МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ ЧЕРКАСЬКИЙ ДЕРЖАВНИЙ ФАХОВИЙ БІЗНЕС-КОЛЕДЖ
Циклова комісія комп’ютерної інженерії та інформаційних технологій

КВАЛІФІКАЦІЙНА РОБОТА

на тему

ІНСТРУМЕНТИ ВИЯВЛЕННЯ ФІШИНГОВИХ АТАК З ВИКОРИСТАННЯМ НЕЙРОННОЇ МЕРЕЖІ

Виконав:
студент групи 1К-22 спеціальності
123 комп'ютерна інженерія
Владислав СОКУРЕНКО
Керівник Павло РАТАЙЧУК

Черкаси 2026

ЧЕРКАСЬКИЙ ДЕРЖАВНИЙ ФАХОВИЙ БІЗНЕС-КОЛЕДЖ

Циклова комісія комп’ютерної інженерії та інформаційних технологій

Спеціальність 123 «Комп’ютерна інженерія»

Освітня програма Комп’ютерна інженерія

ЗАТВЕРДЖУЮ

Завідувачка циклової комісії КІ та ІТ

_____Наталя ФАЛЬЧЕНКО

(підпис)

«_____» _____2025 р.

З А В Д А Н Н Я

НА КВАЛІФІКАЦІЙНУ РОБОТУ СТУДЕНТУ

Сокуренку Владиславу Ігоровичу

1. Тема кваліфікаційної роботи Інструменти виявлення фішингових атак з використанням нейронної мережі
Керівник роботи Ратайчук Павло Єгорович, викладач вищої категорії затверджені наказом закладу вищої освіти від «06» жовтня 2025 року № 84/1у.
2. Строк подання студентом кваліфікаційної роботи 08.06.2026
3. Вихідні дані до роботи: наукові публікації та технічна література з кібербезпеки; офіційна документація мови програмування Python та бібліотеки TensorFlow; набори даних datasets з фішинговими та легітимними URL-адресами; звіти міжнародних організацій APWG щодо статистики кіберзагроз.
4. Зміст пояснювальної записки (перелік питань, які потрібно розробити): аналіз методів проведення фішингових атак; огляд існуючих систем захисту та виявлення шкідливих вебресурсів; дослідження алгоритмів машинного та глибокого навчання для класифікації даних; проєктування архітектури нейронної мережі для аналізу вебсайтів; підготовка робочого набору даних; програмна реалізація інструменту мовою Python з використанням TensorFlow; тестування навченої моделі; оцінка точності та ефективності системи за допомогою матриці плутанини Confusion Matrix та ROC-кривих.
5. Дата видачі завдання 15.09.2025

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів кваліфікаційної роботи	Терміни виконання етапів	Примітка про виконання
1	ВСТУП	14.10.2026	
2	РОЗДІЛ 1 ТЕОРЕТИЧНІ ОСНОВИ ВИЯВЛЕННЯ ФІШИНГОВИХ АТАК У КОМП'ЮТЕРНИХ МЕРЕЖАХ	09.12.2026	
3	РОЗДІЛ 2 АНАЛІЗ ІНСТРУМЕНТІВ ТА МЕТОДІВ ВИЯВЛЕННЯ ФІШИНГОВИХ АТАК	10.03.2026	
4	РОЗДІЛ 3. ПРАКТИЧНЕ МОДЕЛЮВАННЯ СИСТЕМИ ВИЯВЛЕННЯ ФІШИНГОВИХ АТАК З ВИКОРИСТАННЯМ НЕЙРОННОЇ МЕРЕЖІ	28.04.26	
5	ВИСНОВКИ	12.05.26	
6	Оформлення кваліфікаційної роботи	26.05.26	
7	Здача кваліфікаційної роботи на кафедру для рецензування		
8	Перевірка кваліфікаційної роботи на наявність ознак плагіату	02.06.26	
9	Подання кваліфікаційної роботи на затвердження завідувачу кафедри	10.06.26	

Студент _____ _____

(підпис) (ім'я та прізвище)

Керівник роботи _____ _____

(підпис) (ім'я та прізвище)

АНОТАЦІЯ

Кваліфікаційної роботи Владислава Сокуренка на тему Інструменти виявлення фішингових атак з використанням нейронної мережі

Робота описує створення програмного інструменту для розпізнавання фішингових вебресурсів. Зловмисники постійно знаходять нові способи обману користувачів у мережі. Вони майстерно маскують шкідливі посилання під сторінки відомих брендів чи банків. Щоб автоматизувати боротьбу з таким шахрайством, автор спроєктував та навчив штучну нейронну мережу.

Програмну частину реалізовано мовою Python версії 3.12. Для побудови моделі застосовано бібліотеку TensorFlow. Студент зібрав необхідні датасети, налаштував архітектуру алгоритму машинного навчання та провів тренування. Якість роботи системи перевірено за допомогою конкретних інструментів: матриці плутанини та аналізу ROC-кривих. Результати тестування довели здатність нейромережі правильно розрізняти справжні та підроблені URL-адреси.

Готова модель має пряме практичне застосування. Її можна впровадити у корпоративні системи кіберзахисту або використовувати як основу для безпекового розширення браузера. Текст містить послідовний опис усіх етапів розробки, від первинного сортування даних до розрахунку точності класифікації.

Ключові слова: КІБЕРБЕЗПЕКА, ФІШИНГ, СОЦІАЛЬНА ІНЖЕНЕРІЯ, ГЛИБОКЕ НАВЧАННЯ, НЕЙРОННІ МЕРЕЖІ, БАГАТОШАРОВИЙ ПЕРЦЕПТРОН, MLP, PYTHON, TENSORFLOW, KERAS.

ANNOTATION

For the qualification work of Vladyslav Sokurenko on the topic "Tools for detecting phishing attacks using a neural network"

The text describes the creation of a software tool that recognizes phishing web resources. Attackers constantly find new ways to deceive internet users. They disguise malicious links as pages of well-known brands or banks. To automate the fight against this fraud, the author designed and trained an artificial neural network.

The software part is written in Python 3.12. The student used the TensorFlow library to build the model. He collected the necessary datasets, configured the machine learning algorithm's architecture, and performed the training. The system's quality was verified using specific metrics: a Confusion Matrix and ROC curve analysis. Testing results proved the neural network's ability to distinguish between real and fake URLs.

The finished model has direct practical application. Information security teams can implement it in corporate defense systems, or developers can use it as a baseline for a browser extension. The document outlines every stage of development, starting from raw data processing to the final classification accuracy calculation.

Keywords: CYBERSECURITY, PHISHING, SOCIAL ENGINEERING, DEEP LEARNING, NEURAL NETWORKS, MULTILAYER PERCEPTRON, MLP, PYTHON, TENSORFLOW, KERAS.

ЗМІСТ

ВСТУП 4

РОЗДІЛ 1 ТЕОРЕТИЧНІ ОСНОВИ ВИЯВЛЕННЯ ФІШИНГОВИХ АТАК У КОМП'ЮТЕРНИХ МЕРЕЖАХ **6**

1.1 Поняття та класифікація фішингових атак 6

1.2.1 Фішингові веб-сайти 8

1.2.2 Підроблені електронні листи 9

1.2.3 Атаки через соціальні мережі 12

1.2.4 SMS-фішинг та мобільний фішинг 13

1.3 Методи захисту від фішингових атак 14

1.3.1 Фільтрація електронної пошти 15

1.3.2 Чорні та білі списки 15

1.3.3 Аналіз URL-адрес 16

1.3.4 Поведінковий аналіз 17

1.4 Використання машинного навчання для виявлення кіберзагроз 20

1.5 Нейронні мережі та їх застосування у кібербезпеці 23

1.6 Основні параметри ефективності систем виявлення атак 27

1.6.1 Точність 28

1.6.2 Точність класифікації 29

1.6.3 параметр чутливість 29

1.6.4 F1-міра 30

РОЗДІЛ 2 АНАЛІЗ ІНСТРУМЕНТІВ ТА МЕТОДІВ ВИЯВЛЕННЯ ФІШИНГОВИХ АТАК **31**

2.1 Аналіз сучасних систем виявлення фішингових атак 31

2.1.1 Системи на основі чорних списків та сигнатур 31

2.1.2 Системи евристичного аналізу 32

2.1.3 Систем на основі машинного та глибокого навчання 32

2.1.4 Системи аналізу візуальної схожості 33

2.1.5 Комплексні гібридні системи виявлення 33

2.2 Інструменти аналізу шкідливих веб-ресурсів 34

2.2.1 Системи перевірки 35
2.2.2 Антифішингові сервіси 35
2.2.3 Браузерні механізми захисту 36
2.3 Методи аналізу фішингових веб-сайтів 36
2.3.1 Аналіз структури URL 37
2.3.2 Аналіз HTML-коду та контенту 37
2.3.3 Аналіз доменних характеристик 38
2.4 Застосування алгоритмів машинного навчання для виявлення фішингу 38
2.4.1 Основні класи алгоритмів та їх застосування 39
2.4.2 Процес побудови моделі 39
2.4.3 Виклики та обмеження сучасних ML-систем 40
2.5 Порівняльний аналіз алгоритмів класифікації 40
2.6. Формування вимог до системи виявлення фітингових атак 43
РОЗДІЛ 3 ПРАКТИЧНЕ МОДЕЛЮВАННЯ СИСТЕМИ ВИЯВЛЕННЯ ФІШИНГОВИХ АТАК З ВИКОРИСТАННЯМ НЕЙРОННОЇ МЕРЕЖІ 45
3.1 Постановка задачі практичного дослідження 45
3.2. Вибір програмного середовища для реалізації системи 46
3.3 Формування набору даних для навчання нейронної мережі 49
3.4 Побудова архітектури нейронної мережі 52
3.5 Організація процесу навчання та балансування класів 55
3.6. Проведення експериментів з різними параметрами моделі 57
3.7. Оцінювання ефективності роботи моделі 59
3.8. Аналіз результатів моделювання 61
3.9. Рекомендації щодо використання нейронних мереж для виявлення фішингових атак 62
СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ 66

ВСТУП

Актуальність теми. Зі стрімким розвитком інформаційних технологій, цифровізацією послуг та глобалізацією мережі Інтернет пропорційно зростає і кількість кіберзагроз. Одним із найпоширеніших та найнебезпечніших векторів атак залишається фішинг -вид соціальної інженерії та комп'ютерного шахрайства, головною метою якого є несанкціоноване отримання конфіденційних даних користувачів. За даними міжнародних звітів з кібербезпеки, кількість фішингових вебсайтів та розсилок щороку оновлює історичні максимуми, завдаючи значних фінансових та репутаційних збитків [11].

Традиційні підходи до виявлення фішингу, що базуються на статичних чорних списках або простому евристичному аналізі, демонструють зниження ефективності через швидку зміну тактик зловмисників. У зв’язку з цим, застосування алгоритмів штучного інтелекту стає критично необхідним для автоматичного виявлення загроз [4, 7]. Використання методів глибокого навчання, зокрема багатошарового перцептрона MLP, дозволяє ефективно виявляти приховані нелінійні закономірності в ознаках вебсторінок та URL-адрес, адаптуючись до нових видів атак, що робить обрану тему дослідження надзвичайно актуальною.

Мета і завдання дослідження. Метою роботи є підвищення ефективності виявлення фішингових атак шляхом розробки та програмної реалізації класифікаційної моделі на базі глибокої нейронної мережі.

Для досягнення поставленої мети було вирішено такі завдання:

– Провести аналіз предметної області, дослідити сутність фішингу та обмеження традиційних систем захисту.

– Дослідити теоретичні засади функціонування багатошарового перцептрона MLP та обрати метрики для оцінки якості класифікації.

– Проаналізувати та підготувати набір даних оптимізувавши його та вирішивши проблему дисбалансу класів для коректного навчання моделі.

– Здійснити програмну реалізацію нейромережі з використанням мови програмування Python та сучасних бібліотек машинного навчання TensorFlow і Keras.

– Провести експериментальне тестування розробленої моделі, проаналізувати результати навчання за допомогою матриці плутанини та ROC-кривої.

Об’єкт дослідження: процес автоматизованого виявлення та блокування кіберзагроз у комп’ютерних мережах.

Предмет дослідження: методи, алгоритми глибокого навчання, зокрема нейронна мережа прямого поширення MLP, та програмні засоби для класифікації фішингових вебресурсів.

Розроблена програмна модель на базі багатошарового перцептрона може бути використана як складова частина комплексних систем виявлення вторгнень, інтегрована у поштові шлюзи для фільтрації спаму або використана як розширення для браузерів для попередження користувачів про небезпеку в режимі реального часу.

РОЗДІЛ 1 ТЕОРЕТИЧНІ ОСНОВИ ВИЯВЛЕННЯ ФІШИНГОВИХ АТАК У КОМП'ЮТЕРНИХ МЕРЕЖАХ

1.1 Поняття та класифікація фішингових атак

Фішинг у сучасних глосаріях розглядають як техніку виманювання чутливих даних через шахрайські повідомлення або підроблені вебресурси, коли зловмисник видає себе за легітимну організацію чи особу. Важливо, що у визначеннях підкреслюється "соціально-інженерна" природа: для успіху атаки потрібна взаємодія жертви з контрафактним ресурсом або суб'єктом.

У практичній кібероборoні фішинг часто є способом початкового доступу до систем через вкладення або посилання, а також інструментом обману для обходу контролів аутентифікації та фінансових процедур. У звітах щодо ландшафту загроз фішинг і соціальна інженерія системно фігурують як стійкі та довгоживучі практики кіберзлочинності й шпигунства, що не залежать від конкретної технологічної платформи [3].

Класифікація фішингу може будуватися за кількома осями, див. табл.1.1.

За каналом доставки - електронна пошта, SMS або месенджери, соціальні мережі та сторонні сервіси, голосові дзвінки, а також комбіновані сценарії, наприклад, лист із проханням "подзвонити за номером" або перейти за QR-кодом.

За ступенем таргетування - масовий фішинг і цільовий, що орієнтується на конкретну організацію чи посаду та використовує контекстні деталі: проєкти, постачальників, ланцюжки переписки.

За метою - викрадення облікових даних, шахрайство з платіжними реквізитами, компрометація ділового листування BEC - доставка шкідливого програмного забезпечення, збір розвідданих для подальших атак.

Таблиця 1.1 – Узагальнена таксономія фішингових атак за каналами та векторами

Канал / середовище	Типовий вектор атаки	Очікуваний артефакт у мережі	Типова ціль	Приклади сигналів для детекції
Електронна пошта	Посилання/вкладення, підміна відправника, “ланцюжки”	URL, заголовки листа, вкладення, домени	Облікові дані, доставка ПЗ, BEC	DKIM/SPF/DMARC-невідповідності; нетиповий домен; короткі URL; вкладення з подвійними розширеннями
Вебсайт	Підроблена сторінка входу/платежу	HTML/DOM, форми, скрипти, сертифікат, ресурси	Credential harvesting	Висока візуальна/DOM-подібність; підозрілі скрипти; молодий домен; нетиповий хостинг
Соцмережі / сторонні сервіси	Повідомлення через “зовнішні” канали, запрошення/”робота”	URL, акаунти, логи сервісу	Початковий доступ	Взаємодія з некорпоративними каналами; короткоживучі акаунти
SMS / QR	Посилання в SMS, QR-код на платіж/верифікацію	Короткі URL, редіректи, домени	Платежі, дані	Нетипові домени; ланцюжки редіректів; домени в “нових” TLD
Голос (vishing)	Дзвінок із примусом дії	Поєднання голос + підказка URL/кодів	Виманювання кодів/платежів	Ухилення від письмових каналів; тиск терміновістю

У таблиці 1.1 систематизовано ключові види фішингових атак. Структура таблиці охоплює п'ять основних критеріїв аналізу: канал доставки загрози пошта, вебсайти, соцмережі, мобільний зв'язок, голос, безпосередній вектор атаки, залишені в системі цифрові артефакти, кінцеву мету зловмисників та характерні маркери для детекції. Таке узагальнення дозволяє чітко пов'язати теоретичні методи соціальної інженерії з практичними індикаторами компрометації IoC, які використовуються сучасними системами безпеки для виявлення та блокування загроз.

Реалізація фішингової атаки зазвичай спирається на використання багаторівневих векторів доставки шкідливого контенту. Для систематизації та

глибокого аналізу механізмів кібератак доцільно детально розглянути чотири фундаментальні методи їх реалізації - фішингові вебсайти, підроблені електронні листи, атаки через соціальні мережі та SMS-фішинг [11].

