Міністерство освіти і науки України
Харківський національний університет радіоелектроніки

Факультет інформаційно-аналітичних технологій та менеджменту
(повна назва)

Кафедра прикладної математики
(повна назва)

# АТЕСТАЦІЙНА РОБОТА
# Пояснювальна записка

рівень вищої освіти другий (магістерський)

Методи виявлення DDoS-атак, засновані на застосуванні машинного навчання
(тема)

Виконала:
студентка _2_ курсу, групи _САУм-18-1 _____
Безіна К.Ю.
(прізвище, ініціали)
Спеціальність _____
124 Системний аналіз
(код і повна назва спеціальності)
Тип програми _освітньо-професійна_____
(освітньо-професійна або освітньо-наукова)
Освітня програма _____
Системний аналіз і управління
(повна назва освітньої програми)
Керівник ____проф. Кіріченко Л.О. _____
(посада, прізвище, ініціали)

Допускається до захисту

Зав. кафедри ПМ _____ ____Тевяшев А.Д._____
(підпис) (прізвище, ініціали)

2019р.

Харківський національний університет радіоелектроніки

Факультет _інформаційно-аналітичних технологій та менеджменту
Кафедра_прикладної математики
Рівень вищої освіти _другий (магістерський)
Спеціальність _124Системний аналіз
(код і повна назва)
Тип програми _освітньо-професійна
(освітньо-професійна або освітньо-наукова)
Освітня програма _Системний аналіз і управління
(повна назва)

ЗАТВЕРДЖУЮ:

Зав. кафедри ПМ _____
(підпис)
"____" _____ 2019 р.

# ЗАВДАННЯ
НА АТЕСТАЦІЙНУ РОБОТУ

студентці Безіній Катерині Юріївні
(прізвище, ім'я, по батькові)

1. Тема роботи Методи виявлення DDoS-атак, засновані на застосуванні машинного навчання

затверджена наказом по університету від 31жовтня 2019 р. № 1601 Ст

2. Термін подання студентом роботи до екзаменаційної комісії 9грудня 2019 р.

3. Вихідні дані до роботи набір запитів, отриманих з сервера
порівняльний аналіз методів виявлення DDoS-атак

4. Перелік питань, що потрібно опрацювати в роботі
1. Системний аналіз проблеми порівняння методів виявлення DDoS-атак
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 (слайдів)

1. Актуальність теми роботи
2. Постановка задачі
3. Системний аналіз проблеми
4. Побудова нейронної мережі
5. Принцип роботи алгоритму
5. Результати обчислювального експерименту

## КАЛЕНДАРНИЙ ПЛАН

| № | Назва етапів роботи | Терміни виконання етапів роботи | Примітка |
|---|---|---|---|
| 1 | Підбір та вивчення технічної літератури за темою роботи | вересень 2019 р. | виконано |
| 2 | Вибір та обґрунтування методу | жовтень – листопад 2019 р. | виконано |
| 3 | Розробка алгоритму і програми | листопад – грудень 2019 р. | виконано |
| 4 | Проведення аналітичних досліджень та розрахунків | листопад – грудень 2019 р. | виконано |
| 5 | Робота над текстом пояснювальної записки | грудень 2019 р. | виконано |
| 6 | Представлення роботи на рецензію в ЕК | грудень 2019 р. | виконано |

Дата видачі завдання _2_ вересня 2019 р.

Студент _____
(підпис)

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

# РЕФЕРАТ

Пояснювальна записка: 63 с., 20 рис., 8 табл., 1додаток, 20 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.

# ABSTRACT

Introductory note: 63 pages, 20 pictures, 8 tables, 1 appendix, 20 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

Ще декілька десятків років тому комп'ютерні системи були однокористувацькими і обмінювалися даними між кількома, досить обмеженими каналами.

Новий принцип побудови мережі на основі взаємообміну пакетів дозволили значно підвищити рівень гнучкості і живучості систем. У наш час будь-яка сфера діяльності, пов'язана з обміном даних, не обходиться без використання комп'ютерних мереж. Охоплення та пропускна здатність найбільшої мережі – Інтернет зростає щоденно. Це сприяє створенню сайтів, які розраховані на багато користувачів, для роботи та зв’язку у всьому світі. Системи такої будови широко використовуються у банківських справах, страхуванні, кредитуванні, охороні здоров'я, прав, військових програмах, зв'язку та багатьох інших.

Проблема DDoS/DoS-атак набуває все більшої актуальності. Про них пишуть в Інтернеті і говорять на телебаченні. Побачити результат таких дій може кожен, відкривши якийсь сайт новин або стрічку в соціальній мережі, а замість очікуваної інформації побачить повідомлення про недоступність сервісу.

В сучасному світі все більшого значення набувають автоматизовані системи, які управляють різними критичними процесами. 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. Адже своєчасне виявлення DDoS-атаки дозволить зберегти працездатність мережі, так як якщо не зупинити у провайдера трафік, призначений для переповнення мережі, що атакується, то зробити це на вході виявиться неможливим, оскільки вся смуга пропускання буде вже зайнята. Стандартні методи аналізу

статистики не дозволяють виявляти невідомі раніше атаки, тому в якості механізму вирішення даної проблеми активно виступають нейронні мережі.

Штучна нейронна мережа – це математична модель, а також її програмна або апаратна реалізація, побудована за принципом організації та функціонування біологічних нейронних мереж – мереж нервових клітин живого організму. ШНМ використовуються практично у всіх засобах виявлення вторгнень як окремо, так і в комплексі з іншими механізмами захисту.

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.

# 1 СИСТЕМНИЙ АНАЛІЗ ПРОБЛЕМИ ПОРІВНЯННЯ МЕТОДІВ ВИЯВЛЕННЯ DDOS–АТАК ТА ПОСТАНОВКА ЗАДАЧІ ДОСЛІДЖЕННЯ

## 1.1 Системний аналіз проблеми порівняльного аналізу методів виявлення DDoS-атак

### 1.1.1 Вербальна модель системи

Об'єктом дослідження є проблема порівняння методів виявлення DDoS-атак. Метою є прогнозування атаки на сервер різними методами і подальше порівняння цих методів за такими критеріями як: оцінка якості прогнозування, особливості програмної реалізації, складність алгоритму, можливість застосування методу для досить великих вибірок [1].

DDoS-атака – напад на систему чи ресурси з наміром зробити їх недоступними для користувачів, задля яких комп'ютерна система була розроблена.

Завдання методів виявлення DDoS-атак полягає в тому, щоб аналізуючи дані, отримані з серверу, можна зробити найбільш точний прогноз щодо того, атакують сервер на даному проміжку часу, чи ні [2].

Подальше завдання даної роботи полягає в порівняльному аналізі методів, заснованих на оцінці якості отриманих прогнозів та оцінці продуктивності реалізованої програми.

Проте для того, щоб аналізувати методи роботи з даними, отриманими з серверу, необхідно розуміти процес аналізу сигналу, та проблеми, з якими ми можемо зустрітися. Для цього будемо будувати моделі аналізу сигналів, та описувати їх.

### 1.1.2 Морфологічний опис системи

Зовнішнє середовище – це фактори і умови, які, незалежно від діяльності підприємства, існують в навколишньому середовищі. Вони можуть як впливати на функціонування організації, так і відчувати на собі вплив організації.

Всі ресурси, необхідні для функціонування, організація отримує із зовнішнього середовища, в наслідок постійного обміну з ним. Для опису функціонування моделі із середовищем на рисунку 1.1 представлена модель «чорний ящик».

Система, що представлена як «чорний ящик», має певний «вхід», у який поступає інформація, і «вихід» для показу результатів. Зовнішньо на неї впливають управління, а також необхідно враховувати механізм. Процеси, що відбуваються в ході роботи системи, досліднику невідомі, але стан входів функціонально впливає на стан виходів [3].


Рисунок 1.1 – Модель типу «чорний ящик»

### 1.1.3 Функціональна модель системи

Для чіткого формулювання логіки та взаємодії процесів використовується мова моделювання бізнес-процесів IDEF0. Як було вказано вище, ми будуємо процес аналізу вхідного сигналу з сервера.

Процес конструювання будь-якої системи в IDEF0 розпочинається з визначення контексту, що є найбільш абстрактним рівнем опису системи в цілому [4].

Контекст складається з визначення суб’єкта моделювання, точки зору на модель і цілі. Для даної задачі:

– точка зору – дослідник;

– суб’єкт – сигнал, отриманий з сервера;

– ціль – дослідження сигналу на якість прогнозування атак.

Контекстна діаграма IDEF0 зображує функціонування системи в цілому (рисунок 1.2). Основною задачею являється аналіз сигналу з метою прогнозування атак, яка виконується дослідником за допомогою ЕОМ. Інформаційним ресурсом є математичні методи, які застосовуються до вхідних даних. В результаті виконання роботи очікується отримання аналізу сигналу.


Рисунок 1.2 – Контекстна діаграма IDEF0

Щоб розглянути функціональну частину більш детально, виконується декомпозиція системи. Декомпозиція роботи «Дослідження сигналу» зображена на рисунку 1.3. Даний процес розділено на три задачі:

– отримання вхідних даних;

– програмна обробка сигналу;

– побудова статистики, аналіз фінального результату.


Рисунок 1.3 – Декомпозиція роботи «Аналіз сигналу»

Далі більш детально розглянемо процес побудови статистики та фінального аналізу результатів. На рисунку 1.4 представлена декомпозиція роботи «Побудова статистики та аналіз результатів». Основними підзадачами є:

– отримання порогових значень, які можуть буди використані для прогнозування атаки по перевищенню порогу запитів;

– вирахування потенційно небезпечних місць, для розподілення затрат ЕОМ на аналіз сигналів;

– побудова статистики, яка може бути використана для візуалізації результатів;

– оформлення фінальних звітів для передачі потенційному замовнику та для прийняття подальших дій.

