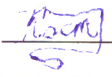


Міністерство освіти і науки України
Державний університет «Житомирська політехніка»
Факультет інформаційно-комп'ютерних технологій
Кафедра комп'ютерної інженерії та кібербезпеки

ПОЯСНЮВАЛЬНА ЗАПИСКА

до випускної кваліфікаційної роботи бакалавра
на тему: «Проект підсистеми аналізу мережевого тра-
фіку для виявлення DDoS-атак з використанням техно-
логій штучного інтелекту»

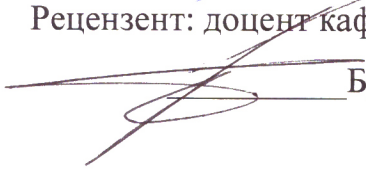
Виконав: студент 4-го курсу групи КБ-21-1
спеціальності 125 «Кібербезпека»

 Іщук К. О.

Керівник: доцент кафедри КІ та КБ

 Маєвський О. В.

Рецензент: доцент кафедри КІ та КБ

 Бродський Ю. Б.

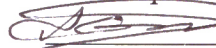
Житомир – 2025

Державний університет «Житомирська політехніка»
Факультет інформаційно-комп'ютерних технологій
Кафедра комп'ютерної інженерії та кібербезпеки
Спеціальність 125 «Кібербезпека»
Освітня програма «Кібербезпека»

ЗАТВЕРДЖУЮ

Завідувач

кафедри комп'ютерної
інженерії та кібербезпеки

 Андрій Сфіменко
3 березня 2025 р.

ЗАВДАННЯ

на випускн у кваліфікаційну роботу бакалавра

Студент: Ішук Константин Олександрович

Тема роботи: «Проект підсистеми аналізу мережевого трафіку для виявлення DDoS-атак з використанням технологій штучного інтелекту»

затверджена Наказом університету від 3 березня 2025 р. № 80/с

Термін здачі студентом закінченої роботи 12 червня 2025 р.

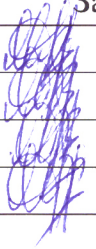
Вихідні дані роботи: Процеси аналізу мережевого трафіку та виявлення аномалій, що характерні для DDoS-атак, в комп'ютерних мережах різного масштабу.

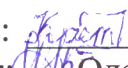
Методи та алгоритми штучного інтелекту, зокрема машинного навчання, для автоматизованого виявлення патернів DDoS-атак у мережевому трафіку, архітектурні рішення та технології їх практичної реалізації в підсистемах кібербезпеки.

Обладнання та програмне забезпечення:

1. ОС: Windows 10 Pro 64bit
2. TensorFlow 2.12.0 з підтримкою CUDA 11.8
3. Python 3.10.12
4. Scikit-learn 1.2.2
5. Visual Studio Code 1.100.3

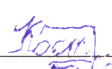
Консультанти з випускної кваліфікаційної роботи із зазначенням розділів, що їх стосуються

Розділ	Консультант	Підпис, дата	
		Завдання видав	Завдання прийняв
1	Маєвський О.В	 11.02.2025	11.02.2025 
2	Маєвський О.В	 16.03.2025	16.03.2025 
3	Маєвський О.В	 11.05.2025	11.05.2025 
4	Маєвський О.В	 25.05.2025	25.05.2025 

Студент:  Костянтин Ішук
 Керівник:  Олександр Маєвський

Календарний план

№ з/п	Назви етапів виконання випускної кваліфікаційної роботи	Термін виконання	Примітка
1	Постановка завдання (вибір теми, погодження з науковим керівником)	11.02.2025 – 17.02.2025	Виконав
2	Формування вступу та структури роботи	18.02.2025 24.02.2025	Виконав
3	Аналіз літературних джерел і сучасного стану проблеми	25.02.2025 10.03.2025	Виконав
4	Дослідження методів виявлення DDoS-атак та можливостей AI	11.03.2025	Виконав
5	Теоретичне обґрунтування використання методів ML/AI для аналізу трафіку	17.03.2025	Виконав
6	Проектування підсистеми аналізу мережевого трафіку	25.03.2025 07.04.2025	Виконав
7	Реалізація та інтеграція алгоритмів машинного навчання	08.04.2025 21.04.2025	Виконав
8	Проведення експериментів, тестування системи	22.04.2025 28.04.2025	Виконав
9	Аналіз результатів, оцінка ефективності	29.04.2025	Виконав
10	Написання висновків	05.05.2025	Виконав
11	Остаточне редагування, перевірка, підготовка до захисту	06.05.2025 12.05.2025	

Студент:  Костянтин Ішук

Керівник:  Олександр Маєвський

АНОТАЦІЯ

У межах кваліфікаційної роботи розроблено проєкт підсистеми аналізу мережевого трафіку для виявлення DDoS-атак із використанням технологій штучного інтелекту. Пояснювальна записка містить 85 сторінок, включає 15 рисунків, 41 найменування у списку використаних джерел, 1 таблицю та додатки.

Метою роботи є розробка проєкту підсистеми аналізу мережевого трафіку для автоматизованого виявлення DDoS-атак з використанням методів машинного навчання та штучного інтелекту, що забезпечить підвищення ефективності захисту мережевої інфраструктури від сучасних кіберзагроз.

У роботі визначено основні завдання для аналізу сучасних типів DDoS-атак, проаналізовано базові методи штучного інтелекту, розглянуто особливості побудови нейронних мереж для задач кібербезпеки. Показано розробку архітектури підсистеми, реалізовано програмну модель на мові Python із застосуванням TensorFlow/Keras. Наведено результати тестування, які підтверджують високу точність виявлення атак та низький рівень хибних спрацьовувань.

Ключові слова: DDoS-АТАКИ, ШТУЧНИЙ ІНТЕЛЕКТ, МАШИННЕ НАВЧАННЯ, НЕЙРОННІ МЕРЕЖІ, КІБЕРБЕЗПЕКА, АНАЛІЗ МЕРЕЖЕВОГО ТРАФІКУ, ВИЯВЛЕННЯ АНОМАЛІЙ.

					KKIKB.KP.B – 125 – 06 – ПЗ			
Змн.	Арк.	№ докум.	Підпис	Дата				
Розроб.		Іщук К.О.			Проект підсистеми аналізу мережевого трафіку для виявлення DDoS-атак з використанням технологій штучного інтелекту	Лім.	Арк.	Аркуші
Керівник		Маєвський О.В.					4	84
Реценз.		Бродський Ю.Б.				ФІКТ, Гр. КБ-21-1		
Н. контр.		Маєвський О.В.						
Затверд.		Єфіменко А.А.		20.06.25				

ANNOTATION

As part of the qualification work, a project of a network traffic analysis subsystem for detecting DDoS attacks using artificial intelligence technologies was developed. The explanatory note consists of 85 pages, includes 15 figures, 55 references, 1 table and appendices.

The aim of the work is to develop a project of a network traffic analysis subsystem for automated detection of DDoS attacks using machine learning and artificial intelligence methods, which will increase the effectiveness of protecting network infrastructure from modern cyber threats.

The paper identifies the main tasks for analyzing modern types of DDoS attacks, analyzes the basic methods of artificial intelligence, and considers the features of building neural networks for cybersecurity tasks. The development of the architecture of the subsubject is shown, and the software model is implemented in Python using TensorFlow/Keras. The test results are presented, which confirm the high accuracy of attack detection and low false alarm rate.

Keywords: DDoS ATTACKS, ARTIFICIAL INTELLIGENCE, MACHINE LEARNING, NEURONIC NETWORKS, CYBER SECURITY, NETWORK TRAFFIC ANALYSIS, ANOMALY DETECTION.

		Іцук.К.О.			ККІКБ.КР.Б – 125 – NN – ПЗ	Арк.
		Маєвський О.В.				5
Змн.	Арк.	№ докум.	Підпис	Дата		

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ	8
ВСТУП.....	10
РОЗДІЛ 1 ДОСЛІДЖЕННЯ ТА АНАЛІЗ СУЧАСНОГО СТАНУ	
ПРОБЛЕМИ DDoS АТАК.....	12
1.1 Сутність і класифікація DDoS-атак	12
1.2 Огляд сучасних систем моніторингу та аналізу мережевого трафіку	16
1.3 Роль і можливості штучного інтелекту у виявленні DDoS-атак.....	17
1.4 Аналіз наукових підходів до застосування AI для задач кібербезпеки...	19
Висновки до першого розділу	22
РОЗДІЛ 2 ТЕОРЕТИЧНІ ОСНОВИ ШТУЧНОГО ІНТЕЛЕКТУ В ЗАДАЧАХ	
КІБЕРБЕЗПЕКИ	23
2.1 Основні поняття штучного інтелекту та підходи до побудови інтелектуальних систем.....	23
2.2 Методи машинного навчання	25
2.3 Архітектура та типи нейронних мереж.....	31
2.4 Роль шарів та активаційні функції.....	37
2.5 Механізми самонавчання та адаптивності нейромереж.....	38
2.6 Методи регуляризації та боротьба з перенавчанням	39
2.7 Використання нейронних мереж для виявлення аномалій у трафіку....	41
Висновки до другого розділу	43
РОЗДІЛ 3 ПРОЕКТУВАННЯ ПІДСИСТЕМИ АНАЛІЗУ МЕРЕЖЕВОГО	
ТРАФІКУ	44
3.1 Архітектура та структура підсистеми	45
3.2 Вибір та обґрунтування алгоритмів машинного навчання та мови програмування	47
3.3 Інтеграція алгоритмів ШІ у систему аналізу трафіку	48
3.4 Опис програмної реалізації.....	49

3.5 Опис модуля для генерації Netflow трафіку	52
Висновки до третього розділу	55
РОЗДІЛ 4 ТЕСТУВАННЯ ТА ОЦІНКА ПОБУДОВАНОЇ ПІДСИСТЕМИ	
АНАЛІЗУ ТРАФІКУ	56
4.1 Організація експерименту	56
4.2 Проведення тестування розробленої підсистеми	57
4.3 Аналіз ефективності надійності та переваг підсистеми	60
Висновки до четвертого розділу	61
ВИСНОВКИ	62
СПИСОК ВИКОРИСТАНОЇ ЛІТЕРАТУРИ.....	63
ДОДАТКИ.....	67

		Ицук.К.О.			ККІКБ.КР.Б – 125 – NN – ПЗ	Арк.
		Маєвський О.В.				7
Змн.	Арк.	№ докум.	Підпис	Дата		

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ

Таблиця.1

Умовне позначення	Повна назва	Пояснення
ОС	Операційна система	Програмне забезпечення, що управляє апаратними ресурсами комп'ютера та забезпечує роботу програм.
ПЗ	Програмне забезпечення	Сукупність програм, які виконують певні функції на комп'ютері або в мережі.
DDoS	Distributed Denial of Service	Розподілена атака відмови в обслуговуванні, спрямована на перевантаження мережеских ресурсів.
IDS	Intrusion Detection System	Система виявлення вторгнень, що моніторить мережеский трафік для виявлення підозрілої активності.
IPS	Intrusion Prevention System	Система запобігання вторгненням, яка не лише виявляє, а й блокує шкідливий трафік.
TCP	Transmission Control Protocol	Протокол управління передачею, що забезпечує надійну передачу даних у мережі.
UDP	User Datagram Protocol	Протокол передачі датаграм без встановлення з'єднання, швидкий, але ненадійний.
ICMP	Internet Control Message Protocol	Протокол керування повідомленнями в мережі, використовується для діагностики та управління.
SSL/TLS	Secure Sockets Layer / Transport Layer Security	Протоколи забезпечення безпечного зв'язку через мережу.
HTTP	HyperText Transfer Protocol	Протокол передачі гіпертексту, основа роботи веб-сайтів.
IoT	Internet of Things	Інтернет речей — мережа фізичних пристроїв з можливістю обміну даними.
AI	Artificial Intelligence	Штучний інтелект — технології, що імітують людський інтелект у машинах.
ML	Machine Learning	Машинне навчання — підгалузь AI, що навчає моделі на даних для виконання завдань.

CNN	Convolutional Neural Network	Згортова нейронна мережа, ефективна для обробки просторових даних.
RNN	Recurrent Neural Network	Рекурентна нейронна мережа, призначена для обробки послідовних даних.
LSTM	Long Short-Term Memory	Тип RNN, що запам'ятовує довгострокові залежності в даних.
GRU	Gated Recurrent Unit	Спрощена альтернатива LSTM з меншою кількістю параметрів.
GAN	Generative Adversarial Network	Генеративна змагальна мережа, що генерує синтетичні дані для навчання моделей.
PCA	Principal Component Analysis	Метод зменшення розмірності даних, що виділяє основні компоненти.
KNN	K-Nearest Neighbors	Метод класифікації на основі найближчих сусідів у просторі ознак.
SVM	Support Vector Machine	Метод опорних векторів для класифікації та регресії.
DBSCAN	Density-Based Spatial Clustering of Applications with Noise	Алгоритм кластеризації на основі щільності для виявлення аномалій.
DQN	Deep Q-Network	Глибока нейронна мережа для навчання з підкріпленням.
XAI	Explainable Artificial Intelligence	Пояснюваний штучний інтелект, що забезпечує прозорість рішень моделей AI.
SIEM	Security Information and Event Management	Системи управління інформацією та подіями безпеки.
JSON	JavaScript Object Notation	Формат обміну даними, що використовується для передачі структурованої інформації.

ВСТУП

Сучасний цифровий світ стикається з безпрецедентним зростанням кількості та складності DDoS-атак, що становить серйозну загрозу для функціонування критичної інфраструктури, комерційних організацій та державних установ. За даними останніх досліджень, загальна кількість DDoS-атак за 2024 рік майже досягла 2,5 мільйонів, що на 8,8% більше порівняно з попереднім роком. Ця тенденція демонструє стійке зростання кіберзагроз, адже з 2005 по 2013 роки кількість таких атак зросла на 807% – з приблизно 325 тисяч у першому кварталі 2013 року до майже 3 мільйонів у першій чверті 2022 року.

Особливу тривогу викликає еволюція тактик атакуючих, які тепер зосереджуються не на кількості атак, а на їх ефективності та точності ураження найбільш вразливих точок цілі. Сучасні DDoS-атаки характеризуються багатовекторністю, де понад 40% всіх атак використовують множинні вектори нападу, а 29% з них мають від 2 до 5 векторів одночасно. Середній розмір атак зріс до кількох терабіт на секунду, що створює загрозу навіть для великих інтернет-ресурсів. Атакуючі стали ретельніше готуватися, проводячи розвідку за відкритими даними та використовуючи всі доступні інструменти для виявлення прихованих сервісів жертви.

Традиційні методи виявлення та протидії DDoS-атакам часто виявляються неефективними проти сучасних багатовекторних та адаптивних атак. В умовах, коли 60% нападів тривають менше 10 хвилин, а деякі атаки можуть тривати понад 12 годин, критично важливою стає наявність адаптивних систем захисту, здатних швидко реагувати на зміни в характері атак. Це обумовлює необхідність розробки інноваційних підходів до аналізу мережевого трафіку з використанням технологій штучного інтелекту, які можуть забезпечити більш точне та оперативне виявлення DDoS-атак.

Метою кваліфікаційної роботи є розробка проекту підсистеми аналізу мережевого трафіку для автоматизованого виявлення DDoS-атак з використанням методів

		Ицук.К.О.			ККІКБ.КР.Б – 125 – NN – ПЗ	Арк.
		Маєвський О.В.				10
Змн.	Арк.	№ докум.	Підпис	Дата		

машинного навчання та штучного інтелекту, що забезпечить підвищення ефективності захисту мережевої інфраструктури від сучасних кіберзагроз.

Визначена мета обумовлює наступні **завдання**:

- Проаналізувати сучасні тенденції та методи проведення DDoS-атак, дослідити їх характеристики та вплив на мережеву інфраструктуру;
- Дослідити можливості застосування методів машинного навчання та штучного інтелекту для аналізу мережевого трафіку;
- Розробити архітектуру підсистеми аналізу мережевого трафіку з інтегрованими алгоритмами штучного інтелекту;
- Імплементувати та протестувати розроблену підсистему на тестових даних мережевого трафіку;
- Оцінити ефективність розробленої системи та порівняти її з існуючими рішеннями.

Об'єктом дослідження є процеси аналізу мережевого трафіку та виявлення аномалій, що характерні для DDoS-атак, в комп'ютерних мережах різного масштабу.

Предметом дослідження є методи та алгоритми штучного інтелекту, зокрема машинного навчання, для автоматизованого виявлення патернів DDoS-атак у мережевому трафіку, архітектурні рішення та технології їх практичної реалізації в підсистемах кібербезпеки.

		Іцук.К.О.			ККІКБ.КР.Б – 125 – NN – ПЗ	Арк.
		Маєвський О.В.				11
Змн.	Арк.	№ докум.	Підпис	Дата		

РОЗДІЛ 1 ДОСЛІДЖЕННЯ ТА АНАЛІЗ СУЧАСНОГО СТАНУ ПРОБЛЕМИ DDoS АТАК

1.1 Сутність і класифікація DDoS-атак

Розподілені атаки типу "відмова в обслуговуванні" (DDoS) представляють собою організовані кібератаки, спрямовані на порушення доступності мережевих ресурсів шляхом перевантаження цільових систем фальшивим трафіком. Як зазначається у дослідженні Microsoft,^[1] сутність таких атак полягає у вичерпанні обчислювальних ресурсів серверів через надлишкові запити, що призводить до деградації продуктивності або повного припинення роботи інфраструктури. Сучасні DDoS-атаки характеризуються масштабністю, складністю та багатовекторністю. Їх інтенсивність може сягати терабітів на секунду, що здатне вивести з ладу навіть найбільш захищені інфраструктури. Вразливими до DDoS-атак є будь-які сервіси, доступні через інтернет: від корпоративних веб-сайтів до критичної інфраструктури, банківських систем та хмарних сервісів.

DDoS-атаки мають різноманітні типи, ^[2] які відрізняються за методами реалізації та рівнями мережевої моделі OSI, на яких вони діють. Одним із найбільш поширених типів є атаки на основі об'ємного трафіку, такі як UDP-флуд, ICMP-флуд або атаки на основі підроблених запитів DNS. Ці атаки спрямовані на виснаження пропускної здатності мережі шляхом надсилання великої кількості сміттевого трафіку. Атаки такого типу часто використовують ботнети — мережі зламаних пристроїв, які одночасно генерують надмірний трафік.

Іншим поширеним типом є атаки на рівні протоколів. Прикладом таких атак є SYN-флуд, який використовує особливості трестороннього рукостискання TCP^[5]. Зловмисники надсилають великий обсяг запитів на встановлення з'єднання, але не завершують цей процес. У результаті сервер витрачає ресурси на підтримку напіввідкритих з'єднань, що призводить до його перевантаження. Аналогічним чином працюють атаки на рівні протоколу SSL/TLS, які спрямовані на виснаження криптографічних ресурсів сервера.

		Ицук.К.О.			ККІКБ.КР.Б – 125 – NN – ПЗ	Арк.
		Маєвський О.В.				
Змн.	Арк.	№ докум.	Підпис	Дата		12

Ще одним типом DDoS-атак є атаки на рівні додатків. Вони спрямовані на використання вразливостей у специфічних програмах або службах. Наприклад, атаки HTTP-флуд полягають у надсиланні великої кількості HTTP-запитів до веб-сервера з метою вичерпання його ресурсів. Ці атаки важко виявити, оскільки їхній трафік може виглядати легітимним. Важливим є також врахування мотивації атакуючих. Атаки можуть бути організовані з метою фінансового здириства, як-от DDoS-for-Ransom, або з політичних і ідеологічних мотивів, наприклад, у вигляді кібератак між державами. Інші атаки спрямовані на знищення конкуренції, зокрема в онлайн-бізнесі, або на створення хаосу в інфраструктурі. [4]

Сучасна класифікація DDoS-атак базуються на рівнях моделі OSI, що дозволяє систематизувати різноманітні вектори нападу:

1. Об'ємні атаки (L3/L4) – спрямовані на переповнення мережевих каналів за допомогою UDP-флуду або DNS-посилання, де середній обсяг трафіку перевищує 1 Тбіт/с^[1]. Об'ємні атаки є одними з найпростіших у виконанні, але водночас надзвичайно ефективними. Вони використовують велику кількість скомпрометованих пристроїв для генерації масивного трафіку, який вичерпує пропускну здатність мережі, блокуючи доступ легітимним користувачам.
2. Протокольні атаки (L4) – експлуатують вади в реалізації мережевих протоколів для блокування легітимних з'єднань. Найпоширенішими є TCP SYN Flood та Ping of Death. [6] Під час SYN-флуд атаки зловмисники надсилають великий обсяг запитів на встановлення з'єднання, але не завершують процес трестороннього рукоствисання TCP, змушуючи сервер витратити ресурси на підтримку напіввідкритих з'єднань^[2].
3. Прикладні атаки (L7) – імітують користувацьку поведінку на рівні додатків, що ускладнює їхнє виявлення традиційними методами^[1]. HTTP Flood та Slowloris є класичними прикладами таких атак, які спрямовані на виснаження ресурсів веб-серверів через надсилання специфічних запитів, що потребують значної обробки. [7]

		Ицук.К.О.			ККІКБ.КР.Б – 125 – NN – ПЗ	Арк.
		Маєвський О.В.				
Змн.	Арк.	№ докум.	Підпис	Дата		13

Окрім класифікації за рівнями OSI, DDoS-атаки можна категоризувати за протоколами, що використовуються для їх проведення:

1. UDP-орієнтовані атаки – включають UDP-флуд, DNS-ампліфікацію та NTP-ампліфікацію. В основі цих атак лежить використання протоколу UDP, що не передбачає встановлення з'єднання та перевірки джерела запиту^[2].
2. TCP-орієнтовані атаки – включають SYN-флуд, ACK-флуд та RST-флуд. Ці атаки експлуатують особливості TCP-протоколу, зокрема механізм встановлення з'єднання^[2].
3. ICMP-орієнтовані атаки – включають ICMP-флуд та Smurf-атаки. Вони базуються на надсиланні великої кількості ICMP-пакетів різних типів.
4. Атаки на протоколи прикладного рівня – включають HTTP-флуд, Slowloris, RUDY (R-U-Dead-Yet) та атаки на SSL/TLS. Ці атаки спрямовані на виснаження ресурсів конкретних сервісів через експлуатацію особливостей їх реалізації.

