УДК 004.8 DOI 10.18372/2073-4751.87.21680

Волокита А. М., к.т.н.,
orcid.org/0000-0001-9069-5544,
e-mail: artem.volokita@kpi.ua,
Зоткін К. В.,
orcid.org/0009-0001-2320-9021,
e-mail: zotkin.kv@pm.me

АНАЛІЗ МЕТОДІВ ПРОГНОЗУВАННЯ ВІДМОВ У ПОДІЙНО-ОРІЄНТОВАНІЙ АРХІТЕКТУРІ

**Національний технічний університет України
"Київський політехнічний інститут імені Ігоря Сікорського"**

Вступ

Відмовостійкість розподілених [1] систем є критично важливою для забезпечення безперервної роботи сучасних IT-інфраструктур, особливо в умовах підвищення складності задач, високого навантаження на системи, та постійної модифікації програмного забезпечення. Традиційні методи управління відмовами базуються на просторовій та часовій надлишковості, потребують значних ресурсів для оперативного відновлення системи, аналізу причин відмов та прийняття рішень щодо реконфігурації, що в свою чергу знижує швидкість реакції та ефективність всієї системи.

В сучасних високонавантажених розподілених системах все частіше використовується подійно-орієнтована архітектура з окремим сервісом оркестрації подій на який покладається функція керування робочими процесами а також попередження і запобігання відмов. Традиційні методи моніторингу, які ґрунтуються на фіксованих порогах або окремих подіях, не здатні своєчасно виявити приховані закономірності, що призводять до відмов. Внаслідок цього виникають ситуації, коли система реагує вже після факту відмови.

Мета дослідження

Метою дослідження є аналіз існуючих методів прогнозування відмов у подійно-орієнтованій архітектурі та емперичне доведення переваги підходу на основі векторного пошуку як фундаменту для прогнозування на потоках з динамічним складом подій.

Постановка проблеми

Розглянемо задачу прогнозування відмов як задачу аналізу неперервного потоку подій, що надходять до системи керування подійно-орієнтованою архітектурою. Кожна подія несе інформацію про стан компонентів і характеризується описом, часовою міткою та ознакою помилковості. На основі історії спостережень потрібно не лише оцінити ймовірність майбутньої відмови, а й сформувати рекомендовані дії для запобігання їй чи мінімізації наслідків.

Нехай потік подій, що спостерігаються у часовому проміжку $[0, T]$, позначено як

$$E = \{e_1, e_2, \dots, e_t, \dots\} \ (1),$$

де кожна подія має структуру

$$e_t = (x_t, \tau_t, \delta_t), t = 1, \dots, T \ (2)$$

- x_t- опис події (категорія, параметри, контекст);
- τ_t- мітка часу події.
- $\delta_t \in \{0,1\}$- індикатор помилки (1, якщо подія призвела до відмови або помилки, 0 - інакше).

Метою є побудова функції:

$$f: E_{1:t} \to (p_t, \pi_t) \ (3),$$

яка для поточної послідовності подій одночасно визначає:

- $p_t = P(E_{1:t})$ – ймовірність майбутньої відмови,

• $\pi_t = \pi(E_{1:t}, p_t)$ — політику реагування, тобто множину рекомендованих дій для мінімізації наслідків відмови.

Політика реагування π_t— це відображення, що задає дію a_tіз множини можливих дій A залежно від поточного стану та ризику:

$$\pi_t: (E_{1:t}, p_t) \to a_t, a_t \in A \quad (4)$$

Виділення недосліджених частин загальної проблеми

Попри значну кількість досліджень у галузі виявлення аномалій у логах розподілених систем, кілька важливих питань залишаються відкритими. По-перше, більшість робіт зосереджена лише на точності виявлення на фіксованому наборі даних, тоді як експлуатаційні характеристики такі як вартість оновлення моделі, затримка прогнозу, розмір артефакту для розгортання майже не аналізуються. Через це порівняння методів є неповним через те, що модель із вищою точністю на бенчмарку може виявитися непридатною на практиці через надто дороге перенавчання чи високу затримку.

Огляд існуючих підходів

Усі методи прогнозування відмов у подійно-орієнтованих системах можна класифікувати за принципом обробки послідовності подій та способом моделювання стану системи:

- детерміновані підходи
- імовірнісні моделі станів,
- класичні статистичні методи та ML,
- послідовнісні нейронні моделі,
- векторно-аналогові методи за схожістю

Детерміновані моделі засновані на заздалегідь узгоджених правилах. Формалізований підхід до опису правил закладено в методі «Оброблення складних поді» або Complex Event Processing (CEP) [2] у подійно-орієнтованій архітектурі. Метод, заснований на початку 1990х років, дозволяє безперервно аналізувати потоки подій в реальному часі та реагувати на певні визначені комбінації або послідовності подій. Рис. 1. демонструє типові компонентами в CEP-орієнтованих систем: Рушій який виконує обробку послідовності подій, джерело подій (Event Source) і Тека правил обробки яка використовується рушієм обробки подій.

Типове правило має вигляд:

$$IF\ (A \wedge B \wedge \neg C) within\ T \Rightarrow Alert: "failure", (5),$$

де A, B, C— події певних типів, а T— часовий інтервал їхньої появи.

Метод, заснований на обробці складних подій (CPA), характеризується прозорістю результатів і простотою їх пояснення – кожна операція і умова чітко детермінована в правилах переходу, за рахунок детермінованого алгоритмічного підходу час реакції таких систем наближений до миттєвого, що підходить для систем чуттєвих до затримок у прийнятті рішень (наприклад «збільшити ресурси сервісу якщо з'являєтеся подія про високе використання CPU).

Попри простоту та ефективність у відносно статичних подійно-оріеєнтованих середовищах, CEP методи мають низку істотних обмежень [3] для систем з великою кількість подій і можливих комбінацій подій за рахунок того що вони вимагають ручного визначення правил. Детермінований аналіз і визначення правил може потребувати значних ресурсних затрат у системах де сутність або процес може включати в себе десятки і сотні подій, кожна з яких мати велику кількість атрибутів. Також такі системи не здатні виявляти нетипові комбінації які не були описані в правилах і не мають механізму узагальнення.

