КОМП'ЮТЕРНІ НАУКИ

УДК 004.03:004.45:004.75 https://doi.org/10.32515/2664-262X.2025.12(43).2.9-19

П. В. Віжевський, **О. С. Савенко**, проф., д-р техн. наук
Хмельницький національний університет, м. Хмельницький, Україна
e-mail: vizhevskyipv@khmnu.edu.ua, savenko_oleg_st@ukr.net

Еволюційна адаптація політик DLP за умов дрейфу концепції у потокових даних

У сучасних DLP-системах, що обробляють потокові дані у хмарних і гібридних середовищах, фіксовані політики швидко деградують через дрейф концепції. Оператори змушені одночасно утримувати ризик-зважену вартість пропусків на прийнятному рівні, обмежувати навантаження від хибних спрацювань, виконувати SLO за затримкою та забезпечувати стабільність тривог за жорстких бюджетів пам'яті й обчислень. Це створює суперечливі вимоги, які традиційні детектори або ручне тюнінгування політик вирішують неповною мірою. Робота пропонує онлайновий еволюційний метод адаптації політик, який формулює синхронне налаштування як багатокритеріальну оптимізацію за обмежених ресурсів. Застосовано хромосомне кодування політик, кероване дрейфом перемикання «дослідження/експлуатація», архів перевірених рішень для «теплих» стартів, компактну активну суміш та захищені відкочування для операційної безпеки. Експериментальна оцінка на восьми потоках (синтетичних і реальних) демонструє, що інтегральна вартість утримується в межах 0–3,5% від найкращого базового методу (середній абсолютний розрив ≈1,6%), дотримано p95 < 100 мс часу обробки, а мінливість частоти тривог знижено на 50–63%, при тому частота хибних спрацювань лишається співставною або нижчою. Виявлено дві чутливості: поріг «брами дрейфу», що керує балансом дослідження/експлуатації, і короткі обчислювальні сплески одразу після виявленої зміни. Їх пом'якшують архівні «теплі» старти, компактна суміш і бюджети на інтенсивність мутацій без втрати чутливості.

генетичні алгоритми, запобігання витоку даних, виявлення аномалій, дрейф концепції, хмарна безпека

Постановка проблеми. Системи запобігання витоку даних (DLP) працюють у середовищах, де поведінка користувачів і характеристики трафіку постійно змінюються. Такі зсуви руйнують припущення про стаціонарність, спричиняють концептуальний дрейф, погіршують точність виявлення інцидентів і стабільність тривог у потокових сценаріях. У підсумку зростає навантаження на операторів і ризик пропуску критичних подій, що перетворює підтримання якості на безперервне завдання експлуатації. [1]. До цього додаються й практичні обмеження виробничих умов: невеликі допуски на затримку, високі вимоги до пропускної здатності, неповна або відкладена розмітка подій. Підтримувати працездатність виключно періодичним перенавчанням стає дедалі дорожче та повільніше. Виникає потреба в механізмах онлайнової адаптації, які б ураховували ресурсні обмеження та не зупиняли сервіс під час змін [2]. Щоб така адаптація була своєчасною, потрібні індикатори змінності, які є водночас інформативними та недорогими обчислювально. Одним із практичних сигналів виступає прогностична невизначеність моделей, що дозволяє фіксувати появу нових режимів даних ще до накопичення достатньої кількості міток [3]. У домені DLP додаткові труднощі породжує строката природа каналів передачі, сезонність бізнес-процесів і висока ціна помилкових спрацювань. Фіксовані пороги та статичні політики не витримують таких коливань, тому потрібні процедури спостереження і керованої реакції, здатні утримувати баланс між чутливістю й стабільністю [4]. Неправильний добір детекторів і порогів безпосередньо відгукується на вартості реагування. Тому рішення мають спиратися на відтворювані порівняння за різних типів дрейфу і режимів наванта-ження, щоб уникати прихованих упереджень і забезпечувати стабільність тривог [5].

Для об'єктивної оцінки таких рішень потрібні загальнодоступні колекції потокових наборів з чітко визначеними сценаріями змінності та узгодженими метриками. Це робить можливим коректне співставлення точності, стабільності та експлуатаційних витрат між різними реалізаціями [6].

Аналіз останніх досліджень і публікацій. За останні роки дослідження дрейфу концепції у потокових системах досягло помітної зрілості. Про це свідчить поява системних оглядів, які встановлюють спільну термінологію та вимоги до методів виявлення змін. Hinder зі співавторами у своїй оглядовій роботі детально розглянули питання моніторингу в середовищах, що еволюціонують, виокремивши ключові типи дрейфу та підходи до їхнього спостереження [1]. Така робота закладає понятійну основу для подальших прикладних рішень, адже без чіткого розуміння природи змінності важко обирати адекватні інструменти реагування.

У прикладному руслі окреслилися дві перспективні стратегії роботи з нестаціонарністю даних. Перша опирається на прогностичну невизначеність нейромережевих моделей як сигнал раннього попередження про зміни. Baier та колеги показали, що зростання невизначеності передбачає дрейф ще до накопичення достатньої кількості розмічених прикладів – особливо цінно у реальному часі, коли мітки надходять із затримкою або зовсім відсутні [3]. Другий напрямок пропонує випереджальне моделювання майбутніх розподілів даних, щоб класифікатор заздалегідь готувався до очікуваних змін. Li зі співавторами запропонували метод DDG-DA, який генерує синтетичні дані для імітації ймовірних дрейфів і таким чином скорочує провали точності під час перехідних періодів [2].

Чимало зусиль дослідників спрямовано на створення надійних референтних умов для порівняння різних методів. de Barros і Santos провели масштабне зіставлення детекторів дрейфу, показавши, що стабільність тривог залежить не лише від алгоритму, а й від налаштувань порогів і характеру змінності самих даних [5]. Ця робота підкреслює важливість стандартизованих протоколів оцінювання. Losing із колегами доповнили цю інфраструктуру, зібравши колекцію публічно доступних наборів потокових даних із різними типами дрейфу, що робить експерименти відтворюваними та дозволяє чесно порівнювати нові підходи з базовими [6].

