Міністерство освіти і науки України
Харківський національний університет радіоелектроніки

Факультет інформаційно-аналітичних технологій та менеджменту
(повна назва)

Кафедра прикладної математики
(повна назва)

КВАЛІФІКАЦІЙНА РОБОТА
Пояснювальна записка

рівень вищої освіти другий (магістерський)

Застосування нейронних мереж для виявлення інформаційних кібератак
(тема)

Виконав:
студент _2_ курсу, групи _ПМм-22-1
Макляков В. Д.
(прізвище, ініціали)
Спеціальність
113 Прикладна математика
(код і повна назва спеціальності)
Тип програми _освітньо-професійна
(освітньо-професійна або освітньо-наукова)
Освітня програма
Прикладна математика
(повна назва освітньої програми)
Керівник проф. Кіріченко Л. О.
(посада, прізвище, ініціали)

Допускається до захисту

Зав. кафедри ПМ _____ Сидоров М.В.
(підпис) (прізвище, ініціали)

2024 р.

Харківський національний університет радіоелектроніки

Факультет _інформаційно-аналітичних технологій та менеджменту
Кафедра _прикладної математики
Рівень вищої освіти _другий (магістерський)
Спеціальність _113 Прикладна математика
(код і повна назва)
Тип програми _освітньо-професійна
(освітньо-професійна або освітньо-наукова)
Освітня програма _Прикладна математика
(повна назва)

ЗАТВЕРДЖУЮ:

Зав. кафедри ПМ _____
(підпис)
“_06_” _листопада_ 2023 р.

ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ РОБОТУ

студентові _Маклякову Володимиру Дмитровичу
(прізвище, ім’я, по батькові)

1. Тема роботи _Застосування нейронних мереж для виявлення інформаційних кібератак
затверджена наказом по університету від 2 листопада 2023 р. № 1276 Ст

2. Термін подання студентом роботи до екзаменаційної комісії 10 січня 2024 р.

3. Вихідні дані до роботи модельні та реальні реалізації мережного трафику

4. Перелік питань, що потрібно опрацювати в роботі
1. Аналіз предметної області
2. Вибір і обґрунтування методу розв’язання
3. Програмна реалізація
4. Результати обчислювального експерименту
5. Аналіз можливих застосувань

5. Перелік графічного матеріалу із зазначенням креслеників, схем, плакатів, комп'ютерних ілюстрацій

1. Актуальність теми роботи
2. Постановка задачі
3. Аналіз предметної області
4. Метод чисельного аналізу
5. Результати обчислювального експерименту

КАЛЕНДАРНИЙ ПЛАН

№	Назва етапів роботи	Терміни виконання етапів роботи	Примітка
1	Підбір та вивчення технічної літератури за темою роботи	6 – 12 листопада 2023 р.	виконано
2	Вибір та обґрунтування методу	13 – 26 листопада 2023 р.	виконано
3	Розробка алгоритму і програми	27 листопада – 10 грудня 2023 р.	виконано
4	Проведення аналітичних досліджень та розрахунків	11 грудня – 24 грудня 2023 р.	виконано
5	Робота над текстом пояснювальної записки	25 грудня 2023 р. – 9 січня 2024 р.	виконано
6	Представлення роботи на рецензію в ЕК	10 січня 2024 р.	виконано

Дата видачі завдання _6_ листопада 2023 р.

Студент _____
(підпис)

Керівник роботи _____ проф. Кіріченко Л. О.
(підпис) (посада, прізвище, ініціали)

РЕФЕРАТ

Пояснювальна записка: 86 с., 6 табл., 29 рис., 1 дод., 19 джерел.

ПОКАЗНИК ХЕРСТА, САМОПОДІБНІЙ ПРОЦЕС, БРОУНІВСБКІЙ РУХ, ГАУСІВСКІЙ ШУМ, DDOS АТАКА, ШТУЧНІ НЕЙРОННІ МЕРЕЖІ.

Об’єкт дослідження – процес детектування DDOS-атак.

Мета роботи – метою кваліфікаційної роботи є розробка моделі та методу виявлення кібератак за реалізаціями мережного трафіку.

Методи дослідження – метод штучних нейронних мереж та метод мультифрактального аналізу.

У кваліфікаційній роботі розглянуто завдання розпізнавання DDoS атак у мережевому трафіку з використанням штучних нейронних мереж для розв’язання задачі класифікації. Побудована програмна реалізація штучної нейронної мережі. Побудована модель і її програмна реалізація мережевого трафіка. Штучна нейронна мережа була обучена на даних, які було згенеровано на базі цієї моделі. Досліджена точність класифікації в залежності від параметрів моделі та типу DDoS атаки.

ABSTRACT

Introductory note: 86 pages, 6 tables, 29 figures, 1 appendix, 19 sources.

HURST EXPONENT, SELF-SIMILAR PROCESS, BROWNIAN MOTION, GAUSSIAN NOISE, DDOS ATTACK, ARTIFICIAL NEURAL NETWORKS.

Object of research – the process of detecting DDOS attacks.

Purpose of work – the goal of the qualification work is to develop a model and methodology for detecting cyber attacks through network traffic implementations.

Methods of research – artificial neural network method and multifractal analysis method.

In the master's work the problem of recognition of DDOS attacks in network traffic using artificial neural networks for solving the classification problem is considered. The program realization of the artificial neural network is constructed. The model and its program realization of network traffic are constructed. An artificial neural network was trained on data that was generated based on this model. The classification accuracy is studied depending on the parameters of the model and the type of DDOS attack.

ЗМІСТ

ПЕРЕЛІК СКОРОЧЕНЬ, УМОВНИХ ПОЗНАК, ОДИНИЦЬ І ТЕРМІНІВ

DOS – Denial of Service;

DDOS – Distributed Denial of Service;

TCP – Transmission Control Protocol;

UDP – User Datagram Protocol;

ШНМ – штучна нейронна мережа;

ФБР – фрактальний броунівський рух;

ФГШ – фрактальний гаусівський шум.

ВСТУП

Актуальність теми. Актуальність роботи зумовлена ростом кількості кібератак у мережі Internet. Дослідження аналізу трафіку в комп'ютерній мережі може виявити атаку зловмисника ще на етапі її підготовки.

Стрімкий розвиток обчислювальної техніки привів до того, що комп’ютерна мережа стала використовуватися як повно-функціональний розподілений обчислювальний пристрій для обробки і передачі даних.

Зараз проводиться велика кількість досліджень, що мають на меті виявлення характерних властивостей і параметрів технології передачі даних, а також пошук оптимально гарантованих способів виявлення аномалій в роботі комп'ютерної мережі, які можуть будь-яким образом вплинути на процеси що протікають в ній.

При проектуванні, запуску і експлуатації інформаційних комунікаційних мереж однією з основних проблем є завдання забезпечення якості обслуговування (заданих рівнів затримок, втрат і ін.) при обробці потоку даних – трафіку, що є наслідком інформаційного обміну між системами.

Інтернет стрімко перетворюється в ринок послуг з дуже великим оборотом грошових коштів. Збій або недоступність інформаційного сервера тягне величезні фінансові втрати. Останнім часом стала особливо актуальною проблема DoS / DDoS-атак. Для забезпечення інформаційної безпеки в організаціях необхідно вміти відображати або запобігати DoS / DDoS-атаки, але на сьогоднішній день немає ефективного вирішення.

На даний момент існує безліч приватних методів боротьби з DoS-атаками, але окремо один від одного їх застосування неефективно, тому що при виникненні DoS-атаки незрозуміло, до якого типу вона відноситься. Універсальних рецептів протидії DoS / DDoS атакам немає. Але вже зараз можна зробити ряд заходів щодо зниження ймовірності реалізації таких атак, грунтуючись на їх класифікації. Саме вона допомагає швидко виявити початок

DoS-атаки і використовувати відомі методи боротьби для її запобігання. Одним із методів виявлення атаки є дослідження фрактальних властивостей трафіку.

Однією з основних проблем в захисті від DDoS є недосконаліть законодавства в області розслідування комп'ютерних інцидентів і покарання винних, а також простота і доступність програм для організації бот-мереж і створення DdoS-атак. Проблема все ще не втратила своєї актуальності, незважаючи на тривалий період її вивчення. Вирішення цих завдань має не тільки теоретичне, а й практичне значення.

На тлі стрімкого розвитку і модифікації шкідливого програмного забезпечення (ПЗ) и підвищення рівня хакерства, антивірусне ПЗ не завжди може повною мірою захистити користувачів комп'ютерних мереж від дій зловників. Все частіше поряд зі статистичними методами, які аналізують відповідність конкретної дії в мережі певним шаблоном і записів журналів (брандмауери), використовується аналіз поведінки трафіку під час роботи мережі. Аналізуючи певні параметри, можна виявити в будь-який момент часу в поведінці трафіку що з'являються аномальні зміни (сплески).

Мета і завдання кваліфікаційної роботи. Метою кваліфікаційної роботи є розробка моделі та методу виявлення кібератак за реалізаціями мережного трафіку. Для досягнення поставленої мети необхідно виконати наступні завдання:

– вивчення характеристик і властивостей реалізацій мультифрактального трафіку;

– вивчення характеристик і властивостей реалізацій трафіку під впливом різних типів DDOS-атак;

– побудова моделі мультифрактального мережного трафіку та реалізацій трафіку під впливом DDOS-атак.

Об’єктом дослідження є процес детектування DDOS-атак.

Предметом дослідження є модельні та реальні реалізації мережного трафіку.

Методи дослідження. У роботі використовуються метод штучних нейронних мереж та метод мультифрактального аналізу.

Публікації. Результати, отримані у роботі, було представлено на 27-му Міжнародному молодіжному форумі «Радіоелектроніка та молодь у XXI столітті» (м. Харків, 10-12 травня 2023 р.) [1].

1 АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ ТА ПОСТАНОВКА ЗАДАЧ ДОСЛІДЖЕННЯ

1.1 Види DDOS-атак та їх характеристики

Кібератака (хакерська атака) – спроба реалізації загрози. Тобто, це дії кібер-зловмисників (хакерів) або шкідливих програм, які спрямовані на захоплення інформаційних даних віддаленого комп'ютера, отримання повного контролю над ресурсами комп'ютера або на виведення системи з ладу.

Під атакою (англ. attack, англ. intrusion) на інформаційну систему розуміють дії (процеси) або послідовність зв'язаних між собою дій порушника, які приводять до реалізації загроз інформаційним ресурсам, шляхом використання уразливостей цієї інформаційної системи.

З погляду інформаційної безпеки зловмисник може реалізувати такі погрози:

– модифікація інформації при передачі її по мережі або в процесі обробки й зберігання на комп'ютері користувача;

– порушення конфіденційності інформації;

– знищення інформації;

– порушення працездатності комп'ютера ("denial of service");

– несанкціоноване використання ресурсів комп'ютера;

– запис довільних даних на локальний комп'ютер;

− роздратування користувача.

Однією із найнебезпечніших погроз є віртуальні DDOS-атаки. Непрямі збитки від однієї подібної атаки можуть становити сотні тисяч доларів у день.

DOS-атака (Denial of Service – "відмова в обслуговуванні") і DDOS-атака (Distributed Denial of Service – "розподілена відмова обслуговування") – це різновиди атак зловмисника на комп'ютерні системи. Їхньою метою є створення таких умов, при яких легітимні (правомірні) користувачі системи не можуть одержати доступ до надаваних системою ресурсів, або цей доступ ускладнений.

Наприклад, вони особливо небезпечні для компаній, які пов'язані з телекомунікаційним ринком і просто не можуть собі дозволити стати жертвами зловмисників. Компанії несуть цілком відчутні фінансові збитки, як прямі, так і непрямі. Заблокований сайт для організації означає втрачену вигоду, а витрати на відбиття атаки або звертання до сторонніх сервіс-провайдерів досить відчутні для бюджету. Захист від DDOS-атак особливо важливий для інтернет-магазинів, ресурсів новин та інших компаній, діяльність яких передбачає постійні звернення користувача до ресурсу. У випадку з віртуальною війною DDOS-атаки наносять більш значні іміджеві втрати, ніж матеріальні. Зміст атаки на офіційні сайти владних органів очевидні – продемонструвати слабкість структур безпеки. Якщо такі атаки будуть вестися на DNS-сервери і поштові сервери, то у держустанов із засобів зв'язку залишаться лише телефон і факс.

Атака на відмову в обслуговуванні, розподілена атака на відмову в обслуговуванні (англ. DoS attack, DDoS attack, (Distributed) Denial-of-service attack) – напад на комп'ютерну систему з наміром зробити комп'ютерні ресурси недоступними користувачам, для яких комп'ютерна система була призначена.

Одним із найпоширеніших методів нападу є насичення атакованого комп'ютера або мережевого устаткування великою кількістю зовнішніх запитів (часто безглуздих або неправильно сформульованих) таким чином атаковане устаткування не може відповісти користувачам, або відповідає настільки повільно, що стає фактично недоступним.

Взагалі відмова сервісу здійснюється:

– примусом атакованого устаткування до зупинки роботи програмного забезпечення/устаткування або до витрат наявних ресурсів, внаслідок чого устаткування не може продовжувати роботу;

– заняттям комунікаційних каналів між користувачами і атакованим устаткуванням, внаслідок чого якість сполучення перестає відповідати вимогам.

Якщо атака відбувається одночасно з великої кількості IP-адрес, то її називають розподіленою (англ. Distributed Denial-of-Service – DDoS).

Небезпека більшості DDoS-атак – в їх абсолютній прозорості і «нормальності». Адже якщо помилка в ПЗ завжди може бути виправлена, то повна витрата ресурсів — явище майже буденне. З ними стикаються багато адміністраторів, коли ресурсів машини (ширини каналу) стає недостатньо, або web-сайт піддається слешдот-ефекту. І, якщо різати трафік і ресурси для всіх підряд, то можна врятуватися від DDoS, у той же час, втративши велику частину клієнтів.

Виходу з цієї ситуації фактично немає, проте наслідки DDoS-атак і їх ефективність можна істотно понизити за рахунок правильного налаштування маршрутизатора, брандмауера і постійного аналізу аномалій в мережевому трафіку.

З погляду інформаційної безпеки зловмисник може реалізувати такі погрози:

– модифікація інформації при передачі її по мережі або в процесі обробки й зберігання на комп'ютері користувача;

– порушення конфіденційності інформації;

– знищення інформації;

– порушення працездатності комп'ютера ("denial of service");

– несанкціоноване використання ресурсів комп'ютера;

– запис довільних даних на локальний комп'ютер;

– роздратування користувача.

Однією із найнебезпечніших погроз є віртуальні DDOS-атаки. Непрямі збитки від однієї подібної атаки можуть становити сотні тисяч доларів у день.

DDOS-атаки особливо небезпечні для компаній, які пов'язані з телекомунікаційним ринком і просто не можуть собі дозволити стати жертвами зловмисників. Компанії несуть цілком відчутні фінансові збитки, як прямі, так і непрямі. Заблокований сайт для організації означає втрачену вигоду, а витрати на відбиття атаки або звертання до сторонніх сервіс-провайдерів досить відчутні для бюджету. Захист від DDOS-атак особливо важливий для інтернет-магазинів, ресурсів новин та інших компаній, діяльність яких передбачає постійні звернення користувача до ресурсу.

У випадку ж з віртуальною війною DDOS-атаки наносять більш значні іміджеві втрати, ніж матеріальні.

Зміст атаки на офіційні сайти владних органів очевидні – продемонструвати слабкість структур безпеки. Якщо такі атаки будуть вестися на DNS-сервери і поштові сервери, то у держустанов із засобів зв'язку залишаться лише телефон і факс.

Якщо на початку DDOS-атаками "бавилися" більше заради слави або з почуття помсти, то зараз цей бізнес приносить своїм організаторам цілком відчутні прибутки. Наприклад, DDOS-атаки використовуються для шантажу, у цих випадках злочинна група обіцяє припинити потік flood в обмін на деяку кількість готівки. Проте, відомо, що більшість онлайн-казіно й офшорних компаній платять хакерам превентивно.

DDOS-атака може використовуватися і як прикриття для запуску інших шкідливих програм, за допомогою яких зловмисник може викрасти конфіденційні дані, які потім продаються конкурентам.

Схематично DDOS-атака виглядає приблизно так: на обраний в якості жертви сервер навалюється величезна кількість помилкових запитів з безлічі комп'ютерів з різних кінців світу. У результаті сервер витрачає всі свої ресурси на обслуговування цих запитів і стає практично недоступним для звичайних користувачів. Цинічність ситуації полягає в тому, що користувачі комп'ютерів, з яких направляються помилкові запити, можуть навіть не підозрювати про те, що їхня машина використовується хакерами. Програми, встановлені зловмисниками на цих комп'ютерах, прийнято називати "зомбі". Відомі безліч шляхів "зомбіювання" комп'ютерів – від проникнення в незахищені мережі до використання програм-троянів. Цей підготовчий етап є для зловмисника найбільш трудомістким.

Найчастіше зловмисники при проведенні DDOS-атак використовують трьохрівневу архітектуру, що називають "кластер DDOS". Така ієрархічна структура містить:

– консоль керування (їх може бути декілька), тобто той комп'ютер, з якого зловмисник подає сигнал про початок атаки;

– головні комп'ютери. Це ті машини, які одержують сигнал про атаку з консолі керування й передають його агентам-зомбі. На одну керуючу консоль залежно від масштабності атаки може доводитися до декількох сотень головних комп'ютерів;

– агенти – безпосередньо самі "зомбі"-комп'ютери, що своїми запитами атакують вузол-мішень.

Простежити таку структуру у зворотньому напрямку практично неможливо. Як відомо, і комп'ютери-агенти, і головні комп'ютери є також потерпілими в даній ситуації й називаються "скомпрометованими". Така структура робить практично неможливим відстежити адресу вузла, що організував атаку.

Інша небезпека DDOS полягає в тому, що зловмисникам не потрібно мати якісь спеціальні знання й ресурси. Програми для проведення атак вільно поширюються в мережі. За роки це програмне забезпечення постійно модифікувалося й до теперішнього часу фахівці з інформаційної безпеки виділяють наступні види DDOS-атак:

– UDP flood – відправлення на адресу системи-мішені безлічі пакетів UDP (User Datagram Protocol). Цей метод використовувався на ранніх атаках і в цей час вважається найменш небезпечним. Програми, що використовують цей тип атаки легко виявляються, тому що при обміні головного контролера й агентів використовуються нешифровані протоколи TCP і UDP.

– TCP flood – відправлення на адресу мішені безлічі TCP-пакетів, що також приводить до "зв'язування" мережних ресурсів.

– TCP SYN flood – відправлення великої кількості запитів на ініціалізацію TCP-з'єднань із вузлом-мішенню, якому в результаті доводиться витрачати всі свої ресурси на те, щоб відслідковувати ці частково відкриті з'єднання.

– Smurf-атака – ping-запити ICMP (Internet Control Message Protocol) за адресою спрямованого широкомовного розсилання з використанням у пакетах цього запиту фальшивої адреси джерела, яка в результаті виявляється мішенню атаки.

– ICMP flood – атака, аналогічна Smurf, але без використання розсилання.

Природно, найнебезпечнішими є програми, що використовують одночасно кілька видів описаних атак. Вони одержали назву TFN і TFN2K і вимагають від хакера високого рівня підготовки.

Однією з останніх програм для організації DDOS-атак є Stacheldracht, що дозволяє організовувати всілякі типи атак і лавини широкомовних ping-запитів із шифруванням обміну даними між контролерами й агентами.

1.2 Методи детектування DDOS-атак

З погляду інформаційного захисту DDOS-атаки є однією із найскладніших мережевих погроз, тому вживання ефективних заходів протидії є винятково складним завданням для організацій, діяльність яких залежить від інтернету. Міри протидії DDOS-атакам можна розділити на пасивні й активні, а також на превентивні й реакційні. Розглянемо основні методи.

Запобігання. Профілактика причин, що спонукують тих або інших осіб організовувати DDOS-атаки. Дуже часто атаки є наслідками особистої образи, політичних, релігійних розбіжностей, що провокує поводження жертви.

Розосередження. Побудова розподілених і резервних систем, які не припинять обслуговувати користувачів навіть якщо деякі їхні елементи стануть недоступні через атаку.