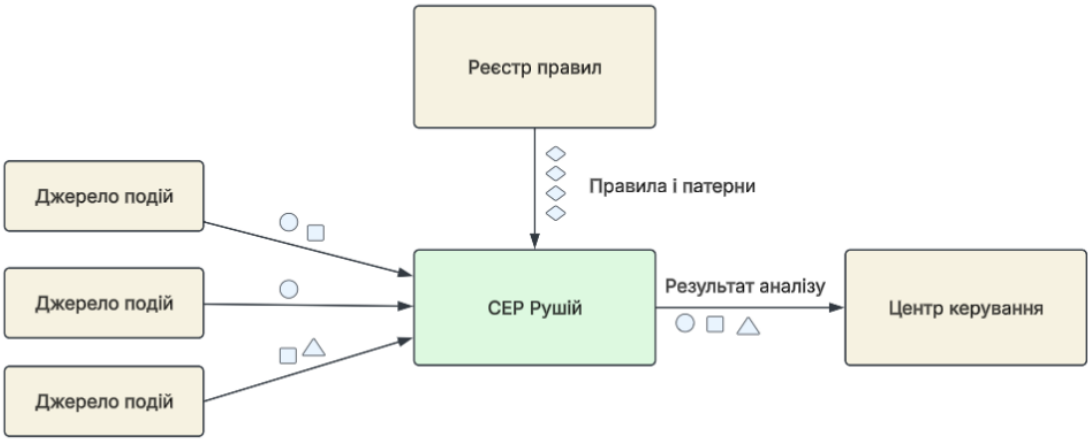


Рис. 1. Концептуальна модель CEP-орієнтованої архітектури

Таким чином, системи побудовані на детермінованих підходах та CEP-методах, забезпечують прийнятний час реакції, використання ресурсів і швидкість роботи але не є ефективними для задач прогнозування майбутніх, раніше не спостережених відмов.

Імовірнісні моделі станів [4] ґрунтуються на припущенні, що поведінка системи може бути описана як послідовність прихованих або явних станів, між якими відбуваються стохастичні переходи з певними ймовірностями. Кожна подія у потоці спостережень розглядається як наслідок переходу системи між станами, а завдання прогнозування полягає у визначенні ймовірності того, що поточна або майбутня послідовність подій призведе до відмови.

Найвідомішими представниками цього класу є ланцюги Маркова (Markov Chains) [5], приховані Марківські моделі (Hidden Markov Models, HMM) [6] Їх спільна риса – опис динаміки системи через ймовірнісні залежності між станами та подіями.

Розглянемо ланцюги Маркова як найбільш ширшу модель.

Модель описує послідовність станів системи

$$S = \{s_1, s_2, \dots, s_N\} \quad (6)$$

де ймовірність переходу у наступний стан залежить лише від поточного:

$$P(s_{t+1} \mid s_t, s_{t-1}, \dots, s_1) = P(s_{t+1} \mid s_t) \quad (7)$$

Перехідна матриця $A = [a_{ij}]$, де $a_{ij} = P(s_{t+1} = s_j \mid s_t = s_i)$, визначає усі можливі зміни станів.

Ймовірність перебування системи у стані, що відповідає відмові, оцінюється за накопиченим розподілом:

$$P(failure\ at\ t+k) = \sum_i P(s_{t+k} = s_i \mid s_t) \cdot I_{failure}(s_i) \quad (8)$$

де $I_{failure}(s_i)$— індикатор відмовного стану.

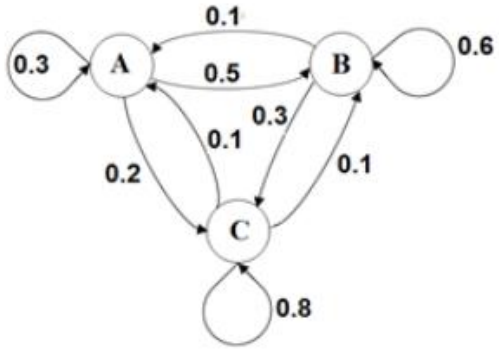


Рис. 2. Граф переходів однорідного ланцюга Маркова

Імовірнісні моделі станів широко застосовуються в індустріальних системах для моделювання процесів зносу обладнання, виявлення аномалій у мережевому трафіку [7] та аналізу ланцюгів транзакцій у фінансових установах [8]. Такі моделі дають можливість формально описати поведінку системи і добре підходять для систем із

чітко визначеними станами деградації або режиму роботи.

Суттєвими обмеженнями таких моделей є експоненційне зростання складності станів в залежності від кількості атрибутів і подій [9], а також вимога до попереднього визначення множин станів , що значно ускладнює використання цих моделей у системах, описаних в постановці задачі. Також імовірнісні моделі станів вимагають оновлення розподілу ймовірностей після кожної нової події що ускладнює застосування їх у потоковій обробці подій у реальному часі.

Статистичні та класичні ML-методи [10] прогнозування відмов базуються на принципі апроксимації явища чи процесу. Об'єктом моделювання витупає статистична сукупність, в якій реалізується закономірність.

Типовим прикладом статистичних методів прогнозування який відповідав би постановці задачі є дерева рішень, такі як Random Forest [11] або Gradient Boosting [12]. Ці методи будують модель основану на великій кількості дерев рішень – структур, що послідовно ділять простір ознак на підмножини.

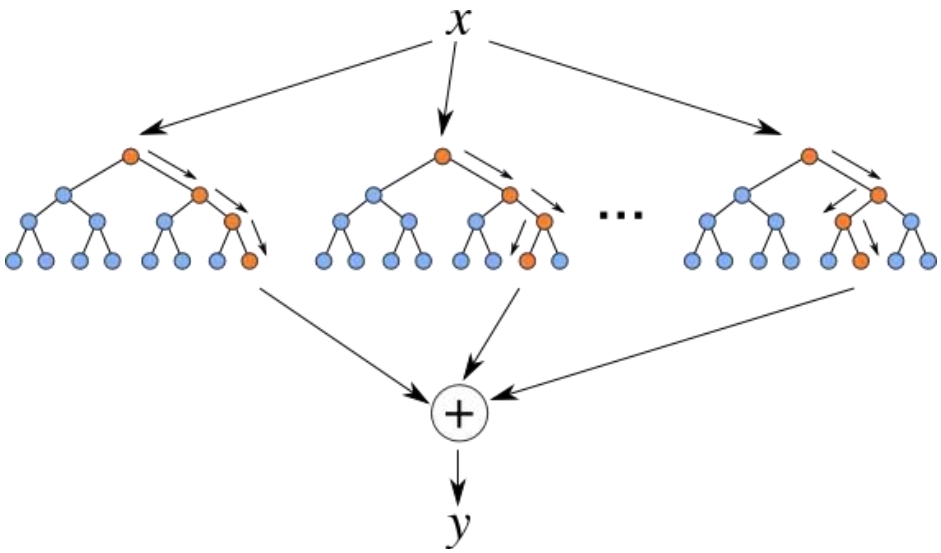


Рис. 3. Візуалізація дерева рішень

Формально одне дерево реалізує функцію:

$$f(x) = \sum_{l=1}^{L} w_l \, I(x \in R_l) \quad (9)$$

де R_l— область простору ознак, а w_l— прогноз у цій області.

