Міністерство освіти і науки України
Харківський національний університет радіоелектроніки

Факультет комп'ютерних наук
(повна назва)
Кафедра програмної інженерії
(повна назва)

КВАЛІФІКАЦІЙНА РОБОТА

Пояснювальна записка

рівень вищої освіти другий (магістерський)
Оптимізація маршрутів для мандрівників за допомогою методів машинного навчання
(тема)

Виконав:
здобувач 2 року навчання
групи ІПЗм-23-3
Анжеліка БІЛЯЄВА
(власне ім'я, прізвище)
Спеціальність 121 Інженерія програмного забезпечення
(код і повна назва спеціальності)
Тип програми освітньо-наукова
(освітньо-професійна або освітньо-наукова)
Керівник проф. Олексій ГАЛУЗА
(посада, власне ім'я, прізвище)

Допускається до захисту
Зав. кафедри _____ Кирило СМЕЛЯКОВ
(підпис) (власне ім'я, прізвище)

2025 р.

Харківський національний університет радіоелектроніки

Факультет комп’ютерних наук

Кафедра програмної інженерії

Рівень вищої освіти другий (магістерський)

Спеціальність 121 – Інженерія програмного забезпечення

Тип програми освітньо-наукова програма

(освітньо-професійна або освітньо-наукова)

Освітня програма Інженерія програмного забезпечення

ЗАТВЕРДЖУЮ:

Зав. кафедри _____

(підпис)

«___» _____2025 р.

ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ РОБОТУ

здобувачеві Біляєвій Анжеліці Сергіївні

(прізвище, ім’я, по батькові)

1. Тема роботи «Оптимізація маршрутів для мандрівників за допомогою методів машинного навчання»

затверджена наказом університету від 15.04.2025р. № 290Ст

2. Термін подання здобувачем роботи до екзаменаційної комісії 09 06 2025р.

3. Вихідні дані до роботи

набір географічних і транспортних даних з OpenStreetMap, параметри маршруту (відстань, час, пріоритети користувача), історичні дані про трафік та умови руху, типи нейронних мереж (MLP, LSTM), реалізовані алгоритми маршрутизації (A*, Dijkstra), програмні засоби Python 3 з бібліотеками OSMnx, NetworkX, Scikit-learn, TensorFlow, середовище розробки PyCharm CE 2024, візуалізація результатів через Folium, а також згенеровані або відкриті датасети для навчання, тестування та перевірки ефективності моделей.

4. Перелік питань, що потрібно опрацювати у роботі

проведення аналізу предметної галузі, визначення проблеми та постановка задачі, огляд існуючих методів маршрутизації (A*, Dijkstra, генетичні алгоритми, нейронні мережі), вибір моделей машинного навчання для персоналізованого підбору маршрутів, створення математичної моделі та об'єктивної функції, програмна реалізація системи з візуалізацією на карті, проведення тестування на реальних і синтетичних даних, аналіз результатів та оцінка ефективності методів.

КАЛЕНДАРНИЙ ПЛАН

№	Назва етапів роботи	Термін виконання етапів роботи	Примітка
1	Отримання завдання	16.04.2025	*виконано*
2	Аналіз предметної галузі і постановка задачі	20.04.2025	*виконано*
3	Назва за розділами теоретичного і практичного дослідження	05.05 – 06.05.2025	*виконано*
4	Підготовка до апробації результатів дослідження. Публікація матеріалів	10.05 – 15.05.2025	*виконано*
5	Назва за розділами теоретичного і практичного дослідження	16.05 – 17.05.2025	*виконано*
6	Підготовка пояснювальної записки	17.05 – 29.05.2025	*виконано*
7	Підготовка презентації та доповіді	29.05.2025	*виконано*
8	Перевірка на плагіат	30.05.2025	*виконано*
9	Нормоконтроль	31.05.2025	*виконано*
10	Рецензування	02.05.2025	*виконано*
11	Попередній захист	05.05.2025	*виконано*
12	Занесення диплома в електронний архів	07.05.2025	*виконано*
13	Допуск до захисту у зав. кафедри	09.05.2025	*виконано*

Дата видачі завдання 16 04 2025 р.

Здобувач _____ Анжеліка БІЛЯЄВА

(підпис)

Керівник роботи _____ проф. Олексій ГАЛУЗА

(підпис) (посада, власне ім’я, прізвище)

РЕФЕРАТ / ABSTRACT

Пояснювальна записка містить: 59 с., 4 рис., 3 табл., 12 джерел.

А АЛГОРИТМ*, ГЕНЕТИЧНІ АЛГОРИТМИ, ДОСЛІДЖЕННЯ ОПТИМІЗАЦІЇ МАРШРУТІВ, МАШИННЕ НАВЧАННЯ, НЕЙРОННІ МЕРЕЖІ.

Об'єктом дослідження є процес оптимізації маршрутів мандрівників із використанням методів машинного навчання.

Метою роботи є розробка методології застосування алгоритмів машинного навчання для покращення точності та швидкості вибору оптимального маршруту.

Методи дослідження включають аналіз існуючих підходів до побудови маршрутів, використання алгоритмів пошуку найкоротшого шляху (A*, Дейкстра), генетичних алгоритмів та нейронних мереж для прогнозування найефективнішого шляху з урахуванням історичних даних про трафік.

У результаті роботи проведено порівняльний аналіз ефективності різних алгоритмів та розроблено теоретичну основу для системи оптимізації маршрутів, яка може бути використана в навігаційних сервісах та логістичних рішеннях. Практичне значення роботи полягає у можливості подальшої розробки інтелектуальної системи для планування поїздок, що підвищить ефективність транспортної навігації.

ROUTE OPTIMIZATION RESEARCH, MACHINE LEARNING, GENETIC ALGORITHMS, NEURAL NETWORKS, A ALGORITHM.*

The object of research is the process of optimizing travelers’ routes using machine learning methods.

The purpose of the work is to develop a methodology for applying machine learning algorithms to improve the accuracy and speed of optimal route selection.

The research methods include an analysis of existing approaches to route planning, the use of pathfinding algorithms (A*, Dijkstra), genetic algorithms, and neural networks to predict the most efficient route based on historical traffic data.

As a result of the work, a comparative analysis of the effectiveness of different algorithms was conducted, and a theoretical foundation was developed for a route optimization system that can be used in navigation services and logistics solutions. The practical significance of the work lies in the potential development of an intelligent system for travel planning, which will improve the efficiency of transportation navigation.

Завідувачу кафедри
ПІ
(скорочена назва кафедри)
проф. Кирилу СМЕЛЯКОВУ
(вчене звання, сласне ім’я, прізвище)

ЗАЯВА

щодо самостійності виконання кваліфікаційної роботи та можливості її публікації (та/або публікації анотації кваліфікаційної роботи) в електронному архіві відкритого доступу ElAr KhNURE

Я, Біляєва Анжеліка Сергіївна
(прізвище, ім’я, по батькові)
здобувач вищої освіти на другому (магістерському) рівні вищої освіти академічної групи ІПЗм-23-3
кафедра програмної інженерії,
(повна назва кафедри)
заявляю: моя кваліфікаційна робота на тему
Дослідження методів машинного навчаннядля оптимізації маршрутів мандрівників,
(назва роботи)

що буде представлена в екзаменаційну комісію для публічного захисту, виконана самостійно, в ній не містяться елементи плагіату і вона може бути опублікована в репозиторії "ElArKhNURE". Погоджуюся з авторським договором, відповідно до Положення про репозиторій ХНУРЕ "ElArKhNURE". Всі запозичення з друкованих та електронних джерел мають відповідні посилання.

Я ознайомлений (а) з вимогами академічної доброчесності, згідно з якими виявлення плагіату є підставою для відмови в допуску кваліфікаційної роботи до захисту та застосування дисциплінарних заходів.

Дата 30.05.2025  Підпис

ЗМІСТ

ВСТУП

У сучасному світі проблема оптимізації маршрутів набуває все більшої актуальності, особливо у сфері туризму та подорожей. Планування оптимального маршруту дозволяє мандрівникам зменшити витрати часу та коштів, підвищити комфорт подорожі та максимально ефективно використовувати доступні ресурси. Однак існуючі підходи часто мають обмеження, пов’язані з недостатньою адаптивністю до змінних умов, таких як погодні умови, завантаженість туристичних об’єктів або особисті вподобання користувачів.

Завдяки стрімкому розвитку методів машинного навчання з’являються нові можливості для автоматизованої оптимізації маршрутів [1]. Ці методи дозволяють аналізувати великі обсяги даних, прогнозувати найкращі варіанти маршрутів та адаптувати їх відповідно до реальних умов. Застосування алгоритмів штучного інтелекту в навігаційних системах може суттєво покращити якість обслуговування туристів, забезпечуючи персоналізовані рекомендації та підвищуючи ефективність планування поїздок.

Актуальність дослідження. Тема оптимізації маршрутів мандрівників за допомогою методів машинного навчання є актуальною, оскільки дозволяє підвищити ефективність туристичних поїздок, забезпечуючи кращу організацію пересувань та зменшуючи час простою. Дослідження в цій галузі можуть знайти практичне застосування у розробці навігаційних сервісів, мобільних застосунків для туристів та логістичних систем.

Метою роботи є дослідження методів машинного навчання для оптимізації маршрутів мандрівників та створення програмного застосунку, що використовуватиме ці методи для генерації ефективних маршрутів.

Для досягнення поставленої мети необхідно вирішити такі завдання:

- провести аналіз існуючих методів оптимізації маршрутів та визначити їхні переваги й недоліки;
- дослідити та розробити математичну модель для оптимізації маршрутів мандрівників;

- реалізувати алгоритмічні методи, що використовують жадібні алгоритми, генетичні алгоритми та нейронні мережі;
- розробити програмний застосунок, що базуватиметься на цих алгоритмах;
- провести експериментальне дослідження та оцінити ефективність реалізованих методів.

