МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ДЕРЖАВНИЙ УНІВЕРСИТЕТ
“КИЇВСЬКИЙ АВІАЦІЙНИЙ ІНСТИТУТ”
ФАКУЛЬТЕТ АЕРОНАВІГАЦІЇ, ЕЛЕКТРОНІКИ ТА ТЕЛЕКОМУНІКАЦІЙ
КАФЕДРА ТЕЛЕКОМУНІКАЦІЙНИХ ТА РАДІОЕЛЕКТРОННИХ СИСТЕМ

ДОПУСТИТИ ДО ЗАХИСТУ
Завідувач кафедри
Віктор ГНАТЮК
«_____»_____ 2025 р

КВАЛІФІКАЦІЙНА РОБОТА
(ПОЯСНЮВАЛЬНА ЗАПИСКА)

ВИПУСКНИКА ОСВІТНЬОГО СТУПЕНЯ МАГІСТР

Тема: «Система інтелектуального аналізу трафіку в мережах мобільного зв’язку»

Виконавець: Данило ЛОЄНКО
(підпис)

Керівник: Віктор ГНАТЮК
(підпис)

Консультанти з окремих розділів пояснювальної записки:

Консультант розділу «Охорона праці» Катерина КАЖАН
(підпис)

Консультант розділу «Охорона навколишнього середовища»
Лариса ЧЕРНЯК
(підпис)

Нормоконтролер: Богдан ЧУМАЧЕНКО
(підпис)

Київ 2025

ДЕРЖАВНИЙ УНІВЕРСИТЕТ
“КИЇВСЬКИЙ АВІАЦІЙНИЙ ІНСТИТУТ”

Факультет аеронавігації, електроніки та телекомунікацій .
Кафедра телекомунікаційних та радіоелектронних систем .
Спеціальність 172 «Електронні комунікації та радіотехніка» .
Освітньо-професійна програма «Телекомунікаційні системи та мережі» .

ЗАТВЕРДЖУЮ
Завідувач кафедри

Віктор ГНАТЮК
«_____» _____ 2025 р.

ЗАВДАННЯ

на виконання кваліфікаційної роботи

Лоєнка Данила Романовича
(прізвище, ім’я, по батькові випускника в родовому відмінку)

1. Тема кваліфікаційної роботи: «Система інтелектуального аналізу трафіку в мережах мобільного зв’язку»

   Затверджена наказом ректора від «02» вересня 2025 р. № 1672 /ст.
2. Термін виконання роботи: з 29.09.2025 р. по 31.12.2025 р.
3. Вихідні дані для роботи: проаналізувати виклики мереж 5G/6G та неефективність DPI в умовах шифрування. Обґрунтувати необхідність виявлення аномалій у мобільному трафіку. Як вхідні дані для розробки прототипу використати стандартизований набір даних KDD Cup.
4. Зміст пояснювальної записки: Аналіз архітектури 5G та типів трафіку. Дослідження методів ML, зокрема Автоенкодера, для виявлення загроз. Розробка модульної дворівневої системи та експериментальна оцінка її ефективності на тестових двних.
5. Перелік обов’язкового графічного (ілюстративного) матеріалу: презентація, слайди.

6. Календарний план-графік

№ пор.	Завдання	Термін виконання	Відмітка про виконання
1.	Розробити деталізований зміст розділів кваліфікаційної роботи	10.09.2025-15.09.2025	Виконано
2.	Вступ	20.09.2025	Виконано
3.	Теоретичне обґрунтування: Дослідження математичних моделей та методів ML для аналізу трафіку.	22.09.2025-25.09.2025	Виконано
4.	Проектування системи: Розробка дворівневої модульної архітектури та вимог до реалізації (Python/Keras).	25.09.2025-27.09.2025	Виконано
5.	Розробка прототипу: Реалізація ключового модуля виявлення аномалій на базі Автоенкодера.	02.10.2025-10.10.2025	Виконано
6.	Експериментальне дослідження: Проведення тестування на даних KDD Cup та валідація ефективності	28.10.2025-01.11.2025	Виконано
7.	Фіналізація: Підготовка електронної версії звіту, ілюстративних матеріалів (графіки, скріншоти коду) та подання.	01.11.2025-11.11.2025	Виконано
8.	Охорона праці	12.12.25-13.12.25	Виконано
9.	Охорона навколишнього середовища	12.12.25-13.12.25	Виконано
10.	Усунення недоліків та захист кваліфікаційної роботи	13.12.25-15.12.25	Виконано

7. Консультанти з окремих розділів

Розділ	Консультант (посада, П.І.Б.)	Дата, підпис	
		Завдання видав	Завдання прийняв
Охорона праці	к.т.н, доц. Катерина КАЖАН	05.12.2025	*Виконано*
Охорона навколишнього середовища	д.т.н, доц. Лариса ЧЕРНЯК	10.12.2025	*Виконано*

8. Дата видачі завдання: «01» вересня 2025 р.

Керівник кваліфікаційної роботи _____ Віктор ГНАТЮК
(підпис керівника) (П.І.Б.)

Завдання прийняв до виконання _____ Данило ЛОЄНКО
(підпис випускника) (П.І.Б.)

РЕФЕРАТ

Кваліфікаційна робота «Система інтелектуального аналізу трафіку в мережах мобільного зв’язку» містить 95 сторінки, 27 рисунків, 2 таблиці 24 використаних джерела.

Метою цієї кваліфікаційної роботи є аналітичне дослідження, розробка та експериментальна валідація системи інтелектуального аналізу трафіку для ефективного виявлення аномалій у мережах 5G/IoT

Об’єктом дослідження процеси аналізу, обробки та контролю трафіку в сучасних мобільних мережах (4G, 5G)

Предметом дослідження є архітектурні рішення, алгоритми машинного навчання та статистичні ознаки для побудови системи виявлення аномалій на базі Автоенкодера

Метод дослідження – програмний прототип, реалізований на Python/TenserFlow.

Розроблений прототип реалізує дворівневу логіку, здатну ідентифікувати аномалії (DDoS, атаки "нульового дня") у зашифрованому трафіку, базуючись на аналізі помилки відновлення Автоенкодера.

Рекомендується використовувати матеріал дисертації для інтеграції в системи моніторингу трафіку телекомунікаційних операторів, а також у навчальному процесі при викладанні дисциплін "Кібербезпека" та "Мережі 5G".

ЗМІСТ

СПИСОК СКОРОЧЕНЬ, УМОВНИХ ПОЗНАЧЕНЬ

1) EPC – Evolved Packet Core
2) MME – Mobility Management Entity
3) IoT – Internet Of Things
4) DPI – Deep Packet Inspection
5) LTE – Long Term Evolution
6) PGW – Packet Data Network Gateway
7) SGW – Serving Gateway
8) ML – Machine Learning
9) QoS – Quality of Service
10) eMBB – Enhanced Mobile Broadband
11) mMTC – Massive Machine-type Communication
12) HLS – HTTP Live Streaming
13) DASH – Dynamic Adaptive Streaming over HTTP
14) QUIC – Quick UDP Internet Connections
15) DDoS – Disturbed Denial of Service
16) RF – Random Forrest
17) SVM – Support Vector Machines
18) ML – Machine Learning
19) AE – Autoencoder
20) AI – Artificial Intelligence

ВСТУП

Стрімкий розвиток мереж мобільного зв’язку є висококонкурентною сферою, що змінюється щодня, тому вкрай важливо досліджувати еволюцію методів аналізу трафіку для забезпечення якості обслуговування (QoS) та безпеки в мережах 4G/5G. Суть роботи полягає у дослідженні новітніх інтелектуальних технологій у цій галузі, порівняльному аналізі традиційних та сучасних AI/ML-підходів та визначенні ролі, яку системи на базі машинного навчання відіграють у покращенні аналізу та класифікації мобільного трафіку.

Актуальність теми - це вибухове зростання трафіку: Стрімке розгортання мереж 5G, розвиток Інтернету речей (IoT) та поширення "важких" сервісів (4K-стрімінг, AR/VR) призводять до експоненційного зростання обсягів мобільного трафіку. Ускладнення загроз та неефективність DPI: Переважна більшість (понад 90%) мобільного трафіку зараз шифрується (TLS 1.3, QUIC). Це робить традиційні методи аналізу, як-от Deep Packet Inspection (DPI), "сліпими" до кіберзагроз (ботнети, шпигунське ПЗ), прихованих усередині шифрованих потоків. Потреба в інтелектуальних рішеннях: В умовах шифрування єдиним дієвим підходом є інтелектуальний аналіз (AI/ML), що дозволяє класифікувати трафік та виявляти аномалії без дешифрування, базуючись на статистичних та поведінкових характеристиках потоків.

Тому розробка системи, здатної проводити такий аналіз, є вкрай актуальною задачею для забезпечення якості обслуговування (QoS) та кібербезпеки мобільних мереж.

Мета роботи - підвищення ефективності аналізу трафіку в мобільних мережах шляхом розробки та дослідження системи інтелектуального аналізу, здатної класифікувати трафік та виявляти аномалії в умовах шифрування.

Об’єкт дослідження - процеси аналізу та обробки трафіку в сучасних мережах мобільного зв’язку (4G, 5G).

Предмет дослідження це - методи, алгоритми та архітектурні рішення для побудови системи інтелектуального аналізу мобільного трафіку на основі технологій машинного навчання.

Методи дослідження: для вирішення поставлених завдань у роботі використано комплекс наукових методів, методи системного аналізу та теорії телекомунікацій (для аналізу архітектури мобільних мереж та особливостей трафіку), теорія ймовірностей та математична статистика (для аналізу характеристик та побудови моделей трафіку), методи машинного навчання (зокрема, ансамблеві методи, нейронні мережі для класифікації трафіку та виявлення аномалій), методи імітаційного моделювання (для проведення експериментальних досліджень системи), методи об'єктно-орієнтованого програмування (при реалізації програмного прототипу).

Практичне значення отриманих результатів полягає в тому, що розроблений прототип може слугувати основою для впровадження комерційних систем моніторингу мереж.

Апробація: брав участь у науковій конференції: Наукова і практична конференція "Проблеми функціонування та захисту інформаційно-комунікаційних систем," м. Київ, 2025.

РОЗДІЛ 1
АНАЛІЗ СУЧАСНОГО АНАЛІЗУ ТРАФІКУ В МОБІЛЬНИХ МЕРЕЖАХ

1.1. Характеристика мобільних мереж та їх розвиток (4G, 5G перспективи 6G)

Сучасні системи інтелектуального аналізу трафіку функціонують у складному та гетерогенному середовищі мобільних мереж, що стрімко еволюціонують. Для розуміння специфіки трафіку та викликів, пов'язаних з його аналізом, необхідно розглянути архітектурні особливості та ключові технології мереж четвертого, п'ятого та майбутнього шостого покоління.

Мережі четвертого покоління (4G), стандартизовані як LTE, стали революційним кроком, що забезпечило перехід до повністю IP-орієнтованої архітектури.

В основі 4G лежить архітектура Evolved Packet Core (EPC). Вона складається з ключових компонентів, таких як Mobility Management Entity (MME) для управління сигналізацією та мобільністю, Serving Gateway (SGW) та Packet Data Network Gateway (PGW) для маршрутизації та обробки пакетів даних користувача. Архітектура EPC, хоч і ефективна, проте, є відносно монолітною.

Головною метою 4G було надання високошвидкісного мобільного широкосмугового доступу (Mobile Broadband - MBB). Саме це стимулювало розвиток відеострімінгу, соціальних мереж та хмарних сервісів на мобільних пристроях, що призвело до значного зростання обсягів трафіку останнім часом.

Мережі п'ятого покоління (5G) – це не просто еволюція швидкості, а фундаментальна перебудова архітектури для підтримки кардинально нових та сучасних бізнес-моделей та сценаріїв використання.

5G впроваджує Сервіс-Орієнтовану Архітектуру (Service-Based Architecture - SBA) в ядрі мережі (5GC). На відміну від EPC, де функції були тісно пов'язані з конкретними вузлами, SBA використовує принципи віртуалізації (NFV) та

програмно-конфігурованих мереж (SDN). Мережеві функції (NF) реалізовані у вигляді програмних мікросервісів, що взаємодіють між собою через стандартизовані API (наприклад, RESTful API). Це забезпечує безпрецедентну гнучкість, масштабованість та швидкість впровадження нових послуг у будь-який момент. Ключові сценарії використання 5G спроєктовано для одночасної підтримки трьох основних класів послуг, які генерують абсолютно різні типи трафіку:

eMBB (Enhanced Mobile Broadband): Покращений мобільний широкосмуговий доступ. Це прямий розвиток 4G, що забезпечує пікові швидкості до 10-20 Гбіт/с. Тип трафіку: "важкий" контент, як-от 4K/8K стрімінг, Доповнена (AR) та Віртуальна (VR) реальність. URLLC (Ultra-Reliable Low-Latency Communication): Наднадійний зв'язок з низькою затримкою (до 1 мс). Тип трафіку: короткі, критично важливі пакети даних, де надійність важливіша за швидкість (наприклад, телемедицина, дистанційне керування роботами, автономний транспорт). mMTC (Massive Machine-Type Communication): Масові міжмашинні комунікації. Забезпечує підтримку до 1 мільйона пристроїв на квадратний кілометр. Тип трафіку: велика кількість пристроїв (IoT-сенсори, "розумні" лічильники), кожен з яких генерує малий обсяг нечутливих до затримок даних.

Технологія "Network Slicing": є ключовою інновацією 5G, що стало можливо завдяки SBA. "Нарізка мережі" дозволяє операторам створювати декілька віртуальних, ізольованих логічних мереж ("слайсів") поверх єдиної фізичної інфраструктури. Кожен "слайс" може бути оптимізований під конкретний сценарій (наприклад, окремий слайс для eMBB, окремий для URLLC). Це створює нові виклики для аналізу трафіку, оскільки необхідно окремо моніторити та забезпечувати QoS для кожного "слайсу".

Хоча стандартизація 6G очікується ближче до 2030 року, що можна побачити нижче на (Рисунок 1.1), ключові концепції вже активно досліджуються. 6G розглядається як платформа, що об'єднає фізичний, цифровий та біологічний світи. Штучний інтелект буде не просто "надбудовою" (як у 5G), а невід'ємною, вбудованою частиною архітектури 6G.

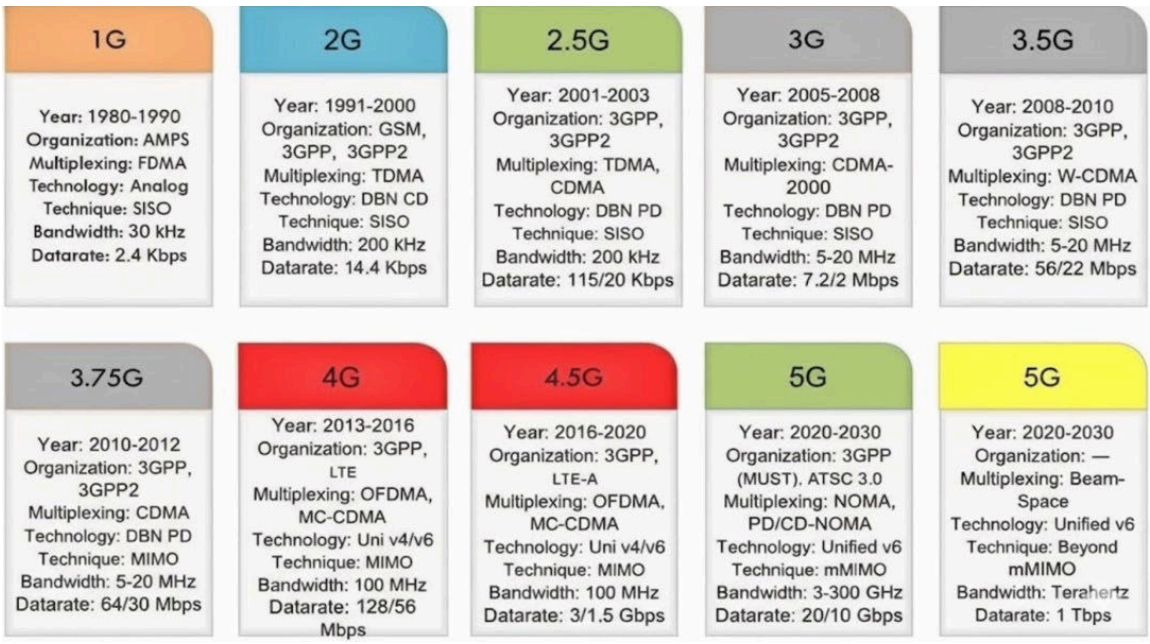

Рис. 1.1. Наглядна різниця поколінь

Мережа зможе автономно оптимізувати свої ресурси, прогнозувати збої та виявляти загрози в реальному часі. Цікаво, що 6G планує використовувати спектр у діапазоні 100 ГГц – 10 ТГц, що потенційно дозволить досягти швидкостей до 1 Тбіт/с. Очікується поява таких сервісів, як голографічний зв'язок, "Інтернет Почуттів" (Tactile Internet), та створення глобальних цифрових двійників і все це очікує на нас в найближчому майбутньому.

1.2. Типи трафіку в мобільних мережах і їх особливості

Основою для аналізу є розуміння поточного стану та складу трафіку в мобільних мережах. Обсяги даних зростають експоненційно. Згідно з останнім звітом Ericsson Mobility Report, загальний місячний глобальний трафік мобільних мереж у другому кварталі 2025 року досяг 180 Ексабайт (EB), показавши зростання на 19% порівняно з минулим роком.

Ключовою особливістю сучасного мобільного трафіку є його абсолютна гетерогенність, однак у ньому є явний домінант. На кінець 2024 року відео трафік складав 74% усього мобільного трафіку, як показано нижче на (Рисунок 1.2).

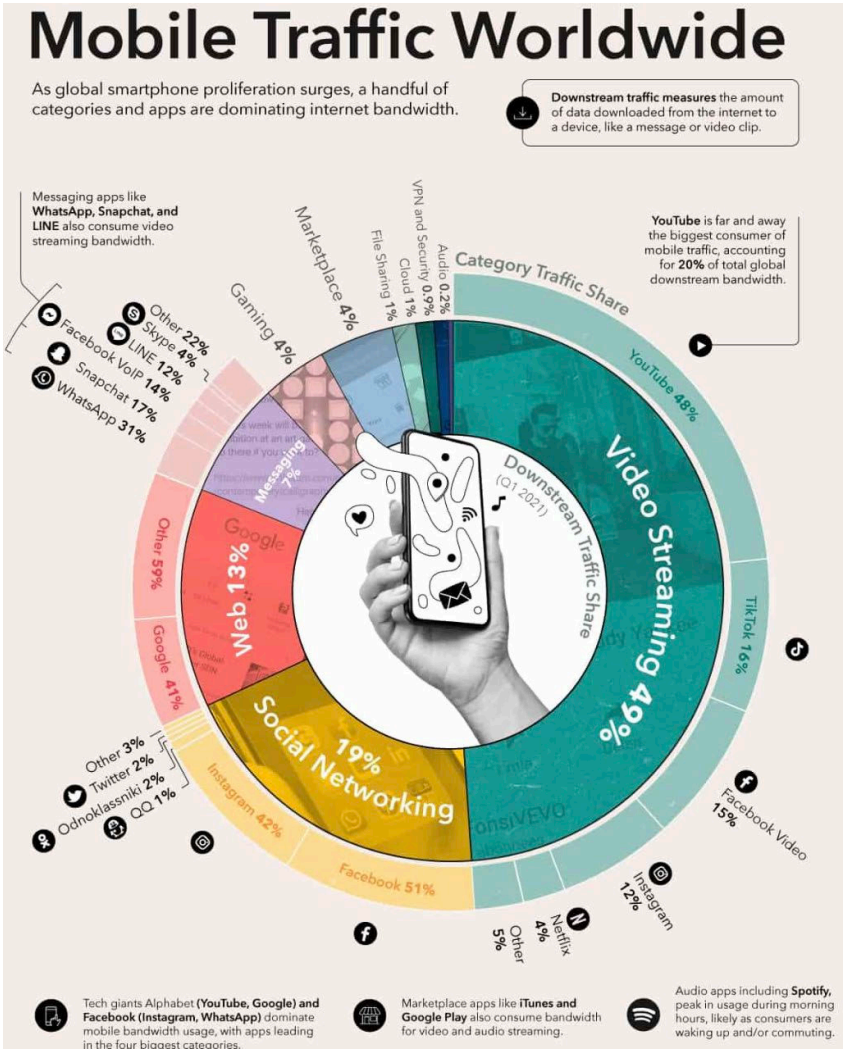

Рис. 1.2. Візуалізація використання мобільного трафіку

Це зростання стимулюється як збільшенням кількості підписок на смартфонах, так і, головним чином, збільшенням перегляду відеоконтенту. Цей факт створює перший виклик: система аналізу має бути оптимізована для обробки переважно "важких" мультимедійних потоків [1].

Шифрування утвердилося як фундаментальний компонент сучасної цифрової бізнес-інфраструктури, що зумовлено зростаючою потребою у захисті даних в умовах, коли клієнти, співробітники та програмні застосунки все частіше взаємодіють з корпоративними ресурсами через загальнодоступні мережі. Статистичні дані, зокрема від FortiGuard Labs, ілюструють цю тенденцію: частка зашифрованого веб-трафіку зросла з приблизно 55% у 2017 році до 85% на поточний момент.

Ця технологія, безсумнівно, мала позитивний вплив на рівень конфіденційності та кібербезпеки, однак її стрімке поширення має і зворотний бік. Зловмисники дедалі

частіше експлуатують шифрування для уникнення виявлення та реалізації масштабних кібератак, що вимагає від організацій розробки нових підходів до безпеки.

За своєю суттю, шифрування є методом захисту даних, який перетворює читабельну інформацію (відкритий текст) на незрозумілий шифр (шифротекст). Доступ до вихідної інформації після такого перетворення можливий лише для авторизованих суб'єктів, які володіють відповідним ключем або використовують спеціалізовані методи декодування. Це дозволяє організаціям безпечно передавати конфіденційні дані, запобігаючи їх несанкціонованому перехопленню. Проте, такий мережевий трафік є надзвичайно складним для моніторингу та інспекції, оскільки більшість організацій не володіють достатніми можливостями для дешифрування всього обсягу внутрішнього трафіку в реальному часі. Ця "сліпа зона" створює умови, за яких зловмисники можуть приховано розповсюджувати несанкціоноване програмне забезпечення та шкідливі коди.

Кіберзлочинці активно використовують шифрування для обходу систем безпеки, усвідомлюючи, що значна частина трафіку не піддається ретельній перевірці. Вони застосовують технології шифрування, такі як SSL (Secure Sockets Layer), для приховування своєї активності, незалежно від того, чи йдеться про доставку шкідливого ПЗ, чи про викрадення конфіденційних даних. Це дозволяє обходити традиційні інструменти безпеки. Шифрування використовується для маскування початкової атаки, обфускації комунікацій з командно-контрольною інфраструктурою (C&C), а також для захисту викраденої інформації, такої як паролі чи банківські реквізити, під час її виведення з периметра організації. Аналітичні прогнози, зокрема від Gartner, вказували, що понад 70% шкідливих кампаній будуть використовувати певні форми шифрування саме тому, що це значно знижує ефективність захисних механізмів.

Хоча інспекція зашифрованого трафіку є логічним рішенням цієї проблеми, її практична реалізація пов'язана зі значними труднощами. Основна перешкода полягає у часі та обчислювальних ресурсах, необхідних для повного циклу дешифрування, аналізу та подальшого повторного шифрування даних. Вимоги сучасного цифрового

ринку диктують необхідність миттєвого доступу до даних, і пов'язана з інспекцією затримка (латентність) та додаткове навантаження часто є неприйнятними. Лише обмежена кількість інструментів безпеки здатна виконувати такі операції зі швидкістю, що відповідає потребам бізнесу, створюючи для організацій складну дилему між рівнем безпеки та операційною продуктивністю [2].

