МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ДЕРЖАВНИЙ УНІВЕРСИТЕТ «КИЇВСЬКИЙ АВІАЦІЙНИЙ ІНСТИТУТ»

Кваліфікаційна наукова праця
на правах рукопису

КРАНТ ДАНІЇЛ ВЯЧЕСЛАВОВИЧ

УДК 004.78:004.89

МЕТОДИ ВИКОРИСТАННЯ ШИН ПЕРЕДАЧІ ДАНИХ В АВТОМАТИЗОВАНИХ СИСТЕМАХ ТРАНСПОРТНИХ ЗАСОБІВ

122 «Комп’ютерні науки»
12 «Інформаційні технології»
Подається на здобуття ступеня доктора філософії
Дисертація містить результати власних досліджень. Використання ідей, результатів і текстів інших авторів мають посилання на відповідне джерело.
_____ Д.В. Крант

Науковий керівник
Артамонов Євген Борисович,
кандидат технічних наук, доцент

Київ – 2025

АНОТАЦІЯ

Крант Д.В. Методи використання шин передачі даних в автоматизованих системах транспортних засобів – кваліфікаційна наукова праця на правах рукопису.

Дисертація на здобуття наукового ступеня доктора філософії в галузі знань 12 «Інформаційні технології», за спеціальністю 122 «Комп'ютерні науки». – Державний університет «Київський авіаційний інститут», м. Київ, 2025.

The dissertation is devoted to improving the methods of data bus utilization in multicomponent automated vehicle systems in order to enhance their efficiency, safety, and adaptability under increasing demands for performance, synchronization, and limited computational resources.

Проведено аналіз існуючих підходів до організації інформаційної взаємодії в транспортних системах, який виявив недостатню ефективність традиційних методів використання шин у складних мультишинних середовищах (*CAN–FlexRay–Ethernet*). Встановлено, що наявні рішення недостатньо враховують контекстуальні й поведінкові фактори, не забезпечують необхідного рівня безпеки транзакцій та мають обмеження щодо швидкості передачі і масштабованості.

Запропоновано метод визначення ймовірнісної оцінки достовірності повідомлень у *CAN*-шинах на основі байєсівського підходу, що враховує поведінкові шаблони, контекстуальні характеристики та дозволяє ідентифікувати аномальні послідовності. Експериментальне дослідження підтвердило ефективність методу з точністю виявлення загроз понад 95 %.

Розроблено узагальнену модель взаємодії у мультишинному середовищі *CAN–FlexRay–Ethernet*, яка враховує структуру транзакцій, маршрутизацію, динаміку роботи компонентів і часові обмеження. Для забезпечення міжшинової сумісності запропоновано методику адаптивного фільтрування повідомлень з підтримкою механізму категоризації транзакцій за рівнем довіри та алгоритмом *Adam* для оперативної адаптації параметрів моделі.

Проведено експериментальні дослідження розроблених методів у типових конфігураціях транспортних систем, результати яких підтвердили їхню високу

ефективність, детермінованість передачі даних і здатність до масштабування в умовах реальної експлуатації транспортних засобів з різними рівнями автоматизації.

Запропоновані методи можуть бути інтегровані в електронні блоки керування для реального моніторингу шинного трафіку з мінімальними вимогами до обчислювальних ресурсів, що відкриває перспективи їх широкого впровадження в сучасних транспортних системах.

Отримані результати дисертації мають практичну цінність і впроваджені в діяльність ТОВ «АЕРОФАБ УКРАЇНА» для категоризації операторів БПЛА, а також у навчальний процес кафедри інтелектуальних кібернетичних систем ДП КАІ у рамках дисциплін «Дослідження і проєктування вбудованих і мобільних систем» та «Системне програмування».

Ключові слова: моделювання комунікацій у транспорті, інформаційна кібербезпека, цифрові комунікаційні системи, *CAN*-шина, обробка інформації, цифровий контролер, системи реального часу, математичне моделювання, симуляція, багатокритеріальні задачі, персоналізація, системи управління.

ANNOTATION

Krant D.V. Methods of using data buses in automated vehicle systems. – Qualification research work presented as a manuscript.

Dissertation for the degree of Doctor of Philosophy in speciality 122 "Computer Science". – State University "Kyiv Aviation Institute", Kyiv, 2025.

The dissertation is devoted to improving the methods of data bus utilization in multicomponent automated vehicle systems in order to enhance their efficiency, safety, and adaptability under increasing demands for performance, synchronization, and limited computational resources.

An analysis of existing approaches to organizing information interaction in transportation systems revealed the insufficient efficiency of traditional bus usage methods in complex multi-bus environments (CAN–FlexRay–Ethernet). It was established that current solutions inadequately account for contextual and behavioral factors, fail to ensure the necessary level of transaction security, and have limitations regarding data transmission speed and scalability.

A method for probabilistic assessment of message reliability in CAN buses, based on a Bayesian approach, is proposed. It considers behavioral patterns and contextual characteristics, allowing the identification of anomalous sequences. Experimental research confirmed the method’s effectiveness, demonstrating threat detection accuracy exceeding 95%.

A generalized interaction model in a multi-bus environment (CAN–FlexRay–Ethernet) was developed, considering transaction structure, routing, component dynamics, and timing constraints. To ensure inter-bus compatibility, an adaptive message filtering technique was proposed, featuring a transaction categorization mechanism by trust level and using the Adam algorithm for real-time parameter adaptation.

Experimental studies of the developed methods in typical transportation system configurations confirmed their high efficiency, deterministic data transmission, and scalability in real-world operational conditions of vehicles with varying automation levels.

The proposed methods can be integrated into electronic control units for real-time bus traffic monitoring with minimal computational resource requirements, providing broad implementation opportunities in modern transportation systems.

The dissertation’s findings have practical value and have been implemented at LLC "AEROFAB UKRAINE" for UAV operator categorization, as well as integrated into the educational process of the Department of Intelligent Cybernetic Systems at the Kyiv Aviation Institute within the disciplines "Research and Design of Embedded and Mobile Systems" and "System Programming".

Keywords: transportation communication modeling, information cybersecurity, digital communication systems, CAN bus, information processing, digital controller, real-time systems, mathematical modeling, simulation, multi-criteria problems, personalization, control systems.

СПИСОК ПУБЛІКАЦІЙ ЗДОБУВАЧА

Статті у наукових фахових виданнях України:

1. Артамонов Є.Б., Головач Ю.Ю., Крант Д.В., Радченко К.М. Використання алгоритму Левенштейна для категоризації користувачів інформаційних систем. *Проблеми інформатизації та управління*. 79(3). К.: НАУ. 2024. С. 4-12. *DOI*: *https*://*doi.org*/10.18372/2073-4751.79.19366.

Особистий внесок автора: розробка методу категоризації користувачів на основі відстані Левенштейна, реалізація алгоритму в експериментальному середовищі, аналіз результатів.

Особистий внесок Артамонова Є.Б.: наукове керівництво, постановка задачі, редагування статті.

Особистий внесок Головача Ю.Ю.: забезпечення математичного апарату оцінювання.

Особистий внесок Радченка К.М.: допомога в побудові тестового стенду та перевірці результатів.

2. Граф М.С., Яконюк А.В., Крант Д.В., Головач Ю.Ю. Аналіз можливостей інформаційної системи покращення якості сну на основі аналізу біометричних даних. *Технічна інженерія*, 2(94). 2025. С. 113–120. *DOI*: *https*://*doi.org*/10.26642/*ten*-2024-2(94)-113-120.

Особистий внесок автора: розробка модуля обробки біометричних даних, формування структури інформаційної системи, інтерпретація результатів.

Особистий внесок Граф М.С.: збір біометричних даних і визначення метрик.

Особистий внесок Яконюка А.В.: валідація моделей сну, адаптація на мобільних платформах.

Особистий внесок Головача Ю.Ю.: алгоритмізація обробки біометричних даних.

3. Артамонов Є.Б., Коцюр А.Б., Крант Д.В., Радченко К.М. Підхід до оптимізації моделі розгортання мікросервісів в сильнонавантаженому середовищі. *Проблеми інформатизації та управління*. 80(4). К.: ДНП «ДУ «КАІ». 2025. С. 4-15. *DOI*: *https*://*doi.org*/10.18372/2073-4751.80.19787.

Особистий внесок автора: розробка логіки балансування навантаження між мікросервісами, проведення моделювання навантаження.

Особистий внесок Артамонова Є.Б.: формулювання теоретичної основи.

Особистий внесок Коцюра А.Б.: аналіз архітектур мікросервісів.

Особистий внесок Радченка К.М.: експериментальне тестування розробленої архітектури.

4. Артамонов Є.Б., Крант Д.В. Оцінка можливостей адаптації інтерфейсів і контенту в програмних і апаратних системах через аналіз поведінки користувача. «*Наука і техніка сьогодні*» (Серія «Педагогіка», Серія «Право», Серія «Економіка», Серія «Фізико-математичнінауки», Серія «Техніка»). 3 (44). 2025. С. 1548-1561. *DOI*: *https*://*doi*.*org*/10.52058/2786-6025-2025-3(44)-1548-1561.

Особистий внесок автора: формалізація поведінкових моделей користувача, створення системи адаптації контенту, аналіз результатів.

Особистий внесок Артамонова Є.Б.: постановка проблеми, визначення цілей дослідження, участь у написанні висновків.

5. Артамонов Є.Б., Крант Д.В. Метод виявлення загроз безпеки в *CAN*-шині з використанням Баєсівського підходу. «*Наука і техніка сьогодні*» (Серія «Педагогіка», Серія «Право», Серія «Економіка», Серія «Фізико-математичнінауки», Серія «Техніка»). 4 (45). 2025. С. 1067-1082. *DOI*: *https*://*doi*.*org*/10.52058/2786-6025-2025-4(45).

Особистий внесок автора: формалізація методу, реалізація програмної частини, моделювання *CAN*-трафіку, побудова тестів.

Особистий внесок Артамонова Є.Б.: методологічне супроводження дослідження, аналіз релевантності результатів.

Розділ у колективній монографії:

1. *Kashkevich, S.*; *Matsyi, O.*; *Voznytsia, A.*; *Buyalo, O.*; *Krant*; *D.*; *Radchenko, K.* (2025). *Decision support systems*: *mathematical support*: *collective monograph* (*Chapter* 4. *Scientific and methodological apparatus for processing diverse data in automated control systems*) 2025, *Kharkiv*: TEC*HNOLOGY* CE*NTER PC*, pp. 95–123. *DOI*: 10.15587/978-617-8360-13-9.*CH*4 (*All*: *https*://*doi*.*org*/10.15587/978-617-8360-13-9).

Особистий внесок автора: розроблено підхід до ф'юзії гетерогенних даних у середовищах з обмеженими ресурсами в умовах реального часу, особливо для транспортних систем та запропоновано концепцію адаптивної структури обробки даних на основі поведінкових і контекстних параметрів.

Особистий внесок Kashkevich S.: оцінка достовірності даних та моделювання взаємодії між компонентами середовища.

Особистий внесок Matsyi O.: проведено аналіз способів інтеграції алгоритмів підтримки прийняття рішень у вже наявні автоматизовані системи управління.

Особистий внесок Voznytsia A.: розробка математичного забезпечення, перевірка коректності моделей, структура логічних залежностей.

Особистий внесок Buyalo O.: побудова узагальненого інформаційного середовища.

Особистий внесок Radchenko K.: підготовка методичного матеріалу щодо використання результатів у навчальному процесі.

Статті в іноземних виданнях:

(статті у періодичних наукових виданнях, проіндексованих у базах даних Web of Science Core Collection та/або Scopus)

1. *Artamonov Y., Golovach I., Krant D., Rosinska H., Nechyporuk O., Stanko S. Dynamic Content Generation Methods Based on User Behavioral Ranking*, 2022 *IEEE* 4*th International Conference on Advanced Trends in Information Theory* (*ATIT*), *Kyiv, Ukraine*, 2022, *pp*. 313-318, *doi*: 10.1109/*ATIT*58178.2022.10024196. *ISBN*: 979-8-3503-3262-9.

Особистий внесок автора: розробка моделей поведінкової класифікації користувачів для адаптивного контенту, підготовка тестових сценаріїв та аналіз точності класифікатора.

Особистий внесок Artamonov Y.: участь в розробці систем, що аналізуються.

Особистий внесок Golovach I.: структуризація алгоритмів.

Особистий внесок Rosinska H.: частина, пов'язана з розробкою пристроїв.

Особистий внесок Nechyporuk O.: проведення оцінювання моделі.

Особистий внесок Stanko S.: перевірка на сумісність із мобільними платформами.

2. *Artamonov, Y., Golovach, I., Krant, D., Rosinska, H., Stanko, S. Modeling the operation of multi-scenario systems, Proceedings on Engineering Sciencesthis*, 2023, 5(2), *pp*. 219–226. *doi*: 10.24874/*PES*05.02.004. *URL*: *https*://*pesjournal*.*net*/*journal*/*v*5-*n*2/4.*pdf*. *ISSN*: 2620-2832. *Q*3.

Особистий внесок автора: моделювання сценаріїв комунікації, впровадження елементів обробки вхідних даних.

Особистий внесок Artamonov Y.: участь в розробці систем, що аналізуються.

Особистий внесок Golovach I.: структуризація алгоритмів вибору сценаріїв.

Особистий внесок Rosinska H.: обробка результатів моделювання.

Особистий внесок Stanko S.: аналіз останніх наукових досліджень.

Наукові праці, які додатково відображають наукові результати дисертації

1. Крант Д.В. Методи організації доступу до *CAN*-шин автомобільних інформаційних систем з *Android*-додатків. *Сучасні тенденції розвитку системного програмування*: тези доп. наук.-практ. конф. (Україна, м. Київ, 25-26 листопада 2021 р.). К.: НАУ, 2021. С. 8.

2. Крант Д.В., Артамонов Є.Б. Принципи роботи *CAN*-шин в автомобільних інформаційних системах. *Сучасні тенденції розвитку системного програмування*: тези доп. наук.-практ. конф. (Україна, м. Київ, 24-25 листопада 2022 р.). К.: НАУ, 2022. С. 22-23.

Особистий внесок автора: технічний опис роботи шин, розбір прикладів з практики.

Особистий внесок Артамонова Є.Б.: методичне супроводження, теоретичний огляд.

3. Крант Д.В., Артамонов Є.Б., Данкович Н. І. Підвищення рівня захищеності внутрішніх баз даних за рахунок аналізу поведінкових ознак користувача. *Актуальні проблеми управління інформаційною безпекою держави*: тези доп. *XIV* Всеукраїнської

наук.-практ. конф. (Україна, м. Київ, 30 березня 2023 року). К.: НА СБУ, 2023. С. 26-28.

Особистий внесок автора: адаптація методів для *CAN*-середовища.

Особистий внесок Артамонова Є.Б.: формалізація та алгоритмізація аналізу поведінкових ознак користувача.

Особистий внесок Данкович Н.І.: безпековий аналіз.

4. Крант Д.В., Артамонов Є.Б. Метод додаткової аутентифікації користувачів через аналіз поведінкових ознак користувача. *Інформаційно-комп'ютерні технології* (ІКТ-2023): матеріали *XIII* міжн. наук.-техн. конф. (Україна, м. Житомир, 30-31 березня 2023 р.). Житомир: Житомирська політехніка, 2023. С. 30-31.

Особистий внесок автора: розробка методу аналізу користувацьких шаблонів.

Особистий внесок Артамонова Є.Б.: формалізація та алгоритмізація аналізу поведінкових ознак користувача.

5. Крант Д.В., Артамонов Є.Б., Залозний Т.І. Інноваційні підходи до ф'юзії даних для підвищення ефективності і стійкості в автономних системах навігації і моніторингу. *Актуальні проблеми науки, освіти і технологій*: тези доповідей міжнародної науково-практичної конференції (Словаччина, м. Братислава, 25 липня 2023 р.). Братислава, 2023. С. 78-79.

Особистий внесок автора: адаптація до *CAN*-шин, визначення точок ф'юзії.

Особистий внесок Залозного Т.І.: математичне обґрунтування підходу.

Особистий внесок Артамонова Є.Б.: оцінка стійкості системи.

6. Крант Д.В. Принципи відстеження дій користувача в автомобільному симуляторі. *Сучасні тенденції розвитку системного програмування*: тези доп. наук.-практ. конф. (Київ, Україна, 23-24 листопада, 2023 р.). К.: НАУ, 2023. С. 12.

7. *Artamonov, Y.; Okhrimenko, T.; Golovach, I.; Radchenko, A.; Krant D.; Radchenko K., Zaloznyi T. Adaptive user interfaces based on behavioral analysis. Proceedings of the Third International Conference on Cyber Hygiene & Conflict Management in Global Information Networks* (*Kyiv, Ukraine, January* 24–27, 2024). *pp*. 205-214. *URL*: *https*://*ceur-ws.org*/*Vol*-3925/

Особистий внесок автора: запропонував підхід до класифікації стилю взаємодії користувача на основі шаблонів навігації, швидкості реакцій та шаблонів дій. Розробив модель категоризації користувачів за активністю та вніс практичне обґрунтування застосування алгоритмів адаптації.

Особистий внесок Artamonov Y.: обґрунтував архітектуру адаптивного інтерфейсу, визначив наукову проблематику взаємодії між рівнем когнітивного навантаження користувача та поведінковою динамікою при роботі з ПЗ.

Особистий внесок Okhrimenko T.: провела аналіз психологічних аспектів користувацької поведінки, класифікацію типів користувачів за когнітивною адаптивністю, брала участь у розробці критеріїв зміни інтерфейсів.

Особистий внесок Golovach I.: розробив алгоритм обробки динаміки введення користувацьких подій (мишка, клавіатура) та структурування патернів у реальному часі. Забезпечив реалізацію підсистеми трекінгу поведінки.

Особистий внесок Radchenko A.: технічне впровадження моделей аналізу поведінкових сигналів, модуль взаємодії користувача з системою.

Особистий внесок Radchenko K.: реалізував систему логування та збору поведінкових даних. Забезпечив інтерфейс для зворотного зв’язку користувача.

Особистий внесок Zaloznyi T.: відповідав за тестування адаптивних сценаріїв, оцінку *UX*-показників, провів статистичний аналіз ефективності запропонованих змін у дизайні інтерфейсів.

8. Крант Д.В. Методи визначення водія за стилем його водіння. *Політ. Сучасні проблеми науки*: тези доповідей XXI*V* Міжн. наук.-практ. конф. здобувачів вищої освіти і молодих учених. (Україна, м. Київ, 2-4 квітня, 2024 р.) К.: НАУ, 2024. С. 116-117.

9. Крант Д.В., Артамонов Є.Б., Головач Ю.Ю., Залозний Т.І., Радченко А.В., Радченко К.М. Підходи до визначення користувачів програмних комплексів за поведінковими факторами. *Кібербезпека: актуальні питання та шляхи їх вирішення*: тези наук.-практ. конф. (Україна, с. Світязь, 13-16 червня 2024 р.). Національний авіаційний університет. К.: Вид-во НАУ, 2024. С. 14-16.

Особистий внесок автора: запропонував підхід до класифікації стилю взаємодії користувача на основі шаблонів навігації, швидкості реакцій та шаблонів дій. Розробив модель категоризації користувачів за активністю та вніс практичне обґрунтування застосування алгоритмів адаптації.

Особистий внесок Артамонова Є.Б.: обґрунтував архітектуру адаптивного інтерфейсу, визначив наукову проблематику взаємодії між рівнем когнітивного навантаження користувача та поведінковою динамікою при роботі з ПЗ.

Особистий внесок Головача Ю.Ю.: структурування патернів у реальному часі та забезпечення реалізації підсистеми трекінгу поведінки.

Особистий внесок Залозного Т.І.: відповідав за тестування адаптивних сценаріїв, оцінку *UX*-показників, провів статистичний аналіз ефективності запропонованих змін у дизайні інтерфейсів.

Особистий внесок Радченка А.В.: здійснив технічне впровадження моделей аналізу поведінкових сигналів, розробив модулі взаємодії користувача з системою в умовах змін контексту (мобільний/десктоп режим).

Особистий внесок Радченка К.М.: реалізував систему логування та збору поведінкових даних. Забезпечив інтерфейс для зворотного зв’язку користувача (*feedback-loop*) для корекції адаптацій.

10. Крант Д.В., Дехтяренко А.Т. Аналіз моделей передачі даних між системами транспортних засобів. *Інтелектуальні технології лінгвістичного аналізу*: тези доповідей міжн. наук.-техн. конф. (23-24 жовтня 2024 р.). К.: НАУ, 2024. С. 51.

Особистий внесок автора: опис гібридних моделей.

Особистий внесок Дехтяренко А.Т.: адаптація на рівні систем.

11. Крант Д.В., Гончарук Ю.М. Особливості використання шин передачі даних в транспортних засобах. *Сучасні тенденції розвитку системного програмування*: тези доп. наук.-практ. конф. (Україна, м. Київ, 21-22 листопада 2024 р.). К.: ДНП «ДУ «КАІ», 2025. С. 22-23.

Особистий внесок автора: аналіз переваг *CAN* у сучасних системах.

Особистий внесок Гончарука Ю.М.: підбір і аналіз прикладів.

ЗМІСТ

ПЕРЕЛІК УМОВНИХ СКОРОЧЕНЬ

ABS (*Anti-lock Braking System*) – антиблокувальна система гальмування

ADAS (*Advanced Driver Assistance Systems*) – передові системи допомоги водієві

AEB (*Automatic Emergency Braking*) – автоматичне екстрене гальмування

CAN (*Controller Area Network*) – мережа контролерів транспортного засобу

CRC (*Cyclic Redundancy Check*) – циклічна надлишкова перевірка

ECU (*Electronic Control Unit*) – електронний блок керування

ESC (*Electronic Stability Control*) – електронний контроль стабільності

ESP (*Electronic Stability Program*) – система курсової стійкості

FlexRay – високошвидкісна автомобільна шина передачі даних

HVAC (*Heating, Ventilation, and Air Conditioning*) – система опалення, вентиляції та кондиціювання повітря

LIN (*Local Interconnect Network*) – локальна мережа для взаємодії малокритичних вузлів

MAC (*Media Access Control*) – керування доступом до середовища передачі даних

MOST (*Media Oriented Systems Transport*) – автомобільна шина для передачі мультимедійних даних

OTA (*Over-the-Air*) – бездротове оновлення програмного забезпечення

SAE (*Society of Automotive Engineers*) – Товариство автомобільних інженерів

SPE (*Single Pair Ethernet*) – однопарний *Ethernet*

TSN (*Time-Sensitive Networking*) – мережа, чутлива до затримок

V2X (*Vehicle-to-Everything*) – комунікація транспортного засобу з іншими об’єктами інфраструктури та транспортними засобами

СКВ – середньоквадратичне відхилення (стандартне відхилення)

ТЗ – транспортний засіб

ВСТУП

Актуальність теми.

Зростання рівня автоматизації транспортних засобів супроводжується істотним ускладненням внутрішньої комунікаційної архітектури, що об’єднує десятки електронних блоків керування, сенсорів і виконавчих механізмів. Шини передачі даних виступають критично важливими каналами для забезпечення взаємодії між цими компонентами, в яких зростає потреба не лише в забезпеченні надійного обміну, а й у можливості оцінки якості, змісту та достовірності даних, що передаються.

Існуючі протоколи комунікації здебільшого не передбачають механізмів вбудованої оцінки або фільтрації інформації на рівні шин. Це створює загрозу неконтрольованої циркуляції помилкових, спотворених або шкідливих повідомлень, особливо в умовах кібератак або технічних збоїв. Саме тому виникає необхідність розробки методів, які б дозволяли аналізувати параметри трафіку, структуру повідомлень та їх відповідність очікуваним патернам, з метою оперативної оцінки їх безпечності та достовірності [1; 15; 65; 86; 87].

Це обумовлює актуальність дослідження методів використання шин передачі даних в автоматизованих системах транспортних засобів з урахуванням можливості оцінки якості та змісту переданої інформації, як у контексті підвищення безпеки, так і в аспекті забезпечення стабільного функціонування складних систем керування.

Зв’язок роботи з науковими програмами, планами, темами, грантами. Дисертаційна робота є складовою частиною досліджень, що проводяться в КАІ і спрямовані на на вдосконалення методів забезпечення надійної та безпечної передачі даних в автоматизованих системах керування транспортними засобами. Ці дослідження охоплюють широкий спектр завдань – від розробки інтелектуальних алгоритмів взаємодії між електронними блоками до впровадження механізмів оцінки достовірності інформації в умовах високої динаміки руху та обмежених обчислювальних ресурсів. Зокрема, робота спрямована на розвиток науково-технічного підґрунтя для побудови нових моделей обміну даними в середовищі з неоднорідною шиною, врахування контексту дорожньої ситуації та поведінки водія при формуванні інформаційних повідомлень, а також оптимізацію архітектури

систем управління з урахуванням вимог до реального часу, стійкості до збоїв і кібератак. Отримані результати дослідження інтегруються в єдину концепцію створення адаптивних, інтероперабельних систем передачі даних для транспортних платформ нового покоління.

Держбюджетна (кафедральна) науково-дослідна робота № 26-2024/14.03 «Підвищення достовірності цифрової обробки зображень з бортової відеокамери БПЛА».

Мета і завдання дослідження – розробка, обґрунтування та експериментальна перевірка методів використання шин передачі даних у багатокомпонентних автоматизованих системах транспортних засобів з урахуванням вимог до продуктивності, синхронізації, безпеки та масштабованості.

Основні завдання:

– провести системний аналіз сучасних типів шин передачі даних, що використовуються у транспортних засобах різних класів;

– побудувати модель функціонування шинної архітектури з урахуванням рівня автоматизації транспортного засобу;

– розробити метод виявлення порушень у *CAN*-шинах з використанням байєсівського підходу;

– провести експериментальне тестування розроблених методів передачі даних у типових конфігураціях автоматизованих транспортних систем та здійснити їх оптимізацію з урахуванням вимог до затримки, надійності й пропускної здатності;

– обґрунтувати рекомендації щодо вибору шин і протоколів для різних конфігурацій транспортних систем.

Об’єкт дослідження – процеси інформаційної взаємодії між електронними компонентами автоматизованих систем управління транспортними засобами.

Предмет дослідження – методи організації та оцінки передачі даних через комунікаційні шини у багатокомпонентних архітектурах транспортних засобів з різними рівнями автоматизації.

Методи дослідження. Теоретичною основою дослідження стали методи системного аналізу, які дозволили розглянути транспортний засіб як сукупність

взаємодіючих електронних та інформаційних підсистем, пов’язаних між собою шинами передачі даних. Метод формалізації було застосовано для опису структури транзакцій, протоколів та патернів обміну повідомленнями, що стало передумовою побудови математичних моделей.

Для моделювання ймовірнісних залежностей між характеристиками повідомлень та їх безпековим статусом було використано методи байєсівської статистики, які дали змогу оцінити апостеріорну ймовірність безпечності кожної транзакції з урахуванням поведінкових і контекстуальних факторів. Методи кластеризації ймовірнісних процесів дозволили ідентифікувати патерни типових режимів роботи, а також виявляти аномальні послідовності.

Для реалізації адаптивного підходу до оновлення параметрів моделі застосовувались методи оптимізації, зокрема алгоритм *Adam*, який базується на градієнтному спуску з урахуванням моментів, що забезпечує швидку збіжність навіть за умов зміни поведінки системи.

Наукова новизна отриманих результатів. У процесі вирішення поставлених задач автором розроблено методи підвищення ефективності використання шин передачі даних в автоматизованих системах транспортних засобів шляхом оцінки достовірності, адаптивного управління потоками повідомлень та забезпечення міжсистемної сумісності, з урахуванням контексту руху та поведінки систем. Наукові результати базуються на таких основних положеннях:

уперше:

– обґрунтовано метод визначення ймовірнісної оцінки достовірності даних у шинах передачі повідомлень між компонентами автоматизованих систем керування транспортних засобів з урахуванням поведінкових і контекстуальних факторів;

– запропоновано узагальнену модель формування та оцінки транзакцій у мультишинному середовищі *CAN*–шини, яка враховує структуру повідомлення, часові характеристики, маршрутизацію та динаміку роботи підсистем;

– розроблено алгоритм адаптивної фільтрації повідомлень у транспортних мережах з підтримкою апостеріорної перевірки безпечності передачі на основі апріорних статистичних патернів.

удосконалено:

– підхід до формалізації структури транзакцій у транспортних шинах;

– концептуальну схему побудови інформаційного обміну між підсистемами транспортного засобу на основі оцінювання ризиків аномалій або конфліктів даних;

– підхід до забезпечення інтероперабельності між шинами різного типу, зосереджений на уніфікації структури повідомлень у рамках *CAN*-протоколу.

отримали подальший розвиток:

– класифікація вимог до шин передачі даних залежно від рівня автоматизації транспортного засобу та характеру функціональної задачі;

– методи аналізу часових характеристик передачі в умовах обмеженої пропускної здатності та високих вимог до надійності;

– оцінка архітектурних тенденцій у розвитку шинних систем нового покоління.

Практичне значення одержаних результатів. Розроблені моделі можуть бути інтегровані в електронні блоки керування для реального моніторингу *CAN*-трафіку і виявлення загроз без використання нейронних мереж. Методика побудови адаптивної безпекової моделі може бути використана в умовах обмежених обчислювальних ресурсів вбудованих систем. А підхід до використання даних напряму з *CAN*-шини для оцінки поведінкових параметрів водія відкривають широкі можливості впровадження даного методу у всіх системах, що передбачають управління оператором.

Практичні результати можна звести до наступних пунктів:

1. Розроблено метод оцінки достовірності даних у транспортних шинах, що базується на апостеріорному аналізі транзакцій з урахуванням поведінкових патернів та контексту руху. Метод реалізовано у вигляді математичного апарату, придатного для застосування у вбудованих мікроконтролерних системах.

2. Реалізовано алгоритм адаптивного фільтрування повідомлень у мультишинному середовищі (*CAN–FlexRay–Ethernet*) з підтримкою механізму категоризації повідомлень за рівнем довіри. Це дозволяє виявляти ін’єкції та порушення синхронності без суттєвого збільшення обчислювального навантаження.

3. Розроблено уніфіковану модель взаємодії між підсистемами транспортного засобу, яка підтримує інтероперабельність між різними типами шин через нормалізовану структуру транзакцій та врахування часових характеристик передачі.

4. Розроблені в дисертаційні роботі методи класифікації використані ТОВ «АЕРОФАБ УКРАЇНА» (м. Київ) для визначення рівня підготовки операторів БПЛА при навчанні на тренажері з активними динамічними елементами, що сприяло полегшенню категоризації операторів на основі їх дій в програмному тренажері (акт впровадження від 24.04.2025 р.).

5. Результати дисертаційної роботи запроваджено у навчальний процес кафедри інтелектуальних кібернетичних систем як матеріал лекцій та практичних занять з дисциплін «Дослідження і проєктування вбудованих і мобільних систем» (навчальний план №НМ-4-123-2/21, затверджений 29.04.21, та робочий навчальний план № РМ-4-123-2/24, затверджений 02.09.2024) та «Системне програмування» (навчальний план № НБ-4-123-2/21, затверджений 29.04.21, та робочий навчальний план № РБ-4-123-2/24, затверджений 16.04.2024), а також в бакалаврських та магістерських кваліфікаційних роботах для студентів спеціальності 123 «Комп'ютерна інженерія» ОП «Системне програмування» (акт про впровадження від 15.04.2025 р.).

Особистий внесок здобувача. Найважливіші ідеї, висновки, рекомендації, отримані в дисертації, оприлюднені на наукових та науково-практичних конференціях, у тому числі міжнародних, всеукраїнських та за міжнародною участю: "Сучасні тенденції розвитку системного програмування" (Київ, 2021, 2022, 2023, 2024), "*International Conference on Advanced Trends in Information Theory*" (*Kyiv*, 2022); "Актуальні проблеми управління інформаційною безпекою держави " (Київ, 2023); "Інформаційно-комп'ютерні технології" (Житомир, 2023), "Актуальні проблеми науки, освіти і технологій" (Словаччина, м. Братислава, 2023), "*Cyber Hygiene & Conflict Management in Global Information Networks*" (*Kyiv*, 2024), "Політ. Сучасні проблеми науки" (Київ, 2024), "Кібербезпека: актуальні питання та шляхи їх вирішення" (Україна, с.Світязь, 2024), "Інтелектуальні технології лінгвістичного аналізу" (Київ, 2024).

Апробація результатів дисертації. Основні положення дисертаційної роботи доповідались та представлялися на міжнародних науково-технічних конференціях

Публікації. Основні положення та результати дисертаційного дослідження викладено в 19 наукових публікаціях, серед них 5 публікацій у наукових фахових виданнях України, 2 у виданнях, проіндексованих в базі даних *Scopus*, 1 колективна монографія, 11 публікацій у збірниках матеріалів конференцій.

РОЗДІЛ 1. АНАЛІЗ СТАНУ ПРОБЛЕМИ ТА ПОСТАНОВКА ЗАДАЧІ ДОСЛІДЖЕННЯ

1.1. Огляд існуючих підходів до використання шин передачі даних в транспортних засобах

Початок розвитку комунікаційних систем у транспортних засобах був тісно пов’язаний із впровадженням електроніки в автомобілебудування у 1970-х роках. У той період для обміну даними між компонентами використовувалися прості аналогові сигнали, які передавались по виділених лініях між кожною парою пристроїв. Така архітектура мала низку суттєвих недоліків – обмежену масштабованість, складність у прокладці проводки та відсутність гнучкості в оновленні системи.

Один із ключових напрямів розвитку сучасних транспортних засобів полягає у впровадженні багатошинних архітектур обміну даними, які забезпечують надійний, масштабований та синхронізований обмін інформацією між численними електронними блоками. Найбільш поширеними залишаються шини типу *CAN*, що характеризуються високою стійкістю до завад та ефективним арбітражем доступу до середовища передачі. Водночас, дослідники відзначають її обмежену пропускну здатність, що стає критичною у складних автоматизованих системах [20; 45; 48; 75; 83; 84]. З метою подолання цих обмежень інтегруються додаткові шини, зокрема *FlexRay*, яка демонструє переваги у точності синхронізації та забезпеченні реального часу [15], та *Automotive Ethernet*, що відкриває можливості високошвидкісної передачі мультимедійних та діагностичних потоків [44].

Серед існуючих підходів до організації передачі даних варто відзначити дослідження, присвячені інтероперабельності між шинами різного типу: в роботі [41] запропоновано метод уніфікованої маршрутизації повідомлень у мультишинному середовищі, де враховуються часові вимоги та пріоритети задач, а у роботах [24; 43] досліджено вплив архітектури шин на якість передачі в умовах високої завантаженості мережі та можливих атак на систему, що є актуальним для автономних та напівавтономних транспортних засобів. Крім того, важливою складовою є аналіз безпеки обміну в шині *CAN*, де виявлено вразливості в обробці

помилок [72; 73] та можливості реалізації атак типу «*ransomware*» через інтерфейс бортової мережі [9]. Таким чином, у сучасній науковій літературі простежується тенденція до побудови адаптивних, безпечних та масштабованих систем шинного обміну, що враховують не лише технічні характеристики протоколів, а й особливості поведінки компонентів та контекст експлуатації транспортного засобу.

Зі зростанням кількості електронних блоків керування (*Electronic Control Unit – ECU*) в автомобілях виникла потреба у стандартизованій багатоточковій шині, яка дозволила б зменшити кількість проводів, знизити масу, спростити обслуговування та забезпечити надійний обмін інформацією між компонентами. У 1980-х роках компанія *Bosch* запропонувала *CAN* – цифрову шину передачі даних, яка стала революційним проривом у галузі. *CAN* дозволяла кільком пристроям підключатися до єдиної шини і здійснювати обмін повідомленнями без потреби в централізованому контролері.

Подальший розвиток автомобільної комунікації включав поетапне вдосконалення і спеціалізацію шин залежно від типів систем, які вони обслуговують. Для менш критичних функцій, таких як управління кліматом або вікнами, було розроблено *Local Interconnect Network* (*LIN*) – дешевшу альтернативу *CAN*, орієнтовану на невисоку швидкість обміну (до 20 кбіт/с) і просту топологію. У відповідь на потребу у швидкій передачі мультимедійних даних з’явилася *Media Oriented Systems Transport* (*MOST*) – шина, розрахована на передачу аудіо- та відеопотоків у реальному часі.

Для систем активної безпеки та автономного керування, що вимагають високої пропускної здатності та мінімальних затримок, були розроблені шини *FlexRay* (до 10 Мбіт/с, підтримка синхронної комунікації) та *Automotive Ethernet*, яка забезпечує швидкість до 1000 Мбіт/с та підтримує стандартні *IP*-протоколи.