На рівні класифікаційних алгоритмів домінують ансамблеві підходи, здатні адаптуватися без повної зупинки конвеєра обробки. Gomes та співавтори розробили Adaptive Random Forests, які динамічно поновлюють набір дерев у відповідь на зміни у вхідних даних, забезпечуючи стабільну точність навіть за частих дрейфів [7]. Cano і Krawczyk пішли далі, запропонувавши спочатку Kappa Updated Ensemble для загального випадку потокових класифікацій [8], а згодом – ROSE, спеціально налаштований на роботу з дисбалансованими класами та складними поєднаннями раптових і поступових зсувів [9]. Ці рішення демонструють, як багатомодельна архітектура підвищує стійкість системи до непередбачуваних змін.

Швидкість реагування на дрейф також привернула значну увагу. Mahdi зі співавторами розглянули випадок, коли позначки класів відсутні, а система має миттєво відреагувати на появу нових режимів поведінки [10]. Їхній підхід дозволяє виявляти зміни на ранніх стадіях і ініціювати адаптацію до того, як відбудеться істотне погіршення якості. У близькому контексті В близькому контексті Adams та колеги зосередилися на поясненні дрейфу, намагаючись зробити процес виявлення змін прозорішим для експертів і операторів – критично важливо у системах із високою ціною помилок [11].

Еволюційні підходи до адаптації представляють окрему гілку досліджень. Ghomeshi із колегами запропонували метод EACD, який використовує еволюційні

механізми для пошуку оптимальних конфігурацій моделі в умовах змінності даних, уникаючи дорогих повних перенавчань і зберігаючи різноманітність варіантів політик [12]. Такий підхід особливо корисний у багатокритеріальних задачах, де потрібно збалансувати точність, швидкість і витрати на обчислення. Yu та співавтори розширили цю ідею на онлайновий буст-алгоритм, який поєднує адаптивне навчання з мульти-потоковою класифікацією, що дає змогу ефективно обробляти паралельні джерела даних [13].

Останні роботи також звертають увагу на архітектурні аспекти виявлення дрейфу. Hu зі співавторами застосували глибокі нейронні мережі та автокодувальники для детекції змін, показавши, що такі моделі можуть виявляти складні, нелінійні зсуви у високорозмірних просторах ознак [14]. Це розширює арсенал інструментів для роботи з неочевидними формами дрейфу, які важко вловити традиційними статистичними тестами.

У контексті систем запобігання витоку даних питання адаптації до дрейфу набувають додаткової ваги через специфіку предметної області. Alneyadi зі співавторами у своєму огляді DLP-систем підкреслили, що успішна робота таких рішень залежить не лише від якості моделей, а й від керованості процесів оновлення політик, контролю ризиків, пов'язаних із самою адаптацією, та прозорості ухвалення рішень [4]. Будь-які зміни в конфігурації потенційно можуть привести до появи нових каналів витоку або послабити захист. Тому механізми онлайн-адаптації повинні включати аудит, відкат до попередніх станів та чітку верифікацію безпеки оновлень.

Попри успіхи у виявленні дрейфу, алгоритмічній адаптації та стандартизації оцінювання, поєднання цих компонентів у єдиний механізм для DLP-систем не є досконало дослідженим. Серед доступних наукових робіт зазвичай розглядаються окремі аспекти: моніторинг змін, ансамблеві методи або еволюційний пошук, але рідко вони об'єднуються в цілісне рішення. Актуальність даної роботи заключається в розробці комплексного рішення з керованими ресурсними обмеженнями, багатокритеріальною оптимізацією та архівним механізмом безпечного відкочуванням. Таке рішення буде враховувати специфіку потокових середовищ з високими вимогами до безпеки і надійності в експлуатації.

Постановка завдання. Мета дослідження полягає у розробці та перевірці методу онлайнової еволюційної адаптації політик для систем запобігання витоку даних (DLP). Його розробка включає в себе:

– формалізування моделі оптимізації DLP-політик, що складається з ризик-зваженої вартості пропущених витоків, навантаження від хибних спрацювань, час на прийняття рішення, складність політики та стабільність частоти тривог з урахуванням експлуатаційних обмежень;

– розробка механізму кодування хромосом політик, що включає адаптивну стратегію перемикання режимів еволюційного пошуку залежно від неконтрольованих індикаторів дрейфу концепції (динаміка невизначеності моделі, прогнозовані зсуви розподілів, маркери змін у латентному просторі);

– створення архівного механізму керованих відкочуваннь до перевірених політик минулих режимів та використання зваженої суміші різних активних політик для підтримки стійкості до повторюваних патернів дрейфу;

– проведення експериментального оцінювання на потокових наборах даних із різними типами дрейфу з колекції driftDatasets [6], порівнюючи розроблений метод з базовими детекторами (ADWIN, DDM, EDDM, UDD) за інтегральною вартістю, стабільністю тривог та при дотриманні SLO.

Реалізація поставлених завдань сприятиме створенню науково обґрунтованого методу онлайнової адаптації DLP-систем, здатного підтримувати баланс між точністю

виявлення інцидентів, стабільністю тривог та ресурсними обмеженнями у нестаціонарних середовищах.

Виклад основного матеріалу. Потік подій трактується як послідовність документів/ повідомлень із контекстом каналу та ролі користувача. Щоб уніфікувати опис, запровадимо ризиковий скор, порогову політику прийняття рішень, подієву вартість, формальне означення дрейфу, багатометову ціль і експлуатаційні обмеження.

$$s_t = f_\theta(x_t, m_t) \in [0,1], \tag{1}$$

де $x_t \in \mathbb{R}^d$ – ознаки події/документа в момент t, $m_t \in \mathcal{M}$ – контекст (канал, роль, час, географія тощо), $f_\theta(\cdot)$ – скоринг-функція з параметрами політики θ, s_t – ризиковий скор (ступінь/ймовірність ризику витоку).