Окрім фактора шифрування, мобільний трафік володіє й іншими комплексними властивостями, що ускладнюють його аналіз. По-перше, йому притаманний імпульсний характер, або "вибуховість" (burstiness). На відміну від стаціонарного, рівномірного потоку, більшість додатків генерують трафік нерівномірно; наприклад, завантаження веб-сторінки провокує короткочасний сплеск запитів, за яким настає фаза неактивності, коли користувач споживає контент. Подібний патерн демонструє і відеострімінг, де буферизація даних відбувається дискретними порціями. По-друге, трафік характеризується надзвичайною гетерогенністю. Профіль трафіку, що генерується IoT-датчиком (кілька байт на годину), кардинально відрізняється від профілю онлайн-гри, який вимагає передачі малих, але частих пакетів із мінімальною затримкою, або ж від процесу завантаження файлу, орієнтованого на максимальну утилізацію пропускної здатності. Саме ця синергія трьох факторів – непрозорість через шифрування, імпульсний характер та гетерогенність – робить традиційні методи аналізу, що покладаються на інспекцію вмісту, неефективними. Натомість, це виводить на провідні позиції методи машинного навчання, які здатні ідентифікувати типи трафіку та виявляти аномалії, базуючись виключно на статистичних "відбитках" потоків, таких як розподіл розмірів пакетів, часові інтервали між ними та загальна тривалість сесії, без необхідності дешифрування самого вмісту [3].

Трафік, що належить до категорії eMBB (покращений мобільний широкосмуговий доступ), є домінуючим у сучасних мережах, складаючи понад 74% від загального обсягу, та включає в себе передачу "важкого" контенту, зокрема OTT-відео (наприклад, YouTube, Netflix), доповнену та віртуальну реальність (AR/VR). Механіка потоку для цього класу трафіку, особливо для відеострімінгу, характеризується не суцільною передачею даних, а використанням технологій адаптивного бітрейту (ABR), таких як DASH та HLS. Принцип їх роботи полягає у

сегментації відеофайлу на сервері на короткі "фрагменти" (зазвичай тривалістю 2-10 секунд), які мобільний пристрій послідовно завантажує у буфер для відтворення.

Цей механізм породжує чітко виражену "вибуховість" (burstiness) трафіку: плеєр завантажує сегмент на максимальній доступній швидкості, що створює короткий інтенсивний сплеск активності, після чого настає "пауза" з майже нульовим трафіком, доки відбувається відтворення з буфера. Цей цикл "сплеск-пауза" повторюється. Водночас, весь цей трафік передається через зашифровані протоколи, такі як HTTPS (на базі TCP) або QUIC (на базі UDP), що унеможливлює традиційний аналіз вмісту (DPI).

З точки зору вимог до якості обслуговування (QoS), eMBB-трафік потребує високої пропускної здатності для швидкого заповнення буфера та підтримки високої якості зображення. Він чутливий до "джиттера" (коливань затримки), оскільки стабільність швидкості є критичною для запобігання спустошенню буфера (ре-буферизації), що є важливішим за абсолютне значення затримки. Завдяки наявності буфера, який може вміщувати 10-30 секунд відео, цей тип трафіку не є чутливим до помірної затримки (наприклад, 100-200 мс).

Для інтелектуальних систем аналізу, такий трафік є ідеальним об'єктом для статистичної класифікації. Навіть у зашифрованому вигляді, характерний патерн "burst-pause" функціонує як унікальний "відбиток", який дозволяє моделям машинного навчання з високою точністю ідентифікувати цей потік як відеострімінг, базуючись на метаданих, а не на вмісті пакетів [4].

Наступний клас трафіку, що має вирішальне значення для нових сервісів 5G, висуває вимоги, які є діаметрально протилежними до eMBB. Цей клас (URLLC) охоплює такі сфери, як онлайн-ігри, конференц-зв'язок, промислова автоматизація, телемедицина та автономне водіння. Механіка потоку цих інтерактивних сервісів виключає можливість значної буферизації, оскільки дані мають доставлятися миттєво; наприклад, дія гравця у грі вимагає негайної реакції сервера. Характеристики трафіку включають малі пакети (десятки чи сотні байт), що передаються з високою частотою, часто симетрично між висхідним та низхідним каналами, із пріоритетним використанням протоколу UDP для мінімізації затримки.

Вимоги до якості обслуговування (QoS) є надзвичайно жорсткими: наднизька затримка (близько 1 мс), мінімальний "джиттер" (оскільки стабільність доставки є критичною) та висока надійність, що сягає 99.9999% для критичних додатків. Відповідно, "відбиток" цього трафіку - постійний, симетричний потік малих пакетів - дозволяє легко відрізнити його від імпульсного відео. Його ідентифікація є пріоритетною для мережевого аналізу, щоб забезпечити найвищий пріоритет, навіть за рахунок інших типів трафіку.

Третя унікальна парадигма, mMTC (Інтернет Речей), орієнтована на масову кількість пристроїв, а не на швидкість. До представників належать "розумні" лічильники, різноманітні датчики та логістичні трекери. Архітектура IoT передбачає, що пристрої переважну частину часу (до 99.9%) перебувають у "сплячому" режимі для збереження енергії, періодично "прокидаючись" для надсилання надмалих пакетів даних (кілька байт корисного навантаження). Патерни трафіку є періодичними або керованими подіями, причому домінують легковагові протоколи, як-от MQTT або CoAP, оптимізовані для ненадійних мереж та пристроїв з обмеженими ресурсами. Ключовими вимогами QoS є енергоефективність (для тривалої роботи від батарей), масштабованість для підтримки мільйонів підключень та нечутливість до затримки чи швидкості. "Відбиток" IoT-трафіку (надмалі пакети, довгі "періоди сну") полегшує його відокремлення від людського трафіку. Однак, головний виклик для аналізу полягає у виявленні аномалій: якщо пристрій, що має надсилати 50 байт на добу, раптом починає генерувати "вибуховий" трафік, це є беззаперечною ознакою його компрометації та ймовірного включення до ботнету. Саме для вирішення таких завдань аномальної детекції можуть бути застосовані підходи, що ґрунтуються на автоенкодерах [5].

1.3. Методи аналізу трафіку: традиційні та сучасні підходи

Історично, для вирішення завдань управління мережевими ресурсами та забезпечення кібербезпеки, оператори зв'язку застосовували різноманітні методології

аналізу трафіку. Прийнято виділяти два фундаментальні покоління цих методів: перше, традиційне покоління, яке ґрунтувалося на аналізі безпосереднього вмісту пакетів, та друге, сучасне покоління, яке фокусується на аналізі поведінкових патернів та статистичних характеристик потоків даних.

1.3.1. Традиційний підхід: Deep Packet Inspection (DPI)

Технологія глибокої перевірки пакетів (DPI) предаставлена нижче на (Рисунок 1.3.) - є інструментом, що допомагає інтернет-провайдерам у профілюванні мережевих програм. Базуючись на системах DPI, інтернет-провайдери отримують можливість застосовувати диференційовані політики тарифікації, здійснювати формування трафіку або пропонувати гарантовану якість обслуговування (QoS) для окремих користувачів чи специфічних програм. Оскільки функціонування критично важливих мережевих сервісів значною мірою залежить від точної характеристики мережевих потоків, побудова адаптивних та ефективних систем DPI останнім часом стала важливою сферою наукових досліджень.

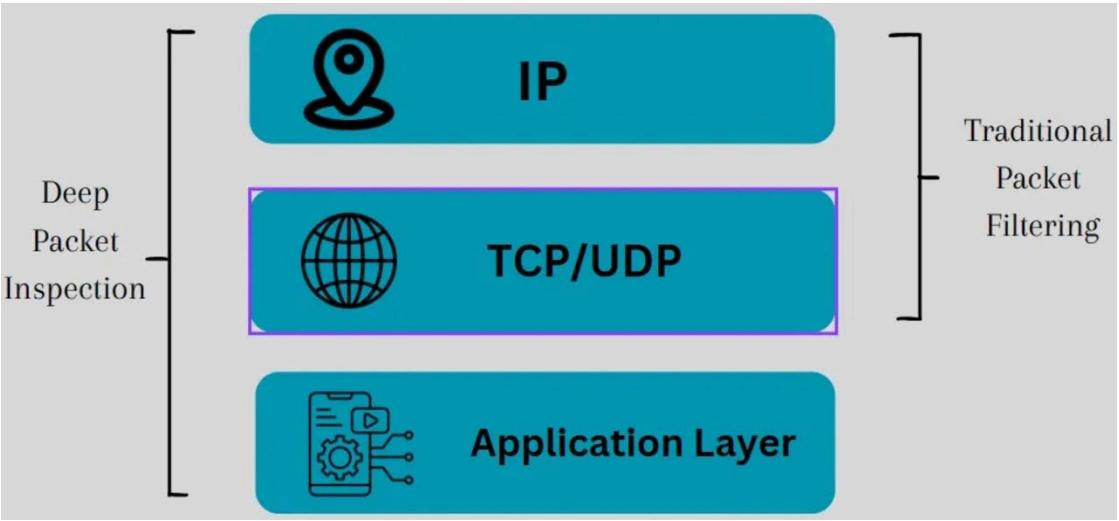

Рис. 1.3. Схема глибокої перевірки пакетів

1.3.2. Проблема DPI: "Стіна" тотального шифрування)

Головною проблемою, що нівелює практичну придатність глибокої перевірки пакетів (DPI) у сучасних умовах, є повсюдне впровадження шифрування. Домінування протоколу HTTPS, який наразі охоплює понад 90-95% усього веб-

трафіку, означає, що корисне навантаження (payload) кожного пакету є криптографічно захищеним. Ситуація ускладнюється еволюцією самих протоколів шифрування: якщо старіші версії, зокрема TLS 1.2, залишали певні метадані (наприклад, індикацію імені сервера, SNI) у відкритому вигляді під час "рукостискання" (handshake), дозволяючи DPI ідентифікувати принаймні цільовий домен, то новітні стандарти, такі як TLS 1.3 та QUIC, шифрують навіть цю службову інформацію. В результаті, для системи DPI переважна більшість мережевого трафіку виглядає як непроникний "чорний ящик" або беззмістовна послідовність зашифрованих бітів. Спроби активного дешифрування трафіку "на льоту", зокрема через атаки типу "Man-in-the-Middle", є надзвичайно ресурсозатратними, складними у технічній реалізації та водночас порушують фундаментальні принципи конфіденційності користувачів [6].

1.3.3. Машине навчання

У зв’язку з неможливістю аналізу вмісту пакетів (payload), сучасний підхід до аналізу трафіку фокусується на поведінкових аспектах та метаданих потоків, які неможливо приховати навіть за допомогою шифрування. Цей метод, відомий як Інтелектуальний аналіз трафіку або Encrypted Traffic Analytics (ETA), полягає у тому, що система аналізує не вміст комунікації, а її логістичні та статистичні характеристики. Моделі машинного навчання (ML) навчаються на великих обсягах трафіку з метою виявлення унікальних "відбитків" (fingerprints), що характеризують різні типи додатків.

Ключові ознаки (features), що використовуються для такого аналізу, включають статистичні характеристики потоку, такі як розподіл розмірів пакетів, часові інтервали між їх прибуттям (Inter-Arrival Time), загальна тривалість сесії, а також сумарна кількість байтів та пакетів у висхідному та низхідному напрямках. Крім того, аналізуються поведінкові патерни, що дозволяють диференціювати трафік: чи є він "вибуховим" (bursty), що характерно для відеострімінгу; чи є він симетричним та постійним, як у онлайн-іграх; або ж чи складається він з малих пакетів із тривалими паузами, що притаманно пристроям Інтернету Речей. Навіть залишкові метадані, такі

як інформація з DNS-запитів або дані з TLS-рукостискання (де послідовність, розмір та часові параметри повідомлень ClientHello/ServerHello створюють "відбиток" навіть у TLS 1.3), надають цінну інформацію.

Використовуючи сукупність цих ознак, моделі машинного навчання, такі як Random Forest, SVM, а також нейронні мережі та автоенкодери, можуть з високою точністю класифікувати зашифрований трафік. Найважливішою перевагою цього підходу є здатність виявляти аномалії, наприклад, ідентифікувати діяльність ботнетів, яка маскується під звичайний HTTPS-трафік [6].

1.4. Огляд систем інтелектуального аналізу трафіку (AI/ML-рішення)

Перехід від методів глибокої інспекції пакетів (DPI) до інтелектуального аналізу (AI/ML) є процесом, що активно відбувається як у комерційному секторі, так і в академічних наукових колах. Ця трансформація являє собою не просто оновлення інструментарію, а фундаментальну зміну парадигми в управлінні мережами та забезпеченні їхньої безпеки. Традиційні системи DPI, хоч і були ефективними в ідентифікації трафіку на основі відомих сигнатур та чітких правил, в епоху вибухового зростання обсягів даних, повсюдного шифрування та стрімкого ускладнення кіберзагроз, демонструють дедалі більшу обмеженість своїх можливостей. На противагу статичному та реактивному підходу DPI, штучний інтелект та машинне навчання пропонують динамічний та проактивний аналіз. Замість простої констатації типу трафіку, інтелектуальні системи здатні аналізувати поведінкові патерни, виявляти ледь помітні аномалії та прогнозувати майбутні події. Це уможливлює ідентифікацію нових, раніше невідомих загроз, що не мають сигнатур, та виявлення складних атак, замаскованих під легітимну мережеву активність.

Машинне навчання (ML) визначається як галузь штучного інтелекту (ШІ), що надає комп'ютерним системам здатність до навчання та вдосконалення на основі досвіду без необхідності явного програмування. Ця дисципліна охоплює розробку алгоритмів, які дозволяють системам автоматично навчатися та формувати прогнози

чи приймати рішення на основі даних. По суті, алгоритми ML використовують статистичні методи для ідентифікації закономірностей та кореляцій у наборах даних, що дає їм змогу узагальнювати попередні приклади та робити точні прогнози на основі раніше невідомих даних.

У контексті кібербезпеки машинне навчання відіграє вирішальну роль у вдосконаленні процесів виявлення загроз, оцінки ризиків та ідентифікації аномалій. Завдяки аналізу величезних обсягів даних, алгоритми ML здатні виявляти патерни, що вказують на шкідливу діяльність, уможливлюючи раннє виявлення та пом'якшення наслідків загроз безпеці. Методи ML є особливо ефективними у цій сфері через їхню здатність обробляти великомасштабні дані в режимі реального часу та адаптуватися до загроз, що постійно еволюціонують.

Основні сфери застосування ML для підвищення рівня захисту в кібербезпеці є різноманітними. До них належить виявлення загроз, де алгоритми аналізують мережевий трафік, системні журнали та поведінку користувачів для ідентифікації аномальних патернів, що свідчать про зараження шкідливим ПЗ чи спроби вторгнення. Тісно пов'язаним є виявлення аномалій, де моделі ML вивчають нормальну поведінку систем і виявляють будь-які відхилення від цієї базової лінії, що є корисним для ідентифікації атак "нульового дня" та внутрішніх загроз. Крім того, ML застосовується для прогностичного аналізу, передбачаючи потенційні інциденти безпеки на основі історичних даних, що дозволяє проактивно керувати ризиками. Методики ML також використовуються в управлінні вразливостями для аналізу коду та конфігурацій з метою пріоритезації усунення слабких місць. Нарешті, поведінковий аналіз на основі ML дозволяє виявляти підозрілу активність та складні постійні загрози (APT), які обходять традиційні засоби захисту [7].

У комерційному секторі, особливо в галузях телекомунікацій та кібербезпеки, впровадження AI/ML стає критичною операційною необхідністю. Компанії використовують ці технології для автоматизації захисту мереж, миттєвого виявлення інцидентів та аналізу поведінки користувачів і пристроїв для запобігання внутрішнім загрозам. Окрім аспектів безпеки, це сприяє оптимізації продуктивності мережі, прогнозуванню пікових навантажень та гарантуванню якості послуг (QoS) для

критично важливих додатків, що безпосередньо впливає на рівень задоволеності клієнтів. В академічному середовищі науковці активно досліджують та розробляють нові, більш ефективні алгоритми для аналізу мережевого трафіку в реальному часі. Дослідження зосереджені на створенні моделей, здатних навчатися на величезних масивах зашифрованих даних, не порушуючи приватності, а також на розробці нових підходів до управління складними, віртуалізованими мережевими інфраструктурами, де ШІ бере на себе завдання автоматичного конфігурування та самовідновлення. Таким чином, цей перехід є логічною еволюцією, зумовленою тим, що мережі стають занадто складними для ефективного управління вручну або за допомогою простих правил. Майбутнє належить "розумним" мережам, які здатні самостійно адаптуватися, оптимізуватися та захищатися, і саме інтеграція AI/ML є ключовим рушієм цієї неминучої трансформації [3].

1.4.1. Комерційні рішення (На прикладі Cisco ETA)

Аналіз зашифрованого трафіку Cisco (Encrypted Traffic Analysis, ETA) є методом, що використовує машинне навчання для виявлення шкідливого контенту та програм у зашифрованих даних без необхідності їх дешифрування. Для розуміння цього процесу важливо зазначити, що у зашифрованих даних використовуються два ключові елементи для ідентифікації шкідливої активності в тунелі TLS. Першим елементом є початковий пакет даних (IDP), який надає інформацію про узгодження "рукостискання" (handshake) TLS між клієнтом та сервером. Повідомлення цього "рукостискання", такі як версія TLS, набори шифрів, цифрові сертифікати та відкритий ключ сервера, обмінюються у відкритому тексті. Функція ETA може використовувати цю інформацію як ознаку незвичної активності; наприклад, самопідписаний сертифікат може вказувати на ненадійний сервер, який потенційно є шкідливим веб-сайтом або командно-контрольним сервером (C2C).

Другим елементом є послідовність довжин та часу пакетів (SPLT), яка фіксує довжину пакетів, якими обмінюються клієнт і сервер, а також часові інтервали між надходженнями цих пакетів. ETA може генерувати звіти на основі довжин та часу пакетів для аналізу зашифрованого трафіку та виявлення шкідливої діяльності.

Поєднуючи інформацію з IDP та SPLT, ETA здатна розрізняти легітимний (звичайний) трафік та аномальний трафік за допомогою звітів, що відображають деталі узгодження TLS (з IDP) та патерни довжини/часу надходження пакетів, які можуть вказувати на з'єднання з C2C-сервером .

При аналізі інформації IDP, звичайний зашифрований трафік, як правило, використовує цифровий сертифікат, підписаний легітимним центром сертифікації (CA), тоді як шкідливий трафік (наприклад, від ПЗ Bestafera) може використовувати самопідписаний сертифікат. Аналіз інформації SPLT також виявляє відмінності: звіт для легітимного запиту (наприклад, пошук Google) відображає кілька пакетів, надісланих клієнтом на сервер, після чого сервер відповідає багатьма пакетами, що містять результати пошуку. На противагу, звіт про аномальний трафік може демонструвати велику кількість трафіку, надісланого клієнтом на сервер, що вказує на незвичну вихідну активність, яка може свідчити про витік даних. Ця активність супроводжується наступною послідовністю з кількох пакетів, що надсилаються сервером клієнту з іншим часовим інтервалом порівняно з вихідним трафіком. Таким чином, зіставлення часу прибуття та розмірів пакетів як у вхідному, так і у вихідному напрямках, у поєднанні з метаданими TLS-підтвердження, надає функції ETA достатньо інформації для розрізнення легітимного трафіку та шкідливих даних, що представлено нижче на (Рисунок 1.4.).

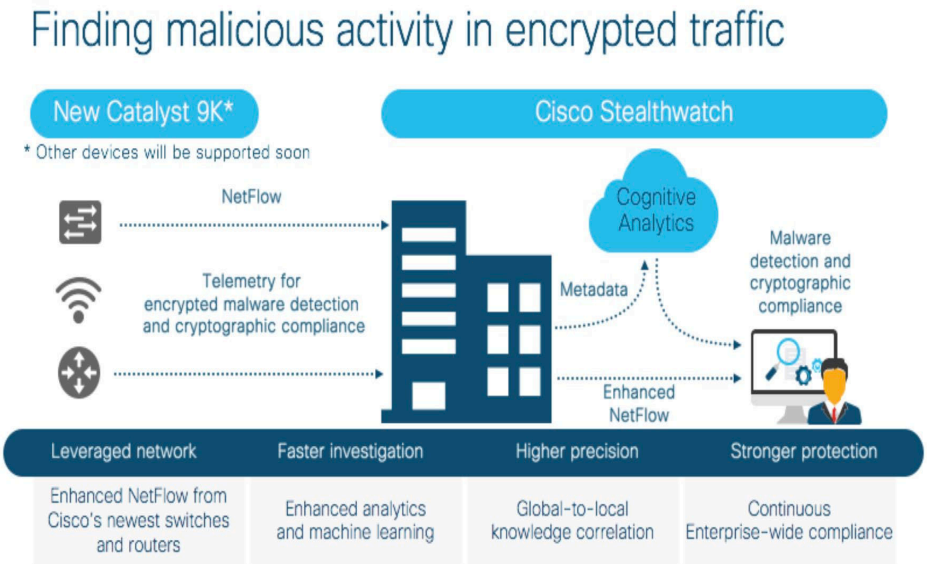

Рис. 1.4. Схема аналізу трафіку ETA

1.4.2. Академічні дослідження (Стан науки, SOTA)

В академічному середовищі було запропоновано та досліджено широкий спектр моделей машинного навчання для аналізу зашифрованого трафіку. Як зазначається у фундаментальних оглядових роботах, ці підходи можна умовно розділити на два покоління. До першого покоління належать "класичні" моделі машинного навчання, які, хоча й продемонстрували високу ефективність, часто вимагають ретельного ручного конструювання ознак ("feature engineering"). Серед них – Random Forest (RF), ансамблевий метод, що добре зарекомендував себе при роботі зі статистичними ознаками потоків, та Support Vector Machines (SVM), ефективний для вирішення завдань бінарної класифікації, наприклад, "VPN" проти "не-VPN".

Друге покоління представлене моделями глибокого навчання (Deep Learning), які наразі вважаються передовим станом науки (State-of-the-Art, SOTA) завдяки їхній здатності автоматично вивчати складні патерни без необхідності ручного виділення ознак. У цьому контексті, згорткові нейронні мережі (CNN) часто застосовуються для класифікації, представляючи потік трафіку як двовимірне "зображення", де пікселями є параметри пакетів, такі як розмір та час. Рекурентні нейронні мережі (RNN/LSTM) є ефективними для аналізу трафіку як часового ряду, оскільки вони враховують послідовну природу пакетних даних. Особливу популярність саме для задачі виявлення аномалій здобули автоенкодери (Autoencoders). Цей тип нейронних мереж навчається на "нормальному" трафіку, опановуючи його стиснення та подальше відновлення. Коли на вхід моделі подається "аномальний" трафік, наприклад, від ботнета або атаки "нульового дня", автоенкодер не може коректно його реконструювати. Це призводить до високої "помилки відновлення" (reconstruction error), яка слугує надійним сигналом про наявність загрози. Таким чином, наукова спільнота підтверджує, що майбутнє аналізу трафіку лежить саме у сфері глибокого навчання, зокрема у використанні автоенкодерів для виявлення аномалій, що є центральною задачею для відповідних магістерських досліджень [19].

ВИСНОВКИ ДО РОЗДІЛУ 1

У даному розділі було проведено комплексний аналіз сучасного стану проблеми аналізу трафіку в мобільних мережах. У ході дослідження було розглянуто еволюцію мобільних архітектур від 4G (LTE) до 5G (SBA), різницю між ними та визначено ключові сценарії використання (eMBB, URLLC, mMTC), які генерують суттєво гетерогенні типи трафіку. Було здійснено детальну класифікацію цих типів, зокрема відео, інтерактивного трафіку та трафіку Інтернету Речей, а також описано їхні унікальні "відбитки" та специфічні вимоги до якості обслуговування (QoS). Робота продемонструвала, що традиційні методи аналізу, такі як глибока інспекція пакетів (DPI), втрачають свою ефективність в умовах тотального поширення сучасних протоколів шифрування, включно з TLS 1.3 та QUIC. Натомість, було обґрунтовано необхідність переходу до сучасних методів на основі штучного інтелекту та машинного навчання (AI/ML), які здатні аналізувати поведінкові та статистичні характеристики потоків без необхідності їх дешифрування. Проведений огляд існуючих комерційних (наприклад, Cisco ETA) та академічних (CNN, Autoencoders) систем підтвердив актуальність та практичну реалізованість обраного інтелектуального підходу.

