ISSN 2519-2310 (Online)

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
MINISTRY OF EDUCATION AND SCIENCE OF UKRAINE

ХАРКІВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ імені В.Н.КАРАЗІНА
V.N. KARAZIN KHARKIV NATIONAL UNIVERSITY

КОМП'ЮТЕРНІ НАУКИ ТА КІБЕРБЕЗПЕКА
COMPUTER SCIENCE AND CYBERSECURITY
(CS&CS)

№ 2(26) 2024
Issue 2(26) 2024

Заснований 2015 року

Міжнародний електронний науково-теоретичний журнал
International electronic scientific journal

УДК 004(051)

Засновник і видавець Харківський національний університет імені В.Н. Каразіна Міністерства освіти і науки України. Засновано у 2015 році. Періодичність виходу –2 рази на рік.

В журналі публікуються наукові статті з теоретичних і науково-технічних проблем, що пов'язані зі створенням ефективних засобів комп'ютерних інформаційно-комунікаційних систем та питань захисту інформації, на основі передових математичних методів, інформаційних технологій і технічних засобів.

Схвалено до розміщення в мережі Інтернет Вченою радою Харківського національного університету імені В.Н. Каразіна (23.12.2024 р., Протокол № 26).

DOI (Онлайн): **10.26565/2519-2310-2024-2**.

Редакція:
Харківський національний університет імені В.Н. Каразіна
майдан Свободи, 6, офіс 315а, Харків, 61022, Україна (*Північний корпус університету, 3 поверх*)

Електронна пошта: cscsjournal@karazin.ua
Веб-сторінка: http://periodicals.karazin.ua/cscs (Open Journal System)

Автори опублікованих матеріалів несуть повну відповідальність за достовірність наведених фактів, власних імен тощо. Опубліковані статті пройшли внутрішнє та зовнішнє рецензування.

UDC 004(051)

Founder and publisher V.N. Karazin Kharkiv National University of the Ministry of Education and Science of Ukraine. Established in 2015. Published 2 times a year.

The journal publishes research articles on theoretical, scientific and technical problems of effective facilities development for computer information-communication systems and information security questions based, on advanced mathematical methods, information technologies and technical means.

Approved for placement on the Internet by Academic Council of the Karazin Kharkiv National University (December 23, 2024, Protocol No.26).

The journal has Digital Object Identifier: **10.26565/2519-2310-2024-2** (Online).

Editorial office:
V.N. Karazin Kharkiv National University
Svobody Sq., 6, office 315a, Kharkiv, 61022, Ukraine (*North building of University, 3th floor*)

E-mail: cscsjournal@karazin.ua
Web-page: http://periodicals.karazin.ua/cscs (Open Journal System)

The authors of the published materials are solely responsible for the selection, accuracy of the facts, proper names, etc. Published articles have been internally and externally peer reviewed.

ЗМІСТ

CONTENTS

ISSN 2519-2310 (Online)

КОМП'ЮТЕРНІ НАУКИ ТА КІБЕРБЕЗПЕКА, 2024, № 2 (26)

COMPUTER SCIENCE AND CYBERSECURITY, 2024, Iss. 2 (26)

DOI: https://doi.org/10.26565/2519-2310-2024-2-01
УДК 004.056.5

ДОСЛІДЖЕННЯ ТА КЛАСИФІКАЦІЯ ОСНОВНИХ ТИПІВ АТАК НА СИСТЕМИ ШТУЧНОГО ІНТЕЛЕКТУ В КІБЕРБЕЗПЕЦІ

Владислав Вілігура[1], PhD, старший викладач кафедри КІСМТ, e-mail: v.v.vilihura@karazin.ua, ORCID: https://orcid.org/0000-0002-1137-2382

Єлизавета Острянська[1], молодший науковий співробітник, e-mail: antelizza@gmail.com, ORCID: https://orcid.org/0000-0003-1412-8470

[1]*Харківський національний університет імені В.Н. Каразіна, майдан Свободи, 4, Харків, 61022, Україна*

Рукопис надійшов 1 листопада 2024 р. Отримано після рецензування 2 грудня 2024 р. Прийнято 23 грудня 2024 р.

Анотація: Сучасний розвиток штучного інтелекту (ШІ) та машинного навчання (ML) відкриває нові можливості у сфері кібербезпеки, проте водночас створює серйозні виклики у вигляді інтелектуальних кібератак. Дослідження присвячене аналізу та класифікації способів використання ШІ у зловмисних цілях та вивченню ефективних методів протидії таким загрозам. Зокрема, стаття охоплює основні види атак, що використовують технології ML, які демонструють, як зловмисники можуть маніпулювати алгоритмами машинного навчання, підривати довіру до даних та обходити системи захисту. Окрему увагу приділено механізмам атак отруєння даних, так як вони вважаються найбільш небезпечними при машинному навчанні, які передбачають внесення шкідливих даних у процес навчання моделей, що призводить до викривлення результатів та підриву ефективності алгоритмів безпеки. Також розглядаються атаки проникнення, у яких атакуючі створюють унікальні зразки даних, що можуть залишатися невидимими для традиційних систем виявлення загроз. Атаки на конфіденційність аналізуються як спосіб отримання конфіденційної інформації з ML-моделей, що може використовуватися для викрадення даних користувачів. Атаки зловживання демонструють, як зловмисники можуть використовувати ШІ-інструменти для автоматизації атак, масштабування фішингових кампаній та аналізу слабких місць захисних систем. Актуальність дослідження зумовлена тим, що традиційні підходи до кіберзахисту вже не здатні ефективно протистояти загрозам, які адаптуються та еволюціонують завдяки машинному навчанню. Стаття наголошує на критичній важливості дослідження методів захисту, зокрема побудови надійних систем машинного навчання, що мають вбудовані механізми виявлення аномалій та адаптацію до нових загроз. Одним із ключових підходів є федеративне навчання, яке дозволяє тренувати моделі без централізованого зберігання даних, зменшуючи ризик витоку інформації. Також розглянуто розвиток глибокого навчання у сфері кіберзахисту, що дозволяє аналізувати поведінкові патерни загроз у режимі реального часу. Важливим аспектом залишається поєднання технологічних заходів із людським контролем, оскільки, незважаючи на потужність ШІ-інструментів, людський фактор залишається ключовим у процесі забезпечення кібербезпеки. Отже, стаття демонструє баланс між можливостями та загрозами ШІ у сфері кібербезпеки, підкреслюючи необхідність подальших досліджень у напрямку стійких ML-моделей, які можуть ефективно протистояти атакам. Без належного регулювання та контролю ШІ може стати не лише

захисником, а й інструментом зловмисників, що вимагає розробки нових стратегій безпеки та міжнародного врегулювання у сфері кіберзахисту.

Ключові слова: *штучний інтелект, кібератаки, машинне навчання, кібербезпека, федеративне навчання*

Як цитувати: Вілігура В., Острянська Є. Дослідження та класифікація основних типів атак на системи штучного інтелекту в кібербезпеці. *Комп'ютерні науки та кібербезпека.* 2024; № 2(26): С. 6–18. https://doi.org/10.26565/2519-2310-2024-2-01

In cites: Vilihura V., Ostrianska Ye. (2024). Research and classification of the main types of attacks on artificial intelligence systems in cybersecurity. *Computer Science and Cybersecurity.* 2(26): 6–18. https://doi.org/10.26565/2519- 2310-2024-2-01 (in Ukrainian)

1. Вступ

У сучасному світі штучний інтелект (ШІ) дедалі активніше інтегрується в різні сфери людської діяльності – від фінансів і медицини до автономного транспорту та кібербезпеки. Разом із розвитком технологій ШІ виникають нові виклики, зокрема загрози, пов'язані з атаками на штучний інтелект. Такі атаки можуть мати серйозні наслідки, включаючи компрометацію безпеки даних, маніпулювання алгоритмами та створення вразливостей у критично важливих системах.

Машинне навчання, як основа більшості сучасних ШІ-систем, дозволяє їм самостійно знаходити закономірності в даних, адаптуватися до змін і вдосконалюватися з часом. Однак, оскільки моделі ML значною мірою покладаються на якість вхідних даних та гіпотези, на яких вони ґрунтуються, вони стають вразливими до певних типів атак. Зловмисники можуть маніпулювати навчальними даними, підмінювати вхідні параметри або експлуатувати слабкі місця алгоритмів для досягнення своїх цілей.

По суті, методологія машинного навчання, яка використовується в сучасних системах штучного інтелекту, сприйнятлива до атак через загальнодоступні API, які розкривають модель, і проти платформ, на яких вони розгорнуті. Для атак на моделі безпеки зловмисники можуть порушити захист конфіденційності та захист даних і моделі, просто використовуючи загальнодоступні інтерфейси та надаючи вхідні дані, які знаходяться в межах прийнятного діапазону. У цьому сенсі виклики, з якими стикається AML, подібні до тих, що постають перед криптографією. Сучасна криптографія спирається на безпечні алгоритми в теоретичному сенсі інформації. Таким чином, люди повинні зосередитися лише на їх надійному та безпечному впровадженні, що і є першочерговою задачею для спільноти науковців-криптологів. На відміну від криптографії, немає інформаційно-теоретичних доказів безпеки для широко використовуваних алгоритмів машинного навчання. Як наслідок, багато досягнень у розробці засобів пом'якшення різних класів атак мають емпіричний та обмежений характер.

Але багато компаній та структур в останні роки активно працюють над врегулюванням використання ШІ в свої системах. Так, наприклад, серед широкого спектру діяльності NIST робить внесок у дослідження, стандарти, оцінки та дані, необхідні для розвитку, використання та забезпечення надійного штучного інтелекту (ШІ). У 2024 NIST опублікував звіт [1] щодо загроз на основі машинного навчання, а 2025 доповнив його [25]. У цьому звіті розроблено таксономію понять і визначено термінологію у сфері змагального машинного навчання (AML).

Атаки на системи AI/ML можна поділити на кілька категорій залежно від цілей зловмисника, методів реалізації та рівня впливу на модель. Серед найбільш поширених типів

атак – атаки на навчання (poisoning attacks), атаки на проникнення (evasion attacks), атаки на конфіденційність (privacy attacks) та атаки на доступність (denial-of-service attacks). Кожен із цих типів атак використовує різні механізми впливу, від підміни навчальних даних до експлуатації вразливостей у вже навчених моделях.

Актуальність дослідження обумовлена зростаючою кількістю випадків злому та маніпуляцій ШІ-системами, що може призвести до фінансових втрат, загроз безпеці та втрати довіри до технологій. Тому, метою цієї статті є дослідження загроз та ризиків пов’язаних з використанням ШІ. В статті визначаються типи атак, які можуть бути спрямовані на AI/ML-моделі, етапи життєвого циклу атаки, цілі та завдання зловмисника, а також можливості зловмисника та знання процесу навчання. Дослідження дозволяє краще зрозуміти сучасні ризики у сфері штучного інтелекту та визначити заходи для підвищення безпеки таких систем.

2. Класифікація атак на основі ШІ

Атаки на основі ШІ можна класифікувати за багатьма різними параметрами. Так, наприклад, кілька систем класифікації атак були представлені в роботах [7, 8]. Також NIST в своєму звіті щодо атак на машинне навчання ШІ представив різні типи класифікації [1, 25]. В цьому розділі будуть розглянуті деякі типи класифікацій.

2.1. Основні типи атак

Атаки на основі машинного навчання та ШІ прийнято класифікувати за такими загальними параметрами [1]:

1) метод навчання та етап процесу навчання, коли атаку встановлюють;
2) цілі та завдання зловмисника;
3) можливості зловмисника;
4) знання зловмисника про процес навчання.

На рис. 1 представлено загальну класифікацію атак, що спрямовані на AI/ML-моделі.

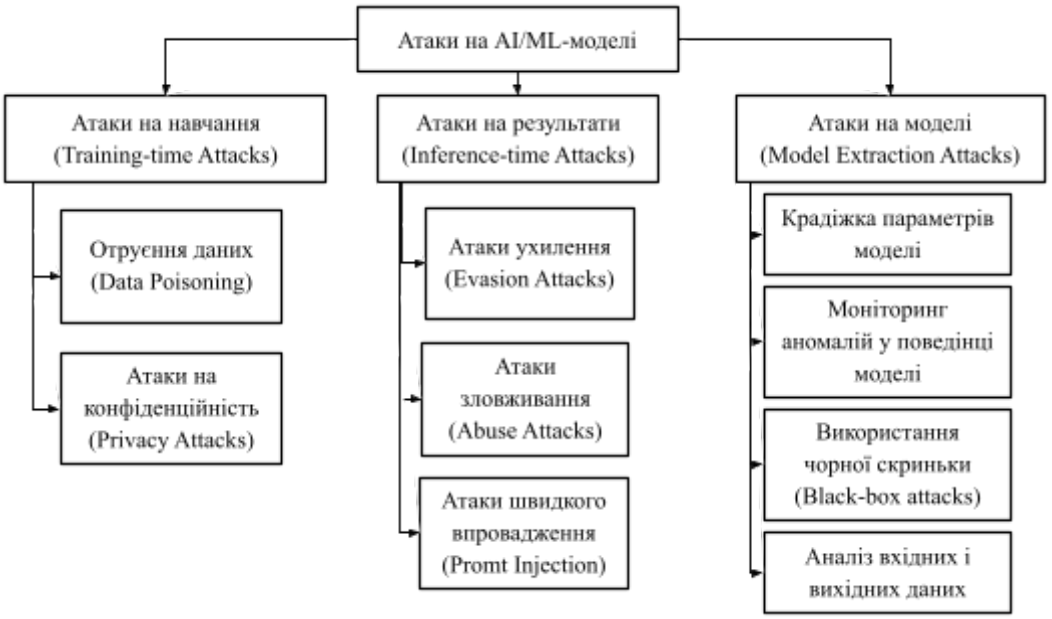

Рис. 1 – Класифікація атак на системи ШІ

Fig. 1 – Classification of attacks on AI systems

На рис. 1 зображено схему класифікації атак на системи ШІ. В свою чергу кожну з зазначених атак можна поділити на наступні основні підгрупи:

- Отруєння даних (Data Poisoning):
  - Внесення шкідливих даних у навчальний набір.
  - Підміна міток класів (Label-flipping Attack).
  - Використання прихованих тригерів (Backdoor Attack).
  - Маніпуляція вагами нейромережі.
- Атаки на конфіденційність (Privacy Attacks):
  - Витяг даних з навчального набору.
  - Інверсія моделі (Model Inversion Attack).
  - Атака на членство (Membership Inference Attack).
  - Злам параметрів моделі.
- Атаки ухилення (Evasion Attacks):
  - Маніпуляція вхідними даними.
  - Атака на зображення/текст/аудіо.
  - Обхід антивірусних-систем.
  - Зміна параметрів розпізнавання.
- Атаки зловживання (Abuse Attacks):
  - Використання генеративного ШІ для створення фальшивого контенту.
  - Deepfake (відео, аудіо, зображення).
  - Атаки соціальної інженерії.
  - Створення шкідливого коду.
- Атаки швидкого впровадження (Prompt Injection):
  - Вплив на текстові моделі (ChatGPT, Bard тощо).
  - Нав’язування небажаних відповідей.
  - Обхід обмежень моделі.
  - Генерація фейкової інформації.

2.2. Класифікація атак за метою зловмисника

Цілі зловмисника класифікуються за трьома критеріями відповідно до трьох основних типів порушень безпеки [1], які розглядаються під час аналізу безпеки системи: порушення доступності, порушення цілісності та компрометація конфіденційності даних. Відповідно, успіх зловмисника вказує на досягнення однієї або кількох із цих цілей.

Атака на доступність – це невибіркова атака на машинне навчання (ML), під час якої зловмисник намагається порушити продуктивність моделі під час розгортання. Атаки на доступність можуть бути влаштовані через отруєння даних, коли зловмисник контролює частину навчального набору або шляхом отруєння моделі, коли зловмисник контролює параметри моделі.

Атака на цілісність спрямована на цілісність вихідних даних моделі ML, що призводить до неправильних прогнозів, які виконує модель ML. Зловмисник може спричинити порушення цілісності, здійснивши атаку ухилення під час розгортання або атаку отруєння під час навчання. Атаки ухилення вимагають модифікації тестових зразків для створення змагальних прикладів, які неправильно класифікуються моделлю до іншого класу, залишаючись прихованими та непомітними для людей. Приклади таких атак можна знайти в роботах [12, 13].

При атаках спрямованих на порушення конфіденційності, зловмисники можуть бути зацікавлені в отриманні інформації про навчальні дані або про модель ML (що призводить до атак щодо конфіденційності даних та моделі відповідно). Зловмисник може мати різні цілі для

компрометації конфіденційності навчальних даних, наприклад, зміна даних (виведення вмісту або особливостей навчальних даних), викрадення даних [14, 15] (можливість витягувати навчальні дані з генеративних моделей) і викрадення властивостей щодо розподілу навчальних даних [16].

2.3. Класифікація атак за знанням зловмисника

Ще одним критерієм для класифікації атак є те, наскільки зловмисник має знання про систему машинного навчання. Існує три основних типи атак за цим критерієм [1]: біла скринька, чорна скринька та сіра скринька, див. рис. 2.

● Атаки білої скриньки. Вони припускають, що зловмисник працює з повним знанням системи машинного навчання, включаючи дані навчання, архітектуру моделі та додаткові параметри моделі. Хоча ці атаки діють на основі дуже сильних припущень, головною причиною їх аналізу є перевірка вразливості системи від найгірших зловмисників і оцінка потенційних засобів пом’якшення.

● Атаки чорної скриньки. Ці атаки передбачають мінімальні знання про систему ML. Зловмисник може отримати доступ для запитів до моделі, але він не має іншої інформації про те, як модель навчена. Ці атаки є найбільш практичними, оскільки вони припускають, що зловмисник не знає системи ШІ та використовує системні інтерфейси, що легко доступні для звичайного використання.

● Атаки сірої скриньки. Існує цілий ряд атак сірої скриньки, які фіксують суперечливі знання між атаками чорної скриньки та білої скриньки. В роботі [17] представлено структуру для класифікації атак сірого ящика. Зловмисник може знати архітектуру моделі, але не знати її параметри, або зловмисник може знати модель та її параметри, але не знати навчальні дані. Інші поширені припущення для атак сірого ящика полягають у тому, що зловмисник має доступ до даних, розподілених ідентично навчальним даним, і знає представлення функції. Останнє припущення важливе в додатках, де вилучення функцій використовується перед навчанням моделі ML, таких як кібербезпека, фінанси та охорона здоров’я.

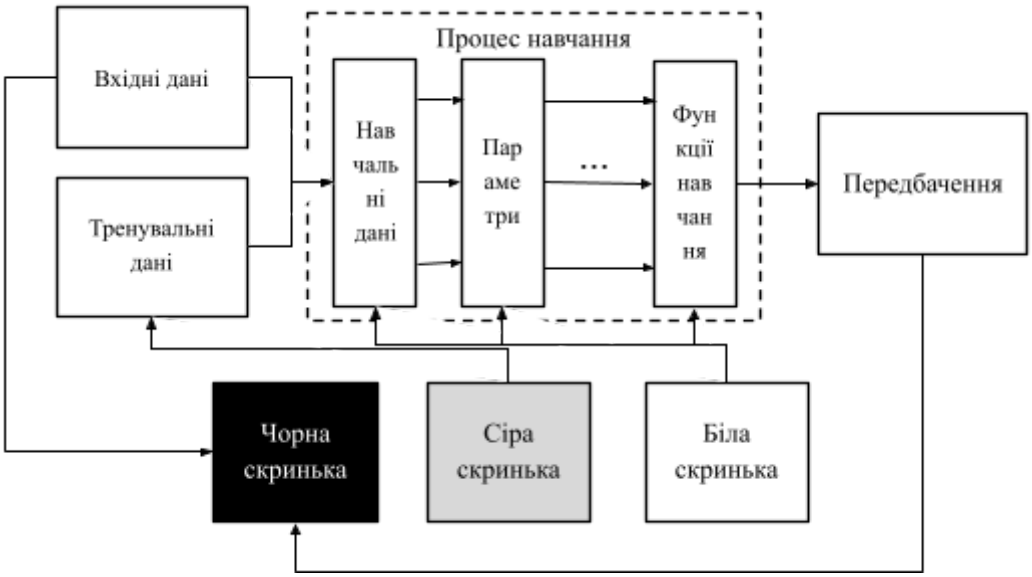

Рис. 2 – Схема ступеню обізнаності зловмисника
Fig. 2 – Scheme of the degree of awareness of the attacker

Тобто, загально кажучи, з точки зору інформації, якщо зловмисник має повні знання про модель, такі як параметри, функції та навчальні дані, ми говоримо про атаку білого ящика. І навпаки, якщо зловмисник не має жодних знань про внутрішню роботу моделі та має лише доступ до її прогнозів, ми називаємо це атакою чорної скриньки. Все, що знаходиться між цими двома, потрапляє в категорію сірої скриньки [22]. Схематично це зображено на рис. 2.

На практиці зловмисник часто починає з точки зору чорної скриньки та намагається підвищити свої знання, наприклад, виконуючи логічні висновки або оракул-атаки, коли зловмисник запитує модель, щоб отримати підказки про внутрішні елементи моделі або дані навчання. Часто конфіденційну інформацію про цільову модель можна отримати більш традиційними засобами, такими як розвідка з відкритим кодом (OSINT), соціальна інженерія, кібершпигунство тощо.

3. Атака отруєння даних

Атаки на етапі навчання ML називаються атаками отруєння [1, 9]. Під час атаки отруєння даних [5, 9] зловмисник контролює підмножину навчальних даних, вставляючи або змінюючи навчальні зразки. У атаці отруєння моделі [10] зловмисник контролює модель та її параметри. Атаки з отруєнням даних можуть застосовуватися до всіх парадигм навчання, тоді як атаки з отруєнням моделі є найбільш поширеними у федеративному навчанні [11], де клієнти надсилають локальні оновлення моделі на сервер, що обробляє вхідні дані, і в атаках на ланцюг поставок, де шкідливий код може бути доданий до моделі постачальниками технології моделі. Під федеративним навчанням тут мається на увазі метод машинного навчання, орієнтований на умови, в яких кілька суб'єктів (часто званих клієнтами) спільно навчають модель, при цьому дані, які використовуються для навчання, розподіляються децентралізовано. Це відрізняє його від машинного навчання, в якому дані зберігаються централізовано.

Перші атаки отруєння, виявлені в додатках кібербезпеки, були атаками на доступність проти генерації профілів хробака та класифікаторів спаму, які невибірково впливають на всю модель машинного навчання та, по суті, спричиняють атаку типу «відмова в обслуговуванні» для користувачів системи ШІ.

Атаки отруєння вважаються одними з найнебезпечніших серед атак на ШІ та можуть спричинити або порушення доступності, або порушення цілісності. Зокрема, атаки з порушенням доступності спричиняють деградацію моделі машинного навчання на всіх етапах, тоді як цільові та бекдорні атаки з отруєнням є більш прихованими та викликають порушення цілісності на невеликому наборі даних. Атаки отруєння використовують широкий спектр конкурентних можливостей, таких як отруєння даних, отруєння моделі, контроль міток, контроль вихідного коду та контроль тестових даних, що призводить до кількох підкатегорій атак отруєння. За моделлю загрози вони можуть використовуватись як у сценаріях білої скриньки, так і чорної скриньки [18], що були розглянуті в розділі 1.3 цієї статті.

Серед методів запобігання атакам отруєння даних виділяють два найефективніші [1]:

● Очищення навчальних даних. Ці методи використовують той факт, що отруєні набори зазвичай відрізняються від звичайних навчальних наборів, які не контролюються зловмисниками. Таким чином, методи очищення даних призначені для очищення навчального набору та видалення отруєних наборів перед виконанням навчання машинного навчання.

● Надійне навчання. Альтернативним підходом до пом'якшення атак з порушенням доступності є модифікація алгоритму навчання ML і проведення надійного навчання замість звичайного навчання. У кількох статтях визначено методи надійної оптимізації, такі як використання функції скорочених втрат [20] або випадкове згладжування для додавання шуму під час навчання [21].

4. Атака ухилення

Evasion Attacks (атаки ухилення) – це тип кібератак, при яких зловмисники модифікують вхідні дані, щоб обійти систему машинного навчання та спричинити неправильну класифікацію або прийняття хибних рішень. Такі атаки зазвичай відбуваються на етапі використання моделі, коли вона вже навчена та розгорнута. Зловмисники вносять незначні, часто непомітні для людини зміни у вхідні дані, які, однак, суттєво впливають на результат роботи моделі.

