Міністерство освіти і науки України
Харківський національний університет радіоелектроніки

Факультет _____Комп’ютерної інженерії та управління_____
(повна назва)

Кафедра _____Комп’ютерних інтелектуальних технологій та систем_____
(повна назва)

КВАЛІФІКАЦІЙНА РОБОТА
Пояснювальна записка

рівень вищої освіти _____другий (магістерський)_____

Виявлення загроз в комп’ютерних мережах на основі нейромережевого підходу та штучних імунних систем
(тема)

Виконав:
здобувач ___2___ року навчання,
групи ___КІТм-23-1_____
_____Торубара О.А._____
(прізвище, ініціали)

Спеціальність ____123 Комп'ютерна_____
_____інженерія_____
(код і повна назва спеціальності)
Тип програми ____освітньо-професійна____
(освітньо-професійна або освітньо-наукова)
Освітня програма ___Комп’ютерні _____
інтелектуальні технології_____
(повна назва освітньої програми)

Керівник _____проф. Руденко О. Г._____
(посада, прізвище, ініціали)

Допускається до захисту

Зав. кафедри _____ __Руденко О. Г.___
(підпис) (прізвище, ініціали)

2024 р.

Харківський національний університет радіоелектроніки

Факультет Комп'ютерної інженерії та управління

Кафедра Комп'ютерних інтелектуальних технологій та систем

Рівень вищої освіти другий (магістерський)

Спеціальність 123 Комп'ютерна інженерія
(код і повна назва)

Тип програми освітньо-професійна
(освітньо-професійна або освітньо-наукова)

Освітня програма Комп'ютерні інтелектуальні технології
(повна назва)

ЗАТВЕРДЖУЮ:

Зав. кафедри _____
(підпис)

«____»_____ 2025 р.

ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ РОБОТУ

здобувачеві Торубарі Олегу Андрійовичу
(прізвище, ім'я, по батькові)

1. Тема роботи Виявлення загроз в комп'ютерних мережах на основі нейромережевого підходу та штучних імунних систем

затверджена наказом університету від 28 жовтня 2024 р. № 1156Ст

2. Термін подання здобувачем роботи до екзаменаційної комісії ___ _____ 20__ р.

3. Вихідні дані до роботи 1) Статистичні дані про мережеві сесії 2) методики попередньої обробки даних 3) Виокремлення ознак 4) засоби штучного інтелекту 5) архітектура нейронних мереж

4. Перелік питань, що потрібно опрацювати в роботі
1) аналіз предметної області;
2) аналіз доцільності дослідження;
3) вибір технологій розробки та інструментальних засобів;
4) розробка рішення
5) аналіз отриманних результатів
6) висновки

5. Перелік графічного матеріалу із зазначенням креслеників, схем, плакатів, комп'ютерних ілюстрацій (п.5 включається до завдання за рішенням випускової кафедри) _____
10 слайдів_ _____
_____
_____

6. Консультанти розділів роботи (п.6 включається до завдання за наявності консультантів згідно з наказом, зазначеним у п.1)

Найменування розділу	Консультант (посада, прізвище, ім'я, по батькові)	Позначка консультанта про виконання розділу	
		підпис	дата

КАЛЕНДАРНИЙ ПЛАН

№	Назва етапів роботи	Терміни виконання етапів роботи	Примітка
1	Видача та узгодження теми проєкту	28.10.2024	Виконано
2	Огляд стану проблеми та постановка задачі	30.11-29.11.2024	Виконано
3	Аналіз літератури за напрямком дослідження	30.11- 07.12.2024	Виконано
4	Аналіз існуючих підходів до виявлення загроз в комп'ютерних мережах	08.12-10.12.2024	Виконано
5	Аналіз обраних архітектур нейронних мереж	11.12-13.12.2024	Виконано
6	Аналіз та підготовка даних для навчання	14.12-16.12.2024	Виконано
7	Розробка рішень виявлення загроз в комп'ютерних мережах	17.12-21.12.2024	Виконано
8	Оцінка ефективності запропонованих рішень	22.12-25.12.2024	Виконано
9	Оцінка ефективностікомп'ютерних мережах	26.12-29.12.2024	Виконано
10	Підготовка ПЗ та презентаційного матеріалу	30.12-23.01.2025	Виконано
1	Подання до ЕК	23.01.2025	Виконано
12	Захист проєкту	24.01.2025	

Дата видачі завдання «28» жовтня 2024 р.

Здобувач _____
(підпис)

Керівник роботи _____ _____проф. Руденко О. Г.___
(підпис) (посада, прізвище, ініціали)

РЕФЕРАТ

Пояснювальна записка кваліфікаційної роботи: 110 с., 8 табл., 32 рис., 3 дод., та 16 джерел.

НЕЙРОННІ МЕРЕЖІ, ЗАГРОЗИ, ТРАФІК, КОМП'ЮТЕРНІ МЕРЕЖІ, INTRUSION DETECTION SYSTEM, FIREWALL, МОДЕЛЬ, ПРОГРАМНЕ ЗАБЕЗПЕЧЕННЯ, ПРОГРАМНО-АПАРАТНИЙ КОМПЛЕКС

Метою роботи є дослідження методів виявлення загроз в компю'терних мережах використовуючи нейромережевий підхід.

Методи дослідження - проведення аналізу літератури, попередньо проведених робіт на цю тему та практична реалізація проекту.

Об'єктом дослідження є методи виявлення загроз в комп'терних мережах на основі нейромережевого підходу та методів машинного навчання.

Предметом дослідження є алгоритми для класифікації та визначення мережевих загроз, що засновані на виділенні критеріїв властивостей для глибокої нейронної мережі.

ABSTRACT

Master’s thesis: 110 pages, 8 tables, 32 figures, 3 appendix, 16 sourses.

NEURAL NETWORKS, THREATS, TRAFFIC, COMPUTER NETWORKS, INTRUSION DETECTION SYSTEM, FIREWALL, MODEL, SOFTWARE, HARDWARE AND SOFTWARE COMPLEX

The purpose of the study is to create a model of a program for detecting threats in computer networks.

Research methods - analysis of literature, previous works on this topic and practical implementation of the project.

The object of research is the methods of detecting threats in computer networks based on neural network approach and machine learning methods.

The subject of the study is algorithms for classifying and detection network threats based on the selection of property criteria for a deep neural network.

Міністерство освіти і науки України
Харківський національний університет радіоелектроніки

Факультет Комп'ютерної інженерії та управління
Кафедра Комп'ютерних інтелектуальних технологій та систем

АНОТАЦІЯ

КВАЛІФІКАЦІЙНОЇ РОБОТИ

рівень вищої освіти другий (магістерський)

Виявлення загроз в комп'ютерних мережах на основі нейромережевого підходу та штучних імунних систем

(тема)

Виконав:
здобувач 2 курсу, групи КІТм-23-1
Торубара О. А.
(прізвище, ініціали)

Спеціальність 123 Комп'ютерна інженерія
Тип програми освітньо-професійна
Освітня програма Комп'ютерні інтелектуальні технології
Керівник проф. Руденко О. Г.
(посада, прізвище, ініціали)

2024 р.

Торубара О. А. Виявлення загроз в комп’ютерних мережах на основі нейромережевого підходу та штучних імунних систем

У магістерській кваліфікаційній роботі вирішено актуальну задачу виявлення кіберзагроз в комп’ютерних мережах.

Метою кваліфікаційної роботи є проведення дослідження перспективності використання та впровадження нових методів для виявлення кіберзагроз в трафіку в комп’ютерних мережах використовуючи нейромережевий підхід та штучні імунні системи, а також практичне застосування отриманих знань для навчання нейронних мереж різних архітектур з подальшим порівнянняльним аналізом отриманих результатів.

Об’єктом дослідження для цієї роботи є вивчення області комп’ютерних інтелектуальних технологій, аналізу даних, нейронних мереж та машинного навчання, їх класифікацій, властивостей, методів навчання та оптимізації. Оскільки нейронні мережі демонструють позитивні результати в здатноті до узагальнення та виділення закономірностей при прийнятті решення на основі структурованих, графічних або неструктурованих даних — вважається доцільним проведення дослідження з метою аналізу перспектив та доцільності застосування нейромереж для аналізу та виявлення загроз в мережевому трафіку.

Предметом дослідження у даній роботі є методи аналізу даних, а також побудови, навчання та оптимізації нейронних мереж для виявлення кіберзагроз. Розкриваючи саме цю тему, можна оцінити потенціал застосування алгоритмів машинного навчання та нейромереж в сфері кібербезпеки та подальшої інтеграції в сучасні рішення в цій сфера, включно з програмним забезпечення яке працює в мережі в пасивному режими для моніторингу та програмно-апаратними комплексами, що є комплексними рішеннями зпроектовані та оптимізовані його компоненти мають найкращим чином працювати в симбіозі для досягнення оптимальної продуктивності при виконанні подібними системами своїх задач — аналізу та виявлення кіберзагроз в комп’ютерних мережах, в випадку кваліфікаційної роботи.

Кібербезпека є однією з найбільш динамічно розвиваючихся сфер сучасності, що зумовлено стрімкою цифровізацією практично всіх аспектів повсякденного життя. Сучасний світ, зокрема, спостерігає активне впровадження цифрових технологій у таких галузях, як документообіг, дистанційна робота, фінансові операції, управління бізнес-процесами тощо. Домінуюча частка цих процесів залежить від функціонування комп'ютерних мереж, які забезпечують можливість у реальному часі здійснювати транзакції, отримувати доступ до персональних даних, керувати системами й виконувати багато інших операцій.

Комп'ютерні мережі стали фундаментальною складовою діяльності сучасних організацій, освітніх установ, державного управління та приватного життя. Проте разом із розширенням можливостей цифрових технологій збільшується й кількість кіберзагроз, що вимагає розробки нових підходів до захисту інформаційних систем. Щодня кожен з вузлів всесвітньої мережі перебуває під потенційною загрозою атак та моніторингом з боку зловмисників. Зростання складності атак, таких як фішинг, шкідливе програмне забезпечення, атаки типу «відмова в обслуговуванні» (DDoS), робить завдання забезпечення безпеки надзвичайно актуальним. Виявлення, аналіз та нейтралізація загроз стають критичними елементами для захисту даних, збереження конфіденційності інформації та гарантування безперебійного функціонування систем.

Забезпечення надійного захисту інформаційних мереж передбачає створення комплексної системи безпеки, яка включає розробку політик, налаштування правил доступу, регулярне проведення аудиту безпеки, моніторинг інцидентів та своєчасне оновлення технологічних рішень. Важливо зазначити, що жодна система не може бути захищеною «за замовчуванням», тому підтримка кібербезпеки є безперервним процесом.

Актуальність тематики кібербезпеки постійно зростає через збільшення масштабу атак, що орієнтовані на порушення роботи критично важливих інфраструктур, викрадення конфіденційних даних та завдання фінансових

збитків. Це зумовлює необхідність вдосконалення існуючих підходів до захисту інформації та розробки інноваційних методів виявлення загроз. Впровадження нових технологій та стратегій безпеки сприяє мінімізації ризиків, забезпечує стабільність функціонування систем і дозволяє знизити ймовірність негативних наслідків, пов’язаних із витоками даних чи збоями в роботі.

Таким чином, дослідження у сфері виявлення загроз і розробка ефективних технологій захисту є важливим фактором для збереження інформаційної безпеки в умовах стрімкого розвитку цифрового середовища.

Аналіз у реальному часі за обраною темою дослідження здійснювався із використанням мови програмування Python та спеціалізованих бібліотек, призначених для інтелектуальної обробки даних, машинного навчання та високоякісної візуалізації результатів.

Кілька десятиліть тому людство зіштовхнулося з викликом управління величезними потоками інформації, які постійно збільшуються в обсязі та складності. Основною ідеєю, яка лягла в основу сучасних підходів до обробки інформації, стало створення систем, здатних до самонавчання. Це дало змогу «навчити» електронні машини виконувати складні аналітичні задачі, які раніше були можливими лише за участі людини. Штучний інтелект, у поєднанні з методами машинного навчання та глибокими нейронними мережами, дозволив досягти якісно нового рівня автоматизації та інтелектуалізації процесів у різних сферах. Сьогодні штучний інтелект широко застосовується в економіці для прогнозування ринкових тенденцій і оптимізації бізнес-процесів; у фінансах для управління ризиками та виявлення шахрайства; в кліматології для моделювання змін клімату; у спорті для аналізу ефективності тренувань та покращення стратегії змагань; в медицині для діагностики захворювань і розробки індивідуальних методів лікування. Не менш значущими є його досягнення у сфері електроніки, де штучний інтелект використовується для створення «розумних» пристроїв і систем автоматизації, що полегшують життя мільйонам людей.

Термін «виявлення» визначається як процес проведення аналізу наявних даних, використовуючи попередньо набуті знання з метою дійти висновку відносно характеру даних, що надійшли для аналізу.

У кваліфікаційній роботі був проведений аналіз нейронних мереж, процес навчання нейронної мережі та застосування їх на практиці для вирішення задачі бінарної класифікації, що описується в проекті. Нейронна мережа (НМ) – це математична модель, побудована на принципах організації та функціонування біологічних нейронних мереж – нейронів. Для ефективної обробки даних, НМ навчають на заздалегідь підготовлених даних використовуючи спеціалізовані алгоритми оптимізації цільової функції. Математично такі моделі описуються множинами нейронів, множинами зв’язків між ним. Розглянуті в роботі нейромережі є мережами прямого поширення, тобто поширення сигналів/вхідних даних відбувається шляхом прямого поширення від вхідного шару нейронів до вихідного через приховані шари. На практиці було проведено навчання нейронної мережі, яка побудована за архітектурою «багатошаровий перцептрон» (MLP) та «згорткова нейронна мережа» (CNN).

Дослідження показує, що нейронні мережі здатні навчатися на структурованих та анотованих даних, але слід враховувати, що результати їхньої роботи часто важко піддаються інтерпретації. Не завжди зрозуміло, чому нейромережа прийняла те чи інше рішення. Це обмеження підкреслює, що навіть найсучасніші нейронні мережі не можуть повністю замінити людину, особливо в умовах динамічного середовища, яке постійно змінюється через появу нових додатків, протоколів та загроз. У таких умовах досвід, експертний аналіз, знання сучасних технологій та розуміння тенденцій залишаються ключовими чинниками для ефективного аналізу кризових ситуацій та прийняття рішень у відповідь на інциденти, що регулярно трапляються в комп'ютерних мережах.

Ця робота може стати корисною для розв’язання завдань виявлення загроз у комп’ютерних мережах, оскільки людський фактор досі відіграє важливу роль в аналізі інцидентів. Нейронні мережі можуть суттєво

трансформувати роботу спеціалістів із кібербезпеки, допомагаючи ефективніше аналізувати мережевий трафік і виявляти потенційні загрози.

Ключові слова: НЕЙРОННІ МЕРЕЖІ, МАШИННЕ НАВЧАННЯ, КІБЕРБЕЗПЕКА, ВИЯВЛЕННЯ ЗАГРОЗ, PYTHON, MLP, CNN, КІБЕРЗАГРОЗИ

ПЕРЕЛІК ПУБЛІКАЦІЙ ЗДОБУВАЧА ЗА ТЕМОЮ РОБОТИ

1. Торубара О. А., Сердюк Н. М. Model of Threat Analysis and Detection System in Computer Networks Based on the Danger Theory of Artificial Immune Systems //Матеріали 12-ї Міжнародної науково-технічної конференції “Інформаційні системи та технології”. 28 листопада 2023, м. Харків, Україна
2. Торубара О. А., Руденко О. Г. Model of Threat Analysis and Detection System in Computer Networks Based using Artificial Intelligence approach //Матеріали 13-ї Міжнародної науково-технічної конференції “Інформаційні системи та технології”. 28 листопада 2024, м. Харків, Україна

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СИМВОЛІВ, ОДИНИЦЬ, СКОРОЧЕНЬ І ТЕРМІНІВ

ШІ — штучний інтелект.

ШІС — штучні імунні системи.

MLP – (англ. Multi Layer Perceptron) – багатошаровий перцептрон.

CNN – (англ. Convolutional Neural Network) – згорткова нейронна мережа.

Трафік — потік даних, що проходить комп’ютерною мережею від одного комп’ютера до іншого.

Кіберзагрози — будь-які обставини або події, що можуть бути причиною порушення політики безпеки інформації і/або нанесення збитків автоматизованій системі.

ВСТУП

Сфера кібербезпеки є однією з сфер, що найбільш стрімко розвиваються в 21-му столітті. Така тенденція спостерігається здебільшого через зростання попиту на цифровізацію великої кількості аспектів нашого повсякденного життя, таких як: документообіг, перехід на онлайн формат роботи великої кількості співробітників компаній, цифровізація банківських операцій тощо. Невід’ємною частиною для забезпечення функціонування подібних сервісів є комп’ютерні мережі, за допомогою яких користувачі по всьому світу в реальному часі можуть проводити транзакції, отримати доступ до свої персональних документів тощо.

Комп’ютерні мережі стали невід'ємною частиною повсякденного життя, бізнесу, освіти та інших сфер діяльності. Разом з тим, зростання кількості та складності мережевих загроз створює серйозні виклики для забезпечення безпеки інформаційних систем. Виявлення загроз в комп’ютерних мережах є ключовим аспектом для захисту даних, збереження конфіденційності та забезпечення безперебійного функціонування систем.

Дані фактори спричиняють зростання потреби в надійному захисті, оскільки жодна система чи мережа не є захищено за замовчуванням і потребує ретельно вибудованої мережевої безпеки, а також потребує постійної підтримки, оновлення політик безпеки, правил та безпекового аудиту.

Актуальність даної теми зумовлена постійним збільшенням обсягу та різноманітності атак на комп’ютерні системи, що потребує вдосконалення існуючих і розробки нових методів виявлення загроз. Вивчення та впровадження ефективних технологій захисту дозволяє зменшити ризики витоку інформації, фінансових втрат та порушення функціонування критично важливих інфраструктур.

1 АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ

1.1 Комп’ютерні мережі

В сучасному світі кібербезпека є ледь не основним компонентом успішного функціонування будь-якого підприємства, державних установ та й усіх організацій, які тим чи іншим чином зберігають, оброблюють дані.

Оскільки в сучасному світі важко уявити організацію, яка не веде принаймні облік співробітників, бухгалтерії в електронному вигляді. Отже завдання захисту в комп’ютерних мережах постає перед будь-якою організацією на певному етапі її зростання.

Загально визнаним є використання спеціалізованого програмного забезпечення та/або спеціалізованих програмно-апаратних комплексів для виконання функцій фільтрації мережевого трафіку, перевірки трафіку на відповідність до заданого адміністраторами набору правил тощо, блокуванні загроз, детекції та нотифікації адміністратора про можливу загрозу тощо.

Різні системи спеціалізуються на різних аспектах та функціонують на різних рівням мережевої моделі. Розглянемо модель сучасної комп’ютерної мережі, її особливості, набір протоколів та закони за якими відбувається обмін інформацією в сучасних мережах.

Першою спробою стандартизації комп’ютерних мереж (далі мережі) була мережева модель OSI (наведено на рисунку 1.1). Дана модель являла собою семи рівневий стек, кожен з рівнів якого декларував відведені йому функції та протоколи для виконання цих функції. Модель була запропонована в кінці 1970х років міжнародною організацією зі стандартизації (ISO).[1] Метою було стандартизувати роботу та обмін повідомлень в мережах того часу. Модель визначала сім рівнів абстракції які охоплювали увесь процес надсилання та отримані пакетів від одного хоста до іншого починаючи з рівня додатку (сьомий рівень) де оперує користувач і

закінчуючи фізичним рівнем (перший рівень) на якому дані передаються через фізичне середовище.

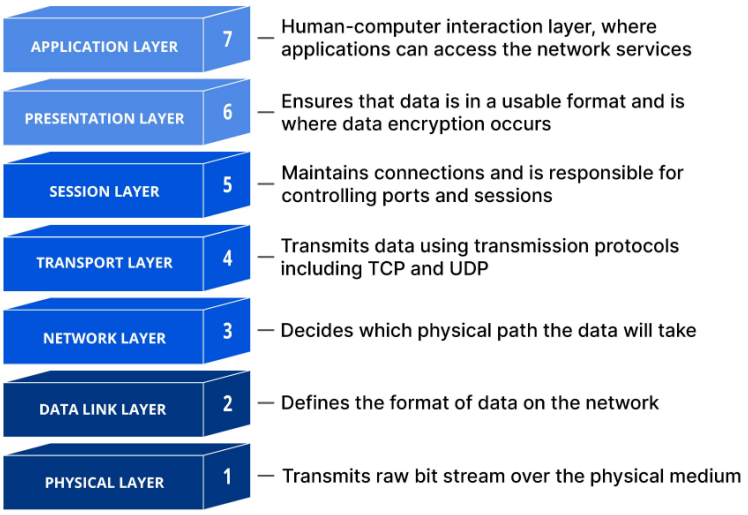

Рисунок 1.1 - Графічна схема мережевої моделі OSI

Кожен з рівнім моделі декларує «вертикальну комунікацію», що означає, що для кожного комп’ютера який підтримує модель OSI процес передачі пакету з фізичного середовища до програми адресата буде складатися з семи етапів і відповідно в зворотному напрямі.

Передача пакету незалежно від рівня на якому він знаходить відбувається або вверх наступному/попередньому рівню, або донизу наступному/попередньому, в залежності від напряму руху пакету додаючи до пакету заголовок того протоколу який було використано, що проілюстровано на рисунку два.

На кожному рівні моделі визначенні протоколи, які оперують виключно на ньому і забезпечували «горизонтальну синхронізацію», що означає, що до пакету проходячи через, наприклад, четвертий рівень (транспортний) додається заголовок (наведено на рисунку 1.2) четвертого рівня, який містить номер порту відправника та номер порту отримувача і отримавши такий пакет комп’ютер отримувач за допомогою інформації з

заголовка зрозуміє на якому порті «слухає» програма, якій призначено даний пакет і перенаправить його туди.