РОЗДІЛ 2
ТЕОРЕТИЧНІ ОСНОВИ ІНТЕЛЕКТУАЛЬНОГО АНАЛІЗУ ТРАФІКУ

2.1. Математичні моделі опису трафіку в мобільних мережах

Динамічний розвиток мобільних мереж, зокрема впровадження стандартів п’ятого (5G) та розробка шостого (6G) поколінь, призводить до експоненціального зростання обсягів трафіку та появи нових, гетерогенних типів послуг. Ефективне управління ресурсами мережі, забезпечення гарантованої якості обслуговування (QoS) та оптимізація пропускної здатності вимагають глибокого розуміння природи трафіку, яке досягається шляхом розробки та застосування точних математичних моделей. Ці моделі є фундаментальним інструментом для аналізу продуктивності, планування ємності та розробки протоколів управління в бездротових системах.

Історично моделювання трафіку, особливо в мережах з комутацією каналів, значною мірою спиралося на процес Пуассона. Ця модель, що базується на припущенні про незалежність та стаціонарність подій (надходження викликів), є аналітично зручною завдяки своїй властивості відсутності пам’яті. На основі пуассонівських потоків будується класична теорія масового обслуговування (ТМО), зокрема моделі Ерланга, які дозволяють розраховувати ключові показники продуктивності, такі як ймовірність блокування виклику або середній час очікування. Ці підходи були достатньо адекватними для опису традиційного голосового трафіку.

Однак із переходом до мереж з комутацією пакетів та домінуванням Інтернет-трафіку (веб-браузинг, потокове відео, файлообмінні мережі) було виявлено, що пуассонівська модель суттєво недооцінює "імпульсність" або "пакетність" (burstiness) реальних потоків даних. Емпіричні дослідження, починаючи з 1990-х років, продемонстрували наявність ефектів самоподібності (self-similarity) та довгострокової залежності (Long-Range Dependence, LRD) у трафіку даних. Це означає, що імпульсність спостерігається на різних часових масштабах, від

мілісекунд до годин, на відміну від пуассонівського процесу, який згладжується при усередненні.

Для опису самоподібного трафіку використовуються складніші математичні апарати, такі як дробовий броунівський рух (fBm), моделі авторегресії дробово-інтегрованого ковзного середнього (FARIMA) та розподіли з "важкими хвостами" (heavy-tailed distributions), зокрема розподіл Парето. Ці моделі краще відображають реальність, де невелике число тривалих сесій або передачі великих файлів генерує значну частину навантаження, що критично впливає на розміри буферів та затримки в мережі.

Специфіка мобільних мереж вносить додаткові ускладнення. На відміну від стаціонарних дротових мереж, тут необхідно враховувати мобільність абонентів, що призводить до процедур хендовера, та мінливість умов радіоканалу (завмирання, інтерференція). Сучасні моделі для LTE та 5G намагаються інтегрувати ці аспекти, поєднуючи просторово-часові моделі мобільності (наприклад, Random Waypoint) з моделями надходження трафіку. Крім того, диверсифікація послуг у 5G (eMBB, URLLC, mMTC) вимагає розробки гібридних моделей, здатних описувати трафік з абсолютно різними характеристиками – від інтенсивного потокового відео до спорадичних коротких повідомлень від пристроїв Інтернету речей (IoT).

Напрямки майбутніх досліджень зосереджені на використанні методів машинного навчання (ML) та штучного інтелекту (AI) для створення адаптивних моделей трафіку. Ці підходи дозволяють в режимі реального часу аналізувати складні патерни, прогнозувати навантаження та динамічно оптимізувати ресурси мережі, що є життєво необхідним для управління ультращільними та гетерогенними мобільними системами майбутнього [9].

Для ефективного управління мережею, прогнозування навантаження та виявлення аномалій необхідне математичне розуміння природи трафіку. Протягом десятиліть було запропоновано декілька моделей, що еволюціонували разом із самими мережами.

2.1.1. Класична модель: Пуассонівський процес

Процес Пуассона є одним із найбільш фундаментальних та поширених процесів підрахунку у теорії ймовірностей. Він, як правило, використовується для моделювання сценаріїв, у яких підраховуються випадки певних подій, що відбуваються з певною фіксованою середньою інтенсивністю (частотою), однак час настання цих подій є абсолютно випадковим і не має жодної регулярної структури. Як ілюстративний приклад, якщо з історичних даних відомо, що землетруси в певній місцевості трапляються із середньою частотою два на місяць, але конкретні моменти їх виникнення здаються стохастично розподіленими, процес Пуассона може слугувати адекватною моделлю для опису цього явища. Ця модель також широко застосовується для опису інших феноменів, таких як кількість автомобільних аварій у певній місцевості, просторове розташування користувачів у бездротовій мережі, надходження запитів на окремі документи до веб-сервера, моменти виникнення воєнних конфліктів або реєстрація фотонів, що потрапляють на фотодіод, що показано на формулі (2.1) ймовірності $P(k)$ та інтенсивності λ.

$$P(k,\lambda t) = \frac{e^{-\lambda t}(\lambda t)^{k}}{k!}, \tag{2.1}$$

В історичній ретроспективі, перші моделі, що описували функціонування телефонних мереж, а згодом і ранніх комп'ютерних мереж, базувалися на Пуассонівському процесі. Ця стохастична модель слугує наріжним каменем теорії масового обслуговування. Її фундаментальне припущення полягає в тому, що події, наприклад, надходження телефонних дзвінків або мережевих пакетів, відбуваються незалежно одна від одної з певною постійною середньою інтенсивністю, що позначається як λ. Відповідно до цієї моделі, часові інтервали між послідовними надходженнями подій описуються експоненційним розподілом. Зазначена модель адекватно описувала трафік у традиційних телефонних мережах із комутацією каналів, де виклики дійсно були значною мірою незалежними та відносно рідкісними подіями. Проте, численні дослідження, проведені у 1990-х роках, емпірично довели,

що трафік сучасних мереж передачі даних не підпорядковується Пуассонівському закону. Головний недолік моделі полягає в її нездатності врахувати феномен "вибуховості" трафіку, який детально обговорювався в розділі 1. На відміну від припущення про незалежність, у реальному трафіку пакети часто надходять "пачками" де прибуття одного пакета значно підвищує ймовірність майже негайного прибуття наступного [17].

Процес Пуассона в моделюванні з'єднань LTE-A - це фундаментальна концепція, що використовується в аналітичному моделюванні динаміки встановлення з'єднань Long-Term Evolution Advanced (LTE-A), яка служить граничним розподілом для кількості запитів, що беруть участь у системі.

Процедура з'єднання LTE-A включає перевірку заборони доступу, а потім суперечку за преамбулу. Попередні аналітичні дослідження цієї процедури часто передбачали характеристики Пуассона для ключових випадкових величин системи, таких як кількість запитів користувацького обладнання, схема представлена нижче на (Рисунок 2.1). Однак ці дослідження зазвичай не надавали підтвердження цього припущення.

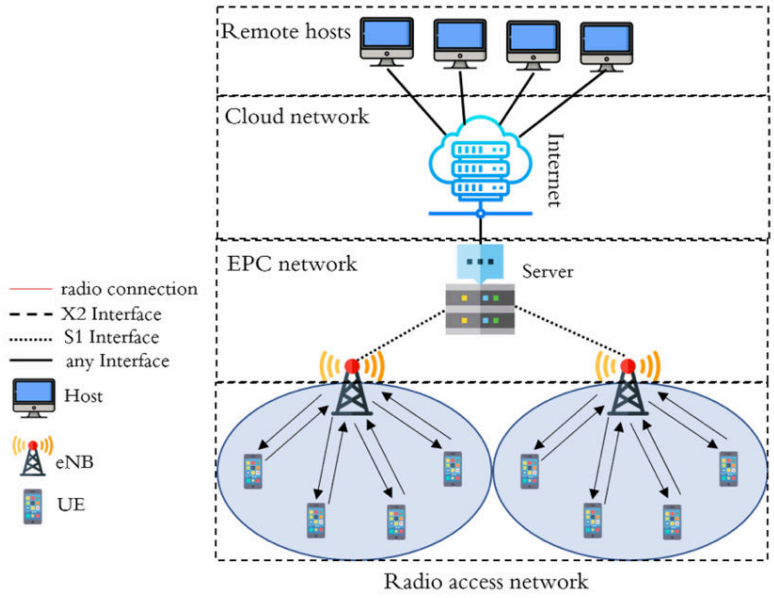

Рис. 2.1. Типовий зв'язок користувацького обладнання (UE) в мережі 5-го покоління (5G)

Структура рівноважного аналізу моделює як ABC, так і суперечку за преамбулу, використовуючи випадкові величини Пуассона. Ключовим кроком у цьому виведенні є моделювання стохастичного процесу (x_{n+1}), що представляє кількість кандидатів на запити UE в послідовних слотах, щоб він мав характеристики процесу Пуассона. Якщо загальна кількість запитів-кандидатів на UE (x_{n+1}) відповідає цій моделі, то результуюча кількість фактично переданих запитів UE (t_{n+1}), які очищають заборону, також має розподіл Пуассона. Цей підхід до моделювання особливо важливий для систем з кількома преамбулами (O>1), оскільки він спирається на властивість розподілу Пуассона для аналізу кількості щойно згенерованих запитів UE (a_{n+1}) та запитів-кандидатів на UE (x_{n+1}).

Центральним внеском цього аналізу є формальна демонстрація того, що механізм рівномірного відкладення, властивий LTE-A, призводить до збіжності системної динаміки до цих характеристик Пуассона. Цей механізм обробляє запити UE, що завантажилися, - ті, що не пройшли ABC або зіткнулися під час суперечки преамбул, - шляхом рівномірного та випадкового перепланування їх на інтервал відкладення $(0, T_{max})$ слотів. Збіжність до динаміки Пуассона формально демонструється шляхом аналізу спрощеної «системи встановлення циркуляційного з'єднання», де запити циркулюють, але не надходять і не виходять [20].

2.1.2. Застосування Теорії черг: Аналіз QoS та ємності мережі

Мережі мобільного зв'язку стикаються із серйозними викликами у забезпеченні належної якості обслуговування (QoS). У сучасній економічній ситуації, що характеризується глобальною економічною кризою та конкурентним бізнес-середовищем з низькими тарифами, надання високої якості обслуговування стає необхідністю. Щоб вижити в цих умовах, оператори мереж змушені зосереджуватися на ефективному використанні доступних ресурсів, зокрема каналів, що вимагає ефективного проектування та планування мережі. Основними параметрами, що використовуються для цього, є ймовірність блокування, ймовірність блокування при передачі (хендовері) та коефіцієнт використання.

Ймовірність блокування слугує для контролю кількості заблокованих викликів, що виникають через брак ємності мережі (каналів) для розміщення всіх запитів у певний момент часу, і визначається кількістю доступних каналів та навантаженням трафіку в Ерлангах, це наглядно показано в формулі (2.2) Ерланга для розрахунку ймовірності блокування P_B (модель Ерланга B)

$$P_B(A, N) = \frac{A^N / N!}{\sum_{i=0}^{N} A^i / i!}, \tag{2.2}$$

При оцінці продуктивності ймовірності блокування хендовера необхідно враховувати низку параметрів, таких як швидкість хендовера, ймовірність скидання виклику та відповідні часові характеристики. Ймовірність скидання викликів - це ймовірність того, що виклик, якому початково було надано доступ до каналу, примусово обривається під час розмови через технічні помилки (наприклад, електромагнітні причини). Оцінка таких скидань викликів у контексті переходу між сотами називається ймовірністю блокування хендовера.

Швидкість хендовера використовується для оцінки інтенсивності надходження трафіку хендовера і є вхідним параметром при визначенні загальної ймовірності блокування. Час утримання каналу визначається часом перебування мобільної станції (MS) в одній соті під час виклику, на що впливає мобільність абонента, географічне положення та схеми розподілу каналів. На відміну від цього, час утримання виклику - це загальна тривалість виклику, яка може охоплювати декілька процесів передачі обслуговування (хендоверів) при переміщенні MS між сотами. Для оцінки обсягу пропонованого навантаження трафіку та частки заблокованих викликів у системі необхідно розгортати спеціалізовані моделі. Огляд літератури свідчить, що моделі теорії масового обслуговування (моделі черг) були успішно застосовані для вирішення таких завдань аналізу ємності [6].

2.1.3. Сучасна модель: Самоподібний (фрактальний) трафік

Парадигма моделювання трафіку в комп’ютерних мережах зазнала кардинальних змін на початку 1990-х років. До цього періоду домінуючою моделлю слугував процес Пуассона, який адекватно описував телефонний трафік, базуючись на припущеннях про незалежність подій (прибуття викликів) та відсутність пам'яті (марковська властивість). Пуассонівські моделі прогнозували, що при агрегації багатьох незалежних джерел, сумарний трафік стає "гладким" та статистично передбачуваним. Однак емпіричні вимірювання реального трафіку в мережах Ethernet виявили суттєву розбіжність із цією теорією. Історичний прорив стався завдяки дослідженню, проведеному в Bellcore групою науковців, включно з Віллом Леландом, Мюрадом Такку та Вальтером Віллінгером. У своїй фундаментальній роботі "On the Self-Similar Nature of Ethernet Traffic" (1993-1994) вони, проаналізувавши високоточні вимірювання трафіку, виявили разючі результати. На відміну від пуассонівського трафіку, який згладжувався при усередненні, реальний трафік Ethernet демонстрував "імпульсивність" (burstiness) на всіх часових масштабах. Іншими словами, структура трафіку виглядала однаково "колючою" незалежно від обраного масштабу, що є властивістю, ідентичною фрактальним об'єктам у математиці. Це призвело до виникнення термінів "самоподібний" або "фрактальний" трафік [18].

Самоподібність трафіку математично описується двома взаємопов'язаними концепціями: довгостроковою залежністю (Long-Range Dependence, LRD) та параметром Херста (H). LRD є ключовою властивістю, що означає наявність статистичної кореляції між значеннями трафіку в даний момент та значеннями в далекому майбутньому. Це свідчить про наявність "довгої пам'яті" у процесі: інтенсивний сплеск трафіку підвищує ймовірність майбутніх сплесків. У пуассонівських моделях (з короткостроковою залежністю) кореляція зникає експоненційно швидко, тоді як у LRD-процесах вона спадає гіперболічно, тобто значно повільніше. Параметр Херста (H) є кількісним показником для вимірювання ступеня самоподібності, зі значеннями в діапазоні [0, 1]. Значення $H = 0.5$ вказує на відсутність LRD і характерне для класичних процесів, як-от пуассонівський. Натомість діапазон $0.5 < H < 1$ вказує на наявність довгострокової залежності,

причому чим ближче H до 1, тим сильніший ступінь самоподібності. Дослідження Bellcore емпірично встановили, що для Ethernet-трафіку H стабільно знаходиться в діапазоні 0.7-0.9, що стало прямим доказом неадекватності пуассонівських моделей, що було доведено в формулі (2.3) формального визначення параметра Херста (H).

$$H = 1 - \frac{\beta}{2}, \tag{2.3}$$

Подальші дослідження встановили, що фізичною причиною самоподібності на рівні агрегованого трафіку є наявність розподілів із "важкими хвостами" (heavy-tailed distributions), наприклад, розподілу Парето, на рівні окремих джерел. На відміну від експоненціальних розподілів, де ймовірність екстремальних значень дуже мала, "важкі хвости" припускають нетривіальну ймовірність дуже великих значень, таких як розміри файлів чи тривалість сесій користувачів. Агрегація тисяч таких процесів породжує сумарний потік із властивостями самоподібності та LRD.

Відкриття самоподібності мало революційні інженерні наслідки, оскільки моделі, що базувалися на Пуассоні, систематично недооцінювали імпульсивність трафіку, даючи надто оптимістичні прогнози. У теорії масового обслуговування це пояснило, чому просте збільшення буферів у маршрутизаторах часто не вирішувало проблему втрати пакетів: на відміну від короткочасних пуассонівських черг, самоподібні сплески можуть тривати досить довго, щоб переповнити навіть дуже великі буфери. Це розуміння також вплинуло на розробку більш стійких механізмів управління заторами (congestion control). Незважаючи на зміну протоколів та додатків, властивості самоподібності та LRD продовжують спостерігатися в сучасних мережах 5G при агрегації потокового відео та трафіку IoT. Таким чином, самоподібна модель трафіку являє собою парадигмальний зсув, що замінив класичні моделі як більш точний інструмент для аналізу продуктивності та забезпечення QoS у сучасних мережах [5].

2.1.4. Компромісні моделі: Марковські процеси

Між простим Пуассонівським процесом і складним самоподібним існують компромісні моделі, що базуються на Марковських процесах.

У багатьох прикладних задачах, особливо в галузі моделювання телекомунікацій, виникають точкові процеси, де інтенсивність надходження подій випадково змінюється з часом. Традиційні пуассонівські моделі, які базуються на припущенні про постійну середню інтенсивність, виявляються нездатними адекватно описати складну, "імпульсну" природу сучасного трафіку, такого як пакетні дані чи потоки, що переповнюються. Для моделювання таких процесів широке застосування знайшов Марковський модульований процес Пуассона (MMPP). Його популярність обґрунтована тим, що він якісно відображає інтенсивність надходження, яка динамічно змінюється, і, що важливо, він здатний враховувати значні кореляції між часовими інтервалами надходження. Незважаючи на цю імпліцитну складність, MMPP залишається аналітично керованим, що дозволяє використовувати його для детального аналізу продуктивності, зокрема в теорії масового обслуговування. MMPP та пов'язані з ним моделі черг стали предметом багатьох досліджень, і даний огляд узагальнює основні визначення, властивості та алгоритми, спираючись на раніше представлені результати, зокрема, у збірці Фішера та Майєр-Геллстерн. Фундаментально, Марковський модульований процес Пуассона визначається як подвійно стохастичний процес Пуассона. У такій конструкції інтенсивність надходження сама по собі є випадковим процесом. Конкретно у випадку MMPP, ця інтенсивність детермінується станом J(t) базового, неперервного в часі Марковського ланцюга. Припускається, що цей базовий ланцюг є незвідним і має m станів. Коли ланцюг перебуває у стані i, надходження (наприклад, пакетів) відбуваються відповідно до стандартного процесу Пуассона з інтенсивністю λi. Таким чином, процес MMPP повністю параметризується двома ключовими елементами: інфінітезимальною матрицею-генератором (Q), яка є $m \times m$ матрицею, що описує ймовірності переходів та час перебування у кожному стані базового ланцюга, та матрицею інтенсивностей λ, яка є діагональною $m \times m$ матрицею, що містить відповідні інтенсивності $\lambda_1, \ldots, \lambda_m$.

Передбачається, що процес є однорідним у часі, тобто матриці Q і λ не залежать від t. Для аналізу також важливим є стаціонарний вектор ймовірностей π базового Марковського ланцюга, який відображає довгострокову частку часу, яку процес проводить у кожному зі станів. Поширеною помилкою є припущення, що MMPP є процесом відновлення. Процес відновлення характеризується тим, що часові інтервали між послідовними подіями є незалежними та однаково розподіленими, однак MMPP, у загальному випадку, не задовольняє цю умову. Насправді, MMPP коректніше описується як Марковський процес відновлення (MRP). Це можна зрозуміти інтуїтивно: якщо розглянути два послідовних надходження, де перше відбулося у стані i, а друге – у стані j, то розподіл часу між цими подіями залежатиме як від i, так і від j, а також від шляху, який Марковський ланцюг пройшов між цими станами. Оскільки розподіл часу між $(k-1)$ -м та k-м надходженнями залежить від станів ланцюга в моменти цих надходжень, процес не є простим процесом відновлення. Послідовність пар (стан ланцюга при n-му надходженні, час до n-го надходження) утворює Марковську послідовність відновлення, перехідні ймовірності якої виражаються через матричні експоненти. Для повного визначення процесу необхідно вказати його початковий стан, що призводить до двох основних версій стаціонарного MMPP. Перша, інтервально-стаціонарна (Interval-stationary), виходить, якщо процес починається у "довільний" момент надходження, де початковий розподіл p є стаціонарним вектором для вкладеного Марковського ланцюга. Друга, стаціонарна за середовищем (Environment-stationary), виходить, якщо початковий розподіл обирається як π, тобто стаціонарний розподіл самого базового ланцюга. У цьому випадку "час нуль" є довільним моментом часу, коли базовий "середовищний" процес J(t) вже досяг стаціонарного режиму, і ця версія є стохастично еквівалентною так званій часово-стаціонарній версії. Хоча MMPP загалом не є процесом відновлення, існують спеціальні умови, за яких він ним стає. Достатньою умовою є те, що надходження відбуваються лише в одному з m станів. Найпростішим і найважливішим таким спеціальним випадком є Переривчастий процес Пуассона (Interrupted Poisson Process, IPP). IPP визначається як 2-державний MMPP, в якому одна з інтенсивностей надходження дорівнює нулю. Це означає, що процес

чергується між станом "ON", де надходження відбуваються з інтенсивністю $\lambda > 0$, та станом "OFF", де надходження відсутні. Було доведено, що IPP є стохастично еквівалентним гіперекспоненціальному процесу відновлення (H2). Це означає, що розподіл часу між надходженнями в IPP є ідентичним до розподілу в процесі відновлення з гіперекспоненціальним розподілом, і існують точні формули для перетворення параметрів між 2-державним IPP та еквівалентним H2-розподілом. Аналітична керованість MMPP дозволяє виводити багато його властивостей. На відміну від стандартного процесу Пуассона, час між надходженнями в MMPP є корельованим; можна отримати матричні вирази для умовних моментів часу, диференціюючи матрицю перетворень Лапласа-Стілтьєса, що дозволяє обчислювати коваріаційні матриці. Важливою характеристикою є також рахункова функція N_t, тобто кількість надходжень в інтервалі $(0, t]$. Еволюція ймовірностей $P_{ij}(n, t)$ описується системою диференціальних рівнянь Чапмена-Kolмогорова, розв'язок якої отримується в термінах матричної породжуючої функції. З неї, в свою чергу, можна отримати моменти N_t. Однією з найпотужніших властивостей MMPP є те, що суперпозиція (накладання) n незалежних MMPP є знову MMPP, що має величезне практичне значення, оскільки мережевий трафік часто є результатом агрегації багатьох джерел. Якщо кожне i-те джерело описується матрицями (Q_i, λ), то результуючий сукупний процес MMPP має матриці (Q, λ) які обчислюються як суми Кронекера. Хоча це елегантно, розмірність результуючого MMPP швидко зростає до добутку всіх k_i станів. Однак, якщо накладаються n однакових 2-державних MMPP, складність значно зменшується, і результуючий процес має лише n+1 станів, що робить його обчислювально керованим. Основна аналітична цінність MMPP розкривається при його використанні як вхідного потоку для систем масового обслуговування, зокрема в моделі MMPP/G/1, що є одноканальною СМО із загальним (General) розподілом часу обслуговування. Аналіз цієї системи спирається на розгляд вкладеного Марковського процесу відновлення в моменти завершення обслуговування. Матриця перехідних ймовірностей цієї СМО має специфічну блокову структуру "типу M/G/1".

Наглядна схема MMPP представлена нижче на (Рисунку 2.2.), в якій представлено Верхній графік (ланцюг Маркова): Показано прихований ланцюг Маркова, що перемикається між станами (наприклад, "Стан 1 - Низька швидкість", "Стан 2 - Висока швидкість").

