

**МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ДЕРЖАВНИЙ УНІВЕРСИТЕТ ІНФОРМАЦІЙНО-
КОМУНІКАЦІЙНИХ ТЕХНОЛОГІЙ**

*Кваліфікаційна наукова праця
на правах рукопису*

НІЩЕМЕНКО ДМИТРО ОЛЕКСАНДРОВИЧ

УДК 004.89:681.5:004.77

**ДИСЕРТАЦІЯ
МЕТОДИ ОПТИМІЗАЦІЇ КЕРУВАННЯ РОЗУМНИМ БУДИНКОМ НА
ОСНОВІ ІНТЕРНЕТУ РЕЧЕЙ**

123 «Комп'ютерна інженерія»

12 «Інформаційні технології»

Подається на здобуття ступеня доктора філософії

Дисертація містить результати власних досліджень. Використання ідей, результатів і текстів інших авторів мають посилання на відповідне джерело

_____ **Д.О. Ніщенко**

(підпис, ініціали та прізвище здобувача)

Науковий керівник

Аронов Андрій Олексійович

кандидат технічних наук

Київ 2025

АНОТАЦІЯ

Ніщепенко Д.О. Методи оптимізації керування розумним будинком на основі Інтернету речей. – Кваліфікаційна наукова праця на правах рукопису.

Дисертація на здобуття наукового ступеня доктора філософії з галузі знань 12 «Інформаційні технології» за спеціальністю 123 «Комп'ютерна інженерія» – Державний університет інформаційно-комунікаційних технологій Міністерства освіти і науки України, Київ, 2025.

В дисертаційній роботі вирішено актуальне науково-практичне завдання розробки та дослідження методів оптимізації керування розумним будинком на основі Інтернету речей. Робота висвітлює дослідження, спрямовані на усунення фундаментальних прогалин існуючих систем, пов'язаних з їх недостатньою ефективністю, гнучкістю та адаптивністю, що обмежує потенціал підвищення комфорту, енергоефективності та зниження навантаження на користувачів.

В частині архітектур керування, проаналізовано еволюцію від статичних, реактивних парадигм на основі жорстких правил "Якщо-То" (IF-THEN) до адаптивних підходів, що використовують навчання з підкріпленням (RL) та контекстну обізнаність. Встановлено, що ключовим недоліком існуючих систем є їхня негнучкість, нездатність адаптуватися до динамічних вподобань користувачів та необхідність постійного ручного втручання, що обґрунтувало потребу в розробці проактивної архітектури, здатної автономно навчатися контекстуально-залежним, а не лише глобальним, намірам на основі неявної поведінки.

В частині прогнозування енергоспоживання, проаналізовано класичні статистичні моделі (ARIMA) та сучасні методи глибокого навчання (LSTM, GRU, TCN). Встановлено, що класичні методи нездатні моделювати нелінійні та нестационарні дані, тоді як найбільш перспективними є гібридні архітектури, що поєднують глибоке навчання з ансамблями дерев рішень (наприклад, TCN-LightGBM). Однак, було виявлено прогалину в дослідженнях, що стосується оптимізації їхньої обчислювальної ефективності для пристроїв IoT.

В частині якості даних, проаналізовано традиційні методи фільтрації (ковзне середнє, медіанний фільтр, фільтр Калмана) та підходи на основі машинного

навчання (SVM, автокодувальники). Встановлено, що ці методи неефективні при роботі зі складною сумішшю гетерогенних шумів (викиди, дрейф, константні значення), характерних для IoT. Більшість методів не здатні класифікувати тип аномалії, що призводить до застосування неоптимальних стратегій корекції. Таким чином, проведений аналіз підтвердив актуальність теми та дозволив сформулювати чітке завдання дослідження, спрямоване на розробку цілісного комплексу методик, що усувають виявлені прогалини.

Для усунення цих недоліків розроблено цілісний комплекс методик, що складається з трьох ключових наукових результатів.

По-перше, розроблено та вдосконалено гібридний метод короткострокового прогнозування енергоспоживання на основі архітектури TCN-LightGBM. Метод використовує ефективну стратегію "прогнозування залишків", де темпоральна згорткова мережа (TCN) виступає як потужний базовий прогнозист, а ансамбль градієнтного бустингу (LightGBM) навчається коректувати її помилки (залишки), використовуючи розширений набір табличних ознак. Наукова новизна полягає у розробці методу оптимізації цієї гібридної моделі шляхом цілеспрямованого відбору найважливіших ознак лише для моделі-коректора. Експериментально доведено, що цей підхід дозволяє скоротити час навчання в 5,4 рази з 305,72 до 56,47 секунд, при цьому практично без втрати точності прогнозування критичних пікових навантажень, погіршення лише на 0.06%. Це вирішує ідентифіковану проблему компромісу між точністю та обчислювальною вартістю.

По-друге, розроблено метод адаптивного очищення різнорідних сенсорних даних. Формалізовано проблему гетерогенних шумів як суміші аномалій – викиди, дрейф, константні значення. Запропоновано метод ACRA (Adaptive Classification-based Real-time Anomaly cleaning), що поєднує класифікатор Random Forest, евристичне правило на основі дисперсії та адаптивний вибір операторів корекції. На його основі створено вдосконалену гібридну архітектуру H-AD-CLEAN, що використовує значно потужніший класифікатор (на базі 1D-CNN + DWT) для паралельного аналізу "форми" та "текстури" сигналу. Архітектура H-AD-CLEAN реалізує робастну "вентильну логіку" (Gating Logic): вона з високою

точністю (F1-score 0,85) ідентифікує "чисті" сегменти даних і захищає їх від обробки (Pass-Through), водночас застосовуючи потужний Фільтр Калмана до всіх ідентифікованих типів шуму. Експериментально доведено, що H-AD-CLEAN перевершує неадаптивний Фільтр Калмана на 21,1% за метрикою MAE, оскільки він не вносить спотворень у валідні дані.

По-третє, розроблено проактивну архітектуру керування на основі контекстуальних намірів. Архітектура реалізує перехід від низькорівневих правил до парадигми високорівневих намірів де користувач визначає "що", а не "як", наприклад, "баланс комфорту та економії". Ядро системи функціонує на основі багатоцільової оптимізації функції корисності. Наукова новизна полягає у механізмі адаптивного навчання, який автономно навчається не єдиним глобальним, а контекстуальним намірам. Система аналізує ручні втручання користувача як неявний зворотний зв'язок. При втручанні система фіксує поточний контекст (напр., "вечір, високий тариф") та інкрементально коригує ваги $w_{comfort}$ та w_{energy} саме для цього контексту.

Для валідації архітектури розроблено імітаційний стенд на базі Gymnasium , що моделює термодинаміку приміщення, зміну сезонів, динамічні тарифи та імітаційного користувача, здатного генерувати контекстно-залежні ручні втручання.

Валідність розробок підтверджена комплексним експериментальним дослідженням. Результати 60-денного моделювання показали, що Проактивний агент досяг найвищого рівня комфорту 41.5% часу в цільовій зоні та, що найважливіше, зменшив потребу в ручному втручанні користувача на понад 97% порівняно з базовою моделлю на правилах 28 втручань проти 240. Аналіз динаміки підтвердив, що система успішно навчилася складним контекстуальним вподобанням, наприклад, вищий пріоритет комфорту ввечері.

Описаний комплекс методик усуває виявлені недоліки існуючих систем та формує такі переваги, як висока точність прогнозування на пристроях з обмеженими ресурсами, надійність вхідних даних та здатність системи автономно

адаптуватися до мешканців, кардинально мінімізуючи їхнє когнітивне навантаження.

Ключові слова: розумний будинок, Інтернет речей (IoT), оптимізація керування, проактивне керування, керування на основі намірів, неявний зворотний зв'язок, багатоцільова оптимізація, прогнозування енергоспоживання, TCN, LightGBM, очищення даних, гетерогенні шуми, ACRA, H-AD-CLEAN, імітаційне моделювання, прогнозування, очищення даних, машинне навчання.

SUMMARY

Nishchemenko D.O. Methods for optimizing smart home management based on the Internet of Things. – Qualification scientific work in the form of a manuscript.

Dissertation for the degree of Doctor of Philosophy in the field of knowledge 12 "Information Technologies" in the specialty 123 "Computer Engineering" – State University of Information and Communication Technologies of the Ministry of Education and Science of Ukraine, Kyiv, 2025.

The dissertation solves the current scientific and practical task of developing and researching a set of methods for optimizing smart home management based on the Internet of Things. The work highlights research aimed at eliminating fundamental gaps in existing systems associated with their insufficient efficiency, flexibility and adaptability, which limits the potential for increasing comfort, energy efficiency and reducing the load on users.

In terms of control architectures, the evolution from static, reactive paradigms based on rigid "If-Then" (IF-THEN) rules to adaptive approaches using reinforcement learning (RL) and context awareness is analyzed. It is found that the key drawback of existing systems is their inflexibility, inability to adapt to dynamic user preferences and the need for constant manual intervention, which justified the need to develop a proactive architecture capable of autonomously learning context-dependent, not only global, intentions based on implicit behavior.

In terms of energy consumption forecasting, classical statistical models (ARIMA) and modern deep learning methods (LSTM, GRU, TCN) were analyzed. It was found

that classical methods are unable to model nonlinear and non-stationary data, while the most promising are hybrid architectures that combine deep learning with ensembles of decision trees (e.g., TCN-LightGBM). However, a gap in research was identified regarding the optimization of their computational efficiency for IoT devices.

In terms of data quality, traditional filtering methods (moving average, median filter, Kalman filter) and machine learning-based approaches (SVM, autoencoders) were analyzed. It was found that these methods are ineffective when working with a complex mixture of heterogeneous noise (outliers, drift, constant values) typical of IoT. Most methods are unable to classify the type of anomaly, which leads to the application of suboptimal correction strategies. Thus, the analysis confirmed the relevance of the topic and allowed us to formulate a clear research task aimed at developing a holistic set of methods that eliminate the identified gaps.

To eliminate these shortcomings, a holistic set of methods has been developed, consisting of three key scientific results.

First, a hybrid method for short-term energy consumption forecasting based on the TCN-LightGBM architecture has been developed and improved. The method uses an effective strategy of "residual forecasting", where the temporal convolutional network (TCN) acts as a powerful basic forecaster, and the gradient boosting ensemble (LightGBM) learns to correct its errors (residuals) using an expanded set of tabular features. The scientific novelty lies in the development of a method for optimizing this hybrid model by purposefully selecting the most important features only for the corrector model. It has been experimentally proven that this approach allows to reduce the training time by 5.4 times (from 305.72 to 56.47 seconds), with practically no loss of accuracy in predicting critical peak loads (deterioration by only 0.06%). This solves the identified problem of trade-off between accuracy and computational cost.

Secondly, a method for adaptive cleaning of heterogeneous sensor data is developed. The problem of heterogeneous noise is formalized as a mixture of anomalies (outliers, drift, constant values). First, the ACRA (Adaptive Classification-based Real-time Anomaly cleaning) method is proposed, which combines the Random Forest classifier, a heuristic rule based on variance, and adaptive selection of correction

operators. On its basis, an improved hybrid architecture H-AD-CLEAN is created, which uses a much more powerful classifier (based on 1D-CNN + DWT) for parallel analysis of the "shape" and "texture" of the signal. The H-AD-CLEAN architecture implements robust "valve logic" (Gating Logic): it identifies "clean" data segments with high accuracy (F1-score 0.85) and protects them from processing (Pass-Through), while applying a powerful Kalman Filter to all identified types of noise. It has been experimentally proven that H-AD-CLEAN outperforms the non-adaptive Kalman Filter (by 21.1% according to the MAE metric), since it does not introduce distortions into valid data.

Thirdly, a proactive control architecture based on contextual intentions has been developed. The architecture implements the transition from low-level rules to the paradigm of high-level intentions (the user defines "what", not "how", e.g., "balance of comfort and economy"). The core of the system operates on the basis of multi-objective optimization of the utility function. The scientific novelty lies in the adaptive learning mechanism, which autonomously learns not a single global, but contextual intentions. The system analyzes the user's manual interventions as implicit feedback. When intervening, the system fixes the current context (e.g., "evening, high tariff") and incrementally adjusts the weights w_{comfort} and w_{energy} specifically for this context.

To validate the architecture, a simulation stand based on Gymnasium was developed, which simulates room thermodynamics, seasonal changes, dynamic tariffs, and a simulated user capable of generating context-dependent manual interventions.

The validity of the developments was confirmed by a comprehensive experimental study. The results of the 60-day simulation showed that the Proactive Agent achieved the highest level of comfort (41.5% of time in the target area) and, most importantly, reduced the need for manual user intervention by more than 97% compared to the basic rule-based model (28 interventions versus 240). Dynamics analysis confirmed that the system successfully learned complex contextual preferences (e.g., higher priority for comfort in the evening).

The described set of techniques eliminates the identified shortcomings of existing systems and creates such advantages as high forecasting accuracy on devices with

limited resources, reliability of input data and the ability of the system to autonomously adapt to residents, radically minimizing their cognitive load.

Keywords: smart home, Internet of Things (IoT), control optimization, proactive control, intent-based control, implicit feedback, multi-objective optimization, energy consumption forecasting, TCN, LightGBM, data cleaning, heterogeneous noise, ACRA, H-AD-CLEAN, simulation modeling, forecasting, data cleaning, machine learning.

Список опублікованих праць за темою дисертації

Матеріали й тези наукових конференцій

1. **Ніщепенко Д.О.** Розробка системи керування розумним будинком на основі протоколу MQTT. *IV Всеукраїнська студентська наукова конференція: «Експериментальні та теоретичні дослідження в контексті сучасної науки».* матеріали (м.Чернігів, 29 вересня 2023 р.), 2023. С. 154.

2. **Ніщепенко Д.О.** Протокол MQTT в додатку NESTJS: реалізація обміну повідомленнями. *Всеукраїнська науково-практична конференція: «Цифрова гуманістика: Інформаційні технології та інформаційне моделювання на сучасному етапі розвитку суспільства».* матеріали (м. Кропивницький, 4-5 червня 2024 р.), 2024. С. 119—123.

3. **Ніщепенко Д.О.** Безпека та захист обміну даними за допомогою протоколу MQTT для керування системами інтернету речей. *Всеукраїнська науково-практична конференція: «Актуальні проблеми безпеки інформаційно-телекомунікаційних систем».* збірник тез (м. Київ, 03 листопада 2024 р.), 2024. С. 98-99.

4. **Ніщепенко Д.О.** Роль мікроядра в енергоефективності операційних систем для інтернету речей. *Науково-практична конференція «Проблеми комп'ютерної інженерії».* Збірник тез (К., 2024), 2024. С. 200-202.

5. **Ніщепенко Д.О., Майборода М.В.** Порівняння ARIMA та LSTM у контексті прогнозування споживання енергії в умовах розумного будинку. *VI Міжнародна науково-практична конференція «Сучасні досягнення компанії Hewlett Packard Enterprise в галузі IT та нові можливості їх вивчення і застосування».* Збірник тез (К., 2024 р.), 2024. С. 22-24.

6. **Ніщепенко Д.О.** Реалізація збереження та управління даними в IoT-додатках на основі Java Spring Framework. *II Міжнародна науково-практична конференція «Сучасні аспекти діджиталізації та інформатизації в програмній та комп'ютерній інженерії».* Збірник тез (К., 2024 р.), 2024. С. 33–36.

7. **Ніщепенко Д.О.** Адаптивний метод ACRA для очищення різномірних сенсорних даних в системах розумного будинку. *II Всеукраїнська науково-*

практична конференція «Цифрова гуманістика: Інформаційні технології та інформаційне моделювання на сучасному етапі розвитку суспільства». матеріали (м. Кропивницький, травень 2025 р.), 2025. С 238-243.

8. **Волощук О.В., Ніщепенко Д.О.** Проблематика класифікації типів шумів для адаптивного очищення різномірних сенсорних даних. *XIV Міжнародна науково-практична конференція: «Математика. Інформаційні технології. Освіта»*. матеріали, 2025. С. 76-78.

9. **Nishchemenko D., Zhebka V., Shlianchak S., Popereshnyak S.** Real-time adaptive cleaning of IoT sensor data using machine learning noise classification and rule-based refinement. *MoMLet 2025 Modern Machine Learning Technologies Workshop*. CEUR Workshop Proceedings (2025). *Scopus*

10. **Nishchemenko D., Dubinin V., Panasenko Y., Volynets S., Zhebka V.** Analyzing the Rationality of using Different Merkle Tree Constructions in Blockchain-based Accounting Systems *DECaT 2025 Digital Economy Concepts and Technologies Workshop*. CEUR Workshop Proceedings (2025). *Scopus*

Статті в наукових фахових виданнях

1. **Ніщепенко Д.О., Аронов А.О.** Дослідження методик оптимізації параметрів системи керування розумним будинком з використанням ІОТ. *Телекомунікаційні та інформаційні технології*, №1, 2025. С. 141-150.

2. **Ніщепенко Д.О., Волощук О.В.** Адаптивне очищення різномірних сенсорних даних у системах розумного будинку на основі класифікації шумів. *Електронне фахове наукове видання «Кібербезпека: освіта, наука, техніка»*, 4(28), 2025. С. 740–750.

3. **Ніщепенко Д.** Оптимізація гібридних моделей на основі TCN, LSTM, LIGHTGBM для прогнозування енергоспоживання в розумних будинках. *Електронне фахове наукове видання «Кібербезпека: освіта, наука, техніка»*, 2(30), 2025. С. 224–237.

4. **Ніщепенко Д.О., Олейніков І.А.** Проактивна архітектура керування розумним будинком на основі контекстуальних намірів користувача та

багатоцільової оптимізації. *Телекомунікаційні та інформаційні технології*, № 3, 2025. С. 141-149.

5. Ніщепенко Д.О., Аронов А.О., Гавор А.С., Герцюк, М.М., Гордієнко, К.О. Аналіз мов парадигми ООП для реалізації масштабованих систем із Docker *Наука і техніка сьогодні*, 10(51), 2025 С. 1406–1418.

ЗМІСТ

ВСТУП	14
РОЗДІЛ 1. АНАЛІЗ СУЧАСНОГО СТАНУ ПРОБЛЕМИ КЕРУВАННЯ СИСТЕМАМИ РОЗУМНОГО БУДИНКУ	18
1.1. Концепція Інтернету речей та її роль у системах розумного будинку.	18
1.2. Огляд та аналіз методів прогнозування часових рядів для задач енергоспоживання.	27
1.3. Аналіз методів очищення сенсорних даних у режимі реального часу.	34
1.4. Аналіз архітектур та парадигм керування розумним будинком.	40
1.5. Постановка завдання дисертаційного дослідження	47
Висновки до розділу 1	50
РОЗДІЛ 2. ВДОСКОНАЛЕННЯ ГІБРИДНОГО МЕТОДУ ПРОГНОЗУВАННЯ ЕНЕРГОСПОЖИВАННЯ.....	52
2.1. Теоретичні основи темпоральних згорткових мереж (TCN).....	52
2.2. Теоретичні основи градієнтного бустингу LightGBM	60
2.3. Розробка гібридної моделі TCN-LightGBM для прогнозування енергоспоживання.	66
2.4. Розробка методики оптимізації моделі для досягнення балансу точності та обчислювальної ефективності.	73
Висновки до розділу 2	79
РОЗДІЛ 3. МЕТОД АДАПТИВНОГО ОЧИЩЕННЯ РІЗНОРІДНИХ СЕНСОРНИХ ДАНИХ (ACRA)	81
3.1. Формалізація проблеми гетерогенних шумів у сенсорних даних розумного будинку.....	81
3.2. Розробка та алгоритм роботи методу ACRA.	88
3.3. Вдосконалена гібридна архітектура очищення даних HAD-CLEAN на основі глибокого навчання та вентиляційної логіки.....	96
3.4. Алгоритм роботи методу H-AD-CLEAN у режимі реального часу....	101
Висновки до розділу 3	104
РОЗДІЛ 4. ПРОАКТИВНА АРХІТЕКТУРА КЕРУВАННЯ НА ОСНОВІ КОНТЕКСТУАЛЬНИХ НАМІРІВ	106
4.1. Теоретичні основи проактивного керування на основі контекстуальних намірів.....	106
4.2. Розробка компонентної архітектури системи проактивного керування.	111
4.3. Математична модель прийняття рішень на основі багатоцільової оптимізації.....	118

4.4. Механізм адаптивного навчання контекстуальним намірам на основі неявного зворотного зв'язку.....	121
Висновки до розділу 4.	125
РОЗДІЛ 5. ЕКСПЕРИМЕНТАЛЬНЕ ДОСЛІДЖЕННЯ ТА АПРОБАЦІЯ РЕЗУЛЬТАТІВ.....	127
5.1. Розробка імітаційного стенду та методика проведення експериментів.....	127
5.2. Експериментальна перевірка ефективності вдосконаленого методу прогнозування TCN-LightGBM.	143
5.3. Апробація та порівняльний аналіз методів адаптивного очищення даних ACRA та H-AD-CLEAN.	150
5.3.1. Порівняльний аналіз якості очищення для різних типів сенсорних даних.....	150
5.3.2. Порівняльний аналіз ефективності внутрішніх класифікаторів ..	155
5.3.3. Порівняльний аналіз фінальної якості очищення.....	158
5.4. Валідація проактивної архітектури керування шляхом імітаційного моделювання.....	161
5.4.1. Порівняльний аналіз ефективності з системами на основі правил	162
5.4.2. Аналіз динаміки адаптації системи до намірів користувача.....	165
Висновки до розділу 5.	168
ЗАГАЛЬНІ ВИСНОВКИ	172
СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ.....	175
Додаток А. Лістинг програмного коду реалізації класу гібридної моделі прогнозування TCN-LightGBM (метод прогнозування залишків).....	192
Додаток Б. Лістинг програмного коду реалізації алгоритму адаптивного очищення даних H-AD-CLEAN (CNN+DWT класифікатор та механізм Gating) ..	197
Додаток В. Лістинг програмного коду реалізації класу проактивного агента та компонентів адаптації	202
Додаток Г. Структурна схема апаратно-програмного комплексу	208
Додаток Д. Повні результати порівняння точності моделей ARIMA, LSTM, GRU, TCN, TCN- XGBoost, TCN-LightGBM.....	210
Додаток Е. Акти впровадження	214

ВСТУП

Актуальність теми. Інтенсивний розвиток та імплементація технологій Інтернету речей (IoT) в інфраструктуру житлових об'єктів зумовлюють трансформацію систем керування будівлями та перехід до концепції розумного будинку (Smart Home). Функціонування таких систем супроводжується генерацією значних обсягів гетерогенних даних, отриманих від розподіленої сенсорної мережі, що створює передумови для підвищення рівня автоматизації, безпеки та енергоефективності. Враховуючи, що житловий сектор є одним із критичних споживачів енергоресурсів, розробка та впровадження інтелектуальних методів керування енергоспоживанням набуває пріоритетного значення як з економічної, так і з екологічної точок зору.

Разом з тим, аналіз існуючих рішень свідчить, що ефективність функціонування систем розумного будинку обмежується низкою невирішених проблем, пов'язаних з якістю даних, точністю предиктивної аналітики та адаптивністю архітектури керування.

По-перше, якість вхідних даних. Сенсорні потоки в мережах IoT характеризуються наявністю складних комбінацій шумів та аномалій (викиди, дрейф сенсорів, константні значення), що суттєво знижує достовірність інформаційної бази для прийняття рішень.

По-друге, складність прогнозування. Процеси енергоспоживання мають виражений стохастичний, нелінійний та нестаціонарний характер, що ускладнює застосування класичних моделей для високоточного короткострокового прогнозування на пристроях з обмеженими обчислювальними ресурсами.

По-третє, обмеженість методів керування. Більшість поширених систем базуються на реактивних парадигмах та статичних наборах правил, які не забезпечують адаптацію до динамічних змін контексту та індивідуальних патернів поведінки користувачів, що призводить до неоптимального розподілу ресурсів та необхідності частого ручного втручання в роботу системи.

Таким чином, розробка комплексу методів і моделей оптимізації, спрямованих на попередню обробку сенсорних даних, підвищення точності прогнозування навантажень та реалізацію проактивного, контекстно-залежного керування, є актуальним науково-прикладним завданням.

Зв'язок роботи з науковими програмами, планами, темами. Дисертаційна робота була виконана в рамках:

- науково-дослідної роботи «Актуальні питання сучасної інформатики та інформаційних технологій в освіті та науці» (Державний реєстраційний номер 0124U001430), Центральноукраїнського державного університету імені Володимира Винниченка;

- науково-дослідної роботи "Підвищення ефективності процесу управління 3D принтером з використанням методів машинного навчання" (Державний реєстраційний номер РК 0124U001849), кафедри Технологій цифрового розвитку Державного університету інформаційно-комунікаційних технологій.

Мета і задачі дослідження. Метою дисертаційної роботи є оптимізація керування розумним будинком на основі Інтернету речей за допомогою розробленого комплексу методів, що дозволяють підвищити якість даних, точність прогнозування енергоспоживання та ефективність системи керування за рахунок адаптації до контекстуальних намірів користувача.

Для досягнення поставленої мети було визначено наступні **задачі**:

1. Проаналізувати сучасні підходи до керування системами розумного будинку, виявити їхні обмеження та обґрунтувати напрямки вдосконалення.
2. Вдосконалити метод короткострокового прогнозування енергоспоживання на основі гібридної моделі TCN-LightGBM та розробити методику його оптимізації для досягнення балансу між точністю та обчислювальною ефективністю.
3. Розробити метод адаптивного очищення різнорідних сенсорних даних у режимі реального часу (ACRA), здатний ідентифікувати та усувати гетерогенні типи шумів на основі класифікації та правил.

4. Розробити проактивну архітектуру керування розумним будинком, що функціонує на основі багатоцільової оптимізації та здатна автономно навчатися контекстуально-залежним намірам користувача.
5. Провести комплексне експериментальне дослідження розроблених методик за допомогою імітаційного моделювання для підтвердження їх ефективності.

Об'єкт дослідження – процес функціонування інтелектуальної системи керування розумним будинком на основі даних з мережі IoT-сенсорів.

Предмет дослідження – методи та моделі прогнозування енергоспоживання, очищення сенсорних даних та архітектури проактивного керування для систем розумного будинку.

Методи дослідження. Для вирішення поставлених задач використовувалися: методи системного аналізу та теорії керування для формалізації архітектур; методи машинного та глибокого навчання (темпоральні згорткові мережі, ансамблі дерев рішень) для розробки моделей прогнозування та очищення даних; методи багатоцільової оптимізації для реалізації ядра прийняття рішень; методи імітаційного моделювання для перевірки ефективності розроблених підходів.

Наукова новизна одержаних результатів полягає в наступному:

1. Удосконалено гібридний метод короткострокового прогнозування енергоспоживання на основі архітектури TCN-LightGBM, який відрізняється від існуючих застосуванням стратегії цілеспрямованого відбору ознак виключно для моделі-коректора, що дозволило вирішити проблему компромісу між точністю та обчислювальною вартістю, забезпечивши прискорення навчання в 5,4 рази при збереженні високої точності прогнозування пікових навантажень на пристроях з обмеженими ресурсами.

2. Розроблено метод адаптивного очищення сенсорних даних (ACRA/H-AD-CLEAN), наукова новизна якого полягає у поєднанні класифікатора на основі машинного навчання для ідентифікації специфічних типів шумів (викиди, дрейф, константні значення) з евристичним правилом на основі дисперсії. Такий підхід,

на відміну від відомих, дозволяє реалізувати адаптивний вибір оператора корекції для різнорідних потоків даних у режимі реального часу.

3. Набув подальшого розвитку метод проактивного керування розумним будинком (або людино-орієнтованого керування), який відрізняється переходом від навчання глобальним намірам до навчання контекстуальним намірам користувача (залежним від часу доби та тарифу), реалізовано шляхом аналізу ручних втручань користувача як неявного зворотного зв'язку для адаптивної корекції вагових коефіцієнтів у задачі багатоцільової оптимізації.

Практичне значення одержаних результатів. Розроблений комплекс методів дозволяє створювати гнучкі, адаптивні системи керування розумним будинком нового покоління. Оптимізована модель прогнозування TCN-LightGBM може бути впроваджена на пристроях з обмеженими ресурсами для ефективного управління піковими навантаженнями. Метод очищення ACRA підвищує надійність та якість вхідних даних, що є основою для коректної роботи всіх інтелектуальних додатків системи. Проактивна архітектура керування дозволяє створювати системи, що самостійно адаптуються до поведінки мешканців, значно зменшуючи потребу в ручному втручанні (на понад 97%) та підвищуючи рівень комфорту при збереженні енергоефективності.

Структура та обсяг дисертації. Дисертаційна робота складається з анотації, змісту, вступу, п'яти розділів, загальних висновків, списку використаних джерел та додатків. Робота містить 41 рисунок, 19 таблиць та 26 сторінок додатків. Список використаних джерел налічує 143 найменування.

Загальний обсяг дисертації становить 217 сторінок, з них 162 сторінки основного тексту.

РОЗДІЛ 1. АНАЛІЗ СУЧАСНОГО СТАНУ ПРОБЛЕМИ КЕРУВАННЯ СИСТЕМАМИ РОЗУМНОГО БУДИНКУ

1.1. Концепція Інтернету речей та її роль у системах розумного будинку.

Інтернет речей (Internet of Things, IoT) є сучасною технологічною парадигмою, яка описує глобальну мережу взаємопов'язаних фізичних об'єктів, що інтегровані з сенсорами, актуаторами, програмним забезпеченням та іншими комунікаційними технологіями, що надає їм можливість підключатися та автоматично обмінюватися даними з іншими пристроями і системами через глобальну мережу Інтернет. Така концепція радикально трансформує традиційні механізми взаємодії з фізичним світом, перетворюючи раніше пасивні сутності на активних учасників складних інформаційно-керуючих процесів [67, 69].

Інтеграція інтелектуальних технологій у житловий простір, відома як концепція «розумного будинку» (Smart Home), є однією з найбільш репрезентативних галузей практичного застосування IoT, що активно переходить від дослідницьких концепцій до повсякденної експлуатації.

Фундаментальна роль Інтернету речей у системах «розумного будинку» полягає у формуванні єдиної, динамічної та адаптивної екосистеми, де відбувається безшовна взаємодія між гетерогенними побутовими приладами, кліматичним обладнанням, системами освітлення, моніторингу та безпеки, забезпечуючи їхню автоматичну координацію та ефективну комунікацію як між собою, так і з кінцевим користувачем.

Враховуючи, що житловий сектор є одним із провідних глобальних споживачів енергетичних ресурсів, питання підвищення його енергоефективності набуває стратегічної та невідкладної актуальності. У цьому контексті, впровадження інтелектуальних систем керування на базі IoT відкриває принципово нові можливості для оптимізації енергоспоживання [3, 10, 24].

Такі рішення дозволяють реалізувати не лише скорочення фінансових витрат домогосподарств завдяки автоматичному регулюванню використання електроенергії, теплопостачання чи освітлення відповідно до реальних потреб та зовнішніх умов, але й роблять вагомий внесок у зниження пікового навантаження на централізовані енергетичні мережі та скорочення емісії шкідливих речовин у довкілля [27].

Впровадження систем «розумний будинок» в Україні відбувається в умовах формування відповідного нормативно-правового поля, яке визначає стратегічні пріоритети цифровізації економіки та суспільства.

Основоположним вектором розвитку вітчизняного ІТ-сектору є курс на інноваційну діяльність. Згідно з розпорядженням Кабінету Міністрів України «Про затвердження Стратегії розвитку сфери інноваційної діяльності на період до 2030 року», пріоритетними напрямками визначено розвиток високотехнологічних виробництв, цифровізацію економічних процесів та підтримку розробок у сфері штучного інтелекту і обробки великих даних [13]. Це створює необхідне підґрунтя для розробки нових методик керування інтелектуальними системами, які здатні забезпечити енергоефективність та комфорт користувачів на якісно новому рівні.

Важливо зазначити, що системи розумного будинку не можна розглядати як ізольовані об'єкти. Вони виступають базовими структурними елементами більш масштабних екосистем. У «Концепції розвитку технологій Smart City в Україні», затвердженій Міністерством цифрової трансформації у 2024 році, підкреслюється, що інтелектуалізація житлової інфраструктури є критично важливою складовою сталого розвитку урбанізованих територій. Концепція передбачає впровадження систем моніторингу енергоресурсів та автоматизованого керування комунальними послугами, що вимагає розробки адаптивних алгоритмів обробки даних, які генеруються сенсорними мережами IoT [7, 17].

Функціонування розподілених сенсорних мереж та обмін даними між компонентами системи розумного будинку (сенсорами, контролерами, хмарними сервісами) регламентується законодавством у сфері телекомунікацій. Закон

України «Про електронні комунікації» визначає правові та організаційні засади функціонування електронних комунікаційних мереж, забезпечуючи уніфікацію стандартів передачі даних та доступність спектру радіочастот для пристроїв IoT. Зазначений закон створює правову основу для розгортання надійних каналів зв'язку, що є критичним фактором для забезпечення стабільності процесів керування в реальному часі [12].

Окремої уваги потребує питання інформаційної безпеки та цілісності даних, що циркулюють у системах керування розумним будинком. Оскільки методики оптимізації керування базуються на аналізі персональних даних користувачів та патернів їхньої поведінки, забезпечення конфіденційності та захисту від несанкціонованого втручання є обов'язковою вимогою. У цьому контексті розробка програмного забезпечення та архітектурних рішень повинна відповідати положенням ДСТУ ISO/IEC 27001:2015 «Інформаційні технології. Методи та засоби безпеки. Системи управління інформаційною безпекою. Вимоги». Цей стандарт встановлює критерії оцінки ризиків та вимоги до побудови захищених інформаційних систем, що є необхідною умовою для валідації та практичного впровадження пропонованих у дисертаційній роботі методів [4].

Таким чином, аналіз нормативної бази свідчить про те, що розробка методів оптимізації керування розумним будинком є актуальним науково-технічним завданням, яке відповідає стратегічним цілям держави у сферах інноваційного розвитку, розбудови розумної інфраструктури та кібербезпеки [14, 15].

Архітектури IoT визначають фундаментальну парадигму того, де і як збираються, передаються, обробляються та зберігаються дані. Вибір архітектурної моделі є стратегічним рішенням, що безпосередньо впливає на ключові нефункціональні характеристики системи, такі як латентність, вимоги до пропускної здатності мережі, рівень конфіденційності, відмовостійкість та загальна вартість експлуатації. Сучасні дослідження та індустріальні практики виокремлюють три основні моделі, що являють собою, по суті, різні рівні в ієрархії обчислень [9].

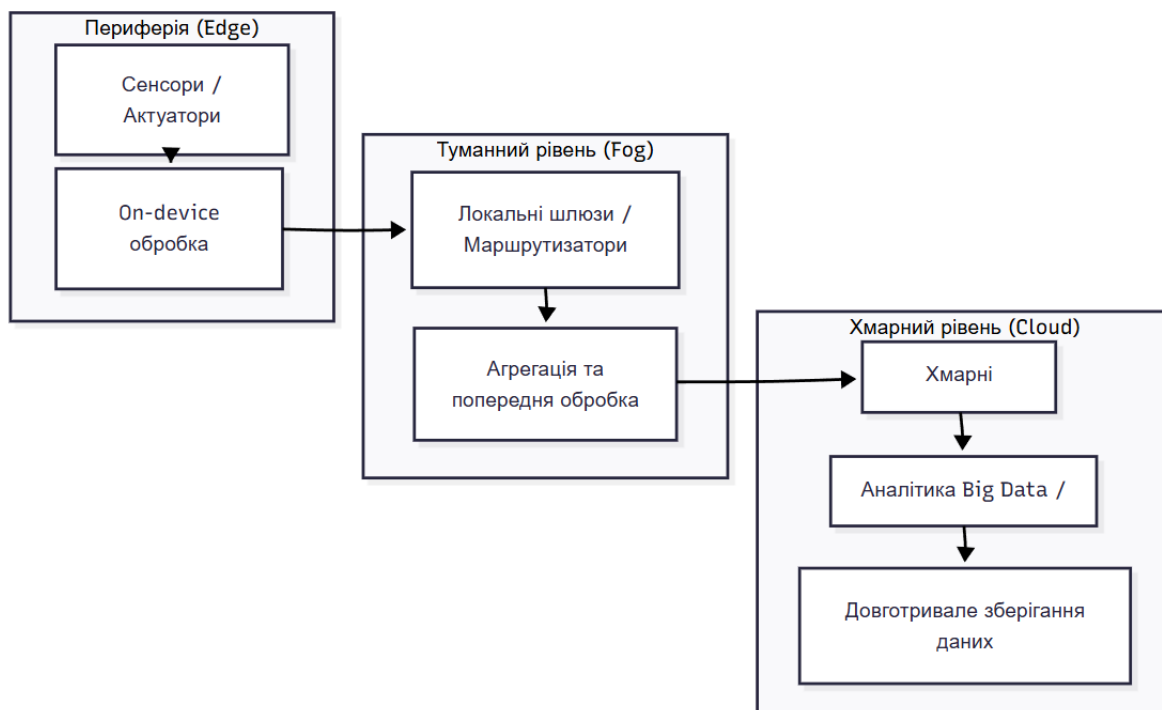


Рисунок 1.1: Трирівнева архітектура обробки даних у системах розумного будинку

Хмарна архітектура розглядається як класичний централізований підхід до організації обчислювальних процесів, у межах якого всі дані від кінцевих сенсорів та пристроїв розумного будинку передаються до віддалених хмарних платформ. У такій моделі локальні вузли виконують лише роль засобів агрегації та транспортування телеметрії, тоді як основні операції з обробки, аналітики, прийняття рішень і тривалого зберігання даних реалізуються у хмарних дата-центрах. До переваг цього підходу належать практично необмежені обчислювальні ресурси та ємність сховищ, що створює сприятливі умови для застосування складних алгоритмів машинного навчання та аналізу великих історичних вибірок. Водночас використання суто хмарної моделі супроводжується істотними недоліками, зокрема високою латентністю, залежністю від стабільності інтернет-з'єднання, значними обсягами вихідного трафіку та ризиками для конфіденційності користувацьких даних, які повністю залишають межі локального середовища.

Альтернативою повній централізації стала концепція туманних обчислень, що пропонує проміжний рівень обробки між периферійними пристроями та хмарною інфраструктурою. Туманне середовище базується на використанні локальних, але достатньо продуктивних вузлів, таких як IoT-шлюзи або спеціалізовані сервери, які здійснюють попередню обробку, фільтрацію та агрегацію даних безпосередньо поблизу джерела їх виникнення. Такий підхід дає змогу суттєво зменшити затримки, оптимізувати трафік та скоротити навантаження на магістральні канали зв'язку, оскільки до хмари надсилаються лише узагальнені чи репрезентативні дані. Завдяки цьому підвищується оперативність реагування системи, що є необхідним фактором для забезпечення належного рівня комфорту та функціональності розумного будинку [23].

Найбільш децентралізованим варіантом організації обчислень є периферійна архітектура, у межах якої обробка даних відбувається безпосередньо на кінцевих пристроях або в найближчих локальних вузлах. Такий підхід мінімізує латентність та забезпечує детерміновану швидкість реакції системи, що є особливо важливим для критичних застосунків: систем безпеки, виявлення аварійних ситуацій, автоматизованого керування освітленням та інших процесів, що потребують миттєвої взаємодії з фізичним середовищем. Периферійна модель сприяє підвищенню відмовостійкості, оскільки система зберігає працездатність навіть за відсутності зовнішнього мережевого підключення, а також покращує рівень конфіденційності, оскільки чутливі дані можуть не покидати меж локальної інфраструктури.

У практичних реалізаціях сучасні інтелектуальні системи для розумних будинків зазвичай не покладаються на одну архітектурну модель у чистому вигляді. Найбільш ефективним вважається формування гібридних ієрархічних архітектур, що поєднують елементи хмарних, туманних і периферійних підходів. Така інтеграція дає змогу досягти збалансованого поєднання гнучкості, масштабованості, продуктивності та надійності, оптимізуючи функціонування системи відповідно до конкретних вимог і умов експлуатації [108].

Розглянуті архітектурні моделі Інтернету речей порівняно у Таблиці 1.1.

Таблиця 1.1

Порівняльний аналіз архітектурних моделей IoT для розумного будинку

Характеристика	Хмарна (Cloud)	Туманна (Fog)	Периферійна (Edge)
Латентність	Висока, повний цикл до хмари	Середня, локальна обробка на шлюзі	Мінімальна, миттєві реакції
Приватність даних	Низька, дані виходять з будинку	Середня, часткове зберігання локально	Висока, дані не покидають мережу
Залежність від інтернету	Повна	Часткова	Майже відсутня
Обчислювальна потужність	Дуже висока	Середня	Низька–середня
Типові задачі	Big Data, тренування моделей	Сценарії кімнат, агрегація	Миттєве керування, безпека

Таким чином, периферійний рівень (Edge) відповідає за миттєві реакції та базову автоматизацію, наприклад, ввімкнення світла за сигналом датчика руху. Туманний рівень (Fog) керує виконанням складних локальних сценаріїв, що охоплюють кілька пристроїв, та агрегує дані. Хмарний рівень (Cloud) використовується для довготривалого зберігання даних, тренування складних аналітичних моделей та надання сервісів віддаленого доступу для користувача.

Окремим перспективним напрямком є розробка повністю децентралізованих архітектур на основі технології блокчейн. Такі системи спрямовані на вирішення фундаментальних проблем централізованих моделей, пов'язаних з безпекою, прозорістю та довірою до даних. Використання розподіленого незмінного реєстру дозволяє створювати надійні механізми для автентифікації пристроїв, контролю доступу та аудиту даних без необхідності покладатися на єдиного довіреного хмарного провайдера.

Література з оптимізації систем керування розумним будинком на основі Інтернету речей (IoT) розкриває кілька спільних напрямків досліджень, що охоплюють архітектурні проекти, протоколи зв'язку та стратегії управління даними, спрямовані на підвищення енергоефективності та комфорту користувачів.

Акцент робиться на використанні машинного навчання та фреймворків на основі штучного інтелекту для прогнозування споживання енергії та виявлення аномалій, а також на надійному очищенні даних датчиків та механізмах відмовостійкості для забезпечення надійності системи.

Інтеграція передових обчислювальних парадигм, таких як туманні та периферійні обчислення, а також блокчейн, лежить в основі зусиль щодо масштабування та безпеки розгортання IoT для розумного будинку. Орієнтовані на користувача підходи, що використовують петлі зворотного зв'язку та цифрових двійників, ще більше підкреслюють взаємодію між технологіями та поведінкою мешканців в оптимізації управління енергією в розумному будинку [47].

Еволюція систем керування розумним будинком, що використовують технології Інтернету речей, значно просунулася від початкових концепцій профілювання енергії та розгортання датчиків до складних фреймворків на основі штучного інтелекту, що інтегрують машинне навчання, цифрових двійників та передові протоколи зв'язку. Ранні роботи були зосереджені на базовому моделюванні поведінки користувачів, моніторингу енергії та базових протоколах зв'язку, що стосуються сумісності систем та збору даних. З часом акцент досліджень змістився на оптимізацію споживання енергії за допомогою прогнозової аналітики, навчання з підкріпленням та інтеграції туманних/хмарних обчислень, зі зростанням уваги до безпеки, надійності та орієнтованої на користувача адаптації [33].

Існуючі дослідження демонструють цілісні фреймворки, що поєднують штучний інтелект, периферійні обчислення, блокчейн та моделі великих мов програмування для підвищення масштабованості системи, якості даних, відмовостійкості та персоналізованого управління енергією в розумних будинках.

У проаналізованій літературі підкреслюється критична роль інтегрованих архітектур Інтернету речей та різноманітних протоколів зв'язку в оптимізації систем керування розумним будинком. Масштабовані фреймворки, що включають периферійні, туманні та хмарні обчислення, стали ефективними стратегіями для покращення реагування в режимі реального часу, зниження

навантаження на мережу та забезпечення безперешкодної інтеграції різнорідних пристроїв.

Такі протоколи, як Zigbee, MQTT, Wi-Fi та OCF, демонструють високу продуктивність у балансуванні низької затримки, пропускну здатності та сумісності, проте проблеми залишаються через неоднорідність пристроїв та відсутність стандартизованих фреймворків. Нові підходи, що використовують адаптивний зв'язок та управління перешкодами, ще більше підвищують ефективність протоколів, але підкреслюють постійну потребу в надійних, гібридних рішеннях [25, 89].

Покращення якості даних залишається ключовим питанням, оскільки багатосенсорне об'єднання, виявлення аномалій та передові методи очищення є необхідними для надійної роботи системи [19]. Основою функціонування будь-якої інтелектуальної системи розумного будинку є безперервні потоки даних, що надходять від численних сенсорів. Однак якість цих необроблених даних часто є низькою. Сенсорні потоки містять складні комбінації шумів та аномалій, включаючи випадкові викиди, поступовий дрейф показників, періоди незмінних («залиплих») значень та загальний фоновий шум. Такі артефакти можуть бути спричинені несправностями сенсорів, електромагнітними перешкодами або непередбачуваним впливом середовища.

Низька якість даних суттєво знижує надійність системи, що може призводити до помилкових рішень, неефективного управління та, як наслідок, до погіршення комфорту мешканців та невиправданих енерговитрат. Тому очищення даних (data cleaning) та забезпечення їх цілісності є фундаментальним і критично важливим етапом, що передуює будь-якому подальшому аналізу, прогнозуванню чи керуванню [20].

Відсутність надійних, адаптивних методів очищення, здатних працювати в режимі реального часу з різнорідними типами шумів, є однією з ключових невирішених проблем, що стримує розвиток по-справжньому ефективних систем розумного будинку.

Прогнозування споживання енергії значно виграло від гібридних моделей та моделей глибокого навчання, включаючи комбінації LSTM, TCN, CNN та ARIMA, досягаючи високої точності прогнозування та підтримуючи адаптивне управління енергією. Інтеграція поведінки мешканців та екологічного контексту в моделі прогнозування підвищує точність та дозволяє використовувати більш орієнтовані на користувача стратегії управління. Однак, можливість узагальнення цих моделей часто не перевіряється в різних умовах, а їхні обчислювальні вимоги ускладнюють реалізацію в режимі реального часу на пристроях з обмеженими ресурсами. Поєднання прогнозування з багатоцільовою оптимізацією сприяє балансуванню енергоефективності, вартості та комфорту користувача, але вимагає подальшого вдосконалення для прозорого управління компромісами [26].

Надійність системи та відмовостійкість підсилюються завдяки локалізованій обробці, адаптивному навчанню та багаторівневим конвеєрам обробки даних, які разом забезпечують надійну роботу в умовах несправностей датчиків та перебоїв у роботі мережі. Структури периферійних та туманних обчислень сприяють покращенню реагування та зменшенню залежності від централізованих серверів. Незважаючи на ці досягнення, підтримка стабільної надійності на гетерогенних пристроях та складних мережах залишається складною, оскільки вразливості безпеки та складність інтеграції продовжують підривати стабільність системи.

Систематичний аналіз недавніх досліджень, що використовують ієрархічні, периферійні та туманні обчислення, виявив масштабовані фреймворки, які покращують швидкість реагування в режимі реального часу, але стикаються з обмеженнями сумісності через неоднорідність пристроїв.

Оцінки протоколів висвітлили багатопроTOCOLьну інтеграцію та адаптивні методи зв'язку, що покращують надійність передачі даних, хоча прогалини у стандартизації зберігаються.

Покращення якості даних було досягнуто завдяки вдосконаленому виявленню аномалій, мультисенсорному об'єднанню даних та децентралізованому управлінню даними, проте реальна валідація та обчислювальні витрати залишаються проблемою.

Моделі прогнозування споживання енергії, засновані на гібридних методах машинного навчання, продемонстрували високу точність прогнозування та підтримували адаптивне керування, хоча узагальнюваність та обмеження ресурсів обмежують розгортання.

У межах концепції Інтернету речей особливу увагу приділяють забезпеченню надійності та безперервності функціонування систем розумного будинку. Для досягнення цієї мети активно впроваджуються стратегії, що поєднують локалізовану (edge) обробку даних, технології блокчейну та інтелектуальні методи виявлення несправностей на основі штучного інтелекту. Такі підходи дозволяють підвищити стійкість і автономність систем, мінімізуючи залежність від центральних серверів і зменшуючи ризики збоїв або кібератак. Водночас, попри значні досягнення в цій сфері, зазначені технології потребують подальшого вдосконалення, особливо у контексті масштабних гетерогенних середовищ, де взаємодіють численні пристрої з різними характеристиками, протоколами та рівнем обчислювальних можливостей. Ефективна інтеграція таких рішень є ключовою умовою для розвитку більш безпечних, надійних та адаптивних систем розумного житла [4].

Загалом, висновки підкреслюють необхідність цілісних, орієнтованих на користувача підходів, які інтегрують масштабовані архітектури, надійні протоколи та розширену аналітику для покращення управління енергією в розумному будинку. Огляд визначає напрямки майбутніх досліджень, спрямованих на покращення сумісності, цілісності даних та точності прогнозування в різних житлових екосистемах Інтернету речей.

1.2. Огляд та аналіз методів прогнозування часових рядів для задач енергоспоживання.

Точне короткострокове прогнозування навантаження (Short-Term Load Forecasting, STLF) є фундаментальним завданням для забезпечення стабільності функціонування сучасних енергосистем, оптимізації управління енергоресурсами,

підтримки прийняття економічно обґрунтованих рішень у диспетчеризації та інтеграції відновлюваних джерел енергії. Помилки у прогнозуванні, навіть на рівні 1%, можуть призводити до значних фінансових втрат, що підкреслює критичну важливість розробки високоточних та надійних алгоритмів. Впродовж останнього десятиліття методи прогнозування енергоспоживання зазнали значної еволюції, перейшовши від класичних статистичних моделей до прогресивних підходів, що базуються на машинному та глибокому навчанні. Цей розвиток зумовлений зростаючою складністю патернів споживання, викликану появою нових потужних споживачів (електромобілі, теплові насоси) та інтеграцією непередбачуваної генерації, а також необхідністю обробки великих обсягів даних з високою часовою роздільною здатністю, що надходять від систем інтелектуального обліку.

Класичні статистичні підходи (ARIMA, SARIMA) довгий час були основним інструментом для прогнозування часових рядів енергоспоживання. Серед них найпоширенішими є моделі ARIMA (Autoregressive Integrated Moving Average) та її сезонна варіація SARIMA [16, 58].

Модель ARIMA описує часовий ряд як комбінацію авторегресійної частини, інтегрованої компоненти (диференціювання) та ковзного середнього. Її загальний вигляд записують так:

$$\left(1 - \sum_{i=1}^p \phi_i L^i\right) (1 - L)^d X_t = \left(1 + \sum_{j=1}^q \theta_j L^j\right) \varepsilon_t, \quad (1.1)$$

де L – оператор зсуву,

ϕ_i – коефіцієнти авторегресії,

θ_j – коефіцієнти ковзного середнього.

Модель SARIMA розширює ARIMA, додаючи сезонні компоненти для врахування періодичних коливань (добових, тижневих, річних), що є характерними для енергоспоживання.

$$\Phi(L^S)\phi(L)(1 - L)^d(1 - L^S)^D X_t = \Theta(L^S)\theta(L)\varepsilon_t, \quad (1.2)$$

де $\phi(L)$ – несезонна AR частина,

$\theta(L)$ – несезонна МА частина,

$\Phi(L^S)$ – сезонна AR частина,

$\Theta(L^S)$ – сезонна МА частина.

Перевагами цих підходів є їхня відносна простота, висока інтерпретованість коефіцієнтів, міцна теоретична обґрунтованість та ефективність для рядів з чітко вираженою лінійною структурою та сезонністю. Однак їхній фундаментальний недолік полягає у припущенні про лінійність та стаціонарність часового ряду або можливості досягнення стаціонарності шляхом диференціювання. Реальні дані про енергоспоживання є більш складними: нелінійними, нестаціонарними та волатильними. На них впливає ряд зовнішніх факторів, таких як: погодні умови (температура, вологість, сонячна радіація), економічна активність, свята, соціальні події та поведінка споживачів. Класичні статистичні моделі недостатньо гнучкі, щоб ефективно вловлювати ці складні, нелінійні взаємозв'язки, що суттєво обмежує їхню точність у сучасних динамічних умовах.

З розвитком обчислювальних потужностей та алгоритмів штучного інтелекту, моделі машинного та глибокого навчання стали домінуючим напрямком у сфері STLF. Вони здатні автоматично вивчати складні нелінійні залежності та ієрархічні ознаки безпосередньо з сирих даних. Особливу увагу привернули архітектури, розроблені спеціально для обробки послідовних даних.

Мережі з довгою короткостроковою пам'яттю (Long Short-Term Memory, LSTM) є типом рекурентних нейронних мереж (RNN), спеціально розробленим для подолання проблеми зникаючого градієнта та ефективного вивчення довгострокових часових залежностей. Завдяки своїй складній структурі з вентильними механізмами (input, forget, output gates), LSTM здатні вибірково зберігати та забувати інформацію впродовж тривалих періодів, що робить їх дуже ефективними для моделювання складних патернів енергоспоживання. Численні дослідження підтверджують їхню високу точність у задачах STLF [28, 30, 32].

Forget gate (вентиль забування)

$$f_t = \sigma(W_f x_t + U_f h_{t-1} + b_f), \quad (1.3)$$

Input gate (вентиль оновлення)

$$i_t = \sigma(W_i x_t + U_i h_{t-1} + b_i), \quad (1.4)$$

Candidate memory

$$\tilde{c}_t = \tanh(W_c x_t + U_c h_{t-1} + b_c), \quad (1.5)$$

Cell state update

$$c_t = f_t \odot c_{t-1} + i_t \odot \tilde{c}_t, \quad (1.6)$$

Output gate

$$o_t = \sigma(W_o x_t + U_o h_{t-1} + b_o), \quad (1.7)$$

Hidden state

$$h_t = o_t \odot \tanh(c_t), \quad (1.8)$$

Вентильні рекурентні блоки (Gated Recurrent Units, GRU) є спрощеною варіацією LSTM з меншою кількістю параметрів, що об'єднує вентиля забування та входу.

Update gate

$$z_t = \sigma(W_z x_t + U_z h_{t-1} + b_z), \quad (1.9)$$

Reset gate

$$r_t = \sigma(W_r x_t + U_r h_{t-1} + b_r), \quad (1.10)$$

Candidate state

$$\tilde{h}_t = \tanh(W_h x_t + U_h (r_t \odot h_{t-1}) + b_h), \quad (1.11)$$

Final state

$$h_t = (1 - z_t) \odot h_{t-1} + z_t \odot \tilde{h}_t, \quad (1.12)$$

GRU часто демонструють порівнянню з LSTM точність, але є обчислювально ефективнішими (швидшими у навчанні), що робить їх більш пріоритетним вибором, особливо для систем реального часу.

Темпоральні згорткові мережі (Temporal Convolutional Networks, TCN) представляють потужну альтернативу рекурентним архітектурам. Принцип їх роботи полягає у використанні розширених причинно-наслідкових згорток, що дозволяє моделювати довгострокові залежності з експоненційно зростаючим рецептивним полем [16, 88].

$$y(t) = \sum_{k=0}^{K-1} w(k) x(t - d \cdot k), \quad (1.13)$$

де d – коефіцієнт дилатації,

K – розмір ядра згортки.