Рішення приймається порогово, що дозволяє узгодити точність і вартість втручань.

$$a_t = \begin{cases} allow, & \text{якщо } s_t < \tau_q, \\ quarantine, & \text{якщо } \tau_q \le s_t < \tau_b, \\ block, & \text{якщо } s_t \ge \tau_b, \end{cases} \tag{2}$$

де $a_t \in \mathcal{A} = allow, quarantine, block$ – дія політики; $\tau_q, \tau_b \big(0 \le \tau_q \le \tau_b \le 1\big)$ – пороги прийняття рішень.

Важливою є сервісна затримка – час від надходження події до застосування дії.

$$\ell_t(\theta) = t_{out} - t_{in}, \tag{3}$$

$\ell_t(\theta)$ – час обробки події t, t_{in} – час надходження події, t_{out} – час винесення дії a_t.

Щоб коректно зважити наслідки інцидентів, вводимо коефіцієнт ризику, що враховує чутливість вмісту, канал і роль.

$$\varphi(m_t, x_t) = s(x_t) \cdot \gamma_{chan}(m_t) \cdot \gamma_{role}(m_t), \tag{4}$$

де: $\varphi(\cdot)$ – ваговий коефіцієнт ризику інциденту, $s(x_t) \in [0,1]$ – чутливість вмісту, $\gamma_{chan}(m_t) > 0$ – вага каналу (e-mail, чат, хмара тощо), $\gamma_{role}(m_t) > 0$ – вага ролі/ідентичності.

Подієва вартість інтегрує ризик пропущеного витоку, хибні спрацювання, час обробки і складність політики.

$$\begin{aligned} c_t(\theta) = \lambda_{FN} \cdot 1\{y_t = 1, a_t = allow\} \cdot \varphi(m_t, x_t) \\ +\lambda_{FP} \cdot 1\{y_t = 0, a_t \in quarantine, block\} \\ +\lambda_{LAT} \cdot \ell_t(\theta) + \lambda_{OPS} \cdot \kappa(\theta), \end{aligned} \tag{5}$$

де $c\_t(\theta)$ – подієва вартість, $y_t \in 0,1$ – істинна мітка (1 = витік), може надходити із затримкою/бути відсутньою, $\lambda_{FN}, \lambda_{FP}, \lambda_{LAT}, \lambda_{OPS} > 0$ – ваги витрат за пропуск витоку, хибні спрацювання, час обробки та операційну складність, $\kappa(\theta) \ge 0$ – міра складності політики (напр., кількість правил/довжина виразу), $1 \cdot \{\ \}$ – індикаторна функція.

Для оцінки продуктивності на відрізку часу використаємо середню (очікувану) вартість.

$$\bar{C}_T(\theta) = \frac{1}{T} \sum_{t=1}^{T} \mathbb{E}[c_t(\theta)], \tag{6}$$

де $\bar{C}_T(\theta)$ – середня (очікувана) вартість на горизонті T, T – довжина інтервалу оцінювання, $\mathbb{E}[\cdot]$– математичне сподівання (за випадковістю потоків/міток).

Керування шумністю операцій досягається контролем інтенсивності тривог і їх дисперсії.

$$\sigma^2_{alert,W}(\theta) = \frac{1}{W}\sum_{i=0}^{W-1}\left(A_{t-i}(\theta) - \bar{A}_{t,W}(\theta)\right)^2,$$
$$\bar{A}_{t,W}(\theta) = \frac{1}{W}\sum_{i=0}^{W-1} A_{t-i}(\theta), \tag{7}$$

де $A_t(\theta) \in 0{,}1$ – індикатор наявності тривоги (quarantine/block) для події t, W – розмір ковзаючого вікна, $\sigma^2_{alert,W}$ – дисперсія частоти тривог у вікні, $\bar{A}_{t,W}$ – середня частота тривог у вікні.

Дрейф концепції визначаємо як статистично значущу зміну розподілу даних, у неконтрольованих сценаріях використовуємо проксі через невизначеність моделі.

$$driftatt \Leftrightarrow D\big(P_t(x_t, y_t) \parallel P_{t-\Delta}(x_t, y_t)\big) > \delta, \tag{8}$$

де P_t – розподіл (x_t, y_t) у момент t, Δ – часовий крок порівняння, $D(\cdot\|\cdot)$ – обрана дивергенція/відстань між розподілами (напр., KL, Hellinger, MMD), $\delta > 0$ – поріг дрейфу. Без контролю: $D\big(P_t(u_t) \parallel P_{t-\Delta}(u_t)\big) > \delta$, де u_t – проксі-невизначеність моделі (ентропія/дисперсія прогнозів).

Оптимізаційна постановка розкладається на кілька цілей: ризик пропуску витоків, FP-навантаження, часову затримку на обробку події, складність політики та стабільність тривог.

$$J_1(\theta) = \frac{1}{T}\sum_{t=1}^{T}\lambda_{FN}\cdot 1\{y_t = 1, a_t = allow\}\cdot\varphi(m_t, x_t),$$
$$J_2(\theta) = \frac{1}{T}\sum_{t=1}^{T}\lambda_{FP}\cdot 1\{y_t = 0, a_t \in \{quarantine, block\}\},$$
$$J_3(\theta) = p95(\ell_t(\theta)_{t=1}^{T}), \quad J_4(\theta) = \kappa(\theta), \quad J_5(\theta) = \sigma^2_{aert,W}l(\theta), \tag{9}$$

де J_1 – складова очікуваної вартості, пов'язана з пропущеними витоками (ризик-зважена), J_2 – складова витрат за хибні спрацювання, J_3 – 95-й перцентиль часу обробки події, $p95(\cdot)$ – оператор 95-го перцентиля, J_4 – штраф за складність політики, J_5 – штраф за нестабільність тривог.

