НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ
імені ІГОРЯ СІКОРСЬКОГО»
НАВЧАЛЬНО-НАУКОВИЙ ФІЗИКО-ТЕХНІЧНИЙ ІНСТИТУТ
Кафедра інформаційної безпеки

До захисту допущено
Завідувач кафедри

_____ **Дмитро ЛАНДЕ**
(підпис)

«_____»_____2025 р.

Дипломна робота

на здобуття ступеня бакалавра

за освітньо-професійною програмою «Системи, технології та математичні методи кібербезпеки»

спеціальності 125 «Кібербезпека»

на тему: **Класифікація протоколів та відновлення мережевої конфігурації ОТ-систем**

Виконав (-ла): здобувач вищої освіти **IV** курсу, групи ФБ-13
(шифр групи)

Стягайло Данило Андрійович
(прізвище, ім'я, по батькові) (підпис)

Керівник Кандидат технічних наук доцент Ільїн Микола Іванович
(посада, науковий ступінь, вчене звання, прізвище, ім'я, по батькові) (підпис)

Рецензент
(посада, науковий ступінь, вчене звання, науковий ступінь, прізвище, ім'я, по батькові) (підпис)

Засвідчую, що у цій дипломній роботі немає запозичень з праць інших авторів без відповідних посилань.
Здобувач вищої освіти _____
(підпис)

Київ – 2025 року

НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ
імені ІГОРЯ СІКОРСЬКОГО»
НАВЧАЛЬНО-НАУКОВИЙ ФІЗИКО-ТЕХНІЧНИЙ ІНСТИТУТ
Кафедра інформаційної безпеки

Рівень вищої освіти – перший (бакалаврський)
Спеціальність – 125 «Кібербезпека»
Освітньо-професійна програма «Системи, технології та математичні методи кібербезпеки»

ЗАТВЕРДЖУЮ
Завідувач кафедри
_____ Дмитро ЛАНДЕ
(підпис)
«___»_____2025 р.

ЗАВДАННЯ
на дипломну роботу здобувачу вищої освіти

Стягайло Данило Андрійович
(прізвище, ім'я, по батькові)

1. Тема роботи
Класифікація протоколів та відновлення мережевої конфігурації ОТ-систем

Переклад теми дипломної роботи (англ.):

Protocols Classification and OT-systems Network Configuration Recovery

Науковий керівник роботи
Кандидат технічних наук доцент Ільїн Микола Іванович,
(прізвище, ім'я, по батькові, науковий ступінь, вчене звання)
затверджені наказом по університету від « 26 » травня 2025 р. №

2. Термін подання роботи здобувачем вищої освіти « 17 » червня 2025 р.

3. Вихідні дані до роботи: Літературні джерела на тему ОТ-мереж, методологічні розробки аналізу ОТ-трафіку, датасети з дампами мережевого трафіку

4. Зміст роботи: дослідження методів відновлення мережевої конфігурації

5. Перелік ілюстративного матеріалу (із зазначенням плакатів, презентацій тощо): презентація до захисту дипломної роботи

6. Дата видачі видачі завдання: 6 вересня 2024р.

Календарний план

№ з/п	Назва етапів виконання дипломної роботи	Термін виконання етапів дипломної роботи	Примітка
1	Отримання завдання	06.09.2024	Виконано
2	Огляд наукової літератури	01.03.2025 – 30.03.2025	Виконано
3	Робота над першим розділом	31.03.2025 – 13.04.2025	Виконано
4	Аналіз існуючих методів вирішення задачі	14.04.2025 – 27.04.2025	Виконано
5	Робота над другим розділом. Проектування алгоритму	28.04.2025 – 11.05.2025	Виконано
6	Проведення експерименту. Валідація методології на реальних данних	12.05.2025 – 18.05.2025	Виконано
7	Робота над третім розділом. Аналіз результатів	19.05.2025 – 01.06.2025	Виконано
8	Створення презентації	12.06.2025 – 15.06.2025	Виконано
9	Передзахист дипломної роботи	17.06.2025	Виконано
10	Доопрацювання	18.06.2025 – 23.06.2025	Виконано
11	Захист дипломної роботи	24.06.2025	

Здобувач вищої освіти _____ Данило СТЯГАЙЛО
(підпис) (Власне ім’я, ПРІЗВИЩЕ)

Керівник роботи _____ Микола ІЛЬЇН
(підпис) (Власне ім’я, ПРІЗВИЩЕ)

РЕФЕРАТ

Обсяг дипломної роботи 59 сторінки. Дипломна робота містить 45 рисунків, 3 таблиці, 4 додатки та 23 літературні джерела.

В роботі розглянуто методи відновлення мережевої конфігурації та атрибутів кожного з об’єктів мережі: IP-адреса, MAC-адреса , відкриті порти, ОС, вендор, роль, мережевий OT-протокол. На основі проведеного аналізу з врахуванням особливостей OT-середовища запропоновано використовувати поведінковий аналіз для знаходження ролі OT-пристрою та ідентифікації використаного OT-протоколу.

Запропонована модель визначає алгоритм відновлення мережевої конфігурації пасивними методами. У доповнення до типового аналізу трафіку було запропоновано використання двох додатковий модулів, кожен з яких вирішує задачі визначення ролі та ідентифікації протоколів відповідно. Обидва модулі побудовані з використанням методів машинного навчання та опираються на поведінкові ознаки мережевої активності, що дозволяє уникнути необхідності в аналізі вмісту пакетів чи попередньому знанні специфікацій протоколів.

Проведено експериментальне дослідження запропонованої моделі, зокрема визначено найкращі математичні моделі для вирішення поставленої задачі. Їх ефективність визначена за допомогою як класичних метрик так і тестування на вплив шуму.

Ключові слова: операційні технології, SCADA, ICS, машинне навчання, розпізнавання протоколів, мережева конфігурація

ABSTRACT

The volume of the thesis is 59 pages. The thesis includes 45 figures, 3 tables, 4 appendices, and a 23 references.

The work examines methods for restoring the network configuration and identifying the attributes of each network object: IP address, MAC address, open ports, operating system, vendor, device role, and the industrial OT protocol used. Based on the conducted analysis and taking into account the specifics of the OT environment, it is proposed to use behavioral analysis to determine the role of an OT device and to identify the OT protocol it uses.

The proposed model defines an algorithm for restoring the network configuration using passive methods. In addition to typical traffic analysis, the model introduces two additional modules — each designed to solve the tasks of role identification and protocol recognition, respectively. Both modules are built using machine learning techniques and rely on behavioral patterns of network activity, which allows the system to avoid deep packet inspection or prior knowledge of protocol specifications.

An experimental evaluation of the proposed model was conducted. In particular, the most effective machine learning models for solving the defined tasks were identified. Their performance was assessed using both classical metrics and testing under varying levels of input noise.

Keywords: operational technology, SCADA, ICS, machine learning, protocol recognition, network configuration.

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СИМВОЛІВ, ОДИНИЦЬ, СКОРОЧЕНЬ І ТЕРМІНІВ

OT	операційні технології
IT	інформаційні технології
ICS	Industrial Control System, автоматизована система керування
SCADA	Supervisory Control and Data Acquisition, програмний пакет, призначений для розробки або забезпечення роботи в реальному часі систем збору, обробки, відображення та архівування інформації про об'єкт моніторингу або управління
DCS	Distributed Control System, автоматизована система керування технологічним процесом, що характеризується побудовою розподіленої системи введення-виведення та децентралізацією обробки данних
RTU	Remote Terminal Unit, віддалений пристрій збору данних і передачі команд, що забезпечує зв’язок між польовим обладнанням та центральною системою управління в OT-мережах
HMI	Human Machine Interface, інтерфейс, який дозволяє оператору візуалізувати та керувати технологічними процесами через графічні панелі або програмне забезпечення
PLC	Programmable Logic Controller, програмований контролер, призначений для автоматизації керування виробничими або технологічними процесами шляхом обробки сигналів і генерації команд
Vendor-Specific	характерний для певного вендора
Вендор	постачальник програмного або апаратного забезпечення
Дамп/ Дамп трафіку	перехоплений та збережений мережевий трафік
Датасет	структурований набір данних

NLP	загальний напрям інформатики, штучного інтелекту та математичної лінгвістики. Він вивчає проблеми комп'ютерного аналізу та синтезу природної мови

ВСТУП

Будь-яка сучасна компанія має регулярно стежити за станом своєї мережевої інфраструктури. Це включає інвентаризацію мережевих активів, аналіз досяжності між окремими вузлами мережі, моніторинг звичайного трафіку чи виявлення аномалій, контроль використання мережевих протоколів, тощо. Особливо важливо застосовувати ці заходи в OT-середовищі, де навіть мінімальна зміна цілісності системи може призвести до критичних наслідків.

Більшість OT-мереж експлуатуються десятками років, що призводить до застарівання використаних технологій та елементів мережі. Мережа стає чутливою до будь-яких змін чи втручання, це ускладнює інтеграцію сучасних засобів моніторингу та безпеки.

До того ж, OT-системи часто містять vendor-specific реалізації протоколів — спеціально модифіковані під певні потреби або закриті варіанти публічних протоколів, що частіше всього не мають документації.

Враховуючи сукупність вище згаданих факторів, OT-сектор потребує спеціальних підходів до управління мережевою інфраструктурою, включно з глибоким аналізом протоколів, врахуванням вендорських особливостей, а також адаптивними методами моніторингу стану мережі, які не порушують стабільність роботи критичних систем.

Актуальність теми зумовлена тим, що протягом багатьох років OT-системи розроблялися з пріоритетом на надійність, стабільність та безперервність процесів, але без належної уваги до питань кібербезпеки. Більшість таких систем створювалися як ізольовані середовища, і взаємодія з IT-мережею або зовнішнім світом не передбачалась. Однак сучасні OT-системи все частіше підключаються до загальних корпоративних або публічних мереж, це пов’язано з розвитком IIoT технології ,що створює нові вектори атак. У поєднанні з складністю моніторингу стану таких систем ризик кіберінцидентів значно підвищується

За статистикою [6], 73% OT-пристроїв залишаються без належного управління; тільки 24% компаній завершили проекти по розгортанню засобів безпеки; не дивлячись на зростаючі інвестиції в кібербезпеку OT-систем, рівень кваліфікації та кількість експертів залишаються критично низькою; за останні роки 61% вторгнень в систему змогли вплинути на неї; кількість IoT трафіку в промисловому середовищі складає близько 56.8 %. Надана статистика показує, що OT-сектор критично потребує уваги

Метою дослідження є аналіз та класифікація мережевих протоколів, що використовуються в OT-системах, а також розробка ефективних підходів до відновлення мережевої конфігурації з урахуванням специфіки промислового середовища.

Об'єктом дослідження є особливості функціонування мереж в розрізі ICS/SCADA

Предметом дослідження є особливості популярних OT-протоколів, методи їх класифікації та ідентифікації, а також методи збору інформації про стан мережі для відновлення її конфігурації

1. АНАЛІЗ МЕТОДІВ КЛАСИФІКАЦІЇ ОТ-ПРОТОКОЛІВ ТА ВІДНОВЛЕННЯ КОНФІГУРАЦІЇ МЕРЕЖІ

Сучасні промислові мережі, відіграють ключову роль у забезпеченні стабільної та безпечної роботи виробничих процесів. Однією з основних складностей у їх аналізі та захисті таких систем є різноманіття та специфічність використовуваних протоколів зв’язку, які часто є приватними або сильно адаптованими під конкретні виробничі системи. Класифікація ОТ-протоколів є важливим кроком для ідентифікації, моніторингу та виявлення аномалій у мережах промислової автоматизації. Одночасно, відновлення конфігурації мережі після збоїв або атак є надзвичайно актуальним завданням для підтримки працездатності промислових систем. Воно потребує не лише точного визначення типів протоколів, але й глибокого розуміння структури та взаємодії елементів мережі. Розглянемо сучасні методи класифікації приватних протоколів та відновлення мережевої конфігурації, звернемо увагу на особливості ОТ-середовища.

1.1 Класичні методи класифікації протоколів

Класичні методи класифікації та ідентифікації протоколів передбачають два підходи signature-based та port-based. Кожен з цих підходів має свої особливості, переваги та недоліки

1.1.1 Signature-based підхід

Аналіз сигнатур є одним із найпоширеніших підходів в IT-інфраструктурі, проте для ОТ-сегменту є дуже обмеженим. Цей метод базується на знаходженні певного набору байт, що однозначно співставляються використаному протоколу. Особливістю ОТ-мереж є використання приватних протоколів, публічної документації на які немає. Крім цього, через високу популярність кастомізації ОТ-протоколів під потреби компанії, їх структура може відрізнятись від стандартної,

що ускладнює аналіз. Також варто врахувати, що більшість OT-протоколів знаходяться на прикладному рівні моделі OSI, до них може бути використане шифрування, що робить сигнатуурний аналіз не ефективним підходом

1.1.2 Port-based підхід

Аналіз портів також є розповсюдженим рішенням. Метод базується на тому, що певні мережеві протоколи зазвичай використовують стандартні номери портів для встановлення з'єднання. Наприклад, протокол Modbus TCP зазвичай працює через порт 502, а DNP3 — через порт 20000. Цей підхід дозволяє ідентифікувати використання певного OT-протоколу в мережевому трафіку за номером порту, який фігурує у заголовках пакетів. Він широко використовується в системах IDS/IPS, фаєрволах та інших рішеннях мережевої безпеки.

Проте призначення порту до певного протоколу під час комунікації легко налаштувати, через що цей підхід не є надійним [13]

1.2 Глибокий аналіз пакетів

Глибокий аналіз пакетів [5] – метод при якому виконується аналіз вмісту пакету, який проходить через чекпоінт в мережі. Такий підхід дозволяє більш грунтовно виконувати класифікацію, і за часту підходить для аналізу невідомих/ приватних протоколів.

1.2.1 Використання методу аналізу частот n-грам

Метод [21] базується на автоматичному виявленні ключових ознак (сигнатур) мережевого протоколу шляхом аналізу частоти n-грам у потоках данних. Основна ідея полягає в тому, що повторювані послідовності байтів або символів можуть слугувати характерними рисами протоколу. Алгоритм складається з генерації n-грам, підрахунку частот та позиції, далі з використанням коефіцієнту Жаккара розраховується найбільш статистично значимі частоти та

ознаки. На основі цих ознак генеруються правила детектування для сигнатурного аналізу.

Метод добре підходить для ОТ-середовища, оскільки не потребує маркування данних та може кластеризувати невідомі приватні або модифіковані протоколи.

1.2.2 Обробка природньої мови

Обробка природньої мови [15] – сучасний напрям дослідження в сфері інформатики та машинного навчання, що займається розробкою алгоритмів та методів, які дозволяють комп'ютерам розуміти, обробляти та генерувати людську мову.

Прикладне застосування ця сфера знайшла і в аналізі мережевого трафіку [23]. При такому підході мережевий трафік сприймається як текст, що піддається аналізу методами NLP, а вміст протоколів як речення, що мають певну структуру та повторювані закономірності. Метод добре підходить для кластеризації невідомих протоколів і відновлення структури пакету.

1.3 Відновлення мережевої конфігурації

У сучасному мережевому середовищі, де постійно змінюються пристрої, додатки, сервіси та підключення, отримання точної та актуальної інформації про мережеву інфраструктуру є критично важливим. Відновлення мережевої конфігурації дозволяє ефективно інвентаризувати активи та оцінити ризики, вразливості, аномалії. Загально прийнято виділяти два методи відновлення мережевої конфігурації, активне сканування та пасивне виявлення пристроїв [19], кожен з яких має ряд переваг і недоліків . Розглянемо ці два поняття, та порівняємо урахуванням особливостей ОТ-середовища.

1.3.1 Active Assets Discovery

Активний метод передбачає цілеспрямовану генерацію запитів до пристроїв у мережі з метою отримання інформації про їхні характеристики, служби, відкриті порти тощо. Аналізується відповідь з кінцевої точки, отримаються цілові атрибути та агрегуються з метою створити звіт. Інформація яку можна отримати цим методом є більш насиченою, структурованою та добре піддається аналізу. Водночас, оскільки цей метод генерує мережевий трафік, він іноді може спричинити перебої в роботі сервісів або бути виявленим чутливими мережевими пристроями. Тому сканування необхідно планувати з урахуванням цих ризиків.

Згідно статистики [6, 10] більшість мереживих пристроїв в OT-секторі є застарілими, та погано піддаються оновлення, через це активне сканування може привести до критичних наслідків з дуже високою ймовірністю.

1.3.2 Passive Assets Discovery

Пасивний метод – метод при якому прослуховується трафік без прямої взаємодії з пристроями та на основі поведінки та взаємодії пристроїв. Оскільки ми не ініціюємо запити, а лише відслідковуємо звичайний трафік, інформації буде набагато менше. Метод потребує встановлення спеціального програмного або апаратного забезпечення на точку через яку проходить увесь трафік (наприклад, маршрутизатор)

Пасивне виявлення не створює додаткового навантаження на мережу, не змінює її поведінку та, головне, не становить загрози для стабільності чутливих компонентів, які широко представлені в ICS/SCADA системах.

1.4 Типові атрибути, що піддаються відновленню

Аналізуючи популярні open-source рішення, можна отримати розуміння про типові атрибути кожної мережі, які піддаються відновленню. Ці атрибути

формують базу для подальшої інтерпретації мережевих подій і є основою для роботи систем моніторингу. Розбиремо кожен з цих атрибутів.

1.4.1 MAC-адреса , IP-адреса ,відкриті порти

Першою групою параметрів є мережеві адреси та відкриті порти. Відновлення цих параметрів є задачею тривіальною, та не викликає складнощів. Отримати їх можна аналізуючи Ethernet , IP та TCP/UDP протоколи відповідно. Сучасні OT-мережі побудовані з використанням TCP/IP стеку [11], та більшість OT-протоколів є протоколами прикладного рівня, тому відновлення цих параметрів не є складною задачею

1.4.2 Операційна система

Операційна система є типовим атрибутом для відновлення. Для отримання інформації про ОС можна використати техніку OS fingerprinting [13,14]. Вона базується на тому, що кожна операційна система при відправці трафіку встановлює свої початкові параметри для ряду атрибутів, а саме: TTL, Window Size, Syn Size. Відповідно маючи базу данних сигнатур (цих трійок атрибутів), можна однозначно відобразити їх в ОС. Важливо розуміти, що ОС, які використовуються в промисловому середовищі відрізняються від звичайних нам, проте методологія залишається такою самою.

1.4.3 Вендор

Розуміння вендора надає більше інформації для комплексного аналізу структури мережі. Методологія отримання інформації про вендора базується на розумінні структури MAC-адреси. Перші 3 байти якої є OUI (Organization Unique Identifier), визначенням OUI для кожного вендора займається компанія IEEE [8], вона ж має базу данних співставлення OUI до назви компанії. Використання цієї бази забезпечує нам точну ідентифікацію постачальника.

1.4.4 Тип пристрою

Роль пристрою є одним із найскладніших для аналізу параметрів, оскільки вона не фіксується безпосередньо в заголовках мережевих пакетів і не має очевидного цифрового ідентифікатора. Визначення ролі потребує комплексного аналізу поведінки пристрою в мережі, його комунікаційних шаблонів, протоколів, що використовуються, та напрямків взаємодії з іншими вузлами.

У контексті OT-систем можна умовно виділити кілька типових ролей:

1) PLC (програмований логічний контролер) — пристрій, який приймає та виконує команди з вищого рівня (SCADA) та безпосередньо взаємодіє з фізичними процесами;
2) HMI (людино-машинний інтерфейс) — інтерфейс для оператора, зазвичай має інтенсивний трафік до PLC або SCADA;
3) SCADA-сервер — центральний координатор системи, що здійснює опитування контролерів, збір данних та керування процесами;
4) RTU — віддалений термінальний пристрій, який використовується для збору данних з польових сенсорів і передавання їх до SCADA-системи. Часто використовується в енергетичних, водоканальних або нафтогазових системах. Може працювати в складних умовах і підтримувати протоколи, як-от DNP3 або Modbus;
5) інженерна станція — пристрій, з якого здійснюється налаштування, програмування або оновлення ПЗ на інших компонентах OT-мережі;
6) сенсори, виконавчі пристрої, шлюзи тощо.

Під час дослідження було виявлено декілька методів, які намагаються визначити роль OT-пристрою

Перший [18] базується на ідеї того, що пристрої з однаковим типом/роллю будуть мати маленьку відстань між MAC-адресами. Це пов’язано з тим, що мережеве обладнання, встановлене та налаштоване одночасно (наприклад, PLC одного виробника), часто постачається з адресами, згенерованими з однакового або суміжного діапазону, що належить конкретному вендору.

Таким чином, шляхом обчислення відстані між MAC-адресами можна згрупувати пристрої в кластери, які з великою ймовірністю виконують схожі функції. Такий метод є досить ефективним у випадках, коли мережа була змонтована централізовано, а обладнання поставлялося партіями.

Проте даний підхід має низку обмежень. По-перше, він втрачає ефективність у випадках, коли обладнання постачається від різних виробників або оновлюється нерівномірно, що призводить до значного розкиду MAC-адрес у просторі. По-друге, метод потребує великого обсягу вихідних данних — чим більший датасет зібраних MAC-адрес, тим вища ймовірність точного виділення кластерів. При малій кількості активів або обмеженому часовому вікні спостереження кластеризація може бути неточною або взагалі неможливою.

Таким чином, використання методу доцільне лише в масштабних мережах з достатнім числом однотипних пристроїв та умовно стабільною структурою, або як допоміжний засіб у складі гібридного аналізу

Другий метод [9] пропонує використати поведінковий аналіз. При цьому визначається ранг SCADA-з'єднання – показник, який дозволяє розмежувати SCADA та звичайні з’єднання. Для того аби прорахувати метрики, з’єднання треба сегментувати. Сегментування з’єднання (четвірки IP-адреси відправника та отримувача та використані порти) використовує різницю в часі між пакетами. Якщо пакет прийшов пізніше критичного значення (наприклад 1 секунда), то це означає, що наступна комунікація відбувається в іншому сегменті .Метод передбачає розрахунок наступних метрик:

1) Періодичність (Periodicity) – метрика, що базується на спостереженні регулярності у передаванні пакетів. У типових OT-сценаріях, наприклад, SCADA-система з певною частотою опитує контролери (PLC), тому з'єднання між ними демонструють високу періодичність. На противагу цьому, трафік до інженерних станцій або HMI може мати менш регулярний характер

$$pR(ft_i) = \frac{mean(IAT_{ft_i})}{variance(IAT_{ft_i})} \quad \text{, де} \tag{1.1}$$

$ft_i = (IP_{src}, Port_{src}, IP_{dst}, Port_{dst}, SegSize)$ – сегмент комунікації (1.2)

IAT – час між приходом пакетів

2) Стійкість зв'язку (Communication durability) – метрика, що базується на ідеї того, що як тільки зв'язок між SCADA та PLC встановлений, він буде підтримуватись досить довго. На відміну від тимчасових або сервісних підключень, характерних для інженерних станцій або зовнішнього втручання, основні OT-компоненти взаємодіють у стабільних і довготривалих сесіях

$$dR(ft_i) = \sum IAT_{ft_i} * \log n_{ft_i} \quad \text{, де} \tag{1.3}$$

n_{fti} – кількість появи сегменту комунікації в датасеті

3) Різниця складності пристроїв (Device Complexity Gap) – метрика, що відображає рівень функціональної складності пристроїв у мережі, базуючись на різноманітності використаних мережевих портів. У типових OT-мережах існує суттєвий контраст між польовими та централізованими вузлами. Польові пристрої зазвичай мають фіксовану функцію — періодичну передачу данних або реакцію на аварійні команди — і використовують обмежену кількість мережевих портів, часто лише один, зарезервований для специфічного OT-протоколу. На відміну від них, SCADA-сервери повинні обробляти одночасні з'єднання з великою кількістю пристроїв, а також надавати доступ до додаткових сервісів.

$$cR(ft_i) = \frac{|PT_{SrcIp}|}{|PT_{DstIp}|}, \quad \text{де} \tag{1.4}$$

PT – множина відкритих портів пристрою

4) Популярність мережевого сервісу (Network Service Popularity) — метрика, яка дозволяє відрізнити типові SCADA-з’єднання від допоміжних або клієнт-серверних взаємодій, ґрунтуючись на тому, наскільки часто певний порт використовується різними пристроями. Ідея полягає в наступному: польові пристрої зазвичай використовують фіксовані порти для взаємодії з SCADA, тому один і той самий порт зустрічається у багатьох з’єднаннях між різними IP-адресами. Це робить такі порти "популярними" в мережі. Натомість мастер-сервер (наприклад, SCADA) ініціює з’єднання з багатьма польовими пристроями, використовуючи динамічний вибір портів, через що кожен порт має низьку повторюваність.

$$uR(ft_i) = \frac{|PU_{srcPort}|}{|PU_{dstPort}|}, \quad \text{де} \tag{1.5}$$

PU – множина з’єднань з іншими пристроями використовуючи цей порт

Кожна з метрик поодинці не виділяє ефективно тип девайсу, проте за допомогою цих метрик можна вирахувати певний ранг для сегменту комунікації, який буде показувати приналежність цього сегменту до ОТ-з’єднання. Запропонована формула рангу:

$$f^* = \underset{ft}{argmax}\ (pR(ft_i) * dR(ft_i) * cR(ft_i) * uR(ft_i) * segSize(ft_i)) \tag{1.6}$$

За допомогою рангу можна відфільтрувати не ОТ-з’єднання та на основі заздалегідь відомої інформації про структуру мережі та аналізі степіня з’єднання кожного з пристроїв визначити типи активів.

