МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Сумський державний університет
Факультет електроніки та інформаційних технологій
Кафедра комп'ютерних наук

«До захисту допущено»
В.о. завідувача кафедри
Ігор ШЕЛЕХОВ
(підпис)
19 травня 2023 р.

КВАЛІФІКАЦІЙНА РОБОТА

на здобуття освітнього ступеня магістр

зі спеціальності 122 - Комп'ютерних наук,
освітньо-наукової програми «Інформатика»
на тему: «Інформаційна технологія глибокого машинного навчання системи виявлення кіберзагроз»
здобувачки групи ІН.м-11н Зарудної Катерини Олександрівни

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

Катерина ЗАРУДНА
(підпис)

Керівник,
професор,
доктор технічних наук, професор Анатолій ДОВБИШ
(підпис)

Суми – 2023

Сумський державний університет
Факультет електроніки та інформаційних технологій
Кафедра комп'ютерних наук

«Затверджую»
В.о. завідувача кафедри
_____ Ігор ШЕЛЕХОВ
(підпис)

ЗАВДАННЯ НА КВАЛІФІКАЦІЙНУ РОБОТУ

на здобуття освітнього ступеня магістр
зі спеціальності 122 - Комп'ютерних наук, освітньо-наукової програми «Інформатика»
здобувача групи ІН.м-11н Зарудної Катерини Олександрівни

1. Тема роботи «Інформаційна технологія глибокого машинного навчання системи виявлення кіберзагроз»
затверджую наказом по СумДУ від «08» травня 2023 р. № 0475-VI

2. Термін здачі здобувачем кваліфікаційної роботи до 19 травня 2023 року

3. Вхідні дані до роботи

4. Зміст розрахунково-пояснювальної записки (перелік питань, що їх належить розробити) *1) Аналіз проблеми дослідження. 2) Аналіз методу дослідження. 3) Інформаційна технологія глибокого машинного навчання системи виявлення кіберзагроз. 4) Розробка інформаційного та програмного забезпечення інформаційної технології глибокого машинного навчання системи виявлення кіберзагроз*

5. Перелік графічного матеріалу (з точним зазначенням обов'язкових креслень): *Розробка презентаційних слайдів (актуальність теми, вхідні дані, категорійна функціональна модель, опис алгоритму машинного навчання, результати комп'ютерного моделювання, висновки).*

6. Консультанти до проекту (роботи), із значенням розділів проекту, що стосується їх

Розділ	Консультант	Підпис, дата	
		Завдання видав	Завдання прийняв

7. Дата видачі завдання «____» _____ 20 ___ р.

Завдання прийняв до виконання _____ Керівник _____
(підпис) (підпис)

КАЛЕНДАРНИЙ ПЛАН

№ п/п	Назва етапів кваліфікаційної роботи	Термін виконання	Примітка
1.	*Аналіз проблеми дослідження. Постановка задачі дослідження.*	13.03.23 – 17.03.23	
2.	*Аналіз методу дослідження.*	18.03.23 – 26.03.23	
3.	*Дослідження інформаційної технології глибокого машинного навчання системи виявлення кіберзагроз.*	27.03. 23 – 07.04.23	
4.	*Розробка інформаційного та програмного забезпечення інформаційної технології глибокого машинного навчання системи виявлення кіберзагроз*	08.04.23 – 23.04.23	
5.	*Оформлення пояснювальної записки до дипломної роботи*	24.04.23 – 30.04.23	

Здобувач вищої освіти _____ Керівник _____
(підпис) (підпис)

АНОТАЦІЯ

Записка: 58 стор., 11 рис., 4 табл., 1 додаток, 18 джерел.

Мета роботи — підвищення функціональної ефективності системи виявлення кіберзагроз, із застосуванням технології глибокого машинного навчання системи виявлення кіберзагроз.

Об'єкт дослідження — процес виявлення кіберзагроз.

Предмет дослідження — модель і метод глибокого інформаційно-екстремального машинного навчання системи виявлення атак із послідовною оптимізацією системи контрольних допусків.

Метод дослідження – для досягнення поставленої мети, кваліфікаційна магістерська робота виконувалась у раках інформаційно-екстремальної інтелектуальної технології аналізу даних, яка ґрунтується на максимізації інформаційної спроможності у процесі її машинного навчання.

Результати — розроблено програмний комплекс глибокого інформаційно-екстремального машинного навчання з використанням трьох класів розпізнавання на мові C#. Визначено, що використання алгоритму глибокого інформаційно-екстремального машинного навчання дозволяє сформувати вирішальні правила з точністю більше ніж 83%, що перевищує ефективність базового алгоритму на 5%. Розроблений програмний комплекс глибокого інформаційно-екстремального машинного навчання може бути використаний для ефективного виявлення кіберзагроз та забезпечення безпеки інформаційної інфраструктури.

СИСТЕМА ВИЯВЛЕННЯ КІБЕРЗАГРОЗ, ГЛИБОКЕ МАШИННЕ НАВЧАННЯ, ІНФОРМАЦІЙНО-ЕКСТРЕМАЛЬНА ІНТЕЛЕКТУАЛЬНА ТЕХНОЛОГІЯ, НАВЧАЛЬНА МАТРИЦЯ, ІНФОРМАЦІЙНИЙ КРИТЕРІЙ, КАТЕГОРІЙНА МОДЕЛЬ, ТРАФІК

ЗМІСТ

ВСТУП

Забезпечення безпеки функціонування суб'єктів інформаційних відносин, а також захисту потоків інформації є надзвичайно важливою проблемою. Використання інформаційних технологій та керуючих систем, що зберігають та обробляють інформацію зростають, що зробило цю проблему ще більш актуальною.

Відомі випадки кіберзагроз та порушення безпеки даних завдають серйозної шкоди фізичним особам, підприємствам та урядовим структурам в усьому світі, серед яких атака на компанію Equifax у 2017 році, яка призвела до витоку більше ніж 143 мільйонів особистих даних клієнтів та понесла збитку на суму 87.5 мільйонів доларів США [1], а також кіберзагроза на українську мережу магазинів «Епіцентр К» в січні 2022 року, коли зловмисники зашифрували більше 2 терабайт даних і вимагали від компанії викуп у розмірі 7 мільйонів доларів США за їх повернення. Цей інцидент став однією з найбільш масштабних кіберзагроз в Україні за останні роки та нагадав про важливість забезпечення кібербезпеки в усіх галузях діяльності.

Також жертвами атак розповсюдження шкідливого програмного забезпечення, що призвело до витоку конфіденційної інформації користувачів стали деякі веб-ресурси українських університетів та державних органів. Особливої уваги вимагає захист критичної інфраструктури країни, особливо енергетичні мережі, транспортні системи та системи управління водопостачанням, оскільки кіберзлочини спрямовані на них можуть мати серйозні наслідки для життя і здоров'я громадян.

Інформаційні технології – необхідний атрибути науково-технічного прогресу. Їх розвиток призводить до створення нових кіберзагроз, пов’язаних із вторгненням в комп'ютерні системи, крадіжкою конфіденційної інформації, зламом мережевої безпеки, атаками на веб-додатки та електронні платіжні системи, кібершпигунством, кібертероризмом та іншими. Зберігання та обробка інформації у великих масштабах створює підвищену потребу в

забезпеченні безпеки інформаційних і керуючих систем. Розумні пристрої, хмарні сервіси, електронні платіжні системи та інші інноваційні технології, які ми використовуємо щодня, потребують особливої уваги з точки зору захисту інформації. У зв'язку з цим, розробка ефективних засобів захисту інформації, попередження кіберзагроз та мінімізація можливих втрат даних є важливими завданнями для сучасних технологій.

У роботі досліджується проблема кібербезпеки інформаційно-комунікаційної системи (ІКС) та запропоновано рішення цієї проблеми за допомогою інформаційної технології глибокого машинного навчання системи виявлення кіберзагроз, яка забезпечує постійний моніторинг і захист від потенційних кіберзагроз на ІКС. Дослідження орієнтовані на виявлення вразливостей у системах, аналіз потенційних загроз, та запропоновано ефективні методи захисту ІКС. Застосування інформаційної технології машинного навчання для виявлення кіберзагроз забезпечує високу швидкість реакції на аномальні дії та знижує ризик втрати конфіденційної інформації. В результаті, застосування інформаційної технології глибокого машинного навчання дозволить ефективно забезпечити безпеку ІКС і підвищити рівень кіберзахищеності в цілому.

1 АНАЛІЗ ПРОБЛЕМИ ДОСЛІДЖЕННЯ

1.1 Актуальний стан та перспективи розвитку систем виявлення атак

У теперішній час особлива увага приділяється розробці методів прогнозування поведінки комп'ютерних мереж, що функціонують у різних умовах, зокрема, при аналізі ефективності інформаційної інфраструктури під час спроб здійснення цілеспрямованих атак. Цей процес включає ідентифікацію та аналіз мережевого впливу на основі ідентифікації трафіку, який перебуває в них. Для розпізнавання можливих порушень безпеки використовуються евристичні правила та аналіз сигнатур вже відомих атак. Ці методи дозволяють забезпечити безпеку інформаційної інфраструктури в умовах постійних загроз та викликів.

Серед алгоритмів виявлення аномалій (Intrusion Detection System, IDS) найпоширенішими є локальні системи, які встановлюються на окремому комп’ютері (ПК), та системи моніторингу мережі, які відслідковують пакети, що надходять у мережу через пристрої маршрутизації та аналізують їх на наявність аномальних ознак перед відправкою даних до інших мережевих вузлів.

Виділяють три основних методи IDS:

– Метод виявлення аномалій, також відомий як метод поведінки;

– Сигнатурний метод;

– Комбіновані методи, які поєднують у собі метод виявлення аномалій та сигнатурний метод [2].

Метод сигнатур – це метод виявлення аномалій, що ґрунтується на основі сигнатур, які в основному використовуються в системах виявлення вторгнень, де містяться шаблони раніше виявлених атак, створені на основі мережевих пакетів або заголовків. Велика кількість сигнатур може збільшити витрати на обчислення, тому багато дослідників зосереджують увагу на

зменшенні кількості сигнатур та використанні ефективніших алгоритмів для їх розпізнавання [3].

Сигнатурні методи виявлення вторгнень в IDS базуються на шаблонах (сигнатурах) типових атак, які створюються з вихідних даних, зібраних мережевими та хостовими датчиками IDS.

Метод контекстного пошуку є одним з найбільш популярних серед сигнатурних методів та використовується для детектування визначеної множини символів серед вихідних даних. Цей метод є ефективним у виявленні атак на основі аналізу вхідного мережевого трафіку, оскільки дозволяє точно визначити сигнатурні параметри, що необхідно виявити.

Крім того, існує ще два методи сигнатурного розпізнавання кіберзагроз: метод на основі експертних систем та метод аналізу станів. Однак, велика кількість сигнатур може збільшити витрати на обчислення.

Системи, які базуються на експертних методах, дають можливість описати шаблони атак природними мовами з високим рівнем абстракції. Експертна система, яка складає основу методів цього типу, містить базу даних фактів та правил. Факти – це результати роботи ІС, а правила – це логічні алгоритми прийняття рішень про атаку на основі фактів. Усі правила системи будуть записані у форматі «Якщо... то...». База даних правил має включати характерні ознаки атак, які має виявляти IDS.

Метод аналізу станів або метод контролю частоти подій ґрунтується на створенні сигнатур атак у формі послідовності переходів системи з одного стану подій в інший. Кожен такий перехід визначається, коли в системі настає певна подія, а зведена інформація про всі такі події визначається параметрами сигнатури атаки. Цей метод дозволяє виявляти аномальну поведінку системи, що може бути пов'язана з потенційно небезпечними діями з боку зловмисників [2].

Методи, описані вище, мають ряд переваг. Вони дозволяють ефективно виявляти атаки на ІС, зменшують кількість помилкових спрацьовувань, та

дають можливість оцінити використання конкретного інструмента або методу атаки, а також безпомилково визначити параметри сигнатур. Проте, ці методи мають певні недоліки, такі як потребу в постійному оновленні баз даних сигнатур атак для виявлення нових аномалій, неможливість виявити атаки, шаблони яких ще не описані в експертній системі, або атаки, сигнатурні шаблони яких мають відмінності від тих, що вже існують у системі [3].

