Міністерство освіти і науки України
Харківський національний університет радіоелектроніки

Факультет Комп'ютерних наук
(повна назва)
Кафедра Штучного інтелекту
(повна назва)

КВАЛІФІКАЦІЙНА РОБОТА
Пояснювальна записка

рівень вищої освіти другий (магістерський)

Дослідження еволюційних методів для персоналізованого календарного планування на основі аналізу повідомлень про зустрічі в месенджері
(тема)

Виконав:
студент 2 курсу, групи СШІм-22-3
Бурцева А.Д.
(прізвище, ініціали)

Спеціальність 122 Комп'ютерні науки
(код і повна назва спеціальності)
Тип програми освітньо-наукова
(освітньо-професійна або освітньо-наукова)
Освітня програма Системи штучного інтелекту
(повна назва спеціалізації)
Керівник проф. Чалий С.Ф.
(посада, прізвище, ініціали)

Допускається до захисту

Зав. кафедри _____ В.О. Філатов
(підпис) (прізвище, ініціали)

2024 р.

Харківський національний університет радіоелектроніки

Факультет Комп'ютерних наук
(повна назва)

Кафедра Штучного інтелекту
(повна назва)

Рівень вищої освіти другий (магістерський)

Спеціальність 122 Комп'ютерні науки
(код і повна назва)

Тип програми освітньо-наукова
(освітньо-професійна або освітньо-наукова)

Освітня програма Системи штучного інтелекту
(повна назва)

ЗАТВЕРДЖУЮ:
Зав. кафедри _____
(підпис)
«_____« _____ 20 ___ р.

ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ РОБОТУ

студентові Бурцевій Анастасії Денисівні
(прізвище, ім'я, по батькові)

1. Тема роботи Дослідження еволюційних методів для персоналізованого календарного планування на основі аналізу повідомлень про зустрічі в месенджері

затверджена наказом університету від 1 квітня 20 24 р. № 260Ст

2. Термін подання студентом роботи до екзаменаційної комісії 12 червня 20 24 р.

3. Вихідні дані до роботи Документація до мови програмування Python, пакетів Deep, Numpy, Pandas, власноруч створена вибірка текстів для вибору типу зустрічей, документація та наукові матеріали за темою генетичних алгоритмів

4. Перелік питань, що потрібно опрацювати в роботі

1) Аналіз предметної галузі та постановка задачі

2) Дослідження існуючих методів та рішень у сфері використання генетичного алгоритму для формування розкладів

3) Теоретичне дослідження застосування генетичного алгоритму для побудови персонального розкладу

4) Реалізація алгоритму на мові програмування Python, а також реалізація попереднього етапу обробки даних з месенджера

КАЛЕНДАРНИЙ ПЛАН

№	Назва етапів роботи	Терміни виконання етапів роботи	Примітка
1	Отримання завдання на кваліфікаційну роботу	01.04.2024	виконано
2	Аналіз методів щодо вирішення поставленої	02.04.2024–08.04.2024	виконано
	задачі		
3	Постановка задачі	08.04.2024–10.04.2024	виконано
4	Експериментальне моделювання та навчання	10.04.2024–18.04.2024	виконано
5	Розробка моделей	19.04.2024–25.04.2024	виконано
6	Написання пояснювальної записки	26.04.2024–19.05.2024	виконано
7	Попередній захист	24.05.2024	виконано
8	Захист перед ЕК	12.06.2024	виконано

Дата видачі завдання 1 квітня 20 24 р.

Студент _____
(підпис)

Керівник роботи _____ проф. Чалий С.Ф.
(підпис) (посада, прізвище, ініціали)

РЕФЕРАТ

Пояснювальна записка: 77 с., 14 рис., 10 табл., 2 дод., 28 джерел.

АНАЛІЗ ТЕКСТУ, ЕВОЛЮЦІЙНІ МЕТОДИ, ГЕНЕТИЧНІ АЛГОРИТМИ, КЛАСИФІКАЦІЯ, МАШИННЕ НАВЧАННЯ, ОБРОБКА ПОВІДОМЛЕНЬ, ПЕРСОНАЛІЗОВАНЕ КАЛЕНДАРНЕ ПЛАНУВАННЯ, ФУНКЦІЯ ПРОСТОСОВАНОСТІ.

Об'єкт дослідження – процес персоналізованого календарного планування, що ґрунтується на аналізі повідомлень про зустрічі в чаті.

Предмет дослідження – еволюційні методи (генетичні алгоритми) персоналізованого календарного планування.

Мета роботи – розробки та реалізації еволюційного алгоритму для персоналізованого календарного планування з використанням аналізу повідомлень про зустрічі в чаті з урахуванням індивідуальних потреб користувача.

Методи дослідження: аналіз існуючих теоретичних матеріалів і вивчення існуючих методів оптимізації планування зустрічей за допомогою еволюційних алгоритмів, розробка моделі генетичногоалгоритму для задачі планування, яка враховує індивідуальні вимоги та обмеження користувача, обробка та аналіз отриманих результатів з метою оцінки ефективності розробленого методу.

У результаті виконання роботи буде розроблена модель генетичного алгоритму для персоналізованого календарного планування з урахуванням інформації з повідомлень про зустрічі в чаті та з можливістю врахування параметрів та потреб користувача.

ABSTRACT

Master’s thesis contains: 77 pp., 14 fig., 10 tabl., 2 ann., 28 references.

CLASSIFICATION, EVOLUTIONARY METHODS, GENETIC ALGORITHMS, MACHINE LEARNING, MESSAGE PROCESSING, PERSONALIZED CALENDAR SCHEDULING, TEXT ANALYSIS, USABILITY FUNCTION.

The object of the study is the process of personalized calendar planning based on the analysis of messages about meetings in the chat.

The subject of research is the use and research of evolutionary methods (genetic algorithms) to optimize the process of personalized calendar planning, taking into account individual requirements and limitations set by the user.

The purpose of the work – development and research of the effectiveness of the application of evolutionary algorithms for personalized calendar planning using the analysis of messages about meetings in the chat. The work involves the creation of a system for the meeting planning process by improving the genetic algorithm.

Research methods: analysis of theoretical materials and existing meeting planning optimization methods using evolutionary algorithms, development of a genetic model considering user requirements and limitations, experimental tests to measure the genetic algorithm's effectiveness on real or simulated data, and processing and analysis of results to evaluate the method's effectiveness and model accuracy.

A genetic algorithm model will be developed for personalized calendar planning, incorporating meeting information from chat messages and user needs. Additionally, an algorithm for processing data in messages using text processing methods and a classifier will be implemented.

ЗМІСТ

Перелік умовних позначень, символів, одиниць, скорочень і термінів 8

Вступ 9

1 Аналіз предметної області та постановка задачі 11

1.1 Аналіз процесів персонального календарного планування 11

1.2 Дослідження підходів до обробки текстової інфоомації щодо календарного планування 13

1.3 Дослідження еволюційних алгоритмів в задачах календарного планування 16

1.3.1 Проблема підтримки вимог користувача в генетичних алгоритмах 19

1.3.2 Існуючі підходи вирішення проблеми з формуванням функції пристосованості та підтримкою вимог користувача 20

1.3.3 Практики використання генетичних алгоритмів в задачах формування розкладів 23

1.4 Постановка задачі дослідження 24

2 Теоретичне дослідження обраної проблеми 26

2.1 Удосконалення функції пристосованості 26

2.2 Розробка методу персонального календарного планування на основі аналізу повідомлень з використанням генетичного алгоритму 29

2.3 Формування вхідного набору даних для персоналізованого планування 33

2.4 Класифікація типів зустрічей 37

2.5 Налагодження функції пристосованості та імплементація генетичного алгоритму 38

3 Програмна реалізація та експериментальна перевірка 45

3.1 Обґрунтування вибору інструментарію для реалізації розробленого методу 45
3.2 Підготовка вхідних даних для персоналізованого календарного планування 45
3.3 Реалізація класифікатора для типів зустрічей 48
3.4 Персональне календарне планування з використанням генетичного алгоритму 52
3.5 Результати експериментальної перевірки 56
3.5.1 Обговорення ефективності генетичного алгоритму 59
3.5.2 Порівняння результатів з існуючими рішеннями 61
3.6 Аналіз перспектив та варіанти застосування розробленого методу .. 62
Висновки 64
Перелік джерел посилання 66
Додаток А Лістинг коду алгоритм 69
Додаток Б Відомість кваліфікаційної роботи 77

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СИМВОЛІВ, ОДИНИЦЬ, СКОРОЧЕНЬ І ТЕРМІНІВ

ГА – генетичні алгоритм;

ЕА – еволюційні алгоритми;

МН – машинне навчання;

НАН – національна академія наук;

ПКП – персональне календарне планування;

ТД – так далі;

ТП – тому подібне;

AI – Artificial Intelligence – штучний інтелект;

API – Application Programming Interface – інтерфейс програмування додатків;

CSV – Comma-Separated Values – значення, розділені комами;

IGA – Interactive Genetic Algorithms – інтерактивні генетичні алгоритми;

MOGA – Multi-Objective Genetic Algorithms – багатоцільові генетичні алгоритми;

NER – Named Entity Recognition – розпізнавання іменованих сутностей;

NLP – Natural Language Processing – обробка природної мови;

NLTK – Natural Language Toolkit – інструментарій для обробки природної мови;

PERT – Program Evaluation And Review Technique – техніка оцінки та перегляду програм;

SVM – Support Vector Machine – метод опорних векторів.

ВСТУП

Сучасний ритм життя характеризується інтенсивною динамікою, що пов’язано з вирішенням практичних задач з використанням сучасних інформаційних технологій. Вважається, що сучасна людина виконує протягом дня майже вдвічі більше завдань, ніж це було 200 років тому. Відповідно побудова індивідуальних планів є важливою задачею.

