НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ
імені ІГОРЯ СІКОРСЬКОГО»

Навчально-науковий інститут телекомунікаційних систем
Кафедра інформаційних технологій в телекомунікаціях

«На правах рукопису»
УДК ___004.942___

«До захисту допущено»
Завідувач кафедри
_____ Марія СКУЛИШ
“___”_____2025 р.

Магістерська дисертація

зі спеціальності 172 Електронні комунікації та радіотехніка
на тему: **Засоби штучного інтелекту для виявлення та запобігання збоїв у телекомунікаційних мережах**

Виконав: студент VI курсу, групи ЦІ-41мп
Дикий Микола Ігорович _____ _____
(підпис)

Науковий керівник: доцент кафедри ІТТ ННІТС,
кандидат технічних наук, доцент
Новогрудська Ріна Леонідівна _____ _____
(підпис)

Рецензент: професор кафедри ЕКІР,
доктор технічних наук, професор
Мошинська Аліна Валентинівна _____ _____
(підпис)

Засвідчую, що у цій магістерській дисертації немає запозичень з праць інших авторів без відповідних посилань.
Студент _____

Київ – 2025

Національний технічний університет України
«Київський політехнічний інститут
імені Ігоря Сікорського»

Навчально науковий інститут телекомунікаційних систем
Кафедра інформаційних технологій в телекомунікаціях
Рівень вищої освіти – другий магістерський за освітньо-професійною програмою Інформаційно-комунікаційні технології

Спеціальність 172 Електронні комунікації та радіотехніка

ЗАТВЕРДЖУЮ
Завідувач кафедри
_____ Марія СКУЛИШ
“___”_____2025 р.

ЗАВДАННЯ
на магістерську дисертацію студенту
Дикому Миколі Ігоровичу

1. Тема дисертації **Засоби штучного інтелекту для виявлення та запобігання збоїв у телекомунікаційних мережах**

Науковий керівник дисертації Новогрудська Ріна Леонідівна, доцент кафедри ІТТ НН ІТС, кандидат технічних наук, доцент затверджені наказом по університету від «03» листопада 2025 р. № 4772-с

2. Строк подання студентом дисертації «12» грудня 2025 р.

3. Об’єкт дослідження

Процеси моніторингу, тестування та діагностики стану телекомунікаційних мереж і систем радіодоступу.

4. Предмет дослідження (Вихідні дані – для магістерської дисертації за освітньо-професійною програмою)

Методи та моделі ШІ і машинного навчання, що використовуються для виявлення та прогнозування аномалій у телекомунікаційних даних.

5. Перелік завдань, які потрібно розробити

1. Аналіз існуючих методів виявлення аномалій у часових рядах (традиційні та глибинні підходи).
2. Підготовка даних для навчання моделей (масштабування, обробка пропусків, створення навчального та тестового наборів).
3. Розробка та реалізація моделей прогнозування

4. Розробка та реалізація моделей реконструкції
5. Проведення експериментів та оцінка ефективності моделей (precision, recall, F1-score).
6. Аналіз результатів та порівняння продуктивності моделей (вплив порогів, розміру даних, здатність до виявлення аномалій).

6. Перелік графічного (ілюстративного) матеріалу

Електронна презентація в редакторі PowerPoint

7. Дата видачі завдання 23 листопада 2024 року

Календарний план

№ з/п	Назва етапів виконання магістерської дисертації	Строк виконання етапів магістерської дисертації	Примітка
1	Дослідження та вивчення отриманого завдання	23.11.24	Виконано
2	Дослідження проблеми виявлення збоїв у RAN	1.12.24-1.01.25	Виконано
3	Аналіз архітектури RAN та формулювання мети й завдань дослідження	01.01.25 – 03.02.25	Виконано
4	Огляд традиційних методів виявлення аномалій у часових рядах	04.02.25 – 10.03.25	Виконано
5	Вивчення глибинних методів виявлення аномалій (прогнозування, реконструкція)	12.03.25 – 19.03.25	Виконано
6	Розробка моделей прогнозування та реконструкції	3.10.25 – 2.10.25	Виконано
7	Проведення експериментів та оцінка ефективності моделей	4.10.25 – 1.12.25	Виконано
8	Аналіз результатів: вплив порогів, розміру даних, якість виявлення аномалій	9.12.25 – 11.12.25	Виконано

Студент _____ Микола ДИКИЙ
(підпис) (ім’я, прізвище)

Науковий керівник дисертації _____ Ріна НОВОГРУДСЬКА
(підпис) (ім’я, прізвище)

РЕФЕРАТ

Тема магістерської дисертації «Засоби штучного інтелекту для виявлення та запобігання збоїв у телекомунікаційних мережах». Робота містить 72 сторінки тексту, 17 рисунків, 10 таблиць, 4 додатки, використано 34 літературних джерела.

Актуальність: У сучасних програмних системах обсяги даних зростають у геометричній прогресії, що робить ручний аналіз практично неможливим і підвищує ризики пропуску критичних збоїв. Значна частина таких даних є неструктурованою та не має попередньої розмітки, тому традиційні методи моніторингу та діагностики вже не забезпечують потрібного рівня ефективності. Це створює потребу у впровадженні автоматизованих підходів, здатних виявляти аномалії без участі експертів.

Застосування машинного навчання, зокрема методу некерованих алгоритмів, створює можливість своєчасно знаходити нетипові події, підвищувати надійність системи та скорочувати час реагування на потенційні відмови. З огляду на зростання складності інфраструктур і вимоги до безперервної роботи сервісів, дослідження моделей для автоматичного виявлення аномалій у системних журналах є актуальним та важливим завданням.

Мета роботи: Метою роботи є підвищення ефективності виявлення аномалій у сигналах багатовимірних часових рядів (RAN) радіомереж доступу за рахунок використання методів машинного навчання.

Об’єкт дослідження: Процес виявлення аномалій у телекомунікаційних мережах і системах радіодоступу.

Предмет дослідження: Методи й моделі машинного навчання для підвищення ефективності виявлення та прогнозування аномалій у телекомунікаційних даних.

Методи дослідження: Методи інтелектуального аналізу даних, методи штучного інтелекту, системний аналіз

ABSTRACT

Master's Thesis Topic is Artificial Intelligence Tools for Detecting and Preventing Failures in Telecommunication Networks. The thesis consists of 72 pages of text, 17 figures, 10 tables, 4 appendices, and references to 34 sources.

The relevance of the research topic:

In modern software systems, data volumes are growing exponentially, making manual analysis practically impossible and increasing the risk of missing critical failures. A significant portion of such data is unstructured and lacks prior labeling, which means that traditional monitoring and diagnostic methods no longer provide the required level of efficiency. This creates the need for automated approaches capable of detecting anomalies without expert involvement.

The application of machine learning, particularly unsupervised algorithms, enables timely identification of atypical events, improves system reliability, and reduces response time to potential failures. Considering the increasing complexity of infrastructures and the demand for uninterrupted service operation, research into models for automatic anomaly detection in system logs is both relevant and essential.

The goal of the work:

The objective of this work is to improve the efficiency of anomaly detection in signals of multidimensional time series of radio access networks (RAN) through the use of machine learning methods.

The object of research:

The process of anomaly detection in telecommunication networks and radio access systems.

Subject of Research:

Machine learning methods and models aimed at improving the efficiency of anomaly detection and forecasting in telecommunication data.

Research Methods:

Data mining methods, artificial intelligence methods, and systems analysis.

ЗМІСТ

СПИСОК СКОРОЧЕНЬ ТА УМОВНИХ ПОЗНАЧЕНЬ

ШІ	Штучний інтелект
GenAI	(General Artificial Intelligence) генеративний ШІ
LLM	(Large Language Models) Великі мовні моделі
MVTS	(Multivariate Time Series) Багатовимірні часові ряди
RAN	(Radio Access Network) радіомережа доступу
UE	(User Equipment) Користувацьке обладнання
TS	(Time Series) Часовий ряд
МН	Машинне навчання
CNN	(Convolutional Neural Network) Згорткові нейронні мережі
LSTM	(Long Short-Term Memory) Довга короткочасна пам'ять
БВЧР	Application Programming Interface
k-NN	Алгоритм k-близьких сусідів
RNN	Рекурентні нейронні мережі
GRU	Gated Recurrent Unit
DNN	Глибинні нейронні мережі
AE	Автоенкодер
BiLSTM	Bidirectional LSTM
MSL	Mars Science Laboratory
SMAP	Soil Moisture Active Passive

VAE	Варіаційний автоенкодер
BiFo	Bidirectional Forecasting
MSE	Mean Square Error
CAE	Convolutional Autoencoder
LAE	LSTM Autoencoder
SMD	Server Machine Dataset
NLP	(Natural Language Processing) Обробка природної мови

ВСТУП

У цій магістерській роботі досліджено застосування різних методів машинного навчання (МН) до телекомунікаційних даних, зокрема їх використання у тестуванні програмного забезпечення (ПЗ). Основна мета роботи — спроектувати та навчити модель для виявлення аномалій, здатну розпізнавати відхилення у багатовимірних наборах даних, що генеруються в середовищі досліджень і розробок.

Ключовим аспектом діяльності телекомунікаційної є надання телекомунікаційних послуг операторам. Розробка систем, здатних обробляти величезні обсяги даних із постійно зменшуваною затримкою, створює значну складність у проектуванні програмного забезпечення. Зі зростанням цієї складності підтримка та інтеграція нових оновлень ПЗ стають дедалі складнішими та можуть призводити до непередбачуваної поведінки системи. Такі ситуації можуть бути дорогими для телекомунікаційних компаній, оскільки їх основною метою є відповідність очікуванням клієнтів.

Виявлення таких збоїв у системі є корисним, адже їхнє запобігання може суттєво зменшити витрати [1]. Проте визначення цих збоїв, або ж аномалій, часто є складним завданням через велику складність системи. Традиційні методи, наприклад ручне встановлення порогів, певною мірою можуть допомогти, але потребують значних зусиль для налаштування та підтримки — особливо з огляду на експоненціальне зростання обсягів даних. До того ж визначення оптимального порогу для таких методів є трудомістким процесом, що вимагає глибоких галузевих знань.

Зважаючи на ці виклики, для телекомунікаційних компаній, є вкрай важливим розробляти методи виявлення збоїв, які ефективно вирішують зазначені проблеми.

РОЗДІЛ 1
ПРОБЛЕМИ ВИЯВЛЕННЯ ЗБОЇВ У РАДІОМЕРЕЖАХ ДОСТУПУ

1.1 Визначення проблеми

Одним із ключових компонентів телекомунікаційної інфраструктури є радіомережа доступу (RAN), яка відповідає за з’єднання користувацького обладнання (UE) з основною мережею. Фактично RAN виступає своєрідним мостом між мережею та нашими пристроями. З роками RAN перетворилася на надзвичайно складну систему, особливо з розвитком мереж п’ятого покоління (5G). Попри цей стрімкий прогрес, усунення несправностей і виявлення збоїв у телекомунікаційній галузі здебільшого досі здійснюється вручну — розробники аналізують великі обсяги журналів застосунків або користуються традиційними методами.

Як було згадано раніше, впровадження нових оновлень програмного забезпечення у цю постійно зростаючу систему є складним процесом, який може призводити до збоїв у мережі. Щоб вирішити цю проблему, розробники збирають системні дані у вигляді багатовимірних часових рядів (MVTS), де кожен ряд представляє певний показник продуктивності системи. Для виявлення збоїв встановлюються граничні значення за допомогою математичних операцій для кожного часового ряду (TS). Якщо значення TS перевищує або опускається нижче визначеного порогу, це розглядається як збій. Однак такий підхід має низку недоліків:

- Ручне встановлення порогів для кожного TS є трудомістким і потребує глибоких професійних знань, особливо в умовах такої складної системи.
- Методи, як-от порогове виявлення, обмежуються вже відомими аномаліями й не здатні визначати приховані або слабко виражені збої.
- Такі методи не дозволяють точно визначити момент виникнення аномалії, оскільки порушення часто проявляються як тривала поведінка системи, що ускладнює ідентифікацію початкової точки збою.

Попри успіхи методів машинного навчання (МН) у виявленні аномалій, вони також мають певні обмеження. По-перше, деякі такі підходи потребують навчання на попередньо розмічених даних. Створення таких датасетів є складним процесом, що вимагає участі експертів галузі. По-друге, моделі МН зазвичай потребують великих обсягів даних для ефективного навчання, що може бути проблемою при обмеженій кількості тренувальних даних [2] [3].

Щоб подолати зазначені виклики, у межах цього дослідження було поставлено такі наукові питання:

1. Наскільки інтеграція методів МН у тестову платформу RAN компанії може підвищити ефективність виявлення збоїв порівняно з традиційними методами, якщо оцінювати за показниками precision, recall та F1-score?
2. Як порівнюється продуктивність малих моделей і моделей із більшою кількістю параметрів на наборі даних RAN з погляду точності та обчислювальної ефективності?
3. Який обсяг даних є достатнім для ефективного навчання моделі МН для виявлення аномалій у багатовимірних часових рядах (MVTS)?
4. Чи покращують системи, підсилені МН, точність визначення моменту появи аномалій?

1.2 Радіомережа доступу (RAN)

Щоб краще зрозуміти суть проблеми, у цьому розділі подано загальне пояснення архітектури комунікаційних мереж і способів, якими розробники здійснюють діагностику цієї складної системи для виявлення збоїв.

Телекомунікаційна мережа складається з п’яти основних компонентів:

1. UE (User Equipment) — це може бути звичайний мобільний телефон, ноутбук або промислове обладнання.
2. gNodeB — базова станція мережі 5G.
3. RAN (Radio Access Network) — включає gNodeB і UE, відповідає за підключення кожного пристрою до різних частин мережі через радіоз’єднання, тобто за з’єднання UE з ядром мережі.

4. Ядро мережі (Core Network) — контролює роботу мережі та забезпечує ключові сервіси, такі як управління доступом і мобільністю.
5. Транспортна мережа (Transport Network) — з’єднує ядро мережі з RAN.

Спрощену схему цих компонентів наведено на рисунку 1.1.

Щоб кожен елемент цієї складної системи функціонував ефективно, розробники RAN проводять кілька етапів тестування для оцінки її працездатності. Завдяки безперервному циклу інтеграції (continuous integration loop) розробники можуть поступово тестувати й оновлювати систему [4]. Кожен етап тестування зосереджується на певних функціональних можливостях, і в міру просування цей процес стає дедалі тривалішим і дорожчим.

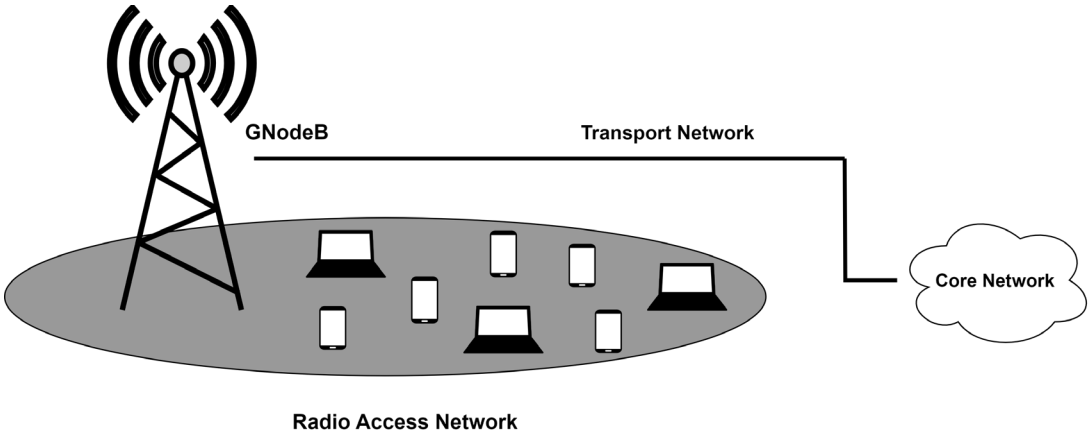

Рисунок 1.1 – Приклад архітектури мережі мобільного зв'язку.

Тому надзвичайно важливо мати систему виявлення збоїв, здатну визначати помилки на ранніх етапах тестування, а не лише на пізніх. На рисунку 1.2 показано, як працює цикл безперервної інтеграції, у межах якого програмний код постійно оновлюється через коміти багатьох розробників. Використовуючи правиловий підхід, розробники тестують систему на різних етапах і шукають збої, аналізуючи журнали тестів.