На відміну від RNN, згорткові операції в TCN можуть виконуватися паралельно для всієї послідовності, що призводить до значно вищої обчислювальної ефективності, а саме швидкого навчання та прогнозування. Дослідження показують, що TCN часто перевершують LSTM як за точністю, так і за швидкістю навчання в задачах прогнозування часових рядів, включаючи енергоспоживання.

Підведемо підсумок розглянутих методів прогнозування енергоспоживання у вигляді блок-схеми за класифікацією на Рисунку 1.2.

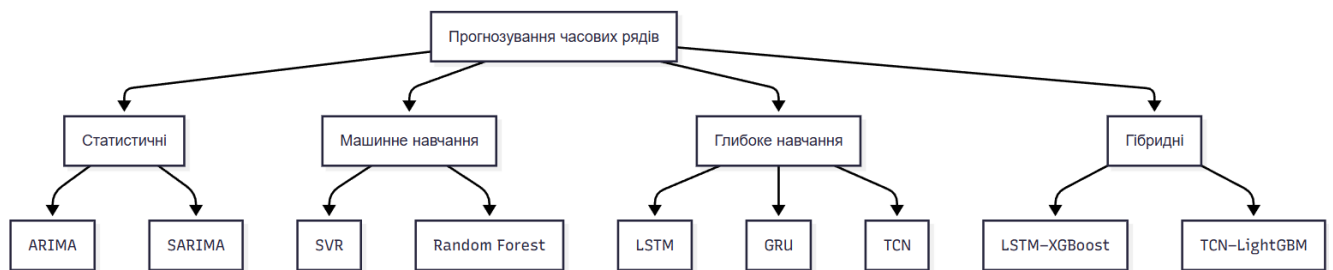


Рисунок 1.2: Класифікація методів прогнозування енергоспоживання.

Незважаючи на високу ефективність, моделі глибокого навчання мають свої обмеження. Вони часто вимагають великих обсягів даних для навчання, можуть бути чутливими до вибору гіперпараметрів, а їхня робота може бути складною для інтерпретації. Крім того, навіть найсучасніші архітектури можуть допускати систематичні помилки, особливо при прогнозуванні рідкісних, але важливих подій, таких як екстремальні піки навантаження.

Сучасним та перспективним напрямком у прогнозуванні часових рядів є розробка гібридних моделей, які прагнуть поєднати сильні сторони різних алгоритмів для досягнення вищої точності, робастності та ефективності. Особливу увагу привертають архітектури, що комбінують моделі глибокого навчання

(LSTM, GRU, TCN) з ансамблями дерев рішень на основі градієнтного бустингу (LightGBM, XGBoost).

LightGBM (Light Gradient Boosting Machine) та XGBoost (Extreme Gradient Boosting) – це високоефективні реалізації алгоритму градієнтного бустингу. Вони чудово справляються з табличними даними, здатні вловлювати складні нелінійні взаємодії між великою кількістю ознак та є відносно стійкими до шумів та викидів. LightGBM, зокрема, вирізняється високою швидкістю навчання та меншим споживанням пам'яті завдяки інноваційним технікам, таким як GOSS та EFB.

Ідея гібридизації полягає у синергетичному використанні переваг обох підходів. У типовій гібридній архітектурі модель глибокого навчання використовується для автоматичного вилучення складних часових ознак та патернів безпосередньо з сирого часового ряду енергоспоживання, ефективно моделює послідовні залежності. Модель градієнтного бустингу використовується на другому етапі для фінального прогнозування. На її вхід подаються як ознаки, вилучені моделлю глибокого навчання, так і додаткові структуровані (табличні) ознаки, такі як погодні дані, календарні змінні (день тижня, година, свята), характеристики споживачів тощо. LightGBM/XGBoost ефективно обробляє цей гетерогенний набір ознак та моделює їхню складну взаємодію.

Часто використовується стратегія "прогнозування залишків" (Residual Fitting), де модель глибокого навчання робить базовий прогноз, а модель бустингу навчається прогнозувати помилки цього базового прогнозу. Фінальний прогноз отримується як сума базового прогнозу та прогнозу залишків.

Численні дослідження переконливо демонструють, що такі гібридні структури, як LSTM-LightGBM, LSTM-XGBoost, TCN-LightGBM стабільно перевершують за точністю як класичні статистичні моделі, так і одиночні моделі машинного чи глибокого навчання. Гібридизація дозволяє ефективно поєднати здатність глибокого навчання до моделювання послідовностей із потужністю градієнтного бустингу в роботі зі структурованими ознаками та корекції помилок.

Для подальшого підвищення ефективності гібридних моделей прогнозування часових рядів у задачах енергоспоживання активно застосовуються додаткові вдосконалені техніки.

Одним із ключових підходів є методи розкладання сигналу (Decomposition Techniques), такі як Empirical Mode Decomposition (EMD), Variational Mode Decomposition (VMD) та Complete Ensemble Empirical Mode Decomposition with Adaptive Noise (CEEMDAN). Вказані методи дають змогу розкласти вихідний часовий ряд на кілька простіших компонентів, кожна з яких відображає певні частотні або трендові характеристики сигналу. Такий попередній аналіз дозволяє моделям більш ефективно вловлювати локальні закономірності, зменшувати волатильність та покращувати точність прогнозу.

Ще одним потужним інструментом є механізми уваги (Attention Mechanisms), що широко використовуються в сучасних нейронних архітектурах. Вони забезпечують здатність моделі динамічно визначати та підсилювати вплив найбільш релевантних частин вхідної послідовності або набору ознак під час формування прогнозу, що особливо важливо для даних із нерівномірними або нелінійними залежностями.

Крім того, для оптимізації структури моделей та налаштування їхніх параметрів дедалі частіше використовуються алгоритми автоматичного пошуку оптимальних гіперпараметрів, зокрема Bayesian Optimization або метаввристичні методи, такі як Grey Wolf Optimizer та Sand Cat Swarm Algorithm. Ці підходи дають змогу зменшити трудомісткість процесу налаштування моделі та підвищити її загальну продуктивність.

Таким чином, гібридні архітектури, що поєднують TCN або LSTM з LightGBM/XGBoost та підсилені сучасними техніками обробки даних та оптимізації, представляють найбільш перспективний напрямок для вирішення складних завдань прогнозування енергоспоживання в системах розумного будинку, пропонуючи найкращий компроміс між точністю, робастністю та обчислювальною ефективністю.

1.3. Аналіз методів очищення сенсорних даних у режимі реального часу

Забезпечення високої якості та надійності даних, що надходять від сенсорів, є абсолютно критичною передумовою для ефективного функціонування будь-якої інтелектуальної системи керування розумним будинком. Сучасні системи Інтернету речей (IoT) генерують значні обсяги даних, особливо від сенсорів, що моніторять фізичні параметри, такі як температура. Однак ці необроблені потоки даних, що збираються в режимі реального часу, часто є далекими від ідеальних і містять значну кількість різноманітних пошкоджень, шумів та аномалій.

Артефакти можуть бути спричинені широким спектром факторів: несправностями або деградацією самих сенсорів, електромагнітними перешкодами під час передачі даних, непередбачуваним впливом зовнішнього середовища, збоями програмного забезпечення або іншими стохастичними подіями. Присутність таких пошкоджень у даних може призвести до катастрофічних наслідків: невірних інтерпретацій стану контролюваного середовища, неефективного управління наявними ресурсами, помилкових рішень систем автоматизації та, як наслідок, до суттєвого зниження загальної ефективності, надійності та безпеки функціонування систем розумного будинку.

Саме тому розробка ефективних методів очищення даних, здатних працювати в режимі реального часу, є фундаментальною та надзвичайно актуальною науково-практичною задачею.

Проблематиці очищення даних часових рядів, зокрема в контексті IoT та сенсорних мереж, присвячено значну кількість наукових праць, які можна умовно поділити на традиційні та сучасні підходи.

Історично першими та найпростішими підходами до боротьби з шумами та аномаліями в даних були методи, засновані на статистичній фільтрації, наразі отримали загальне найменування як класичні методи. До них належать такі широко відомі техніки, як ковзне середнє (Moving Average), експоненційне згладжування (Exponential Smoothing) та медіанна фільтрація (Median Filtering). Ці методи обчислюють поточне очищене значення на основі певного статистичного

агрегату (середнього, медіани, зваженого середнього) значень у локальному ковзному вікні. Їхньою головною перевагою є простота реалізації та низька обчислювальна вартість, що робить їх привабливими для базових застосувань або попередньої обробки.

Більш складним модельним підходом є фільтр Калмана та його численні модифікації. Фільтр Калмана є рекурсивним алгоритмом, який оцінює стан динамічної системи на основі послідовності зашумлених вимірювань. Він використовує модель процесу, наприклад, припущення, що температура змінюється плавно та модель вимірювань, що враховує характеристики шуму, для отримання оптимальної оцінки стану. Фільтр Калмана демонструє хороші результати за умов, коли статистичні характеристики сигналу та шуму відповідають прийнятим модельним припущенням, найчастіше — припущенню про гаусівський характер шуму вимірювань та процесу.

Однак, головним недоліком традиційних методів є їхня обмежена ефективність при роботі зі складними, негаусівськими та структурованими аномаліями, які є типовими для реальних сенсорних даних. Прості фільтри, такі як ковзне середнє чи експоненційне згладжування, можуть призводити до небажаного згладжування важливих деталей сигналу, як от різких, але легітимних змін температури або енергоспоживання, та є малоефективними проти таких артефактів, як поступовий дрейф показників (sensor drift) або тривалі періоди константних, "залиплих", значень (constant/stagnant values). Медіанний фільтр краще справляється з імпульсними викидами, але також може спотворювати сигнал.

Фільтр Калмана, хоч і є більш потужним, втрачає свою оптимальність та ефективність, коли припущення про гаусівський шум порушуються, що часто трапляється в реальних умовах експлуатації сенсорів через наявність змішаних типів шумів. Таким чином, традиційні методи часто виявляються недостатньо гнучкими і точними для сучасних вимог до якості даних в IoT-системах.

Сучасні підходи на основі машинного навчання з розвитком обчислювальних потужностей та алгоритмів машинного навчання (МН) з'явилися нові, значно

потужніші підходи до очищення даних. Моделі МН здатні автоматично вивчати складні нелінійні патерни та залежності безпосередньо з даних, що дозволяє їм виявляти та коригувати аномалії значно ефективніше за традиційні методи. Ці підходи можна класифікувати за типом навчання та використовуваними алгоритмами.

У випадку, якщо доступні розмічені дані, де кожен тип аномалії або несправності позначений відповідною міткою, можна застосувати методи навчання з учителем для побудови класифікаторів аномалій. Ці моделі навчаються розрізняти "нормальні" дані від різних типів "аномальних" даних на основі вилучених ознак, наприклад, статистичних показників з ковзного вікна. Розглянемо алгоритми, що часто використовуються для таких задач.

Опорно-векторні машини (Support Vector Machines, SVM) є ефективними для задач бінарної та багатокласової класифікації, добре працюють на даних високої розмірності. Дослідження показують їхню ефективність у виявленні дрейфу сенсорів навіть на пристроях з обмеженими ресурсами. Основний принцип роботи для класифікації полягає в тому, щоб знайти оптимальну гіперплощину, яка найкращим чином розділяє точки даних, що належать до різних класів. "Оптимальність" досягається шляхом максимізації відстані між цією гіперплощиною та найближчими до неї точками кожного класу, які називаються опорними векторами.

Дерева рішень (Decision Trees) та ансамблі на їх основі (Random Forest, XGBoost, LightGBM) здатні моделювати складні нелінійні взаємозв'язки між ознаками. В літературі вони часто демонструють високу точність та інтерпретованість. Такі алгоритми використовують ієрархічну структуру "if-then-else", щоб розділяти дані на основі значень ознак, доки не буде прийнято остаточне рішення в "листку".

Нейронні мережі (Neural Networks), багатошарові перцептрони та інші архітектури, також можуть використовуватися для класифікації. Для класифікації багатошаровий перцептрон (MLP) пропускає вхідні дані через приховані шари, які нелінійно трансформують дані для вивчення ознак, а вихідний шар обчислює

ймовірності належності вхідного зразка до кожного з можливих класів. Спеціалізовані архітектури, як-от CNN або LSTM, оптимізують цей процес вивчення ознак для специфічних типів даних.

Перевагою керованого підходу є висока точність класифікації, якщо доступна якісна розмічена вибірка. Однак головним недоліком є саме вимога до наявності розмічених даних, які в реальних умовах експлуатації сенсорів часто є дефіцитними, дорогими або взагалі відсутніми. Розмітка великих обсягів даних вручну є трудомістким процесом.

У ситуаціях, коли розмічені дані відсутні, застосовуються методи навчання без учителя, які спрямовані на виявлення аномалій, тобто знаходження точок або патернів, що суттєво відрізняються від "нормальної" поведінки даних. Ці методи не потребують попередньої інформації про типи аномалій. Поширені підходи включають:

Алгоритми, такі як K-Means, виконують класифікацію – групують схожі точки даних разом. Точки, що не належать до жодного щільного кластера або знаходяться далеко від центрів кластерів, можуть вважатися аномаліями.

Методи на основі густини (Density-based methods), як-от Local Outlier Factor (LOF), ідентифікують аномалії як точки, що знаходяться в областях з низькою густиною даних порівняно з їхніми сусідами.

Ансамблевий метод Ізоляційний ліс (Isolation Forest), який "ізолює" аномалії, будуючи випадкові дерева рішень. Аномальні точки зазвичай потребують меншої кількості розбиттів для ізоляції.

Автокодувальники (Autoencoders) – нейронні мережі, навчені відтворювати вхідні дані на виході. Нормальні дані добре реконструюються, тоді як аномалії призводять до великої помилки реконструкції, що дозволяє їх виявити. Особливо ефективними для часових рядів є LSTM-автокодувальники, що враховують часові залежності.

Однокласові класифікатори (One-Class Classifiers) навчаються моделі "нормальних" даних і класифікують будь-які точки, що виходять за межі цієї моделі, як аномалії.

Перевагою некерованих методів є їхня незалежність від розмічених даних, що робить їх широко застосовними. Однак вони можуть мати нижчу точність порівняно з керованими методами, особливо у виявленні тонких або нових типів аномалій, а також можуть генерувати більшу кількість хибних спрацьовувань.

Окремо варто виділити методи глибокого навчання, які демонструють результати у багатьох задачах аналізу часових рядів, включаючи очищення та виявлення аномалій. Окрім згаданих автокодувальників, використовуються:

- рекурентні нейронні мережі можуть бути навчені прогнозувати наступне значення ряду. Значні відхилення між прогнозом та фактичним значенням вказують на аномалію;

- згорткові нейронні мережі ефективні для вилучення локальних патернів та ознак, які потім можуть бути використані для виявлення аномалій. Гібридні архітектури CNN-LSTM або CNN-BILSTM поєднують переваги обох підходів [61, 73, 83, 114];

- темпоральні згорткові мережі як альтернатива RNN, TCN можуть використовуватися в автокодувальниках або прогнозних моделях для виявлення аномалій [31, 54].

Таблиця 1.2

Методи очищення сенсорних даних та їх властивості

Метод	Принцип дії	Переваги	Недоліки
Ковзне середнє	Усереднення в вікно	Простота, швидкість	Згладжує різкі зміни
Експоненційне згладжування	Зважене минуле	Гнучкість	Схильність до відставання
Медіанний фільтр	Медіана у вікні	Добре прибирає викиди	Спотворює різкі переходи
Фільтр Калмана	Статистична модель стану	Оптимальність при гаусівських шумах	Погіршується при негаусівському шумі
Autoencoder	Реконструкція послідовності	Працює без розмітки, висока точність	Велика ресурсомісткість
Isolation Forest	Ізоляція аномалій	Добре працює з різними шумами	Чутливість до параметрів

Перевагами глибокого навчання є здатність автоматично вивчати складні ознаки та часові залежності без необхідності ручної інженерії ознак. Однак вони вимагають великих обсягів даних для навчання, є обчислювально витратними та часто працюють як "чорна скринька", що ускладнює інтерпретацію їхніх рішень.

Розуміючи обмеження окремих методів, дослідники все частіше звертаються до гібридних підходів, які прагнуть поєднати сильні сторони різних технік:

1. Поєднання статистичного попереднього оброблення з МН, наприклад, використання медіанного фільтра для усунення грубих викидів перед подачею даних до LSTM-моделі або використання статистичних показників як ознак для класифікатора МН.

2. Комбінування прогнозів кількох різних моделей (наприклад, LSTM, Isolation Forest, статистичних методів) для підвищення надійності та зменшення хибних спрацьовувань, називають ансамблями моделей.

3. Поєднання моделей машинного навчання з евристичними правилами або знаннями предметної області для покращення точності та інтерпретованості.

Гібридні підходи демонструють великий потенціал для створення більш робастних та точних систем очищення даних.

Незважаючи на значний прогрес, застосування методів очищення даних у режимі реального часу в системах IoT стикається з низкою викликів. Багато IoT-пристроїв, особливо на периферії (edge devices) мають обмежену потужність процесора, пам'ять та енергоспоживання, що ускладнює розгортання складних моделей глибокого навчання. Тому існує потреба в легких та енергоефективних алгоритмах.

Для багатьох застосувань (наприклад, систем безпеки або керування) рішення повинні прийматися майже миттєво, що накладає жорсткі обмеження на час обробки даних.

Дані надходять безперервним потоком, і алгоритми повинні обробляти їх послідовно, часто без можливості повернутися до попередніх даних. Це вимагає використання онлайн або інкрементальних алгоритмів навчання та виявлення аномалій.

Характеристики даних та типи аномалій можуть змінюватися з часом (концептуальний дрейф), що вимагає від методів очищення здатності адаптуватися до нових умов [68].

Крім загальних викликів, як було детально формалізовано в підрозділі 3.1, ключовою проблемою є гетерогенність шумів у сенсорних даних розумного будинку. Різні типи аномалій такі, як: викиди, дрейф, константні значення, тощо, вимагають різних стратегій корекції. Застосування універсального методу, наприклад, простої фільтрації або бінарного виявлення аномалій, є неефективним, оскільки він або не зможе усунути всі типи пошкоджень, або призведе до спотворення корисного сигналу.

Саме ця проблема обґрунтовує необхідність розробки адаптивних методів очищення, які здатні не просто виявляти аномалії, а й класифікувати їхній тип та на основі цієї класифікації динамічно обирати найбільш доцільний оператор корекції. Такий підхід, що поєднує точність машинного навчання в ідентифікації типів шумів з гнучкістю адаптивного вибору стратегій очищення, є найбільш перспективним напрямком для створення по-справжньому надійних та ефективних систем обробки сенсорних даних у розумному будинку.

1.4. Аналіз архітектур та парадигм керування розумним будинком.

Архітектура та парадигма керування є визначальними компонентами будь-якої системи розумного будинку. Вони формують логічну структуру, принципи взаємодії елементів та, що найважливіше, визначають інтелектуальний рівень системи – її здатність реагувати на зміни, навчатися, адаптуватися до потреб мешканців та оптимізувати свою роботу для досягнення поставлених цілей, таких як комфорт, безпека та енергоефективність. Історичний розвиток систем домашньої автоматизації демонструє чітку еволюцію від простих, жорстко запрограмованих реактивних моделей до складних, гнучких, проактивних та людино-орієнтованих архітектур, що використовують методи штучного інтелекту.

Аналіз існуючих підходів дозволяє виділити кілька ключових парадигм та архітектурних рішень [5].

Принцип роботи основних парадигм керування порівняно на Рисунку 1.3.

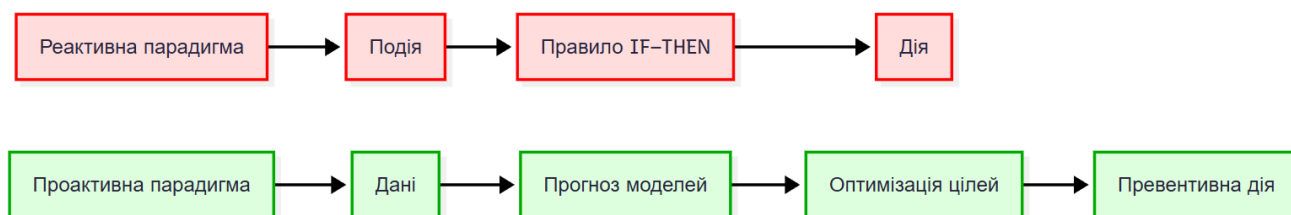


Рисунок 1.3: Порівняння реактивної та проактивної парадигм

Основою для перших поколінь систем домашньої автоматизації слугували системи на основі статичних правил, що функціонують за принципом "Якщо-То" (IF-THEN). Ця парадигма, також відома як Trigger-Action Programming (TAP), передбачає, що система реагує на певну заздалегідь визначену подію (тригер), таку як натискання кнопки, спрацювання сенсора руху або досягнення певного значення температури, виконуючи відповідну, жорстко запрограмовану дію. Наприклад, "ЯКЩО сенсор руху виявив рух у коридорі вночі, ТО увімкнути світло на 30% яскравості" [116].

Перевагами таких систем є їхня простота, передбачуваність та детермінованість. Логіку їхньої роботи легко зрозуміти, налаштувати та відлагодити, а користувач чітко знає, як система відреагує на ту чи іншу подію. Проте ця простота має свою ціну. Фундаментальною вадою реактивних систем на правилах є їхня критична негнучкість та відсутність адаптивності. Вони нездатні:

- адаптуватися до динамічних змін у зовнішньому середовищі;
- враховувати складні, неявні або мінливі вподобання користувачів;
- вирішувати конфлікти між правилами або цілями.

Перечислені недоліки призводять до необхідності створення значної кількості деталізованих правил для опису всіх можливих сценаріїв, що робить систему громіздкою та складною в управлінні. Найголовніше, такі системи вимагають постійного ручного втручання з боку мешканців, коли автоматична

поведінка не відповідає їхнім поточним потребам, що нівелює саму ідею автоматизації та призводить до низької задоволеності користувачів.

Для подолання обмежень статичних систем активно застосовуються методи машинного навчання (МН) та концепція контекстної обізнаності, що дозволяє системам навчатися та адаптуватися до змін [11].

Методи МН такі, як навчання з підкріпленням (Reinforcement Learning, RL), дозволяють створювати системи, які навчаються оптимальним стратегіям керування шляхом безпосередньої взаємодії з середовищем. RL-агент отримує інформацію про поточний стан середовища, виконує дію та отримує "винагороду" або "штраф" залежно від того, наскільки ця дія сприяла досягненню мети. Поступово, методом спроб і помилок, агент навчається обирати дії, що максимізують сукупну винагороду [113].

Перевагою RL-систем є їхня висока адаптивність та здатність автономно знаходити оптимальний баланс між суперечливими цілями без необхідності явного програмування правил. Такі системи демонструють високу ефективність в оптимізаційних задачах.

Однак вони мають і недоліки. RL часто потребує значного періоду "розвідки" та великих обсягів даних для навчання ефективної політики. В цей же час процес прийняття рішень глибокими RL-моделями може бути непрозорим для користувача, що ускладнює довіру до системи, її налаштування та діагностику помилок.

Парадигма контекстно-обізнаних систем (Context-Aware Systems) являє собою фундаментальний зсув у проєктуванні інтелектуальних систем, відходячи від статичних, заздалегідь запрограмованих алгоритмів. Вона наголошує на критичній необхідності наділення системи здатністю не лише отримувати дані, але й активно сприймати, інтерпретувати та враховувати максимально широкий спектр контекстної інформації. Саме ця здатність є передумовою для прийняття виважених, адекватних поточній ситуації та обґрунтованих рішень [41]. Контекст, у цьому розумінні, визначається як будь-яка інформація, що може бути

використана для характеристики ситуації сутності (людини, місця чи об'єкта), яка вважається релевантною для взаємодії між користувачем та системою [8].

Для побудови ефективної моделі система повинна агрегувати та аналізувати дані з багатьох джерел. Класифікація контекстної інформації зазвичай включає наступні домени:

Фізичне середовище – це базовий рівень сприйняття, що включає прямі показники сенсорів: температура, освітленість (у люксах), вологість, якість повітря (рівень CO₂, наявність летких органічних сполук), а також рівень шуму. Ці дані формують об'єктивну картину фізичного стану простору.

Часовий контекст включає не лише абсолютний час (час доби, день тижня), але й більш широкі патерни, такі як пора року, календарні події (робочий день, вихідний, свята). Цей контекст є ключовим для передбачення рутинних дій та зміни патернів поведінки.

Просторовий контекст (або Контекст місцезнаходження) визначає фізичне місцезнаходження користувача чи об'єкта. Це може бути як макро-рівень (напр., в будинку або поза ним), так і мікро-рівень (перебування в конкретній кімнаті – кухні, спальні, вітальні), що дозволяє надавати послуги, специфічні для даної локації.

Контекст користувача один із найскладніших, але найважливіших типів. Він охоплює інформацію про самого мешканця: його поточна діяльність, наприклад, "сон", "приготування їжі", "робота", "відпочинок", стан здоров'я, ідентифікований настрій або рівень стресу. Це дозволяє перейти до по-справжньому персоналізованого ("people-centric") керування.

Соціальний контекст описує соціальні аспекти ситуації, зокрема кількість людей у приміщенні, їхні особистості та, у складних моделях, їхні стосунки або спільну діяльність. Наприклад, система повинна реагувати по-різному, коли в кімнаті одна людина читає, або коли там відбувається вечірка.

Системний (або Операційний) контекст це "мета-контекст", що описує стан самої системи керування. Він включає стан пристроїв (ввімкнено/вимкнено, рівень заряду), доступність мережевих ресурсів, пропускну здатність каналів

зв'язку та, що критично важливо для енергоефективності, поточні тарифи на енергію.

Для моделювання та інтерпретації таких складних, багатовимірних та часто неповних контекстних даних, які надходять з різнорідних джерел, у сучасній науці про дані використовуються різноманітні підходи та обчислювальні моделі.

Онтології та семантичні моделі (Ontologies) базуються на методах інженерії знань. Вони дозволяють формально описувати знання про предметну область (наприклад, "розумний будинок", його приміщення, пристрої, користувачів) та встановити чіткі відношення між поняттями (напр., "Спальня є Приміщення", "Термостат керує Температурою"). Використання онтологій та логічних вивідників (reasoners) дозволяє системі не лише зберігати факти, але й логічно виводити нові знання про контекст, які не були отримані з сенсорів напряму.

Нечітка логіка (Fuzzy Logic) є надзвичайно корисною, оскільки вона дозволяє системі працювати з нечіткими, лінгвістичними поняттями та правилами, які краще відображають людське мислення та сприйняття. Замість бінарних значень ("холодно/тепло"), нечітка логіка оперує ступенями належності до множин (напр., "трохи прохолодно", "дуже тепло"). Дозволяє створювати гнучкі та більш плавні правила керування, що адекватно реагують на неточні вхідні дані [21, 64].

Імовірнісні моделі (Probabilistic Models) є незамінним для роботи в умовах невизначеності. Вони дозволяють системі моделювати невизначеність та робити імовірнісні висновки про поточний контекст на основі неповних або зашумлених сенсорних даних. Застосування таких моделей критичне для процесу "сенсорного злиття" (sensor fusion), де дані з кількох джерел об'єднуються для отримання більш надійної оцінки. До найбільш поширених у цій галузі відносяться Байєсівські мережі (Bayesian Networks) для моделювання причинно-наслідкових зв'язків, приховані Марковські моделі (HMM) для розпізнавання послідовностей дій (наприклад, активності користувача) та Марковські логічні мережі (MLN), що поєднують імовірнісний підхід з логічним.

Багатоагентні системи (Multi-Agent Systems, MAS) дозволяє моделювати систему як сукупність взаємодіючих автономних агентів [18]. Кожен агент може

відповідати за певний вузький аспект контексту (напр., "агент освітлення", "агент температури") або керування. Вони взаємодіють між собою шляхом переговорів або обміну повідомленнями для досягнення спільних (або приватних) цілей. Такий підхід забезпечує високу модульність, гнучкість та відмовостійкість системи [86].

Підсумовуючи, контекстна обізнаність перестає бути лише додатковою функцією і перетворюється на необхідну умову для створення по-справжньому інтелектуальних, проактивних та персоналізованих систем. Здатність адекватно моделювати та інтерпретувати багатовимірний контекст дозволяє системі вийти за межі жорстких реактивних сценаріїв та приймати оптимальні рішення, що максимально відповідають реальній поточній ситуації та динамічним потребам користувача.

Найсучаснішим напрямком розвитку є проактивне та людино-орієнтоване керування, яке поєднує адаптивність, контекстну обізнаність та здатність діяти на випередження на основі передбачень та високорівневих цілей користувача.

На відміну від реактивних систем, що лише відповідають на події, проактивні системи намагаються передбачити майбутні стани та потреби і діяти на випередження, щоб запобігти небажаним ситуаціям або оптимізувати роботу. Це вимагає наявності прогностичних моделей, які можуть передбачати, наприклад, зміну температури, майбутнє енергоспоживання або поведінку користувача. На основі цих прогнозів система може приймати превентивні рішення (наприклад, заздалегідь увімкнути опалення, щоб досягти комфортної температури до приходу користувача).

Концепція проактивної системи полягає у тому, щоб дозволити користувачеві визначати абстрактні цілі, а не конкретні інструкції. Система самостійно інтерпретує наміри, такі як "баланс комфорту та економії", і знаходить оптимальну послідовність дій для їх реалізації в поточному контексті. Це кардинально знижує когнітивне навантаження на користувача та підвищує гнучкість системи.

Людино-орієнтованість ставить у центр уваги потреби, вподобання та комфорт людини. Вона передбачає моделювання та навчання вподобань користувача, вирішення конфліктів, прозорість та пояснюваність. Система повинна навчатися індивідуальним уподобанням, аналізуючи природну поведінку та ручні втручання. Важливою є здатність навчатися контекстуальним намірам, оскільки вподобання змінюються залежно від ситуації [95].

У багатокористувацьких середовищах система повинна мати механізми для виявлення та вирішення конфліктів між вподобаннями різних мешканців або між цілями системи. Для цього використовуються переговори між агентами, онтологічні фреймворки, пріоритезація та інші стратегії.

Для підвищення довіри користувача система повинна бути здатною пояснювати свої рішення. Пояснюваний ІІІ (Explainable AI, XAI) стає все більш важливим компонентом людино-орієнтованих систем.

Для реалізації описаних парадигм використовуються різні архітектурні підходи. Як зазначалося, багатоагентні системи є популярним вибором для моделювання складних, розподілених систем, де кожен агент (пристрій, користувач, сервіс) має свої цілі та може взаємодіяти з іншими для досягнення спільного результату. MAS добре підходять для реалізації контекстної обізнаності, вирішення конфліктів та масштабування.

Сервіс-орієнтовані архітектури (Service-Oriented Architecture, SOA) та мікросервіси дозволяють будувати гнучкі системи з незалежних, слабо зв'язаних компонентів (сервісів), що полегшує інтеграцію нових пристроїв та функцій.

Рівневі (Layered) архітектури часто використовуються для структурування складних систем, розділяючи їх на рівні, наприклад, рівень сприйняття (сенсори), рівень обробки/прийняття рішень та рівень виконання (актуатори).

Гібридні архітектури поєднують елементи різних підходів, наприклад, MAS з рівневою структурою або інтеграцію хмарних, туманних (fog) та периферійних (edge) обчислень для оптимізації обробки даних та часу реакції.

Незважаючи на значний прогрес, розробка ефективних архітектур та парадигм керування розумним будинком стикається з низкою викликів: точне

моделювання людської поведінки, вподобань та складних контекстів залишається складним завданням, робота з зашумленими, неповними та суперечливими даними від сенсорів та користувачів, забезпечення ефективної роботи в умовах великої кількості пристроїв та користувачів, особливо на ресурсообмежених платформах, розробка швидких та справедливих механізмів вирішення конфліктів.

Сучасні дослідження спрямовані на вирішення цих викликів шляхом розробки більш досконалих гібридних моделей, адаптивних алгоритмів навчання, ефективних методів обробки невизначеності та пояснюваних систем штучного інтелекту, що наближає нас до створення по-справжньому інтелектуальних, автономних та комфортних житлових просторів.

1.5. Постановка завдання дисертаційного дослідження

Проведений у розділі 1 аналіз сучасного стану проблеми оптимізації керування розумним будинком дозволяє виявити низку невирішених науково-практичних завдань, що стримують розвиток по-справжньому автономних, ефективних та людино-орієнтованих систем. Незважаючи на значний прогрес, існуючі підходи не пропонують комплексного рішення для мінімізації сукупних витрат, які включають не лише вартість енергії та рівень дискомфорту, але й "вартість зусиль користувача", що виражається у частоті ручних втручань.

Проблема полягає в тому, що ефективність системи керування залежить від трьох взаємопов'язаних рівнів, на кожному з яких існують свої прогалини:

1. Необроблені дані від IoT-сенсорів часто містять складні комбінації гетерогенних шумів (викиди, дрейф, константні значення).

2. Існуючі методи очищення даних часто демонструють обмежену ефективність при роботі з такими змішаними типами шумів у реальному часі та не враховують специфіку даних від сенсорів з різними фізичними характеристиками.

3. Точне короткострокове прогнозування енергоспоживання є ключовим для проактивного керування. Хоча гібридні моделі на основі LSTM та градієнтного

бустингу довели свою ефективність, залишається недостатньо дослідженим їхнє систематичне порівняння з новітніми архітектурами, такими як TCN.

4. Важливо порівняти ефективність методів прогнозування в контексті компромісу між точністю прогнозу та обчислювальною вартістю, що є критичним для імплементації на пристроях з обмеженими ресурсами.

5. Більшість сучасних підходів або покладаються на статичні правила, що є негнучкими, або вимагають складних налаштувань та великих обсягів даних для навчання, не враховуючи динамічні та контекстуально-залежні вподобання користувачів.

6. Існує потреба в архітектурі, яка б могла автономно навчатися неявним намірам користувача, аналізуючи його природну поведінку, та проактивно приймати оптимальні рішення в динамічному середовищі.

Таким чином, завдання дисертаційного дослідження полягає у розробці та дослідженні комплексного набору методів для оптимізації керування розумним будинком, які б усували виявлені прогалини шляхом:

1. Створення адаптивного методу очищення даних, здатного працювати з гетерогенними шумами в реальному часі.

2. Вдосконалення гібридного методу прогнозування, що забезпечує оптимальний баланс точності та обчислювальної ефективності.

3. Розробки проактивної архітектури керування, яка навчається контекстуальним намірам користувача для мінімізації його втручань.

Виходячи з поставленого завдання, розроблювані в рамках дисертаційної роботи методи та моделі повинні відповідати таким ключовим вимогам:

1. Методологія повинна бути здатною ідентифікувати та розрізняти декілька типів аномалій та застосовувати до них відповідні стратегії корекції, на відміну від універсальних фільтрів.

2. Алгоритм має бути обчислювально ефективним для обробки даних у потоковому режимі з мінімальною затримкою, що є критичним для оперативного реагування системи керування.

3. Метод повинен демонструвати ефективність на різномірних типах сенсорних даних (наприклад, температура, вологість, енергоспоживання), що мають різні фізичні властивості та характеристики шумів.

Вимоги до методу прогнозування:

1. Метод повинен забезпечувати високу точність прогнозування, особливо для критичних пікових навантажень, що є ключовим для ефективного управління енергоресурсами.

2. Модель має бути достатньо легкою для розгортання на пристроях з обмеженими ресурсами, характерними для екосистеми Інтернету речей.

3. Методологія повинна надавати інструменти для аналізу та оптимізації балансу між точністю прогнозування та обчислювальними витратами (часом навчання), дозволяючи вибирати конфігурацію залежно від пріоритетів завдання.

Вимоги до архітектури керування:

1. Архітектура повинна діяти на випередження, використовуючи прогнози майбутніх станів для прийняття оптимальних рішень, а не просто реагувати на поточні події.

2. Система повинна навчатися вподобанням користувача неявно, аналізуючи його звичайну поведінку (наприклад, ручні втручання), без необхідності у складному програмуванні чи налаштуванні.

3. Методологія повинна розпізнавати, що наміри користувача можуть змінюватися залежно від контексту (час доби, тариф на енергію, пора року), і адаптувати свою поведінку відповідно.

4. Кінцевою метою є мінімізація потреби в ручних втручаннях користувача, що є ключовим показником успішної автоматизації.

Таким чином, для усунення виявлених прогалин та реалізації поставленого завдання дисертаційного дослідження пропонується комплексний набір методів, що охоплює послідовну обробку даних від рівня сенсора до проактивного керування. Логіка інтеграції цих методів, а також структура взаємодії між основними етапами візуально представлена на Рисунку 1.4.

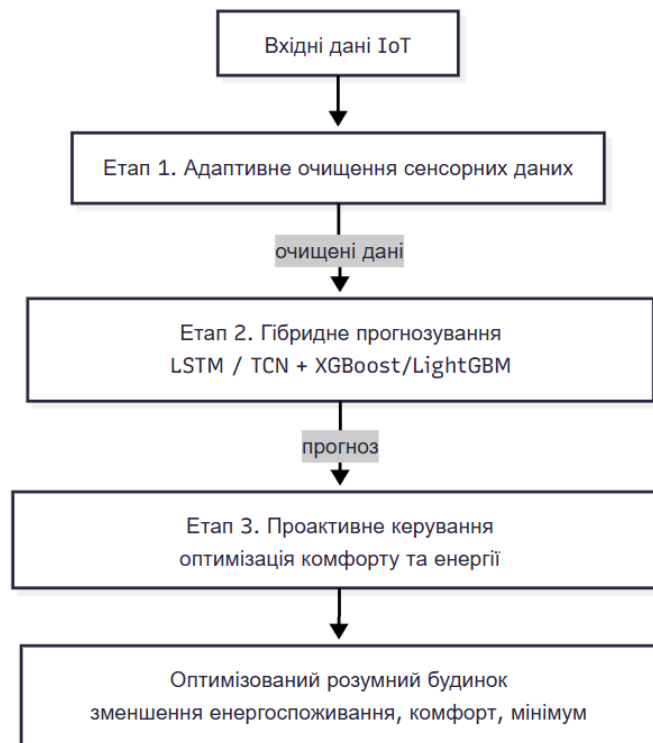


Рисунок 1.4: Структурно-логічна схема дослідження.

Висновки до розділу 1

У розділі проведено комплексний аналіз сучасного стану проблеми оптимізації керування системами розумного будинку на основі Інтернету речей. Встановлено, що концепція IoT створює технологічну основу для побудови інтелектуальних житлових просторів, однак її повний потенціал обмежується низкою невирішених проблем.

Проаналізовано ключові складові ефективної системи керування:

1. Огляд показав еволюцію методів від класичних статистичних підходів до сучасних гібридних моделей глибокого навчання. Виявлено перспективність архітектур TCN та LightGBM, а також прогалину в дослідженнях, що стосується оптимізації їхньої обчислювальної ефективності.

2. Визначено, що якість даних є фундаментальною вимогою. Аналіз існуючих методів виявив їхню недостатню адаптивність до гетерогенних типів шумів, що характерні для IoT-середовищ, та обґрунтував потребу в розробці інтелектуального методу, що базується на класифікації аномалій.

3. Проаналізовано перехід від статичних реактивних систем на основі правил до адаптивних та проактивних моделей. Виявлено брак комплексних архітектур, здатних автономно навчатися динамічним, контекстуально-залежним намірам користувача на основі його неявної поведінки.

Таким чином, проведений аналіз підтверджує актуальність теми дисертаційної роботи та дозволяє сформулювати чітке завдання дослідження, спрямоване на розробку цілісного комплексу методик, що усувають виявлені прогалини. Наступні розділи дисертації присвячені безпосередній розробці та експериментальному дослідженню запропонованих рішень.

РОЗДІЛ 2. ВДОСКОНАЛЕННЯ ГІБРИДНОГО МЕТОДУ ПРОГНОЗУВАННЯ ЕНЕРГОСПОЖИВАННЯ

2.1. Теоретичні основи темпоральних згорткових мереж (TCN).

Темпоральні згорткові мережі (Temporal Convolutional Networks, TCN) представляють собою відносно новий, але надзвичайно потужний клас архітектур глибокого навчання, спеціально розроблених для ефективної обробки послідовних даних, таких як часові ряди. Вони виникли як альтернатива домінуючим раніше рекурентним нейронним мережам (RNN), зокрема мережам з довгою короткостроковою пам'яттю (LSTM) та вентиляним рекурентним блокам (GRU), пропонуючи низку суттєвих переваг у контексті точності, стабільності навчання та обчислювальної ефективності. Завдяки своїй унікальній згортковій архітектурі, TCN здатні ефективно моделювати як короткострокові, так і дуже довгострокові залежності в даних, що робить їх особливо перспективними для вирішення складних задач прогнозування, таких як короткострокове прогнозування енергоспоживання (STLF) в системах розумного будинку.

Архітектурна філософія TCN базується на поєднанні трьох ключових концепцій: причинно-наслідкових згорток, розширених згорток та залишкових блоків. Розглянемо кожну з них детальніше.

Фундаментальним принципом, що відрізняє TCN від стандартних згорткових мереж (CNN) є використання причинно-наслідкових згорток. Стандартна 1D-згортка при обробці послідовності зазвичай використовує інформацію як з минулих, так і з майбутніх часових кроків відносно поточної точки для обчислення вихідного значення. Однак, у задачах прогнозування часових рядів доступ до майбутніх значень є неприпустимим, оскільки це призведе до "витоку інформації з майбутнього" і, як наслідок, до нереалістично завищених показників точності під час тестування та абсолютної непридатності моделі для реального прогнозування.

Причинно-наслідкові згортки вирішують цю проблему, будучи односпрямованими. Згортковий фільтр застосовується таким чином, що вихідне значення y_t в момент часу t залежить виключно від вхідних значень $x_t, x_{t-1}, x_{t-2}, \dots$ у поточний та попередні моменти часу. Іншими словами, обчислення в момент t не мають доступу до значень x_{t-1}, x_{t-2}, \dots . Це гарантує дотримання принципу причинності, що є абсолютно необхідним для будь-якої моделі, яка претендує на роль прогнозної. Принцип є ключовим для валідності прогнозування. Пояснення причинності відрізняє TCN від звичайних CNN і підкреслює їхню придатність саме для часових рядів.

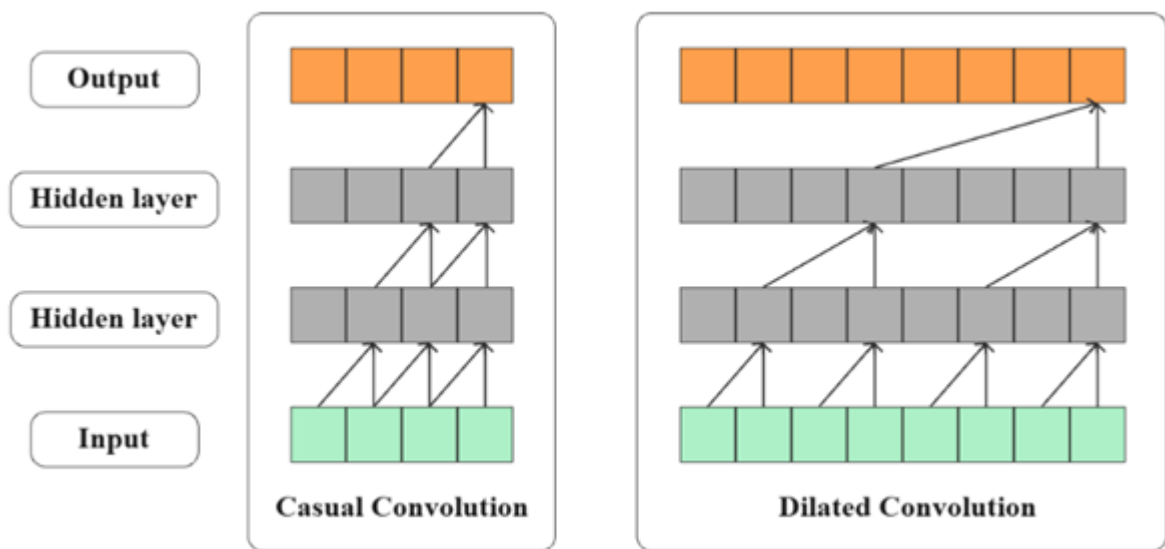


Рисунок 2.1: Схематичне зображення причинно-наслідкової згортки

Просте використання причинно-наслідкових згорток має суттєвий недолік: для того, щоб модель могла враховувати залежності на великих часових відстанях, необхідно або використовувати дуже великі розміри фільтрів, або будувати надзвичайно глибокі мережі. Обидва варіанти є обчислювально неефективними.

Архітектури часових згорткових мереж (TCN) пропонують ефективно вирішення цієї проблеми обмеженого рецептивного поля шляхом імплементції розширених згорток (dilated convolutions), також відомих як згортки "atrous". Суть розширеної згортки полягає в тому, що ядро (фільтр) згортки застосовується до вхідних даних з певними пропусками, або "дірками". Ці пропуски контролюються

фактором розширення (dilation factor, d). При $d=1$ розширена згортка є еквівалентною стандартній згортці, оскільки фільтр застосовується до кожного вхідного елемента поспіль. При $d=2$ фільтр застосовується до кожного другого вхідного елемента, ефективно "перестрибуючи" через один; при $d=4$ — до кожного четвертого, і так далі. Такий підхід дозволяє ядру згортки охоплювати ширшу область вхідних даних без збільшення кількості параметрів або обчислювальної складності.

Ключова архітектурна ідея TCN полягає не просто у використанні розширених згорток, а в їх стратегічному пошаровому компонуванні. Фактор розширення d експоненційно збільшується з кожним наступним шаром мережі. Зазвичай ця прогресія слідує степеню двійки: $d = 2^l$, де l — номер шару мережі, що починається з 0 (тобто, $d = 1, 2, 4, 8, 16, \dots$). Це призводить до надзвичайно швидкого, експоненційного зростання сукупного рецептивного поля — тобто області вхідної послідовності, яка має вплив на одне єдине вихідне значення на останньому шарі.

Саме це експоненційне зростання є критично важливим для розуміння ефективності TCN. Наприклад, розглянемо мережу з розміром фільтра $k=3$ та чотирма шарами, де фактори розширення послідовно дорівнюють $d=1, 2, 4, 8$. Рецептивне поле RF такої мережі можна розрахувати за формулою:

$$RF = 1 + (k - 1) \sum_{l=0}^{L-1} \{l = 0\} \cdot 2^{l \cdot d}, \quad (2.1)$$

де L - кількість шарів.

Для нашого випадку:

$$RF = 1 + (3 - 1) \cdot (1 + 2 + 4 + 8) = 1 + 2 \cdot (15) = 31$$

Таким чином, вихідне значення після всього чотирьох шарів з дуже малими фільтрами (лише 3 параметри на шар) вже "бачить" 31 вхідну точку в минулому. Якби використовувалися стандартні згортки ($d=1$), рецептивне поле після 4 шарів склало б лише $1 + 2 \cdot (1 + 1 + 1 + 1) = 9$ точок.

Такий архітектурний підхід дозволяє TCN ефективно "заглядати" в дуже довгі історичні послідовності та моделювати довгострокові залежності (long-term

dependencies), що є фундаментальною проблемою для багатьох рекурентних мереж. При цьому TCN використовують відносно неглибоку архітектуру та зберігають високу обчислювальну ефективність, оскільки згорткові операції, на відміну від рекурентних, можуть виконуватися паралельно. Важливо підкреслити, що розширені згортки в TCN застосовуються в рамках суворої причинно-наслідкової структури (causal structure). Це досягається шляхом спеціального "причинного" доповнення (causal padding), яке гарантує, що вихідне значення в момент часу t генерується лише на основі вхідних даних з моментів $t, t-1, t-2, \dots$ і ніколи не "заглядає в майбутнє", тобто, не використовує дані з $t+1$). Ця комбінація експоненційного рецептивного поля та суворої причинності і є однією з головних інновацій TCN.

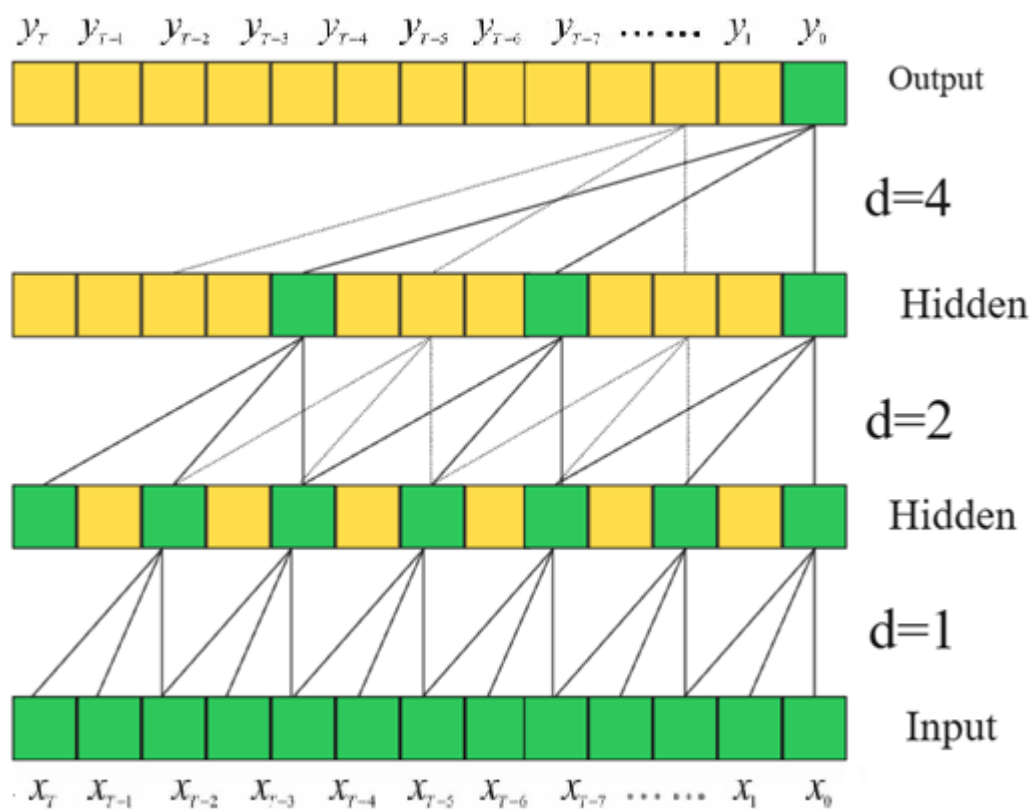


Рисунок 2.2: Ілюстрація розширених причинно-наслідкових згорток з факторами розширення $d=1$, $d=2$ та $d=4$

При побудові та тренуванні глибоких нейронних мереж, особливо тих, що складаються з десятків послідовних шарів, виникає фундаментальна проблема

нестабільності градієнтів. Вона проявляється у двох формах: "зникаючого" або, рідше, "вибухаючого" (exploding) градієнта. У процесі зворотного поширення помилки (backpropagation) градієнт багаторазово перемножується, і якщо ваги або похідні функції активації стабільно менші за одиницю, сигнал про помилку експоненційно згасає, практично не доходячи до ранніх шарів мережі. Це унеможливорює або значно ускладнює процес навчання, не дозволяючи моделі налаштувати свої початкові ваги.

Для ефективної боротьби з цією проблемою та забезпечення стабільного, збіжного навчання глибоких архітектур, TCN інтегрують у свою структуру залишкові з'єднання (residual connections). Цей підхід був вперше запропонований та довів свою високу ефективність в архітектурах ResNet (Residual Networks) у галузі обробки зображень, дозволивши успішно тренувати мережі глибиною в сотні шарів. TCN адаптували цю концепцію для роботи з часовими послідовностями.

Базовим будівельним елементом архітектури TCN є залишковий блок, структура спеціально розроблена для реалізації залишкового навчання. Типовий залишковий блок отримує на вхід тензор x та обробляє його у двох паралельних гілках.

Перша гілка – це трансформаційний шлях $F(x)$, який складається з послідовного стеку операцій:

1. Перший шар розширеної причинно-наслідкової згортки (з певним фактором розширення d).
2. Нормалізація ваг (Weight Normalization), яка застосовується замість більш традиційної пакетної нормалізації для стабілізації процесу навчання.
3. Нелінійна функція активації ReLU (Rectified Linear Unit).
4. Шар Dropout для регуляризації та запобігання перенавчанню (overfitting).
5. Другий шар розширеної причинно-наслідкової згортки з тим самим фактором розширення d .
6. Повторення нормалізації ваг, активації ReLU та шару Dropout.

Друга гілка – це ключовий елемент: залишкове з'єднання (або "skip connection"), яке просто пропускає вхідний тензор x в обхід трансформаційного шляху.

На виході з блоку результат трансформації $F(x)$ поелементно додається до входу x . Фінальний вихід блоку обчислюється як:

$$\text{Activation}(x + F(x)), \quad (2.2)$$

де Activation – це фінальна функція активації.

У випадку, якщо вхід x та вихід трансформації $F(x)$ мають різну розмірність, наприклад, через зміну кількості фільтрів/каналів у згорткових шарах, до входу x застосовується додаткова "проекційна" згортка розміром 1×1 . Ця операція слугує виключно для узгодження розмірностей та дозволяє коректно виконати операцію додавання.

З точки зору процесу навчання, залишкові з'єднання мають вирішальне значення. Вони створюють "короткі шляхи" або "магістралі" для градієнтів під час зворотного поширення помилки. Завдяки операції додавання $(x + F(x))$, градієнт на виході блоку може текти назад до входу двома шляхами: через складний трансформаційний блок $F(x)$ і безпосередньо через "ідентичне" з'єднання x (де похідна дорівнює одиниці). Цей другий шлях забезпечує безперешкодний потік градієнтного сигналу через всю глибину мережі, ефективно вирішуючи проблему його "зникання".

Таким чином, це значно полегшує та стабілізує навчання дуже глибоких архітектур TCN. Пояснення механізму роботи залишкових блоків є критично важливим для розуміння того, як TCN вдається ефективно масштабуватися у глибину. Саме ця здатність до навчання глибоких архітектур дозволяє накопичувати велику кількість шарів з експоненційно зростаючими факторами розширення. Це, в свою чергу, є необхідною умовою для досягнення дуже великого рецептивного поля, здатного охоплювати найдовші історичні залежності у вхідних часових рядах.

Повна модель TCN зазвичай складається зі стека таких залишкових блоків, де фактори розширення d експоненційно зростають у кожному наступному блоці

(або шарі всередині блоку). Наприклад, стек може мати фактори розширення $d = 1, 2, 4, 8, 16, \dots$. Кількість блоків, розмір фільтрів k , кількість фільтрів (каналів) у кожному шарі та фактори розширення є гіперпараметрами моделі, які підбираються залежно від конкретної задачі та характеристик даних. На виході останнього блоку зазвичай застосовується ще один згортковий або повнозв'язний шар для отримання прогнозу потрібної розмірності (наприклад, прогнозу на один або кілька кроків у майбутнє).