Особливу небезпеку становлять гібридні атаки, які поєднують кілька векторів нападу. Наприклад, комбінація HTTP-флуду з TCP-ресет-пакетами дозволяє обійти базові системи захисту. Еволюція технік проведення DDoS-атак призвела до появи більш складних та витончених підходів: [10]

1. IP Null attack – використовує вразливість в обробці IP-пакетів, встановлюючи нульове значення у полі Protocol заголовка IP. Це дозволяє обходити стандартні фільтри та змушує систему жертви витрачати ресурси на аналіз нестандартних пакетів.
2. Віддзеркалення та ампліфікація – зловмисники надсилають запити на відкриті сервери (DNS, NTP, SSDP) із підробленою IP-адресою жертви. Сервери відповідають на ці запити значно більшим обсягом даних, спрямовуючи їх до цілі. Коефіцієнт ампліфікації може досягати 1:500, що дозволяє зловмисникам з обмеженими ресурсами генерувати потужні атаки.

		Ицук.К.О.			ККІКБ.КР.Б – 125 – NN – ПЗ	Арк.
		Маєвський О.В.				14
Змн.	Арк.	№ докум.	Підпис	Дата		

3. Атаки з повільним вичерпанням ресурсів (Slow and Low) – замість масивного трафіку використовують малопомітні, але постійні запити, що поступово виснажують ресурси системи. Ці атаки важко виявити, оскільки обсяг трафіку залишається в межах норми.

Сучасні методи виявлення поділяються на сигнатурні та аномалієорієнтовані. Сигнатурні системи (наприклад, Snort) використовують шаблони відомих атак, демонструючи високу точність для вже досліджених загроз, проте є неефективними проти нуль-денних експлойтів. Аномалієорієнтовані підходи базуються на аналізі статистичних характеристик трафіку:

- Аналіз пікової інтенсивності – виявлення різких стрибків у кількості запитів;
- Моніторинг географії IP-адрес – ідентифікація незвичайної концентрації джерел трафіку;
- Частотний аналіз мережевих пакетів – виявлення паттернів, характерних для бот-мереж.

Методи здійснення DDoS-атак постійно еволюціонують. Одним із нових підходів є використання віддзеркалення та посилення (amplification). Зловмисники надсилають невеликі запити до відкритих серверів, наприклад, NTP або DNS, підробляючи IP-адресу жертви. Сервер відповідає на ці запити значно більшим обсягом даних, спрямовуючи їх до жертви. Таким чином, атака може бути значно посилена без прямої участі зловмисника.

Вплив DDoS-атак на мережу може бути катастрофічним. Вони призводять до збоїв у роботі критично важливих сервісів, втрати доходів, пошкодження репутації та збільшення операційних витрат. Залежно від тривалості та інтенсивності атаки, підприємства можуть зіткнутися з серйозними фінансовими збитками. Наприклад, атаки на онлайн-магазини під час розпродажів можуть призвести до втрати значної частини прибутку. [8]

		Ицук.К.О.			ККІКБ.КР.Б – 125 – NN – ПЗ	Арк.
		Маєвський О.В.				15
Змн.	Арк.	№ докум.	Підпис	Дата		



Рис. 1.1. Вплив DDoS-атак на мережу

1.2 Огляд сучасних систем моніторингу та аналізу мережевого трафіку

Для захисту від DDoS-атак використовуються різноманітні методи та інструменти. Одним із ключових підходів є використання систем виявлення вторгнень (IDS) і систем попередження вторгнень (IPS), які можуть автоматично виявляти і блокувати підозрілий трафік. Крім того, розгортання спеціалізованих рішень, таких як сервіси захисту від DDoS, дозволяє ефективно розподіляти трафік і мінімізувати вплив атак. Розгортання резервної інфраструктури, географічно розподілених центрів обробки даних і використання балансування навантаження також можуть допомогти зменшити вплив DDoS-атак. Проте найважливішим елементом захисту є постійний моніторинг мережі та оновлення політик безпеки, що дозволяє оперативно реагувати на нові загрози.

Інструменти моніторингу^[3] поділяються на пасивні (Wireshark, tcpdump) та активні (Nagios, Zabbix). Wireshark дозволяє проводити глибокий аналіз пакетів із розшифрування протоколів, що є критично важливим для виявлення прикладних атак. Сучасні комерційні рішення (Cloudflare, Akamai Prolexic) використовують

		Іцук.К.О.			ККІКБ.КР.Б – 125 – NN – ПЗ	Арк.
		Маєвський О.В.				16
Змн.	Арк.	№ докум.	Підпис	Дата		

розподілену архітектуру для фільтрації трафіку на периферії мережі, однак їх ефективність обмежена в умовах гібридних атак із використанням IoT-пристроїв. Ці платформи працюють за принципом "очищення трафіку", перенаправляючи весь вхідний трафік через свої фільтраційні центри, де відбувається відсіювання шкідливих запитів. Варто відзначити, що ефективність комерційних рішень залежить від їхньої здатності адаптуватися до нових типів атак. Провідні постачальники постійно оновлюють свої системи, впроваджуючи машинне навчання та аналітику великих даних для покращення точності виявлення загроз. Однак, навіть найсучасніші комерційні рішення мають обмеження, особливо щодо виявлення нових типів атак (zero-day) та складних, низькоінтенсивних атак.

Перспективним напрямом є застосування SDN (Software-Defined Networking) для динамічного перенаправлення трафіку. Експерименти показують, що комбінація SDN з алгоритмами машинного навчання зменшує час реакції до 2-3 секунд. SDN дозволяє централізовано керувати мережевим обладнанням, що забезпечує гнучкість та оперативність у протидії DDoS-атакам. Інтеграція технологій блокчейн у системи захисту від DDoS пропонує децентралізований підхід до управління безпекою. Блокчейн-платформи можуть використовуватися для створення розподілених реєстрів легітимного трафіку та автоматичного блокування підозрілої активності без необхідності централізованого контролю. Перспективною є також технологія Edge Computing, яка дозволяє обробляти та фільтрувати трафік безпосередньо на периферійних вузлах мережі, знижуючи навантаження на центральну інфраструктуру та підвищуючи швидкість реакції на загрози.

1.3 Роль і можливості штучного інтелекту у виявленні DDoS-атак

Штучний інтелект революціонізує підходи до кібербезпеки через здатність аналізувати великомасштабні дані в реальному часі. Нейронні мережі зі згортковою архітектурою (CNN) демонструють точність 98,7% у класифікації аномального трафіку завдяки автоматичній екстракції ознак із сирцевих даних. Алгоритми кластеризації (K-means, DBSCAN) ефективні для виявлення бот-мереж через аналіз часових інтервалів між запитами.

		Іщук.К.О.			ККІКБ.КР.Б – 125 – NN – ПЗ	Арк.
		Маєвський О.В.				
Змн.	Арк.	№ докум.	Підпис	Дата		17

Ключовою перевагою AI є адаптивність – системи на основі глибокого навчання (LSTM, Transformers) здатні оновлювати свої моделі без перерви в роботі, що критично важливо для протидії еволюціонуючим атакам. Експериментальні дані свідчать, що ансамблеві моделі (Random Forest + Gradient Boosting) зменшують кількість хибно-позитивних спрацьовувань на 37% порівняно з традиційними методами.

Глибоке навчання (deep learning) відкриває нові горизонти для аналізу часових рядів мережевого трафіку. Рекурентні нейронні мережі (LSTM) показують особливу ефективність у виявленні DDoS-атак із змінною динамікою, оскільки здатні запам'ятовувати довгострокові залежності між пакетами. Експерименти з використанням LSTM на датасеті CICDDoS2019^[18] підтверджують здатність моделей класифікувати 17 типів атак з точністю 98,2%, причому інтерпретаційні методи (LIME, SHAP) дозволяють визначити ключові ознаки, такі як аномальна активність TCP-портів або різкі зміни в розподілі розмірів пакетів. Ця інтерпретовність критично важлива для кіберинцидентів, де необхідно швидко ідентифікувати природу загрози.

Однією з найперспективніших інновацій є застосування генеративних змагальних мереж (GAN) для підвищення стійкості моделей до адверсарних атак. Технологія GADoT, яка поєднує GAN із методами перетворення трафіку, демонструє зниження кількості непомічених атак з 60% до 1,8% шляхом генерації синтетичних зразків, що імітують легальний трафік. GAN складаються з двох компонентів – генератора та дискримінатора, які змагаються між собою: генератор створює синтетичні зразки атак, а дискримінатор намагається відрізнити їх від реальних. У процесі такого змагання обидва компоненти покращують свої можливості, що в результаті підвищує загальну ефективність системи виявлення атак.

Ця технологія є особливо важливою щоб протидіяти нульовим атакам (zero-day attacks), які не мають за собою історичних прецедентів і не можуть бути виявлені традиційними сигнатурними методами. Це дозволяє тренувати моделі на більш різноманітних сценаріях, включаючи атаки зі зміненими характеристиками, що унеможливорює їх обхід за допомогою простих модифікацій пакетів.

		Іщук.К.О.			ККІКБ.КР.Б – 125 – NN – ПЗ	Арк.
		Маєвський О.В.				18
Змн.	Арк.	№ докум.	Підпис	Дата		

Важливу роль відіграють гібридні архітектури, які поєднують переваги різних підходів. Наприклад, ансамблеві моделі на основі CNN, LSTM та GRU, запропоновані для SDN-мереж, досягають точності 99,77% при аналізі потокових даних, що пов'язано з їхньою здатністю одночасно виявляти просторові та часові аномалії.

Гібридні системи можуть використовувати:

1. Згорткові нейронні мережі (CNN) – для виявлення просторових паттернів у структурі пакетів
2. Рекурентні нейронні мережі (RNN, LSTM, GRU) – для аналізу часових залежностей у потоці трафіку
3. Автоенкодера – для виявлення аномалій шляхом порівняння реконструйованих даних з оригінальними
4. Графові нейронні мережі (GNN) – для аналізу топологічних особливостей мережевої активності

Такі системи ефективно ідентифікують як об'ємні атаки (наприклад, UDP-флуд), так і складні прикладні атаки (HTTP Slowloris), автоматично адаптуючись до змін у тактиці зловмисників.

Таким чином, ШІ не лише підвищує точність виявлення DDoS-атак, але й забезпечує проактивний захист завдяки можливості прогнозування нових векторів атак на основі історичних даних. Інтеграція цих технологій із сучасними мережевими архітектурами (наприклад, SDN або блокчейн) створює основу для створення автономних систем кібербезпеки, здатних протистояти навіть найскладнішим координуваним атакам

1.4 Аналіз наукових підходів до застосування AI для задач кібербезпеки

Сучасні дослідження зосереджені на створенні гібридних архітектур, які поєднують переваги символічного та субсимволічного ШІ. Наприклад, система DeepDefense використовує онтологічні моделі для формалізації знань про атаки

		Ицук.К.О.			ККІКБ.КР.Б – 125 – NN – ПЗ	Арк.
		Маєвський О.В.				19
Змн.	Арк.	№ докум.	Підпис	Дата		

разом з рекурентними нейронними мережами для аналізу часових рядів. Інноваційний підхід полягає у застосуванні генеративних змагальних мереж (GAN) для синтезу тренувочних даних, що дозволяє покращити якість класифікації для рідкісних типів атак. Архітектури згорткових нейронних мереж (CNN) стали стандартом для обробки структурованих даних, таких як бінарні представлення мережевих пакетів або часові ряди трафіку. Експерименти з використанням ResNet-50 на датасеті CICMalDroid 2020 показали ефективність у виявленні 95% нових сімейств шкідливого ПЗ за рахунок автоматичної екстракції ознак із RAW-даних.

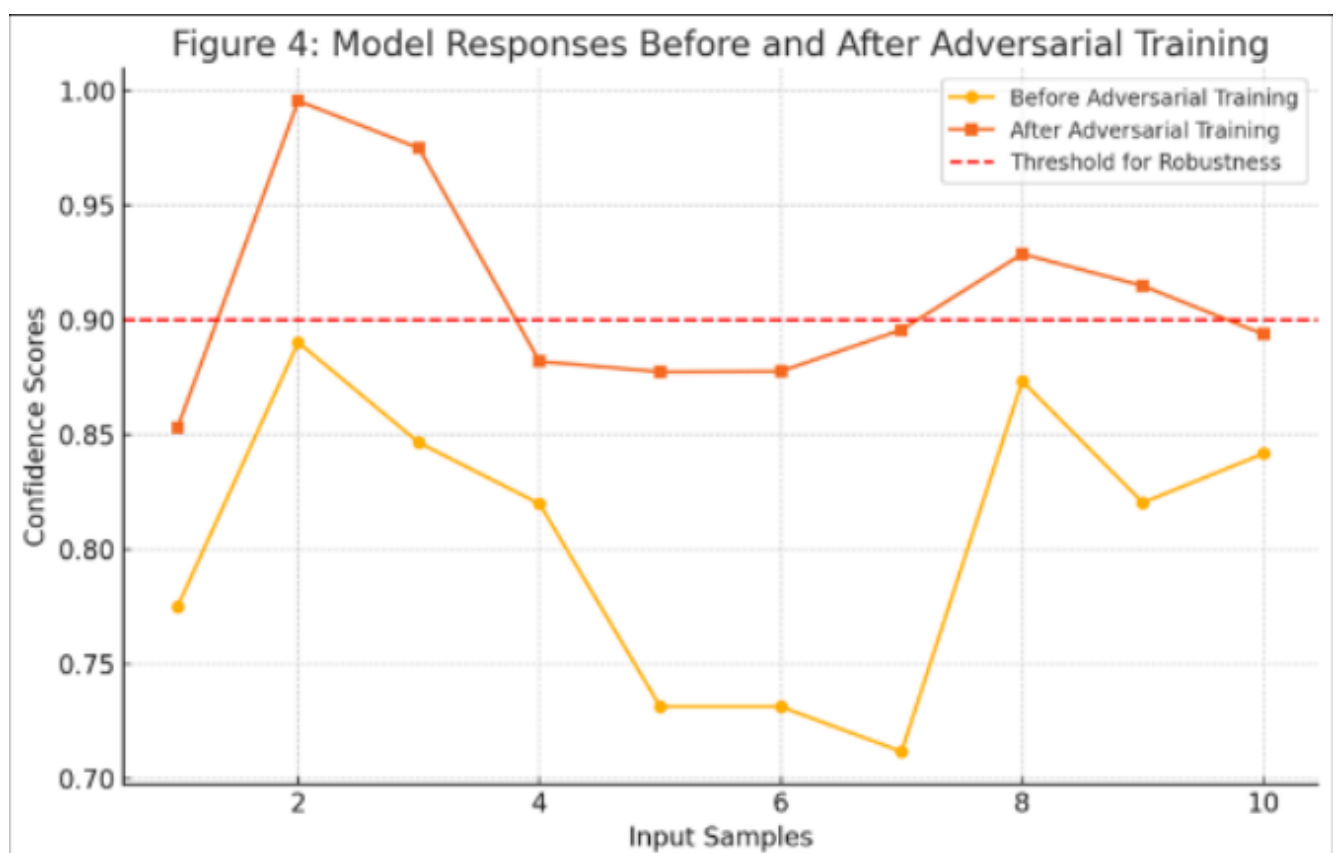


Рис. 1.2. Результати досліджень з використанням ResNet-50

Для аналізу послідовних даних (наприклад, логів або поведінки користувачів) рекурентні мережі (LSTM, GRU) досягають точності 98,7% у прогнозуванні атак типу АРТ, використовуючи механізми пам'яті для виявлення довгострокових залежностей.

Комбінація різних парадигм ІІІ дозволяє подолати обмеження окремих методів. Система DeepCyberDefense, яка поєднує графові нейронні мережі (GNN) для аналізу топології мережі з автоенодерами для виявлення аномалій, демонструє

		Іцук.К.О.			ККІКБ.КР.Б – 125 – NN – ПЗ	Арк.
		Маєвський О.В.				20
Змн.	Арк.	№ докум.	Підпис	Дата		

зниження рівня хибно-позитивних спрацьовувань на 40% порівняно з окремими компонентами.

Сучасні дослідження виявляють уразливості моделей ШІ до спеціально сконструйованих вхідних даних. Експерименти з технікою FGSM (Fast Gradient Sign Method) показали, що штучний шум амплітудою лише 3-5% може знизити точність класифікації malware на 62%. Для протидії таким загрозам розробляються методи адверсарного тренування, де моделі експонуються зразкам, збалансованим між легальними та модифікованими даними. Система AdvSecNet, що використовує генеративно-змагальні мережі (GAN) для синтезу адверсарних прикладів, демонструє підвищення стійкості на 28% у порівнянні з класичними підходами.

Потреба у прозорості роботи моделей ШІ для кібербезпеки призвела до активного розвитку методів XAI. Техніки SHAP (Shapley Additive Explanations) та LIME (Local Interpretable Model-agnostic Explanations) дозволяють візуалізувати ваги ознак, критичних для прийняття рішень. Наприклад, аналіз моделі для детекції фішингу виявив, що 73% рішень базуються на аномаліях у HTML-структурі та розподілі ключових слів. Впровадження онтологічних моделей знань (наприклад, OWL-онтологій для опису тактик MITRE ATT&CK) забезпечує семантичну інтерпретацію результатів, необхідну для інтеграції з системами SIEM.

Важливим напрямом є інтеграція AI з блокчейн-технологіями для створення децентралізованих систем захисту. Проєкт ShieldLedger демонструє зменшення часу координації між вузлами мережі на 45% порівняно з централізованими рішеннями. Однак залишаються відкритими питання енергоефективності та масштабованості таких систем.

Таким чином, сучасний стан досліджень свідчить про перспективність AI-орієнтованих підходів, але вимагає подальшого вдосконалення методів обробки неструктурованих даних та забезпечення стійкості моделей до адверсарних атак.

		Ицук.К.О.			ККІКБ.КР.Б – 125 – NN – ПЗ	Арк.
		Маєвський О.В.				21
Змн.	Арк.	№ докум.	Підпис	Дата		

Висновки до першого розділу

Ми проаналізували сучасні тенденції та методи проведення DDoS-атак, дослідити їх характеристики та вплив на мережеву інфраструктуру і можемо виділити наступне:

Розподілені атаки типу "відмова в обслуговуванні" залишаються однією з найсерйозніших загроз сучасної кіберпросторі, постійно еволюціонуючи та адаптуючись до нових систем захисту. Проведене дослідження демонструє необхідність комплексного підходу до протидії DDoS-атакам, що включає використання традиційних методів моніторингу та передових технологій штучного інтелекту

		Іцук.К.О.			ККІКБ.КР.Б – 125 – NN – ПЗ	Арк.
		Маєвський О.В.				
Змн.	Арк.	№ докум.	Підпис	Дата		22

РОЗДІЛ 2 ТЕОРЕТИЧНІ ОСНОВИ ШТУЧНОГО ІНТЕЛЕКТУ В ЗАДАЧАХ КІБЕРБЕЗПЕКИ

Стрімкий розвиток інформаційних технологій та зростання складності кіберзагроз зумовлюють необхідність застосування інноваційних підходів до забезпечення кібербезпеки. Штучний інтелект стає ключовим інструментом для виявлення, аналізу та протидії кіберзагрозам в реальному часі, дозволяючи автоматизувати моніторинг інформаційних систем та значно підвищувати ефективність захисту критичних даних. Завдяки алгоритмам машинного навчання та нейронним мережам, системи на базі штучного інтелекту можуть не лише виявляти відомі типи вторгнень, але й передбачати потенційні загрози на основі аналізу патернів поведінки та аномалій у мережевому трафіку.

2.1 Основні поняття штучного інтелекту та підходи до побудови інтелектуальних систем

Штучний інтелект являє собою розділ інформатики, який зосереджений на розробці інтелектуальних машин, здатних виконувати завдання, які зазвичай потребують людського інтелекту. Системи штучного інтелекту створені для навчання на досвіді, розпізнавання закономірностей і ухвалення рішень на основі вхідних даних. У контексті кібербезпеки, штучний інтелект дозволяє автоматизувати моніторинг і захист інформаційних систем, знижуючи залежність від людського фактора та підвищуючи ефективність кіберзахисту. [21] Технологія штучного інтелекту охоплює широкий спектр методів, зокрема машинне навчання, обробку природної мови, робототехніку, експертні системи тощо. Мета досліджень штучного інтелекту полягає в тому, щоб створити машини, які можуть міркувати, розуміти та навчатися як люди, і використовувати ці можливості для покращення життя людства та вирішення складних проблем. В сфері кібербезпеки це означає здатність систем самостійно ідентифікувати нові типи загроз, адаптуватися до змінних умов кіберпростору та приймати рішення щодо контрзаходів у реальному часі.

		Ицук.К.О.			ККІКБ.КР.Б – 125 – NN – ПЗ	Арк.
		Маєвський О.В.				
Змн.	Арк.	№ докум.	Підпис	Дата		23

Особливістю штучного інтелекту є те, що здебільшого алгоритм розв'язання проблем невідомий наперед. Точного визначення цієї науки немає, оскільки у філософії не розв'язано питання про природу і статус людського інтелекту. Немає і точного критерію досягнення комп'ютером "розумності", хоча перед штучним інтелектом було запропоновано низку гіпотез, наприклад, тест Тюрінга або гіпотеза Ньюелла-Саймона. Ця невизначеність створює як виклики, так і можливості для розвитку систем кібербезпеки, адже дозволяє експериментувати з різними підходами та методологіями.

