ДЕРЖАВНИЙ УНІВЕРСИТЕТ ТЕЛЕКОМУНІКАЦІЙ

НАВЧАЛЬНО-НАУКОВИЙ ІНСТИТУТ ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ

Кафедра Інженерії програмного забезпечення

Пояснювальна записка

до магістерської роботи
на ступінь вищої освіти магістр

на тему: **«ПІДВИЩЕННЯ ЕФЕКТИВНОСТІ КЛАСИФІКАЦІЇ ДОДАТКІВ ІНТЕРНЕТ-ТРАФІКУ ІЗ ЗАСТОСУВАННЯМ МЕТОДІВ МАШИННОГО НАВЧАННЯ»**

Виконала: студент 6 курсу, групи ПДМ-61
спеціальності 121 Інженерія программного забезпечення
(шифр і назва спеціальності)
Козиряцький А.П.
(прізвище та ініціали)
Керівник Жебка В.В.
(прізвище та ініціали)
Рецензент _____
(прізвище та ініціали)

Київ – 2020

ДЕРЖАВНИЙ УНІВЕРСИТЕТ ТЕЛЕКОМУНІКАЦІЙ

НАВЧАЛЬНО-НАУКОВИЙ ІНСТИТУТ ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ

Кафедра Інженерії програмного забезпечення
Ступінь вищої освіти «Магістр»
Напрям підготовки 121 Інженерія программного забезпечення

ЗАТВЕРДЖУЮ
Завідувач кафедри
Інженерії програмного забезпечення
_____ О.В.Негоденко
“____” _____2020 року

З А В Д А Н Н Я
НА МАГІСТЕРСЬКУ РОБОТУ СТУДЕНТУ

Козиряцький Антон Павлович
(прізвище, ім’я, по батькові)

1.Тема роботи Підвищення ефективності класифікації додатків інтернет-трафіку із застосуванням методів машинного навчання
Керівник роботи к.т.н., доцент Жебка В.В.
(прізвище, ім’я, по батькові, науковий ступінь, вчене звання)
затверджені наказом вищого навчального закладу від “13” 10 2020 року № 230
2. Строк подання студентом роботи 24.12.2020
3. Вихідні дані до роботи:
3.1 Інтернет-трафік;
3.2 Програмне забезпечення;
3.3 Навчальна вибірка;
3.4 Машинне навчання;
3.5 Науково-технічна література з питань, пов‘язаних з темою работи.

4. Зміст розрахунково-пояснювальної записки (перелік питань, які потрібно розробити:
4.1 Дослідження моделей і алгоритмів класифікації додатків інтернет-трафіку в комп'ютерних мережах;
4.2 Аналіз задач формування вихідних даних та програмного забезпечення, що виникають при розробці системи класифікації трафіку;
4.3 Практичне дослідження класифікації інтернет-трафіку в умовах наявності небажаного фонового впливу.

5.Перелік графічного матеріалу (з точним зазначенням обов’язкових креслень)
5.1 Мета та завдання магістерської роботи;
5.2 Поширення інтернету та прогноз зростання трафіку;
5.3 Напрямки розвитку технологій аналізу мережевого трафіку;
5.4 Методи і алгоритми класифікації мережевого трафіку;

5.5 Формування вихідних даних і аналіз програмного забезпечення;
5.6 Оцінка алгоритмів виділення інформативних ознак та їх порівняння;
5.7 Експериментальні результати класифікації на етапі навчання;
5.8 Ефективність застосування алгоритму RF в задачах класифікації додатків;
5.9 Аналіз алгоритмів кластеризації та їх порівняння;
5.10 Застосування програмного забезпечення для вирішення задачі класифікації інтернет-трафіку методом машинного навчання;
5.11 Висновки та апробація результатів магістерської роботи.

6. Дата видачі завдання 02.11.2020

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів магістерської роботи	Строк виконання етапів роботи	Примітка
1.	Підбір науково-технічної літератури	07.11.2020	Виконано
2.	Дослідження моделей і алгоритмів класифікації додатків інтернет трафіку в комп'ютерних мережах	13.11.2020	Виконано
3.	Аналіз задач раціонального формування вихідних даних та огляд програмного забезпечення, що виникають при розробці системи класифікації трафіку	17.11.2020	Виконано
4.	Практичне дослідження класифікації інтернет-трафіку в умовах наявності небажаного фонового впливу	01.12.2020	Виконано
5.	Висновки по роботі, оформлення.	11.12.2020	Виконано
6.	Розробка обов`язкових демонстраційних матеріалів	15.12.2020	Виконано

Студент _____ Козиряцький А.П.
(підпис) (прізвище та ініціали)

Керівник роботи _____ Жебка В.В.
(підпис) (прізвище та ініціали)

РЕФЕРАТ

Текстова частина магістерської роботи 89 с., 57 рис., 21 табл., 25 джерел.

ТРАФІК, МАШИННЕ НАВЧАННЯ, IOT, IPV4, M2M-З'ЄДНАННЯ, МЕТОД, SPI, ПАКЕТ, CHECK POINT, АЛГОРИТМ, КЛАСИФІКАТОР, ПРОГРАМНЕ ЗАБЕЗПЕЧЕННЯ, АРХІТЕКТУРА, АТРИБУТ.

Об'єктом дослідження – процес класифікації IP-трафіка.

Мета роботи – підвищення функціонування ефективності класифікації додатків інтернет-трафіку із застосуванням машинного навчання.

Предмет дослідження – методи та засоби класифікації інтернет-трафіка

Методи дослідження. У магістерській роботі використані методи теорії ймовірності, математичної статистики, математичне імітаційне моделювання, машинного навчання та інтелектуального аналізу (обробки) даних.

Виконуючи поставлені завдання, магістерської роботи, в першу чергу було виконано аналіз поширення інтернету та прогноз зростання трафіку. Проведено аналіз напрямків розвитку технологій аналізу мережевого трафіку Розглянуто особливості класифікації інтернет-трафіку та методи і алгоритми його класифікації.

У другому розділі аналізується задача раціонального формування вихідних даних, огляд програмного забезпечення, що виникають при розробці системи класифікації трафіку.

У практичній частині роботи проведено експериментальне дослідження класифікації інтернет-трафіку в умовах наявності небажаного фонового впливу та представлено спеціалізоване програмне забезпечення, що реалізує методи машинного навчання в реальному часу для класифікації інтернет-трафіку.

ЗМІСТ

Стор.

ВСТУП……..9

1 ДОСЛІДЖЕННЯ МОДЕЛЕЙ І АЛГОРИТМІВ КЛАСИФІКАЦІЇ ДОДАТКІВ ІНТЕРНЕТ ТРАФІКУ В КОМП'ЮТЕРНИХ МЕРЕЖАХ.....11

1.1 Поширення інтернету та прогноз зростання трафіку……11

1.2 Напрямки розвитку технологій аналізу мережевого трафіку…….....13

1.3 Облік стану потоку при аналізі інтернет трафіку……..18

1.4 Методи і алгоритми класифікації мережевого трафіку……20

2 АНАЛІЗ ЗАДАЧ РАЦІОНАЛЬНОГО ФОРМУВАННЯ ВИХІДНИХ ДАНИХ ТА ОГЛЯД ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ, ЩО ВИНИКАЮТЬ ПРИ РОЗРОБЦІ СИСТЕМИ КЛАССИФИКЦІЇ ТРАФІКУ……...35

2.1 Формування вихідних даних і аналіз програмного забезпечення……...35

2.2 Оцінка алгоритмів виділення інформативних ознак та їх порівняння……42

2.3 Аналіз впливу обсягу навчальної вибірки на якість класифікації інтернет- трафіку……..46

2.3.1 Оцінка якості класифікації по пакетам і потокам та їх порівняння...46

2.3.2 Результати класифікації додатків на етапі тестування……...49

2.3.3 Експериментальні результати класифікації на етапі навчання……...54

2.4 Дослідження ефективності застосування алгоритму RF в задачах класифікації додатків……..59

3 ПРАКТИЧНЕ ДОСЛІДЖЕННЯ КЛАСИФІКАЦІЇ ІНТЕРНЕТ-ТРАФІКУ В УМОВАХ НАЯВНОСТІ НЕБАЖАНОГО ФОНОВОГО ВПЛИВУ……....65

3.1 Аналіз впливу фонового трафіку на якість класифікації……..65

3.2 Аналіз неконтрольованої кластеризації інтернет-трафіку……...72

3.3 Аналіз алгоритмів кластеризації та їх порівняння……76

3.3.1 Порівняльний аналіз алгоритмів контрольованого і неконтрольованого навчання……..85
3.3.2 Порівняння алгоритмів кластеризації заснованих на RF з класичними алгоритмами EM і K-Means……...88
3.4 Приклад реалізації та застосування програмного забезпечення для вирішення задачі класифікації інтернет-трафіку методом машинного навчання……...91
ВИСНОВКИ……..96
ПЕРЕЛІК ПОСИЛАНЬ……..98
Додаток А. Код для створення бази даних flowclass……...101
ДЕМОНСТРАЦІЙНІ МАТЕРІАЛИ (Презентація)……102

ВСТУП

Актуальність теми. Класифікація інтернет-трафіку дозволяє виявляти його тип і структуру, що критично важливо для управління такими технологіями, як мережева безпека, диференціація сервісів, управління параметрами трафіку і ін.

Для ефективного управління мережею необхідне точне відповідність використовуваних мережевих додатків відповідного трафіку, а також повноцінний контроль над використовуваними додатками. При цьому проводяться певні заходи з управління та безпеки за допомогою впровадження набору вивірених правил доступу до конкретних типів додатків, сервісів або контенту. Безперервний моніторинг типів додатків необхідний для вирішення таких практичних завдань, як аналіз тенденцій розвитку Інтернет, планування мережевої інфраструктури, розробка мережевих пристроїв. Це дозволяє, зокрема, підвищити ефективність функціонування різних центрів обробки даних (ЦОД).

Обмеженість традиційних методів призвела до того, що в останні роки інтенсифікувалися дослідження з пошуку і розвитку альтернативних підходів до ефективної класифікації мережевого трафіку. У зв'язку з цим перспективними визнані методи статистичної класифікації, засновані на виявленні стійких відмінностей в статистичних параметрах трафіку. Із статистичних методів класифікації трафіку особливий інтерес представляють технології машинного навчання (Machine Learning) і інтелектуального аналізу даних (Data Mining), що опинилися найбільш ефективними в різних областях інформатики, радіотехніки та інших напрямках. Вищесказане обумовлює актуальність дослідження щодо ефективної класифікації IP-трафіку на основі методу машинного навчання.

Метою магістерської роботи є підвищення функціонування ефективності класифікації додатків інтернет-трафіку із застосуванням машинного навчання.

Джерела дослідження:

- https://habr.com/ru/hub/machine_learning;
- https://www.it.ua/ru/knowledge-base/technology-innovation/machine-learning;
- http://www.dut.edu.ua/uploads/p_1831_14796413.pdf;

- https://tproger.ru/translations/top-machine-learning-algorithms;

- https://er.chdtu.edu.ua/bitstream/ChSTU/304/1/IMA-2019.pdf.

Завдання магістерської роботи: дослідження моделей і алгоритмів класифікації додатків інтернет-трафіку в комп'ютерних мережах; формування вихідних даних та аналіз програмного забезпечення, що виникають при розробці системи класифікації трафіку; практичне дослідження класифікації інтернет-трафіку в умовах наявності небажаного фонового впливу.

Об'єктом дослідження є процес класифікації IP-трафіка.

Предметом дослідження є методи та засоби класифікації інтернет-трафіка

Методи дослідження. У магістерській роботі використані методи дослідження теорія ймовірності, математична статистика, математичне імітаційне моделювання, машинного навчання та інтелектуального аналізу (обробки) даних.

Наукова новизна одержаних результатів. У магістерській роботі обґрунтовано підвищення якості класифікації додатків, в умовах наявності фонового трафіку, яке може здійснюватися не тільки шляхом використання «непомічених» примірників даних, але і введенням додаткових обмежень - наборів еквівалентності у вигляді «напівконтрольованого навчання», в якому частковий контроль реалізується у вигляді обмежень на основі наборів, а не у вигляді міток класу.

Практична значення одержаних результатів. Впровадження досліджених в роботі підходів, дозволить виробляти класифікацію, аналіз і фільтрацію мережевого трафіку шкідливих і небажаних програм, з більш високою ефективністю відповідно до запропонованими показниками, в порівнянні з іншими методами класифікації мережевого трафіку, такими як Deep Packet Inspection (DPI) або аналіз номерів портів.

Апробація результатів магістерської роботи. Основні положення і результати магістерської роботи доповідались і обговорювались на двох науково-практичних конференціях.

Публікації. За матеріалами роботи опубліковано одну статтю у науковому журналі.

1 ДОСЛІДЖЕННЯ МОДЕЛЕЙ І АЛГОРИТМІВ КЛАСИФІКАЦІЇ ДОДАТКІВ ІНТЕРНЕТ-ТРАФІКУ В КОМП'ЮТЕРНИХ МЕРЕЖАХ

1.1 Поширення інтернету та прогноз зростання трафіку

Більшість тенденцій телекомунікаційної галузі говорять про її розвиток, масштабування і глобальному проникненні інтернет технологій в усі галузі життя.

Незалежні аналітики пророкують стрімке зростання трафіку в найближчі роки. Пояснюють це розвитком технологій і збільшенням числа інтернет-користувачів, як наслідок, збільшення навантаження на мережі операторів, масове впровадження об'єктів Інтернету речей, примноження кількості підключених до мережі гаджетів, користувачів, використання різного роду додатків, «розгін» швидкості передачі даних і т.д.[1].

За даними Cisco за останні два десятиліття обсяг інтернет-трафіку істотно зріс. З 1992 по 2017 роки денний обсяг зріс з 100 ГБ до 45 000+ ГБ. А до 2022 року світ чекає триразове збільшення інтернет трафіку.

Щорічні прогнози Cisco засновані на думках аналітиків-експертів, власних оцінках і прямому зборі даних.

За колишніми прогнозами в період 2013-2018 рр. кількість мобільних користувачів мала збільшитися до 4,9 млрд чоловік. За даними Hootsuite на 2019 рік їх число відповідає 5,1 млрд.

Мобільному відео передбачали захоплення 69% світового інтернет трафіку. У 2019 другим за відвідуваністю ресурсом визнаний YouTube. Щомісячне число відео-глядачів - 92% користувачів, або 4 млрд осіб.

У 2022 році на одну людину доведеться 3,6 онлайн-пристроїв (в цілому 28,5 млрд). Розподіл трафіку за типами пристроїв виглядає так:

- 71% - бездротові і мобільні, з них 44% на смартфони;
- 29% - дротяні, з них 19% на ПК.

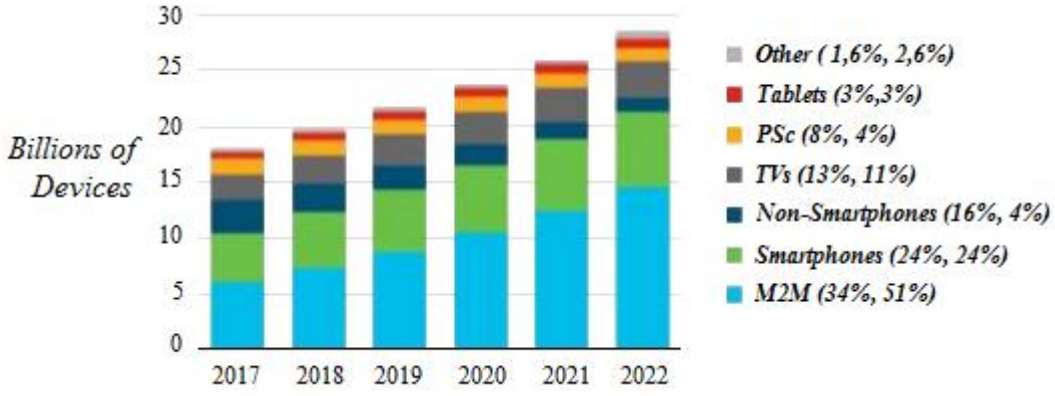


Рисунок 1.1 - Глобальний ріст пристроїв і підключень

82% від світового трафіку займе відео і 17% припаде на лайф-відео. Інтернет-відео на ТБ зросте втричі, до 27% від фіксованого відео-трафіку. Удвічі додасться споживчий трафік відео за запитом, що відповідає 10 млрд DVD-дисків на місяць.

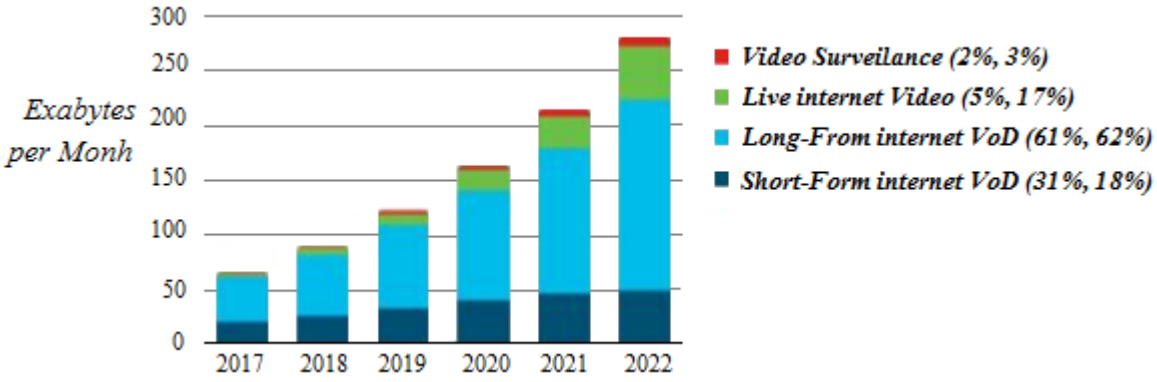


Рисунок 1.2 - Глобальне інтернет-відео по підсегментам

Трафік інтернет-ігор виросте на 55%, а трафік віртуальної і доповненої реальності - на 65%.

Серед основних тенденцій:

- поєднання різноформатних пристроїв і різнотипних з'єднань. До 2022 року 81% IP-трафіку і інтернет-трафіку доведеться на пристрої, що не відносяться до ПК.

- перехід від середовища IPv4 до середовища IPv6. Завдяки цьому стає можливим підключення до Інтернету речей (IoT). До 2022 року 64% усіх

стаціонарних і мобільних мережевих пристроїв перейдуть на IPv6, в порівнянні з 32% в 2017 році [2];

- посилення зростання IoT за рахунок додатків M2M. Інтернет речей поступово об'єднує людей, предмети, процеси і дані. До 2022 року на кожну людину буде підключено 1,8 M2M з'єднання.

До головних трендів відносяться:

- зростання трафіку додатків. До 2022 року загальна частка всіх форм IP-відео, що передаються відеофайлів, відео-потокових ігр та відеоконференцій становитиме 80-90% від загального IP-трафіку;

- пріоритет інтернет-відео над традиційним ТВ. Обсяг трафіку по інтернет-ТБ зросте на 72%;

- посилення уваги до онлайн-безпеки. При цьому до 2022 року загальна кількість DDoS-атак збільшиться вдвічі і досягне 14,5 млн.;

- збільшення швидкості інтернет-доступу. Середньостатистична швидкість на 2022 рік складе 75,4 Мбіт / с проти 39,0 Мбіт / с в 2017 році;

- поширення Wi-Fi технологій. У 4 рази зросте кількість громадських точок доступу Wi-Fi, до 549 млн до 2022 року.

В цілому очікуваний обсяг глобального інтернет-трафіку складе 4,8 ZB в рік (1,9+ трлн ГБ). Щомісячний IP-трафік на душу населення зросте з 16 ГБ до 50 ГБ.

1.2 Напрямки розвитку технологій аналізу мережевого трафіку

Аналіз мережевого трафіку на сьогоднішній день - дуже велика тема. Під «аналізом мережевого трафіку» розуміється сукупність технологій і їх реалізацій, що дозволяють проводити накопичення, обробку, класифікацію, контроль і модифікацію мережевих пакетів в залежності від їх вмісту в реальному часі [3].

Можна виділити два основних напрямків розвитку:

- зростання «глибини» аналізу для окремого мережевого пакету, тобто збільшення рівня моделі OSI, дані якого піддаються аналізу;

- повнота обліку стану потоку, до якого належить пакет, а також інших потоків, пов'язаних з даними.

Глибина аналізу мережевих пакетів. З цієї «осі» технології аналізу трафіку розвивалися послідовно, кожна наступна успадковувала частина попередніх механізмів і додавала свої. Можна виділити три рівня розвитку технології, які наведені на рис. 1.3.[4]

Рисунок 1.3 - Рівні розвитку технології аналізу мережевого трафіку по «глибині»

Розглянемо ці рівні більш детально:

1.Поверховий аналіз пакетів (SPI). Технологія аналізу трафіку, яка грунтується виключно на заголовках пакету рівнів L1-L3 по моделі OSI. Пред'являє низькі вимоги до обчислювальних ресурсів, що дозволяє аналізувати великі обсяги рафіку. Технологія широко поширена, на її основі працює більшість міжмережевих екранів операційних систем (зокрема в ОС Windows), маршрутизаторів і інших мережевих пристроїв. На її основі реалізовані мережеві списки контролю доступу на рівні IP адрес і портів (Access Control List, ACL). Таким чином, дана технологія добре підходить для розмежування доступу ззовні до окремих комп'ютерів (IP) і сервісів (порти) внутрішньої мережі [5].

2. Середній аналіз пакетів (MPI). Технологія аналізу трафіку, яка грунтується на інспектуванні сесій і сеансів зв'язку, ініційованих додатком, але встановлюваних шлюзомпосередником рис. 1.4. Також застосовується термін «проксі додатків» (Application proxy).

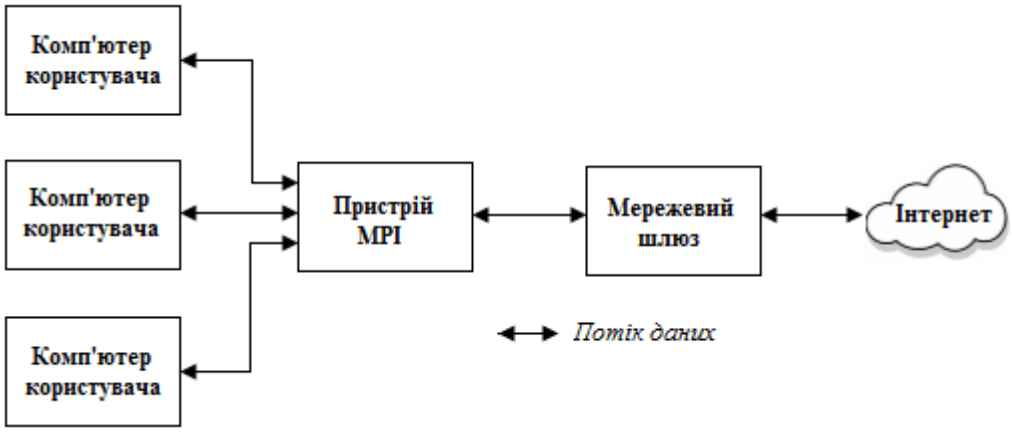


Рисунок 1.4 - Схема застосування пристроїв аналізу на основі технології MPI

В рамках даної технології вміст пакетів аналізується частково і по визначеним правилам. Не використовуються складні методи аналізу типу сигнатурного. Пристрої, що реалізують даний функціонал розміщуються між провайдером інтернету і кінцевим користувачам.

Дані пристрої розбирають заголовки аж до транспортного рівня і невелику частину даних пакета для зіставлення розібраної частини з деяким списком розбору (parse list), з подальшою реакцією в разі їх виявлення. Дані списки зазвичай коротше списків ACL і надають більш широкий діапазон дій на відміну від «дозволити/заборонити» в разі ACL. Ці списки також більш виразні, так як дозволяють прив'язуватися ні до IP-адрес, а до формату даних пакетів і даними деяких протоколів рівня додатку, наприклад, URL-адресами в разі протоколу HTTP. За допомогою MPI можна, наприклад, заблокувати можливість отримання flash-файлів або картинок з певних інтернет сервісів (на рівні уявлення OSI) або заблокувати частину команд (на рівні додатку OSI) в окремих протоколах. Набір протоколів, як правило, дуже обмежений. Наприклад, в перших версіях Check Point

FireWall-1 (CheckPoint FW-1) підтримувалися протоколи Telnet, FTP, HTTP, а в Cisco Private Internet Exchange (Cisco PIX) - FTP, HTTP, RSH, SMTP і SQLNET. Згодом дані набори незначно розширювалися. Також відомо, що дана технологія використовується в продуктах компаній McAfee і Symantec. Міжмережеві екрани, що використовують цю технологію, відносяться до другого покоління [6].

Дана технологія більш гнучка в порівнянні з SPI і, крім розмежування доступу, підходить для більшого числа завдань – кешування вмісту, аналіз стисненого/шифрованого трафіку, обмеження функціонала окремих протоколів шляхом заборони окремих команд. Завдяки підключенню в режимі проксі, може служити в якості Wan Optimizer'a (див. вище).

Основний недолік MPI - погана масштабованість: кожна команда і протокол вимагають окремого «шлюзу» (вхідний-вихідний порти). Крім того, робота в режимі проксі сильно знижує швидкість обробки. Для зниження навантаження на проксі-сервер був розроблений протокол ICAP [7], дозволяє проксі-серверів відправляти проходять через них дані для проведення аналізу стороннім серверам на предмет безпеки або аналізу вмісту. Ця схема реалізована в антивірусному продукті ClamAV, який може підключатися до проксі-серверів Squid і NetCache, згаданим вище.

Ці фактори сильно обмежують застосування даної технології на рівні провайдерів інтернету внаслідок необхідності аналізу великого числа протоколів і команд на широких каналах зв'язку.

Глибокий аналіз пакетів (DPI). Іноді вживають більш вузький термін - DPP (Deep Packet Processing), який має на увазі такі дії над пакетами, як модифікація, фільтрація або перенаправлення. Сьогодні обидва терміни часто використовуються як взаємозамінні. Дана технологія є логічним розвитком MPI. В рамках даного підходу аналізатор переглядає вміст кожного пакета повністю. Одним з важливих відмінностей від попередніх технологій є те, що системи на базі DPI можуть приймати рішення не тільки по вмісту пакетів, але і за непрямими ознаками, властивим якимось певним мережевим програмами і протоколам. Для цього може використовуватися статистичний аналіз. Наприклад, аналіз частоти зустрічі певних

символів, довжин пакетів, відстань між мітками часу послідовних пакетів і т.д. Також, в порівнянні з попередніми підходами, значно розширено список застосувань технології: класифікація, обмеження смуги, пріоритезація, маркування, кешування і т. д. Технологія DPI отримала розвиток, перш за все, через стрімкого зростання обчислювальних здібностей процесорів, їх швидкодії і, відповідно, можливостей для більш повного і точного аналізу мережевих даних.

На відміну від MPI, дана технологія спочатку розроблялася для високошвидкісної обробки та ідентифікації великої кількості додатків в реальному часі. Таким чином, рішення на основі DPI добре масштабируются як по ширині мережевого каналу (відомі рішення, що працюють на каналах близько 100 Гбіт / сек), так і за кількістю ідентифікованих додатків (в існуючих рішеннях – порядку декількох тисяч). З точки зору реалізації, основний компонент будь-якого рішення DPI - модуль класифікації, що відповідає за класифікацію мережевих потоків. При цьому в залежності від цілей застосування DPI, класифікація може виконуватися з різною точністю:

- тип протоколу або додатка (наприклад, Web, P2P, VoIP);
- конкретний протокол рівня додатка (HTTP, BitTorrent, SIP);
- додаток, що використовує протокол (Google Chrome, µTorrent, Skype).

Важливо відзначити, що відповідність між класами різних рівнів точності неоднозначно, що показано на рис. 1.5.

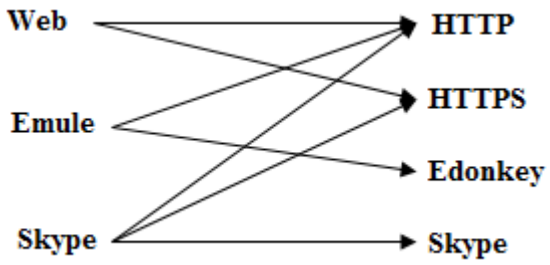


Рисунок 1.5 - Різниця між ідентифікацією додатків (зліва) і протоколів (праворуч).

Технологія DPI на даний момент є поточним стандартом де-факто для засобів аналізу мережевого трафіку і відноситься до області критично важливих технологій необхідних для забезпечення, як мережевої безпеки, так і вимог законодавства. Внаслідок цього останнім часом на міжнародному рівні було прийнято низку стандартів, вимог і рекомендацій щодо особливостей реалізації, внутрішньою будовою та набору функцій відповідних коштів. Ця технологія рідко застосовується в міжмережевих екранах - це скоріше область IDS / IPS систем, як винятків можна вказати екрани Hogwash і Shield. Однак міжмережеві екрани, які стосуються четвертого покоління [6] можуть враховувати дані IDS / IPS систем в процесі аналізу.

1.3 Облік стану потоку при аналізі інтернет-трафіку

Другим напрямком розвитку технології аналізу можна назвати облік стану протоколу (потоку) в процесі аналізу – stateless/statefull види аналізу. Даний напрямок актуально тільки для протоколів, що використовують транспортний протокол із установленням з'єднання (connection-oriented). Це означає, що перед будь-яким обміном командами і даними відбувається процес «Встановлення з'єднання», в ході якого сторони обмінюються фіксованою послідовністю пакетів, яка часто називається «Рукостисканням» (handshake), а після завершення обміну відбувається аналогічний процес «закриття з'єднання». До connection-oriented протоколам, зокрема, відноситься протокол TCP, але не UDP. Однак слід врахувати, що поверх UDP може бути реалізований інший транспортний протокол, з встановленням з'єднання. Як приклад можна привести протокол Quick UDP Internet Connections (QUIC) [8] – протокол транспортного рівня з встановленням з'єднання, що використовує UDP. З цього випливає, що, в загальному випадку, не можна повністю виключити statefull аналіз для UDP пакетів.

Для опису відмінностей описаних підходів потрібно дати визначення поняттю «потік пакетів». Відомі різні визначення даного поняття. Частина з

найбільш широко використовуваних приведена на сайті Center for Applied Internet Data Analysis (CAIDA).

З урахуванням даного визначення, можна сформулювати відміну statefull від stateless підходу. Воно полягає в тому, що в разі statefull підходу враховується той факт, до якого саме потоку відноситься аналізований пакет, і результат (Стан) аналізу попередніх пакетів цього ж потоку, якщо даний пакет не перший. У разі якщо пакет перший - перевіряється, що він є коректним пакетом встановлення з'єднання. Слід також зазначити, що поняття «statefull» не цілком чітке і може мати різні градації з різним «станом», що призводить до різного балансу точність аналізу/ресурсомісткість/швидкість роботи [9, 10]. Один з варіантів градації можна бачити на рис1.6. Список рівнів обліку стану потоку, який там відбитий - наступний:

- аналіз окремих пакетів без урахування потоків і станів (Packet Based No State, PBNS).