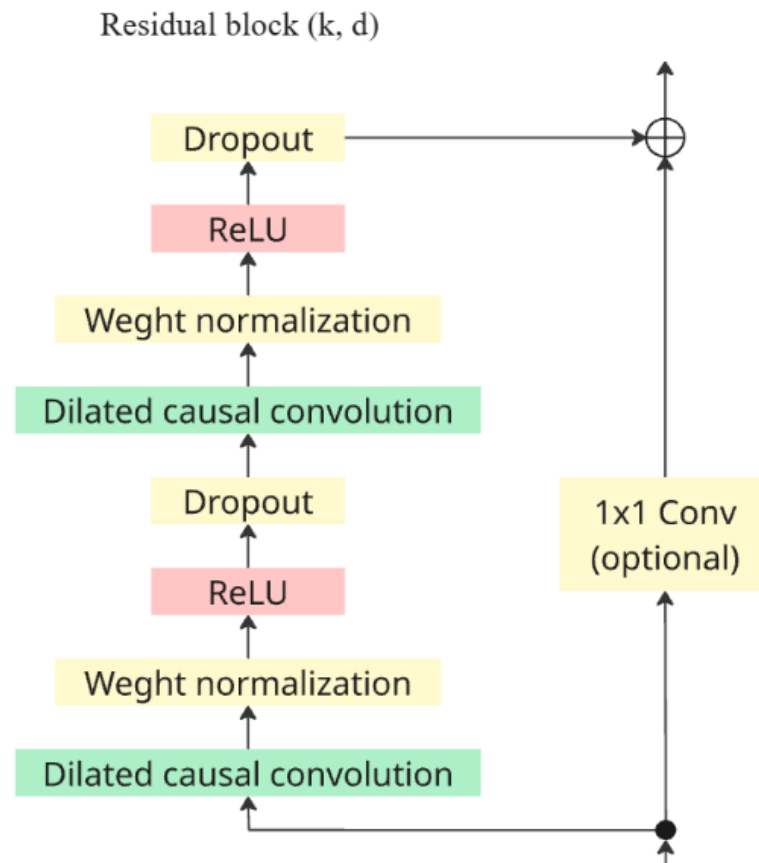


Рисунок 2.3: Архітектура залишкового блоку TCN

У порівнянні з традиційними рекурентними архітектурами, TCN пропонують низку значних переваг, особливо актуальних для задач прогнозування енергоспоживання.

На відміну від RNN, які за своєю природою є послідовними (обчислення в момент t залежать від результату в момент $t - 1$), згорткові операції в TCN можуть виконуватися паралельно для всієї вхідної послідовності. Це дозволяє ефективно використовувати сучасні апаратні прискорювачі (GPU/TPU) і

призводить до значно швидшого процесу навчання та генерації прогнозів порівняно з LSTM або GRU еквівалентної ємності.

Завдяки використанню згорток та залишкових з'єднань, шлях зворотного поширення градієнтів у TCN є коротшим та стабільнішим порівняно з розгортанням RNN у часі. Це робить TCN менш схильними до проблем вибухаючого та зникаючого градієнтів, що полегшує навчання глибоких моделей для захоплення довгострокових залежностей.

Розмір рецептивного поля TCN чітко визначається архітектурними параметрами (глибина мережі, розмір фільтра, фактори розширення), що забезпечує гнучке та контрольоване рецептивне поле. Це дозволяє налаштовувати модель під конкретну задачу та очікувану довжину часових залежностей у даних, на відміну від LSTM, де контроль над тим, наскільки далеко в минуле "дивиться" модель, є менш явним.

При навчанні RNN за допомогою алгоритму Backpropagation Through Time (BPTT) необхідно зберігати проміжні активації для всієї послідовності, що може призводити до значного споживання пам'яті. TCN, завдяки згортковій структурі, зазвичай потребують менше пам'яті для зберігання проміжних результатів під час навчання.

Пряме порівняння з LSTM/GRU підкреслює переваги TCN, що обґрунтовує вибір саме цієї архітектури як базової для гібридної моделі. Завдяки описаним вище властивостям, TCN є надзвичайно перспективним інструментом для вирішення завдань прогнозування енергоспоживання в системах розумного будинку:

Здатність ефективно моделювати довгострокові залежності дозволяє TCN враховувати добову, тижневу та сезонну періодичність, що є характерною для патернів енергоспоживання. Висока обчислювальна ефективність робить їх придатними для обробки великих обсягів даних з високою частотою та потенційного розгортання на пристроях з обмеженими ресурсами. Стабільність навчання та гнучкість архітектури полегшують побудову точних та надійних прогнозних моделей.

Таким чином, теоретичні основи та архітектурні особливості TCN роблять їх потужним інструментом для аналізу часових рядів та обґрунтованим вибором як базової моделі для розробки вдосконаленого гібридного методу прогнозування енергоспоживання, що є предметом даного розділу дисертації.

2.2. Теоретичні основи градієнтного бустингу LightGBM

Градiєнтний бустинг (Gradient Boosting) є однією з найпотужніших та найпоширеніших технік ансамблевого машинного навчання, яка демонструє виняткову ефективність у широкому спектрі завдань регресії та класифікації, особливо при роботі зі структурованими (табличними) даними. Ця методологія належить до сімейства алгоритмів бустингу, основна ідея яких полягає в послідовному навчанні ансамблю "слабких" базових моделей (weak learners), найчастіше неглибоких дерев рішень (decision trees). На відміну від методів типу беггінгу (як Random Forest), де моделі навчаються паралельно та незалежно, бустинг є ітераційним процесом: кожна наступна модель в ансамблі фокусується на виправленні помилок, допущених попередніми моделями.

Градiєнтний бустинг, запропонований Джеромом Фрідманом, формалізує цей процес як адитивне моделювання та оптимізацію у функціональному просторі. Замість того, щоб безпосередньо виправляти помилки класифікації чи регресії, кожна нова слабка модель $h_m(x)$ навчається апроксимувати негативний градієнт функції втрат

$$L(y, F_{m-1}(x)), \quad (2.3)$$

відносно прогнозу попереднього ансамблю

$$F_{m-1}(x), \quad (2.4)$$

є напрямком найшвидшого зменшення помилки (аналогічно до градієнтного спуску в просторі параметрів). Нова модель додається до ансамблю з певним кроком навчання (learning rate), поступово покращуючи загальний прогноз

$$F_{m(x)} = F_{\{m-1\}(x)} + u \cdot h_m(x), \quad (2.5)$$

де u – крок навчання.

Цей ітераційний підхід дозволяє ансамблю поступово "спускатися" до мінімуму функції втрат, створюючи все більш точну та робастну композитну модель.

Стандартна реалізація градієнтного бустингу на деревах рішень (Gradient Boosting Machine, GBM) є потужною, однак при роботі з великими наборами даних, великою кількістю спостережень та ознак, традиційні алгоритми GBM стикаються з обчислювальними вузькими місцями. Основна проблема полягає у процесі побудови дерев рішень на кожній ітерації. Для знаходження оптимального розбиття у кожному вузлі дерева необхідно перебрати всі ознаки та всі можливі значення, або точки розбиття для неперервних ознак, для кожної ознаки, обчислюючи при цьому приріст інформації (information gain). Цей процес є дуже ресурсомістким, особливо при великій кількості даних та ознак високої розмірності. Крім того, багато традиційних реалізацій використовують level-wise (порівневий) ріст дерева, що може бути неефективним.

LightGBM (Light Gradient Boosting Machine) — це відносно нова, але вже популярна, високоефективна, розподілена реалізація алгоритму градієнтного бустингу. Її ключова перевага полягає у досягненні значно вищої швидкості навчання, часто в десятки разів швидше, та меншому споживанні пам'яті порівняно з іншими популярними реалізаціями, такими як стандартний GBM або XGBoost, при збереженні високої або навіть кращої точності. Такі вражаючі переваги досягаються завдяки кільком ключовим інноваційним технікам, що оптимізують процес побудови дерев рішень.

Однією з головних інновацій LightGBM є Gradient-based One-Side Sampling (GOSS) – розумна техніка семплювання даних для побудови дерев. Традиційні методи бустингу використовують усі навчальні екземпляри для розрахунку градієнтів та пошуку розбиттів. GOSS базується на спостереженні, що екземпляри даних з великими градієнтами, тобто ті, на яких поточний ансамбль сильно помиляється, вносять значно більший внесок у розрахунок приросту інформації і, отже, є більш "важливими" або "інформативними" для подальшого навчання.

Екземпляри з малими градієнтами (на яких модель вже добре навчилася) є менш важливими.

Алгоритм GOSS працює наступним чином:

1. Всі навчальні екземпляри сортуються за абсолютним значенням їхніх градієнтів.
2. Вибирається певна частка a екземплярів з найбільшими градієнтами (ці екземпляри завжди залишаються у вибірці).
3. З решти екземплярів (з меншими градієнтами) випадковим чином вибирається невелика частка b .

Таким чином, для побудови наступного дерева використовується значно менша підмножина даних, що складається з усіх "важких" екземплярів та невеликої вибірки "легких". Щоб зберегти точність розрахунку приросту інформації та не спотворити розподіл даних, екземпляри з малими градієнтами, що потрапили у вибірку, отримують збільшену вагу при обчисленнях (множаться на константу $(1 - a)/b$).

Завдяки GOSS, LightGBM може значно скоротити кількість даних, що розглядаються на кожному кроці побудови дерева, що призводить до суттєвого прискорення навчання, особливо на великих датасетах, без значної втрати точності.

Детальне пояснення GOSS підкреслює, як LightGBM досягає швидкості, зберігаючи точність, що є ключовим для його вибору.

Другою важливою інновацією є Exclusive Feature Bundling (EFB) – техніка для зменшення ефективної кількості ознак. У багатьох реальних наборах даних, особливо з великою кількістю ознак, багато з них є розрідженими, тобто містять багато нульових значень. Часто серед таких ознак є групи, які є взаємовиключними, тобто вони рідко (або ніколи) не приймають ненульові значення одночасно для одного й того ж екземпляра даних, наприклад, one-hot encoded категоріальні змінні.

EFB використовує цю властивість для об'єднання таких взаємовиключних ознак в єдину "зв'язку" (bundle) або нову синтетичну ознаку. Наприклад, якщо дві

бінарні ознаки A і B ніколи не дорівнюють 1 одночасно, їх можна представити однією новою ознакою C , де $C=0$, якщо $A=0$ і $B=0$; $C=1$, якщо $A=1$ і $B=0$; $C=2$, якщо $A=0$ і $B=1$. Цей процес дозволяє зменшити розмірність простору ознак без суттєвої втрати інформації. LightGBM використовує спеціальний алгоритм (greedy bundling або на основі побудови графу) для знаходження оптимальних зв'язок ознак.

Зменшення кількості ознак, що розглядаються при побудові дерева, призводить до додаткового значного прискорення процесу навчання, оскільки на кожному вузлі потрібно перевіряти менше потенційних розбиттів. EFB пояснює ще один механізм прискорення LightGBM, особливо релевантний для даних з великою кількістю ознак, що може виникнути після інженерії ознак у гібридній моделі.

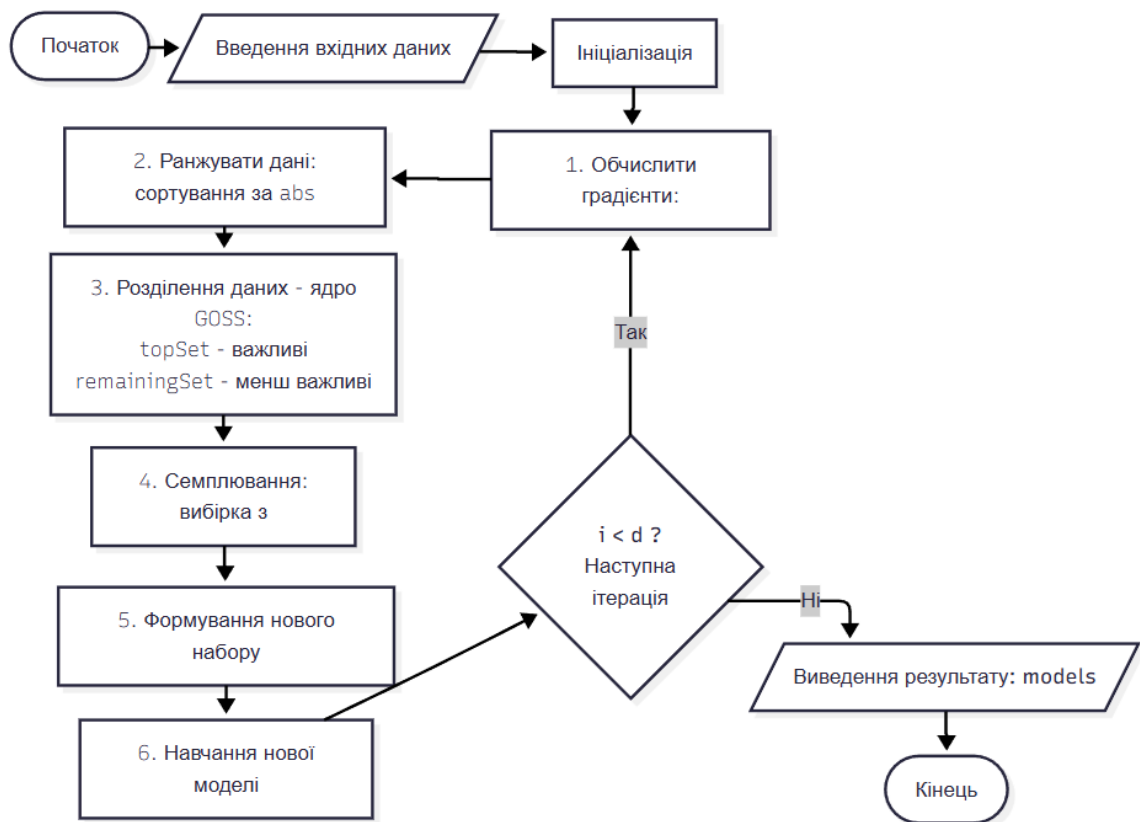


Рисунок 2.4: Ілюстрація Gradient-based One-Side Sampling (GOSS)

На відміну від більшості поширених реалізацій градієнтного бустингу (зокрема, базової конфігурації XGBoost), які ґрунтуються на стратегії порівневої

побудови (level-wise tree growth), алгоритм LightGBM за замовчуванням використовує стратегію зростання за листами (leaf-wise tree growth).

Сутність стратегії зростання за листами полягає в тому, що на кожній ітерації алгоритм обирає для розгалуження той листовий вузол, розділення якого забезпечує максимальне зменшення значення глобальної функції втрат (maximum loss reduction). Такий підхід зумовлює формування асиметричних дерев, які, як правило, мають більшу глибину порівняно з симетричними деревами порівневої стратегії. Це дозволяє моделі ефективніше адаптуватися до складної структури даних та досягати вищої точності апроксимації при фіксованій кількості термінальних вузлів.

Водночас варто зазначити, що стратегія зростання за листами характеризується підвищеним ризиком перенавчання, особливо при роботі з вибірками обмеженого обсягу. Тому при використанні LightGBM важливо ретельно налаштовувати параметри регуляризації, що обмежують складність дерева, зокрема максимальну глибину (max_{depth}) та мінімальну кількість екземплярів у листі (min_{data}).

Розуміння механізму зростання за листами є критичним для обґрунтування вищої прогностичної здатності LightGBM та вибору стратегії запобігання перенавчанню.

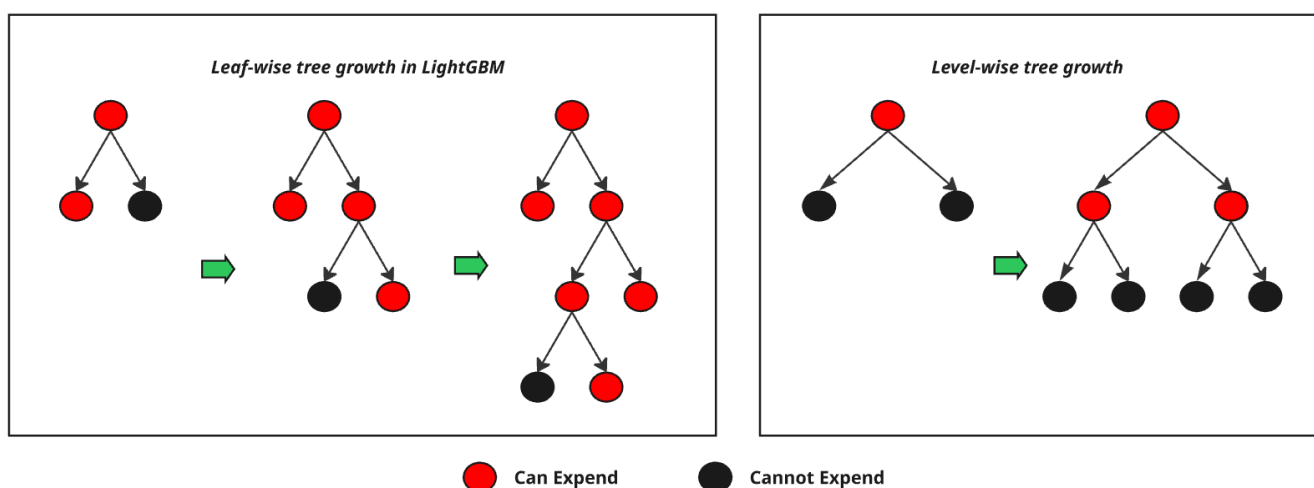


Рисунок 2.5: Порівняння level-wise та leaf-wise росту дерева

Хоча це не є унікальним для LightGBM, варто зазначити, що LightGBM ефективно застосовує алгоритм на основі гістограм для знаходження оптимальних точок розбиття для неперервних ознак. Замість того, щоб сортувати всі значення ознаки та перебирати всі можливі точки розбиття, що є дуже повільним,

$$O(data \times features),$$

алгоритм на першому етапі дискретизує неперервні ознаки, розбиваючи їх на невелику кількість бінів (bins) та будуючи гістограми. Потім пошук оптимального розбиття виконується набагато швидше, перебираючи лише межі бінів

$$O(bins \times features).$$

Це значно зменшує обчислювальну складність та споживання пам'яті. Коли набір даних містить значну кількість записів, кількість точок даних (data) часто набагато більша за кількість вибраних інтервалів (bins). Розглянемо приклад.

$$data = 10\,000\,000$$

$$features = 100$$

$$bins = 256$$

Алгоритм, який дорівнює $O(n \times d)$, матиме складність, пропорційну:

$$10\,000\,000 * 100 = 1\,000\,000\,000$$

Крок пошуку розщеплення алгоритму на основі гістограм становитиме $O(k \times d)$, зі складністю, пропорційною:

$$256 * 100 = 25\,600$$

Ця оптимізація пояснює, чому методи на основі гістограм значно швидші для навчання на великих наборах даних.

Завдяки описаним інноваціям, LightGBM пропонує низку ключових переваг:

1. Висока швидкість навчання та прогнозування, значно швидший за інші реалізації GBM.
2. Низьке споживання пам'яті, ефективно працює на великих датасетах.
3. Висока точність, часто досягає кращої або порівнянної точності з іншими state-of-the-art алгоритмами.
4. Підтримка паралельного та розподіленого навчання, добре масштабується на багатоядерних процесорах та кластерах.

5. Здатність обробляти великі обсяги даних, ефективно працює з мільйонами екземплярів та тисячами ознак.

Перелічені характеристики роблять LightGBM кандидатом для використання у гібридних моделях прогнозування енергоспоживання, зокрема у поєднанні з TCN або LSTM. Здатність швидко та ефективно обробляти великі обсяги табличних даних, включаючи інженерні ознаки, такі як лаги, статистики, календарні змінні, та виявляти складні нелінійні взаємозв'язки між ними чудово доповнює здатність TCN/LSTM моделювати темпоральні залежності у сирих часових рядах. У контексті стратегії "прогнозування залишків", LightGBM виступає як потужна модель-коректор, що ефективно навчається на помилках базової моделі глибокого навчання, значно підвищуючи загальну точність прогнозу, особливо для складних патернів, таких як пікові навантаження. Його обчислювальна ефективність також є критично важливою для можливості оптимізації гібридної моделі та її потенційного розгортання в системах реального часу або на пристроях з обмеженими ресурсами.

2.3. Розробка гібридної моделі TCN-LightGBM для прогнозування енергоспоживання.

Для вирішення складної задачі короткострокового прогнозування енергоспоживання (STLF) в системах розумного будинку, яка характеризується нелінійними патернами, сезонністю, волатильністю та впливом зовнішніх факторів, було розроблено та реалізовано гібридний підхід. Цей підхід спрямований на подолання обмежень, властивих одиночним моделям (standalone models), шляхом синергетичного поєднання сильних сторін двох передових технологій машинного навчання: темпоральної згорткової мережі (TCN) та градієнтного бустингу LightGBM.

Аналіз літератури (Розділ 1) та результати попередніх експериментів показали, що хоча моделі глибокого навчання, такі як TCN, є надзвичайно ефективними у автоматичному вилученні складних часових залежностей та

патернів безпосередньо з сирих даних часового ряду, вони можуть допускати систематичні помилки. Особливо це стосується прогнозування нелінійних пікових навантажень або точного врахування впливу різнорідних зовнішніх факторів. З іншого боку, моделі на основі дерев рішень, зокрема LightGBM, демонструють виняткову продуктивність при роботі зі структурованими (табличними) даними. Вони здатні ефективно моделювати складні нелінійні взаємозв'язки між великою кількістю різнорідних ознак і є відносно стійкими до шумів.

Для подолання обмежень окремих алгоритмів розроблено гібридну архітектуру, що базується на синергетичній інтеграції двох методів машинного навчання: темпоральної згорткової мережі (TCN) та градієнтного бустингу (LightGBM).

Обґрунтуванням вибору саме такої комбінації є взаємодоповнюючий характер переваг обраних методів. TCN забезпечує ефективну автоматичну екстракцію складних латентних часових залежностей безпосередньо з «сирих» даних, тоді як LightGBM демонструє високу узагальнюючу здатність при обробці структурованих табличних даних та моделюванні нелінійних взаємозв'язків у багатовимірному просторі ознак.

У ході дослідження проведено порівняльний аналіз двох стратегій гібридизації:

1. Коригування піків (Peak Correction): використання LightGBM для прогнозування моменту та амплітуди піку з подальшою точковою корекцією прогнозу TCN.

2. Прогнозування залишків (Residual Fitting): навчання LightGBM на часовому ряді помилок (залишків) моделі TCN для адитивної корекції всього горизонту прогнозування.

Порівняльний аналіз показав, що хоча обидві стратегії дозволили покращити точність порівняно з базовою моделлю TCN, стратегія "прогнозування залишків" виявилася значно ефективнішою. Зокрема, модель TCN-LGBM (Прогн. залишків) продемонструвала кардинальне покращення точності прогнозування саме пікових навантажень, знизивши показник Peak Magnitude MAPE з 22.40% у базовій TCN

до 16.71%, що було значно краще за результат найкращої моделі з "коригуванням піків" (20.89%). Це пояснюється тим, що "прогнозування залишків" дозволяє коригувати всі систематичні помилки базової моделі вздовж усього горизонту прогнозування, а не лише точково в моменти піків. Тому саме ця стратегія була обрана для розробки фінальної гібридної архітектури.

Результати експериментальної верифікації стратегій наведено в Таблиці 2.1.

Таблиця 2.1

Порівняльна ефективність стратегій гібридизації моделі.

Архітектура моделі	Стратегія взаємодії	Принцип дії	Peak Magnitude MAPE, %	Ефективність відносно базової TCN
TCN (Baseline)	—	Монолітна модель глибокого навчання	22.40	—
TCN + Peak Correction	Точкова корекція	Прогнозування параметрів піку окремими регресорами	20.89	Незначне покращення (+1.51 в.п.)
TCN + Residual Fitting	Адитивна корекція	Прогнозування систематичної помилки (залишків) на всьому горизонті	16.71	Суттєве покращення (+5.69 в.п.)

Загальна архітектура розробленої гібридної моделі TCN та LightGBM, що реалізує стратегію "прогнозування залишків", є двоступеневою (two-stage), послідовною композицією. У цій композиції нейронна мережа TCN виступає як основний (базовий) прогностичний модуль, а ансамбль моделей LightGBM – як вторинний модуль-коректор. Детальна компонентна діаграма цієї архітектури та потоків даних представлена на рис. 2.6.

Процес отримання прогнозу на нових даних (тобто етап використання моделі або inference), який виконується після завершення всіх етапів навчання, складається з наступних чотирьох послідовних етапів.

Першим етапом є генерація базового прогнозу з використанням лише навченої моделі TCN для отримання первинної оцінки майбутнього споживання.

На вхід базової моделі TCN подається вхідна послідовність (lookback window), що містить історичні дані. Ця послідовність включає два типи ознак:

- цільовий часовий ряд агрегованого енергоспоживання (P_{agg}),
- допоміжні інженерні часові ознаки (наприклад, синусоїдальні та косинусоїдальні перетворення години доби, дня року тощо).

У рамках даного дослідження було встановлено, що оптимальна довжина вхідної послідовності становить 96 кроків, що відповідає 24 годинам історичних даних з 15-хвилинною дискретизацією.

Модель TCN обробляє цю вхідну послідовність. Архітектура моделі, що використовується, складається з одного блоку TCN (який внутрішньо містить стек розширених згорток з 64 фільтрами, розміром ядра $k=3$ та експоненційно зростаючими факторами розширення $d = (1, 2, 4, 8)$), за яким слідує два повнозв'язних (Dense) шари. Така структура дозволяє мережі автоматично вилучати складні багаторівневі темпоральні патерни та довгострокові залежності з вхідних даних.

Результатом роботи вихідних повнозв'язних шарів TCN є формування вектору базового прогнозу агрегованого енергоспоживання $P_{agg\ pred\ TCN}$ на встановлений горизонт прогнозування. Відповідно до визначених умов задачі, глибина прогнозування становить 96 дискретних часових кроків (що еквівалентно часовому інтервалу в 24 години). Отриманий результат інтерпретується як перше наближення, що відтворює основні детерміновані закономірності динаміки часового ряду.

На другому етапі відбувається підготовка та збагачення ознак для моделі-коректора LightGBM. Формується вектор ознак (feature vector), який буде використаний другим компонентом системи – моделлю LightGBM – для прогнозування помилок (залишків) базового прогнозу.

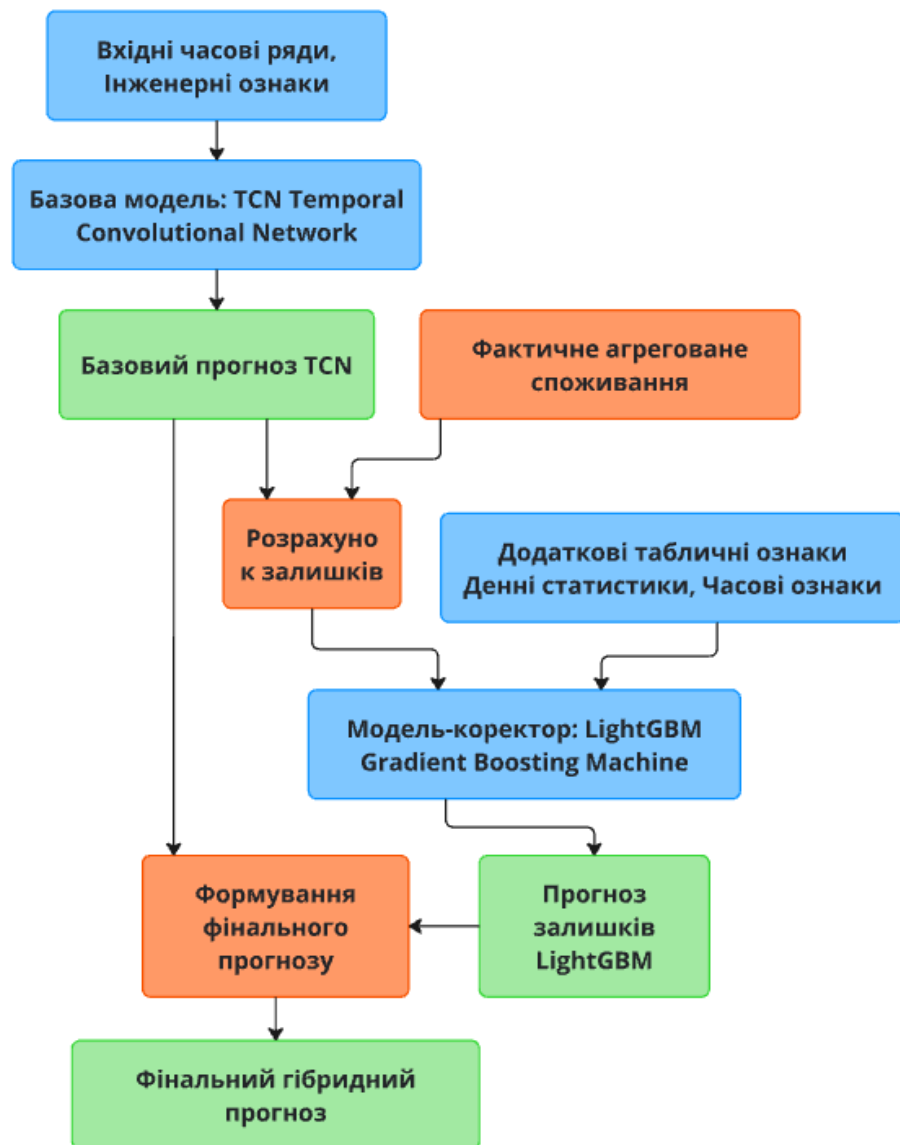


Рисунок 2.6: Архітектура гібридної моделі TCN та LightGBM зі стратегією «прогнозування залишків»

Для ефективної генерації прогнозу залишків, модель-коректор LightGBM потребує набору високоінформативних ознак. Цей набір є агрегацією трьох різних груп даних:

Базовий прогноз від TCN $P_{agg\ pred\ TCN}$ отриманий на першому етапі, використовується як одна з найважливіших ознак для коректора. Це логічно, оскільки щоб спрогнозувати помилку, модель повинна знати сам прогноз, який вона коригує.

Часові ознаки (\sin/\cos години, дня тощо), що подавалися на вхід TCN. Їх повторне використання дозволяє моделі LightGBM явно "бачити" календарний контекст, що може допомогти виявити систематичні помилки TCN, прив'язані до певного часу доби.

Додаткові табличні ознаки (External Features) ключовий елемент гібридизації. Він дозволяє моделі LightGBM врахувати ту інформацію, яка була потенційно недоступною або складною для інтерпретації моделлю TCN (яка в даній конфігурації бачила лише агрегований ряд).

У даному дослідженні було емпірично встановлено, що ефективними ознаками є денні статистичні показники (середнє, максимум, мінімум, стандартне відхилення, медіана тощо) індивідуального споживання за попередні дні (лаги 1, 2, ...). Причому ці статистики розраховувалися лише для 20 домогосподарств з найбільшим середнім споживанням. Вказані показники виступають як потужні індикатори загальних патернів поведінки та потенційних аномалій саме у найбільших споживачів. Оскільки піки агрегованого споживання часто визначаються саме такими "важкими" споживачами, надання цієї дезагрегованої інформації моделі-коректору допомагає їй значно краще прогнозувати помилки TCN саме в періоди пікових навантажень.

На третьому етапі прогнозування залишків використовується ансамбль моделей LightGBM, навчених на етапі тренування виключно задачі прогнозування помилок TCN. Сформований на попередньому кроці розширений набір ознак подається на вхід навчених моделей LightGBM.

Для підвищення ефективності, гнучкості та точності прогнозування на довгий горизонт, використовується не одна велика модель, а набір з 96 окремих моделей LightGBM, що функціонує за принципом "одна модель – один крок прогнозу". Тобто, кожна i -та модель LightGBM (де $i = 1, \dots, 96$) навчалася (на етапі тренування) прогнозувати лише одне значення – залишок (помилку) $TCN_{residuals}$ саме для i -го кроку вперед. При цьому всі вони використовують ознаки, доступні на момент початку прогнозування (час t).

Така пряма стратегія прогнозування дозволяє кожній окремій моделі глибоко спеціалізуватися на прогнозуванні помилки для свого конкретного часового кроку, наприклад, модель для кроку $t + 1$ може мати інший набір важливих ознак, ніж модель для кроку $t + 96$), що дає більш точні результати, ніж одна універсальна модель для всього горизонту. Також це розбиває одну велику складну задачу на 96 менших, що полегшує процес навчання та може допомогти уникнути проблем з об'ємом пам'яті.

Кожна з 96 моделей LightGBM генерує свій прогноз залишку. Всі разом вони формують фінальний вектор прогнозів залишків (*Residuals pred LGBM*) на весь 24-годинний горизонт.

Етап 4: Формування фінального гібридного прогнозу. Завершальний етап, на якому відбувається синтез результатів роботи обох модулів.

Фінальний гібридний прогноз енергоспоживання (*Final HybridForecast*) отримується шляхом простої та обчислювально ефективною операції – поелементного додавання (сумування) двох векторів: (1) базового прогнозу від TCN та (2) прогнозу залишків (помилки) від LightGBM.

$$FinalHybridForecast = P_{aggpred_{TCN}} + Residuals_{pred_{LGBM}}, \quad (2.6)$$

Отриманий в результаті фінальний прогноз є синергетичним продуктом. Він успадковує та враховує як складні нелінійні темпоральні залежності, виявлені глибинною мережею TCN, так і корекцію систематичних помилок, яку здійснив ансамбль LightGBM на основі аналізу розширеного набору табличних та статистичних ознак.

Під час проведення експериментального дослідження, всі нейронні мережі (TCN та інші базові моделі для порівняння) навчалися з використанням адаптивного оптимізатора Adam зі стандартною швидкістю навчання 0.001. В якості функції втрат (loss function), яку мінімізував оптимізатор, була обрана середньоквадратична помилка ('mean_squared_error', MSE). Для запобігання перенавчанню та для автоматичного визначення оптимальної кількості епох навчання, обов'язково застосовувався механізм ранньої зупинки (EarlyStopping). Цей механізм відстежував значення функції втрат на окремому валідаційному

наборі даних і зупиняв процес навчання, якщо покращення помилки на валідації не спостерігалось протягом 5 послідовних епох (параметр $patience=5$).

Розроблена гібридна модель TCN-LightGBM зі стратегією "прогнозування залишків" є обґрунтованим та ефективним рішенням для задачі короткострокового прогнозування енергоспоживання. Вона синергетично поєднує здатність TCN до моделювання складних часових послідовностей та потужність LightGBM в аналізі різномірних табличних ознак для корекції помилок базової моделі. Використання спеціалізованих моделей LightGBM для кожного кроку прогнозу та розширеного набору ознак дозволяє досягти високої точності, особливо у прогнозуванні критично важливих пікових навантажень. Подальша оптимізація цієї архітектури шляхом відбору ознак робить її обчислювально ефективною для практичного застосування.

2.4. Розробка методики оптимізації моделі для досягнення балансу точності та обчислювальної ефективності.

Однією з ключових проблем при впровадженні складних гібридних моделей у реальні системи розумного будинку є компроміс між точністю прогнозування та обчислювальною вартістю. Як було продемонстровано в попередніх підрозділах та підтверджено експериментально, розроблена гібридна модель TCN + LightGBM зі стратегією "прогнозування залишків" досягла найвищої точності у прогнозуванні критично важливих пікових навантажень (Peak Magnitude MAPE 16.71%). Ця здатність точно передбачати піки є надзвичайно цінною для систем управління попитом (Demand Side Management) та оптимізації роботи енергосистем. Однак ця висока точність була досягнута ціною значних обчислювальних витрат. Середній час навчання цієї моделі на одному блоці крос-валідації становив 305,72 секунд.

Такий тривалий час навчання може стати суттєвим бар'єром для практичного впровадження моделі, особливо в контексті систем Інтернету речей (IoT). Пристрої IoT, такі як контролери розумного будинку або навіть периферійні

шлюзи, часто мають обмежені обчислювальні ресурси (процесорну потужність, обсяг пам'яті, енергоспоживання). Необхідність періодичного перенавчання моделі (наприклад, для адаптації до змін у патернах споживання) може виявитися нездійсненною або надто енергозатратною на таких пристроях, якщо процес займає багато часу. Крім того, навіть при навчанні в хмарі, скорочення часу навчання дозволяє швидше ітерувати, тестувати гіпотези та розгортати оновлені моделі.

Результати аналізу фахових літературних джерел свідчать про те, що питання пошуку балансу між точністю прогнозування та обчислювальними витратами (часом навчання, апаратними вимогами) у гібридних моделях STLFF залишається недостатньо дослідженим. Встановлено, що домінуючий вектор сучасних розробок спрямований переважно на максимізацію метрик точності, тоді як оптимізація швидкості навчання часто залишається поза увагою, що складає наукову прогалину в даній предметній області.

З метою подолання виявленої проблеми високої обчислювальної складності, у роботі запропоновано спеціалізований метод оптимізації. Основна мета розробки полягає у мінімізації сукупного часу навчання гібридної моделі TCN-LightGBM при забезпеченні умови збереження високої точності прогнозування, зокрема в частині коректного моделювання критичних пікових навантажень.

Теоретична логіка запропонованої методики базується на обґрунтованому припущенні, що переважна частина обчислювального навантаження та, відповідно, часу навчання, припадає не на відносно швидку нейромережеву TCN-компоненту, а на навчання багатофакторної моделі-коректора LightGBM. Ця модель у вихідній конфігурації змушена обробляти великий та різномірний за своєю природою набір вхідних ознак. Висувається гіпотеза, що далеко не всі ці ознаки роблять однаково значущий внесок у кінцеву якість корекції помилок, особливо для специфічних випадків прогнозування пікових значень. Значна частина ознак може бути надлишковою, статистично малоінформативною або навіть вносити шум, що ускладнює навчання. Незважаючи на їхню низьку корисність, їхня формальна обробка, зокрема, ітеративний пошук оптимальних

розбиттів у процесі побудови ансамблю дерев рішень, все одно вимагає значних обчислювальних ресурсів та часу.

Тому запропонована методика оптимізації полягає у цілеспрямованому, багатоетапному відборі найважливіших ознак. Принциповим є те, що ця процедура застосовується таргетовано та виключно до вхідного простору моделі-коректора LightGBM. Базова модель TCN, що відповідає за моделювання часових залежностей, залишається незмінною, що дозволяє зберегти її основні прогностичні переваги. Описана схема оптимізації наведена на Рисунку 2.5.

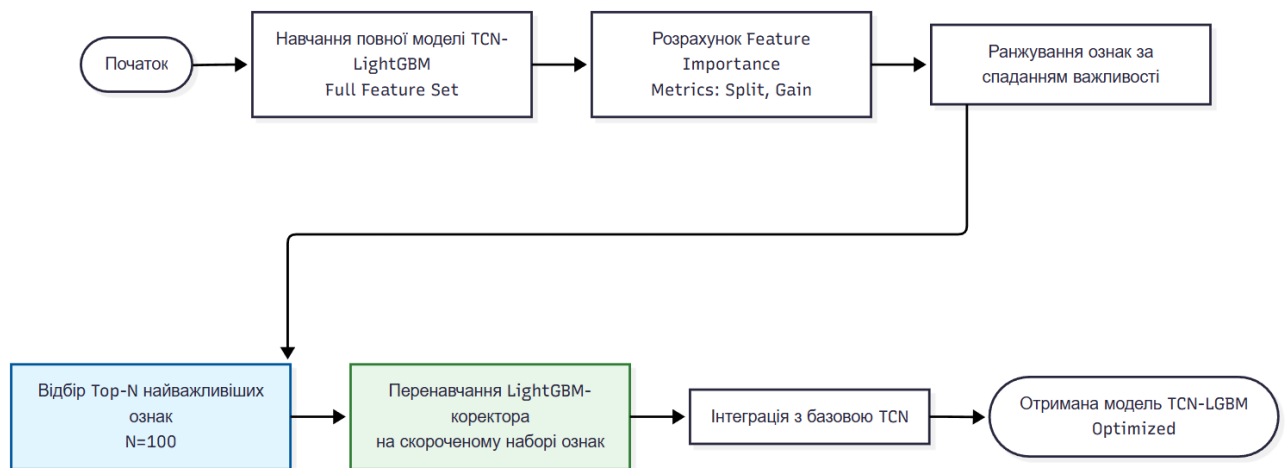


Рисунок 2.7: Блок-схема методу оптимізації гібридної моделі шляхом відбору ознак

Процедура оптимізації була реалізована як послідовність чітко визначених кроків.

На першому етапі проводиться повне навчання вихідної, неоптимізованої гібридної моделі TCN + LightGBM, що використовує стратегію "Прогнозування залишків", на всьому доступному наборі тренувальних даних. На вхід коректора подається повний простір ознак, що включає базовий прогноз TCN, усі часові та календарні ознаки, а також всі згенеровані денні статистичні показники для домогосподарств. Безпосередньо після завершення цього ресурсоємного навчання, використовуючи вбудований у бібліотеку LightGBM механізм, розраховується важливість (feature importance) кожної ознаки, що подавалася на вхід навченої моделі-коректора.

Оцінка важливості проводилася з використанням двох стандартних та інтерпретованих критеріїв, що надаються фреймворком:

- 'split': підраховує кількість разів, коли конкретна ознака використовувалася для розбиття внутрішнього вузла в деревах ансамблю;

- 'gain': оцінює сукупне зменшення значення функції втрат (або сумарний приріст інформації), який було досягнуто завдяки всім розбиттям, здійсненим за цією ознакою. Цей критерій зазвичай вважається більш показовим з точки зору впливу на точність.

В результаті було отримано ранжований список всіх вхідних ознак коректора, відсортований від найбільш до найменш важливих. Варто зазначити, що сам по собі аналіз цього рейтингу є цінним проміжним науковим результатом. Він дозволяє глибше зрозуміти, яка саме інформація (зовнішні фактори, часові характеристики, статистичні агрегати чи власне прогноз TCN) є ключовою для корекції помилок базової нейромережевої моделі. На основі отриманого рейтингу важливості було виконано безпосередньо крок відбору, що полягав у виборі фіксованої кількості N найвпливовіших ознак.

Вибір значення N є гіперпараметром самої методики оптимізації і потребує певного експериментального підбору або визначення на основі аналізу кривої спадання кумулятивної важливості. У рамках даного дослідження шляхом проведення серії попередніх експериментів було емпірично встановлено, що відбір $N = 100$ найважливіших ознак становить оптимальний компроміс між радикальним скороченням кількості ознак, як наслідок, очікуваного часу навчання, та збереженням достатнього обсягу інформації, необхідної для високоточної корекції.

На завершальному етапі було сформовано нову, оптимізовану конфігурацію гібридної моделі, яка отримала умовне позначення TCN-LGBM (Оптимізована). Процес її навчання мав дві ключові особливості. Базова модель TCN залишалася незмінною: використовувалися ті ж самі ваги, що були навчені на початковому етапі, або ж вона навчалася ідентично до вихідної моделі. Модель-коректор LightGBM навчалася заново, використовуючи ту саму методологію крос-валідації,

але цього разу на вхід їй подавався лише відібраний, суттєво скорочений набір зі 100 найважливіших ознак.

Саме порівняльний аналіз показників точності та сукупного часу навчання цієї оптимізованої моделі з вихідною "повною" моделлю дозволив зробити обґрунтовані висновки про ефективність та доцільність запропонованої методики.

Апробація розробленої методики оптимізації в рамках експериментального дослідження продемонструвала її надзвичайну ефективність та практичну значущість. Порівняння оптимізованої моделі TCN-LGBM (Оптимізована) з повною моделлю TCN-LGBM (Прогн. залишків) показало наступні результати.

Застосування відбору ознак дозволило скоротити середній час навчання моделі-коректора (а отже, і всієї гібридної моделі на етапі корекції) в 5,4 рази – з 305,72 секунд до 56,47 секунд, що є ключовим результатом, який переводить модель з розряду потенційно ресурсомістких у розряд цілком прийнятних для багатьох практичних сценаріїв, включаючи періодичне перенавчання на периферійних пристроях або швидкі ітерації при розробці.

Найбільш вражаючим і неочевидним результатом є те, що це кардинальне прискорення було досягнуто практично без погіршення здатності моделі прогнозувати критичні пікові навантаження. Показник Peak Magnitude MAPE для оптимізованої моделі склав 16,77%, що лише на 0,06 відсоткових пункти гірше, ніж у повної моделі (16,71%). Ця різниця є статистично незначущою і свідчить про те, що відібрані 100 ознак містять практично всю необхідну інформацію для ефективної корекції саме пікових помилок базової моделі TCN. Це експериментально доводить, що для вирішення найскладнішої частини задачі, а саме прогнозування піків, не потрібен повний, потенційно надлишковий набір ознак.

Загальна точність прогнозування, що вимірюється метриками MAPE та RMSE, зазнала невеликого погіршення. MAPE зріс з 13,43% до 13,82%, а RMSE – з 1,7121 до 1,7504. Це вказує на те, що відкинуті (менш важливі) ознаки все ж мали певний, хоч і незначний, позитивний вплив на прогнозування загальної

форми часового ряду (непікових значень), але їхня відсутність не вплинула на якість прогнозування екстремальних значень.

Описана порівняльна характеристика ефективності наведена в Таблиці 2.1.

Таблиця 2.2

Порівняльна характеристика ефективності повної та оптимізованої моделей

Показник ефективності	TCN-LGBM (Повна)	TCN-LGBM (Оптимізована)	Відносна зміна
Час навчання (Training Time), с	305,72	56,47	Прискорення в 5,4 рази
Peak Magnitude MAPE, % (Точність піків)	16,71	16,77	0,06 в.п.
Загальна MAPE, %	13,43	13,82	0,39 в.п.
RMSE, кВт	1,7121	1,7504	0,0383
Кількість вхідних ознак коректора	Повний набір	100 (Top-N)	Суттєве скорочення розмірності

Аналіз важливості ознак, проведений на першому кроці методики, теж надав цінні інсайти. Було виявлено, що найбільший вплив на корекцію помилок базової моделі TCN мають статистичні показники споживання за попередній день, зокрема максимальні (max), мінімальні (min) та середні (mean) значення як для агрегованого навантаження, так і для окремих "сигнальних" домогосподарств. Це підтверджує логічність та інтерпретованість гібридного підходу, де LightGBM використовує інформацію, яка може бути складною для прямого моделювання TCN, наприклад, екстремальні значення окремих споживачів як індикатори загального піку.

Розроблена методика оптимізації гібридної моделі TCN + LightGBM шляхом цілеспрямованого відбору найважливіших ознак для моделі-коректора є ефективним, обґрунтованим та практично значущим результатом дослідження. Вона дозволяє кардинально (в 5,4 рази) скоротити час навчання при збереженні високої точності прогнозування критичних пікових навантажень. Це вирішує ідентифіковану проблему компромісу між точністю та обчислювальною ефективністю, що є критичним для практичного впровадження потужних

гібридних моделей в реальні системи управління енергією в розумних будинках, особливо на пристроях з обмеженими ресурсами. Запропонований підхід надає збалансовану конфігурацію, яка є одночасно високоточною та обчислювально доступною.

Висновки до розділу 2

У цьому розділі було розроблено та досліджено вдосконалений гібридний метод для короткострокового прогнозування енергоспоживання, що є ключовим компонентом для оптимізації керування системами розумного будинку. На основі проведеного аналізу та експериментів було зроблено наступні висновки.

Обґрунтовано вибір компонентів гібридної моделі. Проаналізовано теоретичні основи та архітектурні переваги темпоральних згорткових мереж (TCN) та градієнтного бустингу LightGBM. Встановлено, що TCN є обчислювально ефективнішою альтернативою рекурентним мережам для аналізу часових послідовностей завдяки паралелізму, тоді як LightGBM демонструє високу швидкість та точність при роботі з табличними даними. Це обґрунтовує доцільність їх синергетичного поєднання.

Розроблено та валідовано ефективну гібридну архітектуру TCN + LightGBM. Експериментально доведено, що стратегія "прогнозування залишків" є значно ефективнішою за інші підходи, зокрема "коригування піків". Запропонована двоступенева модель, де TCN виступає як базовий прогнозист, а LightGBM — як модель-коректор помилок, дозволила досягти суттєвого підвищення точності, особливо у прогнозуванні критичних пікових навантажень.

Розроблено методику оптимізації для досягнення балансу між точністю та обчислювальною ефективністю. Запропоновано алгоритм, що базується на відборі найважливіших ознак для моделі-коректора LightGBM. Апробація даної методики дозволила створити оптимізовану конфігурацію моделі, яка продемонструвала виняткові результати.

Досягнуто ключового практичного результату. Шляхом відбору 100 найважливіших ознак вдалося скоротити середній час навчання моделі в 5,4 рази (з 305,72 до 56,47 секунд). Найважливішим є те, що таке кардинальне прискорення було досягнуто практично без втрати точності у прогнозуванні пікових навантажень (погіршення склало лише 0,06%). Це підтверджує, що розроблений оптимізований підхід є не лише високоточним, але й достатньо ефективним для практичного впровадження на пристроях з обмеженими обчислювальними ресурсами, що є типовим для систем Інтернету речей.

Таким чином, у розділі представлено завершене рішення для задачі прогнозування енергоспоживання, яке вирішує існуючу в науковій літературі проблему компромісу між точністю та обчислювальною вартістю.

РОЗДІЛ 3. МЕТОД АДАПТИВНОГО ОЧИЩЕННЯ РІЗНОРІДНИХ СЕНСОРНИХ ДАНИХ (ACRA)

3.1. Формалізація проблеми гетерогенних шумів у сенсорних даних розумного будинку.

Ефективне, надійне та оптимальне функціонування сучасних систем "розумного будинку", спрямованих на проактивне підвищення якості життя, персоналізованого комфорту, безпеки мешканців та радикальну оптимізацію енергоспоживання, нерозривно та фундаментально пов'язане з якістю та надійністю інформаційних потоків, що генеруються розгалуженою, географічно розподіленою мережею Інтернету речей (IoT-сенсорів). Різноманітні сенсорні пристрої – зокрема високоточні датчики температури, вологості, концентрації CO₂, освітленості, присутності (PIR), руху, а також інтелектуальні прилади обліку споживання енергетичних і комунальних ресурсів (електроенергії, води, газу) – фактично виконують функції системи чутливого сприйняття або своєрідної “нервової системи” будівлі. Вони постачають критично важливі вхідні дані для всіх керуючих підсистем. Для алгоритмів автоматизованого управління (наприклад, клімат-контролю), модулів інтелектуальної аналітики, предиктивних моделей прогнозування (навантаження, поведінки) та систем підтримки прийняття рішень.

Відповідно, точність, повнота, своєчасність та загальна достовірність цих даних мають вирішальне значення. Будь-які, навіть короточасні помилки, спотворення чи пропуски на вході неминуче "отруюють" увесь ланцюжок обробки інформації, що призводить до некоректної роботи інтелектуальних алгоритмів, прийняття субоптимальних або відверто помилкових керуючих рішень, неефективного управління ресурсами, наприклад, марного витрачання енергії через хибні показники температури), помилкових тривог у системах

безпеки та, в кінцевому підсумку, до катастрофічного зниження загальної ефективності, надійності та сприйманої користувачем цінності всієї системи.

Слід чітко розуміти, що необроблені ("сирі") сенсорні дані, які збираються в режимі реального часу, практично ніколи не бувають ідеальними. В реальних умовах експлуатації вони майже завжди містять значну кількість різноманітних пошкоджень, артефактів, шумів та аномалій. Природа цих спотворень є багатофакторною і може виникати з багатьох причин на різних етапах життєвого циклу даних:

1. Фізичні несправності або деградація сенсорів, адже з часом чутливі елементи сенсорів можуть виходити з ладу, втрачати заводське калібрування, зазнавати механічних пошкоджень або просто "старіти".

2. Електромагнітні перешкоди (EMI) такі, як зовнішні електромагнітні поля від потужних побутових приладів (мікрохвильові печі, двигуни) можуть впливати на роботу чутливих аналогових сенсорів або спотворювати сигнали під час передачі даних.

3. Проблеми з передачею даних, наприклад, втрата пакетів, значні затримки (latency), помилки контрольних сум або повторні передачі в завантажених бездротових мережах (Wi-Fi, Zigbee, LoRaWAN) є буденним явищем.

4. Непередбачуваний вплив середовища – робота пристроїв в екстремальних умовах (надто висока/низька температура, вологість, конденсат), вібрації, забруднення або запилення оптичних елементів можуть суттєво впливати на адекватність показників.

5. Збої програмного забезпечення, помилки у вбудованій прошивці (firmware) кінцевих пристроїв (наприклад, переповнення буфера, помилки синхронізації часу) або в логіці програмного забезпечення для збору даних (на шлюзі чи сервері).

6. Інші стохастичні фактори, а саме випадкові, непередбачувані події, які важко класифікувати або передбачити заздалегідь.

Найбільш суттєвою та методологічно складною проблемою є те, що ці пошкодження не є однорідними за своєю природою чи статистичними

властивостями. Навпаки, вони представляють собою складну, динамічну в часі суміш різноманітних артефактів. Такий тип забруднення даних доцільно характеризувати саме як гетерогенний шум.

Нехай $Z = \{z_1, z_2, \dots, z_t, \dots\}$ – це послідовність необроблених (зашумлених) значень, що надходять від одновимірного (univariate) сенсора (наприклад, датчика температури) в дискретні моменти часу $t = 1, 2, \dots$ з певним періодом дискретизації.

В ідеальному, абстрактному випадку, кожне спостереження z_t можна було б представити як просту суму двох компонент: істинного, неспотвореного значення фізичної величини s_t та деякої адитивної компоненти шуму ϵ_t :

$$z_t = s_t + \epsilon_t, \quad (3.1)$$

Однак, на принципову відміну від класичних моделей обробки сигналів, де компонента ϵ_t часто припускається простим випадковим процесом, стаціонарним білим гаусовим шумом з нульовим математичним сподіванням $\epsilon_t \sim N(0, \sigma^2)$, у реальних IoT-системах таке припущення є нереалістичним та надто спрощеним. В дійсності, компонент ϵ_t є складною, нестаціонарною та часто залежною від контексту функцією, яка може приймати абсолютно різні форми та мати різні статистичні характеристики в різні моменти часу.

Детальний аналіз реальних сенсорних даних з експлуатації систем розумного будинку та огляд сучасної наукової літератури в галузі обробки даних IoT дозволяє виділити кілька основних, найбільш поширених та впливових типів аномалій, що в сукупності й складають цей гетерогенний шум.

Точкові викиди (Point Outliers/Spikes) – це короточасні, зазвичай поодинокі, різкі відхилення значення z_t від очікуваного фізіологічного діапазону або від локального тренду істинного сигналу s_t . Вони характеризуються дуже великою амплітудою, але надзвичайно малою тривалістю (найчастіше 1-2 точки даних). Причиною їх виникнення є короточасні електромагнітні імпульси, помилки при аналого-цифровому перетворенні, поодинокі помилки передачі даних.

Дрейф показників (Drift / Bias) – це поступове, систематичне та довготривале зміщення вимірюваних значень z_t від істинних значень s_t . Дрейф може бути

лінійним або нелінійним і розвивається повільно, впродовж тривалого часу (години, дні або навіть місяці). Це особливо "підступний" тип помилки, оскільки його важко виявити простими пороговими методами. Виникає через фізичне старіння сенсора, поступова втрата калібрування під впливом температури або інших зовнішніх факторів. Демонстрація дрейфу даних візуалізовано на графіку з Рисунку 3.1 для показників з 300 по 450.

