наприклад, за комп'ютерними науками, комп'ютерною інженерією та телекомунікаціями налічується 1735 видань. У деяких країнах національні та закордонні журнали явно диференціюються. Наприклад, в Індонезії [2] усі студенти та аспіранти перед складанням випускного іспиту зобов'язані опублікувати статті. Студенти бакалаврату мають опублікувати одну статтю в будь-якому журналі, студенти магістратури – одну статтю в національному акредитованому журналі, а аспіранти – одну статтю в національному акредитованому журналі та одну статтю в міжнародному журналі. Щороку в індонезійських журналах публікується щонайменше 150 000 статей, авторами яких є студенти та аспіранти місцевих університетів. В Індонезії все більше журналів проходять національну акредитацію. Якщо у 2013 р. із 5 900 журналів лише 342 журнали мали національну акредитацію [2], то у

2021 р. кількість акредитованих журналів становила 5 990, а загальна кількість індонезійських наукових журналів перевищила 14 000 [3].

Під час управління академічною діяльністю виникають питання, чи потрібна саме така кількість національних наукових журналів, чи повною мірою вони виконують усі свої функції і наскільки ефективно. Для відповіді на ці питання необхідно знати не лише загальну кількість національних наукових журналів, але і їх спектр – належність до тих чи інших галузей та спеціальностей, а також їх зв'язок з іншими показниками академічної діяльності. Щодо описової статистики, то в [3] зазначено кількість видавців індонезійських журналів, їх належністю до п'яти напрямів (освіта, соціальні науки, право, бізнес та менеджмент, аграрні та біологічні науки), а також динаміка кількості журналів. У статті [4] наводиться динаміка кількості литовських журналів за 2015–2022 рр. за природничими науками, гуманітарними дослідженнями, медициною і охороною здоров'я, соціальними науками, технологіями, аграрними науками та міждисциплінарними дослідженнями. Ще одним прикладом аналізу належності журналів до різних сфер науки є доповідь [5], у якій журнали розглядаються в межах системи класифікації видань бази Scopus. У тій доповіді, зокрема, проводиться аналіз динаміки кількості журналів за 2011–2018 рр., які індексовано у Scopus. Кількість журналів за ці роки зросла з 28 335 до 36 189. Розподіл журналів за галузями та спеціальностями розглядається без географічної прив'язки до країн, у яких ці журнали видають. Найбільша кількість журналів припадає на такі галузі: медицина, соціальні науки, інженерія, мистецтво та гуманітарні науки, біохімія, генетика і молекулярна біологія, аграрні та біологічні науки, природні науки, науки про Землю та планети, інформатика. У статті [6] проводиться аналіз схожості цитованих та цитуючих журналів на основі їх розподілів до наукових галузей. Для визначення схожості використовується індекс Чекановського. Знайдене значення схожості в роботі розглядається як потенційний індикатор міждисциплінарності наукових галузей. Подібні дослідження також проводились і у [7–10]. Отже, належність журналів до тих чи інших наукових галузей та спеціальностей аналізувалася, проте без виокремлення національної належності журналів. Публікацій, які розкривають взаємозв'язок між кількістю національних журналів та показниками академічної діяльності, знайти також не вдалося. Тому за *мету* роботи обрано збір статистичних даних та виявлення кореляцій між кількістю журналів та показниками академічної діяльності, щоб на їх базі сформувати гіпотези щодо реальних функцій, які виконують вітчизняні фахові журнали.

Датасет фахових видань

За переліком фахових видань на 10 грудня 2024 р. нами створено відповідний датасет [11]. У ньому, як і в інших датасетах нашого дослідження, використовується такий перелік спеціальностей:

011 – Освітні, педагогічні науки;

012 – Дошкільна освіта;

013 – Початкова освіта;

014 – Середня освіта;

015 – Професійна освіта;

016 – Спеціальна освіта;
017 – Фізична культура і спорт;
021 – Аудіовізуальне мистецтво та виробництво;
022 – Дизайн;
023 – Образотворче мистецтво, декоративне мистецтво, реставрація;
024 – Хореографія;
025 – Музичне мистецтво;
026 – Сценічне мистецтво;
027 – Музеєзнавство, пам'яткознавство;
028 – Менеджмент соціокультурної діяльності;
029 – Інформаційна, бібліотечна та архівна справа;
031 – Релігієзнавство;
032 – Історія та археологія;
033 – Філософія;
034 – Культурологія;
035 – Філологія;
041 – Богослов'я;
051 – Економіка;
052 – Політологія;
053 – Психологія;
054 – Соціологія;
061 – Журналістика;
071 – Облік і оподаткування;
072 – Фінанси, банківська справа та страхування;
073 – Менеджмент;
075 – Маркетинг;
076 – Підприємництво, торгівля та біржова діяльність;
081 – Право;
091 – Біологія та біохімія;
101 – Екологія;
102 – Хімія;
103 – Науки про Землю;
104 – Фізика та астрономія;
105 – Прикладна фізика та наноматеріали;
106 – Географія;
111 – Математика;
112 – Статистика;
113 – Прикладна математика;
121 – Інженерія програмного забезпечення;
122 – Комп'ютерні науки;
123 – Комп'ютерна інженерія;
124 – Системний аналіз;

125 – Кібербезпека та захист інформації;
126 – Інформаційні системи та технології;
131 – Прикладна механіка;
132 – Матеріалознавство;
133 – Галузеве машинобудування;
134 – Авіаційна та ракетно-космічна техніка;
135 – Суднобудування;
136 – Металургія;
141 – Електроенергетика, електротехніка та електромеханіка;
142 – Енергетичне машинобудування;
143 – Атомна енергетика;
144 – Теплоенергетика;
145 – Відновлювані джерела енергії та гідроенергетика;
161 – Хімічні технології та інженерія;
162 – Біотехнології та біоінженерія;
163 – Біомедична інженерія;
171 – Електроніка;
172 – Телекомунікації та радіотехніка;
173 – Авіоніка;
174 – Автоматизація, комп’ютерно-інтегровані технології та робототехніка;
175 – Інформаційно-вимірювальні технології;
176 – Мікро- та наносистемна техніка;
181 – Харчові технології;
182 – Технології легкої промисловості;
183 – Технології захисту навколишнього середовища;
184 – Гірництво;
185 – Нафтогазова інженерія та технології;
186 – Видавництво та поліграфія;
187 – Деревообробні та меблеві технології;
191 – Архітектура та містобудування;
192 – Будівництво та цивільна інженерія;
193 – Геодезія та землеустрій;
194 – Гідротехнічне будівництво, водна інженерія та водні технології;
201 – Агрономія;
202 – Захист і карантин рослин;
203 – Садівництво та виноградарство;
204 – Технологія виробництва і переробки продукції тваринництва;
205 – Лісове господарство;
206 – Садово-паркове господарство;
207 – Водні біоресурси та аквакультура;
208 – Агроінженерія;
211 – Ветеринарна медицина;

212 – Ветеринарна гігієна, санітарія і експертиза;
221 – Стоматологія;
222 – Медицина;
223 – Медсестринство;
224 – Технології медичної діагностики та лікування;
225 – Медична психологія;
226 – Фармація, промислова фармація;
227 – Фізична терапія, ерготерапія;
228 – Педіатрія;
229 – Громадське здоров'я;
231 – Соціальна робота;
232 – Соціальне забезпечення;
241 – Готельно-ресторанна справа;
242 – Туризм;
251 – Державна безпека;
252 – Безпека державного кордону;
253 – Військове управління;
254 – Забезпечення військ (сил);
255 – Озброєння та військова техніка;
256 – Національна безпека;
257 – Управління інформаційною безпекою;
261 – Пожежна безпека;
262 – Правоохоронна діяльність;
263 – Цивільна безпека;
271 – Морський та внутрішній водний транспорт;
272 – Авіаційний транспорт;
273 – Залізничний транспорт;
274 – Автомобільний транспорт;
275 – Транспортні технології;
281 – Публічне управління та адміністрування;
291 – Міжнародні відносини, суспільні комунікації та регіональні студії;
292 – Міжнародні економічні відносини;
293 – Міжнародне право.

Назва кожної спеціальності складається з цифрового шифру та змістовного словосполучення. Надалі на рисунках спеціальності позначаються цифровим шифром.

В Україні налічується 1 699 фахових видань, із них 175 належать до категорії А та 1 524 до категорії Б. Засновниками фахових видань є переважно університети та наукові установи. У деяких із них у портфелі 20, 30 і навіть понад 60 таких видань. Фахові видання розподілені за спеціальностями нерівномірно (рис. 1). У десятку надпопулярних спеціальностей увійшло 5 спеціальностей із галузі знань *07 Управління і адміністрування*, а також споріднені спеціальності *051 – Економіка* та *281 – Публічне управління і адміністрування*. Найбільш популярною є спеціальність *051 – Економіка*, за якою фахову категорію мають 275 видань. У

хвості рейтингу – *257 – Управління інформаційною безпекою* з одним фаховим виданням, а також спеціальності *027 – Музеєзнавство, пам'яткознавство*, *041 – Богослов'я*, *145 – Відновлювані джерела енергії та гідроенергетика*, *187 – Деревообробні та меблеві технології* та *252 – Безпека державного кордону*, кожна з яких має по 3 фахові видання. Медіана розподілу дорівнює 29, квартильний коефіцієнт дорівнює 14.8.

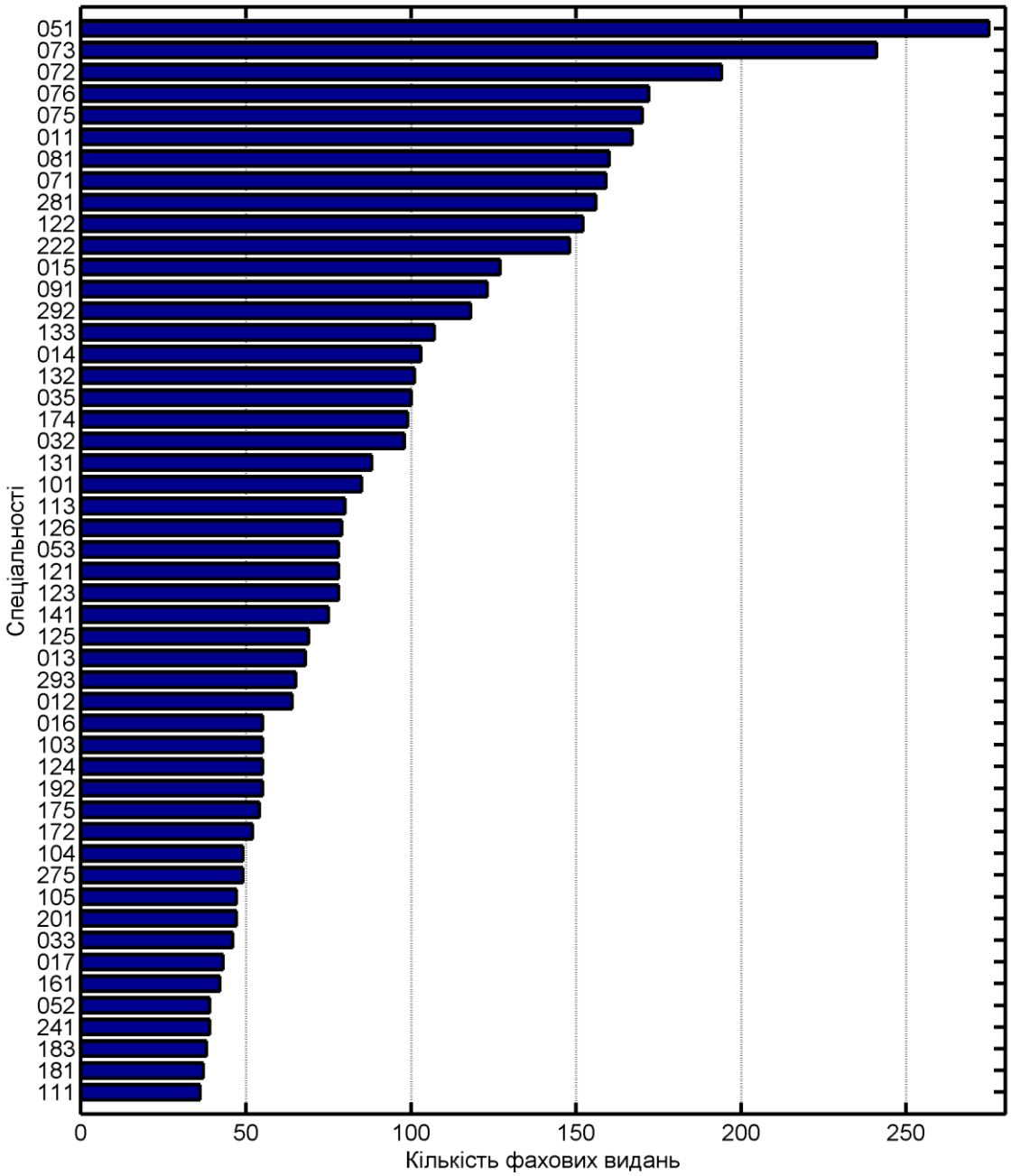

Рисунок 1. Топ-50 рейтингу спеціальностей за кількістю фахових видань

В одному фаховому виданні може бути представлено кілька спеціальностей (рис. 2). Широта тематики вітчизняних фахових видань варіюється від 1 до 45 спеціальностей. Для порівняння, найбільша кількість галузей знань, за якими видається один журнал з бази

Scopus, становить 9, а максимальна кількість спеціальностей – 13 [5]. Це дані на 2018 р., тоді в Scopus було 26 галузей та 307 спеціальностей.

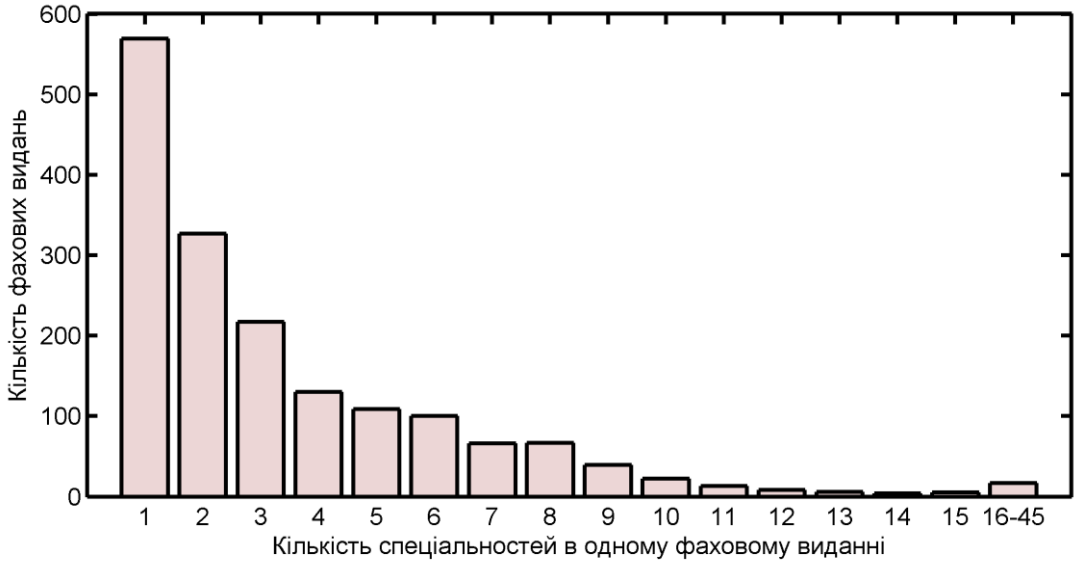

Рисунок 2. Розподіл фахових видань за кількістю спеціальностей

Майже 2/3 вітчизняних видань є фаховими не більше ніж за трьома спеціальностями. 60 спеціальностей представлено уно-виданнями, тобто виданнями лише за однією спеціальністю. Найбільше уно-видань (73) припадає на спеціальність *035 – Філологія*. Лідерами також є спеціальності *032 – Історія та археологія* та *222 – Медицина*, кількість уно-видань для яких становить 67 та 66. 103 спеціальності представлено уно- або дуо-виданнями. Найбільше таких видань (110) припадає на спеціальність *081 – Право*. Лідерами також є спеціальності *222 – Медицина* та *035 – Філологія*, кількість таких видань для них становить 97 та 89.

Для оцінювання кількості статей за тією чи іншою спеціальністю будемо вважати, що портфель статей фахового видання рівномірно розподілено за його спеціальностями. Тоді нормований внесок видання у потік публікацій за відповідними спеціальностями можна оцінити числом, оберненим до широти тематики. Наприклад, якщо видання є фаховим за трьома спеціальностями, тоді нормований внесок за кожною спеціальністю дорівнюватиме 0.33. З урахуванням такого нормування представимо модифікований рейтинг спеціальностей фахових видань (рис. 3). Лідери змінилися – ними стали такі спеціальності: *081 – Право, 222 – Медицина, 035 – Філологія* та *032 – Історія та археологія*. Замикають рейтинг спеціальності *257 – Управління інформаційною безпекою, 145 – Відновлювані джерела енергії та гідроенергетика* та *027 – Музеєзнавство, пам'яткознавство*. За кожною із аутсайдерських спеціальностей нормований внесок не дотягує і до половини одного уно-журналу. Медіана розподілу дорівнює 8, вилка розподілу – [0.1, 98], квартильний коефіцієнт – 20.1.

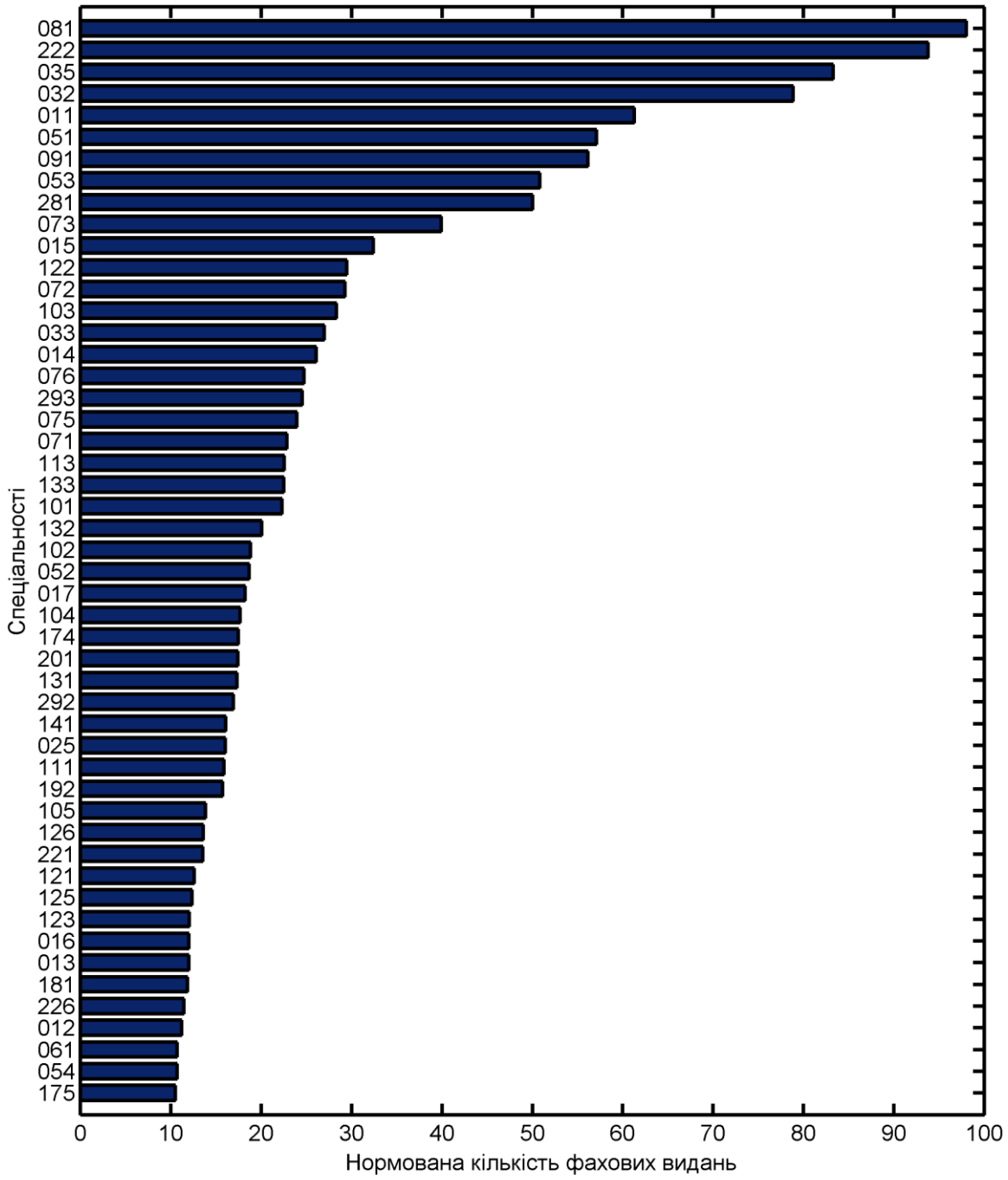

Рисунок 3. Топ-50 рейтингу спеціальностей за нормованою кількістю фахових видань

Оцінимо кількість науковців, які задіяні в редколегіях фахових видань. Відповідно до «Порядку формування Переліку наукових фахових видань України» кожна спеціальність має бути представлена щонайменше трьома членами редколегії, а кожна галузь знань – сімома. З урахуванням цих вимог на рис. 4 наведено рейтинговий розмір мінімальної кількості членів редколегії за фаховими виданнями. Вилка розподілу дорівнює [7, 135], а медіана – 14. Сумарна кількість членів редколегій усіх журналів дорівнює 25 642. Це оцінка знизу – в багатьох журналах редколегія сформована з деяким запасом. Але є також обмеження на те, щоб науковець не входив у редколегії понад трьох фахових видань. Кількість науковців, які

задіяні у редколегіях усіх вітчизняних фахових видань, можна оцінити у 17 000–20 000 осіб. Це багато. Виникає питання чим же мотивована така тяга науковців до роботи в редколегіях, з урахуванням того, що ця діяльність зазвичай не оплачується? Для формування відповідної гіпотези нам потрібні інші розподіли науковців за спеціальностями.

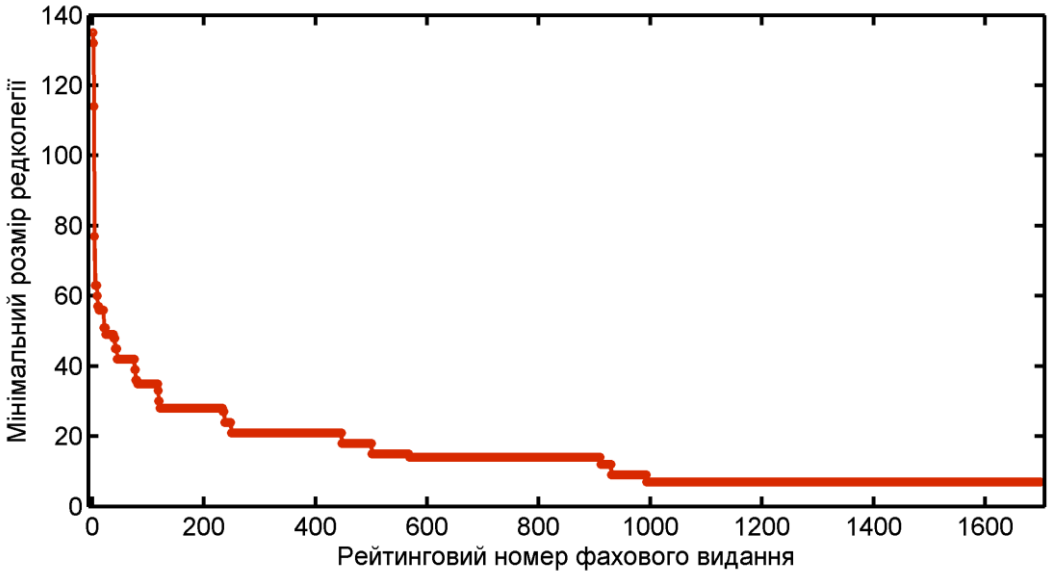

Рисунок 4. Розподіл мінімальної кількості членів редколегії фахового видання

Датасет експертів НАЗЯВО

НАЗЯВО – це скорочена назва Національного агентства із забезпечення якості вищої освіти. Ключовою функцією НАЗЯВО є акредитація освітніх програм, яку виконують експерти агентства. Експертів НАЗЯВО можна вважати активним прошарком академічної спільноти, яка, окрім усього іншого, зацікавлена і у статусних атрибутах. Бути експертом НАЗЯВО певною мірою престижно, як і бути членом редколегії фахового видання.

Нами створено датасет експертів НАЗЯВО [12]. Первинними документами для формування датасету є *публічні* реєстри НАЗЯВО – «Реєстр експертів з числа науково-педагогічних, наукових працівників» та «Реєстр експертів з числа здобувачів вищої освіти» на 1 лютого 2024 р. Верхня частина рейтингового розподілу кількості експертів за спеціальностями наведена на рис. 5. Вилка розподілу становить [0, 382], медіана дорівнює 33.5, а квартильний коефіцієнт – 17. Більш детально датасет описано в [13].

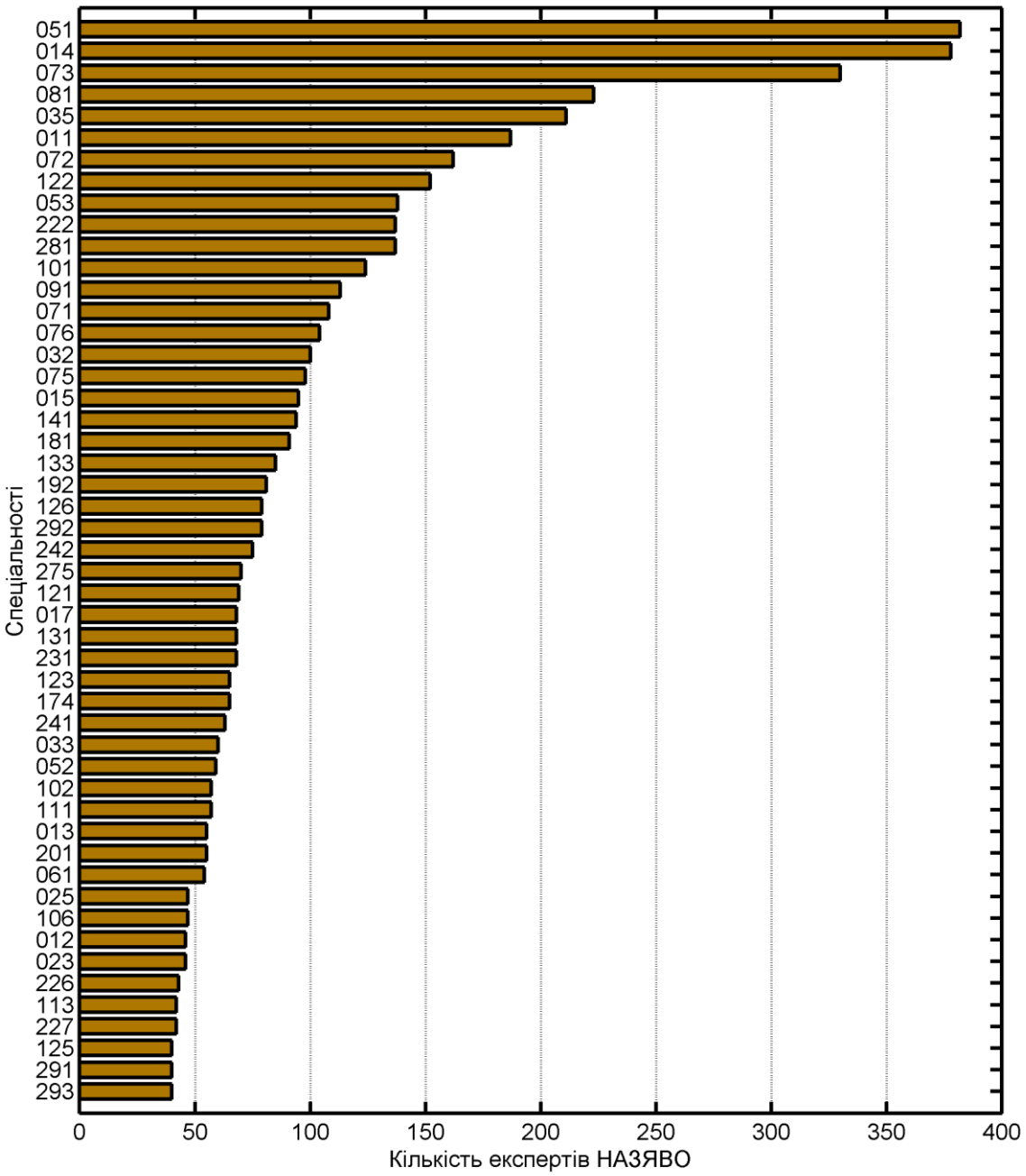

Рисунок 5. Топ-50 рейтингу спеціальностей за кількістю експертів НАЗЯВО

Датасет PhD-дисертацій

Як зазначено у Вступі, для захисту PhD-дисертації потрібно опублікувати 3 статті у фахових виданнях або у журналах із баз Scopus чи Web of Science. Отже, є деякий нормативний зв'язок захистів дисертацій з фаховими виданнями. Для досліджень статистичних проявів такого зв'язку нами створено датасет захищених в Україні PhD-дисертацій [14]. Він базується на даних інформаційної системи NAQA.svr, у якій НАЗЯВО веде облік захистів PhD-дисертацій. Інформаційна система запрацювала в першій половині 2022 р. Відтоді до 28 лютого 2025 р. успішно захищено 7 130 дисертацій. Розподіл захистів за спеціальностями дуже нерівномірний (рис. 6) – на топ-5 спеціальностей припадає 40%

захистів. Сильно виділяються спеціальності *081 – Право* та *222 – Медицина.* За спеціальностями *041 – Богослов'я*, *173 – Авіоніка* та *229 – Громадське здоров'я* захищено лише по одній дисертації. Відсутні дані про захисти за спеціальностями *027 – Музеєзнавство, пам'яткознавство*, *028 – Менеджмент соціокультурної діяльності*, *135 – Суднобудування*, *145 – Відновлювані джерела енергії та гідроенергетика*, *194 – Гідротехнічне будівництво, водна інженерія та водні технології*, *208 – Агроінженерія*, *232 – Соціальне забезпечення*, *241 – Готельно-ресторанна справа*, *256 – Національна безпека*, *257 – Управління інформаційною безпекою* та *262 – Правоохоронна діяльність*. Медіана розподілу дорівнює 20.5, а квартильний коефіцієнт – 81.7.

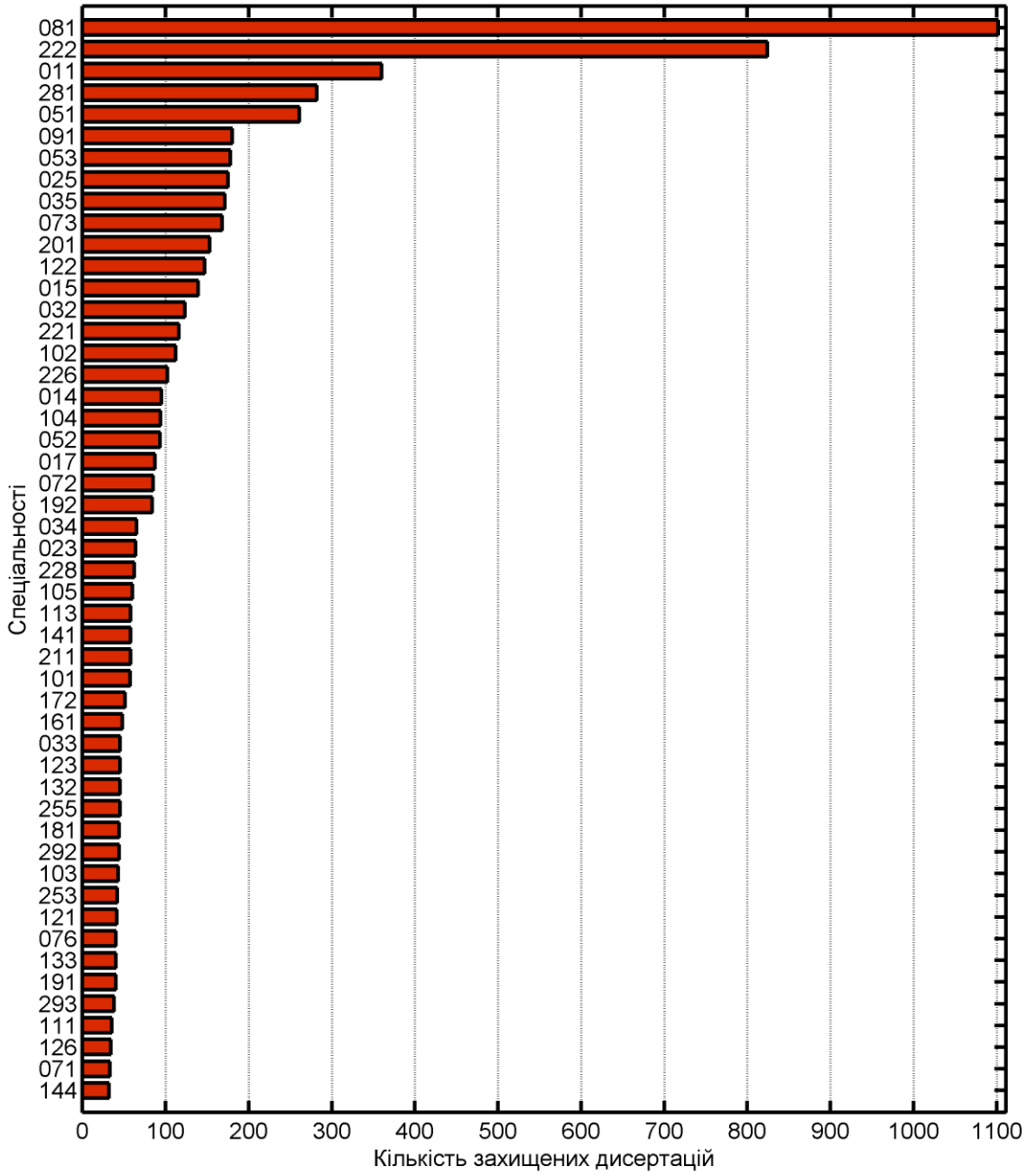

Рисунок 6. Топ-50 рейтингу спеціальностей за кількістю захистів PhD-дисертацій

Аналіз кореляцій

Для оцінювання попарної схожості розподілів переведемо їх у відносні одиниці – у відсоткові значення часток. Як метрику схожості двох розподілів розподіли X та Y оберемо індекс Чекановського [6, 15]. Для нашого випадку розподіли X та Y задані відсотковими значеннями часток $(\mu_1(X), \mu_2(X), ..., \mu_m(X))$ та $(\mu_1(Y), \mu_2(Y), ..., \mu_m(Y))$, де m=122 – кількість спеціальностей. У цьому випадку індекс Чекановського розраховуємо так: $F(X,Y) = \sum_{i=\overline{1,m}} \min(\mu_i(X), \mu_i(Y))$. Результати розрахунків зведемо у табл. 1.

Таблиця 1. Схожість за індексом Чекановського

Перший розподіл	Другий розподіл	
	Кількість експертів НАЗЯВО	Кількість PhD-дисертацій
Кількість фахових видань	0.791	0.606
Нормована кількість фахових видань	0.755	0.719

З табл. 1 видно, що розподіл експертів НАЗЯВО добре узгоджується як із розподілом фахових видань за спеціальностями, так і з розподілом нормованої кількості фахових видань. Із цього можна припустити, що експертів цікавить не лише можливість опублікувати свої статті у фаховому виданні, але і щось таке, що безпосередньо пов'язане з кількістю фахових журналів та збірників. Імовірно, це членство в редколегіях. Причому членство в редколегії для них навіть більше значуще, між можливість публікацій – розподіли більш подібні.

Точкова діаграма розподілів за спеціальностями фахових видань та експертів НАЗЯВО з відповідною лінійною регресійною залежністю між ними наведені на рис. 7. На ній помітно викид для спеціальності 014 – Середня освіта. Для цієї спеціальності кількість експертів НАЗЯВО аномально перевищує трендові значення, які розраховано на основі кількості фахових видань. Чим обумовлена така аномалія – нам невідомо. Можливо, є якась специфіка фахової діяльності чи освіти саме за цією спеціальністю, яка спонукає стати експертом НАЗЯВО.

Розподіл PhD-дисертацій узгоджується лише з розподілом нормованої кількості фахових видань. Отже, аспірантів фахові видання цікавлять лише як майданчик для опублікування статей. Точкова діаграма для цих двох розподілів з відповідною лінійною регресійною залежністю між ними наведені на рис. 8. На ній помітно 4 викиди. Для спеціальностей *081 – Право* та *222 – Медицина* кількість захистів значно перевищує трендові значення на основі нормованої кількості фахових видань. Це може бути пов'язано з тим, що за цими спеціальностями аспіранти більше публікуються в зарубіжних виданнях. Також можливо, що за цими спеціальностями щільність статей у фахових виданнях значно перевищує середні показники – або статті значно лаконічніші, або журнальні випуски роздуті. Для інших двох викидів – для спеціальностей *032 – Історія та археологія* та *035 – Філологія* – ситуація діаметрально протилежна. Для цих спеціальностей, навпаки, кількість захистів відстає від трендових значень на основі нормованої кількості фахових видань. Але ці висновки поверхові – усі 4 аномальні випадки потребують додаткового аналізу.