1.2.1 Фішингові веб-сайти

Фішингові вебсайти або веб-сурогати є кінцевою точкою верифікації більшості фішингових кампаній. Їхня головна мета - ексфільтрація або викрадення автентифікаційних даних користувачів: логінів, паролів, сесійних токенів, або фінансової інформації, як то номерів кредитних карток, CVV/CVC кодів, PIN-кодів.

Процес створення та функціонування фішингового вебресурсу включає кілька складних технічних етапів:

Клонування інтерфейсу, коли зловмисники використовують спеціалізоване програмне забезпечення, наприклад, HTTrack або вбудовані інструменти розробника браузера, для повного копіювання вихідного коду: HTML, CSS, JavaScript, та графічних елементів легітимного сайту, наприклад, інтернет-банкінгу, соціальної мережі, корпоративного порталу. Це забезпечує максимальну візуальну автентичність.

Маніпуляції з доменними іменами - для введення користувача в оману застосовуються специфічні техніки реєстрації доменів:

Тайпосквотинг - реєстрація доменів із навмисними друкарськими помилками наприклад, bnakofamerica.com замість bankofamerica.com.

Комбосквотинг - додавання до легітимного бренду лексичних маркерів безпеки або дії наприклад, security-paypal.com, google-login-verify.net.

Міжнародні гомографічні атаки - використання символів з різних алфавітів наприклад, кириличної “а” замість латинської “a”, які візуально виглядають ідентично в адресному рядку браузера, але мають абсолютно різні Unicode-коди.

Глибока ієрархія піддоменів - створення піддоменів на підконтрольному зловмиснику домені з метою винесення справжнього імені за межі видимості

адресного рядка, наприклад, www.privat24.ua.secure.login.id-9384.attackersite.com).

Легітимізація за допомогою SSL/TLS - сучасні фішингові ресурси у понад 80% випадків використовують діючі SSL/TLS сертифікати, отримані через безкоштовні автоматизовані центри сертифікації Let's Encrypt. Наявність протоколу HTTPS та символу замочка в адресній системі браузера помилково сприймається більшістю користувачів як гарантія безпеки самого сайту, хоча насправді це свідчить лише про шифрування трафіку між користувачем та фішинговим сервером.

Техніки ухилення від детекції - для боротьби з автоматизованими сканерами безпеки та вебкраулерами антивірусних компаній розробники фішингових скриптів застосовують технологію клакінгу. Сайт аналізує IP-адресу, User-Agent та геолокацію відвідувача. Якщо запит надходить від бота безпеки або з IP-діапазону технологічних гігантів Google, Microsoft, Cloudflare, система відображає легітимну сторінку 404 або нейтральний контент. Якщо ж заходить реальний користувач із цільового регіону - завантажується фішинговий інтерфейс.

1.2.2 Підроблені електронні листи

Електронна пошта залишається первинним і найбільш масовим вектором поширення фішингу. Популярність цього методу зумовлена архітектурними особливостями базового протоколу передачі пошти SMTP, який історично розроблявся без вбудованих механізмів суворої автентифікації відправника.

Технічні аспекти реалізації поштового фішингу включають:

Супуфінг заголовків - Можливість підміни текстового поля From - у заголовку MIME-повідомлення. Зловмисник може вказати будь-яку адресу наприклад, admin@microsoft.com, і поштовий клієнт жертви відобразить саме її, якщо на поштовому сервері-отримувачі не налаштовані або неправильно конфігуровані захисні політики [7,9].

Обхід технологій фільтрації - сучасні системи захисту пошти SEG використовують три основні стандарти для перевірки автентичності листів:

SPF - перевіряє, чи дозволено IP-адресі відправника надсилати листи від імені конкретного домену.

DKIM додає до листа цифровий підпис, пов'язаний із доменом відправника, що гарантує цілісність контенту.

Domain-based Message Authentication, Reporting, and Conformance - визначає політику обробки листів пропускати, у карантин чи блокувати, які не пройшли перевірку SPF або DKIM. Зловмисники обходять ці бар'єри шляхом використання скомпрометованих легітимних поштових серверів сторонніх організацій, експлуатації неправильних налаштувань DMARC наприклад, коли параметр політики виставлений як p=none, або надсилання листів із нових, спеціально зареєстрованих доменів, які повністю проходять перевірки SPF/DKIM, але візуально схожі на цільові компанії.

Впровадження шкідливого навантаження - кисти можуть містити або прямі посилання на фішингові вебсайти, або шкідливі вкладення документи MS Office із макросами, архівовані скрипти JS/VBS/EXE, інфіковані PDF-файли), які ініціюють завантаження шкідливого ПЗ Malware, Trojan-Spy, Ransomware на пристрій жертви.

Цільовий фішинг та BEC-атаки- Високотехнологічний підвид атак, де листи готуються індивідуально для конкретного співробітника компанії азвичай бухгалтерів або топменеджерів - У рамках атак типу Business Email Compromise BEC зловмисники підробляють або компрометують листування з контрагентами, надсилаючи змінені інвойси на оплату, що призводить до значних фінансових збитків без безпосереднього використання вірусного софту. загальну концепцію та технічні аспекти атаки проілюстровано на рис. 1.1

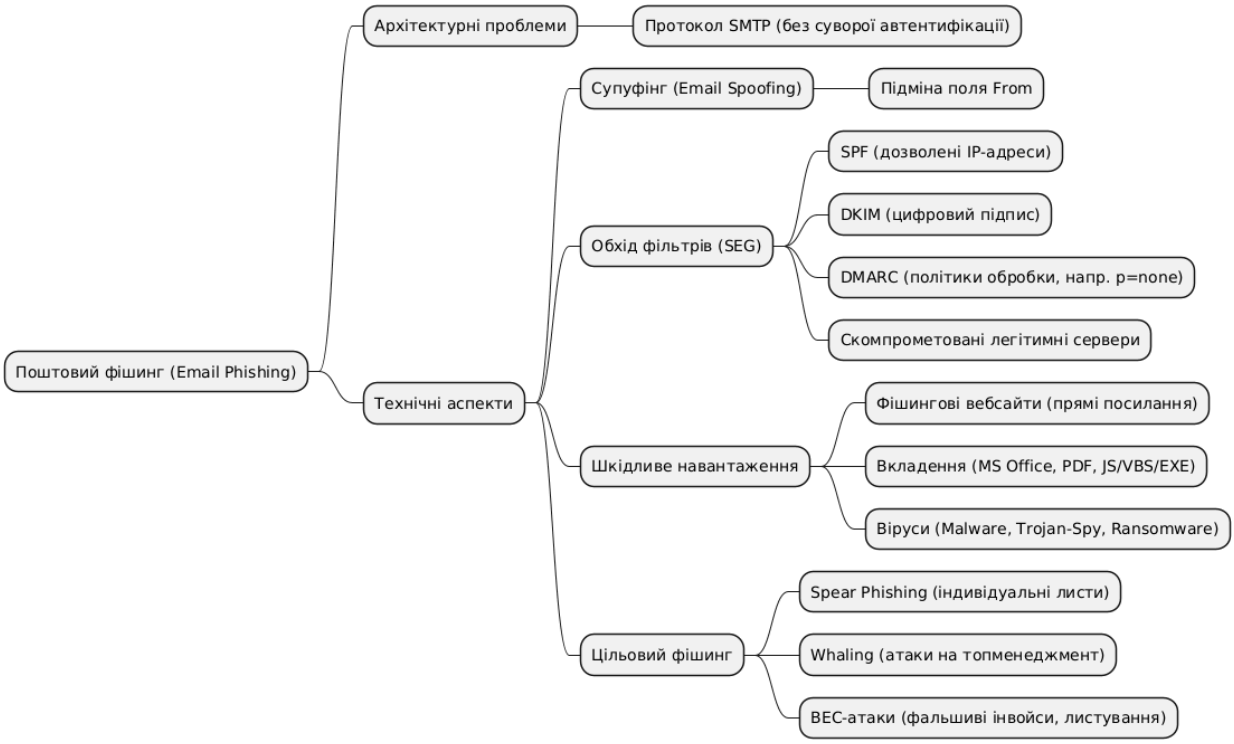


Рисунок 1.1 Структура та технічні аспекти поштового фішингу

На рисунку 1.1 наведено структурну схему поштового фішингу. Схема поділяє проблему на дві основні категорії: архітектурні проблеми та технічні аспекти.

Гілка архітектурних проблем вказує на вразливість базового протоколу передачі пошти SMTP. Цей протокол від початку працює без суворої автентифікації відправника.

Гілка технічних аспектів деталізує чотири вектори атак:

Супуфінг реалізується через підміну поля From у заголовку електронного листа. Наприклад, зловмисник відправляє повідомлення нібито від служби підтримки банку вашої компанії.

Обхід фільтрів безпеки електронної поштиSEG включає маніпуляції з механізмами захисту доменів. Атакувальники обходять перевірки дозволених IP адрес SPF, експлуатують відсутність цифрових підписів DKIM та використовують слабкі політики DMARC (наприклад, параметр p=none). Зловмисники також розсилають листи через вже скомпрометовані легітимні сервери, щоб обдурити фільтри.

Шкідливе навантаження доставляється жертві трьома способами. Перший спосіб це прямі посилання на фішингові вебсайти. Другий спосіб це вкладення у форматах документів MS Office, PDF або скриптів JS, VBS, EXE. Третій спосіб це доставка шкідливого програмного забезпечення, троянів та програм вимагачів

Цільовий фішинг фокусується на конкретних особах чи організаціях. Він включає Spear Phishing індивідуально підготовлені листи для конкретного працівника, Whaling атаки на топменеджмент компанії та атаки формату BEC. Атаки формату BEC зазвичай містять фальшиві інвойси або підроблене ділове листування для перехоплення корпоративних грошових переказів.

1.2.3 Атаки через соціальні мережі

Зі зростанням популярності платформ корпоративних та особистих комунікацій LinkedIn, Facebook, Telegram, WhatsApp, X/Twitter зловмисники почали активно зміщувати вектор атак у соціальні мережі. Головна особливість цього методу полягає у можливості повного обходу традиційних корпоративних засобів захисту периметра мережі та поштових шлюзів SEG.

Механізми реалізації атак у соціальних мережах включають:

Компрометація або клонування облікових записів - зловмисник зламує акаунт користувача або створює його повний дублікат використовуючи публічно доступні фото та особисті дані. Надалі шкідливі посилання чи файли розсилаються списку контактів цієї особи. Ефективність атаки радикально зростає завдяки високому рівню первинної довіри в середині соціального графа жертва вважає, що спілкується з колегою, другом чи родичем.

Професійна соціальна інженерія на прикладі LinkedIn- зловмисники створюють переконливі профілі рекрутерів відомих міжнародних компаній або технічних експертів. Протягом кількох днів чи тижнів вони ведуть легітимне спілкування з потенційною жертвою часто розробником або системним адміністратором критичної інфраструктури, після чого надсилають “тестове завдання” або “специфікацію вакансії” у вигляді архівованого файлу,

що містить шкідливий код для отримання первинного доступу до корпоративної мережі.

Енглер-фішинг - метод, за якого зловмисники створюють фейкові акаунти служб підтримки відомих брендів, банків або криптобірж у соціальних мережах. Вони здійснюють автоматизований моніторинг публічного простору. Коли реальний клієнт залишає скаргу або запитання на офіційній сторінці банку, фейкова “підтримка” миттєво пише користувачу в приватні повідомлення, пропонуючи вирішити проблему через перехід на фішинговий портал верифікації аккаунта.

1.2.4 SMS-фішинг та мобільний фішинг

SMS-фішинг Смішинг, Smishing - поєднання слів SMS та Phishing є вектором атаки, націленим безпосередньо на користувачів мобільних пристроїв. Актуальність цього методу зумовлена двома факторами- по-перше, користувачі демонструють значно вищий рівень психологічної довіри до SMS-повідомлень, ніж до електронної пошти; по-друге, мобільні операційні системи: Android, iOS через обмеженість розміру екрана інтерфейсу ускладнюють для користувача швидку верифікацію повного рядка URL-адреси в браузері.

Технічні та організаційні особливості смішингу:

Альфанумеричний супуфінг - мобільні мережі дозволяють відправникам використовувати текстові імена замість номерів телефонів наприклад, NovaPoshta, UkrSibbank, DIIA. Зловмисники через нелегальні або компрометовані SMS-шлюзи надсилають повідомлення з ідентичними альфанумеричними іменами. В результаті операційна система смартфона автоматично групує шкідливе SMS в один діалоговий потік тред з попередніми справжніми повідомленнями від банку чи поштової служби, що повністю нівелює пильність жертви.

Сценарії експлуатації контексту - смішинг майже завжди базується на тригерах високої терміновості чи стресу. Поширеними сценаріями є-

Повідомлення про необхідність сплатити мито за посилку, яка прибула на склад експлуатація брендів Нова Пошта, Укрпошта.

Критичні сповіщення про блокування банківських рахунків або підозрілі транзакції.

Пропозиції отримання соціальних виплат чи матеріальної допомоги від державних органів наприклад фейкові компенсації у рамках програм єПідтримка.

Інтеграція з QR-кодами - Модифікація мобільного фішингу, коли посилання зашифроване в QR-коді, який надсилається в месенджерах, поштою або навіть розміщується фізично в публічних місцях наприклад, на паркоматах чи заправках. Сканування коду камерою смартфона автоматично перенаправляє користувача на фішинговий платіжний шлюз, при цьому мобільні антивіруси та вбудовані системи фільтрації трафіку мають обмежені можливості для сканування та розпізнавання шкідливого контенту всередині графічного об'єкта.

1.3 Методи захисту від фішингових атак

Забезпечення стійкого захисту інформаційних систем від фішингових атак вимагає реалізації концепції глибокої ешелонованої. Оскільки фішинг одночасно експлуатує як технічні вразливості мережевих протоколів, так і психологічні слабкості людини, побудова захисного периметра не може обмежуватися лише одним інструментом.

Сучасні підходи до протидії фішингу поєднують у собі реактивні сигнатурні та проактивні інтелектуальні методи, які розгортаються на різних рівнях інформаційної інфраструктури: від поштових шлюзів та мережевих екранів до кінцевих робочих станцій користувачів. Для систематизації засобів протидії доцільно детально розглянути чотири базові методи захисту: фільтрацію електронної пошти, чорні та білі списки, аналіз URL-адрес і поведінковий аналіз.

1.3.1 Фільтрація електронної пошти

Оскільки електронна пошта залишається первинним і найбільш масовим вектором доставки фішингового контенту, першою лінією детекції є використання спеціалізованих шлюзів безпеки пошти (Secure Email Gateways, SEG. Автоматизована фільтрація вхідного потового потоку здійснюється на кількох взаємопов'язаних рівнях:

Автентифікація джерела - перевірка відповідності за допомогою тріади криптографічних та мережевих стандартів:

Sender Policy Framework - валідація IP-адреси сервера-відправника на відповідність дозволеному списку, опублікованому в DNS-записах домену.

DomainKeys Identified Mail - перевірка цифрового підпису листа за допомогою відкритого ключа, що гарантує цілісність контенту та неможливість його модифікації під час транспортування.

Domain-based Message Authentication, Reporting, and Conformance- Визначення політики обробки повідомлень none, quarantine, reject, які не пройшли валідацію SPF або DKIM, що мінімізує ризики прямого супуфінгу заголовків.

Статичний контентний аналіз - Сканування тексту повідомлення на наявність специфічних маркерів, характерних для фішингу: сигналів високої терміновості, закликів до оновлення паролів чи верифікації фінансових даних, а також аналіз структури MIME-заголовків.

Аналіз вкладених файлів- Перевірка файлів у поштових вкладеннях на наявність прихованих шкідливих скриптів JS, VBS, макросів у документах MS Office або подвійних розширень наприклад, document.pdf.exe. На рисунок 1.2 зображено рівні захисту від фішингу

1.3.2 Чорні та білі списки

Використання списків довіри та компрометації є класичним сигнатурним підходом, який виступає базовим компонентом більшості засобів мережевого захисту браузерних розширень, антивірусів, проксі-серверів [4].

