METHOD OF COMPREHENSIVE ASSESSMENT OF THE EFFICIENCY OF APPLICATION OF CONTENT DISPOSAL AND RECONSTRUCTION TECHNOLOGY IN NETWORK INFRASTRUCTURE

The article explores the possibilities of using Content Disarm and Reconstruction (CDR) technology to increase the effectiveness of network resource protection during processing and transfer of files of various formats. The relevance of the study is due to the rapid increase in the dangers associated with the growth of the number of hidden threats implemented through legitimate file structures and capable of bypassing traditional means of detecting malicious content, which causes a high level of complexity in their prevention or neutralization. The main attention is focused on developing a method for comprehensive experimental assessment of the effectiveness of using CDR solutions as a means of preventive countermeasures against threats that arise during the exchange of files with potentially malicious content. The work considers the basics of the content security problem and the idea of its cleaning as one of the ways to solve it, as well as the principle of operation of the content disarm and reconstruction technology, which consists in removing active and potentially dangerous elements of the file with the subsequent formation of its safe copy based on the standard structure of the corresponding file format that was processed. The features of CDR application to various types of files, in particular documents, archives, images and multimedia content, are analyzed, and its advantages compared to classical signature and behavioral protection tools, such as antiviruses and sandbox detection methods, are also identified. An experimental methodology for mass testing CDR solutions on large volumes of files is proposed with further analysis of the results and formation of a system of quantitative performance indicators that take into account the level of neutralization of hidden threats, reconstruction correctness, processing performance and impact on network and computing resources. As a result, directions for further research are identified, in particular, clarification of optimal scenarios for the use of content neutralization and reconstruction technology, formalization of efficiency criteria and development of methods for optimizing performance while maintaining high quality of file reconstruction.

Keywords: cybersecurity; information protection; network security; content disarm.

Надійшла до редакції (Received): 17.05.2026
Прийнята до друку (Accepted): 12.06.2026
Опубліковано онлайн (Available online): 25.06.2026

УДК 004.85:004.414.2 DOI: 10.31673/2409-7292.2026.027419

Яценко Дмитро Володимирович
аспірант
Державний університет інформаційно-комунікаційних технологій, Київ, Україна
ORCID: 0009-0001-2610-222X
E-mail: d.yatsenko@stud.duikt.edu.ua

Садовенко Володимир Сергійович
кандидат фізико-математичних наук, доцент
Державний університет інформаційно-комунікаційних технологій, Київ, Україна
ORCID: 0000-0001-7341-6021
E-mail: v.sadovenko@duikt.edu.ua

ПАТЕРНИ ПРОЕКТУВАННЯ АРХІТЕКТУР ДЛЯ СИСТЕМ МАШИННОГО НАВЧАННЯ

У статті здійснено розширений систематизований огляд патернів проєктування архітектур для систем машинного навчання (ML-систем), що застосовуються під час розв’язання задач предиктивної аналітики в умовах динамічних бізнес-процесів і зростаючих обсягів даних. Обґрунтовано доцільність використання архітектурних патернів як формалізованих практик, що забезпечують структурованість, масштабованість, відтворюваність експериментів і керованість життєвим циклом моделей. Запропоновано багаторівневу класифікацію патернів за етапами життєвого циклу машинного навчання (збір, очищення та підготовка даних, інженерія ознак, навчання і валідація, розгортання, моніторинг, повторне навчання) та за архітектурними рівнями системи (рівень даних, рівень моделей, прикладний та інфраструктурний рівні). Проаналізовано функціональне призначення, переваги, потенційні ризики та обмеження застосування ключових патернів, зокрема Data Pipeline,

© Яцев А.В., Журавель І.М. Метод комплексного оцінювання ефективності застосування технології знешкодження та реконструкції контенту у мережевій інфраструктурі. Сучасний захист інформації, 2(66), 210–220. https://doi.org/10.31673/2409-7292.2026.029318

Feature Store, Training–Serving Split, Model Registry, Microservices, Event-Driven Architecture, а також практик MLOps. Розкрито їхню роль у забезпеченні узгодженості даних між середовищами навчання та продакшену, управлінні версіями моделей і наборів даних, автоматизації CI/CD-процесів, організації моніторингу якості прогнозів і виявленні деградації моделей. Окрему увагу приділено питанням інтеграції ML-компонентів у корпоративні інформаційні системи, забезпеченню відмовостійкості та підтримці безперервного циклу вдосконалення моделей на основі зворотного зв'язку. Сформульовано практичні рекомендації щодо вибору та комбінування патернів з урахуванням специфіки предметної області, вимог до продуктивності, масштабів інфраструктури, регуляторних обмежень і рівня зрілості команди розробки. Отримані результати можуть бути використані під час проєктування адаптивних і масштабованих архітектур систем предиктивної аналітики різного рівня складності.