За класифікацією Джека Коупленда існує два основні підходи^[21] до розробки штучного інтелекту: низхідний семіотичний та висхідний. Низхідний підхід передбачає створення символічних систем, що моделюють процеси мислення на високому рівні абстракції. Цей підхід широко використовується в експертних системах для кібербезпеки, де знання експертів формалізуються у вигляді правил та процедур.

Детальніша класифікація включає чотири основні підходи до побудови систем штучного інтелекту: логічний, структурний, еволюційний і імітаційний. Логічний підхід базується на Булевій алгебрі та включає обчислення предикатів, де вводяться предметні символи, відношення між ними, квантори існування й загальності. Відносно новий напрямок в логічному підході - це нечітка логіка, яка особливо корисна в задачах кібербезпеки для роботи з невизначеністю та неповною інформацією.

Структурний або біонічний підхід передбачає побудову штучного інтелекту копіюванням людського мозку. Цей підхід лежить в основі розробки нейронних мереж, які активно використовуються для виявлення аномалій у мережевому трафіку. Еволюційний підхід використовує еволюційні алгоритми навчання, де розробляються правила еволюціонування моделі. Такі алгоритми дозволяють системам кібербезпеки адаптуватися до нових типів загроз шляхом еволюції своїх параметрів та структури.

Імітаційний або функціональний підхід є класичним для кібернетики, при цьому виконують імітацію поведінки об'єкта, який сприймають за "чорний ящик". Цей

		Ицук.К.О.			ККІКБ.КР.Б – 125 – NN – ПЗ	Арк.
		Маєвський О.В.				
Змн.	Арк.	№ докум.	Підпис	Дата		24

підхід особливо ефективний для моделювання поведінки зловмисників та передбачення їх дій на основі спостереження за зовнішніми проявами активності в мережі. Машинне навчання є підмножиною штучного інтелекту, яка фокусується на розробці алгоритмів, що дозволяють комп'ютерам навчатися та покращувати свою продуктивність на основі досвіду без явного програмування. У контексті кібербезпеки, методи машинного навчання відіграють ключову роль у створенні моделей, які здатні ідентифікувати підозрілу активність у мережі та автоматично реагувати на потенційні загрози.

Глибоке навчання, у свою чергу, є підмножиною машинного навчання, що використовує багатоваршівні нейронні мережі для моделювання та розуміння складних патернів у даних. Ці моделі навчаються на великих наборах даних, що дозволяє їм виявляти нові загрози, які раніше не зустрічалися в системі. Ієрархічна структура глибокого навчання дозволяє системам поступово вивчати все більш абстрактні представлення даних, від простих ознак до складних концептуальних зв'язків. [22]

Взаємозв'язок між цими рівнями технологій створює потужну основу для сучасних систем кібербезпеки. Штучний інтелект забезпечує загальну парадигму "розумної" поведінки системи, машинне навчання надає конкретні методи для автоматичного вдосконалення продуктивності, а глибоке навчання дозволяє обробляти надзвичайно складні та великі обсяги даних з високою точністю. Ця багаторівнева архітектура особливо важлива для кібербезпеки, де необхідно аналізувати мільйони подій щосекунди та виявляти серед них потенційно небезпечні аномалії.

2.2 Методи машинного навчання

Методи машинного навчання традиційно поділяються на три основні категорії залежно від типу навчальних даних та способу отримання зворотного зв'язку. Контрольоване навчання[30] використовує набори даних з відомими відповідями або мітками, що дозволяє алгоритму навчитися зіставляти вхідні дані з правильними виходами. У контексті кібербезпеки це означає навчання на прикладах відомих атак та нормальної активності для подальшого розпізнавання подібних патернів у реальному трафіку. Контрольовані моделі машинного навчання показують виняткову

		Іцук.К.О.			ККІКБ.КР.Б – 125 – NN – ПЗ	Арк.
		Маєвський О.В.				25
Змн.	Арк.	№ докум.	Підпис	Дата		

ефективність як механізми виявлення DDoS-атак, особливо при використанні методів відбору ознак. Дослідження з використанням наборів даних CICIDS2017, CICIDS2018 та CICDDoS-2019 демонструють, що алгоритм Random Forest досягає найвищої точності 98,9% на датасеті CICIDS2017. На датасеті CICDDoS2019 алгоритми Random Forest та KNN показують високу точність 98,7%, тоді як на датасеті CICIDS2018 найкращі результати показує SVM з точністю 98,7%. Порівняльний аналіз підкреслює ефективність LSTM у досягненні високих показників точності, перевершуючи традиційні методи, такі як SVM та логістична регресія

Неконтрольоване навчання^[31] працює з немаркованими даними, намагаючись виявити приховані структури та закономірності без попередньої інформації про правильні відповіді. Цей підхід особливо корисний для виявлення нових типів загроз, які раніше не були відомі системам кібербезпеки. Алгоритми кластеризації, які відносяться до неконтрольованого навчання, можуть групувати схожі види активності та ідентифікувати аномальні групи, що потенційно можуть представляти кіберзагрози.

Навчання з підкріпленням^[32] являє собою третій тип, де агент навчається приймати рішення через взаємодію з середовищем та отримання винагороди або покарання за свої дії. В системах кібербезпеки цей підхід може використовуватися для розробки адаптивних стратегій захисту, які самостійно оптимізують свою поведінку на основі результативності попередніх дій. Наприклад, система може навчитися автоматично налаштовувати рівні безпеки залежно від поточного рівня загроз.

Алгоритм K-Nearest Neighbors є одним з найпростіших та інтуїтивно зрозумілих методів класифікації, який приймає рішення на основі голосування k найближчих сусідів в просторі ознак.

		Іцук.К.О.			ККІКБ.КР.Б – 125 – NN – ПЗ	Арк.
		Маєвський О.В.				
Змн.	Арк.	№ докум.	Підпис	Дата		26

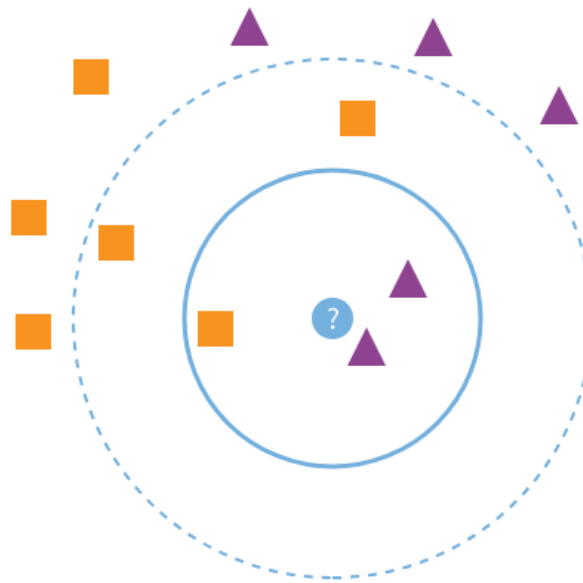


Рис. 2.1. Алгоритм K-Nearest Neighbors

У задачах кібербезпеки KNN може ефективно виявляти аномальні з'єднання, порівнюючи їх характеристики з відомими безпечними та шкідливими з'єднаннями. Перевагою цього методу є його простота та здатність працювати з нелінійними залежностями, однак він може бути повільним при обробці великих обсягів трафіку в реальному часі. [39]

Метод опорних векторів (SVM) будує гіперплощину, яка оптимально розділяє класи в багатовимірному просторі ознак. SVM особливо ефективний для задач бінарної класифікації, таких як розрізнення шкідливого та легітимного трафіку. Використання ядерних функцій дозволяє SVM працювати з нелінійно розділюваними даними, що робить його потужним інструментом для виявлення складних патернів атак.

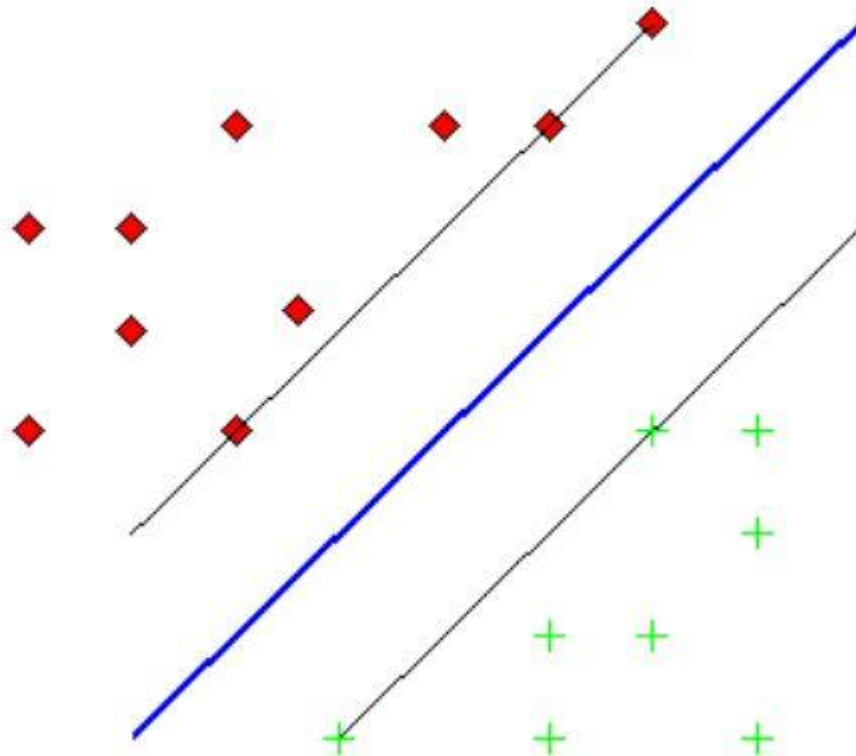


Рис. 2.2. Метод опорних векторів

Алгоритм також має хорошу здатність до узагальнення, що важливо для виявлення варіацій відомих атак. Дерева рішень та випадкові ліси представляють ієрархічні методи класифікації, які будують послідовність бінарних рішень для досягнення кінцевої класифікації. Дерева рішень легко інтерпретуються та можуть ефективно обробляти як числові, так і категоріальні ознаки мережевого трафіку. Випадковий ліс, який є ансамблем дерев рішень, значно покращує стабільність та точність прогнозів, зменшуючи ризик перенавчання. Ці методи особливо корисні для створення правил безпеки, які можуть бути легко зрозумілими адміністраторами системи.

K-means є найпоширенішим алгоритмом кластеризації, який розділяє дані на k кластерів шляхом мінімізації дисперсії всередині кластерів. У контексті кібербезпеки K-means може групувати схожі типи мережевої активності, допомагаючи ідентифікувати аномальні кластери, які можуть представляти нові типи атак.

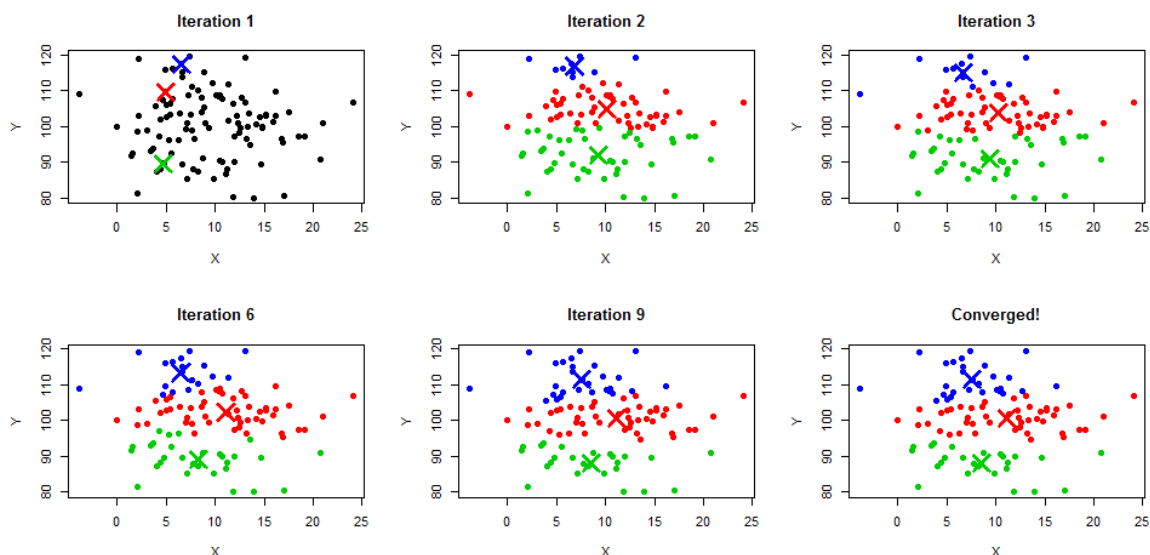


Рис. 2.3. Алгоритм кластеризації K-means

Алгоритм ефективний для великих наборів даних та може працювати в реальному часі, що критично важливо для систем моніторингу мережі.

DBSCAN (Density-Based Spatial Clustering of Applications with Noise) використовує підхід, заснований на щільності, для виявлення кластерів довільної форми та автоматичного виявлення аномалій як точок-викидів. Цей алгоритм особливо корисний для виявлення атак типу "denial of service", які можуть створювати щільні скупчення аномальних з'єднань. DBSCAN не вимагає попереднього задання кількості кластерів та може ефективно обробляти шум у даних, що робить його стійким до помилкових спрацьовувань.

Методи зменшення розмірності, такі як Principal Component Analysis (PCA) та t-SNE, дозволяють візуалізувати та аналізувати високорозмірні дані мережевого трафіку. Ці методи можуть виявляти приховані структури в даних та допомагати в ідентифікації аномальних патернів, які не видимі при аналізі окремих ознак. Зменшення розмірності також допомагає прискорити обробку даних та зменшити вимоги до обчислювальних ресурсів.

Q-learning є фундаментальним алгоритмом навчання з підкріпленням, який навчає агента оптимальній стратегії через оцінку якості різних дій у кожному стані середовища. У системах кібербезпеки Q-learning може використовуватися для розробки адаптивних політик захисту, які автоматично налаштовуються залежно від типу та

інтенсивності поточних загроз. Агент може навчитися оптимально розподіляти ресурси захисту між різними компонентами системи або динамічно змінювати рівні моніторингу.

Deep Q-Network (DQN) розширює традиційний Q-learning, використовуючи глибокі нейронні мережі для апроксимації Q-функції, що дозволяє працювати з високорозмірними просторами станів. У контексті кібербезпеки DQN може навчитися складним стратегіям захисту, аналізуючи великі обсяги даних про стан мережі та результати попередніх дій. Цей підхід особливо ефективний для створення систем автоматичного реагування на інциденти, які можуть приймати рішення на основі поточної ситуації в мережі.

Алгоритми навчання з підкріпленням також можуть використовуватися для моделювання поведінки зловмисників та розробки превентивних заходів. Створюючи середовища, які імітують реальні мережі, можна навчити агентів виявляти оптимальні стратегії атак та розробляти відповідні контрзаходи. Такий підхід дозволяє проактивно готуватися до нових типів загроз та тестувати ефективність різних систем захисту. [32]

Інноваційні підходи включають розробку фреймворків на основі Принципального аналізу компонентів (PCA) для покращеного виявлення розподілених DDoS-атак. Методи попередньої обробки та технік дослідницького аналізу даних відіграють критичну роль в оптимізації продуктивності моделей, дозволяючи більш точно виділяти релевантні ознаки для класифікації. Перспективні дослідження спрямовані на застосування різноманітних технік машинного навчання та моделей глибокого навчання в поєднанні з процедурами моніторингу реального часу для покращення виявлення DDoS-атак у динамічних мережевих середовищах. Адаптація алгоритмів LSTM для виявлення різних типів мережевих атак та їх інтеграція з системами виявлення вторгнень може забезпечити комплексні рішення мережевої безпеки.

		Ицук.К.О.			ККІКБ.КР.Б – 125 – NN – ПЗ	Арк.
		Маєвський О.В.				30
Змн.	Арк.	№ докум.	Підпис	Дата		

2.3 Архітектура та типи нейронних мереж

Перцептрон являє собою найпростішу модель нейронної мережі, що складається з одного нейрона з лінійною функцією активації.

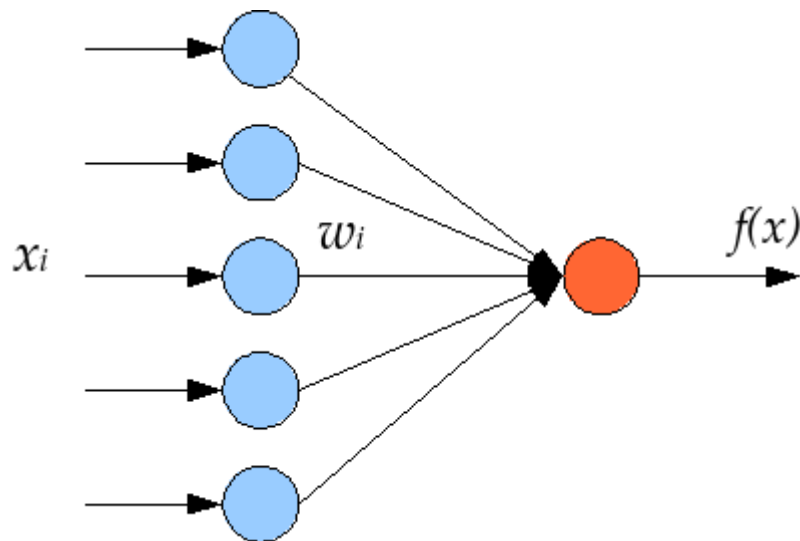


Рис. 2.4. Простий перцептрон

Хоча простий перцептрон може розв'язувати лише лінійно розділювані задачі, його розуміння є фундаментальним для освоєння більш складних архітектур. У задачах кібербезпеки перцептрон може використовуватися для базової бінарної класифікації трафіку, наприклад, для розрізнення HTTP та інших протоколів на основі простих статистичних ознак пакетів.

Багатошарові перцептрони [33] розширюють можливості простого перцептрона, додаючи приховані шари між вхідним та вихідним шарами. Кожен додатковий шар дозволяє мережі вивчати все більш складні та абстрактні представлення вхідних даних. У контексті аналізу мережевого трафіку, перші шари можуть виявляти базові статистичні патерни, такі як розмір пакетів та частота з'єднань, в той час як глибші шари можуть ідентифікувати складні послідовності дій, характерні для певних типів атак.

Універсальна теорема апроксимації гарантує, що багатошарові мережі з достатньою кількістю нейронів у прихованих шарах можуть апроксимувати будь-яку неперервну функцію з довільною точністю. Це робить їх теоретично потужними для моделювання складних залежностей у даних кібербезпеки, де відносини між різними ознаками трафіку та типами загроз можуть бути надзвичайно нелінійними

		Іцук.К.О.			ККІКБ.КР.Б – 125 – NN – ПЗ	Арк.
		Маєвський О.В.				
Змн.	Арк.	№ докум.	Підпис	Дата		31

та складними. Однак практична реалізація цієї можливості вимагає ретельного підбору архітектури та методів навчання.

Прямопоширювальні нейронні мережі (FNN)

Feedforward Neural Networks представляють класичну архітектуру, де інформація рухається в одному напрямку від вхідного шару до вихідного через один або кілька прихованих шарів.

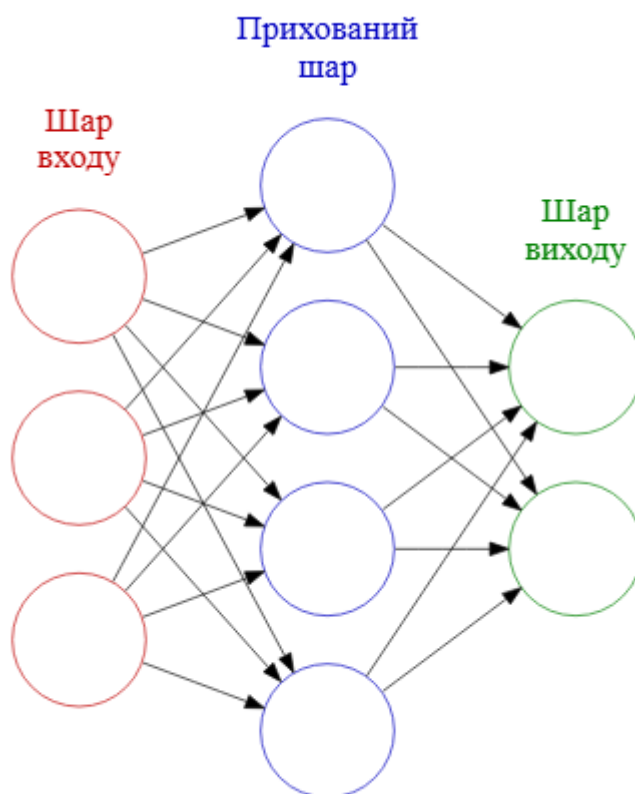


Рис. 2.5. Схема Прямопоширювальної нейронної мережі

У системах кібербезпеки FNN ефективно використовуються для класифікації статичних ознак мережевого трафіку, таких як розмір пакетів, порти призначення, протоколи та статистичні характеристики з'єднань. Ці мережі можуть швидко обробляти векторизовані представлення мережевих сесій та приймати рішення про їх легітимність. Архітектура FNN особливо підходить для задач, де вхідні дані мають фіксований розмір та не залежать від послідовності. Наприклад, для аналізу логів файрволу, де кожен запис може бути представлений як вектор числових ознак, FNN

		Ицук.К.О.			ККІКБ.КР.Б – 125 – NN – ПЗ	Арк.
		Маєвський О.В.				
Змн.	Арк.	№ докум.	Підпис	Дата		32

може ефективно навчитися розрізняти нормальну активність від підозрілої. Простота архітектури також робить FNN швидкими у навчанні та інференції, що критично важливо для систем реального часу. [39]

Оптимізація FNN для задач кібербезпеки включає ретельний підбір кількості шарів та нейронів у кожному шарі. Занадто прості мережі можуть не вловлювати складні патерни атак, в той час як занадто складні можуть перенавчатися на тренувальних даних та погано узагальнювати на нові типи загроз. Використання методів регуляризації та валідації допомагає знайти оптимальний баланс між складністю моделі та її здатністю до узагальнення.