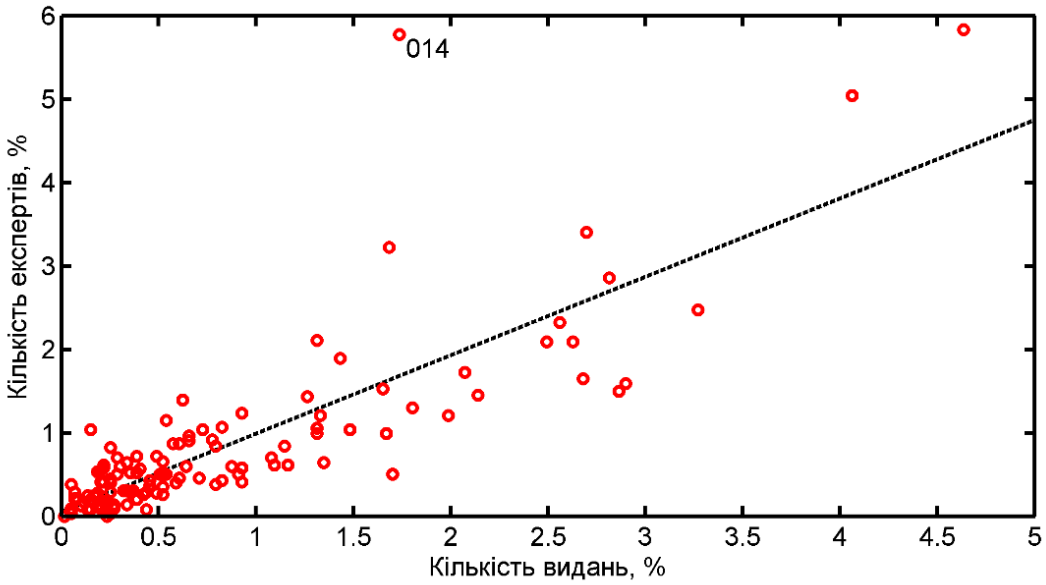

Рисунок 7. Зіставлення кількості фахових видань із кількістю експертів НАЗЯВО за відповідними спеціальностями

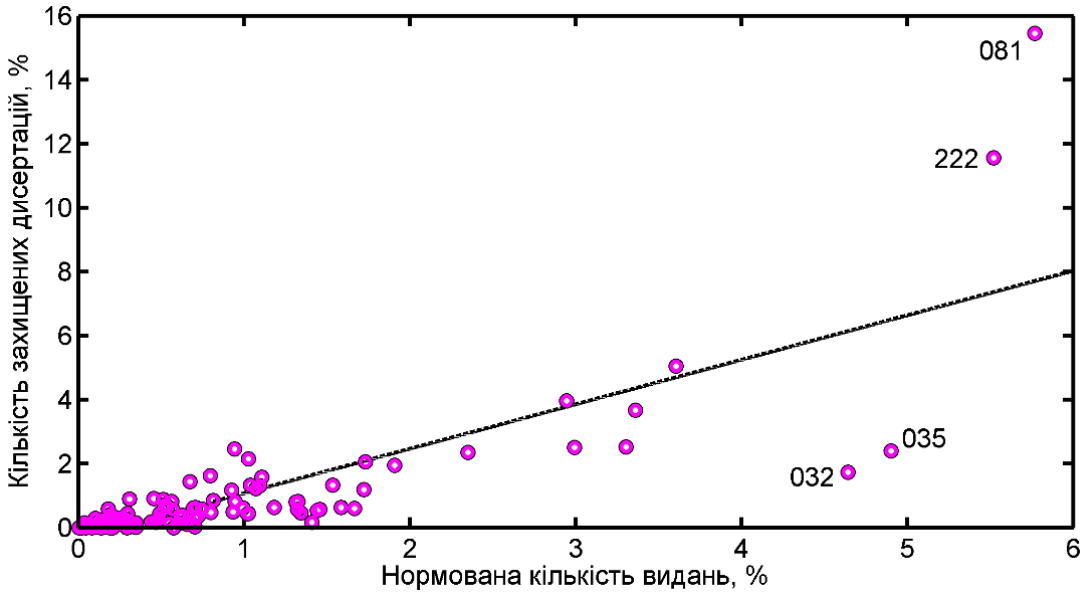

Рисунок 7. Зіставлення кількості PhD-дисертацій з нормованою кількістю фахових видань за відповідними спеціальностями

Висновки

Зібрано експериментальні дані і побудовано за ними статистичні розподіли кількості вітчизняних фахових видань, нормованої кількості вітчизняних фахових видань, кількості експертів НАЗЯВО та кількості захистів PhD-дисертації. Нормована кількість вітчизняних фахових видань розглядається як груба оцінка кількості фахових публікацій у розрізі спеціальностей, статистичні дані за якими нам недоступні. Розподіли побудовано за освітніми спеціальностями відповідно до переліку, що був чинним на 30 жовтня 2024 р. Усі

розподіли виявилися нерівномірними з квартильними коефіцієнтами від 14.8 до 81.7. Схожість розподілів оцінено за індексом Чекановського. Найбільш схожими виявилася розподіли кількості фахових видань та кількості експертів НАЗЯВО, а також розподіли нормованої кількості фахових видань та кількості захищених PhD-дисертацій.

Висока схожість розподілу кількості фахових видань, а, відповідно і дотичного до нього розподілу кількості членів редколегій, з розподілом кількості експертів НАЗЯВО дає змогу висунути гіпотезу про подібність мотивів цих двох академічних спільноти. Імовірно, головним мотивом є набуття додаткової статусності, відчуття деякої престижності від входження в ці спільноти. Лише представники спеціальності *014 – Середня освіта* випадають із загального тренду – кількість експертів НАЗЯВО за цією спеціальністю значно перевищує трендові значення, які розраховано на основі кількості фахових видань. Можливо, ця аномалія пов'язана зі специфікою фахової діяльності чи освіти саме за цією спеціальністю, яка спонукає до експертної роботи в НАЗЯВО.

Висока схожість розподілу нормованої кількості фахових журналів із розподілом кількості захищених PhD-дисертацій дає змогу висунути гіпотезу про те, що фахові видання позиціонуються як майданчик для аспірантських статей. За результатами зіставлення розподілів виявлено 4 викиди. Для спеціальностей *081 – Право* та *222 – Медицина* кількість захистів значно перевищує трендові значення, які розраховано на основі нормованої кількості фахових видань. Це може бути пов'язано з тим, що за цими спеціальностями аспіранти більше публікуються в зарубіжних журналах. Також можливо, що за цими спеціальностями щільність статей у фахових виданнях значно перевищує середні показники через більш лаконічні статті або більш роздуті номери. Для інших двох викидів – для спеціальностей *032 – Історія та археологія* та *035 – Філологія* – ситуація діаметрально протилежна.

Подальші дослідження можуть бути спрямовані на експериментальну перевірку сформульованих гіпотез, наприклад, шляхом експериментальної ідентифікації частки аспірантських публікацій, які припадають на вітчизняні фахові видання або персоналізованого аналізу складу редколегій. Але подібні дослідження потребують спеціальних програмних засобів для видобутку такої слабкоструктурованої інформації.

Література

1. Ярошенко, Т., & Ярошенко, О. (2024). Чи є майбутнє в наукових журналів? Зміни, виклики й тенденції в академічному видавництві. *Відкрита наука та іновації,* 1(2), 36–50. https://doi.org/10.62405/osi.2024.02.04
2. Wiryawan, K. G. (2014). The current status of science journals in Indonesia. *Science Editing*, *1*(2), 71–75. https://doi.org/10.6087/kcse.2014.1.71
3. Putera, P. B., Suryanto, S., Ningrum, S., Widianingsih, I., & Rianto, Y. (2021). Policies of scholarly journal accreditation in Indonesia. *Science Editing*, *8*(2), 166–171. https://doi.org/10.6087/kcse.250
4. Petreikis, T., Šuminas, A., Grigas, V., & Gudinavičius, A. (2024). Quantitative Changes in the Publishing of Lithuanian Scientific Journals in 2015–2022: Public and Private Sectors'

Adaptation to the Changing Conditions. *Knygotyra*, *82*, 206–242. https://doi.org/10.15388/Knygotyra.2024.82.8

5. Bordignon, F. (2019). Tracking content updates in Scopus (2011–2018): A quantitative analysis of journals per subject category and subject categories per journal. In *17th International Conference on Scientometrics and Informetrics, ISSI 2019 – Proceedings* (Vol. 2, pp. 1630–1640). International Society for Scientometrics and Informetrics.
6. Schubert, A. (2013). Measuring the similarity between the reference and citation distributions of journals. *Scientometrics*, *96*(1), 305–313. https://doi.org/10.1007/s11192-012-0889-0
7. Wolfram, D., & Zhao, Y. (2014). A comparison of journal similarity across six disciplines usingciting discipline analysis. *Journal of Informetrics*, *8*(4), 840–853. https://doi.org/10.1016/j.joi.2014.08.003
8. Yan Ma, Yingkun Han, Jiangtao Xu, Qian Chen, Shilin Zhao, & Zaiyi Zhang (2024). Analysis of the Distribution Characteristics of Journal Disciplines. In *Proceedings of the 2024 Fifth International Conference on Education, Knowledge and Information Management (ICEKIM 2024）* (pp. 1031–1041). Atlantis Press. https://doi.org/10.2991/978-94-6463-502-7_110
9. Aviv-Reuven, S., & Rosenfeld, A. (2023). A logical set theory approach to journal subject classification analysis: intra-system irregularities and inter-system discrepancies in Web of Science and Scopus. *Scientometrics*, *128*(1), 157–175. https://doi.org/10.1007/s11192-022-04576-3
10. Mongeon, P., & Paul-Hus, A. (2016). The journal coverage of Web of Science and Scopus: a comparative analysis. *Scientometrics*, *106*(1), 213–228. https://doi.org/10.1007/s11192-015-1765-5
11. Shtovba, S. & Petrychko, M. (2024). Ukrainian Journals 2025 Dataset. *ResearchGate*. https://doi.org/10.13140/RG.2.2.22868.51842
12. Shtovba, S. & Petrychko, M. (2024). NAQA Experts Dataset. *ResearchGate*. https://doi.org/10.13140/RG.2.2.16273.13922
13. Штовба, С. Д., & Петричко, М. В. (2024). Ідентифікація рівня спорідненості освітніх спеціальностей на основі аналізу профілей експертів НАЗЯВО. *Наукові праці Вінницького національного технічного університету*, (1). https://doi.org/10.31649/2307-5376-2024-1-48-59
14. Shtovba, S. (2024). Ukrainian PhD Theses Dataset. *ResearchGate*. https://doi.org/10.13140/RG.2.2.26405.00487
15. Bloom, S. (1981). Similarity Indices in Community Studies: Potential Pitfalls. *Marine Ecology Progress Series*, *5*, 125–128. https://doi.org/10.3354/meps005125

Рукопис отримано – 12/03/2025 р.; прийнято до публікації – 24/03/2025 р.

Express identification of real functions of Ukrainian scientific professional journals based on the analysis of statistical distributions

Serhiy Shtovba, Mykola Petrychko

Abstract

There are about 1,700 scientific professional journals in Ukraine, of which about 10% belong to the category A. Scientific journals, in addition to the basic function of disseminating new knowledge and fixing the priority of scientific results, also perform credit and commercial functions. When managing academic activities, questions arise as to whether such a number of national scientific journals is needed, whether they fully perform all their functions and how effectively. To answer these questions, it is necessary to know not only the total number of national scientific journals, but also their spectrum - belonging to certain research fields and specialties, as well as their connection with other indicators of scientific and educational activity. We have collected experimental data and based on it, we have constructed statistical distributions of the number of Ukrainian professional journals, the normalized number of Ukrainian professional journals, the number of experts of the National Agency for Higher Education Quality Assurance of Ukraine and the number of PhD thesis defenses. The normalized number of Ukrainian professional journals is considered as a rough estimate of the number of professional journals in terms of specialties, statistical data for which are not available to us. All distributions turned out to be uneven with quartile coefficients from 14.8 to 81.7. The similarity of the distributions was assessed using the Czekanowski index. A high similarity was found in the distribution of the number of professional journals, and, accordingly, the distribution of editorial board members tangent to it, with the distribution of the number of experts of the National Agency for Higher Education Quality Assurance of Ukraine, which allows the hypothesis to be put forward about the similarity of the motives of these two academic communities. Most likely, the main motive is the acquisition of additional status, a feeling of some prestige from joining these communities. A high similarity was found in the distribution of the normalized number of professional journals with the distribution of the number of defended PhD theses, which allows the hypothesis to be put forward that professional journals are positioned precisely as a platform for postgraduate articles.

Keywords: scientific journals; editorial board; PhD thesis; specialties; dataset; statistical distribution; unevenness; anomalies; correlation; Czekanowski index; Ukraine.

References

1. Yaroshenko, T., & Yaroshenko, O. (2024). Chy ye maibutnie v naukovykh zhurnaliv? Zminy, vyklyky y tendentsii v akademichnomu vydavnytstvi. Vidkryta nauka ta inovatsii, 1(2), 36–50. https://doi.org/10.62405/osi.2024.02.04
2. Wiryawan, K. G. (2014). The current status of science journals in Indonesia. *Science Editing*, *1*(2), 71–75. https://doi.org/10.6087/kcse.2014.1.71
3. Putera, P. B., Suryanto, S., Ningrum, S., Widianingsih, I., & Rianto, Y. (2021). Policies of scholarly journal accreditation in Indonesia. *Science Editing*, *8*(2), 166–171. https://doi.org/10.6087/kcse.250
4. Petreikis, T., Šuminas, A., Grigas, V., & Gudinavičius, A. (2024). Quantitative Changes in the Publishing of Lithuanian Scientific Journals in 2015–2022: Public and Private Sectors' Adaptation to the Changing Conditions. *Knygotyra*, *82*, 206–242. https://doi.org/10.15388/Knygotyra.2024.82.8
5. Bordignon, F. (2019). Tracking content updates in Scopus (2011-2018): A quantitative analysis of journals per subject category and subject categories per journal. In *17th International Conference on Scientometrics and Informetrics, ISSI 2019 – Proceedings* (Vol. 2, pp. 1630–1640). International Society for Scientometrics and Informetrics.
6. Schubert, A. (2013). Measuring the similarity between the reference and citation distributions of journals. *Scientometrics*, *96*(1), 305–313. https://doi.org/10.1007/s11192-012-0889-0

7. Wolfram, D., & Zhao, Y. (2014). A comparison of journal similarity across six disciplines usingciting discipline analysis. *Journal of Informetrics*, *8*(4), 840–853. https://doi.org/10.1016/j.joi.2014.08.003
8. Yan Ma, Yingkun Han, Jiangtao Xu, Qian Chen, Shilin Zhao, & Zaiyi Zhang (2024). Analysis of the Distribution Characteristics of Journal Disciplines. In *Proceedings of the 2024 Fifth International Conference on Education, Knowledge and Information Management (ICEKIM 2024）* (pp. 1031–1041). Atlantis Press. https://doi.org/10.2991/978-94-6463-502-7_110
9. Aviv-Reuven, S., & Rosenfeld, A. (2023). A logical set theory approach to journal subject classification analysis: intra-system irregularities and inter-system discrepancies in Web of Science and Scopus. *Scientometrics*, *128*(1), 157–175. https://doi.org/10.1007/s11192-022-04576-3
10. Mongeon, P., & Paul-Hus, A. (2016). The journal coverage of Web of Science and Scopus: a comparative analysis. *Scientometrics*, *106*(1), 213–228. https://doi.org/10.1007/s11192-015-1765-5
11. Shtovba, S. & Petrychko, M. (2024). Ukrainian Journals 2025 Dataset. *ResearchGate*. https://doi.org/10.13140/RG.2.2.22868.51842
12. Shtovba, S. & Petrychko, M. (2024). NAQA Experts Dataset. *ResearchGate*. https://doi.org/10.13140/RG.2.2.16273.13922
13. Shtovba, S. D., & Petrychko, M. V. (2024). Identyfikatsiia rivnia sporidnenosti osvitnikh spetsialnostei na osnovi analizu profilei ekspertiv NAZIaVO. *Naukovi pratsi Vinnytskoho natsionalnoho tekhnichnoho universytetu*, (1). https://doi.org/10.31649/2307-5376-2024-1-48-59
14. Shtovba, S. (2024). Ukrainian PhD Theses Dataset. *ResearchGate*. https://doi.org/10.13140/RG.2.2.26405.00487
15. Bloom, S. (1981). Similarity Indices in Community Studies: Potential Pitfalls. *Marine Ecology Progress Series*, *5*, 125–128. https://doi.org/10.3354/meps005125

УДК 004.89:004.41

Інженерія MLOps: метасинтез інструментів, практик та архітектур для автоматизації машинного навчання

Данило Олегович Ганчук
ORCID: 0009-0004-6474-3521
danilhanchuk@gmail.com

Криворізький державний педагогічний університет

Сергій Олексійович Семеріков
професор, старший дослідник, д-р пед. наук
ORCID: 0000-0003-0789-0272
semerikov@gmail.com

Криворізький державний педагогічний університет
Інститут цифровізації освіти НАПН України
Державний університет "Житомирська політехніка"
Криворізький національний університет
Академія когнітивних та природничих наук

Ключові слова:
MLOps;
автоматизація;
інструменти;
фреймворки;
архітектура;
розгортання моделей;
конвеєри машинного навчання;
метасинтез.

Автоматизація повного життєвого циклу моделей машинного навчання критично важлива для їх ефективного впровадження в робочому середовищі. За останні роки з'явилися різноманітні інструменти, фреймворки та архітектури для підтримки практик Machine Learning Operations (MLOps). У цій статті представлено метасинтез оглядів для комплексного аналізу технологій, що забезпечують реалізацію MLOps. Проведено порівняння можливостей та функцій, що пропонуються популярними комерційними та відкритими MLOps-платформами. Ідентифіковано шаблони в архітектурі MLOps та філософіях проєктування. Розглянуто роль контейнеризації, оркестрації, управління конфігурацією та автоматизації інфраструктури в ML-конвеєрах. Також обговорюються підходи до розгортання моделей у хмарі та на периферійних пристроях. Основні отримані результати: 1) проведено метасинтез систематичних оглядів для узагальнення знань щодо практик MLOps та визначено, що MLOps є перспективним підходом для ефективного розгортання моделей машинного навчання, який вимагає подальших досліджень; 2) проаналізовано зв'язки між принципами, процесами та практиками MLOp та запропоновано схему взаємозв'язків між ключовими принципами, етапами розробки і впровадження моделей та основними практиками MLOps; 3) визначено найбільш ефективні практики MLOps для розгортання моделей – безперервна інтеграція / доставка, версіонування моделей і даних, автоматизація конвеєрів машинного навчання, моніторинг продуктивності, управління експериментами, життєвим циклом, безпека та конфіденційність даних, пояснюваність моделей, управління якістю даних, конфігурацією, стратегії розгортання, автоматизація інфраструктури, співпраця, управління ризиками. Отримані результати мають теоретичну значущість в узагальненні та систематизації знань про практики MLOps і практичну значущість для впровадження та вдосконалення процесів MLOps в організаціях.

DOI: https://doi.org/10.31558/2786-9482.2024.2.3

Вступ

У сучасному світі машинне навчання стає все більш важливою технологією, яка знаходить застосування в різноманітних галузях, як-от фінанси, охорона здоров'я, промисловість, роздрібна торгівля тощо [1]. Незважаючи на значний прогрес у розробці алгоритмів та моделей машинного навчання, їх ефективне розгортання у виробничих середовищах залишається складним [2, 3]. Це обумовлено низкою факторів, як-от необхідність забезпечення масштабованості, відтворюваності, безпеки та надійності моделей, а також складністю інтеграції процесів розробки й експлуатації. Для вирішення цих проблем виникла методологія MLOps (Machine Learning Operations), яка має на меті застосування принципів та практик DevOps (Development & Operations) до процесів розробки й розгортання моделей машинного навчання [4, 5]. MLOps охоплює широкий спектр практик, як-от автоматизація конвеєрів машинного навчання, версіонування даних та моделей, моніторинг продуктивності моделей, управління експериментами тощо [6, 7]. Дослідження показують, що застосування практик MLOps дає змогу суттєво підвищити ефективність та надійність розгортання моделей машинного навчання у виробничих середовищах [8, 9].

Водночас, незважаючи на значний інтерес до теми MLOps з боку як науковців, так і практиків, все ще існують певні прогалини та невирішені проблеми. Зокрема, відсутні загальноприйняті стандарти за кращими практиками впровадження MLOps, недостатньо досліджені питання інтеграції MLOps з DataOps (Data Operations), ModelOps (Model Operations), AIOps (Artificial intelligence for IT operations) та з іншими підходами), а також існує потреба в розробці нових інструментів та платформ для автоматизації процесів MLOps [5, 10, 11].

Ця робота спрямована на вирішення актуальної проблеми визначення та аналізу практик MLOps, необхідних для ефективного розгортання моделей машинного навчання. Підставою для виконання роботи є необхідність систематизації та узагальнення знань щодо практик MLOps, а також потреба в розробці рекомендацій щодо їх впровадження в організаціях для підвищення ефективності та надійності розгортання моделей машинного навчання у виробничих середовищах.

Відповідно до мети визначено такі основні завдання дослідження:

1) виконати метасинтез систематичних оглядів для узагальнення знань щодо практик MLOps, необхідних для ефективного розгортання моделей машинного навчання;
2) проаналізувати зв'язки між принципами, процесами та практиками MLOps;
3) виявити найбільш ефективні практики MLOps для розгортання моделей машинного навчання.

Метасинтез практик MLOps

Основні поняття дослідження

DevOps набуває все більшого поширення, і компанії застосовують його методи в різних галузях [6]. У цьому контексті MLOps автоматизують робочі процеси машинного навчання,

як-от конвеєри, застосовуючи практики DevOps, а саме CI / CD (Continuous Integration / Continuous Deployment) для проєктів машинного навчання [6, 8, 12].

Згідно з [12], ключові практики MLOps можуть бути реалізовані за допомогою GitHub Actions і CML (Continuous Machine Learning). Хоча деякі робочі процеси автоматизують завдання ML за допомогою GitHub Actions і CML, наскрізні конвеєри MLOps виробничого рівня у проаналізованих проєктах із відкритим кодом на GitHub зустрічаються рідко. Практики більше зосереджені на звітності та метриках, ніж на перенавчанні чи розгортанні.

Методика дослідження

"*Систематичні огляди ... можуть забезпечити узагальнення стану знань у певній галузі, на основі яких можна визначити пріоритети майбутніх досліджень; вони можуть відповісти на питання, на які в іншому випадку не можуть дати відповіді окремі дослідження; вони можуть виявити проблеми в первинних дослідженнях, які повинні бути виправлені в майбутніх дослідженнях; і вони можуть породжувати або оцінювати теорії про те, як або чому відбуваються ті чи інші явища*" [13]. Основною метою систематичного огляду є сприяння прийняттю обґрунтованих рішень – рішень, які ґрунтуються на доведених фактах [13]. Основна відмінність між систематичним оглядом і оглядом літератури полягає в тому, яку роль відіграє ідея. Огляд літератури може бути керованим ідеєю: у цьому випадку всі джерела можуть бути відібрані для підтвердження певної ідеї. У систематичному огляді, попри наявність певної провідної ідеї, джерела відбираються за процедурою, метою якої є не підтвердження ідеї, а перевірка гіпотез та надання відповіді на дослідницькі питання. Результатом використання систематичного огляду як наукового методу дослідження є отримання нового знання.

23 лютого 2024 р. ми здійснили такий пошуковий запит до бази даних Scopus:

```
TITLE ( ( systematic OR review OR survey ) AND mlops )
```

Виявлено 5 документів (табл. 1), з яких 3 [5, 6, 8] належать до систематичних оглядів. Роботи [14] та [15] не є систематичними оглядами, проте відповідно до [16] їх зміст взято до уваги під час метасинтезу для об'єднання та тематичного аналізу результатів, отриманих у систематичних оглядах.

Метасинтез виконаємо за схемою з [17]:

1) визначення предмета дослідження – практики MLOps для ефективного розгортання моделей;
2) визначення релевантних джерел – систематичні літературні огляди [5, 6, 8] та огляд продуктів і постачальників MLOps [15];
3) ретельне вивчення з метою визначення спільного для аналізованих робіт часового проміжку, спільного та відмінного у меті, дослідницьких питаннях, джерелах, критеріях включення, виключення та якості, визначеннях MLOps та етапів MLOps;
4) визначення зв'язку між роботами через виявлення та групування ключових тем;
5) взаємна трансляція результатів різних робіт через визначення спільної термінології, пояснення суперечностей у результатах та узагальнення результатів із різних робіт;
6) синтез результатів;
7) оприлюднення метасинтезу.

Таблиця 1. Результати пошуку систематичних оглядів у Scopus

Джерело	Зміст огляду
[6]	Здійснено "багатоголосий" огляд літератури (multivocal literature review) – різновид систематичного огляду, який використовує як "білі" (статті, розділи книг тощо), так і "сірі" джерела (дописи в блогах, технічні документи, відео тощо). Метою авторів було *визначити інструменти* створення конвеєрів MLOps та проаналізувати їх основні характеристики та особливості. Автори дослідили функціональність 13 інструментів MLOps та показали, що більшість із них підтримують однакові функції, але застосовують різні підходи, які можуть надавати різні переваги залежно від вимог користувача
[8]	Здійснено систематичний огляд 30 статей з метою визначення практик, стандартів, ролей, моделей зрілості, викликів та інструментів MLOps. Результати дослідження дали змогу зробити висновок, що MLOps все ще перебуває на початковій стадії
[14]	Здійснено огляд застосувань MLOps у проєктах Data Science. За сучасних підходів MLOps наголос робиться на розробці та розгортанні моделі, тоді як організаційним аспектам (бізнес-розуміння, оцінювання) приділяється недостатня увага. Оскільки успіх проєкту з Data Science залежить не лише від технічних питань, автори пропонують у майбутніх дослідженнях продовжувати розвивати сферу MLOps шляхом подолання розриву між бізнес-цілями організації та способом, яким ці цілі представлені та моделюються за допомогою відповідних концепцій
[15]	Представлено огляд продуктів і постачальників MLOps, щоб надати аналітикам даних можливість проактивно налаштувати відповідну інфраструктуру машинного навчання. Встановлено, що як частина цифрової стратегії, машинне навчання стало поширеним інструментарієм і можливістю для багатьох компаній. Однак MLOps-аспекти часто залишаються в проєктах поза увагою доти, поки вони не будуть встановлені і не почнуть виконуватися в бізнес-середовищі
[5]	Проаналізовано відкриті питання, можливості й тенденції, з якими стикаються організації під час впровадження MLOps та AIOps, відповідні фреймворки й архітектури, а також сфери їх використання. Систематичний огляд 93 досліджень надав можливість виявити: 1) для успішної реалізації проєктів зі штучного інтелекту потрібні спільна культура та комбінація навичок програмної інженерії, науки про дані та DevOps; 2) для підтримки життєвого циклу MLOps / AIOps корисні контейнеризація, версіювання даних і моделей, FaaS (Function-as-a-Service) та безсерверні архітектури; 3) для перенавчання і повторного впровадження компонентів важливий моніторинг середовища; 4) AIOps використовуються переважно у складних середовищах, як-от технології 5G і 6G, тоді як MLOps більш поширені у традиційних промислових середовищах

Ретельне вивчення та визначення зв'язку між роботами

Розподіл оглядів за роками

Дві роботи [6, 8] належать до 2022 р., одна [5] – до 2023 р. Джерела, що аналізуються у [6], обмежені 2020 р., у [8] – 2021 р., а у [5] – 2023 р. До того ж робота [5] згадує роботу [8] як попередню.

Цілі оглядів

Спільні аспекти цілей розглянутих оглядів:

1) усі огляди [5, 6, 8] спрямовані на дослідження та узагальнення знань щодо методології MLOps, її практик, інструментів та викликів;
2) огляди [6, 8] мають на меті виявлення та аналіз інструментів MLOps, які використовуються для автоматизації процесів розробки та розгортання моделей машинного навчання;

3) огляди [5, 8] прагнуть надати розуміння щодо загального стану впровадження практик MLOps в індустрії та в академічній сфері.

Відмінні аспекти цілей оглядів:

1) огляд [6] більше фокусується на виявленні та аналізі функціональних можливостей інструментів MLOps для створення конвеєрів машинного навчання;
2) огляд [8] приділяє увагу ширшому колу аспектів MLOps, як-от практики, ролі, моделі зрілості, виклики, на додачу до інструментів;
3) огляд [5], окрім методології MLOps, також розглядає споріднену концепцію AIOps; більше уваги приділяється висвітленню можливостей, викликів та майбутніх трендів в обох областях.

Незважаючи на певні відмінності у фокусі та широті охоплення, всі розглянуті огляди об'єднує спільна мета – дослідити та узагальнити знання щодо методології MLOps, її практичного застосування, інструментів, викликів та стану впровадження для сприяння подальшому розвитку цієї області.

Дослідницькі питання оглядів

Спільні аспекти дослідницьких питань:

1) усі огляди [5, 6, 8] містять питання щодо інструментів та платформ, які використовуються для впровадження практик MLOps, автоматизації процесів розробки, розгортання та моніторингу моделей машинного навчання;
2) огляди [5, 8] включають питання щодо викликів та відкритих проблем, із якими стикаються організації під час впровадження MLOps;
3) огляди [5, 8] розглядають питання щодо можливостей та майбутніх трендів в області MLOps.

Відмінні аспекти дослідницьких питань:

1) огляд [6] містить більш специфічні питання щодо функціональних можливостей та особливостей інструментів MLOps для створення конвеєрів машинного навчання;
2) огляд [8] включає питання щодо ролей та обов'язків спеціалістів, залучених до впровадження MLOps, а також моделей зрілості для оцінки рівня автоматизації процесів розгортання моделей;
3) огляд [5], окрім MLOps, також розглядає питання, специфічні для методології AIOps, та приділяє увагу поточним і майбутнім сферам застосування цих підходів.

Дослідницькі питання розглянутих оглядів охоплюють широкий спектр аспектів MLOps, від інструментів та платформ до викликів, можливостей та сфер застосування. Незважаючи на деякі відмінності у фокусі питань, всі огляди прагнуть дослідити ключові компоненти й фактори, які впливають на впровадження та розвиток MLOps-практик в організаціях.

Інформаційні джерела оглядів

Спільні аспекти інформаційних джерел:

1) усі огляди [5, 6, 8] використовували електронні бази даних наукових публікацій для пошуку релевантних досліджень;
2) огляди [5, 6] включали в пошук як академічні (рецензовані), так і неакадемічні ("сірі") джерела літератури, як-от блоги, вебсайти, відео, репозитарії коду тощо.

Відмінні аспекти інформаційних джерел:

1) огляд [6] використовував Google Scholar для пошуку наукових публікацій та звичайний пошук Google для "сірої" літератури;
2) огляд [8] обмежився пошуком лише в академічних базах даних ACM Digital Library, IEEE Xplore, ScienceDirect та SpringerLink;
3) огляд [5] використовував декілька баз даних (Scopus, arXiv, Springer, IEEE), але основним джерелом була база даних Scopus.

Розглянуті огляди демонструють різні підходи до вибору інформаційних джерел. Деякі дослідження [5, 6] включають як академічні, так і неакадемічні джерела для отримання більш повної картини щодо практичного застосування MLOps. Інші [8] фокусуються виключно на рецензованих наукових публікаціях. Вибір джерел може впливати на охоплення й тип знайдених досліджень, а отже, – і на результати та висновки оглядів.

Критерії включення інформаційних джерел до оглядів

Спільні аспекти критеріїв включення:

1) усі огляди [5, 6, 8] включали дослідження, які безпосередньо стосуються теми MLOps, її практик, інструментів та застосування;
2) огляди [6, 8] розглядали дослідження, які описують досвід, практики, архітектуру або реалізацію інструментів та процесів MLOps.

Відмінні аспекти критеріїв включення:

1) огляд [6] включав дослідження, які описують компоненти мінімального життєвого циклу MLOps або представляють досвід та думки експертів щодо MLOps;
2) огляд [8] додатково включав дослідження, які оцінюють зрілість процесів MLOps, розглядають ролі та обов'язки в життєвому циклі розробки моделей машинного навчання, а також визначають виклики в розробці та впровадженні рішень MLOps;
3) огляд [5] мав більш загальні критерії включення, розглядаючи дослідження, опубліковані з 2018 р. до 2023 р., які містять нові ідеї і тісно пов'язані з темою MLOps та AIOps.

Незважаючи на деякі відмінності, критерії включення у розглянутих оглядах переважно зосереджені на дослідженнях, які безпосередньо стосуються MLOps, описують практичний досвід, інструменти та процеси, а також розглядають різні аспекти життєвого циклу розробки моделей машинного навчання. Огляди [5, 8] мають дещо ширші критерії, включно з дослідженнями, пов'язаними з оцінюванням зрілості, ролями та викликами MLOps.

Критерії виключення інформаційних джерел з оглядів

Спільні аспекти критеріїв виключення інформаційних джерел:

1) огляди [6, 8] виключали дослідження, які не надають достатньо деталей щодо архітектури, реалізації або застосування інструментів та процесів MLOps;
2) огляди [5, 8] виключали дослідження, опубліковані не англійською мовою.

Відмінні аспекти критеріїв виключення інформаційних джерел:

1) огляд [6] виключав дослідження, які просувають комерційні платформи MLOps без надання деталей щодо їх реалізації або використання;

2) огляд [8] виключав дослідження, які стосуються лише застосування моделей ML без розгляду аспектів MLOps, а також короткі статті, постери та дослідження без доступу до повного тексту;
3) огляд [5] виключав дослідження з недостатньою кількістю цитувань (залежно від року публікації), а також матеріали з обмеженим доступом (за передплатою) та статті, опубліковані в недостатньо надійних джерелах.

Розглянуті огляди застосовують різні критерії виключення для відсіювання досліджень, які не відповідають вимогам. Спільним є виключення досліджень із недостатнім описом MLOps процесів та інструментів, а також неангломовних публікацій. Відмінності полягають у додаткових критеріях, як-от виключення комерційних платформ без технічних деталей [6], коротких статей та постерів [8], а також матеріалів з обмеженим доступом та низькою кількістю цитувань [5].

Критерії якості інформаційних джерел в оглядах

Спільні аспекти критеріїв якості:

1) огляди [5, 8] оцінювали якість досліджень на основі повноти опису методології, контексту та результатів;
2) огляди [5, 8] враховували наявність обґрунтованих доказів та аргументів на підтримку висновків дослідження.

Відміні аспекти критеріїв якості:

1) огляд [6] використовував кількісні показники популярності (кількість зірок на Github, кількість переглядів на YouTube) для оцінювання якості та релевантності “сірої” літератури;
2) огляд [8] оцінював, чи представляє дослідження емпіричні результати, а не лише думки експертів, а також чи валідовані результати у належний спосіб;
3) огляд [5] використовував розширений набір критеріїв, включно з наявністю всебічного огляду літератури, перевіркою результатів на прикладах використання, кількістю досліджуваних питань, наявністю відкритого доступу, опублікуванням у журналах із високим імпакт-фактором та кількістю цитувань.

Розглянуті огляди застосовують різні підходи до оцінювання якості досліджень. Загальним є прагнення включати дослідження з повним описом методології та обґрунтованими результатами. Однак конкретні критерії якості різняться – від використання показників популярності для “сірої” літератури [6] до оцінки емпіричності результатів [8] та врахування бібліометричних показників, як-от імпакт-фактор журналу та кількість цитувань [5].

Взаємна трансляція результатів різних робіт та синтез результатів

Трансляція результатів – це процес інтерпретації та перенесення результатів з одного дослідження або контексту в інший. У межах метасинтезу трансляція результатів означає пошук спільних тем і висновків у різних дослідженнях та їх узагальнення.

Для взаємної трансляції результатів різних робіт, окрім систематичних оглядів [5, 6, 8], залучимо огляд продуктів та постачальників MLOps [15].

Визначення MLOps

Спільні аспекти визначень MLOps:

1) усі огляди [5, 6, 8, 15] розглядають MLOps як набір практик, принципів та процесів для автоматизації та управління життєвим циклом моделей машинного навчання;
2) огляди [5, 6] наголошують на використанні в MLOps підходів та практик з DevOps, як-от безперервна інтеграція, доставка та моніторинг;
3) огляди [8, 15] підкреслюють роль MLOps в операціоналізації рішень машинного навчання та передачі їх в промислову експлуатацію.

Відмінні аспекти визначень MLOps:

1) огляд [6] більше фокусується на технічних аспектах MLOps, як-от управління життєвим циклом моделей, автоматизація конвеєрів та моніторинг продуктивності;
2) огляд [8] розглядає MLOps як набір практик конкретно для операціоналізації рішень науки про дані (Data Science);
3) огляд [5] наголошує на використанні в MLOps принципів інженерії програмного забезпечення та машинного навчання для створення продуктів на основі моделей;
4) огляд [15] розглядає MLOps як окрему сферу, що фокусується на автоматизації життєвого циклу моделей машинного навчання як частини цифрової стратегії компаній.

Незважаючи на деякі відмінності в акцентах та формулюваннях, всі розглянуті огляди визначають *MLOps як підхід для управління, автоматизації та операціоналізації процесів розробки, розгортання та підтримки моделей машинного навчання на основі практик з інженерії програмного забезпечення та DevOps.* MLOps є ключовим компонентом для успішного впровадження рішень машинного навчання в промисловому середовищі.

Етапи робочого процесу MLOps

Спільні етапи робочого процесу MLOps:

1) усі огляди [5, 6, 8, 15] включають етапи збору та обробки даних, розробки та навчання моделей, а також розгортання моделей в робочому середовищі;
2) огляди [5, 6, 15] виділяють етапи моніторингу продуктивності та деградації моделей після розгортання як важливу частину робочого процесу MLOps;
3) огляди [6, 15] включають етап повторного навчання моделей на основі нових даних або за розкладом як частину життєвого циклу MLOps.

Відмінні аспекти етапів робочого процесу MLOps:

1) огляд [6] пропонує детальний розподіл етапів робочого процесу MLOps, включно з кроками вилучення, аналізу, очищення та трансформації даних, а також валідації моделей;
2) огляд [8] менше фокусується на детальних етапах, а більше на загальних функціях MLOps, як-от збір даних, трансформація, навчання та впровадження моделей;
3) огляд [5] групує етапи в ширші категорії, як-от управління даними, розподілене навчання, розгортання та моніторинг;
4) огляд [15] додатково виділяє етапи генерування прогнозів та управління моделями й даними як частину робочого процесу MLOps.

Незважаючи на різні рівні деталізації та групування, розглянуті огляди демонструють загальну узгодженість щодо основних етапів робочого процесу MLOps. Ці етапи охоплюють весь життєвий цикл моделей машинного навчання – від збору й обробки даних до

розгортання, моніторингу та повторного навчання моделей. Відмінності в представленні етапів відображають різні підходи до структурування та опису робочого процесу MLOps.

Фреймворки та архітектури, що сприяють впровадженню MLOps

Спільні фреймворки та архітектури, що сприяють впровадженню MLOps:

1) огляди [5, 6, 8] виділяють платформи та фреймворки з відкритим кодом, як-от MLflow, Kubeflow та TensorFlow Extended, як ключові компоненти екосистеми MLOps;
2) огляди [5, 6, 15] підкреслюють важливість використання хмарних обчислювальних платформ та сервісів, як-от AWS, Google Cloud та Azure, для розгортання та масштабування рішень MLOps;
3) огляди [5, 8] зазначають, що архітектури, які засновано на контейнеризації (наприклад, з використанням Docker) та оркестрації контейнерів (наприклад, за допомогою Kubernetes), є ключовими для забезпечення переносимості та масштабованості рішень MLOps.

Відмінні аспекти розглянутих фреймворків та архітектур:

1) огляд [6] додатково виділяє платформи для оркестрації конвеєрів MLOps, як-от Apache Airflow, Jenkins та Polyaxon;
2) огляд [8] згадує специфічні фреймворки та платформи, як-от Kafka-ML та MLModelCI, які використовуються для управління життєвим циклом моделей ML;
3) огляд [5] розглядає ширший спектр архітектурних підходів, включно з використанням периферійних обчислень (edge computing), безсерверних обчислень (serverless) та архітектур, керованих подіями (event-driven architectures);
4) огляд [15] фокусується переважно на пропрієтарних платформах та рішеннях від комерційних постачальників, як-от Iguazio, Domino Data Lab, Comet та Valohai.

Отже, існує багато фреймворків і архітектурних підходів, що сприяють впровадженню MLOps – від відкритих платформ та бібліотек до комерційних рішень і хмарних сервісів. Ключовими факторами є підтримка автоматизації, масштабованості, переносимості та інтеграції з наявними системами та інструментами. Вибір фреймворків і архітектур залежить від конкретних вимог та обмежень організації, а також від рівня зрілості її процесів MLOps.

