Міністерство освіти і науки України

Харківський національний університет радіоелектроніки

Факультет Комп'ютерних наук

Кафедра Програмної інженерії

АТЕСТАЦІЙНА РОБОТА

Пояснювальна записка

рівень вищої освіти – другий (магістерський)

Дослідження методів побудови нейронних мереж для захисту від DDoS-атак

Виконав: студент 2 курсу, групи ІПЗм-18-1

Зельонкін Д.О.

(прізвище, ініціали)

спеціальності 121 – Інженерія програмного забезпечення

(код і повна назва спеціальності)

Освітньо-наукової програми

(тип програми)

Інженерія програмного забезпечення

(повна назва освітньої програми)

Керівник д.т.н. проф. Четвериков Г.Г.

(посада, прізвище, ініціали)

Допускається до захисту

Зав. кафедри, проф. З.В. Дудар

20__р.

ХАРКІВСЬКИЙ НАЦІОНАЛЬНИЙ УНИВІРСИТЕТ РАДІОЕЛЕКТРОНІКИ

Факультет_Комп'ютерних наук_____

Кафедра_Програмної інженерії_____

Рівень вищої освіти - другий (магістерський)

Спеціальність _121-Інженерія програмного забезпечення)_____
(код і повна назва)

Тип програми освітньо-наукова програма_____

Освітня програма _Інженерія програмного забезпечення_____

ЗАТВЕРДЖУЮ:
Зав. кафедри _____
(підпис)
«_____»_____20___ р.

ЗАВДАННЯ

НА АТЕСТАЦІЙНУ РОБОТУ

студентові _Зельонкіну Дмитру Олександровичу_____
(прізвище, ім'я, по батькові)

1. Тема роботи _Дослідження методів побудови нейронних мереж для захисту від DD0S-атак_____
затверджена наказом університету від «___» _____20___ р. №_____
заповнюється вручну після отримання наказу
2. Термін подання студентом роботи до екзаменаційної комісії
_____
3. Вихідні дані до роботи нейронні мережі, дослідження DDoS-атак, пояснювальна записка_____
_____
_____
4. Перелік питань, що потрібно опрацювати в роботі мета роботи, аналіз проблемної області і постановка задачі, аналіз можливих методів прогнозування DDoS-атак, побудова штучних нейронних мереж, навчання нейронної мережі, аналіз отриманих результатів._____
_____

5. Консультанти розділів роботи

Найменування розділу	Консультант (посада, прізвище, ім’я, по батькові)	Позначка консультанта про виконання розділу	
		підпис	дата
Спецчастина	д.т.н., проф. Четвериков Г.Г.		

КАЛЕНДАРНИЙ ПЛАН

№	Назва етапів роботи	Терміни виконання етапів роботи	Примітка
1	Аналіз предметної області	15 січня 2020 р	
2	Аналіз можливих методів прогнозування DDoS-атак	10 лютого 2020 р.	
3	Дослідження методів навчання нейронної мережі	17 лютого 2020 р	
4	Підготовка пояснювальної записки	23 березня 2020 р	
5	Спецчастина	13 квітня 2020 р.	
6	Підготовка презентації та доповіді	04 травня 2020 р.	
7	Нормоконтроль, рецензування	11 травня 2020 р.	
8	Попередній захист		
9	Занесення диплома в електронний архів		
10	Допуск до захисту у зав. кафедри		
* заповнюється вручну після виконання чергового пункту			

Дата видачі завдання ___ _____ 2020 р.

Студент _____
(підпис)

Керівник роботи _____ __проф. Четвериков Г.Г._____
(підпис) (посада, прізвище, ініціали)

РЕФЕРАТ / ABSTRACT

Пояснювальна записка: 63 с., 23 рис., 5 табл., 1 додаток, 19 джерел.

DDoS-АТАКА, МАШИННЕ НАВЧАННЯ, НЕЙРОННІ МЕРЕЖІ, КЛАСИФІКАЦІЯ, СЛОВНИК ОЗНАК, ВИБІРКА.

Об'єкт дослідження – проблема виявлення DDoS-атак та її вирішення методами машинного навчання.

Мета – розробка механізму, що дозволяє класифікувати дані сервера з метою аналізу їх на предмет ведення DDoS-атаки.

Метод дослідження – класифікація даних сервера на предмет атаки використовуючи тренувальні вибірки.

В сучасному світі все більшого значення набувають автоматизовані системи. Відмови в обслуговуванні в таких системах можуть привести до непередбачуваних наслідків. Тому актуальність питань захисту від DDoS-атак з часом буде тільки зростати.

Для протидії DDoS-атакам застосовується цілий ряд механізмів, проте особливу значимість мають засоби виявлення вторгнень. Стандартні методи аналізу статистики не дозволяють виявляти невідомі раніше атаки, тому в якості механізму вирішення даної проблеми активно виступають нейронні мережі.

Метою дослідження є аналіз заходів протидії і вдосконалення механізму захисту від DDoS-атак у вигляді засоби виявлення вторгнень на основі штучних нейронних мереж, результати якої можливо застосувати в уже існуючих системах захисту.

У даній роботі відображені результати досліджень, налаштування і застосування алгоритмів нейронної мережі для ефективного виявлення DDoS- атак.

Introductory note: 64 pages, 23 pictures, 5 tables, 1 appendix, 17 sources

DDoS-ATTACK, MASHING EDUCATION, NEURAL NETWORKS, CLASSIFICATION, DIGITAL STUDIO, CHOICE.

The object of research – the problem of detecting DDoS attacks and their solution by machine learning.

The purpose is to develop a mechanism for classifying server data for the purpose of analyzing them for the purpose of DDoS attack.

The research method is the classification of server data for an attack using training samples.

In today's world, automated systems are becoming increasingly important. Denial of service in such systems can have unpredictable consequences. Therefore, the relevance of protection against DDoS attacks will only increase over time.

A number of mechanisms are used to counteract DDoS attacks, but intrusion detection is of particular importance. Standard methods of statistics analysis do not allow detecting previously unknown attacks, so neural networks are actively acting as a mechanism for solving this problem.

The purpose of the study is to analyze counter-measures and improve the mechanism of protection against DDoS attacks in the form of means of intrusion detection based on artificial neural networks, the results of which can be applied in already existing systems of protection.

This paper presents the results of research, tuning, and application of neural network algorithms to effectively detect DDoS attacks.

ЗМІСТ

ВСТУП

Мережеві атаки є одним із лідерів рейтингу по нанесеному збитку компаніям. Найбільш небезпечною та розповсюдженою є Distributed Denial of Service (DDoS-атака). DDoS-атак є досить простотою та ефективною водночас завдяки відкритим відомостями про її реалізаціію та не потребує потужних обчислювальних ресурсів. Основна ідея цієї атаки полягає в тому, що зловмисник намагається унеможливити правильне обслуговування системи.

Одним з перспективних напрямів забезпечення безпеки мережі є використання методів виявлення в основі яких використовують штучні нейронні мережі (ШІНМ). Вони є досить ефективними для вирішення таких складних задач як розпізнавання, класифікаціія чи управління і виявлення. Штучна нейронна мережа являє собою математичну модель, а також її програмна або апаратна реалізація, побудована за принципом організації та функціонування біологічних нейронних мереж – мереж нервових клітин живого організму.

Застосування ШНМ дозволить створити ефективну систему виявлення мережевих вторгнень та підвищити рівень захисту систем від зовнішніх атак.

Метою дослідження є аналіз та вдосконалення механізму захисту протидії на основі нейромережевого підходу задля підвищення точності виявлення DDoS-атак.

Об’єктом дослідження є процес виявлення DDoS-атак.

Предметом дослідження є методи та засоби виявлення DDoS-атак. Для реалізації поставлених задач були використані методи теорії штучного інтелекту (ШІ) для проектування ШНМ; статистичні методи для підготовки початкової та кінцевої інформації для моделювання ШНМ, підходи до проектування програмного забезпечення та дослідження інтелектуальної системи.

У даній роботі описані результати досліджень, навчання та застосування алгоритмів ШНМ для результативного виявлення DDoS-атак.

1 АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ

1.1 Загальна характеристика DDoS-атак

Атака типу «відмова в обслуговуванні» (DDoS) - це спроба завдати шкоди, зробивши недоступною цільову систему, наприклад веб-сайт або додаток, для звичайних кінцевих користувачів. Зазвичай зловмисники генерують велику кількість пакетів або запитів, які в кінцевому рахунку перевантажують роботу цільової системи. Для здійснення атаки типу «розподілена відмова в обслуговуванні» (DDoS) зловмисник використовує безліч зламаних або контрольованих джерел.

Розглядаючи методи запобігання таких атак, корисно розділити їх на дві групи: атаки рівня інфраструктури і атаки рівня додатки.

До атак рівня інфраструктури зазвичай відносять атаки на транспортному та мережевому рівнях. Це найбільш поширений тип DDoS-атак, який включає в себе такі вектори, як SYN-флуд, і інші атаки відображення, такі як UDP-флуд. Подібні атаки зазвичай масові і спрямовані на те, щоб перевантажити пропускну здатність мережі або сервери додатків. Проте, такий тип атак має певні ознаки, тому їх легше виявити.

До атак рівня додатків зазвичай відносять атаки на рівні додатку. Ці атаки менш поширені, але в той же час є більш складними. Як правило, вони не настільки масові, як атаки рівня інфраструктури, але націлені на певні дорогі частини програми і призводять до того, що воно стає недоступним для реальних користувачів. Як приклад можна привести потік HTTP-запитів на сторінку входу в систему, дорогий API пошуку або навіть потоки XML-RPC Wordpress (також відомі як атаки Wordpress Pingback).

Зловмисники знаходять нові вразливості та розробляють нові підходи до створення та реалізації атак. Так, у 2018 році хакери активно використовували вразливості в протоколах DNS, NTP, SSDP, CLDAP, Chargen та інших, щоб максимально розширити масштаби своїх атак [1]. Крім того, спостерігається значне

збільшення експлуатації пристроїв IoT для генерації великого потоку пакетів та атак на рівні додатків. Найбільша атака сталася в березні 2018 року на GitHub. В історії сталася потужна DDoS-атака, яка становила 1,35 Тбіт/с або 126,9 мільйона пакетів в секунду. На рисунку 1.1 показана динаміка інтенсивності DDoS-атак.

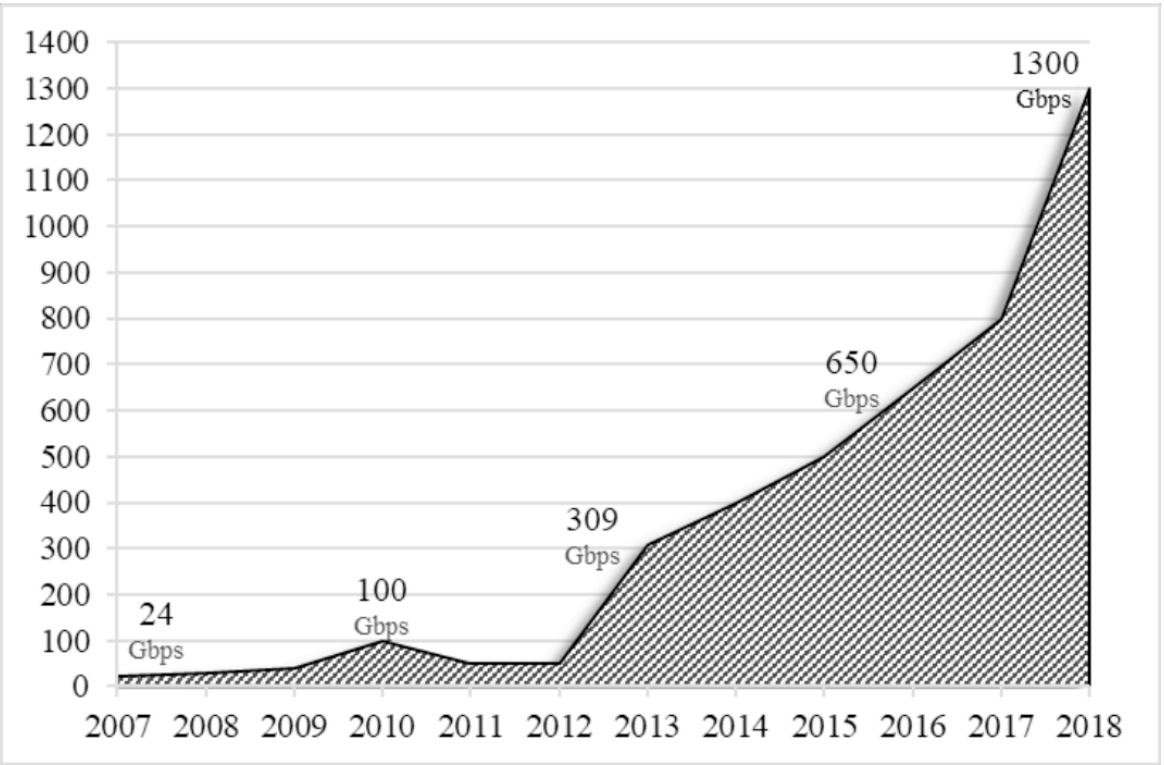

Рисунок 1.1 – Динаміка інтенсивності DDoS-атак

Типова DDoS-атака виконується в декілька етапів [2].

