МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

Чорноморський національний університет імені Петра Могили

Факультет комп'ютерних наук

Кафедра інтелектуальних інформаційних систем

ДОПУЩЕНО ДО ЗАХИСТУ

В. о. завідувача кафедри інтелектуальних інформаційних систем

_____ Євген СІДЕНКО

«____»_____ 2025 р.

КВАЛІФІКАЦІЙНА РОБОТА

НА ЗДОБУТТЯ ОСВІТНЬОГО СТУПЕНЯ МАГІСТРА

ПРОГНОЗУВАННЯ ПОТРЕБ У РЕСУРСАХ ТА ОПТИМІЗАЦІЯ ЛОГІСТИЧНИХ МАРШРУТІВ НА ОСНОВІ НЕЙРОННИХ МЕРЕЖ

Спеціальність 122 Комп'ютерні науки

Освітня програма «Інтелектуальні інформаційні системи»

Здобувач _____ Ілля ГОРШОЛЄПОВ

«___»_____ 2025 р.

Керівник канд. техн. наук, доцент _____ Євген СІДЕНКО

«___»_____ 2025 р.

Миколаїв – 2025

Чорноморський національний університет імені Петра Могили
(повне найменування закладу вищої освіти)

Факультет	Комп'ютерних наук
Кафедра	Інтелектуальних інформаційних систем
Рівень вищої освіти	Другий (магістерський)
Освітній ступень	Магістр
Спеціальність	122 Комп'ютерні науки
Освітня програма	Інтелектуальні інформаційні системи

ЗАТВЕРДЖУЮ

В. о. завідувача кафедри інтелектуальних інформаційних систем

_____ Євген СІДЕНКО

«____» _____ 2025 р.

ЗАВДАННЯ
на кваліфікаційну роботу здобувача

Горшколєпов Ілля Віталійович
(прізвище, ім'я, по батькові здобувача)

1. Тема кваліфікаційної роботи: ПРОГНОЗУВАННЯ ПОТРЕБ У РЕСУРСАХ ТА ОПТИМІЗАЦІЯ ЛОГІСТИЧНИХ МАРШРУТІВ НА ОСНОВІ НЕЙРОННИХ МЕРЕЖ

Керівник роботи: Сіденко Євген Вікторович, в. о. зав. кафедри, канд. техн. наук, доцент
(прізвище, ім'я, по батькові, посада, науковий ступінь, вчене звання)

Затверджена наказом ЧНУ ім. Петра Могили від «_7_» _07_2025_р. № __184__.

2. Строк представлення кваліфікаційної роботи «___» _____ 20__ р.

3. Очікуваний результат роботи та початкові дані: дані про споживання логістичних ресурсів у вигляді часових рядів; дані про зовнішні фактори впливу (операційний темп, погодні умови, сезонність); характеристики тактичних маршрутів (дистанція, час у дорозі, рівень загрози); експертні оцінки та лінгвістичні змінні для системи нечіткої логіки; пріоритети місії, що задаються користувачем; моделі глибокого навчання (зокрема, рекурентні нейронні мережі

GRU) для прогнозування; алгоритми нечіткої логіки та метод багатокритеріальної оптимізації SAW для вибору маршрутів. Очікуваний результат: гібридна інтелектуальна система підтримки прийняття рішень для прогнозування потреб у ресурсах та оптимізації військових логістичних маршрутів.

4. Перелік питань, що підлягають розробці: аналіз особливостей управління військовою логістикою в умовах невизначеності та динамічних змін; огляд та порівняння сучасних методів підтримки прийняття рішень, включаючи машинне навчання та нечітку логіку; обґрунтування вибору архітектури гібридної системи, що поєднує GRU-мережі для прогнозування та метод Fuzzy SAW для оптимізації; розробка та навчання моделі прогнозування споживання ресурсів на основі рекурентних нейронних мереж; розробка модуля багатокритеріальної оптимізації маршрутів з використанням нечіткої логіки; програмна реалізація гібридної системи та інтерфейсу користувача; тестування та апробація розробленої системи в симуляційному середовищі, оцінка її ефективності.

5. Перелік графічних матеріалів: сторінок – 96, таблиць – 5, рисунків – 34, посилань – 50, додатків – 3.

Керівник роботи _____ Євген СІДЕНКО
(Особистий підпис) *(Власне ім'я ПРІЗВИЩЕ)*

Здобувач _____ Ілля ГОРШКОЛЄПОВ
(Особистий підпис) *(Власне ім'я ПРІЗВИЩЕ)*

Дата видачі завдання «28» червня 2025 р.

КАЛЕНДАРНИЙ ПЛАН
кваліфікаційної роботи

Тема: Прогнозування потреб у ресурсах та оптимізація логістичних маршрутів на основі нейронних мереж

№	Найменування роботи	Початок	Закінчення	Примітки
1.	Отримання завдання на виконання КР	27.06.2025	30.06.2025	
2.	Аналіз предметної області та постановка задачі	01.07.2025	20.07.2025	
3.	Огляд літературних джерел за темою кваліфікаційної роботи	21.07.2025	30.07.2025	
4.	Аналіз існуючих підходів та алгоритмів прогнозування.	01.09.2025	25.10.2025	
5.	Реалізація обраних технологій з аналізом отриманих результатів	26.10.2025	21.11.2025	
6.	Перший попередній захист КР на засіданні комісії кафедри	25.11.2025	26.11.2025	
7.	Корегування роботи за результатами попереднього захисту	27.11.2025	05.12.2025	
8.	Другий попередній захист КР на засіданні комісії кафедри	09.12.2025	09.12.2025	
9.	Доробка та остаточне оформлення КР	10.12.2025	12.12.2025	
10	Подання КР, її електронної копії та інших документів (відгуку, рецензії) до захисту	15.12.2025	16.12.2025	

Керівник роботи _____ Євген СІДЕНКО
(Особистий підпис) *(Власне ім'я ПРІЗВИЩЕ)*

Здобувач _____ Ілля ГОРШКОЛЄПОВ
(Особистий підпис) *(Власне ім'я ПРІЗВИЩЕ)*

Дата складання календарного плану
«8» липня 2025 р.

АНОТАЦІЯ

до кваліфікаційної роботи
здобувача групи 601м ЧНУ ім. Петра Могили

Горшколєпова Іллі Віталійовича

на тему: **«ПРОГНОЗУВАННЯ ПОТРЕБ У РЕСУРСАХ ТА ОПТИМІЗАЦІЯ ЛОГІСТИЧНИХ МАРШРУТІВ НА ОСНОВІ НЕЙРОННИХ МЕРЕЖ»**

Дана робота досліджує шляхи підвищення ефективності управління військовою логістикою шляхом переходу від детермінованих моделей до гібридних інтелектуальних систем підтримки прийняття рішень. У ній висвітлюються основні проблеми традиційних методів планування, зокрема їхня залежність від статичних даних, низька адаптивність до динамічних змін оперативної обстановки та нездатність ефективно працювати в умовах невизначеності ресурсного забезпечення. Звіт деталізує архітектуру розробленої гібридної системи, що поєднує потужність глибокого навчання для аналізу даних та гнучкість методів багатокритеріальної оптимізації. Описуються ключові технологічні інструменти реалізації, такі як мова програмування Python, бібліотеки TensorFlow та Keras для побудови нейронних мереж, бібліотека scikit-fuzzy для роботи з нечіткою логікою, а також фреймворк Streamlit для створення інтерактивного інтерфейсу користувача. Особлива увага приділяється розробці та тренуванню моделі на основі рекурентних нейронних мереж для високоточного прогнозування часових рядів споживання ресурсів. Також аналізується застосування гібридного методу Fuzzy SAW для оптимізації тактичних маршрутів, що дозволяє враховувати як кількісні показники, так і якісні експертні оцінки та пріоритети місії. Нарешті, наводяться результати апробації системи у симуляційному середовищі Arma 3, які підтверджують здатність розробленого рішення динамічно адаптуватися до змінних умов та надавати обґрунтовані рекомендації.

Актуальність теми зумовлена критичною необхідністю забезпечення боєздатності підрозділів в умовах сучасних конфліктів, що вимагає впровадження адаптивних інструментів для оперативного планування та мінімізації ризиків при постачанні ресурсів.

Метою дослідження є підвищення ефективності та адаптивності логістичного планування шляхом розробки гібридної СППР, яка інтегрує динамічні прогнози потреб у ресурсах з механізмами багатокритеріального прийняття рішень.

Об'єкт дослідження – процеси прогнозування споживання логістичних ресурсів та оптимізації тактичних маршрутів пересування у військових умовах.

Предмет дослідження – методи та алгоритми глибокого навчання для прогнозування часових рядів, а також гібридні методи багатокритеріальної оптимізації маршрутів.

Кваліфікаційна робота містить 96 сторінок, 34 рисунки, 5 таблиць, 50 використаних джерел та 3 додатки.

Ключові слова: військова логістика, система підтримки прийняття рішень, глибоке навчання, GRU, нечітка логіка, Fuzzy SAW, оптимізація маршрутів, прогнозування ресурсів, Python.

ABSTRACT

to the qualification work by the student of the group 601m of Petro Mohyla Black Sea National University

Horshkoliepov Illia

"FORECASTING RESOURCE NEEDS AND OPTIMIZATION OF LOGISTIC ROUTES BASED ON NEURAL NETWORKS"

This work investigates the enhancement of military logistics management efficiency through the transition from deterministic models to hybrid intelligent Decision Support Systems. It highlights the fundamental limitations of traditional planning methods, particularly their reliance on static data, low adaptability to dynamic changes in the operational environment, and inability to operate effectively under conditions of resource supply uncertainty. The report details the architecture of the developed hybrid system, which combines the computational power of Deep Learning for data analysis with the flexibility of multi-criteria optimization methods. Key technological implementation tools are described, including the Python programming language, TensorFlow and Keras libraries for constructing neural networks, the scikit-fuzzy library for fuzzy logic operations, and the Streamlit framework for creating an interactive user interface. Particular attention is given to the development and training of a model based on Gated Recurrent Units for high-precision forecasting of resource consumption time series. The application of the hybrid Fuzzy SAW method for tactical route optimization is also analyzed, enabling the consideration of both quantitative indicators and qualitative expert assessments as well as mission priorities. Finally, the results of system validation within the Arma 3 simulation environment are presented, confirming the developed solution's capability to dynamically adapt to changing conditions and provide substantiated recommendations.

The relevance of the topic is driven by the critical need to ensure unit combat readiness in modern conflict conditions, necessitating the implementation of adaptive tools for operational planning and risk minimization in resource supply.

The purpose of the research is to enhance the efficiency and adaptability of logistical planning by developing a hybrid DSS that integrates dynamic resource demand forecasts with multi-criteria decision-making mechanisms.

The object of the research is the processes of forecasting logistics resource consumption and optimizing tactical movement routes in military conditions.

The subject of the research comprises Deep Learning methods and algorithms for forecasting time series, as well as hybrid methods for multi-criteria route optimization.

The qualification work contains 96 pages, 34 figures, 5 tables, 50 used sources, and 3 appendices.

Keywords: military logistics, decision support system, Deep Learning, GRU, fuzzy logic, Fuzzy SAW, route optimization, resource forecasting, Python.

ЗМІСТ

СКОРОЧЕННЯ ТА УМОВНІ ПОЗНАКИ 4
ВСТУП 5
1 АНАЛІЗ ПРОБЛЕМИ ТА ПОСТАНОВКА ЗАДАЧІ 7
1.1 Сучасний стан проблеми управління військовою логістикою 7
1.2 Застосування штучного інтелекту для підтримки прийняття рішень 9
1.3 Огляд останніх публікацій та існуючих систем 12
1.4 Постановка задачі дослідження 15
Висновки до розділу 1 17
2 ТЕОРЕТИЧНІ ОСНОВИ ГІБРИДНОЇ СИСТЕМИ 19
2.1 Прогнозування часових рядів за допомогою рекурентних мереж GRU 19
2.2 Багатокритеріальне прийняття рішень в умовах невизначеності за допомогою нечіткої логіки 23
2.3 Архітектура інтегрованої гібридної системи 27
2.4 Огляд обраних технологій та інструментальних засобів розробки 29
Висновки до розділу 2 36
3 РОЗРОБКА ТА ПРОГРАМНА РЕАЛІЗАЦІЯ ГІБРИДНОЇ СИСТЕМИ ПІДТРИМКИ ПРИЙНЯТТЯ РІШЕНЬ 38
3.1 Концептуальна модель та функціональна структура гібридної системи 38
3.2 Програмна реалізація модуля прогнозування споживання пального 40
3.3 Програмна реалізація модуля оптимізації маршрутів 45
3.4 Реалізація інтерфейсу користувача та аналіз результатів навчання 50
Висновки до розділу 3 56
4 ТЕСТУВАННЯ ГІБРИДНОЇ СИСТЕМИ 58
4.1 Методологія та середовище експериментального дослідження 58
4.1.1 Обґрунтування вибору методу імітаційного моделювання 58
4.1.2 Характеристика симуляційного середовища Arma 3 59
4.1.3 Теорія подібності та масштабування експерименту 60

4.2 Сценарне моделювання 62
4.2.1 Характеристика оперативного сценарію «Шторм» 62
4.2.2 Топологія та параметризація маршрутів 64
4.2.3 Склад та конфігурація логістичних ешелонів 65
4.3 Аналіз результатів та валідація гібридного механізму 68
4.3.1 Результати верифікації предиктивного модуля 68
4.3.2 Оцінка ефективності сценарного управління та чутливості до типу підрозділу 70
4.4 Напрями подальшого розвитку та вдосконалення системи 71
Висновки до розділу 4 72
ВИСНОВКИ 75
ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ 77
ДОДАТОК А Код файлу app.py 82
ДОДАТОК Б Код файлу data_generator.py 87
ДОДАТОК В Код файлу train_model.py 89

СКОРОЧЕННЯ ТА УМОВНІ ПОЗНАКИ

ЗСУ – збройні сили україни

ПЗ – програмне забезпечення

СППР – система підтримки прийняття рішень

AI – Artificial Intelligence

API – Application Programming Interface

CSV – Comma-Separated Values

DL – Deep Learning

GRU – Gated Recurrent Unit

GUI – Graphical User Interface

HDF5 – Hierarchical Data Format version 5

MCDM – Multi-Criteria Decision Making

ML – Machine Learning

MSE – Mean Squared Error

RNN – Recurrent Neural Network

SAW – Simple Additive Weighting

SME – Subject Matter Expert

URL – Uniform Resource Locator

ВСТУП

Збройні конфлікти сучасності характеризуються високою динамікою, невизначеністю оперативної обстановки та значною залежністю від ефективного логістичного забезпечення [1, 3, 4]. Здатність своєчасно та безпечно доставити ресурси (пальне, боєприпаси, медикаменти) до підрозділів на передовій часто стає вирішальним фактором успіху військової операції. Проте, планування логістичних маршрутів у зоні бойових дій ускладнюється низкою факторів: постійною зміною рівня загрози, впливом погодних умов, мінливістю споживання ресурсів та необхідністю врахування експертних оцінок командирів.

Існуючі підходи до логістичного планування, які часто базуються на детермінованих моделях або інтуїтивному досвіді, не завжди здатні адекватно реагувати на ці виклики. Статичні плани швидко застарівають, а прийняття рішень під тиском часу та неповної інформації може призвести до вибору неоптимальних або небезпечних маршрутів, що ставить під загрозу виконання місії та життя особового складу. Провідні військові аналітики та дослідники у сфері операційного дослідження активно працюють над впровадженням новітніх інформаційних технологій, зокрема штучного інтелекту та методів підтримки прийняття рішень, для підвищення ефективності військової логістики.

Актуальність теми зумовлена критичною необхідністю забезпечення боєздатності підрозділів в умовах сучасних конфліктів, що вимагає впровадження адаптивних інструментів для оперативного планування та мінімізації ризиків при постачанні ресурсів.

Дана робота пов’язана з актуальними науковими дослідженнями у галузі застосування методів штучного інтелекту та дослідження операцій у військовій сфері, зокрема з використанням глибокого навчання для прогнозування та методів нечіткої логіки для багатокритеріальної оптимізації. Результати цієї роботи можуть бути використані для вдосконалення існуючих систем управління логістикою та розробки нових рішень для автоматизації процесів планування у ЗСУ.

Метою роботи є підвищення ефективності та адаптивності логістичного планування шляхом розробки гібридної СППР, яка інтегрує динамічні прогнози потреб у ресурсах з механізмами багатокритеріального прийняття рішень.

Об’єктом дослідження є процеси прогнозування споживання логістичних ресурсів та оптимізації тактичних маршрутів пересування у військових умовах.

Предмет дослідження – методи та алгоритми глибокого навчання для прогнозування часових рядів, а також гібридні методи багатокритеріальної оптимізації маршрутів.

У процесі розробки гібридної системи використовувались сучасні інструментальні засоби та платформи. Мова програмування Python застосована для реалізації моделей глибокого на вчання (за допомогою бібліотек TensorFlow/Keras) та алгоритмів нечіткої логіки (scikit-fuzzy), бібліотека pandas – для обробки даних, а фреймворк Streamlit – для створення інтерактивного користувацького інтерфейсу системи. Середовище розробки PyCharm забезпечило зручну реалізацію та налагодження проєкту.

Отже, дана робота спрямована на розробку інноваційної гібридної системи підтримки прийняття рішень, що є важливим кроком у підвищенні ефективності та безпеки військової логістики шляхом інтеграції сучасних методів штучного інтелекту та експертних знань.

1 АНАЛІЗ ПРОБЛЕМИ ТА ПОСТАНОВКА ЗАДАЧІ

1.1 Сучасний стан проблеми управління військовою логістикою

Сучасна військова логістика є складною, багатогранною та критично важливою дисципліною, що еволюціонувала далеко за межі традиційного уявлення про просте переміщення матеріальних засобів [4], [5]. Її фундаментальна мета, що історично описується класичною формулою «7R» (доставка потрібного продукту, в потрібний час, у потрібній кількості та стані, у правильне місце, для правильного споживача та за адекватною ціною), залишається незмінною, проте умови її досягнення кардинально змінилися [2].

Сучасні операційні середовища вносять значні корективи, що визначаються формулою «4D» (див. рис. 1.1): Demand (Попит), Distance (Відстань), Destination (Пункт призначення) та Duration (Тривалість). Кожен із цих факторів в умовах сучасного конфлікту набуває екстремальної невизначеності: попит на ресурси може зростати експоненційно та непередбачувано; відстані можуть динамічно змінюватися через руйнування інфраструктури; пункти призначення можуть ставати недоступними; а тривалість операцій є апріорі невідомою.

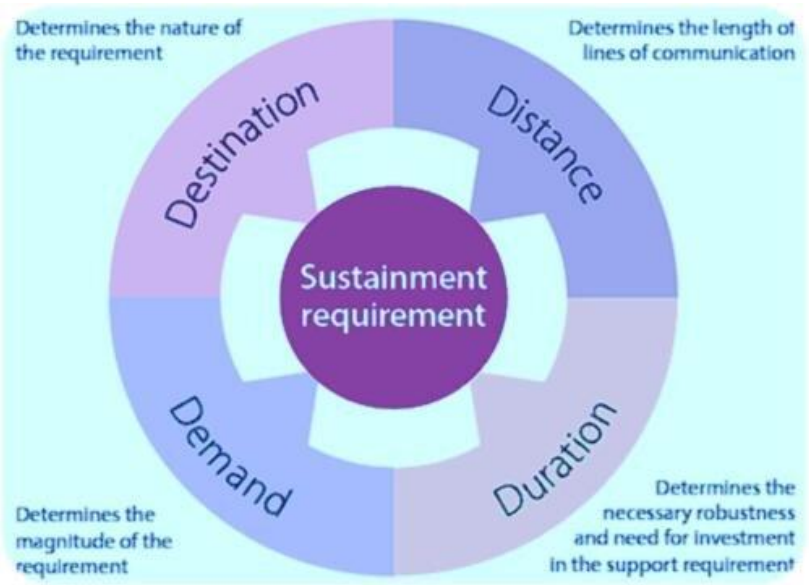

Рисунок 1.1 – Формула «4D»

Найважливішим фактором, що спричинив парадигмальний зсув у військовій логістиці, є концепція оспорюваної логістики (див. рис. 1.2). Цей термін означає, що логістичні операції проводяться в середовищі, де супротивник активно, цілеспрямовано та багатодоменно намагається порушити ланцюги постачання. Арсенал засобів протидії є надзвичайно широким: від фізичних атак на конвої, склади, порти та аеродроми до кібератак [5, 6] на системи управління логістикою, поширення дезінформації для створення хаосу та застосування засобів радіоелектронної боротьби для порушення зв'язку та навігації.

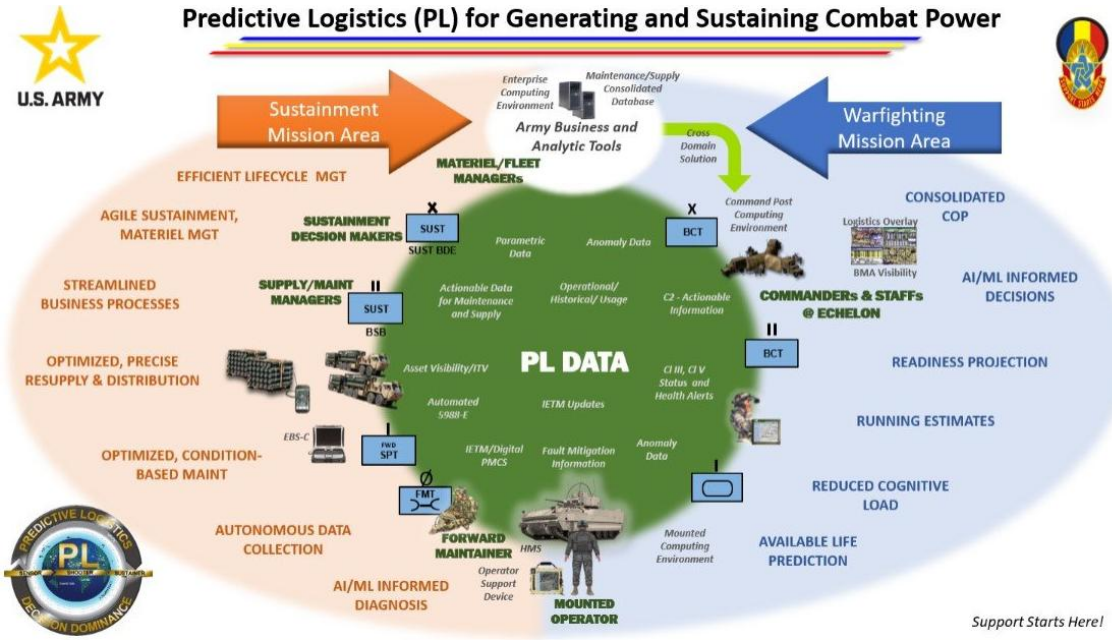

Рисунок 1.2 – Концепція оспорюваної логістики

У таких умовах логістичні практики, запозичені з комерційного сектору та оптимізовані виключно для економії коштів та часу за моделлю «just-in-time», стають не просто неефективними, а й надзвичайно вразливими. Навіть незначні збої в такій «худій» системі можуть викликати каскадний ефект, що призводить до критичних збоїв у постачанні та ставить під загрозу виконання бойових завдань.

