Спосіб розгортання AI платформи з використанням методології MLOPS

Володимир Русінов

Національний технічний університет України
«Київський політехнічний інститут імені Ігоря Сікорського»
Берестейський проспект, 37, Київ, 03056
https://orcid.org/0000-0002-4362-0248

Received 03.02.2025, accepted 29.03.2025
https://doi.org/10.32347/st.2025.3.1202

Анотація. Специфіка сучасного застосування задач ШІ потребує розробки плафторми, яка уможливлює постійну підтримку та швидку реконфігурацію. Наукова та інженерна література свідчить, що використання методів Machine Learning Operations та Machine Learning Pipeline (MLOps) при створенні AI-платформ для вбудованих систем дозволяє підвищити ефективність розробки та впровадження штучного інтелекту. MLOps надає можливість автоматизувати процеси розробки, навчання та експлуатації моделей машинного навчання, забезпечуючи швидку реакцію на зміни в навколишньому середовищі або вхідних даних. Це допомагає скоротити час на розгортання та обслуговування системи в цілому.

Для вирішення задачі розгортання AI-платформи пропонується використовувати спосіб, який базується на гібридній Edge-Cloud архітектурі. Даний підхід часто використовується в сучасній науковій літературі і знаходить прикладне застосування в різних сферах. Архітектура платформи, заснована на ієрархічних зв'язках між рівнями Edge і Cloud, диктує необхідність в створенні спеціалізованого підходу до розгортання платформи. Особливістю даної архітектури є застосування вбудованих систем безпосередньо в задачі ШІ, а саме, розпізнаванні образів. Даний підхід дозволяє реконфігурувати платформу, за умови оновлення моделі ШІ, тим самим підтримуючи її працездатність і коректне виконання задачі.

Враховуючи, що ключовим є баланс швидкості реконфігурації подібної платформи та відмовостійкості, для того, щоб виміряти загальну працездатність системи з певним ступенем точності, ми введемо кілька метрик, які допоможуть встановити параметри системи та порівняти її з альтернативними рішеннями. Розгортання програмного забезпечення для штучного інтелекту повинно супроводжуватися аналізом цільової системи. В рамках даної роботи проведено ряд тестів для кількісної оцінки продуктивності системи.

Ключові слова: штучний інтелект, MLOps, IoT, відмовостійкість, комп'ютерні мережі.

Володимир Русінов
Аспірант кафедри
обчислювальної техніки

ПОСТАНОВКА ПРОБЛЕМИ

Сучасні AI рішення потребують створення платформ, які здатні підтримувати їх життєвий цикл, починаючи від попереднього тестування нових моделей ШІ, закінчуючи постійною підтримкою впроваджених моделей. Часто, використання моделей ШІ має суто експериментальний характер і впровадження таких моделей не супроводжується планом розгортання, але все більше моделей впроваджується для постійного користування. Окрім того, що система може бути нестабільною і не підтримувати впровадження більш великих моделей ШІ, такий підхід призводить до складнощів при пошуку причин зменшення точності моделі. Тому, існують підходи до створення платформ.

Розгортання платформи для задач штучного інтелекту має супроводжуватися ретельним аналізом цільової системи.

Практичними результатами способу, який базується на використання методології MLOps, є прискорення процесів інтеграції вбудованих пристроїв в існуючі Cloud рішення та підвищення відмовостійкості AI-плафторми за рахунок постійного моніторингу стану платформи та її реконфігурації. Програмно-апаратні рішення на основі запропонованого підходу можуть знайти використання в багатьох сферах, зокрема в медицини, безпеки, логістики. Рішення про впровадження таких практик може значно покращити швидкість та правильність прийняття рішень щодо оновлення моделі, або поступового донавчання моделі.

Для того, щоб виміряти загальну зручність використання системи з певним ступенем точності, необіхдно проводити тестування та аналіз платформи, з фокусом на процесі розгортання та постійні підтримці працездатності системи. Такий підхід дозволяє не тільки виявляти потенційні проблеми на ранніх етапах, але й оцінити швидкодію платформи в реальному часі. У майбутньому інтеграція таких систем на рівні організацій зможе значно підвищити рівень автоматизації і точності в роботі не лише AI-рішень, але й у всіх аспектах бізнес-процесів, що веде до більш стабільного і надійного використання штучного інтелекту в реальних умовах.