Ключові слова: машинне навчання, архітектура програмних систем, патерни проєктування, предиктивна аналітика, MLOps.

Вступ

Стрімкий розвиток технологій машинного навчання, поширення хмарних обчислень і зростання обсягів даних суттєво трансформували підходи до створення інформаційних систем, орієнтованих на предиктивну аналітику. Якщо раніше програмні рішення будувалися переважно на детермінованій логіці обробки даних, то сучасні ML-системи інтегрують процеси експериментального навчання моделей, їх адаптації до змінних умов середовища та безперервного вдосконалення на основі нових даних.

Особливістю таких систем є їхня залежність не лише від програмного коду, а й від структури, якості та динаміки даних, що обумовлює необхідність переосмислення традиційних архітектурних підходів. Архітектура ML-системи повинна враховувати повний життєвий цикл моделі − від підготовки даних і навчання до розгортання, моніторингу та повторного навчання. У цьому контексті важливого значення набуває формалізація архітектурних рішень, що забезпечують узгодженість компонентів, керованість процесів і масштабованість інфраструктури.

Одним із перспективних напрямів систематизації таких рішень є використання патернів проєктування − узагальнених моделей організації компонентів і взаємодій, що відображають накопичений практичний досвід побудови складних програмних систем. Їх адаптація до специфіки машинного навчання створює методологічну основу для формування цілісних і стійких архітектур предиктивної аналітики.

Постановка проблеми

Незважаючи на активний розвиток інструментів і платформ машинного навчання, практична реалізація систем предиктивної аналітики часто супроводжується низкою архітектурних труднощів. До них належать розрив між середовищами розробки та продакшену, відсутність узгодженого управління версіями даних і моделей, складність відтворення експериментів, а також проблеми масштабування та супроводу систем у довгостроковій перспективі.

У багатьох випадках архітектурні рішення приймаються ситуативно, без формалізованої моделі взаємодії компонентів, що призводить до накопичення технічного боргу та зниження стабільності роботи системи. Особливо гостро ці проблеми проявляються в умовах зростання обсягів даних, підвищених вимог до продуктивності та необхідності швидкої інтеграції нових моделей у бізнес-процеси. Таким чином, виникає потреба у науково обґрунтованій систематизації архітектурних патернів, що застосовуються під час проєктування ML-систем, а також у розробленні рекомендацій щодо їх раціонального вибору та комбінування залежно від типу задачі, масштабів інфраструктури й організаційного контексту. Вирішення цієї проблеми дозволить підвищити ефективність проєктування систем предиктивної аналітики та забезпечити їх стійкість до змін даних і вимог середовища.

Аналіз останніх досліджень

У сучасних наукових працях дедалі більшої актуальності набувають проблеми архітектурного проєктування систем машинного навчання, особливо в контексті побудови систем предиктивної аналітики та застосування підходів MLOps. Сучасні дослідження

https://doi.org/10.31673/2409-7292.2026.027419

наголошують на тому, що традиційні підходи до архітектури програмних систем є недостатніми для ML-систем через складність життєвого циклу моделей, залежність від даних та необхідність безперервного моніторингу якості прогнозів.

У роботі Kreuzberger, Kühl та Hirschl запропоновано узагальнений огляд концепції MLOps, у якому MLOps розглядається як архітектурна парадигма, що охоплює компоненти управління даними, моделями, інфраструктурою та експериментами [1]. Автори підкреслюють, що ефективна архітектура ML-системи повинна будуватися на основі модульних і повторно використовуваних архітектурних рішень, які фактично відповідають патернам проєктування.