Серед типів атак ухилення за класифікацією NIST AI 100-2e2025 [25] можна виділити наступні:

1. Атаки з використанням градієнта (Gradient-based attacks): Зловмисники використовують інформацію про градієнти функції втрат моделі для визначення найефективніших змін вхідних даних, які призведуть до помилкової класифікації [22].
2. Атаки на основі балів (Score-based attacks): Атакуючі отримують доступ до оцінок впевненості моделі (confidence scores) і використовують методи оптимізації для створення підроблених прикладів, які модель класифікує неправильно.
3. Атаки на основі рішень (Decision-based attacks): Зловмисники мають доступ лише до кінцевих рішень моделі (наприклад, міток класів) і застосовують методи оптимізації для створення підроблених прикладів, які змушують модель робити помилки.
4. Атаки переміщення (Transfer attacks) [23]: Атакуючі тренують замінну модель, генерують на ній підроблені приклади та переносять ці атаки на цільову модель, використовуючи схожість між моделями.

Атаки ухилення становлять серйозну загрозу для систем кібербезпеки. Зловмисники можуть змінювати характеристики шкідливого програмного забезпечення, щоб обійти антивірусні системи, або модифікувати мережевий трафік для уникнення виявлення системами вторгнень.

На даний момент основними методами захисту [1] від атак ухилення є:

1. Змагальне навчання (Adversarial training): Включення підроблених прикладів у процес навчання моделі для підвищення її стійкості до атак.
2. Використання ансамблів моделей (Ensemble methods) [24]: Комбінування кількох моделей для зменшення ймовірності успішної атаки на всі моделі одночасно.
3. Моніторинг та оновлення (Continuous monitoring and updating): Регулярне оновлення моделей та систем виявлення для адаптації до нових стратегій атак та покращення стійкості.

Тим не менш, ці методи мають різні обмеження, такі як знижена точність для змагального навчання та випадкового згладжування, а також обчислювальна складність для формальних методів. Тому потрібно завжди шукати компроміс між надійністю та точністю. Розуміння та впровадження цих методів захисту є критично важливим для забезпечення безпеки та надійності систем машинного навчання в умовах зростаючих загроз атак ухилення.

5. Атака на конфіденційність

Атаки на конфіденційність (Privacy Attacks) – це тип кібератак, спрямованих на отримання конфіденційної інформації з моделей штучного інтелекту (ШІ), їхніх навчальних даних або вихідних даних. Ці атаки можуть бути використані для крадіжки персональних даних, компрометації комерційної інформації або зламу моделей машинного навчання.

Нижче розглянемо основні типи атак на конфіденційність:

1. Атаки з відновленням даних (Data Reconstruction Attacks): Зловмисники намагаються відтворити вихідні дані, використовуючи доступ до моделі ШІ або її вихідних

результатів. Це може статися, коли модель видає занадто детальні відповіді або має вразливості, що дозволяють відновити частини навчальних даних.

2. Атаки з інверсією моделі (Model Inversion Attacks): У цьому випадку зловмисники використовують вихідні дані моделі для відтворення вхідних даних або характеристик, що використовувалися під час навчання. Це може розкрити конфіденційну інформацію про сторін, представлених у навчальних даних.
3. Атаки з визначення принадності (Membership Inference Attacks): Зловмисники намагаються визначити, чи були конкретні записи включені в навчальний набір даних моделі. Це може бути використано для розкриття участі особи в певних заходах або її належності до певних груп.
4. Атаки на основі метаданих (Metadata-based Attacks): Навіть, якщо самі дані залишаються захищеними, зловмисники можуть використовувати метадані (наприклад, час доступу, розмір файлів) для отримання конфіденційної інформації або встановлення патернів, які можуть бути використані в подальших атаках.
5. Атаки через бічні канали (Side-channel Attacks): Зловмисники можуть аналізувати поведінку системи ШІ, наприклад, час відгуку або споживання енергії, щоб отримати інформацію про внутрішні процеси або дані моделі.

Для захисту від таких атак рекомендується:

- Підвищення анонімності та агрегування даних. Використання методів, які зменшують ризик ідентифікації індивідуальних записів у навчальних даних.
- Федеративне навчання (Federated Learning) – навчання моделей на локальних пристроях без передачі даних на сервер.
- Диференційна приватність. Додавання контрольованого шуму до даних або результатів моделі, щоб запобігти відновленню вихідних даних без значного впливу на точність моделі.
- Обмеження доступу та моніторинг: Контроль доступу до моделей та даних, а також постійний моніторинг використання для виявлення підозрілої активності.
- Оцінка вразливостей: Регулярне тестування моделей на стійкість до атак на конфіденційність та впровадження відповідних заходів захисту.

Атаки на конфіденційність є серйозною загрозою для систем ШІ, в тому числі і в сфері кібербезпеки. Тому захист конфіденційності в системах ШІ є критично важливим для збереження довіри користувачів та дотримання нормативних вимог. Зловмисники використовують різні методи, щоб отримати доступ до конфіденційних даних або параметрів моделей. Захист таких систем вимагає комплексного підходу, включаючи сучасні криптографічні методи, моніторинг активності та підвищення обізнаності користувачів про ризики.

6. Атака на зловживання

Ще одним типом атак на системи ШІ є атаки зловживання (abuse attacks) [25]. Ці атаки спрямовані на зловживання або маніпуляцію системами штучного інтелекту (ШІ) з метою отримання небажаних або шкідливих результатів. Ці атаки використовують вразливості в структурі або реалізації моделей ШІ, щоб змусити систему поводитися неналежним чином.

Приклади атак зловживання:

1. Використання упередженості моделі (Bias Exploitation): Атака, при якій зловмисник використовує існуючі упередження або слабкі місця в моделі ШІ, щоб отримати певні результати або посилити дискримінаційні тенденції.
2. Зловживання функціональністю (Functionality Misuse): Маніпуляція системою ШІ для

виконання дій, які не були передбачені розробниками, наприклад, використання чат-бота для генерації небажаного контенту або спаму.

3. Атаки на основі підказок (Prompt Injection Attacks): Введення спеціально створених запитів або команд, які змушують модель ШІ генерувати небажаний або шкідливий контент.

Для упередження таким атакам NIST рекомендує наступні заходи безпеки:

- Удосконалення алгоритмів виявлення аномалій – розвиток механізмів, які можуть розпізнавати підозрілі взаємодії зі ШІ.
- Жорсткіші політики перевірки даних – аналіз вхідних даних для виявлення можливих маніпуляцій.
- Підвищення прозорості ШІ-систем – покращення документації процесів прийняття рішень у моделях.
- Механізми протидії зловмисного проникнення – наприклад, введення додаткових рівнів перевірки в моделях безпеки.
- Розробка стандартів етичного використання ШІ – активне регулювання та контроль за впровадженням таких технологій.

Таким чином, атаки зловживання є серйозною загрозою для систем ШІ, оскільки вони дозволяють зловмисникам використовувати їх у несподіваний спосіб. Використання ШІ для автоматизації шахрайства, маніпуляції суспільною думкою або обходу обмежень створює нові виклики для кібербезпеки. Запобігання таким атакам вимагає комплексного підходу, включаючи вдосконалення алгоритмів безпеки, розробку політик відповідального використання ШІ та постійний моніторинг загроз.

7. Загальний підсумок щодо атак на системи ШІ

В розділах 2-5 цієї статті було розглянуто 4 найвпливовіші типи атак на системи ML/AI. Підсумки аналізу розглянутих атак наведено в таблиці 1 нижче.

Таблиця 1 – Порівняльна характеристика атак на AI/ML-системи
Table 1 – Comparative characteristics of attacks on AI/ML systems

Тип атаки	Мета атаки	Фаза атаки	Методи	Наслідки
Poisoning Attack (атака отруєння даних)	Вплив на якість навчання	Навчання моделі	Додавання шкідливих даних до навчального набору	Погіршення якості прийнятих рішень, хибне спрацьовування, зниження безпеки для подальших атак
Evasion Attack (атака ухилення)	Обхід механізмів безпеки моделі	Виконання моделі	Маніпуляція вихідними даними, генерація спеціальних зловмисних даних	Обхід захисних механізмів, зниження точності моделі
Privacy Attack (атака на конфіденційність)	Викрадення даних, на яких навчалася модель	Виконання моделі	Аналіз відповідей моделі, відновлення даних	Витік конфіденційних даних
Abuse Attacks (атаки зловживання)	Використання ШІ для створення шкідливих даних	Після розгортання моделі	Генеративні моделі для атак, маніпуляції	Маніпуляція інформацією, автоматизація шахрайства

Таким чином, було розглянуто:

Атаки отруєння даних – проводяться на етапі навчання та можуть довгостроково впливати на точність моделі, змушуючи її робити неправильні висновки.

Атаки ухилення – це атаки на фазі виконання, коли зловмисники намагаються обдурити модель, вводячи в неї спеціально створені дані.

Атаки на конфіденційність – спрямовані на витяг конфіденційних даних із моделі, що ставить під загрозу безпеку особистої інформації користувачів.

Атаки зловживання – пов'язані з неправомірним використанням можливостей ШІ, наприклад, для створення фейкових відео, зловмисного коду чи маніпуляцій у соціальних мережах.

8. Висновки

1. ШІ є гарним інструментом для автоматизації процесів виявлення та реагування на атаки та загрози, і значно підвищує ефективність захисту систем та компаній. Використання алгоритмів машинного навчання сприяє швидкому аналізу великих обсягів даних та ідентифікації аномалій у поведінці користувачів і систем.

2. Проте ШІ не тільки забезпечує ефективний захист, а й створює нові загрози через його можливе використання зловмисниками. Саме тому питання щодо виявлення атак та протидії їм є дуже важливим питанням в сучасному кіберпросторі. Тому у цій статті було розглянуто та проведено всебічний аналіз атак на сучасні моделі ML/AI.

3. Надійність системи ШІ залежить від усіх атрибутів, які її характеризують. Наприклад, система штучного інтелекту, яка є точною, але легко сприйнятливою до агресивних дій, навряд чи заслуговує на довіру. Так само навряд чи можна довіряти системі ШІ, яка дає шкідливо упереджені або несправедливі результати, навіть, якщо вона надійна. Існують також компроміси між прозорістю та конкурентоспроможністю. На жаль, неможливо одночасно максимізувати продуктивність системи ШІ щодо цих атрибутів. Наприклад, системи ШІ, оптимізовані тільки для точності, як правило, мають недостатню ефективність з точки зору конкурентної надійності та справедливості. І навпаки, система ШІ, оптимізована для змагальності, може продемонструвати нижчу точність і погіршити результати надійності.

4. Атаки отруєння даних є найнебезпечнішим видом атак на основі ШІ у довгостроковій перспективі, оскільки вони впливають на саму модель і можуть лишатися непомітними протягом тривалого часу.

5. Атаки на конфіденційність та атаки зловживання особливо небезпечні через можливість витоку конфіденційних даних та створення шкідливого контенту.

6. В результаті дослідження було виявлено загальні наслідки атак на основі ШІ:

- Підрив довіри до ШІ – постійні атаки та маніпуляції можуть зробити ШІ менш надійним інструментом для ухвалення рішень.
- Витік конфіденційної інформації – атаки на конфіденційність призводять до масштабних втрат персональних та корпоративних даних.
- Обхід засобів захисту – ухилення та отруєння моделей ставлять під загрозу сучасні системи кібербезпеки, знижуючи їхню ефективність.
- Масштабованість атак – зловмисники можуть використовувати ШІ для автоматизації та прискорення атак, що збільшує їхній вплив.
- Ризики на державному рівні – атаки на основі ШІ можуть загрожувати національній безпеці, економіці та критичній інфраструктурі.

7. Тому наразі основними шляхами захисту від атак на основі ШІ є:

- Розробка стійких ШІ-моделей, які менш вразливі до отруєння або ухилення.

- Впровадження механізмів виявлення атак, таких як моніторинг змін у навчальних даних та алгоритмах.
- Підвищення прозорості ШІ – створення пояснюваних моделей, які можна перевірити на наявність маніпуляцій.
- Посилення регулювання та стандартів – держави та організації повинні встановлювати правила щодо використання ШІ.

8. Кожен розглянутий тип атак на системи ШІ має різні механізми впливу, але всі вони можуть суттєво знизити ефективність та безпеку моделей машинного навчання. Захист від таких атак вимагає комплексного підходу. Окрім визначених вище методів протидії атакам, важливим залишається людський фактор – навчання спеціалістів та користувачів, адже багато атак базуються саме на соціальній інженерії.

9. У більшості випадків організаціям доведеться прийняти компроміс між бажаними властивостями та вирішити, яким із них віддати пріоритет залежно від системи ШІ, варіанту використання та потенційно багатьох інших міркувань щодо економічних, екологічних, соціальних, культурних, політичних і глобальних наслідків технології ШІ.

10. Важливо зазначити, що з розвитком технологій ШІ з’являються нові типи атак, тому постійний моніторинг та оновлення знань у цій сфері є необхідними для забезпечення кібербезпеки.

11. Підсумовуючи важливо зазначити, що безпечне використання ШІ у кібербезпеці є балансом між технологічними інноваціями та загрозами, що виникають внаслідок їхнього розвитку. Розробка адаптивних методів захисту та вдосконалення моделей машинного навчання допоможуть зменшити ризики, пов’язані з атаками, та забезпечити надійний рівень кіберзахисту в майбутньому.

Конфлікт інтересів

Автори повідомляють про відсутність конфлікту інтересів.

References

1. Vassilev A., Oprea A., Fordyce A., Anderson H. (2024) Adversarial Machine Learning: A Taxonomy and Terminology of Attacks and Mitigations. (National Institute of Standards and Technology, Gaithersburg, MD) *NIST Artificial Intelligence (AI) Report, NIST Trustworthy and Responsible AI NIST AI 100-2e2023*. – Access mode: https://doi.org/10.6028/NIST.AI.100-2e2023
2. Booth H., Souppaya M., Vassilev A., Ogata M., Stanley M., Scarfone K. (2024) Secure Development Practices for Generative AI and Dual-Use Foundation AI Models: An SSDF Community Profile. (National Institute of Standards and Technology, Gaithersburg, MD), *NIST Special Publication (SP) NIST SP 800-218A*. – Access mode: https://doi.org/10.6028/NIST.SP.800-218A
3. Oprea A., Singhal A. and Vassilev A.. (2022) Poisoning Attacks Against Machine Learning: Can Machine Learning Be Trustworthy?, *in Computer*, vol. 55, no. 11, pp. 94-99, Nov. URL: https://doi.org/10.1109/MC.2022.3190787
4. Hui Wei, Hao Tang, Xuemei Jia, Zhixiang Wang, Hanxun Yu, Zhubo Li, Shin’ichi Satoh, Luc Van Gool, Zheng Wang. (2024) Physical Adversarial Attack Meets Computer Vision: A Decade Survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol.46, no.12, pp.9797-9817, URL: https://doi.org/10.48550/arXiv.2209.15179
5. Koundinya A., Patil S., Chandu B. (2024) Data Poisoning Attacks in Cognitive Computing, *IEEE 9th International Conference for Convergence in Technology (I2CT)*, pp.1-4, https://doi.org/10.1109/I2CT61223.2024.10544345
6. National Institute of Standards and Technology. (2023) *Artificial Intelligence Risk Management Framework. (AI RMF 1.0)*. – Access mode: https://doi.org/10.6028/NIST.AI.100-1

7. Biggio B., Roli F. (2018) Wild patterns: Ten years after the rise of adversarial machine learning. *Pattern Recognition,* 84:317–331 https://doi.org/10.48550/arXiv.1712.03141
8. Octavian Suciu, Radu Marginean, Yigitcan Kaya, Hal Daume III, and Tudor Dumitras. (2018) When does machine learning FAIL? generalized transferability for evasion and poisoning attacks. In 27th USENIX Security Symposium (USENIX Security 18), pp. 1299–1316, https://www.usenix.org/conference/usenixsecurity18/presentation/suciu
9. Biggio B., Nelson B., Laskov P. (2012) Poisoning attacks against support vector machines. *In Proceedings of the 29th International Conference on International Conference on Machine Learning, ICML*, URL: https://doi.org/10.48550/arXiv.1206.6389
10. Yingqi Liu, Shiqing Ma, Yousra Aafer, Wen-Chuan Lee, Juan Zhai, Weihang Wang, and Xiangyu Zhang. (2018) Trojaning attack on neural networks. *In NDSS*. The Internet Society, URL: https://dx.doi.org/10.14722/ndss.2018.23291
11. Kairouz, Peter; McMahan, H. Brendan; Avent, Brendan; Bellet, Aurélien; Bennis, Mehdi; Bhagoji, Arjun Nitin; Bonawitz, Kallista; Charles, Zachary; Cormode, Graham (2021). Advances and Open Problems in Federated Learning. *Foundations and Trends in Machine Learning 14 (1–2)*: https://doi.org/10.1561/2200000083. ISSN 1935-8237
12. Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian Goodfellow, Rob Fergus. (2014) Intriguing properties of neural networks. *In International Conference on Learning Representations*, URL: https://doi.org/10.48550/arXiv.1312.6199
13. Ian Goodfellow, Jonathon Shlens, Christian Szegedy. (2015) Explaining and harnessing adversarial examples. *In International Conference on Learning Representations*, URL: https://doi.org/10.48550/arXiv.1412.6572
14. Nicholas Carlini, Chang Liu, Ulfar Erlingsson, Jernej Kos, Dawn Song. (2019) The Secret Sharer: Evaluating and testing unintended memorization in neural networks. *In USENIX Security Symposium, USENIX 19*, pages 267–284. – URL: https://arxiv. org/abs/1802.08232
15. Nicholas Carlini, Florian Tramer, Eric Wallace, Matthew Jagielski, Ariel Herbert - Voss, Katherine Lee, Adam Roberts, Tom Brown, Dawn Song, Ulfar Erlingsson, Alina Oprea, Colin Raffel. (2021) Extracting training data from large language models. *In 30th USENIX Security Symposium (USENIX Security 21)*, pages 2633–2650. USENIX Association, URL: https://doi.org/10.48550/arXiv.2012.07805
16. Karan Ganju, Qi Wang, Wei Yang, Carl A. Gunter, and Nikita Borisov. (2018) Property inference attacks on fully connected neural networks using permutation invariant representations. *In Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security, CCS '18*, page 619–633, New York, NY, USA. Association for Computing Machinery. URL: https://doi.org/10.1145/3243734.3243834
17. Octavian Suciu, Radu Marginean, Yigitcan Kaya, Hal Daume III, Tudor Dumitras.(2018) When does machine learning FAIL? generalized transferability for evasion and poisoning attacks. *In 27th USENIX Security Symposium (USENIX Security 18)*, pages 1299–1316. URL: https://www.usenix.org/conference/usenixsecurity18/presentation/suciu
18. Battista Biggio, Blaine Nelson, Pavel Laskov. (2012) Poisoning attacks against support vector machines. *In Proceedings of the 29th International Conference on International Conference on Machine Learning, ICML*, URL: https://doi.org/10.48550/arXiv.1206.6389
19. Nihad Hassan. (2024) What is data poisoning (AI poisoning) and how does it work? *Search Enterprise AI, TechTarget*. – URL: https://www.techtarget.com/searchenterpriseai/definition/data-poisoning-AI-poisoning
20. Ilias Diakonikolas, Gautam Kamath, Daniel Kane, Jerry Li, Jacob Steinhardt, and Alistair Stewart. (2019) Sever: A robust meta-algorithm for stochastic optimization. *In International Conference on Machine Learning*, pages 1596–1606. PMLR, URL: https://doi.org/10.48550/arXiv.1803.02815
21. Elan Rosenfeld, Ezra Winston, Pradeep Ravikumar, and Zico Kolter. (2020) Certifed robustness to label-flipping attacks via randomized smoothing. *In International Conference on Machine Learning*, pages 8230–8241. PMLR, URL: https://doi.org/10.48550/arXiv.2002.03018
22. The Tactics & Techniques of Adversarial Machine Learning. HiddenLayer/ (2022). – URL: https://hiddenlayer.com/innovation-hub/the-tactics-and-techniques-of-adversarial-ml
23. Chi Zhang, Zifan Wang, Ravi Mangal, Matt Fredrikson, Limin Jia, Corina Pasareanu. (2023) Transfer Attacks and Defenses for Large Language Models on Coding Tasks. – URL: https://doi.org/10.48550/arXiv.2311.13445

24. D. Li and Q. Li,(2023) Adversarial Deep Ensemble: Evasion Attacks and Defenses for Malware Detection, in *IEEE Transactions on Information Forensics and Security*. – URL: https://doi.org/10.48550/arXiv.2006.16545
25. Vassilev A, Oprea A, Fordyce A, Anderson H (2025) Adversarial Machine Learning: A Taxonomy and Terminology of Attacks and Mitigations. *(National Institute of Standards and Technology, Gaithersburg, MD) NIST Artificial Intelligence (AI) Report, NIST Trustworthy and Responsible AI NIST AI 100-2e2025*. – URL: https://doi.org/10.6028/NIST.AI.100-2e2025

RESEARCH AND CLASSIFICATION OF THE MAIN TYPES OF ATTACKS ON ARTIFICIAL INTELLIGENCE SYSTEMS IN CYBERSECURITY

Vladyslav Vilihura[1], PhD, Senior Lecture; e-mail: v.v.vilihura@karazin.ua;
ORCID: https://orcid.org/0000-0002-1137-2382
Yelyzaveta Ostrianska[1], junior researcher; e-mail: antelizza@gmail.com;
ORCID: https://orcid.org/0000-0003-1412-8470
[1] *V. N. Karazin Kharkiv National University, Ukraine*
Manuscript was received November 1, 2024; Received after review December 2, 2024;
Accepted December 23, 2024

Abstract. The modern development of artificial intelligence (AI) and machine learning (ML) opens up new opportunities in the field of cybersecurity, but at the same time creates serious challenges in the form of intelligent cyberattacks. The study is devoted to the analysis and classification of ways to use AI for malicious purposes and the study of effective methods to counter such threats. In particular, the article covers the main types of attacks using ML technologies, which demonstrate how attackers can manipulate machine learning algorithms, undermine trust in data, and bypass protection systems. Special attention is paid to the mechanisms of data poisoning attacks, as they are considered the most influential in machine learning, which involve introducing malicious data into the process of training models, which leads to distortion of results and undermines the effectiveness of security algorithms. Privacy attacks are analyzed as a way to obtain confidential information from ML models, which can be used to steal user data. Abuse attacks demonstrate how attackers can use AI tools to automate attacks, scale phishing campaigns, and analyze vulnerabilities in defense systems. The relevance of the study is due to the fact that traditional approaches to cyber defense are no longer able to effectively counter threats that adapt and evolve due to machine learning. The article emphasizes the critical importance of researching defense methods, in particular, building reliable machine learning systems that have built-in mechanisms for detecting anomalies and adapting to new threats. One of the key approaches is federated learning, which allows training models without centralized data storage, reducing the risk of information leakage. The development of deep learning in the field of cyber defense is also considered, which allows analyzing behavioral patterns of threats in real time. The combination of technological measures with human control remains an important aspect, since, despite the power of AI tools, the human factor remains key in the process of ensuring cybersecurity. Thus, the article demonstrates the balance between the opportunities and threats of AI in the field of cybersecurity, emphasizing the need for further research in the direction of resilient ML models that can effectively resist attacks. Without proper regulation and control, AI can become not only a defender, but also a tool for attackers, which requires the development of new security strategies and international regulation in the field of cybersecurity.

Keywords: *artificial intelligence, cyberattacks, machine learning, cybersecurity, federated learning*

Conflicts of Interest: the authors declare no conflict of interest.

ISSN 2519-2310 (Online)

КОМП'ЮТЕРНІ НАУКИ ТА КІБЕРБЕЗПЕКА, 2024, № 2 (26)

COMPUTER SCIENCE AND CYBERSECURITY, 2024, Iss. 2 (26)

DOI: https://doi.org/10.26565/2519-2310-2024-2-02
УДК 004.032.26

PROBLEMS OF THE NEURAL NETWORKS OUTPUT DATA QUALITY ASSESSMENT

Yurii Halaichuk[1], PhD student, department of theoretical and applied systems engineering, e-mail: yurii.halaichuk@student.karazin.ua, ORCID: https://orcid.org/0009-0004-1048-9425
Maryna Miroshnyk[1], Doctor of Technology, professor, professor of the department of theoretical and applied systems engineering, e-mail: m.miroshnyk@karazin.ua, ORCID: https://orcid.org/0000000222312529
[1]*V. N. Karazin Kharkiv National University, Svobody Square, 4, Kharkiv, 61022, Ukraine*
Manuscript was received October 20, 2024; Received after review November 23, 2024; Accepted December 13, 2024

Abstract: Today, artificial intelligence, particularly neural networks, is increasingly being used in software in a variety of industries, from mission-critical applications such as healthcare and the military to commerce and entertainment. One of the main stages of development and implementation of such software is the stage of quality control. To prevent fatal errors and to survive in a highly competitive environment, the software needs proper testing taking into account the peculiarities inherent in the data obtained as a result of the neural network. This article presents the relevance of using artificial intelligence systems in general and neural networks in particular and analyzes the main challenges that arise when assessing the quality of such networks. The author compares the properties of the output data of the artificial intelligence systems of the previous generation and the latest neural networks, highlights the key differences of the latter, such as the potential infinity of the input data sets and their relative unpredictability, the dependence of the results on the network training stage, and the subjective nature of the evaluation of such results. Based on the analysis, the author formulates a set of problems that can be solved using mathematical algorithms and methods. The main part of an article contains a general overview of existing solutions, with an emphasis on such algorithms and methods as calculating accuracy and loss, finding the F-score, interpretation methods and imitation modeling. As a result of the research, the author comes to the conclusion that, despite a sufficient number of existing solutions that can be used to solve the highlighted problems, they still have to be improved to increase the accuracy of neural network evaluation, as one hundred percent accuracy in evaluating data obtained as a result of the operation of neural networks has not yet been achieved.