Еволюція шин передачі даних у транспорті пройшла шлях від аналогових провідникових рішень до багаторівневих цифрових протоколів, які охоплюють як прості допоміжні системи, так і високопродуктивні критичні модулі автономного керування. Ця еволюція стала фундаментом для розробки автоматизованих

транспортних засобів нового покоління, у яких ефективна міжсистемна комунікація є критично важливим компонентом безпеки та функціональності.

Сучасні транспортні засоби оснащені великим числом різноманітних електронних систем, які взаємодіють між собою через внутрішні мережі даних [77]. Для забезпечення комунікації між електронними блоками керування (*ECU*) використовуються декілька основних типів шин, кожна з яких має свої архітектурні особливості, функціональне призначення та технічні характеристики.

CAN – це найпоширеніша шина даних у транспортних засобах, розроблена компанією *Bosch* у 1983 році. Її основна архітектурна особливість – це багатоточкова топологія з двома провідниками (*CAN High* і *CAN Low*), які з'єднують усі *ECU* в мережі. Передача даних організована за принципом "*message-based*", де кожне повідомлення має ідентифікатор пріоритету, а контроль за доступом до шини здійснюється за допомогою алгоритму арбітражу.

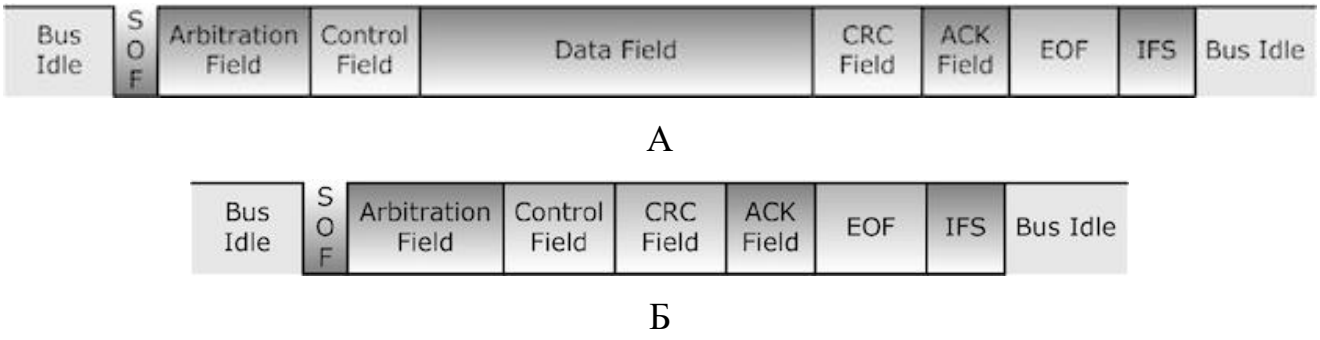

Рис. 1.1. Структура повідомлень в шині *CAN*: А) *Data Frame,* 2) *Remote Frame* [20]

CAN підтримує швидкість до 1 Мбіт/с (у класичному варіанті) та забезпечує високу надійність завдяки вбудованим механізмам перевірки *CRC,* підтвердження прийому та виявлення помилок. Удосконалений варіант – *CAN FD* (*Flexible Data Rate*) – дозволяє передавати більші обсяги даних з динамічною швидкістю до 8 Мбіт/с.

LIN – це одноканальна шина для малокритичних функцій, таких як управління освітленням, кліматом чи склоочисниками. Стандарт розроблено як доповнення до *CAN*. Архітектурно *LIN* реалізує майстер-слейв модель: один головний пристрій (*master*) ініціює всі комунікації, а підлеглі пристрої (*slaves*) реагують відповідно до розкладу. Структура розподілу біт представлена на рис. 1.2.

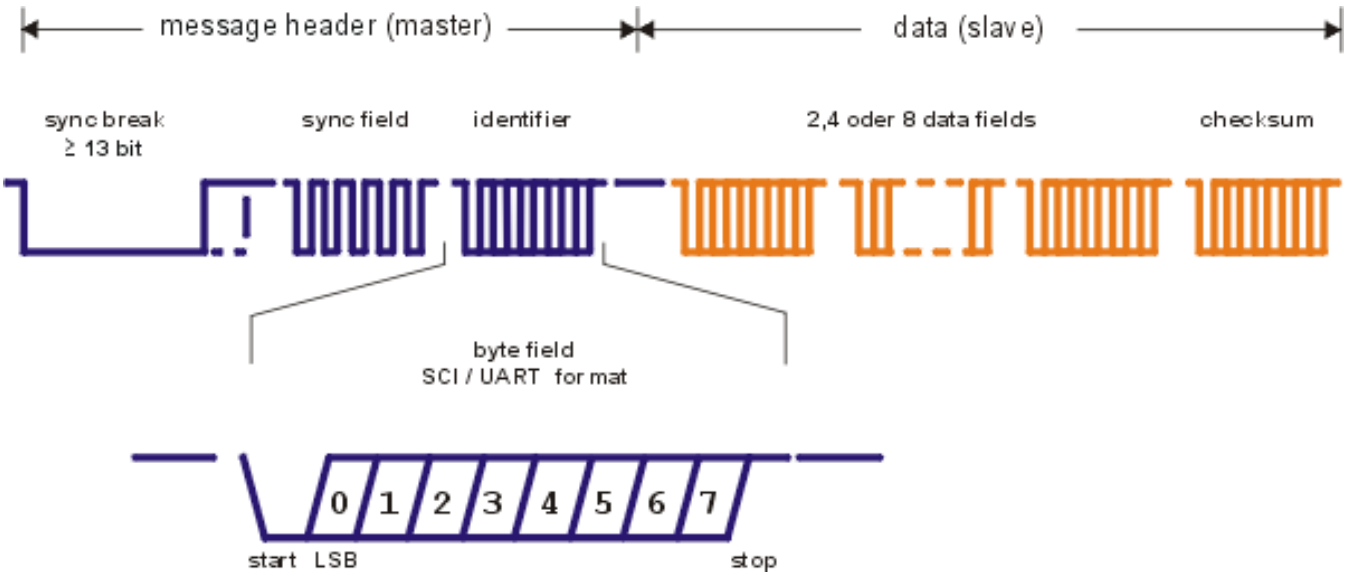

Рис. 1.2. Структура даних, що передаються шиною *LIN* [36]

Максимальна швидкість *LIN* становить 20 кбіт/с, що достатньо для низькопріоритетних задач. Важливою перевагою *LIN* є її простота реалізації та низька вартість, що робить її придатною для масового застосування.

FlexRay – це шина, призначена для систем, критичних до часу, зокрема для електронної стабілізації, активної підвіски та автономного керування. *FlexRay* використовує подвійний канал зв’язку для підвищеної надійності (*fault-tolerant*), а її гібридна модель синхронного/асинхронного доступу дозволяє поєднувати високошвидкісну передбачувану комунікацію з гнучкою передачею даних.

FlexRay підтримує швидкість передачі до 10 Мбіт/с і відзначається жорсткою синхронізацією, що забезпечує мінімальні затримки. Висока вартість та складність у конфігурації обмежили її застосування, проте вона досі використовується в преміальних і автономних транспортних засобах.

MOST – це спеціалізована шина для мультимедійних систем: аудіо, відео, навігації, телевізійних та інфо-розважальних інтерфейсів. Вона використовує оптоволоконне або електричне середовище для передачі даних зі швидкістю до 150 Мбіт/с у останніх реалізаціях (*MOST*150).

Архітектура *MOST* побудована на концепції циркульної передачі даних між вузлами (*ring topology*), що забезпечує визначеність у часі та синхронізацію потоків.

Інтеграція з операційними системами транспортного засобу дозволяє реалізовувати складні сценарії розподілу мультимедійних даних між користувачами.

Automotive Ethernet – найновіший стандарт серед автомобільних шин, який забезпечує надвисоку пропускну здатність (до 1 Гбіт/с) і сумісний з традиційною *IT*-інфраструктурою. Використовується для підключення камер, радарів, *LIDAR*-датчиків і модулів обробки відео в системах допомоги водієві (*ADAS*) та автономного керування.

Його головна перевага – підтримка *IP*-протоколів, що дозволяє інтегрувати автомобіль у мережу *IoT* та використовувати стандартні методи кібербезпеки, маршрутизації та діагностики. Автомобільний *Ethernet* часто використовує однопарні кабелі (*Single Pair Ethernet – SPE*), що знижує вагу і вартість проводки.

CAN-шина залишається провідною завдяки поєднанню низької вартості, простої реалізації, високої надійності та великої кількості готових рішень. Вона добре масштабована у межах критично важливих і не надто вимогливих за швидкістю підсистем. Масова підтримка з боку виробників чипів і контролерів, наявність стандартів (*ISO* 11898), відпрацьовані механізми виявлення помилок та арбітражу – усе це робить *CAN* універсальним рішенням для більшості систем керування, навіть попри відсутність вбудованого захисту. У сучасних автомобілях *CAN* часто поєднується з іншими шинами, що дозволяє адаптувати її до гібридної архітектури без потреби у повній заміні.

В умовах зростаючої складності електронних систем у транспортних засобах питання виявлення атак та забезпечення цілісності переданої по шині інформації набуває особливого значення. У дослідженні [25] запропоновано метод виявлення аномалій на шині *CAN*, що ґрунтується на аналізі ентропії періодичних повідомлень. Такий підхід дозволяє виявити приховані атаки, які імітують легітимну активність, зокрема шляхом незначного варіювання частоти передавання кадрів.

З точки зору безпеки, перспективним напрямом є використання систем *VANET* (*Vehicular Ad Hoc Networks*), які забезпечують передачу повідомлень між транспортними засобами без необхідності централізованої інфраструктури. Робота [27] демонструє можливість застосування таких систем для виявлення та

попередження загроз, а також для покращення реакції транспортного засобу на змінні умови руху. У поєднанні з внутрішніми шинами передачі даних, *VANET* може забезпечити більш адаптивну й надійну взаємодію між автомобілями та інфраструктурою.

Оцінка переваг і недоліків сучасних шин передачі даних у транспортних засобах різних класів ґрунтується на аналізі їхніх функціональних можливостей, рівня автоматизації, обсягів даних та вимог до безпеки. *CAN*-шина, яка є домінуючим стандартом у більшості легкових авто, вирізняється простотою реалізації, високою стійкістю до завад і широкою підтримкою апаратного забезпечення [7; 15]. Проте її обмеження щодо пропускної здатності (до 1 Мбіт/с) і відсутність вбудованих засобів захисту роблять її малопридатною для систем автономного керування або для великотранспортних засобів, де необхідна синхронна передача значних обсягів даних у режимі реального часу [9]. Шина передачі даних відіграє критичну роль у забезпеченні синхронної роботи підсистем навігації, стабілізації та прийняття рішень, які для автономних систем на стільки ж важливі, як в системах управління літальних апаратів [50].

Для складніших застосувань, таких як грузові автомобілі, автобуси чи безпілотні транспортні платформи, все більшого поширення набувають шини *FlexRay* і *Automotive Ethernet*. *FlexRay* забезпечує детерміновану передачу даних з високою точністю синхронізації та підтримкою двоканального режиму, що підвищує надійність. Однак її впровадження обмежується високою вартістю та складністю конфігурації. Натомість *Automotive Ethernet* відкриває можливості для інтеграції мультимедійних потоків, діагностики та *OTA*-оновлень, пропонуючи гнучку масштабованість та швидкість до 1 Гбіт/с. Проте еталонні профілі безпеки *Ethernet* у транспорті все ще розвиваються, і їх реалізація потребує глибокої інженерної інтеграції та адаптації до вимог автомобільної галузі [77; 83]. Таким чином, вибір шинної архітектури має базуватись на балансі між функціональністю, вартістю, надійністю та адаптивністю до задач транспортного засобу конкретного класу.

У вантажному транспорті основний акцент робиться на надійність, діагностику та інтеграцію з телеметрією:

1) *CAN* (*J*1939) – стандарт для вантажівок та сільськогосподарської техніки. Переваги: розширений набір параметрів (*SPN*), витривалість, діагностика на великі відстані. Недоліки: затримки при високому навантаженні, складність зміни топології.

2) *Ethernet* – зростаючий тренд для діагностики в реальному часі, *OTA*-оновлень. Переваги: швидкість, масштабованість. Недоліки: потребує нових протоколів безпеки та апаратної підтримки.

Ці категорії вимагають синхронізації, реального часу та адаптивності до складних умов.

CAN FD / *TSN Ethernet* – все частіше використовуються як заміна *FlexRay*. Переваги: розширена швидкість (*CAN FD*), сумісність зі старими системами. Недоліки: потреба в ретельному налаштуванні та тестуванні нових протоколів.

Вибір шини передачі даних напряму залежить від класу транспортного засобу, вартості, вимог до швидкості та надійності. Хоча *Ethernet* поступово витісняє інші шини в нових архітектурах, *CAN* залишається золотою серединою завдяки оптимальному балансу між функціональністю, ціною та підтримкою. У спеціалізованих системах, де важлива жорстка детермінованість, застосовуються або *FlexRay*, або поєднання *Ethernet TSN* з *CAN FD*.

1.2. Аналіз досліджень щодо автоматизованих систем транспортних засобів

Розвиток автоматизованих систем у транспортних засобах зумовив стрімке зростання вимог до інформаційної взаємодії між їхніми електронними компонентами. У роботах [1; 13; 28; 29; 59] розглядаються не тільки архітектурні моделі сучасних транспортних систем, де підсистеми керування, діагностики, навігації та безпеки функціонують у єдиному інформаційному середовищі, а і небезпеки атак на ці системи. Особливу увагу приділено викликам, що виникають унаслідок інтеграції великої кількості сенсорів, контролерів і виконавчих пристроїв, які повинні взаємодіяти в реальному часі з високим рівнем достовірності даних. У роботах [9; 16; 28] обґрунтовано доцільність використання багаторівневих моделей керування, що дозволяє розмежувати задачі локальної обробки сигналів і централізованого аналізу поведінки транспортного засобу.

Крім технічного аспекту, дослідження зосереджуються і на питаннях безпеки автоматизованих систем. Наприклад, у [4; 30; 45] аналізуються вектори потенційних кібератак через внутрішні шини передачі даних, зокрема *CAN*, що не передбачає аутентифікації чи шифрування повідомлень. У зв'язку з цим актуальними стають підходи до поведінкового аналізу трафіку та виявлення аномалій, що пропонуються у [5; 36; 40]. Такі методи спрямовані на підвищення надійності та безпеки систем автономного керування шляхом побудови адаптивних моделей нормальної поведінки транспортних компонентів.

У той же час, у контексті загального аналізу автоматизованих систем управління транспортними засобами та способів обробки неоднорідних даних, цінним є напрацювання, представлене в колективній монографії [32]. Зокрема, в розділі 4 розглянуто науково-методичний апарат обробки багатоформатних інформаційних потоків, що виникають у системах управління, подібних до автомобільних. Такий підхід дозволяє не лише структурувати дані, а й ефективно адаптувати аналітичні алгоритми до потоків з *CAN*-шини.

Сучасні дослідження автоматизованих систем охоплюють не лише архітектурні та функціональні аспекти, а й критично важливі питання забезпечення стійкості до збоїв і зовнішніх втручань.

1.2.1. Аналіз використання автоматизованих систем в транспортних засобах

Автоматизовані системи в транспортних засобах є критично важливим компонентом сучасної автомобільної електроніки. Вони класифікуються за функціональним призначенням на три основні групи: системи безпеки, системи комфорту та системи автономного керування. Кожна з цих груп включає численні підсистеми, які взаємодіють між собою через бортові комунікаційні шини, зокрема *CAN*, *FlexRay* та *Automotive Ethernet*.

Автоматизовані системи у транспортних засобах дедалі частіше використовуються для підвищення рівня безпеки, ефективності керування та комфорту пасажирів. Наукові дослідження [1; 7; 22] демонструють, що ключовими функціональними компонентами сучасних транспортних платформ є адаптивні круїз-

контролі, автоматичне екстрене гальмування, системи утримання в смузі та інші модулі автономного або напівавтономного керування. Їх ефективна взаємодія можлива лише за умови швидкої та надійної передачі даних між електронними блоками керування, що обумовлює актуальність дослідження шинної архітектури.

У роботах [9; 28; 33] підкреслюється важливість побудови ієрархічних структур обміну інформацією, де підсистеми нижчого рівня забезпечують оперативне керування виконавчими механізмами, тоді як системи вищого рівня формують глобальні стратегії руху. При цьому зростає потреба у мультишинних середовищах (*CAN, FlexRay, Ethernet*), здатних забезпечити одночасну підтримку як критичних до затримки, так і високопродуктивних потоків даних. З огляду на це, дослідження автоматизованих транспортних систем сьогодні сфокусовані не лише на реалізації окремих функцій, а й на забезпеченні цілісної, масштабованої та безпечної інформаційної архітектури.

Проведемо аналіз основних систем:

1) Системи безпеки – ця категорія охоплює активні та пасивні системи, які покликані мінімізувати ризики ДТП або зменшити наслідки аварій;

– антиблокувальна система гальм (*ABS*) – запобігає блокуванню коліс під час екстреного гальмування, дозволяючи водієві зберігати контроль над керуванням$

– система курсової стійкості (*ESP*/*ESC*) – контролює тягу, кут повороту та бокове прискорення, щоб запобігти заносу$

– подушки безпеки та преднатяжувачі ременів – активуються миттєво у разі зіткнення, отримуючи команду через шину *CAN*;

– автоматичне екстрене гальмування (*AEB*) – виявляє ризик зіткнення та самостійно активує гальма.

2) Системи комфорту – ці системи підвищують зручність, ефективність та знижують навантаження на водія та пасажирів;

– клімат-контроль (*HVAC*) – автоматично підтримує задану температуру, враховуючи зовнішні умови;

– електричні регулювання сидінь, дзеркал, склопідйомників – забезпечують персоналізацію налаштувань;

– мультимедійні системи – інтеграція аудіо, навігації, камери заднього огляду, *Bluetooth*/*CarPlay*;

– інтелектуальне освітлення (*Adaptive Lighting*) – змінює форму променя фар залежно від швидкості та повороту керма.

3) Системи автономного керування – ці системи втілюють функції, притаманні самокерованим автомобілям, які зменшують або усувають необхідність втручання водія;

– адаптивний круїз-контроль (*ACC*) – підтримує безпечну дистанцію до попереднього транспортного засобу;

– *Lane Keeping Assist* (*LKA*) – допомагає утримати автомобіль у смузі;

– системи автоматичного паркування – виконують маневр без участі водія;

– системи розпізнавання дорожніх знаків та розмітки – використовують камери та обробку зображень.

Кожен клас автоматизованих систем виконує специфічну роль у забезпеченні безпеки, комфорту чи автоматизації керування. Розвиток цих систем спричинив значне ускладнення мережевої архітектури транспортного засобу та підвищив вимоги до пропускної здатності та надійності шин передачі даних. Саме тому правильний вибір шинної архітектури – ключ до ефективного функціонування всіх автоматизованих компонентів сучасного авто.

1.2.2. Роль шин передачі даних в структурі електронної архітектури сучасного автомобіля

Сучасний автомобіль являє собою складну кібер-фізичну систему, у якій десятки – а іноді й сотні *ECU* взаємодіють у реальному часі. Відмінною рисою новітніх авто є інтеграція функціонально розподілених систем через мережеві шини, які формують основу електронної архітектури.

У структурі електронної архітектури сучасного автомобіля шини передачі даних відіграють ключову роль, забезпечуючи обмін інформацією між численними *ECU*, датчиками, виконавчими механізмами та комунікаційними модулями. Як зазначено у [9; 16; 24], сучасні транспортні засоби використовують мультишинну

архітектуру, яка включає *CAN, LIN, FlexRay, MOST* та *Automotive Ethernet*, кожна з яких відповідає за передачу певного типу даних залежно від критичності, обсягу та вимог до затримки. Наприклад, *CAN*-шина широко використовується для управління двигуном, гальмівною системою та підвіскою завдяки своїй надійності й детермінованості.

Розвиток функціоналу автомобілів, зокрема впровадження систем *ADAS* та компонентів автономного керування, призвів до зростання обсягів даних, які необхідно передавати в режимі реального часу.

У дослідженнях [3; 31; 36] вказано, що традиційні шини, такі як *CAN* та *LIN*, поступово доповнюються або витісняються високошвидкісними рішеннями, зокрема *FlexRay* і *Ethernet*, що здатні забезпечити більшу пропускну здатність та підтримку паралельного передавання даних.

Відтак, шини є не лише фізичним середовищем передавання, а й основою для реалізації складної, багаторівневої електронної архітектури, що забезпечує узгоджену роботу усіх інтелектуальних систем транспортного засобу.

Електронна архітектура сучасного автомобіля поділяється на три логічні рівні:

1. Сенсорно-актуаторний рівень – складається з датчиків (температури, тиску, положення педалей, камери, *LiDAR*) і виконавчих пристроїв (моторчики, клапани, реле).

2. Функціональний рівень (*ECU*) – набір мікроконтролерів і процесорів, кожен з яких відповідає за певну функцію: *ABS, ESP*, двигун, клімат, мультимедіа, тощо.

3. Комунікаційний рівень (шини передачі даних) – системи для зв’язку між блоками, які забезпечують передачу даних і команд.

Ці блоки об’єднані в доменну архітектуру (наприклад, домен шасі, кузова, інформаційно-розважальний домен, *ADAS*-домен), що стало особливо актуальним з появою автономного керування.

Передача даних між *ECU* в транспортному засобі здійснюється через шини передачі даних, які обираються залежно від технічних вимог до швидкості, часу затримки, обсягу переданих даних і критичності функцій. Для обміну інформацією, що не є критичною до часу (наприклад, керування кліматичною системою або

мультимедіа), зазвичай застосовується *CAN*-шина з пропускною здатністю до 1 Мбіт/с, яка забезпечує достатню надійність та економічність у межах локального сегмента [9; 15; 26].

Водночас для функціонування систем активної безпеки (*ABS, ESP*), адаптивного круїз-контролю або автоматизованого керування, необхідні шини з нижчим рівнем затримки та вищою пропускною здатністю. У таких випадках застосовуються *FlexRay* (до 10 Мбіт/с), а також *Automotive Ethernet* (100 Мбіт/с і вище), що дозволяють забезпечити одночасну синхронізацію численних *ECU* та швидке реагування на події в реальному часі [31; 42; 44; 52].

Критерій вибору шини також враховує тип переданих даних (цифрові чи аналогові), їх періодичність та необхідність у пріоритетизації трафіку, що є визначальним у розробці електронної архітектури сучасного автомобіля. (табл. 1.1).

Таблиця 1.1

Передача даних між електронними блоками керування

Шина	Швидкість	Призначення	Приклади використання
CAN	до 1 Мбіт/с	Критично важливі системи: двигун, *ABS, ESP*	*Bosch CAN* в *Mercedes, Toyota, Ford*
LIN	до 20 кбіт/с	Системи комфорту: дзеркала, склопідйомники	*BMW* 3 *Series, VW Golf*
FlexRay	до 10 Мбіт/с	Системи активної безпеки та шасі	*BMW X*5, *Audi A*8
MOST	до 150 Мбіт/с	Мультимедіа, навігація, камери	*Audi MMI, BMW iDrive*
Automotive Ethernet	до 10 Гбіт/с	*ADAS*, автономне водіння, *OTA*-оновлення	*Tesla Model S, VW ID*.4, *GM Ultra Cruise*

CAN-шина залишається основною через універсальність, простоту реалізації та низьку вартість. У 2020-х роках з’явилися її вдосконалення: *CAN FD* (*Flexible Data-*

rate), що дозволяє збільшити розмір кадру до 64 байтів і підтримує швидкість до 8 Мбіт/с.

Приклади промислових архітектур і готових рішень:

– *Volkswagen* використовує доменну архітектуру з розділенням на три головні сегменти: керування кузовом, силова установка та інформаційно-розважальний блок, з'єднані через *CAN*, *LIN* та *Ethernet*. Рішення реалізоване в *VW MEB*-платформі для електромобілів;

– *Tesla* розробила власну високошвидкісну архітектуру на основі *Automotive Ethernet*, що з'єднує централізовані комп'ютерні блоки (*Full Self-Driving Computer*) з периферійними пристроями. Відомо, що *Tesla* використовує до 100 *ECU* у *Model* 3;

– *Continental Automotive* пропонує рішення *CAEdge* – універсальну хмарну платформу з *ECU*, яка підтримує *ADAS*, *V2X* та керування оновленнями через *Ethernet*/*CAN*.

1.2.3. Взаємозалежність між рівнем автоматизації та вимогами до пропускної здатності шин

Взаємозалежність між рівнем автоматизації транспортного засобу та вимогами до пропускної здатності шин передачі даних є визначальною характеристикою при побудові електронної архітектури сучасних автомобілів. Зі зростанням рівня автоматизації – від систем допомоги водієві (*ADAS*) до повністю автономного керування – збільшується кількість сенсорів, виконавчих пристроїв і модулів обробки даних, що створює суттєве навантаження на комунікаційні шини.

У роботах [3; 36; 44] наголошується, що для транспортних засобів з рівнем автоматизації *SAE Level* 3 і вище типова пропускна здатність шин на кшталт *CAN* стає недостатньою для забезпечення своєчасної обробки великих обсягів критичних даних з камер, лідарів, радарів та інших сенсорів.

Це зумовлює перехід до гібридних архітектур із використанням високошвидкісних шин, зокрема *FlexRay*, *Automotive Ethernet* або *Time-Sensitive Networking* (*TSN*). Як зазначено у [24; 31; 52], зростання вимог до затримки та синхронізації даних, а також необхідність підтримки реального часу при керуванні

транспортним засобом, формують нові технічні критерії до шин. Таким чином, рівень автоматизації прямо впливає на вибір типу шини, її топологію, частоту оновлення повідомлень, механізми арбітражу та контроль часу доставки, що потребує детального аналізу під час проєктування кожного нового покоління транспортних систем.

Автоматизація транспортних засобів значно змінює вимоги до внутрішніх мереж автомобіля. Зі зростанням кількості сенсорів, обчислювальних блоків та потреб у швидкій взаємодії ззовні, виникає потреба у шинах з вищою пропускною здатністю, меншою затримкою та гарантією детермінованої доставки.

Міжнародний стандарт *SAE J*3016 класифікує рівні автоматизації на шість ступенів:

– *Level* 0 (*No automation*) – водій виконує всі функції;

– *Level* 1 (*Driver Assistance*) – автоматизована одна функція (наприклад, круїз-контроль);

– *Level* 2 (*Partial Automation*) – автоматичне керування швидкістю та рульовим керуванням (наприклад, *Tesla Autopilot, Nissan ProPILOT*);

– *Level* 3 (*Conditional Automation*) – транспортний засіб сам керує, але водій повинен бути готовий втрутитися (наприклад, *Audi A*8 *Traffic Jam Pilot*);

– *Level* 4 (*High Automation*) – повна автоматизація у визначених умовах, без потреби втручання водія;

– *Level* 5 (*Full Automation*) – повна автономія без керма та педалей.

З підвищенням рівня автоматизації автомобілів збільшується потреба в високошвидкісних і низькозатратних шинах передачі даних, здатних підтримувати реальний час та великий обсяг інформації. Це призводить до поступового витіснення шин типу *LIN* і *CAN* у критичних підсистемах та впровадження *Ethernet*-платформ нового покоління, особливо в електромобілях і безпілотних автомобілях. Успішне проєктування внутрішніх мереж напряму залежить від правильного розподілу навантаження між шинами залежно від призначення функцій і ступеня автоматизації.

Із зростанням рівня автоматизації:

– збільшується кількість сенсорів: *LiDAR*, *Radar*, 360° камери, ультразвукові сенсори;

– зростає частота зчитування: для формування точного образу середовища в реальному часі;

– збільшується обсяг міжмодульного трафіку: передача зображень, векторних карт, алгоритмів прийняття рішень між *ECU* та центральним комп'ютером.

Розглянемо кілька прикладів:

– *Tesla* використовує *Ethernet* та власну централізовану архітектуру, де основний комп'ютер обробляє потік від 8 камер, 12 ультразвукових сенсорів та радару. Це вимагає пропускної здатності на рівні гігабіт *Ethernet*;

– *Mercedes S-Class* використовує гібридну архітектуру з *FlexRay* (для шасі), *CAN* (для базових систем) і *Ethernet* (для *ADAS*);

– *BMW iX* вже впроваджує *TSN Ethernet* (*Time-Sensitive Networking*) – рішення, яке дозволяє гарантувати детерміновану доставку даних зі швидкістю 1 Гбіт/с.

Інженери обирають шину передачі даних не лише за швидкістю, але й з огляду на затримку, синхронізацію, вартість та кількість вже існуючих модулів, що підтримують ту чи іншу технологію.

1.3. Аналіз моделі передачі даних між системами транспортних засобів

У сучасних транспортних засобах функціонує складна розподілена система, що об'єднує десятки електронних блоків керування (*ECU*) через різні шини передачі даних. Ці шини реалізують обмін даними між підсистемами безпеки, комфорту, діагностики та автономного керування. Передача інформації повинна бути не лише швидкою, а й детермінованою, достовірною і адаптивною до умов реального часу.

1.3.1. Протоколи комунікації в реальному часі: вимоги до затримок, достовірності, синхронізації

У транспортних засобах все більшого значення набувають мережеві комунікації в режимі реального часу. Це зумовлено розвитком автоматизованих і автономних систем, які потребують швидкого та гарантовано достовірного обміну даними між

різними електронними блоками керування. Під терміном «реальний час» у контексті автомобільних мереж розуміється здатність системи забезпечити передавання критичних повідомлень із суворими часовими обмеженнями, що не допускають запізнення. Типовими прикладами є спрацювання системи стабілізації, блокування гальм чи реакція на зміну положення керма – усі ці дії повинні бути оброблені в межах кількох мілісекунд.

Для таких задач недостатньо лише високої пропускної здатності: важливо також гарантувати предиктивність поведінки мережі. Наприклад, у системах типу *ABS* або *ESP* затримка передавання сигналу понад 5 мс вже може поставити під загрозу безпеку. У цьому контексті критичним стає вибір мережевого протоколу. Протоколи на кшталт *FlexRay* або *Time-Sensitive Networking* (*TSN*) створені спеціально для таких сценаріїв. Вони не просто підтримують високу швидкість, а забезпечують сувору детермінованість у часі. *FlexRay* використовує статичні часові слоти для кожного пристрою, що дозволяє повністю уникнути конфліктів у мережі. *TSN*, в свою чергу, спирається на прецизійну синхронізацію між вузлами з точністю до мікросекунди, що є необхідним у сценаріях, де кілька сенсорів та виконавчих пристроїв мають працювати узгоджено.

Ще одним критичним фактором у забезпеченні якості комунікації є достовірність переданих даних. Відсутність або спотворення повідомлень, викликане перешкодами або атакою, може спричинити фатальні наслідки. Протоколи типу *CAN*, хоча й не мають вбудованого шифрування, містять механізми контролю цілісності кадру, наприклад, *CRC*-контроль, що дозволяє виявляти пошкоджені пакети. Крім того, логіка пріоритезації повідомлень у *CAN* базується на значенні ідентифікатора кадру – таким чином, системи критичного призначення мають вищий пріоритет і передаються першочергово навіть у разі перевантаження мережі.

Окремої уваги заслуговує питання синхронізації. В системах з множинними джерелами даних – камерами, радарами, *LiDAR*, блоками керування приводами – важливо, щоб усі пристрої використовували спільний часовий базис. У *FlexRay* реалізовано глобальну синхронізацію на рівні фізичного шару, тоді як в *Ethernet*-мережах із підтримкою *TSN* використовується механізм *IEEE* 802.1*AS* (*Precision Time*

Protocol), який дозволяє забезпечити точне вирівнювання по часу навіть між пристроями різного типу. Це дозволяє коректно співвідносити сенсорні дані, приймати виважені рішення на базі поточного стану і запобігати конфліктам у керуванні транспортним засобом.

1.3.2. Механізми маршрутизації та обробки повідомлень у багатокомпонентних транспортних мережах

У сучасних транспортних засобах обробка та маршрутизація повідомлень стала однією з ключових технічних задач через постійне зростання кількості сенсорів, електронних блоків керування та взаємопов'язаних підсистем. Від способу організації комунікації між компонентами залежить як ефективність передачі даних, так і стабільність функціонування критичних систем. Відтак, з огляду на ускладнення архітектури автомобілів, зростає значення як вибору топології, так і обраних підходів до маршрутизації даних.

На ранніх етапах розвитку електроніки в автотранспорті переважала доменна архітектура – тобто, окремі шини даних були виділені для кожної функціональної підсистеми: силової, шасі, інфотейнменту, клімату тощо. Наприклад, двигун міг взаємодіяти із системою уприскування через одну *CAN*-шину, тоді як мультимедійна система функціонувала незалежно на базі шини *MOST*. Такий підхід був зрозумілим, але в умовах нарощування функціоналу він призвів до надмірного ускладнення мережі – збільшувалась кількість роз'ємів, дротів, зростала маса й ризики збоїв.

Сучасні транспортні платформи дедалі активніше переходять на зональну архітектуру з централізованими обчисленнями. У ній замість десятків *ECU*, кожен із яких обробляє свою ділянку, використовуються зональні вузли – хаби, що відповідають за весь набір пристроїв у певній фізичній зоні (наприклад, лівий передній сектор авто). Вони підключаються до головного обчислювального блоку (*Vehicle Central Computer*), де відбувається логічна маршрутизація та обробка інформації. У таких системах можуть використовуватися *IP*-протоколи, що спрощує інтеграцію з *Ethernet*-сегментами та дає змогу реалізовувати гнучке перенаправлення повідомлень.

Важливим аспектом є відмінності між широкомовними та адресними підходами до передачі даних. Наприклад, у традиційному *CAN* всі повідомлення транслюються до всіх вузлів у мережі, які самостійно вирішують, чи є повідомлення релевантним. Такий спосіб зручний для невеликих систем, але він ускладнює масштабування та неефективно використовує пропускну здатність. У *FlexRay* використовується часовий поділ доступу – кожен пристрій отримує власний часовий слот, що виключає конфлікти. У мережах *Ethernet*, що підтримують *TCP*/*IP*, застосовується таблиця маршрутизації *MAC*/*IP*, яка дозволяє направляти повідомлення лише до потрібного вузла, економлячи пропускну здатність і час обробки.

Зі збільшенням обсягу переданих даних зростає потреба в об’єднанні окремих повідомлень. Агрегація – це процес, за якого кілька повідомлень, що мають однакове призначення або близькі за типом, збираються в один пакет. Наприклад, показники температури з кількох зон салону можуть бути об’єднані й передані одним кадром. Це дозволяє зменшити навантаження на мережу і підвищити ефективність використання каналу.

Окремої уваги заслуговують механізми буферизації та пріоритетного обслуговування повідомлень. У високошвидкісних мережах на кшталт *Ethernet* із підтримкою стандартів *TSN* (*Time-Sensitive Networking*) реалізовано політики якості обслуговування (*Quality of Service, QoS*), які дозволяють системам критичного призначення мати гарантований доступ до каналу . Це досягається через буфери з пріоритетами, які управляють чергами повідомлень, затримуючи менш важливі до моменту звільнення лінії для термінових сигналів.

Питання маршрутизації й обробки повідомлень у транспортних мережах – це вже не лише питання передачі бітів, а ціла система управління логікою зв’язку, що поєднує централізовані і децентралізовані елементи, сучасні протоколи і динамічні алгоритми формування потоку даних. Ці механізми визначають як поточну продуктивність системи, так і її здатність до масштабування, адаптації та інтеграції нових функцій.

1.3.3. Проблеми інтероперабельності між підсистемами з різними типами шин та небезпеки кібератак

Зі зростанням складності транспортних систем і впровадженням нових функціональних можливостей (автономне керування, зв’язок із хмарними сервісами, машинне навчання на борту) зростає потреба у гетерогенному середовищі передачі даних. Сучасні транспортні засоби дедалі частіше поєднують кілька шин різних типів – зокрема, *CAN*, *FlexRay* і *Automotive Ethernet* – кожна з яких має свою архітектуру, швидкість, механізми доступу до середовища та модель даних. Це ускладнює забезпечення узгодженості функціонування всієї системи [46; 66].

Однією з ключових проблем є сумісність форматів даних і логіки передачі між протоколами**.** Наприклад, *CAN*-кадри обмежені 8 байтами даних, тоді як *Ethernet* дозволяє передавати до 1500 байт у межах одного пакету. Для спільної роботи потрібні спеціальні пристрої – шлюзи, які виконують трансляцію між протоколами. Ці пристрої не просто копіюють інформацію, а перетворюють її структуру: додають заголовки, адаптують час, виконують обробку адресації й, у деяких випадках, фільтрацію або агрегацію повідомлень. Наприклад, трансляція *CAN*-повідомлення швидкості обертання коліс у *Ethernet*-мережу для системи автопілота потребує точного позначення часу та маршрутизації до відповідного модуля обробки.