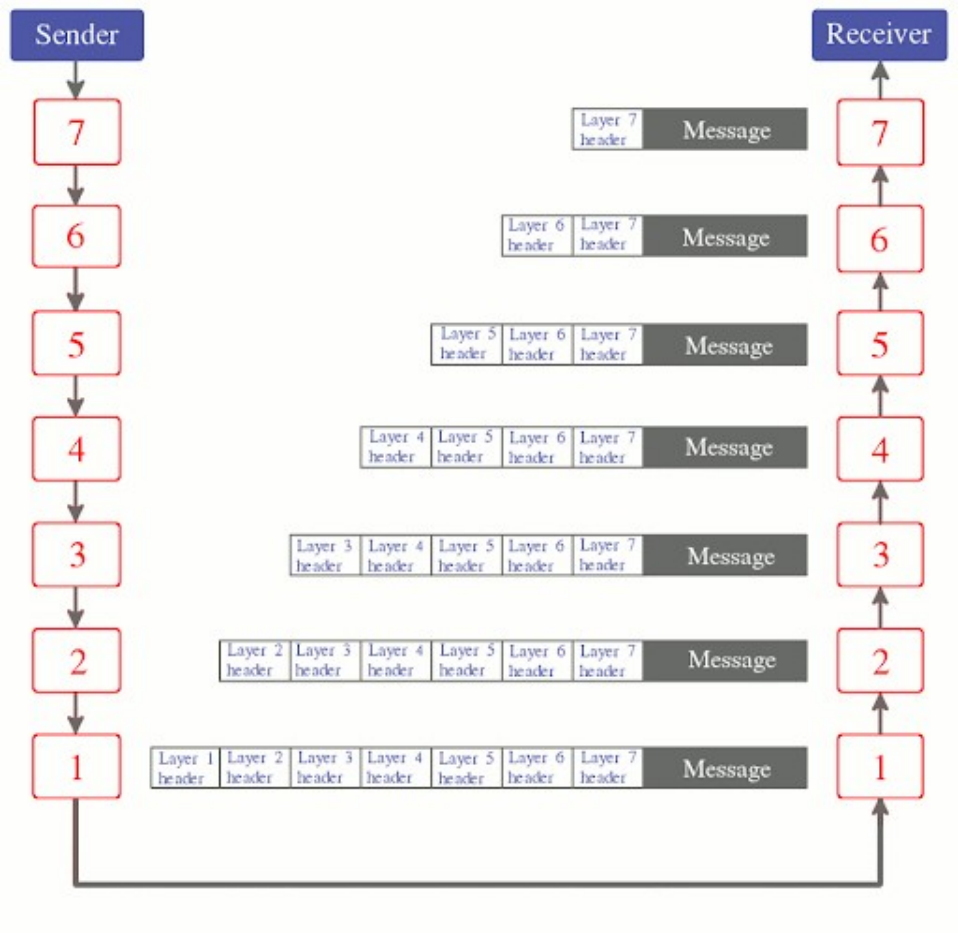

Рисунок 1.2 - Порядок додавання заголовків протоколів до пакету

На рисунку три продемонстровано приблизний список (наведено на рисунку 1.3) протоколів, що використовувалися в моделі OSI та їх відповідність певним рівням моделі. Дані протоколи, на своєму рівні, визначають задачі для яких призначений пакет. Таким чином використання HTTP протоколу додатком користувача означає, що швидше за все користувач виконує запит веб-сторінки з браузера до веб-сервера, використання TCP протоколу на транспортному рівні свідчить про намір встановити сесію між хостами для передачі даних, в той час як UDP протокол не встановлює сесію, а лише відправляє повідомлення від порта n до порта m.

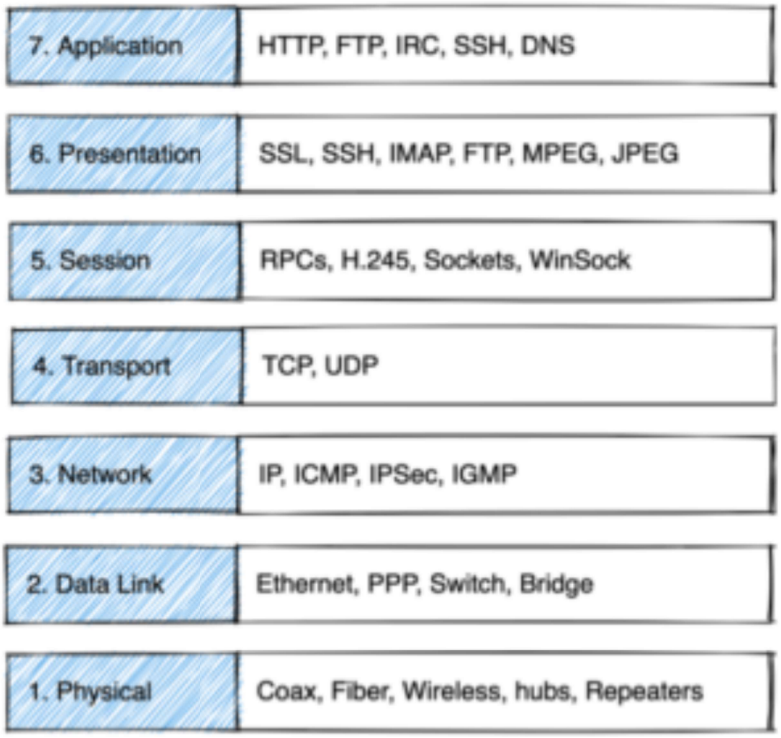

Рисунок 1.3 - Відповідність протоколів рівню моделі OSI

З розвитком інформаційних технологій та мереж модель OSI було витіснено більш простою та в той ж час зручною моделлю TCP/IP.

Дана модель склалася на базі протоколів мережевого та транспортного рівнів: TCP та IP, тому і отримала таку назву. На відміну від моделі OSI дана модель мала чотири рівні (або 5, існують визначення моделі TCP/IP, як п’яти рівневої моделі).

Модель TCP/IP була більш простою від своєї попередниці та при тому не втрачала функціональності. Найбільших змін зазнали рівні 5 — 7, які було злито в один, також злиття відбулося для рівнів 1 і 2 моделі OSI (у випадках з п’яти рівневою моделлю саме ці рівні залишаються окремими).

На рисунку 1.4 видно наскільки більш компактною стала нова модель в порівнянні зі своєю попередницею.

Чотирьохрівнева модель TCP/IP на момент виконання роботи залишається актуальною і найбільш уживаною в глобальних мережах.

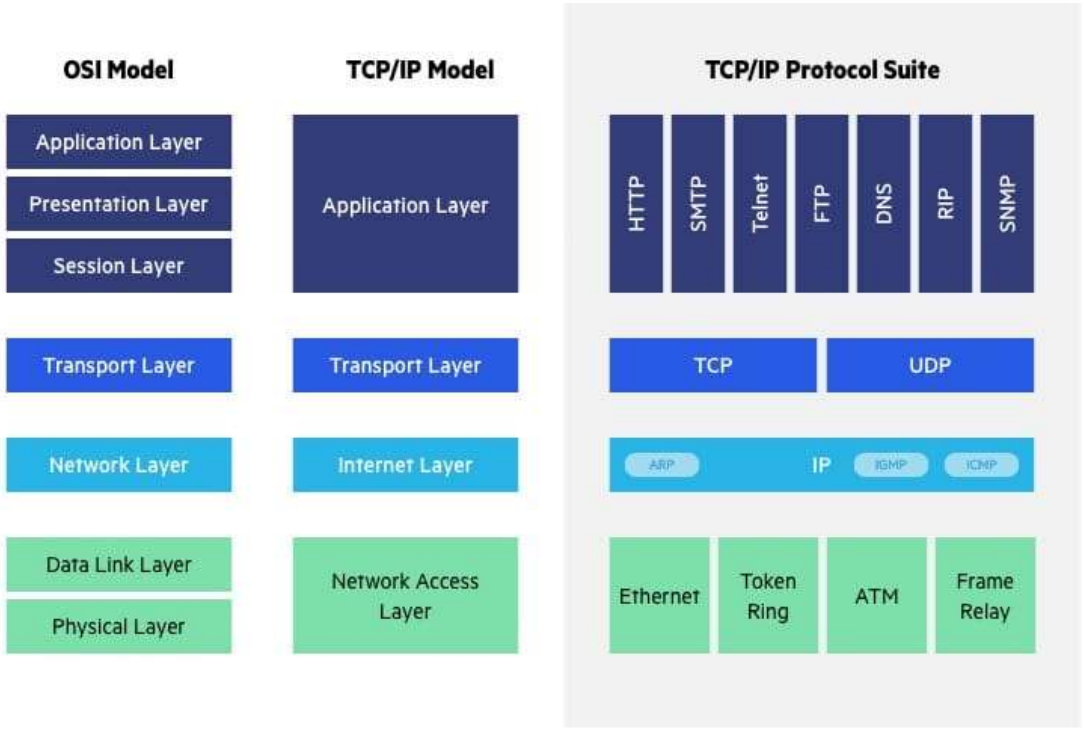

Рисунок 1.4 - Порівняння моделей OSI та TCP/IP

Хоч і спостерігається тенденція до набуття все більшої популярності SD-WAN (Software defined wide area network), ця технологія залишається перспективною і знаходить своє місце в інфраструктурі хмарних провайдерів та закритих мережах компаній, проте не в змозі змінити класичні глобальні мережі.

1.2 Сучасні рішення в сфері кібербезпеки

Розглянувши середовище де застосовуються оперують системи, що покликані забезпечувати безпеку розглянемо основні типи подібних систем, які задачі вони вирішують та як імплементуються.

1.2.1 Host-based Firewall

Подібні системи є програмним продуктом, який функціонує на хості кінцевого користувача. Системи подібного типу слугують останньою ланкою

«оборони» та виконують примітивні функції, на кшталт, закриття/відкриття портів, заборони отримувати пакети з певної ір адреси тощо. Даний тип програмного забезпечення є найбільш примітивним і виконує просто фільтрацію пакетів що дійшли до мережевого адаптера комп'ютера відповідно до набору визначених правил.

1.2.2 Web Application Firewall

Даний тип систем також можна віднести програмних продуктів. Подібні системи є мережевим екраном рівня додатків та призначені для виявлення та блокування атак на веб-додатки. Сучасні веб-додатки є повноцінними програмними продуктами зі своєю інфраструктурою (наведено на рисунку 1.5) та мають певні вразливості через особливості їх реалізації. Серед типів атак існує окремий тип, що відноситься саме до веб-додатків (у таблиці 1.1)

Таблиця 1.1 - Найбільш поширені атаки на веб-додатки

SQL Injection	SQL ін'єкції
Cross Site Scripting (XSS)	Міжсайтовий скриптинг
Bruteforce	Підбір паролів
Auth Bypass	Обхід аутентифікації

Типовим місцем розташування Web Application Firewall є позиція перед веб-сервером, яка дозволяє йому бачити увесь трафік, що прямує до веб-сервера та вчасно визначати та блокувати пакети, що можуть нести загрозу для сервера. Зазвичай для таких систем виділяють окремий сервер, щоб нічого не впливало на ефективність системи. Системи такого типу спираються на набір правил, що визначено адміністратором мережі та виконують сканування вмісту пакету на предмет сигнатур, що можуть експлуатувати вже відомі вразливості веб-додатків. Системи цього клас

оперують на останньому рівні мережевої моделі, тобто на їх цільовими пакетами є пакети рівня Application Layer.

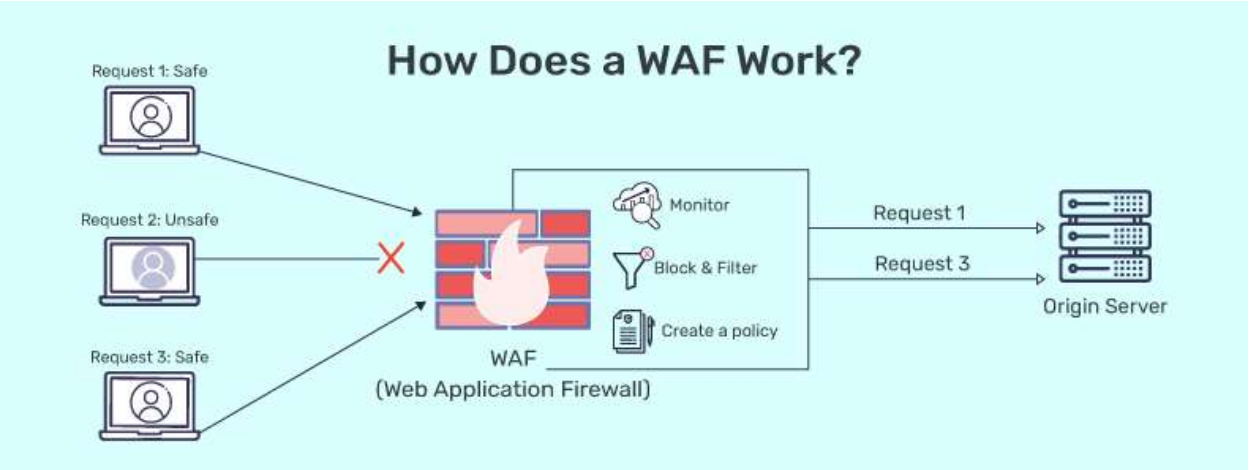

Рисунок 5.5 - Типовий приклад топології з використанням Web Application Firewall

1.2.3 Firewall

Даний тип відноситься до програмно-апаратних комплексів, адже готовий продукт містить не лише програмний код, а є готовим пристроєм, що встановлюється в мережі (наведено на рисунку 1.6) та конфігурується відповідно до потреб. Такий клас мережевого обладнання також називається міжмережевими екранами і оперують вона на мережевому і транспортному рівні мережі другий, третій рівні моделі TCP/IP). На пристрої такого класу покладено функції фільтрації, блокування трафіку, що проходить по мережі. Системи такого типу дуже поширені в великих корпоративних мережах, адже мають доволі високу ефективність і гнучкість в налаштуванні.

1.2.4 Intrusion Detection System та Intrusion Prevention System

Системи цього класу є розширенням можливостей звичайного міжмережевого екрану з точки зору інтелектуалізації його для детекції

шкідливого трафіку, загроз тощо. Програмно-апаратні комплекси, що включать в себе Firewall, IDS та IPS називаються Next-Generation Firewall (NGFW). Подібні системи окрім простої фільтрації трафіку за набором правил виконують також виявлення сигнатур атак, перевірку репутації клієнта або сервера та загалом мають розширений набір правил за якими виконується фільтрація.

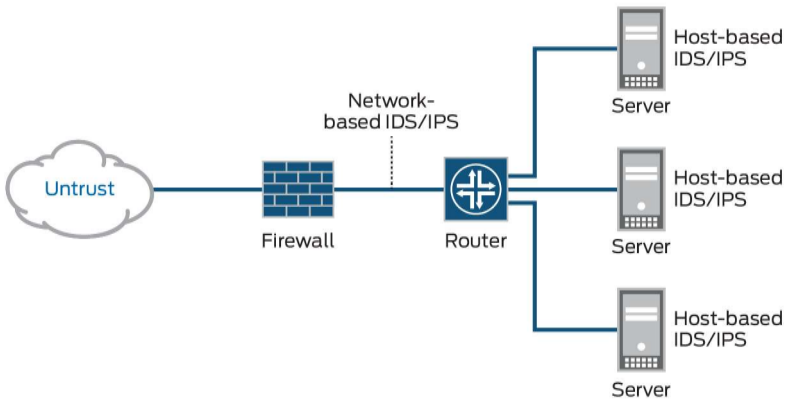

Рисунок 1.6 - Типова мережева топологія з IDS/IPS

Зазвичай подібні системи можуть працювати як в режимі IPS так і IDS, різницею є те, що IDS виконує лише детекцію і може лише повідомити адміністратора про потенційну загрозу, в той час як IPS може одразу заблокувати підозрілу сесію, та попередити потенційні втрати від можливої атаки.

Підсумовуючи можна сказати, що спеціалісти з кібербезпеки мають доволі широкий набір засобів для забезпечення перевірки трафіку на легітимність, проте усі ці засоби мають одну спільну рису — алгоритми і логіка якою вони керуються. На разі, усі подібні системи використовують класичні ітераційні алгоритми пошуку сигнатур у випадку з IDS/IPS або просто виконують перевірку на відповідність певних полів заголовків пакетів на відповідність правилам.

Метою ж цієї роботи є дослідити та запропонувати альтернативні методи виявлення мережевих загроз, атак тощо використовуючи методи

сучасних інтелектуальних систем. Такі системи, які є широко поширеними в наш час та є доволі розповсюдженими використовуються широким спектром компаній, підприємств та корпорацій, адже жодна сучасна організація не хоче стати жертвою зловмисників, втратити конфіденційну інформацію, допустити розкриття конфіденційної інформації тощо.

Подібні системи та програмно апаратні комплекси мають одну спільну рису — вони базуються на класичних алгоритмах пошуку відповідностей даних сигнатурам або правилам, що зумовлено віком технології, проте подібна імплементація несе як плюси, так і мінуси.

До мінусів можна віднести прогресію складності алгоритмів з часом, що означає зростання рівня складності алгоритмів детекції/комплексності логіки з плином часу, що ускладнює роботу над продуктом, який давно розвивається. Також серед мінусів можна відзначити відсутність можливості виявлення аномалій або нових загроз, оскільки вживані алгоритми будуються на чіткій логіці та шукають певні відповідності до шаблону в отриманих даних, то вони просто не в змозі виявляти, нові загрози, які ще не були виявлені дослідниками та задокументовані.

До плюсів можна віднести те що такі системи є більш прогнозовані в своїй поведінці та логічні, адже логіка, що використовується для кожного правила чи сигнатури можна чітко відстежити.

Дана робота ставить на меті дослідження методів виявлення загроз та аномалій в мережевому трафіку використовуючи нейромережевий підхід та теорію небезпеки на базі штучних імунних систем.

Підходи засновані на біології по природі своїй є що не найкраще потенційно придатними до використання в системах виявлення аномалій та загроз, а також навчання та адаптації. Адже саме цю функцію несуть несуть їх біологічні прототипи в реальному світі. Імунна система організму виробляє антитіла для виявлення та нейтралізації антигенів, що несуть загрозу для нього, нейрони та їх зв’язки є ледь не найменш вивченою системою в біологічних організмах — відповідають за навчання та адаптацію

до нових, не бачених до цього умов. А також побудови логічних ланцюгів та висновків, наприклад, що є добре, а що погано

За своєю природою алгоритми, що реалізують подібні підходи є оптимальними для використання в сфері кібербезпеки, адже мають на меті виконання тих самих задач, що й системи застосовувані в сфері кібербезпеки, адже в ідеальному випадку спеціалізовані системи мають не лише визначати набір відомих даних за вже відомими ознаками та заздалегідь запрограмованим алгоритмом, а вміти адаптуватись до змін та виявляти аномалії.

Не зважаючи на вищеозначені переваги біологічних алгоритмів вони на даний момент мають певні недоліки, які частково обумовлюють їх непоширеність в системах кібербезпеки. До таких недоліків відносяться складність відстеження логіки, найбільш це відноситься до нейромереж, які є розгалуженою системою ваг і зв’язків, що отримує вхідні дані та надає відповідь в результаті проходження даних по свій системі. Основну складність складає те, що типова «enterprise» нейромережа може мати більше мільйона зв’язків, які будуть враховані при розрахунку результату.

Також варто відзначити, те що системи побудовані, що «навчаються» схильні до перенавчання або упередженості. Оскільки біологічні системи базуються на навчанні, то мають місце доволі високі вимоги до якості підготовки даних що будуть використовуватися для навчання моделей. Як приклади кількох можливих негативних наслідків використання не якісних або не достатньо підготовлених даних можуть слугувати: перенавчання та упередженість. Перенавчання полягає в тому, що модель занадто добре адаптується до розпізнавання саме тих даних на яких вона навчалась і як наслідок може мати не задовільну ефективність на незнайомому наборі даних. При перенавчанні навіть невелике відхилення від вихідних даних може призвести до помилки системи. В випадку, з «упередженістю» моделі в результаті навчання на не якісних даних виявляються схильними до надання переваг певних результатів над іншими.

2 АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ ВИЗНАЧЕНОЇ ПРОБЛЕМИ

2.1 Основні відомості про нейронні мережі

Нейронні мережі — це є обчислювальним моделями, натхненні біологічними нейронними системами, яскравим прикладом яких є мозок. Нейронні мережі складаються з перцептронів (штучних нейронів) та зв’язків між ними. Кожен нейронний зв’язок має свою вагу, яка репрезентує важливість значення нейрона попереднього шару при обчисленні значення поточного нейрона.

Класична повнозв’язна нейромережа прямого поширення (наведено на рисунку 2.1) є моделлю де кожен нейрон пов’язаний з кожним, а отже значення кожного нейрону впливає на кожен інший.

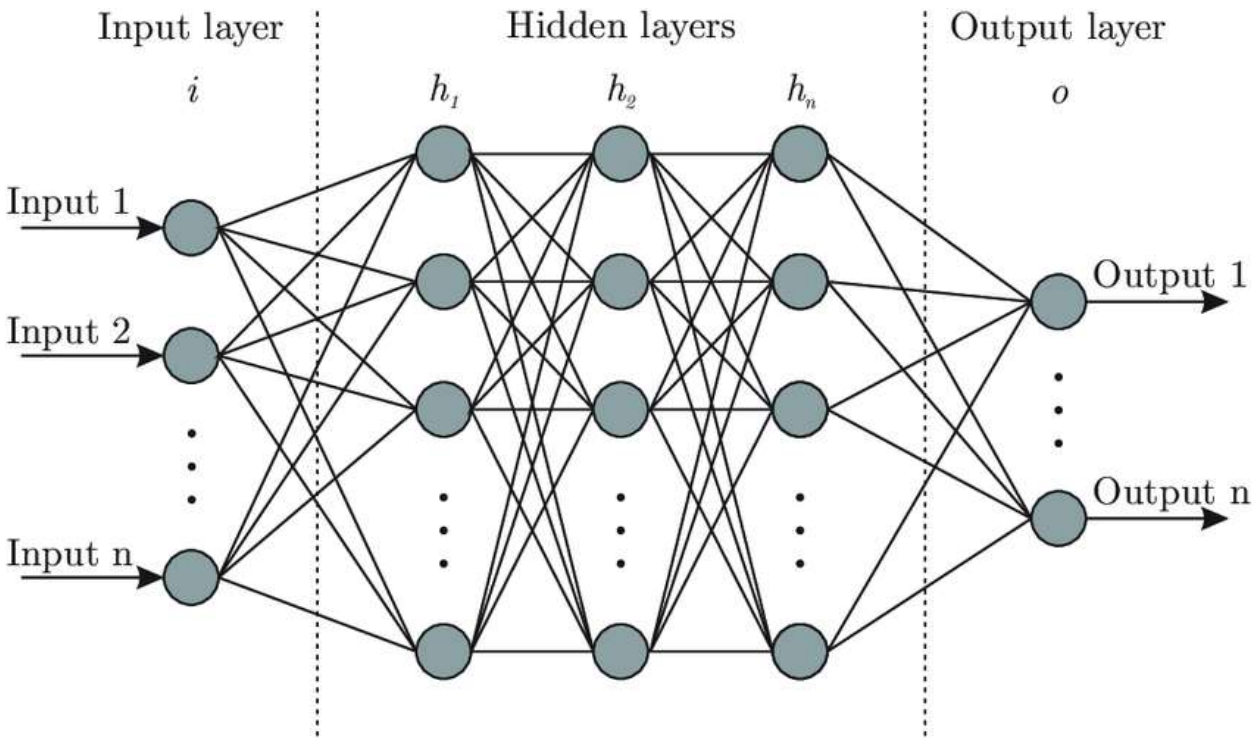

Рисунок 2.1 - Приклад повнозв'язної нейромережі прямого поширення

Для кожного нейрону в процесі поширення сигналу мережею обраховується власне значення[2], враховуючи значення всіх нейронів з

якими він пов’язаний та ваг цих зв’язків. Розрахунок значення нейрону здійснюється за наступною формулою (2.1):

$$y = f\left(\Sigma_{i=1}^{n} w_i x_i + b\right) \tag{2.1}$$

де y - вихідне значення нейрона;

f - функція активації;

w_i - вага, яка відповідає і-му вхідному сигналу;

x_i - і-й вхідний сигнал;

b - зсув (bias);

n - кількість вхідних сигналів.

2.1.1 Структура перцептрона

Розберемо структуру нейрона, який є базовим примітивом для побудови нейронних мереж. Найпростіші нейрони, що використовуються досі в нейромережах сьогодні засновані на запропонованій в 1943 році моделі нейрона Маккалоха-Пітса[3], є однією з перших моделей нейрона.. Ця модель є спрощеним математичним описом біологічного нейрона і служить основою для багатьох подальших досліджень у сфері нейронних мереж та штучного інтелекту. Розберемо структуру сучасного нейрона (наведено на рисунку 2.2), які використовуються в мережах прямого поширення.