АНАЛІЗ ОСТАННІХ ДОСЛІДЖЕНЬ

Сучасні наукові дослідження показують, що MLOps - це новий спосіб розгортання систем ШІ, який успадкував свої методи від DevOps, що показав позитивні результати в різних програмних системах. У різних статтях пропонуються різні варіації інструментів і конструкцій конвеєрів CI/CD, які дозволяють виконувати задачі ШІ масштабовано і підвищують загальну відмовостійкість системи. Ці рішення також пропонують метод автоматизації процесів, які потребують додаткових людських ресурсів, таких як моніторинг, ведення логування і реконфігурація. Сучасні рішення дають мінімальне уявлення про продуктивність таких систем.

В науковій літературі було виявлено, що підхід на основі інтеграції вбудованих пристроїв в процес MLOps є багатообіцяючим, але потребує додаткових досліджень. В літературі сказано про шляхи оптимізації взаємодії із Edge-кластерами, проте недостатньо дослідженим залишається питання відмовостійкості системи при високій завантаженості платформи та можливість використання AI-задач. Важливим аспектом, який потребує подальшого дослідження, є збір часових характеристик щодо використання таких гібридних систем, та порівняння з традиційними підходами.

МЕТА І ЗАДАЧІ ДОСЛІДЖЕННЯ

Метою статті є дослідження та впровадження принципів MLOps для розгортання AI платформи.

Для досягнення мети поставлені задачі:

— Проаналізувати сучасні підходи до створення AI-платформ, зокрема, методологію MLOps та виділити необхідні етапи для створення платформи;

— Розробити власний спосіб розгортання AI платформи на основі конвеєру MLOps за допомогою використання CI/CD конвеєру;

— Розробити тестові сценарії перевірки працездатності платформи за умов реконфігурації та високого навантаження запитами на платформу;

— Провести аналіз результатів тестування платформи, на основі запропонованого способу, згідно описаних сценаріїв та виділити напрямки подальшого дослідження.

ВИКЛАД ОСНОВНОГО МАТЕРІАЛУ ДОСЛІДЖЕННЯ

Операції машинного навчання (Machine Learning Operations, MLOps) з'явилися як концепція, заснована на методах DevOps, на меті яких створити набір процедур, які об'єднують інструменти, процедури розгортання та робочі процеси для швидкого та доступного створення моделей

машинного навчання [1]. Метододологія MLOps рекомендує автоматизувати та контролювати кожен крок створення та впровадження систем машинного навчання з подальшим логу-ванням та дослідженням внутрішніх процесів [2]. Практики MLOps можна сприймати як складну координацію набору принципів, прийомів та програмних компонентів, що використовуються інтегровано для виконання щонайменше п'яти базових завдань: збір даних, перетворення даних, безперервне навчання моделі машинного навчання, безперервна реалізація моделі, представлення результатів кінцевому користувачеві. Кінцевий продукт та його підтримка досягається завдяки використанню принципів MLOps.

Крім того, продуктивність рішення, яке не використовує стратегії MLOps, може з часом погіршитися, оскільки програми машинного навчання покладаються на дані, що використовуються для навчання моделі штучного інтелекту, і оскільки свіжі дані, тобто ті на яких модель не була натренована, постійно надходять до програми [5]. Для того, щоб гарантувати ефективність і якість рішення штучного інтелекту протягом тривалого часу, одним з найбільш доречних заходів MLOps є моніторинг рішень машинного навчання.

Запропоноване рішення передбачає створення спеціального наскрізного (end-to-end) конвеєра CI/CD. Конвеєр CI/CD для системи ШІ зазвичай починається з етапу інтеграції. Це передбачає витягування останніх змін коду з системи контролю версій, наприклад, Git, і об'єднання їх у спільну гілку. Потім запускаються автоматизовані тести для перевірки змін у коді. У випадку системи ШІ ці тести можуть включати модульні тести для окремих компонентів, а також інтеграційні тести, які оцінюють продуктивність моделей машинного навчання за набором попередньо визначених метрик.

AI-платформа, як програмно-апаратне рішення для задач штучного інтелекту, вимагає розробки окремої моделі ML, яка за своєю суттю є експериментальною. Типовий робочий процес розробки моделі машинного навчання починається зі збору додаткової інформації про дані та поставлене завдання. Дані необхідно проаналізувати (Exploratory Data Analysis (EDA)), очистити та попередньо обробити (інжиніринг та виділення ознак), щоб виділити основні ознаки з сирих даних. Після цього формується набір даних для навчання, тестування та верифікації [13].

