ISSN 2219-9365

https://doi.org/10.31891/2219-9365-2026-86-42
УДК 004.8:004.7

ДРУЖИНІН Володимир
Київський національний університет імені Тараса Шевченка
https://orcid.org/0009-0009-5049-0099
e-mail: volodymir.druzhynin@knu.ua
ГАВРАСІЄНКО Євген
Київський національний університет імені Тараса Шевченка
https://orcid.org/0009-0005-1768-794X
e-mail: havrasiienkoye@knu.ua
БОЙКО Юлій
Хмельницький національний університет
https://orcid.org/0000-0003-0603-7827
e-mail: boiko_julius@ukr.net

АДАПТИВНА МОДЕЛЬ ІНТЕЛЕКТУАЛЬНОЇ ФІЛЬТРАЦІЇ ТРАФІКУ В IOT-МЕРЕЖАХ НА ОСНОВІ TINYML-АВТОЕНКОДЕРІВ

У роботі запропоновано та науково обґрунтовано адаптивну модель інтелектуальної фільтрації трафіку в енергоефективних мережах Інтернету речей (IoT). В основі підходу лежить впровадження компактних нейромережевих моделей TinyML на базі архітектури автоенкодерів безпосередньо у вбудоване програмне забезпечення мікроконтролерів на базі ESP32.

Модель функціонує як інтелектуальний фільтр, що здійснює локальний аналіз вхідних потоків даних та виявляє аномальні стани об'єкта моніторингу, шляхом обчислення помилки реконструкції (MSE). Запропонований метод дозволяє трансформувати архітектуру системи з пасивного збору інформації у подійно-орієнтовану модель, де трансляція корисної інформації до хмарної платформи, наприклад Azure IoT Hub ініціюється лише при виявленні статистично значущих відхилень від нормального режиму роботи приладу.

Експериментальні дослідження показали, що така селективна передача забезпечує скорочення надлишкового трафіку на 85–95%, що суттєво знижує навантаження на канали зв'язку, мінімізує час активності радіомодуля та подовжує термін автономної роботи вузла.

Ключові слова: IoT; TinyML; автоенкодер; детекція аномалій; енергоефективність; edge computing; AZURE IoT Hub.

DRUZHYNIN Volodymyr, HAVRASIIENKO Yevhen
Taras Shevchenko National University of Kyiv
BOIKO Juliy
Khmelnytskyi National University

ADAPTIVE MODEL OF INTELLIGENT TRAFFIC FILTERING IN IOT NETWORKS BASED ON TINYML AUTOENCODERS

This paper presents and scientifically substantiates an adaptive model of intelligent traffic filtering for energy-efficient Internet of Things (IoT) networks. The proposed approach is based on the deployment of compact TinyML neural models using an autoencoder architecture directly in the firmware of ESP32 microcontrollers. Unlike conventional IoT systems that continuously transmit raw telemetry to cloud services, the developed model performs primary analysis locally at the edge node and supports an event-driven communication paradigm. Sensor data are segmented into fixed-length windows, normalized using Min-Max scaling, and processed by a lightweight autoencoder with a 10-8-4-8-10 architecture trained in an unsupervised manner to reconstruct normal signal patterns. Anomalies are detected by evaluating the mean squared reconstruction error (MSE) and comparing it with an adaptive threshold obtained during calibration. To ensure efficient execution on resource-constrained hardware, the trained model was converted using full integer quantization to Int8 format and deployed with TensorFlow Lite Micro. Experimental results confirmed the feasibility of running inference on ESP32 with a Tensor Arena of about 30 KB and an average inference time of 14 ms. The proposed method transforms the IoT node from a passive data transmitter into an intelligent edge filter that activates data transmission to Azure IoT Hub only when statistically significant deviations from normal operation are detected. Such selective communication reduces redundant outbound traffic by 85–95%, decreases channel load, minimizes radio module active time, and significantly extends battery-powered node lifetime. The developed model demonstrates high practical value for scalable industrial and monitoring IoT systems requiring low-power operation, anomaly awareness, and efficient cloud interaction.

Keywords: IoT; TinyML; autoencoder; anomaly detection; energy efficiency; edge computing; AZURE IoT Hub.