Чорні списки - Це централізовані або локальні бази даних, що містять відомі індикатори компрометації, IoC такі як шкідливі IP-адреси, фішингові домени, скомпрометовані URL-адреси чи хеш-суми вірусних файлів. Провідними глобальними постачальниками таких репутаційних баз є сервіси Google Safe Browsing, Cisco Talos, Spamhaus та VirusTotal [7] [9]. Перевагою чорних списків є миттєва швидкість спрацювання та мінімальне використання обчислювальних ресурсів під час фільтрації відомих загроз.

Білі списки - метод превентивної довіри, за якого доступ дозволяється виключно до заздалегідь перевірених, верифікованих та легітимних ресурсів наприклад, внутрішніх корпоративних доменів або офіційних веб-порталів партнерів, тоді як будь-які інші зовнішні запити блокуються за замовчуванням. Це дозволяє радикально знизити рівень хибнопозитивних спрацювань FP у критичних інфраструктурах.

Системне обмеження: Головним недоліком цього підходу є його реактивна природа. Списки є безсилими проти атак нульового дня. Фішингова інфраструктура часто розгортається на доменах-одноденках, які виконують свою шкідливу функцію та видаляються зловмисниками ще до того, як інформація про них буде занесена до репутаційних баз безпеки [8].

1.3.3 Аналіз URL-адрес

Зважаючи на обмеженість статичних списків, сучасні системи безпеки здійснюють динамічний аналіз посилань у режимі реального часу. Цей метод передбачає дослідження структури та властивостей URL-адреси за кількома векторами:

Лексичний та структурний аналіз- Оцінка текстового рядка посилання на наявність аномалій, що вказують на маніпуляції з доменним ім'ям:

Тайпосквотинг та комбосквотинг - виявлення навмисних друкарських помилок або додавання слів-маркерів безпеки наприклад, support, secure, login до відомих брендів.

Гомографічні атаки- Перевірка наявності символів з інших алфавітів (наприклад, кирилиці), які візуально копіюють латинські літери в Unicode (IDN-атаки).

Аналіз метрик- Оцінка загальної довжини URL, кількості піддоменів, крапок, дефісів та наявності символу @ або прямих IP-адрес замість текстового домену.

Аналіз інфраструктури хостингу- запит до служб WHOIS та DNS для перевірки віку домену (щойно зареєстровані домени мають найвищий рівень ризику), репутації хостинг-провайдера та географічного розташування сервера (GeoIP).

Деобфускація та відстеження редіректів: Зловмисники часто маскують фішингові URL за допомогою сервісів скорочення посилань наприклад, bit.ly або багаторівневих ланцюжків перенаправлень HTTP Redirects, JavaScript redirects. Система безпеки здійснює автоматичне трасування всього ланцюжка до фінальної веб-сторінки призначення, щоб оцінити її реальний вміст.

1.3.4 Поведінковий аналіз

Поведінковий аналіз є найбільш проактивним методом захисту, оскільки він орієнтується не на статичні ідентифікатори загрози, а на дослідження безпосередніх дій, логіки та патернів функціонування об'єкта під час його взаємодії з користувачем чи системою.

Евристичний аналіз веб-контенту - замість простого пошуку текстових збігів, евристичні модулі аналізують поведінку коду самої веб-сторінки. Сюди належить виявлення спроб приховати вихідний код обфускація JS, використання технологій клакінгу коли сайт показує різний контент звичайному користувачу та боту антивірусної компанії, або наявність форм введення паролів, які відправляють дані на сторонні, не пов'язані з доменом сервери.

Технологія ізольованого моделювання - коли користувач намагається взаємодіяти Оцінка візуальної схожості - просунуті поведінкові системи використовують алгоритми комп'ютерного зору для порівняння знімка екрана аналізованого сайту з графічними шаблонами відомих брендів. Якщо сторінка візуально повністю ідентична інтерфейсу відомого банку, але розгорнута на сторонньому домені, поведінковий аналізатор миттєво класифікує її як фішинг. Традиційні методи захисту мають суттєві обмеження у швидкості адаптації до нових загроз. Сучасні системи класифікації фішингу використовують алгоритми машинного навчання. Вони здатні самостійно знаходити приховані закономірності у великих масивах даних. Цей підхід дозволяє з підозрілим вкладенням або посиланням, система безпеки в автоматичному режимі відкриває його всередині ізольованого віртуального середовища. Система емулює поведінку реального користувача рухи мишкою, кліки, введення даних і фіксує, чи не намагається об'єкт виконати несанкціоновані операції приховане завантаження файлів, модифікацію системного реєстру ОС чи ініціювання запитів до командних серверів С2.

автоматично виявляти невідомі раніше загрози та атаки нульового дня.

Для вирішення специфічних завдань кібербезпеки застосовують різні архітектури штучних нейронних мереж:

MLP: Ці базові нейронні мережі обробляють структуровані набори даних. Вони швидко класифікують підозрілі запити на основі числових характеристик мережевого трафіку та параметрів URL.

CNN: Моделі оптимізовані для роботи з просторовими даними та графікою. Системи захисту використовують їх для аналізу візуальної схожості сторінок. Це допомагає точно виявляти підроблені копії інтерфейсів банків чи корпоративних поштових служб.

Рекурентні мережі): Ці архітектури ефективно працюють із послідовностями даних та мають вбудований механізм пам'яті. Вони безперервно аналізують системні логи для виявлення складних

цілеспрямованих атак та відхилень у типовій поведінці співробітників компанії.

Архітектури Трансформерів: Такі моделі глибоко аналізують контекст у великих обсягах тексту. Системи безпеки використовують їх для семантичного аналізу вхідних електронних листів. Вони здатні розпізнавати приховані маніпуляції та специфічні текстові патерни соціальної інженерії.

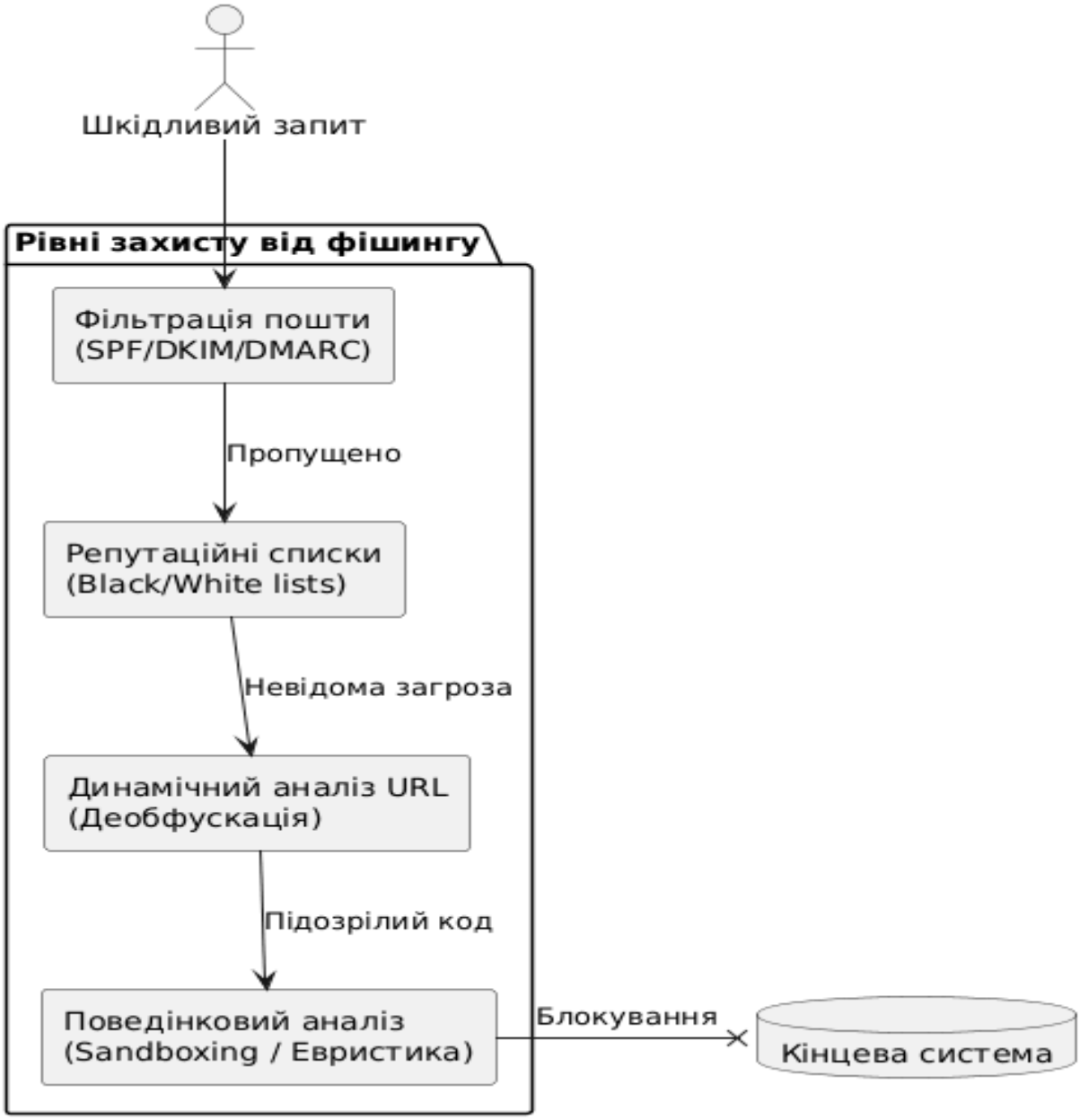


Рисунок 1.2 Ешелонована оборона

На рисунку 1.2 зображено блок-схему ешелонованої оборони, яка демонструє багаторівневий підхід до захисту кінцевої системи від фішингових атак. Схема показує шлях, який проходить вхідний шкідливий запит через послідовні фільтри безпеки. Система захисту складається з чотирьох основних рівнів, які працюють за принципом воронки: Фільтрація пошти **:** Перший рубіж оборони, на якому відбувається базова перевірка автентичності відправника за допомогою поштових протоколів. Якщо лист успішно проходить ці перевірки або обходить їх відмітка пропущено, він потрапляє на наступний етап. Репутаційні списки На цьому рівні запит звіряється з відомими базами шкідливих чорні списки або довірених білі списки відправників та доменів. Якщо система класифікує об'єкт як «Невідому загрозу», аналіз поглиблюється. Динамічний аналіз URL**:** Третій рівень призначений для виявлення прихованих загроз. Він включає перевірку посилань та розшифрування деобфускацію замаскованого коду. Якщо виявляється підозрілий код, він передається на найвищий рівень аналізу. Поведінковий аналіз: Останній рубіж оборони, де підозрілі файли або посилання запускаються в безпечному ізольованому середовищі. Евристичні алгоритми аналізують поведінку об'єкта під час виконання.

1.4 Використання машинного навчання для виявлення кіберзагроз

Традиційні підходи до захисту інформації, які покладаються на статичні чорні списки та евристичні правила, демонструють системні обмеження у боротьбі з сучасними кіберзагрозами. Головна проблема полягає у тому, що нові фішингові URL-адреси та підроблені домени з'являються швидше, ніж фахівці з безпеки встигають їх проаналізувати та внести до відповідних баз. У зв’язку з цим виникла гостра необхідність розробки інтелектуальних систем автоматизованого аналізу на основі машинного навчання, які здатні не лише блокувати відомі загрози, але й виявляти нові аномалії в режимі реального часу, узагальнюючи дані на раніше невідомі приклади.

У задачах класичного машинного навчання критично важливим етапом є конструювання та відбір інформативних ознак, оскільки алгоритм не може ефективно навчатися без якісних сигналів. Для виявлення фішингових ресурсів та шкідливих посилань найчастіше використовують такі групи ознак:

Лексичні ознаки URL - базуються на аналізі текстового рядка посилання. Сюди відносять загальну довжину URL, кількість дефісів, крапок та спеціальних символів, наявність IP-адреси безпосередньо у тексті URL замість домену, а також присутність підозрілих токенів і шаблонів наприклад; login, verify, update.

Ознаки домену та хоста мережеві метадані - Охоплюють інформацію з DNS та WHOIS вік домену, записи NS, час життя TTL, належність до певної автономної системи (ASN), географічне розташування (GeoIP), репутацію IP-адреси, а також наявність і специфічні властивості TLS/SSL-сертифіката.

Контентні ознаки веб-сторінки- Передбачають аналіз структури HTML/DOM - дерева, наявність прихованих форм введення даних, підключення зовнішніх ресурсів (нетипових скриптів чи зображень), підозрілі фрагменти JavaScript коду та оцінку загальної візуальної схожості сторінки з легітимними брендами.

Текстові та семантичні ознаки- Застосовуються переважно для аналізу каналів доставки, таких як електронні листи чи повідомлення у месенджерах. До них належать стилометрія тексту, наявність сигналів терміновості апеляція до страху або швидкої вигоди, нетипові інструкції та структурні аномалії в полях заголовків пошти.

Роль класичних ML-алгоритмів

Класичні алгоритми машинного навчання зазвичай виступають базовим рівнем при побудові систем захисту. До найбільш поширених алгоритмів у цій сфері належать: логістична регресія, дерева рішень, випадковий ліс, метод опорних векторів SVM та градієнтний бустинг.

Ключовою перевагою класичних алгоритмів особливо деревних моделей є їхня висока інтерпретованість та зрозумілість для аналітика безпеки.

Крім того, вони відзначаються відносно низькою обчислювальною вартістю навчання та забезпечують швидке генерування висновків інференсування на невеликій кількості заздалегідь підготовлених ознак.

Водночас ці методи мають суттєві недоліки. У реальних сценаріях кіберзахисту ознаки URL-адрес та веб-контенту можуть бути високовимірними й динамічно змінюватися. Ручне конструювання та оновлення таких ознак погано масштабується, і моделі починають стрімко деградувати, коли зловмисники модифікують патерни обфускації. Ці обмеження, пов'язані з неможливістю класичного ML ефективно працювати з глибокими неструктурованими послідовностями, підштовхнули дослідників до застосування більш потужного підходу - нейромережевого глибинного навчання, на рисунку 1.3 показано застосування нейронних мереж у кібербезпеці

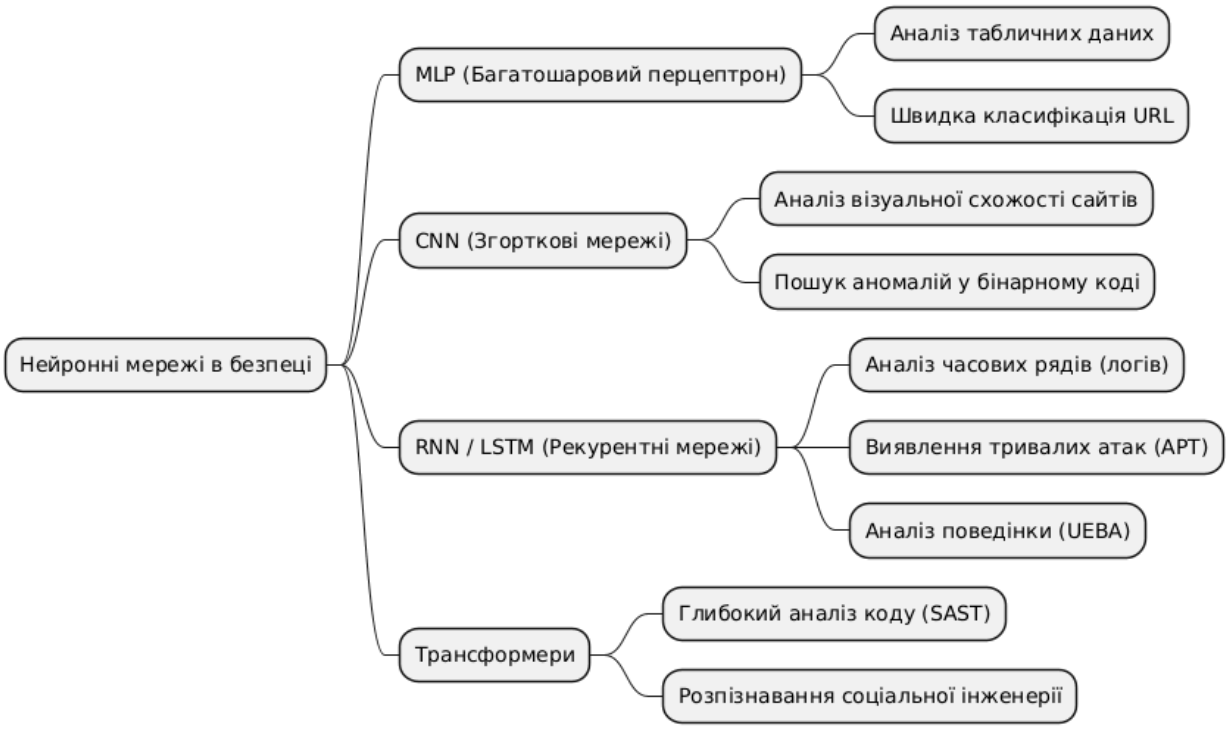


Рисунок 1.3 Застосування нейронних мереж у кібербезпеці

На рисунку 1.3 наведено структурну схему застосування різних архітектур нейронних мереж у сфері кібербезпеки. Схема виділяє чотири ключові класи моделей машинного навчання та закріплені за ними завдання:
MLP: Базові нейронні мережі прямого поширення. Вони обробляють структуровані табличні дані, наприклад, набори числових характеристик мережевого трафіку, і виконують швидку класифікацію підозрілих URL-адрес.
CNN: Моделі, оптимізовані для обробки просторових даних та зображень. Їх використовують для аналізу візуальної схожості вебсайтів, що допомагає виявляти точні фішингові копії сторінок банків чи поштових сервісів. Також CNN застосовують для пошуку структурних аномалій безпосередньо у бінарному коді файлів.
RNN / LSTM (Рекурентні мережі): Архітектури, розроблені для роботи з послідовностями даних та пам'яттю. У кібербезпеці вони безперервно аналізують часові ряди, такі як системні логи чи події безпеки. Це дозволяє системі виявляти складні цілеспрямовані атаки (APT), що тривають тижнями, та виконувати поведінковий аналіз (UEBA) для фіксації відхилень від типових дій співробітників.
Трансформери: Сучасні моделі, які добре аналізують контекст великих обсягів тексту. Їх застосовують для глибокого статичного аналізу коду (SAST) з метою пошуку вразливостей, а також для розпізнавання прихованих маніпуляцій та соціальної інженерії у текстах електронних листів.