Підхід до реалізації MLOps конвеєра для AI-платформи передбачає багатоступеневий процес, що забезпечує валідацію, тестування та інтеграцію AI компонентів. Етапи слідують один за одним, утворюючи більший конвеєр, який називається CI/CD. Спосіб включає наступні етапи:

1. Безперервна інтеграція (CI) - це практика інтеграції змін коду в існуючу кодову базу таким чином, щоб будь-які конфлікти між різними змінами коду, зробленими розробниками, швидко виявлялися і вирішувалися [2]. Оскільки фази планування та кодування виконуються до фази побудови, їх можна вважати пов'язаними з безперервною інтеграцією. Під час виконання фази CI, готуються нові зміни коду до розгортання, тому ця практика має безпосереднє значення для підвищення ефективності розгортання з точки зору якості та коректного виконання коду.

1.1. На етапі підготовки здійснюється комплексне тестування вихідного коду машинного навчання, включаючи синтаксичний аналіз, перевірку типів даних (семантичний аналіз), оцінку метрик продуктивності моделі. Платформа перевіряє відповідність кодової бази встановленим стандартам якості та вимогам архітектурної сумісності. Фаза збірки та тестування є компонентами DevOps, які беруть участь у безперервній інтеграції, оскільки кожна фіксація коду супроводжується автоматизованою генерацією коду з подальшим автоматизованим тестуванням.

1.2. Наступним етапом конвеєра є контейнеризація та ізоляція обчислювального середовища з використанням технологій як, наприклад, Docker. Даний етап забезпечує уніфіковане

розгортання AI-компонентів з високим рівнем масштабованості,

гнучкості та відмовостійкості. Підхід на основі конетейнеризації дозволяє ефективно розгортати нові ітерації моделі, швикдко вносити зміни з залежностями в програмній частині та ефективно сегрегувати програмну частину, яка відповідає за AI модель від модулів комунікації з іншими сервісами платформи та користувачами.

2. Безперервна доставка (CD) включає в себе фазу впровадження (релізу) коду, її мета - тримати додаток готовим до розгортання у виробничому середовищі. Безперервне розгортання виконується пізніше, оскільки воно автоматизує розгортання програмної частини платформи. Якщо модель недоступна, виробнича версія буде розгорнута користувачем вручну.

2.1. Першим етапом конвеєру CD є етап компіляції дистрибутиву, що, в рамках даного дослідження, передбачає трансформацію вихідного коду в артефакти розгортання. Реалізується контейнеризація компонентів з використанням сучасних технологічних підходів віртуалізації та ізоляції обчислювального середовища. Технології такі як Kubernetes можуть бути використані на цьому етапі для створення середовищ для тестування, валідації та використання AI-моделі.

механізми паралельного розгортання та інші стратегії направлені на прискорення розгортання та міграції частини програмних компонентів для відмовостійкості системи.

2.3. Завершальним етапом є (постійний) моніторинг та збір аналітики використання впровадженої платформи. Реалізується безперервне діагностування параметрів продуктивності, акумуляція системних метрик та реалізується автоматизована реакція на потенційні девіації функціонування інформаційної інфраструктури. Під девіаціями розуміються відмови як апаратного або програмного рівня. Стратегії відновлення працездатності системи включають або повне, або часткове виконання конвеєру, для відновлення працездатності.

Наявний шлях від отримання нової ітерації моделі в MLOps до подальшого розгортання показаний на Рис. 1. Модель, яка пройшла тест на точність в цьому процесі, надсилається менеджеру моделей разом з відповідним тестовим звітом. Щоб ефективно побудувати інфраструктуру під додатки, які застосовують штучний інтелект, необхідно враховувати різні фактори, як вибір правильних інструментів і технологій. Вибір правильних інструментів і технологій також має важливе значення

