**НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ**

**«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ**

**імені ІГОРЯ СІКОРСЬКОГО»**

**НАВЧАЛЬНО-НАУКОВИЙ ФІЗИКО-ТЕХНІЧНИЙ ІНСТИТУТ**

**КАФЕДРА МАТЕМАТИЧНОГО МОДЕЛЮВАННЯ ТА АНАЛІЗУ ДАНИХ**

«До захисту допущено»

Виконувачка обов'язків

завідувача кафедри

_____ Ганна ЯЙЛИМОВА

(підпис) (ініціали, прізвище)

"____ "__ 2024 р.

# Дипломна робота

## на здобуття ступеня бакалавра

зі спеціальності 113 Прикладна математика на тему:

«Система пошуку аномального трафіку за допомогою методів машинного навчання»

Виконала: студентка 4 курсу, групи ФІ-02

Чередниченко Марія Вікторівна _____

Керівник: ст. викладач кафедри ММАД Д.В. Тітков _____

Рецензент: ст. викладач кафедри ІБ О.В. Козленко _____

Засвідчую, що у цій дипломній роботі немає

запозичень з праць інших авторів

без відповідних посилань.

Студент _____ _____ (підпис)

Київ – 2024 року

**НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ**

**«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ**

**імені Ігоря СІКОРСЬКОГО»**

**НАВЧАЛЬНО-НАУКОВИЙ ФІЗИКО-ТЕХНІЧНИЙ**

**ІНСТИТУТ**

**Кафедра математичного моделювання та аналізу даних**

Рівень вищої освіти – перший (бакалаврський)

Спеціальність: 113 Прикладна математика

ЗАТВЕРДЖУЮ

Виконувачка обов'язків

завідувача кафедри

_____ Ганна ЯЙЛИМОВА

(підпис) (ініціали, прізвище)

«____ »_____ 2024 р.

**ЗАВДАННЯ**

**на дипломну роботу**

Студентка: Чередниченко Марія Вікторівна

1. Тема роботи: «*Система пошуку аномального трафіку за допомогою методів машинного навчання*»,

керівник роботи: ст. викладач кафедри ММАД ННФТІ Д.В. Тітков

затверджені наказом по університету від «____»_____2024 р. № _____

2. Термін подання студентом роботи: «7» червня 2024 р.

3. Вихідні дані до роботи: *дані про нормальний та аномальний трафік*

*4.* Зміст роботи: *було досліджено наявні роботи по розв'язанню задачі детекції аномального трафіку, проаналізовані різноманітні методи та їхню придатність до вирішення поставленої задачі, та реалізовано модель, що успішно класифікує трафік на аномальний та нормальний.*

5. Перелік ілюстративного матеріалу: *презентація доповіді*

6. Дата видачі завдання «5» вересня 2023 р.

Календарний план

| № з/п | Назва етапів виконання дипломного проекту | Термін виконання етапів проекту | Примітка |
|---|---|---|---|
| 1 | Дослідження теоретичного матеріалу | Жовтень-Грудень 2023р. | Виконано |
| 2 | Постановка технічного завдання | Січень 2024р. | Виконано |
| 3 | Аналіз методів реалізації технічного завдання | Лютий-Березень 2024р. | Виконано |
| 4 | Реалізація технічного завдання | Квітень 2024р. | Виконано |
| 5 | Оформлення звіту | Травень-Червень 2024р. | Виконано |

Студент _____

(підпис) (ініціали, прізвище)

Керівник роботи _____

(підпис) (ініціали, прізвище)

# РЕФЕРАТ

Дипломна робота за обсягом становить 39 сторінок тексту, містить 3 додатки. Для дослідження було використано 9 бібліографічних найменувань.

Метою роботи є дослідження наявних рішень і моделей детекції надзвичайного мережевого трафіку, аналіз існуючих методів машинного навчання детекції і класифікації і створення моделі класифікації мережевого трафіку за його поведінкою на різних проміжках часу. Об’єктом дослідження є мережевий трафік та дані про нього.

**Ключові слова**: мережевий трафік; аномальний трафік; детекція аномалій; мережеві атаки.

# ABSTRACT

The thesis includes 39 pages of the text, and 3 supplements. The research was based on 9 bibliographic references.

The aim of the thesis is to investigate existing solutions and models for detecting abnormal network traffic, analyze current machine learning methods for detection and classification, and develop a model for classifying network traffic based on its behavior over different time intervals. The object of the research is network traffic and its related data.



**Keywords:** network traffic; traffic anomaly; anomaly detection; network attacks.

**ЗМІСТ**

## Перелік використаних термінів і скорочень

**RNN** – рекурентна нейронна мережа.

**CNN** – конволютивна (згорткова) нейронна мережа.

**CSE-CIC-IDS2018** – Установа Безпеки Комунікацій—Канадський Інститут Кібербезпеки—Система Детекції Втручань 2018.

**Комп’ютерна мережа -** система взаємопов'язаних між собою пристроїв, що можуть обмінюватися даними та ресурсами.

**Мережеві протоколи** – визначені набори правил та процедур для регуляції передачі даних у мережі.

**Мережеві аномалії** – непередбачувані, незвичайні або відхилені від звичайних станів події чи патерни, що виникають в комп'ютерних мережах.

**Аномалії трафіку** – підвид мережевих аномалій, що вказують на неочікувані та надзвичайні патерни передачі даних.

**Мережева атака** – певна дія, що здійснюється з метою отримання контролю над будь-якою локальною або віддаленою обчислювальною системою або комп'ютером.

**Bruteforce** – мережева атака, спрямована на отримання доступу до ресурсу методом підбору паролів/ключів.

**Benign** – стандартний термін позначення нормального трафіку.

# Вступ

**Актуальність**. З розвитком інформаційних технологій та збільшенням обсягу мережевого трафіку зростає необхідність забезпечення кібербезпеки. Виявлення аномального мережевого трафіку є критично важливою задачею для запобігання кібератакам, витоку інформації та інших зловмисних дій. Сучасні методи глибокого навчання, такі як рекурентні нейронні мережі (RNN) та згорткові нейронні мережі (CNN), дозволяють ефективніше вирішувати ці задачі завдяки здатності обробляти часові послідовності та складні взаємозв'язки в даних.

**Об’єкт дослідження**: мережевий трафік сучасних комп'ютерних мереж, що включає як звичайні, так і аномальні дані.

**Предмет дослідження**: методи глибокого навчання, зокрема рекурентні нейронні мережі (RNN) та згорткові нейронні мережі (CNN), для виявлення аномального мережевого трафіку.

**Мета роботи**: розробка ефективної моделі детекції аномального мережевого трафіку на основі методів глибокого навчання, яка забезпечить високу точність і швидкість виявлення аномалій у реальному часі.

**Завдання**:

1. Провести огляд сучасних методів детекції аномального мережевого трафіку.
2. Проаналізувати переваги та недоліки існуючих підходів.
3. Зібрати та підготувати датасет для навчання моделі.
4. Розробити модель на основі методів глибокого навчання (RNN та CNN).
5. Провести тренування та валідацію моделі.
6. Оцінити ефективність моделі на тестових даних.
7. Визначити можливості практичного застосування розробленої моделі.

**Методи дослідження**:

1. Аналіз літератури та наукових джерел щодо методів детекції аномального трафіку.
2. Обробка та аналіз даних за допомогою бібліотек Python (Pandas, scikit-learn, matplotlib).

3. Розробка та тренування моделей глибокого навчання за допомогою бібліотек TensorFlow та Keras.
4. Оцінка точності та ефективності моделей за допомогою метрик MSE, точності та втрат.

**Наукова новизна дослідження** полягає у розробці та впровадженні моделі класифікації аномального мережевого трафіку на основі згорткових нейронних мереж (1D CNN), яка показала високу точність і ефективність у виявленні аномалій. Запропонований підхід дозволяє обробляти великі обсяги даних у реальному часі, що є значним внеском у сфері кібербезпеки.

**Практичне значення дослідження** полягає у можливості застосування розробленої моделі для моніторингу мережевого трафіку у реальному часі. Це дозволяє своєчасно виявляти потенційні загрози та підвищувати загальну стійкість інформаційної інфраструктури, що є важливим для захисту від кіберзагроз та забезпечення безпеки даних.

# РОЗДІЛ 1. ОГЛЯД ПРОБЛЕМИ ТА АНАЛІЗ НАЯВНИХ РІШЕНЬ

## 1.1 Актуальність проблеми детекції аномального трафіку

У сучасному світі інформаційні технології та комп'ютерні мережі стали невід'ємною частиною життя, забезпечуючи комунікацію, обмін даними та доступ до різноманітних сервісів. Однак, із розвитком мережевих технологій зростає і загроза безпеці інформації, що передається через ці мережі. Однією з найважливіших задач у сфері кібербезпеки є детекція аномального трафіку, який може свідчити про різноманітні види атак, зловмисну діяльність або неправомірне використання мережевих ресурсів.

Визначимо, що собою являє комп’ютерна мережа. Комп’ютерна мережа - це система взаємопов'язаних між собою пристроїв, що можуть обмінюватися даними та ресурсами. Мережі дозволяють пристроям передавати інформацію, мати спільний доступ до ресурсів та забезпечують спільне використання програм і обладнання. Мережі мають різні конфігурації розмірів та топології. Також усі учасники мережі спілкуються між собою за допомогою мережевих протоколів - визначених наборів правил та процедур для регуляції передачі даних у мережі. Протоколи визначають формати даних, способи їхньої адресації, правила взаємодії тощо.