Систематичний огляд Eken та співавторів охоплює як академічні, так і інженерні джерела та акцентує увагу на архітектурних практиках, що використовуються в сучасних ML-системах, зокрема на побудові конвеєрів даних, версіонуванні моделей, автоматизованому розгортанні та моніторингу [2]. У роботі зазначається, що відсутність узагальненої класифікації архітектурних патернів ускладнює масштабування та супровід систем предиктивної аналітики.

У низці сучасних досліджень також розглядаються практичні архітектурні підходи до побудови ML-конвеєрів і систем підтримки життєвого циклу моделей. Зокрема, в роботі Ганчука та Семерікова проаналізовано архітектурні рішення для автоматизації процесів навчання, тестування, розгортання та перевчання моделей, що дозволяє інтерпретувати MLOps як сукупність архітектурних патернів, орієнтованих на предиктивну аналітику [3].

Мета роботи

Запропонувати класифікацію архітектурних патернів за етапами життєвого циклу машинного навчання та за архітектурними рівнями системи, яка дозволяє структуровано описати типові архітектурні рішення та встановити логічні й технологічні взаємозв'язки між компонентами ML-систем, а також надати практичні рекомендації щодо використання й комбінування зазначених патернів для побудови ефективної, масштабованої та відтворюваної архітектури систем розв'язання задач предиктивної аналітики на основі машинного навчання.

Особливості архітектур систем машинного навчання

З метою формалізованого представлення структури та функціонування ML-системи, її можна подати як композицію відображень між відповідними множинами даних, ознак і прогнозів, що утворює функціональну модель ML-системи:

$$\hat{y} = f_{\theta(\varphi(D))},$$

де $\hat{y}$ – прогнозоване значення, D – множина сирих даних, $\varphi(D)$ – оператор формування ознак, f_θ – модель з параметрами.

У задачах предиктивної аналітики оператор формування ознак $\varphi(D)$ реалізує перетворення сирих даних у структурований набір інформативних характеристик шляхом їх очищення, агрегації та трансформації. Результатом є вектор ознак $x = \varphi(D)$, що подається на вхід моделі.

Параметри θ інтерпретуються як множина внутрішніх параметрів моделі (вагові коефіцієнти, порогові значення, матриці ваг або інші структурні характеристики), які оптимізуються в процесі навчання шляхом мінімізації функції втрат.

Архітектура ML-систем має низку специфічних рис, які суттєво відрізняють її від традиційної архітектури класичних програмних продуктів.

1. Даноцентричність. Основним ресурсом ML-систем є дані, і їх якість, повнота та актуальність безпосередньо впливають на точність моделей. Архітектура має забезпечувати збір, зберігання, очищення, трансформацію, верифікацію та моніторинг якості даних протягом життєвого циклу моделей.

2. Життєвий цикл моделі. Моделі ML проходять етапи навчання, валідації, розгортання, моніторингу та перевчання. Кожен етап потребує специфічних архітектурних рішень:

© Д.В. Яценко, В.С. Садовенко. Патерни проєктування архітектур для систем машинного навчання. Сучасний захист інформації, 2(66), 220–227.
https://doi.org/10.31673/2409-7292.2026.027419

конвеєри обробки даних, середовища тренування, інструменти тестування та відстеження продуктивності моделей.

3. Стохастичність результатів. Результати ML-систем не завжди детерміновані через варіативність даних і алгоритмів. Потрібні механізми контролю якості, валідації, логування та аудиту, а патерни повинні враховувати невизначеність прогнозів.

4. Експериментування та тестування гіпотез. Розробка передбачає активне тестування моделей, методів підготовки даних та конфігурацій алгоритмів. Архітектура має підтримувати швидкі та безпечні експерименти, зберігати результати і масштабувати компоненти без порушення роботи системи [4].

5. Інтеграція з бізнес-процесами. ML-системи взаємодіють із корпоративними базами даних, вебсервісами, аналітичними платформами та іншими IT-компонентами. Архітектура повинна забезпечувати стандартизовані API та протоколи обміну даними.

6. Масштабованість та надійність. Зростання даних і складності моделей вимагає масштабування зберігання та обробки, розподіленого навчання, балансування навантажень і відмовостійкості. Патерни мають підтримувати горизонтальне та вертикальне масштабування й автоматичне відновлення компонентів.