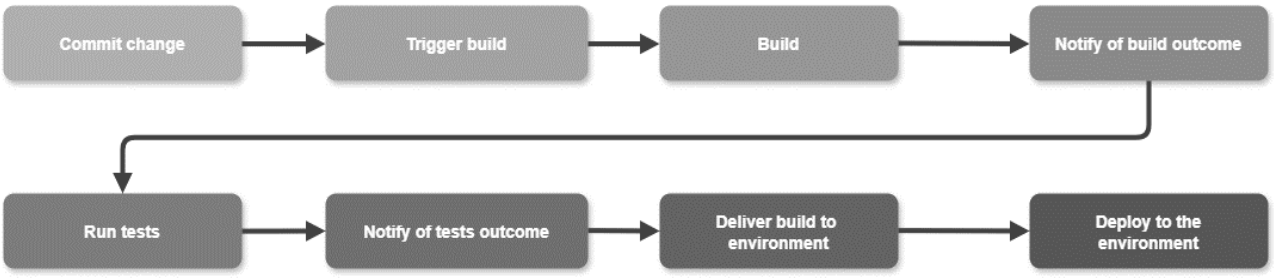

Рис. 1. Етапи CI/CD конвеєра

2.2. Наступним етапом є розгортання, яке реалізується через стратегію інкрементальної оновлення, що передбачає поетапне розгортання оновлених контейнерів (артефактів) із забезпеченням неперервності операційних процесів, тобто, під час оновлення програмної частини, платформа має зберігати працездатний стан. На цьому етапі можуть використовуються для побудови масштабованої та гнучкої інфраструктури для робочих навантажень ШІ.

Етап розгортання і моніторингу є фінальним етапом шляху MLOps. На цьому етапі розробники програмного забезпечення інтегрують перевірені моделі у відповідні додатки і забезпечують стабільність всієї прикладної реалізації платформи. Оскільки

на цьому етапі інтеграції можна отримати різні аспекти рішень і конфігурацій всього конвеєра, інженери MLOps повинні здійснювати безперервний моніторинг моделі, всього додатку і споживаних даних. [34] Іншу роль на цьому етапі розгортання інфраструктури відіграє інженер, який відповідає за проведення, побудову та тестування робочої системи.

Навчена модель розгортається для використання зовнішніми користувачами, за допомогою різних підходів. Два найпоширеніші підходи - це «модель як послуга» та «вбудована модель». У моделі як сервісі модель представлена у вигляді кінцевих точок API Representational State Transfer (REST), тобто модель розгортається на веб-сервері, щоб вона могла взаємодіяти через REST API, і будь-яка програма могла взаємодіяти з AI-платформою, передаючи вхідні дані через виклик API. Веб-сервер може працювати локально або в хмарі. З іншого боку, у випадку «вбудованої моделі», модель упаковується в програму, яка потім публікується. Цей варіант використання є практичним, коли модель розгортається на периферійному пристрої. Потрібно зауважити, що спосіб розгортання AI задачі на платформі залежить від виду взаємодії кінцевого користувача з платформою, в випадку, описаному в даній роботі, передбачається саме модель як сервіс.

Безперервний моніторинг всієї платформи на метрики, наприклад, продуктивності моделі та додатку, а також стану інфраструктури та даних, що використовуються моделлю, єосновою надійного продукту машинного навчання. Зосереджуючись на відстеженні даних через конвеєрні операції, бази даних логів можуть допомогти керувати і підтримувати зв'язок між даними і пов’язними з ними моделями. Зворотній зв’язок, в даному випадку, грає ключову роль у встановленні постійного потоку даних на основі розглянутого конвеєру, з можливістю постійного оновлення даних.

Апаратна частина платформи реалізується за допомогою поєднання Cloud технологій та Edge (вбудованих) пристроїв. Основою такої системи є топологія, вона диктує, фактично, яким чином взаємодіють вузли системи. Ієрархічна структура топології дерева підходить для створення гібридної системи із застосуванням Cloud та Edge технологій. Топологія двійкового дерева є однією з фундаментальних структур у теорії алгоритмів, графів, статистиці та інших комп’ютерних наук. Для бінарного дерева, кожен вузол може мати не більше двох дочірніх вузлів, відомих як лівий і правий, які в свою чергу можуть також бути внутрішніми вузлами або листями.

Основним недоліком бінарного дерева є відносно низька відмовостійкість, коли при одній відмові, можливе фактичне розбиття топології на дві частини. Вирішенням цієї проблеми є використання додаткових

$$\mathrm{D(n,k)} = \mathrm{k} \cdot \mathrm{D(n-1,k)} + (-1)^k * (k-1) \cdot \mathrm{D(n-1,k-1)} \quad (1)$$

дебруйнівських зв’язків. Додаткові зв’язки будуються за наступною формою:

де D(n,k) - число Дебруйна, S(n,k) - число Стерлінга другого роду, n - кількість елементів, k - кількість циклів, (k/j) - біноміальний коефіцієнт "k по j" (кількість способів вибрати j елементів з k без урахування порядку), а (-1)m - альтернатуюча сума.