- Random Forest — ансамбль незалежних дерев, що зменшує дисперсію прогнозу.
- Gradient Boosting (XGBoost, LightGBM) — послідовно навчає дерева, кожне з яких мінімізує помилку попереднього.

Іншим статистичним методом для прогнозування відмов є Опорні векторні машини *(SVM)* [13]. Цей метод є одним із найпопулярніших типів класифікторів в машиному навчанні. Машини опорних векторів працюють, малюючи межі рішень між точками даних, прагнучи до межі рішень, яка найкраще розділяє точки даних на класи (або є найбільш узагальненою). Мета під час використання машини опорних векторів полягає в тому, щоб межа рішення між точками була якомога більшою, щоб відстань між будь-якою заданою точкою даних і лінією кордону була максимальною.

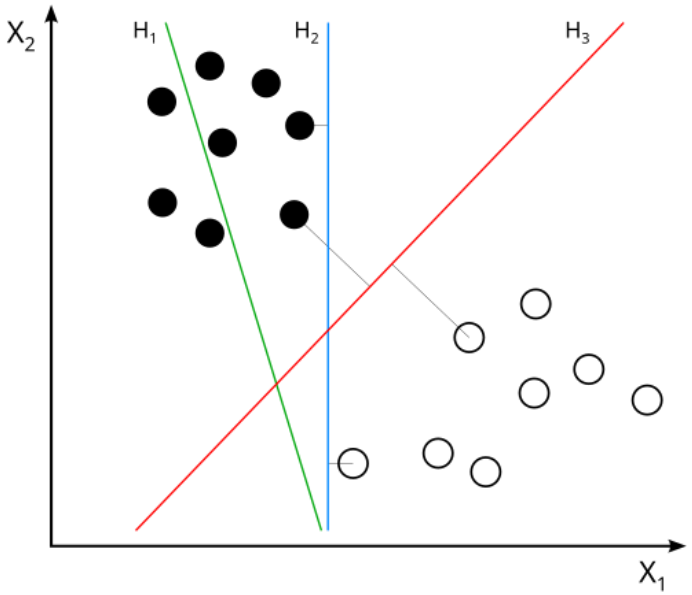


Рис. 4. Принцип роботи опорної векторної машини з лінійним ядром

Мета SVM — знайти гіперплощину, яка з максимально можливим зазором розділяє класи “норма” та “відмова”:

$$min_{w,b}\frac{1}{2}\parallel w \parallel^2 \textit{ за умови} y_i(w^\top \phi(x_i)+b) \geq 1 \#(10)$$

де $\phi(x)$— відображення у вищий простір ознак через ядро (kernel).

Цей метод має високу точність та ефективність, особливо при наявності нелінійних меж між класами. Однак потребують попередньої нормалізації ознак перед навчанням і постійного перенавчання при додаванні додаткових ознак події.

Загалом статистичні та класичні методи машиного навчання залишаются ефективним інструментом прогназування на відомих потоках подій і, найголовніше, ретроспективного аналізу відмов. Проте вони оперують фіксованими векторами ознак, які описують систему у конкретний момент часу (припускають стаціонарність процесу) та погано масштабуются на тисячі різних подій через велику розмірність векторів, що робить модель непрактичною для систем реального часу.

Проблеми з аналізом часової послідовності в базових методах статистичного аналізу вирішуються за допомогою розширених методов такі як ARIMA [14] та Фільтр Калмана [15]. Однак всі вони мають суттєвий недолік для розподілених систем реального часу – це відсутність інкрементального навчання. Для оновлення моделі при надходжені нових подій або атрибутів в існуючих подіях необхідно повністью перенавчати всю модель з нуля. У випадку з ансамблевими методами – це повне перебудова дерева або проходження бустінггово циклу. Для опорно векторної машини – це нове обчислення гіперплощини розділення. Складність таких операцій зазвичай лінійна або квадратична відносно кількості подій та атрибутів.

Послідовніні нейронні моделі [16] являють собою клас методів глибокого навчання, спроектованих для опрацювання впорядкованих послідоностей даних. На відміну від класичних ML-методів, що перетворюють вікно подій на фіксованих вектор ознак, ці методи явно враховують контекст в якому вони працюють.

Формальний опис можелі можна записати в вигляді послідовність ідентифікаторів подій:

$$\boldsymbol{i_1, i_2, \dots, i_T, i_t} \in \{\mathbf{1}, \dots, |\boldsymbol{V}|\} \quad (11)$$

де $\boldsymbol{V}$ - фіксований словник типів подій, визначений під час навчання. Кожен ідентифікатор відображається у щільний вектор через матрицю ембедингів:

$$v_t = W_e \cdot one - hot(i_t), W_e \in R^{d \times |V|} \quad (12)$$

Рекурентний шар LSTM [17] обчислює прихований стан h_t, що акумулює контекст усієї попередньої послідовності:

$$h_t = LSTM(h_{t-1}, v_t) \quad (13)$$

де внутрішні вентилі (forget, input, output) регулюють, яку частину попереднього контексту зберегти, а яку відкинути. Вихідний прогноз — ймовірність відмови у поточному вікні:

$$p_t = \sigma(W_{out} \cdot h_T + b) \quad (14)$$

Таким чином, послідовнісні нейронні моделі реалізують частину цільової функції $f: E_{1:t} \to (p_t, \pi_t)$ — а саме прогноз імовірності p_t — не реалізуючи політику реагування π_t.