- аналіз пакетів в рамках потоків (Packet Based Per Flow State, PBFS);

- аналіз повідомлень в рамках потоку (Message Based Per Flow State, MBFS), тобто проведена збірка IP-фрагментів в IP-пакети (IP нормалізація) і збірка TCP-сегментів в TCP-сеанси (TCP нормалізація) ;

- аналіз повідомлень в рамках протоколу (Message Based Per Protocol State, MBPS), тобто враховується стан автомата протоколу (Можливість приймати той чи інший тип повідомлень).

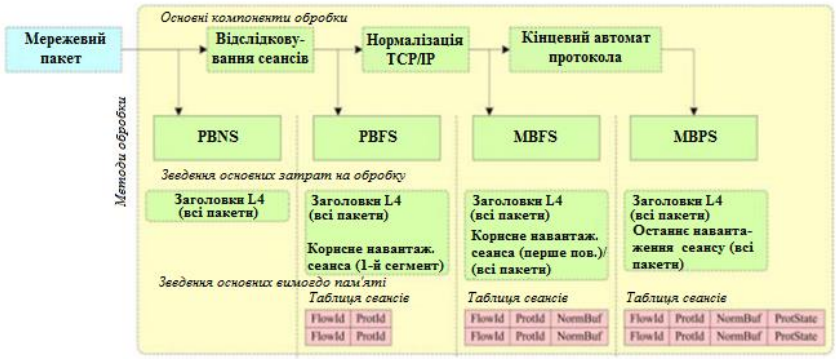


Рисунок 1.6 - Градації повноти обліку стану потоку.

Базові реалізації технології DPI часто ставляться до stateless-аналізу, тобто аналіз виконується на рівні окремих пакетів, стан між аналізом декількох пакетів одного мережевого потоку не зберігається. цього рівня точності вистачає для багатьох практичних застосувань і дозволяє значно економити ресурси.

У той же час, існують завдання, для яких такого рівня точності мало. Як приклади можна привести дві технології, які використовують statefull підхід – інспекція пакетів зі зберіганням стану (statefull packet inspection, SPI) і глибокий аналіз вмісту (deep content inspection, DCI).

1.4 Методи і алгоритми класифікації мережевого трафіку

В основі класифікації мережевого трафіку рис.1.7. лежить глибокий аналіз номерів портів пакетів на транспортному рівні (класифікація, заснована на портах), відновлення сигнатури протоколу з його корисного навантаження (класифікація, заснована на корисне навантаження), статистичних методів аналізу характеристик обміну пакетами між хостами і статистичних властивостей мережевого трафіку. Кожен з підходів має свої достоїнства і недоліками.

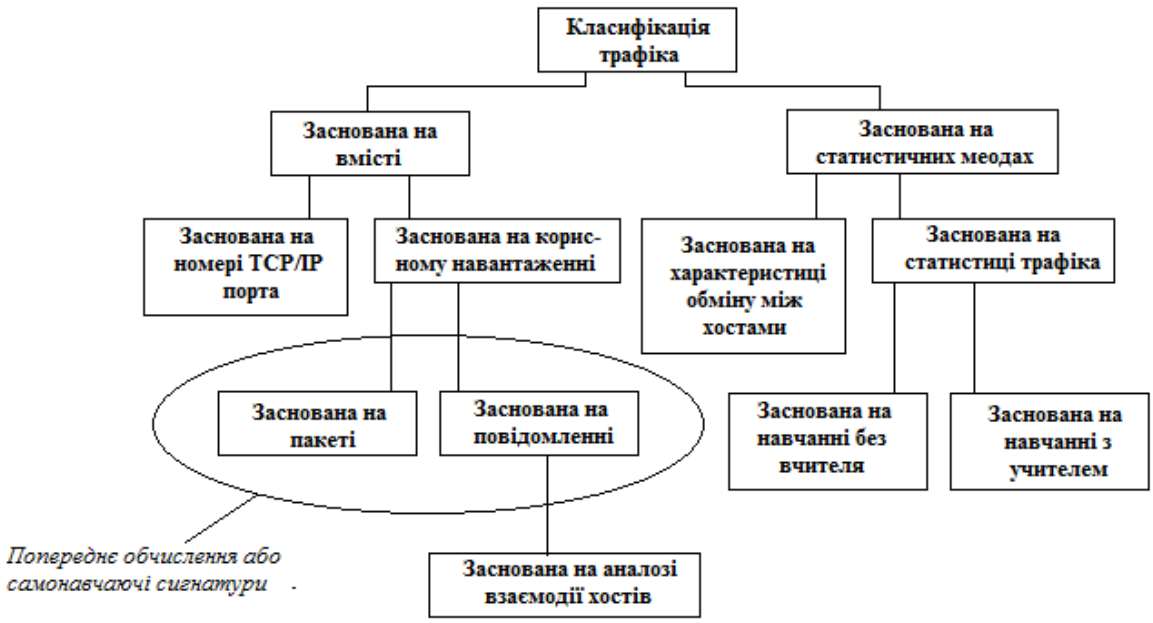


Рисунок 1.7 - Класифікації мережевого трафіку

Традиційні методи класифікації мережевого трафіку, засновані як на номерах портів, так і на інформаційному навантаженні, покладаються на пряме вивчення мережевих пакетів. Схема класифікації по портам перевіряє заголовки пакету. Аналізуються поля заголовка, з номерами портів джерела і одержувача, а потім визначається протокол додатка згідно зі списком зареєстрованих відомих номерів портів, який підтримується організацією IANA (Internet Assigned Numbers Authority).

Класифікація трафіку по портах інтегрована в більшість сучасних мережевих пристроїв і ПЗ. Це ефективний і швидкий підхід до ідентифікації протоколів додатків в перший час використання мережі Інтернет, коли велика частина додатків використовували їх стандартні номери портів, зареєстровані в IANA [5]. Проте, з того моменту, як отримали своє розвиток додатки на протоколі P2P. який найчастіше обирає довільні номери портів для того, щоб уникнути виявлення і фільтрації, метод класифікації по портам ставав все менше і менше точним.

В [11] була розширена робота з ідентифікації P2P трафіку шляхом об'єднання маркування номерів портів, підтвердження сигнатур протоколів і асоціювання хостів. Ці методики були застосовані для аналізу тенденції P2P трафіку в даних з магістральної лінії зв'язку. Результати показали, що частка P2P трафіку продовжувала збільшуватися, в той час як використання зареєстрованих портів навпаки, суттєво зменшилася. Таким чином, зменшення обсягу P2P трафіку, як це показав метод, заснований на номерах портів, виявилося хибним.

В [11] досліджувався трафік п'яти популярних P2P протоколів, включаючи BitTorrent, eDonkey. Gnutella. Kazaa. and Direct Connect, на основі розроблених сигнатур протоколів. Результати, отримані з набору даних Інтернет трафіку і VPN показали, що більшість трафіку BitTorrent і eDonkey використовує зареєстровані порти, в той час як більша частка трафіку Gnutella (34%), Direct Connect (38%) і Kazaa (72%) користується нестандартними номерами портів.

В [12] досліджувалася неточність класифікації трафіку за номерами портів шляхом ідентифікації і підрахунку різних типів помилок. Робота заснована на наборі даних з повним корисним навантаженням пакетів, зібраному в мережі Gbase

Ethernet в кампусі Genome, що включає кілька установ і споруд, які виконують роботи в області біології. Схема класифікації трафіку складалася з дев'яти методів, таких як аналіз номерів портів, аналіз заголовків пакетів, зіставлення сигнатур в навантаженні одного пакета, аналіз семантики протоколу в навантаженні одного пакета, зіставлення сигнатур протоколів в перших

К-байтах навантаження потоку, декодування протоколу в контрольних потоках, декодування протоколу у всіх потоках, аналіз історії взаємодій хостів. Результати показали, що класифікація трафіку по портах не тільки недооцінює трафік для одних класів (близько 20% трафіку BULK і 6% WWW трафіку), але також і переоцінює трафік інших класів (близько 0.5% трафіку INTERACTIVE). У загальному і цілому, класифікація по портах правильно ідентифікує приблизно 70% всього трафіку в наборі даних Genome. Наприклад, порт під номером 80 використовується для передачі веб – трафіку (HTTP), а порт 21 - для передачі файлів по протоколу FTP. Такий підхід дуже ефективний, оскільки включає в себе тільки лише поверхневий аналіз заголовка пакета і подальший пошук по списку цілочисельних значень.

Проте, широко поширена думка, що класифікація по портах на даний момент стає все більш неточною і ненадійною по ряду причин.

По-перше, деякі додатки, як наприклад FTP, створюють кілька з'єднань для однієї єдиної сесії, причому для контрольного з'єднання використовується порт за замовчуванням. а самі дані можуть передаватися через динамічно вибрані незареєстровані порти.

По-друге, деякі додатки могутній не мати зареєстрованого для них номера порту, що особливо проявляється у недавно створених додатків.

По-третє, деякі додатки можуть цілеспрямовано використовувати замість стандартного інші відомі порти, щоб обійти фаєрвол. Крім того вони можуть «обгортати» свій трафік в інші відомі протоколи (Технологія тунелювання).

По-четверте, широко поширена в мережі Інтернет технологія трансляції адрес NAT (Network Address Translation) через що скорочується кількості IPv4

адрес. Ця технологія заснована на зміні початкових номерів портів в пакеті у міру того, як вони проходять через шлюзи NAT.

Ранні спроби досліджень класифікації, заснованої на корисної навантаженні були конструювання бібліотеки сигнатур протоколів і проблеми масштабованості в глибокому аналізі пакетів. З цією метою в деяких роботах представлений набір сигнатур протоколів, які були вручну виділені з доступних специфікацій протоколів і наборів даних.

Однак, обчислювальні витрати глибокого аналізу пакетів значно вище, ніж при аналізі номерів портів. В результаті, були запропоновані нові підходи для швидкого і ефективного пошуку сигнатур протоколів в навантаженні пакетів, які були покликані дозволити використовувати класифікацію, засновану на навантаженні, в високошвидкісних мережах рис.1.8.

Як згадувалося вище, підходи, засновані на навантаженні, використовувалися, щоб показати неточність класифікації по портам. Зокрема, в [11] були розроблені сигнатури протоколів для п'яти P2P додатків, включаючи BitTorrent, eDonkey, Gnutella, Kazaa, and Direct Connec. на основі ручного аналізу як специфікацій протоколу, так і зібраних наборів даних.

Сигнатури були спроектовані для порівняння з першими кількома пакетами в поєднанні TCP і представляли собою фіксовані рядки на певної або різних позиціях всередині навантаження TCP пакетів.

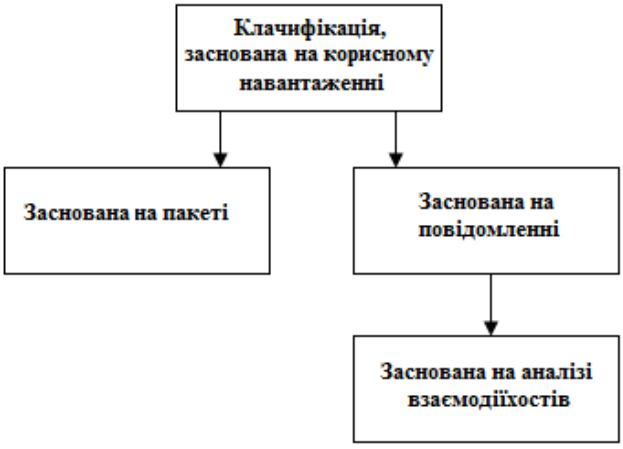


Рисунок 1.8 - Класифікація, заснована на корисному навантаження

В роботі використовувалися два набори даних для тестування сигнатур, які в підсумку генерували рідкісні помилково-позитивні помилки (Ідентифікуючи НЕ-P2P трафік як P2P) і менше 10% помилково-негативних рішень (ідентифікуючи P2P трафік що не-P2P). Як частина евристики для ідентифікації P2P трафіку, в [5-6] було виділено набір байтових послідовностей - сигнатур, які зіставлялися з контрольними пакетами і пакетами з даними восьми популярних P2P протоколів. Рядки були спроектовані для порівняння з першими 4-ій байтами пакетної навантаження, що пов'язано з обмеженнями в використаних наборах даних (де пакети були об'ємом 44 байта максимум). Як результат, набір сигнатур породжував порівняно висока кількість хибно-позитивних помилок.

В [12] в схемі класифікації також були застосовані різні методи аналізу навантаження, включаючи порівняння сигнатур протоколів і аналіз семантики протоколу в навантаженні одного пакета або перших К-байтах навантаження потоку. Використані сигнатури протоколів або семантика були представлені в роботі, але стверджується, що спочатку використовувалися добре відомі сигнатури протоколів, а вже потім нові сигнатури, створені в

Внаслідок трудомісткого процесу аналізу непомічених наборів даних. З цих досліджень, можна побачити, що отримання точних сигнатур протоколів є далеко не простим завданням. Ручний процес їх виділення дуже трудомісткий, а також повинні бути зроблені деякі компроміси щодо доступних даних, знань і якості підсумкових сигнатур.

Вищезазначені сигнатури протоколів. виділені вручну, представляюшіе собою фіксовані рядки на певній або різних позиціях в навантаженні, могли бути описані за допомогою простих регулярних виразів. Більш того, вони широко застосовуються в реальних системах, таких як Cisco IPS, Snort IDS, Bro IDS, 17-filter, IPP2P і т.д.

Інший традиційною схемою є вивчення даних рівня додатки в мережевих пакетах, зазвичай зване Deep Packet Inspection (DPI). Отримуючи доступ до навантаження пакета, класифікатор може або спробувати декодувати і затвердити повідомлення додатків (використовуючи відомі аналізатори мережевих протоколів

EtherealAVireshark або просто зіставити інформацію в навантаженні і сигнатури протоколів. Він широко використовується як в комерційних продуктах. так і в проектах з відкритим вихідним кодом на кшталт 17-filter і IPP2P. Існують різні типи сигнатур протоколів, починаючи простими незмінними рядками і закінчуючи складними регулярними виразами.

Підхід, заснований на інформаційній навантаженні пакета, вважається найбільш точної і надійною схемою класифікації трафіку. Як би там не було, у нього є певні обмеження і складності.

В першу чергу, обчислювальні витрати такого методу є головним перешкодою до його застосування в високошвидкісних мережевих з'єднаннях. З одного боку, глибока перевірка пакета вимагає великих обчислювальних ресурсів для вилучення його вмісту.

Але з іншого боку, процес декодування протоколу і зіставлення сигнатур привносить навіть більшу складність обчислення. Дослідники ріпової подолати цю перешкоду шляхом впровадження ефективних методів порівняння сигнатур [13].

Другий складністю є отримання апріорної інформації про мережеві протоколах (формати повідомлень або сигнатури протоколів), що представляє собою непросту задачу. На практиці ця інформація зазвичай виділяється професіоналами шляхом ручного аналізу специфікацій протоколу (якщо доступні) або пакетних Трейсі. що займає багато часу і вимагає автоматизації [14]. Нарешті, інформаційний підхід не застосовний, коли немає доступу до навантаження пакета, як. наприклад, в разі класифікації зашифрованого трафіку. Ранні спроби досліджень класифікації, заснованої н корисного навантаження були присвячені конструюванню бібліотеки сигнатур протоколів і проблеми масштабованості в глибокому аналізі пакетів. З одного боку, деякі роботи представили набір сигнатур протоколів, які були вручну виділені з доступних специфікацій глибокого аналізу пакетів значно вище, ніж при аналізі номерів портів. Таким чином, були запропоновані нові підходи для швидкого і ефективного пошуку сигнатур протоколів в навантаженні пакетів, які були покликані дозволити використовувати класифікацію, засновану на навантаженні, в високошвидкісних мережах.

Технологія класифікації на основі корисного навантаження пакета DPI з'явилася в результаті заміни класифікації потоків по портам. Вважалося, що в ній будуть усунені недоліки, і загальна точність класифікації підвищиться. DPI класифікатори прийнято розглядати, розділяючи на типи по зростанню ступеня аналізу і обробки вмісту пакетів, і вимог до пам'яті:

- до першого типу відносяться сигнатурні класифікатори, мета яких полягає у пошуку збігів за деякими маскам, або сигнатурам в корисного навантаження прикладного рівня. Більшість пакетів різних протоколів прикладного рівня починаються з певного заголовка, на основі якого можна створити сигнатуру для гарантованого визначення протоколу або додатка;

- на другому рівні DPI класифікаторів вводиться синтаксична перевірка, яка є покращеною версією першого типу і дозволяє не тільки визначати до якого додатку відноситься пакет, але і перевіряти правильність переданого пакета синтаксично. Припустимо HTTP пакет має чітку структуру, де кожен пакет починається з команди (GET, POST, PUT і т.д.), за якими слідують заголовки, і тіло повідомлення, яке повинно бути укладено в певні теги. Синтаксична перевірка забезпечує правильність сформованого пакета;

- третій рівень вводить так званий протокол відповідності, контролюючий вміст пакету на рівні сеансу - тобто при HTTP GET запиті до сервера. Відповіддю буде правильний статут відповіді від сервера. Завдяки протоколу відповідності з'являється можливість перевіряти відповідність поведінку з дійсною специфікацією різних протоколів;

- останній, четвертий рівень визначає семантику даних. З допомогою неї можна перевірити все те. що було введено на попередніх рівнях, а також додатково перевірити, чи дійсно об'єкт, переданий по даному протоколі є тим чим він оголошений. DPI, що працюють на четвертому рівні можуть, наприклад, перевіряти, чи дійсно вміст в тегах зображення протоколу HTTP є зображенням і т.д.

Аналіз корисного навантаження рівня додатки дозволяє розширити можливості класифікації, забезпечуючи перевірку семантики даних, протоколу

відповідності та багато іншого. Однак це накладає додаткові вимоги до продуктивності і пам'яті при обробці трафіку. DPI класифікатори повинні отримувати своєчасні оновлення при появі нових, і внесення змін в існуючі протоколи і додатки, що неможливо без великого штату співробітників, що в свою чергу призводить до значних фінансових витрат. Слід так само враховувати про те. що даний підхід може бути неможливий в поточних реаліях з великими обсягами зашифрованого і тунелюючого трафіку.

В [15] показано, що класифікація, заснована на корисне навантаження, для P2P- трафіку (шляхом дослідження сигнатур трафіку для прикладного рівня) може скоротити помилки першого і другого роду до 5% від загального числа байт більшості досліджуваних P2P- протоколів.

Сучасні практичні розробки (NetPDL, NBAR, SML, BinPac) повністю не вкладаються в цю таксономію, оскільки одна і та ж технологія може належати кільком категоріям. В [14] використовуючи технології, що поєднують порти і корисне навантаження, визначали мережеві додатки. Процедура класифікації починалася з вивчення номера порту потоку. Якщо використовувався невідомий порт, потік передавався на наступний етап. На другому етапі перевірявся перший пакет на наявність відомої сигнатури.

Якщо вона не була знайдена, тоді пакет проглядався на наявність відомого протоколу. Якщо ця перевірка закінчувалася невдачею, проглядався перший кілобайт потоку на наявність сигнатури протоколу. Потоки, що залишилися некласифікованими після цього рівня, зажадають перегляду всієї корисного навантаження потоку. Результати показали, що при наявності інформації про порте правильно класифікувати можливо 69% від загальної кількості байт.

Включаючи інформацію, отриману з першого кілобайти кожного потоку, точність збільшується майже до 79%. Найвища точність (майже 100%) може бути досягнута тільки вивченням корисного навантаження залишилися некласифікованих потоків.

Хоча дослідження корисного навантаження уникає фіксованих номерів портів, воно накладає суттєві складності і завантажує пристрій ідентифікації

трафіку. Такий пристрій має володіти великими знаннями про семантику прикладних протоколів, має бути достатньо потужним, щоб виконувати одночасно аналіз потенційно великого числа потоків. Цей підхід може бути утруднений або неможливий, коли мова йде про запатентованих протоколах або зашифрованому трафіку.

Моделювання та характеристика мережевого трафіку вивчалася довгий час з найперших років використання Інтернету. Для прикладу, в дослідженнях виділяють аналітичну модель трафіку для переліку додатків, таких як SMTP, Telnet, FTP. Були визначені випадкові змінні для опису емпіричної вимірювань TCP з'єднання, на зразок байтів джерела, байтів відповідача, тривалості байтів з'єднання, байтів сесії, потоків байтів; потім їх розподіл виділялося в математичній формі, яка найкраще підходила до випадкових змінних. В [12] було запропоновано параметризрвані підхід до профілізації потоків, в якому виділялися метрики потоків, залежать від типу додатка, включаючи тайм-аут потоку, особливості агрегації трафіку, мережеве оточення і використання інформації на транспортному рівні і рівні додатків. Також в попередніх роботах був представлений перелік глибоких аналізів певних мережевих додатків, включаючи системи онлайн-чатів, онлайн гри, Skvpe, YouTube.

Через обмеженою традиційних методів, в останні роки з'являлося безліч досліджень і робіт по альтернативних підходів до точної і ефективної класифікації мережевого трафіку. найбільш перспективним напрямком є статистична класифікація, яка використовує статистичні параметри потоків трафіку. Суть в тому. що трафік, генерується різними типами додатків, носить свої відмінні характеристики, що відображають унікальне веління і внутрішню природу додатків. Характеристики потоку можуть бути виражені у вигляді векторів ознак і статистичні методи або техніки МН можуть бути застосовані для класифікації.

При наявності повного і поміченого тренувального набору даних, дуже просто побудувати класифікатор, використовуючи нові технології машинного навчання. Деякі алгоритми навчання були застосовані для цієї мети в попередніх роботах. В [16] було виділено статистичний метод розмічування зі зведених

показників {dstIP, dstPort} і досліджені різної статистики трафіку, включаючи рівень пакета (середнє значення, дисперсія і середньоквадратичне значення розміру пакета), рівень потоку (середнє значення, дисперсія тривалості потоку, обсяг даних, кількість пакетів в потоці), рівень TC'P з'єднання, всередині потоку (статистика інтервалів між пакетами в потоці), і властивості сукупності потоків (обсяг даних в контрольних потоках і потоках даних). Для класифікації були застосовані два алгоритми машинного навчання з учителем: Лінійний Дискримінантний Аналіз і метод Найближчих Сусідів. В [16] було запропоновано метод класифікації трафіку, заснований на гістограмах BCP - рівня. Були застосовані процеси суміші Діріхле для моделювання емпіричних гістограм, виділених зі зведених показників із загальним конкретним BCP - префіксом.

В [17] представлений набір даних для дослідження статистичної класифікації трафіку. Було визначено повний набір властивостей трафіку, що складається з 248 характеристик для опису потоків трафіку TCP. Ці ознаки містили набір загальних статистик про розмір пакетів, обсязі даних і інтервал між пакетами; також вони включали велику кількість вимірювань, специфічних для TCP, такі як статистика прапорів SYN. RST і ACK. Оцінки RTP. розмірів сегментів і загальна кількість повторних передач. Деякі з ознак (перші десять частотних складових перетворення Фур'є міжпакетних інтервалів) було дуже важко обчислювати, а тому вони можуть не використовуватися в класифікації на високошвидкісних мережах.

До класифікації мережевого трафіку, заснованих на статистичних методах може бути застосовано два типи алгоритмів рис.1.9, поведінкові і статистичні алгоритми мережевого і транспортного рівнів.

Концепція поведінкового алгоритму полягає в оцінці, які саме додатки створюють певні потоки трафіку. Аналізуючи взаємодіючі хости, в рамках мережі, можна визначити, які види додатків запущені на хості. Співвідношення між класом трафіку і його поведінковими статистичними властивостями були описані в [13], в яких для ряду спеціальних TCP-додатків проведено аналіз і побудовані емпіричні моделі характеристик з'єднання.

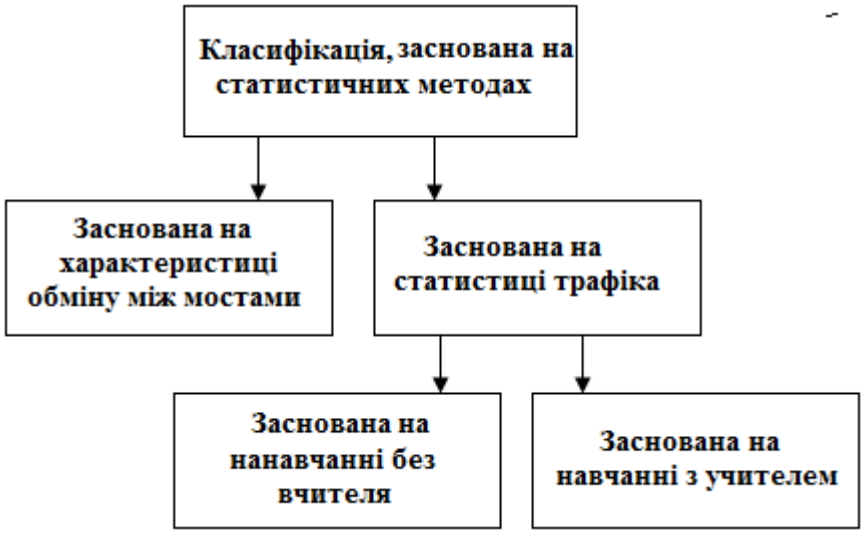


Рисунок 1.9 - Класифікація, заснована на статистичних методах

Статистичні методи спираються на статистичні характеристики трафіку для ідентифікації додатка і базуються на унікальності статистичних характеристик для певних класів додатків, що і дозволяє розділити різні вихідні додатки. Статистичні методи, засновані на статистиці трафіку діляться на дві групи: методи класифікації або навчання з учителем і методи кластеризації або навчання без учителя.

Типова схема класифікації за статистикою трафіку складається з двох окремих етапів. Перший - навчання в офлайні, в якому тренувальний набір даних надходить на вхід якого навчають алгоритму, а на виході виходить класифікатор (імовірнісна модель або набір правил класифікації). Другий етап - класифікація, в якому класифікатор використовується для передбачення класу додатка в тестовому трафіку.

Залежно від того, чи є тренувальний набір даних позначеним чи ні. поділяють два основні методи, які застосовуються на першому етапі. Алгоритми навчання з учителем призначені для складання класифікаційної моделі на основі тренувальних наборів даних. В противагу цьому, алгоритми навчання без вчителя (або кластеризація) намагаються виявити внутрішні структури в наборі непомічених даних і розділити їх на кластери відповідно до деякими ознаками. Тут метою є створення чистих кластерів за типами програм так, щоб результативні кластери

могли бути пов'язані з реальними класами додатків і на їх основі могли бути побудовані підсумкові класифікатори.

Було показано, що рівень точності, який досягається статистичними алгоритмами навчання з учителем, може бути таким же високим, як і у класифікаторів, заснованих на навантаженні пакетів, але при менших обчислювальних витратах. Порівняння розглянутих підходів класифікації представлено в табл.1.1.

В [18] були застосовані і оцінені методи Байеса для класифікації того набору даних, представлених в [17].

Зокрема, алгоритм Naive Bayes був використаний для навчання класифікаторів, і два методи для їх удосконалення, включаючи модель Kernel Estimation in Naive Bayes і Fast Correlation-Based Filter для вибору ознак.

Таблиця 1.1 - Порівняння сучасних підходів класифікації

	Заснований на номерах портів	Заснований на навантаженні пакету	Заснований на статистиці потоків
Точність	Низька	Висока	Висока
Складність виділення ознак	Низька	Висока (DPI)	Залежить від набора ознак
Складність класифікації	Низька	Висока	Нижче середнього
Закодований трафік	Так	НІ	Так
Апріорна інформація	Список портів	Сигнатури	Помічені дані
Відкидання невідомого трафіка	Так	Так	Шум

Результати експерименту показали, що класифікатори Naive Bayes могли досягти 65% і 95% точності по потокам без і з удосконаленнями відповідно. Була проведена додаткова оцінка на тренувальному наборі даних, зібраному через один рік після навчального, щоб показати тимчасову стабільність поліпшених класифікаторів. В [14] робота була розширена за рахунок застосування класифікаторів з Байєсовою Нейронною Мережею для збільшення точності до 99%. В [16] були запропоновані «відбитки пальців» протоколів для моделювання деяких

певних ранніх під-потокових властивостей, таких як розмір пакету, міжпакетні інтервал і послідовність початкового обміну пакетами в потоці. Відбитки певного протоколу представляє собою згладжений PDF. описує емпіричне розподіл пар (розмір пакету, межпакетное інтервал) кожних перших кількох пакетів в потоці протоколу. Порівняння відбитків з тестовими потоками засноване на простому методі відношення правдоподібності. Були виділені відбитки для трьох протоколів, включаючи HTTP, SMTP і POP3. Точність їх виявлення становила 91%, 94% і 94% відповідно, а кількість хибно-позитивних помилок - 6%, 3% і 3%. Також в [12] протоколів була застосована для виявлення Тунелів HTTP.

В [17] були використані однокласових алгоритм Машин Опорних Векторів (SVM) для навчання класифікатора трафіку на основі тільки лише спостереження за довжиною потоку і напрямком перших декількох пакетів. Для кожного протоколу спочатку класифікатор | SVM навчався за даними одного певного протоколу, а потім поліпшувався з використанням інших протоколів. На додаток, для вирішення конфліктів в класифікації, був навчений багатокласових SVM класифікатор.

В [11] виділялися показові під-потокові характеристики для класифікації довгих інтерактивних потоків, початок яких могло бути пропущено і припущення повинні бути зроблені в будь-який момент на основі самих недавніх пакетів потоку. Був запропонований спосіб навчання класифікатора з використанням властивостей трафіку, виділених з ковзних вікон (до 25 пакетів в довжину) потоків. Для ситуації, коли спостерігатися можуть тільки потоки, що йдуть в одному напрямку, в [11] було запропоновано внести штучні пів-потоки (віддзеркалювати копії спостережуваних пів-потоків) в навчальний набір даних. Більш того, в [14, 15] проводилася кластеризація пів-потоків для вибору найбільш показових пів-потоків для навчання. У цих роботах для класифікації використовувалися алгоритми Naïve Bayes і C4.5.

Також було проведено порівняння і оцінка різних статистичних класифікаторів трафіку з урахуванням алгоритмів навчання, особливостей трафіку, проблем реалізації, труднощів масштабування і т.д. В проаналізованих роботах

представлено порівняння продуктивності різних алгоритмів машинного навчання (Naive Bayes, Naive Bayes with Kernel Estimation, C4.5 decision tree, Bayesian Network, and Naive Bayes Tree) для класифікації трафіку з великою увагою до вибору ознак і продуктивності. В [18] також представлена оцінка класифікації ігрового трафіку. В [18] порівнювався статистичний метод класифікації з методом за номерами портів і з новим методом, заснованим на поведінці хостів (ВИХС [72]). В [19] ідентифікувалися номера портів, довжина перших кількох пакетів і дискретизація властивостей на трьох критичних джерелах дискримінантності для класифікації мережевого трафіку. В [20] аналізувалася стабільність інформації про статистичні властивості потоку з урахуванням часу і місця спостереження. В [18,20] оцінювалася статистична класифікація трафіку з точки зору ADSL провайдерів, де класифікатори були протестовані на наборах даних з різних мереж. також було запропоновано використовувати рівень впевненості, виведений з алгоритму C4.5 для дослідження трафіку, який не зміг ідентифікувати себе за допомогою DBI.