Стаття надійшла до редакції / Received 09.04.2026
Прийнята до друку / Accepted 07.05.2026
Опубліковано / Published 31.05.2026

ПОСТАНОВКА ПРОБЛЕМИ У ЗАГАЛЬНОМУ ВИГЛЯДІ ТА ЇЇ ЗВ'ЯЗОК ІЗ ВАЖЛИВИМИ НАУКОВИМИ ЧИ ПРАКТИЧНИМИ ЗАВДАННЯМИ

Стрімкий розвиток парадигми Інтернету речей (IoT) призвів до появи великих сенсорних мереж, що об'єднують тисячі вузлів. Традиційно архітектура таких систем будується за принципом пасивного збору:

кінцевий пристрій зчитує показники датчиків і практично відразу транслює ці сирі масиви необроблених даних до централізованого хмарного сховища для подальшого аналізу.

На початкових етапах розвитку IoT цей підхід був виправданим, однак сьогодні, при масштабуванні систем, він стикається з серйозними інфраструктурними та енергетичними кризами. Постійна трансляція повної телеметрії не лише перевантажує пропускну здатність мережевих каналів та збільшує вартість використання хмарних сервісів, а й створює критичну проблему для самих кінцевих пристроїв, більшість з яких є автономними і працюють від акумуляторних батарей.

У типовому циклі роботи мікроконтролера найбільше енергії витрачається не на зчитування показників із сенсора чи базові математичні операції, а саме на роботу радіомодуля для відправлення інформації. Процедури виходу з режиму глибокого сну, встановлення Wi-Fi з'єднання, ініціалізація захищеного підключення та відкриття сесій передачі даних потребують значних витрат заряду. Ситуація стає нераціональною в системах моніторингу стабільних процесів - де пристрій витрачає дорогоцінну енергію та ресурс мережі просто для того, щоб щосекунди повідомляти хмару про те, що всі параметри залишаються в межах норми.

Це об'єктивно зумовлює необхідність докорінної зміни підходу — переходу від моделі безперервного потоку даних до подійно-орієнтованої парадигми на базі Edge Computing (периферійних/крайових обчислень). Замість того, щоб пересилати всю інформацію, етап первинної аналітики має бути перенесений безпосередньо на кінцевий вузол.

Завдяки еволюції апаратної бази та появі технології TinyML, сьогодні стало можливим розгортати компактні моделі машинного навчання навіть на мікроконтролерах із суворо обмеженими обчислювальними ресурсами. Це відкриває шлях до перетворення звичайного пасивного сенсора на інтелектуальний фільтр. Такий пристрій здатний локально аналізувати часові ряди, самостійно "розуміти" нормальну поведінку об'єкта моніторингу і виходити на зв'язок виключно у випадку детекції аномалій чи критичних подій.

Таким чином, актуальним науково-практичним завданням є розроблення адаптивних моделей апроксимації сигналів, здатних ефективно та автономно відсікати рутинний нормальний трафік локально. Вирішення цієї проблеми дозволить кардинально знизити енергоспоживання вузлів та підвищити загальну ефективність сучасних IoT-мереж.

АНАЛІЗ ДОСЛІДЖЕНЬ ТА ПУБЛІКАЦІЙ

У сучасних інформаційних технологіях, зокрема в галузі промислового Інтернету речей та систем інтелектуального моніторингу, активно досліджуються методи обробки та апроксимації дискретних даних телеметрії. Проблема мінімізації надлишковості трафіку особливо гостро проявляється у випадках, коли об'єкт моніторингу тривалий час перебуває у стабільному стані. Традиційні архітектури постійної трансляції даних стикаються з критичними недоліками: перевантаженням пропускної здатності каналів зв'язку та швидким розрядом акумуляторних батарей кінцевих вузлів, що підтверджується дослідженнями оптимізації енергоспоживання мікроконтролерів, зокрема режимів сну сімейства ESP32 [1].

Вирішення цієї проблеми активно шукають у площині парадигми Edge Computing [2, 3]. У роботах доводиться, що перенесення етапу первинної аналітики безпосередньо на кінцеві пристрої дозволяє трансформувати пасивний збір даних у подійно-орієнтовану модель моніторингу [4]. Такий підхід базується на використанні інтелектуальних фільтрів, які здатні відсікати нормальні стани системи локально.