В результаті цієї задачі будуть отримані прогнози, по яким можна буде зробити висновки щодо якості роботи методів, використовуючи такі параметри як точність, енергозатрати, час відпрацювання, що будуть вказані у фінальних звітах.


Рисунок 1.4 – Декомпозиція «Побудова статистики та аналіз результатів»

### 1.1.4 Інформаційна модель системи

Інформаційна модель системи представлена у вигляді DFD діаграм.

DFD-модель (діаграма) – (Data Flow Diagrams) – діаграма потоків даних – є засобом для моделювання функціональних вимог до проектованої системи.

Вимоги представляють у вигляді ієрархії функціональних процесів, пов'язаних потоками даних. Основне завдання DFD-моделі (діаграми) показати перетворення вхідних даних у вихідні кожного процесу і позначити відносини між процесами [5].

На рисунку 1.5 представлена інформаційна модель системи, яка зображує потік даних між отриманим сигналом та фінальним звітом. Вхідні дані, над якими будуть проводитися дослідження – сигнали, отримані з сервера.

Дослідження, які будуть проводитися – прогнозування можливих атак, та обрання найкращого метода для цього. На виході буде отримано фінальні звіти на основі яких можна зробити висновки щодо якості, швидкості на ресурсо-витрат певних алгоритмів.

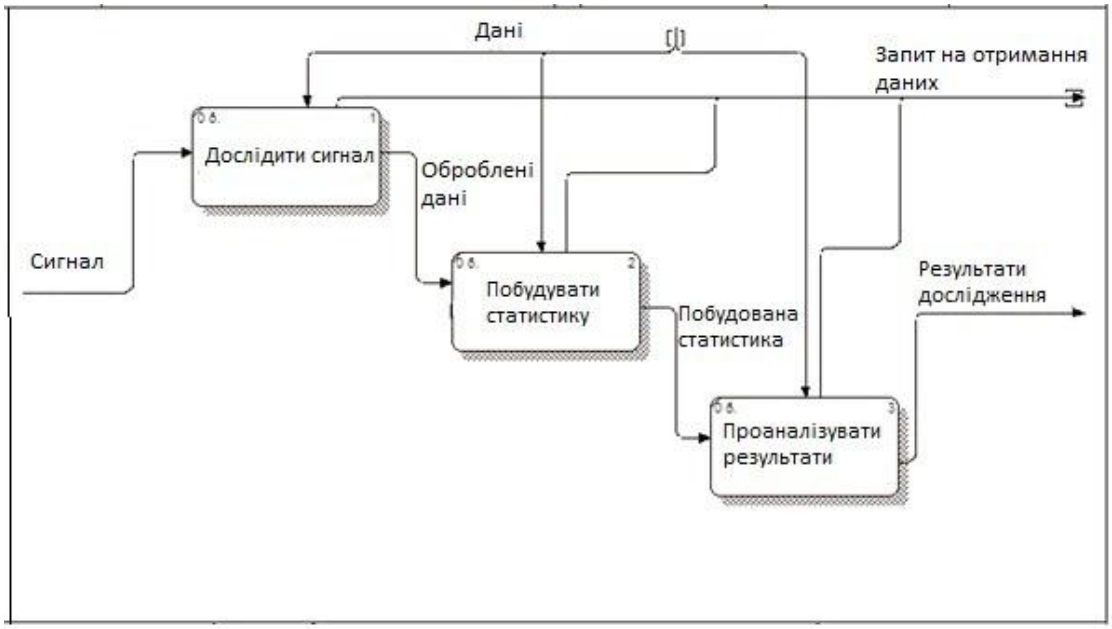


Рисунок 1.5 – DFD-діаграма

Більш детально процес порівняльного аналізу методів розглянуто на рисунку 1.6 у вигляді декомпозиції роботи.

Отже процес «Побудова статистики та аналіз результатів» поділено на підзадачі:

– обробка статистичних даних, які можуть буди використані для прогнозування атаки по перевищенню порогу запитів;

– вирахування потенційно небезпечних місць, для розподілення затрат ЕОМ на аналіз сигналів;

– оформлення фінальних звітів для передачі потенційному замовнику та для прийняття подальших дій.

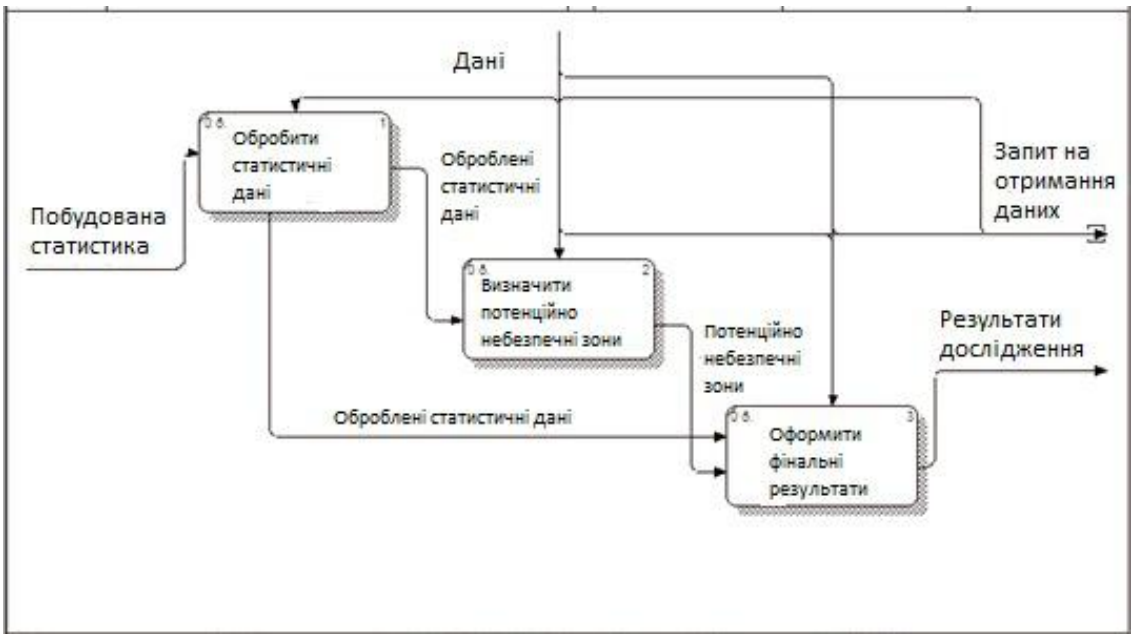


Рисунок 1.6 – DFD-діаграма першого рівня

## 1.2 Аналіз сценаріїв вирішення проблеми порівняльного аналізу методів прогнозування DDoS-атак

### 1.2.1 Модель аналізу проблеми

Об’єкт дослідженні – модель порівняльного аналізу методів розпізнавання DDoS-атак.

Мета дослідження – знайти найбільш оптимальний та точний метод для отримання висновку щодо стану сервера на предмет атак.

Математична модель має бути вирішена при застосуванні різних алгоритмів з подальшим їх порівнянням з точки зору точності висновків та особливостей програмної реалізації. В результаті очікується визначення найбільш оптимального алгоритму, який може бути застосований для вирішення даної задачі.

Виділимо критерії, які на наш погляд здійснюють найбільший вплив на очікуваний результат. Такими критеріями є:

– швидкодія (К1);

– універсальність (К2);

– надійність (К3);

– ресурсоємкість (К4).

Розглянемо вищевказані критерії більш детально та конкретизуємо що саме буде порівнюватися у запропонованих далі альтернативах. В свою чергу, альтернативами будуть обрані алгоритми розпізнавання атак.

Порівнюючи ресурсоємкість альтернатив, маємо на увазі ресурси ЕОМ, які будуть необхідні для успішного закінчення роботи алгоритмів без технічних помилок: об’єм оперативної пам’яті, машинний час виконання розрахунків.

Порівнюючи альтернативи на універсальність, нас цікавить можливість застосування алгоритмів для вибірок достатньо великого об’єму і вхідні дані, які потребує алгоритм. Перевага буде віддана тому алгоритму, який потребує менше інформації на вхід.

Надійність алгоритмів: очікується, що у всіх випадках ми отримаємо однозначний результат від алгоритму щодо атаки на сервер.

При порівнянні альтернативи на швидкодію, перевага віддаватиметься алгоритму, який матиме мінімальний сумарний час роботи.

Будуть розглянуті такі альтернативи:

– використання дерев рішень;

– використання нейронних мереж;

– порівняння показників Херста.

Відомо, що нейронна мережа досить ресурсномістка, за рахунок складності структури. Але, за рахунок її методів аналізу досить проста у реалізації, так як працює за рахунок повторення одноманітних дій, але вже з використанням попередніх результатів, що значно економить час на програмний опис. Проте вона потребує зберігання результатів попередньої ітерації для донавчання, що потребує більше оперативної пам’яті. Однак попри

велику ресурсоємкість, вона є більш універсальною, на відміну від дерев рішень та показників Херста, та дає більш точний результат.

Метод дерев рішень є досить простим у плані математичного розуміння, однак для нашої задачі потребує багато розгалужень для більш точної відповіді, що значно збільшує час відпрацювання алгоритму. По показнику надійності він поступається нейронній мережі, однак є надійнішим за метод порівняння показників Херста.

Метод порівняння показників Херста є найменш ресурсномістким на найбільш простим у реалізації алгоритмом у порівнянні з нейронними мережами та деревами рішень. Однак він, водночас, є найменш точним та надійним методом. По часу відпрацювання алгоритму він прирівнюється до використання дерев рішень так як формула потребує набагато більше арифметичних дій. На рисунку 1.7 зображена ієрархічна модель процесу аналізу незадоволеностей.