Нижній графік (Процес Пуассона): Ілюструє результуючий процес Пуассона, де інтенсивність (швидкість подій) змінюється відповідно до поточного стану ланцюга Маркова. Зверніть увагу, як події скупчуються щільніше в періоди "Високої швидкості" та рідше в періоди "Низької швидкості".

Стрілки: Вказують на взаємозв'язок та вплив ланцюга Маркова на процес Пуассона.

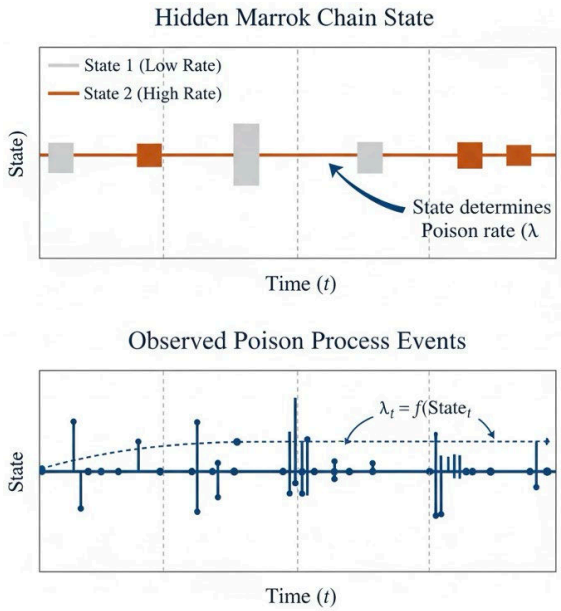

Рис. 2.2. Ілюстрація MMPP

Центральним елементом аналізу є матриця G, елемент G_{ij}якої являє собою ймовірність того, що період зайнятості, який почався зі станом MMPP i, зрештою завершиться зі станом MMPP i Ця матриця є розв'язком нелінійного матричного рівняння, що пов'язує G з матрицями Q, λ та перетворенням Лапласа часу обслуговування. Як тільки матриця G знайдена, можна обчислити багато ключових показників продуктивності. Наприклад, ймовірність порожньої системи (вектор x_0) може бути виражена явно через G Повний розподіл довжини черги в моменти відходу

(xi) можна обчислити за допомогою ефективного та чисельно стійкого рекурсивного алгоритму, розробленого Рамасвамі. Використовуючи ключову теорему відновлення, можна також пов'язати розподіл у моменти відходу (xi) з розподілом у довільний момент часу (yi) . Крім того, перетворення Лапласа для віртуального часу очікування можна отримати як узагальнення класичної формули Поллачека-Хінчіна, що дозволяє обчислити моменти часу очікування. Якщо вхідний потік є суперпозицією, модель навіть дозволяє обчислювати показники для кожного окремого потоку. MMPP широко використовується в телекомунікаціях, де дві основні сфери застосування — це моделювання потоків переповнення (overflow) з груп каналів з обмеженою ємністю та моделювання корельованого трафіку пакетних даних і голосу, наприклад, при аналізі продуктивності мереж Asynchronous Transfer Mode (ATM). Незважаючи на аналітичну потужність моделі, її практичне використання стикається з серйозною проблемою параметризації. Це завдання полягає в тому, щоб підібрати параметри матриць Q і λ таким чином, щоб згенерований MMPP адекватно відображав поведінку реального потоку трафіку. Найпоширенішим підходом для вирішення цієї проблеми є узгодження моментів (moment-matching). Цей метод полягає у виборі параметрів MMPP (часто 2-державного) таким чином, щоб збігалися кілька ключових статистичних характеристик моделі та реального процесу. Типовий набір для узгодження включає середню інтенсивність надходження, відношення дисперсії до середнього для кількості надходжень у короткому часовому інтервалі (для фіксації "імпульсивності") та довгострокове відношення дисперсії до середнього (для фіксації кореляцій). Для спрощення аналізу суперпозиції багатьох потоків часто використовують апроксимацію, де кожен окремий потік моделюється простим 2-державним IPP, параметри якого підібрані за методом узгодження моментів, а потім ці IPP об'єднуються в один сукупний MMPP. У підсумку, Марковський модульований процес Пуассона є фундаментальним і потужним інструментом в аналітичному моделюванні. Він являє собою вдалий компроміс між реалістичністю, завдяки здатності моделювати змінні в часі інтенсивності та кореляції між надходженнями, та аналітичною керованістю. Його властивості, такі як замикання відносно суперпозиції, та наявність ефективних алгоритмічних рішень для аналізу СМО типу MMPP/G/1,

зробили його стандартом де-факто для аналізу продуктивності в телекомунікаційних мережах. Хоча його застосування вимагає ретельної параметризації, MMPP залишається однією з найбільш корисних і універсальних моделей для процесів надходження в стохастичному моделюванні [4].

2.2. Методи машинного навчання для класифікації та прогнозування трафіку

Даний розділ представляє методи машинного навчання, що застосовуються для класифікації та прогнозування трафіку, і є логічним продовженням попереднього аналізу. Встановивши обмеженість суто математичних, керованих моделлю (model-driven) підходів, дослідження переходить до розгляду емпіричних, керованих даними (data-driven) методів, тобто машинного навчання (ML). Насамперед будуть описані "класичні" алгоритми ML, які слугують фундаментальною основою для більш складних нейронних архітектур. На відміну від аналітичних моделей, підхід машинного навчання полягає у тому, що модель самостійно вивчає складні, часто нелінійні залежності безпосередньо з даних. У контексті аналізу трафіку найчастіше використовуються методи керованого навчання (Supervised Learning). Суть цього методу полягає в тому, що модель "навчається" на великому наборі даних, де кожна сесія трафіку попередньо розмічена експертом (наприклад, як "відео", "гра" чи "ботнет"). У процесі навчання модель вчиться асоціювати набір вхідних ознак трафіку, таких як статистика пакетів, часові характеристики та протоколи, з відповідною вихідною міткою класу.

2.2.1. Метод k-Найближчих Сусідів (K-Nearest Neighbors, K-NN)

Метод k-Найближчих Сусідів (K-NN) являє собою непараметричний, інстанс-орієнтований (instance-based) метод, який базується на фундаментальному припущенні про близькість схожих об'єктів у просторі ознак. Теоретична основа цього алгоритму відрізняється від більшості інших методів машинного навчання, оскільки він не будує експліцитної "моделі" під час фази навчання; натомість, він

просто зберігає (запам'ятовує) весь навчальний набір даних. Процес класифікації для нового, нерозміченого об'єкта, такого як потік трафіку, представлений у вигляді вектора ознак, ініціюється вимірюванням відстані, наприклад, Евклідової, від цього нового вектора до кожного вектора у збереженому навчальному наборі. Алгоритм ідентифікує k найближчих "сусідів", тобто k найбільш схожих потоків з навчальної вибірки. Фінальна класифікація відбувається шляхом "голосування": новий потік отримує ту класову мітку, яка є домінуючою (зустрічається найчастіше) серед його k обраних сусідів. У контексті аналізу трафіку, K-NN може демонструвати ефективність для вирішення простих задач класифікації, проте він має суттєві практичні недоліки. По-перше, він є обчислювально дорогим на етапі інференції (у реальному часі), оскільки вимагає порівняння нового потоку з усією, потенційно багатомільйонною, навчальною вибіркою. По-друге, метод є чутливим до феномену "прокляття розмірності" (curse of dimensionality), коли його ефективність стрімко деградує зі збільшенням кількості ознак, що описують потік [16].

2.2.2. Метод Опорних Векторів (Support Vector Machines, SVM)

Метод Опорних Векторів (Support Vector Machines, SVM) є потужним алгоритмом керованого навчання, фундаментальною метою якого є знаходження оптимальної гіперплощини, що розділяє класи даних. Теоретична основа цього методу ґрунтується на кількох ключових концепціях. По-перше, це поняття самої гіперплощини: у простому двовимірному випадку (де об'єкти описуються двома ознаками) вона являє собою звичайну лінію, що розділяє дві групи точок, наприклад, "аномальний" та "нормальний" трафік. Однак у багатовимірному просторі ознак, де для опису потоку трафіку можуть використовуватися десятки чи сотні параметрів, ця розділова поверхня коректно називається гіперплощиною. По-друге, ключовою ідеєю SVM є принцип максимального зазору (maximum margin). Алгоритм шукає не просто будь-яку довільну розділову гіперплощину, а ту єдину, яка має максимальну відстань (зазор) до найближчих точок кожного з класів. Ці найближчі точки, що фактично "підтримують" гіперплощину і визначають її положення, називаються опорними векторами (support vectors); такий підхід, що максимізує зазор, забезпечує кращу

здатність моделі до узагальнення на нових даних. По-третє, для випадків, коли дані не є лінійно роздільними у вихідному просторі ознак, SVM застосовує так званий "ядерний трюк" (kernel trick). За допомогою спеціальних ядерних функцій (наприклад, радіальної базисної функції, RBF) метод неявно відображає дані у простір вищої розмірності, де лінійне розділення стає можливим. Завдяки своїй здатності ефективно оперувати у високорозмірних просторах ознак (працюючи з десятками і сотнями статистичних характеристик трафіку) та високій стійкості до перенавчання, метод опорних векторів протягом тривалого часу вважався одним із найкращих та найбільш надійних методів для вирішення завдань класифікації мережевого трафіку [15].

2.2.3. Випадковий Ліс (Random Forest)

Метод Випадкового Лісу (Random Forest, RF) являє собою потужний ансамблевий підхід у машинному навчанні, який функціонує шляхом побудови великої кількості окремих "дерев рішень" (Decision Trees) під час тренування та агрегування їхніх індивідуальних прогнозів для отримання остаточного результату. Теоретична основа цього методу ґрунтується на концепції дерева рішень, яке саме по собі є простим класифікатором, що має структуру, подібну до блок-схеми. Таке дерево послідовно ставить запитання до ознак об'єкта, наприклад, "Чи тривалість потоку перевищує 10 секунд?" або "Чи середній розмір пакета менший за 100 байт?", щоб дійти до класифікаційного рішення. Фундаментальна проблема окремого, особливо глибокого, дерева рішень полягає у його високій схильності до перенавчання; воно може ідеально "запам'ятати" навчальні дані, але виявиться нездатним до узагальнення на нових, раніше не бачених прикладах.

Випадковий Ліс вирішує цю проблему, навчаючи сотні або тисячі відносно неглибоких дерев, причому кожне дерево тренується на випадковій підвибірці з усього набору даних, а також використовує лише випадковий набір ознак для пошуку найкращих розгалужень. У задачах класифікації, коли потрібно віднести новий потік трафіку до певного класу, кожне дерево в "лісі" "голосує" за свій варіант, і остаточним рішенням стає той клас, який отримав більшість голосів. Завдяки такому механізму

усереднення, Випадковий Ліс є надзвичайно популярним та ефективним методом для класифікації мережевого трафіку. Його ключові переваги включають високу стійкість до перенавчання, відносну простоту використання, оскільки він не вимагає складного налаштування гіперпараметрів, і, що є особливо цінним для аналітичних завдань, здатність надавати оцінку "важливості ознак" [14]. Ця можливість дозволяє досліднику зрозуміти та кількісно оцінити, які саме характеристики трафіку, наприклад, "розмір пакета" чи "час між пакетами", мали найбільший вплив на фінальне рішення моделі. Схему Випадкового лісу можна побачити нижче на (Рисунку 2.3.).

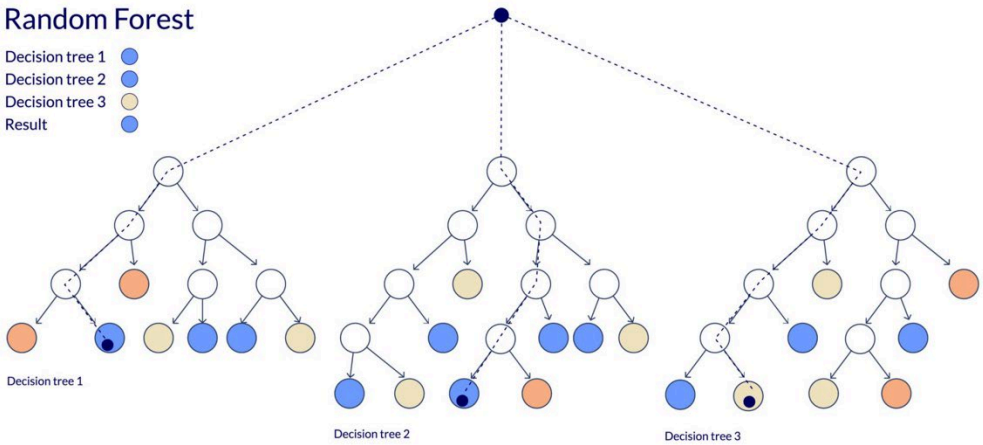

Рис. 2.3. Схема системи Випадкового лісу

2.2.4. Методи прогнозування: Рекурентні Нейронні Мережі (LSTM)

Задачі класифікації, що полягають у визначенні типу трафіку, та задачі прогнозування, спрямовані на передбачення майбутнього навантаження, вимагають застосування різних методологічних підходів. Якщо для вирішення завдань класифікації окремий потік трафіку часто можна описати статичним набором ознак, то для ефективного прогнозування критично важливою є часова послідовність даних та залежності всередині неї. Саме для роботи з послідовностями були розроблені Рекурентні Нейронні Мережі (RNN). На відміну від архітектур прямого поширення, RNN володіє внутрішнім станом, або "пам'яттю", що реалізується шляхом подання вихідного сигналу мережі з часового кроку $t-1$ на вхід мережі на наступному кроці

t. Це дозволяє моделі враховувати попередню інформацію при обробці поточного стану. Проте, класичні RNN-архітектури страждають від фундаментальної проблеми "зникаючого" або "вибухового" градієнта. Цей ефект означає, що під час навчання на довгих послідовностях, градієнти помилки, що поширюються у зворотному напрямку, мають тенденцію або експоненційно зменшуватися до нуля, або зростати, що унеможливлює ефективне навчання моделі та призводить до того, що мережа "забуває" інформацію, яка була на самому початку послідовності.

Для вирішення цієї проблеми були запропоновані моделі Довгої Короткострокової Пам'яті (Long Short-Term Memory, LSTM). LSTM є особливим, більш складним типом рекурентної нейронної мережі, що має унікальну внутрішню структуру, яка складається з так званих "вентилів". Основою комірки LSTM є "стан комірки" (cell state), який функціонує як "конвеєрна стрічка", що несе пам'ять вздовж усієї послідовності. Цей стан регулюється спеціальними вентильними механізмами, які вибірково керують тим, яку інформацію слід додавати до цього стану, а яку - видаляти. Зокрема, Вентиль забування (Forget Gate) приймає рішення про те, яку інформацію з попереднього стану комірки слід "забути". Вентиль входу (Input Gate) визначає, яка нова інформація з поточного вхідного сигналу має бути збережена у стані комірки. Нарешті, Вентиль виходу (Output Gate) вирішує, яка частина поточного стану комірки буде використана для формування вихідного сигналу на даному часовому кроці. Завдяки цій складній архітектурі, LSTM здатні ефективно вивчати та моделювати довгострокові залежності в даних. Це робить їх ідеальним інструментом для вирішення завдань прогнозування навантаження трафіку (traffic forecasting). Модель LSTM, навчена на часових рядах мережевого трафіку, зібраних, наприклад, за минулий тиждень, здатна з високою точністю передбачити навантаження на мережу в наступні години, оскільки вона може захопити та екстраполювати складні добові та тижневі цикли активності користувачів [11].

2.3. Методи виявлення аномалій і кіберзагроз у мобільних мережах

Виявлення аномалій та кібернетичних загроз являє собою одну з найбільш складних задач у сфері аналізу мережевого трафіку. Ця складність зумовлена тим, що зловмисники постійно еволюціонують у своїх методах, а нові атаки, відомі як "атаки

нульового дня" (zero-day attacks), за своїм визначенням не мають відомих сигнатур для їх детектування. У той час як для завдань класифікації, що розглядалися у Розділі 2.2, зазвичай застосовуються підходи керованого навчання (supervised learning), ефективне виявлення аномалій вимагає принципово іншого підходу, а саме - навчання без учителя або напів-керованого навчання. Фундаментальна ідея, що лежить в основі цих підходів, полягає у навчанні моделі на еталонному наборі даних, що репрезентує виключно "нормальну" поведінку системи. Відповідно, будь-яка мережева активність або потік трафіку, що демонструє суттєві відхилення від цієї вивченої "норми", автоматично ідентифікується та позначається як потенційна аномалія. Встановлено, що саме для вирішення такого класу завдань ідеально підходять нейромережеві архітектури, відомі як Автоенкодери (Autoencoders).

2.3.1. Теоретичні основи: Автоенкодери (Autoencoders)

Автоенкодер (Autoencoder, AE) являє собою специфічну архітектуру нейронної мережі, яка належить до класу алгоритмів навчання без учителя. Його фундаментальна мета полягає не у вирішенні завдань класифікації, а в навчанні моделі ефективно стискати вхідні дані та згодом відновлювати їх із цього стисненого представлення. Архітектурно, будь-який автоенкодер складається з двох основних компонентів, що виконують симетричні, але протилежні функції. Першим компонентом є Енкодер (кодер), який приймає вхідні дані, наприклад, вектор ознак, що описує мережевий трафік, і послідовно стискає їх шляхом нелінійних перетворень у представлення значно меншої розмірності. Це компактне стиснене представлення отримало назву "прихований простір" (latent space) або, через свою форму, "пляшкове горло"ботлнек. Другим компонентом є Декодер (декодер), який отримує на вхід стиснене представлення з "пляшкового горла" і виконує зворотне перетворення, намагаючись реконструювати початкові вхідні дані з якомога меншими втратами. Процес навчання такої моделі є самокерованим: модель навчається на вхідних даних X таким чином, щоб її вихід X' був максимально схожим на X. Функція втрат, що оптимізується під час навчання, по суті, являє собою "помилку відновлення" (reconstruction error), яка часто обчислюється як середньоквадратична помилка (MSE)

між X та X'. Завдяки наявності вузького "пляшкового горла", мережа під час навчання змушена ігнорувати стохастичний "шум" та виділяти лише найважливіші, фундаментальні закономірності (патерни), присутні в даних, оскільки тільки ця суттєва інформація може бути ефективно пропущена через шар стиснення і потім успішно використана для відновлення вихідного сигналу [22].

2.3.2. Застосування Автоенкодерів для виявлення аномалій

Ефективність застосування автоенкодерів для виявлення аномалій, що є центральним елементом пропонованої наукової новизни, ґрунтується на специфічному двохетапному процесі навчання та експлуатації. На першому етапі, етапі навчання (Training), модель автоенкодера тренується виключно на масиві даних, що репрезентує "нормальний", легітимний трафік. Це фундаментальне припущення методології. Наприклад, для навчання використовуються мільйони сесій, що відповідають еталонній поведінці IoT-пристроїв, як-от передача малих пакетів з періодичними виходами в ефір. Модель змушують навчитися ідеально стискати ці патерни у приховане представлення та згодом точно їх відновлювати, що перетворює її на "експерта" виключно з нормальної активності, це відбувається завдяки функції (2.4) втрат Автоенкодера впізнаваної як середньоквадратична помилка.

$$L_{MSE} = \frac{1}{n} \sum_{i=1}^{n} (x_i - x_i')^2, \tag{2.4}$$

Де, x_i - вихідний вектор, а x_i' - відновлений вектор.

На другому етапі, етапі виявлення (Inference), навчена модель починає обробляти новий, реальний трафік, який вона раніше не бачила. Якщо цей новий потік трафіку за своїми характеристиками відповідає "нормальному", автоенкодер, як "експерт", успішно його відновить; "помилка відновлення" (reconstruction error), тобто різниця між вхідним вектором X та відновленим X', буде очікувано низькою. Однак, якщо на вхід надходить "аномалія", наприклад, потік, згенерований DDoS-атакою з того ж IoT-пристрою, що має "вибуховий" характер, схожий на відео, модель

"не впізнає" цей патерн. Оскільки вона ніколи не стикалася з такими даними під час навчання, вона виявиться нездатною коректно реконструювати цей аномальний потік зі свого стисненого латентного простору. Це неминуче призведе до генерації високої "помилки відновлення". Таким чином, аналітичний висновок ґрунтується на встановленні певного порогового значення (threshold) для цієї помилки відновлення. Будь-який потік трафіку, чия індивідуальна помилка відновлення перевищує цей встановлений поріг, миттєво маркується як "аномалія" і скеровується на подальший, більш глибокий аналіз. Такий підхід, що базується на навчанні без учителя, дозволяє ефективно виявляти раніше невідомі атаки "нульового дня" та будь-яку іншу нетипову поведінку, для якої не існує попередньо визначених сигнатур, що є критично важливим для забезпечення безпеки сучасних мереж, зокрема 5G [12].

2.4. Критерії оцінювання ефективності аналізу трафіку

Для об'єктивного порівняння різноманітних моделей аналізу трафіку та валідної оцінки їхньої продуктивності використовуються стандартизовані математичні метрики. Особливої важливості це набуває у вирішенні завдань класифікації та виявлення аномалій, де кінцевий результат не завжди є очевидним або легко інтерпретованим. Застосування таких метрик є фундаментальною вимогою в сучасних телекомунікаційних дослідженнях, оскільки вони формують єдиний, об'єктивний базис для порівняльного аналізу. Це є особливо актуальним в умовах високо гетерогенних мереж, таких як 5G та Інтернет Речей (IoT), де експоненційне зростання обсягів та складності трафіку унеможливлює проведення ручного аналізу чи застосування суб'єктивних оцінок. У галузі кібернетичної безпеки ці метрики є незамінним інструментарієм. Вони використовуються для здійснення ретельної оцінки ефективності систем виявлення вторгнень (IDS), моделей, призначених для протидії DDoS-атакам, та алгоритмів, що ідентифікують приховані канали передачі даних або поширення шкідливого програмного забезпечення. Саме в цьому контексті стандартизована оцінка дозволяє чітко кількісно визначити, наскільки добре модель здатна диференціювати легітимну мережеву активність від зловмисної, таким чином

мінімізуючи як пропущені загрози, так і помилкові спрацювання. Окрім того, зазначені метрики широко застосовуються в управлінні продуктивністю мережі (Network Performance Management). Вони сприяють оцінці точності моделей, що прогнозують виникнення заторів, або алгоритмів, які класифікують трафік за відповідним типом (наприклад, потокове відео, VoIP, веб-серфінг) з метою подальшого застосування диференційованих правил якості обслуговування (QoS). Таким чином, інженерний персонал отримує можливість об'єктивно визначити, який саме аналітичний підхід забезпечує найбільш надійні та достовірні результати для оптимізації процесів розподілу мережевих ресурсів. Отже, ці математичні інструменти функціонують як спільна мова для дослідницької спільноти та інженерів-практиків, дозволяючи не просто розробляти нові моделі, але й доказово демонструвати їхню перевагу та практичну придатність для експлуатації у складних, динамічних мережевих середовищах.

2.4.1. Основа оцінки: Матриця Плутанини (Confusion Matrix)

Для завдань бінарної класифікації, таких як диференціація трафіку на "Аномалію" та "Норму", використовується матриця помилок формату 2x2, яка вводить чотири ключові поняття для оцінки продуктивності моделі. По-перше, True Positive (TP), або Істинно Позитивний результат, фіксується тоді, коли модель коректно прогнозує клас "Аномалія", і вхідні дані справді є аномалією; це являє собою ідеальний результат виявлення загрози, ілюстрація розподілу базових термінів у Матриці плутанини зображена ничже на (Рисунку 2.4.).