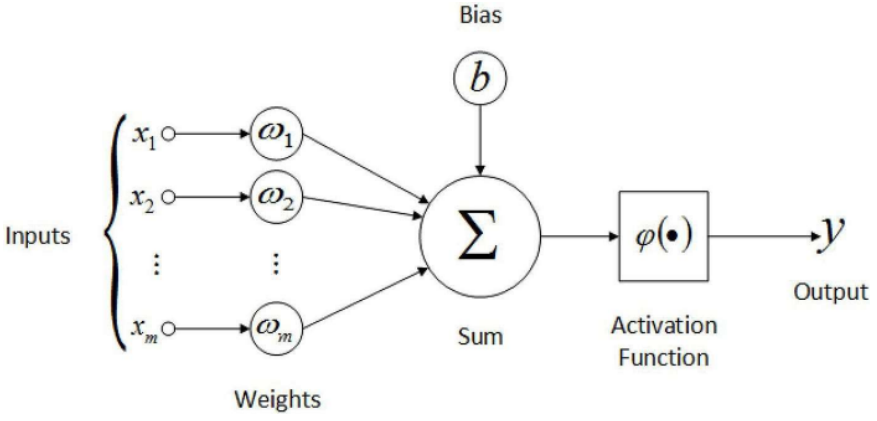

Рисунок 2.2 — Модель штучного нейрона

Нейрон складається з таких складових, як вхідні параметри та їхні ваги (кількість може варіюватись в залежності від задачі, даних тощо.), суматор — виконує функцію підрахунку суми додатків ваг та вхідних даних, які потім передаються до функції активації, результат якої є значенням нейрона.

2.1.2 Функції активації

Різні функції активації працюють по різному[3]. Найпростішою є порогова функція активації.

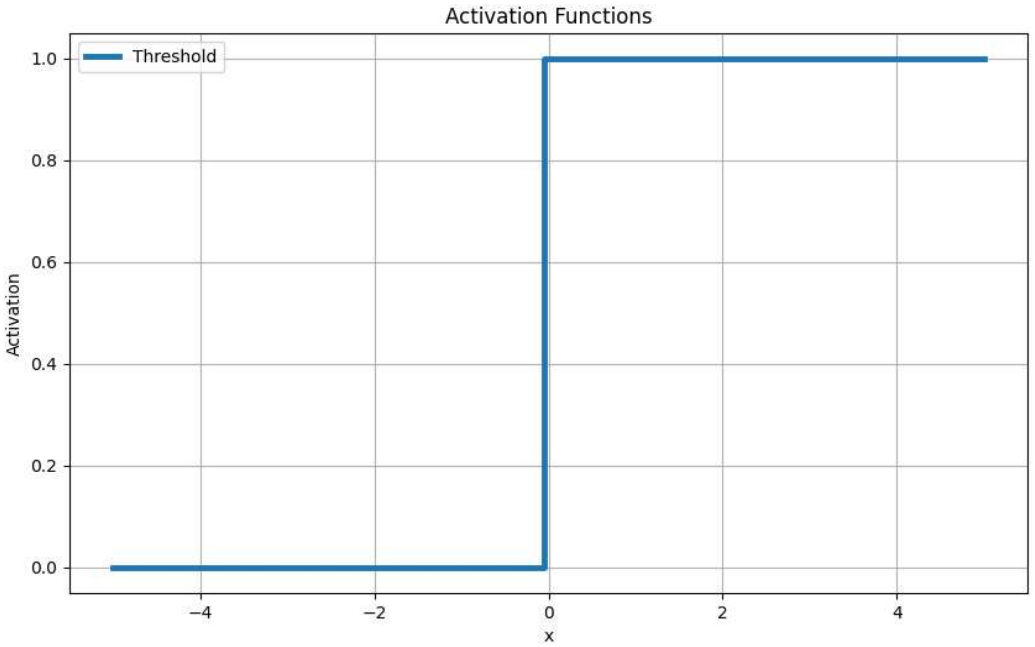

Рисунок 2.3 - Графік порогової функції активації

Формулу функції наведено на рисунку 2.3, результатом її роботи є бінарні значення = {0, 1}.

Дана функція працює за логікою «проходження порогу» за формулою (2.2)

$$\begin{cases} a>n, x=1 \\ a<n, x=0 \end{cases} \quad (2.2)$$

де a - результат роботи суматора;

n - порогове значення.

тобто якщо значення, яке суматор передав до даної функції активації більше встановленого порогу, то нейрон вважається «активованим» і функція активації повертає «1», в протилежному випадку «0» і нейрон вважається «не активованим». Дану функцію ще називають бінарною функцією активації.

Порогова функція активації використовується в найпростіших типах нейронних мереж, зокрема в перцептронах. Незважаючи на простоту та історичне значення, порогова функція активації має суттєві обмеження, через які вона поступилася місцем більш складним і ефективним функціям активації, таким як сигмоїдна функція, функція гіперболічного тангенсу (tanh) і ReLU (Rectified Linear Unit). Серед таких є: нездатність обробляти складні задачі. Через бінарний вихід вона не може добре працювати з задачами регресії або більш складними класифікаційними задачами, де потрібні неперервні вихідні значення.

Відсутність градієнта не дозволяє ефективно використовувати методи оптимізації на основі градієнтів, які є основою навчання сучасних нейронних мереж.

Порогова функція активації є важливим концептуальним кроком у розвитку нейронних мереж, але для сучасних додатків в машинному навчанні використовуються більш просунуті функції активації, які забезпечують кращу навчальну здатність та ефективність моделі.

Функція активації сигмоїд (наведено на рисунку 2.4) , або логістична функція[5], є однією з найпопулярніших функцій активації, що використовуються в нейронних мережах, особливо в ранніх моделях машинного навчання. Її формула виглядає наступним чином (2.3):

$$\sigma(x)=\frac{1}{1+e^{-x}} \tag{2.3}$$

де $\sigma(x)$ - вихідне значення функції;

x - вхідне значення;

e - основа натурального логарифма (приблизно 2.71828).

До переваг цієї функції відносять «нормалізацію» - сигмоїд стискає будь-яке вхідне значення до діапазону (0, 1), що може бути корисним для нормалізації вихідних значень та «інтерпретацію» вихідного сигналу як ймовірність, що є доволі корисним в задачах класифікації.

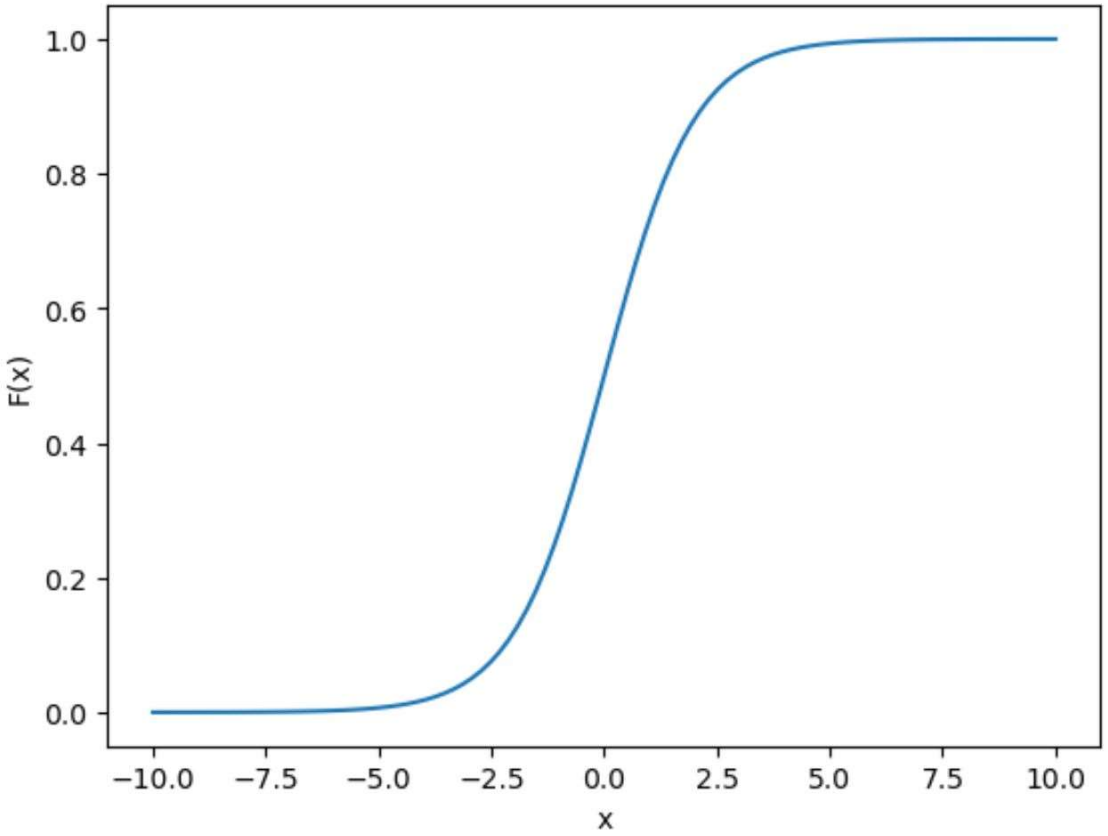

Рисунок 2.4 - Графік функції активації "сигмоїда"

До недоліків цієї функції відносять таку властивість як «витік градієнта» це означає, що в областях, де значення «x» є дуже великими або дуже малими, градієнт сигмоїду стає дуже малим, що призводить до проблеми "зникання градієнта"[4]. Це ускладнює навчання глибоких нейронних мереж. Ще одним недоліком є не центрований вивід. Вихід сигмоїду не нульовий (він знаходиться в межах від 0 до 1), що може ускладнювати навчання моделей з великою кількістю шарів.

Сигмоїд часто використовується в останньому шарі моделей для задач бінарної класифікації, де вихід інтерпретується як ймовірність належності до одного з двох класів. В середніх шарах сучасних глибоких нейронних мереж сигмоїд використовується рідше через проблему зникання градієнта; частіше використовуються такі функції активації, як ReLU та її варіанти.

Функція активації ReLU є однією з найбільш використовуваних функцій активації в сучасних нейронних мережах, особливо в глибокому навчанні. Вона проста, як наведено на рисунку 2.5, ефективна і допомагає вирішувати проблему зникнення градієнта, що робить її популярною у багатьох архітектурах нейронних мереж.

Принцип роботи ReLU полягає в тому, що за умови, якщо вхід «x» більше нуля, вихід буде рівним самому вхідному значенню «x». Іншими словами, ReLU пропускає всі позитивні значення без змін за формулою (2.4).

$$f(x)=max(0,x) \tag{2.4}$$

де x – вхідне значення для функції активації.

Якщо вхід «x» менше або дорівнює нулю, вихід буде нульовим. Це означає, що всі негативні значення обнуляються.

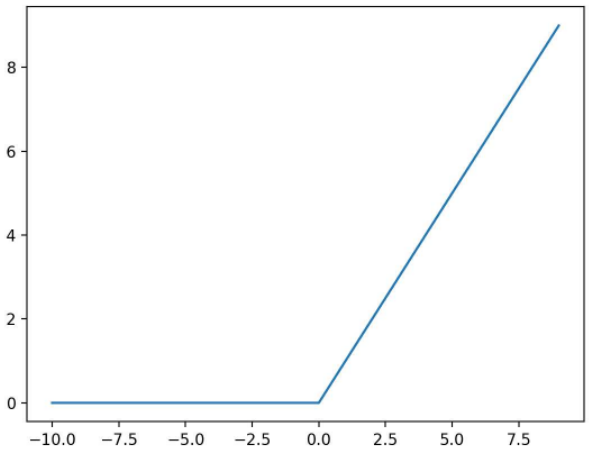

Рисунок 2.5 - Графік активаційної функції ReLu

До переваг ReLu відносять «нелінійність», хоча ReLU є простою функцією, вона додає нелінійність до моделі, що дозволяє нейронним мережам навчатися складним функціям і моделям, «швидке обчислення» - обчислення ReLU є дуже простим і швидким, оскільки це просто порівняння з нулем і вибір максимального значення. Також ReLu сприяє збереженню градієнта. На відміну від сигмоїдних чи гіперболічних тангенсових функцій активації, ReLU не насичується при великих значеннях, що допомагає уникнути проблеми зникнення градієнта і сприяє швидшому і ефективнішому навчанню нейронних мереж.

Серед недоліків цієї функції виділяють - "вмирання" нейронів (Dying ReLU). Під час навчання деякі нейрони можуть стати ніколи не активними, тобто вихід завжди буде нульовим для всіх вхідних даних. Також ReLU обнуляє всі негативні значення, що може призвести до втрати інформації,

Розглянемо варіації ReLu[5]. Для подолання деяких недоліків ReLU були розроблені його варіації: Leaky ReLU — дозволяє невелике негативне значення, визначене параметром «α», зазвичай дуже малим, наприклад, 0.01 відповідно до формули (2.5).

$$f(x)=max(\alpha * x, x) \tag{2.5}$$

де a – коефіцієнт функції;

x – вхідне значення для функції активації.

Parametric ReLU (PReLU): Подібна до Leaky ReLU, але параметр «α» навчальний, а Exponential Linear Unit (ELU) дає експоненційне значення для негативних вхідних значень згідно формули (2.6).

$$\begin{cases} x, x \geq 0 \\ \alpha(e^x - 1), x < 0 \end{cases} \tag{2.6}$$

де a – коефіцієнт функції;

e - основа натурального логарифма (приблизно 2.71828).

x - вхідне значення для функції активації.

ReLU є важливим інструментом у глибокому навчанні, завдяки своїй простоті та ефективності, і залишається предметом подальших досліджень і вдосконалень у різних контекстах нейронних мереж.

2.1.3 Багатошаровий перцептрон

Багатошаровий перцептрон (multilayer perceptron, MLP) — це тип штучної нейронної мережі, що складається з трьох основних типів шарів: вхідного шару, прихованих шарів і вихідного шару (наведено на рисунку 2.6). Кожен з цих шарів має свої цілі та особливості.

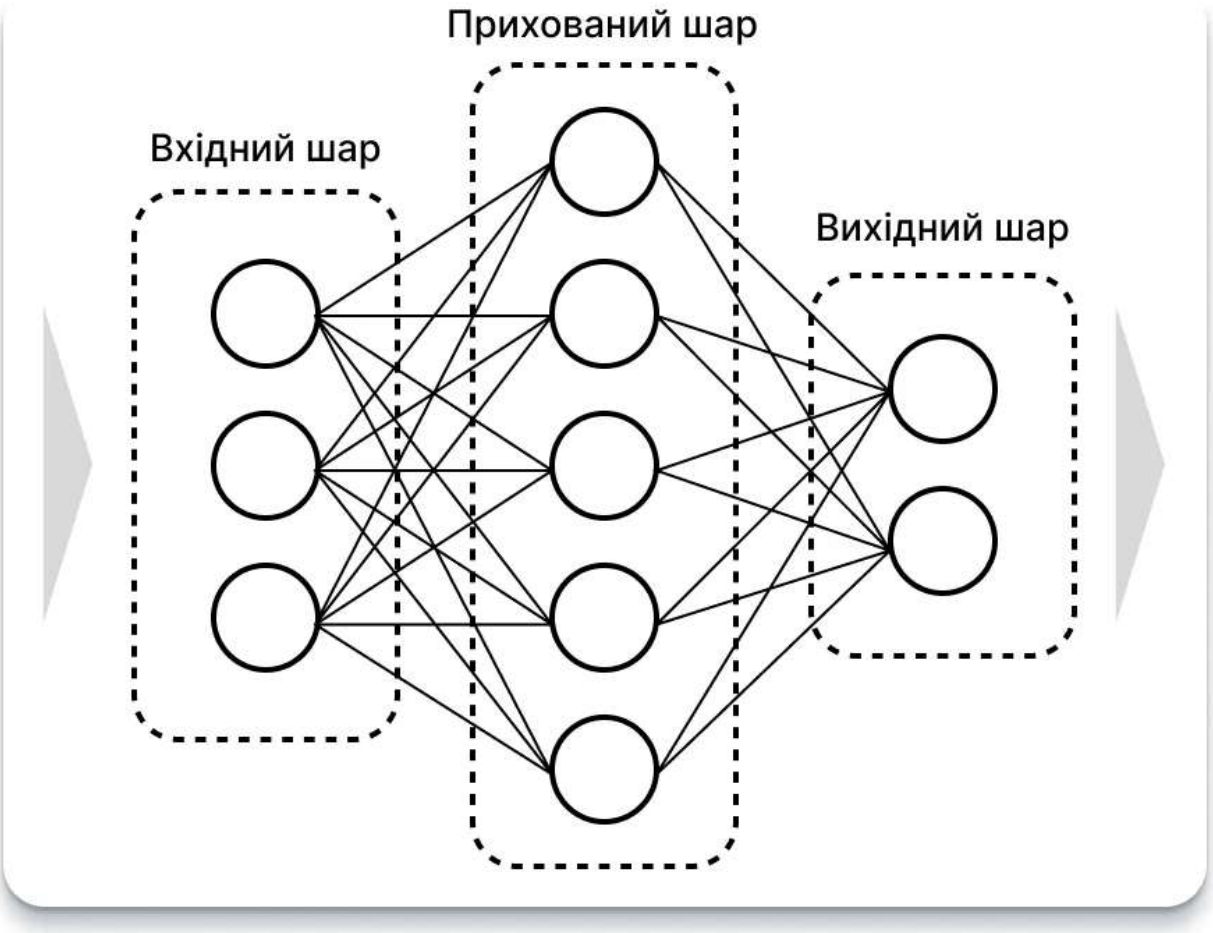

Рисунок 2.6 - Загальна структура багатошарового перцептрона

Вхідний шар - це перший шар мережі, який отримує вхідні дані. Кількість нейронів у вхідному шарі зазвичай відповідає кількості ознак або вхідних параметрів моделі, а кожен нейрон в цьому шарі представляє одну вхідну ознаку і приймає значення цієї ознаки як свій вхід.

Приховані шари знаходяться між вхідним і вихідним шарами. Кожен прихований шар складається з декількох нейронів, які приймають вхідні сигнали(зв'язки) з попереднього шару, таким шаром може бути або вхідний шар або інший прихований шар. Кількість прихованих шарів і кількість нейронів у кожному з них може варіюватися в залежності від реалізації моделі. Вихідний шар — останній шар мережі, який генерує вихідні прогнози або результат. Кількість нейронів у вихідному шарі зазвичай відповідає кількості можливих класів або кількості вихідних змінних, які очікується отримати.

Функція активації є складовою кожного нейрона у мережі, крім можливо вихідного шару, визначає для кожного нейрона його вихід на основі зваженої суми вхідних сигналів. Тренування мережі виконується з використанням алгоритму зворотного поширення помилки (backpropagation), який використовується для оптимізації ваг нейронів, а ваги нейронів оновлюються з метою зменшення помилки між прогнозованими і правильними вихідними значеннями.

MLP є потужним і загальним інструментом для вирішення багатьох завдань машинного навчання, включаючи класифікацію та регресію, за умови, що він правильно налаштований і відповідно навчений на відповідних даних.

2.1.4 Навчання нейронних мереж

Навчання нейромереж — це процес налаштування параметрів нейромережі за допомогою навчальних даних з метою здійснення певного виду обчислень або передбачень. Навчання нейромережі включає кілька

основних етапів, наприклад, підготовка даних. Підготовка даних є комплексним завданням, можливо навіть складнішим ніж побудова нейромережевої моделі. Включає в себе збір, очищення і підготовку даних для навчання. Дані мають бути у форматі, зрозумілому для моделі. Даний етап є доволі комплексним та потребує від дослідника обізнаності в математиці, предметній області, теорії нейромереж для того аби належним чином інтерпретувати дані, видалити не інформативні, масштабувати тощо

Визначення архітектури моделі є однією з визначних задач, адже вибір типу нейромережі (наприклад, згорткова, рекурентна, або комбінована модель) та конфігурація її структури, включаючи кількість шарів, кількість нейронів на кожному шарі напряму впливають на результати дослідження, адже навіть добре підготовані дані в комбінації з непідхожою моделі не дадуть хороших результатів. Не менш важливим етапом є навчання моделі під час якого модель навчається на вхідних даних. Це зазвичай включає подачу даних на вхід до моделі, обчислення вихідного значення, порівняння цього значення з очікуваним результатом і корекцію параметрів моделі згідно зі здійсненими помилками. Після навчання моделі виконується її оцінка, зазвичай на нових даних, яких модель не бачила під час навчання для оцінки її точності і ефективності в реальних умовах[3].

Налаштування гіперпараметрів є процесом оптимізація параметрів, які не входять в процес навчання, але впливають на якість моделі, таких як швидкість навчання (learning rate), розмір батча (batch size) тощо.

Цей процес є складним і вимагає глибоких знань у галузі машинного навчання та нейронних мереж, а також експериментування з різними конфігураціями моделі для досягнення найкращих результатів.

Одним з поширених методів навчання моделей градієнтний спуск та його різновиди. Стохастичний градієнтний спуск та алгоритми, що на ньому засновані, відіграють вирішальну роль на етапі навчання моделей глибокого навчання. Глибоке навчання, підгалузь штучного інтелекту, зосереджується на навчанні нейронних мереж із кількома рівнями для вивчення складних

шаблонів і створення точних прогнозів або класифікацій. Процес навчання передбачає ітераційне коригування параметрів моделі. Алгоритми оптимізації, такі як SGD, допомагають досягти цієї мети, ефективно оновлюючи параметри моделі на основі виявлених помилок.

Стохастичний градієнтний спуск це варіант градієнтного спуску, який є загальною технікою оптимізації для знаходження мінімуму функції (наведено на рисунку 2.7). Основна ідея полягає в оновленні параметрів моделі після кожного прикладу навчання. Градієнт представляє напрямок найкрутішого підйому, і, інвертуючи значення градієнта, SGD визначає напрямок найкрутішого спуску. Потім виконується оновлення параметрів в цьому напрямку, за рахунок чого модуль ефективно рухається до кращого рішення.

У кожній ітерації SGD[2] швидкість навчання, яка визначає розмір кроку для оновлення параметрів, є ключовим гіперпараметром. Висока швидкість навчання може призвести до того, що процес оптимізації перевищить оптимальне рішення, тоді як низька швидкість навчання може призвести до повільної конвергенції. Пошук відповідної швидкості навчання часто є процесом проб і помилок, і для покращення конвергенції можна використовувати такі методи, як графіки темпів навчання або адаптивні темпи навчання.

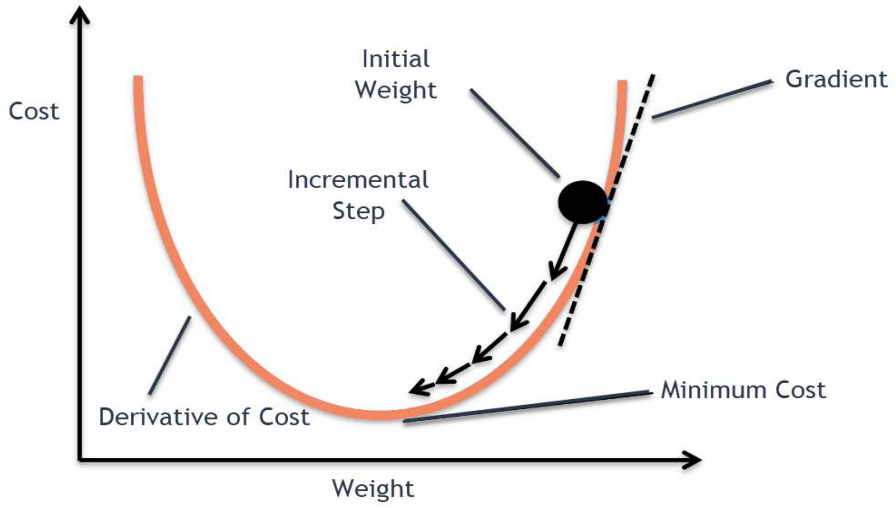

Рисунок 2.7 - Графічне зображення алгоритму SGD

2.2 Основні відомості про штучні імунні системи

Штучні імунні системи (ШІС) і теорія небезпеки є концепціями, які використовуються в кібербезпеці для виявлення і запобігання загрозам.