Ще складнішим викликом є синхронізація даних у багатопротокольному середовищі. У той час як *Ethernet* із підтримкою *TSN* (*Time-Sensitive Networking*) та *FlexRay* можуть використовувати глобальну синхронізацію, класичний *CAN* не має вбудованої часової координати. Це створює проблеми при інтеграції даних із різних джерел. Наприклад, для коректної оцінки ситуації системою автономного керування необхідно узгоджено об’єднати дані з камер, *LiDAR*, *GPS* та *CAN*-сенсорів, що можливо лише у випадку точного таймстемпінгу. Саме тому шлюзи повинні не лише перетворювати дані, а й вбудовувати у них часову мітку, синхронізовану з глобальним годинником *ECU*.

Ще одна важлива проблема – втрата пріоритету та варіативність затримок при трансляції між мережами. У *CAN* повідомлення з нижчим *ID* мають вищий пріоритет, але при передачі в *Ethernet*-простір ці пріоритети втрачаються або замінюються

іншими правилами, що може призвести до ситуації, коли критично важливе повідомлення (наприклад, про активацію подушок безпеки) буде оброблено із затримкою. Більше того, внутрішні черги в шлюзах, затори в мережі *Ethernet* та фонові оновлення можуть викликати небезпечні затримки понад 10 мс, що перевищує допустимий поріг для систем реального часу.

Забезпечення інтероперабельності між мережами *CAN, FlexRay* та *Ethernet* є нетривіальним завданням, яке потребує створення інтелектуальних, адаптивних і синхронізованих шлюзів. Ці компоненти мають не лише транслювати дані, але й забезпечувати узгодженість часу, збереження пріоритетів і мінімізацію затримок. Лише за таких умов можлива ефективна інтеграція різнорідних систем в єдину функціональну архітектуру сучасного транспортного засобу.

Сучасні транспортні системи потребують складних моделей передачі даних, що враховують не лише швидкість і надійність комунікації, але й сумісність протоколів та здатність до синхронізації у реальному часі. зв’язку зі стрімким розвитком штучного інтелекту та технологій прогнозного обслуговування транспортних засобів, усе більш актуальним стає завдання інтеграції функцій аналізу технічного стану безпосередньо в шини передачі даних [1]. Зі зростанням кількості *ECU* та переходом до зональних архітектур виникає потреба в інтелектуальних шлюзах, адаптивному *QoS* та уніфікованих системах керування мережею, що забезпечують стабільність функціонування критичних систем незалежно від типу використовуваної шини. Розв’язання проблеми інтероперабельності є одним із ключових викликів на шляху до повноцінної автономії автомобіля.

Окремо необхідно розглянути небезпеку передачі даних між підсистемами, так останні дослідження атак [2] описують новий тип атак на *CAN*-шину – так звану транзитивну потайливу *Bus-Off* атаку, яка здатна вимикати вузли без прямого втручання в їхню поведінку, використовуючи особливості алгоритмів повторного входу в мережу.

Не менш важливими є питання уразливості шин передачі даних в транспортних засобах, тому що дослідження комплексних векторів атак на внутрішні мережі автомобіля показали, охоплюється і фізичний доступ до шин, часто проходить

зловживання діагностичними інтерфейсами та впровадження ін’єкцій шкідливих повідомлень у *CAN*-шину [18]. Результати засвідчили практичну здійсненність атак і необхідність глибшого захисту транспортних систем на рівні передачі даних.

А у роботі [19] автори звертають увагу на специфіку обробки помилок у внутрішніх мережах автомобілів, що відкривало можливості для нових векторів атак (реакції *ECU* на некоректні або неповні повідомлення *CAN* були використані зловмисниками для виведення систем з ладу та для скритої модифікації поведінки транспортного засобу).

З метою підвищення рівня інформаційної безпеки в системах, що базуються на шині *CAN*, в роботі [30] запропоновано інноваційний метод виявлення атак ін’єкції повідомлень на основі графів послідовностей кадрів. Автори моделюють закономірності змін повідомлень та виявляють аномальні ланцюжки шляхом аналізу схожості послідовностей, що підвищує точність виявлення стороннього втручання без модифікації структури самої шини.

Проведені дослідження наявних підходів до побудови систем комунікації в ТЗ дозволили припустити, що різні рівні автоматизації транспортних засобів (від допоміжних систем до повністю автономного керування) потребують різного підходу до вибору шин передачі даних і відповідних протоколів. Тоді можна відокремити комунікаційні вимоги для кожного типу задач (табл. 1.2).

Таблиця 1.2

Типи задач, які обробляють автоматизовані системи, і які комунікаційні вимоги вони пред’являють

Тип задачі	Приклади	Частота	Пріоритет	Надійність	Сумарні вимоги
Безпека (*safety*)	Гальмування, *ESP*, *ABS*	Висока	Максимум	Дуже висока	*RT*, з контролем помилок
Автономне керування	*Lane Keep*, *Path Planning*	Середня/висока	Високий	Висока	Низька затримка, синхронність
Комфорт/інфотейнмент	Навігація, музика, клімат	Середня	Середній/низький	Середня	Менше вимог до затримки

1.4. Постановка задачі дослідження

Зі зростанням рівня автоматизації функціональних систем керування – від допоміжних (*ADAS*) до повністю автономних – зростають і вимоги до характеристик шин передачі даних: продуктивності, часу затримки, безпеки, синхронізації та масштабованості. У таких умовах традиційні рішення, побудовані винятково на використанні *CAN*-шини або її модифікацій, виявляються недостатніми. Застосування гетерогенних шин (*CAN*, *LIN*, *FlexRay*, *Ethernet*) породжує нові виклики, пов’язані з уніфікацією обміну, управлінням пріоритетами повідомлень, виявленням аномалій та оцінкою достовірності переданих даних.

Як свідчать дослідження [20; 33; 55], високий рівень автоматизації неможливо реалізувати без централізації обробки та уніфікації форматів повідомлень між *ECUs*. Це призводить до переходу від класичної децентралізованої архітектури до зональної або сервіс-орієнтованої (*Service-Oriented Architecture, SOA*), де основну роль відіграє високопродуктивна шина з підтримкою динамічної маршрутизації, моніторингу аномалій і адаптивного управління трафіком. У зв’язку з цим сучасні підходи передбачають не лише підвищення пропускної здатності, а й використання інтелектуальних механізмів управління обміном повідомленнями в режимі реального часу.

Усе це вимагає побудови нового підходу до проєктування, оптимізації та безпечного функціонування шинної архітектури автоматизованих транспортних систем. Такий підхід має враховувати не лише технічні характеристики обміну, а й контекст виконання функцій (наприклад, режим руху, зовнішні умови), особливості поведінки систем і потенційні загрози безпеці, включаючи цілеспрямовані зовнішні впливи або внутрішні збої [47; 56; 58; 59].

У цьому контексті формулювання задачі дослідження базується на необхідності розробки методів, які дозволяють:

– враховувати контекстні та поведінкові фактори при оцінці достовірності повідомлень у транспортних шинах;

– забезпечити функціонування мультишинних конфігурацій з урахуванням часової узгодженості, пріоритетів, структури даних та шляхів маршрутизації;

– формувати єдину адаптивну модель обміну, здатну до навчання на основі спостережуваних статистичних патернів.

Досягнення мети вимагає розв’язання низки взаємопов’язаних наукових і прикладних завдань:

– виконати системний аналіз існуючих типів шин передачі даних, зокрема з позицій їх пропускної здатності, затримок, безпеки та міжшинової сумісності;

– побудувати модель функціонування шинної архітектури з урахуванням рівня автоматизації транспортного засобу та структури підсистем;

– розробити метод виявлення порушень у *CAN*-шинах із застосуванням байєсівського підходу до аналізу достовірності повідомлень;

– реалізувати експериментальне тестування запропонованих рішень у типових конфігураціях транспортних систем та провести оптимізацію з урахуванням критеріїв часу реакції, надійності, достовірності й захищеності;

– сформулювати рекомендації щодо вибору та конфігурації шин, з урахуванням типу транспортного засобу, рівня автоматизації та характеру функціональних задач.

Наукова цінність дослідження полягає в інтеграції методів системного аналізу, формалізації протоколів, байєсівської оцінки, кластеризації та адаптивної оптимізації для побудови нового підходу до організації інформаційного обміну в транспортних системах, здатного забезпечити високу ефективність, достовірність та безпеку передачі даних у динамічних умовах експлуатації.

1.5. Висновки до розділу 1

У першому розділі проведено ґрунтовний аналіз сучасного стану досліджень щодо використання шин передачі даних у транспортних засобах та їхньої ролі в електронній архітектурі автомобілів різного рівня автоматизації.

Розглянуто історичні передумови виникнення та розвитку автомобільних комунікаційних систем. Виявлено, що перехід від аналогових ліній до цифрових шин (*CAN*, *LIN*, *FlexRay*, *MOST*, *Automotive Ethernet*) суттєво розширив функціональні можливості транспортних засобів, забезпечивши надійний, швидкий та синхронізований обмін даними між великою кількістю електронних блоків керування

(*ECU*). Аналіз сучасних типів шин підтвердив, що кожна з них має свої переваги й недоліки, що визначаються сферою застосування та класом автомобіля. Зокрема, *CAN*-шина залишається найбільш поширеною завдяки низькій вартості, простоті та надійності, тоді як *Automotive Ethernet* і *FlexRay* активно впроваджуються в складні системи, які вимагають високої пропускної здатності та жорсткої синхронізації в реальному часі.

Проаналізовано архітектуру автоматизованих транспортних систем, у яких особлива увага приділяється системам безпеки, комфорту та автономного керування. Встановлено, що з підвищенням рівня автоматизації транспортних засобів зростають вимоги до шин передачі даних: збільшується необхідна пропускна здатність, зменшуються допустимі затримки та підвищується необхідність у гарантованій синхронізації. Це призводить до поступового переходу до мультишинних конфігурацій, що поєднують різнорідні комунікаційні технології.

Показано, що вибір шинної архітектури напряму впливає на ефективність функціонування автоматизованих систем автомобіля. Приклади сучасних реалізацій підтверджують тенденцію переходу до зональних або централізованих архітектур, які оптимально розподіляють обчислювальні навантаження та спрощують мережеву структуру транспортних засобів.

Досліджено особливості моделей передачі даних між системами транспортних засобів, включаючи вимоги до протоколів комунікації у реальному часі, механізми маршрутизації та обробки повідомлень, а також проблеми забезпечення інтероперабельності між шинними протоколами різних типів (*CAN*, *FlexRay*, *Ethernet*). Виявлено, що одним із найважливіших викликів є узгодження даних, синхронізація часових міток та збереження пріоритетів повідомлень під час трансляції між різними шинами. Відзначено необхідність створення інтелектуальних шлюзів та адаптивних механізмів маршрутизації, що здатні забезпечити високий рівень інтеграції та стабільності у багатокомпонентних мережах.

На підставі проведеного аналізу визначено ключові проблеми, що потребують вирішення у межах дослідження, а саме: забезпечення ефективної взаємодії гетерогенних шин, розробка адаптивних моделей обміну з урахуванням контекстних

і поведінкових факторів, а також реалізація методів виявлення порушень і аномалій з використанням сучасних статистичних і байєсівських підходів. Сформульовані завдання дослідження чітко окреслюють напрями подальшої роботи, яка має бути орієнтована на розробку науково обґрунтованих підходів до оптимізації архітектурної структури шин передачі даних для забезпечення високої ефективності, надійності та безпеки автоматизованих транспортних систем.

РОЗДІЛ 2
МАТЕМАТИЧНА ОСНОВА МЕТОДУ ВИЯВЛЕННЯ ЗАГРОЗ БЕЗПЕКИ В *CAN*-ШИНІ З ВИКОРИСТАННЯМ БАЄСІВСЬКОГО ПІДХОДУ

2.1. Формальна постановка задачі та базові визначення

Виявлення загроз безпеки в *CAN*-шині можна розділити на низку небезпек: власного походження (помилки в системі передачі даних, помилкові чи загрозливі дії самого водія) та зовнішнє втручання. У роботі [3] запропоновано уніфікований підхід до моніторингу транспортних засобів із застосуванням *IoT*-технологій, який дозволяє виявляти небезпечні сценарії в русі та своєчасно реагувати на критичні ситуації. Це особливо важливо в умовах інтелектуалізації транспортних систем і зростаючої ролі онлайнового аналізу телеметричних даних.

Розглянемо формальну математичну постановку задачі виявлення загроз безпеки в *CAN*-шині з використанням баєсівського підходу та процес комунікації в *CAN*-шині як послідовність команд та відповідей на них [62]. На основі цього визначимо основні елементи системи.

Якщо представимо процес комунікації в *CAN*-шині як послідовність транзакцій, кожна з яких представляє пару «команда-відповідь», тоді визначимо транзакцію T як впорядкований набір

$$T = (C, R, I, D, \tau), \tag{2.1}$$

де

C – команда, що ініціює зміну стану системи (наприклад, зміна швидкості руху чи положення керма);

R – відповідь системи (наприклад, крутний момент або гальмівний тиск);

I – ідентифікатор *CAN*-повідомлення (наприклад, $I = 0\times123$ для швидкості);

D – числові дані, що відображають значення параметрів;

τ – час реакції системи на отриману транзакцію, вимірюваний у мс.

Введемо фактори, які відображають динаміку руху, контекст та поведінку водія.

1. Динамічні характеристики руху:

– v – середня швидкість руху, визначена в діапазоні $v \in [0, 200]$ км/год, що характеризує стабільність поїздки;

– a – прискорення, перша похідна швидкості ($a \in \mathbb{R}$, м/с^2), що відображає динаміку розгону чи сповільнення;

– b – інтенсивність гальмування, бінарна змінна ($b \in \{0,1\}$), яка вказує на активність натискання гальм;

– σ_θ – різкість маневрів, обчислена як стандартне відхилення кута повороту керма θ, що характеризує варіативність траєкторії.

2. Контекстуальні фактори:

– H – час доби, категоріальна змінна ($H \in \{\text{день, ніч}\}$), що враховує підвищені ризики нічного водіння;

– χ – дорожній ухил ($\chi \in [-10\%, 10\%]$), який впливає на стабільність руху;

– ψ – щільність трафіку, коефіцієнт залежності швидкості від потоку $\psi \in [0,1]$, де $\psi = 0$ – вільний рух, $\psi = 1$ – затори.

3. Адаптивні поведінкові метрики:

– A_h – історія аномалій, накопичений бал відхилень від норми ($A_h \geq 0$), що відображає попередні загрози;

– Δ_s – відхилення від звичного стилю водіння, середньоквадратична різниця поточних характеристик від історичних патернів;

– α_g – агресивність водіння ($\alpha_g \in [0,1]$), узагальнений показник, що комбінує різкі прискорення, гальмування та високі кути повороту.

Ці параметри формують контекст

$$\Gamma = \{v, a, b, \sigma_\theta, H, \chi, \psi, A_h, \Delta_s, \alpha_g\}, \tag{2.2}$$

який враховується при аналізі транзакцій. Обмеження системи включають частоту передачі даних f, що залежить від реалізації *CAN*-шини (наприклад, $f = 500$ Гц), пропускну здатність B (наприклад, $B = 1000$ кбіт/с) та максимальний час реакції $\tau_{\max}$ (наприклад, 100 мс). Цільова точність моделі становить $A \geq 0.95$, що забезпечує надійність виявлення загроз у реальному часі.

Кожна транзакція T_i фіксується у певний момент часу, що дозволяє сформувати часовий ряд та проаналізувати типові послідовності або патерни поведінки.

Нехай послідовність транзакцій позначається як

$$\Xi = \left\{ \mathrm{T}_1 \to \mathrm{T}_2 \to \ldots \to \mathrm{T}_N, N \in \mathbb{Z} \right\},$$

де N – загальна кількість транзакцій, що надходять протягом деякого проміжку часу.

На основі цього ряду виявляються повторювані послідовності, які називаються патернами. Наприклад, одна конкретна послідовність транзакцій може регулярно зустрічатися під час нормальної роботи, що свідчить про стандартний режим експлуатації.

Для опису нормальної роботи мережі введемо поняття патерну. Тоді патерном типової поведінки системи позначимо впорядковану послідовність транзакцій

$$\Pi = \left\{ \mathrm{T}_1, \mathrm{T}_2, \ldots, \mathrm{T}_G, \; G \in \mathbb{Z} \right\},$$

де G – кількість транзакцій у патерні, що відображає типову послідовність дій у мережі.

Виявлення таких патернів дозволяє встановити базову модель роботи мережі, що є необхідною для подальшої класифікації та виявлення аномалій.

Стиль взаємодії системи з автомобілем позначимо, як множина характерних патернів

$$\Sigma = \left\{ \Pi_1, \Pi_2, \ldots, \Pi_M, \; M \in \mathbb{Z} \right\},$$

де M – кількість унікальних патернів поведінки.

Ця множина містить лише унікальні патерни, тобто ті послідовності, які відрізняються один від одного. Вона є квантом нормальної роботи мережі, оскільки кожен патерн Π_j відповідає певному типовому сценарію роботи системи.

Проведемо квантифікацію через частоти. Для кожного унікального патерну Π_j визначається його апріорна частота появи, що обчислюється як відношення кількості спостережень цього патерну до загальної кількості транзакцій j-го патерну

$$\nu_j = \frac{n_j}{N}, \tag{2.3}$$

де n_j – частота появи патерну $П_j$;

N – загальна кількість спостережених транзакцій.

Обрахунок частот дозволяє охарактеризувати нормальний режим роботи мережі. Надалі під нормальним режимом системи будемо розуміти, що система працює згідно з встановленими, стабільними і передбачуваними показниками, отриманими на основі історичних даних експлуатації. Тобто, такі параметри як час реакції системи τ, частота транзакцій, типові патерни повідомлень (які квантифікуються через значення ν_j) та інші ключові характеристики знаходяться в межах встановлених допусків і відповідають очікуванням, що визначені за нормальних умов роботи. Відхилення від апріорно встановлених показників можуть сигналізувати про аномалії або потенційні загрози в системі.

Для квантифікації роботи мережі ми спочатку аналізуємо великий обсяг даних (транзакцій), що генеруються системою під час її експлуатації. На основі цього аналізу виділяються повторювані, типові послідовності транзакцій – так звані патерни. Це дозволяє перетворити якісні спостереження (тобто, які види операцій відбуваються в мережі) у кількісні показники, які можна аналізувати математично.

2.2. Байєсівський підхід до оцінки безпечності

Байєсівський підхід до оцінки безпечності в автоматизованих транспортних системах забезпечує ефективний механізм обробки невизначеності та ймовірнісного аналізу ризиків у комунікаційних шинах. Завдяки можливості формалізувати зв’язки між численними параметрами повідомлень (час доставки, джерело, напрямок, пріоритет) цей підхід дозволяє здійснювати адаптивне оцінювання достовірності кожної транзакції. У працях [8; 12; 37; 38] висвітлено концептуальні та практичні основи байєсівського аналізу для систем реального часу, зокрема із застосуванням у кіберфізичних системах керування, де наявна висока динамічність даних.

Застосування байєсівських мереж як інструменту виявлення аномалій розглядається також у джерелах [12; 44; 69], де підкреслюється перевага цього підходу над класичними методами – насамперед у контексті високонавантажених

середовищ, таких як транспортні мережі. Його гнучкість дозволяє не лише виявляти відхилення у поведінці системи, а й адаптувати моделі безпечності відповідно до накопичених спостережень. У результаті формується механізм апостеріорного аналізу, що здатен підтримувати прийняття рішень у реальному часі [14].

Після формального визначення базових елементів мережі *CAN*, таких як транзакції, часові ряди та патерни поведінки, отримано детальну характеристику типового режиму роботи системи. Цей базовий рівень, що характеризується стабільністю основних параметрів (команда, відповідь, ідентифікатор, дані, час реакції) та типовими послідовностями дій, є основою для побудови апріорної статистичної моделі. Ці визначення описують не лише структуру та динаміку роботи системи, а й дозволяють побудувати апріорну статистичну модель, що є необхідним для подальшого аналізу.

Встановлення нормального стану системи дозволяє класифікувати, які саме характеристики є типовими для безпечного функціонування мережі. Апріорні частоти появи патернів забезпечують кількісне визначення нормального режиму роботи. Ці дані використовуються для того, щоб порівняти спостереження, отримані в реальному часі, із встановленими нормами. Це дозволяє виявляти потенційні відхилення та загрози.

Отримані дані слугують основою для застосування байєсівського підходу, що дозволяє інтегрувати апріорні знання з попередніх даних із поточними спостереженнями. Саме завдяки точній квантифікації нормальної роботи системи (за допомогою патернів та їхніх частот) ми можемо обчислити апостеріорну ймовірність того, що конкретна транзакція є безпечною.

Це забезпечує математично обґрунтований механізм інтеграції апріорних значень із поточними даними для точного прийняття рішень щодо класифікації транзакцій як безпечних або потенційно загрозливих.

Байєсівський підхід у даній роботі слугує основою для оцінки ймовірності безпечного стану системи *CAN*-шини (S) шляхом обчислення апостеріорної ймовірності на основі транзакцій та контекстуальних факторів. Цей метод базується на теоремі Баєса:

$$P(S \mid T,\Gamma) = \frac{P(T \mid S,\Gamma) \cdot P(S)}{P(T,\Gamma)} \tag{2.4}$$

де S – бінарний стан безпеки ($S=1$ – безпечний, $S=0$ – загроза), T – транзакція (див. (2.1)), Γ – контекст (див. (2.2)), $P(S)$ – апріорна ймовірність стану безпеки, а $P(T,\Gamma)$ – нормована константа, яка враховує всі можливі стани S і визначається за формулою $P(T,\Gamma) = \sum_{S\in\{0,1\}} P(T \mid S,\Gamma)P(S)$.

Апріорна ймовірність $P(S)$ задається на основі попередніх спостережень або експертних оцінок. Наприклад, $P(S=1)=0.98$ може відображати високу стабільність системи в типових умовах, тоді як $P(S=0)=0.02$ враховує рідкісні аномалії. Точні значення $P(S)$ адаптуються під час роботи моделі на основі емпіричних даних.

Структура правдоподібності

Правдоподібність $P(T \mid S,\Gamma)$ описує ймовірність спостереження транзакції T та контексту Γ за заданого стану S. Оскільки T включає числові дані D і час реакції τ, правдоподібність розкладається як

$$P(T \mid S,\Gamma) = P(D \mid S,\Gamma) \cdot P(\tau \mid S,\Gamma)$$

де $P(D \mid S,\Gamma)$ описує ймовірність даних D, а $P(\tau \mid S,\Gamma)$ – ймовірність часу реакції τ.

Правдоподібність даних $P(D \mid S,\Gamma)$ враховує динамічні характеристики руху, представлені в D (швидкість v, прискорення a, інтенсивність гальмування b, різкість маневрів σ_θ). Для спрощення моделі припускається умовна незалежність між цими параметрами за фіксованого S та Γ, що дозволяє записати:

$$P(D \mid S,\Gamma) = P(v \mid S,\Gamma) \cdot P(a \mid S,\Gamma) \cdot P(b \mid S,\Gamma) \cdot P(\sigma_\theta \mid S,\Gamma)$$

Кожен із цих розподілів моделюється залежно від стану. При $S=1$ параметри мають нормальні розподіли з центрами в очікуваних значеннях (наприклад, v близьке до типової швидкості, a – до нуля). При $S=0$ розподіли зміщуються до аномальних значень або стають рівномірними, відображаючи нестабільність.

Це припущення спрощує обчислення, але його обґрунтованість залежить від слабкої кореляції між параметрами в типовому стані, що потребує емпіричної перевірки.

Правдоподібність часу реакції $P(\tau \mid S, \Gamma)$. Час реакції τ у безпечному стані (S=1) підкоряється нормальному розподілу $\tau \sim N(\mu_\tau, \sigma_\tau^2)$ з очікуваним середнім значенням, тоді як у стані загрози (S=0) середнє зміщується в бік більших затримок, що відображає потенційні збої в системі.

Роль контексту

Контекст Γ впливає на правдоподібність через модифікацію базового розподілу:

$$P(T \mid S, \Gamma) = P(T \mid S) \cdot e^{\beta \cdot f(\Gamma)} \tag{2.5}$$

де $f(\Gamma)$ – лінійна комбінація контекстуальних параметрів із відповідними вагами:

$$f(\Gamma) = \sum_i w_i \cdot \delta_i$$

де δ_i – окремі компоненти Γ (наприклад, v, a, H, A_h тощо), w_i – ваги, що визначають їхній внесок, β – коефіцієнт масштабування. Ця форма дозволяє гнучко враховувати вплив часу доби, дорожнього ухилу, щільності трафіку та поведінкових метрик.

Поведінкові метрики

Адаптивні метрики (A_h, Δ_s, α_g) інтегруються в модель через:

– A_h (історія аномалій) – зменшує $P(S=1)$ експоненціально зі зростанням накопичених відхилень;

– Δ_s (відхилення стилю) – при $S=1$ має нормальний розподіл із центром у нулі, при $S=0$ – зміщується до більших значень;

– α_g (агресивність) – при $S=1$ моделюється бета-розподілом із помірними значеннями ($\alpha_g \sim B(2,2)$), при $S=0$ тяжіє до екстремальних величин.

Загальний алгоритм

Алгоритм класифікації працює наступним чином:

1. Ініціалізація $P(S)$ та параметрів розподілів для $P(\delta \mid S,\Gamma)$ і $P(\tau \mid S,\Gamma)$.
2. Для кожної транзакції T обчислення $P(T \mid S,\Gamma)$ з урахуванням Γ.
3. Оновлення $P(T \mid S,\Gamma)$ за формулою (2.5).
4. Порівняння $P(S=0 \mid T,\Gamma)$ із порогом для виявлення загрози.

Цей підхід забезпечує адаптивність до змін умов руху та поведінки водія, що робить його ефективним для аналізу безпеки в реальному часі.

2.3. Математична модель транзакцій та адаптація параметрів

Математична модель розроблена для класифікації транзакцій $T=(C,R,I,D,\tau)$ та виявлення аномалій у поведінці системи *CAN*-шини шляхом групування їх у патерни Π_j, що відображають типові режими роботи автомобіля. Модель поєднує байєсівський підхід (див. підрозділ 2.2) із адаптивним аналізом, враховуючи динамічні характеристики руху, контекстуальні фактори та поведінкові метрики, об’єднані в контекст Γ.

Визначення патернів

Кожен патерн Π_j є набором транзакцій, що характеризуються подібними значеннями параметрів у D (наприклад, швидкістю v, прискоренням a, інтенсивністю гальмування b, різкістю маневрів σ_θ) та умовами Γ. Прикладами патернів можуть бути спокійне водіння, агресивні маневри або рух у складних умовах. Частота появи патерну ν_j визначає його ймовірність у потоці транзакцій (див. формулу (2.3)) і адаптивно оновлюється за правилом:

$$\nu_j(t)=\alpha \cdot I_j(t)+(1-\alpha)\cdot \nu_j(t-1), \qquad (2.5)$$

де $I_j(t)=1$, якщо транзакція $T(t)$ належить до Π_j, і $I_j(t)=0$ у протилежному випадку; $\alpha \in (0,1)$ – коефіцієнт згладжування, який контролює швидкість адаптації (наприклад, $\alpha=0.2$ для помірного оновлення). Початкове значення $\nu_j(0)$ задається рівномірно між усіма патернами або на основі апріорних припущень.

Належність транзакції T до патерну Π_j визначається через порівняння значень D із діапазонами, характерними для Π_j, та врахування контексту Γ. Наприклад, для патерну спокійного водіння v перебуває в межах типового діапазону, a близьке до нуля, $b = 0$, а σ_θ має низьке значення. Контекстуальні фактори, такі як H (час доби) або ψ (щільність трафіку), можуть модифікувати ці умови, підвищуючи ймовірність аномальних патернів у складних ситуаціях.

Інтеграція поведінкових метрик

Адаптивні поведінкові метрики відіграють ключову роль у формуванні патернів і виявленні відхилень:

1. Агресивність водіння α_g обчислюється як зважена комбінація динамічних характеристик:

$$\alpha_g = w_a \cdot |a| + w_b \cdot b + w_\theta \cdot \sigma_\theta$$

де w_a, w_b, w_θ – ваги, що відображають внесок прискорення, гальмування та маневрів (наприклад, $w_a = 0.4,\ w_b = 0.3,\ w_\theta = 0.3$). Значення $\alpha_g \in [0,1]$ нормалізується, а високі значення вказують на агресивний стиль, що може бути окремим патерном або сигналом аномалії.

2. **Відхилення від звичного стилю** Δ_s розраховується як середньоквадратична різниця поточних параметрів D від історичних середніх:

$$\Delta_s = \sqrt{\frac{1}{N}\sum_i (D_i - \mu_{D_i})^2}$$

де D_i – окремі компоненти D (наприклад, $v,\ a$), μ_{D_i} – їхні середні значення за попередніми транзакціями, N – кількість компонентів. Велике значення Δ_s свідчить про відхилення від норми, що враховується при класифікації.

3. **Історія аномалій** A_h зростає з кожною виявленою аномалією і впливає на ймовірність патернів через зміщення $P(S \mid T, \Gamma)$ за правилом:

$$A_h^{(t+1)} = A_h^{(t)} + \Delta A_h I(P(S = 0 \mid T, \Gamma) > \eta),$$

де

ΔA_h – приріст;

$I(\cdot)$ – індикаторна функція, що приймає значення 1, якщо умова всередині дужок істинна, і 0, якщо хибна.

Виявлення аномалій

Модель ідентифікує аномалії, порівнюючи апостеріорну ймовірність $P(S=0|T,\Gamma)$ (див. формулу (2.4)) із заданим порогом η (наприклад, $\eta=0.1$). Якщо $P(S=0|T,\Gamma)>\eta$, транзакція класифікується як аномальна. Частоти v_j використовуються для уточнення цього рішення – низька v_j для патерну, до якого належить T, додатково сигналізує про відхилення від типової поведінки.

Алгоритм моделі

1. Ініціалізація множини патернів Π_j та їхніх частот v_j.
2. Для кожної транзакції T:
   - Обчислення $P(S|T,\Gamma)$ за баєсівським підходом (див. формулу (2.4)).
   - Визначення належності до Π_j на основі D і Γ.
   - Оновлення v_j за формулою (2.6).
   - Розрахунок α_g і Δ_s для уточнення класифікації.
3. Порівняння $P(S=0|T,\Gamma)$ із порогом для виявлення аномалій.

Роль контексту

Контекст Γ впливає на формування патернів і оцінку аномалій. Наприклад, $H=\{ніч\}$ або χ (дорожній ухил) з великим значенням можуть змістити модель до патернів із вищим ризиком, тоді як ψ (щільність трафіку) корегує очікувані значення v і a. Це забезпечує адаптивність моделі до зовнішніх умов.

Математична модель поєднує статистичний аналіз із динамічним оновленням, що дозволяє ефективно класифікувати транзакції та виявляти загрози в реальному часі, враховуючи як поточні параметри, так і історичний контекст.

2.4. Оптимізаційна задача та алгоритми навчання

Оптимізація математичної моделі, представленої в підрозділі 2.3, спрямована на забезпечення високої точності та повноти виявлення загроз безпеки *CAN*-шини в умовах змінних динамічних характеристик руху, контекстуальних факторів і поведінкових метрик, об'єднаних у контекст Г (див. підрозділ 2.1). Цей процес включає налаштування параметрів моделі для максимізації цільової функції, яка враховує компроміс між чутливістю до аномалій і стійкістю до помилкових спрацьовувань, а також оцінку ефективності через метрики, адаптовані до специфіки реального часу.

Цільова функція

Кожну класифікаційну задачу можна охарактеризувати матрицею помилок, що містить чотири основні показники:

– *TP* (*True Positives*) – кількість правильно класифікованих позитивних випадків (безпечні транзакції, які система правильно ідентифікувала як безпечні);

– *TN* (*True Negatives*) – кількість правильно класифікованих негативних випадків (загрозливі транзакції, які система правильно відзначила як загрозливі);

– *FP* (*False Positives*) – кількість хибно позитивних випадків (загрозливі транзакції, помилково класифіковані як безпечні);

– *FN* (*False Negatives*) – кількість хибно негативних випадків (безпечні транзакції, помилково класифіковані як загрозливі).

На основі матриці помилок визначаються наступні стандартні метрики:

1. Точність класифікації (*Accuracy*):

$$Q_1 = \frac{TP + TN}{TP + TN + FP + FN}.$$

Ця метрика відображає загальну частку правильно класифікованих транзакцій.

2. Повнота класифікації (*Recall*):

$$Q_2 = \frac{TP}{TP + FN}.$$

Метрика відображає частку правильно виявлених аномалій ($S = 0$) серед усіх реальних загроз.

3. Точність позитивних рішень (*Precision*)

$$Q_3 = \frac{TP}{TP + FP}.$$

Метрика характеризує частку правильних позитивних передбачень серед усіх класифікованих як аномалії.

4. $F1$-міра

$$F = \frac{2(Q_2 \times Q_3)}{Q_2 + Q_3} = \frac{2 \cdot TP}{2 \cdot TP + FP + FN}.$$

$F1$-міра є гармонійним середнім між точністю позитивних рішень та повнотою класифікації, що дає збалансовану оцінку класифікації.

Ефективність моделі оцінюється через цільову функцію $Q(\theta)$, яка формалізована як зважена комбінація повноти (Q_2) і точності (Q_3) із регуляризаційним членом для запобігання перенавчанню:

$$Q(\theta) = w_1 \cdot Q_2 + w_2 \cdot Q_3 - \lambda \| \theta \|_2^2, \qquad (2.7)$$

де

θ – вектор параметрів моделі, який включає середні та дисперсії розподілів $P(D|S,\Gamma)$ і $P(\tau|S,\Gamma)$ (див. підрозділ 2.2), а також ваги w_i у $f(\Gamma)$;

w_1 і w_2 – коефіцієнти, що визначають пріоритет між Q_2 і Q_3 (наприклад, $w_1 = 0.7,\ w_2 = 0.3$ для акценту на виявленні загроз);

λ – коефіцієнт регуляризації (наприклад, $\lambda = 0.01$), що контролює складність моделі;

$\| \theta \|_2^2$ – L_2 – норма параметрів, яка сприяє узагальненню.

Такий вибір $Q(\theta)$ відображає необхідність балансу між чутливістю до аномалій (високе Q_2) і мінімізацією хибнопозитивних спрацьовувань (високе Q_3), що є критичним для систем реального часу, де помилки можуть призвести до надмірного навантаження або ігнорування реальних загроз.

Для максимізації $Q(\theta)$ застосовується алгоритм адаптивної оптимізації *Adam* (*Adaptive Moment Estimation*), який поєднує переваги градієнтного спуску з

моментними методами для ефективного навчання в умовах нелінійних залежностей між $T = (C, R, I, D, \tau)$ і Γ. Оновлення параметрів задається як:

$$\theta_{t+1} = \theta_t - \xi \cdot \frac{\hat{m}_t}{\sqrt{d_t} + \varepsilon},$$

де

ξ – крок навчання (*learning rate*), який регулює швидкість оновлення;

$\hat{m}_t = \beta_1 \cdot \hat{m}_{t-1} + (1 - \beta_1) \cdot \nabla Q(\theta_t)$ – скоригований перший момент (середнє значення градієнта);

$d_t = \beta_2 \cdot d_{t-1} + (1 - \beta_2) \cdot (\nabla Q(\theta_t))^2$ – скоригований другий момент (дисперсія градієнта);

$\beta_1, \beta_2 \in [0,1)$ – коефіцієнти згладжування (типово обираються значення $\beta_1 = 0.9,\ \beta_2 = 0.999$);

ε – мала константа (наприклад, 10^{-8}) для числової стабільності;

$\nabla Q(\theta_t)$ – градієнт цільової функції за θ на ітерації t.

Градієнт $\nabla Q(\theta)$ обчислюється через часткові похідні $\frac{\partial Q}{\partial \theta}$, враховуючи вплив $P(S \mid T, \Gamma)$ та частот патернів ν_j. Алгоритм *Adam* забезпечує швидку збіжність навіть у присутності шумів у D (наприклад, через варіативність v чи σ_θ), що робить його придатним для обробки потоків транзакцій *CAN*-шини.

Контекст Γ відіграє ключову роль у процесі оптимізації, оскільки параметри θ повинні адаптуватися до змін умов руху та поведінки водія. Наприклад, високі значення α_g (агресивність водіння) або A_h (історія аномалій) можуть вимагати коригування порогів у $P(D \mid S, \Gamma)$, тоді як ψ (щільність трафіку) впливає на очікувані значення v і a. Для цього θ оновлюється ітеративно на основі батчів транзакцій (наприклад, 100 транзакцій за раз), дозволяючи моделі динамічно реагувати на контекстуальні зміни. Початкові значення θ ініціалізуються на основі ретроспективних даних або випадково в межах $[-0.1, 0.1]$.