На першому етапі відбувається сканування мережі на наявність вразливих до атаки комп’ютерів, які в подальшому будуть використані для здійснення нападу. На сьогодні існує велика кількість сервісів які мають свої комп’ютерні мережі (ботнет) що продаються як інструмент нанесення DDoS-атаки.

На другому етапі машини агенти (ботнет) використовуються для введення в них шкідливого коду. Також на цьому етапі проводиться налаштування параметрів атаки. Вказуються IP адреса жертви, тривалість атаки, періодичність, властивості пакету.

На третьому етапі хакер дає команду на початку ведення атаки. Боти починають відсилати запити на атаковану машину. Хоча підробка IP-адрес не є необхідною умовою для успішної DDoS-атаки, зловмисники часто використовують динамічні IP-адреси, щоб приховати реальних ботів під час атаки.

Детальна класифікація DDoS-атак зображена на рисунку 1.2 [2].

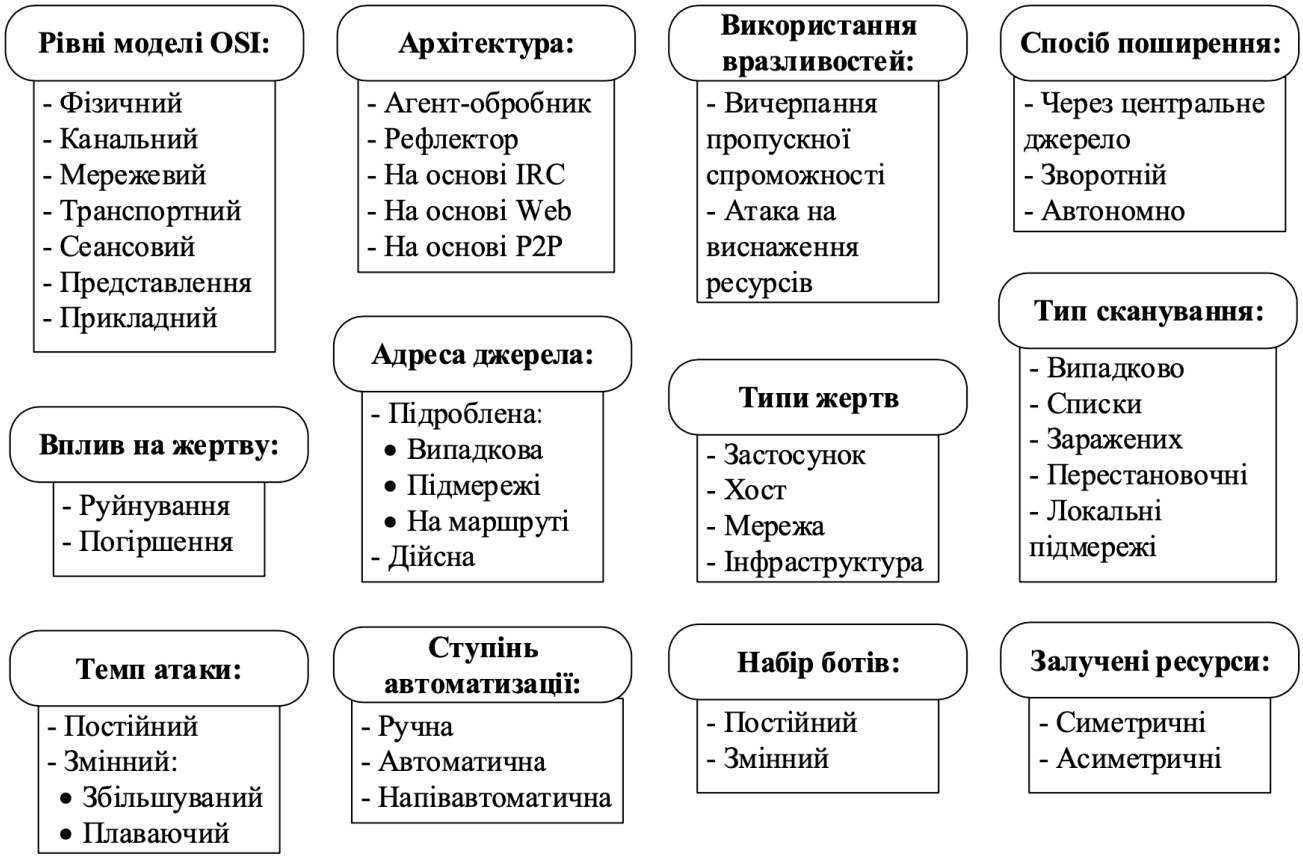

Рисунок 1.2 – Класифікація DDoS-атак

Для ефективного виконання атаки зловмисник може комбінувати різні способи, змінювати темп атак, змінювати набір боті, тощо. Всі ці чинники затрудняє процес ідентифікації атаки.

1.2 Системний аналіз методів виявлення DDoS-атак

1.2.1 Вербальна модель системи

Об'єктом дослідження є процес виявлення DDoS- атак.

Метою роботи – передбачити атаку на сервер з використанням різними методами та подальше порівняння цих методів за такими критеріями: якість прогнозування, складність алгоритму, особливості програмної реалізації, ресурсозатратність при застосуванні на великому наборі даних.

Головне завдання методів виявлення DDoS-атак полягає в аналізу запитів, відправлених на сервер, та прогнозування щодо того, знаходиться сервер під атакою на даному проміжку часу, чи ні [3].

Завданням даної роботи є порівняльний аналіз методів, заснованих на оцінці якості прогнозів та оцінці ефективності програми. Для аналіз методу роботи з даними необхідно зрозуміти як процес аналізує сигналу та можливі проблеми. Для вирішення даної проблеми будуть збудовані моделі аналізу сигналів.

1.2.2 Функціональна модель системи

Чіткого формулювання логіки та взаємодії процесів здійснюється з використанням мови моделювання бізнес-процесів IDEF0. Модельована в IDEF0 система розглядається як довільна підмножина навколишнього світу. Система має межу, яка відділяє її від зовнішнього середовища. Взаємодія системи із зовнішнім середовищем описується як вхід (дані, що маємо на початку), вихід (результат роботи), управління (алгоритм виконання роботи) і механізм (ресурси що будуть витрачені протягом виконання роботи). Перебуваючи під управління, система трансформує початкову та кінцеву інформацію використовуючи механізми. Спочатку проаналізуємо вхідний сигнал (запит до серверу).

Побудова системи в IDEF0 розпочинається з визначення контексту.

Контекстом є найбільш абстрактний опис системи та включає в себе визначення суб’єкта моделювання, точки зору на модель та цілі. В нашому випадку точкою зору є дослідник, сигнал, отриманий з серверу є суб’єктом а метою є дослідження сигналу на якість прогнозування атаки.

Контекстна діаграма IDEF0 показує функціонування системи в цілому (рисунок 1.3).

Рисунок 1.3 – Контекстна діаграма IDEF0

Основе завдання – аналіз сигналу для прогнозування атак, які виконує дослідник за допомогою комп’ютера. Інформаційним ресурсом є математичні методи, які застосовуються до вхідних даних. Результатом виконання роботи є отримання аналізу сигналу.

Проведемо декомпозицію функціональної системи щоб розглянути частини більш детально. Декомпозиція зображена на рисунку 1.4.

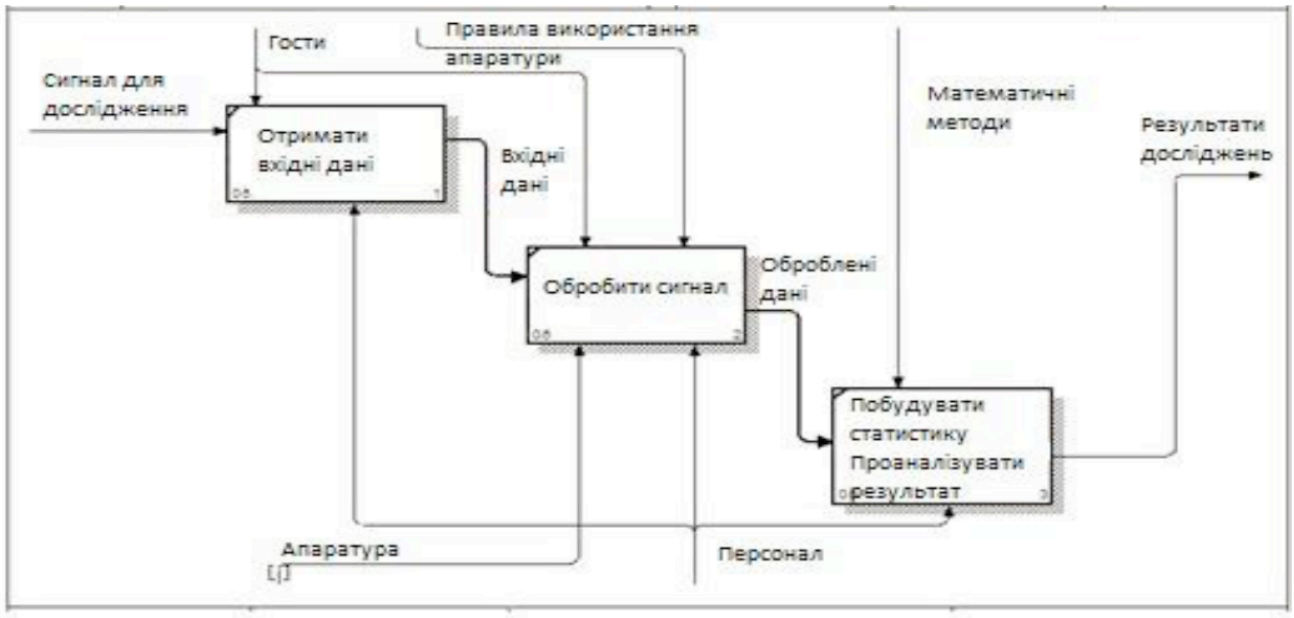

Рисунок 1.4 – Декомпозиція роботи «Аналіз сигналу»

Пункт «аналізу сигналу» с розділено на декілька задач:

– отримання вхідних даних;
– апаратна обробка сигналу;
– аналіз фінального результату;
– побудова статистики.

Відповідно до стандарту IDEF0 на кожному рівні декомпозиції повинен використовуватися принцип обмеження об'єкта, тому відповідно до цього принципу вважається, що єдність-ний блок і кілька стрілок на самому верхньому (контекстному) рівні використовуються для визначення кордону всієї системи. Відповідно, стрілки, що стосуються цього блоку, описують головні управління, входи, виходи і механізми цієї системи.

Деталізуючи розглянуту систему на етапі збору та аналізу попередньої інформації, необхідно звертати увагу на вхідні і вихідні об'єкти самої системи і складових її підсистем.

На рисунку 1.5 представлена декомпозиція роботи «Побудова статистики та аналіз результатів».