Поведінкові методи виявлення атак базуються на моделях нормального функціонування інформаційних систем, а не на моделях самої атаки. Вони використовують методику порівняння поточного стану системи зі зразковим режимом її функціонування. Якщо виявлено розбіжності активності, то вона розглядається як аномальна. Перевагами цих методів є розпізнавання атак без володіння знаннями конкретних сигнатур, підвищена чутливість до зміни станів системи та можливість розпізнавання нових атак без необхідності в модифікації параметрів моделі. Однак, складністю такого підходу є розробка точної моделі-зразка нормального режиму функціонування системи.

Поведінкові методи мають перевагу у тому, що вони дозволяють виявляти атаки без потреби знати конкретні сигнатури, мають високу чутливість до змін станів інформаційної системи та можуть виявляти нові атаки без необхідності модифікувати або оновлювати параметри моделі. Однак, створення точної зразкової моделі «нормального» функціонування ІС досить складно [2].

Методи даного типу мають недоліки, серед яких можна визначити: помилкові спрацювання при непередбачуваній поведінці користувачів та мережевих взаємодій, а також значні витрати часу на етапі навчання системи [3].

Методи, що ґрунтуються на статистичних моделях, є найпоширенішими серед поведінкових методів. Ці моделі встановлюють параметри, що описують «нормальну» поведінку інформаційної системи. Якщо під час виконання було виявлено деяке відхилення від встановлених параметрів зразкової моделі, то

фіксується факт кібератаки. Такі параметри можуть включати рівень навантаження мережі, а також міра навантаження на процесор, загальний час роботи користувачів системи та кількість звернень до зовнішніх та внутрішніх ресурсів комп’ютерної мережі.

Під час вторгнення можна виявити атаку за допомогою обох методів, оскільки відмінні від нормальних дії мають характерні ознаки, які ідентифікуються як сигнатури або відхилення від «нормальної» поведінки ІС. Оптимальним варіантом є поєднання цих методів для отримання найбільш ефективних результатів [2].

Із розвитком інформаційних технологій з’являються нові методи виявлення атак, таких як сіткова аналітика та машинне навчання.

Сіткова аналітика – це метод виявлення атак, який базується на аналізі мережі зв'язків між об'єктами та елементами в ІС. Цей метод полягає в зборі та аналізі великої кількості даних про взаємодії між об'єктами в мережі з метою виявлення незвичайних або підозрілих змін у мережевому трафіку.

Основна ідея сіткової аналітики полягає в тому, що атаки на ІС зазвичай пов'язані зі зміною звичайних мережевих патернів, наприклад, збільшенням обсягу передачі даних, зміною мережевої топології або збільшенням кількості запитів до серверів. Шляхом аналізу великої кількості даних про взаємодії між об'єктами в мережі, сіткова аналітика може виявити зміни, які можуть свідчити про потенційну атаку на ІС.

Сіткова аналітика може бути виконана за допомогою спеціального програмного забезпечення, яке збирає та аналізує дані про взаємодії між об'єктами в мережі. В результаті аналізу програмне забезпечення може видавати сповіщення про можливу атаку на ІС або про незвичайні зміни в мережевому трафіку [4].

Машинне навчання – це метод виявлення атак, що базується на аналізі великої кількості даних, з метою розпізнавання відхилень від «нормальної» поведінки системи. Цей метод полягає у навчанні комп'ютерних систем на

прикладах, щоб вони могли визначати, що є «нормальною» поведінкою системи, а що є відхиленням від цієї поведінки.

Для того, щоб машинне навчання було ефективним, необхідно мати достатню кількість даних, які можуть бути використані для навчання алгоритмів. Ці дані можуть бути отримані з журналів системи, даних про використання мережі, а також з додаткових джерел інформації. Після того, як алгоритм буде навчений на даних, він може виявляти аномальні дії або поведінку, які можуть бути пов'язані зі зловмисними діями.

Машинне навчання може використовуватися для виявлення різних типів атак, таких як атаки з використанням вразливостей, атаки на доступ до мережі, атаки на веб-додатки та інші. Однак, необхідно враховувати, що цей метод може бути недостатнім для виявлення складних атак, які можуть бути здійснені з високим рівнем кваліфікації [5].

Ці системи зазвичай використовуються для виявлення відомих шаблонів атак та вразливостей у системах.

Тенденції розвитку систем виявлення атак полягають у розробці та застосуванні інноваційних технологій, таких як глибинне навчання, аналіз поведінки користувачів та машинного навчання на основі нейронних мереж.

1.2 Методи формування і аналізу мережевих і хостових трафіків

Методи формування і аналізу мережевих і хостових трафіків використовуються для виявлення атак на мережеві та інформаційні системи. Один із методів – це аналіз пакетів мережевого трафіку, який дає змогу виявити підозрілі пакети та визначити, як саме відбувається атака.

Інші методи можуть включати аналіз журналів подій на хостах, що дає можливість виявити аномальні дії користувачів або програм, які можуть свідчити про вторгнення. Також до методів аналізу мережевих і хостових трафіків можуть входити інструменти для моніторингу мережі, які забезпечують збір і аналіз даних про мережеву активність, що дозволяє

виявляти несанкціоновані дії та атаки на ранніх етапах їх розвитку. Усі ці методи забезпечують можливість збору та аналізу інформації про мережеві та інформаційні системи для виявлення та запобігання атакам на них.

Методи аналізу трафіку та виявлення аномалій, а також їх комбінації, широко застосовуються при створенні програмно-апаратних систем для виявлення вторгнень. Залежно від типу сенсора, його розміщення та методів аналізу підсистеми, існують різні типи систем виявлення вторгнень (IDS). Технології виявлення вторгнень можна класифікувати за рівнями аналізу окремих пакетів даних, що використовують модель OSI для збільшення глибини аналізу, як показано на рис.1.1.

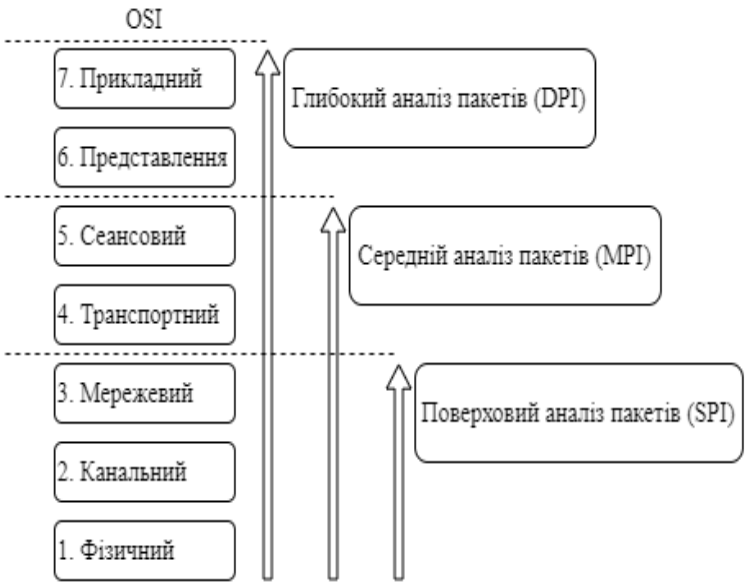


Рисунок 1.1 – Рівні розвитку технології аналізу мережевого трафіку за «глибиною».

Технологія поверхневого аналізу пакетів (Shallow Packet Inspection, SPI) використовується для аналізу трафіку, враховуючи лише заголовки пакетів рівнів 1-3 моделі OSI. Цей метод аналізу вимагає мінімальних обчислювальних можливостей, що дозволяє ефективно обробляти великі обсяги трафіку [6].

Поверхневий аналіз пакетів є методом аналізу мережевого трафіку, що базується на перевірці заголовків пакетів 1-3 рівнів моделі OSI, та може

ефективно розпізнати аномальний трафік, якщо атака має відмінну поведінку від звичайних потоків даних. При цьому, основні характеристики, які відстежуються, включають IP-адреси відправника та отримувача, номери портів та імена протоколів вхідних пакетів даних. Цей метод може бути корисним для виявлення певних типів атак, адже більшість з них використовують певні порти, протоколи та шаблони поведінки, що відрізняються від нормального трафіку.

Цей метод аналізу трафіку в реальному часі, застосовується для ідентифікації програм за їх ім'ям всередині пакетів даних і порівнює їх з номерами, присвоєними Інтернет-організацією з призначення номерів (Internet Assigned Numbers Authority, IANA) для конкретних портів. У разі збігу назв програми та номера порту, трафік вважається безпечним, в іншому випадку – небезпечним і SPI класифікує його як аномалію. Цей метод використовується для аналізу трафіку в режимі реального часу і дозволяє виявляти підозрілі активності, що можуть вказувати на можливість атаки [7].

SPI є широко використовуваною технологією, яка є основою для функціонування більшості міжмережевих екранів, маршрутизаторів та інших мережевих пристроїв. Використання SPI дозволяє розробляти списки контролю доступу на рівні портів та IP-адрес, забезпечуючи ефективну роботу системи з розмежування та контролю доступу для окремих комп'ютерів (IP) та сервісів (портів) в межах внутрішніх мереж. Це дозволяє системі ефективно функціонувати, забезпечуючи безпеку мережі та захист від потенційних загроз [6].

Середній аналіз пакетів (Medium Packet Inspection, MPI) – це метод аналізу трафіку, який спеціалізується на перевірці вмісту сесій зв'язку, які були встановлені за допомогою шлюзу-посередника та ініційовані додатком. Ця технологія зазвичай аналізує вміст пакетів вибірково, використовуючи встановлені правила. Алгоритми MPI можуть розрізняти певні види трафіку, які SPI не може ідентифікувати, такі як деякі типи VPN-з'єднань та

зашифрованого трафіку. Це робить технологію корисною для виявлення комп'ютерних загроз та захисту мережі від атак [6].

Ця технологія моніторингу трафіку, що використовує вузли-посередники, відомі як «середні». Ці вузли розташовані по всій мережі і вони перевіряють пакети, використовуючи спеціальні програми, які працюють на цих пристроях і можуть розпізнавати інформацію в заголовках та корисному навантаженні даних. За допомогою цих вузлів у мережі MPI може бути використана для моніторингу та аналізу вхідних та вихідних пакетів даних. MPI працює як шлюз між кінцевими комп'ютерами та постачальником інтернет-послуг, і вона може перевіряти пакети тільки на рівні транспортного, мережевого, лінійного передачі даних та фізичного рівнів моделі OSI. Застосовуючи технологію середнього пакетного аналізу, адміністратори мережі можуть обмежувати завантаження або отримання шкідливих відео, зображень, mp3 і т.д. через мережу Інтернет [7].

MPI є більш гнучкою технологією в порівнянні з SPI та може виконувати більш широкий спектр завдань, таких як кешування вмісту, аналіз стисненого або зашифрованого трафіку та функціональні обмеження протоколів, що дозволяють забороняти певні команди. Проте головним недоліком MPI є погана масштабованість, оскільки кожен протокол або команда потребує окремого «шлюзу» з вхідними та вихідними портами. Крім того, робота в режимі проксі може значно знизити швидкість обробки пакетів та потужність системи. Однак, MPI залишається ефективним рішенням для моніторингу та аналізу трафіку, що проходить через мережу, та для захисту від шкідливих атак [6].

Технологія глибокого аналізу пакетів (Deep Packet Inspection, DPI) – це інноваційний метод перевірки пакетів у режимі реального часу, який використовується для одночасного моніторингу та аналізу як заголовків, так і корисного навантаження даних. Ця технологія дозволяє детально вивчати кожен пакет, що проходить через мережу, і визначати тип протоколу, вид

даних, що передаються, а також рівень сервісу. DPI зазвичай використовується в мережевих пристроях, таких як маршрутизатори, файрволи та інші, для контролю трафіку та забезпечення безпеки мережі. Одним з переваг DPI є можливість розпізнавати навіть зашифрований або стиснутий трафік, що дозволяє виявляти шкідливі програми та інші загрози безпеці мережі [7].

Іноді можна використовувати більш конкретний термін – Deep Packet Processing (DPP), який описує технологію, що здатна модифікувати, фільтрувати або перенаправляти пакети даних. Сьогодні терміни DPI та DPP часто використовуються як синоніми. Ці технології є результатом розвитку MPI [6].