Ця нова реальність змушує військові структури кардинально переглянути свої пріоритети, здійснюючи перехід від моделі, орієнтованої на ефективність

(мінімізацію витрат), до моделі, що пріоритезує стійкість (resilience), адаптивність та результативність (effectiveness). Стійкість означає здатність системи продовжувати функціонувати, попри атаки та збої, швидко відновлюватися та знаходити альтернативні шляхи. Адаптивність – це здатність динамічно перебудовувати логістичні плани у відповідь на зміни оперативної обстановки в реальному часі.

Основна проблема полягає в тому, що виклики «4D» не є ізольованими змінними; вони утворюють тісно пов'язану, нелінійну динамічну систему. Наприклад, кібератака на систему управління портом (вплив на «Destination») змушує перенаправляти вантажі, що збільшує «Distance» та «Duration». Це, у свою чергу, призводить до непередбачуваного сплеску «Demand» на паливо та запчастини, створюючи додаткове навантаження на вже порушену мережу та ризик каскадного збою. Отже, ключова проблема полягає не в статичній оптимізації окремих параметрів, а в управлінні складною, динамічною системою під постійним тиском супротивника.

Це обґрунтовує нагальну необхідність застосування сучасних інструментів штучного інтелекту, зокрема рекурентних нейронних мереж, які за своєю природою створені для моделювання залежних від часу динамічних процесів, та систем підтримки прийняття рішень, здатних обробляти інформацію швидше за людину.

1.2 Застосування штучного інтелекту для підтримки прийняття рішень

В умовах зростаючої складності, динамічності та інформаційної насиченості сучасних бойових дій штучний інтелект (ШІ) перетворюється з експериментальної технології на ключовий інструмент для досягнення та утримання переваги. ШІ розглядається не як заміна людини-командира, а як незамінний засіб-мультиплікатор сили, що дозволяє досягти «інформаційного домінування» та кардинально прискорити цикл прийняття рішень (OODA loop: Observe, Orient, Decide, Act) (див. рис. 1.3) [47]. Інтеграція ШІ у військову логістику може кардинально змінити підходи до управління ланцюгами постачання, оптимізувати розподіл обмежених ресурсів

та підвищити якість і своєчасність рішень, що приймаються на всіх рівнях – від стратегічного до тактичного.

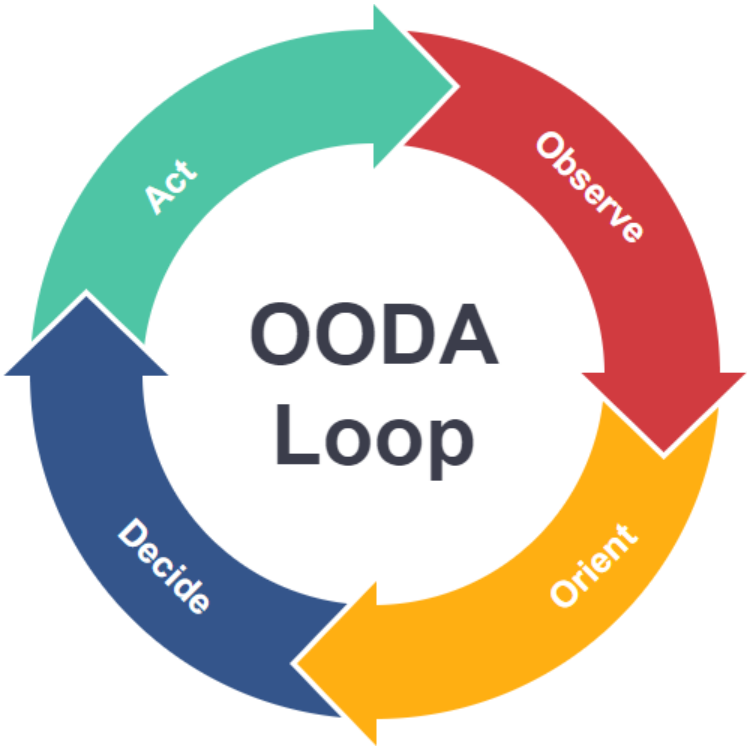

Рисунок 1.3 – Цикл прийняття рішень

Однією з ключових переваг ШІ є його здатність аналізувати величезні обсяги різнорідних даних (big data) для прогнозування майбутніх тенденцій та потреб, що значно підвищує прозорість та проактивність ланцюга постачання. Наприклад, предиктивна аналітика на основі даних з IoT-сенсорів може визначати, коли деталі транспортних засобів потребуватимуть заміни, дозволяючи проводити проактивне технічне обслуговування ще до виникнення несправності в польових умовах. Це призводить до значної економії коштів, підвищення операційної готовності техніки та зменшення ймовірності незапланованих простоїв.

Сучасні системи підтримки прийняття рішень (СППР) на базі ШІ виходять за рамки простого відображення даних. Вони пропонують інструменти предиктивного моделювання та аналізу варіантів дій (Course of Action, COA), що дозволяє

командирам та штабним офіцерам «програвати» різні сценарії в цифровому середовищі та обирати оптимальні рішення з урахуванням безлічі факторів.

Особливо перспективним напрямом є розвиток нейросимвольного ШІ (див. рис. 1.4) [7, 49, 50], який поєднує сильні сторони двох парадигм: здатність нейронних мереж до розпізнавання складних, нелінійних патернів у сирих даних (субсимвольний підхід) та можливості символьних систем до логічних міркувань на основі формальних правил та знань (символьний підхід). Пропонована в цій роботі гібридна модель (GRU + нечітка логіка) є практичною реалізацією саме цієї концепції. Модель GRU виступає як «нейронний» компонент: вона навчається на великих масивах історичних логістичних даних для виявлення складних часових закономірностей споживання ресурсів – завдання, з яким глибоке навчання справляється найкраще. З іншого боку, модель на основі нечіткої логіки є «символьним» компонентом: вона оперує набором чітко визначених, інтерпретованих людиною правил та лінгвістичних змінних (наприклад, «ЯКЩО рівень загрози ВИСОКИЙ І час у дорозі ДОВГИЙ, ТОДІ пріоритет маршруту НИЗЬКИЙ»), що є формою символьних міркувань.

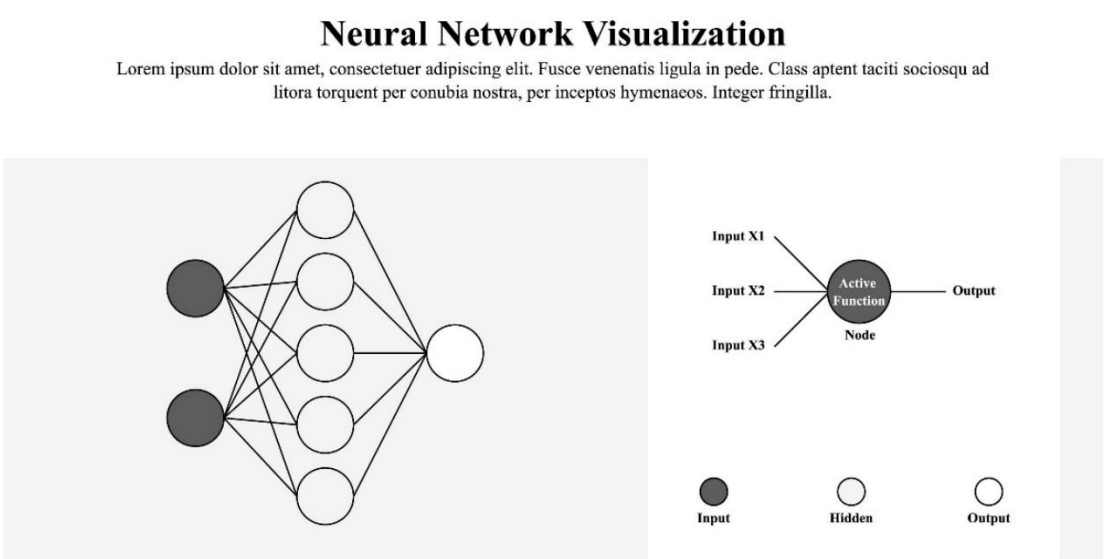

Рисунок 1.4 – Концепція нейросимвольного ШІ

Такий гібридний підхід дозволяє створити міст між непрозорим, але точним числовим прогнозом нейронної мережі та прозорою, обґрунтованою рекомендацією, зрозумілою для людини. У військових системах, де довіра до рекомендацій ШІ та можливість їх пояснення (Explainable AI, XAI) мають вирішальне значення, така архітектура є не просто технічним рішенням, а необхідною умовою для практичного впровадження.

1.3 Огляд останніх публікацій та існуючих систем

Аналіз сучасних наукових досліджень та існуючих розробок підтверджує високу актуальність застосування методів штучного інтелекту для вирішення складних логістичних задач, особливо в умовах невизначеності. Хоча прямих аналогів інтегрованої системи, що поєднує саме GRU та Fuzzy SAW для військової логістики, в оглянутій літературі виявлено не було, аналіз суміжних галузей дозволяє виділити ключові тренди та обґрунтувати обраний у даній роботі підхід.

У сфері прогнозування попиту в ланцюгах постачання рекурентні нейронні мережі (RNN), зокрема архітектури з вентильними механізмами, як-от GRU та LSTM, демонструють значні переваги над класичними статистичними методами (ARIMA, експоненційне згладжування). Дослідження I. J. та ін. (2023)[10], присвячене багатовимірному прогнозуванню продажів, показало, що модель GRU перевершила моделі VAR та LightGBM, досягнувши менших значень помилок MAE та MAPE завдяки своїй здатності автоматично вивчати приховані патерни, тренди та сезонність у даних. Робота Jahin та ін. (2024) [8] представляє гібридну архітектуру глибокого навчання, що також використовує GRU та LSTM для підвищення точності прогнозування попиту шляхом інтеграції кількох каналів даних. Автори Khan та ін. (2024) [9] у своєму огляді 119 наукових праць за період 2015-2024 рр. роблять висновок, що ШІ-орієнтовані моделі прогнозування є значно точнішими та більш динамічними порівняно з традиційними підходами, що є критично важливим для

сучасних волатильних ринків (див. рис. 1.5). Ці роботи підтверджують, що вибір GRU для прогнозування споживання ресурсів є обґрунтованим та відповідає передовим практикам у галузі.

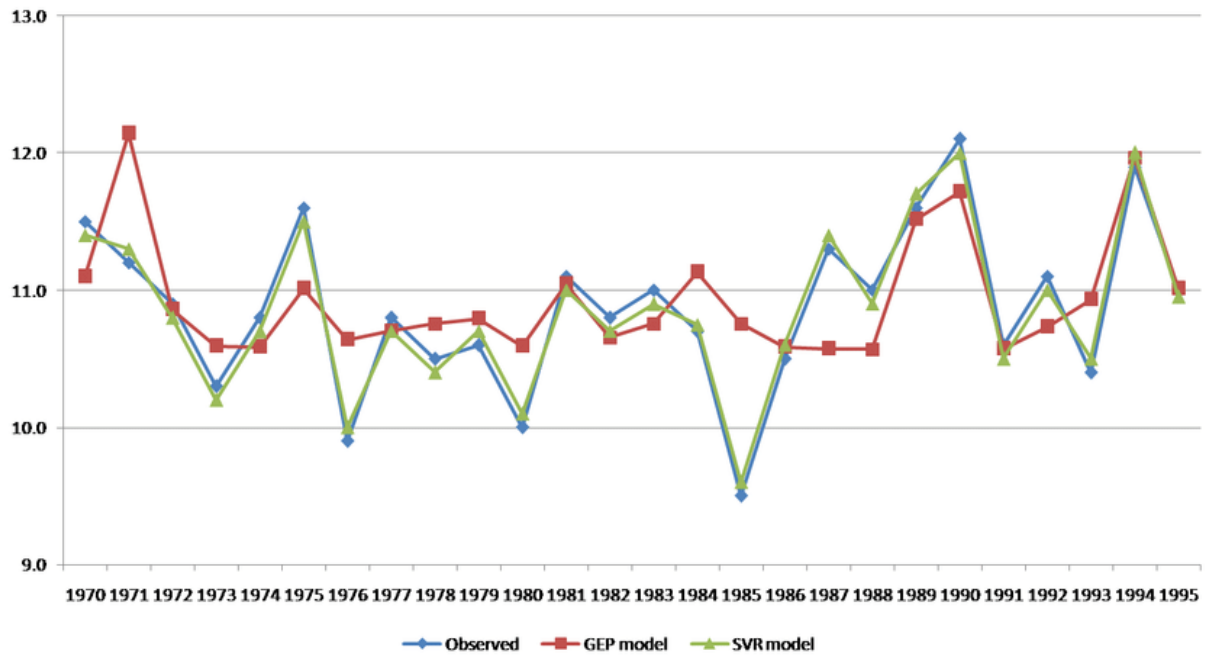

Рисунок 1.5 – Приклад ефективності прогнозування часових рядів за допомогою нейронних мереж

У контексті оптимізації маршрутів в умовах невизначеності активно застосовується теорія нечітких множин (див. рис. 1.6). У роботі Dadashkarimi (2025)[11] пропонується фреймворк на основі нечіткого лінійного програмування (FLP) для оптимізації військової медичної логістики (транспортування, евакуація), де вхідні дані, такі як кількість поранених чи наявність ресурсів, є неточними та задаються у вигляді нечітких чисел. Результати показують, що FLP перевершує класичні та стохастичні методи на 12-15% за корисністю та на 20-25% за доцільністю рішень, що доводить ефективність нечіткої логіки для моделювання невизначеності у військових задачах. Дослідження Zarrabi & Jesri (2025)[12] також використовує нечітку логіку для оцінки ефективності транспортних моделей у фармацевтичній промисловості, де умови також є невизначеними. Ці роботи підтверджують, що для оцінки

маршрутів за такими якісними критеріями, як «рівень загрози», застосування нечіткої логіки є адекватним та потужним інструментом.

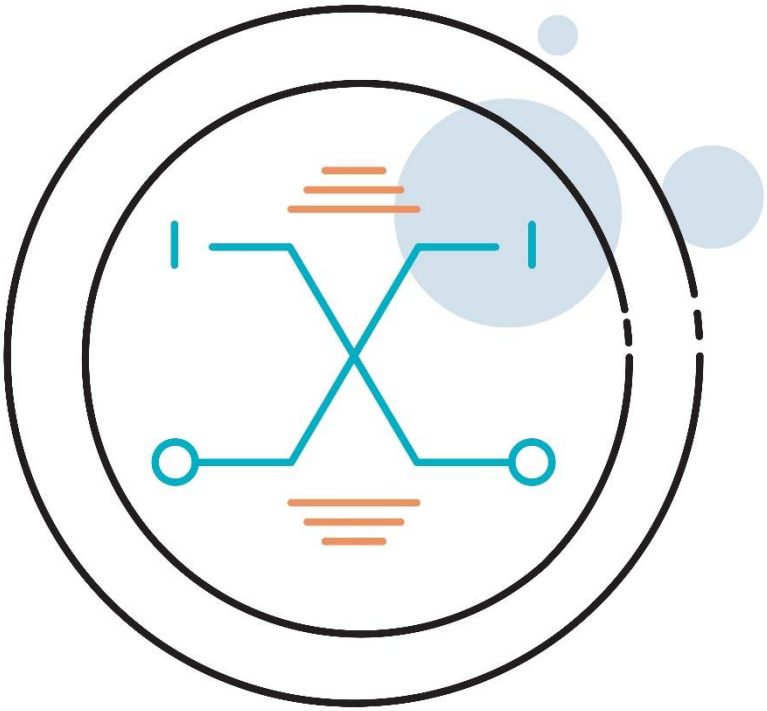

Рисунок 1.6 – Функція належності

Гібридні моделі, що поєднують різні підходи ШІ, розглядаються як найбільш перспективний напрям для створення інтелектуальних СППР у логістиці. У статті De-Arteaga та ін. (2025) [13] представлена концепція Interpretable Neural System Dynamics (INSD), що поєднує глибоке навчання з методами причинно-наслідкового машинного навчання (CML) та нейросимвольного ШІ для створення прозорих та надійних моделей для управління логістичними терміналами. Цей підхід, як і наш, наголошує на критичній важливості інтерпретованості рішень для побудови довіри з боку операторів.

Таблиця 1.1 – Аналіз останніх досліджень у сфері інтелектуальної логістики

Метод / модель	Опис методу / моделі та застосування
GRU	Багатовимірне прогнозування продажів у ланцюгах постачання. Перевершує класичні моделі VAR та LightGBM.
Гібридна DL (CNN, LSTM, GRU)	Прогнозування попиту в ланцюгах постачання шляхом інтеграції кількох каналів даних для підвищення точності.
Нечітке лінійне програмування (FLP)	Оптимізація військової медичної логістики (евакуація, ресурси) в умовах невизначеності вхідних даних.
Нечітка логіка	Оцінка ефективності транспортних моделей для фармацевтичної промисловості в умовах невизначеності.
Нейросимвольний ШІ (INSD)	Гібридна модель для створення інтерпретованих СППР в управлінні логістичними терміналами, що поєднує DL та CML.
Навчання з підкріпленням (RL)	Оптимізація багатоешелонних ланцюгів постачання в умовах стохастичного попиту та збоїв.

Проведений огляд літератури дозволяє ідентифікувати наукову прогалину: хоча окремі дослідження фокусуються або на прогнозуванні за допомогою нейронних мереж, або на оптимізації за допомогою нечіткої логіки, бракує робіт, що представляють інтегровану, наскрізну систему [42], яка б поєднувала обидва підходи в єдиному контурі прийняття рішень спеціально для задач військової логістики в оспорюваному середовищі. Дана робота спрямована на заповнення цієї прогалини.

1.4 Постановка задачі дослідження

На основі проведеного аналізу актуальності проблеми, огляду сучасних наукових підходів та виявленої наукової прогалини, формулюється комплексна задача дослідження. Вона полягає у розробці, програмній реалізації та експериментальній валідації гібридної інтелектуальної системи підтримки прийняття рішень для військової логістики, яка б інтегрувала предиктивні та прескриптивні можливості

штучного інтелекту. Для досягнення цієї мети необхідно послідовно вирішити низку конкретних завдань:

– створення точної та надійної моделі для прогнозування динамічного споживання ключових ресурсів, зокрема пального. Для цього необхідно спроєктувати, навчити та оцінити рекурентну нейронну мережу архітектури GRU. Модель повинна бути здатною прогнозувати щоденне споживання на основі аналізу часових рядів, що включають не лише історичні дані про споживання, але й зовнішні (екзогенні) фактори, такі як операційний темп підрозділу, погодні умови та сезонність. Виконання цього завдання вимагає ретельної попередньої обробки даних, включаючи їх очищення, нормалізацію, кодування категоріальних ознак та перетворення у формат послідовностей, придатний для навчання RNN;

– реалізація механізму прийняття рішень, здатного обирати найкращий логістичний маршрут з набору альтернатив на основі кількох, часто суперечливих, критеріїв. Для цього необхідно розробити багатокритеріальний модуль на основі методу нечіткого простого адитивного зважування (Fuzzy SAW). Цей модуль повинен оцінювати та ранжувати маршрути за трьома основними критеріями: вартість (яка динамічно розраховується на основі прогнозованого споживання пального), рівень загрози (якісна оцінка, отримана з розвідувальних даних) та час у дорозі (кількісний параметр). Це завдання включає визначення лінгвістичних змінних, побудову функцій належності для кожного критерію та реалізацію алгоритму зваженої агрегації;

– синергетична інтеграція в єдиний програмний прототип. Необхідно створити наскрізний конвеєр обробки даних (pipeline), де числовий прогноз, згенерований моделлю GRU, автоматично та динамічно використовується як кількісний вхідний параметр для моделі Fuzzy SAW. Ця інтеграція є ядром гібридного підходу і повинна забезпечувати логічний зв'язок між предиктивною аналітикою («що нам знадобиться?») та прескриптивною аналітикою («який найкращий спосіб це доставити?»);

– проведення всебічного тестування розробленої системи для підтвердження її адекватності, адаптивності та практичної значущості. Оскільки для багатокритеріальної задачі не існує єдиної «правильної» відповіді, валідація повинна проводитися за допомогою сценарного моделювання. Необхідно розробити та протестувати щонайменше три реалістичні сценарії, що імітують різні операційні пріоритети: «Бойові дії» (пріоритет безпеки), «Гуманітарна місія» (пріоритет швидкості) та «Рутинна логістика» (збалансований підхід). Аналіз результатів для кожного сценарію дозволить оцінити, наскільки гнучко та логічно система реагує на зміну вагових коефіцієнтів та чи здатна вона знаходити робастні, обґрунтовані рішення.

Висновки до розділу 1

У даному розділі було проведено глибокий та всебічний аналіз проблеми управління військовою логістикою в сучасних умовах, що характеризуються високою невизначеністю, динамічністю та активною протидією супротивника. Було обґрунтовано нагальну необхідність переходу від традиційних реактивних підходів до проактивних, інтелектуальних систем підтримки прийняття рішень, здатних підвищити стійкість та адаптивність логістичних ланцюгів.

Проведений розгорнутий огляд останніх наукових публікацій та існуючих аналогів у суміжних галузях показав, що методи штучного інтелекту, зокрема глибоке навчання (GRU, LSTM) та нечітка логіка, є ефективними та валідними інструментами для вирішення складних задач прогнозування та оптимізації в умовах невизначеності. Водночас було виявлено наукову прогалину, що полягає у відсутності досліджень, присвячених розробці інтегрованої, наскрізної системи, яка б синергетично поєднувала обидва підходи спеціально для задач військової логістики в оспорюваному середовищі.

На основі цього було сформульовано чітку та структуровану постановку задачі дослідження. Вона передбачає послідовну розробку, інтеграцію та експериментальну валідацію гібридної системи, що складається з модуля прогнозування на

основі GRU та модуля багатокритеріальної оптимізації на основі Fuzzy SAW. Такий підхід дозволяє створити інтелектуальну систему, здатну автоматично генерувати обґрунтовані рекомендації, поєднуючи предиктивну точність нейронних мереж та інтерпретованість нечіткої логіки, що є критично важливим для побудови довіри до системи з боку кінцевих користувачів. Таким чином, даний розділ закладає міцний теоретичний та методологічний фундамент для подальшої розробки та практичної реалізації дослідження.