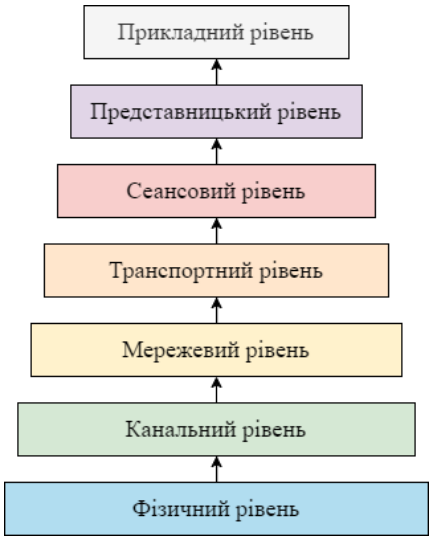


Рисунок 1.1. Діаграмне представлення моделі OSI

Одним з найпоширеніших представлень комп’ютерної мережі є так звана “модель OSI” (Open System Interconnections), яку розробила і запровадила Міжнародна організація зі стандартизації (International Organization for Standardization, ISO) у кінці 1970-их років [1]. Модель OSI складається з семи шарів, як показано на Рис. 1.1:

1. Фізичний рівень (Physical Layer) - перший, фундаментальний рівень моделі. Відповідає за фізичне з’єднання між пристроями. Визначає електричні, механічні, процедурні та функціональні характеристики інтерфейсів та середовища передачі. Приклади представлення першого рівня - Ethernet кабелі, повторювачі, мережеві адаптери, тощо.

2. Канальний рівень (Data Link Layer) - другий рівень, що відповідає за контроль потоку даних, а також виявлення та виправлення помилок у комунікації між двома сусідніми вузлами.

3. Мережевий рівень (Network Layer) - третій шар моделі. Відповідає за за маршрутизацію пакетів даних між вузлами, що знаходяться у різних мережах. Забезпечує логічну адресацію та вибір маршруту для передачі даних. Приклад цього шару - протокол Internet Protocol (IP); цей рівень працює з IP-адресами пристроїв у мережі.

4. Транспортний рівень (Transport Layer) - четвертий рівень, що забезпечує надійну передачу даних між вузлами від кінця до кінця. Відповідає за сегментацію даних, контроль потоку та управління помилками. Також відповідає за правильну послідовність сегментів і повторну передачу у разі виявлення помилок. Найпоширеніші протоколи транспортного рівня - Transport Control Protocol (TCP) та User Diagram Protocol (UDP).

5. Сеансовий рівень (Session Layer) - п’ятий шар моделі OSI. Цей рівень забезпечує управління сеансами або з'єднаннями між додатками. Відповідає за встановлення,

управління та завершення сеансів, а також за синхронізацію та відновлення сеансів у разі розриву з'єднання.

6. Представницький рівень (Presentation Layer) - шостий рівень моделі, який займається перетворенням представлення даних для розуміння їх пристроями та додатками. Також забезпечує кодування, декодування, шифрування та дешифрування даних, і відповідає за їхнє стискання та розпакування. Шостий шар представляється протоколами SSL/TLS (Secure Sockets Layer/Transport Layer Security).

7. Прикладний рівень (Application Layer) - сьомий, найвищий шар моделі OSI. Саме цей рівень відповідає за інтерфейс між користувачем та мережею. Найпоширеніші протоколи, що представляють цей рівень - HTTP/HTTPS (Hyper Text Transfer Protocol/Hyper Text Transfer Protocol Secure) , DNS (Domain Name System), FTP (File Transfer Protocol), SSH (Secure Shell Protocol), тощо.

Потік даних між учасниками мережі називається мережевим трафіком. Дані, що передаються у трафіку, можуть представлятися різними типами інформації, як-от файлами, електронними листами, веб-сторінками. Основні аспекти мережевого трафіку це: дані про відправника і отримувача; тип даних; протоколи, за якими передаються дані; обсяг трафіку та швидкість його передачі.

Випадки нетипової поведінки мережевого трафіку називають мережевими аномаліями. Мережеві аномалії - це непередбачувані, незвичайні або відхилені від звичайних станів події чи патерни, що виникають в комп'ютерних мережах. Аномалії можуть бути наслідком різних факторів і вказувати на потенційні проблеми або атаки на мережеву безпеку.

Ця робота зосереджена на дослідженні аномалій трафіку. Аномалії трафіку є підвидом мережевих аномалій, що вказують на неочікувані та надзвичайні патерни передачі даних. Вони можуть бути спричинені технічними проблемами, атаками тощо. Деякі основні види аномалій трафіку включають збільшення обсягу трафіку, потребу у надмірному пропускному потоці та велику кількість відмов до доступу

(Denial of Service, DoS) або вимог до обслуговування (Distributed Denial of Service, DDoS).

Небезпека аномалій трафіку полягає в тому, що вони можуть свідчити про атаку або проблеми, що можуть призвести до відмови в обслуговуванні, витоку конфіденційної інформації, порушення безпеки даних, пошкодження обладнання інфраструктури систем тощо.

Приклади аномалій включають неочікувану активність на серверах, що може свідчити про атаку або витік даних, надмірне використання ресурсів сервера, яке є ознакою атаки або технічної проблеми, зміни у пропускній здатності та надмірне споживання електроенергії, що може свідчити про аномальну активність сервера.

Таким чином, проблема детекції аномального трафіку є надзвичайно актуальною у сучасному контексті швидкого зростання кіберзагроз. У зв'язку з цим, необхідно розробляти та вдосконалювати методи та інструменти для ефективного виявлення аномалій у мережевому трафіку, що є головною метою цього дослідження.

## 1.2 Огляд наявних рішень

Була проаналізована література пов’язана з вивченням трафіку та його аномаліями. Основними теоретичними джерелами стали статті, що описують методи машинного навчання для аналізу даних, а також публікації, присвячені темі комп’ютерних мереж.

Задача детекції аномалій трафіку є дуже давньою, адже вона з’явилася з самого початку створення Інтернету в сучасному його представленні. З початком розвитку машинного навчання та нейронних мереж комп’ютерні науковці шукали і досі продовжують шукати способи більш успішного та точного виявлення підозрілої активності в мережі. За аномаліями трафіку можуть ховатися як і цілком очікувана активність (наприклад, інформаційний ажіотаж, популярне або необхідне джерело), так і зловмисна (наприклад, різні види атак, що спрямовані на крадіжку інформації чи підрив роботи технічного обладнання). Наразі методи машинного

навчання добре описані і пояснені багатьма науковцями, представлені в різних джерелах.

Розглянемо декілька досліджень, що демонструють роботу з трафіком та нейронні мережі для створення моделей, які б вирішували задачу виявлення аномалій в потоці.

Робота [2] зосереджена на виявленні мережевих аномалій, зокрема двох найпоширеніших: сканування портів (port scan) і мережі (netscan). У статті представлено та проаналізовано результати застосування традиційних підходів на платформі Apache Spark. Використовуючи технології big data, дослідження значно скорочує час виконання алгоритму виявлення, що робить його придатним для високошвидкісних мереж. У статті описується використаний підхід і наводяться експериментальні результати щодо ефективності виявлення та часу виконання. Використовуються дані реального трафіку з архіву MAWI, а результати порівнюються з детекторами аномалій MAWILab, які вважаються золотим стандартом у цій галузі.

Недоліки роботи включають детальний порівняльний аналіз сильних і слабких сторін алгоритму порівняно з MAWILab. Дослідження не дає поняття масштабованості алгоритму та вимог до ресурсів, що могло б дати розуміння щодо його застосовності в різних мережевих середовищах. У статті визнається, що, незважаючи на численні дослідження, ефективне й точне виявлення мережевих аномалій залишається складною проблемою.

Стаття [3] представляє новий підхід до аналізу онлайн-трафіку, зосереджуючись на узагальненні даних вимірювання трафіку. На відміну від попередніх методів, обмежених підсумками однієї змінної, цей підхід фіксує стан мережі з багатоваріантних атрибутів за допомогою кластеризації для агрегування багатовимірних змінних. Кластеризовані результати представляють стан мережі щодо змінних, що контролюються, і дозволяють порівняти з шаблонами з попередніх часових вікон, полегшуючи інтуїтивно зрозумілий аналіз. Дослідження демонструє здійсненність методу з двома випадками використання: оцінка змін стану та виявлення аномальних станів. Масштабні експерименти з використанням

загальнодоступних трасувань і даних із трасування трафіку ESnet підтверджують ефективність цього підходу на основі шаблонів для багатовимірного аналізу мережевого трафіку в Інтернеті, надаючи як візуальні, так і кількісні інструменти для аналізу.

Прогалини роботи включають проблему масштабованості, оскільки створення кластерів може зайняти багато часу, займаючи приблизно 10 секунд із 16 000 точками даних. Щоб зробити підхід більш практичним для великомасштабних мереж, важливо вирішити цю проблему. Хоча масштабованість можна покращити за допомогою вибірки, прості методи вибірки можуть призвести до відхилень порівняно з результатами, отриманими в результаті аналізу всього набору даних, що свідчить про необхідність більш надійних і ефективних методів вибірки для аналізу мережевого трафіку. Кластеризація може бути чутливою до невеликої кількості шуму, що може вплинути на точність представлення стану мережі.

### 1.3 Аналіз переваг та недоліків існуючих методів

Дослідження наявних методів і розв’язків задачі детекції аномалій мережевого трафіку охоплює аналіз різних технологій і підходів, використовуваних для ідентифікації неправильних, незвичайних або зловмисних дій у мережах. Серед основних методів можна виділити статистичні методи, які виявляють відхилення від нормального мережевого трафіку, а також застосування машинного навчання та глибокого навчання. Машинне навчання включає як методи навчання з учителем, так і без учителя, що дозволяє ідентифікувати аномальні патерни, зокрема за допомогою кластеризації або нейронних мереж. Зокрема, LSTM і GRU є популярними у використанні для аналізу мережевих даних.

Ця область досліджень також зіштовхується з низкою викликів, включаючи масштабованість системи для обробки великих обсягів даних у реальному часі, адаптацію до динамічно змінюваного мережевого трафіку та зменшення помилкових спрацьовувань, які можуть негативно впливати на користувацьке прийняття та довіру. Науковці також активно впроваджують технологічні інновації, такі як автоматизація та інтеграція детекторів аномалій з іншими

захисними системами, використання хмарних рішень для підвищення масштабованості та доступності захисту. Ці інновації відіграють ключову роль у вдосконаленні методів детекції, адаптації до нових викликів кібербезпеки та забезпеченні захисту на сучасному етапі розвитку інформаційних технологій.