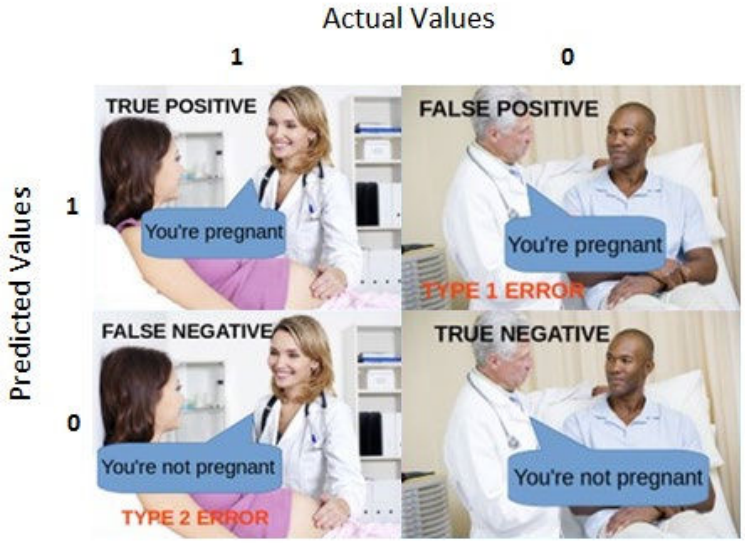

Рис. 2.4. Ілюстрація основних термінів матриці плутанини

По-друге, True Negative (TN), або Істинно Негативний результат, реєструється, коли модель правильно прогнозує клас "Норма", і трафік дійсно є нормальним, що також є бажаним коректним результатом. По-третє, False Positive (FP), або Хибно Позитивний результат, відомий також як Помилка I роду, виникає у випадку, коли модель прогнозує "Аномалію", але насправді трафік є "Нормальним"; така ситуація являє собою "фальшиву тривогу", яка може призвести до небажаних наслідків, наприклад, блокування легітимних користувачів. По-четверте, False Negative (FN), або Хибно Негативний результат, що відповідає Помилці II роду, трапляється, коли модель помилково прогнозує "Норму" для даних, які насправді є "Аномалією"; цей сценарій "пропущеної атаки" вважається найнебезпечнішим типом помилки в контексті систем безпеки [7].

2.4.2. Ключові метрики класифікації

Першою та найпростішою метрикою є Accuracy або ж точність, що обчислюється за формулою (2.5).

$$(TP + TN) / (TP + TN + FP + FN), \qquad (2.5)$$

По суті, вона відображає відсоток прогнозів, які модель зробила правильно, враховуючи як істинно позитивні, так і істинно негативні випадки, від загальної кількості всіх прогнозів. Однак, ця метрика може вводити в оману при роботі з незбалансованими даними, що є типовим для задач виявлення аномалій. Наприклад, якщо 99.9% трафіку є "нормальним" і лише 0.1% - "аномальним", модель, яка завжди прогнозує "Норму", матиме 99.9% точності, але при цьому буде абсолютно марною, оскільки вона пропускає абсолютно всі атаки. Через таку недієвість точності в задачах, пов'язаних з безпекою, необхідно використовувати більш специфічні та інформативні метрики.

Однією з таких метрик є Precision (Точність позитивного прогнозу), що розраховується за формулою (2.6).

$$P = \frac{TP}{TP + FP}, \tag{2.6}$$

Ця метрика відповідає на практичне питання: "З усіх потоків, які модель позначила як 'Аномалія', яка частка справді була аномаліями?". Головна мета при оптимізації Precision полягає у мінімізації кількості Хибно Позитивних спрацювань (False Positives, FP). Високий показник Precision є особливо важливим у сценаріях, де ціна "фальшивої тривоги" є надзвичайно високою, наприклад, при ризику блокування важливого легітимного клієнта.

Іншою критично важливою метрикою є Recall (Повнота), що визначається формулою (2.7).

$$R = \frac{TP}{TP + FN}, \tag{2.7}$$

Recall відповідає на фундаментальне для безпеки питання: "З усіх реальних аномалій, які були присутні в трафіку, яку частку ми змогли фактично знайти та ідентифікувати?". Метою максимізації Recall є мінімізація кількості Хибно Негативних спрацювань (False Negatives, FN). Високий показник Recall є

найважливішою метрикою для сфери кібербезпеки, оскільки ціна "пропущеної атаки" (FN) може бути катастрофічною для всієї системи.

Нарешті, для отримання збалансованої оцінки використовується F1-Score, що обчислюється за формулою (2.8).

$$F1 = 2 \times \frac{P \times R}{P + R}, \tag{2.8}$$

За своєю суттю, це є гармонійне середнє між Precision та Recall. F1-Score функціонує як єдина інтегральна метрика, яка дозволяє комплексно оцінити баланс між цими двома показниками. F1-Score вважається найкращою метрикою для оцінки моделей на незбалансованих наборах даних, оскільки її значення буде низьким, якщо або Precision, або Recall має низьке значення, вимагаючи від моделі одночасної ефективності в обох аспектах [3].

ВИСНОВКИ ДО РОЗДІЛУ 2

У даному розділі було закладено повне теоретичне та математичне підґрунтя, необхідне для розробки системи інтелектуального аналізу трафіку. Було проведено всебічний аналіз класичних математичних моделей, що традиційно використовуються для опису трафіку, включно з Пуассонівською, Самоподібною та Марковською, та детально обґрунтовано їхні іманентні обмеження в сучасних умовах. На додаток, було розглянуто "класичні" методи машинного навчання, такі як K-NN, SVM та Random Forest, у контексті їх застосування для вирішення завдань класифікації, а також проаналізовано рекурентні моделі, зокрема LSTM, для задач прогнозування часових рядів. Особливу увагу було приділено детальному опису теоретичної архітектури та фундаментальних принципів роботи Автоенкодерів, які позиціонуються як основний інструмент для виявлення аномалій, що становить ядро наукової новизни даної роботи. На завершення, було визначено та обґрунтовано вибір ключових метрик ефективності, таких як Матриця помилок, Precision, Recall та F1-

Score, які є необхідним інструментарієм для об'єктивної оцінки та валідації розробленої моделі, особливо в умовах незбалансованих даних. Таким чином, результати, отримані в цьому розділі, формують повний теоретичний апарат, який буде безпосередньо використаний у наступному розділі для обґрунтування практичної розробки архітектури та реалізації програмного прототипу системи.

РОЗДІЛ 3
РОЗРОБКА СИСТЕМИ ІНТЕЛЕКТУАЛЬНОГО АНАЛІЗУ ТРАФІКУ

3.1. Постановка задачі та вимоги до системи

У попередніх розділах було проведено аналіз сучасного стану мобільних мереж, у ході якого було обґрунтовано неефективність традиційних методів аналізу трафіку. Крім того, було закладено теоретичні основи для застосування методів машинного навчання у цій галузі. Поточний розділ, базуючись на цих теоретичних засадах, присвячений безпосередній розробці програмного прототипу системи, призначеної для здійснення інтелектуального аналізу мережевого трафіку.

3.1.1. Постановка задачі

Базуючись на раніше визначеній меті роботи та сформульованій науковій новизні, головна задача розробки полягає у створенні програмного прототипу. Ця система має бути здатною функціонувати в умовах, наближених до реальних, та виконувати аналіз мережевого трафіку, з особливим акцентом на шифрованих потоках у мобільних мережах. Перед прототипом ставиться вирішення двох ключових завдань.

Перше завдання - це класифікація трафіку, що полягає у визначенні типу програми або сервісу (наприклад, Відео, Ігри, або IoT), який згенерував конкретний потік; ця ідентифікація має ґрунтуватися на аналізі унікальних статистичних "відбитків" потоку.

Друге, і не менш важливе, завдання - це виявлення аномалій. Ця функція передбачає ідентифікацію нетипових патернів трафіку, таких як DDoS-атаки, що походять з IoT-пристроїв, або атаки "нульового дня", котрі за визначенням не були представлені у початковій навчальній вибірці. Для реалізації цієї специфічної задачі буде використано нейронну мережу типу автоенкодер.

3.1.2. Вимоги до системи

Для успішного вирішення поставленої задачі, до програмного прототипу висувається низка чітких функціональних та нефункціональних вимог. На функціональному рівні, система повинна володіти здатністю зчитувати та розбирати (парсити) дані мережевого трафіку, що надходять із файлів стандартизованого формату .pcap, або отримувати дані безпосередньо з мережевого інтерфейсу. Далі, прототип має забезпечувати автоматичне виділення ознак (Feature Extraction), що включає обробку "сирих" пакетів, їх агрегацію у потоки (flows) та розрахунок для кожного потоку вектора ознак (наприклад, тривалість, кількість пакетів, середній розмір, час між пакетами), як це було описано в попередніх розділах. Система повинна містити навчений модуль класифікації, такий як Random Forest (2.2.3), для виконання задачі багатокласової ідентифікації потоків. Критично важливою вимогою є наявність модуля виявлення аномалій, що реалізований на базі навченої моделі Автоенкодера (2.3.1), яка була навчена виключно на "нормальному" трафіку. Завершальною функціональною вимогою є здатність системи надавати результат аналізу у зрозумілому вигляді, зокрема, виводити мітку "Аномалія" та відповідне значення "помилки відновлення" для ідентифікованих аномальних потоків.

Для реалізації прототипу буде використана мова програмування Python, для реалізації мають бути використані стандартизовані бібліотеки з відкритим кодом, зокрема Pandas та NumPy для маніпуляцій з даними, Scapy або dpkt для парсингу .pcap файлів, Scikit-learn для імплементації класичних моделей та розрахунку метрик (Precision, Recall), а також TensorFlow або Keras для побудови та навчання моделі Автоенкодера. З точки зору ефективності , система повинна демонструвати високі показники метрик F1-Score та Recall у задачі виявлення аномалій, оскільки пріоритетом є мінімізація кількості пропущених атак (False Negatives). Архітектура коду повинна бути модульною, щоб забезпечити можливість легкої заміни або оновлення окремих компонентів, наприклад, моделі класифікатора, у майбутньому.

3.2. Архітектура системи інтелектуального аналізу трафіку

Виходячи з вимог, визначених у підрозділі 3.1, була розроблена модульна, конвеєрна архітектура для програмного прототипу. Ця архітектура складається з п'яти послідовних логічних блоків, кожен з яких реалізує окрему функціональну вимогу. Такий модульний підхід є принциповим, оскільки він дозволяє незалежно розробляти, тестувати та в майбутньому легко замінювати будь-який компонент системи, наприклад, модель класифікатора, не порушуючи при цьому загальної логіки роботи.

Першим компонентом архітектури є Модуль збору даних (Data Collector), завданням якого є зчитування "сирих" мережевих даних. У рамках реалізації, цей модуль відповідає за читання файлів у форматі .pcap, використовуючи для цього бібліотеки, такі як Scapy або dpkt. Ці інструменти дозволяють здійснювати парсинг кожного окремого пакета, вилучаючи з нього необхідні заголовки, зокрема IP та TCP/UDP, а також ключові метадані, як-от час прибуття та розмір пакета.

У роботі запропоновано та обґрунтовано архітектурний підхід, що базується на методології виявлення аномалій, який демонструє суттєві переваги порівняно з традиційними однорівневими класифікаторами сигнатурного типу. Фундаментальною відмінністю розробленої системи є її здатність до ефективної протидії атакам "нульового дня". На відміну від класичних рішень, які обмежені детекцією лише заздалегідь відомих загроз, запропонована модель на базі автоенкодера формує еталонний профіль нормального трафіку та реагує на будь-які статистичні девіації, що дозволяє ідентифікувати навіть новітні, раніше невідомі вектори атак. Крім того, важливим експлуатаційним аспектом є незалежність системи від трудомісткого процесу розмітки даних шкідливого ПЗ та постійного оновлення баз сигнатур. Запропонована модель потребує лише актуалізації профілю нормальної поведінки мережі, що значно спрощує її підтримку в умовах динамічного середовища 5G та Інтернету речей. З архітектурної точки зору, такий підхід забезпечує оптимізацію обчислювальних ресурсів, оскільки автоенкодер виконує роль первинного фільтра ("фільтра чистоти"), який відсіює легітимний трафік і скеровує на поглиблений, ресурсоємний аналіз виключно підозрілі потоки даних.

Наступним у конвеєрі йде Модуль агрегації та виділення ознак (Flow Feature Extractor). Його функціональна задача полягає у перетворенні набору окремих пакетів на агреговані потоки та розрахунку їхніх унікальних статистичних "відбитків". Цей блок є центральним елементом підготовки даних, групуючи пакети в потоки за унікальним 5-компонентним ключем (IP-джерела/призначення, Порт-джерела/призначення, Протокол). Після завершення потоку, наприклад, за тайм-аутом, модуль розраховує для нього комплексний вектор ознак, що може включати до 40-50 параметрів. До них належать тривалість потоку, загальна кількість пакетів та байтів в обох напрямках (вперед/назад), а також статистичні показники (мінімум, максимум, середнє значення та стандартне відхилення) для розмірів пакетів та часових інтервалів між ними. На виході цей блок генерує уніфікований вектор ознак, представлений, наприклад, у вигляді рядка Pandas DataFrame, який слугує стандартизованим входом для всіх наступних моделей машинного навчання.

Третім компонентом є Модуль виявлення аномалій, побудований на базі Автоенкодера. Його завданням є оцінка "нормальності" вхідного потоку. Сформований на попередньому етапі вектор ознак подається на вхід навченому Автоенкодеру, реалізованому на платформі TensorFlow/Keras. Модель намагається відновити цей вектор, після чого розраховується "помилка відновлення" (Reconstruction Error). Ця помилка порівнюється з попередньо встановленим пороговим значенням (Threshold). Якщо помилка перевищує поріг, потік маркується як "Аномалія"; в іншому випадку, якщо помилка є меншою або дорівнює порогу, потік класифікується як "Норма".

Паралельно з попереднім блоком працює Модуль класифікації (Classifier), що реалізує завдання визначення конкретного типу трафіку. Той самий вектор ознак, що був згенерований Блоком 2, подається на вхід навченій "класичній" моделі, наприклад, Випадковому Лісу (Random Forest), реалізованій за допомогою бібліотеки Scikit-learn. Ця модель видає дискретну мітку класу, наприклад, "Відео", "IoT" або "Гра".

Завершальним елементом архітектури є Модуль прийняття рішень (Decision Logic), який відповідає за формування фінального вердикту системи. Цей логічний

блок агрегує виходи з Блоків 3 та 4 для надання комплексного результату. Наприклад, якщо Блок 3 видає маркер "Норма", а Блок 4 ідентифікує потік як "Відео", фінальний висновок буде "Нормальний потік, тип: Відео". Альтернативно, якщо Блок 3 виявляє високу помилку відновлення і маркує потік як "Аномалія", в той час як Блок 4 класифікує його як "IoT", система згенерує тривожне сповіщення, наприклад: "УВАГА: АНОМАЛІЯ! (Потік, схожий на IoT, але поводиться нетипово. Можлива DDoS-атака)", візуалізацію схеми можна побачити нижче на (Рисунку 3.1.).

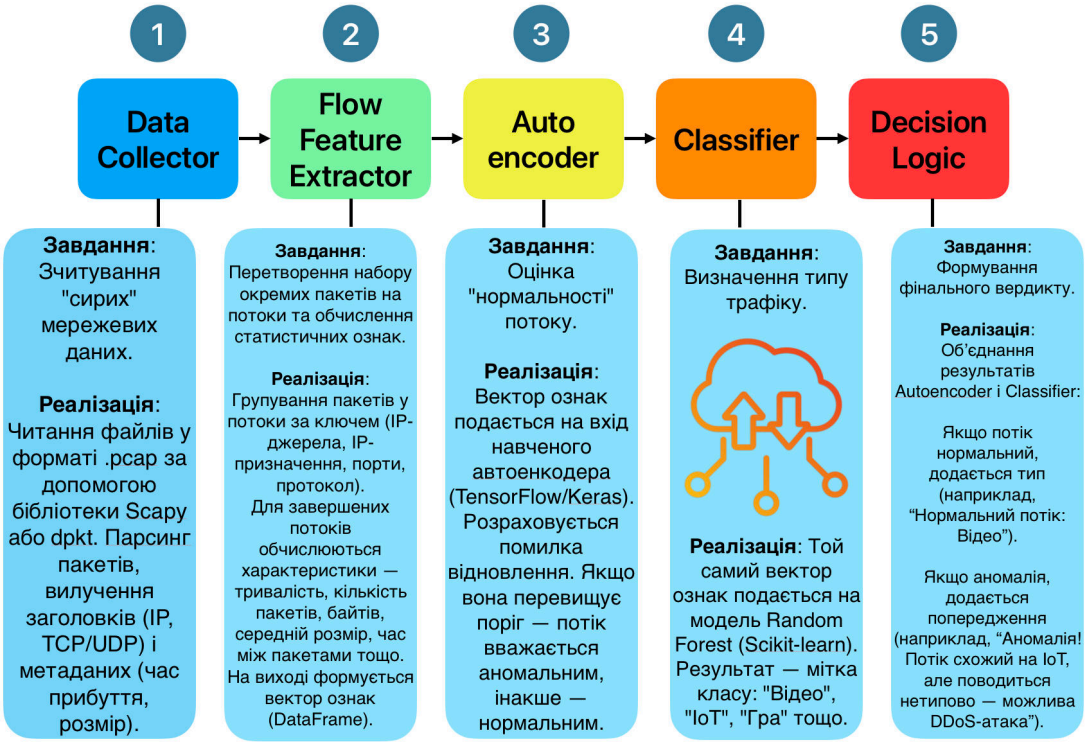

Рис. 3.1. Архітектура системи інтелектуального аналізу трафіку

3.3. Алгоритми збору, попередньої обробки та класифікації трафіку

Даний підрозділ присвячено детальному розгляду практичної реалізації конвеєра обробки даних (data pipeline), що охоплює повний цикл, починаючи від початкового етапу зчитування "сирих" мережевих пакетів і завершуючи отриманням фінальної мітки класу для кожного потоку.

3.3.1. Збір та парсинг пакетів (Блок 1)

Початковим етапом процесу є отримання "сирих" даних, які в лабораторних умовах, як правило, представлені у вигляді файлів формату .pcap, отриманих за допомогою мережевих сніферів, таких як Wireshark або tcpdump. Для здійснення читання та парсингу цих файлів у рамках програмного прототипу, реалізованого мовою Python, використовується спеціалізована бібліотека Scapy. Цей інструментарій надає функціонал, що дозволяє ітеративно обробляти кожен окремий пакет у .pcap файлі та програмно отримувати доступ до будь-якого поля його заголовків. Алгоритмічно процес збору починається із завантаження прототипом цільового .pcap файлу, що можна побачити нижче на (Рисунку 3.2.).

```
def load_data(file_path):
    """
    Loading files from CSV-file and returning Pandas DataFrame.
    """
    print(f"Loading files from {file_path}...")
    try:
        data = pd.read_csv(file_path, header=None, names=COLUMN_NAMES)
        print("Data successfully uploaded.")
        return data
    except FileNotFoundError:
        print(f"Error: File {file_path} was not found.")
        return None
```

Рис. 3.2. Функція завантаження даних

За допомогою Scapy кожен пакет послідовно деконструюється, і в ході цього процесу для нього витягуються ключові метадані. До цих метаданих належать: час прибуття (Timestamp) з високою точністю до мікросекунд, загальний розмір пакета (Packet Size) в байтах, а також унікальний ідентифікатор потоку (Flow ID). Цей ідентифікатор формується як кортеж із п'яти елементів, що включає IP-адресу джерела, порт джерела, IP-адресу призначення, порт призначення та ідентифікатор протоколу. Крім того, у разі застосовності, витягується інформація про наявність специфічних прапорів TCP, зокрема SYN, FIN та RST. Увесь цей набір витягнутих

метаданих для кожного пакета далі передається до Блоку 2, призначеного для виконання агрегації [21].

3.3.2. Агрегація у потоки та виділення ознак (Блок 2)

Моделі машинного навчання не оперують на рівні окремих пакетів, оскільки один пакет не несе достатнього обсягу інформації для аналізу. Натомість, модель повинна аналізувати поведінкові характеристики потоку (flow) в цілому. Потік визначається як двонаправлена послідовність пакетів між двома кінцевими точками, що однозначно ідентифікується унікальним 5-компонентним ключем (Flow ID) та обмежена в часі, наприклад, тайм-аутом неактивності у 60 секунд. Алгоритм виділення ознак (Feature Extraction) передбачає, що система зберігає стан активних потоків у структурі даних типу "словник" (Python dictionary). Коли надходить новий пакет, він ідентифікується, і його метадані додаються до сукупної статистики відповідного потоку. У момент, коли потік вважається завершеним, що може бути визначено за прапором TCP FIN або за спрацюванням тайм-ауту неактивності, модуль розраховує для цього потоку фінальний вектор ознак. Як було зазначено в теоретичних оглядах, цей вектор повинен містити набір статистичних характеристик, які разом створюють унікальний "відбиток" потоку. Для розробленого прототипу був обраний набір з N ознак (наприклад, 40), що включає декілька категорій.

До загальних ознак належать: загальна тривалість потоку, загальна кількість пакетів у прямому та зворотному напрямках, а також загальна кількість байт у прямому та зворотному напрямках. Друга категорія охоплює ознаки розміру пакетів, такі як середнє значення та стандартне відхилення розміру пакета у прямому напрямку, аналогічні показники для зворотного напрямку, а також мінімальний та максимальний розмір пакета у всьому потоці. Третя категорія описує час між пакетами (Inter-Arrival Time, IAT), включаючи середнє та стандартне відхилення часу між пакетами у прямому напрямку, відповідні показники для зворотного напрямку, та загальні мінімальний і максимальний час між пакетами. Перед подачею у модель, ці необроблені ознаки обов'язково проходять етап попередньої обробки, а саме

стандартизацію (Standardization), наприклад, за допомогою StandardScaler з бібліотеки scikit-learn, що можна побачити на (Рисунку 3.3.).

```
        normal_df = normal_df.drop(columns=['label', 'score'])

        normal_df_processed = pd.get_dummies(normal_df)

        print(f"\nData after 'One-Hot Encoding':")
        print(f"   Number of rows BEFORE (after .drop): {len(normal_df.columns)}")
        print(f"   Number of  columns AFTER (from .get_dummies): {len(normal_df_processed.columns)}")

        print("\text disappeared:")
        print(normal_df_processed.head())

#  Масштабування даних (Standardization)

        # 1. Створення "масштабувальника"
        scaler = StandardScaler()

        # 2. "Навчаємо" його: він вивчає середнє значення та
        # стандартне відхилення кожного з 77 стовпців.
        # Ми навчаємо його тільки на 'normal' даних без аномалій
        scaler.fit(normal_df_processed)

        # 3. Масштабування даних.
        normal_data_scaled = scaler.transform(normal_df_processed)

        print("\nData successfully scaled (StandardScaler).")
        print(f"   Final data shape: {normal_data_scaled.shape}")

        print("\nExample of each data set after scaling:")
        print(normal_data_scaled[0])
```

Рис. 3.3. Попередня обробка та масштабування даних

Ця операція приводить всі значення до єдиного масштабу, з середнім значенням 0 та стандартним відхиленням 1, що є критично важливою вимогою для коректної роботи та збіжності нейронних мереж, зокрема, Автоенкодера [19].