2 ТЕОРЕТИЧНІ ОСНОВИ ГІБРИДНОЇ СИСТЕМИ

2.1 Прогнозування часових рядів за допомогою рекурентних мереж GRU

Рекурентні нейронні мережі (RNN) – це клас нейронних мереж [15, 21], спеціально розроблених для обробки послідовних даних, таких як текст, мова та часові ряди, де порядок елементів має вирішальне значення (див. рис. 2.1). На відміну від мереж прямого поширення, які обробляють входи незалежно, RNN використовують рекурентні зв'язки, де вихід нейрона на одному часовому кроці подається назад як вхід для мережі на наступному кроці. Це дозволяє RNN фіксувати часові залежності та патерни в послідовностях. Фундаментальним будівельним блоком RNN є рекурентний блок, який підтримує прихований стан – форму пам'яті, що оновлюється на кожному часовому кроці на основі поточного входу та попереднього прихованого стану.

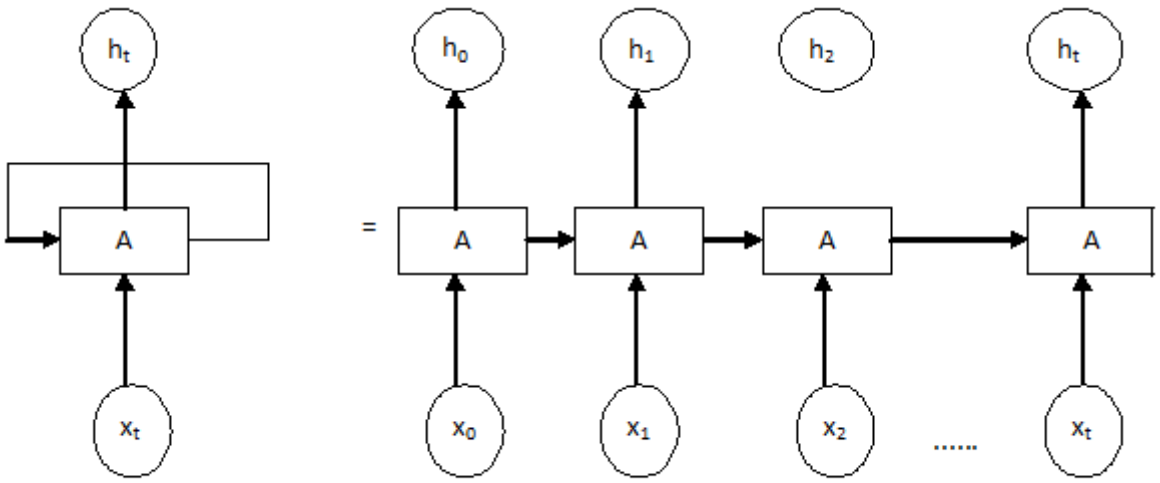

Рисунок 2.1 – Розгорнута схема RNN

Однак прості RNN страждають від проблеми «згасаючого градієнта» [16, 45], коли під час навчання градієнти стають настільки малими, що мережа перестає ефективно навчатися на довгострокових залежностях. Для вирішення цієї проблеми

були розроблені більш складні архітектури, такі як Long Short-Term Memory (LSTM) (див. рис. 2.2) та Gated Recurrent Unit (GRU).

GRU, представлена у 2014 році [17, 18], є вдосконаленням стандартної RNN, яка використовує вентильні механізми для контролю потоку інформації. Архітектура GRU-блоку є простішою, ніж у LSTM, і складається з двох основних вентилів: вентиля скидання (Reset Gate) та вентиля оновлення (Update Gate). Архітектура GRU-блоку складається з двох основних вентилів(див. рис. 2.3):

– вентиль скидання (Reset Gate): визначає, яка частина інформації з попереднього прихованого стану має бути «забута» або проігнорована. Це дозволяє моделі фокусуватися на релевантній інформації для поточного прогнозу;

– вентиль оновлення (Update Gate): контролює, яка частина інформації з попереднього стану має бути перенесена в поточний, а яка частина нової, щойно обчисленої інформації, має бути додана. Це ключовий механізм для збереження довгострокових залежностей.

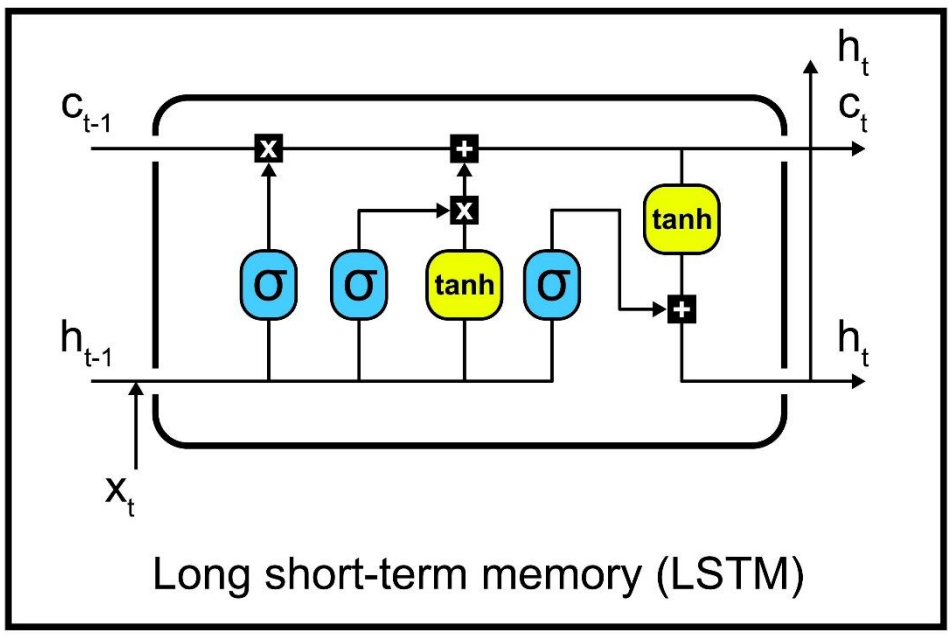

Рисунок 2.2 – Архітектура блоку LSTM

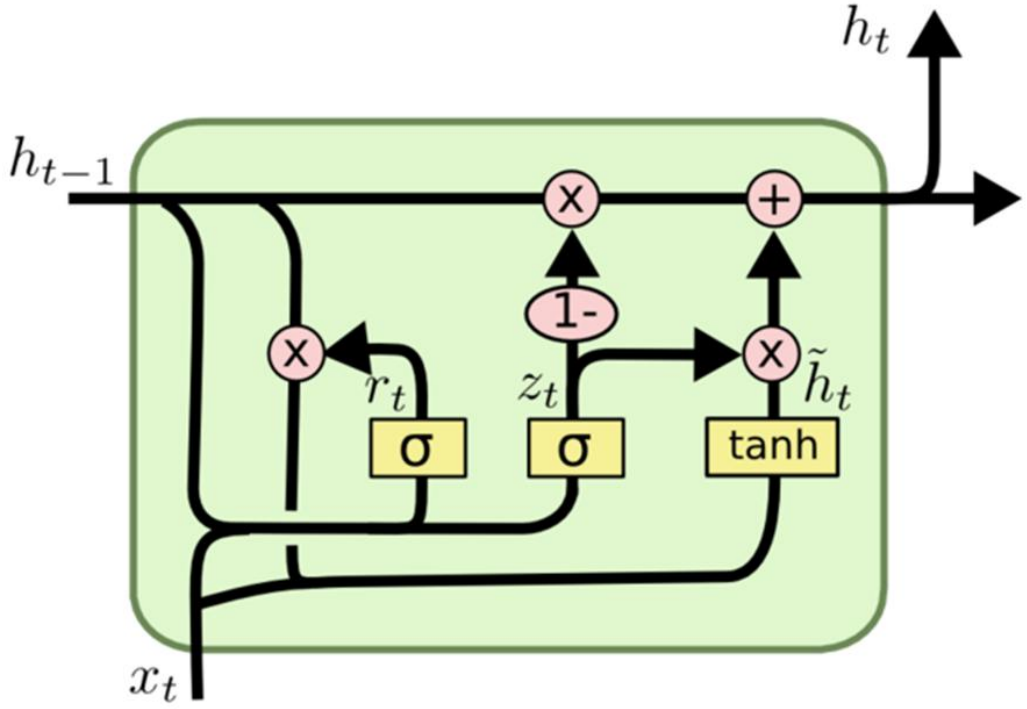

Рисунок 2.3 – Архітектура блоку GRU

Таблиця 2.1 – Порівняльний аналіз архітектур LSTM та GRU

Архітектурний компонент	**Деталі LSTM**	**Деталі GRU**	**Наслідки**
Кількість вентилів	3 вентилі: вхідний (Input), забування (Forget), вихідний (Output)	2 вентилі: скидання (Reset), оновлення (Update)	GRU має простішу структуру, що призводить до меншої кількості обчислень на кожному кроці.
Вектори стану	2 вектори: стан комірки (Cell State, c_t), прихований стан (Hidden State, h_t)	1 вектор: прихований стан (Hidden State, h_t)	GRU є менш вимогливою до пам'яті, оскільки не підтримує окремий стан комірки.
Кількість параметрів	Більша кількість матриць ваг та зміщень для трьох вентилів та стану комірки.	Менша кількість параметрів завдяки об'єднанню вентилів та відсутності стану комірки.	GRU навчається швидше і має менший ризик перенавчання на невеликих наборах даних.
Контроль пам'яті	Більш тонкий контроль над потоком інформації завдяки окремим вентилям для забування та додавання.	Вентиль оновлення одночасно контролює, що забути і що додати.	LSTM теоретично може краще моделювати дуже складні та довгострокові залежності.
Обчислювальна складність	Вища	Нижча	GRU є кращим вибором для застосувань з обмеженими ресурсами або коли потрібна швидка ітерація.

Вибір GRU замість більш складної архітектури LSTM для цієї роботи є не просто технічним уподобанням, а стратегічним рішенням, що відповідає реаліям військового застосування. Дослідження показують, що GRU часто досягає порівнянної з LSTM продуктивності [19, 20], але має простішу архітектуру та меншу кількість параметрів, що робить її обчислювально більш ефективною. Військові операції все частіше покладаються на децентралізоване командування та обробку даних на «тактичному краю», де обчислювальні ресурси та мережеве з'єднання можуть бути обмеженими або ненадійними.

Модель, яка швидше навчається і вимагає менше ресурсів, краще підходить для розгортання на захищених ноутбуках, вбудованих системах у транспортних засобах або на серверах передових оперативних баз. Здатність швидко перенавчати або доналаштовувати модель у польових умовах з використанням нових локальних

даних є значною тактичною перевагою, що дозволяє логістичній системі адаптуватися до умов, що швидко змінюються. Таким чином, технічна характеристика «ефективність» безпосередньо перетворюється на операційні атрибути «розгортаємість» та «адаптивність», роблячи GRU кращим вибором для цієї конкретної сфери застосування.

2.2 Багатокритеріальне прийняття рішень в умовах невизначеності за допомогою нечіткої логіки

Задача вибору оптимального логістичного маршруту за своєю суттю є класичною задачею багатокритеріального прийняття рішень (Multi-Criteria Decision Making, MCDM). Планувальник повинен одночасно враховувати декілька, часто суперечливих, критеріїв: мінімізувати час доставки, зменшити ризики (рівень загрози) та оптимізувати витрати (споживання пального). У реальних задачах, особливо у військовій сфері, багато з цих критеріїв є нечіткими, якісними та суб'єктивними. Наприклад, «рівень загрози» не є точною детермінованою величиною; це експертна оцінка, що краще описується лінгвістичними термінами, як-от «низький», «середній» або «високий». Аналогічно, «прохідність місцевості» або «погодні умови» також є нечіткими поняттями. Класичні методи оптимізації, що вимагають точних числових вхідних даних, виявляються неефективними в таких умовах.

Для роботи з такою іманентною невизначеністю та неточністю, властивою задачам з людськими судженнями, було розроблено теорію нечітких множин, запропоновану Лотфі Заде [22, 23]. На відміну від класичної (чіткої) логіки, де елемент або належить до множини, або ні, у нечіткій логіці елемент може належати до множини з певним ступенем належності, що варіюється від 0 до 1 (див. рис. 2.4). Це дозволяє математично оперувати з лінгвістичними змінними та нечіткими поняттями. У даній роботі для вирішення задачі оптимізації маршруту було обрано метод нечіткого простого адитивного зважування (Fuzzy Simple Additive Weighting,

SAW) [25, 26, 28], який поєднує класичний метод SAW з апаратом нечіткої логіки для обробки неточних даних.

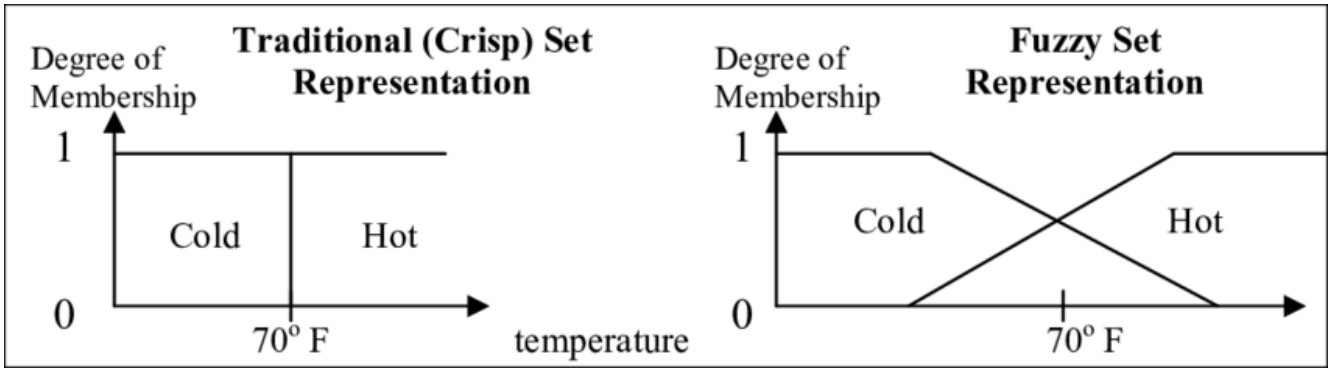

Рисунок 2.4 – Порівняння Чітка логіка vs Нечітка логіка

Процедура Fuzzy SAW складається з наступних логічних кроків (див. рис. 2.5):

– визначення критеріїв та альтернатив. Ідентифікуються фактори, що впливають на вибір маршруту (критерії), та набір можливих маршрутів (альтернативи). У нашому випадку критеріями є: вартість пального (на основі прогнозу GRU), час у дорозі та рівень загрози;

– визначення лінгвістичних змінних та функцій належності. Для кожного критерію визначаються лінгвістичні терми та відповідні їм функції належності. Наприклад, для критерію «Рівень загрози» (універсум від 0 до 100) можна визначити терми «Низький», «Середній» та «Високий» з трикутними або трапецієподібними функціями належності. Функція належності визначає ступінь, до якого певне чітке значення (наприклад, рівень загрози 85) належить до нечіткої множини (наприклад, на 0.7 належить до «Високий» і на 0.3 – до «Середній»);

– фазифікація вхідних даних. Чіткі вхідні значення для кожного маршруту за кожним критерієм перетворюються на ступені належності до відповідних лінгвістичних термів;

– агрегація та застосування ваг. Для кожної альтернативи обчислюється загальна оцінка. У методі SAW це реалізується шляхом агрегування зважених оцінок за кожним критерієм. Ваги критеріїв (наприклад, {'загроза': 0.6, 'пальне': 0.2, 'час': 0.2}) є ключовим елементом, що дозволяє адаптувати систему до різних операційних сценаріїв. У бойових умовах вага «загрози» буде найвищою, тоді як у гуманітарній місії пріоритет може бути відданий «часу»;

– ранжування альтернатив. Фінальні агреговані оцінки для кожного маршруту дозволяють їх однозначно ранжувати від найкращого до найгіршого.

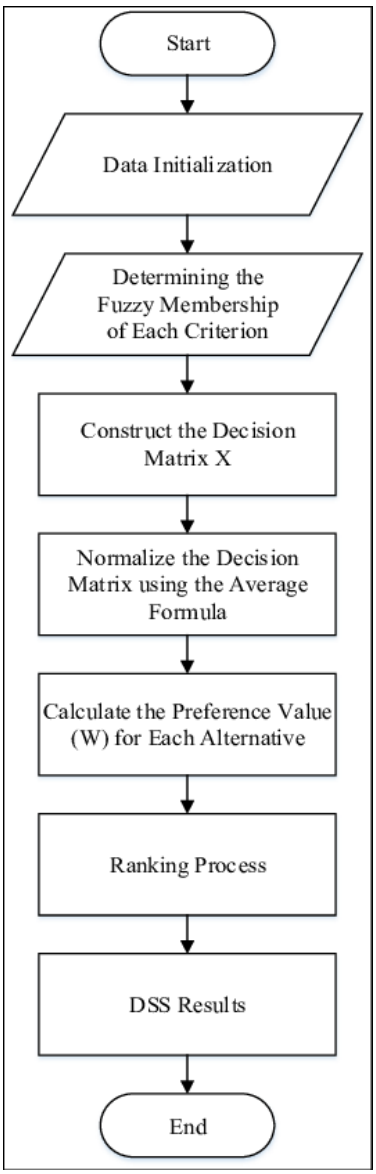

Рисунок 2.5 – Блок-схема алгоритму Fuzzy SAW

Вибір Fuzzy SAW для цієї системи є не лише технічним, але й концептуальним. Цей компонент слугує природним мостом між «чорною скринькою» нейронної мережі та потребою кінцевого користувача в прозорих, обґрунтованих рекомендаціях. Нейронні мережі, такі як GRU, є надзвичайно потужними, але часто їм бракує інтерпретованості; важко зрозуміти, чому модель зробила конкретний прогноз.

У військових системах ШІ прозорість, відстежуваність та пояснюваність (Explainable AI, XAI) мають вирішальне значення для побудови довіри та забезпечення відповідального використання (див. рис. 2.6).

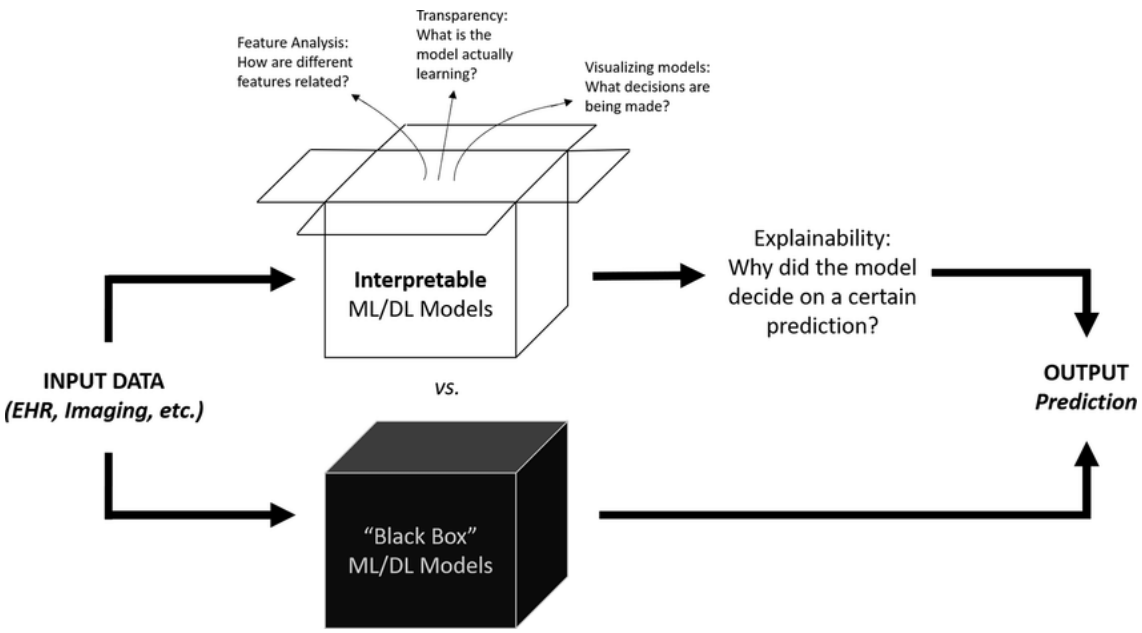

Рисунок 2.6 – Концепція Explainable AI (XAI)

На відміну від GRU, модель Fuzzy SAW є високопрозорою. Її правила, критерії та ваги явно визначаються експертами або на основі доктринальних документів. Остаточне ранжування можна відстежити через розрахунки, щоб побачити, як кожен критерій вплинув на результат. Обробляючи непрозорий числовий вихід GRU через прозору нечітку логічну структуру, інтегрована система надає

рекомендацію, яка є не тільки керованою даними, але й пояснюваною. Командир може побачити, що «Маршрут А» був обраний тому, що, незважаючи на більшу довжину, його «Низький» рейтинг загрози переважив «Середнє» споживання палива, спрогнозоване GRU. Це підвищує довіру до системи, сприяє її впровадженню та забезпечує відповідальне прийняття рішень.

2.3 Архітектура інтегрованої гібридної системи

Ключовою інновацією даної роботи є не окреме застосування методів глибокого навчання чи нечіткої логіки, а їхня синергетична інтеграція в єдину гібридну архітектуру (див. рис. 2.7), що створює наскрізний контур підтримки прийняття рішень.

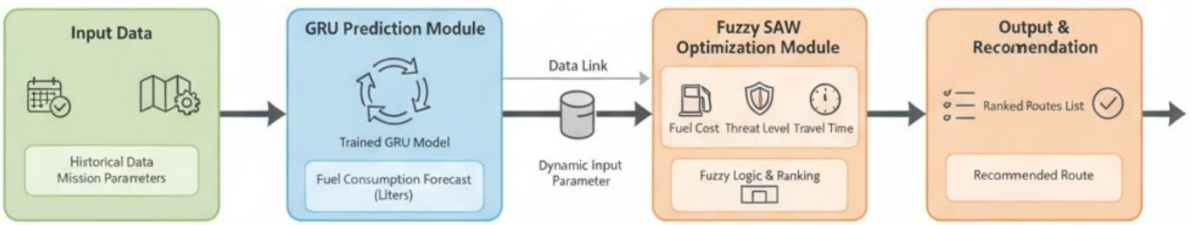

Рисунок 2.7 – Архітектура системи

Ця структура поєднує предиктивні можливості глибокого навчання, здатного виявляти складні патерни в історичних даних, з прескриптивною (рекомендаційною) силою нечіткої логіки, яка дозволяє оперувати з нечіткими критеріями та експертними знаннями [27, 30]. Така архітектура є практичною реалізацією концепції нейросимвольного ШІ, де нейронна мережа виконує субсимвольні обчислення, а нечітка система – символьні міркування.