Інструменти MLOps для створення конвеєрів машинного навчання та операціоналізації моделей

Інструменти MLOps – це програмні засоби, які автоматизують та спрощують різні етапи життєвого циклу моделі машинного навчання, від підготовки даних до розгортання та моніторингу. Вони включають платформи для організації експериментів, версіонування моделей, автоматизації конвеєрів, розгортання та моніторингу.

Операціоналізація моделей – це переведення моделі машинного навчання з етапу розробки та експериментування у стан, готовий до використання у виробничому середовищі. Це включає розгортання моделі, інтеграцію в бізнес-процеси, моніторинг продуктивності та підтримку. Важливим складником цього процесу є версіонування – практика відстеження та управління змінами в артефактах (як-от код, дані, моделі) з часом. Це дає змогу зберігати історію змін, повертатися до попередніх версій та співпрацювати над артефактами. У MLOps версіонування застосовується як до коду, так і до моделей та даних.

Спільними інструменти MLOps, згаданими в розглянутих оглядах, є такі:

1) у [6, 8] виділяють MLflow як популярну платформу з відкритим кодом для управління життєвим циклом моделей машинного навчання, експериментами та розгортанням;
2) у [6, 15] як інструменти для операціоналізації моделей згадують хмарні платформи AWS SageMaker, Google Cloud AI Platform та Azure Machine Learning;
3) у [5, 8] зазначають, що для розгортання моделей часто використовуються інструменти контейнеризації, як-от Docker, та оркестрації, як-от Kubernetes.

Відмінними аспектами розглянутих інструментів MLOps є такі:

1) огляд [6] надає детальний перелік інструментів для різних етапів конвеєра MLOps, включно з платформами оркестрації (Apache Airflow, Jenkins, Kubeflow, Polyaxon, Seldon Core та ін.) та розгортання (TensorFlow Extended);
2) огляд [8] додатково згадує інструменти Kubeflow, Polyaxon, Comet.ml, Kafka-ML, MLModelCI для управління конвеєрами та розгортання моделей;
3) огляд [5] більше фокусується на загальних категоріях інструментів, як-от системи управління експериментами, версіонування даних і моделей, автоматизація інфраструктури;
4) огляд [15] деталізує функціональність популярних комерційних платформ MLOps, як-от Iguazio, Domino Data Lab, Comet, Valohai та ін. (табл. 2).

Таблиця 2. Популярні платформи та продукти MLOps (на основі [15])

Платформа / продукт	Постачальник	Посилання
MLflow	MLflow	https://mlflow.org/
Google Cloud AI	Google	https://cloud.google.com/products/ai
Kaggle	Kaggle	https://www.kaggle.com/
SageMaker	Amazon	https://aws.amazon.com/sagemaker/
Cloud-Native Toolkit	IBM	https://develop.cloudnativetoolkit.dev/resources/workshop/ai/
Iguazio MLOps Platform	Iguazio	https://www.iguazio.com/
Azure Machine Learning	Microsoft	https://azure.microsoft.com/en-us/products/machine-learning
Huawei Cloud ModelArts	Huawei	https://www.huaweicloud.com/intl/en-us/product/modelarts.html
SparkCognition Generative AI Suite	SparkCognition	https://www.sparkcognition.com/products/sparkcognition-generative-ai-suite
Comet	Comet	https://www.comet.com/site/
Grid.AI	Grid.AI	https://www.grid.ai/
Modzy ModelOps Platform	Modzy	https://github.com/modzy
Valohai MLOps Platform	Valohai	https://valohai.com/
HPE Ezmeral ML Ops	Hewlett Packard Enterprise	https://www.hpe.com/us/en/software/ezmeral-ml-ops.html
Domino Enterprise MLOps Platform	Domino	https://domino.ai/

Отже, існує широкий спектр інструментів MLOps для створення конвеєрів машинного навчання та операціоналізації моделей – від відкритих платформ, як-от MLflow, до

комерційних рішень від хмарних провайдерів і спеціалізованих компаній. Вибір конкретних інструментів залежить від потреб і масштабу організації, а також сумісності з існуючим стеком технологій.

Основні функції, що надаються інструментами MLOps

Спільні функції інструментів MLOps:

1) у [5, 6, 8] зазначають, що інструменти MLOps зазвичай надають можливості для відстеження експериментів, версіонування моделей та даних;
2) у [6, 8, 15] підкреслюють важливість функцій автоматизації та оркестрації робочих процесів MLOps, як-от конвеєри навчання та розгортання моделей;
3) у [5, 6, 15] вказують на наявність в інструментах MLOps компонентів для моніторингу продуктивності та деградації розгорнутих моделей.

Відмінні аспекти функцій інструментів MLOps:

1) у [6] запропонована детальна класифікація функцій у межах таких трьох категорій:
   - загальні функції, що стосуються всіх етапів конвеєра MLOps (підтримка відкритого вихідного коду, масштабованість та еластичність, розширюваність, безхмарність або підтримка хмарних середовищ, управління метаданими, безперервна інтеграція та доставка та GUI-, CLI- та API-інтерфейси користувача);
   - функції управління даними (передача даних у реальному часі, зберігання даних, аналіз, очищення і трансформація даних, моніторинг даних, управління метаданими та забезпечення доступу до даних через API);
   - функції управління моделями (підтримка різних бібліотек та фреймворків машинного навчання, відстеження експериментів та версіонування моделей, реєстр моделей, автоматична оптимізація гіперпараметрів, тестування моделей, виявлення аномалій і дрейфу моделей, моніторинг продуктивності моделей, управління метаданими моделей та розгортання моделей через API);
2) у [8] додатково виділено функції автоматичної оптимізації гіперпараметрів моделей та підтримки мобільності для розгортання в різних середовищах;
3) у [5] зазначена важливість інтеграції інструментів MLOps із наявними системами та підтримки колаборативної роботи команд;
4) у [15] детально описано функції комерційних платформ MLOps, як-от:
   - розробка моделей – середовище для аналізу даних, розробки функцій, навчання та експериментів з моделями;
   - операціоналізація навчання моделей – створення повторюваних конвеєрів навчання та тестування моделей;
   - безперервне навчання моделей – автоматична підтримка частоти перенавчання моделей на основі розкладу, подій або ad-hoc запитів;
   - розгортання моделей – упаковка, тестування та розгортання навчених моделей у виробничому середовищі;
   - генерування прогнозів – надання прогнозів або класифікацій у режимі реального часу або пакетної обробки;

- моніторинг продуктивності моделей – відстеження ефективності та деградації моделей, попередження про необхідність перенавчання;
- управління даними та функціями – підтримка зберігання, обробки та доступу до даних і згенерованих функцій.

Отже, інструменти MLOps надають широкий спектр функцій для підтримки життєвого циклу моделей машинного навчання з акцентом на автоматизацію, відстеження експериментів, версіонування, моніторинг та розгортання моделей. Деякі інструменти пропонують більш спеціалізовані функції, як-от оптимізація гіперпараметрів або управління даними. Вибір інструменту з відповідним набором функцій залежить від конкретних потреб і цілей організації в області MLOps.

Способи розгортання моделей машинного навчання у виробничих середовищах

Способи розгортання моделей – це різні підходи та стратегії для інтеграції навчених моделей машинного навчання в робочі середовища, де вони можуть використовуватись для отримання прогнозів чи рішень на основі реальних даних. Це може включати розгортання на локальних серверах, у хмарі, на периферійних пристроях тощо.

Спільні способи розгортання моделей машинного навчання у виробничих середовищах:

1) у [5, 6, 15] зазначають, що моделі часто розгортаються з використанням контейнерних технологій, як-от Docker, що забезпечує мобільність та ізоляцію моделей;
2) у [6, 8, 15] вказують на поширеність розгортання моделей у хмарних середовищах з використанням платформ і сервісів AWS, Google Cloud та Azure;
3) у [5, 6] зазначають, що моделі часто розгортаються як вебсервіси з використанням REST API або інших протоколів для забезпечення доступу до прогнозів у режимі реального часу.

Відмінні аспекти способів розгортання моделей:

1) у [6] додатково описно розгортання моделей з використанням платформ оркестрації Apache Airflow, Jenkins, Kubeflow, MLflow, Polyaxon та Seldon Core, Valohai для автоматичного масштабування та управління контейнерами;
2) у [8] зазначено, що деякі інструменти MLOps, як-от MLflow, Kubeflow та Kafka-ML, мають вбудовані можливості для полегшення розгортання моделей у різних середовищах;
3) у [5] розглянуто розгортання моделей не лише в хмарі, але й на периферійних пристроях з використанням спеціальних фреймворків, як-от TensorFlow Lite та Core ML;
4) [15] надає такий детальний опис основних етапів та особливостей розгортання моделей машинного навчання з використанням конвеєрів CI / CD та підтримкою різних середовищ на прикладі комерційних платформ MLOps:
   - створення конвеєра CI / CD для моделей – платформи MLOps, як-от SageMaker, Azure ML та Databricks, дають змогу створювати конвеєри CI / CD для автоматизації процесу розгортання моделей, які включають етапи збирання, тестування і розгортання моделей, а також відстеження артефактів та управління версіями;
   - підтримка різних середовищ розгортання – платформи MLOps зазвичай підтримують кілька середовищ розгортання, як-от середовища розробки,

тестування та виробничі середовища; моделі можуть бути розгорнуті в різних середовищах із використанням відповідних конфігурацій та політик доступу;

- процес розгортання моделей – навчені моделі упаковуються в стандартизований формат (наприклад, Docker-контейнер) разом з необхідними залежностями; модель проходить через етапи тестування та валідації, щоб переконатися у її коректності та відповідності вимогам; після успішного проходження тестів модель розгортається в цільовому середовищі, зокрема для виробничого середовища можуть застосовуватись додаткові заходи безпеки та моніторингу;
- автоматизація та оркестрація розгортання – платформи MLOps використовують інструменти автоматизації, як-от Jenkins або GitLab CI / CD, для забезпечення безперервної інтеграції та розгортання моделей, яке може бути налаштоване на автоматичний запуск за певних подій, як-от оновлення коду моделі або поява нових даних;
- моніторинг та управління розгорнутими моделями – платформи MLOps надають інструменти моніторингу продуктивності та метрик розгорнутих моделей у режимі реального часу і у разі виявлення проблем або деградації моделі платформа може автоматично ініціювати перенавчання або відкат до попередньої версії.

Отже, найбільш поширеними способами розгортання моделей машинного навчання у виробничих середовищах є використання контейнерних технологій, хмарних платформ і сервісів, а також розгортання моделей як вебсервісів. Вибір конкретного підходу залежить від вимог до латентності, масштабованості та доступності моделей, а також від наявної інфраструктури та системи інструментів в організації.

Моделі зрілості для оцінювання рівня автоматизації розгортання моделей машинного навчання

Моделі зрілості MLOps – це концептуальні рамки, які описують різні рівні або стадії зрілості процесів MLOps в організації. Вони допомагають оцінити поточний стан практик MLOps, визначити області для вдосконалення та надати орієнтири для досягнення вищих рівнів зрілості.

У [8] розглянуто такі моделі зрілості для оцінювання рівня покращення процесу розробки рішень машинного навчання:

1) заснована на Capability Maturity Model та методиці Six Sigma модель зрілості [18] перевіряє, чи діяльність (а) має визначені цілі, (б) послідовно реалізована, (в) задокументована, (г) автоматизована, (д) вимірюється і відстежується, і (е) постійно вдосконалюється;
2) за моделлю [19] організації класифікуються за трьома рівнями зрілості розробки рішень машинного навчання, а саме такі, що орієнтовані на дані, що орієнтовані на модель, що орієнтовані на конвеєр;
3) за моделлю [20] є 5 етапів вдосконалення практик розробки – (а) ручний процес на базі науки про дані, (б) стандартизований процес експериментально-операційної симетрії, (в) автоматизований процес робочого процесу машинного навчання, (г) інтегровані процеси розробки програмного забезпечення та робочого процесу машинного

навчання, і (д) автоматизований і повністю інтегрований процес робочого процесу CD і машинного навчання.

4) у [21] запропоновано адаптацію моделі Capability Maturity Model з визначенням п'яти рівнів зрілості для кожної оціненої здатності – (а) початковий, (б) повторюваний, (в) визначений, (г) керований та (д) оптимізований.

Усі систематичні огляди [5, 6, 8] вказують, що рівень автоматизації процесів MLOps є одним із ключових факторів для оцінювання зрілості організації в цій сфері. Хоча розглянуті огляди не надають вичерпного опису моделей зрілості MLOps, вони підкреслюють важливість оцінювання рівнів автоматизації процесів розробки, тестування та розгортання моделей як ключового фактора зрілості організації. Адаптація моделей зрілості розробки програмного забезпечення до специфіки MLOps може бути ефективним підходом до оцінювання та вдосконалення процесів машинного навчання в організації.

Ролі та обов'язки, визначені в діяльності з операціоналізації моделей машинного навчання

Ролі та обов'язки в MLOps – це набір функцій та сфер відповідальності різних членів команди, залучених до процесу розробки, розгортання та підтримки моделей машинного навчання. Вони можуть включати дата-сайєнтистів, інженерів даних, інженерів MLOps, менеджерів проєкту тощо.

Метою метасинтезу ролей та обов'язків, визначених у діяльності з операціоналізації моделей машинного навчання, в оглядах [5, 6, 8, 15] було визначити ключових учасників процесу MLOps та їх функції.

Спільні ролі та обов'язки:

1) усі огляди [5, 6, 8, 15] згадують про залучення фахівців з даних та наукових співробітників з даних, які відповідають за розробку, навчання та експерименти з моделями машинного навчання;
2) у [5, 6, 8, 5] виділяють роль інженерів з даних та провайдерів даних, які займаються вилученням, обробкою, перетворенням та забезпеченням якості даних для навчання моделей;
3) у [5, 6, 15] зазначають важливість інженерів DevOps, інженерів ML/MLOps та інженерів програмного забезпечення в операціоналізації моделей, автоматизації процесів розгортання, створенні конвеєрів та управлінні середовищами;
4) у [5, 6] підкреслюють роль менеджерів, керівництва та бізнес-зацікавлених сторін у визначенні вимог до моделей, прийнятті рішень та підтримці стратегії MLOps.

Відмінні аспекти ролей та обов'язків:

1) у [8] додатково виділено такі ролі:
   - *фахівець з предметної галузі* (domain specialist), який має глибокі знання предметної галузі та відіграє важливу роль в отриманні даних та валідації результатів;
   - *науковець або інженер з обчислювальних наук* (computational scientist / engineer), який має високі технічні навички для підготовки середовища для роботи моделей машинного навчання;

- *науковець або інженер з машинного навчання* (machine learning scientist / engineer), який має поглиблені знання статистики та алгоритмів машинного навчання та відповідає за проєктування нових моделей машинного навчання;
- *провайдер* (provenance specialist), який має знання як предметної галузі, так і машинного навчання, та керує постачанням даних у життєвому циклі розробки рішень машинного навчання;
- *менеджер* (manager), який оцінює моделі перед їх оприлюдненням;
- *розробник додатків* (application developer), який розробляє додатки, в яких будуть реалізовано створені моделі;
- *керівник з розгортання* (deployment lead), який оцінює компоненти інфраструктури для розгортання моделей машинного навчання у виробництво;

2) у [5] згадується роль експертів предметної галузі в розміченні даних у специфічних доменах.

Незважаючи на деякі відмінності в деталізації ролей, розглянуті огляди визнають необхідність залучення фахівців з різних сфер – з розробки програмного забезпечення, інженерії даних, машинного навчання, експертів предметної області та менеджменту, для успішної операціоналізації моделей машинного навчання. Тісна співпраця та комунікація між цими ролями є критичною для реалізації MLOps практик в організаціях (рис. 1).

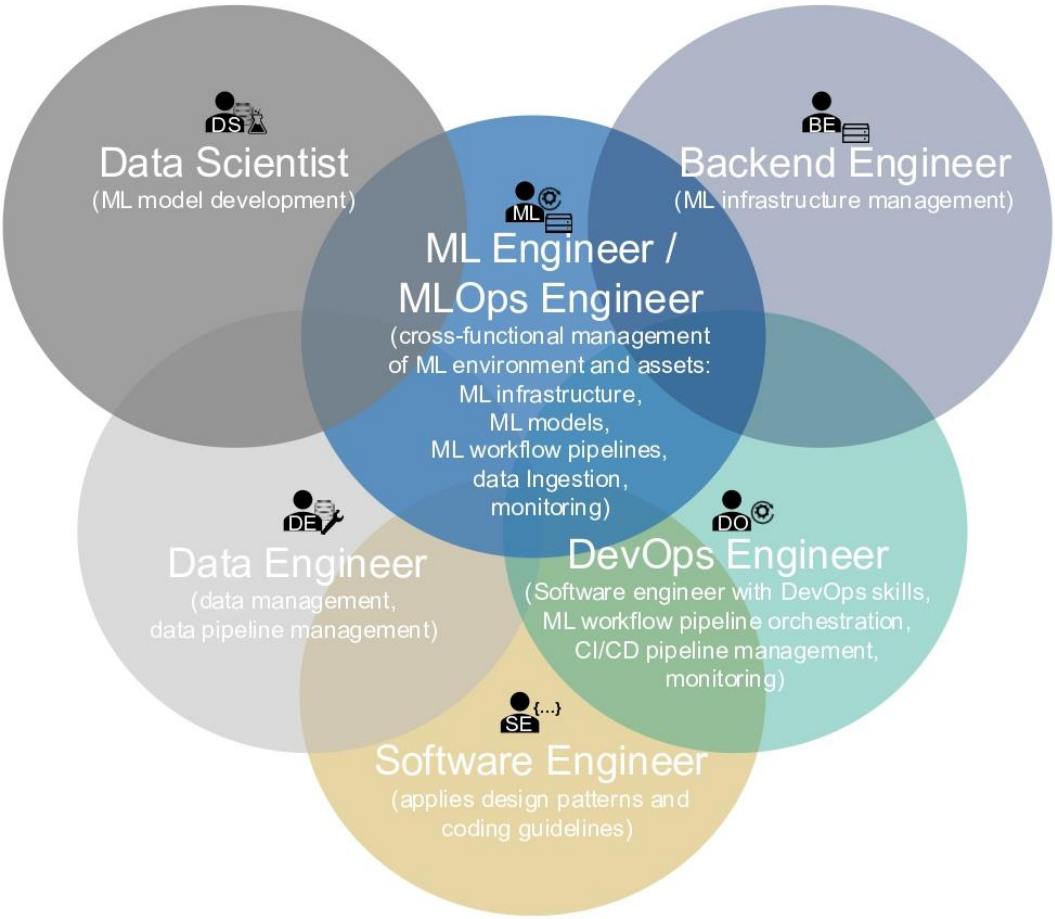

Рисунок 1. Перетин ролей та обов’язків [2]

Виклики розгортання моделей машинного навчання у виробничих середовищах

Виклики в MLOps – це труднощі та перешкоди, які виникають під час впровадження моделей машинного навчання в робочі середовища. Вони можуть стосуватися технічних аспектів (наприклад, забезпечення масштабованості та продуктивності), організаційних питань (наприклад, співпраці між командами), а також дотримання регуляторних вимог та етичних норм.

У [6] та [15] явно не наводяться конкретні виклики, однак їх можна визначити, базуючись на обговоренні MLOps та автоматизації конвеєрів машинного навчання у [6] та описі різних етапів MLOps і необхідності у відповідних інструментах у [15].

Спільні виклики:

1) у [5, 6, 8, 15] зазначено складності управління життєвим циклом моделей машинного навчання, зокрема версіонування, відстеження та відтворюваність моделей і даних, а також проблему забезпечення масштабованості та продуктивності моделей в реальних умовах використання з великими обсягами даних та запитів;
2) у [5, 8, 15] вказано на виклики моніторингу та підтримки моделей у виробничому середовищі, включно з виявленням дрейфу даних та деградації продуктивності моделей;
3) у [6, 8] розглянуто виклики інтеграції розробки програмного забезпечення із конвеєром машинного навчання;
4) у [5, 6] розглянуто виклики забезпечення безпеки та конфіденційності даних під час розгортання моделей машинного навчання;
5) у [5, 15] вказано, що якість, доступність, підготовка, маркування та інтеграція даних з різних джерел є значним викликом, що потребує багато часу та ресурсів; також виділено проблему інтерпретації й пояснення результатів роботи моделей для кінцевих користувачів та бізнес-стейкхолдерів.

Відмінні виклики:

1) у [6] підкреслена необхідність автоматизації всіх етапів конвеєра MLOps та інтеграції з наявними системами і процесами розробки програмного забезпечення;
2) у [8] зазначено проблему вибору та управління інфраструктурою для розгортання моделей, зокрема вибір між хмарними та локальними середовищами.
3) у [5] зазначено такі проблеми: а) розрив між інженерією програмного забезпечення та навичками машинного навчання – науковці з даних часто не розуміють вимоги виробничих середовищ, а розробники програмного забезпечення не мають достатніх навичок машинного навчання; б) неефективний розподіл, паралелізація та оркестрація даних і завдань машинного навчання; в) різноманіття обчислювальної інфраструктури.

Розглянуті огляди демонструють, що розгортання моделей машинного навчання у виробничих середовищах пов’язане з низкою викликів, як-от управління життєвим циклом моделей, забезпечення масштабованості та продуктивності, моніторинг і підтримка моделей у реальних умовах використання. Вирішення цих викликів вимагає комплексного підходу, що включає автоматизацію процесів MLOps, вибір відповідної інфраструктури, забезпечення безпеки та конфіденційності даних, а також ефективну комунікацію з бізнес-стейкхолдерами.

Відкриті питання, виклики та особливості MLOps

В оглядах [5, 6, 8, 15] визначено найбільш актуальні та перспективні напрями досліджень та розробок у цій галузі. У [6] та [15] безпосередньо не обговорюються відкриті проблеми, виклики та особливості MLOps, однак їх можна виокремити на основі аналізу інструментів MLOps і їх можливостей у [6] та опису компонентів та функцій платформ MLOps [15].

Спільні відкриті питання та виклики MLOps:

1) у [5, 6, 8, 15] вказано на потребу в розробці методів та інструментів для забезпечення інтерпретовності, відтворюваності й відповідального використання моделей машинного навчання в контексті MLOps;
2) у [5, 6, 15] підкреслена важливість розробки підходів до управління даними в MLOps, включно із забезпеченням якості, конфіденційності та безпеки даних;
3) у [6, 8] відзначена необхідність розробки та впровадження стандартів і кращих практик MLOps для забезпечення узгодженості та сумісності між різними інструментами та платформами;
4) у [5, 8] підкреслена важливість людського фактора в MLOps, включно із необхідністю забезпечення ефективної комунікації та співпраці між різними ролями й командами та підготовку кваліфікованих кадрів із кросфункціональними навичками програмування, обробки даних та операційної діяльності.

Особливості MLOps:

1) у [6] зазначено, що MLOps має враховувати специфіку процесу розробки моделей машинного навчання, який відрізняється від традиційної розробки програмного забезпечення;
2) у [15] розглянуто MLOps у контексті загальної цифрової стратегії організації та підкреслено необхідність узгодження практик MLOps із бізнес-цілями й потребами.

Отже, розглянуті огляди визначають низку відкритих питань та викликів в MLOps, як-от необхідність розробки стандартів і кращих практик, забезпечення інтерпретовності та відповідального використання моделей, а також ефективне управління даними. Особливості MLOps, як-от відмінність від традиційної розробки програмного забезпечення, важливість людського фактора та необхідність інтеграції знань з різних галузей, вимагають врахування під час впровадження практик MLOps в організаціях.

Можливості, майбутні тенденції та сфери застосування MLOps

У [6] перспективні напрями розвитку та потенційні області, в яких практики MLOps можуть принести значну користь, безпосередньо не обговорюються, однак їх можна визначити, базуючись на представлених інструментах MLOps та їх можливостях.

Можливості та майбутні тенденції MLOps:

1) у [5, 6, 8] відзначено потенціал розвитку стандартизованих платформ та інструментів MLOps, які дають змогу спростити та прискорити впровадження моделей машинного навчання в виробництво;
2) у [5, 8] відзначено перспективи інтеграції MLOps з іншими підходами, як-от DataOps, ModelOps та DevSecOps, для забезпечення комплексного управління життєвим циклом моделей машинного навчання;

3) у [8] вказано на: а) значні можливості для подальших академічних досліджень та розробок через те, що MLOps все ще знаходиться на початковій стадії; б) очікування зростання попиту на інструменти та платформи MLOps із поширенням застосування рішень штучного інтелекту; в) появу нових ролей та компетенцій, пов'язаних з MLOps;
4) у [5] вказано на: а) можливості застосування практик MLOps у контексті розподіленого та федеративного навчання моделей, що дасть змогу ефективно використовувати децентралізовані дані; б) залучення бізнес-підрозділів та навчання керівництва принципів MLOps; в) використання апаратних платформ FPGA та IoT для покращення продуктивності та конфіденційності.

Поточні та майбутні сфери застосування MLOps:

1) у [5, 6, 8] відзначено, що MLOps уже активно застосовується у фінансах, охороні здоров'я, торгівлі, маркетингу та виробництві, де моделі машинного навчання використовуються для вирішення реальних бізнес-задач;
2) у [5] вказано на потенціал застосування MLOps у сфері IoT та периферійних обчислень, де моделі машинного навчання можуть бути розгорнуті на пристроях з обмеженими ресурсами, а також на застосуванні MLOps для технологій 5G й 6G, та в навчальній і науковій діяльностях;
3) у [8] відзначено перспективи використання MLOps у транспорті та логістиці.

Розглянуті огляди окреслюють низку можливостей та тенденцій розвитку MLOps, як-от створення стандартизованих платформ, застосування в контексті розподіленого навчання та інтеграція з іншими підходами управління життєвим циклом даних та моделей. Поточні та майбутні сфери застосування MLOps включають широкий спектр галузей, від фінансів та охорони здоров'я до IoT та обробки природної мови, що свідчить про значний потенціал впливу цього підходу.

Аналіз практик MLOps

Співвідношення принципів, процесів і практик MLOps

MLOps базується на наборі принципів [2] та процесів, які забезпечують ефективні практик MLOps – розроблення, розгортання та підтримку моделей машинного навчання.

Принципи MLOps визначають фундаментальні основи проєктування конвеєрів машинного навчання. Принципи є такими:

- *автоматизація* – максимальна автоматизація всіх етапів життєвого циклу моделей машинного навчання для зменшення ручних втручань та покращення ефективності;
- *відтворюваність* – забезпечення можливості відтворення результатів експериментів та процесів розгортання моделей;
- *співпраця* – налагодження ефективної співпраці та комунікації між різними командами, залученими до розробки та впровадження моделей;
- *безперервне навчання та покращення* – регулярне оновлення моделей на основі нових даних та зворотного зв'язку, постійне вдосконалення процесів MLOps;
- *керованість даними* – забезпечення якості, безпеки та конфіденційності даних протягом усього життєвого циклу моделей.

Процеси MLOps визначають порядок дій із проєктування та реалізації конвеєрів машинного навчання. Вони є такими:

- *визначення бізнес-цілей та вимог* – узгодження цілей розробки моделей машинного навчання з бізнес-стратегією організації;
- *збір та підготовка даних* – збір, очищення, трансформація та збагачення даних для навчання моделей;
- *розробка та навчання моделей* – вибір алгоритмів, розробка архітектури моделей, навчання та валідація моделей;
- *оцінювання та тестування моделей* – оцінювання продуктивності моделей на тестових даних, проведення тестів на надійність, безпеку та відповідність вимогам;
- *розгортання моделей* – пакування моделей з необхідними залежностями, розгортання в цільових середовищах;
- *моніторинг та обслуговування моделей* – відстеження продуктивності моделей, виявлення й вирішення проблем, оновлення моделей за потреби;
- *управління життєвим циклом моделей* – координація всіх етапів розробки, розгортання та підтримки моделей, забезпечення відповідності регуляторним вимогам.

Практики MLOps визначають найбільш ефективні методи та технології реалізації конвеєрів машинного навчання. Основні практики MLOps є такими:

- *безперервна інтеграція та доставка* (CI / CD) – автоматизація процесів збирання, тестування та розгортання моделей машинного навчання;
- *версіонування моделей та даних* – відстеження змін у моделях та датасетах, забезпечення відтворюваності результатів;
- *автоматизація конвеєрів машинного навчання* – створення автоматизованих робочих процесів для збору, обробки даних, навчання та оцінки моделей;
- *моніторинг продуктивності моделей* – відстеження метрик якості моделей у виробничому середовищі, виявлення деградації продуктивності;
- *управління експериментами* – організація, відстеження та порівняння різних експериментів з моделями та гіперпараметрами;
- *розгортання моделей* – упакування моделей з необхідними залежностями, розгортання в різних середовищах (хмара, периферія тощо);
- *управління життєвим циклом моделей* – координація процесів розробки, тестування, розгортання та моніторингу моделей для найкращого врахування вимог.

Додаткові практики MLOps є такими:

- *безпека та конфіденційність даних* – забезпечення захисту даних, що використовуються для навчання моделей, відповідність регуляторним вимогам;
- *пояснюваність та інтерпретовність моделей* – використання методів та інструментів для розуміння і пояснення поведінки моделей, особливо в регульованих галузях;
- *управління якістю даних* – моніторинг та забезпечення якості даних, які використовуються для навчання й оцінювання моделей, виявлення та обробки аномалій;
- *управління конфігурацією* – версіонування та управління конфігураціями середовищ, у яких розгортаються моделі, забезпечення узгодженості між різними середовищами;

- *стратегії розгортання моделей* – вибір та реалізація стратегій розгортання;
- *автоматизація інфраструктури* – використання Infrastructure as Code для автоматизації управління інфраструктурою навчання та розгортання моделей;
- *співпраця та комунікація* – налагодження ефективної співпраці між командами науки про дані, розробки, операційної діяльності та бізнес-підрозділами;
- *управління ризиками та комплаєнс* – ідентифікація ризиків використання моделей машинного навчання, запобігання їм та задоволення регуляторних вимог.

Рис. 2 ілюструє зв'язки між ключовими принципами MLOps (блакитні прямокутники), основними процесами (зелені прямокутники) та поширеними практиками (помаранчеві прямокутники). Стрілки показують, як принципи впливають на процеси, а процеси реалізуються через конкретні практики. Наприклад, принцип автоматизації впливає на всі процеси MLOps, від визначення цілей до управління життєвим циклом моделей. Процес розробки моделей пов'язаний із практиками версіонування, автоматизації конвеєрів, управління експериментами та інтерпретовності моделей.

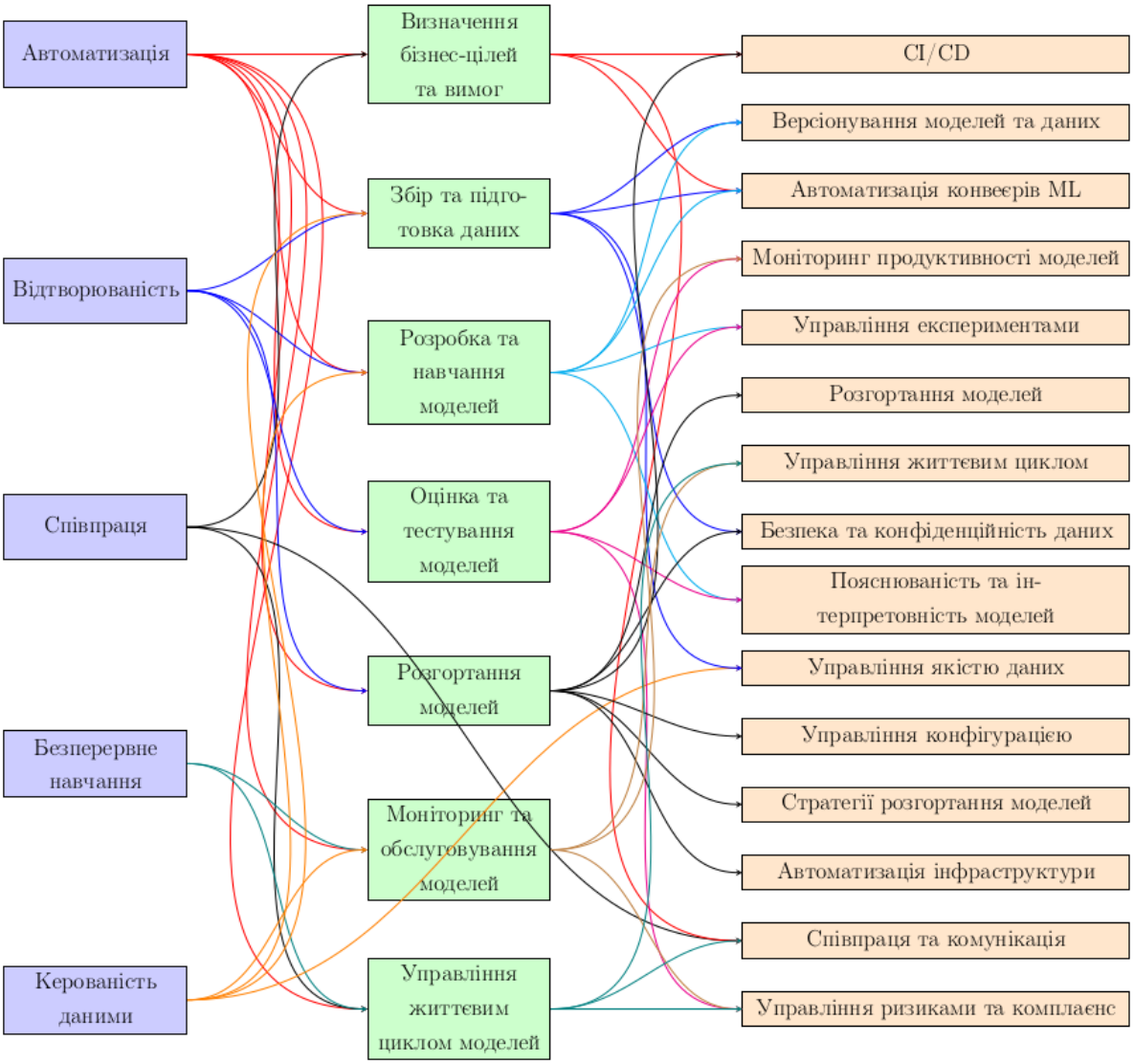

Рисунок 2. Взаємозв'язок принципів, процесів та практик MLOps

CI / CD

CI / CD – це ключовий елемент DevOps для автоматичного тестування та розгортання коду, даних і моделей у виробничому середовищі (рис. 3). У MLOps CI / CD розширюється для автоматизації процесу розробки та розгортання моделей машинного навчання, включно з етапами побудови, тестування, доставки та розгортання [2].

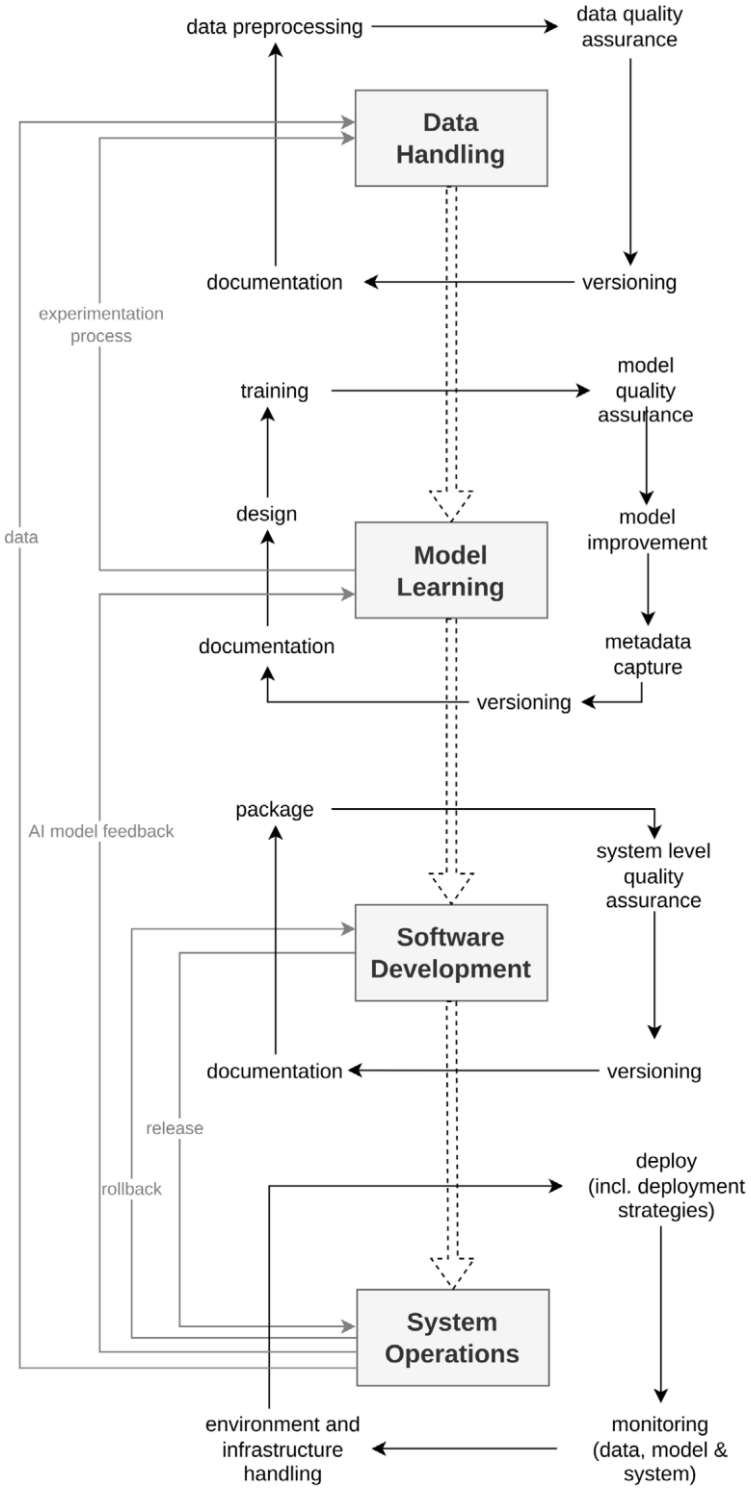

Рисунок 3. Безперервний конвеєр життєвого циклу для програм штучного інтелекту [7]

Процес CI / CD у MLOps включає в себе етапи побудови (build), тестування (test), доставки (delivery) та розгортання (deploy) [2]. Однак, на відміну від традиційного CI / CD, у MLOps також можуть бути додаткові етапи, як-от донавчання моделі. CI / CD у MLOps є частиною архітектури систем MLOps та забезпечує розробникам швидкий зворотний зв'язок у випадку успішності чи невдачі певних етапів, що підвищує загальну продуктивність [2]. Типовими тригерами запуску процесу CI / CD у MLOps на GitHub є події git push та pull_request [12]. Також можуть використовуватися події issue_comment, release та schedule. У статті [7] також досліджено потенційні основні тригери, як-от системи зворотного зв'язку та оповіщення, служба оркестрування за розкладом, традиційні оновлення репозиторію та ручні тригери. Ці тригери запускають виконання конвеєра, який складається з чотирьох етапів: обробка даних; навчання моделі; розробка програмного забезпечення; введення системи в експлуатацію. Етап обробки даних складається з повторюваного наскрізного життєвого циклу завдань, пов'язаних з даними, як-от попередня обробка, забезпечення якості, версіонування та документування. На етапі навчання моделі здійснюється проєктування моделі, її навчання, забезпечення якості, збір метаданих для вдосконалення моделі, керування версіями та документування. Після того, як конвеєр завершує навчання моделі, етап розробки програмного забезпечення готує модель до розгортання за допомогою пакування, забезпечення якості на програмному рівні та версіонування системи. На завершальному етапі введення системи в експлуатацію модель розгортається в конкретному середовищі за допомогою різних стратегій, а також здійснюється моніторинг системи [7].