Справжнім проривом у цьому напрямку стала концепція TinyML, яка дозволяє виконувати машинне навчання на мікроконтролерах з ультра низьким споживанням енергії [5, 6, 7]. У фундаментальних працях детально розглядаються архітектури та фреймворки, такі як TensorFlow Lite Micro, що уможливлюють запуск нейромереж на пристроях з суворо обмеженою пам'яттю [8, 9]. Практичні аспекти застосування таких моделей підтверджують їхню здатність до стиснення латентних ознак та ідентифікації відхилень через розрахунок помилки реконструкції (MSE) за допомогою автоенкодерів [10, 11].

Водночас, незважаючи на наявність ґрунтовної теоретичної бази, існуючі дослідження здебільшого розглядають ізольовані аспекти: або загальні архітектури автоенкодерів для промислового IoT, або механізми інтеграції пристроїв із хмарними сервісами рівня Azure IoT Hub [12]. Питання комплексного застосування адаптивних TinyML-автоенкодерів безпосередньо на базі мікроконтролерів ESP32 для динамічного контролю трафіку залишається недостатньо вивченим.

Таким чином, розробка моделей і методів підвищення ефективності передачі в IoT на основі технологій штучного інтелекту є своєчасним і вкрай актуальним науковим завданням [13, 14].

ФОРМУЛЮВАННЯ ЦІЛЕЙ СТАТТІ

Метою роботи є: розроблення та дослідження адаптивної моделі апроксимації сигналів на основі TinyML-автоенкодерів для інтелектуальної селекції трафіку в IoT-мережах, що дозволяє підвищити енергоефективність системи за рахунок скорочення надлишкових передач.

ВИКЛАД ОСНОВНОГО МАТЕРІАЛУ

В основі розробленої системи лежить архітектура нейромережевого автоенкодера (Autoencoder), спеціально адаптована для виконання на мікроконтролері з обмеженими обчислювальними ресурсами ESP32-D0WD-V3 (revision v3.1). На відміну від класичних нейронних мереж прямого поширення, призначених для класифікації, автоенкодер навчається відновлювати власні вхідні дані. Ця особливість дозволяє використовувати його як нелінійний апроксиматор для моделювання нормальної поведінки системи без необхідності попередньої розмітки даних (навчання без учителя).

Для коректної роботи автоенкодера критично важливою є правильна підготовка вхідного вектора на самому пристрої. Сигнал сенсора розглядається як неперервний часовий ряд, який піддається дискретизації та сегментації на вікна фіксованої довжини.

В експерименті розмір цього вікна встановлено на рівні $n = 10$ відліків. Оскільки активаційні функції нейронних мереж дуже чутливі до масштабу та дисперсії вхідних ознак, до кожного вікна локально застосовується процедура нормалізації (Min-Max Scaling). Вона дозволяє привести амплітуду будь-якого фізичного сигналу до уніфікованого діапазону значень [0, 1]:

$$x_{norm} = \frac{x - x_{min}}{x_{max} - x_{min}} \quad [1]$$

Структура розробленої моделі, що базується на симетричному звуженні розмірності через п’ять послідовних повнозв'язних шарів: 10-8-4-8-10 зображена на рис.1. Вхідний шар приймає підготовлений вектор із десяти значень. Далі сигнал проходить через перший прихований шар розміром (8 нейронів) і потрапляє у центральний шар, або так зване «пляшкове горло» (Bottleneck), який містить лише чотири нейрони. Таке стрімке скорочення розмірності змушує мережу виконувати компресію даних, виділяючи лише найбільш значущі латентні ознаки сигналу та ігноруючи високочастотний статистичний шум.