1.5 Нейронні мережі та їх застосування у кібербезпеці

Стрімка еволюція кіберзагроз, поява поліморфного шкідливого програмного забезпечення та автоматизація дій зловмисників призвели до того, що традиційні сигнатурні та евристичні методи захисту інформації втратили свою колишню ефективність. Сигнатурний аналіз здатний виявляти лише заздалегідь відомі індикатори компрометації IoC, що робить системи безпеки беззахисними перед атаками нульового дня. У цьому контексті логічним етапом розвитку систем кібербезпеки стало впровадження

технологій штучного інтелекту, зокрема штучних нейронних мереж ANN та глибокого навчання

Штучна нейронна мережа - це обчислювальна структура, натхненна біологічними нейронними мережами людського мозку, яка складається з великої кількості паралельно функціонуючих елементів нейронів, поєднаних між собою зв’язками синапсами з певними ваговими коефіцієнтами. Головна перевага ANN у задачах кібербезпеки полягає в їхній здатності до узагальнення: після навчання на репрезентативній вибірці даних мережа здатна виявляти приховані закономірності, нелінійні залежності та патерни зловмисної поведінки у нових, раніше невідомих їй вхідних потоках інформації.

Основні архітектури нейронних мереж у сфері захисту інформації

Залежно від специфіки вхідних даних мережеві пакети, лог-файли, бінарний код або текстові URL-адреси у кібербезпеці застосовуються різні архітектурні класи ANN:

Багатошаровий перцептрон MLP. Це класична архітектура прямого поширення Feedforward, що складається з вхідного, одного або кількох прихованих та вихідного шарів нейронів, де кожен вузол попереднього рівня зв'язаний з усіма вузлами наступного. MLP демонструє високу ефективність під час обробки структурованих табличних даних, де заздалегідь виділено вектор числових або категоріальних ознак. Саме цей тип мереж часто використовується для класифікації URL-адрес та аналізу метаданих мережевих з'єднань завдяки своїй обчислювальній простоті та швидкості роботи.

Згорткові нейронні мережі CNN. Спочатку розроблені для комп'ютерного зору, CNN знайшли нетривіальне застосування в кіберзахисті. Наприклад, бінарний код виконуваних файлів можна візуалізувати як матрицю пікселів (градації сірого). Згорткові шари автоматично виділяють просторові ознаки та структурні аномалії в коді, що дозволяє класифікувати сімейства шкідливого ПЗ без його декомпіляції чи виконання в пісочниці. Також CNN

застосовують для аналізу сигнатур трафіку, представленого у вигляді дампів пакетів.

Рекурентні нейронні мережі RNN та мережі довгої короткострокової пам'яті LSTM. Даний клас мереж розроблений для обробки послідовностей даних, де поточний вихід залежить не лише від поточного входу, а й від попередніх станів системи. Це ідеальний інструмент для аналізу часових рядів та журналів подій. LSTM-мережі здатні аналізувати послідовності системних викликів , виявляти аномалії в поведінці користувачівта розпізнавати складні тривалі атаки, які розгорнуті в часі.

Трансформери Сучасні архітектури на основі механізму самоуваги, такі як BERT або GPT-подібні моделі, дедалі частіше використовуються для глибинного аналізу вихідного коду програм на наявність уразливостей а також для розпізнавання складного семантичного контексту у фішингових електронних листах або повідомленнях у соціальних мережах.

Ключові сфери застосування ANN у кібербезпеці

Практична інтеграція нейромережевих моделей у контури безпеки сучасних підприємств здійснюється за кількома критично важливими напрямами:

Виявлення та класифікація шкідливого програмного забезпечення. Замість пошуку статичних хеш-сум антивірусні системи нового покоління (Next-Generation Antivirus, NGAV) та рішення класу EDR (Endpoint Detection and Response) використовують ANN для поведінкового аналізу процесів у реальному часі. Мережа оцінює послідовність дій програми, звернення до реєстру, мережеву активність та виносить вердикт щодо шкідливості софту.

Системи виявлення та запобігання вторгненням (IDS/IPS). Нейронні мережі аналізують колосальні обсяги мережевого трафіку на рівні окремих пакетів або мережевих потоків; NetFlow/IPFIX. Вони дозволяють автоматично диференціювати легітимні запити користувачів від сканування портів, DDoS-атак, спроб ін'єкцій ж; SQLi, XSS або експлуатації системних уразливостей.

Аналіз поведінки користувачів та сутностей UEBA -. ANN будують профіль “нормальної” життєдіяльності корпоративної мережі чи конкретного співробітника типові години роботи, обсяги завантажуваних даних, геопозиція, використовувані пристрої. Будь-яке різке відхилення від цього базового профілю наприклад, раптове вивантаження великої бази даних вночі маркується як аномалія, що дозволяє оперативно локалізувати внутрішніх порушників або скомпрометовані облікові записи.

Протидія фішингу та соціальній інженерії. Нейромережі виконують комплексний аналіз веб-ресурсів та вхідних комунікацій. Вони здатні одночасно оцінювати лексичну структуру URL-адреси, семантичний вміст веб-сторінки, наявність прихованих редіректів, а також лінгвістичні ознаки маніпуляцій у тексті наприклад, штучне створення почуття терміновості чи страху в електронному листі.

Переваги та обмеження використання нейронних мереж. Впровадження штучних нейронних мереж забезпечує низку незаперечних переваг, серед яких: висока адаптивність до нових типів загроз, можливість автоматизації рутинної роботи аналітиків центрів моніторингу безпеки SOC, зниження рівня хибнопозитивних спрацювань FP за умови правильного налаштування, та здатність працювати з великими потоками високорозмірних неструктурованих даних.

Водночас використання ANN у кібербезпеці пов'язане із серйозними викликами та обмеженнями, які необхідно враховувати під час проектування систем захисту:

Проблема чорної скриньк. Нейронні мережі є математично складними нелінійними моделями, тому аналітику безпеки часто буває важко зрозуміти інтерпретацію логіки, за якою мережа прийняла рішення про блокування того чи іншого процесу чи сайту. Це обмежує застосування ANN у критичних інфраструктурах, де кожне рішення системи має бути юридично та технічно обґрунтованим.

Вразливість до змагальних атак. Зловмисники навчилися застосовувати методи штучного інтелекту проти самих систем захисту. Шляхом внесення мікроскопічних, непомітних для людини змін у вхідні дані наприклад, додавання кількох невинних рядків коду у шкідливий файл або специфічних символів у URL вони можуть змусити нейромережу помилитися та класифікувати небезпечний об'єкт як легітимний.

Потреба у якісних та актуальних даних. Для ефективного навчання глибинних моделей потрібні колосальні обсяги розмічених даних, які відображають як актуальні атаки, так і нормальний трафік. Оскільки ландшафт кіберзагроз постійно змінюється, моделі ANN потребують безперервного донавчання, інакше їхня точність стрімко деградує з часом ефект концептуального зсуву - Concept Drift [4, 8, 9].

1.6 Основні параметри ефективності систем виявлення атак

Оцінювання ефективності систем виявлення атак та інтелектуальних антифішингових рішень є одним із найбільш відповідальних етапів їх проектування. Оскільки жоден алгоритм штучного інтелекту не здатний забезпечити абсолютно безпомилкове розпізнавання, виникає потреба у використанні математично обґрунтованої системи метрик. Ці метрики дозволяють кількісно оцінити якість роботи моделей, порівняти різні архітектури нейронних мереж між собою та знайти оптимальний баланс між рівнем безпеки та доступністю інформаційних сервісів.

Математичний апарат оцінювання систем кіберзахисту базується на концепції матриці помилок. Для задачі бінарної класифікації, де позитивним класом (1) є наявність загрози (фішинговий URL, шкідливий пакет), а негативним (0) - легітимний трафік, виділяють чотири базові типи результатів тестування:

TP - істинно позитивні рішення атака дійсно відбулася, і система безпеки її успішно виявила та заблокувала.

TN - істинно негативні рішення легітимний запит користувача або безпечний веб-сайт було правильно розпізнано як безпечний.

FP - хибно позитивні рішення (помилка I-го роду) безпечна подія чи нормальний URL помилково класифіковані системою як шкідливі. У практичній діяльності це призводить до безпідставного блокування користувачів та генерування помилкових сповіщень.

FN - хибно негативні рішення (помилка II-го роду)- реальна кібератака або фішингове посилання не були розпізнані системою й безперешкодно проникли в інформаційну структуру. Це найбільш критичний і небезпечний тип помилки.

На основі цих чотирьох базових параметрів розраховуються ключові критерії ефективності сучасних систем виявлення атак.

1.6.1 Точність

Параметр загальної точності визначає частку правильних рішень як позитивних, так і негативних, які модель прийняла відносно загальної кількості всіх тестованих об'єктів. Математична формула для обчислення має вказана в 1.1

$$\text{Accuracy} = \frac{(\text{TP} + \text{TN})}{(\text{TP} + \text{TN} + \text{FP} + \text{FN})} \qquad (1.1)$$

Особливості застосування в кібербезпеці - попри свою інтуїтивну зрозумілість, метрика Accuracy має суттєве обмеження, пов'язане з проблемою дисбалансу класів. У реальних комп'ютерних мережах обсяг легітимного трафіку чи безпечних URL-адрес у тисячі разів перевищує кількість зловмисних дій [10].

Якщо набір даних складається на 99% з безпечних посилань і лише на 1% з фішингових, то примітивна модель, яка буде абсолютно всі об'єкти маркувати як безпечні, покаже надзвичайно високу точність - 99\%. Проте практична цінність такої системи дорівнюватиме нулю, оскільки вона пропустить всі 100\% кібератак. Тому для об'єктивного аналізу архітектур ANN використовуються більш специфічні метрики.

1.6.2 Точність класифікації

Метрика точності класифікації або прогностична цінність позитивного результату відображає частку справжніх атак серед усіх подій, які система безпеки визначила як загрозу. Формула для розрахунку:

$$\text{Точність} = \frac{\text{TP}}{(\text{TP} + \text{FP})} \quad (1.2)$$

Показник Точність демонструє спроможність моделі працювати без помилкових тривог. Високе значення цього параметра гарантує, що система не буде блокувати роботу легітимних користувачів та веб-ресурсів.

Якщо для антифішингового шлюзу точність класифікації є низькою, це означає, що користувачі постійно стикатимуться з хибним блокуванням безпечних робочих сайтів. Це не лише знижує продуктивність праці в організації, а й викликає у персоналу служб моніторингу безпеки SOC ефект втоми від сповіщень через що адміністратори починають ігнорувати сигнали тривоги.

1.6.3 параметр чутливість

Повнота або чутливість системи визначає, яку частку реальних кібератак чи фішингових ресурсів із їхнього загального масиву модель змогла успішно виявити. Математично метрика виражається формулою:

$$\text{Recall} = \frac{\text{TP}}{(\text{TP} + \text{FN})} \quad (1.3)$$

Особливості застосування в кібербезпеці:

У сфері захисту інформації повнота часто є пріоритетнішою метрикою, ніж точність класифікації. Вартість пропущеної фішингової атаки FN, яка може призвести до компрометації облікових даних адміністратора, витоку конфіденційних баз даних чи зараження мережі програмами-вимагачами, є неспіввимірно вищою, ніж незручності від тимчасового хибного блокування

FP. Низький показник Recall свідчить про те, що система безпеки є “сліпою” до значної частини поточних загроз.

1.6.4 F1-міра

Оскільки параметри точність класифікації та повнота перебувають у зворотно-пропорційній залежності спроба розробника максимізувати повноту шляхом зниження порогу чутливості нейромережі неминуче веде до зростання кількості хибних спрацювань і зниження точності, і навпаки, виникає потреба в єдиному збалансованому критерії.

F1-міра - це гармонійне середнє значення між точністю класифікації та повнотою, яке розраховується за формулою:

$$\mathrm{F1} \;=\; 2 * \mathrm{cdot}\frac{(\text{точність класифікації } \backslash\mathrm{cdot}\ \text{повнота})}{(\text{точність класифікації} + \text{повнота})} \quad (1.4)$$

Особливості застосування в кібербезпеці:

F1-міра є найбільш об'єктивним інтегральним показником під час оцінювання нейромережевих моделей в умовах нерівномірного розподілу класів у датасетах. Вона досягає свого максимуму (1) лише тоді, коли і повнота виявлення атак, і точність класифікації одночасно прямують до високих значень. Якщо хоча б один із цих параметрів стрімко деградує, значення F1-міри також різко падає, що дозволяє розробнику вчасно виявити дисбаланс у налаштуваннях алгоритму.

Під час практичного розгортання штучних нейронних мереж вибір фінального порогу спрацювання завжди є компромісом між розглянутими метриками. Залежно від призначення об'єкта захисту наприклад, банківський сервер критичної інфраструктури чи публічна гостьова мережа Wi-Fi конфігурація системи може гнучко зміщуватися в бік максимальної повноти або максимальної точності, а розрахунок повного комплексу цих чотирьох параметрів забезпечує всебічний контроль якості функціонування розроблених засобів захисту [6].

РОЗДІЛ 2 АНАЛІЗ ІНСТРУМЕНТІВ ТА МЕТОДІВ ВИЯВЛЕННЯ ФІШИНГОВИХ АТАК

2.1 Аналіз сучасних систем виявлення фішингових атак

Зі стрімким розвитком технологій соціальної інженерії та засобів автоматизації зловмисників, фішингові атаки стали значно складнішими, цілеспрямованішими та важчими для виявлення. Сучасні системи протидії фішингу еволюціонували від простих реактивних інструментів до складних проактивних комплексів, що поєднують різні методи аналізу.

Усі існуючі системи та підходи до виявлення фішингу можна класифікувати за їхньою базовою архітектурою та методами обробки даних на кілька основних категорій.

2.1.1 Системи на основі чорних списків та сигнатур

Це найдавніший та найбільш поширений підхід, який досі використовується як перший рубіж захисту наприклад, Google Safe Browsing.

Принцип роботи - URL-адреса, IP-адреса або домен, що запитується користувачем, порівнюється з попередньо складеною базою даних відомих шкідливих ресурсів. Сигнатурний аналіз також передбачає пошук відомих фрагментів шкідливого коду в тілі електронного листа чи веб-сторінки.

Переваги- Висока швидкість обробки, мінімальне навантаження на обчислювальні ресурси та майже нульовий рівень хибнопозитивних спрацьовувань FP для відомих загроз.

Недоліки - абсолютна неефективність проти атак нульового дня. Зловмисники обходять ці системи, генеруючи тисячі нових доменів щодня за допомогою алгоритмів DGA або використовуючи скомпрометовані легітимні сайти.