DPI дозволяє операторам мережі детально вивчати вміст кожного пакету даних, що пройшов через мережеві концентратори. Вона використовується для ідентифікації мережевих програм шляхом перевірки на наявність конкретних підписів протоколів. DPI може виявляти протоколи та програми за допомогою трьох методів: виявлення портів, виявлення підписів та евристичного методу. Ця технологія може виконувати моніторинг всіх рівнів моделі OSI, що робить її перевагою над технологіями MPI та SPI [7].

Технологія DPI є стандартом для забезпечення функціонування засобів аналізу та моніторингу мережевого трафіку в реальному часі. Вона відноситься до критично важливих технологій, які необхідні для забезпечення мережевої безпеки та виконання вимог законодавства. DPI забезпечує аналіз не тільки заголовків пакетів, а й їхнього корисного навантаження, що дозволяє точно визначати тип і зміст передаваної інформації. Дана технологія є найбільш ефективним інструментом для виявлення та реагування на загрози безпеці мережі, такі як віруси, шкідливі програми, атаки хакерів та інші [6].

Крім технологій MPI, DPI та SPI, існує ще кілька методів формування та аналізу мережевих і хостових трафіків:

NetFlow – це протокол, який використовується для збору та аналізу мережевого трафіку. Він дозволяє збирати інформацію про трафік на різних

рівнях, таких як IP-адреса, порти та протоколи. Цей протокол збору трафіку, розроблений компанією Cisco Systems для збору та аналізу інформації про мережевий трафік. Цей протокол збирає дані про трафік, які можуть включати IP-адресу джерела та призначення, порт призначення, використану пропускну здатність та час виконання.

Дані, зібрані за допомогою NetFlow, можуть бути використані для багатьох цілей, таких як моніторинг мережі, виявлення атак на мережу та оптимізація мережевої пропускної здатності. Окрім цього, NetFlow може допомогти зменшити витрати на мережеву інфраструктуру, шляхом оптимізації мережевої топології та управління пропускною здатністю.

Існує кілька версій NetFlow, включаючи NetFlow v5, v9 та IPFIX. NetFlow v5 є найбільш поширеною версією та підтримується більшістю мережевих обладнань Cisco. NetFlow v9 та IPFIX мають більшу функціональність та можуть передавати більше інформації про трафік.

У цілому, NetFlow є потужним інструментом для збору та аналізу мережевого трафіку, який може допомогти виявити та вирішити проблеми в мережі, забезпечити безпеку мережі та покращити продуктивність мережевої інфраструктури.

Flow Analysis – це метод аналізу мережевого трафіку, який зосереджується на розумінні структури мережевих потоків. Він дозволяє виявляти та аналізувати різні типи потоків, такі як відео, голос та дані, та визначати їх характеристики, такі як розмір, швидкість та тривалість. Flow Analysis – це метод аналізу мережевого трафіку, що полягає в групуванні пакетів за спільними характеристиками інформації, такими як адреса відправника та отримувача, порт призначення та джерела та інші параметри. Кожна така група пакетів називається «потоком» (flow).

У процесі Flow Analysis, мережевий трафік проходить через пристрій, який розподіляє пакети на різні потоки відповідно до їх характеристик. Завдяки цьому, користувач може бачити, які пристрої та додатки

використовують мережевий трафік, скільки трафіку вони генерують, які сервіси та порти використовуються та іншу корисну інформацію.

Flow Analysis є корисним інструментом для моніторингу та управління мережею, оскільки дозволяє виявляти проблеми з пропускною здатністю мережі, ідентифікувати трафік, що споживає більше ресурсів мережі, виявляти причини відмов та багатьох інших проблем.

Flow Analysis може використовуватись разом з іншими методами аналізу мережевого трафіку, такими як DPI та Packet Sniffing, для отримання максимально повної картини трафіку в мережі [8].

Protocol Analysis – це метод аналізу мережевого трафіку, який зосереджується на виявленні та аналізі протоколів, що використовуються в мережі. Він дозволяє виявляти проблеми з протоколами, такі як помилки в конфігурації та неправильне використання протоколів.

Behavioral Analysis – це метод аналізу мережевого трафіку, який зосереджується на виявленні змін в поведінці мережевих пристроїв та користувачів. Він дозволяє виявляти незвичну або підозрілу активність в мережі та вчасно реагувати на неї.

Протокольний аналіз (Protocol Analysis) та Поведінковий аналіз (Behavioral Analysis) – це дві різні методики в області моніторингу та аналізу мережевого трафіку.

Протокольний аналіз використовується для аналізу мережевих протоколів, що використовуються в мережі. Цей метод включає в себе розбір мережевих пакетів, що проходять через мережу, та визначення їх вмісту, протоколу та порту призначення. Протокольний аналіз використовується для виявлення аномальних протокольних поведінок та незвичних мережевих з'єднань. Наприклад, такі аномалії можуть включати незвичайні довжини пакетів, несподівані протоколи, які використовуються, або незвичайні порти призначення.

Поведінковий аналіз в основному зосереджений на аналізі звичайних моделей поведінки користувачів та програм в мережі. Цей метод включає в себе моніторинг мережевої активності та виявлення незвичайних дій, які можуть свідчити про наявність загроз у мережі. Поведінковий аналіз може виявляти такі аномалії, як використання незвичайних програм, злам аккаунтів, відправку незвичайних запитів на сервер та інше.

В цілому, протокольний аналіз та поведінковий аналіз – можуть використовуватися для забезпечення мережевої безпеки та виявлення потенційних загроз. Використання обох методів може допомогти виявляти незвичайні або аномальні активності в мережі, що дозволяє операторам мережі оперативно реагувати на можливі загрози та запобігати їх поширенню [9].

1.3 Методи інтелектуального аналізу даних

Методи інтелектуального аналізу даних (ІАД) – це набір технік та методів, які дозволяють збирати, обробляти, інтерпретувати та використовувати дані з метою отримання нових знань та інсайтів. Використовуються вони у багатьох сферах, включаючи науку, бізнес, медицину, громадську безпеку та інші.

ІАД об'єднують в собі алгоритми машинного навчання, статистичний аналіз, класифікацію, кластеризацію, асоціативний аналіз та інші методи. Вони дають змогу виявляти складні зв'язки між даними, прогнозувати тенденції, відшукувати незвичайні шаблони та ризики, а також знаходити закономірності та викривлення.

Методи ІАД допомагають вирішувати багато завдань, наприклад, прогнозування, класифікація, розпізнавання образів, оптимізація процесів та інші. Зараз ІАД стають все популярнішими завдяки росту обсягів даних та збільшенню обчислювальних можливостей комп'ютерів [10].

Методи інтелектуального аналізу даних можна класифікувати такими ознаками, наприклад:

– За типом виконання задачі: рекомендації, класифікація, кластеризація, прогнозування, розпізнавання образів, оптимізація процесів та інші.

– За методом аналізу: алгоритми машинного навчання, статистичний аналіз, кластерний аналіз, асоціативний аналіз, знаходження закономірностей та багато інших.

– За рівнем автоматизації: мінімальний, коли весь процес проводиться вручну, наприклад, вручну створюються правила асоціації, та максимальний, коли аналіз проводиться автоматично з використанням алгоритмів машинного навчання та штучного інтелекту.

– За типом даних: структуровані, наприклад, бази даних, та неструктуровані, наприклад, текстові дані, зображення, аудіо- та відеодані.

Серед методів ІАД можна визначити:

– Кластерний аналіз;

– Класифікація;

– Регресійний аналіз;

– Машинне навчання;

Кластерний аналіз – це статистичний метод машинного навчання, який використовується для групування об'єктів (наприклад, даних, популяцій, пацієнтів тощо) у кластери, що мають схожі характеристики. Кластерний аналіз можна використовувати для виявлення підгруп або шаблонів, які можуть допомогти у пізнанні даних і прийнятті рішень.

Процес кластеризації включає в себе наступні етапи:

– Вибір даних: на цьому етапі визначається, які дані будуть використані для кластеризації. Це можуть бути числові дані, категоріальні дані або комбінація різних типів даних.

– Вибір міри схожості: на цьому етапі визначається, як будуть порівнюватися дані між собою. Це може бути евклідова відстань, косинусна схожість, кореляційний коефіцієнт та інші.

– Вибір методу кластеризації: на цьому етапі визначається, як будуть групуватися дані.

– Визначення кількості кластерів: на цьому етапі визначається, скільки кластерів потрібно створити. Це може бути визначено експертно, за допомогою статистичних методів або за допомогою автоматичного вибору кількості кластерів.

– Кластеризація: на цьому етапі проводиться процес кластеризації, тобто групування даних в кластери згідно вибраних методів і параметрів.

– Оцінка результатів: на останньому етапі проводиться оцінка результатів кластеризації. Що може включати в себе визначення метрик для оцінки якості кластеризації, таких як коефіцієнт силуету, або візуалізацію кластерів для аналізу результатів [11].

Класифікація є одним з найбільш поширених методів інформаційного аналізу даних. Цей метод використовується для призначення категорій або міток для нових даних на основі попередньо класифікованих прикладів. Основна ідея полягає в тому, щоб знайти шаблони або закономірності у вхідних даних і використати ці закономірності для призначення категорій нових даних.

Процес класифікації можна розбити на наступні етапи:

– Збір та підготовка даних.

– Вибір та підготовка атрибутів (ознак) для моделювання.

– Вибір та підготовка алгоритму класифікації.

– Розділення даних на навчальну та екзаменаційну вибірки.

– Навчання моделі на тренувальних даних.

– Оцінка та налаштування параметрів моделі.

– Перевірка якості моделі на тестовій вибірці.

– Застосування моделі на нових даних для класифікації [11].

Одним з методів ІАД є регресійний аналіз, що в свою чергу є методом статистичного аналізу, який досліджує взаємозв'язок між залежною змінною і

однією або декількома незалежними змінними. Основною метою регресійного аналізу є побудова математичної моделі, яка змогла б допомогти прогнозувати значення залежної змінної на основі відомих значень незалежних змінних.

Основними етапами регресійного аналізу є:

– Збір та підготовка даних: на цьому етапі збираються необхідні дані та виконується їх підготовка до аналізу, включаючи очищення даних, обробку відсутніх значень та інші операції.

– Вибір моделі: вибір моделі регресії залежить від типу даних, їх розподілу та характеру залежності між залежною та незалежними змінними.

– Навчання моделі: на цьому етапі модель навчається на зібраних даних шляхом знаходження оптимальних параметрів.

– Перевірка моделі: на цьому етапі перевіряється якість побудованої моделі за допомогою різних метрик та методів, таких як коефіцієнт детермінації, середньо квадратична помилка тощо.

– Використання моделі: після успішної перевірки моделі вона може бути використана для прогнозування значень залежної змінної на основі значень незалежних змінних [12].

Метод інтелектуального аналізу даних, відомий як машинне навчання, це процес автоматизованого вивчення алгоритмів та статистичних моделей, що здатні прогнозувати майбутні події та приймати рішення на основі вивченого матеріалу. Машинне навчання є підрозділом штучного інтелекту, який дозволяє комп'ютерам вчитися з даних без явного програмування.

Метод машинного навчання можна розбити на три основні категорії:

– Навчання з вчителем (Supervised learning)

– Навчання без вчителя (Unsupervised learning)

– Навчання з підкріпленням (Reinforcement learning)

Основні етапи методу машинного навчання включають:

– Збір і підготовка даних для подальшого використання.

– Підготовка даних: очищення даних, заповнення пропусків, видалення аномалій і т.д.

– Розбиття даних на тренувальну, тестову та, можливо, валідаційну вибірки.

– Вибір необхідної моделі для виконання поставленої задачі.

– Навчання моделі: застосування вибраної моделі до тренувальних даних з метою навчання.

– Оцінка продуктивності моделі за допомогою тестової вибірки.

– Налаштування гіперпараметрів моделі для підвищення її продуктивності.

– Використання моделі на нових даних для отримання результатів.

– Впровадження моделі в експлуатацію [5].

Кожен з розглянутих методів має переваги та недоліки й може бути ефективним за певних умов. Наприклад, класифікація використовується для задач розподілу об'єктів на кілька категорій, регресійний аналіз – для задач прогнозування числових значень. Машинне навчання може ефективне у випадках, коли дані мають складну структуру або залежності між ознаками неочевидні.