Відхилення. Відвести безпосередню ціль атаки (доменне ім'я або IP-адресу) подалі від інших ресурсів, які часто піддаються впливу разом з безпосередньою ціллю.

Фільтрація. Фільтрація трафіку на маршрутизаторах – поки найпоширеніший метод протидії. Фільтри варто вводити можливо ближче до джерела flood. Міжмережеві екрани й спеціалізовані antiflood засоби фільтрації – найбільш ефективна міра, але й найбільш дорога. Наприклад, програмний засіб Ados є динамічним фільтром TCP-пакетів, здатним блокувати в реальному

часі доступ до вебсерверу з IP-адрес, що генерують інтенсивний потік HTTP-запитів. Знизити витрати можна, розділяючи такі системи між багатьма клієнтами (фільтрація на вимогу).

Нарощування. Якщо flood спрямований на вичерпання ресурсів, самий примітивний спосіб протидії flood – нарощування своїх ресурсів, щоб супротивник не зміг їх вичерпати.

Сучасні засоби захисту від DDOS-атак дозволяють із досить високим ступенем ефективності виявити атаку й зменшити або запобігти збитку ресурсам операторів і їхніх клієнтів. Компанія "NVisionGroup" пропонує комплексне рішення для захисту від DDOS-атак на основі технології Cisco Clean Pipes, що забезпечує оперативну реакцію на DDOS-атаки, легко масштабується, має високу надійність і швидкодію.

Технологія Cisco Clean Pipes припускає використання модулів Cisco Anomaly Detector і Cisco Guard, а також різні системи статистичного аналізу мережевого трафіку, засновані на даних, одержуваних з маршрутизаторів за протоколом Cisco Netflow. При цьому Anomaly Detector і системи статистичного аналізу трафіку виступають як системи виявлення DDOS- атак, а Cisco Guard як засіб протидії вже виявленій атаці. У загальному випадку технологія Clean Pipes припускає наявність етапу тестування (навчання), що проводиться в період відсутності DDOS-атак на ресурс, що захищається. На цьому етапі пристрої виявлення визначають і запам'ятовують, який трафік для ресурсу, що захищається, є нормальним. Ситуація, при якій поточний трафік на ресурс, що захищається, різко відрізняється від нормального, уважається DDOS-атакою. При виявленні DDOS, система виявлення повідомляє оператору та активує підсистему захисту Cisco Guard.

DDOS-атаку дуже складно виявити й запобігти, оскільки "шкідливі" пакети не відрізняються від "легітимних". Мережеві пристрої й традиційні технічні рішення для забезпечення безпеки мережевого периметру, такі як міжмережеві екрани й системи виявлення вторгнень (IDS), є важливими компонентами загальної стратегії мережевої безпеки, однак самі ці пристрої не

забезпечують повного захисту від DDOS-атак. Міжмережеві екрани дозволяють або забороняють проходження мережевого трафіку на підставі аналізу різних полів мережевих пакетів. Але DDOS-атака може бути успішно реалізована в рамках дозволених міжмережевим екраном потоків трафіка. Оскільки трафік DDOS-атаки – це звичайні мережеві пакети, кожен з яких окремо собою атаку не представляє, то система IDS не виявить таку атаку. У деяких випадках, при проведенні таких атак використовується підміна IP-адрес джерела, через що стає неможливою ідентифікація шкідливого трафіку від конкретного джерела.

Для боротьби з DDOS-атаками необхідно використовувати комбіновані рішення: на рівні сервера, на рівні сервісів сервера, на рівні мережі, на рівні провайдера, на рівні апаратури, на рівні адміністраторів сервера.

Для вирішення завдання виявлення DDOS-атаки у трафіку скористаємося методами машинного навчання, зокрема нейронними мережами. Щоб навчити нейронну мережу розпізнавати DDOS-атаки в якості входів використаємо згенеровані модельні трафіки.

При проектуванні, запуску і експлуатації інформаційних комунікаційних мереж однією з основних проблем є завдання забезпечення якості обслуговування (заданих рівнів затримок, втрат і ін.) при обробці потоку даних – трафіку, що є наслідком інформаційного обміну між системами.

1.3 Формальна та змістовна постановка задачі

В якості теоретичної бази для проектування систем розподілу інформації використовується теорія, яка є однією з гілок теорії масового обслуговування. Дана теорія добре описує процеси, що відбуваються в таких системах розподілу інформації, як телефонні мережі, побудованих за принципом комутації каналів. Найбільш поширеною моделлю потоку викликів (даних) в даній теорії є найпростіший потік (стаціонарний ординарний потік без післядії), також званий стаціонарним пуассоновским потоком.

Коли ми говоримо про мережевий трафік, то під самоподібностью мається на увазі повторюваність розподілу навантаження в часі при різних масштабах. Це означає, що якщо ми намалюємо графік залежності щільності інформаційного потоку від часу, взявши за одиницю виміру секунду, хвилину, годину і так далі, то кожен раз отримаємо практично однакові діаграми. Мовою математики це звучить наступним чином. Якщо набір значень самоподібної функції (тобто яка проявляє ознаки самокорреляції) розділити на рівні групи, а потім підсумувати значення всередині груп, то набір сум буде підкорятися тій же самій кореляційної функції, що і вихідні дані.

Спочатку цю особливість вдалося помітити в мережах Ethernet. Після того як даний феномен був доведений, безліч дослідників зайнялися проблемою самоподібності мережевого трафіку. Наприклад, відразу після винаходу WWW з'явилися публікації і на тему повторюваності трафіку в цій системі. Вдалося визначити, що можлива причина такого дивного ефекту – в особливостях розподілу файлів по серверам, їх розмірах, а також в типовій поведінці користувачів.

Була поставлена під сумнів основа основ глобальної мережі – протокол TCP. Виявилося, що з самого початку потоки даних, які не проявляють властивостей самоподібності, пройшовши обробку на вузлових серверах і активних мережевих елементах, починають подавати яскраво виражені ознаки самокорреляції. Особливо сильно цей ефект помітний в високошвидкісних мережах.

Сучасні мережі побудовані на основі принципу "усереднення". Згідно зі статистикою, безліч потоків даних з випадковими варіаціями щільності дадуть в результаті якийсь усереднений трафік. На жаль, цей ідеалістичний підхід не працює: мережі, побудовані на базі TCP / IP, схильні до прояву потужних пікових викидів. Такі своєрідні, локалізовані в часі "стовпотворіння" (congestions) викликають значні втрати пакетів, навіть коли сумарна потреба всіх потоків далека від максимально допустимих значень. Отже, автокорреляція в часі безпосередньо позначається на ефективності використання пропускної здатності мереж.

Суть проблеми полягає в методі запобігання перевантаження мережі. Надійність доставки пакетів в мережі гарантується механізмом підтверджень (ACK). Одержувач для кожного прийнятого пакета відсилає своєму "респонденту" повідомлення, що містить номер наступного очікуваного їм пакета. Протокол IP не гарантує дотримання послідовності доставки і синхронності передачі даних в обох напрямках. Отже, єдиним способом уникнути тривалих простоїв системи в очікуванні проходження пакета і підтвердження є використання так званих "вікон". Висилаюча сторона, не чекаючи нічого взамін, передає відразу кілька пакетів (їх кількість визначається величиною або розмірами вікна). Копії відісланих пакетів поміщаються в чергу повторної передачі, і для кожного з них "запускається" таймер, відлічує певний проміжок часу. Як тільки з'являється підтвердження прийому від одержувача, всі пакети, для яких воно було створене, видаляються з черги. Таким чином, до повторної відправки справа доходить тільки в разі неотримання підтвердження у вказаний проміжок часу (він обчислюється динамічно на підставі вимірів часу проходження пакетів).

Трафік, який завідомо не проявляє ознак самоподібності, пройшовши через стек протоколів TCP / IP, модулюється останнім і перетворюється на справжнісінький "мережевий фрактал". І цей фрактал, прихований від наших органів чуття в самоподібних мережевих потоках, в буквальному сенсі пожирає корисні дані.

Процеси, що володіють фрактальними властивостями, можна розділити на дві групи: самоподібні (монофрактальние) і мультифрактальні. Монофрактальні процеси є однорідними в тому сенсі, що їх скейлінгові характеристики залишаються незмінними на будь-якому діапазоні масштабів і володіють одним показником скейлінга. Мультифрактального процеси допускають розкладання на ділянки з різними локальними масштабними властивостями і характеризуються спектром скейлінгових показників.

Мультифрактальні заходи були введені Б. Мандельброт для опису розподілу турбулентної дисипації. З тих пір мультифрактальні структури були

виявлені в безлічі природних і технічних процесів і використовуються для моделювання та дослідження турбулентних процесів, телекомунікаційних потоків, фінансових ринків, медичних даних, геофізичних процесів.

Мультифрактальність – це концепція, яка, з деякими незначними змінами, може бути в рівній мірі добре застосована до функцій і заходів, детермінованих або стохастичних. В описі основних понять і властивостей мультифрактальних процесів існує кілька підходів: глобальний, при якому основні визначення і властивості виводяться через властивості моментів випадкових процесів; локальний, що описує локальні масштабні і сингулярні властивості фрактальних реалізацій і функцій і заснований на властивостях локальних експонент Гельдера; і підхід, заснований на властивостях фрактальних і мультифрактального заходів (множин). В даний час виникла ціла низка монографій, де зроблені спроби об'єднати всі підходи в єдине ціле.

Як вже було зазначено, класична пуассонівська модель трафіку, яка використовувалася при проектуванні мережевих протоколів, які розподіляють трафік в комп'ютерних мережах, не може адекватно відобразити реальну дійсність. Це пов'язано з іншою природою (передача пакетів) мережевого трафіку. Крім того, мережевий трафік має властивість самоподібності, яке не враховує модель Пуассона.

Існують різні підходи придатні для моделювання мультифрактальних процесів. Вперше в якості мультифрактальних моделей для трафіку були використані мультиплікативні каскади. Цей клас моделей є найвідомішим серед мультифрактальних процесів. Найпростішим прикладом мультифрактального процесу є біноміальний каскад, який визначається за допомогою бінарної деревовидної структури. Поєднуючи цей процес з фрактальним броунівським рухом, можна визначити новий клас фрактальних броунівських рухів в мультифрактальному часовому просторі. Отриманий в результаті процес володіє декількома унікальними властивостями. Зокрема, він охоплює довготривалу залежність (ДВЗ) і мультифрактальне масштабування незалежно один від одного.

Мультифрактальна модель мережевого трафіку, що є комбінацією мультифрактального каскаду з випадковим процесом, має всі важливі властивості, які спостерігаються в реальному трафіку, включаючи ДВЗ, мультифрактального і логнормального. Так само вона є достатньо гнучкою, щоб охопити всі мультифрактальні характеристики трафіку.

Наведені моделі генерації мультифрактального трафіку прагнуть врахувати всі характерні особливості реалізацій мережевого трафіку і є вельми громіздкими. Так, наприклад, моделі фрактального точкового процесу залежать від шести-восьми параметрів; моделі, засновані на стійких розподілах, вимагають не менше чотирьох, крім того деякі параметри мають досить складні алгоритми знаходження за реальними даними. Тому в даній роботі розглянута нова математична модель, яка буде враховувати тільки деякі основні властивості телетрафіка, а саме: освіту черг в буфері при проходженні через канал зв'язку і наявність втрати пакетів.

Задача класифікації – формалізована задача, яка містить множину об'єктів (ситуацій), поділених певним чином на класи. Задана скінченна множина об'єктів, для яких відомо, до яких класів вони відносяться. Ця множина називається вибіркою. До якого класу належать інші об'єкти невідомо. Необхідно побудувати такий алгоритм, який буде здатний класифікувати довільний об'єкт з вихідної множини.

Класифікувати об'єкт – означає, вказати номер (чи назву) класу, до якого відноситься даний об'єкт.

Класифікація об'єкта – номер або найменування класу, що видається алгоритмом класифікації в результаті його застосування до даного конкретного об'єкту.

Нехай X – множина описів об'єктів, Y – множина номерів (чи назв) класів. Існує невідома цільова залежність-відображення $y^*: X \to Y$, значення якої відомі лише на елементах скінченної навчальної вибірки $X^m = \{(x_1, y_1), ..., (x_m, y_m)\}$. Потрібно побудувати алгоритм $a: X \to Y$, здатний класифікувати довільний об'єкт $x \in X$.

Загальнішим є імовірнісне формулювання завдання. Припускається, що множина пар «об'єкт, клас» $X \times Y$ є ймовірнісним простором з невідомою ймовірнісною мірою P. Є скінченна навчальна вибірка спостережень $X^m = \{(x_1, y_1),...,(x_m, y_m)\}$, згенерована згідно з ймовірнісною мірою P. Необхідно побудувати алгоритм $a: X \to Y$, здатний класифікувати довільний об'єкт $x \in X$.

Стохастичний процес $X(t),\ t \in \mathbb{R}$ з безперервною змінною часу називається самоподібним у вузькому сенсі з параметром $H,\ 0 < H < 1,$ Якщо для будь-якого дійсного значення $a > 0$ скінченномірні розподіли для $X(at),\ t \in \mathbb{R}$ ідентичні скінченномірним розподілам $a^{-H} X(at),\ t \in \mathbb{R}$ тобто якщо для будь-яких $k \geq 1,\ t_1,\ t_2,\ ...,\ t_k \in \mathbb{R}$ і будь-яких $a > 0$:

$$\text{Law}\{X(t_2),...,\ X(t_k)\} = \text{Law}\left\{a^{-H}\left(X(at_1),\ a^{-H}X(at_2),\ ...,\ a^{-H}X(at_k)\right)\right\}.$$

Коротко рівняння можна записати у вигляді:

$$\text{Law}\{X(t),\ t \in \mathbb{R}\} = \text{Law}\left\{a^{-H}X(at),\ t \in \mathbb{R}\right\}.$$

Позначення $\text{Law}\{\cdot\}$ означає скінченномірні закони розподілу випадкового процесу. Параметр H, званий параметром Херста, являє собою міру самоподібності стохастичного процесу.

Розглянемо поняття самоподібності для процесів з дискретним часом. нехай $X = (X_1, X_2,...)$ – відрізок стаціонарного в широкому сенсі випадкового процесу з дискретним часом $t \in N = \{1,2,...\}$. Припустимо, процес X має автокорреляційну функцію такого вигляду:

$$r(k) \sim k^{-\beta} L_1(k),\ k \to \infty, \qquad (1.1)$$

де $0<\beta<1$ і L_1 – повільно змінюється на нескінченності функція, тобто $\lim\limits_{t\to\infty}\dfrac{L_1(tx)}{L_1(t)}=1$ для всіх $x>0$.

Позначимо через $X^{(m)}=\{X_1^{(m)},X_2^{(m)},...)$ – усереднений по блокам довжини m процес X, компоненти якого визначаються рівністю

$$X_t^{(m)}=\frac{1}{m}(X_{tm-m+1}+...+X_{tm}).$$

Такий ряд називається агрегованим. позначимо через $r_m(k)$ і D_m коефіцієнт кореляції і дисперсію процесу $X^{(m)}$ відповідно. Процес X називається строго самоподібним в широкому сенсі з параметром $H=1-(\beta/2),\ 0<\beta<1$, якщо

$$r_m(k)=r(k), k\in Z_+,\ m\in\{2,3,...\},$$

тобто, процес не змінює свій коефіцієнт кореляції після усереднення по блокам довжини m. Іншими словами, X є самоподібним в широкому сенсі, якщо агрегований процес $X^{(m)}$ відрізняється від початкового процесу X, щодо статистичних характеристик другого порядку.

Дискретні самоподібні процеси мають повільно спадну автокорреляційну функцію агрегованого процесу $X^{(m)}$ при $m\to\infty$, на відміну від поширених стохастичних моделей для яких виконується

$$r_m(k)\to 0,\ m\to\infty,\ k\in N.$$

Параметр Херста H, званий параметром самоподібності, знаходиться в діапазоні $0<H<1$ і являє собою ключову міру самоподібності і міру тривалості довгостроковій залежності стохастичного процесу.

Мультифрактальні заходи були введені Б. Мандельбротом для опису розподілу турбулентної дисипації. З тих пір мультифрактальні структури були виявлені в безлічі природних і технічних процесів і використовуються для моделювання та дослідження турбулентних процесів, телекомунікаційних потоків, фінансових ринків, медичних даних, геофізичних процесів.

На відміну від самоподібних процесів мультифрактальні процеси мають більш різноманітну скейлінгову поведінку:

$$\text{Law}\{X(at)\} = \text{Law}\{M(a)\cdot X(t)\}, \quad a>0, \tag{1.2}$$

де $M(a)$ – незалежна від $X(t)$ випадкова функція.

У разі самоподібного процесу $M(a)=a^H$. Для мультифрактальних процесів узагальнений показник Херста $H(a)=\log_a M(a)$ є випадковою функцією аргументу a. Співвідношення (1.2) можна переформулювати наступним чином:

$$\text{Law}\{X(at)\} = \text{Law}\{a^{H(a)}\cdot X(t)\}.$$

Моменти самоподібного випадкового процесу можна виразити як

$$\text{M}\left[\left|X(t)\right|^q\right] = \text{M}\left[\left|t^H X(1)\right|^q\right] = t^{qH}\,\text{M}\left[\left|X(1)\right|^q\right] = C(q)\cdot t^{qH},$$

де величина $C(q)=\text{M}\left[\left|X(1)\right|^q\right]$.

Для самоподібних процесів скейлінгова експонента лінійна: $\tau(q)=Hq-1$ і значення $\dfrac{\tau+1}{q}$ при $q=2$ збігається зі значенням ступеня самоподібності H.

Основними математичними моделями мультифрактальних процесів є каскадні процеси. Мультиплікативні каскади спочатку були введені А.Н. Колмогоровим для опису нерегулярного розподілу швидкості дисипації енергії турбулентного потоку. Подальше інтенсивне вивчення і розвиток каскадних моделей було викликано роботами Б. Мандельброта.

Найпростішою моделлю мультифрактального процесу з заданими властивостями є детермінований біноміальний мультиплікативний каскад. При його побудові початковий одиничний інтервал ділиться на два рівних інтервали, яким приписуються вагові коефіцієнти p_1 і $p_2 = 1 - p_1$ відповідно. Потім з кожним з інтервалів проробляється аналогічна процедура. В результаті на другому кроці є 4 інтервалу з ваговими коефіцієнтами p_1^2, $p_1 p_2$, $p_2 p_1$ і p_2^2. При числі кроків $n \to \infty$ і $p_1 \neq p_2$ ми приходимо до граничної міри, що є неоднорідною фрактальною безліччю.

Реалізація, побудована для моделювання реалізацій інформаційного трафіку, що володіє мультифрактального властивостями, може бути представлена наступним чином:

$$Y(t) = f(process,\ parameters,\ N),$$
$$parameters = estimates\,(X, N_X),$$

де $Y(t)$, $t = 1,..., N$, – модельна реалізація необхідної довжини N;

process – фрактальний гаусівський шум з дискретним часом;

parameters – параметр H;

f – експоненціальне перетворення;

$X(t)$ – досліджуваний трафік довжини N_X, $t = 1,..., N_X$;

estimates – статистичні і мультифрактальні характеристики.

1.4 Постановка задач дослідження

Трафік комп'ютерних мереж при високих коефіцієнтах використання проявляє властивості самоподібності. Через це можливе швидке перевантаження буферів при невеликих коефіцієнтах використання. Особливо

це проявляється, якщо розмір буферів був розрахований для навантаження з класичними розподілами потоків.

Для більшості мереж реальною є ситуація, коли навантаження, яке надходить ззовні, може стати більше тієї, яке може бути обслуговуваним навіть при оптимальній маршрутизації. При цьому якщо не вжити заходів по обмеженню трафіку, черги на найбільш навантажених лініях будуть необмежено зростати і, врешті-решт, перевищать розміри буферів у відповідних вузлах. Це призводить до того, що пакети, які знову надходять у вузли, у яких немає вільного місця в буфері, будуть скинуті, і повинні будуть передаватися повторно, що призводить до нераціональної витрати ресурсів мережі. Таким чином, при збільшенні навантаження реальна пропускна здатність мережі зменшується, а затримка інформації зростає.

Таким чином, утворилася "проблема самоподібності трафіку", якій присвячено велику кількість робіт і яка досі не втратила своєї актуальності.