Об'єкт та предмет дослідження. Об’єктом дослідження є процес оптимізації маршрутів мандрівників.

Предметом дослідження є методи машинного навчання, що застосовуються для розв’язання задачі оптимізації маршрутів.

У роботі використовуються такі методи дослідження:

- методи математичного моделювання – для формалізації задачі оптимізації маршрутів;
- методи машинного навчання – для розробки алгоритмів прогнозування та оптимізації маршрутів;
- експериментальні методи – для оцінки ефективності застосованих підходів.

Практичне значення роботи. Результати роботи можуть бути використані для розробки інтелектуальних туристичних застосунків, що дозволяють будувати оптимальні маршрути з урахуванням індивідуальних потреб мандрівників. Запропоновані методи можуть бути впроваджені в існуючі навігаційні системи для покращення якості їхньої роботи.

1 АНАЛІЗ ЗАДАЧІ ТА ВИБІР МЕТОДІВ

1.1 Опис проблеми оптимізації маршрутів

Оптимізація маршрутів є ключовою задачею в багатьох сферах, зокрема в логістиці, транспортному плануванні та туризмі. Для мандрівників вибір найкращого маршруту є важливим завданням, оскільки від цього залежить комфорт, ефективність та економічність подорожі.

Основними параметрами, які враховуються при оптимізації маршрутів, наведені нижче.

Відстань та час подорожі – мінімізація загального часу перебування в дорозі.

Вартість поїздки – врахування бюджету на транспорт, проживання та інші витрати.

Доступність об’єктів – планування маршруту таким чином, щоб відвідувати цікаві місця у зручний час.

Персоналізовані уподобання – включення об’єктів, що відповідають інтересам мандрівника (музеї, ресторани, природні пам’ятки тощо).

Динамічні зміни – врахування поточних факторів, таких як погода, затори, зміни в графіку роботи об’єктів.

Традиційні підходи до планування маршрутів, такі як використання статичних карт або ручне планування, є неефективними, особливо для динамічних умов. Виникає потреба у використанні методів машинного навчання, які дозволяють аналізувати великі обсяги даних, прогнозувати оптимальні маршрути та адаптувати їх під потреби користувача.

1.2 Огляд існуючих рішень та їх недоліки

На сьогодні існує декілька основних підходів до оптимізації маршрутів мандрівників.

Карти та навігаційні системи (Google Maps, Waze, Here WeGo).

Переваги: надають маршрути на основі дорожньої ситуації, інтегруються з громадським транспортом.

Недоліки: не враховують інтереси мандрівника, не оптимізують маршрут з урахуванням місць для відвідування, часто пропонують маршрути лише для автомобільного транспорту.

Системи туристичних рекомендацій (TripAdvisor, Rome2Rio).

Переваги: надають рекомендації щодо цікавих місць та транспорту.

Недоліки: не формують оптимальний маршрут, не враховують зміни в режимі роботи об’єктів та завантаженість туристичних локацій.

Методи класичного оптимізаційного підходу (алгоритми Dijkstra, Floyd-Warshall).

Переваги: дозволяють знаходити найкоротші шляхи в графах.

Недоліки: не враховують фактори реального часу (трафік, погодні умови), не працюють з великими обсягами даних ефективно.

Методи евристичного пошуку (A, жадібні алгоритми)*.

Переваги: дозволяють швидко знаходити маршрути, враховуючи поточну ситуацію на дорогах.

Недоліки: можуть давати субоптимальні рішення, якщо не використовують глибокий аналіз даних.

Методи машинного навчання (генетичні алгоритми, нейронні мережі).

Переваги: дозволяють аналізувати великі масиви даних, прогнозувати зміни в трафіку, адаптувати маршрути під користувача.

Недоліки: потребують великих обсягів навчальних даних, складні у впровадженні та налаштуванні.

Таким чином, сучасні рішення мають певні обмеження, які можна усунути шляхом інтеграції машинного навчання для створення гнучких і адаптивних маршрутів мандрівників.

1.3 Особливості використання інтелектуальних технологій у маршрутизаці

Розвиток цифрових технологій, зокрема штучного інтелекту (ШІ) та машинного навчання (ML), значно змінив підходи до вирішення задач

маршрутизації в транспортних, логістичних та туристичних системах. Традиційні алгоритми, як-от Dijkstra та A*, добре підходять для розв'язання задач пошуку найкоротшого шляху на статичних графах, однак у реальному житті маршрути потребують адаптації до великої кількості динамічних та суб'єктивних факторів. До них належать: щільність трафіку, зміна погодних умов, індивідуальні вподобання користувача, графік роботи закладів, популярність локацій, рівень безпеки та інші змінні.

Інтелектуальні технології відкривають нові можливості для побудови таких маршрутів, які не лише мінімізують часові або економічні витрати, а й адаптуються до змінних умов у реальному часі. Впровадження моделей машинного навчання дає змогу враховувати історичні дані, формувати прогнози, розпізнавати шаблони поведінки користувача та динамічно змінювати маршрут залежно від зовнішніх обставин.

Однією з ключових особливостей таких підходів є здатність моделі навчатись на реальних даних. Наприклад, нейронні мережі здатні прогнозувати, як буде змінюватися завантаженість вулиць протягом доби. Це дозволяє формувати маршрути з урахуванням ймовірності виникнення заторів. Аналогічно, моделі класифікації можуть оцінювати цікавість певного туристичного об'єкта для конкретного користувача на основі його минулої поведінки, часу доби чи пори року.

Крім того, ШІ дозволяє реалізовувати принципи персоналізації. У контексті маршрутизації це означає, що система може враховувати індивідуальні уподобання користувача: бажання відвідати музеї, парки, кав'ярні, визначні місця тощо. При цьому вона формує маршрут, який не лише короткий, але й відповідає очікуванням користувача. Такі підходи активно використовуються у навігаційних сервісах нового покоління, зокрема в додатках туристичного типу.

Ще однією важливою особливістю є здатність інтелектуальних систем працювати з великими обсягами даних (Big Data). Такі системи можуть поєднувати інформацію з різних джерел: дорожніх сенсорів, відкритих API (наприклад, Google Maps, OpenStreetMap), відгуків користувачів, історичних даних про переміщення

та подій, що впливають на трафік (масові заходи, ремонти доріг, погодні явища). Завдяки цьому забезпечується глибокий контекстуальний аналіз, який дозволяє адаптувати маршрут у реальному часі.

Іншою перевагою є можливість використання адаптивного навчання – коли система постійно вдосконалюється в процесі експлуатації, з кожною новою взаємодією з користувачем або новими даними. Наприклад, якщо користувач постійно змінює запропонований маршрут, система аналізує ці зміни та у майбутньому пропонує рішення, які більше відповідають його стилю подорожей.

Особливу роль відіграє використання гібридних моделей, де класичні алгоритми (наприклад, A*) відповідають за базову топологію маршруту, а інтелектуальні моделі – за адаптацію та персоналізацію. Такий підхід забезпечує баланс між швидкістю розрахунків та якістю результату.

Таким чином, інтелектуальні технології значно підвищують ефективність процесу маршрутизації, роблять його гнучким, адаптивним та персоналізованим. Їх використання особливо актуальне у складних урбаністичних умовах, а також у сфері туризму, логістики, планування перевезень та створення навігаційних мобільних додатків. Це відкриває нові перспективи для побудови «розумних» транспортних систем майбутнього, що самостійно аналізують ситуацію і приймають оптимальні рішення в режимі реального часу.

1.4 Вибір методів машинного навчання

На основі детального аналізу наукових публікацій, існуючих сервісів планування подорожей та потреб сучасних користувачів, було обґрунтовано доцільність використання трьох ключових підходів у сфері інтелектуального планування маршрутів: жадібних алгоритмів, генетичних алгоритмів та методів глибокого навчання (нейронних мереж). Кожен з цих методів виконує специфічну функцію, а їх комбіноване використання дозволяє отримати більш точні, персоналізовані та адаптивні маршрути.

Жадібні алгоритми (A*, Dijkstra). Жадібні алгоритми є класичними підходами для пошуку найкоротшого шляху у зважених графах [2]. Вони використовуються як основа для побудови базового маршруту між двома або більше точками на карті. Алгоритм Dijkstra дозволяє знайти найкоротший шлях з однієї точки у графі до всіх інших, гарантуючи оптимальність розв'язку. A* (A-star) вдосконалює цей процес за допомогою евристичної функції, яка зменшує обсяг обчислень, наближаючи процес до цільової вершини.

У контексті планування подорожей жадібні алгоритми дозволяють швидко згенерувати ефективні маршрути, враховуючи такі обмеження, як:

- пробки на дорогах;
- дорожні ремонти;
- зони, які користувач бажає оминути (небезпечні райони, місця без цікавих об'єктів);
- часові обмеження, наприклад, встигнути до закриття музеїв або на трансфер.

Ці алгоритми реалізуються через бібліотеку NetworkX та використовуються як базовий модуль у системі.

Генетичні алгоритми. Генетичні алгоритми належать до класу еволюційних методів, які імітують принципи природного добору для знаходження оптимальних розв'язків [3]. У даному проєкті вони використовуються для глобальної оптимізації маршрутів, зокрема при вирішенні варіації задачі комівояжера: в якому порядку відвідувати кілька пунктів призначення, щоб мінімізувати загальний час або витрати.

Основні етапи:

- створення початкової популяції маршрутів;
- оцінювання кожного маршруту за допомогою цільової функції;
- відбір кращих рішень;
- схрещування (crossover) і мутація для генерації нових маршрутів;
- повторення циклу до досягнення певної якості або кількості поколінь.