Keywords: *artificial intelligence, expert systems, F-score, loss function, neural networks, quality assessment*

In cites: Halaichuk Y., Miroshnyk M. (2024). Problems of the neural networks output data quality assessment. *Computer Science and Cybersecurity.* 2(26): 19–24. https://doi.org/10.26565/2519-2310-2024-2-02

1. Introduction

According to Statista.com, the global artificial intelligence (AI) market is expected to reach nearly $126 billion by 2025, up from just $10.1 billion in 2016. There are various information systems and software that use neural networks algorithms to process data and generate results. Examples of such software include services for finding similar images, navigation devices for finding the most optimal route, dating sites, text generators that use machine learning, computer games, and many others.

Evaluation of neural networks is a very relevant and timely topic given the growing integration of AI into various aspects of our lives. Neural networks are becoming an important tool in decision-making, creative endeavors and problem-solving. As society increasingly relies on the results of neural networks, understanding how to evaluate their performance becomes critical. Currently, quite effective methods, models, and techniques for assessing software quality are widely used, including such software quality indicators as functionality, reliability, usability, efficiency, mobility, interactivity, etc. However, neural networks have peculiarities that should be considered separately.

2. Setting of the problem and the aim of the article

One of the peculiarities of neural networks is that unlike other algorithms, where the decision-making logic is often directly coded and is transparent, as in the case of expert and scoring systems, neural networks operate as complex, interconnected layers of nodes whose internal workings are hidden from data assessors.

One of the most important challenges in assessing neural networks is their ability to generate an infinite variety of outputs. Unlike traditional algorithms that produce a fixed set of numerical or categorical results, neural networks can generate diverse outputs such as text, images, music, or even complex behaviors in games. For instance, large language models (LLMs) can produce countless variations of a story or response to a prompt, each differing in tone, style, or content. This infinite output set complicates assessment because it is impossible to evaluate every possible result.

Another significant challenge is the variability of neural network outputs based on their training stage. Neural networks are iterative learners, meaning their performance evolves as they are exposed to more data or undergo additional training. Consequently, the same input can produce different outputs depending on when the model is evaluated. For example, a partially trained image recognition model might misclassify a cat as a dog, while a fully trained version of the same model correctly identifies the cat. This variability makes it difficult to assess the quality of a neural network consistently. The dependence on the training stage introduces uncertainty in evaluation. A model that performs well during one phase of training might degrade or behave unpredictably later due to overfitting, underfitting, or changes in the training data. This is particularly problematic for applications requiring stable outputs, such as medical diagnosis or autonomous driving, where inconsistent results could have serious consequences.

The next complex problem is the lack of clearly defined criteria for assessing the quality of creative or non-numerical outputs, such as text, images, or game behaviors. In creative tasks such as art, literature, or music, where subjectivity plays a significant role, the understanding of a single correct result becomes unclear. This subjectivity introduces a significant human factor into the assessment process. Human evaluators often disagree on what is a correct output, leading to inconsistent and biased evaluations. For instance, one tester might approve a generated image for its aesthetics, while another critiques it for lacking realism. This raises profound questions about how we define success and failure in the context of neural network-generated content and requires a shift to more nuance-oriented evaluation criteria.

To understand the complexity of testing neural networks in relation to knowledge-based systems, such as expert systems, the following comparison of their output data can be made:

Table 1. Comparison of output data properties of expert systems and neural networks

Output data properties	Expert systems	Neural networks
Potential quantity	Determined by the size of the knowledge base or is a range of numbers	Endless
Uniqueness	All possible sets of output data or possible combinations are known	Each set of output data can be unique and unpredictable
Dependence on input data	The output data depends on the input data according to a known algorithm	The output data depends on the input data according to the logic used by the algorithm at its current learning stage
Complexity of the assessment	Data accuracy can be assessed objectively	Assessment of data accuracy is partly subjective

Thus, we can identify the following key issues that arise when evaluating the performance of neural networks:

- *the potential number of output results (including results that are not a set of numbers) can be infinite;*
- *output data depend on the algorithm training - they can change with the same input data, which makes the assessment of the quality of such an algorithm dependent on its learning stage;*
- *there are no clearly defined criteria for the quality of such output data as, for example, creative text, images, and game behavior, which leads to a large share of human factors in assessment of such data.*

3. Existing solutions

For example, to test expert systems and obtain accuracy which is close to 100% it is sufficient to use the method of boundary values and the method of equivalent classes, which allow covering the entire range of possible outcomes.

However, the difficulties associated with evaluating the performance of neural networks, emphasize the need for a more systematic approach. Mathematical methods make it possible to quantify and analyze the intricacies of neural networks, offering a more objective and reproducible assessment.

3.1. The most essential approach to obtaining quality metrics of neural networks performance is the calculation of accuracy and loss function values.

Accuracy represents the proportion of correctly classified or predicted instances out of the total number of samples in the test dataset. It offers a straightforward and easily interpretable measure of how well the neural network generalizes to unseen data. A high accuracy score suggests that the model is making correct predictions for a significant portion of the input, showing confidence in its ability to perform well in real-world applications. For tasks like image classification, where the goal is

to assign a single correct label to each image, accuracy serves as a natural and intuitive benchmark. A 95% accuracy, for instance, directly translates to the model correctly identifying 95 out of every 100 images.

$A = \frac{N(correct)}{N}$, where N is the amount of test samples (1)

The loss function quantifies the discrepancy between the network's predictions and the actual ground truth labels. It provides a more granular and continuous measure of the model's performance, reflecting the degree of error in its predictions. Unlike accuracy, which focuses on discrete correctness, the loss function captures the confidence and correctness of each individual prediction. Different types of loss functions are used depending on the specific task and the neural network architecture.

The general rule of application of obtained accuracy and loss values to assess the quality of neural network output is described in Table 2:

Table 2. Relation of Accuracy / Loss values to overall model performance

	Low Loss	High Loss
High Accuracy	The model correctly predicts most instances with high confidence.	The model makes correct predictions with low confidence or incorrect predictions have large errors.
Low Accuracy	The model consistently predicts values close to the true values but does not cross a classification threshold correctly.	The model makes many incorrect predictions with significant errors.

3.2. The F1 score method provides a single, robust metric for evaluating classification models, especially in situations with imbalanced classes. By considering both the accuracy of positive predictions and the ability to identify all positive instances, it offers a more insightful assessment than accuracy alone, allowing to build more reliable and effective classification systems.

The algorithm assigns a score weight to all the output data and emphasizes the following criteria:

$$Precision = \frac{True\ Positives}{True\ Positives + False\ Positives}, \quad (2)$$

$$Recall = \frac{True\ Positives}{True\ Positives + False\ Negatives}, \quad (3)$$

where:

- *True Positives* is the number of data samples correctly predicted as "positive";
- *False Positives* is the number of data samples wrongly predicted as "positive";
- *True Negatives* is the number of data samples correctly predicted as "negative";
- *False Negatives* is the number of data samples wrongly predicted as "negative".

Then F1 score is being calculated:

$$F1\ score = 2 * \frac{Precision * Recall}{(Precision + Recall)}, \quad (4)$$

which will provide the percentage values of positive results for the investigated model. A high F1 score indicates that the model has both high precision and high recall, signifying a well-balanced classifier.

3.3. Another approach is to use model interpretation methods to explain the logic which was used during the prediction of output results. The most widely used methods are SHAP (SHapley Additive exPlanations) and LIME (Local Interpretable Model-agnostic Explanations), which are based on game theory and local surrogate models respectively and allow to conduct black-box testing of neural network algorithms.

LIME focuses on explaining individual predictions by approximating the complex black-box model with a more simple, interpretable model (like a linear model) to analyze the specific data point. It is relatively fast and easy to understand, but its explanations can be unstable depending on the perturbation and sampling methods, and it only provides a local view without guarantees of global consistency.

SHAP provides a more theoretical approach using game theory, specifically Shapley values. It aims to quantify the contribution of each feature to a specific prediction by considering all possible combinations. SHAP offers stronger theoretical guarantees and more consistent explanations, but at the cost of execution speed and simplicity.

3.4. The capabilities of imitational modeling allow testing of neural networks in a similar way it is done with static data using the combination of boundary values and equivalent classes methods. Such methods as Synthetic Data Generation, Environment Simulation, Adversarial Attack Simulation, Monte Carlo Simulation can perform testing covering many non-standard cases, however, they require the creation of new data sets for testing and are more appropriate for using the development flow and unit testing of neural networks.

The algorithms and methods described above can help to improve the quality of neural network output data evaluation and require further research to improve the accuracy of that evaluation

4. Conclusions

To solve the problems associated with evaluating neural networks, the integration of mathematical methods is a crucial component. Quantitative metrics, algorithms, and mathematical models provide a structured framework for objective assessment of the quality and performance of neural networks.

Despite a sufficient number of existing methods and algorithms that can be used to perform tasks of evaluation of the quality of neural networks, they still need further improvement, as they do not yet allow for 100% accuracy in forming an assessment of the output data of AI algorithms.

Conflicts of Interest

The authors declare no conflict of interest.

References

1. Giesecke K., Horel E., (2020) Significance Tests for Neural Networks by Enguerrand Horel. *Journal of Machine Learning Research.* July 20, 2020. 1-29. https://jmlr.csail.mit.edu/papers/volume21/19-264/
2. Saurabh Ranjan Srivastava., (2016) F1 Score Analysis of Search Engines. *Skit Research Journal*, 2(6)
3. Ponakala R., Dailey M., (2019). Testing Deep Neural Networks for Classification Tasks Through Adversarial Perturbations on Test Datasets. December, 2019. Asian Institute of Technology, School of Engineering and Technology, Thailand. https://doi.org/10.31237/osf.io/r7wcn

4. Christian Terwiesch, (2023) Would Chat GPT Get a Wharton MBA? A Prediction Based on Its Performance in the Operations Management Course, Mack Institute for Innovation Management at the Wharton School, University of Pennsylvania, 2023. URL: https://mackinstitute.wharton.upenn.edu/2023/would-chat-gpt3-get-a-wharton-mba-new-white-paper-by-christian-terwiesch/
5. ISO/IEC/IEEE 29119-1:2013. Software and Systems Engineering. Software Testing Part 1: Concepts and Definitions. Geneva: International Organization for Standardization, 2013
6. Lundberg S. M., Lee S.-I. A Unified Approach to Interpreting Model Predictions // Advances in Neural Information Processing Systems (NeurIPS), 2017, Vol. 30, с. 4765-4774. DOI: https://doi.org/10.48550/arXiv.1705.07874
7. Russell S., Norvig P. Artificial Intelligence: A Modern Approach: підручник, 4th ed, Boston: Pearson, 2020, 1152 с.
8. Zhang J. M., Harman M., Ma L., Liu Y. Machine Learning Testing: Survey, Landscapes and Horizons // IEEE Transactions on Software Engineering. 2020, Vol. 48 No. 1, с. 1-36. DOI: https://doi.org/10.1109/TSE.2019.2962027
9. Hossain E. Machine Learning Crash Course for Engineers / Eklas Hossain., 2024. - 453 с.

ПРОБЛЕМИ ОЦІНЮВАННЯ ВИХІДНИХ ДАНИХ НЕЙРОННИХ МЕРЕЖ

Юрій Галайчук[1], аспірант кафедри теоретичної та прикладної системотехніки;
e-mail: yurii.halaichuk@student.karazin.ua; ORCID: https://orcid.org/0009-0004-1048-9425
Марина Мірошник[1], доктор технічних наук, професор; професор кафедри теоретичної та прикладної системотехники; e-mail: m.miroshnyk@karazin.ua;
ORCID: https://orcid.org/0000000222312529
[1]*Харківський національний університет імені В.Н. Каразіна, майдан Свободи, 4, Харків, 61022, Україна*
Рукопис надійшов 20 жовтня 2024 р. Отримано після рецензування 23 листопада 2024 р.
Прийнято 13 грудня 2024 р.

Анотація: Станом на сьогодні штучний інтелект, зокрема нейронні мережі, все більше використовується у програмному забезпеченні в різних галузях, від критично важливих, таких як охорона здоров’я та військова справа, до комерції та сфери розваг. Одним із основних етапів розробки та впровадження такого програмного забезпечення є етап контролю якості. Для запобігання фатальних помилок та втримання у надто конкурентному середовищі програмне забезпечення потребує належного тестування з урахуванням особливостей даних, що отримані у результаті роботи нейронної мережі. У статті наводиться актуальність використання нейронних мереж та аналізуються основні виклики, що виникають при оцінюванні якості таких мереж. Автор робить порівняння властивостей вихідних даних систем штучного інтелекту минулого покоління та новітніх нейронних мереж, виділяє ключові відмінності останніх, такі як потенційна нескінченність наборів вихідних даних, залежність результатів від етапу навчання мережі та суб’єктивну природу оцінювання таких результатів. Основний зміст складає узагальнений огляд існуючих рішень, з акцентом на такі алгоритми та методи, як розрахунок функції втрат, знаходження F-score, методи інтерпретації та імітаційне моделювання. У результаті дослідження автор приходить до висновку, що, не зважаючи на достатню кількість наявних рішень, вони все ще потребують вдосконалення для підвищення точності оцінювання нейронних мереж, адже стовідсоткова точність оцінювання даних все ще не досягнута.

Ключові слова: *F-score, експертні системи, нейронні мережі, оцінювання якості, функція втрат, штучний інтелект*

Конфлікт інтересів: автори повідомляють про відсутність конфлікту інтересів.

ISSN 2519-2310 (Online)

КОМП'ЮТЕРНІ НАУКИ ТА КІБЕРБЕЗПЕКА, 2024, № 2 (26)

COMPUTER SCIENCE AND CYBERSECURITY, 2024, Iss. 2 (26)

DOI: https://doi.org/10.26565/2519-2310-2024-2-03
УДК 004.896

ІНТЕЛЕКТУАЛЬНА БЕЗПЕКА ЕЛЕКТРОСАМОКАТІВ З ТЕХНОЛОГІЄЮ SMARTSTOP

Тетяна Коробейнікова[1], кандидат технічних наук, доцент,
e-mail: tetiana.i.korobeinikova@lpnu.ua, ORCID: https://orcid.org/0000-0003-2487-8742
Олександр Ремінний[1], кандидат технічних наук, асистент, e-mail: sasha.reminnyi@gmail.com,
ORCID: https://orcid.org/0009-0005-5119-3695
Даніель Гада[1], студент кафедри комп'ютеризованих систем автоматики,
e-mail: daniel.hada.ir.2023@lpnu.ua
Артем Гада[1], студент кафедри комп'ютеризованих систем автоматики,
e-mail: artem.hada.ir.2023@lpnu.ua
Назарій Дмитрів[1], студент кафедри комп'ютеризованих систем автоматики,
e-mail: nazarii.dmytriv.ir.2024@lpnu.ua
[1]*Національний університет «Львівська політехніка»,*
вул.С.Бандери, 12, Львів, 79000, Україна
Рукопис надійшов 21 жовтня 2024 р. Отримано після рецензування 22 листопада 2024 р.
Прийнято 22 грудня 2024 р.

Анотація: У статті розглянуто проблему підвищення безпеки електросамокатів у міському середовищі через розробку та впровадження інтелектуальної системи SMARTSTOP. Запропонована система базується на адаптивному контролі швидкості та автоматизованому гальмуванні з використанням сенсорних технологій та алгоритмів реального часу. Ключовими компонентами системи є ультразвукові датчики, мікроконтролер Arduino Nano, потенціометр, сервопривід, електродвигун, LCD-дисплей та п'єзодинамік. SMARTSTOP дозволяє ефективно виявляти статичні та динамічні перешкоди у межах 5-6 метрів з кутом огляду 150 градусів, визначати рівень загрози і відповідно реагувати, знижуючи швидкість або ініціюючи автоматичне гальмування. Для тестування системи використано методику Hardware-in-the-Loop (HiL) на платформі MATLAB/Simulink, що дозволило змоделювати різні дорожні сценарії. Результати тестувань підтвердили високу ефективність і точність роботи системи, яка забезпечує своєчасне реагування на потенційні небезпеки. Подальші напрями розвитку передбачають вдосконалення алгоритмів для складних умов експлуатації та інтеграцію SMARTSTOP з технологіями «розумного міста».

Ключові слова: *електросамокат, безпека руху, адаптивний контроль швидкості, автоматичне гальмування, SMARTSTOP, ультразвукові датчики, Arduino Nano, HiL, міська мобільність*

Як цитувати: Коробейнікова Т., Ремінний О., Гада Д., Гада А., Дмитрів Н. (2024) Інтелектуальна безпека електросамокатів з технологією SMARTSTOP. *Комп'ютерні науки та кібербезпека*. 2024; № 2(26): С. 25–40. https://doi.org/10.26565/2519-2310-2024-2-03

In cites: Korobeynikova T., Reminnyi O., Gada D., Gada A., Dmytriv N. (2024). Intellectual safety of electric scooters with SMARTSTOP technology. *Computer Science and Cybersecurity.* 2(26): 25–40. https://doi.org/10.26565/2519- 2310-2024-2-03 (in Ukrainian)

1. Вступ

Сучасні міста з їх інтенсивним трафіком та обмеженим простором для транспорту потребують розв’язання проблеми ефективності, безпеки та екологічності міського транспорту. Водночас зростальна популярність електричних транспортних засобів, зокрема електросамокатів, відкриває нові можливості для мобільності в межах урбанізованих територій. Однак, разом із позитивними аспектами, такими як зручність і екологічність, з'являються й нові виклики, особливо у контексті безпеки руху.

Одним з основних аспектів безпеки є інтеграція інтелектуальних систем контролю, здатних адаптувати швидкість та забезпечити своєчасну реакцію на змінні умови навколишнього середовища. Важливу роль у таких системах відіграють сенсорні технології, які забезпечують моніторинг простору навколо транспортного засобу, а також алгоритми, що дозволяють автоматично виявляти перешкоди та регулювати швидкість. Подібні технології активно розвиваються і вже знайшли застосування в комерційних продуктах, що включають алгоритми для контролю швидкості та запобігання зіткненням.

Однак, у той час, коли більшість існуючих систем розроблені для велосипедів або автомобілів, розробка системи, орієнтованої на електросамокати, залишається малодослідженою. Це обумовлено необхідністю адаптації технологій до специфіки руху електросамокатів, зокрема у контексті маневрів в умовах щільного міського середовища, де частіше виникають перешкоди та непередбачувані ситуації.

В рамках цієї роботи досліджується та розробляється система SMARTSTOP, яка поєднує адаптивне пригальмовування з автоматичним контролем швидкості. Система здійснює моніторинг оточення на відстані 5-6 метрів і забезпечує реакцію на зміни дорожніх умов, гарантуючи безпечне маневрування для користувачів електросамокатів. Вона адаптується до різних ситуацій, таких як перешкоди на дорозі чи зміна швидкості інших учасників дорожнього руху, знижуючи ймовірність ДТП та забезпечуючи зручність у користуванні.

Детальніше розглянувши *проблеми безпеки в галузі адаптивного пригальмовування*, можна відзначити, що, всупереч численним досягненням в галузі систем допомоги водіям для різних типів транспорту, адаптивний контроль швидкості та виявлення перешкод у міському середовищі залишається відкритим питанням. Особливо це стосується електросамокатів, де динаміка руху є більш непередбачуваною, а маневри, такі як швидкі повороти або зміна напрямку, є звичними для користувачів. Це створює додаткові труднощі для розробки систем, які можуть своєчасно реагувати на виникаючі загрози.

Однією з основних проблем є виявлення перешкод на велодоріжках і в прилеглих зонах. Природа таких перешкод різноманітна: це можуть бути пішоходи, інші транспортні засоби або об’єкти, що знаходяться на доріжці або поблизу неї. Враховуючи змінні умови освітлення та погоди, а також часту присутність об'єктів з обмеженою видимістю (наприклад, припарковані транспортні засоби), ефективне виявлення таких перешкод та своєчасне реагування на них є важливими для забезпечення безпеки.

Крім того, складність ситуації полягає в необхідності точного визначення відстані та швидкості об'єкта, що наближається, та правильної оцінки рівня загрози. Перешкоди можуть знаходитися в межах видимості, але на різній відстані від електросамоката, що впливає на вибір стратегії реагування, зокрема щодо активації пригальмовування чи обмеження швидкості. У

випадку множинних перешкод необхідно здійснювати не лише просте їх виявлення, але й аналіз їхнього впливу на траєкторію руху.

Таким чином, розробка алгоритмів адаптивного гальмування, які враховують положення перешкод, їхню швидкість і близькість, є *актуальним завданням* для підвищення безпеки.

2. Статистика безпеки галузі адаптивного пригальмовування

Аналіз ДТП за участю електросамокатів підтверджує актуальність проблеми. У США кількість інцидентів з електросамокатами зросла на 222% з 2014 по 2018 роки, а у 2020 році було зафіксовано понад 3300 випадків. У Франції у 2021 році сталося 408 аварій, що спричинило посилення регулювання. В Україні за останні 5 років було зафіксовано щонайменше 330 випадків ДТП, з яких 34% припали на 2023 рік. У Німеччині лише за 2020 рік було зареєстровано 2155 ДТП за участю електросамокатів, з яких 386 призвели до серйозних травм. Окрім задекларованих інцидентів, значну частку становлять приховані ДТП, які не реєструються офіційно. Оцінка показує, що в середньому 30-40% інцидентів залишаються поза увагою, що створює додаткові труднощі для аналізу та розробки ефективних заходів безпеки (рис.1).

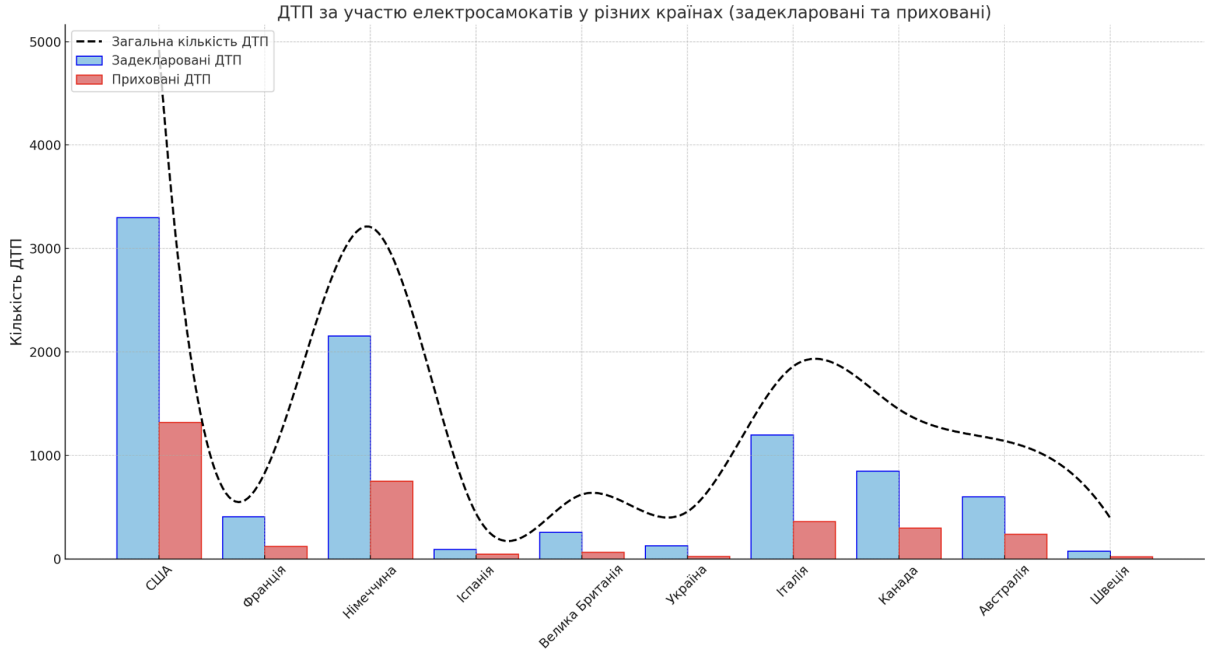

Рис. 1 – Статистика безпеки галузі адаптивного пригальмовування
Fig. 1 – Safety statistics for the adaptive braking industry

Саме тому **метою цієї роботи** є розробка та впровадження *системи адаптивного контролю швидкості та попередження про перешкоди для електросамокатів, яка підвищить безпеку руху в міських умовах*. Система повинна здійснювати моніторинг навколишнього середовища, виявляти як статичні, так і динамічні перешкоди, а також обчислювати відстань до них. Залежно від отриманих даних, система повинна відповідним чином реагувати, активуючи сповіщення користувача та автоматично обмежуючи швидкість або ініціюючи гальмування, коли перешкоди знаходяться в небезпечній близькості. Система має бути інтегрована як комбінація фізичних елементів та програмного модуля, який працюватиме автономно, без підключення до зовнішніх мереж. Її основною функцією є підвищення безпеки електросамоката

в реальному часі, надаючи додаткові можливості для адаптивного управління рухом в умовах міського середовища. Система повинна забезпечувати рівень безпеки не менше 99% у всіх типах ситуацій, запобігаючи потенційним зіткненням та аваріям. Датчики системи повинні безперервно відстежувати ситуацію на дорозі, виявляючи перешкоди в радіусі 5-6 метрів з кутом огляду 150 градусів, і автоматично реагувати на зміну умов, забезпечуючи надійний захист від аварій. Система повинна також мати можливість налаштування або адаптації до різних умов руху та дорожнього середовища.