Значна частина життя сучасної людини відбувається в цифровому просторі, а багато зустрічей, включаючи робочі, плануються через переписку в месенджерах, тому підхід до планування, що використовує повідомлення з чатів, стає актуальним напрямком дослідження. Однак існуючі підходи до персонального календарного планування зазвичай використовують не текстові, а табличні вхідні данні і не орієнтовані на планування з урахуванням комбінації ваг показників ,пріоритетів зустрічей а також обмежень на кількість та час зустрічей, що свідчить про важливість розробки комбінованого підходу. Таких підхід має поєднувати формування набору вхідних даних із текстів чату та побудови індивідуальних планів з урахуванням визначених пріоритетів та обмежень.

Останніми роками спостерігається тенденція до зростання застосування методів штучного інтелекту в різних сферах життєдіяльності. У контексті персоналізованого календарного планування особливої уваги заслуговують еволюційні методи, зокрема, генетичні алгоритми. На даний момент маємо багато досліджень у сфері використання генетичних алгоритмів як інструмента оптимізації. Використовуючи еволюційні алгоритми, ми використовуємо комп'ютерні моделі здатні адаптуватися до змін у своєму оточенні і покращувати свою роботу в результаті навчання.

Застосування генетичних алгоритмів для оптимізації розкладу на основі переписок в месенджері – оскільки це задача з великою кількістю обмежень і потенційних рішень, генетичний алгоритм може допомогти знайти баланс між різними факторами. Важливо, що такий підхід дасть

змогу підтримати персональні вимоги користувача, тобто бути адаптивним і мати змогу підтримати будь-які зміни у побажаннях людини. Саме реалізація такої динамічності буде досліджуватись у рамках цієї роботи, де для врахуванні індивідуальних потреб користувача, важливо сформувати функції пристосованості, що враховує індивідуальних критеріїв, заданих кінцевим користувачем.

Щодо інструментів для реалізації дослідження, в даному проекті планується використовувати мову програмування Python. Для аналізу текстових даних, зокрема повідомлень про зустрічі в чаті, будуть використовуватися бібліотеки для обробки природної мови, такі як NLTK і Spacy. Застосування генетичних алгоритмів буде реалізовано за допомогою бібліотеки DEAP.

1 АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ ТА ПОСТАНОВКА ЗАДАЧІ

1.1 Аналіз процесів персонального календарного планування

Процес персонального календарного планування – це невід'ємна частина життя сучасної людини, підприємства, державної структури чи бізнесу. Розглядаючи критичні елементи цього процесу можна виділити визначення цілей і завдань, пріоритизацію, розподіл часу, перевірку та корекцію, а також можливість удосконалення. Визначення цілей і завдань означає чітке розуміння того, що для нас є важливим у контексті планування. Наші цілі і завдання повинні бути конкретними та чітко розподіленими. Крім того, важливим етапом є вміння розподіляти свій час. Розподіл часу це більше про управління нашими ресурсами так, аби ми могли досягнути максимальної ефективності. Здалеку не завжди ми розуміємо, скільки часу необхідно витратити на кожне завдання, але з досвідом це стає легше. Перевірка та корекція важливі для того, щоб ми могли постійно оцінювати наше планування та вносити корективи, коли це необхідно. Зрештою, удосконалення після певного періоду дозволяє нам вивчити власний досвід, шукати те, що працювало добре, що не працювало і що потребує змін. Загальний процес персонального календарного планування можна побачити на рисунку 1.1.

Для календарного планування існують різні технічні підходи та системи. Вони використовуються в різних галузях, мають різний рівень складності та варіанти реалізації. Їх можна віднести до кількох основних груп, представлених у таблиці 1.1.

Таблиця 1.1 – Підходи до календарного планування

Назва підходу	Опис
Традиційне Планування	Традиційні системи планування, такі як Google Calendar або Microsoft Outlook.

Продовження таблиці 1.1

Автоматизоване Планування	Використання AI та машинного навчання.
Проектне планування	Використання PERT (Program Evaluation and Review Technique).
Онтологічне планування	Використання онтологій.
Квантове планування	Цей найновіший підхід заснований на використанні квантових комп'ютерів для вирішення надскладних задач планування.

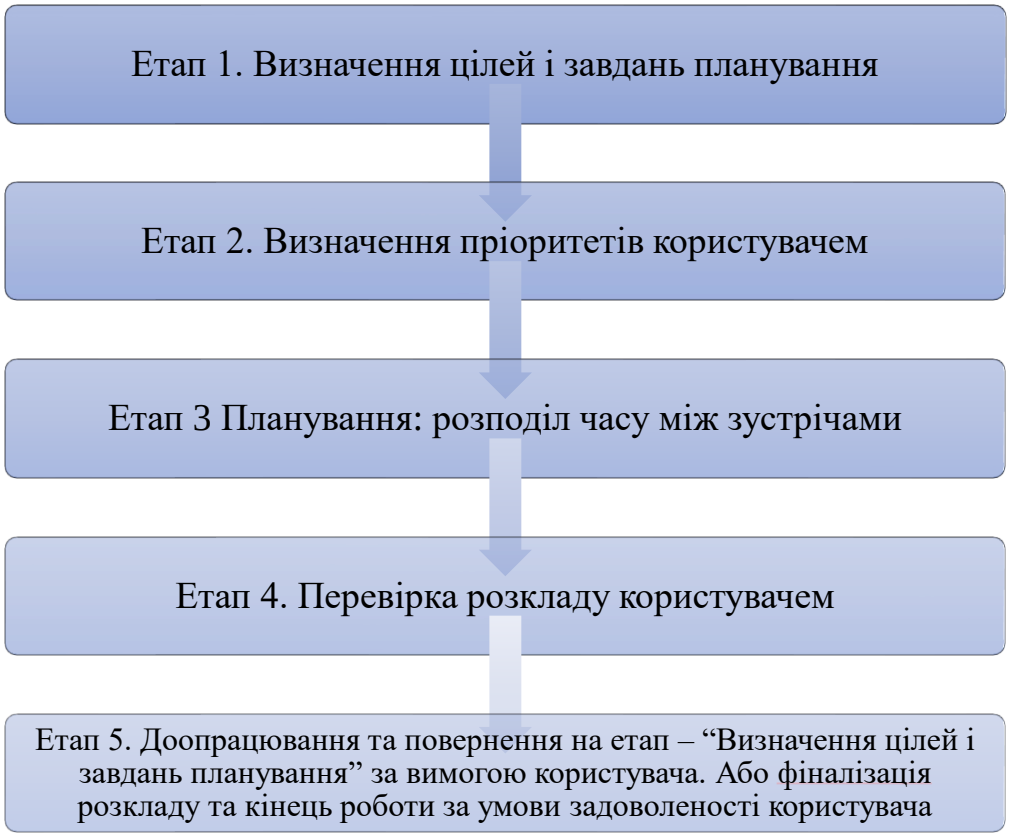

Рисунок 1.1 – Процес персоналізованого календарного планування

Існуюче дослідження можна віднести до типу автоматизованого планування. Так як в роботі ми працюємо з еволюційними алгоритмами для оптимізації календарного плануванням на основі переписок в месенджері, які зазвичай надаються в текстовій формі, що потребує розгляду підходів до обробки природньої мови.

1.2 Дослідження підходів до обробки текстової інформації щодо календарного планування

Процес персонального календарного планування на основі переписок в месенджері потребує обробки повідомлень та підготовки цих даних для використання в алгоритмі. Одним з етапів роботи є створення методу, який буде працювати з аналізом тексту інструментами NLP.

Спершу дослідимо дисципліну, відому як обробка природних мов або Natural Languages Processing (NLP) англійською. Дана направленість належить до галузі комп'ютерних наук, а в конкретному випадку, до сфери штучного інтелекту. Основною ціллю обробки природних мов є створення алгоритмів та систем, що надають комп'ютерам змогу аналізувати і інтерпретувати людську мову у письмовій та усній формі на рівні, що приблизно еквівалентний людському сприйняттю. Порівняльний аналіз підходів до обробки природньої мови наведено у таблиці 1.2.

Таблиця 1.2 – Аналіз підходів до обробки природньої мови

Назва підходу	Опис	Переваги	Недоліки
Аналіз тексту	Процес вивчення структури тексту, що включає морфологічний, синтаксичний, та семантичний аналіз.	Може дати внутрішне розуміння мови, ідентифікувати ключові слова, теми та настрої в текстах.	Може стикнутися з проблемами, коли обробляються мови з багатозначністю або складними граматичними правилами.
Статистичний NLP	Зосереджений на використанні статистичних методів, наприклад, ймовірнісних моделей.	Володіє силою виявлення складних шаблонів в даних за допомогою статистичного аналізу.	Може бути обмежене умовами, що передбачають сталість структури мови, яка в реальності не є постійною.

Продовження таблиці 1.2

Підхід на основі правил	У цьому підході використовуються мовні правила, які формуються людьми-експертами.	Може забезпечити високу точність в окремих сценаріях.	Процес може бути трудомістким і вимагає глибокого розуміння мови, а також дуже залежить від якості встановлених правил.
Глибоке навчання	Використовує штучні нейронні мережі для моделювання та обробки природних мовних даних.	Нейронні мережі, можуть виявляти та вчитися складних шаблонів у великому обсязі мовних даних.	Вимагають значного обсяга даних для ефективного навчання та можуть бути витратними в плані обчислювальних ресурсів.

Ці підходу орієнтовані на структуризацію вхідних текстових даних. Це дає можливість в подальшому їх використовувати для календарного планування. Крім того в рамках роботи було досліджено методи для обробки природньої мови з акцентом на дані з месенджерів.

Переписки в месенджерах часто містять неструктуровані дані у вигляді тексту, що може мати емодзі, стікери, гіфки тощо, що робить процес аналізу більш складним. Крім цього користувачі месенджерів часто використовують жаргон, скорочення, сленг, помилки в написанні і емотикони, що може ускладнити розуміння тексту. Також треба відмітити що обробка переписок в месенджерах повинна бути чутливою до приватності та захисту даних користувачів. Методи для аналізу повідомлень з акцентом на особливості даних та їх структуру наведено у таблиці з порівняльним аналізом підходів до обробки природньої мови в месенджерах (таблиця 1.3).