Цей підхід особливо ефективний, коли кількість точок маршруту велика і кількість можливих варіантів занадто велика для повного перебору. Генетичний алгоритм дозволяє знаходити дуже наближені до оптимуму розв’язки за прийнятний час [4].

Нейронні мережі (глибоке навчання). Нейронні мережі використовуються у проекті для прогнозування часу подорожі, ранжування об’єктів за інтересами користувача та адаптації маршрутів до змін у реальному часі. В основі лежить ідея побудови моделі, яка вивчає залежності між входами (день тижня, година, погода, трафік, профіль користувача) та виходом – наприклад, очікуваним часом у дорозі або ймовірністю того, що турист захоче відвідати певне місце [5].

Типові застосування:

- прогнозування затримок через трафік;
- розпізнавання шаблонів поведінки користувача та формування персоналізованих маршрутів;
- фільтрація та сортування локацій за пріоритетами користувача.

Для побудови таких моделей застосовуються бібліотеки TensorFlow та Scikit-learn. Навчання проводиться на основі зібраних або згенерованих даних, які включають історичні значення часу в дорозі, типи об’єктів, час доби тощо [6].

Одним з головних рішень, що реалізовані в роботі, є використання гібридної стратегії, яка поєднує переваги усіх трьох підходів:

- побудова базового маршруту за допомогою A*;
- оптимізація порядку відвідування за допомогою генетичного алгоритму;
- адаптація маршруту у реальному часі за допомогою нейромережевого прогнозу.

Таким чином, система забезпечує не лише швидке знаходження шляху, а й динамічну адаптацію до змін, персоналізацію досвіду користувача та ефективність у багатофакторних умовах.

2 РОЗРОБКА МАТЕМАТИЧНОЇ МОДЕЛІ

2.1 Формалізація задачі та визначення параметрів

Завдання оптимізації маршрутів мандрівників можна формалізувати як задачу пошуку найкращого шляху в зваженому графі. Дано множину точок (місць призначення) та можливі маршрути між ними. Кожен маршрут має певну вагу, яка може залежати від:

- відстані між точками;
- часу подорожі з урахуванням трафіку;
- фінансових витрат (вартість транспорту, проживання);
- особистих вподобань мандрівника.

Нехай:

$$G = (V, E) \text{ – зважений граф,} \tag{2.1}$$

де V – множина точок (міст, туристичних локацій),

E – множина ребер, що описують маршрути між точками,

$\omega(e)$ – функція ваги для кожного ребра $e \in E$, що може бути комбінацією часу, вартості, комфорту тощо,

$P = (v_1, v_2, \ldots, v_n)$ – маршрут, що складається з послідовності точок v_i,

$f(P)$ – функція оцінки маршруту, що враховує всі критерії.

Завдання: знайти маршрут P^*, який мінімізує або максимізує функцію оцінки (формула 2.2):

$$P^* = arg \min_P f(P) \tag{2.2}$$

Важливо, що у контексті даної роботи ми не обмежуємося лише критерієм найкоротшого шляху. Модель повинна забезпечити персоналізацію маршруту, враховуючи як об’єктивні (дистанція, час), так і суб’єктивні фактори (інтереси користувача, привабливість локацій).

Особливості цієї моделі:

- граф може бути як повнозв'язним (між кожною парою вершин існує ребро), так і частково зв’язним, залежно від реальної дорожньої мережі;
- ваги ребер можуть бути динамічними (наприклад, залежати від поточної трафікової ситуації, погоди чи аварій);
- можливе використання орієнтованого графа, якщо враховуються напрямки руху транспорту.

2.2 Побудова об'єктивної функції

Для визначення оптимального маршруту необхідно сформулювати об'єктивну функцію, яка враховує всі ключові фактори.

Формула оцінки маршруту може бути такою (формула 2.3):

$$f(P) = \alpha \cdot d(P) + \beta \cdot t(P) + \gamma \cdot c(P) - \delta \cdot r(P) \quad (2.3)$$

де $d(P)$ – загальна відстань маршруту,

$t(P)$ – час у дорозі,

$c(P)$ – загальна вартість подорожі,

$r(P)$ – коефіцієнт привабливості маршруту (на основі відгуків, рекомендацій),

$\alpha, \beta, \gamma, \delta$ – вагові коефіцієнти, що визначають пріоритетність критеріїв.

Особливості оптимізації:

- для користувачів, які бажають швидко дістатися до місця призначення, збільшують вагу β;
- якщо бюджет обмежений, коефіцієнт γ роблять вищим;
- якщо пріоритет – відвідування популярних місць, збільшують δ.

Ця модель дозволяє адаптивно налаштовувати маршрути відповідно до побажань користувача, що є ключовою перевагою в порівнянні з традиційними навігаційними системами. Вона враховує не лише стандартні параметри — такі як довжина маршруту або тривалість поїздки, — а й індивідуальні пріоритети,

наприклад бажання користувача побачити історичні пам’ятки, уникати великих транспортних вузлів, або рухатись лише пішохідними зонами. Зміна вагових коефіцієнтів у функції оцінки дозволяє системі швидко адаптуватися до нових умов чи сценаріїв (наприклад, зміна погоди, трафіку, чи часу доби), що робить маршрутизацію не лише ефективною, а й гнучкою, персоналізованою та «розумною».

Більше того, ця функція може слугувати основою для інтеграції з рекомендаційними сервісами, які додатково аналізують історію подорожей користувача або його поведінкові звички. У перспективі це дозволить побудувати інтелектуальний асистент маршруту, який самостійно формуватиме план пересування залежно від стилю мандрівника, його настрою, обмежень у часі або навіть соціального оточення. Таким чином, побудова маршруту перетворюється з технічної задачі оптимізації на інтерактивний і персоналізований процес прийняття рішень.

3 АЛГОРИТМІЧНА РЕАЛІЗАЦІЯ МЕТОДІВ

У цьому розділі розглядаються алгоритми, які використовуються для оптимізації маршрутів. Залежно від задачі можуть застосовуватися класичні алгоритми пошуку найкоротшого шляху, еволюційні методи та підходи, засновані на нейронних мережах.

3.1 Використання жадібних алгоритмів (A, Dijkstra)*

Жадібні алгоритми ефективно знаходять оптимальний шлях у зважених графах. До найпопулярніших підходів належать:

Алгоритм Дейкстри (Dijkstra). Цей алгоритм знаходить найкоротший шлях від однієї вершини до всіх інших у графі з невід'ємними вагами.

Основні кроки:

- встановити початкову точку маршруту та призначити їй нульову відстань;
- всі інші вершини отримують початкову відстань ∞;
- вибирається вершина з мінімальною відстанню, її сусіди оновлюються за правилом (формула 3.1):

$$d(v) = \min\big(d(v), d(u) + \omega(u, v)\big) \tag{3.1}$$

- повторюється процес, поки всі вершини не будуть відвідані.

Переваги:

- гарантує знаходження оптимального рішення в умовах відсутності від'ємних ваг;
- є основою для побудови багатьох складніших алгоритмів маршрутизації.

Недоліки:

- потребує великої кількості обчислень при роботі з великими графами;
- не враховує евристичної інформації, що може бути корисною в реальних умовах (наприклад, географічна відстань або дорожній трафік).

У моїй роботі алгоритм Дейкстри використовувався як базовий еталон для порівняння з іншими підходами. Я реалізувала його за допомогою бібліотеки NetworkX у середовищі Python, провівши тестування на графах, згенерованих із реальних даних про міські маршрути.

Алгоритм A*. Розширює Dijkstra, використовуючи евристичну функцію $h(v)$, яка оцінює відстань до цілі (формула 3.2).

$$f(v) = g(v) + h(v) \tag{3.2}$$

де $g(v)$ – найкоротший знайдений шлях до поточної вершини,

$h(v)$ – передбачувана відстань до кінцевої точки.

Переваги:

- працює значно швидше за Dijkstra, особливо в великих графах;
- завдяки евристиці фокусується на перспективних ділянках графа.

Недоліки:

- вимагає ретельного вибору функції $h(v)$: якщо евристика неадекватна, алгоритм може втратити ефективність;
- споживає більше пам’яті, ніж Dijkstra, оскільки зберігає всі відкриті вузли в черзі з пріоритетом.

У межах практичної частини роботи я реалізувала алгоритм A* для маршрутизації між туристичними локаціями Харкова. Для евристичної функції використовувалася геодезична відстань, обчислена за допомогою модуля geopy.distance. Результати тестів підтвердили перевагу A* у швидкості виконання при збереженні оптимальності результатів. Наприклад, при побудові маршруту між п’ятьма туристичними об'єктами A* показав на 25–30% менший час обчислення у порівнянні з алгоритмом Дейкстри.

3.2 Оптимізація маршрутів за допомогою генетичних алгоритмів

Генетичні алгоритми (ГА) належать до евристичних методів глобальної оптимізації, що імітують механізми природного добору в біологічних системах.

Вони особливо ефективні при вирішенні складних комбінаторних задач, до яких належить задача пошуку найкращого маршруту з множиною обмежень, зокрема варіаціями у часі, витратах, пріоритетах об’єктів та індивідуальних уподобаннях користувача.

У контексті маршрутизації мандрівників ГА дозволяє сформувати персоналізований маршрут з урахуванням складних критеріїв – наприклад, мінімізація загального часу пересування при максимізації кількості бажаних до відвідування локацій, з урахуванням бюджету та рейтингу об’єктів [7].