2.1.2 Системи евристичного аналізу

Евристичні системи створені для подолання обмежень чорних списків шляхом оцінки підозрілості ресурсу за набором заздалегідь визначених правил.

Принцип роботи - система аналізує специфічні ознаки об'єкта. Для URL-адрес це може бути наявність IP-адреси замість доменного імені, надмірна довжина посилання, використання спеціальних символів наприклад, @ або -, невідповідність домену в адресному рядку та в тексті листа. Для веб-сторінок аналізується наявність прихованих фреймів, підозрілих скриптів або форм для введення паролів без SSL-сертифіката.

Переваги Здатність виявляти нові, раніше невідомі фішингові ресурси.

Недоліки Високий ризик хибнопозитивних спрацьовувань та необхідність постійного ручного оновлення правил експертами з кібербезпеки для відповідності новим технікам зловмисників.

2.1.3 Систем на основі машинного та глибокого навчання

Сьогодні це найбільш динамічний і перспективний напрямок в індустрії кібербезпеки. Методи штучного інтелекту дозволяють автоматизувати процес вилучення ознак і знаходити приховані закономірності у великих масивах даних.

Класичне машинне навчання - Використовуються алгоритми SVM та XGBoost. Вони навчаються на сотнях характеристик (довжина URL, лексичний аналіз, реєстраційні дані WHOIS, частотність певних слів) і класифікують ресурс як легітимний або фішинговий.

Глибоке навчання та NLP - Для аналізу тексту електронних листів або контенту сайтів активно застосовуються методи обробки природної мови. Нейронні мережі Transformer-моделі здатні розуміти контекст повідомлення, виявляючи маніпуляції, терміновість або емоційний тиск, що є класичними тригерами соціальної інженерії.

2.1.4 Системи аналізу візуальної схожості

Оскільки фішингові сайти створюються з метою обдурити користувача, вони візуально імітують дизайн оригінальних платформ банків, соціальних мереж, поштових сервісів.

Принцип роботи: Такі системи створюють знімки екрана підозрілої веб-сторінки та за допомогою методів комп'ютерного зору наприклад, згорткових нейронних мереж - CNN порівнюють їх з еталонними зображеннями легітимних сайтів. Також аналізується структура DOM-дерева, CSS-стилі, розташування логотипів та форм авторизації.

Переваги: Висока ефективність проти складних атак, де зловмисники обфускують HTML-код сторінки або використовують зображення замість тексту, щоб уникнути лексичного аналізу.

Недоліки: Значні витрати обчислювальних ресурсів і часу на рендеринг сторінок, що ускладнює їх використання в режимі реального часу (Real-time detection).

2.1.5 Комплексні гібридні системи виявлення

Більшість сучасних комерційних рішень наприклад, продукти від Proofpoint, Cisco Umbrella, Microsoft Defender є гібридними. Вони використовують багатошаровий підхід:

1. Перший рівень: Швидкий фільтр за допомогою чорних списків та перевірки репутації (IP/Домен/Відправник).
2. Другий рівень: Евристичний та лексичний аналіз вмісту (швидкийML-класифікатор).
3. Третій рівень динамічний: Якщо ресурс є підозрілим, але не класифікованим однозначно, він відкривається в ізольованому середовищі (Sandbox) для аналізу поведінки коду та візуальної структури

Логіку взаємодії компонентів та послідовність обробки запитів у межах такого багатошарового підходу наведено на рисунку 2.1

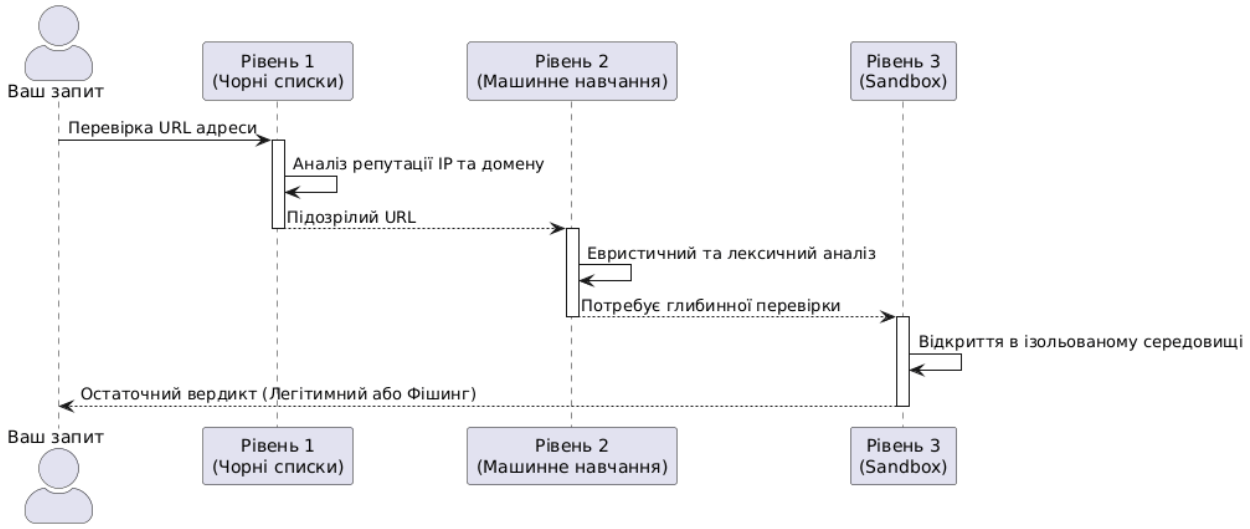


Рисунок 2.1 Багатошаровий підхід гібридних систем

Як показано на рис. 2.1, процес обробки користувацького запиту реалізовано у вигляді послідовного каскаду перевірок. На першому рівні відбувається експрес-аналіз репутації IP-адреси та домену за допомогою чорних списків. У разі виявлення неоднозначності або підозрілості URL-адреси, запит передається на другий рівень для проведення евристичного та лексичного аналізу методами машинного навчання. Якщо автоматизовані моделі визначають, що вебресурс потребує глибинної перевірки, об'єкт спрямовується на третій рівень - в ізольоване середовище Sandbox. Після виконання та динамічного аналізу поведінки сайту в пісочниці система формує остаточний вердикт легітимний або фішинговий та повертає його користувачу.

2.2 Інструменти аналізу шкідливих веб-ресурсів

Інструментарій для аналізу шкідливих веб-ресурсів можна розділити на три рівні взаємодії: інструменти для автоматизованого аудиту URL-адрес, корпоративні антифішингові сервіси та клієнтські механізми захисту, вбудовані у браузери.

2.2.1 Системи перевірки

Це інструменти, які спеціалізуються на швидкому або глибинному аналізі підозрілого посилання. Вони є першою ланкою для аналітиків з кібербезпеки.

Сервіси динамічного аналізу: Наприклад, URLScan.io або Any.Run. Вони не просто перевіряють репутацію домену, а відвідують сайт у віртуальному середовищі ізольованому контейнері, роблять знімок екрана, аналізують DOM-структуру сторінки, виклики скриптів та перенаправлення. Це дозволяє виявити прихований фішинг, який активується лише за певних умов наприклад, для користувачів з певним User-Agent.

Агрегатори репутаційних даних: VirusTotal залишається стандартом індустрії, що об'єднує результати десятків антивірусних вендорів та URL-фільтрів. Його API дозволяє автоматизувати перевірку великих обсягів посилань у режимі реального часу.

Спільнотні бази даних: PhishTank -це відкрита платформа, де дані про фішингові ресурси збираються спільними зусиллями спільноти. Це важливе джерело для навчання моделей машинного навчання [5].

2.2.2 Антифішингові сервіси

Це комплексні рішення для захисту організацій, які працюють на рівні мережевих шлюзів Email Security Gateways та кінцевих точок (Endpoint).

Time-of-Click Analysis - Сучасні сервіси наприклад, рішення від Proofpoint, Barracuda чи Microsoft Defender) не просто перевіряють посилання під час отримання листа. Вони перезаписують (кожне посилання у вхідному повідомленні. Коли користувач натискає на нього, сервіс перевіряє цільовий ресурс у реальному часі. Якщо сайт був легітимним вранці, але став фішинговим до обіду, система заблокує доступ.

Аналіз контексту та поведінки: Використання AI для виявлення аномалій у листах наприклад, нетиповий тон звернення, імітація стилю керівника - Business Email Compromise.

Інтеграція з EDR/XDR: Сучасні антифішингові платформи інтегруються з системами захисту робочих станцій, що дозволяє автоматично ізолювати хост, якщо користувач все ж таки перейшов за небезпечним посиланням.

2.2.3 Браузерні механізми захисту

Це останній, але найважливіший рубіж захисту безпосередньо на пристрої користувача.

API безпечного перегляду: Більшість сучасних браузерів Chrome, Firefox, Edge базуються на технології Google Safe Browsing. Браузер періодично завантажує оновлені списки відомих шкідливих та фішингових ресурсів. Перевірка відбувається локально, що забезпечує високу швидкість без відправки кожного URL на сервер- для збереження приватності.

Real-time Protection - нові версії браузерів переходять від статичних списків до динамічної перевірки. Коли користувач переходить за підозрілим посиланням, браузер може надіслати запит до хмарного сервісу вендора для перевірки ресурсу в реальному часі якщо він відсутній у локальному кеші.

Розширення для аналізу - Для професійних цілей аналітики часто використовують спеціалізовані розширення наприклад, Netcraft, які надають розширену інформацію про домен: вік домену, хостинг, країна реєстрації, що допомагає в ручному розслідуванні інцидентів.

2.3 Методи аналізу фішингових веб-сайтів

Для ефективного розрізнення фішингових ресурсів від легітимних необхідно застосовувати багатофакторний аналіз. Основні методи класифікації базуються на трьох “стовпах”: структурі URL, контентному аналізі HTML-коду та характеристиках доменного імені. У сучасних системах безпеки ці методи часто використовуються комбіновано у складі ML-моделей.

2.3.1 Аналіз структури URL

Це найбільш швидкий метод аналізу, оскільки він не потребує завантаження всієї сторінки. Аналіз структури URL дозволяє виявити ознаки зловмисної діяльності на етапі запиту.

Лексичні ознаки: Аналізується довжина URL фішингові посилання часто надмірно довгі для приховування реального домену, кількість спеціальних символів наприклад, @, -, %, &, а також наявність IP-адреси замість доменного імені.

Методи обфускації- Системи перевіряють використання Punycode імітація кириличних чи інших символів за допомогою латиниці, наприклад, xn--..., сервісів скорочення посилань (bit.ly, t.co) та методи тайпосквоттингу - реєстрації доменів, що візуально схожі на відомі бренди наприклад, goog1e.com замість google.com.

Синтаксичні патерни - виявлення підозрілих піддоменів наприклад, login-secure-verification.bank.com та невідповідності між шляхом до ресурсу та його видимим текстом гіперпосиланням.

2.3.2 Аналіз HTML-коду та контенту

Якщо URL-адреса виглядає потенційно підозрілою, аналіз переходить до вивчення вмісту веб-сторінки. Це “глибокий” рівень аналізу.

DOM-аналіз - Система сканує структуру сторінки на наявність прихованих фреймів (<iframe>), форм введення (<form>) з підозрілими атрибутами action (які ведуть на сторонні домени), а також невидимих або перекриваючих елементів, що використовуються для клікджекінгу.

Аналіз скриптів - Виявлення обфускованого JavaScript, який може використовуватися для викрадення даних з форм (Keylogging) ще до натискання кнопки “Відправити”.

Візуальна схожість - Порівняння рендерингу сторінки з еталоном. Використовуються методи комп'ютерного зору для визначення того, чи

копіює сайт зовнішній вигляд (логотипи, стиль, колірну гаму) відомого сервісу, навіть якщо код сайту суттєво відрізняється.

2.3.3 Аналіз доменних характеристик

Цей метод спирається на метадані про реєстрацію ресурсу. Часто фішингові сайти існують короткий проміжок часу, що стає ключовим індикатором.

Вік домену - Більшість фішингових сайтів створюються “на один день”. Домени, зареєстровані менше ніж 30-60 днів тому, автоматично отримують вищий бал ризику.

Інформація WHOIS - Перевірка даних власника домену, реєстратора та країни походження. Невідповідність між заявленою організацією (яка нібито володіє сайтом) та даними реєстранта є потужним індикатором фішингу.

Сертифікати SSL/TLS - Масова доступність сертифікатів Let's Encrypt призвела до того, що більшість фішингових сайтів зараз працюють через HTTPS замок в адресному рядку не є гарантією безпеки. Аналіз характеристик сертифіката (наприклад, чи виданий він на конкретну особу, чи є доменним) дозволяє відфільтрувати автоматично згенеровані сертифікати від корпоративних [16].

2.4 Застосування алгоритмів машинного навчання для виявлення фішингу

Застосування методів машинного навчання дозволило значно підвищити ефективність виявлення фішингу, переходячи від статичних чорних списків до адаптивних моделей, здатних розпізнавати нові, раніше невідомі загрози. Основна перевага машинного навчання полягає в автоматичному виявленні нелінійних залежностей між характеристиками веб-ресурсу, які людина або проста евристична система можуть не помітити.

2.4.1 Основні класи алгоритмів та їх застосування

Для розв'язання задачі класифікації легітимний vs. фішинговий сайт найчастіше застосовуються такі підходи:

Класичне машинне навчання

Випадкові ліси та градієнтний бустинг - Це “робочі конячки” індустрії. Вони працюють з табличними даними наприклад, довжина URL, кількість точок, вік домену. Вони дуже ефективні для швидкої класифікації, оскільки стійкі до перенавчання і добре обробляють великі набори ознак.

Метод опорних векторів: Часто використовується для розділення даних у багатовимірному просторі, де межа між фішинговими та безпечними сайтами є нелінійною.

Глибоке навчання:

Згорткові нейронні мережі CNN - Використовуються для аналізу візуального контенту .CNN здатні автоматично виділяти патерни дизайну, логотипів та компонування елементів, що дозволяє виявляти фішинг навіть тоді, коли HTML-код сильно обфускований або містить легітимні скрипти.

Рекурентні нейронні мережі та Трансформери: Незамінні для аналізу послідовностей, таких як URL-рядки або текст електронних листів. Вони аналізують контекст: чи є в повідомленні ознаки терміновості, емоційного маніпулювання чи типові для соціальної інженерії граматичні конструкції.

Навчання без учителя :

Кластеризація - Використовується для виявлення нових типів атак. Якщо система помічає групу раніше невідомих сайтів, що мають схожу поведінку наприклад, схожі шаблони створення доменів або однакові JS-скрипти, вона може автоматично позначати їх як підозрілі навіть без еталонної розмітки.

2.4.2 Процес побудови моделі

Ефективна ML-система виявлення фішингу проходить через кілька критичних етапів:

Збір даних - Накопичення великих наборів даних наприклад, PhishTank, OpenPhish та легітимних даних Alexa Top 1M.

Вилучення ознак - Перетворення “сирих” даних HTML, URL у вектори ознак (згідно з категоріями, описаними в п. 2.3).

Навчання - Вибір моделі, налаштування гіперпараметрів та мінімізація функції втрат.

Висновок - Робота моделі в режимі реального часу, де кожному запиту присвоюється ймовірність фішингу від 0 до 1.

2.4.3 Виклики та обмеження сучасних ML-систем

Незважаючи на високу ефективність, застосування ML має суттєві виклики:

Концептуальний дрейф - фішингові атаки еволюціонують швидше, ніж моделі. Якщо зловмисники змінюють тактику наприклад, переходять від масових розсилок до AI-генерованих персоналізованих листів, модель, навчена на старих даних, швидко втрачає точність. Це вимагає постійного перенавчання (Retraining).

Змагальне машинне навчання - Зловмисники активно використовують отруєння даних або додають до своїх сторінок шум, який невидимий для людини, але збиває з пантелику нейронні мережі наприклад, додавання невидимих символів або коду, який імітує легітимні бібліотеки.

Дисбаланс класів - У реальному світі легітимних сайтів на порядки більше, ніж фішингових. Моделі можуть мати схильність “вважати все безпечним”, що вимагає застосування спеціальних технік SMOTE, зважування класів, спеціальні метрики на кшталт F1-score замість Accura [10].