Метою кваліфікаційної роботи є розробка моделі та методу виявлення кібератак за реалізаціями мережного трафіку на основі машинного начання. Для досягнення поставленої мети необхідно виконати наступні завдання:

– вивчення характеристик і властивостей реалізацій мультифрактального трафіку;

– вивчення характеристик і властивостей реалізацій трафіку під впливом різних типів DDOS-атак;

– побудова моделі мультифрактального мережного трафіку та реалізацій трафіку під впливом DDOS-атак;

– розробка методу виявлення реалізацій трафіку під впливом DDOS-атак на основі машинного навчання.

2 ВИБІР ТА ОБҐРУНТУВАННЯ МЕТОДУ РОЗВ’ЯЗАННЯ

2.1 Огляд методів машинного навчання

Машинне навчання (англ. machine learning) — це підгалузь інформатики, яка еволюціювала з дослідження розпізнавання образів та теорії обчислювального навчання в галузі штучного інтелекту. У 1959 року Артур Семюель визначив машинне навчання як «Галузь досліджень, яка дає комп'ютерам здатність навчатися без того, щоби їх явно програмували». Машинне навчання досліджує вивчення та побудову алгоритмів, які можуть навчатися з даних, і виконувати передбачувальний аналіз на них. Такі алгоритми діють шляхом побудови моделі зі зразкового тренувального набору вхідних спостережень, щоби здійснювати керовані даними прогнози або ухвалювати рішення, виражені як виходи, замість того, щоби суворо слідувати статичним програмним інструкціям.

Машинне навчання тісно пов'язане (і часто перетинається) з обчислювальною статистикою, дисципліною, яка також фокусується на прогнозуванні шляхом застосування комп'ютерів. Воно має тісні зв'язки з математичною оптимізацією, теорією матриць, лінійною алгеброю та копулами, які забезпечують цю галузь методами, теорією та прикладними областями. Машинне навчання застосовують в ряді обчислювальних задач, в яких розробка та програмування явних алгоритмів є нездійсненними. Приклади таких застосувань включають відфільтровування спаму, оптичне розпізнавання символів (ОРС), пошукові системи та комп'ютерний зір. Машинне навчання іноді з'єднують з добуванням даних, де друга підгалузь фокусується більше на дослідницькому аналізі даних, і є відомою як навчання без учителя.

В межах галузі аналізу даних машинне навчання є методом, який використовується для винаходження складних моделей та алгоритмів, які слугують прогнозуванню – в комерційному застосуванні це відоме як передбачувальна аналітика. Ці аналітичні моделі дозволяють дослідникам,

науковцям з даних, інженерам та аналітикам «виробляти надійні, повторювані рішення та результати» та розкривати «приховані розуміння» шляхом навчання з історичних співвідношень та тенденцій в даних.

Том Мітчелл запровадив широко цитоване формальніше визначення: «Кажуть, що комп'ютерна програма вчиться з досвіду E по відношенню до якогось класу задач T та міри продуктивності P, якщо її продуктивність у задачах з T, вимірювана за допомогою P, покращується з досвідом E.» Це визначення є видатним тому, що воно визначає машинне навчання через фундаментально операційні, а не когнітивні, терміни, слідуючи таким чином пропозиції Алана Тюрінга в його праці «Обчислювальні машини та розум», щоби питання «Чи можуть машини думати?» замінити питанням «Чи можуть машини робити те, що можемо робити ми (як істоти, які думають)?».

Задачі машинного навчання, як правило, поділяють на три широкі категорії, в залежності від природи «сигналу», якого навчається система, або «зворотного зв'язку», доступного системі, яка навчається. Цими категоріями є:

а) навчання з учителем (кероване навчання, англ. supervised learning), комп'ютерові представляють приклади входів та їхніх бажаних виходів, задані «вчителем», і метою є навчання загального правила, яке відображає входи на виходи;

б) навчання без учителя (спонтанне навчання, англ. unsupervised learning), алгоритмові навчання не дається міток, залишаючи його самому знаходити структуру в своєму вході, навчання без учителя може бути метою саме по собі (виявлення прихованих закономірностей у даних), або засобом досягнення мети (навчання ознак);

в) навчання з підкріпленням (англ. reinforcement learning): комп'ютерна програма взаємодіє з динамічним середовищем, у якому вона повинна виконувати певну мету (таку як водіння автівки), без учителя, який явно казав би їй, чи підійшла вона близько до мети, іншим прикладом є навчання грі через гру із суперником.

Між керованим та спонтанним навчанням є напівавтоматичне навчання (англ. semi-supervised learning), в якому вчитель дає неповний тренувальний

сигнал: тренувальний набір, в якому відсутні деякі (часто численні) цільові виходи. Окремим випадком цього принципу є трансдукція (англ. transduction), коли під час навчання відомий повний набір випадків задачі, крім частини цілей, яких бракує.

Серед інших задач машинного навчання навчання навчатися навчається свого власного індуктивного упередження на основі попереднього досвіду. Еволюційне навчання (англ. developmental learning), розроблене для навчання роботів, породжує свої власні послідовності навчальних ситуацій (також звані навчальним планом, англ. curriculum), щоби накопичувально отримувати репертуари нових навичок шляхом автономного само-дослідження та соціальної взаємодії з вчителями-людьми, і застосування провідних механізмів, таких як активне навчання, дозрівання, рухова синергія та імітація.

Інша класифікація завдань машинного навчання виникає при розгляді бажаного виходу системи з машинним навчанням.

У класифікації (англ. classification) входи поділяються на два або більше класів, і система-учень мусить породити модель, яка відносить небачені входи до одного або більше (багатоміткова класифікація) з цих класів. Це, як правило, намагаються розв'язувати керованим чином. Прикладом класифікації є фільтри спаму, в яких входами є повідомлення електронної пошти (або чогось іншого), а класами є «спам» та «не спам».

У регресії (англ. regression), також керованій задачі, виходи є безперервними, а не дискретними.

У кластеруванні (англ. clustering) набір входів повинно бути поділено на групи. На відміну від класифікації, групи не відомі заздалегідь, що зазвичай робить це завданням для спонтанного навчання.

Оцінка густини знаходить розподіл входів у деякому просторі.

Зниження розмірності спрощує входи шляхом відображення їх на простір меншої розмірності. Пов'язаною задачею є тематичне моделювання, в якому програмі надається перелік документів людською мовою, і дається завдання з'ясувати, які документи охоплюють подібні теми.

Основна мета системи, яка навчається – це робити узагальнення зі свого досвіду. Узагальнення в цьому контексті є здатністю машини, яка вчиться, працювати точно на нових, не бачених прикладах/задачах після отримання досвіду навчального набору даних. Тренувальні приклади походять з якогось загалом невідомого розподілу ймовірності (який вважається представницьким для простору випадків), і система, яка вчиться, має будувати загальну модель цього простору, яка дозволяє їй виробляти достатньо точні передбачення в нових випадках.

Обчислювальний аналіз алгоритмів машинного навчання та їхньої продуктивності є галуззю теоретичної інформатики, відомої як теорія обчислювального навчання (англ. computational learning theory). Оскільки тренувальні набори є скінченними, а майбутнє є непевним, теорія навчання зазвичай не дає гарантій продуктивності алгоритмів. Натомість доволі поширеними є ймовірнісні рамки продуктивності. Одним зі шляхів кількісної оцінки похибки узагальнення є компроміс зсуву та дисперсії.

Те, наскільки добре модель, натренована наявними прикладами, передбачує виходи для невідомих випадків, називається узагальненням. Щоб узагальнення було найкращим, складність його гіпотези повинна відповідати складності функції, яка лежить в основі даних. Якщо гіпотеза є менш складною за цю функцію, то ми недовчилися. Тоді ми підвищуємо складність, і похибка тренування знижується. Але якщо наша гіпотеза є занадто складною, то ми перевчилися. Після цього ми повинні знайти гіпотезу, яка має мінімальну похибку тренування.

На додачу до рамок продуктивності, теоретики обчислювального навчання досліджують часову складність та здійсненність навчання. В теорії обчислювального навчання обчислення вважається здійсненним, якщо його може бути виконано за поліноміальний час. Є два види результатів часової складності. Позитивні результати показують, що певного класу функцій може бути навчено за поліноміальний час. Негативні результати показують, що певних класів не може бути навчено за поліноміальний час.

Існує багато схожостей між теорією машинного навчання та статистичним висновуванням, хоча вони використовують відмінні терміні.

2.2 Підходи машинного навчання

Навчання дерев рішень. Навчання дерев рішень (англ. decision tree learning) використовує як передбачувальну модель дерево рішень, яке відображує спостереження про предмет на висновки про цільове значення предмету.

Навчання асоціативних правил. Навчання асоціативних правил (англ. association rule learning) є методом виявлення цікавих зв'язків між величинами у великих базах даних.

Штучні нейронні мережі. Алгоритм навчання штучної нейронної мережі (ШНМ, англ. artificial neural network, ANN), зазвичай званої «нейронною мережею» (НМ), є алгоритмом навчання, натхненим структурою та функційними аспектами біологічних нейронних мереж. Обчислення структуруються в термінах взаємозв'язаних груп штучних нейронів, які обробляють інформацію із застосуванням конективістського підходу до обчислень. Сучасні нейронні мережі є нелінійними статистичними інструментами моделювання даних. Їх зазвичай застосовують для моделювання складних взаємозв'язків між входами та виходами, для пошуку закономірностей в даних, або для виявлення статистичної структури в невідомому спільному розподілі ймовірності спостережуваних величин.

Глибинне навчання. Падіння цін на апаратне забезпечення та розвиток графічних процесорів для особистого використання протягом останніх кількох років зробили свій внесок у розвиток поняття глибинного навчання (англ. Deep Learning), яке складається з кількох прихованих шарів штучної нейронної мережі. Цей підхід намагається моделювати спосіб, яким людський мозок обробляє світло та звук для зору та слуху. Деякими з успішних застосувань глибинного навчання є комп'ютерний зір та розпізнавання мовлення.

Індуктивне логічне програмування. Індуктивне логічне програмування (ІЛП, англ. Inductive logic programming, ILP) є підходом до навчання правил із застосуванням логічного програмування як універсального представлення вхідних прикладів, зворотного поширення, та гіпотез. Маючи кодування відомого основного знання та набір прикладів, представлені як логічна база даних фактів, система ІЛП виводитиме гіпотетичну логічну програму, яка має наслідками всі позитивні й жодні з негативних прикладів. Пов'язаною областю є індуктивне програмування, яке для представлення гіпотез розглядає будь-які види мов програмування (а не лише логічне програмування), такі як функційні програми.

Метод опорних векторів. Метод опорних векторів (англ. support vector machines, SVMs) є набором пов'язаних методів навчання з учителем, які використовуються для класифікації та регресії. Маючи набір тренувальних прикладів, кожен з яких помічено як належний до однієї з двох категорій, алгоритм тренування методу опорних векторів будує модель, яка передбачує, чи новий приклад потрапляє до однієї категорії, чи до іншої.

Кластерування. Кластерний аналіз (англ. cluster analysis) є розподілом набору спостережень на підмножини (які називають кластерами), так, що спостереження в межах одного й того ж кластеру є подібними відповідно до деякого наперед встановленого критерію або критеріїв, в той час як спостереження, взяті з різних кластерів, є несхожими. Різні методики кластерування роблять різні припущення про структуру даних, часто визначені деякими мірами подібності, і оцінювані, наприклад, внутрішньою компактністю (подібністю членів одного й того ж кластеру) та відокремленістю між різними кластерами. Інші методи ґрунтуються на оцінюваній густині та зв'язності графа. Кластерування є методом навчання без учителя, і поширеною методикою статистичного аналізу даних.

Баєсові мережі. Баєсова мережа, мережа переконань, або спрямована ациклічна графічна модель (англ. bayesian network, belief network, directed acyclic graphical model) — це ймовірнісна графічна модель, яка представляє

набір випадкових величин та їхніх умовних незалежностей через спрямований ациклічний граф. Наприклад, баєсова мережа може представляти ймовірнісні взаємозв'язки між хворобами та симптомами. Маючи в розпорядженні симптоми, таку мережу можна використовувати для обчислення ймовірностей наявності різних хвороб. Існують ефективні алгоритми для виконання висновування та навчання.

Навчання з підкріпленням. Навчання з підкріпленням (англ. reinforcement learning) переймається тим, як агент повинен вчиняти дії в середовищі таким чином, щоби максимізувати деяке уявлення про довготермінову винагороду. Алгоритми навчання з підкріпленням намагаються знайти політику, яка відображує стани світу на дії, які агент повинен вчиняти в цих станах. Навчання з підкріпленням відрізняється від навчання з учителем тим, що пари правильних входів/виходів йому ніколи не представляються, і не зовсім оптимальні дії ніколи явно не виправляються.

Навчання представлень. Декотрі алгоритми навчання, головно алгоритми навчання без учителя, мають на меті виявлення кращих представлень входів, наданих протягом тренування. Класичні приклади включають метод головних компонент та кластерний аналіз. Алгоритми навчання представлень (англ. representation learning) часто намагаються зберегти інформацію в своїх входах, але перетворити її таким чином, щоби зробити її зручною, часто як крок попередньої обробки перед виконанням класифікації або передбачень, уможливлюючи відбудову входів, які йдуть з невідомого розподілу, що породжує дані, й у той же час не будучи обов'язково точними для конфігурацій, які є малоймовірними за того розподілу.

Алгоритми навчання многовидів намагаються робити це за обмеження, щоби навчені представлення мали низьку розмірність. Алгоритми розрідженого кодування намагаються робити це за обмеження, щоби навчені представлення були розрідженими (мали багато нулів). Алгоритми навчання полілінійного підпростору мають на меті навчання представлень низької розмірності безпосередньо з тензорних представлень багатовимірних даних без

перетворення їх на (багатовимірні) вектори. Алгоритми глибинного навчання знаходять кілька рівнів представлення, або ієрархію ознак, в якій високорівневі, абстрактніші ознаки визначаються в термінах ознак нижчого рівня (або породжують їх). Було висловлено думку, що розумна машина — це така, яка навчається представлення, що розплутує чинники, які лежать в основі варіацій, що описують спостережувані дані.

Навчання подібностей та мір. У цій задачі машині, яка навчається, надають пари прикладів, які розглядаються як подібні, і пари менш подібних об'єктів. Їй потрібно навчитися функції подібності (або функції міри відстані), яка може передбачувати, чи є нові об'єкти подібними. Це іноді використовується в рекомендаційних системах.

Навчання розріджених словників. В цьому методі дані представляються лінійною комбінацією базисних функцій, і передбачається, що коефіцієнти є розрідженими. Нехай x є d-вимірними даними, а D є матрицею d на n, кожен стовпчик якої представляє базисну функцію. r є коефіцієнтом для представлення x за допомогою D. З математичної точки зору, навчання розрідженого словника (англ. sparse dictionary learning) означає розв'язання x = Dr, де r є розрідженим. Взагалі кажучи, передбачається, що n є більшим за d, щоби дати свободу для розрідженого представлення.

Навчання словника разом із розрідженим представленням є строго NP-складним, і є також складним і для наближеного розв'язання. Популярним евристичним методом навчання розріджених словників є K-CPM.

Навчання розріджених словників застосовувалося в кількох контекстах. У класифікації задачею є визначення, до яких класів належать раніше не бачені дані. Припустімо, що словник для кожного з класів вже було побудовано. Тоді нові дані асоціюються з таким класом, у словникові якого вони розріджено представлені найкраще. Навчання розріджених словників застосовувалося також у знешумлюванні зображень. Ключова ідея полягає в тому, що чистий клаптик зображення може бути розріджено представлено словником зображення, а шум – ні.

Генетичний алгоритм (ГА, англ. genetic algorithm, GA) – це евристичний алгоритм пошуку, який імітує процес природного добору, і використовує такі методи як мутація та схрещування для породження нового генотипу в надії знайти добрі розв'язки заданої задачі. В машинному навчанні генетичні алгоритми знаходили деякі застосування в 1980-х та 1990-х роках.

2.3 Штучні нейронні мережі

Штучні нейронні мережі, або конективістські системи – це обчислювальні системи, натхнені біологічними нейронними мережами, що складають мозок тварин. Такі системи навчаються задач (поступально покращують свою продуктивність на них), розглядаючи приклади, загалом без спеціального програмування під задачу. Наприклад, у розпізнаванні зображень вони можуть навчатися ідентифікувати зображення, які містять котів, аналізуючи приклади зображень, мічені як «кіт» і «не кіт», і використовуючи результати для ідентифікування котів в інших зображеннях. Вони роблять це без жодного апріорного знання про котів, наприклад, що вони мають хутро, хвости, вуса та котоподібні писки. Натомість, вони розвивають свій власний набір доречних характеристик з навчального матеріалу, який вони оброблюють.

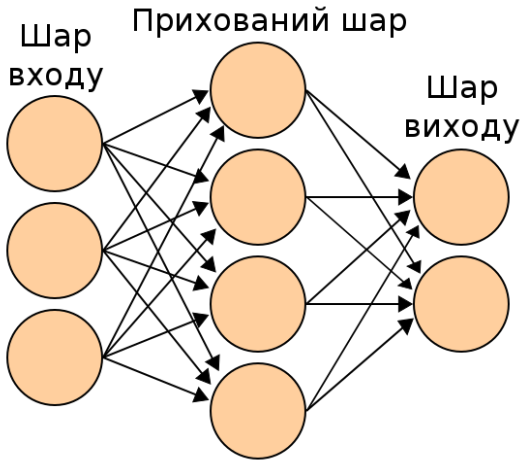

Рисунок 2.1 – Штучна нейронна мережа

Штучні нейронні мережі ґрунтуються на сукупності з'єднаних вузлів, що називають штучними нейронами (аналогічно до біологічних нейронів у головному мозку тварин). Кожне з'єднання (аналогічне синапсові) між штучними нейронами може передавати сигнал від одного до іншого. Штучний нейрон, що отримує сигнал, може обробляти його, й потім сигналізувати штучним нейронам, приєднаним до нього.

В поширених реалізаціях штучних нейронних мереж сигнал на з'єднанні між штучними нейронами є дійсним числом, а вихід кожного штучного нейрону обчислюється нелінійною функцією суми його входів. Штучні нейрони та з'єднання зазвичай мають вагу, яка підлаштовується в перебігу навчання. Вага збільшує або зменшує силу сигналу на з'єднанні. Штучні нейрони можуть мати такий поріг, що сигнал надсилається лише якщо сукупний сигнал перетинає цей поріг. Штучні нейрони зазвичай організовано в шари. Різні шари можуть виконувати різні види перетворень своїх входів. Сигнали проходять від першого (входового) до останнього (виходового) шару, можливо, після проходження шарами декілька разів.

Первинною метою підходу штучних нейронних мереж було розв'язання задач таким же способом, як це робив би людський мозок. З часом увага зосередилася на відповідності певним розумовим здібностям, ведучи до відхилень від біології. Штучні нейронні мережі використовували в ряді різноманітних задач, включно з комп'ютерним баченням, розпізнаванням мовлення, машинним перекладом, соціально-мережевим фільтруванням, грою в настільні та відеоігри, та медичним діагностуванням.

Воррен Маккалох та Уолтер Піттс (1943) створили обчислювальну модель для нейронних мереж на основі математики та алгоритмів, названою пороговою логікою. Ця модель проклала шлях до поділу досліджень нейронних мереж на два підходи. Один підхід зосереджується на біологічних процесах у мозку, тоді як інший зосереджується на застосуванні нейронних мереж до штучного інтелекту. Ця праця привела до роботи над мережами нервів та їхнього зв'язку зі скінченними автоматами.

Наприкінці 1940-х років Дональд Гебб створив гіпотезу навчання, засновану на механізмі нейропластичності, яка стала відомою як геббове навчання. Геббове навчання є спонтанним навчанням. Воно розвинулося в моделі довготривалого потенціювання. Дослідники почали застосовувати ці ідеї до обчислювальних моделей 1948 року в машинах Тюрінга типу B.

Фарлі та Кларк (1954) вперше використали обчислювальні машини, звані тоді «калькуляторами, щоби відтворити геббову мережу. Інші нейромережеві обчислювальні машини було створено Рочестером, Голландом, Гебітом та Дудою (1956).

Розенблат (1958) створив перцептрон, алгоритм для розпізнавання образів. За допомогою математичного запису Розенблат описав схему не примітивного перцептрону, таку як схема виключного «або», яке в той час обробляти нейронними мережами було неможливо.