Ці особливості визначають специфіку архітектури ML-систем і зумовлюють застосування спеціалізованих архітектурних патернів, які поєднують модульність, гнучкість, автоматизацію та підтримку життєвого циклу моделей, забезпечуючи ефективну та надійну роботу систем розв’язання задач предиктивної аналітики.

Класифікація патернів проєктування для ML-систем

У межах дослідження пропонується класифікація патернів проєктування архітектур для систем машинного навчання на основі двох взаємодоповнювальних критеріїв: етапів життєвого циклу машинного навчання та архітектурних рівнів системи [5].

Подана класифікація (табл. 1) архітектурних патернів дозволяє комплексно описати архітектурні рішення, враховуючи як динаміку розвитку ML-моделі, так і структурну організацію системи в цілому.

Аналіз переваг та обмежень патернів проєктування

Застосування архітектурних патернів у проєктуванні систем машинного навчання є важливим інструментом підвищення їх масштабованості, керованості та надійності. Патерни дозволяють формалізувати типові архітектурні рішення, зменшити ризики проєктування та забезпечити повторне використання перевірених підходів. Водночас їх упровадження супроводжується низкою обмежень, пов’язаних зі зростанням складності системи та вимог до ресурсів.

Переваги застосування патернів. Однією з ключових переваг є підвищення масштабованості ML-систем. Такі патерни, як Microservices, Model-as-a-Service та Batch/Online Inference, дозволяють гнучко масштабувати окремі компоненти системи залежно від навантаження, що є критично важливим для систем предиктивної аналітики з великими обсягами даних. Іншою суттєвою перевагою є покращення керованості життєвого циклу моделей. Патерни Model Registry, Experiment Tracking та CI/CD for ML забезпечують контроль версій моделей, відтворюваність експериментів і автоматизацію процесів розгортання. Це зменшує ризик використання некоректних або застарілих моделей у продукційному середовищі. Застосування патернів управління даними, зокрема Data Pipeline та Feature Store, сприяє підвищенню якості даних і узгодженості ознак між етапами навчання та інференсу. Централізоване зберігання ознак зменшує дублювання, спрощує повторне використання та позитивно впливає на стабільність прогнозів [6].

Крім того, архітектурні патерни забезпечують підтримку експериментальної діяльності, що є невід’ємною складовою ML-систем. Ізоляція експериментів, автоматичний збір метрик і порівняння моделей дозволяють швидше перевіряти гіпотези та приймати обґрунтовані рішення щодо вибору моделей.

Таблиця 1

Класифікація архітектурних патернів проєктування для систем машинного навчання

Критерій класифікації	Група патернів	Назва патерну	Призначення та коротка характеристика
Етапи життєвого циклу ML	Управління даними	Data Pipeline	Послідовна обробка та передача даних між компонентами
		ETL / ELT	Витяг, перетворення та завантаження даних у сховища
		Feature Store	Централізоване зберігання та повторне використання ознак
	Навчання та експерименти	Offline Training	Навчання моделей поза продукційним середовищем
		Experiment Tracking	Фіксація параметрів і результатів експериментів
		Training–Serving Split	Розмежування середовищ навчання та використання моделей
	Розгортання та інференс	Model-as-a-Service	Надання доступу до моделі через API
		Batch Inference	Пакетна обробка великих обсягів даних
		Online Inference	Прогнозування в режимі реального часу
	Моніторинг і перевчання	Model Monitoring	Контроль якості прогнозів
		Drift Detection	Виявлення змін у розподілах даних
		Automated Retraining	Автоматичне повторне навчання моделей
Архітектурні рівні системи	Рівень даних	Data Lake	Зберігання сирих та напівструктурованих даних
		Data Warehouse	Аналітичне сховище даних
		Data Versioning	Версіонування наборів даних
	Рівень моделей	Model Registry	Керування версіями моделей
		Ensemble Models	Комбінування кількох моделей
		Champion–Challenger	Порівняння продукційної та альтернативних моделей
	Сервісний рівень	Microservices	Декомпозиція системи на сервіси
		REST / gRPC API	Стандартизований доступ до ML-сервісів
		Event-Driven Architecture	Асинхронна взаємодія компонентів
	Інфраструктурний рівень	Containerization	Ізоляція компонентів у контейнерах
		Orchestration (Kubernetes)	Керування розгортанням і масштабуванням
		CI/CD for ML	Автоматизація життєвого циклу ML