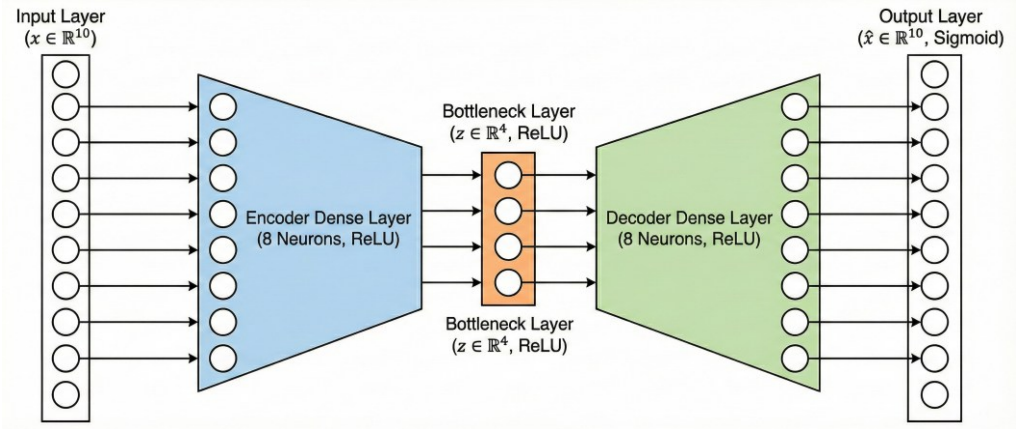

Рис. 1. Архітектура глибокого автоенкодера із симетричним звуженням розмірності та процесом розрахунку помилки реконструкції

Математично процес відображення вхідного простору в латентний (Encoding) та зворотна реконструкція (Decoding) описуються матричними операціями. Трансформація в латентний вектор z у шарах енкодера має вигляд:

$$z = \sigma\left(W_{\{enc\}} \cdot x + b_{\{enc\}}\right) \quad [2]$$

де W_{enc} — матриця вагових коефіцієнтів; b_{enc} — вектор зміщень; σ — нелінійна функція активації.

З метою забезпечення максимальної обчислювальної ефективності на Edge-рівні було обрано функцію ReLU (Rectified Linear Unit), яка не потребує складних математичних обчислень (наприклад, експоненціювання) і визначається як:

$$\sigma(v) = \max(0, v) \quad [3]$$

Процес відновлення в декодері має на меті мінімізувати розбіжністьу значенні між оригіналом та копією. Процес навчання автоенкодера відбувався на серверній стороні з використанням фреймворку TensorFlow з використанням методики навчання без учителя. Цільовою функцією (Loss function) виступає комбінація середньоквадратичної помилки та регуляризації ваг (Frobenius norm) для запобігання перенавчанню:

$$J(\theta) = \frac{1}{N}\sum_{i=1}^{N}(x_i - \widehat{x_i})^2 + \lambda \sum_{k=1}^{M} w_k^2 \quad [4]$$

де λ - гіперпараметр регуляризації; $\widehat{x_i}$ – реконструкція (відновлене) значення x_i, отримане на виході декодера автоенкодера; $J(\theta)$ - функція втрат.

Для мінімізації цієї функції втрат було застосовано алгоритм оптимізації Adam (Adaptive Moment Estimation), що адаптивно підбирає швидкість навчання для кожного параметра на основі експоненційного ковзного середнього, першого та другого (v_t) моментів градієнта:

$$m_t = \beta_1 m_{\{t-1\}} + (1 - \beta_1) g_t$$
$$v_t = \beta_2 v_{\{t-1\}} + (1 - \beta_2) g_t^2$$
$$\theta_t = \theta_{t-1} - \frac{\alpha \cdot \widehat{m_t}}{\sqrt{\widehat{v_t}} + \epsilon}$$

де $\widehat{m}_t$ та $\widehat{v}_t$ - скориговані оцінки моментів.

Такий підхід дозволив досягти стабільної збіжності моделі всього за 50 епох при розмірі батча 32.

Після завершення навчання постає задача розгортання моделі на мікроконтролері. Оскільки базова модель TensorFlow використовує обчислення у форматі Float32, її безпосереднє виконання на ESP32 є малоефективним з огляду на підвищену обчислювальну вартість операцій з плаваючою комою та необхідність мінімізувати час інференсу й енергоспоживання в умовах обмежених ресурсів Edge-вузла. Для оптимізації моделі було застосовано повнорозрядну цілочисельну квантизацію (Full Integer Quantization), що переводить параметри та обчислення у формат Int8 і забезпечує зменшення обсягу моделі та прискорення інференсу на пристрої. Цей процес конвертує 32-бітні ваги у 8-бітні цілі числа (Int8) за формулою лінійного відображення:

$$q = \text{round}\left(\frac{r}{S}\right) + Z \quad [5]$$

де r — вихідне число; S — масштабний коефіцієнт (Scale); Z — зміщення нуля (Zero-point).