Метод показав чудові результати, проте він вимагає попереднього розуміння архітектури і структури мережі на яку застосовується, чого хотілося б уникнути.

Також можна використовувати **класичний метод**, коли у відповідність запущеному ПО, та фактору того чи є актив ініціатором з’єднання, ставиться у відповідність тип активу, так наприклад знаючи, що комунікація відбувається між

пристроями А та Б, протокол комунікації Modbus і ініціатором є пристрій А, то ймовірно, що А – це SCADA, а Б – PLC. Проте належних матриць відповідності для вирішення задачі класифікації таким методом немає.

1.4.5 Відновлення топології мережі

На відміну від звичайних IT-мереж, ICS-середовища мають специфічні обмеження: активне сканування або опитування пристроїв може бути небезпечним і порушити роботу критичної інфраструктури. Через це застосування класичних методів виявлення топології стає обмеженим або зовсім неприйнятним. Водночас, ICS-мережі мають властивості, які можна ефективно використати для пасивного виявлення структури мережі. Більшість взаємодій у таких системах відбувається в рамках автоматизованих процесів, а отже, характеризується регулярністю та стабільністю. Наприклад, пристрої рідко змінюються або додаються, а шаблони трафіку мають чітку періодичність. Це дозволяє не лише реконструювати топологію, але й виявляти нові або аномальні пристрої та зв'язки. Можна виділити три основних підходи: **наївний підхід**, **підхід на основі спеціальних протоколів** та **підхід на основі аналізу базових протоколів** [14].

Наївний підхід не використовує ніяких додаткових методик окрім простого аналізу трафіку 3-4 рівня. Під час використання цього методи будуються зв’язки на основі IP-адрес, що фігурували в трафіку без додаткової обробки. Результуючий граф при цьому є графом досяжності, але не справжньою фізичною топологією

Підхід на основі аналізу спеціальних протоколів передбачає парсинг SNMP, CDP, STP, OSPF, SSDP, RSTP, RIP, IGRP — це протоколи, які можуть використовуватись пристроями другого або третього рівня моделі OSI з метою оптимізації роботи мережі, її моніторингу та побудови маршрутизації. Такі протоколи часто містять інформацію про топологію мережі, маршрути, сусідні пристрої, інтерфейси, статус з'єднань та інші параметри, що можуть бути корисними для пасивного виявлення структури мережі. Проблема такого підходу це те, що монітор на якому встановлене ПЗ по прослуховуванню трафіка може

знаходитись в ділянці мережі, в яку не надходять ці протоколи, або ж вони просто не імплементовані на пристроях 3-4 рівня

Підхід на основі аналізу базових протоколів пропонує дивитись на IP-трафік. При аналізі використовується певний набір правил:

- Якщо на одній LAN перехоплюються кадри з однаковими MAC-адресами джерела, але різними IPv4-адресами джерела — ці пакети, ймовірно, надходять із іншої LAN. При цьому MAC-адреса, ймовірно, належить локальному інтерфейсу маршрутизатора, що знаходиться на межі контрольованої LAN.
- Якщо захопити пакет і знати ймовірне початкове значення TTL, то, віднявши поточне TTL від початкового, можна отримати кількість мережевих переходів (hops) між сенсором і джерелом. Наприклад, якщо отримано пакет із TTL 125, то можна припустити, що початкове TTL було 128 (2^7), отже, різниця — 3. Це означає, що пакет надійшов з LAN, яка знаходиться за 3 переходи від сенсора.

Такий підхід на додачу до наївного дозволяє сегментувати мережу на певні LAN, та частково визначити маршрутизатори, що використовувались під час комунікації .

.

ВИСНОВКИ ДО РОЗДІЛУ 1

У першому розділі було проведено комплексний аналіз сучасних підходів до відновлення мережевої конфігурації та класифікації OT-протоколів. Визначено проблематику теми, підкреслено особливості OT-середовища, які зумовлюють розробку спеціальних рішень та підходів.

. Було виявлено, що більшість OT-протоколів імплементовані на 7 рівні моделі OSI Задача класифікації OT-протоколів на разі має 4 основні підходи, проте вони мають низку недоліків, таких як нестійкість до шифрування, неможливість застосування при кастомізації протокола чи застосунку, який його відправляє. Саме тому зростає потреба в створені методу, який не базується на аналізі корисного навантаження пакету, а послуговується іншою інформацією. Таким методом може бути поведінковий аналіз.

Задача відновлення мережевої конфігурації має два основі підходи: активне сканування та пасивне виявлення. Задача є актуальною як для IT- так і для OT-середовища, проте було виявлено, що через застарілість технологій, які використовує OT-сектор, активний метод не є доцільним для використання, що ускладнює вирішення задачі. Проаналізувавши роботу популярних Network Mapping – застосунків, було виділено ряд атрибутів, що підлягають відновленню. Проаналізувавши методи їх отримання було виявлено, що відновлення типу пристрою є не тривіальною задачею, особливо в OT-середовищі. Було виявлено дефіцит наукових праць чи методологій, які описують вирішення задачі, а існуючі не опираються принцип no prior knowledge, що робить їх не адаптивними для різних інфраструктур. Також у зв’язку з природою пасивного методу відновлення мережевої конфігурації, отримання справжньої топології мережі є складною задачею та не має стандартних методів аналізу.

Таким чином завдання дослідження можна звести до пунктів: 1) Розробка методу класифікації OT-протоколів шляхом поведінкового аналізу 2) Розробка

методу відновлення ролі пристрою, попередньо не знаючи інфраструктуру мережі.
3) Розробка загального методу відновлення мережевої конфігурації з використанням методів 1) та 2)

2 МОДЕЛЬ ВІДНОВЛЕННЯ МЕРЕЖЕВОЇ КОНФІГУРАЦІЇ З ВИКОРИСТАННЯМ ЕЛЕМЕНТІВ ПОВЕДІНКОВОГО АНАЛІЗУ

У складних мережах промислових систем керування традиційні методи відновлення мережевої конфігурації, які базуються виключно на статичних атрибутах пристроїв, часто виявляються недостатніми. Особливо в ICS-мережах, де активне сканування може бути неприйнятним, виникає потреба в більш гнучких та ненав'язливих способах аналізу мережі. Поведінковий аналіз, який базується на спостереженні реальної та типової мережевої активності пристроїв у процесі їх роботи, дозволяє отримати більш точну й актуальну картину топології без порушення технологічного процесу. У цьому розділі буде запропоновано загальну модель відновлення мережевої конфігурації OT-систем

2.1 Архітектура системи

Запропонована модель відновлення мережевої конфігурації, зображеної на рисунку 2.1, об’єднує методології описані в розділі 1. Архітектура поділена на модулі, реалізації яких може змінюватись, доповнюватись та покращуватись в залежності від потреб підприємства, що використовує модель.

Основний потік обробки трафіку передбачає агрегацію та структурування данних для її ефективного аналізу; ідентифікацію пристроїв разом з базовим набором атрибутів; побудови наївних з’єднань; шляхом комплексного аналізу ідентифікація справжніх пристроїв та з’єднань, відновлення сегментів LAN та пристроїв 2 та 3 рівні моделі OSI; генерацію звіту. Модель доповнена двома додатковими модулями: модуль типізації пристроїв, та модуль ідентифікації OT-протоколів, які по-можливості насичують загальну структуру данних додатковою інформацією. Розглянемо кожен з етапів і модулів окремо.

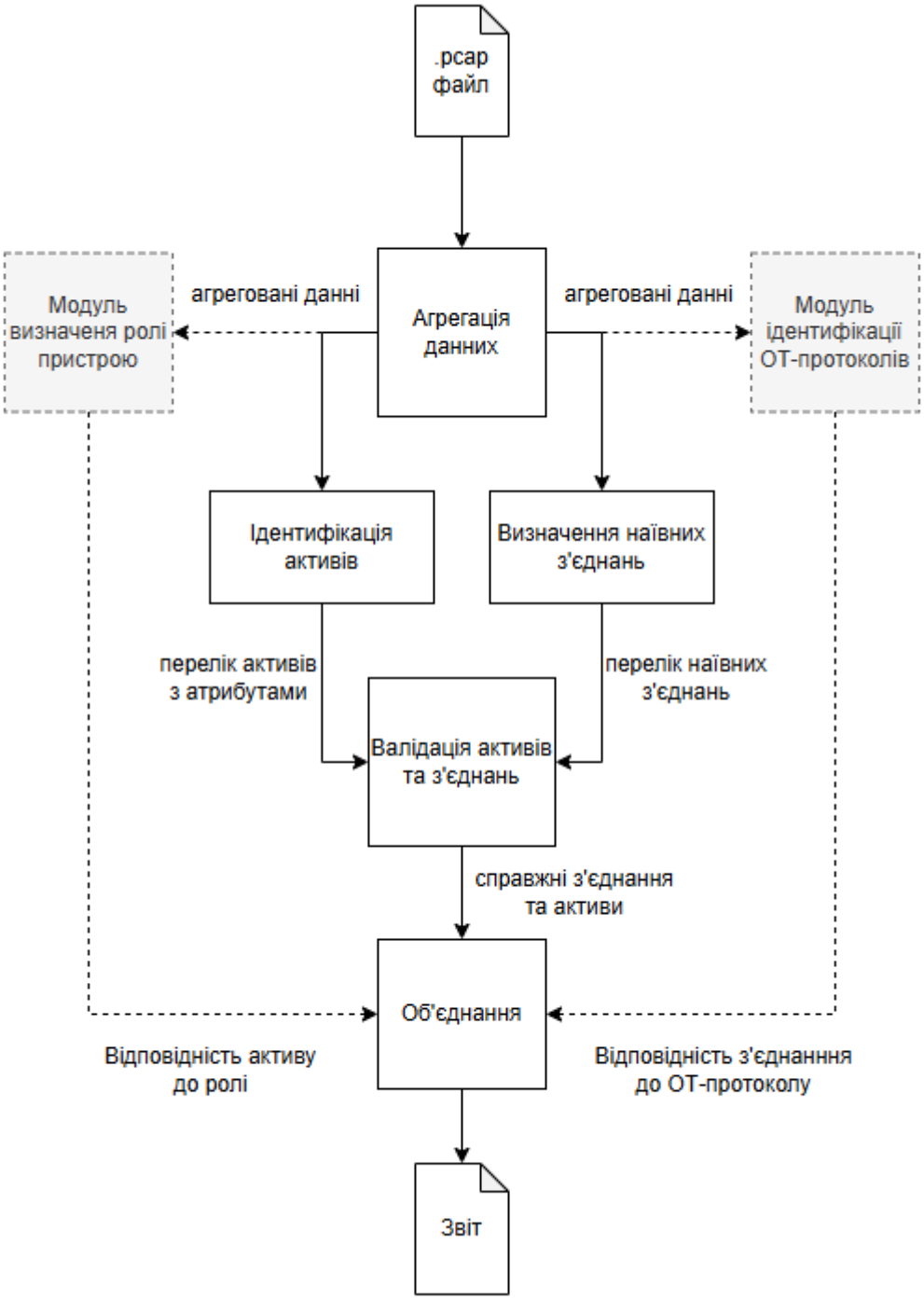

Рисунок 2.1 – Блок-схема моделі

2.2 Агрегація та обробка данних

Перший етап передбачає зчитування дампу трафіка, аналіз протоколів 2-4 рівнів моделі OSI та структуризацію данних. При цьому параметри, які потрібно отримати під час аналізу наступні: MAC-, IP-адреси, використані порти, використаний протокол транспортного рівня, TTL, Window size, TCP-флаги та timestamp. Такі параметри дають основу для подальшого аналізу структури мережі, визначення взаємозв'язків між хостами та виявлення активних пристроїв.

Таблиця 2.1 – Структура данних після первинної агрегації

Поле	Опис	Тип
srcmac	MAC-адреса відправника	Str
dstmac	MAC-адреса отримувача	Str
srcip	IP-адреса відправника	Str
dstip	IP-адреса отримувача	Str
srcport	Порт, який використовувався під час відправки	Int
dstport	Порт, який використовувався для отримання	Int
timestamp	Час отримання пакету	Float
hkproto	Використаний протокол транспортного рівня	TCP | UDP
ttl	Time to Live – кількість стрибків, які ще може зробити пакет	Int
winsz	Windows Size – розмір вікна що фігурує в TCP-заголовку	Int
flags	TCP-прапорці	Str
pktsz	Загальний розмір пакету	Int

Агреговані данні використовують 4 різні методи підготовки інформації для 4 потоків обробки:

Задля ідентифікації активів та насичення їх атрибутами має бути застосоване групування навколо IP-адрес. Кожна IP-адреса буде виступати в якості ідентифікатора активу, при цьому ми будемо нехтувати ймовірністю того, що один і той самий пристрій може мати декілька мережевих інтерфейсів, для моделі це будуть декілька різних пристроїв, які характеризуються по-різному.

```
syn_mask = df['flags'].apply(lambda x: 'S' in str(x) if pd.notna(x) else False)
syn_df = df[syn_mask]


syn_sizes = syn_df.groupby('srcip')['pktsz'].agg(
    synsz=lambda x: x.dropna().mode().iloc[0] if not x.dropna().mode().empty else
None
).reset_index()


src_assets = df.groupby('srcip').agg(
    macs=('srcmac', lambda x: set(x.dropna())),
    ports=('srcport', lambda x: set(x.dropna())),
    ttl=('ttl', lambda x: x.dropna().mode().iloc[0] if not x.dropna().mode().empty
else None),
    winsz=('winsz', lambda x: x.dropna().mode().iloc[0] if not
x.dropna().mode().empty else None),
)


src_assets = src_assets.merge(syn_sizes, on='srcip', how='left')


dst_assets = df.groupby('dstip').agg(
    macs=('dstmac', lambda x: set(x.dropna())),
    ports=('dstport', lambda x: set(x.dropna()))
)


src_assets = src_assets.reset_index().rename(columns={'srcip': 'ip'})
dst_assets = dst_assets.reset_index().rename(columns={'dstip': 'ip'})


all_assets = pd.concat([src_assets, dst_assets], ignore_index=True)


assets = all_assets.groupby('ip').agg({
    'macs': lambda sets: set.union(*sets),
    'ports': lambda sets: set.union(*sets),
    'ttl': lambda series: next((x for x in series if pd.notna(x)), None),
    'winsz': lambda series: next((x for x in series if pd.notna(x)), None),
```

```
    'synsz': lambda series: next((x for x in series if pd.notna(x)), None)
}).reset_index()
```

Також на цьому етапі буде відбуватись групування за MAC-адресою, в результаті такого групування буде зберігатись лише кількість IP-адрес, відповідно до призначеного MAC-адресу, ця інформація буде використана під час етапу валідації.

```
src_macs = df.groupby('srcmac').agg(ips=('srcip', lambda x: set(x.dropna())))
src_macs = src_macs.reset_index().rename(columns={'srcmac': 'mac'})

dst_macs = df.groupby('dstmac').agg(ips=('dstip', lambda x: set(x.dropna())))
dst_macs = dst_macs.reset_index().rename(columns={'dstmac': 'mac'})

macs = pd.concat([src_macs, dst_macs], ignore_index=True)
macs = macs.groupby('mac').agg(
    ips=('ips', lambda sets: set.union(*sets))
).reset_index()
macs['count'] = macs['ips'].apply(len)
```

Етап відновлення з’єднань буде використовувати групування за IP-адресами описаному вище.

```
def get_naive_connections(df):
    return df.groupby(['srcip', 'dstip', 'srcport', 'dstport']).size()
.reset_index(name='count'))
```

Модуль визначення ролі пристрою використовує сегментування, яке засноване на особливості OT-пристроїв такій, як періодичність та стійкість зв’язку. Кожен потік між фіксованими IP-адресами та портами ділиться на основі того, як

давно прийшов попередній фрейм, експериментально було визначено цей час в 1 секунду. Для кожного сегменту розраховуються поведінкові метрики.

```
@profiling_logger('Periodicity metric')
def get_periodicity(df:pd.DataFrame)->List[int]:
    return df.apply(lambda row: row['iiat_mean']/row['iiat_var'] if
pd.notna(row['iiat_mean']) and pd.notna(row['iiat_var']) and row['iiat_var'] != 0
else 0.0, axis=1)
```

```
@profiling_logger('Durability metric')
def get_durability(df:pd.DataFrame, extra_columns=[])->List[int]:
    group_columns = ['source_ip', 'source_port', 'destination_ip',
'destination_port', 'seg_size', *extra_columns]

    group_counts =
df.groupby(group_columns).size().reset_index(name='group_count')
    df = df.merge(group_counts, on=group_columns, how='left')

    return df.apply(lambda row: row['iiat_sum']*np.log(row['group_count']),axis=1)
```

```
@profiling_logger('Complexity Gap metric')
def get_complexity(df:pd.DataFrame, extra_columns = [])->List[int]:

    src_ports = df[['source_ip', 'source_port'] +
extra_columns].rename(columns={'source_ip': 'ip', 'source_port': 'port'})
    dst_ports = df[['destination_ip', 'destination_port'] +
extra_columns].rename(columns={'destination_ip': 'ip', 'destination_port':
'port'})

    all_ports = pd.concat([src_ports, dst_ports], ignore_index=True)
```

```
    unique_ports_count = (
        all_ports.groupby(['ip'] + extra_columns)['port']
        .nunique()
        .reset_index()
        .rename(columns={'ip': 'ip', 'port': 'unique_ports'})
    )


    src_unique = unique_ports_count.rename(columns={'ip': 'source_ip',
'unique_ports': 'src_unique_ports'})
    dst_unique = unique_ports_count.rename(columns={'ip': 'destination_ip',
'unique_ports': 'dst_unique_ports'})


    df = df.merge(src_unique, on=['source_ip'] + extra_columns, how='left')
    df = df.merge(dst_unique, on=['destination_ip'] + extra_columns, how='left')


    df['dst_unique_ports'].replace(0, pd.NA, inplace=True)
    df['complexity_gap'] = df['src_unique_ports'] / df['dst_unique_ports']
    df['complexity_gap'] = df['complexity_gap'].fillna(0)


    return df['complexity_gap']
```

```
@profiling_logger('Service Popularity metric')
def get_service_popularity(df:pd.DataFrame, extra_columns=[])->List[int]:
    groupby_src_cols = ['source_port', 'source_ip', 'destination_ip'] +
extra_columns
    groupby_dst_cols = ['destination_port', 'source_ip', 'destination_ip'] +
extra_columns


    src_triplet_popularity = (
        df.groupby(groupby_src_cols)
        .size()
        .reset_index(name='src_triplet_count')
```

```
    )

    dst_triplet_popularity = (
        df.groupby(groupby_dst_cols)
        .size()
        .reset_index(name='dst_triplet_count')
    )
    df = df.merge(
        src_triplet_popularity,
        on=groupby_src_cols,
        how='left'
    )
    df = df.merge(
        dst_triplet_popularity,
        on=groupby_dst_cols,
        how='left'
    )
    df['dst_triplet_count'] = df['dst_triplet_count'].replace(0, pd.NA)
    df['uR'] = df['src_triplet_count'] / df['dst_triplet_count']
    df['uR'] = df['uR'].apply(lambda x: 1/x if pd.notna(x) and x < 1 else x)
    df['uR'] = df['uR'].fillna(0)

    return df['uR']
```

Модуль ідентифікації ОТ-протоколів передбачає фільтрацію усіх потоків, що були нижче транспортного рівня. Далі потрібно виконати першочергове групування на основі timestamp-у, загальний трафік буде поділений на сегменти по 30 секунд.

```
def split_into_time_frames(df, time_col = 'timestamp', gap = 30):
    df = df.copy()
    df = df.sort_values(time_col)
```

```
    parts = []
    part_id = 0
    min_time = df[time_col].min()
    max_time = df[time_col].max()
    current_start = min_time
    while current_start < max_time:
        current_end = current_start + gap
        mask = (df[time_col] >= current_start) & (df[time_col] < current_end)
        part_df = df[mask].copy()
        if not part_df.empty:
            part_df['part_id'] = part_id
            part_df['part_start'] = current_start
            part_df['part_end'] = current_end
            parts.append(part_df)
        current_start = current_end
        part_id += 1
    if not len(parts): parts = [df]
    return parts
```

Таке сегментування дозволяє отримати чіткіші метрики поведінки. Далі використовується групування на основі IP-адрес та портів разом з розрахунком специфічних поведінкових метрик.

```
df = df.groupby(['srcip','srcport','dstip','dstport', 'hkproto']).agg(
            packet_count=('pktsz', 'count'),
            byte_count=('pktsz', 'sum'),
            mean_packet_size=('pktsz', 'mean'),
            std_packet_size=('pktsz', 'std'),
            min_packet_size=('pktsz', 'min'),
            max_packet_size=('pktsz', 'max'),
            duration=('timestamp', lambda x: x.max() - x.min()),
            inter_arrival_time_mean=('timestamp', lambda x:
x.sort_values().diff().mean() if x.count() > 1 else np.nan),
```

```
            inter_arrival_time_std=('timestamp', lambda x:
x.sort_values().diff().std() if x.count() > 2 else np.nan),
        ).reset_index()
```

Формули поведінкових метрик є адаптованими формулами 1.1 – 1.5 для використання з іншим сегментуванням. Приклад розрахунку наведено в лістингу.

```
source_portusage = df.groupby(['source_ip']).agg(src_port_usage=('source_port',
'nunique'))
destination_portusage =
df.groupby(['destination_ip']).agg(dst_port_usage=('destination_port', 'nunique'))


df = df.merge(source_portusage, on=['source_ip'], how='left')
df = df.merge(destination_portusage, on=['destination_ip'], how='left')


df['periodicity'] = df.apply(
    lambda row: row['inter_arrival_time_mean'] / row['inter_arrival_time_std']
    if row['inter_arrival_time_std'] != 0 else 0,
    axis=1
)


df['durability'] = df.apply(
    lambda row: row['inter_arrival_time_mean'] * row['packet_count'],
    axis=1
)


df['complexity'] = df.apply(
    lambda row: row['src_port_usage'] / row['dst_port_usage']
    if row['dst_port_usage'] != 0 else 0,
    axis=1
)
```

2.3 Ідентифікація активів та визначення наївних з’єднань

Після процесу агрегації данних для ідентифікації активів навкого IP-адреси більшість потрібних атрибутів стає автоматично відомими, а саме: IP-, MAC-адреси та відкриті порти. Далі ми застосовуємо підходи OS-fingerprinting та ідентифікації за OUI , використовуючи потрібні бази данних, для того аби відновити ОС та вендора відповідно.

```
#os_df – database with os-fingerprinting parameters mapping to os name and vesrion
def get_os(ttl, win_size, syn_size):
    if pd.isna(ttl) or not ttl or pd.isna(win_size) or not win_size: return None

    # get_true_ttl – method which returns the nearest power of 2 rounding up
    ttl = get_true_ttl(ttl)
    os = os_df[
        (os_df['ttl'] == int(ttl)) &
        (os_df['win size'] == int(win_size)) &
        (os_df['win size'] == int(syn_size) if syn_size else True)
    ]

    if not len(os): return None
    os = os.iloc[0]

    return f"{os['os']} {os['major version']}"

#oui – database with oui mapping to vendor name
def get_vendor(mac):
    if not mac: return None
    return oui.get(mac.upper().replace(':','')[:6])
```

Наївні з’єднання не потребують додаткової обробки, все що потрібно це структурувати вже відомі данні та передати їх на наступний етап.

2.4 Валідація базової структури мережі

Етап валідації намагається відновити якомога більше інформації про справжню топологію мережі. При цьому використовується аналіз зібраної заздалегідь інформації про активи та їх графи досяжності. Якщо одній MAC-адресі відповідають багато IP-адрес, то ймовірно, що ця MAC-адреса належить маршрутизатору, а IP-адреси належать пристроям, що знаходяться в іншій ділянці LAN. Аналогічно, якщо одній IP-адресі відповідають декілька MAC-адрес, то ймовірно що пристрої зі знайденими MAC-адресами об’єднані під одним роутером зі знайденою MAC-адресою. Відповідні з’єднання так само мають бути оновлені.

```
def infer_topology(macs, assets, connections):
    assets_dict, cons = {}, set()
    for i, row in assets.iterrows():
        row_dict = row.to_dict()
        # row_dict.pop('ip', None)
        row_dict.pop('macs', None)
        row_dict['ports'] = drop_not_service_ports(row['ports'])
        if len(row['macs']) > 1:
            assets_dict[row['ip']] = {"type": ROUTER, 'ip': row['ip'], 'id':
row['ip'], "ttl":row_dict['ttl']+1 if row_dict['ttl'] else None}
            for mac in row['macs']:
                row_dict['mac'] = mac
                row_dict['vendor'] = macs[macs['mac'] == mac].iloc[0]['vendor']
                row_dict['id'] = f"{row['ip']}_{mac}"
                assets_dict[f"{row['ip']}_{mac}"] =
{**row_dict,"ttl":row_dict['ttl']+1 if row_dict['ttl'] else None }
                cons.add(tuple(sorted((row['ip'], row_dict['id']))))
        if len(row['macs']) == 1:
            row_dict['mac'] = list(row['macs'])[0]
            row_dict['vendor'] = macs[macs['mac'] ==
row_dict['mac']].iloc[0]['vendor']
            row_dict['ip'] = row['ip']
```

```
            row_dict['id'] = row['ip']
            assets_dict[f"{row_dict['ip']}"] = row_dict
    for key in list(assets_dict.keys()):
        values = assets_dict[key]
        ttl = values.get('ttl')
        if ttl:
            true_ttl = get_true_ttl(ttl)
            prev_k = key
            for i in range(true_ttl - ttl):
                k = f"{key}_mw_{i}"
                prev_o = dict(assets_dict[prev_k])  # make a shallow copy
                assets_dict[prev_k] = {"type": ROUTER, "id": prev_k,
"ttl":prev_o['ttl'] + 1}
                assets_dict[k] = {**prev_o, 'id': k}
                print(prev_k,k)
                cons.add(tuple(sorted((prev_k, k))))
                prev_k = k

    for i, row in connections.iterrows():
        src = row['srcip']
        dst = row['dstip']
        cons.add(tuple(sorted((src, dst))))

    return assets_dict, cons
```

2.5 Поведінковий аналіз в OT-секторі

Для вирішення задачі типізації пристрою та ідентифікації протоколу буде використано поведінковий аналіз. Такий метод не потребує попереднього знання про інфраструктуру та базується на тому, що типи пристроїв та протоколи мають

свій специфічний профіль комунікації, а конкретні випадки кожного з класів реалізують цей профіль з доданням деякого шуму.