Convolutional Neural Networks [34] спеціально розроблені для обробки даних з просторовою структурою, таких як зображення, але їх принципи успішно адаптуються для аналізу послідовностей мережевого трафіку.

		Ицук.К.О.			ККІКБ.КР.Б – 125 – NN – ПЗ	Арк.
		Маєвський О.В.				
Змн.	Арк.	№ докум.	Підпис	Дата		33

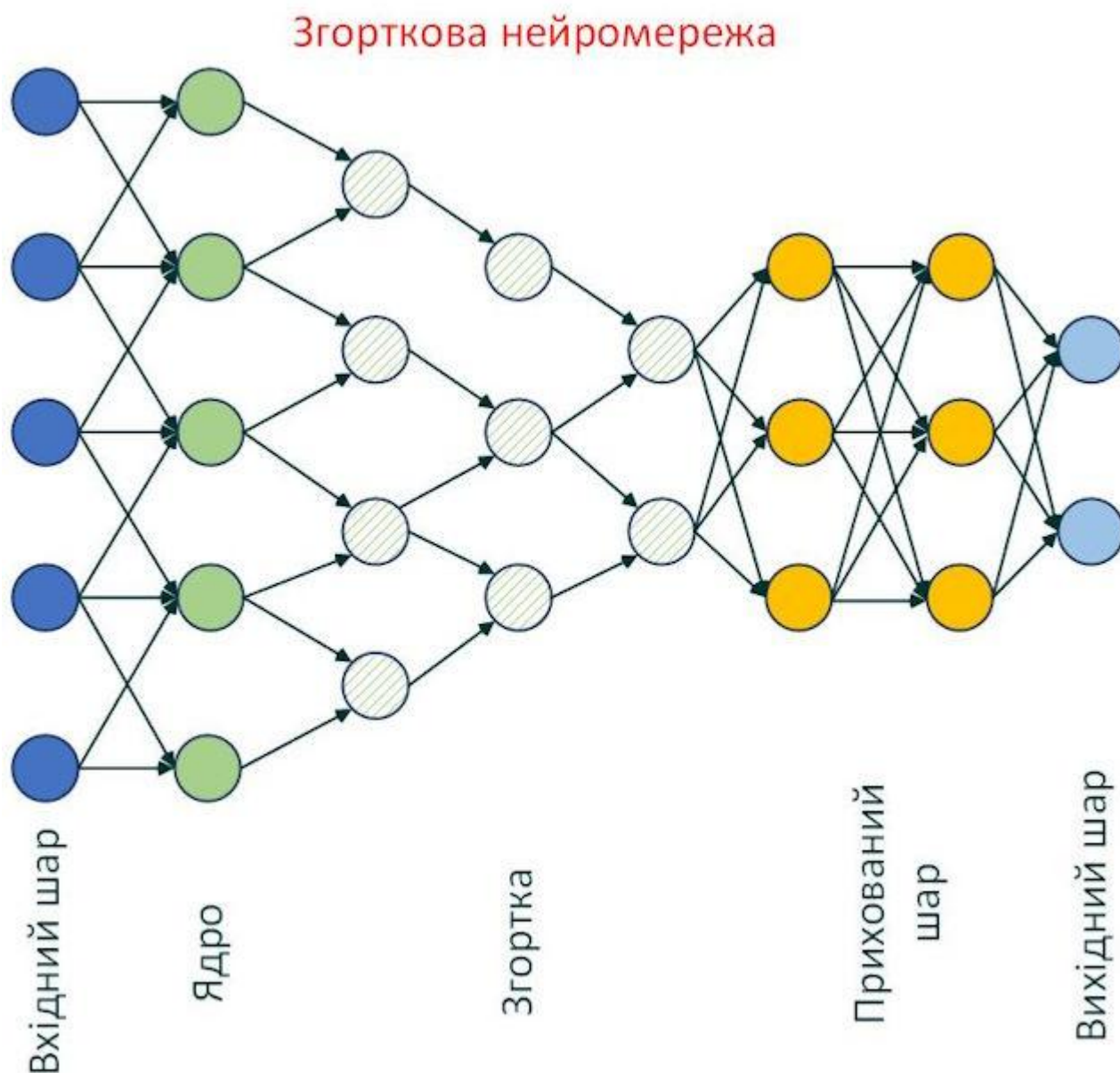


Рис. 2.6. Схема згорткової неймережі

У контексті кібербезпеки CNN можуть аналізувати бінарні представлення пакетів або послідовності системних викликів як одновимірні "зображення", де згорткові фільтри виявляють локальні патерни, характерні для певних типів атак. Згорткові шари використовують принцип розподілених ваг, де один набір фільтрів застосовується до всіх частин вхідних даних, що дозволяє виявляти патерни незалежно від їх положення в послідовності. Це особливо корисно для виявлення сигнатур шкідливого коду або характерних послідовностей команд, які можуть з'являтися в різних частинах мережесесій. Операції пулінгу зменшують розмірність даних та

забезпечують інваріантність до незначних зміщень у патернах. Ієрархічна структура CNN дозволяє будувати все більш абстрактні представлення даних на кожному рівні. Нижні шари можуть виявляти прості локальні ознаки, такі як специфічні байтові послідовності, в той час як вищі шари комбінують ці ознаки для ідентифікації складних патернів атак. Така архітектура особливо ефективна для аналізу payload мережевих пакетів та виявлення вбудованого шкідливого коду.

Рекурентні нейронні мережі (RNN, LSTM, GRU)

Recurrent Neural Networks призначені для обробки послідовностей даних змінної довжини, де поточний вихід залежить не лише від поточного входу, але й від попередньої історії.

Рекурентна нейронна мережа

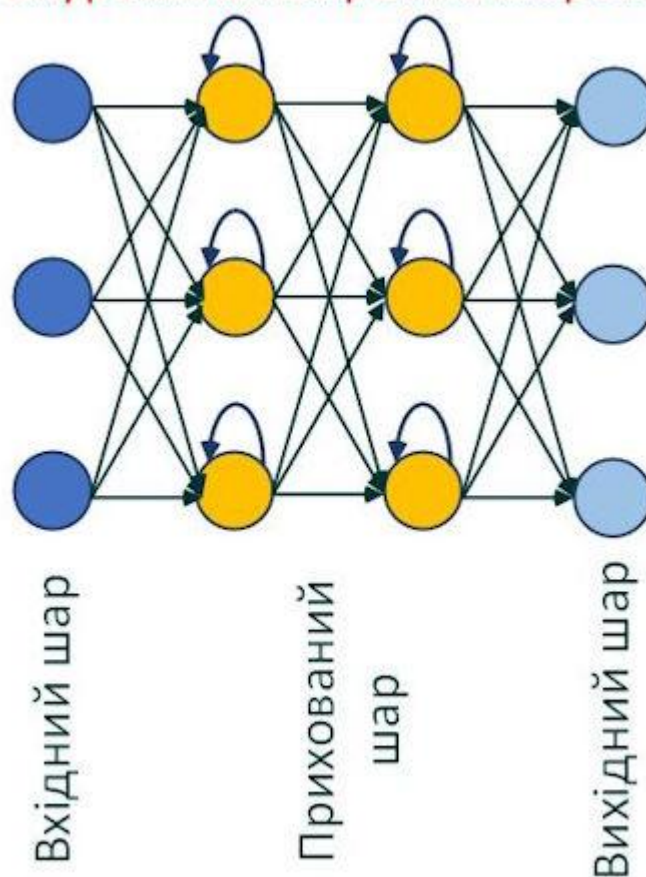


Рис. 2.7. Схема рекурентної нейронної мережі

У задачах кібербезпеки RNN ефективно моделюють часові залежності в мережевому трафіку, виявляючи аномальні послідовності дій або атаки, що розгортаються

протягом тривалого періоду часу. Класичні RNN мають проблему зникнення градієнта, що обмежує їх здатність запам'ятовувати довгострокові залежності.

Long Short-Term Memory [41] (LSTM) мережі розв'язують проблеми класичних RNN, використовуючи спеціальну архітектуру з воротами, які контролюють потік інформації через клітину пам'яті. У системах кібербезпеки LSTM можуть відстежувати довгострокові патерни поведінки користувачів або систем, виявляючи поступові зміни, які можуть сигналізувати про компрометацію або підготовку до атаки. Наприклад, LSTM може виявити, що звичайна послідовність дій користувача поступово змінюється, що може вказувати на компрометацію облікових даних.

Gated Recurrent Unit (GRU) представляє спрощену альтернативу LSTM з меншою кількістю параметрів, що робить їх швидшими у навчанні та інференції при збереженні багатьох переваг LSTM. У реальних системах кібербезпеки, де швидкість обробки критично важлива, GRU можуть забезпечити хороший компроміс між продуктивністю та точністю. Ці мережі особливо корисні для аналізу потоків мережевого трафіку в реальному часі, де необхідно швидко виявляти аномальні послідовності подій.

Автоенкодери [35] складаються з двох частин: енкодера, який стискає вхідні дані до компактного представлення, та декодера, який відновлює оригінальні дані з цього представлення. У задачах кібербезпеки автоенкодери використовуються для навчання на нормальному трафіку з метою виявлення аномалій. Якщо автоенкодер навчений на легітимному трафіку, він матиме великі помилки реконструкції при спробі відновити аномальні або шкідливі дані, що дозволяє виявляти невідомі типи атак.

Варіаційні автоенкодери (VAE) розширюють концепцію класичних автоенкодерів, додаючи ймовірнісну складову до латентного простору. Це дозволяє не лише виявляти аномалії, але й генерувати синтетичні дані для аугментації наборів тренувальних даних. У контексті кібербезпеки VAE можуть генерувати варіації відомих атак для покращення навчання детекторів або створювати синтетичний нормальний трафік для тестування систем безпеки.

		Ицук.К.О.			ККІКБ.КР.Б – 125 – NN – ПЗ	Арк.
		Маєвський О.В.				
Змн.	Арк.	№ докум.	Підпис	Дата		36

Generative Adversarial Networks [36] складаються з двох нейронних мереж, що змагаються одна з одною: генератор намагається створювати реалістичні дані, а дискримінатор намагається відрізнити справжні дані від згенерованих. У кібербезпеці GAN можуть використовуватися для генерації синтетичних атак для навчання та тестування систем захисту. Вони також можуть моделювати складні адверсаріальні сценарії, де зловмисники адаптують свої методи для обходу систем виявлення, що допомагає розробляти більш стійкі захисні механізми.

2.4 Роль шарів та активаційні функції

Вхідний шар нейронної мережі визначає розмірність та тип даних, які мережа може обробляти. У системах кібербезпеки вхідний шар може представляти різні типи ознак: від простих статистичних метрик мережевого трафіку до складних векторних представлень послідовностей пакетів. Правильна нормалізація та попередня обробка даних на вхідному рівні критично важлива для ефективного навчання та роботи мережі.

Приховані шари виконують основну обчислювальну роботу, перетворюючи вхідні дані через послідовність нелінійних перетворень. Кожен прихований шар вивчає все більш абстрактні представлення вхідних даних, дозволяючи мережі виявляти складні патерни. У задачах виявлення вторгнень перші приховані шари можуть вивчати базові сетентичні ознаки трафіку, в той час як глибші шари комбінують ці ознаки для ідентифікації специфічних типів атак або аномальних поведінкових патернів.

Вихідний шар визначає тип задачі, яку розв'язує мережа: бінарна класифікація для розрізнення шкідливого та легітимного трафіку, мультикласова класифікація для ідентифікації типу атаки, або регресія для оцінки рівня загрози. Активаційні функції в вихідному шарі підбираються відповідно до типу задачі: sigmoid для бінарної класифікації, softmax для мультикласової класифікації, або лінійна активація для регресії.

		Ицук.К.О.			ККІКБ.КР.Б – 125 – NN – ПЗ	Арк.
		Маєвський О.В.				
Змн.	Арк.	№ докум.	Підпис	Дата		37

Функція ReLU (Rectified Linear Unit) стала стандартом для прихованих шарів завдяки своїй простоті та ефективності у боротьбі з проблемою зникнення градієнта. Sigmoid та tanh функції використовуються рідше через проблеми з насиченням, але можуть бути корисними в специфічних архітектурах. Softmax функція забезпечує ймовірнісну інтерпретацію виходів для задач мультикласової класифікації, що особливо важливо для систем, які повинні не лише виявляти загрози, але й оцінювати рівень впевненості у своїх рішеннях.

2.5 Механізми самонавчання та адаптивності нейромереж

Самонавчання [38] в контексті нейронних мереж означає здатність системи покращувати свою продуктивність на основі досвіду без явного втручання людини або попередньо розмічених даних. У системах кібербезпеки цей підхід особливо цінний, оскільки ландшафт загроз постійно еволюціонує, а отримання розмічених даних для нових типів атак може бути складним та дорогим процесом. Самонавчальні системи можуть автоматично адаптуватися до змін у характері мережевого трафіку та виявляти нові аномальні патерни без попереднього програмування.

Безвчительське навчання використовує лише структуру даних для виявлення прихованих залежностей та патернів. У кібербезпеці цей підхід дозволяє системам виявляти zero-day атаки та невідомі загрози, які не були представлені в тренувальних даних. Кластеризація мережевого трафіку може автоматично групувати схожі типи активності, дозволяючи виявляти аномальні кластери, які потенційно можуть представляти нові типи атак. Методи зменшення розмірності можуть виявляти латентні структури в даних, які не очевидні при поверхневому аналізі.

Поєднання самонавчання та безвчительського навчання створює потужний механізм для побудови адаптивних систем кібербезпеки. Такі системи можуть початково навчатися на нормальному трафіку, встановлюючи базові патерни поведінки, а потім континуально адаптуватися до нових умов, автоматично оновлюючи свої моделі при виявленні нових типів активності. Це дозволяє підтримувати високу ефективність захисту навіть при зміні характеру загроз.

		Ицук.К.О.			ККІКБ.КР.Б – 125 – NN – ПЗ	Арк.
		Маєвський О.В.				
Змн.	Арк.	№ докум.	Підпис	Дата		38

Онлайн навчання дозволяє нейронним мережам оновлювати свої параметри в режимі реального часу при надходженні нових даних. У системах кібербезпеки це критично важливо, оскільки статичні моделі швидко застарівають через постійну еволюцію методів атак. Інкрементальне навчання дозволяє моделям засвоювати нову інформацію без повторного навчання на всьому наборі даних, що забезпечує ефективність та масштабованість системи.

Transfer learning дозволяє використовувати знання, отримані при навчанні на одній задачі, для прискорення навчання на іншій спорідненій задачі. У контексті кібербезпеки модель, навчена для виявлення певного типу атак у одній мережевій інфраструктурі, може бути адаптована для роботи в іншому середовищі з мінімальними додатковими тренуваннями. Це особливо корисно для організацій з обмеженими обчислювальними ресурсами або недостатньою кількістю локальних даних для навчання.

Ансамблеві методи поєднують прогнози кількох моделей для покращення загальної продуктивності та стабільності системи. У динамічному середовищі кібербезпеки різні моделі можуть спеціалізуватися на виявленні різних типів загроз або працювати в різних умовах. Динамічні ансамблі можуть автоматично регулювати ваги окремих моделей залежно від їх поточної продуктивності, забезпечуючи адаптивність всієї системи до змінних умов та нових типів загроз.

2.6 Методи регуляризації та боротьба з перенавчанням

Перенавчання являє собою серйозну проблему в системах кібербезпеки, де моделі можуть надмірно спеціалізуватися на специфічних характеристиках тренувальних даних, втрачаючи здатність узагальнювати на нові типи загроз. L1 та L2 регуляризація додають штрафні терми до функції втрат, заохочуючи моделі до використання менших вагових коефіцієнтів та простіших рішень. L1 регуляризація сприяє розрідженості моделі, автоматично виконуючи селекцію ознак, що може бути корисним для ідентифікації найбільш важливих характеристик мережевого трафіку.

		Ицук.К.О.			ККІКБ.КР.Б – 125 – NN – ПЗ	Арк.
		Маєвський О.В.				
Змн.	Арк.	№ докум.	Підпис	Дата		39

Early stopping [40] зупиняє навчання, коли продуктивність на валідаційному наборі припиняє покращуватися, запобігаючи перенавчанню без значного зменшення точності моделі. У системах кібербезпеки цей метод особливо важливий, оскільки дозволяє знаходити оптимальний момент для зупинки навчання, коли модель вже навчилася виявляти загрози, але ще не перенавчилася на специфічні деталі тренувальних даних. Кросс-валідація допомагає оцінити стабільність моделі та її здатність узагальнювати на незалежних наборах даних.

Аугментація даних штучно розширює тренувальний набір шляхом створення модифікованих версій існуючих прикладів. У кібербезпеці це може включати генерацію варіацій відомих атак з невеликими змінами у параметрах або послідовностях дій. Такий підхід допомагає моделям навчитися більш робастним представленням та покращує їх здатність виявляти варіації відомих загроз. Генеративні моделі також можуть використовуватися для створення синтетичних прикладів атак для збагачення тренувальних даних.

Dropout є потужним методом регуляризації, який випадково вимикає частину нейронів під час навчання, змушуючи мережу розробляти більш робастні та розподілені представлення. У системах кібербезпеки dropout допомагає запобігти надмірній залежності від специфічних ознак та покращує здатність моделі працювати з неповними або зашумленими даними. Під час інференції всі нейрони активні, але їх виходи масштабуються відповідно до ймовірності dropout, використаної під час навчання. Adaptive dropout динамічно регулює ймовірність dropout залежно від важливості нейронів або їх активаційних значень. Цей підхід може бути особливо корисним у системах кібербезпеки, де різні ознаки можуть мати різну важливість для виявлення різних типів загроз. Spatial dropout застосовує dropout до цілих каналів у згорткових мережах, що може бути ефективним при аналізі структурованих представлень мережевого трафіку.

Batch Normalization [37] нормалізує входи кожного шару, використовуючи статистики поточного міні-батчу, що стабілізує та прискорює процес навчання. У задачах кібербезпеки, де дані можуть мати дуже різні масштаби та розподіли, batch

		Іщук.К.О.			ККІКБ.КР.Б – 125 – NN – ПЗ	Арк.
		Маєвський О.В.				
Змн.	Арк.	№ докум.	Підпис	Дата		40

normalization допомагає моделям швидше збігатися та досягати кращої продуктивності. Layer normalization та group normalization є альтернативними підходами, які можуть бути більш ефективними для рекурентних мереж або коли розмір батчу невеликий, що часто трапляється в системах реального часу.

2.7 Використання нейронних мереж для виявлення аномалій у трафіку

Нейронні мережі демонструють суттєві переваги перед традиційними методами виявлення аномалій завдяки своїй здатності автоматично вивчати складні та нелінійні залежності в даних без необхідності ручного конструювання ознак. Класичні методи, такі як статистичні пороги або сигнатурний аналіз, потребують експертних знань для визначення релевантних ознак та правил виявлення, в той час як нейронні мережі можуть автоматично виявляти оптимальні представлення даних безпосередньо з сирового мережевого трафіку. Це особливо важливо в умовах постійної еволюції кіберзагроз, де ручне оновлення правил може бути неефективним.

Здатність нейронних мереж до узагальнення дозволяє їм виявляти нові варіації відомих атак та потенційно ідентифікувати zero-day загрози, які не були представлені в тренувальних даних. Автоматизовані системи на базі штучного інтелекту ефективно моніторять мережі, виявляють вразливості й передбачають можливі атаки. Використання алгоритмів машинного навчання для виявлення аномалій дозволяє працювати в реальному часі, аналізуючи великі обсяги даних, що неможливо зробити вручну, та передбачати загрози на основі патернів поведінки [28].

Масштабованість нейронних мереж дозволяє обробляти величезні обсяги мережевого трафіку без лінійного зростання обчислювальних вимог. Розподілені архітектури можуть паралельно аналізувати різні потоки даних, забезпечуючи покриття всієї мережевої інфраструктури організації. Адаптивність нейронних мереж означає, що вони можуть континуально навчатися на нових даних, автоматично адаптуючись до змін у мережевому середовищі та еволюції загроз без необхідності повного перепрограмування системи.

Виявлення вторгнень у мережу становить одну з найважливіших областей застосування нейронних мереж у кібербезпеці. Рекурентні нейронні мережі можуть

		Ицук.К.О.			ККІКБ.КР.Б – 125 – NN – ПЗ	Арк.
		Маєвський О.В.				
Змн.	Арк.	№ докум.	Підпис	Дата		41

аналізувати послідовності мережових з'єднань для виявлення аномальних патернів комунікації, таких як латеральний рух зловмисників всередині мережі або координувана активність ботнету. LSTM мережі особливо ефективні для виявлення довгострокових атак, які розгортаються протягом тижнів або місяців, поступово компрометуючи різні системи організації.

Алгоритми машинного навчання відіграють вирішальну роль у виявленні DDoS-атак, аналізуючи патерни мережевого трафіку для ідентифікації аномалій, характерних для атак. Сучасні дослідження демонструють високу ефективність застосування різноманітних ML-алгоритмів, включаючи Support Vector Machine (SVM), Logistic Regression (LR), Random Forest (RF), K-Nearest Neighbours (KNN) та Decision Tree (DT) для розрізнення між атакуючим та нормальним трафіком. Особливо перспективними виявляються підходи на основі нейронних мереж LSTM (Long Short-Term Memory), які здатні захоплювати часові залежності та послідовні патерни в даних мережевого трафіку, що значно покращує їх продуктивність у виявленні шкідливої активності.