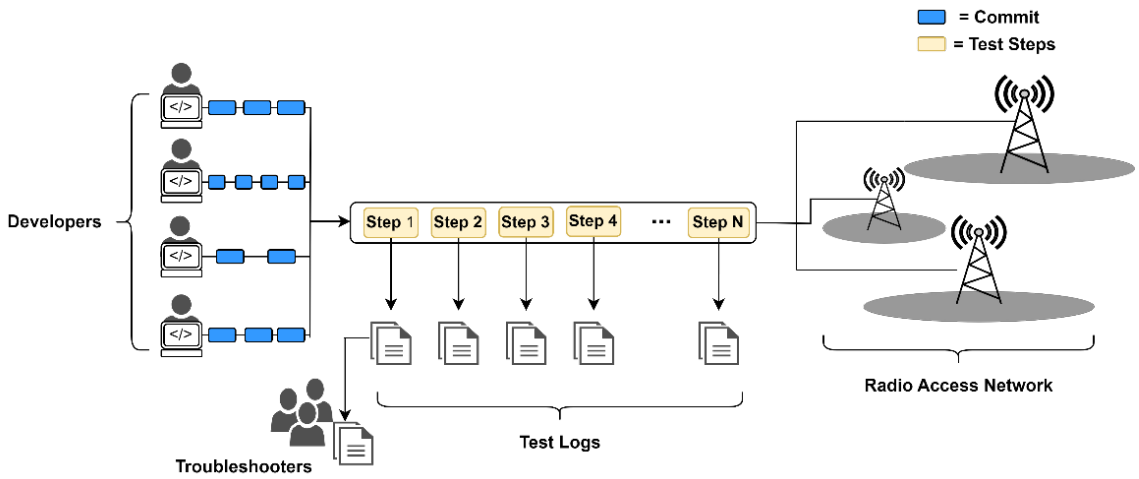

Рисунок 1.2 – Демонстрація потоку доставки ПЗ.

1.3 Мета та завдання дослідження

Метою цієї магістерської роботи є дослідження методів машинного навчання (МН) для виявлення аномалій у сигналах багатовимірних часових рядів (MVTS) RAN та порівняння отриманих результатів із традиційними методами. Це, у свою чергу, може дати змогу розширити обсяг тестування, зменшивши кількість необхідних тестових сценаріїв. Такий підхід дозволить розробникам впроваджувати більш реалістичні сценарії тестування на ранніх етапах циклу розробки, що сприятиме забезпеченню високої якості продукту під час безперервної доставки коду.

Крім того, розширення сфери тестування допоможе скоротити кількість тестів, необхідних для перевірки цілісності системи, що забезпечить довгострокові й стабільні переваги. Важливо зазначити, що навчання будь-якої моделі МН потребує ресурсів, що може мати певний негативний вплив на довкілля. Однак очікується, що довгострокові переваги перевищать початкові витрати.

Нарешті, розроблення цієї концепції виявлення аномалій не викликає жодних етичних питань. Усі дані, що використовуються, генеруються виключно в середовищі досліджень і розробок (R&D), і для створення та навчання моделі МН не залучаються жодні дані користувачів.

1.4 Методологія дослідження та обмеження

Вибір методології дослідження базується на нових висновках із різних наукових робіт. Як зазначалося в попередньому розділі, головна мета цього проєкту — дослідження методів на основі машинного навчання для створення системи виявлення аномалій, здатної визначати ненормальну поведінку в телекомунікаційних даних.

Важливо підкреслити, що хоча проєкт спирається на нові дослідження, його метою не є реалізація найсучаснішого методу, який вирішує всі можливі проблеми. Тому цей проєкт слід розглядати радше як перший крок до покращення виявлення та усунення несправностей у RAN, а не як загальне дослідження.

У роботі основну увагу приділено простим архітектурам моделей, таким як згорткові нейронні мережі (CNN), довготривалі короткочасні пам’яті (LSTM) та моделі автокодування. Вивчення більш складних архітектур може бути досліджене у майбутніх роботах.

Протягом роботи терміни «помилка» та «аномалія» можуть використовуватися як синоніми. У тестуванні програмного забезпечення «помилка» зазвичай означає основну причину проблеми. Важливо зазначити, що цей проєкт зосереджується передусім на виявленні та ідентифікації ненормальної поведінки системи, а не на аналізі її причин.

Крім того, зібрані дані не мають розмітки, і в межах цього проєкту анотація даних неможлива. Тому основна увага приділяється вивченню підходів без нагляду (unsupervised learning).

Висновки

У першому розділі було розглянуто ключові проблеми, пов’язані з виявленням збоїв у радіомережах доступу (RAN). Аналіз показав, що:

1. RAN є критично важливим компонентом телекомунікаційної інфраструктури, який забезпечує з’єднання користувацького обладнання з ядром мережі, проте його складність зростає разом із розвитком технологій 5G.
2. Традиційні методи діагностики, засновані на пороговому аналізі та ручному опрацюванні журналів, мають суттєві обмеження: вони трудомісткі, не здатні виявляти приховані або слабко виражені аномалії та не дозволяють точно визначати момент їх виникнення.
3. Методи машинного навчання відкривають нові можливості для автоматизації процесу виявлення збоїв, проте їх застосування стикається з низкою викликів: потребою у великих обсягах даних, необхідністю якісної розмітки та значними обчислювальними витратами.
4. У межах дослідження сформульовано чотири наукові питання, що стосуються ефективності інтеграції методів МН у тестову платформу RAN, порівняння продуктивності моделей різної складності, визначення мінімального обсягу даних для навчання та оцінки здатності МН покращувати точність визначення моменту появи аномалій.
5. Методологія дослідження орієнтована на використання простих архітектур моделей (CNN, LSTM, автоенкодери) та підходів без нагляду, що дозволяє зробити перший крок до створення більш надійної системи автоматизованого виявлення аномалій у RAN.

РОЗДІЛ 2
МЕТОДИ ВИЯВЛЕННЯ АНОМАЛІЙ У ЧАСОВИХ РЯДАХ

Цей розділ надає базову інформацію про виявлення аномалій, застосованих до часових рядів (ЧР). Він обґрунтовує необхідність використання глибинного виявлення аномалій, порівнюючи його з традиційними методами. Також тут розглядаються популярні підходи до виявлення аномалій у даних часових рядів: методи на основі прогнозування та методи на основі реконструкції. Крім того, розділ торкається суміжних досліджень у цій галузі.

2.1 Часові ряди

Часовий ряд (ЧР) можна розглядати як перелік зафіксованих вимірювань із зазначенням часу, коли ці значення були зафіксовані. Якщо потрібно відстежувати кілька метрик одночасно, такий ряд називають багатовимірним часовим рядом (БВЧР). БВЧР, окрім часових залежностей, часто демонструють також просторові залежності [5].

2.2 Аномалії

Аномалії можна визначити як закономірності, які не відповідають нормальній поведінці. Існують три типи аномалій: точкові аномалії, контекстуальні аномалії та колективні аномалії [6].

Точкові аномалії — найпростіший тип, який виникає, коли окремий елемент даних значно відрізняється від інших [6]. На рисунку 2.1 наведено приклад точкових аномалій.

Контекстуальні аномалії виникають у певному контексті. Це означає, що елемент даних може вважатися аномалією в одному контексті, але бути нормальним в іншому. Такий тип аномалій часто спостерігається в даних часових рядів [6]. Рисунок 2.2 ілюструє цю концепцію на прикладі набору даних ЧР, де один часовий штамп позначений як аномалія.

Колективні аномалії з’являються, коли група елементів даних поводиться

ненормально порівняно із загальним розподілом даних. Хоча окремі точки можуть виглядати нормальними, разом вони проявляють аномальну поведінку [6].

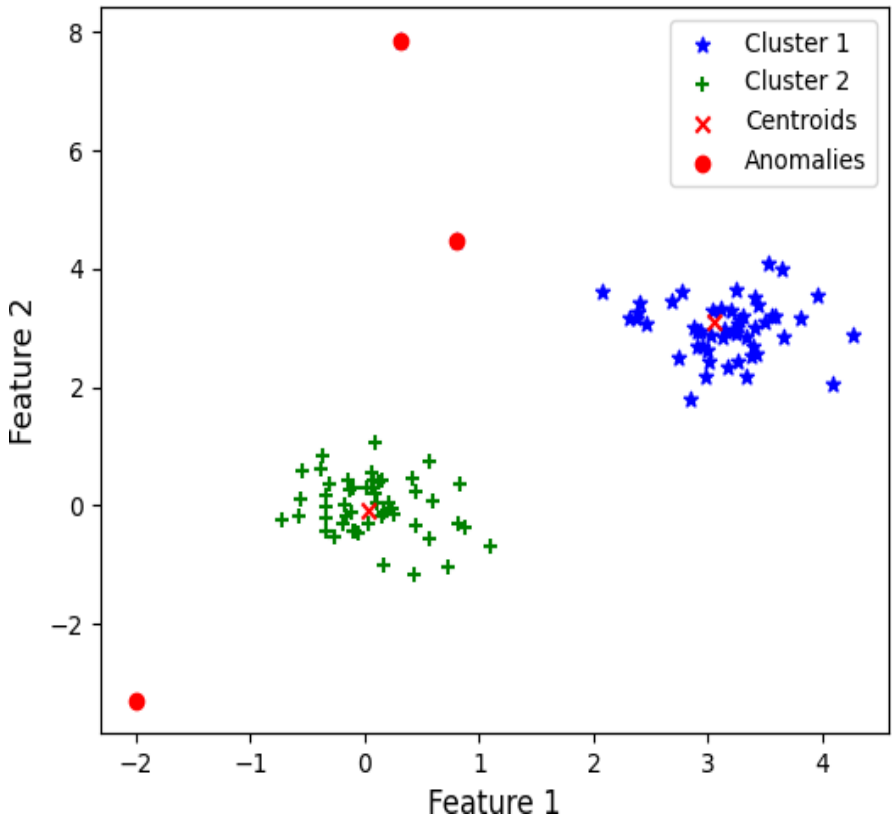

Рисунок 2.1 – Приклад точкових аномалій у двовимірному наборі даних.

2.3 Традиційне виявлення аномалій

Традиційні методи виявлення аномалій можна поділити на методи, засновані на близькості, параметричні та непараметричні [7].

У методах, заснованих на близькості, ми не робимо попередніх припущень щодо розподілу даних. Одним із найвідоміших таких методів є алгоритм k-близьких сусідів (k-NN). Зазвичай, для нового елемента даних алгоритм k-NN обчислює відстань між новою точкою та всіма іншими елементами у наборі даних. Вибираючи k найближчих сусідів, алгоритм визначає, чи є точка аномалією. Інший популярний підхід — K-means, який кластеризує дані на k груп. Кожен клас представлений прототипним вектором, що відповідає його середньому значенню. Якщо нова точка не потрапляє в

жоден клас, її можна вважати аномалією [7]. На рисунку 2.1 наведено приклад застосування K-means.

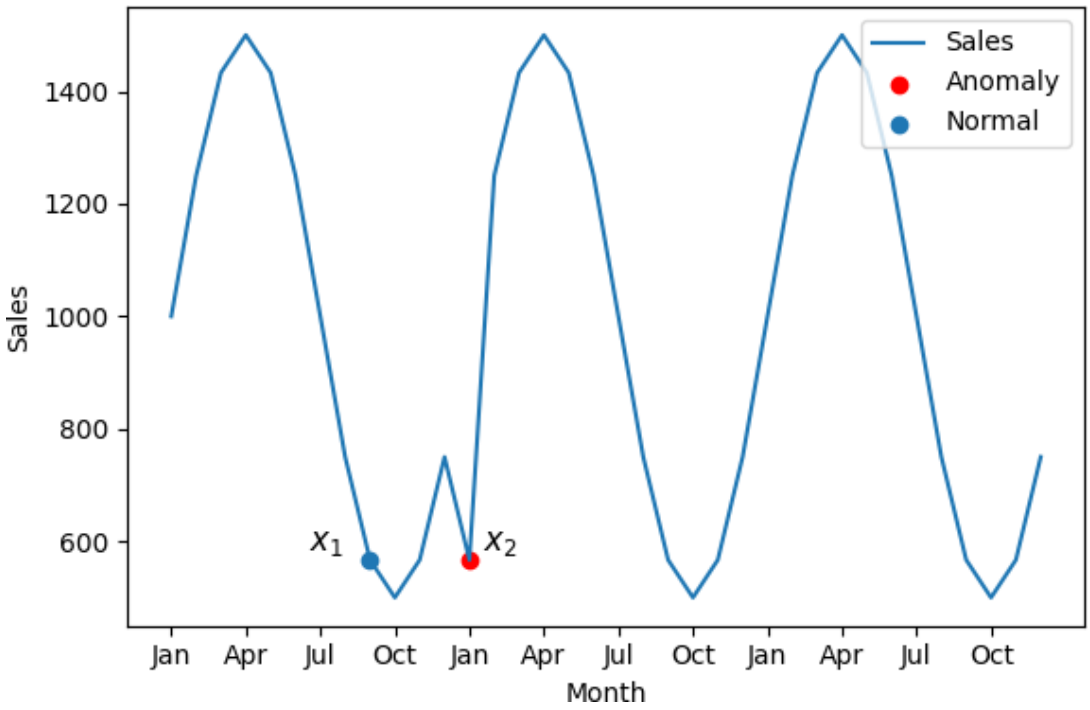

Рисунок 2.2 – Приклад контекстуальної аномалії в наборі даних часових рядів, де x2 визначається як контекстуальна аномалія. Зверніть увагу, що ми вважаємо аномалією саме x2, а не x1, незважаючи на однакове значення. Рисунок створено на основі джерела [6].

Недоліком методів, заснованих на близькості, є те, що складність алгоритму сильно залежить від розмірності даних і обсягу набору даних.

Параметричні методи підходять для великих наборів даних, оскільки складність моделі масштабовано незалежно від розміру даних. У параметричному підході ми моделюємо базовий розподіл нормальних даних і вважаємо аномаліями ті нові елементи, що значно відхиляються від моделі [7]. Ще одним прикладом методів, заснованих на близькості, є Minimum Volume Ellipsoid [8].

Хоча параметричні методи ефективні для деяких наборів даних, багато наборів не підпорядковуються певному розподілу, що робить ці методи неефективними у деяких випадках. Щоб вирішити цю проблему, були

запропоновані непараметричні методи, у яких не робиться жодних припущень щодо розподілу даних. Мета цього підходу — навчитися нормальній поведінці даних, уникаючи моделювання їхнього розподілу [7].

2.4 Глибинне виявлення аномалій

В останні роки спостерігається зростання популярності методів глибинного навчання, що вплинуло й на сферу виявлення аномалій [9]. У глибинному виявленні аномалій основна увага приділяється використанню методів навчання представлень (representation learning) для виявлення прихованих закономірностей у даних. Такий підхід дозволяє моделям машинного навчання навчатися безпосередньо на необроблених даних і автоматично виділяти закономірності, необхідні для виявлення аномалій [10]. Завдяки цьому моделі глибинного навчання ефективно працюють із складними та високорозмірними даними, такими як часові, просторові та графові набори даних [11].

Перехід до методів глибинного навчання виявився особливо важливим через обмеження традиційних технік, які часто є субоптимальними. Кілька методів глибинного навчання, запропонованих останніми роками, продемонстрували кращу ефективність у порівнянні з традиційними підходами [11]. Chalpathy та ін. [5] обґрунтовують цей перехід такими причинами:

- Традиційні алгоритми погано справляються з виявленням аномалій у зображеннях та послідовних даних через складність і внутрішні залежності цих наборів даних.
- Використання традиційних статистичних методів для великих обсягів даних майже неможливе.
- Глибинне навчання дозволяє ієрархічно навчати моделі прихованим закономірностям, що дає змогу автоматично виявляти аномалії безпосередньо з необроблених даних.

2.4.1 Контрольоване виявлення аномалій

Контрольоване (supervised) навчання — це галузь машинного навчання, яка спирається на дані з мітками для навчання моделей. Зазвичай у таких наборах даних кожен вхід має відповідний правильний вихід. Класифікація вважається одним із найпоширеніших підходів у контрольованому навчанні. Модель отримує на вхід дані та формує вектор оцінок для кожної наявної категорії, а мета полягає в тому, щоб бажана категорія отримала найвищий бал [10].

Таким чином, контрольоване виявлення аномалій можна спростити до бінарної класифікації з двома категоріями: «ненормально» та «нормально». Хоча цей підхід може підвищити ефективність виявлення аномалій, на практиці він часто є мало практичним через брак мічених навчальних даних [11][5].

2.4.2 Неконтрольоване виявлення аномалій

На відміну від контрольованого навчання, неконтрольоване (unsupervised) навчання не потребує мічених даних. Його мета — вивчити приховані закономірності або структури безпосередньо з необроблених даних [10].

У неконтрольованому виявленні аномалій підходи можна поділити на два типи: на основі прогнозування та на основі реконструкції. Для виявлення аномалій у часових рядах за допомогою цих підходів дані потрібно підготувати за допомогою техніки ковзного вікна (sliding window). Ця техніка передбачає створення фіксованого вікна довжиною L і поступове його зміщення на один часовий штамп, що утворює перекриваючі інтервали. Такий підхід покращує виявлення аномалій, дозволяючи зосередитися на конкретних ділянках необроблених даних [12].