Для практичного керування компромісами зручно застосувати скаляризацію або безпосередню Парето-оптимізацію.

$$F(\theta) = w_1J_1(\theta) + w_2J_2(\theta) + w_3J_3(\theta) + w_4J_4(\theta) + w_5J_5(\theta),$$
$$F(\theta) \to min_{\theta\in\Theta}, \tag{10}$$

де $F(\theta)$ – скаляризована ціль, $w_i > 0$ – ваги цілей, Θ – допустима множина параметрів політики/скорера.

Експлуатаційні обмеження відображають SLO на затримку, а також бюджети пам'яті та обчислень, характерні для реальних DLP-пайплайнів.

$$P(\ell_t(\theta) \le L_{max}) \ge 1 - \varepsilon, \quad M(\theta) \le M_{max},$$
$$\bar{\chi}(\theta) = \frac{1}{T}\sum_{t=1}^{T}\chi_t(\theta) \le B_{max}, \tag{11}$$

де L_{max} – SLO-поріг на час обробки події, ε – допустима частка порушень SLO, $M(\theta)$ – споживання пам'яті системою політик, $\chi_t(\theta)$ – обчислювальна вартість на подію t, $\bar{\chi}(\theta)$ – середня обчислювальна вартість, B_{max} – бюджет обчислень/часу.

Оптимізаційна задача онлайнової еволюції параметрів політики, яка поєднує точність, стабільність і ресурси, та розв'язується в умовах дрейфу концепції: знайти $\theta^* = \min_{\theta\in\Theta} \mathrm{F}(\theta)$ за умов (11) та онлайново еволюціонувати θ на потоці $S = (x_t, m_t)$.

Під «визначенням дрейфу» розуміємо сигнал від зовнішнього детектора, який може працювати у контрольованому або неконтрольованому режимі. Метод не зупиняє сервіс, працює у ковзних вікнах і поважає обмеження продуктивності, що є типовими для DLP-пайплайнів.

Хромосома відбиває рішення про те, як саме система реагує на ризикові події, та з яких правил складається політика. Фіксована частина контролює роботу на рівні порогів, вікон, інтенсивностей мутації та параметрів детектора дрейфу. Змінна частина описує набір правил і їхні ваги. Початкова популяція ініціалізується з наявних у організації політик, а також випадковими легальними конфігураціями, щоб забезпечити різноманіття.

$$\chi = [\tau_q, \tau_b, W, \rho, \eta, \mu_0, \mu_1, \psi, u_1, \dots, u_L], \tag{12}$$

де χ – хромосома, τ_q, τ_b – пороги з формули (2), W – розмір ковзного вікна для оцінювання, $\rho \in (0,1)$ – коефіцієнт експоненційного забування, $\eta > 0$ – крок оновлення ваг у механізмах пам'яті, $\mu_0, \mu_1 \geq 0$ – базова та додаткова інтенсивності мутації, ψ – параметри чутливості детектора дрейфу, u_i – гени правил із вагами, $L \leq L_{max}$ – кількість генів правил.

Правило u_i кодує умовний вираз на ознаках і контексті (наприклад: «канал = e-mail; роль = HR; збіг шаблону IBAN»), а також цільову дію та вагу. Щоб стримати зростання довжини виразів, застосовується обмеження L_{max} і штраф у функції якості на складність (J_4 з формули (9)). Під час ініціалізації дотримується інваріант $\tau_q \leq \tau_b$, для чого використовуємо проєкцію на допустиму область після мутацій.

Якість кандидатів оцінюється у мікровікнах, що дозволяє відслідковувати короткотермінову динаміку та не накопичувати застарілу статистику. Рекомендовано обирати W з урахуванням середнього часу реагування на інциденти та обсягу трафіку. Усі компоненти якості доступні у потоковій формі: p95 часу обробки події рахуємо приблизними порядковими статистиками, стабільність тривог – одночасно з оновленням індикатора A_t.

$$\begin{aligned} F_k(\chi) = w_1 J_{1,k}(\chi) + w_2 J_{2,k}(\chi) + w_3 J_{3,k}(\chi) \\ + w_4 J_{4,k}(\chi) + w_5 J_{5,k}(\chi), \end{aligned} \tag{13}$$

де $F_k(\chi)$ – значення цілі у вікні I_k, $J_{j,k}$ – компоненти (9), обчислені на I_k, $w_j > 0$ – ваги цілей. Останні вікна важать більше завдяки експоненційному забуванню. Цим керує параметр ρ: за малих ρ метод швидше реагує, але стає менш стабільним. $\tilde{F}_t(\chi)$ – згортка якості за останні K вікон з коефіцієнтом забування ρ, K – кількість вікон в агрегуванні.

$$\tilde{F}_t(\chi) = \frac{\sum_{k=1}^{K} \rho^{K-k} F_k(\chi)}{\sum_{k=1}^{K} \rho^{K-k}}. \tag{14}$$

Алгоритм працює по вікнах. На початку вікна отримуємо сигнал z_k від детектора. Якщо у середовищі відбулася значуща зміна, збільшуємо інтенсивність мутацій, розширюємо пошук і тимчасово зменшуємо вплив штрафів за складність.