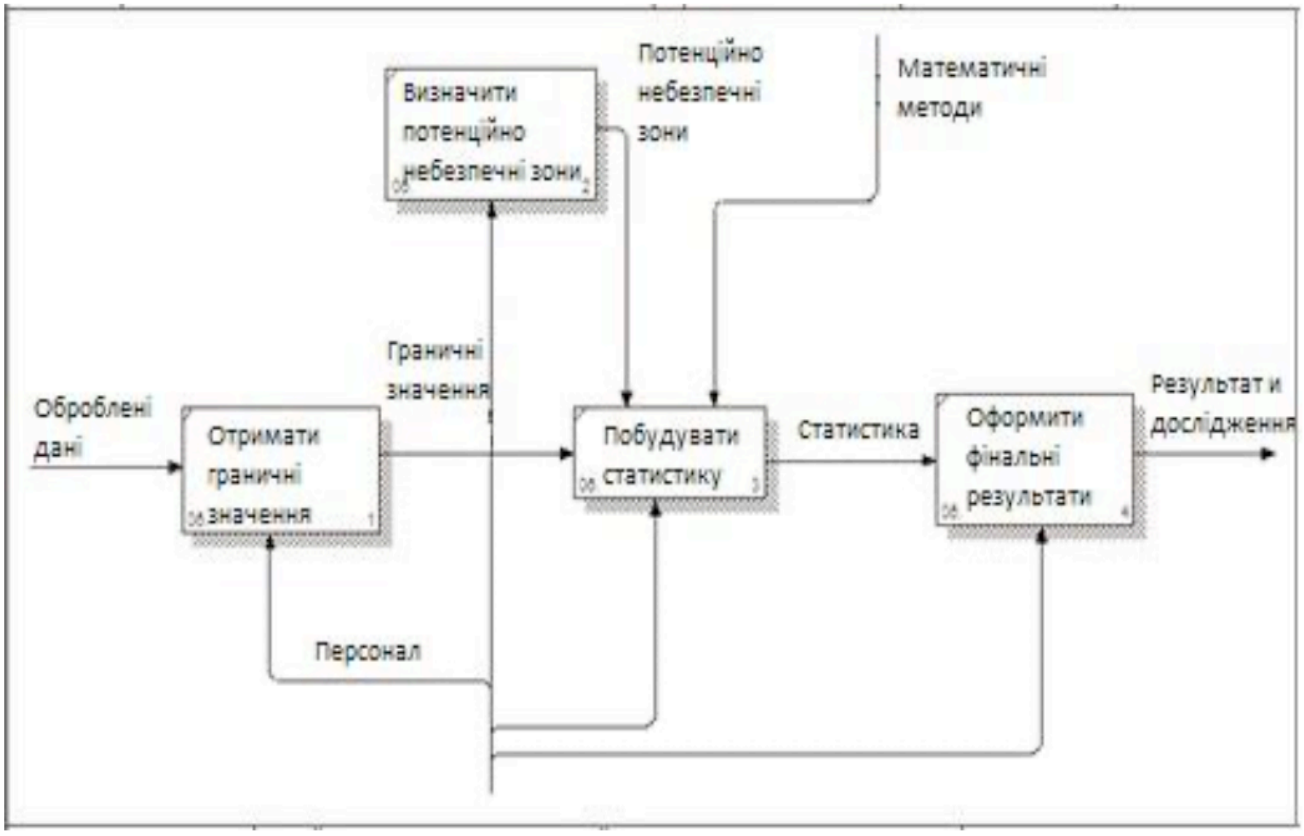

Рисунок 1.5 – Декомпозиція «Побудова статистики та аналіз результатів»

Основними підзадачами є:

– отримання граничних значень, які будуть використані для прогнозування атаки при перевищенні порогу запитів;
– знаходження потенційно затратних місць для розподілення обчислювальних затрат комп’ютера на аналіз сигналів;
– побудова статистики;
– оформлення фінальних результатів дослідження.

Базуючись на отримані результати можна зробити висновки щодо якості роботи методів за наступними параметрам: точність, енергозатрати, час відпрацювання тощо.

1.2.3 Інформаційна модель системи

Інформаційну модель системи відображено у вигляді data flow diagram (DFD).

DFD (діаграма потоків даних) – один з основних інструментів структурного аналізу та проектування інформаційних систем.

Вимоги подаються у вигляді ієрархії функціональних процесів, пов'язаних з потоками даних. Основне завдання DFD продемонструвати перетворення вхідних даних у вихідні дані для кожного процесу та вказати зв’язки між ними [4].

На рисунку 1.6 зображена інформаційна модель системи, яка показує потік даних від отриманого на вхід сигналу до результату дослідження.

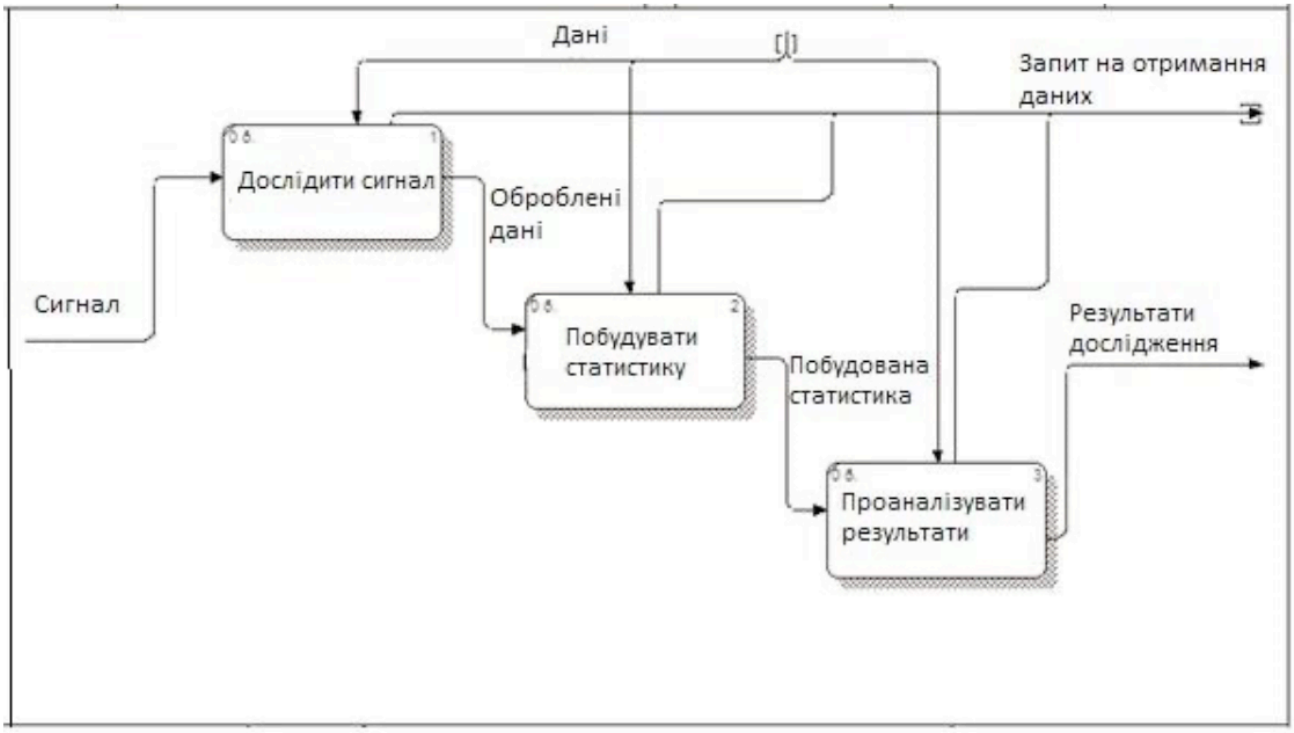

Рисунок 1.6 – data flow diagram

Вхідними даними (сигналами), над якими буде проводитися дослідження, є запити до сервера.

Дослідженням буде прогнозування можливих атак, та обрання найкращого методу для виявлення. Результатом дослідження будуть звіти, на основі яких можна зробити висновки щодо якості виявлення, швидкості роботи та об’єму витрачених ресурсів розглянутих методів.

Процес обробки статистики та аналіз результатів поділено на декілька під задач:

– опрацювання статистики, яка буде використана для прогнозування атаки при перевищенні порогу запитів;

– знаходження потенційно загрозливих місць;

– балансування затрат ЕОМ та дослідження сигналів.

1.3 Дослідження сценаріїв вирішення проблеми порівняльного аналізу методів прогнозування DDoS-атак

1.3.1 Модель аналізу проблеми

Предметом дослідження є процес виявлення DDoS-атак.

Мета дослідження – знайти найкращий за точністю та швидкістю метод отримання висновку про наявність атак на сервер [5].

Математичну модель слід вирішувати використовуючи різні алгоритми з подальшим їх порівнянням на точність результатів, складність програмної реалізації та ресурсозатратність. Спираючись на результати очікується виділення найбільш оптимального алгоритму, що може бути використаний для вирішення задачі ідентифікації атаки.

Виділимо параметри, які, на наш погляд, мають найбільший вплив на очікуваний результат. Такими параметрами є:

– швидкість (К1);

– універсальність (К2);

– надійність (К3);

– ресурсозатратність (К4).

Розглянемо вищенаведені параметри більш детально та уточнимо що саме буде порівняно у запропонованих альтернативах. В свою чергу алгоритми розпізнавання атак будуть обрані в якості альтернатив.

При порівнянні альтернативи в першу чергу нас цікавить можливість застосування алгоритмів з використанням великих об’ємів даних (вибірки та вхідні дані), які потребує алгоритм. Перевага буде віддана алгоритму який потребує менше інформації на вхід та використовує менший об’єм оперативної пам’яті.

Надійність алгоритму означає що в результаті роботи алгоритму буде отримано найбільш точний висновок щодо наявності атаки.

Буду розглянуто наступні рішення:

– використання дерев рішень;

– використання нейронних мереж;

– порівняння показників Херста.

Нейронні мережі використовують велику кількість ресурсів за рахунок внутрішньої структури, з іншої сторони нейронні мережі є досить простими у реалізації, так як працюють за рахунок повторення одноманітних операцій що дозволяє використовувати попередні результатів для економії часу. Результати попередніх ітерацій зберігаються в оперативній пам’яті тому вона потребує більше оперативної пам’яті для зберігання інформації. Незважаючи на досить значну ресурсозатратність, нейронні мережі є більш універсальними в порівнянні з деревами рішень та показниками Херста, а також показують точніший результат.

Метод дерев рішень є досить простим проте для вирішення задачі на предмет виявлення атаки потребує багато розгалуджень для збільшення точності результату, що значно збільшує час роботи алгоритму. За показником надійності він поступається ШНМ, проте є надійнішим ніж метод порівняння показників Херста. Метод дерев рішень використовують у випадках, коли результати одного рішення впливають на наступні рішення, тобто, для послідовного прийняття рішень.

Недоліком використання дерева рішень є досить велика ресурсозатратність та довгий час роботи.

До переваг можна віднести те, що за допомогою цього методу можливо оцінити різні шляхи і обрати найменш ризикований.

На рисунку 1.7 зображена ієрархічна модель процесу аналізу незадоволеностей.

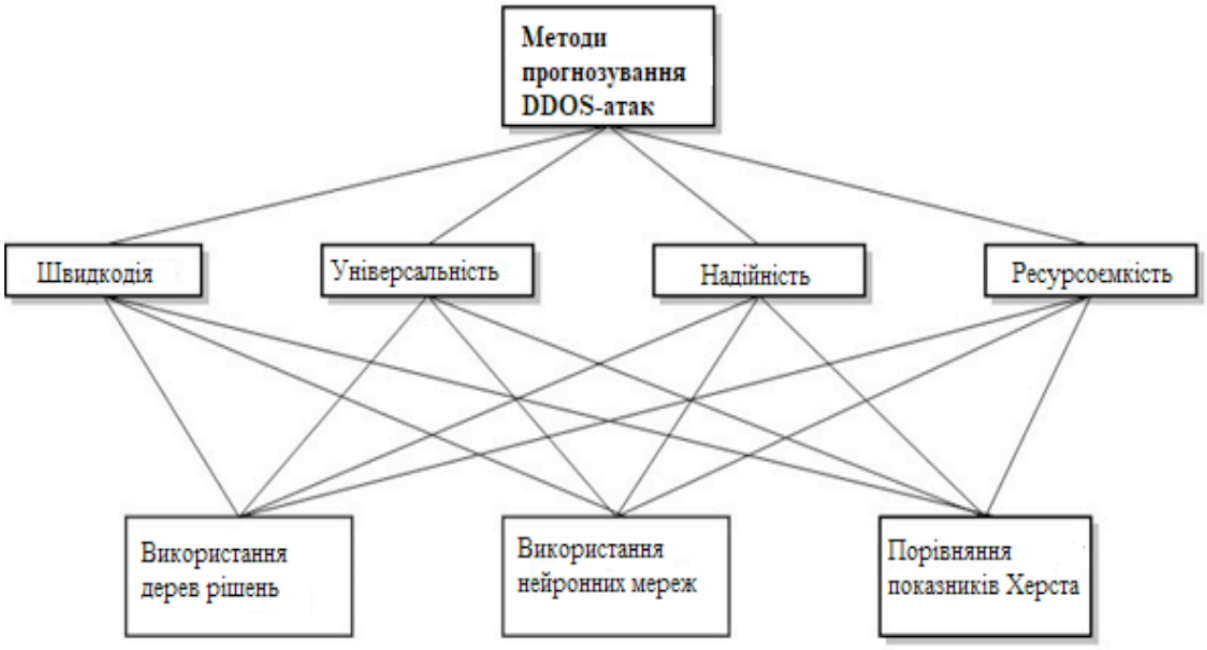

Рисунок 1.7 – Ієрархічна модель процесу аналізу незадоволеностей

Метод порівняння показників Херста є найменш ресурсомістким та найпростішим з боку реалізації в порівнянні з ШНМ та деревами рішень. Недоліком є найменша точність та ненадійність. За часом роботи алгоритму він прирівнюється до дерев рішень так як формула потребує більшу кількість арифметичних дій.