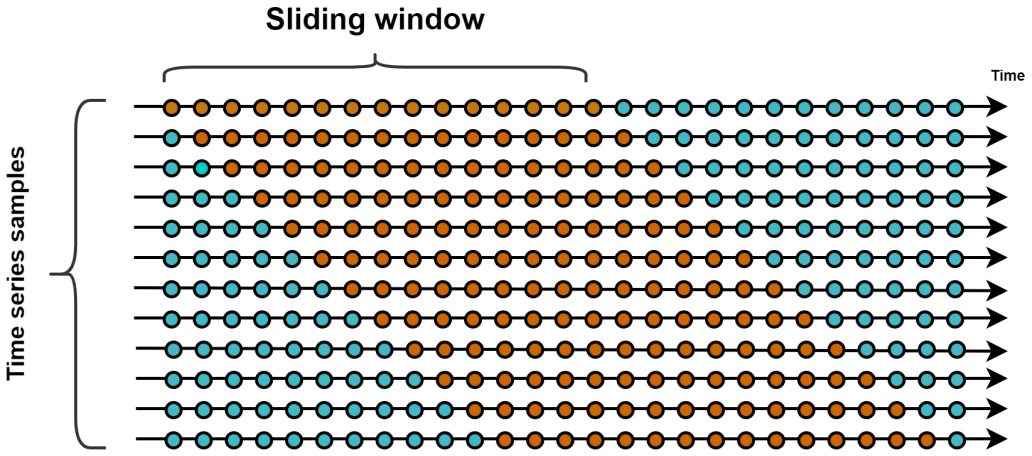

Рисунок 2.3 – Демонстрація методики ковзного вікна на даних часових рядів, що показує створення перекриваючихся інтервалів при кожному зсуві вікна. Рисунок створено на основі джерела [13].

2.4.2.1 Метод на основі прогнозування

Методи на основі прогнозування передбачають навчання моделі на послідовних даних для передбачення наступних послідовностей. Мета — передбачити наступний інтервал даних на основі поточного. Іншими словами, для даного інтервалу x_n, який є n-м інтервалом, створеним ковзним вікном, модель прогнозує наступний інтервал $\hat{x}_{n+1}$. *Якщо фактичний (n+1)-й інтервал* x_{n+1} значно відрізняється від прогнозу $\hat{x}_{n+1}$, його позначають як аномалію [14].

Підходи на основі рекурентних нейронних мереж (RNN), такі як LSTM та Gated Recurrent Unit (GRU), виявилися ефективними для врахування довгострокових історичних залежностей у даних, що важливо для прогнозування ознак. Хоча щільні глибинні нейронні мережі (DNN) також можна використовувати для прогнозування, вони можуть не справлятися з довгостроковими залежностями та обробкою високо комплексних часових даних [15].

2.4.2.2 Метод на основі реконструкції

У підході на основі реконструкції модель вивчає представлення даних і приховані закономірності, кодуючи вхідні дані в латентний простір на фазі

кодування та відтворюючи їх на фазі декодування. Тобто, для інтервалу x_n модель генерує відновлений вихід $\hat{x}_n$. Як і у методі прогнозування, якщо фактичний вхід x_n суттєво відрізняється від відновленого $\hat{x}_n$, його позначають як аномалію [14].

Архітектура автоенкодера (AE) є одним із найпопулярніших підходів для вивчення прихованих закономірностей у даних через реконструкцію. AE складається з енкодера та декодера. Вхідні дані подаються на AE, де енкодер стискає їх у латентний простір, після чого декодер відновлює дані для отримання виходу AE. Під час навчання відновлений вихід порівнюється з оригінальним вхідним сигналом, і обчислюється помилка за заданим критерієм. Ця помилка використовується для оновлення вагів через зворотне поширення [16].

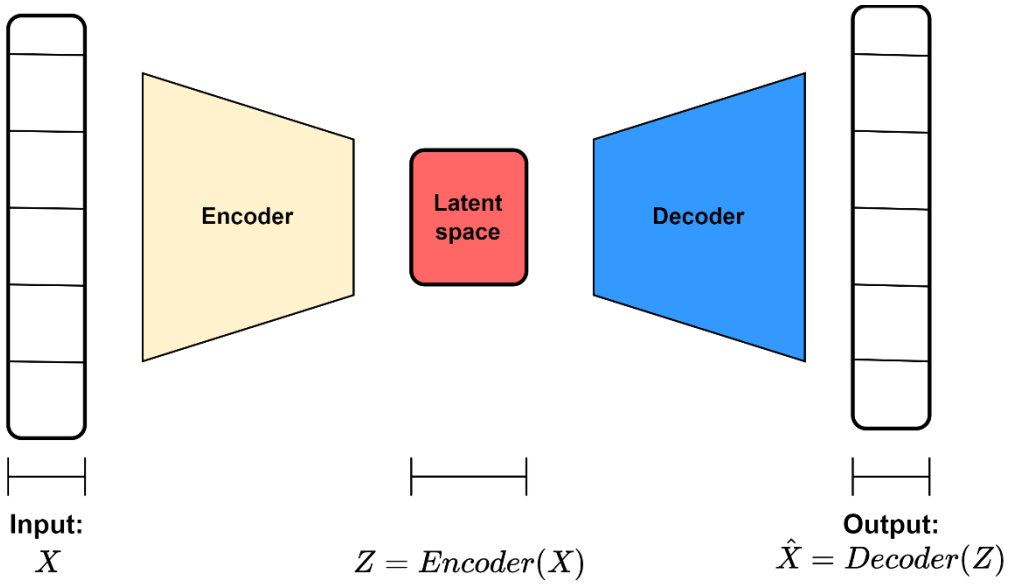

Рисунок 2.4 – Схема архітектури AE: вхід X стискається енкодером у латентний простір Z та відновлюється декодером у $\hat{X}$

Важливо зазначити, що AE не просто копіює вхід у вихід. Натомість вхід стискається до латентного простору, при цьому зберігаються ключові ознаки. Використовуючи це латентне представлення, модель намагається відновити оригінальний вхід. AE можна будувати на основі повнозв’язних шарів, CNN і

навіть RNN-архітектур, таких як LSTM [16].

2.5 Пов’язані дослідження

Виявлення аномалій відіграє ключову роль у виявленні та запобіганні помилок у різних сферах. Багато реальних способів застосуваня, таких як виявлення вторгнення, виявлення шахрайства, виявлення шкідливого ПЗ, медичний контроль і оцінка ризиків, безпека ШІ та управління ризиками, покладаються на методи виявлення аномалій. Зростаючий попит на ефективні рішення сприяв розвитку великої кількості наукових праць та методів, що вирішують пов’язані з цією проблемою завдання.

Наприклад, у сфері виявлення шахрайства Marco Schreyer та співавт. [17] запропонували глибоку нейронну мережу AE для виявлення шахрайства у великих бухгалтерських даних. Для виявлення шкідливого ПЗ було досліджено кілька методів із застосуванням CNN та AE, зокрема роботи Kolosnjaji та співавт. [18], Suciu та співавт. [19], Srisakaokul та співавт. [20], Mohammadi та співавт. [21], Yousefi-Azar та співавт. [22]. У медичному виявленні аномалій застосовували підходи на основі RNN. Наприклад, Siddique Latif та співавт. [23] досліджували використання Bidirectional LSTM (BiLSTM), LSTM та GRU для аналізу серцевих тонів з метою виявлення ненормального серцебиття.

Окрім того, виявлення аномалій у часових рядах (ЧР) привернуло значну увагу останніми роками. Було зроблено численні спроби вирішити цю проблему за допомогою різних методів машинного навчання [14]. Для одновимірних ЧР Markus Thill та співавт. [2] запропонували нову архітектуру темпорального AE, що використовує CNN для вивчення часових закономірностей у електрокардіограмних записах пацієнтів із серцевою аритмією. Hundman та співавт. [15] застосували мережу на основі LSTM разом із динамічним пороговим відсіканням, створюючи одноканальні моделі для кожного ЧР замість використання однієї моделі для всіх, щоб виявляти аномалії космічних апаратів за даними Mars Science Laboratory (MSL) та Soil Moisture Active Passive (SMAP).

Хоча ці підходи показали обнадійливі результати, дослідники довели, що використання багатовимірних ЧР (БВЧР) є більш надійним для виявлення аномалій у системі в цілому порівняно з одновимірними моделями для окремих метрик [14, 24, 25]. Причини цього такі: по-перше, раптові відхилення однієї метрики не обов'язково свідчать про збій усієї системи; по-друге, важливішим є загальний стан системи, ніж окремої метрики; по-третє, навчання окремої моделі для кожної ознаки метрики є трудомістким і витратним за часом; нарешті, використання просторових залежностей БВЧР дає більше інформації та забезпечує надійніше виявлення аномалій при аналізі всіх метрик разом [14, 24].

У БВЧР H.D. Nguyen та співавт. [16] використали мережу LSTM для прогнозування, поєднану з LSTM-AE та однокласовими машинами опорних векторів, для виявлення аномалій у продажах, що сприяло кращому прийняттю рішень у управлінні ланцюгами постачання. Tung Kieu та співавт. [26] запропонували підхід ансамблевого навчання з розрідженими AE для виявлення аномалій у одновимірних та багатовимірних ЧР, досліджуючи два способи навчання AE: незалежне навчання та спільну оптимізацію у стилі багатозадачного навчання.

Модель OmniAnomaly [24], стохастична RNN, показує, що стохастичні зв'язки покращують надійність захоплення закономірностей у БВЧР. Модель використовує варіаційний автоенкодер (VAE) для вивчення представлень даних. Hang Zhao та співавт. [14] представили модель MTAD-GAT, яка включає два шари графової уваги разом із моделями на основі реконструкції та прогнозування. Вони виявили, що поєднання обох підходів зі спільною оптимізацією дає кращу ефективність. Подібно до цього, Chaoyue Ding та співавт. [27] досліджували виявлення аномалій у БВЧР за допомогою мережі, що інтегрує графове навчання, графову мережу уваги та моделі реконструкції і прогнозування через спільну оптимізацію, стверджуючи, що їхнє рішення перевершує всі попередні базові моделі. Yuxin Zhang та співавт. [3] також використали методи спільної оптимізації для вирішення задачі виявлення

аномалій у багатосенсорних даних.

Згадані дослідження застосовували різні методи для виявлення аномалій у часових рядах. Проте, на думку автора, жодне з них спеціально не розглядало виявлення помилок у RAN на основі обмежених даних, зібраних у R&D середовищі. Більшість підходів зосереджувалися на створенні складних сучасних моделей, часто нехтуючи потенційною ефективністю простіших моделей. Тому метою цієї роботи є часткове усунення цих недоліків.

Висновки

У другому розділі було систематизовано основні підходи до виявлення аномалій у часових рядах:

1. Часові ряди є базовим типом даних для телекомунікаційних систем, а їх багатовимірні варіанти (БВЧР) дозволяють враховувати як часові, так і просторові залежності, що робить їх більш інформативними для аналізу.
2. Аномалії поділяються на точкові, контекстуальні та колективні, кожен тип має власні особливості й потребує різних методів для ефективного виявлення.
3. Традиційні методи (засновані на близькості, параметричні та непараметричні) забезпечують базовий рівень діагностики, проте їх ефективність обмежена високою розмірністю даних, відсутністю універсального розподілу та складністю масштабування.
4. Глибинне виявлення аномалій демонструє значні переваги завдяки здатності автоматично виділяти приховані закономірності у даних, працювати з великими та складними наборами й забезпечувати більш високу точність порівняно з традиційними підходами.
5. Серед методів глибинного навчання виділяються два основні напрями:
6. Методи прогнозування (LSTM, GRU), що дозволяють виявляти відхилення між очікуваними та фактичними значеннями;
7. Методи реконструкції (автоенкодери), які визначають аномалії на основі різниці між відновленими та реальними даними.

Таким чином, розділ показав, що традиційні методи мають обмеження у складних телекомунікаційних середовищах, тоді як глибинні моделі відкривають нові можливості для точного та масштабованого виявлення аномалій у часових рядах. Це створює підґрунтя для подальших експериментів і практичного застосування методів машинного навчання у сфері RAN.

РОЗДІЛ 3
ОГЛЯД МЕТОДІВ ДОСЛІДЖЕННЯ

Цей розділ надає огляд методів дослідження, використаних у цій роботі. У розділі 3.1 розглядаються техніки збору та обробки даних, застосовані для цього дослідження. Розділ 3.2 описує запропоновані методи виявлення аномалій та загальну архітектуру моделі. Нарешті, у розділі 3.3 пояснюються методи оцінки ефективності запропонованих моделей виявлення аномалій.

3.1 Дані

Дані для цієї роботи взяті з відкритих джерел на GitHub, та включають різні показники та метрики, що дозволяють переглядати ключові показники продуктивності RAN. У цій роботі увага зосереджена на вибірці метрик у вигляді багатовимірних часових рядів (MVTS). Платформа фіксує значення цих метрик кожні 30 секунд. Часові ряди мають періодичну поведінку, збільшуючись до часу скидання, який відбувається кожні 15 хвилин. Таким чином, для кожного періоду очікується наявність 30 вимірювань певної метрики.

Важливо зазначити, що через конфіденційний характер цього набору даних детальні пояснення метрик надавати неможливо, проте будуть представлені форми сигналів (waveforms).

Рисунок 3.1 ілюструє три MVTS. На графіку видно залежність між точками даних: кожне значення залежить від попередніх. Цю часову залежність виділено зеленим кольором. Крім того, раптові зміни в одному часовому ряді можуть впливати на значення та поведінку інших рядів — це просторова залежність, яка виділена жовтим кольором, показуючи, як різні TS впливають один на одного.

Для навчання та тестування запропонованих моделей виявлення аномалій були створені два окремі набори даних. Навчальний набір містить нормальну поведінку системи, тоді як тестовий набір включає вручну введені аномалії для оцінки ефективності моделі. Важливо, щоб навчальний набір був вільний від аномалій, інакше модель могла б сприймати аномалії як нормальну поведінку,

що призвело б до неспроможності їх виявлення. Характеристики наборів даних наведено в таблиці 3.1.

Щодо збору даних існує багато етичних аспектів. Як зазначалося у Розділі 1, усі дані для цього проекту були зібрані виключно в R&D середовищі. Жодні дані користувачів не використовувалися на жодному етапі проєкту, що забезпечує дотримання всіх етичних стандартів і гарантує, що персональна інформація користувачів не обробляється і не доступна.

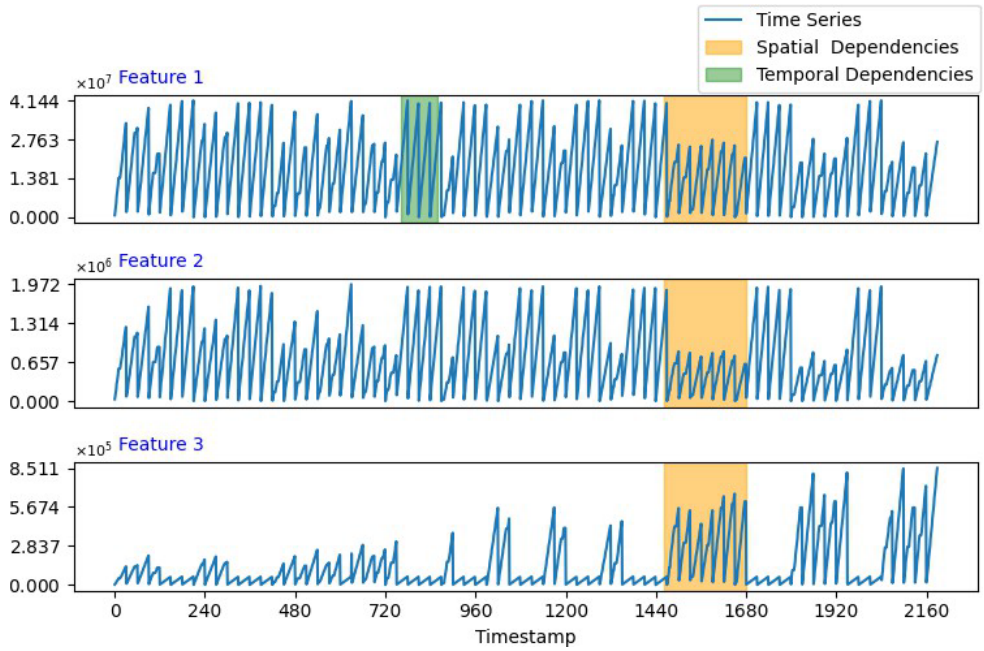

Рисунок 3.1 – Демонстрація даних у вигляді MVTS. Часові та просторові залежності виділено відповідно зеленим і жовтим кольорами.