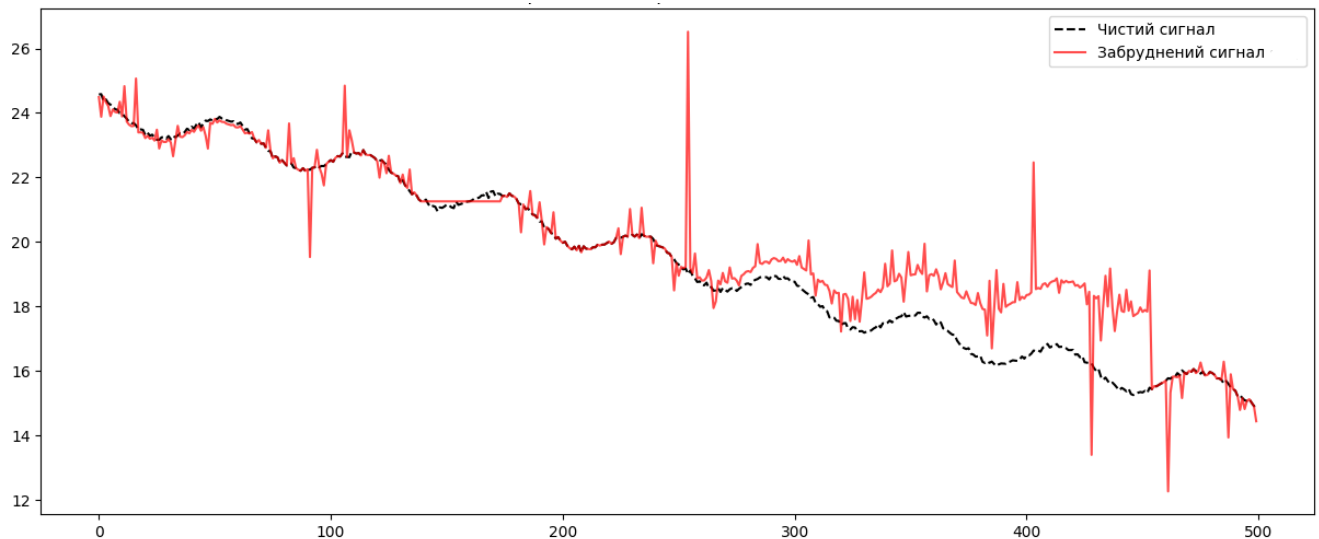


Рисунок 3.1: Візуалізація різних типів гетерогенних шумів на прикладі згенерованих наборів

Періоди константних значень (Constant Values / Stagnant Data / Stuck-at Faults) – це цілі сегменти даних, де значення сенсора z_t залишається абсолютно незмінним (або майже незмінним) впродовж певного, часто тривалого, періоду. При цьому система повністю ігнорує реальні зміни істинного сигналу s_t , який в цей час може змінюватись. Тимчасова несправність ("зависання") сенсора, проблеми з аналого-цифровим перетворювачем (АЦП), втрата зв'язку з шлюзом, яка компенсується програмним повторенням останнього отриманого значення ("last value carry-forward").

Загальний фоновий шум (General / Background Noise) – це випадкові, зазвичай височастотні та низькоамплітудні коливання сигналу z_t навколо його істинного значення s_t . Цей тип шуму в тій чи іншій мірі є невід'ємною частиною роботи будь-якого реального вимірювального приладу і відображає межі його

точності. Його часто моделюють як гаусівський, але в реальності він може мати складніший статистичний характер.

Пропущені значення (Missing Values / NaNs) – це повна відсутність даних у певні очікувані моменти часу. Це призводить до утворення "розривів" у часовому ряді. Причиною є втрата пакетів при мережевій передачі без механізму повтору, збої програмного забезпечення на сервері збору даних, планове або аварійне вимкнення пристрою (наприклад, через розрядження батареї).

Таблиця 3.1

Класифікація гетерогенних шумів у сенсорних мережах IoT

Тип аномалії (Noise Type)	Характеристика сигналу	Часовий характер	Типові причини виникнення
Точкові викиди (Point Outliers)	Різке, значне відхилення амплітуди від локального тренду	Імпульсний, короткотривалий ($t \approx 1-2$ відліки)	Електромагнітні завади, збої АЦП, помилки передачі бітів
Дрейф (Drift / Bias)	Систематичне, поступове зміщення від істинного значення (s_t)	Довготривалий, низькочастотний тренд	Деградація сенсора, температурний вплив, втрата калібрування
Константні значення (Stuck-at Faults)	Нульова дисперсія, значення $z_t = const$ протягом інтервалу	Тривалий, безперервний сегмент	Зависання контролера, обрив ланцюга, програмні помилки ("last value")
Фоновий шум (Background Noise)	Високочастотні коливання малої амплітуди навколо s_t	Постійний, стохастичний процес	Тепловий шум електроніки, межа точності вимірювання
Пропуски даних (Missing Values)	Відсутність значень ($z_t = \emptyset$ або NaN)	Дискретні розриви часового ряду	Втрата пакетів у мережі, перезавантаження пристрою

Ключова обчислювальна та методологічна складність полягає в тому, що всі ці різні типи шумів можуть з'являтися одночасно (накладатися один на одного) або послідовно (чергуватися) в одному й тому ж потоці даних. Більше того, їхні характерні ознаки (амплітуда, тривалість, частота появи, статистичні властивості)

суттєво відрізняються не тільки між типами аномалій, але й для різних типів сенсорів. Наприклад, часовий ряд температури є "повільним", інерційним та відносно гладким, тоді як ряд миттєвого енергоспоживання є "швидким", стрибкоподібним та дуже волатильним. Відповідно, типові пошкодження та методи їх виявлення для них будуть різними.

Саме ця виражена гетерогенність робить більшість існуючих ("класичних" або "універсальних") методів очищення даних неефективними або недостатньо ефективними в даному контексті.

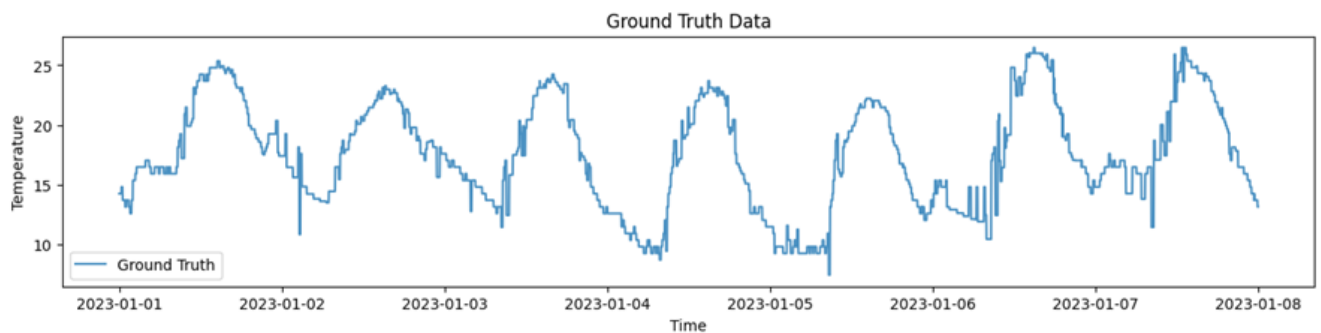


Рисунок 3.2: Візуалізація реального набору зашумлених даних на прикладі часового ряду температури

Традиційні статистичні фільтри, такі як ковзне середнє, медіанний фільтр, фільтр Савіцького-Голя, зазвичай добре справляються лише з якимось одним конкретним типом шуму, наприклад, медіанний фільтр ефективний проти точкових викидів, але при цьому можуть катастрофічно спотворювати сигнал при наявності інших типів, наприклад, той же медіанний фільтр "розміє" легітимні різкі зміни сигналу або буде безсилим проти дрейфу. Вони не адаптуються до зміни типу шуму.

Модельні підходи (наприклад, фільтр Калмана) часто базуються на дуже сильних та нереалістичних в нашій задачі припущеннях про лінійність системи та виключно гаусівський характер шуму як процесу, так і вимірювань. Вони втрачають ефективність при наявності структурованих, нелінійних аномалій (як дрейф чи константні значення).

Сучасні методи машинного навчання (напр., Isolation Forest, One-Class SVM, автоенкодер) хоча є значно більш потужними у виявленні складних залежностей, в переважній більшості вони зосереджені на задачі бінарної класифікації ("норма" vs "аномалія"). Вони можуть успішно виявити факт аномалії, але не розрізняють її тип, що є недоліком, оскільки призводить до застосування єдиної, універсальної стратегії корекції, наприклад, видалення точки та її подальшої імпутації середнім або прогнозним значенням. Така універсальна стратегія може бути неоптимальною або навіть шкідливою для конкретного типу пошкодження (наприклад, прогнозна імпутація під час тривалого періоду константних значень може дати абсолютно нереалістичні фантазійні результати).

Виходячи з вищесказаного, задача ефективного очищення даних від гетерогенних шумів у системах розумного будинку повинна бути формалізована не просто як задача "фільтрації" або "виявлення аномалій", а як розробка комплексного, багатоетапного адаптивного методу, який би функціонував у режимі реального часу (або "майже реального часу") і був здатний виконувати наступну послідовність дій:

1. Аналізувати локальні характеристики вхідного потоку даних Z у ковзному часовому вікні.
2. Ідентифікувати та Класифікувати домінуючий тип компоненти шуму ϵ_t у цьому вікні, наприклад, визначити, чи це стан Outlier, Drift, ConstantValue, GeneralNoise чи Clean стан.
3. Динамічно Обирати (селектувати) та застосовувати найбільш відповідний (релевантний) оператор корекції або фільтрації, який є оптимальним саме для виявленого типу пошкодження, наприклад, застосувати медіанну заміну для Outlier, алгоритм згладжування для GeneralNoise, спеціалізовану імпутацію для ConstantValue і не робити нічого (ігнорувати) для Clean стану).
4. Генерувати на виході очищену послідовність значень $Y = \{y_1, y_2, \dots, y_t, \dots\}$, де кожне y_t є найкращою можливою оцінкою істинного сигналу s_t (тобто $y_t \approx s_t$).

Кінцевою метою є мінімізація сукупної помилки реконструкції (наприклад, RMSE або MAE відносно невідомого сигналу S) та збереження при цьому корисних динамічних характеристик та морфології істинного сигналу (уникаючи надмірного згладжування, спотворення амплітуд або внесення затримок).

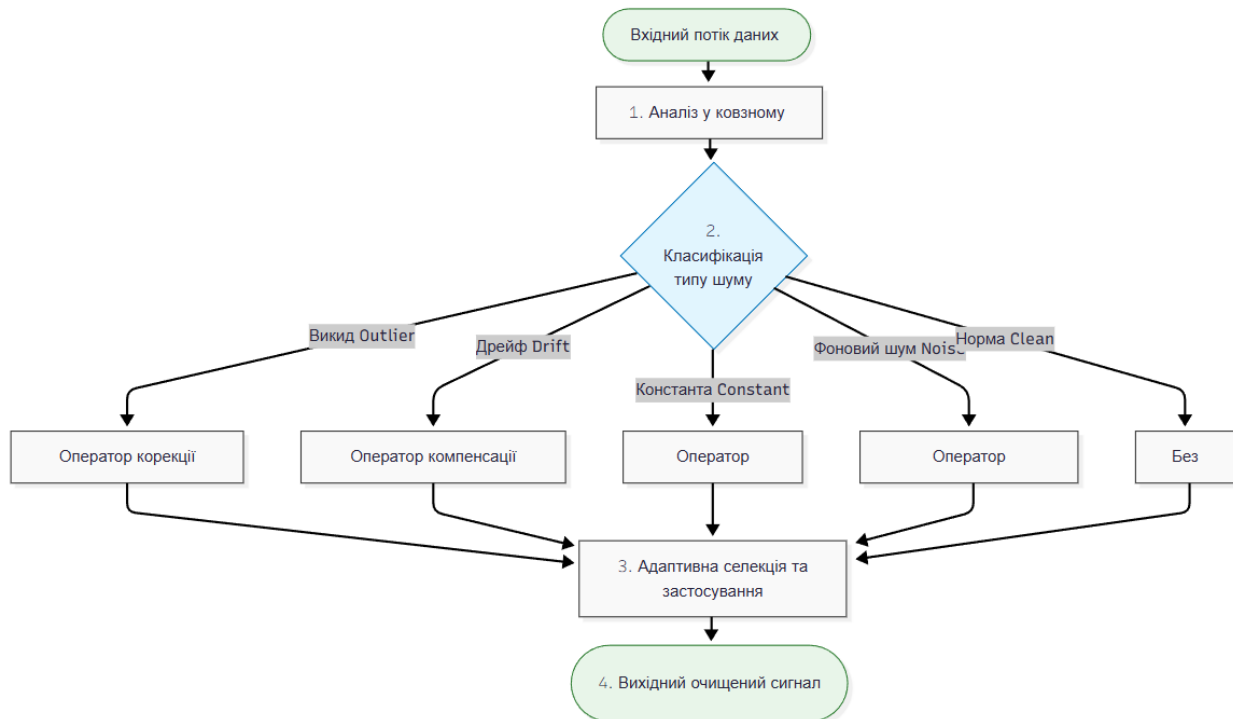


Рисунок 3.3: Концептуальна схема адаптивного методу очищення гетерогенних даних на основі класифікації типу завади.

Саме такий класифікаційно-орієнтований, контекстуально-залежний та адаптивний підхід дозволяє подолати фундаментальні обмеження традиційних та універсальних методів. Він забезпечує стабільно високу якість та надійність вхідних даних, що є обов'язковою передумовою для надійної та ефективної роботи інтелектуальних систем керування сучасним розумним будинком.

3.2. Розробка та алгоритм роботи методу ACRA.

Як було формалізовано в попередньому підрозділі, ключовою проблемою при обробці сенсорних даних у системах розумного будинку є наявність гетерогенних шумів — складної суміші різнорідних аномалій (викидів, дрейфу,

константних значень, фонового шуму, пропущених значень), які можуть з'являтися одночасно або послідовно та мають різні характеристики для різних типів сенсорів. Традиційні методи фільтрації та навіть багато сучасних підходів на основі машинного навчання демонструють обмежену ефективність у таких умовах, оскільки вони або застосовують універсальну стратегію очищення, не враховуючи специфіку аномалії, або не здатні працювати адаптивно в режимі реального часу.

Для вирішення цієї проблеми було розроблено новий метод ACRA (Adaptive Classification-based Real-time Anomaly cleaning). Головна мета ACRA — забезпечити високоякісне очищення одновимірних часових рядів сенсорних даних у режимі реального часу шляхом реалізації адаптивного підходу, керованого класифікацією шумів. Замість застосування єдиного фільтра, ACRA прагне інтелектуально ідентифікувати домінуючий тип шуму, присутній у локальному сегменті даних, і на основі цього діагнозу динамічно обирати та застосовувати найбільш відповідну (оптимальну) стратегію корекції. Це дозволяє ефективно усувати різноманітні аномалії, мінімізуючи при цьому спотворення корисного сигналу та зберігаючи його важливі характеристики.

Архітектура ACRA побудована за модульним принципом та реалізує послідовний (конвеєрний) процес обробки даних, що надходять потоково. Кожне нове значення від сенсора проходить через каскад аналітичних модулів, які приймають рішення про необхідність та спосіб корекції на основі локального контексту, що зберігається у ковзному вікні. Процес є ітераційним і повторюється для кожної точки даних, забезпечуючи функціонування в режимі реального часу. Загальну логіку роботи методу ілюструє блок-схема, представлена на Рисунку 3.4.

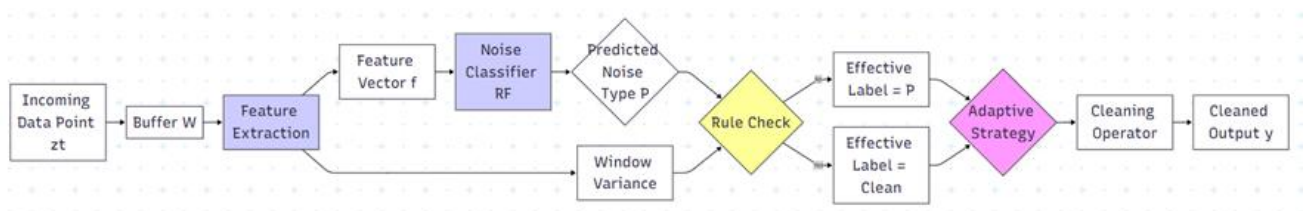


Рисунок 3.4: Блок-схема роботи методу ACRA

Розглянемо детально кожен компонент та етап роботи архітектури ACRA.

Процес починається з обробки вхідних даних та пропущених значень (Input & NaN Handling). Після отримання нового значення z_t від сенсора в момент часу t , першим кроком є важлива перевірка, чи не є отримане значення пропущеним (NaN). Пропущені значення є поширеною проблемою в IoT-системах і можуть виникати через проблеми зі зв'язком, збої сенсора або помилки запису. Їхня некоректна обробка може призвести до помилок у подальших обчисленнях (наприклад, при розрахунку статистичних ознак) або до спотворення результатів очищення.

Якщо z_t є NaN, ACRA негайно виконує імпутацію (заповнення пропуску). Стратегія імпутації полягає у використанні медіанного значення кількох (наприклад, 3-5) останніх, вже очищених значень $y_{\{t-1\}}, y_{\{t-2\}}, \dots$, які зберігаються системою у внутрішньому буфері. Вибір медіани обґрунтований її робастністю до можливих викидів серед останніх очищених значень. Використання саме очищених значень для імпутації (а не сирих z_{t-k}) є важливим, оскільки це дозволяє базувати оцінку пропуску на найнадійніших доступних даних, що відображають поточний рівень істинного сигналу s_t . Отримане числове значення (результат імпутації або оригінальне z_t , якщо воно не було NaN) передається на наступний етап.

Для аналізу даних у контексті їхньої часової динаміки використовується буфер ковзного вікна (Sliding Window Buffer) фіксованого розміру W , що представляє структуру даних (для ефективності часто реалізується за допомогою deque), що зберігає W останніх, вже оброблених на етапі NaN, значень часового ряду. У даному дослідженні розмір вікна W було встановлено на 10 точок, що відповідає 10-хвилинному інтервалу для даних з хвилинною частотою.

При надходженні нового числового значення з попереднього етапу, воно додається в кінець буфера. Якщо буфер вже містив W елементів, найстаріше значення (з початку буфера) видаляється за принципом FIFO (First-In, First-Out). Таким чином, буфер завжди містить актуальний "зріз" локальної історії сигналу. Цей локальний контекст є абсолютно необхідним для наступних етапів:

вилучення статистичних ознак, що характеризують поточну поведінку ряду, та прийняття обґрунтованого рішення щодо очищення останньої точки z_t .

Щоразу, коли буфер ковзного вікна оновлюється, і за умови, що він містить достатню кількість даних (W точок), активується модуль вилучення ознак (Feature Extraction Module). Його завдання — розрахувати вектор f_t статистичних показників, що кількісно описують характеристики даних у поточному вікні. Ці ознаки слугують "описом" локальної поведінки сигналу і є входом для класифікатора шумів. Ідея полягає в тому, що різні типи аномалій (викиди, дрейф, константні значення) по-різному впливають на статистичні властивості даних у вікні, і ці відмінності можуть бути використані для їх розрізнення.

Набір ознак, що розраховуються, є досить широким і включає:

1. Міри центральної тенденції: середнє арифметичне (mean).
2. Міри розкиду: дисперсія (variance), стандартне відхилення (std dev). Ці показники важливі для оцінки загального рівня шуму або стабільності сигналу.
3. Порядкові статистики: мінімальне (min) та максимальне (max) значення, розмах (range = max - min). Вони чутливі до наявності екстремальних значень (викидів).
4. Робастні міри розкиду: інтерквартильний розмах (IQR), медіанне абсолютне відхилення (MAD). Ці статистики є менш чутливими до викидів порівняно зі стандартним відхиленням і тому корисні для характеристики розкиду "основної маси" даних.
5. Відносні міри: коефіцієнт варіації ($CV = \text{std dev} / \text{mean}$). Допомогає оцінити відносний рівень шуму.
6. Міри змін: середня абсолютна різниця між послідовними точками у вікні. Високі значення можуть вказувати на шум або різкі зміни, низькі — на константні значення або плавний дрейф.
7. Індикатори пропущених даних: кількість та частка NaN у вікні (розраховуються до етапу імпутації, якщо потрібно використовувати цю інформацію як ознаку, хоча в поточній реалізації імпутація відбувається раніше).

При розрахунку цих ознак важливо коректно обробляти можливі NaN, що могли залишитися у вікні на початкових етапах заповнення буфера (наприклад, використовуючи функції `nanmean`, `nanmedian`, `nanstd` тощо), щоб забезпечити робастність процесу вилучення ознак.

Розрахований вектор ознак f_t подається на вхід класифікатора типів шуму (Noise Classifier), який є ядром інтелектуального компонента методу ACRA. У даній реалізації в якості класифікатора використовується ансамблевий алгоритм Random Forest (Випадковий ліс) [38].

Обґрунтування вибору Random Forest полягає у наборі причин. Random Forest часто демонструє високу точність класифікації на різноманітних задачах, має робастність до перенавчання, ансамблева природа (усереднення прогнозів багатьох дерев) робить його менш схильним до перенавчання порівняно з окремими деревами рішень. Дерева рішень ефективно моделюють складні взаємодії між ознаками. Можна отримати оцінку важливості кожної ознаки, що допомагає зрозуміти, які характеристики даних є ключовими для розрізнення типів шуму. Сучасні реалізації Random Forest є досить швидкими для багатьох застосувань реального часу. Класифікатор навчається заздалегідь (offline) на розміченому наборі даних. Цей набір створюється шляхом взяття "чистих" часових рядів та синтетичного додавання до них різних типів шумів (Outlier, ConstantValue, Drift, GeneralNoise) з відомими параметрами. Для кожного часового кроку в навчальному наборі розраховується вектор ознак з ковзного вікна, і йому присвоюється мітка відповідного типу шуму.

На етапі роботи ACRA навчений класифікатор приймає вектор ознак f_t і видає прогноз P_t домінуючого типу шуму у поточному вікні. У даній роботі класифікатор навчався розрізняти чотири основні категорії: Outlier, ConstantValue, Drift та GeneralNoise. Важливо підкреслити, що класифікатор не навчається розпізнавати стан "Clean" (відсутність аномалій), оскільки це завдання виявляється надзвичайно складним через плавність переходу між низьким рівнем шуму та його відсутністю.

Для компенсації нездатності класифікатора надійно ідентифікувати "чисті" сегменти даних, архітектура ACRA включає додатковий евристичний модуль — евристичне правило для визначення "чистого" стану (Rule-Based Refinement), що базується на аналізі дисперсії даних у ковзному вікні.

Сегменти даних з дуже низьким рівнем шуму (GeneralNoise) статистично схожі на чисті дані, особливо якщо сигнал у цей період є стабільним. Класифікатор може помилково відносити їх до класу GeneralNoise. Застосування навіть легкого згладжування (передбаченого для GeneralNoise) до таких майже чистих даних є небажаним, оскільки може призвести до непотрібного спотворення сигналу.

Правило спрацьовує після отримання прогнозу P_t від класифікатора. Якщо класифікатор передбачив $P_t = \text{GeneralNoise}$ (іноді використовується спеціальна мітка "PassThrough" для випадків низької впевненості класифікатора), то система додатково обчислює, або використовує вже обчислену на етапі вилучення ознак, дисперсію значень у поточному ковзному вікні W . Ця дисперсія порівнюється з попередньо визначеним пороговим значенням. Поріг встановлюється індивідуально для кожного типу сенсора (температура, вологість тощо) на основі аналізу дисперсії на "чистих" ділянках навчальних даних.

Якщо $\text{variance}(W) < \text{threshold}_{\text{variance}}$, то система робить висновок, що рівень шуму є незначним, і перевизначає мітку для поточної точки: $\text{EffectiveLabel}_t = \text{'Clean'}$.

В іншому випадку (якщо P_t не GeneralNoise, або якщо $P_t = \text{'GeneralNoise'}$, але дисперсія висока), ефективна мітка залишається рівною прогнозу класифікатора: $\text{EffectiveLabel}_t = P_t$.

Гібридний підхід до прийняття рішень (класифікатор + правило) дозволяє використовувати потужність машинного навчання для розпізнавання складних аномалій, одночасно застосовуючи просту та надійну евристику для збереження сигналу в періоди низької зашумленості.

Фінальним етапом обробки точки z_t є застосування адаптивної стратегії очищення, яка керується ефективною міткою EffectiveLabel_t , отриманою на

попередньому кроці. Цей модуль реалізує основний принцип методу ACRA: різним типам шуму — різні методи корекції.

На основі ефективної мітки (або прогнозу класифікатора, або мітки "Clean" від евристичного правила), модуль адаптивної стратегії динамічно обирає та застосовує найбільш доцільний оператор для корекції (або збереження) поточного значення даних z_t . Кожному типу шуму відповідає своя, найбільш ефективна для нього, стратегія очищення:

1. Мітка "Clean" означає, що жодної корекції не відбувається. Оригінальне значення z_t пропускається на вихід без змін (pass-through). Максимальне збереження вихідного сигналу, коли аномалій не виявлено.

2. Мітка "Outlier" застосовується робастна процедура заміни. Поточне значення замінюється медіаною всіх значень у ковзному вікні. Медіана є стійкою до екстремальних значень, тому вона дає надійну оцінку "нормального" рівня сигналу, ігноруючи сам викид у вікні. (У деяких реалізаціях може додатково використовуватися перевірка робастного Z-score перед заміною для підвищення надійності).

3. Мітка "ConstantValue" для значення замінюється медіаною кількох останніх точок у буфері. Припускається, що останні перед "залипанням" значення є більш надійним орієнтиром для істинного сигналу, ніж поточне константне значення. Медіана використовується для стійкості до можливих викидів серед цих останніх точок.

4. Мітка "Drift" означає заміну значення середнім арифметичним кількох останніх точок у буфері. Використання середнього забезпечує плавне згладжування, яке допомагає компенсувати повільний тренд дрейфу, усереднюючи локальні коливання.

5. Мітка "GeneralNoise" встановлюється якщо не було застосовано правило "Clean", тоді застосовується легке згладжування шляхом заміни поточного значення на середнє арифметичне кількох останніх точок, що дозволяє зменшити амплітуду випадкових коливань.

Параметри операторів (кількість точок для розрахунку середнього/медіани) можуть бути налаштовані залежно від характеристик конкретного сенсора та бажаного ступеня згладжування. Результатом роботи обраного оператора є фінальне очищене значення y_t .

Узагальнення описаного алгоритму роботи методу подано на Рисунку 3.4.

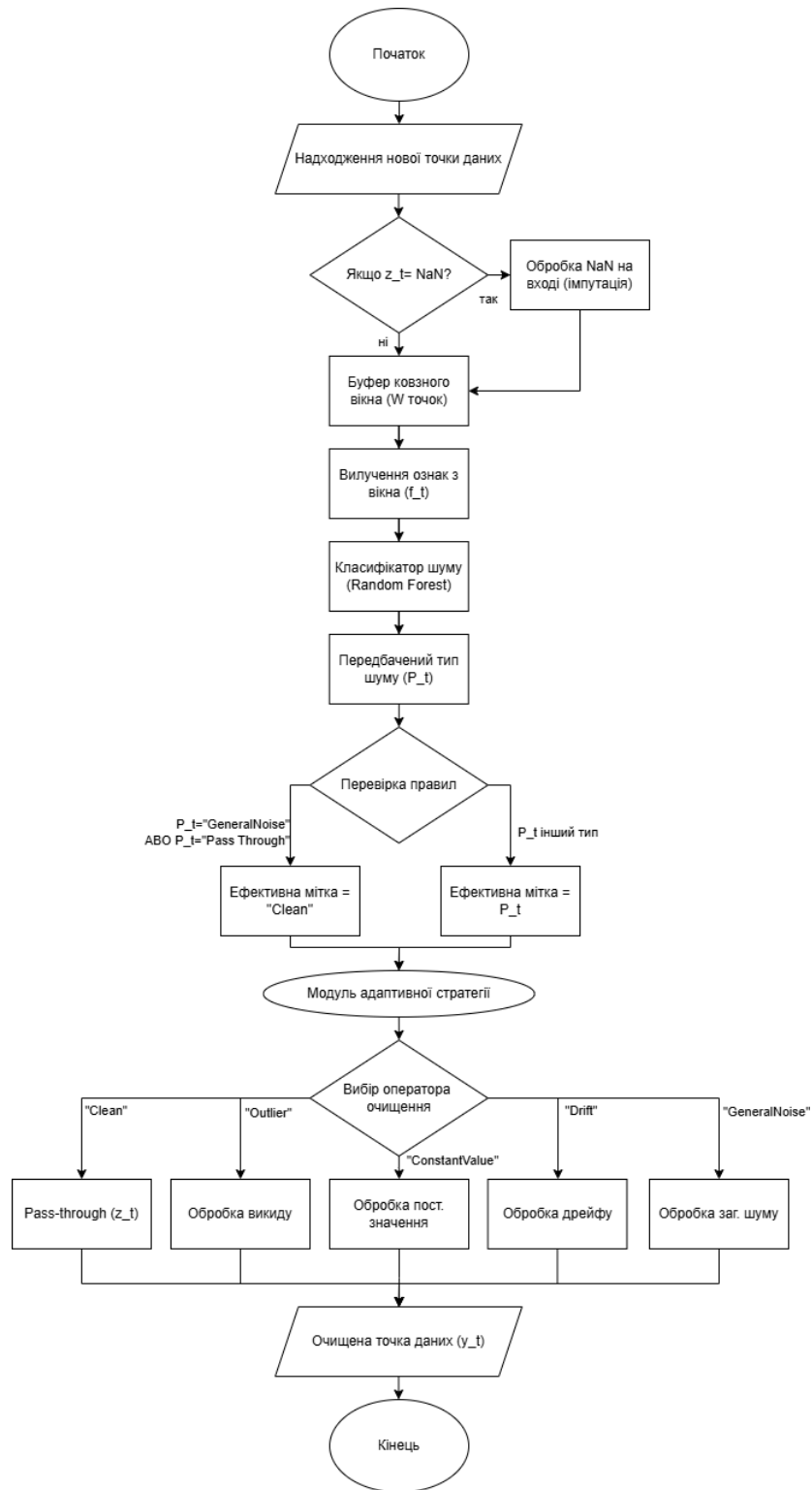


Рисунок 3.5: Алгоритм роботи методу ACRA

Розроблена архітектура методу ACRA представляє собою багатоступеневий, інтелектуальний конвеєр для обробки сенсорних даних. Вона поєднує локальний контекстний аналіз (ковзне вікно), вилучення інформативних ознак, класифікацію типів шумів за допомогою машинного навчання (Random Forest), евристичне уточнення для "чистих" станів та адаптивний вибір найбільш доцільної стратегії корекції. Така комплексна структура дозволяє методу гнучко реагувати на різноманітні та мінливі пошкодження даних, характерні для реальних IoT-систем, забезпечуючи високу якість очищення ($y_t \approx s_t$) при збереженні обчислювальної ефективності, достатньої для роботи в режимі реального часу. Це створює надійну основу для подальшого використання очищених даних у системах аналізу, прогнозування та інтелектуального керування розумним будинком.

3.3. Вдосконалена гібридна архітектура очищення даних HAD-CLEAN на основі глибокого навчання та вентиляної логіки

Розроблений метод ACRA довів життєздатність та принципову ефективність фундаментального підходу "в першу чергу класифікуй, потім очищуй". Його апробація показала переваги над базовими неадаптивними фільтрами. Проте, поглиблений аналіз результатів його роботи (представлений у Розділі 5.3) виявив два ключові напрямки для вдосконалення. По-перше, класифікатор на основі Random Forest та стандартних статистичних ознак демонстрував помірну точність (F1-score $\sim 0.48-0.59$), особливо при розрізненні "чистих" даних (Clean) та "загального шуму" (GeneralNoise). Це призводило до ризику помилкового застосування операторів очищення до валідних даних. По-друге, логіка вибору специфічного оператора корекції (медіана, середнє тощо) для кожного з 4 типів аномалій виявилася складною в налаштуванні та не завжди гарантувала оптимальний результат.

Подальше сегментне експериментальне дослідження (деталізоване в Розділі 5.3) дозволило зробити ключовий висновок: потужні модельні фільтри, зокрема

Фільтр Калмана, є ефективними для згладжування та очищення всіх типів досліджуваних шумів. Однак їхній фундаментальний недолік полягає в неадаптивності. При застосуванні до "чистих" сегментів даних Фільтр Калмана призводить до погіршення якості сигналу, вносячи значну помилку (MAE 0.2013) через надмірне згладжування (oversmoothing).

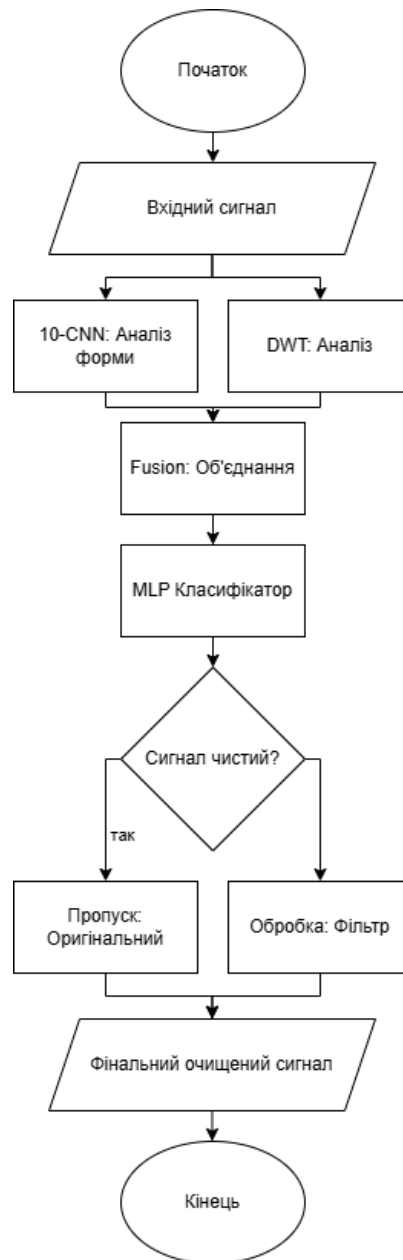


Рисунок 3.6: Блок-схема архітектури H-AD-CLEAN.

Відкриття дозволило кардинально переглянути та оптимізувати стратегію очищення. Задача була переформульована: замість складної логіки вибору найкращого оператора для кожного типу шуму, більш робастним та ефективним рішенням є надійна ідентифікація "чистих" даних та їх захист від втручання.

Для реалізації цього підходу була розроблена вдосконалена гібридна архітектура H-AD-CLEAN (Hybrid Adaptive-Dynamic CLEANing). Вона базується на двох ключових компонентах: (1) високоточному гібридному класифікаторі для надійної діагностики стану сигналу та (2) простій, але ефективній вентильній логіці (Gating Logic), яка використовує прогноз класифікатора для захисту чистих даних.

Загальна архітектура системи, представлена на Рисунку 3.6, функціонує як "розумний диспетчер". Вона приймає на вхід необроблений сигнал у форматі ковзного вікна ($W=60$ точок) і виконує двоетапну обробку: діагностику та керовану фільтрацію.

Таблиця 3.2

Параметри моделі H-AD-CLEAN

Шар(Тип)	Форма виводу	Параметри	Підключення до шару
cnn_input (InputLayer)	(None, 60, 1)	0	-
conv1d (Conv1D)	(None, 60, 32)	256	cnn_input[0][0]
max_pooling1d (MaxPooling1D)	(None, 30, 32)	0	conv1d[0][0]
dropout (Dropout)	(None, 30, 32)	0	max_pooling1d[0][0]
conv1d_1 (Conv1D)	(None, 30, 64)	14,400	dropout[0][0]
max_pooling1d_1 (MaxPooling1D)	(None, 15, 64)	0	conv1d_1[0][0]
dwt_input (InputLayer)	(None, 12)	0	-
dropout_1 (Dropout)	(None, 15, 64)	0	max_pooling1d_1[0][0]
dense (Dense)	(None, 32)	416	dwt_input[0][0]
flatten (Flatten)	(None, 960)	0	dropout_1[0][0]
dropout_2 (Dropout)	(None, 32)	0	dense[0][0]
cnn_features (Dense)	(None, 64)	61,504	flatten[0][0]
dwt_features (Dense)	(None, 32)	1,056	dropout_2[0][0]
fusion_layer (Concatenate)	(None, 96)	0	cnn_features[0][0], dwt_features[0][0]
dense_1 (Dense)	(None, 128)	12,416	fusion_layer[0][0]
dropout_3 (Dropout)	(None, 128)	0	dense_1[0][0]
output_classifier (Dense)	(None, 5)	645	dropout_3[0][0]

Структурно модель складається з двох функціональних блоків: діагностичного модуля та виконавчого модуля.

Діагностичний модуль системи побудований за схемою з двома паралельними потоками обробки, що дозволяє одночасно оцінювати як морфологічні, так і частотні характеристики вхідного сигналу у ковзному вікні.

Перший канал обробки базується на одновимірній згортковій нейронній мережі (1D-CNN). Завдяки здатності автоматично виділяти локальні патерни, цей модуль відповідає за ідентифікацію структурних аномалій, що мають виражену геометричну форму у часовій області: різких імпульсів (Outlier), плато (Constant Value) або лінійних трендів (Drift). Результатом роботи згорткових шарів є формування вектора ознак форми (Feature Vector A).

Другий канал реалізує дискретне вейвлет-перетворення (DWT). Цей метод здійснює багаторівневу декомпозицію сигналу на апроксимуючі (cA) та деталізуючі (cD) коефіцієнти. Такий підхід дозволяє сепарувати високочастотні шуми (GeneralNoise, що проявляється у коефіцієнтах cD_1 , cD_2) від низькочастотних змін сигналу або дрейфу (компоненти cA_3). Статистичні метрики, розраховані на основі вейвлет-коефіцієнтів (дисперсія, ентропія енергії), формують вектор ознак текстури (Feature Vector B).

Порівняльна характеристика потоків обробки наведена в Таблиці 3.3.

Таблиця 3.3.

Функціональний розподіл потоків гібридного класифікатора

Потік обробки	Базовий метод	Тип аналізу	Цільові класи аномалій	Вихідні дані
Stream A	1D-CNN	Морфологічний (часова область)	Outlier, Constant Value, Drift	Вектор ознак форми
Stream B	DWT (Wavelet)	Частотно-часовий (спектральна область)	GeneralNoise, Clean	Вектор ознак текстури

Фінальна класифікація здійснюється шляхом конкатенації (Fusion) отриманих векторів у єдиний дескриптор стану, який подається на вхід багатошарового перцептрону (MLP) з функцією активації Softmax. Це забезпечує прийняття рішення про клас аномалії P з урахуванням як форми, так і частотної

структури сигналу. Схематичне зображення потоків даних у класифікаторі наведено на Рисунку 3.7.

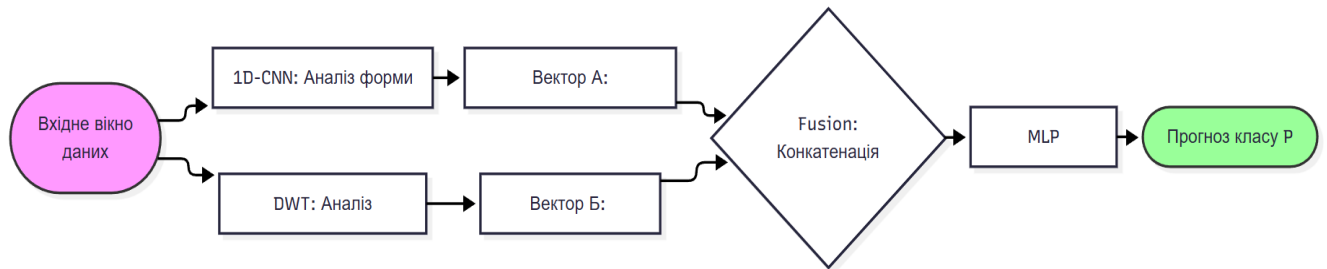


Рисунок 3.7: Архітектура гібридного класифікатора (1D-CNN + DWT)

Виконавчий модуль архітектури H-AD-CLEAN відмовляється від складної системи перемикавання операторів, характерної для попередніх ітерацій (методу ACRA), на користь робастної бінарної логіки («чистити» / «не чистити»). Підхід отримав назву «Вентильна логіка» (Gating Logic).

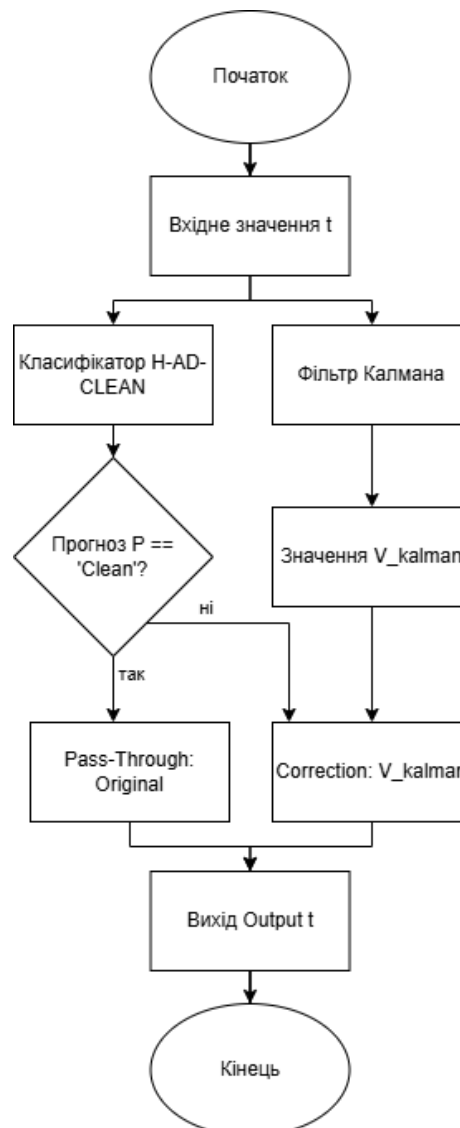


Рисунок 3.8: Схема роботи вентильної логіки (Gating Logic)

Принцип дії виконавчого модуля полягає у паралельному обчисленні потенційно очищеного значення V_{kalman} (за допомогою фільтра Калмана) та отриманні прогнозу стану P від класифікатора. Система функціонує як інтелектуальний диспетчер.

У випадку ідентифікації стану *Clean* ($P = Clean$), клапан блокує роботу фільтра, пропускаючи оригінальний сигнал без змін (Pass-Through) – це критично важливо для збереження автентичності валідних даних.

У випадку детекції будь-якої аномалії ($P \neq Clean$), активується маршрут очищення, і на вихід системи подається значення, скориговане фільтром Калмана (V_{kalman}).

Алгоритм прийняття рішень формалізовано на Рисунку 3.8.

Описана гібридна конструкція дозволяє вирішити проблему «переочищення» (over-smoothing), застосовуючи потужні математичні інструменти корекції виключно до пошкоджених сегментів даних, при цьому демонструючи високу стійкість до хибних спрацювань (F1-score 0.85 для класу *Clean*).

3.4. Алгоритм роботи методу H-AD-CLEAN у режимі реального часу

Алгоритм роботи вдосконаленого методу H-AD-CLEAN реалізує архітектурні рішення, детально описані у підрозділі 3.4, та призначений для послідовної, потокової (streaming) обробки одновимірних часових рядів сенсорних даних. Функціонування системи забезпечується шляхом ітераційного виконання циклу діагностики та керованої фільтрації для кожного нового спостереження z_t , що надходить від сенсора в момент часу t . Це забезпечує відповідність вимогам функціонування у режимі реального часу.

Перед початком обробки потоку даних виконується етап ініціалізації системи. На цьому етапі завантажуються попередньо навчені ваги компонентів гібридного класифікатора (Компонент 1), що включає підмережу 1D-CNN, параметри DWT та екстракторів статистичних ознак, а також ваги фінального MLP-класифікатора 1. Одночасно ініціалізується модель Фільтра Калмана

(Компонент 2) з початковими параметрами стану та коваріації. Також створюється порожній буфер ковзного вікна W фіксованого розміру (у даному дослідженні $W=60$ точок 2).

Основний цикл обробки активується при надходженні кожного нового спостереження z_t . Дане значення додається до буфера W , витісняючи найстаріше значення за принципом FIFO (First-In, First-Out). Таким чином, буфер W завжди містить 60 останніх часових точок, що формують локальний контекст (вікно) для аналізу. У випадку надходження пропущеного значення (NaN), може бути застосована попередня процедура імпутації (аналогічно до методу ACRA), або ж вікно може оброблятися з урахуванням пропусків, якщо це підтримується екстракторами ознак.

Після оновлення буфера W (і за умови його повного заповнення), система виконує два ключові процеси, які концептуально можуть розглядатися як паралельні. Перший процес – діагностика (Компонент 1). Вміст вікна W одночасно подається на два потоки гібридного класифікатора: потік аналізу форми (1D-CNN) та потік аналізу текстур (DWT). Перший потік генерує Вектор Ознак А (форма), другий – Вектор Ознак Б (текстура). обидва вектори ознак одразу конкатенуються і подаються на вхід MLP-класифікатора, який миттєво повертає фінальний категоріальний прогноз (мітку) стану сигналу P_t (наприклад, Clean, Noise, Drift, Constant, Outlier).

Другий процес – превентивна фільтрація (Компонент 2). Незалежно від роботи класифікатора, Фільтр Калмана (KF) обробляє поточне значення z_t , використовуючи свій попередній стан, та генерує "кандидатне" очищене значення V_{kalman} . Цей крок виконується завжди, щоб очищене значення було готове до використання у випадку, якщо класифікатор ідентифікує шум.

Ключовим етапом алгоритму є прийняття рішення на основі вентильної логіки (Gating Logic). Цей модуль функціонує як "розумний диспетчер" або "вентиль", який керує вибором фінального вихідного сигналу y_t . Логіка є бінарною, робастною та базується виключно на прогнозі P_t , отриманому від гібридного класифікатора:

Якщо прогноз класифікатора $P_t = 'Clean'$, система вважає поточне значення валідним і таким, що не потребує втручання. Вентиль спрацьовує на "захист". Обирається Оператор 1 Pass-Through (Пропустити). На вихід y_t подається оригінальне, необроблене вхідне значення ($y_t = z_t$).

Якщо прогноз класифікатора P_t є будь-яким типом шуму (наприклад, Outlier, Drift, GeneralNoise або Constant), система вважає поточне значення пошкодженим. Вентиль спрацьовує на "очищення". Обирається Оператор 2 Kalman Filter. На вихід y_t подається попередньо розраховане "кандидатне" значення, отримане від Фільтра Калмана ($y_t = V_{kalman}$).

Після вибору оператора та формування фінального очищеного значення y_t , воно передається на вихід системи для подальшого використання іншими модулями розумного будинку, наприклад, модулем прогнозування, а також для збереження у базі даних. Одночасно стан Фільтра Калмана оновлюється для підготовки до наступного кроку. Після цього система переходить у режим очікування наступного спостереження z_{t+1} , і весь описаний цикл обробки (оновлення вікна, діагностика, фільтрація, вентильне рішення) повторюється.

Отже, запропонований алгоритм є циклічним, послідовним та адаптивним. Він реалізує архітектуру H-AD-CLEAN в режимі реального часу, забезпечуючи інтелектуальний аналіз кожної точки даних у її локальному темпоральному контексті. Ключовою перевагою є те, що обчислювально складний процес класифікації використовується не для вибору з багатьох складних операторів, а для простого, але критично важливого бінарного рішення: втручатися чи не втручатися. Це дозволяє ефективно використовувати потужний Фільтр Калмана для боротьби з шумами, одночасно надійно захищаючи валідні дані від спотворення, що було фундаментальною проблемою неадаптивних методів.

Ефективність даного алгоритму, зокрема точність роботи гібридного класифікатора та фінальна якість очищення сигналу в порівнянні з базовими методами та попередньою версією (ACRA), буде детально проаналізована та кількісно оцінена в рамках експериментального дослідження у Розділі 5.

Висновки до розділу 3

У даному розділі було вирішено науково-практичне завдання, пов'язане з низькою якістю сенсорних даних у системах розумного будинку. На основі проведеного аналізу та розробки зроблено наступні висновки:

Формалізовано проблему гетерогенних шумів. Проблема якості даних у системах IoT була представлена не як наявність випадкового шуму, а як складна суміш різнорідних (гетерогенних) аномалій, включаючи точкові викиди, дрейф, періоди константних значень та фоновий шум. Доведено, що існуючі методи є неефективними саме через нездатність адаптивно реагувати на таку різноманітність пошкоджень.

Розроблено архітектуру методу ACRA. Для вирішення поставленої проблеми запропоновано новий метод адаптивного очищення даних ACRA (Adaptive Classification-based Real-time Anomaly cleaning). Його архітектура є модульною і поєднує сильні сторони машинного навчання та евристичних правил. Ключовими компонентами є модуль вилучення статистичних ознак з ковзного вікна, класифікатор типів шуму на основі Random Forest та модуль адаптивного вибору операторів очищення.

Інтегровано гібридний підхід до прийняття рішень. Важливою особливістю архітектури є поєднання прогнозу класифікатора з евристичним правилом на основі аналізу дисперсії. Такий гібридний підхід дозволяє значно точніше ідентифікувати періоди з низьким рівнем шуму ("чисті" стани), запобігаючи надмірному згладжуванню та спотворенню корисного сигналу.

Сформульовано алгоритм роботи методу в режимі реального часу. Розроблено покроковий алгоритм, що реалізує потокову обробку даних. Алгоритм включає етапи оновлення буфера, вилучення ознак, класифікації, застосування правила та адаптивного вибору оператора корекції для кожної точки даних, що надходить. Це підтверджує відповідність методу вимогам до роботи в системах реального часу.

Таким чином, у розділі представлено завершене архітектурне та алгоритмічне рішення для задачі адаптивного очищення різнорідних сенсорних

даних. Розроблений метод ACRA є гнучким та інтелектуальним інструментом, що створює надійну інформаційну основу для подальшого аналізу, прогнозування та прийняття рішень у системах керування розумним будинком. Експериментальна перевірка його ефективності буде проведена у розділі 5.

РОЗДІЛ 4. ПРОАКТИВНА АРХІТЕКТУРА КЕРУВАННЯ НА ОСНОВІ КОНТЕКСТУАЛЬНИХ НАМІРІВ

4.1. Теоретичні основи проактивного керування на основі контекстуальних намірів

Еволюція технологій розумного будинку (Smart Home) характеризується стійким вектором розвитку, спрямованим від простої автоматизації окремих підсистем до інтеграції по-справжньому інтелектуальних, автономних та людино-орієнтованих (Human-Centric) просторів. Незважаючи на значний технологічний прогрес, фундаментальним обмеженням традиційних систем керування залишається їхня базова парадигма взаємодії. Вона здебільшого базується на імперативному підході, що вимагає від кінцевого користувача мислити мовою машини — тобто формулювати свої потреби через жорсткі, детерміновані правила формату "Якщо <умова> То <дія>" (IF-THEN).

Такий підхід, хоч і є концептуально простим та відносно легким для реалізації базових сценаріїв, неминуче створює значний когнітивний розрив. Цей розрив виникає між природним, цілісним та часто абстрактним бажанням людини (наприклад, "підтримувати затишок" або "не витратити зайвого") та тією формальною, низькорівневою мовою логічної автоматизації, яка є необхідною для його машинного виконання. Як наслідок, для реалізації відносно простого, але багатокритеріального бажання, такого як "підтримувати комфортний мікроклімат, але активно економити електроенергію вдень, коли нікого немає вдома", користувачу доводиться вручну створювати, тестувати та підтримувати десятки складних, часто взаємопов'язаних правил. Правила повинні враховувати час доби, поточні та прогнозовані показники сенсорів, динамічні тарифи на енергоносії та інші умови, що є не тільки надзвичайно обтяжливим завданням, що вимагає від користувача технічних навичок та значних витрат часу, але й породжує систему, що є принципово негнучкою та нестабільною. Статичні, запрограмовані правила

нездатні адекватно адаптуватися ані до високо динамічної природи реального світу (непередбачувані зміни погоди, раптова зміна планів мешканців), ані до мінливих вподобань самих користувачів.

Для подолання цих фундаментальних обмежень в академічному та науково-дослідницькому середовищі активно формується та розвивається нова парадигма – керування на основі високорівневих намірів користувача (Intent-Based Control). Суть цієї концепції полягає в радикальній зміні моделі взаємодії: користувач переходить від імперативного підходу ("ЯК" робити) до декларативного ("ЩО" зробити). Замість програмування детальних інструкцій, користувач виражає свої вподобання у вигляді абстрактних, інтуїтивно зрозумілих цілей, таких як "максимальний комфорт", "пріоритет жорсткої економії" або, що найбільш реалістично для більшості сценаріїв, "пошук гнучкого балансу між комфортом та економією".

В рамках цієї парадигми, все обчислювальне навантаження з інтерпретації та реалізації переноситься на інтелектуальну систему. Завдання системи полягає в тому, щоб самостійно інтерпретувати ці високорівневі наміри та автономно трансформувати (або декомпозувати) їх у послідовність оптимальних, низькорівневих керуючих дій для виконавчих пристроїв (актуаторів). Такий декларативний підхід дозволяє створювати гнучкі, по-справжньому людино-орієнтовані системи, що здатні самостійно та безперервно адаптуватися до унікальної поведінки та звичок мешканців, при цьому мінімізуючи їхнє когнітивне навантаження та зводячи до мінімуму потребу в постійному ручному втручанні.

Практична реалізація цієї концепції є складною науково-технічною задачею та вимагає від архітектури системи наявності та тісної синергетичної інтеграції трьох ключових інтелектуальних спроможностей: глибокого моделювання вподобань користувача, розвиненої контекстуальної обізнаності та ефективних механізмів багатоцільової оптимізації.

Узагальнене порівняння зазначених підходів наведено в Таблиці 4.1, яка демонструє відмінності за основними критеріями та підкреслює переваги декларативної моделі в запропонованій системі.

Таблиця 4.1.

Порівняльна характеристика імперативного та декларативного підходів до керування розумним будинком

Критерій порівняння	Імперативний підхід (Rule-Based)	Декларативний підхід (Intent-Based)
Парадигма взаємодії	«ЯК робити» (Інструкція)	«ЩО зробити» (Намір/Ціль)
Формат правил	Жорсткі конструкції «Якщо-То» (IF-THEN)	Функції корисності та цільові метрики
Роль користувача	Програміст сценаріїв (високе когнітивне навантаження)	Джерело вподобань (низьке когнітивне навантаження)
Адаптивність	Низька (статичні правила)	Висока (динамічна оптимізація)
Механізм вирішення конфліктів	Ручне налаштування пріоритетів правил	Автоматичний пошук компромісу (Trade-off)
Масштабованість	Складність зростає експоненційно з кількістю пристроїв	Масштабується за рахунок абстракції цілей

Першочерговою умовою для функціонування системи на основі намірів є її здатність формалізувати та зрозуміти, що саме абстрактні поняття "комфорт" чи "економія" означають для конкретного користувача. Вподобання за своєю природою є глибоко суб'єктивними, індивідуальними та, що важливо, часто неявними – користувач не завжди може чітко артикулювати власні бажання у вигляді конкретних числових параметрів. Сучасні дослідження виділяють два основні, часто комплементарні, підходи до побудови моделі вподобань.

Перший – це явне моделювання. В цьому випадку користувач безпосередньо та свідомо налаштовує параметри системи, наприклад, встановлює бажаний діапазон температур, цільовий рівень освітлення чи графік роботи пристроїв. Цей підхід є необхідним для початкової ініціалізації системи ("холодного старту"), проте він залишається статичним та негнучким, оскільки не враховує динаміку та еволюцію вподобань з часом.