**Висновки.**

У першому розділі роботи було проведено детальний огляд проблеми детекції аномального мережевого трафіку та аналіз наявних рішень. Встановлено, що з розвитком інформаційних технологій та комп'ютерних мереж зростає загроза безпеці інформації, що передається через ці мережі. Детекція аномального трафіку є важливою задачею для забезпечення кібербезпеки, оскільки вона дозволяє виявляти різноманітні види атак, зловмисну діяльність або неправомірне використання мережевих ресурсів. Було розглянуто моделі комп'ютерних мереж, типи мережевого трафіку та методи виявлення аномалій, що є необхідним підґрунтям для подальших досліджень і розробки нових методів детекції аномалій.

# РОЗДІЛ 2. АНАЛІЗ ІСНУЮЧИХ МЕТОДІВ ВИРІШЕННЯ ЗАДАЧІ ДЕТЕКЦІЇ АНОМАЛІЙ

## 2.1 Задача детекції аномалій

Задача детекції аномалій є критично важливою у контексті забезпечення безпеки мережевих систем та інформаційних ресурсів. Виявлення аномалій передбачає ідентифікацію незвичайних або підозрілих патернів у мережевому трафіку, які можуть свідчити про потенційні загрози, такі як кібератаки, зловмисні дії або технічні несправності. Ефективна детекція аномалій дозволяє оперативно реагувати на інциденти, мінімізувати шкоду та підвищити загальну стійкість інформаційної інфраструктури.

У цьому розділі розглянуто основні аспекти задачі детекції аномалій, включаючи різні підходи та технології, що використовуються для виявлення аномалій у мережевому трафіку. Зокрема, проаналізуємо статистичні методи, методи машинного навчання та глибокого навчання, а також сучасні інновації в цій галузі.

В аналізі даних виявлення аномалій (також відоме як виявлення викидів і іноді як виявлення новизни) зазвичай розуміється як ідентифікація рідкісних елементів, подій або спостережень, які значно відхиляються від більшості даних і не відповідають чітко визначеному уявленню про нормальну поведінку. Такі приклади можуть викликати підозру, що вони створені іншим механізмом або здаються невідповідними решті набору даних. [4]

Виявлення аномалій знаходить застосування в багатьох галузях, включаючи кібербезпеку, медицину, комп’ютерний зір, статистику, нейронауку, правоохоронні органи та фінансове шахрайство. Спочатку аномалії шукали для явного відхилення або виключення з даних для полегшення статистичного аналізу, наприклад, для обчислення середнього значення або стандартного відхилення. Їх також виділяли, щоб покращити прогнозування моделей, таких як лінійна регресія, і нещодавно їх виділення покращує роботу алгоритмів машинного навчання. Однак у багатьох випадках самі аномалії становлять інтерес і є найбажанішими

спостереженнями у всьому наборі даних, які потрібно виявити та відокремити від шуму або неактуальних викидів.

Існують три широкі категорії методів виявлення аномалій. Керовані методи виявлення аномалій вимагають наявності набору даних, позначених як "нормальні" та "аномальні", і передбачають навчання класифікатора. Однак цей підхід рідко використовується через загальну недоступність позначених даних і неврівноважену природу класів.

Напівкеровані методи виявлення аномалій припускають, що частина даних позначена. Це може бути будь-яке поєднання нормальних або аномальних даних, але частіше за все ці методи будують модель, яка представляє нормальну поведінку на основі заданого нормального навчального набору даних, і потім перевіряють ймовірність того, що тестовий випадок буде відтворений цією моделлю.

Некеровані методи виявлення аномалій припускають, що дані не позначені, і є найбільш часто використовуваними через їх ширше та більш релевантне застосування.

Концепція виявлення вторгнень, яка є критичним компонентом виявлення аномалій, значно еволюціонувала з часом. Спочатку це був ручний процес, коли системні адміністратори спостерігали за незвичайною активністю, наприклад, доступом до облікового запису користувача, який перебуває у відпустці, або несподіваною активністю принтера. Цей підхід був непрактичним для масштабування і згодом був замінений аналізом аудиторських і системних журналів на предмет ознак зловмисної поведінки.

До кінця 1970-х і початку 1980-х років аналіз цих журналів використовувався переважно ретроспективно для розслідування інцидентів, оскільки обсяги даних робили неможливим їх моніторинг у реальному часі. Зниження вартості цифрового зберігання зрештою призвело до того, що аудиторські журнали почали аналізуватися онлайн, з використанням спеціалізованих програм для обробки даних. Ці програми, однак, зазвичай запускалися в неробочий час через їх високу обчислювальну інтенсивність.

У 1990-х роках з'явилися системи виявлення вторгнень у реальному часі, здатні аналізувати аудиторські дані в момент їх створення, що дозволило негайно виявляти та реагувати на атаки. Це стало значним зрушенням у бік проактивного виявлення вторгнень.

З подальшим розвитком галузі увага змістилася на створення рішень, які можуть бути ефективно реалізовані в великих і складних мережевих середовищах, адаптуючись до постійно зростаючої різноманітності загроз безпеці та динамічного характеру сучасних обчислювальних інфраструктур.

В цьому випадку, задача детекції аномального трафіку полягає в створенні моделі, яка б на деяких часових проміжках визначала, чи є даний відрізок мережевого трафіку надзвичайним. Важливо використовувати саме проміжки часу, адже якщо взяти лише один запис без попереднього і майбутнього контексту - не буде зрозуміло, чи це частина аномалії, чи норми. Частину аномального трафіку займають мережеві атаки; наприклад, брутфорс (Bruteforce) - атака, що спрямована на отримання доступу до ресурсу завдяки підбору ключів доступу до нього. Одним з важливих аспектів для урахування при створенні моделі є факт того, що трафік атак має патерн, коли нормальний трафік має хаотичну натуру через велику підключень різних клієнтів і їхню різноманітну активність за один вказаний проміжок часу.

## 2.2 Вирішення задачі детекції аномалій за допомогою RNN

Найбільш поширеним методом визначення аномалій є рекурентні нейронні мережі (recurrent neural networks, RNN), а саме - за допомогою використання підвиду RNN, LSTM (long short-term memory).

Рекурентна нейронна мережа (RNN) - це один із двох найпоширеніших типів штучних нейронних мереж, які характеризуються напрямком потоку інформації між своїми шарами. На відміну від однонаправленої прямопропускної нейронної мережі, вона є двонаправлена штучна нейронна мережа - це означає, що вона дозволяє виходу з деяких вузлів впливати на наступні входи до тих самих вузлів. Їх здатність використовувати внутрішню пам'ять для обробки довільних

послідовностей вхідних даних робить їх придатними для завдань, таких як розпізнавання нерозділеного, зв'язаного рукописного тексту або розпізнавання мови. Термін "рекурентна нейронна мережа" використовується для позначення класу мереж з нескінченною імпульсною характеристикою, тоді як "конволюційна нейронна мережа" стосується класу з обмеженою імпульсною характеристикою. Обидва класи мереж демонструють тимчасову динамічну поведінку. Скінченна імпульсна рекурентна мережа можна представити спрямованим ациклічним графом, який можна розгорнути та замінити строго прямопропускною нейронною мережею, тоді як нескінченна імпульсна рекурентна мережа є спрямованим циклічним графом, який не можна розгорнути. [5]

Додаткові збережені стани та сховища під прямим контролем мережі можуть бути додані як до нескінченних, так і до скінченних імпульсних мереж. Інша мережа або граф може також замінити сховище, якщо він включає часові затримки або має зворотні зв'язки. Такі керовані стани називаються керованими станами (керованою пам'яттю) і є частиною мереж довгострокової короткочасної пам'яті (LSTM) і керованих рекурентних блоків. Рекурентні нейронні мережі теоретично є повними за Тюрінгом і можуть виконувати довільні програми для обробки довільних послідовностей вхідних даних.

Довготривала короткочасна пам'ять (LSTM) - це тип рекурентної нейронної мережі (RNN), спрямованої на вирішення проблеми зникнення градієнта, яка присутня в традиційних RNN. Її відносна нечутливість до довжини проміжків є її перевагою над іншими RNN, прихованими марковськими моделями та іншими методами навчання послідовностей. LSTM забезпечує короткочасну пам'ять для RNN, яка може тривати тисячі часових кроків, звідси і назва "довготривала короткочасна пам'ять". Вона застосовується для класифікації, обробки та прогнозування даних на основі часових рядів, таких як рукописний текст, розпізнавання мови, машинний переклад, виявлення мовної активності, керування роботами, відеоігри та охорона здоров'я.

Звичайний блок LSTM складається з комірки (cell), вхідних воріт (input gate), вихідних воріт (output gate) та воріт забування (forget gate). Комірка запам'ятовує значення протягом довільних часових інтервалів, а три ворота регулюють потік інформації до та з комірки. Ворота забування визначають, яку інформацію відкинути з попереднього стану, присвоюючи попередньому стану, порівняно з поточним вхідним сигналом, значення від 0 до 1. Округлене значення 1 означає зберегти інформацію, а значення 0 означає відкинути її. Вхідні ворота вирішують, які частини нової інформації зберегти в поточному стані, використовуючи ту саму систему, що і ворота забування. Вихідні ворота контролюють, яку інформацію з поточного стану вивести, присвоюючи їй значення від 0 до 1, враховуючи попередній та поточний стани. Вибірковий вивід релевантної інформації з поточного стану дозволяє мережі LSTM підтримувати корисні довготривалі залежності для здійснення прогнозів як у поточних, так і в майбутніх часових кроках.

Найбільш наглядним прикладом використання рекурентних нейронних мереж для виявлення аномалій трафіку є робота [6]. У цій роботі науковці розглядають використання RNN із довгою короткочасною пам’яттю (LSTM) для виявлення аномалій у мережевому трафіку. Вони показують, що LSTM RNN здатні виявляти патерни трафіку, що вказують на зловмисне використання комп'ютерних систем без необхідності в анотованих даних для навчання та без доступу до внутрішнього стану або процесів кожної машини.