1959 року біологічну модель, запропоновану нобелівськими лауреатами Г'юбелем та Візелем, було засновано на їхньому відкритті двох типів клітин у первинній зоровій корі: простих клітин та складних клітин.

Перші працездатні мережі з багатьма шарами було опубліковано Івахненком та Лапою 1965 року, вони стали методом групового урахування аргументів.

Дослідження нейронних мереж зазнало застою після дослідження машинного навчання Мінського та Пейперта (1969), які відкрили дві ключові проблеми з обчислювальними машинами, що обробляли нейронні мережі. Першою було те, що базові перцептрони були нездатні обробляти схему виключного «або». Другою було те, що комп'ютери не мали достатньої обчислювальної потужності для ефективного виконання роботи, потрібної великим нейронним мережам. Дослідження нейронних мереж уповільнилося, поки комп'ютери не досягли набагато більшої обчислювальної потужності.

Значну частину штучного інтелекту було зосереджено на оброблюваних алгоритмами високорівневих (символьних) моделях, які характеризують, наприклад, експертні системи зі знаннями, втіленими в правилах «якщо — то»,

поки наприкінці 1980-х років дослідження не поширилися на низькорівневе (суб-символьне) машинне навчання, що характеризується втіленням знання в параметрах пізнавальної моделі.

Ключовим активатором відновлення зацікавленості нейронними мережами та навчанням був алгоритм зворотного поширення Вербоса (1975), який ефективно розв'язував проблему виключного «або», і загалом прискорив навчання багатошарових мереж. Зворотне поширення розповсюджувало член похибки шарами в зворотному напрямку, змінюючи ваги в кожному вузлі.

В середині 1980-х років набула популярності розподілена паралельна обробка під назвою конективізму. Румельхарт та МакКлелланд (1986) описали застосування конективізму для моделювання нейронних процесів.

Метод опорних векторів та інші, значно простіші методи, такі як лінійні класифікатори, поступово наздогнали нейронні мережі за популярністю в машинному навчанні.

Попередні виклики в тренуванні глибинних нейронних мереж було успішно розв'язано за допомогою таких методів, як спонтанне попереднє тренування, в той час як доступна обчислювальна потужність зросла через застосування ГП та розподілених обчислень. Нейронні мережі було розгорнуто в великому масштабі, зокрема, в задачах розпізнавання зображень та відео. Це стало відомим як «глибинне навчання», хоча глибинне навчання не є строго синонімічним до глибинних нейронних мереж.

1992 року було представлено максимізаційне агрегування, щоби допомогти з інваріантністю відносно найменшого зсуву та терпимістю до деформації для допомоги в розпізнаванні тривимірних об'єктів.

Проблема зникання градієнту впливає на багатошарові мережі прямого поширення, які використовують зворотне поширення, а також на рекурентні нейронні мережі (РНМ). З поширенням похибок від шару до шару, вони скорочуються експоненційно з кількістю шарів, стримуючи налаштування ваг нейронів, яке ґрунтується на цих похибках, й особливо вражаючи глибинні мережі.

Щоби подолати цю проблему, Шмідгубер обрав багатошарову ієрархію мереж (1992), попередньо тренованих по одному шарові за раз за допомогою спонтанного навчання, а потім тонко налаштовуваних зворотним поширенням. Бенке (2003) в таких задачах, як відбудова зображень та визначення положень облич, покладався лише на знак градієнту (еластичне зворотне поширення).

Хінтон та ін. (2006) запропонували навчання високорівневих представлень із застосуванням послідовних шарів двійкових або дійснозначних латентних змінних з обмеженою машиною Больцмана для моделювання кожного шару. Щойно навчено достатньо багато шарів, можна застосовувати глибинну архітектуру як породжувальну модель, відтворюючи дані здійсненням вибірки моделлю донизу («спадковий прохід») від збудження ознак верхнього рівня. 2012 року Ин та Дін створили мережу, яка вчилася розпізнавати високорівневі поняття, такі як коти, лише з перегляду немічених зображень, взятих з відео YouTube.

Починаючи з 2011 року, передовою в мережах прямого поширення глибинного навчання була почерговість згорткових шарів та шарів максимізаційного агрегування, увінчаних декількома повно або частково зв'язаними шарами, за якими йде рівень остаточної класифікації. Навчання зазвичай виконується без спонтанного попереднього навчання.

Такі керовані методи глибинного навчання були першими, що досягли в певних задачах продуктивності, порівняної з людською.

Штучні нейронні мережі змогли гарантувати інваріантність до зсуву, щоби обходитися з маленькими та великими природними об'єктами у великих загромаджених сценах, лише коли інваріантність поширилася за межі зсуву, на всі навчені штучною нейронною мережею поняття, такі як розташування, тип (мітка класу об'єкта), масштаб, освітлення та інші. Це було реалізовано в еволюційних мережах, чиїми втіленнями є мережі «де–що» (англ. Where-What Networks), від WWN-1 (2008) до WWN-7 (2013).

2.4 Моделі штучних нейронних мереж

Штучна нейронна мережа – це мережа простих елементів, званих нейронами, які отримують вхід, змінюють свій внутрішній стан (збудження) відповідно до цього входу, і виробляють вихід, залежний від входу та збудження. Мережа утворюється з'єднанням виходів певних нейронів зі входами інших нейронів з утворенням орієнтованого зваженого графу. Ваги, як і функції, що обчислюють збудження, можуть змінюватися процесом, званим навчанням, який керується правилом навчання.

Складові штучної нейронної мережі: нейрони, з'єднання та ваги, функція поширення, правило навчання.

Нейрони. Нейрон з міткою j, що отримує вхід $p_j(t)$ від нейронів-попередників, складається з наступних складових:

– збудження $a_j(t)$, що залежить від дискретного параметра часу;

– можливо, порогу θ_j, що залишається незмінним, якщо його не змінить функція навчання;

– функції збудження f, яка обчислює нове збудження в заданий час $t+1$ з $a_j(t)$, та мережевого входу $p_j(t)$, даючи в результаті відношення $a_j(t+1) = f\left(a_j(t), p_j(t), \theta_j\right)$;

– та функції виходу f_{out}, яка обчислює вихід з активації $o_j(t) = f_{out}\left(a_j(t)\right)$.

Функція виходу часто є просто тотожною функцією.

Нейрон входу не має попередників, а слугує інтерфейсом входу для всієї мережі. Аналогічно, нейрон виходу не має наступників, і відтак слугує інтерфейсом виходу для всієї мережі.

З'єднання та ваги. Мережа складається зі з'єднань, кожне з яких передає вихід нейрону i до входу нейрону j. В цьому сенсі i є попередником j, а j є наступником i. Кожному з'єднанню призначено вагу w_{ij}.

Функція поширення. Функція поширення обчислює вхід $p_j(t)$ до нейрону j з виходів $o_i(t)$ нейронів-попередників, і зазвичай має вигляд $p_j(t) = \sum_i o_i(t) w_{ij}$.

Правило навчання. Правило – це правило або алгоритм, який змінює параметри нейронної мережі, щоби заданий вхід до мережі видавав придатний вихід. Цей процес навчання зазвичай полягає в зміні ваг та порогів змінних мережі.

Нейромережеві моделі можна розглядати як прості математичні моделі, що визначають функцію $f : X \to Y$, або розподіл над X, або над X та Y. Іноді моделі тісно пов'язують з певним правилом навчання. Поширене використання фрази «модель ШНМ» насправді є визначенням класу таких функцій (де членів цього класу отримують варіюванням параметрів, ваг з'єднань, або особливостей архітектури, таких як число нейронів або їхня зв'язність).

З математичної точки зору, нейромережеву функцію $f(x)$ визначають як композицію інших функцій $g_i(x)$, які може бути розкладено далі на інші функції. Це може бути зручно представляти як мережеву структуру, де стрілки зображують залежність між функціями. Широко вживаним способом компонування є нелінійна зважена сума, де $f(x) = K\left(\sum_i w_i g_i(x)\right)$, де K (що часто називають функцією збудження) є визначеною наперед функцією, такою як гіперболічний тангенс, або сигмоїдна функція, або нормована експоненційна функція. Наведене нижче розглядає набір функцій g_i як вектор $g = (g_1, ..., g_n)$.

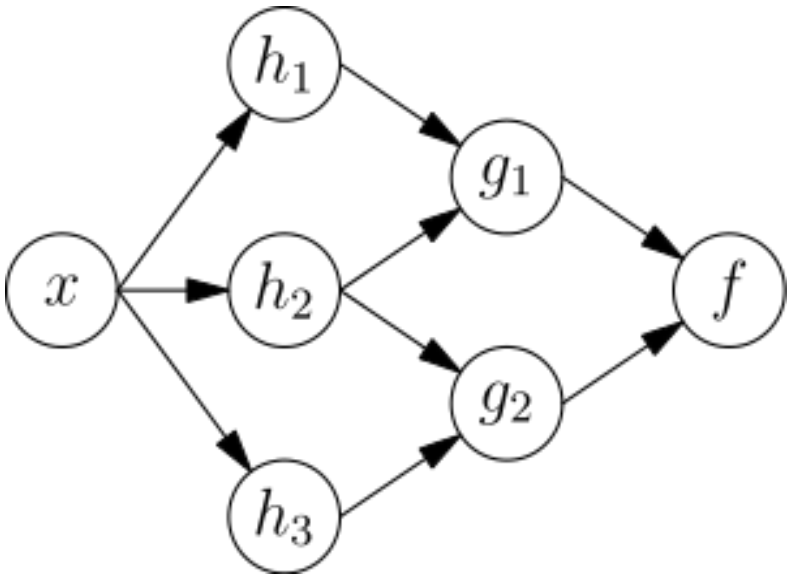

Рисунок 2.2 – Граф залежностей ШНМ

Ця схема зображує такий розклад f, із залежностями між змінними, показаними стрілками. Їх може бути інтерпретовано двома способами.

Перший погляд є функційним: вхід x перетворювано на 3-вимірний вектор h, який відтак перетворювано на 2-вимірний вектор g, який нарешті перетворювано на f. Цей погляд найчастіше зустрічається в контексті оптимізації.

Другий погляд є ймовірнісним: випадкова змінна $F = f(G)$ залежить від випадкової змінної $G = g(H)$, яка залежить від $H = h(X)$, яка залежить від випадкової змінної X. Цей погляд найчастіше зустрічається в контексті графічних моделей.

Ці два погляди є здебільшого рівнозначними. В кожному з випадків, для цієї конкретної архітектури, складові окремих шарів не залежать одна від одної (наприклад, складові g є незалежними одна від одної за заданого їхнього входу h). Це природно уможливлює якусь міру паралелізму в реалізації.

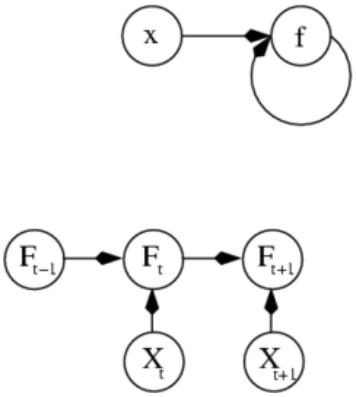

Рисунок 2.3 – Два окремі зображення графу залежностей рекурентної ШНМ

Такі мережі, як попередня, зазвичай називають мережами прямого поширення, оскільки їхній граф є спрямованим ациклічним графом. Мережі з циклами зазвичай називають рекурентними. Такі мережі зазвичай зображують у спосіб, показаний у верхній частині рисунка, де f показано як залежну від самої себе. Проте, не показано часову залежність, що мається на увазі.

2.5 Навчання

Найбільше зацікавлення нейронними мережами викликала можливість навчання. Для заданої конкретної задачі для розв'язання та класу функцій F навчання означає використання набору спостережень для знаходження $f^* \in F$, яка розв'язує цю задачу в певному оптимальному сенсі.

Це тягне за собою визначення такої функції витрат $C: F \to \mathbb{R}$, що, для оптимального розв'язку f^*, $C(f^*) \leq C(f) \ \ \forall f \in F$ – тобто, жоден розв'язок не має витрат, менших за витрати оптимального.

Функція витрат C є важливим поняттям у навчанні, оскільки вона є мірою того, наскільки далеким є певний розв'язок від оптимального розв'язку задачі, яку потрібно розв'язати. Алгоритми навчання здійснюють пошук простором розв'язків, щоби знайти функцію, яка має найменші можливі витрати.

Для тих застосувань, де розв'язок залежить від даних, витрати обов'язково мусять бути функцією від спостережень, бо інакше модель не матиме зв'язку з даними. Їх часто визначають як статистику, для якої може бути зроблено лише наближення. Як простий приклад, розгляньмо задачу знаходження моделі f, яка зводить до мінімуму $C = E\left[(f(x) - y)^2 \right]$ для пар даних (x, y), що витягають з певного розподілу D. В практичних ситуаціях ми матимемо лише N зразків з D, і, відтак, для наведеного вище прикладу ми будемо зводити до мінімуму лише $\hat{C} = \frac{1}{N} \sum_{i=1}^{N} \left(f(x_i) - y_i \right)^2$. Таким чином, витрати зводяться до мінімуму над вибіркою з даних, а не над усім розподілом.

Коли $N \to \infty$, мусить застосовуватися якийсь різновид інтерактивного машинного навчання, в якому витрати знижуються з кожним побаченим зразком. І хоча інтерактивне машинне навчання часто застосовують за незмінного D, найкориснішим воно є у випадку, коли цей розподіл повільно змінюється з часом. В нейромережевих методах якісь різновиди інтерактивного машинного навчання часто застосовують для скінченних наборів даних.

Навіть коли можливо визначити функцію витрат ad hoc, часто використовують конкретні витрати (функцію витрат), або через те, що вони мають бажані властивості (такі як опуклість), або через те, що вони природно виникають з певного формулювання задачі (наприклад, у ймовірнісному формулюванні як обернені витрати можна використовувати апостеріорну ймовірність моделі). Кінець кінцем, функція витрат залежить від задачі.

Штучні нейронні мережі може бути треновано розрізнювально за допомогою стандартного алгоритму зворотного поширення. Зворотне поширення – це метод обчислення градієнту функції втрат (видає витрати, пов'язані з заданим станом) по відношенню до ваг в штучної нейронної мережі.

Основи неперервного зворотного поширення було виведено в контексті теорії керування Келлі 1960 року та Брайсоном 1961 року з використанням принципів динамічного програмування. 1962 року Дрейфус опублікував простіше виведення, засноване лише на ланцюговому правилі. Брайсон та Хо описали його як метод багатоетапної оптимізації динамічних систем 1969 року. 1970 року Ліннаінмаа остаточно опублікував загальний метод автоматичного диференціювання (АД) дискретних зв'язних мереж вкладених диференційовних функцій. Він відповідає сучасному баченню зворотного поширення, яке є ефективним навіть коли мережі є розрідженими. 1973 року Дрейфус застосував зворотне поширення для пристосування параметрів контролерів пропорційно градієнтам похибок. 1974 року Вербос зазначив можливість застосування цього принципу до штучних нейронних мереж, і 1982 року він застосував метод АД Ліннаінмаа до нейронних мереж способом, який широко застосовується сьогодні. 1986 року Румельхарт, Хінтон та Вільямс зазначили, що цей метод може породжувати корисні внутрішні представлення вхідних даних в прихованих шарах нейронних мереж. 1993 року Ван став першим переможцем міжнародного змагання з розпізнавання образів за допомогою зворотного поширення.

Уточнення ваг зворотного поширення можливо здійснювати за допомогою стохастичного градієнтного спуску із застосуванням наступного рівняння:

$$w_{ij}(t+1) = w_{ij}(t) + \eta \frac{\partial C}{\partial w_{ij}} + \xi(t),$$

де η – темп навчання;

C – функція витрат (втрат);

$\xi(t)$ – стохастичний член.

Вибір функції витрат залежить від таких чинників як тип навчання (кероване, спонтанне, з підкріпленням тощо) та функції збудження. Наприклад, при здійсненні керованого навчання на задачі багатокласової класифікації поширеними варіантами вибору функції збудження та функції витрат є нормована експоненційна функція та функція перехресної ентропії відповідно. Нормалізовану експоненційну функцію визначають як $p_j = \frac{\exp(x_j)}{\sum_k \exp(x_k)}$, де p_j представляє ймовірність класу (вихід вузла j), а x_j та x_k представляють загальний вхідний сигнал вузлів j та k одного й того ж рівня відповідно. Перехресну ентропію визначають як $C = -\sum_j d_j \log(p_j)$, де d_j представляє цільову ймовірність для вузла виходу j, а p_j є виходом ймовірності для j після застосування функції збудження.

Це можливо використовувати для виведення обмежувальних коробок об'єкта у вигляді двійкової маски. Їх також використовують для багатомасштабної регресії для підвищення точності визначення положення. Регресія на основі ГНМ може навчатися ознак, що схоплюють геометричну інформацію, на додачу до того, що вони слугують добрим класифікатором. Вони усувають вимогу явного моделювання частин та їхніх взаємозв'язків. Це допомагає розширити розмаїття об'єктів, яких можна навчитися. Модель складається з декількох шарів, кожен з яких має випрямляльний лінійний вузол як функцію збудження для нелінійного перетворення. Деякі шари є згортковими, тоді як деякі є повнозв'язними. Кожен згортковий шар має

додаткове максимізаційне агрегування. Мережу тренують для зведення до мінімуму похибки L2 для передбачування маски, що пробігає весь тренувальний набір, що містить обмежувальні коробки, представлені як маски.

До альтернатив зворотному поширенню належать машини екстремального навчання, «безпоширні» мережі, тренування без пошуку з вертанням, «безвагові» мережі та не-конективістські нейронні мережі.

2.6 Парадигми навчання

Існує три основні парадигми навчання, кожна з яких відповідає певній навчальній задачі. Ними є кероване навчання, спонтанне навчання та навчання з підкріпленням.

Кероване навчання. Кероване навчання використовує набір прикладів пар $(x, y), x \in X, y \in Y$, і має на меті пошук функції $f : X \to Y$ в дозволеному класі функцій, яка відповідає цим прикладам. Іншими словами, ми хочемо вивести відображення, на яке натякають ці дані; функцію витрат пов'язано з невідповідністю між нашим відображенням та даними, і вона неявно містить апріорне знання про предметну область.

Широко вживаними витратами є середньоквадратична похибка, яка намагається звести до мінімуму усереднену квадратичну похибку між виходом мережі, $f(x)$, та цільовим значення y над усіма прикладами пар. Зведення до мінімуму цих витрат за допомогою градієнтного спуску для класу нейронних мереж, званого багатошаровими перцептронами (БШП), дає алгоритм зворотного поширення для тренування нейронних мереж.

Задачами, що вписуються до парадигми керованого навчання, є розпізнавання образів (відоме також як класифікація) та регресія (відома також як наближення функцій). Парадигма керованого навчання є застосовною також і до послідовнісних даних (наприклад, до розпізнавання писання вручну, мовлення та жестів). Його можна розглядати як навчання з «учителем» у

вигляді функції, яка забезпечує постійний зворотний зв'язок стосовно якості отриманих досі розв'язків.

Спонтанне навчання. У спонтанному навчанні даються якісь дані x та функція витрат для зведення до мінімуму, якою може бути будь-яка функція від даних x та виходу мережі f.

Функція витрат залежить від задачі (предметної області моделі) та наявних апріорних припущень (неявних властивостей моделі, її параметрів, та спостережуваних змінних).

Як тривіальний приклад, розгляньмо модель $f(x) = a$, де a є сталою, а витрати $C = E\left[(x - f(X))^2\right]$. Зведення до мінімуму цих витрат дає значення a, яке дорівнює середньому значенню даних. Функція витрат може бути набагато складнішою. Її вигляд залежить від застосування: наприклад, у стисненні її може бути пов'язано зі взаємною інформацією між x та $f(x)$, тоді як у статистичному моделюванні її може бути пов'язано з апостеріорною ймовірністю моделі за заданих даних (зауважте, що в обох цих прикладах ці величини зводитимуться до максимуму, а не до мінімуму).

Задачі, що вписуються до парадигми спонтанного навчання, є загалом задачами оцінювання; до застосувань належать кластерування, оцінювання статистичних розподілів, стиснення та фільтрування.