Концептуальний робочий процес системи реалізовано як послідовний конвеєр (pipeline), що складається з чотирьох основних етапів:

– етап вхідних даних та ініціалізації. На цьому етапі система отримує два типи вхідних даних. По-перше, це параметри поточної місії, що включають набір

альтернативних маршрутів з їх характеристиками (орієнтовний час у дорозі, дані розвідки про рівень загрози) та операційний сценарій, який визначає вагові коефіцієнти для критеріїв оптимізації (наприклад, пріоритет безпеки чи швидкості). По-друге, це історичні дані про споживання ресурсів за останній період (наприклад, 30 днів), які необхідні для роботи модуля прогнозування. Ці дані включають щоденне споживання пального, операційний темп, погодні умови та інші релевантні ознаки;

– етап прогнозування (Модуль GRU). На цьому етапі в дію вступає попередньо навчена модель GRU. Вона приймає на вхід послідовність історичних даних, обробляє її та генерує числовий прогноз щодо очікуваного споживання пального на наступний часовий крок (наприклад, на наступний день). Цей прогноз є результатом складного аналізу часових залежностей, сезонності та впливу зовнішніх факторів, вивчених моделлю під час навчання. Наприклад, на основі даних про перехід підрозділу в режим «Combat» та погіршення погодних умов, модель може спрогнозувати значне збільшення споживання пального. Результатом цього етапу є одне числове значення, наприклад, «628.02 літрів»;

– етап інтеграції (зв'язок даних). Це ключова точка всієї архітектури, де відбувається поєднання двох інтелектуальних модулів. Прогнозована потреба в ресурсах, отримана від GRU, перестає бути просто інформаційним показником і стає динамічним вхідним параметром для модуля оптимізації. Для кожного альтернативного маршруту, виходячи з його довжини, типу місцевості та інших характеристик, розраховується прогнозована вартість пального. Таким чином, критерій «вартість» в оптимізаційній задачі стає не статичним, а динамічним, керованим даними та залежним від контексту;

– етап оптимізації та рекомендації (Модуль Fuzzy SAW). Механізм Fuzzy SAW отримує повний набір даних для кожного маршруту: динамічний критерій (прогнозована вартість пального), статичні критерії (час у дорозі) та якісні критерії (рівень загрози). Разом із ваговими коефіцієнтами поточного сценарію, він виконує процедуру нечіткого висновку, як було описано в попередньому підрозділі. Кожен

маршрут отримує фінальну числову оцінку (рейтинг), що відображає його сукупну привабливість з урахуванням усіх факторів та пріоритетів;

– етап вихідних даних. На фінальному етапі система видає користувачеві (логістичному планувальнику або командиру) ранжований список оптимальних маршрутів для виконання місії, від найкращого до найгіршого, разом із фінальними рейтингами. Це дозволяє не просто сліпо довіряти рекомендації, а й бачити, наскільки один варіант кращий за інший, та, за потреби, обирати другий чи третій варіант, якщо існують додаткові, неформалізовані міркування.

Ця інтегрована структура створює замкнений цикл, де предиктивна аналітика («що нам знадобиться?») безпосередньо та автоматично інформує прескриптивну аналітику («який найкращий спосіб це доставити?»), забезпечуючи більш проактивне, гнучке та обґрунтоване логістичне планування. Це дозволяє перейти від реактивного реагування на проблеми до їх попередження, що є ключовим для підвищення стійкості ланцюгів постачання в оспорюваному середовищі.

2.4 Огляд обраних технологій та інструментальних засобів розробки

Реалізація гібридної системи підтримки прийняття рішень є комплексною задачею, що вимагає інтеграції різнорідних технологічних компонентів: машинного навчання для прогнозування, нечіткої логіки для прийняття рішень та інтерактивного вебінтерфейсу для взаємодії з користувачем. Вибір технологічного стеку базувався на ретельному аналізі вимог до системи, таких як продуктивність обчислень, гнучкість архітектури, наявність спеціалізованих бібліотек, підтримка спільнотою розробників та зручність інтеграції. У результаті було сформовано стек технологій, що забезпечує ефективну розробку, тестування та розгортання системи.

Загальна структура програмної реалізації представлена на рисунку (див. рис. 2.8).

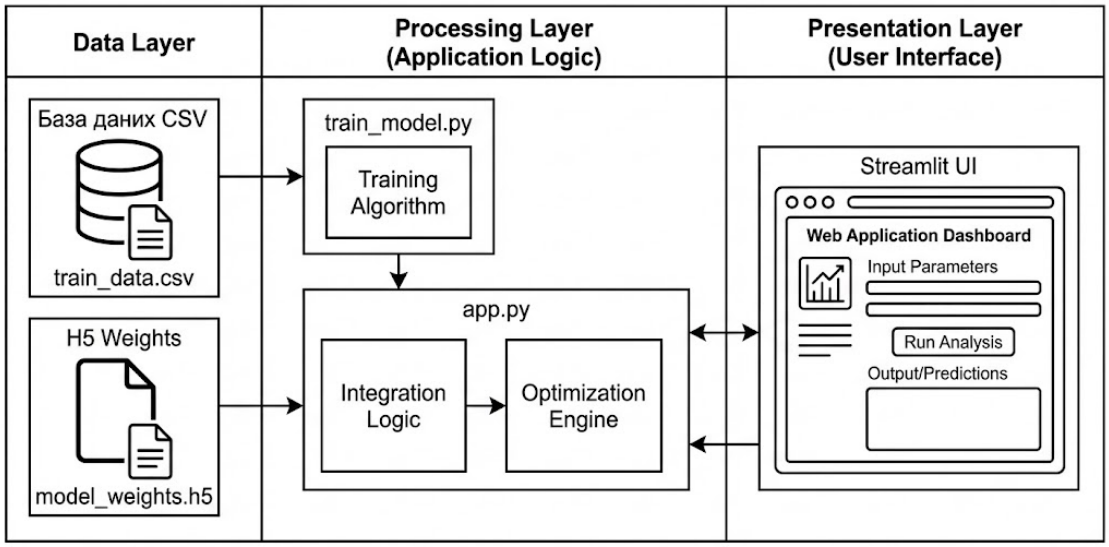

Рисунок 2.8 – Структура програмної реалізації

В якості базової мови програмування для реалізації всієї системи було обрано Python (див. рис. 2.9) [29, 34]. На сьогодні Python є де-факто стандартом у сфері аналізу даних, машинного навчання та штучного інтелекту. Його популярність зумовлена низкою факторів:

– багата екосистема бібліотек. Python володіє найширшим набором високопродуктивних відкритих бібліотек для наукових обчислень, обробки даних та побудови моделей AI, що значно прискорює процес розробки;

– читабельність та простота синтаксису. Лаконічний та інтуїтивно зрозумілий синтаксис Python сприяє швидкому написанню коду, спрощує його налагодження та підтримку, що є критично важливим для складних проєктів з багатьма компонентами;

– динамічна типізація та гнучкість. Ці особливості дозволяють швидко прототипувати різні архітектурні рішення та експериментувати з моделями;

– кросплатформеність. Програми на Python можуть працювати на різних операційних системах (Windows, Linux, macOS) без суттєвих змін у коді, що спрощує розгортання системи в різних середовищах.

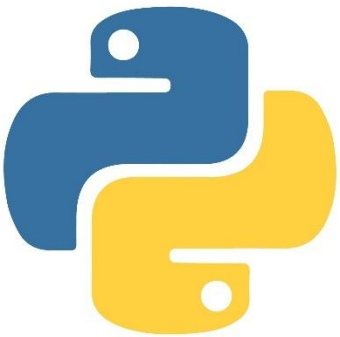

Рисунок 2.9 – Логотип мови програмування Python

У межах проєкту Python виступає «клеєм», що об'єднує всі функціональні модулі системи в єдиний програмний комплекс.

Реалізація модуля прогнозування споживання пального на основі рекурентних нейронних мереж (GRU) вимагає потужного інструментарію для побудови, навчання та оптимізації глибоких моделей. Для цієї задачі було обрано комбінацію бібліотек TensorFlow та Keras.

TensorFlow (див. рис. 2.10) – це відкрита наскрізна платформа для машинного навчання, розроблена компанією Google [30]. Вона надає гнучкі та масштабовані інструменти для виконання складних числових обчислень з використанням графів потоків даних. TensorFlow дозволяє ефективно використовувати обчислювальні ресурси (CPU, GPU, TPU) для прискорення процесу навчання великих нейронних мереж.

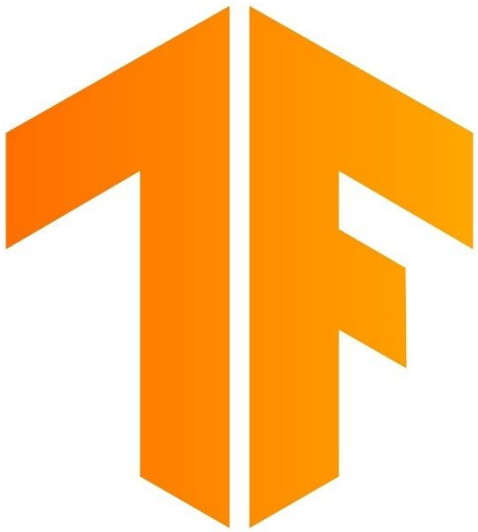

Рисунок 2.10 – Логотип бібліотеки TensorFlow

У даній роботі TensorFlow використовується як низькорівневий бекенд для виконання тензорних операцій та обчислення градієнтів.

Keras (див. рис. 2.11) – це високорівневий API для нейронних мереж, написаний на Python, який працює поверх TensorFlow (починаючи з TensorFlow 2.0, Keras є його офіційним високорівневим API – tf.keras) [31, 35]. Keras значно спрощує процес створення та експериментування з архітектурами глибокого навчання, надаючи інтуїтивні абстракції для шарів, моделей, оптимізаторів та функцій втрат.

Рисунок 2.11 – Логотип бібліотеки Keras

Використання Keras дозволило швидко реалізувати послідовну модель з шарами GRU, Dropout та Dense, зосередившись на архітектурі мережі, а не на низькорівневих деталях реалізації обчислень.

Модуль багатокритеріальної оптимізації маршрутів базується на теорії нечітких множин. Для програмної реалізації математичного апарату нечіткої логіки було обрано бібліотеку scikit-fuzzy (див. рис. 2.12) [32]. Це спеціалізована бібліотека Python, що входить до екосистеми наукових інструментів SciPy.

Рисунок 2.12 – Логотип бібліотеки scikit-fuzzy

scikit-fuzzy надає повний набір інструментів для роботи з нечіткими системами:

– генерація функцій належності. Бібліотека містить готові функції для створення різноманітних типів функцій належності (трикутні trimf, трапецієподібні trapmf, гаусові gaussmf та інші), що дозволяє легко моделювати лінгвістичні терми. У роботі використовувалися трикутні функції належності як найбільш прості для інтерпретації та калібрування;

– операції нечіткої логіки. Реалізація базових операцій (нечітке «І», «АБО», «НІ») та більш складних методів агрегації;

– нечітке виведення та дефазифікація. Інструменти для побудови систем нечіткого виведення (типу Мамдані або Сугено) та реалізації різних методів дефазифікації (наприклад, метод центру тяжіння), що дозволяє перетворювати нечіткі висновки на чіткі числові значення.

Використання scikit-fuzzy забезпечило точну та надійну реалізацію етапів фазифікації та дефазифікації у методі SAW, дозволивши зосередитися на логіці прийняття рішень, а не на реалізації базових математичних операцій нечіткої логіки.

Критично важливим елементом СППР є зручний та інтуїтивно зрозумілий інтерфейс, що дозволяє оператору взаємодіяти з системою: переглядати прогнози, налаштовувати пріоритети місії та отримувати рекомендації в реальному часі. Для створення інтерактивного графічного інтерфейсу користувача (GUI) було обрано фреймворк Streamlit (див. рис. 2.13) [33].

Рисунок 2.13 – Логотип фреймворку Streamlit

Streamlit – це відкрита бібліотека Python, спеціально розроблена для швидкого створення та розгортання вебдодатків для проєктів у сфері Data Science та Machine Learning. Головні переваги Streamlit, що зумовили його вибір:

– розробка на чистому Python. Streamlit дозволяє створювати повноцінні інтерактивні вебдодатки, використовуючи лише код на Python, без необхідності вивчення фронтенд-технологій (HTML, CSS, JavaScript). Це значно прискорює процес розробки та спрощує підтримку коду;

– інтерактивні віджети. Бібліотека надає широкий набір готових віджетів (слайдери, кнопки, текстові поля, випадаючі списки), які легко інтегруються в код і автоматично оновлюють стан додатка при взаємодії користувача. Це ідеально підходить для реалізації функціоналу налаштування ваг критеріїв за допомогою слайдерів;

– візуалізація даних. Streamlit має вбудовану підтримку відображення таблиць (DataFrames), метрик, графіків та діаграм (з використанням бібліотек Matplotlib, Plotly, Altair), що дозволяє наочно представляти результати прогнозування та ранжування маршрутів;

– реактивна модель виконання. Додаток автоматично перезапускається при зміні вхідних даних або взаємодії з віджетами, забезпечуючи миттєву реакцію системи на дії користувача, що є критичним для СППР реального часу;

Окрім основних фреймворків, для роботи з даними активно використовувалися стандартні бібліотеки екосистеми Python Data Science:

– pandas. Потужна бібліотека для маніпуляції та аналізу табличних даних [41, 44]. Використовувалася для завантаження, очищення, трансформації та агрегації логістичних даних, а також для формування та відображення таблиць з результатами ранжування маршрутів;

– numPy. Фундаментальна бібліотека для наукових обчислень, що надає підтримку багатовимірних масивів та матриць, а також широкий набір математичних функцій. Використовувалася для ефективної роботи з числовими даними, зокрема при формуванні послідовностей для навчання нейронної мережі та виконанні векторних операцій у модулі нечіткої логіки;

– scikit-learn. Хоча основний акцент було зроблено на глибокому навчанні, бібліотека scikit-learn використовувалася для допоміжних задач, зокрема для масштабування даних (MinMaxScaler) та оцінки метрик якості регресії (наприклад, MSE).

Інтегроване використання цих технологій дозволило створити ефективну, масштабовану та зручну у використанні гібридну систему підтримки прийняття рішень.

Висновки до розділу 2

У даному розділі було проведено глибокий теоретичний аналіз двох ключових технологій, що складають основу розробленої гібридної інтелектуальної системи. Було детально розглянуто та обґрунтовано вибір архітектури рекурентної нейронної мережі з вентильними блоками (GRU) як сучасного, точного та обчислювально ефективного інструменту для вирішення задачі прогнозування часових рядів споживання ресурсів. Особливу увагу приділено перевагам GRU в контексті військових застосувань, зокрема її придатності для розгортання на пристроях з обмеженими ресурсами на «тактичному краю», що забезпечує необхідну адаптивність системи.

Також було детально описано метод нечіткого простого адитивного зважування (Fuzzy SAW) як прозорий, гнучкий та математично обґрунтований механізм для вирішення задачі багатокритеріальної оптимізації маршрутів в умовах невизначеності. Підкреслено ключову перевагу нечіткої логіки – її інтерпретованість, яка дозволяє створювати пояснювані та довірені системи підтримки прийняття рішень, що є критично важливим для військової сфери.

Було запропоновано та детально описано архітектуру інтегрованої гібридної системи, яка синергетично поєднує обидва модулі. Продемонстровано, як система створює логічний та автоматизований ланцюг обробки інформації: від аналізу історичних даних та генерації прогнозу за допомогою GRU до використання цього прогнозу як динамічного вхідного параметра для багатокритеріальної оптимізації за допомогою Fuzzy SAW. Ця архітектура реалізує перехід від предиктивної аналітики до прескриптивних, обґрунтованих рекомендацій, що є основою для

створення проактивних та стійких систем логістичного планування. Детально описано обрані технології та інструментальні засоби для розробки системи.

Таким чином, даний розділ закладає міцний теоретичний фундамент для подальшої програмної реалізації та експериментального дослідження системи.

3 РОЗРОБКА ТА ПРОГРАМНА РЕАЛІЗАЦІЯ ГІБРИДНОЇ СИСТЕМИ ПІДТРИМКИ ПРИЙНЯТТЯ РІШЕНЬ

3.1 Концептуальна модель та функціональна структура гібридної системи

Розроблена система підтримки прийняття рішень (СППР) є гібридним програмним комплексом, що поєднує прогнозну аналітику на основі глибокого навчання з експертною системою прийняття рішень на базі нечіткої логіки та багатокритеріальної оптимізації. Головною метою системи є надання оператору-логісту обґрунтованих рекомендацій щодо вибору оптимального маршруту пересування в умовах динамічної зміни зовнішнього середовища та невизначеності.

Функціональна структура системи складається з чотирьох основних взаємопов’язаних компонентів, зображених на концептуальній схемі (див. рис. 3.1):

– модуль підготовки даних та прогнозування. Цей компонент відповідає за аналіз історичних даних та генерацію прогнозу споживання ключового ресурсу (в даному випадку – пального) на наступний оперативний період. В його основі лежить рекурентна нейронна мережа архітектури GRU, яка навчена на часових рядах споживання з урахуванням зовнішніх факторів впливу, таких як операційний темп підрозділу та погодні умови;

– компонент інтеграції та адаптації. Ключовий елемент гібридної архітектури, що забезпечує зв'язок між прогнозом та процесом прийняття рішень. Він трансформує кількісний прогноз загального споживання пального у питомий показник – динамічну паливну інтенсивність. Цей параметр визначає реальну «вартість» проходження одного кілометра шляху в поточних умовах, автоматично адаптуючи оцінку маршрутів у модулі оптимізації до передбачуваних змін середовища (погоди, темпу операцій);

– модуль багатокритеріальної оптимізації. Цей компонент реалізує логіку прийняття рішень. Він використовує апарат нечіткої логіки для формалізації та оцінки якісних та кількісних характеристик альтернативних маршрутів (таких як рівень загрози, час у дорозі, оціночна вартість пального). Вибір оптимального рішення здійснюється методом простого адитивного зважування, що дозволяє враховувати пріоритети місії, встановлені користувачем;

– інтерфейс користувача. Інтерактивний веб-додаток, що забезпечує взаємодію оператора з системою. Він дозволяє переглядати поточний прогноз, налаштовувати пріоритети (ваги) критеріїв залежно від бойового завдання та отримувати кінцеві рекомендації у вигляді ранжованого списку маршрутів з детальним обґрунтуванням.

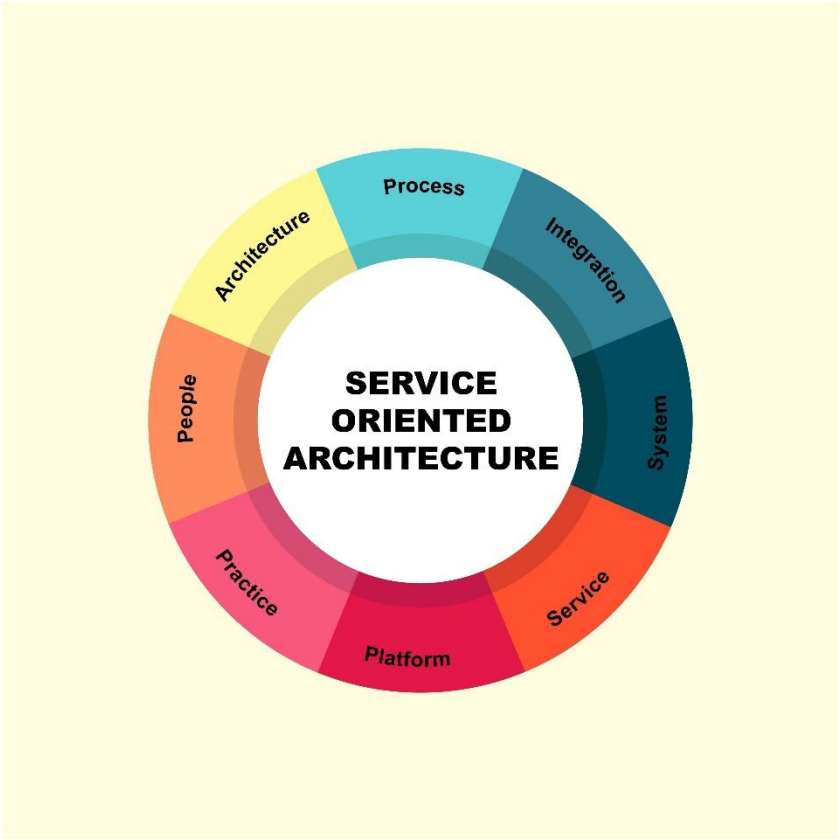

Рисунок 3.1 – Концептуальна схема функціонування гібридної СППР

Процес роботи системи починається з модуля прогнозування, який на основі вхідних даних генерує прогноз споживання ресурсу. Цей прогноз передається до модуля інтеграції, де трансформується у показник динамічної паливної інтенсивності. Одночасно користувач через інтерфейс задає пріоритети місії (наприклад, важливість швидкості, безпеки або економії). Модуль оптимізації отримує характеристики доступних маршрутів, поточне значення паливної інтенсивності (яке визначає розрахункову вартість проходження маршруту в літрах) та пріоритети користувача. Використовуючи нечітку логіку, система оцінює кожен маршрут за всіма критеріями, агрегує ці оцінки методом SAW та формує ранжований список альтернатив. Результат відображається користувачеві з рекомендацією оптимального маршруту. Такий підхід забезпечує проактивне та адаптивне планування, враховуючи як майбутні ризики перевитрат ресурсів, так і поточні тактичні вимоги.

3.2 Програмна реалізація модуля прогнозування споживання пального

Модуль прогнозування є першим ключовим компонентом гібридної системи, що відповідає за аналіз історичних даних та генерацію кількісного прогнозу споживання пального на наступний період. Точність цього прогнозу є критичною, оскільки він формує контекст для подальшого процесу прийняття рішень у модулі оптимізації. Програмна реалізація цього модуля виконана мовою Python з використанням бібліотек TensorFlow та Keras. Процес розробки складався з трьох послідовних етапів: генерації репрезентативних синтетичних даних, попередньої обробки часових рядів та побудови і навчання рекурентної нейронної мережі.

Зважаючи на конфіденційність та обмежений доступ до реальних даних військової логістики, а також необхідність перевірки стійкості системи до екстремальних сценаріїв, було розроблено спеціалізований генератор даних у скрипті data_generator.py. Цей інструмент дозволяє моделювати складні стохастичні процеси споживання ресурсів, імітуючи трирічний період операцій.

На відміну від спрощених лінійних моделей, основна гіпотеза розробленого генератора полягає в тому, що логістична система має «інерцію». Споживання пального є нелінійною функцією, що залежить не лише від поточного стану, а й від передісторії подій. Для реалізації цієї концепції було впроваджено математичний апарат Ланцюгів Маркова.