3.3.3. Алгоритм класифікації (Блок 4)

У той час як Блок 3, реалізований у вигляді Автоенкодера, несе відповідальність за первинне виявлення аномалій, Блок 4 виконує завдання деталізованої класифікації тих потоків, які на попередньому етапі були ідентифіковані як "нормальні". Для практичної реалізації цього модуля було обрано алгоритм Випадковий Ліс (Random Forest), теоретичні основи якого були розглянуті в підрозділі 2.2.3. Обґрунтування такого вибору ґрунтується на кількох ключових перевагах даного методу. По-перше, Random Forest демонструє стабільно високу точність і є одним із найефективніших "класичних" алгоритмів, що часто показує результати, співмірні з продуктивністю

нейронних мереж, особливо при роботі з табличними даними, якими по суті і є згенеровані вектори ознак. По-друге, на відміну від нейронних мереж, які здебільшого функціонують як "чорна скринька", Random Forest надає можливість інтерпретації завдяки механізму "важливості ознак" як показано на (Рисунку 3.4.).

```
# Створення вхідного шару (Input Layer)

#  "вхідні двері"  моделі.
input_layer = keras.layers.Input(shape=(INPUT_DIM,), name="Encoder_Input")

print("   Input layer was created (Input layer).")

# Створення шарів Енкодера (Encoder)
# Перший прихований шар (стискаємо 77 -> 64)
encoder = keras.layers.Dense(64, activation='relu', name="Encoder_Hidden_1")(input_layer)

# "Пляшкове горло" (стискаємо 64 -> 32)
encoder_output = keras.layers.Dense(BOTTLENECK_DIM, activation='relu', name="Bottleneck_Output")(encoder)
print("   Encoder layers created (77 -> 64 -> 32).")
```

Рис. 3.4. Структура Енкодера

Це надає безцінну інформацію для дослідника, оскільки дозволяє кількісно оцінити, які саме статистичні характеристики, наприкла, середній час між пакетами чи стандартне відхилення розміру пакета, зробили найбільший внесок у процес розрізнення одного класу трафіку, як-от "Відео", від іншого, наприклад, "Ігор". По-третє, алгоритм вирізняється високою стійкістю: він є відносно нечутливим до різного масштабу вхідних ознак, хоча у прототипі вони все одно стандартизуються для узгодженості з Автоенкодером, і добре справляється з незбалансованими класами. Процес навчання цієї моделі відбувається на попередньо розміченому наборі даних, який, що принципово важливо, містить виключно "нормальний" трафік; усі аномальні екземпляри з нього виключені. Таким чином, модель Random Forest навчається диференціювати легітимні класи, такі як "Відео", "IoT", "Гра", "Веб" тощо, базуючись виключно на їхніх унікальних статистичних "відбитках".

3.4. Реалізація програмного прототипу системи

Даний підрозділ зосереджений на описі безпосередньої програмної реалізації ключових компонентів розробленої системи. Особлива увага приділяється детальній

імплементації Модуля виявлення аномалій, що відповідає Блоку 3 запропонованої архітектури, а також Модуля прийняття рішень, який становить Блок 5.

3.4.1. Архітектура Автоенкодера та її програмна реалізація

Для експериментальної перевірки ефективності розробленого прототипу було використано загальновизнаний у науковій спільноті набір даних KDD Cup 1999, зокрема його модифікації KDDTrain+ та KDDTest+. Цей датасет вважається стандартом де-факто для тестування систем виявлення вторгнень, оскільки містить репрезентативну вибірку як легітимних мережевих з'єднань, так і широкого спектра атак, включаючи відмови в обслуговуванні (DoS), сканування портів та спроби несанкціонованого отримання прав доступу (U2R, R2L). Підготовка даних розпочалася з розділення класів, де критично важливою умовою було формування навчальної вибірки виключно з "нормального" трафіку, тоді як тестова вибірка залишалася змішаною для об'єктивної оцінки здатності моделі до детекції аномалій. Оскільки нейронні мережі вимагають числових вхідних даних, категоріальні ознаки, такі як тип протоколу чи сервісу, були трансформовані за допомогою методу One-Hot Encoding, що призвело до розширення вхідного вектора до 77 вимірів. Наступним кроком стала стандартизація всіх ознак за допомогою StandardScaler, що дозволило привести дані до нульового середнього та одиничного стандартного відхилення; при цьому параметри масштабування обчислювалися лише на основі "нормального" трафіку, щоб уникнути витоку інформації про аномалії в процес навчання. Стосовно архітектури Автоенкодера, було імплементовано симетричну модель, яка приймає на вхід сформований вектор із 77 ознак. Мережа послідовно стискає інформацію через проміжний шар із 64 нейронів до "пляшкового горла" розмірністю 32 нейрони, після чого дзеркально відновлює дані через декодер. Процес навчання моделі керувався оптимізатором Adam, а в якості цільової функції для мінімізації було обрано середньоквадратичну помилку (MSE), яка в даній конфігурації слугує безпосередньою мірою точності відновлення легітимних патернів трафіку.

На (Рисунку 3.5.) нижче наведено фрагмент програмного коду, який деталізує архітектуру моделі, охоплюючи компоненти як Енкодера, так і Декодера, при цьому

для компіляції мережі було обрано адаптивний оптимізатор Adam у поєднанні з функцією втрат середньоквадратичної помилки (MSE).

```
        # Перший прихований шар (стискаємо 77 -> 64)
        encoder = keras.layers.Dense(64, activation='relu', name="Encoder_Hidden_1")(input_layer)

        # "Пляшкове горло" (стискаємо 64 -> 32)
        encoder_output = keras.layers.Dense(BOTTLENECK_DIM, activation='relu', name="Bottleneck_Output")(encoder)
        print("   Encoder layers created (77 -> 64 -> 32).")

# Створення шарів Декодера

        #Побудова "дзеркальної" архітектури

        # Перший шар Декодера (розширюємо 32 -> 64)
        # Він "приєднується" до виходу Енкодера
        decoder = keras.layers.Dense(64, activation='relu', name="Decoder_Hidden_1")(encoder_output)
        decoder_output = keras.layers.Dense(INPUT_DIM, activation='linear', name="Decoder_Output")(decoder)

        print("   Decoder layers created (32 -> 64 -> 77).")

        # Створення та компіляція моделі

        #  Збірка моделі
        # Ми кажемо Keras, де початок (вхід) і де кінець (вихід)
        autoencoder_model = keras.Model(inputs=input_layer,
                                        outputs=decoder_output,
                                        name="Autoencoder")

        #  Компіляція моделі (підготовка до начання)
        # 'mean_squared_error' (MSE) - являє собою "лінійку" для вимірювання помилки
        autoencoder_model.compile(optimizer='adam', loss='mean_squared_error')
```

Рис. 3.5. Створення та компіляція повної моделі

3.4.2. Навчання та прийняття рішень

Процес навчання моделі автоенкодера був побудований на використанні виключно легітимного трафіку, для чого з наявного набору даних було відібрано масив із 67 343 нормальних з'єднань. Навчальний цикл тривав протягом 10 епох, за результатами яких модель продемонструвала стабільну динаміку збіжності: значення функції втрат на валідаційній вибірці, яке склало 0.1765, послідовно корелювало зі зменшенням втрат на навчальній вибірці до рівня 0.0780. Така поведінка кривих навчання свідчить про ефективне узагальнення ознак та відсутність ефекту перенавчання, що також підтверджується відповідними графічними залежностями. Наступним етапом реалізації стало налаштування модуля прийняття рішень шляхом емпіричного визначення критичного порогового значення для детекції аномалій. З цією метою було розраховано помилку відновлення для всього масиву нормальних з'єднань, і як граничну межу було обрано 95-й перцентиль статистичного розподілу

цих помилок, що чисельно дорівнює приблизно $Threshold \approx 0.1791$, це ілюстровано на (Рисунку 3.6.).

```
print("...Model trained successfully!")

# Errors print-out
print("\nModel training results:")
print(f"   Final Training Loss: {history.history['loss'][-1]:.4f}")
print(f"   Final Validation Loss: {history.history['val_loss'][-1]:.4f}")

#  Визначення порогу (Threshold) для виявлення аномалій
print("\nDetermination of the threshold for anomalous detection...")

# 1. Отримуємо "відновлені" дані
reconstructions = autoencoder_model.predict(normal_data_scaled)

# 2. Обчислюємо "помилку відновлення" (MSE) за допомогою NumPy
# Формула: Середнє від квадрату різниці (Mean Squared Error)
# axis=1 означає, що ми рахуємо середнє для кожного рядка окремо
reconstruction_errors_np = np.mean(np.power(normal_data_scaled - reconstructions, 2), axis=1)

# 3. Знаходимо 95-й перцентиль
THRESHOLD = np.percentile(reconstruction_errors_np, 95)

print("...Threshold determined successfully!")
print(f"   Recovery error (95-th percentile): {THRESHOLD:.4f}")

print("\nExample of the first 10 recovery errors (for normal data):")
print(reconstruction_errors_np[:10])
```

Рис. 3.6. Розрахунок помилки відновлення та порогового значення

Згідно з імплементованою логікою, будь-який новий потік трафіку, помилка відновлення якого перевищує цей визначений показник, автоматично класифікується системою як аномалія, що дозволяє забезпечити високий рівень повноти (Recall), визначений як пріоритет для систем кібербезпеки.

На попередньому етапі, використовуючи виключно "нормальні" дані, було розраховано критичне порогове значення. У нашому випадку для цього було обрано 95-й перцентиль розподілу помилок відновлення, що є стратегічним рішенням для забезпечення максимально високого показника повноти (Recall).

ВИСНОВКИ ДО РОЗДІЛУ 3

У межах даного розділу було успішно реалізовано інженерну задачу, пов'язану з проєктуванням та програмною реалізацією прототипу системи для інтелектуального

аналізу трафіку, що становить основу практичної складової магістерського дослідження. На початковому етапі розробки було сформульовано вичерпний перелік функціональних та нефункціональних вимог до системи, а також наведено обґрунтування вибору технологічного стеку, що базується на мові Python та бібліотеках TensorFlow/Keras, із визначенням стандартизованого набору даних KDD Cup для валідації результатів. Основою розробки стала запропонована модульна дворівнева архітектура конвеєрного типу, яка забезпечує логічну послідовність процесів обробки інформації від моменту перехоплення пакетів до етапу прийняття фінальних рішень.

Структурно система була декомпонована на незалежні функціональні модулі, що відповідають за збір даних, автоматизоване виділення характерних ознак, виявлення аномалій із застосуванням автоенкодера та логічний блок прийняття рішень. У ході роботи було деталізовано алгоритмічне забезпечення системи, включаючи методи попередньої підготовки даних, такі як One-Hot Encoding та стандартизація, а також описано принципи роботи класифікатора Random Forest і нейронної мережі автоенкодера. Практичним підсумком розділу стало створення діючого програмного прототипу, в якому реалізовано ключовий модуль на базі Keras для компресії та відновлення векторів ознак, а також імплементовано алгоритм розрахунку порогового значення на основі 95-го перцентиля для ідентифікації аномалій. Таким чином, розроблений прототип досяг стадії повної готовності до проведення експериментальних досліджень.

РОЗДІЛ 4
ЕКСПЕРИМЕНТАЛЬНЕ ДОСЛІДЖЕННЯ РОЗРОБЛЕНОЇ СИСТЕМИ

4.1. Методика проведення досліджень

Методологія експериментального дослідження базувалася на парадигмі навчання без учителя, маючи на меті детекцію аномальних відхилень у структурі мережевого трафіку. Для реалізації експерименту було обрано сценарій, що імітує умови захисту від атак "нульового дня", за яких модель проходить навчання виключно на масиві "нормальних" даних, тоді як тестування здійснюється на вибірці, що містить раніше невідомі системі кіберзагрози.

У якості емпіричної бази для валідації результатів було використано набори даних KDDTrain+ для тренувального етапу та KDDTest+ для етапу тестування, які на сьогодні визнані стандартом для оцінки ефективності систем виявлення вторгнень. Набори даних були додані в теку проєкту як показано нижче на (Рисунку 4.1.).

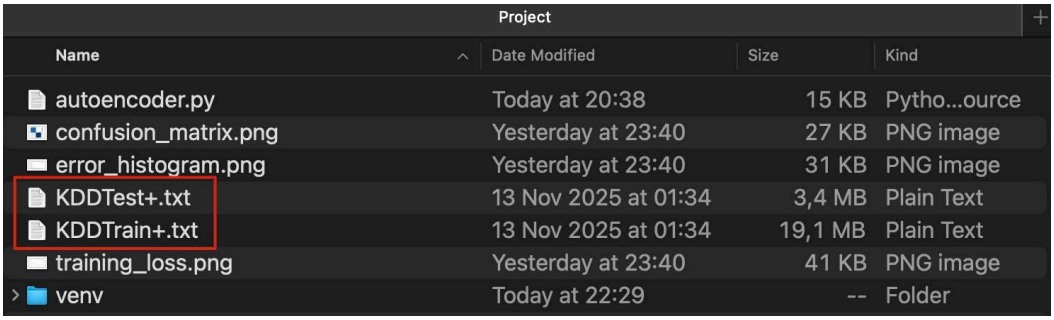

Рис. 4.1. Імпортовані набори даних в теці проєкту

Процедура попередньої підготовки даних включала трансформацію категоріальних ознак за допомогою методу One-Hot Encoding, а також масштабування числових значень із використанням алгоритму StandardScaler, параметри якого розраховувалися виключно на основі легітимного трафіку для уникнення витоку інформації. Визначальним етапом налаштування системи стало встановлення порогового значення для класифікації потоку як "Аномалія", яке, з метою мінімізації

помилкових спрацювань, першочергово було зафіксовано на рівні 99-го перцентиля розподілу помилок відновлення навчального набору, а згодом замінено на 95-ий перцинтиль в ході експериментів із програмою. На (Рисунку 4.2.) нижче зображено, як обробляється проблема різної кількості колонок у тренувальному та тестовому наборах.

```
# One-Hot Encoding
test_df_processed = pd.get_dummies(test_df)

# Тестовий файл може мати менше (або інші) категорії, ніж навчальний.
# Наприклад, у навчанні був протокол "icmp", а в тесті його немає.
# Ми маємо ПРИМУСОВО привести колонки тесту до колонок навчання.
test_df_aligned = test_df_processed.reindex(columns=normal_df_processed.columns, fill_value=0)

# Масштабування (використовуємо ТОЙ САМИЙ scaler, що й для навчання!)
test_data_scaled = scaler.transform(test_df_aligned)
```

Рис. 4.2. Фрагмент коду попередньої обробки та вирівнювання тестового набору даних

На (Рисунку 4.3.), нижче зображений процес розрахунку помилки з подальшим передбаченням та оцінкою результатів проведеного тестування на даних KDDTest+, що є набором даних який був використаний для тестування моделі після навчання на KDDTrain+, оскільки він містить в собі набір аномалій, що і підходить для випробування запропонованого прототипу моделі автоенкодера.

```python
test_data_scaled = scaler.transform(test_df_aligned)

print(f"   Test data processed. Shape: {test_data_scaled.shape}")

# 4. Проганяємо через модель
test_reconstructions = autoencoder_model.predict(test_data_scaled)

# 5. Рахуємо помилки
test_errors = np.mean(np.power(test_data_scaled - test_reconstructions, 2), axis=1)

# 6. Робимо передбачення
y_test_pred = [1 if e > THRESHOLD else 0 for e in test_errors]

# 7. Оцінка результатів (Metrics)
print("\nAnomaly detection results:")
precision = precision_score(y_test_true, y_test_pred)
recall = recall_score(y_test_true, y_test_pred)
f1 = f1_score(y_test_true, y_test_pred)

print(f"   Precision (Точність): {precision:.4f}")
print(f"   Recall (Повнота):    {recall:.4f}")
print(f"   F1-Score:            {f1:.4f}")

#Summary
print(f"\n   We found {sum(y_test_pred)} anomalies with {sum(y_test_true)} real.")
```

Рис. 4.3. Фрагмент коду фіналізації обробки KDDTest+ із подальшим виводом результатів

Для проведення експериментальної валідації розробленого прототипу в якості емпіричної бази було обрано набір даних KDD Cup 1999. Вибір саме цього датасету, попри відсутність у ньому прикладів використання сучасних комунікаційних протоколів, таких як TLS 1.3 або QUIC, а також специфічних сигнатур мереж п'ятого покоління, є методологічно обґрунтованим з низки причин. Насамперед це зумовлено фокусом дослідження на алгоритмічній верифікації запропонованої архітектури, де ключовим завданням є перевірка здатності автоенкодера ідентифікувати статистичні аномалії. Математична природа таких відхилень, що проявляється у різких сплесках активності або нетипових патернах, у наборі KDD Cup структурно корелює з поведінкою аномалій у сучасних мережах, що дозволяє підтвердити коректність роботи алгоритму без прив'язки до змісту корисного навантаження пакетів. Крім того, статус KDD Cup як загальновизнаного академічного еталону забезпечує можливість коректного порівняння ефективності розробленої системи з альтернативними рішеннями в даній предметній області. Разом з тим, необхідно чітко окреслити

обмеження отриманих результатів: вони демонструють принципову ефективність запропонованого методу детектування, проте не свідчать про повну готовність системи до безпосереднього розгортання в інфраструктурі 5G. Повноцінна адаптація до реальних умов експлуатації в середовищі IoT вимагатиме додаткового етапу донавчання моделі на актуальних масивах даних, наприклад CIC-IDS2017 або реальному трафіку мобільного оператора, для врахування специфіки шифрованих заголовків та сучасних поведінкових патернів пристроїв.

4.2. Вибір середовища та інструментів для експериментів

Програмна реалізація прототипу системи та виконання експериментальної частини дослідження здійснювалися з використанням уніфікованого технологічного стеку, базовим інструментом якого виступила мова програмування Python 3. Для безпосередньої побудови архітектури нейронної мережі автоенкодера та проведення процедур її навчання було застосовано можливості спеціалізованих бібліотек глибокого навчання TensorFlow та Keras, бібліотеки були встановлені та імпортовано в теку проєкту у віртуальному серидовищі VENV, що показано на (Рисунку 4.4.).

autoencoder.py > ...

```
1  import pandas as pd
2  import numpy as np
3  import tensorflow as tf
4  import matplotlib.pyplot as plt
5  import seaborn as sns
6  from sklearn.metrics import confusion_matrix
7  from tensorflow import keras
8  from sklearn.model_selection import train_test_split
9  from sklearn.preprocessing import StandardScaler
10 from sklearn.metrics import precision_score, recall_score, f1_score
```

Рис. 4.4. Імпорт бібліотек глибоко навчання TensorFlow та Keras

Задачі, пов'язані з попередньою обробкою масивів даних та виконанням числових операцій, вирішувалися засобами пакетів Pandas та NumPy, тоді як розрахунок метрик ефективності та візуалізація отриманих результатів у вигляді графічних залежностей

забезпечувалися функціоналом бібліотек Scikit-learn, Matplotlib та Seaborn які також були встановлені та імпортовані в проєкт як зображено на (Рисунку 4.5.).

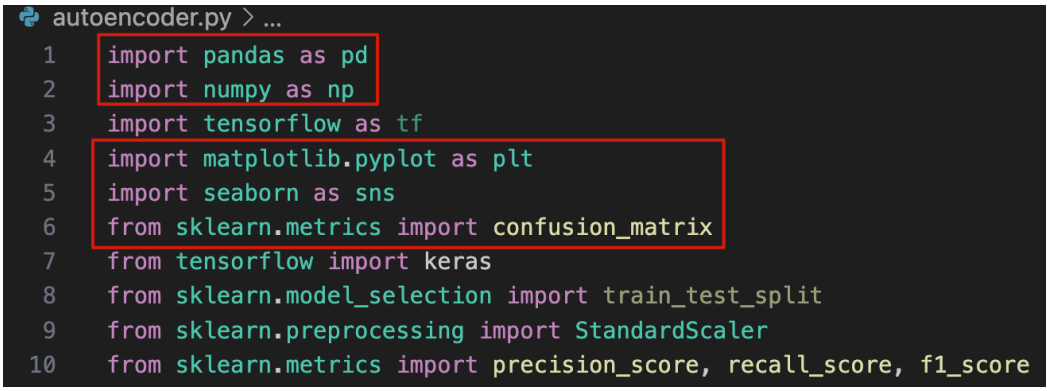

Рис. 4.5. Імпорт Pandas, NumPy, Scikit-learn, Matplotlib та Seaborn

4.3. Результати навчання, аналізу та класифікації трафіку

Розроблений прототип автоенкодера був створений без розробки візуального інтерфейсу для спрощення процесу розгортання моделі, саме тому був обраний варіант візуалізації у CLI – Command Line Interface.

Після ініціалізації програми командою в терміналі: ./venv/bin/python3 autoencoder.py, програма дає користувачу зрозуміти, що відбувається на кожному етапі роботи, завдяки вбудованим текстовим блокам-підказкам. Виділені блоки на ілюстрації (див. рис. 4.6.) були детально описані в підрозділі (3.4.1. Архітектура Автоенкодера та її програмна реалізація).

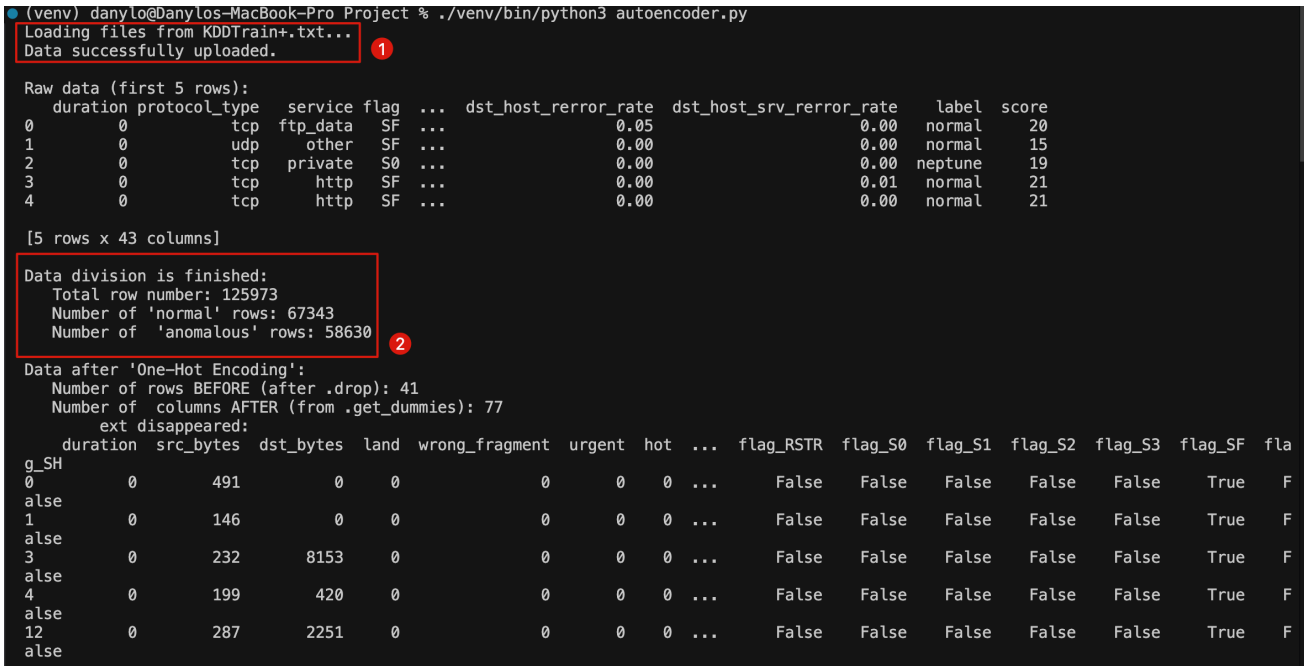

Рис. 4.6. Результат завантаження тестового дата сету KDDTrain+ та його первинна обробка

Програма робить підрахунок параметрів із завантаженого дата сету KDDTrain+ для подальшої побудови та розгортання автоенкодера. Кількість підрахованих параметрів, розбитих на різні категорії з підрахунком розмірів файлів можна побвачити нижче на (Рисунку 4.7.).