Видно, що попередні роботи показали застосовність статистичних класифікаторів трафіку в різних наглядових точках мережі Інтернет, незалежно від того, розташовані вони всередині автоматизованих систем (АС) або ж на кордоні кількох АС. Однак такі класифікатори чутливі до мережі, а тому класифікатори, навчені з даними з однієї мережі, можуть бути неефективними з даними іншої мережі, і тому потребуватимуть перенавчання з новим трафіком.

Для ефективної ідентифікації додатків необхідно вибрати невелике кількість ознак (атрибутів), які повинні бути повними і придатними для класифікації. На рис.1.10. представлена схема класифікації.

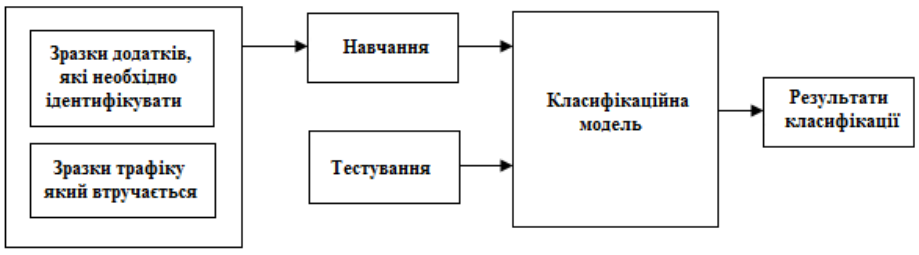


Рисунок 1.10 - Навчання і тестування класифікатора

Попередньо необхідно визначити відповідні зразки додатків для трафіку, які повинні бути ідентифіковані в майбутньому, в тому числі і для втручається трафіку сторонніх додатків.

На наступному етапі обидва компонента трафіку об'єднуються. Результат об'єднання включає в себе як зразки цікавить нас додатки так і зразки інших додатків. Для формування ознак класифікації необхідно визначити статистичні характеристики потоків трафіку, які необхідні для навчання класифікатора.

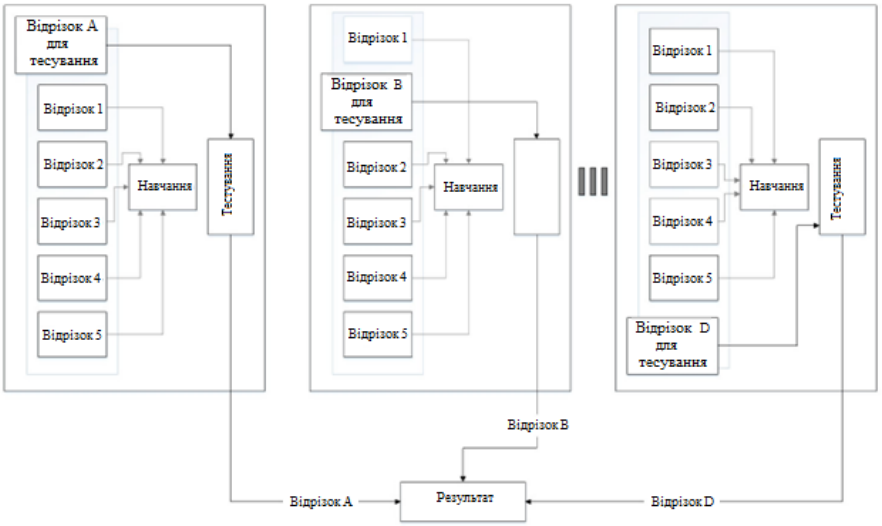


Рисунок 1.11 – Перехресна перевірка

Для оцінки точності під час навчання може бути використана перехресна перевірка (крос-валідація). Суть перехресної перевірки складається у разі необхідності розділення даних на n-частин.

Навчання відбувається на *n*-1 частинах даних, а на що залишилися даних відбувається тестування, таким чином забезпечується точність класифікації. Цей процес повторюється багато разів, п раз. В результаті кожна з n-частин потрапить на тестування *n* раз і точність алгоритму може бути оцінена п раз. В підсумку, як показано на рис.1.11 (де project-це відрізки часу для навчання), розраховується середня точність і виходить результат перехресної перевірки для класифікації.

Для кращого і точного результату вихідний набір даних повинен містити змішаний трафік, який був знятий в різний час і в різних точках мережі.

2 ФОРМУВАННЯ ВИХІДНИХ ДАНИХ ТА АНАЛІЗ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ, ЩО ВИНИКАЮТЬ ПРИ РОЗРОБЦІ СИСТЕМИ КЛАСИФІКАЦІЇ ТРАФІКУ

2.1 Формування вихідних даних і аналіз програмного забезпечення

Однією з головних проблем, яку потрібно вирішити при розробці системи класифікації трафіку використовуючи алгоритми машинного навчання (МН) є формування вихідних даних і використовується для цих цілей програмне забезпечення (ПЗ). Вихідні дані представляють собою зразки пакетів, класифікованих за додатками їх створив. В даний час не існує єдиного набору вихідних даних, які є стандартом в області класифікації трафіку, як не існує і єдиного підходу до їх отримання. Водночас точність алгоритмів МН безпосередньо залежить від обсягу, якості та репрезентативності набору вихідних даних, що використовувалися в процесі навчання. Тому отримання якісного набору вихідних даних є важливим завданням [21].

Розглянемо особливості отримання вихідних даних на основі захоплення трафіку за допомогою програми або програмно-апаратного комплексу сниффер, призначеного для перехоплення трафіку. Крім базового функціоналу багато програм-сніфери дозволяють здійснювати фільтрацію потоків трафіку, відновлювати TCP сесії та ін. за допомогою пакетного сніффера Арбітр спільно з бібліотекою НЬсар [21].

Розглянемо архітектуру і реалізацію програмного забезпечення для отримання вихідних даних. Tracedump є консольної утилітою, що працює під управлінням 32-х бітових операційних систем сімейства Linux. Оскільки tracedump є відкритим програмним забезпеченням існує кілька модифікацій, fork-ів вихідної програми. в яких присутній функціонал, відсутній в оригінальному додатку. дані модифікації розробляються і підтримуються незалежними розробниками. Розглянемо особливості використання fork програми під назвою tracedump64,

дозволяє запускати програму в 64-х бітному оточенні, а так само володіє підвищеною стабільністю, в порівнянні з оригінальною програмою.

Для того щоб розглянути архітектуру програми проаналізуємо, послідовність відкриття і посилки TCP з'єднання в Linux. Операційна система надає API для інтернет-з'єднань за допомогою системних викликів - socket (), connect () і send (). В результаті, на початку роботи додаток використовує функцію socket () для отримання унікального з'єднання. потім, викликається функція connect (), з адресою віддаленого вузла, і нарешті для передачі даних, може бути використаний системний виклик send ().

При проектуванні пакетного сніффер однієї програми слід враховувати дві умови.

По-перше (завдання А) - додаток не бере участі в побудові заголовків на рівні мережного і транспортного протоколу - це є завданням операційної системи. Отже, недостатньо перехопити лише дані, передані як аргументи на системному виклику.

По-друге (завдання В), системний виклик на отримання з'єднання connect () згенерує кілька пакетів ще до того, як з'єднання буде отримано. В

Внаслідок, сніффер повинен почати захоплення пакетів ще до того як виклик виповниться в ядрі ОС.

Tracedump розділений на три функціональних модуля, реалізованих у вигляді потоків: ptrace, pcap. і збирач сміття (GC- garbage collection). модуль ptrace підключається до всіх потоків обраного процесу і. використовуючи функцію ptraceQ Linux а створює список всіх локальних TCP і UDP портів, відкритих перехоплює всі пакети з усіх мережевих інтерфейсів, причому робить це на рівні ядра, вирішуючи завдання А. розглянуте раніше. Всякий раз. коли змінюється список портів до модуля захоплення pcap застосовується BPF фільтр, таким чином, що всі пакети, які не належать оскільки він розглядався з додатком, ігноруються. BPF фільтр оновлюється до того, як ядро ОС виконає системний виклик - тим самим, вирішуючи завдання В. Завдання «збирача сміття» полягає у визначенні більш що не використовуються портів. Кожну хвилину він збирає список усіх

відкритих системних з'єднань, і оновлює список, отриманий модулем ptrace. Архітектура програми представлена на рис. 2.1.

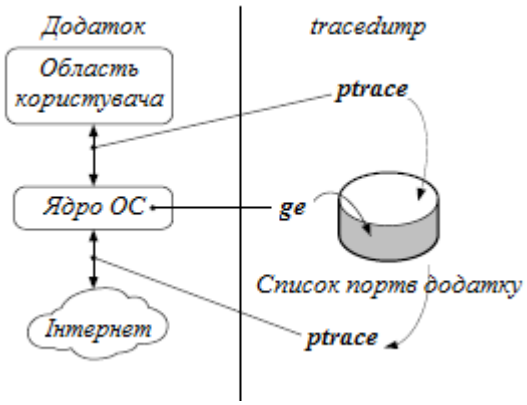


Рисунок 2.1 - Архітектура програми tracedump

Список портів формується з ядра ОС. і використовується для захоплення пакетів. Складальник сміття періодично очищає список.

Модуль ptrace відстежує лише три системних виклику: bind (), connect (), і sendto (). Результати роботи модуля представлені на рис. 2.2. шляхом аналізу вихідного коду ядра Linux було доведено те. що пропонована архітектура програми є достатньою для того, щоб жоден пакет не був загублений. Всі захоплені пакети будуть належати тільки клієнту ctorrent.

Для UDP і TCP серверів, програму слід використовувати bind () для настройки номера локального порту. Для клієнтських програм потрібно здійснити дзвінок connect або sendto (). При цьому можливо, що локальний порт ще не присвоєно, і ядро виконає операцію присвоювання автоматично. Однак, через обмеження (Б) це являється небажаної ситуацією, в цьому випадку tracedump розбиває системний виклик, змушуючи спочатку використовувати bind (), що і реалізується шляхом ін'єкції машинного коду в спостережуваний процес.

Результати застосування програмного забезпечення для отримання необхідних вихідних даних була використана мережу рис.2.3, що складається з одного комп'ютера під управлінням Windows 7 – локальної робочої станції, на якому була запущена віртуальна машина з гостьової операційною системою Ubuntu

16.04 - віртуальної робочої станцією, необхідної для функціонування програми tracedump.

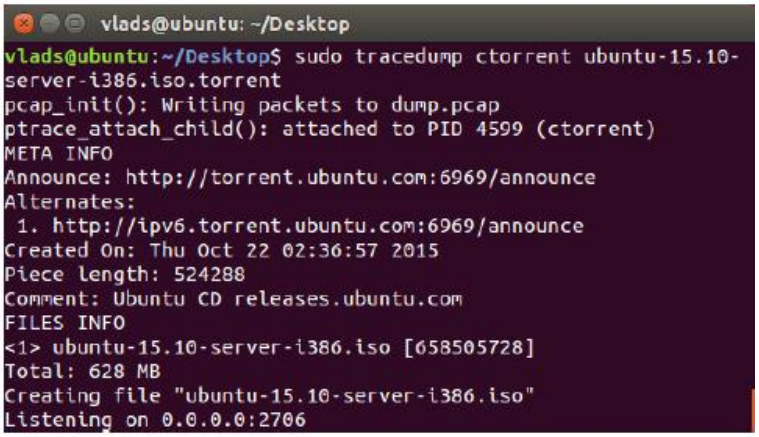


Рисунок 2.2 - Процес роботи tracedump на прикладі скачування клієнта Ubuntu через P2P-мережу.

На локальної робочої станції додатково був запущений сервер бази даних PostgreSQL, необхідний в подальшому при функціонуванні системи класифікації трафіку як в режимі реального часу, так і в автономному режимі, тобто при класифікації вже захопленого і збереженого в файл трафіку в форматі «PCAP». Підключення до мережі Інтернет здійснювалося через роутер.

У представленій мережі перехоплювати наступний трафік [22]:

- FTP-сервер використовувався для захоплення трафіку протоколів FTP-data і FTP- Command. При скачуванні був обраний пасивний режим роботи, при якому використовувався 21-й порт для команд і 20-й порт для передачі даних. Трафік даних протоколів був об'єднаний в клас «FTP»;

- WEB-сервер використовувався для захоплення трафіку HTTP, HTTPS використовують, як правило 80 і 443 порти. Дампи було присвоєно клас «WEB»;

- MAIL-сервер використовувався для захоплення трафіку SMTP, IMAP. Для додатків SMTP, IMAP використовувалися різні поштові служби, поштові клієнти - так як реалізація роботи з протоколом в рамках стандарту може відрізнятися [IMAP, SMTP]. Додатків був привласнений клас - «MAIL»;

- сервер управління дозволяє здійснювати управління масивом або кластером фізичних або віртуальних серверів. Взаємодія з сервером здійснювалося по захищеному протоколу SSH. Його перехоплення здійснювався захопленням трафіку програми терміналу на віртуальній робочої станції. Присвоєний клас - «SSH»;

- дистанційна робоча станція використовувалася для захоплення трафіку додатки SKYPE. У своїй роботі Skype використовує протоколи транспортного рівня TCP - для встановлення захищеного з'єднання, так і UDP - для аудіо- та відео-дзвінків. У разі відсутності довіри іншій стороні останні версії Skype дозволяють здійснювати всі з'єднання примусово використовуючи TCP. Трафік знімався у всіх режимах, як для аудіо-, так і для відео- дзвінків. Присвоєний клас - «Skype».

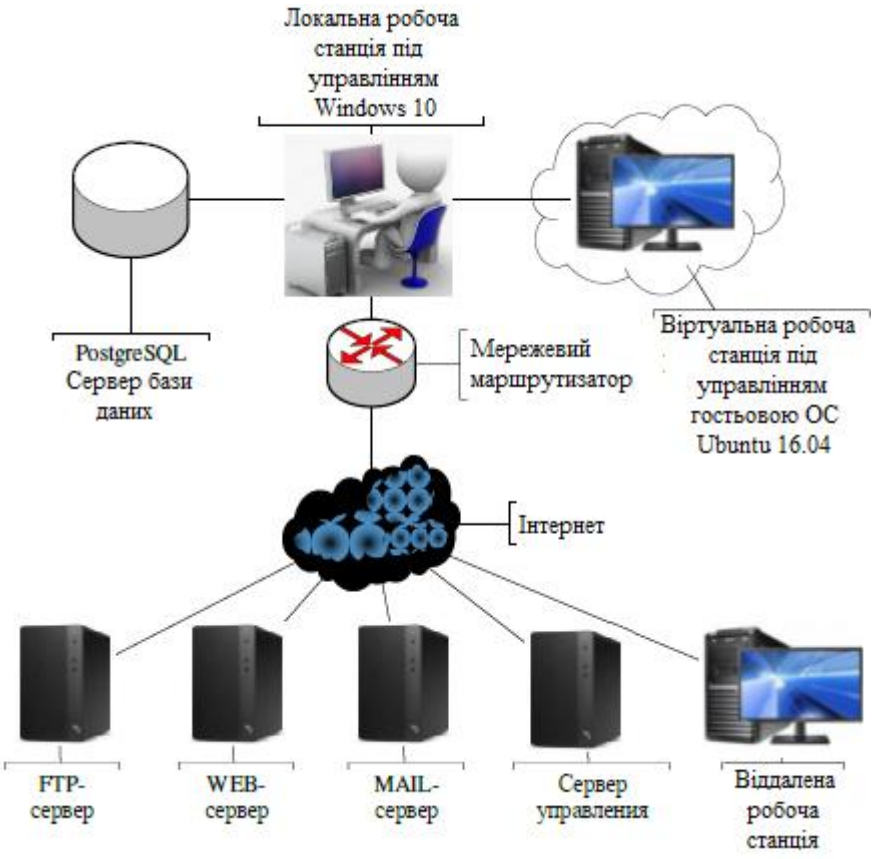


Рисунок 2.3 – Топологія мережі

Отримані в результаті вимірів дампів трафіку, наведені в табл.2.1.

Для коректної реалізації машинного навчання слід попередньо вирішити завдання вибору оптимальних атрибутів і скорочення їх кількості, а також досліджувати можливості застосування методів кластеризації для попереднього виділення окремих груп протоколів. В якості первинних атрибутів були обрані параметри мережевих потоків протоколів мережевого (TCP-, UDP-) і транспортного (IP) рівнів, представлені в табл.2.3.

Таблиця 2.2 - Кількість потоків отриманого трафіку по класам додатків

Кількість потоків мережевого трафіка по групам						Загальна кількість потоків
WEB, (HTTP, HTTPS)	MAIL (SMTP, IMAP)	FTP (FTP-DATA, FTP-COMMANDS)	SSH	SKYPE	P2P	
5054	3360	3470	5824	3737	6992	28437

При первинному виборі атрибутів було вибрано максимальне кількість атрибутів, значення яких тим чи іншим чином залежать від типу додатки, який є їх джерелом. Однак дослідження показали, що ні все з них слід використовувати в моделі класифікатора, так як інформаційний виграш від їх використання незначний. Для цілей виділення атрибутів був використаний метод ранжирування на основі інформаційного виграшу InfoGain [23].

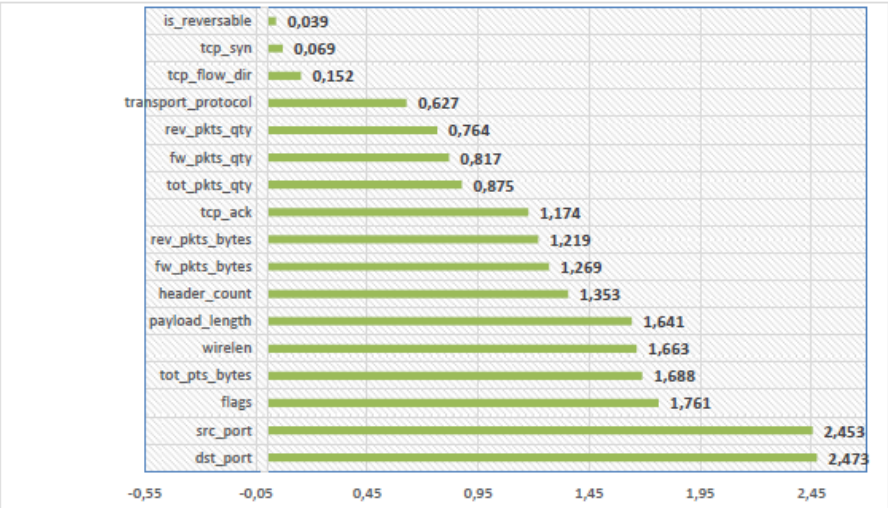


Рисунок 2.4 - Результат застосування алгоритму виділення атрибутів InfoGain

Результати застосування алгоритму InfoGain, з використанням 10-ти кратною перехресної перевірки представлені на рис. 2.4. Атрибути розташовані в послідовності від гіршого до кращого.

Таблиця 2.3 - Першочергові вибрані атрибути класифікації

Атрибут	Опис
classname	Назва укрупненого класу протоколу (WEB. MAIL. FTP н т.д.) класифікованого трафіку, яке буде використовуватися при створенні моделі класифікатора.
tot_pkts_qty	Загальна кількість пакетів в даному потоці в обох напрямках.
tot_pkts_bytes	Загальний розмір в байтах всіх пакетів в даному потоці в обох напрямках.
rev_pkts_qty	Кількість пакетів потоку в зворотному напрямку в разі, якщо потік двонаправлений.
rev_pkts_bytes	Розмір в байтах всіх пакетів потоку в зворотному напрямку.
fw_pkts_qty	Кількість пакетів потоку в прямому напрямку.
fv_pkts_bytes	Розмір в байтах всіх пакетів потоку в прямому напрямку
is_reversable	Булева змінна, яка відображає чи є даний потік двонаправленим.
transport_protocol	Протокол транспортного рівня (TCP-. або UDP-)
src_port	Протокол транспортного рівня джерела(як для TCP, так і для UDP)
dst_port	Порт адресата
wirelen	Вихідна довжина всіх пакетів потоку в фізичному каналі, поділена на загальну кількість пакетів
header_count	Кількість всіх заголовків всіх пакетів поділене кількість пакетів
Tcp_syn	Відсоток пакетів з прапором SYN протоколу транспорту рівня TCP. У разі, якщо використовується UDP його значення дорівнює GAP- відстані в байтах між заголовків і корисним навантаженням пакета, поділене на кількість пакетів.
Tcp_ask	Відсоток пакетів з прапором ACK TCP-протоколу, для UDP береться GAP OFFSET - відстань від початку пакета до кінця заголовків, поділене на кількість пакетів.
flags	Середня кількість прапорів TCP-протоколу, для UDP береться середня кількість заголовків.
payload_length	Середній розмір корисного навантаження протоколу транспортного рівня в потоці.
tcp_flow_dir	Булева змінна, яка визначає напрямок потоку. Якщо 1, то потік є вихідним, якщо -1 - вхідним.
is_fragment	Відсоток фрагментованих потоків.
hlen	Кількість заголовків протоколу IP.

З представлених результатів видно, що атрибути порту відправника і джерела вносять великий інформаційний виграш і їх слід використовувати при класифікації трафіку статистичними методами.

Після 14-го рангу відбувається різке падіння інформаційного виграшу. Аналізуючи значення і діапазон зміни даних було встановлено. Що атрибути tcp_syn, tcp_flow_dir, is_reversable мають незначний виграш показника інформаційного виграшу, і їх можна не використовувати в надалі.

Порівняльний аналіз існуючих сніфферов Wireshark, Libtrace, Утиліт Gt, nDPI, показав переваги використання утиліти tracedump як пакетного сніффер, що дозволяє ефективно перехоплювати дані.

Запропонована архітектура і реалізовано ПР для формування вихідних даних в задачах класифікації різних додатків.

На базі реалізованого програмно-апаратного комплексу створена експериментальна база даних для завдання класифікації додатків трафіку WEB (http, https), mail (smtp, imap), FTP (FTP -data. FTP -commands), SSH, Skype, P2P.

2.2 Оцінка алгоритмів виділення інформативних ознак та їх порівняння

Проведемо порівняльний аналіз та вибір алгоритмів виділення ознак атрибутів класифікації на прикладі програм FTP, Web, Mail, SSH, Skype на етапі навчання (створення моделі класифікатора), а також здійснимо порівняльну оцінку ефективності алгоритмів класифікації методами МН на етапі тестування.

Для отримання необхідних вихідних даних використовувалася мережу структура якої показана на рис. 2.3. що складається з одного комп'ютера під управлінням Windows 7 - локальної робочої станції з віртуальною машиною під керуванням гостьовий операційної системи Ubuntu 16.04.

Отримані в результаті вимірів дампи трафіку наведені в табл.2.3.

Для аналізу було підготовлено два набори даних, що містять мережеві пакети, розбиті по потокам. Класифіковані потоки були розділені наступним

чином: 66% від вихідних даних використовувалися як навчальний набір, решта 34% - для тестування і його оцінки.

Для першого набору даних були обрані протоколи прикладного рівня: HTTP, HTTPS,SMTP, IMAP, FTP-DATA, FTP-COMMANDS, SSH, Skype, P2P, об'єднані в укрупнені групи табл. 6. Навчальний, загальний тестовий і все групові тестові дампи є незалежними один від одного наборами трафіку, що дозволяє проводити незалежну оцінку класифікатора.

В якості першого алгоритму виділення атрибутів трафіку, представленого в табл. 2.3. використовувався метод ранжирування на основі інформаційного виграшу InfoGain.

Таблиця 2.3 - Підготовлені дампи трафіку

Дамп трафіка	Кількість потоків мережевого трафіку по групам						Загальна кількість потоків
	Web (HTTP, HTTPS)	Mail (SMTP, IMAP)	FTP (FTP-DATA, FTP-COMMANDS)	SSH	Skype	P2P	
Навчаючий	3356	2209	2311	3975	2525	4677	19053
Загальний тестовий	1698	1151	1159	1849	1212	2315	9384
Web тестовий	1696	-	-	-	-	-	1696
Mail тестовий	-	1092	-	-	-	-	1092
FTP тестовий	-	-	1146	-	-	-	1146
SSH тестовий	-	-	-	2023	-	-	2023
Skype тестовий	-	-	-	-	1261	-	1261
Pear-to-Pear тестовий	-	-	-	-	-	2308	2308

Результати застосування алгоритму InfoGain, розташовані від гіршого до кращого, з урахуванням 10-кратної перехресної перевірки представлені на рис.2.4. З малюнка видно, після 14-го рангу відбувається різке падіння інформаційного виграшу від використання атрибута. Встановити причини цього допомагає аналіз значень і діапазону зміни даних атрибутів. Так атрибут tcp_flow_dir показує, в якому напрямку рухається потік. При цьому в кожному використовуваному класі

приблизно половина потоків - що входять, а половина – вихідні, що не дозволяє характеризувати клас даними атрибутом.

Атрибут tcp_syn. що відображає відсоток потоків, для яких встановлено прапор SYN. відповідає за синхронізацію номерів послідовності: його середнє значення незмінно і дорівнює нулю. Атрибут is_reversable. який показує, чи є потік двостороннім чи ні. в даному випадку завжди дорівнює одиниці, так як майже всі потоки двосторонні. Таким чином, атрибути tcp_syn, tcp_flow_dir, is_reversable не мають значного інформаційного виграшу і надалі можуть не використовуватися.

При пошуку і виділення ознак на навчальному наборі] також використовувався алгоритм CFS, за допомогою якого було оцінено в цілому 151 підмножина. Алгоритм показав, що найкращим є підмножина з атрибутами transport_protocol, src_port. dst_port. flags і показником оцінки якості Merit 0.806. Результати вибору ознак з використанням алгоритму CFS досить передбачувані і в цілому збігаються з ознаками, обраними алгоритмом InfoGain.

Третій використаний для виділення атрибутів алгоритм – обгортковий Wrapper - застосовується тільки спільно з цільовим алгоритмом класифікації.

Одна з основних складових алгоритму виділення атрибутів – оцінка підмножин - була реалізована за допомогою класифікації і оцінки результатів на обраних параметрах класифікатора

Оскільки при застосуванні різних цільових класифікаторів можливий вибір різних атрибутів, а також щоб уникнути «перенавчання» під конкретний класифікатор обгортковий алгоритм використовувався з різними класифікаторами табл.2.4.

Такі атрибути, як tot_pkts_qty, tot_pkts_bytes, rev_pkts_qty, fw_pkts_bytes, is reversable, header_count, tcp_flow_dir не були обрані обгорткового методом при використанні різних цільових класифікаторів.

Дослідження показують, що використання обгорткового алгоритму вибору ознак є ресурсномісткої обчислювальної операцією, яка при великій кількості атрибутів займає тривалий час, як це видно з табл.2.5. Найбільш тривалі

обчислення відзначалися при використанні алгоритму SVM: він реалізується лише для бінарної класифікації.

Таблиця 2.4 - Атрибути, обрані обгорткового алгоритмом Wrapper, при використанні різних класифікаторів

	C4.5(J48)	SVM	AdaBoost	Bagging	Naïve Bayes
rev_pkts_bytes		+			+
fv_pkts_qty		+			+
transport_protocol	+	+			
src_port	+	+	+	+	+
dst_port	+	+		+	+
wirelen		+			+
tcp_syn		+			+
tcp_ack		+			+
flags	+	+		+	+
payload_lenght	+	+		+	+

Коли кількість класів перевищує два, SVM будує безліч моделей класифікаторів за принципом «один проти всіх», що збільшує час, необхідний для побудови класифікатора. В результаті час, витрачений на виділення ознак при цільовому класифікаторі SVM, більш ніж в 15 разів перевищує час другого гіршого по часу цільового класифікатора - Bagging. Кращі результати демонструють алгоритми C4.5 (J48). Bagging.

Таблиця 2.5 - Показники обгорткового алгоритму Wrapper при виборі атрибутів

Показники	C 4.5(J48)	SVM	AdaBoost	Bagging	Naïve Bayes
Точність	0,999	0,88	0,429	0,999	0,86
Кількість оцінених підмножин ознак	130	154	93	109	145
Витраченого часу, хв.	6	>150	1	9	2,5

2.3 Аналіз впливу обсягу навчальної вибірки на якість класифікації інтернет- трафіку

2.3.1 Оцінка якості класифікації по пакетам і потокам та їх порівняння

Ефективність алгоритмів МН оцінювалася по метриках на тестовій вибірці, що складається з 900 тисяч тестових екземплярів.

Дослідження показали, що для всіх алгоритмів збільшення розмірів навчальної вибірки призводить до підвищення якості класифікації. Поточна достовірність класифікації для повного набору ознак приймає своє мінімальне значення в 60,3% при розмірі навчальної вибірки в 300 тисяч примірників, тоді як для скороченого набору ознак, потокова достовірність при такому ж розмірі навчальної вибірки майже становить 74%.

Інші метрики також показують приріст, що свідчить про підвищення якості передбачення. Зростання точності для скороченого кількості атрибутів при всіх розмірах навчальних вибірок свідчить про зменшення кількості помилок, а зрослий відгук говорить про збільшення істинно позитивних результатів.

Для порівняння використовувалися потокова і байтова точності. На рис. 2.5. зображені графіки залежності часу навчання від розміру навчальної вибірки, згруповані за алгоритмами класифікації (*а*), і діаграма середнього зросту часу побудови моделі за алгоритмами (*б*).

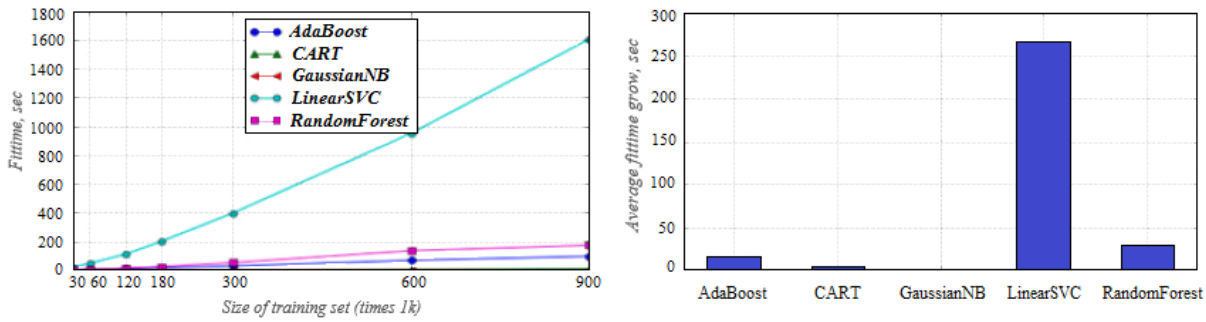


Рисунок 2.5 - Оцінка часу навчання різними алгоритмами класифікації:
а) залежність часу побудови моделі від розміру навчальної вибірки;
б) діаграма середнього зросту часу побудови моделі.

Як видно, час навчання алгоритму SVM зі збільшенням навчальної вибірки зростає швидше, ніж у інших алгоритмів, що обумовлено високою обчислювальною складністю даного алгоритму. Найшвидшим алгоритмом є класифікатор Naive Bayes. Другим за швидкістю є алгоритм CART. Алгоритми Random Forest і AdaBoost в якості базових класифікаторів використовують алгоритм CART, тому. очевидно, що ансамблеві алгоритми навчаються повільніше. При побудові моделі алгоритму AdaBoost використовується 37 базових алгоритмів, в той час як у алгоритму Random Forest їх 197. рис. 2.6 ілюструє залежність достовірності та I-заходи різних алгоритмів від розміру навчальної вибірки. Отримані експериментально. З представлених графіків видно, що використовуючи сформований вектор ознак достовірність і точність алгоритмів CART і Random Forest зростають до максимальних значень.