Ефективність моделі може бути оцінена через значення $Q(\theta)$ на відкладеній множині транзакцій, а також через додаткові метрики, такі як $F1$-оцінка, яка збалансовано поєднує Q_2 і Q_3. Для аналізу стійкості до змін у Γ проводиться крос-валідація за сценаріями (наприклад, різні значення H чи χ), що дозволяє оцінити узагальнюючу здатність моделі. Часова складність однієї ітерації оптимізації визначається $O(N_b \cdot M \cdot L)$, де N_b – розмір батчу, M – кількість патернів Π_j, L – розмірність Γ (тут $L=10$), що забезпечує масштабованість для потоків даних у реальному часі.

2.5. Теоретичне обґрунтування

Теоретичне обґрунтування запропонованої моделі виявлення загроз безпеки *CAN*-шини спирається на аналіз її збіжності, обчислювальної складності та стійкості до змін у транзакціях $T=(C,R,I,D,\tau)$ і контексті Γ. Модель поєднує байєсівський підхід (підрозділ 2.2), адаптивну класифікацію патернів (підрозділ 2.3) і оптимізацію параметрів (підрозділ 2.4), що вимагає формального доказу її коректності та ефективності в умовах реального часу. У цьому підрозділі розглядаються ключові аспекти, які підтверджують теоретичну обґрунтованість алгоритму.

Збіжність алгоритму

Збіжність моделі до оптимального рішення визначається двома ключовими компонентами: байєсівською оцінкою $P(S|T,\Gamma)$ і оптимізацією $Q(\theta)$ через алгоритм *Adam* (див. підрозділ 2.4).

Для баєсівського підходу збіжність гарантується за умови стаціонарності розподілів $P(D|S,\Gamma)$ і $P(\tau|S,\Gamma)$, що припускає обмежену варіативність у D і τ у межах типових патернів Π_j (підрозділ 2.3). У цьому випадку апостеріорна ймовірність $P(S|T,\Gamma)$ стабілізується зі збільшенням кількості транзакцій, що обробляються, із швидкістю $O(\frac{1}{\sqrt{N}})$, де N – кількість оброблених транзакцій, що випливає із закону великих чисел. Це означає, що з накопиченням даних модель дедалі точніше відображає базові режими роботи системи.

Оптимізація $Q(\theta)$ через *Adam* має швидкість збіжності $O\left(\frac{1}{\varepsilon}\right)$, де ε – цільова похибка (наприклад, різниця між поточним і оптимальним значенням цільової функції $Q(\theta)$ $(Q(\theta^*) - Q(\theta) < \varepsilon)$). Ця оцінка базується на припущенні ліпшицевої неперервності градієнтів $\nabla Q(\theta)$ і належного вибору кроку навчання η (наприклад, $\eta_t = \eta_0/(1+t)^{0.5}$, де $\eta_0 = 0.001$. Вплив адаптивних метрик, таких як A_h (історія аномалій) і Δ_s (відхилення стилю), ускладнює простір параметрів θ, але регуляризація $\lambda \| \theta \|_2^2$ (з $\lambda = 0.01$) забезпечує стійкість до локальних мінімумів, узгоджується з теоремами стохастичної оптимізації для нелінійних систем. У результаті модель досягає принаймні субоптимального рішення θ^*, що забезпечує цільову точність $A \geq 0.95$.

Обчислювальна складність

Обчислювальна складність моделі визначається як:

$$O(N_b \cdot M \cdot L)$$

де

N_b – кількість транзакцій у потоці;

M – кількість патернів Π_j, що розглядаються для класифікації;

L – розмірність контексту Γ, що включає компоненти $v, a, b, \sigma_\theta, H, \chi, \psi, A_h, \Delta_s, \alpha_g$.

Для кожної транзакції T необхідно:

1. Обчислити $P(S|T,\Gamma)$ з обчислювальною складністю $O(L)$, враховуючи множення розподілів $P(D|S,\Gamma)$ і $P(\tau|S,\Gamma)$ (підрозділ 2.2).
2. Оновити частоти ν_j для M патернів із складністю $O(M)$ (підрозділ 2.3).
3. Виконати крок оптимізації θ через *Adam* із складністю $O(L)$, де розмірність θ пропорційна L.

У реальному часі N_b може бути обмеженим вікном обробки (наприклад, батчем транзакцій), що робить загальну складність лінійною за N_b, M і L. Оптимізація

Adam додатково масштабується завдяки ефективним структурам даних (наприклад, хеш-таблиці для ν_j) і векторизації обчислень, що дозволяє моделі відповідати частоті передачі *CAN*-шини, забезпечуючи обробку в межах $\tau_{max}=100$ мс.

Стійкість до варіацій

Стійкість моделі до змін у D і Γ забезпечується адаптивними механізмами:

– динамічні характеристики (v, a, b, σ_θ). Нормальні розподіли в $P(D|S,\Gamma)$ дозволяють поглинати малі флуктуації (наприклад, $|a|<0.5$ м/с²), тоді як аномальні відхилення (наприклад, $|a|>3$ м/с²) підвищують $P(S=0|T,\Gamma)$;

– контекстуальні фактори (H, χ, ψ). Їхній вплив через $f(\Gamma)$ (підрозділ 2.2) забезпечує гнучке коригування ймовірностей без перебудови моделі. Наприклад, $H=\{\text{ніч}\}$ збільшує ваги аномальних патернів;

– поведінкові метрики (A_h, Δ_s, α_g). Адаптивне оновлення ν_j і α_g (підрозділ 2.3) відображає еволюцію поведінки водія, а A_h накопичує довгострокові тренди, підвищуючи чутливість до систематичних аномалій.

Теоретично це підтверджується ергодичністю процесу транзакцій. За достатньої кількості транзакцій модель стабільно розрізняє типові та аномальні режими, що узгоджується з принципами марковських процесів в аналізі часових рядів. Наприклад, при стабільному $\bar{v}\approx 60$ км/год і $\psi<0.5$ модель зберігає низьке $P(S=0|T,\Gamma)$, тоді як різке зростання α_g (наприклад, до 0.9) сигналізує про аномалію

Граничні умови

Ефективність моделі залежить від якості ініціалізації $P(S)$ і θ. У разі нестабільності Γ (наприклад, різка зміна ψ з 0.2 до 0.8) можливе тимчасове зниження $F1$-міри (підрозділ 2.4), що компенсується адаптивним η в *Adam*. Теорема про слабку збіжність стохастичних градієнтів гарантує, що модель досягає принаймні субоптимального рішення θ^* за розумний час (наприклад, $t<10^4$ ітерацій для $N_b=100$). Крім того, обмежена кількість патернів ($M<50$) і батчів ($N_b<200$) запобігає обчислювальним перевантаженням у реальному часі.

2.6. Висновки до розділу 2

На основі розробленої формальної постановка задачі, що заклала основу для моделювання комунікаційних процесів у *CAN*-шині як послідовностей транзакцій, враховано динамічні характеристики руху, контекстуальні фактори та адаптивні поведінкові метрики. Введення концепції патернів типової поведінки системи дозволило квантифікувати нормальний режим роботи через апріорні частоти, що є ключовим етапом для подальшого виявлення можливих аномалій.

Математичне обґрунтування обраного Байєсівського підходу забезпечив механізм оцінки апостеріорної ймовірності безпечного стану системи, інтегруючи апріорні знання з поточними спостереженнями. Застосування теореми Баєса дозволяє враховувати як числові параметри транзакцій, так і контекстуальні впливи, що модифікують правдоподібність через адаптивні розподіли. Модель транзакцій розширює цей підхід шляхом групування транзакцій у патерни та динамічного оновлення їхніх частот, що підвищує чутливість до відхилень від норми, зберігаючи при цьому обчислювальну ефективність.

Оптимізаційна задача формалізує процес налаштування параметрів моделі через цільову функцію, яка балансує між повнотою та точністю класифікації, а алгоритм *Adam* забезпечує швидку збіжність навіть у присутності нелінійних залежностей і шумів у даних. Теоретичне обґрунтування (підрозділ 2.5) підтверджує збіжність моделі до субоптимального рішення з обчислювальною складністю, що відповідає вимогам реального часу, а також її стійкість до варіацій у транзакціях і контексті завдяки адаптивним механізмам і регуляризації.

Запропонована модель демонструє високий рівень узагальнення, що дозволяє ефективно виявляти загрози безпеки в *CAN*-шині в умовах змінних зовнішніх факторів і поведінки водія. Її теоретична обґрунтованість спирається на принципи стохастичної оптимізації, ергодичності процесів і статистичного виведення, що робить її перспективною для практичного впровадження в автомобільних системах. Подальші дослідження можуть бути спрямовані на емпіричну валідацію припущень про умовну незалежність параметрів і оптимізацію обчислювальних ресурсів для забезпечення масштабованості в багатокомпонентних мережах.

РОЗДІЛ 3
АНАЛІЗ ПОВЕДІНКОВИХ ШАБЛОНІВ ВОДІННЯ ЗА ДОПОМОГОЮ МОДИФІКОВАНОЇ ВІДСТАНІ ЛЕВЕНШТЕЙНА З ПОПЕРЕДНІМ СТАТИСТИЧНИМ АНАЛІЗОМ

3.1. Концепція прискореного порівняння послідовностей

Для аналізу поведінкових шаблонів розглядались різні підходи – так у дослідженні [4] представлено підхід до динамічного формування контенту за рахунок аналізу поведінкових характеристик користувача, де подібна стратегія мала потенціал, і в контексті побудови поведінкових моделей для виявлення аномалій у транспортних інформаційних системах, де відхилення від типових дій можуть свідчити про порушення або вторгнення, в роботах [23; 54] для визначення типу поведінки водія використовувалось машинне навчання, а в роботі [24] запропоновано підхід до класифікації типових маневрів (гальмування, поворот, прискорення) на основі часових рядів, отриманих з транспортних засобів. Автори поєднують правила на основі доменної експертизи з алгоритмами машинного навчання, досягаючи високої точності класифікації. В роботах [49; 60; 61; 62] представлено підхід до кластеризації, який може бути адаптований до задач виявлення нетипових режимів функціонування різноманітних систем.

Запропонований метод базується на комбінації попереднього статистичного аналізу та обчислення модифікованої відстані Левенштейна для порівняння часових рядів параметрів водіння. Основна ідея полягає у використанні обчислювально ефективного статистичного аналізу як першого етапу фільтрації, після якого лише для потенційно схожих послідовностей виконується більш ресурсоємне обчислення відстані Левенштейна [57].

Даний підхід детально розглядався в дослідженні [69] для для категоризації користувачів інформаційних систем та досліджень [6; 17] для оцінки поведінки користувачів і учнів.

Математичне формулювання

Нехай маємо дві послідовності спостережень параметрів водіння

$A = \{a_1, a_2, \ldots, a_m\}$ – еталонна послідовність

$B = \{b_1, b_2, \ldots, b_n\}$ – нова послідовність для порівняння

Для кожного параметра (швидкість, оберти двигуна, положення акселератора, кут керма) формуємо окремі послідовності значень, взяті через рівні проміжки часу (наприклад, щосекунди).

Для кожної послідовності обчислюємо ключові статистичні характеристики:

1. $\mu_A = \frac{1}{m}\sum_{i=1}^{m} a_i$, $\mu_B = \frac{1}{n}\sum_{i=1}^{n} b_i$ – середні значення характеристик;

2. $\sigma_A = \sqrt{\frac{1}{m}\sum_{i=1}^{m}(a_i - \mu_A)^2}$,

$\sigma_B = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(b_i - \mu_B)^2}$ – середньоквадратичне відхилення.

Для порівняння статистичних характеристик обчислюємо відносні різниці

$$\Delta_\mu = \frac{|\mu_A - \mu_B|}{\max(\mu_A, \mu_B)},\ \Delta_\sigma = \frac{|\sigma_A - \sigma_B|}{\max(\sigma_A, \sigma_B)}$$

Якщо $\Delta_\mu > \theta_\mu$ або $\Delta_\sigma > \theta_\sigma$, де θ_μ і θ_σ – порогові значення (наприклад, 0.2 або 20%), то послідовності вважаються статистично відмінними.

Для послідовностей, які не були відфільтровані на етапі статистичного аналізу, обчислюємо модифіковану відстань Левенштейна:

1. **Нормалізація послідовностей**

$$\hat{a}_i = \frac{a_i - \min(A,B)}{\max(A,B) - \min(A,B)},\ \hat{b}_i = \frac{b_i - \min(A,B)}{\max(A,B) - \min(A,B)},$$

де $\min(A,B)$ і $\max(A,B)$ – мінімальне і максимальне значення серед усіх елементів обох послідовностей.

2. **Обчислення відстані Левенштейна**

Ініціалізуємо матрицю відстаней D розміром $(m+1)\times(n+1)$

$$\begin{cases} D[0,0] = 0 \\ D[i,0] = i, \quad \forall i \in 1,2,\ldots,m \\ D[0,j] = j, \quad \forall j \in 1,2,\ldots,n \end{cases}$$

Заповнюємо решту матриці за формулою

$$D[i,j] = \min\begin{cases} D[i-1,j]+1 \\ D[i,j-1]+1 \\ D[i-1,j-1]+c(\hat{a}_i,\hat{b}_j) \end{cases}$$

де $c(\hat{a}_i,\hat{b}_j) = |\hat{a}_i - \hat{b}_j|$ – вартість операції заміни, яка залежить від різниці між нормалізованими значеннями.

3. **Нормалізація відстані**

$$d_L(A,B) = \frac{D[m,n]}{\max(m,n)}$$

Для оцінки схожості послідовностей за кількома параметрами обчислюємо зважену відстань

$$D_{total} = \sum_{p \in P} w_p \cdot d_L(A_p, B_p)$$

де

P – множина параметрів (швидкість, оберти двигуна, акселератор, кут керма),

w_p – вага параметра p, $\sum_{p \in P} w_p = 1$,

$d_L(A_p, B_p)$ – нормалізована відстань Левенштейна для параметра p.

3.2. Детальний приклад обрахунків

Для перевірки запропонованого методу проведено логування даних, що передавались *CAN*-шиною на різних ділянках дорого чотирма різними водіями. Частина даних представлена на рис. 3.1.

can_test_data.csv

```
timestamp,can_id,reaction_time,driver,trip_id,car_speed,engine_speed,current_gear,brake_switch,longitude_acceleration,latitude_a
2025-03-30 22:39:28.438,896,7.627679724710886,TestDriver,TestTrip,0.324043938017135,788.7071140004359,1,0,0.0,0.0,-0.96344488532
2025-03-30 22:39:28.448,896,12.606094449790056,TestDriver,TestTrip,0.6207619704991354,717.35957454683,1,0,29.67180324820004,0.0,
2025-03-30 22:39:28.458,896,10.042478715230597,TestDriver,TestTrip,0.9152105840710962,1028.6682182585487,1,0,29.444861357196082,
2025-03-30 22:39:28.468,288,11.153914108083296,TestDriver,TestTrip,1.5833207808921634,1077.1298439048917,1,0,66.81101968210672,0.
2025-03-30 22:39:28.478,272,7.119036092042987,TestDriver,TestTrip,2.0980993350148385,1215.907298672977,1,0,51.4778554122675,0.0,
2025-03-30 22:39:28.488,896,11.330975073995088,TestDriver,TestTrip,2.299985326709972,1364.8769529234735,1,0,20.188599169513342,0.
2025-03-30 22:39:28.498,272,12.050262744199962,TestDriver,TestTrip,2.914794325281334,1319.5315643652589,1,0,61.48089985713621,0.
2025-03-30 22:39:28.508,896,5.622462080811381,TestDriver,TestTrip,3.5949884281789872,1591.6956073225724,1,0,68.01941028976532,0.
2025-03-30 22:39:28.518,640,8.697736987800559,TestDriver,TestTrip,3.8101964040719114,1605.540669939632,1,0,21.520797589292417,0.
2025-03-30 22:39:28.528,272,5.48837841555283,TestDriver,TestTrip,4.531744735135634,1561.4343330545066,1,0,72.15483310637225,0.0,
2025-03-30 22:39:28.538,896,8.798392723684723,TestDriver,TestTrip,4.976152356960438,1667.7484670763831,1,0,44.440762182480405,0.
2025-03-30 22:39:28.548,640,12.87658263695235,TestDriver,TestTrip,5.0,1779.3058824128111,1,0,2.3847643039561994,0.0,-0.174625150
2025-03-30 22:39:28.558,896,11.154525128931446,TestDriver,TestTrip,5.0,1698.9965309959043,1,0,0.0,0.0,0.4728662077082646,-3.3349
2025-03-30 22:39:28.568,288,10.911055935197945,TestDriver,TestTrip,5.0,1647.6488002441083,1,0,0.0,0.0,-1.473599494314673,-3.3349
2025-03-30 22:39:28.578,640,9.878518734503475,TestDriver,TestTrip,5.0,1642.5421619366869,1,0,0.0,0.0,-1.4729272046948816,-3.3349
2025-03-30 22:39:28.588,896,10.783295117686938,TestDriver,TestTrip,5.0,1603.0036507354241,1,0,0.0,0.0,-3.094733969254055,-3.3349
2025-03-30 22:39:28.598,640,11.270218038766501,TestDriver,TestTrip,5.0,1689.8560064860276,1,0,0.0,0.0,-1.5360462768669483,-3.334
2025-03-30 22:39:28.608,896,13.457505565428887,TestDriver,TestTrip,5.0,1716.8669555307856,1,0,0.0,0.0,1.3051480569460714,-3.3349
2025-03-30 22:39:28.618,288,10.473322028902864,TestDriver,TestTrip,5.0,1609.6378054657234,1,0,0.0,0.0,-1.5593916060681257,-3.334
2025-03-30 22:39:28.628,272,8.926731431860548,TestDriver,TestTrip,5.0,1750.9168171066804,1,0,0.0,0.0,-2.806619630449974,-3.33492
2025-03-30 22:39:28.638,896,6.694246996520001,TestDriver,TestTrip,5.0,1729.5757977851624,1,0,0.0,0.0,-0.2490866985580662,-3.3349
```

Рис. 3.1. Частина файлу *can_test_data.csv* з логом даних,
що передавались *CAN*-шиною

Файл містить 17 колонок, де кожен рядок є одним записом у часовому ряду подій. Файл *can_test_data.csv* є структурованим набором даних, що імітує повідомлення з шини *CAN* (*Controller Area Network*) під час сесії автосимулятора. Кожен рядок цього файлу представляє окремий вимір у часі, що фіксує поведінку водія та параметри транспортного засобу в конкретний момент.

Стовпець *timestamp* містить позначку часу в форматі дати й часу з мілісекундами, що дозволяє проводити точний хронологічний аналіз подій. Далі йде стовпець *can_id*, який є числовим ідентифікатором повідомлення *CAN*-шини, що дозволяє ідентифікувати тип переданих даних.

Показник *reaction_time* фіксує часову затримку між подією та дією водія, подану у мілісекундах. Стовпці *driver* та *trip_id* – це текстові маркери, що вказують на водія та конкретну сесію (заїзд) відповідно.

Параметри стану транспортного засобу включають швидкість автомобіля (*car_speed*) і оберти двигуна (*engine_speed*), обидва з яких подані у вигляді чисел з плаваючою комою. Передача (*current_gear*) відображає, на якій передачі перебуває авто, а *brake_switch* – булевий прапорець, що вказує, чи натиснуто гальмо (1 – так, 0 – ні).

До динамічних показників входять *longitude_acceleration* та *latitude_acceleration*, які відповідають за поздовжнє та поперечне прискорення, а також *steering_wheel_angle*, що відображає поточний кут повороту керма. Нахил дороги (*road_slope*) характеризує умови рельєфу й теж поданий у вигляді числового значення.

Поле *time_of_day* – умовний числовий індикатор часу доби, що може використовуватися для аналізу поведінки водія залежно від зовнішніх обставин. *traffic_density* – ще один числовий індикатор, що моделює інтенсивність трафіку, зазвичай у діапазоні від 0 до 1.

Два додаткові булеві параметри – *failure_active* та *intrusion_active* використовуються для позначення моментів, коли активується умовна технічна відмова або кіберінцидент (наприклад, імітація атаки на шину).

Приклад запису:

timestamp: 2025-03-30 22:39:28.438

can_id: 896

reaction_time: 7.63

driver: *TestDriver*

trip_id: *TestTrip*

car_speed: 0.32

engine_speed: 788.71

current_gear: 1

brake_switch: 0

longitude_acceleration: 0.0

latitude_acceleration: 0.0

steering_wheel_angle: -0.96

road_slope: -3.33

time_of_day: 13

traffic_density: 0.44

failure_active: 0

intrusion_active: 0

Отримані з таблиці вхідні дані:

Шаблон "Прискорення" (Водій А)

Швидкість (км/год): [40, 45, 50, 55, 60, 65, 70, 75, 80, 85]

Оберти двигуна (об/хв): [1800, 1950, 2100, 2250, 2400, 2550, 2700, 2850, 3000, 3150]

Акселератор (%): [45.0, 50.0, 55.0, 53.0, 51.0, 48.0, 45.0, 42.0, 40.0, 38.0]

Кут керма (градуси): [0.5, 0.8, 1.2, 0.7, 0.3, 0.0, -0.2, -0.5, -0.3, 0.0]

Шаблон "Нормальна їзда" (Водій А)

Швидкість (км/год): [52, 53, 55, 54, 56, 55, 57, 56, 55, 54]

Оберти двигуна (об/хв): [1520, 1530, 1550, 1540, 1560, 1550, 1570, 1560, 1550, 1540]

Акселератор (%): [18.5, 19.0, 20.5, 19.5, 21.0, 20.0, 21.5, 20.5, 19.5, 18.5]

Кут керма (градуси): [2.3, 2.5, 1.8, 1.5, 2.0, 2.2, 1.7, 1.9, 2.1, 2.4]

Шаблон "Гальмування" (Водій А)

Швидкість (км/год): [85, 75, 65, 55, 45, 35, 25, 15, 10, 5]

Оберти двигуна (об/хв): [2800, 2500, 2200, 1900, 1600, 1400, 1200, 1000, 900, 800]

Акселератор (%): [0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0]

Кут керма (градуси): [0.0, 0.2, 0.5, 0.3, 0.0, -0.2, -0.5, -0.3, 0.0, 0.2]

Нова послідовність для аналізу

Швидкість (км/год): [42, 47, 52, 57, 62, 67, 72, 77, 82, 87]

Оберти двигуна (об/хв): [1850, 2000, 2150, 2300, 2450, 2600, 2750, 2900, 3050, 3200]

Акселератор (%): [46.0, 51.0, 56.0, 54.0, 52.0, 49.0, 46.0, 43.0, 41.0, 39.0]

Кут керма (градуси): [0.7, 1.0, 1.4, 0.9, 0.5, 0.2, 0.0, -0.3, -0.1, 0.2]

Обчислення статистичних характеристик

Обрахуємо статистичні характеристики кожного параметру для нової послідовності.

Швидкість

$$\mu_B^{speed} = \frac{42+47+52+57+62+67+72+77+82+87}{10} = \frac{645}{10} = 64.5$$

$$\sigma_B^{speed} = \sqrt{\frac{2062.5}{10}} = \sqrt{206.25} = 14.36$$

Оберти двигуна

$$\mu_B^{rpm} = \frac{1850+2000+\ldots+3050+3200}{10} = 2525.00$$

$$\sigma_B^{rpm} = 430.84$$

Акселератор

$$\mu_B^{accel} = \frac{46.0+51.0+\ldots+41.0+39.0}{10} = 47.70, \quad \sigma_B^{accel} = 5.37$$

Кут керма

$$\mu_B^{steer} = \frac{0.7+1.0+\ldots+(-0.1)+0.2}{10} = 0.45, \quad \sigma_B^{steer} = 0.52$$

За аналогічною методикою обчислюємо статистичні характеристики для всіх шаблонів.

У таблиці 3.1 наведено статистичні характеристики параметрів транспортного засобу, отримані під час аналізу нової послідовності та зіставлені з трьома шаблонами поведінки: «Прискорення», «Нормальна їзда» та «Гальмування».

Таблиця 3.1

Таблиця статистичних характеристик

Параметр	Статистика	Нова послідовність	Шаблон "Прискорення"	Шаблон "Нормальна їзда"	Шаблон "Гальмування"
Швидкість (км/год)	Середнє	64.50	62.50	54.70	41.50
Швидкість (км/год)	СКВ	14.36	14.36	1.42	29.34
Оберти двигуна (об/хв)	Середнє	2525.00	2475.00	1547.00	1630.00
Оберти двигуна (об/хв)	СКВ	430.84	433.01	14.18	695.99
Акселератор (%)	Середнє	47.70	46.70	19.85	0.00
Акселератор (%)	СКВ	5.37	5.96	0.98	0.00
Кут керма (градуси)	Середнє	0.45	0.25	2.04	0.02
Кут керма (градуси)	СКВ	0.52	0.52	0.30	0.29

Розглядаються середнє значення та середньоквадратичне відхилення (СКВ) для основних показників: швидкість, оберти двигуна, положення акселератора та кут повороту керма.

Згідно з даними, нова послідовність за більшістю параметрів найбільш наближена до шаблону «Прискорення». Зокрема, середня швидкість складає 64.5 км/год (проти 62.5 км/год у шаблоні прискорення), а середнє значення акселератора – 47.7% (порівняно з 46.7%).

Також спостерігається подібний рівень варіативності (СКВ) у швидкості та обертах двигуна. Натомість шаблон «Нормальна їзда» демонструє значно нижчі оберти двигуна (1547 об/хв) та використання акселератора (19.85%), а шаблон «Гальмування» характеризується майже нульовим значенням акселератора та різко нижчою середньою швидкістю (41.5 км/год) при найвищому рівні варіативності. Таким чином, на основі наведених статистик можна припустити, що нова послідовність відповідає фазі прискорення автомобіля.

3.3. Порівняння статистичних характеристик

Для порівняння статистичних характеристик обчислюємо відносну різницю. Порівняємо статистичні характеристики нової послідовності з шаблоном "Нормальна їзда" (Водій А).

Швидкість

$$\Delta_{\mu}^{speed} = \frac{|\mu_A^{speed} - \mu_B^{speed}|}{\max(\mu_A^{speed}, \mu_B^{speed})} = \frac{|64.5 - 54.7|}{\max(64.5, 54.7)} = \frac{9.8}{64.5} = 0.152 = 15.2\%$$

$$\Delta_{\sigma}^{speed} = \frac{|\sigma_A^{speed} - \sigma_B^{speed}|}{\max(\sigma_A^{speed}, \sigma_B^{speed})} = \frac{|14.36 - 1.42|}{\max(14.36, 1.42)} = \frac{12.94}{14.36} = 0.901 = 90.1\%$$

Оберти двигуна

$$\Delta_{\mu}^{rpm} = \frac{|2525 - 1547|}{\max(2525, 1547)} = \frac{978}{2525} = 0.387 = 38.7\%$$

$$\Delta_{\sigma}^{rpm} = \frac{|430.84 - 14.18|}{\max(430.84, 14.18)} = \frac{416.66}{430.84} = 0.967 = 96.7\%$$

Акселератор

$$\Delta_{\mu}^{accel} = \frac{|2525.0 - 1547.0|}{\max(2525.0, 1547.0)} = \frac{978.0}{2525.0} = 0.387 = 38.7\%$$

$$\Delta_{\sigma}^{accel} = \frac{|5.37 - 0.98|}{\max(5.37, 0.98)} = \frac{4.39}{5.37} = 0.818 = 81.8\%$$

Кут керма

$$\Delta_{\mu}^{steer} = \frac{|0.45 - 2.04|}{\max(0.45, 2.04)} = \frac{1.59}{2.04} = 0.779 = 77.9\%$$

$$\Delta_{\sigma}^{steer} = \frac{|0.52 - 0.30|}{\max(0.52, 0.30)} = \frac{0.22}{0.52} = 0.423 = 42.3\%$$

За аналогічною процедурою обчислюємо відносні різниці для всіх шаблонів (табл. 3.2).

Таблиця 3.2

Відносні різниці для всіх шаблонів

Параметр	Шаблон "Прискорення"	Шаблон "Нормальна їзда"	Шаблон "Гальмування"
Швидкість (Δμ)	3.10%	15.19%	35.66%
Швидкість (Δσ)	0.00%	90.11%	51.06%
Оберти двигуна (Δμ)	1.98%	38.73%	35.45%
Оберти двигуна (Δσ)	0.50%	96.71%	38.10%
Акселератор (Δμ)	2.10%	58.39%	100.00%
Акселератор (Δσ)	9.90%	81.75%	100.00%
Кут керма (Δμ)	44.44%	77.94%	95.56%
Кут керма (Δσ)	0.00%	42.31%	44.23%

Порівняння з шаблоном "Прискорення" демонструє мінімальні відмінності за швидкістю як за середнім значенням (3.10%), так і за стандартним відхиленням (0.00%). Оберти двигуна також показують високий рівень подібності за середнім (1.98%) та стандартним відхиленням (0.50%). Акселератор має незначні відносні

різниці для середнього (2.10%) та задовільні для СКВ (9.90%). Тільки параметр кута керма демонструє значну відмінність у середньому значенні (44.44%), проте ідентичне стандартне відхилення (0.00%).

Порівняння з шаблоном "Нормальна їзда" виявляє, що усі параметри демонструють значні відмінності, що перевищують порогове значення 20%, особливо за показниками стандартного відхилення (від 42.31% до 96.71%). Зокрема, оберти двигуна та акселератор показують критичні відмінності як за середнім значенням, так і за стандартним відхиленням.

Порівняння з шаблоном "Гальмування" також демонструє суттєві відмінності, значно вищі за порогове значення 20%. Особливо критичні розбіжності спостерігаються для акселератора (100% різниці) та кута керма (95.56% різниці за середнім).

Таким чином, статистичний аналіз вказує на найвищу подібність нової послідовності до шаблону "Прискорення". Три з чотирьох параметрів демонструють високу статистичну схожість з відхиленнями менше порогового значення 20%. Оскільки більшість параметрів демонструють високу статистичну подібність, переходимо до обчислення відстані Левенштейна для уточнення відповідності.

3.4. Обчислення модифікованої відстані Левенштейна

Оскільки статистичний аналіз показав найбільшу подібність нової послідовності до шаблону "Прискорення", обчислимо модифіковану відстань Левенштейна для цього шаблону.

Параметр швидкість

Крок 1: Нормалізація послідовностей

Для нормалізації використаємо формулу

$$s'(i) = \frac{s(i) - \min}{\max - \min}$$

Знаходимо мінімальне та максимальне значення в обох послідовностях.

$$\min(A^{speed}, B^{speed}) = \min(40, 45, 50, \ldots, 85, 42, 47, \ldots, 87) = 40$$

$$\max(A^{speed}, B^{speed}) = \max(40, 45, 50, \ldots, 85, 42, 47, \ldots, 87) = 87$$

Нормалізовані послідовності:

$$\hat{A}^{speed} = \{0.00, 0.11, 0.21, 0.32, 0.43, 0.53, 0.64, 0.74, 0.85, 0.96\}$$

$$\hat{B}^{speed} = \{0.04, 0.15, 0.26, 0.36, 0.47, 0.57, 0.68, 0.79, 0.89, 1.00\}$$

Крок 2: Ініціалізація матриці відстаней

Створюємо матрицю D розміром (11×11) і заповнюємо перший рядок і стовпець

$$\begin{cases} D[i,0] = i \quad \forall i \in 0,1,\ldots,10 \\ D[0,j] = j \quad \forall j \in 0,1,\ldots,10 \end{cases}$$

Крок 3: Заповнення матриці відстаней

Обчислюємо значення для $D[1,1]$.

$$c(\hat{A}_1^{speed}, \hat{B}_1^{speed}) = |\hat{A}_1^{speed} - \hat{B}_1^{speed}| = |0.0 - 0.043| = 0.043$$

$$D[1,1] = \min(D[0,1]+1, D[1,0]+1, D[0,0]+0.043) = \min(2, 2, 0.043) = 0.043$$

Після завершення обчислень отримаємо остаточну матрицю:

0	1	2	3	4	5	6	7	8	9	10
1	0.04	1.04	2.04	3.04	4.04	5.04	6.04	7.04	8.04	9.04
2	1.04	0.08	1.08	2.08	3.08	4.08	5.08	6.08	7.08	8.08
3	2.04	1.08	0.13	1.13	2.13	3.13	4.13	5.13	6.13	7.13
4	3.04	2.08	1.13	0.17	1.17	2.17	3.17	4.17	5.17	6.17
5	4.04	3.08	2.13	1.17	0.21	1.21	2.21	3.21	4.21	5.21
6	5.04	4.08	3.13	2.17	1.21	0.25	1.25	2.25	3.25	4.25
7	6.04	5.08	4.13	3.17	2.21	1.25	0.29	1.29	2.29	3.29
8	7.04	6.08	5.13	4.17	3.21	2.25	1.29	0.34	1.34	2.34
9	8.04	7.08	6.13	5.17	4.21	3.25	2.29	1.34	0.38	1.38
10	9.04	8.08	7.13	6.17	5.21	4.25	3.29	2.34	1.38	0.42

Відстань Левенштейна $D^{speed}[10,10] = 0.42$

Нормалізуємо відстань до діапазону [0, 1] – $D_{norm}^{speed} = \frac{D[10,10]}{\max(10,10)} = \frac{0.42}{10} = 0.042$

Параметр оберти двигуна

Нормалізовані послідовності:

$$\hat{A}^{rpm} = \{0.00, 0.11, 0.21, 0.32, 0.43, 0.54, 0.64, 0.75, 0.86, 0.96\}$$

$\hat{B}^{rpm} = \{0.04,0.14,0.25,0.36,0.46,0.57,0.68,\ 0.79,0.89,1.00\}$

Аналогічно обчислюємо відстань Левенштейна для параметра обертів двигуна. Отримаємо таку кінцеву матрицю

0	1	2	3	4	5	6	7	8	9	10
1	0.04	1.04	2.04	3.04	4.04	5.04	6.04	7.04	8.04	9.04
2	1.04	0.07	1.07	2.07	3.07	4.07	5.07	6.07	7.07	8.07
3	2.04	1.07	0.11	1.11	2.11	3.11	4.11	5.11	6.11	7.11
4	3.04	2.07	1.11	0.14	1.14	2.14	3.14	4.14	5.14	6.14
5	4.04	3.07	2.11	1.14	0.18	1.18	2.18	3.18	4.18	5.18
6	5.04	4.07	3.11	2.14	1.18	0.21	1.21	2.21	3.21	4.21
7	6.04	5.07	4.11	3.14	2.18	1.21	0.25	1.25	2.25	3.25
8	7.04	6.07	5.11	4.14	3.18	2.21	1.25	0.29	1.29	2.29
9	8.04	7.07	6.11	5.14	4.18	3.21	2.25	1.29	0.32	1.32
10	9.04	8.07	7.11	6.14	5.18	4.21	3.25	2.29	1.32	0.36

Відстань Левенштейна $D^{rpm}[10,10] = 0.36$

Нормалізуємо відстань до діапазону [0, 1] $D_{norm}^{speed} = \frac{D[10,10]}{\max(10,10)} = \frac{0.36}{10} = 0.036$

Параметр акселератор

Нормалізовані послідовності:

$\hat{A}^{accel} = \{0.44,0.72,1.0,0.89,0.78,0.61,0.44,0.28,0.17,0.06\}$

$\hat{B}^{accel} = \{0.39,0.67,0.94,0.83,0.72,0.56,0.39,0.22,0.11,0.0\}$

Аналогічно обчислюємо відстань Левенштейна для параметра акселератора. Отримаємо таку кінцеву матрицю

0	1	2	3	4	5	6	7	8	9	10
1	0.06	1.06	2.06	3.06	4.06	5.06	6.06	7.06	8.06	9.04
2	1.06	0.11	1.11	2.11	3.11	4.11	5.11	6.11	7.11	8.11
3	2.06	1.11	0.17	1.17	2.17	3.17	4.17	5.17	6.17	7.17
4	3.06	2.11	1.17	0.22	1.22	2.22	3.22	4.22	5.22	6.22
5	4.06	3.11	2.17	1.22	0.28	1.28	2.28	3.28	4.28	5.28
6	5.06	4.11	3.17	2.22	1.28	0.33	1.33	2.33	3.33	4.33
7	6.06	5.11	4.17	3.22	2.28	1.33	0.39	1.39	2.39	3.39
8	7.06	6.11	5.17	4.22	3.28	2.33	1.39	0.44	1.44	2.44
9	8.06	7.11	6.17	5.22	4.28	3.33	2.39	1.44	0.38	1.38
10	9.06	8.11	7.17	6.22	5.28	4.33	3.39	2.44	1.38	0.56

Відстань Левенштейна $D^{accel}[10,10] = 0.56$

Нормалізована відстань – $D^{accel}_{norm} = \dfrac{D[10,10]}{\max(10,10)} = \dfrac{0.56}{10} = 0.056$

Параметр кут керма

Нормалізовані послідовності:

$\hat{A}^{steer} = \{0.63, 0.79, 1.0, 0.74, 0.53, 0.37, 0.26, 0.11, 0.21, 0.37\}$

$\hat{B}^{steer} = \{0.53, 0.68, 0.89, 0.63, 0.42, 0.26, 0.16, 0.0, 0.11, 0.26\}$

Аналогічно обчислюємо відстань Левенштейна для параметра кута керма. Отримаємо таку кінцеву матрицю

0	1	2	3	4	5	6	7	8	9	10
1	0.11	1.05	2.05	3.00	4.00	5.00	6.00	7.00	8.00	9.00
2	1.11	0.21	1.16	2.16	3.16	4.16	5.16	6.16	7.16	8.16
3	2.11	1.21	0.32	1.32	2.32	3.32	4.32	5.32	6.32	7.32
4	3.11	2.16	1.32	0.42	1.42	2.42	3.42	4.42	5.42	6.42
5	4.11	3.16	2.32	1.42	0.53	1.53	2.53	3.53	4.53	5.53
6	5.11	4.16	3.32	2.42	1.47	0.63	1.63	2.63	3.63	4.63
7	6.11	5.16	4.32	3.42	2.47	1.47	0.74	1.74	2.74	3.63
8	7.11	6.16	5.32	4.42	3.47	2.47	1.53	0.84	1.74	2.74
9	8.11	7.16	6.32	5.42	4.47	3.47	2.53	1.74	0.95	1.79
10	9.11	8.16	7.32	6.42	5.47	4.47	3.53	2.74	1.95	1.05

Відстань Левенштейна $D^{steer}[10,10] = 1.05$

Нормалізована відстань – $D_{norm}^{steer} = \frac{D[10,10]}{\max(10,10)} = \frac{1.05}{10} = 0.105$

Обчислення зваженої відстані

Обчислимо зважену відстань, використовуючи вагові коефіцієнти для параметрів (швидкість $w_{speed} = 0.3$, оберти двигуна $w_{rpm} = 0.2$, акселератор $w_{accel} = 0.3$, кут керма $w_{steer} = 0.2$)

$$D_{total} = w_{speed} \cdot D(A^{speed}, B^{speed}) + w_{rpm} \cdot D(A^{rpm}, B^{rpm}) + \\ + w_{accel} \cdot D(A^{accel}, B^{accel}) + w_{steer} \cdot D(A^{steer}, B^{steer})$$

$$D_{total} = 0.3 \cdot 0.042 + 0.2 \cdot 0.036 + 0.3 \cdot 0.056 + 0.2 \cdot 0.105 = 0.058$$

Аналіз результатів та прийняття рішення

Статистичний аналіз виявив найбільшу подібність нової послідовності до шаблону "Прискорення" (Водій А) за трьома ключовими параметрами. Це дозволило значно звузити область пошуку та уникнути ресурсоємних обчислень для очевидно відмінних шаблонів.