Окрім стандартних метрик, що використовуються при поведінковому аналізі, розраховуються і специфічні для ОТ-систем, які експлуатують їх особливості, а саме: періодичність, різниця в складності пристроїв, популярність мережевого пристрою, стійкість зв'язку – детально вони описані в розділі 1 та приклади розрахунку в попередніх пунктах. Стандартні ж метрики наступні: розмір пакету (середній, мінімальний, максимальний, стандартне відхилення), час між пакетами (середній, мінімальний, максимальний, стандартне відхилення), кількість пакетів та тривалість зв'язку.

Для того аби визначити профілі комунікації, буде використано машинне навчання. При цьому потрібен підготовчий етап, який включає в себе збір еталонних данних, їх обробку, визначення параметрів, навчання математичної моделі та валідацію результатів.

2.6 Математична модель ідентифікації протоколів

Математична модель, що використовується для ідентифікації протоколів, що використовуються в ОТ-мережах , базується на гіпотезі про наявність усталених поведінкових патернів для кожного типу протоколу. Ці патерни проявляються у вигляді специфічних характеристик трафіку на транспортному та мережевому рівнях, зокрема у структурі сесій, періодичності обміну повідомленнями, обсягах даних та часових параметрах передачі.

Модель реалізується як задача багатокласової класифікації, де кожен клас відповідає певному протоколу. Для побудови моделі застосовуються методи машинного навчання з вчителем, які дозволяють виявляти залежності між вхідними параметрами трафіку та мітками протоколів. Навчання проводиться на основі

попередньо зібраного та обробленого датасету, що включає приклади трафіку з відомою приналежністю до певного протоколу. При цьому частина данних буде спеціально зашумлена за для збільшення стійкості до шуму. Після навчання модель здатна приймати на вхід нові зразки трафіку та з високим ступенем імовірності визначати приналежність до одного з відомих протоколів або сигналізувати про появу невідомої структури, що може свідчити про аномалію або новий тип протоколу.

Таким чином, математична модель ідентифікації протоколів виконує функцію автоматизованого аналізу поведінки мережевих об'єктів на основі статистично значущих характеристик комунікації без використання глибокого аналізу вмісту пакету.

2.7 Математична модель типізації пристроїв

Математична модель типізації пристроїв ґрунтується на припущенні, що кожен тип мережевого пристрою , наприклад, контролер, SCADA, тощо, характеризується унікальним поведінковим профілем, який можна формалізувати через набір статистичних ознак мережевої активності. Типізація передбачає віднесення спостережуваного пристрою до одного з наперед визначених класів на основі параметрів його комунікації в мережі.

З математичної точки зору, процес типізації формалізується як задача багатокласової класифікації. Для цього агреговані характеристики трафіку кожного пристрою репрезентуються у вигляді вектора ознак, який описує кількісні та якісні параметри взаємодії з іншими мережевими об'єктами.

Навчання моделі здійснюється на основі маркованого набору даних, у якому кожному пристрою зіставлений відповідний клас. При цьому частина данних буде спеціально зашумлена за для збільшення стійкості до шуму. Для моделювання

використовуються як лінійні, так і нелінійні алгоритми машинного навчання. Таким чином, математична модель типізації пристроїв дозволяє автоматизовано визначати функціональну роль об'єктів у мережі без попереднього знання про інфраструктуру.

2.8 Визначення вхідних параметрів для моделей

Етап інженерії вхідних параметрів для математичних моделей є дуже важливим. Наявність надлишкових або нерелевантних ознак може не лише ускладнювати процес навчання моделі, але й знижувати її узагальнюючу здатність, призводити до перенавчання або втрати інформативності при інтерпретації результатів.

У рамках даного дослідження вибір вхідних параметрів базується як на доменних знаннях щодо специфіки OT-мереж, так і на попередньому статистичному аналізі зібраного трафіку. Основна мета полягає у визначенні таких характеристик, які дозволяють відобразити поведінкові відмінності між класами об'єктів або протоколів за мінімального рівня кореляції між самими ознаками.

Першочергово було запропоновано ряд метрик, описаних в розділі 2.5. Далі було проаналізовано кореляційні матриці параметрів для кожної з моделей, матриці зображені на рисунках 2.2-2.3. Матриць було дві оскільки застосовано різні принципи групування та обробки сирих данних. Далі шляхом аналізу впливу кожного з параметрів на модель, найменш значущі прибирались до поки метрики точності описані в розділі 2.10 не ставали гіршими. Таким чином отримано наступні вхідні параметри для кожної з задач класифікації:

Ідентифікація OT-протоколів: кількість пакетів, розмір пакету (мінімальний, максимальний, середній, стандартне відхилення), тривалість, середній час між пакетами, популярність порту відправника, популярність порту отримувача, періодичність, стійкість зв’язку, різниця складності.

Визначення ролі пристрою: розмір сегменту, розмір пакету (мінімальний, максимальний), тривалість, мінімальний час між пакетами, періодичність, різниця складності, популярність, ранг

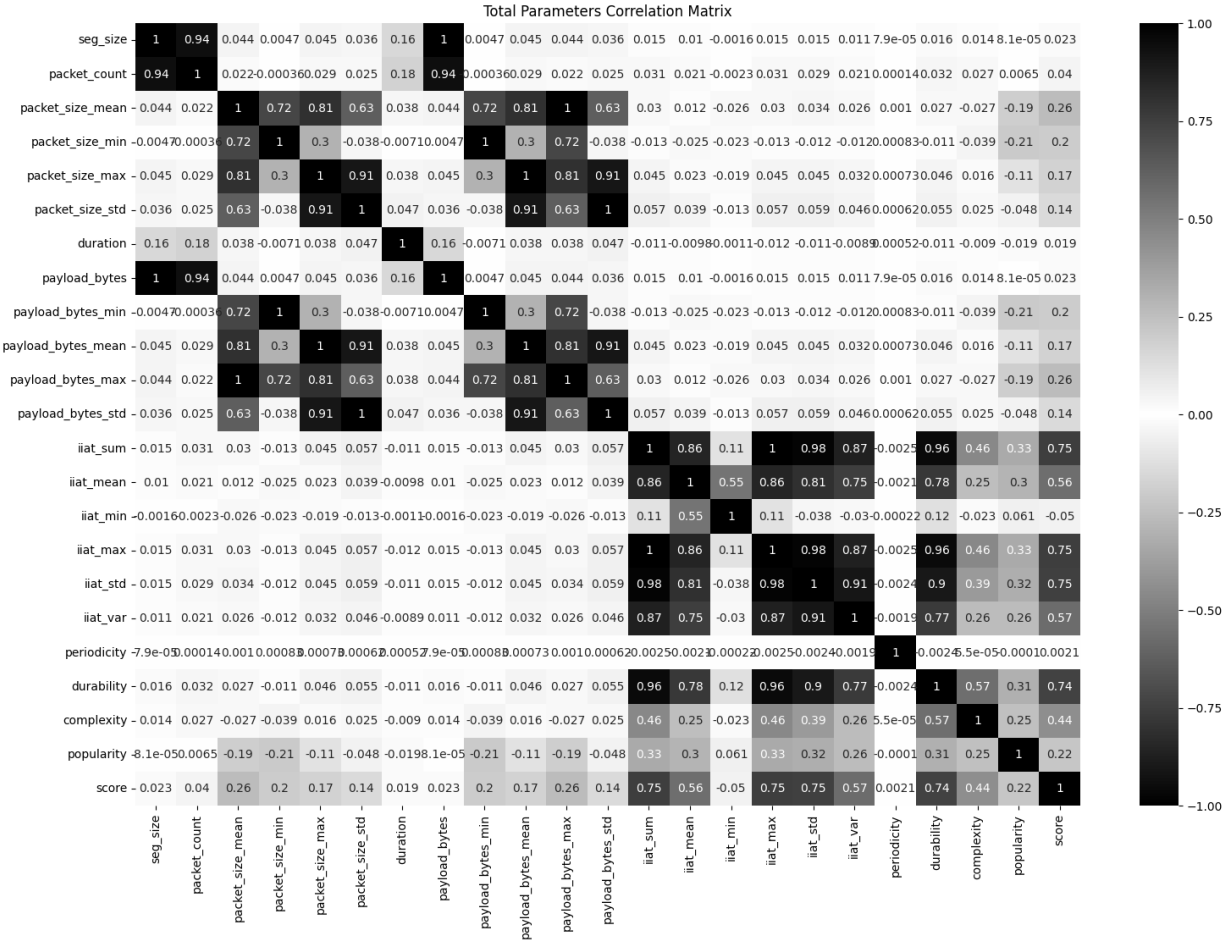

Рисунок 2.2 – Кореляційна матриця параметрів для задачі типізації активів

2.9 Вирішення проблеми дисбалансу класів

Під час навчання математичних моделей може виникнути проблема дисбалансу класів. Це обумовлено тим, що більшість інфраструктур може реалізовувати та підтримувати обмежену кількість OT-протоколів, оскільки частина з них прив’язані до конкретного вендора. Або ж у ряді випадків, присутні лише окремі функціональні ролі пристроїв, які репрезентують вузьке коло класів, тоді як інші типи пристроїв або протоколів зовсім не зустрічаються або

представлені одинично. Такий дисбаланс призводить до перекосу навчання моделі на користь домінуючих класів, що знижує точність розпізнавання малочисельних або рідкісних класів.

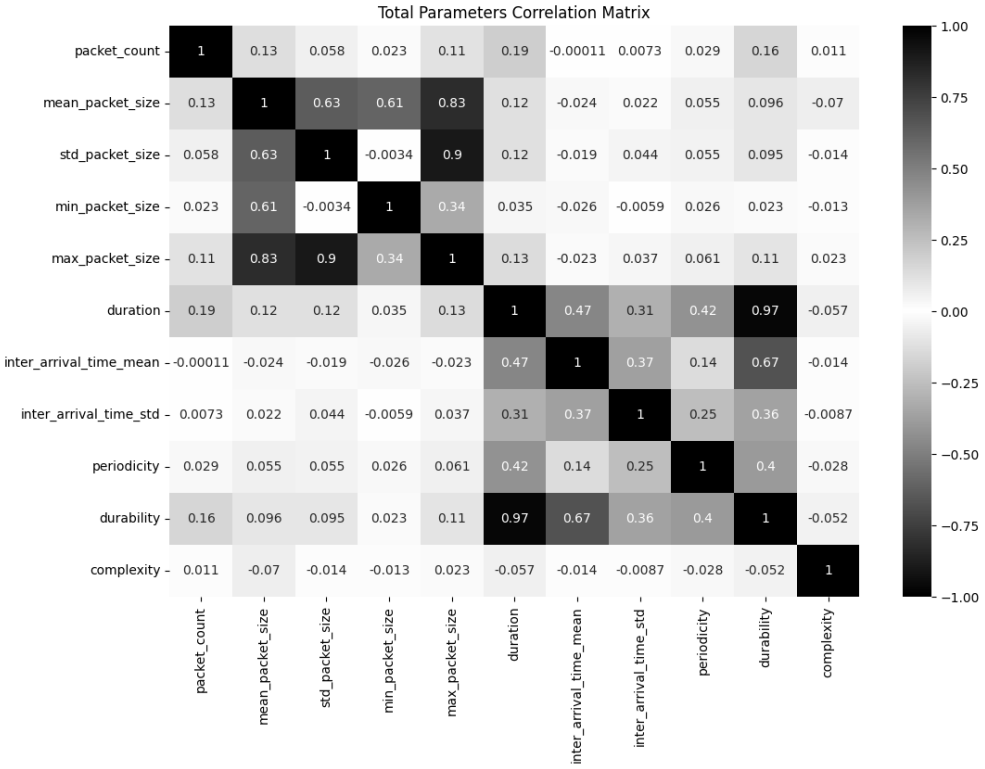

Рисунок 2.3 – Кореляційна матриця параметрів для задачі ідентифікації OT-протоколу

Для вирішення такої проблеми існує 3 різні підходи:

1) **Minority oversampling** – підхід передбачає додання значень до данних, таким чином аби зрівняти розподіл класів. Метод який використовується частіше всього – це SMOTE [25] (Synthetic Minority Oversampling Technique), який інтерполює значення відносно n сусідів, та генерує нові вхідні данні.
2) **Majority undersampling** – підхід, який випадковим чином прибирає вхідні данні з метою зрівняти кількість класів.
3) **Class weights** – підхід, що надає навчальним данним ваги на основі репрезентативності класу. Під час навчання математична модель має

спиратись на ці ваги та корегувати свої параметри сильніше, якщо клас мало представлений, та слабкіше , якщо клас переважає в датасеті.

В нашому підході буде використано метод SMOTE, оскільки навчальна вибірка є не великою, та навіть мінімальна втрата даних, може значимо зменшити точність

2.10 Методи оцінки якості математичних моделей

Результати тренування кожної з математичних моделей мають бути оцінені з метою об'єктивного визначення їх ефективності, здатності до узагальнення та придатності до практичного застосування. Якість моделі не може визначатися виключно рівнем точності на навчальній вибірці, оскільки це може вказувати на перенавчання — тобто адаптацію моделі до особливостей навчальних даних без здатності правильно обробляти нові спостереження.

Для визначення якості моделі буде використано матрицю помилок; метрики accuracy, recall, precision, f1-score; аналіз стійкості до шуму.

Матриця помилок (confusion matrix) дозволяє візуально представити результати класифікації у вигляді співвідношення між передбаченими та фактичними класами. Ця матриця відображає кількість правильно класифікованих прикладів (істинно позитивних та істинно негативних), а також типи помилок: хибно позитивні та хибно негативні передбачення. Аналіз структури помилок у цій матриці дозволяє виявити, які саме класи модель плутає між собою, і чи є певні типи пристроїв або протоколів більш уразливими до неправильної класифікації.

$$M = \begin{pmatrix} TP & FN \\ FP & TN \end{pmatrix} \tag{2.1}$$

На основі матриці помилок обчислюються класичні метрики ефективності класифікації —accuracy, recall, precision та F1-міра. Точність відображає загальну частку правильно класифікованих прикладів, однак може бути необ'єктивною в умовах дисбалансу класів. У таких випадках ключовими є метрики precision і recall, які характеризують здатність моделі уникати хибних спрацьовувань або пропусків.

F1-міра, як гармонійне середнє між precision і recall, дозволяє оцінити баланс між цими характеристиками і є особливо корисною для оцінки моделі в умовах обмежених або нерівномірно представлених даних.

$$accuracy = \frac{TP+TN}{TP+FN+TN+FP}, \quad (2.2)$$

$$precision = \frac{TP}{TP+FP}, \quad (2.3)$$

$$recall = \frac{TP}{TP+FN}, \quad (2.4)$$

$$F1 = \frac{2*precision*recall}{precision+recall}, \text{ де} \quad (2.5)$$

TP (True Positive) — це кількість випадків, коли модель правильно передбачила позитивний клас. FP (False Positive) означає кількість помилкових передбачень позитивного класу, коли об'єкт насправді належить до негативного. FN (False Negative) — це ситуації, коли модель не змогла виявити позитивний клас і класифікувала його як негативний. TN (True Negative) — це кількість випадків, коли модель правильно розпізнала об'єкти, що належать до негативного класу.

У зв’язку з специфікою застосування моделей, було прийнято рішення аналізувати стійкість моделей до шуму. Для цього буде використано Ґауссівський шум. Вхідні данні розділяються на класи, далі для кожного параметру моделі для кожного класу генерується шум з середнім рівним 0, та стандартним відхиленням рівним стандартному відхиленню параметру у відповідному класі. Шум додається до данних з певним впливом. Далі знову відбувається тестування моделі з визначенням стандартних метрик ефективності. Аналізуючи зміну метрик можна визначити стійкість моделі до шуму.

$$m = \{m_n \mid \forall\, n \in N\}, \text{ де} \quad (2.6)$$

N – множина рівнів впливу шуму на вхідні данні

n – конкретне значення впливу шуму

m_n – метрика ефективності після застосованого рівня шуму n

m – множина метрик ефективності після застосування усіх рівнів шуму

$$R_m = -\frac{\sum_{i=1}^{k}(n_i-\bar{n})(m_i-\bar{m})}{\sum_{i=1}^{k}(n_i-\bar{n})^2}, де \quad (2.7)$$

R – коефіцієнт впливу шуму. По суті R являє собою коефіцієнт наклону лінійної регресії та показує те, наскільки певна метрика, що тестується, спадає при збільшені шуму

2.11 Звітування

Одним з найважливіших етапів є звітування. Зібрані результати передаються далі для ручної або автоматизованої обробки. Було обрано два типи звітування: графічне зображення топології та JSON-файл із детальною інформацією про активи та з’єднання. Графічне представлення мережі дозволяє швидко оцінити загальну структуру. JSON-звіт, у свою чергу, надає повну структуровану інформацію про виявлені пристрої, їхні властивості, параметри з’єднань, а також поведінкові характеристики. Цей формат зручний для подальшої інтеграції з іншими системами.

ВИСНОВКИ ДО РОЗДІЛУ 2

Запропоновано модель відновлення мережевої конфігурації з використанням елементів поведінкового аналізу. Модель заснована на методі Passive Device Discovery, що є безпечним для ОТ-мереж. Для аналізу ролі ОТ-пристрою та застосованого ОТ-протоколу під час комунікації запропоновано використовувати поведінковий аналіз та математичні моделі для виокремлення поведінкових профілів для кожного з класів.

Описана методологія підготовки, навчання та тестування математичних моделей для вирішення проблеми класифікації. Підготовка данних передбачає прорахунок класичних та специфічних для ОТ метрик. Навчання відбувається на маркованих данних, частина з яких штучно зашумлена. Методи тестування математичних моделей основані на прорахунку таких метрик, як : accuracy, precision, recall, f1-score та зміна цих параметрів залежно від шуму вхідних данних.

Результатом виконання запропонованої програмної моделі є звіти у двох форматах: графічному та JSON

3 ЕКСПЕРЕМЕНТАЛЬНЕ ДОСЛІДЖЕННЯ МОДЕЛІ

У цьому розділі буде досліджено ефективність застосування математичних моделей ідентифікації OT-протоколів та відновлення типу пристрою з використанням поведінкового аналізу. Будуть порівнянні метрики точності та здатність моделі витримувати додатковий шум. Також буде продемонстровано результати загальної моделі на реальних данних.

3.1 Огляд вхідних данних

Навчання математичних моделей та огляд результатів загальної моделі буде проводитись на декількох датасетах:

Датасет 1 [20] – S4x15 ICS Village CTF PCAP Collection, збірка PCAP-файлів, отриманих під час Capture-the-Flag змагання на конференції S4x15. Кожен файл демонструє специфічні функції OT-протоколів у симульованому середовищі промислової автоматизації. Разом з цим надається детальна архітектура мережі з визначеними ролями

Датасет 2 [2] – 4SICS Geek Lounge ICS PCAP Collection, збірка PCAP-файлів, захоплених у рамках конференції 4SICS (тепер CS3Sthlm) в ICS-лабораторії "Geek Lounge". У цій лабораторії було представлено реальне OT-середовище з PLC, RTU, серверами та мережевим обладнанням, що дозволяло учасникам взаємодіяти з промисловими пристроями в режимі реального часу. Також надається відповідність IP адреси до характеристик приладу

Датасет 3 [22] – CIC Modbus Dataset 2023, набір PCAP-файлів та журналів атак, згенерованих у симульованій мережі підстанції, орієнтованій на протокол Modbus/TCP. Датасет поділений на два піднабори — з атакуючим трафіком і з легітимним (безпечним) трафіком. Надається відповідність IP адреси до типу приладу

Датасет 4 [3] – збірка pcap файлів, для деяких OT-протоколів, кожен pcap присвячений деякій функції яку може виконувати той чи інший протокол.

Датасет 5 [16] – дамп трафіку з Modbus/TCP та Siemens S7comm протоколами з ICS CTF проведеного в Китаї.

Умовно набір датасетів можна поділити на дві частини: ті, які містять інформацію про ролі активів (Датасети 1-3), та ті, що ні (Датасети 4-5). Таким чином модель типізації OT-пристроїв будуть тренуватись на першій групі датасетів, а модель ідентифікації протоколів на другій.

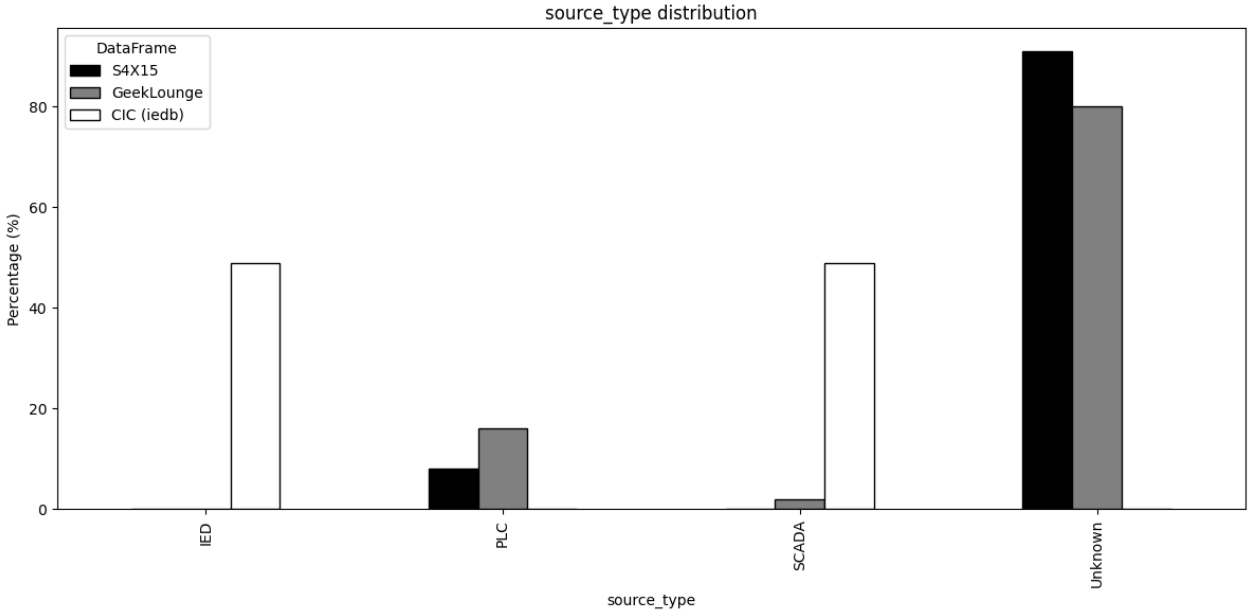

Рисунок 3.1 – Розподіл типу відправника в датасетах 1-3

Зчитаємо наші датасети та подивимось на їх характеристики. Як можна побачити з графіків 3.1 – 3.3, в датасетах присутні 4 типи OT-пристроїв: SCADA, IED, PLC, Unknown. Важливо відмітити, що навіть за наявності інформації про тип не OT-активів, їх типи замінені на Unknown. Малюнок 3.4 демонструє розподіл OT протоколів в датасетах. Можна побачити 3 найбільші групи, це S7comm, Modbus, - це найбільш поширені протоколи в OT-мережах, та Unknown – клас, який фактично є збіркою будь-яких інших мережевих протоколів, що не відносяться до OT.