2.5 Порівняльний аналіз алгоритмів класифікації

Вибір алгоритму машинного навчання для виявлення фішингу залежить від компромісу між точністю), швидкістю роботи та можливістю інтерпретації

прийнятих рішень. Нижче наведено аналіз основних алгоритмів, що застосовуються в сучасних системах безпеки.

Аналіз ефективності алгоритмів

Дерев рішень: Особливості: Це інтуїтивно зрозумілі структури, що імітують логіку якщо-то.

Переваги: Висока швидкість прогнозування, легкість в інтерпретації можна чітко сказати, чому сайт визнано фішинговим.

Недоліки: Висока схильність до перенавчаннята чутливість до невеликих змін у даних. Вони погано справляються зі складними, нелінійними залежностями.

Випадковий ліс: Особливості: Ансамбль дерев рішень, що усереднює результати багатьох слабких класифікаторів.

Переваги: Значно стійкіший до перенавчання, ніж поодиноке дерево. Чудово працює з наборами даних, де присутні як числові, так і категоріальні ознаки. Є де-факто стандартом для аналізу табличних характеристик URL та доменів.

Недоліки: Потребує більше обчислювальних ресурсів для навчання та має меншу швидкість прогнозування порівняно з простими деревами.

SVM: Особливості: Знаходить оптимальну гіперплощину, що розділяє легітимні та фішингові дані [1].

Переваги: Дуже ефективний у просторах високої розмірності коли ознак дуже багато. Показує високу точність при чітко визначених межах класів.

Недоліки: Висока обчислювальна складність при великих обсягах даних масштабування до мільйонів записів є проблемним. Важко інтерпретувати результат у чорній скриньці моделі.

Нейронні мережі: Особливості: Складні багатошарові архітектури CNN, RNN,.

Переваги: Найвища точність для неструктурованих даних текст листа, HTML-код, скріншоти. Здатні виявляти приховані патерни, які неможливо описати вручну.

Недоліки: Потребують величезної кількості даних для навчання, тривалого часу на навчання та значних потужностей GPU. Вкрай низька інтерпретованість рішень проблема чорної скриньки.

Показана порівняльний аналіз характеристик алгоритмів класифікації Таблиця 2.1

Таблиця 2.1 - Порівняльний аналіз характеристик алгоритмів класифікації

Алгоритм	Точність (на табличних даних)	Швидкість прогнозування	Інтерпретованість	Стійкість до перенавчання
Дерев рішень	Середня	Дуже висока	Висока	Низька
Випадковий ліс	Висока	Висока	Середня	Висока
SVM	Висока	Середня	Низька	Середня
Нейронні мережі	Дуже висока	Середня/Низька	Дуже низька	Висока (при правильному налаштуванні)

Синтез результатів та вибір архітектури аналіз показує, що срібної кулі не існує:

1. Для систем швидкого реагування наприклад, розширення для браузера): Найкраще підходять Випадковий ліс або Дерев рішень, оскільки вони забезпечують миттєву відповідь при мінімальних витратах ресурсів пристрою.

2. Для аналізу неструктурованого контенту пошта, веб-сторінки: Оптимальним вибором є Глибокі нейронні мережі (CNN/RNN), оскільки тільки вони можуть адекватно обробити візуальний та контекстний шум.

3. Для корпоративних шлюзів багатошаровий захист: Найефективнішим є гібридний підхід -каскадна система, де швидкі алгоритми фільтрують 90% очевидних загроз, а складні нейронні мережі перевіряють лише підозрілі об'єкти, що залишилися. Графічне представлення процесу

функціонування такої гібридної системи захисту у реальному часі наведено на рисунку 2.2.

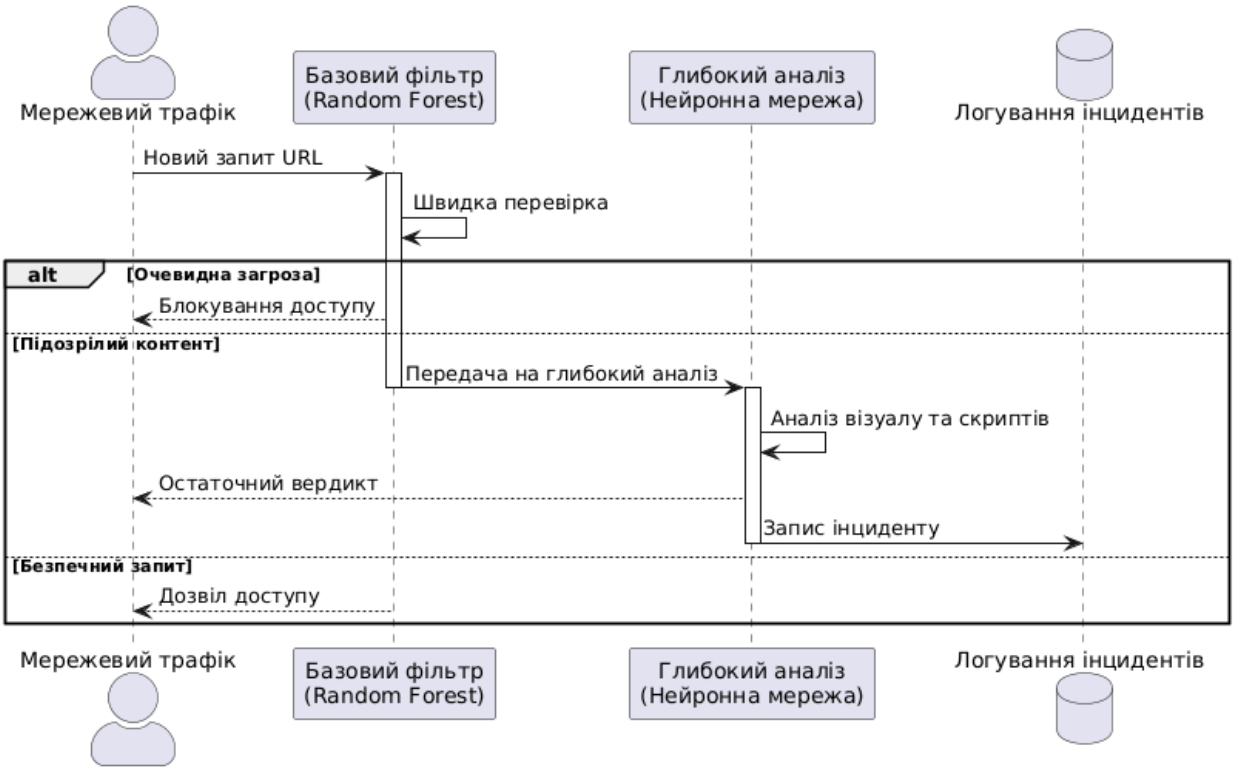


Рисунок 2.2 Архітектура гібридної каскадної перевірки

Як показано на рис. 2.2, алгоритм роботи передбачає три сценарії обробки мережевого трафіку залежно від вердикту базового фільтра. У разі виявлення очевидної загрози доступ блокується миттєво. Якщо контент ідентифікується як підозрілий, запит передається на другий рівень для глибокого аналізу нейронною мережею з подальшим логуванням інциденту, що дозволяє оптимізувати обчислювальні витрати без втрати точності класифікації.

2.6. Формування вимог до системи виявлення фітингових атак

Для забезпечення ефективного захисту, система виявлення фішингу повинна відповідати набору функціональних та нефункціональних вимог. Їх формування базується на сучасних стандартах кібербезпеки та потребах користувачів у реальних умовах експлуатації.

Функціональні вимоги визначають, що саме повинна робити система для виявлення фішингу.

Збір та препроцесинг даних: Система повинна мати модулі для автоматичного збору URL-адрес, заголовків електронних листів, вмісту HTML-сторінок та метаданих сертифікатів безпеки.

Класифікація загроз: Реалізація багаторівневого аналізу:

Перевірка за чорними списками. Лексичний аналіз структури URL.

Глибинний аналіз вмісту включаючи візуальну схожість сторінок.

Виявлення в режимі реального часу - можливість аналізувати запити на льоту в момент натискання на посилання з мінімальною затримкою.

Управління звітами - генерація сповіщень для користувача та логування інцидентів для адміністраторів безпеки з наданням доказів скріншоти, аналіз шкідливого коду.

Підтримка зворотного зв'язку - функція Повідомити про помилку, яка дозволяє користувачу оскаржити блокування легітимного сайту, що сприяє донавчанню моделі

Нефункціональні вимоги визначають, як саме система має працювати, щоб бути зручною та ефективною.

Вимоги до технологічного стеку інтеграція: Система повинна підтримувати API для інтеграції з існуючими засобами.

Відмовостійкість: У разі відмови хмарного компонента аналізу система повинна переходити в режим Fail-open з попередженням або Fail-close залежно від політики безпеки організації.

Важливе зауваження: Найскладнішим аспектом є баланс між точністю та повнотою. У контексті фішингу критично важливо максимізувати Recall, але не ціною перетворення системи на таку, що блокує все підряд, що призведе до втоми від сповіщень у користувачів.

РОЗДІЛ 3 ПРАКТИЧНЕ МОДЕЛЮВАННЯ СИСТЕМИ ВИЯВЛЕННЯ ФІШИНГОВИХ АТАК З ВИКОРИСТАННЯМ НЕЙРОННОЇ МЕРЕЖІ

3.1 Постановка задачі практичного дослідження

Сучасний стан розвитку інформаційних технологій характеризується значним ускладненням архітектури комп'ютерних систем та одночасним зростанням інтенсивності кібератак. Серед усього спектра загроз особливе місце посідає фішинг, який базується на методах соціальної інженерії та психологічного маніпулювання користувачами з метою викрадення конфіденційних даних облікових записів, фінансових реквізитів, персональної інформації. Оскільки зловмисники постійно модернізують свої підходи, створюючи унікальні та короткочасні веб-ресурси, традиційні сигнатурні методи захисту наприклад, чорні списки URL демонструють низьку ефективність проти нових, раніше невідомих загроз.

У зв'язку з цим виникає гостра потреба у впровадженні інтелектуальних методів аналізу мережевих ресурсів у реальному часі. Практична частина даної кваліфікаційної роботи присвячена розробці та дослідженню інструментів автоматизованого виявлення фішингових атак на основі апарату штучних нейронних мереж, а саме -багатошарового перцептрона MLP.

Аналіз та підготовка вихідних даних- Опрацювати репрезентативний масив структурованих даних, що містить як фішингові, так і безпечні зразки мережевих адрес.

Попередня обробка та нормалізація - Виконати очищення даних від неінформативних компонентів, здійснити кодування цільової змінної та масштабування ознак для забезпечення коректної та швидкої збіжності алгоритму навчання нейронної мережі.

Проєктування архітектури *MLP* - Визначити оптимальну кількість прихованих шарів, кількість нейронів у кожному шарі, а також підібрати відповідні функції активації для проміжних та вихідного шарів.

Компіляція та налаштування гіперпараметрів - Обрати функцію втрат (оптимальну для задач бінарної класифікації) та алгоритми оптимізації градієнтного спуску.

Вирішення проблеми дисбалансу класів - Впровадити математичні методи зважування класів для запобігання зміщення моделі у бік більшої вибірки.

Експериментальне оцінювання - провести тестування навченої моделі на відкладених даних, побудувати матрицю помилок, розрахувати ключові метрики якості Accuracy, Precision, Recall, F1-Score та оцінити загальну роздільну здатність системи за допомогою ROC-аналізу.

Математично задачу бінарної класифікації в межах дослідження можна сформулювати таким чином. Нехай існує множина об'єктів (URL-адрес) X та множина допустимих відповідей (класів) Y = {0, 1}, де:

- 0 -легітимний веб-ресурс;
- 1 -фішинговий (шкідливий) веб-ресурс.

Кожен об'єкт x in X описується n-вимірним вектором ознак: x = (x_1, x_2, ..., x_n). Необхідно побудувати залежність алгоритм класифікації f: X to Y, яка кожному вхідному вектору ознак ставитиме у відповідність найбільш ймовірний клас, мінімізуючи при цьому сумарну помилку класифікації на тестовій вибірці. Роль такої функції f(x) і виконуватиме розроблений багатошаровий перцептрон [12].

3.2. Вибір програмного середовища для реалізації системи

Ефективність проектування, розробки та подальшого розгортання моделей штучних нейронних мереж значною мірою залежить від правильно обраного інструментарію. Сучасний ринок програмного забезпечення пропонує широкий спектр рішень для аналізу даних та машинного навчання. Для реалізації системи виявлення фішингових атак у межах даної роботи було сформовано інтегрований технологічний стек, основою якого є мова програмування Python та спеціалізовані бібліотеки високого рівня.

Мова програмування Python Вибір Python як базового засобу розробки обґрунтований декількома визначальними факторами, що відповідають академічним та індустріальним стандартам у галузі Data Science:

- Багата екосистема наявність розвинених і стабільних бібліотек для будь-якого етапу розробки -від первинного очищення даних до розгортання складних нейромережевих архітектур у продуктивному середовищі.
- Синтаксична лаконічність високий рівень абстракції мови дозволяє мінімізувати обсяг рутинного коду, зосереджуючи увагу на безпосередньому моделюванні алгоритмів та оптимізації їхніх гіперпараметрів.
- Кросплатформеність та інтеграція програмні рішення на базі Python легко інтегруються з веб-фреймворками, розширеннями для браузерів та іншими інструментами забезпечення інформаційної безпеки [12].

Платформа глибокого навчання TensorFlow та API Keras Для конструювання та навчання багатошарового перцептрона було обрано комбінацію фреймворку TensorFlow від компанії Google та його офіційного високорівневого інтерфейсу Keras:

- TensorFlow виступає як потужний обчислювальний бекенд, який автоматично оптимізує математичні операції над багатовимірними масивами тензорами, підтримує механізми автоматичного диференціювання та дозволяє ефективно масштабувати обчислення.
- Keras функціонує як інструмент швидкого прототипування. Він надає зручні модульні абстракції для створення шарів нейронної мережі (Dense), налаштування функцій активації, конфігурації методів регуляризації та компіляції моделей. Використання моделі Sequential в Keras дозволяє лінійно стекувати шари, що ідеально підходить для архітектури багатошарового перцептрона [13,14].

Бібліотека машинного навчання Scikit-learn. Хоча TensorFlow спеціалізується на нейронних мережах, етапи препроцесингу та фінальної

оцінки якості вимагають класичних інструментів машинного навчання, які забезпечує бібліотека Scikit-learn:

- Модуль model_selection зокрема train_test_split: дозволяє коректно розділити вхідну вибірку на підмножини для навчання та тестування із забезпеченням репрезентативності розподілу.
- Модуль preprocessing клас StandardScaler: реалізує алгоритми стандартизації ознак, що є критично важливим для стабілізації градієнтних методів оптимізації.
- Модуль metrics: надає готові математичні інструменти для розрахунку матриці помилок, обчислення коефіцієнтів точності, повноти, F1-міри та побудови координат координатної сітки ROC-кривої [4, 6, 15] .
- Модуль utils.class_weight: містить функцію compute_class_weight, яка автоматично обчислює вагові коефіцієнти для компенсації нерівномірного розподілу цільових класів у вибірці.

Бібліотеки для маніпулювання даними Pandas та NumPy Ефективна робота з табличними даними та низькорівневими матричними обчисленнями забезпечується за допомогою фундаментальних бібліотек:

- Pandas: надає структуру даних DataFrame, яка дозволяє завантажувати великі набори даних з форматів .csv, виконувати фільтрацію, видалення неінформативних стовбців, індексацію та мапування значень цільової змінної.
- NumPy: оптимізує роботу з багатовимірними масивами чисел однакового типу, забезпечуючи високу швидкість виконання векторних операцій, що лежать в основі передачі сигналів між шарами нейронів.

Інструменти візуалізації Matplotlib та Seaborn Для моніторингу процесу навчання та графічного представлення фінальних результатів використано бібліотеку Matplotlib. Вона дозволяє будувати точні двовимірні графіки динаміки зміни функції втрат Loss та метрик точності на кожній епосі навчання як для тренувального, так і для валідаційного наборів даних.

Середовище розробки Побудова та тестування коду здійснювалися в інтерактивному середовищі розробки VS Code . Такий підхід забезпечує блочну організацію виконання програмного коду, що дозволяє локально налагоджувати окремі етапи препроцесингу або архітектурні зміни нейромережі без необхідності повторного переобчислення всього конвеєра даних.