Якщо зміни не визначено, працюємо у консервативному режимі, що мінімізує коливання частоти тривог.

$$\mu_k = \mu_0 + \mu_1 z_k, \tag{15}$$

де μ_k – інтенсивність мутації у вікні k, $z_k \in 0,1$ – бінарний сигнал від детектора дрейфу у вікні k (1 якщо подію зміни виявлено, 0 інакше), μ_0, μ_1 – параметри з (12).

$$p_i = \frac{exp\left(-\beta \tilde{F}_t(\chi_i)\right)}{\sum_{j=1..N} exp\left(-\beta \tilde{F}_t(\chi_j)\right)}, \tag{16}$$

де p_i – імовірність вибору хромосоми χ_i як батьківської, $\beta > 0$ – параметр селекційного тиску, N – розмір популяції.

Практично це означає, що у періоди стабільності нові кандидати мало відрізняються від поточних лідерів, а за сигналу про зміну з'являються «віддалені» варіанти, здатні швидше пристосуватися.

Практично це означає, що у періоди стабільності нові кандидати мало відрізняються від поточних лідерів, а за сигналу про зміну з'являються «віддалені» варіанти, здатні швидше пристосуватися.

Кросовер поєднує частини двох успішних рішень. Це переносить корисні фрагменти політик між кандидатами, зберігаючи сумісність структур.

$$\chi' = \left[\chi^A_{1:j}, \chi^B_{j+1:L}\right], \quad \chi'' = \left[\chi^B_{1:j}, \chi^A_{j+1:L}\right], \tag{17}$$

де χ^A, χ^B – батьківські хромосоми, j – точка кросоверу, L – довжина кодування, χ', χ'' – нащадки. Мутації порогів виконуються у межах допустимих діапазонів, щоб не порушити інваріант $\tau_q \leq \tau_b$. Відсікання після додавання шуму гарантує коректність.

$$\tau'_q = \left[\tau_q + \sigma_q \varepsilon_1\right]_0^{\tau_b}, \quad \tau'_b = \left[\tau_b + \sigma_b \varepsilon_2\right]_{\tau'_q}^1, \tag{18}$$

де τ'_q, τ'_b – мутовані пороги, $\sigma_q, \sigma_b > 0$ – масштаби шуму, $\varepsilon_1, \varepsilon_2 \sim \mathcal{N}(0,1)$, $[\cdot]_a^b$ – відсікання до інтервалу.

Ваги правил змінюються адаптивно. Крок пропорційний μ_k, тому у періоди визначеного дрейфу пошук активніший.

$$w'_r = \left[w_r + \alpha_k \varepsilon_r\right]_0^{W_{max}}, \tag{19}$$

де w_r – вага r-го правила у гені u_i, w'_r – мутована вага, $\alpha_k = c\mu_k$ – адаптивний крок, $c > 0$ – константа масштабу, $\varepsilon_r \sim \mathcal{N}(0,1)$, w_{max} – верхня межа ваги.

Кількість правил у політиці може змінюватися. У стабільні періоди довжина коду утримується близько до поточного значення, під час змін дозволяється додати або видалити правило.

$$L' = L + \xi_k, \quad \xi_k \in -1,0,+1, \tag{20}$$

де L' – нова кількість генів правил, ξ_k – випадкова зміна довжини зі схильністю до 0 у стаціонарі та до ±1 за сигналу від детектора; імовірності залежать відμ_k, L обмежено $[0, L_{max}]$.

Дрейф часто має повторюваний характер. Архів дає змогу швидко повернути перевірену політику, якщо середовище повертається до попереднього стану. Паралельно підтримуємо набір із кількох політик, що зменшує ризик просідань на локальних зсувах.

$$\pi_j(t+1) = \frac{\pi_j(t) exp\left(-\eta \tilde{F}_t(\chi_j)\right)}{\sum_{m=1}^{M} \pi_m(t) exp\left(-\eta \tilde{F}_t(\chi_m)\right)}, \quad (21)$$

де $\pi_j(t)$ – вага j-ї політики в наборі на час t, M – кількість активних політик в наборі.

Щоб уникнути затяжних деградацій, перевіряємо, чи не стало гірше від останньої архівної опори на значущу величину. Якщо так, виконуємо кероване відкочування.

$$r_k = F_k(\chi_{\text{cur}}) - F_k(\chi_{\text{arch}}) > \delta_{\text{rec}}, \quad (22)$$

де r_k – різниця вартостей між поточною політикою χ_{cur} і найкращою архівною χ_{arch} у вікні k, $\delta_{\text{rec}} > 0$ – поріг відкочування. Розмір архіву вибирається з урахуванням пам'яті та різноманіття історичних станів. Практично доцільно тримати кілька останніх політик із найнижчою $\tilde{F}_t$ та невисокою кореляцією рішень.

Будь-який кандидат має відповідати сервісним вимогам. Це перевіряється до розгортання. За потреби використовується «тіньовий» режим або канарейкове розгортання, що знижує ризик.

$$I_{acc}(\chi) = 1\{ P(\ell_t(\chi) \le L_{max}) \ge 1 - \varepsilon \wedge M(\chi) \le M_{max} \wedge \tilde{\chi} \le B_{max} \}, \quad (23)$$

де $I_{acc}(\chi) \in 0{,}1$ – індикатор прийнятності, ℓ_t, L_{max}, ε, $M(\cdot)$, $\tilde{\chi}(\cdot)$, B_{max} – як у формулі (11). Ймовірнісний контроль затримки оцінюємо емпірично на ковзному вікні. Пам'ять і обчислення вимірюємо профайлінгом, що входить у конвеєр оцінювання.

Узагальнений цикл роботи:

1. На початку кожного вікна обчислюємо F_k для активних політик і оновлюємо згортку $\tilde{F}_t$ за (14).
2. Отримуємо сигнал z_k від детектора та встановлюємо μ_k за (15).
3. Виконуємо селекцію за (16), кросовер за (17) і мутації за (18)–(20).
4. Перераховуємо ваги суміші за (21).
5. Перевіряємо умову відкочування (22) і прийнятність за SLO та бюджетами (23).
6. В реальному середовищі використовується найкраща політика або суміш із вагами $\pi_j(t)$, інші кандидати залишаються у популяції для наступних ітерацій.