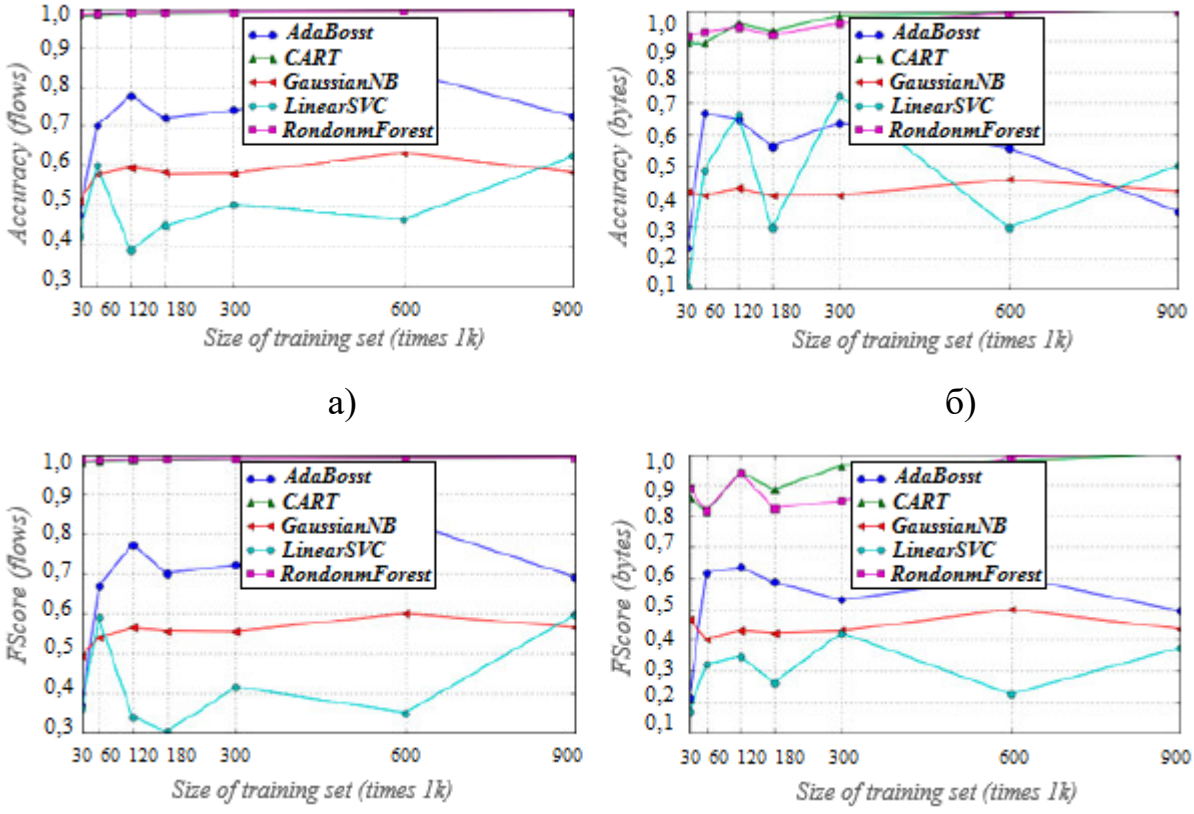


Рисунок 2.6 - Залежність вірогідності і F-міри алгоритмів від розміру навчальної вибірки: а) поточна достовірність; б) байтова достовірність: в) поточна F-міра: г) байтова F-міра

При всіх розмірах навчальних вибірок потокова достовірність коливається від 98 до 99%. Байтова достовірність, в переважній більшості випадків менше, ніж поточна.

При невеликих розмірах навчальних вибірок – 30, 60 тисяч примірників, достовірність алгоритму CART становить 89%. в той час, як для алгоритму Random Forest - 91-93%.

Відмінності практично в 10% від точності класифікації потоків говорять про суттєві відмінності в природі цього показника.

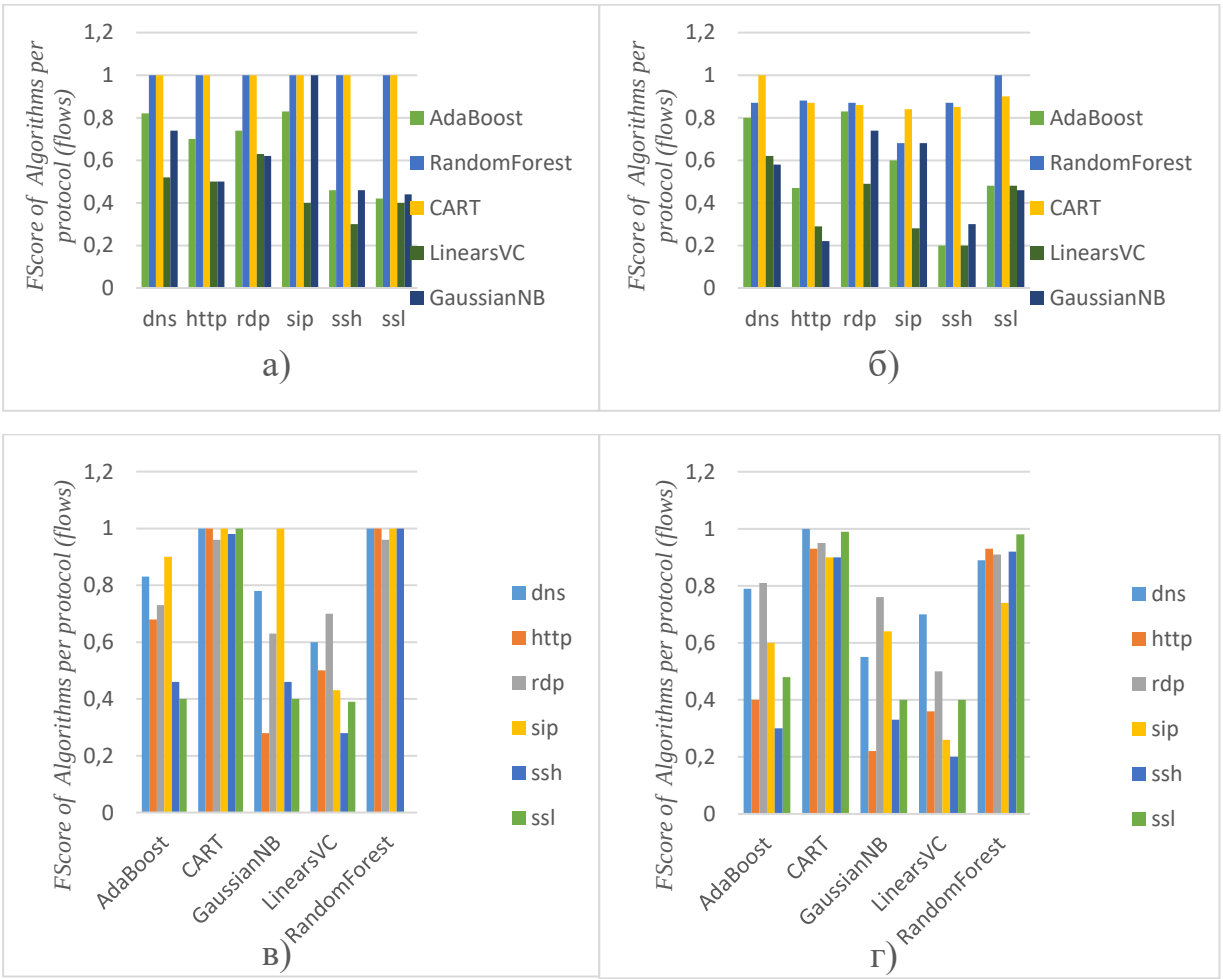


Рисунок 2.7 - Достовірність різних алгоритмів в розрізі по протоколам: а) потокова достовірність кожного алгоритму по протоколам; б) байтова достовірність кожного алгоритму по протоколам; в) поточна достовірність класифікації протоколу певним алгоритмом; г) байтова достовірність класифікації протоколу певним алгоритмом.

Алгоритми SVM і Naive Bayes показують низьку достовірність результатів, що не дозволяє, говорити про їх застосовності для класифікації трафіку. Хоча алгоритм AdaBoost і використовує алгоритм CART для побудови базових класифікаторів, його достовірність і F-міра коливаються від 21 до 66%, що також робить його застосування для класифікації трафіку недоцільним.

Важливою особливістю алгоритму класифікації є його здатність однаково добре ідентифікувати кожен протокол по байтам і по потокам.

З діаграм, представлених на рис. 2.7 можна бачити, що такою здатністю володіє алгоритм CART, що показав найкращі показники.

Наведені експериментальні результати вимірювань трафіку в ЦОД показали, що розподіл найбільш значущих програм у зібраному дампі трафіку за розміром в потоках і пакетах (байтах) істотно розрізняються, що має бути враховано при класифікації трафіку.

Оцінка важливості атрибутів класифікації додатків трафіку проведена за допомогою алгоритму Random Forest дозволила зменшити кількість ознак до 12 при забезпеченні високої достовірності класифікації.

Оцінка моделей класифікації за часом побудови показує, що найгіршим є алгоритм SVM, а найкращим Naive Bayes. Отримані залежності достовірності класифікації і F-заходи показали, що алгоритм CART і Random Forest більш кращі ніж алгоритми SVM, Adaboost і Naive Bayes.

Показано, що у всіх аналізованих випадках ефективність класифікації по потокам вище, ніж по байтам.

2.3.2 Результати класифікації додатків на етапі тестування

Розглянемо результати класифікації додатків на етапі тестування алгоритму SVM. Оскільки цей алгоритм дозволяє проводити класифікацію тільки за двома класами, то при кількості класів більше двох для кожної алгоритму SVM. Оскільки цей алгоритм дозволяє проводити класифікацію тільки за двома класами, то при кількості класів більше двох для кожної комбінації класів будувався окремий

класифікатор. В результаті для шести класів за допомогою SVM було створено 15 класифікаторів, які вони порівнювалися між собою, і визначалася їх приналежність того чи іншого класу. Після навчання підготовленим набором і тестуванням були отримані результати, представлені в табл.2.6. з яких видно, що найбільша помилка має місце при роботі з класами Peer-to-Peer (17.85%) і Web (19,28%).

Таблиця 2.6 - Результати тестування алгоритмів SVM і C4.5 (J48) на дампи тестового трафіку

Дамп тестового трафіка	Web	Mail	FTP	SSH	Skype	P2P	Вихідне	Похибка, %
SVM								
Web	1369	263	10	-	52	2	1696	19,28
Mail	44	1048	-	-	-	-	1092	4,03
FTP	-	-	1146	-	-	-	1146	0
SSH	3	-	-	2020	-	-	2023	0,15
Skype	-	-	-	-	1192	69	1261	5,47
P2P	40	225	-	-	147	1896	2308	17,85
C4.5 (J48)								
Web	1696	-	-	-	-	-	1696	0
Mail	8	1084					1092	0,7326
FTP	-	-	1146	-	-	-	1146	0
SSH	-	-	-	2023	-	-	2023	0
Skype	-	-	-	-	1261	-	1261	0
P2P	-	-	-	-	-	2308	2308	0

Вважається, що алгоритми машинного навчання з використанням «дерев рішень» найкращим чином підходять для класифікації при великій кількості класів. На відміну від алгоритмів типу SVM (лінійні алгоритми, спочатку призначені для бінарної класифікації), «дерева рішень» складаються з вузлів, на яких приймається рішення. Кінцевим результатом безлічі таких рішень є «листя» дерева, відповідні передбачуваному класу.

Дослідження показали, що оцінку алгоритму C4.5 (J48) проводити важко, оскільки всі показники класифікації мають або максимальне, або майже максимальне значення.

Розглянемо результати класифікації з алгоритмом AdaBoost - ансамблевим алгоритмом класифікації з безліччю простих класифікаторів, об'єднаних в ансамбль. Простий класифікатор в AdaBoost є налаштованим, тому спочатку проводилася його оцінка при використанні різних базових класифікаторів.

Показники точності класифікації (як з використанням AdaBoost, так і без нього) говорять про те. що в більшості випадків об'єднання безлічі класифікаторів в ансамбль і прийняття рішення на основі «голосування» дозволяють поліпшити результати класифікації. Найбільший ефект від об'єднання класифікаторів – більше 10% - демонструє алгоритм Naive Bayes. Погіршення результату в разі AdaBoost спостерігається для алгоритму SVM. Це викликано тим, що при перерозподілі ваг в алгоритмі AdaBoost найбільший показник ваги отримали неправильні SVM-класифікатори, які в цьому випадку самі стали джерелом помилок.

Ансамблеві алгоритми дозволяють об'єднувати безліч «простих» класифікаторів для прийняття рішення шляхом голосування, що на прикладі AdaBoost ілюструє можливість значного поліпшення показників точності класифікації, особливо для класифікаторів з самого початку невеликими значеннями точності, таких, наприклад, як Naive Bayes. Однак при використанні деяких складних алгоритмів (типу SVM) об'єднання в ансамбль призводить до погіршення результатів. Тоді застосовувати ансамблеві алгоритми з іншими ансамблями в якості простого класифікатора не рекомендується навіть при хороших показниках класифікації.

Як і AdaBoost. алгоритм Bagging є ансамблевим класифікатором, тому аналізувати його роботу і результати слід з різними «Простими» класифікаторами рис.2.8. Очевидно, що використання алгоритму Bagging дозволяє поліпшити результати практично для всіх розглянутих алгоритмів, крім Naive Bayes. Другий найкращий результат показав алгоритм C4.5. Поліпшення точності при використанні Bagging не така значне, як з AdaBoost. оскільки при побудові підсумкової моделі класифікатора наступні прості класифікатори навчаються на найбільш складних об'єктах, що не були коректно класифіковані на попередніх ітераціях.

У разі використання алгоритму Bagging все класифікатори працюють незалежно один від одного, причому підмножини, які подаються на вхід, випадкові. Результати класифікації, представлені в табл.2.7. Були отримані з використанням алгоритму Bagging класифікатора C4.5 (J48).

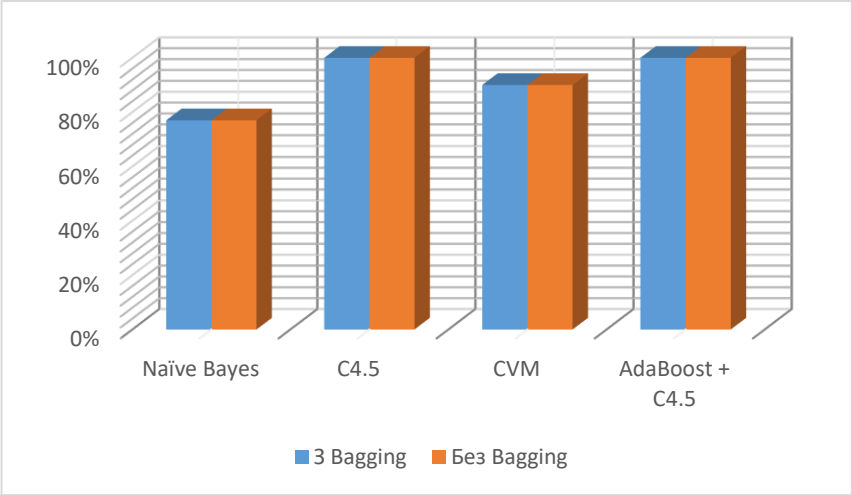


Рисунок 2.8 - Показник точності при використанні алгоритму Bagging з різними класифікаторами

Таблиця 2.7 - Результати тестування алгоритму Bagging на дампи тестового трафіку

Дамп тестового трафіка	Web	Mail	FTP	SSH	Skype	P2P	Вихідне	Помилка, %
Web	1696	-	-	-	-	-	1696	0
Mail	9	1083	-	-	-	-	1092	0,8242
FTP	-	-	1146	-	-	-	1146	0
SSH	-	-	-	2023	-	-	2023	0
Skype	-	-	-	-	1261	-	1261	0
P2P	-	-	-	-	1	2307	2308	0,0433

З графіка залежності порту призначення транспортного рівня від класу рис.2.9*а* видно, що FTP використовує в роботі один, фіксований порт, який відрізняється від присвоєного IANA 20-го. Відзначимо, що Skype і особливо P2P використовують в своїй роботі множини рівномірно розподілених у всьому діапазоні динамічних портів. Це означає, що в ряді випадків багато атрибути

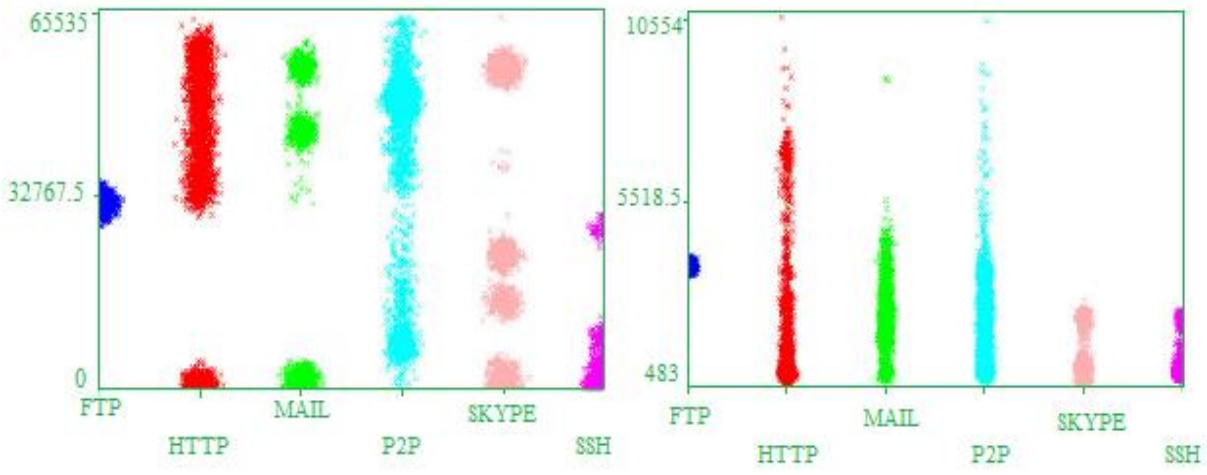
класифікації трафіку не несуть значного інформаційного виграшу і негативно впливають на показники ефективності класифікації. До подібних атрибутів відносяться, зокрема, розмір і кількість пакетів в потоці, а також в прямому і зворотному напрямку.

З графіка залежності розміру потоку від належного класу рис.2.9 *б* слід, що розмір потоку не впливає на клас додатки для всіх класів, крім FTP. Більш того, в разі обгорткового алгоритму вибору ознак всі розглянуті цільові алгоритми класифікації показали гірші результати при використанні даних атрибутів в підмножині ознак.

Таким чином, сформовані для оцінки ефективності алгоритмів тестова вибірки трафіку для додатків FTP, Web, Mail, SSH, Skype показано, що використання при виборі ознак класифікації алгоритму InfoGain дозволило скоротити кількість атрибутів класифікації до 14.

На етапі вибору атрибутів обгорткового методом кращі результати спостерігаються при використанні сукупності алгоритмів C4.5 (J48) і Bagging.

На етапі тестування найбільш високу ефективність показали ансамблеві алгоритми Adaboost і Bagging в поєднанні з C4.5 (J48).

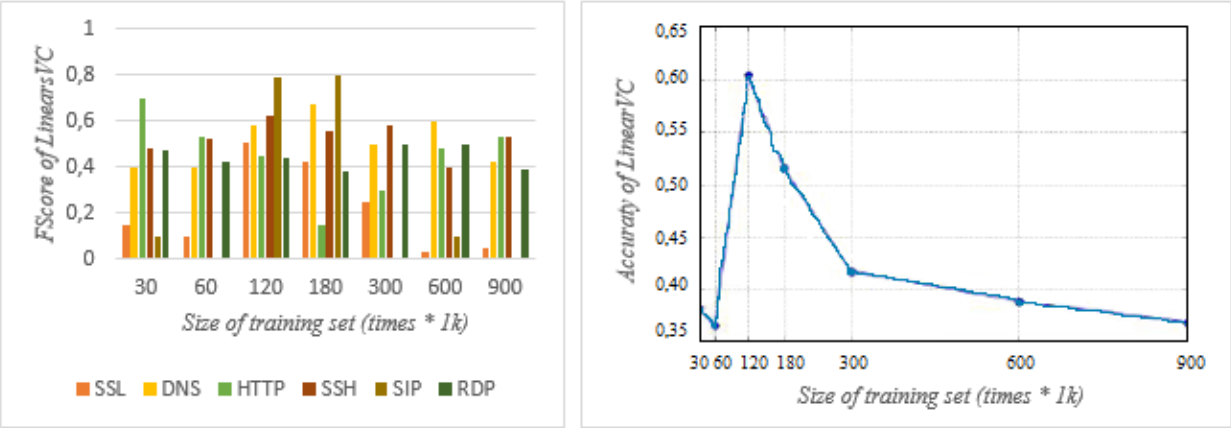


а) б)

Рисунок 2.9 - Залежно порту призначення (а) і розміру потоку (б) від класу додатка

2.3.3 Експериментальні результати класифікації на етапі навчання

Алгоритм SVM. Розглянемо експериментально отримані залежності достовірності і F-міри алгоритму SVM від розміру навчальної вибірки, представлені на рис. 2.10*а*. Спосіб оцінки якості алгоритмів в розрізі розміру навчальної вибірки є популярною практикою. Як можна помітити з рисунка 2.10*б*, достовірність перевищує 50% тільки при розмірах навчальної вибірки в 120 і 180 тисяч примірників. Найкраще SVM впорався з ідентифікацією SSH, а найгірше з SIP.



а) б)

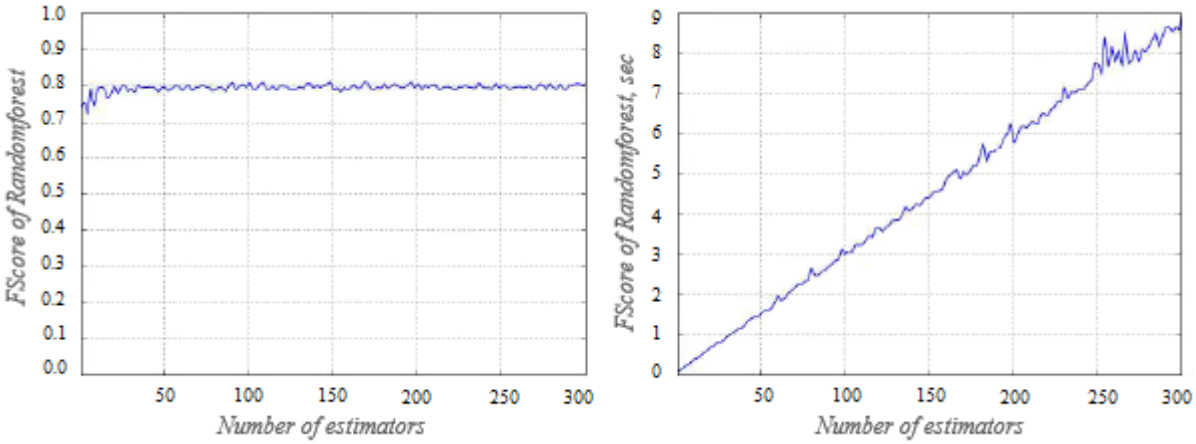
Рисунок 2.10 - Залежність вірогідності алгоритму SVM від розміру навчальної вибірки

Алгоритм AdaBoost. На рис.2.11 представлені графіки експериментально отриманих залежностей якості класифікації і часу побудови моделі від кількості базових класифікаторів.

Оскільки F-міра коливається дуже незначно рис.2.11 *а*, кількість дерев мало впливає на кінцеву якість класифікації.

Розглянутий параметр досягає максимального значення при 197 базових класифікаторах і становить 80.78%.

Практично лінійний характер зміни часу класифікації представлений на рис.2.11 *б* робить передбачуваним необхідний час побудови моделі.

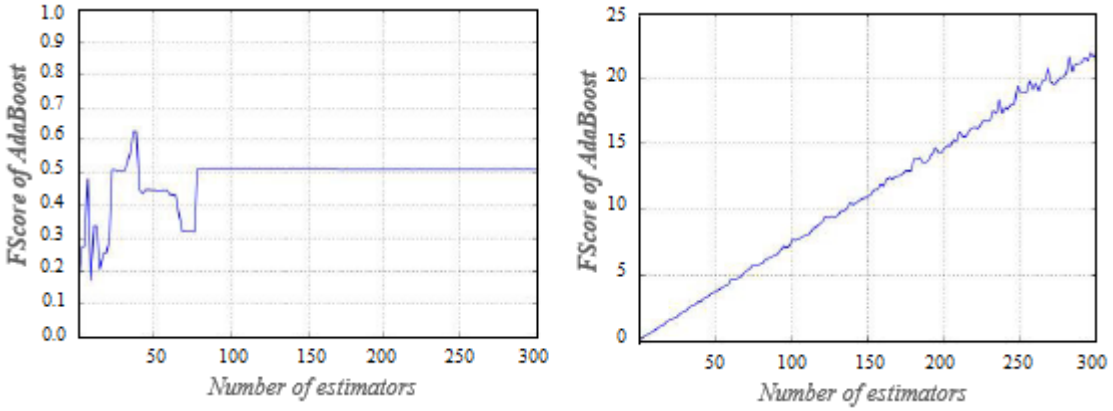


а) б)

Рисунок 2.11 - Залежність параметрів алгоритму AdaBoost від кількості базових класифікаторів: а) залежність Е-заходи: б) залежність часу побудови моделі

Як параметра, що набудовує в алгоритмі AdaBoost виступає максимальну кількість базових алгоритмів. Для оптимізації даного параметра була використана необроблена вибірка даних, що складається з 30 тисяч примірників, по 5 тисяч примірників кожного класу. В якості тестової вибірки використовувалася вибірка з 60 тисяч примірників, по 10 тисяч кожного класу. Мірою оцінки обрана F-міра. На рис.2.12*а* представлена залежність F-заходи від кількості базових алгоритмів, а рис. 2.12*б* ілюструє криву зростання часу побудови моделі в залежності від кількості базових класифікаторів.

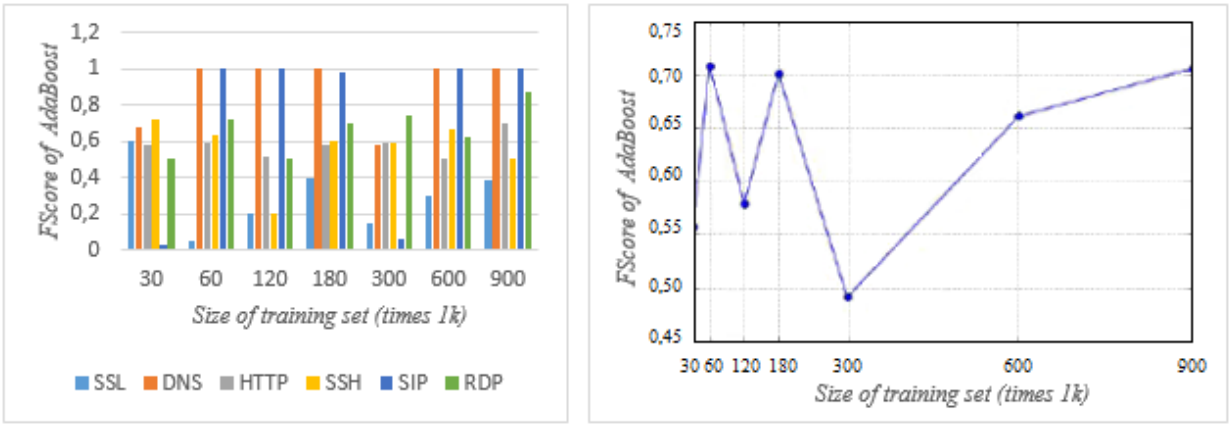
З рис. 2.12*а* видно, що спостерігається значне коливання графіка в межах 70 базових класифікаторів. Максимальне значення Р-заходи припадає на 37 базових алгоритмів і становить 62%. Хоча зазвичай вважається, що чим більше базових алгоритмів в ансамблі, тим вища якість. В данному випадку, коливання графіка припиняються вже при 80 базових алгоритмах. F-міра при цьому встановлюється в межах 50-51%. Час побудови ансамблю зростає практично лінійно, що дозволяє прогнозувати час створення моделі при будь-якій кількості базових класифікаторів.



а) б)

Рисунок 2.12 -Залежність параметрів алгоритму AdaBoost від кількості базових класифікаторів а) Залежність F-заходи від б) Залежність часу побудови моделі

Для оцінки залежності якості класифікації від обсягу обучаюшей вибірки можна скористатися результатами опеньки достовірності і F-міри. представленими на рис. 2.13.



а) б)

Рисунок 2.13 - Залежність вірогідності і F-міри алгоритму AdaBoost від розміру навчальної вибірки а) F-міра; б) достовірність.

Найбільшу вірогідність алгоритм показав при розмірі навчаючої вибірки в 60 тисяч екземплярів. Коливальний характер представленої залежно обумовлений

зашумленим характером вихідних даних, як і інші алгоритми. AdaBoost найкраще ідентифікував SIP і DNS трафік.

Недоліком алгоритму є труднощі визначення потрібної кількості ітерацій навчання. Крім того, потрібно досить довгих навчальних вибірок.

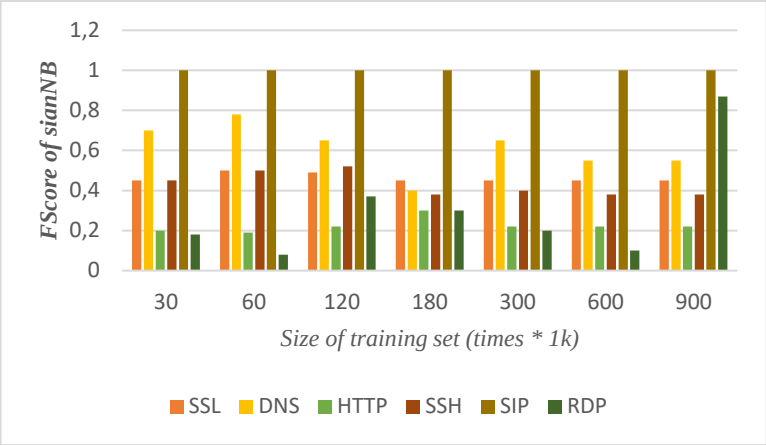


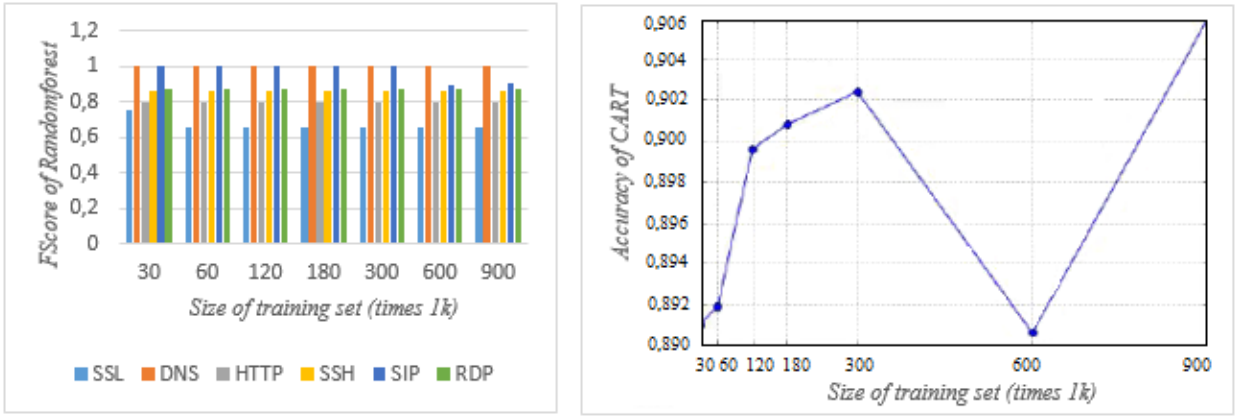
Рисунок 2.14 - Залежність F-міри класифікатора Naive Bayes від розміру навчальної вибірки

Класифікатор Naive Bayes (NB). перевагами класифікатора NB є простота реалізації і висока продуктивність, однак достовірність даного алгоритму при класифікації трафіку зазвичай не висока рис. 2.14. Показники помилок практично по всіх протоколах високі за винятком протоколу SIP. про що свідчать показники F-міри.