Таблиця 1.3 – Аналіз підходів до обробки природньої мови в месенджерах

Назва	Опис	Переваги	Недоліки
Використання NLP	Використання NLP для обробки тексту з метою збору інформації.	Може допомогти в отриманні цінної інформації з переписок у месенджерах.	Стиль написання, скорочення, сленг потребують більш глибокого розуміння контексту.
Машинне навчання	Вивчення моделей поведінки користувача, основане на їх повідомленнях в месенджерах.	Є ефективним для виявлення шаблонів і навчання на основі великих даних, що призводить до більш точних прогнозів.	Може вимагати значних обчислювальних ресурсів.
Семантичний аналіз	Цей підхід до обробки природної мови оснований на розумінні значення слів в контексті.	Розуміння внутрішніх відношень між словами в переписці.	Він вимагає досконалого розуміння мови та виразних форм.

Таким чином методи обробки текстів в месенджері мають враховувати додаткову інформацію у вигляді сленгу, емодзі, жаргону. Потрібно відкинути цю додаткову інформацію при підготовці даних для задач календарного планування.

Після підготовки даних для вирішення проблем оптимізації планувалось використати один із варіантів еволюційних алгоритмів, тому було доцільно зробити дослідження та аналіз їх використання в задачах календарного планування.

1.3 Дослідження еволюційних алгоритмів в задачах календарного планування

Оптимізація розкладу є важливим завданням у різних сферах, починаючи від промисловості й закінчуючи освітою або транспортом. Яким би складним не було це завдання, з появою еволюційних обчислень воно отримало нові рішення і підходи. Еволюційні алгоритми, особливо генетичні алгоритми, продемонстрували високу ефективність у вирішенні таких задач. Вони здатні шукати рішення в великих просторах пошуку, де традиційні методи оптимізації часто обмежені, і, що найважливіше, вони можуть адаптуватися до змінних умов та обставин.

На сьогоднішній день в академічних та промислових дослідженнях використовуються різні варіації генетичних алгоритмів, включаючи багатоцільову оптимізацію, паралельні генетичні алгоритми, генетичні алгоритми з рідкими серіями та багато інших.

Генетичні алгоритми активно застосовуються для вирішення проблем календарного планування в різних галузях. Наприклад, в промисловості ми маємо планування заводських графіків і ліній виробництва, де генетичні алгоритми і були успішно застосовані. У 2017 році дослідники з Інституту автоматики і системного проектування НАН України розробили генетичний алгоритм для оптимізації заводського розкладу.

В телекомунікаціях генетичні алгоритми використовуються для оптимізації розміщення та керування антенами в мобільних мережах. В логістиці їх можна зустріти для створення оптимальних маршрутів для поставки товарів. В освіті це проблема створення розкладу для університетів чи шкіл. Якщо говорити загально, генетичні алгоритми входять у групу еволюційних алгоритмів – це тип алгоритмів пошуку і оптимізації, заснований на природних принципах еволюції, таких як наслідування, мутація, вибір та кросовер. Види еволюційних алгоритмів наведені у таблиці 1.4.

Таблиця 1.4 – Види еволюційних алгоритмів

Назва	Опис	Переваги	Недоліки
Генетичні алгоритми	Вони працюють за принципом «виживання найпридатніших», де найкращі «хромосоми» передаються до наступного покоління.	Здатні знаходити глобальні, а не локальні мінімуми.	Мають тенденцію до передчасної збіжності, оскільки вони можуть застрягти в місцевих оптимумах.
Алгоритми генетичного програмування	Цей підхід використовує ті ж основні принципи, але рішеннями тут є комп'ютерні програми. Ці варіанти програм можуть бути поєднані та модифіковані.	Може автоматично вирішити проблему.	Складно визначити функцію пристосування.
Стратегії еволюції	В цьому випадку еволюційний процес зосереджений на оптимізації вектора параметрів, а не на їх пошуку.	Досить прості для розуміння та обчислень. Здатні адаптуватися до змін проблеми в процесі її розв'язання.	Можуть виявитися недостатніми для складних проблем. Дуже залежать від параметрів.
Диференційна еволюція	Це ще один тип еволюційної стратегії, який особливо ефективний для неперервних просторів пошуку.	Ефективна для неперервних просторів пошуку. Стабільна і нечутлива до вибору початкових параметрів.	Може бути неефективною для дискретних просторів пошуку. Схильна до застрягання в місцевих мінімумах.

Однією із ключових характеристик еволюційних алгоритмів є їх здатність розв'язувати складні, недетерміновані задачі. Як вже вище зазначалось в задачах планування, де ми можемо мати велику кількість можливих рішень, еволюційні алгоритми можуть бути ефективним методом знаходження оптимального варіанту. В таблиці 1.5 наведені переваги та недоліки застосування еволюційних алгоритмів в задачах планування.

Таблиця 1.5 – Особливості застосування генетичних алгоритмів для задач планування

Переваги застосування ГА до задач планування	Недоліки застосування ГА до задач планування
Вони можуть знайти рішення в складних просторах пошуку, де часто немає точних алгоритмів для вирішення задачі.	Вони можуть не гарантувати знаходження глобального оптимуму, тільки локального.
Вони можуть працювати з обмеженнями та багатоцільовими задачами.	Вони можуть бути обчислювально важкими, оскільки вимагають великої кількості оцінювальних рішень.
Вони гнучкі і можуть бути адаптовані для різних задач.	Проблеми з врахуванням індивідуальних потреб користувача.

Одним з основних обмежень використання еволюційних алгоритмів є потреба в значних обчислювальних ресурсах. Так як генетичні алгоритми зазвичай потребують великої кількості обчислень. Це блокує їх використання у випадках, коли доступ до обчислювальних ресурсів обмежений або коли реактивність є важливим фактором. Також треба зазначити що параметризація генетичного алгоритму може бути технічно складною і вимагати високої кваліфікованості спеціалістів. Неправильні настройки можуть призвести до прискореної збіжності або, навпаки, до затримки у знаходженні оптимального рішення.

Втілення цих проблем в реальному світі можна зустріти в промисловості, коли обчислювальних ресурсів часто не вистачає. Наприклад, для складних завдань планування виробництва з великою кількістю змінних, вивчення всього простору рішень за допомогою генетичних алгоритмів може вимагати величезних обчислювальних потужностей та часу. Проблема параметризації актуальна для сфер зі складними розкладами (університети, великі бізнеси, багатопрофільні лікарні) при використанні генетичних алгоритмів для розкладів потрібно налаштувати багато параметрів. В сфері фінансів можна натрапити на проблему переобчислення: при оптимізації фінансових портфелів, в яких розглядаються тисячі можливих комбінацій активів, генетичні алгоритми можуть непотрібно витрачати обчислювальний час на повторне розглядання попередньо відхилених комбінацій.

Але в цій роботі було вирішено зробити акцент саме на такий недолік використання генетичних алгоритмів, як можливість підтримки індивідуальних потреб користувача при оптимізації його календарного планування.

1.3.1 Проблема підтримки вимог користувача в генетичних алгоритмах

Генетичні алгоритми застосовуються для вирішення широкого спектра оптимізаційних завдань. Але, що стосується підтримки вимог користувача, є декілька потенційних проблем, з якими ці алгоритми можуть стикнутися.

Визначення функції пристосованості при використанні генетичних алгоритмів полягає в тому, щоб правильно визначити функцію, яка відображає вимоги користувача. Ця проблема є досить складною в реальному світі, де вимоги користувача можуть бути нечіткими, суперечливими або невизначеними. Функція пристосованості є серцевиною

будь-якого генетичного алгоритму. Вона дозволяє визначити, наскільки «добре» кожний індивідуум (рішення) відповідає поставленій цілі. В контексті календарного планування, це могло б виглядати як оптимізація ефективності використання часу користувача, задоволення його потреб або балансу між роботою та вільним часом.

Вимоги користувача рідко бувають статичними або однозначними. Вони можуть змінюватися з часом або в залежності від контексту, можуть бути суперечливими або навіть неявними. Наприклад, користувач може хотіти провести більше часу з сім'єю, але також збільшити продуктивність на роботі. Ці цілі можуть конфліктувати одна з одною. Подібні неоднозначності і динаміка вимог користувача можуть викликати труднощі з визначенням функції пристосованості.

1.3.2 Існуючі підходи вирішення проблеми з формуванням функції пристосованості та підтримкою вимог користувача

Проблема визначення функції пристосованості, яка добре відображає вимоги користувача в генетичних алгоритмах, є частою темою дослідження. Є кілька широкоприйнятих підходів до вирішення цієї проблеми. Першим варіантом можна описати інтерактивні генетичні алгоритми (IGA). В IGA функція пристосованості базується на оцінках, наданих користувачем. Користувач оцінює різні рішення, і ці оцінки використовуються для визначення пристосованості. Представлений підхід включає три основні етапи: ініціалізацію популяції, еволюцію популяції за допомогою генетичних операцій (селекція, схрещування, мутація) та оцінку пристосованості з використанням оцінок користувача. Після декількох ітерацій процес зупиняється, коли знаходиться прийнятне рішення або коли досягнуто максимальної кількості ітерацій.

Перевагою цього підходу є те, що він може бути використаний для різних проблем, які вимагають участі користувача. Цей підхід дозволяє

користувачу краще контролювати процес пошуку, отримуючи рішення, які враховують його власні вимоги та бажання.

Іншим підходом є багатоцільові генетичні алгоритми (MOGA). В MOGA кілька цілей оптимізується одночасно, що дозволяє краще врахувати різноманітні інтереси користувача. Цей підхід може бути застосований для календарного планування. Проблеми планування календаря формуються як багатоцільова проблема оптимізації. Зазначається кілька цілей, які важливі для користувача, таких як мінімізація часу очікування між подіями, максимізація продуктивності та максимізація особистого часу. Маємо генетичний алгоритм, який оптимізує ці цілі одночасно. Результати включають колекцію рішень, які представляють кращий компроміс між різними цілями. Цей підхід дозволяє користувачам вибирати відповідне рішення відповідно до їх власних побажань. В цілому, стверджується, що MOGA забезпечує більш гнучкий та користувацький спосіб планування подій, порівняно з класичними одноцільовими генетичними алгоритмами.