3. Аналіз наукових досліджень

Враховуючи важливість підвищення безпеки електросамокатів у сучасному міському середовищі, аналіз чинних наукових робіт є необхідним етапом для розуміння технологічних рішень, що можуть бути інтегровані у наш проєкт. Оскільки головним завданням є адаптивне управління швидкістю та гальмами на основі аналізу оточення, особливо з виявленням перешкод, ми зосереджуємося на дослідженнях, які стосуються інтелектуальних систем контролю та гальмування, а також вдосконалення методів запобігання аварійним ситуаціям. У цьому контексті варто звернути увагу на три ключові дослідження, які висвітлюють різні аспекти проблеми безпеки електричних транспортних засобів.

Дослідження класифікації систем інтелектуального контролю для електротранспорту фокусується на різних типах систем безпеки, які сприяють підвищенню безпеки користувачів, зокрема для електротранспорту. В умовах швидко зростаючої популярності цих засобів транспорту необхідно розробляти інтегровані технології, що дозволяють забезпечити надійний захист від аварій та адаптивно реагувати на зміни умов дорожнього руху.

В ході дослідження запропоновано такі категорії інтелектуальних систем контролю.

1. Системи виявлення та запобігання зіткненням (Collision Avoidance Systems): ці системи мають на меті своєчасне виявлення перешкод на шляху транспорту, таких як пішоходи, інші транспортні засоби або нерухомі об'єкти. Вони використовують комбінацію сенсорних технологій, таких як радари, ультразвукові датчики та камери, для забезпечення точного виявлення перешкод. Крім того, ці системи можуть автоматично активувати механізми гальмування для запобігання аварії.

2. Системи адаптивного контролю швидкості (Adaptive Speed Control Systems): ці системи здатні адаптувати швидкість транспортного засобу залежно від умов руху. Вони здійснюють моніторинг відстані до об'єктів попереду та аналізують інтенсивність дорожнього потоку. Система може знижувати швидкість транспортного засобу або обмежувати її, зважаючи на змінювані умови на дорозі, що важливо для забезпечення безпеки в умовах міського трафіку.

3. Системи розпізнавання об'єктів та аналізу дорожнього простору (Object Recognition and Road Analysis Systems): вони використовують алгоритми машинного навчання і комп’ютерного зору для розпізнавання перешкод на дорозі, таких як пішоходи, тварини, інші транспортні засоби, а також для оцінки їхнього розміру та відстані. Це дозволяє системі не лише виявляти небезпечні об'єкти, але й коригувати поведінку транспортного засобу в реальному часі, забезпечуючи максимально безпечний рух.

Проміжні результати дослідження. Подані системи базуються на інтеграції сенсорних технологій і алгоритмів для обробки даних. Для точного визначення відстані до об'єктів використовуються радіолокаційні, ультразвукові та інфрачервоні датчики, а також ширококутні камери. Алгоритми машинного навчання та комп’ютерного зору дозволяють системам ефективно класифікувати об'єкти та реагувати на змінювані умови.

Дослідження підкреслює важливість інтеграції різних сенсорних технологій для забезпечення безпеки електричних транспортних засобів. Впровадження цих систем дозволяє

знижувати ймовірність аварій та підвищувати ефективність транспорту в умовах міського середовища, де постійно змінюються умови руху.

Дослідження «Bicycle Hardware-in-the-Loop Simulator for Braking Dynamics Assistance System» присвячене розробці інтелектуальної системи допомоги в гальмуванні (Braking Dynamics Assistance, BDA) для велосипедів, зокрема електричних. Воно використовує методологію Hardware-in-the-Loop (HiL) для симуляції фізичних процесів без проведення реальних фізичних випробувань. Цей підхід дозволяє уникнути ризиків для користувачів і забезпечує прискорений процес розробки та тестування гальмівних систем. Система BDA адаптується до змінних дорожніх умов і забезпечує стабільність при гальмуванні, знижуючи ймовірність аварій, таких як блокування коліс чи перекидання велосипеда.

У ході дослідження була розроблена система, яка складається з декількох ключових компонентів:

1. Сенсори: Включають датчики швидкості коліс, акселерометри для оцінки динаміки велосипеда, а також датчики тиску в гальмівній системі. Ці сенсори надають інформацію про стан руху, дозволяючи визначити, чи є необхідність у коригуванні гальмівної сили.

2. Контролер: Алгоритм, який аналізує дані від сенсорів і визначає оптимальний рівень гальмівного зусилля. Це важливо, оскільки електричні велосипеди можуть досягати значних швидкостей, що потребує точного контролю.

3. Актуатори: Це гідравлічні системи, які регулюють тиск в реальному часі. Вони дозволяють адаптивно змінювати силу гальмування, залежно від отриманих даних, щоб уникнути блокування коліс або перекидання.

Hardware-in-the-Loop (HiL) є важливою частиною цього дослідження; дозволяє симулювати реальні фізичні процеси, використовуючи ПЗ для тестування та моделювання віртуальних умов. HiL надає перевагу перед традиційними методами тестування, оскільки дозволяє:моделювати різноманітні дорожні умови (погані дороги, мокра поверхня, гравій та ін.); імітувати складні сценарії гальмування та маневрів без необхідності фізичних випробувань; виявляти потенційні проблеми в роботі системи до її впровадження в реальні умови.

Проміжні результати дослідження показали ефективність системи BDA при адаптації до різних дорожніх умов. Система ефективно управляє гальмівним тиском, запобігаючи блокуванню коліс і зменшуючи ризик перекидання велосипеда під час екстреного гальмування. Це особливо важливо для електричних велосипедів, де швидкість руху може бути вищою порівняно з традиційними велосипедами. Однак всупереч позитивним результатам, дослідження також вказує на деякі обмеження. Зокрема, точність симуляцій в рамках HiL може бути обмежена наявністю лише певних сценаріїв, які були модульовані. Наприклад, непередбачувана поведінка велосипедистів або складні погодні умови не можуть бути повною мірою відтворені в симуляторі. Дослідження також зазначає необхідність проведення реальних випробувань для перевірки адаптивності системи до складних, нестандартних ситуацій.

Дослідження «Empirical Survey on Bicycle Accidents to estimate the Potential Benefits of Braking Dynamics Assistance Systems» має на меті оцінити потенційні вигоди від впровадження системи допомоги в гальмуванні для підвищення безпеки велосипедистів, зокрема у контексті електровелосипедів. Враховуючи зростаючу популярність електровелосипедів і пов'язані з цим ризики, це дослідження ставить за мету оцінити, як інтеграція адаптивних гальмівних систем може знизити кількість аварій і підвищити стабільність при русі. Дослідження ґрунтується на аналізі аварій, що сталися за участю електричних велосипедів, і визначенні найбільш ефективних методів підвищення безпеки через використання систем допомоги в гальмуванні. Одним з основних інструментів є експериментальна оцінка ефективності гальмівних систем, яка ґрунтується на реальних даних ДТП, зібраних у різних країнах.

Ключові аспекти:

1. Аналіз аварій: дослідження охоплює статистику ДТП за участю електричних велосипедів, звертаючи увагу на основні причини аварій, такі як надмірна швидкість, екстрене гальмування та погана видимість на дорогах; вивчається вплив дорожніх умов, таких як поверхня дороги (мокра, гравійна), а також погодні умови, на ефективність гальмування.

2. Адаптивні гальмівні системи: у дослідженні наголошується на важливості систем, які здатні адаптуватися до змінюваних умов руху, зокрема через використання сенсорних технологій, які виявляють перешкоди або зміни в дорожній ситуації; обговорюється роль гідравлічних систем та інтеграції електричних компонентів, які дозволяють точніше регулювати гальмівну силу та забезпечувати більш стабільне і безпечне гальмування при високих швидкостях або в складних умовах.

Проміжні результати дослідження продемонстрували значні переваги від використання адаптивних систем гальмування для електричних велосипедів, зокрема в умовах екстреного гальмування. Виявлено, що використання гідравлічних і електричних компонентів дозволяє не тільки покращити стабільність при високих швидкостях, а й знижує ймовірність блокування коліс та інших аварійних ситуацій. Незважаючи на значні досягнення, дослідження вказує на деякі обмеження в застосуванні адаптивних систем, зокрема при адаптації до складних погодних умов або поведінки інших учасників дорожнього руху, які важко прогнозувати. Також наголошено на необхідності проведення додаткових тестів для перевірки ефективності таких систем в реальних умовах. Отже, впровадження адаптивних гальмівних систем може значно *зменшити кількість аварій*, особливо на електричних велосипедах, де швидкість та потужність можуть створювати додаткові ризики.

4. Виклад основного матеріалу

Дослідження показали, що підвищення безпеки електросамокатів у міському середовищі є надзвичайно важливим завданням для сучасного транспорту. Аналіз наукових досліджень показує ключові напрямки, які можуть сприяти значному зниженню аварій та підвищенню стабільності електричних транспортних засобів. Розглянуті дослідження підкреслюють важливість інтеграції сенсорних технологій та адаптивного контролю швидкості і гальмування, що дозволяє системам реагувати в реальному часі на змінювані дорожні умови. Використання Hardware-in-the-Loop (HiL) для симуляцій дає змогу тестувати нові системи без фізичних випробувань, що знижує ризики для користувачів на етапах розробки. Особливу увагу варто приділити адаптивним гальмівним системам, які інтегрують гідравлічні та електричні компоненти для забезпечення стабільного гальмування при високих швидкостях та екстремальних умовах. Проте, для досягнення повної ефективності таких систем, необхідно враховувати складні умови, такі як погодні фактори та непередбачувану поведінку учасників руху.

Загалом, ці дослідження демонструють значний потенціал для розвитку безпечніших і ефективніших електричних транспортних засобів, підвищуючи рівень їх стабільності та знижуючи ймовірність аварій в умовах міського трафіку.

Після введення у контекст проблеми безпеки електросамокатів і опису наукових досліджень, що підтверджують актуальність розробки інтелектуальних систем для підвищення безпеки, наступним важливим кроком є детальне розкриття архітектури прототипу SMARTSTOP. Система, яку ми розробляємо, поєднує кілька інноваційних підходів, включаючи автоматизоване пригальмовування, контроль швидкості та інтерактивне сповіщення для користувача. Метою цього розділу є представити структуру електричної схеми та опис основних компонентів, які забезпечують роботу нашої системи. Зокрема, зосередимося на

підключеннях, функціональності та принципах роботи кожного елемента в системі SMARTSTOP, яка реалізована на базі мікроконтролера Arduino Nano (рис. 2, а), спрощена функціональна схема прототипу наведена на рис. 2, б.

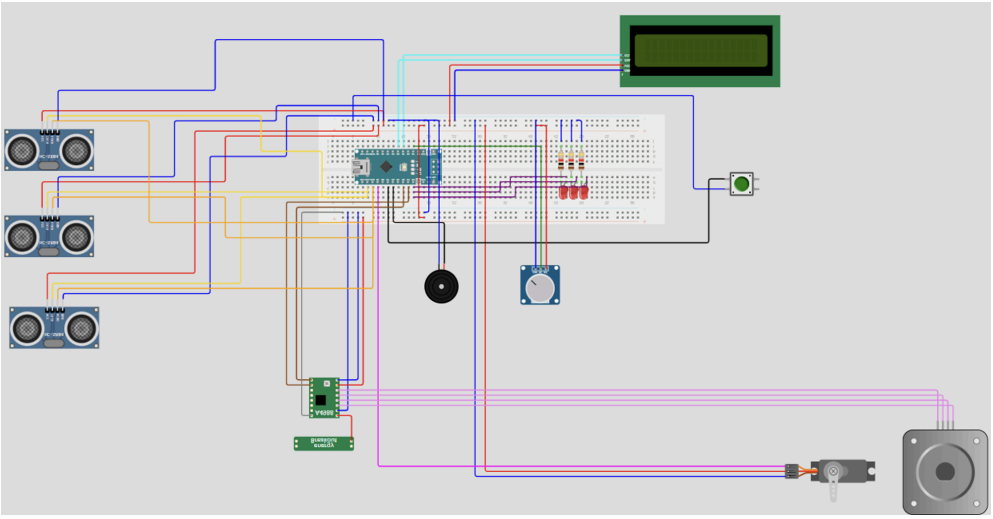

Рис. 2.а – Схема роботи системи SMARTSTOP (electrical circuit (Wokwi platform))
Fig. 2.a – SMARTSTOP system operation diagram (electrical circuit (Wokwi platform))

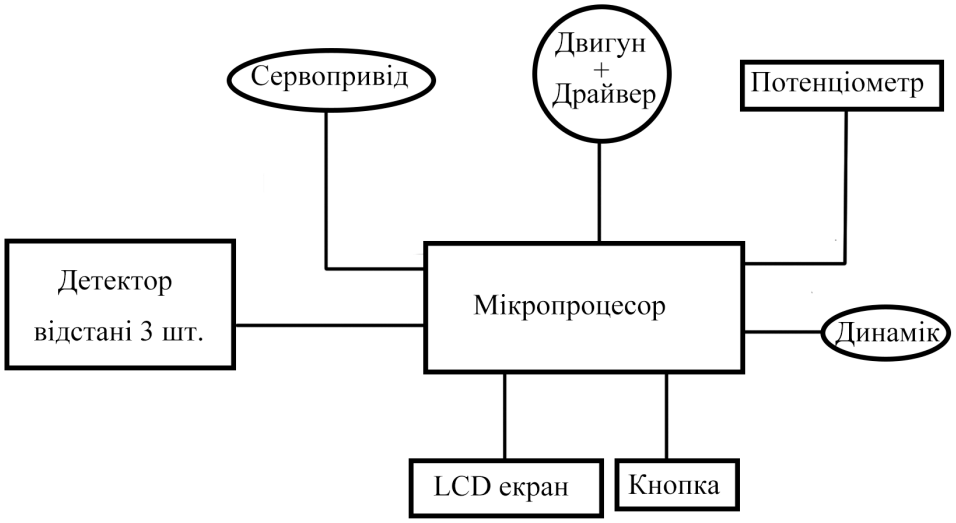

Рис. 2.б – Схема роботи системи SMARTSTOP (спрощена фунціональна схема)
Fig. 2.b – SMARTSTOP system operation diagram (simplified functional diagram)

Схема, представлена в прототипі SMARTSTOP (рис. 2.б), базується на інтеграції кількох технологій, що дозволяє створити систему, яка ефективно реагує на зміни в навколишньому середовищі та адаптується до різних ситуацій на дорозі. Ключовими компонентами є:

1. Arduino Nano – центральний елемент схеми, який здійснює збір даних із сенсорів та управління всіма підсистемами. Він забезпечує обробку сигналів від ультразвукових датчиків, акселерометра та інших елементів, здійснює керування мотором та сервоприводом, а також виводить необхідну інформацію на дисплей.

2. Ультразвукові датчики (HC-SR04) – встановлені для виявлення перешкод перед електросамокатом у радіусі 4-5 метрів. Вони мають поле огляду 150 градусів і передають ультразвукові імпульси, що дозволяє точно вимірювати відстань до об’єкта, аналізуючи його положення відносно велодоріжки.

3. Потенціометр – імітує ручку газу, надаючи користувачу можливість регулювати швидкість самоката. Однак у разі небезпечної ситуації, система блокує передачу сигналу на двигун, щоб обмежити швидкість.

4. Сервопривід – активує механізм гальмування, забезпечуючи плавне і безпечне пригальмовування в критичних ситуаціях.

5. Електродвигун з драйвером (L-293D) – мотор, який регулюється за допомогою Arduino для зміни швидкості руху самоката, зменшуючи напругу при необхідності зменшити швидкість або зупинити транспортний засіб.

6. LCD-дисплей (16x2) – відображає корисну інформацію для користувача: відстань до перешкоди, поточну швидкість, попередження щодо обмеження швидкості тощо.

7. П’єзодинамік – генерує звукові сигнали для попередження користувача про небезпеки на дорозі, такі як наближення до перешкоди або перевищення швидкості.

8. Світлодіоди і кнопка – використовуються для індикації стану системи і для введення користувачем певних команд, таких як вмикання/вимикання системи.

Ця схема є основою для *створення інтелектуальної системи безпеки*, яка зможе інтегруватися з іншими транспортними засобами або використовуватися як окремий модуль для електросамокатів, щоб забезпечити підвищену безпеку руху в міському середовищі.

Підключення та інтеграція компонентів містить технічний опис підключень між сенсорами, контролером та іншими виконавчими елементами.

1. Arduino Nano як центральний елемент системи має піни для підключення інших елементів, таких як ультразвукові датчики, акселерометр, дисплей, п’єзодинамік та інші компоненти.

2. Ультразвукові датчики підключені до цифрових пінів Arduino через два піні: TRIG (відповідає за ініціацію ультразвукового імпульсу) та ECHO (отримання імпульсу після відбиття від об'єкта). Для кожного датчика виділяються окремі піні на Arduino для забезпечення коректної роботи всієї системи.

3. Потенціометр для регулювання швидкості самоката підключений до аналогового входу Arduino. Його сигнал передається безпосередньо до мікроконтролера для регулювання мотора самоката через драйвер A4988.

4. Сервопривід використовується для активації механізму гальмування. Він підключається до цифрового піну Arduino через ШІМ-сигнал, що дозволяє точно контролювати рух гальмівного механізму.

5. LCD-дисплей (16x2) підключений через I2C шину, що дозволяє зекономити піні на Arduino, використовуючи тільки дві піни для передачі даних (SDA та SCL) разом із живленням (VCC і GND).

6. П’єзодинамік для генерування звукових сигналів підключається до цифрового піну, де мікроконтролер керує звуковими сигналами через ШІМ-сигнал.

7. Електродвигун підключається до драйвера, який відповідає за керування швидкістю двигуна на основі ШІМ-сигналу від Arduino.

Розбір роботи кожного сенсора, сервоприводу, дисплея та інших частин схеми.

1. Arduino Nano є мозком системи, обробляючи дані з датчиків та керуючи іншими компонентами. Воно виконує кілька основних функцій: 1) обробка сигналів від ультразвукових датчиків для визначення відстані до перешкод; 2) оцінка швидкості самоката через акселерометр та потенціометр; 3) управління гальмуванням через сервопривід та регулювання

швидкості мотора через драйвер A4988; 5) виведення інформації на дисплей та активація звукових сигналів для попередження користувача.

2. Ультразвукові датчики (HC-SR04) використовуються для виявлення перешкод на дорозі. Кожен датчик має два ключових етапи роботи: 1) тригери (TRIG): виводять ультразвуковий імпульс у вигляді хвилі; 2) Exo (ECHO): отримує відбиту хвилю і на основі часу її повернення обчислюється відстань до об'єкта. Інформація передається на Arduino для подальшого аналізу та прийняття рішень про активацію гальмування чи обмеження швидкості.

3. Акселерометр вимірює зміну швидкості та прискорення самоката. Він дозволяє Arduino отримувати точні дані про рух, що є необхідним для управління швидкістю самоката та визначення, коли потрібно зменшити швидкість через небезпеку.

4. Потенціометр: імітує ручку газу на електросамокаті. Він дозволяє користувачу регулювати бажану швидкість. Залежно від положення потенціометра, Arduino аналізує сигнал і визначає максимальну швидкість, яка може бути досягнута. Якщо відстань до перешкоди стає критичною, система автоматично обмежує або знижує швидкість.

5. Сервопривід використовується для імітації дії гальмівного механізму. Якщо система визначає небезпеку або перешкоду, сервопривід активує механізм гальмування, забезпечуючи плавне уповільнення електросамоката.

6. Електродвигун з драйвером забезпечує рух самоката. Драйвер A4988 контролює швидкість мотора за допомогою ШІМ-сигналу, що надходить від Arduino. Коли система визначає необхідність зменшити швидкість, Arduino посилає команду для зменшення напруги на моторі, знижуючи швидкість руху.

7. LCD-дисплей (16x2) служить для виведення важливої інформації для користувача, зокрема відстані до перешкоди, поточної швидкості, попереджень і обмежень. Дисплей працює через шину I2C, що дозволяє економити піні мікроконтролера, спрощуючи підключення.

8. П’єзодинамік генерує звукові сигнали для попередження користувача про небезпеки на дорозі. Коли система виявляє потенційну загрозу (наприклад, наближення до перешкоди), мікроконтролер активує п’єзодинамік для генерування звукового сигналу.

5. Алгоритми роботи запропонованої системи

На рис. 3 наведено базовий алгоритм, що є основою для розробки алгоритмів адаптивного гальмування та сповіщення в системі SMARTSTOP, спрямованої на підвищення безпеки електросамокатів у міському середовищі.

1. Старт системи. Система активується, запускаючи процес збору даних із сенсорів: CurrentSpeed – поточна швидкість самоката; Sensor1, Sensor2, Sensor3 – три ультразвукові сенсори, що відстежують перешкоди з різних сторін; DistanceSensor1, DistanceSensor2, DistanceSensor3 – визначають відстань до об'єкта для кожного сенсора; RelativeSpeed – вимірює відносну швидкість наближення до об'єкта.

2. Перевірка швидкості. Якщо поточна швидкість (CurrentSpeed) = 0, система не виконує жодних дій, оскільки самокат не рухається. У разі руху (CurrentSpeed > 0) система переходить до аналізу даних із сенсорів.

3. Перевірка сенсорів. Система послідовно перевіряє, чи активовано кожен із трьох сенсорів: Sensor1 = True – перешкода виявлена сенсором 1; Sensor2 = True – перешкода виявлена сенсором 2; Sensor3 = True – перешкода виявлена сенсором 3. Якщо всі сенсори не виявляють перешкод, система продовжує спостереження.

4. Визначення відстані. Для кожного активного сенсора система визначає відстань до перешкоди за допомогою відповідного DistanceSensor. Якщо перешкоди немає, система пропускає обчислення для цього сенсора.

5. Вимірювання відносної швидкості. Для кожного активного сенсора обчислюється RelativeSpeed – відносна швидкість об'єкта щодо самоката.

6. Призначення рівнів ризику. Відстань (Distance) і відносна швидкість (RelativeSpeed) порівнюються, щоб визначити рівень ризику: MIN – низький ризик; MID – середній ризик; MAX – високий ризик.

7. Реакція системи. В залежності від рівня ризику активуються відповідні заходи безпеки: MIN: Звукове попередження, незначне обмеження швидкості; MID: Звукове попередження, обмеження швидкості; MAX: Сильне звукове попередження, суворе обмеження швидкості, або екстрене гальмування.

8. Циклічне повторення. Після виконання дії система повертається до початку, оновлює дані та знову аналізує ситуацію. Особливості реалізації: 1. Мультисенсорний підхід забезпечує високу точність виявлення перешкод за допомогою трьох ультразвукових датчиків; 2. Рівні ризику диференційовані реакції залежно від відстані до перешкоди та її швидкості; 3. Безперервний моніторинг ситуації забезпечує безперервний захист.