Другий, значно більш потужний підхід – це неявне навчання. Система самостійно та проактивно навчається вподобанням, пасивно аналізуючи поведінку користувача "в контурі". Найціннішим та найбільш достовірним джерелом інформації в цьому процесі є випадки ручного втручання. Момент, коли користувач скасовує автоматичне рішення системи, наприклад, вручну вмикає обігрівач, який система вимкнула задля економії, розглядається як потужний, хоч і неявний, зворотний зв'язок. Втручання користувача є чітким сигналом про невідповідність поточної стратегії системи справжнім намірам користувача в даний момент. Аналізуючи контекст, в якому відбуваються такі втручання, система може динамічно коригувати свою внутрішню модель вподобань.

Людські наміри та пріоритети рідко бувають статичними чи абсолютними; вони є високо динамічними та сильно залежать від поточного контексту – унікальної сукупності обставин у певний момент часу. Наприклад, намір "максимальна економія" може бути домінуючим пріоритетом у денні години робочого дня, під час дії дорогого "пікового" тарифу та за відсутності мешканців удома. Однак, увечері, під час відпочинку родини, пріоритетом безумовно стає "комфорт", і користувач готовий до більших витрат задля його досягнення.

Таким чином, інтелектуальна система повинна бути не просто адаптивною, а глибоко контекстуально-чутливою. Її завдання полягає не в тому, щоб навчитися єдиному глобальному чи усередненому набору вподобань, а в тому, щоб побудувати складну динамічну модель, що відображає пріоритети користувача залежно від широкого спектру контекстуальних факторів. До таких факторів належать час доби та день тижня, пора року та поточні/прогнозовані погодні умови, динамічні тарифи на електроенергію, дані про присутність та про тип діяльності мешканців у будинку.

Здатність системи навчатися та моделювати саме контекстуальні наміри є ключовою відмінністю просунутих проактивних архітектур від простих адаптивних систем.

Зведена класифікація факторів, котрі впливають на наміри користувача наведені у Таблиці 4.2.

Таблиця 4.2

Класифікація контекстуальних факторів впливу на формування ситуативних
намірів користувача

Категорія контексту	Ключові параметри (змінні стану)	Характер впливу на функцію корисності
Часові параметри	Час доби, день тижня, сезонність	Визначають циклічні патерни поведінки та базові очікування комфорту
Зовнішнє середовище	Температура, вологість, інсоляція (поточна/прогноз)	Коригують енерговитрати, необхідні для досягнення цільового комфорту
Економічні чинники	Динамічні тарифи на електроенергію, обмеження потужності	Змінюють вагові коефіцієнти критерію «Економія» (w_{cost})
Стан мешканців	Присутність, тип активності (сон, робота, відпочинок)	Змінюють вагові коефіцієнти критерію «Комфорт» ($w_{comfort}$)

Навіть у межах одного визначеного контексту, наміри користувача часто залишаються внутрішньо суперечливими. Завжди існує фундаментальний конфлікт або компроміс між конкуруючими цілями. Прагнення до максимального комфорту (наприклад, підтримка ідеально стабільної температури 22°C) прямо конфліктує з прагненням до максимальної економії, що вимагає мінімізації часу роботи обігрівача. Завдання системи полягає не у виборі однієї з крайніх точок ("все або нічого"), а в безперервному пошуку оптимального компромісу між ними, який би найкращим чином відповідав поточним пріоритетам користувача.

Для формального вирішення цієї задачі використовуються методи багатоцільової оптимізації. Система на кожному кроці прийняття рішення повинна оцінювати всі потенційні керуючі дії з точки зору їхніх прогнозованих наслідків для кожної з цілей. Найбільш поширеним інструментом для такої оцінки є формалізована функція корисності. Математично, вона часто реалізується як зважена сума нормалізованих оцінок (балів) для кожної цілі, наприклад, оцінка рівня комфорту та оцінка енергоефективності/вартості. Ключовий аспект полягає в тому, що вагові коефіцієнти, наприклад, $w_{comfort}$ та w_{energy} , в цій функції не є

статичними. Вони динамічно оновлюються та завантажуються з моделі вподобань на основі вивчених контекстуальних намірів. Таким чином, система обирає ту дію, яка максимізує сукупну зважену корисність, знаходячи найкращий баланс між суперечливими цілями в даному конкретному контексті.

Отже, концепція керування на основі високорівневих намірів є парадигматичним зсувом у проєктуванні систем керування будівлями. Вона знаменує перехід від жорсткого, імперативного програмування статичних правил до гнучкої, автономної, декларативної оптимізації, керованої цілями. Такий підхід не лише знімає з користувача когнітивне навантаження, пов'язане зі складним низькорівневим налаштуванням, але й дозволяє створювати системи, що демонструють значно вищий рівень адаптивності та персоналізації.

Реалізація такого підходу дозволяє створювати системи, які не просто сліпо виконують команди, а здатні "розуміти" та навіть передбачати неявні бажання користувача. Вони самостійно адаптуються для досягнення найкращого компромісного результату в умовах складного та динамічного середовища розумного будинку. Для практичної реалізації описаної концепції необхідно розробити відповідну модульну архітектуру, яка б забезпечувала ефективну взаємодію між модулями прогнозування, оптимізації та адаптивного навчання на основі зворотного зв'язку від користувача.

4.2. Розробка компонентної архітектури системи проактивного керування.

Для практичної реалізації та подальшої експериментальної валідації концепції керування на основі намірів, яка була теоретично обґрунтована в попередньому підрозділі, було спроектовано та детально опрацьовано модульну, масштабовану та інтелектуальну архітектуру системи проактивного керування. Її фундаментальний дизайн та компонування спрямовані на те, щоб ефективно та синергетично поєднати три ключові технологічні стовпи: високоточне короткострокове прогнозування, динамічну багатоцільову оптимізацію та

безперервне адаптивне навчання. Кінцевою метою такої синергії є створення повністю автономної, ресурсоефективної та, що найважливіше, людино-орієнтованої системи керування.

В основі запропонованої архітектури лежить принцип розмежування відповідальності (Separation of Concerns). Система декомпована на три логічно та функціонально відокремлені рівні, що взаємодіють через стандартизовані інтерфейси:

1. Рівень взаємодії з середовищем (Environment & Interaction Layer);
2. Обчислювальне ядро проактивного керування (Proactive Control Core);
3. Рівень управління знаннями та адаптації (Knowledge & Adaptation Layer).

Детальну компонентну діаграму архітектури, що ілюструє взаємозв'язки між цими рівнями та їхніми внутрішніми модулями, наведено на Рисунку 4.1.

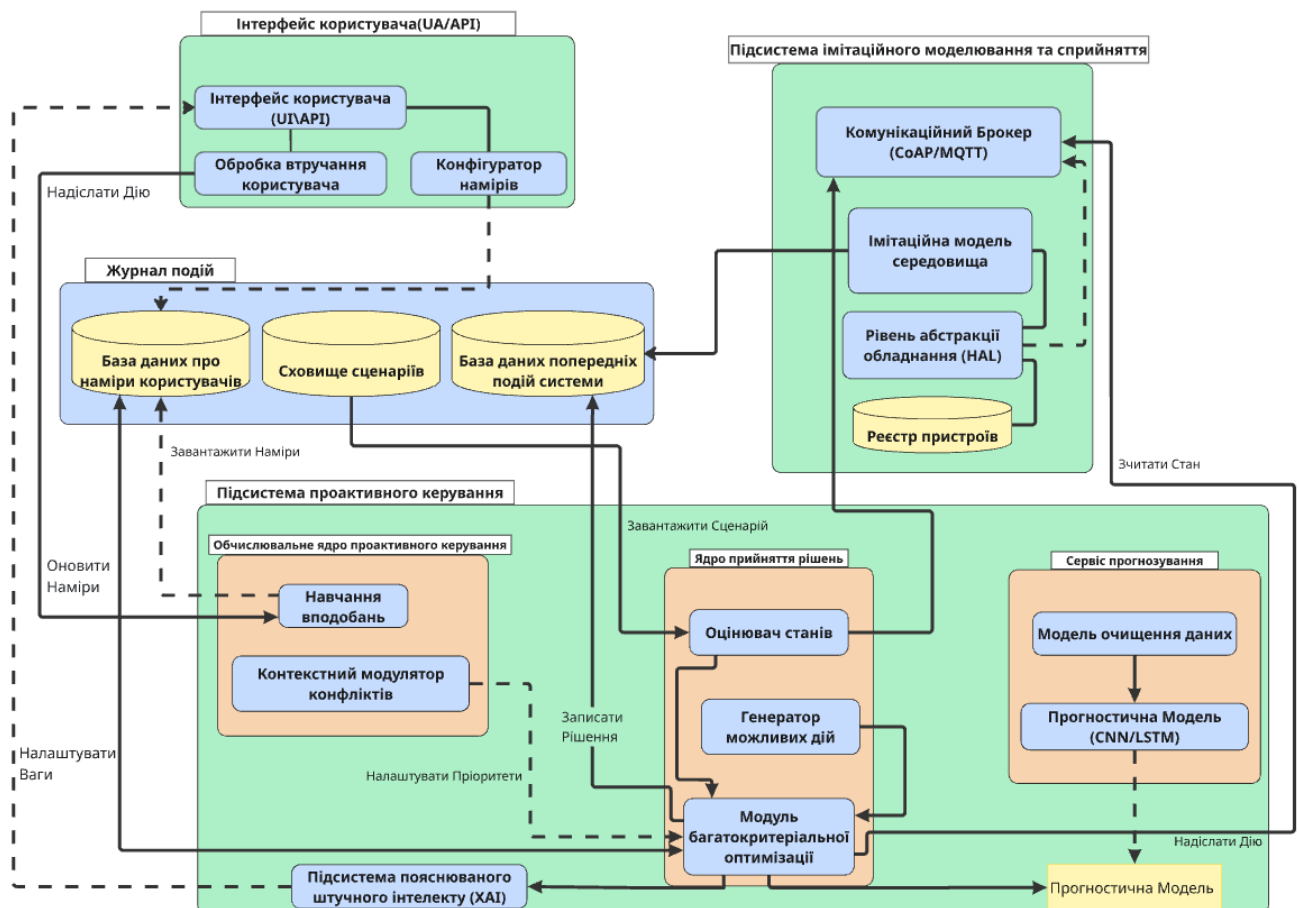


Рисунок 4.1: Компонентна діаграма архітектури системи проактивного керування

Рівень взаємодії з середовищем виконує функцію рівня апаратної абстракції (Hardware Abstraction Layer — HAL). Він виступає двоспрямованим посередником між цифровим обчислювальним ядром та фізичним або, в межах експерименту, імітованим середовищем розумного будинку.

Фундаментальне завдання цього рівня полягає в інкапсуляції низькорівневої складності та гетерогенності фізичного обладнання. Він абстрагує ядро системи від специфіки конкретних моделей сенсорів, типів виконавчих механізмів та особливостей комунікаційних протоколів (таких як Zigbee, MQTT, Z-Wave).

Замість прямої взаємодії з різнорідними пристроями, у висхідному напрямку рівень забезпечує агрегацію та попередню обробку (очищення, як описано в Розділі 3) даних від сенсорів, передаючи до обчислювального ядра стандартизований вектор стану середовища.

У низхідному напрямку рівень взаємодії з середовищем забезпечує отримання уніфікованих команд керування, наприклад, цільового значення потужності обігріву, та їх трансляцію у специфічні сигнали, зрозумілі для конкретних виконавчих механізмів.

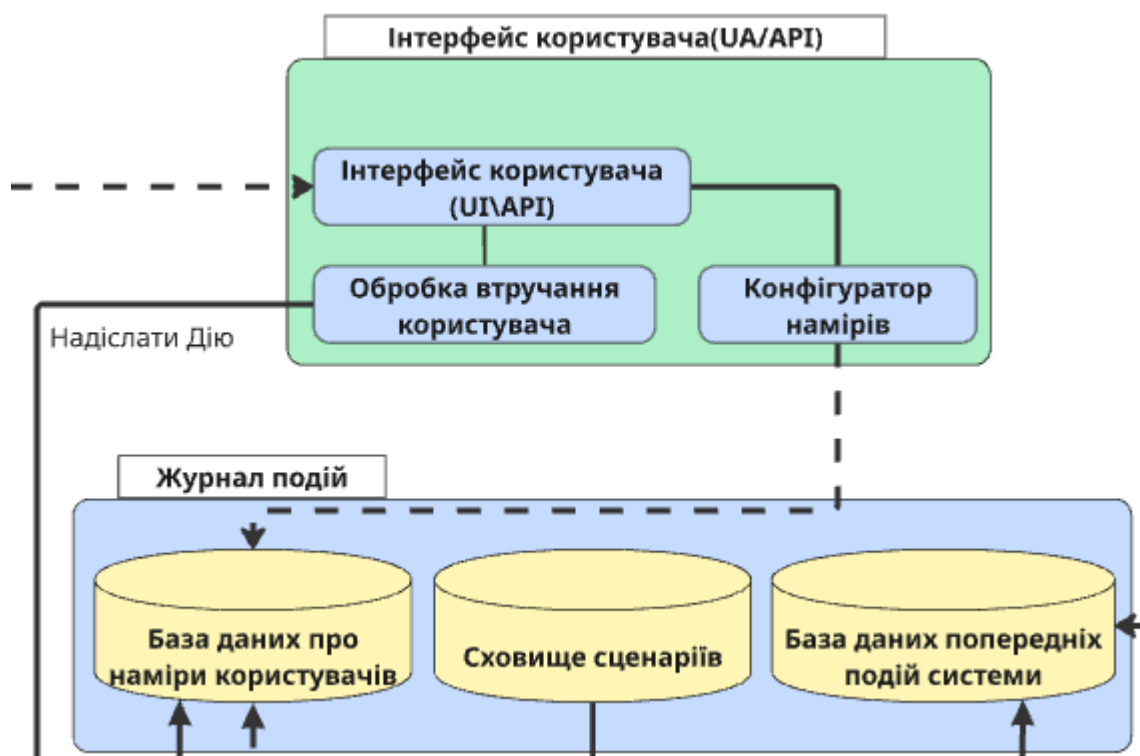


Рисунок 4.2: Структура рівня взаємодії з середовищем

У межах дисертаційного дослідження ключовим елементом цього рівня є програмна імітаційна модель середовища (далі — імітатор). Такий вибір є методологічно обґрунтованим, оскільки проведення натурних експериментів у фізичному середовищі характеризується значною тривалістю, високою вартістю, складністю контролю параметрів та низькою відтворюваністю.

Натомість використання імітатора забезпечує контрольоване середовище для проведення серії відтворюваних експериментів у прискореному масштабі часу, що є необхідною умовою для навчання, валідації та порівняльного аналізу агентів керування.

Основним завданням імітатора є математичне моделювання динаміки ключових фізичних процесів об'єкта керування, що передбачає, зокрема, реалізацію термодинамічної моделі, яка описує еволюцію внутрішньої температури приміщення під впливом теплових потоків від виконавчих механізмів (обігрівачів) та тепловтрат через огорожувальні конструкції, інтенсивність яких залежить від зовнішніх умов.

Окрім моделювання внутрішньої фізики, імітатор забезпечує генерацію або відтворення (на основі історичних даних) реалістичних послідовностей зовнішніх впливів. До них належать стохастичні та детерміновані процеси, що не підлягають контролю з боку системи керування: добові коливання зовнішньої температури, динаміка тарифів на електроенергію (зональні тарифи, тарифи реального часу тощо). Ці чинники розглядаються як вхідні збурення для фізичної моделі та, водночас, як екзогенні змінні для прогностичних та оптимізаційних модулів обчислювального ядра.

Важливо зазначити, що імітаційна модель є реалізацією рівня взаємодії виключно для цілей наукового дослідження. У промисловій реалізації системи функції цього компонента виконуватиме відповідний програмно-апаратний комплекс. Він включатиме драйвери пристроїв, що забезпечують комунікацію з сенсорами та виконавчими механізмами через специфічні промислові (Modbus, BACnet, KNX) або побутові протоколи, а також шлюзи Інтернету речей (IoT Gateways).

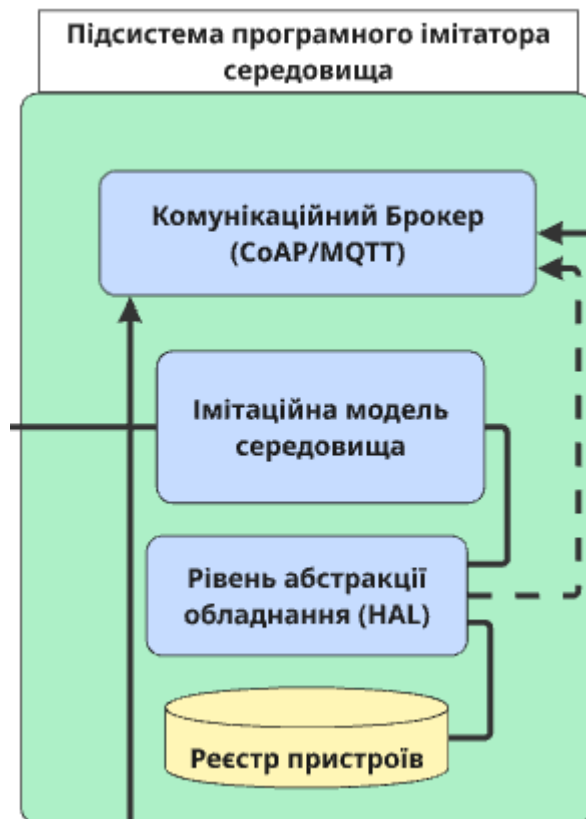


Рисунок 4.3: Схема програмного імітатора середовища

Завданням шлюзів є агрегація даних з гетерогенних бездротових мереж (Wi-Fi, Zigbee, Z-Wave, LoRaWAN) та надання уніфікованого високорівневого інтерфейсу (API), наприклад, через протокол MQTT, для обчислювального ядра. При цьому архітектурна роль рівня як абстрактного посередника залишається незмінною.

Обчислювальне ядро проактивного керування — це центральний компонент системи, відповідальний за аналіз даних, прогнозування та прийняття оптимальних рішень. Його функціонування базується на безперервному ітеративному циклі "генерація — прогнозування — оцінювання", що дозволяє реалізувати стратегію випереджувального впливу замість реактивної реакції на події.

Обчислювальне ядро структурно об'єднує три функціональні модулі: модуль генерації рішень, прогностичну модель та модуль багатокритеріальної оптимізації.

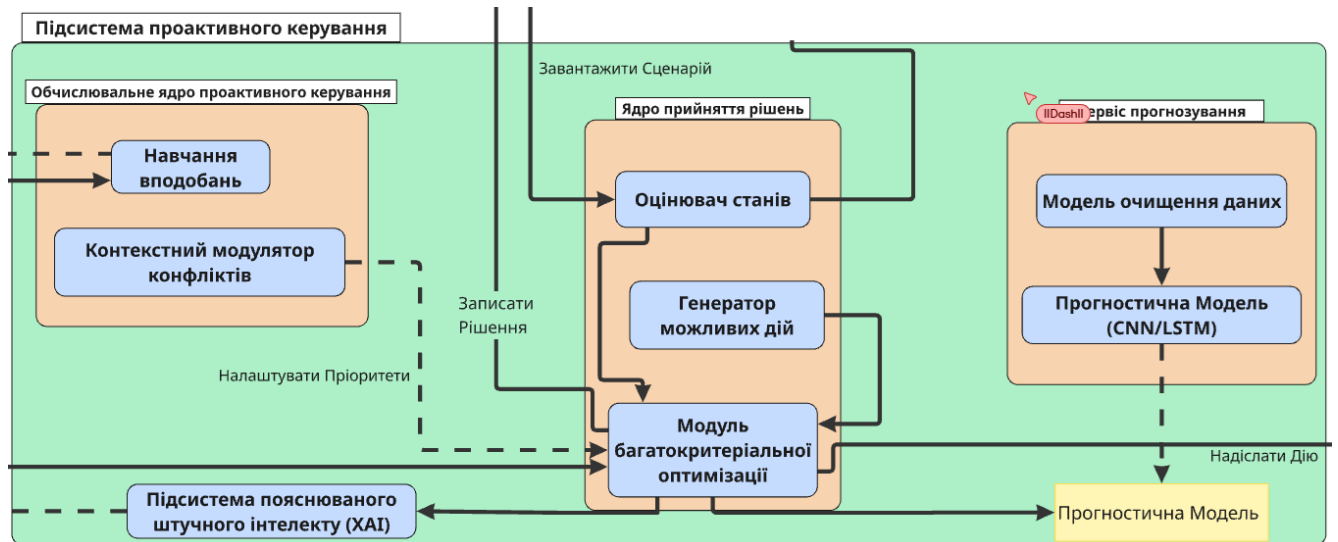


Рисунок 4.4: Структура обчислювального ядра проактивного керування.

Отримавши вектор поточного стану середовища, модуль генерації рішень формує простір допустимих (потенційних) керуючих впливів на наступний дискретний часовий інтервал. Наприклад, для підсистеми опалення це може бути набір дискретних станів: "вимкнути обігрів", "встановити 50% потужності", "встановити 100% потужності".

Для кожного сформованого варіанту керування прогностична модель виконує оцінку майбутнього стану середовища, який настане внаслідок реалізації цього впливу. Наприклад, модель розраховує прогностичне значення температури у приміщенні через 15 хвилин за умови роботи обігрівача на повну потужність. У межах даного дослідження для валідації ефективності механізму прийняття рішень використовується ідеалізована аналітична модель ("біла скринька"). Однак, архітектура дозволяє імплементацію моделей на основі машинного навчання (наприклад, нейронних мереж LSTM) для реальних експлуатаційних умов.

Модуль багатокритеріальної оптимізації реалізує стратегію керування на основі намірів, тобто оцінює кожен прогнозований стан за допомогою формалізованої функції корисності. Функція базується на поточних контекстуальних намірах користувача, виражених через вагові коефіцієнти

пріоритеті, що завантажуються з Баз знань. Проаналізувавши всі альтернативи та їхні наслідки, оптимізатор обирає керуючий вплив, що максимізує функцію корисності, та передає його на виконання [6].

Рівень управління знаннями та адаптації забезпечує здатність системи до навчання та еволюційного розвитку. Він складається з двох ключових компонентів.

База знань – сховище структурованої інформації, де містяться контекстуальні наміри користувача. Вони представлені у вигляді векторів вагових коефіцієнтів для різних ситуативних контекстів, наприклад, "вечірній час / високий тариф" або "денний час / низький тариф".

Модуль адаптивного навчання – елемент, що забезпечує людино-орієнтованість системи. Модуль безперервно аналізує поведінку користувача, зокрема випадки ручної корекції керування коли користувач скасовує автоматичне рішення. Такі втручання інтерпретуються як неявний зворотний зв'язок, що свідчить про розбіжність поточної стратегії системи з реальними потребами користувача. На основі цього модуль коригує вагові коефіцієнти у Базі знань для відповідного контексту, дозволяючи системі динамічно адаптуватися до прихованих вподобань мешканців та мінімізувати необхідність майбутніх втручань.

Робочий цикл системи ініціюється значущою подією зміни стану середовища (наприклад, початком нового часового слоту або відхиленням температури понад порогове значення):

1. Рівень взаємодії реєструє новий стан середовища і передає вектор стану до обчислювального ядра.
2. Ядро звертається до Баз знань та отримує актуальні для поточного контексту наміри (вагові коефіцієнти).
3. Ядро виконує повний ітеративний цикл "генерація — прогнозування — оцінювання", обираючи єдиний оптимальний керуючий вплив.
4. Сформована команда передається на Рівень взаємодії для виконання у фізичному середовищі (або імітаційній моделі).

Весь процес, включаючи вектори станів, прогнози, оцінки корисності, прийняті рішення та факти ручної корекції, підлягає детальній реєстрації (протоколюванню). Цей масив даних слугує не лише для аналізу результатів експериментів, але й виступає навчальною вибіркою для Модуля адаптивного навчання, забезпечуючи безперервне вдосконалення стратегій керування.

4.3. Математична модель прийняття рішень на основі багатоцільової оптимізації

В основі роботи розробленої проактивної архітектури керування лежить математична модель, що формалізує процес прийняття рішень в умовах багатоцільовості та невизначеності. Ця модель реалізує ключові принципи керування на основі намірів, дозволяючи системі автономно обирати оптимальні дії для досягнення балансу між суперечливими цілями, такими як комфорт користувача та економія енергоресурсів. Модель базується на принципі максимізації функції корисності (Utility Function), яка кількісно оцінює прогнозовані наслідки кожної потенційної керуючої дії з точки зору поточних, контекстуально-залежних намірів користувача.

Стан середовища в момент часу t визначається вектором S_t :

$$S_t = (T_{in,t}, T_{out,t}, H_t, C_t), \quad (4.1)$$

де $T_{in,t}$ — внутрішня температура в приміщенні ($^{\circ}\text{C}$), $T_{out,t}$ — зовнішня температура ($^{\circ}\text{C}$), H_t — година доби (від 0 до 23,75), C_t — поточний тариф на електроенергію (напр., 0 для нічного, 1 для пікового).

Агент може виконати одну дію a_t із скінченного набору можливих дій $A = \{a_0, a_1, \dots, a_n\}$, де кожна дія відповідає певному режиму роботи обігрівача з потужністю $\text{Power}(a_i)$.

Проактивність системи забезпечується прогностичною функцією P , яка для поточного стану S_t та гіпотетичної дії повертає прогнозований стан середовища S'_{t+1} у наступний момент часу:

$$S'_{t+1} = P(S_t, a). \quad (4.2)$$

Функція P інкапсулює фізичну модель середовища (наприклад, термодинамічну модель приміщення), яка дозволяє розрахувати, як зміниться стан (зокрема, T_{t+1}) в результаті застосування дії a в умовах стану S_t . У рамках даного дослідження використовувалася ідеальна модель для фокусування на механізмі прийняття рішень.

Наміри користувача, що визначають баланс між комфортом та економією, представлені у вигляді вагових коефіцієнтів, які залежать від поточного контексту. Контекст визначається функцією $K(S_t)$, яка на основі поточного стану (наприклад, години доби H_t та тарифу C_t) повертає ключ контексту (наприклад, "evening_peak", "night_offpeak").

Для кожного контексту $K(S_t)$ у Сховищі Знань зберігається пара вагових коефіцієнтів:

$$W(K(S_t)) = (w_{\text{comfort}}, w_{\text{energy}}), \quad (4.3)$$

$$w_{\text{comfort}} + w_{\text{energy}} = 1, \quad (4.4)$$

де w_{comfort} — вага пріоритету комфорту, а w_{energy} — вага пріоритету економії, причому виконується умова нормування:

Ці ваги не є статичними; вони динамічно оновлюються Модулем Адаптивного Навчання на основі аналізу ручних втручань користувача.

Центральним елементом математичної моделі є функція корисності U , яка кількісно оцінює "якість" або "бажаність" прогнозованого стану S'_{t+1} , що є результатом виконання дії a_t у поточному стані S_t , з точки зору поточних контекстуальних намірів користувача. Функція U є зваженою сумою двох компонентів: оцінки комфорту та оцінки енергоефективності:

$$U(S'_{t+1}, a_t | S_t) = w_{\text{comfort}} \cdot \text{Score}_{\text{comfort}}(S'_{t+1}) + w_{\text{energy}} \cdot \text{Score}_{\text{energy}}(a_t, S_t), \quad (4.5)$$

де w_{comfort} та w_{energy} є вагами з $W(K(S_t))$.

Компонента оцінка комфорту $\text{Score}_{\text{comfort}}$ оцінює, наскільки прогнозована внутрішня температура T'_{t+1} відповідає зоні комфорту користувача $[T_{\min}, T_{\max}]$. Функція має нелінійний, асиметричний характер, щоб сильно стимулювати агента перебувати всередині зони комфорту та жорстко штрафувати за вихід за її межі:

$$\text{Score}_{\text{comfort}}(S'_{t+1}) = \begin{cases} 1 - \frac{|T'_{t+1} - T_{\text{center}}|}{T_{\text{max}} - T_{\text{center}}}, & \text{якщо } T_{\text{min}} \leq T'_{t+1} \leq T_{\text{max}}, \\ \text{П}_{\text{base}} - |T'_{t+1} - T_{\text{bound}}|, & \text{в іншому випадку} \end{cases} \quad (4.6)$$

де:

T_{center} — центр зони комфорту,

П_{base} — великий базовий штраф за дискомфорт (напр., -100),

T_{bound} — найближча межа зони комфорту T_{min} або T_{max} .

Оцінка енергоефективності $\text{Score}_{\text{energy}}$ компонента оцінює економічність дії a_t , враховуючи спожиту потужність $\text{Power}(a_t)$ та поточний тарифний мультиплікатор $M(C_t)$ (наприклад, 1 для нічного тарифу, 3 для пікового):

$$\text{Score}_{\text{energy}}(a_t, S_t) = 1 - \left(\frac{\text{Power}(a_t)}{\text{Power}_{\text{max}}} \cdot M(C_t) \right), \quad (4.7)$$

де:

$\text{Power}_{\text{max}}$ — максимальна потужність обігрівача,

$M(C_t)$ — мультиплікатор тарифу.

Значення оцінки варіюється від 1 (нульове споживання) до негативних значень при високому споживанні під час дії дорогого тарифу.

Оптимізаційна задача. На кожному часовому кроці t система вирішує оптимізаційну задачу: знайти таку дію a_t^* , яка максимізує функцію корисності U для прогнозованого стану, що є результатом цієї дії:

$$a_t^* = \arg \max_{(a \in A)} U(P(S_t, a), a | S_t). \quad (4.8)$$

Ця формула є математичним вираженням проактивного, багатоцільового (через зважену суму комфорту та економії) та керованого намірами (через динамічні ваги w_{comfort} , w_{energy}) процесу прийняття рішень, який реалізовано в розробленій архітектурі. Вибір дії a_t^* є оптимальним рішенням системи на даному кроці, що найкращим чином відповідає поточним цілям та умовам.

4.4. Механізм адаптивного навчання контекстуальним намірам на основі неявного зворотного зв'язку

Ключовою властивістю, що визначає переваги розробленої проактивної архітектури, є її іманентна здатність до автономного навчання та безперервної адаптації, що спрямована на динамічне підлаштування стратегії керування до індивідуальних, мінливих у часі та, що найбільш важливо, часто неявних вподобань кінцевого користувача.

На принципову відміну від класичних систем, що вимагають від мешканця обтяжливого явного програмування детермінованих правил або складного, детального налаштування чисельних параметрів, запропонований в даній роботі підхід реалізує механізм навчання на основі неявного зворотного зв'язку. Цей механізм використовує природну поведінку мешканця, що взаємодіє з середовищем, як основне та найбільш достовірне джерело інформації для ітеративного коригування системної стратегії керування.

Центральним виконавчим елементом цього механізму є Модуль Адаптивного Навчання, детально описаний в архітектурі системи (підрозділ 4.2). Його функціонування повністю базується на аналізі випадків ручного втручання користувача. Ручне втручання визначається як подія, під час якої користувач, висловлюючи незадоволення поточним автоматичним рішенням системи, власноруч змінює стан керованого виконавчого пристрою. Типовим прикладом є ситуація, коли користувач вручну вмикає обігрівач, який система автоматично вимкнула з метою економії, або, у протилежній ситуації, вимикає його, коли система вважає за потрібне підтримувати вищу температуру. Таке втручання інтерпретується системою не як помилка чи випадкова дія користувача, а як надзвичайно цінний сигнал корекції. Він свідчить про явну невідповідність між поточною моделлю вподобань системи, яка формалізована через вагові коефіцієнти W_{comfort} та W_{energy} , та справжньою, прихованою функцією корисності користувача у даному конкретному контексті.

Принцип роботи механізму навчання є ітеративним та базується на наступній послідовності кроків. Система, функціонуючи в штатному режимі, здійснює безперервний моніторинг стану керованих пристроїв. У випадку, якщо вона детектує зміну стану, наприклад, перехід обігрівача зі стану "Вимкнено" у стан "Увімкнено", що була ініційована користувачем, а не самою системою, і яка скасовує попереднє автоматичне рішення, ця подія негайно реєструється та кваліфікується як факт ручного втручання.

У момент фіксації втручання система негайно здійснює захоплення та категоризацію поточного контексту $K(S_t)$. Контекст являє собою вектор ознак, що описує ситуацію, в якій відбулося втручання (наприклад, "вечірній час, піковий тариф, присутні люди" або "ранок, низький тариф, відсутність людей"). Цей крок є критично важливим та становить наріжний камінь усього механізму. Він дозволяє системі навчатися не єдиним, усередненим "глобальним" вподобанням, а саме контекстуально-залежним намірам, диференціюючи поведінкові патерни користувача для різних життєвих ситуацій.

Наступним кроком є семантичний аналіз напрямку втручання. Модуль Адаптивного Навчання встановлює, в який бік користувач скоригував дію системи, щоб зрозуміти суть невідповідності.

Якщо дія користувача була спрямована на підвищення рівня комфорту, наприклад, увімкнення обігрівача, збільшення його потужності, яке система вважала недоцільним, це однозначно інтерпретується як сигнал. Сигнал свідчить, що поточна вага комфорту w_{comfort} для даного контексту $K(S_t)$ є заниженою, а політика системи – надто орієнтованою на економію.

У протилежному випадку, якщо користувач зменшив рівень комфорту (вимкнув або зменшив потужність пристрою), який система активно підтримувала, це інтерпретується як сигнал, що поточна вага комфорту w_{comfort} є завищеною, а пріоритет економії w_{energy} – заниженим.

На основі результатів аналізу напрямку втручання, система виконує інкрементальне коригування вагових коефіцієнтів у Сховищі Знань. Важливо

підкреслити, що коригування стосується виключно того контексту $K(S_t)$, в якому відбулося втручання, не зачіпаючи налаштування для інших ситуацій.

Якщо аналіз виявив потребу збільшити пріоритет комфорту, вага $w_{comfort}$ незначно збільшується на фіксований крок Δw :

$$w_{comfort}(K) \leftarrow w_{comfort}(K) + \Delta w. \quad (4.9)$$

Якщо потрібно збільшити пріоритет економії, вага $w_{comfort}$ відповідно зменшується:

$$w_{comfort}(K) \leftarrow w_{comfort}(K) - \Delta w. \quad (4.10)$$

Вага економії w_{energy} при цьому автоматично перераховується для збереження нормалізації

$$w_{energy} = 1 - w_{comfort}. \quad (4.11)$$

Величина кроку коригування Δw є важливим гіперпараметром системи, що визначає швидкість навчання та впливає на баланс між стабільністю моделі та швидкістю її збіжності до вподобань користувача.

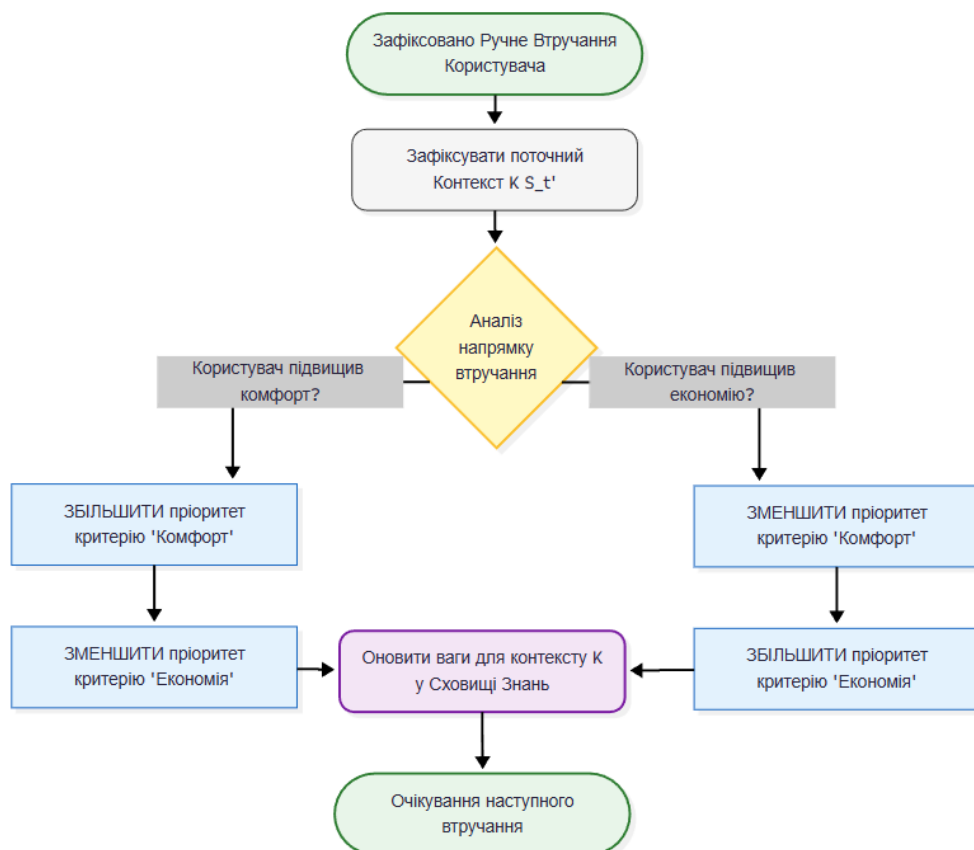


Рисунок 4.5: Блок-схема алгоритму адаптивного навчання на основі втручання

Процес повторюється при кожному наступному ручному втручанні, тобто реалізується ітеративний процес стохастичної апроксимації, де кожне втручання користувача є "порцією" нової інформації, що уточнює модель. З часом, завдяки послідовним коригуванням, ваги пріоритетів для кожного окремого контексту асимптотично збігаються до значень, які найкраще відображають справжні, індивідуальні компроміси між комфортом та економією, притаманні даному користувачу.

Переваги навчання на основі неявного зворотного зв'язку є суттєвими. По-перше, цей механізм не вимагає від користувача жодних спеціальних дій для "навчання" системи, зміни налаштувань чи програмування; навчання відбувається повністю автоматично, як природний побічний продукт звичайної експлуатації житла. По-друге, реалізується фундаментальний принцип людино-орієнтованого дизайну: система адаптується до користувача, а не навпаки, що радикально знижує когнітивне навантаження. По-третє, такий підхід дозволяє системі вивчити надзвичайно складні, залежні від ситуації та нелінійні вподобання, які користувач часто й сам не здатен чітко формалізувати через явні правила. Кінцевою метою цього процесу є мінімізація кількості майбутніх ручних втручань до нуля, оскільки система стає все кращою у передбаченні та проактивному задоволенні потреб користувача, що, в свою чергу, є головним індикатором успішності адаптації.

Таким чином, розроблений механізм адаптивного навчання є ключовим компонентом, що забезпечує гнучкість, персоналізацію та справжню автономність проактивної архітектури керування. Він перетворює систему зі статичного інструменту на динамічного "партнера", що здатен еволюціонувати разом з користувачем та його звичками, створюючи по-справжньому інтелектуальне, комфортне та водночас ресурсоефективне житлове середовище.

Висновки до розділу 4.

У даному розділі було розроблено нову архітектуру проактивного керування розумним будинком, спрямовану на подолання обмежень традиційних реактивних систем та створення більш гнучких, адаптивних та людино-орієнтованих рішень. На основі проведеної роботи зроблено наступні висновки:

Обґрунтовано концепцію керування на основі високорівневих намірів. Проаналізовано недоліки керування через низькорівневі правила та показано переваги переходу до парадигми, де користувач визначає бажані цілі (наприклад, "баланс комфорту та економії"), а система самостійно знаходить оптимальні шляхи їх досягнення. Встановлено, що ключовими елементами такої концепції є моделювання вподобань користувача, контекстна обізнаність та багатоцільова оптимізація.

Розроблено компонентну архітектуру проактивної системи. Запропоновано модульну архітектуру, що складається з Блоку середовища і взаємодії, Ядра проактивного керування та Блоку знань і адаптації. Ядро системи реалізує проактивний цикл "генеруй-прогнозуй-оцінюй", використовуючи прогностичну модель для оцінки майбутніх станів та багатоцільовий оптимізатор для вибору найкращої дії відповідно до поточних намірів.

Формалізовано математичну модель прийняття рішень. Розроблено математичну модель, що базується на максимізації функції корисності. Ця функція є зваженою сумою оцінок комфорту та енергоефективності, де вагові коефіцієнти відповідають поточним контекстуальним намірам користувача. Модель враховує як прогнозований стан середовища, так і вартість дії, дозволяючи знаходити оптимальний компроміс між суперечливими цілями.

Запропоновано механізм адаптивного навчання контекстуальним намірам. Розроблено ключовий компонент архітектури — Модуль Адаптивного Навчання, що реалізує навчання на основі неявного зворотного зв'язку. Механізм аналізує ручні втручання користувача як сигнал невідповідності поточної стратегії його справжнім бажанням у певному контексті та використовує цю інформацію для інкрементального коригування вагових коефіцієнтів намірів. Це дозволяє системі

автономно навчатися складним, динамічним та контекстуально-залежним вподобанням мешканців.

Таким чином, у розділі представлено завершене концептуальне, архітектурне та математичне обґрунтування проактивної системи керування розумним будинком. Розроблена архітектура поєднує прогнозування, багатоцільову оптимізацію та адаптивне навчання контекстуальним намірам, що створює теоретичну основу для побудови інтелектуальних систем нового покоління, здатних мінімізувати когнітивне навантаження на користувача та ефективно керувати ресурсами в динамічному середовищі. Експериментальна перевірка ефективності цієї архітектури буде проведена у наступному розділі.

РОЗДІЛ 5. ЕКСПЕРИМЕНТАЛЬНЕ ДОСЛІДЖЕННЯ ТА АПРОБАЦІЯ РЕЗУЛЬТАТІВ

5.1. Розробка імітаційного стенду та методика проведення експериментів

Для об'єктивної оцінки ефективності та валідації розроблених у попередніх розділах методик — вдосконаленого методу прогнозування TCN-LightGBM, адаптивного методу очищення даних ACRA та проактивної архітектури керування — було проведено комплексне експериментальне дослідження. Зважаючи на складність, вартість та тривалість розгортання та тестування таких систем у реальних умовах, ключовим інструментом дослідження стало імітаційне моделювання. Воно дозволяє відтворювати динамічні процеси в контрольованому середовищі, проводити багаторазові експерименти зі зміною умов та отримувати кількісні показники ефективності для порівняльного аналізу.

5.1.1. Імітаційний стенд для валідації проактивної архітектури керування

Для експериментальної валідації, всебічної перевірки та змістовного порівняльного аналізу ефективності розробленої проактивної архітектури керування (детально описаної у Розділі 4), було спроектовано та імплементовано спеціалізований, гнучкий імітаційний стенд (середовище симуляції).

Технологічною основою для реалізації цього стенду було обрано мову програмування Python, зважаючи на її потужну екосистему для наукових обчислень та моделювання. Ключовим компонентом стала бібліотека Gymnasium, що є прямим наступником та сучасною реінкарнацією фундаментальної бібліотеки OpenAI Gym, являє собою визнаний де-факто академічний та індустріальний стандарт. Його використання методологічно обґрунтоване, оскільки Gymnasium надає стандартизований, надійний та гнучкий інтерфейс

(API) для розробки, тестування та бенчмаркінгу різноманітних агентів керування, зокрема тих, що базуються на методах навчання з підкріпленням (Reinforcement Learning). Це дозволяє агентам ітеративно навчатися оптимальним стратегіям поведінки шляхом безпосередньої взаємодії з контрольованим середовищем.

Центральним об'єктом моделювання у стенді є процес керування мікрокліматом (зокрема, системою опалення) в межах однієї умовної житлової кімнати. Таке навмисне спрощення дозволяє ізолювати дослідження від надлишкової складності багатозонних систем та глибоко сфокусуватися на аналізі саме динаміки прийняття рішень агентом в умовах зовнішніх факторів, що постійно та непередбачувано змінюються.

Все середовище функціонує як динамічна система, що еволюціонує у часі з дискретним кроком в 15 хвилин. Цей інтервал є типовим для сучасних систем смарт-метрингу та керування будівлями. В кожен момент часу t , повний стан середовища, який агент отримує в якості спостереження, описується комплексним вектором ознак, що включає всю критично необхідну для прийняття рішення інформацію:

1. Поточна внутрішня температура в кімнаті $T'_{in,t}$ ключовий контрольований параметр, що відображає рівень комфорту.
2. Поточна зовнішня температура $T_{out,t}$ основне джерело теплових втрат та головний "збурюючий" фактор середовища.
3. Година доби H_t циклічна ознака, що дозволяє агенту враховувати добові патерни (наприклад, очікувану присутність людей чи зміну тарифів).
4. Поточний тариф на електроенергію C_t економічний сигнал, що визначає вартість дії (ввімкнення обігріву).

Динаміка зміни стану середовища базується на фізично обґрунтованій термодинамічній моделі першого порядку. Розрахунок еволюції внутрішньої температури $T_{in,t+1}$ здійснюється шляхом розв'язання рівняння теплового балансу, що враховує два різноспрямовані вектори впливу.

Тепловтрати визначаються коефіцієнтом теплопровідності огорожувальних конструкцій та пропорційні градієнту температур між внутрішнім простором та зовнішнім середовищем ($T_{\{in,t\}} - T_{\{out,t\}}$).

Теплонадходження детермінуються роботою електричного нагрівального елемента, функціонування якого залежить від дискретної керуючої дії агента на поточному часовому кроці.

З метою верифікації адаптивних властивостей системи в умовах нестаціонарності середовища, експериментальний період тривалістю 60 діб структуровано на два послідовні етапи (сезони) з відмінними кліматичними характеристиками (див. Таблицю 3.2).

Таблиця 5.1

Параметри сценарного моделювання середовища

Етап моделювання	Часовий інтервал	Характеристика середовища	Мета етапу
Етап 1: «Осінь»	1–30 доба	Помірні зовнішні температури, низька амплітуда коливань.	Формування базового сценарію навантаження.
Етап 2: «Зима»	31–60 доба	Низькі середні температури, високі тепловтрати.	Стрес-тестування: перевірка адаптації стратегій до суворих умов.

Економічний аспект середовища формалізовано через впровадження динамічної двозонної тарифної сітки («день/ніч»). Наявність суттєвого диспаритету вартості електроенергії стимулює інтелектуального агента до відмови від реактивних стратегій на користь проактивного планування, наприклад, попереднє нагрівання приміщення у пільговий нічний період, що створює задачу багатокритеріальної оптимізації: мінімізація витрат при збереженні комфорту.

Критичним елементом розробленого стенду є модель стохастичної поведінки мешканця — SimUser, функціонал якої виходить за межі статичного задання уставок. Модель виконує функцію генератора неявного зворотного зв'язку для

модуля адаптивного навчання (A-RL), імітуючи реакцію реальної людини на дискомфорт.

Алгоритм функціонування SimUser враховує фактор часової толерантності: реакція на відхилення температури від цільового діапазону (наприклад, 20–22°C) настає не миттєво, а після закінчення періоду очікування (latency period), встановленого на рівні 1 години.

Логіка генерації «ручних втручань» (overrides), які виступають навчальним сигналом для системи, базується на двох контекстуально-залежних пріоритетах. Блок-схема алгоритму прийняття рішень SimUser наведена на Рисунку 5.1.

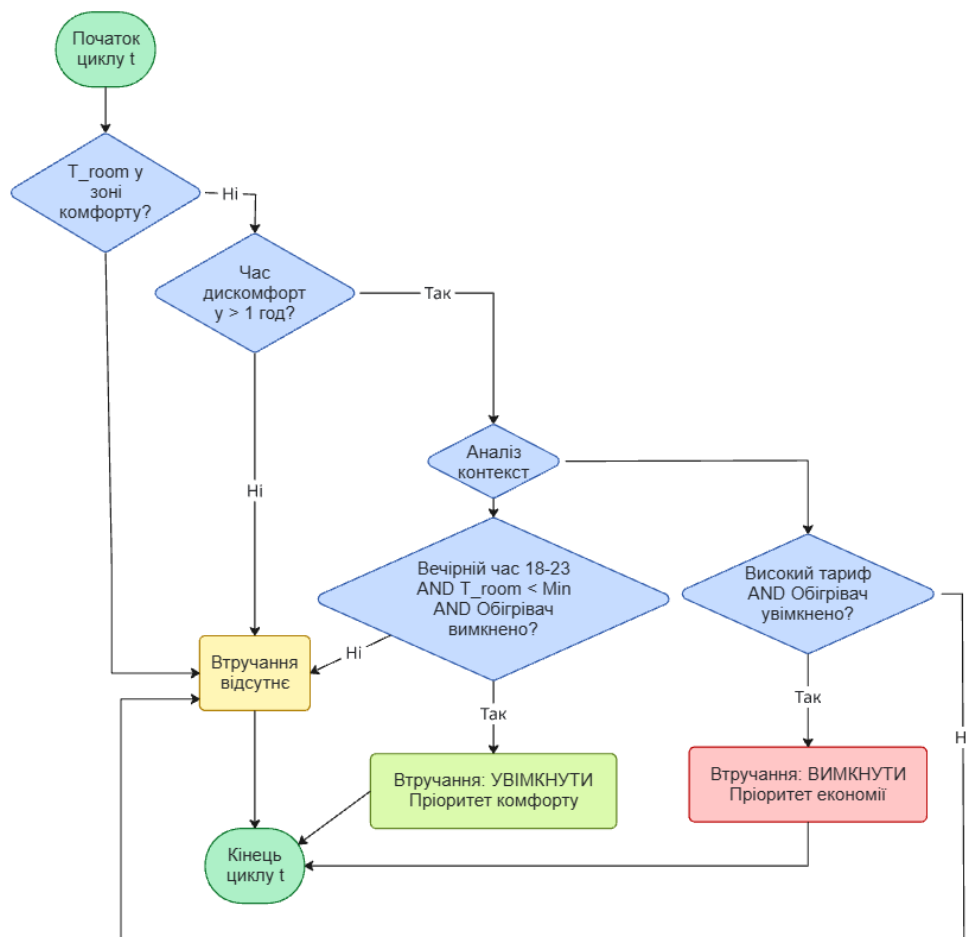


Рисунок 5.1: Алгоритм генерації втручань моделлю SimUser

Пріоритезація економічної ефективності (Cost-driven override). Втручання типу «Примусове вимкнення» генерується, якщо система здійснює активний обігрів під час дії високого тарифу, за умови, що температура ще не досягла

критичного мінімуму. Це моделює поведінку користувача, схильного жертвувати ідеальним комфортом заради економії.

Пріоритезація комфорту (Comfort-driven override). Втручання типу «Примусове увімкнення» ініціюється у випадку, якщо система внаслідок помилки прогнозу або надмірної економії допускає зниження температури нижче допустимої межі у пріоритетні вечірні години (18:00–23:00).

Застосування такої контекстно-орієнтованої моделі дозволяє оцінити здатність агента навчатися нелінійній логіці прийняття рішень, притаманній людині, а не лише слідувати детермінованим правилам.

Виконавчий модуль архітектури H-AD-CLEAN відмовляється від складної системи перемикання операторів, характерної для попередніх ітерацій (методу ACRA), на користь робастної бінарної логіки («чистити» / «не чистити»). Підхід отримав назву «Вентильна логіка» (Gating Logic).

5.1.2. Набори даних для експериментів з прогнозування та очищення

Для всебічного дослідження прогнозної ефективності та обчислювальної продуктивності запропонованої гібридної моделі TCN-LightGBM (детально описаної у Розділі 2), було свідомо обрано стандартизований та загальнодоступний відкритий набір даних. Такий підхід гарантує прозорість, порівнюваність та можливість відтворення отриманих результатів іншими дослідниками у цій галузі.

Джерелом слугував масив даних, що містить високочастотні 15-хвилинні показники споживання електроенергії, зібрані за допомогою "розумних" лічильників. Вибірка охоплює 130 різноманітних приватних домогосподарств на території Німеччини, а період збору даних охоплює повний календарний 2019 рік, що забезпечує наявність усіх сезонних патернів [35].

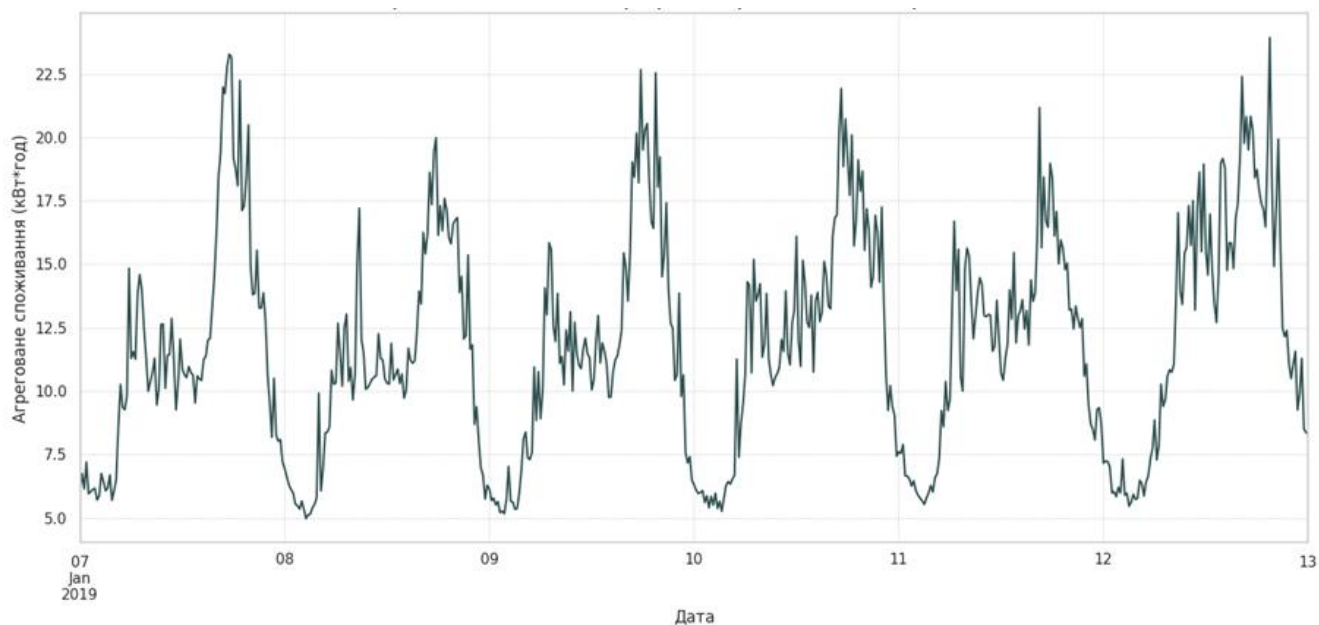


Рисунок 5.2. Приклад тижневого профілю агрегованого енергоспоживання

На етапі підготовки даних, для формування цільової змінної дослідження було створено єдиний агрегований часовий ряд P_{agg} . Його було отримано шляхом поточкового сумування показників споживання всіх 130 домогосподарств для кожного 15-хвилинного інтервалу. Такий агрегований сигнал є більш стабільним, менш "зашумленим" і типово використовується для задач прогнозування навантаження на рівні локальної трансформаторної підстанції чи мікрорайону.

З метою суттєвого покращення якості прогнозів було проведено розширений інжиніринг ознак. На основі часових міток було згенеровано додаткові календарні та циклічні ознаки: година доби (0-23), день року (1-365), день тижня та окрема бінарна ознака вихідного дня.

Для коректного представлення циклічності ознак та збереження безперервності часових циклів у просторі ознак, до них було застосовано синусоїдальне перетворення.