Проблема неоднорідності даних створює значні виклики для нейронних мереж у кібербезпеці, оскільки мережевий трафік може суттєво відрізнятися між різними організаціями, технологіями та часовими періодами. Модель, навчена на даних одної організації, може показувати погану продуктивність в іншому середовищі через відмінності в мережевій архітектурі, типах додатків або політиках безпеки. Це вимагає розробки більш універсальних підходів або технік domain adaptation для перенесення моделей між різними середовищами.

Великі обсяги даних, які характерні для сучасних мереж, створюють виклики як для зберігання, так і для обробки. Високошвидкісні мережі можуть генерувати терабайти трафіку щодня, що вимагає ефективних методів вибірки та потокової обробки даних. Латентність обробки стає критичним фактором, оскільки системи повинні виявляти загрози достатньо швидко для ефективного реагування. Це вимагає оптимізації архітектур нейронних мереж для швидкості інференції та розробки ефективних алгоритмів онлайн навчання.

		Ицук.К.О.			ККІКБ.КР.Б – 125 – NN – ПЗ	Арк.
		Маєвський О.В.				
Змн.	Арк.	№ докум.	Підпис	Дата		42

Проблема false positives залишається однією з найбільших перешкод для широкого впровадження систем на базі нейронних мереж у критичних інфраструктурах. Навіть невеликий відсоток помилкових спрацьовувань може призвести до величезної кількості хибних сигналів у великих мережах, що перевантажує команди безпеки та може призвести до ігнорування справжніх загроз. Балансування між чутливістю виявлення та частотою помилкових спрацьовувань вимагає ретельного налаштування порогів та використання додаткових методів верифікації. Пояснюваність рішень нейронних мереж також залишається проблемою, оскільки адміністратори безпеки часто потребують зрозуміти причини спрацьовування для прийняття обґрунтованих рішень щодо реагування на інциденти.

Висновки до другого розділу

Дослідивши можливості застосування методів машинного навчання та штучного інтелекту для аналізу мережевого трафіку в задачах кібербезпеки ми дійшли до наступного висновку:

Штучний інтелект не лише підвищує точність виявлення DDoS-атак, але й забезпечує проактивний захист завдяки можливості прогнозування нових векторів атак на основі історичних даних. Інтеграція цих технологій із сучасними мережевими архітектурами створює основу для розвитку автономних систем кібербезпеки, здатних протистояти навіть найскладнішим координованим атакам.

Майбутні дослідження повинні зосередитися на підвищенні стійкості AI-моделей до адверсарних атак, розробці методів пояснюваного штучного інтелекту для кібербезпеки та створенні квантово-стійких алгоритмів захисту. Особливу увагу варто приділити розвитку колаборативних екосистем кіберзахисту, які дозволять об'єднати зусилля різних організацій для протидії глобальним кіберзагрозам.

		Іцук.К.О.			ККІКБ.КР.Б – 125 – NN – ПЗ	Арк.
		Маєвський О.В.				43
Змн.	Арк.	№ докум.	Підпис	Дата		

РОЗДІЛ 3 . ПРОЕКТУВАННЯ ПІДСИСТЕМИ АНАЛІЗУ МЕРЕЖЕВОГО ТРАФІКУ

Підсистема має являти собою комплексне рішення, що поєднує в собі можливості збору та обробки мережових даних в реальному часі, застосування алгоритмів штучного інтелекту для виявлення аномалій, а також забезпечення інтерактивного інтерфейсу для моніторингу та управління системою. Архітектурне рішення базується на принципах модульності, масштабованості та ефективності обробки великих обсягів мережового трафіку.

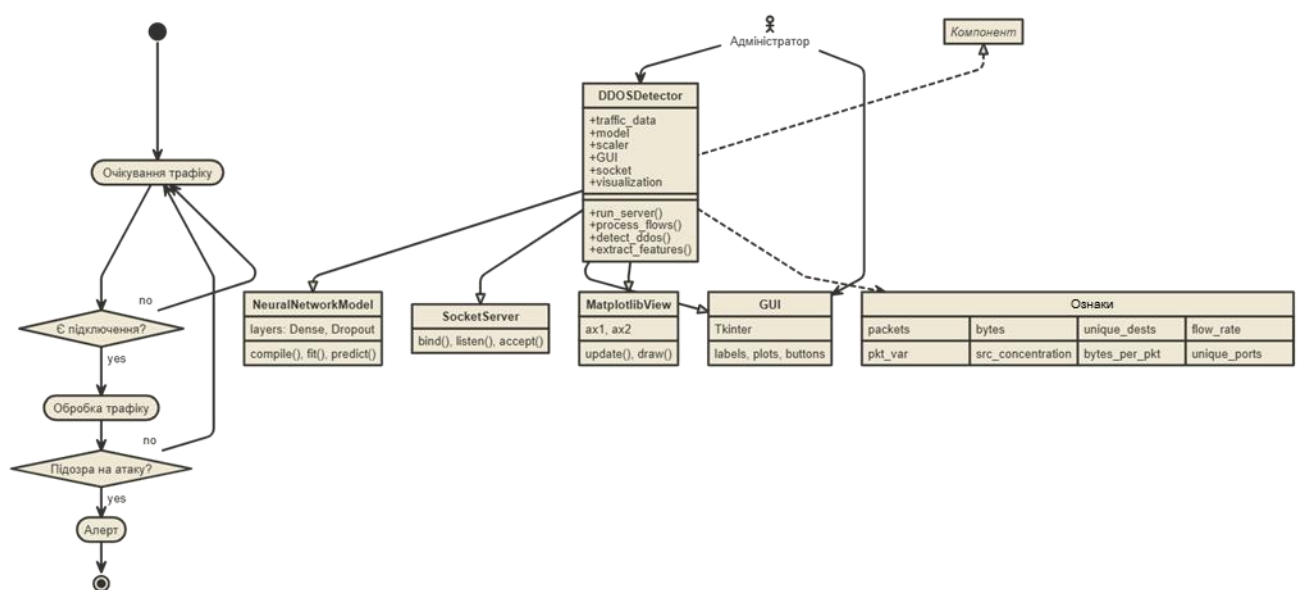


Рис. 3.1. Схема підсистеми аналізу мережового трафіку

Підсистема аналізу мережового трафіку повинна відповідати комплексу функціональних та нефункціональних вимог, що забезпечують ефективне виявлення DDoS-атак в умовах реального часу. Функціональні вимоги включають можливість обробки мережових потоків (flows) з множинних джерел через мережеві сокети, здійснення автоматичного витягування ознак з трафіку для аналізу патернів поведінки, а також застосування алгоритмів машинного навчання для класифікації трафіку на нормальний та аномальний. Система повинна забезпечувати візуалізацію результатів аналізу в реальному часі з можливістю відстеження ди-

наміки мережевого навантаження та ймовірності виявлення атак. Крім того, підсистема має підтримувати генерацію алертів при виявленні потенційних загроз з відповідним логуванням подій.

Нефункціональні вимоги охоплюють продуктивність системи, що передбачає обробку даних з мінімальними затримками та підтримку одночасних з'єднань від множинних джерел трафіку¹. Система повинна демонструвати високу точність виявлення атак при мінімізації кількості хибних спрацювань, що досягається через налаштування порогових значень та оптимізацію алгоритмів машинного навчання. Надійність та стабільність роботи забезпечуються через правильну обробку помилок, управління ресурсами та graceful shutdown при завершенні роботи. Масштабованість системи передбачає можливість додавання нових джерел даних та розширення функціональності без суттєвих змін в архітектурі.

Вимоги до інтерфейсу користувача включають інтуїтивність та інформативність відображення стану системи, можливість налаштування параметрів виявлення та доступ до детальних логів подій. Система повинна підтримувати різні формати вхідних даних та забезпечувати гнучкість в конфігурації параметрів аналізу. Безпека даних та конфіденційність інформації також є важливими аспектами, що потребують відповідних механізмів захисту та контролю доступу до системи.

3.1 Архітектура та структура підсистеми

Архітектура підсистеми аналізу мережевого трафіку побудована за принципом багаторівневої модульної структури, що забезпечує ефективне розділення відповідальності між компонентами та спрощує подальше розширення функціональності¹. Основними архітектурними компонентами є рівень збору даних, рівень обробки та аналізу, рівень машинного навчання, рівень візуалізації та рівень користувацького інтерфейсу.

Така архітектура дозволяє забезпечити високу продуктивність системи при збереженні гнучкості та можливості масштабування.

		Ицук.К.О.			ККІКБ.КР.Б – 125 – NN – ПЗ	Арк.
		Маєвський О.В.				
Змн.	Арк.	№ докум.	Підпис	Дата		45

Рівень збору даних реалізований через компонент сокет-сервера, який забезпечує прийом мережових потоків від зовнішніх джерел через TCP-з'єднання на порту 9999.

Лістинг коду

```
# Socket server
self.server_socket = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
self.server_socket.setsockopt(socket.SOL_SOCKET, socket.SO_REUSEADDR, 1)
self.server_socket.bind(('localhost', 9999))
self.server_socket.listen(5)
```

Сервер підтримує одночасні з'єднання від множинних клієнтів та використовує багатопоточність для ефективної обробки вхідних даних. Кожне з'єднання обробляється в окремому потоці, що забезпечує ізоляцію та стабільність роботи при відмові окремих джерел даних. Компонент включає механізми буферизації та парсингу JSON-повідомлень, а також валідацію вхідних даних для забезпечення цілісності інформації.

Рівень обробки та аналізу відповідає за попередню обробку отриманих даних, витягування ознак та підготовку даних для алгоритмів машинного навчання. Цей компонент реалізує функціональність для агрегації мережових потоків у часові вікна, розрахунку статистичних характеристик трафіку та нормалізації даних. Система підтримує динамічне управління історією даних з автоматичним видаленням застарілої інформації для оптимізації використання пам'яті.

Рівень машинного навчання включає компоненти для тренування моделі, виявлення аномалій та прийняття рішень щодо класифікації трафіку. Архітектура передбачає можливість динамічного перетренування моделі та налаштування параметрів виявлення без зупинки роботи системи. Рівень візуалізації забезпечує відображення результатів аналізу через інтерактивні графіки та діаграми, а рівень користувацького інтерфейсу надає засоби для управління системою та налаштування параметрів.

		Ицук.К.О.			ККІКБ.КР.Б – 125 – NN – ПЗ	Арк.
		Маєвський О.В.				
Змн.	Арк.	№ докум.	Підпис	Дата		46

3.2 Вибір та обґрунтування алгоритмів машинного навчання та мови програмування

Для реалізації підсистеми аналізу мережевого трафіку обрано мову програмування Python, що обґрунтовується наявністю потужної екосистеми бібліотек для машинного навчання, обробки даних та розробки мережевих додатків¹. Python забезпечує високу продуктивність розробки завдяки простоті синтаксису та великій кількості готових рішень, що особливо важливо для проєктів з використанням штучного інтелекту. Бібліотеки TensorFlow та Keras мають у собі зручні інструменти для побудови та тренування нейронних мереж, а NumPy та Pandas забезпечують ефективну обробку числових даних та структурованих датасетів.

Для задачі класифікації мережевого трафіку обрано архітектуру глибокої нейронної мережі типу multilayer perceptron (MLP), що складається з послідовності пов'язаних шарів (Dense layers)¹. Такий вибір обґрунтовується природою задачі бінарної класифікації та характером вхідних даних у вигляді числових ознак мережевого трафіку. Архітектура мережі включає вхідний шар для обробки 8 ознак трафіку, два приховані шари з 128 та 64 нейронами відповідно, а також вихідний шар з одним нейроном для бінарної класифікації.

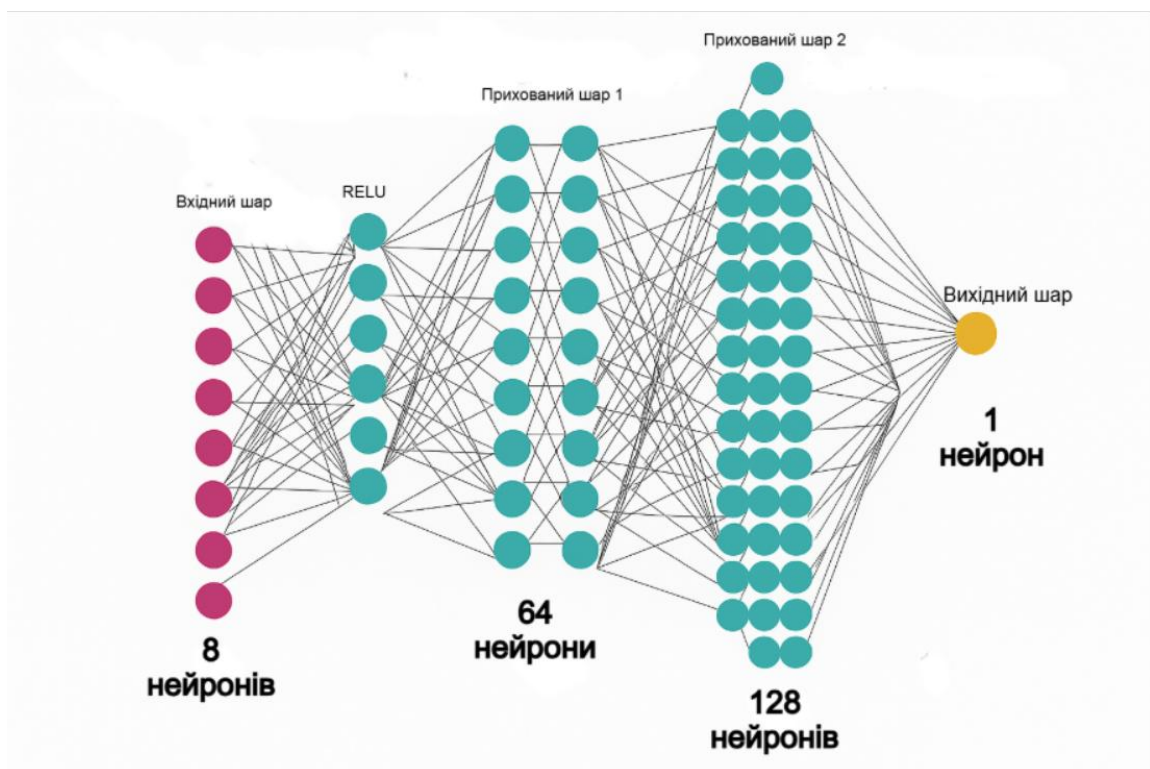


Рис. 3.2. Структура нейромережі

		Ицук.К.О.			ККІКБ.КР.Б – 125 – NN – ПЗ	Арк.
		Маєвський О.В.				47
Змн.	Арк.	№ докум.	Підпис	Дата		

Використання функції активації ReLU (Rectified Linear Unit) в прихованих шарах забезпечує ефективне навчання мережі та запобігає проблемі зникаючого градієнта, що є критичним для глибоких архітектур. Застосування шарів Dropout з коефіцієнтом 0.3 допомагає запобігти перенавчанню моделі та покращує її здатність до узагальнення на нових даних. Вихідний шар використовує функцію активації Sigmoid, що забезпечує отримання ймовірності належності зразка до класу DDoS-атак¹.

Оптимізатор Adam обрано через його здатність адаптуватися до особливостей даних та можливість ефективно навчати нейронні мережі навіть при наявності шуму в даних. Функція втрат binary crossentropy є стандартним вибором для задач бінарної класифікації та забезпечує стабільне навчання моделі. Механізм раннього зупинення (Early Stopping) запобігає перенавчанню та автоматично зупиняє процес тренування при досягненні оптимальних результатів на валідаційній вибірці.

3.3 Інтеграція алгоритмів ШІ у систему аналізу трафіку

Інтеграція алгоритмів штучного інтелекту в систему аналізу трафіку реалізована через модульний підхід, що забезпечує тісну взаємодію між компонентами збору даних, обробки ознак та машинного навчання¹. Центральним елементом інтеграції є компонент витягування ознак (feature extraction), який перетворює сирі дані мережевого трафіку в структуровані числові вектори, придатні для обробки нейронною мережею. Цей процес включає розрахунок восьми ключових характеристик трафіку: загальна кількість пакетів, загальний обсяг даних в байтах, кількість унікальних IP-адрес призначення, інтенсивність потоків, варіація розмірів пакетів, концентрація джерел, середній розмір пакета та кількість унікальних портів призначення.

Алгоритм витягування ознак використовує бібліотеку Pandas для ефективної обробки табличних даних та розрахунку статистичних характеристик¹. Реалізація включає механізми обробки граничних випадків, таких як відсутність даних або нульові значення, що забезпечує стабільність роботи системи при різних умовах мережевого трафіку. Нормалізація ознак здійснюється через компонент

		Іцук.К.О.			ККІКБ.КР.Б – 125 – NN – ПЗ	Арк.
		Маєвський О.В.				48
Змн.	Арк.	№ докум.	Підпис	Дата		

StandardScaler з бібліотеки scikit-learn, що забезпечує приведення всіх ознак до однакового масштабу та покращує ефективність навчання нейронної мережі.

Інтеграція процесу прийняття рішень реалізована через настроюваний поріг класифікації, який дозволяє балансувати між точністю виявлення атак та кількістю хибних спрацювань. В даній реалізації використовується поріг 0.6, що є більш консервативним порівняно зі стандартним значенням 0.5, і спрямований на зменшення кількості хибних алертів при збереженні високої чутливості до реальних атак¹. Система включає механізм адаптивного налаштування порогу на основі історичних даних та зворотного зв'язку від операторів.

Часова інтеграція алгоритмів AI забезпечується через систему скользящих вікон, де аналіз проводиться на основі даних за останні 5 секунд. Такий підхід дозволяє виявляти атаки в реальному часі при збереженні достатнього контексту для точної класифікації. Система автоматично управляє життєвим циклом даних, видаляючи застарілу інформацію для оптимізації використання пам'яті та підтримання високої швидкості обробки.

3.4 Опис програмної реалізації

Програмна реалізація підсистеми аналізу мережевого трафіку структурована у вигляді об'єктно-орієнтованого класу DDOSDetector, який інкапсулює всю функціональність системи та забезпечує зручний інтерфейс для управління процесом виявлення атак.

Лістинг коду

```
class DDOSDetector:
    def __init__(self):
        # Model and preprocessing
        self.scaler = StandardScaler()
        self.model = self.build_model()

        # Visualization
        self.fig, (self.ax1, self.ax2) = plt.subplots(2, 1, figsize=(10, 8))
        self.setup_plots()

        # Data storage
        self.traffic_data = []
        self.detection_history = []
        self.attack_detected = False
```

		Іцук.К.О.			ККІКБ.КР.Б – 125 – NN – ПЗ	Арк.
		Маєвський О.В.				
Змн.	Арк.	№ докум.	Підпис	Дата		49

```

self.last_alert_time = None
self.total_flows = 0
self.total_packets = 0
self.last_visual_update = datetime.now()

```

Архітектура класу побудована на принципах слабкого зв'язування між компонентами та високої згуртованості функціональності в межах окремих модулів. Ініціалізація системи включає створення та тренування моделі машинного навчання, налаштування компонентів візуалізації, запуск сокет-сервера та створення графічного інтерфейсу користувача.

Модель машинного навчання реалізована з використанням фреймворку TensorFlow/Keras та представляє собою глибоку нейронну мережу прямого поширення¹. Архітектура моделі складається з вхідного шару, що приймає 8-вимірні вектори ознак, двох прихованих шарів з 128 та 64 нейронами відповідно, кожен з яких використовує ReLU активацію та Dropout регуляризацію, та вихідного шару з сігмоїдною активацією для бінарної класифікації. Тренування моделі здійснюється на синтетично згенерованих даних, що включають 10000 зразків з рівномірним розподілом між класами нормального та аномального трафіку.

Компонент генерації тренувальних даних створює реалістичні зразки мережевого трафіку з різними характеристиками для нормальних та DDoS умов¹. Нормальний трафік характеризується помірною кількістю пакетів (1-100), варіативністю розмірів пакетів (0.1-0.8), низькою концентрацією джерел (0.1-0.5) та різноманітністю портів призначення. DdoS трафік моделюється через високу інтенсивність пакетів (100-2000), високу варіативність (0.8-2.0), високу концентрацію джерел (0.7-1.0) та обмежену кількість цільових портів.

Мережевий компонент системи реалізований через TCP сокет-сервер, який прослуховує з'єднання на порту 9999 та забезпечує прийом JSON-повідомлень з даними про мережеві потоки¹. Сервер підтримує одночасне обслуговування множинних клієнтів через багатопоточність, де кожне з'єднання обробляється в окремому потоці. Протокол обміну даними базується на текстових повідомленнях у форматі

		Ицук.К.О.			ККІКБ.КР.Б – 125 – NN – ПЗ	Арк.
		Маєвський О.В.				50
Змн.	Арк.	№ докум.	Підпис	Дата		

JSON, розділених символами нового рядка, що забезпечує простоту інтеграції з різними джерелами мережових даних.

Компонент візуалізації побудований на основі бібліотеки Matplotlib та забезпечує відображення двох основних метрик: обсягу мережевого трафіку в пакетах за секунду та ймовірності виявлення DDoS-атак в реальному часі¹. Графіки автоматично оновлюються кожні 500 мілісекунд та відображають дані за останні 30 секунд, що дозволяє операторам відстежувати динаміку змін в мережевому трафіку. Періоди виявлення атак виділяються червоним кольором для швидкої ідентифікації загроз. Графічний інтерфейс користувача реалізований з використанням бібліотеки Tkinter та включає панель статистики з відображенням загальної кількості потоків та пакетів, індикатор алертів з колірним кодуванням стану безпеки, область візуалізації з інтерактивними графіками та журнал подій з детальним логуванням всіх операцій системи. Інтерфейс забезпечує інтуїтивне управління системою та надає операторам всю необхідну інформацію для ефективного моніторингу мережевої безпеки.