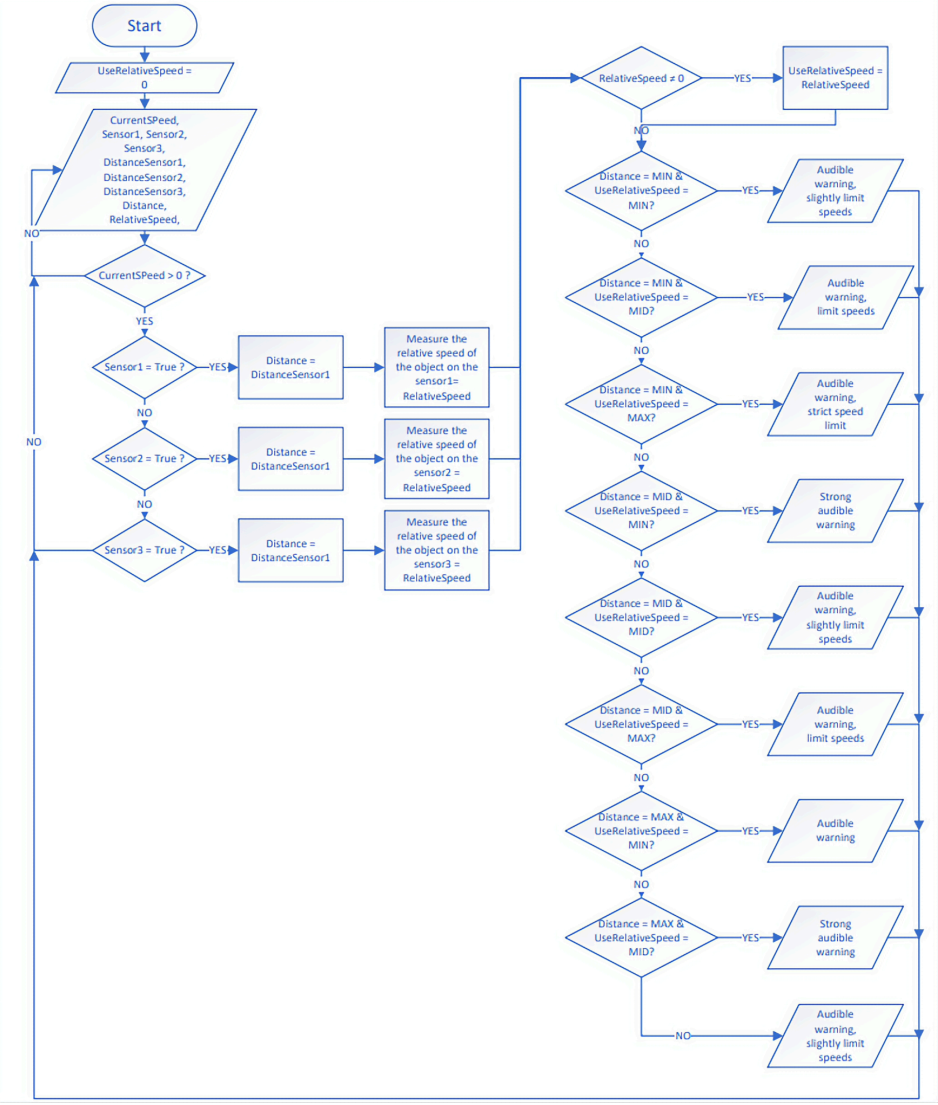

Рис. 3. – Базовий алгоритм розробки алгоритмів адаптивного гальмування та сповіщення

Fig. 3. - Basic algorithm for developing adaptive braking and warning algorithms

На рис. 4 наведено алгоритм роботи прототипу автоматичного виявлення перешкоди за допомогою трьох ультразвукових датчиків та контролю швидкості моторчика для уникнення зіткнень. Управління здійснюється кнопкою для вмикання/вимикання системи, а регулювання швидкості – потенціометром. Реакція системи змінюється залежно від відстані до перешкоди.

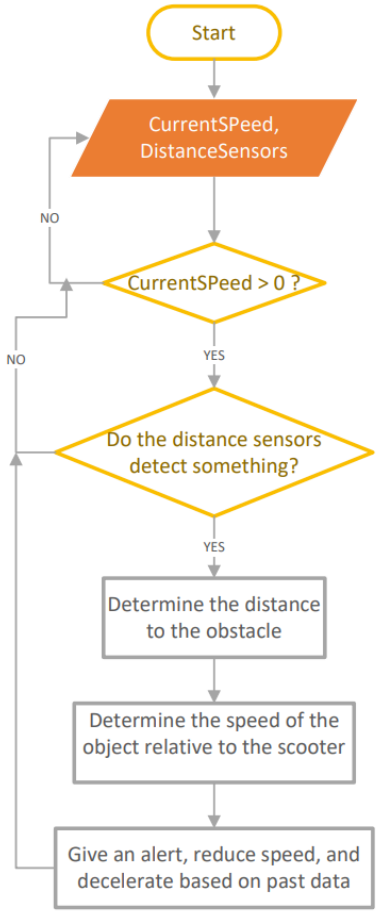

Рис.4. – Алгоритм роботи прототипу
Fig. 4. – Algorithm of the prototype operation

Передній датчик – контролює об'єкти, що знаходяться прямо перед пристроєм. Лівий датчик – нахилений для огляду лівої зони на кут 50°. Правий датчик – нахилений для огляду правої зони на кут 50°. Кут огляду системи: 150° загального простору.

Основні стани системи:

1. Безпечно (немає перешкод). Умови: Жоден датчик не фіксує об'єкти на критичній відстані. Дії: 1) Моторчик обертається зі швидкістю, заданою потенціометром. 2) На екрані відображається повідомлення: «Безпечно». 3) Користувач може регулювати швидкість без обмежень.

2. Попередження (перешкода > 40 см). Умови: Один або кілька датчиків виявили об'єкт на відстані понад 40 см. Дії: 1) Виводиться попередження на екрані про можливу небезпеку. 2) Лунає короткий звуковий сигнал. 3) Моторчик продовжує працювати у звичайному режимі, але користувачеві рекомендується зменшити швидкість.

А. Перешкода попереду (передній датчик). Низька швидкість: виводиться повідомлення: «Перешкода попереду». Висока швидкість: Активується сервопривід для поступового

зменшення швидкості до нуля; потенціометр блокується, щоб уникнути збільшення швидкості; після зникнення перешкоди система чекає 2 секунди, перш ніж дозволити відновлення швидкості.

Б. Перешкода збоку (лівий/правий датчик): Низька швидкість: виводиться повідомлення: «Перешкода збоку. Будьте обережні». Висока швидкість: швидкість моторчика поступово знижується, але не до нуля; потенціометр блокується до зникнення перешкоди; виводиться повідомлення: «Зменшення швидкості. Перешкода збоку».

Дії після зникнення перешкоди з поля зору датчиків: 1) Система очікує 2 секунди для перевірки стабільності. 2) Потенціометр розблоковується. 3) Швидкість моторчика поступово відновлюється до рівня, встановленого користувачем. 4) Виводиться повідомлення: «Все добре».

Циклічність роботи систем. Система працює в циклічному режимі, оновлюючи дані з датчиків кожні кілька мілісекунд. Аналізується інформація з усіх трьох сенсорів для забезпечення швидкого реагування.

Особливості алгоритму:

1. Три датчики забезпечують широкий кут огляду (150°).
2. Сервопривід імітує автоматичне гальмування.
3. Потенціометр блокується при виявленні критичної перешкоди для безпеки.
4. Екран надає текстові повідомлення про стан системи.
5. Звукові сигнали додатково попереджають користувача про небезпеку.

Запропонована система виявлення перешкод з автоматичним регулюванням швидкості демонструє надійну та ефективну роботу. Використання трьох ультразвукових датчиків забезпечує широкий кут огляду у 150°, дозволяючи своєчасно виявляти перешкоди як попереду, так і збоку. Реакція системи залежить від критичності ситуації: система знижує швидкість або активує автоматичне гальмування для уникнення зіткнень. Блокування потенціометра та інформування користувача за допомогою текстових повідомлень і звукових сигналів підвищує безпеку та зручність експлуатації. Завдяки циклічному оновленню даних з датчиків система швидко реагує на зміни навколишнього середовища, що робить її надійною для застосування в різних сценаріях.

6. Тестування пристрою

Тестування системи SMARTSTOP є важливим етапом для оцінки її працездатності та ефективності. Оскільки система поєднує різні технології для забезпечення безпеки електросамокатів, такі як автоматизоване гальмування, контроль швидкості та сповіщення, необхідно здійснити детальне тестування її компонентів на різних етапах розробки. У цьому контексті використання платформи MATLAB з підтримкою HiL (Hardware-in-the-Loop) технології дозволяє провести віртуальне тестування системи, поєднуючи реальні фізичні компоненти з віртуальними моделями для аналізу і перевірки алгоритмів без необхідності створення фізичних моделей.

Моделювання в MATLAB.

1. Arduino Nano моделюється як центральний елемент системи, який здійснює обробку даних з ультразвукових датчиків, акселерометра та інших сенсорів, а також керує виконавчими елементами, такими як електродвигун і сервопривід.

2. Ультразвукові датчики (HC-SR04) змодельовані через функції, що генерують імпульси для виявлення перешкод на різних відстанях. Дані від датчиків передаються на Arduino Nano, який обробляє інформацію і визначає, чи потрібно активувати гальмування чи обмеження швидкості.

3. Акселерометр для вимірювання швидкості руху самоката та потенціометр, що регулює швидкість руху, моделюються як аналогові входи, що передають сигнали на Arduino для подальшої обробки.

4. Сервопривід і електродвигун з драйвером були змодельовані як елементи, що реагують на сигнали від мікроконтролера Arduino, імітуючи зміну швидкості руху та активацію гальмівного механізму.

5. LCD-дисплей та п’єзодинамік змодельовані для виведення інформації та звукового сповіщення користувача в залежності від рівня небезпеки.

Імітація тестових сценаріїв:

1. Тестування на перешкоди: Для кожного з трьох ультразвукових датчиків були створені кілька сценаріїв з різними типами перешкод.

2. Тестування адаптивного контролю швидкості: Моделювалися різні умови, коли система повинна автоматично обмежувати швидкість самоката в залежності від наявності перешкоди або небезпечної ситуації

3. Імітація швидкості та змін в поведінці користувача: Положення потенціометра змінювалося для регулювання швидкості, що дозволяло перевірити реакцію системи на бажану швидкість та її обмеження в разі наближення до перешкоди.

4. Автоматичне гальмування: Створено сценарії, де швидкість самоката перевищує встановлену безпечну межу і система активує гальмівний механізм або обмежує швидкість.

Результати тестування.

Тестування системи SMARTSTOP за допомогою MATLAB/Simulink показало досить високу ефективність алгоритму і компонентів системи:

1. Точність виявлення перешкод: Усі ультразвукові датчики показали хорошу точність виявлення перешкод на відстанях до 4 метрів, а також ефективно визначали положення об'єктів відносно велодоріжки.

2. Адаптивне управління швидкістю: Система коректно адаптувала швидкість самоката залежно від наявності перешкод, автоматично знижуючи швидкість у разі необхідності.

Автоматичне гальмування: У випадку наближення до перешкоди на небезпечну відстань, система ефективно активувала гальмівний механізм через сервопривід, що забезпечувало плавне і безпечне уповільнення.

3. Інтерактивне сповіщення користувача: Звукові сигнали та повідомлення на LCD-дисплеї успішно інформували користувача про поточні умови на дорозі та стан системи.

Результати тестування в MATLAB/Simulink з підтримкою HiL довели ефективність і стабільність системи SMARTSTOP у забезпеченні безпеки електросамокатів. Моделювання і тестування дозволили точно налаштувати алгоритми адаптивного гальмування, обмеження швидкості і сповіщення користувача, що забезпечує високий рівень безпеки в реальних умовах. Подальші етапи розвитку включатимуть верифікацію результатів у реальних умовах, а також оптимізацію енергоспоживання та алгоритмів для покращення реакції системи на складніші сценарії.

7. Висновки

У статті представлено інноваційну систему SMARTSTOP, яка вирішує нагальну проблему безпеки користувачів електросамокатів у сучасних містах. Розроблена система інтегрує сенсорні технології та інтелектуальні алгоритми, забезпечуючи адаптивне управління швидкістю й автоматичне гальмування в режимі реального часу. Завдяки використанню ультразвукових датчиків, системі вдається своєчасно виявляти як статичні, так і динамічні перешкоди на відстані до 6 метрів з широким кутом огляду.

Результати випробувань, проведених за допомогою платформи MATLAB/Simulink із застосуванням технології Hardware-in-the-Loop, підтвердили високу точність та надійність SMARTSTOP. Система ефективно адаптується до різних дорожніх ситуацій, швидко реагує на потенційні загрози та забезпечує безпеку руху на рівні понад 99%. Інтерактивні сповіщення та автоматичні дії системи значно підвищують комфорт та впевненість користувачів під час пересування містом. Отже, SMARTSTOP – це сучасне та перспективне рішення, що може суттєво знизити кількість аварій і зробити міський транспорт безпечнішим. Подальший розвиток системи відкриває нові можливості для інтеграції у концепцію «розумного міста» та покращення міської мобільності загалом.

Конфлікт інтересів

Автори повідомляють про відсутність конфлікту інтересів.

Список літератури

1. Wanganoo, L., Shukla, V., & Mohan, V. (2022). Intelligent micro‑mobility e‑scooter: revolutionizing urban transport. *Trust‑based communication systems for internet of things applications*, 267-290. DOI: https://doi.org/10.1002/9781119896746.ch11
2. D'Addato, M. (2021). Design of a smart sensor for obstacles detection on electric scooters or bicycles (*Doctoral dissertation, Politecnico di Torino*), URL: http://webthesis.biblio.polito.it/id/eprint/18054
3. Vinayaga-Sureshkanth, N., Wijewickrama, R., Maiti, A., & Jadliwala, M. (2020). Security and privacy challenges in upcoming intelligent urban micromobility transportation systems. *In Proceedings of the Second ACM Workshop on Automotive and Aerial Vehicle Security*, pp. 31-35, DOI: https://doi.org/10.1145/3507657.3528551
4. Jang, W. J., Kim, D. H., & Lim, S. H. (2023). An AI safety monitoring system for electric scooters based on the number of riders and road types. *Sensors*, 23(22), 9181, DOI: https://doi.org/10.3390/s23229181
5. Alai, H., Jeon, W., Alexander, L., & Rajamani, R. (2024). A smart e-scooter with embedded estimation of rear vehicle trajectories for rider protection. *Mechanical Systems and Signal Processing*, 222, 111786, DOI: https://doi.org/10.1016/j.ymssp.2024.111786
6. Ma, Q., Yang, H., Mayhue, A., Sun, Y., Huang, Z., & Ma, Y. (2021). E-Scooter safety: The riding risk analysis based on mobile sensing data. *Accident Analysis & Prevention*, 151, 105954, DOI: https://doi.org/10.1016/j.aap.2020.105954
7. Бабій, М. В., Бабій, В. А., Мартинчук, А. О. (2023). Інтелектуальні системи безпеки руху. *Матеріали V Міжнародної науково-практичної конференції «Підвищення надійності і ефективності машин, процесів і систем»*. Кропивницький: ЦНТУ, 156.
8. Яковенко, Т. К. (2024). Датчики руху на основі мікроконтролера Arduino в інтелектуальних системах безпеки.

INTELLECTUAL SAFETY OF ELECTRIC SCOOTERS WITH SMARTSTOP TECHNOLOGY

Tetiana Korobeynikova[1], PhD (Engineering), Associate Professor;
e-mail: tetiana.i.korobeinikova@lpnu.ua; ORCID: https://orcid.org/0000-0003-2487-8742
Oleksandr Reminnyi[1], PhD (Engineering), Assistant; e-mail: sasha.reminnyi@gmail.com;
ORCID: https://orcid.org/0009-0005-5119-3695
Daniel Gada[1], student of the Department of Computerised Automation Systems;
e-mail: daniel.hada.ir.2023@lpnu.ua;
Artem Gada[1], student of the Department of Computerised Automation Systems;
e-mail: artem.hada.ir.2023@lpnu.ua;
Nazariy Dmytriv[1], student of the Department of Computerised Automation Systems;
e-mail: nazarii.dmytriv.ir.2024@lpnu.ua
[1]*Lviv Polytechnic National University, Ukraine*
Manuscript was received October 21, 2024; Received after review November 22, 2024; Accepted December 22, 2024

Abstract. The article considers the problem of improving the safety of electric scooters in the urban environment through the development and implementation of the SMARTSTOP intelligent system. The proposed system is based on adaptive speed control and automated braking using sensor technologies and real-time algorithms. The key components of the system are ultrasonic sensors, an Arduino Nano microcontroller, a potentiometer, a servo drive, an electric motor, an LCD display, and a piezoelectric speaker. SMARTSTOP allows you to effectively detect static and dynamic obstacles within 5-6 m. with a viewing angle of 150 degrees, determine the level of threat and respond accordingly by slowing down or initiating automatic braking. The system was tested using the Hardware-in-the-Loop (HiL) methodology on the MATLAB/Simulink platform, which allowed us to simulate various road scenarios. The test results confirmed the high efficiency and accuracy of the system, which ensures timely response to potential hazards. Further development areas include improving algorithms for difficult operating conditions and integrating SMARTSTOP with smart city technologies.

Keywords: *electric scooter, traffic safety, adaptive speed control, automatic braking, SMARTSTOP, ultrasonic sensors, Arduino Nano, HiL, urban mobility*

Conflicts of Interest: the authors declare no conflict of interest.

References

1. Wanganoo, L., Shukla, V., & Mohan, V. (2022). Intelligent micro‑mobility e‑scooter: revolutionizing urban transport. *Trust‑based communication systems for internet of things applications*, 267-290. DOI: https://doi.org/10.1002/9781119896746.ch11
2. D'Addato, M. (2021). Design of a smart sensor for obstacles detection on electric scooters or bicycles (*Doctoral dissertation, Politecnico di Torino*), URL: http://webthesis.biblio.polito.it/id/eprint/18054
3. Vinayaga-Sureshkanth, N., Wijewickrama, R., Maiti, A., & Jadliwala, M. (2020). Security and privacy challenges in upcoming intelligent urban micromobility transportation systems. *In Proceedings of the Second ACM Workshop on Automotive and Aerial Vehicle Security*, pp. 31-35, DOI: https://doi.org/10.1145/3507657.3528551
4. Jang, W. J., Kim, D. H., & Lim, S. H. (2023). An AI safety monitoring system for electric scooters based on the number of riders and road types. *Sensors*, 23(22), 9181, DOI: https://doi.org/10.3390/s23229181

5. Alai, H., Jeon, W., Alexander, L., & Rajamani, R. (2024). A smart e-scooter with embedded estimation of rear vehicle trajectories for rider protection. *Mechanical Systems and Signal Processing*, 222, 111786, DOI: https://doi.org/10.1016/j.ymssp.2024.111786
6. Ma, Q., Yang, H., Mayhue, A., Sun, Y., Huang, Z., & Ma, Y. (2021). E-Scooter safety: The riding risk analysis based on mobile sensing data. *Accident Analysis & Prevention*, 151, 105954, DOI: https://doi.org/10.1016/j.aap.2020.105954
7. Babiy, M. V., Babiy, V. A., & Martynchuk, A. O. (2023). Intelligent traffic safety systems. *Materials of the V International Scientific and Practical Conference ‘Improving the reliability and efficiency of machines, processes and systems’*. Kropyvnytskyi: TSNTU, 156 [in Ukrainian]
8. Yakovenko, T. K. (2024). Motion sensors based on the Arduino microcontroller in intelligent security systems [in Ukrainian]

ISSN 2519-2310 (Online)

КОМП'ЮТЕРНІ НАУКИ ТА КІБЕРБЕЗПЕКА, 2024, № 2 (26)

COMPUTER SCIENCE AND CYBERSECURITY, 2024, Iss. 2 (26)

DOI: https://doi.org/10.26565/2519-2310-2024-2-04
УДК 004.41

ВИЗНАЧЕННЯ КРИТЕРІЇВ ВИКОРИСТАННЯ СЕРВІСНОЇ (SOA) ТА МІКРОСЕРВІСНОЇ АРХІТЕКТУРИ (MSA) ПОБУДОВИ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

Олег Сєдашев[1], аспірант факультету математики і інформатики,
e-mail: oleh.siedashev@student.karazin.ua, ORCID: https://orcid.org/0009-0001-3259-3141
[1]Харківський національний університет імені В.Н. Каразіна,
майдан Свободи, 4, Харків, 61022, Україна

Рукопис надійшов 7 листопада 2024 р. Отримано після рецензування 8 грудня 2024 р.
Прийнято 22 грудня 2024 р.

Анотація: У сучасній розробці програмного забезпечення одним із ключових завдань є вибір відповідної архітектури для системи на ранніх етапах її проектування. Ця стаття розглядає два популярні підходи побудови програмного забезпечення: сервіс-орієнтованої архітектури (SOA) та мікросервісної архітектури (MSA). На основі аналізу архітектурних особливостей, переваг та недоліків цих підходів, досліджуються критерії, які впливають на вибір архітектурної моделі в залежності від специфіки системи. Мікросервісна архітектура, завдяки своїй незалежності та можливості швидкого масштабування, краще підходить для динамічних систем з високими вимогами до гнучкості. Сервісно-орієнтована архітектура, навпаки, орієнтована на централізоване управління сервісами через ESB (Enterprise Service Bus) і надає кращі можливості для інтеграції та повторного використання компонентів у великих корпоративних системах, та які не потребують частих змін функціоналу. Основна увага у статті приділяється розробці методу оцінки, який дозволить розробникам програмного забезпечення та системним інженерам на ранніх стадіях проектування визначити яку з архітектур, SOA чи MSA, доцільніше використовувати для конкретної системи. Враховуючи різні технічні та вимоги, метод виділяє ключові критерії на які слід звертати увагу при обранні архітектури програмного забезпечення додатку.

Ключові слова: *інформаційні системи, архітектура програмного забезпечення, сервіс-орієнтована архітектура, мікросервісна архітектура*

Як цитувати: Сєдашев О. Визначення критеріїв використання сервісної (SOA) та мікросервісної архітектури (MSA) побудови програмного забезпечення. *Комп'ютерні науки та кібербезпека*. 2024; № 2(26): С. 41–50. https://doi.org/10.26565/2519-2310-2024-2-04

In cites: Siedashev O. (2024). Determination of software architecture (SOA) and microservice architecture (MSA) usage criteria. *Computer Science and Cybersecurity.* 2(26): 41–50. https://doi.org/10.26565/2519- 2310-2024-2-04 (in Ukrainian)

Постанова проблеми

Проектування програмного забезпечення є одним з найважливішим етапів його розробки, потребує детального аналізу функціональних та нефункціональних вимог, а також обрання архітектури та технологій, які будуть використовуватися. Наразі існує безліч інструментів та описаних підходів рекомендованих для використання побудови додатків з нуля, але розвиток побудови програмного забезпечення починався з підходу використання «монолітної архітектури». Підхід, коли уся система є одним цілим, фізично розташований на одній машині та виконує усі операції як один процес. Через виклики масштабування, підтримки та оновлення системи цей підхід еволюціонував в сервіс-орієнтований підхід. У сервіс-орієнтованій архітектурі (SOA) система розділяється на окремі сервіси, які спілкуються між собою через спільні протоколи, часто з використанням шини (ESB). Це дозволило підвищити модульність і спрощувати інтеграцію різних частин системи. Мікросервісна архітектура (MSA) виникла з подальшою потребою у ще більшій гнучкості та незалежності сервісів. На відміну від SOA, де сервіси можуть бути більш тісно пов’язані, в MSA сервіси є дрібнішими і повністю незалежними один від одного. Кожен сервіс може мати свою базу даних і бути розгорнутим та масштабованим автономно. Цей підхід знижує складність підтримки системи та дозволяє швидше впроваджувати зміни, але вимагає більш складного управління і координації. Наразі не існує чіткого визначення, або критеріїв за якими слід обирати використання SOA чи MSA, та супутніх технологій на ранніх етапах проектування програмного забезпечення. Неправильний вибір без урахування усіх факторів може дуже негативно позначитись на усій системі та її здатності адаптуватися до змін бізнес-вимог у майбутньому або змін технологій.

Аналіз останніх досліджень і публікацій

Дослідження сервісного підходу побудови програмного забезпечення не зупиняються, та практично використовуються в індустрії. У роботі «Monolith to Microservices: Evolutionary Patterns to Transform Your Monolith» [1] Сем Ньюман досліджує перехід від монолітних архітектур до мікросервісних, та занурюється в тему декомпозиції системи, як концепту. Книга складається з 5 глав, які описують опис мікросервісної архітектури, коли ця архітектура є актуальним рішенням, проблеми та технічні аспекти міграції з монолітної архітектури, та патерни, які допомагають вирішити ці виклики. В іншій роботі Сема Ньюмана «Building Microservices: Designing Fine-Grained Systems» [2] розглядаються загальні підходи архітектури мікросервісів, їх проектування та реалізація. Автор акцентує увагу на принципах автономності сервісів, управлінні даними та інтеграції, надаючи практичні рекомендації для створення мікросервісних систем. Ньюман також підкреслює важливість тестування та моніторингу для забезпечення стабільності та продуктивності системи. У персональному блозі [3] про мікросервісну архітектуру Chris Richardson дає визначення поняттям пов’язаним з мікросервісами та описує найкращі практики індустрії, які використовуються при побудові мікросервісних додатків. У книзі Mark Richards, Neal Ford «Fundamentals of Software Architecture: An Engineering Approach» [4] детально описується що таке архітектура програмного забезпечення, архітектурне мислення, поняття модульності, характеристики архітектури ПЗ, архітектурні стилі та патерни проектування. Зокрема розглядаються наступні стилі архітектури програмного забезпечення: монолітна, MVC (Model-View-Controller), Event-Driven Architecture, SOA, MSA. Особлива увага приділяється інженерним компромісам і рішенням, які виникають в процесу вибору між цими архітектурними рішеннями та надаються приклади оцінки архітектури програмного забезпечення, включаючи аналіз її структури, співвідношення сервісів до команд та практик розробки.