На фінальному етапі препроцесингу, абсолютно всі вхідні дані (як цільовий ряд P_{agg} , так і всі згенеровані ознаки) було масштабовано до єдиного числового діапазону $[0, 1]$. Для цього використовувався стандартний метод Min-Max нормалізації. Ця операція є критично важливою для забезпечення стабільного

навчання компонента TCN (Temporal Convolutional Network) та запобігання домінуванню ознак з більшим масштабом.

Для забезпечення високої статистичної значущості отриманих результатів та надійної оцінки здатності моделі до узагальнення, була застосована спеціалізована процедура 5-блокової (5-fold) перехресної валідації, адаптованої для часових рядів.

Використовувався принцип "ковзного вікна" (sliding window cross-validation), який, на відміну від класичної випадкової k-fold валідації, суворо зберігає хронологічний порядок даних, уникаючи "заглядання в майбутнє" для оцінки стійкості та узагальнюючої здатності моделі на п'яти різних, незалежних ділянках історичних даних.

Вибір експериментального набору даних обумовлений його високою репрезентативністю та статистичною значущістю. Агрегація показників споживання від 130 домогосподарств забезпечує ефективне нівелювання стохастичних флуктуацій (випадкових шумів), притаманних індивідуальним профілям навантаження.

Висока часова роздільна здатність з інтервалом дискретизації 15 хвилин відповідає галузевим стандартам оперативного керування енергосистемами, а наявність виражених добових та тижневих циклічних патернів робить даний масив релевантною основою для апробації та верифікації моделей короткострокового прогнозування (STLF).

Для об'єктивної оцінки ефективності розробленого методу адаптивного очищення та відновлення даних (ACRA) застосовано методологію напівсинтетичного моделювання (semi-synthetic modeling). Сутність підходу полягає у використанні валідованого масиву реальних даних як еталонної бази («ground truth»), на яку здійснюється контрольована інжекція синтетичних аномалій та шумів із заданими параметрами розподілу.

Типологія та кількісні параметри модельованих спотворень наведені у Таблиці 5.2.

Таблиця 5.2.

Параметри модельованих аномалій та шумів

Тип аномалії	Опис дефекту	Параметри інжекції
Точкові викиди (Spikes)	Раптові аномальні відхилення амплітуди сигналу одиничної тривалості.	~3% від загального обсягу вибірки.
Константні значення (Stuck-at faults)	Імітація апаратного збою сенсора («зависання»), що характеризується незмінністю значень протягом певного інтервалу.	25 сегментів варіативної тривалості.
Дрейф сенсора (Drift)	Поступове систематичне зміщення (bias) вимірних значень відносно істинного сигналу.	20 сегментів лінійного або нелінійного тренду.
Адитивний шум (Additive Gaussian Noise)	Модельовання фоновому вимірювального шуму обладнання з нормальним розподілом.	Спотворення ~36% точок даних.
Втрата даних (Missing Values, NaN)	Імітація помилок передачі даних або втрати зв'язку з вузлом мережі.	Видалення ~5% точок даних.

В якості джерела еталонних даних використано показники сенсорної мережі IoT, розгорнутої на базі об'єкта житлової інфраструктури у північно-східному регіоні Мексики. Збір даних здійснювався з високою частотою дискретизації (1 вимірювання на хвилину) протягом 14-місячного періоду спостережень (з листопада 2022 року по січень 2024 року), що забезпечило достатній обсяг вибірки для навчання та тестування алгоритмів [22].

Для проведення детального експериментального дослідження було відібрано три репрезентативні одновимірні часові ряди. Вибір базувався на критерії охоплення процесів різної фізичної природи та динамічних властивостей, що дозволяє комплексно оцінити ефективність алгоритмів обробки в Таблиці 5.3. В якості базового часового інтервалу виділено безперервні сегменти даних, що відповідають календарному періоду січень 2023 року.

Таблиця 5.3.

Характеристика тестових часових рядів

Позначення	Фізична величина	Характеристика динаміки сигналу
Temp	Внутрішня температура приміщення	Висока інерційність. Сигнал характеризується плавними змінами («гладкий» профіль), низькою волатильністю та залежністю від теплової інерції будівлі.
Humidity	Внутрішня відносна вологість	Середня волатильність. Динаміка сигналу є більш вираженою порівняно з температурою, з помірними флуктуаціями.
Active_Power	Активне енергоспоживання будівлі	Стохастичний характер. Сигнал відзначається високою волатильністю, імпульсною природою змін та наявністю різких стрибків, зумовлених дискретним увімкненням приладів.

Дослідницька методологія базується на концепції «напівсинтетичного» моделювання. Вихідні верифіковані дані розглядалися як еталонний сигнал («ground truth»). Для формування тестового середовища, що відтворює реальні умови експлуатації сенсорних мереж IoT, до еталонних даних було застосовано процедуру контрольованої інжекції комплексної суміші синтетичних шумів та аномалій.

Слід зазначити, що параметри генерації шумів (амплітуда, тривалість сегментів, дисперсія розподілу) підлягали індивідуальній калібровці для кожного часового ряду (Temp, Humidity, Active_Power). Адаптація параметрів здійснювалася з урахуванням фізичної специфіки сигналів, їх динамічного діапазону та типової швидкості зміни, що забезпечило високий рівень достовірності імітаційної моделі.

Використання такого напівсинтетичного підходу є методологічно необхідним у даному випадку у зв'язку з тим, що надає унікальну перевагу: дослідник має абсолютно точну інформацію про істинний ("еталонний") сигнал та володіє

повною картою (мітками) типу та місцезнаходження помилки в кожний момент часу. Це є єдиним надійним способом об'єктивно оцінити якість роботи саме алгоритмів очищення відновлення даних (на відміну від задачі простого виявлення аномалій, де такий рівень деталізації не завжди потрібен).

На завершальному етапі, отримані зашумлені дані (разом з "еталонними" значеннями та мітками типу шуму) були розділені зі збереженням хронологічного порядку на дві вибірки:

- Навчальна вибірка (Training set): Перші 70% даних послідовності.
- Тестова вибірка (Test set): Останні 30% даних послідовності.

Класифікатор аномалій, що є ключовим компонентом методу ACRA, навчався виключно на даних навчальної вибірки. Всі фінальні оцінки ефективності та якості очищення (за метриками RMSE та MAE) проводилися виключно на небаченій раніше тестовій вибірці, що гарантує об'єктивність оцінки узагальнюючої здатності методу.

5.1.3. Метрики оцінювання ефективності

Для забезпечення об'єктивної кількісної оцінки та всебічного порівняльного аналізу ефективності функціонування розроблених методик, було сформовано та валідовано спеціалізований комплекс метрик. Вибір саме цих показників обумовлений необхідністю детально проаналізувати різноманітні аспекти та характеристики роботи систем, включаючи їх загальну точність, робастність (стійкість) до аномалій та здатність адекватно опрацьовувати дані різного масштабу.

Середньоквадратична помилка (Root Mean Squared Error, RMSE) є одним із найбільш поширених стандартних статистичних інструментів для вимірювання величини похибки (різниці) між прогнозованими та фактичними значеннями? розраховується як квадратний корінь із середнього арифметичного квадратів відхилень:

$$\text{RMSE} = \sqrt{\left(\frac{1}{N} \sum_{t=1}^N (y_t - \hat{y}_t)^2\right)}, \quad (5.1)$$

де y_t – це фактичне, істинне значення в момент часу t ;

\hat{y}_t – це відповідне прогнозоване або відновлене значення, отримане від моделі;

N – загальна кількість спостережень (точок даних) у вибірці.

Ключова характерна особливість RMSE полягає в операції піднесення помилки ($y_t - \hat{y}_t$) до квадрату ще до усереднення. Це призводить до того, що метрика стає особливо чутливою до великих, аномальних помилок (викидів). Такі значні відхилення отримують диспропорційно велику "штрафну" вагу в загальній оцінці порівняно з малими відхиленнями. Саме ця властивість робить RMSE критично важливою метрикою для оцінки тих моделей і систем, де навіть поодинокі, але грубі помилки та значні відхилення є категорично неприпустимими та можуть призвести до серйозних негативних наслідків в управлінні.

Середня абсолютна помилка (Mean Absolute Error, MAE) також оцінює середню величину помилок прогнозу чи відновлення, однак її розрахунок базується на усередненні модулів (абсолютних значень) відхилень:

$$\text{MAE} = \frac{1}{N} \sum_{t=1}^N |y_t - \hat{y}_t| \quad (5.2)$$

На принципову відміну від RMSE, де застосовується квадратичне штрафування, MAE використовує лінійне. Це означає, що усі помилки – як малі, так і великі – враховуються пропорційно до їхньої фактичної величини, без додаткового "посилення" для викидів. Як наслідок, MAE є значно менш чутливою до присутності аномальних значень (outliers) у даних і вважається більш робастною оцінкою.

Важливою перевагою MAE є те, що вона надає більш пряму та інтуїтивно зрозумілу оцінку середньої величини помилки, оскільки її результат виражається в тих самих одиницях вимірювання, що й вихідні дані (наприклад, у °C для

температури або у кВт·год для енергоспоживання). У рамках даного дослідження метрика MAE використовувалася переважно для оцінки ефективності розробленого методу очищення та відновлення пропущених даних ACRA, де важлива саме середня точність відновлення.

Середня абсолютна відсоткова помилка (Mean Absolute Percentage Error, MAPE) представляє собою відносну міру похибки, яка вимірює середнє абсолютне відхилення у вигляді відсотка від відповідного істинного значення:

$$\text{MAPE} = \frac{100\%}{N} \sum_{t=1}^N \left| \frac{y_t - \hat{y}_t}{y_t} \right|, \quad (5.3)$$

Ключовою перевагою MAPE є її нормалізація відносно фактичного значення y_t . Завдяки діленню помилки на істинне значення, метрика стає безрозмірною величиною (вираженою у відсотках). Це робить її надзвичайно популярною та зручною метрикою у задачах прогнозування, оскільки вона дозволяє безпосередньо порівнювати точність прогнозів на часових рядах, які мають абсолютно різні масштаби та одиниці вимірювання (наприклад, порівнювати якість прогнозу споживання електроенергії в кВт та прогнозу кількості мешканців у приміщенні в особах). У роботі дана метрика слугувала одним з основних індикаторів якості для оцінки прогнозової моделі споживання TCN-LightGBM. Слід зазначити відоме обмеження цієї метрики: вона стає нестабільною, невизначеною (ділення на нуль) або дає надмірно великі значення, коли істинні значення y_t є нульовими або дуже близькими до нуля.

Пікова середня абсолютна відсоткова помилка (Peak Magnitude MAPE, Peak-MAPE) являє собою спеціалізовану, адаптовану версію стандартної MAPE, спрямовану на оцінку якості прогнозу в найбільш критичних точках. Методологія її розрахунку є ідентичною до MAPE, однак він проводиться не на всьому наборі даних, а виключно на цільовій підвибірці. Підвибірка формується лише з тих моментів часу, що відповідають піковим навантаженням, наприклад, це можуть бути дані, що потрапляють у верхній 10-й перцентиль або 10% найвищих значень споживання протягом доби.

Впровадження та аналіз цієї метрики має критично важливе значення для адекватної оцінки моделей прогнозування саме в галузі енергоспоживання. Це зумовлено тим, що помилки в прогнозуванні саме пікових значень, а не середніх чи низьких, мають найбільші та найбільш відчутні економічні та операційні наслідки. Неточне прогнозування піків може призвести до значних фінансових втрат через штрафи від енергопостачальних компаній (за перевищення лімітів), неправильного планування закупівель енергії на ринку, збоїв у роботі програм керування попитом (Demand-Response) та, в глобальному масштабі, до ризиків перевантаження та дестабілізації локальної або навіть регіональної енергосистеми.

Для комплексної та багатокритеріальної оцінки ефективності розробленого проактивного агента керування було обрано три ключові системні метрики. На відміну від метрик, що оцінювали точність окремих підсистем (прогнозування, очищення даних), цей набір показників оцінює безпосередньо якість прийнятих керуючих рішень та їхній сукупний вплив на систему «користувач – будівля – енергосистема». Ці метрики відображають фундаментальний баланс між трьома основними цілями керування: економічною доцільністю, якістю обслуговування (комфортом) та рівнем автоматизації (користувацьким досвідом).

Сукупні фінансові витрати (Total Financial Cost) – інтегральний показник, що кількісно виражає загальну вартість електроенергії (у гривнях або інших умовних одиницях, у.о.), яка була фактично спожита системою опалення (та/або кондиціонування) за весь період симуляційного експерименту.

Розрахунок проводиться шляхом агрегування спожитих кіловат-годин (кВт·год) на кожному часовому кроці симуляції та подальшого множення цих обсягів на відповідну вартість енергії згідно з динамічною тарифною сіткою (наприклад, багатозонним тарифом "день/ніч" або більш складними тарифами реального часу). Ця метрика безпосередньо відображає економічну ефективність та оптимальність стратегії керування, яку обрав агент. Здатність агента "переносити" енергетичне навантаження з "пікових" (дорогих) годин на "долини"

(дешеві), не втрачаючи при цьому комфорту, є ключовим проявом його проактивної інтелектуальної поведінки.

Рівень дотримання комфорту (Comfort Compliance Level) – ключовий показник якості обслуговування (QoS – Quality of Service), що виражається у відсотках (%) від загального часу симуляції. Він демонструє, наскільки успішно система справлялася зі своїм основним завданням.

Розраховується як відношення сумарного часу, впродовж якого фактична температура в приміщенні (або інший контрольований параметр) знаходилася всередині чітко визначеного користувачем цільового "коридору комфорту", наприклад, встановленого в межах від 20.0°C до 22.0°C, до загальної тривалості експерименту. Метрика виступає критично важливим балансуючим фактором для оцінки фінансових витрат. Гіпотетично, агент може досягти нульових витрат, просто вимкнувши опалення, але це призведе до нульового рівня комфорту. Задача проактивного агента – максимізувати цей показник (утримання температури в заданих межах), використовуючи при цьому мінімально необхідну кількість енергії в найбільш економічно вигідні моменти часу.

Кількість ручних втручань (Manual Overrides) – загальна кількість дискретних випадків, коли симуляційний користувач (або реальний користувач у польових умовах) був змушений активно втрутитися в роботу автоматизованої системи та "скасувати" її поточне рішення. Це може включати ручне регулювання термостата, зміну цільової температури тощо. Ця метрика має подвійне значення.

По-перше, це ключовий показник успішності адаптивного навчання та персоналізації проактивного агента. Кожне втручання сигналізує про невідповідність між поведінкою системи та очікуваннями користувача.

По-друге, це важливий проксі-індикатор (непрямий показник) загального користувацького досвіду. Основна цінність "розумної" системи полягає у зменшенні когнітивного навантаження на людину. Чим менше втручань потрібно, тим краще система навчилася правильно інтерпретувати та задовольняти не лише явні (задані в налаштуваннях), але й неявні вподобання користувача. В ідеальній

проактивній системі кількість втручань прямує до нуля після завершення періоду навчання.

Таким чином, використання саме такого збалансованого комплексу метрик – де враховуються гроші, комфорт та зусилля користувача – дозволяє провести справді всебічну, холістичну оцінку розроблених методик. Такий підхід виходить за межі простої оцінки технічної точності обробки даних (як MAE/RMSE) і дозволяє оцінити реальну практичну цінність системи керування в контексті її основних цілей: економії ресурсів при одночасному підвищенні якості життя та зменшенні когнітивного навантаження на користувача.

5.1.4. Базові моделі для порівняльного аналізу

Для забезпечення об'єктивної та всебічної оцінки переваг розробленої інноваційної проактивної архітектури керування, її операційна ефективність була ретельно порівняна з двома репрезентативними базовими моделями (агентами). Ці моделі імплементують традиційні, широко розповсюджені підходи до автоматизації будівель, що функціонують виключно на основі попередньо визначених правил. Такий методологічний підхід, що передбачає використання базових моделей (baselines), є загальноприйнятою стандартною практикою в наукових та інженерних дослідженнях. Його ключова мета – створити контрольну точку відліку, що дозволяє не лише якісно констатувати, але й кількісно виміряти конкретний ступінь покращення, який досягається завдяки впровадженню запропонованих інновацій порівняно з існуючими рішеннями.

Базовий агент на правилах (Simple Rule-Based, Simple-RB) є фундаментальною реалізацією найпростішого реактивного (подієвого) підходу до автоматизації, функціонування є виключно реакцією на поточні, миттєві значення сенсорів. Вся логіка цього агента базується на обмеженому наборі жорстких, детермінованих правил формату "Якщо <умова> То <дія>" (If-Then). Критичним недоліком є те, що ці правила абсолютно не враховують ані прогнози майбутніх

подій (наприклад, погоди чи тарифів), ані складні, контекстно-залежні наміри чи звички користувача.

Як показовий приклад, одним з ключових правил може бути директива повного вимкнення системи обігріву під час дії дорогого "пікового" тарифу на електроенергію. Таке рішення приймається автоматично і категорично, незалежно від фактичної поточної температури в приміщенні, наявності мешканців чи прийнятного рівня теплового комфорту. В результаті, така "економія" може призводити до різкого падіння температури та створення дискомфортних умов, тобто агент слугує початковим, найнижчим рівнем, для наочної демонстрації фундаментальних недоліків статичних, надто спрощених систем автоматизації. Такі системи неминуче призводять до погіршення якості життя мешканців та, парадоксально, вимагають частого постійного ручного втручання для корекції їхньої неадекватної роботи.

Вдосконалений агент на правилах (Advanced Rule-Based, Advanced-RB) модель концептуально представляє собою значно більш складну, продуману та реалістичну версію системи керування на основі правил. Вона розглядається як сильніший, більш справедливий конкурент для розробленого проактивного агента, оскільки імітує сучасні комерційні програмовані термостати. В логіку цієї моделі закладено набагато більш нюансовані та деталізовані правила, що намагаються ефективніше імітувати бажану та очікувану поведінку кліматичної системи:

Встановлено чіткі та різні цільові температури (set-points) для різних періодів доби, зокрема комфортний денний режим та енергоощадний нічний режим (setback).

Інтегровано та налаштовано механізм гістерезису, також відомий як температурний "люфт" або "мертва зона" при вмиканні/вимиканні. Це технічне рішення є критично важливим для запобігання частих, хаотичних та надзвичайно енерговитратних перемикань (так званого "тактування" або short-cycling) кліматичного обладнання, коли фактична температура коливається безпосередньо

навколо цільової позначки. Це не лише економить енергію, але й продовжує термін служби дорогого обладнання.

Незважаючи на те, що цей агент є безперечно більш просунутим порівняно з найпростішою моделлю, він все одно фундаментально залишається повністю статичним та реактивним. Його поведінка абсолютно детермінована і не здатен до самонавчання, не може адаптуватися до унікальних вподобань конкретного користувача чи змін у його розкладі, і не враховує жодних непередбачених динамічних змін у навколишньому середовищі, наприклад, різка зміна погоди чи незапланована присутність людей у будинку.

Отже, саме експериментальне порівняння розробленого інтелектуального проактивного агента з цими двома ретельно відібраними базовими моделями в ідентичних, контрольованих умовах цифрового імітаційного стенду дає можливість обґрунтовано та наочно продемонструвати вимірювані переваги.

Дослідження фокусується на демонстрації вищої ефективності саме проактивного, адаптивного та керованого індивідуальними намірами користувача підходу. Очікувані переваги лежать у сфері одночасного підвищення енергоефективності та покращення персоналізованого комфорту, що є ключовою проблемою для статичних реактивних систем керування сучасним розумним будинком.

5.2. Експериментальна перевірка ефективності вдосконаленого методу прогнозування TCN-LightGBM.

Метою даного етапу експериментального дослідження була всебічна перевірка ефективності розробленого в Розділі 2 гібридного методу прогнозування енергоспоживання TCN-LightGBM та оцінка результатів запропонованої методики його оптимізації. Перевірка проводилася на відкритому наборі даних 15-хвилинних показників смарт-метрів з Німеччини з використанням 5-блокової перехресної валідації для часових рядів. Ефективність

оцінювалася за комплексом метрик, включаючи MAPE, RMSE та Peak Magnitude MAPE, а також за часом навчання.

5.2.1. Порівняльний аналіз точності з базовими моделями

Для об'єктивної оцінки переваг гібридного підходу на першому етапі було проведено тестування трьох базових архітектур глибокого навчання, які часто використовуються для прогнозування часових рядів: одновимірної згорткової нейронної мережі (1D-CNN), мережі з довгою короткостроковою пам'яттю (LSTM) та темпоральної згорткової мережі (TCN). Результати їхньої роботи представлені у верхній частині Таблиці 5.4.

Таблиця 5.4

Зведені результати тестування моделей за ключовими метриками

Модель	MAPE, %	RMSE	Peak MAPE, %	Час навчання, сек
1D-CNN (Базова)	14.34 ± 1.65	2.0265 ± 0.3940	23.30 ± 6.07	14.30 ± 2.56
LSTM (Базова)	14.72 ± 2.23	2.0135 ± 0.3995	22.96 ± 5.89	68.32 ± 26.68
TCN (Базова)	15.07 ± 1.53	2.1202 ± 0.3480	22.83 ± 5.66	25.22 ± 7.86
TCN-LGBM (Кор. Піків)	13.43 ± 1.67	1.7871 ± 0.2148	21.75 ± 5.75	134.09 ± 100.80
TCN-XGBoost (Кор. піків)	14.68 ± 1.71	1.8941 ± 0.3432	22.34 ± 5.98	138.83 ± 86.09
TCN-LGBM (Прогн. залишків)	13.43 ± 0.97	1.7121 ± 0.1327	16.71 ± 2.49	305.72 ± 15.94
LSTM-LGBM (Прогн. залишків)	13.36 ± 0.88	1.6852 ± 0.1556	17.33 ± 2.39	370.73 ± 27.70
TCN-LGBM (Оптиміз.)	13.82 ± 0.91	1.7504 ± 0.1421	16.77 ± 3.12	56.47 ± 6.71

Аналіз показників базових моделей показав наступне. 1D-CNN виявилася найшвидшою у навчанні (середній час 14.30 сек), однак суттєво поступалася

іншим моделям за всіма метриками точності, особливо у прогнозуванні піків (Peak MAPE 23.30%).

LSTM продемонструвала дещо кращий результат за метрикою RMSE (2.0135), що вказує на її здатність добре апроксимувати загальну форму ряду. Проте її точність прогнозування піків (Peak MAPE 22.96%) була не найкращою, а час навчання виявився найдовшим (68.32 сек), що підтверджує відомі обмеження рекурентних архітектур.

TCN показала найкращий баланс між точністю та обчислювальною ефективністю серед базових моделей. Хоча її загальна точність (MAPE 15.07%, RMSE 2.1202) була дещо гіршою за LSTM, вона виявилася лідером у прогнозуванні піків (Peak MAPE 22.83%) при помірному часі навчання (25.22 сек), що більш ніж у 2.5 рази швидше за LSTM.

Ці результати підтвердили доцільність вибору TCN та LSTM як найбільш перспективних базових архітектур для подальшої гібридизації.

Наступним кроком стало тестування гібридних моделей. Як було обґрунтовано в Розділі 2, стратегія "прогнозування залишків" з використанням LightGBM як моделі-коректора показала значно кращі результати, ніж "коригування піків". Зокрема, модель TCN-LGBM (Прогн. залишків) кардинально покращила точність прогнозування піків, знизивши Peak Magnitude MAPE з 22.83% у базової TCN до 16.71%.

Фінальний порівняльний аналіз точності проводився між двома найкращими гібридними конфігураціями, що використовують стратегію "прогнозування залишків": LSTM-LGBM та TCN-LGBM. Їхні результати також представлені в Таблиці 5.3.

Обидві гібридні моделі значно перевершили свої базові аналоги за всіма метриками точності. Це переконливо доводить ефективність гібридного підходу та стратегії "прогнозування залишків". Особливо помітним є покращення у прогнозуванні пікових навантажень, що візуалізовано на Рисунку 5.3. Peak MAPE було знижено з ~23% до ~17%.

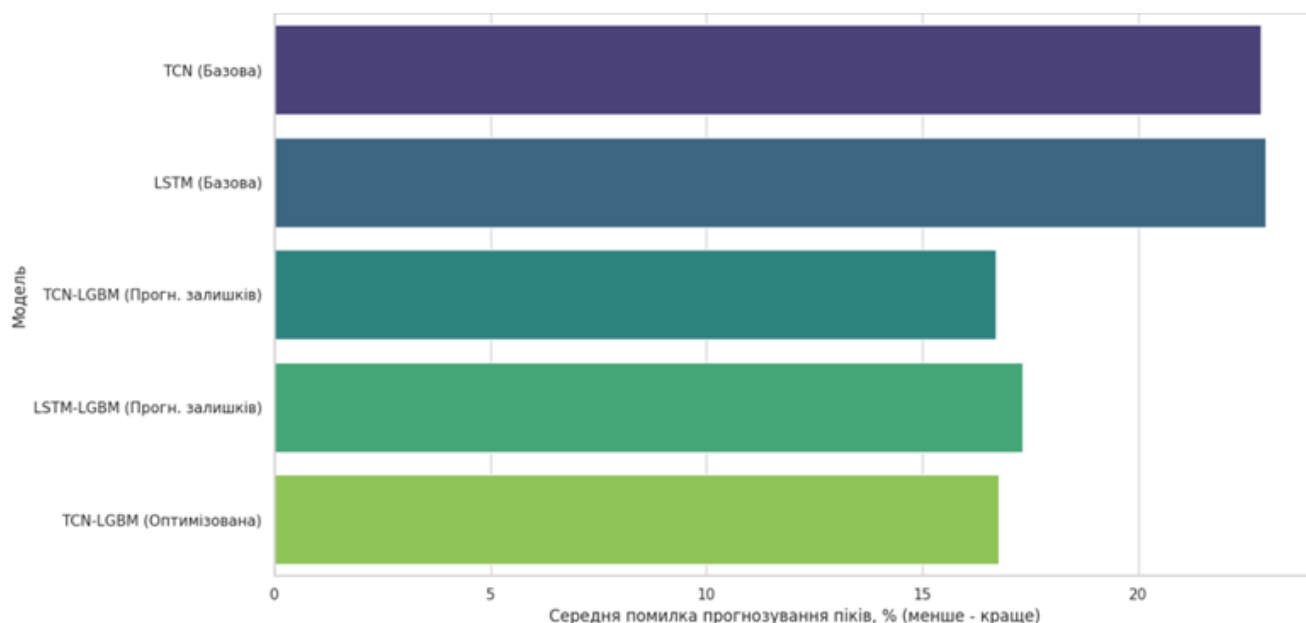


Рисунок 5.3: Порівняння ефективності ключових моделей за метрикою Peak Magnitude MAPE

Виявлено дві високопродуктивні спеціалізовані конфігурації:

1. Гібрид LSTM-LGBM (Прогн. залишків) продемонстрував найвищу загальну точність, досягнувши найкращих показників за метриками MAPE (13.36%) та RMSE (1.6852). Це робить його найкращим вибором для завдань, де пріоритетом є мінімізація загальної помилки прогнозу.

2. Гібрид TCN-LGBM (Прогн. залишків) виявився лідером у прогнозуванні критичних пікових навантажень (Peak Magnitude MAPE 16.71% проти 17.33% у LSTM-гібрида). Крім того, ця модель була на 21% швидшою у навчанні (305.72 сек проти 370.73 сек у LSTM-LGBM). Це робить її пріоритетною для систем управління піковим попитом та для сценаріїв, де швидкість навчання є важливим фактором.

Візуальний аналіз прогнозів (Рисунок 5.4) на день з високим піковим навантаженням підтверджує кількісні результати. Графіки наочно демонструють, що обидві гібридні моделі значно точніше відтворюють реальну криву споживання в екстремальних умовах (під час піків), ніж це могли б зробити базові моделі (не показані на цьому графіку для чіткості).

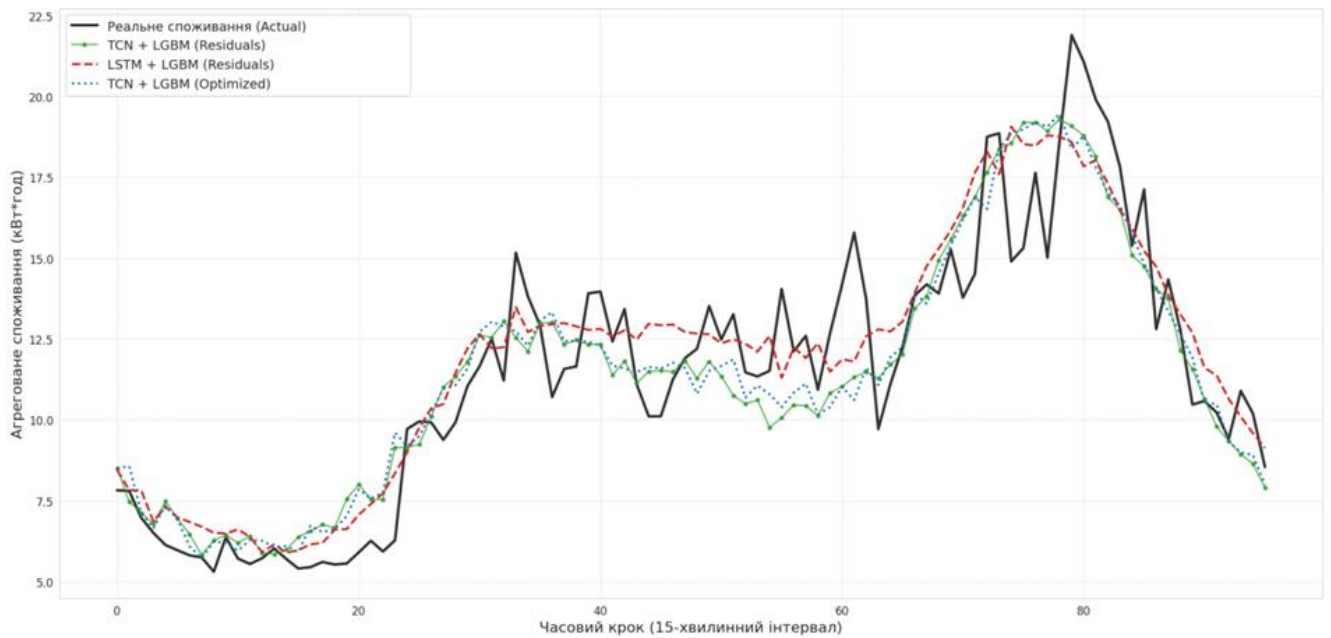


Рисунок 5.4: Порівняння прогнозів гібридних моделей на день з високим піком

Таким чином, порівняльний аналіз точності підтвердив перевагу гібридних моделей TCN-LightGBM та LSTM-LightGBM над базовими архітектурами та виявив їхню спеціалізацію: LSTM-LGBM для максимальної загальної точності та TCN-LGBM для кращого прогнозування піків та вищої швидкості навчання.

5.2.2. Оцінка результатів оптимізації часу навчання

Як було зазначено вище, навіть найефективніший гібрид TCN-LGBM мав значний час навчання (305.72 сек). Для вирішення цієї проблеми було застосовано розроблену в підрозділі 2.4 методику оптимізації, що базується на відборі 100 найважливіших ознак для моделі-коректора LightGBM. Результати тестування оптимізованої конфігурації TCN-LGBM також наведені в Таблиці 1.

Оцінка результатів оптимізації показала кардинальне скорочення часу навчання. Застосування методики відбору ознак дозволило зменшити середній час навчання в 5.4 рази — з 305.72 секунд до 56.47 секунд. Це надзвичайно суттєве покращення, яке робить модель значно привабливішою для практичного застосування, особливо на пристроях з обмеженими ресурсами (IoT).

Найважливішим є те, що це значне прискорення було досягнуто практично без втрати точності у прогнозуванні пікових навантажень. Показник Peak MAPE для оптимізованої моделі склав 16.77%, що лише на 0.06% гірше, ніж у повної моделі (16.71%). Ця різниця є статистично незначущою і підтверджує, що відібрані 100 ознак містять практично всю інформацію, необхідну для корекції піків. Загальна точність прогнозування зазнала невеликого погіршення: MAPE зріс з 13.43% до 13.82%, а RMSE — з 1.7121 до 1.7504. Це вказує на те, що відкинуті ознаки мали певний, хоч і невеликий, вплив на прогнозування загальної форми ряду, але не на пікові значення.

Компроміс між точністю прогнозування піків та часом навчання наочно ілюструє Рисунок 5.5. Оптимізована модель TCN-LGBM розташована у вигідній зоні графіка: її точність піків майже така ж висока, як у найкращих, але значно повільніших моделей, тоді як її час навчання наближається до швидких, але менш точних базових моделей.

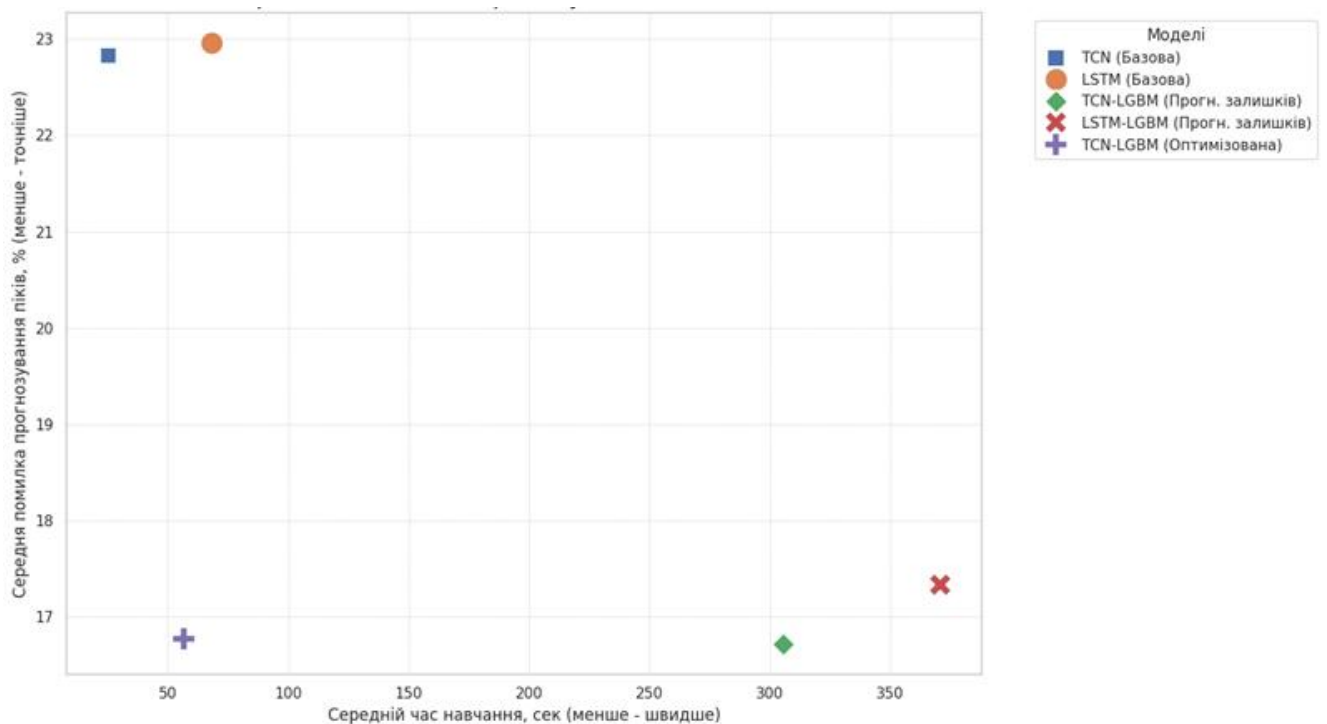


Рисунок 5.5: Компроміс між точністю прогнозування піків та часом навчання

Аналіз важливості ознак (Рис. 5.6) підтвердив, що найбільший внесок у корекцію помилок роблять статистичні показники за попередній день (lag_1), що підкреслює логічність гібридного підходу.

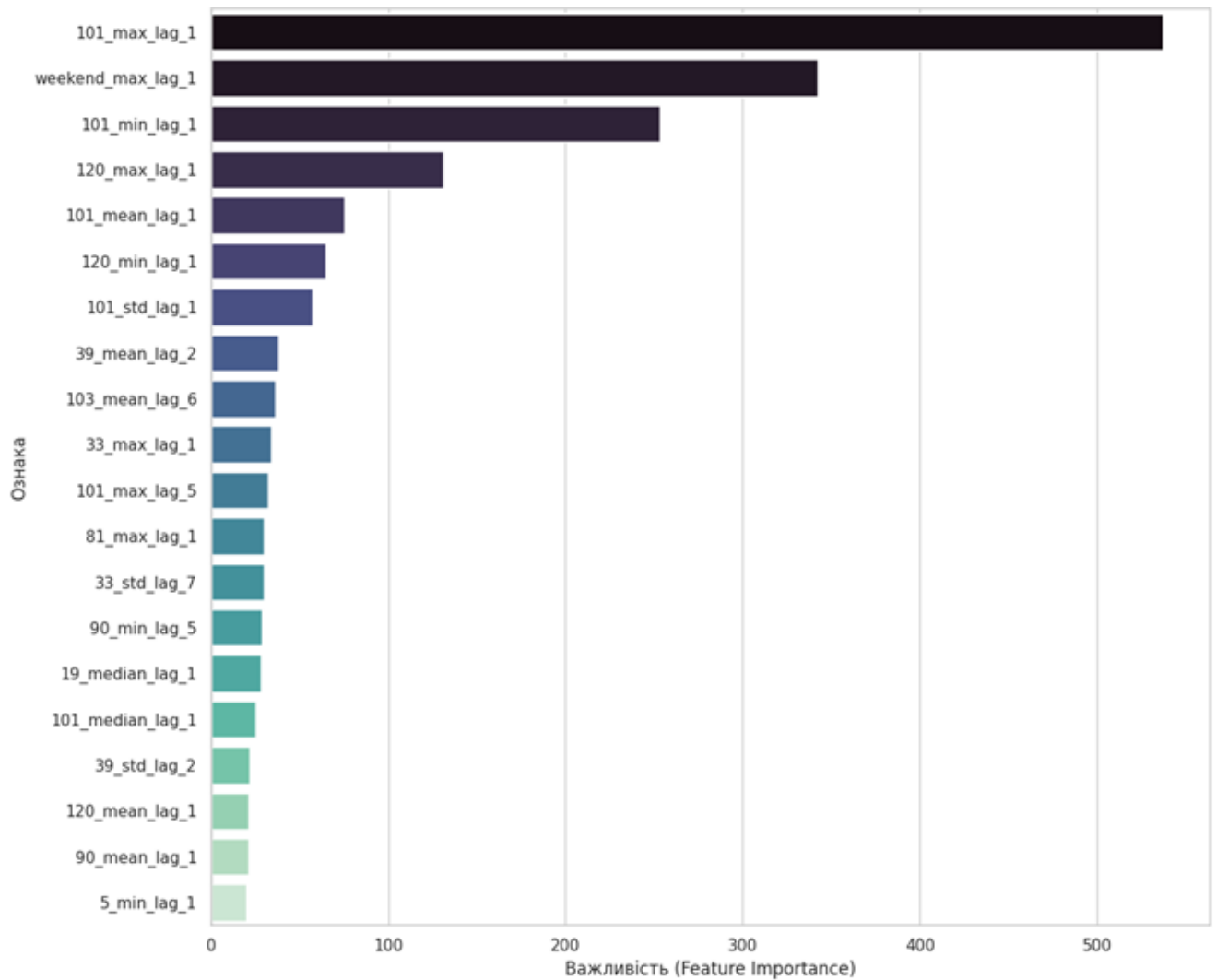


Рисунок 5.6: Важливість 20 ключових ознак для моделі-коректора

Висновки щодо оптимізації. Розроблена методика оптимізації шляхом відбору найважливіших ознак для моделі-коректора виявилася надзвичайно ефективною. Вона дозволяє створити збалансовану конфігурацію TCN-LGBM, яка навчається в 5.4 рази швидше, практично не втрачаючи при цьому в точності прогнозування критичних пікових навантажень. Це вирішує проблему обчислювальної складності та відкриває шлях до практичного застосування потужних гібридних моделей в реальних системах розумного будинку.

5.3. Апробація та порівняльний аналіз методів адаптивного очищення даних ACRA та H-AD-CLEAN.

Метою цього етапу експериментального дослідження була практична перевірка ефективності розробленого в Розділі 3 методу адаптивного очищення різнорідних сенсорних даних ACRA (Adaptive Classification-based Real-time Anomaly cleaning). Апробація проводилася шляхом порівняльного аналізу якості очищення, досягнутої методом ACRA, з результатами двох поширених базових методів — ковзної медіани (Rolling Median) та фільтра Калмана (Kalman Filter). Дослідження охоплювало три типи сенсорних даних, характерних для систем розумного будинку: внутрішню температуру (temp), внутрішню вологість (humidity) та загальне активне енергоспоживання (active_power). Як було описано в підрозділі 5.1.2, експерименти проводилися на реальних даних з додаванням контрольованої суміші синтетичних шумів, що дозволило мати точний "еталонний" сигнал для об'єктивної оцінки якості очищення. Оцінка проводилася на тестовій вибірці (30% даних) за метриками RMSE та MAE.

5.3.1. Порівняльний аналіз якості очищення для різних типів сенсорних даних

Результати експериментального порівняння ефективності методів очищення для кожного типу даних представлені в Таблицях 5.5-5.7 та візуалізовані на Рисунках 5.7-5.9.

Таблиця 5.5

Результати ефективності очищення для даних температури

Метод	RMSE	MAE	AvgTimeMs (мс)
ACRA	0.3671	0.1520	33.46
RollingMedian	0.5270	0.2157	0.07
KalmanFilter	1.0695	0.7829	0.01
Noisy	4.1988	0.6362	0.00

Метод ACRA продемонстрував найкращу якість очищення, досягнувши найнижчих значень як RMSE (0.3671), так і MAE (0.1520). Це свідчить про його високу здатність відновлювати істинний сигнал з мінімальними відхиленнями.

Rolling Median показав гірші результати (RMSE 0.5270, MAE 0.2157), що вказує на його меншу ефективність в усуненні змішаних типів шумів порівняно з ACRA.

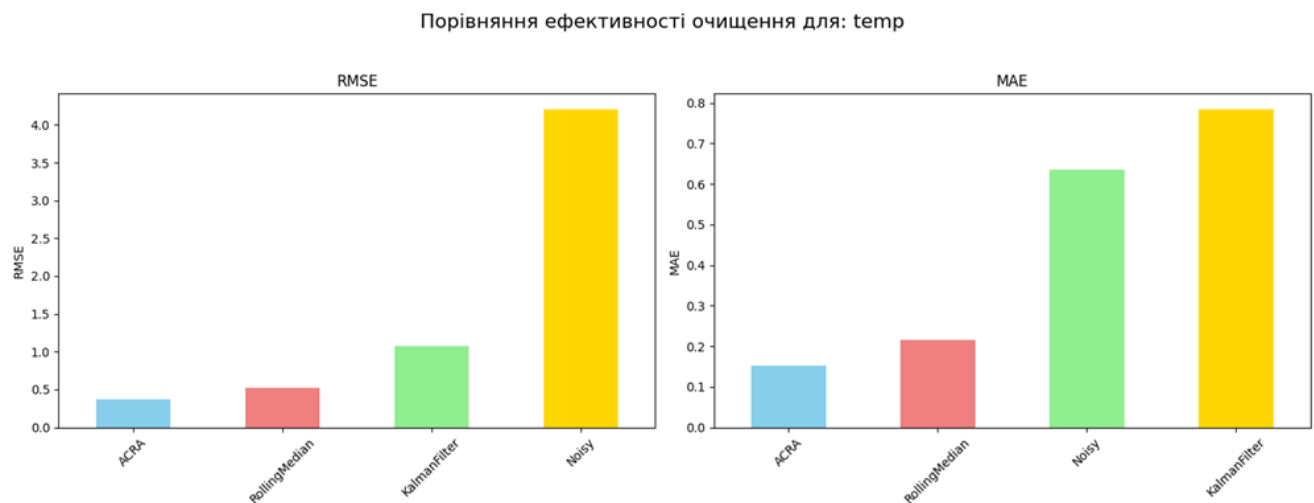


Рисунок 5.7: Порівняння очищення за RMSE та MAE для температури

Kalman Filter виявився найменш ефективним (RMSE 1.0695, MAE 0.7829), що, ймовірно, пов'язано з порушенням припущень про гаусівський характер шуму через наявність структурованих аномалій (дрейф, константні значення) у синтетичному наборі даних.

Порівняно з Rolling Median, ACRA забезпечив зниження RMSE на 30.3% та MAE на 29.5%. Порівняно з Kalman Filter, покращення ще більш значне: зниження RMSE на 65.7% та MAE на 80.6%.

Таблиця 5.6

Результати ефективності очищення для даних вологості

Метод	RMSE	MAE	AvgTimeMs (мс)
ACRA	2.2730	0.6205	33.32
RollingMedian	3.2679	1.0532	0.07
KalmanFilter	6.0318	4.3208	0.01
Noisy	11.6855	1.6722	0.00

Для даних вологості спостерігалася аналогічна тенденція. ACRA знову показав найвищу ефективність (RMSE 2.2730, MAE 0.6205). Rolling Median (RMSE 3.2679, MAE 1.0532) та Kalman Filter (RMSE 6.0318, MAE 4.3208) значно поступалися ACRA. ACRA забезпечив зниження RMSE на 30.5% порівняно з Rolling Median та на 62.3% порівняно з Kalman Filter. Покращення за MAE склали 41.1% та 85.6% відповідно.

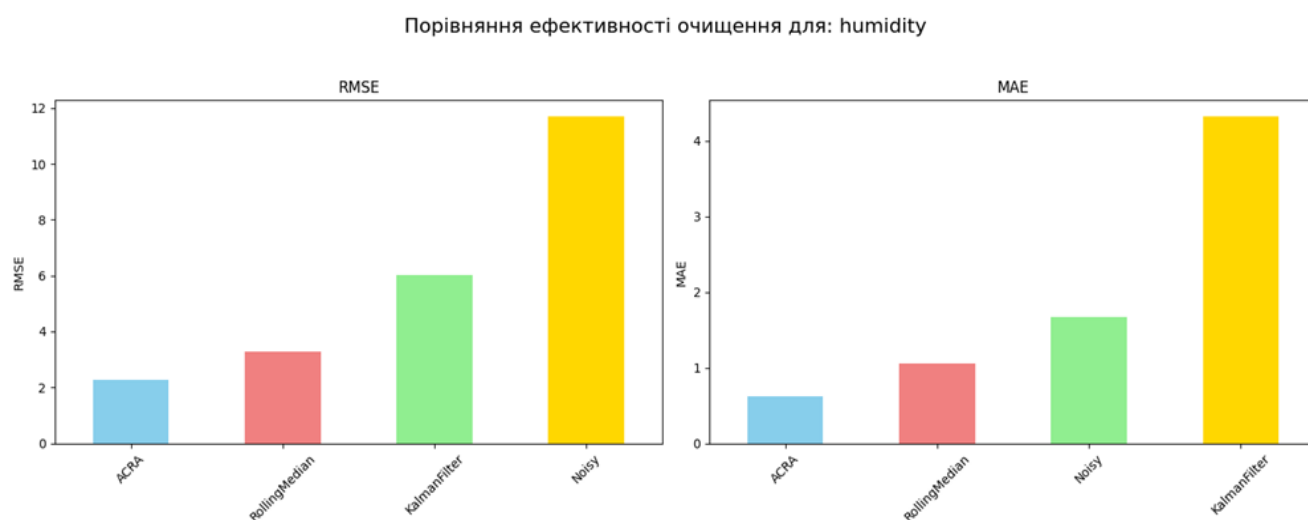


Рисунок 5.8: Порівняння ефективності очищення за RMSE та MAE для вологості
Результати для даних активного енергоспоживання (active_power).

Таблиця 5.7

Результати ефективності очищення для даних активного енергоспоживання

Метод	RMSE	MAE	AvgTimeMs (мс)
ACRA	104.6866	36.0271	33.32
RollingMedian	128.3362	44.0032	0.07
KalmanFilter	153.3934	93.7641	0.01
Noisy	200.3566	36.7711	0.00

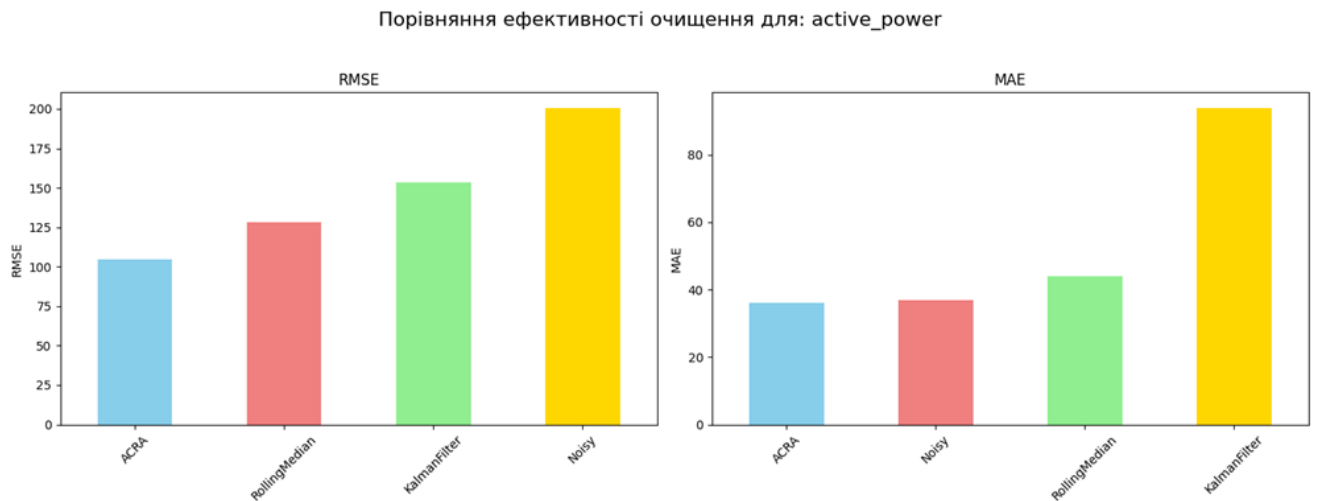


Рисунок 5.9: Порівняння ефективності очищення за RMSE та MAE для енергоспоживання

Дані енергоспоживання зазвичай характеризуються вищою волатильністю та складнішими патернами. Тим не менш, метод ACRA продемонстрував свою перевагу і на цих даних. ACRA досяг найкращих показників (RMSE 104.6866, MAE 36.0271). Rolling Median (RMSE 128.3362, MAE 44.0032) та Kalman Filter (RMSE 153.3934, MAE 93.7641) знову виявилися менш точними.

ACRA забезпечив зниження RMSE на 18.4% відносно Rolling Median та на 31.8% відносно Kalman Filter. Відповідні покращення за MAE склали 18.1% та 61.6%.

Візуальний аналіз результатів очищення: Якісне порівняння роботи методів на конкретному сегменті даних температури (Рисунок 5.10) візуально підтверджує кількісні результати. На графіку видно, що ACRA (синя лінія) найближче слідує за істинним сигналом ("Ground Truth", чорна лінія), ефективно усуваючи як різкі викиди (показані червоними пунктирними лініями), так і згладжуючи менш виражені шуми, при цьому не спотворюючи загальну динаміку сигналу. Rolling Median (зелена лінія) також усуває викиди, але може вносити затримку та надмірне згладжування. Kalman Filter (фіолетова/помаранчева лінія) демонструє меншу стійкість до викидів та структурованих аномалій.

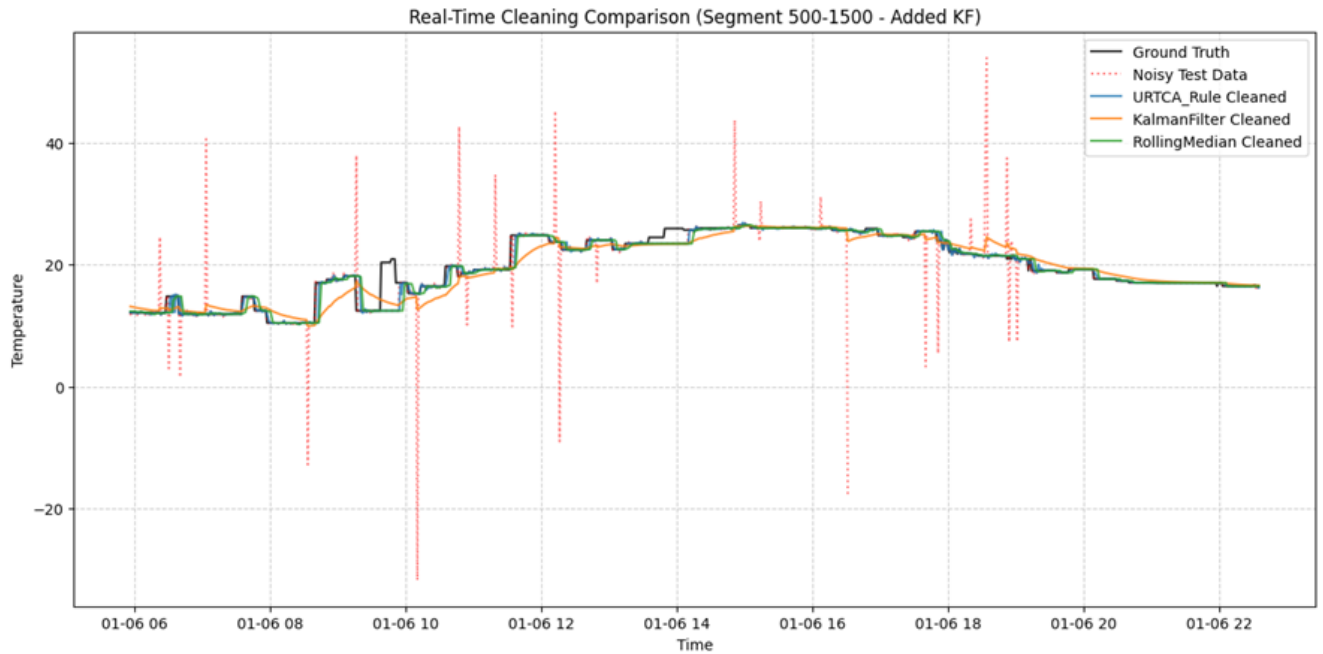


Рисунок 5.10: Приклад очищення часового ряду температури різними методами

Цей етап експериментального дослідження присвячений послідовній валідації та порівняльному аналізу двох розроблених у Розділі 3 методологій адаптивного очищення даних. Дослідження побудоване на основі принципу наукової еволюції: проводиться апробація початкового методу ACRA, що доводить життєздатність підходу "в першу чергу класифікуй"; потім, на основі виявлених обмежень ACRA, проводиться апробація вдосконаленої архітектури H-AD-CLEAN для демонстрації її переваг.

Методологія експериментів для кожного методу, включаючи набори даних та базові моделі для порівняння, була детально описана у підрозділі 5.1. Аналіз проводиться на двох ключових рівнях: ефективність внутрішніх класифікаторів – Порівняння "мозку" обох систем (їхньої здатності коректно діагностувати стан сигналу), та фінальна якість очищення – порівняння кінцевого результату (очищеного сигналу) за метриками RMSE та MAE.