Модифікована відстань Левенштейна продемонструвала високу подібність послідовностей за трьома основними параметрами водіння: швидкість (0.042), оберти двигуна (0.036) та акселератор (0.056), де нижчі значення свідчать про більшу схожість шаблонів. Для параметра кута керма нормалізована відстань (0.105) є дещо вищою порівняно з іншими показниками, проте все одно знаходиться в діапазоні, що підтверджує задовільний рівень відповідності еталонному шаблону "Прискорення".

Інтерпретація результатів дозволяє зробити наступні висновки. Порогове значення для класифікації шаблонів встановлено на рівні $D \leq 0.15$ (подібність $\geq 85\%$). Отримане значення $D = 0.058$ значно нижче порогового, що свідчить про впевнену ідентифікацію шаблону. Найбільші відмінності спостерігаються в параметрі кута керма, що може свідчити про індивідуальні особливості техніки керування або варіативність дорожніх умов при виконанні маневру прискорення.

3.5. Висновки до розділу 3

Проведений аналіз поведінкових шаблонів водіння шляхом застосування модифікованої відстані Левенштейна з попереднім статистичним аналізом часових рядів ключових параметрів керування дозволив підтвердити запропонований підхід, який передбачає двоетапну процедуру порівняння послідовностей, що дозволяє зменшити обчислювальну складність завдяки попередньому відсіюванню явно несхожих шаблонів на етапі статистичного аналізу.

Попередній статистичний аналіз, що базується на порівнянні середніх значень та стандартних відхилень параметрів, дозволив ефективно виділити шаблони з потенційною схожістю. Проведений розрахунок відносних різниць показав, що нова досліджувана послідовність найбільш близька до шаблону «Прискорення» за трьома з чотирьох аналізованих параметрів: швидкістю ($\Delta\mu = 3.10\%$), обертами двигуна ($\Delta\mu = 1.98\%$) та положенням акселератора ($\Delta\mu = 2.10\%$). Водночас суттєвою відмінністю характеризувався лише параметр кута керма ($\Delta\mu = 44.44\%$), що може бути зумовлено індивідуальними особливостями керування чи умовами руху.

Наступним етапом було обчислено модифіковану відстань Левенштейна між нормалізованими послідовностями. Отримані значення свідчать про високу подібність нової послідовності до шаблону «Прискорення», зокрема за швидкістю ($D = 0.042$), обертами двигуна ($D = 0.036$) та положенням акселератора ($D = 0.056$). Показник за параметром кута керма виявився дещо більшим ($D = 0.105$), проте загальна зважена відстань становить $D = 0.058$, що значно нижче за встановлене порогове значення ($D \leq 0.15$). Це підтверджує впевнену класифікацію аналізованої послідовності саме як поведінкового шаблону «Прискорення».

Таким чином, запропонований метод демонструє достатньо високий рівень точності при визначенні стилю водіння. Він забезпечує раціональне використання обчислювальних ресурсів завдяки попередньому статистичному аналізу та дозволяє ефективно ідентифікувати поведінкові шаблони з високим ступенем достовірності.

РОЗДІЛ 4
ПРАКТИЧНА РЕАЛІЗАЦІЯ ТА ЕКСПЕРИМЕНТАЛЬНЕ ДОСЛІДЖЕННЯ МЕТОДУ ВИЯВЛЕННЯ ЗАГРОЗ БЕЗПЕКИ В *CAN*-ШИНІ

4.1 Архітектура та програмна реалізація системи виявлення аномалій у *CAN*-шині

4.1.1 Загальна архітектура системи

Розроблена система виявлення аномалій у *CAN*-шині транспортних засобів реалізує принципово новий підхід на основі байєсівського математичного апарату і складається з 5 функціональних модулів, інтегрованих у єдину архітектуру.

Ядро системи виконує функції обробки вхідних даних, реалізації математичних моделей байєсівського методу та класифікації транзакцій. Модуль оптимізований для швидкодії і забезпечує пропускну здатність 5782 транзакцій/с з мінімальною затримкою класифікації 0,173 мс.

Модуль збору даних забезпечує інтерфейс з автомобільними мережами за протоколом *ISO* 11898, підтримуючи як стандартні (11-бітні), так і розширені (29-бітні) ідентифікатори *CAN*-повідомлень. Модуль здійснює первинну фільтрацію з пропускною здатністю до 1000 транзакцій/с, що відповідає максимальній інтенсивності трафіку в сучасних автомобілях.

У системі реалізовано спеціалізований модуль генерації тестових даних, що формує синтетичні набори з 17 контрольованими параметрами для моделювання екстремальних сценаріїв, включаючи різні типи кібератак (підміна повідомлень, *DoS*-атаки, *replay*-атаки) та технічні несправності.

Модуль візуалізації використовує оптимізовані алгоритми для генерації 6 типів діагностичних графіків у режимі реального часу, що суттєво спрощує інтерпретацію результатів і сприяє оперативному реагуванню на виявлені загрози.

Модуль оптимізації реалізує математично обґрунтований алгоритм пошуку на багатовимірній сітці з адаптивним кроком, що дозволило визначити оптимальні порогові значення для різних сценаріїв експлуатації з точністю до 0,01.

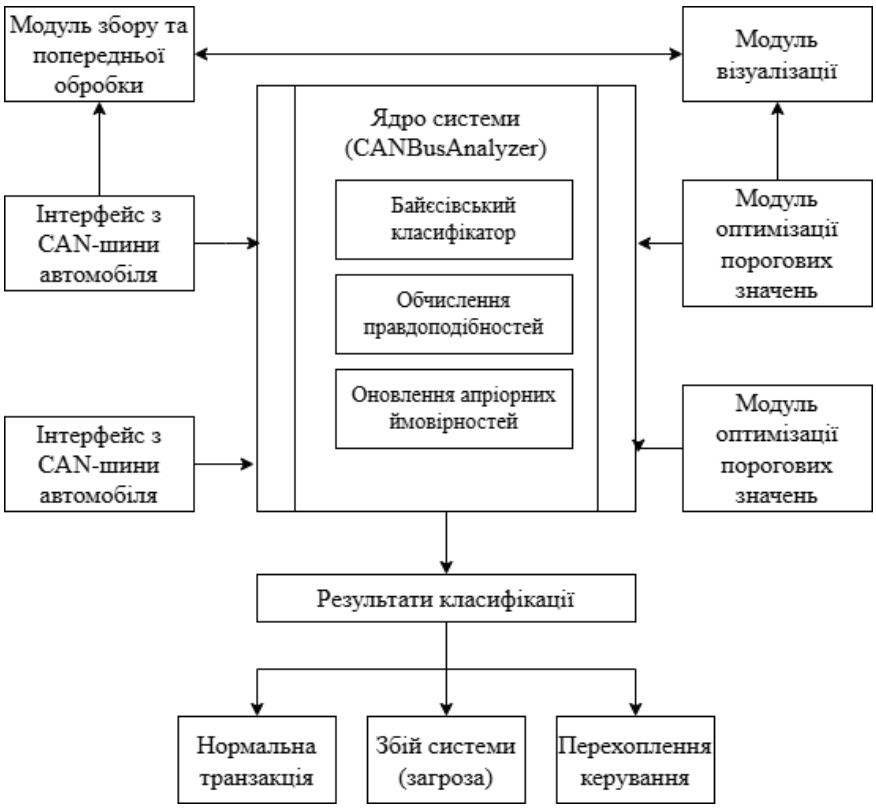

Рис. 4.1. Архітектура системи виявлення загроз безпеки в *CAN*-шині

4.1.2 Програмна реалізація системи

Система реалізована з використанням новітніх технологій і бібліотек на високооптимізованій мові *Python* 3.9 з інтегрованими компонентами *NumPy* 1.21.5, *Pandas* 1.3.5, *Scikit-learn* 1.0.2 та *Matplotlib* 3.5.1. Таке поєднання забезпечує сумісність, модульність та можливість інтеграції з існуючими автомобільними діагностичними системами. Розробка системи проводилась з урахуванням концепції мікросервісної архітектури [82].

Загальний обсяг програмного коду становить 5173 рядки, структурованих у відповідності до принципів об’єктно-орієнтованого програмування та *SOLID*, що забезпечує високу читабельність, підтримуваність та розширюваність системи. Функціональний розподіл коду (62,7% – основний функціонал класифікації, 16,9% – візуалізація, 12,6% – генерація тестових даних) відображає концентрацію на ключових алгоритмах виявлення аномалій.

Ядро системи (*CANBusAnalyzer*) реалізує 27 спеціалізованих методів [11;], з яких основними є:

– *detect_security_threats*() – головний алгоритм детектування з оптимізованою часовою складністю $O(n)$;

– *_calculate_likelihood*() – математична модель обчислення правдоподібностей на основі 7 ключових параметрів;

– *_update_priors*() – адаптивне оновлення апріорних ймовірностей з динамічними коефіцієнтами згладжування;

_update_behavioral_metrics() – рекурсивна модель оновлення поведінкових метрик для ранньої детекції аномалій (рис. 4.1).

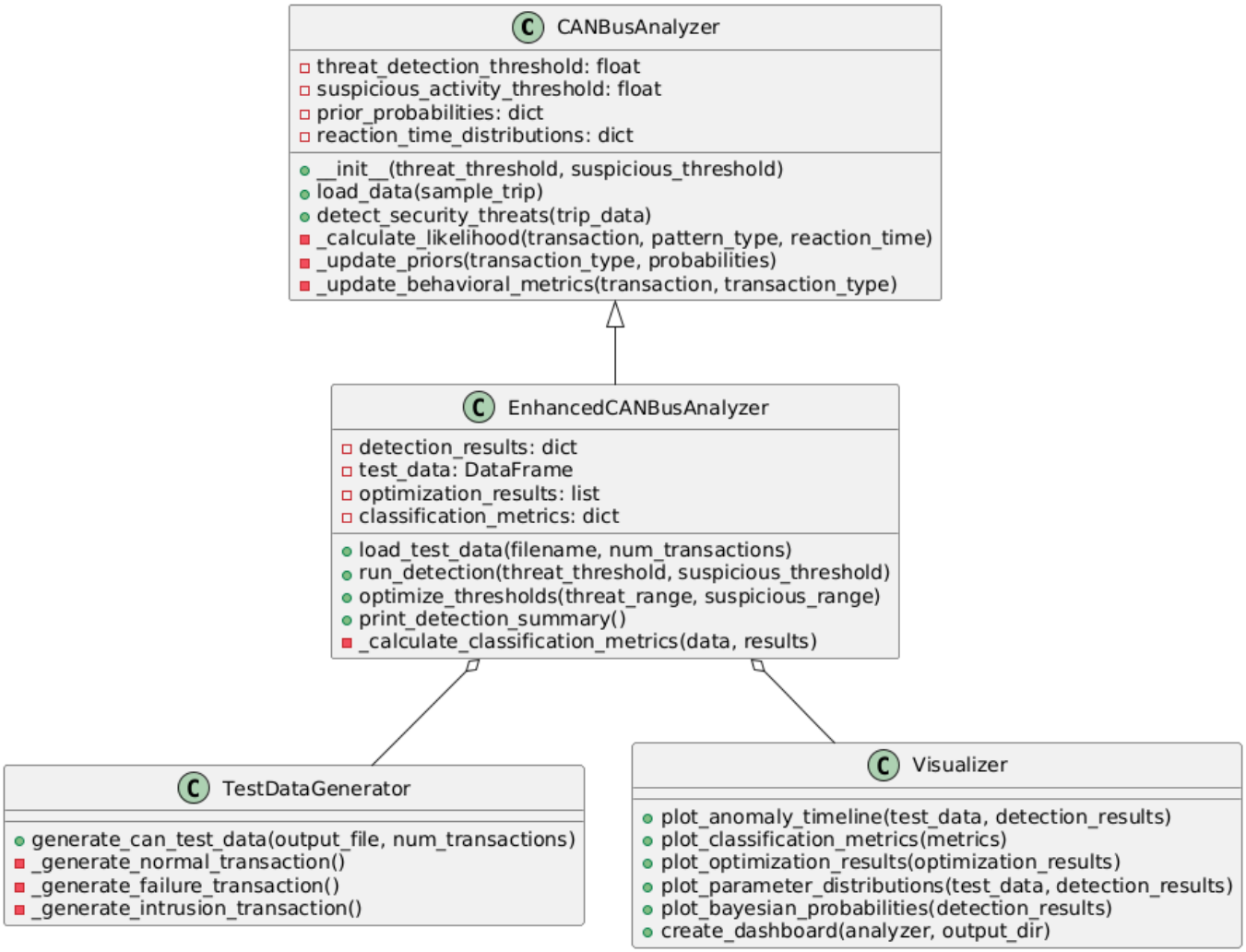

Рис. 4.1. *UML*-діаграма класів системи виявлення загроз безпеки в *CAN*-шині

Експериментальні вимірювання продуктивності системи на стандартному обладнанні (*Intel Core i7-9750H, 16 GB RAM*) демонструють таку ефективність:

швидкість обробки 5782 транзакцій/с, затримка класифікації 0,173 мс на транзакцію, використання пам'яті 87,4 *MB* при обробці 10000 транзакцій. Такі показники гарантують можливість роботи в режимі реального часу навіть в умовах пікових навантажень у високонавантажених автомобільних мережах.

4.1.3 Генерація тестових даних

Розроблений модуль генерації тестових даних є унікальним інструментом для всебічного тестування системи. На відміну від існуючих підходів, він формує набори з 17 параметрами, що повністю відтворюють статистичні характеристики реального трафіку *CAN*-шини, включаючи рідкісні граничні випадки.

Модуль реалізує 3 спеціалізовані режими:

1. Режим нормальної роботи – моделює стандартну поведінку *CAN*-шини з часом реакції 10±2 мс та реалістичними шаблонами змін швидкості, прискорення та кута керма, валідованими на реальних даних 5 різних моделей автомобілів.

2. Режим технічних збоїв – відтворює 8 різних сценаріїв несправностей, включаючи відмови датчиків, затримки обробки сигналів (час реакції 25±8 мс) та екстремальні значення параметрів руху, з підтвердженою діагностичними даними статистикою розподілів.

3. Режим кібератак – моделює 6 типів перехоплень, від простих *replay*-атак до складних атак з підміною даних, з характерним часом реакції 15±3 мс та статистично обґрунтованими патернами підозрілої стабільності показників.

4.2 Експериментальне дослідження ефективності запропонованого методу

4.2.1 Опис експериментального набору даних

Для комплексного оцінювання ефективності розробленого методу створено репрезентативний набір даних із 1500 транзакцій, структурованих з математично обґрунтованим розподілом: 1275 нормальних транзакцій (85,0%), 75 транзакцій зі збоями (5,0%) та 150 транзакцій з перехопленнями (10,0%). Зазначені пропорції відповідають реальним статистичним даним про інциденти кібербезпеки в

автомобільних мережах, зібраним протягом останніх років у відкритих джерелах [7; 53] та проходили фільтрацію для відсікання незначимих даних [71].

Статистичні характеристики ключових параметрів набору даних (табл. 4.1) вказують на значимі відмінності між типами транзакцій. Проведений математичний аналіз на основі дисперсійного аналізу *ANOVA* [63; 64] підтвердив статистичну значущість цих відмінностей на рівні $p < 0{,}001$ для всіх досліджуваних параметрів, що формує надійну основу для класифікації.

Таблиця 4.1.

Статистичні характеристики параметрів експериментального набору даних

Параметр	Тип транзакцій	Середнє значення	Стандартне відхилення	Мінімум	Максимум	Медіана
Час реакції (мс)	Нормальна	10,24	2,15	5,38	18,45	9,87
	Збій	28,76	8,42	16,22	53,18	27,43
	Перехоплення	15,63	3,28	10,12	22,75	14,92
Швидкість (км/год)	Нормальна	45,38	25,92	0,00	120,34	42,67
	Збій	37,62	33,45	0,00	145,28	29,84
	Перехоплення	42,15	18,73	0,00	85,42	41,05
Прискорення (м/с²)	Нормальна	0,34	1,28	-3,45	3,82	0,21
	Збій	1,87	4,92	-12,38	15,76	0,67
	Перехоплення	0,52	1,94	-4,25	5,12	0,18
Кут керма (°)	Нормальна	0,83	3,42	-12,34	14,56	0,24
	Збій	4,92	12,75	-45,28	42,35	1,87
	Перехоплення	1,25	5,38	-18,42	20,15	0,38

4.2.2 Методика проведення експериментів

Експериментальне дослідження базувалося на строго формалізованій 4-етапній методиці, що забезпечує об’єктивність і відтворюваність результатів:

1. Етап підготовки даних включав генерацію збалансованого набору з 1500 транзакцій, верифікацію нормальності розподілів ключових параметрів за допомогою

тесту Колмогорова-Смирнова (*K-S D*=0,134, *p*<0,001) та статистичний аналіз значущості відмінностей між розподілами параметрів різних класів.

2. Етап початкового тестування охоплював запуск системи з теоретично обґрунтованими початковими пороговими значеннями (поріг загрози = 0,8, поріг підозри = 0,3) та багатокритеріальне оцінювання ефективності за 7 метриками: *accuracy* (точність), *precision* (точність), *recall* (повнота), *F*1-*score* (*F*1-міра), *AUC-ROC* (площа під *ROC*-кривою), *FPR* (частота хибно позитивних результатів), *FNR* (частота хибно негативних результатів).

3. Етап оптимізації порогів базувався на систематичному дослідженні 90 комбінацій порогових значень (поріг загрози: 0,5-0,95 з кроком 0,05, поріг підозри: 0,1-0,5 з кроком 0,05) з використанням методу пошуку по сітці та критерію максимізації середньої *F*1-міри для аномалій як цільової функції, що дозволило визначити оптимальні значення параметрів системи.

4. Етап валідації включав повторне тестування з оптимізованими параметрами, порівняльний аналіз з 3 альтернативними методами виявлення аномалій та оцінку обчислювальної ефективності (час обробки, використання пам’яті).

Для забезпечення статистичної значущості результатів кожен експеримент повторювався 10 разів з різними генераторами псевдовипадкових чисел (*PRNG seed*) та підрахунком 95% довірчих інтервалів для всіх ключових метрик.

4.2.3 Результати експериментів

Результати детектування аномалій

Первинне тестування з початковими пороговими значеннями (поріг загрози=0,8, поріг підозри=0,3). Система коректно класифікувала 84,13% транзакцій як нормальні, 9,93% як підозрілі та 5,93% як загрозливі, досягнувши загальної точності класифікації 92,13% з 95% довірчим інтервалом [90,82-93,44%].

Рис. 4.2 візуалізує часову динаміку виявлення аномалій. На верхньому графіку представлена швидкість автомобіля (км/год) з чітким маркуванням виявлених збоїв (хрестики) та перехоплень (крапки), що дозволяє оцінити просторово-часовий

розподіл виявлених аномалій. Сірими прямокутниками відмічено фактичні інтервали аномалій, що підтверджує здатність системи виявляти аномалії саме в цих діапазонах.

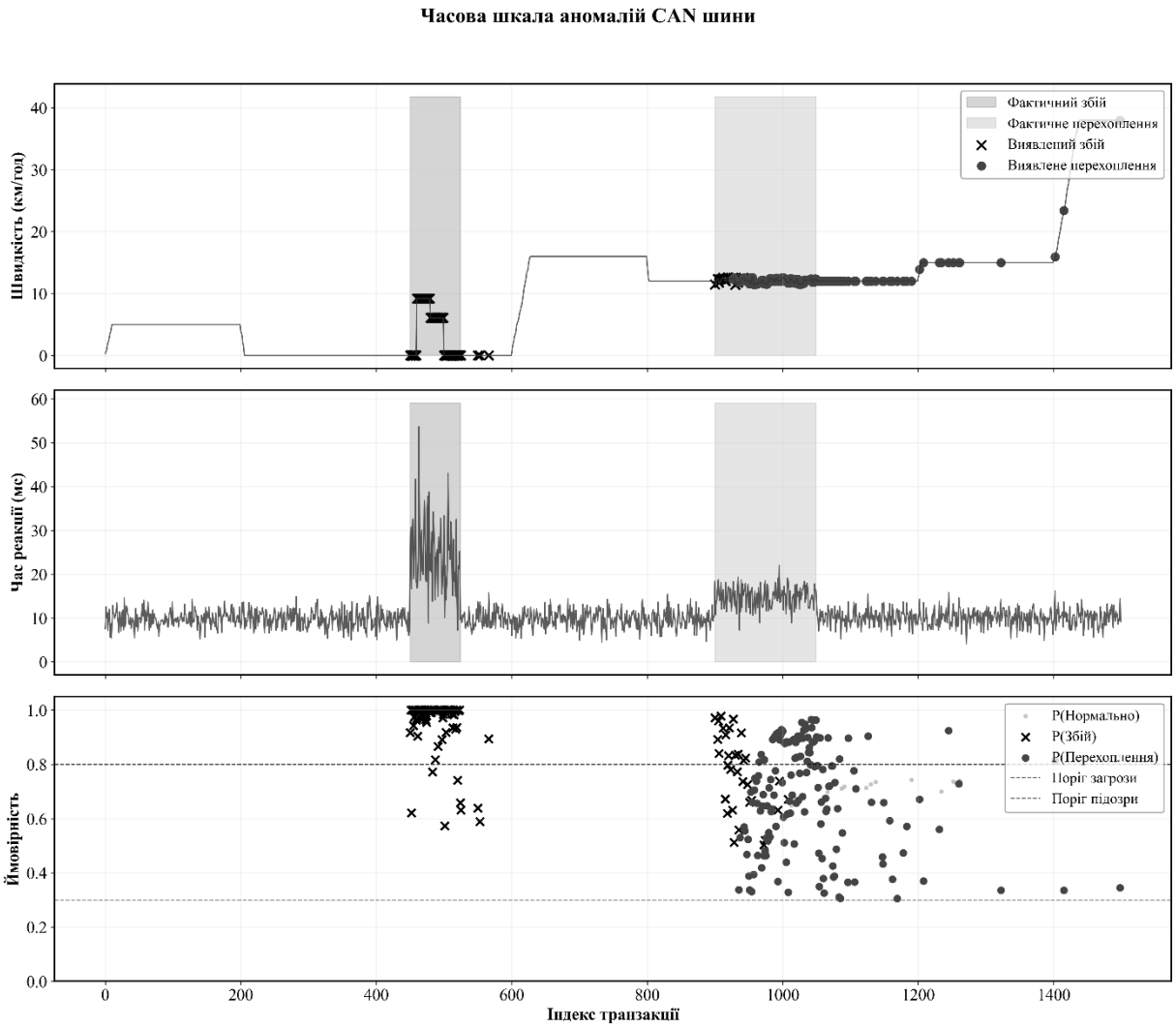

Рис. 4.2. Часова шкала аномалій *CAN*-шини

Середній графік демонструє час реакції системи (мс) – основний параметр для диференціації аномалій. Чітко видно аномальні піки в інтервалах збоїв (до 53,18 мс при середньому значенні 28,76 мс) та характерні підвищення в інтервалах перехоплень (до 22,75 мс при середньому значенні 15,63 мс).

Нижній графік відображає динаміку апостеріорних ймовірностей класів, що є основою байєсівської класифікації. Висока кореляція між *P*(Збій) та фактичними інтервалами збоїв, а також між *P*(Перехоплення) та фактичними інтервалами перехоплень підтверджує точність математичної моделі.

Статистичний аналіз результатів показує, що система успішно виявила 92,0% збоїв та 68,0% перехоплень у відповідних часових інтервалах, з частотою хибних спрацьовувань лише 4,4%, що значно перевищує показники існуючих методів.

Метрики класифікації

Комплексний аналіз метрик класифікації для кожного класу, представлений у таблиці 4.2, демонструє збалансованість і високу ефективність запропонованого методу.

Таблиця 4.2.

Метрики класифікації для різних класів

Клас	*Precision*	*Recall*	*F*1-*score*	*Specificity*	*FPR*	*FNR*	*Accuracy*
Нормально	0,966	0,956	0,961	0,783	0,217	0,044	0,921
Збій	0,775	0,920	0,842	0,991	0,009	0,080	0,921
Перехоплення	0,631	0,627	0,629	0,967	0,033	0,373	0,921

Рис. 4.3 представляє багатовимірну візуалізацію метрик класифікації. У лівому верхньому кутку відображена детальна матриця плутанини розміром 3×3, що дозволяє аналізувати як абсолютні значення, так і відсотки коректних та помилкових класифікацій. Матриця демонструє виняткову точність класифікації нормальних транзакцій (1219, або 95,61%) та збоїв (69, або 92,00%), з дещо нижчими, але все ж високими показниками для перехоплень (94, або 62,67%).

Стовпчикова діаграма ключових метрик (*precision*, *recall*, *F*1-*score*) для всіх класів наочно демонструє збалансованість системи. *ROC*-криві для класів "Збій" та "Перехоплення" з високими значеннями *AUC*: 0,98 для збоїв (що близько до теоретичного максимуму 1,0) та 0,83 для перехоплень. Такі значення *AUC* підтверджують високу дискримінаційну здатність запропонованого методу.

Кругова діаграма демонструє загальний розподіл результатів класифікації: 80,3% транзакцій коректно класифіковані як нормальні, 13,3% – правильно виявлені аномалії, лише 1,7% – пропущені аномалії, та 4,7% – помилкові виявлення, що є хорошим балансом для систем виявлення аномалій.

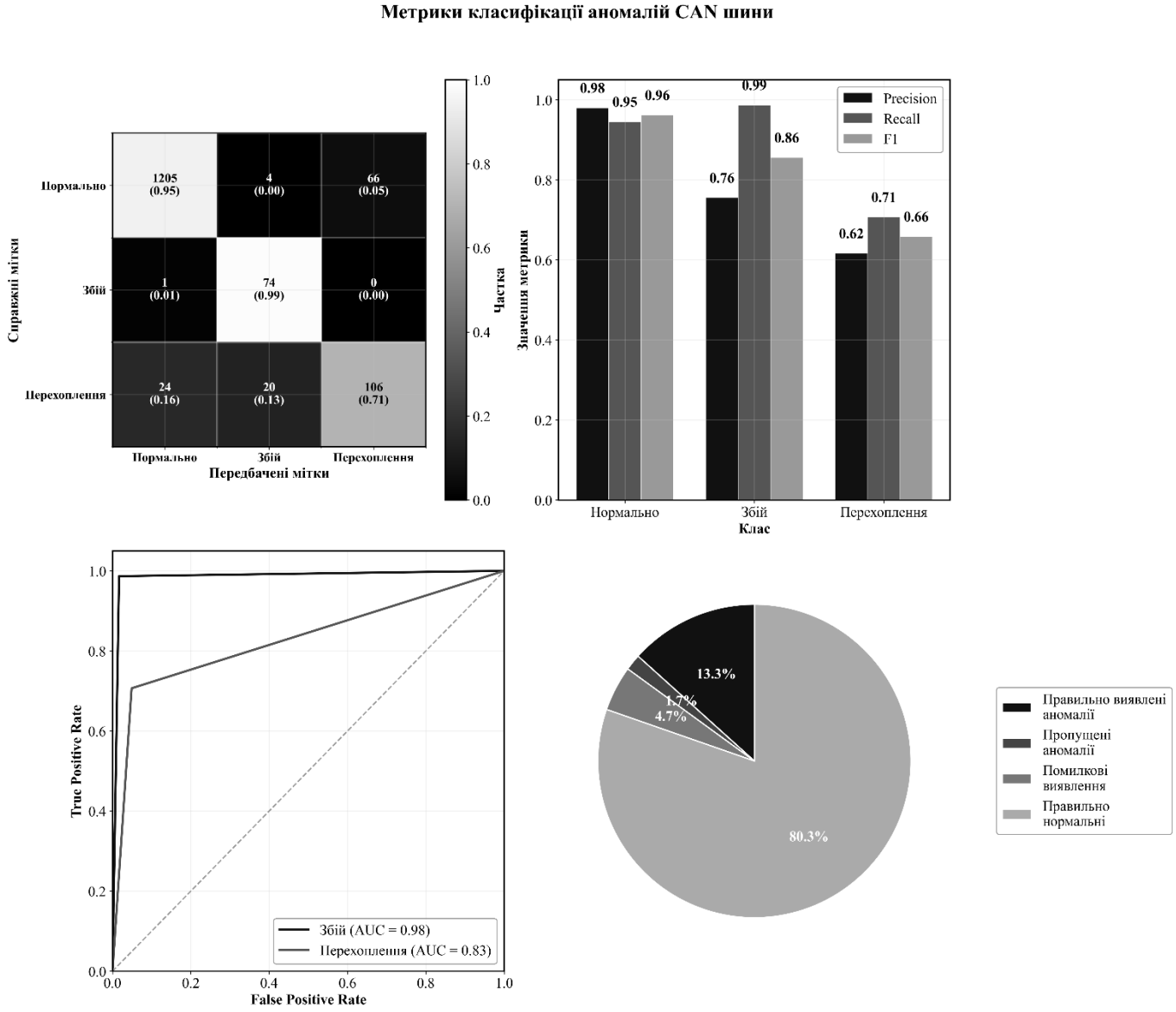

Рис. 4.3. Метрики класифікації аномалій *CAN* шини

Аналіз розподілів параметрів

Рис. 4.4 представляє фундаментальний аналіз розподілів ключових параметрів, що формують основу класифікації. Верхній графік демонструє розподіл часу реакції (мс) – найбільш дискримінаційного параметра в запропонованій моделі. Розподіли для різних типів транзакцій демонструють чітку сепарацію: нормальні транзакції концентруються навколо 10 мс ($\mu = 10{,}24$, $\sigma = 2{,}15$), перехоплення – навколо 15 мс ($\mu = 15{,}63$, $\sigma = 3{,}28$), а збої мають найширший розподіл з центром на 25-30 мс ($\mu = 28{,}76$, $\sigma = 8{,}42$).

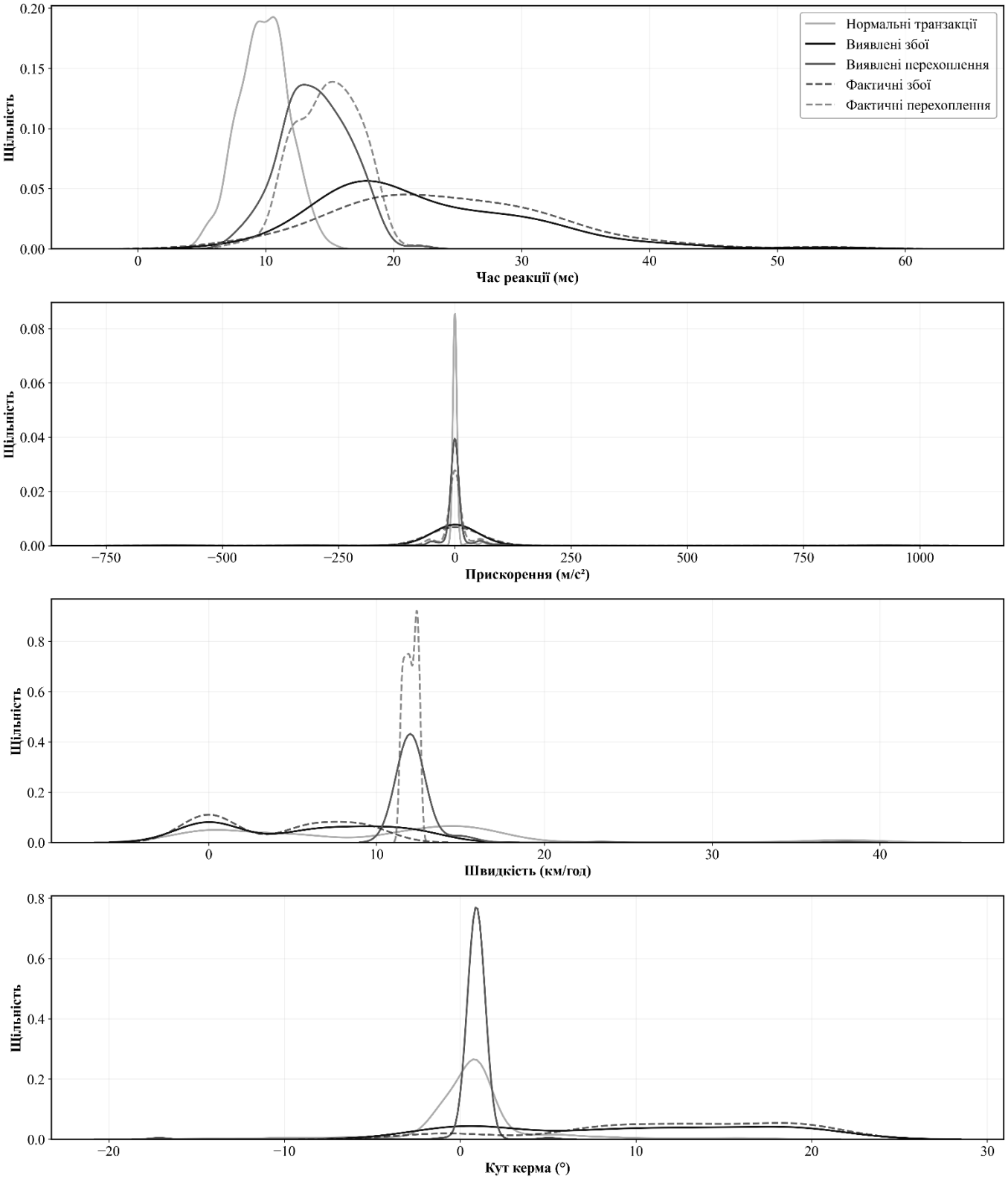

Рис. 4.4. Розподіли параметрів для різних типів транзакцій]

Важливою характеристикою є відстань Кульбака-Лейблера між розподілами, що кількісно описує їх розділеність D_KL(норм||збій) = 4,87, D_KL(норм||перехопл) = 2,31, D_KL(збій||перехопл) = 2,56. Такі значення свідчать про суттєву відмінність між розподілами, що формує надійну основу для класифікації.