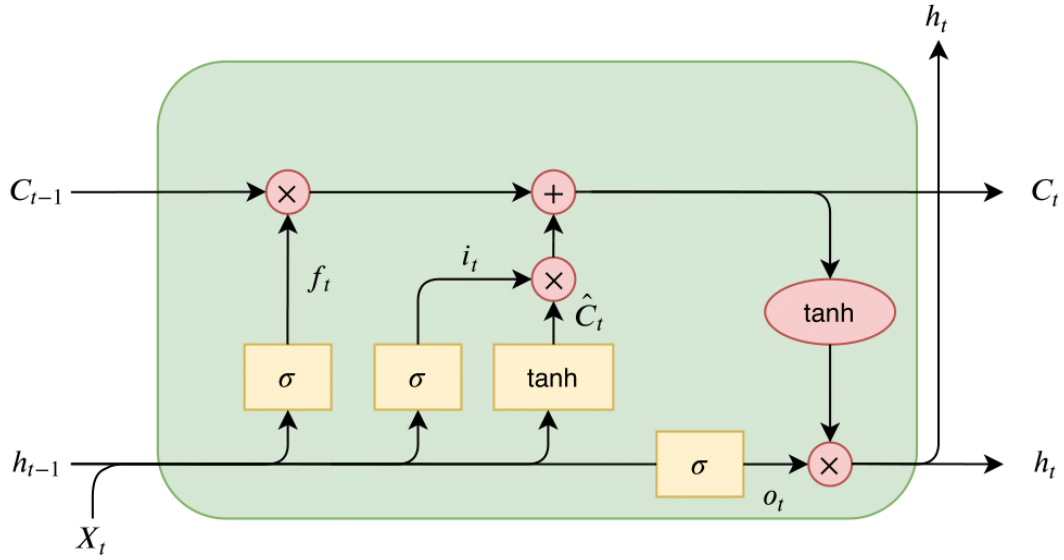


Рис. 5. Архітектура LSTM-осередку

Перевагою LSTM є можливість утримувати релевантний контекст на десятки кроків та на відміну він детермінованих і йомовірностних методів не вимагають ручного формулювання правил та визначення дискретного простору станів.

Однак попри продемонстровану ефективність на відомих потоках подій, послідовнісні нейроні моделі мають суттєві обмеження в контексті поставленох задчі і ключвою є залежність від фіксованого словника $\boldsymbol{V}$ і нові події або дані в подіях вимагають повного перенавчання. Нейронні послідовності подій також не реалізують політику реагування – вихід моделі є скалярна ймовірність відмови, але вона не містить пояснень причин чи рекомендації щодо дій.

Послідовнісні нейронні моделі усувають потребу в ручному описі правил і станів та знадні виявляти приховані часові залежності – проте прив'язані до фіксованого словника подій. Експериментально буде доведено на скільки сильно це впливає.

Векторно-аналогові методи за схожістю [18] ґрунтуються на принципі, що схожі послідовності подій ведуть до схожих наслідків: якщо поточне вікно подій семантично близьке до вікон, що передували відомим відмовам, це є підставою для прогнозу відмови.

Для формального опису метода задамо вікно подій $w = (e_{t-T}, \dots, e_t)$ яку відображається у щільний вектор за допомогою функції кодування φ:

$$v = \varphi(w), v \in R^d \quad (15)$$

де φ — попередньо навчений або загальний енкодер (наприклад, sentence-transformer [x]), що приймає на вхід текстове представлення вікна. Схожість двох вікон оцінюється через косинусну міру: $sim(\boldsymbol{v_q}, \boldsymbol{v_i}) = \frac{\boldsymbol{v_q} \cdot \boldsymbol{v_i}}{\|\boldsymbol{v_q}\| \cdot \|\boldsymbol{v_i}\|}$ (16)

Прогноз ймовірності відмови для запитового вікна $\boldsymbol{w_q}$ визначається через $\boldsymbol{k}$ найближчих сусідів у індексі $\boldsymbol{I}$, зважених за схожістю:

$$\boldsymbol{p_t} = \frac{\sum_{\boldsymbol{i}\in \boldsymbol{N_k}(\boldsymbol{v_q})} \boldsymbol{sim}(\boldsymbol{v_q}, \boldsymbol{v_i}) \cdot \boldsymbol{\delta_i}}{\sum_{\boldsymbol{i}\in \boldsymbol{N_k}(\boldsymbol{v_q})} \boldsymbol{sim}(\boldsymbol{v_q}, \boldsymbol{v_i})} \quad (17)$$

де $\delta_i \in \{0, 1\}$ — мітка відмови $\boldsymbol{i}$-го сусіда, $\boldsymbol{N_k}(\boldsymbol{v_q})$ — множина $\boldsymbol{k}$ векторів індексу з найвищою косинусною схожістю до $\boldsymbol{v_q}$.

Ключовою властивість цього підходу є можливість оброблювати нові типи подій або даних в події без повного перенавчання – достатньо додати нові вектори для аналізу

Операція вставки у FAISS/HNSW-індекс [19] має складність $\boldsymbol{O}(\boldsymbol{d})$ або $\boldsymbol{O}(\boldsymbol{d log\, N})$ — на порядки дешевше, ніж повне перенавчання RF ($\boldsymbol{O}(\boldsymbol{N} \cdot | \boldsymbol{F} | \cdot \boldsymbol{log\, N})$) або LSTM ($\boldsymbol{O}(\boldsymbol{N} \cdot \boldsymbol{T} \cdot \boldsymbol{d^2})$) на тій самій вибірці.

Данний метод природно масштабується на велику кількість типів подій і атрибутів в той час як розмірність простору $\boldsymbol{d}$ залишається фіксованою незалежно від зростання словника подій. Інкрементальне оновлення дозволяє інтегрувати нові інциденти у базу знань у реальному часі без зупинки системи.

Результати експериментів

Проведений аналіз різних класів методів прогнозування відмов викладено в Таблиці 1 і проранжовано по критеріям, впливаючім на постановку задачі Проведений аналіз показує, що моделі які забезпечують виконання поставленної задачі – це класичні ML моделі, послідовністі нейронні моделі і векторно-аналоговий підхід. Емперичним шляхов проведемо перевірку впливу нових подій на деградацію якості прогнозування моделі.

Таблиця 1. Фрагмент порівняльного аналізу вибору лідера

Критерій	**Детерміновані**	**Ймовірнісні**	**Класичні ML**	**Послідовні сні нейронні моделі**	**Векторно-аналогові**
Низький час реакції	так	частково	Так	частково	так
Інкрементальне навчання (нові події без перенавчання)	Ні	Ні	Ні	Ні	так
Масштаб на велику кількість типів подій / атрибутів	Ні	Ні	Частково	Частково	так
Виявлення нових / нетипових комбінацій	Ні	Ні	Частково	Частково	Так
Виявлення неявних залежностей між подіями	Ні	Частково	Частково	Так	Частково
Пояснюваність прогнозу	Так	Частково	Частково	Ні	Так
Видача політики реагування π_t (рекомендовані дії)	Частково	Ні	Ні	Ні	Так

Основне припущення, яке висувається на даному етапі, полягає в тому, що деградація у векторно-аналогового метода буде значно менша ніж у методів з фіксованими словниками (Random Forest, LSTM). Ключовою метрикою вимірювання виберемо Area Under the Precision-Recall Curve (PR-AUC) [20], як найкращий показник при дисбалансі класів з можливістю задати інтерпретованої точки відліку.