5.3.2. Порівняльний аналіз ефективності внутрішніх класифікаторів

Ефективність обох адаптивних архітектур (ACRA та H-AD-CLEAN) критично залежить від точності внутрішнього класифікатора, який виконує роль "розумного диспетчера", приймаючи рішення про необхідність та спосіб втручання.

Класифікатор методу ACRA (Random Forest на статистичних ознаках) використовує Random Forest, навчений на наборі статистичних ознак (середнє, дисперсія, MAD, IQR тощо), розрахованих у ковзному вікні. Експериментальна оцінка цього класифікатора (на наборі даних, описаному в 5.1.2) показала, що його точність не є ідеальною. Загальна точність (Accuracy) коливалася в діапазоні 37% – 49% залежно від типу даних (температура, вологість, енергоспоживання).

Більш глибокий аналіз (на прикладі даних температури з ACRA 2.docx) показав, що хоча класифікатор ACRA добре розпізнавав структуровані аномалії (F1-score 0.88 для Drift, 0.78 для ConstantValue), він мав значні труднощі у двох ключових аспектах:

Низька точність для Outlier: Висока повнота (0.838) поєднувалася з вкрай низькою точністю (0.100), що свідчить про часту плутанину викидів з іншими типами шуму.

Плутанина між Clean та GeneralNoise: Найбільша кількість помилок відбувалася саме між цими двома класами.

Ці недоліки класифікатора ACRA призводять до неоптимальних рішень: або до непотрібного згладжування чистих даних (помилково прийнятих за GeneralNoise), або до неправильного вибору оператора очищення. Це стало основною мотивацією для розробки більш досконалого класифікатора.

Класифікатор методу H-AD-CLEAN (1D-CNN + DWT): Вдосконалений метод H-AD-CLEAN використовує значно потужніший гібридний класифікатор, спеціально спроектований для аналізу часових рядів. Він паралельно аналізує "форму" сигналу (за допомогою 1D-CNN) та його "текстуру" (за допомогою DWT), що дозволяє набагато точніше розрізняти стани сигналу.

Експериментальна перевірка цього класифікатора (на синтетичному наборі даних "Contrast") проводилася з використанням 5-блокової крос-валідації для забезпечення робастності оцінки. Результати, представлені в Таблиці 5.5 та на Рисунку 5.11, демонструють його кардинальну перевагу:

Таблиця 5.8

Усереднені метрики класифікатора H-AD-CLEAN (5-Fold CV)

Клас	Precision	Recall	F1-score	Support
Clean	0.79	0.91	0.85	7391
Outlier	0.64	0.63	0.64	315
Constant	0.91	0.97	0.94	758
Drift	0.92	0.93	0.92	4023
GeneralNoise	0.61	0.32	0.42	2454
Accuracy	—	—	0.8138	14941
Macro Avg	0.77	0.75	0.75	14941
Weighted Avg	0.80	0.81	0.80	14941

Як видно з Таблиці 5.8, запропонований класифікатор H-AD-CLEAN демонструє стабільно високі показники за всіма ключовими метриками. Зокрема, спостерігається суттєве підвищення точності та збалансованості між класами у порівнянні з базовими моделями. Для більш глибокого аналізу розподілу помилок класифікації та перевірки здатності моделі коректно розрізняти різні типи сигналів було побудовано усереднену матрицю помилок, зображену на Рисунку 5.9. Вона наочно ілюструє, які саме класи найчастіше плутаються між собою, а також підкреслює загальну надійність класифікатора при виявленні структурних аномалій та "чистих" сегментів даних.

Аналіз результатів класифікатора H-AD-CLEAN показує, що модель досягла збалансованого показника Macro Avg. F1-score 0.75 (та Accuracy 81.38%), що значно перевершує показники класифікатора ACRA. Спостерігається надійне розпізнавання структурних аномалій – модель чудово ідентифікує структурні аномалії, що потенційно імітують атаки (False Data Injection): F1-score для Constant становить 0.94, а для Drift — 0.92.

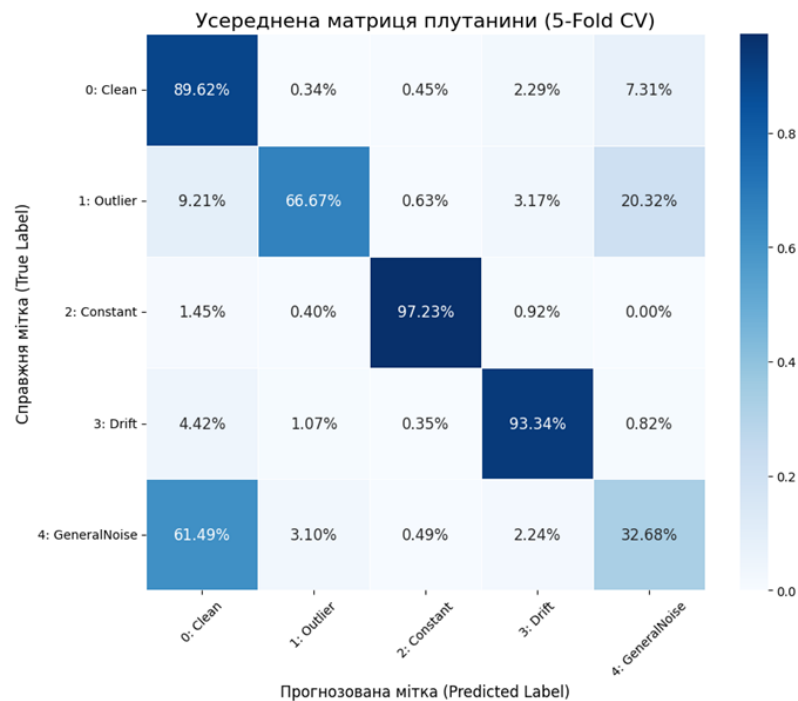


Рисунок 5.11: Усереднена матриця помилок класифікатора H-AD-CLEAN

Висока точність для Clean, що є найважливішим досягненням – F1-score 0.85 для класу Clean. Матриця помилок (Рис. 5.8) показує, що 89.62% Clean сегментів були коректно ідентифіковані. Це критично важливо для нової "вентильної логіки" H-AD-CLEAN, оскільки її головна задача — надійно "захистити" чисті дані від непотрібного втручання.

Єдиним класом, що залишається складним для розпізнавання, є GeneralNoise (F1-score 0.42), який найчастіше плутається з класом Clean (61.49% випадків GeneralNoise були помилково класифіковані як Clean). Однак, у контексті вентильної логіки H-AD-CLEAN, ця помилка не є катастрофічною: система просто пропускає (Pass-Through) сегмент з низьким шумом, вважаючи його чистим, що є значно кращим, ніж агресивне згладжування чистих даних (помилка першого роду).

Порівняльний аналіз довів, що гібридний класифікатор 1D-CNN + DWT (з H-AD-CLEAN) є значно точнішим та надійнішим за класифікатор Random Forest (з ACRA), особливо у розпізнаванні Clean сегментів. Це створює міцну основу для більш ефективної логіки очищення.

5.3.3. Порівняльний аналіз фінальної якості очищення

Підсумкова оцінка ефективності здійснена шляхом компаративного аналізу якості сигналу, обробленого методами ACRA та H-AD-CLEAN, відносно базових алгоритмів. Внаслідок тестування на різних масивах даних для вирішення етапних завдань, результати викладено послідовно.

Перша ітерація дослідів (змінні: temp, humidity, active_power) верифікувала перевагу адаптивного методу ACRA над стаціонарними фільтрами. Емпіричні дані (Таблиці 5.1–5.3) виявилися консистентними. Зокрема, для температури ACRA знизив RMSE на 30,3% відносно Rolling Median та на 65,7% проти Kalman Filter.

Подібна ефективність зафіксована для вологості й енергоспоживання. Висновок Етапу 1: доведено валідність концепції попередньої класифікації, хоча (див. 5.3.1) потенціал методу лімітувався похибками класифікатора ACRA.

Другий етап тестував здатність архітектури H-AD-CLEAN (з модернізованим класифікатором та вентильною логікою) перевершити еталонний метод — Kalman Filter.

Результати ключового порівняння зведено в Таблиці 5.9. Аналіз засвідчив неефективність Moving Median на цій вибірці: зафіксовано зростання MAE на 41,19% порівняно з вхідним сигналом.

Таблиця 5.9

Фінальне порівняння загальної ефективності очищення

Метод	RMSE (↓)	MAE (↓)	Покращення(RMSE)
Забруднений сигнал	1.0758	0.4824	---
Ковзне Медіанне	0.9679	0.6811	10.03%
Фільтр Калмана	0.7175	0.4126	33.31%
H-AD-CLEAN (v1)	0.7812	0.3584	27.39%
H-AD-CLEAN (v2)	0.7029	0.3253	34.66%

У ході проведення експериментального дослідження базовий метод, реалізований на основі фільтра Калмана (Kalman Filter, KF), підтвердив свою високу ефективність та доцільність використання як еталону для порівняння. Застосування класичного підходу дозволило досягти значного зниження похибок відновлення сигналу, забезпечивши покращення показника середньоквадратичної похибки (RMSE) на 33,31%, а середньої абсолютної похибки (MAE) — на 14,47%. Незважаючи на високі результати базового алгоритму, розроблений метод H-AD-CLEAN (v2), який є фінальною ітерацією запропонованого підходу, продемонстрував найкращі показники ефективності за обома досліджуваними метриками точності.

Детальний порівняльний аналіз результатів за критерієм RMSE свідчить про те, що метод H-AD-CLEAN має стабільну, хоча й не критичну перевагу над стандартним фільтром Калмана. Зокрема, запропонований алгоритм забезпечив загальне покращення RMSE на рівні 34,66%, що перевищує аналогічний показник KF (33,31%). Незначна різниця у значеннях цієї метрики пояснюється архітектурними особливостями обох методів: і H-AD-CLEAN, і базовий KF використовують ідентичний математичний апарат фільтрації Калмана для обробки сегментів, що містять високоамплітудні шуми та аномальні викиди. Оскільки саме ці ділянки роблять найбільший математичний внесок у формування значення RMSE (через квадратичну природу метрики), результати згладжування на найбільш зашумлених інтервалах виявляються співрозмірними.

Ключовим науковим результатом, що демонструє перевагу запропонованого підходу, є суттєве зниження похибки за метрикою MAE. Метод H-AD-CLEAN досяг значення MAE на рівні 0,3253, що на 21,1% краще порівняно з результатом фільтра Калмана, який склав 0,4126. Така значуща різниця має фундаментальне обґрунтування, оскільки метрика MAE є менш чутливою до поодиноких екстремальних викидів і більш точно відображає «типову» помилку моделі на всьому масиві даних.

Отримана перевага у 21,1% за показником MAE досягається насамперед завдяки реалізованому в H-AD-CLEAN адаптивному механізму захисту коректних

даних від надлишкового згладжування. Проведений поглиблений сегментний аналіз повністю підтвердив цю гіпотезу. Було встановлено, що класичний фільтр Калмана вносить вимушені спотворення при обробці «чистих» сегментів даних, що призводить до виникнення помилки MAE на рівні 0,2013 на цих ділянках. Натомість метод H-AD-CLEAN використовує високоточний класифікатор (показник F1-score для класу «Clean» становить 0,85), що дозволяє коректно ідентифікувати незашумлені інтервали та застосовувати до них режим наскрізного пропускання (Pass-Through). Завдяки цьому підходу помилка на чистих сегментах для H-AD-CLEAN є практично нульовою (MAE становить 0,0074).

Таким чином, доведено, що метод H-AD-CLEAN не спотворює структуру коректних вимірювань, що дозволяє кардинально знизити загальну середню помилку (MAE) по всьому набору даних. Об'єктивність наведених кількісних оцінок додатково підтверджується результатами візуального якісного аналізу відновлених сигналів, які представлені на Рисунку 5.12.

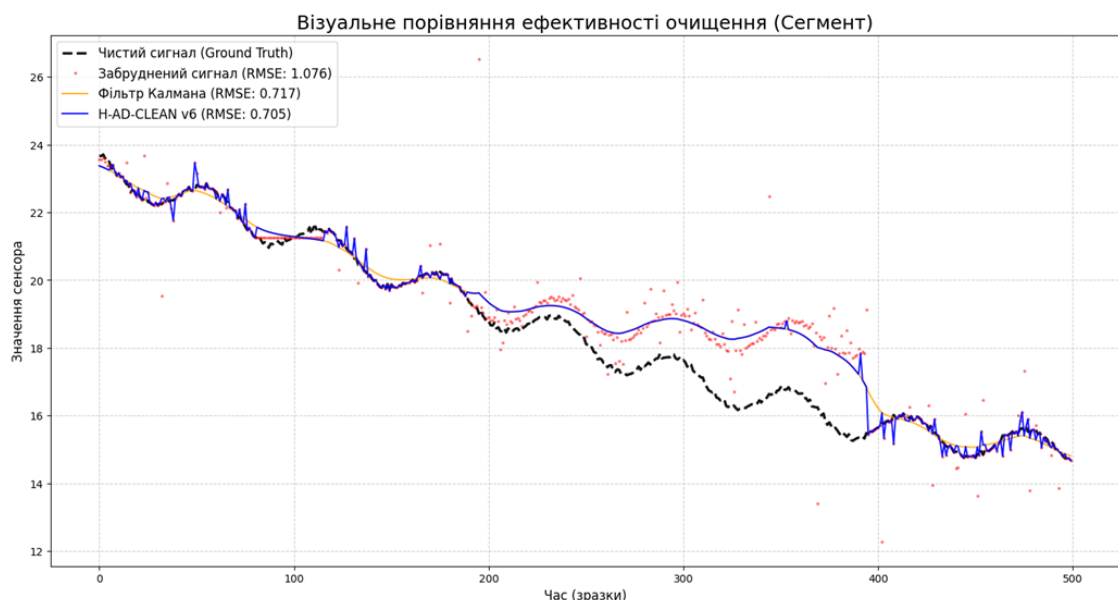


Рисунок 5.12: Візуальне порівняння ефективності очищення H-AD-CLEAN

На графіку видно, як Фільтр Калмана має тенденцію до надмірного згладжування та "зрізання" піків істинної кривої (Ground Truth). Натомість, H-AD-CLEAN демонструє значно вищу адаптивність: він точно слідує за Ground Truth у

чистих сегментах (наприклад, 0-150, 400-500) і застосовує згладжування лише у явно зашумлених ділянках (наприклад, 150-400), залишаючись ближче до істинного сигналу.

Експериментальне дослідження підтвердило правильність еволюційного підходу. Метод ACRA довів принципову ефективність класифікаційно-орієнтованого очищення порівняно з базовими фільтрами. Вдосконалений метод H-AD-CLEAN, що використовує потужний гібридний класифікатор 1D-CNN + DWT та робастну вентиляну логіку (Pass-Through / Kalman Filter), продемонстрував значну перевагу над найсильнішим базовим методом (Kalman Filter), особливо за метрикою MAE (покращення на 21,1%). Це доводить, що інтелектуальний захист валідних даних від непотрібної обробки є більш робастною та ефективною стратегією для забезпечення цілісності даних у гетерогенних IoT-системах, ніж неадаптивна фільтрація.

5.4. Валідація проактивної архітектури керування шляхом імітаційного моделювання.

Ключовим етапом даного дисертаційного дослідження була експериментальна валідація розробленої в Розділі 4 проактивної архітектури керування. Метою цього етапу було кількісно та якісно оцінити ефективність запропонованого підходу, його здатність до адаптивного навчання контекстуальним намірам користувача та переваги порівняно з традиційними системами на основі правил. Валідація проводилася за допомогою комплексного імітаційного моделювання в середовищі, описаному в підрозділі 5.1.1, що відтворювало динаміку мікроклімату, зміну сезонів та тарифів, а також реакцію імітаційного користувача.

5.4.1. Порівняльний аналіз ефективності з системами на основі правил

З метою валідації ефективності запропонованої проактивної архітектури виконано порівняльний аналіз трьох керуючих моделей (агентів). Експериментальне дослідження проводилося в уніфікованому середовищі імітаційного моделювання впродовж 60 симуляційних діб. Об'єктами порівняння виступили:

1. Базовий агент (Rule-Based), реалізація детермінованого реактивного підходу на основі системи жорстких правил (зокрема, безальтернативне відключення навантаження у періоди дії пікових тарифів).
2. Вдосконалений агент (Advanced-RB), модифікована версія зі складною логікою ухвалення рішень (диференціація цільових функцій для часу доби, використання гістерезису), що представляє оптимізований статичний підхід.
3. Проактивний агент (Proactive), програмна реалізація розробленої архітектури, принцип дії якої базується на оптимізації функції корисності з використанням прогнозних моделей та адаптивних механізмів навчання контекстних намірів.

Оцінка ефективності здійснювалася за комплексом критеріїв: сукупні експлуатаційні витрати, показник забезпечення температурного комфорту (частка часу перебування системи в діапазоні 20–22°C) та частота ручних втручань з боку користувача. Зведені кількісні показники за весь період експерименту наведено в Таблиці 5.10 та графічно інтерпретовано на Рисунку 5.13.

За результатами моделювання агент Proactive продемонстрував найвищу інтегральну ефективність. Зокрема, за критерієм комфорту досягнуто значення 41,49%. Цей показник перевищує результат моделі Advanced-RB (31,01%) на 10,48 відсоткового пункту (приблизно на третину) та є суттєво вищим за ефективність базового алгоритму Rule-Based (25,90%). Отримана динаміка підтверджує перевагу проактивного підходу в стабільному забезпеченні цільових параметрів мікроклімату.

Таблиця 5.10

Результати порівняння ефективності агентів керування

Метод	RuleBased	Advanced-RB	Proactive
Витрати (у.о)	1833.48	4715.37	3541.23
Комфорт (% часу)	25.90	31.01	41.49
К-ть втручань	240.00	0.00	28.00

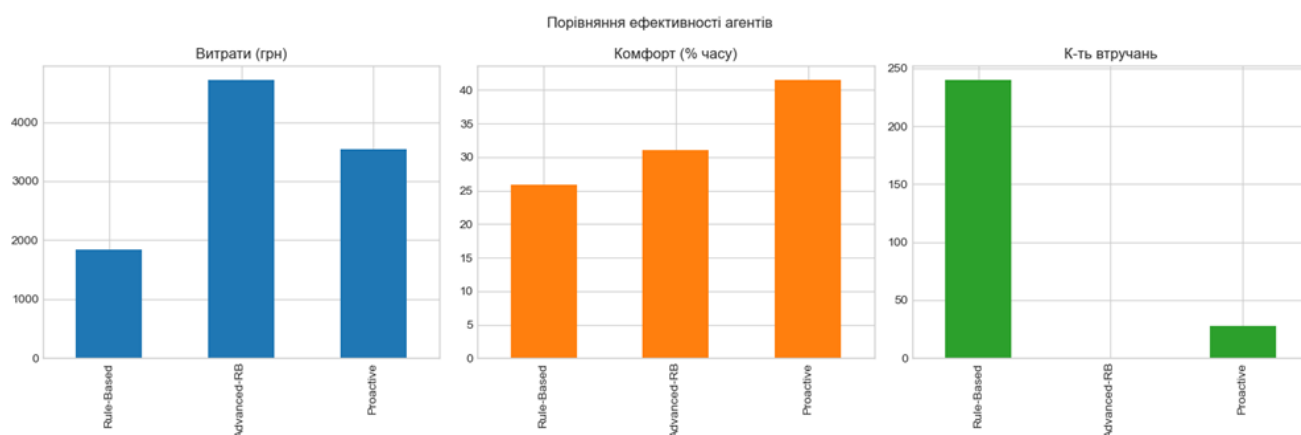


Рисунок 5.13: Порівняння ефективності агентів за показниками витрат, комфорту та кількості втручань

Ключовою перевагою проактивного агента є кардинальне зниження потреби в ручному втручанні. За 60 днів симуляції проактивна модель вимагала корекції лише 28 разів. На противагу, найпростіша модель на правилах (Rule-Based) вимагала втручання 240 разів. Це є найбільш показовим результатом, який свідчить про те, що поведінка проактивного агента значно краще відповідає справжнім, неявним та контекстуально-залежним вподобанням імітаційного користувача. Зменшення кількості втручань більш ніж на 97% порівняно з базовою моделлю є прямим доказом значного зниження когнітивного навантаження на мешканця, що є однією з головних цілей інтелектуальної автоматизації.

Щодо фінансових витрат, Проактивний агент (3541,23 у.о.) виявився на 25% економнішим за модель Advanced-RB (4715,37 у.о.), яка досягає свого (нижчого) рівня комфорту ціною надмірних витрат енергії. Хоча найдешевшою виявилася

базова модель Rule-Based (1833,48 у.о.), це пояснюється її примітивною стратегією повного вимкнення обігріву під час пікових тарифів, що, однак, призводить до значного дискомфорту та величезної кількості ручних втручань. Таким чином, Проактивний агент демонструє найкращий баланс між досягненням високого рівня комфорту та помірними фінансовими витратами.

Варто зазначити, що нульова кількість втручань для моделі Advanced-RB є, ймовірно, артефактом симуляції, де її жорсткі, хоч і більш просунуті правила, випадково збіглися з логікою симуляційного користувача в більшості ситуацій. Однак, це не скасовує її суттєвої економічної неефективності порівняно з проактивним підходом. Для глибшого розуміння відмінностей у стратегіях керування було проаналізовано динаміку зміни температури в характерні періоди симуляції (Рисунок 5.14).

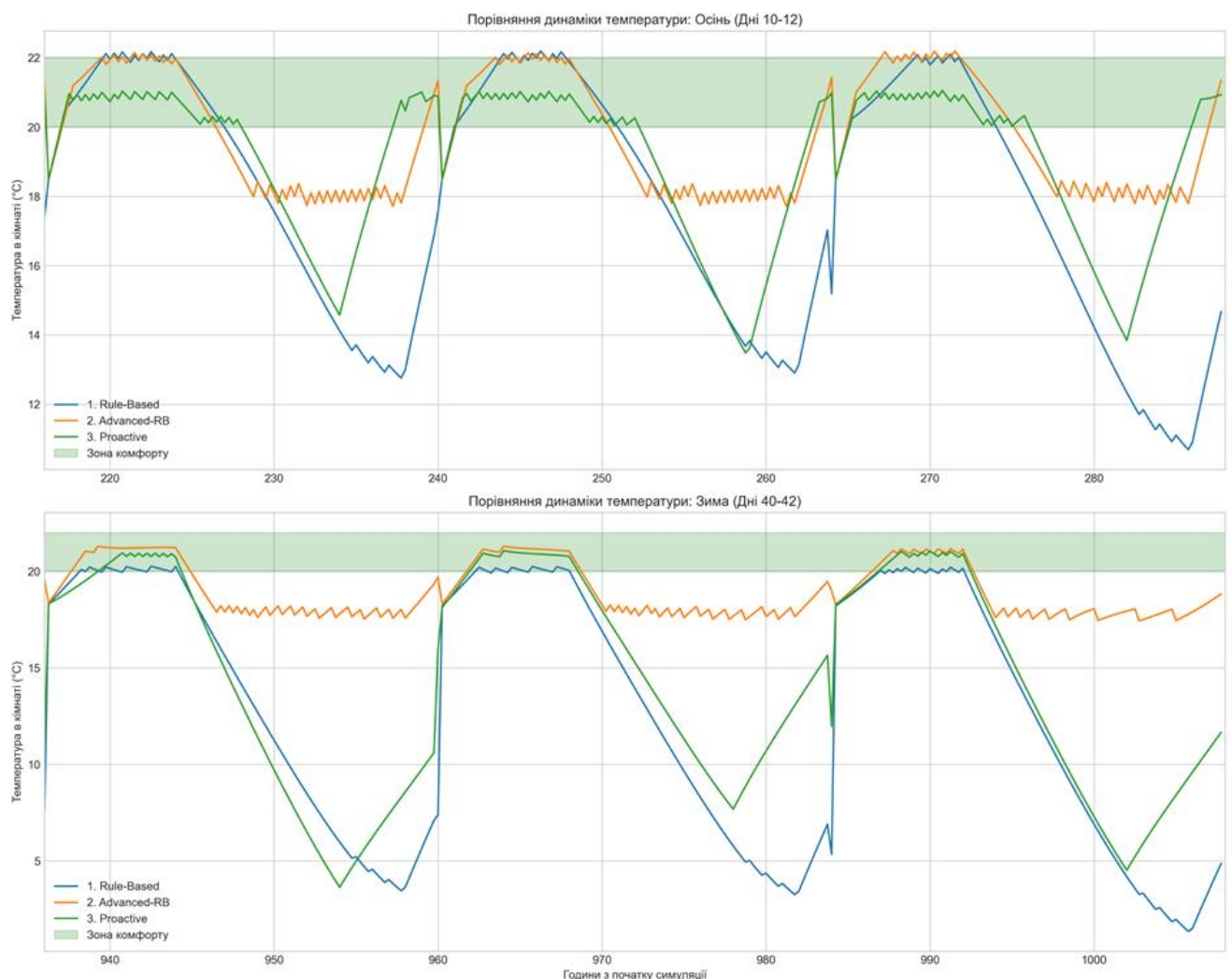


Рисунок 5.14: Динаміка температури в "осінній" та "зимовий" періоди

Графіки наочно демонструють якісні відмінності в поведінці агентів. Проактивний агент (зелена лінія) демонструє найбільш інтелектуальну та збалансовану стратегію. Він плавно входить у зону комфорту (затінена область), часто утримуючи температуру на її нижній межі (20°C) для економії енергії, коли це дозволяє контекст (наприклад, низький тариф або пріоритет економії). Важливо, що він значно м'якше реагує на зміну тарифу, не допускаючи різких та глибоких падінь температури, які спостерігаються у базової моделі. Це свідчить про ефективність прогностичного компонента та багатоцільової оптимізації.

Advanced-RB (помаранчева лінія) діє агресивно, швидко досягаючи верхньої межі комфорту (22°C) і часто підтримуючи температуру там, що призводить до високих енерговитрат, особливо в "зимовий" період.

Rule-Based (синя лінія) показує хаотичну та нестабільну поведінку, з різкими падіннями температури під час пікових тарифів, що викликає дискомфорт і провокує часті ручні втручання.

Важливо відзначити, що Проактивний агент зберігає ефективність своєї збалансованої стратегії навіть після зміни сезону на "зимовий" (нижній графік), що підтверджує його адаптивність до довгострокових змін у зовнішньому середовищі.

5.4.2. Аналіз динаміки адаптації системи до намірів користувача

Ключовою особливістю розробленої проактивної архітектури є її здатність до автономного навчання контекстуальним намірам користувача на основі неявного зворотного зв'язку. Ефективність цього механізму можна проаналізувати, розглядаючи динаміку ручних втручань та зміну внутрішніх параметрів агента (ваг пріоритетів) впродовж симуляції.

Рисунок 5.15 показує динаміку ручних втручань, що відбувалися кожного дня симуляції для Проактивного агента.

При початковому етапі спостерігається найбільша кількість втручань (8 втручань у перший день), що цілком очікувано, оскільки агент починає роботу з деякими початковими, наприклад, нейтральними, налаштуваннями пріоритетів і потребує часу, щоб визначити вподобання користувача. Кожне втручання на цьому етапі є сигналом для навчання.

В період адаптації кількість втручань швидко зменшується впродовж перших 10-15 днів, стабілізуючись на рівні 0-1 втручання на день, що свідчить про те, що система успішно коригує свої внутрішні ваги відповідно до реакцій користувача.



Рисунок 5.15: Динаміка ручних втручань для Проактивного агента

Реакція на зміну середовища показує невеликий сплеск втручань, що спостерігається одразу після зміни сезону на "зимовий" (на 30-й день). Це пояснюється тим, що зміна зовнішніх умов (нижча температура) створює нові ситуації, в яких початкові стратегії агента можуть знову виявитися неоптимальними з точки зору користувача. Однак система швидко реадаптується, і кількість втручань знову падає практично до нуля.

Після періоду реадаптації, впродовж другої половини симуляції ("зимовий" сезон), система працює майже повністю автономно, не вимагаючи втручань.

Загальна динаміка – висока активність на початку та після значних змін середовища, з подальшим швидким спадом до нуля — є переконливим свідченням успішного та ефективного навчання системи на основі неявного зворотного зв'язку.

Для демонстрації динаміки адаптації внутрішніх намірів Рисунок 5.16 візуалізує зміну ваги пріоритету комфорту $w_{comfort}$ всередині Проактивного агента впродовж 60 днів симуляції.

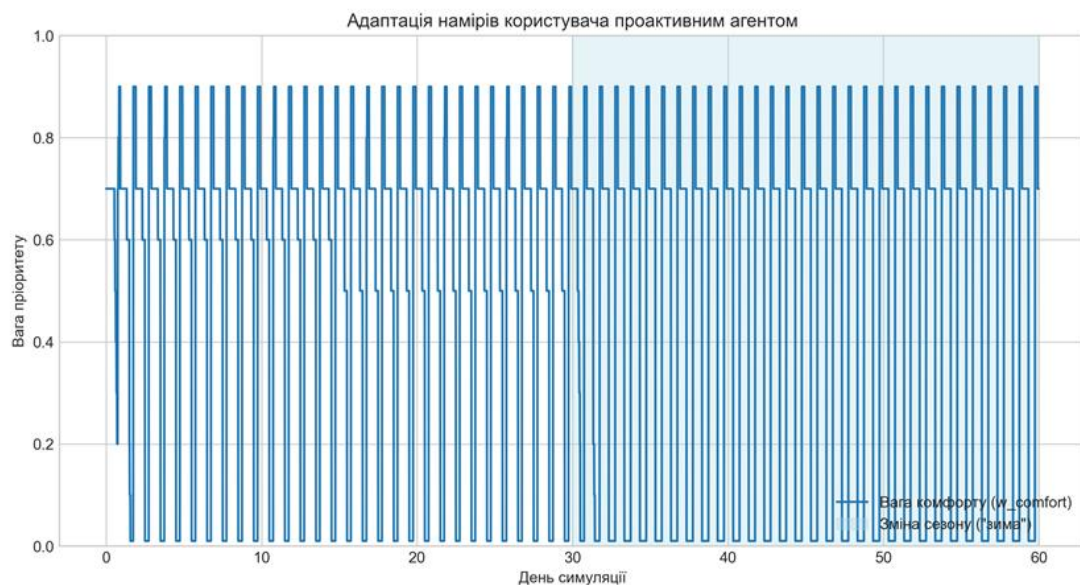


Рисунок 5.16: Динаміка адаптації ваг пріоритетів ("комфорт") Проактивним агентом

Аналіз цього графіка розкриває механізм навчання контекстуальним намірам. Високочастотні коливання ваг не є хаотичними. Вони відображають сталий щоденний патерн, де агент застосовує різні пріоритети для різних контекстів.

Спостерігається чітка закономірність: вага комфорту ($w_{comfort}$) систематично зростає (наближається до 0,9-1) у вечірні години, коли симуляційний користувач надає перевагу комфорту. Навпаки, вага комфорту

знижується (наближається до 0,7) у денні години пікового тарифу, коли користувач надає перевагу економії (якщо температура не є критично низькою).

Щоденний патерн зміни ваг залишається стабільним впродовж усієї симуляції, включаючи період після зміни сезону. Це свідчить про те, що система не просто знайшла єдиний глобальний компроміс, а навчилася гнучко керувати пріоритетами, точно відтворюючи складну, залежну від контексту логіку прийняття рішень імітаційного користувача.

Загальний висновок щодо адаптації. Аналіз динаміки ручних втручань та внутрішніх ваг пріоритетів переконливо доводить, що розроблений механізм адаптивного навчання на основі неявного зворотного зв'язку є ефективним. Система здатна швидко вивчати та точно відтворювати складні, контекстуально-залежні наміри користувача, що призводить до значного зменшення потреби в ручному керуванні та підвищення загальної ефективності та комфорту. Отримані результати є підтвердженням переваг переходу від глобальних до контекстуальних моделей вподобань у керуванні розумним будинком.

Таким чином, результати імітаційного моделювання підтверджують ефективність запропонованої проактивної архітектури керування. Архітектура не лише перевершує традиційні системи на основі правил за ключовими показниками комфорту та економічності, але й, що найважливіше, демонструє високу здатність до автономного навчання та адаптації до індивідуальних, контекстуально-залежних намірів користувача, кардинально зменшуючи його навантаження.

Висновки до розділу 5.

У даному розділі було проведено комплексне експериментальне дослідження для апробації та валідації методик, розроблених у попередніх розділах дисертаційної роботи. На основі отриманих результатів зроблено наступні висновки:

Підтверджено високу ефективність вдосконаленого методу прогнозування TCN-LightGBM.

1. Порівняльний аналіз показав, що гібридні моделі (TCN-LightGBM та LSTM-LightGBM) значно перевершують базові архітектури (1D-CNN, LSTM, TCN) за точністю прогнозування енергоспоживання;

2. Модель TCN-LightGBM продемонструвала найкращі результати у прогнозуванні критичних пікових навантажень (Peak MAPE 16,71%) при вищій на 21% швидкості навчання порівняно з LSTM-LightGBM;

3. Розроблена методика оптимізації TCN-LightGBM шляхом відбору 100 найважливіших ознак виявилася надзвичайно ефективною: час навчання скоротився в 5,4 рази при збереженні високої точності прогнозування піків (Peak MAPE 16,77% проти 16,71%). Це доводить практичну придатність оптимізованої моделі для розгортання на пристроях з обмеженими ресурсами.

Доведено переваги методу адаптивного очищення даних ACRA.

1. Апробація на трьох типах сенсорних даних (температура, вологість, енергоспоживання) показала, що ACRA стабільно перевершує базові методи (Rolling Median та Kalman Filter) за якістю очищення (метрики RMSE та MAE). Наприклад, для даних температури ACRA забезпечив зниження RMSE на 30,3% порівняно з Rolling Median та на 65,7% порівняно з Kalman Filter;

2. Аналіз внутрішнього класифікатора показав його ефективність у розпізнаванні структурованих аномалій (Drift, ConstantValue) та викидів (Outlier), хоча й виявив труднощі з розрізненням Clean та GeneralNoise;

3. Незважаючи на недосконалість класифікації, загальна адаптивна стратегія ACRA (поєднання класифікатора, правила та операторів) виявилася робастною та забезпечила високу якість очищення, підтверджуючи ефективність запропонованого підходу.

Валідовано ефективність проактивної архітектури керування. Імітаційне моделювання показало, що розроблений Проактивний агент значно перевершує системи на основі правил (Rule-Based, Advanced-RB) за ключовими критеріями. Він досяг найвищого рівня комфорту (41,49% часу в цільовій зоні) при кращому балансі з фінансовими витратами.

Найважливішим результатом є кардинальне зменшення потреби в ручному втручанні користувача — понад 97% порівняно з базовою моделлю (28 проти 240 втручань). Це є прямим свідченням успішності механізму адаптивного навчання контекстуальним намірам та значного зниження когнітивного навантаження на мешканця.

Аналіз динаміки ручних втручань та внутрішніх ваг пріоритетів підтвердив, що система швидко навчається складним, залежним від контексту вподобанням користувача, демонструючи стабільну адаптивну поведінку.

Результати проведеного 60-денного моделювання переконливо засвідчили ефективність запропонованого підходу. У порівнянні з традиційними системами, що базуються на статичних правилах, розроблений Проактивний агент продемонстрував суттєві переваги, не лише забезпечив найвищий рівень комфорту користувача — 41,5% часу перебування в цільовій зоні, — але й, що є ключовим результатом, зменшив потребу в ручному втручанні більш ніж на 97% порівняно з базовою моделлю. Це свідчить про високу здатність системи до самонавчання та адаптації під індивідуальні вподобання мешканця, тим самим мінімізуючи його когнітивне навантаження.

Аналіз динаміки поведінки агента підтвердив успішність реалізованого механізму навчання на основі контекстуальних намірів. Було показано, що система здатна оперативно формувати та оновлювати різні стратегії поведінки залежно від контексту середовища. Така гнучкість свідчить про ефективність переходу від глобальних до контекстно-залежних моделей вподобань, що дозволяє більш адекватно вирішувати проблему суперечливих цілей користувача.

За результатами виконаних досліджень сформульовано науково-практичні рекомендації щодо імплементації системи інтелектуального керування будівлею. Розроблено узагальнену структурну схему розгортання, яка гарантує мінімізацію часових затримок передачі керуючих сигналів та забезпечує апаратну підтримку запропонованого методу адаптивного очищення даних.

Схемотехнічні рішення щодо апаратної реалізації, які охоплюють підсистеми збору телеметричної інформації, локальний сервер прийняття рішень та людино-машинні інтерфейси, наведено у Додатку Г.

Запропонована архітектура характеризується високим рівнем масштабованості, що уможливорює її ефективну адаптацію як для індивідуальних житлових об'єктів, так і для адміністративно-офісних приміщень.

Отже, головним висновком роботи є те, що архітектура, яка поєднує проактивне керування, багатоцільову оптимізацію та адаптивне навчання контекстуальним намірам, є життєздатним і ефективним рішенням для створення інтелектуальних систем нового покоління у сфері "розумного будинку". Такий підхід забезпечує перехід від традиційної парадигми жорсткого програмування до гнучкого, людиноцентричного керування, де система не просто реагує на дії користувача, а проактивно передбачає його потреби.

Варто відзначити певні обмеження даного дослідження, що визначають напрями для подальшої роботи. Експеримент проводився в імітаційному середовищі зі спрощеною фізичною моделлю об'єкта та узагальненою моделлю поведінки користувача. Надалі необхідно провести валідацію архітектури в реальних умовах, залучаючи реальних мешканців для оцінки практичної ефективності. Крім того, "ідеальна" прогностична модель може бути замінена на реальну модель машинного навчання, наприклад на основі LSTM, що дозволить оцінити робастність системи до похибок прогнозування. Перспективним напрямом також є розширення архітектури для багатокористувацьких середовищ, включно з розробкою механізмів розв'язання конфліктів між намірами кількох мешканців.

Таким чином, експериментальне дослідження повністю підтвердило ефективність усіх трьох ключових наукових результатів дисертації. Розроблені методики прогнозування, очищення даних та проактивного керування демонструють значні переваги порівняно з існуючими підходами та створюють міцну основу для побудови інтелектуальних, ефективних та людино-орієнтованих систем керування розумним будинком.

ВИСНОВКИ

У дисертаційній роботі вирішено актуальну науково-практичну задачу розробки та дослідження комплексу методик оптимізації керування розумним будинком на основі Інтернету речей. Проблема полягала в недостатній ефективності, гнучкості та адаптивності існуючих систем, що обмежувало їхній потенціал у підвищенні комфорту, енергоефективності та зниженні когнітивного навантаження на користувачів. Це було зумовлено прогалинами на трьох ключових рівнях: низькою якістю сенсорних даних через гетерогенні шуми, обмеженим балансом між точністю та обчислювальною вартістю методів прогнозування, а також статичністю та недостатньою людино-орієнтованістю архітектур керування.

Метою роботи було створення цілісного рішення, що усуває ці недоліки. Для досягнення мети було поставлено та вирішено низку завдань, результатом чого стали три основні наукові результати, підтверджені комплексним експериментальним дослідженням.

На основі проведеного дослідження зроблено наступні висновки.

Розроблено та експериментально доведено ефективність вдосконаленого гібридного методу прогнозування енергоспоживання TCN-LightGBM зі стратегією "прогнозування залишків". Підтверджено, що гібридні моделі значно перевершують базові архітектури (1D-CNN, LSTM, TCN) за точністю. Виявлено, що конфігурація TCN-LightGBM є лідером у прогнозуванні критичних пікових навантажень (Peak MAPE 16,71%) при вищій на 21% швидкості навчання порівняно з LSTM-LightGBM. Ключовим досягненням стала розробка методики оптимізації цієї моделі шляхом відбору 100 найважливіших ознак для коректора LightGBM. Це дозволило скоротити час навчання в 5,4 рази (з 305 до 56 секунд) при збереженні високої точності прогнозування піків (погіршення лише на 0,06%). Наукова новизна полягає у доведенні можливості досягнення такого значного прискорення без суттєвої втрати точності для даної архітектури, що вирішує проблему компромісу між точністю та обчислювальною вартістю.

Практичне значення полягає у створенні високоточної та обчислювально ефективної моделі, придатної для розгортання на пристроях Інтернету речей з обмеженими ресурсами.

Розроблено та апробовано метод адаптивного очищення різномірних сенсорних даних, реалізований у базовій (ACRA) та вдосконаленій (H-AD-CLEAN) архітектурах. Метод вирішує проблему неефективності існуючих підходів при роботі з гетерогенними шумами (викиди, дрейф, константні значення) в режимі реального часу. Якщо архітектура ACRA базується на класифікаторі Random Forest та евристичних правилах, то H-AD-CLEAN розвиває цей підхід, інтегруючи глибоке навчання (1D-CNN з вейвлет-перетворенням) та механізм «вентильної логіки» (Gating Logic). Це дозволило реалізувати стратегію вибіркової корекції: система з високою точністю (F1-score 0,85) ідентифікує «чисті» сегменти даних і захищає їх від спотворень, застосовуючи фільтрацію виключно до аномалій. Експериментально доведено, що запропонований підхід стабільно перевершує традиційні методи: зокрема, H-AD-CLEAN забезпечує зниження середньої абсолютної помилки (MAE) на 21,1% порівняно зі стандартним фільтром Калмана, усуваючи його головний недолік — згладжування валідних піків сигналу. Наукова новизна полягає у створенні гібридної стратегії, що адаптивно обирає режим обробки (Pass-Through або фільтрація) на основі розпізнавання локальної структури та «текстури» сигналу. Практичне значення полягає у наданні надійного інструменту для формування якісних вхідних даних, що є критичним для коректної роботи всіх інтелектуальних підсистем розумного будинку.

Розроблено та валідовано проактивну архітектуру керування розумним будинком на основі контекстуальних намірів користувача та багатоцільової оптимізації. Архітектура реалізує парадигму керування на основі високорівневих намірів (баланс комфорту та економії), використовуючи прогностичне моделювання та багатоцільову оптимізацію функції корисності. Ключовою інновацією є механізм адаптивного навчання контекстуальним намірам, який аналізує ручні втручання користувача як неявний зворотний зв'язок для

коригування ваг пріоритетів у різних контекстах (час доби, тариф). Імітаційне моделювання показало, що розроблений Проактивний агент досяг найвищого рівня комфорту (41,5% часу в цільовій зоні) при кращому балансі з фінансовими витратами порівняно з системами на основі правил. Найважливішим результатом є кардинальне зменшення потреби в ручному втручанні користувача – понад 97% порівняно з базовою моделлю. Наукова новизна полягає у здатності системи автономно навчатися саме контекстуальним, а не глобальним намірам, використовуючи неявний зворотний зв'язок для багатоцільової оптимізації. Практичне значення полягає у створенні гнучких, людино-орієнтованих систем, що самостійно адаптуються до поведінки мешканців, мінімізуючи їхнє когнітивне навантаження та підвищуючи загальну ефективність керування.

Таким чином, усі поставлені в дисертаційній роботі задачі було успішно вирішено. Розроблений комплекс взаємопов'язаних методик – від очищення даних та прогнозування до проактивного керування – створює цілісне рішення для оптимізації функціонування систем розумного будинку на основі Інтернету речей. Отримані результати мають як наукову новизну, вдосконалюючи існуючі підходи, так і значне практичне значення, відкриваючи шлях до створення більш інтелектуальних, ефективних та комфортних житлових просторів.

Перспективними напрямками подальших досліджень є валідація розроблених методик у реальних умовах експлуатації, розширення методів для обробки багатовимірних часових рядів та роботи в багатокористувацьких середовищах, розробка механізмів інкрементального навчання для швидкої адаптації до змін, а також створення стандартизованих протоколів оцінки для об'єктивного порівняння різних архітектур у майбутньому.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Алгоритми керування інтелектуальними системами: монографія. Костенко В. П. Київ: Наукова думка, 2021. 312 с.
2. Голобородько В. В., Шматков С. І. Моделі і методи побудови багаторівневих систем керування в Інтернеті речей. Проблеми програмування. 2022. № 3. С. 37–45. DOI: 10.15407/pp2022.03.037.
3. Дослідження та розробка проактивних систем керування енергоспоживанням у Smart Grid на базі навчання з підкріпленням. Кравченко А. О. Вісник Національного технічного університету «ХПІ». Серія: Інформатика та моделювання. 2023. № 1. С. 82–90. URL: http://nbuv.gov.ua/UJRN/VKhPU_Nitu_2023_1_12 (дата звернення: 04.12.2025).
4. ДСТУ ISO/IEC 27001:2015 (ISO/IEC 27001:2013, IDT). Інформаційні технології. Методи та засоби безпеки. Системи управління інформаційною безпекою. Вимоги. [Чинний від 2015-09-01]. Київ: ДП «УкрНДНЦ», 2015. 48 с.
5. Інтелектуальні системи автоматизації: підручник. Мельник А. І., Іванов О. С. Харків: ХНУРЕ, 2022. 450 с.
6. Коваленко І. В., Приймак О. В. Багатоцільова оптимізація в інтелектуальних системах керування: методи та застосування. Кібернетика та системний аналіз. 2021. Т. 57, № 6. С. 3–15.
7. Концепція розвитку технологій Smart City в Україні. Київ: Міністерство цифрової трансформації України, 2024. 45 с. URL: <https://thedigital.gov.ua/ua/smartcity> (дата звернення: 04.12.2025).
8. Методи та алгоритми побудови контекстно-залежних систем для розумних будинків. Ляшенко В. С. Наукові праці Донецького національного технічного університету. Серія: Інформатика, кібернетика та обчислювальна техніка. 2020. Вип. 2. С. 101–108.
9. Олійник А. В. Архітектури розподілених систем Інтернету речей та проблеми їх масштабування. Інформаційні технології та комп'ютерна інженерія. 2023. Т. 2, № 1. С. 25–36. DOI: 10.30837/itce.2023.1.025.

10. Оптимізація енергоефективності Smart Home на основі прогнозування поведінки користувачів. Петренко О. І. Збірник наукових праць Інституту проблем моделювання в енергетиці. 2019. Вип. 95. С. 76–85.
11. Особливості застосування машинного навчання для проактивного керування в системах автоматизації. Смирнов П. В., Ткаченко Р. Г. Автоматика. Автоматизація. Електротехнічні комплекси та системи. 2024. № 1. С. 45–52. URL: <https://doaj.org/article/xyz-id> (дата звернення: 04.12.2025).
12. Про електронні комунікації: Закон України від 16.12.2020 № 1089-IX. Відомості Верховної Ради України. 2021. № 3. Ст. 14. URL: <https://zakon.rada.gov.ua/laws/show/1089-20> (дата звернення: 04.12.2025).
13. Про затвердження Стратегії розвитку сфери інноваційної діяльності на період до 2030 року: розпорядження Кабінету Міністрів України від 10.09.2018 № 668-р. Урядовий кур'єр. 2018. 18 вересня.
14. Про основні засади забезпечення кібербезпеки України: Закон України від 05.10.2017 № 2163-VIII. Відомості Верховної Ради України. 2017. № 45. Ст. 403. URL: <https://zakon.rada.gov.ua/laws/show/2163-19> (дата звернення: 04.12.2025).
15. Про схвалення Концепції розвитку цифрової економіки та суспільства України на 2018-2020 роки та затвердження плану заходів щодо її реалізації: розпорядження Кабінету Міністрів України від 17.01.2018 № 67-р. Урядовий кур'єр. 2018. 2 лютого.
16. Прогнозування часових рядів в задачах керування: монографія. Шевчук І. А. Львів: Видавництво Львівської політехніки, 2020. 270 с.
17. Теорія адаптивних систем: навчальний посібник. За редакцією проф. Зінченка М. І. Київ: КНУ ім. Т. Шевченка, 2018. 350 с.
18. Федоренко Д. В. Розробка багатоагентної системи керування для оптимізації комфорту та ресурсів у розумному будинку. Вісник Київського національного університету імені Тараса Шевченка. Серія: Фізико-математичні науки. 2022. № 4. С. 112–119.

19. A. Karkouch, H. Mousannif, H. Al Moatassime, T. Noel, Data quality in internet of things: A state-of-the-art survey, *J. Netw. Comput. Appl.* 73 (2016) 57–81. doi:10.1016/j.jnca.2016.08.002.
20. A. Zhang, S. Song, J. Wang, P. S. Yu, Time Series Data Cleaning: From Anomaly Detection to Anomaly Repairing, *Proc. VLDB Endow.* 10.10 (2017) 1046–1057. doi:10.14778/3115404.3115410.
21. Adaptive Management of Heterogeneous IoT Subsystems in Smart Homes Using Fuzzy Logic and Hybrid Optimization Framework / U. Kuppusamy та ін. *SSRN Electronic Journal*. 2025. URL: <https://doi.org/10.2139/ssrn.5089027> (дата звернення: 15.09.2025).
22. Aguirre-Fraire B. A Comprehensive Dataset Integrating Household Energy Consumption and Weather Conditions in a North-eastern Mexican Urban City [Electronic resource] / Baldemar Aguirre-Fraire, Jessica Beltrán, Valeria Soto-Mendoza // *Data in Brief*. – 2024. – P. 110452. – Mode of access: <https://doi.org/10.1016/j.dib.2024.110452> (date of access: 09.05.2025). – Title from screen.
23. Ahmed, A. A., Najim, A. H., Mahdi, A. J., Alheeti, K. M. A., Satar, N. S. M., & Hashim, A. H. A. (2024). Empowering smart homes with fog computing for iot connectivity. <https://doi.org/10.1109/dasa63652.2024.10836355>
24. Ajadalu S. O. Optimizing Energy Efficiency in Smart Home Automation through Reinforcement Learning and Iot. *Asian Journal of Research in Computer Science*. 2024. Т. 17, № 11. С. 9–24. URL: <https://doi.org/10.9734/ajrcos/2024/v17i11516> (дата звернення: 15.09.2025).
25. Akhmetzhanov, B. K., Akhmetzhanov, B., Yedilkhan, D., Medeshova, A., Rabie, K., & Zhakiyev, N. (n.d.). Multi-layer integration of heterogeneous wireless sensor networks for smart home optimization. *Procedia Computer Science*, . <https://doi.org/10.1016/j.procs.2023.12.166>
26. Al-Alami, H., & Jamleh, H. O. (2023). Use of Convolutional Neural Networks and Long Short-Term Memory for Accurate Residential Energy Prediction.

27. Almufti, S. M., Hani (2024). Smart home energy saving with big data and machine learning. *Jurnal ilmiah ilmu terapan Universitas Jambi*, 8 (1), 11-20. <https://doi.org/10.22437/jiituj.v8i1.32598>
28. Amune, A., Rajurkar, S., Pawar, A. S., Rane, S., Pichare, P., Shendre, R., & Musale, V. (2025). Iot-based smart energy monitoring and forecasting using long short-term memory and cloud integration. <https://doi.org/10.1109/icscds65426.2025.11166789>
29. An, N., Ding, Y., and Zhao, H. Statistical feature analysis and preprocessing assisted artificial neural network for cleaning multi-type concurrent anomalies in time series data. *Proc. ISCAS*, 2024. doi: 10.1109/ISAS61044.2024.10552599
30. Anomaly Detection in Fractal Time Series with LSTM Autoencoders [Electronic resource] / Lyudmyla Kirichenko [et al.] // *Mathematics*. – 2024. – Vol. 12, no. 19. – P. 3079. – Mode of access: <https://doi.org/10.3390/math12193079> (date of access: 09.05.2025). – Title from screen.
31. B. Zhang, C. Song, X. Jiang ta Y. Li, “Electricity price forecast based on the STL-TCN-NBEATS model”, *Heliyon*, № 1(9), 2023. <https://doi.org/10.1016/j.heliyon.2023.e13029>
32. .Babu, U. R., Kumar, N., Padmaja, K., et al. (2024). Predicting Electricity Consumption in Commercial and Residential Buildings Using A-CNN-LSTM Model.
33. Barbato, A., Borsani, L., Capone, A., & Melzi, S. (2009). Home energy saving through a user profiling system based on wireless sensors. <https://doi.org/10.1145/1810279.1810291>
34. Barhate, M. Y., Mishra, A., Mehar, V. N., Mishra, P., Mhashakhatri, K. M., More, K. K., & Meshram, S. S. (2025). Intelligent home energy management systems: Budget-based predictions and iot controls. <https://doi.org/10.1109/incet64471.2025.11141014>
35. Beyertt, A., Verwiebe, P., Seim, S., Milojkovic, F., Müller-Kirchenbauer, J. Felduntersuchung zu Behavioral Energy Efficiency Potentialen privater Haushalte: Working Paper Energie und Ressourcen im Rahmen des BAFA-Förderprogramms

Einsparzähler sowie des BMWi-geförderten Forschungsprojekts DemandRegio. Berlin, 2020. DOI: 10.5281/zenodo.3855575.

36. Bikku, T., Sivakumar, S., Chinthamani, S. A. M., & Patro, P. (2024). Iot-based renewable energy management systems in apartment. <https://doi.org/10.1002/9781394231522.ch3>

37. Bindushree, G. T. (2022). Iot-based energy management in smart homes: A literature review. *World Journal Of Advanced Research and Reviews*, 16 (3), 1392-1400. <https://doi.org/10.30574/wjarr.2022.16.3.1296>

38. Breiman, L. Random Forests. *Machine Learning* 45, 5–32 (2001), doi:10.1023/A:1010933404324.

39. Brenon, Alexis & Portet, François & Vacher, Michel. (2018). ARCADES: A deep model for adaptive decision making in voice controlled smart-home. *Pervasive and Mobile Computing*. 49. 10.1016/j.pmcj.2018.06.011.

40. Bridging the gap between natural user expression with complex automation programming in smart homes. Shi, Yingtian & Liu, Xiaoyi & Yu, Chun & Yang, Tianao & Gao, Cheng & Liang, Chen & Shi, Yuanchun. (2024). 10.48550/arXiv.2408.12687.

41. Chahuara P., Portet F., Vacher M. Context-aware decision making under uncertainty for voice-based control of smart home. *Expert Systems with Applications*. 2017. T. 75. C. 63–79. URL: <https://doi.org/10.1016/j.eswa.2017.01.014> (дата звернення: 15.09.2025).

42. Chenaru, O., & Popescu, D. (2020). Iot gateway for personalized user comfort management in smart home applications. <https://doi.org/10.1109/MED48518.2020.9182926>

43. Choubey, P. K., Pateria, S., Saxena, A., Sb, V. P. C., Jha, K. K., & Pm, S. B. (2015). Power efficient, bandwidth optimized and fault tolerant sensor management for iot in smart home. <https://doi.org/10.1109/IADCC.2015.7154732>

44. Claeys, R., Cleenwerck, R., Knockaert, J., et al. (2024). Capturing multiscale temporal dynamics in synthetic residential load profiles through Generative Adversarial Networks (GANs). *Applied Energy*.