Також можна звернути увагу на генетичні алгоритми з адаптивними функціями пристосованості. В цих алгоритмах функція пристосованості змінюється в процесі еволюції, що дозволяє врахувати динамічні вимоги користувача.

Вводиться адаптивна функція пристосованості, яка змінюється в залежності від стану популяції. Зокрема, цей підхід автоматично переключається між функціями пристосованості з високою та низькою селективністю в залежності від рівня різноманіття в популяції. Цей підхід дозволяє алгоритму краще впоратися з проблемами попередньої конвергенції (коли алгоритм зациклюється на декількох рішеннях замість пошуку нових, потенційно кращих рішень). Такий адаптивний підхід може також бути корисним, коли цілі або потреби користувача змінюються.

Ці методи можуть допомогти врахувати неоднозначність, динаміку і складність вимог користувача при використанні генетичних алгоритмів. У таблиці 1.6 представлені переваги та недоліки цих методів.

Таблиця 1.6 – Порівняння методів для вирішення задач персонального планування з урахуванням вимог користувача

Метод	Переваги	Недоліки
Інтерактивні генетичні алгоритми (IGA)	Оскільки користувач постійно задає оцінки, це може більш точно відображати його вимоги.	Може бути часомістним для користувача, оскільки він повинен постійно надавати оцінки.
Багатоцільові генетичні алгоритми (MOGA)	Генерує набір оптимальних рішень, з яких користувач може вибирати.	Може бути складно визначити, на скільки важливі різні цілі для користувача. Результатом є не одне рішення, а набір рішень, які користувачу потрібно обрати.
Генетичні алгоритми з адаптивними функціями пристосованості	Може враховувати зміни в потребах користувача. Може краще реагувати на динамічні умови.	Може бути складно визначити, як має відбуватися адаптація. Рішення можуть стати непередбачуваними або неконсистентними.

1.3.3 Практики використання генетичних алгоритмів в задачах формування розкладів

Як вказано в статті Ahmed Suleiman [3], генетичні алгоритми використовують для оптимізації призначення ресурсів, виконання завдань і створення ефективних графіків проектів. Він пропонує загальний опис процесу використання генетичних алгоритмів для планування проектів, Сулейман також зазначає, що ефективність генетичного алгоритму для планування і розкладу проектів залежить від таких чинників, як складність проблеми, якість функції придатності, обраний метод кодування і настройки параметрів самого генетичного алгоритму.

В іншому дослідженні [4] матеріал описує, як оптимізувати прокладання телепрограм за допомогою генетичних алгоритмів на Python.

Планування залежить від різних факторів. Ці фактори включають уподобання глядачів, програми конкуруючих каналів, святкові дні, спеціальні події, сезонні тренди, свіжий та старий контент, сюжетні лінії і хвилювання. Було запропоновано використання генетичного алгоритму для вирішення цієї задачі. Запропонована реалізація генетичного алгоритму містить такі етапи: ініціалізація популяції, встановлення функції пристосування, відбір, мутація. Реалізація була протестована на прикладі датасету, який містить рейтинги різних програм протягом деякого часу. В результаті було створено оптимальний графік телепрограм.

Також у 2021-му році вийшло дослідження [5] Проблема, над якою працює автор, полягає в створенні шкільного розкладу для його середньої школи, використовуючи евристичні та генетичні алгоритми. Завдання полягає в розробці розкладу, в якому один викладач в той же час може навчати тільки один клас, максимальна кількість уроків для предмета за день становить 2, викладачі можуть проводити максимум 12 уроків на тиждень та 5 уроків за день, кожний предмет для будь-якого класу може бути викладений тільки одним викладачем, а предмети, які повинні бути

проведені вранці, повинні відбуватися до перерви. У решті статті автор детально розглядає процес застосування генетичних алгоритмів для вирішення цієї проблеми, включаючи ініціалізацію класів, використання генетичних алгоритмів для призначення викладачів для кожного предмета в кожному класі, призначення класів до кожного класу та використання евристичних рішень для створення послідовності призначення класів.

На основі результуючих даних, він приходить до висновку, що евристичний метод був більш підходящим і масштабованим для впорядкування шкільного графіку. Він також наводить декілька пунктів, які можуть бути використані для поліпшення цього проекту в майбутньому.

1.4 Постановка задачі дослідження

Завданням дослідження є використання генетичних алгоритмів для формування персонального календарного планування на основі переписок в месенджері з урахуванням індивідуальних потреб користувача.

Актуальність дослідження полягає в тому, що сучасний ритм життя вимагає від людей виконання численних завдань, що підкреслює важливість ефективного планування часу. Більшість нашого життя відбувається в цифровому середовищі, де багато зустрічей організовуються через месенджери. Проте, існуючі методи персонального календарного планування зазвичай базуються на табличних даних, а не на текстових, і не призначені для планування, враховуючи комбінацію ваг показників, пріоритетів зустрічей та обмежень на кількість та час зустрічей. Це підкреслює необхідність розробки інтегрованого підходу, який би об'єднав формування вхідних даних з текстів чату та створення індивідуальних планів, враховуючи встановлені пріоритети та обмеження. У контексті персоналізованого календарного планування, еволюційні методи, зокрема генетичні алгоритми, заслуговують особливої уваги для реалізації такого підходу.

Об'єктом дослідження є процес персоналізованого календарного планування, що ґрунтується на аналізі повідомлень про зустрічі в чаті.

Предметом дослідження є еволюційні методи (генетичні алгоритми) персоналізованого календарного планування.

Метою дослідження є розробка та реалізація еволюційного алгоритму для персоналізованого календарного планування з використанням аналізу повідомлень про зустрічі в чаті та з урахуванням індивідуальних потреб користувача.

Задачі дослідження:

– аналіз процесів персонального календарного планування;

– дослідження підходів до обробки текстової інформації щодо календарного планування;

– дослідження еволюційних алгоритмів в задачах персонального календарного планування;

– удосконалення функції пристосованості з використанням комбінації ваг показників, критеріїв та обмежень заданих користувача;

– розробка методу персонального календарного планування на основі аналізу повідомлень з чату;

– тестування і оцінка ефективності розробленого методу на симульованих даних;

В результаті цього дослідження буде розроблено метод персоналізованого календарного планування з урахуванням інформації з повідомлень про зустрічі в чаті та з можливістю врахування типів зустрічей та потреб користувача.

2 ТЕОРЕТИЧНЕ ДОСЛІДЖЕННЯ ОБРАНОЇ ПРОБЛЕМИ

2.1 Удосконалення функції пристосованості

У роботі досліджувалась ідея створення системи планування зустрічей, яка враховує індивідуальні вимоги користувача шляхом встановлення ваг для різних критеріїв. Наприклад, для однієї особи може бути найважливішим провести більше часу з сім'єю, тоді як для іншої особи можуть пріоритетними бути робочі зустрічі. Добавляючи можливість користувачам встановлювати ваги і враховуючи ці ваги при формуванні розкладу, система може більше зосереджуватись на розкладах, які враховують індивідуальні потреби користувача. Тому перш за все, впроваджувались змінні ваги в оцінюванні розкладу.

Особливим методом, що застосовувався для реалізації цієї ідеї, була модифікація функції пристосованості в генетичному алгоритмі. Функція пристосованості використовується для визначення «якості» окремого розкладу, тобто наскільки добре він враховує введені ваги.

Функція пристосованості була налаштована на розрахунок комбінованого «рахунку», який враховує кожний критерій та його вагу у розкладі. Ці ваги були налаштовані користувачами, вони можуть бути динамічно змінені на основі потреб користувача. Наприклад, критерій пріоритетів зустрічей. Чим вище пріоритет певного типу зустрічі, тим більша ймовірність того, що цьому типу зустрічей буде віддано перевагу при формуванні розкладу. Відповідно до цього, користувачам надається можливість встановлювати пріоритети для різних типів зустрічей, які вони планують. Наприклад, переговори з клієнтами можуть важити більше, ніж внутрішні наради, або у когось може бути бажання максимізувати час відпочинку без роботи. Такі пріоритети тоді використовуються для налаштування ймовірностей вибору певних типів зустрічей під час формування розкладу. Це забезпечує більшу гнучкість в системі

планування, дозволяючи користувачам активно управляти своїм розкладом, зосереджуючись на важливих для них зустрічах.

Встановлені користувачами пріоритети для типів зустрічей тісно взаємодіють з вагами в функції пристосованості. Ці два аспекти системи взаємно посилюють один одного, створюючи більш гнучке та індивідуалізоване рішення для планування. Це означає, що розклади, які містять більше високо пріоритетних зустрічей, отримають більшу оцінку пристосованості. З іншого боку, ваги, встановлені для кожного типу критеріїв, впливають на важливість такого критерію як максимізація зустрічей з найвищим пріоритетом. Тим самим, роль у формуванні розкладу відіграють ваги, встановлені користувачами для кожного типу критеріїв. Крім пріоритетів критеріями можуть бути наприклад: мінімізація або максимізація перерв між зустрічами, максимізація зустрічей певного типу.

Генетичний алгоритм може бути налаштований з урахуванням різних параметрів, що відображають специфічні вимоги задачі та користувача [6]. Зокрема, можна налаштувати функцію пристосованості, яку можна записати у вигляді [6]:

$$f(X) = w1 * c1(X) + w2 * c2(X) + \ldots + wn * cn(X), \qquad (2.1)$$

де X – сукупність генів (параметрів планування);

$ci(X)$ – і-тий критерій якості, що вимірюється на основі X;

wi – вага і-того критерія у загальній сумі [6].