2.2.1 Штучні імунні системи

ШІС — це обчислювальні системи, які моделюють принципи імунної системи людини для виявлення та протидії аномаліям, таким як віруси, черв'яки(Worms) або інші шкідливі програми[9]. Ідея полягає в тому, що так само, як біологічна імунна система виявляє та знищує патогени, ШІС здатні виявляти і нейтралізувати кіберзагрози.

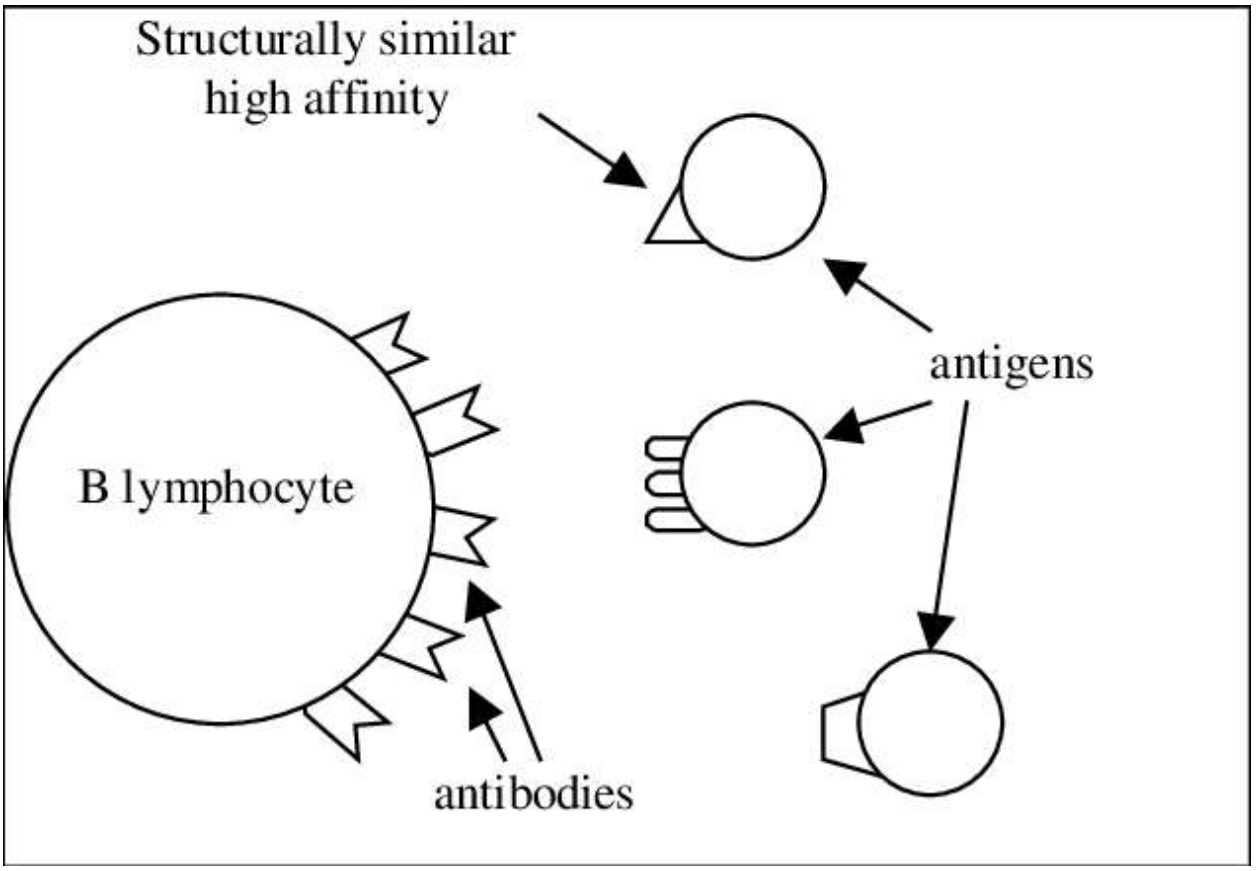

Рисунок 2.8 - Виявлення антитілами антигенів засноване на афінності

До основних компонент/складових штучних імунних систем відносяться: антитіла (детектори — представляють собою алгоритми, які

виявляють аномальні дії або зміни в системі, антигени — елементи, які представляють потенційні загрози, наприклад, фрагменти шкідливого коду. Для вимірювання ступені схожості між антигеном та антитілом введено поняття «афінність»(Affinity), воно є мірою вимірювання того, наскільки добре антитіло розпізнає антиген(наведено на рисунку 2.8). Висока афінність означає, що загроза виявлена з високою точністю. На рисунку 2.9 продемонстровано принцип покриття шкідливих даних(антигенів) антитілами[10].

Популяція антитіл є ще одним важливим структурним компонентом штучної імунної системи. Вона представляє собою набір різних антитіл, які працюють разом для виявлення широкого спектра загроз.

Принцип роботи та мета систем заснованих на положеннях штучних імунних систем полягає в виявленні аномалій, а саме відхилень від нормальної поведінки, розрізнення «свого» та «чужого». Даний процес полягає в виявленні компонентів системи, які є нормальними (своїми), і тих, які є чужорідними або аномальними (чужими).

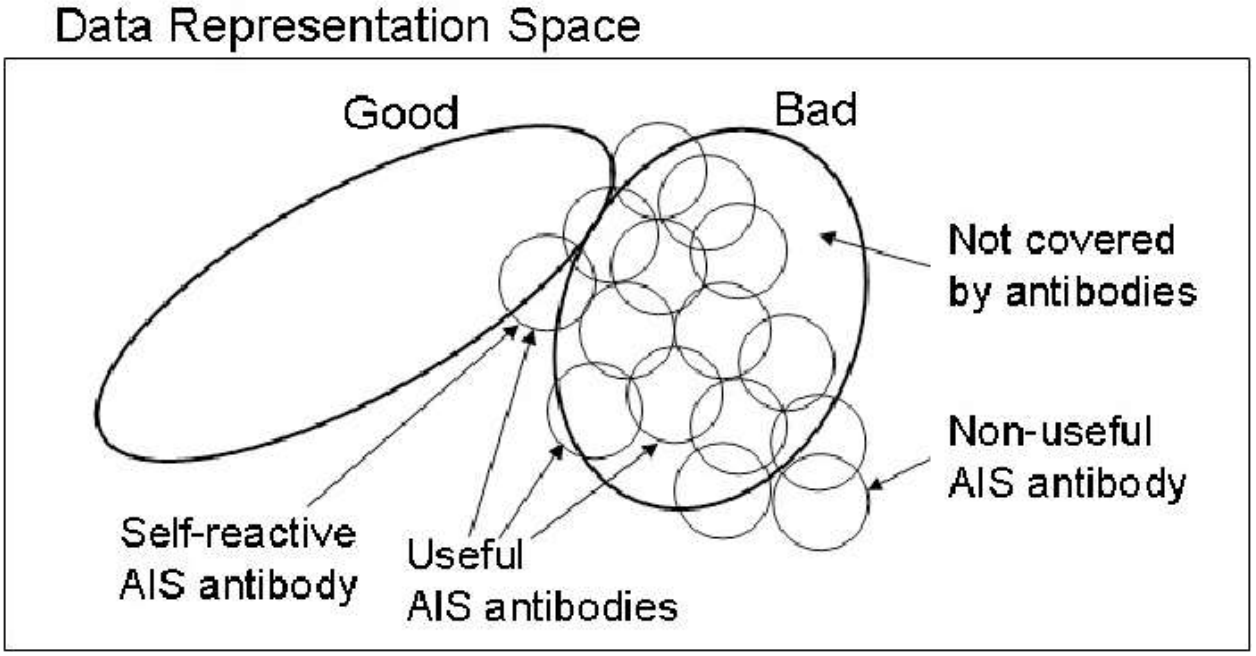

Рисунок 2.9 - Покриття шкідливих даних(антигенів) антитілами.

Головним аспектом, що виділяє штучні імунні систем це — механізм адаптації. Здатність систем вчитися з часом і адаптуватися до нових загроз відкриває широкі можливості для застосування в кібербезпеці, адже сфера перебуває в стані постійного розвитку, що дня з'являється нове програмне забезпечення і в ньому за лічені дні знаходяться вразливості, топології комп'ютерних мереж постійно змінюються, змінюються принципи побудови мереж, розвивається SD-WAN тощо. Всі ці флуктуації в сфері комп'ютерних технологій призводять до необхідності швидко адаптуватися до мінливого середовища та появи до того не знайомих даних, загроз, шаблонів поведінки. Механізм адаптації в штучних імунних системах базується на декількох ключових принципах та алгоритмах, які дозволяють системі вчитися з часом і ефективно адаптуватися до нових загроз. Ці механізми багато в чому натхненні біологічними процесами, але мають специфічні обчислювальні реалізації.

Основні принципи та алгоритми механізму адаптації потягають в використанні таких методів як еволюційні алгоритми, алгоритми навчання з підкріпленням, адаптивні нейронні мережі, системи з імунною пам'яттю, алгоритми навчання на основі прикладів.

Еволюційні алгоритми можливо поділити на кілька підгруп і виділити серед них наступні: генетичні алгоритми (GA), які використовуються для еволюції наборів антитіл. Через процеси селекції, схрещування та мутації генетичні алгоритми дозволяють створювати нові покоління антитіл, які краще підходять для виявлення нових загроз, а також алгоритми диференціальної еволюції, які покликані оптимізувати популяції антитіл шляхом зміни їх характеристик на основі різниць між випадковими парами індивідів.

Алгоритми навчання з підкріпленням діляться на Q-навчання за якого агент вивчає політику дій, яка максимізує суму винагород у часі, шляхом взаємодії з середовищем і отримання зворотного зв'язку у вигляді винагород або покарань та SARSA (State-Action-Reward-State-Action) яке є варіантом Q-

навчання, який враховує наступну дію, обрану політикою, під час оновлення оцінки Q-функції.

Серед адаптивні нейронні мережі можна відзначити конволюційні нейронні мережі (CNN), які використовуються для розпізнавання образів та класифікації аномалій в трафіку або поведінці програм та рекурентні нейронні мережі (RNN), які застосовуються для аналізу часових рядів даних, таких як мережевий трафік або лог-файли, з метою виявлення аномалій.

Системи з імунною пам'яттю мають на меті вивчення і запам'ятовування шаблонів. Система такого типу зберігає шаблони загроз, з якими стикалася раніше, і використовує їх для швидкого виявлення схожих загроз у майбутньому.

Афінність і кластеризація використовуються для виявлення подібності між новими загрозами і збереженими шаблонами, що дозволяє швидко адаптуватися до нових атак.

Механізми адаптації можуть бути реалізовані по різному, серед головних можна відзначити наступні — «динамічне оновлення бази даних загроз», за якого система постійно оновлює базу даних відомих загроз на основі нових вхідних даних. Це дозволяє швидко адаптуватися до нових вірусів, черв'яків або інших шкідливих програм, «поведінковий аналіз» - є моніторингом та аналізом поведінки системи і користувачів. Дана реалізація дозволяє виявляти відхилення від норми. На основі цих даних система може створювати нові правила і моделі для виявлення аномалій можливим вважається застосування селективного навчання, яке полягає в використанні відбору найефективніших антитіл для створення нових поколінь, що забезпечує поступове поліпшення здатності системи виявляти загрози.

Отже можна резюмувати, що механізм адаптації штучних імунних систем дозволяє їм ефективно вчитися на основі досвіду, аналізувати нові дані, оптимізувати свою роботу і залишатися актуальними в умовах постійно змінюваних кіберзагроз.

Це досягається за рахунок використання складних обчислювальних алгоритмів, які імітують принципи роботи біологічних систем.

2.2.2 Теорія небезпеки

Теорія небезпеки, запропонована Полою Маццингою, розширює традиційну концепцію свій/чужий розрізнення в імунній системі. Вона стверджує, що імунна система активується не лише присутністю чужорідних елементів, а й сигналами небезпеки, які вказують на пошкодження тканин або клітин[8].

Можна виділити основні положення теорії:

1. Сигнали небезпеки (Danger signals) відповідно до теорії призначені для виявлення пошкоджених або стресових клітин, які виділяють певні молекули, що сигналізують про небезпеку.

2. Контекстуальне розпізнавання є оцінкою контексту, в якому виявлено аномалію. Це дозволяє уникнути помилкових спрацьовувань.

3. Толерантність означає, що система повинна відрізняти шкідливі загрози від нешкідливих аномалій, щоб уникнути помилкових позитивних результатів.

ШІС і теорія небезпеки знаходять своє застосування в різних аспектах кібербезпеки. Подібні системи можуть бути застосовані для виявлення вторгнень, наприклад, бути застосованими для реалізації Intrusion Detention System. Системи виявлення вторгнень можуть використовувати ШІС для виявлення несанкціонованих дій або аномалій у мережевому трафіку. Головною перевагою є те що використання створювати системи, які можуть адаптуватися до нових загроз на основі сигналів небезпеки, що надходять з різних джерел. Розрізнення за принципом «свій»/«чужий» в комп’ютерних мережах можливо використовувати визначення нормального і аномального трафіку, що корисно у виявленні шкідливих дій в мережі.

3 РОЗРОБКА РІШЕННЯ

3.1 Формулювання проблем при створенні системи

Ключовим дослідним завданням є дослідження, аналіз методів для своєчасного виявлення загроз в комп’ютерних мережах. Розглянуто кілька методів та підходів до аналізу мережевого трафіку на предмет кіберзагроз, описано основні переваги та недоліки які мають розглянуті методи, а також наведено перелік проблем, які можуть виникати в процесі інтеграції зазначених методів в наявні системи або побудові нових беручи за основу модулі штучного інтелекту для виконання аналізу та прийняття рішень в комп’ютерних мережах також сформульовано ключові виклики, які потрібно вирішити для забезпечення їх ефективності з акцентом на нейромережевий підхід.

Розробка системи для виявлення загроз у комп’ютерних мережах вимагає вирішення численних проблем, які охоплюють як технічні аспекти, так і питання інтеграції та конфіденційності. Застосування нейромережевих підходів забезпечує значні переваги у масштабованості, адаптивності та точності аналізу. Успішна реалізація таких систем базується на використанні сучасних методів глибокого навчання та ефективної інтеграції з наявною інфраструктурою.

Розроблено декілька моделей штучного інтелекту використовуючи різні архітектурні рішення для побудови моделей, такі як: багатошаровий перцептрон та згорткові нейронні мережі. Нейромережевий підхід використання для навчання моделей ШІ використовуючи статистичні дані трафіку в комп’ютерній мережі для виявлення аномалій та викидів в великих обсягах неструктурованих даних.

3.2 Попередня обробка та аналіз даних використаних для навчання

Для навчання обрано датасет: CIC-IDS2017[6]. Набір даних CIC-IDS-2017 містить безпечні та найновіші поширені атаки, які нагадують справжні реальні дані (PCAP). Він також містить результати аналізу мережевого трафіку за допомогою CICFlowMeter із позначеними потоками на основі міток часу, IP-адрес джерела та призначення, портів джерела та призначення, протоколів і атак (файли CSV).

3.2.1 Опис датасету

Створення реалістичного фонового трафіку було нашим головним пріоритетом у створенні цього набору даних. Було використано запропоновану систему B-Profile для профілювання абстрактної поведінки людських взаємодій і створення натуралістичного «доброго» фонового трафіку. Для цього набору даних було створено абстрактну поведінку 25 користувачів на основі протоколів HTTP, HTTPS, FTP, SSH і електронної пошти.

Період збору даних розпочався о 9 годині ранку в понеділок, 3 липня 2017 року, і закінчився о 17 годині. у п’ятницю, 7 липня 2017 р., загалом 5 днів. Понеділок є звичайним днем і включає лише щадний трафік. Реалізовані атаки включають грубу силу FTP, грубу силу SSH, DoS, Heartbleed, веб-атаку, проникнення, ботнет і DDoS. Вони відбувалися вранці та вдень у вівторок, середу, четвер і п'ятницю.

Канадський Інститут з Кібербезпеки[6] визначив систему оцінки набору даних (Gharib et al., 2016) згідно якої визначено одинадцять критеріїв, необхідних для створення надійного еталонного набору даних. Жоден із попередніх наборів даних для IDS не міг охопити всі 11 критеріїв. Даний

датасет відповідає сучасним потребам та охоплює усі одинадцять критеріїв визначених Канадським Інститутом з Кібербезпеки.

До переліку критеріїв належать - «Повна конфігурація мережі», дане поняття наявність в мережі наступних класів пристроїв та програмного забезпечення: модем, брандмауер, комутатори, маршрутизатори та наявність різноманітних операційних систем, таких як Windows, Ubuntu і Mac OS X.

«Повний трафік» — наявність повних даних про мережевий трафік на протязі кожної сесії. Досягнуто завдяки наявності агента профілювання користувачів і 12 різних машин у мережі жертв і реальних атак з мережі нападу.

«Повна взаємодія» полягає в тому, що дані охоплюють як внутрішні, так і між внутрішньою локальною мережею, маючи дві різні мережі та Інтернет-зв’язок.

Вимога до повного захоплення виконується оскільки було використовувано дзеркальний порт, наприклад систему перехоплення, отже увесь трафік було захоплено та записано на сервер зберігання.

Датасет містить дані про трафік з доволі широким спектром використаних мережевих протоколів. Наявні усі поширені доступні протоколи обміну інформацією, такі як HTTP, HTTPS, FTP, SSH і протоколи електронної пошти.

Різноманітність атак досягнуто за рахунок включення найпоширеніших атаки на основі звіту McAfee за 2016 рік, такі як веб-атаки, груба сила, DoS, DDoS, інфільтрація, Heart-bleed, бот і сканування, охоплені цим набором даних.

Для досягнення «неоднорідності» записано мережевий трафік від головного комутатора, дампа пам’яті та системні виклики від усіх комп’ютерів-жертв під час виконання атак.

Датасет складається з восьми файлів, кожен з яких містить дані про мережеву активність в певний день або проміжок часу за який відбувається

різна активність в мережі, на рисунку 3.1 наведено таблиця відповідності даних до часового проміжку та типи трафіку, що зустрічаються в цих даних.

Name of Files	Day Activity	Attacks Found
Monday-WorkingHours.pcap_ISCX.csv	Monday	Benign (Normal human activities)
Tuesday-WorkingHours.pcap_ISCX.csv	Tuesday	Benign, FTP-Patator, SSH-Patator
Wednesday-workingHours.pcap_ISCX.csv	Wednesday	Benign, DoS GoldenEye, DoS Hulk, DoS Slowhttptest, DoS slowloris, Heartbleed
Thursday-WorkingHours-Morning-WebAttacks.pcap_ISCX.csv	Thursday	Benign, Web Attack – Brute Force, Web Attack – Sql Injection, Web Attack – XSS
Thursday-WorkingHours-Afternoon-Infilteration.pcap_ISCX.csv	Thursday	Benign, Infiltration
Friday-WorkingHours-Morning.pcap_ISCX.csv	Friday	Benign, Bot
Friday-WorkingHours-Afternoon-PortScan.pcap_ISCX.csv	Friday	Benign, PortScan
Friday-WorkingHours-Afternoon-DDos.pcap_ISCX.csv	Friday	Benign, DDoS

Рисунок 3.1 - Відповідність частин датасету до часу та типів трафіку

3.2.2 Попередня обробка даних

Для проведення дослідження з метою побудови моделей штучного інтелекту та їх навчання використовуючи зазначений датасет проведено попередній аналіз та обробку даних з метою підготовки датасету до обробки та навчання.

Оскільки датасет містить статистичні дані за певну кількість днів та не звільнений від викидів, допоміжних даних які не представляють цінності при

навчанні, таких як, наприклад: ідентифікатор потоку, IP адреси та часові мітки, тощо — проведено попередню оброку даних з метою очищення датасету від даних, що не впливають на хід та результати досліджнення, тобто не несуть цінності при побудові моделі та її подальшому навчанні.

Оскільки датасет є фрагментованим і розбитий на частини які містять дані відповідно до дня коли вони були записані, також розрізняється вміст файлів з яких складається датасет:

- Monday-WorkingHours.pcap_ISCX.csv
- Tuesday-WorkingHours.pcap_ISCX.csv
- Wednesday-workingHours.pcap_ISCX.csv
- Thursday-WorkingHours-Morning-WebAttacks.pcap_ISCX.csv
- Thursday-WorkingHours-Afternoon-Infilteration.pcap_ISCX.csv
- Friday-WorkingHours-Morning.pcap_ISCX.csv
- Friday-WorkingHours-Afternoon-PortScan.pcap_ISCX.csv
- Friday-WorkingHours-Afternoon-Ddos.pcap_ISCX.csv

Програмну реалізацію, маніпуляції з моделями, навчання нейронних мереж, отримання графічних та числових результатів буде виконано в додатку Jupyter Notebook.

Для початку імпортуємо набір бібліотек, які потрібні для початку роботи з датасетом, а саме: pandas, nympy (лістинг 3.1):

Лістинг 3.1:

```
import pandas as pd
import numpy as np
```

Далі підготуємо дані для завантаження та створенні відповідних об’єктів pandas.DataFrame (лістинг 3.2):

Лістинг 3.2:

```
import os
dspaths = []
```

```
for dirname, _, filenames in os.walk('./dataset'):
     for filename in filenames:
          if filename.endswith('.csv'):
               pds = os.path.join(dirname, filename)
               dspaths.append(pds)
               print(pds)
```

При завантажені даних та формувані pandas.DataFrame об’єктів для подальшої обробки отримано структури даних наступних розмірностей:

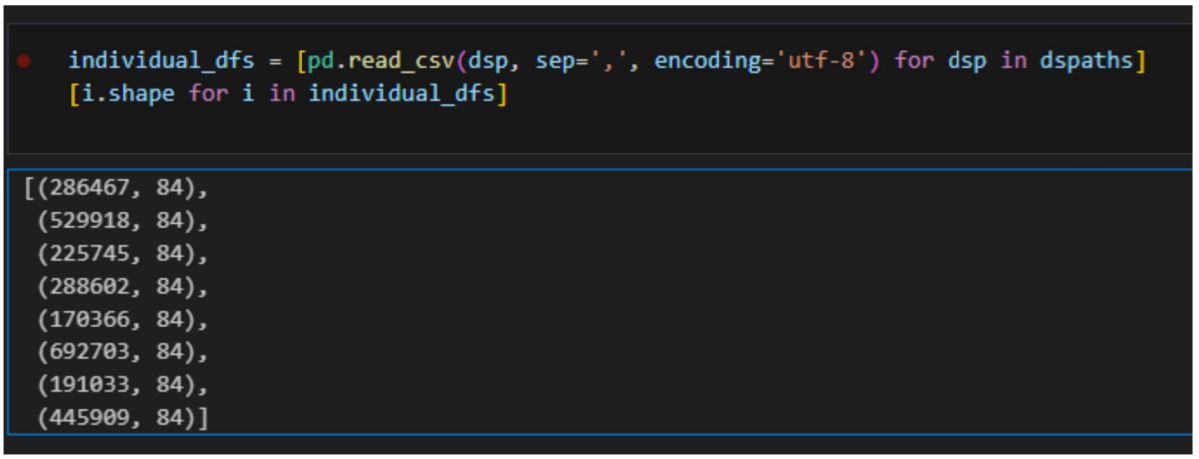

Рисунок 3.2 - Розмірності датафреймів кожного з файлів датасету

При обробці даних датасету виявлено та видалено незначущі дані, до таких даних відносяться: Flow ID, Fwd Header Length.1, Source IP, Destination IP, Destination Port та Timestamp оскільки вони є або не статичними, тобто можуть змінюватися в залежності від мережі або ж взагалі є метаданими, наприклад: Timestamp і не несуть цінності при навчанні моделей (лістинг 3.3).