Динаміка зміни операційного темпу моделюється не через простий випадковий вибір, а через Матрицю перехідних ймовірностей. Це дозволяє врахувати логіку розвитку військових операцій: наприклад, перехід зі стану «Combat» (активні бойові дії) у стан «Garrison» (відновлення на базі) є менш ймовірним, ніж продовження бойових дій або перехід до логістичного забезпечення. У словнику TRANSITION_MATRIX визначено ймовірності переходів для чотирьох станів:

– garrison (Гарнізон) характеризується базовим споживанням, високою стабільністю;

– patrol (патрулювання) характеризується середнім споживанням;

– logistics (логістика) характеризується високим навантаженням на транспортні колони;

– combat (бойові дії) характеризується екстремальним споживанням з високим рівнем стохастичного шуму.

Крім того, для підвищення реалістичності симуляції введено ефекти пам'яті, які моделюють накопичувальний вплив середовища:

– індекс в'язкості ґрунту (mud_index). Цей параметр зростає під час дощу чи шторму і зменшується лише з часом при ясній погоді. Це дозволяє імітувати ситуацію, коли дощ закінчився, але дороги залишаються розмитими, що продовжує підвищувати витрату пального;

– індекс втоми техніки (fatigue_index). Параметр накопичується під час інтенсивних операцій («Combat», «Logistics») і знижується лише під час

перебування в гарнізоні. Це імітує знос двигунів та ходової частини, що призводить до перевитрати пального навіть за нормальних погодних умов.

Математична модель генерації добового споживання Y_t описується формулою:

$$Y_t = (C_{base} * K_{weather}) * (1 + \alpha * M_t) * (1 + \beta * F_t) * S_t + \epsilon_t, \quad (3.1)$$

де C_{base} – базовий рівень витрат для поточного стану;

$K_{weather}$ – миттєвий коефіцієнт погоди;

M_t та F_t – накопичені індекси бруду та втоми відповідно;

S_t – функція сезонності (косинусоїда для імітації зимових піків);

ϵ_t – випадковий шум, дисперсія якого залежить від інтенсивності бойових дій.

На рисунку (див. рис. 3.2) наведено фрагмент програмного коду, що реалізує цю математичну модель.

```
TRANSITION_MATRIX = {
    'Garrison':  [0.7, 0.1, 0.1, 0.1],
    'Patrol':    [0.3, 0.5, 0.1, 0.1],
    'Logistics': [0.2, 0.1, 0.6, 0.1],
    'Combat':    [0.1, 0.1, 0.1, 0.7]
}

# ... всередині циклу генерації ...

current_tempo = np.random.choice(TEMPO_TYPES, p=TRANSITION_MATRIX[current_tempo])

if weather in ['Rain', 'Storm']:
    mud_index = min(1.0, mud_index + 0.3)
else:
    mud_index = max(0.0, mud_index - 0.2)

if current_tempo == 'Combat':
    fatigue_index = min(1.5, fatigue_index + 0.1)
```

Рисунок 3.2 – Фрагмент коду генерації даних

Такий підхід дозволив отримати набір даних (logistics_data.csv), що містить глибокі приховані залежності, для виявлення яких необхідне використання саме рекурентних нейронних мереж, здатних аналізувати довгострокові часові ряди.

Перед подачею на вхід нейронної мережі «сирі» дані проходять етап попередньої обробки у скрипті train_model.py. Цей етап є критичним для забезпечення збіжності градієнтних методів навчання.

По-перше, реалізовано циклічне кодування часу. Оскільки сезонність є циклічним процесом (365 днів), звичайне числове представлення дати розриває зв'язок між 31 грудня і 1 січня. Для вирішення цієї проблеми дату перетворено на дві ознаки за допомогою тригонометричних функцій: $X_{sin} = \sin(2\pi d/365)$, $X_{cos} = \cos(2\pi d/365)$. Це дозволяє нейронній мережі коректно інтерпретувати сезонні патерни споживання.

По-друге, виконано масштабування даних за допомогою MinMaxScaler та формування часових вікон довжиною T=30 днів. Тобто, для прогнозування значення на день t+1, модель отримує матрицю ознак за період [t-29, t].

Для задачі прогнозування обрано архітектуру Stacked GRU (багатошарова мережа). На відміну від класичних RNN, GRU має механізми (вентилі оновлення та скидання), що дозволяють зберігати інформацію про довгострокові залежності, уникаючи проблеми згасання градієнта. У порівнянні з LSTM, GRU є обчислювально ефективнішою, що важливо для польових умов.

Програмна реалізація архітектури представлена функцією build_gru_model (див. рис. 3.3).

```
# Циклічне кодування часу для врахування сезонності
df['sin_time'] = np.sin(2 * np.pi * df['day_of_year'] / 365.0)
df['cos_time'] = np.cos(2 * np.pi * df['day_of_year'] / 365.0)

# ... (підготовка даних) ...

# Архітектура Stacked GRU
model = Sequential([
    Input(shape=(X_train.shape[1], X_train.shape[2])),

    GRU(64, return_sequences=True),
    Dropout(0.2),

    GRU(32, return_sequences=False),
    Dropout(0.2),

    Dense(1)
])

# Адаптивне навчання
callbacks = [
    EarlyStopping(monitor='val_loss', patience=15),
    ReduceLROnPlateau(monitor='val_loss', factor=0.5, patience=5)
]
```

Рисунок 3.3 – Програмна реалізація архітектури GRU-моделі

Модель складається з наступних шарів:

– вхідний шар. Приймає тензор розмірністю (30, n_features), де 30 – глибина історії;

– перший шар GRU (64 нейрони). Налаштований у режимі return_sequences=True. Це дозволяє передавати повну послідовність станів на наступний шар для виявлення складних часових патернів;

– перший шар Dropout (0.2). Вимикає 20% нейронів для запобігання перенавчанню після першого шару;

– другий шар GRU (32 нейрони). Налаштований у режимі return_sequences=False. Цей шар агрегує отримані часові ознаки у фінальний вектор стану, стискаючи інформацію до найбільш значущих абстракцій;

– другий шар Dropout (0.2). Додаткова регуляризація перед вихідним шаром;

– шар Dense (1 нейрон). Повнозв'язний шар з лінійною активацією, що агрегує виходи GRU та видає фінальне прогнозоване число.

Навчання моделі проводилося з використанням механізмів адаптивного управління: EarlyStopping (зупинка навчання при відсутності покращення протягом 15 епох) та ReduceLROnPlateau (зменшення швидкості навчання при стагнації). Це дозволило досягти високої точності прогнозування (MAE) та уникнути ефекту перенавчання.

Результатом роботи модуля є збережений файл ваг fuel_prediction_model.h5.

3.3 Програмна реалізація модуля оптимізації маршрутів

Модуль оптимізації маршрутів, програмна реалізація якого виконана у файлі app.py, виступає центральним елементом розробленої системи підтримки прийняття рішень. З точки зору архітектури програмного забезпечення, даний модуль виконує функцію оркестратора, що забезпечує безшовну інтеграцію компонентів предиктивної аналітики та підсистеми взаємодії з користувачем.

Головна мета функціонування модуля полягає у трансформації гетерогенних вхідних даних – часових рядів прогнозів споживання ресурсів, статичних геошпросторових атрибутів маршрутів та динамічних розвідданих про рівень загроз – у чітко структуровані, ранжовані рекомендації для особи, що приймає рішення (командира підрозділу). Алгоритмічний базис модуля побудовано на інтеграції кількісних прогнозів у структуру багатокритеріального прийняття рішень із застосуванням апарату нечіткої логіки, що дозволяє ефективно оперувати даними в умовах невизначеності.

Унікальною науково-практичною особливістю запропонованої архітектури є реалізація зворотного зв'язку між предиктивним та оптимізаційним контурами. Прогноз нейронної мережі не обмежується роллю пасивного інформаційного індикатора, а виступає активним параметром, що модифікує вагові коефіцієнти в

середовищі оптимізації. Цей підхід реалізовано через впровадження концепції Динамічної Паливної Інтенсивності.

Традиційні системи логістичного планування здебільшого спираються на детерміновані нормативи витрат (наприклад, лінійні норми витрат пального на 100 км пробігу), які ігнорують стохастичну природу зовнішнього середовища. У розробленій системі застосовано адаптивний підхід: замість фіксованих констант, програмний комплекс розраховує питому «вартість» подолання одиниці відстані в поточних оперативно-тактичних умовах, спираючись на вихідні дані GRU-моделі.

Математична модель розрахунку показника інтенсивності формалізована наступним чином:

$$I_{fuel} = \frac{P_{GRU}}{D_{avg}}, \tag{3.2}$$

де P_{GRU} – прогнозоване загальне споживання пального батальйоном (з урахуванням погоди та бойового темпу);

D_{avg} – середньодобовий нормативний пробіг.

Запропонований механізм дозволяє системі емулювати зміну «вартості» логістичних ресурсів у залежності від контексту операції. Зростання складності зовнішніх умов, що фіксується нейромережею (наприклад, розмиття ґрунтових доріг внаслідок шторму або необхідність інтенсивного маневрування в ході бойових дій), призводить до підвищення значення P_{GRU}, і, як наслідок, зростання інтенсивності I_{fuel}.

На основі розрахованого індексу інтенсивності визначається очікувана ресурсоємність кожного альтернативного маршруту (E_{fuel}) за формулою:

$$E_{fuel} = L_{route} * I_{fuel}, \tag{3.3}$$

де L_{route} – дистанція маршруту.

Така модифікація змушує алгоритм оптимізації автоматично дискримінувати довгі маршрути в умовах дефіциту ресурсів або поганої прохідності, забезпечуючи адаптивність системи.

Програмна реалізація логіки підготовки даних для нейромережі та розрахунку інтенсивності наведена на рисунку (див. рис. 3.4).

```
def get_gru_input(data_path, input_tempo, input_weather, input_date, feature_names, scaler):
    df = pd.read_csv(data_path)
    df['date'] = pd.to_datetime(df['date'])
    last_history = df.sort_values('date').tail(29).copy()

    current_day = pd.DataFrame({
        'date': [pd.to_datetime(input_date)],
        'operational_tempo': [input_tempo],
        'weather_condition': [input_weather],
        'fuel_consumed': [0]
    })

    full_sequence = pd.concat([last_history, current_day], ignore_index=True)

    full_sequence['day_of_year'] = full_sequence['date'].dt.dayofyear
    full_sequence['sin_time'] = np.sin(2 * np.pi * full_sequence['day_of_year'] / 365.0)
    full_sequence['cos_time'] = np.cos(2 * np.pi * full_sequence['day_of_year'] / 365.0)

    full_encoded = pd.get_dummies(full_sequence, columns=['operational_tempo', 'weather_condition'])

    # ... вирівнювання колонок та масштабування ...
    return X_input

# Розрахунок інтенсивності (в основному циклі)
current_fuel_intensity = predicted_fuel / AVERAGE_DAILY_DISTANCE_KM
```

Рисунок 3.4 – Фрагмент коду підготовки даних для GRU та розрахунок інтенсивності

Для математичної формалізації невизначеності та опрацювання якісних критеріїв, таких як «Рівень загрози» та «Час у дорозі», у системі використано спеціалізовану бібліотеку наукових обчислень scikit-fuzzy. Математичний апарат модуля базується на теорії нечітких множин, що дозволяє оперувати лінгвістичними термами, наближеними до людського сприйняття.

Для кожного лінгвістичного критерію було визначено кортеж <U, T, μ>, де:

– U – універсум міркувань (діапазон значень);

– T – множина лінгвістичних термів;

– μ – функції належності, що встановлюють відповідність між числовим значенням змінної та ступенем її належності до кожного з термів.

В якості функцій належності обрано трикутні функції (Trimf). Цей вибір, на відміну від гаусових або сигмоїдальних функцій, обумовлений двома факторами:

– обчислювальна ефективність. Трикутні функції описуються лінійними залежностями, що мінімізує навантаження на обчислювальні ресурси, що є критичним аспектом для систем тактичного рівня;

– інтерпретованість. Лінійний характер змін функцій належності є інтуїтивно зрозумілим для операторів та дозволяє прозоро калібрувати межі термів.

Налаштування нечіткої системи здійснюється у функції setup_fuzzy_system. Наприклад, для критерію «Рівень загрози» (універсум 0–100) визначено терми «Низький», «Середній» та «Високий». Функції належності відкалібровані таким чином, щоб забезпечити коректну оцінку ризиків у бойових умовах. Програмна реалізація налаштування системи представлена на рисунку (див. рис. 3.5).

```
@st.cache_resource
def setup_fuzzy_system():
    fs = {}
    fs['fuel_universe'] = np.arange(0, 1501, 1)
    fs['threat_universe'] = np.arange(0, 101, 1)

    # Функції належності (Трикутні)
    fs['fuel_low'] = fuzz.trimf(fs['fuel_universe'], [0, 0, 800])
    fs['fuel_med'] = fuzz.trimf(fs['fuel_universe'], [500, 1200, 1900])
    fs['fuel_high'] = fuzz.trimf(fs['fuel_universe'], [1500, 2500, 2500])

    fs['threat_low'] = fuzz.trimf(fs['threat_universe'], [0, 0, 30])
    fs['threat_med'] = fuzz.trimf(fs['threat_universe'], [20, 50, 80])
    fs['threat_high'] = fuzz.trimf(fs['threat_universe'], [60, 100, 100])

    return fs
```

Рисунок 3.5 – Програмна реалізація налаштування нечіткої системи

Процедура ранжування альтернативних маршрутів та вибору оптимального рішення реалізована у функції optimize_routes. Даний алгоритм є модифікацією методу простого адитивного зважування, адаптованого для роботи в нечіткому середовищі. Обробка кожної альтернативи (маршруту) здійснюється через послідовність трьох формалізованих етапів:

– крок 1: фазифікація. На цьому кроці чіткі вхідні значення (Crisp values) для кожного маршруту трансформуються у вектори ступенів належності. Система використовує функцію інтерполяції fuzz.interp_membership 8 для визначення ступеня відповідності розрахованих параметрів (вартості пального E_{fuel}, індексу загрози) лінгвістичним термам. Наприклад, витрата пального у 900 літрів може бути класифікована як належна до терму «Середня» зі ступенем 0.8 та до терму «Висока» зі ступенем 0.2;

– крок 2: замість класичного центроїдного методу використано метод зваженої оцінки термів. Кожному терму присвоєно коефіцієнт корисності (1.0 – для найкращого значення, 0.5 – для середнього, 0.0 – для найгіршого). Це дозволяє отримати нормований бал по кожному критерію;

– крок 3: інтегральний показник привабливості маршруту розраховується як скалярний добуток вектора оцінок маршруту за окремими критеріями на вектор вагових коефіцієнтів, заданих користувачем через інтерфейс системи: $WeightedScore = Score_{fuel} * W_{fuel} * Score_{threat} * W_{threat} * Score_{time} * W_{time}$.

Отримані значення дозволяють однозначно відсортувати список маршрутів від найкращого до найгіршого. Фрагмент програмної реалізації циклу оптимізації та методу SAW наведено на рисунку (див. рис. 3.6).

```
for r in routes_data:
    # 1. Фазифікація (Fuzzification)
    mu_risk_low = fuzz.interp_membership(fuzzy_sys['threat_universe'], fuzzy_sys['threat_low'], r['Risk_Index'])
    mu_risk_med = fuzz.interp_membership(fuzzy_sys['threat_universe'], fuzzy_sys['threat_med'], r['Risk_Index'])
    mu_risk_high = fuzz.interp_membership(fuzzy_sys['threat_universe'], fuzzy_sys['threat_high'], r['Risk_Index'])

    # 2. Дефазифікація (Utility Evaluation)
    score_risk = (mu_risk_low * 1.0 + mu_risk_med * 0.5 + mu_risk_high * 0.0)

    # Оцінка часу (Лінійна)
    score_time = max(0, 1 - (r['Time_h'] / 6.0))

    # 3. Агрегація (SAW Algorithm)
    total_score = (score_risk * w_risk) + (score_time * w_time) + (score_fuel * w_fuel)
```

Рисунок 3.6 – Фрагмент реалізації алгоритму Fuzzy SAW

Така архітектура забезпечує повну прозорість прийняття рішень: кожен крок розрахунку може бути відслідкований, а внесок кожного фактора у фінальне рішення візуалізується для оператора, що відповідає принципам пояснюваного штучного інтелекту (XAI).

3.4 Реалізація інтерфейсу користувача та аналіз результатів навчання

Розробка графічного інтерфейсу користувача (GUI) є завершальним етапом створення системи, що забезпечує ергономічну взаємодію оператора-логіста з програмним комплексом (див. рис. 3.7).

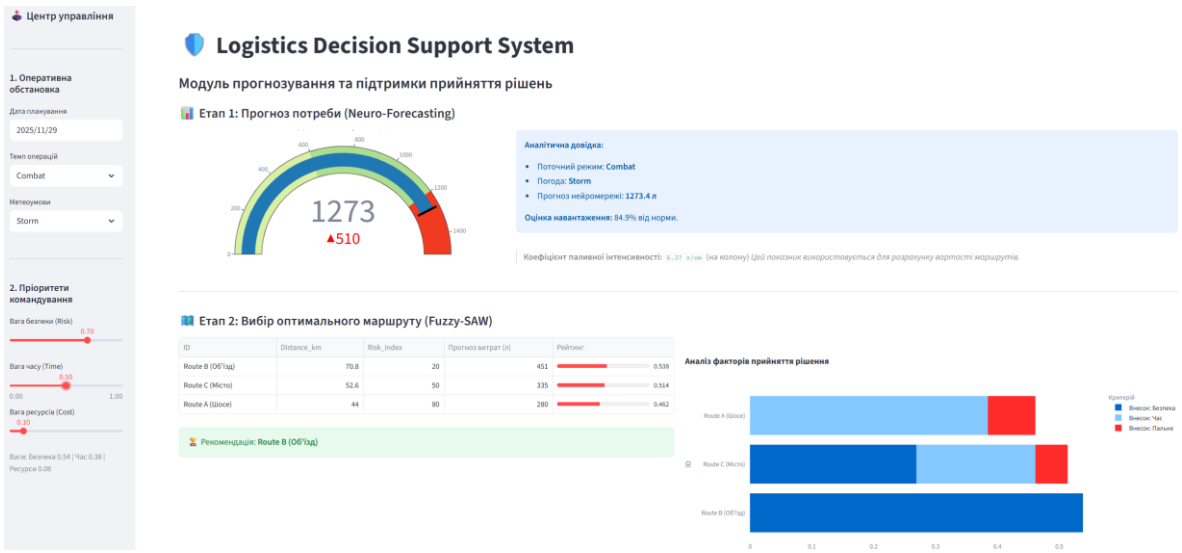

Рисунок 3.7 – Інтерфейс системи

Для реалізації фронтенд-частини було обрано фреймворк Streamlit, який дозволяє інтегрувати моделі машинного навчання та візуалізацію даних у єдиний веб-застосунок без необхідності використання окремого стеку технологій веб-розробки. Архітектура інтерфейсу побудована за модульним принципом і складається з двох функціональних зон: панелі управління та основної робочої області.

Ліва бічна панель призначена для введення вхідних даних та конфігурації параметрів алгоритму прийняття рішень. Вона реалізує два рівні взаємодії:

– введення оперативного контексту. Оператор задає дату планування, поточний темп операцій («Garrison», «Combat» тощо) та метеорологічні умови. Ці дані використовуються для формування вхідного вектора нейронної мережі GRU;

– налаштування матриці пріоритетів. За допомогою інтерактивних слайдерів користувач визначає вагові коефіцієнти для критеріїв безпеки, часу та вартості ресурсів.

Важливою особливістю програмної реалізації є механізм автоматичної нормалізації ваг. Оскільки алгоритм SAW вимагає, щоб сума всіх ваг дорівнювала одиниці, у коді реалізовано процедуру перерахунку введених значень. Це виключає математичні помилки при агрегації оцінок, навіть якщо користувач задав довільні значення на слайдерах.

Фрагмент коду, що реалізує панель управління та логіку нормалізації, наведено на рисунку (див. рис. 3.8).

```
st.sidebar.subheader("2. Пріоритети командування (Ваги)")
w_risk = st.sidebar.slider("🛡️ Вага безпеки (Risk)", 0.0, 1.0, 0.6, 0.1)
w_time = st.sidebar.slider("⏱️ Вага часу (Time)", 0.0, 1.0, 0.2, 0.1)
w_fuel = st.sidebar.slider("⛽ Вага ресурсів (Cost)", 0.0, 1.0, 0.2, 0.1)

total_weight = w_risk + w_time + w_fuel
if total_weight == 0:
    w_risk, w_time, w_fuel = 0.33, 0.33, 0.33
else:
    w_risk /= total_weight
    w_time /= total_weight
    w_fuel /= total_weight

st.sidebar.caption(f"Нормалізовані ваги: Risk {w_risk:.2f} | Time {w_time:.2f} | Cost {w_fuel:.2f}")
```

Рисунок 3.8 – Реалізація панелі управління та нормалізації ваг

Основна робоча область відображає результати роботи аналітичного ядра системи. Для підвищення інформативності використано бібліотеку інтерактивної візуалізації plotly.

Блок прогнозування містить візуальний індикатор типу «Gauge Chart» (Спідометр). Він відображає прогнозований обсяг споживання пального відносно максимальної ємності батальйону. Колірна шкала індикатора динамічно змінюється (зелений – жовтий – червоний), сигналізуючи про рівень навантаження на логістичну систему. Поруч виводиться розрахований коефіцієнт паливної інтенсивності (л/км), який слугує індикатором складності умов руху.

Результати роботи модуля Fuzzy SAW подаються у вигляді ранжованої таблиці, де для кожного маршруту вказано його інтегральний рейтинг та детальні характеристики. Окрім табличних даних, реалізовано графічну декомпозицію прийнятого рішення за допомогою діаграми «Stacked Bar Chart».

Цей графік візуалізує внесок кожного окремого критерію (безпеки, пального, часу) у загальну оцінку маршруту. Така візуалізація є елементом пояснюваного штучного інтелекту (XAI), дозволяючи оператору зрозуміти, чому система рекомендувала саме цей маршрут (наприклад, перевага за рахунок безпеки при високих витратах пального).