45. Context-Aware IoT System Development Approach Based on Meta-Modeling and Reinforcement Learning / A. Hallou та ін. International Journal of Online and Biomedical Engineering (iJOE). 2024. Т. 20, № 06. С. 25–42. URL: <https://doi.org/10.3991/ijoe.v20i06.46545> (дата звернення: 15.09.2025).
46. Corrales D. How to Address the Data Quality Issues in Regression Models: A Guided Process for Data Cleaning [Electronic resource] / David Corrales, Juan Corrales, Agapito Ledezma // Symmetry. – 2018. – Vol. 10, no. 4. – P. 99. – Mode of access: <https://doi.org/10.3390/sym10040099> (date of access: 09.05.2025). – Title from screen.
47. Cotti, L., Guizzardi, D., Barricelli, B. R., & Fogli, D. (2024). Enabling end-user development in smart homes: A machine learning-powered digital twin for energy efficient management. Future Internet, 16 (6), 208-208. <https://doi.org/10.3390/fi16060208>
48. Dasappa, N. S., G, K. K., & Somu, N. (2024). Multi-sensor data fusion framework for energy optimization in smart homes. Renewable & Sustainable Energy Reviewsnull, . <https://doi.org/10.1016/j.rser.2023.114235>
49. Data quality in internet of things: A state-of-the-art survey [Electronic resource] / Aimad Karkouch [et al.] // Journal of Network and Computer Applications. – 2016. – Vol. 73. – P. 57–81. – Mode of access: <https://doi.org/10.1016/j.jnca.2016.08.002> (date of access: 09.05.2025). – Title from screen.
50. Devane, S. (2025). Smart energy management using iot. Indian Scientific Journal Of Research In Engineering And Management, 09 (04), 1-9. <https://doi.org/10.55041/ijsrem45578>
51. Dinesh, P. M., Basavaraja, G., Valarmathi, C., G, L. P., & Bhargavi, L. N. J. (2024). “powering the future: Smart home energy management unleashed”. <https://doi.org/10.59544/nzop7063/icrcct24p41>
52. Dobrovolskis A., Kazanavičius E., Kižauskienė L. Building XAI-Based Agents for IoT Systems. Applied Sciences. 2023. Т. 13, № 6. С. 4040. URL: <https://doi.org/10.3390/app13064040> (дата звернення: 15.09.2025).

53. Ebrahimi, M., Fonseca, J., Shafie-khah, M., Osório, G. J., & Catalão, J. P. S. (2024). Machine-learning-based home energy management framework via residents' feedback. <https://doi.org/10.1109/sest61601.2024.10694198>
54. Deep Temporal Convolution Network for Time Series Classification / B. H. D. Koh та ін. *Sensors*. 2021. Т. 21, № 2. С. 603. <https://doi.org/10.3390/s21020603>.
55. Geetha, B. T., Lova, P., Naidu, I. E. S., Sharma, M. K., Dhaygude, P. S., & Govekar, N. S. (2024). Hems-iot: Smart home energy saving with big data and machine learnings. <https://doi.org/10.1109/ic3i61595.2024.10828998>
56. Gowda, D., Samal, R., Reddy, P. B., Marthanda, A., Billady, R. K., & Rajlakshmi, P. (2024). A novel framework for ai-driven, cloud-integrated energy-efficient iot solutions in smart homes. 2020 4th International Conference on Electronics, Communication and Aerospace Technology (ICECA) null, 327-332. <https://doi.org/10.1109/iceca63461.2024.10800957>
57. Grassi, M., Nucci, M., & Piazza, F. (2011). Towards a semantically-enabled holistic vision for energy optimisation in smart home environments. <https://doi.org/10.1109/ICNSC.2011.5874943>
58. Gu, Y., Chen, Q., Liu, K., et al. (2019). GAN-based Model for Residential Load Generation Considering Typical Consumption Patterns.
59. H. N. Akouemo, R. J. Povinelli, Data Improving in Time Series Using ARX and ANN Models, *IEEE Trans. Power Syst.* 32.5 (2017) 3352–3359. doi:10.1109/tpwrs.2017.2656939.
60. Haq, M., Sohail, M., Qureshi, I. M., Shoaib, H. M., Saha, I., Duhis, M., John, A., & Shoaib, M. (2025). Shaping the next generation of ai-integrated smart homes: Innovations in intelligence, security, sustainability, and user experience. *Scholars journal of engineering and technology*, 13 (08), 644-656. <https://doi.org/10.36347/sjet.2025.v13i08.004>
61. He, Q., & Su, Y. (2022). Residential Load Forecasting Based on CNN-LSTM and Non-Uniform Quantization.

62. Hu, Y., Li, Y., Song, L., et al. (2024). MultiLoad-GAN: A GAN-Based Synthetic Load Group Generation Method Considering Spatial-Temporal Correlations. *IEEE Transactions on Smart Grid* .
63. Iao, H.-W., & Lao, K.-W. (2023). Integrated Load Consumption and PV Output Forecasting of Net-zero Energy Buildings Considering KNN-GAN Data Augmentation.
64. Ibebuchi C. C. Fuzzy time series clustering using autoencoders neural network [Electronic resource] / Chibuike Chiedozie Ibebuchi // *AIMS Geosciences*. – 2024. – Vol. 10, no. 3. – P. 524–539. – Mode of access: <https://doi.org/10.3934/geosci.2024027> (date of access: 09.05.2025). – Title from screen.
65. Ibude, F. E., Otebolaku, A., Ameh, J. E., et al. (2024). Multi-Timescale Energy Consumption Management in Smart Buildings Using Hybrid Deep Artificial Neural Networks. *Journal of Low Power Electronics and Applications* .
66. Integration of time series forecasting in a dynamic decision support system for multiple reservoir management to conserve water sources [Electronic resource] / Hamed Zamani Sabzi [et al.] // *Energy Sources, Part A: Recovery, Utilization, and Environmental Effects*. – 2018. – Vol. 40, no. 11. – P. 1398–1416. – Mode of access: <https://doi.org/10.1080/15567036.2018.1476934> (date of access: 09.05.2025). – Title from screen.
67. Internet of Things (IoT): A vision, architectural elements, and future directions [Electronic resource] / Jayavardhana Gubbi [et al.] // *Future Generation Computer Systems*. – 2013. – Vol. 29, no. 7. – P. 1645–1660. – Mode of access: <https://doi.org/10.1016/j.future.2013.01.010> (date of access: 09.05.2025). – Title from screen.
68. J. Gama, I. Žliobaitė, A. Bifet, M. Pechenizkiy, A. Bouchachia, A survey on concept drift adaptation, *ACM Comput. Surv.* 46.4 (2014) 1–37. doi:10.1145/2523813.

69. J. Gubbi, R. Buyya, S. Marusic, M. Palaniswami, Internet of Things (IoT): A vision, architectural elements, and future directions, *Future Gener. Comput. Syst.* 29.7 (2013) 1645–1660. doi:10.1016/j.future.2013.01.010.
70. Jagannathan, S., Deepa, V., Durai, B. S. K., Rajan, M. S., Celin, J. J. A., & Dhamodaran, S. (2024). Smart home automation with iot sensor systems for energy conservation using zigbee and wifi. <https://doi.org/10.1109/icesc60852.2024.10690054>
71. Ji, X., Huang, H., Chen, D., et al. (2022). A Hybrid Residential Short-Term Load Forecasting Method Using Attention Mechanism and Deep Learning. *Buildings* .
72. K. Yasumoto, H. Yamaguchi, H. Shigeno, Survey of Real-time Processing Technologies of IoT Data Streams, *J. Inf. Process.* 24.2 (2016) 195–202. doi:10.2197/ipsjip.24.195.
73. Kaya, M., Utku, A., & Canbay, Y. (2024). A Hybrid CNN-LSTM Model for Predicting Energy Consumption and Production Across Multiple Energy Sources. *Journal of Soft Computing and Artificial Intelligence* , 5(2), 63-73.
74. Khan, A. N. (2023). An ocf-iotivity enabled smart-home optimal indoor environment control system for energy and comfort optimization. *Internet of things*, 22 null, 100712-100712. <https://doi.org/10.1016/j.iot.2023.100712>
75. Khan, Q. W., Ahmad, R., Rizwan, A., Khan, A., Lee, K., & Kim, D. (2024). Optimizing energy efficiency and comfort in smart homes through predictive optimization: A case study with indoor environmental parameter consideration. *Energy Reports*null, . <https://doi.org/10.1016/j.egyr.2024.05.038>
76. Kormpakis, G., Lekidis, A., Papias, I., Dimitropoulos, N., Serepas, F., & Marinakis, V. (2025). Holistic framework for household energy management services. <https://doi.org/10.23919/splitech65624.2025.11091616>
77. Kotsopoulos, S. D., Casalegno, F., Recasens, A., & Graybill, W. (2014). Developing a restful communication protocol and an energy optimization algorithm for a connected sustainable home. <https://doi.org/10.4108/ICST.URB-IOT.2014.257303>
78. Krishna, S. R., Kumar, R., Gaurav, A., Singh, N., Sharma, M., & Almas, S. (2024). Intelligent adaptive systems and methods thereof for household energy control and management. <https://doi.org/10.1109/icepes60647.2024.10653541>

79. Kumar, P. (2025). Spot-energy: A user-centric platform for energy simulation, prediction and optimization. *International Journal For Science Technology And Engineering*, 13 (8), 2223-2227. <https://doi.org/10.22214/ijraset.2025.73928>
80. Lambor, P. S., Nimbolkar, A., Patil, A., Rathod, K., Wadile, P., & Lakde, V. (2024). Intelligent energy management with predictive analysis. <https://doi.org/10.1109/i-smac61858.2024.10714650>
81. Li, S., Malleeshwaran, T., Prasanna, T. S., & Daniel, A. (2024). Ai-driven iot framework for optimal energy management in consumer devices. <https://doi.org/10.1109/icsadl61749.2024.00129>
82. Liu, J., & Hsiao, Y. (2025). Smart plug-based home energy management with iot and anomaly detection. *Journal of the Chinese Institute of Engineers*, 1-16. <https://doi.org/10.1080/02533839.2025.2557233>
83. Liu, Y., Liang, Z., Li, X., et al. (2023). Generative Adversarial Network and CNN-LSTM Based Short-Term Power Load Forecasting.
84. M. K. Shende, A. E. Feijóo-Lorenzo, N. D. Bokde, cleanTS: Automated (AutoML) Tool to Clean Univariate Time Series at Microscales, *Neurocomputing* (2022). doi:10.1016/j.neucom.2022.05.057.
85. Madhav, P. V., & Anand, N. V. (2025). Optimized cost-effective energy management system for iot-enabled smart homes using price-based ego approach. *Intelligent Decision Technologies*. <https://doi.org/10.1177/18724981251344141>
86. Mekuria, Dagmawi & Sernani, Paolo & Falcionelli, Nicola & Dragoni, Aldo Franco. (2019). Reasoning in Multi-agent Based Smart Homes: A Systematic Literature Review: Italian Forum 2018. 10.1007/978-3-030-05921-7_13.
87. Moghimi, S. M., Gulliver, T. A., Ilamparithi, T., & Teimoorinia, H. (2025). Occupant-centric load optimization in smart green townhouses using machine learning. *Energies*, 18 (13), 3320-3320. <https://doi.org/10.3390/en18133320>
88. Natarajan, Y., S. K. R., Wadhwa, G., et al. (2024). Enhancing Building Energy Efficiency with IoT-Driven Hybrid Deep Learning Models for Accurate Energy Consumption Prediction. *Sustainability*.

89. Navaz, K., Paul, M. M. R., Bee, J. Y., Kavitha, S., Bhuvanesh, A., & Mariappan, R. (2024). Smart home automation with smart metering using zigbee technology and deep belief networks – optimization using mb-abcoa algorithm. <https://doi.org/10.1109/inc460750.2024.10649324>
90. Naz, L. F., Qamar, R., Asif, R., Hina, S., Imran, M., & Ahmed, S. (2025). Intelligent energy management in iot-enabled smart homes: Anomaly detection and consumption prediction for energy-efficient usage. *Mehran University Research Journal of Engineering and Technology*, 44 (1), 113-113. <https://doi.org/10.22581/muet1982.3291>
91. Nuha, M. F. A. U., & Muklason, A. (2024). Improving Energy Consumption Forecasting through Conditional Generative Adversarial Networks.
92. Nurgaliyev, Kenzhegali & Di Mauro, Dario & Khan, Nawaz & Augusto Wrede, Juan. (2017). Improved Multi-user Interaction in a Smart Environment Through a Preference-Based Conflict Resolution Virtual Assistant. 100-107. 10.1109/IE.2017.21.
93. Oseiwe, A. S. (2024). Optimizing energy efficiency in smart home automation through reinforcement learning and iot. *Asian Journal of Research in Computer Science*, 17 (11), 9-24. <https://doi.org/10.9734/ajrcos/2024/v17i11516>
94. Outlier Detection using Clustering Techniques [Electronic resource] / Srividya . [et al.] // *International Journal of Engineering & Technology*. – 2018. – Vol. 7, no. 3.12. – P. 813. – Mode of access: <https://doi.org/10.14419/ijet.v7i3.12.16508> (date of access: 09.05.2025). – Title from screen.
95. Papaioannou, C., Tzitzios, I., Papaioannou, A., Dimara, A., Anagnostopoulos, C., & Krinidis, S. (2025). Energiq: A prescriptive large language model-driven intelligent platform for interpreting appliance energy consumption patterns. *Sensors*, 25 (16), 4911-4911. <https://doi.org/10.3390/s25164911>
96. R, S. K., Bangera, H. D., HN, H., Harshitha, M., & Harshitha, M. (2025). A review: Efficient energy saving techniques in smart homes through iot. <https://doi.org/10.1109/icdici66477.2025.11134994>

97. R. Ahmad, E. H. Alkhamash, Online Adaptive Kalman Filtering for Real-Time Anomaly Detection in Wireless Sensor Networks, *Sensors* 24.15 (2024) 5046. doi:10.3390/s24155046.
98. R. E. Kalman, A New Approach to Linear Filtering and Prediction Problems, *J. Basic Eng.* 82.1 (1960) 35–45. doi:10.1115/1.3662552.
99. Raj, N., Sinha, A. R., Bharrdwaj, S., & Yadav, A. L. (2024). Ai-powered energy consumption optimization for smart homes using iot. <https://doi.org/10.1109/iccica60014.2024.10585239>
100. Raju K. H. P. Supervised SVM Classification of Rainfall Datasets [Electronic resource] / K. Hari Prasada Raju, N. Sandhya, Raghav Mehra // *Indian Journal of Science and Technology*. – 2017. – Vol. 10, no. 15. – P. 1–6. – Mode of access: <https://doi.org/10.17485/ijst/2017/v10i15/106115> (date of access: 09.05.2025). – Title from screen.
101. Raju, M., Kumar, J. K., Supriya, K., et al. (2024). Leveraging Weather Parameters in Generative Adversarial Networks for Energy Consumption Prediction.
102. Regi, R. J., Prakash T, K., Joseph, J., et al. (2023). Predictive Modeling of Residential Energy Usage with Deep Learning and IoT Devices.
103. Rong K., Bailis P. ASPA: Adaptive Smoothing for Streaming Time Series, *Proc. VLDB Endow.* 10.11 (2017) 1358–1369. doi:10.14778/3137628.3137645.
104. Sasikala, P., Sivakumar, S., Kalipindi, M., & Kumbhkar, M. (2024). Development of a framework to integrate smart home and energy operation systems to manage energy efficiency through AI. <https://doi.org/10.1002/9781394231522.ch1>
105. Sengaliappan, D. (2025). Optimizing energy consumption in smart homes using ML techniques. *Indian Scientific Journal Of Research In Engineering And Management*, 09 (03), 1-9. <https://doi.org/10.55041/ijsrem41971>
106. Sethi S. Data Governance in Smart Home Systems: The S.H.I.E.L.D. Framework [Electronic resource] / Simran Sethi // *International Journal For Multidisciplinary Research*. – 2024. – Vol. 6, no. 1. – Mode of access: <https://doi.org/10.36948/ijfmr.2024.v06i01.39041> (date of access: 09.05.2025). – Title from screen.

107. Shahriar, M. S., & Rahman, M. S. (2015). Urban sensing and smart home energy optimisations: A machine learning approach. <https://doi.org/10.1145/2820975.2820979>
108. Siswipraptini, P. C., Aziza, R. N., Ruli, R., Siregar, A., & Ramadhan, A. (2024). Smart home energy management systems: A systematic review of architecture, communication, and algorithmic trends. *Journal of system and management sciencesnull*, . <https://doi.org/10.33168/jsms.2024.1108>
109. Song, S., Zhang, A., Wang, J., & Yu, P.S., SCREEN: Stream Data Cleaning under Speed Constraints, Proceedings of the 2015 ACM SIGMOD International Conference on Management of Data.
110. State-Space Model and Kalman Filter Gain Identification by a Kalman Filter of a Kalman Filter [Electronic resource] / Minh Q. Phan [et al.] // *Journal of Dynamic Systems, Measurement, and Control*. – 2017. – Vol. 140, no. 3. – Mode of access: <https://doi.org/10.1115/1.4037778> (date of access: 09.05.2025). – Title from screen.
111. Stefanopoulou, A., Dimara, A., Michailidis, I., Karatzinis, G. D., Papaioannou, A., Krinidis, S., Anagnostopoulos, C., Kosmatopoulos, E., Ioannidis, D., & Tzovaras, D. (2023). Ensuring reliability in smart building iot operations through real-time holistic data treatment. https://doi.org/10.1007/978-3-031-34171-7_16
112. Stogia, M., Naserentin, V., Dimara, A., Eleftheriou, O., Tzitzios, I., Papaioannou, C., Pantusheva, M., Papaioannou, A., Spaias, G., Anagnostopoulos, C., Logg, A., & Krinidis, S. (2024). A scalable and user-friendly framework integrating iot and digital twins for home energy management systems. *Applied Sciences*, 14 (24), 11834-11834. <https://doi.org/10.3390/app142411834>
113. Sutton R. S., Barto A. G., Bach F. Reinforcement Learning: An Introduction. MIT Press, 2018. 552 c.
114. Triban, S., & Lawgali, A. (2023). Residential Short-Term Load Forecasting Based on CNN-LSTM with Consumer Behavior Pattern.
115. UCI Machine Learning Repository. (n.d.). Appliances Energy Prediction. Retrieved from <https://archive.ics.uci.edu/dataset/374/appliances+energy+prediction>

116. Ur, Blase & McManus, Elyse & Ho, Melwyn & Littman, Michael. (2014). Practical trigger-action programming in the smart home. Conference on Human Factors in Computing Systems - Proceedings. 10.1145/2556288.2557420.
117. V. Chandola, A. Banerjee, V. Kumar, Anomaly detection, ACM Comput. Surv. 41.3 (2009) 1–58. doi:10.1145/1541880.1541882.
118. V. Hodge, J. Austin, A Survey of Outlier Detection Methodologies, Artif. Intell. Rev. 22.2 (2004) 85–126. doi:10.1023/b:aire.0000045502.10941.a9..
119. Vastardis, N., Kampouridis, M., & Yang, K. (2016). A user behaviour-driven smart-home gateway for energy management. Journal of Ambient Intelligence and Smart Environments, 8 (6), 583-602. <https://doi.org/10.3233/AIS-160403>
120. Venkatesh, A. N. (2024). Iot-based smart home automation systems: Enhancing energy efficiency and security. International Journal For Science Technology And Engineering, 12 (12), 2243-2253. <https://doi.org/10.22214/ijraset.2024.66177>
121. Wang X. Time Series Data Cleaning: A Survey [Electronic resource] / Xi Wang, Chen Wang // IEEE Access. – 2020. – Vol. 8. – P. 1866–1881. – Mode of access: <https://doi.org/10.1109/access.2019.2962152> (date of access: 09.05.2025). – Title from screen.
122. Wang, Yan & Zheng, Liwei & He, Jingjing & Cui, Zhanqi. (2023). Adaptive IoT Decision Making in Uncertain Environments. 265-269. 10.1109/SmartIoT58732.2023.00048.
123. X. Ding, H. Wang, J. Su, Z. Li, J. Li, H. Gao, Cleanits: a data cleaning system for industrial time series, Proc. VLDB Endow. 12.12 (2019) 1786–1789. doi:10.14778/3352063.3352066.
124. Xu, R. (2024). Optimizing smart home systems: Utilizing blockchain and edge computing. <https://doi.org/10.1109/icesc60852.2024.10689819>
125. Yin, M., and Yue, K. Time Series Data Cleaning Method Based on Optimized ELM Prediction. JIPS, vol. 13, no. 2, 2017, pp. 432-445. doi:10.3745/JIPS.04.0268.

126. Yogita, D. Toshniwal, A Framework for Outlier Detection in Evolving Data Streams by Weighting Attributes in Clustering, *Procedia Technol.* 6 (2012) 214–222. doi:10.1016/j.protcy.2012.10.026.
127. Youssef, H., Osman, R., & El-Bary, A. A. (2024). Efficient connectivity in smart homes: Enhancing living comfort through iot infrastructure. <https://doi.org/10.3390/s24092761>
128. Zafoune, Y. (2023). An iot ml-based system for energy efficiency in smart homes. <https://doi.org/10.1109/AIPoT58121.2023.10174484>
129. Zerroug, A. (2024). Review of technics used in smart house. <https://doi.org/10.1109/iceeac61226.2024.10576205>.
130. Ніщепенко Д.О. Розробка системи керування розумним будинком на основі протоколу MQTT. IV Всеукраїнська студентська наукова конференція: «Експериментальні та теоретичні дослідження в контексті сучасної науки». матеріали (м.Чернігів, 29 вересня 2023 р.), 2023. С. 154.
131. Ніщепенко Д.О. Протокол MQTT в додатку NESTJS: реалізація обміну повідомленнями. Всеукраїнська науково-практична конференція: «Цифрова гуманістика: Інформаційні технології та інформаційне моделювання на сучасному етапі розвитку суспільства». матеріали (м. Кропивницький, 4-5 червня 2024 р.), 2024. С. 119—123.
132. Ніщепенко Д.О. Безпека та захист обміну даними за допомогою протоколу MQTT для керування системами інтернету речей. Всеукраїнська науково-практична конференція: «Актуальні проблеми безпеки інформаційно-телекомунікаційних систем». збірник тез (м. Київ, 03 листопада 2024 р.), 2024. С. 98-99.
133. Ніщепенко Д.О. Роль мікроядра в енергоефективності операційних систем для інтернету речей. Науково-практична конференція «Проблеми комп'ютерної інженерії». Збірник тез (К., 2024), 2024. С. 200-202.
134. Ніщепенко Д.О., Майборода М.В. Порівняння ARIMA та LSTM у контексті прогнозування споживання енергії в умовах розумного будинку. VI Міжнародна науково-практична конференція «Сучасні досягнення компанії

Hewlett Packard Enterprise в галузі ІТ та нові можливості їх вивчення і застосування». Збірник тез (К., 2024 р.), 2024. С. 22-24.

135. Ніщенко Д.О. Резалізація збереження та управління даними в IoT-додатках на основі Java Spring Framework. II Міжнародна науково-практична конференція «Сучасні аспекти діджиталізації та інформатизації в програмній та комп'ютерній інженерії». Збірник тез (К., 2024 р.), 2024. С. 33–36.

136. Ніщенко Д.О. Адаптивний метод ACRA для очищення різномірних сенсорних даних в системах розумного будинку. II Всеукраїнська науково-практична конференція «Цифрова гуманістика: Інформаційні технології та інформаційне моделювання на сучасному етапі розвитку суспільства». матеріали (м. Кропивницький, травень 2025 р.), 2025. С 238-243.

137. Волощук О.В., Ніщенко Д.О. Проблематика класифікації типів шумів для адаптивного очищення різномірних сенсорних даних. XIV Міжнародна науково-практична конференція: «Математика. Інформаційні технології. Освіта». матеріали, 2025. С. 76-78.

138. D. Nishchemenko, V. Zhebka, S. Shlianchak, S. Popereshnyak. Real-time adaptive cleaning of IoT sensor data using machine learning noise classification and rule-based refinement. MoMLeT 2025 Modern Machine Learning Technologies Workshop. CEUR Workshop Proceedings (2025). Scopus

139. Ніщенко Д.О., Аронов А.О. Дослідження методик оптимізації параметрів системи керування розумним будинком з використанням ІОТ. Телекомунікаційні та інформаційні технології, №1, 2025. С. 141-150.

140. Ніщенко Д.О., Волощук О.Б. Адаптивне очищення різномірних сенсорних даних у системах розумного будинку на основі класифікації шумів. Електронне фахове наукове видання «Кібербезпека: освіта, наука, техніка», 4(28), 2025. С. 740–750.

141. Ніщенко Д. Оптимізація гібридних моделей на основі TCN, LSTM, LIGHTGBM для прогнозування енергоспоживання в розумних будинках. Електронне фахове наукове видання «Кібербезпека: освіта, наука, техніка», 2(30), 2025. С. 224–237.

142. Ніщенко Д.О., Олейников І.А. Проактивна архітектура керування розумним будинком на основі контекстуальних намірів користувача та багатоцільової оптимізації. Телекомунікаційні та інформаційні технології, № 3, 2025. С. 141-149

143. Ніщенко Д.О., Аронов А.О., Гавор А.С., Герцюк, М.М., Гордієнко, К.О. Аналіз мов парадигми ООП для реалізації масштабованих систем із Docker Наука і техніка сьогодні, 10(51), 2025 С. 1406–1418.

Додаток А. Лістинг програмного коду реалізації класу гібридної моделі прогнозування TCN-LightGBM (метод прогнозування залишків)

```
import numpy as np
import lightgbm as lgb
import tensorflow as tf
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Dense
from tensorflow.keras.callbacks import EarlyStopping
from tcn import TCN

class TCNLightGBMHybrid:
    """
    Реалізація гібридного методу прогнозування енергоспоживання.
    Поєднує глибоке навчання (TCN) для виявлення глобальних трендів
    та градієнтний бустинг (LightGBM) для корекції локальних залишків.
    """

    def __init__(self, n_steps_in, n_steps_out, n_features,
                  tcn_filters=64, tcn_kernel=3, tcn_dilations=(1, 2, 4, 8),
                  lgbm_estimators=50):
        """
        Ініціалізація параметрів гібридної моделі.

        Args:
            n_steps_in (int): Довжина вхідної послідовності.
            n_steps_out (int): Горизонт прогнозування.
            n_features (int): Кількість вхідних ознак.
            tcn_filters (int): Кількість фільтрів у шарах TCN.
            tcn_kernel (int): Розмір ядра згортки.
            tcn_dilations (tuple): Коефіцієнти розширення (dilation) для TCN.
        """
```



```

        lgbm_estimators (int): Кількість дерев для моделей LightGBM.
        """

        self.n_steps_in = n_steps_in
        self.n_steps_out = n_steps_out
        self.n_features = n_features

        # Параметри архітектури
        self.tcn_filters = tcn_filters
        self.tcn_kernel = tcn_kernel
        self.tcn_dilations = tcn_dilations
        self.lgbm_estimators = lgbm_estimators

        # Компоненти моделі
        self.model_tcn = None
        self.corrector_models = [] # Ансамбль коректорів (по одному на часовий
        крок)

    def _build_tcn_model(self):
        """Побудова архітектури базової нейронної мережі (TCN)."""
        model = Sequential([
            TCN(
                input_shape=(self.n_steps_in, self.n_features),
                nb_filters=self.tcn_filters,
                kernel_size=self.tcn_kernel,
                dilations=self.tcn_dilations,
                return_sequences=False
            ),
            Dense(100, activation='relu'),
            Dense(self.n_steps_out, activation='linear')
        ])

```

```
model.compile(optimizer='adam', loss='mean_squared_error')  
return model
```

```
def fit(self, X_seq_train, y_train, X_features_train,  
        epochs=50, batch_size=64, validation_split=0.2):  
    """
```

Навчання гібридної моделі у два етапи.

Args:

X_seq_train: 3D-тензор для TCN (зразки, кроки, ознаки).

y_train: 2D-масив цільових значень.

X_features_train: 2D-масив табличних ознак для LightGBM.

```
    """
```

```
# Етап 1: Навчання базової моделі TCN
```

```
self.model_tcn = self._build_tcn_model()
```

```
early_stopping = EarlyStopping(  
    monitor='val_loss', patience=5, restore_best_weights=True
```

```
)
```

```
self.model_tcn.fit(  
    X_seq_train, y_train,
```

```
    epochs=epochs,
```

```
    batch_size=batch_size,
```

```
    validation_split=validation_split,
```

```
    callbacks=[early_stopping],
```

```
    verbose=0
```

```
)
```

```
# Етап 2: Розрахунок залишків (помилки) базової моделі
```

```
y_pred_tcn = self.model_tcn.predict(X_seq_train, verbose=0)
```

```

residuals = y_train - y_pred_tcn

# Етап 3: Навчання ансамблю LightGBM для корекції залишків
self.corrector_models = []

# Навчаємо окремих регресор для кожного кроку горизонту
прогнозування
for i in range(self.n_steps_out):
    y_res_i = residuals[:, i]
    lgbm = lgb.LGBMRegressor(
        n_estimators=self.lgbm_estimators,
        random_state=42,
        n_jobs=-1,
        verbose=-1
    )
    lgbm.fit(X_features_train, y_res_i)
    self.corrector_models.append(lgbm)

def predict(self, X_seq_test, X_features_test):
    """
    Генерація прогнозу: сума виходу TCN та корекції LightGBM.
    """
    if self.model_tcn is None:
        raise Exception("Модель не навчена.")

    # Прогноз базового тренду
    base_pred = self.model_tcn.predict(X_seq_test, verbose=0)

    # Прогноз помилки (корекції)
    predicted_residuals_list = []
    for i in range(self.n_steps_out):

```

```
model = self.corrector_models[i]
pred_res = model.predict(X_features_test)
predicted_residuals_list.append(pred_res)

predicted_residuals = np.stack(predicted_residuals_list, axis=1)

# Фінальна агрегація
return base_pred + predicted_residuals
```

**Додаток Б. Лістинг програмного коду реалізації алгоритму адаптивного
очищення даних H-AD-CLEAN (CNN+DWT класифікатор та механізм
Gating)**

```
import numpy as np
import pandas as pd
from collections import deque
import tensorflow as tf
from tensorflow.keras.models import Model
from tensorflow.keras.layers import Input, Conv1D, MaxPooling1D, Flatten,
Dense, concatenate, Dropout

class HADCleanAlgorithm:
    """
    Реалізація методу H-AD-CLEAN для відновлення якості даних сенсорів
    IoT.
    Використовує гібридну неймережу для класифікації типу аномалії
    та адаптивний механізм вибору методу відновлення.
    """

    def __init__(self, window_size, num_classes=5):
        self.window_size = window_size
        self.num_classes = num_classes
        self.model = None

    def build_classifier_model(self, cnn_input_shape, dwt_input_shape):
        """Побудова гібридної архітектури CNN + MLP (на основі DWT)."""
        # Гілка 1: CNN для аналізу форми сигналу
        input_cnn = Input(shape=cnn_input_shape, name='cnn_input')
        x_cnn = Conv1D(filters=32, kernel_size=7, activation='relu',
padding='same')(input_cnn)
```

```

x_cnn = MaxPooling1D(pool_size=2)(x_cnn)
x_cnn = Dropout(0.2)(x_cnn)
x_cnn = Conv1D(filters=64, kernel_size=7, activation='relu',
padding='same')(x_cnn)
x_cnn = MaxPooling1D(pool_size=2)(x_cnn)
x_cnn = Flatten()(x_cnn)
cnn_features = Dense(64, activation='relu')(x_cnn)

# Гілка 2: MLP для аналізу спектральних ознак (DWT)
input_dwt = Input(shape=dwt_input_shape, name='dwt_input')
x_dwt = Dense(32, activation='relu')(input_dwt)
x_dwt = Dropout(0.2)(x_dwt)
dwt_features = Dense(32, activation='relu')(x_dwt)

# Об'єднання ознак (Fusion)
combined = concatenate([cnn_features, dwt_features])
x_combined = Dense(128, activation='relu')(combined)
x_combined = Dropout(0.4)(x_combined)
output = Dense(self.num_classes, activation='softmax')(x_combined)

self.model = Model(inputs=[input_cnn, input_dwt], outputs=output)
self.model.compile(optimizer='adam', loss='categorical_crossentropy',
metrics=['accuracy'])
return self.model

def _apply_gating_mechanism(self, predicted_class, window_buffer,
last_valid_val):
"""
Вибір оператора очищення на основі класифікованого типу аномалії.

```

Типи класів:

0: Clean (Норма) -> Pass-through

1: Outlier (Викид) -> Медіанна фільтрація

2: Constant (Залипання) -> Лінійна екстраполяція

3: Drift (Дрейф) -> Згладжування

4: Noise (Шум) -> Експоненційне згладжування

"""

```
window = np.array(window_buffer)
```

```
current_val = window[-1]
```

```
if predicted_class == 0: # Clean
```

```
    return current_val
```

```
elif predicted_class == 1: # Outlier
```

```
    # Медіана вікна без поточної точки (стійка до викидів)
```

```
    return np.median(window[:-1])
```

```
elif predicted_class == 2: # Constant
```

```
    # Лінійна екстраполяція за попередніми точками
```

```
    if len(window) > 3:
```

```
        return 2 * window[-2] - window[-3]
```

```
    return last_valid_val
```

```
elif predicted_class == 3: # Drift
```

```
    # Локальне усереднення для вирівнювання тренду
```

```
    return np.mean(window[-4:-1])
```

```
elif predicted_class == 4: # General Noise
```

```
    # Експоненційне згладжування (Low-pass filter)
```

```
    alpha = 0.3
```

```
return alpha * current_val + (1 - alpha) * last_valid_val
```

```
return current_val
```

```
def process_signal(self, signal, scaler_cnn, scaler_dwt, dwt_extractor_func):
```

```
    """
```

```
    Поточкова обробка сигналу (імітація реального часу).
```

```
    """
```

```
    cleaned_signal = []
```

```
    buffer = deque(maxlen=self.window_size)
```

```
    last_cleaned = signal[0]
```

```
    for i, point in enumerate(signal):
```

```
        # Обробка пропущених значень на вході
```

```
        if pd.isna(point):
```

```
            cleaned_point = last_cleaned
```

```
        else:
```

```
            buffer.append(point)
```

```
        # Якщо буфер не повний, пропускаємо без обробки
```

```
        if len(buffer) < self.window_size:
```

```
            cleaned_point = point
```

```
        else:
```

```
            # Підготовка вхідних векторів
```

```
            raw_window = np.array(buffer)
```

```
            cnn_in = scaler_cnn.transform(raw_window.reshape(-1, 1)).reshape(1,  
self.window_size, 1)
```

```
            dwt_feats = dwt_extractor_func(raw_window)
```

```
            dwt_in = scaler_dwt.transform([dwt_feats])
```



```

# Класифікація стану
probs = self.model.predict([cnn_in, dwt_in], verbose=0)
pred_class = np.argmax(probs[0])

# Адаптивне відновлення
cleaned_point = self._apply_gating_mechanism(
    pred_class, buffer, last_cleaned
)

cleaned_signal.append(cleaned_point)
last_cleaned = cleaned_point

return np.array(cleaned_signal)

```

Додаток В. Лістинг програмного коду реалізації класу проактивного агента та компонентів адаптації

```
import numpy as np

class KnowledgeBase:
    """
    База знань агента: зберігає контекстуальні наміри (ваги пріоритетів)
    та параметри комфорту користувача.
    """
    def __init__(self):
        # Наміри зберігаються як словник контекстів (час доби + тариф)
        self.intents = { }
        self.comfort_zone = (20, 22)
        # Початкові ваги: пріоритет комфорту вищий за економію
        self.default_weights = {'w_comfort': 0.7, 'w_energy': 0.3}

    def get_weights_for_context(self, context_key):
        """Повертає ваги пріоритетів для конкретного контексту."""
        if context_key not in self.intents:
            self.intents[context_key] = self.default_weights.copy()
        return self.intents[context_key]

class PredictiveModel:
    """
    Модель прогнозування стану середовища.
    Використовується агентом для оцінки наслідків дій (Model-Based
    approach).
    """
    def __init__(self, env_params):
        self.step_duration_hours = env_params['step_duration_hours']
```

```

self.heat_loss_coeff = env_params['heat_loss_coefficient']
self.heater_efficiency = env_params['heater_efficiency']
self.heater_power_levels = env_params['heater_power_levels']

```

```

def predict(self, state, action):

```

```

    """

```

Прогнозує температуру та енергоспоживання на наступний крок.

Args:

state: Поточний стан (температура, зовн. темп, час, тариф).

action: Обрана дія (рівень потужності).

Returns:

predicted_temp: Прогнозована температура.

energy_consumed_kwh: Прогнозовані витрати енергії.

```

    """

```

```

    current_temp, outdoor_temp, _, _ = state

```

```

    power_kw = self.heater_power_levels[action]

```

```

    # Розрахунок фізики процесу

```

```

    energy_consumed_kwh = power_kw * self.step_duration_hours

```

```

    heat_gain = power_kw * self.heater_efficiency * 2

```

```

    heat_loss = (current_temp - outdoor_temp) * self.heat_loss_coeff

```

```

    predicted_temp = current_temp + (heat_gain - heat_loss) *
self.step_duration_hours

```

```

    return predicted_temp, energy_consumed_kwh

```

```

class AdaptiveLearningModule:

```

```

    """

```

Модуль навчання: коригує ваги функції корисності на основі

втручань користувача (Feedback Loop).

```
"""

def learn_from_override(self, knowledge_base, context_key, system_action,
user_action):
    """
    Оновлює ваги (comfort vs energy) залежно від того, як користувач
    змінив дію системи.
    """

    weights = knowledge_base.get_weights_for_context(context_key)

    # Якщо система гріла, а користувач вимкнув -> пріоритет економії
    if system_action > 0 and user_action == 0:
        weights['w_comfort'] = max(0.01, weights['w_comfort'] - 0.1)
        weights['w_energy'] = min(0.99, weights['w_energy'] + 0.1)

    # Якщо система не гріла, а користувач увімкнув -> пріоритет комфорту
    elif system_action == 0 and user_action > 0:
        weights['w_comfort'] = min(0.99, weights['w_comfort'] + 0.1)
        weights['w_energy'] = max(0.01, weights['w_energy'] - 0.1)

    # Нормалізація ваг
    total = weights['w_comfort'] + weights['w_energy']
    weights['w_comfort'] /= total
    weights['w_energy'] /= total

    # Збереження оновленого профілю
    knowledge_base.intents[context_key] = weights
```

```
class ProactiveAgent:
```

```
    """
```

Головний клас проактивного агента.

Приймає рішення на основі максимізації функції корисності (Utility Function),

враховуючи прогноз стану та адаптивні ваги.

"""

```
def __init__(self, env_params):
```

```
    self.knowledge_base = KnowledgeBase()
```

```
    self.predictive_model = PredictiveModel(env_params)
```

```
    self.learning_module = AdaptiveLearningModule()
```

```
    self.action_space_size = len(env_params['heater_power_levels'])
```

```
def _get_context_key(self, hour, tariff):
```

```
    """Формує унікальний ключ контексту (Час доби + Тип тарифу)."""
```

```
    if 6 <= hour < 12: time_of_day = "morning"
```

```
    elif 12 <= hour < 18: time_of_day = "afternoon"
```

```
    elif 18 <= hour < 23: time_of_day = "evening"
```

```
    else: time_of_day = "night"
```

```
    tariff_str = "peak" if tariff == 1 else "off-peak"
```

```
    return f"{time_of_day}_{tariff_str}"
```

```
def _calculate_utility(self, predicted_temp, action, weights, tariff):
```

```
    """
```

Розраховує корисність дії (Utility) як зважену суму комфорту та економії.

```
    """
```

```
    comfort_min, comfort_max = self.knowledge_base.comfort_zone
```

```
    power_kw = self.predictive_model.heater_power_levels[action]
```

```
    max_power = max(self.predictive_model.heater_power_levels.values())
```

```

# 1. Оцінка комфорту
if comfort_min <= predicted_temp <= comfort_max:
    # Чим ближче до центру зони комфорту, тим краще
    comfort_center = np.mean(self.knowledge_base.comfort_zone)
    deviation = abs(predicted_temp - comfort_center)
    comfort_score = 1.0 - (deviation / (comfort_max - comfort_center))
else:
    # Штраф за вихід за межі комфорту
    base_discomfort_penalty = -100.0
    if predicted_temp < comfort_min:
        comfort_score = base_discomfort_penalty - (comfort_min -
predicted_temp)
    else:
        comfort_score = base_discomfort_penalty - (predicted_temp -
comfort_max)

# 2. Оцінка енергоефективності
energy_penalty = (power_kw / max_power) if max_power > 0 else 0

# Штраф потроюється під час пікового тарифу
if tariff == 1 and power_kw > 0:
    energy_penalty *= 3
energy_score = 1 - energy_penalty

# Фінальна корисність
utility = (weights['w_comfort'] * comfort_score + weights['w_energy'] *
energy_score)
return utility

def choose_action(self, observation):

```

```

"""
Основний метод прийняття рішень.
Перебирає всі можливі дії, прогнозує їх наслідки та обирає найкращу.
"""

best_action = -1
max_utility = -np.inf

_, _, hour, tariff = observation
context_key = self._get_context_key(hour, tariff)

# Отримуємо актуальні ваги для поточного контексту
current_weights =

self.knowledge_base.get_weights_for_context(context_key)

# Жадібний пошук (Greedy Search) найкращої дії
for action in range(self.action_space_size):
    # Прогноз майбутнього стану
    predicted_temp, _ = self.predictive_model.predict(observation, action)

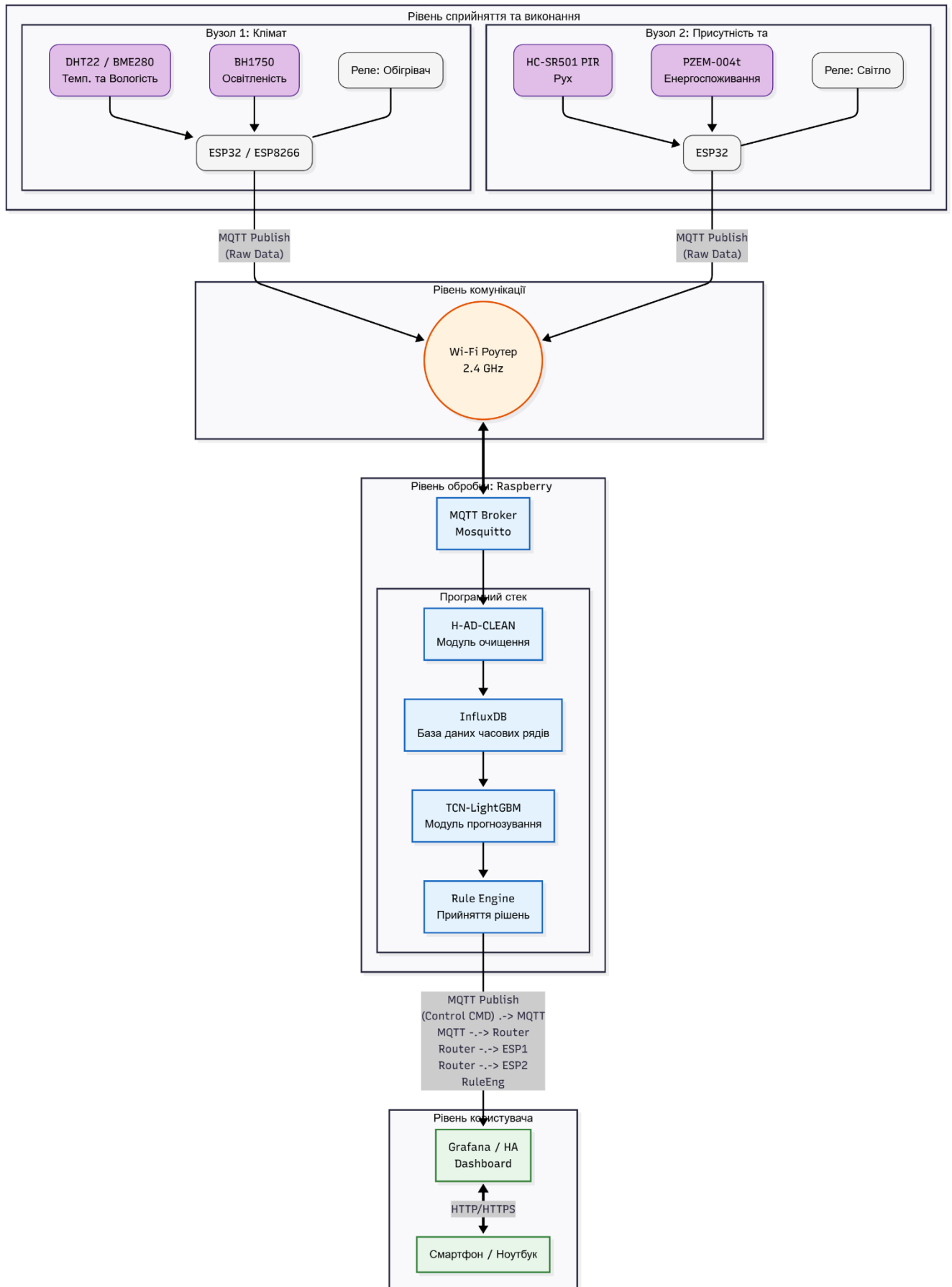
    # Розрахунок корисності
    utility = self._calculate_utility(predicted_temp, action, current_weights,
tariff)

    if utility > max_utility:
        max_utility = utility
        best_action = action

return best_action

```

Додаток Г. Структурна схема апаратно-програмного комплексу



На схемі зображено взаємодію основних компонентів системи:

1. Сенсорна підсистема, побудована на базі мікроконтролерів ESP32, що забезпечують збір даних про параметри мікроклімату (температура, вологість) та поведінку користувача (рух, енергоспоживання) в режимі реального часу;
2. Комунікаційне середовище, обмін даними здійснюється через бездротову мережу Wi-Fi з використанням протоколу MQTT, що гарантує низьку затримку передачі пакетів;
3. Обчислювальне ядро, реалізовано на одноплатному комп'ютері Raspberry Pi 4, розгорнуто програмні реалізації розроблених методів:
 - H-AD-CLEAN для адаптивного очищення вхідного потоку даних;
 - TCN-LightGBM для короткострокового прогнозування навантаження;
4. Інтерфейс користувача, забезпечує візуалізацію аналітики та можливість налаштування параметрів системи проактивного керування.

**Додаток Д. Повні результати порівняння точності моделей ARIMA,
LSTM, GRU, TCN, TCN- XGBoost, TCN-LightGBM**

№	Model	Етап тесту	MAPE (%)	RMSE	MAPE (%)	Час навчання (с)	Час прогнозу (s)
1	1D-CNN	1	15.758153	2.088639	17.480118	17.778822	2.118967
2	1D-CNN	2	13.642932	1.657971	20.879948	15.235325	1.763769
3	1D-CNN	3	11.757180	1.585949	23.179379	12.959534	1.797669
4	1D-CNN	4	15.157686	2.464960	33.502151	14.619279	1.775319
5	1D-CNN	5	15.359263	2.334771	21.440630	10.927810	1.767560
6	LSTM	1	15.233769	2.050329	18.818564	46.931767	3.117644
7	LSTM	2	14.028418	1.710447	20.546121	39.451314	3.129537
8	LSTM	3	11.759223	1.513058	20.129781	84.737137	3.121581
9	LSTM	4	14.606359	2.418063	33.300177	66.343684	3.107054
10	LSTM	5	17.948227	2.375435	21.986706	104.145816	3.142816
11	TCN	1	13.843282	2.009430	20.605460	36.206027	3.195782
12	TCN	2	15.505135	1.867661	20.203284	15.407063	2.939619
13	TCN	3	13.400612	1.750581	22.829142	22.449214	2.909280
14	TCN	4	15.308453	2.527652	32.493922	22.759816	2.952762
15	TCN	5	17.268540	2.445482	18.037076	29.281976	2.947872
16	TCN + LightGBM (Peak Correction)	1	13.499244	2.132800	27.755374	41.255784	0.030315
17	TCN + LightGBM (Peak Correction)	2	12.594801	1.635638	26.360370	54.722800	0.033618

№	Model	Етап тесту	MAPE (%)	RMSE	MAPE (%)	Час навчання (с)	Час прогнозу (s)
18	TCN + LightGBM (PeakCorr.)	3	16.025996	1.839563	21.678703	110.515184	0.043055
19	TCN + LightGBM (PeakCorr.)	4	13.493546	1.597020	19.586503	177.146739	0.044952
20	TCN + LightGBM (PeakCorr.)	5	11.516278	1.730600	13.360358	286.795089	0.052169
21	TCN + XGBoost (PeakCorr.)	1	17.062165	2.496608	28.286398	39.312481	0.268985
22	TCN + XGBoost (PeakCorr.)	2	14.631212	1.802729	25.491498	80.804424	0.272273
23	TCN + XGBoost (PeakCorr.)	3	15.571596	1.773917	23.033274	121.541806	0.297030
24	TCN + XGBoost (PeakCorr.)	4	13.254272	1.631946	22.403767	206.242277	1.038177
25	TCN + XGBoost (Peak Correction)	5	12.898132	1.765246	12.487990	246.224766	0.275404

№	Model	Етап тесту	MAPE (%)	RMSE	MAPE (%)	Час навчання (с)	Час прогнозу (s)
26	TCN + LightGBM (Residual Fitting)	1	12.692838	1.908578	17.411865	282.494646	0.357695
27	TCN + LightGBM (Residual Fitting)	2	12.600779	1.556825	15.128560	296.760626	0.320989
28	TCN + LightGBM (Residual Fitting)	3	13.380521	1.650827	19.496013	311.933314	0.290841
29	TCN + LightGBM (Residual Fitting)	4	15.017094	1.678793	18.218913	321.906297	0.284830
30	TCN + LightGBM (Residual Fitting)	5	13.468780	1.765406	13.291565	315.508550	0.222775
31	TCN + LightGBM (Residuals, Optimized)	1	12.683999	1.977820	18.894386	62.908974	0.177354

№	Model	Етап тесту	MAPE (%)	RMSE	MAPE (%)	Час навчання (с)	Час прогнозу (s)
32	TCN + LightGBM (Residuals, Optimized)	2	14.500980	1.743711	15.127402	46.232094	0.167892
33	TCN + LightGBM (Residuals, Optimized)	3	13.648431	1.633612	20.604825	53.999524	0.150091
34	TCN + LightGBM (Residuals, Optimized)	4	14.951336	1.627749	16.582076	61.638414	0.131437
35	TCN + LightGBM (Residuals, Optimized)	5	13.338478	1.769197	12.646350	57.546829	0.115815

Додаток Е. Акти впровадження

ЗАТВЕРДЖУЮ

Директор Інституту
телекомунікацій і глобального
інформаційного простору
Національної академії наук
України



[Signature] О.М. Трофимчук
5 " 11 2025 р.

АКТ

впровадження наукових результатів дисертації
аспіранта Державного університету інформаційно-комунікаційних
технологій

НІЩЕМЕНКА Дмитра Олександровича

Комісія у складі голови комісії – завідувача відділу дослідження навколишнього середовища Інституту телекомунікацій і глобального інформаційного простору Національної академії наук України д.т.н., професора Триснюка В.М., та членів комісії: головного наукового співробітника Інституту телекомунікацій і глобального інформаційного простору Національної академії наук України д.т.н. с.н.с. Яковлева Є.О., старшого наукового співробітника, к.т.н., старшого дослідника Охарева В.О. розглянула запропоновані матеріали в межах дисертаційного дослідження аспіранта Ніщенко Дмитра Олександровича на тему “Методи оптимізації керування розумним будинком на основі Інтернету речей”.

Комісією підтверджено, що реалізація поставленого наукового завдання сприяє суттєвому підвищенню енергоефективності та рівня автоматизації систем керування житловими об'єктами, що досягається завдяки розробці комплексу методів, що забезпечують високоточне прогнозування навантажень та адаптивне керування кліматичним обладнанням.

У рамках дослідження розроблено та експериментально реалізовано програмний модуль прогнозування TCN-LightGBM, який забезпечує

скорочення часу навчання моделі в 5,4 рази. При цьому досягнуто високої точності прогнозування пікових навантажень (похибка менше 17%), що дозволяє ефективно використовувати розробку на пристроях з обмеженими ресурсами. Для підвищення якості вхідної інформації застосовано технологію адаптивного очищення даних H-AD-CLEAN, яка на 21,1% ефективніша за традиційні фільтри Калмана у задачах усунення гетерогенних шумів без спотворення корисного сигналу.

На основі очищених та спрогнозованих даних функціонують розроблені алгоритми проактивного керування, що адаптуються до контекстуальних намірів користувачів. Використання даного підходу дозволило кардинально оптимізувати взаємодію з системою, зменшивши потребу в ручному втручанні користувача на 97% у порівнянні з існуючими базовими моделями, що спираються на жорсткі правила.

Практичне значення отриманих результатів полягає в тому, що впровадження розроблених у дисертаційному дослідженні елементів інформаційних технологій дозволяє створювати масштабовані системи розумного будинку, які автономно адаптуються до поведінки мешканців та динамічних тарифів на електроенергію.

Отримані результати використано для вдосконалення системи моніторингу енергоспоживання підприємства. Крім того, теоретичні та практичні результати дисертаційного дослідження впроваджено в навчальний процес Навчально-наукового інституту інформаційних технологій Державного університету інформаційно-комунікаційних технологій.

Голова комісії:

Члени комісії

	В.М. Триснюк
	С.О. Яковлев
	В.О. Охарева



ЗАТВЕРДЖУЮ
Перший проректор Державного
університету інформаційно-
комунікаційних технологій

Корченко О. Г.
11 2025 року

А К Т

впровадження в навчальний процес Державного університету інформаційно-комунікаційних технологій наукових положень і результатів дисертаційної роботи Ніщепенка Дмитра Олександровича на тему «Методи оптимізації керування розумним будинком на основі Інтернету речей»

Науково-педагогічна комісія у складі голови – директора навчально-наукового інституту Інформаційних технологій доктора технічних наук, професора Нестеренко К.С. та членів комісії: завідувача кафедри Штучного інтелекту доктора технічних наук, доцента Зінченко О.В., завідувача кафедри Інформаційних систем та технологій доктора технічних наук, професора Сторчак К.П. склала даний акт про те, що наукові положення і результати дисертаційної роботи на здобуття наукового ступеня доктора філософії Ніщепенка Д.О. на тему «Методи оптимізації керування розумним будинком на основі Інтернету речей», а саме

1. Гібридний метод короткострокового прогнозування енергоспоживання TCN-LightGBM, який використовує стратегію прогнозування залишків та дозволяє досягти балансу між точністю та обчислювальною ефективністю, введено в навчальний процес кафедри Штучного інтелекту Державного університету інформаційно-комунікаційних

технологій при викладанні дисципліни «Штучний інтелект» для студентів спеціальності 121 «Інженерія програмного забезпечення» денної форми навчання.

2. Метод адаптивного очищення різномірних сенсорних даних (H-AD-CLEAN), що поєднує класифікацію аномалій та адаптивну фільтрацію для підвищення якості даних у реальному часі введено в навчальний процес кафедри Інформаційних систем та технологій Державного університету інформаційно-комунікаційних технологій при викладанні дисципліни «Технології Інтернет речей» для студентів спеціальності 126 «Інформаційні системи та технології» денної форми навчання.

3. Проактивну архітектуру керування на основі контекстуальних намірів, яка базується на багатоцільовій оптимізації та навчанні на неявному зворотному зв'язку, введено в навчальний процес кафедри Інформаційних систем та технологій Державного університету інформаційно-комунікаційних технологій при викладанні дисципліни «Моделювання IoT» для студентів спеціальності 126 «Інформаційні системи та технології» денної форми навчання.

Голова комісії:

Директор навчально-наукового інституту Інформаційних технологій

доктор технічних наук, професор



Нестеренко К.С.

Члени комісії:

Завідувач кафедри Штучного інтелекту

доктор технічних наук, доцент



Зінченко О.В.

Завідувач кафедри Інформаційних систем та технологій

доктор технічних наук, професор



Сторчак К.П.