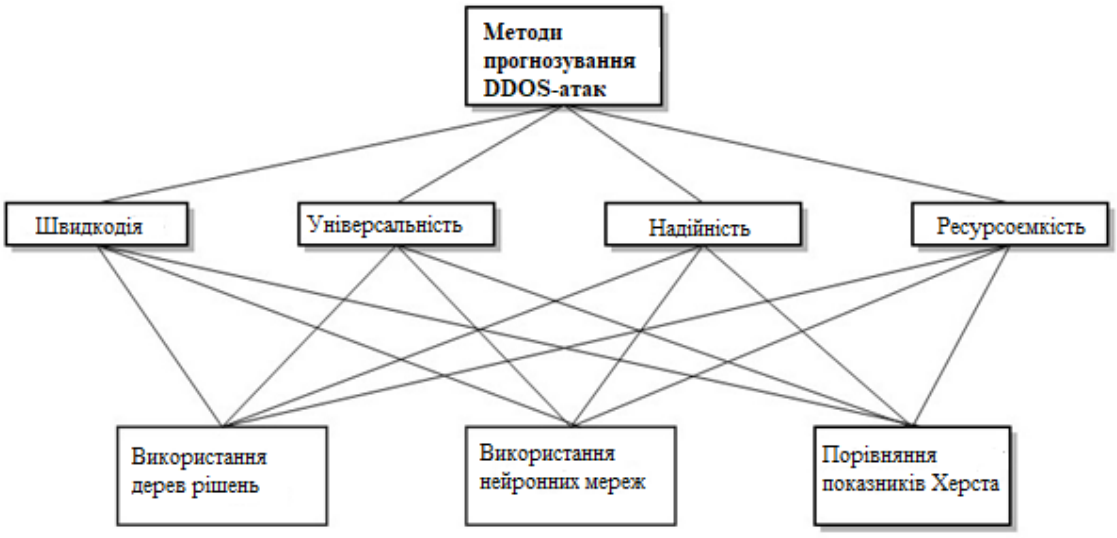


Рисунок 1.7 – Ієрархічна модель процесу аналізу незадоволеностей

### 1.2.2 Оцінювання вектора пріоритетів незадоволеностей методом аналізу ієрархій

За допомогою методу попарних порівнянь побудуємо модель процесу аналізу незадоволеностей. Аналіз незадоволеностей включає в себе такі рівні:

– нульовий рівень – компоненти проблеми;

– перший рівень – класифікація незадоволеностей;

– другий рівень – характеристики компонентів, які впливають на результат поставленої задачі.

На рисунку 1.7 представлений нульовий рівень проблеми. На першому рівні аналізу проблеми побудуємо матрицю попарних порівнянь критеріїв з метою оцінки впливу кожної незадоволеності на поставлену проблему. Результати наведені в таблиці 1.1.

Таблиця 1.1 – Матриця попарних порівнянь критеріїв

| | К1 | К2 | К3 | К4 | Вектор пріоритетів |
|---|---|---|---|---|---|
| К1 | 1,000 | 7,000 | 1,000 | 5,000 | 0,509 |
| К2 | 0,143 | 1,000 | 3,000 | 0,200 | 0,113 |
| К3 | 1,000 | 0,333 | 1,000 | 5,000 | 0,238 |
| К4 | 0,200 | 5,000 | 0,200 | 1,000 | 0,140 |

Для знаходження індексу узгодженості знаходимо суми елементів 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

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

та індекс узгодженості:

$$CI^k = \frac{5,505 - 4}{4 - 1} = 0,68.$$

Оскільки матриця попарних порівнянь критеріїв – це матриця четвертого порядку, то відношення узгодженості:

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

Оскільки відношення узгодженості є близьким до 0.1, то вважатимемо, що матриця попарних порівнянь критеріїв побудована правильно.

Далі формуємо матриці попарних альтернатив за кожним критерієм з метою порівняння методів між собою за кожним критерієм окремо.

Таблиця 1.2 – Матриця порівнянь за критерієм 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 |

Для знаходження індексу узгодженості знаходимо суми елементів матриці за стовбцями:

$$y_1 = 1{,}0 + 9{,}0 + 7{,}0 = 18{,}0,$$
$$y_2 = 0{,}111 + 1{,}0 + 0{,}2 = 1{,}311,$$
$$y_3 = 0{,}143 + 5{,}0 + 1{,}0 = 6{,}243.$$

Тоді

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

та індекс узгодженості:

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

Оскільки матриця попарних порівнянь альтернатив – це матриця третього порядку, то відношення 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 |

Для знаходження індексу узгодженості знаходимо суми елементів матриці за стовбцями:

$$y_1 = 1{,}0 + 0{,}143 + 0{,}143 = 1{,}286,$$

$$y_2 = 7{,}0 + 1{,}0 + 0{,}5 = 8{,}5,$$

$$y_3 = 7{,}0 + 2{,}0 + 1{,}0 = 10{,}0.$$

Тоді

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

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 критерієм 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 | 0,265 |

Для знаходження індексу узгодженості знаходимо суми елементів матриці за стовбцями:

$$y_1 = 1{,}0 + 9{,}0 + 0{,}333 = 10{,}333,$$

$$y_2 = 0{,}111 + 1{,}0 + 0{,}333 = 1{,}444,$$

$$y_3 = 0{,}2 + 3{,}0 + 1{,}0 = 4{,}2.$$

Тоді

$$\lambda_{\max} \approx 0{,}063 \cdot 10{,}333 + 0{,}672 \cdot 1{,}444 + 0{,}265 \cdot 4{,}2 = 3{,}028$$

та індекс узгодженості:

$$CI_{K3}^{A} = \frac{3{,}028 - 3}{3 - 1} = 0{,}014.$$

Відношення узгодженості:

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

Таблиця 1.5 – Матриця порівнянь за критерієм 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 |

Для знаходження індексу узгодженості знаходимо суми елементів матриці за стовбцями:

$$y_1 = 1{,}0 + 5{,}0 + 3{,}0 = 9{,}0,$$
$$y_2 = 0{,}2 + 1{,}0 + 0{,}333 = 1{,}533,$$
$$y_3 = 0{,}333 + 3{,}0 + 1{,}0 = 4{,}333.$$

Тоді

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

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:

$$\vec{p} = \begin{bmatrix} 0{,}051 & 0{,}773 & 0{,}063 & 0{,}105 \\ 0{,}722 & 0{,}139 & 0{,}672 & 0{,}637 \\ 0{,}227 & 0{,}088 & 0{,}256 & 0{,}58 \end{bmatrix} \cdot \begin{bmatrix} 0{,}509 \\ 0{,}113 \\ 0{,}238 \\ 0{,}140 \end{bmatrix} = \begin{bmatrix} 0{,}143 \\ 0{,}632 \\ 0{,}225 \end{bmatrix}.$$

Розрахуємо індекс узгодженості та відношення узгодженості для всієї ієрархії:

$$CI = 0{,}168 + 0{,}509 \cdot 0{,}141 + 0{,}113 \cdot 0{,}048 + 0{,}238 \cdot 0{,}024 + 0{,}140 \cdot 0{,}020 = 0{,}254,$$

$$RI = 0{,}90 + 0{,}58 = 1{,}48,$$

$$CR = \frac{CI}{RI} = \frac{0{,}254}{1{,}48} = 0{,}172,$$

що теж можна вважати доброю узгодженістю.

Найбільша компонента вектора локальних пріоритетів критеріїв відповідає першому критерію. Отже, маємо наступні пріоритети за критеріями порівняння: швидкодія, надійність, ресурсоємність, універсальність.

Порівнюючи альтернативи за обраними критеріями, отримали вектор глобальних пріоритетів, найбільша компонента якого відповідає другій альтернативі – використанню нейронних мереж.

## 1.3 Змістовна та формальна постановка задачі

### 1.3.1 Змістовна постановка задачі

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. Їх потужність збільшилася майже вчетверо в порівнянні з минулими роками. Для забезпечення інформаційної безпеки в організаціях необхідно вміти відображати або запобігати DoS/DDoS-атаки, але на сьогоднішній день немає ефективного вирішення.

DoS (Denial of Service – атака типу «відмова в обслуговуванні») – це хакерська атака на систему з метою вивести її з ладу, довести її до відмови. Це створення умов, при яких звичайні користувачі системи можуть не отримати доступ до конкретних системних ресурсів, або цей доступ є ускладненим.

DDoS (Distributed Denial of Service – розподілена атака «відмова в обслуговуванні») – це такий вид атак, що виконується одночасно з декількох комп'ютерів.

Дуже часто безпорадними жертвами атак DDoS стають іменити компанії, які не подбали про відповідні засоби захисту. Серед постраждалих зустрічалися і такі, хто прийняв не найвдаліше рішення при виборі апаратних засобів захисту від подібних атак, адже і в цій сфері продукти від різних постачальників значно розрізняються за якістю. Іноді в них відсутні елементи, які інтернет-залежні підприємства визнали б, без сумніву, вкрай важливими, якби знали про їхнє існування. При виборі відповідного рішення необхідно приділити пильну увагу деталям пропонованих методів і відмінностей між ними, щоб забезпечити найкращий захист системи.

Для протидії DDoS-атакам застосовується цілий ряд механізмів, проте особливу значимість мають засоби виявлення вторгнень. Адже своєчасне виявлення DDoS-атаки дозволить зберегти працездатність мережі, так як якщо не зупинити у провайдера трафік, призначений для переповнення атакуємої мережі, то зробити це на вході виявиться неможливим, оскільки вся смуга пропускання буде вже зайнята.

Завдання методів виявлення DDoS-атак полягає в тому, щоб аналізуючи дані, отримані з серверу, можна зробити найбільш точний прогноз щодо того, атакують сервер на даному проміжку часу, чи ні. Найважливіше у цьому питанні врахувати швидкодію обраного методу, так як найменше зволікання може привести до значних фінансових втрат.

Системи безпеки вже давно можуть збирати різні події в загальні файли, але питання в тому, що з цим робити – як це інтерпретувати? При правильній архітектурі системи безпеки – система може зібрати всі розрізнені події в одну загальну картину, яка відразу прояснює клієнту що і звідки у нього було атаковано. Зібравши відповідні дані в одну купу постає необхідність виявити ті чи інші закономірності та скласти їх у словник, для наступного використання на вже реальних даних [6].