Особливістю процесу CI / CD у MLOps є необхідність версіонування не лише коду, а й даних та моделей. Це дає змогу забезпечити відтворюваність та можливість відкату до попередніх версій [2]. Серед популярних інструментів для реалізації CI / CD у MLOps в [2] виділяють GitHub Actions та інструменти від хмарних провайдерів AWS CodePipeline, Azure DevOps Pipelines тощо.

На рис. 4 наведено наскрізну архітектуру та робочий процес MLOps із функціональними компонентами та ролями, задіяними на кожному етапі. Розглянемо їх детальніше.

MLOps Project Initiation. На цьому етапі бізнес-стейкхолдер (BS) аналізує проблему та визначає ціль. Науковець з даних (DS) формулює проблему машинного навчання на основі бізнес-цілі. Також визначаються необхідні дані та здійснюється їх первинний аналіз інженером з даних (DE) та науковцем з даних (DS).

Data Engineering Zone. Ця зона включає два підетапи:

Requirements for feature engineering pipeline, на якому визначаються правила для трансформації, очищення та розрахунку нових ознак;

Feature Engineering Pipeline, на якому реалізується конвеєр обробки даних, який отримує дані із різних джерел, застосовує трансформації і завантажує у сховище ознак (Feature store system).

Experimentation. На цьому етапі науковець з даних (DS) здійснює аналіз, підготовку та валідацію даних, а також навчання й валідацію моделі. Найкраща модель зберігається у Model Registry.

ML Production Zone. Ця зона включає автоматизований конвеєр (D. Automated ML Workflow Pipeline), який забезпечує підготовку даних, навчання, валідацію та реєстрацію

моделі у режимі production. Компонент Model Serving розгортає модель, а Monitoring Component забезпечує її постійний моніторинг. У разі виявлення проблем інформація передається через Feedback Loop для ініціювання повторного навчання моделі.

Окрім функціональних компонентів, на рисунку також показані різні ролі та зони їх відповідальності: Business Stakeholder (BS), Data Scientist (DS), Data Engineer (DE), DevOps Engineer (DO), ML Engineer (ML), Software Engineer (SE) та IT Solution Architect (SA).

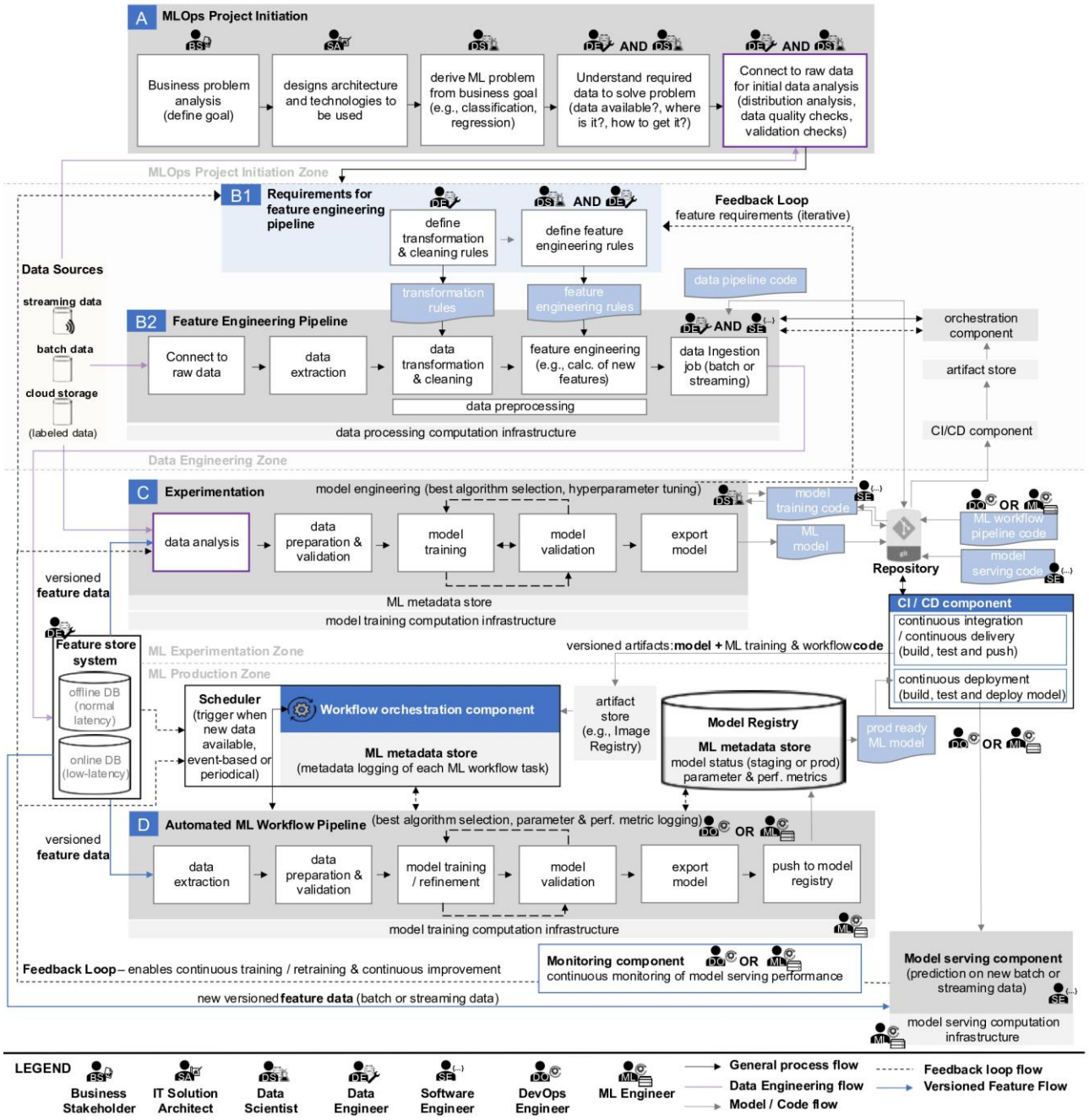

Рисунок 4. Наскрізна архітектура та робочий процес MLOps із функціональними компонентами та ролями [2]

Версіонування моделей та даних

Версіонування є однією з ключових практик MLOps, яка забезпечує відтворюваність (reproducibility) та можливість відстеження (traceability) моделей машинного навчання [2]. Версіонування охоплює дані, код та самі моделі [7]. Версіонування моделі не лише фіксує артефакти моделі версій, але й залежності моделі для відстеження або відтворення різних версій моделі, які швидко змінюються з часом [7]. Метою версіонування даних є гарантування відтворюваності моделей та відповідності регуляторним вимогам. Версіонування даних може реалізовуватись або через збереження знімків даних (data snapshots), або через посилання на оригінальний набір даних. Оскільки традиційні системи керування версіями не можуть впоратися з великими обсягами даних, використовують спеціалізовані інструменти, як-от Data Version Control [7].

У робочому процесі MLOps версіонування моделей відбувається на етапі навчання. Його метою є збереження різних версій моделей разом з метаданими для можливості відкату до попередніх версій та відтворення результатів. Умовами використання версіонування моделей є постійна еволюція моделей з часом та необхідність відстеження залежностей між моделями, даними та кодом [7].

Особливістю версіонування моделей, на відміну від традиційного версіонування коду, є необхідність відстеження більшої кількості артефактів та метаданих, а також потреба у більших обсягах пам’яті через постійний розвиток моделей [7]. Окрім самих даних, версіонуванню підлягають залежності, кроки обробки даних та видобуті ознаки (features) [7]. Для останніх часто використовують спеціальні сховища ознак (feature stores). Залежності фіксують зв’язок із пов’язаними елементами, як-от набір даних, вихідний код і конфігураційні файли. До того ж версії моделі зберігають пов’язані з ними файли журналів і результати оцінювання моделі. Це дає змогу перевірити, чи версії моделі постійно вдосконалюються впродовж безперервного життєвого циклу. Оскільки керування версіями моделей штучного інтелекту є складнішим і вимагає більшого обсягу пам’яті через безперервний розвиток, стандартні системи керування версіями, як-от Git, не можуть бути використані як репозиторії моделей. Потенційні альтернативи – це контейнерні реєстри (MLFlow, H2O, DataRobot), де зберігаються версії зображень, або репозиторії моделей, які зберігають версії моделей, включно з кодом, метаданими, результатами тестів і залежностями [7].

Автоматизація конвеєрів машинного навчання

Автоматизація конвеєрів машинного навчання є ключовою практикою MLOps, яка дає змогу спростити та прискорити розробку, тестування й розгортання моделей у робочому середовищі. Ця практика охоплює автоматизацію різних етапів конвеєра машинного навчання, включно зі збором даних, препроцесінгом, розробкою моделі, навчанням, тестуванням, валідацією та розгортанням [7].

Алгоритм роботи автоматизованого конвеєра машинного навчання (рис. 5) складається з 10 етапів. Цикл зворотного зв’язку забезпечує безперервне покращення моделі шляхом повернення до етапу навчання. За потреби моніторинг може ініціювати тригер донавчання моделі, який запускає процес спочатку. Цей автоматизований конвеєр надає можливість

ефективно керувати життєвим циклом моделі машинного навчання, від підготовки даних до розгортання та моніторингу, забезпечуючи безперервне вдосконалення моделі.

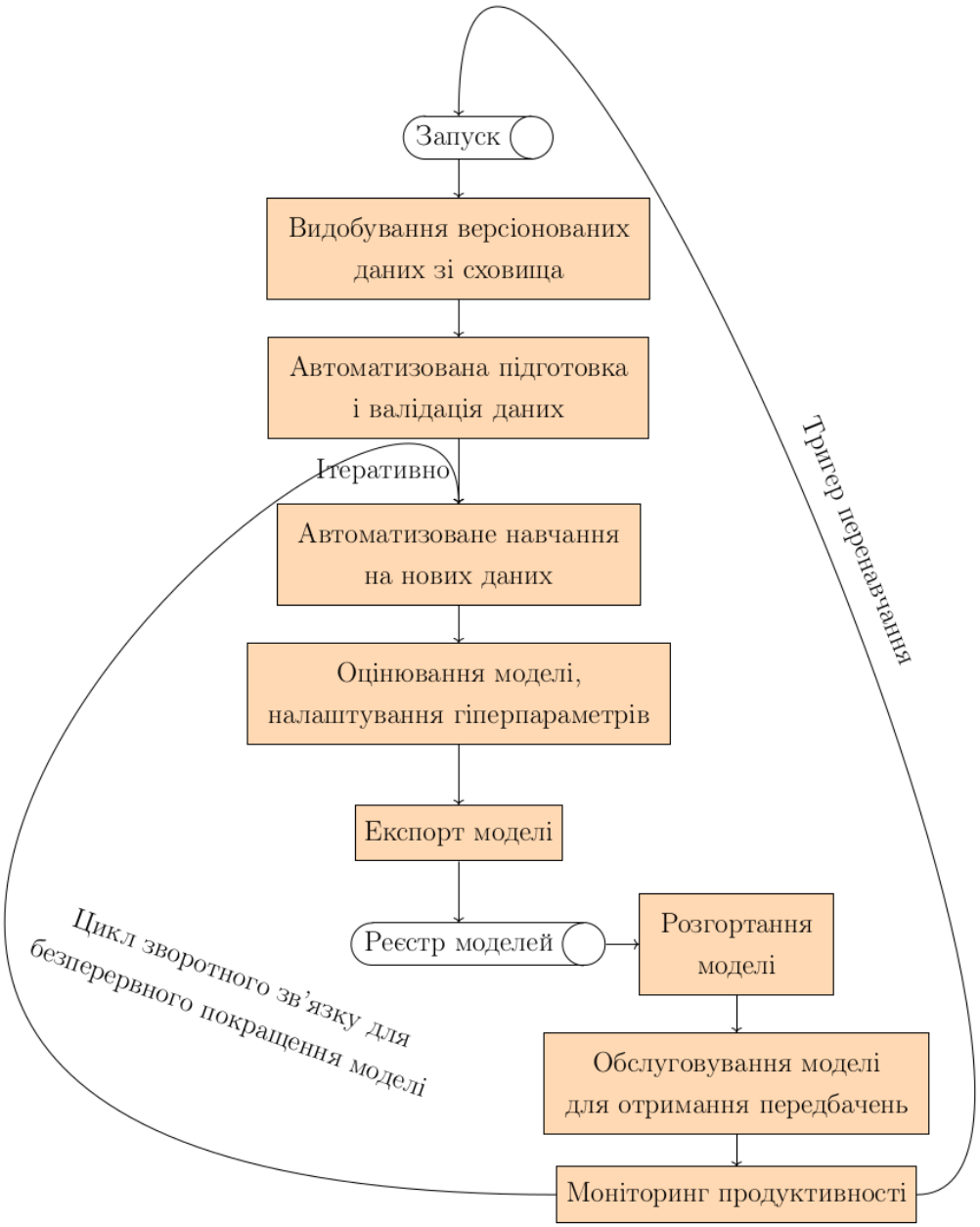

Рисунок 5. Алгоритм роботи автоматизованого конвеєра машинного навчання (за [2])

Особливістю використання автоматизації конвеєрів машинного навчання є врахування гетерогенності моделей, фреймворків та середовищ виконання. Тому кожен крок конвеєра машинного навчання має бути максимально ізольованим та повинен мати чіткі інтерфейси, наприклад, через використання контейнеризації [22]. Також критично важливим є забезпечення можливості відтворення результатів та відслідковування артефактів [23].

Способи автоматизації конвеєрів машинного навчання включають використання систем керування конвеєрами, як-от Kubeflow, TensorFlow Extended [7] або Apache Airflow [6], а

також розробку власних скриптів автоматизації з використанням Jenkins [24] або GitLab CI / CD [2]. Під час цього конвеєр розбивається на окремі кроки, кожен з яких реалізується як код або конфігурація, а потім ці кроки оркеструються та виконуються автоматично [23].

Моніторинг продуктивності моделей

Моніторинг продуктивності моделей машинного навчання є важливою практикою MLOps, яка дає змогу відстежувати роботу моделей у виробничому середовищі, виявляти проблеми та вживати заходів для підтримки їх якості. Ця практика охоплює збір метрик щодо роботи моделі, моніторинг цих метрик у реальному часі та оповіщення у разі виявлення відхилень від норми [25]. Моніторинг продуктивності відбувається на етапі експлуатації моделі, після її розгортання у виробничому середовищі. Він є частиною неперервного циклу MLOps і виконується паралельно з іншими практиками, як-от розробка, тестування та розгортання [25].

Умовами використання моніторингу є наявність розгорнутої моделі машинного навчання, яка обробляє реальні дані та генерує передбачення. До того ж повинна бути налаштована інфраструктура для збору і зберігання даних моніторингу, наприклад, база даних та інструменти візуалізації [25]. Процес моніторингу включає кілька кроків (рис. 6). Спочатку визначаються ключові метрики продуктивності моделі – точність, помилка, латентність тощо. Потім налаштовуються інструменти для збору цих метрик із системи, де розгорнута модель. Далі дані агрегуються та візуалізуються на інформаційних панелях для зручного аналізу. Нарешті, налаштовуються оповіщення, які спрацьовують за виходу метрик за допустимі межі [25].

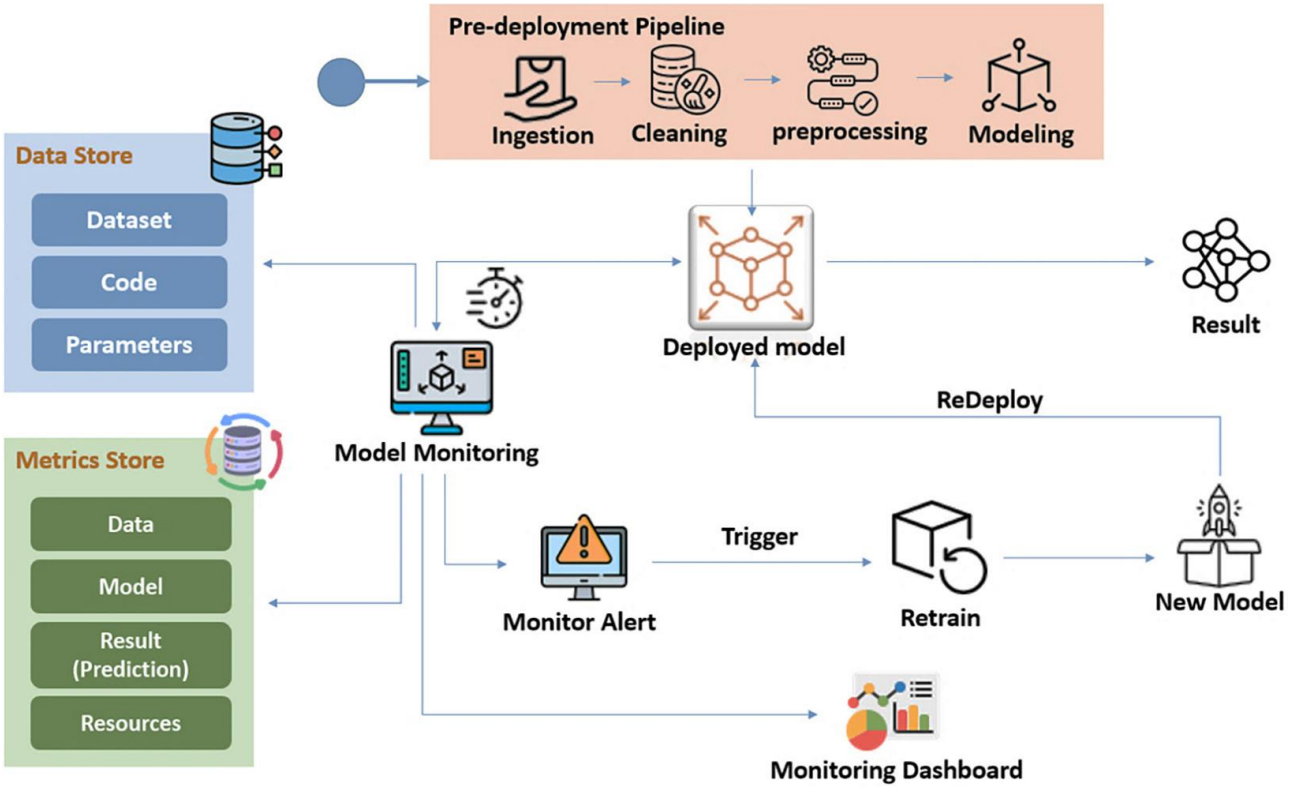

Рисунок 6. Моніторинг у MLOps [25]

Рис. 7 зображує запропоновану в [25] аналогію між моніторингом системи машинного навчання та айсбергом. Вона підкреслює, що “здоров’я системи” машинного навчання залежить не тільки від видимих елементів, як-от надані сервіси (Service), але й від прихованих особливостей, як-от дані (Data) та сама модель (Model). Верхній рівень айсберга представляє “здоров’я системи” (Ecosystem Health), що включає прогнозний дрейф (Prediction Drift) та бізнес-показники ефективності (Business KPI). Метою моніторингу є оцінювання продуктивності, надійності та стабільності цього верхнього рівня. Рівень сервісів включає метрики латентності (Latency), вартості (Cost) та загальної продуктивності системи (System Performance). Рівень даних містить характеристики якості даних (Data Quality), викиди (Outlier Value) та дрейф даних (Data Drift). Найнижчий рівень айсберга – це модель, яка характеризується точністю (Model Accuracy), дрейфом концепції (Concept Drift) та зміщенням моделі (Model Bias).

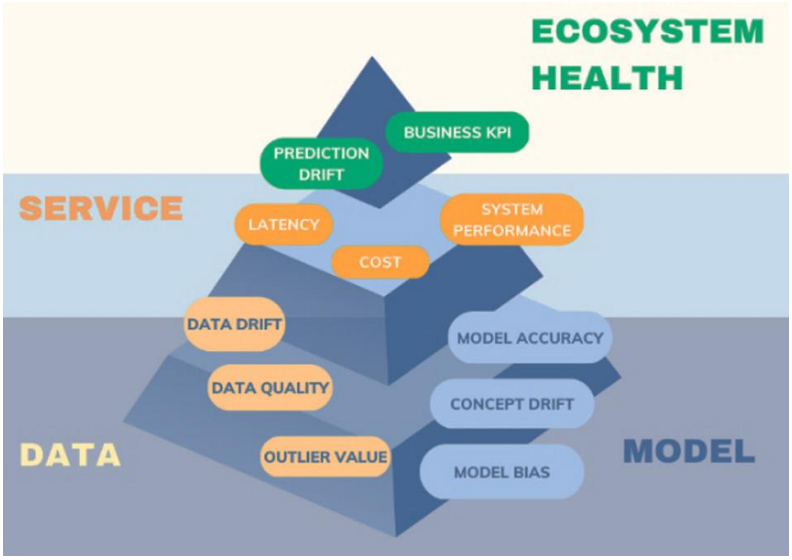

Рисунок 7. Приховані та видимі характеристики, пов’язані з моніторингом [25]

Особливістю моніторингу в контексті MLOps є необхідність відстежувати не лише традиційні метрики продуктивності програмного забезпечення, як то завантаженість процесора та пам’яті, але й специфічні для машинного навчання метрики, наприклад, точність моделювання на нових даних (рис. 8). До того ж MLOps передбачає автоматизацію процесу моніторингу та інтеграцію його в загальний конвеєр розробки й розгортання моделей [25]. Існує низка інструментів для налаштування моніторингу продуктивності моделей машинного навчання. Вони включають платформи з відкритим кодом, як-от Prometheus, Grafana, ELK stack, а також комерційні рішення від хмарних провайдерів, наприклад, AWS CloudWatch, Google Stackdriver, Azure Monitor. Ці інструменти дають змогу зручно збирати, зберігати, візуалізувати метрики та налаштовувати оповіщення [25].

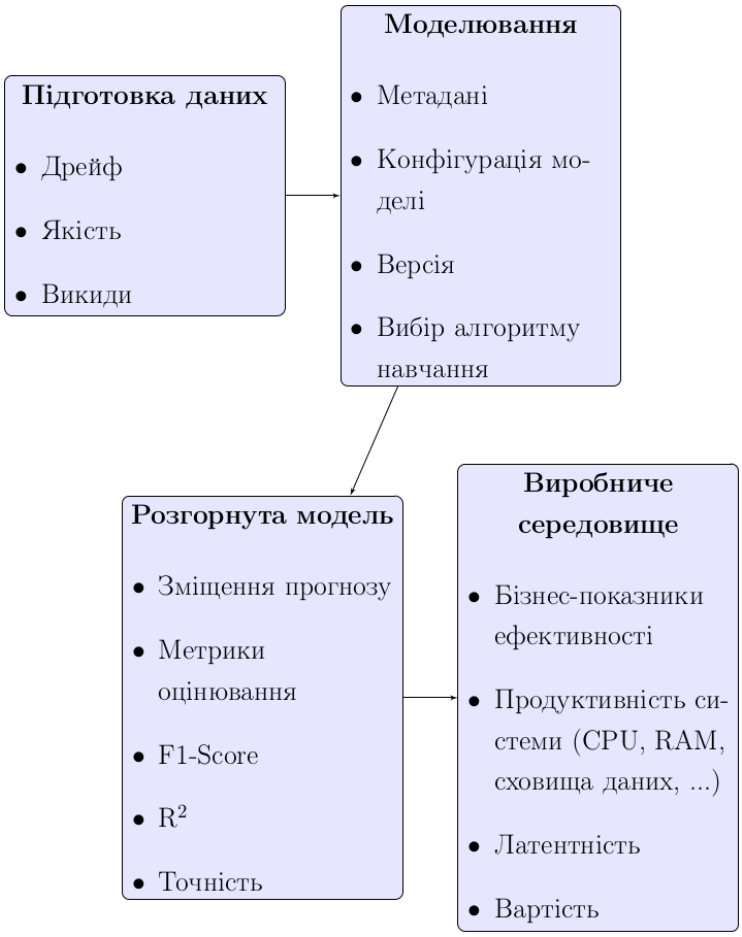

Рисунок 8. Елементи для моніторингу [25]

Управління експериментами

Управління експериментами (experiment tracking) полягає в систематичному збереженні метаданих експериментів машинного навчання [26]. Метадані експерименту можуть включати довільні скрипти для запуску експерименту, файли конфігурації середовища, інформацію про дані для навчання та оцінювання, конфігурації параметрів моделі та навчання, метрики оцінювання, вагові коефіцієнти моделі, візуалізації ефективності (наприклад, матриця сплутувань або ROC-крива) тощо [27]. Управління експериментами (також відоме як реєстрація експериментів) є частиною MLOps, яка орієнтована на підтримку ітеративної розробки моделі – частини життєвого циклу проєкту машинного навчання, де, зокрема, виконується добір гіперпараметрів для досягнення необхідного рівня продуктивності моделі. Управління експериментами тісно переплітається з іншими аспектами MLOps, як-от версіонування даних і моделей. Основною умовою застосування управління експериментами є ітеративний характер процесу розробки та навчання моделі, коли проводиться багато експериментів з різними наборами гіперпараметрів, архітектур моделей та навчальних даних (рис. 9). Це характерно для дослідницьких та прикладних проєктів із машинного навчання.

Найбільш популярними інструментами для управління експериментами є MLflow, Neptune.ai, Weights & Biases (wandb), Guild.ai, Comet.ml та TensorBoard [2]. Вони дають

змогу зберігати у репозиторії інформацію про кожен експеримент – використані дані та їх версії, параметри моделі та її архітектура, метрики ефективності, артефакти моделі тощо.

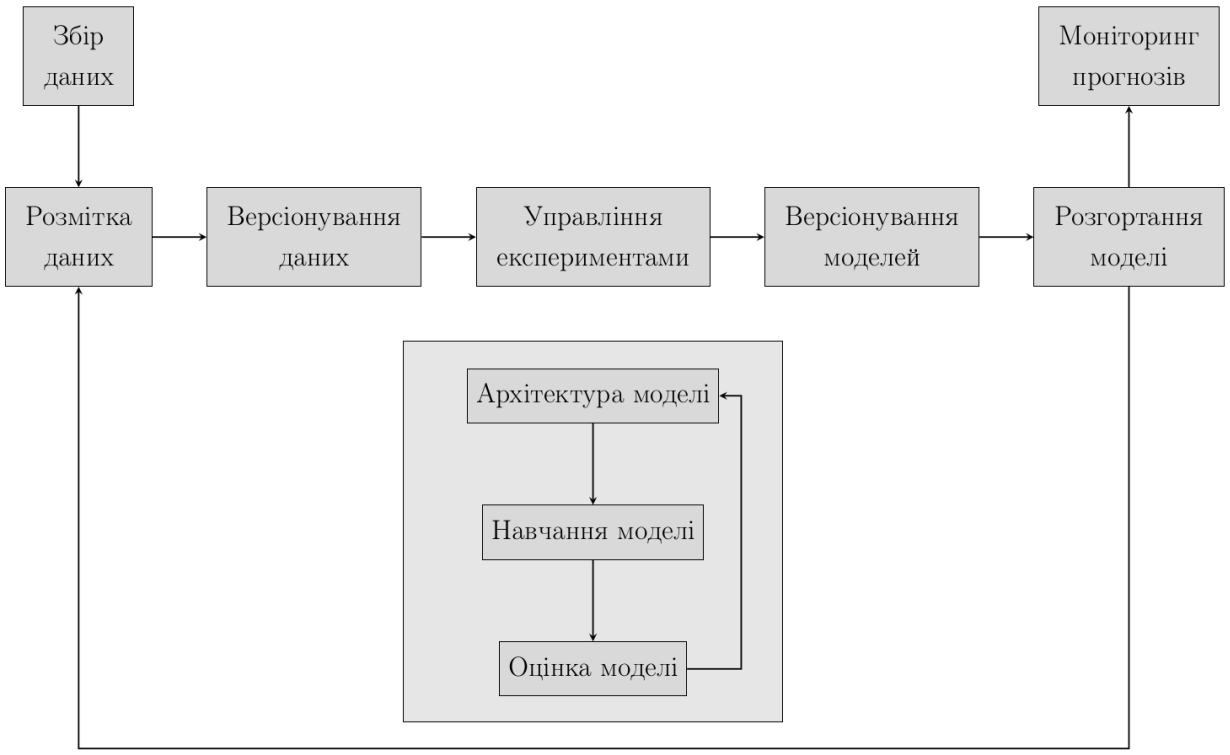

Рисунок 9. Управління експериментами у життєвому циклі MLOps (за [27])

Розгортання моделей

Розгортання моделей (model deployment) є важливою практикою MLOps, що відбувається на етапі операціоналізації (operationalization) у робочому процесі розробки та впровадження моделей машинного навчання. Ця практика полягає у безпосередньому розміщенні навченої та протестованої моделі машинного навчання у виробниче середовище, де вона може використовуватися для отримання прогнозів на реальних даних [11].

Розгортання може відбуватися кількома різними способами. Моделі, упаковані в контейнери, просто запускаються безпосередньо як окремі сервіси. Однак моделі можуть розгортатися в цільовому середовищі, яке відрізняється від того, де вони були упаковані. В цьому випадку перенесення моделі здійснюється за допомогою push- або pull-шаблонів [11]:

- під час розгортання за pull-шаблоном цільове середовище (хост-додаток, що працює, наприклад, на сервері або периферійному пристрої) періодично запитує оновлення моделі та завантажує їх, коли вони доступні;
- під час розгортання за push-шаблоном цільове середовище сповіщається про доступність нової моделі головним сервером через службу обміну повідомленнями.

Розгортання моделей машинного навчання часто відбувається шляхом їх пакування у контейнери, наприклад, за допомогою Docker. Це дає змогу стандартизувати процес розгортання і гарантувати, що модель буде працювати у тому ж середовищі, що і під час розробки і тестування [11].

Існують такі способи розгортання моделей машинного навчання, які обумовлено вимогами й архітектурою системи [28]:

- модель може бути інтегрована безпосередньо у код застосунку;
- модель може бути розгорнута як окремий сервіс (мікросервіс) із REST API;
- модель може бути завантажена у спеціалізоване середовище для розгортання та масштабування моделей (наприклад, AWS SageMaker).

Розгортання моделей машинного навчання пов'язане з низкою проблем, зокрема з потребами забезпечення малої затримки та високої пропускної здатності сервісів прогнозування [11]. Для їх вирішення використовуються різні методи, як-от адаптивні черги з тайм-аутом для пакетного прогнозування, кешування, динамічне переключення між моделями різної точності тощо. З інструментів для розгортання моделей машинного навчання використовують системи оркестрації контейнерів (Kubernetes), сервіси хмарних провайдерів (AWS SageMaker, Azure ML), наскрізні платформи MLOps (MLflow, Kubeflow) [3, 28].

Розгортання моделей машинного навчання є критично важливою практикою MLOps, що дає змогу переводити розроблені моделі у робочий стан і використовувати їх для прогнозування. Воно відбувається на етапі операціоналізації та вимагає врахування низки факторів – від способу пакування моделі до забезпечення необхідної продуктивності сервісу прогнозування. Розгортання спирається на сучасні інструменти контейнеризації, оркестрації та автоматизації інфраструктури.

На основі узагальнення матеріалів з [11, 28] на рис. 10 представимо загальну схему розгортання моделей машинного навчання. На схемі використовуються такі позначення:

Модель ML – навчена та протестована модель машинного навчання;

Упаковка – пакування моделі у відповідний формат (наприклад, у контейнер Docker);

Реєстр моделей – реєстр упакованих моделей;

Розгортання – розгортання упакованої модель у цільовому середовищі (хмара, периферійні пристрої);

Обслуговування – обслуговування моделлю запитів та генерування передбачень;

Моніторинг – відстежування продуктивності моделі та середовища, і за потреби – ініціація повторного навчання моделі.

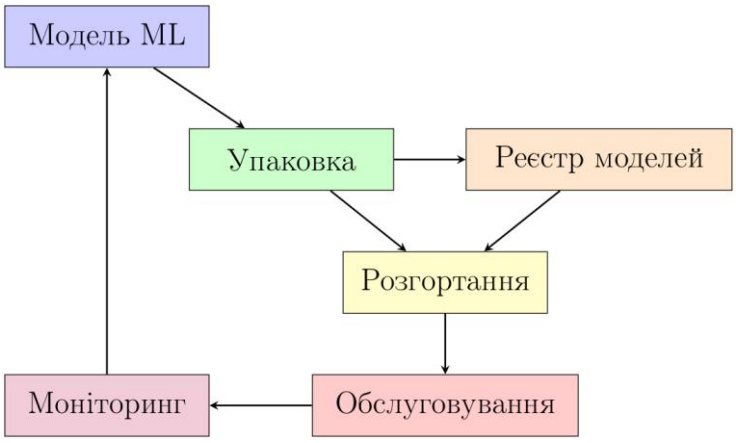

Рисунок 10. Схема розгортання моделей машинного навчання

Управління життєвим циклом

Згідно з [7], безперервне управління життєвим циклом (end-to-end lifecycle management) – це “безперервний конвеєр / поток управління, який описує (автоматичне) виконання певних задач задля забезпечення управління життєвим циклом штучного інтелекту, … який розпочинається зі збору даних та завершується розгортанням та моніторингом моделі штучного інтелекту”. Метою управління життєвим циклом є уніфікація та стандартизація процесів, що сприяє підвищенню продуктивності розробки машинного навчання моделей та їх надійності у промисловому середовищі.

Ключові особливості управління життєвим циклом у MLOps:

- охоплює усі етапи – збір та підготовку даних, розробку моделі, її навчання, валідацію та розгортання [25];
- застосовується як на ранніх стадіях розробки моделей, так і для їх неперервної підтримки після розгортання у виробничому середовищі [5];
- передбачає версіонування даних, коду та самих моделей для відстеження змін і забезпечення відтворюваності;
- передбачає моніторинг продуктивності моделей;
- автоматизує процеси за допомогою конвеєрів [7].

На рис. 11 показано процес розробки проєкту машинного навчання (конвеєр), який складається як із лінійних, так і з ітеративних кроків. Конвеєр починається з видобутку даних, їх перевірки та підготовки, а потім здійснюється навчання моделі, оцінювання та перевірка (управління експериментами). Після цього модель розгортається у виробничому середовищі [25].

Важливим аспектом життєвого циклу є версіонування моделей та даних, яке відбувається як у конвеєрі управління експериментами, так і у виробничому конвеєрі. Останній включає всі етапи створення моделі, а не лише кінцевий результат конвеєра управління експериментами. Ітераційний характер MLOps надає можливість отримання та підтримання найкращої моделі, а поєднання моніторингу на основі ключових показників ефективності та функції попередження забезпечує проактивне втручання, гарантуючи якість і надійність розгорнутих моделей протягом усього життєвого циклу [25].

Основні інструменти, що використовуються для управління життєвим циклом у MLOps є такими [25]:

- платформи для організації workloads та pipelines в машинному навчанні, як-от MLflow та Kubeflow;
- системи версіонування даних, наприклад, Data Version Control;
- інструменти для моніторингу продуктивності моделей у виробничому середовищі, як-от Neptune.ai.

Комплексний підхід до управління життєвим циклом у MLOps реалізовано в Ease.ML – в системі управління життєвим циклом для спрощення всього процесу розробки [29]. Основна мета Ease.ML полягає у наданні систематичних рекомендацій та автоматизації на всіх етапах життєвого циклу машинного навчання, мінімізуючи зусилля користувачів.

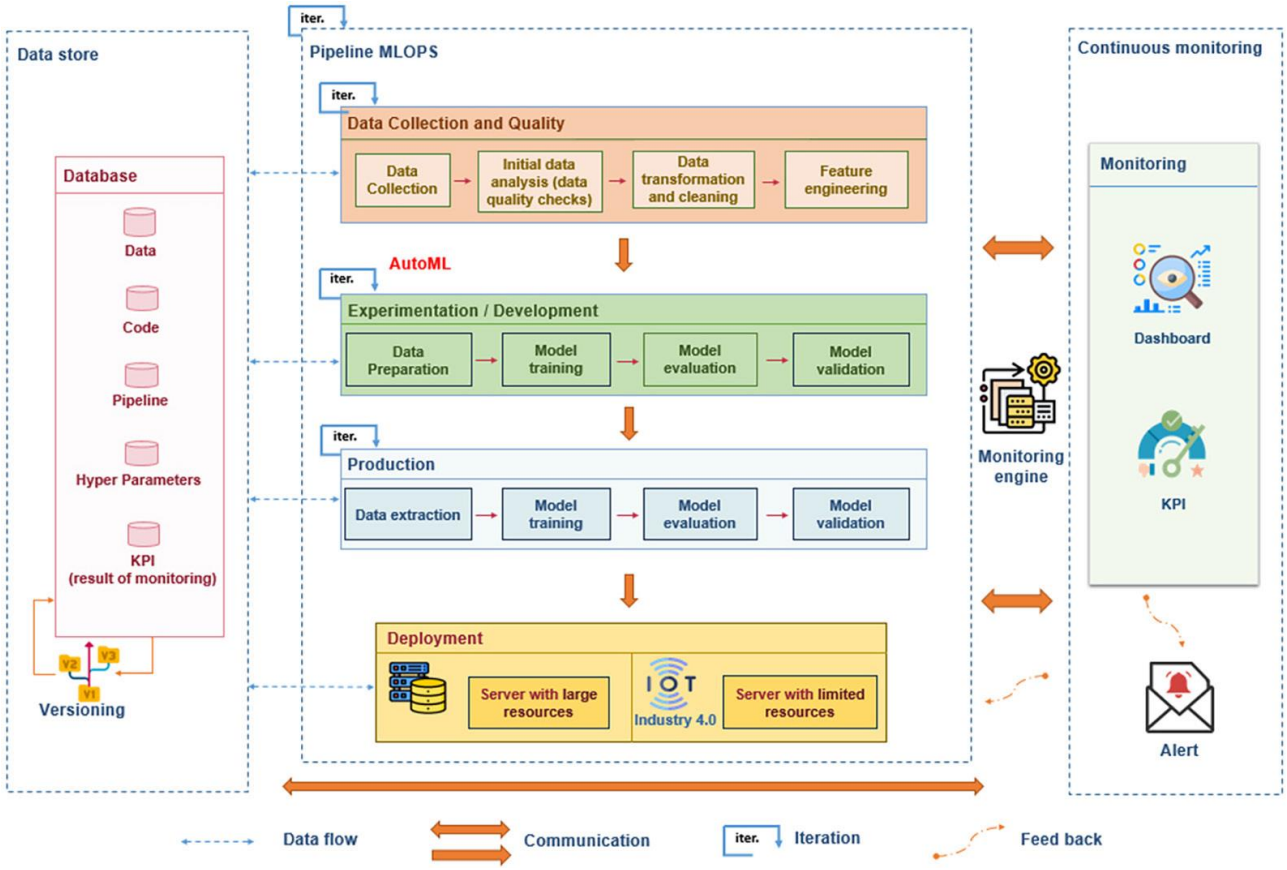

Рисунок 11. Життєвий цикл проєкту машинного навчання у MLOps [25]

Ключові особливості Ease.ML є такими:

1) *процес із залученням людини*, що включає взаємодію з користувачем у структурованій формі, забезпечуючи можливість внесення користувачем даних і прийняття рішень на критичних етапах;
2) *ймовірнісна модель даних*, за якою опрацьовується невизначеність у даних, що може виникати через некоректні дані, слабкий нагляд або через інші причини;
3) *інтерактивне середовище* з блокнотами Jupyter, яке надає змогу користувачам виконувати маніпуляції з даними, запускати операції машинного навчання та візуалізувати результати в інтегрованому середовищі;
4) *графи лінійності* для відстежування взаємодії та операції користувачів під час усього робочого процесу машинного навчання, що сприяє забезпеченню відтворюваності;
5) *автоматичне налаштування якості та рекомендації* для покращення моделей на основі виявлених помилок.