Таблиця 3.1 – Опис зібраних даних

Набір даних	Тривалість	Періодичність	Кількість ознак	Довжина часового ряду	Відсоток аномалій
Навчальний (Train)	17 годин	15 хв	68	1950	0%
Тестовий (Test)	19 годин	15 хв	68	2190	56%

3.1.2 Масштабування

Щоб підвищити стійкість моделі та забезпечити рівний внесок усіх ознак,

застосовується масштабування Min-Max (Min-Max Scaling). Ця техніка нормалізує кожну ознаку, переводячи її в фіксований інтервал [0,1]на основі мінімального та максимального значень із навчального набору.

Формула масштабування виглядає так:

$$x' = \frac{x - \min(X_{\text{train}})}{\max(X_{\text{train}}) - \min(X_{\text{train}})} \tag{3.1}$$

де $\max(X_{\text{train}})$та $\min(X_{\text{train}})$— відповідно максимальні та мінімальні значення у навчальному наборі даних.

3.1.3 Пропущені дані

Моделі виявлення аномалій часто чутливі до викидів у навчальному наборі даних [14]. Пропущені точки даних можуть суттєво вплинути на ефективність моделі та її здатність виявляти аномалії.

Для вирішення цієї проблеми використовується лінійна інтерполяція, яка дозволяє оцінити значення відсутньої точки на основі двох відомих точок.

Формула лінійної інтерполяції має вигляд:

$$\text{Slope} = \frac{y_2 - y_1}{x_2 - x_1}, y = y_1 + \text{Slope} \times (x - x_1) \tag{3.2}$$

де (x_1, y_1)та (x_2, y_2)— значення відомих точок даних.

3.2 Метод виявлення аномалій

Для виявлення аномалій було використаго дані, описані в попередньому розділі, щоб навчити різні архітектури моделей. Як вже зазначалося, MVTS зазвичай мають просторові та часові залежності. Щоб врахувати це, ми застосували як підходи на основі прогнозування, так і підходи на основі реконструкції.

3.2.1 Модель прогнозування

Було розроблено Bidirectional Forecasting (BiFo) для захоплення часових залежностей як від минулого до теперішнього, так і від теперішнього до минулого. Такий підхід дозволяє моделі вловлювати складніші закономірності у часових рядах, що робить модель більш стійкою для виявлення аномалій.

Модель складається з двох LSTM-шарів, після яких йде dense-шар для отримання вихідних даних. Ми експериментували з різною кількістю прихованих шарів для досягнення кращої продуктивності — деталі наведені у розділі 4.1.

Під час навчання модель отримує дані у вигляді міні-батчів, що складаються з фіксованих інтервалів, створених за допомогою ковзного вікна (sliding window). Прогнози моделі порівнюються з тими ж міні-батчами, але з інтервалами, зсунутими на один крок, для обчислення функції втрат Mean Square Error (MSE).

Формула MSE:

$$\text{MSE} = \frac{1}{n \cdot k} \sum_{i=1}^{n} \sum_{j=0}^{k} (\hat{x}_j^{(i)} - y_j^{(i)})^2 \tag{3.3}$$

де $\boldsymbol{n}$ – розмір міні-батчу, $\boldsymbol{k}$ – розмір ковзного вікна, а $\boldsymbol{y}$ – міні-батч з інтервалами, зсунутими на один часовий крок. Для $\boldsymbol{x^{(i)}} = [\boldsymbol{x_1}, \boldsymbol{x_2}, \ldots, \boldsymbol{x_k}]$маємо $\boldsymbol{y^{(i)}} = [\boldsymbol{x_2}, \boldsymbol{x_3}, \ldots, \boldsymbol{x_{k+1}}]$. Обчислене значення функції втрат використовується для оновлення параметрів моделі.

Після навчання моделі для виявлення аномалій обчислюється евклідова відстань між прогнозами моделі та фактичними значеннями (зсунуті інтервали). Якщо це значення перевищує певний поріг, відповідна мітка часу позначається як аномалія.

Формула для розрахунку оцінки:

$$\text{Score} = \boldsymbol{s_i} = \sqrt{(\boldsymbol{y_i} - \widehat{\boldsymbol{x}}_{\boldsymbol{i}})^2} \quad \rightarrow \quad \begin{cases} \boldsymbol{s_i} > \text{Threshold}, & \text{Аномалія} \\ \boldsymbol{s_i} \leq \text{Threshold}, & \text{Норма} \end{cases}, \#(\mathbf{3.4})$$

Методика визначення порогу буде детально описана у розділі 3.2.4.

3.2.2 Модель на основі реконструкції

Основним підходом для моделі на основі реконструкції стало використання архітектури автоенкодера (Autoencoder, AE). Було створено дві різні варіації автоенкодерів: Convolutional Autoencoder (CAE) та LSTM Autoencoder (LAE).

У LAE для кодування в латентний простір використовувалися три LSTM-шари, а для декодування назад — два шари. У CAE застосовувалися два конволюційні шари з max pooling для фази кодування та три конволюційні шари для декодування. Рисунок [3.2] – ілюструє цю архітектуру.

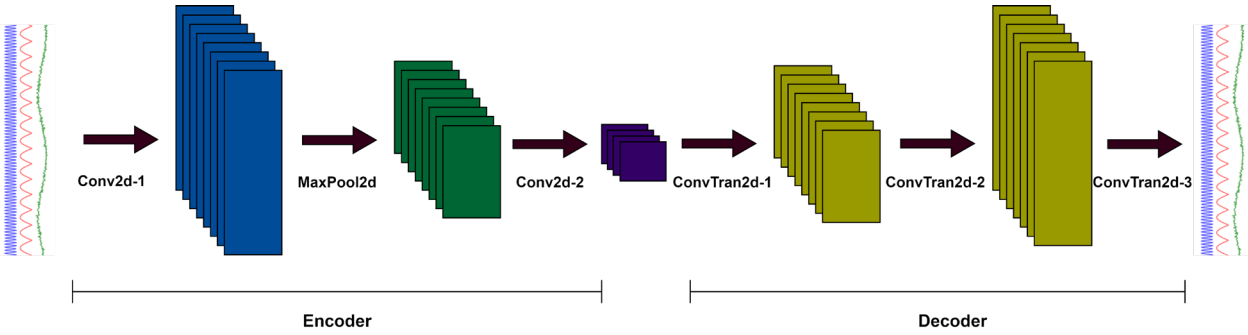

Рисунок 3.2 – архітектура моделі CAE.

Процес навчання та прогнозування цих моделей схожий на модель BiFo, з однією відмінністю: під час обчислення MSE та оцінки (score) ми порівнюємо прогноз моделі з входом самого автоенкодера, а не з зсунутим інтервалом. Мета цих моделей — реконструювати той самий вхід.

Таким чином, використовуються ті самі рівняння, що й у формулах 3.3 та 3.4, але з заміною $\boldsymbol{y_j^{(i)}}$ на $\boldsymbol{x_j^{(i)}}$ та $\boldsymbol{y_i}$ на $\boldsymbol{x_i}$.

3.2.3 Спільна оптимізація

Щоб об’єднати підходи на основі прогнозування та реконструкції, було розроблено мульти-модельну архітектуру CAE_BiFo, яка інтегрує CAE та BiFo моделі. Її оптимізація здійснюється за допомогою стратегії спільної оптимізації, натхненної роботами [14][27].

Для обчислення функції втрат мульти-моделі ми просто комбінуємо втрати кожної складової моделі:

$$\text{Loss} = \text{Loss}_{\text{for}} + \text{Loss}_{\text{rec}}, \qquad (\mathbf{3.5})$$

За отриманим значенням втрат параметри обох моделей оновлюються одночасно.

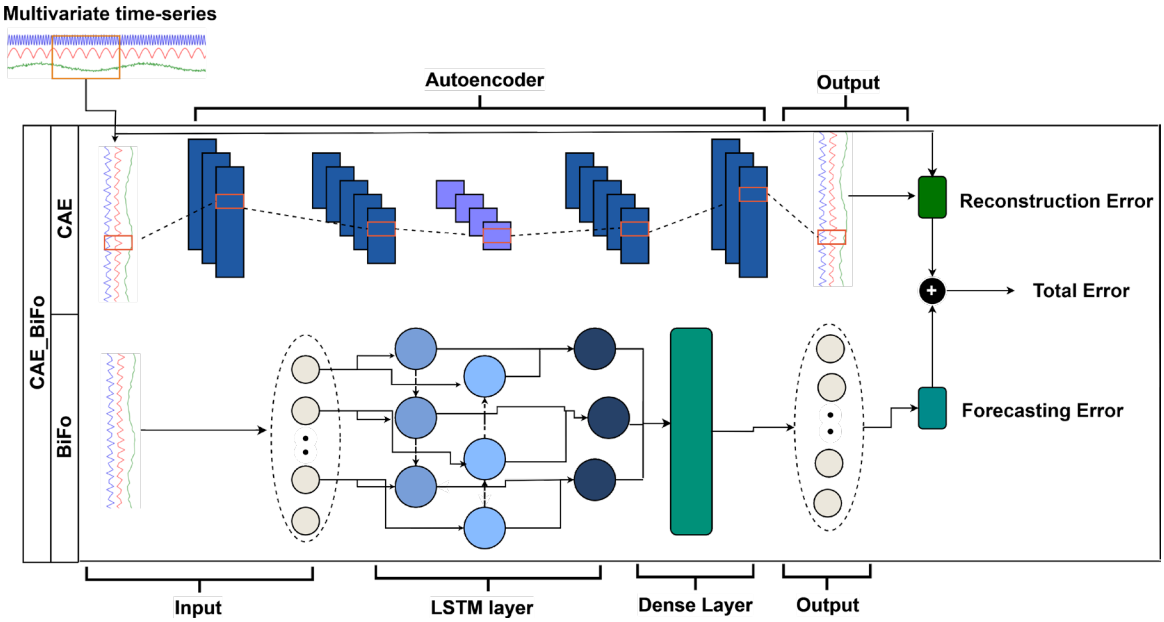

Рисунок 3.3 – ілюструє архітектуру моделі CAE_BiFo та детально показує процес обчислення втрат.

Для прогнозування оцінка (score) обчислюється аналогічно формулі 3.4 для кожної моделі. Загальний score визначається як сума індивідуальних оцінок:

$$\text{Score} = \text{Score}_{\text{for}} + \boldsymbol{\gamma} \cdot \text{Score}_{\text{rec}}, \qquad (\mathbf{3.6})$$

де $\boldsymbol{\gamma}$— гіперпараметр, який дозволяє контролювати внесок оцінки реконструкції у загальну оцінку.

Ідея використання такого підходу була запозичена з роботи [14].

3.2.4 Поріг

Поріг зазвичай обчислюється на основі розподілу оцінок (score) навчального набору. Для навчальних даних $\boldsymbol{x}$та прогнозів моделі $\widehat{\boldsymbol{x}}$можна обчислити помилку для кожної мітки часу $\boldsymbol{t}$за формулою 3.4. Це дає вектор помилок:

$$e = [e_0, e_1, \dots, e_{t-1}, e_t]$$

Поріг визначається шляхом виявлення екстремальних випадків у навчальному наборі, які отримали значні значення помилок. Оскільки навчальний набір вважається вільним від аномалій, його розподіл оцінок зазвичай нижчий, ніж у наборах із аномаліями. Вибираючи ці екстремальні випадки, ми забезпечуємо ефективне виявлення аномалій при мінімізації хибнопозитивних спрацювань.

У цій роботі застосовуються два методи визначення порогу: z-score та динамічний поріг, запропонований у [15].

Метод z-score передбачає, що розподіл помилок слідує гаусовому закону. Обчислюється z-score для заданого рівня значущості $\boldsymbol{\alpha}$:

$$\text{zscore} = \boldsymbol{\Phi}^{-1}\left(\mathbf{1} - \frac{\boldsymbol{\alpha}}{\mathbf{2}}\right), \qquad (\mathbf{3.7})$$

де $\boldsymbol{\Phi}$— стандартний нормальний розподіл $\boldsymbol{N}(\mathbf{0}, \mathbf{1})$. Далі z-score масштабується під наш розподіл помилок:

$$\epsilon = \mu(e) + \text{zscore} \cdot \sigma(e), \tag{3.8}$$

де $\boldsymbol{\epsilon}$— поріг, $\boldsymbol{\mu(e)}$та $\boldsymbol{\sigma(e)}$— середнє та стандартне відхилення вектора помилок.

Хоча метод z-score ефективний, його припущення про розподіл помилок може бути ненадійним. Для подолання цього було запропоновано динамічний підхід [15]. Він заснований на тій же логіці, що і z-score, але замість обчислення z-score використовує набір позитивних значень $\boldsymbol{z}$, що представляють кількість стандартних відхилень вище середнього вектора $\boldsymbol{e}$. Автори пропонують брати $\boldsymbol{z}$у межах від 2 до 12 для кращої продуктивності, хоча конкретні значення залежать від даних.

Алгоритм динамічного порогу:

1. На кожній ітерації обирається нове $\boldsymbol{z}$між 2 та 12 (або інші значення, залежно від задачі).
2. Обчислюється новий поріг: $\boldsymbol{\epsilon = \mu(e) + z \cdot \sigma(e)}$.
3. Значення в $\boldsymbol{e}$, що менші за поріг, відсікаються.
4. Обчислюється оцінка для нового порогу:

$$\text{score} = \frac{\mu(e) - \mu(e_{\text{pruned}})}{\mu(e)} + \frac{\sigma(e) - \sigma(e_{\text{pruned}})}{\sigma(e)}$$

5. Вибирається поріг з найвищою оцінкою.

3.3 Оцінка

Продуктивність моделі оцінюється на основі ручного маркованого тестового набору. Ефективність запропонованих моделей вимірюється за допомогою precision, recall та F1-score. Формули для цих метрик:

$$\text{Precision} = \frac{TP}{TP + FP}; \tag{3.9}$$

$$\text{Recall} = \frac{TP}{TP + FN}; \tag{3.10}$$

$$\text{F1-score} = \frac{\mathbf{2} \cdot \text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}}. \tag{3.11}$$

де $\boldsymbol{TP}$, $\boldsymbol{FP}$ та $\boldsymbol{FN}$— це True Positives, False Positives та False Negatives відповідно.

Виявлення всіх аномалій у наборі даних може бути складним завданням. У реальних застосуваннях аномалії часто з'являються у суцільних сегментах, тому важливішим є виявлення наявності аномалії в сегменті, а не точне виявлення кожної точки. Тому достатньо, щоб модель виявила будь-яку точку всередині сегмента аномалії, якщо затримка виявлення невелика [28][14].

Ми застосовуємо підхід із [28], де весь сегмент позначається як позитивний, якщо хоча б одна точка всередині нього була правильно виявлена. Тобто, якщо модель визначає точку в сегменті аномалії з затримкою менше $\boldsymbol{k}$від початку сегмента, всі точки цього сегмента вважаються правильно виявленими.

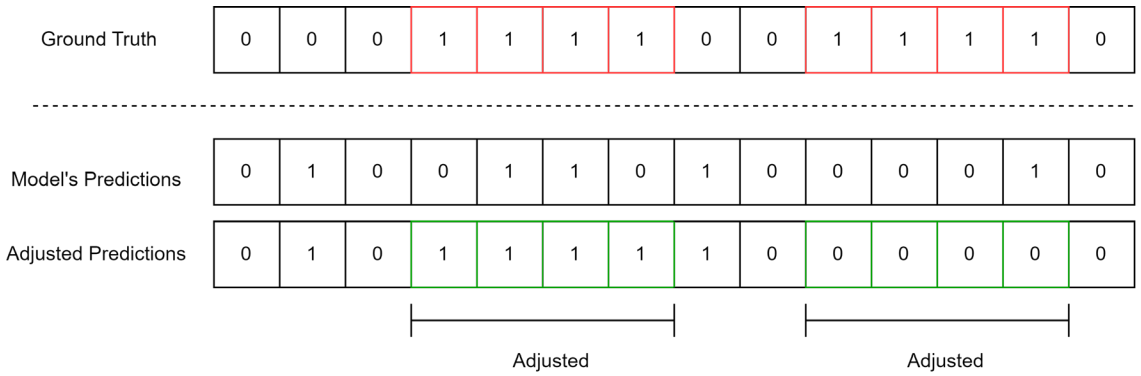

Рисунок 3.4 – ілюструє цю стратегію:0 — нормальна точка; 1 — аномалія; $\boldsymbol{k} = \mathbf{2}$. У першому сегменті всі точки стають 1, оскільки модель виявила аномалії всередині допустимої затримки $\boldsymbol{k}$. У другому сегменті точки залишаються 0 через помилку прогнозу.