Алгоритм CART. При застосуванні вирішального дерева CART до даних потоків трафіку спостерігаються результати, на порядок перевершують результати алгоритмів SVM і NB. Як приклад на рисунку 2.15 наведені графіки залежності показників достовірності і F-заходи від розміру навчальної вибірки. Максимальна достовірність в 90% досягається при 900 тисячах навчальних випадків. Порівняння показника F-заходи по протоколам показує, що найкраще ідентифікуються протоколи DNS і SIP. Мінімальна значення F-міри становить 71% для протоколу SSL.

Випадковий ліс (Random Forest) - ансамблевий алгоритм машинного навчання, використовує для класифікації або регресії кілька дерев рішень [22].

Будь-яке глибоке дерево рішень схильне до перенавчання, тобто занадто сильною підгонці моделі під тренувальні дані.



а) б)

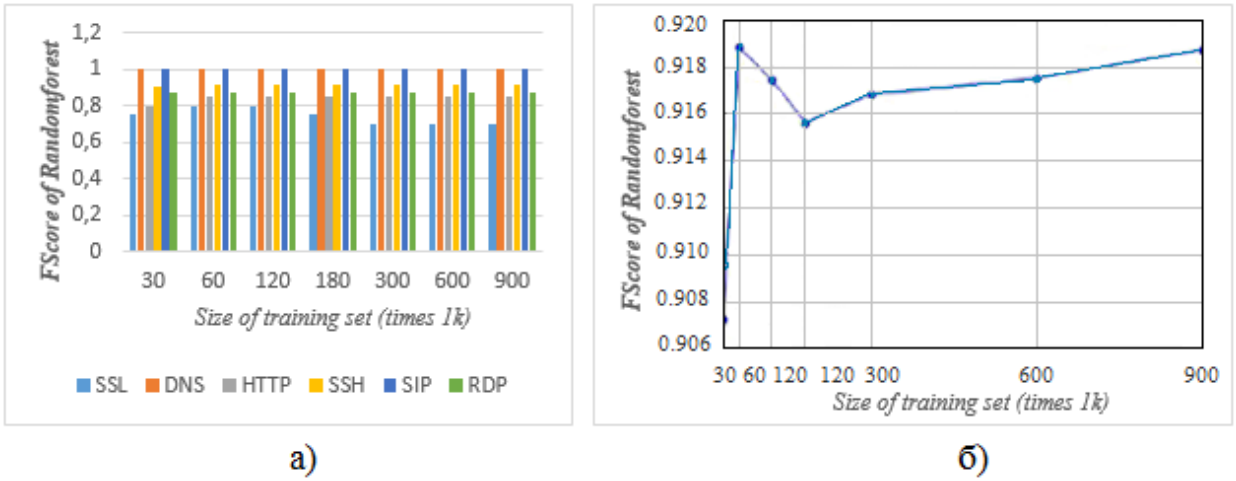
Рисунок 2.15 - Залежність вірогідності і F-міри алгоритму CART від розміру навчальної вибірки а) F-міра; б) достовірність.

На стадії тестування це виражається в неможливості коректно класифікувати нові дані. Така схильність до перенавчання RF обумовлена наявністю великої кількості гілок вирішального дерева, які на кінці містять один і той же клас. Оскільки при побудові вирішальних дерев алгоритм RF використовує ідеї Беггінген і випадкових підпросторів, це дозволяє знизити час перенавчання і поліпшити якість класифікації.

Нагадаємо, що Беггінген – це алгоритм вибору випадкового набору об'єктів з повної вибірки. Алгоритм будує задану кількість вирішальних дерев, подаючи на вхід випадкові набори об'єктів. Докладний підхід дозволяє будувати "лісу" некорельованих між собою дерев, що в свою чергу дозволяє зменшити загальну дисперсію алгоритму, при незмінному змішуванні.

Випадкове підпростір є випадкові вибірки ознак. Якщо представляти дані у вигляді матриці за допомогою стовпців – ознаками і рядками - об'єктами, то за допомогою процедури Беггінген реалізується механізм вибору рядків, а в якості підпростору виступає випадковий набір стовпців в цих рядках. Дослідження показують, що подібний підхід дає хороші результати. Як приклад на рис. 2.16

представлені показники F-заходи і достовірності для алгоритму RF. Як можна помітити, при всіх розмірах навчальної вибірки достовірність прогнозів алгоритму RF вище 90% Середній показник F-заходи по всіх протоколах становить 91.4%.



а) б)

Рисунок 2.16 - Залежність вірогідності і Б-заходи алгоритму RandomForest від розміру навчальної вибірки а) F-міра б) достовірність

2.4 Дослідження ефективності застосування алгоритму RF в задачах класифікації додатків

Формування даних. Як було показано вище, одним з найбільш часто використовуваних і ефективних для класифікації мережевого трафіку методів машинного навчання є вирішальне дерево [21].

RF являє собою ансамблевий метод навчання для класифікації і регресії, який діють шляхом побудови безлічі вирішальних дерев. Оцінимо ефективність роботи алгоритму RF в задачах класифікації додатків в умовах наявності і відсутності фонового мережевого трафіку. Для збору необхідних для аналізу даних була організована лабораторна мережа з декількох комп'ютерів. Схематичне зображення використовуваної мережі наведено на рис. 2.17. Один з комп'ютерів був підключений до глобальної мережі інтернет і на його базі було організовано бездротова точка доступу. На цьому ж комп'ютері здійснювався захоплення всього

проходить через нього трафіку за допомогою програми Wireshark. На інших комп'ютерах, підключених до точки доступу були запущені різні додатки.

Здійснювався перегляд веб-сторінок за допомогою браузерів Google Chrome і Opera, за допомогою програми Skype, проводилися відеодзвінки, здійснювалося скачування файлів за допомогою торрент клієнта μTorrent, використання сервісу цифрового поширення комп'ютерних ігор Steam і т.д. Отримані дані зберігалися в форматі PCAP.

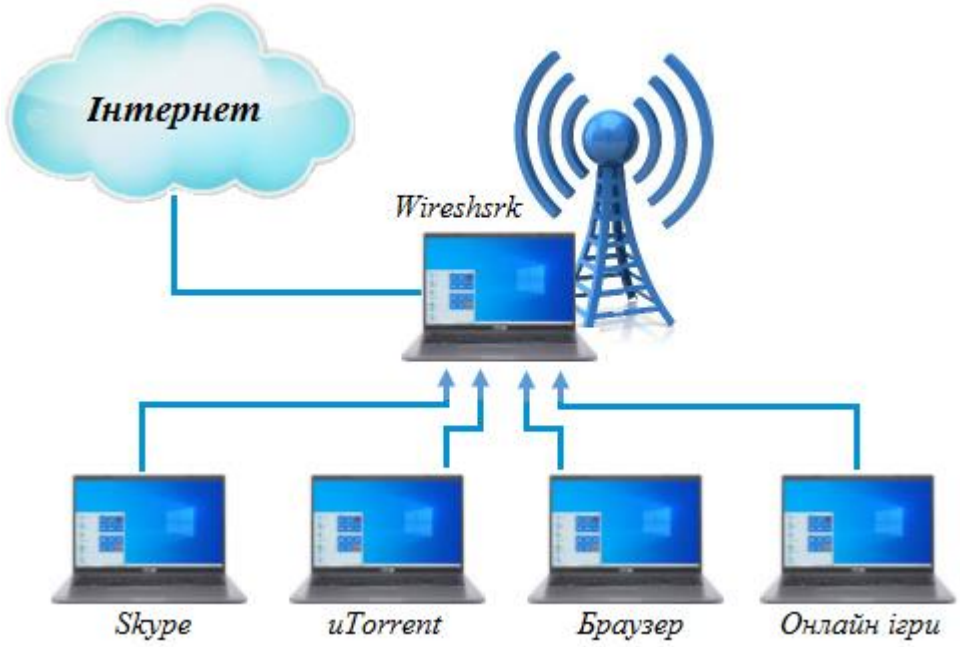


Рисунок 2.17 - Схема використовуваної мережі

Для приведення отриманих даних у відповідність до вимог розв'язуваної завдання, була проведена попередня обробка даних. З цією метою всі пакети були розділені на потоки транспортного рівня. ідентифікуються як і перш за п'ятьма значеннями: протокол транспортного рівня (TCP або UDP), IP- адреса джерела, порт джерела. 1P-адреса одержувача, порт одержувача. Отримані дані були помічені, так що кожному потоку був поставлений в відповідність протокол/додаток, до якого цей потік відноситься. Отриманий набір даних був поділений на дві підвибірки - навчальну і тестову. Склад отриманого датасета наведено в табл.2.8.

Таблиця 2.8 – Склад отриманої вибірки даних

Протокол	Навчаюча вибірка	Тестова вибірка
SSL	1215	295
HTTP	1091	272
DNS	1061	267
BitTorrent	940	232
Steam	775	204
Skype	645	162

За допомогою вбудованого в RF алгоритму відбору інформаційних ознак Feature Importance були відібрані наступні 11 ознак табл.2.9.

Таблиця 2.9 - Відібрані інформаційні ознаки

№	Назва	Опис
1	src_port	Номер порта джерела (джерелом рахується відправник першого пакета)
2	dst_port	Номер порта отримувача
3	max_src-data_port	Максимальний розмір даних в пакеті від джерела
4	min_src_data_ip	Мінімальний розмір даних в пакеті від джерела
5	med_src_data_ip	Медіанний розмір даних в пакет від джерела
6	prop_src_data_ip	Доля даних, переданих джерелом в загальній кількості даних потоку
7	max_dst_data_ip	Максимальний розмір даних в пакеті від отримувача
8	mean_dst_data_ip	Середній розмір даних в пакеті від отримувача
9	src_to_dst_ratio_data_ip	Відношення розміру даних, переданих джерелом до розміру даних, які передані отримувачем
10	min_data_ip	Мінімальне значення розміру даних в потоці
11	var_data_ip	Середньоквадратичне відхилення значення розміру даних в потоці

Результати класифікації. В експерименті було проведено побудову випадкового лісу і оцінка якості класифікації на заданій вибірці. Дослідним шляхом

були підібрані найбільш прийнятні параметри алгоритму. Так експериментально вибрано, що ліс складається з 5 дерев з максимально можливою глибиною.

У табл.2.10 представлена матриця помилок для чистої тестової вибірки. За вертикалі вказані реальні значення, по горизонталі – передбачені навченої моделлю.

Для визначення ефективності алгоритму використовуються наступні метрики: точність, повнота і F-міра, значення яких легко розрахувати на підставі матриці помилок класифікації табл. 13. яка складається для кожного класу окремо.

Таблиця 2.10 - Матриця помилок для тестової вибірки

Передбач. / реальності	SSL	HTTP	DNS	BitTorrent	Steam	Skype
SSL	295	0	0	0	0	0
HTTP	0	267	0	4	1	0
DNS	0	0	266	1	0	0
BitTorrent	1	0	0	230	0	1
Steam	0	3	0	0	201	0
Skype	6	0	0	1	0	155

Графічне представлення даних метрик отриманих експериментально для всіх класів приведено на рисунку 2.18.

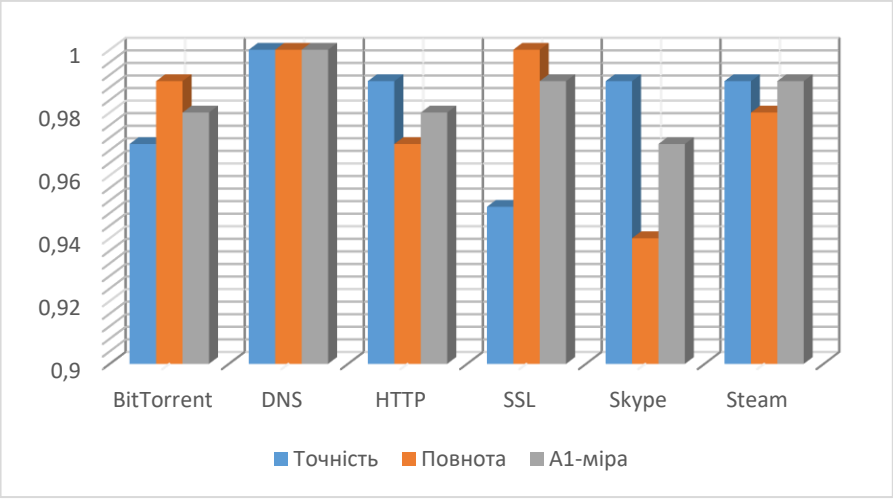


Рисунок 2.18 - Точність, повнота, F-міра для тестової вибірки

Видно, що найбільшу ефективність алгоритм має для даних, що відносяться до DNS трафіку. Крім перевірки роботи алгоритму на тестовій вибірці, що має такий же класовий склад, як і навчальна, оцінка його якості проводилася також в умовах присутності фонового трафіку, тобто в разі коли в тестовій вибірці були присутні екземпляри класів, відсутніх в навчальній вибірці. Склад цієї вибірки наведено в табл. 2.11.

Таблиця 2.11 - Склад тестової вибірки даних з домішками

Протокол	Якість протоколу
SSL	295
HTTP	272
DNS	267
BitTorrent	232
Steam	204
Skype	162
LLMNR	169
Quic	95
RTP	19

Така ситуація, коли в класифікуються даних присутній фоновий трафік, більш наближена до дійсності, в силу різноманіття використовуваних в мережі Інтернет протоколів. В табл.2.12 представлена матриця помилок для даного випадку.

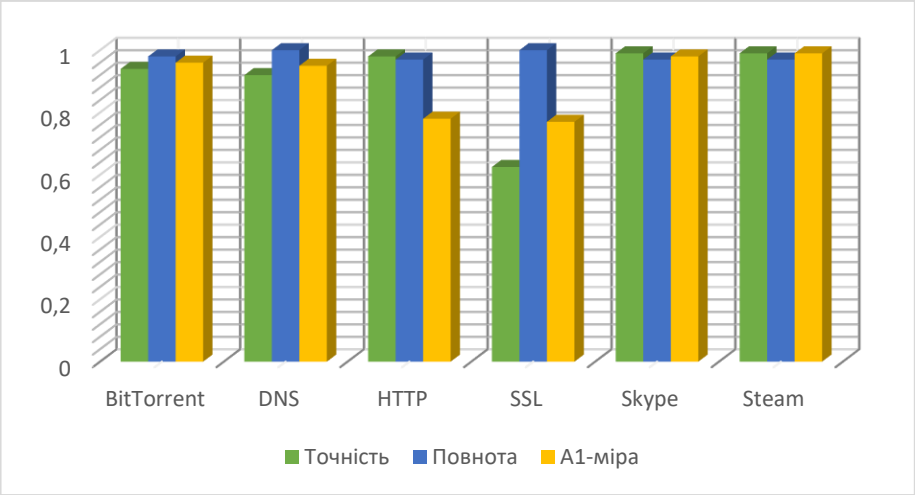


Рисунок 2.19 - Точність, повнота. F-міра при наявності фонового трафіку

Як видно з таблиці, всі екземпляри, що відносяться до класу LLMNR модель класифікувала як SSL, всі екземпляри RTP були віднесені до Skype, а екземпляри класу Quic в основному були розділені між класами DNS і Skype.

Розглянемо, як змінилися показники якості класифікації рис.2.19.

Таблиця 2.12 - Матриця помилок для тестової вибірки при наявності фонового трафіку

Передбач. / реальності	SSL	HTTP	DNS	BitTorrent	Steam	Skype	LLM-NR	Quic	RTP
SSL	295	0	0	0	0	0	0	0	0
HTTP	0	267	0	4	1	0	0	0	0
DNS	0	0	266	1	0	0	0	0	0
BitTorrent	1	0	0	230	0	1	0	0	0
Steam	0	3	0	0	201	0	0	0	0
Skype	6	0	0	1	0	155	0	0	0
LLMNR	169	0	0	0	0	0	0	0	0
Quic	0	0	25	9	0	61	0	0	0
RTP	0	0	0	0	0	19	0	0	0

Як видно, наявність фонового трафіку практично не вплинуло на значення повноти, але значно погіршило значення точності класифікації, оскільки збільшилася кількість False Positive примірників поява яких викликано наявністю фонового трафіку, що належить до класів, які в навчанні не брали участь

Разом з тим, алгоритм RF продемонстрував високу ефективність в режимі off-line, про що, зокрема, свідчить F-міра рівна відповідно 0,987 і 0,759 при відсутності і наявності фонового трафіку.

Наявність фонового трафіку належить до класів, що не брали участь в навчанні алгоритму, значно погіршує точність класифікації.

Таким чином, алгоритм RF мало придатний для класифікації в режимі реального часу через тимчасову складності обробки, що оцінюється співвідношенням про ($Mmn \log [(n)]$), де n - кількість примірників, m – кількість інформаційних ознак, а M - кількість дерев.

3 ПРАКТИЧНЕ ДОСЛІДЖЕННЯ КЛАСИФІКАЦІЇ ІНТЕРНЕТ-ТРАФІКУ В УМОВАХ НАЯВНОСТІ НЕБАЖАНОГО ФОНОВОГО ВПЛИВУ

3.1 Аналіз впливу фонового трафіку на якість класифікації

Більшість алгоритмів машинного навчання з учителем спроектовані для навчання бінарних або мультиклассових класифікаторів [82.98]. На основі навчального набору даних, що складається з об'єктів обох класів бінарні (або біноміальні) класифікатори вибирають між двома класами об'єктів. Відповідно мультикласове (або поліноміальний) класифікатори поділяють об'єкти на множини класів відповідно до тренувального набору даних, що складається з об'єктів всіх класів. Обидва типи класифікаторів засновані на двох припущеннях.

По-перше – всі класи відомі завчасно. По-друге – для кожного класу є ефективний і показовий набір даних.

Іншими словами, класифікатори з учителем нездатні визначити об'єкт невідомого класу, які не представлені в навчальній вибірці. В той же час, ідентифікація невідомого типу трафіку є найважливішою вимогою в сучасній класифікації мережевого трафіку оскільки, в зв'язку з еволюцією Інтернету з'являються нові додатки та протоколи, нові типи трафіку.

З іншого боку, навіть для існуючих додатків і протоколів дуже важко і дорого отримати повноцінний позначений набір даних, характеризують кожен клас.

Таким чином, щоб побудувати практичний класифікатор трафіку методами машинного навчання з учителем, потрібно бути дуже обережними з визначенням класу і побудовою тренувального набору.

Розглянемо вплив невідомого фонового трафіку на якість класифікації з використанням машинного навчання:

- Skype - безкоштовне програмне забезпечення, що дозволяє здійснювати зв'язок різного виду між комп'ютерами по мережі Інтернет;

- Steam - сервіс цифрового поширення комп'ютерних ігор і програм, що належить компанії Valve, відомого розробника комп'ютерних ігр:

- BitTorrent - клієнт, який використовує протокол P2P для обміну різними файлами через мережу Інтернет:

- YouTube, трафік отриманий в процесі використання відомого відеохостингу через браузер;

- Facebook - трафік популярної соціальної мережі, в якій є інформація про зареєстровані в ній користувачів, є можливість прослуховувати музику та переглядати відеозаписи, а також обмінюватися повідомленнями.

Даний трафік знімався на спеціально змодельованої макеті рис. 3.1. що складається з ПК з операційною системою Ubuntu, на якому безпосередньо перехоплює трафік. ПК з операційною системою Windows 7 і планшета під управлінням ОС Android. Всі пристрої з'єднувалися з мережею Інтернет за допомогою маршрутизатора Tp-Link.

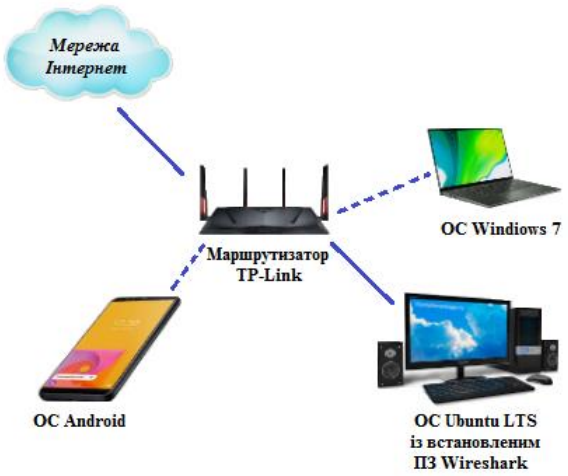


Рисунок 3.1 - Макет для зняття трафіку

Трафік захоплювався наступним чином. Спочатку перевірялася наявність стороннього фонового трафіку в мережі за допомогою Wireshark. Щоб упевнитися, що зібраний в результаті роботи трафік, належить до потрібного додатка. Потім запускалося то додаток, яке генерує необхідний трафік. Паралельно з цим включається аналізатор трафіку Wireshark, який захоплює всі пакети, створювані

працюючою програмою. Процес повторюється для отримання кожного типу необхідного трафіку. Будемо представляти фоновий трафік як дані, що не були представлені в навчальній вибірці, проте присутні в тестовому наборі [15].

Оскільки класифікатори не були готові до такого типу потоків і не були навчені до їх класифікації, вони здатні лише віднести невідомий трафік до одного або декількох відомих класів, визначення яких і є основним завданням.

Проведемо порівняння показників якості роботи класифікаторів без фонового трафіку і при його наявності. Щоб наочно оцінити, на скільки зменшилася точність класифікації цільових класів різними алгоритмами, в табл.3.1 наведені різниці між середніми значеннями критеріїв точності, повноти і F-заходи до додавання в вибірку фонового трафіку і після.

Таблиця 3.1 - Різниця середніх значень критеріїв точності без і з фоновим трафіком

Критерій/ Алгоритм	SVM	Naïve Bayes	One Rule	C4.5	Random Forest	AdaBoost +One Rule
Precision	0,033	0,054	0,016	0,115	0,133	0,117
Recsll	0,000	0,000	0,000	-0,004	0,000	0,000
F-Measure	0,017	0,031	0,063	0,071	0,082	0,068

З представлених даних можна бачити, найбільш відмінності в якості роботи алгоритмів спостерігаються в зменшенні точності класифікації, оскільки різниця критерію Precision у різних умов тестування найбільша. Показник повноти практично не змінився, що в підсумку привели до нульових різниць середніх значень. Кожен з алгоритмів практично не відхилявся від своєї моделі класифікації і при наявності фонового трафіку інші п'ять класів вони розподіляли точно так, як і при його відсутності [17]. Виняток становить тільки алгоритм C4.5, який дещо змінив розподіл потоків по класах, що видно зі збільшення середньої повноти на 0.004.

Помітно також зменшення середньої F-заходи за всіма алгоритмами менше, ніж на 0.1. однак, оскільки ця характеристика комплексна і залежить як від точності, так і від повноти, то найбільший інтерес для розгляду в даному випадку

представляє показник Precision. Також видно, що зміни в алгоритмах SVM і Naive Bayes помітно менше, ніж у інших чотирьох методів, і до того ж вони і перш показували недостатню якість класифікації, тому в подальшому розглядалися наступні чотири алгоритму - One Rule, C4.5, Random Forest і AdaBoost.

У табл. 3.2 наведено порівняння показника Precision по всіх класах в чотирьох алгоритмах до і після додавання в тестову вибірку фонового трафіку.

Таблиця 3.2 - Порівняння критерію Precision до і після додавання фонового трафіку

Алгоритм класифікації	Клас трафіка	Без фонового трафіка	З урахуванням фонового трафіка
One Rule	Skype	0,440	0,335
	Steam	0,858	0,794
	Torrent	0,763	0,550
	Facebook	0,385	0,330
	Youtube	0,590	0,498
	Skype	0,570	0,353
	Steam	0,973	0,951
	Torrent	0,931	0,917
	Facebook	0,691	0,544
	Youtube	0,697	0,522
Random Forest	Skype	0,553	0,364
	Steam	0,970	0,964
	Torrent	0,911	0,861
	Facebook	0,727	0,517
	Youtube	0,771	0,563
AdaBoost+ One Rule	Skype	0,508	0,375
	Steam	0,932	0,906
	Torrent	0,781	0,612
	Facebook	0,466	0,325
	Youtube	0,591	0,447

Для наочності ці значення зведені у вигляді двох гістограм, представлених на рис.3.2 і рис. 3.3.

За результатами в табл.3.2 та зображеним гістограми помітно, що при додаванні фонових потоків сильно впало якість класифікації з точки зору параметра Precision. Значно погіршилася, найбільша різниця помітна в класі SKYPE. Також значно гірше усіма алгоритмами стали визначатися класи Facebook

і YOUTUBE. У класі TORRENT помітне падіння точності визначення видно тільки у One Rule і пов'язаного з ним же AdaBoost.

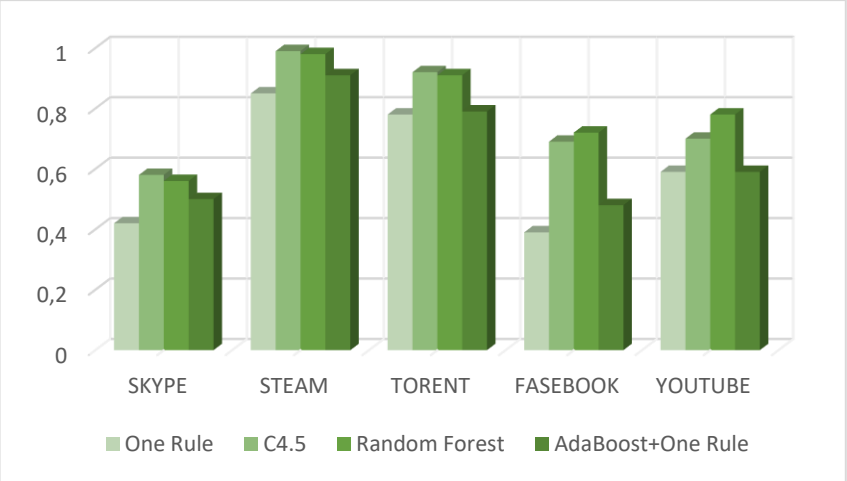


Рисунок 3.2 - Порівняльні гістограми за критерієм Precision при відсутності фонового трафіку

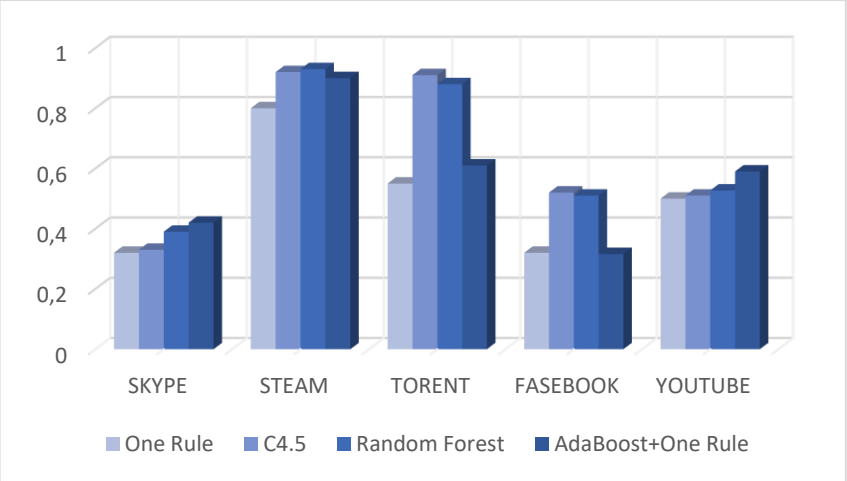


Рисунок 3.3 - Порівняльна гістограма за критерієм Precision при наявності фонового трафіку

Найменш помітні зміни в класі STEAM, де C4.5 і Random Forest практично не показали зменшення точності. Трохи гірше результати у One Rule.

Зміни в критерії Precision легко пояснити. Раніше вже зазначалося, що при такій схемі навчання, коли в тренувальній вибірці відсутні дані про тих потоках,

які присутні в тестовому наборі, класифікатор розподіляє невідомий трафік по інших класів, за якими і проводиться класифікація.

Критерій точності Precision показує, який відсоток з усіх потоків, які були віднесені до певного класу, дійсно йому належать. При додаванні стороннього трафіку в тестову вибірку моделі алгоритмів закономірно помилково розподіляють потоки по інших класів, і у кожного типу трафіку падає показник якості в прямій залежності від того, яка кількість потоків було до нього віднесено [23]. Звідси і щодо невеликої різниці в точності класифікації алгоритмами SVM і Naive Bayes, оскільки і без фонового трафіку ці моделі відносили основну масу потоків до класів TORRENT і Faysbook.Для повноти зіставлення якості класифікації вибірки без фонового трафіку і з його наявністю на рис. 3.4 наведені ROC - криві алгоритмів за деякими класами.

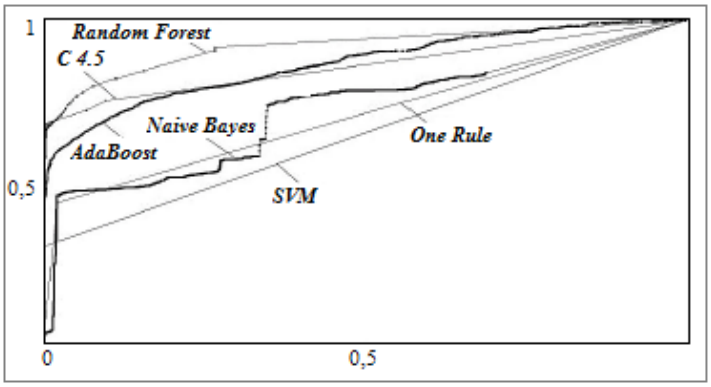


Рисунок 3.4*а* - ROC-криві алгоритмів за класом STEAM

Видно невеликі відмінності від попередніх результатів. Наприклад, помітно зменшення якості класифікації об'єктів даного типу трафіку алгоритмами Naive Bayes і AdaBoost. Також менш опуклою стала крива моделі C4.5.

У випадку з класом TORRENT видно деяка різниця з початковим тестуванням в плані якості роботи алгоритмів. По зображенню видно явне поліпшення достовірності в класифікації AdaBoost і One Rule, криві яких стали ближче до ідеальної моделі. Трохи краще себе показав і алгоритм C4.5.