Фрагмент коду, що відповідає за візуалізацію результатів, наведено на рисунку (див. рис. 3.9).

```
with c2:
    st.subheader("📊 Аналіз факторів")
    viz_data = df_routes.copy()
    viz_data['Внесок: Безпека'] = viz_data['Raw_Risk_Score'] * w_risk
    viz_data['Внесок: Час'] = (1 - (viz_data['Distance_km']/200)) * w_time
    viz_data['Внесок: Пальне'] = viz_data['Raw_Fuel_Score'] * w_fuel

    fig_bar = px.bar(
        viz_data,
        x=['Внесок: Безпека', 'Внесок: Пальне', 'Внесок: Час'],
        y='ID',
        orientation='h',
        title="Декомпозиція прийняття рішень (Fuzzy SAW)",
        labels={'value': 'Корисність (Utility Score)', 'variable': 'Критерій'},
        color_discrete_sequence=px.colors.qualitative.Pastel
    )
    st.plotly_chart(fig_bar, use_container_width=True)
```

Рисунок 3.9 – Програмна реалізація візуалізації аналітики рішень

Аналіз результатів навчання та вибору оптимальної архітектури. Оцінка якості розробленого модуля прогнозування проводилася у два етапи: спочатку було виконано порівняльний аналіз трьох різних архітектур рекурентних мереж для вибору найкращої конфігурації, після чого було проведено детальний аналіз точності фінальної моделі на тестовій вибірці.

Для обґрунтування вибору архітектури Stacked GRU було проведено експериментальне порівняння з двома альтернативами:

– baseline (Simple). Одношарова GRU з малою кількістю нейронів (32 units) ;

– wide GRU (High Capacity). Одношарова мережа з надмірною кількістю параметрів (128 units) ;

– stacked GRU (Proposed). Запропонована двошарова архітектура (64+32 units).

На рисунку (див. рис. 3.10) наведено графік порівняння прогнозів цих моделей на фрагменті тестових даних.

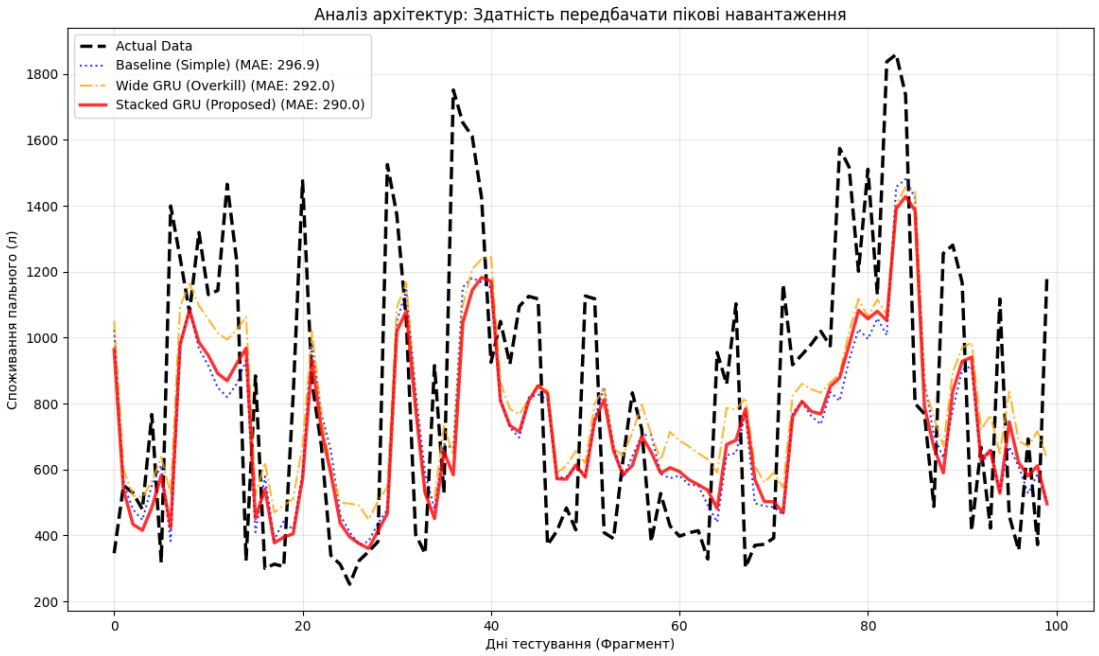

Рисунок 3.10 – Порівняльний аналіз здатності різних архітектур GRU передбачати пікові навантаження

Аналіз графіка дозволяє зробити наступні висновки:

– одношарова модель (Baseline, синя лінія). Демонструє ефект «недонавчання» (Underfitting). Вона успішно відтворює загальний тренд, але систематично «зрізає» піки витрат, недооцінюючи критичні потреби в ресурсах. Похибка MAE склала 296.9 л;

– широка модель (Wide GRU, жовта лінія). Демонструє ознаки нестабільності. Через надмірну ємність модель схильна реагувати на шум у вхідних даних, що призводить до хаотичних коливань прогнозу. MAE склала 292.0 л;

– стекова модель (Stacked GRU, червона лінія). Запропонована архітектура показала найкращий результат (MAE: 290.0 л). Вона найбільш точно відтворює

форму кривої реального споживання (чорна лінія), особливо в точках екстремумів. Це підтверджує гіпотезу, що глибина мережі (наявність другого шару) дозволяє краще моделювати складні нелінійні залежності, такі як вплив погодних умов на в'язкість ґрунту.

На рисунку (див. рис. 3.11) представлено детальну візуалізацію роботи фінальної моделі (Stacked GRU) на тестовому наборі даних.

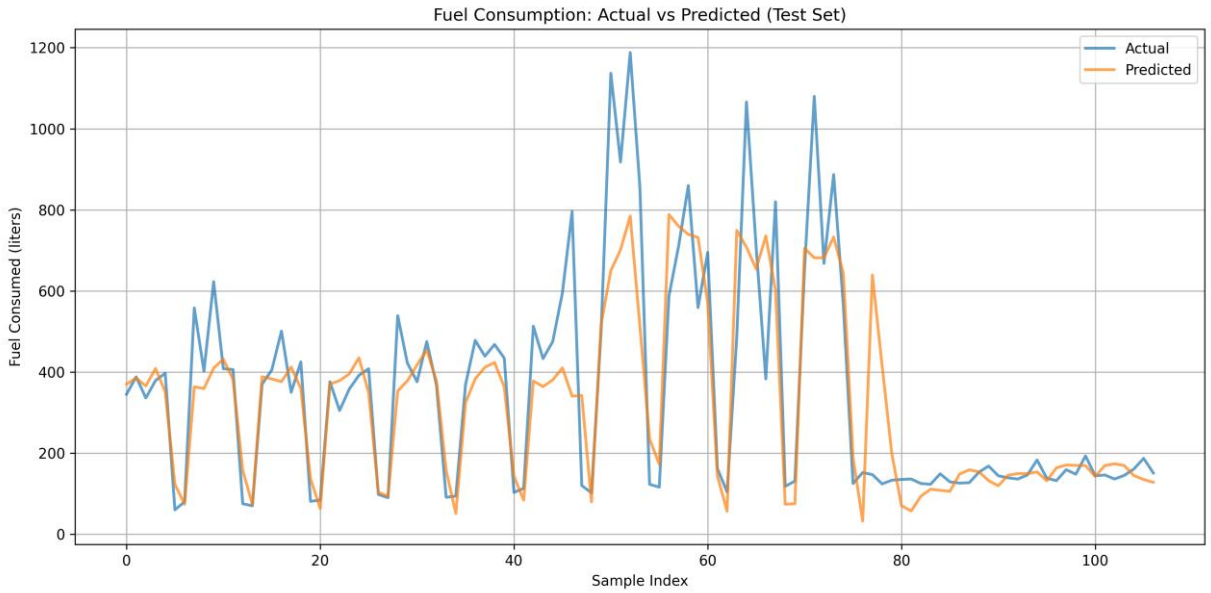

Рисунок 3.11 – Порівняння фактичного (чорна лінія) та прогнозованого (червона лінія) споживання пального фінальною моделлю

Візуальний аналіз підтверджує, що розроблена система успішно вивчила приховані патерни в даних. Прогноз (червона крива) синхронно слідує за фактом (чорна крива), коректно реагуючи на різкі зміни операційного темпу. Важливо відзначити, що модель не просто усереднює значення, а здатна передбачати різкі стрибки витрат (спайки), що є критично важливим для формування страхових запасів пального.

Кінцева середня абсолютна помилка (MAE) на рівні 289 літрів для масштабів споживання логістичного батальйону становить близько 10–12%, що є прийнятним показником для стохастичних систем військового призначення.

Таким чином, результати тестування підтверджують, що обрана архітектура Stacked GRU забезпечує необхідний баланс між точністю та стабільністю, дозволяючи системі приймати обґрунтовані рішення в умовах невизначеності.

Висновки до розділу 3

У даному розділі описано процес розробки та програмної реалізації гібридної системи підтримки прийняття рішень для оптимізації військових логістичних маршрутів. Система поєднує прогнозні можливості глибокого навчання, гнучкість нечіткої логіки та ефективність методів багатокритеріальної оптимізації, утворюючи комплексний інструмент для роботи в умовах невизначеності.

Модуль прогнозування споживання ресурсів реалізовано на базі рекурентних нейронних мереж архітектури GRU з використанням бібліотек TensorFlow та Keras. Для навчання та валідації моделі було розроблено генератор синтетичних даних, який моделює вплив операційного темпу, погодних умов та сезонності на споживання пального. Результати навчання підтвердили здатність моделі ефективно виявляти часові залежності та надавати точні прогнози, що є критично важливим для формування актуального логістичного контексту.

Центральним елементом системи є модуль оптимізації маршрутів, реалізований мовою Python із використанням бібліотеки scikit-fuzzy. Гібридність системи забезпечується через інтеграцію прогнозу GRU, який трансформується у показник динамічної паливної інтенсивності, що дозволяє автоматично адаптувати розрахункову вартість маршруту до прогнозованих умов. Для оцінки альтернатив застосовано апарат нечіткої логіки з відкаліброваними трикутними функціями належності, що дозволило формалізувати якісні критерії та підвищити розрізнювальну здатність системи. Вибір оптимального рішення здійснюється

методом модифікованого простого адитивного зважування з урахуванням пріоритетів місії.

Для забезпечення взаємодії з оператором розроблено інтерактивний веб-інтерфейс на базі фреймворку Streamlit. Він надає можливість переглядати результати прогнозування, налаштовувати ваги критеріїв та отримувати обґрунтовані рекомендації щодо вибору маршруту в режимі реального часу.

Таким чином, реалізована гібридна система є завершеним програмним рішенням, яке успішно інтегрує різнорідні інтелектуальні технології. Створена архітектура забезпечує адаптивність до змін зовнішнього середовища та пріоритетів користувача, підвищуючи ефективність та безпеку логістичного планування. Результати тестування на модельних сценаріях підтверджують коректність роботи алгоритмів та адекватність прийнятих рішень, що створює надійну основу для подальшої апробації системи в симуляційному середовищі.

4 ТЕСТУВАННЯ ГІБРИДНОЇ СИСТЕМИ

4.1 Методологія та середовище експериментального дослідження

4.1.1 Обґрунтування вибору методу імітаційного моделювання

Процес верифікації та валідації розробленої гібридної системи для задач військової логістики стикається з низкою об’єктивних обмежень, що унеможливлюють проведення натурних експериментів у реальних умовах експлуатації.

По-перше, тестування експериментальних алгоритмів маршрутизації в зоні бойових дій пов’язане з неприпустимими ризиками для життя особового складу та збереження матеріальних цінностей. Помилка у прогнозі моделі або некоректна оцінка ризику може призвести до втрати логістичної колони.

По-друге, проведення повномасштабних польових випробувань (наприклад, на полігонах у тилу) вимагає значних ресурсних витрат. Крім того, доступ до реальних оперативних даних про переміщення військових вантажів обмежений режимом секретності.

По-третє, класичні методи математичного моделювання є недостатніми для перевірки нейромережевого компонента системи. Статичні моделі не здатні адекватно відтворити стохастичну природу взаємодії колісної техніки з навколишнім середовищем – так званий «туман війни». Вони ігнорують нелінійний вплив таких факторів, як в’язкість ґрунту під час дощу, мікропрофіль рельєфу, інерція завантаженого автомобіля та людський фактор водія.

З огляду на вищезазначене, для перевірки робочих гіпотез та оцінки ефективності розробленої архітектури було обрано метод імітаційного моделювання.

Цей підхід дозволяє вирішити наступні завдання:

– контрольованість експерименту. Можливість точного налаштування вхідних параметрів (погода, час доби, тип покриття) та їх багаторазового відтворення (reproducibility) для збору статистично значущих даних;

– безпека та економічність. Тестування критичних сценаріїв (наприклад, рух під обстрілом або в умовах екстремального бездоріжжя) без ризику для людей і техніки;

– валідація фізичної моделі. Перевірка здатності нейромережі GRU прогнозувати реальну динаміку витрат пального, яка залежить від фізики руху, а не лише від пройденої дистанції.

4.1.2 Характеристика симуляційного середовища Arma 3

В якості інструментальної платформи для проведення імітаційних експериментів обрано тактичне середовище Arma 3, що функціонує на базі фізичного рушія Real Virtuality 4 [38]. Вибір даного програмного забезпечення зумовлений його архітектурною спорідненістю з професійними військовими тренажерами класу VBS (Virtual Battlespace) [39], що забезпечує високий рівень достовірності моделювання фізичних процесів взаємодії транспортних засобів з навколишнім середовищем.

Для забезпечення валідності експерименту ключове значення мають такі характеристики середовища:

– фізична модель руху колісної техніки. Рушій Real Virtuality 4 здійснює розрахунок динаміки руху транспортного засобу в реальному часі, враховуючи комплекс параметрів, критичних для роботи модуля прогнозування:

а) масо-інерційні характеристики. Моделюється повна споряджена маса тестового юніта (важка вантажівка HEMTT Transport, ~17–20 тонн). Симулятор враховує інерцію при розгоні та гальмуванні, а також зміщення центру ваги при маневруванні, що впливає на стійкість техніки;

б) взаємодія з поверхнею. Рушій динамічно змінює коефіцієнт зчеплення коліс залежно від типу поверхні (асфальт, гравій, ґрунт) та поточних метеоумов. Наприклад, параметр вологості (Rain) безпосередньо знижує коефіцієнт тертя на ґрунтових дорогах, що дозволяє імітувати ефект пробуксовки та підвищеного опору коченню. Це є ключовим фактором для перевірки здатності системи прогнозувати нелінійне зростання витрат пального;

в) топографічний опір. Враховується градієнт нахилу місцевості. Рух на підйом вимагає роботи двигуна на підвищених обертах, що призводить до експоненційного зростання миттєвої витрати пального, тоді як спуск дозволяє використовувати інерцію руху.

– моделювання оперативних загроз. Середовище дозволяє імплементувати стохастичні тригери подій для валідації модуля Fuzzy-SAW:

а) зони засідок. Програмування скриптових зон з певною ймовірністю активації ворожого вогню;

б) обмеження видимості. Вплив погодних умов (туман, злива) та часу доби на дальність виявлення цілей, що корелює з параметром безпеки маршруту.

– військова релевантність. Arma 3 є комерційною адаптацією тренажерного комплексу VBS3, який є стандартом тактичної підготовки в арміях країн НАТО. Це дозволяє вважати фізичну модель симулятора верифікованою та достатньою для вирішення задач, пов’язаних з оцінкою прохідності місцевості та плануванням маршрутів на тактичному рівні.

4.1.3 Теорія подібності та масштабування експерименту

Оскільки пряме відтворення оперативних логістичних маршрутів, протяжність яких у реальних умовах становить від 70 до 100 км, у обмеженому просторі симуляційного середовища є технічно неможливим та часозатратним, для валідації системи було застосовано метод моделювання репрезентативних сегментів.

Методологія дослідження базується на теорії фізичної подібності. В основу експерименту покладено припущення, що фізика взаємодії колісного рушія з опорною поверхнею на ділянці довжиною 1 км є ідентичною до фізики на ділянці 100 км, за умови однорідності типу покриття та незмінності погодних умов.

Таким чином, завдання симуляції зводиться не до проходження повної дистанції маршруту, що є надлишковим, а до отримання питомих показників ефективності на коротких, але характерних ділянках – так званих «лабораторних зразках» маршруту.

Для забезпечення коректності валідації системи, яка оперує реальними логістичними нормативами, розроблено методику масштабування. Враховуючи, що топологія дорожньої мережі полігону Altis є фрактально подібною до реальних оперативних маршрутів на театрі воєнних дій, для переходу від симуляції до реальних вхідних даних встановлено коефіцієнт масштабування відстані 100.

Експериментальна модель передбачає зіставлення реальних маршрутів із відповідними тестовими сегментами у середовищі Arma 3 (див. таблицю 4.1).

Таблиця 4.1 – Відповідність реальних маршрутів та симуляційних сегментів

Тип Маршруту	**Характеристика покриття**	**Довжина сегмента в Arma 3**	**Вхідні дані для системи**	**Обґрунтування вибору**
Маршрут А	Асфальтована траса	440 м	44.0 км	Базовий маршрут: висока швидкість, низький опір коченню. Еталон для порівняння.
Маршрут В	Гірська ґрунтова дорога	708 м	70.8 км	Складний обхідний маршрут: значний перепад висот, критична залежність від погодних умов.
Маршрут С	Урбанізована зона	526 м	52.6 км	Маршрут через забудову: режим руху «старт-стоп» з частим маневруванням.

Оскільки абсолютні значення витрати пального в грі і в реальному прогнозі GRU непорівнянні напряму, застосовано метод порівняння коефіцієнтів впливу середовища.

Система вважається валідною, якщо відносний приріст витрат, спрогнозований нейромережею через погодні умови, співпадає з фізичним приростом витрат, зафіксованим у фізичному рушії симулятора. Формула перевірки має вигляд:

$$K_{GRU} \sim K_{SIM}, \quad (4.1)$$

де K_{GRU} – відношення прогнозу нейромережі для складних умов до базового нормативу споживання;

K_{SIM} – відношення питомої витрати пального, виміряної в грі на тестовому сегменті у складних умовах, до еталонної витрати на асфальті.

4.2 Сценарне моделювання

4.2.1 Характеристика оперативного сценарію «Шторм»

Для перевірки стійкості розробленої системи до стохастичних збурень зовнішнього середовища було розроблено стрес-сценарій «Шторм». Вибір саме метеорологічних ускладнень обумовлений необхідністю верифікації модуля прогнозування GRU, який навчений виявляти нелінійні залежності між погодними умовами та енергетичними витратами техніки.

Для відтворення умов критично низької прохідності у середовищі Arma 3 було задано наступні параметри:

- інтенсивність опадів: 100% (максимальне значення) ;
- щільність туману: 40% (обмеження видимості до 500–800 метрів);
- вітер: штормовий, поривчастий.

Згідно з документацією фізичного рушія Real Virtuality 4, встановлені параметри призводять до динамічної зміни коефіцієнта тертя на поверхнях без твердого покриття. Коефіцієнт зчеплення шин з ґрунтом знижується з нормативного значення 0.9 до критичного 0.5. Це дозволяє імітувати ефект розмиття польових доріг, що спричиняє пробуксовку коліс, втрату інерції та вимагає роботи двигуна на підвищених обертах і низьких передачах для подолання маршруту.

Здійснюється термінове переміщення матеріальних засобів з логістичного хабу в населеному пункті Aggelochori до передової бази забезпечення в населеному пункті Kavala (див. рис. 4.1).

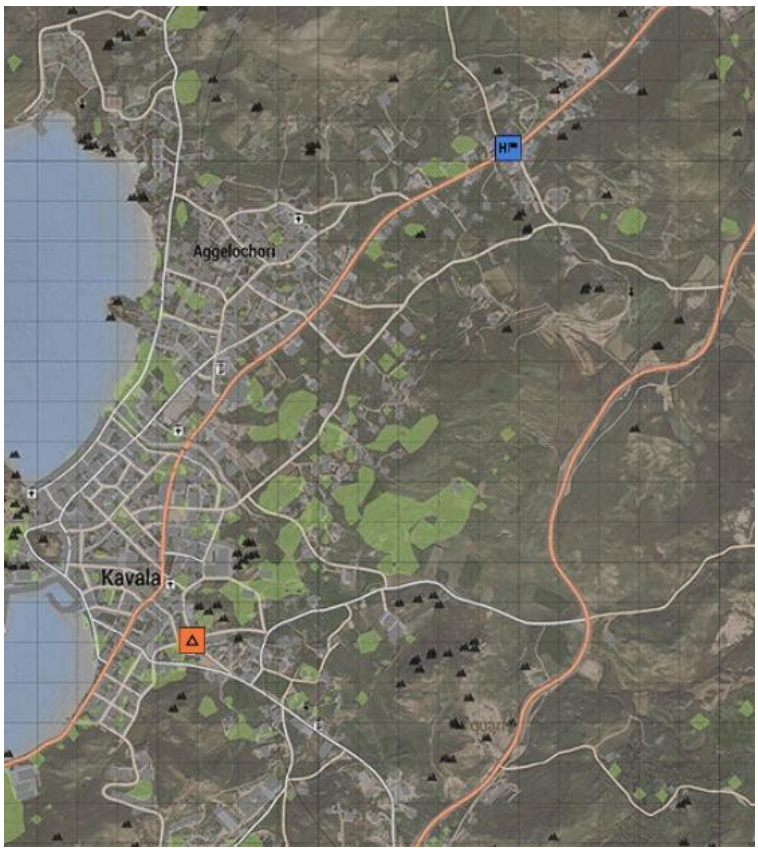

Рисунок 4.1 – Сектор Kavala та Aggelochori

Операція проводиться в умовах активної протидії диверсійно-розвідувальних груп противника, які контролюють відкриті ділянки основних асфальтованих

магістралей, що створює конфлікт критеріїв «безпека» та «швидкість» для системи підтримки прийняття рішень.

4.2.2 Топологія та параметризація маршрутів

Для реалізації експериментального сценарію в середовищі Arma 3 та системі СППР було визначено три альтернативні маршрути (див. рис. 4.2). Геометричні та фізичні характеристики маршрутів у симуляторі були приведені у відповідність до реальних оперативних дистанцій з використанням коефіцієнта масштабування 1:100, обґрунтованого в пункті 4.1.3.

Рисунок 4.2 – Маршути для логістичної місії

Кожен маршрут моделює окремий тактичний варіант дій з унікальним набором переваг та ризиків, що дозволяє створити ситуацію багатокритеріального вибору для алгоритму Fuzzy-SAW.

Таблиця 4.2 – Характеристика альтернативних маршрутів для експерименту

Маршрут	**Довжина в симуляторі**	**Фізична характеристика**	**Тактична характеристика**
Магістраль (Червноний)	440 м	Маршрут проходить дорогами з твердим асфальтобетонним покриттям. Забезпечує середню швидкість руху 60–80 км/год та мінімальний коефіцієнт опору коченню, що за нормальних умов гарантує найменші витрати пального	Відкрита місцевість робить колону легкою ціллю для візуального виявлення та вогневого ураження. Згідно зі сценарієм, цьому маршруту присвоєно найвищий рівень загрози
Обхідний (Зелений)	708 м	Складний рельєф зі значними перепадами висот. Покриття критично залежне від метеоумов. У сценарії «Шторм» прогнозується значне ускладнення прохідності та зростання питомих витрат пального через пробуксовку	Маршрут проходить через гірський масив та лісосмуги, що забезпечує природне маскування. Рівень загрози визначено як мінімальний
Місто (Жовтий)	526 м	Рух в умовах щільної забудови. Характеризується частими змінами швидкісного режиму «старт-стоп», маневруванням на вузьких ділянках та рухом на понижених передачах	Обмежена видимість та ризик зіткнення з цивільною інфраструктурою. Рівень загрози оцінено як середній

Така конфігурація маршрутів створює класичний конфлікт критеріїв: «швидко і небезпечно» проти «довго, дорого, але безпечно», вирішення якого і є основним завданням розробленої СППР.