Висновки

У третьому розділі було представлено методологічну основу дослідження, яка включає:

1. Дані: використано багатовимірні часові ряди (MVTS), що відображають ключові показники продуктивності RAN. Для навчання та тестування моделей створено окремі набори даних, де навчальний набір містить лише нормальну поведінку системи, а тестовий — вручну введені аномалії. Це забезпечує коректність навчання та можливість об’єктивної оцінки моделей.
2. Попередня обробка даних: застосовано масштабування Min-Max для нормалізації ознак та лінійну інтерполяцію для заповнення пропущених значень, що підвищує стійкість моделей до викидів і забезпечує рівний внесок усіх метрик.
3. Методи виявлення аномалій: розглянуто три підходи — прогнозування (BiFo), реконструкція (CAE, LAE) та спільна оптимізація (CAE_BiFo). Кожен із них враховує часові та просторові залежності даних, що дозволяє ефективніше виявляти відхилення від нормальної поведінки.
4. Визначення порогу: описано методи обчислення порогових значень для класифікації аномалій, зокрема z-score та динамічний поріг, які забезпечують баланс між точністю та мінімізацією хибнопозитивних спрацювань.
5. Оцінка моделей: ефективність підходів оцінюється за метриками precision, recall та F1-score. При цьому враховується практична специфіка — важливість виявлення аномалії в межах сегмента, а не лише окремих точок.

РОЗДІЛ 4
РЕЗУЛЬТАТИ ТА АНАЛІЗ ЕФЕКТИВНОСТІ

У цьому розділі представлені отримані результати та їхній аналіз. Розділ 4.1 описує процес вибору оптимальних гіперпараметрів. Розділ 4.2 демонструє продуктивність моделей та порівнює її з методами на основі правил. Розділи 4.3 та 4.4 показують вплив різних порогів та обсягу даних на ефективність моделей. Розділ 4.5 представляє результати виявлення аномалій.

4.1 Налаштування гіперпараметрів

Вибір гіперпараметрів є важливою складовою будь-якої задачі машинного навчання, оскільки безпосередньо впливає на продуктивність моделі. Ці параметри контролюють різні аспекти процесу навчання та задаються перед його початком. Правильний вибір гіперпараметрів може значно підвищити ефективність та стійкість моделі.

Для пошуку оптимальної комбінації гіперпараметрів у цьому дослідженні застосовано grid search. Алгоритм тренує та оцінює модель для кожної комбінації параметрів і виводить результати у форматі журналу. Потім ці результати можна проаналізувати, щоб обрати комбінацію, що дає найвищий F1-score. Псевдокод алгоритму наведено у Додатку С.

Особливу увагу приділено моделі CAE_BiFo, яка поєднує підходи на основі реконструкції та прогнозування. Це дозволяє оптимізувати обидва аспекти через налаштування гіперпараметрів. Результати grid search наведені в Таблиці 4.1. Тут:

- Gamma — гіперпараметр для інтеграції оцінок прогнозування та реконструкції (обговорювалося у 3.2.3).
- Hidden layer size — розмір LSTM-шару у моделі прогнозування. Детальна архітектура моделі наведена у Додатку А. Значення параметрів для пошуку було обрано на основі рекомендацій у різних дослідженнях [3, 13, 14, 16].

Таблиця 4.1: Підсумки grid search

Параметри	Діапазон пошуку	Обране значення
Learning rate	[1e-4, 1e-1]	1e-3
Number of Epochs	{10, 25, 50}	50
Gamma	[0.1, 1]	0.1
Hidden layer size	{25, 50, 100}	100

Вплив алгоритму grid search показаний на Рисунку 4.1, де кожна ітерація відповідає певній комбінації гіперпараметрів, що використовувалася для тренування та оцінки моделі. Після налаштування гіперпараметрів продуктивність моделі зросла майже на 5% за F1-score, а значення MSE loss помітно зменшилося.

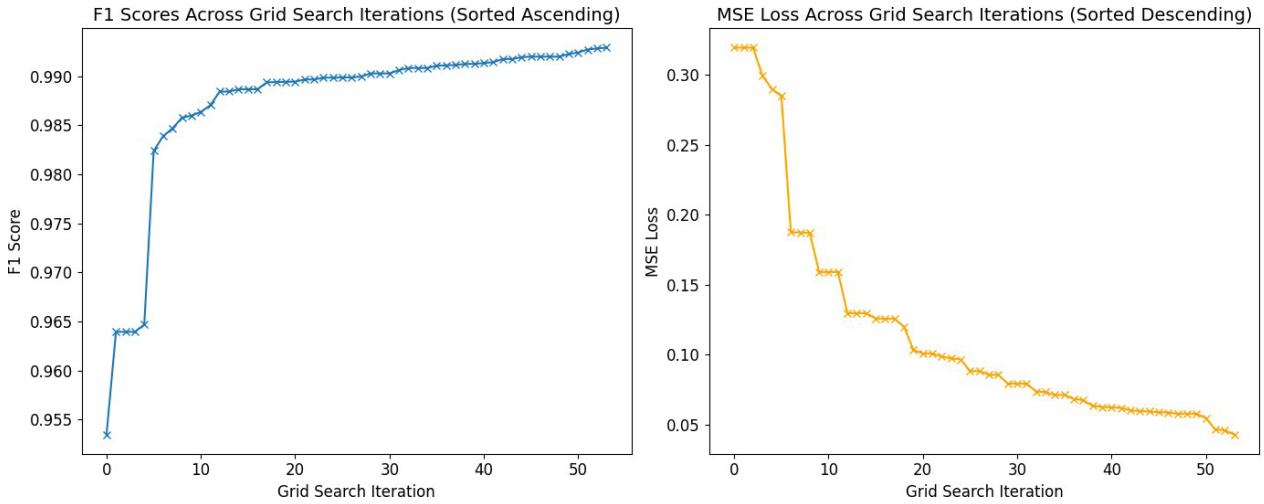

Рисунок 4.1 – Значення F1-score та MSE loss для кожної ітерації grid search, відсортовані за продуктивністю.

4.2 Ефективність моделей

Одним із головних завдань цієї роботи було розробити кілька моделей машинного навчання (МН) як демонстрацію можливості інтеграції МН-підходів у процес усунення несправностей RAN.

Щоб оцінити ефективність запропонованих моделей і порівняти їх із

правиловими методами, моделі було навчено та протестовано на тому ж наборі даних, описаному в розділі 3.1. Для порівняння використовувався метод **Z-score**, який застосовувався для обчислення порогових значень. Отримані результати були зіставлені з ефективністю методів виявлення збоїв на основі порогів. Такий метод вважає увесь період аномальним, якщо пікове значення падає нижче розрахованого порогу, який визначається на основі середнього піку всього часового ряду. Варто зазначити, що тестувалися різні порогові значення, і було обрано результат із найкращими показниками.

Таблиця 4.2 – Результати роботи різних моделей і базового правилового методу на наборі даних RAN.

Моделі машинного навчання	**Precision**	**Recall**	**F1-score**
CAE	0.9869	0.9897	0.9883
LAE	0.9882	0.9996	0.9939
BIFO	0.9845	0.9949	0.9897
CAE_BIFO	0.9895	0.9963	0.9929
Метод на основі правил			
Пороговий метод	0.5498	0.9520	0.6970

Усі 68 ознак було використано для навчання й оцінювання МН-моделей. Водночас правиловий підхід обмежується одноваріантним виявленням аномалій, тобто може аналізувати лише один часовий ряд за раз. Це демонструє ще одну перевагу МН-підходів — окрім їхньої вищої точності, вони можуть працювати з багатовимірними даними.

Через відносну простоту набору даних RAN було складно визначити, яка модель показала найкращий результат — відмінності між значеннями F1 були мінімальними й проявлялися лише в третій десятковій цифрі. Щоб обійти це обмеження, було використано набір даних Server Machine Dataset (SMD), уперше представлений Су та співавт. [24]. Цей відкритий набір часто використовується у дослідженнях [29, 30, 31] і вважається одним із найбільших

для оцінювання методів виявлення аномалій у багатовимірних часових рядах. Його структура є значно складнішою, ніж у наборі RAN. Детальний опис SMD наведено в додатку D.

Через обмеження часу було протестовано лише моделі CAE, LAE та CAE_BiFo, результати яких порівнювалися з OmniAnomaly [24] — однією з провідних базових моделей для цього набору даних.

Таблиця 4.3 – Результати роботи моделей та базової OmniAnomaly на наборі даних SMD.

Модель	Precision	Recall	F1-score
CAE	0.9508	0.4059	0.5689
LAE	0.8516	0.4311	0.5724
CAE_BIFO	0.9030	0.6730	0.7712
OmniAnomaly	0.8334	0.9449	0.8857

Аналіз результатів:

- Загалом, ефективність МН-моделей на наборі RAN є дуже високою, що пояснюється простотою цього набору. Попри це, МН-підходи значно перевищують точність порогового методу.
- Хоча пороговий метод демонструє високий показник Recall, він має низьку точність (Precision), що призводить до великої кількості хибних спрацьовувань і, відповідно, низького значення F1.
- Модель CAE_BiFo показала найвищий F1 на наборі SMD — лише на 11% нижчий, ніж у найкращої еталонної моделі OmniAnomaly. Це свідчить, що спільна оптимізація (joint optimization) може підвищити результативність на складних наборах даних.
- Враховуючи час навчання та інференсу, можна зробити висновок, що для наборів, подібних до RAN, достатньо простішої моделі, як-от CAE. Натомість для більш складних даних ефективнішою є складніша архітектура, така як CAE_BiFo, що підтверджено результатами на SMD.

Отже, ці результати дозволяють відповісти на перші два дослідницькі питання, поставлені в розділі 1.2. Методи машинного навчання перевершують традиційний правиловий підхід. Для складних наборів даних із численними залежностями більші моделі демонструють кращу ефективність, що видно на прикладі CAE_BiFo, яка перевершила CAE і LAE на наборі SMD.

Варто додати, що набір SMD використовувався виключно для оцінювання ефективності моделей у цьому експерименті, тоді як решта досліджень у цьому розділі базується лише на наборі RAN.

4.3 Вплив різних порогів

Для виявлення аномалій, як описано в розділі 3.2.4, було використано два підходи до визначення порогів — Z-score та динамічний поріг (Dynamic Threshold). У таблиці 4.4 наведено результати порівняння ефективності трьох моделей при використанні цих двох методів.

Таблиця 4.4 – Порівняння продуктивності моделей із динамічним і Z-score порогами.

Модель	**Dynamic Threshold**			**Z-score Threshold**		
	Precision	**Recall**	**F1-score**	**Precision**	**Recall**	**F1-score**
CAE	0.9952	0.7079	0.8273	0.9869	0.9897	0.9883
LAE	0.9912	0.9908	0.9910	0.9882	0.9996	0.9939
CAE_BIFO	0.9915	0.9521	0.9714	0.9895	0.9963	0.9929

З отриманих результатів можна зробити висновок, що динамічний поріг надає пріоритет точності (precision) перед повнотою (recall), тоді як підхід Z-score, навпаки, віддає перевагу більш високому показнику recall, залишаючись при цьому досить точним загалом.

Цю відмінність добре видно на рисунку 4.2, який показує значення оцінок тестового набору даних поряд із порогами для обох методів. Із рисунка видно, що динамічний поріг має вищі значення, ніж Z-score, і позначає як аномальні

лише ті приклади, що мають найвищі оцінки, ігноруючи решту. Такий підхід забезпечує більш точне виявлення аномалій, проте може призвести до збільшення кількості помилкових негативних результатів (false negatives).

Хоча значення F1-score для порогу Z-score виявилося вищим, не можна однозначно стверджувати, що цей метод є кращим у всіх випадках, адже його ефективність значною мірою залежить від конкретного набору даних і розподілу значень у ньому. У нашому випадку саме метод Z-score показав себе ефективнішим.

4.4 Вплив розміру даних

Завжди важливо знати, скільки даних потрібно для навчання моделі машинного навчання, щоб вона працювала ефективно. Генерація великого обсягу даних може бути дорогою для компаній і не завжди є найбільш стійким підходом. Тому було проведено експеримент з використанням двох моделей, щоб спостерігати за їхньою поведінкою з різною кількістю зразків даних під час етапу навчання. Результати показано на рисунку 4.3.

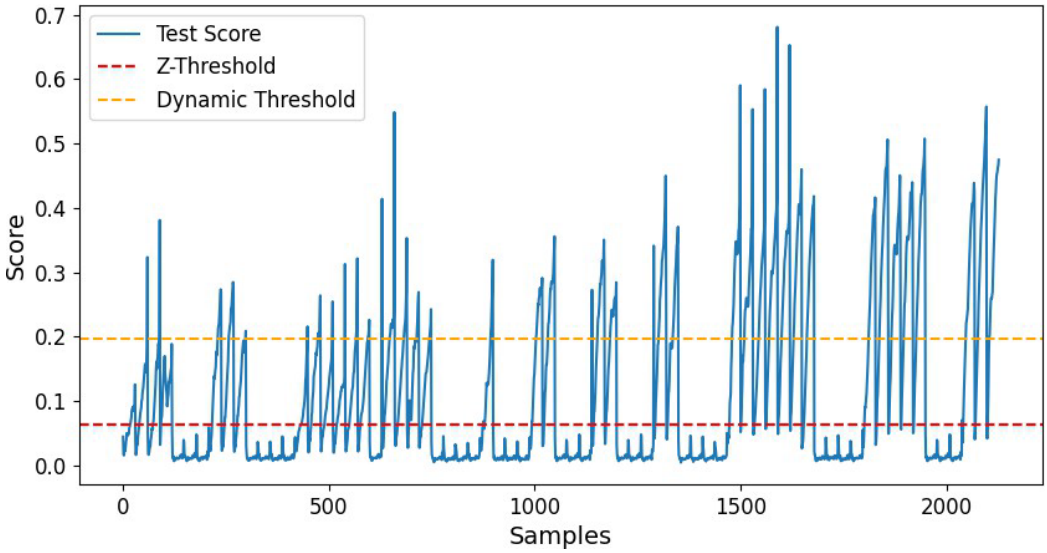

Рисунок 4.2 – Розподіл балів помилок для тестового набору даних, передбаченого моделлю CAE-BiFo, поряд із Z-балами та динамічними пороговими значеннями.

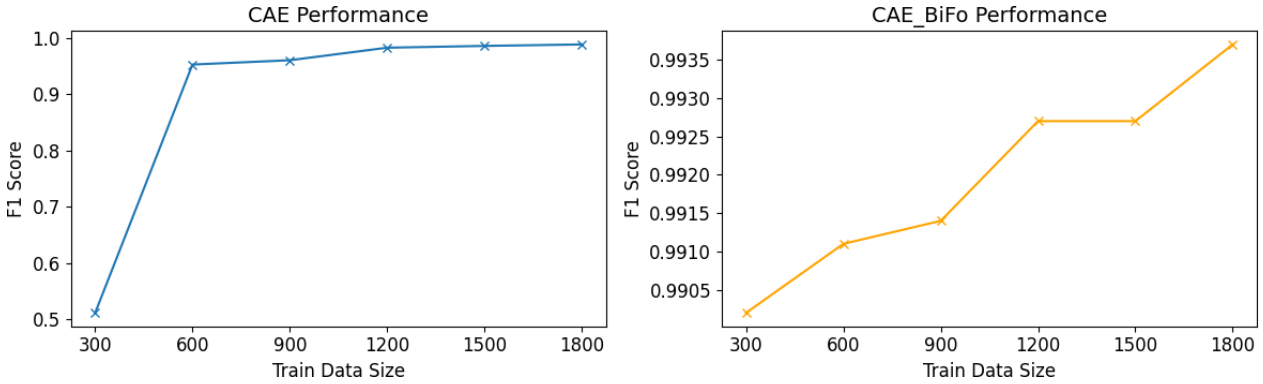

Рисунок 4.3 – Вплив розміру даних на продуктивність моделей CAE та CAE_BiFo.

Як і очікувалося, більший обсяг даних, як правило, призводить до кращої продуктивності. Однак у випадку моделі CAE_BiFo це підвищення продуктивності не є особливо помітно. Для моделі CAE, окрім винятку в першому випадку з 300 навчальними зразками, продуктивність не змінюється істотно при збільшенні обсягу даних. Можна зробити висновок, що завдяки простоті та низькій варіативності набору даних 600 зразків є достатніми для навчання ефективної моделі виявлення аномалій.

4.5 Виявлення аномалій

У цьому розділі показано, як модель CAE_BiFo виявляє аномалії в наборах даних RAN і SMD. На рисунках 4.4 і 4.5 виділено аномалії, виявлені моделлю, а аномальні області позначено червоним кольором. Ці рисунки ілюструють, що модель успішно виявляє аномальну поведінку у всіх TS. Крім того, модель може ідентифікувати початок і кінець кожної аномалії, що неможливо за допомогою методу, заснованого на правилах. Важливо зазначити, що через конфіденційний характер набору даних RAN фактичні назви показників у цьому наборі даних не включені до рисунка 4.4.