Формально гіпотезу для експерименту можна описати формулою

$$\boldsymbol{\Delta PR - AUC_{Retrieval}} < \boldsymbol{\Delta PR - AUC_{RF}} \quad (18)$$

де $\Delta PR - AUC = PR - AUC(\mathbf{0}\%\ нових) - PR - AUC(\boldsymbol{k}\%\ нових)$.

Для емперичної перевірки був вибраний публічний датасет BGL (Blue Gene/L) із репозиторію LogHub. Цей датасет є лог-журналом суперкомп'ютера Lawrence Livermore National Laboratory. Вибір BGL обумовлений великим словниеом шаблонів подій і наявності аномальних сигналів що присутні на рівні послідовності подій. Такий набір даних дозволяє перевірити поведінку моделей.

Таблиця 2. Параметри тестового датасету BGL

Параметр	**Значення**
Загальна кількість рядків логу	4 747 959
Унікальних шаблонів подій	353
Нормальних рядків	4 399 499 (92.66%)
Аномальних рядків	348 460 (7.34%)
Унікальних типів аномалій	41
Розмір вікна	20 подій, крок 10

Рис. 6 демонструє часовий розподіл подій в датасеті BGL. Аномалії не розподілені рівномірно, а групуються у виражені сплески.

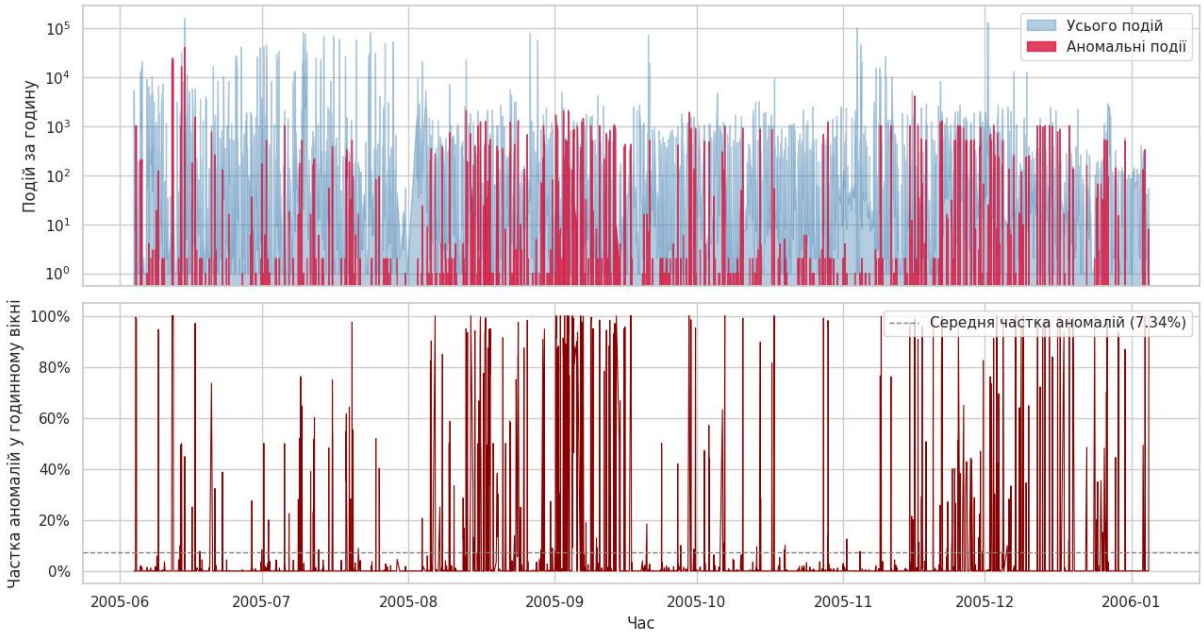


Рис. 6. Часовий розподіл подій в BGL

Для реалізації експерименту поділимо множину усіх 353 типів подій V на відомі при навчанні V_{train} та нові типи, відсутні при навчанні V_{new}. Тренувальна вибірка включала лише вікна, де всі події $\in \boldsymbol{V_{train}}$; тестова включала вікна з $\geq \mathbf{1}$ подією з $\boldsymbol{V_{new}}$. Перевірялись частки: 0%, 10%, 20%, 30% нових типів, відповідно ~37, 75 і 112 типів. Для базової точки (0% нових типів) застосовувався стандартний стратифікований спліт із резервуванням 30% вікон під тестування, що дозволяє оцінити максимально досяжну якість моделі за умови повного збігу словників навчання і тестування. Для кожної частки нових типів виконувалося 7 незалежних

запусків із різними значеннями ініціалізації генератора псевдовипадкових чисел, що забезпечує усереднення ефекту конкретного вибору підмножини $\boldsymbol{V_{new}}$.

Для порівняння було обрано по одному представнику з трьох розглянутих у попередніх розділах класів методів.

Random Forest був вибраний як представник класичних статистичних методів машиного навчання. Кожне вікно подій кодується як вектор частот появи кожного з відомих типів подій.

LSTM представляє клас послідовних нейронних моделей. Послідовність ідентифікаторів подій вікна передається через шар векторних представлень (embedding layer), що відображає кожен тип події у щільний вектор у просторі $\boldsymbol{R^d}$. Невідомі типи з $\boldsymbol{V_{new}}$ відображаються на спеціальний токен «невідомо» (UNK) з нульовою або випадково ініціалізованою позицією у просторі ембедингів.

Retrieval представляє клас векторно-аналогових методів. Кожне вікно перетворюється на текстове представлення шляхом конкатенації шаблонів подій (EventTemplate), розділених спеціальним токеном-роздільником, після чого кодується попередньо навченою моделлю sentence-transformer all-MiniLM-L6-v2 у вектор фіксованої розмірності. Прогноз для тестового вікна обчислюється як зважене за схожістю голосування $\boldsymbol{k = 5}$ найближчих сусідів у FAISS-індексі тренувальних векторів за косинусною мірою. Ключова перевага підходу полягає у тому, що нові типи подій з $\boldsymbol{V_{new}}$ мають визначений текстовий шаблон і кодуються в той самий векторний простір без будь-якого перенавчання енкодера.