1.3.2 Оцінка вектора пріоритетів незадоволеностей за допомогою методу аналізу ієрархій

Використовуючи метод парних порівнянь, ми будуємо модель процесу аналізу незадоволеності. Процес аналізу незадоволеності складається з декількох рівнів.

На нульовому рівні відбувається аналіз проблеми. На рисунку 1.7 представлений нульовий рівень проблеми.

На першому рівні будується матриця попарних рівнянь параметрів для оцінки впливу кожної з незадоволеностей на поставлену проблему. Результати розрахунків наведені в таблиці 1.1.

Таблиця 1.1 – Результат матриці попарних порівнянь параметрів

	K1	K2	K3	K4	Вектор пріоритетів
K1	1,000	7,000	1,000	5,000	0,509
K2	0,143	1,000	3,000	0,200	0,113
K3	1,000	0,333	1,000	5,000	0,238
K4	0,200	5,000	0,200	1,000	0,140

Для знаходження матриці за стовбцями:

$$y_1 = 1{,}0+0{,}143+1{,}0+0{,}2=2{,}343,$$

$$y_2 = 7{,}0+1{,}0+0{,}333+5{,}0=13{,}333,$$

$$y_3 = 1{,}0+3{,}0+1{,}0+0{,}2=5{,}2,$$

$$y_4 = 5{,}0+0{,}2+5{,}0+1{,}0=11{,}2.$$

Тоді

$$\lambda_{max} \approx 2,343 * 0,509 + 13,333 * 0,113 + 5,2 * 0,238 + 11,2 * 0,140 = 5,505$$

та індекс узгодженості:

$$CI^k = \frac{2,505 - 4}{4 - 1} = 0,68$$

Оскільки матриця попарних порівнянь параметрів є матрицею четвертого порядку то отримуємо коефіцієнт узгодженості:

$$CR^k = \frac{CI^k}{0,9} = 0,187.$$

Оскільки коефіцієнт узгодженості наближується до 0,1,вважаємо, що матриця парних порівнянь параметрів построєна правильно.

Далі формуємо матриці парних альтернатив для кожного параметру, щоб порівняти методи з кожним параметром окремо.

Таблиця 1.2 – Матриця порівнянь за параметром К1

К1	А1	А2	А3	Вектор пріоритетів
А1	1,000	0,111	0,143	0,051
А2	9,000	1,000	5,000	0,722
А3	7,000	0,200	1,000	0,227

Щоб знайти індекс узгодженості, знаходимо суми елементів матриці за стовпцями:

$$Y_1 = 1,0+9,0+7,0=18,0,$$

$$y_2 = 0,111+1,0+0,2=1,311,$$

$$y_3 = 0,143+5,0+1,0=6,243.$$

Тоді

$$\lambda_{max} \approx 0{,}051 * 18{,}0 + 10{,}722 * 1{,}311 + 6{,}243 * 0{,}227 = 3{,}282$$

та індекс узгодженості:

$$CI_{K1}^{A} = \frac{3{,}282-3}{3-1} = 0{,}141.$$

Так як матриця парних порівнянь альтернатив є матрицею третього порядку, то коефіцієнт узгодженості:

$$CR_{K1}^{A} = \frac{CI^{k}}{0{,}58} = 0{,}241.$$

Таблиця 1.3 – Матриця порівнянь за критерієм К2

К2	А1	А2	А3	Вектор пріоритетів
А1	1.000	7.000	7.000	0.773
А2	0.143	1.000	2.000	0.139
А3	0.143	0.500	1.000	0.088

Щоб знайти індекс узгодженості, знаходимо суми елементів матриці за стовпцями:

$$y_1 = 1.0 + 0.143 + 0.143 = 1.286,$$

$$y_2 = 7.0 + 1.0 + 0.5 = 8.5,$$

$$y_3 = 7.0 + 2.0 + 1.0 = 10.0.$$

Тоді

$$\lambda_{max} \approx 0{,}773 * 1{,}286 + 0{,}139 * 8{,}5 + 0{,}088 * 10{,}0 = 3{,}056$$

та індекс узгодженості:

$$CI_{K2}^{A} = \frac{3.056 - 3}{3 - 1} = 0.028$$

Відношення узгодженості:

$$CR_{K2}^{A} = \frac{CI^{k}}{0{,}58} = 0{,}048.$$

Таблиця 1.4 – Матриця порівнянь за параметром К3

К3	А1	А2	А3	Вектор пріоритетів
А1	1,000	0,111	0,200	0,063
А2	9,000	1,000	3,000	0,672
А3	0,333	0,333	1,000	0,625

Щоб знайти індекс узгодженості, знаходимо суми елементів матриці за стовпцями:

$$y_1 = 1.0 + 9.0 + 0.333 = 10.333,$$

$$y_2 = 0.111 + 1.0 + 0.333 = 1.444,$$

$$y_3 = 0.2 + 3.0 + 1.0 = 4.2.$$

Тоді

$$\lambda_{\max} \approx 0{,}063 * 10{,}333 + 0{,}672 * 1{,}444 + 0.625 * 4{,}2 = 3{,}028$$

та індекс узгодженості:

$$CR_{K3}^{A} = \frac{CI^{k}}{0{,}58} = 0{,}024.$$

Таблиця 1.5 – Матриця порівнянь за параметром К4

К4	А1	А2	А3	Вектор пріоритетів
А1	1.000	0.200	0.333	0.105
А2	5.000	1.000	3.000	0.637
А3	3.000	0.333	1.000	0.258

Щоб знайти індекс узгодженості, знаходимо суми елементів матриці за стовпцями:

$$y_1 = 1.0+5.0+3.0=9.0,$$

$$y_2 = 0.2+1.0+0.333=1.533,$$

$$y_3 = 0.333+3.0+1.0=4.333.$$

Тоді

$$\lambda_{\max} \approx 0{,}105 * 9{,}0 + 0{,}637 * 1{,}533 + 0{,}258 * 4{,}333 = 3{,}039$$

та індекс узгодженості:

$$CI_{K4}^{A} = \frac{3{,}028 - 3}{3 - 1} = 0{,}020.$$

Відношення узгодженості:

$$CR_{K4}^{A} = \frac{CI^{k}}{0{,}58} = 0{,}034.$$

Розрахуємо вектор глобальних пріоритетів альтернатив. Для цього знаходимо добуток:

$$p = \begin{bmatrix} 0{,}051 & 0{,}773. & 0{,}063 & 0{,}105 \\ 0{,}722 & 0{,}139. & 0{,}672. & 0{,}637 \\ 0{,}227 & 0{,}088. & 0{,}256 & 0{,}580 \end{bmatrix} * \begin{bmatrix} 0{,}509 \\ 0{,}113 \\ 0{,}238 \\ 0{,}140 \end{bmatrix} = \begin{bmatrix} 0{,}143 \\ 0{,}632 \\ 0{,}225 \end{bmatrix}.$$

Обчислимо індекс узгодженості та коефіцієнт узгодженості для всієї ієрархії:

$$CI=0.168+0.509*0.141+0.113*0.048+0.238*0.024+0.140\cdot 0.020=0.254,$$

$$RI = 0.90 + 0.58 =1.48,$$

$$CR = \frac{CI}{RI} = \frac{0.254}{1.48} = 0.172,$$

що можна вважати як правильна узгодженість.

Найбільший компонент вектора локальних пріоритетів параметрів відповідає першому критерію. Маємо наступні пріоритети за параметрами порівняння: швидкість, надійність, ресурсозатратність та універсальність.

Порівнявши альтернативні рішення за обраними параметрами, ми отримали вектор глобальних пріоритетів. Розглянувши його можна зробити висновок що його найбільший компонент збігається з другою альтернативою, а саме використанню ШНМ.

1.4 Змістовна постановка задачі

Одним з найбільш перспективних та ефективних напрямків забезпечення безпеки мереж є використання засобів виявлення атак в основі яких

використовуються інтелектуальні технологій, а саме ШНМ. Нейронна мережа аналізує дані їй на вхід сигнали а на вихід дає оцінку, наскільки вхідні сигнали (дані) сходяться з розпізнаваними їй характеристиками. Сьогодні ШНМ дозволяють створювати високоточні систему для виявлення мережевих атак що забезпечує надійний рівень захисту. Аналізується реакція ШНМ, після чого система налаштовується для досягнення задовільних результатів. Оскільки нейронна мережа аналізує дані, вона набуває додаткового досвіду. Ця технологія є однозначною і може бути адаптована до нових типів атак.

Типовими завданнями, для вирішення яких використовують ШНМ є завдання класифікації, ідентифікації та розпізнавання зображень [7, 8]. Саме до такого класу завдань і відноситься розпізнавання та ідентифікація DDoS-атак. Нейронну мережу навчають за допомогою набору даних котрий містить правильні відповіді. Мережевий трафік можна розділити на нормальний та аномальний. Аномальний трафіком є мережеві атаки,які можна розпізнати та охарактеризувати використовуючи певний набір характеристик які не притаманні нормальному трафіку. В чергу свою аномальний трафік можна розділити ще на декілька класів в залежності від виду атаки.

Головною перевагою використання ШНМ для виявлення DDoS-атаки є гнучкість та адаптивність [9]. Здібність обробляти великі об’єми даних швидко є особливо важливим критерієм так як при веденні атаки проти мережі використовують велику кількість пристроїв.

1.5 Постановка задачі дослідження

Метою дослідження є розробка системи для аналізу даних запитів до сервера з метою виявлення DDoS-атаки. На основі цього цього можна сформулювати задачі для дослідження в рамках даної роботи:

– ознайомлення з алгоритмами машинного навчання та методами побудови та навчання ШНМ;

– зібрання тренувального та тестувального наборів даних для навчання ШНМ та її тестування відповідно;

– вибрати засоби для програмної реалізації;

– розроблення архітектуру програми, в якій будуть реалізовані обрані алгоритми ти проаналізовано отримані результати;

– проведення ряду експериментів змінюючи вхідні набори даних (розмір вибірки, кількість прихованих шарів та епох);

– спираючись на отриманні результати проаналізувати їх та зробити висновки про проведену роботу.

2 ВИБІР ТА ОБГРУНТУВАННЯ МЕТОДА РОЗВ'ЯЗАННЯ

2.1 Машинне навчання

Машинне навчання – це форма штучного інтелекту, яка використовує алгоритми, щоб система могла вчитися на основі даних, а не за допомогою явного програмування.

Розрізняють два типи навчання: індуктивне і дедуктивне. Індуктивне навчання, або навчання за прецедентами, засноване на виявленні загальних властивостей об'єктів на підставі неповної інформації, отриманої емпіричним шляхом. Дедуктивне навчання передбачає формалізацію знань експертів у вигляді баз знань (експертних систем тощо).

Зазначені методи машинного навчання базуються на використанні нейронних мереж, хоча існують також і інші методи, які використовують поняття навчальної вибірки – наприклад, дискримінантний аналіз, який оперує поняттями узагальненої дисперсії і коваріації, або байєсовські класифікатори [10].

Основні типи нейронних мереж, такі як перцептрон і багатошаровий перцептрон, мають можливість вчитися на принципах навчання з викладачем, без вчителя, навчання з підкріпленням та самоорганізацією. Однак деякі нейронні мережі можна віднести лише до одного із методів машинного навчання. Ось чому, якщо є необхідність класифікувати методи навчання з точки зору методу їх самонавчання, невірно класифікувати нейронні мережі до одного із певних типів, тому краще розділяти їх за методами навчання.

Далі наведено перелік класичних задач, що можуть бути вирішені використовуючи машинне навчання:

– класифікація: використовуються нейронні мережі з методом навчання з викладачем;

– кластеризація: у більшості випадків використовуючи метод навчання без викладача;

– регресія: використовуються мереже с методом навчання з викладачем але тільки продовж етапу тестування мережі;

– поновлення щільності розділу імовірностей для певного набору даних;

– зменшення розмірності даних, а також їх візуалізація вирішується методами навчання без викладача;

– побудова рангових залежностей;

– опис взаємозв'язку між групами об’єктів;

– часовий ряд або конкретний сигнал;

– зображення або послідовність відео-кадрів.

Однією з головних цілей машинного навчання є повна або часткова автоматизація розв’язання складних завдань для різних професій для облегшення людської праці.

З кожним днем сфера застосування методів машинного навчання стають все більше. Завдяки переходу до використання цифрової інформації накопичується величезна кількість даних у різних сферах життя. Завдяки цим процесам завдання управління, прогнозування та прийняття рішень часто призводять до навчання за допомогою прецеденту [11].