Такий цикл дає адаптацію «на льоту»: за неконтрольованого визначення дрейфу ми підсилюємо пошук без залучення міток, а за контрольованого – використовуємо обмежені мітки для підтвердження змін і калібрування порогів. Поєднання архіву, суміші політик і адаптивних мутацій утримує баланс між якістю виявлення ризику, стабільністю тривог та вимогами продуктивності.

Мета експерименту полягає у перевірці того, чи зменшує онлайнова еволюційна адаптація політик інтегральну вартість $\tilde{F}_t$ на потоках з різними типами дрейфу та чи утримуються сервісні обмеження на затримку, пам'ять і обчислення. Оцінювання виконується у режимі test-then-train з ковзними вікнами та експоненційним забуванням. Дані беруться зі стандартної колекції driftDatasets [6] (SEA, Rotating Hyperplane, Moving/Interchanging RBF, Mixed Drift, Transient Chessboard, Electricity, Weather).

Синтетичні потоки використовуються для вимірювання затримки визначення змін, частоти хибних визначень і частки пропусків при відомих моментах дрейфу. Реальні потоки Electricity та Weather застосовуються для оцінювання в умовах сезонностей та структурних зсувів. Базові довжини ковзного вікна: $W \in \{256{,}512{,}1024\}$ для синтетики, для ELEC і Weather використовуємо $W = \{512{,}1024\}$.

Коефіцієнт забування $\rho \in [0.85,0.98]$ підбирається на стабільних префіксах кожного потоку. Усі методи дотримуються однакових ресурсних бюджетів: p95-затримка L_{max}=200 мс; пікова пам'ять M_{max}=1024 МБ; середні обчислення на подію B_{max}=5.0 мс; максимальна довжина вікна W_{max}=1024. Ці значення однакові для всіх методів і потоків та застосовуються під час перевірки прийнятності політики за правилом.

Порівнюємо наш підхід з ADWIN, DDM, EDDM і неконтрольованим UDD (визначення змін за невизначеністю моделі). Для ADWIN калібруємо δ на стаціонарному відрізку, щоб вирівняти частоту хибних визначень на рівні α=0.05. Для DDM застосовуємо пороги $p_{min} + 2s_{min}$ (попередження) та $p_{min} + 3s_{min}$ (дрейф) з буфером не менше 1000 подій. Для EDDM використовуємо мінімальну відстань між помилками 30 та стандартні пороги на індикаторах. У UDD індикатором слугує EWMA-згладжена невизначеність моделі з $\lambda \in [0.05,0.3]$ та порогом $h \in [2.5,4.0]$. Для нашого методу: розмір популяції $N \in \{16,32,64\}$, селекційний тиск $\beta \in [0.5,4]$, інтенсивності мутацій $\mu_0 \in [0.01,0.1]$, $\mu_1 \in [0.05,0.3]$, кількість активних політик у суміші $M \in \{2,3,5\}$; ваги цілей $(w_1, \dots, w_5)$ нормалізовані до 1.

Міряли інтегральну вартість $\tilde{F}_t$, стандартне відхилення частоти тривог σ_A та хибні сповіщення на 10 000 подій (FA/10k). У таблиці 1 подано порівняння нашого методу з найкращим серед базових на кожному наборі даних.

Таблиця 1 – Порівняння з базовими методами

Набір	Тип дрейфу	Найкращий базовий метод	$\Delta\tilde{F}_t$	$\Delta\sigma_{alert}$	ΔFA/10k
SEA Concepts	Раптовий (повторюваний	DDM	+3.5%	−60.3%	+2.4%
Rotating Hyperplane	Поступовий	EDDM	+1.0%	−58.2%	+5.4%
Moving RBF	Інкрементальний	EDDM	−1.9%	−53.7%	−7.7%
Interchanging RBF	Раптовий (каскад)	DDM	+1.6%	−65.2%	−7.3%
Mixed Drift	Змішаний	EDDM	+0.0%	−52.8%	−2.5%
Transient Chessboard	Віртуальний + раптовий	ADWIN	+2.6%	−71.6%	+2.0%
Electricity (ELEC)	Сезонний/структурний	UDD	−0.4%	−57.4%	−7.8%
Weather (Offutt AFB)	Сезонний/трендовий	UDD	+0.9%	−63.4%	−6.8%

Джерело: розроблено авторами

Наш метод забезпечує конкурентну інтегральну вартість (на більшості потоків у межах 0–2% від найкращих базових підходів) і суттєво знижує стандартне відхилення частоти тривог. За раптового дрейфу (SEA, Interchanging RBF) ми отримуємо значення σ_alert на 60% нижчі за базові. За поступового дрейфу (Rotating Hyperplane, Moving RBF) метод забезпечує на 50–58% кращу стабільність. Для змішаного та віртуального дрейфу (Mixed Drift, Transient Chessboard) еволюційний підхід дає на 53–59% нижчу σ_alert. Реальні потоки (Electricity, Weather) демонструють 57–63% зменшення стандартного відхилення частоти тривог при використанні нашого методу, що свідчить про те, що механізм архіву допомагає уникати коливань між режимами. Загалом на всіх потоках стабільність тривог поліпшується до 63%, а рівень хибних сповіщень подібний або менший порівняно з базовими.

Метод оперативно реагує на раптові та поступові/інкрементальні зсуви без надмірної корекції, зберігаючи паритет за вартістю та уникаючи тривалих постдрифтових коливань. За метрикою FA/10k метод кращий на 5 із 8 потоків і перебуває в межах незначного відхилення на решті; у поєднанні зі зниженням мінливості це означає менше піків зайвих сповіщень і меншу втому аналітиків.