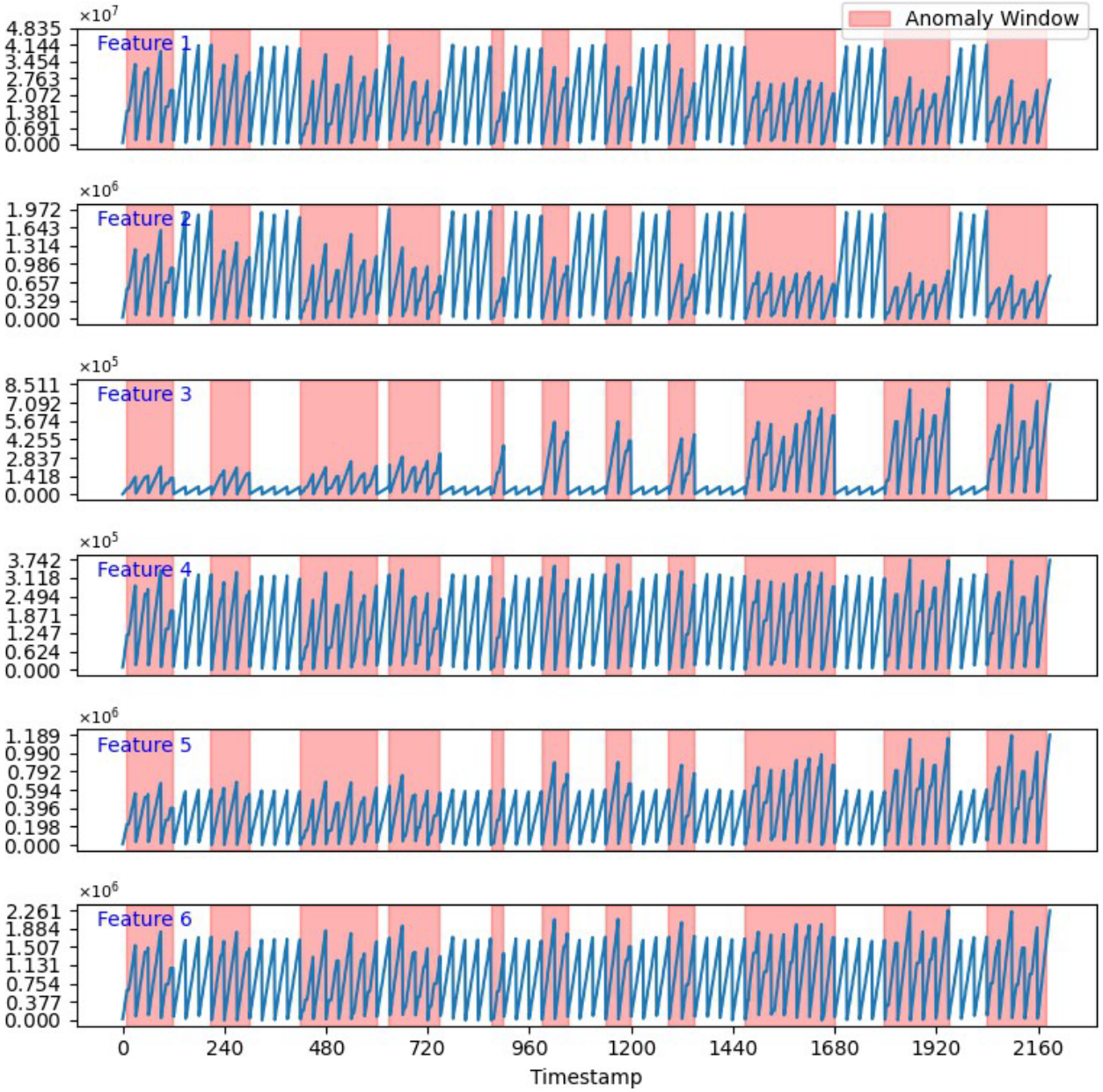

Рисунок 4.4 – Демонстрація виявлення аномалій, виконаного моделлю CAE_BiFo на наборі даних RAN. Зверніть увагу, що для візуалізації було використано лише 6 із 68 TS.

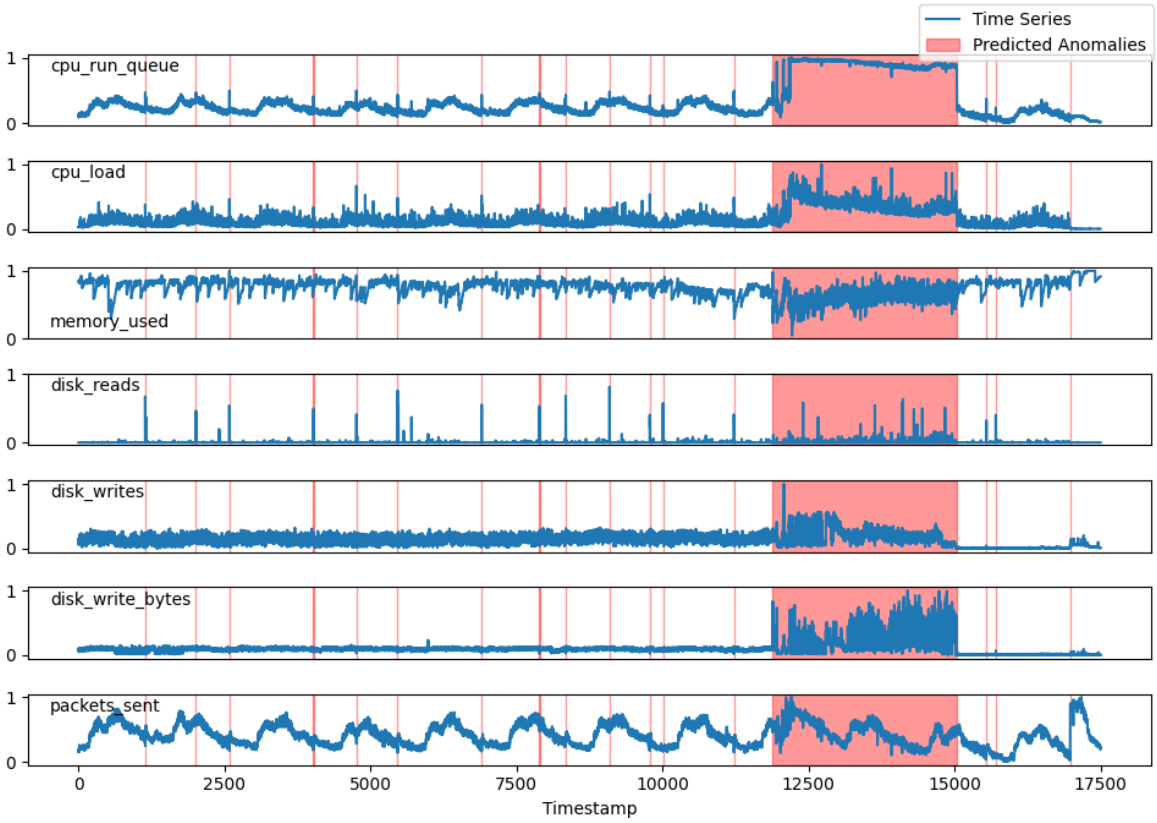

Рисунок 4.5 – Демонстрація виявлення аномалій, виконаного моделлю CAE_BiFo на наборі даних SMD. Зверніть увагу, що для візуалізації було використано лише 7 з 38 TS.

РОЗДІЛ 5

СТАРТАП-ПРОЕКТ НА ОСНОВІ ЗАСОБІВ ШТУЧНОГО ІНТЕЛЕКТУ ДЛЯ ВИЯВЛЕННЯ ЗБОЇВ У RAN

5.1 Ідея стартап-проєкту та її актуальність

Це запропонований стартап-проект, який буде базуватися на тому, що я дізнався у своїй магістерській дисертації. Він передбачає розробку інтелектуального програмного забезпечення AI-RAN Guard. Платформа автоматично виявлятиме та інформуватиме про випадки несправностей у мережі радіодоступу (RAN) шляхом аналізу багатовимірних часових рядів (MVTS). Актуальність стартапу зумовлена такими факторами:

- зростанням складності RAN (особливо у мережах 5G);
- великими обсягами телеметричних даних, непридатних для ручного аналізу;
- обмеженою ефективністю порогових та правилових методів;
- потребою операторів у скороченні часу простою мережі (MTTR).

Стартап орієнтований на використання неконтрольованих моделей машинного навчання (BiFo, CAE, LAE, CAE_BiFo), досліджених у попередніх розділах, що дозволяє впроваджувати рішення без потреби у розмічених даних.

5.2 Опис продукту та функціональних можливостей

AI-RAN Guard являє собою модульну програмну платформу, яка інтегрується у середовище тестування або експлуатації RAN.

Основні функції продукту:

- збір та агрегація MVTS-даних із RAN;
- попередня обробка даних (масштабування, інтерполяція);
- виявлення аномалій на основі прогнозування та реконструкції;
- динамічне визначення порогів;
- візуалізація аномалій та звітність;

- API для інтеграції з CI/CD та OSS/BSS системами.

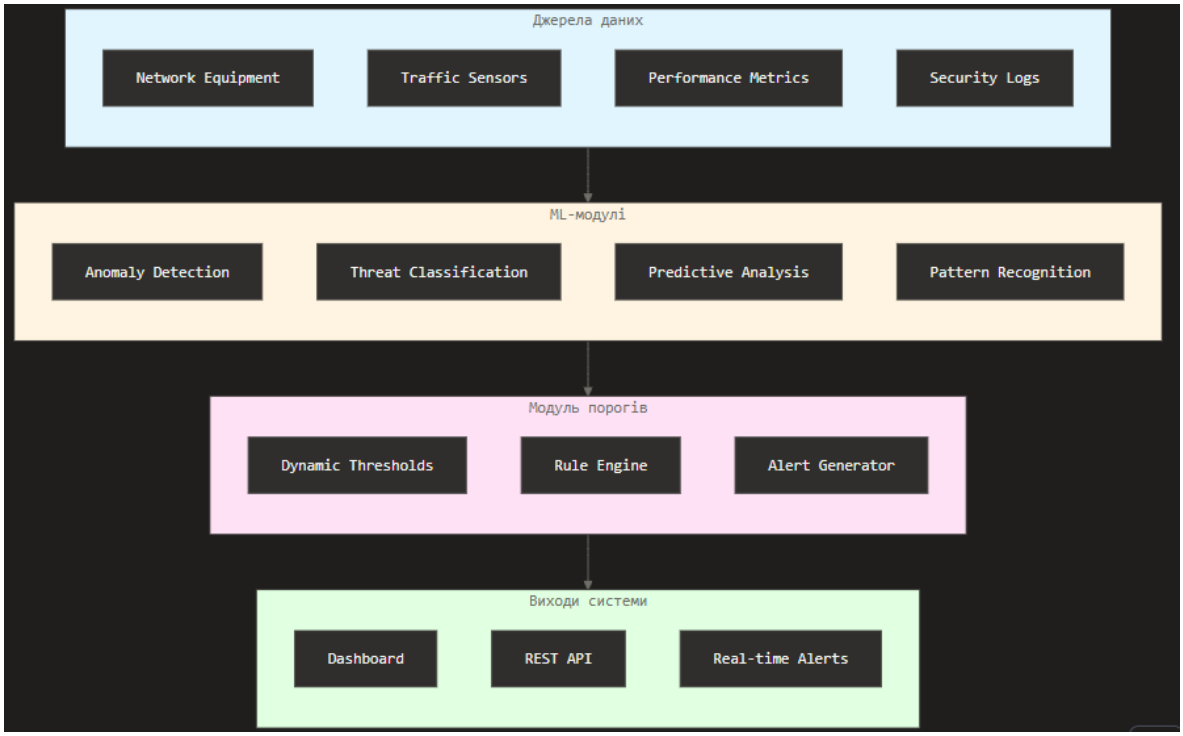

Рисунок 5.1 – Загальна архітектура стартап-продукту AI-RAN Guard (діаграма: джерела даних → ML-модулі → модуль порогів → dashboard/API).

Після етапу збору даних система виконує їх нормалізацію та перевірку цілісності. Це дозволяє зменшити вплив шуму та пропущених значень, що є критично важливим для стабільної роботи моделей машинного навчання. На цьому етапі реалізуються підходи, описані у розділі 3, зокрема масштабування Min-Max та лінійна інтерполяція.

Далі дані передаються до ядра аналітики, яке складається з набору моделей виявлення аномалій. Модель прогнозування BiFo використовується для оцінки відхилень між очікуваною та реальною поведінкою системи, тоді як автоенкодери (CAE, LAE) аналізують здатність системи відновлювати нормальну поведінку сигналів. Поєднання цих підходів у мульти-модельній архітектурі CAE_BiFo забезпечує більш надійне виявлення як різких, так і поступових аномалій.

Оцінки, отримані від моделей, передаються до модуля визначення порогів, де застосовуються статистичні та динамічні методи. Це дозволяє адаптувати систему до змінних умов роботи RAN без ручного переналаштування параметрів.

Останнім етапом є представлення результатів користувачу. Через веб-інтерфейс або API оператор отримує інформацію про наявність аномалій, їх тривалість та приблизний момент виникнення. Це дозволяє швидко усунути проблемні ділянки мережі та скоротити час реагування на збої.

5.3 Ринок, цільова аудиторія та бізнес-модель стартапу

Ринок телекомунікаційних рішень для моніторингу та забезпечення надійності мереж характеризується високою складністю, довгими циклами впровадження та значною вартістю простоїв. Саме тому інструменти автоматизованого виявлення збоїв у RAN мають стабільний попит як на етапі експлуатації мережі, так і під час тестування програмного забезпечення.

Цільовими користувачами стартапу AI-RAN Guard є організації, які безпосередньо відповідають за проєктування, тестування або експлуатацію радіомереж доступу.

Таблиця 5.1 – Цільові сегменти ринку та їх потреби

Сегмент ринку	**Основні потреби**	**Цінність AI-RAN Guard**
Мобільні оператори	Зменшення простоїв, підвищення QoS	Раннє виявлення аномалій та скорочення MTTR
Вендори RAN	Масштабне тестування ПЗ	Автоматизація QA без розмітки даних
R&D підрозділи	Аналіз складних MVTS	Гнучке дослідження поведінки мережі
Private 5G	Висока надійність	Локальне (on-premise) розгортання

Дані, наведені в таблиці 5.1, показують, що всі цільові сегменти мають спільну проблему — складність ручного аналізу телекомунікаційних даних та обмеженість традиційних правилових підходів. Запропонований стартап-продукт створює універсальне рішення, яке може бути адаптоване під специфіку кожного сегмента без суттєвих змін архітектури.

Бізнес-модель стартапу орієнтована на формат **B2B**, що є типовим для телекомунікаційної галузі, з можливістю надання продукту як хмарного сервісу або як локального програмного рішення.

Таблиця 5.2 – Бізнес-модель стартап-проєкту AI-RAN Guard

Компонент	Опис
Ціннісна пропозиція	Автоматичне виявлення збоїв у RAN на основі ШІ
Клієнти	Telco-оператори, вендори, R&D центри
Канали поширення	Прямі B2B-продажі, технічні партнери
Джерела доходу	SaaS-підписка, on-premise ліцензії, консалтинг
Ключові ресурси	ML-моделі, обчислювальна інфраструктура, експерти
Структура витрат	Розробка, підтримка, R&D, хмарні ресурси

Таблиця 5.2 узагальнює ключові елементи бізнес-моделі та показує, що основна економічна цінність стартапу формується за рахунок інтелектуальної складової — моделей машинного навчання та накопиченої експертизи у сфері RAN. Такий підхід дозволяє масштабувати рішення без пропорційного зростання витрат.

5.4 Техніко-економічне обґрунтування

Техніко-економічна доцільність стартап-проєкту AI-RAN Guard підтверджується як результатами експериментальних досліджень, наведених у розділі 4, так і особливостями обраної бізнес-моделі. Використання неконтрольованих методів машинного навчання дозволяє суттєво зменшити

витрати на підготовку даних та залучення експертів для ручної розмітки, що є одним із ключових факторів економічної ефективності рішення.

Таблиця 5.3 – Порівняння AI-RAN Guard з традиційними підходами до моніторингу RAN

Критерій	**Традиційні правила**	**AI-RAN Guard**
Точність виявлення	Низька–середня	Висока
Масштабованість	Обмежена	Висока
Потреба в експертах	Висока	Низька
Адаптивність до змін	Низька	Висока
Готовність до 5G	Обмежена	Повна

Як видно з таблиці 5.3, новий стартап-продукт кращий за класичні методи у важливих технічних та експлуатаційних рисах. Це дає можливість потенційним клієнтам менше пропускати помилок (False Negatives) і скорочувати час на реагування до інцидентів - що прямо впливає на сервіс.

Разом з тим реалізація. стартапу пов'язана з ряд ризиків, що потрібно врахувати на етапі планування та збільшення.