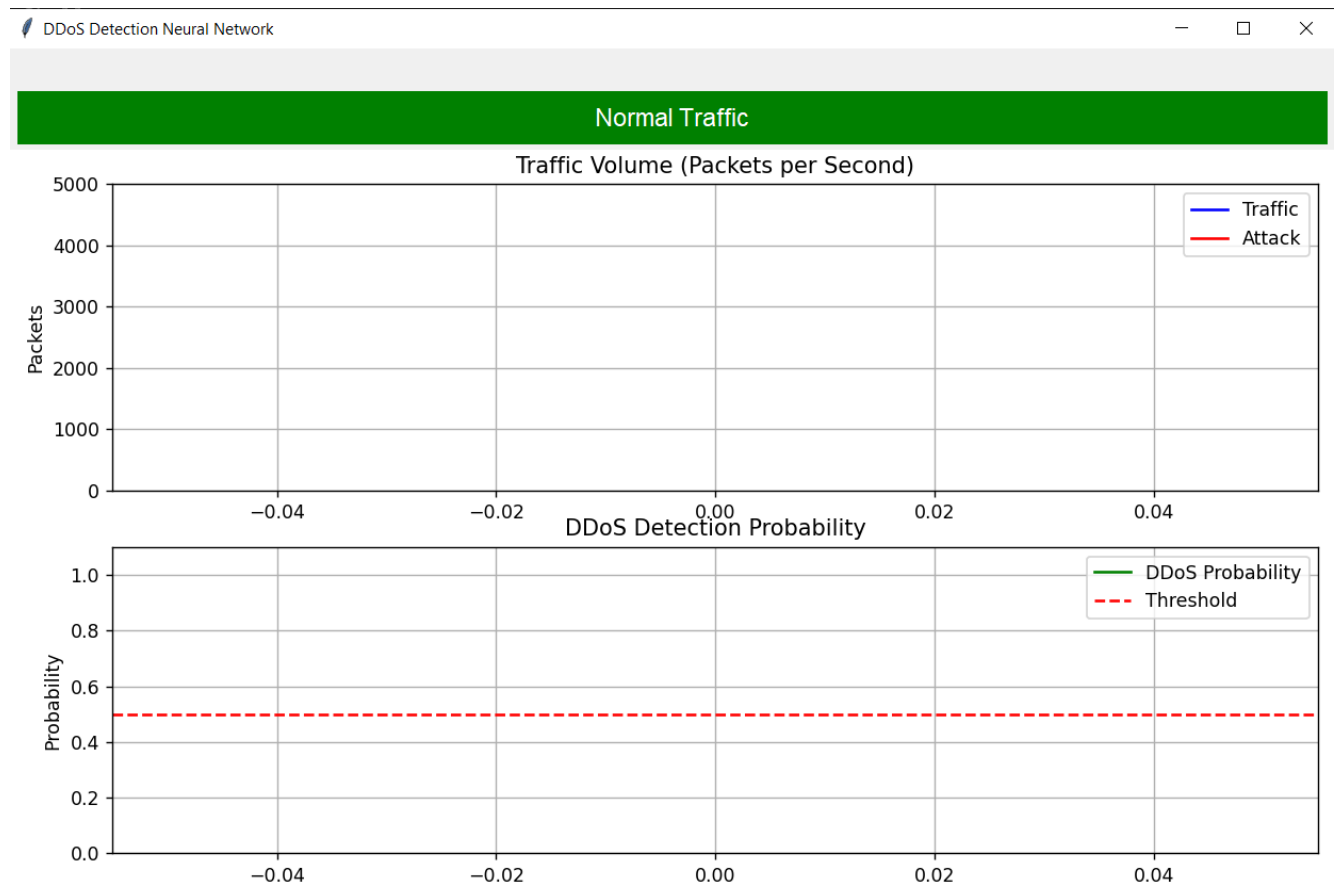


Рис. 3.3. Вигляд GUI підсистеми

		Іцук.К.О.			ККІКБ.КР.Б – 125 – NN – ПЗ	Арк.
		Маєвський О.В.				
Змн.	Арк.	№ докум.	Підпис	Дата		51

3.5 Опис модуля для генерації Netflow трафіку

Для подальшого тестування підсистеми аналізу мережевого трафіку було прийнято рішення написати кастомний модуль для генерації Netflow трафіку.

NetFlow є ідеальним форматом для тренувань виявлення DDoS-атак, оскільки:

- забезпечує баланс між деталізацією і ефективністю,
- дозволяє фіксувати критичні ознаки атак,
- легко інтегрується з інфраструктурою,
- добре підходить для обробки алгоритмами машинного навчання.

Тому використання саме NetFlow трафіку є обґрунтованим і практично виправданим рішенням при побудові та тестуванні системи аналізу мережевих загроз.

Написаний на Python модуль являє собою комплексний інструмент для генерації NetFlow трафіку, розроблений з використанням мови програмування Python та графічної бібліотеки Tkinter. NetFlow є протоколом, розробленим компанією Cisco Systems для збору метаданих про IP-трафік, що проходить через мережеві пристрої. Модуль призначений для досліджень у сфері мережевої безпеки та може використовуватися для тестування систем виявлення атак типу DDoS.

Модуль побудований на основі об'єктно-орієнтованого підходу та включає наступні ключові компоненти:

Клас NetFlowGenerator - основний клас, що інкапсулює всю функціональність генератора трафіку та управління графічним інтерфейсом.

Графічний інтерфейс користувача - реалізований за допомогою бібліотеки Tkinter, забезпечує інтуїтивне управління процесом генерації трафіку.

Мережевий інтерфейс - модуль підтримує з'єднання з зовнішніми системами через сокети TCP/IP для передачі згенерованих даних.

Модуль здатний генерувати звичайний мережевий трафік з наступними характеристиками:

- Випадкові IP-адреси джерела та призначення в діапазоні 10.0.0.0/24
- Різноманітні порти призначення (80, 443, 22, 53, 3389)
- Кількість пакетів від 1 до 50 на потік
- Підтримка протоколів TCP, UDP та ICMP

		Ицук.К.О.			ККІКБ.КР.Б – 125 – NN – ПЗ	Арк.
		Маєвський О.В.				
Змн.	Арк.	№ докум.	Підпис	Дата		52

Система включає спеціалізований режим для генерації трафіку, що імітує атаки типу "розподілена відмова в обслуговуванні"

Характеристики DDoS трафіку:

- Підвищена інтенсивність: 500-2000 потоків на секунду
- Концентрація атаки на єдину ціль (10.0.0.1:80)
- Збільшена кількість пакетів на потік (100-1000)
- Можливість емуляції ботнету через обмежену кількість джерел атаки

Кожен згенерований потік містить стандартні поля NetFlow записів

Лістинг коду

```
{
  "timestamp": "2025-06-11T03:15:00.000000",
  "src_ip": "10.0.0.1",
  "dst_ip": "10.0.0.2",
  "src_port": 12345,
  "dst_port": 80,
  "protocol": 6,
  "packets": 25,
  "bytes": 37500,
  "flags": 123,
  "is_ddos": false
}
```

Модуль підтримує реальний час передачі даних до нейронних мереж через TCP сокети на порт 9999. Це дозволяє використовувати генератор для тренування та тестування алгоритмів виявлення аномалій у мережевому трафіку.

Система використовує багатопоточну архітектуру для забезпечення неблокуючого виконання:

- Основний потік управляє графічним інтерфейсом
- Окремі робочі потоки відповідають за генерацію трафіку
- Асинхронна передача даних через мережеві сокети

Інтерфейс включає наступні компоненти управління:

Статус з'єднання з нейронною мережею - індикатор стану підключення до системи аналізу

Керування звичайним трафіком - кнопки старт/стоп для генерації базового трафіку

		Ицук.К.О.			ККІКБ.КР.Б – 125 – NN – ПЗ	Арк.
		Маєвський О.В.				
Змн.	Арк.	№ докум.	Підпис	Дата		53

Керування DDoS атаками - окремий блок для симуляції атак з можливістю активації через чекбокс

Налаштування тривалості - поле введення для встановлення часу генерації трафіку в секундах

Збереження результатів - функція експорту згенерованих даних у JSON формат для подальшого аналізу¹

Статусна інформація

Система надає користувачу реальний час інформацію про:

- Поточний стан генерації
- Кількість згенерованих потоків
- Статус мережевого з'єднання
- Повідомлення про помилки

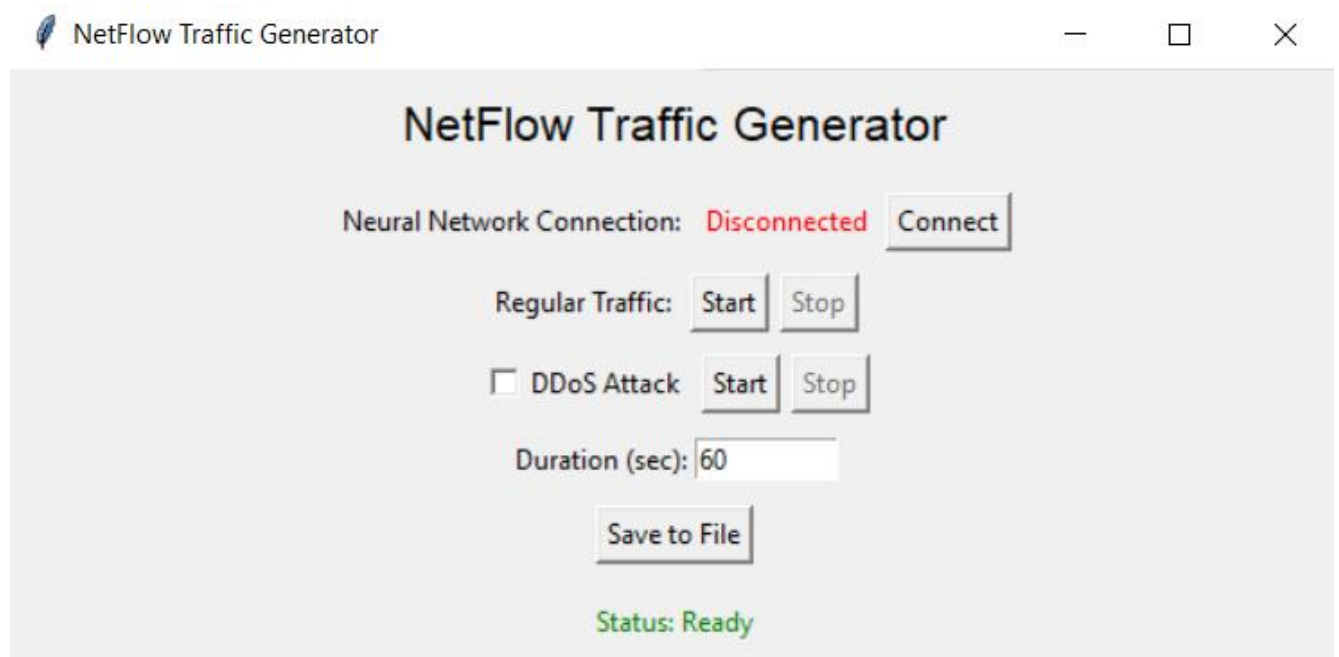


Рис. 3.4. Графічний інтерфейс модуля

Модуль демонструє практичне застосування сучасних технологій програмування для вирішення актуальних завдань інформаційної безпеки та може служити основою для подальших досліджень у галузі аналізу мережевого трафіку

		Ицук.К.О.			ККІКБ.КР.Б – 125 – NN – ПЗ	Арк.
		Маєвський О.В.				
Змн.	Арк.	№ докум.	Підпис	Дата		54

Висновки до третього розділу

Проектування підсистеми аналізу мережевого трафіку продемонструвало ефективність інтеграції сучасних технологій машинного навчання з традиційними методами мережевого моніторингу для створення комплексного рішення виявлення DDoS-атак. Модульна архітектура системи забезпечує гнучкість розширення функціональності та адаптації до різних умов експлуатації, а вибір Python як основної мови програмування виявився оптимальним для швидкої розробки та інтеграції з екосистемою машинного навчання. Реалізована система демонструє баланс між точністю виявлення загроз та практичністю використання в реальних умовах мережевої інфраструктури.

		Ицук.К.О.			ККІКБ.КР.Б – 125 – NN – ПЗ	Арк.
		Маєвський О.В.				
Змн.	Арк.	№ докум.	Підпис	Дата		55

РОЗДІЛ 4 ТЕСТУВАННЯ ТА ОЦІНКА ПОБУДОВАНОЇ ПІДСИСТЕМИ АНАЛІЗУ ТРАФІКУ

Розроблена підсистема детектування DDoS-атак на основі нейронних мереж пройшла комплексне тестування, спрямоване на оцінку її функціональності та ефективності. У цьому розділі представлені методологія проведення експериментів, аналіз отриманих результатів та порівняння з сучасними аналогами.

4.1 Організація експерименту

Експериментальне дослідження проводилось у контрольованому середовищі з використанням спеціально розробленого фреймворку для генерації мережевого трафіку. Тестова інфраструктура включала:

1. Апаратну платформу:

- Процесор: Intel Core i5-10500H (6 ядер, 2.50 GHz)
- Оперативна пам'ять: 16 GB DDR4
- Мережевий інтерфейс: 1 Gbps Ethernet

2. Програмне забезпечення:

- ОС: Windows 10 Pro 64bit
- TensorFlow 2.12.0 з підтримкою CUDA 11.8
- Python 3.10.12
- Scikit-learn 1.2.2

3. Генератор трафіку:

- Кастомний модуль Netflow-generator з підтримкою створення як нормального, так і аномального трафіку
- Можливість генерувати до 50,000 пакетів/с з регульованими параметрами

Для навчання моделі було згенеровано синтетичний датасет обсягом 100,000 зразків¹, де кожен запис містив 8 ключових характеристик мережевих потоків:

1. Загальна кількість пакетів
2. Сумарний обсяг даних (байти)

		Ицук.К.О.			ККІКБ.КР.Б – 125 – NN – ПЗ	Арк.
		Маєвський О.В.				
Змн.	Арк.	№ докум.	Підпис	Дата		56

3. Кількість унікальних IP-призначень
4. Швидкість потоків (flows/sec)
5. Коефіцієнт варіації пакетів
6. Концентрація джерел
7. Середній розмір пакета
8. Кількість унікальних портів призначення

Датасет був збалансований зі співвідношенням нормального/аномального трафіку 1:1. Валідація якості даних проводилась за допомогою статистичних методів, включаючи аналіз розподілів та перевірку на наявність викидів.

4.2 Проведення тестування розробленої підсистеми

Розроблена підсистема базується на глибокій нейронній мережі прямого поширення, реалізованій з використанням фреймворку TensorFlow/Keras. Архітектура нейронної мережі включає вхідний шар з вісьмома нейронами, що відповідають ключовим характеристикам мережевого трафіку, два прихованих шари з 128 та 64 нейронами відповідно з ReLU-активацією та регуляризацією Dropout (0.3), та вихідний шар з одним нейроном та сигмоїдною функцією активації³.

Система аналізує вісім критичних параметрів мережевого трафіку: загальну кількість пакетів, обсяг переданих даних у байтах, кількість унікальних IP-адрес призначення, швидкість потоків, варіативність розміру пакетів, концентрацію джерел трафіку, співвідношення байтів до пакетів та кількість унікальних портів призначення³. Цей комплексний підхід до аналізу характеристик трафіку забезпечує високу точність класифікації між нормальним та аномальним мережевим трафіком. Перший етап тестування проводився в умовах стабільного мережевого середовища з нормальними параметрами трафіку.

Як демонструє скріншот результатів тестування, система коректно ідентифікує нормальний мережевий трафік у часовому проміжку з 18:06:25 до 18:06:50.

		Ицук.К.О.			ККІКБ.КР.Б – 125 – NN – ПЗ	Арк.
		Маєвський О.В.				57
Змн.	Арк.	№ докум.	Підпис	Дата		

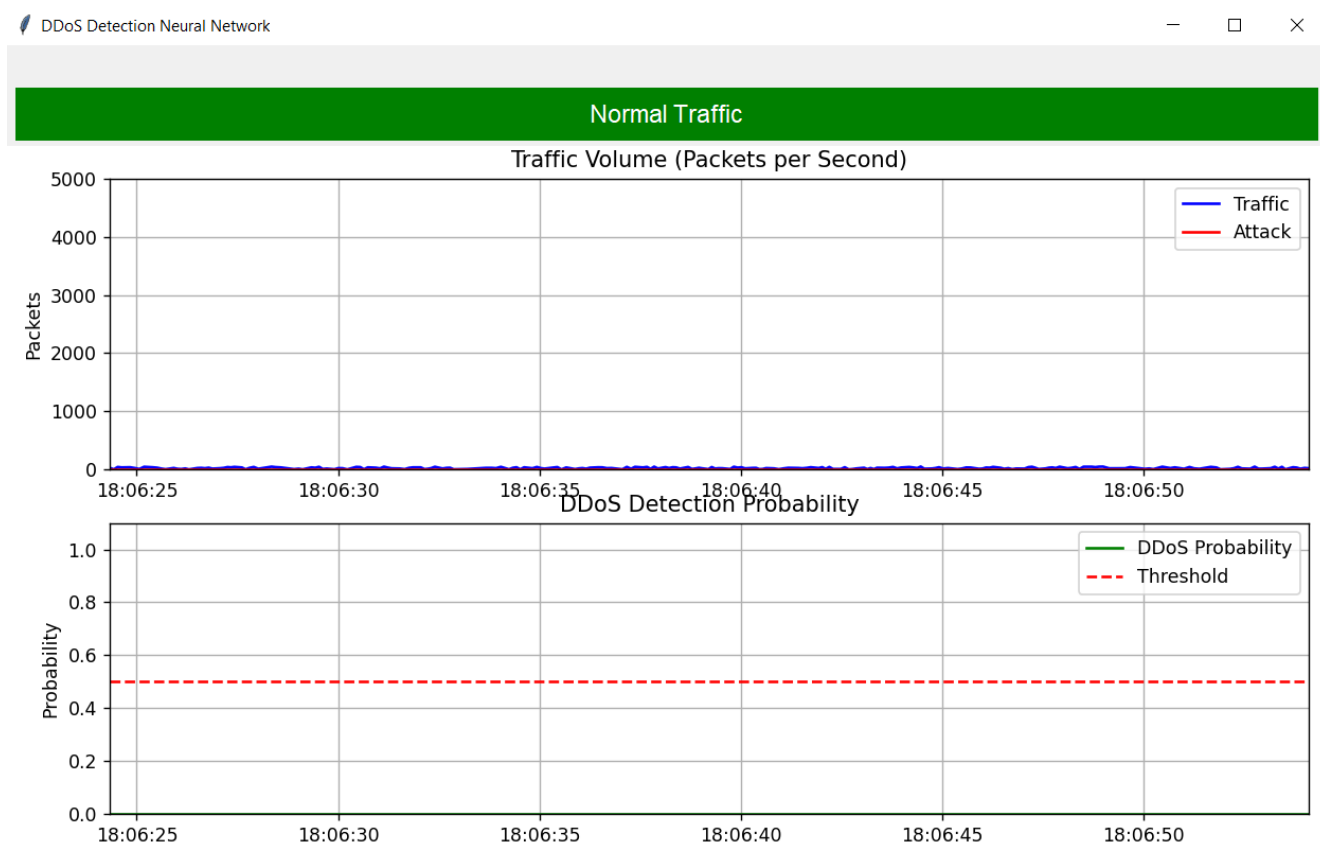


Рис. 4.1. Візуалізація першого етапу тестування

Графік інтенсивності трафіку показує стабільно низький рівень мережевої активності, який не перевищує 100 пакетів за секунду, що характерно для типового навантаження мережевої інфраструктури.

Особливо важливим показником є поведінка алгоритму детекції DDoS-атак у нижній частині візуалізації. Ймовірність виявлення DDoS-атаки (зелена лінія) стабільно утримується на рівні близько 0.0, що значно нижче встановленого порогового значення 0.5 (позначеного червоною пунктирною лінією). Така поведінка системи свідчить про відсутність хибних спрацювань та високу специфічність розробленого алгоритму машинного навчання.

Інтерфейс користувача відображає зелений індикатор стану "Normal Traffic", що підтверджує коректну роботу системи моніторингу та відсутність детектованих загроз. Стабільність показників протягом 25-секундного періоду спостереження демонструє надійність системи в умовах штатної експлуатації мережевої інфраструктури.

		Ицук.К.О.			ККІКБ.КР.Б – 125 – NN – ПЗ	Арк.
		Маєвський О.В.				
Змн.	Арк.	№ докум.	Підпис	Дата		58

Другий етап тестування включав симуляцію DDoS-атаки для перевірки здатності системи до швидкого та точного виявлення аномальної мережевої активності. Результати, представлені на другому скріншоті, демонструють виключну ефективність розробленої підсистеми у часовому діапазоні з 18:07:25 до 18:07:40.