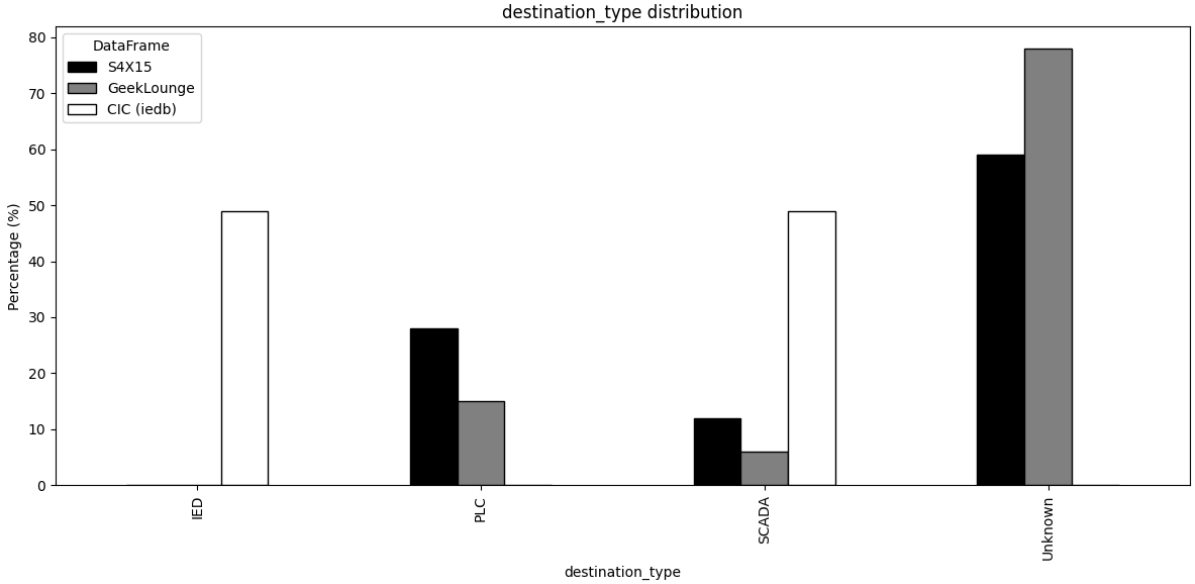

Рисунок 3.2 – Розподіл типу отримувача в датасетах 1-3

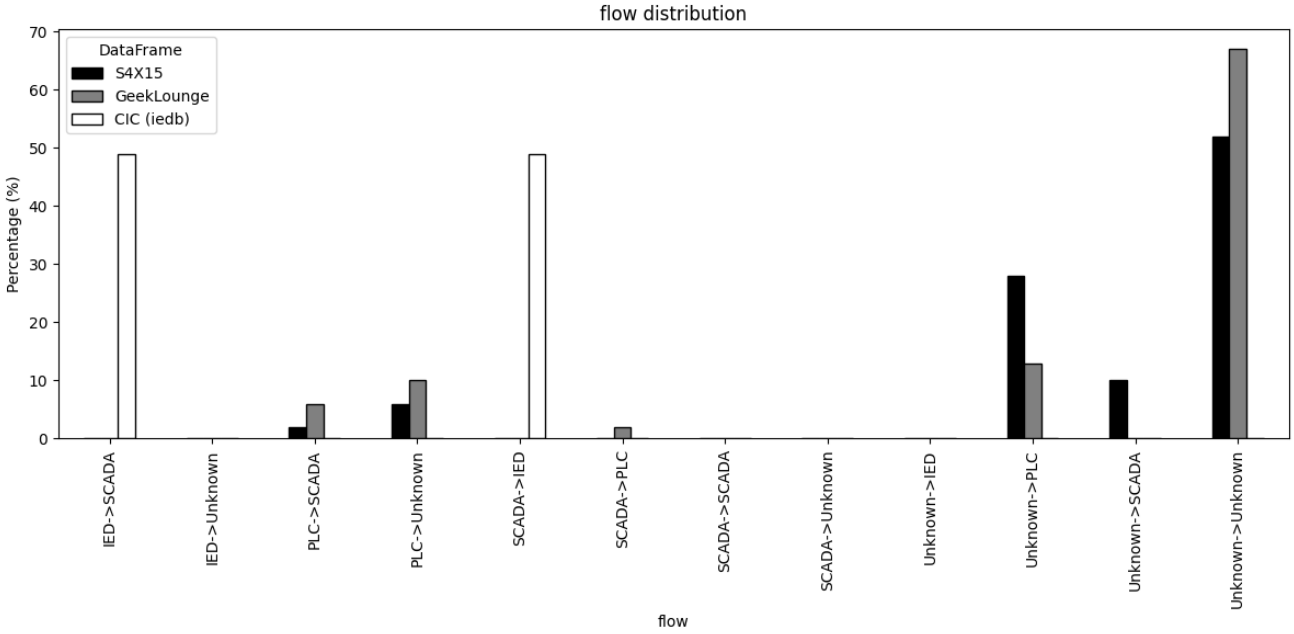

Рисунок 3.3 – Розподіл пар комунікацій в датасетах 1-3

3.2 Навчання математичних моделей

Обробивши данні відповідно до методів описаних в розділі 2, розділимо їх на частини для тренування та навчання за пропорцією 70/30, при цьому класи в

навчальних данних урівнюються з використанням SMOTE. Також 30% тренувальних данних було штучно зашумлено з рівнем впливу в 10% - що відповідає середньому шуму. Було проаналізовано наступні математичні моделі: Linear Regression, Random Forest, MLP, KNN, Gradient Boosting, результат зведені до таблиці 3.1 та рисунків 3.5-3.6.

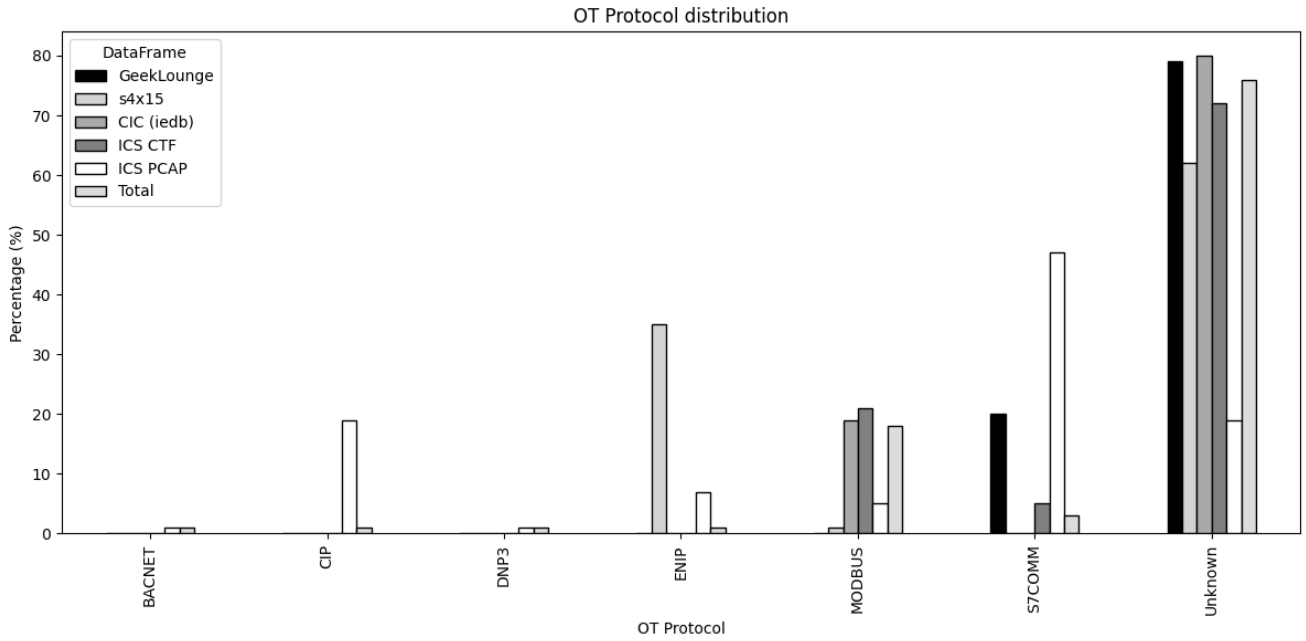

Рисунок 3.4 – Розподіл OT-протоколів в датасетах 1-5

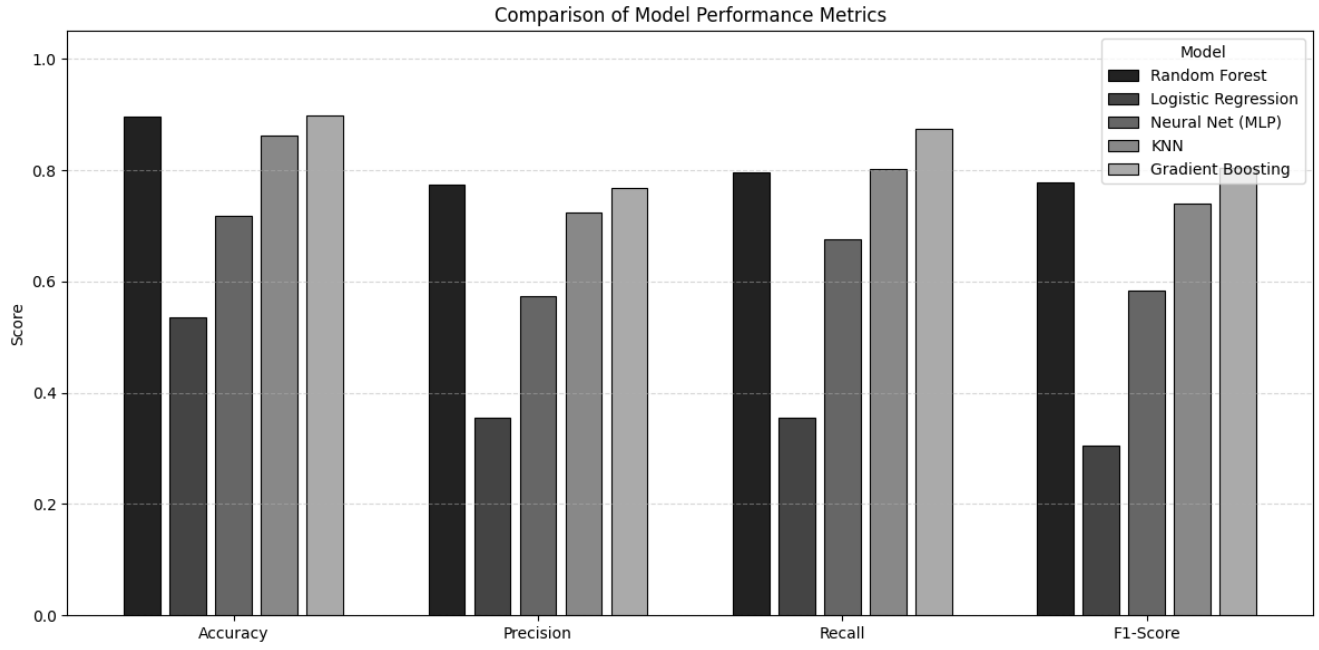

Рисунок 3.5 – Порівняння метрик точності для моделей ідентифікації протоколів

Таблиця 3.1 – метрики ефективності моделі

Модель	Задача	Accuracy	Precision	Recall	F1-score
Linear Regression	Ід. проток.	0.53	0.36	0.35	0.30
	Тип. прист.	0.91	0.89	0.9	0.87
Random Forest	Ід. проток	0.89	0.77	0.79	0.77
	Тип. прист.	0.99	0.99	0.99	0.99
MLP	Ід. проток	0.71	0.57	0.67	0.58
	Тип. прист.	0.99	0.99	0.99	0.99
KNN	Ід. проток	0.86	0.72	0.8	0.74
	Тип. прист.	0.99	0.99	0.99	0.99
Gradient Boosting	Ід. проток	0.9	0.77	0.87	0.8
	Тип. прист.	0.99	0.99	0.99	0.99

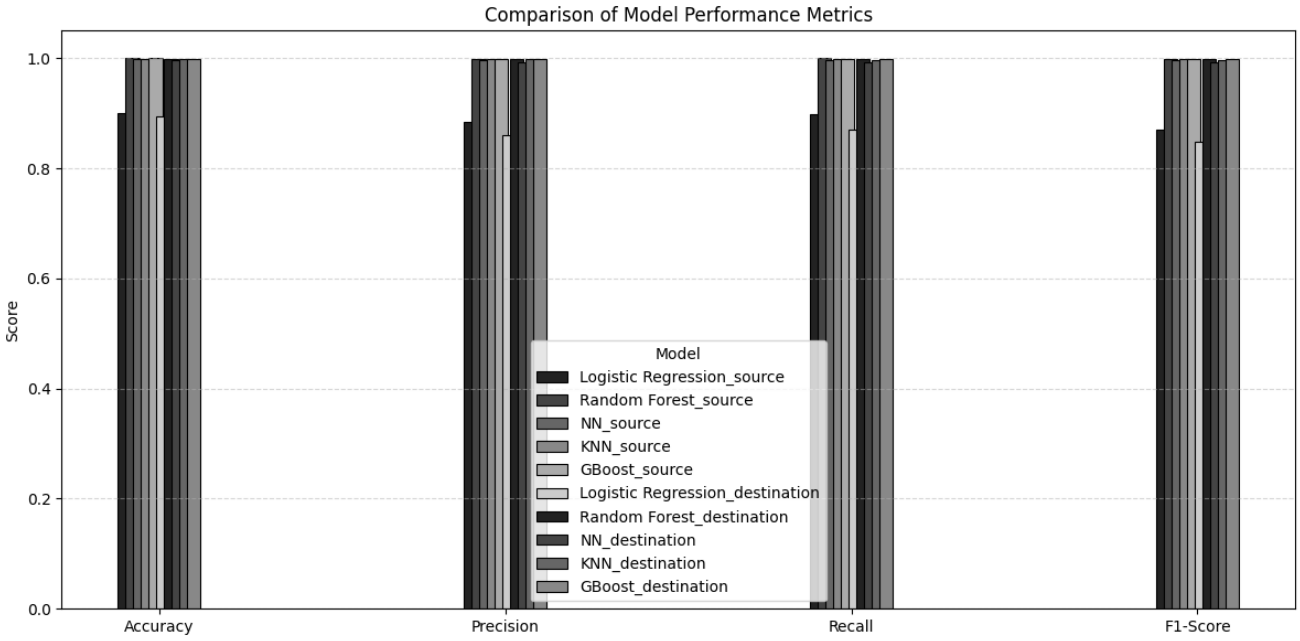

Рисунок 3.6 – Порівняння результативності моделей для вирішення задача класифікації ролі пристрою

Також було проаналізовано стійкість моделей до шуму. В якості ключових точок для аналізу було обрано 10 поділок на діапазоні від 0 до 1. Результати зведені до таблиці 3.2 та додатків Б та Г.

Таблиця 3.2 - Результати тестування стійкості до шуму

Модель	Задача	Діапазон впливу шуму	Min acc.	Max acc..	R acc.	Min prec.	Max prec.	R prec.	Min rec.	Max rec.	R rec.	Min F1	Max F1	R F1
Linear Regression	Ід. проток.	0-1	0.55	0.56	0.001	0.36	0.42	0.01	0.35	0.41	0.004	0.3	0.33	0.003
	Тип. прист.	0-1	0.8	0.8	0	0.78	0.78	0	0.59	0.59	0	0.58	0.58	0
Random Forest	Ід. проток	0-1	0.84	0.9	0.04	0.64	0.79	0.11	0.64	0.78	0.13	0.64	0.77	0.12
	Тип. прист.	0-1	0.63	0.99	0.21	0.68	0.99	0.19	0.74	0.99	0.15	0.62	0.99	0.2
MLP	Ід. проток	0-1	0.63	0.66	0.01	0.53	0.57	0.02	0.58	0.69	0.05	0.51	0.56	0.02
	Тип. прист.	0-1	0.98	0.99	0.01	0.98	0.99	0.01	0.98	0.99	0.01	0.98	0.99	0.01
KNN	Ід. проток	0-1	0.83	0.85	0.01	0.65	0.68	0.01	0.78	0.81	0.01	0.68	0.71	0.01
	Тип. прист.	0-1	0.99	0.99	0	0.99	0.99	0	0.99	0.99	0	0.99	0.99	0
Gradient Boosting	Ід. проток	0-1	0.81	0.88	0.05	0.61	0.75	0.09	0.67	0.86	0.14	0.64	0.78	0.11
	Тип. прист.	0-1	0.63	0.99	0.21	0.68	0.99	0.19	0.74	0.99	0.15	0.62	0.99	0.2

Враховуючі усі наведені вище метрики, було прийнято рішення виділити наступні моделі як найкращі:

- **Задача ідентифікації OT-протоколів**: Random Forest з точністю 90% та впливом шуму в 0.04.
- **Задача визначення ролі OT-пристрою:** KNN з точністю 99% та впливом шуму 0

3.3 Результати моделі

Протестуємо модель на реальних данних та подивимось на результати. Для цього було обрано датасет 1, та pcap-файл Advantech

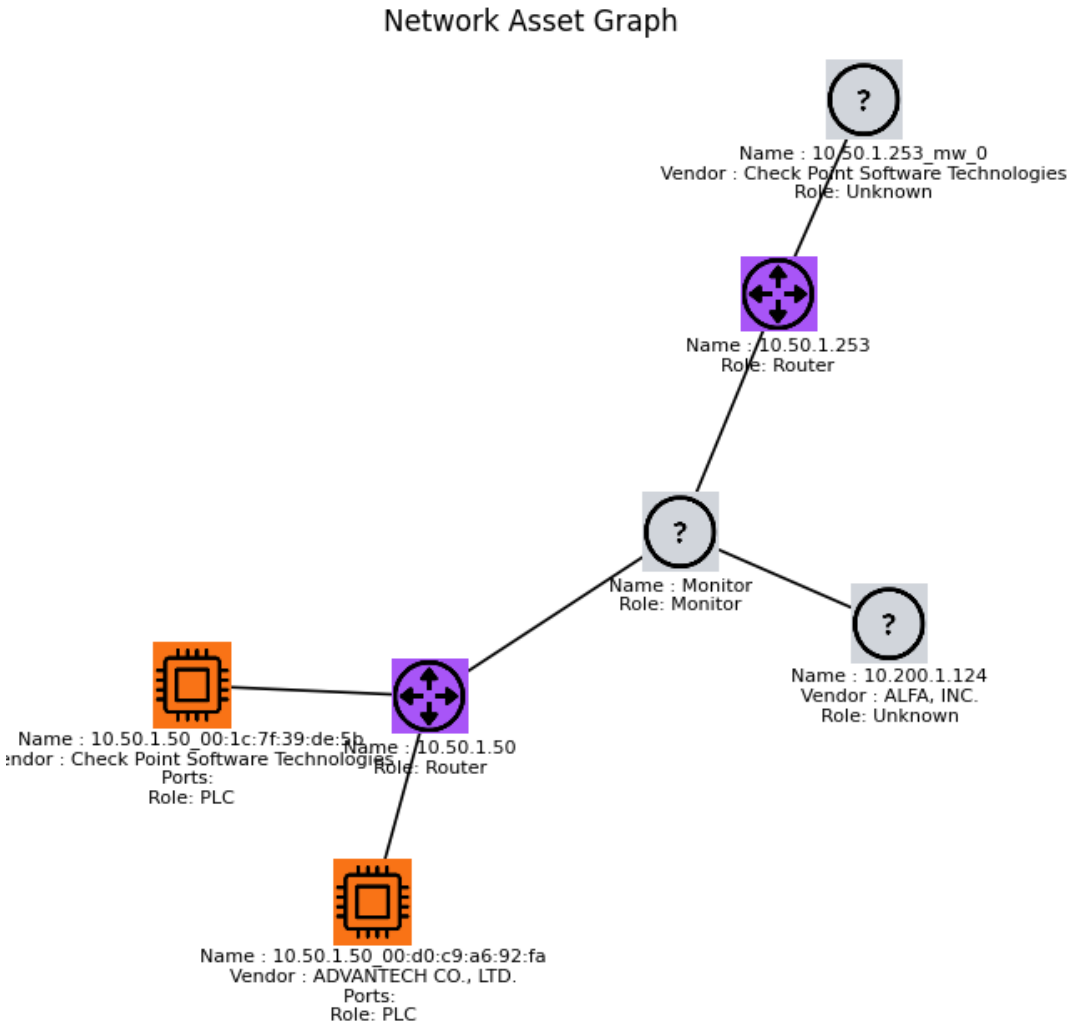

Рисунок 3.7 – Приклад виводу програмної моделі на датасеті 1 (Advantech.pcap)

ВИСНОВКИ ДО РОЗДІЛУ 3

У цьому розділі було проведено дослідження ефективності поведінкового аналізу з використанням математичних моделей для вирішення задач ідентифікації ОТ-протоколів і типізації ОТ-активів. Запропонований підхід базується на аналізі характеристик мережевого трафіку без використання сигнатур чи специфічних знань про структуру протоколу, що робить його придатним до роботи в умовах обмеженої інформації.

Для задачі розпізнавання ОТ-протоколів найкращий результат продемонструвала модель Random Forest, яка досягла точності 90% при впливі шуму 0.04. У свою чергу, для задачі визначення ролі ОТ-пристроїв найвищу ефективність показала модель K-Nearest Neighbors (KNN) з точністю 99% та нульовим впливом шуму.

Отримані результати підтверджують доцільність використання машинного навчання в задачах аналізу ОТ-систем і демонструють можливість точного відновлення як мережевої активності, так і функціонального призначення пристроїв навіть у зашумлених середовищах

ВИСНОВКИ

У межах цієї роботи було виконано комплексне дослідження підходів до автоматизованого аналізу OT-мереж із використанням математичних моделей та поведінкового аналізу. На основі вивчення сучасного стану задач класифікації OT-протоколів і відновлення мережевої конфігурації було виявлено основні труднощі, пов’язані з особливостями OT-середовища — зокрема, обмеженнями щодо активного втручання, використанням застарілих технологій та нестабільністю існуючих методів у разі шифрування чи кастомізації протоколів.

Проаналізовано існуючі підходи до класифікації OT-протоколів і показано, що більшість із них базуються на аналізі вмісту пакетів, що не завжди можливо в реальних умовах. У зв’язку з цим обґрунтовано доцільність застосування альтернативного підходу — поведінкового аналізу, який спирається на непрямі характеристики трафіку.

Також досліджено питання відновлення ролі пристроїв у мережі. Показано, що пасивне виявлення, як безпечна альтернатива активному скануванню, потребує нових підходів для точного визначення функціонального призначення OT-пристроїв без попереднього знання інфраструктури.

В результаті роботи Розроблено модель класифікації OT-протоколів, що ґрунтується на поведінкових ознаках мережевого трафіку. Найвищу ефективність продемонструвала модель Random Forest з точністю 90% при впливі шуму 0.04. Розроблено метод типізації OT-пристроїв на основі аналізу пасивно зібраних атрибутів. Найкращі результати показала модель K-Nearest Neighbors (KNN**)** з точністю **99%** і стійкістю до зашумленості. Запропоновано загальну методику відновлення мережевої конфігурації, яка поєднує в собі поведінковий аналіз, машинне навчання та пасивний збір даних. Метод не вимагає попередніх знань про інфраструктуру, що робить його гнучким і адаптивним.

Методика включає етапи підготовки, навчання й тестування моделей з використанням OT-специфічних метрик, а результати подаються як у графічному, так і в структурованому (JSON) форматах.

Загалом, результати роботи підтверджують ефективність запропонованих підходів для безпечного, точного та масштабованого аналізу OT-мереж, навіть за умов неповних або зашумлених вхідних даних. Отримані методи можуть бути використані для підвищення рівня безпеки, інвентаризації та моніторингу в критичних інфраструктурах.

.

ПЕРЕЛІК ВИКОРИСТАНИХ ДЖЕРЕЛ

2. 4SICS. (без дати). Capture files from 4SICS Geek Lounge. *Netresec*. Отримано з https://www.netresec.com/?page=PCAP4SICS
3. automayt. (без дати). *ICS-pcap.* Отримано з https://github.com/automayt/ICS-pcap
4. Chakraborty, I., Kelley, B. M., & Gallagher, B. (2021). Industrial control system device classification using network traffic features and neural network embeddings. *ScienceDirect*. Отримано з https://www.sciencedirect.com/science/article/pii/S2590005621000291?via%3Dihub#dfn3
5. Fortinet. (без дати). *Deep Packet Inspection (DPI) Definition.* Отримано з https://www.fortinet.com/resources/cyberglossary/dpi-deep-packet-inspection
6. Gyebnár, G. (2023). *The State of OT Security Infographic.* Отримано з https://blackcell.io/the-state-of-ot-security-infographic
7. Holasova, E., Fujdiak, R., & Misurec, J. (May 2024 p.). Comparative Analysis of Classification Methods and Suitable Datasets for Protocol Recognition in Operational Technologies. *Algorithms*. Отримано з https://www.researchgate.net/publication/380553967_Comparative_Analysis_of_Classification_Methods_and_Suitable_Datasets_for_Protocol_Recognition_in_Operational_Technologies
8. IEEE. (без дати). *OUI-Vendor Dataset.* Отримано з https://standards-oui.ieee.org/
9. Jeon, S., Yun, J.-H., Choi, S., & Kim, W.-N. (2016). *Passive Fingerprinting of SCADA in Critical Infrastructure Network without Deep Packet Inspection.*

10. Jones, K., & Wedgbury, A. (September 2015 p.). Automated Asset Discovery in Industrial Control Systems - Exploring the Problem. *3rd International Symposium for ICS & SCADA Cyber Security Research 2015*.

11. Knapp, E. D. (2024). *Industrial Network Security: Securing Critical Infrastructure Networks for Smart Grid, SCADA, and Other Industrial Control Systems.* Elsevier.