Таблиця 5.4 – Основні ризики стартап-проєкту та шляхи їх мінімізації

Тип ризику	**Опис**	**Ймовірність**	**Вплив**	**Шляхи мінімізації**
Технічний	Дрейф даних у RAN	Середня	Високий	Періодичне перенавчання моделей
Інтеграційний	Сумісність з OSS/BSS	Середня	Середній	API та модульна архітектура
Ринковий	Довгий цикл продажів	Висока	Середній	Пілотні проєкти

Фінансовий	Витрати на обчислення	Середня	Середній	Оптимізація інференсу

Подальший розвиток стартапу передбачається поетапно, з поступовим розширенням функціональних можливостей платформи.

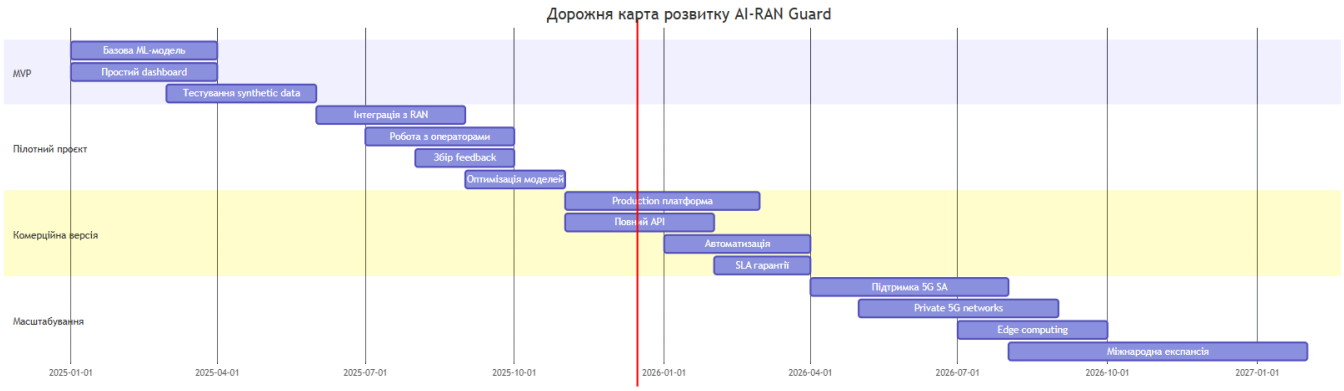

Рисунок 5.2 – Дорожня карта розвитку стартапу AI-RAN Guard (діаграма: MVP → пілотний проєкт → комерційна версія → масштабування на 5G SA та private 5G).

На першому етапі планується створення мінімально життєздатного продукту (MVP) з інтеграцією найефективнішої моделі CAE_BiFo. Другий етап передбачає перевірку рішення у вигляді проєкту з реальним або тестовим RAN-середовищем. На третьому етапі відбувається впровадження продукту у форматі SaaS або on-premise рішення. Завершальним етапом є масштабування платформи для підтримки мереж 5G Standalone та корпоративних private 5G інсталяцій.

Стартап AI-RAN Guard показує практичну цінність вивчених моделей машинного навчання і має шанс на комерційне впровадження у телекомунікаційній сфері.

Висновки

Хоча запропоновані рішення продемонстрували хорошу ефективність, можна стверджувати, що простішого рішення з використанням традиційних методів машинного навчання було б достатньо, зважаючи на простоту набору даних RAN. Однак у цьому дослідженні не були використані певні показники та характеристики. Включення цих показників до набору даних потенційно додає більшої складності, що може вимагати застосування підходів глибокого навчання, таких як той, що представлений у цьому дослідженні.

Використання набору даних SMD мало дві основні цілі: по-перше, це підвищило відтворюваність цього дослідження, оскільки набір даних є загальнодоступним. По-друге, набір даних RAN, використаний у цьому дослідженні, був неповним. Через обмеження в часі ми не змогли зібрати всі відповідні показники або створити повністю репрезентативний тестовий набір. Тому набір даних SMD був використаний для оцінки моделі та демонстрації того, що запропоновані методи також можуть виявляти аномалії в більш складних наборах даних.

Модель CAE_BiFo досягла значного F1-балу 0,77 на наборі даних SMD. Хоча ця модель не перевершила найефективніший еталонний показник OmniAnomaly, вона продемонструвала, що спільна оптимізація може покращити ефективність моделі. Також важливо зазначити, як згадувалося в попередніх розділах, що метою цього дослідження було не розробка найсучаснішого рішення, а надання доказу концепції, що методи машинного навчання можуть покращити виявлення аномалій у RAN.

Хоча найкращий показник F1, досягнутий на наборі даних RAN, становить 0,9939, що є значно високим, найбільш вражаючим є показник відкликання 0,9996. Це особливо важливо для виявлення аномалій, оскільки кілька помилкових спрацьовувань не становлять ризику для системи. Навпаки, помилкове неспрацьовування може призвести до невиявлення аномалії, що може спричинити серйозні проблеми або збій системи. Високий показник

відкликання гарантує виявлення майже всіх аномалій, тим самим забезпечуючи надійність і безпеку системи.

Однією з головних цілей цього проєкту було продемонструвати, як МН може автоматизувати виявлення аномалій у телекомунікаційних мережах. Це було досягнуто шляхом навчання моделі з використанням метрик MVTS від RAN. Після початкового навчання модель виявляє аномальні часові мітки без будь-яких заздалегідь встановлених порогових значень. Це стало можливим завдяки використанню багатовимірного виявлення аномалій. Ось чому зосередження уваги на багатовимірному підході є вигідним. На відміну від однофакторних методів, він фіксує просторові залежності та дозволяє навчати єдину модель для виявлення аномалій. Таким чином, це спрощує впровадження системи виявлення аномалій. Це значно ефективніше, ніж навчання декількох моделей для кожного TS, особливо коли набір даних містить численні TS.

Цікавим висновком цього дослідження стала кількість навчальних даних, необхідних для ефективного навчання моделі. Для ефективного навчання моделі виявлення аномалій нам потрібно лише 600 зразків даних з поточного набору даних RAN.

ЗАГАЛЬНІ ВИСНОВКИ ПО РОБОТІ

Це дослідження надало рішення для виявлення аномалій у RAN за допомогою методів МН. Ми досягли всіх наших цілей, незважаючи на обмеження, з якими зіткнулися. Нижче ми повернемося до кожного питання дослідження та розглянемо їх на основі наших висновків:

1. Наскільки інтеграція методів МН в тестову платформу RAN компанії може підвищити рівень виявлення несправностей у порівнянні з традиційними методами, заснованими на правилах, що вимірюються такими показниками, як точність, відтворюваність і F1-показник?

Модель LAE досягла F1-показника 0,9939 і відтворення 0,9996, тоді як модель CAE_BiFo досягла найвищої точності 0,9895 з використанням набору даних RAN. Для порівняння, підхід на основі правил дав точність 0,5498, відтворюваність 0,9520 і F1-показник 0,6970. Ці результати свідчать про те, що інтеграція технологій МН в платформу телекомунікаційних компаній може призвести до поліпшення виявлення несправностей.

2. Як порівнюється продуктивність малих моделей і моделей із більшою кількістю параметрів на наборі даних RAN з погляду точності та обчислювальної ефективності?

Хоча модель з найкращою продуктивністю мала відносно більший розмір параметрів, менші моделі, такі як CAE, також ідентифікували аномалії з високою точністю та відтворюваністю на наборі даних RAN. Однак більш складні моделі продемонстрували кращу продуктивність на наборі даних SMD, продемонструвавши свою здатність ефективно обробляти більш складні набори даних.

3. Який обсяг даних необхідний для ефективного навчання моделі МН для виявлення аномалій у MVTS?

Для набору даних RAN 600 навчальних зразків було достатньо для ефективного навчання моделей CAE та CAE_BiFo. Однак ця кількість може змінюватися для більш складних наборів даних, і для досягнення оптимальної

продуктивності може знадобитися більший навчальний набір.

4. Чи покращують системи, вдосконалені за допомогою МН, точність виявлення аномалій?

Виявлення початкової точки аномалії є надзвичайно важливим, і, як показано на рисунках 4.4 та 4.5, підхід, запропонований у цій роботі, може спростити виявлення аномалій у наборах даних MVTS RAN.

Основним обмеженням цього дослідження була доступність набору даних. Для значної частини проєкту для навчання моделі та проєктування архітектури моделі використовувалися синтетичні дані. Це було пов'язано з технічними складнощами, які не дозволили нам генерувати необхідні дані в першій половині проєкту. Тому ми покладалися на синтетичні дані, взяті з відкритих джерел GitHub. Лише в останньому кварталі дослідження нам вдалося отримати фактичний набір даних.

Крім того, створення тестового набору даних, що містив аномалії, було складним завданням. Ми мали вручну вносити аномалії в частини набору даних, одночасно забезпечуючи, щоб решта даних не була порушена. Часто це було неможливо, оскільки внесення аномалій впливало на весь набір даних. В результаті ми створили окремий набір даних без аномалій і об'єднали його з дефектним, щоб створити справедливий тестовий набір для оцінки моделі. Хоча такий підхід обмежував оцінку відомими аномаліями і не дозволяв нам імітувати реальні сценарії, ми не змогли придумати більш надійний метод для створення тестового набору.

Нарешті, маркування тестових даних, що містять аномалії, було складним завданням, оскільки точне маркування безпосередньо впливає на оцінку моделі. Ми покладалися на експертів у цій галузі, щоб правильно маркувати ці аномалії, але це знову обмежувало нас відомими аномаліями.

Хоча всі початкові цілі були досягнуті, деякі експерименти залишаються незавершеними через масштабність проблеми та обмеження в часі. Цей розділ

присвячений невирішеним питанням, які можуть бути розглянуті в майбутній роботі.

Хоча для навчання запропонованих моделей було використано багато ознак, все ще існують різні типи метрик, які можна витягти та використовувати для навчання моделей з метою створення більш надійної системи виявлення аномалій.

Крім того, окрім даних MVTS, для виявлення аномалій у RAN також можна використовувати журнали. Ці журнали, що містять інформацію про робочий стан мережі, можна обробляти за допомогою методів обробки природної мови (NLP). Ми можемо навіть поєднати ці журнали з даними TS, використаними в цьому дослідженні, для навчання моделі виявлення аномалій на основі трансформатора [32], яка включає як дані TS, так і дані журналів. Такий підхід дозволяє моделі ідентифікувати складні закономірності та аномалії, які можуть бути пропущені при розгляді даних TS або журналів окремо. Таким чином, підвищується надійність і точність виявлення аномалій в RAN.

Крім того, це дослідження можна розширити, включивши виявлення першопричини збоїв на основі виявлення аномалій. Це означає, що модель, окрім точного визначення аномалій, також може надавати інформацію про причини ненормальної поведінки та про те, які функції були найбільш уражені. Це також відкриє шлях до виявлення аномалій взаємодії функцій, що є однією з кінцевих цілей усього цього проєкту.

Нарешті, запропонований підхід може бути повністю впроваджений та інтегрований у платформу телекомунікаційних компаніях. Це може допомогти підвищити рівень автоматизованого виявлення аномалій та змінити підхід до навчання моделі з офлайн на онлайн. Це означає, що замість пакетного навчання ми могли б навчати модель у міру надходження нових даних, що зробило б модель більш адаптованою до змін у системі [33].

Досягнуті результати підтверджують, що поставлені цілі магістерської

роботи виконано. Запропоновані моделі та методи машинного навчання довели свою ефективність у виявленні аномалій у багатовимірних часових рядах RAN, забезпечивши значне підвищення точності порівняно з традиційними підходами. Отримані висновки свідчать про практичну цінність інтеграції ШІ у телекомунікаційні системи, адже це дозволяє не лише покращити якість обслуговування та оптимізувати використання мережевих ресурсів, але й створює основу для подальшого розвитку автоматизованих рішень у сфері виявлення та запобігання збоїв. Таким чином, результати роботи підтверджують перспективність застосування методів машинного навчання для підвищення надійності та стабільності сучасних телекомунікаційних мереж. . Усі поставлені завдання виконано, а мету роботи — досягнуто.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Satwaliya D. S., Thethi H. P., Dhyani A., Kiran G. R., Al-Taee M., Alazzam M. Predictive Maintenance using Machine Learning: A Case Study in Manufacturing Management // 2023 3rd International Conference on Advance Computing and Innovative Technologies in Engineering (ICACITE). – Greater Noida, India : IEEE, 2023. – С. 872–876. – DOI: 10.1109/ICACITE57410.2023.10183012. – [Електронний ресурс]. – Режим доступу: https://ieeexplore.ieee.org/document/10183012/

2. Thill M., Konen W., Wang H., Bäck T. Temporal convolutional autoencoder for unsupervised anomaly detection in time series // Applied Soft Computing. – 2021. – Т. 112. – Ст. 107751. – DOI: 10.1016/j.asoc.2021.107751. – [Електронний ресурс]. – Режим доступу: https://linkinghub.elsevier.com/retrieve/pii/S1568494621006724

3. Zhang Y., Chen Y., Wang J., Pan Z. Unsupervised Deep Anomaly Detection for Multi-Sensor Time-Series Signals. – 2021. – DOI: 10.48550/ARXIV.2107.12626. – [Електронний ресурс]. – Режим доступу: https://arxiv.org/abs/2107.12626

4. Sundqvist T. Machine learning-based diagnostics and observability in mobile networks : PhD Thesis. – Umeå : Umeå University, Department of Computing Science, 2023. – ISBN 978-91-8070-054-2. – [Електронний ресурс]. – Режим доступу: https://www.umu.se

5. Chalapathy R., Chawla S. Deep Learning for Anomaly Detection: A Survey. – 2019. – arXiv:1901.03407. – [Електронний ресурс]. – Режим доступу: http://arxiv.org/abs/1901.03407

6. Chandola V., Banerjee A., Kumar V. Anomaly detection: A survey // ACM Computing Surveys. – 2009. – Т. 41, № 3. – С. 1–58. – DOI: 10.1145/1541880.1541882. – [Електронний ресурс]. – Режим доступу: https://dl.acm.org/doi/10.1145/1541880.1541882

7. Hodge V., Austin J. A Survey of Outlier Detection Methodologies //

Artificial Intelligence Review. – 2004. – Т. 22, № 2. – С. 85–126. – DOI: 10.1023/B:AIRE.0000045502.10941.a9. – [Електронний ресурс]. – Режим доступу: http://link.springer.com/10.1023/B:AIRE.0000045502.10941.a9

8. Van Aelst S., Rousseeuw P. Minimum volume ellipsoid // WIREs Computational Statistics. – 2009. – Т. 1, № 1. – С. 71–82. – DOI: 10.1002/wics.19. – [Електронний ресурс]. – Режим доступу: https://wires.onlinelibrary.wiley.com/doi/10.1002/wics.19

9. Natha S. A Systematic Review of Anomaly Detection using Machine and Deep Learning Techniques // Quaid-e-Awam University Research Journal of Engineering, Science & Technology. – 2022. – Т. 20, № 1. – С. 83–94. – DOI: 10.52584/QRJ.2001.11. – [Електронний ресурс]. – Режим доступу: http://publications.quest.edu.pk/ojs/index.php/qrj/article/view/66

10. LeCun Y., Bengio Y., Hinton G. Deep learning // Nature. – 2015. – Т. 521, № 7553. – С. 436–444. – DOI: 10.1038/nature14539. – [Електронний ресурс]. – Режим доступу: https://www.nature.com/articles/nature14539

11. Pang G., Shen C., Cao L., Hengel A. V. D. Deep Learning for Anomaly Detection: A Review // ACM Computing Surveys. – 2022. – Т. 54, № 2. – С. 1–38. – DOI: 10.1145/3439950. – [Електронний ресурс]. – Режим доступу: https://dl.acm.org/doi/10.1145/3439950

12. Kulanuwat L. та ін. Anomaly Detection Using a Sliding Window Technique and Data Imputation with Machine Learning for Hydrological Time Series // Water. – 2021. – Т. 13, № 13. – Ст. 1862. – DOI: 10.3390/w13131862. – [Електронний ресурс]. – Режим доступу: https://www.mdpi.com/2073-4441/13/13/1862

13. Bourgerie R., Zanouda T. Fault Detection in Telecom Networks using Bi-level Federated Graph Neural Networks. – 2023. – DOI: 10.48550/ARXIV.2311.14469. – [Електронний ресурс]. – Режим доступу: https://arxiv.org/abs/2311.14469

14. Zhao H. та ін. Multivariate Time-Series Anomaly Detection via Graph Attention Network // 2020 IEEE International Conference on Data Mining (ICDM). –

Sorrento : IEEE, 2020. – С. 841–850. – DOI: 10.1109/ICDM50108.2020.00093. – [Електронний ресурс]. – Режим доступу: https://ieeexplore.ieee.org/document/9338317