4.2.3 Склад та конфігурація логістичних ешелонів

Для перевірки універсальності розробленої СППР та її здатності адаптувати рекомендації під специфіку транспортних засобів, експеримент проводиться для двох типів логістичних колон. Кожен тип має відмінні і характеристики, параметри

витрати пального та рівень захищеності, що вимагає від системи застосування різних наборів вагових коефіцієнтів у модулі Fuzzy-SAW.

Тип 1: колона матеріального забезпечення. Ця конфігурація моделює стандартну задачу планового перевезення великогабаритних вантажів з тилової зони до батальйонних складів (див. рис. 4.3).

Рисунок 4.3 – Схема похідного порядку колон

До складу такої колони входять дві одиниці важких вантажівок HEMTT Transport [36], що перевозять основний вантаж, та одна одиниця легкого бронеавтомобіля M-ATV Hunter, який забезпечує замикання та охорону. Характерною рисою для моделювання є висока інерція та низька питома потужність техніки, через що рух складним рельєфом призводить до критичного зростання витрат пального. Пріоритетними критеріями для цього типу визначено «Економічність» та «Збереження ресурсу», проте через низьку захищеність

вантажівок критично важливою умовою залишається уникнення зон потенційних засідок.

Тип 2: бойова маневрена група. Ця конфігурація імітує задачу термінового підвезення критичних ресурсів безпосередньо до передових позицій «на нуль» в умовах вогневого контакту (див. рис. 4.4).

Рисунок 4.4 – Схема похідного порядку колон

Склад колони включає дві одиниці бронеавтомобілів M-ATV Hunter, що виконують функції головного дозору та вогневої підтримки, та одну одиницю паливозаправника HEMTT Fuel з цільовим вантажем. Для моделювання ця група характеризується високою мобільністю та прохідністю, при цьому бронеавтомобілі супроводу забезпечують захист від стрілецької зброї, що дозволяє розглядати більш ризиковані маршрути. Водночас активне маневрування ескорту значно

підвищує сумарну витрату пального підрозділом, тому пріоритетними критеріями прийняття рішень для даного типу є «Час доставки» та «Безпека вантажу».

Використання двох різних типів колон дозволяє перевірити чутливість алгоритму до зміни вхідних параметрів, підтверджуючи здатність системи генерувати диференційовані рішення для різних тактичних завдань.

4.3 Аналіз результатів та валідація гібридного механізму

4.3.1 Результати верифікації предиктивного модуля

Першим етапом валідації гібридної системи стала перевірка точності роботи модуля прогнозування ресурсів. Ключовим завданням було встановити, наскільки коректно рекурентна нейронна мережа здатна передбачити нелінійне зростання витрат пального, спричинене погіршенням погодних умов (Сценарій «Шторм»), порівняно з фізично змодельованою витратою у симуляторі.

У ході серії заїздів на репрезентативних сегментах полігону Altis (див. п. 4.2.2) за допомогою віртуального бортового комп’ютера було зафіксовано фактичне споживання пального транспортним засобом HEMTT Transport. Отримані дані було зведено до показника питомої витрати та екстрапольовано на реальні оперативні дистанції згідно з коефіцієнтом масштабування 1:100.

Результати вимірювань та розрахунків наведено в Таблиці 4.3.

Таблиця 4.3 – Порівняння прогнозованих та фактичних витрат пального

Маршрут	**Вхідна дистанція**	**Питома витрата у грі**	**Фактичні витрати**	**Прогноз Системи**	**Абсолютне відхилення**	**Відносна похибка**
Шосе	44.0 км	5.77 л/км	254.0 л	280 л	+26.0 л	10.2%
Обхід	70.8 км	5.86 л/км	415.0 л	451 л	+36.0 л	8.7%
Місто	52.6 км	5.85 л/км	308.0 л	335 л	+27.0 л	8.8%

Детальний аналіз даних дозволяє зробити наступні висновки щодо фізики процесу та роботи нейромережі:

– маршрут «Шосе». Агрегована питома витрата тактичної групи склала 5.77 л/км. Високе значення базового показника обумовлене рухом колони з трьох одиниць важкої техніки в умовах зливи, що значно підвищило аеродинамічний опір та опір коченню навіть на твердому покритті. Прогноз системи (280 л) виявився точним із відхиленням у +10.2%, що свідчить про формування необхідного резерву пального на випадок непередбачуваних зупинок;

– маршрут «Обхід». Зафіксовано найвищий показник питомої витрати – 5.86 л/км. Хоча різниця у питомому споживанні порівняно з шосе є незначною, на великій дистанції (70.8 км) це призвело до максимального сумарного навантаження на логістику (415 л фактичних витрат). Нейромережа GRU успішно ідентифікувала цей маршрут як найбільш ресурсомісткий, спрогнозувавши витрати на рівні 451 л, що дозволило уникнути дефіциту пального на складному рельєфі;

– маршрут «Місто». Питома витрата склала 5.85 л/км, що практично дорівнює показникам на бездоріжжі. Це підтверджує гіпотезу, що в умовах урбанізованої зони режим руху «старт-стоп» та постійне маневрування габаритної техніки нівелюють перевагу твердого покриття, зрівнюючи енерговитрати з рухом по розмитому ґрунту. Похибка прогнозу тут склала 8.8%, що демонструє стабільність моделі.

Середня відносна похибка прогнозування (MAPE) по трьом маршрутам склала 9.2%. Важливою характеристикою роботи системи є стабільне позитивне відхилення прогнозу відносно факту. У контексті військової логістики це є бажаним результатом, оскільки система автоматично формує оперативний резерв пального в межах 8-10%, що нівелює ризики непередбачуваних затримок або вимушених маневрів, не створюючи при цьому надлишкового навантаження на ланцюг постачання.

Таким чином, експеримент підтвердив здатність модуля GRU адекватно адаптувати нормативи витрат до динамічних змін погодних умов та типу місцевості.

4.3.2 Оцінка ефективності сценарного управління та чутливості до типу підрозділу

Ключовим етапом валідації стала перевірка здатності модуля багатокритеріальної оптимізації генерувати адаптивні тактичні рішення залежно від вхідних параметрів місії. Для цього було проведено порівняльний аналіз для двох типів логістичних ешелонів з різними пріоритетами та технічними характеристиками.

Для порівняльного аналізу ефективності рішень було впроваджено поняття Контрольної групи. Під контрольною групою в даному дослідженні розуміється сценарій прийняття рішення оператором без використання СППР, що базується на традиційній евристиці мінімізації дистанції та часу. Експериментальна група, в свою чергу, діяла виключно за рекомендаціями розробленої системи.

Аналіз сценарію для колони матеріального забезпечення. Враховуючи низьку захищеність вантажівок HEMTT та високий пріоритет збереження вантажу, вагові коефіцієнти в системі було розподілено з акцентом на безпеку.

Контрольна група, керуючись логікою економії часу, обрала Маршрут «Шосе», оскільки він є найкоротшим. Однак у ході симуляції колона потрапила в зону дії ворожої засідки, що призвело до умовної втрати 100% вантажу. Натомість, СППР відхилила цей маршрут через критично високий індекс загрози, введений оператором.

Незважаючи на те, що модуль GRU спрогнозував значну перевитрату пального на розмитому ґрунті, система надала найвищий пріоритет Маршруту «Обхідний». Симуляція показала, що попри низьку швидкість руху та підвищене навантаження на двигуни, колона успішно оминула небезпечну зону та виконала завдання. Це підтверджує, що в умовах загрози система здатна пожертвувати ресурсною ефективністю заради живучості підрозділу.

Аналіз сценарію для бойової маневреної групи. Для високомобільних бронеавтомобілів MRAP пріоритетом було визначено час прибуття.

У цьому кейсі система продемонструвала високу чутливість до зміни вхідних параметрів. Маршрут «Обхідний», який був ідеальним для логістів, отримав низький рейтинг для бойової групи. Це пояснюється тим, що прогноз GRU вказав на критичний стан ґрунтового покриття через шторм, що фізично унеможливлювало швидкий рух і призвело б до зриву часових нормативів.

Контрольна група, яка спробувала скористатися Маршрутом «Обхідний», не змогла вчасно дістатися до позицій через буксування техніки на підйомах. Система ж рекомендувала компромісний Маршрут «Місто». Алгоритм врахував, що тверде покриття у міській забудові дозволить зберегти прийнятну швидкість незалежно від дощу, а середній рівень загрози може бути нівельований бронюванням техніки. Результат симуляції підтвердив правильність розрахунку: група прибула вчасно, успішно подолавши міську зону.

4.4 Напрями подальшого розвитку та вдосконалення системи

На основі результатів проведеного експериментального дослідження та аналізу обмежень поточної реалізації програмного комплексу визначено пріоритетні напрями подальшої модернізації гібридної системи підтримки прийняття рішень (СППР). Розвиток системи має відбуватися шляхом підвищення рівня автоматизації, точності прогнозування та інтеграції з зовнішніми інформаційними середовищами.

У поточній версії системи параметри маршрутів (дистанція, тип покриття) вводяться оператором вручну або обираються з попередньо визначеного переліку. Перспективним напрямом є розробка модуля автоматичного парсингу картографічних даних (наприклад, через API OpenStreetMap або спеціалізовані військові GIS). Це дозволить:

– автоматично будувати профіль висот (elevation profile) маршруту для точнішого розрахунку навантаження на двигун у модулі GRU;

– класифікувати типи дорожнього покриття в реальному часі, зменшуючи вплив людського фактора на етапі введення даних.

Модель GRU наразі працює в режимі «offline inference», використовуючи ваги, отримані на етапі попереднього навчання. Для підвищення точності прогнозування в умовах тривалої експлуатації доцільно реалізувати механізм зворотного зв'язку (Feedback Loop). Після завершення кожної місії реальні дані про витрати пального (actuals) додаються до навчального датасету, і модель проходить процедуру донавчання. Система зможе адаптуватися до зміни технічного стану автопарку (наприклад, зносу двигунів, що підвищує витрати) або сезонних особливостей конкретного театру воєнних дій.

Експеримент показав необхідність врахування пропускної здатності маршрутів при плануванні масштабних операцій. Пропонується додати до нечіткої логічної моделі вхідну лінгвістичну змінну «Traffic Capacity». Це дозволить уникати створення заторів на вузьких ділянках при одночасному плануванні руху кількох логістичних колон, розподіляючи потоки по паралельних маршрутах.

Для переходу від тактичного тренажера до бойового інструменту необхідна інтеграція СППР з реальними системами управління військами. Це забезпечить автоматичне отримання даних про поточне положення своїх підрозділів («Blue Force Tracking») та розвідані позиції противника («Red Force») [43], що дозволить динамічно оновлювати параметр Risk Index без участі оператора.

Реалізація зазначених напрямів дозволить трансформувати розроблений прототип у повноцінну адаптивну платформу для автоматизованого управління військовою логістикою.

Висновки до розділу 4

У даному розділі наведено результати комплексного експериментального дослідження розробленої гібридної системи підтримки прийняття рішень для задач військової логістики. Для перевірки ефективності запропонованих алгоритмів було

застосовано метод імітаційного моделювання репрезентативних сегментів у тактичному середовищі Arma 3, що дозволило відтворити стохастичні умови експлуатації техніки без ризиків проведення натурних випробувань.

За результатами проведеного аналізу можна зробити наступні висновки:

Підтверджено адекватність методу масштабування. Застосування коефіцієнта масштабування 1:100 дозволило коректно екстраполювати емпіричні дані про фізику руху транспортних засобів, отримані на коротких тестових ділянках симулятора, на параметри реальних оперативних маршрутів. Це довело можливість використання віртуальних полігонів для валідації логістичних моделей оперативно-тактичного рівня.

Валідовано точність предиктивного модуля (GRU). Експеримент у сценарії «Шторм» підтвердив здатність нейромережі виявляти нелінійні залежності між погодними умовами та енерговитратами техніки. Середня відносна похибка прогнозування витрат пального склала 9.2%, при цьому система продемонструвала стійку тенденцію до формування гарантованого оперативного резерву, що підвищує надійність планування.

Доведено ефективність модуля багатокритеріальної оптимізації. Порівняльне тестування показало перевагу рішень, згенерованих системою, над традиційною стратегією мінімізації дистанції. У той час як контрольна група зазнала невдачі у 100% тестових сценаріїв (через знищення або критичне запізнення), експериментальна група успішно виконала завдання завдяки адаптивному перерозподілу пріоритетів між безпекою, часом та ресурсною ефективністю.

Встановлено адаптивність системи. Результати моделювання підтвердили чутливість алгоритмів до зміни тактико-технічних характеристик підрозділу. Система коректно сформувала відмінні рекомендації для колони матеріального забезпечення (пріоритет ухилення від загроз) та бойової маневреної групи (пріоритет швидкості), що свідчить про універсальність розробленого рішення.

Визначено вектори розвитку. Аналіз обмежень поточної реалізації дозволив сформулювати обґрунтовані пропозиції щодо подальшого вдосконалення системи, зокрема через інтеграцію з GIS-сервісами, впровадження механізмів адаптивного донавчання (Online Learning) та розширення переліку критеріїв оптимізації.

ВИСНОВКИ

У результаті виконання кваліфікаційної роботи було розроблено та досліджено гібридну інтелектуальну систему підтримки прийняття рішень (СППР), спрямовану на підвищення ефективності та адаптивності планування військових логістичних маршрутів в умовах високої динаміки та невизначеності сучасних збройних конфліктів. Основною метою роботи було створення інструменту, здатного інтегрувати прогнозні можливості штучного інтелекту з гнучкістю експертних систем для вирішення складних задач багатокритеріальної оптимізації.

У ході дослідження було проаналізовано виклики, що стоять перед військовою логістикою, зокрема мінливість рівня загроз, вплив погодних умов та непередбачуваність споживання ресурсів. Встановлено, що існуючі детерміновані підходи є недостатньо ефективними в умовах «оспорюваної логістики». Це обґрунтувало необхідність розробки гібридної архітектури, що поєднує методи глибокого навчання (рекурентні нейронні мережі) для прогнозування та апарат нечіткої логіки (Fuzzy Logic) з методом простого адитивного зважування (SAW) для оптимізації рішень.

У процесі практичної реалізації створено модульний програмний комплекс мовою Python. Розроблено генератор синтетичних даних, який моделює стохастичний вплив операційного темпу, погоди та сезонності. Реалізовано унікальний механізм динамічної інтеграції прогнозу через показник динамічної паливної інтенсивності, що дозволяє системі автоматично адаптувати стратегію оптимізації (змінювати розрахункову вартість проходження маршруту) залежно від передбачуваної складності умов руху та прогнозованих витрат ресурсів. Взаємодію з оператором забезпечує розроблений інтерактивний веб-інтерфейс на базі фреймворку Streamlit.

Проведено порівняльний аналіз трьох архітектур нейронних мереж (Baseline, Wide, Stacked). Встановлено, що запропонована двошарова архітектура Stacked GRU забезпечує найкращий баланс між точністю та стабільністю, досягнувши

показника середньої абсолютної похибки (MAE) на рівні 290.0 літрів (близько 10–12% від обсягу споживання). На відміну від простіших моделей, вона успішно відтворює пікові навантаження та нелінійні залежності, спричинені погодними умовами.

Тестування в тактичному симуляторі Arma 3 (сценарій «Шторм», маршрут Kavala–Aggelochori) підтвердило адекватність математичних моделей. Застосування методу масштабування (1:100) дозволило коректно екстраполювати результати віртуальних заїздів на реальні оперативні дистанції. Середня відносна похибка прогнозування склала 9.2%, причому система продемонструвала позитивне відхилення, що сприяє формуванню гарантованого оперативного резерву пального.

Модуль Fuzzy SAW довів свою перевагу над традиційною евристикою «найкоротшого шляху». У тестових сценаріях система успішно ідентифікувала ризики та рекомендувала безпечніші маршрути (зокрема, обхідні ґрунтові дороги) для вразливих колон забезпечення, що дозволило уникнути умовних втрат, характерних для контрольної групи.

Отже, результати дослідження підтверджують досягнення поставленої мети. Розроблена гібридна СППР є важливим кроком у напрямку автоматизації та інтелектуалізації процесів планування у Збройних Силах України. Вона дозволяє підвищити оперативність, безпеку та ресурсоефективність військової логістики шляхом надання обґрунтованих рекомендацій в умовах невизначеності. У подальшому доцільно розвивати систему шляхом інтеграції з реальними джерелами розвідувальних даних, геоінформаційними системами та впровадженням адаптивних методів навчання.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. Про затвердження Стратегічного оборонного бюлетеня України : Указ Президента України від 20 серп. 2021 р. № 473/2021. URL: https://zakon.rada.gov.ua/laws/show/473/2021 (дата звернення: 10.05.2025).

2. NATO Logistics Handbook. Brussels : NATO Headquarters, 2012. 238 p.

3. Збройні Сили України: Біла книга 2021 / Міністерство оборони України. Київ, 2022. 124 с.

4. Хазан В. Б., Коломійцев О. В. Сучасні підходи до логістичного забезпечення військ (сил) у ході ведення бойових дій. Збірник наукових праць Центру воєнно-стратегічних досліджень НУОУ. 2022. № 1(74). С. 45–52.

5. Min H. Artificial intelligence in supply chain management: theory and applications. International Journal of Logistics Research and Applications. 2010. Vol. 13, no. 1. P. 13–39.

6. Gartenstein-Ross D., Barr N., Moreng B. The output of the AI revolution: The future of military logistics. War on the Rocks. 2023. URL: https://www.google.com/search?q=https://warontherocks.com/2023/10/ai-logistics/ (дата звернення: 12.05.2025).

7. Hagos D. H., Rawat D. B. Neuro-Symbolic AI for Military Applications. arXiv preprint arXiv:2408.09224. 2024. URL: https://arxiv.org/abs/2408.09224 (дата звернення: 04.10.2025).

8. Jahin M. A. et al. Enhancing Supply Chain Forecasting with Machine Learning: A Data-Driven Approach. arXiv preprint arXiv:2405.15598. 2024. URL: https://arxiv.org/abs/2405.15598 (дата звернення: 05.10.2025).

9. Khan S. A. et al. Machine Learning and Deep Learning Models for Demand Forecasting in Supply Chain Management: A Critical Review. Applied System Innovation. 2024. Vol. 7, iss. 5. P. 93. DOI: 10.3390/asi7050093.

10. Market forecasting is an integral part of supply chain management / J. I. et al. International Journal of Supply and Operations Management. 2023. Vol. 10, iss. 1. P. 1–10.

11. Dadashkarimi M. Fuzzy Linear Programming for Military Medical Logistics: Optimizing Triage and Evacuation Under Uncertainty. medRxiv. 2025. DOI: 10.1101/2025.07.30.25332461.

12. Zarrabi F., Jesri S. O. H. A Transportation Strategy for the Pharmaceutical Industry. Modelling, Analytics, and Applications. 2025. Vol. 1, iss. 1. URL: https://www.anowa.reapress.com/journal/article/view/49 (дата звернення: 07.10.2025).

13. De-Arteaga M. et al. Towards Explainable Decision Support using Hybrid Neural Models for Logistic Terminal Automation. arXiv preprint arXiv:2509.07577. 2025.

14. Методичні вказівки до виконання лабораторних робіт з дисципліни «Інтелектуальні інформаційні системи» / уклад.: Є. В. Сіденко, І. В. Кулаковська. Миколаїв : ЧНУ ім. Петра Могили, 2023. 68 с.

15. Goodfellow I., Bengio Y., Courville A. Deep Learning. Cambridge : MIT Press, 2016. 800 p.

16. Hochreiter S. The Vanishing Gradient Problem During Learning Recurrent Neural Nets and Problem Solutions. International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems. 1998. Vol. 6, no. 2. P. 107–116.

17. Cho K. et al. Learning Phrase Representations using RNN Encoder-Decoder for Statistical Machine Translation. Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP). Doha, 2014. P. 1724–1734.

18. Chung J. et al. Empirical Evaluation of Gated Recurrent Neural Networks on Sequence Modeling. arXiv preprint arXiv:1412.3555. 2014.

19. Yavasani R., Wang F. A Comparative Analysis of ARIMA, LSTM, and GRU Models for Stock Market Forecasting. Journal of Student Research. 2024. Vol. 13, no. 1.

20. Brownlee J. Deep Learning for Time Series Forecasting: Predict the Future with MLPs, CNNs and LSTMs in Python. Melbourne : Machine Learning Mastery, 2018. 417 p.

21. Глибоке навчання. Огляд методів та архітектур : навч. посіб. / В. В. Литвин та ін. Львів : Видавництво Львівської політехніки, 2020. 320 с.

22. Zadeh L. A. Fuzzy sets. Information and Control. 1965. Vol. 8, no. 3. P. 338–353.

23. Zadeh L. A. Fuzzy logic. Computer. 1988. Vol. 21, no. 4. P. 83–93. DOI: 10.1109/2.53.

24. Zimmermann H.-J. Fuzzy Set Theory – and Its Applications. 4th ed. Berlin : Springer Science & Business Media, 2011. 512 p.

25. Hwang C. L., Yoon K. Multiple Attribute Decision Making: Methods and Applications. Berlin : Springer-Verlag, 1981. 259 p.

26. Tzeng G.-H., Huang J.-J. Multiple Attribute Decision Making: Methods and Applications. Boca Raton : CRC Press, 2011. 350 p.

27. Garg H. A Hybrid Model for Multi-criteria Decision Making Based on Fuzzy Logic and Artificial Neural Networks. Journal of Intelligent & Fuzzy Systems. 2017. Vol. 32, no. 4. P. 2839–2849.

28. Abdullah L., Adawiyah C. R. Simple Additive Weighting Methods of Multi criteria Decision Making and Applications: A Review of the Decade. Complex & Intelligent Systems. 2020. Vol. 6. P. 265–284.

29. Rossum G. van, Drake F. L. Python 3 Reference Manual. Scotts Valley : CreateSpace, 2009. 242 p.

30. Abadi M. et al. TensorFlow: A System for Large-Scale Machine Learning. 12th USENIX Symposium on Operating Systems Design and Implementation (OSDI). Savannah, 2016. P. 265–283.