Дослідження надає докази того, що безнаглядні моделі можуть виявляти трафік, що вказує на зловмисну активність, навіть коли вони навчаються на даних мережі, що не є ідеально чистими (без атак). У дослідженні також зазначається, що мережеві дані мають властивості, схожі на природну мову, оскільки комунікації між пристроями також слідують певним правилам. Вони створюють модель, яка вивчає семантичну та синтаксичну граматику цієї нової мови. Вони використовують LSTM RNN для захоплення складних взаємозв'язків та нюансів цієї мови. Потім мовна модель використовується для передбачення комунікацій між двома IP-адресами, і похибка передбачення використовується як міра

типовості або нетиповості спостережуваної комунікації. Методологія включає використання двошарових двонаправлених LSTM мереж для моделювання послідовностей потоків. Модель навчається передбачати наступні елементи послідовності на основі попередніх, що дозволяє ідентифікувати 5 аномалії в трафіку. Для оцінки продуктивності моделей використовуються метрики AUC, які показують здатність моделі перевищувати випадковий вибір.

Результати дослідження демонструють, що моделі ефективно ідентифікують аномалії в мережевому трафіку як у сценаріях з чистими даними (без атак), так і з брудними даними (з атаками). Вони показують, що навіть у випадках, коли навчальні дані містять атаки, модель все ще здатна виявляти зловмисний трафік.

Для своєї реалізації моделі з використанням LSTM RNN було вирішено працювати з даними напряму (про них детальніше буде в наступному розділі). Насамперед, числові дані були промасштабовані так, щоб їхня дисперсія була стандартною - це допомагало моделі ефективніше вчитися, оскільки всі ознаки мали однаковий масштаб - це забезпечувало рівне вагове значення кожної ознаки. Після нормалізації виконується зменшення розмірності за допомогою методу головних компонент (PCA). Це дозволило зменшити кількість вхідних змінних, видаляючи ті, що найменше варіюються, і зосереджуючи модель на найбільш значущих ознаках - тим самим зосереджуючись на головних ознаках трафіку, як-от інтенсивність активності, період підключення, довжина переданих даних, тощо. Після цього категоріальні мітки перетворюються в бінарні вектори за допомогою методу one-hot encoding, що робить можливим їх обробку нейронними мережами, які працюють з числовими даними.

Далі використовується RNN з LSTM блоками для аналізу та прогнозування мережевого трафіку. LSTM особливо ефективні для роботи з послідовностями даних, оскільки вони можуть зберігати інформацію про попередні дані, що є критично важливим для точного аналізу часових рядів та послідовних даних. Загалом інфраструктура моделі складалася з двох шарів LSTM і двох шарів викидів (dropout) для запобігання перенавчання моделі.

Модель тренувалась на великій частині даних. Оцінка моделі включала в собі валідацію на окремій підмножині даних і використання метрик, таких як середньоквадратична помилка (MSE), для вимірювання точності прогнозів моделі.

У пункті 2.1 було згадано, що нормальний трафік має доволі хаотичну натуру, у той час як аномалії, зумовлені атаками, мають впізнавані патерни. Факт хаотичності норми і послужив головною причиною проблеми цієї моделі. Незважаючи на коректно розставлені мітки “аномалія”/”не аномалія”, модель все одно вважала “візерунки” трафіку за норму - і через це нормальний трафік був передбачений з великою помилкою, що мало б статися в ситуації з аномальним. Саме тому ця модель і була виключена з фінальної реалізації.

## 2.3 Розробка моделі класифікації трафіку за допомогою числових рядів

Одним з найпоширеніших методів аналізу послідовних даних, - якими є записи про трафік, - є метод перетворення часових рядів для глибокого навчання. З початком розвитку технологій глибокого навчання, зокрема з'явленням таких структур як згорткові нейронні мережі (CNN) і рекурентні нейронні мережі (RNN), вчені змогли ефективніше обробляти послідовні та часові дані.

Згорткові нейронні мережі (Convolutional neural network**,** CNN) - це особливий вид нейронної мережі, яка автоматично вчиться видобувати важливі ознаки з даних за допомогою процесу, який називається оптимізація фільтрів або ядер. Вона використовує спеціальні методи регуляризації для зменшення кількості з'єднань між нейронами, що допомагає уникнути проблем, таких як зникнення або вибух градієнтів, які часто виникають у старіших типах нейронних мереж під час навчання. Завдяки застосуванню конволюційних фільтрів, які обробляють дані частинами, нейронна мережа може ефективно працювати з великими обсягами інформації, наприклад зображеннями, використовуючи значно меншу кількість нейронів і з'єднань, ніж у традиційних повністю з'єднаних мережах. Особливість такого підходу полягає також в тому, що він дозволяє ефективніше вловлювати складні залежності в даних на різних рівнях абстракції, що робить цей тип мережі дуже потужним інструментом у сучасному машинному навчанні. [7]

CNN, які були вперше розроблені для обробки зображень, незабаром знайшли своє застосування в аналізі часових рядів завдяки своїй здатності виявляти шаблони в дані зі спостереженнями, розташованими в часі. Це стало можливим завдяки використанню фільтрів, які "прокочуються" через дані, виявляючи важливі тренди та особливості без необхідності вручну визначати, які характеристики є важливими.

Згорткові мережі зокрема виявились корисними в аналізі часових рядів завдяки їх здатності до локальної обробки інформації та збереженню просторових взаємозв'язків у даних, що є критично важливим для часових послідовностей, де контекст і порядок подій мають велике значення. Це привело до широкого застосування CNN у фінансовому аналізі, обробці природних мов, прогнозуванні та інших задачах, де дані мають часову структуру.

Протягом останніх років з'явилися різні модифікації та вдосконалення згорткових мереж для специфічних задач обробки часових рядів, зокрема глибинні згорткові мережі та часово-згорткові мережі, які розвивають ідею згортання не тільки за просторовими, але і за часовими осями.

У контексті детекції аномального трафіку, структури, такі як 1D згорткові нейронні мережі, здатні ефективно виділяти важливі особливості з часових послідовностей даних, що робить їх ідеальними для моніторингу та аналізу мережевого трафіку.

Використання згорткових мереж у детекції аномального трафіку полягає в їхній здатності обробляти великі обсяги даних у реальному часі, виявляючи аномалії, які можуть вказувати на кібератаки, злам, або нестандартну поведінку у мережі. Ці мережі використовують фільтри для сканування даних на предмет незвичних патернів або відхилень від норми, що дозволяє ідентифікувати потенційні загрози швидше, ніж традиційні системи моніторингу.

На відміну від статистичних методів або простих порогових алгоритмів, які можуть використовувати загальні метрики або глобальні пороги, 1D CNN здатні розглядати більш складні взаємозв'язки та послідовності в даних. Це робить метод здатним розпізнавати складні мультимодальні патерни поведінки, які часто

супроводжують кібератаки, як наприклад DDoS атаки чи спроби проникнення.

Використання глибокого навчання також сприяє адаптації та самонавчанню систем безпеки, дозволяючи їм «вчитися» від нових даних та вдосконалювати свої моделі без постійного втручання з боку аналітиків. Це особливо важливо у світі кібербезпеки, де зловмисники невпинно модифікують свої методи атаки.

Таким чином, застосування згорткових нейронних мереж для аналізу часових рядів мережевого трафіку може забезпечити значне підвищення якості детекції аномалій, роблячи системи безпеки більш гнучкими, адаптивними та ефективними в боротьбі з сучасними загрозами.
Детальніше про реалізацію цього методу розказано в пункті 3.3.

**Висновки.**

У другому розділі було проведено аналіз існуючих методів вирішення задачі детекції аномалій у мережевому трафіку. Розглянуто три основні підходи: статистичні методи, методи машинного навчання та глибокого навчання. Було детально проаналізовано переваги та недоліки кожного з методів, зокрема їх ефективність, масштабованість, адаптивність до змін у мережевому трафіку та здатність зменшувати помилкові спрацьовування. Особлива увага була приділена методам глибокого навчання, таким як рекурентні нейронні мережі (RNN) та згорткові нейронні мережі (CNN), які показали високу точність у виявленні аномалій завдяки здатності обробляти часові послідовності та складні взаємозв'язки в даних.

# РОЗДІЛ 3. РЕАЛІЗАЦІЯ МОДЕЛІ КЛАСИФІКАЦІЇ ТРАФІКУ

## 3.1 Пошук та вибір датасету

Пошук і отримання найкращих для задач детекції аномалій датасетів є значною проблемою. Часто такі датасети є внутрішніми і не можуть бути поділеними через питання конфіденційності, інші ж сильно анонімізовані і не відображають сучасних тенденцій, або в них відсутні певні статистичні характеристики. Через це дослідникам доводиться вдаватися до використання датасетів, які часто є субоптимальними. Зі змінами в мережевих поведінках і вторгненнях, стає необхідним відмовитися від статичних та одноразових датасетів на користь більш динамічно генерованих датасетів, які не тільки відображають склад трафіку та вторгнень того часу, але й можуть бути модифіковані, розширені та відтворені.

Ідеальним методом для тестування та оцінки є використання реальних маркованих мережевих слідів з широким та всебічним набором вторгнень та аномальної поведінки. Проєкт "CSE-CIC-IDS2018 on AWS" [8], який є спільним зусиллям Канадського Інституту Кібербезпеки (CIC) та Установи Зв'язку та Безпеки Канади (CSE), зосереджений на детекції аномалій. Це дослідження стикається з викликами у своєму застосуванні у реальному світі через складність системи, яка потребує значного обсягу тестування, оцінки та налаштувань перед розгортанням. Для подолання цих недоліків було розроблено систематичний підхід до створення датасетів для аналізу, тестування та оцінки систем виявлення вторгнень, з особливим акцентом на мережеві аномалії.