Висновки. Показано, що за реалістичних обмежень потокової обробки еволюційний метод здатний збалансувати конкуруючі цілі DLP: ризик-зважену вартість виявлення, навантаження від хибних спрацювань, затримку та стабільність

тривог. У розглянутих синтетичних і реальних потоках інтегральна вартість утримується в межах 0–3.5% від найкращого базового методу, середній абсолютний розрив близько 1.6%. Операційно важливо, що протягом експериментів дотримано SLO на час обробки p95 ≤ 200 мс, а стандартне відхилення частоти тривог зменшується орієнтовно на 50–63%, що скорочує пікові навантаження на аналітиків.

Виявлено дві практичні чутливості. По-перше, налаштування «брами дрейфу», яка керує перемиканням між дослідженням і експлуатацією: надмірна чутливість може спричинити короткочасні підвищення FA, надто консервативні пороги сповільнюють адаптацію за швидких зсувів. По-друге, після зафіксованого дрейфу можливе короткочасне збільшення часу обробки подій, хоча середнє значення і p95 залишаються в межах бюджету. Ці ефекти пом'якшуються «теплим стартом» з архіву перевірених політик, компактною сумішшю активних політик та підбором бюджетів на інтенсивність мутацій, який не спричиняє втрати чутливості.

Практичне впровадження потребує використання методу між етапами збору даних і застосування політик, опираючись на потокові дані телеметрії електронної пошти, проксі серверів, компонентів рівня хосту. Оновлення політик впроваджується через оркестраційний шар із хуками для відкочування. Рекомендована траєкторія розгортання оновлених політик: тіньовий режим із моніторингом $\tilde{F}_t$ і SLO, далі канаркове увімкнення в обмежених доменах, з подальшим розгортанням на усю систему. Індикатор прийнятності $I_{acc}(\chi)$ має слугувати «воротами якості», автоматично відхиляючи кандидатів або виконуючи відкочування у разі порушення бюджету. Необхідно мінімізовувати використання реальних даних, натомість використовуючи агреговані ознаки та аудитні журнали (без чутливого контенту).

Подальша робота включає онлайновий контроль розміру вікна та коефіцієнта забування, додавання доменно-обізнаних сигналів для більш стаціонарної частоти тривог при збереженні вартості й SLO, бюджетоване добування міток у напів/неконтрольованих сценаріях, перехід від скаляризації до Парето-оптимізації та дослідження нейросимвольних подань правил із широкою оцінкою на корпоративній телеметрії. Сукупно результати підтверджують доцільність еволюційних алгоритмів як практичного інструмента стійкої та прозорої адаптації DLP у середовищах із дрейфом концепції.

Список літератури

1. Hinder F., Vaquet V., Hammer B. One or two things we know about concept drift – A survey on monitoring in evolving environments. Part A: Detecting concept drift. *Frontiers in Artificial Intelligence*. 2024. Vol. *7*. Art. 1330257. DOI: https://doi.org/10.3389/frai.2024.1330257.
2. Li W., Yang X., Liu W., Xia Y., Bian J. DDG-DA: Data Distribution Generation for Predictable Concept Drift Adaptation. *Proceedings of the AAAI Conference on Artificial Intelligence*. 2022. Vol. *36*, № 4. С. 4092–4100. DOI: https://doi.org/10.1609/aaai.v36i4.20327.
3. Baier L., Schlör T., Schöffer J., Kühl N. Detecting Concept Drift with Neural Network Model Uncertainty. *arXiv preprint*. 2021. DOI: https://doi.org/10.48550/arXiv.2107.01873.
4. Alneyadi S., Sithirasenan E., Muthukkumarasamy V. A Survey on Data Leakage Prevention Systems. *Journal of Network and Computer Applications*. 2016. Vol. *62*. С. 137–152. DOI: https://doi.org/10.1016/j.jnca.2016.01.008.
5. de Barros R. S. M., Santos S. G. T. C. A Large-Scale Comparison of Concept Drift Detectors. *Information Sciences*. 2018. Vol. *451–452*. С. 348–370. DOI: https://doi.org/10.1016/j.ins.2018.04.014.
6. Losing V., Hammer B., Wersing H. Drift Datasets. *GitHub repository*. 2025. URL: https://github.com/vlosing/driftDatasets (дата звернення: 05.11.2025).
7. Gomes H. M., Bifet A., Read J., Barddal J. P., Enembreck F., Pfahringer B., Holmes G., Abdessalem T. Adaptive Random Forests for Evolving Data Stream Classification. *Machine Learning*. 2017. Vol. *106*. С. 1469–1495. DOI: https://doi.org/10.1007/s10994-017-5642-8.
8. Cano A., Krawczyk B. Kappa Updated Ensemble for Drifting Data Stream Mining. *Machine Learning*. 2020. Vol. *109*. С. 175–218. DOI: https://doi.org/10.1007/s10994-019-05840-z.
9. Cano A., Krawczyk B. ROSE: Robust Online Self-Adjusting Ensemble for Continual Learning on Imbalanced Drifting Data Streams. *Machine Learning*. 2022. Vol. *111*. С. 2561–2599. DOI: doi.org/10.1007/s10994-022-06168-x.
10. Mahdi O. A., Pardede E., Ali N., Cao J. Fast Reaction to Sudden Concept Drift in the Absence of Class Labels. *Applied Sciences*. 2020. Vol. *10*, № 2. Art. 606. DOI: https://doi.org/10.3390/app10020606.
11. Adams J. N., van Zelst S. J., Rose T., van der Aalst W. M. P. Explainable Concept Drift in Process Mining. *Information Systems*. 2023. Vol. *114*. Art. 102177. DOI: https://doi.org/10.1016/j.is.2023.102177.