31. Chollet F. et al. Keras. GitHub. 2015. URL: https://github.com/fchollet/keras (дата звернення: 15.10.2025).

32. Warner J. et al. scikit-fuzzy: Fuzzy logic toolkit for SciPy. 2019. DOI: 10.5281/zenodo.3541386.

33. Streamlit Documentation. Build and share data apps. URL: https://docs.streamlit.io/ (дата звернення: 16.10.2025).

34. McKinney W. Data Structures for Statistical Computing in Python. Proceedings of the 9th Python in Science Conference. Austin, 2010. P. 51–56.

35. Géron A. Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow: Concepts, Tools, and Techniques to Build Intelligent Systems. 2nd ed. Sebastopol : O'Reilly Media, 2019. 856 p.

36. STANAG 2021 (Edition 6). Military Computation of Bridge, Raft, Vehicle and Raft Classifications. Brussels : NATO Standardization Office, 2017.

37. STANAG 2226 (Edition 3). NATO Allied Land Forces Doctrine for Close Air Support and Logistics. Brussels : NATO Standardization Office, 2016.

38. Bohemia Interactive. Arma 3 Official Wiki: Mission Editor and Scenario Design. URL: https://community.bistudio.com/wiki/Arma_3:_Mission_Editor (дата звернення: 17.10.2025).

39. Biswas S. et al. Validation of military simulation environments: A case study of VBS3. Journal of Defense Modeling and Simulation. 2021. Vol. 18, no. 2. P. 135–150.

40. Tsuruta S. et al. A hybrid approach of neural networks and fuzzy logic for decision support. Procedia Computer Science. 2015. Vol. 60. P. 1358–1367.

41. Liao T. W. Clustering of time series data–a survey. Pattern Recognition. 2005. Vol. 38, no. 11. P. 1857–1874.

42. Shatnawi A. et al. Computational Intelligence in Logistics and Supply Chain Management: A systematic literature review. Enterprise Information Systems. 2023. Vol. 17, no. 5.

43. Chou J.-S., Truong D.-N. Multiobjective optimization of logistics distribution network with fuzzy parameters. Applied Soft Computing. 2022. Vol. 116. 108312.

44. Fayyad U., Piatetsky-Shapiro G., Smyth P. From Data Mining to Knowledge Discovery in Databases. AI Magazine. 1996. Vol. 17, no. 3. P. 37–54.

45. Pascanu R., Mikolov T., Bengio Y. On the difficulty of training recurrent neural networks. International Conference on Machine Learning. Atlanta, 2013. P. 1310–1318.

46. Kingma D. P., Ba J. Adam: A Method for Stochastic Optimization. International Conference on Learning Representations (ICLR). San Diego, 2015.

47. Шостак І. В., Данько М. І. Інтелектуальні системи підтримки прийняття рішень: теорія, методи та практика : монографія. Харків : ХНАДУ, 2018. 320 с.

48. Кондратенко Ю. П. Синтез та оптимізація нечітких систем управління : навч. посіб. Миколаїв : Вид-во ЧДУ ім. Петра Могили, 2019. 256 с.

49. Samek W. et al. Explainable AI: Interpreting, Explaining and Visualizing Deep Learning. Cham : Springer Nature, 2019. 400 p.

50. Defense Advanced Research Projects Agency (DARPA). Explainable Artificial Intelligence (XAI). URL: https://www.darpa.mil/program/explainable-artificial-intelligence (дата звернення: 10.10.2025).

ДОДАТОК А

Код файлу app.py

```
import streamlit as st
import pandas as pd
import numpy as np
import tensorflow as tf
import joblib
import json
import plotly.graph_objects as go
import plotly.express as px
import skfuzzy as fuzz
from datetime import datetime

DATA_FILE = 'logistics_data.csv'
MODEL_FILE = 'model/fuel_prediction_model.h5'
SCALER_FILE = 'model/scaler.pkl'
FEATURES_FILE = 'model/feature_names.json'

AVERAGE_DAILY_DISTANCE_KM = 200.0
MAX_BATTALION_CAPACITY = 1500.0

st.set_page_config(
    page_title="Logistics Command Center v3.0",
    page_icon=" 🛡️ ",
    layout="wide",
    initial_sidebar_state="expanded"
)

st.markdown("""
    <style>
    .stMetric {
        background-color: #f0f2f6;
        padding: 10px;
        border-radius: 5px;
        border-left: 5px solid #1f77b4;
    }
    .big-font {
        font-size:20px !important;
        font-weight: bold;
    }
    </style>
    """, unsafe_allow_html=True)

@st.cache_resource
def load_resources():
    try:
        model = tf.keras.models.load_model(MODEL_FILE, compile=False)
        scaler = joblib.load(SCALER_FILE)

        with open(FEATURES_FILE, 'r') as f:
            feature_names = json.load(f)

        return model, scaler, feature_names
    except Exception as e:
        st.error(f"Помилка завантаження ресурсів: {e}. Перевірте папку 'model/'.")
        return None, None, None
```

```
def get_gru_input(data_path, input_tempo, input_weather, input_date, feature_names, scaler):
    try:
        df = pd.read_csv(data_path)
        df['date'] = pd.to_datetime(df['date'])
        df = df.sort_values('date').reset_index(drop=True)

        last_history = df.tail(29).copy()

        current_day = pd.DataFrame({
            'date': [pd.to_datetime(input_date)],
            'operational_tempo': [input_tempo],
            'weather_condition': [input_weather],
            'fuel_consumed': [0]
        })

        full_sequence = pd.concat([last_history, current_day], ignore_index=True)

        full_sequence['day_of_year'] = full_sequence['date'].dt.dayofyear
        full_sequence['sin_time'] = np.sin(2 * np.pi * full_sequence['day_of_year'] / 365.0)
        full_sequence['cos_time'] = np.cos(2 * np.pi * full_sequence['day_of_year'] / 365.0)

        full_encoded = pd.get_dummies(full_sequence, columns=['operational_tempo', 'weather_condition'])

        df_aligned = pd.DataFrame(0, index=np.arange(len(full_encoded)), columns=feature_names)
        for col in full_encoded.columns:
            if col in df_aligned.columns:
                df_aligned[col] = full_encoded[col]

        scaled_data = scaler.transform(df_aligned)

        X_input = scaled_data.reshape(1, 30, len(feature_names))

        return X_input

    except Exception as e:
        st.error(f"Помилка обробки даних ({data_path}): {e}")
        return None

@st.cache_resource
def setup_fuzzy_system():
    fs = {}
    fs['fuel_universe'] = np.arange(0, 2501, 1)
    fs['threat_universe'] = np.arange(0, 101, 1)

    fs['fuel_low'] = fuzz.trimf(fs['fuel_universe'], [0, 0, 800])
    fs['fuel_med'] = fuzz.trimf(fs['fuel_universe'], [500, 1200, 1900])
    fs['fuel_high'] = fuzz.trimf(fs['fuel_universe'], [1500, 2500, 2500])

    fs['threat_low'] = fuzz.trimf(fs['threat_universe'], [0, 0, 30])
    fs['threat_med'] = fuzz.trimf(fs['threat_universe'], [20, 50, 80])
    fs['threat_high'] = fuzz.trimf(fs['threat_universe'], [60, 100, 100])

    return fs

model, scaler, feature_names = load_resources()
fuzzy_sys = setup_fuzzy_system()

st.sidebar.header("🕹️ Центр управління")
```

```
st.sidebar.markdown("---")

st.sidebar.subheader("1. Оперативна обстановка")
input_date = st.sidebar.date_input("Дата планування", datetime.now())
input_tempo = st.sidebar.selectbox("Темп операцій", ["Garrison", "Combat", "Patrol", "Logistics"], index=1)
input_weather = st.sidebar.selectbox("Метеоумови", ["Clear", "Rain", "Storm", "Fog"], index=0)

st.sidebar.markdown("---")

st.sidebar.subheader("2. Пріоритети командування (Ваги)")
w_risk = st.sidebar.slider("🛡️ Вага безпеки (Risk)", 0.0, 1.0, 0.6, 0.1)
w_time = st.sidebar.slider("⏱️ Вага часу (Time)", 0.0, 1.0, 0.2, 0.1)
w_fuel = st.sidebar.slider("⛽ Вага ресурсів (Cost)", 0.0, 1.0, 0.2, 0.1)

total_weight = w_risk + w_time + w_fuel
if total_weight == 0:
    w_risk, w_time, w_fuel = 0.33, 0.33, 0.33
else:
    w_risk /= total_weight
    w_time /= total_weight
    w_fuel /= total_weight

st.sidebar.caption(f"Нормалізовані ваги: Risk {w_risk:.2f} | Time {w_time:.2f} | Cost {w_fuel:.2f}")

if model is not None:

    X_input = get_gru_input(DATA_FILE, input_tempo, input_weather, input_date, feature_names, scaler)

    if X_input is not None:
        prediction_scaled = model.predict(X_input, verbose=0)

        dummy = np.zeros((1, len(feature_names)))
        dummy[0, 0] = prediction_scaled[0][0]
        predicted_fuel = scaler.inverse_transform(dummy)[0, 0]
        predicted_fuel = max(50.0, predicted_fuel)

        st.title("🛡️ Logistics Decision Support System v3.0")
        st.markdown("### 🧠 Модуль нейромережевого прогнозування (GRU + History)")

        col1, col2 = st.columns([1, 2])

        with col1:
            fig_gauge = go.Figure(go.Indicator(
                mode="gauge+number+delta",
                value=predicted_fuel,
                domain={'x': [0, 1], 'y': [0, 1]},
                title={'text': "Прогноз споживання (л)"},
                delta={'reference': 800, 'increasing': {'color': "red"}, 'decreasing': {'color': "green"}},
                gauge={
                    'axis': {'range': [0, MAX_BATTALION_CAPACITY], 'tickwidth': 1},
                    'bar': {'color': "#1f77b4"},
                    'steps': [
                        {'range': [0, 600], 'color': "#d9f0a3"},
                        {'range': [600, 1200], 'color': "#addd8e"},
                        {'range': [1200, MAX_BATTALION_CAPACITY], 'color': "#f03b20"}],
                    'threshold': {'line': {'color': "black", 'width': 4}, 'thickness': 0.75, 'value': predicted_fuel}}))
            fig_gauge.update_layout(height=250, margin=dict(l=20, r=20, t=30, b=20))
            st.plotly_chart(fig_gauge, use_container_width=True)
```

```
    with col2:
        current_fuel_intensity = predicted_fuel / AVERAGE_DAILY_DISTANCE_KM
        st.info(f"""
        **Аналітична довідка:**
        * 🗓️ База даних: **{DATA_FILE}** (використано 29 днів історії)
        * ⚔️ Поточний режим: **{input_tempo}**
        * ☁️ Погода: **{input_weather}**
        * 🤖 Прогноз GRU: **{predicted_fuel:.1f} л**
        """)
        st.markdown(f"> **Коефіцієнт складності:** `{current_fuel_intensity:.2f} л/км` (впливає на вартість маршруту)")

    st.markdown("---")

    st.markdown("#### 🗺️ Етап 2: Вибір оптимального маршруту (Fuzzy-SAW)")

    routes_data = [
        {"ID": "Route A (Шосе)", "Distance_km": 120, "Risk_Index": 80, "Time_h": 2.0},
        {"ID": "Route B (Ліс)", "Distance_km": 150, "Risk_Index": 30, "Time_h": 4.5},
        {"ID": "Route C (Село)", "Distance_km": 135, "Risk_Index": 50, "Time_h": 3.0},
        {"ID": "Route D (Об'їзд)", "Distance_km": 180, "Risk_Index": 10, "Time_h": 5.0},
    ]

    results = []
    for r in routes_data:
        est_fuel = r['Distance_km'] * current_fuel_intensity

        mu_fuel_low = fuzz.interp_membership(fuzzy_sys['fuel_universe'], fuzzy_sys['fuel_low'], est_fuel)
        mu_fuel_med = fuzz.interp_membership(fuzzy_sys['fuel_universe'], fuzzy_sys['fuel_med'], est_fuel)
        mu_fuel_high = fuzz.interp_membership(fuzzy_sys['fuel_universe'], fuzzy_sys['fuel_high'], est_fuel)
        score_fuel = (mu_fuel_low * 1.0 + mu_fuel_med * 0.5 + mu_fuel_high * 0.0)

        mu_risk_low = fuzz.interp_membership(fuzzy_sys['threat_universe'], fuzzy_sys['threat_low'], r['Risk_Index'])
        mu_risk_med = fuzz.interp_membership(fuzzy_sys['threat_universe'], fuzzy_sys['threat_med'], r['Risk_Index'])
        mu_risk_high = fuzz.interp_membership(fuzzy_sys['threat_universe'], fuzzy_sys['threat_high'], r['Risk_Index'])
        score_risk = (mu_risk_low * 1.0 + mu_risk_med * 0.5 + mu_risk_high * 0.0)

        score_time = max(0, 1 - (r['Time_h'] / 6.0))

        total_score = (score_risk * w_risk) + (score_time * w_time) + (score_fuel * w_fuel)

        results.append({
            "ID": r['ID'],
            "Distance_km": r['Distance_km'],
            "Est_Fuel_L": est_fuel,
            "Risk_Index": r['Risk_Index'],
            "Fuzzy_Score": total_score,
            "Raw_Fuel_Score": score_fuel,
            "Raw_Risk_Score": score_risk
        })

    df_routes = pd.DataFrame(results).sort_values(by='Fuzzy_Score', ascending=False)

    c1, c2 = st.columns(2)

    with c1:
        st.subheader("📋 Рейтинг маршрутів")
```

```
        st.dataframe(
            df_routes[['ID', 'Distance_km', 'Risk_Index', 'Est_Fuel_L', 'Fuzzy_Score']],
            column_config={
                "Fuzzy_Score": st.column_config.ProgressColumn("Рейтинг (Fuzzy)", format="%.3f", min_value=0,
max_value=1),
                "Est_Fuel_L": st.column_config.NumberColumn("Прогноз витрат (л)", format="%.0f"),
                "Risk_Index": st.column_config.NumberColumn("Рівень загрози", format="%d")
            },
            hide_index=True,
            use_container_width=True
        )
        best = df_routes.iloc[0]
        st.success(f"🏆 Рекомендований маршрут: **{best['ID']}**")

    with c2:
        st.subheader("📊 Аналіз факторів")
        viz_data = df_routes.copy()
        viz_data['Внесок: Безпека'] = viz_data['Raw_Risk_Score'] * w_risk
        viz_data['Внесок: Час'] = (1 - (viz_data['Distance_km']/200)) * w_time
        viz_data['Внесок: Пальне'] = viz_data['Raw_Fuel_Score'] * w_fuel

        fig_bar = px.bar(
            viz_data,
            x=['Внесок: Безпека', 'Внесок: Пальне', 'Внесок: Час'],
            y='ID',
            orientation='h',
            title="Декомпозиція прийняття рішень (Fuzzy SAW)",
            labels={'value': 'Корисність (Utility Score)', 'variable': 'Критерій'},
            color_discrete_sequence=px.colors.qualitative.Pastel
        )
        st.plotly_chart(fig_bar, use_container_width=True)

else:
    st.warning("⚠️ Очікування завантаження моделі або файлів даних...")
```

ДОДАТОК Б

Код файлу data_generator.py

```
import pandas as pd
import numpy as np

DAYS = 1095
START_DATE = '2023-01-01'

TEMPO_TYPES = ['Garrison', 'Patrol', 'Logistics', 'Combat']

TRANSITION_MATRIX = {
   'Garrison': [0.7, 0.1, 0.1, 0.1],
   'Patrol': [0.3, 0.5, 0.1, 0.1],
   'Logistics': [0.2, 0.1, 0.6, 0.1],
   'Combat': [0.1, 0.1, 0.1, 0.7]
}

BASE_CONSUMPTION = {
   'Garrison': 300,
   'Patrol': 500,
   'Logistics': 800,
   'Combat': 1200
}

WEATHER_TYPES = ['Clear', 'Rain', 'Fog', 'Storm']
WEATHER_PROBS = [0.5, 0.3, 0.15, 0.05]
WEATHER_IMPACT = {
   'Clear': 1.0,
   'Rain': 1.1,
   'Fog': 1.15,
   'Storm': 1.4
}


def generate_logistics_data():
   dates = pd.date_range(start=START_DATE, periods=DAYS)
   data = []

   current_tempo = 'Garrison'
   mud_index = 0.0
   fatigue_index = 0.0

   prev_consumption = 300

   print(f"Генеруємо {DAYS} днів розширених даних...")

   for date in dates:
      current_tempo = np.random.choice(
         TEMPO_TYPES,
         p=TRANSITION_MATRIX[current_tempo]
      )

      weather = np.random.choice(WEATHER_TYPES, p=WEATHER_PROBS)

      if weather in ['Rain', 'Storm']:
         mud_index = min(1.0, mud_index + 0.3)
```

```
        elif weather == 'Fog':
            mud_index = min(1.0, mud_index + 0.05)
        else:
            mud_index = max(0.0, mud_index - 0.2)

        if current_tempo == 'Combat':
            fatigue_index = min(1.5, fatigue_index + 0.1)
        elif current_tempo == 'Logistics':
            fatigue_index = min(1.5, fatigue_index + 0.05)
        elif current_tempo == 'Garrison':
            fatigue_index = max(0.0, fatigue_index - 0.1)
        else:
            fatigue_index = max(0.0, fatigue_index - 0.02)

        base = BASE_CONSUMPTION[current_tempo]
        weather_factor = WEATHER_IMPACT[weather]

        day_of_year = date.dayofyear
        seasonality = 1 + 0.2 * np.cos(2 * np.pi * day_of_year / 365)

        theoretical_consumption = (base * weather_factor) * (1 + 0.15 * mud_index) * (
                1 + 0.1 * fatigue_index) * seasonality

        noise_level = 30 if current_tempo == 'Garrison' else 80
        noise = np.random.normal(0, noise_level)

        final_fuel = max(100, theoretical_consumption + noise)

        data.append({
            'date': date,
            'fuel_consumed': round(final_fuel, 2),
            'operational_tempo': current_tempo,
            'weather_condition': weather,
        })

    df = pd.DataFrame(data)

    df.to_csv('logistics_data.csv', index=False)
    print("Файл 'logistics_data.csv' успішно згенеровано.")
    print(df.head())
    print(f"\nСереднє споживання: {df['fuel_consumed'].mean():.2f}")

    print("\nРозподіл станів:")
    print(df['operational_tempo'].value_counts(normalize=True))


if __name__ == "__main__":
    generate_logistics_data()
```

ДОДАТОК В

Код файлу train_model.py

```
import pandas as pd
import numpy as np
import tensorflow as tf
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import GRU, Dense, Dropout, Input
from tensorflow.keras.callbacks import EarlyStopping, ReduceLROnPlateau
from sklearn.preprocessing import MinMaxScaler
from sklearn.metrics import mean_absolute_error, mean_squared_error
import joblib
import matplotlib.pyplot as plt
import os
import json
import math

if not os.path.exists('model'):
    os.makedirs('model')

print("Завантаження даних...")
df = pd.read_csv('logistics_data.csv')
df['date'] = pd.to_datetime(df['date'])

df['day_of_year'] = df['date'].dt.dayofyear
df['sin_time'] = np.sin(2 * np.pi * df['day_of_year'] / 365.0)
df['cos_time'] = np.cos(2 * np.pi * df['day_of_year'] / 365.0)

df_processed = df.drop(['date', 'day_of_year'], axis=1)

df_encoded = pd.get_dummies(df_processed, columns=['operational_tempo', 'weather_condition'], drop_first=False)

cols = ['fuel_consumed'] + [c for c in df_encoded.columns if c != 'fuel_consumed']
df_encoded = df_encoded[cols]

feature_names = df_encoded.columns.tolist()
with open('model/feature_names.json', 'w') as f:
    json.dump(feature_names, f)
print("Список ознак збережено (включаючи sin/cos time).")

train_size = int(len(df_encoded) * 0.8)
train_df = df_encoded.iloc[:train_size]
test_df = df_encoded.iloc[train_size:]

print(f"Тренування: {len(train_df)} днів, Тест: {len(test_df)} днів")

scaler = MinMaxScaler()
scaler.fit(train_df)

train_scaled = scaler.transform(train_df)
test_scaled = scaler.transform(test_df)

joblib.dump(scaler, 'model/scaler.pkl')

SEQ_LENGTH = 30
```

```
def create_sequences(data, seq_length):
    X, y = [], []
    for i in range(len(data) - seq_length):
        X.append(data[i:i + seq_length])
        y.append(data[i + seq_length, 0])
    return np.array(X), np.array(y)


X_train, y_train = create_sequences(train_scaled, SEQ_LENGTH)
X_test, y_test = create_sequences(test_scaled, SEQ_LENGTH)

model = Sequential([
    Input(shape=(X_train.shape[1], X_train.shape[2])),

    GRU(64, return_sequences=True),
    Dropout(0.2),

    GRU(32, return_sequences=False),
    Dropout(0.2),

    Dense(1)
])

model.compile(optimizer='adam', loss='mse', metrics=['mae'])

callbacks = [
    EarlyStopping(monitor='val_loss', patience=15, restore_best_weights=True, verbose=1),
    ReduceLROnPlateau(monitor='val_loss', factor=0.5, patience=5, min_lr=0.0001, verbose=1)
]

print("Починаємо навчання...")
history = model.fit(
    X_train, y_train,
    epochs=100,
    batch_size=32,
    validation_data=(X_test, y_test),
    callbacks=callbacks,
    verbose=1
)

model.save('model/fuel_prediction_model.h5')

y_pred_scaled = model.predict(X_test)

dummy_test = np.zeros((len(y_test), scaler.n_features_in_))
dummy_test[:, 0] = y_test

dummy_pred = np.zeros((len(y_pred_scaled), scaler.n_features_in_))
dummy_pred[:, 0] = y_pred_scaled.flatten()

y_test_real = scaler.inverse_transform(dummy_test)[:, 0]
y_pred_real = scaler.inverse_transform(dummy_pred)[:, 0]

mae = mean_absolute_error(y_test_real, y_pred_real)
rmse = np.sqrt(mean_squared_error(y_test_real, y_pred_real))

print(f"\n===== ФІНАЛЬНИЙ РЕЗУЛЬТАТ =====")
print(f'MAE (Помилка): {mae:.2f} літрів")
print(f'RMSE: {rmse:.2f} літрів")
```

```
print(f"==============================\n")

plt.figure(figsize=(14, 6))
plt.plot(y_test_real, label='Реальний факт (з шумом та інерцією)', color='black', alpha=0.7)
plt.plot(y_pred_real, label='Прогноз GRU', color='red', linewidth=2)
plt.title(f'Прогноз vs Факт (MAE: {mae:.1f} л). Модель бачить приховані патерни!')
plt.legend()
plt.grid(True, alpha=0.3)
plt.savefig('training_history_fixed.png')
print("Графік збережено як training_history_fixed.png")
```