Мета статті

Метою статті є розробка методу оцінки, який дозволить на ранніх стадіях проектування програмного забезпечення визначити доцільність використання сервісно-орієнтованої архітектури (SOA) або мікросервісної архітектури (MSA). Визначення переваг та недоліків підходів, які використовуються сервісно-орієнтованою та мікросервісною архітектурою за допомогою порівняння ключових критеріїв. Це дозволить інженерам та розробникам обґрунтовано приймати рішення, щодо вибору архітектурного підходу, враховуючи вимоги до системи, її масштабованість, гнучкість, а також технічні та бізнесові критерії.

Основний матеріал

Монолітна архітектура є традиційним підходом до побудови програмного забезпечення, коли всі компоненти інтегровані в один єдиний виконуваний файл або додаток. Вона передбачає єдиний кодовий базис, де всі частини системи — від користувацького інтерфейсу до бази даних — об'єднані в одну цілісну структуру. Це робить систему більш керованою, оскільки всі зміни легко відслідковуються та контролюються, а процес розгортання є відносно простим. Такий підхід також спрощує тестування, оскільки весь додаток можна перевіряти як єдине ціле. Однак цей підхід має свої недоліки. Зі зростанням системи код стає важчим для підтримки, що ускладнює внесення змін і оновлень. Масштабування такої архітектури стає складним, оскільки потребує збільшення ресурсів на рівні всієї системи, навіть якщо лише одна її частина потребує додаткових ресурсів. Внесення змін в одну частину системи може вплинути на інші частини, що підвищує ризики та затримує розробку нових функцій. Таким чином, монолітна архітектура підходить для невеликих проектів або систем, що не потребують високої гнучкості та масштабованості, але може бути менш ефективною для великих і складних систем із постійно змінюваними вимогами [1, с. 11-49].

Перехід від монолітної архітектури до сервісно-орієнтованої архітектури (SOA) став важливим етапом у розвитку програмного забезпечення, оскільки дав змогу вирішити багато проблем, притаманних монолітним системам. Головними компонентами SOA архітектури є сервіси, Enterprise Service Bus (ESB), реєстр сервісів, контракти сервісів і композиція сервісів (Рис. 1). Сервіси є основними одиницями, що виконують конкретні бізнес-функції, взаємодіючи через стандартизовані інтерфейси. ESB відіграє роль центрального посередника для обміну повідомленнями між сервісами, відповідаючи за маршрутизацію та інтеграцію. Реєстр сервісів зберігає всі сервіси та їхні інтерфейси, дозволяючи іншим компонентам знаходити потрібні сервіси. Контракти сервісів визначають правила взаємодії між сервісами і клієнтами, описуючи формати даних та протоколи обміну. Композиція сервісів дає змогу об’єднувати кілька сервісів для виконання складних бізнес-процесів.

Переваги SOA включають гнучкість та адаптивність, оскільки зміни або додавання нових сервісів можуть бути виконані без впливу на інші компоненти. Масштабованість є ще однією сильною стороною SOA, дозволяючи кожному сервісу масштабуватися окремо, що забезпечує ефективне використання ресурсів. Архітектура підтримує повторне використання, оскільки сервіси можуть використовуватися в різних системах та бізнес-процесах. Також SOA забезпечує можливість інтеграції різних систем через стандартизовані протоколи взаємодії. Модульність SOA полегшує підтримку, тестування та розвиток системи. Недоліками SOA є складність впровадження, що може вимагати значних ресурсів для побудови необхідної інфраструктури. Високі витрати пов’язані із підтримкою ESB та інших компонентів, що робить архітектуру дорожчою у порівнянні з іншими підходами. Використання ESB може негативно вплинути на продуктивність, особливо у великих системах із високим трафіком даних між сервісами. Управління безпекою у SOA є складним завданням, оскільки сервіси взаємодіють

через мережу, що збільшує ризик безпекових інцидентів. Збої в роботі важливих компонентів, таких як ESB, можуть вплинути на всю систему, тому необхідно враховувати ці ризики при проектуванні архітектури.

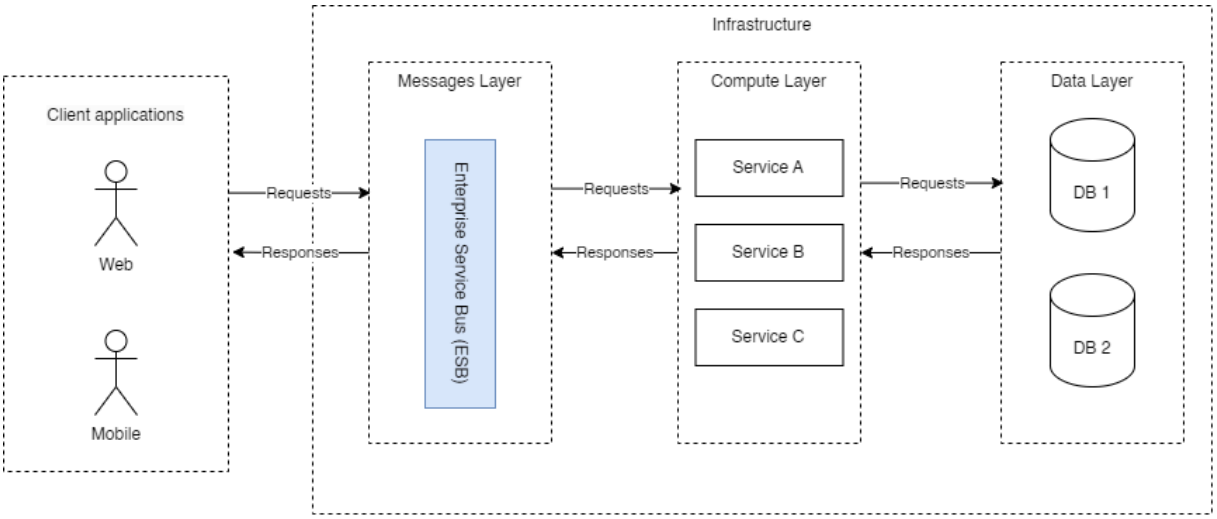

Рис. 1 – Сервіс-орієнтована архітектура (СОА)
Fig. 1 – Service-oriented architecture (SOA)

Порівняння монолітної архітектури та SOA за основними критеріями:

Таблиця 1. Порівняння монолітної та сервіс-орієнтованої архітектури (SOA)
Table 1. Comparison of monolithic and service-oriented architecture (SOA)

Критерій	**Монолітна архітектура**	**Сервіс-орієнтована архітектура (SOA)**
Структура	Єдиний кодовий базис, всі компоненти інтегровані в один додаток.	Система складається з окремих автономних сервісів.
Масштабованість	Масштабування всієї системи цілком, навіть якщо потрібно збільшити потужність тільки однієї частини.	Масштабування окремих сервісів незалежно один від одного.
Гнучкість	Зміни в одній частині можуть впливати на всю систему, що ускладнює гнучкість.	Зміни в одному сервісі не впливають на інші, що збільшує гнучкість.
Повторне використання	Компоненти складно використовувати повторно, оскільки вони тісно пов'язані між собою.	Сервіси можуть використовуватися повторно в інших системах або бізнес-процесах.
Затримка мережі (Latency)	Менша за рахунок відсутності мережевої взаємодії між компонентами.	Вища через необхідність мережевої взаємодії між сервісами.
Інтеграція	Інтеграція з іншими системами є складнішою та менш гнучкою.	Підтримка стандартизованих протоколів полегшує інтеграцію з іншими системами.

Підтримка та оновлення	Важко оновлювати, оскільки навіть незначні зміни можуть потребувати перезапуску всієї системи.	Оновлення одного сервісу не впливає на роботу інших, що спрощує підтримку.
Впровадження	Простіше впровадження для невеликих проектів або додатків.	Складніші вимоги до безпеки через взаємодію між сервісами через мережу.
Безпека	Безпека вбудована в межах одного додатку, але важче контролювати доступ до різних частин.	Складніші вимоги до безпеки через взаємодію між сервісами через мережу.
Управління відмовами	Відмова в одній частині може вплинути на всю систему.	Відмова одного сервісу не обов'язково впливає на інші, але потребує складного управління відмовами.
Модульність	Низька модульність, оскільки компоненти тісно пов’язані між собою.	Висока модульність, кожен сервіс автономний і має чітко визначені інтерфейси.
Розгортання	Розгортання всієї системи одночасно, що може бути проблемним у великих системах.	Можливість розгортання окремих сервісів незалежно один від одного.

Щодо взаємодії з рівнем даних у монолітній архітектурі взаємодія з даними відбувається через єдину спільну базу даних, що забезпечує високу швидкість доступу до даних, але призводить до тісної залежності між компонентами і складності масштабування. У SOA кожен сервіс є незалежним і має власну базу даних, а взаємодія з іншими сервісами відбувається через стандартизовані інтерфейси API. Це дозволяє кращу масштабованість і гнучкість, але ускладнює управління транзакціями та узгодженістю даних [1, с. 100-163].

Як висновок, монолітна архітектура добре підходить для невеликих або середніх проектів, де важлива швидкість розробки і простота управління системою. Але коли система зростає та необхідне обслуговування більшої кількості користувачів виникають проблеми з масштабуванням та підтримкою. SOA є еволюційним рішенням цих викликів та адресує їх краще ніж монолітний підхід, однак складність впровадження, управління розподіленими транзакціями та забезпечення узгодженості даних може стати проблемою. Також SOA вимагає більш складної інфраструктури та управління нею.

SOA це чудове архітектурне рішення для побудови середніх та великих додатків, саме завдяки використанню розподіленню функціональних вимог між сервісами та використанням модульності. Але коли виникає необхідність в ефективному, щодо використання ресурсів інфраструктури, масштабуванні або прискоренні розробки додатку, чи його компонентів, сервісної архітектури стає недостатньо. Еволюційно це призвело до появи мікросервісної архітектури (MSA), яка орієнтується на ще більш менший масштаб сервісів та їх функціональних обов’язків. Перехід від сервісно-орієнтованої архітектури (SOA) до мікросервісної архітектури (MSA) відбувся як відповідь на певні обмеження SOA та зростаючі потреби сучасних систем у більшій гнучкості, незалежності та швидкості розвитку. Виділяються наступні виклики:

1. Залежність від ESB. ESB стає вузьким місцем системи при появі більшої кількості сервісів, з якими потрібно комунікувати. Це призводить до проблем з ефективністю, керованістю та масштабуванні сервісів.
2. Незалежність розгортання. Хоча SOA пропонує модульність, сервісам зазвичай потрібна певна узгодженість при розгортанні, особливо через ESB. Це обмежує незалежне

оновлення або масштабування окремих сервісів.

3. Повторне використання та складність інтеграції. При потребі в повторному використанні сервісів, виникають додаткові виклики в інтеграції та забезпеченні сумісності між різними компонентами.

Мікросервісна архітектура (Рис. 2) — це підхід до проектування програмного забезпечення, при якому додаток розбивається на невеликі, незалежні сервіси, кожен з яких виконує окрему функцію і може розвиватися, розгортатися та масштабуватися автономно. Кожен мікросервіс функціонує як окрема одиниця, яка взаємодіє з іншими мікросервісами через легковагі протоколи, такі як HTTP, або за допомогою черг повідомлень (message queues) [2, с. 1-11].

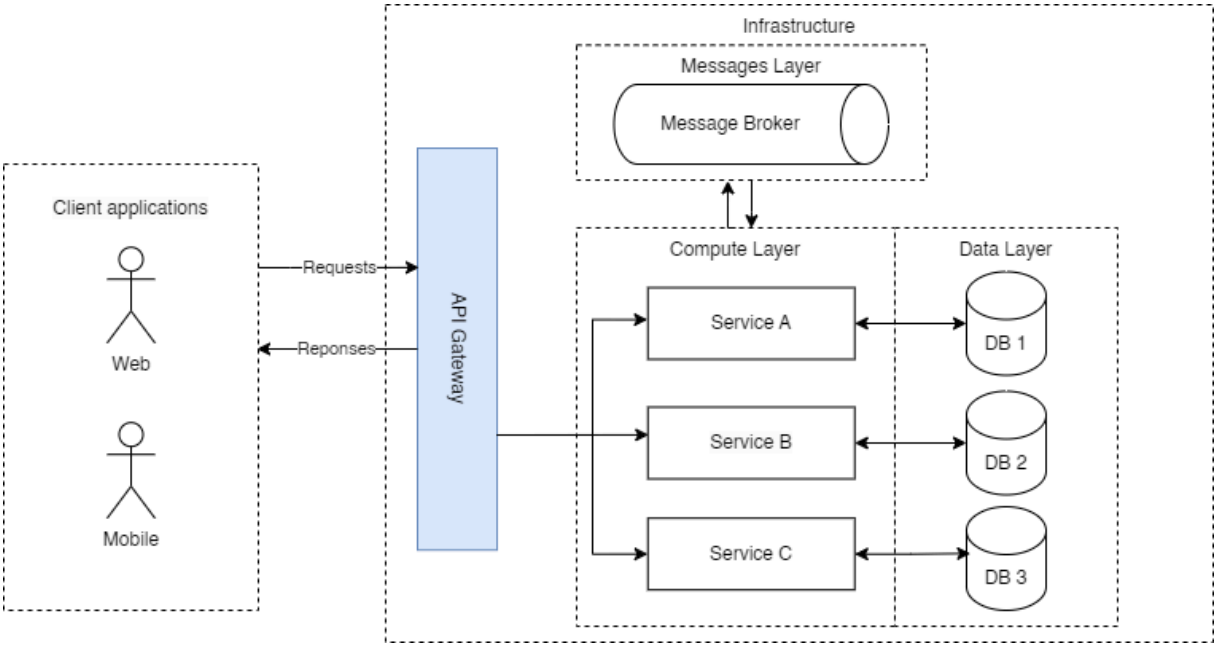

Рис. 2 – Мікросервісна архітектура (MCA)
Fig. 2. – Microservice architecture (MSA)

Основні принципи MSA [2, с. 245-250]:

1. Незалежність сервісів: Кожен мікросервіс є автономним, може працювати окремо від інших та мати власний життєвий цикл розробки.
2. Масштабованість: Мікросервіси можуть масштабуватися незалежно один від одного, що дозволяє оптимізувати використання ресурсів.
3. Мінімальні взаємозалежності: Мікросервіси повинні мати слабкі зв'язки між собою, що забезпечує легку змінюваність і оновлюваність системи.
4. Організаційна відповідність: Мікросервісна архітектура добре підходить для організацій з великими командами, які можуть працювати автономно над різними сервісами.

Порівняння MSA та SOA за основними критеріями [4, с. 235-274]:

Таблиця 2. Порівняння сервіс-орієнтованої архітектури (SOA)
та мікросервісної архітектури (MSA)
Table 2. Comparison of Service-Oriented Architecture (SOA) and Microservice Architecture (MSA)

Критерій	**Сервіс-орієнтована архітектура (SOA)**	**Мікросервісна архітектура (MSA)**
Зв'язність	Сервіси більш тісно інтегровані через ESB.	Слабко зв’язані, сервіси можуть функціонувати незалежно.
Комунікація між сервісами	Використання важких протоколів, таких як SOAP.	Використання легковагих протоколів, таких як HTTP/REST, gRPC.
Інфраструктура	Використання ESB для координації між сервісами.	Немає централізованої шини (ESB); сервіси самостійно керують взаємодією.
Масштабованість	Більш складне масштабування через централізовану структуру.	Легке масштабування кожного окремого сервісу.
Управління транзакціями	Централізоване управління через ESB.	Децентралізоване, кероване на рівні кожного мікросервісу.
Модульність	Менш модульна через інтеграцію з ESB.	Висока модульність; сервіси можуть бути легко змінені або замінені.
Швидкість розробки	Повільніший розвиток через необхідність координації між сервісами.	Швидка розробка, незалежне розгортання частин системи.
Повторне використання	Вищий рівень повторного використання сервісів.	Мікросервіси орієнтовані на вузькі завдання, що може знизити повторне використання.
Масштаб системи	Підходить для середніх і великих систем, але з меншою гнучкістю.	Найкраще підходить для великих, складних систем з частими оновленнями.

Виділяються наступні переваги MSA порівняно з SOA:

1. Децентралізація. Мікросервіси не залежать від централізованої шини (ESB) і спілкуються один з одним через легкі протоколи, такі як HTTP/REST або gRPC. Це дозволяє уникати вузьких місць, характерних для ESB в SOA, та знижує залежність між командами.
2. Масштабування. Кожен мікросервіс може бути масштабований окремо, що робить MSA більш гнучкою та легкою у підтримці. Це особливо корисно для систем з нерівномірним навантаженням, де окремі сервіси можуть мати різні вимоги до продуктивності.
3. Швидке розгортання. Кожен окремий мікросервіс є більш легким, можна розгорнути швидше. Це вирішує проблему координації, яка існувала в SOA, і дозволяє більш часті релізи та швидке реагування на зміни в бізнес-вимогах. Але не вирішує проблеми залежності комунікації між окремими сервісами.
4. Гнучкість технологій. У MSA кожен сервіс може використовувати різні технології та бази даних, що дає більшу свободу вибору інструментів для розв'язання конкретних завдань. У SOA сервіси часто обмежувалися єдиною технологічною платформою або потребували стандартизації для інтеграції.

Мікросервісна архітектура (MSA) стає кращим вибором у випадках, коли система повинна бути високо-гнучкою, потребує швидкого розвитку, незалежного розгортання та масштабування окремих сервісів, і готова до складнішої інфраструктури для підтримки розподілених компонентів. Ця архітектура добре підходить для сучасних хмарних додатків, з великими командами розробників, де кожна команда працює над окремим компонентом системи. Але водночас з’являються наступні фактори, які слід враховувати при обранні MSA:

1. Складність управління інфраструктурою: У порівнянні з монолітною архітектурою або SOA, MSA вимагає більш складної інфраструктури для оркестрації мікросервісів, управління контейнерами, моніторингу та налаштування мережевих взаємодій між сервісами. Це може вимагати значних витрат на підтримку та впровадження автоматизованих систем управління (Kubernetes).
2. Збільшення складності комунікації: У MSA сервіси взаємодіють через мережеві протоколи, що призводить до необхідності ефективного управління мережею, помилками комунікації та розподіленими транзакціями. Це може створити проблеми з латентністю та призвести до складних сценаріїв збоїв (наприклад, якщо один сервіс стає недоступним).
3. Проблеми з узгодженістю даних: Оскільки кожен мікросервіс може мати власну базу даних, управління узгодженістю даних між сервісами може бути складним. Мікросервіси не мають єдиної транзакційної системи, тому необхідно впроваджувати складні механізми узгодження (патерн SAGA, та інші) для забезпечення консистентності даних.
4. Збільшення обсягу мережевого трафіку: Кожен мікросервіс взаємодіє з іншими через мережу, що може призвести до значного збільшення мережевого трафіку. Це особливо критично для додатків, що вимагають високої пропускної здатності і швидкої обробки запитів.
5. Моніторинг і налагодження: Мікросервісна архітектура ускладнює моніторинг та налагодження системи. Необхідно відстежувати стан багатьох окремих компонентів, розподілених у різних середовищах, що вимагає впровадження спеціалізованих інструментів для логування, трасування запитів (distributed tracing) та моніторингу продуктивності.
6. Залежності між сервісами: Попри автономність мікросервісів, в складних системах може виникати сильна залежність між ними. Коли один сервіс виходить з ладу, це може призвести до каскадних збоїв у всій системі. Для запобігання таких проблем потрібні механізми резервування та стійкості до збоїв (circuit breaker patterns).
7. Проблеми з тестуванням: Тестування системи з багатьма мікросервісами може бути складнішим, оскільки потрібно враховувати різні сценарії інтеграції між ними, а також їхню взаємодію з іншими сервісами. Для цього потрібні комплексні методики автоматизованого тестування, що підвищує складність процесу тестування.
8. Витрати на підтримку інфраструктури: Використання мікросервісів може призвести до підвищених витрат на інфраструктуру, оскільки кожен сервіс потребує окремих ресурсів для розгортання, масштабування та моніторингу. Це також ускладнює управління обчислювальними та мережевими ресурсами.
9. Підвищені вимоги до команд розробки: Мікросервісна архітектура вимагає від команд розробки знання багатьох різних технологій, мікросервісних патернів та методів розподіленого програмування.

Висновки

Обидва підходи до побудови архітектури програмного забезпечення мають свої переваги й недоліки. Вибір між ними залежить від потреб бізнесу, специфіки проекту та вимог, щодо можливостей масштабування системи. Він повинен базуватися на аналізі конкретних потреб проекту, технічних обмежень та стратегічних цілей проекту або організації. SOA, краще підходить для більш структурованих систем, де важливо підтримувати стандартизацію, повторне використання сервісів та немає частих оновлень. Також для додатків, що потребують інтеграції зі старими системами і централізованого управління сервісами. MSA є ідеальним вибором для організацій, які прагнуть створювати високо-динамічні, масштабовані системи з частими оновленнями та гнучкістю в розробці, використовуючи хмарну чи подіну інфраструктуру.

Список літератури

1. Newman S. (2020) Monolith to Microservices: Evolutionary Patterns to Transform Your Monolith 1st Edition. O'Reilly Media, 270 p. URL: https://dl.ebooksworld.ir/books/Monolith.to.Microservices.Sam.Newman.OReilly.9781492047841.EBooksWorld.ir.pdf
2. Newman S. (2015) Building Microservices: Designing Fine-Grained Systems 1st Edition. O'Reilly Media, 278 p.
3. Richardson C. Microservice Architecture. microservices.io. URL: https://microservices.io/
4. Richards M., Ford N. (2021) Fundamentals of Software Architecture: An Engineering Approach. O'Reilly Media, 419 p. URL: https://mrce.in/ebooks/Software-Fundamentals%20of%20Software%20Architecture.pdf
5. Mitra R., Nadareishvili I. (2020) Microservices: Up and Running: A Step-by-Step Guide to Building a Microservices Architecture 1st Edition. O'Reilly Media, 316 p.
6. C. Martin. R. (2018) Clean Architecture: A Craftsman's Guide to Software Structure and Design. Pearson, 432 p. URL: https://agorism.dev/book/software-architecture/%28Robert%20C.%20Martin%20Series%29%20Robert%20C.%20Martin%20-%20Clean%20Architecture_%20A%20Craftsman%E2%80%99s%20Guide%20to%20Software%20Structure%20and%20Design-Prentice%20Hall%20%282017%29.pdf
7. О.Ю. Ільїн та ін. Наукові записки УНДІЗ. Особливості розгортання мікросервісних додатків за допомогою системи керування контейнерами. 2019. №4(56). С. 49-60. DOI: https://doi.org/10.31673/2518-7678.2019.044960

DETERMINATION OF SOFTWARE ARCHITECTURE (SOA) AND MICROSERVICE ARCHITECTURE (MSA) USAGE CRITERIA

Oleh Siedashev[1], postgraduate student of the Faculty of Mathematics and Computer Science;
e-mail: oleh.siedashev@student.karazin.ua; ORCID: https://orcid.org/0009-0001-3259-3141
[1]*V. N. Karazin Kharkiv National University, Ukraine*

Manuscript was received November 7, 2024; Received after review December 8, 2024;
Accepted December 22, 2024

Abstract. In modern software development, one of the key tasks is to choose the appropriate architecture for the system in the early stages of its design. This article examines two popular software architecture approaches: service-oriented architecture (SOA) and microservice architecture (MSA). Based on the analysis of architectural features, advantages and disadvantages of these approaches, the criteria that influence the choice of an architectural model depending on the specifics of the system are

investigated. Microservice architecture, due to its independence and the possibility of rapid scaling, is better suited for dynamic systems with high requirements for flexibility. Service-oriented architecture, on the contrary, is focused on centralized management of services through ESB (Enterprise Service Bus) and provides better opportunities for integration and reuse of components in large corporate systems that do not require frequent changes in functionality. The main focus of the article is the development of an evaluation method that will allow software developers and system engineers to determine at the early design stages which of the architectures, SOA or MSA, is more appropriate to use for a specific system. Taking into account various technical and requirements, the method identifies key criteria that should be paid attention to when choosing an application software architecture.