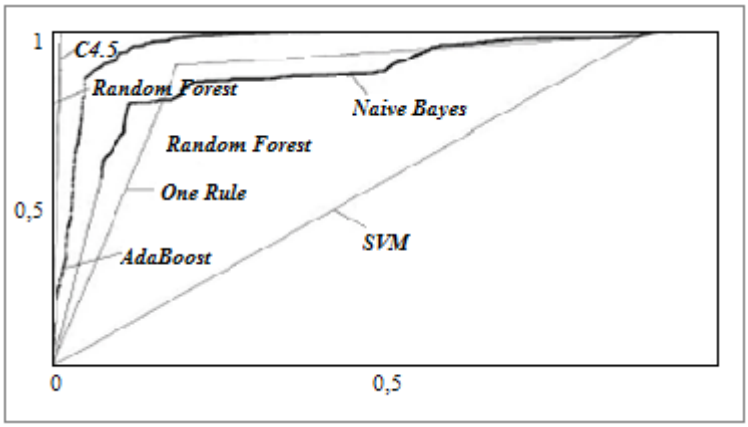


Рисунок 3.4*б* - ROC-криві алгоритмів за класом TORRENT

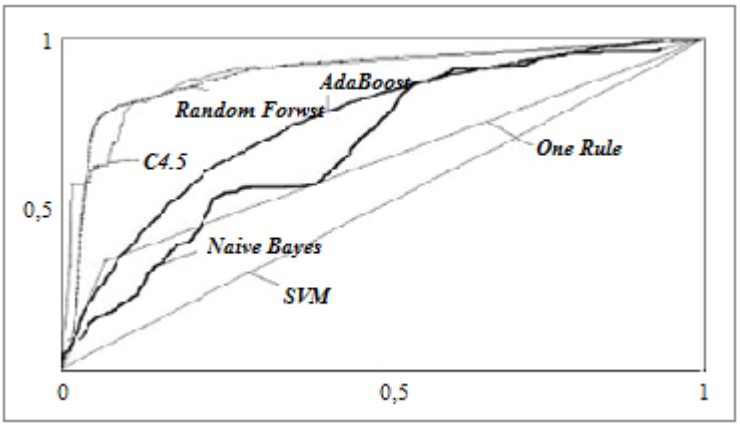


Рисунок 3.4 *в* - ROC-криві алгоритмів за класом Faysbook

По класу Faysbook помітно погіршення якості класифікації, оскільки графіки таких алгоритмів, як Naive Bayes і AdaBoost трохи опустилися на координатної осі, тим самим і зменшивши значення площали AUC. Невеликі зміни є в роботі Random Forest, тому що його характеристика стала кілька поступатися якістю C4.5. Криві One Rule і SVM не показують жодних видимих змін в якості класифікації.

В цілому можна бачити, що загальний вид ROC - кривих не сильно відрізняється при тестуванні без фонових об'єктів. Однак за деякими класами в очі кидаються відмінності в тих алгоритмах, де до розглянутого класу було віднесено велику кількість потоків фонового класу, що відповідним чином впливало на рівень показника помилково - позитивних рішень і відображалося на представлених графіках.

В цілому, за результатами навчання та тестування різних алгоритмів машинного навчання для класифікації трафіку, можна сказати, що алгоритми Random Forest і C4.5 показали найкращі результати.

Класифікація фонового трафіку показала, що алгоритми машинного навчання з учителем, якість роботи яких повністю ґрунтується на повноті і достовірності навчальних вибірок даних, які не здатні визначити нові, невідомі дані, що веде до неминучих і критичним помилок класифікації.

Природним розвитком в даному напрямку є застосування інших алгоритмів навчання або ж методів кластеризації, які як раз і призначені для визначення і розмежування невідомих типів трафіку, які вже потім аналізуються і класифікуються.

3.2 Аналіз неконтрольованої кластеризації інтернет-трафіку

Технології кластеризації. При наявності непоміченого набору трафіку, або ж змішаних даних, що складаються з великої кількості помічених примірників потоків і невеликого числа непомічених об'єктів, можливе застосування технік кластеризації для розподілу потоків в деяке число груп відповідно зі схожістю статистичних показників.

Технології кластеризації дуже важливі для класифікації трафіку, тому що на практиці отримання повного позначеного набору даних для навчання є складним і трудомістким процесом. Перевагою є створення нових шаблонів які представлятимуть раніше невідомі додатки або будь-які зміни в існуючих класах.

На рис. 3.5 зображена система навчання класифікаторів інтернет-трафіку на основі техніки кластеризації.

В першу чергу, зібраний трафік в формі мережевих пакетів збирається в першу чергу, зібраний трафік в формі мережевих пакетів збирається в мережеві потоки, грунтуючись на вищезазначених п'яти параметрах для їх ідентифікації. Для здійснення кластеризації кожен з потоків описується вимірюваними значеннями

заздалегідь визначеного набору властивостей, тобто точкою x = (x_1, ..., x_d) в d - вимірному просторі ознак, де d - кількість ознак.

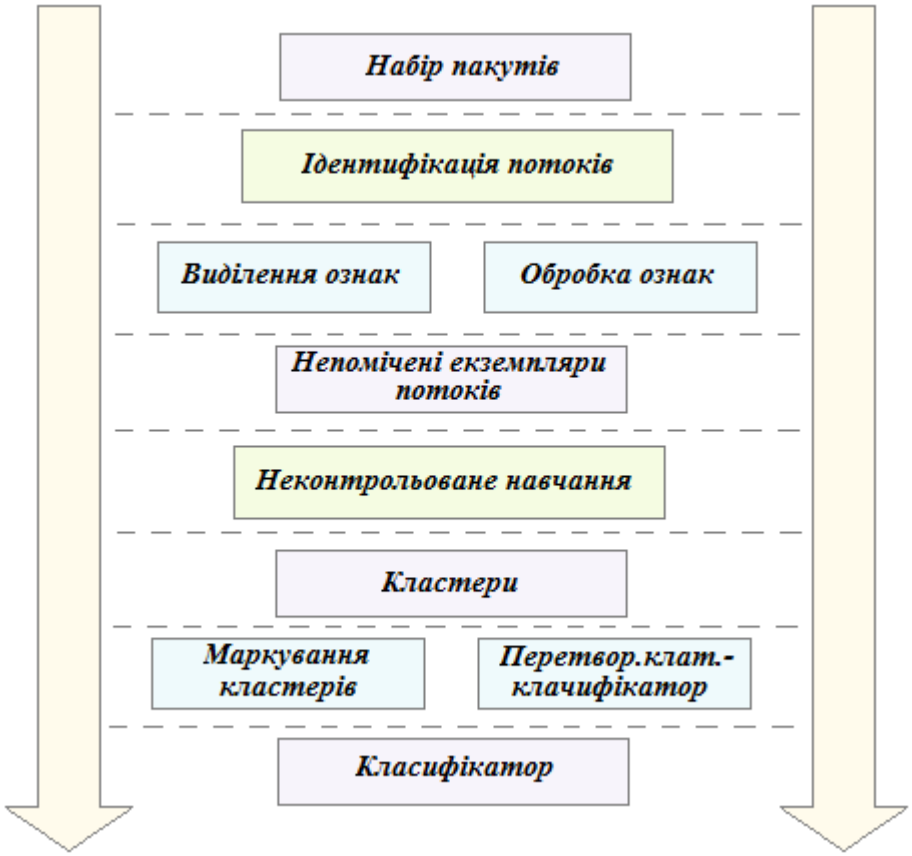


Рисунок 3.5 - Система кластеризації мережевого трафіку

На цій стадії також може вироблятися деяка попередня обробка атрибутів, на кшталт їх відбору та трансформації.

Уявімо підхід до кластеризації мережевого трафіку на основі наближення алгоритму Random Forest.

Будемо як і раніше вважати потоки двонаправленими послідовностями мережевих пакетів, якими обмінюються два кінцевих вузла, і які можуть бути ідентифіковані параметрами {srcIP, dstlP, srcPort, dstPort. Protocol}. Що стосується набору ознак трафіку, то будемо вважати, що вони описуються простими властивостями.

До цього етапу непомічені вектори властивостей можуть бути оброблені алгоритмами кластеризації, які поділяють вхідні дані на якийсь відстань. Метою

цього етапу є створення чистих кластерів. Ідеальним буде випадок, якщо потоки кожного кластера належать тільки одному з додатком.

Для класифікації трафіку в режимі on-line кластери трафіку повинні бути пов'язані з деякими конкретними класами додатків, щоб на їх основі могли бути побудовані фінальні класифікатори. різні методики подібної маркування були розроблені в роботі [24].

Найбільш простим способом є ручна класифікація кількох потоків в кожному кластері з подальшою розміткою цих кластерів відповідно до більшості зберігаються в них потоків. Інший спосіб має на увазі подачу на вхід алгоритму кластеризації змішаних даних, що складаються з великої кількості помічених примірників потоків і невеликого числа непомічених.

В результаті промарковані потоки, що містяться в кластерах, можуть бути використані для найменування кластерів.

Розглянемо випадок отримання максимально чистих кластерів. З цією метою будуть використані відповідний набір трафіку і евристика більшості для іменування результуючих кластерів і обчислення загальної чистоти кластерів.

У методі кластеризації, заснованим на алгоритмах «відстані», кластери представлені якимись центральними точками (центроїдами), а спостережувані об'єкти відносяться до такої найближчій точці відповідно до деякої метрикою відстані (наприклад. Евклідовому відстань). Для прикладу, метод K-Means [103] ітеративно відносить об'єкти до кластерів з найближчим середнім, і потім перетворюється в локальний мінімум суми квадратів відстаней між кожним об'єктом і центром кластера.

У методі кластеризації, заснованому на ймовірності, об'єкти з певною ймовірністю можуть бути віднесені до кожного кластеру і зазвичай до найбільш ймовірного.

Найвідомішою ймовірнісної моделлю для кластеризації є Гаусовські, де кожен кластер представлений Гаусовским розподілом, параметри щільності якого фіксовані, але невідомі.

Застосування алгоритму максимальної правдоподібності (EM - Expectation-max imization) [25] дозволяє випадковим чином форматувати параметри, а потім ітеративно оптимізувати їх, щоб вони найкращим чином описували дані за якими спостерігають.

Традиційні алгоритми кластеризації грунтуються на шаблонах в Евклідовому просторі ознак і припущенні, що всі ознаки мають однакову вагу в сенсі ідентичного вкладу в результат кластеризації. Для прикладу, кластери, отримані методом K-Means, являють собою сфери і не мають чітких меж.

Розглянемо кластеризацію мережевого трафіку, який базується на алгоритмі Random Forest. Подібна кластеризация успішно застосовується в біометричних дослідженнях для пошуку значущих кластерів в даних геномної послідовності.

Random Forest один з популярних контрольованих алгоритмів навчання, який показав найкращі результати в завданні класифікації трафіку. В доповнення до його високої точності, Random Forest має деякі додаткові властивості. Наприклад, він дає оцінку помилки в процесі навчання, а також оцінку близькості між усіма парами вхідних точок даних. Остання властивість дозволяє використовувати його для кластеризації вхідних даних.

Випадковий ліс - ансамбль індивідуальних дерев класифікації. Для класифікації точки даних, кожне дерево в лісі визначає клас окремо (так зване голосування за клас), а ліс визначає фінальний клас на основі більшості голосів. Для побудови лісу потрібно уточнити два базових параметра: кількість дерев (n) і число змінних, які використовуються для поділу вузлів (m). яке повинно бути набагато менше числа вхідних змінних.

Для вирощування кожного дерева, алгоритм спочатку конструює кореневої вузол шляхом випадкового відбору N точок даних з заміною з навчальної вибірки, де N - розмір тренувального набору. Потім він ітеративно розділяє вузли на основі ш змінних, які випадковим чином вибираються з вхідних змінних, а критерієм для поділу служить індекс Джіні.

Дерева ростуть настільки великими, наскільки це можливо, без укорочення (відсікання гілок). Під час початкового процесу відбору при навчанні близько

третини всіх точок даних залишаються поза набором даних, які отримали назву out-of-bag (OOB) дані (поза мішка). Ці OOB дані можна використовувати для оцінки помилки класифікації, яка, як було показано в багатьох тестах, безстороння.

Коли ліс побудований, всі дані можуть бути пропущені через дерева і обчислені міри близькості для кожної пари точок даних. Якщо дві точки потрапляють в один листової вузол, їх близькість збільшується на одиницю. В кінці процесу, близькості до нормальних значень шляхом ділення на число дерев, а близькість між точкою і нею самою вважається за одиницю. В цьому випадку, створюється симетрична матриця близькості P, де кожен елемент має значення в інтервалі [0, 1].

При наявності непомічених даних неможливо безпосередньо побудувати дерево. Для виділення показника близькості в цьому випадку потрібно визначити завдання штучної класифікації, при якій випадковий ліс будується для відділення вихідних даних від штучних. Підсумковий ліс і значення близькості в великій мірі залежать від складу штучних примірників.

3.3 Аналіз алгоритмів кластеризації та їх порівняння

Алгоритм k-среднпх. Програмний клас KMeans в бібліотеці scikit-learm для Python здійснює розбиття вхідних даних на кластери за допомогою однойменного алгоритму. Розглянемо основні вхідні параметри даного класу:

- n_clusters - кількість кластерів: як уже говорилося алгоритму, к-Means на вхід потрібно заздалегідь задати кількість кластерів (кількість пентроідов);

- init - метод ініціалізації, цей параметр може приймати три значення: k-means++ - розумний пошук початкових значень Центроїд кластерів; random - випадковий вибір до точок з вхідних даних в якості початкових центрів; в третьому випадку центроїди задаються врмчнмю у вигляді масиву;

- n_init - число раз, яке буде виконуватися алгоритм на різних наборах початкових центроїдів. Кінцевим виходом буде кращий результат.

Після завершення алгоритму приймають значення такі атрибути:

- cluster_centers - масив, що містить координати центрів утворених кластерів;

- labels - мітки для точок, отримані в результаті кластеризації;

- inertia - сума відстаней від точок до найближчого центру. Цей атрибут показує, наскільки «купчасто» точки розташовуються близько центрів кластерів. Цей параметр зменшується зі збільшенням кількості поставлених кластерів.

Для можливості подальшого порівняння результатів роботи контрольованого і неконтрольованого навчання був узятий той же самий набір даних, з тими ж інформаційними ознаками, що і для класифікації. Склад вибірки наведено в табл.3.3.

Таблиця 3.3 - Склад датасета

Протокол	Кількість потоків
SSL	1415
HTTP	1337
BitTorrent	1172
Skype	990
Steam	979

Як було вже сказано, в алгоритмі *k* -Means є вхідний параметр *k*. Цей вхідний параметр, являє собою число непересічних розділів, використовуваним алгоритмом.

Для нашого набору даних очікується, що для кожного класу трафіку буде принаймні один кластер. Крім того, через різноманітність трафіку в деяких класах, таких як, наприклад, HTTP (який може включати перегляд веб-сторінок, завантаження файлів, потокову передач), очікується наявність більшої кількості кластерів. Тому алгоритм *k*-Means оцінювався з *k*, спочатку рівним 6, що відповідає кількості певних нами класів, а потім цей параметр збільшувався для кожної наступної кластеризації.

Для кожного значення параметра до кластеризация запускалася кілька разів. і бралося середнє значення для кожної метрики. На рис. 3.6-3.10 показані залежності метрик від параметра *k*.

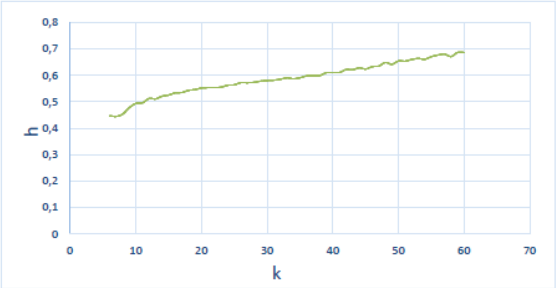


Рисунок 3.6 - Залежність однорідності від параметра *k*

Як видно, однорідність зростає із збільшенням кількості кластерів (k). Що цілком логічно. Даний параметр досягне 1 при k дорівнює кількості об'єктів у вибірці, тобто коли кожен об'єкт буде поміщений в окремий кластер.

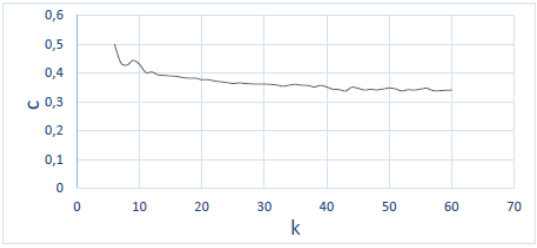


Рисунок 3.7 - Залежність повноти від параметра *k*

Значення повноти максимально при збігу *k* з реальною кількістю кластерів, і, що цілком логічно, зменшується з ростом *k*. Максимальне значення ця метрика прийме при *k* = 1 тобто коли об'єкти всіх класів будуть помішані в один єдиний кластер.

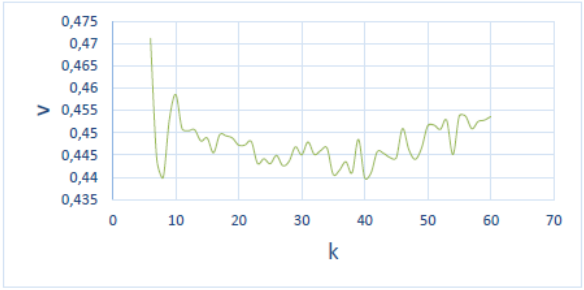


Рисунок 3.8 - Залежність v-заходи від параметра *k*

Як видно, значення v-заходи максимально при збігу параметра *k* з реальною кількістю класів у вибірці (*k* = 6).

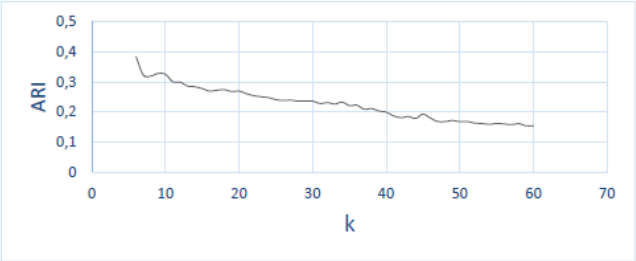


Рисунок 3.9 - Завісімость незміщенного індексу Ранда від параметра *k*

Незміщений індекс Ранда також максимальний при *k* = 6, також слід помітити, що значення ARI позитивно, що говорить про правильну кластеризації, відмінною від випадкового призначення міток.

Коефіцієнт силуету практично не змінюється на відрізку [6; 33] і приймає максимальне значення при *k* = 10, що каже про те, що найбільшою розділених і згруповані кластери мають, коли їх кількість дорівнює 10.

З вищенаведених графіків можна зробити висновок, що алгоритм *k* -Means дійсно виділяє в наших даних 6 кластерів. У табл.3.4 наведені співвідношення справжніх класів і шести певних алгоритмом *k-Means* кластерів.

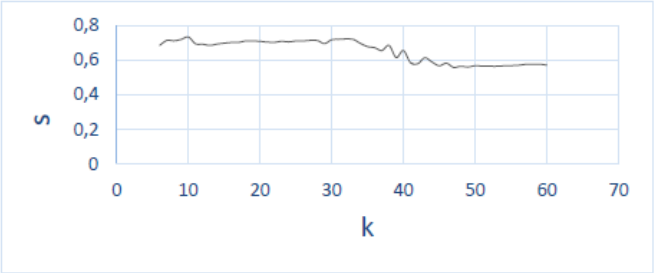


Рисунок 3.10 - Залежність коефіцієнта силуету від параметра *k*

Таблиця 3.4 – Співвідношення дыйсних класыв ы кластерыв *k-Means* при k=6

	0	1	2	3	4	5
SSL	-	1110	231	-	74	-
BitTorrent	188	-	-	370	154	460
HTTP	1112	-	168	-	57	-
Skype	205	31	345	326	70	13
DNS	-	-	1209	-	57	-
Steam	709	-	156	102	12	-

Для наочного уявлення даної інформації на рис.3.11 подано діаграму з процентним співвідношенням класів в кожному з шести виділених алгоритмом кластерів.

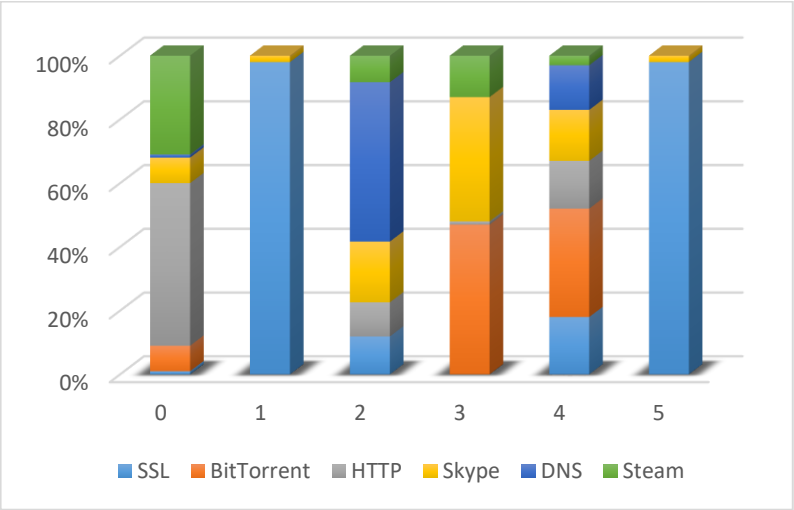


Рисунок 3.11 - Процентне співвідношення класів в кластерах при k = 6

Як видно з даної гістограми, можна стверджувати, що кластери 1 і 5 практично повністю співвідносяться з класами SSL і BitTorrent відповідно, в кластері 2 превалює клас DNS, а в кластері 0 - HTTP, кластер 4 вийшов досить змішаним і, як видно з табл. 3.4 самим нечисленним з усіх запропонованих.

Для порівняння подібні характеристики були знайдені і для $k = 10$, результати представлені в табл. 3.5 та на рис. 3.12.

Таблиця 3.5 - Співвідношення справжніх класів і кластерів *k-Means* при k=10

	0	1	2	3	4	5	6	7	8	9
SSL	-	231	689	-	74	421	-	-	-	-
BitTorrent	235	37	-	128	49	-	93	135	285	120
HTTP	-	168	557	-	57	555	-	-	-	-
Skype	183	344	17	37	61	8	205	12	12	111
DNS	-	1209	-	-	57	-	-	-	-	-
Steam	101	156	-	710	12	-	-	-	-	-

Як видно з даної діаграми, кластери 7, 8 практично повністю співвідносяться з класом BitTorrent; 0, 6, і 9 ділять між собою класи BitTorrent і Skype, кластери 2 і 5 - класи HTTP і SSL. в кластері 3 превалює Steam, а в 1 – DNS, кластер 4 вийшов змішаним і невеликим за обсягом.

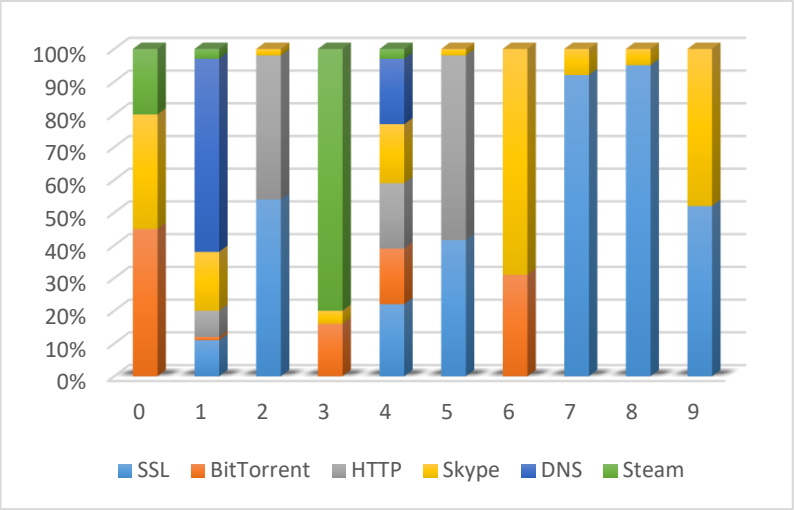


Рисунок 3.12 - Процентне співвідношення класів в кластерах при $k = 10$

Таким чином, можна зробити висновок, що алгоритм *k-Means* справляється з кластеризацією потоків мережевого трафіку досить нерівномірно. одні класи він виділяє досить чітко, інші змішує між собою, що говорить про їх схожості і близькій відстані один від одного.

Алгоритм DBSCAN. DBSCAN - алгоритм просторової кластеризації на основі щільності. Знаходить центри високої щільності і розширює їх в кластери. Добре підходить для даних, що містять кластери з аналогічною щільністю.

У бібліотеці scikit-leam для Python є програмний клас dbscan. який здійснює кластеризацию вхідних даних за допомогою даного алгоритму. Розглянемо основні вхідні параметри класу:

- eps - максимальна відстань між двома об'єктами, при якому вони вважатися такими, що однією околиці;

- min_samples - кількість об'єктів (або загальна вага) в околиці точки, при якому вона вважається центром (сама точка теж входить в це кількість);

- p - параметр метрики Мінковського. використовуваної для обчислення відстані між точками;

- algorithm - алгоритм, який буде використовуватися при обчисленні відстаней між-точками і пошуку найближчих сусідів (ball_tree, kd_tree, brute);

- n_jobs - кількість паралельних обчислень. Якщо -1, тоді кількість потоків обчислень встановлюється рівною кількості ядер ЦП.

Після завершення алгоритму можна отримати наступні атрибути:

- core_sample_indices - індекси об'єктів - центрів класерів;

- components - копія кожного знайденого об'єкта-центру;

- labels - мітки кластерів для кожної точки набору даних.

Нагадаємо, що DBSCAN має два важливих вхідних параметра (min_sample, eps). В ході експерименту застосовувалися різні комбінації цих параметрів.

Параметр min_sample приймав значення 10 і 50. Відстань eps змінювалося від 1 до 500. На рис. 3.13 і рис. 3.14 представлені результати для різних комбінацій значень (min_sample, eps) для нашого набору даних. Як і слід було очікувати, при min_sample рівному 10 досягаються кращі результати, ніж при min_sample рівному 50, так як утворюються більш дрібні кластери.

Оцінка коефіцієнта силуету для даного алгоритму не проводилася, оскільки, як уже було сказано раніше, для алгоритмів, заснованих на щільності, вона приймає низькі значення.

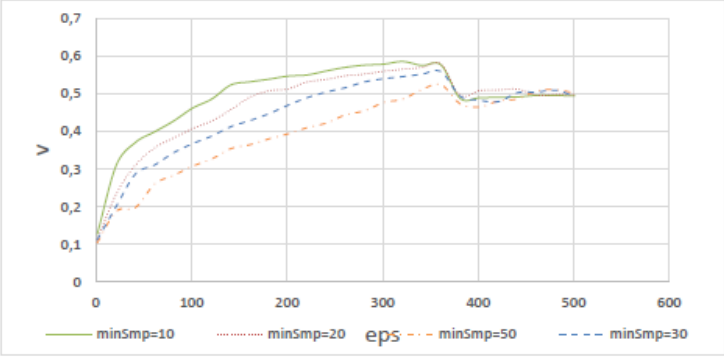


Рисунок 3.13 - Залежність v-заходи від параметра eps для різних значень min_samples

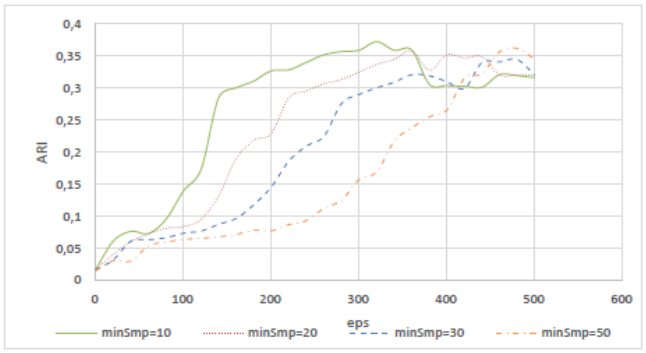


Рисунок 3.14 - Залежність незмішаного індексу Ранд від параметра *eps*, для різних значень min_samples

На рис. 3.15 показана залежність кількості виділених алгоритмом кластерів від параметра EPS.

Як видно, зі збільшенням порогу eps, кількість кластерів зменшується.

За результатами експерименту, найкращий результат алгоритм DBSCAN показав при значеннях min_simples = 3 і eps = 320. при цьому було виділено 60 кластерів.

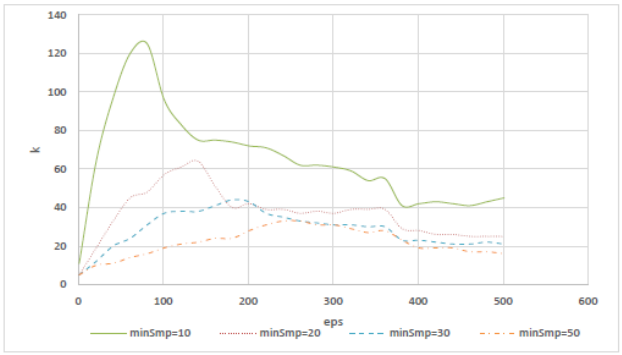


Рисунок 3.15 - Залежність кількості кластерів від параметра *ep*% для різних значень min_samples

На рис. 3.16 представлений склад всіх 60 кластерів. З гістограми видно, що найбільш якісно кластеризація методом DBSCAN спрацювала для класів трафіку DNS і BitTorrent. об'єкти кожного з цих класів майже повністю зосереджені в одному кластері. Також непогано алгоритм впорався з кластеризацией трафіку HTTP і SSL. Класи Steam і Skype сильно розкидані по багатьом кластерам.

Зробимо порівняння якості кластеризації алгоритмів k-Means і DBSCAN. Порівняння проведемо за метрикою k-Means. яка ґрунтується на двох інших метриках - однорідності і повноті. Для обох алгоритмів візьмемо найбільш високе значення метрики, отримане під час експерименту. Результат порівняння наведено в табл.3.6.