Основна мета цього проєкту полягає в розробці систематичного підходу до створення різноманітних і всебічних контрольних датасетів для виявлення вторгнень на основі створення профілів користувачів, які містять абстрактні представлення подій і поведінок, спостережуваних у мережі. Ці профілі будуть комбіновані для генерації різноманітних датасетів, кожен з унікальним набором характеристик, що покриває частину домену оцінки.

Остаточний датасет включає сім різних сценаріїв атак: брутфорс, Heartbleed, ботнет, DoS, DDoS, веб-атаки та інфільтрацію мережі зсередини. Інфраструктура нападників включає 50 машин, а організація-жертва має 5 відділів і включає 420 машин та 30 серверів. Датасет включає захоплені мережеві трафіки та системні логи кожної машини, разом з 80 характеристиками, витягнутими з захопленого трафіку за допомогою CICFlowMeter-V3.

### 3.2 Попередня обробка даних

Для створення своєї моделі була взята частина попередньо згаданого датасету розміром 2 Гб [9]. Датасет складається з трьох видів трафіку: звичайний (Benign) і дві атаки типу Bruteforce – на порт FTP і порт SSH. Ці дві атаки надалі вважатимуться прикладом аномального трафіку.

Датасет складається з 80 колонок і близько 104 тисяч записів. Колонки цього датасету включать в себе:

- Dst Port (Destination port) - порт, на який були послані пакети. Наприклад, порт 80 призначений для HTTP, коли порт 443 - для HTTPS:
- Protocol - протокол, за яким відбувалася передача пакетів - наприклад, протоколи HTTP, FTP, SSH.
- Timestamp - часова мітка інтеракції;
- Flow Duration - тривалість потоку передачі пакетів між клієнтом і сервером;
- Tot Fwd Pkts (Total forward packets) - загальна кількість пакетів, що була надіслана від клієнта до серверу;
- Tot Bwd Pkts (Total backward packets) - загальна кількість пакетів, що була надіслана від сервера до клієнту;
- Label - мітка, яка позначає, чи це є звичайний трафік, чи аномалією (атакою). У колонці присутні усього три різні значення в цій колонці - Benign (звичайний трафік), Bruteforce-FTP і Bruteforce-SSH (атаки на порти FTP і SSH відповідно). Повний опис датасету можна побачити в додатку А.

Датасет і дані з нього були оброблені в середовищі Jupyter Notebook з використанням мови програмування Python. Для перегляду, передобробки та візуалізації даних були використані бібліотеки “Pandas”, “scikit-learn” та “matplotlib”.

Pandas - це бібліотека для аналізу та маніпулювання даними в Python. З її допомогою можна легко читати дані з різних форматів файлів, таких як CSV, Excel, SQL, JSON та інші, а також записувати дані у ці формати. Pandas дозволяє виконувати різні операції з даними, включаючи фільтрацію, групування, агрегацію, об'єднання та злиття (merge, join), перетворення типів даних, обробку пропущених значень та багато іншого.

Scikit-learn - це бібліотека, яка включає в себе широкий спектр алгоритмів машинного навчання для задач класифікації, регресії, кластеризації та зниження розмірності. Бібліотека також підтримує методи попередньої обробки даних, такі як нормалізація, імпутація пропущених значень та кодування категорійних змінних.

Matplotlib - це бібліотека для створення візуалізацій в Python. Вона дозволяє генерувати різноманітні графіки та діаграми, такі як лінійні графіки, гістограми, коробкові діаграми (box plots), розсіювання (scatter plots), гістограми щільності (density plots) та багато інших. Бібліотека також дозволяє налаштовувати майже всі аспекти графіків, включаючи кольори, стилі ліній, шрифти, позначення осей, легенди та багато іншого.

Для передобробки даних в формат, необхідний для навчання та тестування моделі, використовуємо “StandardScaler” і “PCA” з бібліотеки Scikit-learn. Ці дві функції використовуються для нормалізації числових даних та зменшення їх розмірності до одного головного компоненту, що допомагає в зменшенні шуму та виділенні основної інформації.

Далі відбувається кодування категоріальних міток. За категоріальні мітки беремо дані з колонки “Label”. Для перетворення категоріальних міток у формат, придатний для класифікаційних задач у нейронних мережах, використовуємо OneHotEncoder. Цей метод кодування перетворює кожну категоріальну мітку у

бінарний формат, де кожна мітка представлена окремим бінарним вектором.

Дані перетворюються у вид масивів по n елементів, що дозволяє створювати окремі вхідні вектори для нейронної мережі. Кожен такий масив являє собою підпослідовність початкових даних, яка використовується для аналізу часових рядів. В представленій реалізації n = 30.

Далі для кожного блоку з n міток визначається, чи є домінуючий клас, який займає більше ніж 90% всіх міток у цьому блоці. Якщо такий клас існує, блоку присвоюється мітка цього домінуючого класу. У випадку, коли домінуючий клас відсутній, блоку присвоюється нульовий вектор. Це дозволяє зосередити увагу на випадках з чітко вираженими класифікаційними мітками.

Після генерування нових міток виконується фільтрація даних, під час якої видаляються всі випадки, де не вдалося визначити домінуючий клас (нульові вектори). Ця фільтрація допомагає зосередитися на більш чітких та визначених випадках, що покращує якість класифікації.

Потім був розділено датасет на тренувальні та тестувальні виборки за допомогою функції “train_test_split” з бібліотеки scikit-learn. Ця функція випадковим чином перемішує дані і потім ділить їх на дві частини: тренувальний і тестовий набори. Це необхідно для того, щоб під час тренування моделі була можливість оцінити її продуктивність на невідомих даних, що дозволяє зрозуміти, як добре модель буде працювати на реальних даних. У даному випадку масив для тренування моделі має 27069 значень, а масив цільових міток - 6768. Такі ж самі параметри мають тестувальні масиви.

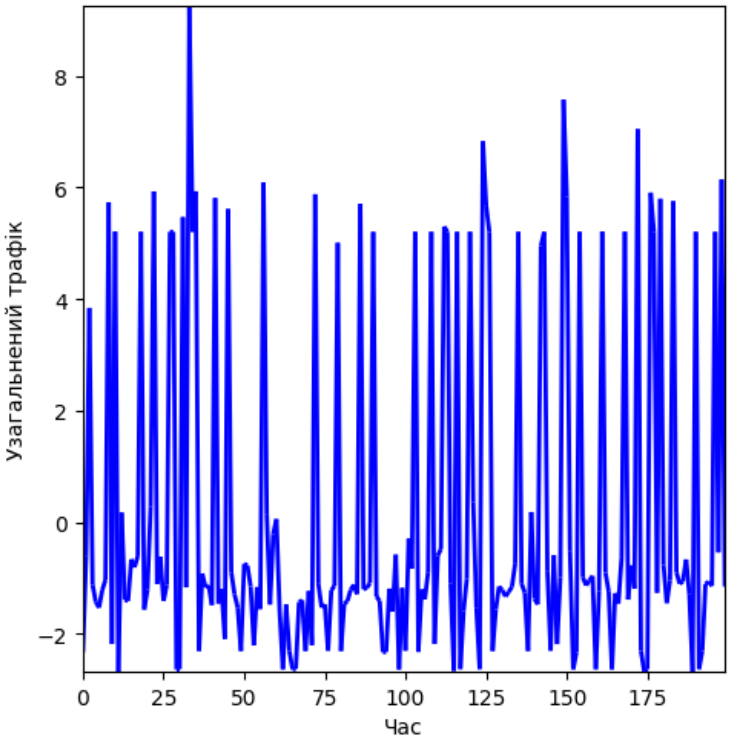


Рисунок 3.1. Графік візуалізації нормального трафіку

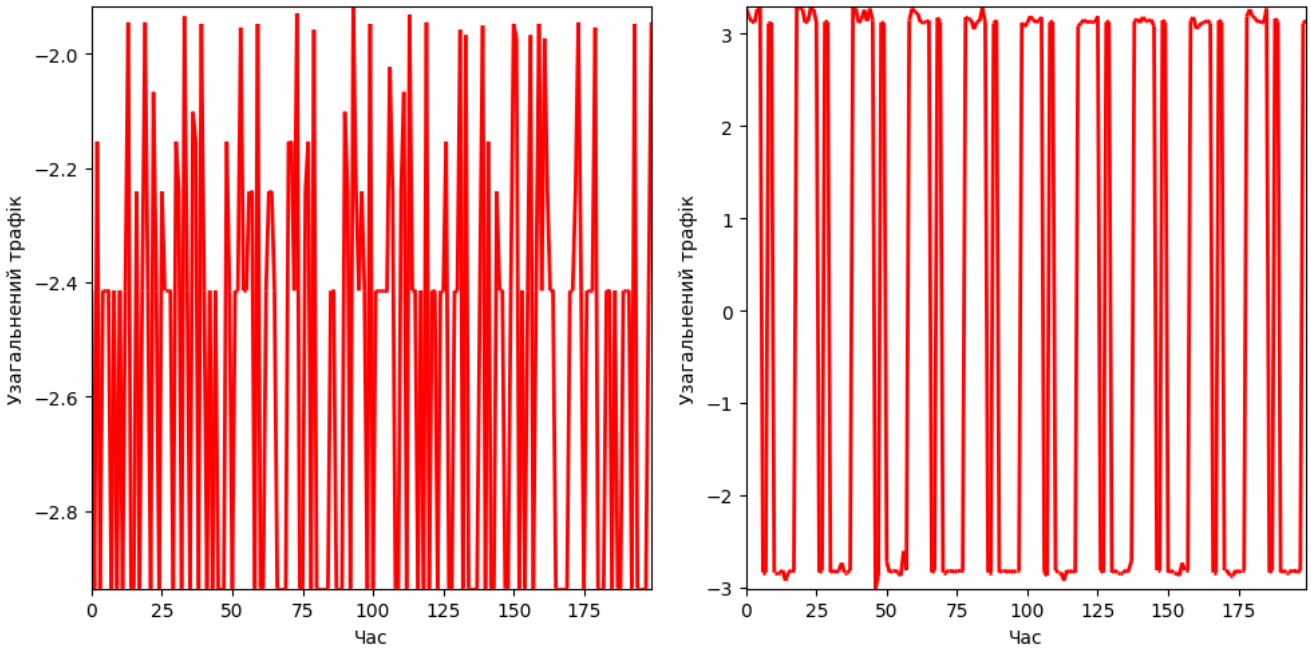


Рисунок 3.2 і Рисунок 3.3. Графіки візуалізації аномального трафіку

При обробці датасету було також створено графік за допомогою Matplotlib, який показує паттерн нормального трафіку та аномального. На Рис. 3.1 можна побачити, що нормальний трафік має доволі хаотичну структуру. На Рис. 3.2 і 3.3 показані графіки аномального трафіку. Можемо побачити, що червоні графіки

мають характерний паттерн, що нехарактерний хаотичному нормальному трафіку.