12. Koay, A. M., Ko, R. K., Hettema, H., & Radke, K. (2022). Machine learning in industrial control system (ICS) security: current landscape, opportunities and challenges. *SpringerNatureLink*. Отримано з https://link.springer.com/article/10.1007/s10844-022-00753-1

13. Lyon, G. “. (2009). *Nmap Network Scanning: The Official Nmap Project Guide to Network Discovery and Security Scanning.*

14. Mavrakis, C. (2015). *Passive Asset Discovery and Operating System Fingerprinting in Industrial Control System Networks.* Отримано з https://pure.tue.nl/ws/portalfiles/portal/46916656/840171-1.pdf

15. Natural language processing. (без дати). Отримано з https://en.wikipedia.org/wiki/Natural_language_processing

16. NewBee119. (без дати). *CTF ICS Traffic*. Отримано з https://github.com/NewBee119/ctf_ics_traffic

17. Niedermaier, M., Hanka, T., Plaga, S., Bodisco, A. v., & Merli, D. (без дати). *EFFICIENT PASSIVE ICS DEVICE DISCOVERY AND IDENTIFICATION BY MAC ADDRESS CORRELATION.* Отримано з https://arxiv.org/pdf/1904.04271

18. Niedermaier, M., Hanka, T., Plaga, S., Merli, D., & Bodisco, A. V. (2018). Efficient Passive ICS Device Discovery and Identification by MAC Address Correlation. Hamburg, Germany. Отримано з https://www.scienceopen.com/hosted-document?doi=10.14236/ewic/ICS2018.3

19. Rapid7. (без дати). *What is IT Asset Discovery?* Отримано з https://www.rapid7.com/fundamentals/what-is-it-asset-discovery/

20. (2015). *S4x15 ICS Village PCAP Files.* Netresec. Отримано з https://www.netresec.com/?page=DigitalBond_S4

21. Shi, J., Yu, X., & Liu, Z. (2021). Nowhere to Hide: A Novel Private Protocol Identification Algorithm. Отримано з https://onlinelibrary.wiley.com/doi/10.1155/2021/6672911

22. UNB. (2023). *CIC Modbus dataset.* Отримано з https://www.unb.ca/cic/datasets/modbus-2023.html

23. Yu, C., Zhang, Z., & Gao, M. (2022). An ICS Traffic Classification Based on Industrial Control Protocol Keyword Feature Extraction Algorithm. *MDPI*. Отримано з https://www.mdpi.com/2076-3417/12/21/11193

24. Zhai, L., Zheng, Q., Zhang, X., Hu, H., Yin, W., Zeng, Y., & Wu, T. (2021). Identification of Private ICS Protocols Based on Raw Traffic. *Symmetry*. Отримано з https://www.researchgate.net/publication/354757962_Identification_of_Private_ICS_Protocols_Based_on_Raw_Traffic

25. Sole Galli (2023). Overcoming Class Imbalance with SMOTE: How to Tackle Imbalanced Datasets in Machine Learning. Отримано з https://www.blog.trainindata.com/overcoming-class-imbalance-with-smote/

ДОДАТОК А АНАЛІЗ ВХІДНИХ ДАННИХ ДЛЯ МОДЕЛІ КЛАСИФІКАЦІЇ ОТ ПРОТОКОЛІВ

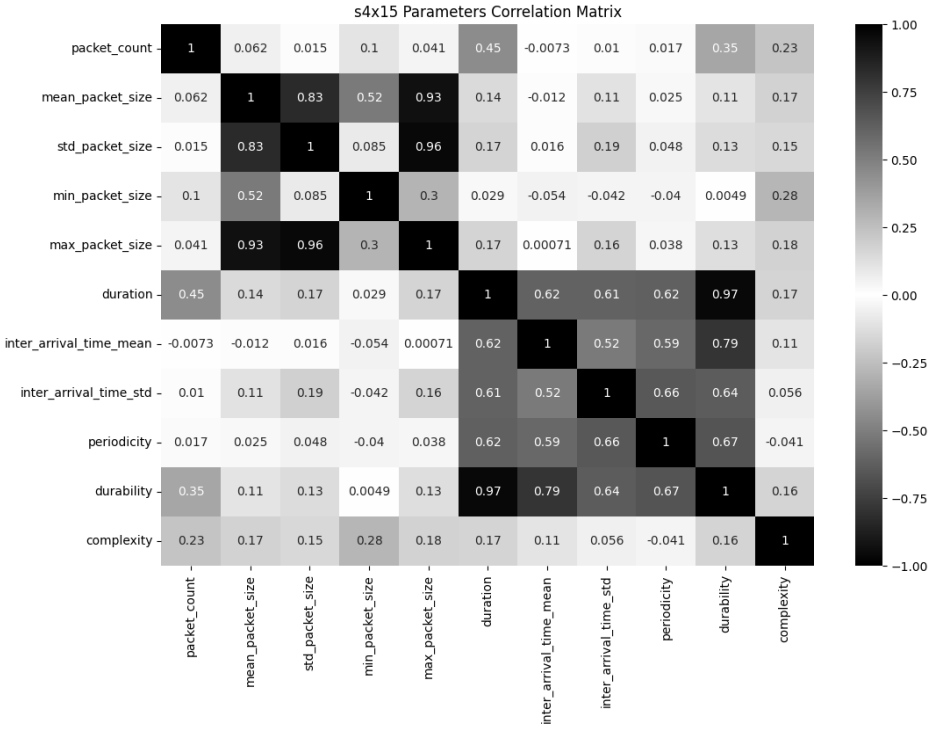

Рисунок А.1 – Кореляційна матриця параметрів для датасету 1

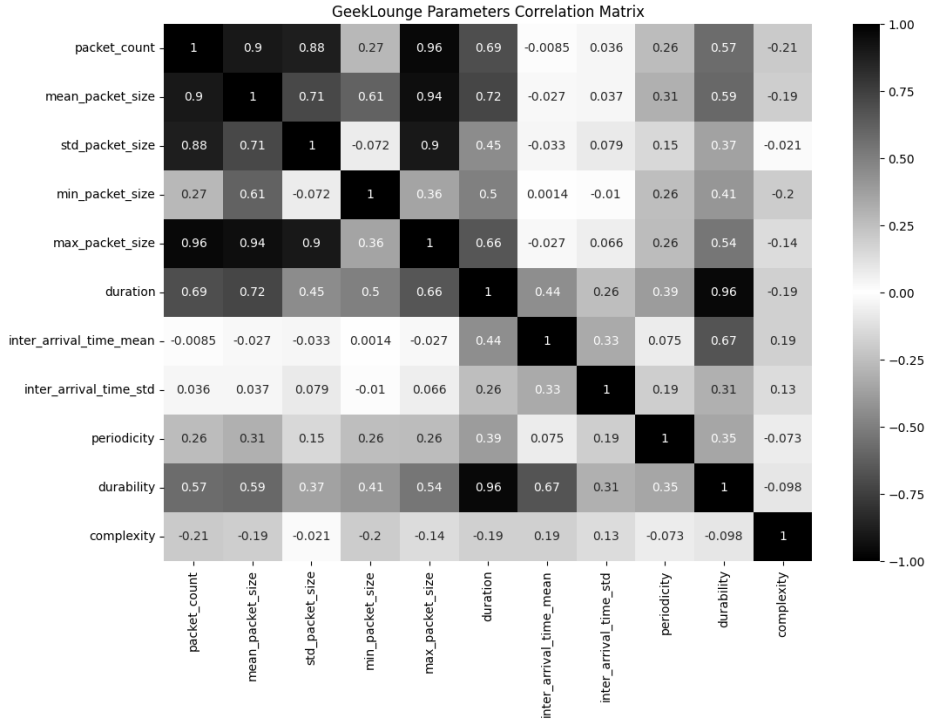

Рисунок А.2 – Кореляційна матриця параметрів для датасету 2

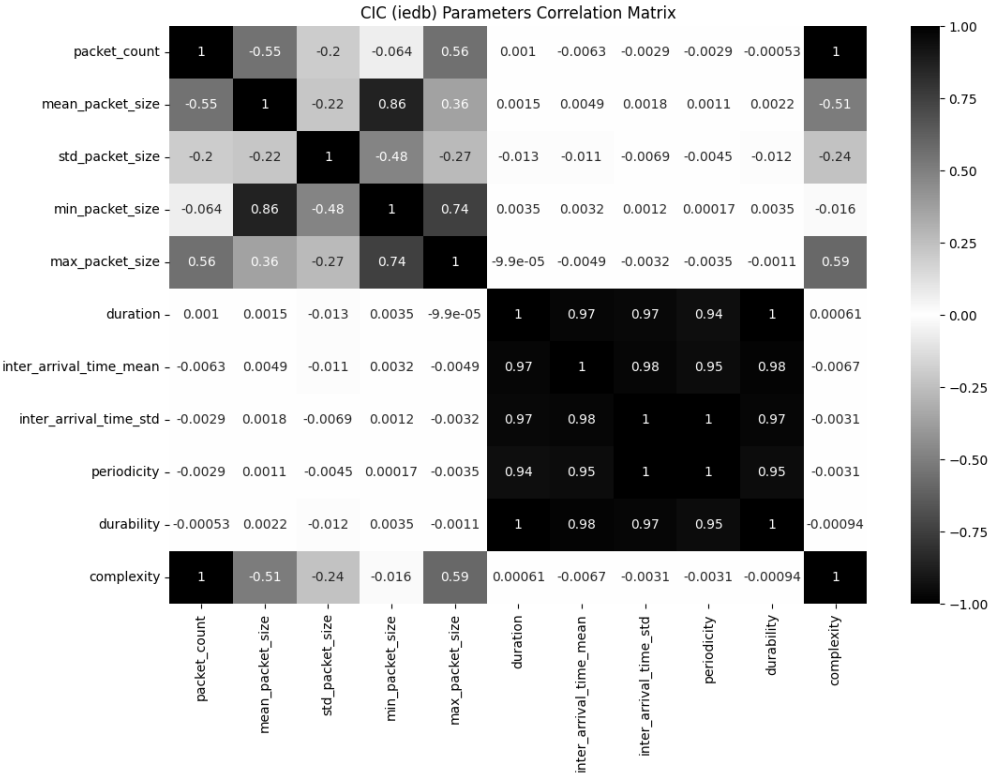

Рисунок А.3 – Кореляційна матриця параметрів датасету 3

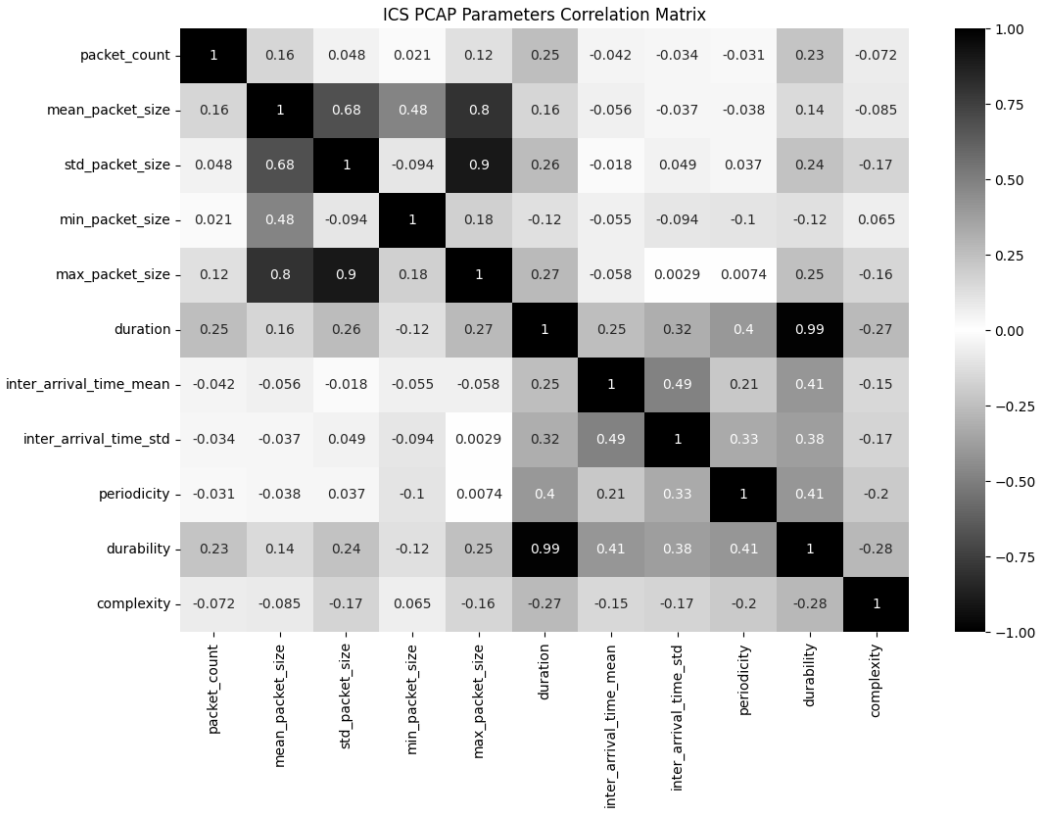

Рисунок А.4 – Кореляційна матриця параметрів датасету 4

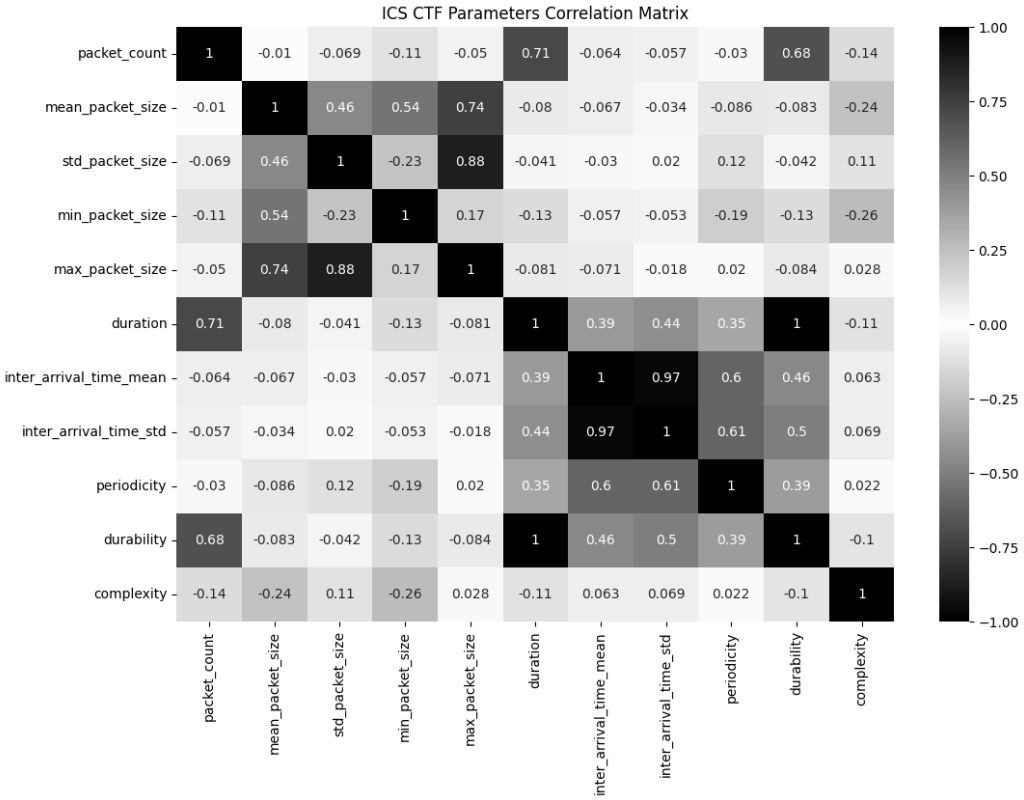

Рисунок А.5 – Кореляційна матриця параметрів датасету 5

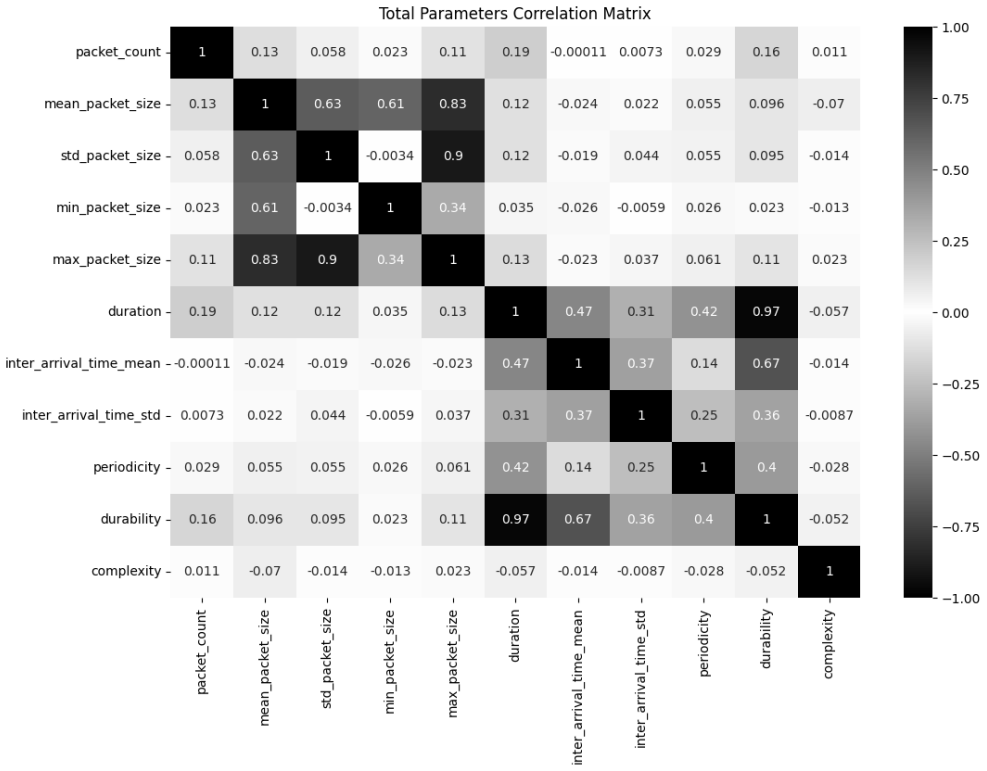

Рисунок А.6 – Кореляційна матриця параметрів для датасетів 1-5 (сумарно)