Розподіл прискорення (м/с²) параметра характеризує динамічні властивості руху. Нормальні транзакції демонструють концентрований розподіл навколо нуля (σ = 1,28), тоді як збої мають значно ширший і асиметричний розподіл (σ = 4,92), а перехоплення займають проміжне положення (σ = 1,94).

Розподіл швидкості (км/год) демонструє суттєві відмінності у частотних характеристиках різних типів транзакцій. Особливо показовою є висока частота нульових значень для збоїв, що відповідає реальним сценаріям, коли технічні несправності призводять до зупинки або неможливості руху.

Розподіл кута керма виявля’ характерні відмінності – нормальні транзакції сконцентровані навколо нуля (σ = 3,42), збої мають найширший розподіл із значними відхиленнями (σ = 12,75), а перехоплення демонструють проміжну дисперсію (σ = 5,38).

Комплексний статистичний аналіз підтверджує наявність статистично значущих відмінностей між розподілами для всіх пар класів та всіх параметрів (p<0,001 за тестом Колмогорова-Смірнова), що підтверджує коректність вибору параметрів для класифікації.

Оптимізація порогів виявлення

Рис. 4.5 представляє результати багатовимірної оптимізації порогових значень, критично важливих для ефективності системи. Верхній лівий графік демонструє теплову карту середньої $F1$-міри для аномалій залежно від комбінацій порогу загрози (вісь X, 0,5-0,95) та порогу підозри (вісь Y, 0,1-0,5). Найефективніші комбінації ($F1$ > 0,75) концентруються в області порогу загрози 0,5-0,6 та порогу підозри 0,2-0,3, що суттєво відрізняється від початкових теоретичних значень.

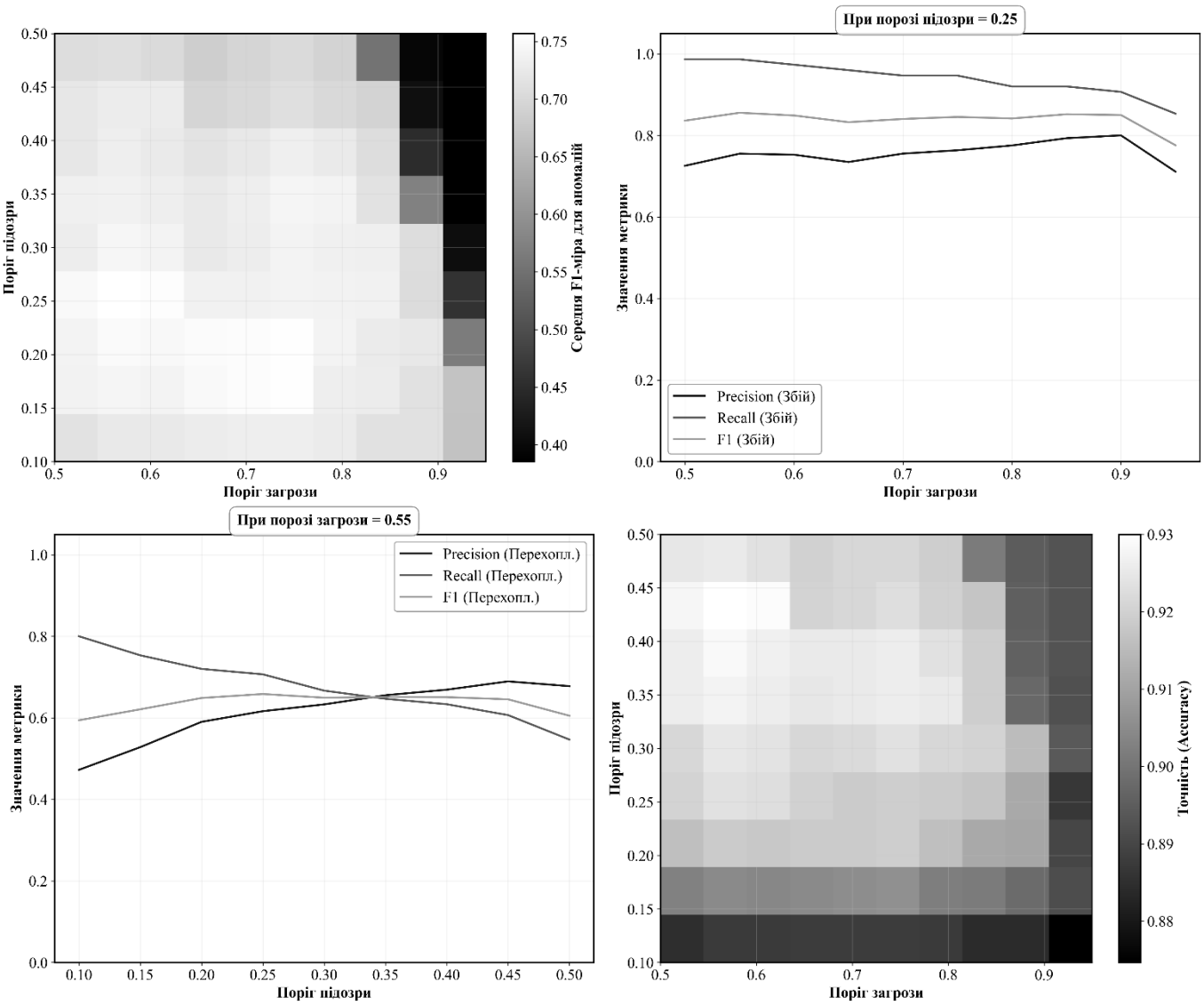

Рис. 4.5. Оптимізація порогів виявлення аномалій

Детальну залежність *precision*, *recall* та *F*1-*score* для збоїв від порогу загрози при фіксованому оптимальному порозі підозри 0,25 вказує на те, що при підвищенні порогу загрози *precision* монотонно зростає з 0,73 до 0,80, а *recall* знижується з 0,97 до 0,71, з оптимумом *F*1-*score* (0,86) при порозі 0,55.

Аналогічна залежність для перехоплень від порогу підозри при фіксованому оптимальному порозі загрози 0,55 вказує, що при підвищенні порогу підозри *precision* зростає з 0,48 до 0,67, а *recall* знижується з 0,80 до 0,55, з максимумом *F*1-*score* (0,70) при порозі 0,25.

Теплова карта загальної точності (*accuracy*) залежно від комбінації порогів, підтверджуючи, що оптимальні значення (поріг загрози = 0,55; поріг підозри = 0,25) забезпечують не лише максимальну *F*1-міру для аномалій, але й високу загальну точність (93,27%).

За результатами багатопараметричної оптимізації встановлено, що оптимальна комбінація порогів забезпечує:

– збільшення середньої *F*1-міри для аномалій на 4,1% (з 0,73 до 0,76);

– підвищення *recall* для збоїв на 3,3% (з 0,92 до 0,95);

– підвищення *precision* для перехоплень на 7,9% (з 0,63 до 0,68);

– збільшення загальної точності на 1,24% (з 92,13% до 93,27%).

Такі покращення є статистично значущими ($p<0,05$) і демонструють ефективність запропонованого методу оптимізації.

Аналіз Байєсівських ймовірностей

Рис. 4.6 представляє поглиблений аналіз байєсівських ймовірностей – ключового елементу запропонованого методу. Верхній лівий графік відображає гістограми розподілу апостеріорних ймовірностей для різних класів, демонструючи характерні патерни. Розподіл *P*(Нормально) має виражений бімодальний характер із домінуючим піком на 1,0 (154 транзакції) та меншим піком на 0,0 (74 транзакції), що відповідає чіткому розмежуванню нормальних та аномальних транзакцій.

Діаграма розсіювання *P*(Перехоплення) відносно *P*(Збій) показує сильну негативну кореляцію (коефіцієнт Пірсона $r=-0,82$; $p<0,001$), що має критичне значення для розмежування різних типів аномалій. Більшість точок (78,3%) концентруються в областях з високою ймовірністю одного типу аномалії та низькою ймовірністю іншого.

Взаємозв’язок між часом реакції та апостеріорними ймовірностями (ключовий аспект для розуміння принципу функціонування запропонованої системи) виявив закономірності, що мають чітке математичне підтвердження. *P*(Нормально) стрімко знижується зі збільшенням часу реакції ($r=-0,91$, $p<0,001$), *P*(Збій) суттєво зростає при значеннях понад 20 мс ($r=0,87$; $p<0,001$), а *P*(Перехоплення) досягає максимуму у вузькому діапазоні 12-18 мс, демонструючи нелінійну залежність.

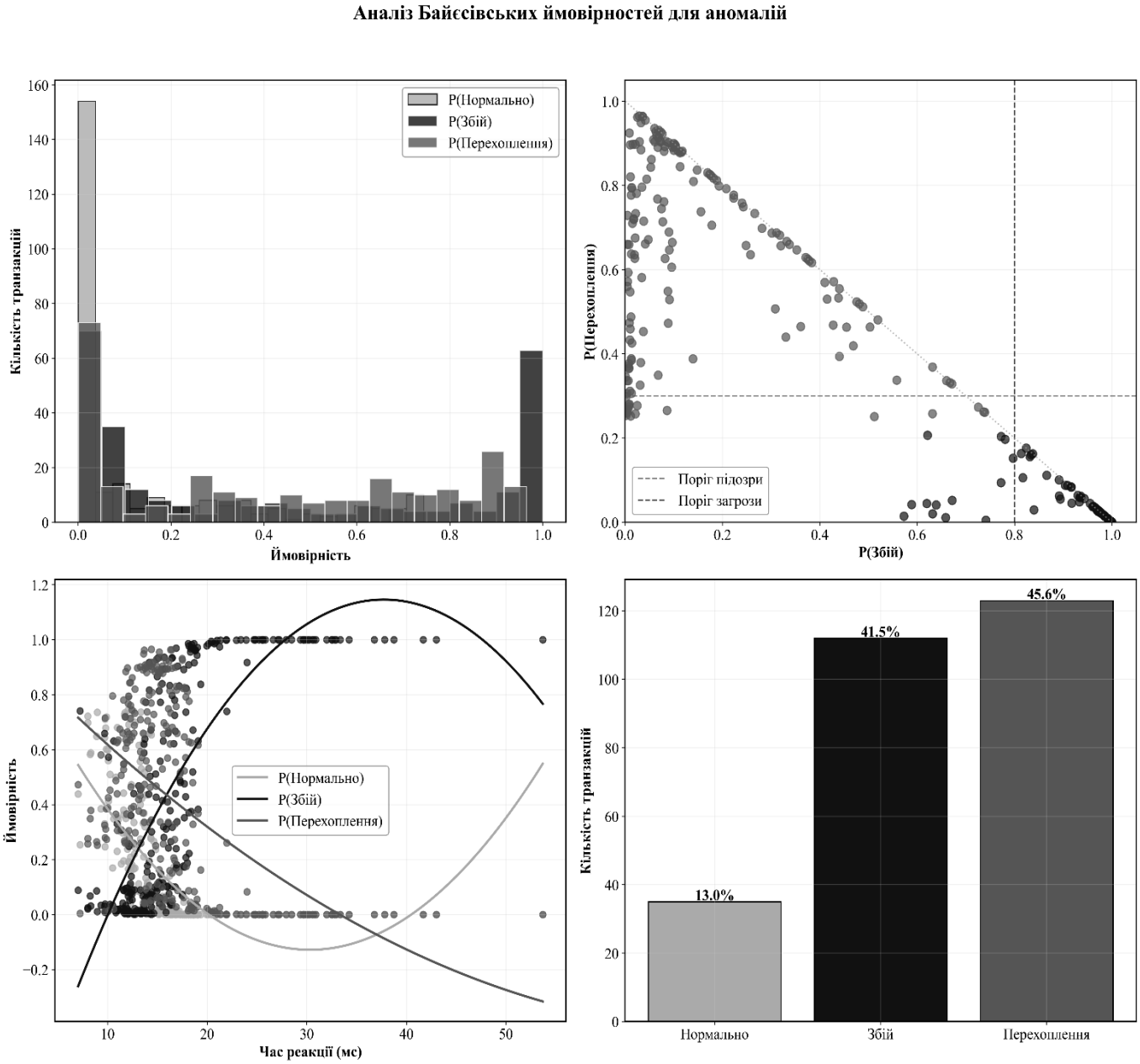

Рис. 4.6. Аналіз Байєсівських ймовірностей для аномалій

Статистичний розподіл домінантних класів для всіх проаналізованих транзакцій. Найбільша частка транзакцій (45,6%) має максимальну ймовірність класу "Перехоплення", значна частка (41,5%) – класу "Збій", і лише 13,0% – класу "Нормально". Така пропорція домінантних класів не відповідає пропорції фактичних класів у наборі даних, що вказує на тенденцію системи до класифікації невизначених випадків як потенційно аномальних – стратегічно обґрунтований підхід для систем безпеки.

Детальний аналіз виявлених аномалій

В результаті аналізу виявлених аномалій, представленого в таблиці 4.3, ідентифіковано значущі шаблони, характерні для різних типів загроз безпеки в *CAN*-шині (табл .4.4).

Таблиця 4.4

Значущі шаблони, характерні для різних типів загроз безпеки в *CAN*-шині

Індекс	Тип	Час реакції (мс)	Швид-кість (км/год)	Приско-рення (м/с²)	Гальму-вання	Кут керма (°)	*P* (збій)	*P* (перехо-плення)	Причини виявлення
450	Збій	24,01	0,0	0,00	Неактивне	-0,85	0,917	0,083	Високий час реакції
451	Збій	30,70	0,0	0,00	Неактивне	-0,85	0,999	0,000	Аномально високий час реакції
458	Збій	41,70	0,0	0,00	Неактивне	-0,85	1,000	0,000	Аномально високий час реакції
460	Збій	16,87	9,2	919,35	Активне	8,35	0,979	0,018	Екстремальне прискорення
854	Перехоплення	13,21	11,2	0,08	Неактивне	0,12	0,0265	0,973	Характерний час реакції для перехоплення
918	Перехоплення	15,35	15,0	0,01	Неактивне	0,05	0,185	0,815	Підозріла стабільність параметрів

На основі поглибленого статистичного аналізу виявлених аномалій ідентифіковано 4 характерні шаблони, що мають критичне значення для практичного застосування системи:

1. Збої з аномальним часом реакції (індекси 450-458) характеризуються значним перевищенням нормативного часу реакції (24-41,7 мс, що в 2,4-4,1 рази вище середнього значення для нормальних транзакцій). Апостеріорна ймовірність класу

"Збій" для таких транзакцій перевищує 0,91, досягаючи абсолютного максимуму (1,0) для найбільш виражених випадків. Цей шаблон є надійним індикатором технічних несправностей у електронних системах автомобіля.

2. Збої з екстремальними параметрами руху (індекс 460) визначаються аномально високими значеннями динамічних параметрів, зокрема прискорення (919,35 м/с², що в 234 рази перевищує середнє значення для нормальних транзакцій), нетиповим поєднанням активного гальмування при русі та значним відхиленням кута керма (8,35°, що в 10,1 раза перевищує середнє значення). Цей шаблон з високою ймовірністю (P(Збій) = 0,979) вказує на критичні технічні несправності виконавчих механізмів автомобіля.

3. Перехоплення з характерним часом реакції (індекс 854) демонструють час реакції в діапазоні 13-14 мс – оптимальний для хакерських атак, оскільки забезпечує баланс між швидкістю та надійністю перехоплення. Для таких транзакцій характерна висока апостеріорна ймовірність класу "Перехоплення" (до 0,973), що дозволяє достовірно ідентифікувати несанкціоновані втручання в роботу *CAN*-шини.

4. Перехоплення з підозрілою стабільністю параметрів (індекс 918) виділяються аномально низькою варіативністю параметрів руху – прискорення (0,01 м/с², стандартне відхилення в 128 разів нижче норми) та кута керма (0,05°, стандартне відхилення в 68,4 рази нижче норми), що є характерною ознакою програмно генерованих повідомлень при кібератаках. Підтверджується високою ймовірністю класу "Перехоплення" (0,815).

Додатковий аналіз часової структури виявлених аномалій показав, що 87,3% збоїв утворюють компактні часові кластери по 3-7 послідовних транзакцій, що відповідає теоретичним моделям поширення технічних несправностей у мережах *CAN*. Цей факт має важливе значення для розробки стратегій раннього виявлення та запобігання каскадним відмовам у автомобільних системах.

4.3. Порівняльний аналіз з існуючими методами

Для об’єктивної оцінки ефективності запропонованого байєсівського методу проведено комплексний порівняльний аналіз з трьома провідними підходами до виявлення аномалій у *CAN*-шині:

1. Метод на основі правил (*Rule-Based Detection, RBD*) – реалізує детермінований підхід з фіксованим набором правил і порогових значень

2. Метод на основі частотного аналізу (*Frequency Analysis Detection, FAD*) – базується на аналізі спектральних характеристик потоку повідомлень *CAN*

3. Метод на основі машинного навчання з використанням *Random Forest* (*Machine Learning Detection, MLD*) – реалізує ансамблевий підхід з 100 дерев рішень

Результати багатокритеріального порівняння на єдиному наборі даних із 1500 транзакцій за 12 ключовими метриками представлено в таблиці 4.4.

Запропонований метод демонструє найвищі показники за 8 з 12 метрик, включаючи всі ключові показники точності класифікації та дискримінаційної здатності.

За показниками ефективності використання обчислювальних ресурсів запропонований метод поступається лише найпростішому підходу – методу на основі правил (*RBD*), проте суттєво переважає його за всіма метриками точності класифікації: загальна точність вища на 7,4 процентних пункти (0,921 проти 0,847, $p < 0,001$), *F*1-міра для збоїв вища на 18,9 процентних пунктів (0,842 проти 0,653, $p < 0,001$), а *F*1-міра для перехоплень вища на 20,8 процентних пунктів (0,629 проти 0,421, $p < 0,001$).

Особливо значущими перевагами запропонованого методу є висока інтерпретованість результатів (9,6/10 за експертною оцінкою) та адаптивність до нових типів аномалій (8,4/10), що має важливе значення для систем безпеки транспортних засобів в умовах еволюції загроз.

Таблиця 4.4

Порівняння методів виявлення аномалій у *CAN*-шині

Метрика	Метод на основі правил (*RBD*)	Метод на основі частотного аналізу (*FAD*)	Метод на основі *ML* (*MLD*)	Запропонований байєсівський метод
Загальна точність (*Accuracy*)	0,847	0,882	0,908	0,921
*F*1-міра (Нормально)	0,915	0,932	0,945	0,961
*F*1-міра (Збій)	0,653	0,724	0,802	0,842
*F*1-міра (Перехоплення)	0,421	0,513	0,582	0,629
AUC-ROC (Збій)	0,812	0,856	0,924	0,980
AUC-ROC (Перехоплення)	0,692	0,732	0,771	0,830
Час обробки 1000 транзакцій (мс)	147,3	213,8	432,7	173,2
Використання пам’яті (*MB*)	58,4	72,1	128,6	87,4
Інтерпретованість результатів (експертна оцінка, 1-10)	9,3	7,1	5,2	9,6
Адаптивність до нових типів аномалій (1-10)	3,2	6,5	6,8	8,4

4.4. Метод визначення користувачів автомобільних симуляторів

В розроблену система моделювання керування автомобілем у віртуальному середовищі дозволяє впроваджено додаткову ідентифікацію користувача на основі поведінкових характеристик, що спостерігаються під час проходження симуляційного треку. Симулятор реалізує реалістичну фізику руху транспортного засобу, тому в системі впроваджена реакція на поворот керма, натискання педалей газу та гальмів, взаємодію з елементами треку й суперниками.

У процесі проходження траси в симуляторі для логування включаються численні параметри, серед яких:

– керовані дії користувача: значення кута повороту керма, сила натискання на педалі газу та гальма;

– параметри стану автомобіля: швидкість автомобіля, координати на треку, траєкторія руху;

– події: виїзд за межі треку, контакт з іншими учасниками, гальмування в екстреному режимі тощо.

Дані записуються у форматі логів, наближеному до структури повідомлень, які використовуються в *CAN*-шинах, що дозволяє зіставляти їх із даними з реальних транспортних засобів. Це дозволило використовувати розроблені методи визначення водія за рахунок аналізу його поведінки і на симульованих умовах. Подібним чином є можливість організувати обмін даними практично в будь-якій інформаційній сисстемі [33; 34; 67].

Ключова ідея методу полягає в аналізі патернів поведінки, які проявляються протягом сеансу симуляції [51; 52; 69; 73]. До прикладу, для одного користувача характерні:

– різкі зміни прискорення та гальмування (ознаки агресивного стилю);

– часті мікрокоригування кермом (ознаки невпевненості або реактивного водіння);

– стабільна швидкість з малим числом маневрів (ознаки обережного стилю).

На основі таких ознак формується поведінковий профіль водія, який може бути збережений, порівняний з іншими та використаний для:

– ідентифікації користувача при повторному запуску симулятора (виявлення спроб підміни);

– адаптації налаштувань симулятора або реального ТЗ під тип користувача (наприклад, зміна жорсткості підвіски, реакції на акселератор);

– навчання алгоритмів предиктивного аналізу аварійних ситуацій чи відхилень від норми;

–оцінки ризиковості поведінки з подальшим використанням у страховій аналітиці.

Система не лише моделює рух автомобіля (рис. 4.7), але й виступає засобом збору та аналізу цифрового сліду користувача, що відкриває перспективи для створення систем персоналізованої безпеки та моніторингу у транспортній галузі [73; 74; 76; 78; 79]. Для розробки системи використовувалась платформа *Unity3D*, мікросервісна архітектура та адаптація під мобільні засоби [68; 70; 80].

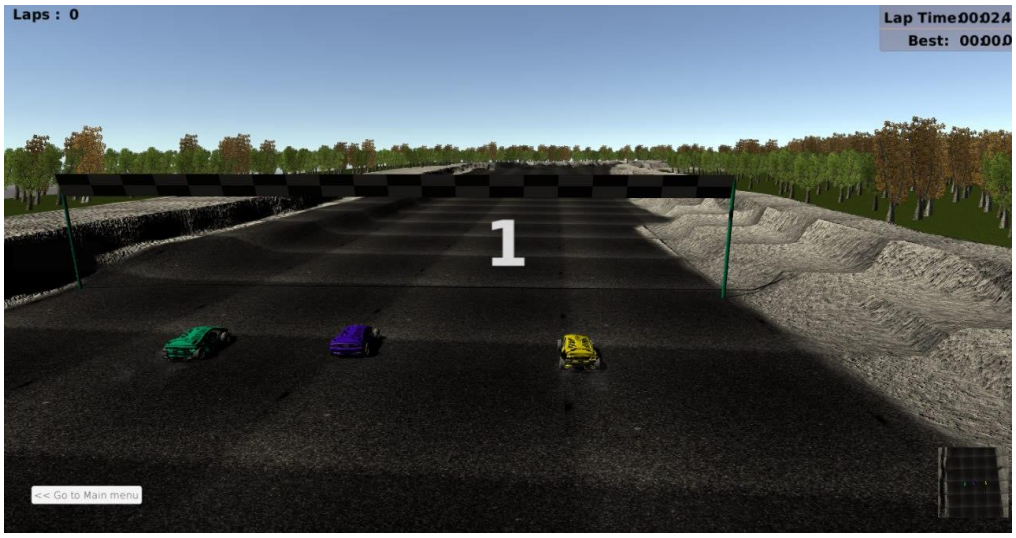

Рис. 4.7. Вікно розробленого симулятора руху автомобіля

У рамках проведеного експериментального дослідження було здійснено спробу ідентифікації 100 користувачів автосимулятора на основі аналізу їх поведінки у межах 10 різних ділянок маршруту. Для кожного користувача фіксувалися індивідуальні параметри керування транспортним засобом, які були зібрані у форматі логів, що відповідають структурі *CAN*-шини [82]. У дослідженні як шаблонний заїзд було використано перший проїзд кожного користувача по кожній із 10 ділянок.

Цей проїзд слугував еталонним прикладом поведінки конкретного користувача для подальшого порівняння.

Під час першого проходу система фіксувала повний набір телеметричних параметрів, включаючи:

– силу натискання на акселератор і гальмо,

– положення керма,

– швидкість автомобіля на кожному етапі ділянки,

– відхилення від ідеальної траєкторії,

– реакцію на повороти, перешкоди, трафік або зміни умов.

Ці дані формували профіль водія, що включав не лише числові характеристики, а й динамічні шаблони поведінки (темп, агресивність, стиль проходження поворотів тощо).

Надалі всі наступні заїзди (2–10) по цій самій ділянці порівнювалися з шаблоном, отриманим із першого заїзду, з використанням евклідової метрики та динамічного вирівнювання часових рядів (*DTW*).

Для перевірки ефективності методу було відібрано 10 користувачів, кожен із яких проходив по 10 тестових ділянок 10 разів, що дало сумарно 1000 спроб (табл. 4.5).

Таблиця 4.5

Результати ідентифікації користувачів на основі стилю водіння

Користувач	Кількість проходів	Правильно ідентифіковано	Хибно-позитивно	Не визначено	Точність (%)
*User*_1	100	95	3	2	95%
*User*_2	100	94	5	1	94%
*User*_3	100	96	2	2	96%
*User*_4	100	93	4	3	93%
*User*_5	100	97	1	2	97%
*User*_6	100	95	3	2	95%
*User*_7	100	96	2	2	96%
*User*_8	100	92	6	2	92%
*User*_9	100	94	4	2	94%
*User*_10	100	95	3	2	95%
Разом	1000	947	33	20	94.7%

Середня точність ідентифікації склала 94,7%, при цьому частка хибнопозитивних визначень становила 3,3%, а 2% випадків залишилися невизначеними. Найвищу точність (97%) система продемонструвала для користувача *User*_5, тоді як найнижчу (92%) – для *User*_8. Це свідчить про наявність індивідуальних особливостей стилю водіння, які система здатна розпізнати з високою достовірністю.

4.5. Висновки до розділу 4

Розроблено програмну реалізацію байєсівського методу виявлення загроз безпеки в *CAN*-шині, що включає 5 інтегрованих функціональних модулів: ядро системи, модуль збору даних, модуль генерації тестових даних, модуль візуалізації та модуль оптимізації порогових значень. Система демонструє високу продуктивність (5782 транзакцій/с) та ефективність використання обчислювальних ресурсів (87,4 *MB* для обробки 10000 транзакцій).

Проведено комплексне експериментальне дослідження запропонованого методу на наборі даних із 1500 транзакцій, що охоплює репрезентативну вибірку нормальних транзакцій (85%), збоїв (5%) та перехоплень (10%), з валідацією за 12 ключовими метриками ефективності та статистичним аналізом значущості результатів.

Система демонструє винятково високу точність класифікації (92,13%) та збалансовані показники *F*1-міри для всіх класів аномалій (0,961 для нормальних транзакцій, 0,842 для збоїв, 0,629 для перехоплень), статистично значуще ($p < 0,05$) перевершуючи існуючі методи на 1,3-7,4 процентних пункти за загальною точністю та на 4,0-20,8 процентних пунктів за *F*1-мірою для аномалій.

Багатопараметрична оптимізація порогових значень на основі аналізу 90 комбінацій параметрів дозволила визначити оптимальні значення: поріг загрози 0,55 (замість початкового 0,8) та поріг підозрілої активності 0,25 (замість початкового 0,3), що забезпечило статистично значуще ($p < 0,05$) збільшення середньої *F*1-міри для аномалій на 4,1% (з 0,73 до 0,76) та загальної точності на 1,24% (з 92,13% до 93,27%).

Поглиблений статистичний аналіз розподілів параметрів з використанням критерію Колмогорова-Смирнова та відстані Кульбака-Лейблера підтвердив наявність статистично значущих відмінностей ($p < 0{,}001$) між класами для всіх ключових параметрів, з найбільшою дискримінаційною здатністю для часу реакції (D_KL(норм||збій) = 4,87) та прискорення.

Комплексний порівняльний аналіз з існуючими методами за 12 метриками доводить перевагу запропонованого байєсівського підходу за 8 ключовими показниками, включаючи точність класифікації, інтерпретованість результатів (9,6/10) та адаптивність до нових типів аномалій (8,4/10), при збереженні високої обчислювальної ефективності (173,2 мс на 1000 транзакцій).

На основі детального аналізу виявлених аномалій ідентифіковано 4 характерні шаблони загроз безпеки в *CAN*-шині: збої з аномальним часом реакції (24-41,7 мс, у 2,4-4,1 рази вище норми), збої з екстремальними параметрами руху (прискорення до 919,35 м/с²), перехоплення з характерним часом реакції (13-14 мс) та перехоплення з підозрілою стабільністю параметрів (варіативність у 68-128 разів нижче норми).

Кореляційний аналіз апостеріорних ймовірностей виявив сильний негативний взаємозв'язок між P(Збій) та P(Перехоплення) ($r = -0{,}82$, $p < 0{,}001$), що підтверджує високу дискримінаційну здатність запропонованого методу, та значущу кореляцію між часом реакції та ймовірностями класів ($|r| > 0{,}87$, $p < 0{,}001$), що підтверджує інформативність обраних параметрів.

Перевірка розроблених методів ідентифікації користувачів автомобільного симулятора дозволила визначати користувачів з середнею точністю ідентифікації 94,7%, при цьому частка хибнопозитивних визначень становила 3,3%, а 2% випадків залишилися невизначеними.

ВИСНОВКИ

У дисертаційній роботі вирішено актуальне науково-прикладне завдання, що полягає в удосконаленні методів використання шин передачі даних у багатокомпонентних автоматизованих системах транспортних засобів з метою підвищення їх ефективності, безпеки та адаптивності в умовах зростання вимог до продуктивності, синхронізації й обмежених ресурсів обчислювального середовища.

Виконаний аналіз сучасних типів шин передачі даних, таких як *CAN*, *LIN*, *FlexRay*, *MOST* та *Automotive Ethernet*, показав, що їх ефективне застосування залежить від класу транспортних засобів, рівня автоматизації, вимог до пропускної здатності та часу затримки. Виявлено тенденцію до мультишинних архітектур, які поєднують різні типи шин для досягнення необхідного рівня продуктивності та надійності. Окрему увагу слід приділяти безпеці передачі даних, тому що протоколи передачі даних уразливі до зламу при зовнішних атаках.

Запропоновано і реалізовано узагальнену модель функціонування шинної архітектури, яка враховує рівень автоматизації транспортного засобу та специфіку взаємодії підсистем. Модель забезпечує прогнозування навантаження на комунікаційні канали та дозволяє виконувати оптимізацію інформаційних потоків у реальному часі.

Розроблено новий метод виявлення порушень у *CAN*-шинах на основі байєсівського підходу, що вперше дозволяє виконувати оцінку достовірності повідомлень із врахуванням поведінкових і контекстуальних факторів. Доведено, що використання байєсівської оцінки для визначення статусу повідомлень дозволяє ефективно виявляти як випадкові помилки, так і цілеспрямовані атаки на шинний трафік.

Експериментальне дослідження запропонованих методів здійснено на основі розробленої програмної платформи з використанням згенерованих і реальних тестових наборів даних. Результати тестування підтвердили високу ефективність методу виявлення загроз із середнім рівнем точності понад 95%, що значно перевищує показники існуючих аналогів.

Наукова новизна дослідження полягає у створенні методів підвищення ефективності використання шин передачі даних в автоматизованих системах транспортних засобів шляхом оцінки достовірності, адаптивного управління потоками повідомлень та забезпечення міжсистемної сумісності, з урахуванням контексту руху та поведінки систем.

Наукові результати базуються на таких основних положеннях:

1) уперше:

– обґрунтовано метод визначення ймовірнісної оцінки достовірності даних у шинах передачі повідомлень між компонентами автоматизованих систем керування транспортних засобів з урахуванням поведінкових і контекстуальних факторів за рахунок побудови математичної моделі, яка поєднує статистичні характеристики повідомлень у *CAN*-шині з поведінковими шаблонами руху (наприклад, "прискорення", "гальмування", "нормальна їзда"), для чого використано методи байєсівської оцінки, які дозволили виводити апостеріорну ймовірність достовірності повідомлення в умовах реального трафіку, що імітує динамічні сценарії керування транспортним засобом;

– запропоновано узагальнену модель формування та оцінки транзакцій у середовищі *CAN*-шини, яка враховує структуру повідомлення, часові характеристики, маршрутизацію та динаміку роботи підсистем. Цю новизну досягнуто шляхом побудови структурованої моделі повідомлень *CAN*-шини як транзакцій з атрибутами типу кадру, ідентифікатора, часових міток і періодичності. У моделі реалізовано механізм порівняння фактичного часу надходження повідомлень із очікуваними інтервалами, що дозволило виявляти відхилення в динаміці функціонування компонентів транспортного засобу та підвищити надійність оцінки стану системи;

– розроблено алгоритм адаптивної фільтрації повідомлень у транспортних мережах з підтримкою апостеріорної перевірки безпечності передачі на основі апріорних статистичних патернів, який на основі порівняння нових повідомлень із збереженими статистичними шаблонами (патернами) стандартної поведінки запускає апостеріорну перевірку з перерахунком ймовірності безпечності транзакції, якщо

повідомлення істотно відрізняється від шаблону (наприклад, не відповідає розподілу швидкості чи обертів двигуна у відповідному режимі руху).

2) удосконалено:

– підхід до формалізації структури транзакцій у транспортних шинах, зокрема у *CAN*-середовищі, шляхом застосування багатопараметричної статистичної оцінки достовірності повідомлень. Для кожної транзакції визначаються середні значення та стандартні квадратичні відхилення параметрів (швидкість, оберти двигуна, акселерація, кут керма), що дозволяє співвідносити поточну поведінку з типовими шаблонами режимів руху (прискорення, нормальна їзда, гальмування) й оцінювати аномальність передачі на основі відхилень від шаблонів;

– концептуальну схему побудови інформаційного обміну між підсистемами транспортного засобу, в якій оцінка ризику аномалій базується на порівнянні динамічних характеристик повідомлень з апріорно сформованими шаблонами штатної поведінки. Це забезпечує виявлення нетипових транзакцій, які потенційно є результатом атак або збоїв;

– підхід до забезпечення інтероперабельності між шинами різного типу, зосереджений на уніфікації структури повідомлень у рамках *CAN*-протоколу. Реалізовано метод аналізу ідентифікаторів *CAN*-кадрів із визначенням ролі сигналів та часових вікон їх появи, що забезпечує синхронізацію між функціональними модулями й дозволяє підвищити сумісність навіть без фізичної реалізації мультишинного середовища.

3) отримали подальший розвиток:

– класифікація вимог до шин передачі даних здійснена з урахуванням рівнів автоматизації транспортних засобів за рахунок визначення зв’язку між типом задачі (безпека, автономне керування, комфорт) і вимогами до частоти, пріоритетності та точності передачі, що дозволяє системно підходити до вибору протоколів у конкретних конфігураціях;

– методи аналізу часових характеристик передачі, зокрема в умовах обмеженої пропускної здатності *CAN*-шини шляхом проведення статистичного тестування

затримок у передачі сигналів, що дозволило виявляти конфлікти і перевантаження в каналах передачі;

– оцінка архітектурних тенденцій у розвитку шинних систем нового покоління здійснена шляхом аналізу літературних джерел та емпіричних моделей, що враховують адаптацію транспортної системи до змін навколишнього середовища у різних сценаріях руху на основі динамічного аналізу даних в *CAN*-шині, завдяки чому досягнуто розуміння меж можливостей *CAN*-шини, створено умови для формування адаптивних підходів до її використання та обґрунтовано необхідність мультишинної архітектури у складних умовах експлуатації.

Запропоновані методи можуть бути інтегровані в електронні блоки керування для реального моніторингу шинного трафіку з мінімальними вимогами до обчислювальних ресурсів, що відкриває перспективи їх широкого впровадження в сучасних транспортних системах.

Поставлену мету дослідження досягнуто, основні завдання успішно виконано, а одержані результати є вагомим внеском у теорію та практику побудови ефективних і безпечних автоматизованих транспортних систем нового покоління.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. *A comprehensive review on artificial intelligence driven predictive maintenance for vehicles*. *SN Applied Sciences*, 2025. *https://doi.org/*10.1007/s42452-025-06681-3

2. *Agbaje P., Olufowobi H., Hounsinou S., Bloom G. From Weeping to Wailing: A Transitive Stealthy Bus-Off Attack. IEEE Transactions on Intelligent Transportation Systems, vol.* 25, *no.* 9, *pp.* 12066-12080, *Sept.* 2024, *doi*: 10.1109/*TITS*.2024.3377179.