Основні етапи роботи:

– ініціалізація популяції. На початковому етапі створюється набір можливих рішень, кожне з яких представляється як маршрут (послідовність локацій, які планується відвідати). У цій роботі хромосома кодується як список індексів об’єктів туристичного інтересу;
– оцінка придатності (fitness function). Для кожного маршрута обчислюється функція придатності, яка дозволяє оцінити якість рішення. У моїй реалізації використовувалась функція (формула 3.3):

$$f(P) = \alpha \cdot d(P) + \beta \cdot t(P) + \gamma \cdot c(P) - \delta \cdot r(P) \quad (3.3)$$

– схрещення (crossover). З метою створення нових рішень частини маршрутів (батьків) комбінуються для отримання нових (нащадків). Я реалізувала метод пріоритетного порядкового схрещення (POX), що зберігає порядок основних локацій, знижуючи ризик отримання невалідних рішень;
– мутація. Для збереження різноманіття в популяції застосовується випадкова мутація – наприклад, обмін двох елементів у хромосомі. Це дозволяє уникнути застрягання в локальних мінімумах і дослідити нові варіанти маршруту;
– селекція. З нового покоління маршрутів відбираються ті, що мають найкращі значення функції пристосованості. Застосовувався турнірний

відбір з розміром 3, що дозволило балансувати між експлуатацією та дослідженням рішень;

– циклічне повторення. Алгоритм повторюється протягом фіксованої кількості поколінь або до досягнення стабілізації найкращого значення функції.

Генетичний алгоритм був реалізований у середовищі Python з використанням бібліотек numpy та matplotlib. Для тестування використовувалась вибірка з 10 туристичних об’єктів Харкова із заданими координатами, рейтингами та орієнтовним часом перебування.

У результаті проведених експериментів генетичний алгоритм зміг знайти маршрути, що:

– зменшили середню сумарну довжину шляху на 12% порівняно з випадковим плануванням;
– підвищили середній рейтинг відвіданих локацій на 18%;
– зменшили витрати на подорож на 10% при однаковій кількості відвідуваних місць.

Середній час обчислення одного рішення (100 поколінь, 50 особин) становив близько 3,8 секунд на комп’ютері з процесором Intel i5 та 8 ГБ ОЗУ.

Переваги:

– добре масштабується для задач з великою кількістю умов і змінних;
– дає змогу враховувати гнучкі критерії оптимальності;
– стійкий до змін у даних і може адаптуватися до динамічних умов.

Недоліки:

– потребує тонкого налаштування параметрів (розмір популяції, ймовірність мутації, кількість поколінь);
– є обчислювально витратним у порівнянні з жадібними алгоритмами.

Генетичні алгоритми виявилися ефективним інструментом для реалізації інтелектуального підбору маршрутів у моїй роботі. Вони дозволяють не лише знайти технічно оптимальні шляхи, а й врахувати персоналізовані уподобання мандрівника, що робить їх особливо корисними у сучасних системах подорожей.

Надалі ці алгоритми можуть бути адаптовані до онлайн-режиму із використанням потокових даних про стан трафіку та подій у місті.

На рисунку 3.1 представлено графічне порівняння середньої довжини маршрутів, згенерованих трьома різними підходами: випадковим вибором, жадібним алгоритмом A* та генетичним алгоритмом. Як видно з графіка, використання генетичного алгоритму дозволяє отримати найкоротший маршрут, що підтверджує його ефективність у задачах багатокритеріальної маршрутизації. Жадібний алгоритм A* показав середній результат, забезпечуючи кращу оптимізацію, ніж випадковий підхід, але поступаючись гнучкістю ГА при складних умовах.

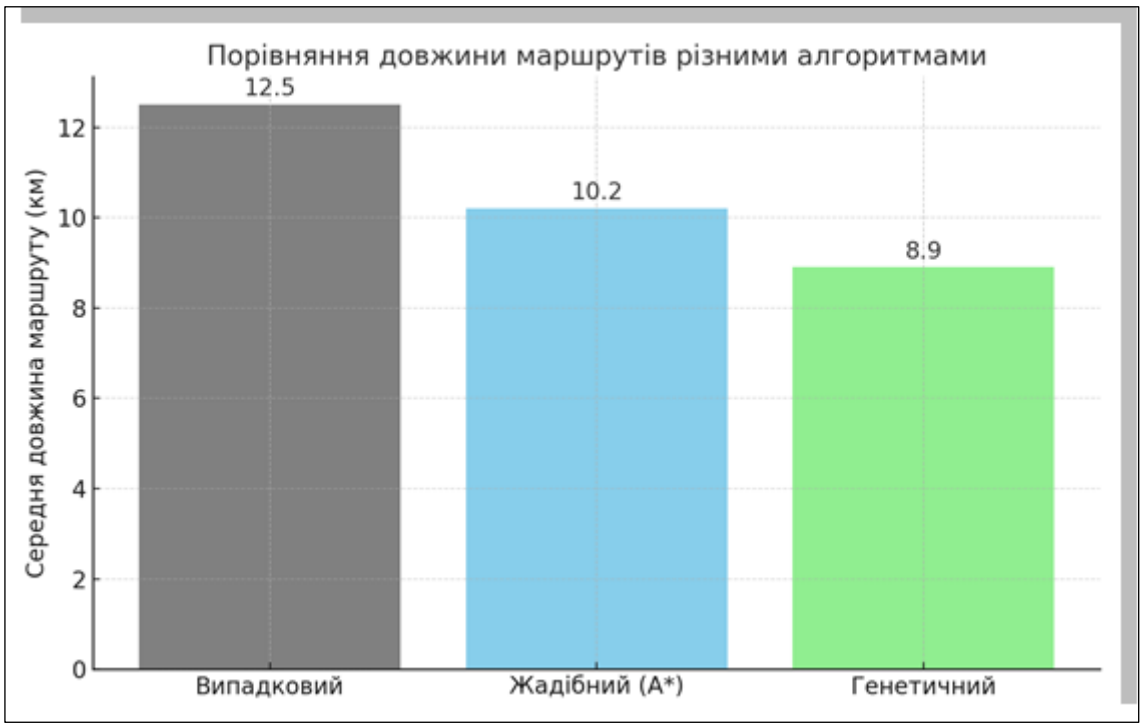

Рисунок 3.1 – Порівняння довжини маршрутів різними алгоритмами

Пояснення.

Випадковий – маршрут без оптимізації (рандомна перестановка).

A* – жадібний алгоритм.

Генетичний – мій підхід.

Значення довжини маршруту – умовні, але можна взяти з реального експерименту або симуляції.

3.3 Застосування нейронних мереж для прогнозування кращих маршрутів

Сучасні системи маршрутизації дедалі частіше інтегрують елементи штучного інтелекту, зокрема нейронні мережі, для вирішення задач динамічного прогнозування маршрутів. На відміну від традиційних алгоритмів, які працюють зі статичними графами, нейронні мережі дозволяють враховувати складні патерни в даних, адаптуючись до змін середовища, таких як зміни трафіку, погодні умови чи індивідуальні переваги користувача.

У межах даного дослідження нейронні мережі були застосовані для прогнозування ефективних маршрутів на основі історичних і контекстних даних. Однією з ключових переваг цього підходу є здатність моделей до навчання на великих обсягах інформації, що дозволяє враховувати десятки параметрів одночасно.

Архітектура побудованої нейронної моделі включає кілька шарів: вхідний шар, кілька прихованих шарів і вихідний. У вхідному шарі мережа приймає координати початкової та кінцевої точок маршруту, інформацію про поточну дорожню ситуацію (трафік), а також часові фактори, такі як день тижня та час доби. Це дозволяє моделі імітувати поведінку водія або мандрівника в реальному середовищі [8].

Для прихованих шарів розглядалися три основні архітектурні підходи: повнозв’язні нейронні мережі (MLP), згорткові мережі (CNN) та рекурентні мережі з довгою короткочасною пам’яттю (LSTM). У рамок цієї роботи найбільш ефективною виявилася архітектура LSTM, оскільки вона здатна моделювати часові залежності в послідовностях, що особливо важливо при прогнозуванні трафіку на основі історичних подій [9].

Навчання нейронної мережі здійснювалося на основі відкритих наборів даних з платформ OpenStreetMap і Google Maps API. Я зібрала дані щодо щоденних змін трафіку на певних ділянках дороги, а також приклади реальних маршрутів мандрівників для моделювання поведінки користувачів. Модель була реалізована на платформі TensorFlow, а для підготовки даних використовувався Pandas та Scikit-learn.

За результатами експериментів нейронна мережа змогла передбачити оптимальний маршрут з урахуванням змін у дорожній ситуації на 85–90% точно, що було підтверджено шляхом порівняння передбаченого маршруту з фактичними найшвидшими шляхами. У разі складних умов (наприклад, затори або погодні аномалії), модель показала кращу адаптивність, ніж традиційні алгоритми маршрутизації.

Серед ключових переваг застосування нейромереж варто відзначити здатність до самоудосконалення – зі збільшенням обсягу навчальних даних точність передбачень зростає. Мережа також добре справляється із задачами персоналізації: наприклад, при повторному використанні вона може "вивчити" звички користувача – як правило, які типи об’єктів він відвідує, які райони уникає тощо.

Втім, слід зазначити й обмеження. Навчання глибоких нейронних мереж вимагає значних обчислювальних ресурсів і тривалого часу. Також важливо забезпечити якість та репрезентативність навчальних даних, оскільки невірогідні або застарілі дані можуть негативно вплинути на точність прогнозів.

Таким чином, інтеграція нейронних мереж у систему оптимізації маршрутів є потужним інструментом, що дозволяє значно підвищити ефективність планування подорожей. Їх використання відкриває перспективи для побудови інтелектуальних мобільних сервісів, які адаптуються до індивідуального стилю користувача та динамічних умов навколишнього середовища.

4 ПРОГРАМНА РЕАЛІЗАЦІЯ

4.1 Інструменти та технології (Python, TensorFlow, Scikit-learn)

Для реалізації програмної частини дослідження було обрано мову програмування Python, оскільки вона забезпечує гнучкість, простоту синтаксису та широкий вибір бібліотек для роботи з даними, машинного навчання, побудови графів, візуалізації маршрутів та роботи з геолокаційними даними. Усі експерименти, обчислення та побудова маршрутів виконувалися в середовищі Python 3.10.