2 АНАЛІЗ МЕТОДУ ДОСЛІДЖЕННЯ

2.1 Основні положення інформаційно-екстремальної технології аналізу даних

У рамках інформаційно-екстремальної інтелектуальної технології (ІЕІТ) – технології аналізу та розпізнавання даних, машинне навчання використовується для перетворення нечіткого розподілу простору ознак на чіткий розподіл класів розпізнавання. Це досягається за допомогою підвищення рівня оптимізації параметрів роботи інформаційної системи та пошуку глобального максимуму багатоекстремальної функції статистичного інформаційного критерію в допустимій (робочій) області [13].

Методи інформаційно-екстремального машинного навчання базуються на таких специфічних принципах, що доповнюють принципи системного аналізу. Основні з них:

– Максимізація інформації, яка обґрунтована екстремальністю сенсорного сприйняття образу. Для досягнення цього принципу використовуються додаткові інформаційні обмеження, які забезпечують більшу різноманітність класифікованих об'єктів.

– Принцип дуальності, який полягає в застосуванні простих алгоритмів на етапі апріорного моделювання, а потім уточнення їх за допомогою поглибленого машинного навчання для наближення до безпомилкових вирішальних правил за навчальною вибіркою.

– Принцип апріорної невизначеності гіпотез (принцип Бернуллі-Лапласа), який передбачає, що апріорні гіпотези мають рівні ймовірності. Таким чином, система приймає рішення за найгірших умов у статистичному розумінні.

– Рандомізація вхідної інформації, яка дозволяє досліджувати детерміновано-статистичні властивості процесу. Це дозволяє зменшити вплив випадкових факторів і забезпечити більш точні результати.

– редукції даних – оптимізацію даних, що визначає необхідність покращення словника ознак шляхом виключення з нього неінформативних ознак в інформаційному розумінні;

– зовнішнє доповнення, яке вимагає використання навчальної або екзаменаційної вибірки для оцінки ефективності функціонування машинного навчання [13].

У рамках технології ІЕІ важливі правила для оптимізації параметрів машинного навчання створюються за допомогою методу відкладених рішень О. Г. Івахненка. Цей метод має багатоциклічну ітераційну структуру пошуку максимального граничного значення усередненого інформаційного критерію оптимізації за алфавітом класів розпізнавання, що можна виразити так:

$$g_{\xi}^{*} = arg \max_{G_{\xi}} \{\max_{G_{\xi-1}} \{\dots \{\max_{G_1 \cap G_E} \frac{1}{M} \sum_{m=1}^{M} E_m\} \dots\}\}, \quad (2.1)$$

де E_m – інформаційний КФЕ оптимізації параметрів навчання системи розпізнавати реалізації класу X_m^o;

G_E – допустима область розрахунку функції інформаційного критерію оптимізації параметрів машинного навчання;

G_{ξ} – допустима область значень ξ -ї ознаки розпізнавання;

Але на даний алгоритм машинного навчання (2.1) накладаються певні обмеження:

$$\left(\forall X_m^o \in \widetilde{\Re}^{|M|}\right)[X_m^o \neq \emptyset], \quad (2.2)$$

де $\widetilde{\Re}^{|M|}$ – розбиття простору ознак на класи розпізнавання з потужністю $Card\widetilde{\Re} = M$;

$$\left(\exists X_k^o \in \widetilde{\Re}^{|M|}\right)\left(\exists X_l^o \in \widetilde{\Re}^{|M|}\right)[X_k^o \neq X_l^o \rightarrow X_k^o \cap X_l^o \neq \emptyset]; \quad (2.3)$$

$$\left(\forall X_k^o \in \widetilde{\Re}^{|M|}\right)\left(\forall X_l^o \in \widetilde{\Re}^{|M|}\right)[X_k^o \neq X_l^o \to KerX_k^o \cap KerX_l^o = \emptyset], \quad (2.4)$$

де $KerX_k^o$ – ядро класу розпізнавання X_k^o;

$KerX_l^o$ – ядро класу розпізнавання X_l^o, найближчого сусіда для класу розпізнавання X_k^o;

$$\left(\forall X_k^o \in \widetilde{\Re}^{|M|}\right)\left(\forall X_l^o \in \widetilde{\Re}^{|M|}\right)[X_k^o \neq X_l^o \to (d_k^* < d(x_k \oplus x_l))\& \\ \&(d_l^* < d(x_k \oplus x_l))], \quad (2.5)$$

де d_k^*– оптимальний радіус контейнера класу розпізнавання X_k^o;

$d(x_k \oplus x_l)$ – кодова відстань між усередненим вектором x_k класу розпізнавання X_k^o і відповідним вектором x_l класу розпізнавання X_l^o;

d_l^* – оптимальний радіус контейнера класу розпізнавання X_l^o;

$$\bigcup_{X_m^o \in \widetilde{\Re}} X_m^o \subseteq \Omega_{\text{Б}}; k \neq l; k, l, m = \overline{1, M}, \quad (2.6)$$

де $\Omega_Б$ – бінарний простір Хеммінга.

Глибина інформаційно-екстремального машинного навчання визначається кількістю параметрів, які були оптимізовані за інформаційним критерієм, тоді як внутрішній цикл процедури (2.1) є базовим алгоритмом IEIT, який забезпечує покращення геометричних параметрів контейнерів класів розпізнавання [13].

Основною метою методу IEI-технології машинного навчання є досягнення максимальної точності класифікаційних рішень системи розпізнавання. Порівняно з нейронними мережами, методи інформаційно-екстремального машинного навчання використовують функціональний підхід до моделювання когнітивних процесів, що відбуваються в мозку людини під

час прийняття рішень. Це дозволяє безпосередньо моделювати природний інтелект. Оптимізація параметрів системи розпізнавання, що впливають на її ефективність, розглядається як машинне навчання. Ці параметри називаються параметрами машинного навчання.

Методи ІЕІ-технології дають можливість використовувати будь-яку статистичну інформаційну міру різноманітності як критерій оптимізації аналізованих об'єктів. Визначення глибини інформаційно-екстремального машинного навчання відбувається згідно з принципами відкладених рішень О. Г. Івахненка, які забезпечують досягнення граничного максимального значення усередненого за алфавітом класів розпізнавання інформаційного критерію оптимізації, тоді як саму інформаційну міру визначають на основі оптимізованих параметрів машинного навчання. Такий підхід відрізняється від інших методів навчання, оскільки він базується на розробці моделей, які безпосередньо моделюють когнітивні процеси, що відбуваються в людини під час формування та прийняття класифікаційних рішень.

Під час машинного навчання формування вирішальних правил здійснюються на основі оптимальних геометричних параметрів контейнерів класів розпізнавання, що відтворюються в радіальному базисі бінарного простору ознак Хеммінга. Застосування геометричного підходу до побудови вирішальних правил забезпечує їх майже повну інваріантність до багатовимірного простору ознак розпізнавання, адже сучасні комп'ютерні системи можуть обробляти двійкові вектори, які містять навіть 2^{85} ознак розпізнавання. Окрім, ці вирішальні правила є дуже швидкими в прийнятті класифікаційних рішень під час моніторингу СВА, що є важливим чинником виявлення атак [13].

2.2 Категорійна функціональна модель глибокого інформаційно-екстремального машинного навчання

Застосовуючи IEI-технологію, можна досягти максимальної інформаційної спроможності системного вхідного математичного опису шляхом оптимізації параметрів машинного навчання за інформаційною мірою. Однак, базовий алгоритм машинного навчання не завжди забезпечує високу якість розпізнавання трафіку в режимі моніторингу, оскільки початкові значення контрольних допусків на ознаки розпізнавання можуть бути не оптимальними. Тому, для покращення якості машинного навчання, можна використовувати методи оптимізації значень системи контрольних допусків (СКД), які впливають на геометричні параметри контейнерів класів розпізнавання й точність класифікаційних рішень. Одним з таких методів є оптимізація параметру δ поля контрольних допусків.

На малюнку 2.1 зображена категорійна модель інформаційного навчання системи виявлення аномального мережевого трафіку з оптимізацією СКД на ознаки розпізнавання.

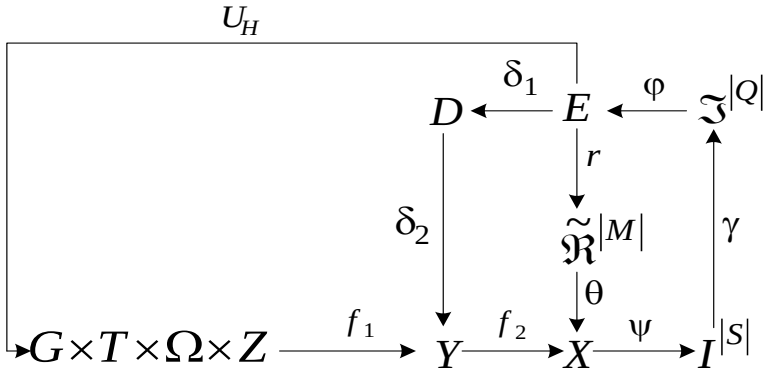


Рисунок 2.1– Категорійна модель машинного навчання з оптимізацією системи контрольних допусків

Рис. 2.1 відображає категорійну модель, яка включає додатковий контур операторів для оптимізації СКД на ознаки розпізнавання. Цей контур закривається терм-множиною D, що складається з допустимих значень для системи контрольних допусків.

Існує три головні підходи до оптимізації СКД на ознаки розпізнавання:

– паралельний алгоритм оптимізації, який виконує оптимізацію СКД для всіх ознак одночасно;

– послідовний алгоритм оптимізації, який оптимізує СКД для кожної ознаки розпізнавання послідовно, з фіксованими значеннями інших ознак;

– алгоритм оптимізації з використанням зведеного поля допусків, який може використовуватися як послідовно-паралельний алгоритм при наявності відмінних одна від одної шкал вимірювання для окремих груп ознак розпізнавання.

Алгоритм паралельної оптимізації СКД має перевагу в швидкості реалізації, але його недолік полягає в тому, що не забезпечує точне значення глобального максимуму інформаційного критерію в робочій області. Натомість, алгоритм послідовної оптимізації СКД забезпечує точні значення глобального максимуму критерія функціональної ефективності у робочій області, але має недолік у повільній швидкості роботи.

Послідовний алгоритм оптимізації системи допусків дозволяє використовувати різні методи оптимізації для кожної окремої ознаки розпізнавання, забезпечуючи більш гнучкий і адаптивний підхід до оптимізації. Він ефективний з точки зору використання ресурсів, оскільки оптимізація проводиться тільки для однієї ознаки розпізнавання за один раз. Алгоритм дозволяє підходити до оптимізації системи контрольних допусків з практичної точки зору, оскільки допуски можуть бути оптимізовані послідовно, виходячи з певного порядку пріоритету ознак, що відображає їх важливість в конкретному використанні.

Для оптимальної оптимізації системи допусків на ознаки розпізнавання доцільно використовувати комбінацію паралельно-послідовного алгоритму, щоб поєднати переваги обох підходів. В цьому випадку застосування паралельного алгоритму дозволить встановити початкові контрольні допуски,

які будуть використовуватись в якості вхідних даних алгоритму послідовної оптимізації [14].

2.3 Інформаційні критерії оптимізації глибокого інформаційно-екстремального машинного навчання

Зазвичай для оцінки ефективності СВА використовують ентропійні інформаційні критерії. Один із найпоширеніших критеріїв цього типу – нормований критерій Шеннона, який має наступний вигляд [15].

$$E = \frac{H_0 - H(\gamma)}{H_0}, \tag{2.7}$$

де H_0– апріорна (безумовна) ентропія:

$$H_0 = -\sum_{l=1}^{M} p(\gamma_l)\, log_2\, p\,(\gamma_l); \tag{2.8}$$

$H(\gamma) -$ Апостеріорна умовна ентропія є мірою невизначеності

$$H(\gamma) = -\sum_{l=1}^{M} p(\gamma_l) \sum_{l=1}^{M} p(\mu_m/\gamma_l)\, log_2\, p\,(\mu_m/\gamma_l), \tag{2.9}$$

де $p(\gamma_l)$– апріорна ймовірність прийняття деякої гіпотези γ_l; $p(\mu_m/\gamma_l)$ – апостеріорна імовірність появи деякої події μ_m за умови прийняття гіпотези γ_l; M – число альтернативних гіпотез.