Keywords: *information systems, software architecture, SOA, MSA*

References

1. Newman S. (2020) Monolith to Microservices: Evolutionary Patterns to Transform Your Monolith 1st Edition. O'Reilly Media, 270 p. URL: https://dl.ebooksworld.ir/books/Monolith.to.Microservices.Sam.Newman.OReilly.9781492047841.EBooksWorld.ir.pdf
2. Newman S. (2015) Building Microservices: Designing Fine-Grained Systems 1st Edition. O'Reilly Media, 278 p.
3. Richardson C. Microservice Architecture. microservices.io. URL: https://microservices.io/
4. Richards M., Ford N. (2021) Fundamentals of Software Architecture: An Engineering Approach. O'Reilly Media, 419 p. URL: https://mrce.in/ebooks/Software-Fundamentals%20of%20Software%20Architecture.pdf
5. Mitra R., Nadareishvili I. (2020) Microservices: Up and Running: A Step-by-Step Guide to Building a Microservices Architecture 1st Edition. O'Reilly Media, 316 p.
6. C. Martin. R. (2018) Clean Architecture: A Craftsman's Guide to Software Structure and Design. Pearson, 432 p. URL: https://agorism.dev/book/software-architecture/%28Robert%20C.%20Martin%20Series%29%20Robert%20C.%20Martin%20-%20Clean%20Architecture_%20A%20Craftsman%E2%80%99s%20Guide%20to%20Software%20Structure%20and%20Design-Prentice%20Hall%20%282017%29.pdf
7. O.Yu. Ilyin et al. (2019) Scientific notes of the UNDIZ. *Features of deploying microservice applications using a container management system*. №4(56). P. 49-60. DOI: https://doi.org/10.31673/2518-7678.2019.044960 [in Ukrainian]

ISSN 2519-2310 (Online)

КОМП'ЮТЕРНІ НАУКИ ТА КІБЕРБЕЗПЕКА, 2024, № 2 (26)

COMPUTER SCIENCE AND CYBERSECURITY, 2024, Iss. 2 (26)

DOI: https://doi.org/10.26565/2519-2310-2024-2-05
УДК 004.77

ДОСЛІДЖЕННЯ ПОТОЧНОГО СТАНУ ТА ПЕРСПЕКТИВИ ЗАСТОСУВАННЯ ШТУЧНОГО ІНТЕЛЕКТУ В КІБЕРБЕЗПЕЦІ

Юрій Голіков[1], e-mail: yuriy@devbrother.com
Єлизавета Острянська[2], молодший науковий співробітник, e-mail: antelizza@gmail.com,
ORCID: https://orcid.org/0000-0003-1412-8470
[1]*DevBrother tech company, США*
[2]*Харківський національний університет імені В.Н. Каразіна,*
майдан Свободи, 4, Харків, 61022, Україна
Рукопис надійшов 1 листопада 2024 р. Отримано після рецензування 2 грудня 2024 р.
Прийнято 23 грудня 2024 р.

Анотація: В сучасному світі, з розвитком нових технологій, штучний інтелекту (ШІ) у сфері кібербезпеки є невід'ємною складовою, тому вивчення його переваг, ризиків та можливі сценарії використання є актуальною темою дослідження. В сучасному цифровому середовищі, де кіберзагрози стають дедалі складнішими, впровадження технологій ШІ значно підвищує ефективність систем захисту, дозволяючи автоматизувати виявлення та реагування на атаки. Тому було розглянуто основні напрями застосування ШІ в кібербезпеці, зокрема виявлення загроз, аналіз шкідливого програмного забезпечення, посилення криптографії, захист від фішингових атак та прогнозування атак. Одним із ключових аспектів є інтеграція ШІ в системи управління інформаційною безпекою (SIEM), які аналізують величезні обсяги даних та допомагають виявляти аномалії. Такі системи значно знижують навантаження на команди безпеки та підвищують точність і швидкість реагування. Окрему увагу приділено аналізу сучасних антивірусних рішень, що використовують штучний інтелект, а саме Microsoft Defender for Endpoint та Darktrace. Вони базуються на алгоритмах поведінкового аналізу та машинного навчання, що дозволяє ефективніше виявляти складні загрози та запобігати інцидентам. Microsoft Defender забезпечує високий рівень захисту кінцевих точок. Натомість Darktrace використовує самонавчальні моделі для аналізу мережевого трафіку, що дозволяє виявляти загрози нульового дня та внутрішні загрози в організаціях. Розглянуто основні ризики, пов'язані із використанням ШІ у кіберзлочинності. ШІ активно застосовується зловмисниками для автоматизації атак, що значно підвищує їхню ефективність і складність виявлення. Розглянуто основні типи загроз на основі ШІ, а саме такі атаки, як атаки отруєння даних, атаки ухилення, атаки швидкого впровадження та соціальна інженерія на основі ШІ. Для зменшення ризиків пропонується розробка стійких моделей ШІ, що менше піддаються атакам, підвищення прозорості алгоритмів, а також впровадження міжнародних стандартів регулювання ШІ, чим активно займаються великі команди дослідників, серед яких і NIST. Також рекомендується підвищення обізнаності користувачів та спеціалістів з кібербезпеки, оскільки людський фактор залишається однією з найбільших вразливостей систем. У підсумку, наголошується, що штучний інтелект є ключовим фактором розвитку кібербезпеки, який може значно покращити захист інформації та критичних систем. Водночас, без належного регулювання та захисних заходів, ШІ може стати потужним інструментом для зловмисників, що створює нові виклики

для безпеки в цифрову епоху. Баланс між інноваціями, етичними нормами та безпекою стане ключовим фактором у формуванні стратегії ефективного використання ШІ у майбутньому.

Ключові слова: *штучний інтелект, кібербезпека, машинне навчання, SIEM, IDS, антивірус на основі ШІ*

Як цитувати: Голіков Ю., Острянська Є. Дослідження поточного стану та перспективи застосування штучного інтелекту в кібербезпеці. *Комп’ютерні науки та кібербезпека.* 2024; № 2(26): С. 51–65. https://doi.org/10.26565/2519-2310- 2024-2-05

In cites: Golikov Yu., Ostrianska Ye. (2024). Research on the current state and prospects of the application of artificial intelligence in cybersecurity. *Computer Science and Cybersecurity.* 2(26): 51–65. https://doi.org/10.26565/2519-2310-2024-2-05 (in Ukrainian)

1. Вступ

Штучний інтелект (ШІ) вже більше десятиліття змінює простір кібербезпеки, завдяки машинному навчанню (ML), яке прискорює виявлення загроз і виявляє аномальну поведінку користувачів і об’єктів. І тому ШІ поступово стає невід’ємною частиною сучасних систем кібербезпеки, допомагаючи ідентифікувати загрози, автоматизувати процеси та підвищувати рівень захисту інформації.

Одним із важливих викликів у використанні ШІ для кібербезпеки є формування довіри. Дані, які використовуються для навчання моделей AI/ML, керують виходом моделей. Якщо навчальні дані не відображають «реальний світ», модель може спотворити її здатність забезпечувати очікувані результати. Деякі дані, такі як інформація про загрози, «хороші» та «погані» характеристики файлів, індикатори компрометації тощо, є для всіх.

Ще одна серйозна проблема – безпека даних. Важливо визначити та контролювати, якими навчальними даними можна ділитися, а які дані залишаються секретними в середині організацій. У чужих руках ці дані можуть допомогти зловмисникам у їхніх атаках, щоб підірвати здатність AI/ML ідентифікувати їхні файли, програми та поведінку як недійсні. У зв’язку з цим державам і комерційним структурам необхідно розробити нормативні акти, стандарти та найкращі практики, щоб запобігти новим загрозам із застосуванням ШІ.

Так, наприклад, NIST [17, 18] вже очолює та бере участь у розробці технічних стандартів, включаючи міжнародні стандарти, які сприяють інноваціям і громадській довірі до систем, які використовують ШІ. Широкий спектр стандартів для даних штучного інтелекту, продуктивності та управління є – і все більше буде – пріоритетом для надійного та відповідального ШІ.

NIST розробив план глобальної взаємодії з просуванням і розробкою стандартів ШІ. Мета полягає в тому, щоб стимулювати розробку та впровадження консенсусних стандартів, пов’язаних зі штучним інтелектом, співпрацю та координацію, а також обмін інформацією. Відображаючи внесок державного та приватного секторів. 26 липня 2024 року, після розгляду коментарів громадськості щодо плану проєкту, NIST опублікував План глобальної взаємодії зі стандартами ШІ [17].

Розвиток технологій ШІ приніс не лише нові можливості, але й нові ризики та небезпеку. Спільнота науковців та дослідників по всьому світу стурбована та попереджають про потенційні проблеми поширення ШІ. Зокрема, нещодавній звіт Національного центру кібербезпеки Великої Британії (NCSC) [14] застерігає, що протягом найближчих двох років технології ШІ напевно збільшать динаміку кібератак та посилять їхній вплив на існуючі

криптосистеми та засоби захисту інформації. За оцінками Центру [14], ШІ відкриває широкі можливості у цьому напрямку навіть для тих криптоаналітиків, що не мають відповідних технічних навичок. І вони стверджують, що вже після 2025 року і далі, коли ШІ пройде достатньо успішні навчання, ШІ, майже напевно покращиться, забезпечуючи швидші й точніші кібероперації.

У найближчі роки роль ШІ лише зростатиме. Наприклад, ШІ здатен виявляти загрози у реальному часі, тобто може аналізувати величезні обсяги даних, виявляючи аномальні дії та потенційні загрози швидше, ніж це здатна зробити людина. Також алгоритми ШІ можуть не тільки ідентифікувати загрози, а й миттєво блокувати шкідливий трафік або змінювати правила доступу до мережі. Що також важливо, автоматизовані рішення допомагають уникнути помилок, які можуть виникати через людську неуважність або недостатню кваліфікацію.

Тому, метою цієї статті є дослідження поточного стану використання ШІ в сфері кібербезпеки, а також загрози та ризики пов’язані з його використанням. В статті розглядаються сучасні антивірусні рішення, найпопулярніші атаки з використанням ШІ та методи протидії. А також перспективи використання ШІ та доведення того, що в сучасному світі ШІ є невід’ємною частиною кіберзахисту.

2. Основні напрямки використання ШІ в сфері кібербезпеки

Нижче розглянемо шість основних можливих варіантів використання ШІ в сфері кібербезпеки (рис. 1):

1. Автоматизоване виявлення загроз та реагування. Традиційні системи кібербезпеки потребують постійного моніторингу та аналізу великих обсягів даних, що складно реалізувати вручну. ШІ може виконувати наступні задачі:

- Виявляти аномальну активність у мережі за допомогою алгоритмів машинного навчання.
- Аналізувати поведінку користувачів та пристроїв для виявлення підозрілих дій.
- Автоматично реагувати на загрози, блокуючи потенційно небезпечні дії без втручання людини.

Прикладом таких систем є, наприклад, система SIEM (Security Information and Event Management) з підтримкою ШІ, що аналізує події в реальному часі та попереджає про можливі атаки. Більш детально системи SIEM будуть розглянуті в розділі 3 цієї статті.

2. Використання ШІ в аналізі шкідливого програмного забезпечення. Традиційні антивіруси використовують підписи для виявлення шкідливого ПЗ, що робить їх менш ефективними проти нових атак. Натомість ШІ дозволяє:

- Використовувати поведінковий аналіз для виявлення нових вірусів та троянських програм.
- Розпізнавати різні види атак, які змінюють свій код для обходу антивірусів.
- Автоматично створювати та оновлювати бази загроз без необхідності ручного внесення змін.

Наприклад, зараз набувають популярності ШІ-антивіруси, такі як Microsoft Defender ATP або Darktrace, які аналізують поведінку програм і виявляють загрози без використання втручання людини. Більш детально дані методи ми розглядаємо в розділі 4 цієї статті.

3. Підсилення криптографії та постквантова безпека. Як відомо, з розвитком квантових комп’ютерів традиційні криптографічні алгоритми можуть стати вразливими. ШІ в свою чергу може допомогти у:

- Створенні адаптивних криптографічних рішень, які зможуть змінювати ключі та алгоритми в реальному часі.
- Оптимізації шифрування та розшифрування, що зробить криптографічні системи ефективнішими.
- Розробці нових постквантових криптографічних алгоритмів, які будуть стійкими до атак квантових комп’ютерів.

Тому ШІ активно використовується для тестування нових стійких шифрів. Оптимізація шифрування та управління ключами – алгоритми машинного навчання можуть покращувати ефективність криптографічних протоколів, забезпечуючи швидше та надійніше шифрування. Наприклад, NIST активно досліджує алгоритми шифрування та підписів постквантової криптографії [1], а ШІ може прискорити їх тестування та впровадження [18].

4. ШІ в протидії фішинговим атакам. Фішинг – це одна з найпоширеніших кіберзагроз, яка стає все складнішою. ШІ може допомогти у:

- Автоматичному розпізнаванні підозрілих електронних листів та URL-адрес.
- Аналізі мови та структури повідомлень для виявлення шахрайських схем.
- Покращенні механізмів аутентифікації, наприклад, через поведінкову біометрію або динамічні капчі.

Одним з найвідоміших прикладів використання ШІ в протидії фішингу є Google, що використовує ШІ в Gmail для фільтрації фішингових листів, що дозволяє блокувати понад 99% шкідливих повідомлень.

5. Використання ШІ для прогнозування атак. Штучний інтелект може допомогти не тільки реагувати на атаки, але й передбачати їх, аналізуючи величезні масиви даних:

- Використання аналізу великих даних для ідентифікації трендів у кібератаках.
- Створення моделей-прогнозів, які дозволяють передбачати потенційні вразливості систем.
- Автоматичний аналіз темної мережі (Dark Web) для виявлення нових загроз та хакерських інструментів.

Наприклад, IBM Watson for Cyber Security [7] використовує когнітивний аналіз для ідентифікації нових загроз, аналізуючи мільйони статей, форумів та повідомлень у даркнеті.

Рис. 1 – Використання ШІ в кібербезпеці
Fig. 1 – Using AI in cybersecurity

3. Захист на основі ШІ та підвищення ефективності кібербезпеки

Системи виявлення вторгнень (IDS) на базі штучного інтелекту (ШІ) є сучасними інструментами кібербезпеки, які використовують можливості машинного навчання та аналізу даних для ідентифікації та запобігання несанкціонованим діям у мережах та системах. Вони здатні адаптуватися до нових загроз, аналізуючи великі обсяги даних та виявляючи аномалії, що можуть свідчити про потенційні атаки.

Існують два основні методи, які використовуються в IDS:

1) Виявлення на основі відомих алгоритмів. Цей метод передбачає порівняння вхідного трафіку з базою даних відомих атак. Якщо знайдено збіг, система генерує сповіщення. Однак цей підхід обмежений у виявленні нових або модифікованих атак, які ще не мають відповідних алгоритмів.

2) Виявлення на основі аномалій: Цей метод використовує моделі нормальної поведінки системи. Відхилення від цієї норми розглядаються як потенційні загрози. ШІ відіграє ключову роль у побудові та оновленні таких моделей, що дозволяє виявляти навіть невідомі раніше атаки.

Зрозуміло, що інтеграція штучного інтелекту в системи виявлення вторгнень значно підвищує ефективність кібербезпеки, дозволяючи виявляти та запобігати як відомим, так і новим загрозам. Однак для максимального використання потенціалу таких систем необхідно враховувати можливі обмеження та етичні аспекти їх застосування, бо, наприклад, використання ШІ може порушувати питання конфіденційності, особливо якщо системи аналізують персональні дані без належного контролю.

4. Система SIEM з підтримкою ШІ

Управління інформацією про безпеку та подіями, або SIEM (Security Information and Event Management), – це рішення безпеки, яке допомагає організаціям розпізнавати й усувати потенційні загрози та вразливості безпеки до того, як у них з’явиться шанс порушити бізнес-операції [7].

Системи SIEM допомагають групам безпеки підприємства виявляти аномалії поведінки користувачів і використовувати штучний інтелект (ШІ) для автоматизації багатьох ручних процесів, пов’язаних із виявленням загроз і реагуванням на інциденти. Наразі, найвідомішими сучасними системами SIEM є такі програмні забезпечення: Splunk Enterprise Security (Splunk), Elastic Security, IBM QRadar SIEM, Wazuh SIEM, Microsoft Sentinel.

Початкові платформи SIEM були інструментами керування журналами. Вони поєднали функції керування інформацією про безпеку (SIM) і керування подіями безпеки (SEM). Ці платформи забезпечували моніторинг і аналіз подій, пов’язаних із безпекою, у реальному часі. Термін SIEM було введено у 2005 році для поєднання технологій SIM і SEM.

Крім того, вони полегшили відстеження та реєстрацію даних безпеки для цілей відповідності або аудиту. Протягом багатьох років програмне забезпечення SIEM розвивалося, щоб включити аналітику поведінки користувачів і об’єктів, а також інші передові засоби аналітики безпеки, штучний інтелект і можливості машинного навчання для виявлення аномальної поведінки та індикаторів передових загроз. Сьогодні SIEM став основним елементом сучасних центрів безпеки (SOC) для моніторингу безпеки та управління відповідністю.

Етапи роботи систем SIEM (рис. 2) можна поділити наступним чином [6]:

- Збір даних – усі джерела інформації про мережеву безпеку, наприклад сервери, операційні системи, брандмауери, антивірусне програмне забезпечення та системи запобігання вторгненням, налаштовані на передачу даних про події в інструмент SIEM.

Більшість сучасних інструментів SIEM використовують агенти для збору журналів подій із систем підприємства, які потім обробляються, фільтруються та надсилаються до SIEM. Деякі SIEM дозволяють збирати дані без агентів. Наприклад, Splunk [8] пропонує безагентний збір даних у Windows за допомогою WMI.

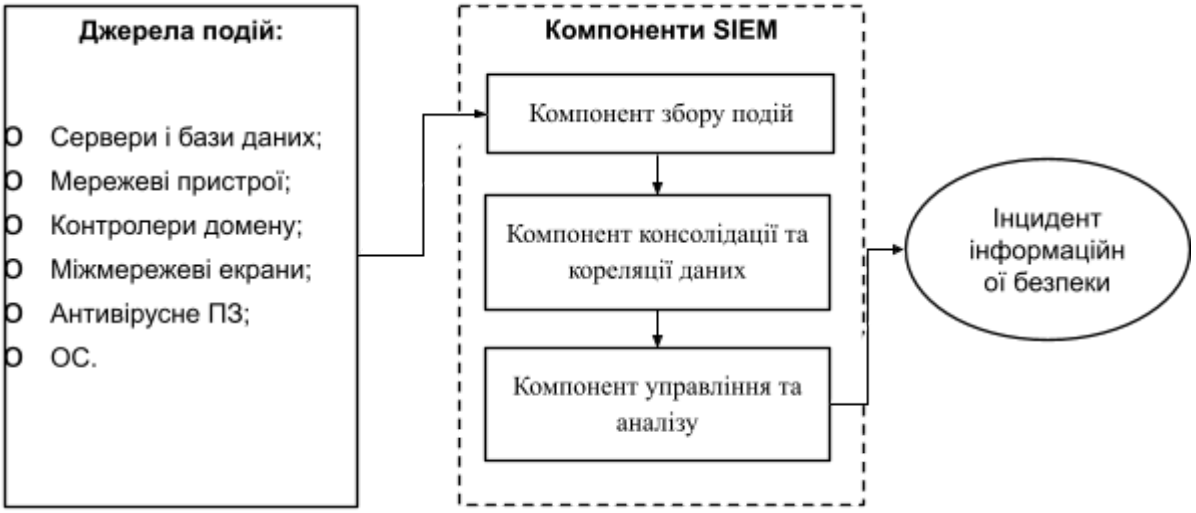

Рис. 2 – Схема роботи системи SIEM
Fig. 2 – SIEM system operation diagram

Системи AI SIEM починаються з агрегування даних із різних джерел, таких як мережеві пристрої, сервери, бази даних і програми. Після прийому необроблені дані перетворюються в стандартизований формат, що забезпечує послідовність і точність аналізу даних незалежно від джерела. AI та ML значно автоматизують ці процеси, підвищуючи швидкість і інтелект, з якими дані безпеки агрегуються та нормалізуються, скорочуючи ручні зусилля та час [9].

- Політики – адміністратор SIEM створює профіль, який визначає поведінку систем підприємства як за звичайних умов, так і під час попередньо визначених інцидентів безпеки. SIEM надають стандартні правила, сповіщення, звіти та інформаційні панелі, які можна налаштовувати відповідно до конкретних потреб безпеки.
- Консолідація та кореляція даних – рішення SIEM консолідують, аналізують та аналізують файли журналів. Потім події класифікуються на основі необроблених даних і застосовуються правила кореляції, які поєднують окремі події даних у значущі проблеми безпеки.
- Системи AI SIEM використовують прогнозовану аналітику для прогнозування потенційних майбутніх загроз шляхом аналізу архівних даних безпеки та виявлення закономірностей. Ця можливість дозволяє організаціям проактивно захищати свої системи, а не реагувати на загрози, щойно вони виникають. Ця база знань дозволяє моделям штучного інтелекту, які є основою рішення, створювати дедалі точніші реакції безпеки та підходи до запобігання інцидентам із часом і накопиченням нових даних.

Постійне вивчення проблем минулого підвищує точність і надійність систем SIEM на основі штучного інтелекту проти дедалі потужніших кіберзагроз. Зрештою, SIEM на основі штучного інтелекту об’єднує різні компоненти, такі як ШІ, ML, глибоке навчання, NLP і UEBA, які розширюють можливості традиційної SIEM. Ця інтеграція веде до більш розумних,

ефективних і проактивних заходів кібербезпеки, що має вирішальне значення в середовищі кіберзагроз, що постійно змінюється [9].

Сповіщення – якщо подія або набір подій запускає правило SIEM, система сповіщає персонал служби безпеки. У разі виявлення загрози ШІ надає системам SIEM можливість автоматизувати частини процесу реагування на інцидент. Це включає в себе автоматичний запуск сповіщень, впровадження попередньо визначених дій відповіді або організацію складних робочих процесів відповіді. Одним із таких прикладів є автоматизований динамічний робочий процес, де робочий процес, що встановлюється після потенційної загрози, адаптований до відповідної загрози.

Без допомоги SIEM кількість часу, який знадобиться аналітикам безпеки для достовірного виявлення підозрілих дій шляхом кореляції журналів між різними типами пристроїв, буде дуже великим з огляду на складність більшості мереж. Рідко вдається вчасно виявити та відреагувати на будь-яку загрозу чи атаку на їхні інфраструктури, щоб запобігти будь-якій шкоді. Крім того, рішення SIEM може розширити можливості використання зібраної інформації [3].

Очевидно, що ШІ ставатиме все більш важливим у майбутньому SIEM, оскільки когнітивні можливості покращуватимуть здатність системи приймати рішення. Це також дозволить системам адаптуватися та розвиватися зі збільшенням кількості кінцевих точок. Як IoT, хмарні обчислення, мобільні та інші технології збільшують обсяг даних, який повинен споживати інструмент SIEM. ШІ пропонує потенціал для рішення, яке підтримує більше типів даних і комплексне розуміння загроз у їх розвитку.

5. Антивірусні рішення на основі ШІ

Штучний інтелект відіграє ключову роль у сучасних антивірусних рішеннях, таких як, наприклад, Microsoft Defender Advanced Threat Protection (ATP) [11] та Darktrace [12]. Ці системи використовують можливості ШІ для покращення виявлення та реагування на кіберзагрози.

Microsoft Defender ATP [11] є корпоративною платформою безпеки кінцевих точок, розробленою для запобігання, виявлення, дослідження та реагування на загрози. Вона інтегрує сенсори поведінки кінцевих точок, аналітику безпеки в хмарі та розвідку загроз для забезпечення комплексного захисту. Сенсори, вбудовані в Windows 10, збирають та обробляють поведінкові сигнали, які надсилаються до ізольованого хмарного середовища Microsoft Defender для подальшого аналізу.

Основні можливості Microsoft Defender включають [11]:

- Зниження поверхні атаки: мінімізація векторів атак для зменшення можливостей проникнення зловмисників.
- Захист нового покоління: використання передових методів машинного навчання для виявлення складних атак.
- Виявлення та реагування на кінцевих точках (EDR): забезпечення глибокого аналізу та візуалізації загроз у реальному часі.
- Автоматизоване розслідування та реагування: зменшення навантаження на команди безпеки шляхом автоматизації процесів.

Серед переваг рішення Microsoft Defender можна виділити те, що система має глибоку інтеграцію з Windows, Microsoft 365 та Azure, але в той же час це можна розглядати і як недолік, оскільки вона не покриває весь мережевий трафік так ефективно, як, наприклад, Daktrace [12], що буде розглянуто далі. Серед переваг також – автоматичне дослідження загроз та вбудовані

можливості SIEM та SOAR через Microsoft Sentinel. Далі розглянемо рішення Daktrace.

Darktrace [12] – це компанія з кібербезпеки, яка використовує штучний інтелект та машинне навчання для виявлення кібератак та вразливостей у комп'ютерних системах. Її технологія спрямована на швидке та ефективне виявлення загроз, використовуючи поведінковий аналіз для ідентифікації аномалій, що можуть свідчити про атаки.