Загальний процес Ease.ML складається з восьми етапів в межах трьох таких підпроцесів:

- Day 0: Pre-ML Subprocess
  1. *Формулювання проблеми* – чітко визначити проблему та цілі машинного навчання.
  2. *Техніко-економічне обґрунтування* – оцінити доцільність використання машинного навчання для вирішення сформульованої проблеми за наявних даних та ресурсів.

- Day 1: AutoML Subprocess
    3. *Підготовка даних* – очищення, попереднє опрацювання та доповнення даних, щоб зробити їх придатними для створення моделей машинного навчання.
    4. *Навчання моделей* – за підготовленими даними навчити моделі з використанням AutoML для автоматизації вибору та налаштування моделей.
    5. *Оцінювання навчених моделей* на валідаційних наборах даних для визначення відповідності критеріям.
    6. *Вибір найкращої моделі* та її розгортання у виробничому середовищі.
- Day 2: Post-ML Subprocess
    7. *Безперервна інтеграція та доставка* – інтеграція моделі у виробниче середовище та налаштування CI / CD конвеєрів для управління оновленнями моделі та моніторингом продуктивності.
    8. *Підтримка моделі* для адаптації до нових даних або умов через постійний моніторинг моделі у виробничому середовищі та виявлення необхідності оновлення або перенавчання моделі.

Безпека та конфіденційність даних

Безпека та конфіденційність даних – це практика захисту інформації та даних, що використовуються в процесах машинного навчання, від несанкціонованого доступу, зловживання та витоку [30]. Вона спрямована на забезпечення цілісності, доступності та конфіденційності даних на всіх етапах життєвого циклу MLOps.

Безпека та конфіденційність даних повинна враховуватися на всіх етапах MLOps, починаючи з визначення проблеми і закінчуючи моніторингом розгорнутої системи. Особливо критичним є забезпечення безпеки на етапах управління даними, розробки та розгортання моделі, оскільки саме тут дані найбільш вразливі [31]. Практика безпеки та конфіденційності даних є обов’язковою у випадках, коли система машинного навчання оперує чутливими даними (персональними, фінансовими, медичними тощо) або розгортається у критично важливих середовищах (охорона здоров’я, автомобільна індустрія, промисловість). Однак навіть для менш чутливих застосувань належний рівень безпеки має бути дотриманий відповідно до нормативних вимог та очікувань користувачів.

На відміну від традиційної розробки програмного забезпечення, у контексті MLOps з’являються нові вектори атак та вразливості, специфічні для машинного навчання (рис. 12). Зокрема, моделі машинного навчання вразливі до атак, як-от отруєння даних (data poisoning), інверсія моделі (model inversion) та атаки з використанням змагальних прикладів (adversarial examples) [30]. Це вимагає застосування спеціалізованих стратегій захисту.

Для дотримання безпеки та конфіденційності даних у MLOps використовуються як стандартні інструменти безпеки (шифрування, автентифікація, журналювання тощо), так і спеціалізовані рішення, орієнтовані на машинне навчання. Прикладами останніх є бібліотеки для безпечного агрегування федеративного навчання, інструменти для тестування моделей на вразливості, фреймворки для конфіденційного навчання на основі гомоморфного шифрування або захищених анклавів [31]. Безпека та конфіденційність даних забезпечується шляхом впровадження різних механізмів контролю та захисту на кожному етапі MLOps. Це

включає, зокрема, шифрування та анонімізацію даних, автентифікацію та контроль доступу, перевірку цілісності, моніторинг активності, реагування на інциденти, а також регулярні перевірки безпеки та тестування на проникнення [30].

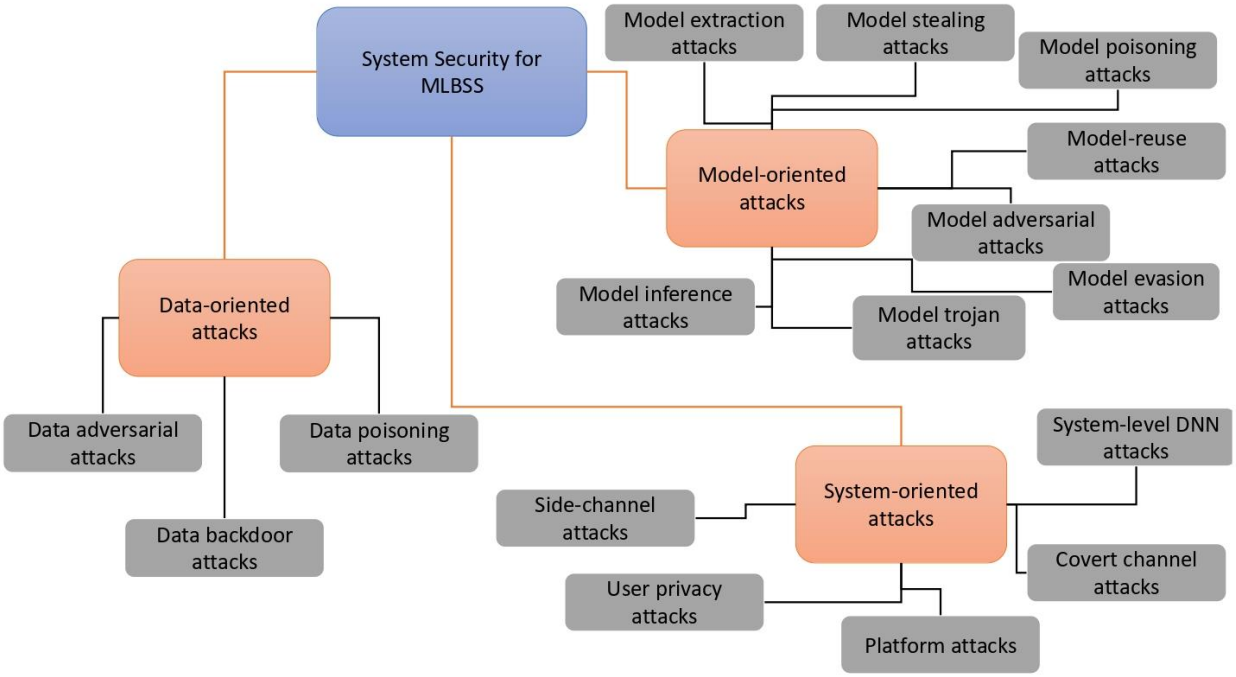

Рисунок 12. Класифікація атак на системи MLOps [30]

На рис. 13 узагальнено подані основні компоненти та практики безпеки у MLOps.

Суттєві компоненти MLOps:

- визначення проблеми (Problem definition) включає розуміння проблеми та вимог до системи;
- управління даними (Data management) охоплює збір, розмітку та верифікацію даних;
- розробка та розгортання моделі (Model construction and deployment) включає вибір, побудову, оптимізацію та оцінювання моделі, а також її розгортання у цільовому середовищі;
- підтримка системи (System maintenance) передбачає моніторинг функціонування розгорнутої системи.

Практики безпеки у MLOps:

- на етапі визначення проблеми – оцінка ризиків (Risk Assessment) та моделювання загроз (Threat Modeling);
- на етапі управління даними – схема потоків даних (Data flow diagram), методологія STRIDE для класифікації загроз, концептуальне моделювання (Conceptual modelling), перевірка даних (Data validation), аналіз якості даних (Data linter) та верифікація даних (Data verification);

- на етапі розробки та розгортання моделі – ідентифікація критеріїв достатності тестування (Test adequacy identification), тестування на змагальних прикладах (Testing against adversarial input), методи “розмиття” (Fuzzing techniques);
- на етапі підтримки системи – засоби моніторингу моделей машинного навчання (MLDEMON, SelfChecker), статичний аналіз (Static Analysis), дослідницькі атаки (Exploratory Attacks), атаки ухилення (Evasion Attacks), атаки отруєння даних (Data Poisoning Attacks), ручне тестування (Manual Testing) та динамічний аналіз (Dynamic Analysis).

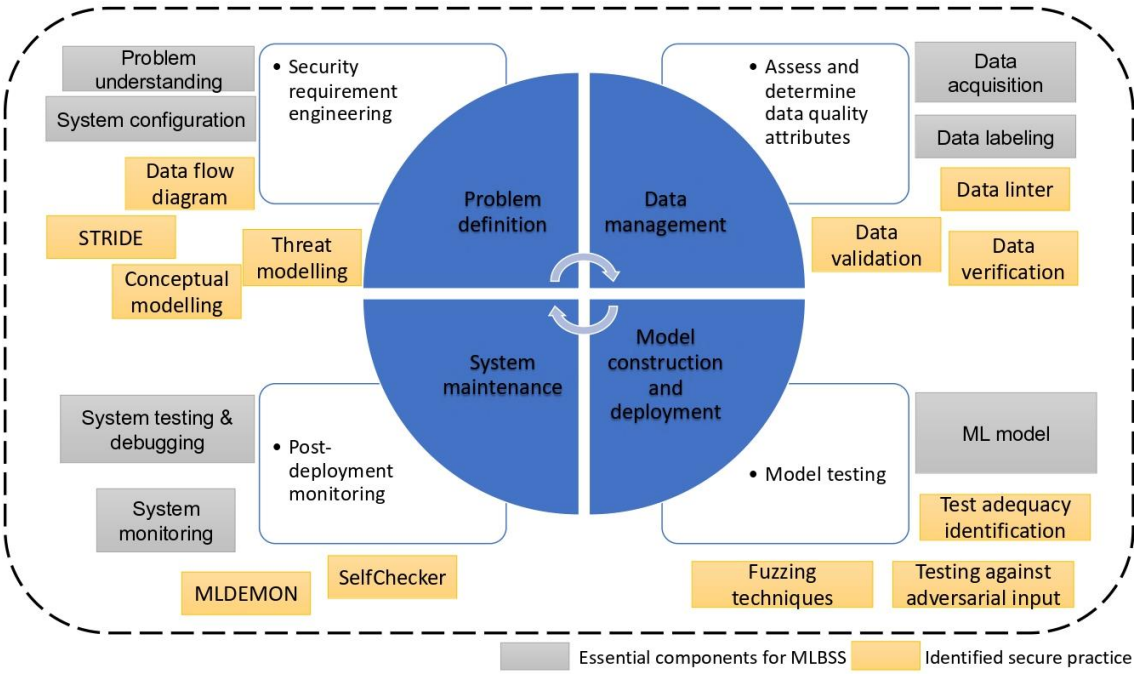

Рисунок 13. Основні компоненти та практики безпеки у MLOps [30]

Отже, практики безпеки у MLOps інтегруються в усі основні етапи життєвого циклу розробки та включають як загальні методи оцінювання та тестування безпеки (моделювання загроз, статичний / динамічний аналіз), так і спеціалізовані підходи, які орієнтовано на особливості систем машинного навчання – тестування на змагальних прикладах, виявлення отруєння даних тощо.

Пояснюваність та інтерпретовність моделей

Пояснюваність та інтерпретовність (Explainability / interpretability) моделей є важливою практикою MLOps для забезпечення прозорості та довіри до моделей машинного навчання. Пояснюваність стосується здатності пояснити та зрозуміти процеси прийняття рішень моделлю машинного навчання [10]. Це особливо важливо для рішень зі значними наслідками, наприклад, у воєнних операціях чи у правоохоронній діяльності [9]. Пояснюваність проєкту машинного навчання дає змогу користувачам довіряти прогнозу. Користувач може перевірити, які фактори вплинули на певні прогнози, що створює додатковий рівень підзвітності. Терміни *пояснюваність* і *інтерпретовність* часто використовуються як

взаємозамінні, однак для MLOPs пояснюваність – це більше, ніж інтерпретовність із погляду важливості, повноти і правильності прогнозів чи класифікацій [4] (рис. 14).

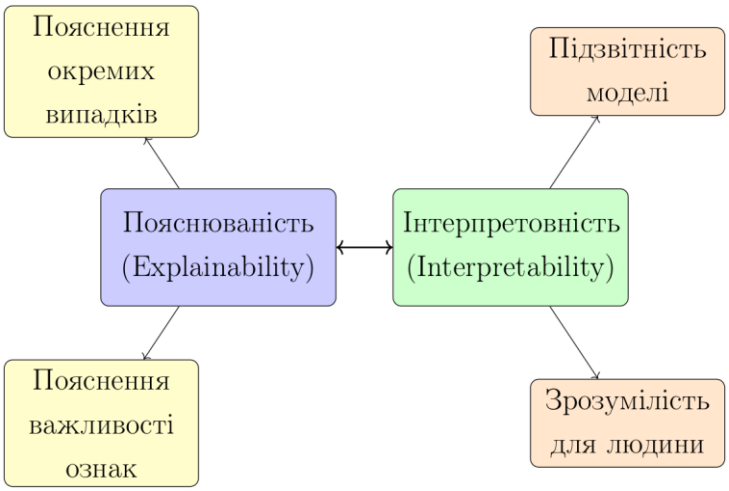

Рисунок 14. Зв'язок пояснюваності та інтерпретовності у MLOps

Пояснюваний штучний інтелект (Explainable Artificial Intelligence, XAI) – це дослідницький напрям, який сприяє прийняттю пояснюваних рішень [4]. Пояснюваність можна визначити як ступінь, до якого людина може зрозуміти причину ухвалення рішення [32]. Система машинного навчання є пояснюваною, коли легко ідентифікуються причинно-наслідкові зв'язки між входами та виходами системи. Чим більш пояснюваною є модель, тим фахівцям-практикам зрозуміліші внутрішні бізнес-процедури, які відбуваються під час прийняття рішень моделлю. Внутрішню логіку або процеси, що лежать в основі пояснюваної моделі, не обов'язково повністю може розуміти людина, але пояснення від моделі дають змогу користувачеві зміцнити довіру до її прогнозів та висновків [4].

Для досягнення пояснюваності використовуються такі способи та інструменти:

- методи на основі атрибуції (Integrated Gradients, Saliency Maps) або пертурбації (SHAP) [33], які пояснюють рішення моделі, призначаючи високі оцінки найвпливовішим вхідним параметрам;
- включення механізму уваги (attention mechanism) у моделі, що дає змогу зосередитись на найбільш релевантних станах мережі та входах [33];
- використання у процесі навчання комбінованого сигналу винагороди (reward), який включає не тільки цільові показники, але й метрики інтерпретовності [33].

Управління якістю даних

Управління якістю даних є важливою практикою у робочому процесі MLOps. У книзі [9] управління якістю даних розглядається в контексті моніторингу та перевірок даних у виробничому середовищі, щоб забезпечити відповідність даних реальності. Згідно зі статтею [7], етап обробки даних включає повний життєвий цикл роботи з даними, зокрема попередню обробку, забезпечення якості, версіонування та документування.

У життєвому циклі MLOps, заснованому на даних (рис. 15), управління якістю даних виконується на таких етапах:

1) збір даних – створюють та отримують з різних джерел;
2) оцінювання якості даних та їх очищення – зібрані дані проходять ретельну перевірку за точністю, повнотою, узгодженістю, своєчасністю та іншими метриками, виявляються та усуваються проблеми з якістю, які обумовлено некоректними значеннями, пропусками, дублікатами та шумом;
3) доповнення та розмітка даних – очищені дані збагачуються додатковою інформацією та розмічаються відповідно до цільової задачі, також контролюється якість розмітки даних, щоб уникнути помилок та неточностей;
4) аналіз якості даних – розмічені дані аналізуються на предмет якості, перевіряється їх репрезентативність, збалансованість класів, наявність викидів та аномалій і за потреби вносяться корективи;
5) навчання моделі з контролем якості – на основі підготовлених даних відбувається навчання моделей з контролем за метриками якості моделей та даних, щоб гарантувати стабільність і надійність результатів;
6) розгортання моделі з забезпеченням якості – модель розгортається у виробничому середовищі лише після ретельного тестування на тестових даних належної якості; також здійснюються заходи для підтримки якості даних у виробничому середовищі;
7) моніторинг якості даних і моделі – розгорнута модель та дані, що надходять, постійно перевіряються на якість – відстежуються метрики якості, наявність аномалій у даних, зміщення розподілів, і за потреби модель донавчається на нових даних належної якості.

Вчасне виявлення і усунення дефектів у даних допомагає запобігти марнуванню обчислювальних ресурсів на неякісних даних [7]. Для гарантування якості даних упродовж усього життєвого циклу їх використання у MLOps необхідно застосовувати такі підходи до валідації та верифікації даних:

- оцінювання якості даних за повнотою, унікальністю, цілісністю, валідністю, точністю та своєчасністю, щоб встановити їх придатність для досягнення бізнес-цілей [26];
- валідація даних у одиночних та перехресних пакетах шляхом порівняння характеристик даних, а також перевірка наявності зсувів даних [7];
- автоматичні модульні тести даних на основі схеми для виявлення помилок у даних та запобігання їх проникнення на етап навчання моделі [7];
- версіонування даних і пов’язаних артефактів (процедур обробки, метаданих тощо) для відслідковуваності та відтворюваності результатів і дотримання регуляторних вимог [7];
- документування даних у обсязі, достатньому для забезпечення верифікованості моделей [7].

У роботі [26] представлено техніки оцінювання якості великих даних, які допомагають виявити набори даних, що можуть спричинити проблеми і зайві витрати.

Для реалізації практик управління якістю даних можна використовувати фреймворки типу TensorFlow Extended, в яких є компоненти для валідації даних, як-от SchemaGen та ExampleValidator [7].

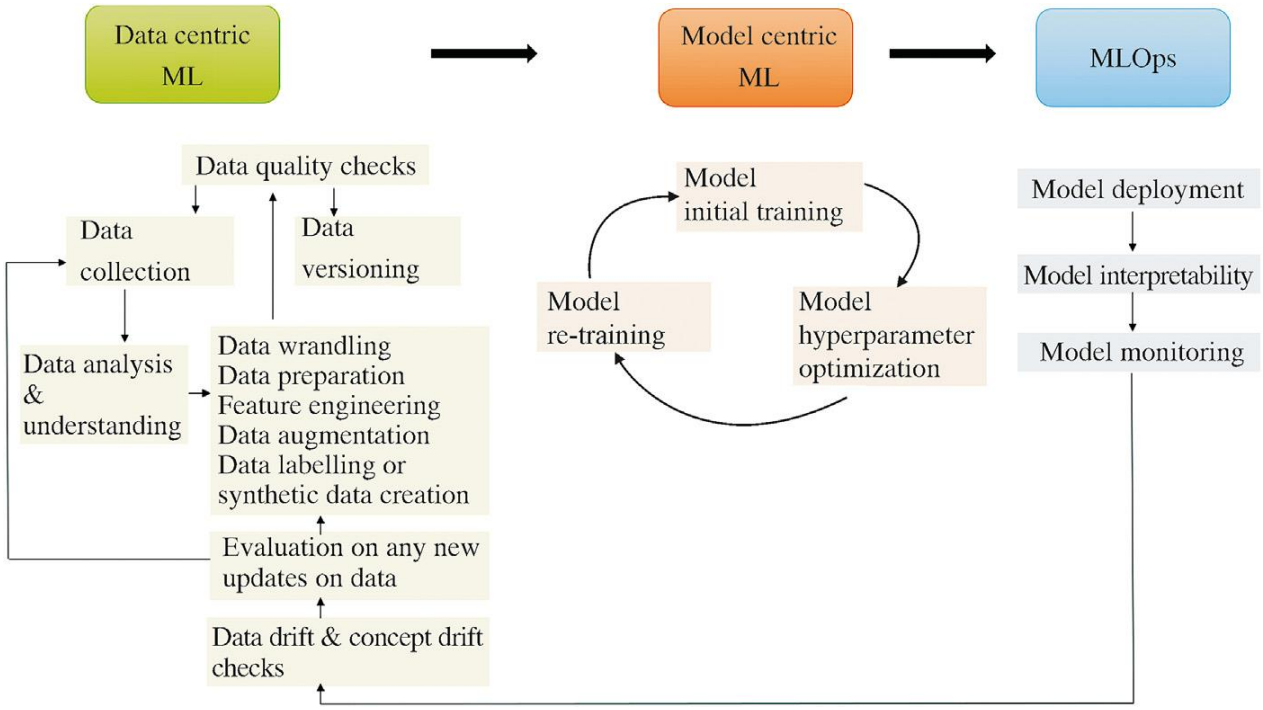

Рисунок 15. Життєвий цикл MLOps, заснований на даних [26]

Управління конфігурацією

Конфігураційні файли надають можливість перемістити всі жорстко закодовані змінні в спеціальні місця, які можуть бути розділені або організовані на розсуд розробника [34]. Це сприяє створенню більш надійного програмного забезпечення.

У [34] запропоновано шаблон, що підтримує 2 типи файлів конфігурації – файл `config.py`, що містить конфігурацію шаблона та може включати додаткові ресурси, як-от бази даних та електронні таблиці, та JSON-файли для збереження специфічних змінних програми, як-от порогові значення чи параметри. Файл `config.py` доступний через інструкції імпорту, а JSON-файли викликаються з диску під час виконання [34].

У [35] розглянуто використання єдиної моделі даних на основі мови YANG для уніфікації опису конфігураційних даних і спрощення управління ними. У [36] зазначено, що інструменти управління конфігурацією, як-от Ansible, Puppet та Chef, можуть використовуватися для автоматизації конфігурування та забезпечення інфраструктури платформ MLOps.

Стратегії розгортання моделей

Стратегії розгортання моделей реалізуються на фінальному етапі процесу MLOps і застосовуються для поступового переходу моделі у виробниче середовище. Вони спираються на контейнеризацію і потребують стандартизації, автоматизації та інкапсуляції моделей. У [28] зазначають такі переваги використання контейнерів для розгортання моделей:

- абстрагування моделей та ізоляція процесів шляхом запуску декількох моделей в окремих контейнерах, що представляють їх залежності під час виконання;
- можливість створити специфічні для кожної моделі контейнери, що відповідають їх вимогам до упаковки;

- призначення на різні процесори – CPU, Mobile GPU тощо;
- можливість виділення окремого контейнера для полегшення функцій постобробки;
- для швидкого розгортання можна використовувати репозиторій моделей та контейнерне сховище;
- забезпечує засоби стандартизованого розгортання;
- майже усі конвеєри неперервної розробки полегшують розгортання моделей у вигляді контейнерів;
- контейнери можна адаптувати до конкретних архітектур периферійних пристроїв;
- контейнери Docker є усталеним стандартом у галузі;
- простота відкату в разі збоїв.

У [37] детально описано цикл розробки та стратегію розгортання моделей на прикладі застосунку DeepClean. Нова версія моделі спочатку розгортається як версія розробника, проходить валідацію в умовах, подібних до виробничих, і тільки після цього замінює поточну модель у виробничому середовищі. Застосовується сервісна архітектура з NVIDIA Triton Inference Server, що підтримує одночасне розміщення версії розробника та виробничої версії моделі.

У [37] виділено 2 основні підходи до розгортання моделей. За традиційного сценарію кожен користувач керує власними ресурсами та версіями моделей. Невідповідності в бібліотеках та залежностях, а також у версіях моделей призводять до непослідовних результатів. Зменшені обчислювальні вимоги до виведення (inference) призводять до недовикористання апаратних ресурсів, що на рис. 16 виділено салатовою заливкою на кожному вузлі. Більш складні сценарії розгортання вимагають використання декількох мереж, шо посилює проблеми.

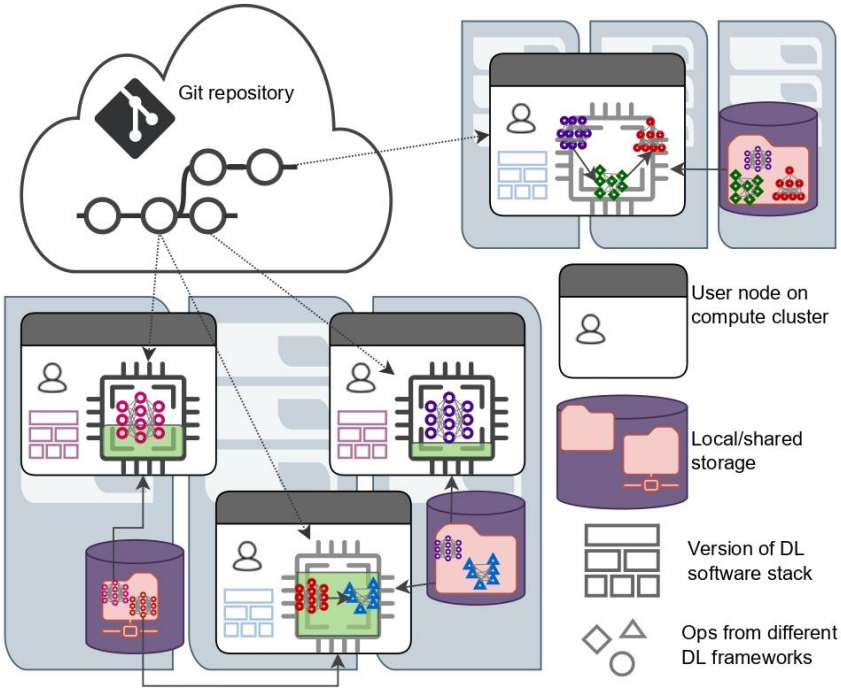

Рисунок 16. Традиційний сценарій розподіленого розгортання [37]

За підходу Inference-as-a-Service централізована служба оркеструє моделі й надає уніфіковані інтерфейси для виклику моделей (рис. 17). Централізоване сховище моделей синхронізує всіх користувачів і підтримує їх в актуальному стані. Конвеєри надсилають gRPC-запити на виведення до сервісу за допомогою стандартизованих API, які абстрагують деталі реалізації самого виконання виведення. Виведення виконується контейнеризованим сервісом, який може ефективно планувати асинхронне виконання моделей, максимально використовуючи обчислювальні можливості апаратних засобів у портативний та у масштабований способи. За такого підходу контейнеризація надає можливість створювати портативні та ізольовані середовища виконання моделей.

Вибір правильної стратегії розгортання моделі дає змогу мінімізувати операційні витрати, забезпечити узгоджену роботу моделі для всіх користувачів та полегшити моніторинг і контроль версій моделі.

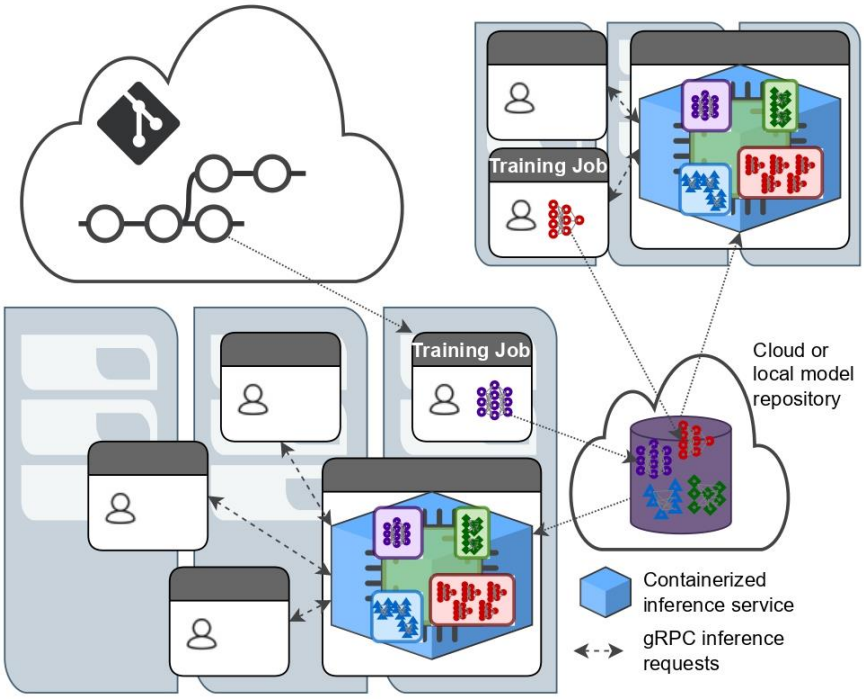

Рисунок 17. Розгортання за сценарієм Inference-as-a-Service [37]

Автоматизація інфраструктури

Автоматизація інфраструктури (Infrastructure as code) є важливою практикою MLOps, яка дає змогу розглядати інфраструктуру як код для її надійного та ефективного розгортання й управління [38]. Автоматизація інфраструктури стосується етапу розгортання та моніторингу у процесі MLOps (рис. 18). Вона забезпечує надійне створення необхідного оточення для розгортання моделей машинного навчання та автоматизує інфраструктурні завдання [38].

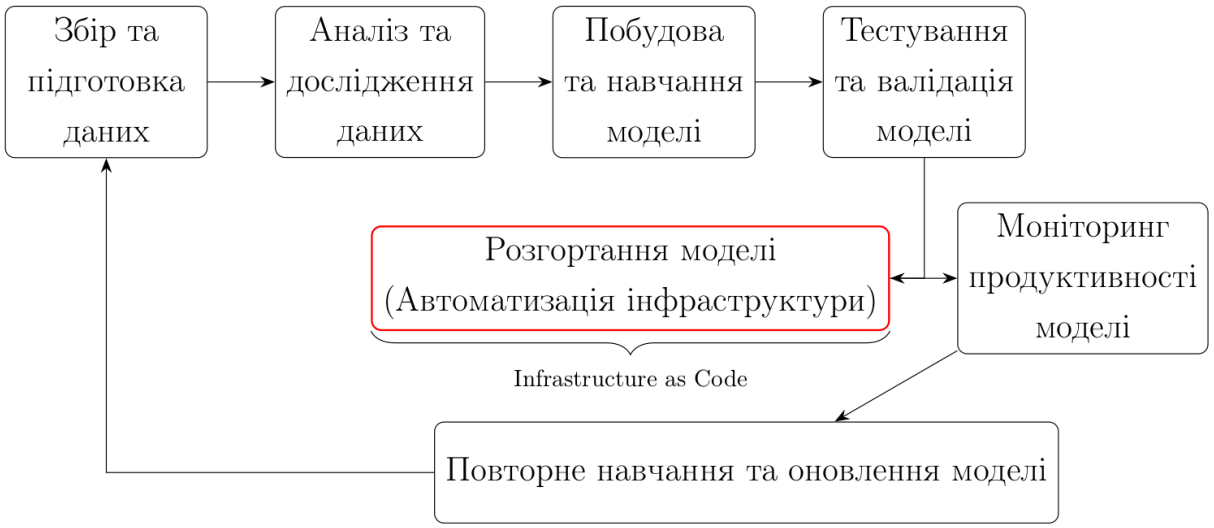

Рисунок 18. Автоматизація інфраструктури у життєвому циклі MLOps

Infrastructure as code передбачає опис конфігурації інфраструктури декларативним способом у спеціальних файлах (наприклад, у форматах YAML чи JSON) або за допомогою спеціальних мов (наприклад, Terraform) чи інструментів. Ці файли описують бажаний стан інфраструктури [38].

Опис налаштувань інфраструктури у вигляді коду дає змогу автоматизувати процес її створення, зміни та управління [38]. Це допомагає повністю відтворити оточення, швидко розгортати ресурси та уникати помилок ручного налаштування [38]. Такий підхід доцільний під час створення складних динамічних інфраструктур, коли потрібна висока повторюваність і узгодженість оточень, а також для підвищення надійності та зниження витрат часу на ручні налаштування [38]. Автоматизований підхід дає змогу тестувати інфраструктуру як код, застосовувати до неї практики з розробки програмного забезпечення – перегляд коду, версіонування тощо. Це сприяє підвищенню стабільності та безпеки і допомагає оперативно відслідковувати та усувати проблеми в інфраструктурі [38].

Для реалізації Infrastructure as code у Azure DevOps застосовуються інструменти Azure Resource Manager (ARM) Templates, Terraform, Ansible та Chef [38]. Вони дають змогу описати інфраструктуру у вигляді коду та автоматизовано створювати чи змінювати її.

Співпраця та комунікація

Співпраця та комунікація між різними стейкхолдерами є ключовою практикою MLOps для успішного впровадження проєктів машинного навчання та штучного інтелекту в організаціях. MLOps акцентує на тому, як кросфункціональні команди з аналітиків даних, системних операторів та інженерів з даних та програмного забезпечення співпрацюють через узгоджений процес [7]. Сутність цієї практики полягає у налагодженні ефективної взаємодії та обміну інформацією між різними командами, залученими до процесу розробки і впровадження моделей машинного навчання – командами науки про дані (data science), розробки (development), операційної діяльності (operations) та бізнес-підрозділами. Така практика є важливим складником етапу розробки та впровадження робочого процесу MLOps.

MLOps передбачає тісну співпрацю між командами науковців з машинного навчання, які займаються підготовкою даних та розробкою моделей, інженерами-розробниками, які відповідають за інтеграцію моделей у виробниче середовище, та операційними командами, які забезпечують розгортання та підтримку моделей [2]. Ефективна комунікація необхідна на всіх етапах життєвого циклу моделей машинного навчання, починаючи від визначення бізнес-цілей та закінчуючи моніторингом моделей у виробничому середовищі. Без налагодженої співпраці розробка може затягуватись, виникатимуть конфлікти інтересів та непорозуміння між учасниками [39].

Рис. 19 відображає основних учасників процесу MLOps та напрями їх взаємодії:

- команда Data Science готує дані та розробляє моделі машинного навчання;
- команда Development інтегрує розроблені моделі у програмні продукти;
- команда Operations розгортає та підтримує моделі у виробничому середовищі;
- команда Business визначає цілі та вимоги до рішень на базі машинного навчання, а також отримує результати від команд Data Science та Operations.

Рисунок 19. Схема співпраці та комунікації у MLOps

Для забезпечення ефективної комунікації у практиці MLOps застосовують такі підходи та інструменти [2]:

- використання інструментів для спільної роботи та обміну знаннями, як-от Slack, Trello, GitLab wiki;
- регулярні зустрічі між командами для обговорення статусу, проблем та планів;
- чітке визначення ролей та зон відповідальності учасників процесу MLOps;
- використання систем контролю версій для спільної роботи над кодом та моделями;
- автоматизація процесів CI / CD для прозорості та відтворюваності розробки.

Управління ризиками та комплаєнс

Оскільки моделі машинного навчання часто приймають важливі рішення, що впливають на людей, управління ризиками та комплаєнс є критично важливою практикою MLOps. Вона полягає у забезпеченні відповідності розроблюваних моделей та систем машинного навчання нормативним вимогам і стандартам (compliance) та управлінні потенційними ризиками їх розробки та експлуатації. Ця практика має наскрізний характер та проявляється на різних етапах життєвого циклу моделі машинного навчання. Управління ризиками в контексті MLOps передбачає виявлення та зменшення потенційних ризиків, пов'язаних із моделями машинного навчання, як-от упередженість даних, порушення конфіденційності, погіршення точності моделі з часом тощо.

Комплаєнс означає забезпечення відповідності моделей нормативним вимогам, наприклад, щодо захисту даних [7]. Основною умовою використання цієї практики є наявність нормативно-правових вимог чи галузевих стандартів, яким повинна відповідати система машинного навчання. Прикладами можуть бути вимоги GDPR щодо захисту персональних даних чи сертифікації у сфері охорони здоров'я [7]. Аспекти комплаєнсу мають враховуватись під час підготовки даних, навчання та валідації моделі, а також під час розгортання та моніторингу. Способи використання цієї практики включають регулярні перевірки якості даних, тестування моделей на упередженість і дискримінацію, впровадження контролю доступу та шифрування даних, документування архітектури моделі і процесу розробки, моніторинг продуктивності моделі після розгортання [7].

Провідними засобами забезпечення практики управління ризиками та комплаєнсу є системи контролю версій для відстеження змін у даних та коді моделі, засоби тестування для виявлення проблем у моделях, системи моніторингу для відстеження точності моделей у реальному часі [7]. Інтерв'ю з практиками виявили, що забезпечення комплаєнсу ML-систем у різних доменах (наприклад, в охороні здоров'я) є серйозним викликом через брак усталених методик та програмних інструментів [7]. Водночас досягнення комплаєнсу є обов'язковою умовою для отримання необхідних сертифікацій та дозволів регуляторів.

Рис. 20 відображає 3 основні етапи MLOps: роботу з даними, навчання моделі та розгортання системи (операціоналізацію). На кожному етапі наявні блоки, які позначають потенційні ризики та заходи із забезпечення відповідності. Схема ілюструє наскрізний характер практики управління ризиками та комплаєнсу в MLOps, показуючи взаємозв'язки на кожному етапі життєвого циклу моделі машинного навчання.

Рис. 21 показує взаємозв'язки між ключовими принципами, процесом розгортання та основними практиками MLOps, які застосовуються на етапі розгортання моделей машинного навчання. Принципи автоматизації і відтворюваності впливають на процес розгортання моделей, який пов'язаний з наступними практиками MLOps, як-ОТ CI / CD, розгортання моделей, безпека та конфіденційність даних, управління конфігурацією, стратегії розгортання моделей та автоматизація інфраструктури.