Таблиця 3.6 - Порівняння алгоритмів k-Means і DBSCAN

	Значення v-Міри	Значення незміщеного індекса Ранда	Кількість кластерів
k-Means	0,46	0,32	10
DBSCAN	0,58	0,37	60

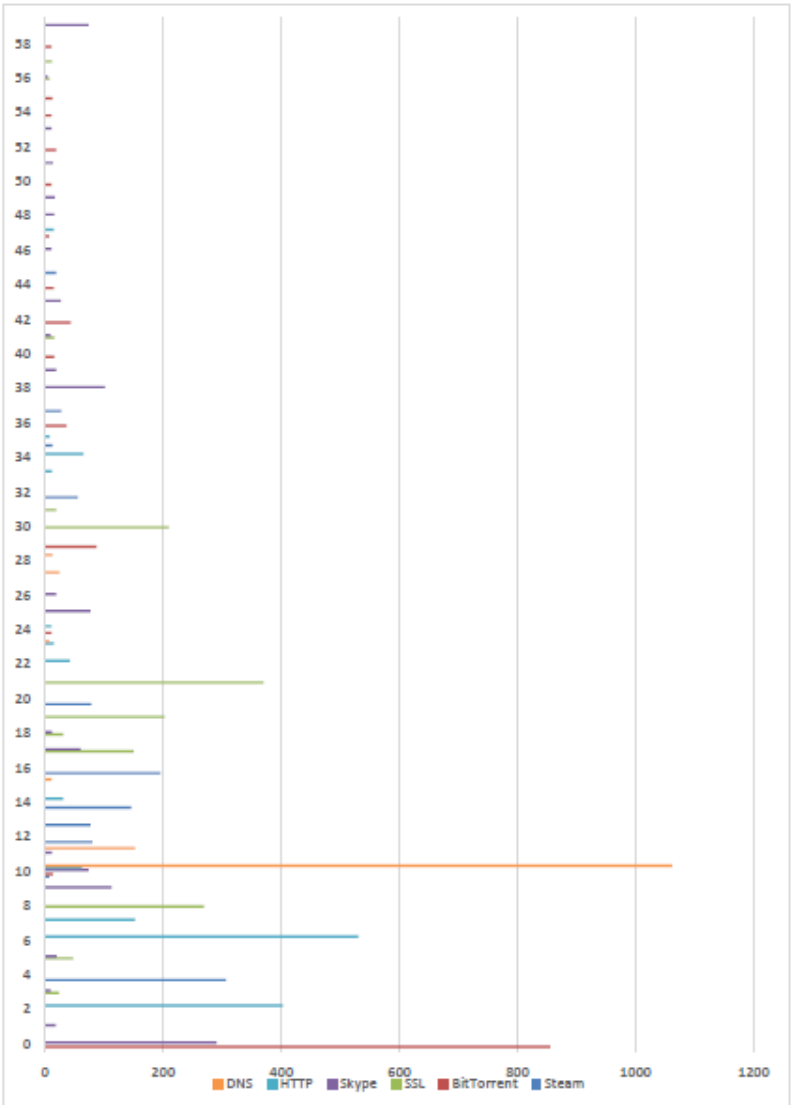


Рисунок 3.16 - Розподіл класів по кластерам після кластеризації DBSCAN

Як видно, алгоритм DBSCAN продемонстрував кращі результати, проте сильно помилився в кількості кластерів в представлених даних, *k-Means* показав найкращі значення оцінок при параметрі $k = 10$. що більш наближене до дійсності (в датасета представлено 6 класів трафіку).

3.3.1 Порівняльний аналіз алгоритмів контрольованого і неконтрольованого навчання

Для можливості провести порівняльну характеристику алгоритму Random Forest і алгоритмів навчання без учителя *k-Means* і DBSCAN необхідно знайти загальну оцінку для цих типів. Такою оцінкою може служити F-міра. Обчислити F-міра можна і для алгоритмів кластеризації, відміну полягає в методі обчислення величин точності (precision) і повноти (Recall).

$$precision(i,j) = \frac{N_{ij}}{N_j}$$

$$recall(i,j) = \frac{N_{ij}}{N_i}$$

Де N_j - кількість елементів в кластері j, N_j - кількість елементів класу i, а N_{ij} - число елементів класу i в кластері j.

Для класу i і кластера j F-міра обчислюється відповідно до наступного висловом:

$$F(i,j) = \frac{2 \cdot recall(i,j) \cdot precision(i,j)}{recall(i,j) + precision(i,j)}$$

Підсумковим значенням F-міри є середньозважене значення точності і повноти для кожного класу, як показано в рівнянні.

$$Fscore = \frac{\sum_i (|i| \cdot F(i))}{\sum_i |i|}$$

де $|i|$ – розмір – го класу

У табл. 22 представлено дані метрики представлені для алгоритм *k-Means*.

Таблиця 3.7 - Значення метрик для алгоритму *k-Means*

	Точність	Повнота	F-міра
BitTorrent	0,97	0,359	0,524
DNS	0,565	0,955	0,71
HTTP	0,493	0,832	0,619
SSL	0,492	0,784	0,604
Skype	0,161	0,348	0,22
Steam	0,641	0,724	0,68

Графічне представлення даних метрик для всіх класів приведено на рис. 3.17. Як видно, найгірше даний алгоритм впорався з класифікацією трафіку, що генерується протоколом додатком Skype.

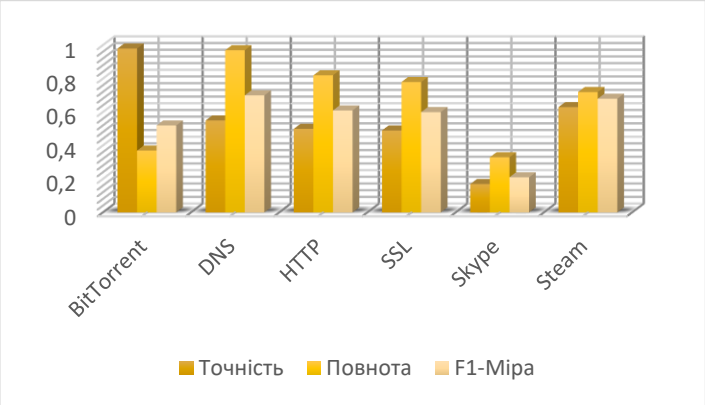


Рисунок 3.17 - Точність, повнота, Fl-мера алгоритму k-Means

У табл. 23 наведені дані метрики для алгоритму DBSCAN для випадку, коли алгоритм запропонував 60 кластерів. На рис. 3.18 наведено графічне представлення даних метрик. Як видно, алгоритм DBSCAN також найгірше впорався з протоколом Skype.

Таблиця 3.8 - Значення метрик для алгоритму DBSCAN

	Точність	Повнота	F1-міра
BitTorrent	0,747	0,75	0,748
DNS	0,873	0,833	0,855
HTTP	1	0,396	0,567
SSL	1	0,27	0,429
Skype	0,25	0,29	0,27
Steam	1	0,31	0,47

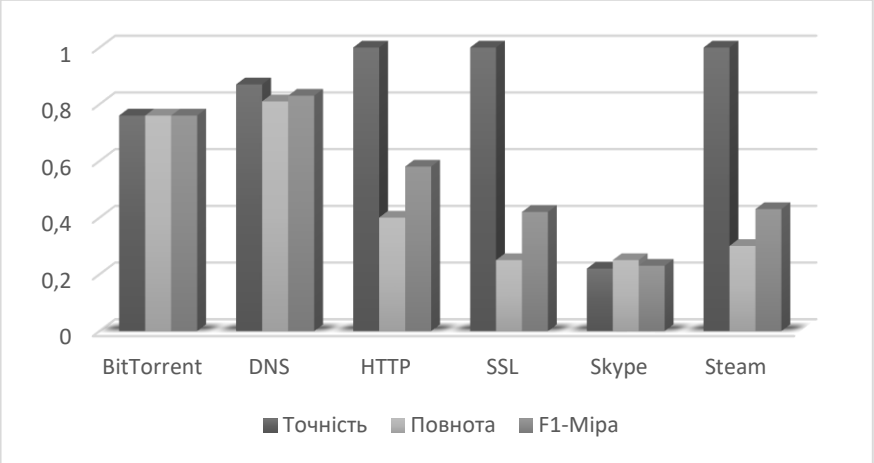


Рисунок 3.18 - Точність, повнота, F1-мера алгоритму DBSCAN

На рис. 3.19 представлено порівняння результатів трьох досліджуваних алгоритмів: Random Forest (при наявності фонового трафіку) - як представника методів навчання з учителем, і алгоритмів неконтрольованого навчання *k-Means* і DBSCAN. Порівняння здійснюється по F-міру.

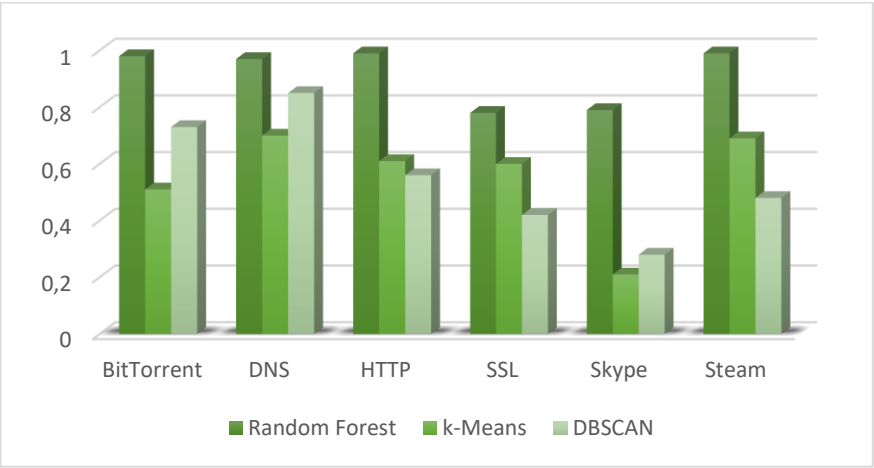


Рисунок 3.19 - Значення F1-міри заходи для досліджуваних алгоритмів

Як можна побачити з даної діаграми, методи неконтрольованого навчання значно поступаються в якості алгоритму навчання з учителем Random Forest, з чого можна зробити висновок, що даний алгоритм є більш відповідним для завдання класифікації мережевого трафіку.

Однак слід звернути увагу, що такий метод не придатний для роботи в режимі online, що є досить критичним при ідентифікації додатків, що генерують мережеві

потоки, особливо в задачах забезпечення інформаційної безпеки. Мережевий трафік неоднорідний і досить швидко змінюється. Це вимагає постійного перенавчання алгоритму, неконтрольовані алгоритми навчаються самостійно в процесі роботи.

Якість роботи алгоритмів k-Means і DBSCAN залежить від типу класифікуючого трафіку. Для протоколів BitTorrent і DNS більш придатним виявився алгоритм DBSCAN. для трафіку SSL і Steam - k-Means. Результати для класів HTTP і Skype у даних алгоритмів досить близькі.

3.3.2 Порівняння алгоритмів кластеризації заснованих на RF з класичними алгоритмами EM і K-Means

Порівняємо метод кластеризації, заснований на RF, з двома класичними алгоритмами EM і K-Means. Всі задіяні алгоритми брали в як вхідний параметр число кластерів К. На наборах трафіку проводилася серія експериментів кластеризації з різним значенням К = 10. 20. .... 90. 100. 200. і результати отримані у вигляді функції від К.

Для кожного набору процедура кластеризації повторювалася десять разів з різними вхідними даними. З цією метою були отримані десять випадкових вибірок для кожного набору трафіку, так що кожен результат був розподілених на 100 експериментів кластеризації.

Як специфічних параметрів RF кластеризації, використовувалися наступні настройки по - замовчуванням. Кількість ознак, які випадковим чином вибиралися для поділу кожного вузла, дорівнювало трьом (m = 3). Крім того, матриця близькості усереднена по п'ятдесяти незалежним випадковим лісам (T = 50), кожен з яких складається з двох тисяч дерев (n = 2000)

Для оцінки результуючих кластерів застосовувалася евристика більшості метою маркування кластерів. Кожен кластер позначався найбільш популярним додатком, що знаходяться в ньому. Нехай $C = \{C_1,\ldots,C_k\}$ це кластери, а $A = \{A_1,\ldots, A_т\}$ це реальні класи додатків.

У цьому випадку функція маркування La: C → A асоціює мітку класу з кожним кластером.

$$L_a = (C) = \arg max_{A_j \epsilon A} \sum x_i \epsilon\, c\, \theta\big(\alpha(x_i), A_j\big) \quad (3.1)$$

де α(x) повертає реальний клас додатки даного потоку x, а θ(X,Y) визначається як

$$\theta(s,t) = f(x) = \begin{cases} 1, & \text{якщо } s = t \\ 0, & \text{інакше} \end{cases} \quad (3.2)$$

Після маркування результати кластеризації вимірювалися із застосуванням загального показника точності і Е-заходи по кожному класу. спочатку, оцінювалася загальна точність потоків (або чистота кластерів) як відношення кількості правильно позначених потоків до загальної кількості потоків в кластері.

Рівень точності і повноти для аналізованого класу A оцінювався по формулами:

$$Precision = \frac{TP}{TP+FP};$$

$$Recall = \frac{TP}{TP+FN}.$$

F-міра. представляє собою баланс між точністю і повнотою є їх гармонійним середнім, обчислювалася за формулою:

$$F1 = \frac{2*Precision*Recall}{Precision+Recall}.$$

Результуючі показники точності для експериментів кластеризації показані на рис.3.20.

Як видно з отриманих графіків рис. 3.20 рівні точності для всіх розглянутих алгоритмів зростають разом із збільшенням кількості кластерів. Ця тенденція продовжується до тих пір, поки число К не стає рівним 100. Для алгоритмів K-Means і EM кластери точні на 69% і 85%, в той час точність RF кластерів досягає 90%. При подальшому збільшенні До точність всіх алгоритмів зростає незначно. Нарешті, при К рівному максимальному оскільки він розглядався значенням (К = 200), K-Means, EM і RF кластеризація створює кластери з точністю 72%, 88% і 93% відповідно.

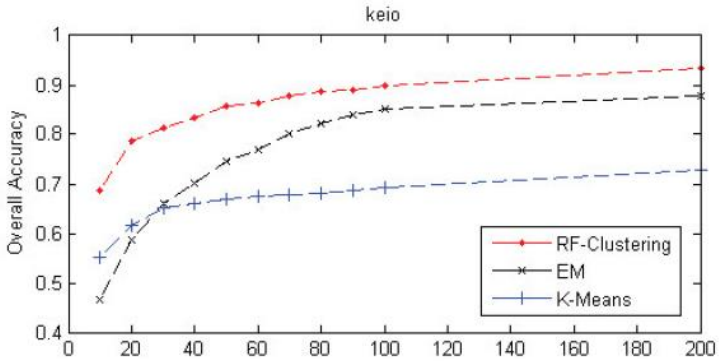


Рисунок 3.20 – Загальні показники точності

Рисунок 3.21 – Показник F- міра для кожного класу [82]

Аналіз отриманих результатів показує, що RF кластеризація працює краще *k-Means* і EM на 17 - 21% і 7% -10% відповідно по точності при різній кількості кластерів в наборі даних Wide-09.

Одним з переваг RF кластеризації є досягнення високих показників точності навіть при малій кількості кластерів. Наприклад, рівня точності в 85% RF може досягти при К = 100, тоді як алгоритм EM вимагає мінімум К = 200.

Рисунок 3.21 показує продуктивність по кожному класу в рамках усередненого критерію F-міри. отриманого з трафіку Wide-09 при К = 50.

Можна бачити, що кластеризація RF досягає найкращого результату для всіх класів додатків за винятком HTTP. Зокрема. F-міра для DNS дорівнює 100%, 68% для HTTP і 80% - 90% для всіх інших додатків.

На практиці реальне число кластерів заздалегідь невідомо і компромісу в виборі відповідного значення добитися не вдається. З одного боку, широко поширена думка [21]. що деякі класи додатків повинні мати кілька показових кластерів, тому що шаблони трафіку одного єдиного додатка можуть бути різні. Візьмемо, наприклад, прикладної протокол HTTP. який застосовується для цілого переліку діяльності, включаючи веб - серфінг, передачу даних, передачу мультимедійного трафіку і т.д .. Виходячи з цього можна очікувати кілька кластерів, що представляють трафік для кожного випадку використання цього додатки. З іншого боку, занадто велика кількість кластерів небажано, оскільки, може виникнути крайній випадок наявності одного кластера для одного об'єкта. Незважаючи на те. що це може збільшити точність кластеризації, отримані підсумкові кластери будуть марні. Збільшення числа кластерів також збільшує обчислювальні витрати і час виконання алгоритму.

3.4 Приклад реалізації та застосування програмного забезпечення для вирішення задачі класифікації інтернет-трафіку методом машинного навчання

Для розробки програмного забезпечення була вибрана мова програмування Java. Java є об'єктно-орієнтованою мовою програмування, основною особливістю якого є те. що всі програми при виконанні транслюються в спеціальний байт-код. виконуваний віртуальною машиною Java (JVM - Java Virtual Machine).

Віртуальна машина під час кожного виклику необхідних інструкцій здійснює переклад з текстової форми байт-коду в виконувані коди відповідного процесора. Цей процес називається інтерпретацією. До переваг даного підходу можна віднести незалежність байт-коду від операційної системи і устаткування, що дозволяє виконувати Java-додатки на будь-якому пристрої, для якого існує відповідна віртуальна машина. Іншою важливою особливістю технології Java VM є гнучка система безпеки, в рамках якої виконання програми повністю контролюється віртуальною машиною. Будь-які операції, які перевищують встановлені

повноваження програми (наприклад, спроба несанкціонованого доступу до даних або з'єднання з іншим комп'ютером), викликають негайне переривання.

Для установки Java-машини необхідна наявність безкоштовно поставляється пакету JRE з поставки Software Development Kit (SDK) [23] – комплекту розробки ПЗ.

Для розробки програмного забезпечення була вибрана технологія Java SE 8, а для побудови графічного інтерфейсу користувача технологія JavaFX.

Так само проект містить спільні бібліотеки jNetPcap і Java-ml. якості середовище розробки була обрана інтегроване середовище розробки Intellij IDEA. jNetPcap є Java-обертку нативной бібліотеки під різні операційні системи (також Windows, Linux, OS X) libcap. Дану бібліотеку також використовують такі програми як Wireshark і tcpdump.

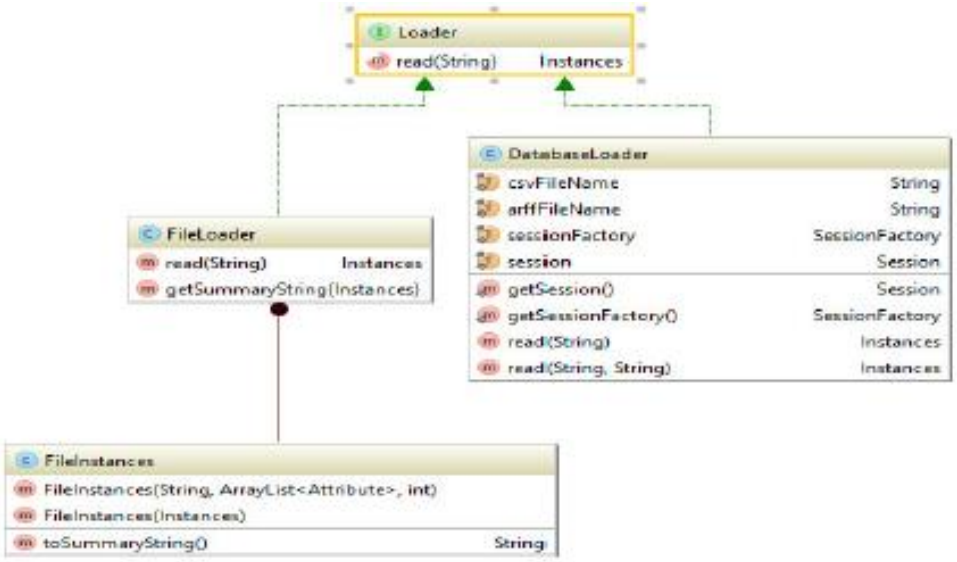


Рисунок 3.22 – Діаграма модуля роботи з файлами і базою даних

Інтерфейс Loader визначає основну функцію read (String). яка буде використовуватися всіма класами, що реалізують даний інтерфейс – діаграма класів представлена на рис. 3.22.

Клас MyFlowHandler рис.3.23 призначений для обробки надходять пакетів, в форматі ".PCAP". виділення потоків з них і передача виділеного потоку в об'єкт класу MyFlowList рис.3.24 який зробить виділення ознак класифікації і сформує

об'єкт типу Instances який можна подати на вхід класифікатора classifier, який є полем даного класу.

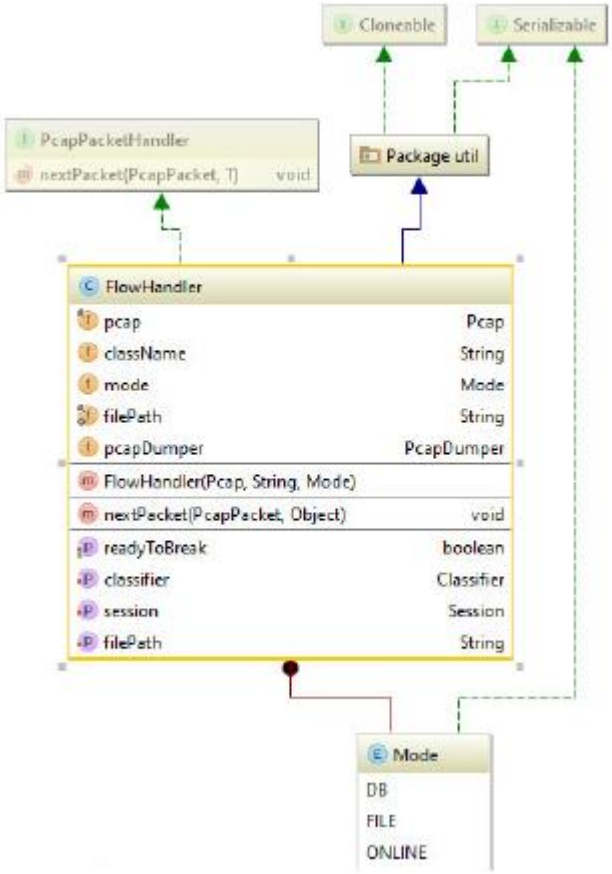


Рисунок 3.23 –Діаграма класу обробки пакетів і виділення потоків MyFlow Handler

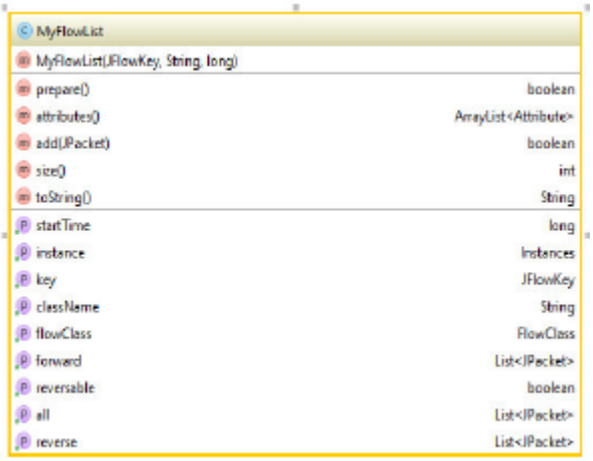


Рисунок 3.24 – Діаграма класу виділення ознак MyFlowList

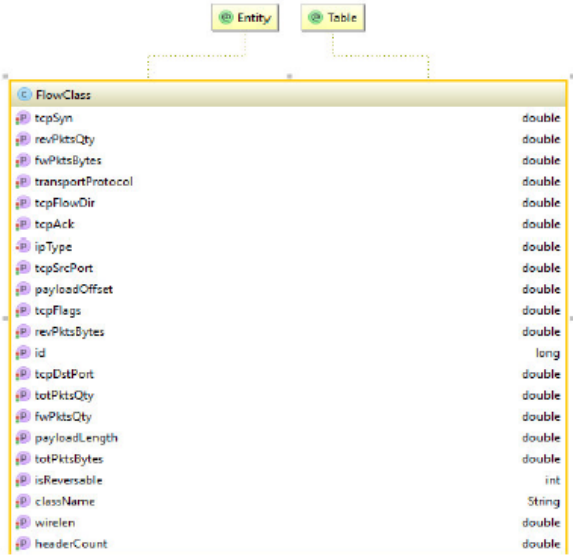


Рисунок 3.25 - Діаграма класу FlowClass, призначеного для роботи з базою даних

FlowClass представляє собою тип даних сутності бази даних. Внаслідок обробки пакетів класами My Flow Handler, MyFlowList і передачі параметра режиму роботи MyFlowHandler.Mode.DB може бути сформовани даний об'єкт, що дозволяє зберігати вектор ознак потоку в розроблену базу даних. Всі поля класу є сформованими значеннями ознак, або атрибутів класифікації. На основі даного класі була створена база даних для зберігання вектора ознак. Код для створення бази даних flowclass представлено в додатку А.

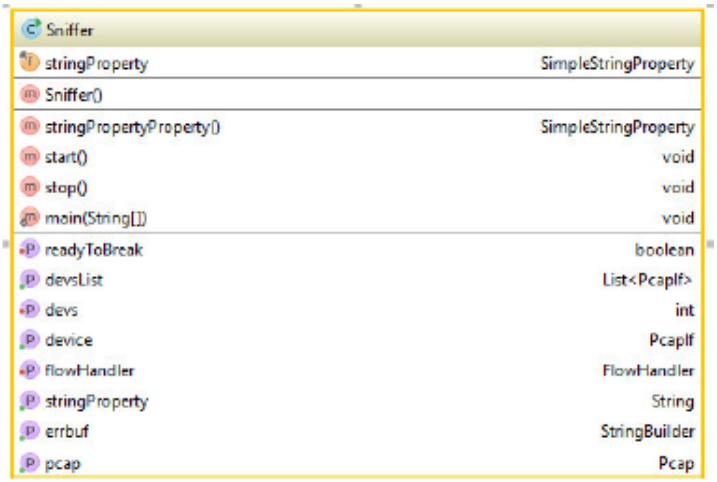


Рисунок 3.26 - Діаграма класу Sniffer для захоплення трафіку в реальному часі

Клас Sniffer дозволяє захоплювати пакети з мережевої картки. Перелік мережевих карток, доступних для захоплення можна отримати з допомогою поля devsList. Потім слід вибрати бажане пристрій devs. передати бажаний обробник пакетів типу FlowHandler і почати захоплення за допомогою функції start (). Захоплені пакети в форматі Pcap. будуть передані оброблювачу так. Як якби читання вироблялося з файлу. Для закінчення захоплення викликається функція stop (). змінює значення булевої змінної readyToBreak на true, викликає переривання циклу.

Графічний інтерфейс користувача призначений для забезпечення взаємодії користувача, або оператора програмного забезпечення з системою класифікації мережевого трафіку.

Графічний інтерфейс являє собою вкладки, на кожній з яких можлива взаємодія з одним модулем системи рис. 3.27 на нижній половині кожної форми знаходиться текстова область, в якій міститься різна інформація - стан системи, запущений модуль і т.д. нижче розташований індикатор виконання поточного завдання (Progress Bar), необхідний для візуального відображення стану виконання поставленого завдання.

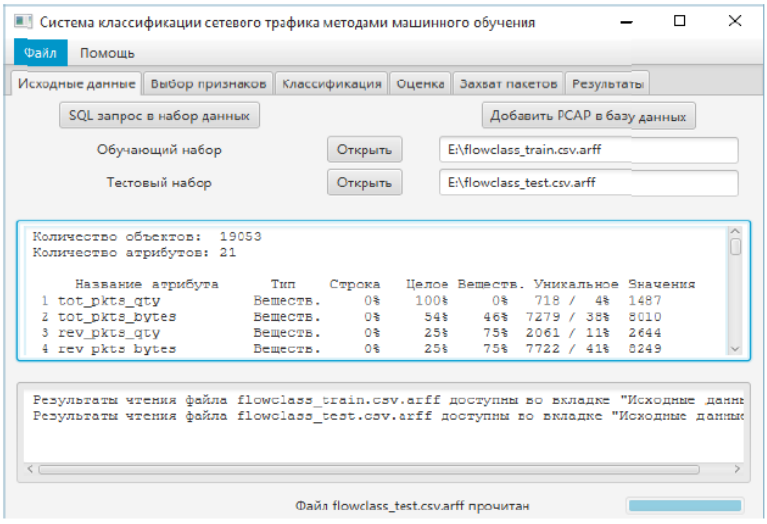


Рисунок 3.27 - Робота модуля отримання вихідних даних і формування набору даних

ВИСНОВКИ

З наведеного огляду можна зробити ряд висновків, а саме констатувати як кількісний (у зв'язку з підйомами обсягів трафіку і ширини каналів зв'язку), так і якісне зростання (в зв'язку з новими прикладними завданнями) потреб в коштах аналізу трафіку. При цьому, незважаючи на величезну різноманіття конкретних рішень, що реалізують різні види аналізу, в основі більшості цих рішень лежить приблизно однакова схема, що добре видно на прикладі впровадження концепції «DPI як сервіс».

У першому розділі аналізуються особливості класифікації IP-трафіку. Як показує аналіз, статистичні методи є оптимальним вибором для класифікації трафіку комп'ютерних мереж в реальному часі, забезпечуючи прийнятне споживання ресурсів при заданої високої достовірності.

Бажаний метод класифікації повинен бути «невидимий» для мережі і її користувачів. Це означає, що вона не повинна впливати на продуктивність мережі і служб, що надаються користувачам. З цієї точки зору, статистична класифікація має безсумнівні переваги, оскільки не уповільнює роботу мережі, але в той же час дає відносно точні результати

Далі виконано дослідження задач раціонального формування вихідних даних і аналіз програмного забезпечення, що виникають при розробці системи класифікації трафіку, використовуючи відомі алгоритми машинного навчання. Була досліджена експериментальна база даних для додатків трафіку WEB (http, https); mail (smtp, imap), Ftp (Ftp-data, Ftp-commands); SSH, Skype, P2P, об'єднаних у великі групи. Навчальний, загальний тестовий і всі групові тестові дампи є по суті незалежними один від одного наборами трафіку, що дозволяє проводити незалежну оцінку якості роботи класифікатора. Була проаналізована архітектура і розглянуто програмне забезпечення для формування вихідних даних в задачах класифікації різних додатків.

Як чисельних критеріїв оцінки ефективності якості класифікаторів в роботі використовувалися такі метрики, як Precision (Точність), Recall (Повнота), F-

Measure (F-міра) і AUC (Area Under Curve) - площа під кривою ROC (Receiver Operating Characteristic- робоча характеристика приймача).

Наведені експериментальні результати вимірювання трафіку в центрі обробки даних за допомогою програми tcpdump. Виконано порівняння показників ефективності алгоритмів класифікації на етапі навчання, які проводились при різних розмірах навчаючих вибірках маючи різний об'єм. Порівняльний аналіз методів показав, що кожен з них має свої переваги та недоліки і ефективно при відповідній ситуації. Показано, що число атрибутів для класифікації не так важливо, як вибір раціонального алгоритму класифікації. Проведено аналіз експериментальних досліджень, який показує що результати класифікації є найбільш точними, якщо класифікатор навчався в тій же мережі в якій виконується процедура класифікації.

У практичній частині виконано дослідження коли в класифікації трафіка є трафік який належить до класу що не брав участь в навчанні алгоритму. Підвищити ефективність класифікації в умовах можливого появи невідомого фонового трафіку можна шляхом введення додаткової класу під назвою «Невідомий».

Експериментальні результати показують, що шляхом навчання бінарних класифікаторів і впровадження такого додаткового класу можливо отримати задовільні алгоритми машинного навчання, які можна застосовувати для побудови класифікаторів, здатних протистояти невідомому трафіку до деякої міри (до 60%).

На завершення, у роботі, показано приклад програмного забезпечення, що реалізує методи машинного навчання в реальному масштабі часу для класифікації інтернет-трафіку.

ПЕРЕЛІК ПОСИЛАНЬ

1. Weyrich, M., Ebert, C. Reference architectures for the internet of things // IEEE Software. — 2018. — Vol. 33, № 1. — P. 112–116.

2. Кучерявый, А. Е. Самоорганизующиеся сети/ А. Е. Кучерявый, А. В. Прокопьев, Е. А. Кучерявый. – СПб.: Любавич, 2019. – 312 с.