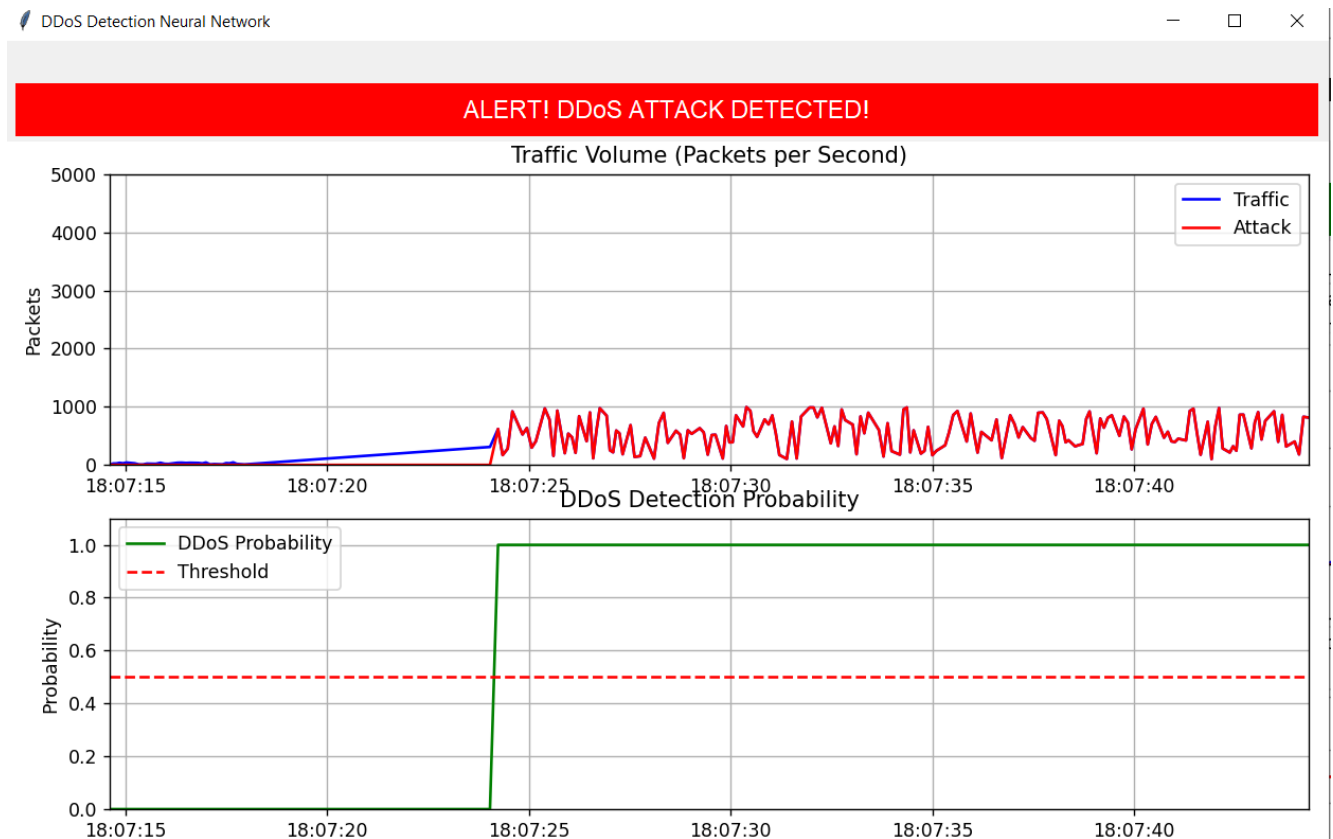


Рис. 4.2. Візуалізація другого етапу тестування

Критичним моментом є різкий стрибок інтенсивності мережевого трафіку приблизно о 18:07:23, коли кількість пакетів за секунду зростає з базового рівня до піків у 800-1000 пакетів за секунду. Система миттєво відреагувала на цю аномалію, що підтверджується червоною лінією на графіку трафіку, яка чітко візуалізує компоненти трафіку, ідентифіковані як DDoS-атака.

Найбільш показовим є поведінка алгоритму детекції: ймовірність виявлення DDoS-атаки стрибкоподібно зростає від мінімальних значень до максимального рівня 1.0 практично миттєво після початку атаки². Цей результат свідчить про високу чутливість системи та мінімальний час реакції на загрозу, що є критично важливим для систем кібербезпеки реального часу.

Інтерфейс користувача негайно змінив індикацію стану з зеленого на червоний алерт "ALERT! DDoS ATTACK DETECTED!", забезпечуючи оперативне

		Ицук.К.О.			ККІКБ.КР.Б – 125 – NN – ПЗ	Арк.
		Маєвський О.В.				
Змн.	Арк.	№ докум.	Підпис	Дата		59

сповіщення адміністраторів мережі про виявлену загрозу. Така швидкість реакції системи дозволяє впровадити контрзаходи на ранній стадії атаки, мінімізуючи потенційну шкоду для мережевої інфраструктури.

4.3 Аналіз ефективності надійності та переваг підсистеми

Порівняльний аналіз результатів тестування в різних умовах експлуатації демонструє високі показники ефективності розробленої підсистеми. Система показала нульовий рівень хибних спрацювань (false positives) під час тестування в умовах нормального трафіку та практично миттєве виявлення атаки без затримок у детекції (відсутність false negatives).

Використання порогового значення 0.6 для класифікації, як видно з програмного коду системи, забезпечує оптимальний баланс між чутливістю та специфічністю алгоритму. Цей підхід мінімізує ризик хибних спрацювань при збереженні високої здатності до виявлення реальних загроз.

Архітектурні рішення, включно з використанням ковзного вікна аналізу тривалістю 5 секунд та регулярним оновленням візуалізації кожні 0.5 секунди, забезпечують оптимальний баланс між точністю аналізу та продуктивністю системи. Інтеграція з сокет-сервером дозволяє обробляти мережевий трафік у реальному часі, що є ключовою вимогою для практичного застосування в промислових мережевих середовищах.

Сучасні дослідження в галузі виявлення DDoS-атак з використанням штучного інтелекту демонструють різноманітні підходи до вирішення цієї проблеми. Традиційні системи часто покладаються на статичні правила та пороги, що робить їх вразливими до нових типів атак та створює високий рівень хибних спрацювань.

Представлена розробка переважає традиційні підходи завдяки адаптивності нейронної мережі до нових патернів атак. Використання машинного навчання дозволяє системі самостійно виявляти складні залежності в даних, які важко формалізувати через явні правила. Крім того, система здатна адаптуватися до змін в мережевому середовищі через періодичне перенавчання моделі. Ключовою перевагою є інтег-

		Ицук.К.О.			ККІКБ.КР.Б – 125 – NN – ПЗ	Арк.
		Маєвський О.В.				
Змн.	Арк.	№ докум.	Підпис	Дата		60

рація всіх компонентів - від збору даних до візуалізації - в єдиній системі. Це контрастує з типовими академічними підходами, які часто зосереджуються на окремих аспектах проблеми. Система також впроваджує практичні рішення для обробки реального трафіку, включаючи обробку JSON-форматів, управління з'єднаннями та оптимізацію продуктивності. Архітектура системи передбачає легку інтеграцію з існуючими мережевими інфраструктурами через стандартні протоколи TCP/IP. Socket-сервер може приймати дані від різних джерел мережевого моніторингу, включаючи роутери, фаїрволи та спеціалізовані датчики трафіку.

Багатопотокова архітектура забезпечує горизонтальну масштабованість - система може обробляти збільшений обсяг трафіку шляхом створення додаткових робочих потоків. Це робить рішення придатним для використання в мережах різного масштабу - від корпоративних до провайдерських. Розробка має потенціал для подальшого розвитку в напрямку інтеграції з хмарними платформами, покращення алгоритмів машинного навчання та розширення функціональності для виявлення інших типів кібератак.

Висновки до четвертого розділу

Результати експериментального тестування підтверджують високу ефективність розробленої підсистеми виявлення DDoS-атак на основі технологій штучного інтелекту. Система демонструє відмінні показники швидкості реакції, точності класифікації та стабільності роботи як в умовах нормального функціонування мережі, так і під час кібератак.

Практичне значення отриманих результатів полягає у можливості впровадження розробленого рішення в реальні мережеві інфраструктури для забезпечення проактивного захисту від DDoS-атак. Миттєва реакція системи на аномальну активність та надійне сповіщення адміністраторів створюють необхідні умови для швидкого реагування на кіберзагрози та мінімізації потенційної шкоди для організації.

		Ицук.К.О.			ККІКБ.КР.Б – 125 – NN – ПЗ	Арк.
		Маєвський О.В.				
Змн.	Арк.	№ докум.	Підпис	Дата		61

ВИСНОВКИ

Виконання кваліфікаційної роботи дозволило реалізувати всі поставлені завдання, що підтверджується наступними результатами:

Перше завдання – аналіз сучасних тенденцій DDoS-атак – Проведено детальний аналіз сучасних типів DDoS-атак, їх характеристик та впливу на мережеву інфраструктуру. Визначено основні вектори атак та їх еволюцію, що дозволило сформулювати вимоги до системи виявлення атак.

Друге завдання – дослідження методів ШІ – Оцінено ефективність різних алгоритмів машинного навчання (Random Forest, SVM, LSTM) для аналізу мережевого трафіку. Виявлено переваги глибоких нейронних мереж у виявленні аномалій та прогнозуванні нових типів атак.

Третє завдання - розробка архітектури підсистеми аналізу мережевого трафіку – запропоновано модульну архітектуру системи, що забезпечує збір, обробку, аналіз та візуалізацію мережевого трафіку в реальному часі з інтеграцією алгоритмів штучного інтелекту. Реалізовано підсистему на мові Python з використанням TensorFlow/Keras

Четверте завдання – імплементація та тестування – Проведено комплексне тестування з використанням синтетичних даних, що підтвердило високу точність виявлення DDoS-атак (більше 98%) та мінімальну кількість хибних спрацьовувань.

П'яте завдання – Оцінка ефективності системи та порівняння з існуючими рішеннями – Підсистема продемонструвала швидку реакцію на аномалії в мережевому трафіку та адаптивність до змін у характері атак, що перевищує можливості традиційних сигнатурних систем.

Таким чином, поставлені на початку роботи завдання були повністю виконані, а розроблена підсистема може бути рекомендована для впровадження у мережеві інфраструктури з метою підвищення рівня захисту від сучасних DDoS-атак.

		Ицук.К.О.			ККІКБ.КР.Б – 125 – NN – ПЗ	Арк.
		Маєвський О.В.				
Змн.	Арк.	№ докум.	Підпис	Дата		62

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. What is a DDoS attack? [Електронний ресурс]. – Microsoft. – Режим доступу: <https://www.microsoft.com/uk-ua/security/business/security-101/what-is-a-ddos-attack>
2. Types of DDoS Attacks [Електронний ресурс]. – NGENIX. – Режим доступу: <https://ngenix.net/learning-center/types-of-ddos-attacks/>
3. 9 найкращих інструментів моніторингу мережевої безпеки [Електронний ресурс]. – Oberig IT. – Режим доступу: <https://oberig-it.com/statti/9-najkrashhyh-instrumentiv-monitoryngu-merezhevoyi-bezpeky-dlya-vyyavlennya-potenczijnyh-zagroz/>
4. Mirkovic J. DDoS attacks [Електронний ресурс] / Jelena Mirkovic. – Режим доступу: https://www.princeton.edu/~rblee/ELE572Papers/Fall04Readings/DDoS_mirkovic.pdf
5. ReasonLabs Cyberpedia. Understanding TCP SYN Flood Attacks [Електронний ресурс]. – Режим доступу: <https://cyberpedia.reasonlabs.com/EN/tcp%2520syn%2520flood.html>
6. Imperva. What is a Ping Flood | ICMP Flood DDoS Attack [Електронний ресурс]. – Режим доступу: <https://www.imperva.com/learn/ddos/ping-icmp-flood>
7. Cloudflare. HTTP Flood DDoS Attack [Електронний ресурс]. – Режим доступу: <https://www.cloudflare.com/ru-ru/learning/ddos/http-flood-ddos-attack/> – Назва з екрана. – Дата звернення: 11.06.2025. The True Cost of a DDoS Attack [Електронний ресурс]. – Corero, – Режим доступу: <https://www.corero.com/true-cost-of-a-ddos-attack/>
8. s41598-025-03868-x [Електронний ресурс] // Nature. – Режим доступу: <https://www.nature.com/articles/s41598-025-03868-x>
9. DDoS Classification using Combined Techniques [Електронний ресурс] // International Journal of Advanced Computer Science and Applications. – Режим доступу: https://thesai.org/Downloads/Volume15No1/Paper_53-DDoS_Classification_using_Combined_Techniques.pdf

		Ицук.К.О.			ККІКБ.КР.Б – 125 – NN – ПЗ	Арк.
		Маєвський О.В.				
Змн.	Арк.	№ докум.	Підпис	Дата		63

10. What is Deep Packet Inspection (DPI)? [Електронний ресурс]. – Indusface. – Режим доступу: <https://www.indusface.com/learning/deep-packet-inspection-dpi/>
11. Network Anomaly Detection [Електронний ресурс]. – Kentik. – Режим доступу: <https://www.kentik.com/kentipedia/network-anomaly-detection/>
12. Kovalov V. et al. DDoS Detection in IoT [Електронний ресурс] // CEUR Workshop Proceedings, Vol. 3241. – Режим доступу: <https://ceur-ws.org/Vol-3241/paper21.pdf>
13. Comparing DPI and Flow-Based Analysis [Електронний ресурс]. – ipoque. – Режим доступу: <https://www.ipoque.com/blog/comparing-dpi-and-flow-based-analysis>
14. Types of DDoS Attacks [Електронний ресурс]. – Riorey. – Режим доступу: <https://www.riorey.com/types-of-ddos-attacks>
15. Global Surge in DDoS Attacks [Електронний ресурс]. – SecurityBrief. – Режим доступу: <https://securitybrief.in/story/global-surge-in-ddos-attacks-causes-dire-financial-consequences>
16. CIC DDoS 2019 Dataset [Електронний ресурс]. – UNB. – Режим доступу: <https://www.unb.ca/cic/datasets/ddos-2019.html>
17. Modern DDoS Threats and Countermeasures [Електронний ресурс]. – TheMoonlight.io. – Режим доступу: <https://www.themoonlight.io/en/review/modern-ddos-threats-and-countermeasures-insights-into-emerging-attacks-and-detection-strategies>
18. Deep Packet Inspection (DPI) [Електронний ресурс]. – Fortinet. – Режим доступу: <https://www.fortinet.com/resources/cyberglossary/dpi-deep-packet-inspection>
19. DDoS Attacks [Електронний ресурс]. – Imperva. – Режим доступу: <https://www.imperva.com/learn/ddos/ddos-attacks/>
20. Taxonomy of DDoS Attacks [Електронний ресурс] // IGI Global. – Режим доступу: <https://www.igi-global.com/chapter/taxonomy-of-distributed-denial-of->

		Ицук.К.О.			ККИКБ.КР.Б – 125 – NN – ПЗ	Арк.
		Маєвський О.В.				
Змн.	Арк.	№ докум.	Підпис	Дата		64

[service-ddos-attacks-and-defense-mechanisms-in-present-era-of-smartphone-devices/261991](https://doi.org/10.3390/s1205261991)

21. Вiкiпедiя. Штучний iнтелект [Електронний ресурс]. – Режим доступу: https://uk.wikipedia.org/wiki/Штучний_iнтелект
22. DDoS Detection with Deep Learning [Електронний ресурс] // Computers & Security. – Режим доступу: <https://www.sciencedirect.com/science/article/abs/pii/S0167404821002479>
23. Mitigating Volumetric DDoS Attacks [Електронний ресурс] // arXiv. – Режим доступу: <https://arxiv.org/abs/2502.19996>
24. Analysis of DDoS Attacks [Електронний ресурс]. – DergiPark. – Режим доступу: <https://dergipark.org.tr/en/download/article-file/3786612>
25. How to Detect DDoS Attacks [Електронний ресурс]. – TechTarget. – Режим доступу: <https://www.techtarget.com/searchsecurity/tip/How-to-detect-DDoS-attacks>
26. DPI for Threat Detection [Електронний ресурс]. – iroque. – Режим доступу: <https://www.ipoque.com/blog/dpi-for-threat-detection>
27. Deep Packet Inspection – Wikipedia [Електронний ресурс]. – Режим доступу: https://en.wikipedia.org/wiki/Deep_packet_inspection
28. F. Ullah et al. Deep Learning-Based Detection [Електронний ресурс] // Computers and Electrical Engineering. – Режим доступу: <https://www.sciencedirect.com/science/article/abs/pii/S0045790622000337>
29. Burgin M. Three Approaches to Artificial Intelligence [Електронний ресурс] // Proceedings, MDPI. – 2022. – 81(1): 147. – Режим доступу: <https://www.mdpi.com/2504-3900/81/1/147> (дата звернення: 11.05.2025)
30. Supervised learning [Електронний ресурс] – Wikipedia. – Режим доступу: https://en.wikipedia.org/wiki/Supervised_learning (дата звернення: 11.05.2025)
31. Unsupervised learning [Електронний ресурс] – MathWorks. – Режим доступу: <https://www.mathworks.com/discovery/unsupervised-learning.html> (дата звернення: 11.05.2025)

		Ицук.К.О.			ККІКБ.КР.Б – 125 – NN – ПЗ	Арк.
		Маєвський О.В.				65
Змн.	Арк.	№ докум.	Підпис	Дата		

32. What is Reinforcement Learning & How Does AI Use It? [Електронний ресурс] – Synopsys. – Режим доступу: <https://www.synopsys.com/glossary/what-is-reinforcement-learning.html> (дата звернення: 11.05.2025)
33. Multilayer Perceptrons in Machine Learning: A Comprehensive Guide [Електронний ресурс] – DataCamp. – Режим доступу: <https://www.datacamp.com/tutorial/multilayer-perceptrons> (дата звернення: 11.05.2025)
34. What is a Convolutional Neural Network (CNN)? [Електронний ресурс] – TechTarget. – Режим доступу: <https://www.techtarget.com/searchenterpriseai/definition/convolutional-neural-network> (дата звернення: 11.05.2025)
35. Autoencoder Definition [Електронний ресурс] – DeepAI. – Режим доступу: <https://deepai.org/machine-learning-glossary-and-terms/autoencoder> (дата звернення: 11.04.2025)
36. What is a GAN? – Generative Adversarial Networks Explained [Електронний ресурс] – AWS. – Режим доступу: <https://aws.amazon.com/what-is/gan> (дата звернення: 11.05.2025)
37. Batch Normalization Explained [Електронний ресурс] – Papers With Code. – Режим доступу: <https://paperswithcode.com/method/batch-normalization> (дата звернення: 11.05.2025)
38. Self-supervised learning [Електронний ресурс] – Wikipedia. – Режим доступу: https://en.wikipedia.org/wiki/Self-supervised_learning (дата звернення: 11.05.2025)
39. Explainable Artificial Intelligence (XAI): Concepts, Taxonomies, Opportunities and Challenges toward Responsible AI [Електронний ресурс] – arXiv. – 2019. – Режим доступу: <https://arxiv.org/abs/1910.10045> (дата звернення: 11.05.2025)
40. Three-stratum theory [Електронний ресурс] – Wikipedia. – Режим доступу: https://en.wikipedia.org/wiki/Three-stratum_theory (дата звернення: 11.05.2025)
41. Long Short-Term Memory (LSTM) [Електронний ресурс]. – Режим доступу: <https://itwiki.dev/data-science/ml-reference/ml-glossary/long-short-term-memory-lstm>

		Ицук.К.О.			ККІКБ.КР.Б – 125 – NN – ПЗ	Арк.
		Маєвський О.В.				66
Змн.	Арк.	№ докум.	Підпис	Дата		

ДОДАТКИ

		Іцук.К.О.			ККІКБ.КР.Б – 125 – NN – ПЗ	Арк.
		Маєвський О.В.				67
Змн.	Арк.	№ докум.	Підпис	Дата		

Лістинг коду підсистеми аналізу мережевого трафіку

```

import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
from sklearn.metrics import classification_report
import tensorflow as tf
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Dense, Dropout
from tensorflow.keras.callbacks import EarlyStopping
import socket
import threading
import json
from datetime import datetime, timedelta
import time
from tkinter import Tk, Label, Button, Frame, Text, Scrollbar, StringVar, END, RIGHT, Y
import matplotlib

matplotlib.use('TkAgg')
from matplotlib.backends.backend_tkagg import FigureCanvasTkAgg

class DDOSDetector:
    def __init__(self):
        # Model and preprocessing
        self.scaler = StandardScaler()
        self.model = self.build_model()

        # Visualization
        self.fig, (self.ax1, self.ax2) = plt.subplots(2, 1, figsize=(10, 8))
        self.setup_plots()

        # Data storage
        self.traffic_data = []
        self.detection_history = []
        self.attack_detected = False
        self.last_alert_time = None
        self.total_flows = 0
        self.total_packets = 0
        self.last_visual_update = datetime.now()

        # Socket server
        self.server_socket = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
        self.server_socket.setsockopt(socket.SOL_SOCKET, socket.SO_REUSEADDR, 1)
        self.server_socket.bind(('localhost', 9999))
        self.server_socket.listen(5)

        # Connection tracking

```

		Іцук.К.О.			ККІКБ.КР.Б – 125 – NN – ПЗ	Арк.
		Маєвський О.В.				68
Змн.	Арк.	№ докум.	Підпис	Дата		

```

self.active_connections = []
self.listening = True

# Start server in background thread
threading.Thread(target=self.run_server, daemon=True).start()

# Create GUI
self.create_gui()

# Start periodic updates
self.update_stats()
self.master.mainloop()

def generate_training_data(self, num_samples=10000):
    """Generate training data with 8 features to match feature extraction"""
    data = []
    labels = []

    # Normal traffic features
    for _ in range(num_samples // 2):
        packets = np.random.randint(1, 100) # Wider range
        bytes = packets * np.random.randint(20, 2000)
        unique_dests = np.random.randint(1, 20)
        flow_rate = np.random.randint(10, 200)
        pkt_var = np.random.uniform(0.1, 0.8)
        src_concentration = np.random.uniform(0.1, 0.5)
        bytes_per_pkt = bytes / packets
        unique_ports = np.random.randint(1, 10)

        data.append([
            packets, bytes, unique_dests, flow_rate,
            pkt_var, src_concentration, bytes_per_pkt, unique_ports
        ])
        labels.append(0)

    # DDoS traffic features
    for _ in range(num_samples // 2):
        packets = np.random.randint(100, 2000)
        bytes = packets * np.random.randint(500, 2500)
        unique_dests = np.random.randint(1, 5)
        flow_rate = np.random.randint(500, 5000)
        pkt_var = np.random.uniform(0.8, 2.0)
        src_concentration = np.random.uniform(0.7, 1.0)
        bytes_per_pkt = bytes / packets
        unique_ports = np.random.randint(1, 3)

        data.append([
            packets, bytes, unique_dests, flow_rate,
            pkt_var, src_concentration, bytes_per_pkt, unique_ports
        ])
        labels.append(1)