При оцінюванні функціональної ефективності СВА, яка навчається, можуть бути зроблені деякі припущення на практиці. Зокрема, може бути прийняте двоальтернативне рішення (M=2), а також може бути допущено, що здатна навчатися СК, що слабо формалізована, функціонує за умови невизначеності, тому прийняття рівноймовірних гіпотез $p(\gamma_1) = p(\gamma_2) = 0{,}5$ є виправданим за визначенням Бернуллі-Лапласа. З урахуванням виразів (2.7) і (2.8), критерій може бути виражений у такій частковій формі [15]:

$$E = 1 + \frac{1}{2}\sum_{l=1}^{2} \sum_{m=1}^{2} p(\mu_m/\gamma_l)\ log_2\ p(\mu_m/\gamma_l). \quad (2.10)$$

При двоальтернативному виборі (M=2) за основну гіпотезу береться гіпотеза γ_1 – значення ознаки розпізнавання, яку контролюють, знаходиться в межах допустимих значень δ, а за альтернативну – гіпотезу γ_2. При такому виборі може бути чотири можливих результати вимірювання ознаки, які характеризуються наступними точнісними характеристиками: помилка першого роду – $-\alpha = \rho(x \notin \delta/z \in \delta)$; помилка другого роду – $-\beta = \rho(x \in \delta/z \notin \delta)$; перша достовірність – $D_1 = \rho(x \in \delta/z \in \delta)$ і друга достовірність – $D_2 = \rho(x \notin \delta/z \notin \delta)$, де x, z позначають виміряне та дійсне значення ознаки розпізнавання відповідно .

Можемо розділити множину значень ознак на дві області μ_1 та μ_2: область μ_1 , що містить значення, які належать до допуску , та область μ_2, яка складається зі значень, що не належать до допуску. Тоді можна вивести формулу: $\alpha = p(\gamma_2/\mu_1); \beta = p\ (\gamma_1/\mu_2);\ D_1 = p(\gamma_1/\mu_1); D_2 = p(\gamma_2/\mu_2)$ [15].

Виразимо апостеріорні ймовірності $p(\mu_m/\gamma_l)$ за формулою Байєса:

$$p(\mu_m/\gamma_l) = \frac{p(\mu_m)p(\gamma_l/\mu_m)}{p(\mu_1)p(\gamma_l/\mu_1) + p(\mu_2)p(\gamma_l/\mu_2)}$$

та, прийнявши $p(\mu 1) = p(\mu 2) = 0{,}5$, отримаємо:

$$p(\mu_1/\gamma_1) = \frac{D_1}{D_1+\beta};\ p(\mu_2/\gamma_1) = \frac{\beta}{D_1+\beta};$$

$$p(\mu_1/\gamma_2) = \frac{\alpha}{\alpha+D_2}; p(\mu_2/\gamma_2) = \frac{p_2 D_2}{p_1\alpha+p_2 D_2}. \quad (2.11)$$

Отримавши значення виразу (2.11), ми можемо використати його для розрахунку критерію функціональної ефективності (КФЕ) за Шенноном, підставивши його у вираз (2.10) [15]:

$$E = 1 + \frac{1}{2}\left(\frac{\alpha}{\alpha + D_2}\ log_2\ \frac{\alpha}{\alpha + D_2} + \frac{D_1}{D_1 + \beta}\ log_2\ \frac{D_1}{D_1 + \beta} + \right.$$
$$\left. + \frac{\beta}{D_1 + \beta}\ log_2\ \frac{\beta}{D_1 + \beta} + \frac{D_2}{\alpha + D_2}\ log_2\ \frac{D_2}{\alpha + D_2}\right). \quad (2.12)$$

Статистична інформаційна міра Кульбака [16] може бути використана для оцінки інформативності ознак розпізнавання. Розглянемо формулу для обчислення інформаційної міри Кульбака й встановимо зв'язок із точнісними характеристиками процесу навчання за методами функціонально-статистичних випробувань. Для цього введемо логарифмічне відношення між повною ймовірністю правильного прийняття рішень $P_t^{(k)}$ щодо належності реалізацій класів X_m^o і X_{m+1}^o k-му контейнеру $K_{m,k}^o \in X_m^o$, та повною ймовірністю помилкового прийняття рішень $P_f^{(k)}$, що для двоальтернативної системи оцінок рішень може бути записана таким чином [15]:

$$\Lambda = log_2 \frac{P_t^{(k)}}{P_f^{(k)}} = log_2 \frac{p(\mu_m)p(\gamma_{1,k}/\mu_m) + p(\mu_{m+1})p(\gamma_{2,k}/\mu_{m+1})}{p(\mu_m)p(\gamma_{2,k}/\mu_m) + p(\mu_{m=1})p(\gamma_{1,k}/\mu_{m+1})},$$

де $\gamma_{1,k}$, $\gamma_{2,k}$– гіпотези про належність контейнеру $K_{m,k}^o$ реалізацій відповідно до класів X_m^o і X_{m+1}^o

Припустимо, що $p(\mu_m) = p(\mu_{m+1}) = 0{,}5$, загальна міра Кульбака набуває остаточного вигляду[15]

$$J_m^{(k)} = 0{,}5\ log_2\left(\frac{D_1^{(k)} + D_2^{(k)}}{\alpha^{(k)} + \beta^{(k)}}\right)\left[(D_1^{(k)} + D_2^{(k)}) - (\alpha^{(k)} + \beta^{(k)})\right] =$$
$$= log_2\left(\frac{2 - (\alpha^{(k)} + \beta^{(k)})}{\alpha^{(k)} + \beta^{(k)}}\right)\left[1 - (\alpha^{(k)} + \beta^{(k)})\right], \quad (2.13)$$

3 ІНФОРМАЦІЙНА ТЕХНОЛОГІЯ ГЛИБОКОГО МАШИННОГО НАВЧАННЯ СИСТЕМИ ВИЯВЛЕННЯ КІБЕРЗАГРОЗ

3.1 Формування вхідної матриці системи виявлення кіберзагроз

Основною метою створення вхідної матриці для системи розпізнавання є створення багатовимірної матриці навчання.

$$||y_{m,i}^{(j)}|m = \overline{1,M}; i = \overline{1,N}, j = \overline{1,n}|| \quad (3.1)$$

Для успішного розв'язання поставленої задачі необхідно виконати такі етапи:

– скласти словник ознак та алфавіту класів розпізнавання;

– визначити необхідний обсяг навчальної вибірки;

– встановити нормовані допуски для ознак розпізнавання.

Отже, формування вхідної матриці системи потребує детального аналізу та вивчення властивостей джерела даних. Вхідний математичний опис можна подати у вигляді теоретико-множинної структури [14].

$$\Delta_{\text{В}} = < G, T, \Omega, Z, Y;\ \text{П}, \Phi >, \quad (3.2)$$

де G – простір вхідних факторів, T – множина проміжків часу, Ω – простір ознак розпізнавання, Z – простір можливих станів системи та Y – множина сигналів на виході системи. Для розв'язання задачі розпізнавання необхідно розглянути ці простори та множини та встановити їх взаємозв'язок. Простір вхідних факторівможе впливати на систему, множина проміжків часу визначає, коли інформація зчитується, а простір ознак розпізнавання визначає, які ознаки використовуються для розпізнавання. У цілому, для успішної реалізації системи розпізнавання необхідно ретельно проаналізувати всі складові цієї структури.

П: $G \times T \times \Omega \to Z$ – оператор переходів, що описує, як система змінює свій стан залежно від різних зовнішніх та внутрішніх факторів;

Ф: $G \times T \times \Omega \times Z \to Y$ – оператор, що формує вхідну вибіркову множину Y для системи.

Таким чином, множина випробувань W може бути представлена як декартів добуток множин $G \times T \times \Omega \times Z$, які відповідають вхідним факторам, проміжкам часу, ознакам розпізнавання та можливим станам системи. Словник ознак розпізнавання $\Sigma^{|N|}$, де $N = Card\Sigma^{|N|}$, складається з первинних ознак, що є безпосередньою характеристикою досліджуваного процесу, та вторинних ознак, які є похідними від первинних. Структурованість словника ознак є необхідною умовою. У практиці використання отримані значення параметрів, які зчитуються з датчиків інформації й можуть використовуватися як первинні ознаки, або експериментальні дані, отримані під час дослідження процесу з урахуванням умов його реалізації. Найбільш поширеними вторинними ознаками є різні статистичні параметри векторів ознак класів $\{x_{m,i}^{(j)} | i = \overline{1, N}\}$, навчальних вибірок $\{x_{m,i}^{(j)} | j = \overline{1, n}\}$ чи загальної навчальної матриці.

Якщо розробник інформаційного забезпечення чи інформаційна система мають можливість працювати в режимі кластерного аналізу, то вони можуть створити алфавіт класів розпізнавання $\{X_m^o\}$. Зауважимо, що збільшення потужності алфавіту при сталому словнику ознак може значно вплинути на точнісні характеристики системного навчання, оскільки збільшується ступінь перетину класів розпізнавання. Для визначеного алфавіту, одним із простих критеріїв перетину класів є відношення помилки другого роду до першої достовірності, яка обчислюється на кожному кроці ітерацій навчання системи: $\eta = \frac{\beta^{(k)}}{D_1^{(k)}}$.

Для поліпшення точнісних характеристик за умови збільшення потужності алфавіту класів, ефективним рішенням є створення ієрархічних

алгоритмів навчання системи, що дозволяють розбити класи на менші групи та проводити навчання для кожної із цих груп, а також створення штучної надлишковості словника ознак [14].

3.2 Алгоритм глибокого інформаційно-екстремального машинного навчання

Одним з ключових елементів розробки інформаційного забезпечення ІС, що навчається, є оптимізація системи контрольних допусків (СКД). Це пов'язано з тим, що контрольні допуски безпосередньо мають вплив на геометричні параметри контейнерів класів розпізнавання й, відповідно, на асимптотичні точнісні характеристики рішень. У нашій роботі ми детально проаналізуємо послідовний алгоритм оптимізації СКД та розглянемо його важливість у контексті використання методів ІЕІ технології. Симетричне (двобічне) поле контрольних допусків на значення i-ї ознаки $y_{m,i}^{(j)}, i = \overline{1,N}$, де $m = \overline{1,M}, i = \overline{1,N}, j = \overline{1,n}, M, N, n$ – кількість класів розпізнавання, ознак розпізнавання та реалізацій образу відповідно, зображено на рисунку 2.2.

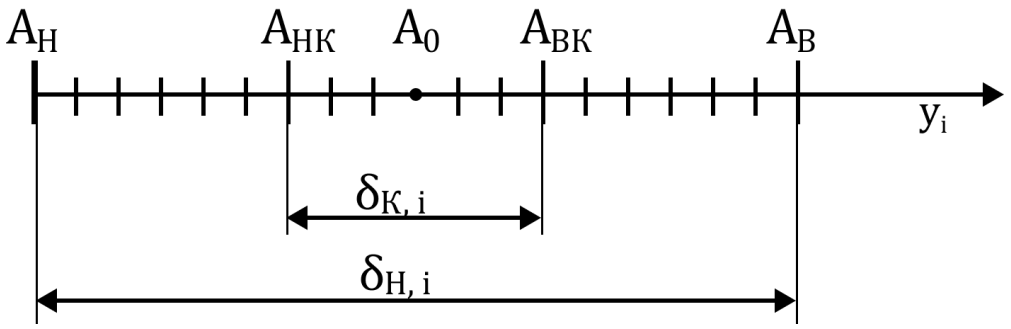


Рисунок 3.1– Симетричне поле допусків

На зображенні 3.1 використано такі умовні позначення: $y_{1,i}$ представляє собою середнє значення i-ї ознаки у векторі-реалізації класу X_1^0; $A_{\text{Н}_i}, A_{B_i}$ відповідають нижньому та верхньому нормованим допускам відповідно; $A_{\text{НК}}$ та $A_{\text{ВК}}$ позначають нижній та верхній контрольні допуски відповідно; $\delta_{\text{Н},i}$ відображає нормоване поле допусків, а $\delta_{K,i}$ – контрольне поле допусків. Параметр δ_i.відповідає за поле контрольних допусків.