2.2 Типи машинного навчання

Існує багато методів машинного навчання, але як правило, вони належать до одного з трьох типів:

– навчання з викладачем або контрольоване навчання (supervised learning);

– навчання без викладача, або самонавчання (unsupervised learning);

– навчання з підкріпленням (reinforcement learning).

В залежності від завдання, деякі моделі можуть бути більш підходящими та ефективнішими, ніж інші.

В останні десятиліття машинне навчання безпрецедентно прогресувало в таких різноманітних сферах, як розпізнавання візерунків, робототехніка та складні ігри. Ці успіхи були досягнуті головним чином завдяки навчанню глибоких нейронних мереж. Усі види навчання потребують розробки сигналів навчання людини, які потім передаються на комп’ютер.

2.2.1 Навчання з викладачем

Контрольоване навчання, як правило, проводиться в контексті класифікації, коли ми хочемо відобразити введення міток виводу або регресії, коли ми хочемо відобразити введення на постійний вихід. Загальні алгоритми в контрольованому навчанні включають логістичну регресію, наївні заливи, підтримуючі векторні машини, штучні нейронні мережі та випадкові ліси. І в регресії, і в класифікації, мета полягає у пошуку конкретних взаємозв'язків або структури у вхідних даних, які дозволяють нам ефективно виробляти правильні вихідні дані. Варто зауважити, що «правильний» вихід визначається повністю з даних тренувань, тому, хоча у нас є основна істина, що наша модель вважатиме, що це правда, це не означає, що мітки даних завжди правильні в реальних ситуаціях. Шумні або неправильні мітки даних явно знизять ефективність вашої моделі.

При проведенні контрольованого навчання основними міркуваннями є складність моделі та компромісність відхилень.

Складність моделі означає складність функції. Належний рівень складності моделі, як правило, визначається характером навчальних даних. Якщо у нас є невеликий обсяг даних або якщо дані не рівномірно розповсюджуються в різних можливих сценаріях, слід обрати модель низької складності. Це пояснюється тим, що модель з високою складністю буде надмірною, якщо її використовувати в невеликій кількості точок даних. Під набором мається на увазі вивчення функції, яка дуже добре відповідає даним про навчання, але не узагальнює інші точки даних

- іншими словами, чітко навчається виробляти свої навчальні дані, не вивчаючи фактичної тенденції чи структури даних.

2.2.2 Навчання без викладача

Найпоширенішими завданнями в рамках некерованого навчання є кластеризація, представницьке навчання та оцінка щільності. У всіх цих випадках ми хочемо вивчити притаманну нам структуру даних без використання чітко передбачених міток. Деякі загальні алгоритми включають кластеризацію k-засобів, аналіз основних компонентів та автокодери. Немає конкретного способу порівняння продуктивності моделі у більшості непідконтрольних методів навчання.

Два поширені випадки використання для неконтрольованого навчання - дослідницький аналіз та зменшення розмірності.

Навчання без нагляду є дуже корисним у дослідницькому аналізі, оскільки може автоматично ідентифікувати структуру даних. Наприклад, якби аналітик намагався сегментувати споживачів, непомітний метод кластеризації був би чудовою відправною точкою для їх аналізу. У ситуаціях, коли людині неможливо або недоцільно запропонувати тенденції в даних, непідконтрольне навчання може дати початкові уявлення, які потім можуть бути використані для перевірки окремих гіпотез [12].

2.2.3 Навчання з підсиленням

Цей тип тренінгу - це суміш перших двох. Зазвичай використовується для вирішення більш складних проблем і вимагає взаємодії з навколишнім

середовищем. Алгоритм може реагувати на дані та вчитися. Підсилений тренінг застосовується, коли вам потрібно вибрати кращий варіант серед багатьох або досягти важкої мети за багато кроків. Алгоритми посилення, що включають поглиблені тренування, можуть перемогти чемпіонів світу в грі Go, починаючи з базового розуміння правил гри та тренувань від гри до гри.

Таким чином, це вже штучний інтелект у дії: машина намагається вирішити проблему різними способами, робить помилки, вчиться на своїх помилках, покращує продуктивність. Цей метод використовується в першу чергу там, де ви хочете навчити машину виживати в реальних умовах.

Цей метод навчання також часто використовується в логістиці, плануванні та тактичному плануванні завдань [13].

2.3 Огляд задач класифікації

Проблеми класифікації - це математичні задачі, які мають багато об’єктів, певним чином поділених на певні класи. Дано набір об'єктів, який є кінцевим і для якого ми знаємо класи, до яких належить кожен член цього набору. Цей набір називається зразком.

Приналежність інших об’єктів до певних класів невідома. Мета: побудувати алгоритм, який класифікує певний довільний об’єкт із заданого набору.

Класифікувати об'єкт – означає надати об’єкту ідентифікаційний номер (чи назву) класу, до якого цей об’єкт відноситься.

Завдання класифікації, у контексті машинного навчання, розв’язується за допомогою методів ШНМ, якщо ставити задачу з використанням методу навчання з викладачем.

Задачі класифікації розділяють на декілька класів:

– двухкласова класифікація (найпростіший випадок, який є основою і використовується для вирішення складних задач);

– багатокласова класифікація (коли кількість класів стає занадто великою, класифікація стає значно складнішою);

– пересічні класи;

– непересічні класи;

– нечіткі класи [14].

Дослідження показують, що найбільш точним методом розв’язання алгоритмів двокласної (бінарної) задачі класифікації з нечіткими класами є використання ШНМ. Під час роботи алгоритмів буде перевірено багато функцій, за якими можна побудувати прогноз атаки на сервер.

2.4 Нейронні мережі

Нейронна мережа - математична модель для вирішення проблеми класифікації та прогнозування. Далі ми розглянемо це лише як класифікатор. По суті, це графік з n вершин-входів і K вершин-виходів. Тут K - кількість класів у задачі класифікації. Вершини в цьому графіку називають нейронами. Нейрони з'єднані підвішеними ребрами. Значення нейрона визначається вагою ребер, що входять до нього, і величиною нейронів на протилежних кінцях цих ребер. Відповідно до n параметрів об'єкта, нейронна мережа дає K числа у відрізку [0,1], індекс максимального числа оголошується класом об’єкта [15].

Нейронна мережа прямого поширення (перцептрон) – це нейронна мережа розбита на шари, де всі ребра направлені в одну сторону. У такий нейронної мережі завжди є вхідний шар і вихідний шар і можливо кілька прихованих шарів. Розглянемо приклад нейронної мережі прямого поширення (рис. 2.1).

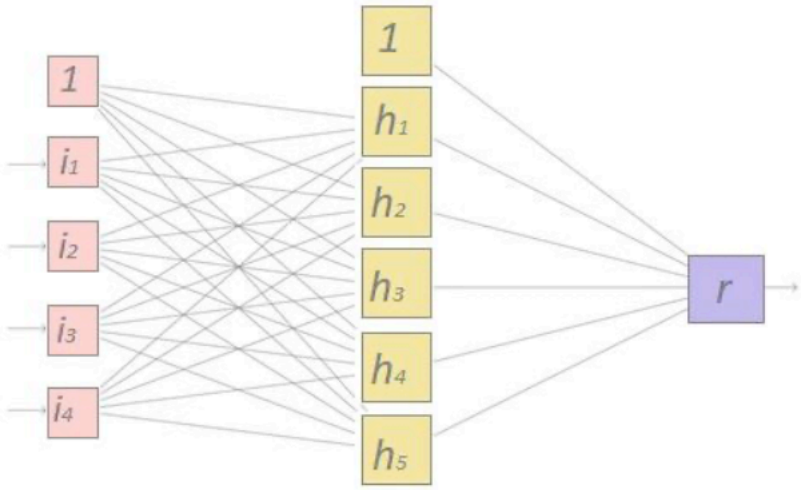

Рисунок 2.1 – Одношарова нейронна мережа прямого поширення

Нейрони вхідного шару позначені i_s , нейрони з прихованого шару h_s і r – нейрон вихідного шару. У кожному шарі, крім останнього додаються фіктивний нейрон зі значенням 1. Ваги ребер між шаром з n нейронами і з m нейронами представляються матрицею Θ розміру $m \times (n-1)$, тобто Θ_{ij} – вага ребем мій i-им і j-им нейронами в різних шарах, причому Θ_{j0} – ваги ребер з фіктивного нейрона. Кожен шар представимо як вектор υ з значень нейронів, позначимо за $\tilde{\upsilon}$ вектор υ розширений одиницею.

Позначимо відповідні вектори як i, h, r, а матриці відповідні ребрам між ними, як $\Theta^{i,h}$, $\Theta^{h,r}$. Тоді обчислення значення в шарах виконується наступним чином:

$$h = \sigma\left(\Theta^{i,h}\tilde{\imath}\right), r = \sigma(\Theta^{h,r}\tilde{h}) \,, \tag{2.1}$$

де m діє покоординатно і має вигляд:

$$\sigma(x) = \frac{1}{1+e^{-x}}. \tag{2.2}$$

Графік функції $\sigma(x)$ показаний на рис. 2.2:

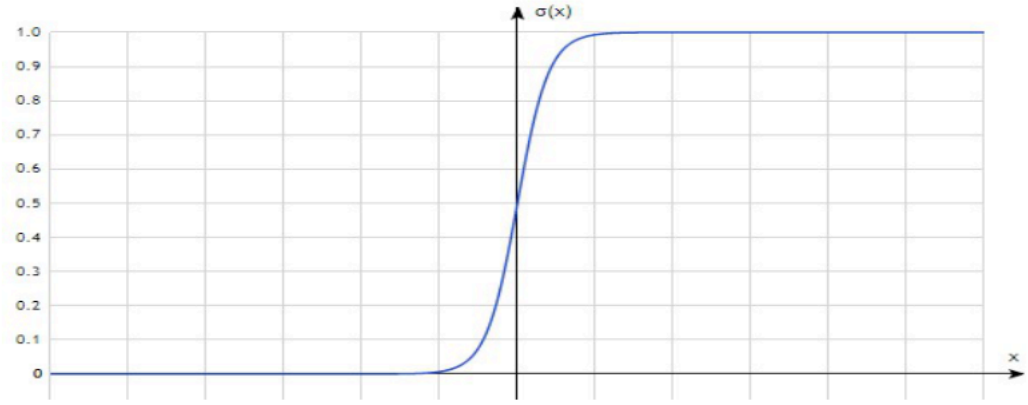

Рисунок 2.2 – Графік функції $\sigma(x)$

Послідовне обчислення значення нейронів у всіх шарах називається прямим проходженням. Аналогічні кроки виконуються для нейронної мережі з великою кількістю шарів, однак, згідно з узагальненою теорією наближення, яка говорить, що нейронна мережа прямого поширення одним прихованим шаром і кінцевою кількістю нейронів дозволяє обчислити значення довільна безперервна функція з будь-якою точністю. Але немає гарантії, що ви зможете легко знайти відповідну вагу країв такої нейронної мережі та кількість вершин у прихованому шарі. Знаходження матриць відповідних ваг між шарами називається процесом навчання нейронної мережі.

2.5 Штрафна функція

Розглянемо приклад використання штрафної функції над тренувальною вибірку з m об'єктів, клас яких вже відомий. Назвемо Θ вектор всіх ваг ребер ШНМ. Наприклад, знайдено довільну Θ, тому треба перевірити наскільки точні передбачення здійснює нейронна мережа з цими вагами. Для цього введемо штрафну функцію $J_i(\Theta)$, яка для i-го об’єкта з вибірки вважає наскільки збігається вгаданий результат з реальним. Нехай i-й об’єкт належить класу $c_i \in 0, \dots, K-1\}$,

тоді введемо такий бінарний вектор, що його c_i-я компонента дорівнюватиме одиниці, а всі інші рівні нулю.

Введемо функцію $h_\theta(x)$, яка видає результат роботи нейронної мережі (вектор з K чисел в інтервалі $[0,1]$) для n-мірного вектору (об'єкта) x.

Функція $J_i(\Theta)$ виглядає так:

$$J_i(\Theta) = \sum_{k=0}^{K} y^{(i)} - \log\left(h_\Theta\left(x^{(i)}\right)_k\right) - \left(1 - y^{(i)}\right)\log\left(1 - h_\Theta\left(x^{(i)}\right)_k\right). \qquad (2.3)$$

З формули (2.3) зроби висновок, що для вірних прогнозів (коли $h_\Theta\left(x^{(i)}\right) = y^{(i)}$) значення цієї функції буде рівне нулю та значення функції збільшуються при віддаленні передбачення від вірного. Тепер розглянемо штрафну функцію $J_i(\Theta)$, що описує якість прогнозування для всіх об'єктів одночасно:

$$J(\Theta) = \frac{1}{m}\sum_{i-1}^{m} J_i(\Theta). \qquad (2.4)$$

2.6 Проблеми перенавчання та недонавчання

Якщо з певної Θ-й нейронна мережа буде занадто точно передбачати результат для тренувальної вибірки, це матиме негативний вплив на якість прогнозів для об'єктів, які не належать до тренувальної вибірки, ця проблема називається проблемою перенавчання.