Навчання з підкріпленням. У навчанні з підкріпленням дані x зазвичай не надаються, а породжуються взаємодією агента з середовищем. В кожен момент часу t агент виконує дію y_t, а середовище породжує спостереження x_t та миттєві витрати c_t відповідно до якоїсь (зазвичай невідомої) динаміки. Метою є визначити таку стратегію вибору дій, яка зводить до мінімуму якусь міру довготривалих витрат, наприклад, очікувані сукупні витрати. Динаміка середовища та довготривалі витрати для кожної зі стратегій є зазвичай невідомими, але їх може бути оцінено.

Формальніше, середовище моделюють як марковський процес вирішування (МПВ) зі станами $s_1, ..., s_n \in S$ та діями $a_1, ..., a_n \in A$ з наступними

розподілами ймовірності: розподілом миттєвих витрат $P(c_t \mid s_t)$, розподілом спостережень $P(x_t \mid s_t)$ та переходом $P(s_{t+1} \mid s_t, a_t)$ тоді як стратегію визначають як умовний розподіл над діями за заданих спостережень. Взята разом, ця двійка відтак утворює марковський ланцюг (МЛ). Метою є визначити таку стратегію (тобто, МЛ), що зводить витрати до мінімуму.

Штучні нейронні мережі часто використовують у навчанні з підкріпленням як частину загального алгоритму. Динамічне програмування було зв'язано з штучними нейронними мережами (давши нейродинамічне програмування) Берцекасом та Цициклісом і застосовано до багатовимірних нелінійних задач, таких як присутні в маршрутизувані транспорту, природокористуванні та медицині, через здатність штучних нейронних мереж пом'якшувати втрати точності навіть при зниженні щільності ґратки дискретизації для чисельного наближення розв'язків первинних задач керування.

Задачами, які вписуються до парадигми навчання з підкріпленням, є задачі керування, ігри та інші задачі послідовного ухвалювання рішень.

2.7 Алгоритми навчання

Тренування нейронної мережі по суті означає вибирання однієї моделі з множини дозволених моделей (або, в баєсовій системі, визначення розподілу над множиною дозволених моделей), що зводить витрати до мінімуму. Доступні численні алгоритми для тренування нейромережевих моделей; більшість із них можна розглядати як безпосереднє застосування теорії оптимізації та статистичного оцінювання.

Більшість використовують градієнтний спуск якогось вигляду, застосовуючи зворотне поширення для обчислення фактичних градієнтів. Це здійснюється просто взяттям похідної від функції витрат по відношенню до параметрів мережі, з наступною зміною цих параметрів у пов'язаному з

градієнтом напрямку. Алгоритми тренування зворотним поширенням поділяються на три категорії:

– найшвидший спуск (зі змінним темпом навчання та імпульсом, еластичним зворотним поширенням);

– квазі-ньютонові (Бройден – Флетчер – Гольдфарб – Шанно, однокрокова січна);

– Левенберг – Марквардт та спряжені градієнти (уточнення Флетчера – Рівза, уточнення Поляка – Ріб'єра, перезапуск Павелла – Біла, масштабований спряжений градієнт).

2.8 Використання та застосування

Використання штучних нейронних мереж вимагає розуміння їхніх характеристик:

– вибір моделі: це залежить від представлення даних та застосування (надмірно складні моделі уповільнюють навчання);

– алгоритм навчання: існують численні компроміси між алгоритмами навчання (майже кожен алгоритм працюватиме добре з правильними гіперпараметрами для тренування на певному наборі даних);

– робастність: якщо модель, функція витрат та алгоритм навчання обрано належним чином, то отримувана в результаті штучна нейронна мережа може стати робастною.

Можливості штучних нейронних мереж підпадають під наступні широкі категорії:

– наближення функцій, або регресійний аналіз, включно з передбачуванням часових рядів, наближенням пристосованості та моделюванням;

– класифікація, включно з розпізнаванням образів та послідовностей, виявленням нововведень та послідовним ухвалюванням рішень;

– обробка даних, включно з фільтруванням, кластеруванням, сліпим відокремлюванням сигналу та стисненням;

– робототехніка, включно зі скеровуванням маніпуляторів та протезів;

– автоматичне керування, включно з числовим програмним керуванням.

Через свою здатність відтворювати та моделювати нелінійні процеси, штучні нейронні мережі знайшли застосування в широкому діапазоні дисциплін.

До областей застосування належать ідентифікація систем та керування (керування транспортними засобами, передбачування траєкторії, автоматизація виробничих процесів, природокористування), квантова хімія, гра в ігри та ухвалювання рішень (короткі нарди, шахи, покер), розпізнавання образів (радарні системи, ідентифікування облич, класифікування сигналів, розпізнавання об'єктів та ін.), розпізнавання послідовностей (жестів, мовлення, рукописного тексту), медична діагностика, фінанси (наприклад, автоматизовані системи торгівлі), добування даних, унаочнення, машинний переклад, соціально-мережеве фільтрування та фільтрування спаму електронної пошти.

Штучні нейронні мережі застосовували в діагностуванні раку, включно з раком легенів, простати, колоректальним раком, а також щоби відрізнювати лінії ракових клітин, сильно схильні до розповсюдження, від менш схильних до розповсюдження ліній, із застосуванням лише інформації про форму клітин.

Штучні нейронні мережі також використовували для побудови чорноскринькових моделей в геонауках: гідрологія, моделювання океану та прибережна інженерія, та геоморфологія є лише деякими з прикладів такого роду.

2.9 Модель самоподібного процесу з важкими хвостами на основі фрактального броунівського руху

Як математична модель природних випадкових процесів, що володіють фрактальними властивостями, часто розглядається фрактальний броунівський

рух, яке знайшло широке застосування у фізиці, хімії, біології, економіці та теорії мережевого трафіку. Однак нормальний закон розподілу, яким володіють реалізації ФБР, не містить важких хвостів, характерних для розподілів більшості фрактальних процесів.

В даному підрозділі представлена розробка математичної моделі самоподібного стохастичного процесу на основі ФБР, що враховує як ступінь самоподібності, так і тяжкість хвоста розподілу ряду. Модель використовує досить простий математичний апарат, що дозволяє без участі експертів по реальному тимчасовому ряду створити адекватний модельний ряд.

Гаусівський процес X(t) є фрактальним броунівським рухом з параметром $H, 0 < H < 1$, якщо збільшення випадкового процесу

$$\Delta X(\tau) = X(t+\tau) - X(t)$$

мають гаусівський розподіл:

$$P(\Delta X < x) = \frac{1}{\sqrt{2\pi}\sigma_0 \tau^H} \cdot \int_{-\infty}^{x} \mathrm{Exp}\left[-\frac{z^2}{2{\sigma_0}^2 \tau^{2H}} \right] dz,$$

де σ_0 – коефіцієнт дифузії.

ФБР з параметром $H = 0{,}5$ збігається з класичним броунівським рухом. Збільшення ФБР називаються фрактальним гаусівським шумом (ФГШ). Дисперсія ФГШ підпорядковується співвідношенню $D[X(t+\tau) - X(t)] = {\sigma_0}^2 \tau^{2H}$.

Збільшення фрактального броунівського руху ΔX мають властивість самоподібності у вузькому сенсі, тобто випадкові процеси $X(t+\tau) - X(t)$ і $a^{-H} X(t + a\tau) - X(t)$ мають однакові скінченномірні розподіли для будь-якого $a > 0$.

Теорема. Збільшення фрактального броунівського руху ΔX мають властивість самоподібності у вузькому сенсі, тобто $X(t+\tau) - X(t) \triangleq a^{-H} \left(X(t + a\tau) - X(t) \right)$, для будь-якого $a > 0$.

Існує кілька методів побудови ФБР для випадку дискретного часу. Одним з найбільш використовуваних на практиці є метод послідовного випадкового складання Фосса. На n-тому кроці алгоритму ми отримуємо значення реалізації ФБР для $1+2^n$ значень часу t_i. Дисперсія доданків n-го покоління дорівнює

$$\sigma_n^{\,2} = \frac{\sigma_{n-1}^{\,2}}{2^{2H}} = \frac{\sigma_0^{\,2}}{2^{2Hn}}. \qquad (2.1)$$

Процес, запропонований Фоссом, призводить до узагальненого броунівського руху при будь-якому дозволі. На рисунку 2.4 представлена реалізація ФБР, отримана методом Фосса.

Теоретично ФГШ є моделлю самоподібного процесу з певним показником Херста і з відповідною довгостроковій залежністю. Однак ця модель має істотний недолік: відсутність важких хвостів, характерних для багатьох природних процесів. На практиці найбільш часто використовуваними розподілами з важкими хвостами є розподіли Парето, Коші, Леві і логарифмічно нормальний розподіл.

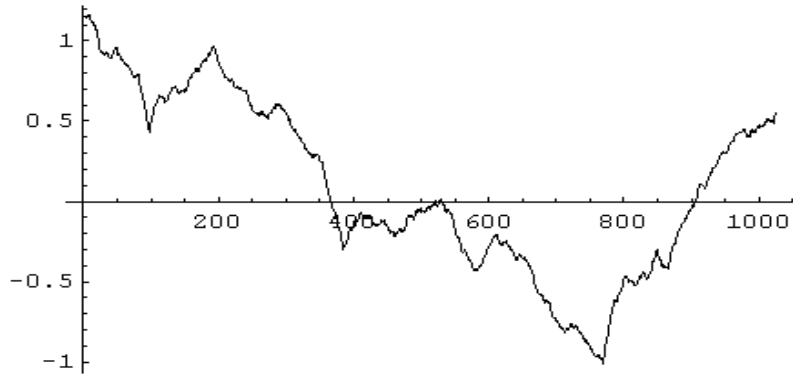

Рисунок 2.4 – Реалізація ФБР, отримана методом Фосса, H = 0.6

Величина η має логарифмічно нормальний розподіл $\mathrm{LogN}(a;\sigma)$, якщо $\eta = \mathrm{Exp}[\xi]$, Де випадкова величина ξ розподілена нормально $N(a;\sigma)$.

Щільність розподілу випадкової величини η

$$p_\eta(y) = \frac{1}{y\sqrt{2a\sigma}} e^{-\frac{(\ln y - a)^2}{2\sigma^2}}, y > 0;\ p_\eta(y) = 0,\quad y \le 0.$$

Для випадкової величини з логнормальний розподілом математичне очікування і дисперсія відповідно рівні:

$$M[\eta] = e^a \cdot e^{\frac{\sigma^2}{2}}, D[\eta] = e^{a^2} \cdot e^{\sigma^2} \cdot (e^{\sigma^2} - 1).$$

На рисунку 2.5 наведені графіки щільності логнормального розподілу з різними параметрами a і σ. Тяжкість хвоста $P[\xi > x]$ залежить від ставлення $F[\eta] = \frac{D[\eta]}{M[\eta]}$, яке називається індексом дисперсії або фактором Фано. Таким чином, величина $F[\eta]$ є кількісною характеристикою тяжкості хвоста логнормального розподілу.

Також було запропоновано підхід, який заснований на функціональному перетворенні ФГШ. Запропоноване перетворення зберігає довгострокову залежність випадкового процесу і переводить його в самоподібний процес з важкими хвостами. Як випадковий процес, що породжує модельні реалізації з заданими характеристиками самоподібності і вагою хвостів, запропоновано використовувати наступне функціональне перетворення ФГШ:

$$Y(\tau) = b \cdot \mathrm{Exp}[k \cdot X(\tau)],$$

де $X(\tau)$ – ряд ФГШ з заданим параметром H на інтервалі часу τ;

b, k – параметри, що регулюють тяжкість хвостів розподілу.

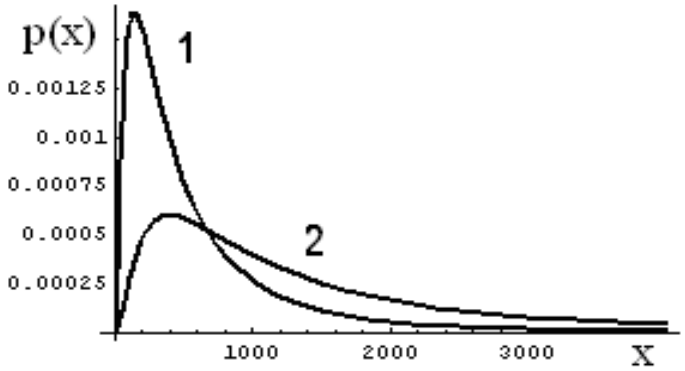

Рисунок 2.5 – Щільності логарифмічно нормальних розподілів,

$$F[\eta_2] > F[\eta_1]$$

ФГШ $X(\tau)$ має нормальний розподіл $X \cong N\left(0,\sigma^2(\tau)\right)$, з дисперсією $\sigma^2(\tau) = \sigma_0{}^2\tau^{2H}$, де σ_0 – коефіцієнт дифузії. В цьому випадку $Y(\tau)$ має логарифмічно нормальний розподіл $Y \cong \mathrm{LogN}(0,\sigma^2)$.

Затвердження. Нехай $X(\tau)$ – ряд збільшень ФБР з параметром самоподібності H на інтервалі часу τ. Випадковий процес, що є експонентою ФГШ $Y(\tau) = \mathrm{Exp}[X(\tau)]$ є самоподібний випадковий процес в широкому сенсі з тим же параметром Херста H, що і вихідний ФГШ.

Доведення. Для доказу самоподібності в широкому сенсі нам досить показати, що процеси $Y(\tau)$ і $a^{-H}Y(a\tau)$ мають однакові одномірні функції розподілу:

$$\mathrm{P}(Y(\tau) < y) = \mathrm{P}(a^{-H}Y(a\tau) < y). \qquad (2.2)$$

Так як $X(\tau) \sim N(0,\sigma\cdot\tau^H)$, то $Y(\tau) = \mathrm{Exp}[X(\tau)] \sim \mathrm{LogN}(0,\sigma\cdot\tau^H)$ і ліва частина виразу (2.2) дорівнює:

$$\mathrm{P}(\mathrm{Y}(\tau) < \mathrm{y}) = \frac{1}{\sqrt{2\pi}\sigma\tau^{\mathrm{H}}}\cdot\int_0^{\ln y}\mathrm{Exp}\left[-\frac{\mathrm{z}^2}{2\sigma^2\tau^{2\mathrm{H}}}\right]\mathrm{dz}.$$

Права частина виразу (2.1) дорівнює:

$$\mathrm{P}(a^{-H}Y(a\tau)<y)=\mathrm{P}(Y(a\tau)<a^{H}y)=\frac{1}{\sqrt{2\pi}\sigma(a\tau)^{H}}\cdot\int_{0}^{\ln a^{H}y}\mathrm{Exp}\left[-\frac{z^{2}}{2\sigma^{2}(a\tau)^{2H}}\right]dz=$$

$$=\begin{bmatrix}z=ua^{H}\\u=za^{-H}\\dz=a^{H}du\end{bmatrix}=\frac{1}{\sqrt{2\pi}\sigma a^{H}\tau^{H}}\int_{0}^{\ln y}\mathrm{Exp}\left[\frac{-(ua^{H})^{2}}{2\sigma^{2}a^{2H}\tau^{2H}}\right]\cdot a^{H}du=$$

$$=\frac{1}{\sqrt{2\pi}\sigma\tau^{H}}\int_{0}^{\ln y}\mathrm{Exp}\left[\frac{-u^{2}}{2\sigma^{2}\tau^{2H}}\right]du\,.$$

Таким чином, твердження доведено.

Числові характеристики випадкового процесу $Y(\tau)=b\cdot\mathrm{Exp}[k\cdot X(\tau)]$ мають вигляд:

$$M[Y(\tau)]=b\cdot\mathrm{Exp}[\frac{1}{2}k^{2}\sigma^{2}(\tau)],\ D[Y(\tau)]=b^{2}\cdot\mathrm{Exp}[k^{2}\sigma^{2}(\tau)]\left(\mathrm{Exp}[k^{2}\sigma^{2}(\tau)]-1\right)$$

або, використовуючи величину індексу дисперсії $F[Y(\tau)]$:

$$\begin{cases}M[Y(\tau)]=b\cdot\mathrm{Exp}[\frac{1}{2}k^{2}\sigma^{2}(\tau)];\\F[Y(\tau)]=b\cdot Exp\left[\frac{1}{2}k^{2}\sigma^{2}(\tau)\right]\left(Exp\left[k^{2}\sigma^{2}(\tau)\right]-1\right).\end{cases}$$

Вирішуючи цю систему алгебраїчних рівнянь щодо невідомих b і k, отримуємо:

$$b=\frac{M[Y(\tau)]}{\sqrt{\frac{F[Y(\tau)]+M[Y(\tau)]}{M[Y(\tau)]}}},\quad k=\frac{\sqrt{\ln\left(\frac{F[Y(\tau)]+M[Y(\tau)]}{M[Y(\tau)]}\right)}}{\sigma(\tau)},$$

де $\sigma^{2}(\tau)$ – дисперсія ряду ФГШ.

При генерації ФБР методом Фосса величина $\sigma^2(\tau) = \sigma_0{}^2 \tau^{2H}$ являє собою дисперсію різниці випадкових величин n-го покоління і залежить від довжини моделюємого тимчасового ряду і показника Херста. Враховуючи, що початкова дисперсія $\sigma_0{}^2 = 1$ і число точок першого покоління дорівнює 3, отримуємо

$$\sigma^2(\tau) = \left(\frac{1}{2}\right)^{2H(n-3)}, \ \tau = 2^n.$$

Вибірковими параметрами, отриманими з досліджуваного ряду, є: оцінка показника Херста H; оцінка параметра $F[Y] = \frac{D[Y]}{M[Y]}$; оцінка математичного очікування процесу $M[Y]$ і довжина реалізації τ.

Висновки за розділом 2

Огляд методів машинного навчання підтверджує, що застосування нейронних мереж у сфері кібербезпеки є перспективним і ефективним методом для виявлення різних типів кіберзагроз. Гнучкість і адаптивність нейронних мереж дають їм змогу ефективно адаптуватися до мінливих патернів і характеристик кібератак, що підвищує їхню загальну ефективність. Модель мультифрактального трафіку надає важливий інструментарій для аналізу та моделювання характеристик мережевого трафіку, що істотно полегшує виявлення підозрілих активностей. Поєднання нейронних мереж і моделі мультифрактального трафіку в запропонованому методі забезпечує комплексний і покращений підхід до виявлення кіберзагроз з огляду на різноманітність загроз у сучасному кіберсередовищі.

3 ПРОГРАМНА РЕАЛІЗАЦІЯ

3.1 Огляд мови програмування Python

Python – інтерпретована об'єктно-орієнтована мова програмування високого рівня з строгою динамічною типізацією. Розроблена в 1990 році Гвідо ван Россумом. Структури даних високого рівня разом із динамічною семантикою та динамічним зв'язуванням роблять її привабливою для швидкої розробки програм, а також як засіб поєднання існуючих компонентів. Python підтримує модулі та пакети модулів, що сприяє модульності та повторному використанню коду. Інтерпретатор Python та стандартні бібліотеки доступні як у скомпільованій так і у вихідній формі на всіх основних платформах. В мові програмування Python підтримується кілька парадигм програмування, зокрема: об'єктно-орієнтована, процедурна, функціональна та аспектно-орієнтована.

Серед основних її переваг можна назвати такі:

– чистий синтаксис (для виділення блоків слід використовувати відступи);

– переносність програм (що властиве більшості інтерпретованих мов);

– стандартний дистрибутив має велику кількість корисних модулів (включно з модулем для розробки графічного інтерфейсу);

– можливість використання Python в діалоговому режимі (дуже корисне для експериментування та розв'язання простих задач);

– стандартний дистрибутив має просте, але разом із тим досить потужне середовище розробки, яке зветься IDLE і яке написане на мові Python;

– зручний для розв'язання математичних проблем (має засоби роботи з комплексними числами, може оперувати з цілими числами довільної величини, у діалоговому режимі може використовуватися як потужний калькулятор);

– відкритий код (можливість редагувати його іншими користувачами).

Python має ефективні структури даних високого рівня та простий, але ефективний підхід до об'єктно-орієнтованого програмування. Елегантний

синтаксис Python, динамічна обробка типів, а також те, що це інтерпретована мова, роблять її ідеальною для написання скриптів та швидкої розробки прикладних програм у багатьох галузях на більшості платформ.

