УДК 004.75:004.8

АНАЛІЗ АРХІТЕКТУРНИХ ПІДХОДІВ ДО РОЗГОРТАННЯ МОДЕЛЕЙ ШТУЧНОГО ІНТЕЛЕКТУ У ХМАРНОМУ СЕРЕДОВИЩІ

Семененко В.О.
e-mail: valerii.semenenko@nure.ua
Науковий керівник – канд. техн. наук, доц. Піскарьов Олексій Миколайович
Харківський національний університет радіоелектроніки, каф. ЕОМ
м. Харків, Україна

This paper analyzes architectural approaches to deploying artificial intelligence models in a cloud environment. The study is motivated by the growing use of intelligent services in applied systems that require stable operation, acceptable response time, scalability, and justified resource consumption. The paper outlines the main deployment scenarios, including real-time request processing, batch data processing, and continuous event processing. Container-based and serverless approaches are considered, and their advantages and limitations are compared. It is shown that the choice of an architectural solution should depend on the workload profile, operating conditions, and maintenance requirements. The obtained results can be used as a basis for further development of a generalized approach to the deployment of artificial intelligence models in cloud systems.

Сучасні інформаційні системи дедалі частіше використовують моделі штучного інтелекту як повноцінну частину робочого середовища. У таких умовах важливого значення набуває не лише якість самої моделі, а й спосіб її розгортання. Невдало організована інфраструктура може призводити до збільшення часу відповіді, нестійкої роботи сервісу та надмірних витрат на обчислювальні ресурси. Тому аналіз архітектурних підходів до розгортання моделей у хмарному середовищі є актуальним завданням сучасної комп’ютерної інженерії.

Вибір архітектурного підходу ускладнюється тим, що на практиці застосовуються різні режими використання моделей. Для одних систем вирішальною є обробка окремих запитів у реальному часі з мінімальною затримкою. Для інших більш важливим є пакетне опрацювання великих обсягів даних за розкладом. Окрему групу становлять задачі безперервного опрацювання потоку подій, де особливе значення мають стійкість роботи та передбачуваність поведінки системи під тривалим навантаженням. Це означає, що універсального рішення для всіх прикладних випадків не існує.

Одним із найуживаніших є підхід, заснований на контейнеризації та використанні засобів оркестрації. Його перевага полягає у відтворюваності налаштувань, зручності перенесення між середовищами та можливості керованого масштабування. Такий спосіб дає змогу ізолювати модель, її

залежності та параметри виконання у стандартизованій формі, а також спрощує поетапне оновлення версій. Саме тому він є доцільним для систем, у яких необхідно підтримувати стабільну роботу сервісу за змінного навантаження.

Водночас контейнеризоване розгортання потребує належного контролю використання ресурсів, спостереження за станом сервісів і налаштування автоматичного масштабування. Якщо ці механізми організовано недостатньо точно, система або не витримуватиме пікових навантажень, або використовуватиме надлишкові ресурси. Поряд із цим поширення набуває безсерверний підхід, який є доцільним за нерівномірної інтенсивності звернень до моделі. Його головна перевага полягає у можливості зменшувати витрати за відсутності запитів, однак для задач із жорсткими вимогами до часу відповіді він не завжди є оптимальним.

Для пакетного опрацювання даних вирішального значення зазвичай набувають пропускна здатність і вартість виконання, тоді як для безперервного опрацювання потоку подій особливо важливими стають стійкість, передбачуваність і здатність системи тривалий час працювати без погіршення якості обслуговування. Отже, оцінювання архітектурних підходів доцільно здійснювати не лише за швидкодією, а за сукупністю критеріїв: затримкою, пропускною здатністю, надійністю, вартістю використання ресурсів і складністю супроводу.

Не менш важливими є питання безпеки та керованості змін. Під час розгортання моделей штучного інтелекту у хмарному середовищі необхідно враховувати розмежування прав доступу, збереження конфіденційності даних, перевірку змін налаштувань і можливість швидкого повернення до попередньої версії. Саме тому подальші дослідження доцільно спрямувати на формування узагальненого підходу до вибору архітектури розгортання моделей відповідно до умов конкретної задачі.

Список використаних джерел:

1. KServe Documentation. Welcome to KServe. URL: https://kserve.github.io/website/docs/intro (дата звернення: 19.03.2026).

2. Kubernetes Documentation. Horizontal Pod Autoscaling. URL: https://kubernetes.io/docs/concepts/workloads/autoscaling/horizontal-pod-autoscale/ (дата звернення: 19.03.2026).

3. Knative Documentation. Autoscaling in Knative Serving. URL: https://knative.dev/docs/serving/autoscaling/ (дата звернення: 19.03.2026).