3. Дроздов С. «Интернет вещей» и «облако устройств» / С. Дроздов, С. Золотарев // Control Engineering. – 2017.

4. Гимранов Р. Р., Киричек Р. В., Шпаков М. Н. Технология межмашинного взаимодействия LoRa // Информационные технологии и телекоммуникации. 2018. № 2 (10). С. 62–73.

5. Кумаритова, Д. Л. Обзор и сравнительный анализ технологий LPWAN сетей/ Д. Л. Кумаритова, Р. В. Киричек // Информационные технологии и телекоммуникации. — 2018. — Т. 4. — № 4. — С. 33–48.

6. David L. Cannon. CISA Certified Information Systems Auditor Study Guide, 2nd Edition, 2018, ISBN: 978-0-470-23152-4

7. ICAP. https://tools.ietf.org/html/rfc3507, дата обращения 20.10.2020.

8.QUIC. https://tools.ietf.org/html/draft-tsvwg-quic-protocol-00, дата обращения 25.10.2020

9. П. Филимонов, М. Иванов. Современные подходы к классификации трафика физических каналов сети Интернет, Труды 18-ой Международной конференции «Распределенные компьютерные и коммуникационные сети: управление, вычисление, связь» (DCCN-2019), 19 - 22 октября 2019 г, стр. 466-474

10. F. Risso, M. Baldi, O. Morandi, A. Baldini, P. Monclus, “Lightweight, payload-based traffic classification: An experimental evaluation” in Proc. IEEE ICC, 2018, pp. 5869-5875.

11.Subhabrata Sen, Oliver Spatscheck, and Dongmei Wang, «Accurate Scalable In-Network Identification of P2P Traffic Using Application Signatures», in Proceedings of the 13th international conference on World (WWW'04), p. 512-521, New York, NY, USA, 2016.

12. Andrew W. Moore and Konstantina Papagiannaki, «Toward the Accurate Identification of Network Applications», in Proceedings of the 6th International Conference on Passive and Active Network Measurement (PAM'15), p. 41-54, Boston, Massachusetts, USA, 2015.

13. Niccolo' Cascarano, Pierluigi Rolando, Fulvio Risso, and Riccardo Sisto, «iNFAnt: NFA pattern matching on GPGPU devices», ACM SIGCOMM Computer Communication Review, vol. 40, no. 5, p. 20-26, ACM, New York, NY, USA, 2016.

14. Yong Tang, Bin Xiao, and Xicheng Lu, «Using a bioinformatics approach to generate accurate exploit-based signatures for polymorphic worms», Computers & Security, vol. 28, no. 8, p. 827–842, Elsevier, 2019.

15. S. Sen, O. Spatscheck, and D. Wang, «Accurate, scalable in network identification of P2P traffic using application signatures», in WWW2004, New York, NY, USA, May 2015.

16. Matthew Roughan, Subhabrata Sen, Oliver Spatscheck, and Nick Duffield, «Classof- Service Mapping for QoS: a Statistical Signature-Based Approach to IP Traffic Classification», in Proceedings of the 4th ACM SIGCOMM Conference on Internet Measurement (IMC'04), p. 135-148, Taormina, Sicily, Italy, 2018.

17. Andrew Moore, Denis Zuev, and Michael Crogan, «Discriminators for Use in Flow-Based Classification», Technical Report RR-05-13, Department of Computer Science, Queen Mary, University of London, 20017.

18. Denis Zuev and Andrew W. Moore, «Traffic Classification using a Statistical Approach», in Proceedings of the 6th international conference on Passive and Active Network Measurement (PAM'05), p. 321-324, Boston, MA, USA, 2015.

19. Hyunchul Kim, K. C. Claffy, Marina Fomenkov, Dhiman Barman, Michalis Faloutsos, and KiYoung Lee, «Internet Traffic Classification Demystified: Myths, Caveats, and the Best Practices», in Proceedings of the 2008 ACM CoNEXT Conference (CoNEXT'08), p. 1-12, Madrid, Spain, 2018.

20. Yeon-sup Lim, Hyun-chul Kim, Jiwoong Jeong, Chong-kwon Kim, Ted «Taekyoung» Kwon, and Yanghee Choi, «Internet Traffic Classification Demystified: on

the Sources of the Discriminative Power», in Proceedings of the 6th ACM CoNEXT Conference (CoNEXT'10), no. 9, p. 1-12, Philadelphia, USA, 2015.

21. T. T. Nguyen and G. Armitage, «A survey of techniques for internet traffic classification using machine learning». IEEE Commun. Surveys Tuts., vol. 10, no. 4, p. 56–76, Fourth Quarter 2018.

22. Шелухин О. И.. Симонян А.Г., Ванюшина А.В. Эффективность алгоритмов выделения атрибутов в задачах классификации приложений при интеллектуальном анализе трафика//Электросвязь. 2017. №11. С. 79-80.

23. Ерохин С.Д.. Ванюшина А.В. Выбор атрибутов для классификации IP-трафика методами машинного обучения // T-comm: Телекоммуникации и транспорт. 2018. Т.12. №9. С.25-29

24. Щербакова Н.Г. Анализ IP-трафика методами Data Mining. Проблема классификации // Проблемы информатики. 2017. № 4. С. 30-36.

25. Richard O. Duda, Peter E. Hart, and David G. Stork, «Pattern Classification (Second Edition)», Wiley, 2016.

Додаток А

Код для створення бази даних flowclass

```
1.  -- Table: flowclass
2.
3.  -- DROP TABLE flowclass;
4.
5.  CREATE TABLE flowclass
6.  (
7.    classname text,
8.    tot_pkts_qty double precision,
9.    tot_pkts_bytes double precision,
10.   rev_pkts_qty double precision,
11.   rev_pkts_bytes double precision,
12.   fw_pkts_qty double precision,
13.   fw_pkts_bytes double precision,
14.   flowclass_id serial NOT NULL,
15.   is  reversable double precision.
16.   transport_protocol double precision,
17.   src_port double precision,
18.   dst_port double precision,
19.   wirelen double precision,
20.   header_count double precision,
21.   tcp_syn double precision,
22.   tcp_ack double precision,
23.   flags double precision,
24.   payload_length double precision,
25.   tcp_flow_dir double precision,
26.   CONSTRAINT flowclass_id PRIMARY KEY (flowclass_id)
27.)
28.WITH (
29.   OIDS=FALSE
30.);
31.ALTER TABLE flowclass
32.   OWNER TO postgres;
```

ДЕМОНСТРАЦІЙНІ МАТЕРІАЛИ (Презентація)

МЕТА ТА ЗАВДАННЯ МАГІСТЕРСЬКОЇ РОБОТИ

Мета роботи – підвищення функціонування ефективності класифікації додатків інтернет-трафіку із застосуванням машинного навчання.

Об'єкт дослідження – процес класифікації інтернет-трафіка.

Предмет дослідження – методи та засоби класифікації інтернет-трафіка.

ЗАВДАННЯ МАГІСТЕРСЬКОЇ РОБОТИ:

1.Дослідження моделей і алгоритмів класифікації додатків інтернет-трафіку в комп'ютерних мережах;
2 Аналіз задач формування вихідних даних та програмного забезпечення, що виникають при розробці системи класифікації трафіку
3 Практичне дослідження класифікації інтернет-трафіку в умовах наявності небажаного фонового впливу

Слайд 3 ПОШИРЕННЯ ІНТЕРНЕТУ ТА ПРОГНОЗ ЗРОСТАННЯ ТРАФІКУ

Класифікація інтернет-трафіку дозволяє виявляти його тип і структуру, що критично важливо для управління такими технологіями, як мережева безпека, диференціація сервісів, управління параметрами трафіку і інші. Для ефективного управління мережею необхідна точна відповідність використовуваних мережевих додатків відповідного трафіку, а також повноцінний контроль над використовуваними додатками. Із найбільш ефективних методів класифікації трафіку особливий інтерес представляють технології машинного навчання (Machine Learning) і інтелектуального аналізу даних (Data Mining), тому актуальною задачею є дослідження ефективної класифікації трафіку на основі методу машинного навчання.

Рис.1. Глобальний ріст пристроїв і підключень

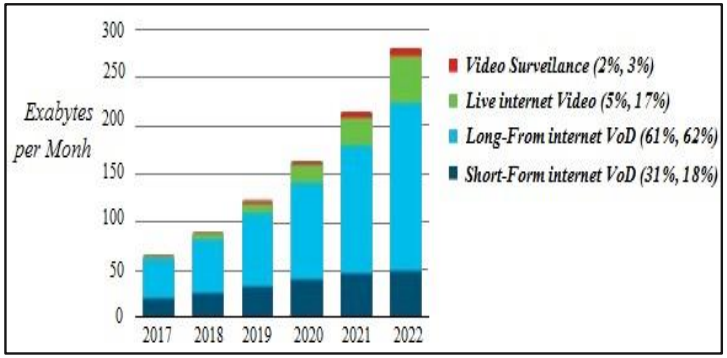


Рис.2.Глобальне інтернет-відео по підсегментам

В цілому очікуваний обсяг глобального інтернет-трафіку складе 4,8 ZB в рік (1,9+ трлн ГБ). Щомісячний IP -трафік на душу населення зросте з16 ГБ до 50 ГБ.

Слайд 4 НАПРЯМКИ РОЗВИТКУ ТЕХНОЛОГІЙ АНАЛІЗУ МЕРЕЖЕВОГО ТРАФІКУ

Можна виділити два основних напрямків розвитку аналізу мережевого трафіку:

1) Зростання «глибини» аналізу для окремого мережевого пакету

2) Повнота обліку стану потоку, до якого належить пакет, а також інших потоків, пов'язаних з даними.

Рис.1. Рівні розвитку технології аналізу мережевого трафіку по «глибині».

Слайд 5 МЕТОДИ І АЛГОРИТМИ КЛАСИФІКАЦІЇ МЕРЕЖЕВОГО ТРАФІКУ

Рис.1. Класифікації мережевого трафіку

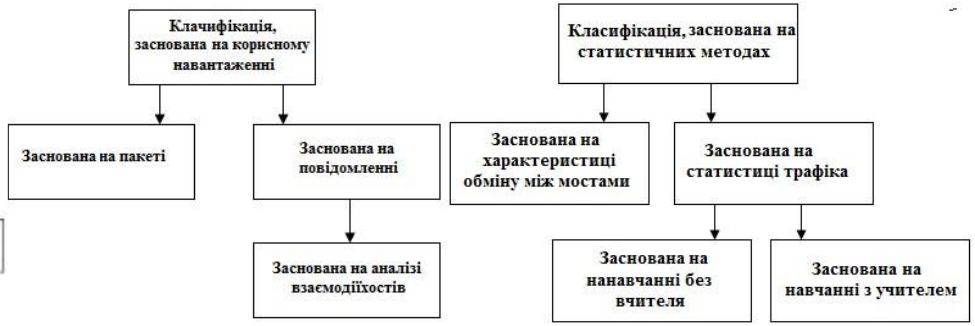


Рис.2. Класифікація, заснована на корисному навантаження

Рис.3.Класифікація, заснована на статистичних методах

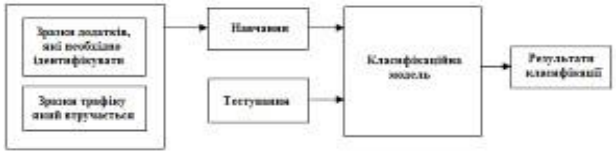


Рис.4 Навчання і тестування класифікатора

	Заснований на номерах портів	Заснований на навантаженні пакету	Заснований на статистиці потоків
Точність	Низька	Висока	Висока
Складність виділення ознак	Низька	Висока (DPI)	Залежить від набора ознак
Складність класифікації	Низька	Висока	Нижче середнього
Закодований трафік	Так	Ні	Так
Апріорна інформація	Список портів	Сигнатури	Помічені дані
Відкидання невідомого трафіка	Так	Так	Шум

Таблиця 1.1 - Порівняння сучасних підходів класифікації

Методи класифікації не повинні впливати на продуктивністьмережі і служб, що надаються користувачам

До класифікації мережевого трафіку, заснованого на статистичних методах може бути застосовано два типи алгоритмів: поведінкові і статистичні алгоритми мережевого і транспортного рівнів. У таблиці 1 наведені результати порівняння сучасних підходів класифікації. Для ефективної ідентифікації додатків необхідно вибрати невелику кількість ознак (атрибутів), які повинні бути повними і придатними для класифікації.

Слайд 6 ФОРМУВАННЯ ВИХІДНИХ ДАНИХ І АНАЛІЗ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

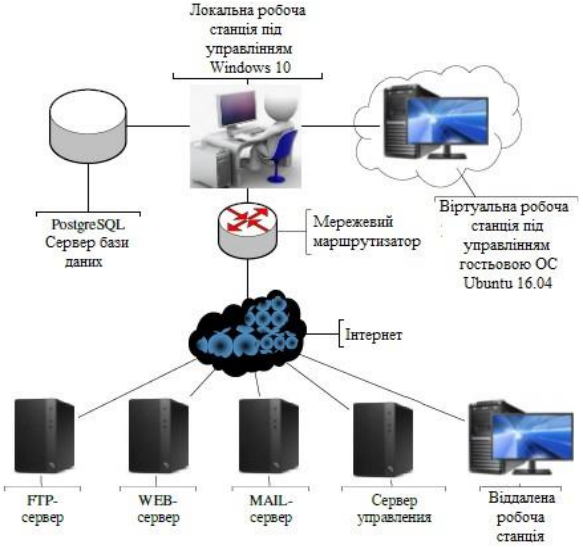


Рис.1.Топологіядосліджуваної мережі

Табл.1.Кількість потоків отриманогтрафіку по класам додатків

Кількість потоків мережевого трафіка по групам						Загальна кількість потоків
WEB, (HTTP, HTTPS)	MAIL (SMTP, IMAP)	FTP (FTP-DATA, FTP-COMMANDS)	SSH	SKYPE	P2P	
5054	3360	3470	5824	3737	6992	28437

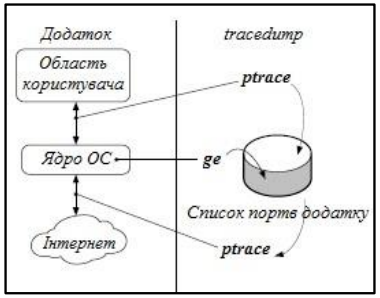


Рис.2. Архітектура програми tracedump

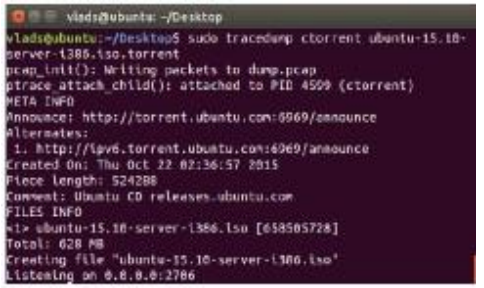


Рис.2.Процес роботи tracedump на прикладі скачування клієнта Ubuntu через P2P-мережу

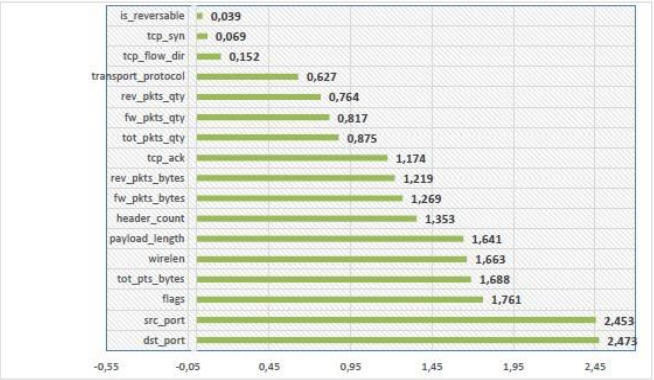


Рис.2.4.Результат застосування алгоритму виділення атрибутів InfoGain

Слайд 7 ОЦІНКА АЛГОРИТМІВ ВИДІЛЕННЯ ІНФОРМАТИВНИХ ОЗНАК ТА ЇХ ПОРІВНЯННЯ

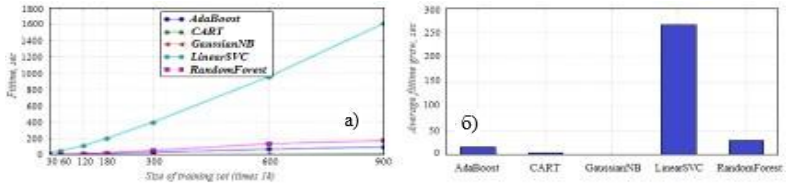


Рис.1. Оцінкачасунавчаннярізнимиалгоритмамикласифікації
а) залежність часу побудови моделі від розміру навчальної вибірки;
б) діаграма середнього зросту часу побудови моделі.

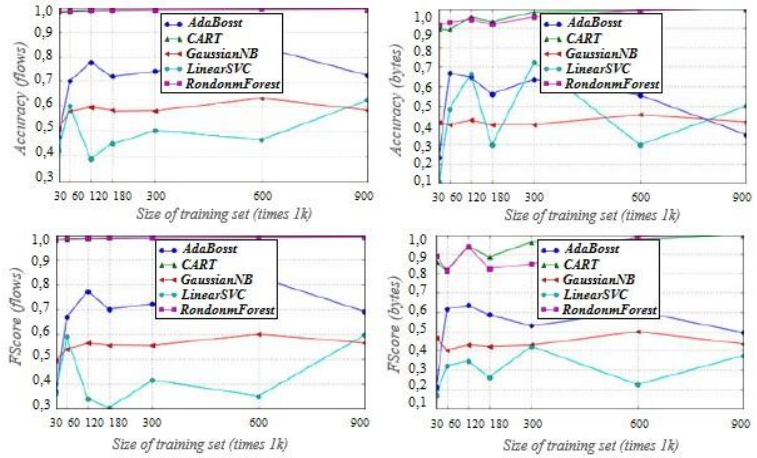


Рис.2. Залежністьвірогідності F-міриалгоритміввідрозміруначальноївибірки:
а) поточнадостовірність; б) байтоваядостовірність; в) поточнаF-міра; г) байтовая F-міра

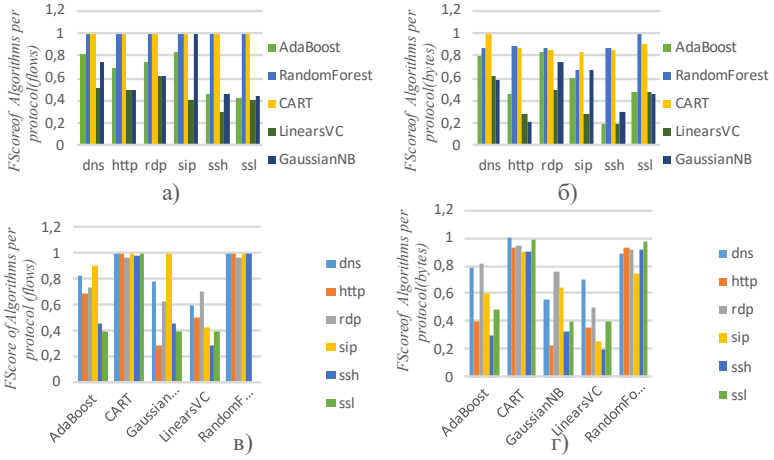


Рис.3. Достовірність різних алгоритмів в розрізі по протоколам: а) потокова достовірність кожного алгоритму по протоколам; б) байтовадостовірність кожного алгоритму по протоколам; в) поточна достовірністькласифікації протоколу певним алгоритмом; г) байтова достовірністькласифікації протоколу певним алгоритмом.

Слайд 8 ЕКСПЕРИМЕНТАЛЬНІ РЕЗУЛЬТАТИ КЛАСИФІКАЦІЇ НА ЕТАПІ НАВЧАННЯ

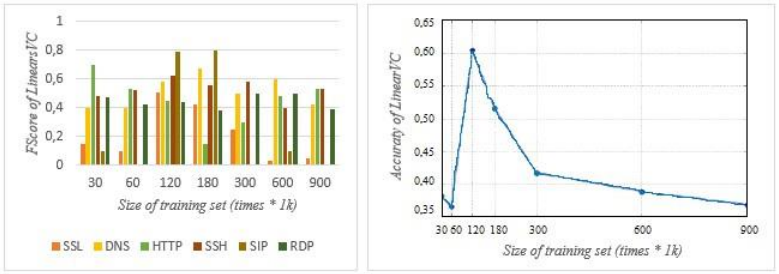


Рис.1.Залежність F-міри та точності алгоритму SVM від розміру навчальної вибірки

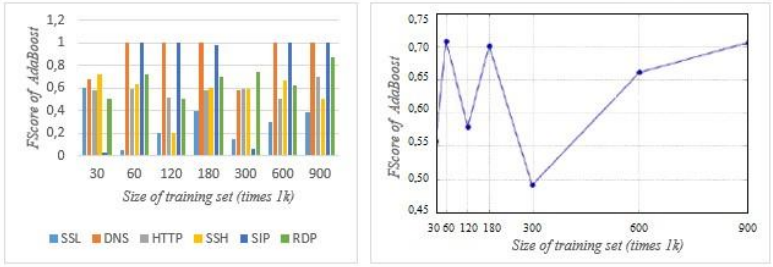


Рис.2.Залежність F-міри та точності алгоритму AdaBoost від розміру навчальної вибірки а) F-міра; б) достовірність.

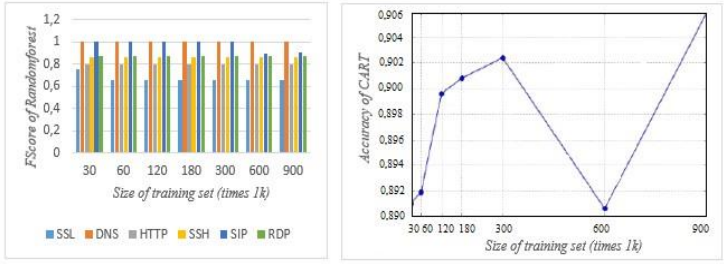


Рис.3.Залежність F-міри та точності алгоритму CART від розміру навчальної вибірки а) F-міра; б) достовірність.

Слайд 9 ЕФЕКТИВНІСТЬ ЗАСТОСУВАННЯ АЛГОРИТМУ RF В ЗАДАЧАХ КЛАСИФІКАЦІЇ ДОДАТКІВ

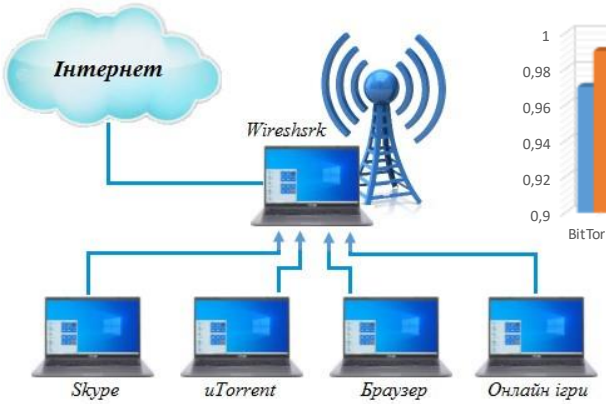


Рис.1. Схема використовуваної мережі

Табл.1. Склад отриманої вибірки даних

Передбач. реальності	SSL	HTTP	DNS	BitTorrent	Steam	Skype
SSL	295	0	0	0	0	0
HTTP	0	267	0	4	1	0
DNS	0	0	266	1	0	0
BitTorrent	1	0	0	230	0	1
Steam	0	3	0	0	201	0
Skype	6	0	0	1	0	155

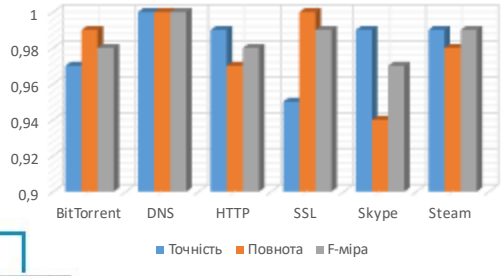


Рис.2. Точність, повнота, F-міра для тестової вибірки

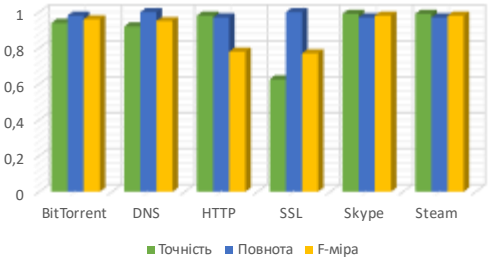


Рис.3. Точність, повнота, Fміра при наявності фонового трафіку

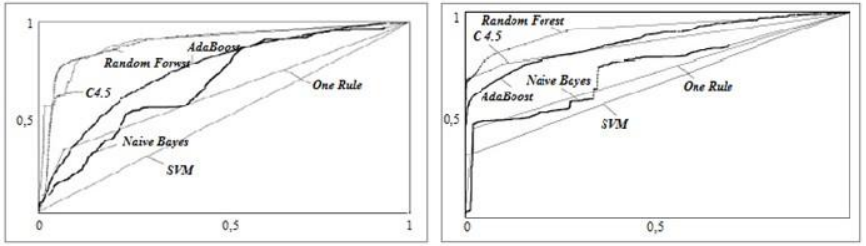


Рис.4. ROCкриві алгоритмів за класом FB: а) при відсутності фону; б) при наявності фону

Слайд 10 АНАЛІЗ АЛГОРИТМІВ КЛАСТЕРИЗАЦІЇТА ЇХ ПОРІВНЯННЯ

Рис.1. Система кластеризації мережевого трафіку

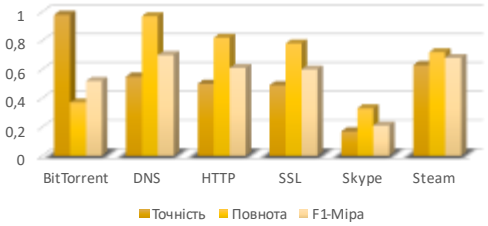


Рис.2. Точність, повнота, F-міра алгоритму DBSCAN

Табл.1. Значення метрик для алгоритму k-Means

	Точність	Повнота	F-міра
BitTorrent	0,97	0,359	0,524
DNS	0,565	0,955	0,71
HTTP	0,493	0,832	0,619
SSL	0,492	0,784	0,604
Skype	0,161	0,348	0,22
Steam	0,641	0,724	0,68

Табл.2. Значення метрик для алгоритму DBSCAN

	Точність	Повнота	F1-міра
BitTorrent	0,747	0,75	0,748
DNS	0,873	0,833	0,855
HTTP	1	0,396	0,567
SSL	1	0,27	0,429
Skype	0,25	0,29	0,27
Steam	1	0,31	0,47

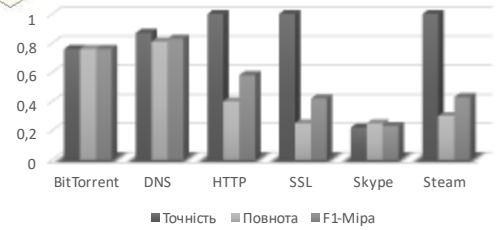


Рис.3.Точність, повнота, F-міра алгоритму DBSCAN

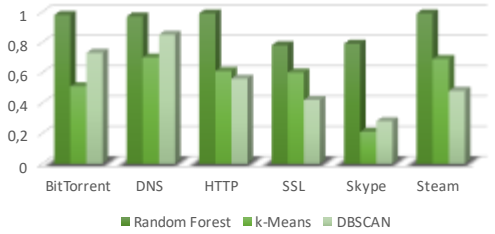


Рис.4. Значення F-міри заходи для досліджуваних алгоритмів

Слайд 11 ЗАСТОСУВАННЯ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ ДЛЯ ВИРІШЕННЯ ЗАДАЧІ КЛАСИФІКАЦІЇ ІНТЕРНЕТ-ТРАФІКУ МЕТОДОМ МН

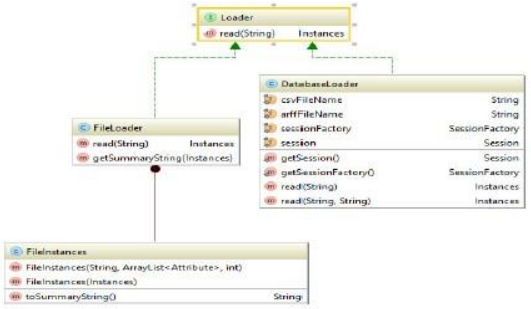


Рис.1. Діаграма модуля роботи з файлами і базою даних

Рис.1 Діаграма класу обробки пакетів і виділення потоків MyFlow Handler

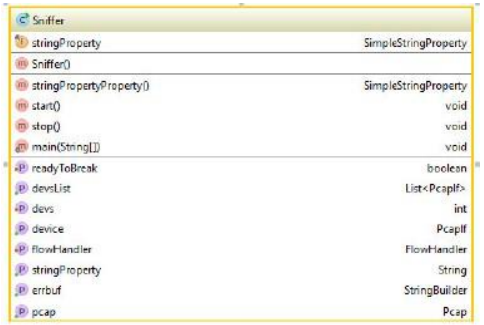


Рис.5. Діаграма класу Sniffer для захоплення трафіку в реальному часі

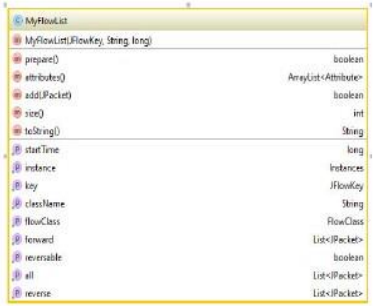


Рис.3. Діаграма класу виділення ознак MyFlowList

Рис.4. Діаграма класу FlowClass призначеного для роботи з базою даних

Слайд 12 ВИСНОВКИ ТА АПРОБАЦІЯ РЕЗУЛЬТАТІВ МАГІСТЕРСЬКОЇ РОБОТИ

Отже, виконуючи поставлені завдання, в першу чергу проведено аналіз поширення інтернету та прогноз зростання трафіку, результат якого показує стрімке зростання. Проаналізовано напрямки розвитку технологій аналізу мережевого трафіку. Запропонована архітектура і реалізовано спеціальне програмне забезпечення для формування вихідних даних в задачах класифікації різних додатків. Показано, що визначальним для класифікації трафіку є вибір не числа атрибутів, а алгоритму класифікації.

Результати тестування показали, що алгоритми МН з використанням «дерев рішень» RF і C4.5 найбільш доцільні для класифікації при великій кількості класів.

Класифікація з використанням алгоритмів AdaBoost і Bagging дає в більшості випадків поліпшення результатів при об'єднанні безлічі класифікаторів в групу і прийняття рішення на основі «голосування».

Наявність фонового трафіку, що належить до трафіку який не брав участі у навчанні класів, помітно погіршує точність класифікації.

Показано, що підвищення ефективності класифікації в умовах невідомого фонового трафіку досягається шляхом введення додаткової класу під умовною назвою «невідомий».

Аналіз показав, що при кластеризації мережевого трафіку доцільно використовувати додаткову інформацію про приналежність точок даних до одного класу.

Апробація результатів магістерської роботи. Основні положення і результати магістерської роботи доповідались і обговорювались на двох науково-практичних конференціях. За матеріалами роботи опублікована одна наукова стаття.