На основі такої структури функції пристосованості можна експліцитно задати важливість різних критеріїв при оцінці якості розкладу [6]. Параметри схрещування та мутації, а також типи операцій (одноточкове схрещування, багатоточкове схрещування, уніформне схрещування, мутація з оберненням, мутація з зміщенням тощо), можна вибрати залежно від потреб конкретної задачі [6].

Наприклад маємо такі критерії від користувача: кількість зустрічей – максимум 4 на день, тривалість кожної зустрічі – максимум 1 година, перерва між зустрічами – мінімум 1 година, час для спортзалу – менш важливий, ніж робочі зустрічі [6]. Кожен із цих критеріїв може бути представлений окремим геном [6]. Цільова функція може виглядати таким чином:

$$F(X) = w1 * c1(X1) + w2 * c2(X2) + w3 * c3(X3) + w4 * c4(X4), \quad (2.2)$$

де X – сукупність генів;

$X1$ – кількість призначених зустрічей;

$X2$ – тривалість кожної зустрічі;

$X3$ – час між зустрічами;

$X4$ – час для тренування;

wi – це вага кожного критерію;

$ci(Xi)$ – це функції, що оцінюють якість кожного параметра [6].

У формулі, ваги можна представити як результат деякої функції, яка визначає важливість кожного критерію на основі вимог користувача [6]. Наприклад:

$$F(X) = f1(X1) * c1(X1) + f2(X2) * c2(X2) + \ldots + fn(Xn) * cn(Xn), \quad (2.3)$$

де $fi(xi)$ – це функція, яка визначає вагу на основі важливості критерію для користувача [6].

Ця функціональність може бути реалізована як параметр, який користувач може встановити, навіть якщо це включає максимізацію кількості робочих зустрічей або скорочення часу на обіди. Вона також може виступати у формі висловлювання, коли користувач стикається з декількома суперечливими вимогами. В цьому випадку, кожен параметр алгоритму вимагатиме власної оцінювальної функції, яка відображатиме його

значимість для користувача. У результаті, ваги w у формулі стають динамічними та адаптованими до індивідуальних потреб користувача, замість того, щоб бути незмінними коефіцієнтами.

2.2 Розробка методу персонального календарного планування на основі аналізу повідомлень з використанням генетичного алгоритму

Основу для процесу планування та оптимізації було побудовано за допомогою генетичних алгоритмів. Генетичні алгоритми працюють за принципом «виживання найпридатніших», де найкращі «хромосоми» (тобто рішення) передаються до наступного покоління. В цьому контексті, ідея полягає в використанні генетичного алгоритму для створення оптимального плану дня, який бере до уваги розподіл часу та потреби користувача.

Ключовим елементом процесу стає робота функції пристосування за допомогою динамічних ваг. Ваги одержуються з аналізу повідомлень користувача і показують важливість кожного потенційного діяння протягом дня або ж математично задаються користувачем. Таким чином, якщо діяння як «зустріч з друзями дитинства» стає важливим на певний час, вона отримує більшу вагу та має більшу ймовірність бути включеною в оптимальний план дня.

Цей підхід дозволяє вивчити переваги та потреби користувача, і таким чином, запропонувати календарний план, який відповідатиме його потребам найкращим способом. Узагальнену схему роботи всієї системи можна побачити на рисунку 2.1.

На першому етапі буде зроблена передобробка даних. Тобто зі звичайного набору повідомлень – набору тексту, ми відберемо потрібні дані і запишемо їх у об’єкт формату, який використовується у генетичному алгоритмі. На другому етапі цей об’єкт буде доповнено типом зустріч. Цей тип в свою чергу буде визначатися за рахунок роботи класифікатора, на вхід

до якого буде подаватися текст, а на виході ми будемо отримувати тип зустрічі.

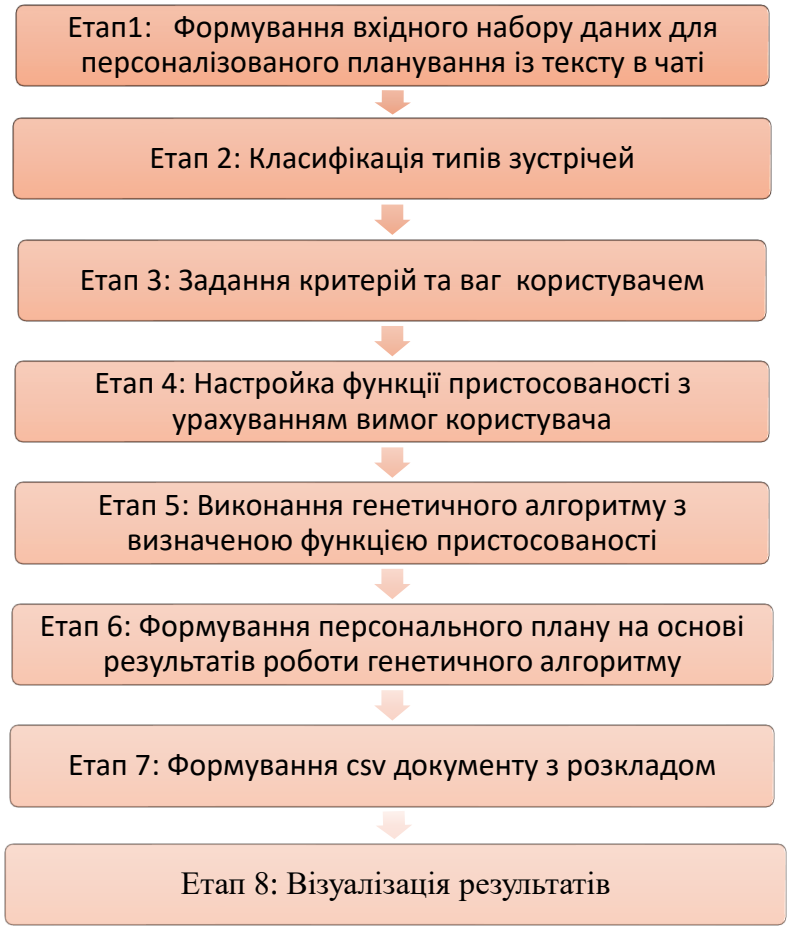

Рисунок 2.1 – Послідовність етапів методу персонального календарного планування

У процесі третього етапу буде проходити збір даних від користувача. Тобто користувач буде задавати свої вимоги щодо планування. Ці вимоги у рамках четвертого етапу будуть використовуватися для налаштування фітнес функції.

На 5-му етапі почнеться основна робота генетичного алгоритму. На цьому кроці ми створимо популяцію, визначимо для кожного індивіда значення фітнес-функції, а також проведемо схрещування і можливі мутації. Після цього буде перевірка показника пристосованості і або зупинка

алгоритму, або еволюція моделі та етапи 6 та 7, поки не буде досягнення визначеного показника пристосованості.

На фінальному, 8-му етапі, буде розроблено функцію для представлення фінального варіанту розкладу користувачу у форматі консольного виводу, а також у форматі csv.

Розроблений метод використовує в основі базові властивості генетичних алгоритмів для побудови індивідуального плану.

Генетичні алгоритми (GA) є класом оптимізаційних алгоритмів, які використовують підхід, натхненний біологічною еволюцією і генетикою. Вони були вперше введені в 1970-х роках Джоном Голландом. Ключові концепції, що лежать в основі можна описати як: індивід – потенційне рішення задачі оптимізації, геном або хромосома – масив, який кодує індивіда, функція пристосованості – функція, яка оцінює якість кожного індивіда з урахуванням задачі оптимізації, селекція – процес вибору індивідів для подальшої репродукції залежно від їх пристосованості, схрещування або кросовер – процес комбінування геномів двох індивідів для створення нового індивіда та мутація – випадкове змінення генів в геномі для запобігання переходу до локального максимума або мінімума.

Генетичний алгоритм працює починаючи від популяції первинних індивідів, які генеруються випадковим чином. Кожного покоління процеси селекції, схрещування та мутації використовуються для створення наступного покоління рішень. В першу чергу, індивіди вибираються для схрещування на основі їх пристосованості через процес селекції. Є лінійна та рангова селекція, що використовують пропорційні ймовірності відносно функції пристосованості. Далі, пари індивідів схрещуються для створення дітей через процес кросовера. Є декілька стратегій для кросовера, такі як одноточковий, багатоточковий або уніформний кросовер. На останок, деякі гени в дітей можуть бути випадково змінені через процес мутації для підтримки різноманітності в популяції. Процес повторюється до тих пір, поки не буде досягнуто критерій зупинки, такий як максимальне число

поколінь або досягнення певної мети пристосованості. Загалом, генетичний алгоритм вважається збіжним, коли наявність індивідів поєднується з великим значенням пристосованості. Загальна схема роботи генетичного алгоритму зображена на рисунку 2.2.

Рисунок 2.2 – Класична схема роботи генетичного алгоритму

Алгоритм генетичного алгоритму може бути описано математично наступним чином. Першим етапом є ініціалізація, яка полягає у створенні популяція розміром λ. Кожен індивід в популяції представляє собою можливе рішення і має випадковий набір генів x_i.

$$x_i \sim U(a, b), i = 1, \ldots, \lambda, \quad (2.4)$$

де $U(a, b)$ – це рівномірний розподіл від a до b.

На другому етапі для кожного індивіда в популяції обчислюється значення функції пристосованості $f(x\_i)$.

Під час третього етапу обираються μ індивідів, які будуть брати участь у схрещуванні, на основі їх пристосованості. Обираються μ індивідів з найкращими значеннями $f(x\_i)$.

На четвертому етапі двоє індивідів x_i і x_j обираються з популяції і схрещуються для створення нових особин y_k.

$$y_k = \alpha_k * x_i + (1 - \alpha_k) * x_j, k = 1, \ldots, \mu, \quad (2.5)$$

де $\alpha\_k \sim U(0, 1)$ випадковий коефіцієнт.
З деякою імовірністю p ген в особини може бути випадково змінені.

$$y_k = y_k + \eta_k, \text{якщо } u_k < p, k = 1, \ldots, \mu, \quad (2.6)$$

де u_k ~ U(0, 1) – випадкове число;

η_k ~ N(0, σ^2) – випадкове число з нормального розподілу.