Таблиця 3. Результати експерименту при різних частках нових подій

Модель	**0%**	**10%**	**20%**	**30%**	$\Delta(0\%\rightarrow30\%)$
Random Forest	1.000 ± 0.000	0.625 ± 0.291	0.473 ± 0.313	0.456 ± 0.197	0.544
LSTM	1.000 ± 0.001	0.533 ± 0.318	0.475 ± 0.301	0.461 ± 0.170	0.539
Retrieval	0.999 ± 0.000	0.561 ± 0.247	0.536 ± 0.257	0.514 ± 0.222	0.486

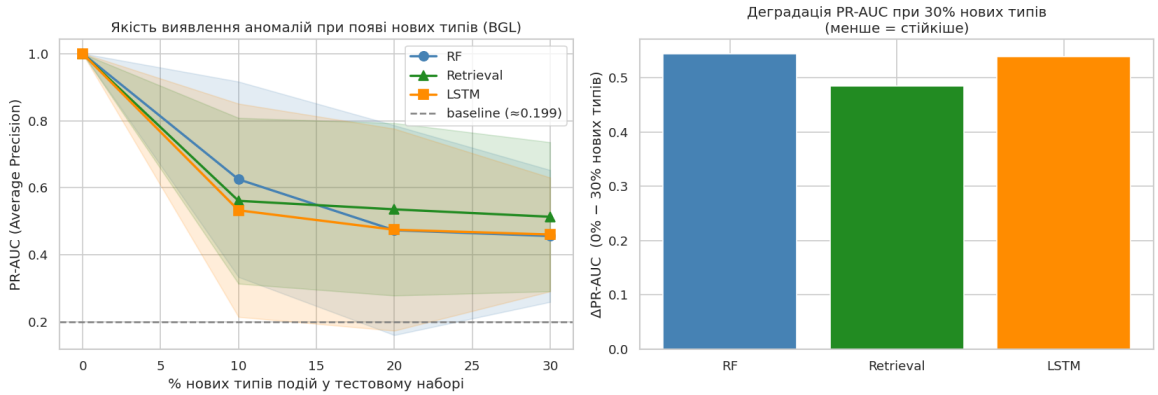


Рис. 7. Деградація PR-AUC при зростанні частки нових типів подій

За відсутності нових типів подій (0% частка) усі три моделі досягають близького до максимального значення PR-AUC ≈ 1.0, що підтверджує наявність достатнього дискримінуючого сигналу у представленні EventTemplate датасету BGL. При зростанні частки нових типів до 30% (112 типів зі 353) Random Forest і LSTM падають до близьких значень PR-AUC ≈ 0.46, тоді як Retrieval зберігає істотно вищий рівень якості — PR-AUC = 0.514. Ця асиметрія підтверджує

структурне обмеження методів з фіксованим словником, тоді як семантичне кодування в векторно-аналоговому підході зберігає інформативність представлення для нових типів.

Хоча у відсотках різниця у деградації на BGL датасеті не є значною (~11% менше падіння), це очікуваний наслідок природи стрес-тесту, коли нові події з'являються один раз. На більш довготривалому періоді часу ця різниця буде збільшуватись за рахунок автоматичного донавчання Retrieval моделі. У цій моделі кожен новий тип кодується семантично через текст шаблону. Наприклад подія "L2 cache parity error" отримує вектор, близький до відомого вектора "data cache parity error", навіть якщо її ідентифікатор моделі ніколи не зустрічався, в той час як в інших моделях ці події вимушено отримують OOV/UNK токен і модель повністю втрачає інформацію про подію.
Іншим ключовим показником є швидкість перенавчання моделі і її схильність швидко адаптуватись до нових типів події і параметрів. Виходячи з аналізу існуючих підходів, висувається гіпотеза що час і обчислювальні ресурси на додавання нового типу подій у векторний індекс на порядок менші, ніж на повне перенавчання статистичних ML моделей або LSTM.
Для емпіричного доведення гіпотези виберемо датасет OpenStack Nova Logs із відкритої колекції LogHub (проєкт LogPAI). Логи зібрано на тестовому стенді під час реальної експлуатації хмарної платформи, переважно з сервісу керування обчислювальними інстансами Nova, що відповідає за життєвий цикл віртуальних машин (створення, запуск, видалення). Вибір іншого дата сету обумовлений меншим розміром із збереженням важливих властивостей самих даний, що дозволить провести повний цикл вимірювання у розумний час.

Таблиця 4. Параметри тестового датасету OpenStack

Параметр	**Значення**
Загальна кількість рядків логу	207 636
Унікальних шаблонів подій	44
Нормальних рядків	189 202 (91.12%)
Аномальних рядків	18 434 (8.88%)

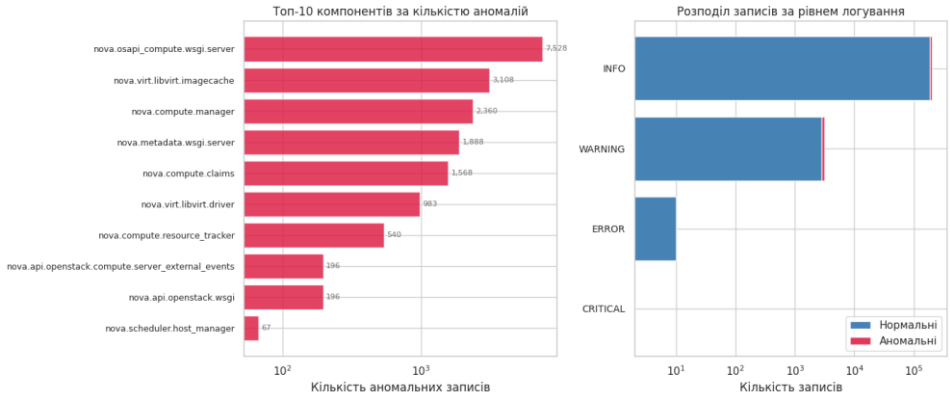


Рис. 8. Розподіл аномалій за компонентами та рівнями логування

Цей експеримент оцінює операційну ефективність інкрементного оновлення моделі — тобто вартість підтримання моделі в актуальному стані в міру надходження нових даних, а не лише якість її прогнозів. Послідовність вікон логу подавалася на вхід порціями (батчами) зростаючого розміру: 1 тис, 2 тис, 4 тис, 8 тис, 16 тис вікон та фінальна порція із залишку. Такий діапазон обрано для того, щоб простежити, як ключові метрики масштабуються з обсягом порції і виявити можливі нелінійності. Для усунення впливу випадкових коливань середовища виконання кожен запуск

повторюється тричі, а підсумковий показник усереднюся.
На кожному кроці фіксується три показники, що характеризують відповідність методу до поставленої задачі: час оновлення; затримка отримання прогнозу; розмір отриманої моделі.