Стандартні методи аналізу статистики не дозволяють виявляти невідомі раніше атаки, тому в якості механізму вирішення даної проблеми активно виступають нейронні мережі. Штучна нейронна мережа – це математична модель, а також її програмна або апаратна реалізація, побудована за принципами функціонування біологічних нейронних мереж – мереж нервових клітин будь-якого живого організму. ШНМ використовуються практично у всіх засобах виявлення вторгнень як окремо, так і в комплексі з іншими механізмами захисту [7].

### 1.3.2 Формальна постановка задачі класифікації

Для розпізнавання та прогнозування DDoS-атак розглянемо двокласову класифікацію з непересічними класами та набором бінарних ознак.

Нехай $X$ – простір об’єктів, а $Y = \{-1,1\}$ – множина допустимих відповідей. Кожен об'єкт описується $n$ набором своїх характеристик, які називаються ознаками $f_i : X \to \mathbb{R}, j = 1,2,...n$. Вектор $x = (x^1,...,x^n) \in \mathbb{R}^n$, де $x^j = f_j(x)$, називається вектором ознак об'єкта $x$.

Якщо дискримінантна функція визначається як скалярний добуток вектора $x$ на вектор параметрів $\omega \in \mathbb{R}^n$, то отримаємо лінійний класифікатор:

$$a(x,\omega) = sign\ \left\langle w,x\right\rangle - \omega_0\ = sign\left(\sum_{j=1}^{n}\omega_j f_j(x) - \omega_0\right). \qquad (1.1)$$

Рівняння $\left\langle \omega,x\right\rangle = 0$ задає гіперплощину, що розділяє класи в просторі $\mathbb{R}^n$. Якщо вектор $x$ знаходиться по одну сторону гіперплощини з її направляючим вектором $\omega$, то об'єкт $x$ відноситься до класу $(+1)$, інакше – до класу $(-1)$. Параметр $\omega_0$ у формулі (1.1) іноді опускають. Іноді вважають, що

серед ознак є константа, $f_j(x) \equiv -1$, і тоді роль вільного коефіцієнта $\omega_0$ грає параметр $\omega_j$.

Завдання полягає в побудові такого вирішального правила $\hat{f}(x)$, щоб розпізнавання проводилося з мінімальним числом помилок. В нашому випадку, множина $X$ – кінцева вибірка спостережень, тобто запити, отримані з сервера, а $Y = \{-1,1\}$ – множина можливого стану системи.

Запити розбиваються та описуються змістовними характеристиками, які нормуються, та перетворюються у $n$-мірний числовий вектор $x = (x^1,...,x^n)$, де $x^j \in \{0,1\}$ , що називають вектором бінарних характеристик [8].

## 1.4 Постановка задачі дослідження

Метою дослідження є розробка механізму, що дозволяє класифікувати дані сервера з метою аналізу їх на предмет ведення DDoS-атаки. Виходячі з цього, сформулюємо задачі для дослідження в рамках даної дипломної роботи:

– ознайомитися з алгоритмами машинного навчання та нейронних мереж, вивчаючи відповідну літературу;

– зібрати тренувальні дані необхідні для нейронної мережі, та тестову вибірку для її подальшої класифікації;

– обрати інструменти програмної розробки та розробити архітектуру програми, в якій будуть реалізовані обрані алгоритми ти якісний аналіз отриманих прогнозів;

– на основі реалізованого програмного забезпечення провести ряд експериментів, змінюючи вхідні дані (об’єм вибірки, кількість епох, прихованих шарів), оформити результат у вигляді таблиці;

– на основі отриманих даних зробити висновок про проведену роботу.

# 2 ВИБІР ТА ОБҐРУНТУВАННЯ МЕТОДА РОЗВ’ЯЗАННЯ

## 2.1 Машинне навчання

Машинне навчання – один з методів штучного інтелекту, основною перевагою яких є непряме розв’язання поставлених задач, а навчання в процесі застосування рішень безлічі подібних завдань. Для побудови таких методів використовуються засоби статистики, чисельні методи, методи оптимізації, теорії ймовірностей, теорії графів, різні техніки роботи з даними в цифровій формі.

Розділяють декілька типів навчання:

– навчання за прецедентами, чи індуктивне навчання, що базується на знаходженні деяких емпіричних закономірностей у вибірці;

– дедуктивне навчання означає певну формалізацію та оцінку експертних знань та перенесення їх у базу знань.

Дедуктивне навчання зазвичай відноситься до експертних систем, саме тому «навчання по прецедентах» та «машинне навчання» вважають ідентичними поняттями. Більшість методів та функцій індуктивного навчання розроблювалися щоб бути альтернативою статистичним підходам. Велика кількість методів тісно пов'язана з інтелектуальним аналізом даних та витяганням інформації.

Машинне навчання, з однієї сторони, утворилося як результат розподілу науки про нейронні мережі на самонавчання мереж і певні підвиди топологій їх архітектури, а з інакшої сторони – взяло під основу статистичні методи.

Зазначені далі методи машинного навчання базуються на нагоді використання нейромереж, хоча існують також і інші методи, які використовують поняття навчальної вибірки – наприклад, дискримінантний аналіз, який оперує поняттями узагальненої дисперсії і коваріації, або байєсовські класифікатори [9].

Основні види нейронних мереж, наприклад перцептрон і багатошаровий перцептрон, мають можливість навчатися по принципам навчання з учителем, без вчителя, навчанням з підкріпленням а також самоорганізацією. Проте деякі нейронні мережі а також переважну більшість методів математичної статистики можливо винести лише до одного із способів машинного навчання. Саме тому, якщо є необхідність класифікації методів навчання стосовно способу їх самонавчання, буде неправильним класифікувати нейромережі до одного з певних видів, краще – типізувати методи самонавчання мереж.

Далі наведемо перелік класичних задач, що можуть бути вирішенівикористовуючи машинне навчання:

– класифікація: виконується у більшості випадків методами навчання з учителем;

– кластеризація: виконується у більшості випадків методами навчання без учителя;

– регресія: виконується у більшості випадків методами навчання з учителем впродовж етапу тестування, також являється окремим випадком прогнозування;

– поновлення щільності розподілу ймовірностей за певною вибіркою;

– зниження розмірності даних а також візуалізація даних вирішується методами навчання без учителя;

– побудова залежностей рангів;

– описання взаємовідносин між групами об'єктів, найбільш відомий випадок: попарні подібності, які виявляються з допомогою матриць відстаней;

– часовий ряд чи певний сигнал;

– зображення чи певний відеоряд.

Одною з головних цілей машинного навчання являється повна чи часткова автоматизація розв’язання складних завдань у різноманітних професіях задля полегшення праці людини.

Областей застосування методів машинного навчання становиться більше з кожним днем. Завдяки глобальній інформатизації накопичуються величезні

об’єми даних в різноманітних сферах життя. Виникаючі завдяки цим процесам задачі управління,прогнозування та прийняття рішень найчастіше призводять до навчання по прецедентах [10].

## 2.2 Типи машинного навчання

Існує безліч моделей для машинного навчання, але вони, як правило, відносяться до одного з трьох типів:

- навчання з учителем (supervised learning);
- навчання без учителя, або самонавчання (unsupervised learning);
- навчання з підкріпленням (reinforcement learning).

Залежно від виконуваної завдання, одні моделі можуть бути більш придатними і більш ефективними, ніж інші.

За останні десятиліття машинне навчання безпрецедентно просунулося в таких різних областях, як розпізнавання образів, робомобілі і складні ігри. Ці успіхи в основному були досягнуті через навчання глибоких нейромереж. Усі типи навчання вимагають розробки людиною навчальних сигналів, що передаються потім до комп'ютера. У разі навчання з учителем це «цілі», у випадку з підкріпленням це «нагороди» за успішну поведінку. Тому межі навчання визначаються людьми [11].

### 2.2.1 Навчання з учителем

У цьому типі коректний результат при навчанні моделі явно позначається для кожного ідентифікованого елемента в наборі даних. Це означає, що при зчитуванні даних у алгоритму вже є правильна відповідь. Тому замість пошуків відповіді він прагне знайти зв'язку, щоб в подальшому, при введенні непозначених даних, виходили правильні класифікація або прогноз.

В контексті класифікації алгоритм навчання може, наприклад, забезпечуватися історією транзакцій по кредитних картах, кожна з яких позначена як безпечна або підозріла. Він повинен вивчити відносини між цими двома класифікаціями, щоб потім зуміти відповідним чином маркувати нові операції в залежності від параметрів класифікації (наприклад, місце покупки, час між операціями і т. д.).

У разі коли дані безперервно пов'язані один з одним, як, наприклад, зміна курсу акцій у часі, регресійний алгоритм навчання може використовуватися для прогнозування наступного значення в наборі даних [12].

### 2.2.2 Навчання без вчителя

В цьому випадку у алгоритму в процесі навчання немає заздалегідь встановлених відповідей. Його мета – знайти смислові зв'язки між окремими даними, виявити шаблони і закономірності. Наприклад, кластеризація – це використання неконтрольованого навчання в рекомендаційних системах (наприклад, люди, яким сподобалася ця пляшка вина, також позитивно оцінили ось цю).

У навчанні без учителя складно обчислити точність алгоритму, тому що в цих відсутні «правильні відповіді» або мітки. Але розмічені дані часто ненадійні або їх занадто дорого отримати. У таких випадках, надаючи моделі свободу дій для пошуку залежностей, можна отримати хороші результати.