12. Ghomeshi H., Gaber M. M., Kovalchuk Y. EACD: Evolutionary Adaptation to Concept Drifts in Data Streams. *Data Mining and Knowledge Discovery*. 2019. Vol. *33*. С. 760–793. DOI: https://doi.org/10.1007/s10618-019-00614-6.
13. Yu E., Lu J., Zhang B., Zhang G. Online Boosting Adaptive Learning under Concept Drift for Multistream Classification. *Proceedings of the AAAI Conference on Artificial Intelligence*. 2024. Vol. *38*, № 15. С. 16522–16530. DOI: https://doi.org/10.1609/aaai.v38i15.29590.
14. Hu L., Lu Y., Feng Y. Concept Drift Detection Based on Deep Neural Networks and Autoencoders. *Applied Sciences*. 2025. Vol. *15*, № 6. Art. 3056. DOI: https://doi.org/10.3390/app15063056.

References

1. Hinder, F., Vaquet, V., & Hammer, B. (2024). One or two things we know about concept drift – A survey on monitoring in evolving environments. Part A: Detecting concept drift. *Frontiers in Artificial Intelligence, 7*, Article 1330257. https://doi.org/10.3389/frai.2024.1330257.
2. Li, W., Yang, X., Liu, W., Xia, Y., & Bian, J. (2022). DDG-DA: Data distribution generation for predictable concept drift adaptation. *Proceedings of the AAAI Conference on Artificial Intelligence, 36*(4), 4092–4100. https://doi.org/10.1609/aaai.v36i4.20327.
3. Baier, L., Schlör, T., Schöffer, J., & Kühl, N. (2021). Detecting concept drift with neural network model uncertainty. *arXiv* preprint arXiv:2107.01873. https://doi.org/10.48550/arXiv.2107.01873.
4. Alneyadi, S., Sithirasenan, E., & Muthukkumarasamy, V. (2016). A survey on data leakage prevention systems. *Journal of Network and Computer Applications, 62*, 137–152. https://doi.org/10.1016/j.jnca.2016.01.008.
5. de Barros, R. S. M., & Santos, S. G. T. C. (2018). A large-scale comparison of concept drift detectors. *Information Sciences, 451–452*, 348–370. https://doi.org/10.1016/j.ins.2018.04.014.
6. Losing, V., Hammer, B., & Wersing, H. (n.d.). Drift datasets. *GitHub repository*. Retrieved November 5, 2025, from https://github.com/vlosing/driftDatasets.
7. Gomes, H. M., Bifet, A., Read, J., Barddal, J. P., Enembreck, F., Pfahringer, B., Holmes, G., & Abdessalem, T. (2017). Adaptive random forests for evolving data stream classification. *Machine Learning, 106*, 1469–1495. https://doi.org/10.1007/s10994-017-5642-8.
8. Cano, A., & Krawczyk, B. (2020). Kappa updated ensemble for drifting data stream mining. *Machine Learning, 109*, 175–218. https://doi.org/10.1007/s10994-019-05840-z.
9. Cano, A., & Krawczyk, B. (2022). ROSE: Robust online self-adjusting ensemble for continual learning on imbalanced drifting data streams. *Machine Learning, 111*, 2561–2599. https://doi.org/10.1007/s10994-022-06168-x.
10. Mahdi, O. A., Pardede, E., Ali, N., & Cao, J. (2020). Fast reaction to sudden concept drift in the absence of class labels. *Applied Sciences, 10*(2), 606. https://doi.org/10.3390/app10020606.
11. Adams, J. N., van Zelst, S. J., Rose, T., & van der Aalst, W. M. P. (2023). Explainable concept drift in process mining. *Information Systems, 114*, Article 102177. https://doi.org/10.1016/j.is.2023.102177.
12. Ghomeshi, H., Gaber, M. M., & Kovalchuk, Y. (2019). EACD: Evolutionary adaptation to concept drifts in data streams. *Data Mining and Knowledge Discovery, 33*, 760–793. https://doi.org/10.1007/s10618-019-00614-6.
13. Yu, E., Lu, J., Zhang, B., & Zhang, G. (2024). Online boosting adaptive learning under concept drift for multistream classification. *Proceedings of the AAAI Conference on Artificial Intelligence, 38*(15), 16522–16530. https://doi.org/10.1609/aaai.v38i15.29590.
14. Hu, L., Lu, Y., & Feng, Y. (2025). Concept drift detection based on deep neural networks and autoencoders. *Applied Sciences, 15*(6), Article 3056. https://doi.org/10.3390/app15063056.

Petro Vizhevskyi, **Oleg Savenko**, Prof., Dr. tech. sci
Khmenlnystskyi National University, Khmenlnystskyi, Ukraine

Evolutionary Adaptation of DLP Policies under Concept Drift in Streaming Data

In modern streaming DLP systems deployed across cloud and hybrid environments, fixed policies degrade rapidly due to concept drift. Operators must simultaneously control the risk-weighted miss cost, limit the false-alarm burden, meet latency SLOs, and keep alert streams stable under tight memory and compute budgets. These competing objectives are not adequately balanced by traditional detectors or manual policy tuning.

We present an online evolutionary controller that casts policy adaptation as constrained multi-objective optimization. The method uses a chromosome encoding with drift-aware exploration–exploitation switching, an archive of vetted policies for warm starts, a compact active mixture, and guarded rollbacks for operational safety. On six streams (synthetic and real), the controller keeps the integrated cost within 0–3.5% of the best baseline (mean absolute gap ≈1.6%), sustains p95 latency below 100 ms, and reduces alert-rate volatility by 50–63% while maintaining comparable or lower false-alarm rates.

Two practical sensitivities emerge: the drift-gate threshold governing the exploration/exploitation balance, and short-lived compute bursts immediately after detected changes. Warm starts, a compact mixture, and mutation-budget guards mitigate these effects without sacrificing responsiveness.

genetic algorithms, Data Loss Prevention, anomaly detection, concept drift, cloud security

Одержано (Received) 06.11.2025 *Прорецензовано (Reviewed) 11.11.2025*
Прийнято до друку (Approved) 25.11.2025