Обмеження та виклики впровадження. Попри значні переваги, застосування архітектурних патернів супроводжується низкою обмежень. Насамперед це зростання складності архітектури системи. Наприклад, використання мікросервісної архітектури призводить до необхідності впровадження додаткових механізмів оркестрації, сервісної взаємодії, моніторингу та обробки відмов. Істотним обмеженням є підвищені вимоги до обчислювальних і людських ресурсів. Патерни, орієнтовані на автоматизацію (CI/CD for ML, Automated Retraining), потребують розвиненої інфраструктури та кваліфікованих фахівців у сфері MLOps, що може бути недоцільним для невеликих проєктів [7].

Застосування Feature Store, попри зменшення дублювання ознак, вимагає чітко визначеної політики управління версіями ознак, контролю їх актуальності та узгодженості між різними моделями. За відсутності таких політик зростає ризик виникнення помилок і зниження довіри до результатів прогнозування.

Додатковим викликом є складність забезпечення інтерпретованості та прозорості системи. Комбінація кількох патернів, зокрема Ensemble Models та Event-Driven Architecture, ускладнює аналіз причинно-наслідкових зв'язків між даними, моделями та результатами прогнозів [8].

Таким чином, архітектурні патерни є ефективним інструментом побудови масштабованих і надійних ML-систем, однак їх застосування має бути контекстно обґрунтованим. Вибір конкретних патернів повинен враховувати клас задач предиктивної аналітики, вимоги до продуктивності, доступні ресурси та рівень зрілості MLOps-практик. Збалансоване поєднання патернів дозволяє мінімізувати їх обмеження та створити ефективну модель архітектури системи машинного навчання.

Рекомендації для проєктування систем предиктивної аналітики

Побудова ефективної моделі архітектури системи розв'язання задач предиктивної аналітики на основі машинного навчання потребує комплексного підходу до вибору та поєднання архітектурних патернів. На основі проведеного аналізу доцільно сформулювати такі рекомендації.

1. Доцільно поєднувати архітектурні патерни різних рівнів, зокрема рівня даних, моделей, сервісів, інфраструктури та MLOps. Таке поєднання забезпечує узгодженість між процесами підготовки даних, навчання моделей і їх експлуатації, а також сприяє формуванню цілісної архітектурної моделі системи предиктивної аналітики. Важливо, щоб патерни рівня даних (конвеєри обробки, централізовані сховища ознак) були логічно пов'язані з патернами рівня моделей (розділення навчання та інференсу, реєстри моделей), а сервісні та інфраструктурні рішення забезпечували їхню безперебійну взаємодію. Такий багаторівневий підхід мінімізує ризик виникнення «розривів» між компонентами системи, підвищує узгодженість процесів і створює основу для еволюційного розвитку архітектури без необхідності її повної перебудови.

2. Необхідно забезпечувати відтворюваність експериментів, що є критично важливою вимогою для науково обґрунтованих і промислових ML-систем. Це досягається шляхом впровадження патернів версіонування даних, ознак і моделей, а також використання засобів відстеження експериментів і централізованого реєстру моделей. Крім того, доцільно фіксувати конфігурації середовища виконання, параметри навчання та метрики оцінювання, що дозволяє здійснювати повний аудит життєвого циклу моделі. Такий підхід забезпечує прозорість процесів, спрощує порівняння альтернативних моделей, сприяє повторному використанню напрацювань і підвищує довіру до отриманих прогнозів [9].

3. Архітектура системи має бути модульною та масштабованою, що дає змогу незалежно розвивати окремі компоненти системи та адаптувати її до зростання обсягів даних і навантаження. Модульність передбачає чітке розмежування відповідальності між компонентами, що спрощує тестування, модернізацію та заміну окремих елементів без впливу на всю систему. Застосування мікросервісних та подієвих архітектурних патернів дозволяє гнучко масштабувати компоненти інференсу, навчання та обробки даних залежно від реального навантаження. Горизонтальне масштабування обчислювальних ресурсів, використання контейнеризації та оркестрації сприяють підвищенню відмовостійкості та ефективності використання інфраструктури.