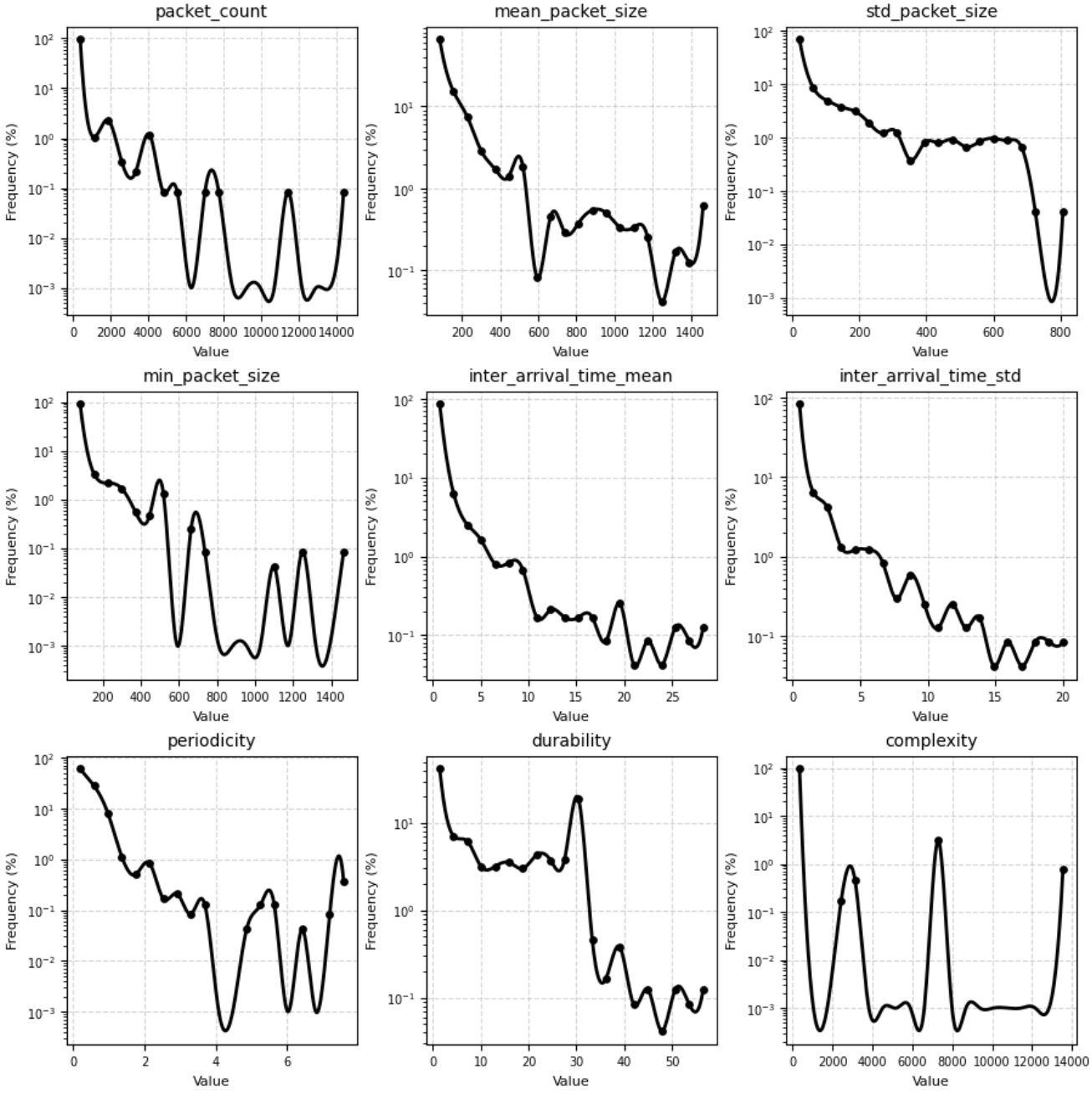

Рисунок А.7 – Розподіл параметрів

ДОДАТОК Б РЕЗУЛЬТАТИ ТРЕНУВАННЯ МОДЕЛЕЙ ДЛЯ ІДЕНТИФІКАЦІЇ ОТ-ПРОТОКОЛІВ

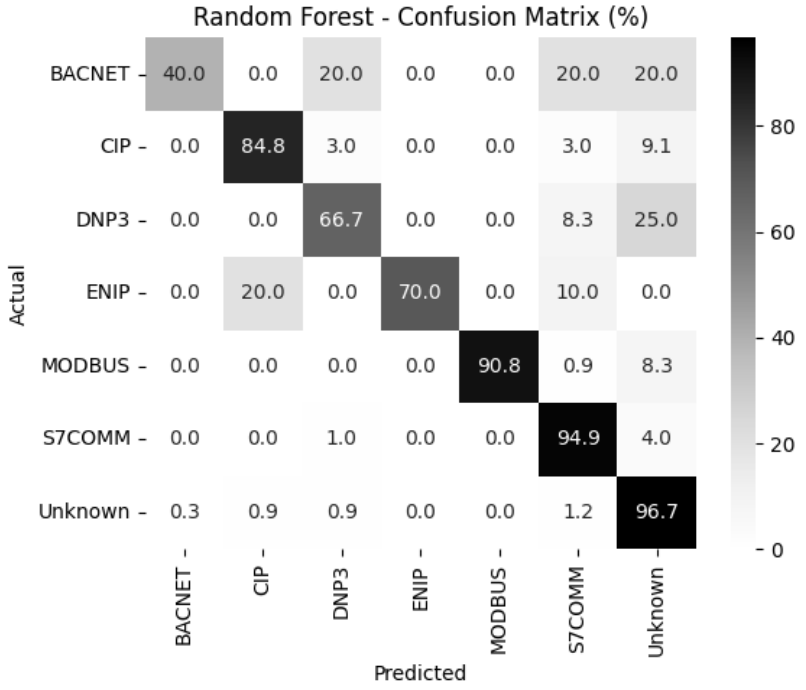

Рисунок Б.1 – Матриця помилок для натренованої моделі Random Forest

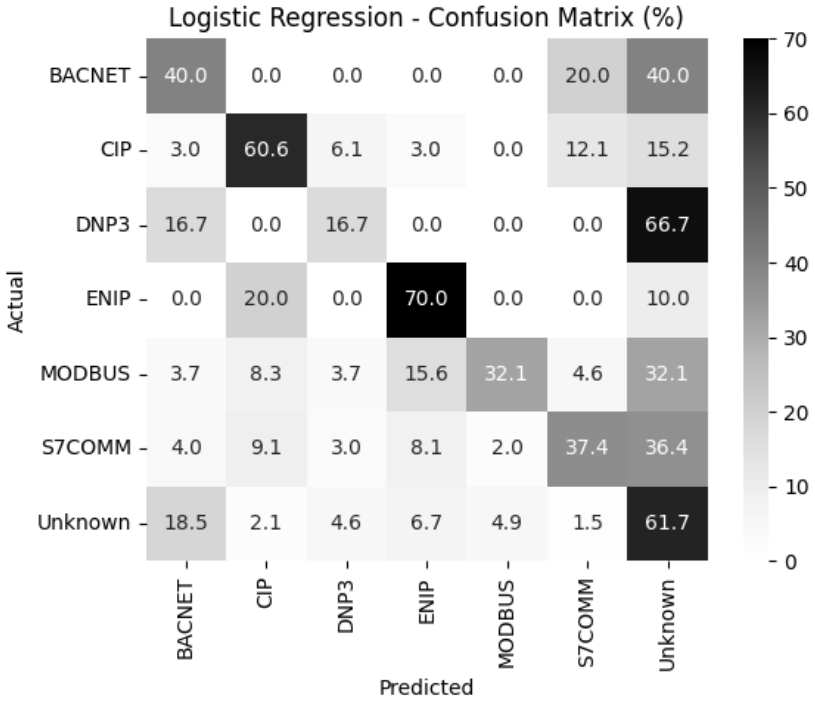

Рисунок Б.2 – Матриця помилок для натренованої моделі Logistic Regression

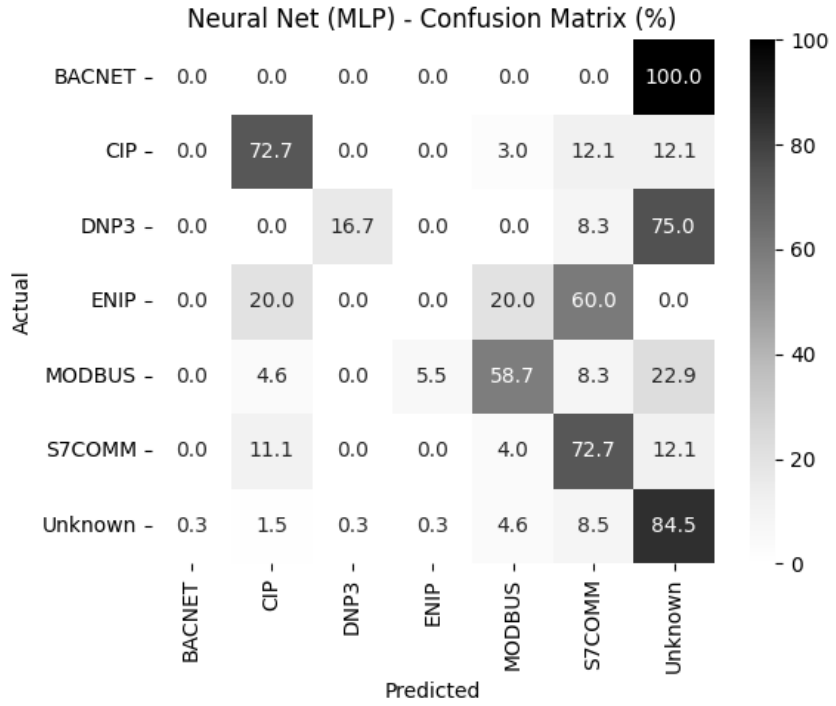

Рисунок Б.3 – Матриця помилок для натренованої моделі MLP

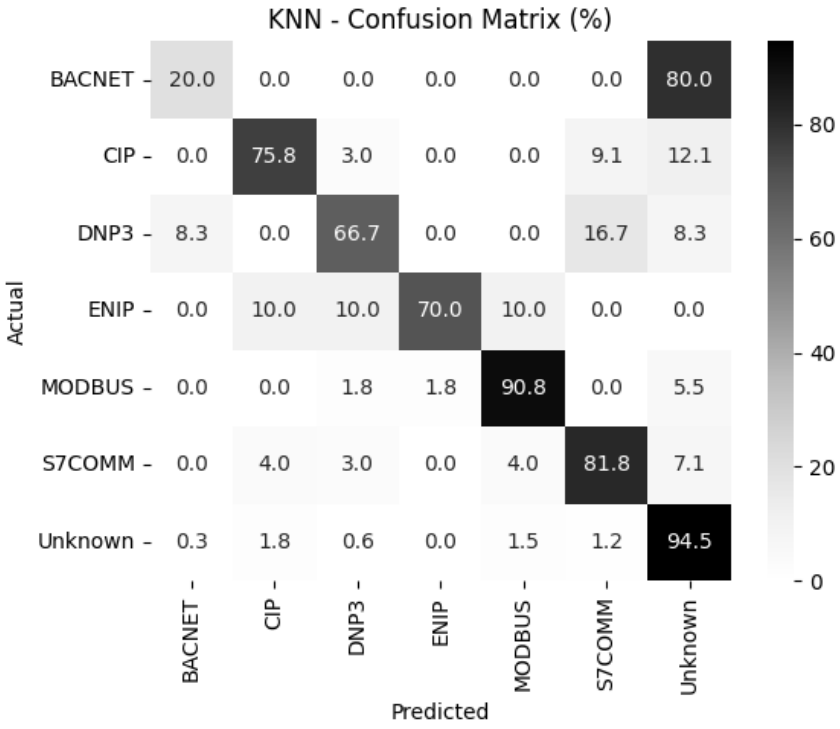

Рисунок Б.4 – Матриця помилок для натренованої моделі KNN

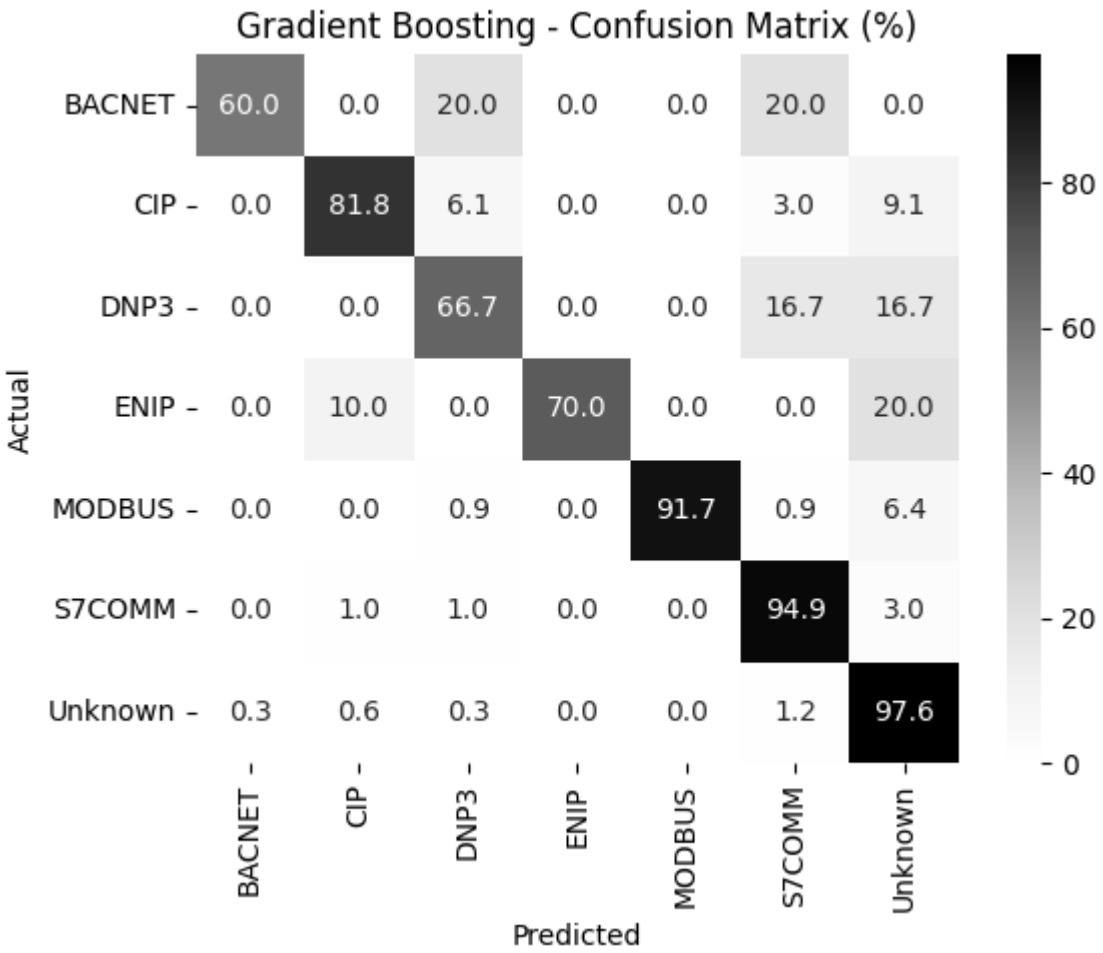

Рисунок Б.5 – Матриця помилок для натренованої моделі Gradient Boosting

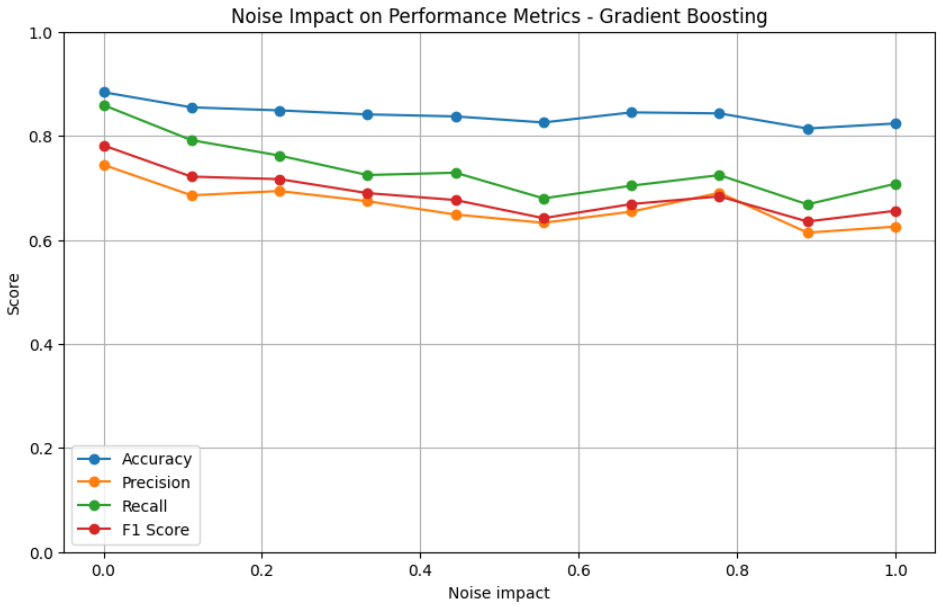

Рисунок Б.6 – тестування на вплив шуму для моделі Gradient Boosting

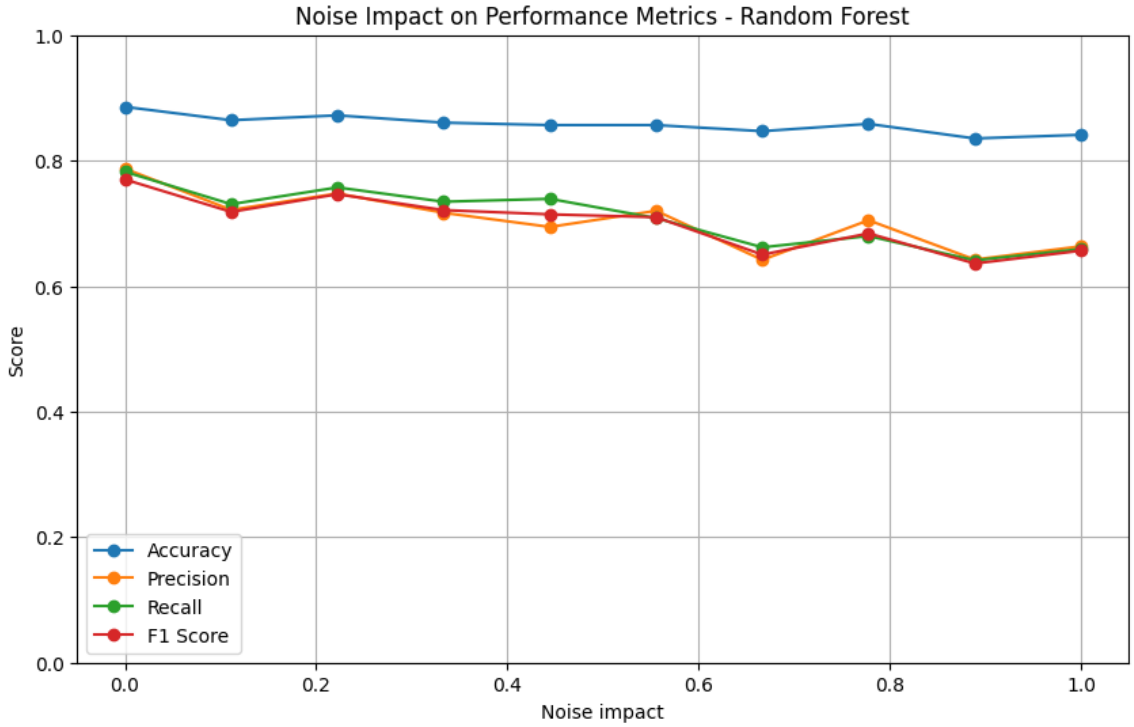

Рисунок Б.7 – тестування на вплив шуму для моделі Random Forest

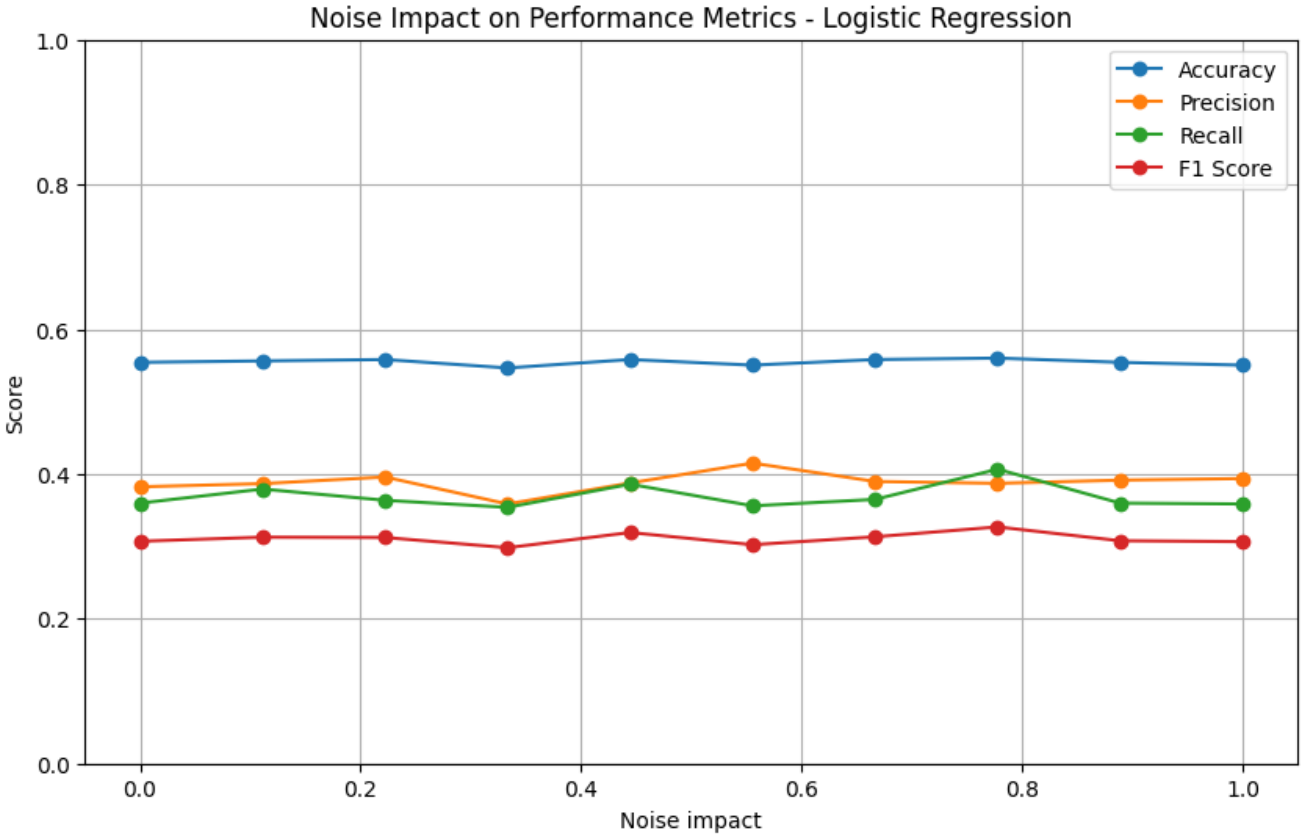

Рисунок Б.8 – тестування на вплив шуму для моделі Logistic Regression

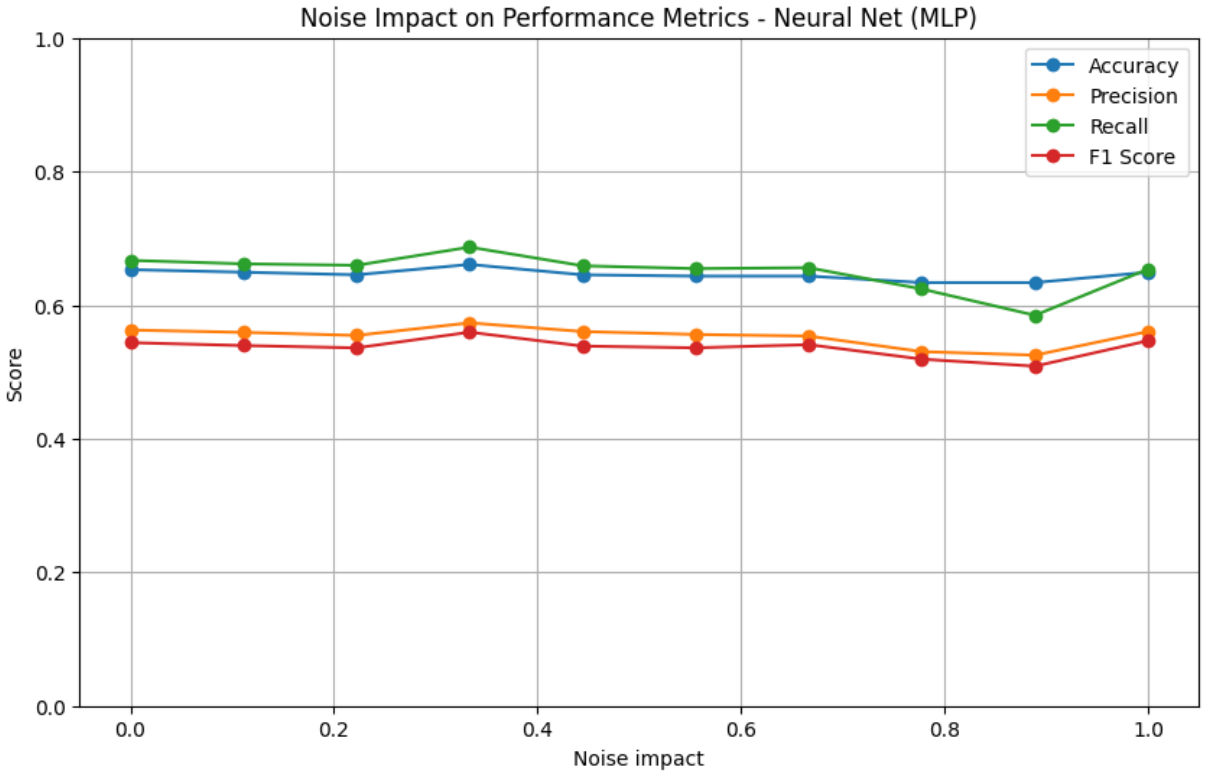