Протилежна ситуація виникає, коли нейронна мережа робить неточні прогнози для навчальної вибірки і значення функції $J(\Theta)$ велике, таке проблема має назву проблема недонавчання.

Розглянемо задачу бінарної класифікації на площині (рис. 2.3).

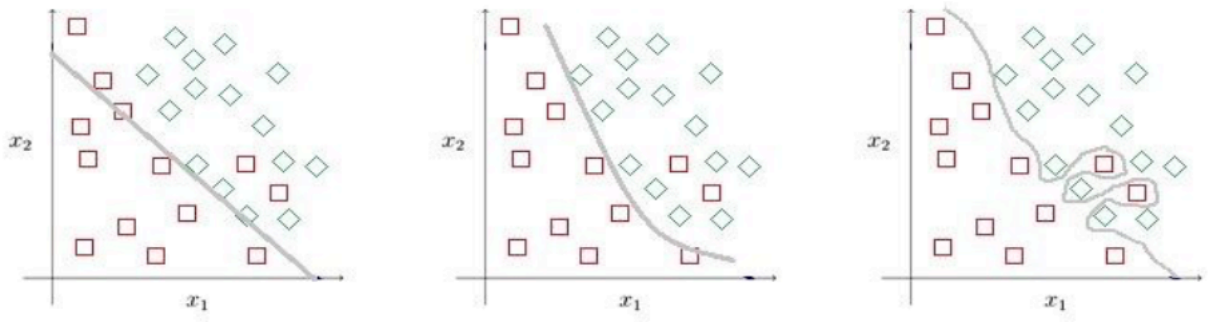

Рисунок 2.3 – а) недонавчання, б) задовільне навчання, в) перенавчання

На рис. 2.3 (а) об'єкти поділяються класифікатором занадто грубо – недонавчання. На рис. 2.3 (в) об'єкти з тестової вибірки поділяються занадто точно, що може привести до помилок прогнозів для нових об'єктів. Це пов’язано з кількістю нейронів в прихованому шарі. Якщо їх занадто велика кількість то трапляється перенавчання, а якщо мало – недонавчання. Ця проблема вирішується за рахунок модифікації функції $J(\Theta)$ в наступний спосіб:

$$J_{reg}(\Theta) = J(\Theta) + \frac{\lambda}{2m}\sum_{l=1}^{L-1}\sum_{i=1}^{s_l}\sum_{j=1}^{s_{l+1}}\Theta_{js}^{l^2}, \tag{2.5}$$

де s_i – кількість нейронів в шарі під номером i;

$\Theta(\mathrm{l})_{ji}$ – вага ребра між i-им нейроном l-го шару і j-им нейроном шару;

$\lambda > 0$ – називається параметром регуляризації: якщо він занадто великий то ШНМ стає упередженою і відбувається недонавчання, перенавчання трапляється якщо він занадто малий [16].

2.7 Вилучення ознак

В машинному навчанні вилучення особливостей починається з первинного набору даних вимірювань, і будує похідні значення (ознаки), інформативними та надмірними, щоб полегшити наступні кроки навчання та узагальнення, а в деяких

випадки для кращої людської інтерпретації. Вибір ознак пов'язаний зі зменшенням розмірності.

Якщо вхідний об’єм даних занадто великими та щоб їх можливо було обробити, і підозрюються на надлишковість (наприклад, одні й ті самі вимірювання як у метрах, так і в футах, або повторюваності в зображеннях, представлених пікселями), тоді їх об’єднати та узагальнити до скороченого набору даних. Цей процес має назву виділення або вилучення ознак. Очікується, що вибрані функції містять відповідну інформацію з вхідних даних, так що бажане завдання може бути виконане за допомогою цього скороченого подання ознак замість повних первинних даних.

Виділяння ознак включає зниження кількості ресурсів, необхідних для опису великого набору даних. Виконуючи аналіз складних даних, одна з основних проблем випливає з кількості залучених змінних. Аналіз з великою кількістю змінних у загальному випадку вимагає великої кількості пам'яті та обчислювальної потужності, або алгоритмів класифікації, які передаються на навчальний зразок і погано узагальнені до нових.

Виділення - це загальний термін для методів побудови таких комбінацій змінних для обходу цих проблем, зберігаючи достатню точність для опису даних.

Оскільки на практиці дані рідко надходять у вигляді готових до використання матриць, кожне завдання починається з обробки та виділення ознак. В нашому випадку достатньо прочитати дані з файлу та створити масив.

3 ПРОГРАМНА РЕАЛІЗАЦІЯ

3.1 Python як високорівнева мова програмування

Python досить простий і має невелику кількість ключових слів, в той же час дуже адаптований і гнучкий. Це більш високорівнева мова аніж Pascal, C ++ чиC, що в основному досягається внаслідок вбудованих у нього високорівневих алгоритмів та структур.

Найважливішою перевагою є те, що реалізація інтерпретатора мови Python є майже на всіх платформах.

Ще одна не менш важлива риса – це розширюваність. Python надає цьому дуже велике значення і ця мова, першочергово, була задумана як розширювана. Це значить, що вона має можливість вдосконалення внаслідок роботи усіма програмістами, що зацікавлені у цьому. Інтерпретатор Python запрограмований на мові C і його код у відкритому доступі для маніпуляцій. Якщо необхідно, його можна вставити як модуль у написану програму задля використання у вигляді вбудованої оболонки.

Python має досить багато модулів, та які можливо інтегрувати з програмою, та забезпечити різноманітні можливості. Ці додатки програмуються на C та власне на Python і можуть бути написаними будь-якими досвідченими розробниками. Стосовно сторонніх додатків виділяють інструменти для програмування веб-модулів, виконання математичних та фізичних розрахунків, створення програм у ігровій індустрії і багато іншого. Наведемо приклад додатку NumPy, який позиціонує себе як вдосконалений еквівалент мови програмування для математичних розрахунків Mathlab.

Так як Python досить проста та гнучка мова, то її можна порекомендувати не тільки програмістам, а й математикам, статистам, економістам, фізикам, і т.ін., які використовують програмування та обчислювальні додатки в своїй діяльності [17].

3.2 Багатопроцесорність у Python

Існує дві найпоширеніші причини використовувати потоки: по-перше, для підвищення ефективності використання багатоядерної архітектури сучасних процесорів, а отже, і продуктивності програми. Також, якщо нам потрібно розділити логіку роботи програми на паралельні повністю або частково асинхронні розділи (наприклад, мати можливість сповіщати кілька серверів одночасно).

Однак використання багатопоточності має декілька недоліків:

– кожен потік потребує місце для роботи з даними (навіть якщо він неактивний і пам’ять не вичерпується);

– якщо два потоки використовують один і той же дефіцитний ресурс, вони можуть конкурувати за ресурс.

Тут інший підхід може нам допомогти. У Python є багатопроцесорний модуль. Процеси можна створити із загальних функцій. Методи роботи з процесами майже всі ті ж, що і для потоків з модуля threading. Але для синхронізації процесів та обміну даними прийнято використовувати інші інструменти. Ми говоримо про черги та канали. Крім того, багатопроцесорний модуль має механізм роботи з спільною пам'яттю.

3.3 Опис програми

Програма виконана в середовищі PyCharm на високорівневій мові Python. Основне завдання программи – класифікувати отриманий запит на предмет ведення атаки за кількома параметрами. Тобто наша мета – отримати класифікатор, який за попереднім записом говорив би нам з деякою часткою ймовірності від кого прийшов запит: від бота або від людини.

Для тренування класифікатора нам знадобляться два набори даних. Перший набір матиме результати позитивних запитів, а другий – негативних. В результаті у нас повинні вийти три файли з даними для тренування і класифікації:

– positive_logs – зберігає позитивні запити до початку атаки;

– negative_logs – зберігає негативні запити від ботів під час атаки;

– access_logs – дані, які нам потрібно класифікувати.

Для того, щоб нейронна мережа «розуміла» по яким компонентам аналізувати отримані запити, потрібно привести дані в єдиний формат. Для контролю вхідних логів був написаний регулярний вираз, який буде приводити дані до бажаного формату.

Після розділення запиту на компоненти, необхідно виділити маркери, скласти їх у так званий словник і по ньому в подальшому складати вектори для кожного запису.

Для нашої програми було обрано такі маркери:

– сам запит, розділений на метод запиту, url і http_version;

– url, розділений на протокол, доменне ім’я, шлях та всі параметри query_string;

– referer, розділений аналогічно url в запиті;

– user-agent;

– response status-code, отриманий у відповідь на надісланий запит.

Після такого розділення отримуємо список всіляких маркерів, які можуть бути присутніми в запиті. Це буде словником ознак. Словник потрібен для того, щоб з будь-якого запиту створити feature-vector.

Бінарний, M-мірний вектор (де M – довжина словника), який відображає присутність кожної ознаки зі словника в запиті.

Для тренування нашої нейронної мережі необхідно розділити наші дані на тренувальні і тестові. В нашому випадку були використані пропорції 70/30. На тренувальній вибірці ми навчаємо нашу нейронну мережу, а на тестовій – перевіряємо, наскільки добре навчена наша нейронна мережа.

Сама нейронна мережа була побудована з одного прихованого шару розміром з подвоєний вхідний шар. Функцією активації для прихованого шару обрана сігмойда, а для вихідного шару – Softmax.

У роботі також використовується модуль multiprocessing мови програмування Python для пришвидшення роботи програми.

4 РЕЗУЛЬТАТИ ОБЧИСЛЮВАЛЬНОГО ЕКСПЕРИМЕНТУ

4.1 Підготовка даних

Більшість дослідників, які досліджували DDoS-атак, використовували загальнодоступні реальні зібрані дані, які називаються набором даних (dataset). Набір даних – це сукупність однотипних даних, що використовуються у завданнях з обробки даних.

У моєму випадку були взяті запити, отримані з сервера у форматі access.log. Такий формат подання запитів є найбільш поширеним та простим у використанні. На рисунку 4.1 та 4.2 зображені приклади нормального та аномального запитів:

```
0.0.0.0 - - [20/Jan/2020:15:00:03 +0400] "GET /forum/rss.php?topic=347425 HTTP/1.0" 200
1685 "-" "Mozilla/5.0 (Windows; U; Windows NT 5.1; pl; rv:1.9) Gecko/2008052906 Firefo
x/3.0"
```

Рисунок 4.1 – Приклад нормального запиту

```
0.0.0.0 - - [20/Jan/2020:20:00:08 +0400] "POST /forum/index.php HTTP/1.1" 503 107 "htt
p://www.mozilla-europe.org/" "-"
```

Рисунок 4.2 – Приклад аномального запиту на момент DDoS атаки

В результаті у нас повинні вийти 3 набора даних:

–dataset з "нормальними" запитами до початку ведення DdoS-атаки;

– dataset з "аномальними" запитами на момент ведення DDoS-атаки;

– dataset який нам необхідно проаналізувати.

Отримані тренувальні дані були розділені у відношенні 70/30 на дві частини:

– тренувальний dataset який використовується для навчання;

– тестовий dataset для перевірки якості навчання.

Такий розділ зумовлено тим, що ШНМ з найменшою помилкою, допущеною під час навчання, може показати велику помилку на реальних даних, із-за проблеми перенавчання (мережа буда заточена під тренувальний набір даних).

4.2 Відбір і виділення ознак

Для побудови словника ознак необхідно визначити, які параметри запиту впливають на рішення, до якого класу належить зразок. Необхідно сформувати словник ознак, який буде використовуватися ШНМ для розпізнавання запитів на наявність параметрів які притаманні DDoS-атакам.

При складанні такого словника очікується, що виділені ознаки містять доречну інформацію з вхідних даних, так що завдання розпізнавання та прогнозування DDoS може бути виконано із застосуванням цього скороченого представлення замість повних первинних даних. Для цього запит було розділено на такі логічні частини:

– запит, розділений на метод та url, що в свою чергу розділено на протокол, домен, шлях та всі параметри запиту;

– referer, розділений аналогічно url в запиті;

– user-agent;

– response status-code, відправлений у відповідь на надісланий запит. Приклад показано на рисунку 4.3.