Отже, перший тест має на меті перевірку швидкості розгортання системи для різних значень кількості вузлів. Отримання результатів дозволить проаналізувати часову характеристику платформи, її здатність обробляти запити при збільшенні кількості пристроїв, і про ефективність масштабування з використанням запропонованого методу.

Таблиця 1

Час розгортання платформи в залежності від кількості вузлів

Кількість пристроїв	Час розгортання
3	925.45 секунди
7	977.08 секунди
15	1122.28 секунди
31	1222.31 секунди
63	1359.14 секунди

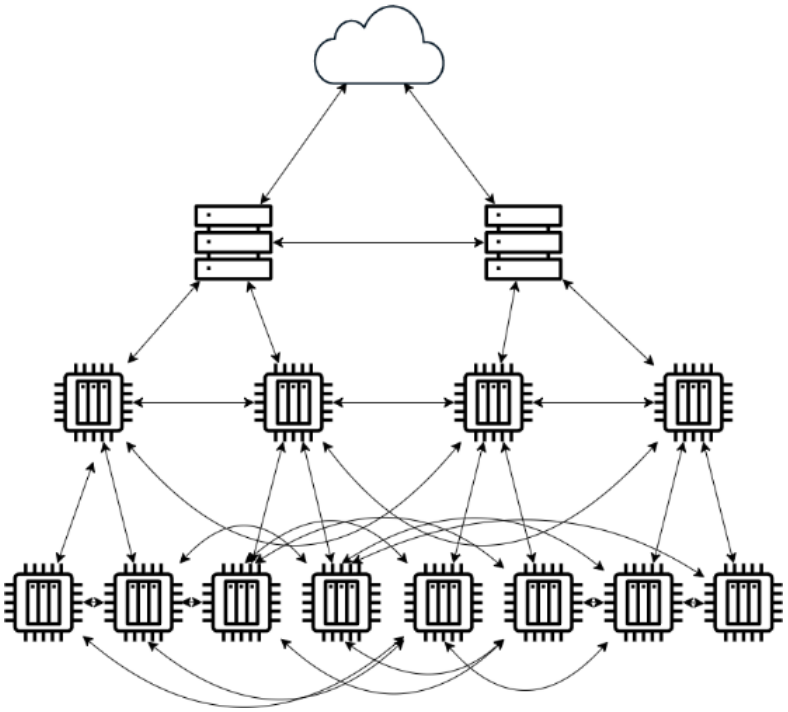

Рис. 2. Схема архітектури платформи на основі дерева з дебруйнівськими зв’язками

Рис. 3. Графік часу розгортання платформи в залежності від кількості вузлів

Вибір технологічного стеку для тестової системи є важливим етапом розробки програмної частини AI платформи, оскільки в залежності від обраного програмного забезпечення та бібліотек, можуть накладатись додаткові обмеження щодо розробки тестової програми. Для функціонального виконання більшості вимог до AI-платформи, зокрема API, завдяки якому можна формалізувати комунікацію з іншими пристроями, управління та логування моделі AI, виконується через мову Python. Python набув популярності завдяки своєму простому синтаксису, інтерпретованості, що дозволило легко впровадити компонентний підхід та розробити широку низку бібліотек для виконання різних функцій, в тому числі, розробки AI.

Другий тест пов'язаний з виконанням тестування навантаження плафторми великою кількістю запитів. Застосування тестування API AI-платформи дозволяє підтвердити її здатність до ефективної роботи у реальних умовах, коли більше ніж один вузол робить запит, і забезпечує певний рівень масштабованості та відмовостійкості. Кожен потік робить 100 запитів на платформу. Характеристиками, які будуть досліджуватись, є середній час відповіді, максимальний, мінімальний час, 99-ий персентиль та стандартне відхилення.

Таблиця 2.