Таблиця 5. Результат експерименту при фінальному розмірі бази

Модель	**Час оновлення (с)**	**Latency інференсу (мс)**	**Артефакт (МБ)**	**PR-AUC**	**Прискорення vs Retrieval**
Random Forest	1.55	49	17.2	0.094	203×
LSTM	19.75	123	0.10	0.104	2585×
Retrieval	0.0076	505	27.5	0.092	1×

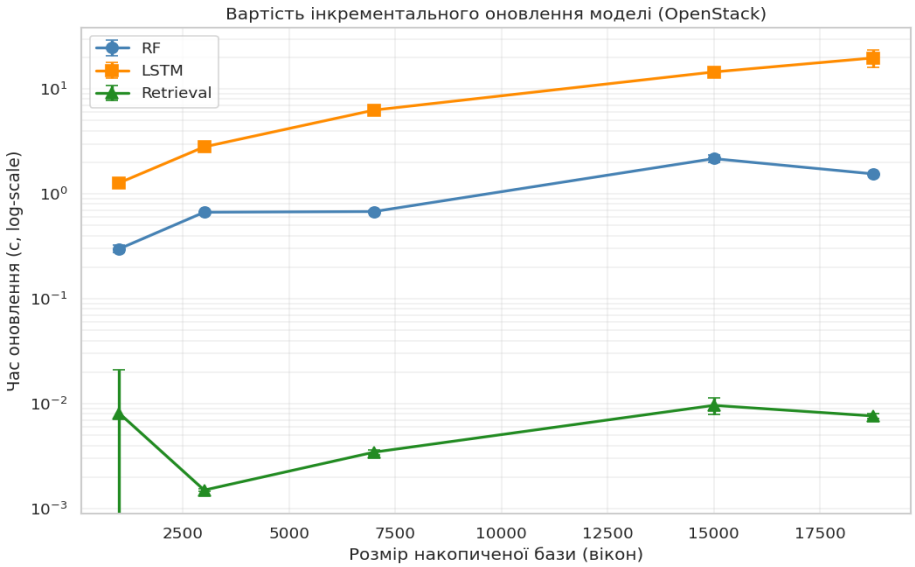


Рис. 9. Вартість інкрементального оновлення моделі

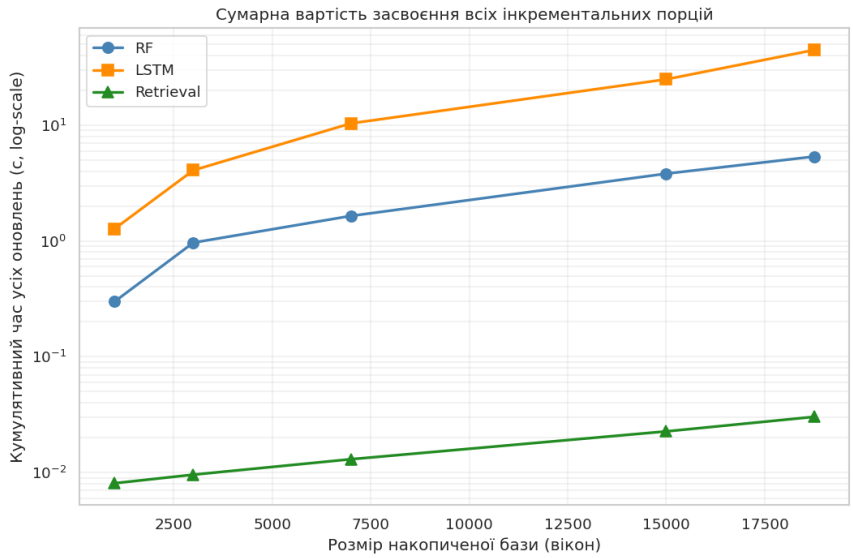


Рис. 10. Кумулятивна вартість засвоєння всіх інкрементальних порцій

Висновки

Емпіричні результати відповідають теоретичним припущенням. Час оновлення векторно-аналогового метода не залежить від розміру баз і має складність $O(\boldsymbol{d})$ на вектор. Статистичні методи демонструють пологе зростання обмежене кількістю паралельних потоків тестового стенду. LSTM зростає майже лінійно відповідно до розміну бази.

У реальних системах, де нові події і ланцюги подій з'являються досить часто, така різниця означає, що Retrival можна оновлювати у реальному часі без зупинки сервісу, тоді як LSTM вимагає окремих циклів перенавчання.

Зростання затримки отримання прогнозу в Retrival зростає лінійно з розміром бази, що є наслідком використання IndexFlatIP [21] пошуку зі складністю O(N). В реальних системах для усунення цього обмеження застосовуються наближені індексні структури (HNSW, IVF-PQ) зі складністю запиту $O(log\, N)$, які вимагають більше обчислювальних ресурсів. Окремий аналіз методів пошуку винесений за межі цього аналізу.

Обидва результати безпосередньо обґрунтовують перехід до архітектури на основі retrieval як фундаменту прогнозування відмов у подійно-орієнтованих системах.

Література

1. Волокита А., Зоткін К. Проблематика проєктування програмних систем на основі подійно-орієнтованих архітектур (EDA). Measuring and Computing Devices in Technological Processes. 2024. № 4. С. 130–136.

2. Margara A., Cugola G., Tamburrelli G. Learning from the past: automated rule generation for complex event processing. Proceedings of the 8th ACM International Conference on Distributed Event-Based Systems. 2014. P. 47–58.

3. Flouris I. et al. Issues in complex event processing: status and prospects in the big data era. Journal of Systems and Software. 2017. Vol. 127. P. 217–236.

4. Takács L., Zádor P. Stochastic processes. London : Methuen, 1960.

5. Norris J. R. Markov chains. Cambridge:Cambridge University Press,1998.

6. Rabiner L., Juang B. An introduction to hidden Markov models. IEEE ASSP Magazine. 2003. Vol. 3, no. 1. P. 4–16.

7. Chen M. et al. Markov approximation for combinatorial network optimization. IEEE Transactions on Information Theory. 2013. Vol. 59, no. 10. P. 6301–6327.