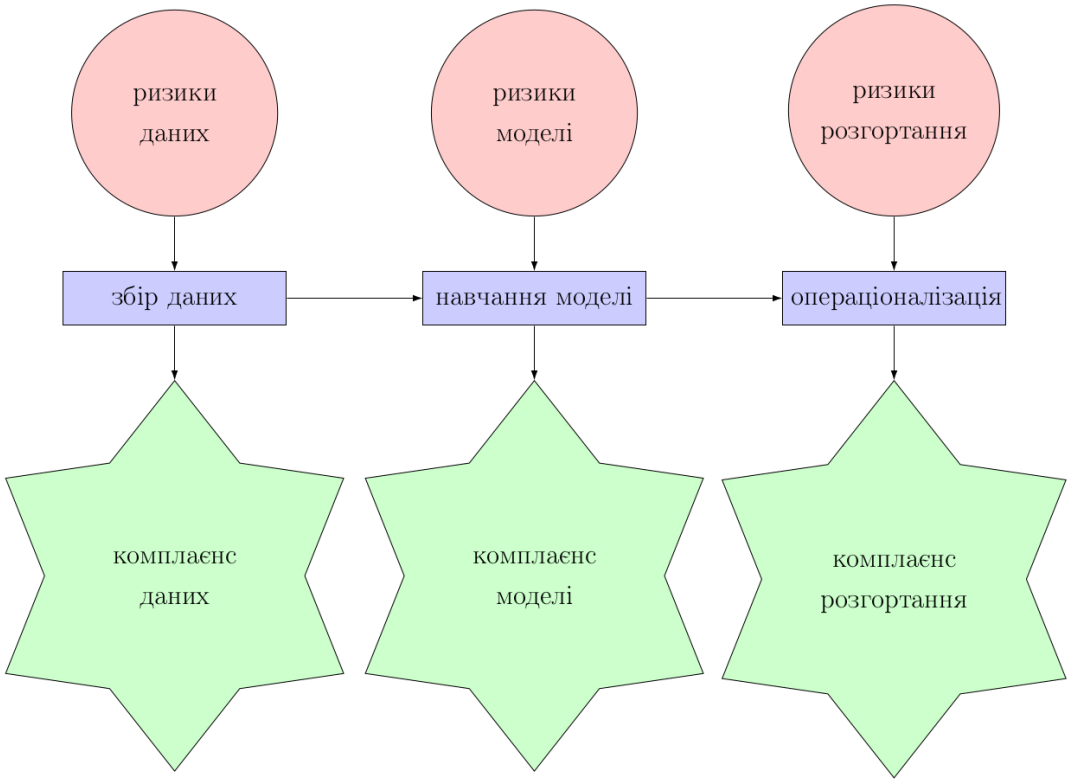

Рисунок 20. Схема практики управління ризиками та комплаєнсу у MLOps

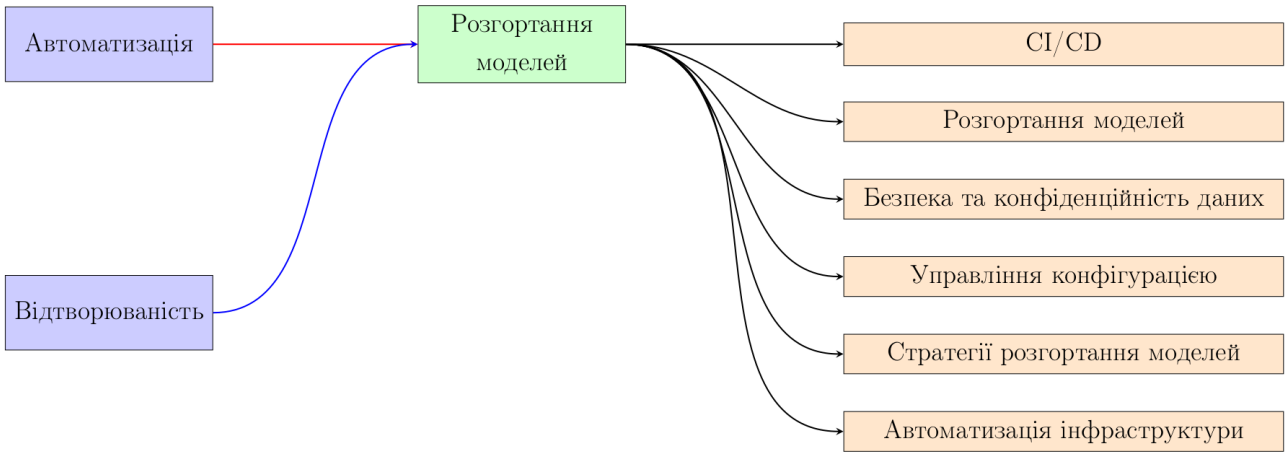

Рисунок 21. Зв'язки принципів, процесів та практик MLOps під час розгортання моделей

Висновки

Здійснено метасинтез систематичних оглядів [5, 6, 8] та огляду продуктів і постачальників [15] з метою узагальнення знань про впровадження практик MLOps для ефективного розгортання моделей машинного навчання. За результатами метасинтезу сформовано 10 висновків.

1. MLOps – це підхід до управління, автоматизації та операціоналізації процесів розробки, розгортання та підтримки моделей машинного навчання на основі практик програмної інженерії та DevOps. MLOps базується на наборі принципів, процесів і практик, що

забезпечують ефективну розробку, розгортання та підтримку моделей машинного навчання.

2. Основні етапи життєвого циклу MLOps включають процеси збору та обробки даних, розробки та навчання моделей, розгортання, моніторинг та донавчання моделей.
3. Для впровадження MLOps використовуються різні фреймворки та архітектури, як-от платформи з відкритим кодом MLflow, Kubeflow та TensorFlow Extended, хмарні обчислювальні платформи AWS, Google Cloud та Azure, контейнеризація (Docker) і оркестрація контейнерів (Kubernetes).
4. Інструменти MLOps пропонують широкий спектр функцій для підтримки життєвого циклу моделі машинного навчання, з акцентом на автоматизацію, відстеження експериментів, версіонування, моніторинг та розгортання моделей.
5. Найпоширенішими способами розгортання моделей машинного навчання в робочих середовищах є використання технологій контейнеризації, хмарних платформ і сервісів, а також розгортання моделей як вебсервісів.
6. Для оцінювання рівня зрілості процесів MLOps в організаціях можуть використовуватися адаптовані моделі зрілості розробки програмного забезпечення, як-от Capability Maturity Model.
7. Успішне впровадження MLOps вимагає залучення фахівців із різних сфер – з розробки програмного забезпечення, інженерії даних, машинного навчання, предметних експертів і менеджменту.
8. Основними викликами під час розгортання моделей машинного навчання в робочих середовищах є управління життєвим циклом моделі, забезпечення масштабованості та продуктивності, моніторинг і підтримка моделей у реальних умовах.
9. Відкритими питаннями та викликами в MLOps є необхідність розробки стандартів і найкращих практик, забезпечення інтерпретованості та відповідального використання моделей, ефективне управління даними та інтеграція знань із різних сфер.
10. Основними можливостями та тенденціями розвитку MLOps є створення стандартизованих платформ, застосування в контексті розподіленого навчання та інтеграція з іншими підходами до управління життєвим циклом даних і моделей. Поточні та майбутні сфери застосування MLOps включають широкий спектр галузей, від фінансів і охорони здоров’я до Інтернету речей та обробки природної мови.

Проведений метасинтез показав, що MLOps є перспективним підходом для ефективного розгортання моделей машинного навчання в робочих середовищах, який вимагає подальших досліджень і розробок для вирішення поточних викликів та реалізації потенційних можливостей.

Також проаналізовано ключові практики MLOps, які необхідні для ефективного розгортання моделей машинного навчання. За результатами аналізу сформовано 5 висновків.

1. MLOps базується на наборі принципів, процесів і практик, що забезпечують ефективну розробку, розгортання та підтримку моделей машинного навчання. Ключовими принципами MLOps є автоматизація, відтворюваність, співпраця, постійне навчання та управління даними.

2. Основні практики MLOps включають безперервну інтеграцію та доставку (CI / CD), версіонування моделей і даних, автоматизацію конвеєрів машинного навчання, моніторинг продуктивності моделей, управління експериментами, розгортання моделей і управління життєвим циклом.
3. Додаткові практики MLOps, як-от безпека даних і конфіденційність, пояснюваність і інтерпретованість моделей, управління якістю даних, управління конфігурацією, стратегії розгортання моделей, автоматизація інфраструктури, співпраця та комунікація, управління ризиками й відповідність вимогам, є важливими для забезпечення надійності, відповідності вимогам та ефективності процесів MLOps.
4. Застосування практик MLOps дає змогу автоматизувати та стандартизувати процеси розробки, розгортання й підтримки моделей машинного навчання, що підвищує ефективність і надійність рішень у робочому середовищі.
5. Успішне впровадження практик MLOps вимагає використання відповідних інструментів і платформ, як-от системи управління експериментами, версіонування даних і моделей, автоматизація інфраструктури та інструменти моніторингу, а також налагодження ефективної співпраці між різними ролями й командами, залученими до розробки та впровадження моделей машинного навчання.

Аналіз показав, що застосування практик MLOps є критично важливим для успішного розгортання моделей машинного навчання в робочих середовищах. Впровадження таких практик сприяє підвищенню ефективності, надійності і відтворюваності процесів розробки та експлуатації рішень на базі машинного навчання, що є необхідною умовою для їх успішного використання в реальних бізнес-задачах.

Внаслідок комплексного дослідження ідентифіковано та проаналізовано практики MLOps, які необхідні для ефективного розгортання моделей машинного навчання. Основні висновки, отримані під час дослідження, є такими:

1) MLOps є перспективним підходом для ефективного розгортання моделей машинного навчання в робочих середовищах, який вимагає подальших досліджень і розробок для вирішення поточних викликів і реалізації можливостей.
2) запропонована діаграма взаємозв’язків між принципами, процесами та практиками MLOps, яка наочно ілюструє їх взаємодію на кожному етапі процесу розробки та впровадження моделей машинного навчання;
3) визначено найбільш ефективні практики MLOps для розгортання моделей, які включають безперервну інтеграцію та доставку (CI / CD), версіонування моделей і даних, автоматизацію конвеєрів машинного навчання, моніторинг продуктивності моделей, управління експериментами, розгортання моделей і управління життєвим циклом, безпеку даних та конфіденційність, пояснюваність і інтерпретованість моделей, управління якістю даних, управління конфігурацією, стратегії розгортання моделей, автоматизацію інфраструктури, співпрацю та комунікацію, управління ризиками та відповідність вимогам.

Теоретична значущість отриманих у статті результатів полягає в узагальненні та систематизації знань про практики MLOps, які необхідні для ефективного розгортання моделей машинного навчання. Практична значущість результатів полягає в можливості їх

використання організаціями для впровадження або вдосконалення процесів MLOps з метою підвищення ефективності та надійності розгортання моделей машинного навчання в робочих середовищах.

Подальші дослідження можуть бути спрямовані на розробку детальних рекомендацій щодо впровадження окремих практик MLOps в організаціях, створення нових інструментів і платформ для автоматизації та управління життєвим циклом моделей машинного навчання, а також на вивчення ефективності практик MLOps у різних галузях і сферах застосування моделей машинного навчання.

Література

1. Zahorodko, P. V., Semerikov, S. O., Soloviev, V. N., Striuk, A. M., Striuk, M. I., & Shalatska, H. M. (2021). Comparisons of performance between quantum-enhanced and classical machine learning algorithms on the IBM Quantum Experience. *Journal of Physics: Conference Series*, *1840*, 012021. https://doi.org/10.1088/1742-6596/1840/1/012021
2. Kreuzberger, D., Kühl, N., & Hirschl, S. (2023). Machine learning operations (MLOps): overview, definition, and architecture. *IEEE Access*, *11*, 31866–31879. https://doi.org/10.1109/ACCESS.2023.3262138
3. Symeonidis, G., Nerantzis, E., Kazakis, A., & Papakostas, G. A. (2022). MLOps – definitions, tools and challenges. In *2022 IEEE 12th Annual Computing and Communication Workshop and Conference (CCWC)* (pp. 0453–0460). https://doi.org/10.1109/CCWC54503.2022.9720902
4. Testi, M., Ballabio, M., Frontoni, E., Iannello, G., Moccia, S., Soda, P., & Vessio, G. (2022). MLOps: A taxonomy and a methodology. *IEEE Access*, *10*, 63606–63618. https://doi.org/10.1109/ACCESS.2022.3181730
5. Diaz-de Arcaya, J., Torre-Bastida, A. I., Zárate, G., Miñón, R., & Almeida, A. (2023). A joint study of the challenges, opportunities, and roadmap of MLOps and AIOps: A systematic survey. *ACM Computing Surveys*, *56*, 84. https://doi.org/10.1145/3625289
6. Recupito, G., Pecorelli, F., Catolino, G., Moreschini, S., Nucci, D. D., Palomba, F., & Tamburri, D. A. (2022). A multivocal literature review of mlops tools and features. In *2022 48th Euromicro Conference on Software Engineering and Advanced Applications (SEAA)* (pp. 84–91). https://doi.org/10.1109/SEAA56994.2022.00021
7. Steidl, M., Felderer, M., & Ramler, R. (2023). The pipeline for the continuous development of artificial intelligence models – current state of research and practice. *Journal of Systems and Software*, *199*, 111615. https://doi.org/10.1016/j.jss.2023.111615
8. Lima, A., Monteiro, L., & Furtado, A. P. (2022). MLOps: practices, maturity models, roles, tools, and challenges – a systematic literature review. In *Proceedings of the 24th International Conference on Enterprise Information Systems – Volume 1: ICEIS* (pp. 308–320). https://doi.org/10.5220/0010997300003179
9. Haller, K. (2022). *Managing AI in the enterprise: Succeeding with AI projects and MLOps to build sustainable AI organizations*. Apress Berkeley, CA. https://doi.org/10.1007/978-1-4842-7824-6

10. e Oliveira, E., Rodrigues, M., Pereira, J. P., Lopes, A. M., Mestric, I. I., & Bjelogrlic, S. (2024). Unlabeled learning algorithms and operations: overview and future trends in defense sector. *Artificial Intelligence Review*, *57*, 66. https://doi.org/10.1007/s10462-023-10692-0
11. Kolltveit, A. B., & Li, J. (2023). Operationalizing machine learning models: A systematic literature review. In *Proceedings of the 1st Workshop on Software Engineering for Responsible AI* (pp. 1–8). https://doi.org/10.1145/3526073.3527584
12. Calefato, F., Lanubile, L., & Quaranta, L. (2022). A preliminary investigation of MLOps practices in GitHub. In *Proceedings of the 16th ACM / IEEE International Symposium on Empirical Software Engineering and Measurement* (pp. 283–288). https://doi.org/10.1145/3544902.3546636
13. Page, M. J., McKenzie, J. E., Bossuyt, P. M., Boutron, I., Hoffmann, T. C., Mulrow, C. D., … Moher, D. (2021). The PRISMA 2020 statement: an updated guideline for reporting systematic reviews. *BMJ*, *372*, n71. https://doi.org/10.1136/bmj.n71
14. Haertel, C., Staegemann, D., Daase, C., Pohl, M., Nahhas, A., & Turowski, K. (2023). MLOps in data science projects: A review. In *2023 IEEE International Conference on Big Data (BigData)* (pp. 2396–2404). https://doi.org/10.1109/BigData59044.2023.10386139
15. Cohen, R. (2023). Digital strategy, machine learning, and industry survey of MLOps. In *Digital Strategies and Organizational Transformation* (pp. 137–150). https://doi.org/10.1142/9789811271984_0008
16. Sipe, T. A., & Curlette, W. L. (1996). A meta-synthesis of factors related to educational achievement: A methodological approach to summarizing and synthesizing meta-analyses. *International Journal of Educational Research*, *25*, 583–698. https://doi.org/10.1016/S0883-0355(96)80001-2
17. Chrastina, J. (2018). Meta-synthesis of qualitative studies: Background, methodology and applications. In *NORDSCI Conference proceedings* (Vol. 1). https://doi.org/10.32008/nordsci2018/b1/v1/13
18. Amershi, S., Begel, A., Bird, C., DeLine, R., Gall, H., Kamar, E., Nagappan, N., Nushi, B., & Zimmermann, T. (2019). Software engineering for machine learning: A case study. In *2019 IEEE/ACM 41st International Conference on Software Engineering: Software Engineering in Practice (ICSE-SEIP)* (pp. 291–300). https://doi.org/10.1109/ICSE-SEIP.2019.00042
19. Dhanorkar, S., Wolf, C. T., Qian, K., Xu, A., Popa, L., & Li, Y. (2021). Who needs to know what, when?: Broadening the explainable AI (XAI) design space by looking at explanations across the AI lifecycle. In *Proceedings of the 2021 ACM Designing Interactive Systems Conference* (pp. 1591–1602). https://doi.org/10.1145/3461778.3462131
20. Lwakatare, L. E., Crnkovic, I., & Bosch, J. (2020). DevOps for AI – challenges in development of AI-enabled applications. In *2020 International Conference on Software, Telecommunications and Computer Networks (SoftCOM)* (pp. 1–6). https://doi.org/10.23919/SoftCOM50211.2020.9238323
21. Akkiraju, R., Sinha, V., Xu, A., Mahmud, J., Gundecha, P., Liu, Z., Liu, X., & Schumacher, J. (2020). Characterizing machine learning processes: A maturity framework. In *D. Fahland, C. Ghidini, J. Becker, & M. Dumas (eds.), Business Process Management* (Vol. 12168, pp. 17–31). Springer International Publishing. https://doi.org/10.1007/978-3-030-58666-9_2

22. Min, C., Mathur, A., Acer, U. G., Montanari, A., & Kawsar, F. (2023). SensiX++: Bringing MLOps and multi-tenant model serving to sensory edge devices. *ACM Transactions on Embedded Computing Systems*, *22*, 98. https://doi.org/10.1145/3617507
23. Bachinger, F., Zenisek, J., & Affenzeller, M. (2024). Automated machine learning for industrial applications – challenges and opportunities. *Procedia Computer Science*, *232*, 1701–1710. https://doi.org/10.1016/j.procs.2024.01.168
24. Filippou, K., Aifantis, G., Papakostas, G. A., & Tsekouras, G. E. (2023). Structure learning and hyperparameter optimization using an automated machine learning (AutoML) Pipeline. *Information*, *14*, 232. https://doi.org/10.3390/info14040232
25. Bodor, A., Hnida, M., & Daoudi, N. (2023). Machine learning models monitoring in MLOps context: Metrics and tools. *International Journal of Interactive Mobile Technologies (iJIM)*, *17*, 125–139. https://doi.org/10.3991/ijim.v17i23.43479
26. Singh, P. (2023). Systematic review of data-centric approaches in artificial intelligence and machine learning. *Data Science and Management*, *6*, 144–157. https://doi.org/10.1016/j.dsm.2023.06.001
27. Czakon, J., & Kluge, K. (2024). ML experiment tracking: What it is, why it matters, and how to implement it. https://neptune.ai/blog/ml-experiment-tracking
28. Peltonen, E., & Dias, S. (2023). LinkEdge: Open-sourced MLOps integration with IoT edge. In *Proceedings of the 3rd Eclipse Security, AI, Architecture and Modelling Conference on Cloud to Edge Continuum* (pp. 67–76). https://doi.org/10.1145/3624486.3624496
29. Melgar, L. A., Dao, D., Gan, S., Gürel, N. M., Hollenstein, N., Jiang, J., … Zhang, C. (2021). Ease.ML: A lifecycle management system for MLDev and MLOps. In *11th Conference on Innovative Data Systems Research, CIDR 2021*. https://www.cidrdb.org/cidr2021/papers/cidr2021_paper26.pdf
30. Chen, H., & Babar, M. A. (2024). Security for machine learning-based software systems: A survey of threats, practices, and challenges. *ACM Computing Surveys*, *56*, 151. https://doi.org/10.1145/3638531
31. Gopalakrishna, N. K., Anandayuvaraj, D., Detti, A., Bland, F. L., Rahaman, S., & Davis, J. C. (2023). "If security is required": engineering and security practices for machine learning-based IoT devices. In *Proceedings of the Fourth International Workshop on Software Engineering Research and Practice for the IoT* (pp. 1–8). https://doi.org/10.1145/3528227.3528565
32. Miller, T. (2019). Explanation in artificial intelligence: Insights from the social sciences. *Artificial Intelligence*, *267*, 1–38. https://doi.org/10.1016/j.artint.2018.07.007
33. Rezazadeh, F., Chergui, H., Alonso, L., & Verikoukis, C. (2024). SliceOps: Explainable MLOps for streamlined automation-native 6G networks. *IEEE Wireless Communications*, *31*, 224–230. https://doi.org/10.1109/MWC.007.2300144
34. Godwin, R. C., & Melvin, R. L. (2024). Toward efficient data science: A comprehensive MLOps template for collaborative code development and automation. *SoftwareX*, *26*, 101723. https://doi.org/10.1016/j.softx.2024.101723

35. Yongqiang, D., Xin, W., Yongbo, L., & Wang, Y. (2020). Building network domain knowledge graph from heterogeneous YANG models. *Journal of Computer Research and Development*, *57*, 699–708. https://doi.org/10.7544/issn1000-1239.2020.20190882
36. Neptune Labs. (2024). MLOps landscape in 2024: Top tools and platforms. https://neptune.ai/blog/mlops-tools-platforms-landscape
37. Gunny, A., Rankin, D., Harris, P., Katsavounidis, E., Marx, E., Saleem, M., Coughlin, M., & Benoit, W. (2022). A software ecosystem for deploying deep learning in gravitational wave physics. In *Proceedings of the 12th Workshop on AI and Scientific Computing at Scale Using Flexible Computing Infrastructures* (pp. 9–17). https://doi.org/10.1145/3526058.3535454
38. Vuppalapati, C., Ilapakurti, A., Chillara, K., Kedari, S., & Mamidi, V. (2020). Automating tiny ML intelligent sensors DevOPS using Microsoft Azure. In *2020 IEEE International Conference on Big Data (Big Data)* (pp. 2375–2384). https://doi.org/10.1109/BigData50022.2020.9377755
39. Sothilingam, R., Pant, V., & Yu, E. S. K. (2022). Using i* to analyze collaboration challenges in MLOps project teams. In A. Maté, T. Li, & E. J. T. Gonçalves (eds.), *Proceedings of the 15th International iStar Workshop (iStar 2022) co-located with 41th International Conference on Conceptual Modeling (ER 2022)* (pp. 1–6). https://ceur-ws.org/Vol-3231/iStar22_paper_1.pdf

Рукопис отримано – 04/03/2025 р.; прийнято до публікації – 26/03/2025 р.

MLOps engineering: A meta-synthesis of tools, practices and architectures for machine learning automation

Danylo Hanchuk, Serhiy Semerikov

Abstract

Automating the end-to-end lifecycle of machine learning models is critical for their effective operationalization in production environments. Various tools, frameworks and architectures have emerged to support Machine Learning Operations (MLOps) practices. This paper presents a meta-synthesis of existing reviews to provide a comprehensive overview of enabling technologies for MLOps. The capabilities and features offered by popular commercial and open-source MLOps platforms are compared. Patterns in MLOps architecture and design philosophies are identified. The paper examines the role of containerization, orchestration, configuration management, and infrastructure automation in ML-pipelines. Approaches for model deployment on cloud and edge are also discussed. The following main results are obtained: 1) a meta-synthesis of systematic reviews was conducted to summarize knowledge about MLOps practices; it was determined that MLOps is a promising approach for effective deployment of machine learning models that requires further research; 2) relationships between MLOps principles, processes, and practices were analyzed. A diagram of the interconnections between key principles, stages of model development and implementation, and main MLOps practices is proposed; 3) the most effective MLOps practices for model deployment were identified – continuous integration/delivery, model and data versioning, ML pipeline automation, performance monitoring, experiment management, lifecycle management, data security and privacy, model explainability, data quality management, configuration management, deployment strategies, infrastructure automation, collaboration, risk management. The results obtained have theoretical significance in generalizing and systematizing knowledge about MLOps practices and practical significance for implementing and improving MLOps processes in organizations.

Keywords: MLOps, automation, tools, frameworks, architecture, model deployment, ML-pipelines, meta-synthesis.

УДК 004.85+004.8.032.26

Еволюція нейронних моделей генерування тексту: систематичний огляд досліджень 2022–2024 років

Артем Валерійович Слободянюк
ORCID: 0009-0007-9425-1255
minekosdid@kdpu.edu.ua

Криворізький державний педагогічний університет

Сергій Олексійович Семеріков
професор, старший дослідник, д-р пед. наук
ORCID: 0000-0003-0789-0272
semerikov@gmail.com

Криворізький державний педагогічний університет
Інститут цифровізації освіти НАПН України
Державний університет "Житомирська політехніка"
Криворізький національний університет
Академія когнітивних та природничих наук

Ключові слова:
нейроне генерування тексту;
глибоке навчання;
систематичний огляд;
обробка природної мови;
метрика;
набори даних;
низькоресурсні мови;
трансформери;
механізми уваги.

Останні роки характеризуються значним прогресом у сфері нейронного генерування тексту завдяки появі великих мовних моделей та зростанню інтересу до цієї галузі. Цей систематичний огляд ідентифікує та узагальнює сучасні тенденції, підходи та методи нейронного генерування тексту за період 2022–2024 рр., доповнюючи попередній огляд за 2015–2021 рр. Відповідно до методології PRISMA, для аналізу було початково відібрано 89 статей з бази даних Scopus, із яких після перевірки критеріїв включення та виключення залишилося 43 статті. Виявлено зміщення акценту в бік інноваційних архітектур моделей, як-от Transformer-based (GPT-2, GPT-3, BERT), механізмів уваги та контрольованого генерування тексту. Метрики BLEU, ROUGE та оцінювання людиною залишаються найпопулярнішими. Але з'явилися і нові метрики, поміж яких виділимо BERTScore. Набори даних охоплюють різноманітні домени і типи даних; спостерігається зростання інтересу до неанотованих даних. Сфери застосування розширилися до областей генерування тексту на основі таблиць та графів знань, синтезу анотацій та машинного перекладу. У галузевому плані виділяється генерування медичних текстів. Хоча англійська мова продовжує домінувати, але спостерігається зростання досліджень для низькоресурсних мов, зокрема до німецької та китайської. Огляд також висвітлює актуальні виклики в цій галузі, зокрема адаптацію моделей для низькоресурсних мов, генерування тексту за умов обмеженості навчальних даних та етичні аспекти використання потужних мовних моделей. Автори підкреслюють важливість розробки більш ефективних та інтерпретовних архітектур, вдосконалення методів контрольованого генерування тексту та створення нових оцінювальних метрик. Результати дослідження підкреслюють швидку еволюцію методів нейронного генерування тексту, розширення сфер його застосування. В огляді також окреслено перспективні напрями для майбутніх досліджень з урахуванням актуальних викликів та етичних принципів.

DOI: https://doi.org/10.31558/2786-9482.2024.2.4

Вступ

Опрацювання природної мови (Natural Language Processing, NLP) – міждисциплінарна галузь інформатики та лінгвістики [1], класифікацію основних задач якої подано на рис. 1.

Рисунок 1. Таксономія популярних задач NLP для генерації тексту (на основі [1])

Генерування тексту – розділ NLP, що поєднує обчислювальну лінгвістику та штучний інтелект для синтезу текстів [2]. Найбільш відомою практичною реалізацією цього розділу NLP є ChatGPT – чат-бот на основі моделі GPT, який представлено OpenAI у 2022 р. [3].

Попередній огляд [2] охоплював 90 джерел із 2015 до 2021 р. Надання користувачам доступу до великих мовних моделей у 2022–2023 рр. [4] призвело до зростання інтересу до них (рис. 2), тому виникла потреба у доповненні попереднього огляду.

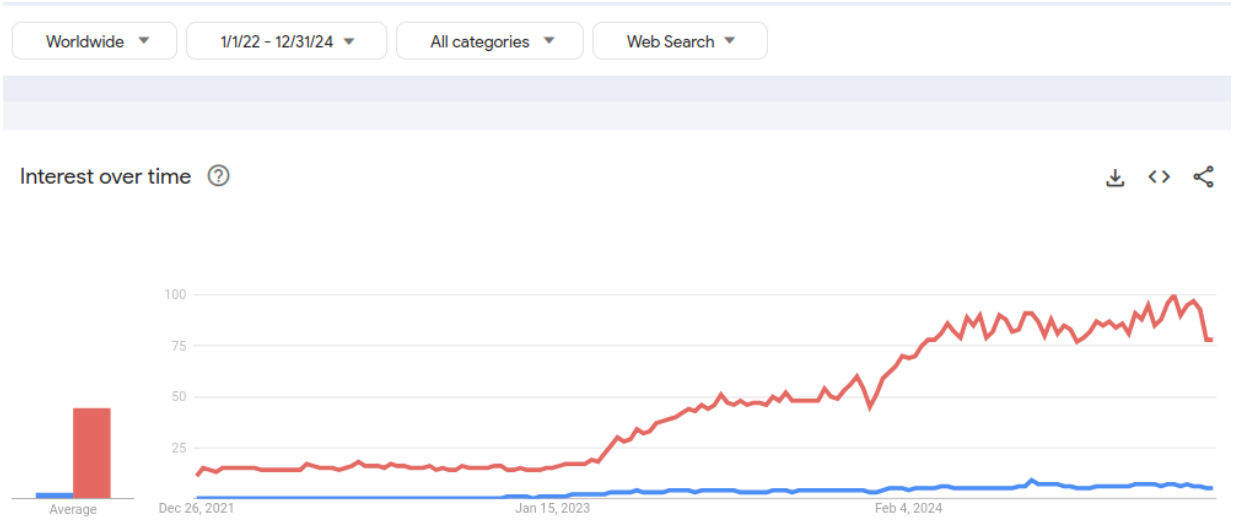

Рисунок 2. Динаміка запитів за пошуковим виразом “large language models” [4]

Основним результатом огляду [2] є класифікація (рис. 3) за п’ятьма ознаками:

1) *за архітектурою нейронної мережі*:

- традиційні:

  RNN – рекурентна нейронна мережа, що використовується для послідовних даних;

  LSTM – мережа з довгою короткочасною пам’яттю, що працює краще за RNN за більших об’ємів даних;

  GRU – вентильний рекурентний вузол (спрощена версія LSTM);

  CNN – згорткова нейронна мережа;

- інноваційні:

  Attention Based – мережі, що використовують механізм уваги для підвищення значущості вхідних даних;

  Transformer – мережі, що використовують механізм уваги без рекурентних або згорткових шарів;

  BERT – розроблена Google нейронна мережа, що поєднує механізми уваги без рекурентних або згорткових шарів із двонапрямленими кодувальниками;

2) *за метриками якості*:

- людино-орієнтовані – Domain-Expert, які залучають фахівців у предметній області для валідації результатів;
- машинно-орієнтовані (автоматичні):

  BLEU (bilingual evaluation understudy) – порівнює кількість і значення токенів (лексем) машинного і людського перекладів без врахування значень слів;

  ROUGE (Recall-Oriented Understudy for Gisting Evaluation) – порівнює анотації та переклади, які згенеровано машиною та людиною;

  Cosine Similartity – порівнянє косинус кута двох ненульових векторів;

  Content Selection – схожа з ROUGE метрика, яка використовує механізм уваги щодо аналізованої задачі;

  Diversity Score – метрика оцінювання різноманіття;

3) *за застосуванням нейронної мережі*:

  AMR (Abstract Meaning Representation) – видобування семантичних співвідношень із тексту;

  Language Generation – генерування тексту, подібного до людського;

  Speech-to-text – перетворення усної промови у текст;

  Script Generation – генерування сценаріїв на основі заданих слів;

  Machine Translation – машинний переклад тексту з однієї мови на іншу;

  Text Summarization – генерування анотації тексту;

  Image Captioning – генерування опису за зображенням;

  Shopping Guide – генерування рекламного опису за зображенням товару;

  Weather Forecast – генерування текстового прогнозу погоди;

4) *за мовою тексту*:

- з тих, що гарно забезпечені ресурсами – англійська, китайська;

- з тих, що недостатньо забезпечені ресурсами – бенгальська, корейська, балійська, іспанська, хінді, словацька, македонська тощо.

5) *за набором даних для навчання нейронної мережі*:

- за розміткою:
  Labeled – розмічені дані;
  Unlabeled – нерозмічені дані;
- за типом:
  Sentence – речення;
  Paragraph – абзац;
  Question / answer – дані типу питання та відповідь;
  Document – дані у вигляді документа.

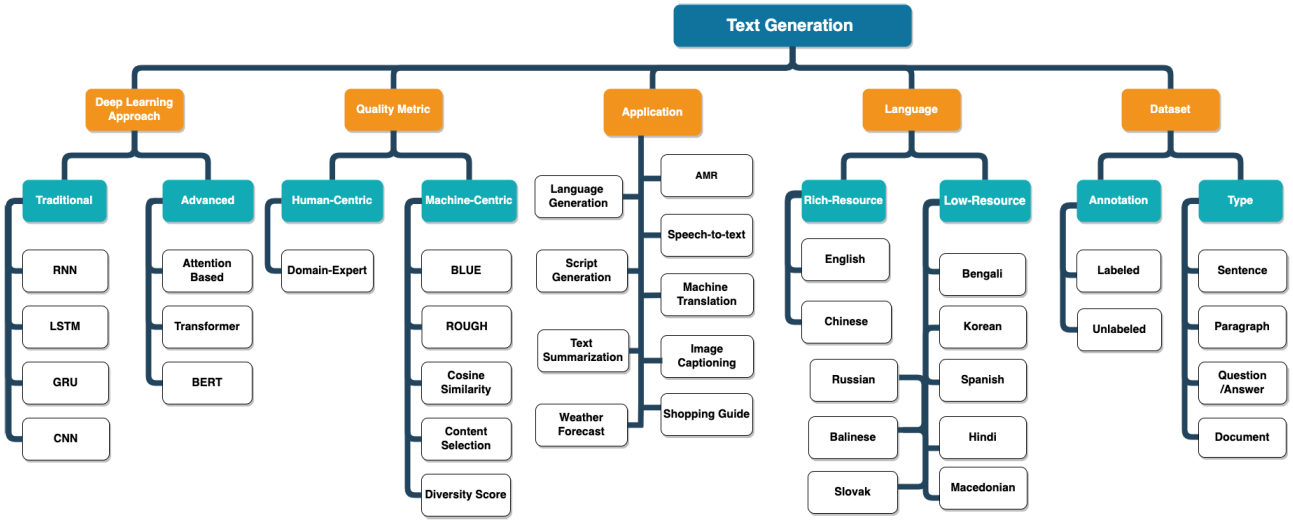

Рисунок 3. Класифікація процесів генерації тексту [2]

Класифікація з рис. 3 є результатом опрацювання таких п’яти задач [2]:

1. Дослідити традиційні та інноваційні методи та підходи глибокого навчання для генерування тексту.
2. Розглянути метрики продуктивності для оцінювання моделей генерування тексту.
3. Дослідити методи оцінювання для вимірювання якості згенерованого тексту.
4. Оглянути нові галузі застосування методів генерування текстового контенту.
5. Обговорити найважливіші проблеми та майбутні напрями дослідження у галузі генерування текстового контенту.

Метою статті є доповнення отриманих у [2] результатів шляхом вирішення таких 5 *дослідницьких питань*:

- які передові методи глибокого навчання використовуються для генерування тексту в літературі 2022–2024 рр.;
- за якими новими метриками оцінюють ефективність моделей генерування тексту в літературі 2022–2024 рр.;
- які набори даних для генерування тексту описано в літературі 2022–2024 рр.;

- які нові застосування методів генерування тексту описано в літературі 2022–2024 рр.;
- які природні мови використовуються для генерування тексту згідно до літератури 2022–2024 рр.

Методика дослідження

Систематичний аналіз літератури є основним **методом цього дослідження,** який дає змогу узагальнити інформацію з великої кількості наукових публікацій (вторинних джерел) за чітко визначеною методикою. Для проведення огляду було обрано методику PRISMA (Preferred Reporting Items for Systematic Reviews and Meta-Analyses), яка є загальновизнаним стандартом для систематичних оглядів та метааналізу у різних галузях науки [5]. Систематичний аналіз за методикою PRISMA передбачає чітке планування дослідження, визначення критеріїв відбору публікацій, проведення ретельного пошуку літератури у провідних наукових базах даних, відбір релевантних досліджень, видобування та синтез даних. Такий підхід забезпечує повноту, надійність і відтворюваність отриманих результатів, що повністю відповідає меті та завданням дослідження.

Джерела інформації та стратегія пошуку

У попередньому огляді [2] як джерела інформації використовувалися наукометричні бази Web of Science та Scopus та бібліотеки IEEE Xplore, SpringerLink, ScienceDirect та ACM Digital Library. Пошуковий запит за назвами статей, анотаціями та ключовими словами, які використано у [2], подано у табл. 1.

Таблиця 1. Формування пошукового запиту в [2]

Назва	Зміст
Група 1: Слова, які стосуються глибокого навчання	deep learning OR natural language processing OR NLP OR neural network OR RNN OR recurrent OR recursive OR LSTM OR GAN OR GPT-2 OR generative adversarial network
Група 2: Слова, які стосуються генерування тексту	text generation OR language generation OR language modelling OR natural language generation OR neural language generation
Пошуковий запит	(Група 1) AND (Група 2)

Наразі Scopus покриває приблизно 90% контенту IEEE Xplore та ACM Digital Library, а Web of Science – приблизно 50%. ScienceDirect та Scopus мають одного й того ж власника. Ураховуючи, що до Scopus входить значна частина вказаних бібліотек, замість 2 баз та 4 бібліотек дослідження проведемо лише за базою Scopus. За пошуковим запитом із попереднього огляду (табл. 1) видача становить 2 580 документів за 2015–2020 рр. проти 100 документів, вказаних у [2]. Якщо шукати виключно за назвами статей, кількість документів зменшується до 109 і спостерігається часткове співпадіння з переліком джерел із [2].

Неможливість відтворення попередніх результатів за запитом із табл. 1 спонукало до створення такого нового запиту:

```
(
    TITLE-ABS-KEY(neural network)
    OR
    TITLE-ABS-KEY(machine learning)
    OR
    TITLE-ABS-KEY(deep learning)
)
AND
TITLE("text generation")
```

Перша частина запиту спрощена до трьох ключових фраз, дві з яких (“neural network” та “deep learning”‘) співпадають із першою групою табл. 1, а третя (“machine learning”) узагальнює усі інші ключові слова першої групи, включно з неіснуючими на момент створення попереднього огляду. До другої частини запиту включена лише ключова фраза “text generation”, пошук якої виконується у заголовках документів, а не в заголовках, анотаціях та авторських ключових словах.

Критерії відбору документів

Для аналізу відбираються лише ті документи, які одночасно задовольняють такі 3 умови:

- опубліковані протягом 2022–2024 рр.;
- стосуються генерування тексту за допомогою штучних нейронних мереж;
- описують підходи, архітектури, метрики якості, мови, набори даних або застосування згенерованого тексту.

Документи виключаються, якщо вони задовольняють хоча б одну з таким умов:

- опубліковані до 2022 року або такі, що не містять даних за 2022–2024 рр.
- не стосуються генерування тексту або не використовують штучні нейронні мережі.
- не містять релевантної інформації щодо поставлених дослідницьких питань (нові методи, метрики, набори даних, застосування, природні мови).

Процес відбору документів

Запит до Scopus 04.03.2024 р. повернув 248 документів, розподіл яких за роками подано на рис. 4. Із них 2 виявились дублікатами, а 157 – датованими раніше 2022 р., тому вони були виключені.

На рис. 5 подана схема відбору даних для систематичного огляду. 89 документів отримано із сайтів видавців, із наукової соціальної мережі ResearchGate та серверів препринтів (насамперед arXiv). 41 документ (насамперед з ACM Digital Library та IEEE Xplore) отримати не вдалось. Отже, для оцінювання відібрано 48 документів, перегляд яких виявив 1 документ, що не містив дані за 2022–2024 рр., та 4 документи, що не містили релевантної інформації щодо поставлених дослідницьких питань. На загал, такі 43 документи відібрано для аналізу: [6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31, 32, 33, 34, 35, 36, 37, 38, 39, 40, 41, 42, 43, 44, 45, 46, 47, 48].