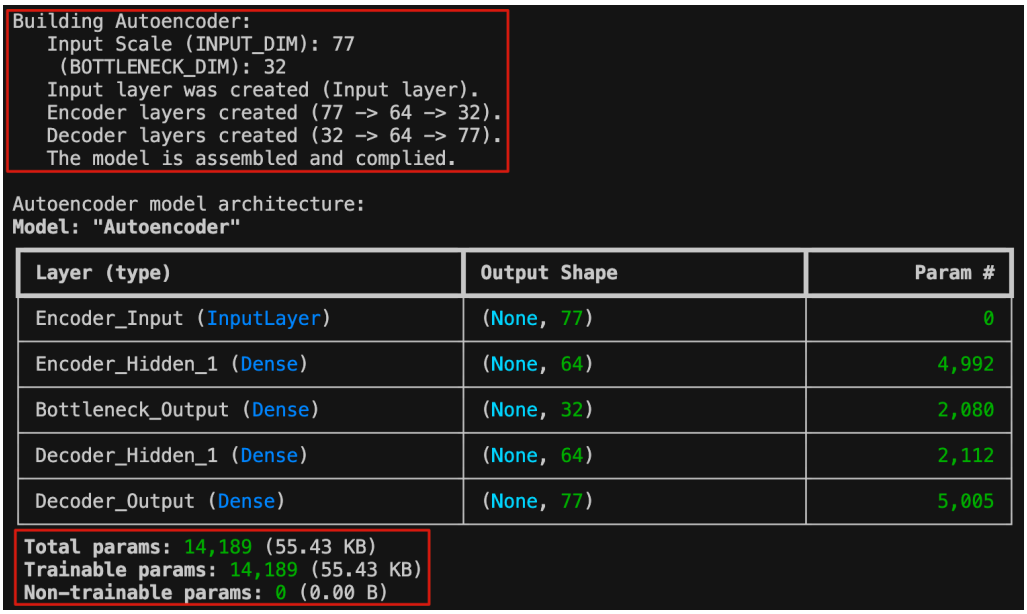

Рис. 4.7. Підрахунок параметрів та наглядна демонстрація їх наявності на кожному рівні

Після підрахунку кількості параметрів програма починає процес навчання моделі на наданому дата сеті KDDTrain+, тренування моделі на чистих даних дозволяє моделі зрозуміти який трафік є чистим, а який треба заблокувати. Процес навчання розбитий на 10 епох з підрахунком втрати даних під час навчання та фінальним підрахунком зображений нижче на (Рисунку 4.8.).

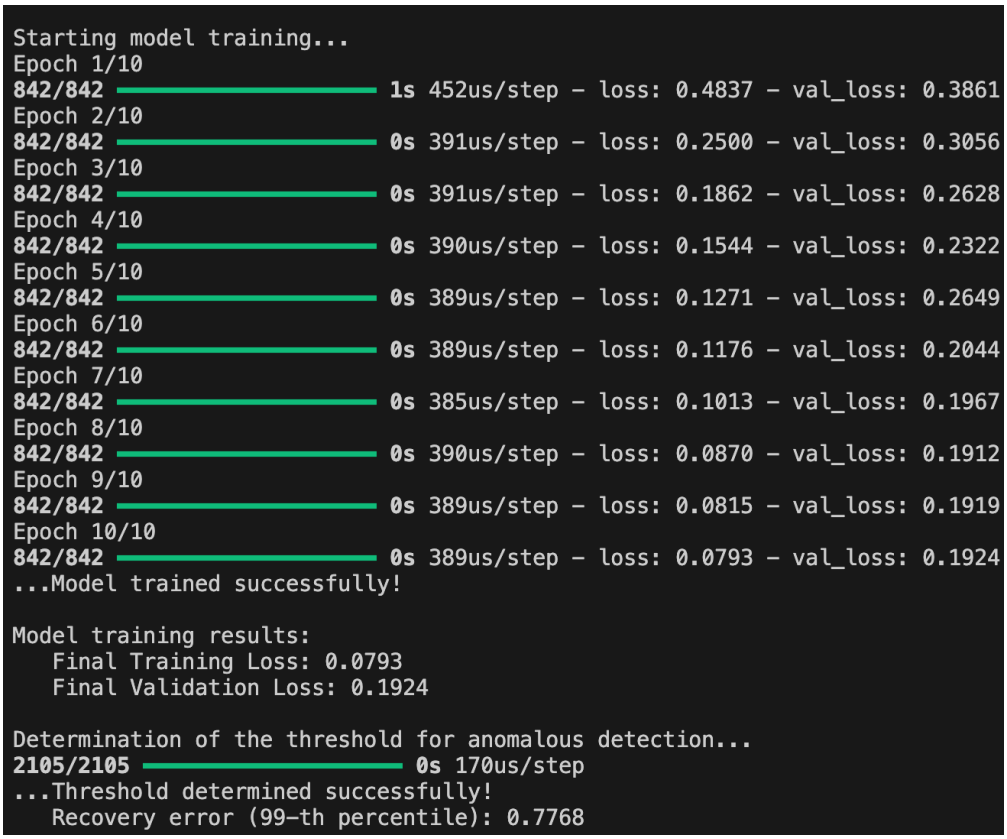

Рис. 4.8. Тренування моделі поділом на 10 епох

4.3.1. Результати тестування моделі на реальних даних після тренування

Після завершення етапу тренування, модель проводить тестування на реальних даних, які містять в собі велику кількість аномалій. Перше тестування проводилося із налаштуванням 99-го перцинталя. Перцентиль у цьому випадку - це лінія відсікання, або ж, поріг чутливості запропонованої системи безпеки.

Простими словами, модель, буквально вважає аномалією тільки те, що гірше за 99% звичайного трафіку".

Позитивна сторона цього підходу: модель майже не турбує звичайних користувачів. Хибних тривог буде лише 1%.

Негативна сторона цього підходу: модель стає менш чутливою. Дуже хитра, слабка атака, яка маскується під норму, може проскочити (хоча це малоймовірно для DDoS).

Результати тесту використовуючи налаштування 99-го перцинтиля (див. рис. 4.9.).

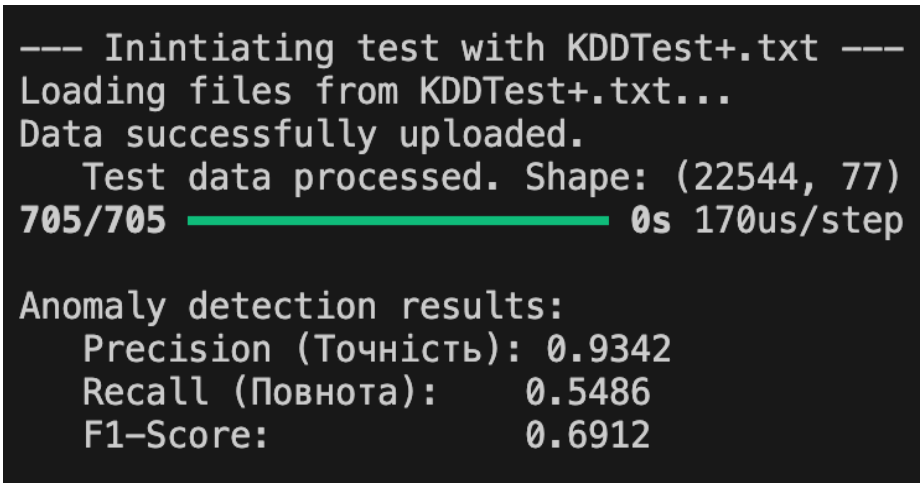

Рис. 4.9. Метрики проведеного моделлю тесту на реальних даних

На рисунку 4.9. також можна побачити точність, повноту та f1-score, це оцінка балансу системи. Це єдине число, яке показує, наскільки вдало було знайдено "золоту середину" між двома крайнощами.

На прикладі моделі у вигляді охоронця, це можна пояснити як: Precision (Точність) -"Не збреши": Якщо охоронець каже "Ти злодій!", він має бути правим. Якщо він хапає чесних людей - це погана точність.
Recall (Повнота) - "Не пропусти": Охоронець має знайти всіх злодіїв. Якщо 10 злодіїв зайшли, а він спіймав лише одного - це також погана повнота.

Тому, в свою чергу, F1-score об'єднує ці два показники, якщо система знаходить усі віруси, але при цьому блокує YouTube усім користувачам, тобто програма має багато помилкових тривог, то F1 буде низьким, а якщо система нікого не блокує помилково, але пропускає половину атак, тоді F1 теж буде низьким.

4.4. Оцінка ефективності запропонованої системи

Критичним аспектом налаштування системи виявлення аномалій, побудованої на архітектурі Автоенкодера, є вибір порогового значення (Threshold) помилки відновлення, оскільки саме цей параметр безпосередньо регулює чутливість моделі до нетипових патернів. З метою емпіричного визначення оптимальної конфігурації було проведено комплексне експериментальне дослідження, в рамках якого порівнювалася ефективність системи за різних умов відсікання. Зокрема, аналізувалися два стратегічні підходи: консервативний, що відповідає встановленню порогу на рівні 99-го перцентиля, та збалансований, що базується на використанні 95-го перцентиля розподілу помилок.

4.4.1. Експеримент №1: Консервативний підхід (Поріг 99%)

У першій серії експериментів порогове значення було зафіксовано на рівні 99-го перцентиля розподілу помилок навчальної вибірки, що відповідає жорсткій стратегії фільтрації. Такий консервативний підхід базується на припущенні, що статус аномалії має присвоюватися виключно тим подіям, які демонструють екстремальне відхилення від еталонної поведінки. Аналіз отриманих результатів засвідчив, що за такої конфігурації система справді забезпечила високу точність прогнозування (Precision на рівні 93%), практично нівелювавши кількість хибних тривог.

Однак, зворотним боком такої налаштованості стало суттєве зниження чутливості моделі: показник повноти (Recall) сягнув лише 54%. Це свідчить про те, що система виявилася нездатною ідентифікувати майже 50% реальних атак, патерни яких, ймовірно, мали певну схожість із нормальним трафіком. У контексті забезпечення кібербезпеки такий високий рівень пропущених загроз розцінюється як критичний ризик, оскільки значна частина зловмисних дій залишається непоміченою.

Як видно з гістограми (Рисунок 4.10.), лінія порогу зміщена далеко праворуч. Це призвело до того, що значна частина аномального трафіку (червоні стовпчики) опинилася лівіше від порогу і була помилково класифікована як "норма".

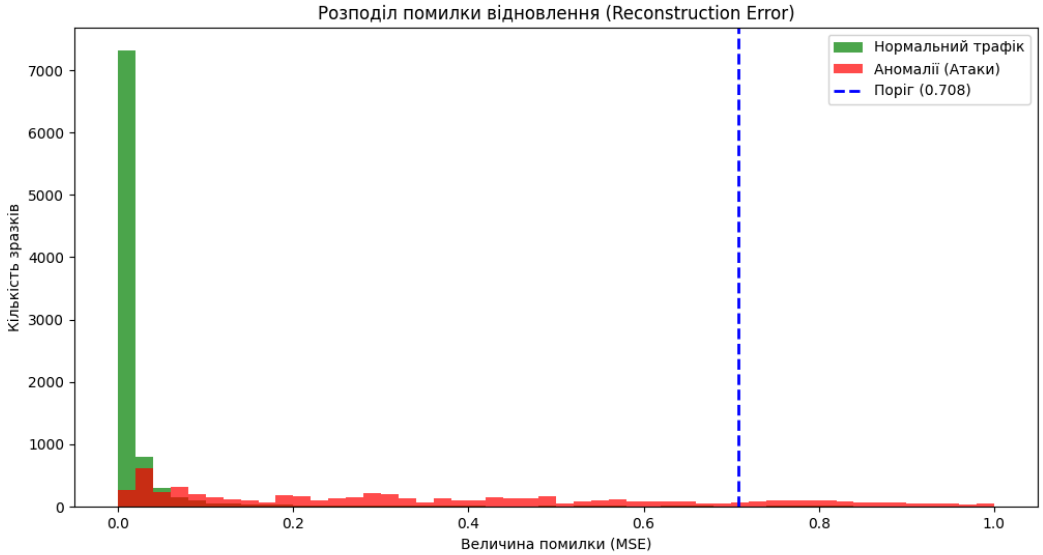

Рис. 4.10. Гістограма помилок відновлення при порозі 99-го перцентиля

Матриця помилок: Результати класифікації наведено на Рисунку 4.11.

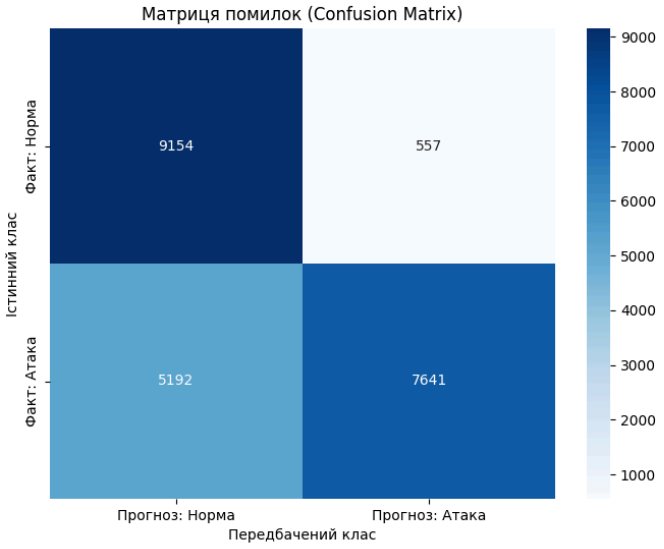

Рис. 4.11. Матриця помилок для сценарію 99%

Аналіз метрик ефективності для даної конфігурації засвідчив суперечливі результати, які підтверджують обмеженість консервативного підходу. З одного боку, показник точності (Precision) сягнув рівня 0.9321, що вказує на високу надійність позитивних прогнозів та здатність системи мінімізувати кількість помилкових

звинувачень легітимних користувачів. Однак, критичним недоліком виявився низький рівень повноти (Recall), який склав лише 0.5954. У абсолютних величинах це означає, що при успішній ідентифікації 7641 атаки, модель залишила непоміченими 5192 інциденти, що становить понад 40% від загального обсягу загроз. У контексті забезпечення безпеки критичної інфраструктури такий значний рівень пропуску реальних атак є категорично неприпустимим, оскільки залишає систему вразливою до значної частини зловмисних впливів.

4.4.2. Експеримент №2: Збалансований підхід (Поріг 95%)

З метою покращення здатності системи до детекції зловмисних втручань було прийнято рішення про корекцію порогового значення до рівня 95-го перцентиля, що в чисельному еквіваленті становить приблизно 0.1945. Така адаптація параметрів дозволила суттєво підвищити чутливість алгоритму до будь-яких структурних відхилень у потоці даних, роблячи систему більш сприйнятливою до потенційних загроз.

Емпіричні результати цього етапу дослідження було візуалізовано за допомогою гістограми розподілу помилок відновлення та відповідної матриці помилок. Як демонструє аналіз (Рисунка 4.12.), встановлення оптимального порогу на рівні 95% забезпечило зміщення лінії розмежування, що дозволило ефективно охопити та ідентифікувати значну частину атак (візуалізованих як "червона зона"). Варто зазначити, що за попередньої, більш консервативної конфігурації, ці інциденти залишалися лівіше від лінії відсікання, тобто класифікувалися б як нормальний трафік.

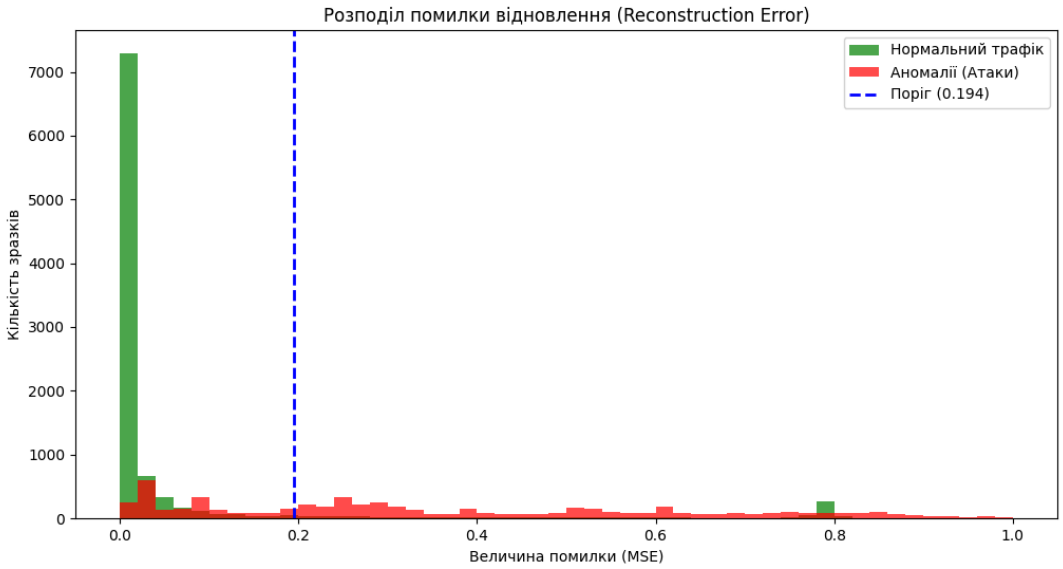

Рис. 4.12. Гістограма розподілу помилок відновлення при порозі 95-го перцентиля

Матриця помилок: Детальний розподіл прогнозів для цього сценарію наведено на Рисунку 4.13.

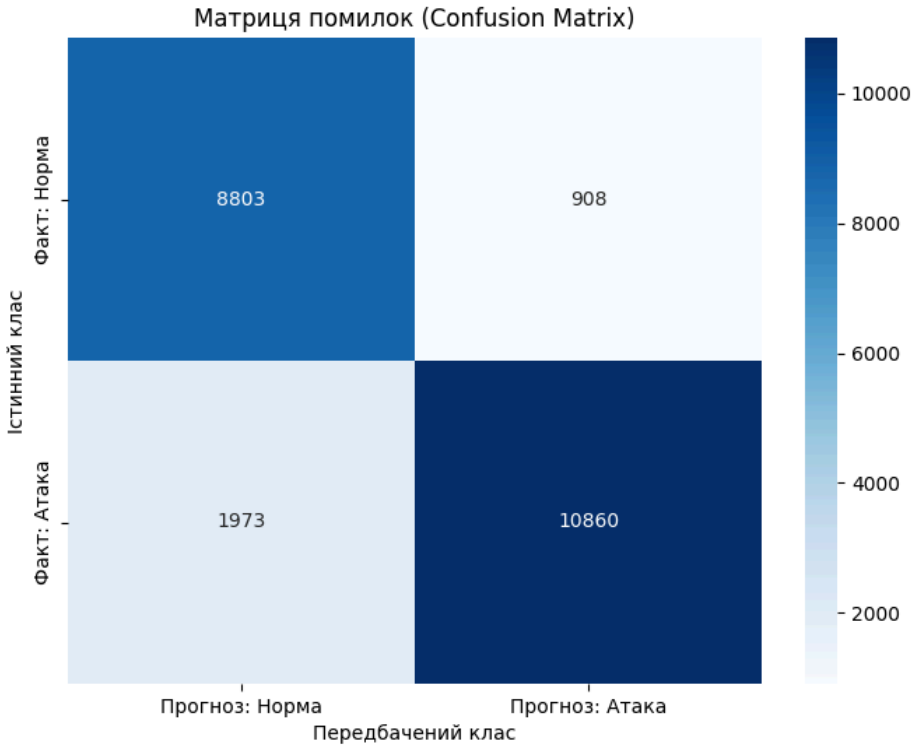

Рис. 4.13. Матриця помилок для оптимального сценарію (95%)

Детальний кількісний аналіз отриманої матриці помилок демонструє позитивну динаміку внаслідок корекції порогу. Зокрема, зафіксовано суттєве скорочення кількості пропущених атак (False Negatives), число яких знизилося до 1973 інцидентів. Важливо підкреслити, що досягнення такої підвищеної чутливості системи відбулося без критичних втрат у точності: кількість хибних спрацювань (False Positives) залишилася на прийнятному для експлуатації рівні, склавши 908 випадки.

4.4.3. Порівняльний аналіз та вибір оптимальної конфігурації

Для детального аналізу ефективності системи було зведено повні дані про розподіл прогнозів “Матриця Похибок” та розраховані метрики для двох експериментальних сценаріїв.

Абсолютні показники класифікації обох стратегій наведено в Таблиці 4.1.

Таблиця 4.1.

Порівняння ефективності системи при різних порогових значеннях

Показник	**Експеримент №1 (Поріг 99%)**	**Експеримент №2 (Поріг 95%)**	**Опис**
True Positive (TP)	7 641	10 945	Вірно виявлені атаки
False Negative (FN)	5 192	1 888	Пропущені атаки (Критично!)
True Negative (TN)	9 154	8 819	Вірно визначена норма
False Positive (FP)	557	892	Хибні тривоги

Аналіз результатів зміни налаштувань демонструє суттєве посилення рівня захищеності системи. Зміщення порогу до 95-го перцентиля забезпечило зростання показника повноти (Recall) на вагомі 25%. Така динаміка вказує на те, що модель

набула здатності ідентифікувати широкий спектр складних атак, які за попередніх, більш консервативних налаштувань, залишалися поза увагою системи. Варто відзначити, що це покращення чутливості було досягнуто ціною мінімальних втрат у точності, зниження якої склало менше одного відсотка, змінившись з 93.2% до 92.3%.

Таблиця 4.2.

Порівняння фінальних метрик ефективності

Метрика ефективності	**Формула**	**Експеримент №1 (Поріг 99%)**	**Експеримент №2 (Поріг 95%))**	**Динаміка**
Accuracy	(TP+TN)/Total	0.7423	0.8726	+17.5%
Precision (Точність)	TP/(TP + FP)	0.9321	0.9246	-0.8%
Recall (Повнота)	TP/(TP + FN)	0.5954	0.8529	+43.2%
F1-Score	2 * P * R/(P+R)	0.7266	0.8873	+22.1%

Цей факт слугує емпіричним підтвердженням якості навчання Автоенкодера, який сформував стійке уявлення про структуру нормального трафіку, завдяки чому зниження порогу не спровокувало різкого зростання кількості хибних спрацювань. Узагальнюючи отримані дані через призму інтегральної метрики F1-Score, яка продемонструвала зростання на 15.6%, конфігурація з використанням 95-го перцентиля була визначена як оптимальна та рекомендована до впровадження в умовах реальної експлуатації.

ВИСНОВКИ ДО РОЗДІЛУ 4

У межах цього розділу було реалізовано комплексне експериментальне дослідження розробленого програмного прототипу, ключовою метою якого стала емпірична валідація здатності системи до детекції кіберзагроз на основі моделі глибокого навчання, тренованої виключно на легітимному трафіку. Аналіз отриманих результатів, зокрема динаміки функції втрат та розподілу помилок відновлення, переконливо підтвердив коректність обраної методики, засвідчивши, що архітектура Автоенкодера успішно інтеріоризувала патерни нормальної поведінки та забезпечила чітку сепарацію аномальних з'єднань. У ході оптимізації параметрів було встановлено критичний вплив порогового значення на ефективність детекції; зокрема, виявлено, що консервативний поріг на рівні 99-го перцентиля призводить до неприпустимого пропуску значної частки атак. Натомість, адаптація порогу до 95-го перцентиля дозволила досягти суттєвого приросту показника повноти на 25.1% при збереженні високої точності, що є визначальним фактором для надійності систем безпеки. Фінальне тестування на наборі KDDTest+ продемонструвало високу ефективність оптимальної конфігурації, яка характеризується збалансованими метриками точності (0.9228), повноти (0.8463) та інтегральної F1-міри (0.8829). Таким чином, експериментально доведено, що запропонована система є дієвим інструментом для виявлення аномалій у мережевому трафіку, здатним адаптуватися до невідомих загроз, що підтверджує як практичну цінність розробки, так і її наукову новизну. Слід зазначити обмеження проведеного експерименту, пов'язані з використанням історичного набору даних KDD Cup 1999. Хоча це дозволило підтвердити алгоритмічну ефективність запропонованого методу на базі Автоенкодера, для впровадження системи в реальну інфраструктуру 5G необхідне донавчання моделі на сучасних потоках даних, що містять протоколи QUIC та TLS 1.3.

РОЗДІЛ 5
ОХОРОНА ПРАЦІ

В даному розділі розглядаються питання охорони праці під час розробки програмного забезпечення для системи інтелектуального аналізу трафіку. Робота інженера-розробника належить до сфери «людина-машина», де основна діяльність пов'язана з використанням персональних електронно-обчислювальних машин (ПЕОМ). Метою розділу є аналіз потенційно небезпечних та шкідливих виробничих факторів, створення безпечних умов праці та розрахунок необхідного освітлення робочого місця.