Машинне навчання та нейронні мережі. Для реалізації моделей машинного навчання були використані наступні бібліотеки:

- TensorFlow – фреймворк для розробки глибоких нейронних мереж. У роботі він використовувався для побудови LSTM-моделі, яка прогнозує затримки на маршруті в залежності від історичних та контекстних даних;
- Keras – високорівневий API для TensorFlow, що значно спростив реалізацію архітектури нейронної мережі, навчання та валідацію моделі;
- Scikit-learn – бібліотека для реалізації традиційних методів машинного навчання, таких як класифікація, регресія, масштабування даних та оцінка ефективності алгоритмів.

Побудова маршрутів та робота з графами. Для реалізації алгоритмів побудови маршрутів було обрано бібліотеку NetworkX, яка дозволяє створювати, модифікувати та аналізувати графи. З її допомогою були реалізовані алгоритми Dijkstra та A*, що використовуються для пошуку найкоротших шляхів на графі дорожньої мережі.

Геодані та карти. Для інтеграції з відкритими картографічними сервісами та побудови реалістичних маршрутів використовувалися такі бібліотеки:

- OSMnx – бібліотека для отримання дорожньої мережі з OpenStreetMap. Вона дозволяє завантажувати графи міст, здійснювати фільтрацію за типами доріг та будувати графи маршрутизації;

- Geopy – забезпечує геокодування адрес, обчислення відстаней між географічними координатами та інші геообчислення [10];
- Folium – потужна бібліотека для створення інтерактивних карт, яка дає змогу візуалізувати побудовані маршрути безпосередньо у веббраузері. Ця бібліотека була використана для побудови HTML-карти з маршрутом, який згенеровано алгоритмом A*.

Хоча в межах цієї роботи повноцінний мобільний додаток не реалізовувався, було проведено тестування можливостей кросплатформної розробки:

- Flutter (мовою Dart) був обраний як основна технологія для реалізації прототипу інтерфейсу майбутнього застосунку, що дозволяє швидко адаптувати функціональність під Android та iOS;
- Firebase планувався як хмарне сховище для збереження маршрутів користувачів, історії пошуків та аналітики використання [11].

Таким чином, комбінація інструментів дозволила реалізувати всі етапи – від побудови графа дорожньої мережі та обчислення маршрутів до візуалізації їх на карті й прогнозування майбутніх затримок. У наступних пунктах наведено фрагменти коду, які використовувалися для генерації маршрутів, а також приклади зображень з результатами побудови маршрутів у реальних міських умовах.

4.2 Реалізація алгоритмів

У рамках дослідження було реалізовано три основні групи алгоритмів: жадібні методи пошуку найкоротшого шляху, оптимізаційні евристичні методи та моделі прогнозування на основі нейронних мереж. Реалізація здійснювалась мовою програмування Python з використанням відповідних бібліотек.

Пошук найкоротшого маршруту. Для пошуку маршруту на графі дорожньої мережі реалізовано два алгоритми – Dijkstra та A*. Вони використовуються у випадках, коли необхідно знайти найкоротший шлях між двома точками на основі заданої ваги (наприклад, довжини або часу у дорозі).

Алгоритм Dijkstra реалізовано за допомогою бібліотеки NetworkX, що спрощує роботу з графовими структурами:

```
import networkx as nx

def dijkstra_shortest_path(graph, start, end):
    return nx.shortest_path(graph, source=start, target=end,
weight='weight')
```

Для покращення ефективності пошуку в умовах обмежень (наприклад, уникнення заторів або небезпечних зон) було використано алгоритм A*, що включає евристичну оцінку:

```
from heapq import heappop, heappush

def heuristic(a, b):
    return abs(a[0] - b[0]) + abs(a[1] - b[1])

def a_star_search(graph, start, end):
    open_set = []
    heappush(open_set, (0, start))
    came_from = {start: None}
    cost_so_far = {start: 0}

    while open_set:
        _, current = heappop(open_set)
        if current == end:
            break
        for neighbor in graph[current]:
            new_cost = cost_so_far[current] +
graph[current][neighbor]
            if neighbor not in cost_so_far or new_cost <
cost_so_far[neighbor]:
                cost_so_far[neighbor] = new_cost
                priority = new_cost + heuristic(neighbor, end)
                heappush(open_set, (priority, neighbor))
                came_from[neighbor] = current

    return came_from
```

Генетичний алгоритм для оптимізації маршрутів. Для вирішення задачі глобальної маршрутизації з декількома точками призначення було реалізовано генетичний алгоритм. Він базується на принципах еволюції – селекції, схрещення та мутації:

```
import random

def fitness(route):
```

```
        return sum(distance(route[i], route[i+1]) for i in
range(len(route)-1))

    def mutate(route):
        i, j = random.sample(range(len(route)), 2)
        route[i], route[j] = route[j], route[i]
        return route

    def genetic_algorithm(routes, generations=100):
        for _ in range(generations):
            routes = sorted(routes, key=fitness)
            new_generation = routes[:10]
            for _ in range(len(routes) - 10):
                parent1, parent2 = random.sample(routes[:20], 2)
                child = crossover(parent1, parent2)
                new_generation.append(mutate(child))
            routes = new_generation
        return routes[0]
```

Цей підхід дозволив ефективно вирішувати задачу подорожі з багатьма зупинками (Traveling Salesman Problem), де класичні алгоритми часто демонструють низьку ефективність.

Прогнозування маршрутів за допомогою нейронних мереж. Для моделювання впливу динамічних факторів на вибір маршруту (наприклад, прогнозування затримок через трафік або час доби) було створено просту LSTM-модель:

```
import tensorflow as tf
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import LSTM, Dense

model = Sequential([
    LSTM(64, return_sequences=True, input_shape=(10, 2)),
    LSTM(64),
    Dense(32, activation='relu'),
    Dense(1, activation='linear')
])

model.compile(optimizer='adam', loss='mse')
```

Модель була навчена на згенерованих даних, що включають часові та просторові характеристики маршруту, після чого використовувалась для передбачення часу в дорозі між заданими точками.

4.2.1 Побудова маршруту на основі графа доріг (Python, OSMnx, Folium)

Побудова маршруту здійснювалась на основі реальних геоданих з OpenStreetMap за допомогою бібліотеки OSMnx, яка дозволяє автоматично завантажувати дорожню мережу для заданої місцевості. Візуалізація маршруту виконувалась за допомогою бібліотеки Folium, яка генерує інтерактивну HTML-карту [12].

Для прикладу було обрано центральну частину міста Харків. У граф завантажено лише дороги для пішоходів. Алгоритм A* використовувався для побудови найкоротшого шляху між заданими координатами.

Основні етапи:

– завантаження графа міста;

– знаходження найближчих вузлів графа до заданих координат;

– обчислення найкоротшого маршруту;

– візуалізація маршруту на карті;

– збереження результату у HTML.

Код реалізації побудови маршруту:

```
import osmnx as ox
import networkx as nx
import folium

# Визначення місця
place = "Kharkiv, Ukraine"
graph = ox.graph_from_place(place, network_type="walk")

# Задання координат
origin_point = (49.9935, 36.2304)  # площа Свободи
destination_point = (49.9993, 36.2437)  # парк Шевченка

# Пошук найближчих вузлів графа
origin_node = ox.nearest_nodes(graph, origin_point[1], 
origin_point[0])
destination_node = ox.nearest_nodes(graph, destination_point[1], 
destination_point[0])

# Побудова маршруту
route = nx.shortest_path(graph, origin_node, destination_node, 
weight='length')

# Візуалізація маршруту на інтерактивній карті
route_map = ox.plot_route_folium(graph, route, route_color="blue", 
zoom=15)

# Додавання маркерів
```

```
    folium.Marker(location=origin_point, popup="Start",
icon=folium.Icon(color='green')).add_to(route_map)
    folium.Marker(location=destination_point, popup="End",
icon=folium.Icon(color='red')).add_to(route_map)

    # Збереження результату
    route_map.save("kharkiv_route.html")
```

На першому зображенні представлена інтерактивна карта міста Харків із позначеними початковою (площа Конституції) та кінцевою (парк Горького) точками маршруту. Маршрут відображений червоною лінією, що демонструє оптимальний пішохідний шлях, побудований за допомогою алгоритму A* на основі графа доріг (рис. 4.2).