Прикладів завдань навчання без вчителя дуже багато. Наприклад, це сегментація користувачів інтернет-магазину або мобільного оператора. Їм часто необхідно виявити групи схожих користувачів, щоб далі, наприклад, займатися маркетингом для кожної групи окремо. Проаналізувати, що такого особливого в цій групі, що всі користувачі в ній схожі, і орієнтувати рекламу саме на цей сегмент. Але при цьому кластеризувати можна не тільки людей. Наприклад,

можна кластеризувати гени, намагаючись знайти такі групи генів, які одночасно включаються або вимикаються у різних людей в різних умовах [13].

### 2.2.3 Навчання з підкріпленням

Цей тип навчання є сумішшю перших двох. Зазвичай він використовується для вирішення більш складних завдань і вимагає взаємодії з навколишнім середовищем. Дані надаються середовищем і дозволяють алгоритму реагувати і вчитися.

Область застосування такого методу є обширно: від контролю роботизованих рук і пошуку найбільш ефективної комбінації рухів, до розробки систем навігації роботів, де поведінковий алгоритм «уникнути зіткнення» навчається досвідченим шляхом, отримуючи зворотний зв'язок при зіткненні з перешкодою.

Логічні ігри також добре підходять для навчання з підкріпленням, так як вони зазвичай містять логічний ланцюжок рішень: наприклад, покер, і нарди, в яку нещодавно виграв AlphaGo від Google.

Цей метод навчання також часто застосовується в логістиці, складанні графіків і тактичному плануванні завдань [14].

## 2.3 Огляд задач класифікації

Задачі класифікації – це математичні задачі, які мають безліч об'єктів, деяким чином розподілених на певні класи. Дана множина об'єктів, яка є кінцевою, та для якої відомо класи, до яких належить кожен член цієї множини. Ця множина має назву вибірка.

Приналежність інших об'єктів до певних класів невідома. Мета задачі: побудувати алгоритм, що класифікує певний довільний об'єкт із заданої множини.

Класифікувати об'єкт – означає надати йому свій номер чи ім’я класу, до якого він належить.

Класифікація об'єкта – це певне найменування або номер такого класу, що видається за рішенням алгоритму внаслідок результату застосування до заданого конкретного об'єкту.

У математичній статистиці завдання класифікації також називають завданнями дискримінантного аналізу. У контексті машинного навчання завдання класифікації розв’язується, зокрема, з допомогою методів штучних нейронних мереж, якщо ставити експеримент у вигляді задачі навчання з учителем.

Для задачі класифікації виділяють такі типи класів:

1) двухкласова класифікація (найпростіший випадок, що є основою, та використовується для вирішення складних завдань);

2) багатокласова класифікація (коли кількість класів стає занадто високою, класифікація стає істотно важчою у реалізації);

3) пересічні класи;

4) непересічні класи;

5) нечіткі класи (необхідно визначати ступінь приналежності об'єкта до кожного класу) [15].

Результати попередніх обчислень довели, що найбільш оптимальним методом для розв’язання алгоритмів задачі двохкласової (бінарної) класифікації з нечіткими класами є використання нейронних мереж. Протягом виконання алгоритмів буде перевірена множина ознак, за якими буде побудований прогноз щодо ведення атаки на сервер.

## 2.4 Нейронні мережі в задачах класифікації

Нехай є $m$ об'єктів, кожен з $n$ параметрами. Завдання $k-$класифікації полягати в пошуку за допомогою цих даних функції:

$$f : \mathbb{R}^n \to \{0,...,K-1\},$$

яка будь-якому об'єкту ставить у відповідність клас.

Нейронна мережа – математична модель для вирішення задачі класифікації і передбачення. Далі ми будемо її розглядати тільки як класифікатор. За своєю суттю – це граф з $n$ вершинами-входами і $K$ вершинами-виходами. Тут $K$ – кількість класів, в завданні класифікації. Вершини в цьому графі називаються нейронами. Нейрони пов'язані зваженими ребрами. Значення нейрона визначається вагами ребер входять в нього і значенням нейронів на протилежних кінцях цих ребер. За $n$ параметрами об'єкту нейронна мережа видає $K$ чисел в відрізку $[0,1]$, індекс максимального числа оголошується класом об'єкта [16].

Нейронна мережа прямого поширення − це нейронна мережа розбита на шари, де все ребра спрямовані 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).

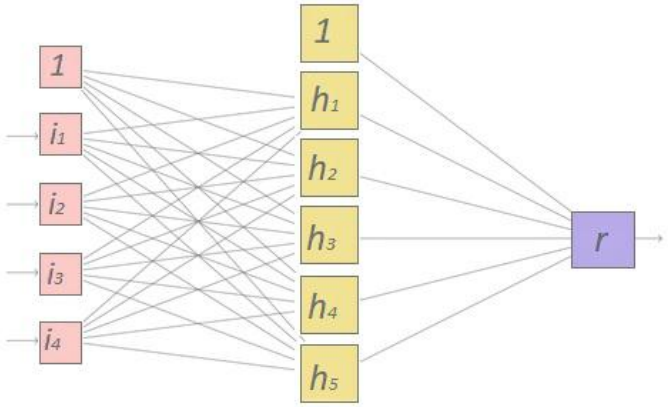


Рисунок 2.1 – Одношарова нейронна мережа прямого поширення

Тут нейрони з вхідного шару позначаються $i_s$, нейрони з прихованого шару $h_s$ і $r$ – нейрон вихідного шару. У кожному шарі, крім останнього додається фіктивний нейрон з значенням 1. Ваги ребер між шаром з $n$ нейронами і з $m$ нейронами представляється матрицею $\Theta$ розміру $m \times (n+1)$, тобто $\theta_{ij}$ – вага ребра між *i*-им і *j*-им нейроном в різних шарах, причому $\theta_{j0}$ – ваги ребер з фіктивного нейрона. Кожен шар представимо як вектор $v$ з значень нейронів, позначимо за $\tilde{v}$ вектор $v$ розширений одиницею.

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

$$h = \sigma(\Theta^{i,h}\tilde{i}), r = \sigma(\Theta^{h,r}\tilde{h}),$$

де $m$ діє покоординатно і має вигляд:

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

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

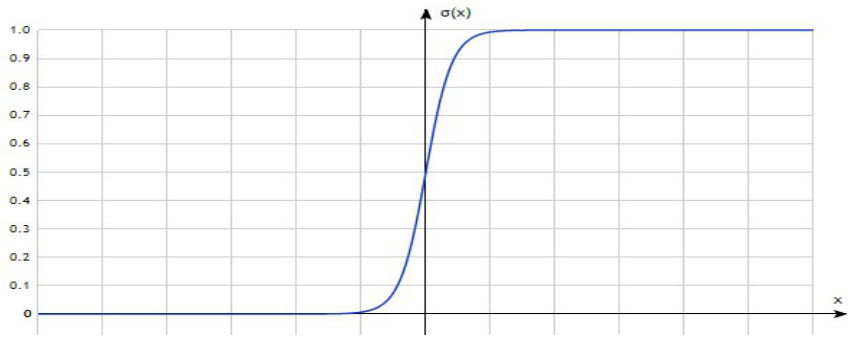


Рисунок 2.2 – Графік функції $\sigma(x)$

Послідовне обчислення значення нейронів у всіх шарах називають прямим проходом. Аналогічні дії робляться для нейронної мережі з великою кількістю шарів, однак, згідно з Узагальненою апроксимаційною теоремою, яка стверджує що нейронна мережа прямого поширення з одним прихованим шаром і з кінцевим числом нейронів дозволяє обчислювати значення довільної неперервної функції, з будь-якою точністю.Тільки немає гарантій, що можна легко знайти відповідні такій нейронній мережі ваги ребер і кількість вершин в прихованому шарі. Пошук матриць $\Theta$ відповідних ваг між шарами називається процесом навчання нейронної мережі. Далі в роботі нейронні мережі прямого поширення будуть називатися просто нейронними мережами [17].

## 2.5 Штрафна функція

Розглянемо вибірку з $m$ об'єктів (тренувальна вибірка), про кожен з котрих відомий клас, до якого він належить. Назвемо $\Theta$ вектор всіх ваг ребер нейронної мережі. Припустимо, що ми знайшли якусь $\Theta$, тепер 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 $i$-ий об'єкт належить класу $c_i \in \{0,\ldots,K-1\}$, тоді введемо бінарний вектор, такий що його $c_i$-я компонента дорівнює одиниці, а інші рівні нулю.

Так само введемо функцію $h_\Theta(x)$ що видає результат роботи нейронної мережі (вектор з $K$ чисел в інтервалі $[0,1]$) для $n$-мірного вектора (об'єкта) $x$.

Функція $J_i(\Theta)$ виглядає так:

$$J_i(\Theta) = \sum_{k=0}^{K} y^{(i)} - \log(h_\Theta(x^{(i)})_k) - (1-y^{(i)})\log(1 - h_\Theta(x^{(i)})_k). \qquad (2.1)$$

З формули (2.1) видно, що для абсолютно вірних прогнозів (коли $h_{\Theta}(x^{(i)}) = y^{(i)}$) значення цієї функції дорівнюватиме нулю, і значення функції збільшується при видаленні передбачення від вірного. Тепер розглянемо штрафну функцію $J_i(\Theta)$ що описує якість передбачення відразу для всіх об'єктів з вибірки:

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

Тепер щоб навчити мережу досить знайти мінімум функції $J_i(\Theta)$.

## 2.6 Проблема перенавчання та недонавчання

Якщо з певної $\Theta$-й нейронна мережа буде занадто точно прогнозувати результат для тренувальної вибірки, а в цій вибірці є шуми або відхилення, то це негативно вплине на якість прогнозів для об'єктів які не перебувають в тренувальній вибірці, ця проблема називається проблемою перенавчання.

Протилежна ситуація трапляється, коли для тренувальної вибірки нейронна мережа робить неточні прогнози і значення функції $J(\Theta)$ велике, таке положення справ називається проблемою недонавчання.