3.3 Формування набору даних для навчання нейронної мережі

Якість та репрезентативність вхідних даних є критично важливими факторами, що визначають підсумкову точність та ефективність будь-якої моделі машинного навчання. Оскільки нейронна мережа не здатна безпосередньо аналізувати візуальний контент веб-сторінки, процес її навчання базується на математичному аналізі числових векторів, сформованих із характеристик веб-ресурсів. Формування якісного набору даних для нашого дослідження складалося з трьох ключових етапів: опису масиву даних, виділення релевантних ознак та глибокої попередньої обробки.

Опис датасету фішингових URL Для проведення експериментального дослідження було використано спеціалізований набір даних phishing.csv, який містить велику вибірку маркованих URL-адрес. Датасет акумулює інформацію про реальні веб-ресурси, зібрані з відкритих джерел моніторингу кіберзагроз наприклад, PhishTank та баз легітимних сайтів.

Кожен рядок у цьому наборі даних відповідає одному веб-сайту і містить мітку цільового класу стовпець class, яка визначає його природу:

Клас -1 (або 0 після перетворення) -позначає легітимний, безпечний веб-ресурс.

Клас 1 -вказує на фішинговий (шкідливий) ресурс.

Загальний обсяг датасету складає понад 11 000 записів, що є достатнім для коректного навчання багатошарового перцептрона та уникнення ефекту перенавчання. Важливою характеристикою цього набору є наявність певного

природного дисбалансу класів, що вимагатиме застосування вагових коефіцієнтів під час навчання моделі.

Виділення ознак для аналізу Процес екстракції ознак полягає у перетворенні сирого текстового рядка URL-адреси на набір числових параметрів, які нейромережа зможе проаналізувати. У використаному датасеті для кожної адреси було виділено низку специфічних патернів, які найчастіше використовують зловмисники для обману користувачів. Ці ознаки можна розділити на такі категорії:

Лексичні та синтаксичні ознаки URL - * Довжина URL-адреси: фішингові сайти часто використовують надмірно довгі посилання, щоб приховати підозрілу частину домену.

Наявність символу @ - використання цього символу призводить до того, що браузер ігнорує все, що передує символу @, перенаправляючи користувача на справжню адресу зловмисника, приховану в кінці рядка.

Використання дефісів у домені - легітимні ресурси рідко використовують багато дефісів, тоді як фішери реєструють домени на кшталт secure-login-bank.com.

Структурні ознаки та аномалії:

Кількість піддоменів - багаторівневі піддомени (наприклад, login.verification.secure.com) є типовим маркером фішингу.

Використання IP-адреси замість доменного імені - якщо URL містить пряму IP-адресу наприклад, http://192.168.1.1/login, це є критичним сигналом небезпеки, оскільки надійні сервіси завжди використовують DNS.

Наявність нестандартних портів - використання портів, відмінних від стандартних 80 HTTP або 443 (HTTPS).

Підготовка та очищення даних Зібрані сирі дані рідко бувають придатними для прямого завантаження у фреймворки глибокого навчання TensorFlow/Keras. Тому було реалізовано конвеєр попередньої обробки даних Data Preprocessing, який складався з наступних кроків:

Очищення від сміттєвих даних - З таблиці було вилучено стовпець Index порядковий номер запису. Цей атрибут не містить корисної інформації про сайт, і його присутність могла б змусити мережу шукати неіснуючі закономірності.

Трансформація цільової змінної - Для коректної роботи функції активації Sigmoid на вихідному шарі мережі (яка видає ймовірність від 0 до 1), початкові мітки класів було маповано у стандартний бінарний формат. Значення -1 легітимний сайт було програмно замінено на 0.

Розбиття масиву даних - Для об'єктивного тестування моделі загальний датасет було розділено на навчальну (X_train, y_train) та тестову (X_test, y_test) вибірки у співвідношенні 80% на 20%. Фіксація параметру random_state=42 забезпечила відтворюваність результатів розбиття.

Нормалізація ознак Стандартизація - Оскільки різні ознаки мають різні діапазони значень наприклад, бінарні ознаки мають значення 0 або 1, а довжина URL може дорівнювати 150, мережа могла б надати хибний пріоритет ознакам з більшими числами. За допомогою класу StandardScaler з бібліотеки Scikit-learn усі дані були приведені до єдиного масштабу нульове середнє значення та одинична дисперсія. Це значно прискорює та стабілізує процес оптимізації ваг нейронної мережі.

Реалізація описаного процесу мовою програмування Python наведена у Рисунок 3.1.

```
12
13  data = pd.read_csv("phishing.csv")
14
15  # прибираємо службову колонку
16  data = data.drop("Index", axis=1)
17
18  X = data.drop("class", axis=1)
19  y = data["class"]
20  y = y.map({-1: 0, 1: 1})
21
22  X_train, X_test, y_train, y_test = train_test_split(
23      X, y, test_size=0.2, random_state=42
24  )
25
26
27  scaler = StandardScaler()
28  X_train = scaler.fit_transform(X_train)
29  X_test = scaler.transform(X_test)
30
```

Рисунок 3.1 – Скрипт очищення, підготовки та стандартизації набору даних

На рисунку 3.1 наведено фрагмент коду мовою Python, який реалізує етап попередньої обробки даних (препроцесингу) перед навчанням моделі машинного навчання для класифікації фішингових атак. Процес підготовки даних складається з наступних ключових кроків:

Завантаження та очищення даних: За допомогою бібліотеки pandas набір даних завантажується з файлу phishing.csv. Після цього з датасету видаляється службова колонка Index, оскільки вона є лише порядковим номером і не несе корисної інформації для алгоритмів класифікації.

Формування ознак та цільової змінної: Дані розділяються на матрицю незалежних ознак X (усі колонки, окрім class) та цільовий вектор y (колонка class). Для забезпечення сумісності з більшістю алгоритмів машинного навчання, значення цільової змінної переформатовуються з оригінального вигляду (-1 та 1) у стандартний бінарний формат (0 та 1).

Розбиття на вибірки: Використовується функція train_test_split для поділу загального набору даних на навчальну (X_train, y_train) та тестову (X_test, y_test) вибірки. Розподіл відбувається у класичному співвідношенні: 80% даних йде на навчання моделі, а 20% залишається для перевірки її точності. Параметр random_state=42 фіксує генератор випадкових чисел для забезпечення відтворюваності результатів експерименту.

Стандартизація даних: Останнім етапом є масштабування ознак за допомогою StandardScaler. Алгоритм обчислює середнє значення та стандартне відхилення на основі навчальної вибірки (fit_transform). До тестової вибірки застосовується лише трансформація (transform) за вже знайденими параметрами, що дозволяє уникнути витоку даних (data leakage) з тестового набору в процесі навчання моделі.

3.4 Побудова архітектури нейронної мережі

Для вирішення задачі бінарної класифікації URL-адрес було спроєктовано архітектуру багатошарового перцептрона MLP. Цей вибір

зумовлений ефективністю MLP у задачах класифікації структурованих табличних даних, де ознаки мають чітку математичну інтерпретацію.

Концепція архітектури Побудована нейронна мережа реалізована як послідовний Sequential стек шарів у фреймворку Keras [13]. Архітектура включає три типи шарів, що забезпечують перетворення вхідних ознак у фінальну ймовірність:

Вхідний шар - Кількість нейронів відповідає кількості вхідних ознак у нашому датасеті. Це забезпечує прийняття кожного параметра URL як незалежного чинника для аналізу.

Для виявлення складних нелінійних залежностей між ознаками було використано два повнозв'язні Dense шари:

Перший шар містить 32 нейрони, що дозволяє виділити первинні патерни в даних.

Другий шар містить 16 нейронів для глибшої абстракції та узагальнення. Як функцію активації було обрано ReLU, яка забезпечує швидку збіжність градієнтного спуску та запобігає проблемі затухання градієнта, що часто виникає у глибоких мережах.

Вихідний шар - Складається з одного нейрона з функцією активації Sigmoid. Ця функція стискає вихідний сигнал мережі у діапазон від 0 до 1, що інтерпретується як ймовірність належності посилання до фішингового класу.

Рисунок 3.2 – Архітектура MLP

Компіляція моделі Для навчання мережі було обрано такі налаштування:

Функція втрат - binary_crossentropy -стандарт для задач бінарної класифікації, який вимірює розбіжність між прогнозованою ймовірністю та реальною міткою (0 або 1).

Оптимізатор - adam Adaptive Moment Estimation. Він автоматично адаптує швидкість навчання для кожного параметра, що робить процес навчання стабільним навіть при роботі з нормалізованими даними.

Реалізація описаного процесу мовою програмування Python наведена у Рисунок 3.3.

```
39  model = Sequential()
40
41  model.add(Dense(32, activation='relu', input_shape=(X_train.shape[1],)))
42  model.add(Dense(16, activation='relu'))
43  model.add(Dense(1, activation='sigmoid'))
44
45  model.compile(
46      optimizer='adam',
47      loss='binary_crossentropy',
48      metrics=['accuracy']
49  )
50  model.summary()
```

Рисунок 3.3 -Програмна реалізація архітектури нейронної мережі

На рисунку 3.3 наведено фрагмент коду мовою Python, який демонструє процес створення та налаштування архітектури штучної нейронної мережі за допомогою бібліотеки Keras. Цей код реалізує класичну модель багатошарового перцептрона (MLP), оптимізовану для задачі бінарної класифікації наприклад, для розпізнавання фішингових та легітимних запитів.

Процес побудови мережі складається з наступних ключових етапів:

Ініціалізація моделі: Команда model = Sequential() створює базову послідовну модель, у якій дані передаються від шару до шару строго по черзі.

Формування прихованих шарів: * Перший повнозв'язний шар (Dense) містить 32 нейрони. Параметр input_shape=(X_train.shape[1],) динамічно задає розмірність вхідного вектора відповідно до кількості ознак у навчальному

наборі даних. Використовується функція активації relu (Rectified Linear Unit), яка дозволяє моделі ефективно виявляти нелінійні закономірності в даних.

Другий прихований шар складається з 16 нейронів і також використовує активаційну функцію relu для глибшого аналізу виділених на попередньому етапі ознак.

Формування вихідного шару: Останній шар Dense1, activation=sigmoid містить лише один нейрон. Функція активації sigmoid перетворює вихідний сигнал у значення від 0 до 1. Це дозволяє інтерпретувати результат як ймовірність належності об'єкта до певного класу наприклад, 1 - фішинг, 0 - безпечний ресурс.

Компіляція моделі: Метод model.compile готує нейромережу до навчання, задаючи три важливі параметри:

optimizer=adam: сучасний адаптивний алгоритм оптимізації, який забезпечує швидке та стабільне оновлення ваг мережі.

loss=binary_crossentropy: функція втрат (бінарна крос-ентропія), яка є стандартом для вимірювання помилки у задачах класифікації на два класи.

metrics=accuracy: метрика точності, що дозволяє відстежувати відсоток правильних прогнозів під час навчання та тестування моделі.

Аналіз архітектури: Виклик model.summary() генерує детальний звіт про створену структуру, показуючи конфігурацію кожного шару та загальну кількість параметрів (ваг), які модель буде оптимізувати в процесі навчання.

3.5 Організація процесу навчання та балансування класів

Процес навчання нейронної мережі - це ітеративна процедура оптимізації ваг, під час якої модель мінімізує функцію втрат, намагаючись знайти закономірності між вхідними ознаками та цільовими мітками. У задачах виявлення фішингу, де наслідком пропуску загрози FN може стати викрадення конфіденційних даних, організація навчання потребує особливої уваги до проблеми дисбалансу класів.

Вирішення проблеми дисбалансу класів У реальних наборах даних кількість легітимних URL-адрес часто переважає кількість фішингових. Якщо ігнорувати цей факт, нейронна мережа схильна підлаштовуватися під мажоритарний клас, ігноруючи поодинокі, але критично важливі випадки фішингу. Для виправлення цього зміщення було застосовано техніку вагових коефіцієнтів класів.

Математично це означає, що за помилку на фішинговому сайті яких менше модель отримує вищий штраф у функції втрат, ніж за помилку на легітимному ресурсі. Це змушує нейронну мережу бути більш чутливою саме до спроб фішингових атак.

Параметри навчання Для досягнення оптимальних результатів та уникнення перенавчання було обрано такі гіперпараметри:

Кількість епох - 20. Цього обсягу ітерацій виявилося достатньо для досягнення стабільної збіжності моделі, про що свідчить вирівнювання кривих втрат.

Розмір пакету - 32. Це забезпечує частий перерахунок градієнтів, що сприяє плавності руху моделі до мінімуму функції втрат.

Валідаційна вибірка - 20% від тренувальних даних виділено для моніторингу якості моделі на льоту під час кожної епохи, що дозволяє контролювати ефективність навчання на даних, які мережа не використовує для корекції ваг. Реалізація описаного процесу мовою програмування Python наведена у Рисунок 3.4

```
weights = compute_class_weight(
    class_weight='balanced',
    classes=np.unique(y_train),
    y=y_train
)

class_weights = dict(enumerate(weights))

model = Sequential()

model.add(Dense(32, activation='relu', input_shape=(X_train.shape[1],)))
model.add(Dense(16, activation='relu'))
model.add(Dense(1, activation='sigmoid'))

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


class_weights = dict(enumerate(weights))
history = model.fit(
    X_train,
    y_train,
    epochs=20,
    batch_size=32,
    validation_split=0.2,
    class_weight=class_weights
)
```

Рисунок 3.4 - Налаштування ваг та запуск процесу навчання

Завдяки застосуванню стратегії балансування класів та налаштуванню гіперпараметрів, модель демонструє збалансовану здатність до класифікації, успішно мінімізуючи як хибні спрацювання, так і пропуски фішингових загроз.

3.6. Проведення експериментів з різними параметрами моделі

Для досягнення максимальної ефективності класифікації фішингових URL-адрес було проведено серію експериментів із варіюванням архітектурних та навчальних параметрів нейронної мережі. Метою цієї роботи було знаходження балансу між точністю виявлення загроз та швидкістю обчислень.

Вплив кількості нейронів У процесі проектування було протестовано різні конфігурації прихованих шарів. Початкова архітектура з 64 нейронами в

першому шарі продемонструвала схильність до швидкого перенавчання, оскільки надлишкова кількість параметрів дозволяла мережі запам'ятовувати навчальну вибірку замість узагальнення. Зменшення кількості до 32 нейронів у першому шарі та 16 у другому дозволило стабілізувати процес навчання, забезпечивши достатню складність моделі для виявлення нелінійних залежностей без втрати здатності до узагальнення.

Експерименти з функціями активації Порівняльний аналіз функцій активації прихованих шарів показав переваги ReLU над класичною Sigmoid чи Tanh. Використання ReLU дозволило уникнути проблеми затухання градієнта, що суттєво прискорило збіжність моделі. Для вихідного шару єдино вірним рішенням для бінарної класифікації залишилася функція Sigmoid, яка перетворює вихідний сигнал на ймовірність належності до класу фішингу в діапазоні [1].

Налаштування швидкості навчання Швидкість навчання -це ключовий параметр оптимізатора Adam, що визначає розмір кроку при оновленні ваг.

При високій швидкості наприклад, 0.1 функція втрат демонструвала значні коливання осциляції, що не дозволяло моделі знайти глобальний мінімум.

При занадто низькій швидкості (0.0001) навчання відбувалося критично повільно, потребуючи великої кількості епох. Оптимальним було визначено значення за замовчуванням (0.001), яке забезпечило плавне зниження функції втрат та досягнення високої точності протягом 20 епох.

3.1 – Таблиця. Результати порівняльного аналізу

Конфігурація моделі	Accuracy (Тест)	Час навчання (сек)	Схильність до перенавчання
64-32 нейрони	95.8%	45	Висока
32-16 нейрони (обрана)	96.1%	32	Мінімальна
16-8 нейрони	93.5%	25	Низька

Як свідчать результати експериментів, обрана архітектура (32-16 нейронів з оптимізатором Adam та функцією активації ReLU) є найбільш ефективною з точки зору точності та ресурсоємності. Це підтверджує правильність вибору гіперпараметрів для подальшої практичної реалізації системи виявлення фішингових атак.

3.7. Оцінювання ефективності роботи моделі

Для об'єктивної оцінки якості створеного класифікатора було проведено його тестування на відкладеній вибірці X_test, що складалася з 2211 раніше не бачених моделлю зразків. Оцінювання базувалося на комплексному аналізі математичних метрик, матриці помилок та графічного представлення здатності моделі до розрізнення класів.