Під час 5-го етапу перевіряється критерій зупинки. Якщо критерій зупинки не досягнуто, нова популяція стає поточною популяцією і процес повертається до кроку 2. Інакше, алгоритм зупиняється і повертає найкраще рішення, яке було знайдено.

Цей циклічний процес дозволяє мати поступову еволюцію популяції в напрямку покращення рішень.

2.3 Формування вхідного набору даних для персоналізованого планування

На першому кроці предпроцесінга даних безпосередньо оброблялись повідомлення з месенджера. Вони могли включати дані про чати, авторів і

час передавання повідомлень, а також власне текст повідомлень. Було важливо розглянути всі потрібні метадані та їх формат.

У цьому розділі обговорюється обробка отриманих текстових даних, щоб зрозуміти та виокремити релевантні частини інформації, які необхідно врахувати в генетичному алгоритмі. Для цього були використані методи обробки природньої мови (NLP), такі як фільтрація шумів, видаляння стоп-слів, стемінг, лематизація, розпізнавання іменованих сутностей (NER) тощо.

Першим кроком в підготовці тексту для обробки було очищення тексту. Регулярні вирази (Regular Expressions) були використані для видалення спеціальних символів, знаків пунктуації, посилань або інших нерелевантних елементів. Також всі слова були оформлені в нижньому регістрі для подальшої обробки.

Для ідентифікації відомостей про дату та час використовувалися модулі Python для обробки дат та часу, такі як dateparser та parsedatetime, для виявлення та обробки виразів дати/часу в тексті.

Для виявлення місця проведення зустрічі використали спеціалізовані інструменти для виявлення Named Entity Recognition (NER), такі як spaCy – для виявлення згадок про місця в тексті.

Для визначення типів зустрічі, ми створили і навчили класифікатор тексту на основі прикладів тексту для кожного типу зустрічі, який нам потрібен. Це могло бути зроблено за допомогою різних методів машинного навчання, включаючи наївний Баєсівський класифікатор, машини опорних векторів (SVM), випадкові ліси, градієнтний бустінг, нейронні мережі тощо. Був обраний Баєсівський класифікатор. Загальну схему обробки даних можна побачити на рисунку 2.3.

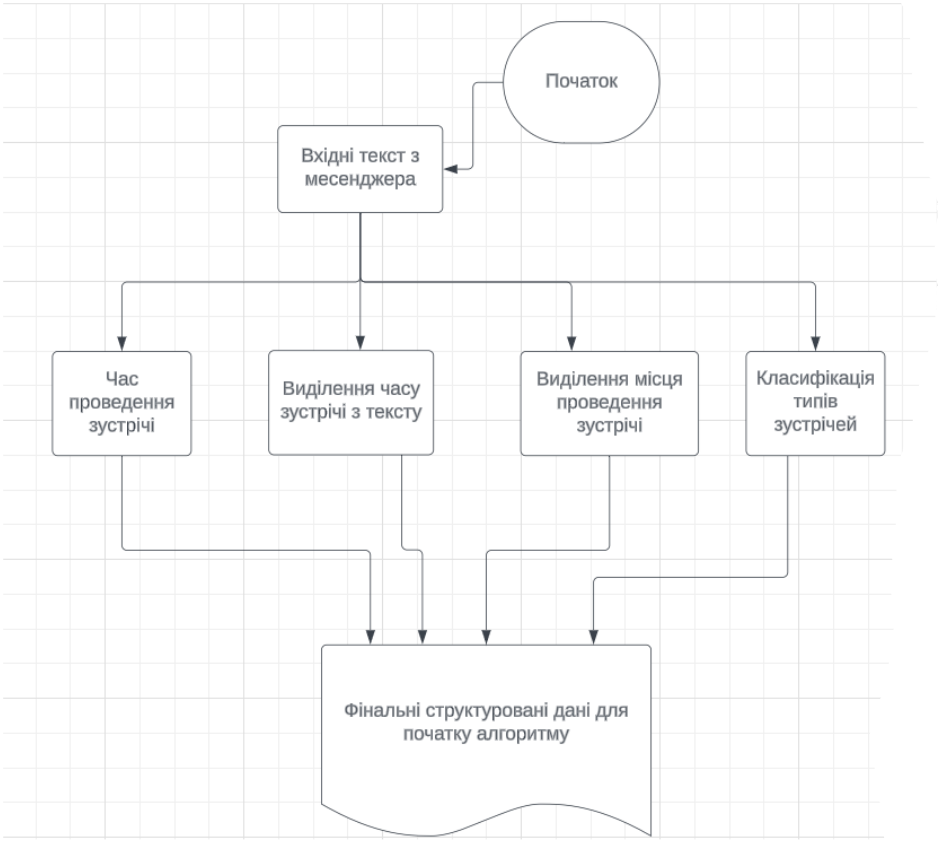

Рисунок 2.3 – Формування вхідного набору даних для персоналізованого планування

Класифікатор, в свою чергу, був навчений за допомогою заздалегідь помічених даних, які мають приклади тексту з різних типів зустрічей. Тобто був використаний процес навчання з вчителем. Так як він вимагає набір тексту, в якому кожен приклад позначено як відповідний до одного з типів зустрічей, було створено вибірку українською мовою.

Так як планувалось що оброблятися буде текст написаний українською мовою, треба було думати над вирішенням проблеми обмеженості деяких готових інструментів для обробки тексту. Багато пакетів, які планувалось використовувати не підтримують обробку української мови.

Обраний підхід базується на використанні перекладу тексту та NLP для визначення важливої інформації. Складається з двох основних етапів:

– переклад тексту: головна мета цього етапу – перетворити український текст на англійську мову, що має більше інструментів для аналізу текстових даних. Для цього використовується сервіс Google Translate через бібліотеку googletrans;

– обробка перекладеного тексту: після перекладу текст аналізується за допомогою бібліотеки spacy, як спеціалізованої для англійської мови. Зокрема, використовується векторна модель en_core_web_sm для визначення сутностей у тексті (місцезнаходження, дата, час і т.д.).

Основна ідея цього підходу полягає в тому, щоб скористатись сильними сторонами вже існуючих інструментів для англійської мови (spacy, googletrans) для вирішення завдання обробки тексту українською мовою.

Якщо говорити про недоліки цього підходу перша проблема – етап перекладу. Googletrans здійснює переклад тексту у потоці, тому на його результати можуть впливати проблеми з швидкістю інтернету або обмеженнями API. Крім того, оскільки структура речень може значно відрізнятися в різних мовах, переклад може переставляти слова в тексті, що ускладнює визначення сутностей.

Вхідні дані для генетичного алгоритму мають бути структурованою формою інформації одержаної на попередніх етапах обробки і аналізу тексту. Генетичний алгоритм аналізує ці дані як «популяцію» потенційних розв'язків. Кожен «індивід» (або «особина») в популяції представляє можливий варіант переліку зустрічей. Кожна зустріч представлена геном, що відповідає її характеристикам. Це дата і час, місце проведення, тип зустрічі та інша важлива інформація(підготовка, тривалість). Зазвичай кожен ген в індивіді кодується як числове значення. Наприклад, дату і час можна представити як часову мітку Unix, місце проведення зустрічі – як індекс в таблиці місць, а тип зустрічі – як індекс в таблиці типів зустрічей.

Параметри задані користувачем: користувач задає ряд обмежень і параметрів, що впливатимуть на процес пошуку оптимального розв'язку генетичним алгоритмом. Наприклад, користувач може задати бажаний час початку й закінчення робочого дня, максимальну кількість зустрічей за день, час подорожі між місцями проведення зустрічей, пріоритетність за типом зустрічі тощо. Ці параметри повинні бути включені в модель генетичного алгоритму як частина функції пристосування, яка визначає «якість» кожного індивіда в популяції.

2.4 Класифікація типів зустрічей

Для побудови класифікатора для визначення типу зустрічі було вирішено мати певний набір типів зустрічей. Ми мали такі можливі варіанти: робоча зустріч, зустріч з близькими, зустріч з друзями, спортзал, час на хобі.

Була побудована модель класифікації на основі машинного навчання, щоб визначити тип зустрічі, основуючись на текстових даних. Першим кроком було формування підготовчого набору даних. Були зібрати деякі дані з описом зустрічей від користувача, з якими асоціюються конкретні типи зустрічей (робоча зустріч, зустріч з близькими, зустріч з друзями, спортзал, час на хобі і т.п.). Зустрічі повинні бути якомога більш репрезентативними. Набор було зібрано на основі власних даних. А також додатково згенеровано за допомогою інструментів генерації тексту. Він має таку структуру: опис – тип події.

Далі проводилась підготовка тексту: очищення тексту від шуму, нормалізація слів (лематизація), видалення стоп-слів і так далі. Після цього проводилась конвертація тексту в числові значення (зазвичай використовується TF-IDF чи методи векторизації слів, такі як Word2Vec). В якості моделі було використано специфічну для задачі текстової класифікації модель, таку як Naive Bayes.

Опишемо приклад конкретної реалізації на Python за допомогою бібліотеки scikit-learn (лістинг 2.1).

Лістинг 2.1 – Програмний код.

```
X = [«...», «...», «...»]
y = [«робоча», «спортзал», «зустріч з друзями», ...]
le = LabelEncoder()
y = le.fit_transform(y)
text_clf = Pipeline([('vect', TfidfVectorizer()), ('clf',
RandomForestClassifier())])
text_clf.fit(X, y)
meeting_desc = «...»
predicted_type = text_clf.predict([meeting_desc])
print(le.inverse_transform(predicted_type))
```

Таким чином, ми мали модель, яка здатна автоматично класифікувати типи зустрічей залежно від інформації, яку надає користувач.