Лістинг 3.3:

```
drop_columns = [
    "Flow ID",
    'Fwd Header Length.1',
    "Source IP", "Src IP",
    "Source Port", "Src Port",
    "Destination IP", "Dst IP",
    "Destination Port", "Dst Port",
```

```
        "Timestamp",
    ]


    for df in individual_dfs:
            df.columns = df.columns.str.strip()
            df.drop(columns=drop_columns, inplace=True,
errors='ignore')
            df.rename(columns=col_name_consistency,
inplace=True)
            df['Label'].replace({'BENIGN': 'Benign'},
inplace=True)
[i.shape for i in individual_dfs]
```

В результатів видалення незначущих даних з датасету було досягнуто суттєве зниження розмірності з 84 ознак(наведено на рисунку 3.2) до 78 (наведено на рисунку 3.3).

```
[(286467, 78),
 (529918, 78),
 (225745, 78),
 (288602, 78),
 (170366, 78),
 (692703, 78),
 (191033, 78),
 (445909, 78)]
```

Рисунок 3.3 - Розміри датафреймів після видалення незначущих ознак

Кожний елемент датасету піддано downsizing`у ознак. Downsizing типів даних є однією з багатьох технік застосовуваних при попередній обробці даних. Його використання призводить до зменшення обсягу даних, які обробляються, що може призвести до кількох переваг, таких як: більш ефективнне використання пам'яті, за рахунок зменшення типів даних (наприклад, зміна з float64 на float32 або з int64 на int32) можна суттєво знизити обсяги використаної пам’яті, що є суттєво при використанні значних обсягів навчальних даних. Менші структури даних можуть покращити швидкість виконання операцій над датасетом, оскільки менше пам'яті

потрібно завантажувати в оперативну пам'ять, а також зменшується обсяг даних, які потрібно обробляти. Використання більш специфічних типів (наприклад, «категорії» для категоріальних змінних) може зменшити ймовірність помилок, оскільки такі типи даних є більш зрозумілими та більш релевантними до змісту. А у випадках, коли дані передаються через мережу або зберігаються на диску, менші типи можуть знизити затримки при зчитуванні та запису, що покращує загальну продуктивність системи.

Downsizing типів даних є важливим кроком в обробці даних[15], який за допомогою невеликої кількості простих операцій може суттєво знизити об’єми необхідної пам’яті для розміщення датасету в пам’яті, а також підвищити ефективність передачі даних по мережі, за потреби, адже загальний розмір даних для передачі стає меншим, що є критично важливим для аналітики та машинного навчання.

Для застосування техніки downsizing використано бібліотеку fastcore та її компонент paralell. Для використання — необхідно завантажити бібліотеку до проекту (лістинг 3.4):

Лістинг 3.4:

```
from fastcore.parallel import *
individual_dfs = parallel(f=df_shrink, items=individual_dfs,
progress=True)
print(individual_dfs[0].dtypes)
```

```
Protocol                      int8
Flow Duration                int32
Total Fwd Packets            int16
Total Backward Packets       int16
Fwd Packets Length Total     int32
                               ...
Idle Mean                  float32
Idle Std                   float32
Idle Max                     int32
Idle Min                     int32
Label                     category
Length: 78, dtype: object
```

Рисунок 3.4 - Результат використання downsizing`y

Важливим кроком для покращення якості датасету та навчання — є усунення нульових або NaN значень. Важливо зазначити, що не завжди відсутність значення є шумом, який варто усунути з набору даних як найшвидше. В залежності від задачі, домену та природи ознаки, яка має нульове значення — ознака може приймати нульове або NaN значення. Проте у випадку з комп'ютерними мережами та встановленням сесій, порожні записи — визначенні беззмістовними та підлягають видаленню (лістинг 3.5).

Лістинг 3.5:

```
for df in individual_dfs:
            df.replace([np.inf, -np.inf], np.nan,
inplace=True)
            df.dropna(inplace=True)
[i.shape for i in individual_dfs]
```

Отримано pandas.DataFrame об'єккти даної розмірності після застосування downsizing`у для зниження розмірності ознак та видалення нульових записів:

```
[(286096, 78),
 (529481, 78),
 (225711, 78),
 (288395, 78),
 (170231, 78),
 (691406, 78),
 (190911, 78),
 (445645, 78)]
```

Рисунок 3.5 - Розмірність датафреймів після видалення нульових записів

Видалення дублікатів з датасету є важливим етапом в обробці даних, оскільки наявність повторюваних записів може негативно вплинути на результати аналізу, навчання моделей і прийняття рішень. Основною проблемою є то що, дублікатні дані можуть призводити до спотворення статистичних показників, оскільки вони створюють ілюзію більшої

представленості певних спостережень, що може призвести до некоректних висновків. Наприклад, якщо модель навчена на даних серед який присутня велика кількість дублікатів — вона може віддавати перевагу цим записам, що вплине на її загальну продуктивність і точність.

Видалення дублікатів дозволяє, окрім безпосереднього впливу на результат, також допомагає оптимізувати використання пам'яті та пришвидшити виконання операцій, що особливо важливо при роботі з великими обсягами інформації.

Варто зазначити, що дублікатні дані можуть ускладнювати інтерпретацію результатів. Без дублікатів дані стають більш прозорими і зрозумілими, що покращує якість аналізу й підвищує рівень впевненості у висновках.

Тим не менш, важливо відзначити, що у певних випадках залишення дублікатів у датасеті може бути корисним, якщо вони відображають частоту подій або популярність явищ, що важливо для коректного аналізу. У задачах з часовістю чи поведінковими даними дублікатні записи допомагають вивчати повторювані дії.

Отже, видалення дублікатів з датасету є важливим кроком у забезпеченні точності, ефективності та зрозумілості даних, що в свою чергу підвищує якість прийнятих рішень на основі цих даних. Оскільки метою дослідження не є вивчення частоти певних подій чи передбачення виникнення загроз зважаючи на частоту тих чи інших подій на часовому проміжку — дублікати записів прийнято рішення видалити з навчальної вибірки.

Для видалення дублікатів записів виконується наступний код (лістинг 3.6):

Лістинг 3.6:

```
for df in individual_dfs:
        print(df.duplicated().sum(), "Total duplicate rows removed")
```

```
            df.drop_duplicates(inplace=True)
            df.reset_index(inplace=True, drop=True)
[i.shape for i in individual_dfs]
```

За результатами проведеного обробки даних отримано наступні DataFrame об’єкти для подальшого аналізу та навчання:

```
4447 Total duplicate rows removed
166574 Total duplicate rows removed
14873 Total duplicate rows removed
70650 Total duplicate rows removed
80765 Total duplicate rows removed
14411 Total duplicate rows removed
55931 Total duplicate rows removed
106415 Total duplicate rows removed
```

Рисунок 3.6 - Результат видалення дублікатів існуючих записів

```
[(286096, 78),
 (529481, 78),
 (225711, 78),
 (288395, 78),
 (170231, 78),
 (691406, 78),
 (190911, 78),
 (445645, 78)]
```

Для зручності подальшої взаємодії з датасетом його переформатовано з CSV формату до Parquet, оскільки Parquet є колонним форматом зберігання даних, який значно ефективніше використовує простір завдяки вбудованому стисненню, що суттєво зменшує розмір файлів у порівнянні з CSV. Це дозволяє знизити витрати на зберігання даних, особливо для великих обсягів інформації.

Окрім цього, Parquet забезпечує швидший доступ до даних. Завдяки колонній структурі він дозволяє вибірково зчитувати лише потрібні стовпці, що особливо корисно при роботі з великими датасетами. Це зменшує обсяг даних, які потрібно передати у пам'ять, і значно пришвидшує аналітичні обчислення. Переформатування виконанно використовуючи вбудовані функції pandas.DataFrame (лістинг 3.7):

Лістинг 3.7:

```
    for i, df in enumerate(individual_dfs):
              df.to_parquet(f"dataset{dspaths[i].split('/')[-
1].replace('.csv', '.parquet')}")
```

Порівняння швидкості завантаження одного файлів датасету:

CSV:

```
    %timeit pd.read_csv('dataset\CIC-IDS-2017\Bruteforce-
Tuesday-WorkingHours.pcap_ISCX.csv', encoding='utf-8')
```

Результат: 4.27 s ± 243 ms per loop (mean ± std. dev. of 7 runs, 1 loop each)

Parquet:

```
    %timeit pd.read_parquet('dataset\CIC-IDS-2017\parquets\
Bruteforce-Tuesday-WorkingHours.pcap_ISCX.parquet')
```

Результат:124 ms ± 14.1 ms per loop (mean ± std. dev. of 7 runs, 1 loop each)

Також останнім кроком обробки даних є конкатенація усіх попередньо оброблених компонентів(файлів) датасету в єдиний датасет для подальшого аналізу та навчання (лістинги 3.8 та 3.9).

Лістинг 3.8:

```
    df_data_1 = pd.read_parquet('dataset\CIC-IDS-2017\parquets\
Benign-Monday-no-metadata.parquet')
    df_data_2 = pd.read_parquet('dataset\CIC-IDS-2017\parquets\
Botnet-Friday-no-metadata.parquet')
    df_data_3 = pd.read_parquet('dataset\CIC-IDS-2017\parquets\
Bruteforce-Tuesday-no-metadata.parquet')
    df_data_4 = pd.read_parquet('dataset\CIC-IDS-2017\parquets\
DDoS-Friday-no-metadata.parquet')
    df_data_5 = pd.read_parquet('dataset\CIC-IDS-2017\parquets\
DoS-Wednesday-no-metadata.parquet')
    df_data_6 = pd.read_parquet('dataset\CIC-IDS-2017\parquets\
Infiltration-Thursday-no-metadata.parquet')
    df_data_7 = pd.read_parquet('dataset\CIC-IDS-2017\parquets\
Portscan-Friday-no-metadata.parquet')
    df_data_8 = pd.read_parquet('dataset\CIC-IDS-2017\parquets\
WebAttacks-Thursday-no-metadata.parquet')
    df_data = pd.concat([df_data_1, df_data_2, df_data_3,
df_data_4, df_data_5, df_data_6, df_data_7, df_data_8], axis=0,
ignore_index=True)
    print('Shape of Dataframe: ',df_data.shape,'\n')
    print('Inspection of Target Feature - y:\n')
```

```
print(df_data['Label'].value_counts())
```

Отримано датасет наступної розмірності: (2231806, 78).

Лістинг 3.9:

```
Inspection of Target Feature - y:
Label
Benign 1895314
DoS Hulk 172846
DDoS 128014
DoS GoldenEye 10286
FTP-Patator 5931
DoS slowloris 5385
DoS Slowhttptest 5228
SSH-Patator 3219
PortScan 1956
Web Attack  Brute Force 1470
Bot 1437
Web Attack  XSS 652
Infiltration 36
Web Attack  Sql Injection 21
Heartbleed 11
Name: count, dtype: int64
```

3.2.3 Аналіз даних

За для оцінки співвідношення типів трафіку в наборі даних виконано бінаризацію цільової ознаки, що дозволить оцінити співвідношення типів трафіку розподіливши їх на відповідні категорії: «легітимний/нормальний» та «зловмисницький»(лістинг 3.10):

Лістинг 3.10:

```
X = df_data.copy()
X = X.drop('Label', axis=1)
y = df_data['Label'].copy()
y = y.map({'Benign': 0}).fillna(1)

plt.figure(figsize=(8, 6))
sns.countplot(x=y, palette='Set1')
plt.title('Distribution of Binary Target Variable')
plt.xlabel('Target Class')
plt.ylabel('Count')
```

```
    plt.xticks(ticks=[0, 1], labels=['Class 0: Benign', 'Class
1: Malicious'])
    plt.show()

    class_counts = y.value_counts()
    class_ratios = class_counts / len(y)

    print(f"Class 0 ratio: {class_ratios[0]*100:.2f}%")
    print(f"Class 1 ratio: {class_ratios[1]*100:.2f}%")
```

Бінаризація багатокласової цільової ознаки має як переваги, так і недоліки. До позитивних сторін такого підходу можна віднести те, що це спрощує задачу, адже заміна багатокласової проблеми на серію бінарних класифікацій робить процес навчання моделі менш складним.

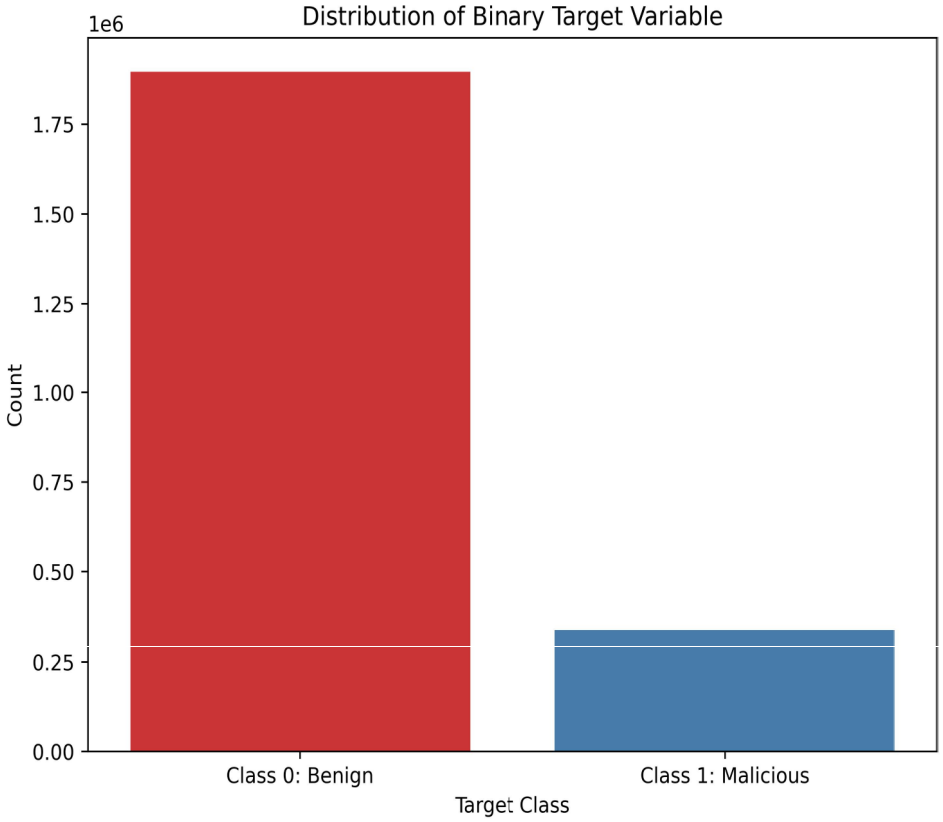

Рисунок 3.7 - Співвідношення бінаризованих ознак

Це також дає змогу використовувати алгоритми, які краще працюють із бінарними класами, наприклад, логістична регресія або ансамблеві методи. Крім того, підхід "один проти всіх" або "один проти одного" дозволяє більш точно враховувати специфіку кожного класу, що може бути корисним у

випадках значного дисбалансу між класами, як у даному випадку, оскільки відповідно до рисунку 3.7 датасет містить незбалансовані класи. Результати бінарних класифікацій також легше інтерпретувати, оскільки модель прогнозує лише один клас.

Однак бінаризація має і свої недоліки, такі як: може виникати втрата контексту між класами, адже моделі працюють окремо і не враховують взаємозв’язків між ними. Багатокласові моделі зазвичай краще узагальнюють структуру даних, тоді як бінаризація зменшує обсяг інформації, що враховується кожною моделлю.

Таким чином, бінаризація може бути корисною в задачах, де важливо спростити процес моделювання або підвищити продуктивність окремих алгоритмів, але вона вимагає додаткових ресурсів і може вплинути на точність у складних взаємозалежних даних.

3.3 Аналіз методів виявлення мережевих загроз

Розуміння розподілу класів є ключовим для оцінки ефективності моделей машинного або глибокого навчання. У цьому наборі даних клас більшості (клас 0) становить близько 84,92% зразків, тоді як клас меншості (клас 1) – лише 15,08%. Це свідчить про те, що найпростіша модель, яка завжди прогнозує клас більшості, матиме базову точність на рівні 84,92%. Відповідно, будь-яка модель машинного або глибокого навчання, створена для роботи з цим набором даних, має прагнути перевершити цей базовий рівень точності, щоб продемонструвати реальну прогностичну цінність.

3.3.1 Підготовка даних для навчання

Для навчання моделей штучного інтелекту типовим є розбиття датасету на три вибірки: «тренувальна», «валідаційна», «тестова». Такий поділ слугує кільком цілям, зокрема оцінці продуктивності моделі, налаштуванню

гіперпараметрів та оцінці узагальнення на невидимих даних. А також забезпечує належну оцінку якості моделі та запобігає її перенавчанню. Для виконання розбиття датасету використано бібліотеку scikit-learn (лістинг 3.11):

Лістинг 3.11:

```
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler,
MinMaxScaler
from sklearn.feature_selection import mutual_info_classif
from sklearn.utils.class_weight import compute_class_weight
from sklearn.metrics import accuracy_score, precision_score,
recall_score, f1_score
```

Для виконання розбиття використано наступний код (лістинг 3.12):

Лістинг 3.12:

```
fraction = p_test / (p_val + p_test)
X_val, X_test, y_val, y_test =
train_test_split(X_temp,y_temp,stratify=y_temp,
                                test_size=fraction,
                                random_state=random_state,
                                shuffle=shuffle)
```

Розбиття даних виконання в співвідношеннях: тренувальна частина — 75%, валідаційна — 10%, тестова — 15%.

3.3.2 Застосування Feature Selection

Застосування техніки відбору ознак(Feature Selection) виконано використовуючи алгоритм XGBoost. Даний алгоритм є заснованим на навчанні ансамблю дерев – алгоритм вважається «вбудованим методом відбору ознак» який сприяє в відборі найбільш значущих ознак і одночасному зменшенню розмірності вектора вхідних ознак. До значних переваг відбору за допомогою XGBoost, які роблять його ефективним і популярним інструментом у задачах машинного навчання можна віднести

наступне: автоматично виконується оцінка важливості ознак у процесі навчання моделі, що дає змогу зрозуміти, які з них найбільше впливають на результат.

XGBoost здатен працювати зі складними, високорозмірними даними. Навіть при великій кількості ознак модель може ефективно виявляти ті, які мають найбільший вплив, ігноруючи менш значущі. Це допомагає зменшити ризик перенавчання, покращити узагальнювальну здатність моделі та зменшити час її обчислення.

Взаємодія між ознаками, як одна з важливих характеристик вхідного набору даних теж приймається до уваги, враховуючи їхні нелінійні залежності. Це дозволяє краще розуміти взаємозв'язки в даних та підвищувати точність прогнозів. Завдяки вбудованим можливостям інтерпретації, таким як побудова графіків важливості ознак, можливо легко проаналізувати результати відбору та приймати обґрунтовані рішення щодо подальшого зменшення розмірності даних або оптимізації моделі.

В алгоритмах на основі дерев рішень, таких як XGBoost, оцінки важливості ознак часто виводяться з впливу ознак на зменшення невизначеностей у деревах рішень. У випадку задач класифікації однією з найпоширеніших мір домішок є невизначеність Джині.

Невизначеність Джині вимірює ймовірність того, що випадково вибраний елемент із множини буде неправильно класифікований, якщо його класифікувати згідно з розподілом міток у підмножині. У задачі бінарної класифікації, де класи представлені як 0 і 1, невизначеність Джині (G) для вузла, що містить N зразків, розраховується за формулою (3.1)[14]:

$$G=1-\sum_{i=0}^{1} p_i^2 \tag{3.1}$$

де p_i – це ймовірність належності до класу i у даному вузлі.

У процесі створення дерев рішень в алгоритмах, таких як XGBoost, у кожному вузлі вибирається ознака, яка забезпечує найбільше зменшення невизначеності Джині, і використовується для розщеплення вузла. Важливість ознаки розраховується на основі сумарного зменшення невизначеності, отриманого завдяки розщепленням за цією ознакою у всіх деревах ансамблю. Ознаки, що значно зменшують невизначеність Джині при розщепленні вузлів, визнаються алгоритмом більш важливими. Такі ознаки відбираються для подальшого аналізу та можуть використовуватися як вхідні дані для інших моделей машинного навчання, наприклад, нейронних мереж.

Результат обчислення важливості ознаки базується на ступені зменшення невизначеності Джині, яке забезпечується використанням кожної ознаки для розщеплення вузлів у деревах рішень. Ці оцінки допомагають визначити, які ознаки є найбільш інформативними для побудови прогнозів у моделі ансамблю.

Для аналізу визначенно допоміжну функцію get_feature_importances, яка виконує тренування та визначення важливості ознак (лістинг 3.13):

Лістинг 3.13:

```
    X_train_val = pd.concat([X_train, X_val], axis=0)
    y_train_val = pd.concat([y_train, y_val], axis=0)
    xgb_model = xgb.XGBClassifier(n_estimators=100, max_depth=3,
learning_rate=0.1, missing=np.inf)
    xgb_model.fit(X_train_val, y_train_val)
    feature_importances = xgb_model.feature_importances_
    if printing:
       max_feature_name_length = max(len(name) for name in
X_train.columns)
       print(f"{'Feature Name':<{max_feature_name_length}}\
tImportance")
       for name, importance in zip(X_train.columns,
                                   feature_importances):
           print(f"{name:<{max_feature_name_length}}\
t{importance*100:.2f}%")
```

У XGBoost невизначеність Джині використовується як метрика за замовчуванням для оцінки під час побудови дерев рішень. Для вузла дерева, індекс Джині (GI) розраховується за формулою (3.2):

$$GI = 1 - \left(p_1^2 + p_2^2 \right) \tag{3.2}$$

де $p_1 + p_2$ - ймовірності класів позначені (з умови $p_1 + p_2 = 1$).

Ця формула вимірює ступінь невизначеності, враховуючи квадрати ймовірностей класів. Нижчі значення вказують на краще розділення класів та підвищення якості моделі. Важливість кожної ознаки визначається на основі сумарного зниження індексу Джині.

Ознаки, які забезпечують значне зниження індексу, розглядаються як найбільш інформативні для прогнозування цільової змінної. За результатами проведеного аналізу та використання техніки Feature Selection отримано результати (лістинг 3.14):

Лістинг 3.14:

```
    X_train_selected, X_val_selected, X_test_selected,
feature_tuples = select_features_by_threshold(original_data,
threshold=0.01, printing=True)
```

Для створення відповідної інфографіки та візуалізації отриманих результатів застосовано бібліотеку matplotlib та наступний код (лістинг 3.15):

Лістинг 3.15:

```
    feature_names       = [tup[0] for tup in feature_tuples]
    significance_scores = [tup[1] for tup in feature_tuples]
    plt.figure(figsize=(10, 8))
    plt.barh(feature_names, significance_scores,
color='skyblue')
    # set labels names
    plt.xlabel('Feature Importance')
    plt.ylabel('Features')
    plt.title('Feature Importance extracted from XGBoost Feature
Selection')
    plt.gca().invert_yaxis()
    # print the plot
    plt.show()
```

З початкових 78 ознак лише 18 демонструють статистичну значущість, що перевищує 1% (наведено на рисунку 3.8). Це дозволяє зосередитися на цих важливих ознаках для покращення точності прогнозування та зменшення розмірності задачі, продовжувати працювати виключно з відібраними ознаками.