Застосуємо паралельно-послідовний алгоритм для оптимізації СКД на ознаки розпізнавання. Після визначення квазіоптимальних контрольних допусків за допомогою паралельного алгоритму, вони використовуються як початкові значення для послідовної оптимізації. Алгоритм паралельної оптимізації контрольних допусків на ознаки розпізнавання працює за ітераційною процедурою пошуку глобального максимуму функції інформаційного коефіцієнта ефективності в межах допустимої області [17]:

$$\left\{\delta_{K,i}^{*} \mid i=\overline{1,N}\right\}=\arg\max_{G_\delta}\max_{G_E\cap G_d}\bar{E}, \tag{3.3}$$

де $\delta_{K,i}$ контрольне поле допусків; області допустимих значень для контрольних допусків G_δ , G_E , G_d, а також радіуси гіперсферичних контейнерів класів розпізнавання, що відновлюються під час навчання в радіальному базисі простору ознак, буде використовуватись алгоритм послідовної оптимізації контрольних допусків на ознаки розпізнавання. Цей алгоритм буде здійснюватись за допомогою ітераційної процедури:

$$\left\{\delta_{K,i}^{*}\right\}=<\arg\left\{\max_{G_{\delta_i}}\left\{\bigotimes_{l=1}^{L}\max_{G_E}\left[\max_{G_d}\bar{E}^{(l)}\right]\right\}\right\}>, i=\overline{1,N}, \tag{3.4}$$

де L – кількість ітерацій процедури.

На етапі паралельної оптимізації були визначені контрольні допуски на ознаки розпізнавання, які можна вважати квазіоптимальними. Їх можна використати як стартові значення для етапу послідовної оптимізації. Для цього застосовується наступна процедура:

1) обнулення лічильника прогонів оптимізації $l:=0$;

2) обчислення значення функції $E_{max,1}^{(l)}$ за базовим алгоритмом навчання для стартової системи допусків;

3) збільшення лічильника прогонів на одиницю l: $l + 1$;

4) обнулення лічильника ознак розпізнавання $i := 0$;

5) збільшення лічильника ознак розпізнавання на одиницю i: $i + 1$;

6) визначення екстремального значення параметра $\delta_{K,i}^{(l)}$ за допомогою внутрішнього циклу оптимізації з використанням базового алгоритму навчання.

7) $\delta_{K,i}^{(l)} := \max_{\delta_{K,i} \in \delta_{\text{Н},i}} \delta_{K,i}^{(l)}$

8) якщо $i \leq N$, то виконати п. 5, інакше – п. 9;

9) якщо $\left| E_{max,1}^{(l-1)} - E_{max,1}^{(l)} \right| < \varepsilon$, де ε – будь-яке мале додатне число, то виконати п. 10, інакше – п. 3;

10) $\{\delta_{K,i}^{*}\} := \max_{\delta_{K,i} \in \delta_{\text{Н},i}} \delta_{K,i}^{(l)}$;

11) ЗУПИН.

Отже, процес оптимізації СКД на ознаки розпізнавання полягає у пошуку глобального максимуму інформаційного коефіцієнта ефективності (КФЕ) з використанням ітераційного підходу, при якому максимальне значення КФЕ наближається до свого максимально можливого значення.

3.3 Короткий опис програмного комплексу

У цій роботі було успішно реалізовано алгоритм глибокого інформаційно-екстремального машинного навчання. Для реалізації даного алгоритму був використаний потужний інструментарій Microsoft .NET Framework, що надає можливості для розробки програмного забезпечення.

Microsoft .NET Framework – це комплексне програмне середовище, що забезпечує зручну модель програмування для створення та запуску різноманітних програмних застосунків. Використовуючи дану платформу, розробники мають можливість використовувати різноманітні мови програмування, такі як C#, VB.NET, F#, та інші. Також .NET надає доступ до

великої кількості бібліотек та інструментів для побудови надійних та масштабованих застосунків різного призначення.

Окрім, .NET забезпечує середовище виконання, яке керує пам'яттю, безпекою та іншими системними ресурсами, забезпечуючи надійну та ефективну платформу для розробки програмного забезпечення. Незалежно від того, чи працюєте ви над розробкою настільного, веб-або мобільного додатку, Microsoft .NET Framework має необхідні інструменти та бібліотеки для досягнення поставленої мети [18].

Для візуалізації результатів обчислень розробленого програмного забезпечення, було використано табличний процесор Microsoft Office Excel та побудовано графіки отриманих результатів.

У таблиці 3.1 наведені основні змінні, які були використані при проектуванні системи виявлення атак глибокого інформаційно-екстремального машинного навчання. Вони були визначені з урахуванням специфіки проєкту.

Таблиця 3.1 – Змінні, що були використані під час проектування СВА

Змінна	Опис
Y	Вхідний інформаційний опис програми
M	Кількість вхідних реалізацій класів розпізнавання
N	Кількість вхідних ознак класів розпізнавання
K	Кількість класів
learn	Екземпляр класу Learning, що реалізує навчання системи
coords	Екземпляр класу Coord, який містить вхідні дані класів розпізнавання
LearningSys	Екземпляр класу LearningSys, що реалізує алгоритм
vDop	Значення параметра верхнього поля СКД на ознаку розпізнавання
nDop	Значення параметра верхнього поля контрольних допусків на ознаку розпізнавання
binMatr	Бінарна матриця
skVal	Масив кодових відстаней від геометричних центрів контейнерів класів до їх реалізацій
skPara	Масив кодових відстаней від геометричних центрів контейнерів класів до найближчих сусідніх реалізацій
etlVec	усереднений еталонний вектор реалізацій
	Найближча сусідня бінарна матриця для класу розпізнавання

d	Міжцентрові відстані від усередненого вектору реалізацій до найближчого сусіднього класу розпізнавання
resKFE	Значення критерію функціональної ефективності за мірою Кульбака.

Функції, які були програмно реалізовані під час розробки системи виявлення атак глибокого інформаційно-екстремального машинного навчання, представлені у таблиці 3.2.

Таблиця 3.2 – Функції системи у режимі глибокого інформаційно-екстремального машинного навчання

Назва	Опис
fileLoader	функція завантаження вхідних даних
makeInput	функція формування вхідних даних у декартовій системі координат
realizProcess	функція запуску процесу навчання
realizAlgor	функція реалізації алгоритму формування еталонного вектору та кодових відстаней
sequentOptim	функція алгоритму послідовної оптимізації СКД на ознаки розпізнавання.
createBinMatr	функція створення робочої бінарної матриці трансформованої з вхідної навчальної матриці
createEtlVektor	функція формування еталонного вектору з робочої бінарної матриці
createSK	функція розрахунку масиву кодових відстаней
KFE	функція розрахунку критерію функціональної ефективності за мірою Кульбака.
avgKFE	функція розрахунку усередненого критерію функціональної ефективності за мірою Кульбака.
localDopusk	функція розрахунку системи контрольних допусків для кожної реалізації
createPara	функція розрахунку масиву кодових відстаней від геометричних центрів контейнерів класів до найближчих сусідніх реалізацій

Програмний код системи наведено у додатку.

3.4 Результати комп'ютерного моделювання

Під час розробки системи виявлення атак було створено математичний опис Y, який дозволяє розпізнати три класи: нормальний трафік X_1^0 та два види аномального трафіку (класи X_2^0 та X_3^0), з базовим класом X_2^0. На першому етапі було застосовано базовий алгоритм з паралельною оптимізацією системи контрольних допусків, щоб порівняти результати алгоритмів. Значення рівня селекції ρ відповідає 0,5 для всіх класів ознак.

Після проведення паралельної оптимізації, було досягнуто максимального значення усередненого критерію функціональної ефективності, який був розрахований за інформаційною мірою Кульбака, рівним 2.22980249814931 на кроці 23. Для подальшого навчання системи, було встановлено оптимальне значення параметру δ, яке дорівнює 23. На рисунку 3.2 зображено розраховану робочу область.

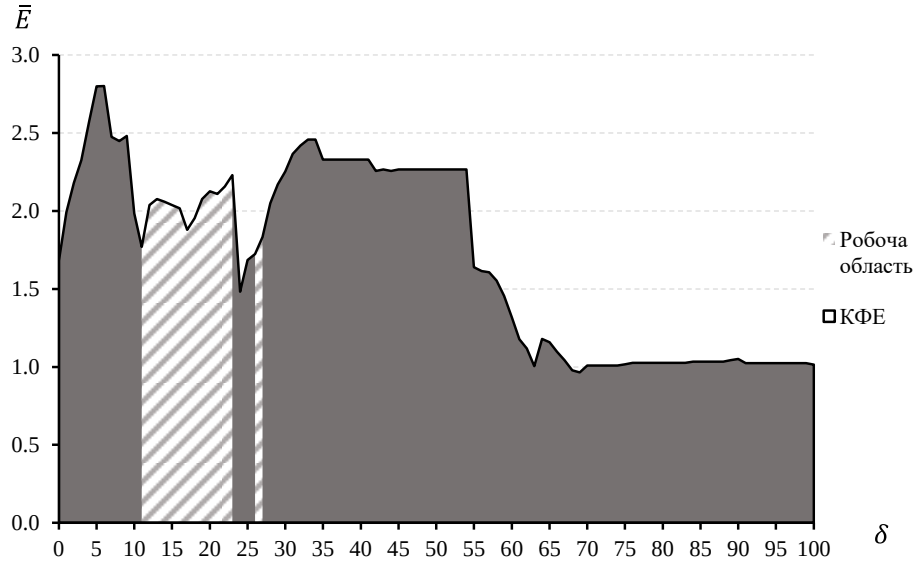
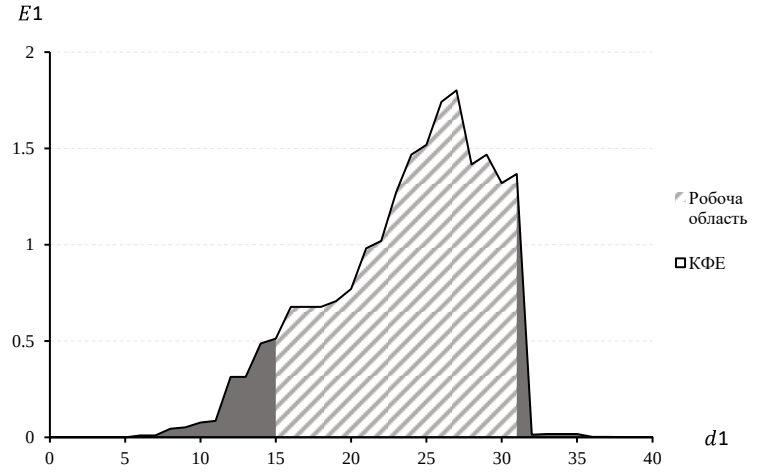
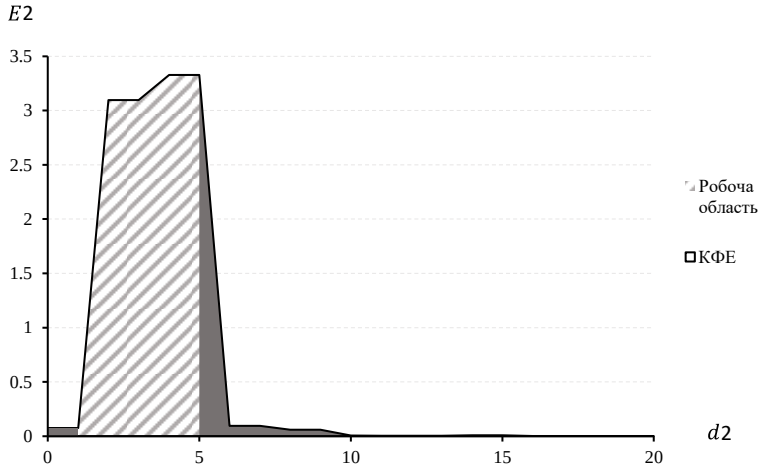


Рисунок 3.2 – Робоча область для базового алгоритму з паралельною оптимізацією системи контрольних допусків

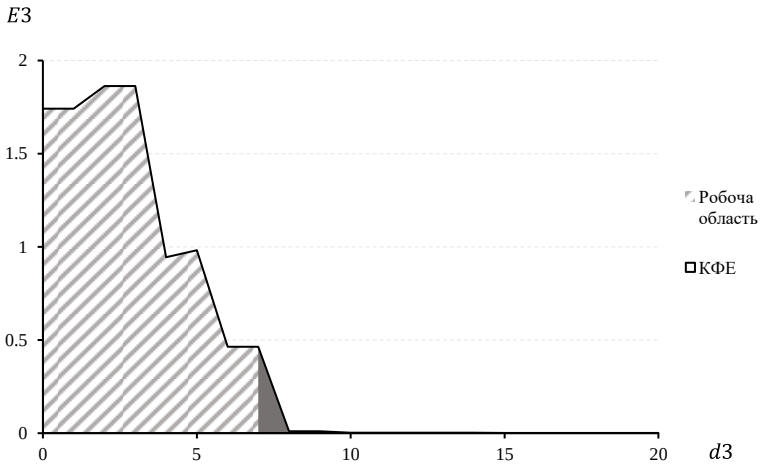
Були визначені оптимальні значення радіусів: $d_1^* = 27, d_2^* = 4, d_3^* = 2$, а максимальні значення КФЕ за мірою Кульбака становили $E_1 = 1.80157, E_2 = 3.32742$ та $E_3 = 1.86329$. Результати моделювання представлені на рисунку 3.3.