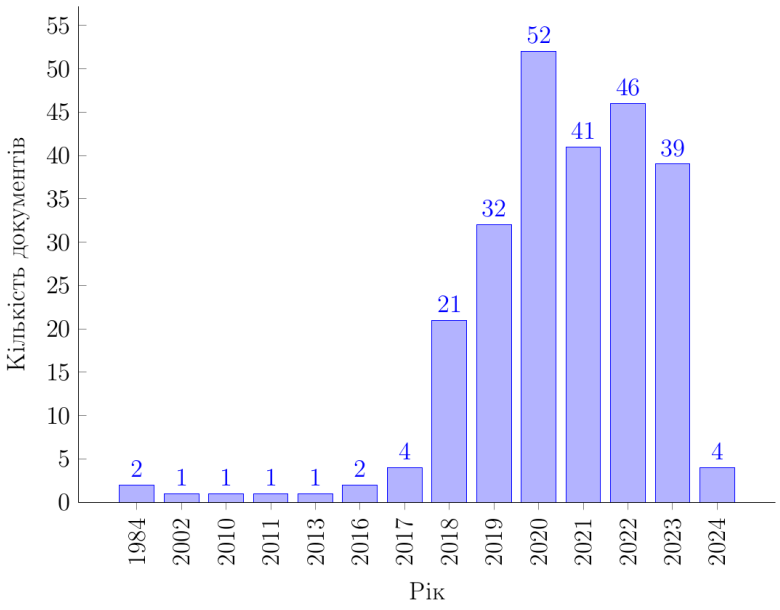

Рисунок 4. Розподіл результатів пошуку за роками

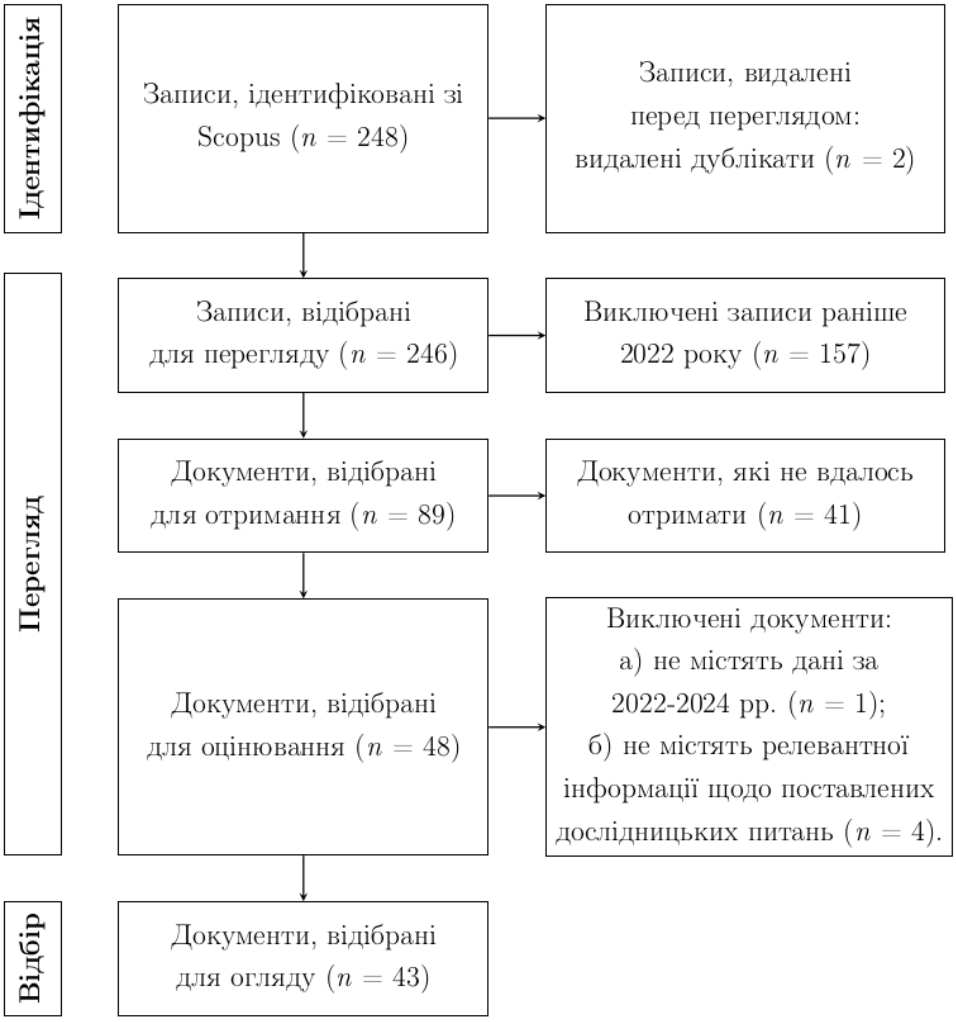

Рисунок 5. Схема відбору даних для систематичного огляду (згідно з методикою PRISMA [5])

Аналіз кожного документа виконувався відповідно до карти огляду за допомогою великої мовної моделі Claude 3 Sonnet [49]. На вхід моделі подавався PDF-файл документа з промптом, який відповідає карті огляди. Промт є таким:

```
Опиши статтю за такими характеристиками:
Тип документу: стаття у журналі (ARTICLE) або стаття у матеріалах
конференції (CONFERENCE)
Назва
Рік публікації
Країни, які представляють автори
Мета статті
Використані архітектури нейронних мереж
Використані метрики якості
Характеристики використаних наборів даних - ім’я
Характеристики використаних наборів даних - тип даних: речення, абзац,
документ, питання-відповідь, не вказаний
Характеристики використаних наборів даних - розмір
Характеристики використаних наборів даних - формат: CSV, JSON, XML,
файли, не вказаний
Характеристики використаних наборів даних - за типом розмітки: розмічені
дані, нерозмічені дані
Характеристики використаних наборів даних - якість даних: попередньо не
опрацьовані (сирі), попередньо опрацьовані
Характеристики використаних наборів даних - за доступності:
загальнодоступний, приватний, не вказаний
Характеристики використаних наборів даних - посилання
Розв’язана задача генерації тексту (для чого застосовувалась нейронна
мережа)
Мова генерації тексту
```

Приклад відповіді подано на рис. 6. Для зовнішньої перевірки застосовано велику мовну модель GPT 4 через розширення Scholar GPT [50]. Заповнені карти огляду для кожної статті наведено у [51]. Для прикладу в табл. 2 подано порівняння результатів автоматичного аналізу статті [47].

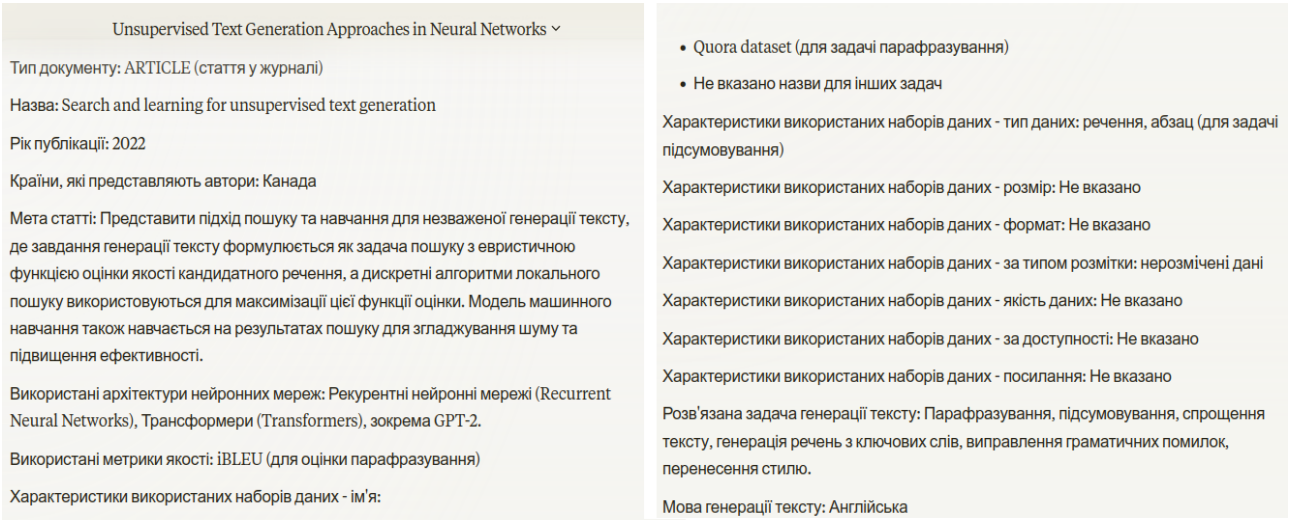

Рисунок 6. Відповідь Claude 3 Sonnet на запит щодо складання карти огляду статті [47]

Таблиця 2. Порівняння результатів роботи великих мовних моделей Claude 3 Sonnet та GPT 4

Запит	**Відповідь Claude 3 Sonnet**	**Відповідь GPT 4**
Тип документу: стаття у журналі (ARTICLE) або стаття у матеріалах конференції (CONFERENCE)	ARTICLE (стаття у журналі)	Article in a journal
Назва	Search and learning for unsupervised text generation	Search and Learning for Unsupervised Text Generation
Рік публікації	2022	2022
Країни, які представляють автори	Канада	Canada (The author, Lili Mou, is affiliated with the University of Alberta in Edmonton, Canada)
Мета статті	Представити підхід пошуку та навчання для незваженої генерації тексту, де завдання генерації тексту формулюється як задача пошуку з евристичною функцією оцінки якості кандидатного речення, а дискретні алгоритми локального пошуку використовуються для максимізації цієї функції оцінки. Модель машинного навчання також навчається на результатах пошуку для згладжування шуму та підвищення ефективності	The article aims to introduce approaches to unsupervised text generation using search and learning methodologies. It focuses on overcoming the need for massive labeled datasets, which are required in supervised methods, thereby making text generation feasible for low-resource languages and reducing the human labor required for data annotation
Використані архітектури нейронних мереж	Рекурентні нейронні мережі (Recurrent Neural Networks), Трансформери (Transformers), зокрема GPT-2	Neural architectures mentioned include recurrent neural networks (RNNs), Transformer architecture, and language models such as RoBERTa and pretrained language models by Radford et al., 2019
Використані метрики якості	iBLEU (для оцінки парафразування)	Metrics such as language fluency, semantic coherency, and task-specific constraints (like sentence simplicity scores) are used to evaluate the quality of generated texts
Характеристики наборів даних		
- ім’я	Quora dataset (для задачі парафразування). Не вказано назви для інших задач	Not specified in detail for individual datasets
- тип даних: речення, абзац, документ, питання-відповідь, не вказаний	речення, абзац (для задачі підсумовування)	Sentences, as the text generation involves tasks like paraphrasing, summarization, etc
- розмір	Не вказано	Specific sizes are not mentioned, but the use of large-scale generic corpora is implied
- формат: CSV, JSON, XML, файли, не вказаний	Не вказано	Not explicitly mentioned, but it includes labeled input-output pairs in various formats typical for NLP tasks

Запит	Відповідь Claude 3 Sonnet	Відповідь GPT 4
- за типом розмітки: розмічені або нерозмічені дані	нерозмічені дані	Primarily deals with labeled data in supervised contexts, though the focus here is on reducing dependence on such datasets
- якість даних: не опрацьовані (сирі), попередньо опрацьовані	Не вказано	The discussion includes the use of both raw and preprocessed data in different contexts
- загальнодоступний, приватний, не вказаний	Не вказано	The text suggests a shift towards methods that do not require publicly available or parallel datasets
- посилання	Не вказано	Not specific dataset URLs provided
Розв'язана задача генерації тексту (для чого застосовувалась нейронна мережа)	Парафразування, підсумовування, спрощення тексту, генерація речень з ключових слів, виправлення граматичних помилок, перенесення стилю	The neural networks are applied to generate text for various applications including paraphrasing, summarization, text simplification, and creating sentences from keywords, all in an unsupervised manner
Мова генерації тексту	Англійська	The tasks are primarily demonstrated with English examples, though the methodologies are applicable to multiple languages

Порівняння опису статті [47], виконаного за допомогою іншої великої мовної моделі та перевіреною людиною, з результатами з табл. 2 показує, що опис добре узгоджується з результатами роботи як Claude 3 Sonnet, так і GPT-4. Обидві моделі точно визначили тип документа, назву, рік публікації, країни авторів, мету статті, використані архітектури нейронних мереж, метрики якості та розв'язані задачі генерування тексту. Щодо характеристик наборів даних, обидві моделі вказали, що деталі про конкретні набори даних не надаються, за винятком набору даних Quora для парафразування. Вони також зазначили, що стаття зосереджується на зменшенні залежності від розмічених або публічно доступних наборів даних, хоча в різних контекстах обговорюються як розмічені, так і нерозмічені дані.

Отже, великі мовні моделі можуть точно видобувати ключову інформацію зі статей, хоча іноді упускають деталі, які явно не вказані в тексті. Для мінімізації ризику таких помилок виконано перевірку людиною результатів роботи Claude 3 Sonnet. Задля уникнення проблем, пов'язаних із перекладом термінології, відповіді великої мовної моделі додатково затребувано мовою відібраних документів, а саме англійською.

Оцінювання якості

Для оцінювання якості процесу видобування ключової інформацію зі статей застосуємо такі критерії:

- чіткість і відповідність критеріїв включення та виключення досліджень меті огляду;
- повнота та систематичність пошуку релевантних досліджень в обраних базах даних;
- послідовність і відтворюваність процесу відбору досліджень згідно з критеріями включення та виключення;
- застосування стандартизованої картки огляду для збору та систематизації даних із відібраних досліджень;

- залучення щонайменше двох незалежних дослідників до процесу відбору, аналізу та синтезу даних для мінімізації ризику упередженості;
- врахування та опис будь-яких розбіжностей або невизначеностей у процесі відбору та аналізу досліджень;
- забезпечення прозорості та відтворюваності процесу огляду шляхом детального опису кожного етапу у звіті.

Дотримання зазначених критеріїв якості забезпечує надійність і обґрунтованість результатів та висновків цього систематичного огляду.

PRISMA передбачає наявність у методиці дослідження таких додаткових компонентів:

- *оцінка ризику упередженості у відібраних дослідженнях* не є релевантною через те, що у цьому огляді розглядаються різні підходи та методи генерації тексту, а не порівнюються результати окремих досліджень;
- *визначення міри ефекту для кожного результату (або типу результату)* не виконується через те, що цей огляд не має на меті здійснення метааналізу чи кількісного синтезу результатів;
- *опис методів синтезу результатів досліджень*, як-от метааналіз, не виконується через те, що огляд не передбачає кількісного синтезу результатів;
- *оцінка ризику упередженості через неповноту подання результатів у публікаціях* не наводиться через те, що цей огляд фокусується на описі та класифікації описаних підходів і методів;
- *оцінювання достовірності та надійності результатів*, отриманих із публікацій, не здійснюється через використання надійних джерел, а саме видань з бази Scopus.

Розподіл відібраних документів за роками

Протягом 2022–2024 рр. кількість журнальних статей (ARTICLE) майже дорівнює кількості матеріалів конференцій (CONFERENCE) – 22 проти 21 (рис. 7). У 2022 р. кількість матеріалів конференції (15) значно перевищувала кількість статей у журналах (4), проте з 2023 р. збільшилась кількість статей у журналах (16), порівняно з матеріалами конференцій (6). За січень та лютий 2024 р. наявні лише статті у журналах (2), а матеріали конференцій відсутні. Така динаміка останніх років може свідчити про більш ґрунтовне висвітлення проблематики у наукових журналах, порівняно з матеріалами конференцій.

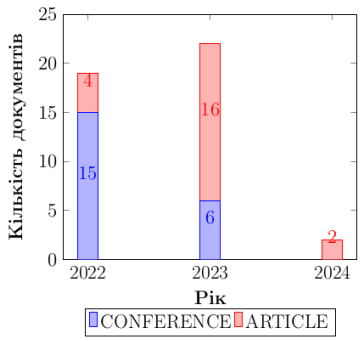

Рисунок 7. Розподіл статей за типом видання

Які передові методи глибокого навчання використовуються для генерування тексту в літературі 2022–2024 рр.?

Табл. 3 представляє огляд архітектур нейронних мереж, що використовуються для генерування тексту, згідно з даними публікацій 2022–2024 рр. Табл. 4 представляє узагальнення підходів до генерації тексту на основі даних табл. 3.

Таблиця 3. Архітектури нейронних мереж для генерації тексту

Архітектура	Опис	Представники	Статті
Традиційні підходи			
RNN (Recurrent Neural Networks)	Рекурентні нейронні мережі, що використовуються для обробки послідовних даних	–	[6, 9, 10, 11, 29, 30, 47]
LSTM (Long Short-Term Memory)	Варіант RNN, що краще запам'ятовує довгострокові залежності	–	[6, 10, 11, 13, 14, 15, 29, 30, 33, 41, 42]
GRU (Gated Recurrent Unit)	Спрощений варіант LSTM з меншою кількістю параметрів	–	–
CNN (Convolutional Neural Networks)	Згорткові нейронні мережі, що часто використовуються для обробки зображень	YOLOv5	[6, 9, 16, 38]
Graph Neural Networks	Моделі, що працюють із графовими структурами даних	GraphWriter, CGE-LW	[7, 9]
Інноваційні підходи			
Autoencoders	Мережі, які використовують для навчання ефективних кодувань нерозмічених даних	AE, VAE, iVAE, clVAE+ MI, β0.4 VAE, SaVAE, LagVAE	[15, 17, 29]
Transformer	Архітектура, що використовує механізм уваги для обробки послідовних даних	T5, CodeT5, TrICY, DETR	[7, 8, 9, 18, 19, 20, 22, 27, 31, 32, 34, 39, 41, 43, 44, 47, 48]
BERT (Bidirectional Encoder Representations from Transformers)	Модель на основі Transformer, що навчається на великих обсягах нерозміченого тексту	PubmedBERT, BioLinkBERT, RoBERTa, XLM-RoBERTa	[8, 9, 12, 13, 18, 19, 20, 26, 28, 30, 32, 35, 37, 39, 40, 45]
GPT-2, GPT-3 (Generative Pre-trained Transformer)	Моделі на основі Transformer, що використовуються для генерації тексту	OPT, Llama, CodeBERT	[6, 8, 10, 11, 12, 13, 15, 18, 19, 21, 22, 23, 24, 25, 26, 32, 33, 34, 36, 37, 39, 43, 44, 45, 47]
Attention-based models	Моделі, що використовують механізм уваги для покращення якості генерованого тексту	–	[8, 20, 26, 43, 44, 47]
Seq2Seq (Sequence-to-Sequence)	Архітектура, що використовує кодувальник та декодувальник для генерування послідовностей	S2ST, S2SL, S2SG, S2ST+, D+ Full, DSG	[15, 28, 31, 39, 42, 43, 46]
GAN (Generative Adversarial Networks)	Генеративно-змагальні мережі, що складаються з генератора та дискримінатора	EGAN, TILGAN, DoubAN-Full, WRGAN, CatGAN, SeqGAN, DGSAN	[6, 25, 29]

Архітектура	Опис	Представники	Статті
Memory Networks	Моделі, що використовують зовнішню пам'ять для зберігання та доступу до інформації	DM-NLG (with memory), MemNNs, Mem2Seq, GLMP	[9, 34]
Diffusion Models	Моделі, що використовують дифузійний процес для генерування тексту	GENIE, NAT, iNAT, ELMER, MASS, CMLM, ProphetNet, InsT, LevT, BANG, ConstLeven	[41]
Prompt-based models	Моделі, що використовують prompt-engineering донавчання для управління генеруванням тексту	–	[23]

Таблиця 4. Підходи до генерування тексту

Категорія	Статті
Традиційні підходи	[14, 16, 38]
Інноваційні підходи	[8, 12, 18, 19, 20, 21, 22, 23, 24, 25 26, 27, 28, 31, 32, 34, 35, 36, 37, 39, 40, 43, 44, 45, 46, 48]
Комбінація традиційних та інноваційних підходів	[6, 7, 9, 10, 11, 13, 15, 29, 30, 33, 41, 42, 47]

Серед інноваційних підходів найбільш популярним є використання моделей на основі архітектури Transformer, зокрема GPT-2, GPT-3, BERT та їх варіацій. Вони демонструють високу ефективність генерування зв'язного та семантично релевантного тексту. Також набувають популярності підходи з використанням механізмів уваги (attention) та контрольованого генерування тексту (controllable text generation). Традиційні підходи, хоча і використовуються рідше, все ще знаходять своє застосування у певних задачах, як-от генерування тексту на основі зображень, машинний переклад та інші. Спостерігається тенденція до переходу від традиційних підходів до більш інноваційних та ефективних моделей на основі архітектури Transformer та механізмів уваги. Це дає змогу покращити якість генерованого тексту та розширити сферу застосування цих технологій. Рис. 8 показує, що у 2022–2023 рр. переважають інноваційні підходи до генерування тексту. У 2024 р. кількість статей з інноваційним та комбінованим підходами однакова, проте вибірка за цей рік є неповною.

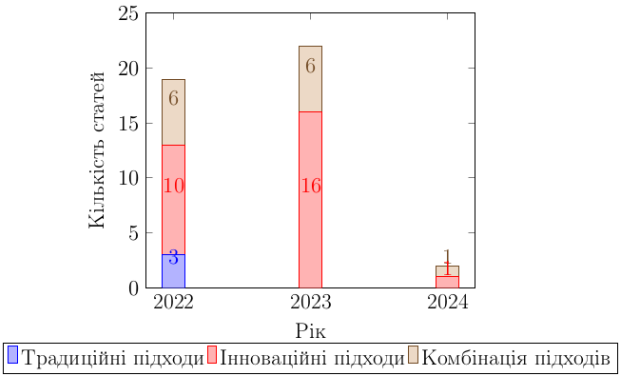

Рисунок 8. Розподіл статей за категоріями підходів до генерації тексту

Порівнюючи отримані результати з даними попереднього систематичного огляду [2], робимо такі висновки:

- традиційні підходи, як-от RNN, LSTM, CNN, все ще використовуються для генерування тексту, але меншою мірою, порівняно з інноваційними підходами;
- архітектура Transformer та її варіанти GPT-2, GPT-3 та BERT набули значної популярності у 2022–2024 рр., демонструючи високу ефективність генерування зв'язного семантично релевантного тексту;
- з'явилися нові архітектури та підходи, як-от Diffusion Models та Memory Networks models;
- значна увага приділяється моделям, що використовують механізми уваги (Attention-based models) та контрольованого генерування тексту (Controllable Text Generation);
- спостерігається тенденція до комбінування традиційних та інноваційних підходів для досягнення кращих результатів;
- у 2022–2024 рр. спостерігається перехід від традиційних підходів до більш інноваційних та ефективних моделей на основі архітектури Transformer та механізмів уваги, що дає змогу покращити якість генерованого тексту та розширити сферу застосування цих технологій.

За якими новими метриками оцінюють ефективність моделей генерування тексту в літературі 2022–2024 рр.?

Табл. 5 представляє огляд метрик якості, що використовуються для оцінювання згенерованого тексту. Метрики розділені на дві категорії: human-centred (орієнтовані на людину) та machine-centred (орієнтовані на машину). До human-centred метрик належать Human Evaluation та Turing Test, які передбачають оцінювання якості згенерованого тексту людьми-експертами або тест на здатність моделі генерувати текст, схожий на написаний людиною. Machine-centred метрики включають широкий спектр автоматичних метрик, як-от BLEU, ROUGE, METEOR, Perplexity, Distinct-n, BERTScore тощо. Ці метрики оцінюють різні аспекти якості згенерованого тексту – схожість з еталонним текстом, плавність, змістовність, різноманітність лексики та синтаксису тощо.

Табл. 6 надає огляд застосованих у статтях метрик якості. Більшість досліджень використовують machine-centred метрики для автоматичного оцінювання якості згенерованого тексту. Значно менша кількість досліджень застосовує human-centred метрики, що може бути пов'язано з трудомісткістю та суб'єктивністю оцінювання. Проте використання human-centred метрик все ще залишається важливим для отримання більш повної та надійної оцінки якості генерації тексту. Деякі дослідження не застосовують жодних метрик якості, що може бути пов'язано з фокусом на інших аспектах генерування тексту, як-от ефективність чи швидкість роботи моделей.

Використання різноманітних метрик якості є важливим для всебічної оцінки ефективності моделей та підходів до генерації тексту. Комбінування machine-centred та human-centred метрик дає змогу отримати більш надійні та валідні результати оцінювання. Діаграма на рис. 9 показує, що найчастіше використовується метрики BLEU (55.8% статей)

та ROUGE (48.8% статей). Також доволі поширеним є оцінювання якості людьми (Human Evaluation) – вона застосовується у 23.3% статей. Інші метрики, як-от Perplexity, METEOR, BERTScore та Distinct-n, використовуються рідше, але все ще мають значну частку згадувань у статтях. Найменш поширеними є метрики Turing Test, Fluency, Coherence, Diversity, N-gram Overlap та Embedding Similarity, кожна з яких згадується лише в одній статті (2.3%).

Таблиця 5. Основні метрики якості згенерованого тексту

Метрика якості	**Опис**	**Представники**	**Статті**
Human-centred metrics			
Human Evaluation	Оцінювання якості згенерованого тексту людьми-експертами	–	[9, 10, 11, 25, 30, 31, 32, 33, 36, 37]
Turing Test	Тест на здатність моделі генерувати текст, який неможливо відрізнити від написаного людиною	–	[33]
Machine-centred metrics			
BLEU	Метрика, що оцінює якість згенерованого тексту шляхом порівняння його з еталонним текстом	BLEU-1, BLEU-2, BLEU-3, BLEU-4, BLEU-5	[7, 8, 9, 10, 13, 15, 18, 19, 23, 27, 29, 31, 32, 33, 34, 36, 37, 41, 42, 43, 44, 45, 46, 47]
ROUGE	Метрика, що оцінює якість автоматичного реферування тексту	ROUGE-1, ROUGE-2, ROUGE-3, ROUGE-L	[7, 8, 9, 10, 13, 18, 19, 23, 27, 28, 33, 34, 35, 36, 41, 42, 43, 44, 45, 46, 48]
METEOR	Метрика, що оцінює якість машинного перекладу	–	[18, 27, 32, 34, 36, 42, 43, 44, 46, 48]
BERTScore	Метрика, що оцінює якість згенерованого тексту з використанням попередньо навченої моделі BERT	–	[8, 13, 18, 19, 26, 34, 32, 37]
CIDEr	Метрика, що оцінює якість автоматичного опису зображень, порівнюючи згенеровані описи з наборами референсних описів	–	[14, 18, 23, 36, 37, 41, 42, 46]
Perplexity	Метрика, що оцінює якість мовної моделі	–	[8, 9, 15, 17, 26, 29, 36, 39]
F1-score	Метрика, що оцінює якість класифікації, зокрема в задачах двокласової класифікації	–	[13, 20, 21, 26, 34, 40]
CHRF++	Метрика, що оцінює якість машинного перекладу, базуючись на збігах символів та n-грам	–	[7, 32, 37, 48]
Distinct-n	Метрика, що оцінює різноманітність згенерованого тексту	Dist-1, Dist-2, Dist-3, Dist-4	[8, 9, 15]

Таблиця 6. Огляд застосованих у статтях метрик якості

Метрики якості	**Статті**
Machine-centred	[7, 8, 11, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 26, 27, 28, 29, 34, 35, 38, 39, 40, 41, 42, 43, 44, 45, 46, 47, 48]
Human-centred	[11, 30]
Обидві	[9, 10, 25, 32, 33, 36, 31, 37]
Не застосовано	[24, 12, 6]

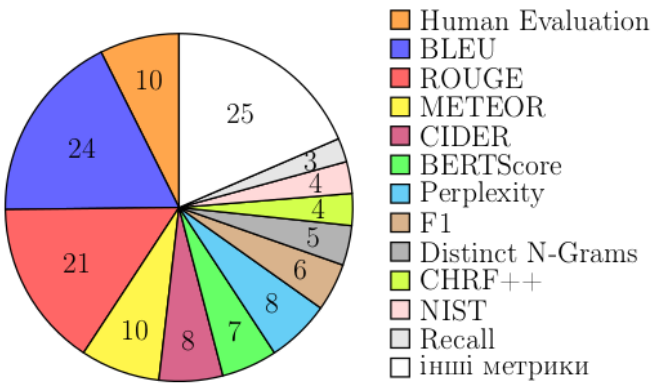

Рисунок 9. Розподіл метрик якості за кількістю статей, у яких вони згадані

Автоматичні метрики якості, як-от BLEU та ROUGE, найчастіше використовуються для оцінювання ефективності моделей генерування тексту, тоді як оцінювання якості людьми застосовується рідше, але залишається важливим компонентом для більш повної та надійної оцінки якості згенерованого тексту.

Порівнюючи отримані результати з даними попереднього систематичного огляду [2], можна виділити такі спостереження:

- BLEU та ROUGE залишаються найпопулярнішими метриками оцінювання якості згенерованого тексту як у 2015–2021 рр., так і у 2022–2024 рр.;
- Human Evaluation все ще широко застосовується для отримання більш повної та надійної оцінки якості генерації тексту, незважаючи на трудомісткість і суб'єктивність такого підходу;
- у 2022–2024 рр. з'явились нові метрики – BERTScore, Fluency, Coherence, Diversity, N-gram Overlap та Embedding Similarity, що засвідчує активний розвиток методів оцінювання якості згенерованого тексту та пошук більш ефективних та інформативних метрик;
- перплексія (Perplexity) набула більшої популярності у 2022–2024 рр., порівняно з попереднім періодом, що може бути пов'язано з її ефективністю в оцінюванні якості мовних моделей.
- метрика METEOR, за якою оцінюють якість машинного перекладу, також частіше використовується у 2022–2024 рр., що може свідчити про зростання інтересу до застосування генерування тексту в задачах машинного перекладу;
- загалом спостерігається тенденція до комбінування різних типів метрик (machine-centred та human-centred) для отримання більш надійних та валідних результатів оцінювання ефективності моделей генерації тексту.

Отже, порівняння результатів двох оглядів демонструє, що хоча традиційні метрики, як-от BLEU та ROUGE, залишаються популярними, у 2022–2024 рр. з'явилися нові метрики, які враховують різні аспекти якості згенерованого тексту. Це свідчить про активний розвиток методів оцінки якості й пошук більш ефективних та інформативних підходів до оцінювання моделей генерування тексту.

Які набори даних для генерування тексту описано в літературі 2022–2024 рр.?

Табл. 7 представляє набори даних з оглянутих статей. Вони спочатку впорядковані за спаданням кількості згадувань, а потім – за алфавітом. Набір даних E2E згадується найчастіше – у семи статтях, за ним слідують XSum, CNN/DailyMail, CommonGen, ToTTo та WebNLG, які згадуються у чотирьох статтях. У проаналізованих дослідженнях використовується широкий спектр наборів даних, що охоплюють різні домени і типи текстів, від відгуків користувачів та новинних статей до медичних і технічних текстів. Це свідчить про активний розвиток та застосування методів генерування тексту у різноманітних сферах.

Таблиця 7. Набори даних, про які згадано в оглянутих статтях

Назва набору даних	Статті
E2E	[19, 23, 30, 31, 34, 36, 44]
CNN/DailyMail (CNN/DM)	[9, 23, 41, 45]
Totto	[18, 31, 43, 46]
CommonGen	[9, 18, 36, 41]
WebNLG	[7, 31, 37, 44]
XSum	[9, 18, 23, 41]
WikiBio	[18, 31, 34]
Abstract Generation Dataset (AGENDA)	[7, 9]
DDI	[9, 12]
NIST	[9, 27]
PubMed	[12, 23]
Quora	[9, 47]
ROCStories	[9, 36]
Snips	[19, 39]
SST-2	[21, 45]
WMT’14 English-German	[18, 27]
WMT’16 Romanian-English	[18, 27]
Yelp	[17, 40]
Baidu Tieba; PersonaChat; Gigawords; Yahoo! Answers; NLPCC; Tencent; SQuAD; ComVE; αNLG-ART; EntDesc; VisualStory; PaperWriting; Reddit-10M; EMNLP dialog; ICLR dialog; NarrativeQA; Wizard of Wikipedia (WoW); MS-MARCO; ELI5; ChangeMyView; Amazon books; Foursquare	[9]
Scratch online community comments	[11]
BC5-Chemical; BC5-Disease; NCBI-Disease; BC2GM; JNLPBA; EBM PICO; ChemProt; GAD; BIOSSES; HoC; PubMedQA; BioASQ	[12]
Logic2Text	[13]
Concadia	[14]
REDIAL	[15]
Custom dataset for Bangla word sign language	[16]
Synthetic dataset; Penn Treebank	[17]
IWSLT’14 De-En	[18]
WMT16 English-German	[45]
WMT17 English-German	[36]
WMT20; WMT21	[37]
WMT’14 German-English	[27]
Multi-News; Java; Python	[18]

Назва набору даних	Статті
English ATIS; ViGGO; TREC; Korean Weather; Rest; KLUE-TC	[19]
C4; M2D2; Political Slant	[20]
Layoff; MC; M&A; Flood; Wildfire; Boston Bombings; Bohol Earthquake; West Texas Explosion; Dublin; New York City	[21]
WSC; CBT-CN; CBT-NE	[22]
Wikihow; SAMSum; DART	[23]
Custom dataset composed of tweets labeled with emotions	[25]
AFQMC; CHIP-STS; QQP; MRPC	[26]
ParaNMT-small; NIST Chinese-English	[27]
GTZAN	[28]
Minions; Japanimation; WikiArt; Nottingham; Lakh MIDI; TheoryTab; Poem-5; Poem-7	[29]
Synthetic date generation dataset	[30]
LDC2020T02 (AMR 3.0 release)	[32]
One Million Urdu News Dataset; Australian Broadcasting Corporation (ABC) news dataset	[33]
DailyMed drug labels	[35]
COCO Image Captioning	[37]
German and French commercial datasets; MASSIVE	[39]
Gold-PMB; Silver-PMB	[42]
numericNLG	[43]
Custom dataset related to text messaging applications	[44]
TweetEval; AGnews; QNLI; IMDB; CC-News	[45]
WITA	[46]
XWIKIREF	[48]

Табл. 8 представляє типи даних, які використані в оглянутих статтях. Типи даних впорядковано за спаданням кількості згадувань. Найчастіше використовуються набори даних, що містять речення – вони згадуються у 26 статтях. Це може бути пов’язано з тим, що багато завдань генерування тексту, як-от машинний переклад, парафразування, відповіді на запитання, вирішуються на рівні речень. Водночас наявність різноманітних типів даних, включно з абзацами, документами, зображеннями, музикою та іншим, свідчить про те, що методи генерування тексту можуть застосовуватись до широкого спектра задач та доменів.

Табл. 9 представляє типи розмітки даних, які використані в оглянутих статтях. Типи розмітки впорядковано за спаданням кількості згадувань. Найчастіше використовуються розмічені набори даних – вони згадуються у 22 статтях. Це може бути пов’язано з тим, що багато завдань генерування тексту, особливо ті, що використовують контрольовані підходи або вимагають відповідності певним шаблонам чи структурам, потребують розмічених даних для навчання моделей. Розмітка може включати такі елементи: частини мови, синтаксичні структури, семантичні ролі, теги для контрольованої генерації тощо.

Водночас наявність досліджень з нерозміченими даними або з комбінацією розмічених та нерозмічених даних свідчить про активний розвиток методів навчання без вчителя та напівавтоматичного навчання в галузі генерування тексту. Ці підходи дають змогу використовувати великі обсяги нерозмічених текстових даних для попереднього навчання моделей та покращення їх здатності до генерування зв’язного та змістовного тексту.

Таблиця 8. Типи даних, які використовуються в оглянутих статтях

Тип даних	Статті
Речення	[7, 9, 11, 12, 14, 15, 17, 18, 19, 20, 22, 23, 25, 26, 29, 30, 31, 32, 33, 36, 39, 40, 41, 45, 46, 48]
Абзац	[7, 9, 12, 15, 17, 18, 19, 20, 23, 29, 37, 14, 39, 40, 41, 45, 46, 48]
Документ	[9, 12, 18, 19, 20, 29, 35, 40, 41, 42, 45]
Питання-відповідь	[7, 9, 12, 17, 18, 19, 22, 26, 45, 47]
Таблиці з описом	[13, 21, 30, 31, 33, 34, 36, 43, 44]
Переклади	[18, 27, 31, 33, 36, 37, 45]
Історії	[9, 31, 33, 36]
Зображення	[14, 29, 37, 38]
Аудіофайли	[29, 28]
Відеокліпи	[16]
Комп’ютерні програми	[18]
Не вказаний	[6, 8, 10, 24]

Таблиця 9. Типи розмітки даних, які використовуються в оглянутих статтях

Тип розмітки	Статті
Розмічені дані	[12, 13, 14, 16, 17, 18, 21, 27, 28, 31, 33, 34, 35, 39, 40, 42, 43, 44, 46, 48, 47, 25]
Нерозмічені дані	[11, 12, 39, 40, 47]
Не вказано	[6, 7, 8, 9, 10, 15, 19, 20, 22, 24, 23, 26, 29, 30, 32, 36, 37, 38, 41, 45]

Табл. 10 представляє рівень якості даних з оглянутих статтях. Попередньо опрацьовані дані зазвичай проходять очищення, нормалізацію, токенізацію, а іноді й додаткову розмітку. Це дає змогу покращити якість і консистентність даних, а також полегшити навчання. Прикладами попередньо опрацьованих даних можуть бути набори даних із корпусів або баз даних, які вже пройшли певну обробку. Сирі дані беруться безпосередньо з реальних джерел, як-от вебсторінки, соціальні мережі, необроблені тексти тощо. Вони можуть містити шум, некоректне форматування, помилки та інші артефакти. Використання сирих даних може бути корисним для навчання моделей, які мають бути стійкими до реальних умов та здатними обробляти неструктуровані дані. Відсутність інформації про рівень якості даних у значній частині статей може свідчити про те, що автори не приділяють достатньої уваги цьому аспекту або вважають його маловажливим. Водночас якість даних є критичним фактором, що впливає на ефективність і узагальнювальність генерування тексту, тому варто приділяти більше уваги опису та аналізу якості даних у майбутніх дослідженнях.

Таблиця 10. Рівень якості наборів даних в оглянутих статтях

Рівень якості даних	Статті
Попередньо опрацьовані	[13, 14, 16, 17, 18, 44, 47, 48, 39, 34, 31, 42]
Сирі	[7, 11, 28, 33, 35, 37, 39, 34, 31, 42]
Не вказано	[6, 8, 9, 10, 12, 15, 20, 19, 21, 22, 24, 25, 23, 26, 27, 29, 30, 32, 36, 38, 40, 41, 43, 45, 46]

Порівнюючи результати огляду 2022–2024 рр. із попереднім оглядом [2], робимо такі висновки:

- у 2022–2024 рр. з'явилися нові набори даних, як-от XWIKIREF, DailyMed, numericNLG, WITA, DIST-ToTTo, що свідчить про активний розвиток ресурсів для дослідження та застосування методів генерування тексту;
- набори даних E2E, WikiBio, ToTTo, CommonGen, CNN/DailyMail та XSum залишаються популярними і широко використовуються в дослідженнях як у 2015–2021 рр., так і в 2022–2024 рр.;
- спостерігається тенденція до використання більш різноманітних типів даних, як-от таблиці з описом, зображення, музика, переклади, питання-відповідь, відеокліпи та комп'ютерні програми, на додачу до традиційних типів, як-от речення, абзаци та документи;
- розмічені дані залишаються найбільш широко використовуваними, але зростає інтерес до нерозмічених даних та комбінації розмічених і нерозмічених даних;
- хоча якість даних є критичним фактором, що впливає на ефективність генерування текстів, у значній частині досліджень 2022–2024 рр. цей аспект не висвітлюється, що може свідчити про необхідність приділяти більше уваги опису та аналізу якості використаних даних у майбутніх дослідженнях.