Таблиця 3.1 — Отримані результати оцінки важливості ознак

Кількість обраних ознак: 18	
Назва ознаки	Важливість
Bwd Packet Length Std	32.52%
Total Fwd Packets	8.19%
Fwd Packet Length Std	6.12%
Fwd IAT Mean	5.51%
Fwd Act Data Packets	4.80%
Avg Packet Size	4.71%
Bwd Packets/s	4.17%
Fwd Packet Length Min	3.32%
Bwd Packet Length Min	2.94%
FIN Flag Count	2.41%
Bwd Packet Length Mean	2.37%
Fwd Packet Length Max	2.07%
Flow Duration	2.04%
Fwd Header Length	1.84%
Fwd IAT Max	1.35%
Init Bwd Win Bytes	1.25%
Fwd PSH Flags	1.11%
Idle Mean	1.08%

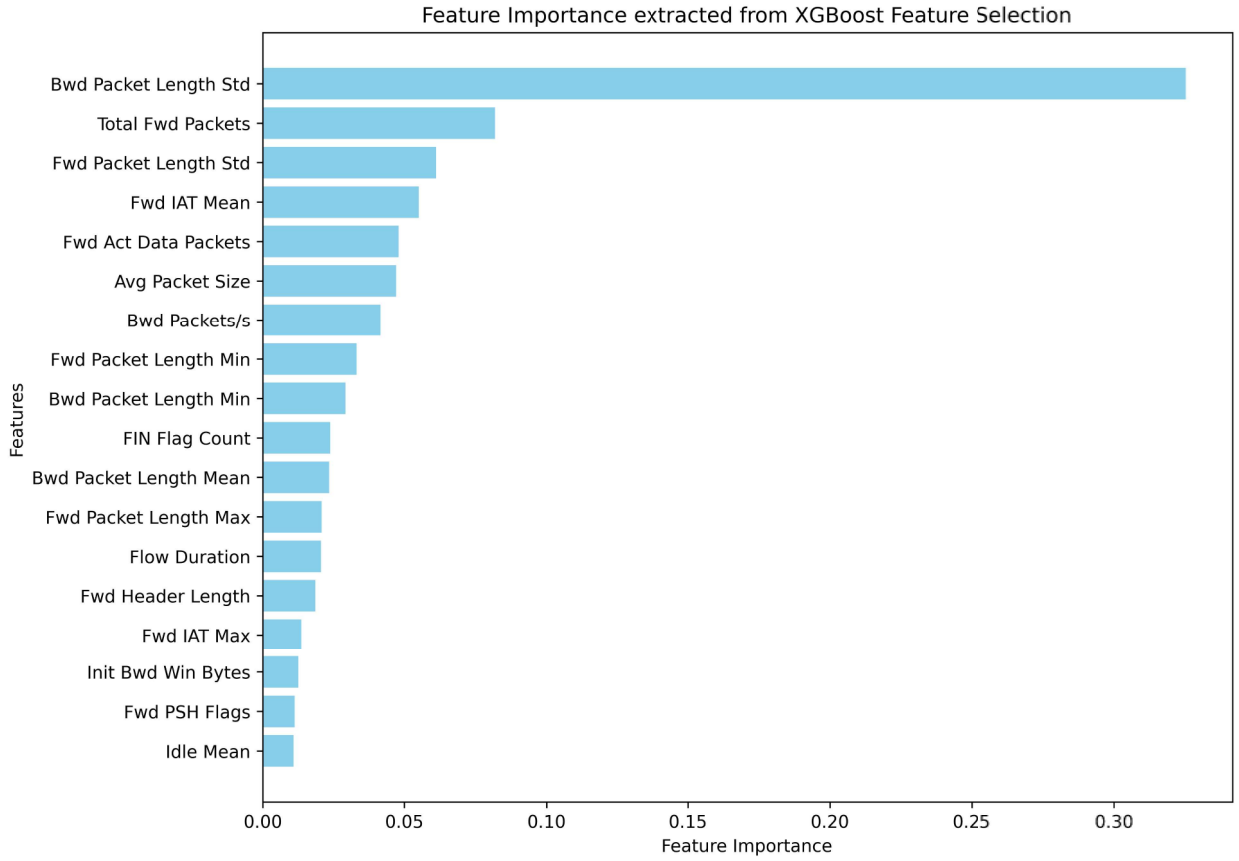

Рисунок 3.8 — Графік результатів оцінки важливості ознак XGBoost

3.3.3 Інженерія ознак

Інженерія ознак є ключовим етапом у підготовці даних для машинного навчання, використовуючи цю техніку можна суттєво збільшити інформативність, релевантність та корисність даних для моделей та їх подальшого навчання. Ця техніка відіграє вирішальну роль в оптимізації продуктивності систем глибокого навчання. Вона передбачає перетворення необроблених даних на значущі ознаки, які дозволяють моделям глибокого навчання ефективно робити точні прогнози. Наприклад, створення нових ознак, що інтегрують кілька початкових характеристик, або нормалізація числових даних можуть значно спростити навчання моделі.

В подальшому застосування цієї техніки стосується лише частини заздалегідь визначених ознак та не охоплює увесь набір з 78 початкових ознак, оскільки попередньо вже були відібрані найбільш значущі ознаки та

враховуючи, що пороговим значенням для оцінки «важливості» ознаки є 1% (у таблиці. 3.1).

Отже для подальшого аналізу та обробки обрано лише ті ознаки критерій «важливості», яких є більшим за 1%. Подальшим кроком є виведення нових ознак на основі вже наявних та відібраних ознак.

Для виведення нових ознак використано чотири підходи: «зважена оцінка ознаки» полягає в присвоєні ваг на основі оцінки важливості та підсумовування значень декількох пов'язаних ознак. Виводиться комбінована оцінка ознак на основі зваженої важливості за результатами z – нормалізації. Z-нормалізація, або стандартизація — є методом обробки числових даних, який приводить їх до стандартного нормального розподілу зі середнім значенням 0 і стандартним відхиленням 1. Даний підхід широко використовується в машинному навчанні та статистиці для масштабування змінних і забезпечення їх порівнюваності відповідно до формули (3.3).

$$z = (x - \mu)/\sigma \tag{3.3}$$

де x – початкове значення змінної;

μ – середнє значення змінної в наборі даних;

σ – стандартне відхилення змінної.

За результатами нормалізації кожне значення показує, наскільки воно відрізняється від середнього у вимірі стандартних відхилень.

Дана техніка дозволяє уникнути проблем, які можуть виникати через різні масштаби ознак на якість навчання, адже кожна ознака може мати різні одиниці виміру або діапазони, це може вплинути на роботу моделей та їх чутливість до певних значень. Для моделей, які використовують градієнтний спуск, стандартизація сприяє стабільнішій та швидшій оптимізації. До переваг застосування такого методу також можна віднести, що значення легше інтерпретувати, адже вони показують відхилення від середнього в

універсальній формі. Отже, z – нормалізація дозволяє досягти наближено однакового впливу ознак на поведінку та навчання моделі.

Для z – нормалізації використано бібліотеку scikit-learn та її клас StandardScaler (лістинг 3.16).

Лістинг 3.16:

```
    scaler = StandardScaler()
    df_scaled_train=
pd.DataFrame(scaler.fit_transform(X_train_selected),
columns=X_train_selected.columns)
    df_scaled_val=
pd.DataFrame(scaler.transform(X_val_selected),columns=X_val_sele
cted.columns)
    df_scaled_test=
pd.DataFrame(scaler.transform(X_test_selected),columns=X_test_se
lected.columns)
```

Далі розраховано зважену оцінку враховуючи «важливість» відібраних попередньо ознак (лістинг 3.17).

Лістинг 3.17:

```
df_scaled_train["Combined_Importance_Score"] =
df_scaled_train[list(feature_names)].dot(weights)
df_scaled_val["Combined_Importance_Score"]  =
df_scaled_val[list(feature_names)].dot(weights)
df_scaled_test["Combined_Importance_Score"] =
df_scaled_test[list(feature_names)].dot(weights)
```

Наступним кроком отримані значення застосовуються до підготовлених вибірок даних, тобто: тестової, валідаційної та тренувальної.

«Співвідношення ознак та відмінності» - даний метод застосовано для продукування додаткових ознак, які можуть бути внесені до переліку значущих, якщо при повторній ітерації feature selection їх показник значущості перевершить показники вже відібраних ознак. За результатами отримано 2 нові ознаки: «Співвідношення розмірів пакетів» або «Packet Size Ratio», яка є співвідношенням між «середнім розміром пакетів» і «середньою довжиною пакетів Fwd» або «середньою довжиною пакетів Bwd». Дана

ознака може слугувати для кращого відображення розподілу розмірів пакетів у потоці. Другою синтезованою ознакою є «Стандартна відмінність довжини пакета», що є різницею між стандартизованими версіями таких характеристик, як `Fwd Packet Length Std` і `Bwd Packet Length Std`. Дана ознака потенційно може виділити та додати ваги відносній різниці у варіаціях довжин пакетів у прямому і зворотному напрямках.

«Взаємодія ознак» - даний підхід до синтезу ознак полягає в тому, щоб виділити ознаки, які можуть бути взаємопов’язаними та залежати одне від одного, або їхня комбінація може додати та розширити контекст ознаки для моделі при навчанні. В результатів отримано три нові ознаки: «Кількість пакетів в прямому напрямі на тривалість потоку», що є добутком кількості пакетів в прямому напрямі на секунду в межах сесії/потоку помноженого на тривалість сесії/потоку. «Кількість пакетів в зворотному напрямі на тривалість потоку», що є добутком кількості пакетів в зворотному напрямі на секунду в межах сесії/потоку помноженого на тривалість сесії/потоку. А також «Відношення пакетів в прямому напрямі на секунду до пакетів в зворотному напрямі на секунду».

Останній підхід до генерації нових ознак є «Відсоток», тобто відсоткова репрезентація обраних ознак. Таким чином видається можливим оцінити призначення пакетів в сесії, якщо вирахувати відсоток пакетів малого/великого розміру в обох напрямах передачі даних та виділити чотири нові ознаки. Такі ознаки можуть репрезентувати характер сесії та її призначення, наприклад, велика частка великих за розміром може свідчити про завантаження великого обсягу даних, що може бути опосередкованим індикатором витоку даних або проникнення. Так само суттєва частка пакетів малого (службового) розміру, може допомогти виявити такі загрози, як: сканування портів, DDoS атака, SYN flood тощо.

За результатами проведеної інженерії ознак виведено десять нових ознак, на основі попередньо відібраних вісімнадцяти ознак, що продемонстрували «важливість» більше 1% на етапі відбору ознак.

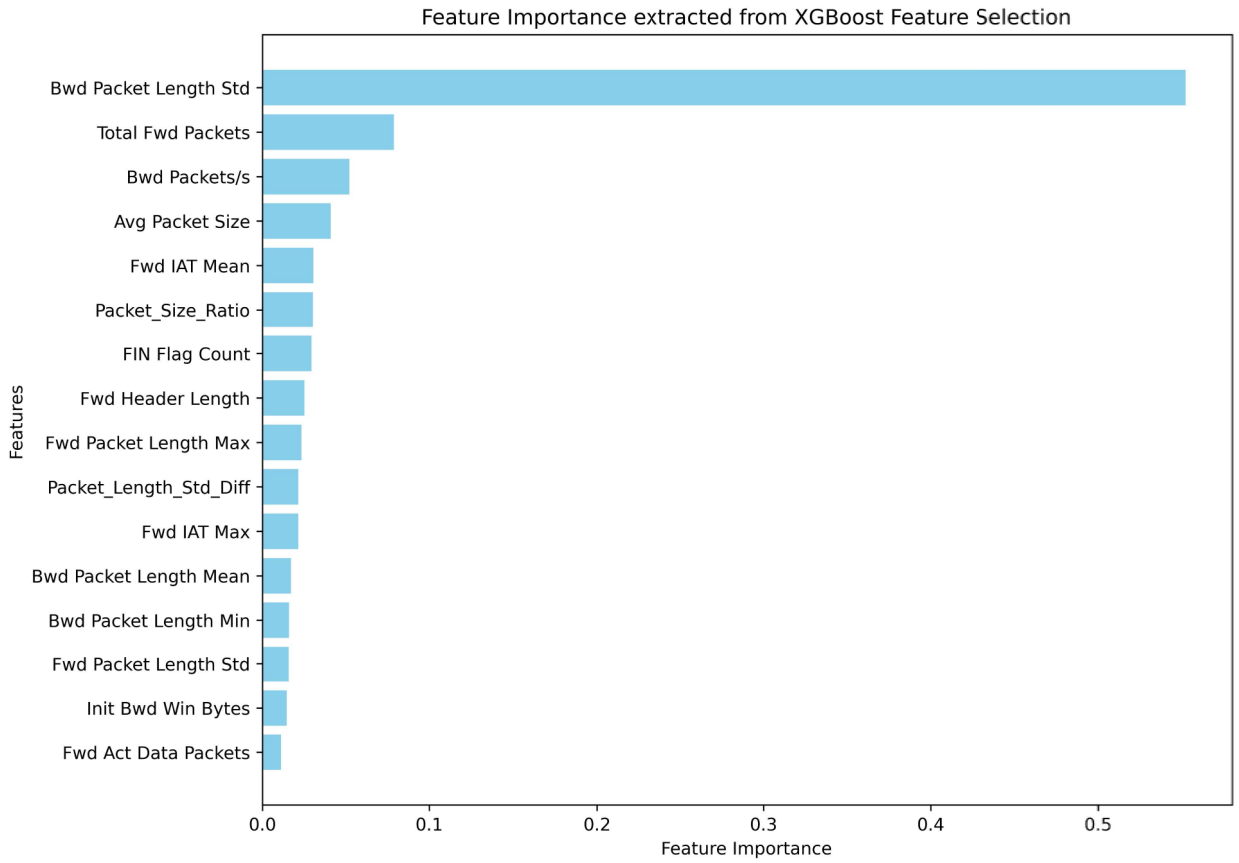

Рисунок 3.9 — Результат повторного відбору ознак

Нові ознаки створені з метою покращення ефективності та навчання моделей, проте оновлений набір ознак має пройти повторний процес відбору ознак за для підтвердження їх значущості перед додаванням до вихідної розмірності простору ознак. Повторним відбором ознак відібрано 16, тобто тих чия «важливість» сягає більше за порогове значення в 1%. Повторний відбір виконано у відповідності до першого, використовуючи алгоритм XGBoost.

Після повторного відбору ознак необхідно повторно провести процедуру масштабування усіх значень з переліку остаточного набору відібраних ознак для покращення збіжності моделі та зменшення впливу ознак які мають більший масштаб значень в силу природи свого походження та привести усі значення до спільного масштабу.

Таблиця 3.2 містить результати другої ітерації відбору ознак, а на рисунку 3.9 візуалізовано отримані результати. У випадку наявності

розбіжностей в масштабі — модель може в результаті віддавати перевагу саме тим параметрам ознаки, що мають більший масштаб при прийнятті рішення, адже вони мають більший вплив на параметри моделі при навчанні.

Таблиця 3.2 — Отримані результати повторної оцінки важливості ознак

Кількість обраних ознак: 16	
Назва ознаки	Важливість
Bwd Packet Length Std	55.20%
Total Fwd Packets	7.89%
Bwd Packets/s	5.24%
Avg Packet Size	4.13%
Fwd IAT Mean	3.03%
Packet_Size_Ratio	3.00%
FIN Flag Count	2.92%
Fwd Header Length	2.49%
Fwd Packet Length Max	2.32%
Packet_Length_Std_Diff	2.14%
Fwd IAT Max	2.13%
Bwd Packet Length Mean	1.70%
Bwd Packet Length Min	1.59%
Fwd Packet Length Std	1.56%
Init Bwd Win Bytes	1.44%
Fwd Act Data Packets	1.10%

3.4 Побудова моделей для дослідження

Для виконання цілей роботи та проведення дослідження обрано наступні типи архітектур нейронних мереж: «багатошаровий перцептрон» або «MLP» та «згорткова нейронна мережа» або «CNN». Моделі мають ціль бути навченими розрізняти аномалії в мережевому трафіку та генерувати інформацію яка може бути в результаті бути використана, більшою системою[11], наприклад, системою виявлення вторгнень(IDS) для роботи в мережі та генерації попереджень про виявленні аномалії в мережевому трафіку спеціалістів з кібербезпеки.

3.4.1 Моделі на базі архітектури MLP

Використовуючи архітектуру MLP для проведення оцінювання якості обраного підходу для даного дослідження обрано розробити дві моделі.

Модель №1 — small, мінімалістична модель з відносно невеликою кількістю параметрів метою якої є досягнення максимальної можливої ефективності в виявленні аномалій при мінімальних вимогах до цільової системи. Оскільки модель на базі штучного інтелекту не формує собою кінцеву систему, яка є самодостатньою кінцевою системою — слід враховувати, що середовище в якому моделі належить існувати матиме обмежені ресурси тож модель ШІ має бути ефективною з точки зору вимог до апаратних потужностей системи, наскільки це можливо. З точки зору вимог до даної моделі потенційним зниженням точності, в межах допустимого, вважається доцільно знехтувати в обмін на зниження апаратних вимог до цільової системи.

Моделі на архітектурі MLP побудовані використовуючи Keras Sequential API[13]. Мінімалістична модель складається з п’яти шарів, включно з вхідним шаром.

При побудові моделі важливим параметром є функції активації, що застосовуються на вхідному, вихідному та проміжних шарах моделі. Для мінімалістичної моделі обрано використовувати Sigmoid, відпвідає формулі (3.4), як функцію активації для вхідного шару(у таблиці 3.3). Дана функція добре підходить для нормалізації даних на вхідному шарі — діапазон 0 — 1.

$$f(x)=1/1+e^{-x} \tag{3.4}$$

де x – вхідне значення нейрона;

e - основа натурального логарифма.

Оскільки на прихованих шарах застосовується функція активації relu, яка описується формулою (3.5), мінімальне значення якої є 0, то дана комбінація функцій активації при переході між вхідним шаром та першим проміжним є плавною та запобігає втраті даних або точності при навчані.

$$f(x)=max(0,x) \tag{3.5}$$

де x – вхідне значення нейрона;

Використання даної функції на проміжних шарах також зумовлено її простотою та ефективністю, що особливо цінно для мінімалістичної моделі, а також дана функція має перевагу через її уникати проблеми згасання градієнтів, яка часто виникає з Sigmoid чи Tanh.

Для навчання моделей обрано наступні параметри: використано техніку ранньої запинки[12] за для відстеження покращення метрик якості при навчанні та можливості перервати навчання коли модель вийшла на плато. В якості метрики якості для ранньої зупинки використано validation accuracy, у випадку відсутності покращень цієї метрики протягом десяти епох — навчання буде зупинено та відновлено найкращі отримані ваги.

Використано наступний код для визначення ранньої зупинки (лістинг 3.18):

Лістинг 3.18:

```
early_stopping = EarlyStopping(
monitor='val_accuracy',
patience=10,
verbose=1,
restore_best_weights=True)
```

Додатково використано техніку зниження Learning rate'у при виході на плато. Даний підхід дозволяє поступово знайти точку оптимуму при оптимізації функції, навіть якщо оптимум є локальним. Використано наступний код для визначення зниження параметра Learning rate (лістинг 3.19):

Лістинг 3.19:

```
reduce_lr = ReduceLROnPlateau(
monitor='val_accuracy',
patience= 5,
min_lr=1e-07,
verbose=1,
factor=0.1)
```

В якості оптимізатора використано «Adam», який є одиним із найпопулярніших оптимізаторів для навчання нейронних мереж завдяки своїм властивостям, таким як: адаптивна швидкість навчання, що дозволяє швидше зближуватися до оптимуму навіть у задачах із високою динамікою градієнтів, даний оптимізатор є стійким до шуму та має прості гіперпараметри. Бінарна крос-ентропія використовується як функція втрат, за формулою (3.6).

$$L = -\left(\frac{1}{N}\right)\sum\left[y_i * \log\left(\hat{y}_i\right) + \left(1 - y_i\right) * \log\left(1 - \hat{y}_i\right)\right] \quad (3.6)$$

де y – істина мітка класу;

$\hat{y}$ - передбачена ймовірність приналежності до класу 1.

Перед початком навчання обов’язковим є зменшення впливу на процес навчання такого фактору, як незбалансованість класів, згідно рисунку 3.7 класи є незбалансованими на користь нормального трафіку в співвідношенні 3:1 порівняння з зловмисницьким трафіком, що відображає більшість реальних ситуацій проте може мати негативні наслідки при навчанні, адже модель буде схилятися до виявлення нормального трафіку для більшості вхідних даних.

Тому необхідно розрахувати ваги для класів використовуючи можливості бібліотеки scikit-learn та метод compute_class_weight. За результатами отримано наступний результат (лістинг 3.20):

Лістинг 3.20:

```
[Weights]: {class 0: 0.588769, class 1: 3.316283}
```

Таблиця 3.3 — Характеристики мінімалістичної MLP моделі

Тип шару	Вихідна форма	Кількість параметрів	Функція активації
Dense	(None, 8)	136	sigmoid
Dense	(None, 4)	36	relu
Dense	(None, 2)	10	relu
Dense	(None, 1)	3	sigmoid
Кількість параметрів		185	
Загальна кількість параметрів для тренування		185	

За результатами навчання модель демонструє високі значення метрик якості під час навчання, а також суттєво мінімізовані втрати. Отримано наступні значення метрик якості за якими проводилася оцінка для мінімалістичної моделі (наведено на рисунку 3.10). Схема відображає архітектуру мінімалістичної нейронної мережі (наведено на рисунку 3.10), що складається з чотирьох повнозв'язних (dense) шарів.

Таблиця 3.4 - Метрики якості мінімалістичної моделі

Метрика	Відсоток	Значення
Test Loss	-	0.0490
Test Accuracy	98.52%	0.9852
Test Precision	96.63%	0.9663
Test Recall	93.42%	0.9342
Test F1 Score	95.00%	0.9500

Кожен шар має відповідну активаційну функцію, а також зазначені розміри вхідного та вихідного простору.

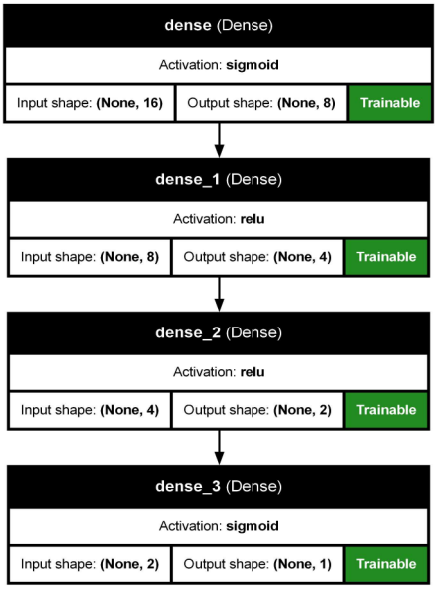

Рисунок 3.10 — Графічне зображення архітектури мінімалістичної MLP моделі

Усі шари позначені як Trainable, що означає, що їх параметри підлягають оптимізації під час навчання. Архітектура має поступове зменшення розмірності, що сприяє вилученню високорівневих ознак із вхідних даних. Використання функцій активації relu на прихованих шарах сприяє нелінійності та покращує здатність моделі до навчання складних залежностей. Фінальна функція активації sigmoid вказує на те, що модель

призначена для вирішення задачі бінарної класифікації, оскільки її вихід значення лежить у діапазоні [0, 1].

Архітектура є простою та добре підходить для задач із невеликою кількістю вхідних ознак і помірною складністю. Графіки навчання та валідації демонструють процес оптимізації моделі протягом 60 епох, навчання було зупинено через ранню зупинки, оскільки в подальшому покращення метрики accuracy не відбувалося.