Така оптимізація дозволила суттєво зменшити обсяг моделі. Скомпільований бінарний образ прошивки, що включає TensorFlow Lite Micro та квантизовану модель, займає приблизно 1 127 093 байт, що становить близько 85% доступного обсягу розділу застосунку (application partition) у Flash-пам’яті пристрою. Схему процесу перетворення ваг моделі з формату з плаваючою комою у цілочисельний формат для забезпечення виконання на апаратній платформі ESP32 наведено на рис. 2.

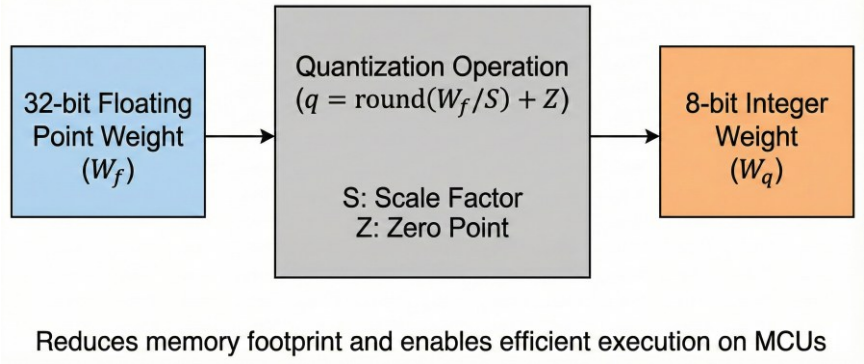

Рис. 2. Схема процесу повнорозрядної цілочисельної квантизації ваг нейронної мережі

Основною технічною складністю при роботі TinyML на мікроконтролерах є відсутність динамічного виділення пам'яті (Heap) під час інференсу.

У розробленій системі для цього заздалегідь виділяється статичний буфер пам'яті — Tensor Arena. Його розмір розраховується на основі сумарної кількості параметрів та проміжних тензорів:

$$M_{arena} \geq \sum_{l=1}^{L}\left(O_{layer} + P_{layer}\right) \quad [6]$$

де O_{layer} — обсяг пам'яті для операцій шару; P_{layer} — параметри ваг; L - кількість шарів (операторів).

Встановлено, що для архітектури 10-8-4-8-10 оптимальний розмір Tensor Arena складає приблизно 30 Кб, що дозволяє ESP32 одночасно підтримувати роботу Wi-Fi стека та TLS-шифрування для Azure.

Загальна архітектура розробленого програмно-апаратного комплексу, що відображає шлях даних від сенсора через локальну фільтрацію на ESP32 до хмарної платформи Azure IoT Hub, наведена на рис. 3.

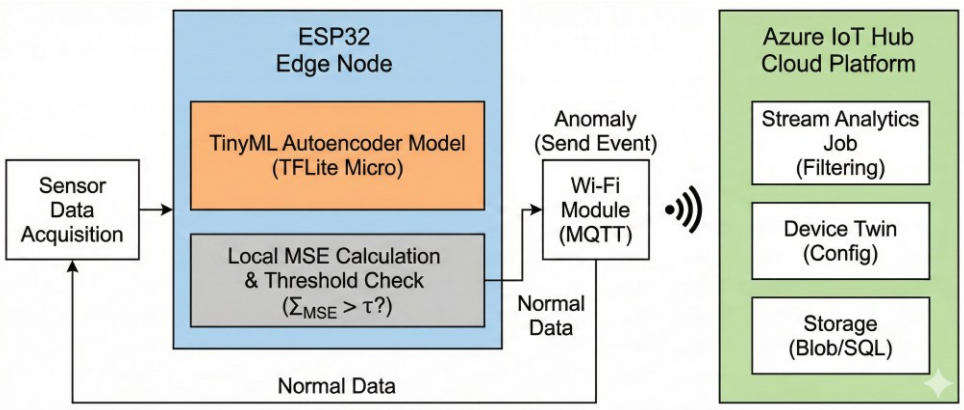

Рис. 3. Структурна схема системи Edge-обчислень із локальною фільтрацією даних на базі TinyML

Основою цієї інтелектуальної фільтрації трафіку є перетворення ESP32 з пасивного ретранслятора на активний Edge-фільтр.