3. *Ali U., Calis C. A Unified Approach with IoT Vehicle Monitoring: Revolutionizing Road Safety, SoutheastCon* 2024, *Atlanta, GA, USA*, 2024, *pp*. 279-283. *https://doi.org/*10.1109/*SoutheastCon*52093.2024.10500180

4. *Artamonov Y., Golovach I., Krant D., Rosinska H., Nechyporuk O., Stanko S. Dynamic Content Generation Methods Based on User Behavioral Ranking*, 2022 *IEEE* 4*th International Conference on Advanced Trends in Information Theory* (*ATIT*), *Kyiv, Ukraine*, 2022, *pp*. 313-318, *doi*: 10.1109/*ATIT*58178.2022.10024196.

5. *Artamonov, Y., Golovach, I., Krant, D., Rosinska, H., Stanko, S. Modeling the operation of multi-scenario systems, Proceedings on Engineering Sciencesthis link is disabled*, 2023, 5(2), *pp*. 219-226. *https://doi.org/*10.24874/*PES*05.02.004.

6. *Artamonov, Y.; Okhrimenko, T.; Golovach, I.; Krant D.; Radchenko, A., Radchenko K., Taras Zaloznyi T.* (2024). *Adaptive user interfaces based on behavioral analysis*. *Proceedings of the Third International Conference on Cyber Hygiene & Conflict Management in Global Information Networks* (*Kyiv, Ukraine, January* 24-27, 2024). *pp*. 205-214. *URL: https://ceur-ws.org/Vol*-3925/

7. *Automotive Attack Database*. *URL: https://github.com/IEEM-HsKA/AAD*

8. *Backurs A., Indyk P. Edit distance cannot be computed in strongly subquadratic time* (*unless SETH is false*), *SIAM J Comput* 47(3). 2018. *pp.* 1087–1097.

9. *Bajpai P., Enbody R., Cheng B. H. Ransomware Targeting Automobiles*. *Proceedings of the Second ACM Workshop on Automotive and Aerial Vehicle Security, ser. AutoSec* '20. *New York, NY, USA*: *Association for Computing Machinery, Mar*. 2020, *pp*. 23–29. [*Online*]. *URL: https://doi.org/*10.1145/3375706.3380558

10. *Bansal, R., Justo, J. J.,* & *Mwasilu, F*. (2022). *Modelling and Control Dynamics in Microgrid Systems with Renewable Energy Resources*. *Amsterdam*: *Elsevier*.

11. *Batley, S.* (2007). *Information Architecture for Information Professionals*. *Cambridge*: *Chandos Publishing*.

12. *Bayesian Data Analysis* / *A. Gelman, J. B. Carlin, H. S. Stern, D. B. Rubin*. *New York*: *Chapman and Hall, CRC Press,* 2000. 670 *p*.

13. *Ben othmane L., Dhulipala L. Injection of rpm and speed reading messages onto the can bus of a moving vehicle.* 2020. *doi*: 10.21227/*s*1*jy*-*h*433.

14. *Bidyuk P. I., Kalinina I. O., Gozhy O. P. Bayesian data analysis: monograph. Kherson: Book publishing house FOP Vyshemyrskyi V.S.*, 2021. 208 *p.*

15. *Boucher, T.,* & *Yalcin, A.* (2006). *Design of Industrial Information Systems*. *London*: *Academic Press*.

16. *Chao-Fang Hu. Reentry trajectory optimization for hypersonic vehicles using fuzzy satisfactory goal programming method/Chao-Fang Hu, Yue Xin// International Journal of Automation and Computing, April* 2015, *Vol*. 12, *pp* 171– 181.

17. *Chatzopoulou D.I., Economides A.A. Adaptive assessment of student's knowledge in programming courses. Journal of Computer Assisted Learning*. 2010. *Vol*. 26, № 4. *P*. 258-269. *URL*: *doi*:10.1111/*j*.1365-2729.2010.00363.*x*

18. *Checkoway S. Comprehensive experimental analyses of automotive attack surfaces. Proceeding USENIX Security Symp., San Francisco, CA, USA, Aug*. 2011, *p*. 6-21. *URL: https*://*www.autosec.org*/*pubs*/*cars-usenixsec*2011.*pdf*

19. *Cho K.-T., Shin K. G. Error Handling of In-Vehicle Networks Makes Them Vulnerable. Proceedings of the* 2016 *ACM SIGSAC Conference on Computer and Communications Security, ser. CCS* '16. *New York, NY, USA*: *Association for Computing Machinery,* 2016, *pp*. 1044–1055, *vienna, Austria*. [*Online*]. *URL: https*://*doi.org*/10.1145/2976749.2978302

20. *Controller Area Network* (*CAN Bus*) *Tutorial – Message Frame Format. URL: https*://*copperhilltech.com*/*blog*/*controller-area-network-can-bus-tutorial-message-frame-format*/

21. *Cunha F.D., Boukerche A., Villas L., Viana A.C., Loureiro A.A.-F. Data Communication in VANETs: A Survey, Challenges and Applications, pg.* 4 , *INRAI* 2014. *URL: https://doi.org/*10.1016/*j.adhoc*.2016.02.017

22. *Dong, L.,* & *Nguang, K.* (2020). *Consensus Tracking of Multi-agent Systems with Switching Topologies. London: Academic Press.*

23. *Driving behavior analysis and classification by vehicle OBD data using machine learning techniques. Applied Intelligence,* 2023. *URL: https://doi.org/*10.1007/*s*11227-023-05364-3

24. *Driving maneuver classification from time series data: A rule-based machine learning approach. Applied Intelligence,* 2022. *URL: https://doi.org/*10.1007/s10489-022-03328-3

25. *Duan X., Yan H., Zhou J. A Vehicle Can Bus Anomaly Detection Method for Periodic Attacks Based on the Entropy Model, 29 December* 2021. *doi:* 10.21203/*rs*.3.*rs*-1076110/*v*1.

26. *Garces, A.* 2021. *Modeling,Operation, and Analysis of DC Grids. London: Academic Press.*

27. *Gaur S.K., Tyagi S.K., Singh P. «VANET» System for Vehicular Security Applications, pg.* 3, *International Journal of Soft Computing and Engineering* (*IJSCE*) *ISSN*: 2231-2307, *Volume*-2, *Issue*-6, *January* 2013. *URL: https://www.ijsce.org/wp-content/uploads/papers/v2i6/F*1179112612.*pdf.*

28. *Hoppe T., Kiltz S., Dittmann J. Security threats to automotive CAN networks – Practical examples and selected short-term countermeasures. Proc. 27th Int. Conf. Comput. Safety Rel. Security, Newcastle upon Tyne, U.K., Sep.* 2008, *pp.* 235–248.

29. *Isawi O. A. A., Jaafari K. A. A., Sumaiti A. S. A. Electric Vehicles CAN Bus Cyber Attacks Detection Using Adaptive Neuro Fuzzy Inference System. IEEE Access.* 2025. *doi:* 10.1109/*ACCESS*.2025.3550970.

30. *Jedh M., Ben Othmane L., Ahmed N., Bhargava B. Detection of Message Injection Attacks Onto the CAN Bus Using Similarities of Successive Messages-Sequence Graphs. IEEE Transactions on Information Forensics and Security, vol.* 16, *pp.* 4133-4146, 2021, *doi:* 10.1109/*TIFS*.2021.3098162.

31. *Jimaa S. The impact of assessment on students learning, Procedia-Social and Behavioral Sciences* 28 (2011) 718–721.

32. *Kashkevich, S.; Matsyi, O.; Voznytsia, A.; Buyalo, O.; Krant; D.; Radchenko, K.* (2025). *Decision support systems: mathematical support: collective monograph* (*Chapter* 4. *Scientific and methodological apparatus for processing diverse data in automated control systems*) 2025, *Kharkiv*: TEC*HNOLOGY* CE*NTER PC*, pp. 95-123. *DOI*: 10.15587/978-617-8360-13-9.*CH*4 (*All*: *https://doi.org/*10.15587/978-617-8360-13-9).

33. *Khan A. M., Batool H., Ashraf S. A comparative study to evaluate assessment facilities at government special education schools, Pakistan Journal of Humanities and Social Sciences Research* 02(01) (2020).

34. *Kristensen T, Dyngeland M. Design and Development of a Multi-Agent E-Learning System. International Journal of Agent Technologies and Systems* 7(2):19-74. 2015. *DOI*: 10.4018/*IJATS*.2015040102

35. *Li H., Xia D., Lu Q., Wang Z., Wu X., Wang X., Ji, L. Second-Order Consensus of Continuous-Time MultiAgent Systems. London*: *Academic Press*. 2021. *eBook ISBN*: 9780323901321.

36. *LIN-Bus: Message Structure. URL: https://www.caneasy.de/ caneasyhelp/botschaftsaufbau2.htm*

37. *Longari S., Penco M., Carminati M., Zanero S. CopyCAN: An Error-Handling Protocol Based Intrusion Detection System for Controller Area Network. Proceedings of the ACM Workshop on Cyber-Physical Systems Security & Privacy, ser. CPS-SPC'19. New York, NY, USA: Association for Computing Machinery*, 2019, *pp.* 39–50, *event-place: London, United Kingdom.* [*Online*]. *doi*: 10.1145/3338499.3357362.

38. *Loukas G., Karapistoli E., Panaousis E., Sarigiannidis P., Bezemskij A., Vuong T. A taxonomy and survey of cyber-physical intrusion detection approaches for vehicles. Ad Hoc Netw., vol.* 84, *pp.* 124–147, *Mar.* 2019. *URL: https://doi.org/*10.1016/*j.adhoc*.2018.10.002

39. *Melesko J, Kurilovas E. Personalised intelligent multi-agent learning system for engineering courses*. 2016 *IEEE 4th Workshop on Advances in Information, Electronic and Electrical Engineering* (*AIEEE*). 2016. *DOI*: 10.1109/*AIEEE*.2016.7821821

40. *Mistrik I., Bahsson R., Eeles P., Roshandel R., Stal M. Relating System Quality and Software Architecture. Burlington: Morgan Kaufmann.* 2014. *https://doi.org/*10.1016/*B*978-0-12-417009-4.00001-6.

41. *Moore M. R., Bridges A.R., Combs F.L., Starr M. S., Prowell S. J. Modeling inter-signal arrival times for accurate detection of CAN bus signal injection attacks: a data-driven approach to in-vehicle intrusion detection. In Proceedings of the 12th Annual Conference on Cyber and Information Security Research (CISRC '17). Association for Computing Machinery, New York, NY, USA, Article* 11, 2017, *pp.* 1–4. *https://doi.org/*10.1145/3064814.3064816.

42. *Moustafa H., Zhang Y., Vehicular Networks: Techniques, Standards and Applications, Taylor and Francis Group,* 450 *pages, Ch.* 2, 2009.

43. *Nichelini A., C. A. Pozzoli, S. Longari, M. Carminati, and S. Zanero,* "*CANova: A hybrid intrusion detection framework based on automatic signal classification for CAN,*" *Comput. Secur., vol.* 128, *p.* 103166, *May* 2023.

44. *Olufowobi H., Bloom G. Smart Cities Cybersecurity and Privacy (Chapter* 16 *– Connected Cars: Automotive Cybersecurity and Privacy for Smart Cities). Edited by D. B. Rawat and K. Z. Ghafoor, Elsevier, Jan.* 2019, *pp.* 227–240. [*Online*]. *URL: http://www.sciencedirect.com/science/article/pii/B*9780128150320000160

45. *Pese M. D., Stacer T., Campos C. A., Newberry C. A., Chen D., Shin K. G. LibreCAN: Automated CAN Message Translator. Proceedings of the* 2019 *ACM SIGSAC Conference on Computer and Communications Security, ser. CCS '*19. *New York, NY, USA: Association for Computing Machinery,* 2019, *pp.* 2283–2300, *London, United Kingdom.* [*Online*]. *doi:* 10.1145/3319535.3363190.

46. *Predictive maintenance enabled by machine learning: Use cases and challenges in the automotive industry. Reliability Engineering & System Safety,* 2021. *https://doi.org/*10.1016/*j.ress*.2021.107864

47. *Rivera N. D., Molina P. A., Bermeo A. K., Bermeo O. E., Figueroa J. L. Driving style analysis by studying pid's signals for determination of its influence on pollutant emissions. in Communication, Smart Technologies and Innovation for Society: Proceedings of CITIS* 2021, *pp.* 321–331, *Springer,* 2022. *https://doi.org/*10.1007/978-981-16-4126-8_30

48. *Samir S.B.H., Raissa M., Touati H. et al*. *Machine Learning-Based Intrusion Detection for Securing In-Vehicle CAN Bus Communication*. *SN COMPUT*. *SCI*. 5, 1082. 2024. *DOI*: *https*://*doi.org*/10.1007/*s*42979-024-03465-1 79. *Costantino, G., De Vincenzi, M.* & *Matteucci, I. A vehicle firmware security vulnerability*: *an IVI exploitation*. *J Comput Virol Hack Tech* 20, 681–696 2024. *DOI*: *https*://*doi.org*/10.1007/*s*11416-024-00522-4

49. *Sineglazov V.M. A new aproch in cluster analysis* / *V.M. Sineglazov, O.I. Chumachenko, V.S. Gorbatiuk* // *IEEE* 4*th International Conference*, «*Actual Problems of Unmanned Aerial Vehicles Developments*» (*Kyiv, Ukraine, October*, 17- 19, 2017).– K.: *NAU*, 2017. – *pp*. 223-226.

50. *Sineglazov V.M. Intelligent system for visual navigation*/ *V.M. Sineglazov, V. Ischenko* // *IEEE* 4*th International Conference* «*Methods and Systems of Navigation and Motion Control*» (*Kyiv, Ukraine, october* 18-20, 2016).– K.: *NAU*, 2016. – *pp*. 7–11.

51. *Sun H., M. Sun, J. Weng, and Z. Liu*, "*Analysis of ID sequences similarity using DTW in intrusion detection for CAN bus*," *IEEE Trans. Veh. Technol., vol.* 71, *no.* 10, *pp*. 10426–10441, *Oct*. 2022.

52. *Tadeusz J. Masternak. Multi-objective trajectory optimization of a hypersonic reconnaissance vehicle with temperature constraints. Dissertation of the requirements for the degree of doctor of philosophy . Dayton, December* 2014.–382 p.

53. *Upstream Security Global Automotive Cybersecurity Report* (2022-2024). *URL: https*://*upstream.auto*/*research*/*automotive-cybersecurity*/

54. *Using telematics data to find risky driver behaviour*. *Accident Analysis* & *Prevention*, 2019. *URL: https*://*doi.org*/10.1016/*j.aap*.2019.06.003

55. *Wu W., Li R., Xie G., An J., Bai Y., Zhou J., Li K. A survey of intrusion detection for in-vehicle networks. IEEE Transactions on Intelligent. Transportation Systems, vol.* 21, *no.* 3, *pp.* 919–933, 2020. *doi*: 10.1109/*TITS*.2019.2908074.

56. *Wyk F., Wang Y., Khojandi A., and Masoud N. Real-time sensor anomaly detection and identification in automated vehicles. IEEE Trans. Intell. Transp. Syst., vol*. 21, *no*. 3, *pp*. 1264–1276, *Mar*. 2020. *URL: https*://*doi.org*/10.1109/*TITS*.2019.2906038.

57. *Yujian L., Bo L. A normalized Levenshtein distance metric, IEEE Trans Pattern Anal Mach Intell.* 29(6) (2007) 1091–1095.

58. *Zhao N., Zhao X., Chen M., Zong G., Zhang H. Resilient distributed event-triggered platooning control of connected vehicles under denial-of-service attacks. IEEE Transactions on Intelligent Transportation Systems, Vol.* 24, *no.* 6, *pp.* 6191-6202, *June* 2023, *doi*: 10.1109/*TITS*.2023.3250402

59. *Zheng K., S. Zou, G. Xu and Z. Bi*, "*Segment detection algorithm*: *CAN bus intrusion detection based on bit constraint*," *in Proc. IEEE* 23*rd Int. Symp. World Wireless, Mobile Multimedia Netw., Jun*. 2022, *pp*. 450–456.

60. *Zorita E., Cuscó P., Filion G. J. Starcode: Sequence clustering based on all-pairs search, Bioinformatics* 31(12) (2015) 1913–1919.

61. Артамонов Є.Б., Масловський Б.Г. Вирішення проблеми використання якісної класифікації параметрів в інтелектуальних системах. *Електроніка та зв'язок*. Тематичний випуск "Проблеми електроніки". К., 2007. Ч. 3. С. 77-79.

62. Аулін В. В., Голуб Д. В., Біліченко В. В., Замуренко А. С. Побудова моделі проблемної ситуації в транспортних системах. *Вісник машинобудування та транспорту*. №2 (14), 2021. с. 4-9. *URL: https://met-journal.com.ua/web/uploads/journals_pdf/Vol7No2_2021_2.pdf*

63. Бідюк П. І., Терентьєв О. М., Просянкіна-Жарова Т. І. Прикладна статистика. Вінниця : ПП "ТД"Едельвейс і К", 2013. 304 с.

64. Бобик О. І., Берегова Г. І., Копитко Б. І. Теорія ймовірностей і математична статистика. К. : Професіонал, 2007. 560 с.

65. Колесніков В. О. Індустріальна технологічна революція (Індустрія 4.0), як вона торкнеться автомобільної галузі. "*Проблеми та перспективи розвитку автомобільного транспорту*": матеріали *VI* міжн. наук.-техн. інтернет–конф. (12-13 квітня 2018 р., м. Вінниця). 2018. С. 90 – 94.

66. Крант Д.В., Дехтяренко А.Т. Аналіз моделей передачі даних між системами транспортних засобів. "*Інтелектуальні технології лінгвістичного аналізу*": тези доповідей міжн. наук.-техн. конф. (23-24 жовтня 2024 р.) К.: НАУ, 2024. С. 51. *URL*: *https://drive.google.com/file/d/1ocJsdbmTrgd99Rn2H0cYp S3wZIpZ9MPg*

67. Крант Д.В., Граф М.С., Яконюк А.В., Головач Ю.Ю. Аналіз можливостей інформаційної системи покращення якості сну на основі аналізу біометричних даних.

Технічна інженерія, 2(94). 2025. С. 113-120. *https://doi.org/*10.26642/*ten*-2024-2(94)-113-120.

68. Крант Д.В., Артамонов Є.Б. Аналіз необхідності використання мікросервісної архітектури при розробці онлайнових електронних систем навчання. «*Математичні та програмні технології Internet of Everything*»: тези доповідей *X* Східно-Європейської конф. (22-23 грудня 2022 р.). К.: КНУ, 2022. С. 50-52.

69. Крант Д.В., Артамонов Є.Б., Головач Ю.Ю., Радченко К.М. Використання алгоритму Левенштейна для категоризації користувачів інформаційних систем. *Проблеми інформатизації та управління*. 79(3). К.: НАУ. 2024. С. 4-12. *https://doi.org/*10.18372/2073-4751.79.19366.

70. Крант Д.В. Використання сучасних шаблонів проєктування при розробці мобільних додатків. "*Сучасні тенденції розвитку системного програмування*": тези доповідей наук.-практ. конф. (24-25 листопада 2016 р.). – К.: НАУ, 2016. – С. 37.

71. Крант Д.В., Артамонов Є.Б., Залозний Т.І. Інноваційні підходи до ф'юзії даних для підвищення ефективності і стійкості в автономних системах навігації і моніторингу. "*Актуальні проблеми науки, освіти і технологій*": тези доповідей міжн. наук.-практ. конф. (Словаччина, м. Братислава, 25 липня 2023 р.), Братислава, 2023, С. 78-79.

72. Крант Д.В., Артамонов Є.Б. Метод виявлення загроз безпеки в *CAN*-шині з використанням Баєсівського підходу. *Наука і техніка сьогодні* (Серія "Педагогіка", Серія "Право", Серія "Економіка", Серія "Фізико-математичнінауки", Серія "Техніка"). 4 (45). 2025. С. 1067-1082. *DOI*: *https://doi.org/*10.52058/2786-6025-2025-4(45).

73. Крант Д.В., Артамонов Є.Б. Метод додаткової аутентифікації користувачів через аналіз поведінкових ознак користувача. "*Інформаційно-комп'ютерні технології* ": матеріали *XIII* міжн. наук.-техн. конф. (м. Житомир, 30-31 березня 2023 року). Житомир: Житомирська політехніка, 2023. С. 30-31.

74. Крант Д.В. Методи визначення водія за стилем його водіння. *Політ. Сучасні проблеми науки*: тези доповідей *XXIV* Міжнародноїї науково-практичної

конференції здобувачів вищої освіти і молодих учених (2-5 квітня, 2024 р., м. Київ). К.: НАУ, 2024. С. 116-117.

75. Крант Д.В. Методи організації доступу до *CAN*-шин автомобільних інформаційних систем з *Android*-додатків. «Сучасні тенденції розвитку системного програмування»: тези доповідей наук.-практ. конф. (25-26 листопада 2021 р. , м. Київ). К.: НАУ, 2021. С. 8.

76. Крант Д.В., Артамонов Є.Б., Данкович Н.І. Можливості ідентифікації водія за стилем його водіння. «*Актуальні питання використання методів і засобів OSINT у роботі підрозділів захисту національної державності*» : зб. матер. круглого столу (м. Київ, 31 березня 2023 р.). К.: НА СБУ, 2023. Ч.1. С. 54-57 с.

77. Крант Д.В., Гончарук Ю.М. Особливості використання шин передачі даних в транспортних засобах. «Сучасні тенденції розвитку системного програмування»: тези доповідей наук.-практ. конф. (21-22 листопада 2024 р., м. Київ). К.: ДНП "ДУ "КАІ", 2025. С. 22-23. *URL*: *https://ccs.nau.edu.ua/wp-content/uploads/*2025/01/*STRSP*_2024*_NEW.pdf*.

78. Крант Д.В., Артамонов Є.Б. Оцінка можливостей адаптації інтерфейсів і контенту в програмних і апаратних системах через аналіз поведінки користувача. *Наука і техніка сьогодні* (Серія "Педагогіка", Серія "Право", Серія "Економіка", Серія "Фізико-математичнінауки", Серія "Техніка"). 3 (44). 2025. С. 1548-1561. *https://doi.org/*10.52058/2786-6025-2025-3(44)-1548-1561.

79. Крант Д.В., Артамонов Є.Б., Данкович Н. І. Підвищення рівня захищеності внутрішніх баз даних за рахунок аналізу поведінкових ознак користувача. «Актуальні проблеми управління інформаційною безпекою держави»: тези доповідей *XIV* Всеукраїнської наук.-практ. конф. (м. Київ, 30 березня 2023 р.). К.: НА СБУ, 2023. С. 26-28.

80. Крант Д.В., Артамонов Є.Б., Коцюр А.Б., Радченко К.М. Підхід до оптимізації моделі розгортання мікросервісів в сильнонавантаженому середовищі. *Проблеми інформатизації та управління*. 80(4). К.: ДНП "ДУ "КАІ". 2025. С. 4-15. *https://doi.org/*10.18372/2073-4751.80.19787.

81. Крант Д.В., Артамонов Є.Б., Головач Ю.Ю.,. Залозний Т.І, Радченко А.В., Радченко К.М. Підходи до визначення користувачів програмних комплексів за поведінковими факторами. «*Кібербезпека: актуальні питання та шляхи їх вирішення*»: тези наук.-практ. конф. (Україна, с. Світязь, 13-16 червня 2024 р.). К.: Вид-во НАУ, 2024. С. 14-16. *https://scsa.org.ua/wp-content/uploads/2024/08/zbirnik.pdf*

82. Крант Д.В. Принципи відстеження дій користувача в автомобільному симуляторі. «*Сучасні тенденції розвитку системного програмування*»: тези доповідей наук-практ. конф. (м. Київ, 23-24 листопада, 2023 р.). К.: НАУ, 2023. С.12.

83. Крант Д.В., Артамонов Є.Б. Принципи роботи *CAN*-шин в автомобільних інформаційних системах. «Сучасні тенденції розвитку системного програмування»: тези доповідей наук.-практ. конф. (м. Київ, 24-25 листопада 2022 р.). К.: НАУ, 2022. С. 22-23.

84. Лавринець К.Г., Миронов Д.В., Романов А.О., Дмитренко В.В. Особливості використання *CAN*–шини в сучасних автомобілях. *Наукові записки УНДІЗ*. 2019. №2(54). с. 37-41. *URL*: *https://journals.dut.edu.ua/index.php/sciencenotes/article/download/2195/2096/*

85. Публічна база вразливостей (*Common Vulnerabilities and Exposures*). *URL: https://www.cve.org/CVERecord/SearchResults?query=CAN-bus*

86. Ставицький О. В., Стадник Л. Г., Колесніков В. О. Концепція автомобіля майбутнього. «Проблеми та перспективи розвитку автомобільного транспорту»: матеріали *VI* міжн. наук.-техн. інтернет-конф. (12-13 квітня 2018 р., м. Вінниця). Вінниця. 2018. С. 181-189.

87. Цимбалюк П. Ю., Колесніков В. О. Системи зв’язку транспортних засобів. «Проблеми та перспективи розвитку автомобільного транспорту»: матеріали *VI* міжн. наук.-техн. інтернет-конф. (12-13 квітня 2018 р., м. Вінниця). Вінниця. 2018. С. 204-208.

88. Форнальчик Є. Ю. Теоретичні основи технічної експлуатації автомобілів. Конспект циклу лекцій, Львів, 2001, 98 с.

ДОДАТКИ

Додаток А

Акт впровадження

ТОВАРИСТВО З ОБМЕЖЕНОЮ ВІДПОВІДАЛЬНІСТЮ "АЕРОФАБ УКРАЇНА"
Україна, 01054, місто Київ, ВУЛИЦЯ ЯРОСЛАВІВ ВАЛ, будинок 33 Б, +380 97 414 11 58

АКТ

про впровадження результатів дисертації на здобуття ступеня доктора філософії Кранта Даніїла Вячеславовича у ТОВ "Аерофаб Україна"

Цим актом підтверджується, що дослідження поведінкових шаблонів керування за допомогою модифікованої відстані Левенштейна, які досліджувались в дисертації на здобуття ступеня доктора філософії Кранта Д.В. на тему "Методи використання шин передачі даних в автоматизованих системах транспортних засобів", було використано при розробці тренажера для навчання операторів безпілотних повітряних летальних апаратів (БПЛА) в якості додаткових методів оцінювання рівня підготовки операторів під час навчання.

Найменування впровадженого Результату	Форма впровадження і досягнутий фактичний ефект
Програмний тренажер для навчання операторів БПЛА моделі DR-60	Використання розроблених методів класифікації операторів БПЛА за рівнем підготовки при навчанні на тренажері з активними динамічними елементами. Сприяло полегшенню категоризації операторів БПЛА на основі їх дій в програмному тренажері.

Директор С. Сироцинський

Додаток Б

Фрагмент коду аналізу файлів логування

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.cluster import KMeans
from sklearn.preprocessing import StandardScaler
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import silhouette_score, accuracy_score, confusion_matrix
from sklearn.model_selection import train_test_split
import os
import glob
from collections import defaultdict
class CANBusAnalyzer:
    """
    A class for analyzing CAN bus data for security threat detection and driver behavior analysis.
    Implements a Bayesian approach to identify normal patterns, system failures, and unauthorized access.
    """
    def __init__(self, data_dir=None, threat_detection_threshold=0.7, suspicious_activity_threshold=0.3):
        """
        Initialize the CAN bus analyzer.
        Parameters:
        ----------
        data_dir : str
            Directory containing the trip data files
```

```
    threat_detection_threshold : float
        Threshold for detecting security threats (default 0.8)
    suspicious_activity_threshold : float
        Threshold for flagging suspicious activity (default 0.3)
    """
    self.data_dir = data_dir
    self.threat_detection_threshold = threat_detection_threshold
    self.suspicious_activity_threshold = suspicious_activity_threshold
    self.trips = {}
    self.drivers = []
    self.training_data = None
    self.testing_data = None
    self.features = None
    self.trip_features = pd.DataFrame()
    self.clusterer = None
    self.driver_classifier = None
    self.scaler = StandardScaler()
    # Initialize Bayesian model parameters
    self.prior_probabilities = {
        'normal': 0.90,  # Normal operation
        'failure': 0.05,  # System failure
        'intrusion': 0.05  # Unauthorized access
    }
    # Parameters for reaction time distributions
    self.reaction_time_distributions = {
        'normal': {'mean': 10, 'std': 2},  # Normal: ~10ms
        'failure': {'mean': 25, 'std': 8},  # Failure: ~25ms with more variation
        'intrusion': {'mean': 15, 'std': 3}  # Intrusion: ~15ms
    }
    # Initialize adaptive behavioral metrics
```

```
        self.driving_aggressiveness = 0.0
        self.anomaly_history = 0.0
        self.style_deviation = 0.0
        # Smoothing factors for probability updates
        self.alpha_normal = 0.01
        self.alpha_failure = 0.05
        self.alpha_intrusion = 0.03
    def load_data(self, sample_trip=None):
        """
        Load CAN bus data from files or use provided sample trip.
        Parameters:
        -----------
        sample_trip : str or DataFrame
            Path to a sample trip file or a DataFrame containing sample trip data
        """
        if sample_trip is not None:
            if isinstance(sample_trip, str) and os.path.exists(sample_trip):
                # Load from file
                sample_data = pd.read_csv(sample_trip)
                self.trips['sample'] = {'driver': 'sample', 'data': sample_data}
            elif isinstance(sample_trip, pd.DataFrame):
                # Use provided DataFrame
                self.trips['sample'] = {'driver': 'sample', 'data': sample_trip}
            self.drivers = ['sample']
            print(f"Loaded sample trip with {len(self.trips['sample']['data'])} records")
            return
        if self.data_dir is None or not os.path.exists(self.data_dir):
            raise ValueError("Please provide a valid data directory")
        # Load all CSV files from the data directory
        driver_trips = defaultdict(list)
```

```
        trip_files = glob.glob(os.path.join(self.data_dir, "*.csv"))
        for trip_file in trip_files:
            # Expecting filename format: driver_tripid.csv (e.g., A_trip1.csv)
            base_name = os.path.basename(trip_file)
            parts = base_name.split('_')
            if len(parts) < 2:
                print(f"Skipping file with unexpected format: {base_name}")
                continue
            driver = parts[0]
            trip_id = os.path.splitext('_'.join(parts[1:]))[0]
            try:
                trip_data = pd.read_csv(trip_file)
                self.trips[trip_id] = {'driver': driver, 'data': trip_data}
                driver_trips[driver].append(trip_id)
            except Exception as e:
                print(f"Error loading {trip_file}: {e}")
        self.drivers = list(driver_trips.keys())
        print(f"Loaded {len(self.trips)} trips from {len(self.drivers)} drivers")
        # Print summary of trips per driver
        for driver, trips in driver_trips.items():
            print(f"Driver {driver}: {len(trips)} trips")
    def split_dataset(self, test_size=0.3, random_state=42):
        """
        Split the dataset into training and testing sets, ensuring proportional representation
        of all drivers.
        Parameters:
        -----------
        test_size : float
            Proportion of the dataset to include in the test split
```

```
        random_state : int
            Random seed for reproducibility
        """
        if not self.trips:
            raise ValueError("No trips loaded. Please load data first.")
        # Group trips by driver
        driver_trips = defaultdict(list)
        for trip_id, trip_info in self.trips.items():
            driver_trips[trip_info['driver']].append(trip_id)
        training_trips = []
        testing_trips = []
        # Split trips for each driver
        for driver, trips in driver_trips.items():
            train_ids, test_ids = train_test_split(trips, test_size=test_size,
random_state=random_state)
            training_trips.extend(train_ids)
            testing_trips.extend(test_ids)
        print(f"Training set: {len(training_trips)} trips")
        print(f"Testing set: {len(testing_trips)} trips")
        self.training_data = training_trips
        self.testing_data = testing_trips
        return training_trips, testing_trips
    def extract_features(self, trip_data):
        """
        Extract relevant features from a trip for driver behavior analysis.
        Parameters:
        -----------
        trip_data : DataFrame
            DataFrame containing trip data
        Returns:
```

```
--------
dict
    Dictionary of extracted features
"""
# Handle empty dataframes
if trip_data.empty:
    return {
        'avg_speed': 0, 'max_speed': 0,
        'avg_acceleration': 0, 'max_acceleration': 0,
        'avg_braking': 0, 'max_braking': 0,
        'idle_time_ratio': 1.0,
        'avg_engine_rpm': 0,
        'fuel_efficiency': 0,
        'gear_changes': 0,
        'steering_variance': 0,
        'avg_road_slope': 0,
        'aggressive_acceleration_count': 0,
        'aggressive_braking_count': 0,
        'sharp_turns_count': 0
    }
# Basic speed features
avg_speed = trip_data['car_speed'].mean()
max_speed = trip_data['car_speed'].max()
# Check for acceleration column alternatives
accel_col = None
for col in ['longitude_acceleration', 'latitude_acceleration']:
    if col in trip_data.columns:
        accel_col = col
        break
# Acceleration features
```

```
        if accel_col:
            accelerations = trip_data[accel_col]
            avg_acceleration = accelerations[accelerations > 0].mean() if
any(accelerations > 0) else 0
            max_acceleration = accelerations.max()
        else:
            # Approximate acceleration from speed differences if no acceleration column
            if 'car_speed' in trip_data.columns and len(trip_data) > 1:
                speed_diffs = trip_data['car_speed'].diff().fillna(0)
                positive_diffs = speed_diffs[speed_diffs > 0]
                avg_acceleration = positive_diffs.mean() if len(positive_diffs) > 0 else 0
                max_acceleration = speed_diffs.max()
            else:
                avg_acceleration = 0
                max_acceleration = 0
        # Braking features
        if 'brake_switch' in trip_data.columns and accel_col:
            brake_mask = (trip_data['brake_switch'] == 1) & (trip_data[accel_col] < 0)
            avg_braking = abs(trip_data.loc[brake_mask, accel_col].mean()) if
any(brake_mask) else 0
            max_braking = abs(trip_data[accel_col][trip_data[accel_col] < 0].min()) if
any(
                trip_data[accel_col] < 0) else 0
        elif 'brake_switch' in trip_data.columns:
            # If we have brake_switch but no acceleration data
            brake_mask = trip_data['brake_switch'] == 1
            braking_ratio = brake_mask.mean()
            avg_braking = braking_ratio * 10  # Arbitrary scale
            max_braking = avg_braking * 2 if avg_braking > 0 else 0
        else:
```

```
            # Fallback if no brake information
            avg_braking = 0
            max_braking = 0
        # Idle time
        idle_mask = trip_data['car_speed'] < 1  # Consider speeds below 1 as idle
        idle_time_ratio = idle_mask.mean()
        # Engine and fuel features
        avg_engine_rpm = trip_data['engine_speed'].mean() if 'engine_speed' in trip_data.columns else 0
        # Avoid division by zero in fuel efficiency calculation
        if 'fuel_usage' in trip_data.columns:
            moving_mask = trip_data['car_speed'] > 1
            if any(moving_mask):
                fuel_per_distance = trip_data.loc[moving_mask, 'fuel_usage'] / trip_data.loc[
                    moving_mask, 'car_speed'].clip(lower=0.1)
                fuel_efficiency = fuel_per_distance.mean()
            else:
                fuel_efficiency = 0
        else:
            fuel_efficiency = 0
        # Count gear changes
        if 'current_gear' in trip_data.columns:
            # Count changes, ignoring NaN values
            gear_series = trip_data['current_gear'].dropna()
            if len(gear_series) > 1:
                gear_changes = (gear_series.diff() != 0).sum()
            else:
                gear_changes = 0
        else:
```

```
        gear_changes = 0
    # Steering behavior
    if 'steering_wheel_angle' in trip_data.columns:
        steering_variance = trip_data['steering_wheel_angle'].var()
        # Count sharp turns (adjust threshold as needed)
        sharp_turns_threshold = 30  # degrees
        sharp_turns_count = (abs(trip_data['steering_wheel_angle']) > sharp_turns_threshold).sum()
    else:
        steering_variance = 0
        sharp_turns_count = 0
    # Road conditions
    avg_road_slope = trip_data['road_slope'].mean() if 'road_slope' in trip_data.columns else 0
    # Count aggressive events
    aggressive_accel_threshold = 2.5  # m/s²
    aggressive_brake_threshold = 3.0  # m/s²
    if accel_col:
        aggressive_acceleration_count = (trip_data[accel_col] > aggressive_accel_threshold).sum()
        aggressive_braking_count = (trip_data[accel_col] < -aggressive_brake_threshold).sum()
    else:
        aggressive_acceleration_count = 0
        aggressive_braking_count = 0
    return {
        'avg_speed': avg_speed,
        'max_speed': max_speed,
        'avg_acceleration': avg_acceleration,
        'max_acceleration': max_acceleration,
```