4. Механізми моніторингу та автоматичного перевчання моделей слід інтегрувати з початкових етапів проєктування, а не розглядати як допоміжні компоненти. Раннє впровадження патернів моніторингу якості моделей, виявлення дрейфу даних і концептуального дрейфу, а також автоматизованого перевчання забезпечує стабільність системи в умовах змінного середовища та підвищує ефективність довгострокової експлуатації [10]. Доцільно передбачати механізми збору зворотного зв'язку, автоматичної перевірки метрик якості та тригери для повторного навчання моделей. Такий проактивний підхід дозволяє своєчасно реагувати на погіршення продуктивності моделі та підтримувати

© Д.В. Яценко, В.С. Садовенко. Патерни проєктування архітектур для систем машинного навчання. Сучасний захист інформації, 2(66), 220–227.
https://doi.org/10.31673/2409-7292.2026.027419

необхідний рівень точності прогнозів. Додатково доцільно враховувати відповідність архітектурних рішень класу задач предиктивної аналітики та організаційним умовам їх використання. Надмірно складні архітектурні патерни можуть бути економічно недоцільними для невеликих або пілотних проєктів, де пріоритетом є швидкість розробки та перевірка гіпотез. Натомість для корпоративних систем із високими вимогами до надійності, безпеки та безперервності функціонування виправданим є застосування повноцінних MLOps-підходів, автоматизованих CI/CD-конвеєрів, централізованого управління інфраструктурою та політик управління доступом. Раціональний баланс між складністю архітектури та практичними потребами організації є ключовою умовою успішного впровадження систем предиктивної аналітики.

Висновки

У статті здійснено комплексний аналіз та систематизацію патернів проєктування архітектур для систем машинного навчання, орієнтованих на розв'язання задач предиктивної аналітики в умовах зростання обсягів даних, підвищених вимог до продуктивності та необхідності безперервного вдосконалення моделей. Розглянуто специфічні особливості архітектур ML-систем, що принципово відрізняють їх від класичних програмних рішень, зокрема залежність від якості й версіонування даних, експериментальний характер розробки моделей, наявність циклів повторного навчання та необхідність постійного моніторингу продуктивності в реальному середовищі експлуатації.

Запропонована класифікація архітектурних патернів забезпечує цілісне бачення архітектури, сприяє зменшенню фрагментарності рішень і створює основу для формалізованого вибору патернів залежно від функціональних і нефункціональних вимог, масштабів системи, типу задачі та рівня зрілості MLOps-практик у команді розробки.

У роботі також проаналізовано переваги, обмеження та потенційні ризики застосування основних груп патернів, що дало змогу сформулювати практичні рекомендації щодо їх комбінування та адаптації під конкретні сценарії використання. Показано, що ефективна архітектура ML-системи повинна базуватися на поєднанні патернів різних рівнів, забезпечувати відтворюваність експериментів і контроль версій даних та моделей, підтримувати горизонтальне та вертикальне масштабування, а також включати механізми автоматизованого моніторингу, виявлення деградації якості прогнозів і організації безперервного циклу перевчання моделей. Запропоновані узагальнення можуть слугувати методологічною основою для проєктування адаптивних і надійних систем предиктивної аналітики різного рівня складності.

Перелік посилань

1. Kreuzberger D., Kühl N., Hirschl S. Machine Learning Operations (MLOps): Overview, Definition, and Architecture // IEEE Access. 2023. Vol. 11. P. 12345–12359. https://doi.org/10.1109/ACCESS.2023.3280048

2. Eken B., Pallewatta S., Tran N. K., Tosun A., Babar M. A. A Multivocal Review of MLOps Practices, Challenges and Open Issues // arXiv. 2024. arXiv:2406.09737. https://arxiv.org/abs/2406.09737

3. Ганчук Д. О., Семеріков С. О. Інженерія MLOps: метасинтез інструментів, практик та архітектур для автоматизації машинного навчання // Ukrainian Journal of Information Systems and Data Science. 2024. No. 3(14). P. 45–62. https://jujisds.donnu.edu.ua/article/view/17343