Інтерпретатор мови Python і багата Стандартна бібліотека (як вихідні тексти, так і бінарні дистрибутиви для всіх основних операційних систем) можуть бути отримані з сайту Python www.python.org, і можуть вільно розповсюджуватися. Цей самий сайт має дистрибутиви та посилання на численні модулі, програми, утиліти та додаткову документацію.

Інтерпретатор мови Python може бути розширений функціями та типами даних, розробленими на C чи C＋＋ (або на іншій мові, яку можна викликати із C). Python також зручна як мова розширення для прикладних програм, що потребують подальшого налагодження.

TensorFlow – відкрита програмна бібліотека для машинного навчання цілій низці задач, розроблена компанією Google для задоволення її потреб у системах, здатних будувати та тренувати нейронні мережі для виявлення та розшифровування образів та кореляцій, аналогічно до навчання й розуміння, які застосовують люди. Її наразі застосовують як для досліджень, так і для розробки продуктів Google, часто замінюючи на його ролі її закритого попередника, DistBelief. TensorFlow було початково розроблено командою Google Brain для внутрішнього використання в Google, поки її не було випущено під відкритою ліцензією Apache 2.0 9 листопада 2015 року.

NumPy – розширення мови Python, що додає підтримку великих багатовимірних масивів і матриць, разом з великою бібліотекою високорівневих математичних функцій для операцій з цими масивами. Попередник Numpy, Numeric, був спочатку створений Jim Hugunin. Numpy — відкрите програмне забезпечення і має багато розробників.

Оскільки Python – інтерпретована мова, математичні алгоритми, часто працюють в ньому набагато повільніше ніж у компільованих мовах, таких як C або навіть Java. NumPy намагається вирішити цю проблему для великої кількості обчислювальних алгоритмів забезпечуючи підтримку багатовимірних масивів і безліч функцій і операторів для роботи з ними. Таким чином будь-

який алгоритм який може бути виражений в основному як послідовність операцій над масивами і матрицями працює також швидко як еквівалентний код написаний на C.

NumPy можна розглядати як гарну вільну альтернативу MATLAB, оскільки мова програмування MATLAB зовні нагадує NumPy: обидві вони інтерпретовані, і обидві дозволяють користувачам писати швидкі програми поки більшість операцій проводяться над масивами або матрицями, а не над скалярами. Перевага MATLAB у великій кількості доступних додаткових тулбоксів, включаючи такі як пакет Simulink. Основні пакети, що доповнюють NumPy, це: SciPy — бібліотека, що додає більше MATLAB-подібної функціональності; Matplotlib — пакет для створення графіки в стилі MATLAB. Внутрішньо як MATLAB, так і NumPy базується на бібліотеці LAPACK, призначеної для вирішення основних задач лінійної алгебри.

Matplotlib – бібліотека на мові програмування Python для візуалізації даних двовимірною 2D графікою (3D графіка також підтримується). Отримувані зображення можуть бути використані як ілюстрації в публікаціях.

3.2 Бібліотека TensorFlow/Keras в мові програмування Python

Бібліотека TensorFlow разом із високорівневим інтерфейсом Keras стали визнаними лідерами у світі машинного навчання та глибокого навчання. Розроблена компанією Google, TensorFlow надає потужні інструменти для створення та тренування штучних нейронних мереж. Keras, який став складовою TensorFlow, пропонує зручний інтерфейс для швидкого створення моделей глибокого навчання без великих труднощів.

Інтеграція TensorFlow і Keras дозволяє розробникам легко використовувати різноманітні алгоритми навчання та будувати ефективні моделі для різних завдань, таких як класифікація, регресія, обробка природних мов, комп'ютерний зір та інші.

Основні переваги TensorFlow/Keras:

– гнучкість та масштабованість: TensorFlow/Keras дозволяє створювати та налаштовувати різні типи нейронних мереж для вирішення різноманітних завдань, від простих одношарових перцептронів до складних глибоких нейронних мереж, бібліотека забезпечує гнучкість для реалізації різних архітектур;

– широкий спектр функцій активації та шарів: TensorFlow/Keras надає розробникам доступ до різних функцій активації та типів шарів, що спрощує налаштування моделей та дозволяє оптимізувати їх для конкретних задач;

– інтеграція з іншими бібліотеками: TensorFlow легко інтегрується з іншими популярними бібліотеками для роботи з даними та візуалізації, такими як NumPy, Pandas та Matplotlib, це робить взаємодію з даними та аналіз результатів більш ефективними;

– зручний інтерфейс Keras: Keras, як високорівневий інтерфейс для TensorFlow, дозволяє швидко створювати та навчати моделі глибокого навчання з меншими зусиллями. Високий рівень абстракції дозволяє легко експериментувати з архітектурою мережі та параметрами.

Приклади використанная:

– TensorFlow/Keras використовується для створення моделей для розпізнавання облич, визначення об'єктів та класифікації зображень (наприклад, застосування в галузі медицини для діагностики за зображеннями з медичних обладнань);

– використання TensorFlow/Keras для розробки моделей обробки природних мов, таких як аналіз тональності текстів, машинний переклад та автоматичне визначення тем;

– застосування глибоких нейронних мереж для рекомендаційних систем, які забезпечують користувачам персоналізовані рекомендації, наприклад, у відео- та музичних сервісах.

Висновки за розділом 3

Алгоритми, реалізовані в мові Python, демонструють високу ефективність, надаючи точні та швидкі результати в реальному часі. Використання бібліотеки TensorFlow/Keras для реалізації нейронних мереж забезпечує зручність у розробці, а також оптимізацію продуктивності та можливості масштабування. Алгоритми забезпечують адаптивність до нових форм кіберзагроз, що розвиваються, завдяки можливості навчання на актуальних даних і періодичному оновленню моделі. Розроблена система навчання та валідації моделей нейронних мереж дає змогу ефективно налаштовувати параметри для досягнення оптимальної продуктивності. Програмна реалізація забезпечує високий рівень точності та надійності у виявленні кіберзагроз, що робить її ефективним інструментом для кібербезпеки.

4 РЕЗУЛЬТАТИ ОБЧИСЛЮВАЛЬНОГО ЕКСПЕРИМЕНТУ ТА ЇХ АНАЛІЗ

4.1 Обчислювальний експеримент

Розв’язується задача класифікації даних на 2 класи: звичайні трафіки, та трафіки, які містять DDOS атаку. Щоб отримати дані для навчання нейронної мережі було використано модель мультифрактального трафіку на базі броунівського руху, дані для трафіків, які містять DDOS, було взято з дослідження Detection of Shoplifting on Video Using a Hybrid Network (Kirichenko L., Radivilova T., Sydorenko B., Yakovlev S.). Далі, щоб отримати трафіки, що містять DDOS атаку, згенерований трафік сумується з DDOS масивом. Було розглянуто два випадки: коли атака присутня упродовж усього масиву, та коли атака починається з певного моменту. Було згенеровано декілька навчальних наборів з різними рівнями DDOS атак та різними здвигами моменту початку атаки. Кожен з наборів містить 1000 різних масивів, половина з них містить атаку, а інша половина – ні. Для кожного з наборів було навчено нейронну мережу та проаналізовано точність класифікації. Також була навчена нейронна мережа, яка використовує всі навчальні набори одразу, та здатна одночасно розпізнавати атаки з різними рівнями інтенсивності.

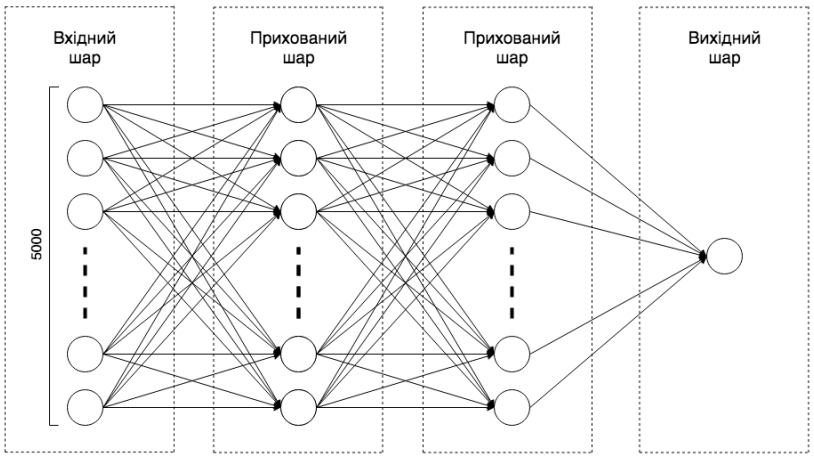

Рисунок 4.1 – Архітектура штучної нейронної мережі

Було використано нейронну мережу наступної архітектури: вхідний шар – 5000 нейронів, 2 прихованих шари – 5000 нейронів кожний, вихідний шар – 1 нейрон. На рисунку 4.1 зображена архітектура використаної нейронної мережі.

На рисунках 4.2 – 4.6 зображені дані DDOS атак, які було використано.

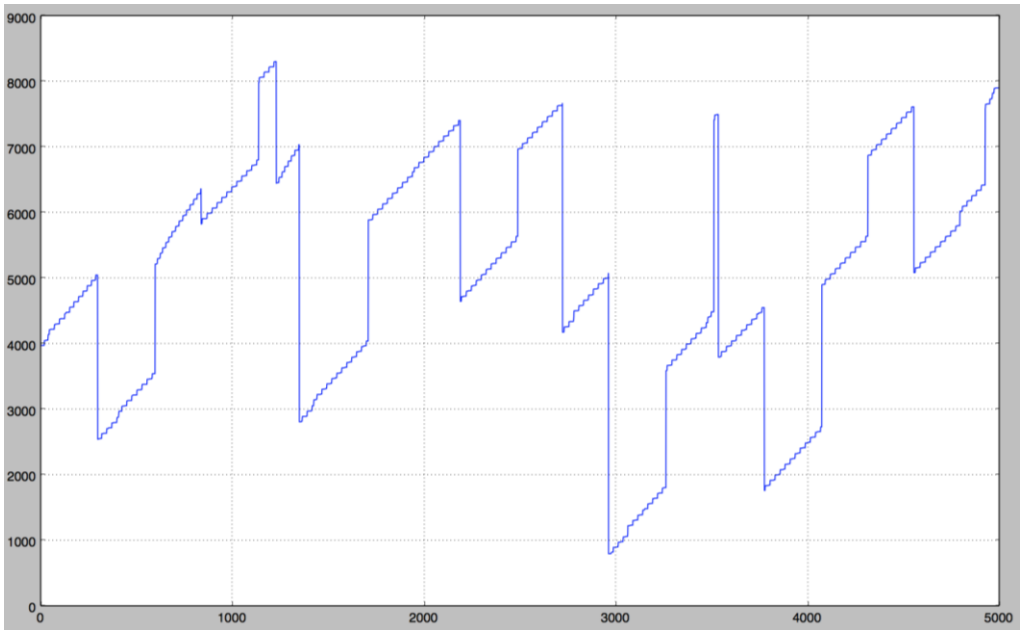

Рисунок 4.2 – DDOS атака

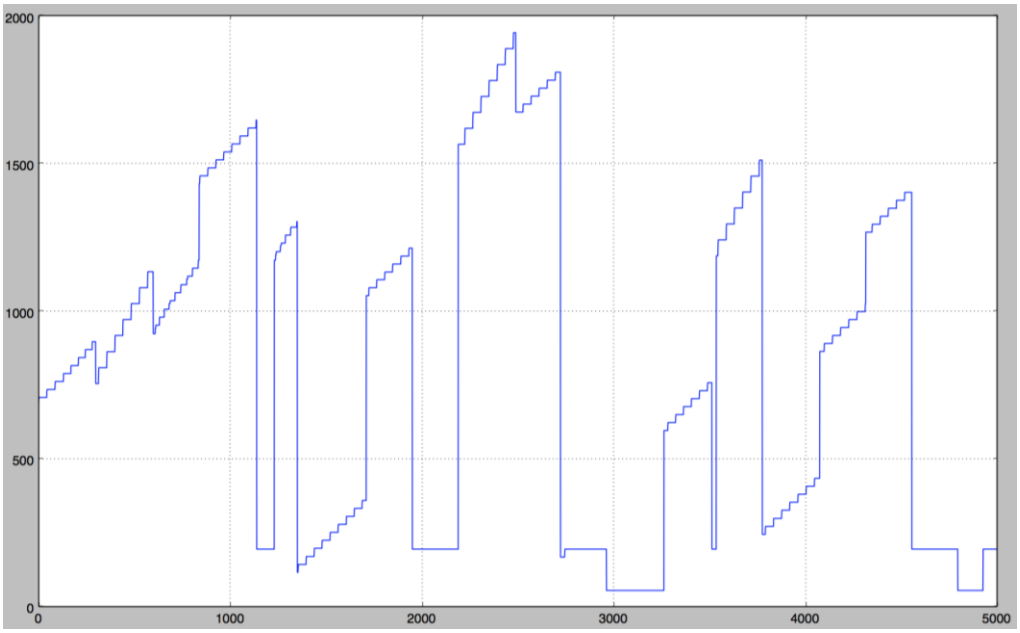

Рисунок 4.3 – DDOS атака

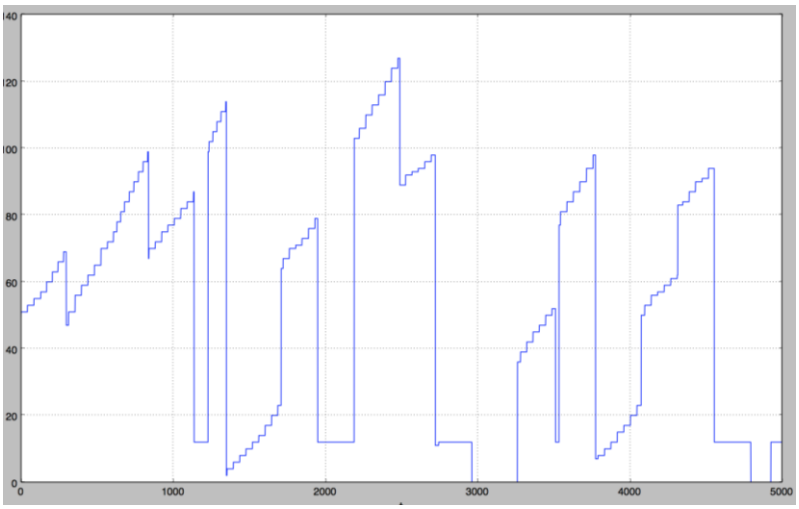

Рисунок 4.4 – DDOS атака

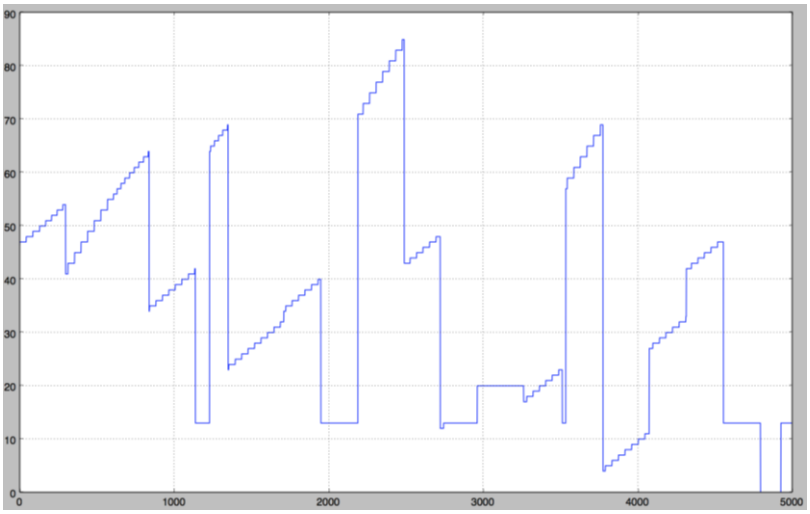

Рисунок 4.5 – DDOS атака

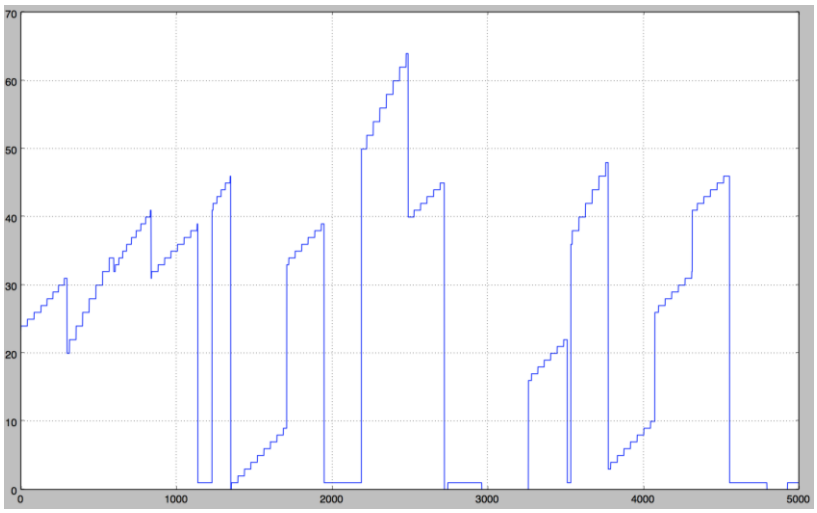

Рисунок 4.6 – DDOS атака

Далі на рисунках 4.7, 4.8 наведено приклади модельних реалізацій трафіків з параметром Херста рівним 0.7

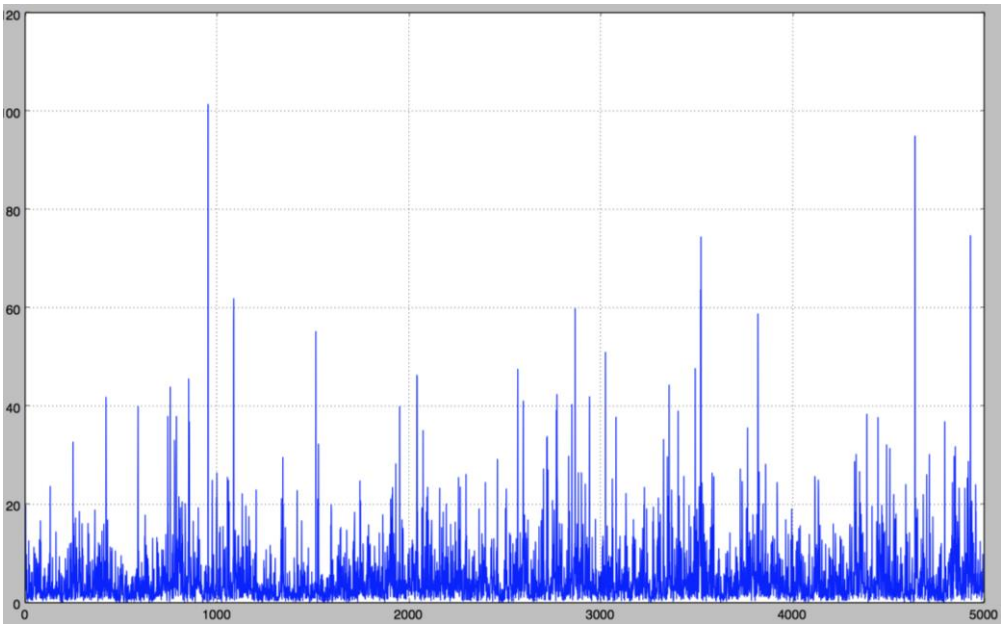

Рисунок 4.7 – Модельний трафік

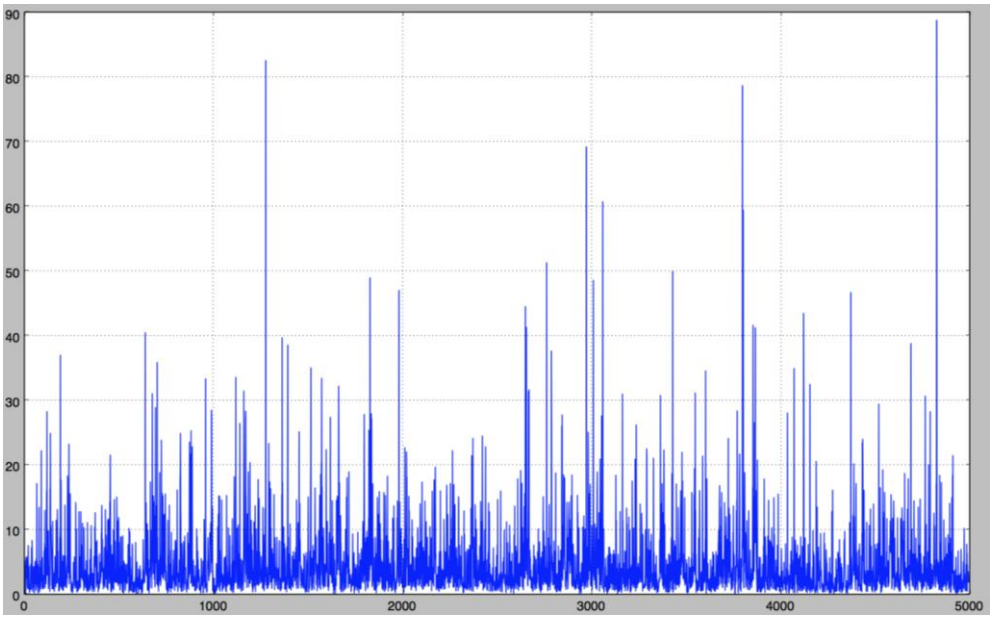

Рисунок 4.8 – Модельний трафік

Далі на рисунках 4.9 – 4.14 наведено приклади модельних трафіків, до яких було додано DDOS атаки різної сили.