2.5 Налагодження функції пристосованості та імплементація генетичного алгоритму

На рисунку 2.4 можна побачити порядок роботи створеного генетичного алгоритму. Він складається з загальних 5-ти етапів. Кожний з який буде описано нижче.

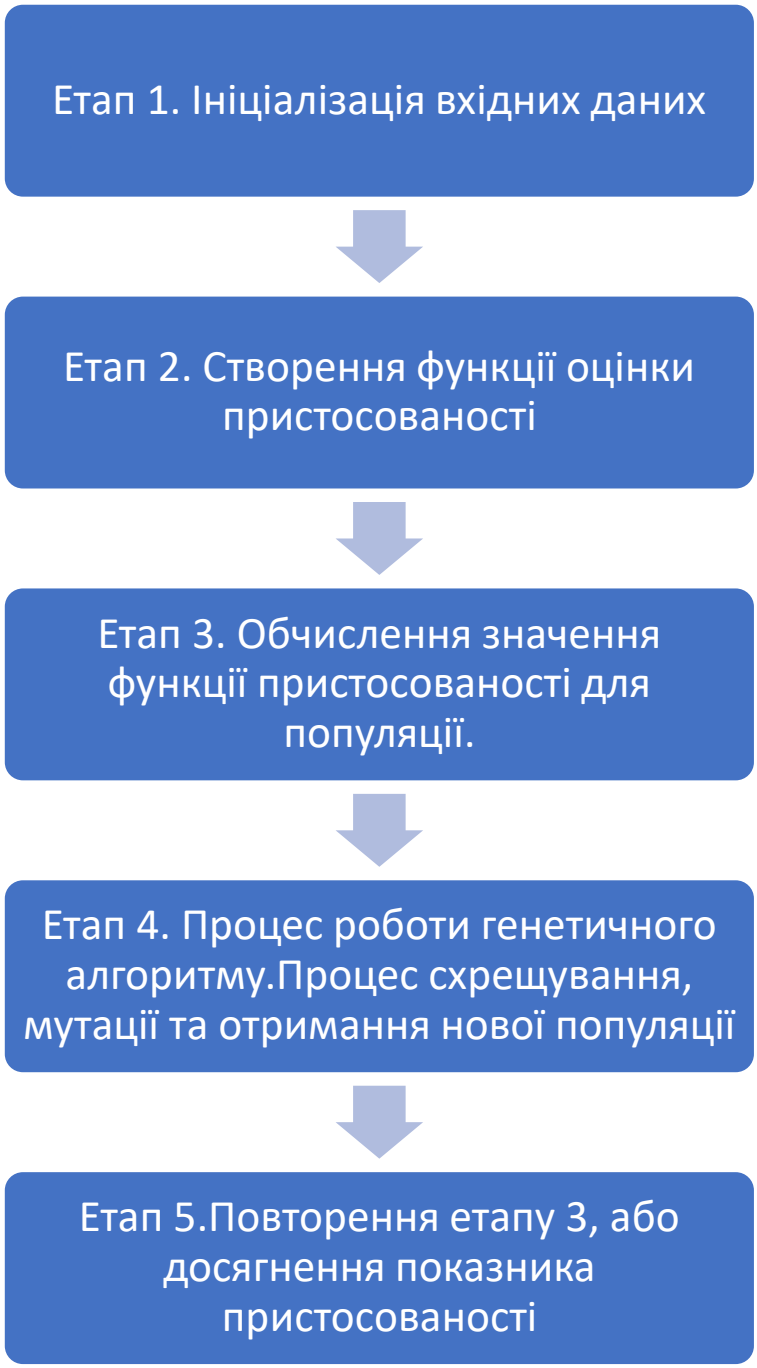

Рисунок 2.4 – Порядок роботи розробленого генетичного алгоритму

Перший етап можна визначити як створення та обробка необхідних вхідних даних. Тобто на цьому етапі задаються вхідні зустрічі та потреби користувача. На другому етапі створюється функція оцінки пристосованості. В неї входить визначення ваг для критеріїв користувача, врахування всіх критеріїв користувача, таких як наприклад функція, де ми перевіряємо загальну кількість пріоритетів зустрічей у розкладі. Далі

створюється процес відповідальний за роботу генетичного алгоритму. Тут описано процес створення початкової популяції потенційних розкладів зустрічей. Цим кроком створюється початкова популяція, яка складається з довільно сформованих індивідуумів (або хромосом).

У контексті нашої проблеми, кожен індивід – це потенційний розклад зустрічей. Для ініціалізації популяції, як правило, використовується випадкова генерація, оскільки на початку у нас немає ніякої інформації про те, як зустрічі можливо розподілити. Розмір популяції (N) - це загальна кількість індивідуумів в популяції. Хромосома - у контексті цього завдання кожна хромосома представляє собою розклад, який складається з набору визначених зустрічей. Ген - кожна хромосома складається з генів, де кожен ген відображає окрему зустріч. Отже, кожна хромосома X може бути представлена як:

$$X = \{g1, g2, \ldots, gn\}, \tag{2.7}$$

де gi – це ген, що представляє інформацію про і-ту зустріч в розкладі.

Кожен ген вміщує інформацію про зустріч. В залежності від обмежень розкладу та параметрів зустрічей, кожний ген можна детально подати у вигляді набору атрибутів, наприклад: {тип зустрічі, час початку, тривалість, місце проведення}.

Повторюючи цей процес N разів, ми отримаємо початкову популяцію потенційних розкладів. На цьому етапі немає гарантії, що будь-який з цих розкладів є вірним рішенням – це просто стартова точка для алгоритму.

На 3 етапі проходить оцінка пристосування, тобто визначення якості кожного розкладу, зокрема згідно з обмеженнями та параметрами користувача. Оцінка пристосування (або функція пристосування) – це процедура, яка визначає якість кожного розкладу в популяції і повертає числове значення, що оцінює, наскільки добре розклад відповідає заданим параметрам та обмеженням. Нехай F(X) – це функція пристосування, де X –

це розклад. Ця функція бере розклад зустрічей і оцінює його відносно критеріїв, встановлених користувачем. Спочатку функція перевіряє, чи можна створити розклад. Якщо він не може бути створений, це означає, що перевірка фітнеса не можлива, відповідно, повертається значення -100, яке є дуже високим штрафом. Тоді для кожної події у розкладі обраховуються наступні показники:

– «Priority_score()», нормована сума пріоритетів всіх зустрічей. Чим більше в розкладі зустрічей з високим пріоритетом, тим вище цей критерій:

– «Total_meetings_min()», це нормована мінімальна кількість зустрічей конкреного типу;

– «Num_meetings_max()», це нормована кількість зустрічей в розкладі конкретного типу,;

– «User_score()», який включає в себе бал користувача. Цей бал буде знижений, якщо буде пропущена обов'язкова робоча подія;

– «Break_len()», який розраховує час на відпочинок.

Ці різні показники об'єднуються для отримання загального балу фітнесу. Кожен показник множиться на відповідний ваговий коефіцієнт, а потім бали додаються разом. Значення фітнес-функції повертається як результат функції, який потім використовується генетичним алгоритмом для оцінки якості кожної особини в популяції.

Селекція в генетичних алгоритмах – це процес вибору розкладів (індивідів), які використовуються для створення наступного покоління. Метод селекції в генетичних алгоритмах робиться так, щоб наступне покоління мало тенденцію до покращення показників пристосування. Тобто ми хочемо вибирати «кращі» розклади, які мають вищу оцінку пристосування. Однак слід зберігати певний рівень генетичного різноманіття, тому деякі «гірші» розклади також повинні бути вибрані.

Схрещування, або кросовер – це процедура «обміну генами» між двома батьківськими розкладами (індивідами) для створення нового розкладу (потомка).

Нехай у нас є два батьківських розкладу P1 та P2:

$$P1 = \{g1, g2, \ldots, gn\}$$
$$P2 = \{h1, h2, \ldots, hn\} \quad , \qquad (2.8)$$

де gi та hi – гени, що відповідають за інформацію про зустрічі в цих розкладах.

Батьківська хромосома розбивається на дві частини в довільно обраному пункті (або пунктах у випадку багато точкового кросоверу) і обмінюються частинами між двома хромосомами для створення потомства. Наприклад, вибравши одно точковий кросовер з пунктом схрещування після гену g3, ми маємо:

$$Offspring1 = \{g1, g2, g3, h4, \ldots, hn\}$$
$$Offspring2 = \{h1, h2, h3, g4, \ldots, gn\} \quad , \qquad (2.9)$$

Таким чином, створюються два нових розклади, які поєднують гени своїх батьків. Цей процес повторюється для різних пар батьківських розкладів, доки не буде сформовано нове покоління розкладів.

Мутація в генетичних алгоритмах – це процедура випадкової зміни деяких генів для створення різноманітності в популяції та запобігання застрягання в локальних мінімумах.

Нехай у нас є розклад X:

$$X = \{g1, g2, \ldots, gn\}, \tag{2.10}$$

де gi – ген, що містить інформацію про і-ту зустріч у розкладі.

Процес мутації відбувається поетапно. Спочатку для кожного гена gi в розкладі X з визначеною ймовірністю мутації (p) впроваджуємо випадкову зміну. Ця зміна може бути, наприклад, зміною часу або місця зустрічі. Якщо випадкове число (з розподілу від 0 до 1) менше p, то мутація відбувається. В протилежному випадку ген залишається без змін. Результатом цього процесу є розклад, що певною мірою є відмінним від оригінального розкладу X. Це дозволяє алгоритму «досліджувати» нові області простору рішень, які можуть бути недоступні лише через схрещування.

Як правило, ймовірність мутації встановлюється досить низькою (наприклад, 0,01 або 1%), оскільки висока частота мутацій може занадто сильно «дестабілізувати» популяцію та заважати збіжності алгоритму.

Після етапу схрещування і мутації утворюється нове покоління розкладів. Наступним етапом є заміна старої популяції новою. Тобто закінчується 4-й етап.