Часові характеристики плафторми під час тестування навантаження

Tavg	Tmax	T_{99}	Tmin	SD	Потоки
MobileNet					
931.61	13872.96	839.44	42.34	1846.33	1
964.92	13776.17	948.56	43.05	1855.14	2
1011.02	11369.37	1008.52	43.75	2068.31	5
1302.33	15227.99	1198.00	45.87	2640.26	10
1535.48	29263.55	1660.89	47.99	3415.72	25
MobileNet Raspberry PI Tree-Debrujin (2 ступінь)					
2525	47039	1653.98	63.517	4010.7	1
2034.1	18519	2458.57	59.989	4346.4	2
2518.8	27836	3252.11	62.106	5607.8	5
3817.9	46181	3882.75	64.929	8098.7	10
4310.1	73512	4861.06	66.623	9674.8	25
MobileNet Raspberry PI Tree-Debrujin (2 ступінь, 2 воркери)					
513.59	13679.38	401.73	42.34	1035.63	1
532.18	13776.17	560.62	42.34	1060.80	2
605.79	11175.8	491.67	43.76	1066.26	5
857.04	10853.17	896.40	48.69	1461.96	10
879.48	15617.72	915.23	46.86	1695.69	25

В табл. 2 представлені результати другого тесту на навантаження платформи. Під час виконання тестування система працювала без відмов і не було перенасичення черги задачі, незважаючи на досить великий об'єм вхідного трафіку, що пояснюється вибором програмного стеку. З результатів видно, що залучення додаткових вузлів до системи значно знижує середній час відповіді.

На Рис. 4 наведені графіки тестування навантаження AI-платформи із застосуванням від 1 до 3 пристроїв, два з яких містять контейнеризовані AI-моделі, а також вузол-агрегатор, який забезпечує балансування навантаження між ними. Це дозволяє зменшити затримки під час обробки запитів та підвищити загальну продуктивність системи, завдяки оптимізації розподілу обчислювальних ресурсів між вузлами платформи. Графіки відображають динаміку швидкості реагування платформи, а також відображають взаємодію між воркерами та контейнерами моделей, що демонструє можливості даної архітектури в умовах реального навантаження.