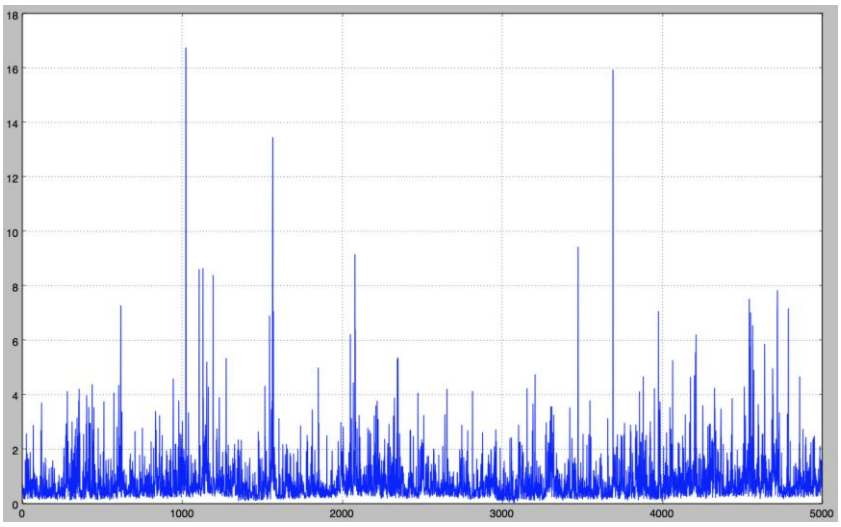

Рисунок 4.9 – Трафік з 5% DDOS

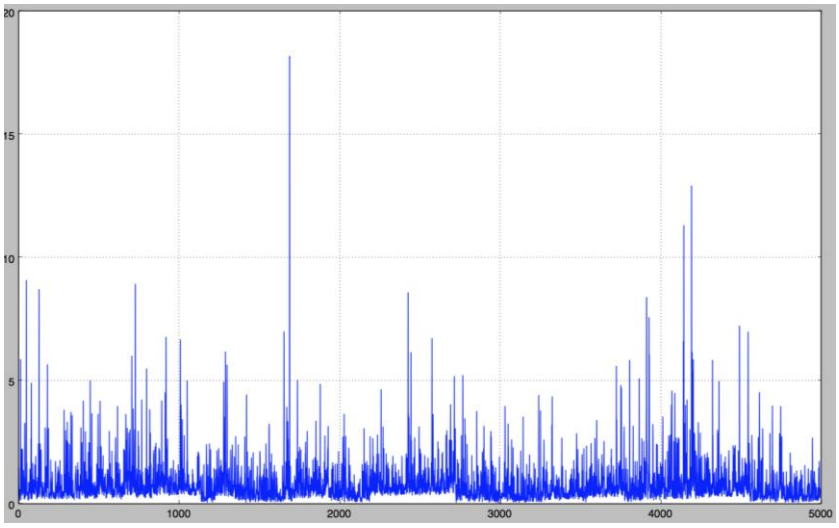

Рисунок 4.10 – Трафік з 10% DDOS

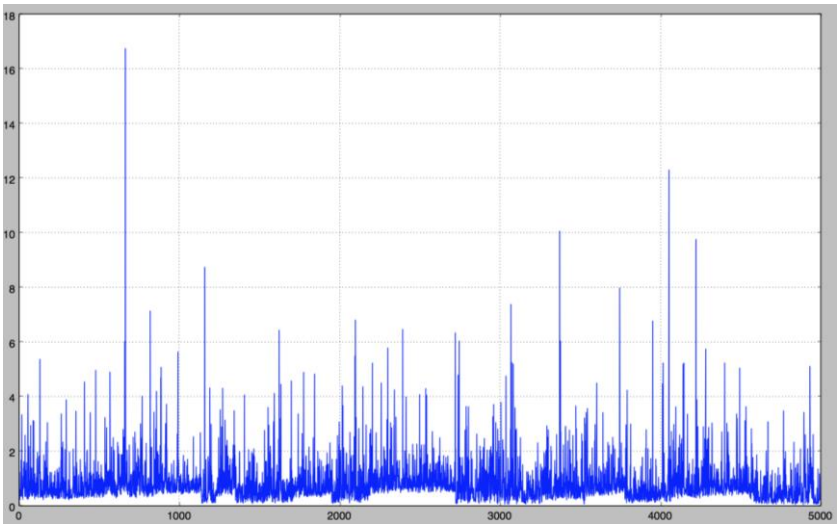

Рисунок 4.11 – Трафік з 15% DDOS

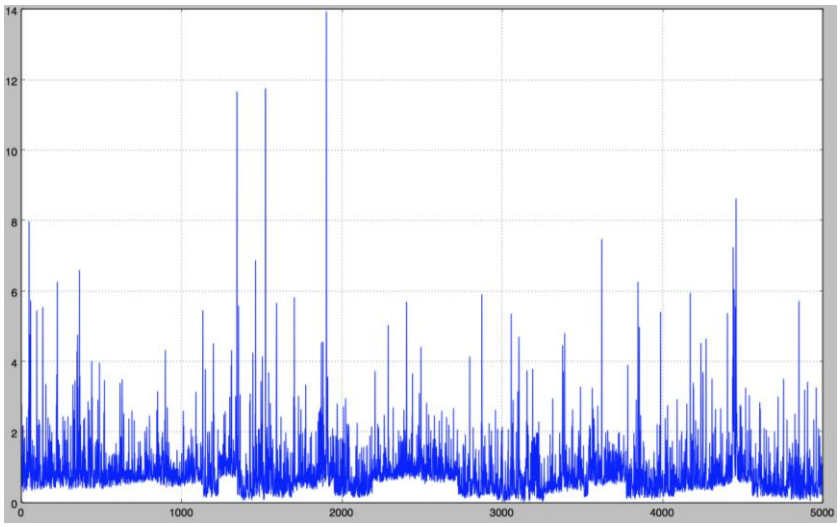

Рисунок 4.12 – Трафік з 20% DDOS

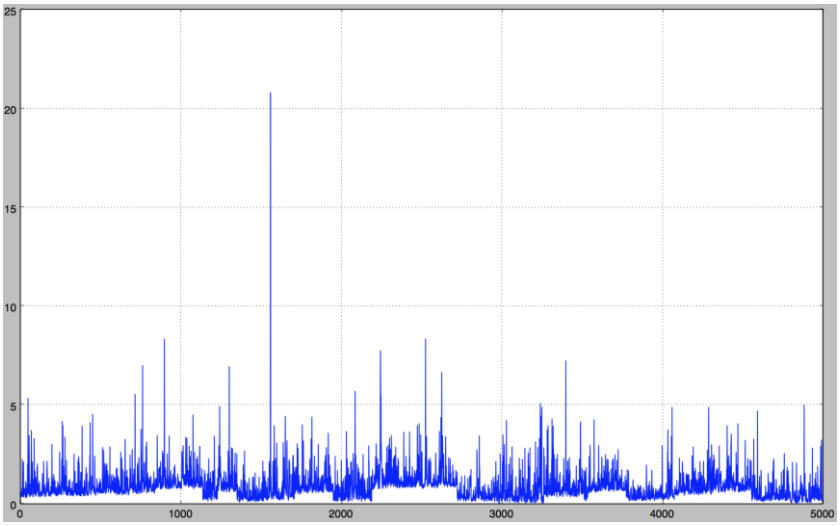

Рисунок 4.13 – Трафік з 25% DDOS

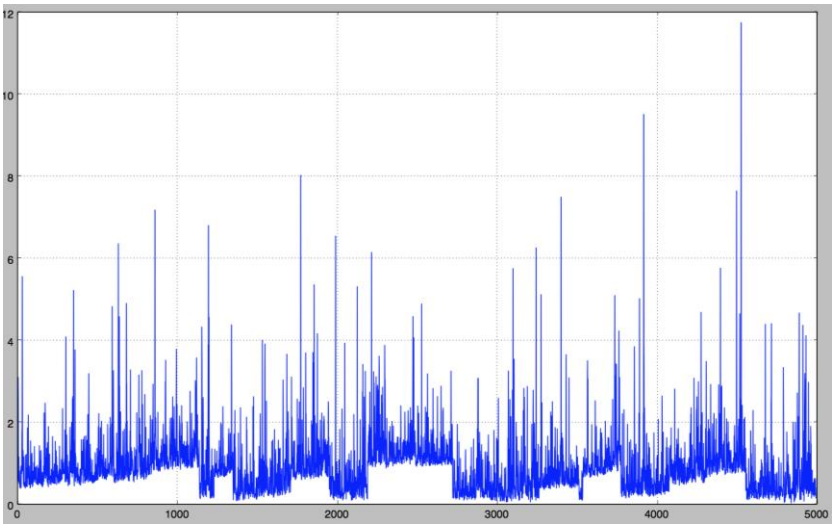

Рисунок 4.14 – Трафік з 30% DDOS

У таблиці 4.1 наведені точність класифікації 6 штучних нейронних мереж, кожну з яких було обучено на даних для певної сили DDOS атаки.

Таблиця 4.1 – Точність класифікації шести різних ШНМ

DDOS	Accuracy
5%	0.624
10%	0.776
15%	0.872
20%	0.996
25%	1.000
30%	1.000

У таблиці 4.2 наведені точність класифікації одної штучної нейронної мережі, яку було обучено на всіх тих самих даних. Загальна точність: 0.891.

Таблиця 4.2 – Точність класифікації однієї ШНМ

DDOS	Accuracy
5%	0.76
10%	0.925
15%	0.905
20%	0.925
25%	0.92
30%	0.945

Далі на рисунках 4.15 – 4.17 наведено приклади модельних трафіків, до яких було додано DDOS атаки різної сили, але тільки 40% трафіку містить атаку.

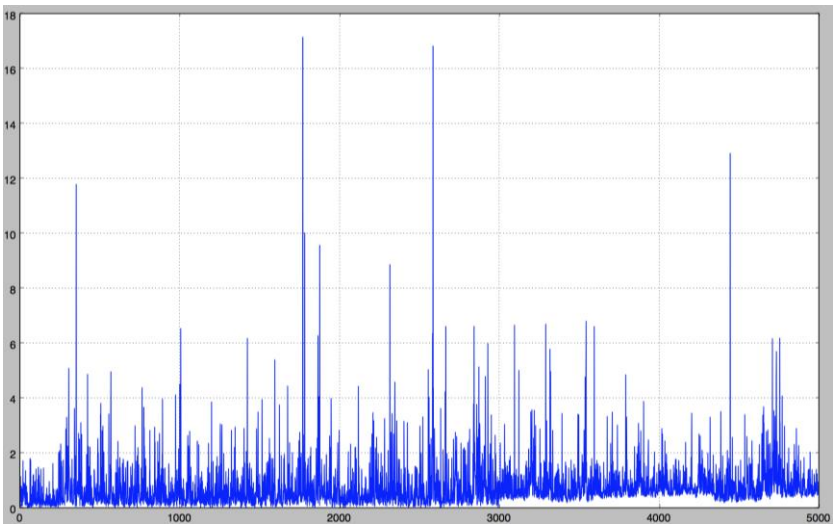

Рисунок 4.15 – Трафік з 10% DDOS. Атака в останніх 40%

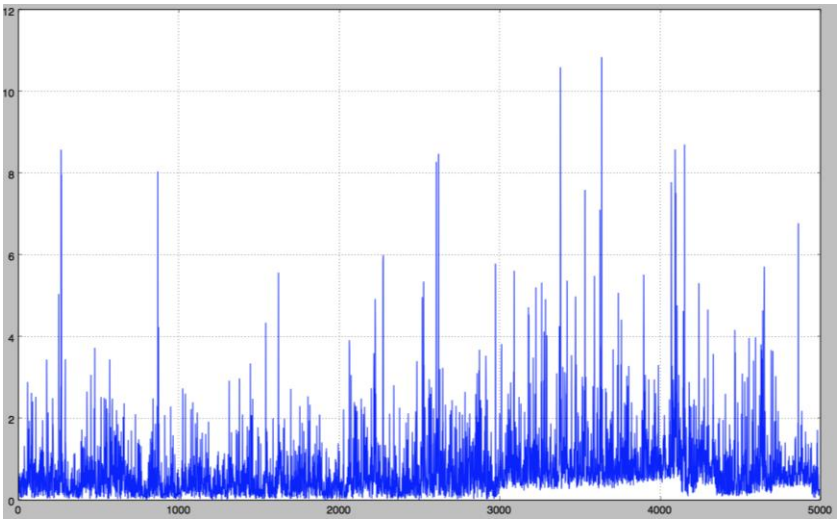

Рисунок 4.16 – Трафік з 15% DDOS. Атака в останніх 40%

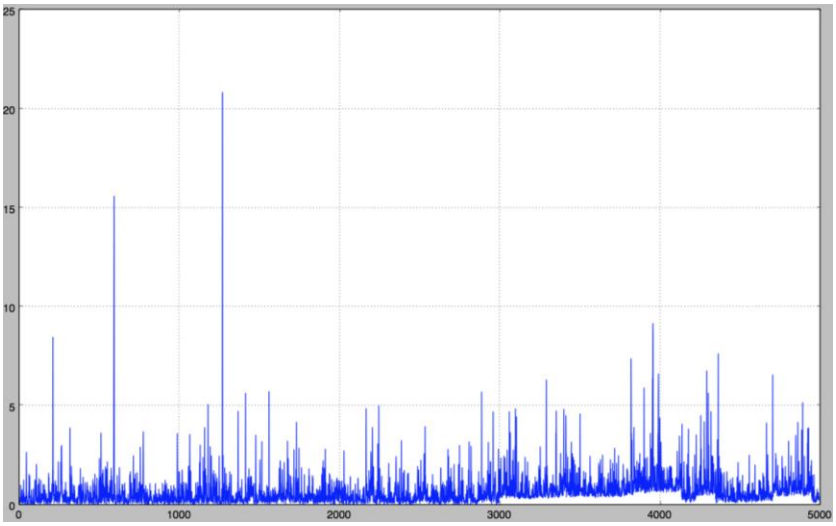

Рисунок 4.17 – Трафік з 20% DDOS. Атака в останніх 40%

Далі на рисунках 4.18 – 4.20 наведено приклади модельних трафіків, до яких було додано DDOS атаки різної сили, але тільки 20% трафіку містить атаку.

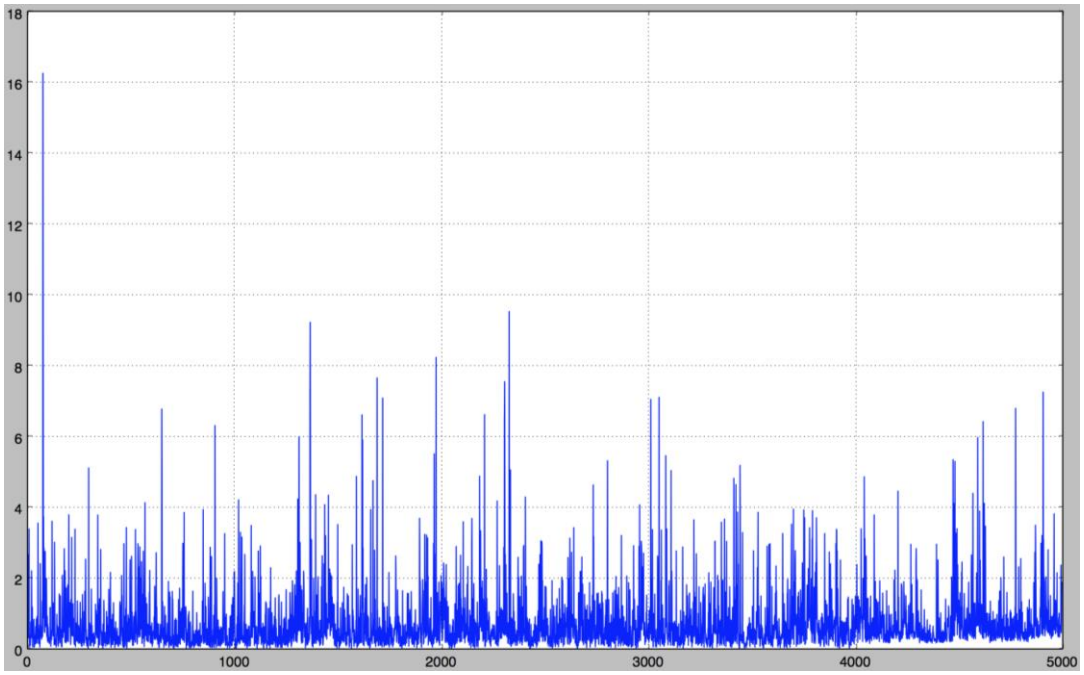

Рисунок 4.18 – Трафік з 10% DDOS. Атака в останніх 20%

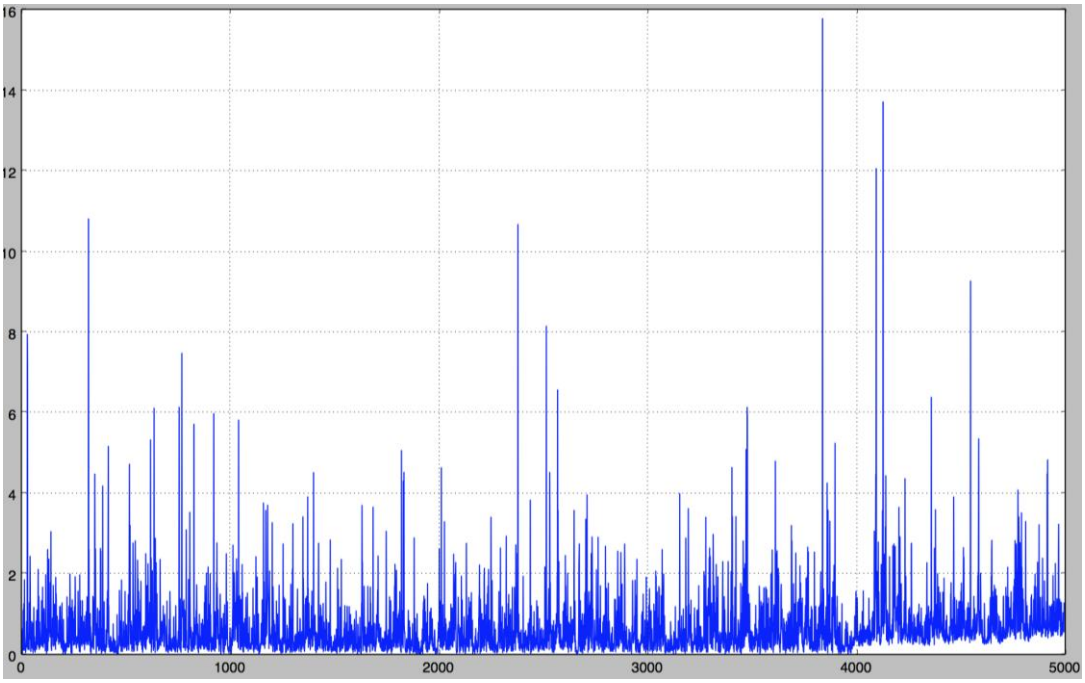

Рисунок 4.19 – Трафік з 15% DDOS. Атака в останніх 20%

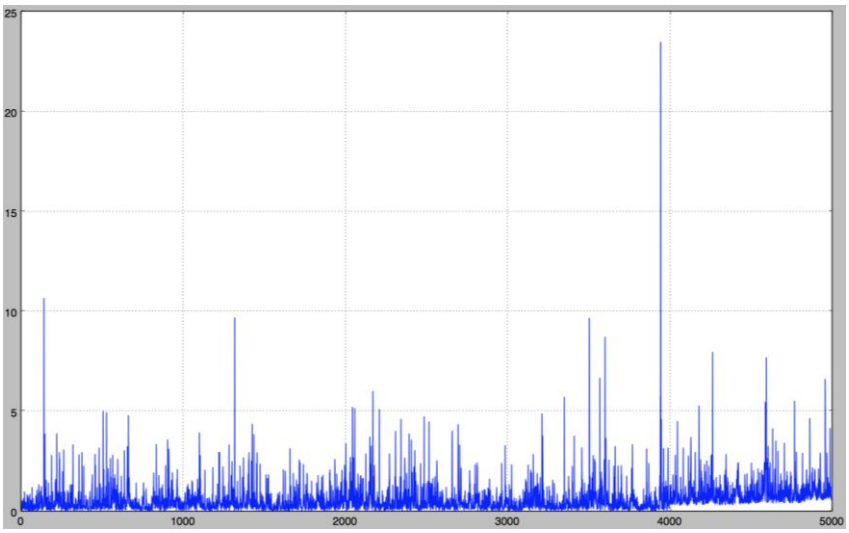

Рисунок 4.20 – Трафік з 20% DDOS. Атака в останніх 20%

У таблиці 4.3 наведені точність класифікації 3 штучних нейронних мереж, кожну з яких було обучено на даних для певної сили DDOS атаки, та здвигу початку атаки у 60%.