а



б



в

Рисунок 3.3 – Графіки залежності КФЕ Кульбака від радіусів центрів розраховані за базовим алгоритмом: а – клас X_1^0, б – клас X_2^0,в – клас X_3^0

За результатами дослідження було встановлено, що при значенні параметру δ=23 та використанні розрахованої на його основі системи контрольних допусків, отримали такі результати: радіуси контейнерів класів розпізнавання за мірою Кульбака становлять $E_1 = 1.80157\, E_2 = 3.32742,\ E_3 = 1.86329.$

Таблиця 3.3 – Результати машинного навчання СВА розраховані за базовим алгоритмом

Клас	Опис класу	Міра Кульбака	Радіус	D_1	D_2
X_1^o	Нормальний трафік	1.80157	27	0.81	0.94
X_2^o	Аномальний трафік	3.32742	4	0.93	1
X_3^o	Аномальний трафік	1.86329	2	0.78	0.98

Після проведення аналізу таблиці 3.3 можна зробити висновок, що застосування алгоритму навчання СВА з паралельною оптимізацією системи контрольних допусків дозволяє сформулювати вирішальні правила з точністю не менше 78% у найгіршому випадку, що є досить високим показником. Таким чином, застосування цього алгоритму може допомогти створити високоточні вирішальні правила для виявлення атак.

Для досягнення кращих результатів виявлення атак було реалізовано алгоритм глибокого інформаційно-екстремального машинного навчання. Цей алгоритм було навчено з використанням вхідних даних, що були використані в базовому алгоритмі, для подальшого порівняння отриманих результатів навчання системи.

Результати побудови системи контрольних допусків під час виконання алгоритму послідовної оптимізації зображено на рис. 3.4 .

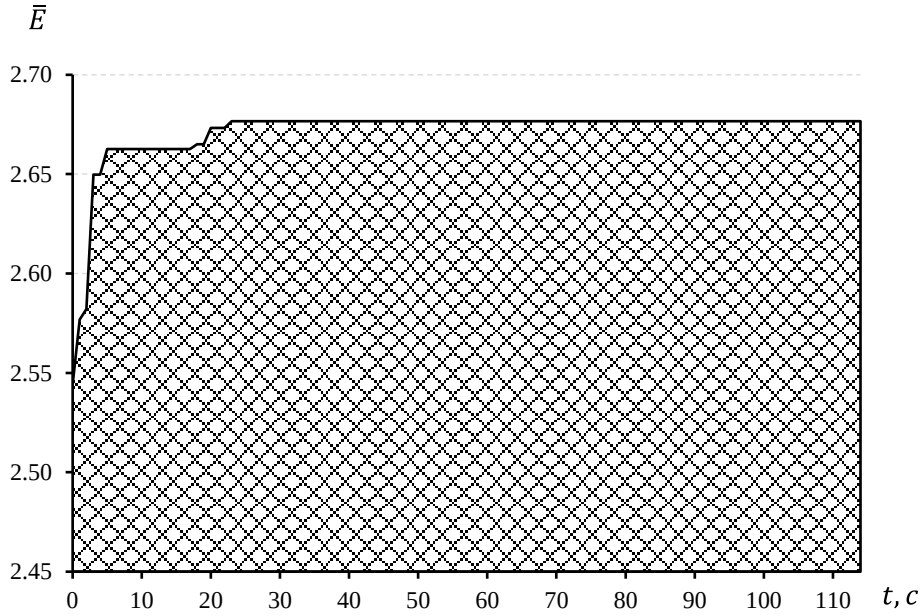
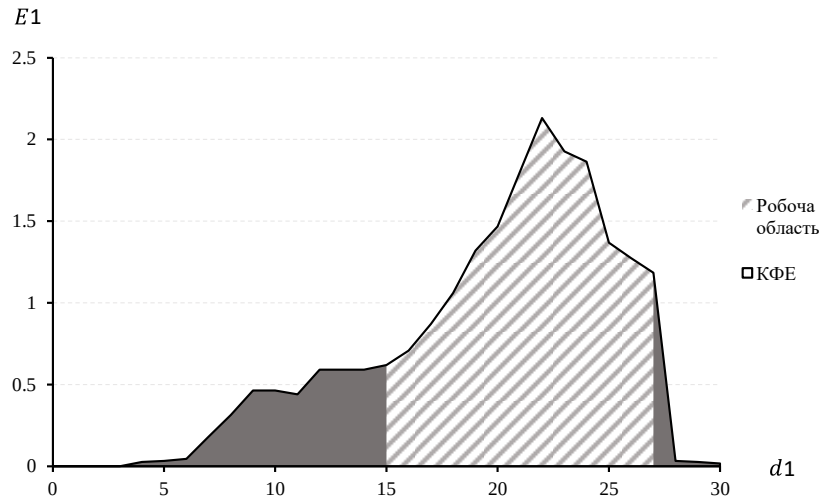


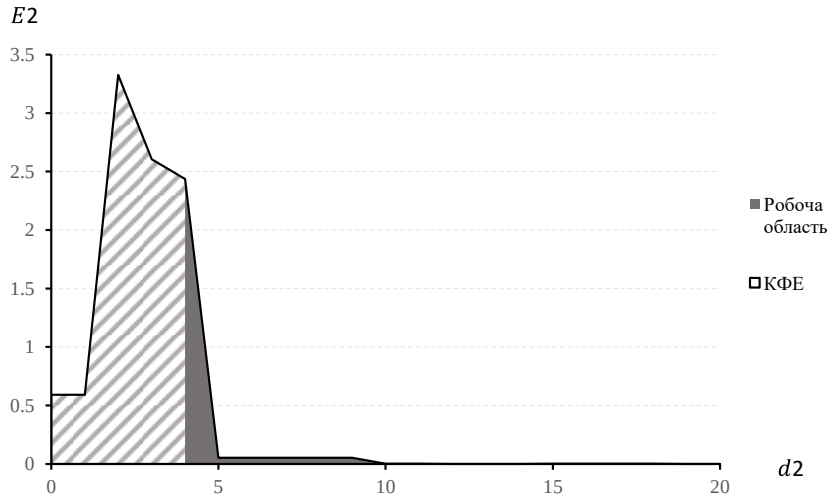
Рисунок 3.3 – Графік зміни усередненого КФЕ Кульбака під час послідовної оптимізації

Після навчання системи виявлення кіберзагроз за алгоритмом глибокого інформаційно-екстремального машинного навчання отримано такі результати:

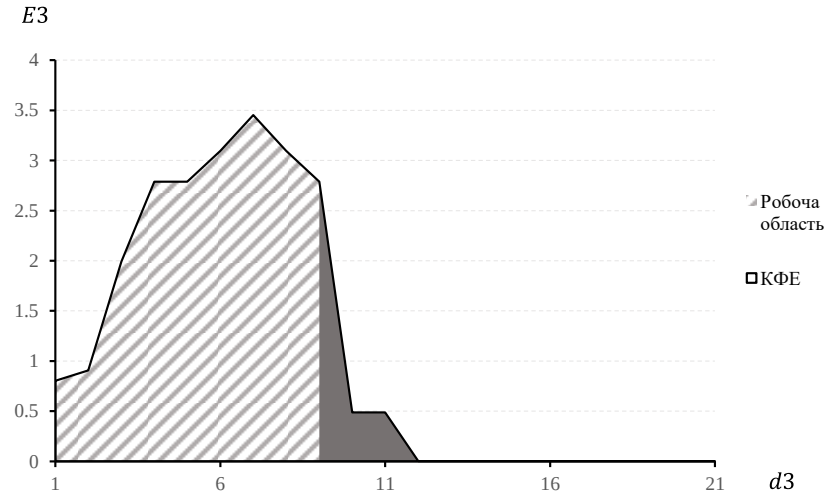
оптимальні значення радіусів: $d_1^* = 22, d_2^* = 2, d_3^* = 6,$ максимальні значення КФЕ за мірою Кульбака становили $E_1 = 2.13037$ $E_2 = 3.32742,$ $E_3 = 3.45208.$ Результати моделювання представлені на рис. 3.5.



а



б



в

Рисунок 3.5 – Графіки залежності КФЕ Кульбака від радіусів центрів розраховані за алгоритмом глибокого інформаційно-екстремального машинного навчання: а – клас X_1^0, б – клас X_2^0,в – клас X_3^0

Результати роботи алгоритму глибокого інформаційно-екстремального машинного навчання наведено в табл. 3.4.

Таблиця 3.4 – результати машинного навчання системи виявлення атак за базовим алгоритмом

Клас	Опис класу	Міра Кульбака	Радіус	D_1	D_2
X_1^0	Нормальний трафік	2.13037	22	0.83	0.97
X_2^0	Аномальний трафік	3.32742	2	0.93	1
X_3^0	Аномальний трафік	3.45208	6	0.94	0.98

Отже за алгоритмом глибокого інформаційно-екстремального машинного навчання визначено, що точність сформованих вирішальних правил становить не менше 83% в найгіршому випадку, що на 5% перевищує точність вирішальних правил, що сформовані за базовим алгоритмом. Таким чином, алгоритмом глибокого інформаційно-екстремального машинного навчання дозволяє сформувати високоточні вирішальні правил з точністю більше ніж 83%, що в свою чергу на 5% перевищує ефективність базового алгоритму.

ВИСНОВКИ

У кваліфікаційній магістерській роботі розглянуто актуальний стан та перспективи розвитку систем виявлення атак. Досліджено такі методи виявлення атак, як метод сигнатур, метод контекстного пошуку, поведінкові методи, сіткова аналітика та машинне навчання. Були досліджені методи формування і аналізу мережевих і хостових трафіків, зокрема SPI, MPI, DPI, NetFlow, Flow Analysis, Protocol Analysis, Behavioral Analysis.

У роботі розглянуто методи інтелектуального аналізу даних, зокрема кластерний аналіз, класифікація, регресійний аналіз та машинне навчання. Досліджено основні положення інформаційно-екстремальної технології аналізу даних та категорійну функціональну модель глибокого інформаційно-екстремального машинного навчання.

Розглянуто інформаційні критерії оптимізації глибокого інформаційно-екстремального машинного навчання.

Визначено, що використання алгоритму глибокого інформаційно-екстремального машинного навчання дозволяє сформувати вирішальні правила з точністю більше ніж 83%, що перевищує ефективність базового алгоритму на 5%.

У роботі досліджено та розроблено програмний комплекс глибокого інформаційно-екстремального машинного навчання з використанням трьох класів розпізнавання на мові C#. Розроблений програмний комплекс глибокого інформаційно-екстремального машинного навчання може бути використаний для ефективного виявлення кіберзагроз та забезпечення безпеки інформаційної інфраструктури. Отже, кваліфікаційна магістерська робота дала важливий внесок у розвиток кіберзахисту та інформаційної безпеки загалом.

СПИСОК ЛІТЕРАТУРИ

1. Новини IT компанії «Аміка» – міжнародного системного інтегратора [Електронний ресурс] / Режим доступу: https://amica.ua/kompaniya-equifax-zaznala-zbytkiv-na-sumu-87-5-mln-v-tretomu-kvartali/.

2. Berkovsky, V. V., & Bessonov, A. S. (2017). Аналіз та класифікація методів виявлення вторгнень в інформаційну систему. Системи управління, навігації та зв'язку. Збірник наукових праць, 3(43), 57-62.