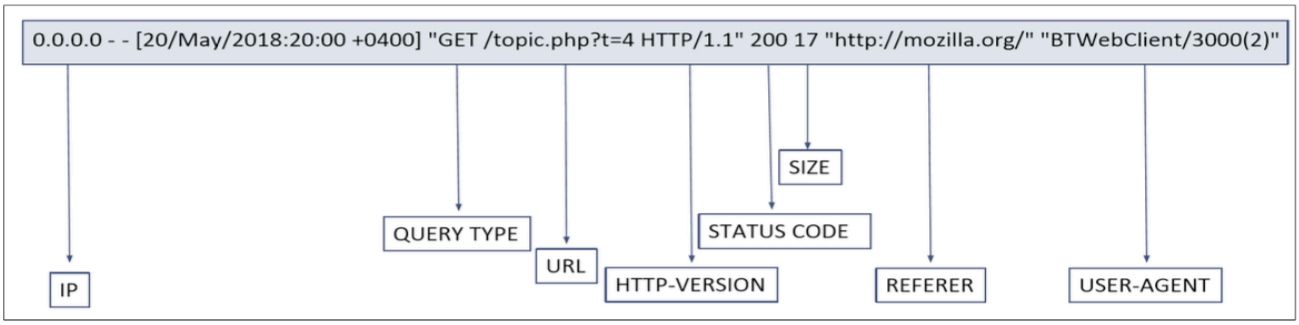

Рисунок 4.3 – Приклад класифікації запиту

Після такої класифікації маємо на руках список маркерів (ознак) які притаманні запиту відправленому задля проведення DDoS-атаки. Приклад такого словника зображений на рисунку 4.4.

```
['__UA___OS_U', '__UA_EMPTY', '__REQ___METHOD_POST', '__REQ___HTTP_VER_HTTP/1.0', '__RE
Q___URL___NETLOC_', '__REQ___URL___PATH_/forum/rss.php', '__REQ___URL___PATH_/forum/ind
ex.php', '__REQ___URL___SCHEME_', '__REQ___HTTP_VER_HTTP/1.1', '__UA___VER_Firefox/3.0'
, '__REFER___NETLOC_www.mozilla-europe.org', '__UA___OS_Windows', '__UA___BASE_Mozilla/
5.0', '__CODE_503', '__UA___OS_pl', '__REFER___PATH_/', '__REFER___SCHEME_http', '__NO_
REFER__', '__REQ___METHOD_GET', '__UA___OS_Windows NT 5.1', '__UA___OS_rv:1.9', '__REQ_
__URL___QS_topic', '__UA___VER_Gecko/2008052906']
```

Рисунок 4.4 – Приклад словника ознак

Кожний запит буде перевірятися на предмет наявність DDoS-ознак для формування feature-vector'a. Це *n*-вимірний вектор числових ознак, який представляє певний об'єкт (запит), що містить у собі інформацію про наявності заданих ознак у запиті. Приклад такого вектора для даного словника (рисунок 4.4) зображений на рисунку 4.5.

```
[False, False, False, False, True, False, False, True, True, False, False, False, False
, False, False, False, False, True, True, False, False, False, False]
```

Рисунок 4.5 – Приклад бінарного вектора для тестового запита

4.3 Проблема перенавчання

Перенавчання – це результат надмірної підгонки мережі до тренувальних прикладів. Щоб запобігти проблеми перенавчання використовують два тренувальні набори (тренувальний та тестовий). Навчальна частина - це мережевий

тренінг. Модель тестується на тестовому наборі даних. Набори повинні бути розрізненими.

Параметри побудованої моделі змінюються на кожному кроці, але значення цільової функції має зменшуватися у навчальній вибірці.

При розподілі множини на дві підмножини можемо побачити зміну помилки прогнозу на тестовій частині вибірки.

Кількість ітерацій значення помилки передбачення зменшується на навчальних наборах і тестові набори зменшуються одночасно. Однак з певного етапу похибка на тестовій частині починає збільшуватися, тоді як помилка на навчальному наборі продовжує зменшуватися. Ця точка вважається закінченням навчання, надалі починається процес перенавчання ШНМ.

Задля вирішення проблеми перенавчання мережі було введено так звані спроби. Розділення вибірки на тренувальну та тестову множини дає можливість спостерігати за процесом реального навчання нейронної мережі (рисунок 4.6). Введення спроб, а саме спроб незалежних запусків нейронної мережі, дає можливість вибору з них, що дає нам найменшу помилку тесту.

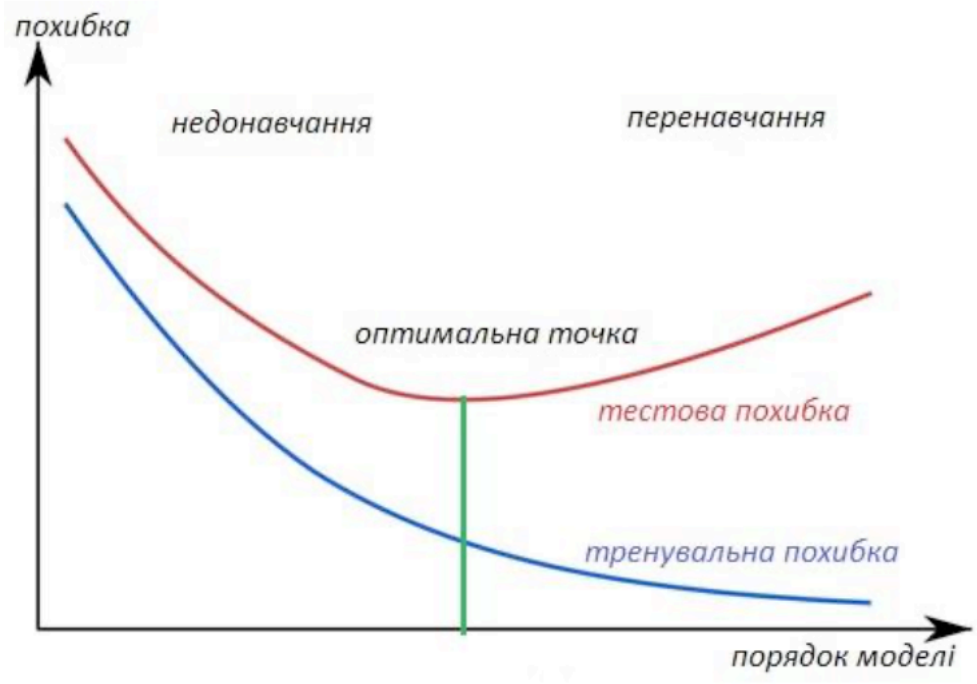

Рисунок 4.6 – Графік тренувальної та тестової похибок

Для вибору оптимальної кількості спроб запустити нейронну мережу були виміряні помилки тесту та час виконання алгоритму для різної кількості спроб. Точність обчислювали як відношення суми класифікованих даних порівняно з округлими даними нейронної мережі до загальної кількості правильних відповідей. Висновки дослідження можна побачити на рисунку 4.7.

К-сть спроб (шт.)	5	10	25	50
Тестова похибка (%)	17,21	10,64	9,12	7,73
Час роботи алгоритму (с)	1,72	2,52	7,83	13,02

Рисунок 4.7 – Залежність похибки та часу роботи від кількості спроб

Залежність показує, що після 10-ї спроби виникає значне збільшення часу роботи алгоритму без збільшення точності. Саме тому для даної ШНМ було зафіксовано 10 спроб.

Щоб пришвидшити алгоритми, його було розділено на процеси завдяки багатопроцесорному модулю – multiprocessing. Паралелізація проводилася шляхом поділу даних на окремі файли, так що окремий процес працював з його файлом. На рисунку 4.8 показана залежність часу роботи алгоритму та тестову похибку від кількості процесів (загальна довжина вибірки – 10000):

К-сть процесів (шт.)	1	2	3	5
Тестова похибка (%)	1,98	3,12	4,86	6,01
Час роботи алгоритму (с)	59,96	38,45	23,30	15,97

Рисунок 4.8 – Залежність похибки та часу роботи від кількості процесів

4.4 Результати досліджень

Після проведених досліджень та аналізу була побудовано нейронну мережа, яка дає можливість аналізу та класифікації запитів на сервер з метою виявлення наявності DDoS-атак. Вона складається з одного прихованого шару з подвоєний вхідниим шаром. Функцією активації для прихованого шару є сигмоїда, а для вихідного – Softmax.

Для досягнення необхідної точності було зафіксовано 10 епох та 10 спроб задля запобігання перенавчання. Точність обчислювали як відношення суми класифікованих даних порівнювана з округленими даними ШНМ до загальної кількості правильних відповідей. Також навчальна зразок було поділено на дві частини у відношенні 70/30 для вирішення цієї ж проблеми. Отримана мережа зображена на рисунку 4.9. На вхід нейронна мережа отримує бінарний вектор приналежності ознак, який був отриманий після аналізу запиту на предмет наявності в ньому ознак DDoS-атаки, використовуючи створений раніше словник ознак. Словник ознак було сформовано на етапі обробки даних шляхом аналізу компонентів запиту.

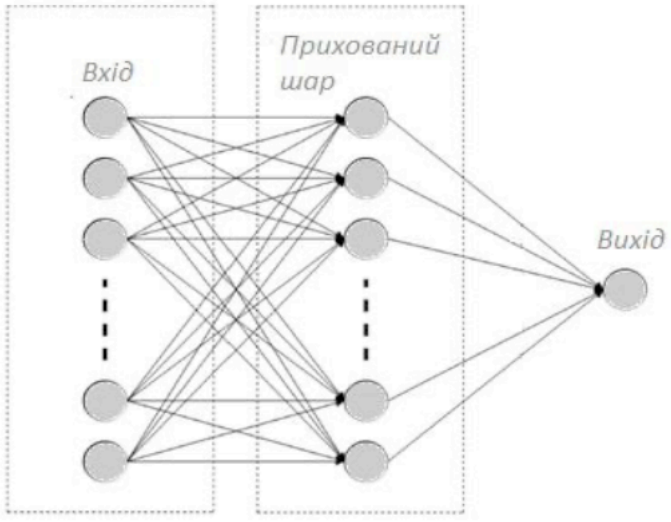

Рисунок 4.9 – Архітектура побудованої нейронної мережі

У отриманому класифікаторі було проаналізовано залежність помилок першого роду (хибне прогнозування класифікатором атаки) та помилок другого роду (хибне прогнозування класифікатором відсутності атаки) а також час відпрацювання алгоритму в залежності від довжини тренувальної вибірки. Результат дослідження зображено на рисунку 4.10.

Довжина вибірки	Помилка першого роду (%)	Помилка другого роду (%)	Час відпрацювання алгоритму (с)
10	10,89	11,47	1,32
25	10,02	10,42	2,68
50	9,65	10,12	4,96
100	8,11	9,25	8,78
200	6,34	8,81	17,01
500	4,77	5,33	33,43
1000	2,76	2, 98	59,96

Рисунок 4.10 – Дослідження результатів

Залежність помилки першого роду від довжини вибірки графічно зображено на рисунку 4.11.

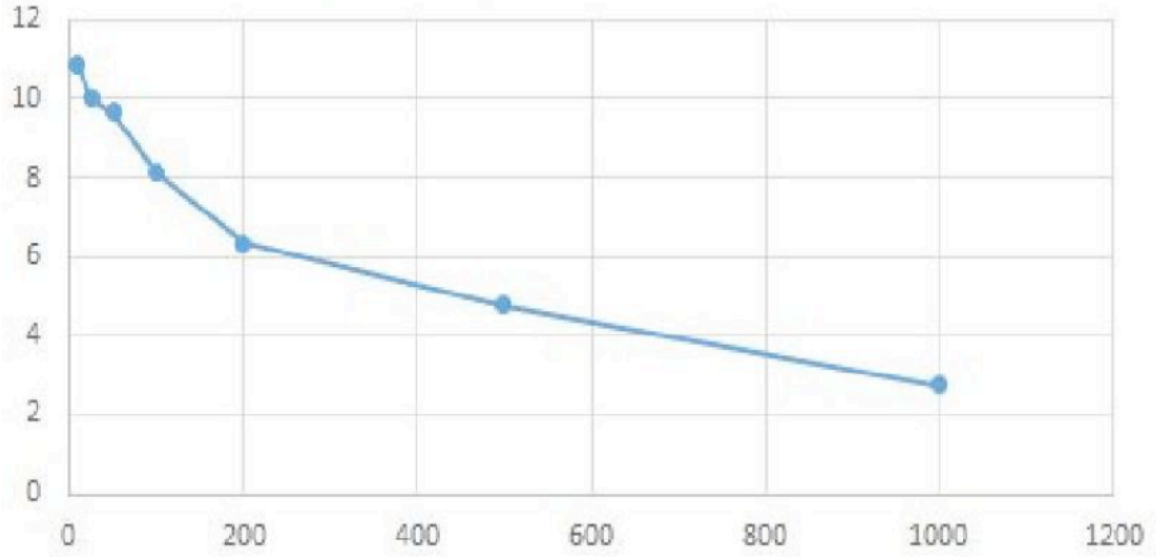

Рисунок 4.11 – Залежність помилки першого роду від довжини вибірки

Залежність помилки другого роду від довжини вибірки графічно відображено на рисунку 4.12.