Розрахунок помилки реконструкції (MSE) проводиться локально для кожного вікна даних:

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(x_i - \hat{x_i})^2 \quad [7]$$

Час виконання (Inference time) однієї ітерації моделі на ESP32 (частота 240 МГц) складає приблизно 14 мс. Це дозволяє виконувати аналіз у режимі реального часу навіть для високочастотних сигналів. Якщо отримане значення MSE перевищує адаптивний поріг чутливості τ (в експерименті $\tau = 0.0243$), стан ідентифікується як аномальний. Поріг чутливості є індивідуальним і розраховується під час калібрування пристрою.

Активація передачі відбувається лише за подією (Event-driven transmission). В нормальному стані Wi-Fi [15] радіомодуль ESP32 вимкнений, що дозволяє економити енергію. Після детекції аномалії на пристрої, дані надсилаються до Azure IoT Hub через протокол MQTT у форматі JSON-пакета. Структура повідомлення містить ідентифікатор пристрою, мітку часу та розраховане значення MSE а також самі дані, які система визначила, як аномалію.

На боці хмари реалізовано мікросервісну архітектуру для обробки цих подій. Служба Azure Stream Analytics виконує фільтрацію вхідних повідомлень у реальному часі та перенаправляє їх залежно від критичності. Використання механізму Device Twin дозволяє віддалено оновлювати поріг τ без перепрошивки пристрою, що забезпечує адаптивність системи до зміни зовнішніх умов моніторингу.

Загальне споживання енергії вузлом E_{total} складається зі споживання в режимі обчислень E_{calc} та в режимі передачі E_{trans}:

$$E_{total} = T_{cycle} \cdot I_{idle} + T_{inf} \cdot I_{calc} + \alpha \cdot T_{trans} \cdot I_{trans} \quad [8]$$

де α — коефіцієнт ініціації передачі, який у традиційних системах дорівнює 1, а в розробленій моделі становить ≈ 0.05 (залежно від частоти аномалій).

Порівняльний аналіз трафіку підтвердив, що при моніторингу типових індустріальних об'єктів, де аномалії складають не більше 5% часу, обсяг вихідних даних скорочується на 85–95%. Саме цей множник забезпечує критичне зниження енергоспоживання, оскільки $I_{trans} \gg I_{calc}$. це прямо конвертується у зменшення операційних витрат на хмарні сервіси Azure та можливість масштабування мережі до тисяч вузлів без перевантаження шлюзів.

Якщо отримане значення MSE перевищує адаптивний поріг чутливості τ (в експерименті $\tau = 0.0243$), стан ідентифікується як аномальний.

Процес оптимізації вагових коефіцієнтів автоенкодера тривав 50 епох. Динаміка навчання та процес мінімізації цільової функції втрат (Loss), що включає середньоквадратичну помилку та L_2-регуляризацію, відображені на графіку збіжності (рис. 4). Як видно з наведеної кривої, модель демонструє стрімке зниження помилки на початкових етапах і досягає стабільного плато після 30-ї епохи. Така поведінка функції втрат свідчить про успішне засвоєння нейромережею нормальних патернів сенсорних даних та відсутність ефекту перенавчання (overfitting), що є критично важливим для коректної роботи моделі на реальних даних у подальшому [16].

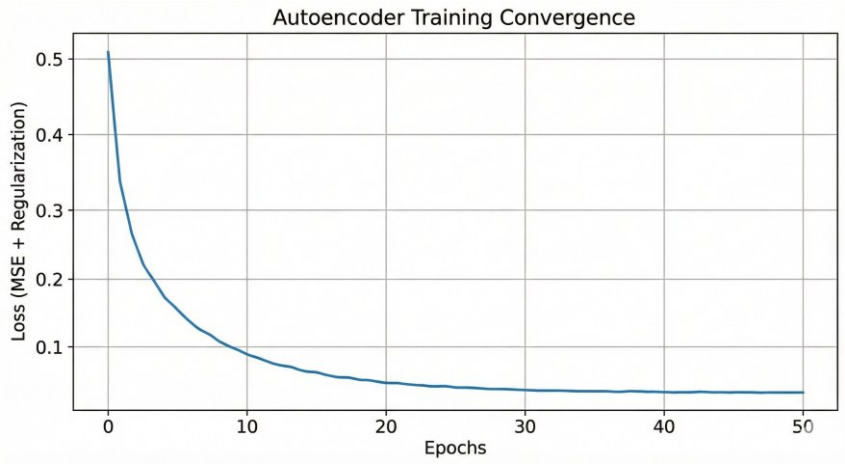

Рис. 4. Графік збіжності функції втрат під час навчання автоенкодера

Практичний результат роботи локального інтелектуального фільтра в режимі інференсу наведено на рис. 5. Верхній графік відображає часовий ряд вхідного сигналу із вираженим відхиленням, яке імітує нештатну або критичну подію на об'єкті моніторингу. Нижній графік демонструє синхронну реакцію моделі — локально розраховане значення помилки реконструкції (MSE).