Таблиця 4.3 – Точність класифікації трьох різних ШНМ для здвигу 60%

Offset	DDOS	Accuracy
60%	10%	0.748
60%	15%	0.792
60%	20%	0.988

У таблиці 4.4 наведені точність класифікації 3 штучних нейронних мереж, кожну з яких було обучено на даних для певної сили DDOS атаки, та здвигу початку атаки у 80%.

Таблиця 4.4 – Точність класифікації трьох різних ШНМ для здвигу 80%

Offset	DDOS	Accuracy
80%	10%	0.716
80%	15%	0.812
80%	20%	0.864

У таблиці 4.5 наведені точність класифікації одної штучної нейронної мережі, яку було обучено на всіх тих самих даних. Загальна точність: 0.843.

Таблиця 4.5 – Точність класифікації однієї ШНМ

Offset	DDOS	Accuracy
60%	10%	0.81
60%	15%	0.87
60%	20%	0.88
80%	10%	0.685
80%	15%	0.825
80%	20%	0.855

У таблиці 4.6 наведені точність класифікації одної штучної нейронної мережі, яку було обучено на всіх попередніх даних. Загальна точність: 0.887.

Таблиця 4.6 – Точність класифікації однієї ШНМ

Offset	DDOS	Accuracy
0%	5%	0.7
0%	10%	0.89
0%	15%	0.94
0%	20%	0.945
0%	25%	0.93
0%	30%	0.935
60%	10%	0.82
60%	15%	0.905
60%	20%	0.925
80%	10%	0.74
80%	15%	0.87
80%	20%	0.91

Для побудування модельного трафіку було згенеровано фрактальний гаусівський шум (рис. 4.21).

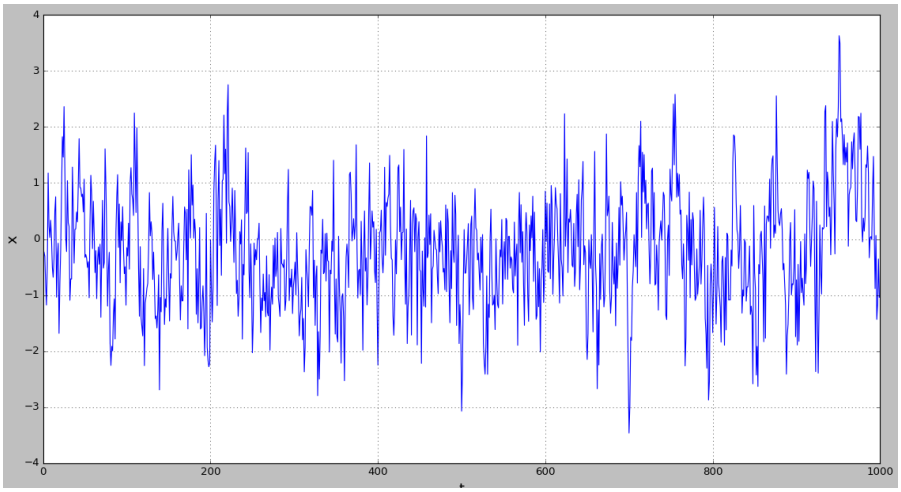

Рисунок 4.21 – Фрактальний гаусівський шум

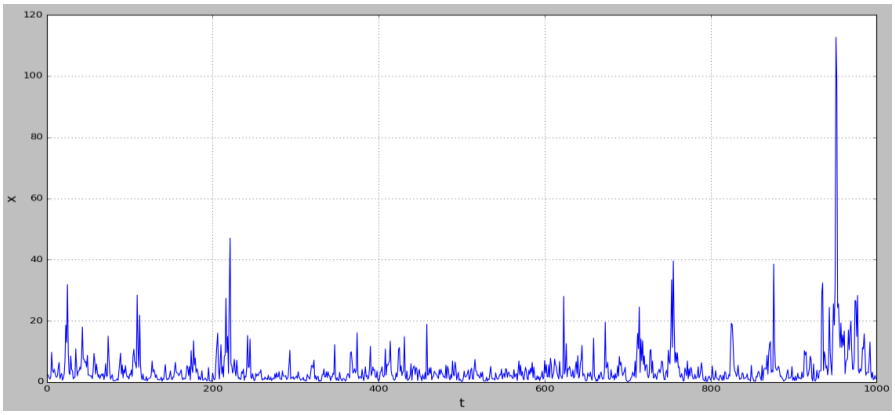

Рисунок 4.22 – Трафік, відповідний ФГШ

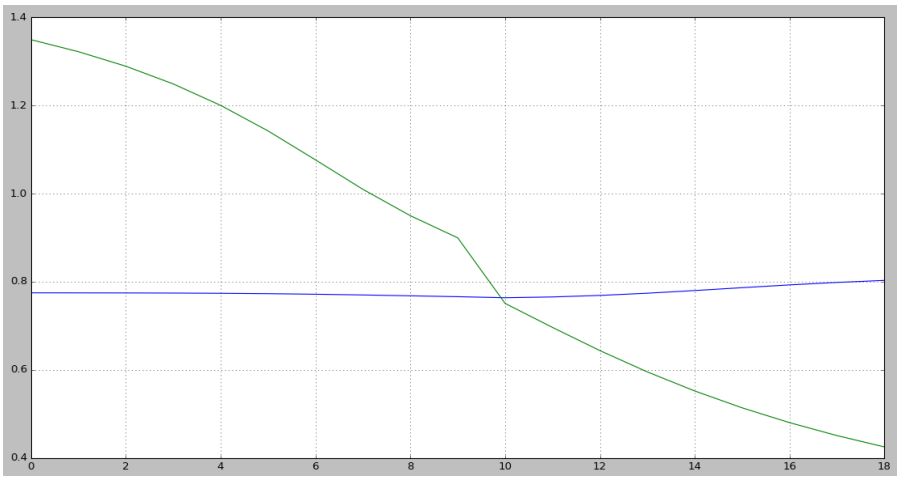

Рисунок 4.23 – Графіки $h(q)$ для ФГШ і відповідного трафіку

На рисунку 4.23 видно, що графік узагальненого показника Херста для фрактального гаусівського шуму являє собою пряму $h(q) = H$.

Для порівняння нижче представлені графіки реального TCP трафіку, його щільності розподілу і узагальненого показника Херста.

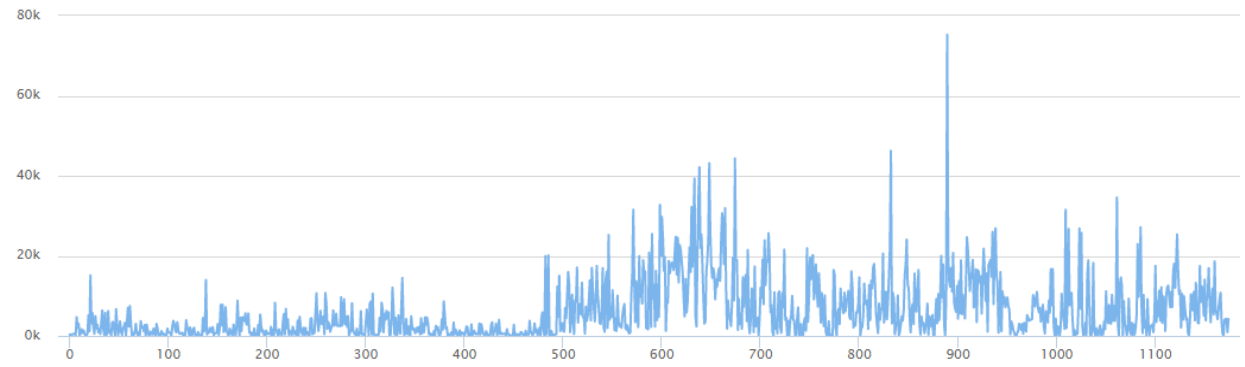

Рисунок 4.24 – Графік реального TCP трафіку

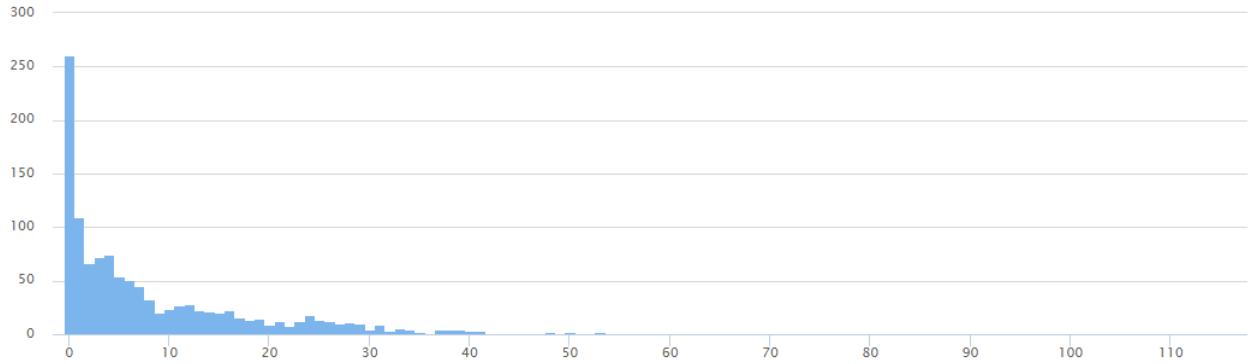

Рисунок 4.25 – Щільність розподілу реального TCP трафіку

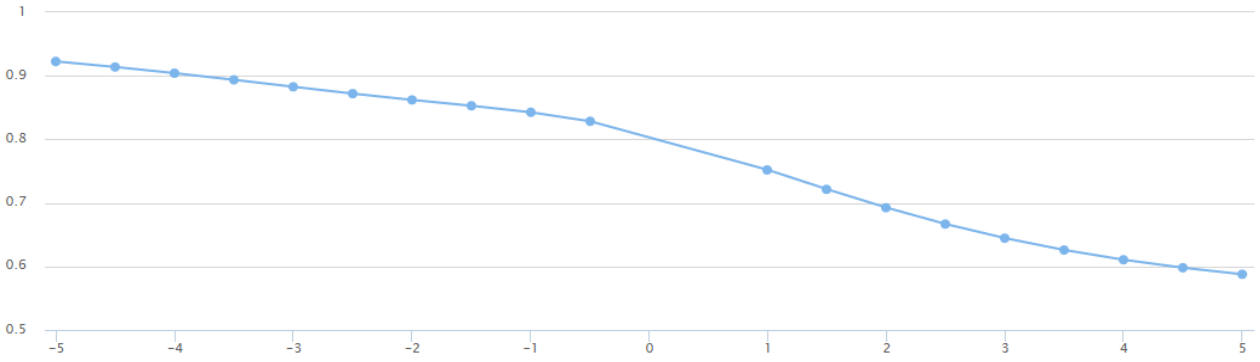

Рисунок 4.26 – Графік $h(q)$ реального TCP трафіку

4.2 Аналіз можливих застосувань

Найважливішим з можливих використань представленого дослідження є системи виявлення атак.

Система виявлення атак (вторгнень) – програмний або апаратний засіб, призначений для виявлення фактів несанкціонованого доступу в комп'ютерну систему або мережу або несанкціонованого управління ними в основному через Інтернет. Відповідний англійський термін – Intrusion Detection System (IDS). Системи виявлення вторгнень забезпечують додатковий рівень захисту комп'ютерних систем разом з системою запобігання вторгненням (IPS – англ. Intrusion Prevention System).

IDS можуть сповістити про початок атаки на мережу, причому деякі з них здатні виявляти раніш не відомі атаки. IPS не обмежуються лише оповіщенням, але й здійснюють різні заходи, спрямовані на блокування атаки (наприклад. розрив з'єднання або виконання скрипта, заданого адміністратором). На практиці досить часто програмно-апаратні рішення поєднують у собі функціональністъ двох типів систем. Їх об'єднання тоді називають IDPS (IDS і IPS).

DDoS-атаки особливо небезпечні для компаній, які пов'язані з телекомунікаційним ринком і просто не можуть собі дозволити стати жертвами зловмисників. Доля компаній, бізнес яких пов’язан з телекоммунікаціями, росте с кожним роком, тож вдосконалення методів виявлення та боротьби з DDoS атаками є дуже актуальним.

Висновки за розділом 4

Результати свідчать про те, що модель мультифрактального трафіку на базі броунівського руху може бути успішно застосована для навчання нейронних мереж у завданні класифікації мережевого трафіку. Згенеровані

атаки DDOS ефективно поєднуються з реальними даними, що сприяє створенню різноманітних навчальних сценаріїв. Створення наборів із різними рівнями атак і часовими зсувами дає змогу нейронній мережі навчатися на широкому спектрі сценаріїв, підвищуючи її стійкість. Аналіз точності класифікації для кожного набору дає уявлення про здатність нейронної мережі розпізнавати атаки за різних умов. Результати показують, що ефективність класифікації може варіюватися залежно від характеристик як атаки, так і трафіку.

ВИСНОВКИ

Під час проходження виконання кваліфікаційної роботи була розглянута актуальна на сьогоднішній день проблема DDoS-атак мережевого трафіку.

Було розглянуто завдання розпізнавання DDoS атак у мережевому трафіку. Була поставлена задача класифікації трафіків на два класи: звичайні трафіки, та атаковані трафіки. Для рішення цієї задачі було вирішено скористатися методами машинного навчання, зокрема було використано штучні нейронні мережі. Була побудована програмна реалізація двохшарової нейронної мережі.

Для того щоб отримати набори даних для навчання штучної нейронної мережі було побудована модель мережевого трафіка, та її програмна реалізація. На базі цієї моделі було згенеровано декілька наборів данних з різним рівнем та здвигом початку атаки, які було використано для навчання штучної нейронної мережі. Також було згенеровано набори даних, які використовувались для оцінки точності класифікації.

Результати досліджень підтверджують, що обраний метод розв’язання задачі класифікації дає досі точні результати, та ці результати можна застосовувати до подальшого вивчення та наступного впровадження в реальні системи виявлення вторгнень.

Також були розглянуті додатки, в яких можна або потрібно використовувати подібні дослідження. Все це свідчить про те, що, незважаючи на тривалий період вивчення проблеми кібер-атак (DDoS-атак) мережевих трафіків, проблема все ще не втратила своєї актуальності.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. Макляков В. Д. Аналіз фрактальних властивостей сумарного інформаційного трафіка. *27-й Міжнародний молодіжний форум «Радіоелектроніка і молодь у XXI столітті»* : зб. матеріалів форуму. Т. 7. Харків : ХНУРЕ, 2023. С. 165-167.

2. Sheluhin O. I., Smolskiy S. M., Osin A.V. Similar processes in telecommunications. London : John Wiley & Sons Ltd, 2007. 306 p.

3. Feldmann A., Gilbert A., Willinger W. Data networks as cascades: Investigating the multifractal nature of Internet WAN trac. *ACM Computer Communication Review*. 1998. № 28. P. 42–55.

4. Calvet L., Fisher A., Mandelbrot B. Large Deviationsand the Distribution of Price Changes. London : Cowles Foundation Discussion Paper, 1997. 30 p.

5. Doukhan P., Oppenheim G. Long Range Dependence: Theory and Applications. Bern : Birkhuser, 2002. 715 p.

6. Veneziano D., Moglen G., Bras R. Multifractal analysis: pitfalls of standard procedures and alternatives. *Phys. Rev. E*. 1995. V. 52, № 10. P. 1387–1398.

7. Kirichenko L., Radivilova T., Ryzhanov V. Applying Visibility Graphs to Classify Time Series. *Lecture Notes on Data Engineering and Communications Technologies*. 2022. № 77. P. 397–409.

8. Kirichenko L., Radivilova T., Sydorenko B. Detection of Shoplifting on Video Using a Hybrid Network. *Computation.* 2022. V. 10, № 199. P. 28–31.

9. Pichugina O., Kirichenko L., Radivilova T. Binary classification: Ensemble Methods Utilizing Decision Theory Tools. *CEUR Workshop Proceedings*. 2022, № 3348. P. 19–33.

10. Kirichenko L., Radivilova T., Bulakh V. Machine Learning in Classification Time Series with Fractal Properties. *Journal of Electronics and Telecommunications*. 2019. V. 9, № 4. P. 1–13.

11. Kirichenko L., Radivilova T., Zinkevich I. Comparative Analysis of Conversion Series Forecasting in E-commerce Tasks. *Systems and Computing*. 2002. V. 689, № 13. P. 230–242.

12. Kang J., Yang M., Zhang J. Accurately Identifying New QoS Violation Driven by High-Distributed Low-Rate Denial of Service Attacks Based on Multiple Observed Features" *Journal of Sensors*. 2015. V. 103, № 402. P. 1–11.

13. Zhou W., Wen J., Gao M., Ren H, Li P. Abnormal Profiles Detection Based on Time Series and Target Item Analysis for Recommender Systems. *Mathematical Problems in Engineerin*. 2015. V. 204, № 261. P. 1–9.

14. Suda H., Natsui M., Hanyu T. Systematic Intrusion Detection Technique for an In-vehicle Network Based on Time-Series Feature Extraction. *International Symposium on Multiple-Valued Logic*. 2018, V. 1, № 18. P. 56–61.

15. Rao U. H., Nayak U. Intrusion Detection and Prevention Systems. *The InfoSec Handbook*. 2014. V. 3, № 104. P. 225–243.

16. Al-Kasassbeh M., Al-Naymat G., Al-Hawari E. Towards Generating Realistic SNMP-MIB Dataset for Network Anomaly Detection. *International Journal Computer Science. Information Security.* 2016. V. 14, № 1. P. 1162–1185.

17. Dadkhah M., Lyashenko V. V., Deineko Z. V., Shamshirband S., Jazi M. D. Methodology of wavelet analysis in research of dynamics of phishing attacks. *International Journal of Advanced Intelligence Paradigms*. 2019. V. 12, № 4. P. 220–238.

18. Deka R., Bhattacharyya D. Self-similarity based DDoS attack detection using Hurst parameter. *Security and Communication Networks*. 2016. V. 9, №. 17. P. 4468–4481.

19. Idhammad M., Afdel K., Belouch M. Detection System of HTTP DDoS Attacks in a Cloud Environment Based on Information Theoretic Entropy and Random Forest. *Security and Communication Networks*. 2018. V. 55, № 123. P. 1–13.