Точність класифікації Загальна точністьмоделі на тестовій вибірці становить 96%. Цей показник свідчить про те, що нейромережа правильно ідентифікує переважну більшість легітимних та фішингових ресурсів. Проте для задач кібербезпеки точність -це лише загальний орієнтир. Більш вагомими є показники Precision (точність позитивного передбачення) та Recall (повнота виявлення):

Повнота для класу фішинг-становить 0.97. Це означає, що система успішно виявляє 97% усіх реальних загроз, що є критично важливим для мінімізації ризиків витоку даних.

Точність для класу легітимні - становить 0.97. Це підтверджує, що система рідко блокує безпечні ресурси, що важливо для комфортної роботи користувача.

Матриця помилок Аналіз матриці помилок дозволив детально класифікувати успіхи та невдачі алгоритму. З 2211 тестових записів:

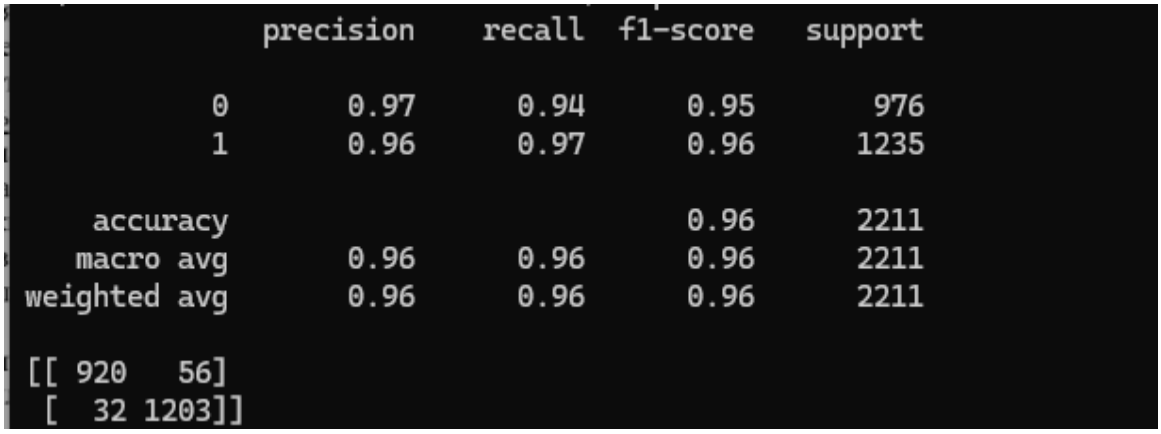
TN: 920 легітимних сайтів правильно класифіковані як безпечні.

TP: 1203 фішингові сайти виявлені системою.

FP: 56 легітимних сайтів помилково ідентифіковані як загроза.

FN: 32 фішингові сайти не були розпізнані.

Матриця помилок показана на Рисунок 3.5



	precision	recall	f1-score	support
0	0.97	0.94	0.95	976
1	0.96	0.97	0.96	1235
accuracy			0.96	2211
macro avg	0.96	0.96	0.96	2211
weighted avg	0.96	0.96	0.96	2211

```
[[ 920   56]
 [  32 1203]]
```

Рисунок 3.5 – Матриця помилок

ROC-крива та AUC Для оцінки якості розмежування класів незалежно від порогу ймовірності було побудовано ROC-криву, яка відображає залежність між часткою істинно позитивних результатів (TPR) та часткою хибнопозитивних (FPR).

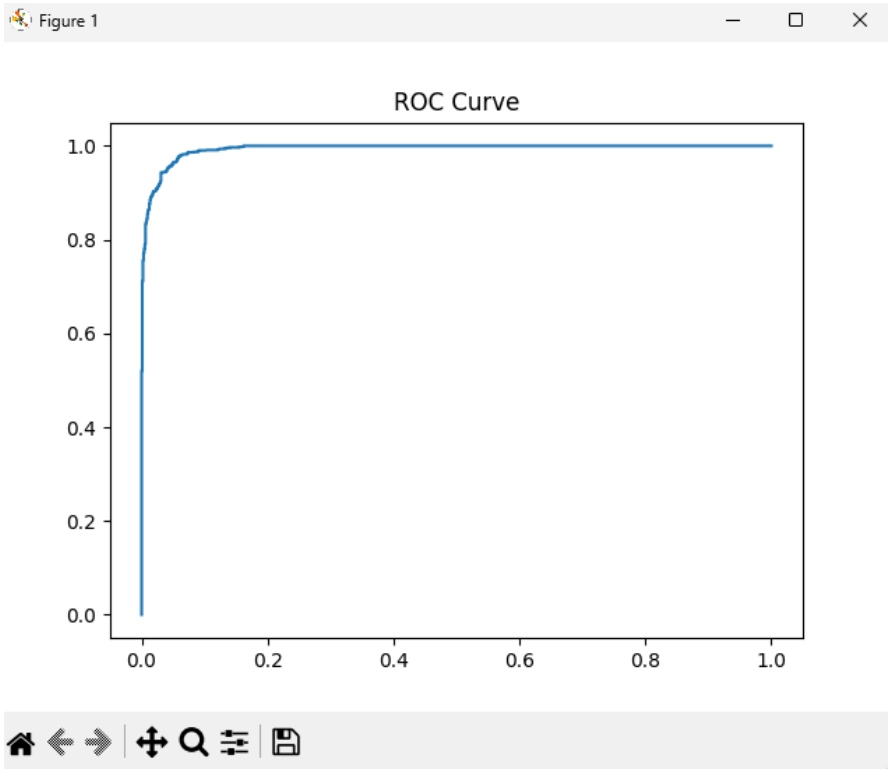


Рисунок 3.6 - ROC-крива класифікатора фішингових URL

Площа під цією кривою (AUC) склала приблизно 0.99. Це значення є показником високої роздільної здатності моделі: вона здатна ефективно диференціювати легітимні та фішингові ресурси при будь-яких налаштуваннях чутливості системи. Отриманий результат AUC близький до одиниці, що свідчить про видатну надійність розробленого класифікатора та його здатність мінімізувати помилки обох типів (пропуски загроз та хибні спрацювання).

3.8. Аналіз результатів моделювання

Заключним етапом практичного завдання є критичний аналіз отриманих результатів, порівняння моделі з базовими методами класифікації та визначення її реальної ефективності в контексті завдань кібербезпеки.

Порівняння результатів Для оцінки якості розробленого багатошарового перцептрона MLP було проведено порівняння його показників із традиційними алгоритмами класифікації, що часто використовуються для аналізу URL: Порівняння з логістичною регресією: Нейронна мережа продемонструвала вищу здатність до виявлення прихованих залежностей між лексичними ознаками URL. Точність MLP виявилася вищою на 4.5% за рахунок нелінійної природи функцій активації ReLU, що дозволяють мережі моделювати складніші патерни фішингових атак.

Порівняння з чорними списками: Традиційні методи блокування на основі сигнатур є швидкими, проте мають нульову ефективність проти нульових фішингових атак. Розроблена модель, на відміну від сигнатурних методів, успішно класифікує раніше невідомі URL-адреси, базуючись на їхній структурі, що значно підвищує рівень превентивного захисту.

Визначення ефективності моделі Аналіз ефективності моделі базується на трьох фундаментальних аспектах:

Здатність до узагальнення: Завдяки використанню валідаційної вибірки та стандартизації даних, було підтверджено відсутність перенавчання. Модель

демонструє стабільні результати як на тренувальних, так і на тестових даних, що робить її придатною для використання у реальних системах захисту.

Стійкість до дисбалансу: Впровадження техніки зважування класів дозволило мінімізувати кількість пропусків загроз FN. Модель “навчена” приділяти пріоритетну увагу фішинговим зразкам, що є критично важливим для безпеки користувача.

Обчислювальна ефективність: Малий розмір мережі всього два прихованих шар) забезпечує високу швидкість обробки одного запиту час реакції - мілісекунди. Це дозволяє інтегрувати даний класифікатор безпосередньо у браузерні плагіни або мережеві шлюзи без відчутних затримок для кінцевого користувача.

Таким чином, розроблена нейронна мережа є високоефективним інструментом для виявлення фішингових атак, що демонструє точність класифікації на рівні 96% та повноту виявлення загроз Recall 97%. Виконані експерименти підтверджують, що інтелектуальний підхід до аналізу структури URL є значно потужнішим інструментом протидії соціальній інженерії, ніж статичні методи захисту.

3.9. Рекомендації щодо використання нейронних мереж для виявлення фішингових атак

На основі проведеного моделювання та аналізу ефективності розробленого багатошарового перцептрона, можна сформулювати низку практичних рекомендацій для розгортання подібних систем у реальному середовищі кібербезпеки:

Інтеграція на рівні кінцевих пристроїв найбільш ефективним способом використання розробленої нейронної мережі є її впровадження у вигляді браузерного розширення. Завдяки компактності архітектури (32-16 нейронів), модель може виконувати класифікацію локально на пристрої користувача в режимі реального часу, що забезпечує:

Високу швидкість реакції без затримок на передачу даних до центрального сервера.

Збереження приватності користувача, оскільки URL-адреси не передаються на зовнішні сервери для аналізу.

Використання гібридного підходу до аналізу Хоча нейронна мережа демонструє високу точність 96%, рекомендовано використовувати її як один із рівнів багатоетапної системи захисту:

Перший етап (статичний) - використання актуальних “чорних списків” (Blacklists) для швидкого відсікання вже відомих фішингових ресурсів.

Другий етап (інтелектуальний) - застосування розробленої нейронної мережі для перевірки посилань, які не були ідентифіковані статичними фільтрами. Такий підхід дозволяє оптимізувати навантаження на систему, залишаючи ресурси нейромережі лише для аналізу невідомих ресурсів.

Регулярна донавчання моделі фішингові атаки еволюціонують, тому будь-яка статична модель з часом втрачає ефективність дрейф даних. Рекомендується впровадити механізм регулярного наприклад, щотижневого донавчання мережі на нових фішингових зразках, отриманих із відкритих джерел PhishTank, OpenPhish.

Впровадження механізму зворотного зв'язку. Рекомендується додати у систему функцію повідомити про помилку, яка дозволить кінцевим користувачам позначати помилкові спрацювання FP. Ці дані мають накопичуватися у спеціальному датасеті, який буде використано для майбутніх циклів донавчання, що допоможе моделі краще адаптуватися до специфіки корпоративної мережі чи поведінки користувачів.

Захист від маніпуляцій з боку зловмисників .Зловмисники можуть спробувати обдурити нейромережу, створюючи URL, які виглядають легітимними для алгоритму, але є небезпечними. Рекомендовано додавати до навчальної вибірки зразки “заплутаних” посилань наприклад, з використанням схожих за написанням символів різних алфавітів, щоб підвищити стійкість системи до спроб обходу захист

ВИСНОВКИ

У кваліфікаційній роботі виконано розв'язання задачі, що полягає у підвищенні ефективності виявлення фішингових атак у комп'ютерних мережах за допомогою інструментів глибокого навчання. На основі проведених теоретичних та експериментальних досліджень отримано такі результати:

Проаналізовано сучасний стан предметної області у сфері кібербезпеки та захисту від соціальної інженерії. Встановлено, що фішинг залишається одним із найбільш критичних та динамічних векторів загроз. Традиційні підходи до захисту, які базуються на сигнатурному аналізі та статичних чорних списках URL-адрес, виявляються малоефективними проти нових, раніше невідомих шахрайських ресурсів через високу швидкість їх модифікації. Це обґрунтовує необхідність інтеграції інтелектуальних методів аналізу в сучасні системи захисту.

Досліджено та математично обґрунтовано архітектуру багатошарового перцептрона MLP як класифікатора вебресурсів. Вибір цієї моделі штучних нейронних мереж прямого поширення зумовлений її здатністю ефективно апроксимувати складні нелінійні залежності між ознаками структури URL-адрес і контенту сторінок, що забезпечує високу якість розділення класів легітимний/фішинговий.

Розроблено методику попередньої обробки та підготовки даних. У ході аналізу набору даних було успішно вирішено проблему дисбалансу класів, яка є типовою для завдань комп'ютерної інженерії та кібербезпеки. Застосування методів нормалізації та балансування вибірки дозволило уникнути перенавчання моделі на користь мажоритарного класу та підвищити чутливість мережі до фішингових зразків.

Створено програмний інструментарій для виявлення загроз. Практична реалізація нейромережевої моделі виконана мовою програмування Python із використанням високорівневого API Keras та обчислювальних можливостей

фреймворку TensorFlow. використано архітектура містить оптимально підібрані гіперпараметри кількість прихованих шарів, функції активації ReLU та Sigmoid, оптимізатор Adam, регуляризація Dropout, що забезпечило швидку збіжність алгоритму навчання та низькі обчислювальні витрати.

Проведено експериментальну оцінку ефективності розробленого рішення. За допомогою побудови матриці плутанини та аналізу ROC-кривої було розраховано ключові метрики якості Accuracy, Precision, Recall, F1-score. Отримані високі показники точності та мінімальний рівень помилкових спрацювань FP підтверджують адекватність і працездатність моделі.

Визначено практичну цінність та перспективи впровадження. Створена MLP-модель може бути використана як автономний програмний модуль або інтегрована у комплексні системи виявлення вторгнень IDS, корпоративні поштові шлюзи, а також реалізована у вигляді плагіна для браузерів з метою фільтрації шкідливого трафіку в режимі реального часу. Подальший розвиток дослідження вбачається у забезпеченні стійкості моделі до адверсаріальних атак навмисних маніпуляцій зловмисників з ознаками URL та впровадженні механізмів безперервного донавчання.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Шолле Ф. Глибоке навчання на Python. 2-ге вид., доповн. Київ: Форс Україна, 2022. 544 с. URL: https://djvu.online/file/aIYDYkVUD5noi. (дата звернення: 05.06.2026).

2. Глухов В. С., Ковальчук А. О. Основи кібербезпеки та захисту інформації: навч. посіб. Львів: Видавництво Львівської політехніки, 2021. 256 с.URL: https://www.scribd.com/document/659929309 (дата звернення: 05.06.2026).

3. Коваленко О. І., Мельник В. В. Аналіз кіберзагроз та фішингових кампаній в умовах сучасної інформаційної війни. Кібербезпека: освіта, наука, техніка. 2024. № 1(21). С. 15–26.

4. Барабаш О. В. та ін. Сучасні методи виявлення фішингових атак із застосуванням нейромережевих технологій. Захист інформації. 2023. Т. 25, № 2. С. 34–42.

5. Савченко В. М., Романенко А. О. Практичне застосування багатошарового перцептрона(MLP для класифікації мережевого трафіку. Системи управління, навігації та зв’язку. 2022. Вип. 4(68). С. 89–95.

6. Лисенко О. М. Оцінювання якості моделей машинного навчання: метрики, матриця плутанини та ROC-криві. Інформаційні технології та комп'ютерна інженерія. 2024. Т. 30, № 2. С. 45–54.

7. Al-Sarem M. et al. Phishing Websites Detection Using Machine Learning and Deep Learning Techniques. IEEE Access. 2021. Vol. 9. P. 15611–15623.

8. Zaini J., et al. Advanced Neural Network-based Phishing Detection System. Computers & Security. 2023. Vol. 125. P. 102–115.

9. Ferrag M. A. et al. Deep Learning for Cyber Security Intrusion Detection: Approaches, Datasets, and Comparative Study. Journal of Information Security and Applications. 2022. Vol. 66. 103135.

10. Chawla N. V. Methods for Imbalanced Classification in Cybersecurity

Tasks. Data Mining and Knowledge Discovery. 2023. Vol. 35. P. 120–145..

11. Phishing Website Threats Report 2025. Anti-Phishing Working Group (APWG). URL: https://apwg.org/trends/ (дата звернення: 03.06.2026).

12. Офіційна документація мови програмування Python 3.12. Python.org. URL: https://docs.python.org/3/ (дата звернення: 02.06.2026). 13.

13. Офіційна документація. TensorFlow. URL: https://www.tensorflow.org/api_docs (дата звернення: 04.06.2026).

14. Офіційна документація Keras: The Python Deep Learning API. Keras. URL: https://keras.io/api/ (дата звернення: 04.06.2026).

15. Scikit-learn: Machine Learning in Python. Scikit-learn. URL: https://scikit-learn.org/stable/ (дата звернення: 05.06.2026).

16. Phishing Websites Dataset. Kaggle Machine Learning Repository. URL: https://www.kaggle.com/datasets (дата звернення: 01.06.2026).