Darktrace застосовує самонавчальний ШІ для забезпечення проактивного кіберзахисту. Основний принцип роботи Darktrace – це використання машинного навчання для аналізу поведінки користувачів, пристроїв і мережевих потоків. Її технологія здатна:

- Виявляти загрози в реальному часі: аналізуючи поведінку мережі та користувачів для виявлення аномалій. І що не менш важливо – виявляти нові, невідомі загрози, тобто без використання відомих алгоритмів або баз відомих вірусів.
- Автономно реагувати на загрози: здійснювати автоматичні дії для нейтралізації атак без втручання людини.
- Забезпечувати захист у різних середовищах: мережах, електронній пошті, хмарних сервісах, операційних технологіях та кінцевих точках, а також зовнішніх платформ, наприклад, AWS, Azure, Google Cloud.

Таким чином, бачимо, що обидві системи і Microsoft Defender for Endpoint [11], і Darktrace [12] використовують штучний інтелект (ШІ) для покращення кібербезпеки, але вони мають різні підходи та сфери застосування. Розглянемо їх у порівняльному аналізі в таблиці 1.

Таблиця 1 – Порівняльний аналіз сучасних антивірусних рішень на основі ШІ
Table 1 – Comparative analysis of modern AI-based antivirus solutions

Характеристика	**Microsoft Defender ATP**	**Darktrace**
Тип рішення	Endpoint Detection and Response (EDR), SIEM/SOAR	Network Detection and Response (NDR), Автономна безпека ШІ
Основний підхід	Використання поведінкового аналізу та баз загроз для захисту кінцевих точок	Самонавчальний ШІ для аналізу аномалій у мережі, електронній пошті та хмарних сервісах
Основні можливості	Виявлення загроз на кінцевих точках, аналіз атак, автоматизоване реагування	Виявлення мережевих загроз, автономне реагування через Darktrace Antigena
Захист від програм-вимагачів	Блокує відомі атаки, аналізує поведінку, зупиняє підозрілі процеси	Виявляє аномальну активність у реальному часі, блокує шкідливі дії на рівні мережі
Фокус на загрози	Віруси, шкідливе ПЗ, експлойти, атакуючі скрипти, компрометація облікових записів	Аномалії в мережевому трафіку, внутрішні загрози, атаки нульового дня
Інтеграція	Глибока інтеграція з екосистемою Microsoft 365, Azure, Defender XDR	Підтримка гібридних середовищ: мережа, електронна пошта, хмара, промислові системи
Хмарна підтримка	Microsoft 365, Azure	AWS, Google Cloud, Azure, локальні мережі

Розглянемо для кого ж найкраще підходить кожне рішення.

Microsoft Defender for Endpoint:

- Великі та середні компанії, які працюють в екосистемі Microsoft.
- Організації, що шукають потужний захист кінцевих точок із вбудованою SIEM-аналітикою.
- Компанії з IT-командою, яка готова керувати безпекою через Microsoft Security Center.

Darktrace краще буде інтегрувати у:

- Підприємства з розгалуженими мережами, що потребують глибокого аналізу трафіку.
- Організації, які хочуть виявляти аномалії в реальному часі та автоматично на них реагувати.
- Компанії, які використовують змішані середовища (локальні сервери, хмарні платформи, IoT).

Ці приклади демонструють, як інтеграція ШІ в антивірусні рішення підвищує ефективність виявлення та реагування на сучасні кіберзагрози, забезпечуючи проактивний захист інформаційних систем.

6. ШІ як інструмент зловмисника – риски та загрози, пов'язані з ШІ

В попередніх розділах було розглянуто переваги використання ШІ в кібербезпеці, але безперечно є і багато викликів щодо його використання, оскільки ШІ не лише відкриває нові можливості для розвитку, але й стає інструментом у руках зловмисників, що створює нові ризики та загрози. Найпоширенішими серед таких загроз у сучасному світі є:

- Атаки на основі ШІ: Кіберзлочинці можуть використовувати ШІ для проведення більш складних та цілеспрямованих атак. Це підвищує ефективність їхніх дій та ускладнює виявлення загроз.
- Дезінформація та глибокі підробки (Deepfake): ШІ дозволяє створювати реалістичні фальшиві відео, аудіо та тексти, що можуть підривати довіру до авторитетних джерел, маніпулювати громадською думкою та втручатися у виборчі процеси.
- Автоматизація кібератак: Зловмисники можуть використовувати ШІ для автоматизації атак, що дозволяє проводити їх швидше та з меншою ймовірністю бути виявленими.
- Атаки на системи ШІ: Зловмисники можуть маніпулювати даними, на яких навчаються системи ШІ, що призводить до неправильних рішень та підвищує вразливість систем.

Зловмисники використовують ШІ для автоматизації та підвищення ефективності атак соціальної інженерії. Наприклад, нейромережі можуть автоматично створювати дуже правдоподібні фішингові повідомлення, які переконують користувачів поділитися своїми паролями або іншою важливою інформацією. Це дозволяє зловмисникам отримати доступ до системи без необхідності проводити технічні атаки, обходячи багато рівнів захисту. Також зловмисники активно використовують ШІ для проведення різноманітних атак, що вимагає від фахівців з безпеки розробки нових стратегій захисту. Тому далі розглянемо основні типи атак на основі ШІ.

6.1. Атаки отруєння даних

Перші атаки отруєння даних (Data Poisoning) [15] були проведені в кібербезпеці ще в 2006 [19] та 2008 [20] роках і з того часу набули популярності серед зловмисників. Дані атаки відбуваються на етапі навчання або донавчання моделі ШІ. Зловмисники вводять шкідливі дані в навчальну базу, що призводить до неправильного функціонування системи та генерації хибних результатів. Це може спричинити помилки в класифікації або прийнятті рішень. Наприклад, мережі GAN можуть створювати штучні дані, які виглядають легітимно, але призначені для введення в оману або псування моделей машинного навчання, що впливає на їх продуктивність і надійність.

Отруєння даними також може посилити існуючі упередження в системах ШІ [16]. Зловмисники можуть націлитися на конкретні підмножини даних, наприклад на певну

демографію, щоб ввести упереджені дані. Через це модель штучного інтелекту може працювати нечесно або неточно. Наприклад, моделі розпізнавання обличь, навчені з упередженими або фальшивими даними, можуть неправильно ідентифікувати людей із певних груп, що призведе до дискримінаційних результатів. Ці типи атак можуть вплинути як на справедливість, так і на точність моделей ML у різних програмах.

Отруєння даних також може відкрити двері для більш складних атак [16], таких як інверсійні атаки, під час яких хакери намагаються реконструювати навчальні дані моделі. Після того, як зловмисник успішно отруює навчальні дані, він може далі використовувати ці вразливості для запуску більш серйозних атак. У системах, розроблених для чутливих завдань, таких як кібербезпека, ці ризики можуть бути особливо небезпечними.

Щоб захиститися від атак з отруєнням даних, організації можуть впроваджувати стратегії, які допоможуть забезпечити цілісність навчальних наборів даних, покращити надійність моделі та постійно контролювати моделі ШІ.

6.2. Атаки ухилення

Атаки ухилення (Evasion Attacks) [18] полягають у створенні спеціальних вхідних даних, які вводять модель ШІ в оману, змушуючи її робити неправильні прогнози або класифікації. Такі атаки можуть бути спрямовані на обхід систем виявлення шкідливого програмного забезпечення або інших захисних механізмів.

Відкриття атак ухилення від моделей машинного навчання викликало підвищений інтерес до змагального машинного навчання, що призвело до значного зростання цього дослідницького простору за останнє десятиліття. Під час атаки з ухиленням метою зловмисника є створення змагальних прикладів, які визначаються як тестові зразки [21].

У програмах кібербезпеки змагальні приклади повинні поважати обмеження, накладені семантикою програми та представленням функцій кіберданих, таких як мережевий трафік або бінарні файли програми [18].

FENCE – це загальна структура для створення атак ухилення білої скриньки з використанням градієнтної оптимізації в дискретних областях і підтримує низку лінійних і статистичних залежностей характеристик [22]. FENCE було застосовано до двох програм безпеки мережі: виявлення зловмисного домену та класифікація шкідливого мережевого трафіку. В [23] цю техніку було застосовано для виявлення мережевих вторгнень і класифікаторів фішингу. В документі [23] зазначено, що атаки з безперервних доменів не можуть бути легко застосовані в обмежених середовищах, оскільки вони призводять до нездійсненних прикладів змагання. Пієрацці та ін. [24] обговорюють труднощі встановлення можливих атак ухилення в кібербезпеці через обмеження в просторі функцій та формалізують атаки ухилення в проблемному просторі та створюють можливі змагальні приклади для шкідливих програм Android.

6.3. Атаки швидкого впровадження

Під час атаки швидкого впровадження (Prompt Injection) [18] зловмисники вставляють шкідливі інструкції в запити до моделей ШІ, змушуючи їх виконувати небажані дії або надавати конфіденційну інформацію, тобто обманюють модель для повернення неочікуваної відповіді та змушуючи додаток діяти незапланованими способами. Успішне впровадження може призвести до витоку конфіденційних даних, знищення інформації та інших типів шкоди залежно від застосунку.

6.4. Атаки соціальної інженерії з використанням ШІ

ШІ використовується для автоматизації та підвищення ефективності атак соціальної інженерії, таких як фішинг або маніпуляції, що ускладнює їх виявлення.

Зловмисники використовують методи соціальної інженерії для приховування своєї справжньої «особистості», представляючись жертвам надійними організаціями або особами. Ці атаки спрямовані на отримання особистої інформації для доступу до цільової мережі шляхом обману та маніпуляцій. Соціальна інженерія використовується як перший етап великої кібератаки для проникнення в систему, установки шкідливого ПЗ, розкриття конфіденційних даних тощо.

Наприклад, у випадку з фішингом [18], раніше було продемонстровано, що великі мовні моделі (LLM) можуть створювати переконливі шахрайства, такі як фішингові електронні листи [25]. Тепер, коли LLM можуть легше інтегруватися з додатками, вони можуть не тільки створювати шахрайські дії, але й широко поширювати такі атаки [26]. Користувачі, швидше за все, будуть більш сприйнятливі до цих нових атак, на відміну від фішингових електронних листів, оскільки їм бракує досвіду та обізнаності про цю нову техніку загроз.

Також сама LLM діє як комп’ютер, на якому працює та поширюється шкідливий код. Наприклад, інструмент автоматичної обробки повідомлень, який може читати та створювати електронні листи та переглядати особисті дані користувачів, може поширити ін’єкцію на інші моделі, які можуть читати ці вхідні повідомлення [27].

Тож, розглянемо заходи протидії атакам на основі ШІ. Щоб убезпечитися від атак з використанням ШІ необхідно зробити наступні дії:

- Покращення якості даних: Забезпечення чистоти та надійності даних для навчання моделей ШІ допомагає зменшити ризик отруєння даних.
- Розробка стійких моделей: Створення моделей ШІ, стійких до атак, шляхом впровадження методів захисту та регулярного тестування.
- Моніторинг та аудит: Постійний нагляд за роботою моделей ШІ та проведення аудитів для виявлення можливих вразливостей.
- Навчання персоналу: Підвищення обізнаності працівників щодо можливих атак на основі ШІ та методів їх виявлення.

У сучасному цифровому середовищі важливо бути готовим до нових викликів, пов'язаних з використанням ШІ, та впроваджувати відповідні заходи для забезпечення кібербезпеки.

7. Висновки

1. У цій статті було проведено всебічний аналіз поточного стану та перспектив застосування штучного інтелекту (ШІ) у сфері кібербезпеки. Розглянуто як переваги впровадження ШІ в системи захисту, так і ризики, пов’язані з його використанням.

2. ШІ дозволяє автоматизувати процеси виявлення та реагування на загрози, що значно підвищує ефективність кіберзахисту. Використання алгоритмів машинного навчання сприяє швидкому аналізу великих обсягів даних та ідентифікації аномалій у поведінці користувачів і систем.

3. Сучасні системи кібербезпеки, такі як SIEM (Security Information and Event Management), значно виграють від інтеграції ШІ, оскільки здатні аналізувати події в реальному часі та попереджати про можливі атаки. Постійне вивчення проблем минулого підвищує точність і надійність систем SIEM на основі штучного інтелекту проти дедалі потужніших кіберзагроз. Зрештою, SIEM на основі штучного інтелекту об’єднує різні компоненти, такі як

AI, ML, глибоке навчання, NLP і UEBA. Ця інтеграція веде до більш розумних, ефективних і проактивних заходів кібербезпеки, що має вирішальне значення в середовищі кіберзагроз, що постійно змінюється. При цьому, велика кількість інтеграцій з різноманітними системами, дозволяє системам SIEM стежити та накопичувати дані щодо поточного стану сфери кіберзахисту інформаційної інфраструктури щодо певних міжнародних та національних стандартів, таких як ISO 27001, GDPR чи PCI DSS.

4. ШІ може аналізувати величезні обсяги даних і виявляти закономірності, що вказують на потенційні загрози, дозволяючи організаціям випереджати хакерів. А також, ШІ здатний швидше і точніше ідентифікувати загрози, а також автоматично блокувати шкідливий трафік без втручання людини. Завдяки поведінковому аналізу, ШІ-антивіруси (наприклад, Microsoft Defender ATP та Darktrace) можуть ефективніше виявляти загрози, ніж традиційні антивірусні програми.

5. Щодо використання систем захисту виявлення атак, то якщо компанія активно використовує Microsoft 365, Azure та Windows – краще обрати Microsoft Defender for Endpoint. Він забезпечить глибоку інтеграцію, автоматизоване реагування та поведінковий аналіз загроз на кінцевих пристроях. Якщо потрібен гнучкий та автономний захист усіх рівнів мережі – варто розглянути Darktrace. Він підійде організаціям, які хочуть виявляти аномалії, аналізувати кіберзагрози в реальному часі та реагувати на них без втручання людей. Але ідеальний варіант – поєднання обох рішень: Microsoft Defender for Endpoint для захисту кінцевих точок, а Darktrace – для мережевого аналізу та автоматичного реагування на загрози.

6. Використання ШІ не лише для захисту, а й для атак становить значну загрозу. Зловмисники можуть застосовувати ШІ для автоматизації атак, маніпуляцій та соціальної інженерії. Зловмисники все частіше використовують ШІ для створення складного шкідливого програмного забезпечення, яке може адаптуватися до захисних заходів. Експерти з безпеки занепокоєні потенційними автономними атаками ШІ, що змушує компанії готуватися вже зараз. Організаціям потрібно впроваджувати комплексні стратегії, щоб скористатися перевагами ШІ, одночасно мінімізуючи його потенційні загрози.

7. Перспективами розвитку та рекомендації щодо використання ШІ в кібербезпеці є:

- Підвищення прозорості алгоритмів ШІ та впровадження етичних стандартів є критично важливими для довіри до технологій ШІ у сфері безпеки.
- Розробка стійких моделей ШІ, які будуть менш вразливими до атак з боку зловмисників.
- Інвестування в дослідження постквантової криптографії та новітніх методів аутентифікації для зміцнення систем кібербезпеки.
- Посилення міжнародного співробітництва у сфері стандартів та регулювання ШІ, зокрема завдяки ініціативам таких організацій, як NIST.

8. Таким чином, штучний інтелект має потенціал повністю змінити підхід до кібербезпеки. Його здатність швидко аналізувати великі масиви даних, передбачати загрози та автоматично реагувати на атаки робить його ключовим елементом захисту у цифровому світі. Проте, слід пам’ятати, що кіберзлочинці також використовують ШІ, тому майбутнє кібербезпеки залежатиме від балансу між інноваціями у захисті та загрозами з боку злочинних угруповань.

9. Майбутнє кібербезпеки – це симбіоз людини та штучного інтелекту, де аналітики та експерти з безпеки співпрацюють із розумними системами для створення надійного кіберпростору.

Конфлікт інтересів

Автори повідомляють про відсутність конфлікту інтересів.

References

1. NIST standardization process “Post-Quantum Cryptography: Digital Signature Schemes”. URL: https://csrc.nist.gov/Projects/pqc-dig-sig/round-1-additional-signatures
2. TAO, Feng; Akhtar, Muhammad Shoaib; Jiayuan, Zhang. (2021) The future of artificial intelligence in cybersecurity: A comprehensive survey. *EAI Endorsed Transactions on Creative Technologies*, 8.28: e3-e3. DOI: https://doi.org/10.4108/eai.7-7-2021.170285
3. Leung, B.K. (2021). Security Information and Event Management (SIEM) Evaluation Report. ScholarWorks. URL: https://scholarworks.calstate.edu/downloads/41687p49q
4. González-Granadillo, G., González-Zarzosa, S., Diaz, R. (2021). Security Information and Event Management (SIEM): Analysis, Trends, and Usage in Critical Infrastructures. *Sensors*, 21(14). DOI: https://doi.org/10.3390/s21144759
5. Muhammad, S., et al. (2023). Effective Security Monitoring Using Efficient SIEM Architecture. *Human-centric Computing and Information Sciences*, 13. DOI https://doi.org/10.22967/HCIS.2023.13.017
6. What is SIEM. Security Information and Event Management Tools. (n.d.). Imperva. URL: https://www.imperva.com/learn/application-security/siem/
7. IBM Security QRadar. What is security information and event management (SIEM)? URL https://www.ibm.com/think/topics/siem
8. Splunk. The Splunk SIEM. URL: https://www.splunk.com/en_us/products/enterprise-security.html
9. Stellar Cyber. AI SIEM: The 6 Components of AI-Based SIEM. URL: https://stellarcyber.ai/learn/ai-driven-siem/
10. ISO/IEC 27001: (2022). Information technology – Security techniques – Information security management systems – Requirements. International standard. 3 Edition.
11. Microsoft Defender for Endpoint. (2024). URL: https://learn.microsoft.com/uk-ua/defender-endpoint/microsoft-defender-endpoint
12. Darktrace. Official website (2024). URL: https://darktrace.com/
13. Mauri L., Damiani E. Modeling Threats to AI-ML Systems Using STRIDE. Sensors (2022), (17), 6662; DOI: https://doi.org/10.3390/s22176662
14. The near-term impact of AI on the cyber threat. URL: https://www.ncsc.gov.uk/report/impact-of-ai-on-cyber-threat
15. Nihad Hassan. What is data poisoning (AI poisoning) and how does it work? Search Enterprise AI, TechTarget, (2024). URL: https://www.techtarget.com/searchenterpriseai/definition/data-poisoning-AI-poisoning
16. Tom Krantz, Alexandra Jonker. What is data poisoning? IBM. URL: https://www.ibm.com/think/topics/data-poisoning
17. NIST Trustworthy and Responsible AI NIST AI 100-5. A Plan for Global Engagement on AI Standards. DOI: https://doi.org/10.6028/NIST.AI.100-5
18. Vassilev A, Oprea A, Fordyce A, Anderson H (2024) Adversarial Machine Learning: A Taxonomy and Terminology of Attacks and Mitigations. (National Institute of Standards and Technology, Gaithersburg, MD) *NIST Artificial Intelligence (AI) Report, NIST Trustworthy and Responsible AI NIST AI* 100-2e2023. DOI: https://doi.org/10.6028/NIST.AI.100-2e2023
19. R. Perdisci, D. Dagon, Wenke Lee, P. Fogla, and M. Sharif. (2006) Misleading worm signature generators using deliberate noise injection. *In 2006 IEEE Symposium on Security and Privacy* (S&P’06), Berkeley/Oakland, CA, IEEE. DOI: https://doi.org/10.1109/SP.2006.26
20. Blaine Nelson, Marco Barreno, Fuching Jack Chi, Anthony D. Joseph, Benjamin I.P. Rubinstein, Udam Saini, Charles Sutton, and Kai Xia. (2008) Exploiting machine learning to subvert your spam filter. *In First USENIX Workshop on Large-Scale Exploits and Emergent Threats (LEET 08)*, San Francisco, CA, USENIX Association. URL: https://people.eecs.berkeley.edu/~tygar/papers/SML/Spam_filter.pdf
21. Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian Goodfellow, and Rob Fergus. (2014) Intriguing properties of neural networks. *In International Conference on Learning Representations.* DOI: https://doi.org/10.48550/arXiv.1312.6199

22. Alesia Chernikova and Alina Oprea. (2022) FENCE: Feasible evasion attacks on neural networks in constrained environments. *ACM Transactions on Privacy and Security (TOPS) Journal.* DOI: https://doi.org/10.48550/arXiv.1909.10480
23. Ryan Sheatsley, Blaine Hoak, Eric Pauley, Yohan Beugin, Michael J. Weisman, and Patrick McDaniel (2021). On the robustness of domain constraints. *In Proceedings of the 2021 ACM SIGSAC Conference on Computer and Communications Security, CCS '21*, page 495–515, New York, NY, USA. Association for Computing Machinery. DOI: https://doi.org/10.48550/arXiv.2105.08619
24. Fabio Pierazzi, Feargus Pendlebury, Jacopo Cortellazzi, and Lorenzo Cavallaro (2020). Intriguing properties of adversarial ML attacks in the problem space. *In 2020 IEEE Symposium on Security and Privacy (S&P)*, pages 1308–1325. IEEE Computer Society. DOI: https://doi.org/10.48550/arXiv.1911.02142
25. Daniel Kang, Xuechen Li, Ion Stoica, Carlos Guestrin, Matei Zaharia, and Tatsunori Hashimoto (2023). Exploiting Programmatic Behavior of LLMs: Dual-use through standard security attacks. DOI: https://doi.org/10.48550/arXiv.2302.05733
26. Kai Greshake, Sahar Abdelnabi, Shailesh Mishra, Christoph Endres, Thorsten Holz, and Mario Fritz (2023). Not what you've signed up for: Compromising Real-World LLM-Integrated Applications with Indirect Prompt Injection. DOI: https://doi.org/10.48550/arXiv.2302.12173

RESEARCH ON THE CURRENT STATE AND PROSPECTS OF THE APPLICATION OF ARTIFICIAL INTELLIGENCE IN CYBERSECURITY

Yuriy Golikov[1], e-mail: yuriy@devbrother.com;
Yelyzaveta Ostrianska[2], junior researcher; e-mail: antelizza@gmail.com;
ORCID: https://orcid.org/0000-0003-1412-8470
[1]*DevBrother tech company, USA*
[2]*V. N. Karazin Kharkiv National University, Ukraine*
Manuscript was received November 1, 2024; Received after review December 2, 2024;
Accepted December 23, 2024

Abstract. In the modern world, with the development of new technologies, artificial intelligence (AI) in cybersecurity has become an integral component. Therefore, studying its advantages, risks, and potential use cases is a highly relevant research topic. In today's digital environment, where cyber threats are becoming increasingly sophisticated, the implementation of AI technologies significantly enhances the effectiveness of security systems by enabling automated threat detection and response. In this study the main applications of AI in cybersecurity were examined, including threat detection, malware analysis, cryptographic security enhancement, phishing protection, and attack prediction. One of the key aspects is the integration of AI into Security Information and Event Management (SIEM) systems, which analyze vast amounts of data and help detect anomalies. Such systems reduce the workload on security teams and improve the accuracy and speed of threat response. Special attention is given to the analysis of modern AI-powered antivirus solutions, particularly Microsoft Defender for Endpoint and Darktrace. These solutions are based on behavioral analysis algorithms and machine learning, allowing for more effective detection of complex threats and incident prevention. Microsoft Defender provide a high level of endpoint protection. Meanwhile, Darktrace utilizes self-learning models to analyze network traffic, enabling the detection of zero-day threats and internal risks within organizations. The study also learns the major risks associated with the use of AI in cybercrime. AI is increasingly leveraged by malicious actors to automate attacks, significantly increasing their effectiveness and making detection more challenging. The primary

AI-based cyber threats discussed include Data Poisoning attacks, Evasion Attacks, Prompt Injection Attacks, and AI-based social engineering. To mitigate these risks, the development of robust AI models resistant to adversarial attacks, increased algorithm transparency, and the implementation of international AI regulation standards is recommended, including NIST. Additionally, raising awareness among users and cybersecurity specialists is crucial, as the human factor remains one of the most significant vulnerabilities in security systems. In conclusion, it is said that AI is a key factor in the advancement of cybersecurity, offering significant improvements in protecting information and critical systems. However, without proper regulation and protective measures, AI can become a powerful tool for cybercriminals, posing new security challenges in the digital age. Striking a balance between innovation, ethical standards, and security will be essential in shaping the future strategy for the effective use of AI.

Keywords: *artificial intelligence, cybersecurity, machine learning, SIEM, IDS, AI-based antivirus*

Conflicts of Interest: the authors declare no conflict of interest.

Електронне наукове видання

КОМП’ЮТЕРНІ НАУКИ ТА КІБЕРБЕЗПЕКА
№ 2(26) 2024

Міжнародний електронний науково-теоретичний журнал

Українською та англійською мовами

Комп’ютерне верстання – Єсіна М.В., Власова В.В.

Підписано до розміщення 23.12.2024. Гарнітура Times New Roman.
Ум. друк. арк. 4,07. Обсяг 5,08 Мб. Зам. № 58/24.

Харківський національний університет імені В. Н. Каразіна,
61022, м. Харків, майдан Свободи, 4.
Свідоцтво суб’єкта видавничої справи ДК № 3367 від 13.01.2009