4. Heiland L., Hauser M., Bogner J. Design Patterns for AI-based Systems: A Multivocal Literature Review and Pattern Repository // Proceedings of the IEEE/ACM International Conference on AI Engineering (CAIN). 2023. P. 184–196. https://doi.org/10.1109/CAIN58948.2023.00029

5. Amou Najafabadi F., Bogner J., Gerostathopoulos I., Lago P. An Analysis of MLOps Architectures: A Systematic Mapping Study // European Conference on Software Architecture (ECSA). 2024. P. 69–85. https://doi.org/10.1007/978-3-031-70797-1_5

6. Zarour M., Alzabut H., Al-Sarayreh K. T. MLOps Best Practices, Challenges and Maturity Models: A Systematic Literature Review // Information and Software Technology. 2025. Vol. 170. Article 107733. https://doi.org/10.1016/j.infsof.2024.107733

7. Microsoft Azure Architecture Center. Machine Learning Operations (MLOps): Architectural Patterns and Pipelines // Microsoft Learn. 2024. [Electronic resource]. https://learn.microsoft.com/azure/architecture/ai-ml/guide/mlops-technical-paper

8. Sculley D., Holt G., Golovin D. et al. Hidden Technical Debt in Machine Learning Systems // Advances in Neural Information Processing Systems (NeurIPS). 2015. Vol. 28. P. 2503–2511.

9. Breck E., Cai S., Nielsen E., Salib M., Sculley D. The ML Test Score: A Rubric for ML Production Readiness and Technical Debt Reduction // Proceedings of the IEEE International Conference on Big Data. 2017. P. 1123–1132.

10. Washizaki H., Uchida H., Khomh F., Gueheneuc Y.-G. Studying Software Engineering Patterns for Machine Learning Systems // Journal of Systems and Software. 2022. Vol. 188. Article 111239. https://doi.org/10.1016/j.jss.2022.111239

Dmytro Yatsenko
Postgraduate
State University of Information and Communication Technologies, Kyiv, Ukraine
ORCID: 0009-0001-2610-222X
E-mail: d.yatsenko@stud.duikt.edu.ua

Volodymyr Sadovenko
Candidate of Physical and Mathematical Sciences, Associate Professor
State University of Information and Communication Technologies, Kyiv, Ukraine
ORCID: 0000-0001-7341-6021
E-mail: v.sadovenko@duikt.edu.ua

ARCHITECTURE DESIGN PATTERNS FOR MACHINE LEARNING SYSTEMS

The article provides an extensive systematic review of architectural design patterns for machine learning systems (ML systems) used in solving predictive analytics problems in dynamic business processes and growing data volumes. The feasibility of using architectural patterns as formalized practices that ensure structuring, scalability, reproducibility of experiments, and model lifecycle management is substantiated. A multi-level classification of patterns is proposed by the stages of the machine learning lifecycle (data collection, cleaning, and preparation, feature engineering, training and validation, deployment, monitoring, retraining) and by system architectural levels (data layer, model layer, application and infrastructure layers). The functional purpose, benefits, potential risks, and limitations of the application of key patterns are analyzed, in particular, Data Pipeline, Feature Store, Training–Serving Split, Model Registry, Microservices, Event-Driven Architecture, as well as MLOps practices. Their role in ensuring data consistency between training and production environments, managing model and dataset versions, automating CI/CD processes, organizing forecast quality monitoring, and detecting model degradation is revealed. Special attention is paid to the issues of integrating ML components into corporate information systems, ensuring fault tolerance, and supporting a continuous feedback-based model improvement cycle. Practical recommendations are formulated for selecting and combining patterns, taking into account the specifics of the subject area, performance requirements, infrastructure scale, regulatory constraints, and the level of maturity of the development team. The results obtained can be used when designing adaptive and scalable architectures of predictive analytics systems of various levels of complexity.

Keywords: machine learning, software systems architecture, design patterns, predictive analytics, MLOps.

Надійшла до редакції (Received): 28.05.2026
Прийнята до друку (Accepted): 12.06.2026
Опубліковано онлайн (Available online): 25.06.2026

© Д.В. Яценко, В.С. Садовенко. Патерни проєктування архітектур для систем машинного навчання. Сучасний захист інформації, 2(66), 220–227.
https://doi.org/10.31673/2409-7292.2026.027419