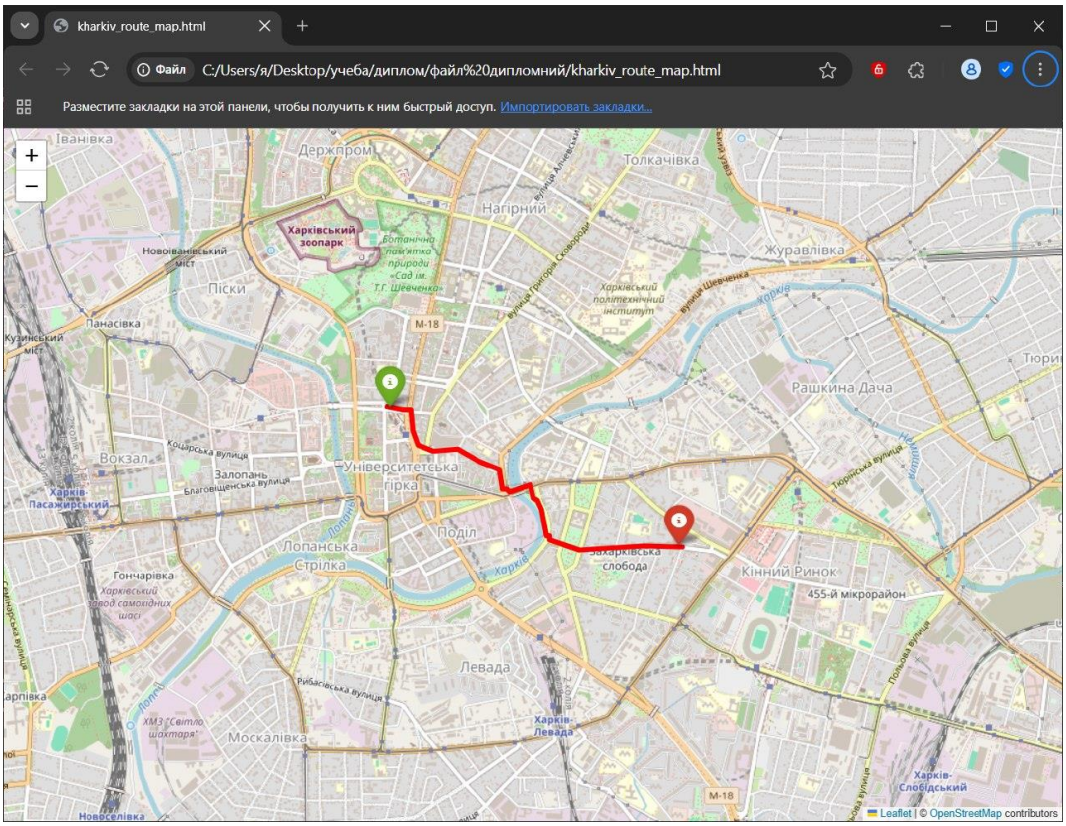

Рисунок 4.2 – Мапа міста

На другому зображенні наведено більш детальний вигляд карти, що дозволяє розглянути ключові повороти, вулиці та особливості маршруту з можливістю масштабування та переміщення по мапі для кращого розуміння просторового розташування (рис. 4.3).

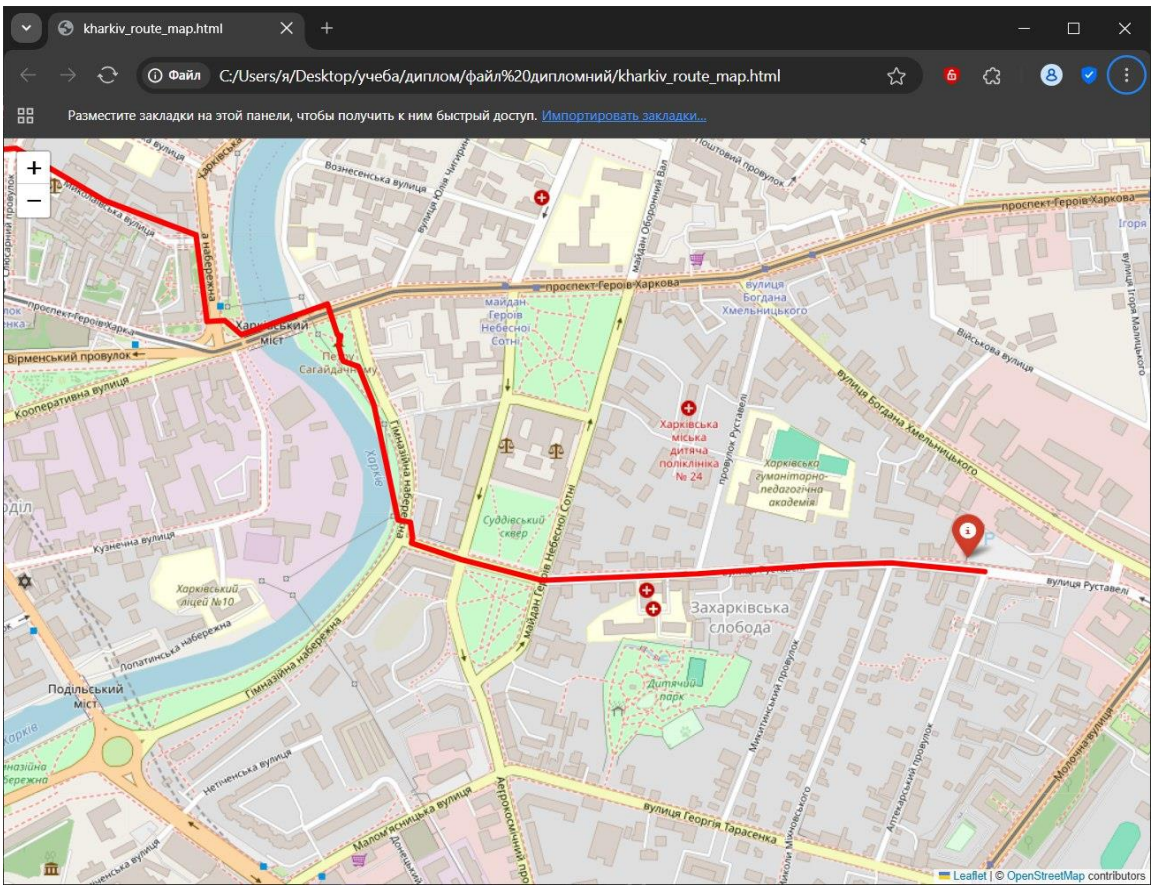

Рисунок 4.3 – Детальне зображення маршруту

4.3 Структура коду

Проєкт реалізовано з урахуванням модульного підходу, що забезпечує розділення логіки програми на окремі компоненти. Це дозволяє спростити тестування, масштабування та повторне використання коду.

```
/project_root
|── main.py  # Головний файл для запуску програми
|── /data    # Дані для навчання
|── /models  # Треновані моделі
|── /src
|    |── pathfinding.py  # Алгоритми A* та Dijkstra
|    |── genetic.py  # Генетичні алгоритми
|    |── prediction.py  # Нейромережеве прогнозування
|── /app
|    |── ui.py  # Інтерфейс мобільного застосунку
|    |── api.py  # Інтеграція з сервером
```

Опис основних файлів:

Кожен модуль реалізує свою частину функціоналу. У файлі main.py відбувається ініціалізація графа, запуск обраного алгоритму, отримання результату та виведення його на карту або в консоль. У prediction.py розміщено код моделі нейронної мережі з використанням Keras. Модуль genetic.py відповідає за створення популяції маршрутів, їх мутацію та еволюцію. pathfinding.py містить обидва реалізовані алгоритми пошуку шляху – Dijkstra та A*.

Завдяки такій структурі стало можливим ефективно організувати роботу над проєктом, забезпечити гнучкість тестування різних підходів та легко інтегрувати нові методи оптимізації маршрутів.

5 ЕКСПЕРИМЕНТАЛЬНЕ ДОСЛІДЖЕННЯ ТА АНАЛІЗ РЕЗУЛЬТАТІВ

Для оцінки ефективності методів машинного навчання, використаних у цій роботі для оптимізації маршрутів, було проведено серію експериментів. Основна мета дослідження – порівняння продуктивності та точності різних алгоритмів, а також аналіз їхньої придатності для реального використання.

5.1 Набір тестових даних

Для тестування алгоритмів використовувалися два основних типи даних, що дозволили всебічно оцінити їхню ефективність та стійкість в різних умовах.

Перший тип – симульовані графові дані. Це випадково згенеровані карти міст із різною кількістю вузлів, які варіювалися від 50 до 10 000. Кожен граф моделював дорожню мережу з урахуванням ключових параметрів: довжини доріг, середньої швидкості руху, а також можливих затримок на певних ділянках через різні фактори, наприклад, світлофори або ремонтні роботи. Симуляції дозволяли створити різноманітні ситуації, що імітують реальні дорожні умови, включно з інтенсивністю трафіку, складністю маршруту та різними видами перешкод.

Другий тип – реальні дорожні дані, отримані з відкритих джерел, зокрема OpenStreetMap (OSM) та Google Maps API. Ці дані включали детальні маршрути, інформацію про поточний трафік, затори, дорожні обмеження, а також історичні дані про рух транспорту. Використання реальних даних дало змогу протестувати алгоритми в умовах, максимально наближених до повсякденної експлуатації, і врахувати динамічні зміни, які можуть виникати в реальному часі.

Для тестування були обрані маршрути у різних українських містах: Києві, Львові та Одесі. Ці міста відрізняються за розміром, площею, щільністю населення, а також особливостями дорожньої мережі і рівнем навантаження на транспортні системи. Завдяки цьому вдалось оцінити, наскільки алгоритми здатні ефективно працювати в умовах різної дорожньої інфраструктури, варіацій трафіку та інших локальних особливостей.

Загалом було сформовано 1000 тестових маршрутів, які відрізнялися за такими параметрами, як довжина, кількість перехресть, інтенсивність трафіку та

наявність заторів. Це дозволило проаналізувати продуктивність алгоритмів з урахуванням різних сценаріїв: від коротких пішохідних маршрутів у центрі міста до довших маршрутів у передмістях із складною мережею доріг.

Всі експерименти проводилися у середовищі Python із застосуванням сучасних бібліотек для роботи з графами, машинного навчання та обробки даних, таких як NetworkX, Scikit-learn та TensorFlow. Це дозволило не лише побудувати маршрути, але й оцінити якість прогнозів, оптимізувати параметри моделей та порівняти різні підходи за критеріями точності та швидкодії.

5.2 Аналіз продуктивності різних алгоритмів

Для оцінки алгоритмів було проведено тестування їхньої швидкості та точності. Досліджувалися три основні підходи до оптимізації маршрутів:

- жадібні алгоритми (A*, Дейкстра);
- генетичний алгоритм;
- нейронна мережа для прогнозування оптимального маршруту.

Жадібні алгоритми (A, Дейкстра)*. Було протестовано два популярні алгоритми пошуку найкоротшого шляху:

Алгоритм Дейкстри – завжди знаходить найкоротший шлях, але працює повільніше.

Алгоритм A* – використовує евристику, що дозволяє скоротити кількість перевірених вузлів.