15. Hundman K. та ін. Detecting Spacecraft Anomalies Using LSTMs and Nonparametric Dynamic Thresholding. – 2018. – DOI: 10.48550/ARXIV.1802.04431. – [Електронний ресурс]. – Режим доступу: https://arxiv.org/abs/1802.04431

16. Nguyen H. та ін. Forecasting and Anomaly Detection approaches using LSTM and LSTM Autoencoder techniques with applications in supply chain management // International Journal of Information Management. – 2021. – Т. 57. – Ст. 102282. – DOI: 10.1016/j.ijinfomgt.2020.102282. – [Електронний ресурс]. – Режим доступу: https://linkinghub.elsevier.com/retrieve/pii/S026840122031481X

17. Schreyer M. та ін. Detection of Anomalies in Large Scale Accounting Data using Deep Autoencoder Networks. – 2017. – DOI: 10.48550/ARXIV.1709.05254. – [Електронний ресурс]. – Режим доступу: https://arxiv.org/abs/1709.05254

18. Kolosnjaji B. та ін. Adversarial Malware Binaries: Evading Deep Learning for Malware Detection in Executables. – 2018. – DOI: 10.48550/ARXIV.1803.04173. – [Електронний ресурс]. – Режим доступу: https://arxiv.org/abs/1803.04173

19. Suciu O., Coull S. E., Johns J. Exploring Adversarial Examples in Malware Detection. – 2018. – DOI: 10.48550/ARXIV.1810.08280. – [Електронний ресурс]. – Режим доступу: https://arxiv.org/abs/1810.08280

20. Srisakaokul S. та ін. MULDEF: Multi-model-based Defense Against Adversarial Examples for Neural Networks. – 2018. – DOI: 10.48550/ARXIV.1809.00065. – [Електронний ресурс]. – Режим доступу: https://arxiv.org/abs/1809.00065

21. Mohammadi S., Namadchian A. A New Deep Learning Approach for Anomaly-Based IDS using Memetic Classifier // International Journal of Computers Communications & Control. – 2017. – Т. 12, № 5. – С. 677. – DOI: 10.15837/ijccc.2017.5.2972. – [Електронний ресурс]. – Режим доступу:

http://univagora.ro/jour/index.php/ijccc/article/view/2972

22. Yousefi-Azar M. та ін. Autoencoder-based feature learning for cyber security applications // 2017 International Joint Conference on Neural Networks (IJCNN). – Anchorage : IEEE, 2017. – С. 3854–3861. – DOI: 10.1109/IJCNN.2017.7966342. – [Електронний ресурс]. – Режим доступу: http://ieeexplore.ieee.org/document/7966342

23. Latif S. та ін. Phonocardiographic Sensing using Deep Learning for Abnormal Heartbeat Detection. – 2018. – DOI: 10.48550/ARXIV.1801.08322. – [Електронний ресурс]. – Режим доступу: https://arxiv.org/abs/1801.08322

24. Su Y. та ін. Robust Anomaly Detection for Multivariate Time Series through Stochastic Recurrent Neural Network // KDD 2019. – Anchorage : ACM, 2019. – С. 2828–2837. – DOI: 10.1145/3292500.3330672. – [Електронний ресурс]. – Режим доступу: https://dl.acm.org/doi/10.1145/3292500.3330672

25. Malhotra P. та ін. LSTM-based Encoder–Decoder for Multi-sensor Anomaly Detection. – 2016. – [Електронний ресурс]. – Режим доступу: https://arxiv.org/abs/1607.00148

26. Kieu T. та ін. Outlier Detection for Time Series with Recurrent Autoencoder Ensembles // IJCAI 2019. – Macao : IJCAI, 2019. – С. 2725–2732. – DOI: 10.24963/ijcai.2019/378. – [Електронний ресурс]. – Режим доступу: https://www.ijcai.org/proceedings/2019/378

27. Ding C., Sun S., Zhao J. MST-GAT: A multimodal spatial–temporal graph attention network for time series anomaly detection // Information Fusion. – 2023. – Т. 89. – С. 527–536. – DOI: 10.1016/j.inffus.2022.08.011. – [Електронний ресурс]. – Режим доступу: https://linkinghub.elsevier.com/retrieve/pii/S156625352200104X

28. Ren H. та ін. Time-Series Anomaly Detection Service at Microsoft. – 2019. – [Електронний ресурс]. – Режим доступу: https://arxiv.org/abs/1906.03821 arXiv+1

29. Lai C-Y. та ін. Nominality Score Conditioned Time Series Anomaly Detection by Point/Sequential Reconstruction. – 2023. – [Електронний ресурс]. –

Режим доступу: https://arxiv.org/abs/2310.15416

30. Zhan J. та ін. Stgat-Mad: Spatial-Temporal Graph Attention Network For Multivariate Time Series Anomaly Detection // ICASSP 2022. – Singapore : IEEE, 2022. – С. 3568–3572. – DOI: 10.1109/ICASSP43922.2022.9747274. – [Електронний ресурс]. – Режим доступу: https://ieeexplore.ieee.org/document/9747274

31. Li W. та ін. StackVAE-G: An efficient and interpretable model for time series anomaly detection. – 2021. – [Електронний ресурс]. – Режим доступу: https://arxiv.org/abs/2105.08397 arXiv+1

32. Vaswani A. та ін. Attention Is All You Need. – 2017. – [Електронний ресурс]. – Режим доступу: https://arxiv.org/abs/1706.03762

33. Hoi S. C., Sahoo D., Lu J., Zhao P. Online learning: A comprehensive survey // Neurocomputing. – 2021. – Т. 459. – С. 249–289. – DOI: 10.1016/j.neucom.2021.04.112. – [Електронний ресурс]. – Режим доступу: https://linkinghub.elsevier.com/retrieve/pii/S0925231221006706

34. Sustainable Development Goals. – [Електронний ресурс]. – Режим доступу: https://sdgs.un.org/goals

ДОДАТОК А

Архітектура моделей

Таблиця А.1: Деталі параметрів моделі *CAE_BiFo*

Модель реконструкції

Шар (тип)	**Форма**	**Кількість параметрів**
Кодер		
cnn.0.weight	torch.Size([8, 1, 2, 2])	32
cnn.3.weight	torch.Size([4, 8, 3, 3])	288
Декодер		
cnn.0.weight	torch.Size([4, 8, 3, 3])	28
cnn.2.weight	torch.Size([8, 8, 2, 2])	256
cnn.4.weight	torch.Size([8, 1, 2, 2])	32
Модель прогнозування		
lstm1.weight_ih_l0	torch.Size([400, 68])	27
lstm1.weight_hh_l0	torch.Size([400, 100])	40
lstm1.weight_ih_l0_reverse	torch.Size([400, 68])	27 200
lstm1.weight_hh_l0_reverse	torch.Size([400, 100])	40 000
lstm2.weight_ih_l0	torch.Size([400, 200])	80
lstm2.weight_hh_l0	torch.Size([400, 100])	40
lstm2.weight_ih_l0_reverse	torch.Size([400, 200])	80
lstm2.weight_hh_l0_reverse	torch.Size([400, 100])	40
out.weight	torch.Size([68, 200])	13 600
Загальна кількість параметрів		**392 193**

Таблиця А.2: Деталі параметрів моделі *CLAE_BiFo*

Шар/Кодер	**Форма**	**Кількість параметрів**
cnn.weight	torch.Size([68, 68, 5])	23
Модель прогнозування		
lstm1.weight_ih_l0	torch.Size([256, 68])	17 408
lstm1.weight_hh_l0	torch.Size([256, 64])	16 384
lstm1.weight_ih_l0_reverse	torch.Size([256, 68])	17 408
lstm1.weight_hh_l0_reverse	torch.Size([256, 64])	16 384
lstm2.weight_ih_l0	torch.Size([128, 128])	16 384
lstm2.weight_hh_l0	torch.Size([128, 32])	4 096
lstm2.weight_ih_l0_reverse	torch.Size([128, 128])	16 384
lstm2.weight_hh_l0_reverse	torch.Size([128, 32])	4 096
lstm3.weight_ih_l0	torch.Size([16, 64])	1 024
lstm3.weight_hh_l0	torch.Size([16, 4])	64
lstm3.weight_ih_l0_reverse	torch.Size([16, 64])	1 024
lstm3.weight_hh_l0_reverse	torch.Size([16, 4])	64
lstm1.weight_ih_l0	torch.Size([128, 8])	1 024
lstm1.weight_hh_l0	torch.Size([128, 32])	4 096
lstm1.weight_ih_l0_reverse	torch.Size([128, 8])	1 024
lstm1.weight_hh_l0_reverse	torch.Size([128, 32])	4 096
lstm2.weight_ih_l0	torch.Size([256, 64])	16 384
lstm2.weight_hh_l0	torch.Size([256, 64])	16 384
lstm2.weight_ih_l0_reverse	torch.Size([256, 64])	16 384
lstm2.weight_hh_l0_reverse	torch.Size([256, 64])	16 384
out.weight	torch.Size([68, 128])	8 704
lstm1.weight_ih_l0	torch.Size([400, 68])	27 200
lstm1.weight_hh_l0	torch.Size([400, 100])	40
Декодер		
lstm1.weight_ih_l0_reverse	torch.Size([400, 68])	27 200
lstm1.weight_hh_l0_reverse	torch.Size([400, 100])	40
lstm2.weight_ih_l0	torch.Size([400, 200])	80
lstm2.weight_hh_l0	torch.Size([400, 100])	40
lstm2.weight_ih_l0_reverse	torch.Size([400, 200])	80
lstm2.weight_hh_l0_reverse	torch.Size([400, 100])	40
out.weight	torch.Size([68, 200])	13 600
Загальна кількість параметрів		**612 860**

ДОДАТОК Б

```python
class Trainer:
    """
    The Trainer class is designed to train jointly-optimized
    forecasting and reconstruction-based models.

    :param epoch: Number of training epochs
    :param lr: Learning rate value
    :param train_loader: DataLoader for training data
    :param val_loader: DataLoader for validation data
    :param model: The model to be trained
    :param save_model: Boolean to save the model's weights after training
    :param name: Name of the model file if it's being saved
    :param patience: Patience for early stopping
    """
    def __init__(self, epoch, lr, train_loader, val_loader, model,
                 save_model=False, name='autoencoder-checkpoint.pth',
                 patience=2):
        device = torch.device("cuda" if torch.cuda.is_available() else "cpu")
        self.model = model.to(device)
        self.epoch = epoch
        self.lr = lr
        self.train_loader = train_loader
        self.val_loader = val_loader
        self.rec_criterion = nn.MSELoss()
        self.for_criterion = nn.MSELoss()
        self.save_model = save_model
        self.name = name
        self.patience = patience

    def train(self):
        print(self.model)
        optimizer = optim.Adam(self.model.parameters(),
                               lr=self.lr, weight_decay=1e-5)
        scheduler = ReduceLROnPlateau(optimizer, mode="min", factor=0.1,
                                      patience=3, verbose=True)

        best_model_w = copy.deepcopy(self.model.state_dict())
```

```
        best_model_w = copy.deepcopy(self.model.state_dict())
        history = {
            'total_train': [], 'total_val': [],
            'train_rec': [], 'train_for': [],
            'val_rec': [], 'val_for': []
        }
        wait = 0
        best_loss = float('inf')
        for e in tqdm(range(self.epoch)):
            # Training
            self.model.train()
            total_loss = 0
            total_rec_loss = 0
            total_for_loss = 0

            for x, y in self.train_loader:

                # ======Forward pass======
                rec, pred = self.model(x)

                rec_loss = self.rec_criterion(rec, x)
                for_loss = self.for_criterion(pred, y)

                loss = for_loss + rec_loss
                loss = torch.sqrt(loss)

                # ======Backward pass======
                optimizer.zero_grad()
                loss.backward()
                optimizer.step()

                total_loss += loss.item()
                total_rec_loss += rec_loss.item()
                total_for_loss += for_loss.item()

            # ======Validation======
            rec_running_loss = 0
```

```
72              # ======Validation======
73              rec_running_loss = 0
74              for_running_loss = 0
75              self.model.eval()
76              with torch.no_grad():
77                  for x, y in self.val_loader:
78
79                      rec, pred = self.model(x)
80                      rec_vloss = self.rec_criterion(rec, x)
81                      for_vloss = self.for_criterion(pred, y)
82
83                      rec_vloss = torch.sqrt(rec_vloss)
84                      for_vloss = torch.sqrt(for_vloss)
85
86                      rec_running_loss += rec_vloss.item()
87                      for_running_loss += for_vloss.item()
88
89              num_batches = len(self.val_loader)
90              rec_avg_val_loss = rec_running_loss / num_batches
91              for_avg_val_loss = for_running_loss / num_batches
92              avg_val_loss = rec_avg_val_loss + for_avg_val_loss
93              scheduler.step(avg_val_loss)
94
95              if avg_val_loss < best_loss:
96                  best_loss = avg_val_loss
97                  best_model_w = copy.deepcopy(self.model.state_dict())
98                  wait = 0
99              else:
100                 wait += 1
101                 if wait >= self.patience:
102                     print('Early stopping!')
103                     break
104
105             history['total_train'].append(total_loss / len(self.train_loader))
106             history['train_rec'].append(total_rec_loss / len(self.train_loader))
107             history['train_for'].append(total_for_loss / len(self.train_loader))
108
```

```
109             history['total_val'].append(avg_val_loss)
110             history['val_rec'].append(rec_avg_val_loss)
111             history['val_for'].append(for_avg_val_loss)
112 
113             if e % 10 == 0 and e != 0:
114                 clear_output(wait=True)
115             print(f'Epoch [{e + 1}/{self.epoch}], '
116                   f'Training Loss: [rec {total_rec_loss / len(self.train_loader):.6
117                   f'for {total_for_loss / len(self.train_loader):.6f}, '
118                   f'total {total_loss / len(self.train_loader):.6f}], '
119                   f'Validation Loss: [rec {rec_avg_val_loss:.6f}, '
120                   f'for {for_avg_val_loss:.6f}, '
121                   f'total {avg_val_loss:.6f}]')
122         self.model.load_state_dict(best_model_w)
123         if self.save_model:
124             torch.save(self.model.state_dict(), self.name)
125         return self.model, history
```

ДОДАТОК В

Цей розділ надає псевдокод для алгоритму пошуку по сітці, який використовується для знаходження найкращого набору гіперпараметрів. Алгоритм ітерує через всі комбінації гіперпараметрів і навчає модель. Після цього він оцінює модель, використовуючи обидва порогові методи (динамічний та z-оцінка) і вибирає найкращий показник F1. Обраний показник F1 та відповідні гіперпараметри зберігаються в масиві у вигляді кортежів. Алгоритми повернуть цей масив після завершення останньої ітерації.

Рядки 5-8 псевдокоду відповідають процесу навчання моделі, тоді як рядки 10-20 зосереджені на процесі оцінювання. Функція get_score використовує рівняння 3.6, представлене в розділі 3.2.3, а adjust_predicts представляє метод коригування прогнозів, обговорений у розділі 3.3.

Змінна search_space є словником Python, що містить різні значення для гіперпараметрів. Її можна представити наступним чином:

Алгоритм 1: Алгоритм пошуку по сітці

Вхідні дані: train_data, val_data, test_data, true_labels, search_space, n_features, sliding_window

1 Initialize: log

2 learning_rates, epochs, gamma, hidden_layers → search_space

3 **foreach** *(lr, e, h)* ∈ *product(learning_rates, epochs, hidden_layers)* **do**

4 Set random seeds to a fixed value

5 model → CAE_BiFo(n_features, h)

6 trainer → Trainer(lr, e, model, train_data, val_data)

7 model → trainer.train() // Start the training loop

8

9 best_f1 → 0

10 **foreach** $g \in \gamma$ **do**

11 test_avg → mean(get_score(model, test_data, g))

```
            thresholds → Calculate thresholds (Dynamic and z-score)
            foreach t ∈ thresholds do
                f1 → evaluate(adjust_predicts(test_avg, true_labels, t))
                if f1 > best_f1 then
                    best_f1 → f1
                    temp_log → (f1, lr, e, h, g)
                end
            end
        end
        log.append(temp_log)
    end
    return log
```

ДОДАТОК Г

Набір даних про серверну машину

SMD складається з даних, зібраних протягом 5 тижнів з 28 серверних машин, кожна з яких оснащена 38 датчиками, в інтернет-компанії. Цей набір даних включає такі показники TS, як завантаження процесора, використання пам'яті, активність запису на диск тощо [24]. Характеристики цього набору даних наведені в таблиці D.1.

Таблиця D.1: Опис SMD.

Набір даних	Час збору даних	Кількість ознак	Довжина часових рядів	Відсоток аномалій
Тренування	5 тижнів	38	708 405	0
Тест	5 тижнів	38	708 420	4,16