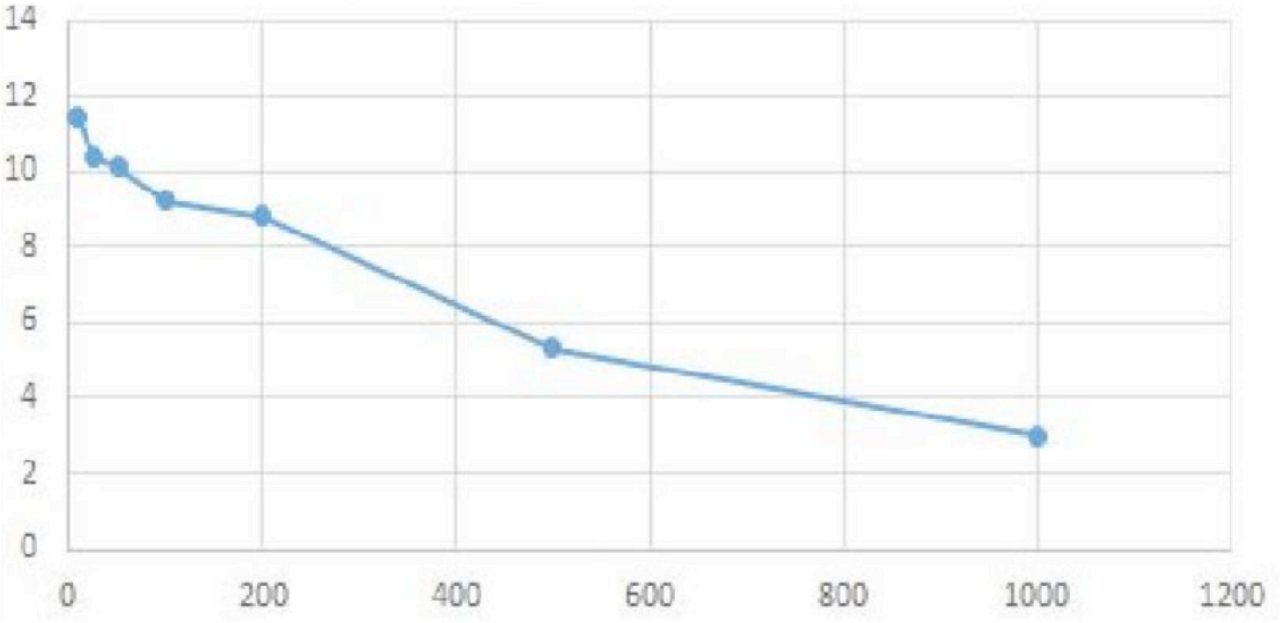

Рисунок 4.12 – Залежність помилки другого роду від довжини вибірки

Проте розширення довжини вибірки впливає на час роботи алгоритму. Графік залежності часі роботи алгоритму від розміру вибірки зображено на рисунку 4.13.

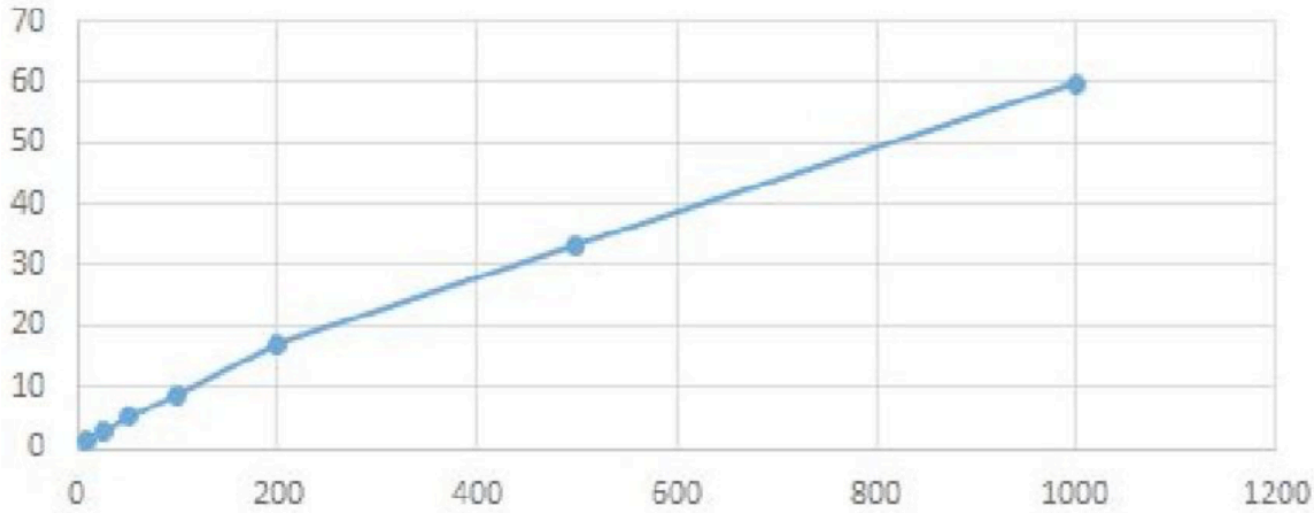

Рисунок 4.13 – Залежність часу відпрацювання алгоритму від довжини вибірки

4.5 Ефективність, новизна та практична цінність досліджень

Розроблений механізм прогнозування є швидким, точним та мало витратним. Він може використовуватися як автономно, так і як прошарок у існуючій системі фільтрації запитів на стороні сервера. Програмна реалізація досить проста у розумінні та може бути налаштована під різні типи користувацьких запитів.

Запропоновано архітектуру штучної нейронної мережі, навчений екземпляр якої дозволяє вирішувати задачу виявлення деяких класів розподілених мережевих атак типу DDoS на віддалений веб-ресурс, яка відрізняється сукупністю оптимально підібраних параметрів, що дає змогу підвищити точність виявлення.

5 АНАЛІЗ МОЖЛИВИХ ЗАСТОСУВАНЬ

В даний час значно збільшилася кількість DDOS-атака середньої і низької інтенсивності, спрямованих, як правило, на веб ресурси. Це збільшення є цілком прогнозованим оскільки з розвитком мереж збільшується потенційна кількість можливих жертв. Крім того, вдосконалюється сам механізм проведення атак. Для зловмисника проведення атаки вже не є настільки складним. А зомбі-комп'ютери намагаються емітувати дії самих користувачів. Все це веде до загального збільшення числа атак.

DDoS-атаки є робочим інструментом незаконної боротьби з конкурентами та ведення кібер-війн. Жертвами стає не тільки середній і малий бізнес.

Аналіз аналогів показує, що в даний час більшого розвитку отримала група засобів протидії, призначена для відображення потужних атак. У цю групу входять, як правило, дорогі засоби, призначені для великих провайдерів або компаній. Засоби протидії невеликим і середнім атакам, що базуються на сервері, представлені вкрай бідно. Це пов'язано з незначною кількістю таких атак в минулому. При цьому аналіз вхідного трафіку на рівні додатків може бути більш ефективним. З одного боку, проведення такого аналізу економічно затратно, з іншого – може бути цілком достатнім для відображення малих і середніх атак, тенденція переважання яких вже намітилася.

ВИСНОВКИ

В ході виконання даної атестаційної роботи досліджено методи побудови нейронних мереж для виявлення DDoS-атак на основі аналізу запитів отриманих з веб сервера. Проаналізовано наукову літературу пов’язаної зі штучними нейронними мережами та машинним навчанням.

Проведено системний аналіз предметної області з метою обрання методів аналізу запитів на предмет ведення атаки. Стандартні методи аналізу запитів не дозволяють розпізнати атаку, тому в якості механізму вирішення даної проблеми було використано нейронні мережі.

Було зібрано декілька наборів даних для навчання мережі та її тестування. Розроблено ряд алгоритмів для обробку вхідних даних, а саме розділення запиту на частини та їх аналізу з виділенням ознак, притаманних певним компонентам, та формування словника ознак, який використовувався ШНМ для подальшої класифікації.

Розроблено механізм, що дозволяє класифікувати дані запитів до сервера на предмет приналежності DDoS-атаки. Проведено ряд експериментів для знаходження найбільш оптимальних налаштувань ШНМ.

На основі реалізованого прототипу програмного забезпечення було проведено ряд експериментів, змінюючи обсяг вибірки, та проаналізовано зміни помилок першого та другого роду а також час роботи алгоритму з елементами паралелізації між процесами.

Отримане програмне забезпечення можна інтегрувати у будь-яку систему, пов’язану з Інтернетом, для виконання фільтрації запитів. Даючи досить точні результати за короткий проміжок часу система придатна для використання як автономна модель класифікації даних, так і у більш складних системах, в якості їх компонента.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. Shin D. How to defend against amplified reflection ddos attacks. URL: https://www.a10networks.com /resources/articles/how-defend-against-amplified-reflection-ddos-attacks (дата звернення 19.01.2020).
2. Сетевые атаки: обзор, классификация. URL: http://oitzi.ru/Materials.aspx?doc_id=5&id=77 (дата звернення: 08.12.2019).
3. Краткий обзор алгоритмов интеллектуального анализа данных. URL : http://www.intuit.ru/studies/courses/2312/612/lecture/13270 (дата звернення: 17.01.2020).
4. Жданов С. А., Соболева М. Л., Алфимова А. С. Информационные системы. Москва: Прометей, 2015. 302 с.
5. Машинное обучение. URL : http://www.machinelearning.ru (дата звернення: 01.03.2020).
6. Kirichenko L., Radivilova T., Carlsson A. Detecting cyber threats through social network analysis: short survey // SocioEconomic Challenges. 2017. No 1. P. 20–34.
7. Штучні нейронні мережі та їх класифікація. URL : https://evergreens. com.ua/ua/articles/neural-network.html (дата звернення: 01.03.2020).
8. Методы машинного обучения для классификации и прогнозирования. URL : http://www.intuit.ru/studies/courses/6/6/lecture/182 (дата звернення: 01.03.2020).
9. Machine Learning. URL : https://www.it.ua/knowledge-base/technology-innovation/machine-learning (дата звернення: 21.03.2020).
10. Types of Machine Learning Algorithms You Should Know. URL : https://towardssdatasciencee.com/types-of-machine-learning-algorithms-you-should- know-953a08248861 (дата звернення: 17.02.20).

11. Введение в Reinforcement Learning, обучение с подкреплением. URL: https://datascience.org.ua/articles/vvedenie-v-reinforcement-learning-ili-obuchenie-s- podkrepleniem/ (дата звернення: 22.02.2020).
12. Класификация предложений с помощью нейронных сетей. URL : https://habr.com/ru/company/meanotek/blog/256593/ (дата звернення: 25.02.2020).
13. Кириченко Л. О., Радивилова Т. А., Барановский А. А. Обнаружение киберугроз с помощью анализа социальных сетей. 2017. No 1. С. 23–48.
14. Краткий обзор объектно ориентированного языка Python. URL : https://www.helloworld.ru/texts/comp/lang/python (дата звернення: 22.01.2020).
15. Bondarenko M.F. Multiple-valued structutes of Intellectual Systems / Bondarenko M.F., Chetverikov G. G., Vechirskaya I.D. // International Book Series “Information Science and Computing”. – 2008. – Vol. 2, Number 5 – P.125-130.
16. Bondarenko M.F. Multiple-valued structutes of Intellectual Systems / Bondarenko M.F., Chetverikov G. G., Vechirskaya I.D. // International Book Series “Information Science and Computing”. – 2008. – Vol. 2, Number 5 – P.125-130.
17. Bondarenko M.F., Konoplyanko Z..D., Chetverikov G.G. Theory fundamentals of multipli-valued structures and coding in artificial intelligence systems.iKharkiv: Factor-druk. 2003.– 336 p.
18. Bondarenko M., Konoplyanko Z., Chetverikov G. Analiz problemy sozdaniya novich technicheskich sredstv dlya realizataii lingvisticheskogo interfeisan // Proc.of the 10th International Conference KDS–2003, Varna, Bulgaria, – 2003. – P. 3–15.
19. Четвериков Г.Г. Концепція уніфікації інформаційно-інтелектуальних технологій в систем мовлення [Текст] / Г.Г. Четвериков, М.Ф.Бондаренко, З.Д.Коноплянко // Научно-технический журнал «Бионика интеллекта». – 2011.– №3(77).– С. 156–162