На верхньому графіку, який відображає втрати для тренувального та валідаційного наборів даних, можна спостерігати стрімке зниження втрат на початкових етапах навчання, що є характерним для етапу швидкої адаптації моделі до даних. Після десятої епохи темп зниження втрат сповільнюється, і близько сорокової епохи криві тренувальних і валідаційних втрат стабілізуються. Стабільність значень втрат для обох наборів даних свідчить про високу узгодженість моделі, що вказує на відсутність перенавчання.

На нижньому графіку, що ілюструє динаміку точності моделі, простежується аналогічна тенденція. На початкових етапах (0–10 епохи) спостерігається швидке зростання точності для тренувального та валідаційного наборів.

У подальшому темп зростання уповільнюється, і після сорокової епохи точність досягає плато, стабілізуючись на рівні близько 98,3%. Відсутність розбіжності між кривими тренувальної та валідаційної точності є свідченням здатності моделі ефективно узагальнювати результати на нових даних.

Узагальнюючи результати можна відзначити, що результати навчання демонструють високу якість моделі, підтверджену низькими втратами та високою точністю як на тренувальному, так і на валідаційному наборах. Стабільність кривих після сорокової епохи свідчить про досягнення оптимальної продуктивності, і подальше збільшення кількості епох, ймовірно, не призведе до суттєвого покращення.

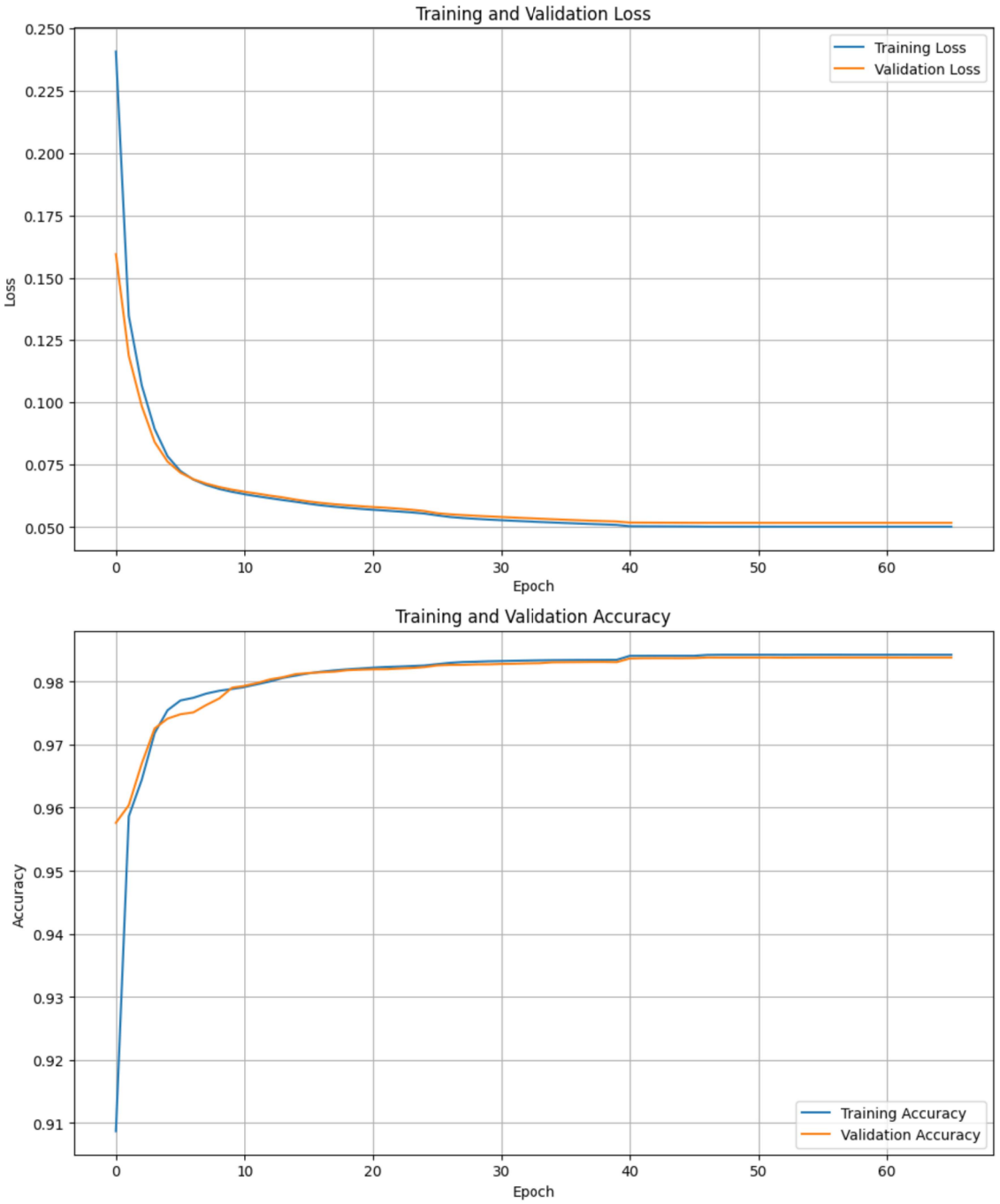

Рисунок 3.11 — Результати навчання мінімалістичної моделі

Таким чином, представлені графіки підтверджують, що модель з описаною архітектурою демонструє збалансовану продуктивність на тренувальних і валідаційних даних. Зниження розмірності на кожному шарі дозволило виділити найбільш значущі ознаки, тоді як обрані функції активації забезпечили необхідну нелінійність, що є критичною для вирішення задач класифікації.

Модель №2 — large, велика модель яка має значно більшу кількість параметрів за мінімалістичну модель та на відміну від неї має на меті досягати максимальної точності при роботі з даними та виявлені аномалій нехтуючи ефективністю цільової системи.

Зважаючи на розмір моделі вона має підвищені вимоги до апаратних потужностей цільової системи до якої буде інтегрована. Дана модель є значно складнішою порівняно з попередньою версією, що має допомогти їх в вирішенні задач із підвищеною складністю.

Таблиця 3.5 - Характеристики великої MLP моделі

Тип шару	Вихідна форма	Кількість параметрів	Функція активації
Dense	(None, 128)	2176	tanh
Dense	(None, 64)	8256	selu
Dense	(None, 32)	2080	selu
Dense	(None, 16)	528	selu
Dense	(None, 8)	136	selu
Dense	(None, 4)	36	selu
Dense	(None, 2)	10	selu
Dense	(None, 1)	3	sigmoid
Загальна кількість параметрів		13255	
Загальна кількість параметрів для тренування		13255	

Архітектура на рисунку Б.3 складається з одинадцяти повнозв'язних (dense) шарів, кожен із яких виконує специфічну роль у перетворенні та аналізі вхідних даних. Вхідна форма для моделі має розмірність (None, 16), а вихід складається з одного нейрона з використанням функції активації sigmoid, що є оптимальним для задач бінарної класифікації.

В якості функції активації використано функцію активації tanh, яка дозволяє перетворювати дані у симетричний діапазон [-1, 1]. Це підходить для обробки даних, розподілених навколо нуля, що сприяє кращій оптимізації на початкових етапах навчання. Цей шар також виконує розширення вхідного простору до розмірності 128, що дозволяє захопити більшу кількість взаємозв'язків між ознаками вхідних даних.

Подальші шари використовують функцію активації selu (Scaled Exponential Linear Unit), яка автоматично виконує нормалізацію даних під час проходження через глибокі шари. Використання selu сприяє стабільності навчання, зменшуючи ризик затухання градієнтів, що часто виникає в багатошарових нейронних мережах. Ці шари поступово зменшують розмірності, починаючи з 128 і закінчуючи 2, що дозволяє виділити найбільш релевантні ознаки для остаточного класифікаційного рішення.

Останній шар, використовує функцію активації sigmoid, яка обмежує вихідні значення в інтервалі [0, 1]. Це необхідно для задачі бінарної класифікації, де ймовірність належності до одного з класів є кінцевою метою моделі. Вихідна форма цього шару має розмірність (None, 1), що узгоджується із завданням передбачення одиничного результату.

Порівняно з попередньою моделлю, ця архітектура є глибшою, що дозволяє обробляти дані зі складнішою структурою або вищою кількістю ознак. Замість функції активації relu, яка використовувалась у прихованих шарах попередньої моделі, тут застосовано selu, що забезпечує додаткову стабільність і автоматичну нормалізацію. Використання функції tanh на першому шарі є важливою відмінністю, яка сприяє більш ефективній роботі з даними, розподіленими навколо нуля.

Графік (наведено на рисунку 3.12) втрат демонструє, що функція втрат як на навчальній, так і на валідаційній вибірках зменшується протягом перших епох навчання, що свідчить про ефективне налаштування ваг нейронної мережі. Однак, починаючи з певної епохи, криві втрат на навчальній та валідаційній вибірках починають розходитися. Це типова

ознака перенавчання моделі, коли модель занадто добре запам'ятовує особливості навчальної вибірки і починає погано узагальнювати на нові, невидимі дані.

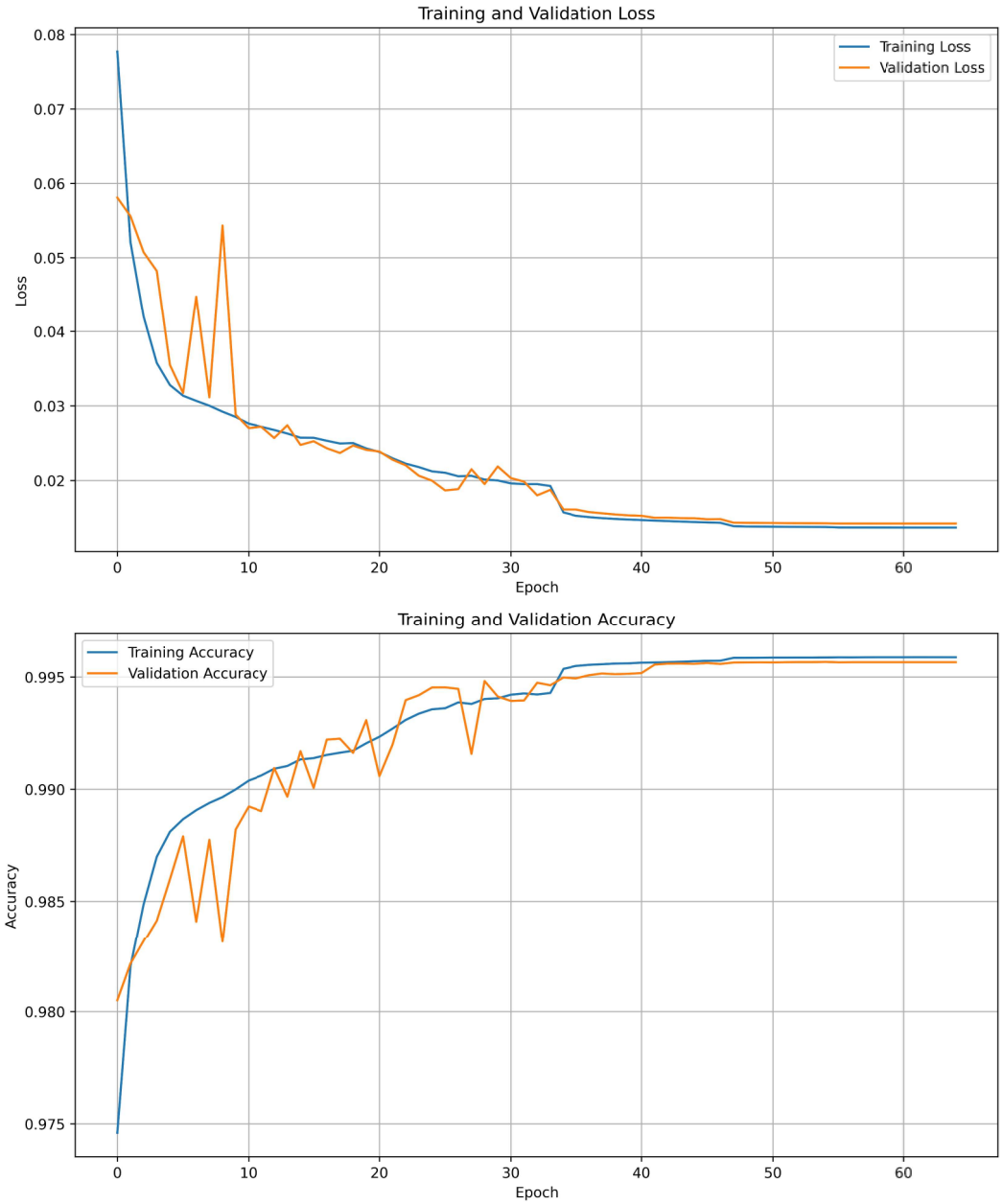

Рисунок 3.12 - Інфографіка навчання та валідації великої MLP моделі

Графік точності підтверджує висновки, з аналізу графіків втрат. Точність моделі на навчальній вибірці продовжує зростати протягом усього процесу навчання, тоді як точність на валідаційній вибірці досягає максимуму на певній епосі і потім починає дещо знижуватися. Це ще раз вказує на перенавчання моделі.

Отримані результати свідчать про те, що модель успішно навчилася виявляти аномалії в мережевому трафіку на навчальній вибірці. Однак, модель демонструє ознаки перенавчання, що може негативно вплинути на її здатність узагальнювати на нові дані.

3.4.2 Моделі на базі архітектури CNN

В межах дослідження також розглянуто доцільність використання моделей штучного інтелекту використовуючи архітектуру згорткова нейронна мережа.

Для навчання та побудови нейронних мереж використано напрацювання попередніх розділів та використано попередньо оброблений датасет. Також використано оптимізатор «adam», який було застосовано також для навчання MLP моделей та бінарну кроссентропію, як функцію мінімізації втрат, в якості метрики оцінки обрано accuracy. Відповідно до попередніх експериментів використано техніки раньої зупинки та зменшення learning rate’у для покращення навчання.

Для проведення дослідження обрано підхід grayscale з використанням згорткових нейронних мереж. Основою цього підходу є перетворення мережевих даних у вигляді сирого трафіку або агрегованих характеристик у зображення. Кожен пакет або послідовність пакетів може бути представленою у вигляді матриці, що описують мережевий сеанс. Ця матриця може бути масштабована та перетворена у зображення в градаціях сірого (grayscale), що дає змогу використовувати CNN для класифікації або ідентифікації аномалій.

Згорткові нейронні мережі є потужним інструментом для виявлення патернів у даних завдяки здатності автоматично екстрагувати релевантні особливості з вхідного зображення. У цьому контексті CNN дозволяють виділяти характерні риси аномального трафіку, які можуть бути важко

виявлені традиційними методами аналізу[11], що базуються на експертних правилах або ручній інженерії ознак.

Перевагою підходу є можливість ефективної роботи з великими масивами даних та автоматичне адаптування моделі до нових типів загроз. Однак важливим аспектом є підготовка якісного набору даних для тренування моделі. Він повинен містити різноманітні сценарії нормального трафіку та приклади аномалій, щоб забезпечити здатність моделі до узагальнення.

Таким чином, використання CNN із grayscale-представленням трафіку є перспективним напрямом досліджень у галузі кібербезпеки, який поєднує передові технології машинного навчання з інноваційними способами обробки мережевих даних.

Побудовано три моделі, кожна з яких збільшує комплексність архітектури та ускладнює її з метою дослідити здатність моделей на базі цієї архітектури бути навченими для виявлення загроз в мережі знаходячи шаблони в вхідних даних.

Модель №1 — мінімальна є найменшою (на рисунку Б.3) за кількістю параметрів моделлю — 164257 тренувальних параметрів. Вхідний шар отримує дані з розмірністю (None, 77), де None вказує на змінну кількість зразків у батчі, а 77 - це розмірність векторного представлення кожного зразка. Послідовність згорткових шарів (Conv2D) виконує згортку для виявлення локальних ознак у даних. Застосовується кілька шарів згортки з різною кількістю фільтрів, що дозволяє виявляти все більш абстрактні ознаки.

Шари пулінгу (MaxPooling2D) зменшують розмірність даних шляхом вибору максимального значення в кожному підмасиві, що дозволяє зберегти найбільш важливі ознаки та зменшити обчислювальну складність. Шар сплескування перетворює вихідний тензор згорткових шарів у одномірний вектор, який подається на вхід пов'язаним шарам. Пов'язані шари (Dense) виконують лінійну трансформацію вхідних даних і додають зміщення.

Шар Dropout випадково відключає деякі нейрони під час тренування, що допомагає запобігти перенавчанню. Вихідний шар видає кінцевий результат, який, ймовірно, є ймовірністю належності до певного класу (якщо це задача класифікації).

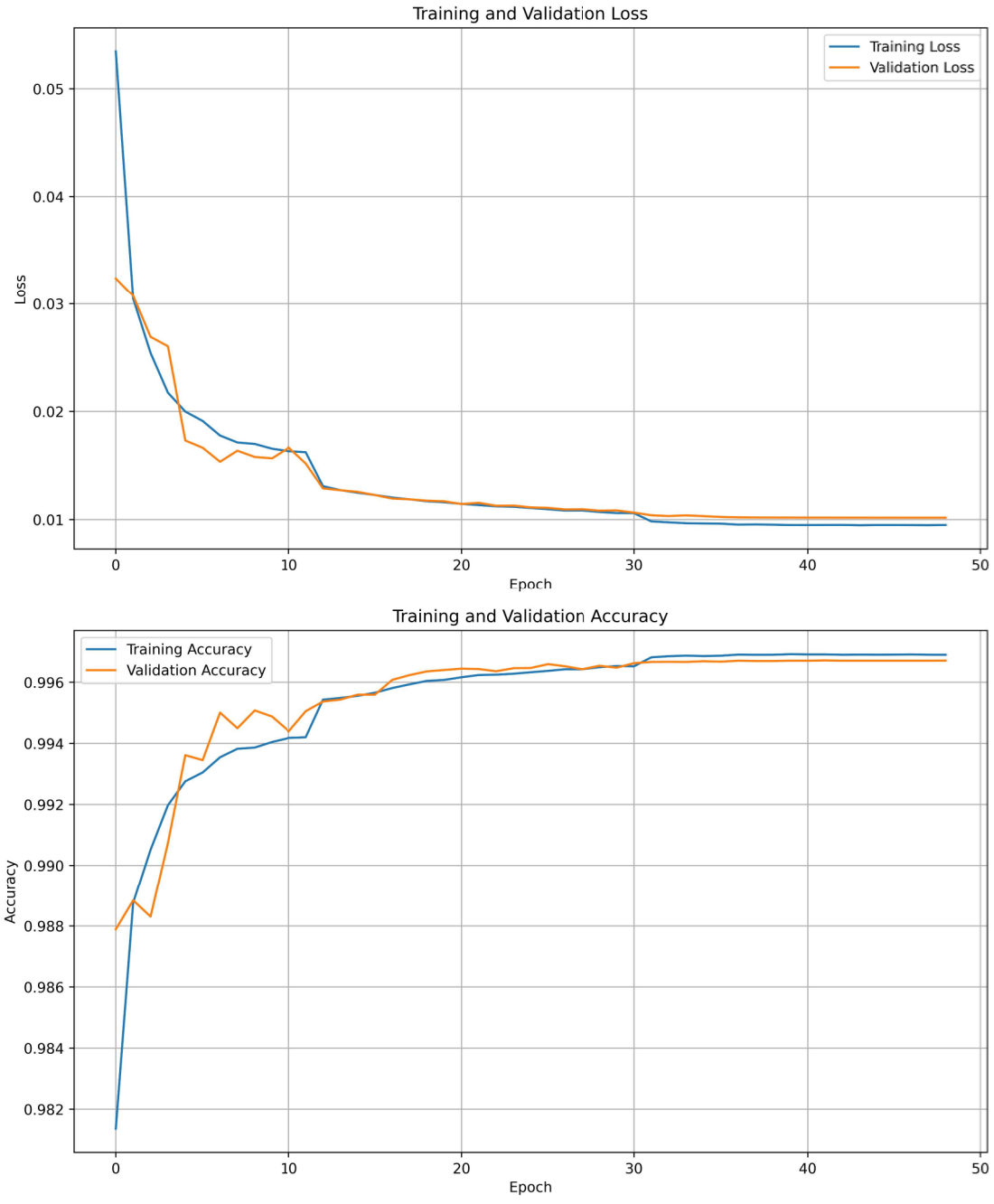

Рисунок 3.13 - Інфографіка навчання мінімалістичної моделі CNN

Потенційними перевагами такої моделі є здатність вивчати локальні ознаки, інваріантність до зсувів, ефективне використання параметрів та ієрархічне представлення даних.

Графік втрат демонструє, що функція втрат на тренувальній вибірці зменшується протягом усього процесу навчання, що свідчить про те, що модель успішно навчається і зменшує свою помилку на тренувальних даних. Однак, функція втрат на валідаційній вибірці також зменшується протягом перших епох, але потім стабілізується на певному рівні. Це свідчить про те, що модель починає перенавчатися, тобто занадто добре запам'ятовує особливості тренувальних даних і починає гірше узагальнювати на нові, невидимі дані. Графік точності підтверджує висновки, зроблені на основі аналізу графіків втрат. Точність на тренувальній вибірці постійно зростає протягом усього процесу навчання, що вказує на те, що модель все краще класифікує тренувальні дані. Точність на валідаційній вибірці також зростає до певного моменту, а потім починає дещо знижуватися. Що також свідчить про наявність перенавчання.

Отримані результати свідчать про те, що модель демонструє хороші результати на тренувальній вибірці, але має схильність до перенавчання.

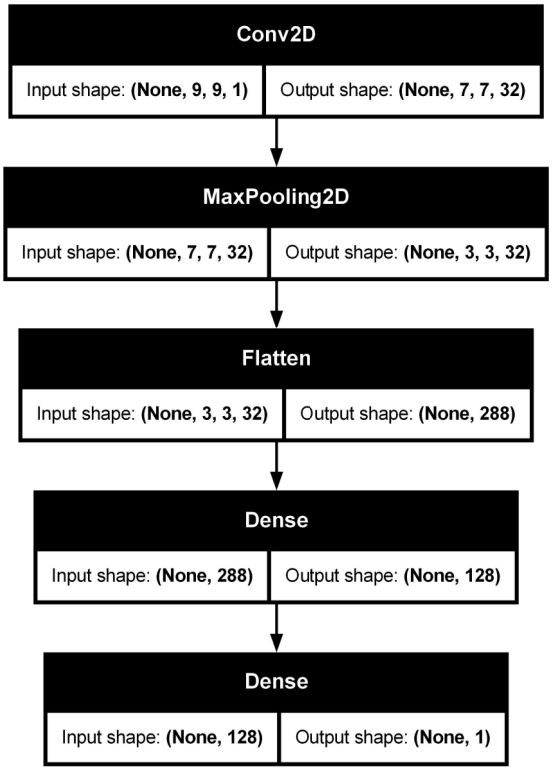

Рисунок 3.14 - Структура шарів зменшеної мінімалістичної моделі CNN

При зменшені розмірності вхідних даних та кількості шарів нейромережі (наведено на рисунку 3.14), що також призводить до зменшення кількості тренувальних параметрів до 37441 помітна тенденція до зниження схильності моделі до перенавчання, а також більш плавної мінімізації функції втрат на початкових етапах навчання(наведено на рисунку 3.15).