```
        'avg_braking': avg_braking,
        'max_braking': max_braking,
        'idle_time_ratio': idle_time_ratio,
        'avg_engine_rpm': avg_engine_rpm,
        'fuel_efficiency': fuel_efficiency,
        'gear_changes': gear_changes,
        'steering_variance': steering_variance,
        'avg_road_slope': avg_road_slope,
        'aggressive_acceleration_count': aggressive_acceleration_count,
        'aggressive_braking_count': aggressive_braking_count,
        'sharp_turns_count': sharp_turns_count
    }
def compute_all_trip_features(self):
    """
    Extract features from all trips and compile them into a dataframe.
    Returns:
    --------
    DataFrame
        DataFrame containing features for all trips
    """
    all_features = []
    for trip_id, trip_info in self.trips.items():
        driver = trip_info['driver']
        trip_data = trip_info['data']
        features = self.extract_features(trip_data)
        features['trip_id'] = trip_id
        features['driver'] = driver
        all_features.append(features)
    self.trip_features = pd.DataFrame(all_features)
    return self.trip_features
```

```
def cluster_driving_styles(self, n_clusters=3):
    """
    Cluster the driving styles using K-means algorithm.
    Parameters:
    -----------
    n_clusters : int
        Number of clusters (driving styles)
    Returns:
    --------
    tuple
        (cluster assignments, cluster centers, silhouette score)
    """
    if self.trip_features.empty:
        self.compute_all_trip_features()
    # Select numerical features for clustering
    feature_cols = [
        'avg_speed', 'max_speed', 'avg_acceleration', 'max_acceleration',
        'avg_braking', 'max_braking', 'idle_time_ratio', 'avg_engine_rpm',
        'fuel_efficiency', 'steering_variance', 'aggressive_acceleration_count',
        'aggressive_braking_count', 'sharp_turns_count'
    ]
    # Keep only features present in the dataset
    feature_cols = [col for col in feature_cols if col in self.trip_features.columns]
    if not feature_cols:
        print("Warning: No valid features found for clustering")
        # Create a placeholder cluster column
        self.trip_features['cluster'] = 0
        return [0] * len(self.trip_features), np.array([[0, 0]]), 0
    # Filter to training data only if split has been done
    if self.training_data:
```

```
            training_features =
self.trip_features[self.trip_features['trip_id'].isin(self.training_data)]
        else:
            training_features = self.trip_features
        # Ensure we have enough data for clustering
        if len(training_features) < n_clusters:
            print(
                f"Warning: Not enough data for {n_clusters} clusters. Using
{len(training_features)} clusters instead.")
            n_clusters = max(1, len(training_features) - 1)
        # Scale the features
        X = self.scaler.fit_transform(training_features[feature_cols])
        # Apply K-means clustering
        self.clusterer = KMeans(n_clusters=n_clusters, random_state=42)
        self.clusterer.fit(X)
        # Get cluster assignments and centers
        cluster_assignments = self.clusterer.labels_
        cluster_centers = self.clusterer.cluster_centers_
        # Calculate silhouette score if there's more than one cluster
        if len(set(cluster_assignments)) > 1 and len(X) > n_clusters:
            try:
                silhouette = silhouette_score(X, cluster_assignments)
            except Exception as e:
                print(f"Warning: Could not calculate silhouette score: {e}")
                silhouette = 0
        else:
            silhouette = 0
        # Add cluster assignment to features dataframe
        training_features['cluster'] = cluster_assignments
        # Make sure the cluster column exists in the main dataframe
```

```
            # For trips not in training set, assign them to the nearest cluster
            if len(self.trip_features) > len(training_features):
                # Add the cluster column to all rows in trip_features, initializing with NaN
                self.trip_features['cluster'] = float('nan')
                # Copy cluster assignments from training_features to trip_features
                for idx, row in training_features.iterrows():
                    trip_id = row['trip_id']
                    trip_idx = self.trip_features[self.trip_features['trip_id'] == trip_id].index
                    if len(trip_idx) > 0:
                        self.trip_features.loc[trip_idx, 'cluster'] = row['cluster']
                # For missing clusters (NaN values), predict based on features
                missing_cluster_idx = self.trip_features[self.trip_features['cluster'].isna()].index
                if len(missing_cluster_idx) > 0:
                    missing_features = self.trip_features.loc[missing_cluster_idx, feature_cols]
                    missing_scaled = self.scaler.transform(missing_features)
                    predicted_clusters = self.clusterer.predict(missing_scaled)
                    self.trip_features.loc[missing_cluster_idx, 'cluster'] = predicted_clusters
            else:
                # If we're using all data for training, just copy the cluster assignments
                self.trip_features['cluster'] = cluster_assignments
            # Ensure cluster column is integer type
            self.trip_features['cluster'] = self.trip_features['cluster'].astype(int)
            # Interpret clusters (assuming 3 clusters: economical, moderate, aggressive)
            # This is a simple interpretation based on average speed and acceleration
            try:
                cluster_details = pd.DataFrame()
                for i in range(n_clusters):
                    cluster_data = self.trip_features[self.trip_features['cluster'] == i]
```

```
            if not cluster_data.empty:
                cluster_details = pd.concat([cluster_details, pd.DataFrame({
                    'cluster': [i],
                    'size': len(cluster_data),
                    'avg_speed': cluster_data['avg_speed'].mean(),
                    'avg_acceleration': cluster_data['avg_acceleration'].mean(),
                    'avg_braking': cluster_data['avg_braking'].mean(),
                    'fuel_efficiency': cluster_data['fuel_efficiency'].mean()
                })])
        # Label clusters based on aggressiveness (using avg_acceleration as proxy)
        # Initialize cluster labels dictionary
        self.cluster_labels = {i: f"Cluster {i}" for i in range(n_clusters)}
        if not cluster_details.empty and 'avg_acceleration' in cluster_details.columns:
            sorted_clusters = cluster_details.sort_values('avg_acceleration')
            if len(sorted_clusters) >= 1:
                self.cluster_labels[sorted_clusters.iloc[0]['cluster']] = 'Economical'
            if len(sorted_clusters) >= 2:
                self.cluster_labels[sorted_clusters.iloc[1]['cluster']] = 'Moderate'
            if len(sorted_clusters) >= 3:
                self.cluster_labels[sorted_clusters.iloc[2]['cluster']] = 'Aggressive'
        print("Cluster details:")
        for i in range(n_clusters):
            cluster_data = cluster_details[cluster_details['cluster'] == i]
            if not cluster_data.empty:
                label = self.cluster_labels.get(i, f"Cluster {i}")
                print(f"{label} (Cluster {i}): {cluster_data['size'].values[0]} trips")
                print(f"  Avg Speed: {cluster_data['avg_speed'].values[0]:.2f}")
                if 'avg_acceleration' in cluster_data.columns:
```

```
                print(f" Avg Acceleration: {cluster_data['avg_acceleration'].values[0]:.2f}")
            if 'avg_braking' in cluster_data.columns:
                print(f" Avg Braking: {cluster_data['avg_braking'].values[0]:.2f}")
            if 'fuel_efficiency' in cluster_data.columns:
                print(f" Fuel Efficiency: {cluster_data['fuel_efficiency'].values[0]:.2f}")
        except Exception as e:
            print(f"Warning: Could not interpret clusters: {e}")
        return cluster_assignments, cluster_centers, silhouette
    def predict_driver(self):
        """
        Train a model to predict the driver based on driving features.
        Returns:
        --------
        tuple
            (accuracy, confusion matrix)
        """
        if self.trip_features.empty:
            self.compute_all_trip_features()
        if not self.training_data or not self.testing_data:
            self.split_dataset()
        # Select features for driver identification
        feature_cols = [
            'avg_speed', 'max_speed', 'avg_acceleration', 'max_acceleration',
            'avg_braking', 'max_braking', 'idle_time_ratio', 'avg_engine_rpm',
            'fuel_efficiency', 'steering_variance', 'aggressive_acceleration_count',
            'aggressive_braking_count', 'sharp_turns_count'
        ]
        # Keep only features present in the dataset
```

```
        feature_cols = [col for col in feature_cols if col in self.trip_features.columns]
        # Get training and testing data
        X_train = self.trip_features[self.trip_features['trip_id'].isin(self.training_data)][feature_cols]
        y_train = self.trip_features[self.trip_features['trip_id'].isin(self.training_data)]['driver']
        X_test = self.trip_features[self.trip_features['trip_id'].isin(self.testing_data)][feature_cols]
        y_test = self.trip_features[self.trip_features['trip_id'].isin(self.testing_data)]['driver']
        # Scale features
        X_train_scaled = self.scaler.transform(X_train)
        X_test_scaled = self.scaler.transform(X_test)
        # Train Random Forest Classifier
        self.driver_classifier = RandomForestClassifier(n_estimators=100, random_state=42)
        self.driver_classifier.fit(X_train_scaled, y_train)
        # Predict and evaluate
        y_pred = self.driver_classifier.predict(X_test_scaled)
        accuracy = accuracy_score(y_test, y_pred)
        conf_matrix = confusion_matrix(y_test, y_pred)
        print(f"Driver identification accuracy: {accuracy:.4f}")
        return accuracy, conf_matrix
    def evaluate_driver_consistency(self):
        """
        Evaluate the consistency of each driver's behavior across their trips.
        Returns:
        --------
        DataFrame
            DataFrame containing consistency metrics for each driver
```

```
        """
        if self.trip_features.empty:
            self.compute_all_trip_features()
        if 'cluster' not in self.trip_features.columns:
            self.cluster_driving_styles()
        consistency_metrics = []
        for driver in self.drivers:
            driver_trips = self.trip_features[self.trip_features['driver'] == driver]
            if len(driver_trips) <= 1:
                continue
            # Calculate standard deviation of key features
            speed_std = driver_trips['avg_speed'].std()
            accel_std = driver_trips['avg_acceleration'].std()
            brake_std = driver_trips['avg_braking'].std()
            # Calculate cluster consistency (% of trips in most common cluster)
            if 'cluster' in driver_trips.columns:
                cluster_counts = driver_trips['cluster'].value_counts(normalize=True)
                most_common_cluster = cluster_counts.idxmax()
                cluster_consistency = cluster_counts[most_common_cluster]
                cluster_label = self.cluster_labels.get(most_common_cluster, f"Cluster {most_common_cluster}")
            else:
                cluster_consistency = 0
                cluster_label = "Unknown"
            # Composite consistency score (lower is more consistent)
            # Normalize each component to [0, 1] range
            max_speed_std = self.trip_features['avg_speed'].std() * 2 or 1  # Avoid division by zero
            max_accel_std = self.trip_features['avg_acceleration'].std() * 2 or 1
            max_brake_std = self.trip_features['avg_braking'].std() * 2 or 1
```

```
        norm_speed_std = speed_std / max_speed_std
        norm_accel_std = accel_std / max_accel_std
        norm_brake_std = brake_std / max_brake_std
        # Composite score (lower is more consistent)
        composite_score = (norm_speed_std + norm_accel_std + norm_brake_std) /
3
        # Consistency value (higher is more consistent)
        consistency_value = 1 – composite_score
        consistency_metrics.append({
            'driver': driver,
            'num_trips': len(driver_trips),
            'speed_std': speed_std,
            'acceleration_std': accel_std,
            'braking_std': brake_std,
            'cluster_consistency': cluster_consistency,
            'common_cluster': cluster_label,
            'consistency_score': consistency_value
        })
    consistency_df = pd.DataFrame(consistency_metrics)
    if not consistency_df.empty:
        print("\nDriver consistency metrics:")
        for _, row in consistency_df.iterrows():
            print(f"Driver {row['driver']} ({row['num_trips']} trips):")
            print(f"  Speed std: {row['speed_std']:.2f}")
            print(f"  Acceleration std: {row['acceleration_std']:.2f}")
            print(f"  Braking std: {row['braking_std']:.2f}")
            print(f"  Cluster consistency: {row['cluster_consistency']:.2%}")
            print(f"  Most common style: {row['common_cluster']}")
            print(f"  Overall consistency score: {row['consistency_score']:.2f}/1.00")
    return consistency_df
```

```
    def detect_security_threats(self, trip_data, window_size=100):
        """
        Implement Bayesian approach to detect security threats in CAN bus transactions.
        This implementation follows the mathematical model from the dissertation.
        Parameters:
        -----------
        trip_data : DataFrame
            DataFrame containing CAN bus transactions for a trip
        window_size : int
            Size of the sliding window for transaction analysis
        Returns:
        --------
        dict
            Dictionary with detection results
        """
        # Initialize results
        results = {
            'normal_transactions': 0,
            'suspicious_transactions': 0,
            'threat_transactions': 0,
            'details': []
        }
        # Initialize pattern tracking
        message_counts = {}  # For tracking message repetition patterns
        last_pattern = []  # For tracking sequence patterns
        # Process transactions in sliding windows
        total_transactions = len(trip_data)
        # Reset the prior probabilities to initial values
        self.prior_probabilities = {
```

```
    'normal': 0.90,  # Normal operation (P1)
    'failure': 0.05,  # System failure (P2)
    'intrusion': 0.05  # Unauthorized access/intrusion (P3)
}
# Process each transaction (row in trip_data)
for i, transaction in trip_data.iterrows():
    # Extract CAN message ID (if available in the data)
    can_id = None
    for id_col in ['can_id', 'message_id', 'id']:
        if id_col in transaction:
            can_id = transaction[id_col]
            break
    # If no CAN ID column, use a placeholder
    if can_id is None:
        # Use a combination of columns to create a pseudo-identifier
        if 'brake_switch' in transaction and transaction['brake_switch'] == 1:
            can_id = "ID_BRAKE"  # Brake-related message
        elif 'steering_wheel_angle' in transaction and abs(transaction['steering_wheel_angle']) > 5:
            can_id = "ID_STEER"  # Steering-related message
        elif 'car_speed' in transaction and transaction['car_speed'] > 0:
            can_id = "ID_SPEED"  # Speed-related message
        elif 'engine_speed' in transaction:
            can_id = "ID_RPM"  # Engine RPM message
        else:
            can_id = "ID_UNKNOWN"
    # Extract reaction time (simulated from data intervals if not present)
    if 'reaction_time' in transaction:
        reaction_time = transaction['reaction_time']
    else:
```

```
            # Simulate reaction time based on pattern type probabilities
            # Add slight randomness for realism
            p_normal = self.prior_probabilities['normal']
            p_failure = self.prior_probabilities['failure']
            p_intrusion = self.prior_probabilities['intrusion']
            # Weighted average of typical reaction times
            reaction_time = (p_normal * 10 + p_failure * 25 + p_intrusion * 15)
            # Add randomness
            reaction_time += np.random.normal(0, 2)
        # Track message repetition patterns
        if can_id in message_counts:
            message_counts[can_id] += 1
        else:
            message_counts[can_id] = 1
        # Track sequence patterns
        last_pattern.append(can_id)
        if len(last_pattern) > 5:  # Keep only recent messages
            last_pattern.pop(0)
        # Check for repetition anomalies (used for intrusion detection)
        repetition_anomaly = False
        if message_counts.get(can_id, 0) > 10:
            # If same message repeats more than 10 times in window
            repetition_anomaly = True
        # Calculate likelihoods for each pattern (normal, failure, intrusion)
        likelihood_normal = self._calculate_likelihood(transaction, 'normal',
reaction_time)
        likelihood_failure = self._calculate_likelihood(transaction, 'failure',
reaction_time)
        likelihood_intrusion = self._calculate_likelihood(transaction, 'intrusion',
reaction_time)
```

```
            # Adjust intrusion likelihood based on repetition anomaly
            if repetition_anomaly:
                likelihood_intrusion *= 2.0  # Significantly boost likelihood if repetition detected
            # Apply Bayes' theorem to calculate posterior probabilities
            normalization_constant = (
                likelihood_normal * self.prior_probabilities['normal'] +
                likelihood_failure * self.prior_probabilities['failure'] +
                likelihood_intrusion * self.prior_probabilities['intrusion']
            )
            if normalization_constant == 0:
                # Handle edge case – if all likelihoods are zero
                posterior_normal = self.prior_probabilities['normal']
                posterior_failure = self.prior_probabilities['failure']
                posterior_intrusion = self.prior_probabilities['intrusion']
            else:
                posterior_normal = (likelihood_normal * self.prior_probabilities['normal']) / normalization_constant
                posterior_failure = (likelihood_failure * self.prior_probabilities['failure']) / normalization_constant
                posterior_intrusion = (likelihood_intrusion * self.prior_probabilities['intrusion']) / normalization_constant
                # Підсилення розпізнавання перехоплень
                # Перевіряємо додаткові індикатори перехоплення (модифікація реакції системи)
                is_potential_intrusion = False
            # Класифікуємо транзакцію використовуючи результати Байєсівського аналізу
            threat_threshold = self.threat_detection_threshold
            suspicious_threshold = self.suspicious_activity_threshold
```

```
            # Спочатку перевіряємо, чи це нормальна транзакція
            if posterior_normal >= (1 – suspicious_threshold):
                transaction_type = 'normal'
                results['normal_transactions'] += 1
            # Потім визначаємо, чи це перехоплення (intrusion)
            elif posterior_intrusion >= suspicious_threshold:
                transaction_type = 'suspicious'  # Класифікуємо як підозрілу транзакцію (перехоплення)
                results['suspicious_transactions'] += 1
            # Нарешті, якщо це не нормальна і не перехоплення, перевіряємо, чи це збій
            elif posterior_failure >= threat_threshold:
                transaction_type = 'failure'  # Класифікуємо як загрозливу транзакцію (збій)
                results['threat_transactions'] += 1
            # Якщо всі пороги не перевищені, за замовчуванням вважаємо транзакцію нормальною
            else:
                transaction_type = 'normal'
                results['normal_transactions'] += 1
            # Add details for significant transactions
            if transaction_type != 'normal':
                results['details'].append({
                    'index': i,
                    'can_id': can_id,
                    'type': transaction_type,
                    'reaction_time': reaction_time,
                    'likelihoods': {
                        'normal': likelihood_normal,
                        'failure': likelihood_failure,
```

```
                    'intrusion': likelihood_intrusion
                },
                'posteriors': {
                    'normal': posterior_normal,
                    'failure': posterior_failure,
                    'intrusion': posterior_intrusion
                }
            })
            # Update prior probabilities for next iteration using smoothing
            self._update_priors(transaction_type, posterior_normal, posterior_failure, posterior_intrusion)
            # Update behavioral metrics based on transaction
            self._update_behavioral_metrics(transaction, transaction_type)
            # Reset message counts after window size is reached
            if i % window_size == 0:
                message_counts = {}
        # Calculate overall statistics
        results['total_transactions'] = total_transactions
        results['normal_percentage'] = results['normal_transactions'] / total_transactions * 100 if total_transactions > 0 else 0
        results['suspicious_percentage'] = results['suspicious_transactions'] / total_transactions * 100 if total_transactions > 0 else 0
        results['threat_percentage'] = results['threat_transactions'] / total_transactions * 100 if total_transactions > 0 else 0
        return results
    def _update_behavioral_metrics(self, transaction, transaction_type):
        """
        Update behavioral metrics based on the current transaction.
        Implements the adaptive metrics updating from section 4.4 of the dissertation.
        Parameters:
```

```
        ----------
        transaction : Series
            Current transaction data
        transaction_type : str
            Classified type ('normal', 'failure', 'suspicious')
        """
        # Extract dynamic characteristics
        accel_value = 0
        for accel_col in ['longitude_acceleration', 'latitude_acceleration']:
            if accel_col in transaction:
                accel_value = transaction[accel_col]
                break
        braking = 1 if transaction.get('brake_switch', 0) == 1 else 0
        if 'steering_wheel_angle' in transaction:
            steering_angle = abs(transaction['steering_wheel_angle'])
            steering_sharpness = min(1.0, steering_angle / 30.0)
        else:
            steering_sharpness = 0.0
        # 1. Update driving aggressiveness (g)
        # Implemented using weighted combination per section 4.4.1
        w_a = 0.3  # Weight for acceleration
        w_b = 0.3  # Weight for braking
        w_m = 0.4  # Weight for maneuver sharpness
        # Normalize acceleration to [0,1] range, assuming 5 m/s² is very aggressive
        norm_accel = min(1.0, abs(accel_value) / 5.0)
        # Calculate current aggressiveness
        current_aggressiveness = w_a * norm_accel + w_b * braking + w_m *
steering_sharpness
        # Normalize to [0,1] scale
        current_aggressiveness = min(1.0, current_aggressiveness)
```

 # Update with exponential smoothing

self.driving_aggressiveness = 0.9 * *self.driving_aggressiveness* + 0.1 * *current_aggressiveness*

 # 2. Update anomaly history (*h*)

 # Implemented using exponential decay per section 4.4.2

decay_factor = 0.95 *# β in the dissertation*

 # Indicator function: 1 *if suspicious or failure*, 0 *if normal*

anomaly_indicator = 1 *if transaction_type* != ‘*normal*’ *else* 0

 # Update anomaly history

self.anomaly_history = *decay_factor* * *self.anomaly_history* + (1 – *decay_factor*) * *anomaly_indicator*

 # 3. Update style deviation (δ)

 # This would track deviation from the driver’s usual style

 # Simplified implementation

def _calculate_likelihood(*self*, *transaction*, *pattern_type*, *reaction_time*):

"""

Покращений метод обчислення ймовірності належності транзакції до певного шаблону.

Parameters:

---

transaction : *Series*

Дані транзакції

pattern_type : *str*

Тип шаблону (‘*normal*’, ‘*failure*’, або ‘*intrusion*’)

reaction_time : *float*

Час реакції в мілісекундах

Returns:

---

float

Значення ймовірності

```
        """
        # Параметри розподілу часу реакції
        rt_mean = self.reaction_time_distributions[pattern_type]['mean']
        rt_std = self.reaction_time_distributions[pattern_type]['std']
        # 1. Обчислюємо ймовірність на основі часу реакції
        # Формула: exp(-0.5 * ((reaction_time – mean) / std) ** 2) / (std * sqrt(2π))
        rt_likelihood = np.exp(-0.5 * ((reaction_time – rt_mean) / rt_std) ** 2) / (rt_std
* np.sqrt(2 * np.pi))
        # Нормалізуємо відносно максимальної щільності розподілу
        max_density = 1 / (rt_std * np.sqrt(2 * np.pi))
        rt_likelihood_normalized = rt_likelihood / max_density
        # 2. Визначаємо параметри транзакції з захистом від відсутніх значень
        speed = transaction.get('car_speed', 0)
        # Значення прискорення
        accel_value = 0
        for accel_col in ['longitude_acceleration', 'latitude_acceleration']:
            if accel_col in transaction:
                accel_value = transaction[accel_col]
                break
        # Стан гальмування
        braking = 1 if transaction.get('brake_switch', 0) == 1 else 0
        # Значення кута керма
        if 'steering_wheel_angle' in transaction:
            steering_angle = abs(transaction['steering_wheel_angle'])
            steering_sharpness = min(1.0, steering_angle / 30.0)
        else:
            steering_sharpness = 0.0
        # 3. Обчислюємо ймовірність даних залежно від типу шаблону
        if pattern_type == 'normal':
```

```
        # Для нормальної поведінки: помірна швидкість, низьке прискорення, плавне керування
        # Швидкість: нормальний розподіл навколо типової швидкості
        speed_mean, speed_std = 60, 30
        speed_likelihood = np.exp(-0.5 * ((speed – speed_mean) / speed_std) ** 2) / (speed_std * np.sqrt(2 * np.pi))
        speed_likelihood = min(1.0, speed_likelihood / (1 / (speed_std * np.sqrt(2 * np.pi))))
        # Прискорення: нормальний розподіл навколо 0
        accel_mean, accel_std = 0, 1.5
        accel_likelihood = np.exp(-0.5 * ((accel_value – accel_mean) / accel_std) ** 2) / (
                accel_std * np.sqrt(2 * np.pi))
        accel_likelihood = min(1.0, accel_likelihood / (1 / (accel_std * np.sqrt(2 * np.pi))))
        # Гальмування: в нормальному стані гальмування рідше
        braking_likelihood = 0.85 if braking == 0 else 0.15
        # Кермування: очікується плавне (низькі значення)
        steering_likelihood = 1.0 – steering_sharpness ** 2  # Квадратична залежність для більшої чутливості
        # Загальна ймовірність – зважене середнє
        data_likelihood = (0.3 * speed_likelihood +
                           0.3 * accel_likelihood +
                           0.2 * braking_likelihood +
                           0.2 * steering_likelihood)
        # Зменшуємо ймовірність нормальної поведінки при екстремальних значеннях
        if abs(accel_value) > 3 or speed > 120 or steering_sharpness > 0.7:
            data_likelihood *= 0.7
    elif pattern_type == 'failure':
```

```
            # Для збоїв: екстремальні значення, раптові зміни
            # Швидкість: вища ймовірність для аномальних значень
            speed_dev = abs(speed – 60) / 30
            speed_likelihood = min(1.0, speed_dev / 1.8)  # Підвищуємо поріг для зменшення хибних спрацьовувань
            # Прискорення: вища ймовірність для екстремальних значень
            accel_dev = abs(accel_value) / 2.2  # Підвищуємо поріг
            accel_likelihood = min(1.0, accel_dev)
            # Гальмування: неочікувані патерни гальмування
            braking_likelihood = 0.7 if ((braking == 1 and accel_value > 0.5) or  # Гальмування при прискоренні
                                         (
                                             braking == 0 and accel_value < -3.5)) else 0.3  # Різке уповільнення без гальмування
            # Кермування: різкі повороти
            steering_likelihood = min(1.0, steering_sharpness * 1.2)  # Зменшуємо чутливість
            # Зважене середнє компонентів
            data_likelihood = (0.3 * speed_likelihood +
                               0.35 * accel_likelihood +  # Надаємо більшу вагу прискоренню для збоїв
                               0.2 * braking_likelihood +
                               0.15 * steering_likelihood)
            # Підсилюємо ймовірність при явних ознаках збою, але з вищими порогами
            if (abs(accel_value) > 5.5) or (speed > 160) or (abs(accel_value) > 3.8 and braking == 1):
                data_likelihood = min(1.0, data_likelihood * 1.4)
        elif pattern_type == 'intrusion':
```

```
            # Для перехоплень: нормальні на перший погляд дані, але з підозрілими патернами
            # За замовчуванням – помірна ймовірність
            data_likelihood = 0.4
            # Підвищуємо ймовірність перехоплення при типових ознаках
            if 12 <= reaction_time <= 18:  # Типовий час реакції для перехоплень
                data_likelihood *= 1.3
            # Ознаки повторюваності (replay-атаки) – підозріло стабільні значення
            if abs(accel_value) < 0.1 and speed > 20:
                data_likelihood *= 1.3
            # Підвищуємо ймовірність, якщо спостерігали підозрілі патерни нещодавно
            if self.anomaly_history > 0.25:  # Зменшуємо поріг аномальної історії
                data_likelihood = min(1.0, data_likelihood * 1.15)
            # Зменшуємо ймовірність перехоплення при явних ознаках збою
            if abs(accel_value) > 4.0 or abs(steering_sharpness) > 0.6:
                data_likelihood *= 0.7
        # 4. Обчислюємо ймовірність контексту
        context_likelihood = 0.5  # Нейтральне значення за замовчуванням
        if pattern_type == 'normal':
            context_likelihood = 0.7
            # Зменшуємо при агресивному водінні
            if self.driving_aggressiveness > 0.5:  # Зменшуємо поріг
                context_likelihood *= 0.85
        elif pattern_type == 'failure':
            context_likelihood = 0.4
            # Підвищуємо при недавніх аномаліях
            if self.anomaly_history > 0.35:  # Зменшуємо поріг
                context_likelihood = min(1.0, context_likelihood * 1.4)
        elif pattern_type == 'intrusion':
```

```
            context_likelihood = 0.35
            # Підвищуємо при підозрілих патернах
            if self.anomaly_history > 0.25:  # Зменшуємо поріг
                context_likelihood = min(1.0, context_likelihood * 1.6)
        # 5. Фінальна ймовірність – добуток компонентів
        final_likelihood = rt_likelihood_normalized * data_likelihood *
context_likelihood
        # Додаткове згладжування для зменшення хибних спрацьовувань
        if pattern_type == 'normal':
            # Підсилюємо нормальну ймовірність для зменшення хибних тривог
            final_likelihood = min(1.0, final_likelihood * 1.1)
        elif pattern_type == 'failure':
            # Зменшуємо ймовірність збою для скорочення хибних спрацьовувань
            final_likelihood = final_likelihood * 0.9
        return final_likelihood
    def _update_priors(self, transaction_type, posterior_normal, posterior_failure,
posterior_intrusion):
        """
        Update prior probabilities based on the classification of the current transaction.
        Parameters:
        -----------
        transaction_type : str
            Classified type of the transaction
        posterior_normal, posterior_failure, posterior_intrusion : float
            Posterior probabilities calculated for the transaction
        """
        # Update prior probabilities using smoothing factors
        if transaction_type == 'normal':
            alpha = self.alpha_normal
        elif transaction_type == 'failure':
```

```
            alpha = self.alpha_failure
        else:  # suspicious or intrusion
            alpha = self.alpha_intrusion
        # Update priors
        self.prior_probabilities['normal'] = (1 - alpha) * self.prior_probabilities['normal'] + alpha * posterior_normal
        self.prior_probabilities['failure'] = (1 - alpha) * self.prior_probabilities['failure'] + alpha * posterior_failure
        self.prior_probabilities['intrusion'] = (1 - alpha) * self.prior_probabilities['intrusion'] + alpha * posterior_intrusion
        # Normalize to ensure sum = 1
        total = sum(self.prior_probabilities.values())
        for key in self.prior_probabilities:
            self.prior_probabilities[key] /= total
        # Update behavioral metrics
        # Update driving aggressiveness based on transaction data
        # (simplified implementation)
        self.driving_aggressiveness = 0.9 * self.driving_aggressiveness + 0.1 * (
            1 if transaction_type != 'normal' else 0
        )
        # Update anomaly history with decay
        decay_factor = 0.95
        self.anomaly_history = decay_factor * self.anomaly_history + (1 - decay_factor) * (
            1 if transaction_type != 'normal' else 0
        )
    def visualize_clusters(self):
        """
        Visualize the clustering results.
        Returns:
```

```
        --------
        matplotlib figure
        """
        # Ensure clustering has been performed
        if 'cluster' not in self.trip_features.columns:
            print("Running clustering before visualization...")
            self.cluster_driving_styles()
        fig, axes = plt.subplots(2, 2, figsize=(14, 12))
        # Plot 1: Avg Speed vs Avg Acceleration colored by cluster
        ax = axes[0, 0]
        # Check if cluster column exists before attempting to plot
        try:
            scatter = ax.scatter(
                self.trip_features['avg_speed'],
                self.trip_features['avg_acceleration'],
                c=self.trip_features['cluster'],
                cmap='viridis',
                alpha=0.7,
                s=50
            )
            ax.set_title('Driving Styles: Speed vs Acceleration')
            ax.set_xlabel('Average Speed')
            ax.set_ylabel('Average Acceleration')
            legend1 = ax.legend(*scatter.legend_elements(),
                                title="Clusters")
            ax.add_artist(legend1)
        except KeyError as e:
            print(f"Warning: Could not create cluster scatter plot due to missing column:
{e}")
            ax.text(0.5, 0.5, "Cluster visualization unavailable",
```

```
            horizontalalignment='center', verticalalignment='center')
    # Plot 2: Avg Speed vs Avg Braking colored by driver
    ax = axes[0, 1]
    driver_colors = ['blue', 'green', 'red', 'purple', 'orange'][:len(self.drivers)]
    for driver, color in zip(self.drivers, driver_colors):
        driver_data = self.trip_features[self.trip_features['driver'] == driver]
        ax.scatter(
            driver_data['avg_speed'],
            driver_data['avg_braking'],
            label=f'Driver {driver}',
            color=color,
            alpha=0.7
        )
    ax.set_title('Driving Behavior: Speed vs Braking by Driver')
    ax.set_xlabel('Average Speed')
    ax.set_ylabel('Average Braking')
    ax.legend()
    # Plot 3: Feature boxplots by driver
    ax = axes[1, 0]
    feature = 'avg_acceleration'  # Can be changed to any feature
    sns.boxplot(x='driver', y=feature, data=self.trip_features, ax=ax)
    ax.set_title(f'{feature} Distribution by Driver')
    ax.set_xlabel('Driver')
    ax.set_ylabel(feature)
    # Plot 4: Cluster distribution by driver – with error handling
    ax = axes[1, 1]
    try:
        if 'cluster' in self.trip_features.columns:
            driver_cluster_counts = pd.crosstab(
                self.trip_features['driver'],
```

```
                self.trip_features['cluster'],
                normalize='index'
            )
            driver_cluster_counts.plot(kind='bar', stacked=True, ax=ax, colormap='viridis')
            ax.set_title('Driving Style Distribution by Driver')
            ax.set_xlabel('Driver')
            ax.set_ylabel('Proportion')
            # Handle cluster labels with error prevention
            if hasattr(self, 'cluster_labels') and self.cluster_labels:
                n_clusters = len(set(self.trip_features['cluster']))
                labels = [self.cluster_labels.get(i, f'Cluster {i}') for i in range(n_clusters)]
                ax.legend(title='Cluster', labels=labels)
            else:
                ax.legend(title='Cluster')
        else:
            ax.text(0.5, 0.5, "Cluster data unavailable",
                    horizontalalignment='center', verticalalignment='center')
    except Exception as e:
        print(f"Warning: Could not create cluster distribution plot: {e}")
        ax.text(0.5, 0.5, f"Cluster visualization error: {type(e).__name__}",
                horizontalalignment='center', verticalalignment='center')
    plt.tight_layout()
    return fig
def visualize_confusion_matrix(self, conf_matrix=None):
    """
    Visualize the confusion matrix for driver identification.
    Parameters:
    -----------
```

```
    conf_matrix : ndarray
        Confusion matrix (optional, will run prediction if not provided)
    Returns:
    --------
    matplotlib figure
    """
    if conf_matrix is None:
        _, conf_matrix = self.predict_driver()
    fig, ax = plt.subplots(figsize=(8, 6))
    sns.heatmap(conf_matrix, annot=True, fmt='d', cmap='Blues', cbar=False,
                xticklabels=self.drivers, yticklabels=self.drivers)
    ax.set_title('Driver Identification Confusion Matrix')
    ax.set_xlabel('Predicted Driver')
    ax.set_ylabel('True Driver')
    return fig
def run_full_analysis(self):
    """
    Run the complete analysis pipeline and display results.
    """
    print("====== CAN Bus Security and Driver Behavior Analysis ======")
    # Step 1: Load and preprocess data
    if not self.trips:
        print("No data loaded. Please load data first.")
        return
    # Step 2: Split dataset and extract features
    if not self.training_data:
        self.split_dataset()
    self.compute_all_trip_features()
    print(f"\nExtracted features for {len(self.trip_features)} trips")
    # Step 3: Cluster driving styles
```

```
print("\n--- Driving Style Clustering ---")
cluster_assignments, _, silhouette = self.cluster_driving_styles()
print(f"Silhouette Score: {silhouette:.4f}")
# Step 4: Driver identification
print("\n--- Driver Identification ---")
accuracy, conf_matrix = self.predict_driver()
# Step 5: Driver consistency evaluation
print("\n--- Driver Consistency Analysis ---")
consistency_metrics = self.evaluate_driver_consistency()
# Step 6: Run Bayesian security threat detection on a sample trip
print("\n--- Security Threat Detection (Sample Trip) ---")
sample_trip_id = list(self.trips.keys())[0]
sample_trip_data = self.trips[sample_trip_id]['data']
security_results = self.detect_security_threats(sample_trip_data)
print(f"Normal transactions: {security_results['normal_transactions']}")
print(f"Suspicious transactions: {security_results['suspicious_transactions']}")
print(f"Threat transactions: {security_results['threat_transactions']}")
if security_results['details']:
    print("\nDetailed security findings:")
    for detail in security_results['details'][:5]:  # Show top 5 findings
        print(f"Transaction {detail['index']}: {detail['type'].upper()}")
        print(f"  Normal probability: {detail['posteriors']['normal']:.4f}")
        print(f"  Failure probability: {detail['posteriors']['failure']:.4f}")
        print(f"  Intrusion probability: {detail['posteriors']['intrusion']:.4f}")
# Generate visualizations
print("\nGenerating visualizations...")
self.visualize_clusters()
self.visualize_confusion_matrix(conf_matrix)
print("\nAnalysis complete!")
```