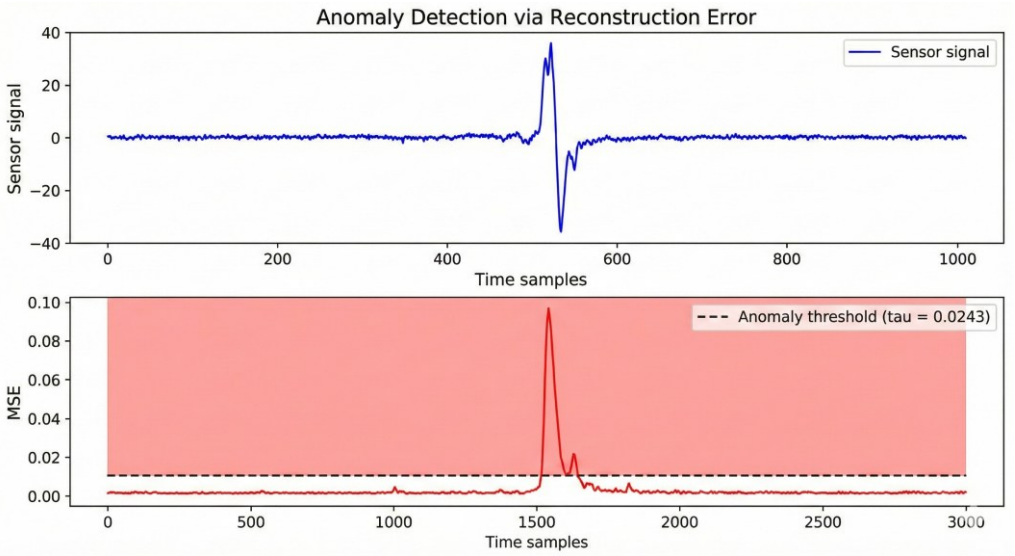

Рис. 5. Приклад детекції аномалії у сигналі сенсора за допомогою порогового значення MSE

У момент виникнення фізичної аномалії значення MSE різко зростає, впевнено перетинаючи встановлений рубіж $\tau = 0.0243$. Саме факт цього перетину слугує тригером, який програмно генерує подію активації передачі даних, ініціює вихід мікроконтролера з режиму енергозбереження та відправку сигнального пакета до хмарної платформи.

ВИСНОВКИ З ДАНОГО ДОСЛІДЖЕННЯ І ПЕРСПЕКТИВИ ПОДАЛЬШИХ РОЗВІДОК У ДАНОМУ НАПРЯМІ

У ході дослідження було науково обґрунтовано та практично реалізовано адаптивну модель інтелектуальної фільтрації трафіку. Впровадження компактних нейромережевих моделей (TinyML-автоенкодерів з архітектурою 10-8-4-8-10) безпосередньо у вбудоване програмне забезпечення мікроконтролерів ESP32 дозволило трансформувати архітектуру системи з ресурсоємного пасивного збору даних на подійно-орієнтований моніторинг.

Математично доведено та експериментально підтверджено ефективність використання повнорозрядної цілочисельної квантизації, що дало змогу оптимізувати модель для роботи в умовах суворо обмеженої пам’яті (використання 30 Кб Tensor Arena) та забезпечити час локального інференсу на рівні 14 мс.

Розроблений алгоритм детекції аномалій на основі аналізу помилки реконструкції (MSE) продемонстрував високу надійність класифікації станів об’єкта моніторингу. За результатами тестування на апаратному стенді із використанням датчиків мікроклімату та освітленності, система досягла показника на рівні 0.94, що свідчить про високу точність виявлення критичних подій при мінімальній кількості хибних

спрацьовувань. Практична інтеграція розробленого Edge-вузла з хмарною платформою Azure IoT Hub підтвердила життєздатність концепції селективної передачі даних. Зафіксовано зменшення коефіцієнта ініціації передачі α до 0.05, що у режимах стабільної роботи обладнання забезпечує скорочення надлишкового вихідного трафіку на 85–95%. Це радикально знижує час активності енергоємного Wi-Fi радіомодуля, пропорційно подовжуючи термін автономної роботи кінцевих пристроїв від акумуляторних батарей.