Повний код передобробки датасету можна побачити в додатку Б.

### 3.3 Реалізація моделі класифікації аномального трафіку

Для реалізації моделі класифікації було також обране середовище Jupyter Notebook та мова програмування Python. Для навчання моделі була також обрана бібліотека TensorFlow.

TensorFlow - це відкрита бібліотека для чисельних обчислень і машинного навчання, розроблена Google. Вона дозволяє створювати та навчати моделі глибокого навчання та інші види моделей машинного навчання. Основою TensorFlow є обчислювальні графи, які складаються з вузлів (операцій) і ребер (даних). Також ця бібліотека включає високорівневий API Keras, який спрощує створення та навчання моделей глибокого навчання. Keras надає інтуїтивний інтерфейс для побудови нейронних мереж, вибору оптимізаторів, функцій втрат та метрик.

Перед початком тренування задаються параметри вхідних даних - вхідна послідовність має довжину 30, а кожен елемент послідовності має одну ознаку.

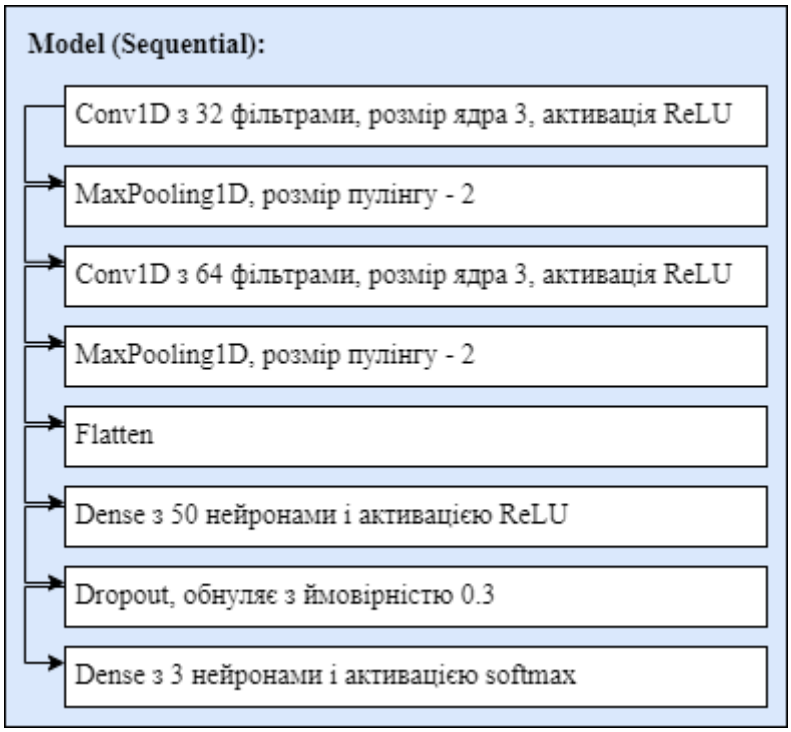


Рисунок 3.4. Діаграмне представлення архітектури моделі

Сама архітектура моделі - послідовна (Sequential) і складається з наступних шарів:

1. перший шар Conv1D. Цей шар застосовується для обробки послідовних даних. Він використовує згорткові ядра, які проходять через вхідні дані, щоб виявляти локальні патерни. Кількість фільтрів визначає, скільки різних патернів модель зможе вивчити, а розмір ядра визначає, наскільки великі ці патерни можуть бути. У даній реалізації шар має 32 фільтри з ядром розміром 3 до вхідних даних, використовуючи активацію ReLU.
2. перший шар MaxPooling1D. Він використовується для зменшення розмірності вхідних даних і зниження обчислювальної складності моделі. Шар обирає максимальне значення з кожного підмножини даних, що проходить через нього - це дозволяє виділяти найважливіші ознаки та зменшувати вплив шуму. В прелставленій архітектурі це шар максимального підбору з розміром пулінгу 2, який зменшує розмірність вихідних даних з попереднього шару.
3. другий шар Conv1D. Другий згортковий шар, який застосовує 64 фільтри з ядром розміром 3, також використовуючи активацію ReLU.
4. другий шар MaxPooling1D: Другий шар максимального підбору з розміром пулінгу 2.
5. шар Flatten. Цей шар перетворює багатовимірний вхід (наприклад, вихід з MaxPooling1D) в одномірний вектор. Це необхідно для переходу від згорткових шарів до повнозв’язних (Dense) шарів, які очікують вхід у вигляді одномірного вектора. Цей шар необхідний для об'єднання результатів згорткових операцій перед передачею їх до повнозв’язного шару.
6. перший шар Dense. Це стандартний нейронний шар, де кожен нейрон пов'язаний з кожним нейроном попереднього шару. Він використовується для навчання нелінійних комбінацій ознак і є основним компонентом багатьох нейронних мереж. Шар може мати різні функції активації, що

визначають, як нейрони будуть активуватися на основі вхідних даних. В даному випадку цей повнозв'язний шар з 50 нейронами і активацією ReLU.

7. Dropout. Він використовується для запобігання перенавчанню моделі. Під час навчання він випадково вимикає (обнуляє) певний відсоток нейронів у шарі, що запобігає моделі від надмірного підлаштування під тренувальні дані. Цей шар відсіву - з ймовірністю 0.3, що допомагає запобігти перенавчанню моделі.
8. другий шар Dense. Вихідний шар з 3 нейронами і активацією softmax, який використовується для багатокласової класифікації (3 класи).

Усі шари є частиною API Keras, яка сама по собі є підчастиною бібліотеки TensorFlow. На Рис. 3.4 також представлена діаграма архітектури даної моделі.

Далі модель компілюється з використанням оптимізатора Adam, функцією втрат categorical_crossentropy (функція кросентропії для багатокласової класифікації ) і метрики точності accuracy. Adam - це метод оптимізації, який є ефективним алгоритмом стохастичного градієнтного спуску для навчання нейронних мереж; categorical_crossentropy це функція втрат, яка часто використовується для задач класифікації, де цільові значення представлені у форматі one-hot encoding; метрика accuracy вимірює відсоток правильно класифікованих прикладів серед усіх прикладів.

Після компіляції йде процес навчання нейронної мережі. Починаючи з методу “model.fit”, модель навчається на раніше оброблених і поділених вхідних даних з відповідними цільовими мітками. Параметр “epochs” визначає кількість епох навчання, що представляє собою кількість повних проходів через весь набір даних для навчання моделі. В цьому випадку моделі достатньо 5 епох для повного навчання. Розмір батчу, встановлений за допомогою параметра “batch_size”, вказує кількість прикладів, які одночасно передаються моделі для обробки перед оновленням вагів. Використання батчів дозволяє покращити ефективність навчання і використовувати оптимізатори градієнта з кращою швидкістю. В випадку навчання

представленої моделі параметр “batch_size” дорівнює 32. Нарешті, параметр “validation_split” визначає частку даних, які будуть використовуватися для валідації моделі під час навчання. Валідація дозволяє оцінити продуктивність моделі на відокремленому наборі даних, що допомагає у контролі процесу навчання та уникненні перенавчання. В даному випадку цей параметр дорівнює 0.2.

Таким чином, представлена модель, що має в собі 9 послідовних шарів, навчається на тренувальному датасеті, що був створений раніше. Повний код моделі, її компіляції та тренування можна побачити в додатку В.

### 3.4 Валідація моделі та результати роботи

Для перевірки результатів роботи моделі на тестових даних використовуємо функцію “model.evaluate”. Ця функція виводить показники втрат і точності моделі.

Точність - це відсоток правильно класифікованих прикладів серед усіх прикладів. Вона дозволяє визначити, як добре модель працює у визначенні правильного класу для нових вхідних даних. Висока точність означає, що модель ефективно впоралася з завданням класифікації.

Втрати ж в свою чергу визначають, наскільки далеко прогнози моделі відхиляються від фактичних значень у навчальному наборі даних під час навчання. Мета полягає в тому, щоб зменшити цю величину протягом процесу навчання.

Під час процесу навчання спостерігалося зменшення втрат з 0.0201 до 0.090 та зростання точності з 0.9279 до 0.9687. Це свідчить про ефективне вдосконалення моделі. Під час валідації на даних точність стабільно трималася на рівні близько 0.9799-0.9899 - це підкреслює високу загальну точність моделі.

При тестуванні моделі з тестової вибірки було рандомним чином обрано проміжок. Задача моделі була визначити, чи є цей проміжок прикладом аномального трафіку, і якщо це так – вивести графік цього проміжку червоного кольору. Якщо трафік за результатами класифікації виявився звичайним, то графік виводиться у синьому кольорі.