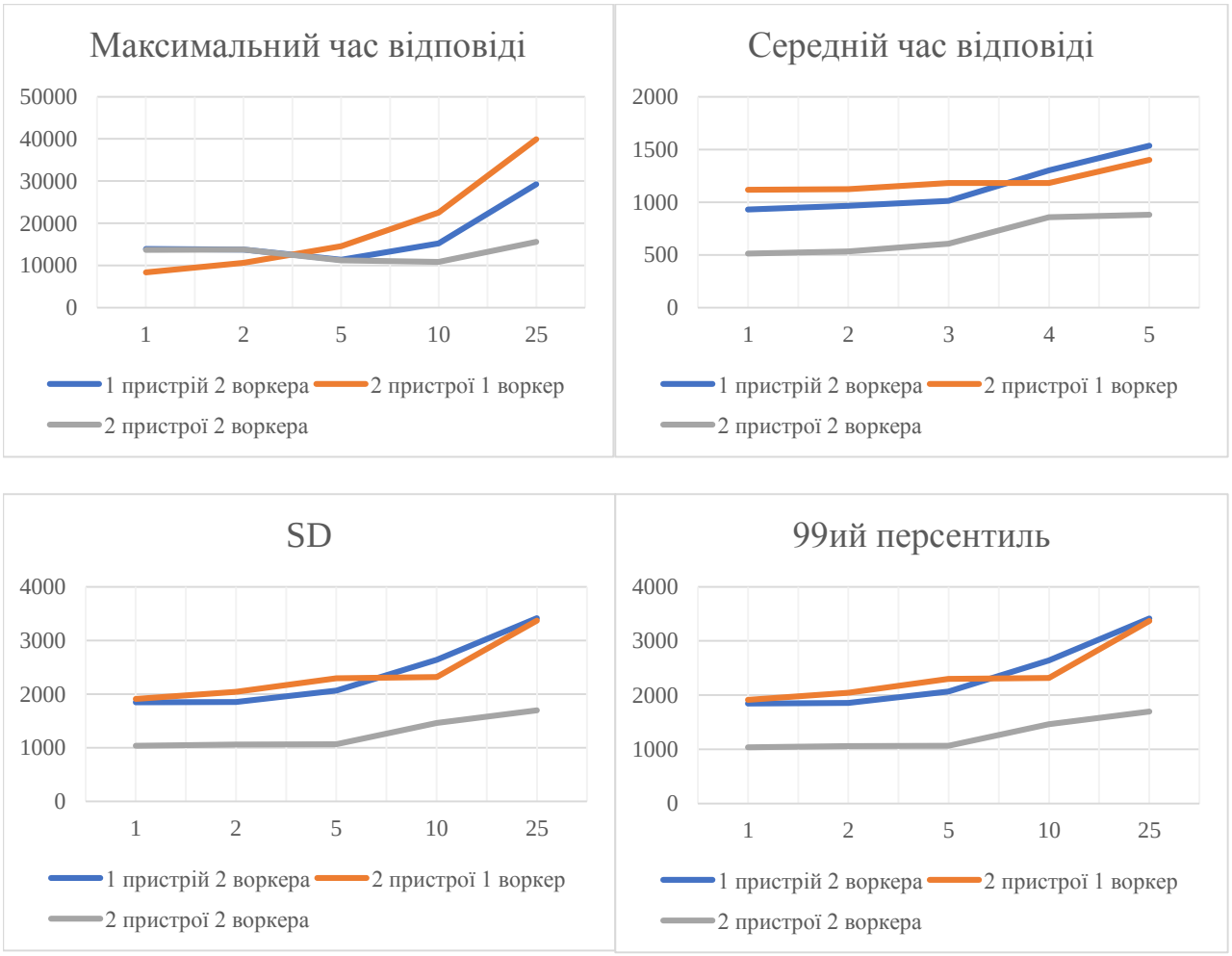

Рис. 4. Порівняльні графіки характеристик тестування системи на навантаження

ВИСНОВКИ

Було досліджено методолгію MLOps і доведено, що її застосування дозволяє створити ефективний спосіб розгортання AI-платформ. Дана платформа передбачає автоматизацію та моніторинг кожного етапу, включаючи збір даних, перетворення, навчання, впровадження та представлення результатів кінцевому користувачеві. Безперервна інтеграція та безперервна доставка сприяють ефективному розгортанні платформи. Етап розгортання забезпечує інтеграцію та стабільність моделі з постійним моніторингом.

Тестування виконано з використанням часових метрик. У таблиці результатів порівнюються метрики для запропонованого підходу, зосереджуючись на часових характеристиках платформи. Тестування розгортання платформи показує, що при масштабуванні системи, з 3 до 63 вузлів час розгортання збільшується лише на 433.69 секунди, що свідчить про здатність платформи до масштабування. Тестування навантаження показує, що платформа працює без відмов, за умови насичення черги запитів до 25 потоків одночасно, при цьому застосування додаткових вузлів в топології дозволяє зменшити середній час відповіді системи.

Запропонований метод є менш складним, ніж його альтернативи, проте він є більш легким, тому потребує менше часу на збірку та переконфігурацію після збоїв. Для підвищення ефективності запропонованого підходу можуть знадобитися додаткові дослідження в галузі оптимізації контейнерів, оптимізації моделей ШІ, а також мережевої взаємодії та її адміністрування. Важливо зазначити, що в реальних умовах можуть знадобитися додаткові кроки для підвищення відмовостійкості системи та збільшення її можливостей масштабування, за умови зміни програмного стеку або апаратної складової.

ЛІТЕРАТУРА

1. **Liu, Y., Ling, Z., Huo, B., Wang, B., Chen, T., & Mouine, E.** (2020). Створення платформи для операцій машинного навчання на основі відкритих джерел. *IFACPapersOnLine*, 53(5), 704–709. https://doi.org/10.1016/j.ifacol.2021.04.161
2. **Granlund, T., Kopponen, A., Stirbu, V., Myllyaho, L., & Mikkonen, T.** (2021). Виклики MLOps у налаштуванні мультиорганізаційної структури: досвід з двох реальних випадків. *2021 IEEE/ACM 1st Workshop on AI Engineering - Software Engineering for AI (WAIN)*, 82–88. https://doi.org/10.1109/WAIN52551.2021.00019
3. **Zhou, Y., Yu, Y., & Ding, B**. (2020). До MLOps: дослідження платформи для конвеєра машинного навчання. *2020 International Conference on Artificial Intelligence and Computer Engineering (ICAICE)*, 494–500. https://doi.org/10.1109/ICAICE51518.2020.00102
4. **Cardoso Silva, L., Rezende Zagatti, F., Silva Sette, B., Nildaimon dos Santos Silva, L., Lucrédio, D., Furtado Silva, D., & de Medeiros Caseli, H**. (2020). Бенчмаркінг рішень машинного навчання в умовах виробництва. *2020 19th IEEE International Conference on Machine Learning and Applications (ICMLA)*, 626–633. https://doi.org/10.1109/ICMLA51294.2020.00104
5. **Wilkes, B., Milani, A. M. P., & Storey, M. A. (2023)**. Рамкова структура для автоматизації вимірювання досліджень та оцінки показників DevOps (DORA). В *2023 IEEE International Conference on Software Maintenance and Evolution (ICSME)* (стор. 62-72). IEEE.
6. John M.M., Olsson, H.H., & Bosch J. (2021). До MLOps: рамкова структура та модель зрілості. В *2021 47th Euromicro Conference on Software Engineering and Advanced Applications (SEAA)*, IEEE, 1-8.