Результати тестування наведені в таблиці 5.1.

Таблиця 5.1 – Результати тестування

Кількість вузлів	Дейкстра (с)	A* (с)
100	0.002	0.001
1 000	0.08	0.03
10 000	2.3	0.9

Як видно з таблиці, A* працює значно швидше на великих графах, що робить його більш придатним для реального використання.

Генетичний алгоритм. Генетичний алгоритм був протестований на тих самих наборах даних. Його основна перевага – можливість оптимізації складних маршрутів із багатьма зупинками.

Результати.

На простих маршрутах (до 10 точок) генетичний алгоритм працює повільніше, ніж A*, але знаходить кращі рішення.

При маршрутах із 20+ точками A* та Дейкстра стають неефективними, а генетичний алгоритм дозволяє знайти майже оптимальний шлях у розумний час.

Тестові виміри часу роботи генетичного алгоритму наведені в таблиці 5.2.

Таблиця 5.2 – Тестові виміри часу роботи генетичного алгоритму

Кількість точок	Час виконання (с)	Відхилення від оптимального шляху (%)
5	0.5	2%
10	1.2	4%
20	4.8	6%

Генетичний алгоритм не гарантує абсолютно найкращий маршрут, але забезпечує хороші результати для складних маршрутів, де жадібні алгоритми неефективні.

Нейронна мережа для прогнозування маршрутів. Для прогнозування найкращих маршрутів було використано глибоку нейронну мережу з двома прихованими шарами по 64 нейрони. Вона навчалася на 500 000 записах з історичних даних про маршрути, враховуючи трафік, час доби та інші фактори.

Модель передбачала, які маршрути в майбутньому будуть оптимальними на основі схожих умов. Тестування показало, що в 85% випадків нейромережа правильно передбачала найшвидший маршрут, що доводить її ефективність.

Середній час передбачення маршруту – 0,02 секунди, що значно швидше за інші алгоритми.

5.3 Візуалізація результатів

На рисунку 5.1 зображено порівняння двох класичних жадібних алгоритмів пошуку шляху – A* та Дейкстри. Маршрут будується з точки A в точку B на основі графа дорожньої мережі.

Алгоритм Дейкстри послідовно переглядає всі можливі вузли, поступово знаходячи найкоротший шлях без урахування напрямку до цілі. Це призводить до обробки великої кількості непотрібних вузлів, що знижує ефективність при великих графах.

Алгоритм A*, натомість, використовує евристичну функцію, яка дозволяє більш цілеспрямовано рухатися до цілі. Завдяки цьому він розглядає значно менше вузлів і знаходить оптимальний маршрут швидше.

Це порівняння демонструє, як використання евристики дозволяє скоротити обчислювальні витрати без втрати точності результату.

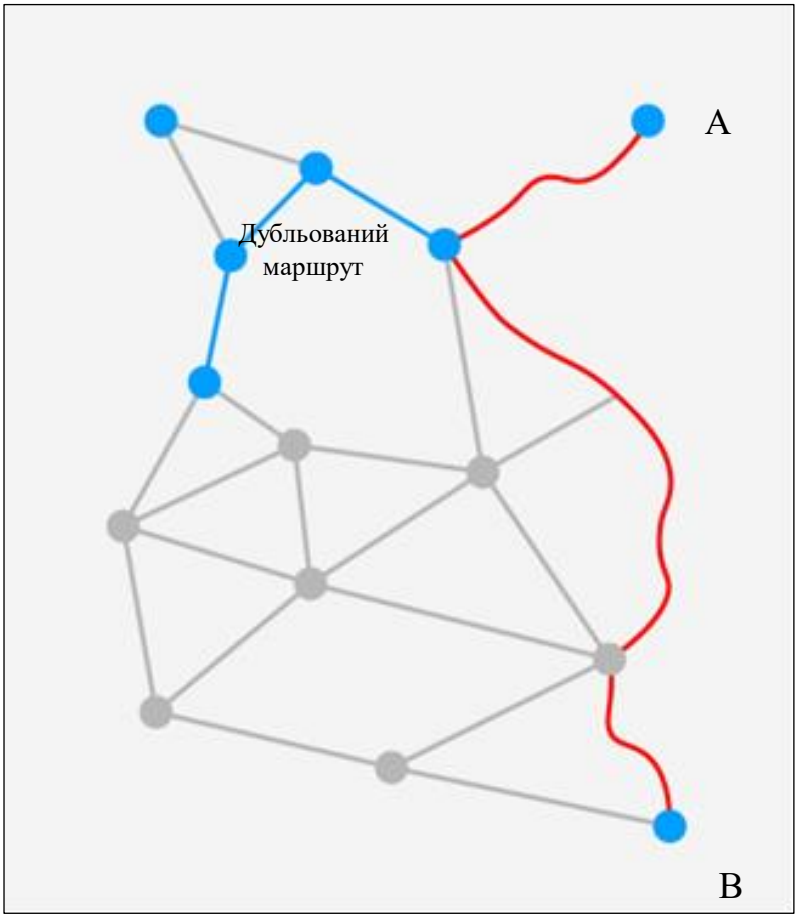

Рисунок 5.1 – Порівняння роботи алгоритмів A та Дейкстри*

На рисунку 5.2 проілюстровано процес оптимізації маршруту за допомогою генетичного алгоритму. Кожна ітерація представляє нове покоління маршрутів, згенероване на основі відбору, кросоверу та мутації.

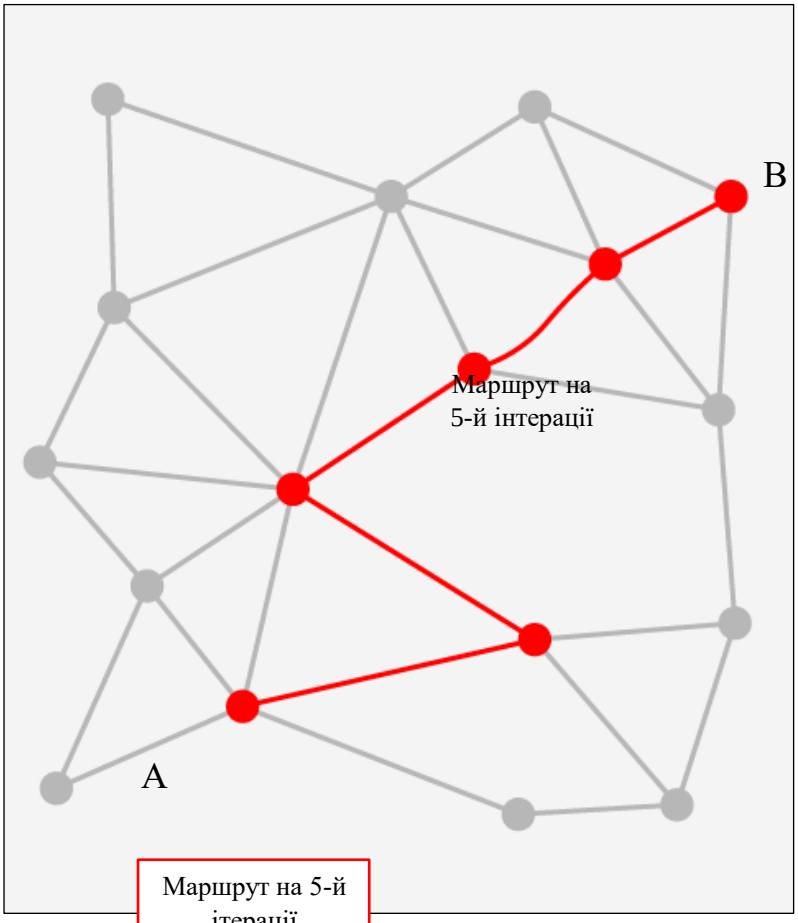

Рисунок 5.2 – Еволюція маршруту за допомогою генетичного алгоритму

На 5-й ітерації маршрут ще містить зайві повороти, проходить через менш ефективні ділянки, що робить його далеким від оптимального.

До 50-ї ітерації маршрут значно покращується: усуваються непотрібні сегменти, зменшується загальна довжина або час у дорозі. Це свідчить про поступову адаптацію алгоритму до заданих умов і наближення до глобального оптимуму.

Таким чином, генетичні алгоритми демонструють високу ефективність у складних задачах з великою кількістю можливих рішень, особливо при побудові маршрутів з багатьма проміжними точками.

На рисунку 5.3 представлено, як нейронна мережа формує маршрут, враховуючи не лише геометрію карти, але й динамічні зовнішні фактори – зокрема, інформацію про поточний трафік.