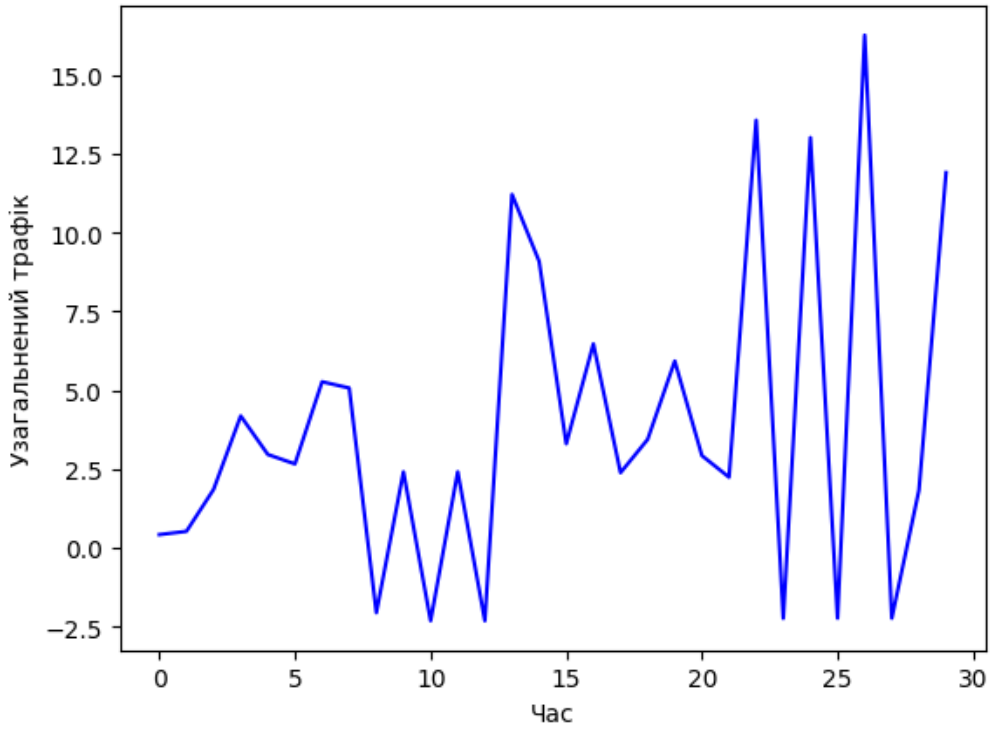


Рисунок 3.5. Приклад нормального трафіку за результатами класифікації

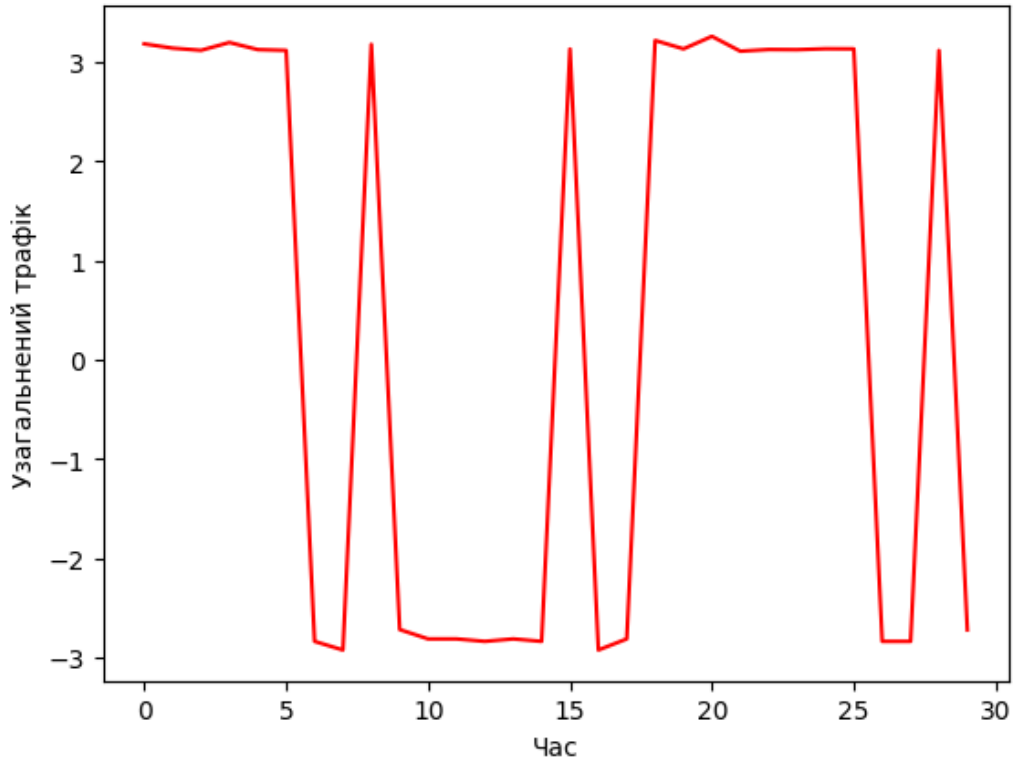


Рисунок 3.6. Приклад аномального трафіку за результатами класифікації

На Рис. 3.5 і 3.6 показано результати класифікації. Помітно, що графік на Рис. 3.5 має характерну нормальному трафіку невпорядковану структуру, коли на Рис. 3.6 проглядається патерн, що вказує на аномалію.

Кінцева оцінка на тестовому наборі даних показала втрати на рівні 0.000768 і точність 0.99985, що демонструє високу ефективність моделі у класифікації тестових даних. Також було виконано підсумовування міток у тестовому наборі, де результат [4422., 1196., 1150.] вказує на кількість прикладів для кожного з трьох класів. Ці дані відображають, що модель має високу точність і низькі втрати, роблячи її дуже ефективною для вирішення поставленої задачі.

Отже, створена модель для класифікації трафіку на нормальний та аномальний показала хороші результати при застосуванні до реальних даних.

**Висновки.**

У третьому розділі було розроблено та реалізовано модель класифікації мережевого трафіку для виявлення аномалій з використанням методів глибокого навчання. Було проведено попередню обробку даних, зокрема нормалізацію, зменшення розмірності та кодування категоріальних міток. Реалізована модель на основі згорткових нейронних мереж (1D CNN) показала високу точність у детекції аномалій трафіку. Під час тренування моделі спостерігалося значне зменшення втрат та підвищення точності. Кінцевий результат на тестових даних показав точність 0.99985, що свідчить про високу ефективність запропонованого підходу. Запропонована модель може бути використана для моніторингу мережевого трафіку в реальному часі, забезпечуючи своєчасне виявлення потенційних загроз та підвищення загальної стійкості інформаційної інфраструктури.

# ВИСНОВКИ

У даній роботі проведено детальний аналіз задачі детекції аномального мережевого трафіку з використанням сучасних методів глибокого навчання. У процесі дослідження розглянуто основні підходи до виявлення аномалій, зокрема використання рекурентних нейронних мереж (RNN) та згорткових нейронних мереж (CNN).

Під час реалізації моделі було використано датасет CSE-CIC-IDS2018, який включає різноманітні види аномального трафіку. Датасет був ретельно підготовлений: проведено нормалізацію даних, зменшення розмірності за допомогою PCA, а також кодування категоріальних міток за допомогою one-hot encoding. Це дозволило створити вхідні дані для моделі у вигляді часових рядів.

Реалізована модель класифікації на основі згорткових нейронних мереж (1D CNN) показала високу точність у детекції аномалій трафіку. Під час тренування моделі спостерігалося значне зменшення втрат та підвищення точності. Кінцевий результат на тестових даних показав точність 0.99985, що свідчить про високу ефективність запропонованого підходу.

З розвитком інформаційних технологій збільшується загроза безпеці мережевих систем - саме тому детекція аномального трафіку є критично важливою задачею для забезпечення кібербезпеки. Використання RNN та CNN дозволяє ефективно аналізувати мережевий трафік та виявляти аномалії завдяки їхній здатності обробляти часові послідовності та складні взаємозв'язки в даних. Саме CNN показали високу точність у виявленні аномалій, зокрема завдяки їхній здатності виявляти локальні патерни в даних та зменшувати вплив шуму. Реалізована в цій роботі модель може бути використана для моніторингу мережевого трафіку в реальному часі, забезпечуючи своєчасне виявлення потенційних загроз та підвищення загальної стійкості інформаційної інфраструктури.

Таким чином, ця дипломна робота демонструє можливості сучасних методів глибокого навчання у вирішенні задач детекції аномального трафіку та підкреслює

необхідність подальших досліджень для вдосконалення існуючих методів та розробки нових підходів у сфері кібербезпеки.

# ПЕРЕЛІК ВИКОРИСТАНИХ ДЖЕРЕЛ


1. ISO/IEC 7498-1:1994 Information technology — Open Systems Interconnection — Basic Reference Model: The Basic Model. June 1999.
2. Spark-Based Port and Net Scan Detection / Antonia Affinito, Alessio Botta, Luigi Gallo, Mauro Garofalo, Giorgio Ventre – 2019. https://doi.org/10.48550/arXiv.1806.11047
3. Multivariate Network Traffic Analysis using Clustered Patterns / Jinoh Kim, Alex Sim, Brian Tierney, Sang Suh, Ikkyun Kim – 2018. https://link.springer.com/article/10.1007/s00607-018-0619-4
4. Intrusion detection: a brief history and overview / R. A. Kemmerer and G. Vigna // *Computer*, vol. 35, no. 4, pp. supl27-supl30, April 2002, doi: 10.1109/MC.2002.1012428.
5. "State-of-the-art in artificial neural network applications: A survey" / Abiodun, Oludare Isaac; Jantan, Aman; Omolara, Abiodun Esther; Dada, Kemi Victoria; Mohamed, Nachaat Abdelatif; Arshad, Humaira // *Heliyon*. **4** (11) – 2018. doi:10.1016/j.heliyon.2018.e00938. ISSN 2405-8440.
6. Network Traffic Anomaly Detection Using Recurrent Neural Networks. / Benjamin J. Radford, Leonardo M. Apolonio, Antonio J. Trias, Jim A. Simpson – 2018. - https://doi.org/10.48550/arXiv.1803.10769
7. Guide to convolutional neural networks : a practical application to traffic-sign detection and classification / Habibi, Aghdam, Hamed Heravi, Elnaz Jahani. Cham, Switzerland – 2017. ISBN 9783319575490.
8. CSE-CIC-IDS2018 on AWS: https://www.unb.ca/cic/datasets/ids-2018.html
9. IDS 2018 Intrusion CSVs (CSE-CIC-IDS2018) - https://www.kaggle.com/datasets/solarmainframe/ids-intrusion-csv/data