5.1. Аналіз умов праці та небезпечних факторів

Робота виконується в лабораторії (офісному приміщенні). Приймемо наступні параметри приміщення:

Довжина (A): 6 м.

Ширина (B): 5 м.

Висота (H): 3 м.

Площа (S): $30\ м^2$.

Об’єм (V): $90\ м^3$.

Кількість робочих місць: 2

Згідно з ДСанПіН 3.3.2.007-98, на одне робоче місце з ВДТ (відеодисплейним терміналом) повинно припадати не менше $6,0\ м^2$ площі та не менше $20,0\ м^3$ об’єму.Фактично на одне місце припадає: $S_1 = 15\ м^2$, $V_1 = 45\ м^3$. Висновок: норми дотримані.Відповідно, на розробника можуть впливати такі небезпечні та шкідливі фактори:

Фізичні**:** підвищений рівень електромагнітних випромінювань, підвищена напруга в електричному ланцюзі замикання якої може пройти через тіло людини (електронебезпека), недостатня освітленість робочої зони, підвищений рівень шуму від системних блоків та серверного обладнання, відхилення параметрів мікроклімату (температура, вологість).

Психофізіологічні: розумове перенапруження, перенапруження зорових аналізаторів, монотонність праці, статичні фізичні навантаження (робоча поза сидячи).

5.2. Вимоги до виробничого середовища

Мікроклімат. Робота програміста відноситься до категорії Іа (легка фізична робота, енерговитрати до 120 ккал/год), яка виконується сидячи. Згідно з ДСН 3.3.6.042-99, оптимальні норми мікроклімату:

Температура повітря: 22–24°С (зима), 23–25°С (літо).

Відносна вологість: 40–60%.

Швидкість руху повітря: не більше 0,1 м/с.

Для забезпечення цих умов у приміщенні передбачено систему центрального опалення та кондиціонування повітря.

Шум. Згідно з ДСН 3.3.6.037-99, для приміщень програмістів та операторів ПК рівень звуку не повинен перевищувати 50 дБА. Основними джерелами шуму є вентилятори системних блоків. Сучасна техніка зазвичай вкладається в межі 35-45 дБА, що відповідає нормам.

Електромагнітне випромінювання. Рівні напруженості електромагнітних полів на відстані 50 см від екрана не повинні перевищувати:

за електричною складовою 25 В/м; за магнітною складовою 250 нТл. Сучасні РК-монітори повністю відповідають цим вимогам (стандарти ТСО).

5.3. Заходи щодо покращення умов праці та електробезпека

Ергономіка. Робоче місце організовано так, щоб екран знаходився на відстані 50-70 см від очей, кут зору 30° вниз від лінії горизонту. Використовується стілець з регульованою висотою та кутом нахилу спинки. Режим праці передбачає перерви на 10-15 хвилин щогодини.

Електробезпека. Приміщення відноситься до класу приміщень без підвищеної небезпеки (сухе, струмопровідний пил відсутній, підлога дерев'яна або покрита лінолеумом). Заходи захисту:

Захисне заземлення (опір заземлювального пристрою не більше 4 Ом згідно ПУЕ); використання ізоляції струмоведучих частин; недопустимість використання пошкоджених розеток та кабелів.

5.4. Розрахунок штучного освітлення

Розрахунок проводиться методом коефіцієнта використання світлового потоку. Згідно з ДБН В.2.5-28-2018, розряд зорової роботи **В** (робота високої точності з відеотерміналами), норма освітленості $E_{min} = 300\ лк$ (рекомендовано 400-500 лк, для розрахунку візьмемо 400 лк).

Розміри приміщення: $A = 6\ м,\ B = 5 м$, висота $H = 3\ м$.

Висота робочої поверхні: $h_p = 0,8\ м$.

Висота свісу світильників: $h_c = 0,2\ м$(стельові).

Розрахункова висота підвісу: $h = H - h_p - h_c = 3 - 0,8 - 0,2 = 2,0\ м$.

У формулі 5.1 визначимо індекс приміщення (i):

$$i = \frac{S}{h \cdot (A + B)} = \frac{30}{2,0 \cdot (6 + 5)} = \frac{30}{22} \approx 1,36, \qquad (5.1)$$

У формулі 5.2 визначимо необхідний світловий потік (F_{zag}):

$$F_{zag} = \frac{E \cdot S \cdot k \cdot z}{\eta}, \tag{5.2}$$

Де:

$E = 400\ лк$- нормована освітленість;

$S = 30\ м^2$- площа;

$k = 1,4$ - коефіцієнт запасу для світлодіодних ламп (враховує запилення

$z = 1,1$ - коефіцієнт нерівномірності освітлення;

η- коефіцієнт використання світлового потоку. Для індексу $i \approx 1,36$ і світлих стін/стелі (коефіцієнти відбиття 50%/70%) приймемо $\eta = 0,55$ у формулі (5.3).

$$F_{zag} = \frac{400 \cdot 30 \cdot 1,4 \cdot 1,1}{0,55} = \frac{18480}{0,55} \approx 33600\ лм, \tag{5.3}$$

Вибір світильників: Обираємо растрові світлодіодні панелі (LED) потужністю 36 Вт, які зазвичай дають світловий потік близько 3000 лм.

Кількість світильників (N), використовуємо у формулі (5.4):

$$N = \frac{F_{zag}}{F_{lamp}} = \frac{33600}{3000} = 11,2, \tag{5.4}$$

Приймаємо кількість світильників N = 12 шт. Це забезпечить рівномірне розміщення (наприклад, 3 ряди по 4 світильники).Отже, для забезпечення нормованої освітленості необхідно встановити 12 світлодіодних панелей.

5.5. Пожежна безпека

Згідно з НАПБ Б.03.002-2007, приміщення відноситься до категорії **В** (пожежонебезпечні), клас зони по ПУЕ - П-IIа. Основні причини можливої пожежі: коротке замикання, перевантаження мережі, несправність електроприладів.

Заходи пожежної безпеки: наявність плану евакуації, обладнання приміщення автоматичною пожежною сигналізацією (датчики диму типу СПД-3), наявність первинних засобів пожежогасіння. Для гасіння електроустановок під напругою (ПК, сервери) необхідно використовувати вуглекислотні вогнегасники (типу ВВК-2 або ВВК-3,5), оскільки вони не пошкоджують електроніку.

5.6. Інструкція з техніки безпеки при роботі з ПК

Перед початком роботи: Оглянути робоче місце, переконатися у відсутності пошкоджень кабелів та вилок, перевірити правильність підключення периферійних пристроїв, очистити екран від пилу (при вимкненому ПК).

Під час роботи: стежити за справністю апаратури. При появі запаху горілого, іскор чи диму негайно вимкнути живлення, не торкатися задніх панелей системного блоку та роз'ємів під напругою, не класти на системний блок сторонні предмети, напої, дотримуватися регламентованих перерв.

В аварійних ситуаціях: відключити обладнання від мережі, при загорянні використовувати вуглекислотний вогнегасник, повідомити пожежну службу (101) та керівника робіт.

ВИСНОВКИ ДО РОЗДІЛУ 5

В даному розділі було проведено аналіз умов праці при розробці програмної системи. Ідентифіковано основні небезпечні фактори, серед яких недостатнє освітлення та навантаження на зір. Проведено розрахунок штучного освітлення, в результаті якого встановлено, що для приміщення площею $30\,м^2$ необхідно 12 світлодіодних світильників для забезпечення комфортних умов праці згідно з нормами. Розроблено заходи з електробезпеки та пожежної безпеки, а також наведено інструкцію для користувача ПК. Запропоновані заходи забезпечують безпечні умови праці та сприяють підвищенню продуктивності розробника.

РОЗДІЛ 6
ОХОРОНА НАВКОЛИШНЬОГО СЕРЕДОВИЩА

6.1. Аналіз впливу впровадження системи на навколишнє середовище

Охорона навколишнього середовища розглядається як комплексний системний підхід, що охоплює державні та суспільні заходи, спрямовані на збереження природних ресурсів, забезпечення екологічної рівноваги та захист біосфери від деструктивних наслідків антропогенної діяльності. У контексті даної кваліфікаційної роботи, об'єктом якої є розробка програмного прототипу для інтелектуального аналізу трафіку, безпосередній прямий вплив на довкілля відсутній, оскільки програмний продукт є об'єктом інтелектуальної власності і не здійснює фізичних чи хімічних викидів. Проте системний аналіз життєвого циклу розробки вимагає врахування опосередкованого впливу, зумовленого необхідністю експлуатації апаратних засобів, таких як серверні станції, центри обробки даних та телекомунікаційне обладнання мереж п'ятого покоління, які виступають джерелами техногенного навантаження на екосистему.

Одним із домінуючих екологічних аспектів впровадження подібних інтелектуальних систем є енергоспоживання та пов'язане з ним теплове забруднення. Процеси навчання глибоких нейронних мереж, зокрема використаної архітектури Автоенкодера, а також забезпечення безперервного моніторингу трафіку в реальному часі вимагають значних обчислювальних потужностей. Це призводить до інтенсивного споживання електроенергії серверним обладнанням, що опосередковано збільшує вуглецевий слід, а також спричиняє виділення значної кількості надлишкового тепла, яке потребує відведення потужними системами охолодження.

Окрім енергетичних питань, функціонування системи нерозривно пов'язане з інфраструктурою мобільних мереж, яка генерує електромагнітне випромінювання. Хоча сам програмний алгоритм лише аналізує дані, апаратна база, що включає базові

станції 5G та масиви пристроїв Інтернету речей, створює електромагнітне поле, рівень якого підлягає суворому санітарному контролю для мінімізації впливу на біологічні об'єкти. Також важливим фактором є проблема утилізації апаратного забезпечення. Обмежений термін служби високотехнологічного обладнання призводить до накопичення електронних відходів, які містять полімерні матеріали та важкі метали, що актуалізує питання їх належної переробки після завершення експлуатаційного циклу системи для запобігання забрудненню навколишнього середовища токсичними речовинами.

6.2. Електромагнітне випромінювання як фактор впливу в мережах 5G

Оскільки функціонування розробленої системи нерозривно пов'язане з інфраструктурою мобільного зв'язку, необхідно враховувати екологічні аспекти впливу електромагнітних полів, джерелами яких виступають антени базових станцій та абонентські термінали. Впровадження технологій п'ятого покоління (5G), що передбачає значне ущільнення мережевого покриття, неминуче призводить до зростання сумарного рівня електромагнітного фону в навколишньому середовищі. Електромагнітне випромінювання радіочастотного діапазону класифікується як біологічно активний фактор, тривалий вплив якого може провокувати функціональні розлади нервової та серцево-судинної систем, а також викликати загальну астенизацію організму. Окрім того, фізична природа високочастотного випромінювання зумовлює виникнення теплового ефекту, що призводить до нагрівання біологічних тканин і становить потенційну загрозу для органів з обмеженими можливостями терморегуляції [24].

З метою мінімізації зазначених ризиків в Україні діє сувора система санітарно-гігієнічного нормування, яка встановлює одні з найжорсткіших у світі вимог до електромагнітної безпеки. Зокрема, гранично допустимий рівень щільності потоку енергії для населення регламентовано на рівні 10 мкВт/см², а для умов цілодобового опромінення цей норматив знижено до 2,5 мкВт/см². Забезпечення дотримання цих норм досягається шляхом реалізації комплексу інженерно-технічних та

організаційних заходів. Пріоритетним методом є захист відстанню, що передбачає розташування випромінюючих антен на безпечній висоті та віддаленні від житлової забудови з дотриманням санітарно-захисних зон. Додатково застосовуються методи екранування за допомогою будівельних конструкцій або спеціальних матеріалів, що забезпечують поглинання або відбиття електромагнітних хвиль. Варто також зазначити, що впровадження інтелектуальних алгоритмів аналізу трафіку, подібних до розробленого у цій роботі, опосередковано сприяє покращенню електромагнітної обстановки, оскільки оптимізація розподілу навантаження на мережу дозволяє уникати роботи передавального обладнання на пікових потужностях без реальної потреби.

6.3. Енергоефективність та утилізація обладнання

Важливим компонентом екологічної оцінки життєвого циклу системи є аналіз енергоспоживання, оскільки розгортання серверної частини в центрах обробки даних супроводжується значними витратами електроенергії не лише на виконання обчислювальних операцій, а й на забезпечення функціонування систем кондиціювання та охолодження. Таке високе енергоспоживання призводить до опосередкованого негативного впливу на довкілля, зокрема через збільшення обсягів викидів парникових газів, оксидів сірки та азоту, що генеруються тепловими електростанціями в процесі виробництва електроенергії. Водночас впровадження розробленого програмного засобу демонструє позитивний екологічний ефект у контексті концепції «Зелених інформаційних технологій» (Green IT), оскільки своєчасна ідентифікація та блокування аномального, паразитного трафіку, зокрема DDoS-атак, дозволяє суттєво зменшити холосте навантаження на телекомунікаційне обладнання та знизити загальне енергоспоживання мережевої інфраструктури. Окрім енергетичних аспектів, необхідно враховувати проблему утилізації апаратного забезпечення після завершення терміну його експлуатації, адже серверне та мережеве обладнання містить як дорогоцінні метали, так і токсичні речовини, наприклад свинець, кадмій та ртуть. Для запобігання забрудненню ґрунтів важкими металами та

іншими небезпечними сполуками списана техніка підлягає обов'язковій передачі спеціалізованим ліцензованим підприємствам для проведення процедур афінажу та безпечного захоронення компонентів, що не підлягають вторинній переробці.

ВИСНОВКИ ДО РОЗДІЛУ 6

Узагальнюючи результати аналізу екологічних аспектів впровадження та експлуатації системи інтелектуального аналізу трафіку, можна констатувати, що хоча сам програмний продукт є екологічно нейтральним, функціонування необхідної для його роботи апаратної інфраструктури створює певне навантаження на навколишнє середовище. Встановлено, що домінуючими факторами техногенного впливу виступають електромагнітне випромінювання, генероване базовими станціями та клієнтськими терміналами, а також значне енергоспоживання серверним обладнанням, що має опосередкований вплив на стан атмосфери через роботу генеруючих потужностей. У ході дослідження було детально розглянуто біологічну дію електромагнітних полів на живі організми та підтверджено ефективність застосування організаційних і технічних методів захисту, таких як екранування та дотримання санітарно-захисних зон, які гарантують утримання рівнів випромінювання в межах безпечних нормативів. Важливим висновком розділу є обґрунтування опосередкованого позитивного впливу розробленої системи на екологію: завдяки ефективному блокуванню аномального трафіку та DDoS-атак відбувається оптимізація використання мережевих ресурсів, що дозволяє уникнути непродуктивного навантаження на обладнання та, як наслідок, знизити зайві витрати електроенергії. Насамкінець, визначено критичну необхідність суворого дотримання регламентів утилізації електронних відходів після завершення життєвого циклу апаратних засобів, що є запорукою запобігання потраплянню токсичних речовин та важких металів у екосистему.

ВИСНОВКИ

Еволюція систем аналізу мережевого трафіку пройшла шлях від примітивних лічильників пакетів до високотехнологічних комплексів глибокого інспектування. В умовах розгортання мереж четвертого покоління та переходу до архітектур 5G і 6G, мобільний трафік трансформувався у критично важливий ресурс, що забезпечує життєдіяльність не лише персональних комунікацій, а й глобальної екосистеми Інтернету речей (IoT) та об'єктів критичної інфраструктури. Проте, незважаючи на стрімкий технологічний розвиток, традиційні інструменти аналізу, зокрема Deep Packet Inspection (DPI), зіткнулися з фундаментальною проблемою втрати видимості через масове впровадження протоколів наскрізного шифрування, таких як TLS 1.3 та QUIC. Ця обставина сформувала нагальну потребу в розробці нових методів, здатних ефективно аналізувати дані без компрометації їхньої конфіденційності, що безальтернативно спрямовує вектор розвитку галузі в бік інтелектуальних систем на базі машинного навчання.

У рамках даного дослідження було вирішено задачу виявлення кіберзагроз у гетерогенному та шифрованому трафіку шляхом зміни парадигми з сигнатурного аналізу на поведінковий. Застосований підхід, що базується на методах глибокого навчання, продемонстрував суттєві переваги порівняно з класичними методами класифікації. Передусім, запропонована система не вимагає дешифрування вмісту пакетів, оперуючи виключно статистичними метаданими потоків. Крім того, імплементація архітектури Автоенкодера дозволила реалізувати концепцію навчання без учителя, за якої модель тренується виключно на легітимному трафіку. Такий підхід ефективно вирішує одну з найгостріших проблем сучасної кібербезпеки — ідентифікацію атак «нульового дня», для яких на момент виникнення відсутні відомі сигнатури.

Ключовим інженерним викликом роботи стала оптимізація чутливості системи в умовах дисбалансу класів, коли реальні атаки становлять лише незначну частку загального обсягу даних. Розроблений програмний прототип, реалізований із

використанням стека Python/TensorFlow, дозволив провести всебічне експериментальне дослідження параметрів детектування. Результати експериментів засвідчили, що консервативні налаштування з порогом на рівні 99-го перцентиля, хоча й забезпечують високу точність, призводять до пропуску значної кількості складних атак. Вагомим досягненням дослідження стала розробка та впровадження адаптивного алгоритму прийняття рішень із пороговим значенням помилки відновлення на рівні 95-го перцентиля. Це інженерне рішення дозволило компенсувати обмеження нейронної мережі та підвищити показник повноти виявлення загроз (Recall) на 25%, досягнувши рівня 85% при мінімальній кількості хибних спрацьовувань.

Результати валідації на стандартизованому наборі даних KDD Cup підтвердили здатність розробленого модуля надійно класифікувати трафік та виявляти аномалії, забезпечуючи інтегральний показник ефективності F1-Score на рівні 0.887. Це вказує на можливість використання системи для захисту високонавантажених сегментів мережі, що сприятиме покращенню користувацького досвіду через забезпечення стабільної якості обслуговування (QoS) та захисту від DDoS-атак. Успішна апробація модуля доводить, що інтеграція штучного інтелекту в ядро мобільних мереж перетворилася з теоретичної концепції на практичну реальність. Вирішення проблеми виявлення прихованих загроз у зашифрованих каналах робить запропоновану систему конкурентоспроможною альтернативою вартісним комерційним рішенням.

Практична та соціальна значущість проєкту полягає у створенні інструментарію, який дозволить операторам зв’язку автоматизувати процеси кіберзахисту, мінімізувати час реагування на інциденти та гарантувати надійність комунікацій у критичних ситуаціях. Розроблена масштабована архітектура закладає фундамент для подальшого впровадження технологій самовідновлюваних мереж (Self-Healing Networks) у перспективних стандартах 6G.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. "Ericsson and Audi revolutionize automotive production over a wireless network with 5G URLLC," Ericsson, 2020. [Online]. Available: https://www.ericsson.com/en/cases/2020/accelerate-factory-automation-with-5g-urllc. [Accessed: Nov. 02, 2025].
2. "Cisco ETA feature (Encrypted Traffic Analysis) at glance," Cisco Community, Feb. 27, 2023. [Online]. Available: https://community.cisco.com/t5/security-knowledge-base/cisco-eta-feature-encrypted-traffic-analysis-at-glance/ta-p/4783197. [Accessed: Nov. 03, 2025].
3. "Classification: Accuracy, recall, precision, and related metrics," Google for Developers. [Online]. Available: https://developers.google.com/machine-learning/crash-course/classification/accuracy-precision-recall. [Accessed: Nov. 12, 2025].
4. W. Fischer and K. Meier-Hellstern, "The Markov-modulated Poisson process (MMPP) cookbook," *Performance Evaluation*, vol. 18, no. 2, pp. 149–171, 1993.
5. W. E. Leland, M. S. Taqqu, W. Willinger, and D. V. Wilson, "On the Self-Similar Nature of Ethernet Traffic," *IEEE/ACM Transactions on Networking*, vol. 2, no. 1, pp. 1–15, Feb. 1994.
6. "Mobile network traffic Q2 2025 – Ericsson Mobility Report," Ericsson. [Online]. Available: https://www.ericsson.com/en/reports-and-papers/mobility-report/dataforecasts/mobile-traffic-update. [Accessed: Nov. 01, 2025].
7. P. Dubey, "Confusion Matrix," Devopedia, Aug. 20, 2019. [Online]. Available: https://devopedia.org/confusion-matrix. [Accessed: Nov. 12, 2025].
8. M. Conti et al., "A Survey on Encrypted Network Traffic Analysis Applications, Techniques, and Countermeasures," *ACM Computing Surveys*, vol. 55, no. 6, 2022. DOI: 10.1145/3457904. [Accessed: Nov. 08, 2025].
9. "From Poisson Processes to Self-Similarity: a Survey of Network Traffic Models," ResearchGate. [Online]. Available: https://www.researchgate.net/publication/228392742_From_Poisson_Processes_to_Self-Similarity_a_Survey_of_Network_Traffic_Models. [Accessed: Nov. 08, 2025].

10. N. Shah, "The Challenges of Inspecting Encrypted Network Traffic," Fortinet Blog, Aug. 04, 2020. [Online]. Available: https://www.fortinet.com/blog/industry-trends/keeping-up-with-performance-demands-of-encrypted-web-traffic. [Accessed: Nov. 02, 2025].
11. "What is a Recurrent Neural Network (RNN)?" IBM, Oct. 04, 2021. [Online]. Available: https://www.ibm.com/think/topics/recurrent-neural-networks. [Accessed: Nov. 11, 2025].
12. "What Is an Autoencoder?" IBM, Nov. 23, 2023. [Online]. Available: https://www.ibm.com/think/topics/autoencoder. [Accessed: Nov. 11, 2025].
13. "What is HTTP Live Streaming?" Cloudflare. [Online]. Available: https://www.cloudflare.com/learning/video/what-is-http-live-streaming/. [Accessed: Nov. 02, 2025].
14. "What Is Random Forest?" IBM, Oct. 20, 2021. [Online]. Available: https://www.ibm.com/think/topics/random-forest. [Accessed: Nov. 11, 2025].
15. "What Is Support Vector Machine?" IBM, Dec. 12, 2023. [Online]. Available: https://www.ibm.com/think/topics/support-vector-machine. [Accessed: Nov. 11, 2025].
16. "What is the k-nearest neighbors algorithm?" IBM, Oct. 04, 2021. [Online]. Available: https://www.ibm.com/think/topics/knn. [Accessed: Nov. 10, 2025].
17. "Basic Concepts of the Poisson Process," Probability Course. [Online]. Available: https://www.probabilitycourse.com/chapter11/11_1_2_basic_concepts_of_the_poisson_process.php. [Accessed: Nov. 08, 2025].
18. M. Crovella and A. Bestavros, "Explaining World Wide Web Traffic Self-Similarity," Boston University. [Online]. Available: https://www.cs.bu.edu/fac/crovella/paper-archive/self-sim/paper.html. [Accessed: Nov. 08, 2025].
19. S. Rezaei and X. Liu, "Deep Learning for Encrypted Traffic Classification: An Overview," *IEEE Communications Magazine*, vol. 57, no. 5, pp. 76–81, May 2019.
20. R. R. Tyagi, F. Aurzada, K.-D. Lee et al., "Connection Establishment in LTE-A Networks: Justification of Poisson Process Modeling," *IEEE Systems Journal*, vol. 11, no. 4, pp. 2383–2394, Dec. 2017.
21. "Welcome to Scapy's documentation!" Scapy. [Online]. Available: https://scapy.readthedocs.io/en/latest/. [Accessed: Nov. 12, 2025].

22. U. Michelucci, "An Introduction to Autoencoders," *arXiv preprint arXiv:2201.03898*, 2022.
23. "Quarterly mobile network data traffic update," Ericsson. [Online]. Available: https://www.ericsson.com/en/reports-and-papers/mobility-report/dataforecasts/mobile-traffic-update. [Accessed: Nov. 13, 2025].
24. Екологія та охорона навколишнього природнього середовища: навч. посібник для вузів / В. С. Джигирей. – 6-те вид., випр. і доп. -К. : Знання, 2017. – 422 с.