Рисунок Б.9 – тестування на вплив шуму для моделі MLP

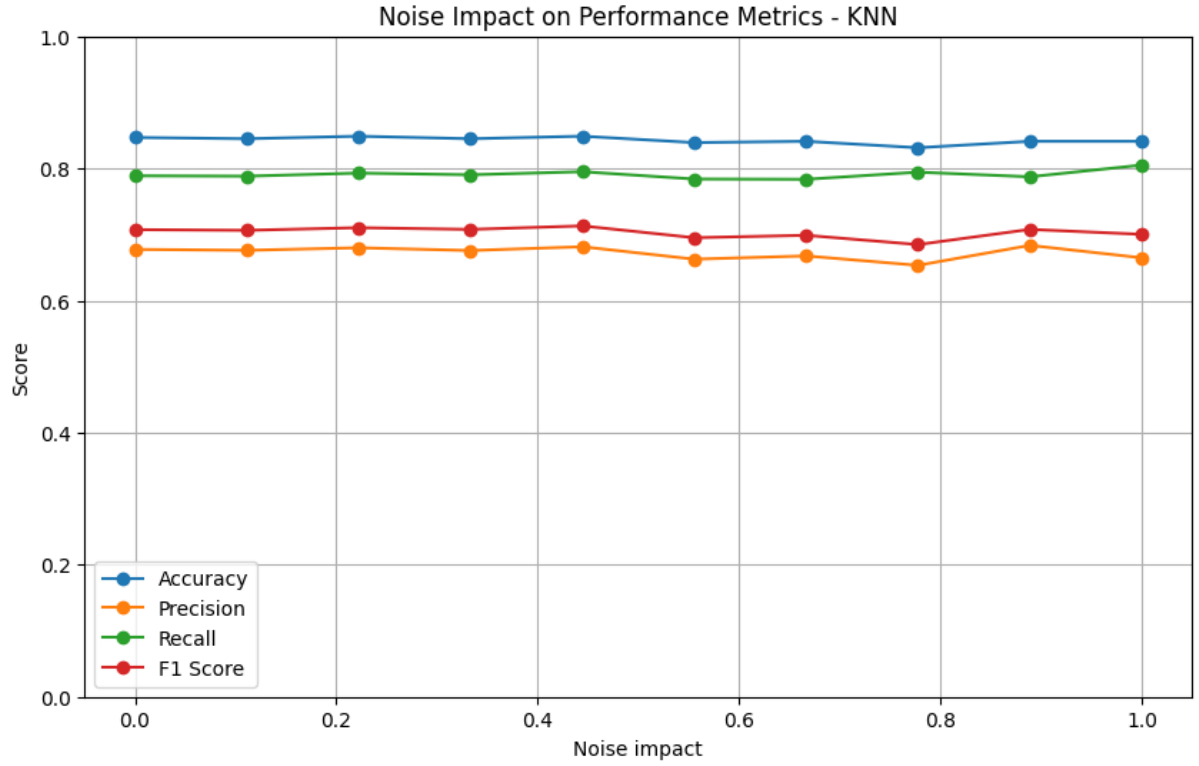

Рисунок Б.10 – тестування на вплив шуму для моделі KNN

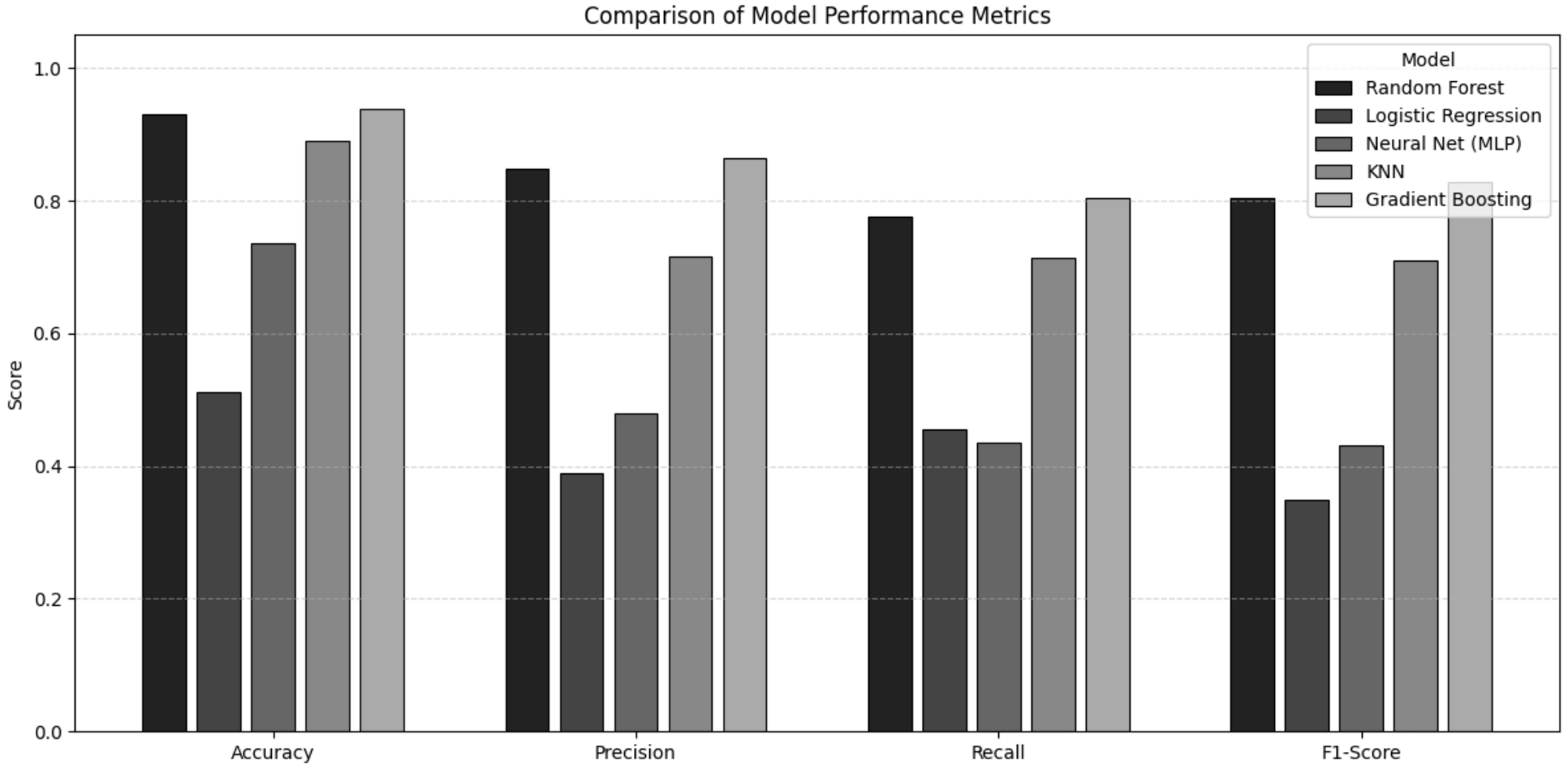

Рисунок Б.6 – Порівняння метрик точності для моделей ідентифікації протоколів

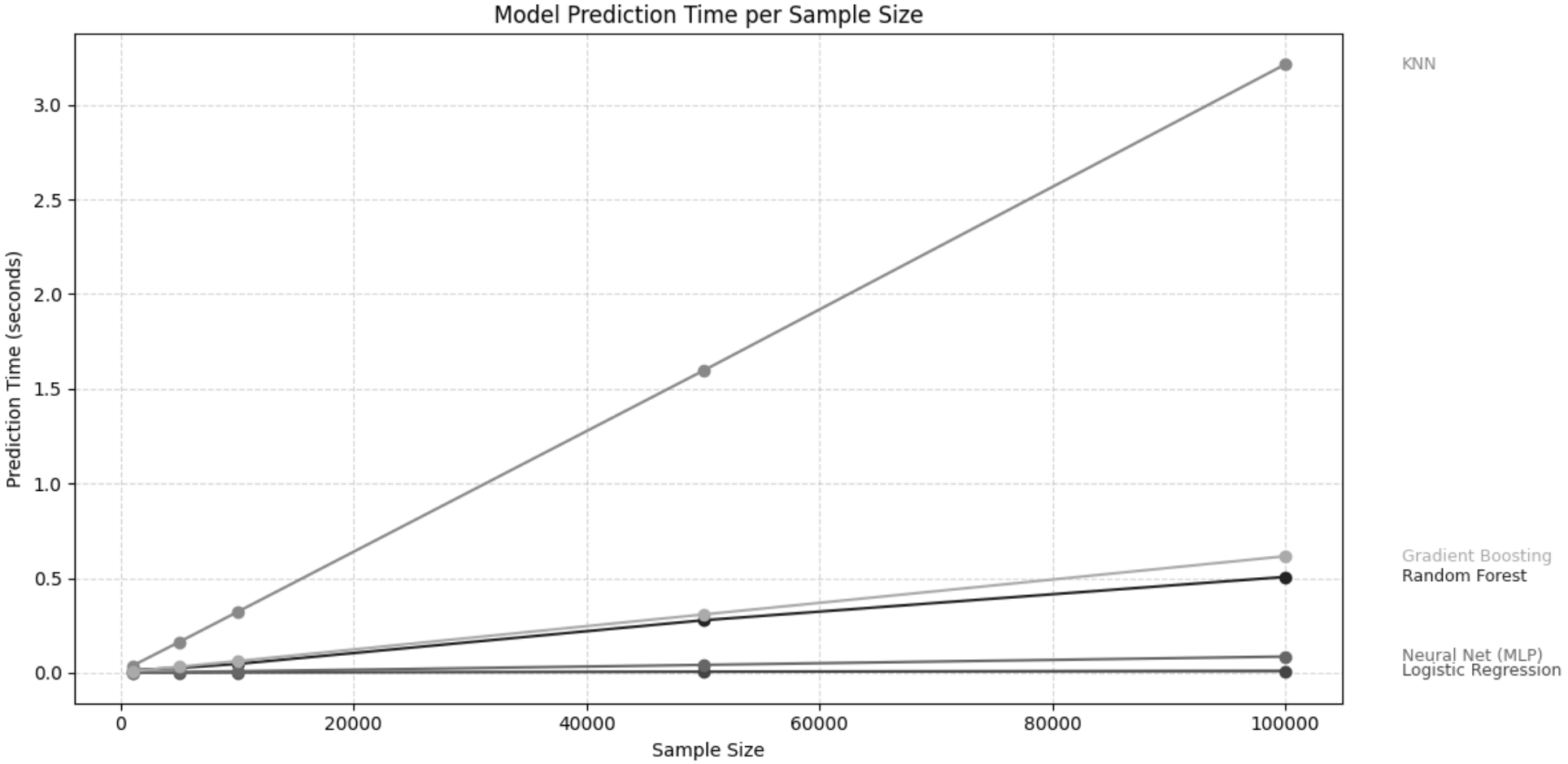

Рисунок Б.7 – Залежність часу від розміру вибірки для кожної з моделей

ДОДАТОК В АНАЛІЗ ВХІДНИХ ДАННИХ ДЛЯ МОДЕЛЕЙ КЛАСИФІКАЦІЇ РОЛІ ОТ-ПРИСТРОЮ ТА РАНГУВАННЯ ОТ-З'ЄДНАННЯ

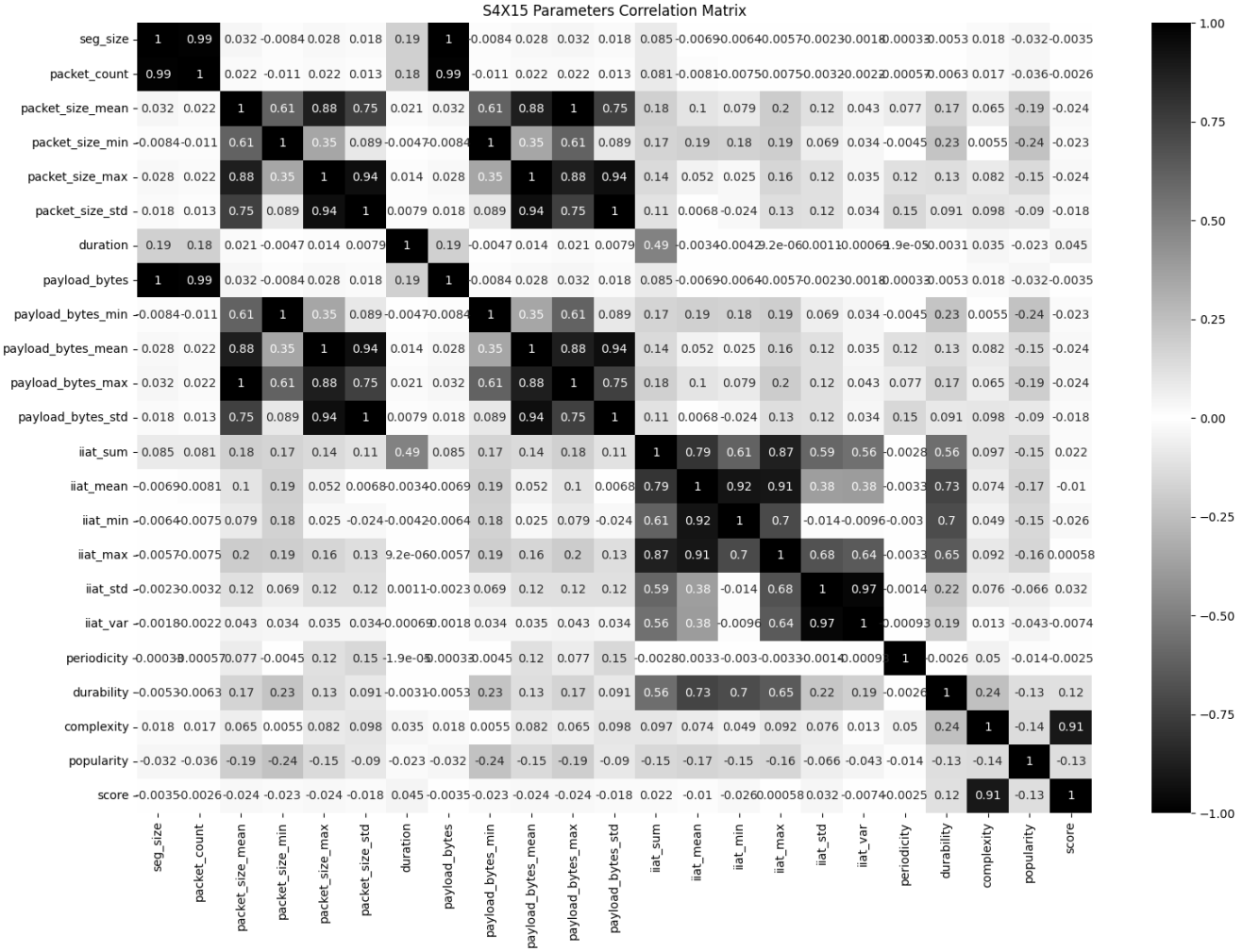

Рисунок В.1 – Кореляційна матриця параметрів датасету 1

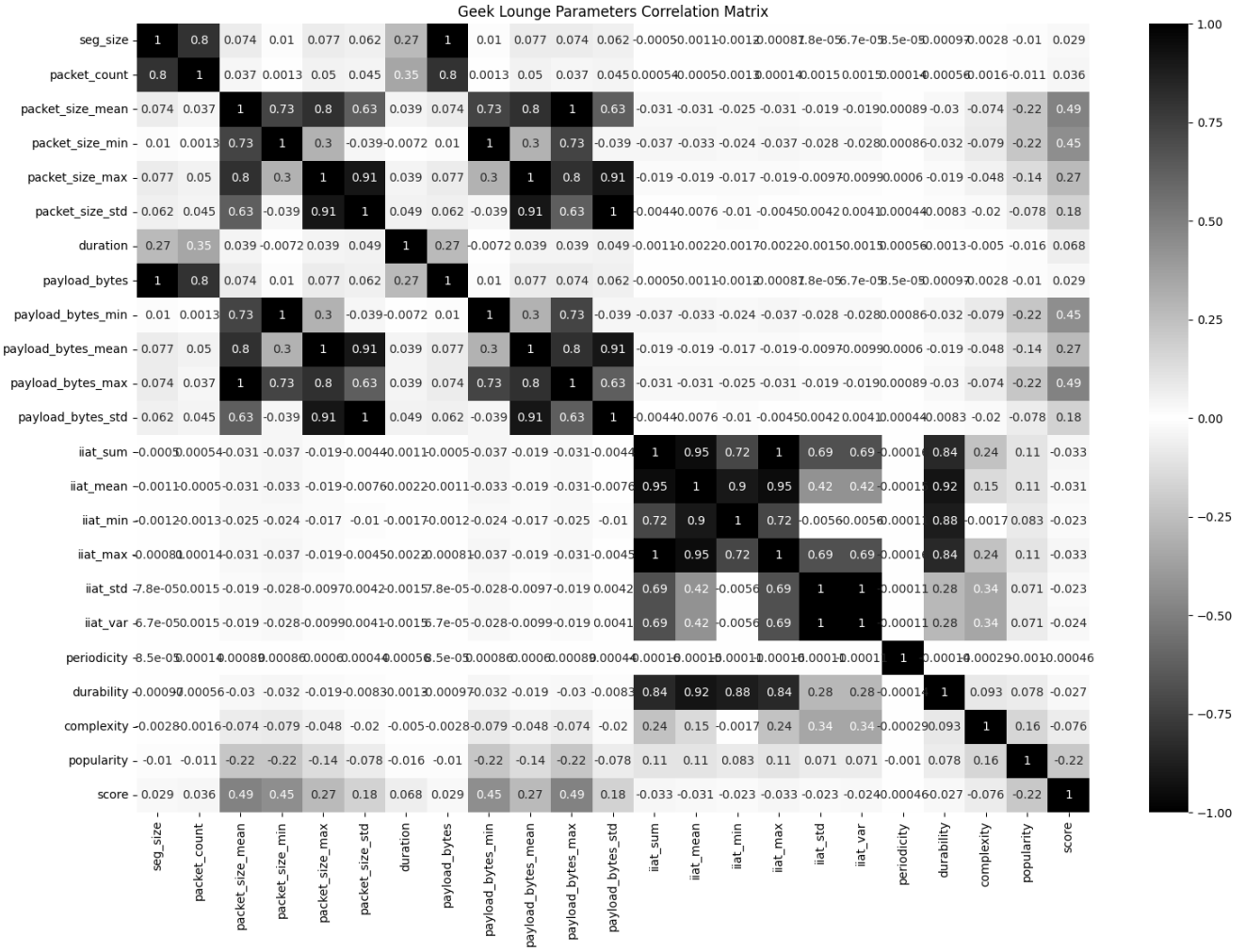

Рисунок В.2 – Кореляційна матриця параметрів датасету 2

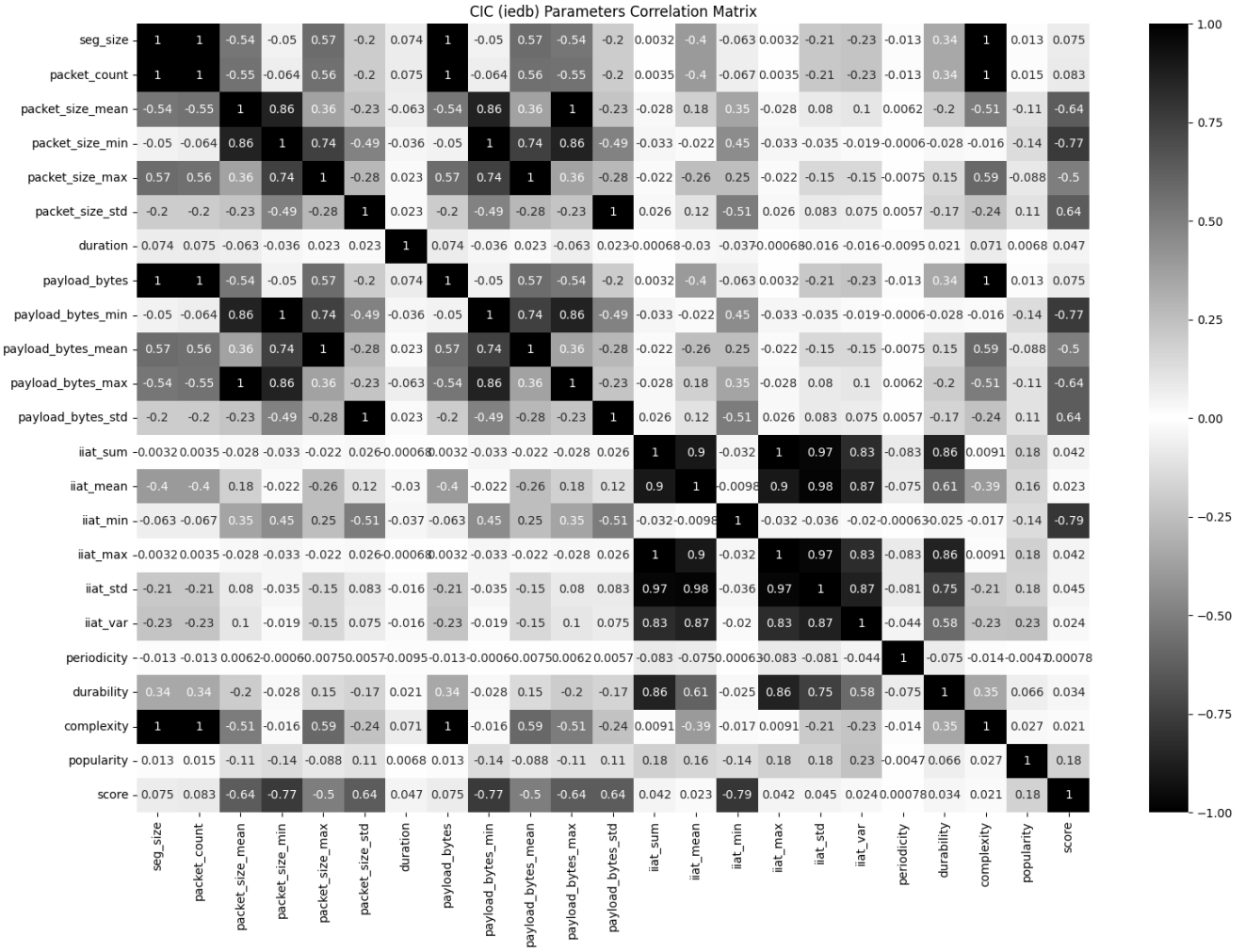

Рисунок В.3 – Кореляційна матриця параметрів датасету 3

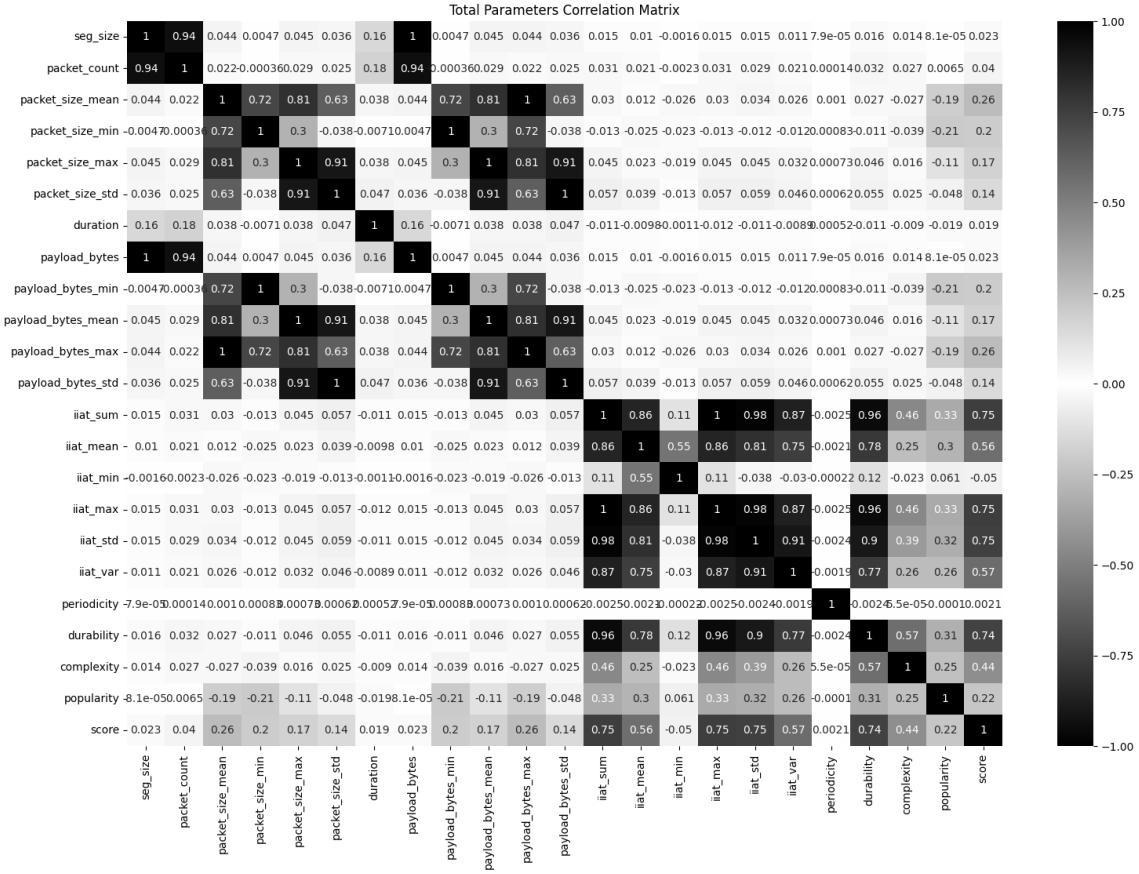

Рисунок В.4 – Кореляційна матриця параметрів датасетів 1-3

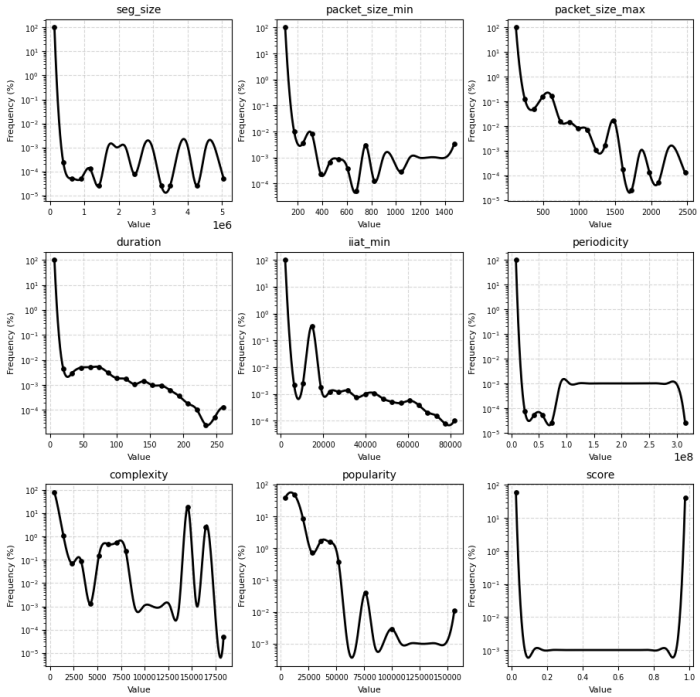

Рисунок В.5 – Розподіл параметрів датасетів 1-3

ДОДАТОК Г РЕЗУЛЬТАТИ ТРЕНУВАННЯ МОДЕЛЕЙ ДЛЯ ВІДНОЛВЕННЯ РОЛІ ОТ-ПРИСТРОЮ