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

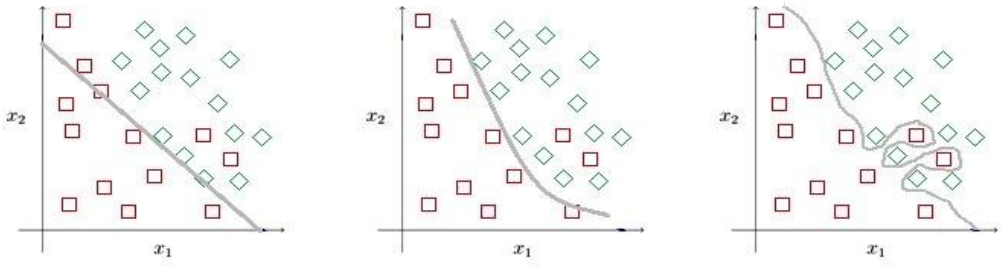


Рисунок 2.3 – а) недонавчання, б) задовільне навчання, в) перенавчання

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.

У нейронних мережах ці проблеми пов'язані з кількістю нейронів в прихованому шарі. Якщо їх досить багато трапляється перенавчання, якщо мало – недонавчання. Для вирішення цих проблем в нейронній мережі функція $J(\Theta)$ модифікується в такий спосіб:

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

де $s_i$ – кількість нейронів в шарі номер $i$;

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

$\lambda > 0$ – називається параметром регуляризації: коли він занадто великий нейронна мережа називається упередженою і відбувається недонавчання, коли ж він не досить великий може статися перенавчання [18].

## 2.7 Вилучення ознак

В машинному навчанні, розпізнаванні образів та в обробці зображень вилучення ознак починається з первинного набору даних вимірювань, і будує похідні значення (ознаки), покликані бути інформативними та ненадлишковими, полегшувати наступні кроки навчання та узагальнення, і в деяких випадках вести до кращих тлумачень людьми. Виділяння ознак пов'язане зі зниженням розмірності.

Коли вхідні дані алгоритму є занадто великими, 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 може бути перетворено на скорочений набір ознак (що також називають вектором ознак). Цей процес називається виділенням або вилученням ознак. Очікується, що виділені ознаки містять доречну інформацію з вхідних даних, так що бажане завдання може бути виконано із застосуванням цього скороченого представлення замість повних первинних даних.

Виділяння ознак включає зниження кількості ресурсів, необхідних для опису великого набору даних. При виконанні аналізу складних даних одна з головних проблем випливає з кількості залучених змінних. Аналіз із великою кількістю змінних в загальному випадку вимагає великої кількості пам'яті та обчислювальних потужностей, або алгоритмів класифікації, що перенавчаються тренувальної вибірки, й погано узагальнюються на нові. Виділяння ознак є загальним терміном для позначення методів побудови таких поєднань змінних, щоби обходити ці проблеми, зберігаючи достатню точність опису даних [19].

Так як на практиці дані рідко надходять у вигляді готових до використання матриць, кожне завдання починається з вилучення ознак. В нашому випадку достатньо прочитати файл і перетворити його в numpy.array.

# 3 ПРОГРАМНА РЕАЛІЗАЦІЯ

## 3.1 Python як високорівнева мова програмування

Python є, в першу чергу, об'єктно-орієнтованою мовою програмування. Він 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. У цьому він перевершив першу таку мову C, так як її алгоритми та структури даних на різноманітних машинах могли займати не однаковийоб’єм пам'яті і це служило перешкодою під час написання програми за принципами адаптованості та гнучкості. Python не володіє подібним недоліком.

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. Друга опція: написати на C свої методи та алгоритми до Python та зібравши свій додаток, можна отримати "розширений" інтерпретатор з написаними вами інноваційними можливостями.

Одна з найбільших переваг мови Python – це присутність багатьох 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 [20].

## 3.2 Багатопроцесорність у Python

Існують дві найпоширеніші причини використовувати потоки: по-перше, для збільшення ефективності використання багатоядерної архітектури сучасних процесорів, а значить, і продуктивності програми. Також, якщо нам потрібно розділити логіку роботи програми на паралельні повністю або частково асинхронні секції (наприклад, мати можливість нотифікувати кілька серверів одночасно). Проте використання багатопоточності має два, досить вагомих, недоліки:

– кожному потоку потрібно місце для роботи з даними, тому кожен потік забирає пам'ять навіть якщо вона не використовується і потік спить;

– якщо два потоку використовують один і той же дефіцитний ресурс, може статися змагання потоків за ресурс.

Тут нам може допомогти інший підхід. В Python є модуль multiprocessing. Процеси можна створювати зі звичайних функцій. Методи роботи з процесами майже всі ті ж самі, що і для потоків з модуля threading. А ось для синхронізації процесів і обміну даними прийнято використовувати інші інструменти. Йдеться про черги і канали.

Крім того в модулі multiprocessing є механізм роботи із загальною пам'яттю. Для цього в модулі є класи змінної і масиву, які можна "узагальнювати" між процесами.

## 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: для "поганих" і "хороших" запитів. В результаті у нас повинні вийти три файли з даними для тренування і класифікації:

– good_logs – зберігає хороші запити до початку атаки;

– bad_logs – зберігає погані 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, нам потрібно ввести єдиний формат даних. 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 тип запиту, url і http_version;

– url, розділена на протокол, ім'я хоста, шлях і всі ключі від query_string;

– referer, розділений аналогічно url в запиті;

– user-agent;

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

Після такого розподілення маємо на руках список всіляких маркерів, які можуть бути присутні в запиті. Це і є наш словник. Словник потрібен для того, щоб з будь-якого можливого запиту створити feature-vector.

Бінарний, М-мірний вектор (де М – 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 multiprocessingмови програмування Python для пришвидшення роботи програми.

# 4 РЕЗУЛЬТАТИ ОБЧИСЛЮВАЛЬНОГО ЕКСПЕРИМЕНТУ

## 4.1 Підготовка даних

В першу чергу необхідно підготувати два набори даних: для "поганих" і "хороших" запитів.

Ці дані повинні бути одного формату та достатньо легкі, так як аналіз даних є найбільш витратним за часом.

Для дослідження у нашому випадку були взяті запити, отримані з сервера у форматі access.log.

Такий формат подання запитів є найбільш загальновживаним та простим у використанні. На рис. 4.1 та 4.2 зображені приклади запитів до початку та на момент атаки відповідно:

```
0.0.0.0 - - [20/Dec/2011:15:00:03 +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 запиту до початку DdoS атаки