8. Akimova O. et al. Application of economic and mathematical modelling to detect and prevent fraud in financial statements. Financial & Credit Activity: Problems of Theory & Practice. 2023. Vol. 6, no. 53.

9. Baier C. et al. Parametric Markov chains: PCTL complexity and fraction-free Gaussian elimination. Information and Computation. 2020. Vol. 272. P. 104504.

10. Muhamedyev R. Machine learning methods: an overview. Computer Modelling & New Technologies. 2015. Vol. 19, no. 6. P. 14–29.

11. Salman H. A., Kalakech A., Steiti A. Random forest algorithm overview. Babylonian Journal of Machine Learning. 2024. Vol. 2024. P. 69–79.

12. Bentéjac C., Csörgő A., Martínez-Muñoz G. A comparative analysis of gradient boosting algorithms. Artificial Intelligence Review. 2021. Vol. 54, no. 3. P. 1937–1967.

13. Valkenborg D. et al. Support vector machines. American Journal of Orthodontics and Dentofacial Orthopedics. 2023. Vol. 164, no. 5. P. 754–757.

14. Shumway R. H., Stoffer D. S. ARIMA models. Time Series Analysis and Its Applications: with R Examples. Cham : Springer International Publishing, 2017. P. 75–163.

15. Welch G. et al. An introduction to the Kalman filter. 1995.

16. Hassoun M. H. Fundamentals of artificial neural networks. Cambridge, MA : MIT Press, 1995.

17. Sherstinsky A. Fundamentals of recurrent neural network (RNN) and long short-term memory (LSTM) network. Physica D: Nonlinear Phenomena. 2020. Vol. 404. P. 132306.

18. Mathur S., Chhabra A. Vector search algorithms: a brief survey. 2024 4th International Conference on Ubiquitous Computing and Intelligent Information Systems (ICUIS). IEEE, 2024. P. 365–371.

19. Danopoulos D., Kachris C., Soudris D. Approximate similarity search with faiss framework using fpgas on the cloud. International Conference on Embedded Computer Systems. Cham : Springer International Publishing, 2019. P. 373–386.

20. Richardson E. et al. The receiver operating characteristic curve accurately assesses imbalanced datasets. Patterns. 2024. Vol. 5, no. 6.

21. Zhan J. et al. Learning to retrieve: how to train a dense retrieval model effectively and efficiently. arXiv preprint arXiv:2010.10469. 2020.

Волокита А. М., Зоткін К. В.

АНАЛІЗ МЕТОДІВ ПРОГНОЗУВАННЯ ВІДМОВ У ПОДІЙНО-ОРІЄНТОВАНІЙ АРХІТЕКТУРІ

Відмовостійкість розподілених систем є критично важливою умовою безперервної роботи сучасних IT-інфраструктур за зростання складності задач, високого навантаження та постійної модифікації програмного забезпечення. Ключову роль у запобіганні відмовам відіграє система керування та оркестрації подій, на яку покладено керування робочими процесами та прийняття рішень щодо реконфігурації. Проте традиційні методи моніторингу, що ґрунтуються на фіксованих порогах або окремих подіях, не здатні своєчасно виявити приховані закономірності, які призводять до відмов, тож система керування реагує вже після факту відмови. Метою дослідження є аналіз існуючих методів прогнозування відмов, придатних для інтеграції в систему керування подійно-орієнтованою архітектурою, та емпіричне обґрунтування переваг підходу на основі векторного пошуку для потоків із динамічним складом подій. Виконано порівняльний аналіз основних класів методів: детермінованих (Complex Event Processing), імовірнісних моделей станів (ланцюги Маркова, HMM), класичних статистичних і ML-методів (Random Forest, Gradient Boosting, SVM, ARIMA, фільтр Калмана), послідовнісних нейронних моделей (RNN/CNN, LSTM) та векторно-аналогових методів за схожістю. Методи проранжовано за критеріями, важливими для системи прийняття рішень: час реакції, інкрементальне навчання, масштабованість та виявлення нових і неявних залежностей. Емпіричний експеримент засвідчив, що, на відміну від LSTM, який потребує майже лінійного перенавчання, підхід на основі векторного пошуку (FAISS/HNSW) дозволяє оновлювати модель у реальному часі без зупинки сервісу за майже сталої вартості оновлення. Отримані результати обґрунтовують застосування retrieval як фундаменту прогнозування відмов у системах керування подійно-орієнтованими середовищами.

Ключові слова: *прогнозування відмов, система керування, оркестрація подій, відмовостійкість, векторний пошук, інкрементальне навчання, LSTM.*

Volokyta A. M., Zotkin K. V.

ANALYSIS OF FAILURE PREDICTION METHODS IN EVENT-DRIVEN ARCHITECTURE

Fault tolerance of distributed systems is a critical condition for the continuous operation of modern IT infrastructures under growing task complexity, high load and constant software modification. A key role in failure prevention is played by the event management and orchestration system, which is responsible for workflow control and decision-making on reconfiguration. However, traditional monitoring methods, based on fixed thresholds or isolated events, cannot detect in time the hidden patterns that lead to failures, so the management system reacts only after a failure has already occurred. The aim of the study is to analyse existing failure-prediction methods suitable for integration into the management system of an event-driven architecture, and to empirically justify the advantages of a vector-search-based approach for streams with a dynamic event composition. A comparative analysis is carried out for the main classes of methods: deterministic ones (Complex Event Processing), probabilistic state models (Markov chains, HMM), classical statistical and ML methods (Random Forest, Gradient Boosting, SVM, ARIMA, Kalman filter), sequential neural models (RNN/CNN, LSTM) and vector-analogue similarity methods. The methods are ranked by criteria important for a decision-making system: reaction time, incremental learning, scalability, and the ability to detect novel and implicit dependencies. The empirical experiment shows that, unlike LSTM, which requires near-linear retraining, a vector-search-based approach (FAISS/HNSW) enables real-time model updates without service interruption at an almost constant update cost. The obtained results justify the use of retrieval as the foundation for failure prediction in management systems of event-driven environments.

Keywords: *failure prediction, management system, event orchestration, fault tolerance, vector search, incremental learning, LSTM.*

Стаття подана до редакції: 13/09/2026
Стаття прийнята до опублікування:15/09/2026
Стаття опублікована: 10/11/2026
Стаття поширюється на умовах ліцензії CC BY 4