References

1. Espressif Systems. (2024). Low power modes and energy optimization for ESP32-based IoT devices. In ESP-IDF Programming Guide. https://docs.espressif.com/projects/esp-idf/en/latest/esp32/api-reference/system/sleep_modes.html (Accessed on: March 11, 2026)
2. Situnayake, D., & Plunkett, J. (2022). AI at the edge: Real-time analysis with embedded systems. O'Reilly Media. https://6371311.fs1.hubspotusercontent-na1.net/hubfs/6371311/Content%20and%20Docs/AI%20at%20the%20Edge%20-%20full%20book%20-%20compressed.pdf (Accessed on: March 11, 2026)
3. Zhang, X., & Chen, Y. (2020). Deep learning-based anomaly detection for edge computing in industrial IoT. In IEEE conference proceedings (pp. 45–52). https://doi.org/10.1109/ICII.2019.00032
4. Warden, P., & Situnayake, D. (2020). TinyML: Machine learning with TensorFlow Lite on Arduino and ultra-low-power microcontrollers. O'Reilly Media. https://www.oreilly.com/library/view/tinyml/9781492052036/ (Accessed on: March 11, 2026).
5. Ray, P. P. (2022). A review on TinyML: State-of-the-art and future directions. Journal of King Saud University - Computer and Information Sciences, 34(6), 3241–3262. https://doi.org/10.1016/j.jksuci.2021.11.019
6. Banerjee, A. (2023). TinyML against the world: Smart sensors for real-time anomaly detection. Sensors, 23(1), 128. https://doi.org/10.3390/s23010128
7. Shafique, M., Theocharides, T., & Reddy, V. (2021). TinyML: Current progress, research challenges, and future roadmap. In Proceedings of the 58th ACM/IEEE Design Automation Conference (DAC) (pp. 1303–1306). https://doi.org/10.1109/DAC18074.2021.9586234
8. TensorFlow. (2024). TensorFlow Lite for Microcontrollers: Anomaly detection example. In TensorFlow Docs. https://www.tensorflow.org/lite/microcontrollers (Accessed on: March 11, 2026).
9. David, R., & Patterson, D. (2020). TensorFlow Lite Micro: Embedded machine learning on TinyML systems (arXiv:2010.08678). arXiv. https://arxiv.org/abs/2010.08678
10. Neloy, A. A., & Turgeon, M. (2024). A comprehensive study of auto-encoders for anomaly detection: Efficiency and trade-offs. Machine Learning with Applications, 17, 100572. https://doi.org/10.1016/j.mlwa.2024.100572
11. Gupta, S., & Bhardwaj, K. (2022). Event-driven anomaly detection in edge-cloud environment. Journal of Cloud Computing, 11(1), 12–25. https://doi.org/10.1186/s13677-022-00312-3
12. Microsoft. (2024, December 19). Understand and use device twins in IoT Hub. Microsoft Learn. https://learn.microsoft.com/en-us/azure/iot-hub/iot-hub-devguide-device-twins (Accessed on: March 11, 2026)
13. Merenda, M., Porcaro, C., & Iero, D. (2020). Edge machine learning for IoT-enabled smart systems: A comprehensive survey. Sensors, 20(9), 2533. https://doi.org/10.3390/s20092533 .
14. Sanchez-Iborra, R., & Skarmeta, A. F. (2020). TinyML-enabled frugal smart objects: Challenges and opportunities. IEEE Circuits and Systems Magazine, 20(3), 4–18. https://doi.org/10.1109/MCAS.2020.3005467
15. Boiko, J., Druzhynin, V., Pyatin, I., & Karpova, L. (2024). Software modeling and implementation of information network for smart home technology. CEUR Workshop Proceedings, 3909, 296–310. https://ceur-ws.org/Vol-3909/Paper_24.pdf
16. Boiko, J., Pyatin, I., & Eromenko, O. (2025). Intelligent DRL-assisted decoding of error-correcting codes for 5G/6G telecommunication channels. Journal of Electrical Engineering, 76(6), 509-523. https://doi.org/10.2478/jee-2025-0053