Отже, порівняння результатів двох оглядів демонструє, що набори даних для генерування тексту продовжують активно розвиватися, охоплюючи нові домени та типи даних. Водночас деякі популярні набори даних залишаються актуальними та широко використовуваними в дослідженнях. Спостерігається тенденція до використання більш різноманітних типів даних та зростання інтересу до нерозмічених даних і комбінованих підходів. Проте опис якості даних все ще потребує більшої уваги.

Які нові застосування генерування тексту описано в літературі 2022–2024 рр.?

Табл. 11 відображає застосування генерування тексту, які виявлено в аналізованих статтях. Найбільш поширеними застосуваннями є генерування анотацій та машинний переклад, за кожним із яких знайдено 8 статей. Аналіз застосувань демонструє широкий спектр можливостей використання генерування тексту у різних галузях, від обробки структурованих даних до створення емоційно забарвлених текстів та перекладу жестової мови в текст. Розвиток нових методів та архітектур нейронних мереж відкриває нові перспективи для подальшого розширення сфер застосування генерації тексту.

Порівнюючи результати огляду 2022–2024 рр. із попереднім оглядом [2], можна виділити такі спостереження:

- машинний переклад (Machine Translation) та генерування анотацій (Text Summarization) набули більшої популярності у 2022–2024 рр., порівняно з попереднім періодом. У 2022–2024 рр. з'явилися публікації з генерування текстів із таблиць та структурованих даних, що може вказувати на зростання інтересу до такої обробки структурованої інформації;

- контрольоване генерування тексту (Controllable Text Generation) також стало більш поширеним, що свідчить про зростання інтересу до управління генеруванням тексту для отримання більш релевантних та якісних результатів;
- генерування медичних текстів (Medical Text Generation) з'явилося як новий окремий напрям, що може бути пов'язано з активним розвитком методів обробки медичних даних та потребою в автоматизації створення медичної документації;
- з'явилися нові застосування, як-от генерування емоційно забарвлених текстів (Emotional Text Generation), генерування енциклопедичних статей (Encyclopedic Text Generation), генерування технічної документації (Technical Documentation Generation) та переклад жестової мови в текст (Sign Language to Text Translation), що свідчить про розширення сфер використання генерування тексту.
- парафразування та доповнення даних є актуальним застосуванням генерування тексту як у 2015–2021 рр., так і в 2022–2024 рр.;
- деякі застосування, які були популярними у попередньому огляді, як-от генерування поезії, діалогові системи, класифікація текстів і тематичне моделювання, у новому огляді не фігурують серед найбільш згадуваних, що може бути пов'язано зі зміною фокусу досліджень та появою нових перспективних напрямів;
- загалом спостерігається зростання різноманітності застосувань у 2022–2024 рр., порівняно з попереднім періодом, що свідчить про активний розвиток досліджень з генерування тексту та розширення можливостей використання генеративних моделей для вирішення прикладних завдань у різних предметних областях.

Таблиця 11. Застосування генерування тексту

Застосування	**Статті**
Генерування анотацій (Text Summarization)	[9, 18, 23, 41, 37, 45, 47, 48]
Машинний переклад (Machine Translation)	[9, 16, 17, 18, 27, 36, 37, 47]
Генерування тексту зі структурованих даних (Data-to-Text Generation)	[8, 31, 34, 44, 46]
Генерування тексту з таблиць (Table-to-Text Generation)	[13, 30, 34, 36, 43]
Парафразування (Paraphrasing)	[9, 27, 39, 47]
Доповнення даних (Data Augmentation)	[8, 21, 40, 42]
Контрольоване генерування тексту (Controllable Text Generation)	[8, 19, 30]
Генерування тексту за зображеням (Image-based Text Generation)	[14, 29, 37]
Генерування тексту з графів знань (Text Generation from Knowledge Graphs)	[7, 9]
Генерування медичних текстів (Medical Text Generation)	[12, 35]
Генерування емоційно забарвленого тексту (Emotional Text Generation)	[11, 25]
Генерування відповідей на запитання (Question Answering)	[9, 15]
Генерування музичних текстів (Music Text Generation)	[28, 29]
Генерування сценаріїв (Script Generation)	[9, 29]
Генерування новинних заголовків (News Headline Generation)	[33]
Генерування технічної документації (Technical Documentation Generation)	[10]
Кібербезпека (Cybersecurity)	[45]
Генерування енциклопедичних статей (Encyclopedic Text Generation)	[48]
Переклад жестової мови в текст (Sign Language to Text Translation)	[16]

Отже, порівняння результатів двох оглядів демонструє, що сфера застосування генерування тексту продовжує активно розширюватися, охоплюючи нові галузі та напрями. Популярність застосувань генерування тексту з таблиць та графів знань, контрольованого генерування тексту та генерування медичних текстів свідчить про зростання інтересу до методів, які дають змогу ефективно обробляти структуровані дані й отримувати більш релевантні та якісні результати. Водночас традиційні застосування, як-от парафразування, генерування анотацій та машинний переклад, залишаються актуальними й широко використовуваними в дослідженнях.

Які природні мови використовуються для генерування тексту відповідно до літератури 2022–2024 рр.?

Табл. 12 представляє щорічну статистику за мовами генерування тексту протягом досліджуваного періоду. Англійська мова є найбільш популярною протягом усіх трьох років. Для генерування англійських текстів використовуються різноманітні архітектури нейронних мереж, включно з Transformer, BERT, GPT-2, GPT-3, RNN, LSTM, CNN, GAN та Seq2Seq.

Таблиця 12. Статистика за мовами генерування

Мова	2022 р.	2023 р.	2024 р.	Разом	Архітектури
Англійська	17 статей [6, 14, 17, 18, 23, 28, 22, 25, 16, 42, 47, 31, 36, 37, 39, 30, 43]	19 статей [7, 8, 11, 12, 15, 19, 41, 20, 21, 26, 27, 32, 33, 34, 35, 40, 45, 46, 48]	2 статті [13, 44]	38 статей	Transformer, BERT, GPT-2, GPT-3, RNN, LSTM, CNN, GAN, Seq2Seq
Німецька	3 статті [39, 37, 18]	2 статті [45, 27]	–	5 статей	Conditional GAN, StyleGAN, DCGAN
Китайська	1 стаття [37]	3 статті [27, 29, 10]	–	4 статті	Graph Neural Networks, B2T
Французька	1 стаття [39]	–	–	1 стаття	Conditional GAN, StyleGAN, DCGAN
Бенгальська	1 стаття [16]	1 стаття [48]	–	2 статті	CNN, YOLO, mBART
Урду	–	1 стаття [33]	–	1 стаття	GPT-2
Хінді, малаялам, маратхі, орія, панджабі, тамільська	–	1 стаття [48]	–	1 стаття	HipoRank, mBART, mT5
Шекспірівська англійська	1 стаття [29]	–	–	1 стаття	Modified DCGAN
Румунська	1 стаття [18]	1 стаття [27]	–	2 статті	DCGAN, BART
Корейська	–	1 стаття [19]	–	1 стаття	Modified DCGAN

Німецька мова представлена у 5 статтях із використанням архітектури GAN (Conditional GAN, StyleGAN та DCGAN). Китайська мова представлена у 4 статтях із використанням Graph Neural Networks та архітектури B2T. Бенгальська мова представлена у 2 статтях з використанням CNN та YOLO. Румунська мова представлена у 2 статтях із використанням архітектур DCGAN та BART. Французька, урду, шекспірівська англійська та

корейська мови згадуються по одній статті кожна, з використанням різних архітектур, як-от Conditional GAN, StyleGAN, DCGAN та GPT-2. У 2023 р. у [48] досліджується генерування текстів одразу кількома індійськими мовами – хінді, малаялам, маратхі, орія, панджабі та тамільська – з використанням архітектур HipoRank, mBART та mT5.

Порівнюючи результати огляду 2022–2024 рр. із попереднім оглядом [2], можна виділити такі спостереження:

- англійська залишається найбільш широко використовуваною мовою для генерування тексту як у 2015–2021 рр., так і в 2022–2024 рр., проте збільшується кількість досліджень щодо генерування текстів іншими мовами, особливо мовами з обмеженими ресурсами;
- у 2022–2024 рр. з'явилися перші дослідження про генерування текстів на урду, хінді, малаялам, маратхі, орія, панджабі та тамільські мові, що свідчить про зростаючий інтерес до мовної диверсифікації моделей генерування тексту;
- стаття [48] демонструє можливість генерування текстів одразу кількома індійськими мовами з використанням сучасних архітектур, як-от HipoRank, mBART та mT5, що не було представлено в попередньому огляді;
- для генерування текстів різними мовами використовуються як традиційні архітектури (RNN, LSTM, CNN), так і більш сучасні підходи, як-от Transformer, BERT, GPT-2, GPT-3, GAN та Graph Neural Networks;
- спостерігається тенденція до розширення спектра мов моделей генерування тексту та використання більш різноманітних архітектур нейронних мереж.

Отже, порівняння результатів двох оглядів демонструє, що хоча англійська мова залишається домінуючою в дослідженнях із генерування тексту, але зростає інтерес до розробки моделей для інших мов, особливо мов з обмеженими ресурсами. Поява досліджень, присвячених генеруванню текстів мовами урду, хінді, малаялам, маратхі, орія, панджабі та тамільською свідчить про розширення мовних можливостей генераторів тексту. Використання сучасних архітектур нейронних мереж, як-от Transformer, BERT, GPT-2, GPT-3, GAN та Graph Neural Networks, дає змогу покращити якість та ефективність генерування тексту на різних мовах.

Порівнюючи розподіл мов у старому та новому оглядах із розподілом мов за кількістю моделей на Hugging Face [52], звернемо увагу на такі факти:

- англійська мова домінує в усіх трьох розподілах. У старому та новому оглядах вона найчастіше використовується для генерування тексту, а на Hugging Face для неї доступно найбільше моделей – 51 738. Це свідчить про значну увагу дослідників і розробників до англійської мови та про велику кількість ресурсів для неї;
- китайська мова посідає друге місце за кількістю моделей на Hugging Face (4 546) та згадується в кількох статтях у новому огляді. Це вказує на зростаючий інтерес до генерації тексту китайською мовою та розвиток відповідних ресурсів;
- французька, іспанська, російська та німецька мови мають значну кількість моделей на Hugging Face – від 2 326 до 4 049, але рідше згадуються в оглядах. Це може свідчити

про те, що, незважаючи на наявність ресурсів, дослідження з генерування тексту для цих мов не так широко представлені в літературі;

- мови з обмеженими ресурсами, як-от бенгальська, урду, арабська та хінді, згадуються в новому огляді, що свідчить про зростаючий інтерес до розробки моделей генерування тексту для цих мов. Однак кількість доступних моделей на Hugging Face для цих мов (від 670 до 1 674) значно менша, ніж для англійської;
- на Hugging Face представлено понад 200 мов, що значно більше, ніж згадується в оглядах. Це вказує на те, що дослідження з генерування тексту охоплюють лише частину мов, для яких доступні моделі та ресурси;
- японська, корейська, індонезійська, арабська та деякі інші мови мають значну кількість моделей на Hugging Face (від 1 674 до 2 920), але майже не згадуються в оглядах. Це може свідчити про потенціал для подальших досліджень із генерування тексту цими мовами.

Результати аналізу розподілів мов показують, що, незважаючи на домінування англійської мови в дослідженнях та наявних ресурсах, спостерігається зростаючий інтерес до генерування тексту іншими мовами, особливо мовами з обмеженими ресурсами. Однак кількість доступних моделей та ресурсів для цих мов все ще значно менша, порівняно з англійською. Доступність великої кількості моделей на Hugging Face для деяких мов, які рідко згадуються в оглядах, вказує на потенціал подальших досліджень та розробок у цій галузі.

Висновки

За результатами проведеного систематичного огляду застосування штучних нейронних мереж для генерування текстового контенту у 2022–2024 рр. виявлено 6 тенденцій:

1. Збільшується кількість статей у наукових журналах, порівняно з матеріалами конференцій, що може свідчити про більш ґрунтовне висвітлення питань генерування тексту.
2. З-поміж передових методів глибокого навчання для генерування тексту домінують моделі на основі архітектури Transformer – GPT-2, GPT-3, BERT та їх модифікації. Також набувають популярності підходи з використанням механізмів уваги та контрольованого генерування тексту. Загалом спостерігається тенденція до переходу від традиційних до більш інноваційних та ефективних моделей.
3. Найбільш широко використовуваними метриками для оцінювання ефективності моделей генерування тексту залишаються BLEU та ROUGE, а також оцінювання якості людьми (Human Evaluation). Водночас у 2022–2024 рр. з'явилися нові метрики, як-от BERTScore, Fluency, Coherence, Diversity, N-gram Overlap та Embedding Similarity, що свідчить про активний розвиток методів оцінювання якості згенерованого тексту.
4. Набори даних для генерування тексту продовжують активно розвиватися, охоплюючи нові домени та типи даних. Використовується більше різноманітних типів даних – таблиці з описом, зображення, музика, переклади тощо – та зростає інтерес до нерозмічених даних і комбінованих підходів.

5. Сфера застосування генерування тексту розширюється, охоплюючи нові галузі та напрями. Популярним є застосування генерування тексту з таблиць та графів знань, контрольоване генерування тексту та генерування медичних текстів, що свідчить про зростання інтересу до методів, які дають змогу ефективно обробляти структуровані дані та отримувати більш релевантні і якісні результати.
6. Хоча англійська мова продовжує домінувати в дослідженнях із генерування тексту, спостерігається зростаючий інтерес до розробки моделей на інших мовах, особливо на мовах з обмеженими ресурсами. Використання сучасних архітектур нейронних мереж дає змогу покращити якість та ефективність генерування тексту для різних мов.

Представлений систематичний огляд доповнює та розширює попередні дослідження, зокрема огляд [2] за 2015–2021 рр., шляхом аналізу новітніх досягнень у сфері нейронного генерування тексту продовж 2022–2024 рр. На відміну від попередніх оглядів, приділено особливу увагу інноваційним архітектурам моделей, як-от Transformer-based (GPT-2, GPT-3, BERT), механізмам уваги та контрольованому генеруванню тексту. До того ж огляд висвітлює появу нових метрик, зокрема BERTScore та Diversity Score, що доповнюють традиційні показники якості, як-от BLEU та ROUGE.

Ще однією відмінністю огляду є акцент на зростанні інтересу до застосування генеративних моделей для низькоресурсних мов та розширенні сфер застосування, включно із створенням анотацій, машинним перекладом, генеруванням тексту на основі таблиць та графів знань, а також генеруванням медичних текстів. Ці результати підкреслюють потенціал нейронного генерування тексту для вирішення широкого спектра прикладних завдань у різних предметних областях.

Серед найбільш перспективних технологій та підходів, що з'явилися у 2022–2024 рр., відзначимо моделі на основі архітектури Transformer, зокрема GPT-3 та його модифікації. Ці моделі демонструють вражаючі результати у генеруванні зв'язного та семантично релевантного тексту; вони здатні генерувати тексти на основі невеликої кількості прикладів (few-shot learning) та можуть бути адаптовані для вирішення широкого спектра завдань NLP. Іншим перспективним напрямом є контрольоване генерування тексту, яке дає змогу управляти стилем, тональністю та семантикою згенерованого тексту за допомогою додаткових сигналів керування.

Водночас актуальними залишаються виклики, пов'язані з адаптацією моделей генерування тексту для низькоресурсних мов та розробкою ефективних підходів до генерування тексту в умовах обмеженості навчальних даних. Попри певний прогрес, все ще існує потреба в розробці спеціалізованих архітектур та методів попереднього навчання, які б дали змогу покращити якість генерування тексту в таких умовах.

Відзначимо потенціал дифузійних моделей (Diffusion Models), які використовують ітеративний денойзинг для генерування високоякісних текстів. Ці моделі можуть бути особливо ефективними для генерування довгих послідовностей і мають потенціал для покращення якості та різноманітності згенерованого тексту.

Важливим напрямом подальших досліджень є розробка більш інтерпретовних та пояснюваних моделей генерування тексту, які дають змогу краще зрозуміти особливості процесу генерації для різних мов та доменів і полегшують взаємодію між людиною та

штучним інтелектом. Окремої уваги заслуговують етичні питання та відповідні потенційні ризики. З розвитком потужних мовних моделей зростають ризики автоматичного створення фейкових новин, дезінформації та маніпулятивного контенту. Тому важливо досліджувати методи детекції та протидії таким загрозам, а також розробити етичні принципи та дієві рекомендації щодо відповідального використання технологій генерування тексту.

Отримані результати демонструють активний розвиток та еволюцію методів генерування текстового контенту з використанням штучних нейронних мереж у 2022–2024 рр., порівняно з попереднім періодом. Застосування нових архітектур, підходів та метрик дає змогу покращити якість і релевантність згенерованого тексту, а також розширити сферу застосування цих технологій.

Результати систематичного огляду можуть бути корисними для дослідників, розробників і практиків у галузі обробки природної мови та штучного інтелекту, оскільки вони надають актуальну інформацію про сучасні тенденції, методи й перспективні напрями розвитку нейронного генерування текстового контенту. Отримані висновки можуть бути використані під час вибору методів, архітектур, метрик та наборів даних для розробки нових моделей і систем генерування тексту, а також для визначення пріоритетних напрямів подальших досліджень у цій області з урахуванням актуальних викликів та етичних аспектів.

Література

1. Ganegedara, T. (2018). *Natural Language Processing with TensorFlow: Teach language to machines using Python's deep learning library*. Packt Publishing. https://tinyurl.com/3xps3c5u
2. Fatima, N., Imran, A. S., Kastrati, Z., Daudpota, S. M., & Soomro, A. (2022). A systematic literature review on text generation using deep neural network models. *IEEE Access*, 10, 53490-53503. https://doi.org/10.1109/ACCESS.2022.3174108
3. OpenAI (2022). Introducing ChatGPT. https://openai.com/blog/chatgpt
4. (2023). Large language models – Google Trends. https://trends.google.com/trends/explore?date=2022-01-01%202023-12-21&q=large%20language%20models&hl=en
5. Page, M. J., McKenzie, J. E., Bossuyt, P. M., Boutron, I., Hoffmann, T. C., Mulrow, C. D., … Moher, D. (2021). The PRISMA 2020 statement: An updated guideline for reporting systematic reviews. *BMJ*, 372, n71. https://doi.org/10.1136/bmj.n71
6. Bas, A., Topal, M. O., Duman, Ç., & Van Heerden, I. (2022). A brief history of deep learning-based text generation. In *J. M. Alja'Am, S. AlMaadeed, S. A. Elseoud, & O. Karam (eds.), Proceedings of the International Conference on Computer and Applications* (pp. 1–4). IEEE. https://doi.org/10.1109/ICCA56443.2022.10039545
7. Zhu, J., Ma, X., Lin, Z., & De Meo, P. (2023). A quantum-like approach for text generation from knowledge graphs. *CAAI Transactions on Intelligence Technology*. https://doi.org/10.1049/cit2.12178
8. Zhang, H., Song, H., Li, S., Zhou, M., & Song, D. (2023). A survey of controllable text generation using transformer-based pre-trained language models. *ACM Computing Surveys*, 56, 64. https://doi.org/10.1145/3617680

9. Yu, W., Zhu, C., Li, Z., Hu, Z., Wang, Q., Ji, H., & Jiang, M. (2022). A survey of knowledge-enhanced text generation. *ACM Computing Surveys*, 54, 227. https://doi.org/10.1145/3512467
10. Wu, T., Wang, H., Zeng, Z., Wang, W., Zheng, H.-T., & Zhang, J. (2023). Enhancing text generation with cooperative training. *Frontiers in Artificial Intelligence and Applications*, 372, 2704-2711. https://doi.org/10.3233/FAIA230579
11. Du, H., Xing, W., & Pei, B. (2023). Automatic text generation using deep learning: providing large-scale support for online learning communities. *Interactive Learning Environments*, 31, 5021–5036. https://doi.org/10.1080/10494820.2021.1993932
12. Chen, Q., Sun, H., Liu, H., Jiang, Y., Ran, T., Jin, X., … Niu, Z. (2023). An extensive benchmark study on biomedical text generation and mining with ChatGPT. *Bioinformatics*, 39, btad557. https://doi.org/10.1093/bioinformatics/btad557
13. Alonso, I., Agirre, E. (2024). Automatic logical forms improve fidelity in table-to-text generation. *Expert Systems with Applications*, 238. https://doi.org/10.1016/j.eswa.2023.121869
14. Kreiss, E., Fang, F., Goodman, N. D., & Potts, C. (2022). Concadia: Towards image-based text generation with a purpose. In *Y. Goldberg, Z. Kozareva, & Y. Zhang (eds.).* Proceedings *of the 2022 Conference on Empirical Methods in Natural Language Processing* (pp. 4667–4684). Association for Computational Linguistics. https://doi.org/10.18653/v1/2022.emnlp-main.308
15. Rao, K. Y., Rao, K. S., & Narayana, S. V. S. (2023). Conditional-aware sequential text generation in knowledge-enhanced conversational recommendation system. *Journal of Theoretical and Applied Information Technology*, 101, 2820–2836. http://www.jatit.org/volumes/Vol101No7/30Vol101No7.pdf
16. Tazalli, T., Aunshu, Z. A., Liya, S. S., Hossain, M., Mehjabeen, Z., Ahmed, M. S., & Hossain, M. I. (2022). Computer vision-based Bengali sign language to text generation. In *5th IEEE International Image Processing, Applications and Systems Conference* (pp. 1–6). IEEE. https://doi.org/10.1109/IPAS55744.2022.10052928
17. Teng, Z., Chen, C., Zhang, Y., & Zhang, Y. (2022). Contrastive latent variable models for neural text generation. In J. Cussens & K. Zhang (Eds.), *Proceedings of Machine Learning Research* (Vol. 180, pp. 1928–1938). ML Research Press. https://proceedings.mlr.press/v180/teng22a.html
18. An, C., Feng, J., Lv, K., Kong, L., Qiu, X., Huang, X. (2022). CONT: contrastive neural text generation. In *Proceedings of the 36th International Conference on Neural Information Processing Systems*, NIPS'22 (p. 160). Curran Associates Inc., Red Hook, NY, USA. https://dl.acm.org/doi/10.5555/3600270.3600430
19. Seo, H., Jung, S., Jung, J., Hwang, T., Namgoong, H., & Roh, Y.-H. (2023). Controllable text generation using semantic control grammar. *IEEE Access*, 11, 26329-26343. https://doi.org/10.1109/ACCESS.2023.3252017
20. Zhou, W., Jiang, Y. E., Wilcox, E., Cotterell, R., & Sachan, M. (2023). Controlled text generation with natural language instructions. In *A. Krause, E. Brunskill, C. K., B. Engelhardt, S. Sabato, & J. Scarlett (eds.). Proceedings of Machine Learning Research* (Vol. 202, pp. 42602–42613). ML Research Press. https://proceedings.mlr.press/v202/zhou23g/zhou23g.pdf

21. Bayer, M., Kaufhold, M.-A., Buchhold, B., Keller, M., Dallmeyer, J., Reuter, C. (2023). Data augmentation in natural language processing: a novel text generation approach for long and short text classifiers. *International Journal of Machine Learning and Cybernetics*, 14, 135–150. https://doi.org/10.1007/s13042-022-01553-3
22. Hong, S., Moon, S., Kim, J., Lee, S., Kim, M., Lee, D., & Kim, J.-Y. (2022). DFX: A low-latency multi-FPGA appliance for accelerating transformer-based text generation. In *Proceedings of the Annual International Symposium on Microarchitecture* (pp. 616–630). IEEE Computer Society. https://doi.org/10.1109/MICRO56248.2022.00051
23. Ghazvininejad, M., Karpukhin, V., Gor, V., & Celikyilmaz, A. (2022). Discourse-aware soft prompting for text generation. In *Y. Goldberg, Z. Kozareva, & Y. Zhang (eds.).* Proceedings *of the 2022 Conference on Empirical Methods in Natural Language Processing* (pp. 4570–4589). Association for Computational Linguistics. https://doi.org/10.18653/v1/2022.emnlp-main.303
24. Koplin, J. J. (2023). Dual-use implications of AI text generation. *Ethics and Information Technology*, 25, 32. https://doi.org/10.1007/s10676-023-09703-z
25. Pautrat-Lertora, A., Perez-Lozano, R., & Ugarte, W. (2022). EGAN: Generatives adversarial networks for text generation with sentiments. In *F. Coenen, A. Fred, & J. Filipe (eds.). International Joint Conference on Knowledge Discovery, Knowledge Engineering and Knowledge Management* (Vol. 1, pp. 249-256). Science and Technology Publications. https://doi.org/10.5220/0011548100003335
26. Wu, J., Guo, Y., Gao, C., & Sun, J. (2023). An automatic text generation algorithm of technical disclosure for catenary construction based on knowledge element model. *Advanced Engineering Informatics*, 56, 101913. https://doi.org/10.1016/j.aei.2023.101913
27. Li, Y., Cui, L., Yan, J., Yin, Y., Bi, W., Shi, S., & Zhang, Y. (2023). Explicit syntactic guidance for neural text generation. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics* (Vol. 1, pp. 14095–14112). Association for Computational Linguistics. https://doi.org/10.18653/v1/2023.acl-long.788
28. Chu, X. (2022). Feature extraction and intelligent text generation of digital music. *Computational Intelligence and Neuroscience*, 2022. https://doi.org/10.1155/2022/7952259
29. Shahriar, S. (2022). GAN computers generate arts? A survey on visual arts, music, and literary text generation using generative adversarial network. *Displays*, 73, 102237. https://doi.org/10.1016/j.displa.2022.102237
30. Strobelt, H., Kinley, J., Krueger, R., Beyer, J., Pfister, H., & Rush, A. M. (2022). GenNI: Human-AI collaboration for data-backed text generation. *IEEE Transactions on Visualization and Computer Graphics*, 28, 1106–1116. https://doi.org/10.1109/TVCG.2021.3114845
31. Yin, X., & Wan, X. (2022). How do Seq2Seq models perform on end-to-end data-to-text generation? In *S. Muresan, P. Nakov, & A. Villavicencio (eds.). Proceedings of the Annual Meeting of the Association for Computational Linguistics* (Vol. 1, pp. 7701–7710). Association for Computational Linguistics. https://doi.org/10.18653/v1/2022.acl-long.531
32. Montella, S., Nasr, A., Heinecke, J., Bechet, F., & Rojas-Barahona, L. M. (2023). Investigating the effect of relative positional embeddings on AMR-to-text generation with structural adapters. In *EACL 2023 – 17th Conference of the European Chapter of the*

Association for Computational Linguistics (pp. 727–736). Association for Computational Linguistics. https://doi.org/10.18653/v1/2023.eacl-main.51

33. Fatima, N., Daudpota, S. M., Kastrati, Z., Imran, A. S., Hassan, S., Elmitwally, N. S. (2023). Improving news headline text generation quality through frequent POS-Tag patterns analysis. *Engineering Applications of Artificial Intelligence*, 125, 106718. https://doi.org/10.1016/j.engappai.2023.106718
34. Seifossadat, E., & Sameti, H. (2024). Improving semantic coverage of data-to-text generation model using dynamic memory networks. *Natural Language Engineering*, 30, 454-479. https://doi.org/10.1017/S1351324923000207
35. Meyer, C., Adkins, D., Pal, K., Galici, R., Garcia-Agundez, A., & Eickhoff, C. (2023). Neural text generation in regulatory medical writing. *Frontiers in Pharmacology*, 14. https://doi.org/10.3389/fphar.2023.1086913
36. Lu, X., Welleck, S., West, P., Jiang, J., Kasai, D., Khashabi, R., Le Bras, L., Qin, Y., Yu, R., Zellers, N. A., Smith, Y., & Choi, Y. (2022). NEUROLOGIC AFesque decoding: Constrained text generation with lookahead heuristics. In *NAACL 2022 – 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies* (pp. 780–799). Association for Computational Linguistics. https://doi.org/10.18653/v1/2022.naacl-main.57
37. Xu, W., Tuan, Y., Lu, Y., Saxon, M., Li, L., & Wang, W. Y. (2022). Not all errors are equal: Learning text generation metrics using stratified error synthesis. In *Y. Goldberg, Z. Kozareva, & Y. Zhang (eds.). Findings of the Association for Computational Linguistics: EMNLP 2022* (pp. 6588–6603). Association for Computational Linguistics. https://doi.org/10.18653/v1/2022.findings-emnlp.489
38. Hanafi, A., Bouhorma, M., & Elaachak, L. (2022). Machine learning-based augmented reality for improved text generation through recurrent neural networks. *Journal of Theoretical and Applied Information Technology*, 100, 518–530. http://www.jatit.org/volumes/Vol100No2/18Vol100No2.pdf
39. Le, H., Le, D.-T., Weber, V., Church, C., Rottmann, K., Bradford, M., & Chin, P. (2022). Semi-supervised adversarial text generation based on Seq2Seq models. In *EMNLP 2022 – Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing: Industry Track* (pp. 264–272). Association for Computational Linguistics. https://doi.org/10.18653/v1/2022.emnlp-industry.26
40. Yue, X., Inan, H. A., Li, X., Kumar, G., McAnallen, J., Shajari, H., Sun, H., Levitan, D., & Sim, R. (2023). Synthetic text generation with differential privacy: A simple and practical recipe. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics* (Vol. 1, pp. 1321–1342). Association for Computational Linguistics. https://doi.org/10.18653/v1/2023.acl-long.74
41. Lin, Z., Gong, Y., Shen, Y., Wu, T., Fan, Z., Lin, C., Duan, N., & Chen, W. (2023). Text generation with diffusion language models: A pre-training approach with continuous paragraph denoise. In *Proceedings of the 40th International Conference on Machine Learning*. JMLR.org. https://dl.acm.org/doi/abs/10.5555/3618408.3619275

42. Amin, M. S., Mazzei, A., Anselma, L. (2022). Towards Data Augmentation for DRS-to-Text Generation. *CEUR Workshop Proceedings*, 3287, 141–152. https://ceur-ws.org/Vol-3287/paper14.pdf
43. Chen, M., Lu, X., Xu, T., Li, Y., Zhou, J., Dou, D., Xiong, H. (2022). Towards Table-to-Text Generation with Pretrained Language Model: A Table Structure Understanding and Text Deliberating Approach. In *Y. Goldberg, Z. Kozareva, Y. Zhang (eds.). Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, EMNLP 2022 (pp. 8199–8210). Association for Computational Linguistics (ACL). https://doi.org/10.18653/v1/2022.emnlp-main.562
44. Agarwal, V., Ghosh, S., BSS, H., Arora, H., Raja, B. R. K. (2024). TrICy: Trigger-Guided Data-to-Text Generation With Intent Aware Attention-Copy. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 32, 1173–1184. https://doi.org/10.1109/TASLP.2024.3353574
45. Si, W. M., Backes, M., Zhang, Y., & Salem, A. (2023). Two-in-one: A model hijacking attack against text generation models. In *32nd USENIX Security Symposium* (Vol. 3, pp. 2223–2240). USENIX Association. https://www.usenix.org/system/files/usenixsecurity23-si.pdf
46. Gong, H., Feng, X., & Qin, B. (2023). Quality control for distantly-supervised data-to-text generation via meta learning. *Applied Sciences*, 13, 5573. https://doi.org/10.3390/app13095573
47. Mou, L. (2022). Search and learning for unsupervised text generation. *AI Magazine*, 43, 344-352. https://doi.org/10.1002/aaai.12068
48. Taunk, D., Sagare, S., Patil, A., Subramanian, S., Gupta, M., & Varma, V. (2023). XWikiGen: Cross-lingual summarization for encyclopedic text generation in low resource languages. In *ACM Web Conference 2023 – Proceedings of the World Wide Web Conference* (pp. 1703–1713). Association for Computing Machinery. https://doi.org/10.1145/3543507.3583405
49. Introducing the next generation of Claude (2024). https://www.anthropic.com/news/claude-3-family
50. Awesomegpts.ai (2024). Scholar GPT. https://chatgpt.com/g/g-kZ0eYXlJe-scholar-gpt?oai-dm=1
51. Slobodianiuk, A. V. (2024). Papers' review. https://docs.google.com/spreadsheets/d/e/2PACX-1vR6ZUaeeBjVgVl-do6QXm-Pua-HdztOxjC4DUqunrSDZ_-YSRz-Ng9xktYH9b0LDT502SiVy3YePx9F/pubhtml
52. Hugging Face (2024). Languages. https://huggingface.co/languages

Рукопис отримано – 04/03/2025 р.; прийнято до публікації – 26/03/2025 р.

Evolution of neural text generation models: A systematic review of research from 2022–2024

Artem Slobodianiuk, Serhiy Semerikov

Abstract

Recent years have witnessed significant advancements in neural text generation driven by the emergence of large language models and growing interest in this field. This systematic review identifies and summarizes current trends, approaches, and methods in neural text generation from 2022 to 2024, complementing a previous review covering 2015–2021. Following the PRISMA methodology, 89 articles were initially selected from the Scopus database, of which 43 articles remained after applying inclusion and exclusion criteria. The review reveals a shift towards innovative model architectures such as Transformer-based models (GPT-2, GPT-3, BERT), attention mechanisms, and controllable text generation. While BLEU, ROUGE, and human evaluation remain the most popular evaluation metrics, new metrics like BERTScore have emerged. Datasets span diverse domains and data types, with growing interest in unlabeled data. Applications have expanded to areas such as text summarization, machine translation, table-to-text generation, knowledge graph-based generation, and medical text generation. Although English dominates, there is increasing research on low-resource languages such as German and Chinese. The review also highlights current challenges in the field, including adapting models for low-resource languages, generating text with limited training data, and ethical considerations related to the use of powerful language models. The authors emphasize the importance of developing more efficient and interpretable architectures, improving controllable text generation methods, and creating new evaluation metrics. Taking into account current challenges and ethical considerations, the review also points to future research.

Keywords: neural architectures, neural text generation, controlled text generation, deep learning, systematic review, natural language processing, evaluation metrics, datasets, applications, low-resource languages, transformers, attention mechanisms.

УДК 001.9

Повідомлення про ретракцію статті «Вплив інтелектуального світлофорного регулювання на пропускну здатність міського перехрестя»

Сергій Дмитрович Штовба
професор, д-р техн. наук
ORCID: 0000-0003-1302-4899
s.shtovba@donnu.edu.ua
головний редактор журналу

Донецький національний університет імені Василя Стуса

Ключові слова:
ретракція;
наукова етика.

Статтю Ніколюка П. К. «Вплив інтелектуального світлофорного регулювання на пропускну здатність міського перехрестя» ретраговано з першого номера журналу «Ukrainian Journal of Information Systems and Data Science» за 2023 р.

DOI: https://doi.org/10.31558/2786-9482.2024.2.5

Статтю Ніколюка П. К. «Вплив інтелектуального світлофорного регулювання на пропускну здатність міського перехрестя», https://doi.org/10.31558/2786-9482.2023.1.3 ретраговано з першого номера журналу «Ukrainian Journal of Information Systems and Data Science» за 2023 р. на підставі висновку комісії редакційної колегії журналу від 24 лютого 2025 р.

Комісія встановила такі чотири факти:

1. Стаття Ніколюка П. К. «Вплив інтелектуального світлофорного регулювання на пропускну здатність міського перехрестя», https://doi.org/10.31558/2786-9482.2023.1.3 опублікована 23 грудня 2023 р. в журналі «Ukrainian Journal of Information Systems and Data Science», в № 1 за 2023 р. Розгляд рукопису цієї статті редколегія розпочала 31 травня 2023 р., і 26 червня 2023 р. ухвалила рішення про прийняття його до публікації. Зміст номера журналу рекомендовано до друку Вченою радою Донецького національного університету імені Василя Стуса, протокол №10 від 30 червня 2023 р.

2. Стаття Ніколюка П. К. «Програмна інтелектуалізація міського транспортного перехрестя», https://doi.org/10.52058/2786-6025-2023-8(22)-396-411 опублікована 29 червня 2023 р. в журналі «Наука і техніка сьогодні», №8 за 2023 р. За повідомленням головного редактора журналу «Наука і техніка сьогодні» Ірини Жукової зазначена стаття Ніколюка П. К. надійшла до редакції 19 червня 2023 р.

3. Аналізовані статті є ідентичними за змістом. Тексти обох статей майже ідентичні, за винятком правок редакторського характеру та дрібних доопрацювань на підставі зауважень рецензентів. Зіставлення дат проходження статей в обох журналах вказує на паралельне направлення змістовно ідентичних рукописів у 2 журнали.

4. Збірник «Наукові нотатки» в 2018 р. у випуску 63 опублікував статтю «Інтелектуальне перехрестя» за авторства Ніколюка П. К., Комарова В. Ф. та Ніколюка П. П., http://nbuv.gov.ua/UJRN/Nn_2018_63_23. Значні фрагменти цієї статті включено у статтю «Вплив інтелектуального світлофорного регулювання на пропускну здатність міського перехрестя» без посилання на джерело.

Висновки. На основі встановлених фактів Комісія вважає, що наявні 2 порушення наукової етики: одночасне подання ідентичних рукописів у 2 журнали та самоплагіат великих фрагментів із раніше опублікованої статті. Такі порушення є підставами для ретракції статті із журналу «Ukrainian Journal of Information Systems and Data Science» відповідно до підпункту 7.1.1 «Положення про редакційну колегію електронного наукового періодичного журналу "Ukrainian Journal of Information Systems and Data Science"», затвердженого наказом ректора № 43/01-02 від 31 січня 2025 р. Відповідно до розділу 7 згаданого положення Комісія ухвалює рішення про ретракцію з журналу «Ukrainian Journal of Information Systems and Data Science» статті Ніколюка П. К. «Вплив інтелектуального світлофорного регулювання на пропускну здатність міського перехрестя». Рішення Комісії ухвалено одноголосно 24 лютого 2025 р.

Голова Комісії	професор Сергій ШТОВБА
Член Комісії	професор Павло КУЛАКОВ
Член Комісії	професор Вадим РОМАНЮК

Рукопис отримано – 04/03/2025 р.; прийнято до публікації – 04/03/2025 р.

Retraction note for article «The impact of intelligent traffic signal control on the capacity of an urban-controlled intersection»

Serhiy Shtovba

Abstract

The article «The influence of intelligent traffic light regulation on the capacity of an urban intersection» authorised by Nikoliuk, P. K. is retracted from Ukrainian Journal of Information Systems and Data Science, Volume 1, Issue 1, 2023.

Keywords: retraction, scientific ethics.