3. Saba T. et al. Anomaly-based intrusion detection system for IoT networks through deep learning model //Computers and Electrical Engineering. – 2022. – Т. 99. – С. 107810.

4. Javaheri, D., Gorgin, S., Lee, J. A., & Masdari, M. (2023). Fuzzy Logic-Based DDoS Attacks and Network Traffic Anomaly Detection Methods: Classification, Overview, and Future Perspectives. Information Sciences.

5. Tsukerman, E. (2019). Machine Learning for Cybersecurity Cookbook: Over 80 recipes on how to implement machine learning algorithms for building security systems using Python. Packt Publishing Ltd.

6. Zola F. et al. Network traffic analysis through node behaviour classification: a graph-based approach with temporal dissection and data-level preprocessing //Computers & Security. – 2022. – Т. 115. – С. 102632.

7. Ghosh, A., & Senthilrajan, A. (2019). Research on packet inspection techniques. International Journal Of Scientific & Technology Research, 8, 2068-2073.

8. Khedker, U., Sanyal, A., & Sathe, B. (2017). Data flow analysis: theory and practice. CRC Press.

9. Wondracek, G., Comparetti, P. M., Kruegel, C., Kirda, E., & Anna, S. S. S. (2008, February). Automatic Network Protocol Analysis. In NDSS (Vol. 8, pp. 1-14).

10. Гороховатський, В. О., & Творошенко, І. С. (2021). Методи інтелектуального аналізу та оброблення даних: навч. посібник.

11. Dalmaijer E. S., Nord C. L., Astle D. E. Statistical power for cluster analysis //BMC bioinformatics. – 2022. – Т. 23. – №. 1. – С. 1-28.

12. Пріщенко, О. П., & Черногор, Т. Т. (2019). Деякі особливості проведення регресійного аналізу (Doctoral dissertation, Національний технічний університет" Харківський політехнічний інститут").

13. Довбиш, А. С., Ободяк, В. К., & Шелехов, І. В. (2021). Сучасні інформаційні технології в кібербезпеці.

14. Довбиш, А. С. (2009). Основи проектування інтелектуальних систем.

15. Довбиш, А. С. (2014) 3694 Методичні вказівки до виконання практичних робіт із дисципліни «Основи проектування інтелектуальних систем» – Суми : СумДУ, 2014. – 56 с.

16. Москаленко, В. В., & Довбиш, А. С. (2016). Вступ до інформаційного аналізу і синтезу інфокомунікаційних систем: навч. посіб. Суми: СумДУ.

17. Довбиш А. С., Боровик В. О., Андрієнко Н. І. Оптимізація параметрів навчання системи підтримки прийняття рішень для керування виробництвом композитних матеріалів // Вісник СумДУ. Серія “Технічні науки”, №3’ 2012 – С. 10 -15.

18. Офіційна документація Microsoft .NET Framework [Електронний ресурс] / Режим доступу: https://dotnet.microsoft.com/en-us/learn/dotnet/what-is-dotnet-framework.

ДОДАТОК

```
using System;
using System.IO;
using Diploma;

class Program
{
  const int M = 100;
  const int N = 115;
  const int K = 3;

  static void Main()
  {
    double[,] a = new double[N, M];
    double[,] b = new double[N, M];
    double[,] c = new double[N, M];

    FileLoader("./1.txt",out a);
    FileLoader("./2.txt",out b);
    FileLoader("./3.txt",out c);

    double[,,] Y = new double[M, N, K];
    double ymin, ymax;

    Learning learn;
    learn = new Learning();
    learn.makeInput(Y);
    learn.realizProcess();

  }

  public static void FileLoader(string filename, out
double[,] result)
  {
    StreamReader file = new StreamReader(filename);
    result = new double[N, M];
```

```
    for (int i = 0; i < N; i++)
    {
      string[] line = file.ReadLine().Split(' ');
      for (int j = 0; j < M; j++)
      {
        result[i, j] = double.Parse(line[j]);
      }
    }
    file.Close();
  }

namespace Diploma;

public class Learning
{
  List<Coord> coords;

  public LearningDecart LearningDecart;

  public void realizProcess()
  {
    LearningDecart = new LearningDecart(coords);
    LearningDecart.sequentOptim();
  }
  public void makeInput(double[,,] Y)
  {
    int m = Y.GetLength(0);
    int n = Y.GetLength(1);
    Coord coord = new Coord();

    for (int i = 0; i < m; i++)
    {
      for (int j = 0; j < n; j++)
      {
        for(int k = 0; k < 3; k++)
        {
          double value = Y[i, j, 0];
          coord.setMatrValue(i, j, value);
```

```
        }
      }
    }
    coords.Add(coord);
  }
}

public class Coord
  {
    double[,] coordMatrix;
    int[,] binMatr;
    int[] etlVec;
    int N;
    int m;
    float roValue = 0.5f;
    float avgValue;


    public int getEtlVec(int i) {
      return etlVec[i];
    }
    public int getbBinMatr(int i, int j) {
      return binMatr[i, j];
    }
    public int getCol() {
      return N;
    }
    public int getRows()
    {
      return m;
    }
    public double getMatrValue(int i, int j) {
      return coordMatrix[i, j];
    }
    public void setMatrValue(int i, int j, double value) {
      coordMatrix[i, j] = value;
    }
    public void createEtlVektor()
```

```
    {
      for (int i = 0; i < N; i++)
      {
        float sum = 0;
        for (int j = 0; j < m; j++)
          sum += binMatr[i, j];
        if ((sum / m) >= roValue) etlVec[i] = 1;
        else etlVec[i] = 0;
      }
    }
    public void createBinMatr(double[] vd, double[] nd)
    {
      for (int i = 0; i < N; i++)
      {
        for (int j = 0; j < m; j++)
        {
          if (coordMatrix[i, j] >= nd[i] && coordMatrix[i,
j] <= vd[i])
            binMatr[i, j] = 1;
          else
            binMatr[i, j] = 0;
        }
      }
    }

  }

public class LearningDecart
{
  List<Coord> coords = new List<Coord>();
  int[] prD;
  int[] pr;
  int[,] skVal;
  int[,] skPara;
  float[] EM;
  int[] d0;

  double[] vDop;
```

```
double[] nDop;

int lim = 80;
int d;
int usedClass = 0;
int[] dSeq;

List<float[]> seqKFE;
float[,,] resKFE;
public LearningDecart(List<Coord> sender)
{
  foreach (var item in sender)
    coords.Add(item);
  skVal = new int[2, coords[0].getRows()];
  skPara = new int[2, coords[0].getRows()];

  prD = new int[coords.Count];
  pr = new int[coords.Count];

  EM = new float[coords.Count];
  d0 = new int[coords.Count];

  vDop = new Double[coords[0].getCol()];
  nDop = new Double[coords[0].getCol()];

  dSeq = new int[coords[0].getCol()];

  resKFE = new float[coords.Count, coords[0].getCol(), 2];
  seqKFE = new List<float[]>();
}

void realizAlgor()
{
  foreach (var decart in coords)
    decart.createBinMatr(vDop, nDop);
  for (int k = 0; k < coords.Count; k++)
    coords[k].createEtlVektor();
  createPara();
```

```
    createDo();
  }
  void createPara()
  {
    for (int k = 0; k < coords.Count; k++)
    {
      prD[k] = 0;
      int evSum1 = coords[k].getCol();
      for (int j = 0; j < coords.Count; j++)
      {
        if (j != k)
        {
          int evSum = 0;
          for (int i = 0; i < coords[k].getCol(); i++)
          {
            if (coords[k].getEtlVec(i) !=
coords[j].getEtlVec(i))
              evSum++;
          }
          if (evSum <= evSum1)
          {
            evSum1 = evSum;
            pr[k] = j;
            prD[k] = evSum1;
          }
        }
      }
    }
  }
  double KFE(int d, ref float d1, ref float betta, int k)
  {
    int p1 = 0, p4 = 0;
    for (int J = 0; J < coords[k].getRows(); J++)
    {
      if (skVal[0, J] <= d) p1++;
      if (skVal[1, J] <= d) p4++;
    }
```

```
    d1 = (float)p1 / (float)coords[k].getRows();
    betta = (float)p4 / (float)coords[k].getRows();

    float d1b = d1 – betta;
    double res = 1 * Math.Log((1 + 1 + 0.001) / (1 – 1 +
0.001));
    return d1b * Math.Log((1 + d1b + 0.001) / (1 – d1b +
0.001)) / res;
  }
  void createSK(int k)
  {
    for (int t = 0; t < coords[k].getRows(); t++)
    {
      skVal[1, t] = 0;
      skVal[0, t] = 0;
      skPara[1, t] = 0;
      skPara[0, t] = 0;
      for (int i = 0; i < coords[k].getCol(); i++)
      {
        if (coords[k].getEtlVec(i) !=
coords[pr[k]].getbBinMatr(i, t)) skVal[1, t] += 1;
        if (coords[k].getEtlVec(i) !=
coords[k].getbBinMatr(i, t)) skVal[0, t] += 1;
        if (coords[pr[k]].getEtlVec(i) !=
coords[pr[k]].getbBinMatr(i, t)) skPara[1, t] += 1;
        if (coords[pr[k]].getEtlVec(i) !=
coords[k].getbBinMatr(i, t)) skPara[0, t] += 1;
      }
    }
  }
  void createDo()
  {
    float td1 = 0, tbetta = 0;
    resKFE = new float[coords.Count, coords[0].getCol(), 2];

    for (int c = coords.Count – 1; c >= 0; c--)
    {
      EM[c] = 0;
```

```
      d0[c] = 0;
      createSK(c);
      float dCorrect = 999;
      float dTmp;
      for (int r = 1; r < coords[c].getCol(); r++)
      {
        double E = KFE(r, ref td1, ref tbetta, c);
        if (E >= EM[c] && td1 >= 0.5 && tbetta <= 0.5 && r <
prD[c])
        {
          if (E > EM[c])
            d0[c] = r;
          EM[c] = (float)E;
          dTmp = (float)r / prD[c];
          if (E == EM[c] && dTmp < dCorrect) {
            dCorrect = dTmp;
            d0[c] = r;
          }
        }
        if (E > EM[c] && d0[c] == 0)
        {
          dCorrect = (float)r / prD[c];
          EM[c] = (float)E;
        }
        if (td1 >= 0.5 && tbetta <= 0.5 && r < prD[c])
          resKFE[c, r, 1] = (float)E;
        resKFE[c, r, 0] = (float)E;
      }
    }
  }
  public float avgKFE()
  {
    float avg = 0;
    for (int c = 0; c < coords.Count; c++)
      avg += EM[c];
    avg /= coords.Count;
    return avg;
  }
```

```
bool localDopusk(int i, int del)
{
  double sum = 0;
  for (int c = 0; c < coords[usedClass].getRows(); c++)
    sum += coords[usedClass].getMatrValue(i, c);

  sum /= coords[usedClass].getRows();

  vDop[i] = sum + del;
  nDop[i] = sum – del;

  if (vDop[i] > 255 || nDop[i] < 0)
    return false;
  return true;
}
public void sequentOptim()
{
  float resG = 0, nowResG = 0;
  for (int c = 0; c < coords[usedClass].getCol(); c++)
    dSeq[c] = d;

  nowResG = avgKFE();

  do
  {
    float[] p = new float[coords[usedClass].getCol()];
    resG = nowResG;

    for (int i = 0; i < coords[usedClass].getCol(); i++)
    {
      float lastResL = 0;
      float rad = 0;

      for (int del = 1; del < lim; del++)
      {
        if (!localDopusk(i, del)) break;
        realizAlgor();
```

```
          float nowResL = avgKFE();
          if (nowResL > lastResL)
          {
            lastResL = nowResL;
            dSeq[i] = del;
            rad = d0[usedClass];
          }
          else if (nowResL == lastResL && rad <
d0[usedClass])
          {
            lastResL = nowResL;
            dSeq[i] = del;
            rad = d0[usedClass];
          }
        }
        localDopusk(i, dSeq[i]);
        realizAlgor();
        p[i] = avgKFE();
      }
      seqKFE.Add(p);
      nowResG = avgKFE();
    } while (Math.Abs(nowResG – resG) > 0.01);
    realizAlgor();
  }
}
```