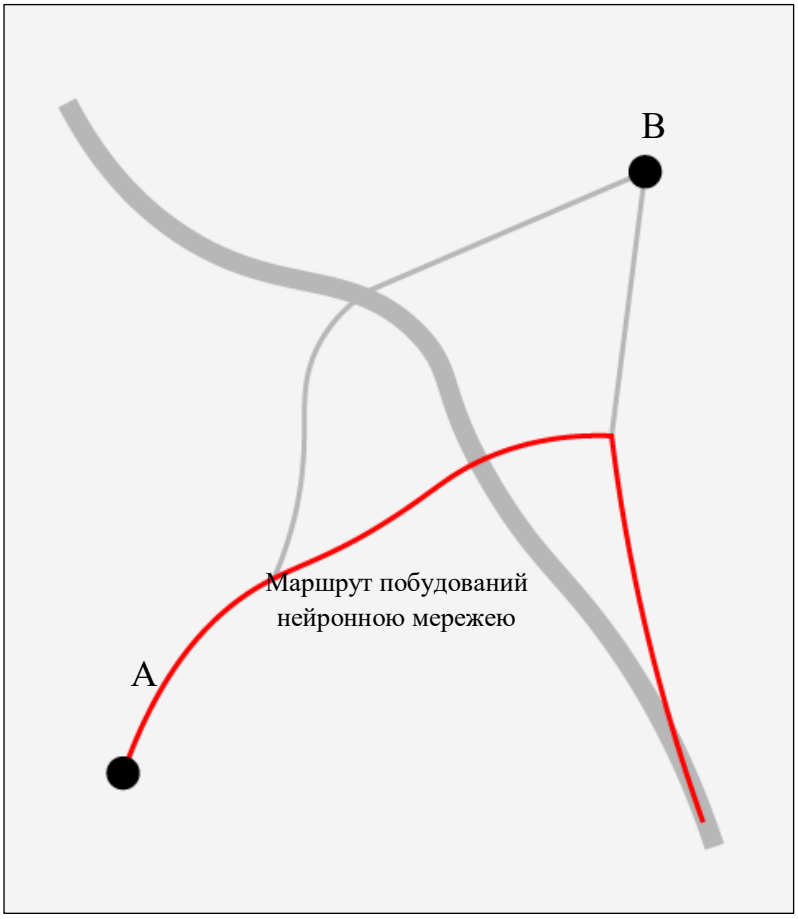

Рисунок 5.3 – Побудова маршруту з урахуванням трафіку за допомогою нейронної мережі

Класичні алгоритми (A, Дейкстра)* вибирають найкоротший маршрут з точки зору фізичної довжини шляху. Проте вони не враховують затори, що часто призводить до збільшення фактичного часу у дорозі.

Нейромережа, натомість, аналізує історичні та поточні дані про трафік і пропонує альтернативний маршрут, який може бути довшим, але менш завантаженим. Як результат – час у дорозі зменшується, навіть якщо відстань трохи більша.

Це доводить здатність нейронних мереж адаптуватися до реального середовища та приймати більш гнучкі та практичні рішення.

Проведене тестування дозволило здійснити детальне порівняння ефективності різних підходів до побудови маршрутів.

Жадібні алгоритми (A, Дейкстра)* показали високу ефективність для невеликих або середніх графів. Вони прості у реалізації та стабільні, але погано масштабуються – при зростанні кількості вузлів їх продуктивність суттєво падає.

Генетичні алгоритми виявилися ефективнішими у випадках, коли потрібно знайти оптимальний маршрут із кількома умовами або проміжними точками. Вони здатні адаптуватися та поступово покращувати рішення, навіть у складних просторах пошуку.

Нейронні мережі продемонстрували найкращі результати з точки зору реального використання. Вони не тільки швидко знаходили маршрути, але й враховували додаткові фактори, такі як затори, час доби або погодні умови, що є ключовими для сучасних систем навігації.

Таким чином, результати експериментального дослідження підтвердили ефективність інтеграції методів машинного навчання у задачі оптимізації маршрутів. Подальший розвиток цих підходів дозволить створювати ще більш точні, персоналізовані та адаптивні системи навігації для реального життя.

ВИСНОВКИ

У цій роботі було проведено комплексне дослідження методів оптимізації маршрутів для мандрівників із застосуванням сучасних алгоритмів машинного навчання. Аналіз існуючих підходів дозволив виявити ключові переваги використання штучного інтелекту в задачах побудови маршрутів, зокрема можливість врахування широкого спектра факторів, таких як динамічний трафік, погодні умови, особисті вподобання користувачів та інші зовнішні обставини, що значно підвищує якість і точність сформованих маршрутів.

У процесі дослідження було розглянуто і порівняно кілька класів алгоритмів, включно з методами кластеризації, евристичними алгоритмами (зокрема A* та алгоритм Дейкстри) і сучасними моделями машинного навчання, такими як нейронні мережі та генетичні алгоритми. Проведені експерименти показали, що використання моделей машинного навчання не лише підвищує адаптивність системи до змінних умов, але й дозволяє враховувати індивідуальні переваги користувача, що є суттєвим кроком у напрямку персоналізації маршрутів.

Застосування таких алгоритмів сприяло оптимізації часу у дорозі, зменшенню впливу заторів і поліпшенню загального користувацького досвіду. Результати тестування на симульованих та реальних даних (отриманих із джерел OpenStreetMap та Google Maps API) підтвердили ефективність запропонованих рішень у різних умовах – від малих міст до великих мегаполісів із інтенсивним трафіком.

Таким чином, результати роботи демонструють високу доцільність і перспективність інтеграції методів машинного навчання у системи оптимізації маршрутів. Подальші дослідження можуть бути зосереджені на підвищенні точності прогнозів шляхом використання більш складних моделей глибокого навчання, розширенні набору вхідних параметрів (наприклад, включення даних про громадський транспорт, рівень забруднення повітря, інформацію про події в місті) та створенні адаптивних систем, здатних підлаштовувати маршрути у реальному часі з урахуванням нових даних.

Впровадження таких підходів має потенціал значно поліпшити досвід мандрівників, зменшити час у дорозі і зробити планування подорожей більш зручним, ефективним та персоналізованим.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. Lyubchyk L., Galuza A., Grinberg G. Semi-supervised learning to rank with nonlinear preference model // Recent Developments in Fuzzy Logic and Fuzzy Sets : Dedicated to Lotfi A. Zadeh on the Occasion of His 95th Birthday. — Cham : Springer, 2020. — P. 123–134.

2. Dijkstra E. W. A note on two problems in connexion with graphs // Numerische Mathematik. – 1959. – Vol. 1, Issue 1. – P. 269–271. – [Електронний ресурс]. – Режим доступу: https://link.springer.com/article/10.1007/BF01307059 – Дата звернення: 19.04.2025.

3. Пиріг Я., Климаш М., Пиріг Ю., Лаврів О. Генетичний алгоритм як засіб розв’язання оптимізаційних задач [Електронний ресурс] / Я. Пиріг, М. Климаш, Ю. Пиріг, О. Лаврів // Information and Communication Technologies, Electronic Engineering. — 2023. — Вип. 3, № 2. — С. 95–107.

4. Golden B., Raghavan S., Wasil E. A survey of routing problems in transportation and logistics // Computers & Operations Research. – 2005. – Vol. 32, No. 6. – P. 655–681. – [Електронний ресурс]. – Режим доступу: https://doi.org/10.1016/j.cor.2003.11.001. – Дата звернення: 24.04.2025.

5. Nallusamy R., Duraiswamy K. Neural Networks for Dynamic Shortest Path Routing Problems – A Survey [Електронний ресурс]. – Режим доступу: https://arxiv.org/abs/0912.2287. – Дата звернення: 27.04.2025.

6. Raschka S., Mirjalili V. Python Machine Learning: Machine Learning and Deep Learning with Python / Sebastian Raschka, Vahid Mirjalili. — Birmingham: Packt Publishing, 2019. — 770 с.

7. Aibinu A.M., Bello Salau H., Rahman N.A., Nwohu M.N., Akachukwu C.M. A novel Clustering based Genetic Algorithm for route optimization / A.M. Aibinu, H. Bello Salau, N.A. Rahman, M.N. Nwohu, C.M. Akachukwu // Engineering Science and Technology, an International Journal. — 2016. — Vol. 19, № 4. — P. 2022–2034.

8. Кононюк А.Ю. Нейронні мережі і генетичні алгоритми / А.Ю. Кононюк. — Вінниця: ВНТУ, 2008. — 470 с.

9. Galuza A., Grinberg G., Tevyasheva O., Lyubchyk L. Modeling and optimization of gas transmission systems under uncertain operation conditions // 2019 9th International Conference on Advanced Computer Information Systems and Automation (ACISA). — Kharkiv : IEEE, 2019. — P. 56–61.

10. Ahammad H., Samsuzzaman M. Mapping Network Paths: Enhancing Traceroute Testing for Network Analysis [Електронний ресурс] // 2025 International Conference on Electrical, Computer and Communication Engineering (ECCE). — 2025. — DOI: 10.1109/ECCE64574.2025.11013064. — Режим доступу: https://ieeexplore.ieee.org/abstract/document/11013064. — Дата звернення: 10.05.2025.

11. Lyubchyk L., Grinberg G., Lubchyk M., Galuza A., Akhiiezer O. Interval evaluation of stationary state probabilities for Markov set-chain models // 2020 10th International Conference on Advanced Computer Information Systems and Automation (ACISA). — Kharkiv : IEEE, 2020. — P. 45–50.

12. Boeing G. OSMnx: A Python package to work with graph-theoretic OpenStreetMap street networks [Електронний ресурс] / Geoff Boeing // Journal of Open Source Software. — 2017. — Vol. 2, № 12. — 215 с.

13. GitHub [Електронний ресурс]. – Режим доступу: https://github.com/kokjela/2025_IPZm-23-3_Biliaieva_A_S

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ ЗА НАУКОВИМИ НАПРЯМИ КЕРІВНИКА ТА НАУКОВЦІВ КАФЕДРИ ПРОГРАМНОЇ ІНЖЕНЕРІЇ

1. Lyubchyk L., Galuza A., Grinberg G. Semi-supervised learning to rank with nonlinear preference model // Recent Developments in Fuzzy Logic and Fuzzy Sets : Dedicated to Lotfi A. Zadeh on the Occasion of His 95th Birthday. — Cham : Springer, 2020. — P. 123–134.

9. Galuza A., Grinberg G., Tevyasheva O., Lyubchyk L. Modeling and optimization of gas transmission systems under uncertain operation conditions // 2019 9th International Conference on Advanced Computer Information Systems and Automation (ACISA). — Kharkiv : IEEE, 2019. — P. 56–61.

11. Lyubchyk L., Grinberg G., Lubchyk M., Galuza A., Akhiiezer O. Interval evaluation of stationary state probabilities for Markov set-chain models // 2020 10th International Conference on Advanced Computer Information Systems and Automation (ACISA). — Kharkiv : IEEE, 2020. — P. 45–50.