```
0.0.0.0 - - [20/Dec/2011:20:00:08 +0400] "POST /forum/index.php HTTP/1.1" 503 107 "htt
p://www.mozilla-europe.org/" "-"
```

Рисунок 4.2 – Приклад запиту на момент DdoS атаки

В результаті у нас повинні вийти 3 набора даних:

– набір даних з "хорошими" запитами до початку DDoS'a;

– набір даних з " поганими " запитами на момент DDoS атаки;

– набір даних, який нам необхідно класифікувати.

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

– тренувальний набір, на якому ми навчаємо нашу нейронну мережу;

– тестовий набір, яким ми перевіряємо якість навчання мережі.

Таке розбиття обумовлено тим фактом, що нейронна мережа з найменшою помилкою на тренувальному наборі може видавати велику помилку на нових даних, бо ми «перенавчили» мережу, заточивши її під тренувальний набір даних.

## 4.2 Відбір і виділення ознак

Для побудови класифікатора необхідно визначити, які параметри впливають на прийняття рішення про те, до якого класу належить зразок.Тому після підготовки даних необхідно сформувати так званий «словник ознак», який в подальшому буде використовуватися нейронною мережею для розпізнавання запитів на предмет DDoS-атак.

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

– сам запит, розділений на тип запиту, url, що в свою чергу розділена на протокол, ім'я хоста, шлях і всі ключі від query_string, і http_version;

– referer, розділений аналогічно url в запиті;

– user-agent;

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

Приклад розділенного запита можна побачити на рис. 4.3

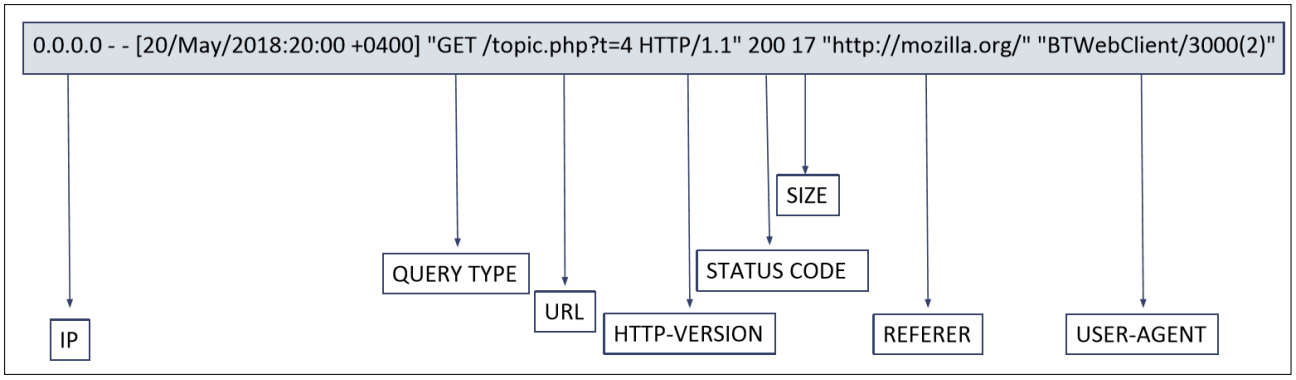


Рисунок 4.3 – Приклад розділення запитів

Після такого розподілення маємо на руках список маркерів, які можуть бути присутні в запиті. Його називають вектором або словником ознак. Приклад такого словника зображений на рис. 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

Отриманий словник буде «пропускати» запити крізь себе для формування feature-vector'a. Це бінарний вектор, що містить у собі інформацію щодо наявності заданих маркерів у тестовому запиті. Приклад такого вектора для даного словника (рис 4.4) зображений на рис. 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

Якщо в результаті навчання нейронна мережа добре розпізнає приклади з навчальної множини, але не набуває властивість узагальнення, тобто не розпізнає або погано розпізнає будь-які інші приклади, крім навчальних, то кажуть, що мережа перенавчилася.

Перенавчання – це результат надмірної підгонки мережі до тренувальних прикладів. Одна з опцій задля боротьби з перенавчанням нейронної мережі – розподіл даних з тренувального набору на дві частини (тренувальну і тестову). На тренувальній частині проводиться навчання мережі. На тестовому наборі даних відбувається тестування моделі. Множини повинні бути непересічними.

Параметри побудованої моделі змінюються кожного кроку, проте значення цільової функції знижується саме на тренувальній вибірці.

Розбиваючи множину на дві ми можемо побачити зміну помилки прогнозу на тестовій частині вибірки водночас зі спостереженнями над тренувальною множиною.

Спочатку кількість ітерацій значення помилки прогнозу зменшується на тренувальній та тестовій множинах водночас. Проте починаючи з певного етапу помилка на тестовій частині починає зростати, при цьому помилка на тренувальній множині продовжує зменшуватися. Ця точка вважається закінченням навчання, а з неї нейронна мережа починає перенавчатися.

Для нашої нейронної мережі був вибраний один із можливих варіантів вирішення проблеми перенавчання: введення так званих спроб. Розділивши вибірку на тренувальну і тестову множини, ми можемо спостерігати за процесом реального навчання нейронної мережі (рис. 4.6). Введення спроб, а саме незалежних прогонів нейронної мережі ми зможемо обрати з них ту, що дає нам найменшу тестову похибку.

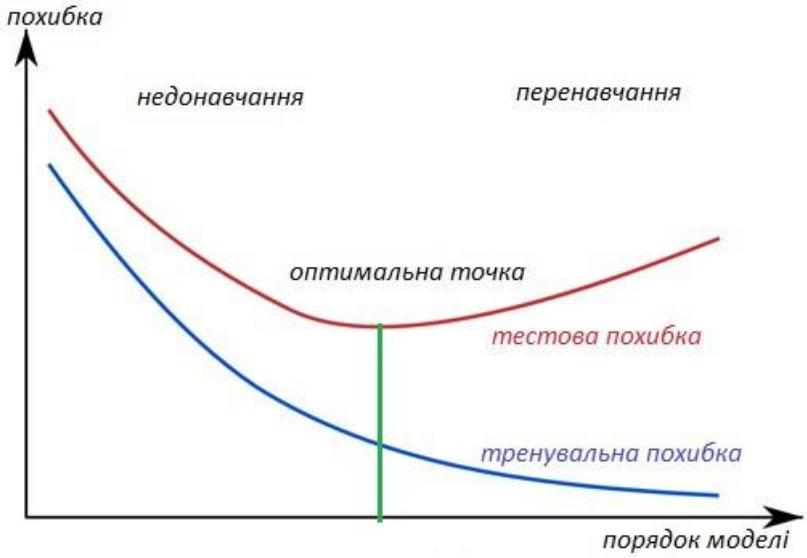


Рисунок 4.6 – Графік тренувальної та тестової похибок у процесі навчання

Для обрання оптимальної кількості спроб прогонів нейронної мережі було виміряні тестові похибки та час виконання роботи алгоритму для різної кількості спроб. Результати можна побачити у таблиці 4.1.

Таблиця 4.1 – Залежність похибки та часу роботи від кількості спроб

| 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 |

Можемо побачити, що після 10 спроб значне збільшення часу виконання алгоритму не дає нам бажаної переваги в його точності. Саме тому, для нашої нейронної мережі було зафіксовано 10 спроб.

Для пришвидшення роботи алгоритмів було вирішено розділити його по процесам завдяки модулю multiprocessing. Паралелізація проводилася розділенням даних на окремі файли, так що окремий процес працював зі своїм файлом. На таблиці 4.2 можна побачити залежність часу відпрацювання алгоритму та тестової похибки від кількості процесів (загальна довжина вибірки – 1000):

Таблиця 4.2 – Залежність похибки та часу роботи від кількості процесів

| 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 |

## 4.4 Результати досліджень

Після проведених досліджень та аналізу була отримана нейронна мережа для класифікації даних сервера з метою виявлення наявності DDoS-атак. Вона складається з одного прихованого шару розміром з подвоєний вхідний шар. Функцією активації для прихованого шару була обрана сигмоїда, а для вихідного – Softmax. Було зафіксована кількість епох у розмірі десяти, для досягнення бажаної точності, та кількість спроб у розмірі десяти, задля запобігання перенавчання. Також навчальна вибірка була розділена у відношенні 70/30 задля вирішення цієї ж проблеми. Отримана мережа зображена на рис. 4.7.На вхід нейронна мережа отримує бінарний вектор приналежності ознак, який утворюється за допомогою словника ознак. Словник, в свою чергу, формується на етапі обробки даних шляхом розбиття запиту та аналізу його компонент.

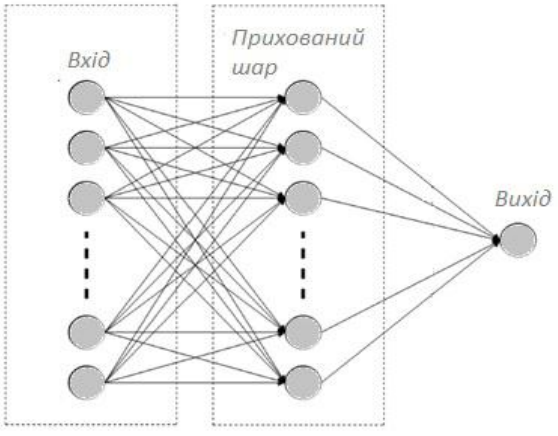


Рисунок 4.7 – Архітектура побудованої нейронної мережі

У отриманому класифікаторі була проаналізована залежність помилок першого роду (хибне прогнозування класифікатором атаки), другого роду (хибне прогнозування класифікатором відсутності атаки) та часу відпрацювання алгоритму в залежності від довжини тренувальної вибірки. Результати досліджень відображені у табл. 4.3.

Таблиця 4.3 – Результати досліджень

| 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 |

Розглянемо залежність помилки першого роду від довжини вибірки графічно. Результати відображені на рис. 4.8.

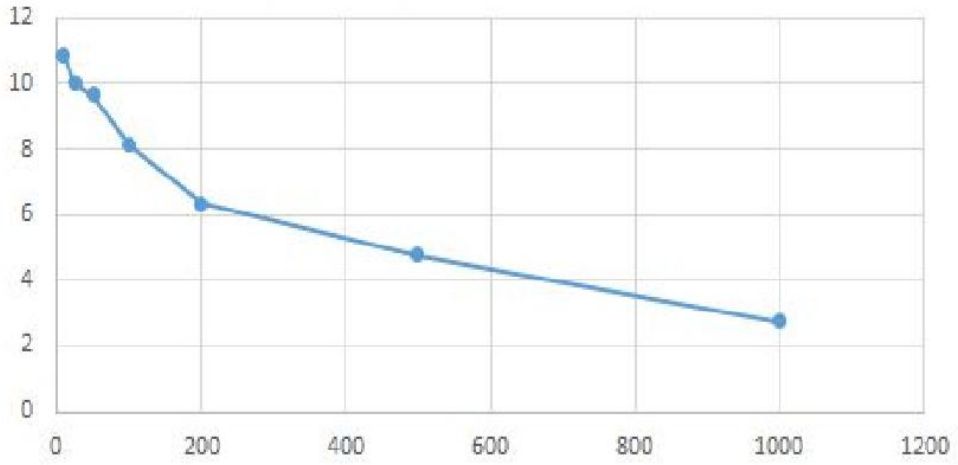


Рисунок 4.8 – Залежність помилки першого роду від довжини вибірки

Далі розглянемо залежність помилки другого роду від довжини вибірки графічно. Результати відображені на рис. 4.9.

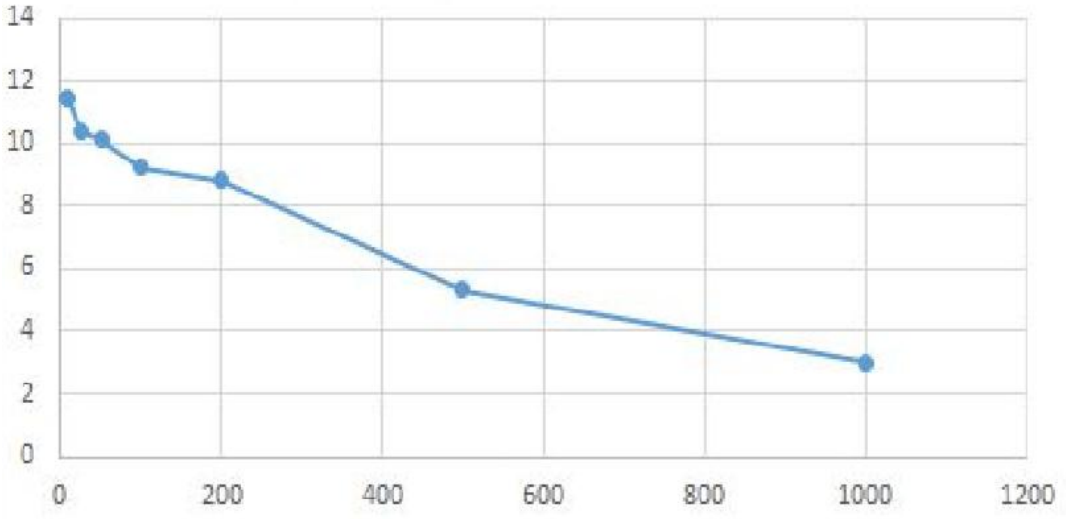


Рисунок 4.9 – Залежність помилки другого роду від довжини вибірки

Проте збільшення довжини вибірки впливає на час роботи алгоритму. Розглянемо залежність часу відпрацювання алгоритму від довжини вибірки графічно. Результати відображені на рис. 4.10.

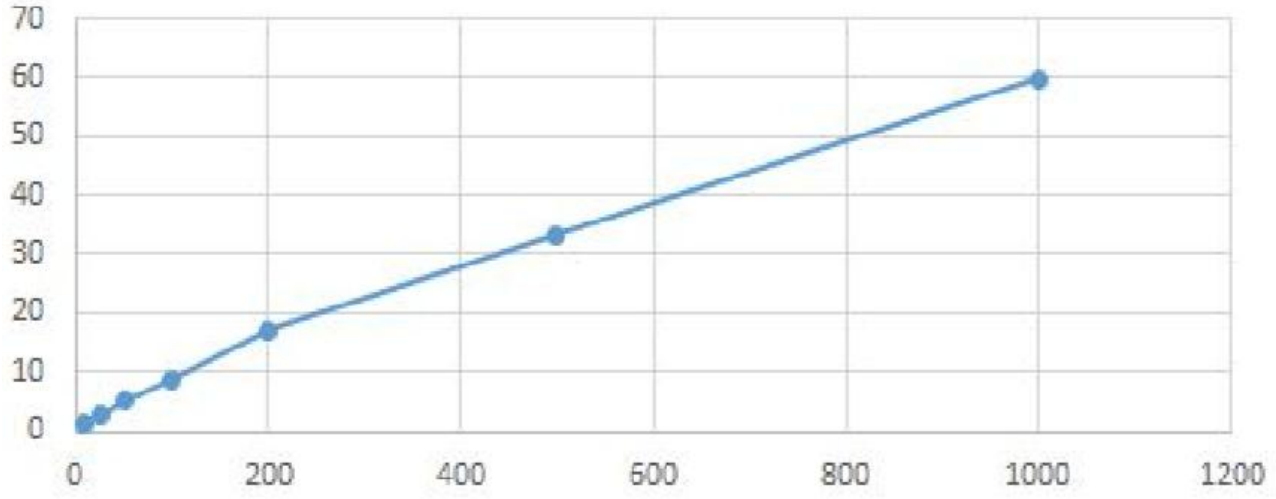


Рисунок 4.10 – Залежність часу відпрацювання алгоритму від довжини вибірки

## 4.5 Ефективність, новизна та практична цінність досліджень

Універсальний та ефективний захист від DDoS-атак – дуже складне і часто надзвичайно незрозуміле питання. Залежно від різних векторів, цілей і технічних умов клієнта для найбільш ефективного захисту підходять певні рішення.