REFERENCES

1. **Liu, Y., Ling, Z., Huo, B., Wang, B., Chen, T., & Mouine, E.** (2020). Building A Platform for Machine Learning Operations from Open Source Frameworks. IFACPapersOnLine, 53(5), 704–709. https://doi.org/10.1016/j.ifacol.2021.04.161
2. **Granlund, T., Kopponen, A., Stirbu, V., Myllyaho, L., & Mikkonen, T.** (2021). MLOps Challenges in MultiOrganization Setup: Experiences from Two Real-World Cases. 2021 IEEE/ACM 1st Workshop on AI Engineering - Software Engineering for AI (WAIN), 82–88. https://doi.org/10.1109/WAIN52551. 2021 .00019
3. **Zhou, Y., Yu, Y., & Ding, B.** (2020). Towards MLOps: A Case Study of ML Pipeline Platform. 2020 International Conference on Artificial Intelligence and Computer Engineering (ICAICE), 494–500. https: //doi.org/10.1109/ICAICE51518.2020.00102
4. **Cardoso Silva, L., Rezende Zagatti, F., Silva Sette, B., Nildaimon dos Santos Silva, L., Lucrédio, D., Furtado Silva, D., & de Medeiros Caseli, H.** (2020). Benchmarking Machine Learning Solutions in Production. 2020 19th IEEE International Conference on Machine Learning and Applications (ICMLA), 626–633. https://doi.org/10.1109/ICMLA51294.2020.00 104
5. **Wilkes, B., Milani, A. M. P., & Storey, M. A**. (2023). A Framework for Automating the Measurement of DevOps Research and Assessment (DORA) Metrics. In 2023 IEEE International Conference on Software Maintenance and Evolution (ICSME) (pp. 62-72). IEEE.
6. **John M.M., Olsson, H.H., & Bosch J.** (2021). Towards MLOps: A framework and maturity model. In 2021 47th Euromicro Conference on Software Engineering and Advanced Applications (SEAA), IEEE, 1-8.

Method of deploying an ai platform using MLOPS methodology

Volodymyr Rusinov

Abstract. The specifics of modern AI applications require the development of a platform that enables continuous support and rapid reconfiguration. Scientific and engineering literature shows that the use of Machine Learning Operations and Machine Learning Pipeline (MLOps) methods when creating AI platforms for embedded systems can increase the efficiency of AI development and implementation. MLOps allows you to automate the development, training, and operation of machine learning models, providing a quick response to changes in the environment or input data. This helps to reduce the time for deployment and maintenance of the system as a whole.

To solve the problem of deploying an AI platform, it is proposed to use a method based on a hybrid Edge-Cloud architecture. This approach is often used in the modern scientific literature and finds application in various fields. The architecture of the platform, based on hierarchical relationships between the Edge and Cloud layers, dictates the need to create a specialized approach to platform deployment. The peculiarity of this architecture is the use of embedded systems directly in AI tasks, namely, pattern recognition. This approach allows reconfiguring the platform, subject to updating the AI model, thereby maintaining its performance and correct task execution.

Given that the key is to balance the speed of reconfiguration of such a platform with fault tolerance, in order to measure the overall performance of the system with a certain degree of accuracy, we will introduce several metrics that will help establish the parameters of the system and compare it with alternative solutions. Deployment of AI software should be accompanied by an analysis of the target system. As part of this work, a number of tests were conducted to quantify system performance.

Keywords: artificial intelligence, MLOps, IoT, fault tolerance, computer networks.