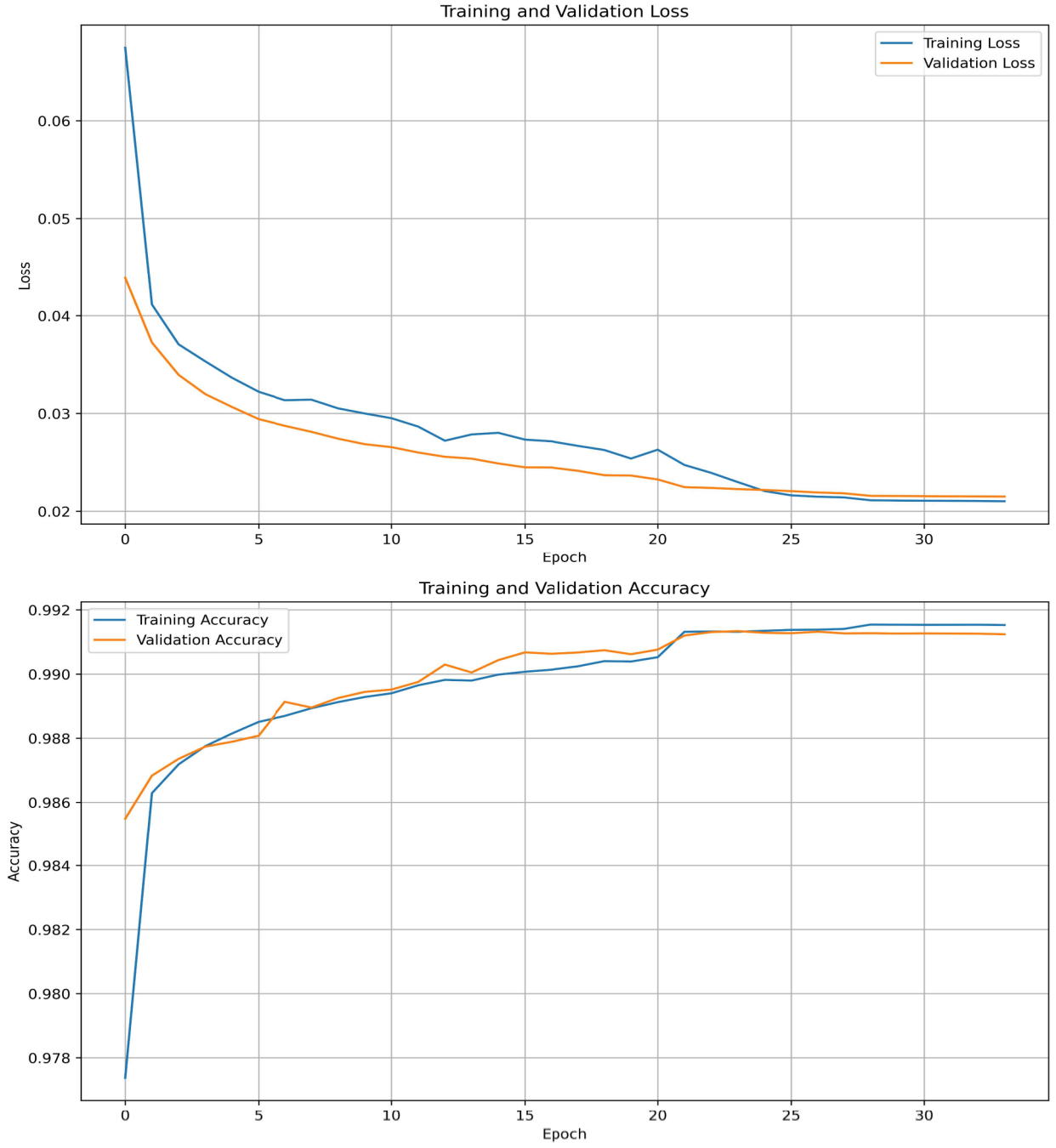

Рисунок 3.15 - Інфографіка навчання зменшеної мінімалістичної моделі CNN

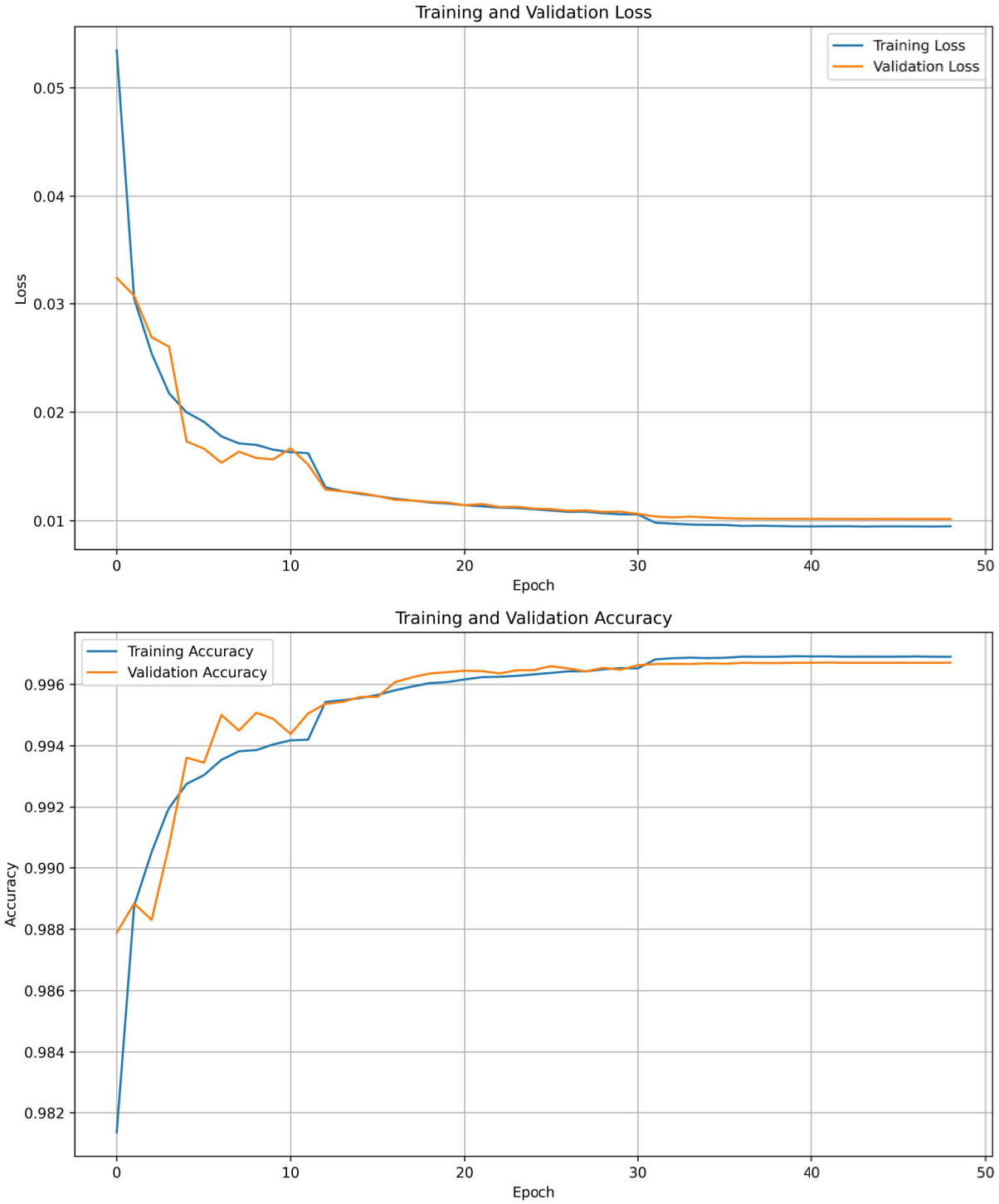

Рисунок 3.16 — Інфографіка навчання великої моделі CNN

Модель №2 — large, велика модель, що має 442031 тренувальних параметрів ставить на меті оцінити потенціал великих нейромереж архітектури (на рисунку Б.2) CNN до навчання на текстових даних для виявлення в них шаблонів та закономірностей за якими можна виявляти аномалії в мережі.

Наведені графіки ілюструють процес навчання на тренувальних даних та узагальнення на валідаційних даних протягом 50 епох. Спостерігається стійке зниження втрат протягом навчання при тренувані, що вказує на те, що модель успішно навчається виявляти закономірності в тренувальних даних.

В ході навчання відзначенно, що спочатку валідаційні втрати знижуються, але після 30ї епохи стабілізуються на низькому рівні. Це може свідчити про те, що модель не перенавчається, тобто не занадто сильно підлаштовується під тренувальні дані, втрачаючи здатність узагальнювати на нові дані. Тренувальна точність швидко зростає і досягає високих значень, що підтверджує ефективність навчання моделі, а валідаційна точність зростаючи і досягаючи високих значень, близьких до тренувальної точності свідчить про хорошу узагальнювальну здатність моделі.

На основі представлених графіків можна зробити такі висновки про успішність навчання моделі. Модель досягає доволі високої точності на тренувальних та валідаційних даних, що свідчить про ефективність. Можна сказати про відсутність суттєвого перенавчання за відсутності значного розриву між тренувальною та валідаційною точністю.

3.5 Аналіз результатів

За для кращого розуміння оцінки ефективності методів, що були обрані предметами дослідження для даної кваліфікаційної роботи результати вимірювань метрик якості класифікації для кожної з моделей винесено в окрему таблицю для підвищення наочності (у таблиці 3.5) та проведення порівняльного аналізу.

Зведена таблиці 3.5 містить метрики критеріїв якості навчання, що були використані в роботі з моделями штучного інтелекту а саме: Loss, Accuracy, Precision, Recall та F1 Score відповідно до кожної з моделей представлених в дослідженні. Відповідно до поставленої мети, «Мінімалістична MLP» модель є найменшою з представлених з переліку та є

компромісом між ефективністю з точки зору точності оцінювання та мінімізації утилізації апаратних потужностей та пам’яті (у таблиці 3.6) цільового приладу.

Таблиця 3.6 — Метрики якості навчання та валідації представлених в дослідженні

Метрика	Мінімалістична MLP	Велика MLP	Мінімалістична CNN	Зменшена мінімалістична CNN	Велика CNN
Loss	0.0490	0.0147	0.0127	0.0134	0.0110
Accuracy	98.52%	99.57%	99.55%	99.57%	99.68%
Precision	96.63%	99.12%	98.64%	99.22%	99.40%
Recall	93.42%	97.99%	98.39%	97.95%	98.45%
F1 Score	95.00%	98.55%	98.52%	98.58%	98.92%

«Велика MLP» має помітну перевагу в точності, проте є в 10 разів більшою та потребує значно більше апаратних ресурсів відповідно(у таблиці 3.6). Також через свою складність, в процесі навчання модель схильна до перенавчання, що може в результаті знижувати ефективність всієї системи, моделі здатна найефективніше визначати лише ті дані на яких вона вчилася.

Моделі на архітектурі CNN демонструють високі результати навчання та ефективність, що підтверджує здатність моделей на архітектурі CNN до обробки та узагальнення статистичних даними замість зображень. Однак при збільшенні моделі помічено схильність моделі до перенавчання, що може бути викликано, як специфікою моделей CNN так і переускладненням моделі.

«Мінімалістична CNN» - є прикладом моделі, що демонструє хороші показники при навчанні та на тестовій вибірці, її архітектура не є занадто складною та має 164,527 параметрів. Дана модель є середнім між «великою»

та «зменшенною» моделями. Розмір моделі сягає 642 кб, що є в 10 разів більше за найкомплекснішу модель MLP представлену в досліджені.

«Зменшена мінімалістична CNN» - має на меті покращення показників «мінімалістичної» моделі за умови досягнення подібних результатів ефективності зі зменшенням складності моделі. За умови збереження ефективності та зменшення вимог до пам’яті цільової системи дана модель може бути корисна в системах з високими вимогами до використання пам’яті та ефективності моделі.

«Велика CNN» - ця модель пристосована для використовується виявлення аномалій у мережевому трафіку, класифікація загроз на основі структурованих даних. CNN ефективно справляється з великими обсягами структурованих даних, але проблема перенавчання може значно знизити її здатність до узагальнення. Перенавчання має серйозні наслідки для аналізу кіберзагроз. Модель може демонструвати високу точність на тренувальних даних, але погано працювати на нових вибірках. Це може призводити до хибнопозитивних результатів, коли невинні події розпізнаються як загрози, або до хибнонегативних, коли реальні загрози залишаються непоміченими.

Для вирішення цієї проблеми можна використовувати кілька підходів. Збільшення обсягу та якості тренувальних даних є одним із ключових рішень. Це може включати аугментацію даних, створення синтетичних вибірок або видалення шуму. Регуляризація також допомагає боротися з перенавчанням: Dropout зменшує коадаптацію нейронів, а L2-регуляризація накладає штраф за великі значення ваг. Рання зупинка тренування може запобігти ситуації, коли модель починає адаптуватися до специфіки тренувальних даних. Розумне проектування архітектури CNN допомагає знизити ризик перенавчання. Зокрема, важливо обмежувати кількість шарів та параметрів, якщо модель є занадто складною для наявних даних.

При роботі з нейронними мережами в незалежність від обраної архітектури завжди гостро постає проблема перенавчання.

Таблиця 3.7 — Характеристики моделей

Модель	Кількість параметрів	Розмір моделі	Затримка обробки 1 цикл
Мінімалістична MLP	185	740.00 Б	0.031 мс
Велика MLP	13,225	51.66 KB	0.035 мс
Мінімалістична CNN	164,527	642.68 KB	0.067 мс
Велика CNN	442,031	1.69 MB	0,083 мс
Зменшена мінімалістична CNN	37,441	146.25 KB	0.037 мс

Дана проблема виникає, коли модель демонструє чудову продуктивність на навчальних даних, але погано узагальнює знання на нові, невідомі дані. Перенавчання може виникати, коли модель надто складна для наявного обсягу даних або коли вона адаптується до шуму та специфічних особливостей навчальної вибірки замість того, щоб виявляти загальні закономірності.

Для багатошарових перцептронів (MLP) та згорткових нейронних мереж (CNN) ця проблема є поширеною через їх високу потужність і здатність моделювати складні залежності. Наприклад, MLP із великою кількістю шарів та вузлів може створити функцію, яка точно відтворює навчальні дані, але має погану продуктивність на тестових даних.

У випадку CNN перенавчання може проявлятися, коли мережа вивчає специфічні деталі даних замість узагальнених ознак. Для запобігання перенавчанню, можна застосовувати до MLP та CNN такі техніки, як регуляризація — є додаванням штрафів до функції втрат для обмеження величин вагових коефіцієнтів. Наприклад, L2-регуляризація змушує модель мати менші ваги, що сприяє узагальненню. У CNN регуляризація також може бути корисною для згорткових фільтрів.

Рання зупинка (early stopping) — передбачає моніторинг продуктивності моделі на валідаційних даних під час навчання. Якщо валідаційна помилка починає зростати, хоча навчальна помилка ще зменшується, процес навчання припиняється, щоб уникнути перенавчання.

Dropout випадково "вимикає" певні вузли або нейрони під час кожної ітерації навчання. Це запобігає залежності моделі від конкретних вузлів і сприяє кращому узагальненню. У CNN dropout зазвичай застосовується до повнозв’язних шарів.

Комбінація методів, таких як регуляризація, dropout, рання зупинка та контроль складності моделі, можу суттєво допомогти в уникнені цієї проблеми як у MLP, так і в CNN, сприяючи створенню моделей, здатних добре працювати на нових даних.

Відповідно до результатів дослідження отримані результати ілюструють, що моделі штучного інтелекту демонструють гарну здатність до узагальнення даних датасету для виявлення загроз в комп’ютерних мережах. В межах дослідження розібрано дві основні архітектури нейромереж, багатошаровий персептрон(MLP) та згорткові нейронні мережі(CNN) – обидві архітектури демонструють гарні результати навчання на датасеті CIC-IDS-2017. Для всіх моделей проведено виміри ефективності на тестових даних, на базі яких припустимо зробити висновок про доцільність використання тої чи іншої архітектури для певних задач.

Важливим фактором, що міг вплинути на результати дослідження є перенавчання(overfitting) – дане явище є основною перешкодою, яка постає перед дослідниками в сфері штучного інтелекту. В незалежності від задачі, даних та комплексності моделі проблема перенавчання може спіткати будь-яку модуль та рішення. Явище виникає, як результат поганого узагальнення моделлю даних та надмірне призвичаєння моделі саме тих даних на основі яких вона навчалася та було встановлено ваги нейронних зв’язків, що негативно впливає на цільову систему до якої у підсумку планується інтегрувати модель.

Узагальнюючи результати можна дійти висновку, що моделі CNN незважаючи на більшу точність виявлення демонструють гірші результати швидкодії на відміну від моделей MLP, навіть найбільша модель демонструє кращу швидкодію за найменшу модель CNN(в таблиці 3.6). Проте фактором при порівнянні зазначених моделей є кількість параметрів для тренування. Оскільки навіть найменша модель CNN з представлених має в 4 рази більше параметрів для навчання можна припустити що за умов більш розгалуженої архітектури самої моделі та застосування більш просунутих методів оптимізації навчання, таких як регуляризація та аугментація даних можливо буде отримати кращі результати навчання та узагальнення моделі, що може скомпенсувати втрати в швидкодії.

За результатами дослідження також визначено методи для потенційного покращення та продовження досліджень. До таких методів можна віднести: використання ансамблів моделі для бінарної класифікації — для навчання кількох моделей з метою визначати лише один клас даних, що може допомогти в вирішені проблеми незбалансованості класів та додати об’єктивності фінальній системі. Навчання ансамблів моделей для багатокласової класифікації — вважається основним вектором подальшого розвитку, адже це дозволить значно збільшити точність класифікації при визначені загроз та дозволить надавати більш точну відповідь за результатами обробки трафіку. До потенційних кандидатів для продовження дослідження віднесено штучні імунні системи(ШІС) та їх ефективність в виявленні ознак загроз в комп’ютерних мережах[10]. Зважаючи на здатність ШІС до виявлення заздалегідь відомих послідовностей та навчанні в процесі роботи, що полягає в виведені нових колоній антитіл відповідно до ростючої бази відомих загроз. Вагомим недоліком цього методу є потенційна низька ефективність ШІС до узагальнення ознак для виявленні нових, досі невідомих, загроз, що не є рідкістю в таких швидко розвиваючихся сферах, як кібербезпека.

ВИСНОВКИ

Виявлення загроз в сучасних комп’ютерних мережах є комплексною задачею, яка в сучасному світі, що швидко розвивається залишає багато простору для подальших досліджень та не може бути охоплена в повному обсязі в межах кількох робіт.

Дослідження проведене в межах даної роботи та написаних за темою публікацій можна вважати початком серії досліджень за даною темою, оскільки сфера кібербезпеки є сферою, швидко розвивається та знаходиться в постійній потребі до нових комплексних рішень, які здатні задовільнити потреби та відповідати на виклики, які щодня постають перед спеціалістами, які працюють в сфері кібербезпеки.

Штучний інтелект — є також сферою, що швидко розвивається та знаходить своє застосування в майже кожній сфері людського життя, через свою здатність до узагальнення ознак та виведення результату ретельно завважуючи, ваги ознак за для прийняття рішення, що робить їх подібними до шаблонів людського мислення. Використовуючи моделі штучного інтелекту, за результатами дослідження, можливо використовувати статистичні дані, які містять дані про мережеві сесії можна можливо навчити модель вирішувати задачі бінарної класифікації для того щоб відрізняти нормальний трафік від аномального, що може свідчити про наявність потенційних загроз в мережі та будуючи інтегрованою в такі системи, як Intrusion Detection System (IDS)[16] може виконувати допоміжну функцію при виявленні аномалій в, з першого погляду, нормальному потоку трафіку для спеціалістів з кібербезпеки.

Дослідження демонструє, що нейромережі здатні навчатися на структурованих та розмічених даних, проте неможливо не зважати на той факт, що результати нейромереж є важкими для аналізу та не завжди є очевидним чому нейромережа прийняла те чи інше рішення, тому завжди варто приймати до уваги, що нейромережі на разі не здатні замінити людину,

оскільки середовище є мінливим та постійно з’являються нові додатки, протоколи та навіть найпросунутіша нейромережа не може на 100% надійно оцінювати поведінку комп’ютерної мережі. Спеціалісти, їх досвід, експертний аналіз, обізнаність в сучасних технологіях та тенденціях того, що які зміни відбуваються в сфері все ще відіграє важливу роль в аналізі кризових ситуацій та прийнятті рішень стосовно інцидентів, що постійно відбуваються в мережах.

Дана робота може бути корисною в перспективі для вирішення проблем виявлення загроз в комп’ютерних мережах, оскільки людський фактор все ще залишається важливим аспектом при аналізі інцидентів — нейромережі можуть відігравати важливу роль в трансформуванні роботи спеціалістів з кібербезпеки, допомагаючи аналізувати мережевий трафік.

Враховуючи рівень розвитку інтелектуальних технологій вважаться доречним вважати, що нейромережі можуть відіграти важливу роль у підтримці та посилені безпекових мір з метою постійного моніторингу комп’ютерних мереж на предмет загроз та потенційних інцидентів, проте людський експертний аналіз залишається вирішальним для прийняття рішень та надання оцінки потенційним інцидентам, що виявленні в комп’ютерних мережах.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. Cloudflare Open System Interconnected [Електронний ресурс] – режим доступу: https://www.cloudflare.com/learning/ddos/glossary/open-systems-interconnection-model-osi/ (дата звернення: 05.01.2025).

2. Goodfellow, I., Bengio, Y., Courville, A. "Deep Learning". MIT Press, 2015. 450с.

3. Bishop, C.M. "Pattern Recognition and Machine Learning". Springer, 2007. 341с.

4. LeCun, Y., Bengio, Y., & Hinton, G. "Deep learning" (2015). Nature, 521(7553), 436-444.

5. Schmidhuber, J. "Deep learning in neural networks: An overview" (2015). Neural Networks, 61, 85-117.

6. Canadian Institute for Cybersecurity CICIDS2017 dataset [Електронний ресурс] – режим доступу: https://www.unb.ca/cic/datasets/ids-2017.html (дата звернення: 04.01.2025).

7. De Castro, L.N. "Artificial Immune Systems: A New Computational Intelligence Approach"

8. Dasgupta, D. (ed.) "Artificial Immune Systems and Their Applications" Timmis, J., Neal, M., & Hunt, J. "An overview of artificial immune systems" (2000). In Computation in Cells and Tissues, 1-10.

9. Dasgupta, D., & Nino, L.F. "Immunological computation: theory and applications" (2008). IEEE Transactions on Evolutionary Computation, 12(3), 243-257.

10. Торубара О. А., Сердюк Н. М. Model of Threat Analysis and Detection System in Computer Networks Based on the Danger Theory of Artificial Immune Systems //Матеріали 12-ї Міжнародної науково-технічної конференції “Інформаційні системи та технології”. 28 листопада 2023, м. Харків, Україна.

11. Торубара О. А., Руденко О. Г. Model of Threat Analysis and Detection System in Computer Networks Based using Artificial Intelligence approach //Матеріали 13-ї Міжнародної науково-технічної конференції “Інформаційні системи та технології”. 28 листопада 2024, м. Харків, Україна.

12. Bengio Y., Courville A., Goodfellow I. Deep learning. MIT Press, 2016. 800 с.

13. Keras Sequential API documentation [Електронний ресурс] – режим доступу: https://keras.io/api/models/sequential/ (дата звернення: 04.01.2025).

14. A. Géron, Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow, 2nd Edition, 2020. URL: https://doi.org/10.1007/s13246-020-00913-z. doi: 10.1007/s13246-020-00913-z.

15. U. Machelucci, Advanced Applied Deep Learning, 2022. URL: https://doi.org/10.1007/978-1-4842-8020-1. Doi: 10.1007/978-1-4842-8020-1.

16. J. Dykstra, Cybersecurity Science: Build, Test, and Evaluate Secure Systems 1st Edition, 2015. URL: urn:oclc:record:1153017022