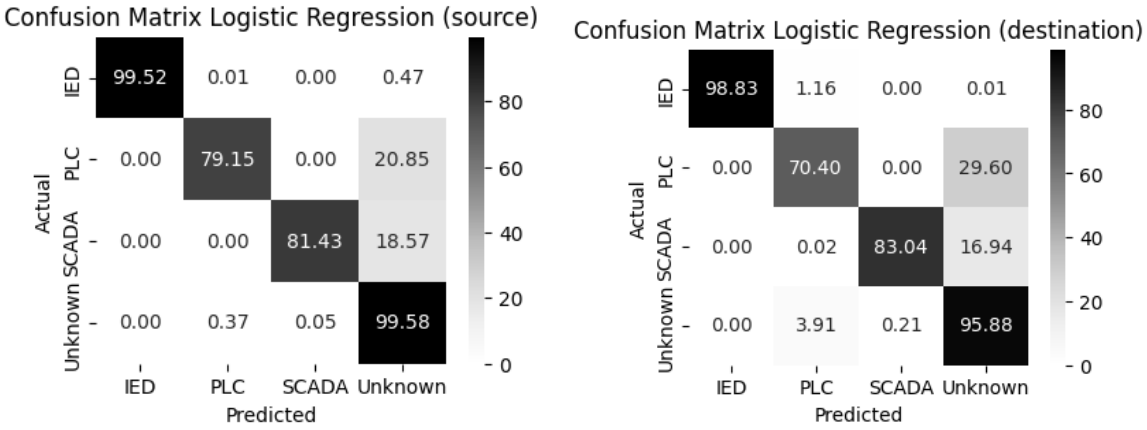

Рисунок Г.1 – Матриця помилок для Logistic Regression

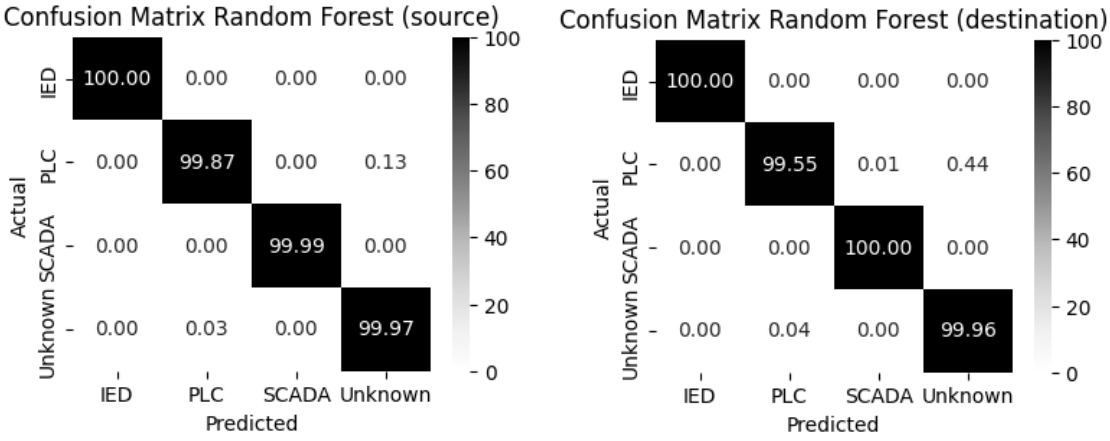

Рисунок Г.2 – Матриця помилок для Random Forest

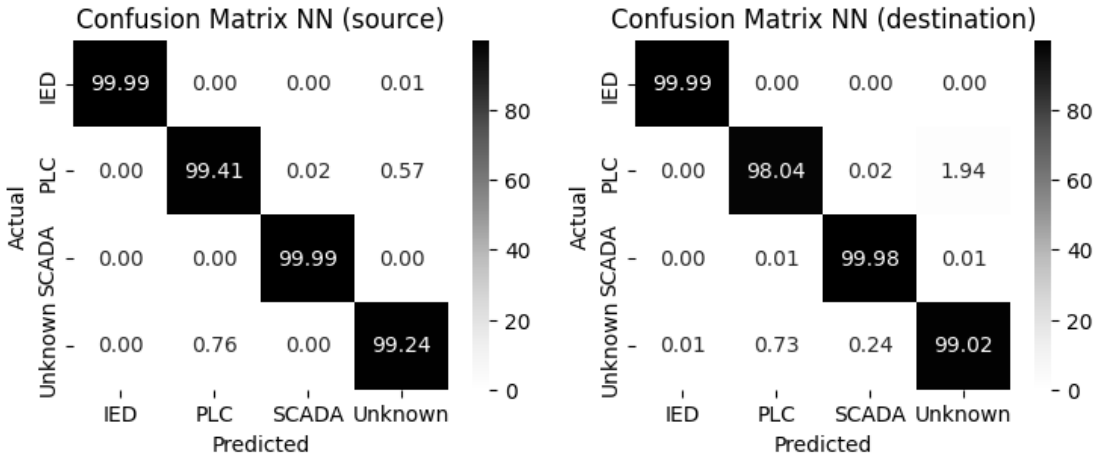

Рисунок Г.3 – Матриця помилок для MLP

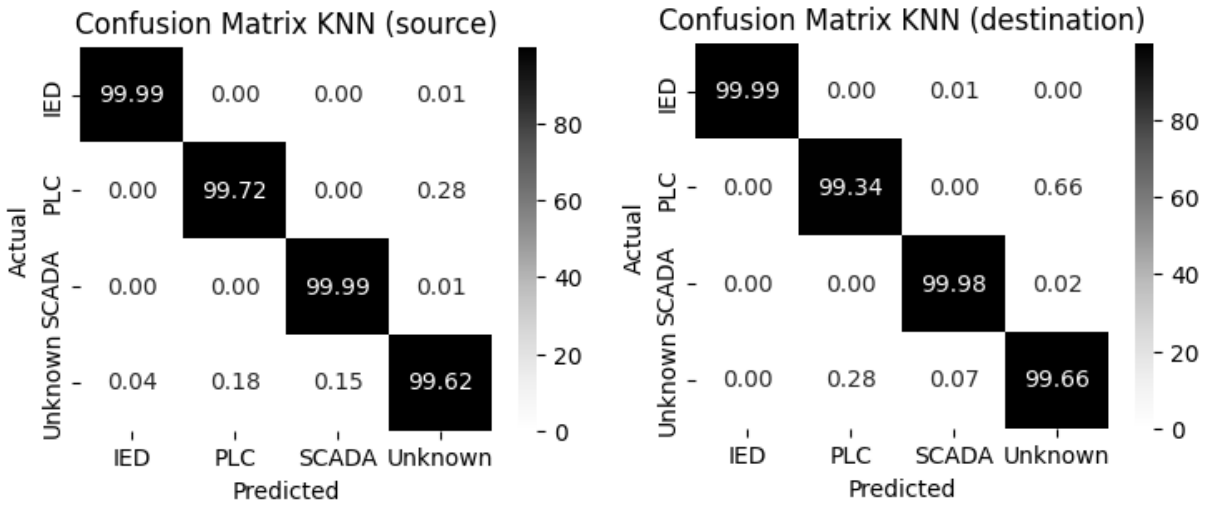

Рисунок Г.4 – Матриця помилок для KNN

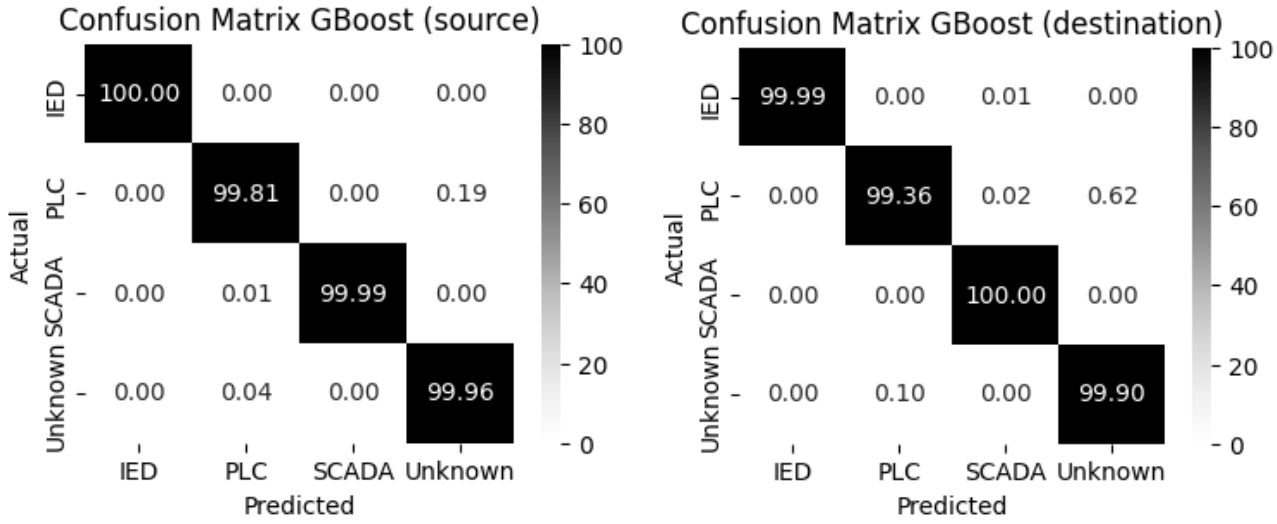

Рисунок Г.5 – Матриця помилок для Gradient Boost

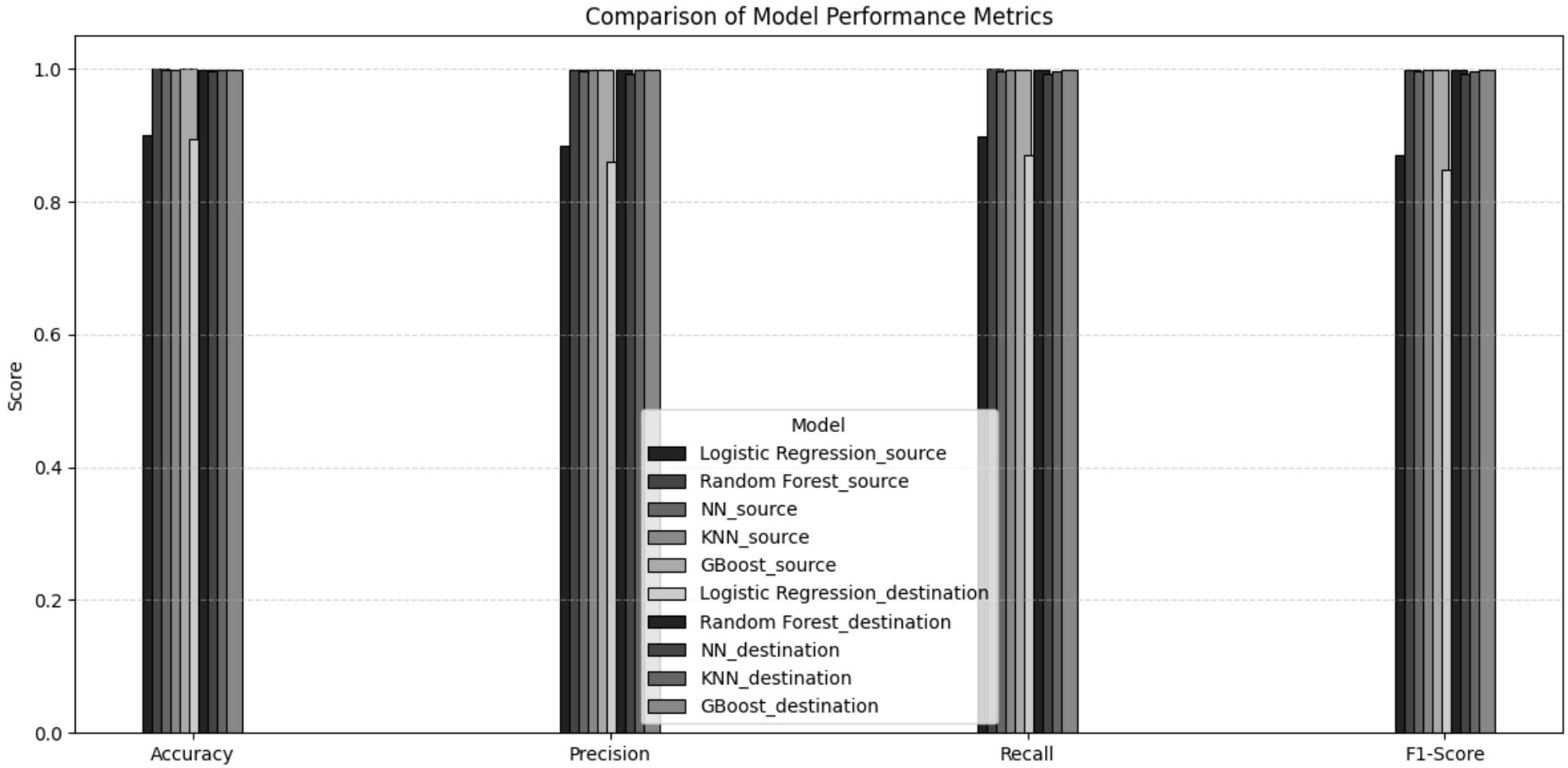

Рисунок Г.6 – Порівняння результативності моделей для вирішення задача класифікації ролі пристрою

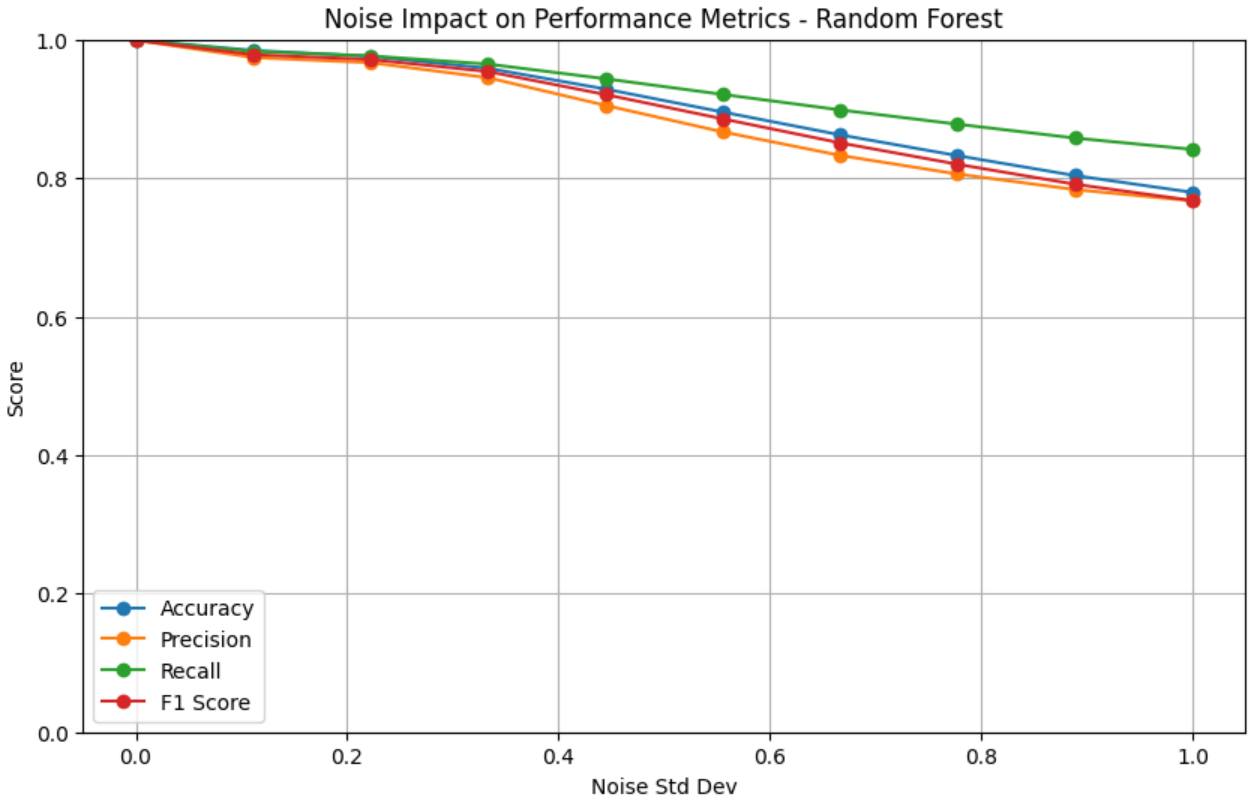

Рисунок Г.7 – Влив шуму на модель Random Forest

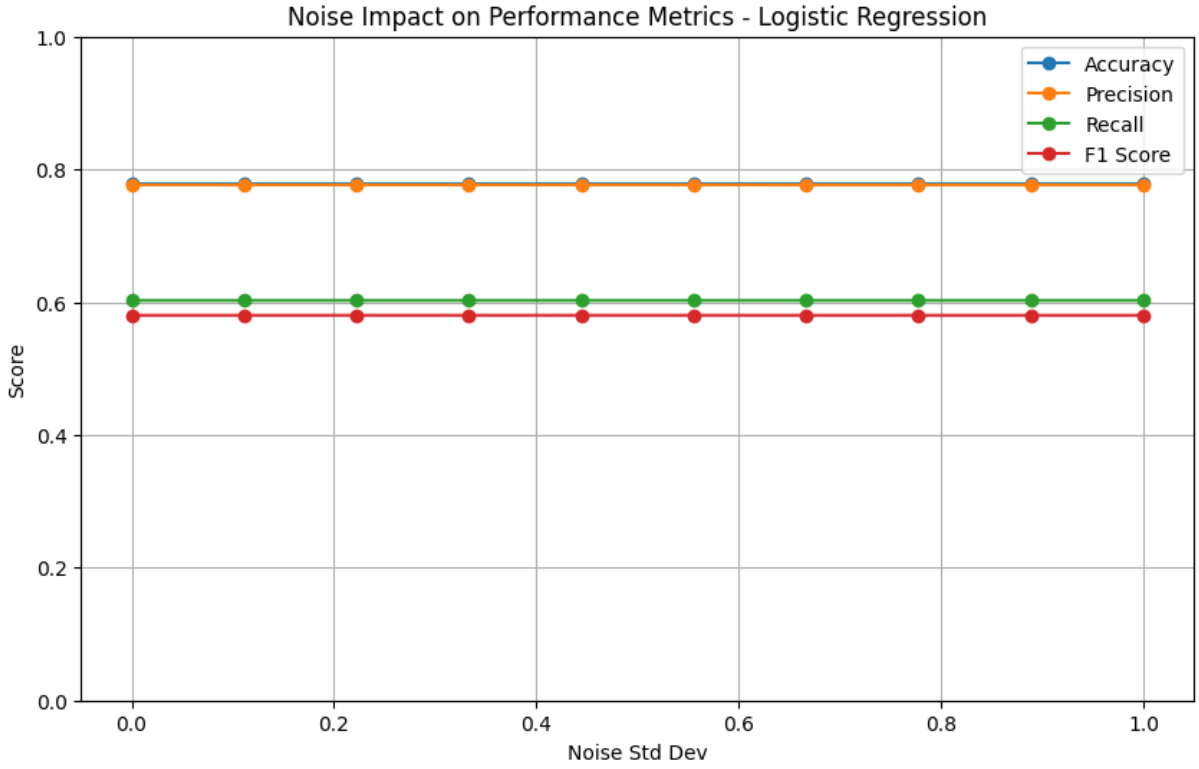

Рисунок Г.8 – Влив шуму на модель Logistic Regression

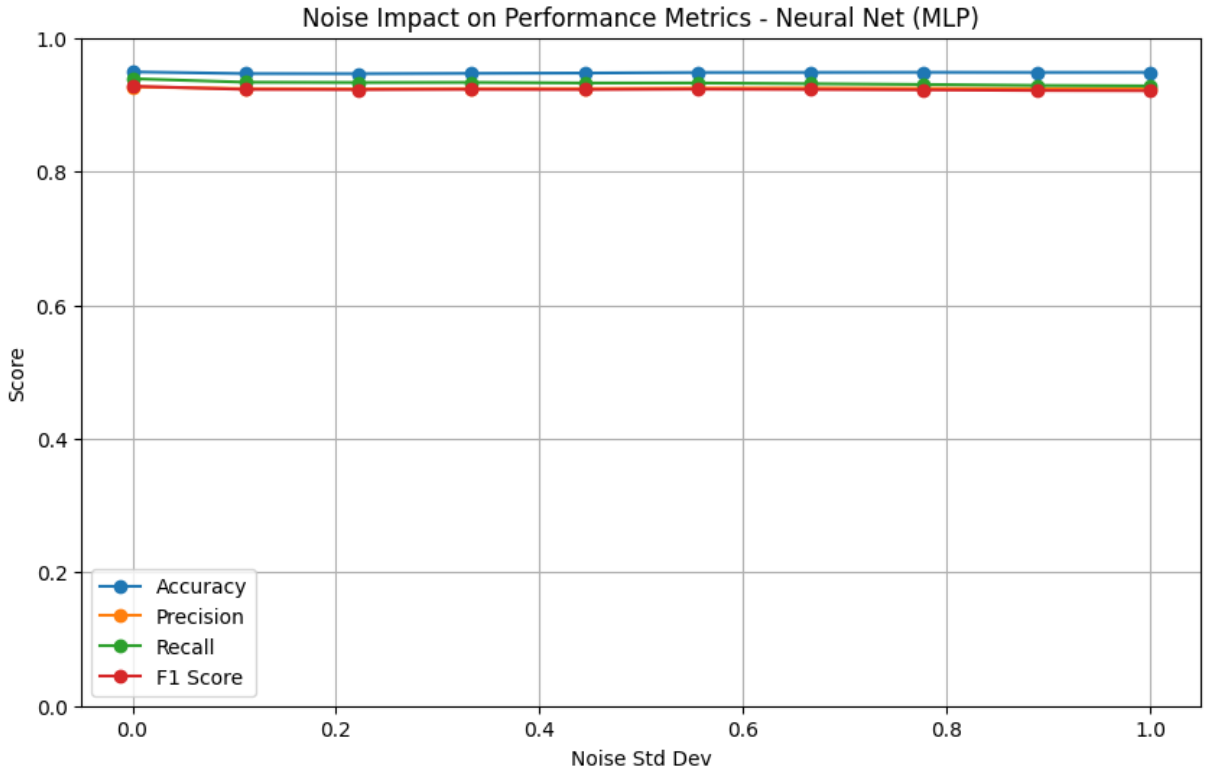

Рисунок Г.9 – Влив шуму на модель MLP

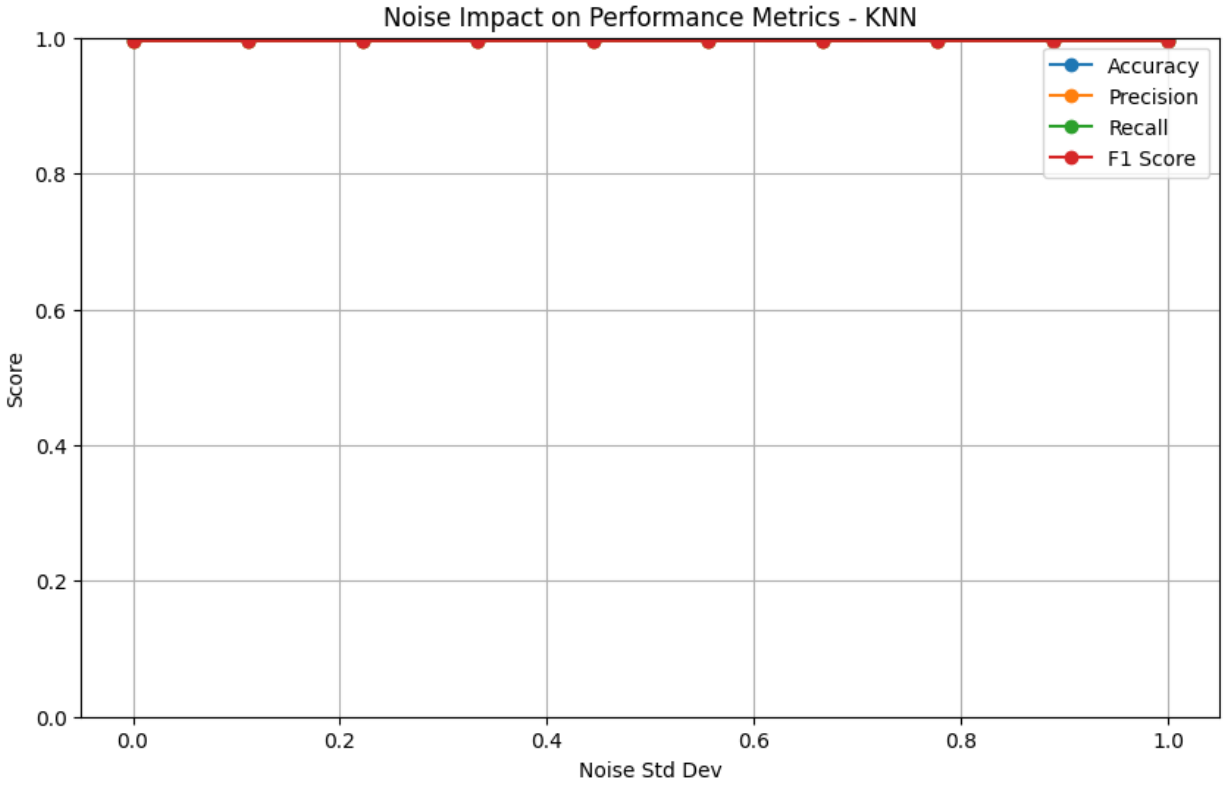

Рисунок Г.10 – Влив шуму на модель KNN

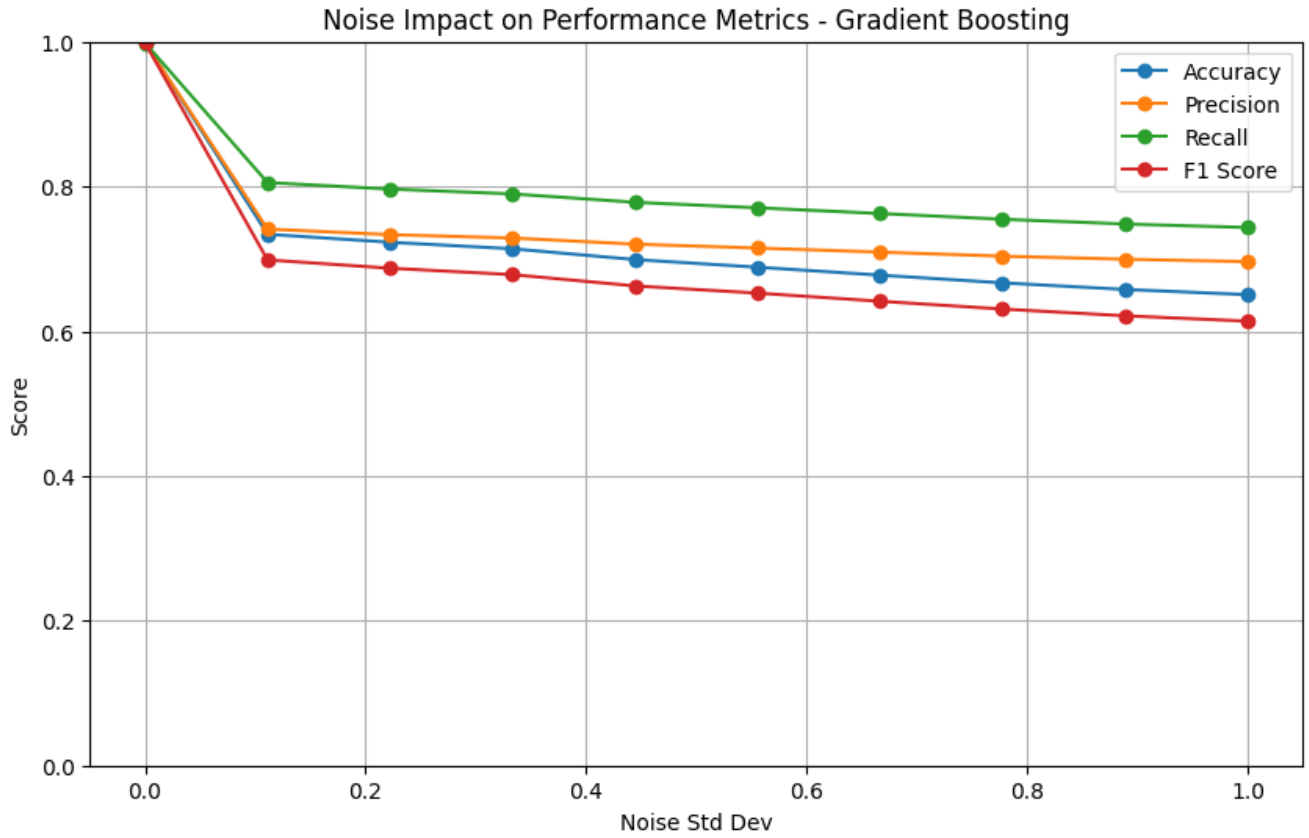

Рисунок Г.7 – Влив шуму на модель Gradient Bossting