# ДОДАТКИ

## Додаток А. Повний опис датасету CSE-CIC-IDS2018

1. Dst Port: Порт, на який спрямований трафік (цільовий порт).
2. Protocol: Протокол, що використовується для зв'язку (наприклад, TCP, UDP).
3. Timestamp: Часовий штамп, коли пакет був зафіксований.
4. Flow Duration: Тривалість потоку у мікросекундах.
5. Tot Fwd Pkts: Загальна кількість пакетів, надісланих вперед.
6. Tot Bwd Pkts: Загальна кількість пакетів, надісланих назад.
7. TotLen Fwd Pkts: Загальна довжина всіх пакетів, надісланих вперед.
8. TotLen Bwd Pkts: Загальна довжина всіх пакетів, надісланих назад.
9. Fwd Pkt Len Max: Максимальна довжина пакета, надісланого вперед.
10. Fwd Pkt Len Min: Мінімальна довжина пакета, надісланого вперед.
11. Fwd Pkt Len Mean: Середня довжина пакета, надісланого вперед.
12. Fwd Pkt Len Std: Стандартне відхилення довжини пакетів, надісланих вперед.
13. Bwd Pkt Len Max: Максимальна довжина пакета, надісланого назад.
14. Bwd Pkt Len Min: Мінімальна довжина пакета, надісланого назад.
15. Bwd Pkt Len Mean: Середня довжина пакета, надісланого назад.
16. Bwd Pkt Len Std: Стандартне відхилення довжини пакетів, надісланих назад.
17. Flow Byts/s: Кількість байтів на секунду для потоку.
18. Flow Pkts/s: Кількість пакетів на секунду для потоку.
19. Flow IAT Mean: Середній міжприхідний час (Inter Arrival Time) для потоку.
20. Flow IAT Std: Стандартне відхилення міжприхідного часу для потоку.
21. Flow IAT Max: Максимальний міжприхідний час для потоку.
22. Flow IAT Min: Мінімальний міжприхідний час для потоку.
23. Fwd IAT Tot: Загальний міжприхідний час для пакетів, надісланих вперед.
24. Fwd IAT Mean: Середній міжприхідний час для пакетів, надісланих вперед.
25. Fwd IAT Std: Стандартне відхилення міжприхідного часу для пакетів, надісланих вперед.
26. Fwd IAT Max: Максимальний міжприхідний час для пакетів, надісланих вперед.
27. Fwd IAT Min: Мінімальний міжприхідний час для пакетів, надісланих вперед.
28. Bwd IAT Tot: Загальний міжприхідний час для пакетів, надісланих назад.
29. Bwd IAT Mean: Середній міжприхідний час для пакетів, надісланих назад.
30. Bwd IAT Std: Стандартне відхилення міжприхідного часу для пакетів, надісланих назад.
31. Bwd IAT Max: Максимальний міжприхідний час для пакетів, надісланих назад.
32. Bwd IAT Min: Мінімальний міжприхідний час для пакетів, надісланих назад.

33. Fwd PSH Flags: Кількість встановлених PSH-флагів у пакетах, надісланих вперед.
34. Bwd PSH Flags: Кількість встановлених PSH-флагів у пакетах, надісланих назад.
35. Fwd URG Flags: Кількість встановлених URG-флагів у пакетах, надісланих вперед.
36. Bwd URG Flags: Кількість встановлених URG-флагів у пакетах, надісланих назад.
37. Fwd Header Len: Загальна довжина заголовків у пакетах, надісланих вперед.
38. Bwd Header Len: Загальна довжина заголовків у пакетах, надісланих назад.
39. Fwd Pkts/s: Кількість пакетів на секунду, надісланих вперед.
40. Bwd Pkts/s: Кількість пакетів на секунду, надісланих назад.
41. Pkt Len Min: Мінімальна довжина пакета в потоці.
42. Pkt Len Max: Максимальна довжина пакета в потоці.
43. Pkt Len Mean: Середня довжина пакета в потоці.
44. Pkt Len Std: Стандартне відхилення довжини пакетів в потоці.
45. Pkt Len Var: Варіація довжини пакетів в потоці.
46. FIN Flag Cnt: Кількість встановлених FIN-флагів у потоці.
47. SYN Flag Cnt: Кількість встановлених SYN-флагів у потоці.
48. RST Flag Cnt: Кількість встановлених RST-флагів у потоці.
49. PSH Flag Cnt: Кількість встановлених PSH-флагів у потоці.
50. ACK Flag Cnt: Кількість встановлених ACK-флагів у потоці.
51. URG Flag Cnt: Кількість встановлених URG-флагів у потоці.
52. CWE Flag Count: Кількість встановлених CWE-флагів у потоці.
53. ECE Flag Cnt: Кількість встановлених ECE-флагів у потоці.
54. Down/Up Ratio: Відношення кількості байтів вниз до кількості байтів вгору.
55. Pkt Size Avg: Середній розмір пакета в потоці.
56. Fwd Seg Size Avg: Середній розмір сегмента, надісланого вперед.
57. Bwd Seg Size Avg: Середній розмір сегмента, надісланого назад.
58. Fwd Byts/b Avg: Середня кількість байтів на пакет, надісланий вперед.
59. Fwd Pkts/b Avg: Середня кількість пакетів на пакет, надісланий вперед.
60. Fwd Blk Rate Avg: Середня швидкість блоків на пакет, надісланий вперед.
61. Bwd Byts/b Avg: Середня кількість байтів на пакет, надісланий назад.
62. Bwd Pkts/b Avg: Середня кількість пакетів на пакет, надісланий назад.
63. Bwd Blk Rate Avg: Середня швидкість блоків на пакет, надісланий назад.
64. Subflow Fwd Pkts: Кількість підпотоків пакетів, надісланих вперед.
65. Subflow Fwd Byts: Кількість підпотоків байтів, надісланих вперед.
66. Subflow Bwd Pkts: Кількість підпотоків пакетів, надісланих назад.
67. Subflow Bwd Byts: Кількість підпотоків байтів, надісланих назад.
68. Init Fwd Win Byts: Початкова кількість байтів у вікні, надісланих вперед.
69. Init Bwd Win Byts: Початкова кількість байтів у вікні, надісланих назад.
70. Fwd Act Data Pkts: Кількість активних пакетів даних, надісланих вперед.
71. Fwd Seg Size Min: Мінімальний розмір сегмента, надісланого вперед.
72. Active Mean: Середній час активності потоку.
73. Active Std: Стандартне відхилення часу активності потоку.

74. Active Max: Максимальний час активності потоку.
75. Active Min: Мінімальний час активності потоку.
76. Idle Mean: Середній час простою потоку.
77. Idle Std: Стандартне відхилення часу простою потоку.
78. Idle Max: Максимальний час простою потоку.
79. Idle Min: Мінімальний час простою потоку.

80. Label - мітка, яка позначає, чи це є звичайний трафік, чи аномалією (атакою)

## Додаток Б. Код підготовки та обробки датасету до використання

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
from sklearn.decomposition import PCA
from sklearn.preprocessing import OneHotEncoder
from sklearn.preprocessing import StandardScaler
from matplotlib.collections import LineCollection

df = pd.read_csv('.../data.csv')
df.replace([np.inf, -np.inf], np.nan, inplace=True)
df.dropna(inplace=True)
df.reset_index(drop=True, inplace=True)

scaler = StandardScaler()
X = scaler.fit_transform(df_num)
pca = PCA(n_components=1)
X = pca.fit_transform(X)
one_hot = OneHotEncoder()
y = one_hot.fit_transform(df_labels).toarray().astype(int)
from sklearn.model_selection import train_test_split
n = 30
length = (X.shape[0] // n) * n
X_trimmed = X[:length]
y_trimmed = y[:length]
X_new = X_trimmed.reshape(-1, n)
y_new = []
for i in range(0, len(y_trimmed), n):
    sub_labels = y_trimmed[i:i + n]
  label_sums = sub_labels.sum(axis=0)
  dominant_class = label_sums >= (n * 0.9)
  if dominant_class.any():
        y_new.append(np.eye(3)[np.argmax(label_sums)])
  else:
        y_new.append(np.array([0, 0, 0]))
y_new = np.array(y_new)
non_zero_mask = ~(y_new == [0, 0, 0]).all(axis=1)
X_new = X_new[non_zero_mask]
y_new = y_new[non_zero_mask]
X_train, X_test, y_train, y_test = train_test_split(X_new, y_new, test_size=0.2,
random_state=42)
X_train.shape, X_test.shape, y_train.shape, y_test.shape
```

## Додаток В. Код тренування та тестування моделі

```
import tensorflow as tf
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Conv1D, MaxPooling1D, Flatten, Dense,
Dropout

input_length = 30
n_features = 1

model = Sequential([
    Conv1D(filters=32, kernel_size=3, activation='relu', input_shape=(input_length,
n_features)),
    MaxPooling1D(pool_size=2),
    Conv1D(filters=64, kernel_size=3, activation='relu'),
    MaxPooling1D(pool_size=2),
    Flatten(),
    Dense(50, activation='relu'),
    Dropout(0.3),
    Dense(3, activation='softmax')
])

model.compile(optimizer='adam', loss='categorical_crossentropy',
metrics=['accuracy'])

model.compile(optimizer='adam', loss='categorical_crossentropy',
metrics=['accuracy'])
model.fit(
    X_new, y_new,
    epochs=5,
    batch_size=32,
    validation_split=0.2
)

test_loss, test_accuracy = model.evaluate(X_test, y_test)

print(f"Test Loss: {test_loss}")
print(f"Test Accuracy: {test_accuracy}")
```