Суворе розміщення, установка і регулювання технологій фільтрування повинні бути узгоджені відповідно до раніше проведеного аналізу. Крім скрупульозного вимірювання атак, призначається відповідна захисна технологія відповідно до місця стикування, де «засмічений» DDoS-ом трафік передається алгоритмічно фільтрує механізмам. Зазвичай, пропоновані комерційні рішення захисту є фінансово витратними та малоефективними.

Сервер, що працює в Інтернеті, – це комплекс, що складається з великого числа компонентів: апаратної платформи, операційної системи, бізнес-додатків, допоміжних програм, мережевої інфраструктури. Щоб зробити додаток недоступним з Інтернету, не обов’язково виводити з ладу саме його – удар може бути спрямований на будь-який з перерахованих елементів. Один з варіантів проведення DDoS-атаки – вивести з ладу канал в Інтернеті, передаючи по ньому так багато «сміттєвих» даних, щоб користувачі не могли дочекатися відповіді.

Така атака є найбільш розповсюдженою та руйнівною, саме тому на захист від такого типу атак було спрямоване наше дослідження.

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.

# 5 АНАЛІЗ МОЖЛИВИХ ВПРОВАДЖЕНЬ

Атака на сервер може проводитися з різними мотивами, але мета одна – вивести ресурс з нормального робочого стану. Зараз з’явився ddos бізнес, в якому професійні хакери виступають в ролі посередника між двома конкурентами. У жорстких ринкових умовах захист сайту від ddos атак стає необхідністю. Грамотне проектування інфраструктури та додатків, які використовуються для надання інтернет-сервісів, дозволяє протистояти невеликим нападам або масштабним атакам в їх початку.

DdoS-атаки стали робочим інструментом в конкуренції і в кібер-війнах. Їх жертвами стає не тільки середній і малий бізнес, а й більш масштабні системи, наприклад:

– соціальні мережі;

– ЗМІ;

– енергетичні системи;

– державні інтернет-ресурси;

– транспортні системи;

– фінансові системи;

– інтернет-магазини, тощо.

Саме тому системи захисту від DDoS потрібні дуже багатьом – переважній більшості. Отриманий в результаті роботи класифікатор може бути вбудований у будь-яку систему, що стикається з Інтернетом, та виконувати функцію фільтра користувацьких запитів. Такий захист значно зменшить вірогідність пропуску DDoS-атаки та здатен реагувати на подібні запити у незначний проміжок часу.

# ВИСНОВКИ

В ході виконання даної атестаційної роботи була розглянута задача виявлення DDoS-атак використовуючи запити, отримані з сервера. Була проведена робота по аналізу літератури, пов’язаної з алгоритмами машинного навчання та нейронних мереж, та вилученням з неї необхідних матеріалів.

Проведено системний аналіз предметної області з метою обрання методів аналізу запитів на предмет ведення атаки. Стандартні методи аналізу статистики не дозволяють виявляти невідомі раніше атаки, тому в якості механізму вирішення даної проблеми активно виступають нейронні мережі.

Були зібрані тренувальні дані необхідні для нейронної мережі, та тестова вибірка для її подальшої класифікації та розроблено ряд алгоритмів, спрямованих на обробку даних, а саме розбиття запиту на компоненти та їх аналізу, виділення ознак, притаманних певним компонентам, та нормуванням їх для подальшої класифікації нейронною мережею.

Вивчаючі інструменти програмної розробки, для побудови класифікатора була обрана високорівнева мова програмування Python, яка містить у собі бібліотеки, що полегшують розробку програмного забезпечення.

Розроблено механізм, що дозволяє класифікувати дані сервера з метою виявлення наявностіDDoS-атаки. Проведено експерименти для підбору оптимальних параметрів нейронної мережі.

На основі реалізованого програмного забезпечення було проведено ряд експериментів, змінюючи обсяг вибірки, та проаналізовано зміни помилок першого та другого роду та час відпрацювання алгоритму.

Отримане програмне забезпечення може бути вбудовано у будь-яку систему, пов’язану з Інтернетом, та виконувати функцію фільтра користувацьких запитів. Даючи досить точні результати за короткий проміжок часу класифікатор придатний для використання як автономна модель класифікації даних, так і у більш складних системах, в якості їх компонента.

# ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. Краткий обзор алгоритмов интеллектуального анализа данных. URL : http://www.intuit.ru/studies/courses/2312/612/lecture/13270 (дата звернення: 12.10.2019).

2. Прикладная статистика: Классификация и снижение размерности / С.А. Айвазян, В.М. Бухштабер, И.С. Енюков, Л.Д. Мешалкин. // Финансы и статистика. 1989. С. 123–125.

3. Черный ящик. URL : https://ru.wikipedia.org/wiki/Black_box (дата звернення: 15.10.2019).

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: 11.11.2019).

6. Kirichenko L., Radivilova T., Carlsson A. Detecting cyber threats through social network analysis: short survey // SocioEconomic Challenges. 2017. № 1. P. 20–34.

7. Штучні нейронні мережі та їх класифікація. URL : https://evergreens.com.ua/ua/articles/neural-network.html (дата звернення: 01.10.2019).

8. Методы машинного обучения для классификации и прогнозирования. URL : http://www.intuit.ru/studies/courses/6/6/lecture/182 (дата звернення: 01.10.2019).

9. Machine Learning. URL : https://www.it.ua/knowledge-base/technology-innovation/machine-learning (дата звернення: 21.11.2019).

10. Машинное обучение. URL : https://postnauka.ru/courses/74896 (дата звернення: 02.10.2019).

11. Types of Machine Learning Algorithms You Should Know. URL : https://towardssdatasciencee.com/types-of-machine-learning-algorithms-you-should-know-953a08248861 (дата звернення: 21.11.19).

12. Навчання з учителем, або контрольоване навчання. URL : https://uk.wikipedia.org/wiki/Навчання_з_учителем (дата звернення: 06.10.2019).

13. Типология задач для машинного обучения без учителя. URL : http://www.machinelearning.ru/wiki/index.php?title=Обучение_без_учителя (дата звернення: 22.10.2019).

14. Введение в Reinforcement Learning, обучение с подкреплением. URL : https://datascience.org.ua/articles/vvedenie-v-reinforcement-learning-ili-obuchenie-s-podkrepleniem/ (дата звернення: 22.10.2019).

15. Глубокое обучение, обработка естественного языка и представление данных. URL: http://colah.github.io/posts/2014-07-NLP-RNNs-Representations (дата звернення: 01.11.2019).

16. Нейронные сети – математический аппарат. URL : https://basegroup.ru /community/articles/math (дата звернення: 15.11.2019).

17. Класификация предложений с помощью нейронных сетей. URL : https://habr.com/ru/company/meanotek/blog/256593/ (дата звернення: 30.11.2019).

18. Горбань А. Н. Обобщенная аппроксимационная теорема и вычислительные возможности нейронных сетей // Сибирский журнал вычислительной математики. 1998. Т. 1, № 1. С.12–24.

19. Кириченко Л. О., Радивилова Т. А., Барановский А. А. Обнаружение киберугроз с помощью анализа социальных сетей // International journal: Information technologies & knowledge volume. 2017. № 1. С. 23–48.

20. Краткий обзор объектно ориентированного языка Python. URL : https://www.helloworld.ru/texts/comp/lang/python(дата звернення: 28.11.2019).