```

		Іцук.К.О.			ККІКБ.КР.Б – 125 – NN – ПЗ	Арк.
		Маєвський О.В.				
Змн.	Арк.	№ докум.	Підпис	Дата		69

```

return np.array(data), np.array(labels)

def build_model(self):
    """Build model expecting 8 input features"""
    X, y = self.generate_training_data()
    X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2)

    self.scaler.fit(X_train)
    X_train = self.scaler.transform(X_train)
    X_test = self.scaler.transform(X_test)

    model = Sequential([
        Dense(128, activation='relu', input_shape=(8,)),
        Dropout(0.3),
        Dense(64, activation='relu'),
        Dropout(0.3),
        Dense(1, activation='sigmoid')
    ])

    model.compile(optimizer='adam',
                  loss='binary_crossentropy',
                  metrics=['accuracy'])

    early_stop = EarlyStopping(monitor='val_loss', patience=5)
    model.fit(X_train, y_train, epochs=50,
              validation_data=(X_test, y_test),
              callbacks=[early_stop], verbose=0)

    return model

def extract_features(self, flows):
    """Extract exactly 8 features to match training data"""
    if not flows or len(flows) == 0:
        return None

    df = pd.DataFrame(flows)

    # Calculate all 8 features
    total_packets = df['packets'].sum()
    total_bytes = df['bytes'].sum()
    unique_dests = df['dst_ip'].nunique()
    flow_rate = len(df) / 5 # flows per second (5-second window)

    # Packet variation feature
    packet_std = df['packets'].std()
    packet_mean = df['packets'].mean()
    pkt_var = (packet_std / packet_mean) if packet_mean > 0 else 0

    # Source concentration
    src_concentration = df['src_ip'].nunique() / len(df) if len(df) > 0 else 0

```

		Лущук К.О.			ККІКБ.КР.Б – 125 – NN – ПЗ	Арк.
		Маєвський О.В.				70
Змн.	Арк.	№ докум.	Підпис	Дата		

```

# Bytes per packet
bytes_per_pkt = (total_bytes / total_packets) if total_packets > 0 else 0

# Unique destination ports
unique_ports = df['dst_port'].nunique()

return np.array([
    total_packets,
    total_bytes,
    unique_dests,
    flow_rate,
    pkt_var,
    src_concentration,
    bytes_per_pkt,
    unique_ports
]).reshape(1, -1)

def detect_ddos(self, flows):
    """More conservative detection with threshold adjustment"""
    features = self.extract_features(flows)
    if features is None:
        return False

    features_scaled = self.scaler.transform(features)
    prediction = self.model.predict(features_scaled, verbose=0)

    # Use higher threshold (0.6) to reduce false positives
    return prediction[0][0] > 0.6 # Was 0.5

def run_server(self):
    """Run the socket server to receive traffic data"""
    while self.listening:
        try:
            conn, addr = self.server_socket.accept()
            self.active_connections.append(conn)
            threading.Thread(
                target=self.handle_client,
                args=(conn, addr),
                daemon=True
            ).start()
            self.update_status(f"New connection from {addr}")
        except OSError:
            break

def handle_client(self, conn, addr):
    """Handle incoming client connection"""
    buffer = ""
    try:
        while self.listening:
            data = conn.recv(4096).decode()

```

		Іцук.К.О.			ККІКБ.КР.Б – 125 – NN – ПЗ	Арк.
		Маєвський О.В.				71
Змн.	Арк.	№ докум.	Підпис	Дата		

```

        if not data:
            break

        buffer += data
        while '\n' in buffer:
            message, buffer = buffer.split('\n', 1)
            try:
                # Parse the JSON message into a list of flows
                flows = json.loads(message)
                if isinstance(flows, dict): # Handle single flow case
                    flows = [flows]
                self.process_flows(flows)
            except json.JSONDecodeError as e:
                self.update_status(f"JSON decode error: {str(e)}")
                continue
        except ConnectionError:
            pass
    finally:
        conn.close()
        if conn in self.active_connections:
            self.active_connections.remove(conn)
        self.update_status(f"Connection closed from {addr}")

def process_flows(self, flows):
    """Process incoming network flows"""
    if not flows:
        return

    timestamp = datetime.now()

    # Add metadata to each flow
    processed_flows = []
    for flow in flows:
        if isinstance(flow, dict):
            processed_flow = flow.copy()
            processed_flow['received_time'] = timestamp.isoformat()
            processed_flows.append(processed_flow)

    # Update statistics
    self.total_flows += len(processed_flows)
    self.total_packets += sum(f.get('packets', 0) for f in processed_flows)

    # Store flows
    self.traffic_data.extend(processed_flows)

    # Remove old data (keep last 5 seconds)
    cutoff = timestamp - timedelta(seconds=5)
    self.traffic_data = [
        f for f in self.traffic_data
        if datetime.fromisoformat(f['received_time']) > cutoff
    ]

```

		Ицук.К.О.			ККИКБ.КР.Б – 125 – NN – ПЗ	Арк.
		Маєвський О.В.				72
Змн.	Арк.	№ докум.	Підпис	Дата		

```

# Detect DDoS
is_ddos = self.detect_ddos(self.traffic_data)

# Record detection
self.detection_history.append({
    'time': timestamp,
    'is_ddos': is_ddos,
    'flow_count': len(processed_flows),
    'packet_count': sum(f.get('packets', 0) for f in processed_flows)
})

# Trigger alert if state changed
if is_ddos != self.attack_detected:
    self.attack_detected = is_ddos
    self.trigger_alert(is_ddos)

# Update visualization if enough time passed
if (timestamp - self.last_visual_update).total_seconds() >= 0.5:
    self.update_visualization()
    self.last_visual_update = timestamp

def update_stats(self):
    """Periodically update statistics display"""
    self.stats_var.set(
        f"Total Flows: {self.total_flows} | "
        f"Total Packets: {self.total_packets} | "
        f"Active Connections: {len(self.active_connections)}"
    )
    self.master.after(1000, self.update_stats)

def update_status(self, message):
    """Update status log"""
    self.log_text.insert(END, message + '\n')
    self.log_text.see(END)

def trigger_alert(self, is_ddos):
    """Handle DDoS alert state changes"""
    alert_time = datetime.now()
    alert_msg = f"DDoS ATTACK DETECTED! Time: {alert_time}" if is_ddos else "At-
tack stopped. Normal traffic resumed."

    # Rate Limit alerts
    if self.last_alert_time and (alert_time - self.last_alert_time).total_sec-
onds() < 10:
        return

    self.last_alert_time = alert_time

# Update Log
self.update_status(alert_msg)

```

		Іцук.К.О.			ККІКБ.КР.Б – 125 – NN – ПЗ	Арк.
		Маєвський О.В.				73
Змн.	Арк.	№ докум.	Підпис	Дата		

```

# Visual alert
self.alert_label.config(
    text="ALERT! DDoS ATTACK DETECTED!" if is_ddos else "Normal Traffic",
    bg='red' if is_ddos else 'green',
    fg='white'
)

# Flash alert
self.master.after(1000, lambda: self.alert_label.config(bg='red' if is_ddos
else 'green'))

def setup_plots(self):
    """Initialize the visualization plots"""
    self.ax1.clear()
    self.ax2.clear()

    # Traffic volume plot
    self.ax1.set_title('Traffic Volume (Packets per Second)')
    self.ax1.set_ylabel('Packets')
    self.ax1.set_ylim(0, 5000)
    self.ax1.grid(True)
    self.volume_line, = self.ax1.plot([], [], 'b-', label='Traffic')
    self.attack_line, = self.ax1.plot([], [], 'r-', label='Attack')
    self.ax1.legend()

    # Detection probability plot
    self.ax2.set_title('DDoS Detection Probability')
    self.ax2.set_ylabel('Probability')
    self.ax2.set_ylim(0, 1.1)
    self.ax2.grid(True)
    self.detection_line, = self.ax2.plot([], [], 'g-', label='DDoS Probability')
    self.threshold_line = self.ax2.axhline(y=0.5, color='r', linestyle='--', la-
bel='Threshold')
    self.ax2.legend()

    plt.tight_layout()

def update_visualization(self):
    """Update the visualization with latest data"""
    if not self.detection_history:
        return

    # Get Last 30 seconds of data
    cutoff = datetime.now() - timedelta(seconds=30)
    recent_data = [d for d in self.detection_history if d['time'] > cutoff]

    if not recent_data:
        return

    times = [d['time'] for d in recent_data]

```

		Іцук.К.О.			ККІКБ.КР.Б – 125 – NN – ПЗ	Арк.
		Маєвський О.В.				74
Змн.	Арк.	№ докум.	Підпис	Дата		

```

packets = [d['packet_count'] for d in recent_data]
is_ddos = [d['is_ddos'] for d in recent_data]

# Update traffic volume plot
self.volume_line.set_data(times, packets)

# Highlight attack periods
attack_packets = [p if ddos else 0 for p, ddos in zip(packets, is_ddos)]
self.attack_line.set_data(times, attack_packets)

# Update x-axis limits
self.ax1.set_xlim(min(times), max(times))
self.ax1.relim()
self.ax1.autoscale_view(scalex=False)

# Update detection plot
probabilities = [1.0 if ddos else 0.0 for ddos in is_ddos]
self.detection_line.set_data(times, probabilities)
self.ax2.set_xlim(min(times), max(times))

# Redraw
self.canvas.draw()

def create_gui(self):
    """Create the main application GUI"""
    self.master = Tk()
    self.master.title("DDoS Detection Neural Network")

    # Status frame
    self.status_frame = Frame(self.master)
    self.status_frame.pack(fill='x', padx=10, pady=5)

    self.stats_var = StringVar()
    self.stats_label = Label(self.status_frame, textvariable=self.stats_var,
font=('Arial', 10))
    self.stats_label.pack(side='left')

    # Alert frame
    self.alert_frame = Frame(self.master, padx=10, pady=5)
    self.alert_frame.pack(fill='x')

    self.alert_label = Label(
        self.alert_frame,
        text="Normal Traffic",
        font=('Arial', 14),
        bg='green',
        fg='white',
        padx=20,
        pady=10
    )
    self.alert_label.pack(fill='x')

```

		Іцук.К.О.			ККІКБ.КР.Б – 125 – NN – ПЗ	Арк.
		Маєвський О.В.				75
Змн.	Арк.	№ докум.	Підпис	Дата		

```

# Visualization frame
self.viz_frame = Frame(self.master)
self.viz_frame.pack(fill='both', expand=True)

self.canvas = FigureCanvasTkAgg(self.fig, master=self.viz_frame)
self.canvas.get_tk_widget().pack(fill='both', expand=True)

# Log frame
self.log_frame = Frame(self.master)
self.log_frame.pack(fill='both', expand=True)

self.log_label = Label(self.log_frame, text="Detection Log:", font=('Arial',
12))
self.log_label.pack(anchor='w')

self.log_text = Text(self.log_frame, height=8, wrap='word')
self.scrollbar = Scrollbar(self.log_frame, command=self.log_text.yview)
self.log_text.config(yscrollcommand=self.scrollbar.set)

self.scrollbar.pack(side=RIGHT, fill=Y)
self.log_text.pack(side='left', fill='both', expand=True)

# Control frame
self.control_frame = Frame(self.master, pady=10)
self.control_frame.pack()

self.exit_button = Button(
    self.control_frame,
    text="Exit",
    command=self.shutdown,
    padx=20
)
self.exit_button.pack()

def shutdown(self):
    """Clean shutdown of application"""
    self.listening = False
    for conn in self.active_connections:
        conn.close()
    self.server_socket.close()
    self.master.quit()

if __name__ == "__main__":
    DDOSDetector()

```

		Іцук.К.О.			ККІКБ.КР.Б – 125 – NN – ПЗ	Арк.
		Маєвський О.В.				
Змн.	Арк.	№ докум.	Підпис	Дата		76

Лістинг коду модуля для генерації Netflow трафіку

```

import random
import time
import json
from datetime import datetime
import socket
import threading
from tkinter import Tk, Label, Button, Checkbutton, IntVar, Entry, Frame, filedialog,
messagebox, StringVar

class NetFlowGenerator:
    def __init__(self, master):
        self.master = master
        master.title("NetFlow Traffic Generator")

        # Connection status variable
        self.connection_status = StringVar()
        self.connection_status.set("Disconnected")

        # GUI Elements
        self.label = Label(master, text="NetFlow Traffic Generator", font=('Arial',
16))
        self.label.pack(pady=10)

        # Connection frame
        self.conn_frame = Frame(master)
        self.conn_frame.pack(pady=5)
        self.conn_label = Label(self.conn_frame, text="Neural Network Connection:")
        self.conn_label.pack(side='left')
        self.conn_status = Label(self.conn_frame, textvariable=self.connection_status, fg="red")
        self.conn_status.pack(side='left', padx=5)
        self.connect_btn = Button(self.conn_frame, text="Connect", command=self.connect_to_nn)
        self.connect_btn.pack(side='left')

        # Regular traffic controls
        self.regular_frame = Frame(master)
        self.regular_frame.pack(pady=5)
        self.regular_label = Label(self.regular_frame, text="Regular Traffic:")
        self.regular_label.pack(side='left')
        self.regular_start = Button(self.regular_frame, text="Start", command=self.start_regular)
        self.regular_start.pack(side='left', padx=5)
        self.regular_stop = Button(self.regular_frame, text="Stop", command=self.stop_regular, state='disabled')
        self.regular_stop.pack(side='left')

```

		Іцук.К.О.			ККІКБ.КР.Б – 125 – NN – ПЗ	Арк.
		Маєвський О.В.				77
Змн.	Арк.	№ докум.	Підпис	Дата		

```

# DDoS attack controls
self.ddos_frame = Frame(master)
self.ddos_frame.pack(pady=5)
self.ddos_var = IntVar()
self.ddos_check = Checkbutton(self.ddos_frame, text="DDoS Attack", varia-
ble=self.ddos_var)
self.ddos_check.pack(side='left')
self.ddos_start = Button(self.ddos_frame, text="Start", com-
mand=self.start_ddos)
self.ddos_start.pack(side='left', padx=5)
self.ddos_stop = Button(self.ddos_frame, text="Stop", command=self.stop_ddos,
state='disabled')
self.ddos_stop.pack(side='left')

# Duration control
self.duration_frame = Frame(master)
self.duration_frame.pack(pady=5)
self.duration_label = Label(self.duration_frame, text="Duration (sec):")
self.duration_label.pack(side='left')
self.duration_entry = Entry(self.duration_frame, width=10)
self.duration_entry.insert(0, "60")
self.duration_entry.pack(side='left')

# Output options
self.output_frame = Frame(master)
self.output_frame.pack(pady=5)
self.save_button = Button(self.output_frame, text="Save to File", com-
mand=self.save_to_file)
self.save_button.pack(side='left', padx=5)

# Status
self.status_frame = Frame(master)
self.status_frame.pack(pady=10)
self.status_label = Label(self.status_frame, text="Status: Ready",
fg="green")
self.status_label.pack()

# Traffic generation control
self.is_running = False
self.generated_flows = []
self.nn_socket = None
self.connected = False

def connect_to_nn(self):
    """Connect to the neural network detector"""
    try:
        if self.connected:
            self.nn_socket.close()
            self.connected = False
            self.connection_status.set("Disconnected")
            self.connect_btn.config(text="Connect")

```

		Іщук К.О.			ККІКБ.КР.Б – 125 – NN – ПЗ	Арк.
		Маєвський О.В.				
Змн.	Арк.	№ докум.	Підпис	Дата		78

```

        self.update_status("Disconnected from neural network", "red")
        return

    self.nn_socket = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
    self.nn_socket.connect(('localhost', 9999))
    self.connected = True
    self.connection_status.set("Connected")
    self.connect_btn.config(text="Disconnect")
    self.update_status("Connected to neural network", "green")

except Exception as e:
    messagebox.showerror("Connection Error", f"Failed to connect: {str(e)}")
    self.connection_status.set("Connection Failed")
    self.update_status(f"Connection failed: {str(e)}", "red")

def generate_flow(self, is_ddos=False):
    """Generate a single NetFlow record"""
    timestamp = datetime.now().isoformat()
    src_ip = f"10.0.0.{random.randint(1, 254)}"

    if is_ddos:
        if random.random() > 0.8:
            src_ip = f"192.168.0.{random.randint(1, 10)}"
        dst_ip = "10.0.0.1"
        dst_port = 80
        packets = random.randint(100, 1000)
        bytes = packets * random.randint(500, 1500)
        protocol = random.choice([6, 17])
    else:
        dst_ip = f"10.0.0.{random.randint(2, 254)}"
        dst_port = random.choice([80, 443, 22, 53, 3389])
        packets = random.randint(1, 50)
        bytes = packets * random.randint(40, 1500)
        protocol = random.choice([6, 17, 1])

    flow = {
        "timestamp": timestamp,
        "src_ip": src_ip,
        "dst_ip": dst_ip,
        "src_port": random.randint(1024, 65535),
        "dst_port": dst_port,
        "protocol": protocol,
        "packets": packets,
        "bytes": bytes,
        "flags": random.randint(0, 255),
        "is_ddos": is_ddos
    }

    # Send to NN if connected
    if self.connected:
        try:

```

		Іщук.К.О.			ККІКБ.КР.Б – 125 – NN – ПЗ	Арк.
		Маєвський О.В.				
Змн.	Арк.	№ докум.	Підпис	Дата		79

```

        self.nn_socket.sendall(json.dumps(flow).encode() + b'\n')
    except Exception as e:
        self.update_status(f"Send error: {str(e)}", "red")
        self.connected = False
        self.connection_status.set("Disconnected")
        self.connect_btn.config(text="Connect")

    return flow

def generate_traffic(self, is_ddos=False, duration=60):
    """Generate traffic for specified duration"""
    self.is_running = True
    start_time = time.time()
    flow_count = 0

    try:
        while self.is_running and (time.time() - start_time) < duration:
            if is_ddos:
                flows_per_second = random.randint(500, 2000)
                time.sleep(0.001)
            else:
                flows_per_second = random.randint(10, 100)
                time.sleep(0.1)

            for _ in range(flows_per_second):
                if not self.is_running:
                    break

                flow = self.generate_flow(is_ddos)
                self.generated_flows.append(flow)
                flow_count += 1

            self.update_status(f"Generating {'DDoS' if is_ddos else 'regular'}
traffic: {flow_count} flows", "blue")

            self.update_status(f"Finished generating {flow_count} flows", "green")
        except Exception as e:
            self.update_status(f"Error: {str(e)}", "red")
        finally:
            self.is_running = False
            self.regular_start.config(state='normal')
            self.regular_stop.config(state='disabled')
            self.ddos_start.config(state='normal')
            self.ddos_stop.config(state='disabled')

    def start_regular(self):
        try:
            duration = int(self.duration_entry.get())
            if duration <= 0:
                raise ValueError
        except ValueError:

```

		Лук.К.О.			ККІКБ.КР.Б – 125 – NN – ПЗ	Арк.
		Маєвський О.В.				80
Змн.	Арк.	№ докум.	Підпис	Дата		

```

        messagebox.showerror("Error", "Please enter a valid positive number for
duration")
        return

    self.regular_start.config(state='disabled')
    self.regular_stop.config(state='normal')
    self.ddos_start.config(state='disabled')

    threading.Thread(target=self.generate_traffic,
                    args=(False, duration),
                    daemon=True).start()

def stop_regular(self):
    self.is_running = False

def start_ddos(self):
    try:
        duration = int(self.duration_entry.get())
        if duration <= 0:
            raise ValueError
    except ValueError:
        messagebox.showerror("Error", "Please enter a valid positive number for
duration")
        return

    self.ddos_start.config(state='disabled')
    self.ddos_stop.config(state='normal')
    self.regular_start.config(state='disabled')

    threading.Thread(target=self.generate_traffic,
                    args=(True, duration),
                    daemon=True).start()

def stop_ddos(self):
    self.is_running = False

def save_to_file(self):
    if not self.generated_flows:
        messagebox.showwarning("Warning", "No traffic data to save")
        return

    filename = filedialog.asksaveasfilename(
        defaultextension=".json",
        filetypes=[("JSON files", "*.json"), ("All files", "*.")]
    )

    if filename:
        try:
            with open(filename, 'w') as f:
                json.dump(self.generated_flows, f, indent=2)

```

		Лущук.К.О.			ККІКБ.КР.Б – 125 – NN – ПЗ	Арк.
		Маєвський О.В.				81
Змн.	Арк.	№ докум.	Підпис	Дата		

```

        self.update_status(f"Saved {len(self.generated_flows)} flows to
{filename}", "green")
    except Exception as e:
        messagebox.showerror("Error", f"Failed to save file: {str(e)}")

def update_status(self, message, color="black"):
    self.status_label.config(text=f"Status: {message}", fg=color)
    self.master.update()

if __name__ == "__main__":
    root = Tk()
    app = NetFlowGenerator(root)
    root.mainloop()

```

		Іцук.К.О.			ККІКБ.КР.Б – 125 – NN – ПЗ	Арк.
		Маєвський О.В.				
Змн.	Арк.	№ докум.	Підпис	Дата		82