Після того, як утворено нову популяцію, алгоритм повертається до етапу 3 і повторює процес – від оцінки пристосування до заміни популяції. Циклічне повторення цієї послідовності дій дозволяє поступово «шукати» у просторі можливих розкладів тих, що максимально відповідають вимогам користувача. Процес повторення триває до досягнення визначеного критерію зупинки.

Після завершення процесу ітерації, коли було досягнуто критерій зупинки, потрібно вибрати остаточне рішення – найкращий розклад з останньої популяції.

Якщо зробити величину пристосування прямим показником якості розкладу, то вибір найкращого розкладу є простим: це просто розклад з найвищою величиною пристосування. Отже, фінальний розклад R буде визначено на основі максимального значення функції пристосування:

$$R = argmax\, F(X)\,, \tag{2.11}$$

де F(X) – це функція пристосування;

X це розклад в популяції.

3 ПРОГРАМНА РЕАЛІЗАЦІЯ ТА ЕКСПЕРИМЕНТАЛЬНА ПЕРЕВІРКА

3.1 Обґрунтування вибору інструментарію для реалізації розробленого методу

Реалізація оновленого генетичного алгоритму для оптимізації розкладу на основі переписок в месенджері вимагає використання відповідного набору інструментів. Під час роботи були використані інструменти обробки природної мови (NLP). Це дозволило аналізувати та інтерпретувати текстові дані, що отримані через месенджери. Python бібліотеки, такі як Natural Language Toolkit (NLTK), spaCy або TextBlob. Для реалізації та покращення генетичного алгоритму використовувались бібліотеками машинного навчання та оптимізації, такі як scikit-learn, TensorFlow, Keras та DEAP (Distributed Evolutionary Algorithms in Python).

В майбутньому при реалізації повноцінного застосунку будуть використовуватись бібліотеки для роботи з API месенджерами: для інтеграції із месенджерами, такими як Telegram або WhatsApp, можуть бути використані відповідні бібліотеки, такі як python-telegram-bot або yowsup. Окрім цього, можливе використання обчислювальних ресурсів хмарних платформ, таких як Google Cloud, Amazon Web Services або Microsoft Azure, які можуть надати можливість використовувати високу обчислювальну потужність, що потрібна для великих або складних оптимізаційних завдань.

3.2 Підготовка вхідних даних для персоналізованого календарного планування

На першому етапі розробки методу ми займалися предпроцесінгом вхідних даних. Перед тим як працювати з генетичним алгоритмом спочатку треба було підготувати дані. Загальна обробка даних включала в себе

отримання повідомлення на вхід, а на вихід ми отримували об’єкт, який в свою чергу вже був вхідним параметром для генетичного алгоритму. Об’єкт містив в собі відомості про час та дату зустрічі, тип зустрічі, місце зустрічі, тривалість.

У нашому підході до обробки повідомлень використовується дві основні функції, що досліджують текст: get_meeting_details і predict_meeting_type. Вони допомагають визначити деталі зустрічі й прогнозувати тип зустрічі, пов'язаний з текстом повідомлення.

Використані бібліотеки:

– pickle – модуль для серіалізації та десеріалізації об'єктів Python;

– re – бібліотека для роботи з регулярними виразами;

– googletrans – безкоштовна та необмежена бібліотека, яка реалізує API Google Translate;

– spacy – бібліотека для авансової обробки природних мов.

На етапі реалізації обробки повідомлень функція get_meeting_details спочатку використовується Google Translate API через модуль googletrans для перекладу тексту з української на англійську мову.

```
translation = translator.translate(text, src='uk', dest='en')
translated_text = translation.text
```

Далі використовується spacy для обробки тексту та виявлення сутностей, таких як дата, час і місце проведення.

```
nlp = spacy.load(«en_core_web_sm»)
doc = nlp(translated_text)
```

Тобто витягуються специфічні характеристики зустрічі, такі як дата, час та місце проведення.

```
date = [ent.text for ent in doc.ents if ent.label_ == 'DATE']
time = [ent.text for ent in doc.ents if ent.label_ == 'TIME']
```

```
location = [ent.text for ent in doc.ents if ent.label_ in ['GPE', 'ORG']]
```

Функція повертає ці три елементи у вигляді кортежу.

Функція predict_meeting_typeи спочатку завантажуються збережені модель та векторизатор з допомогою pickle:

```
model = pickle.load(open(«naive_bayes_model.pkl», «rb»))
vectorizer = pickle.load(open(«count_vectorizer.pkl», «rb»))
```

Текст повідомлення трансформується у числовий формат з використанням CountVectorizer, а потім модель використовується для передбачення типу зустрічі використовуючи наш готовий класифікатор:

```
text_vect = vectorizer.transform([text])
predicted_type = model.predict(text_vect)
```

І в фіналі функція повертає передбачений тип зустрічі.

Приклади роботи представлені на рисунках 3.1 та 3.2.

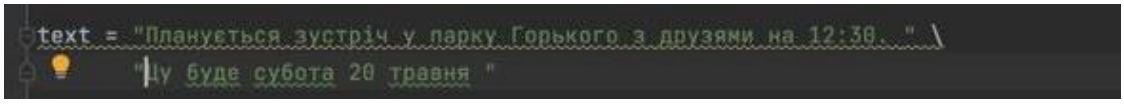

Рисунок 3.1 – Приклад вхідного повідомлення

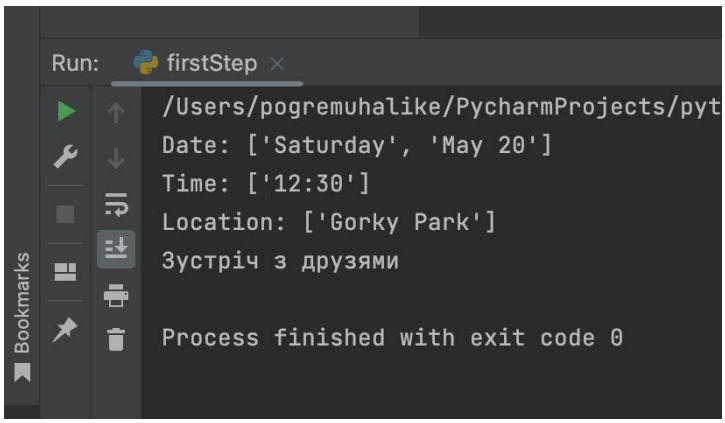

Рисунок 3.2 – Приклад обробки

3.3 Реалізація класифікатора для типів зустрічей

На другому етапі розробки методу була реалізація класифікатора використовувалась мова програмування Python і бібліотеки:

- from sklearn.model_selection import train_test_split;
- from sklearn.feature_extraction.text import CountVectorizer;
- from sklearn.naive_bayes import MultinomialNB;
- from sklearn.metrics import classification_report;
- import re;
- import pandas as pd;
- import pickle.

Бібліотеки pandas і sklearn для роботи з даними і машинним навчанням, re для обробки тексту, і pickle для збереження і завантаження моделей. Pandas - основна бібліотека для обробки даних в Python, яка надає структури даних та функції для швидкої обробки і аналізу даних. Sklearn – набір бібліотек Python для машинного навчання та аналізу даних.

Також використовувались model_selection.train_test_split – розбиває дані на тренувальну і тестову вибірку. Feature_extraction.text.CountVectorizer – використовується для трансформації текстових даних в число як частину передобробки. Naive_bayes.MultinomialNB - реалізація наївного Баєсовского алгоритму заснована на теоремі Байеса з припущенням «наївності» між кожною парою ознак. Metrics.classification_report - функція для побудови звіту про класифікацію, яка включає такі ключові метрики, як точність, повнота, f1-score.

Детальна обробка тексту проходила у функції preprocess_text(), яка виконує ряд операцій, включаючи видалення особливих символів, розділення тексту на токени, переведення тексту в нижній регістр. Далі проходить розділення датасету на тренувальний та тестовий

```
X_train, X_test, y_train, y_test = train_test_split(df['Приклад'], df['ТИП події'], test_size=0.2)
```

Наступним етапом створюється CountVectorizer, ми перетворюємо текст у числовий формат, що дозволяє використовувати його в моделях машинного навчання. Запускаємо модель наївного Баєса на тренувальних даних, виконуючи .fit():

```
clf = MultinomialNB()
clf.fit(X_train_vect, y_train)
```

Після цього перетворюємо тестову вибірку за допомогою vectorizer.transform і застосуємо модель, щоб отримати прогнози:

```
X_test_vect = vectorizer.transform(X_test)
y_pred = clf.predict(X_test_vect)
```

І в кінці зберігаємо навчену модель і векторайзер для подальшого використання.

Для реалізації класифікатора типу зустрічі нам був потрібен дата-сет з різними варіантами повідомлень про зустрічі різного типу. Перед тим, як розробляти та тестувати модель машинного навчання, потрібно підготувати і оптимізувати дата-сет.

На етапі збору даних використовувались різні ресурси такі як API, веб-скрапінг, генерація тексту. Важливо було забезпечити репрезентативність дата-сету для проблеми. Також було важливо враховувати зміст повідомлень, вони повинні були чітко прив'язані до якогось типу події, не повинні були бути абстрактними і загальними. Було проведено етап очистки даних від дублікатів, помилок та екземплярів які не були інформативними для вирішення нашої задачі.

Дуже важливо було перевірити дані на наявність неточностей, щоб упевнитися, що вони надійні. А також, що кожен екземпляр дійсно відповідав своєму типу зустрічі.

В результаті було створено дата-сет на 1000 екземплярів зі структурою: повідомлення – тип зустрічі описаний в повідомленні. В дата-

сеті є такі типи зустрічей і приклади для них: робочі зустрічі, зустрічі з друзями, зустрічі з сім'єю, хобі, обід, спорт, відпочинок. Приклади даних в дата-сеті можна побачити на рисунку 3.3 та в таблиці 3.1.

Таблиця 3.1 – Приклад даних в дата-сеті

Приклад повідомлення	**Тип зустрічі**
На вихідних плануємо сімейний вечір у ресторані з моїми батьками	Зустріч з родиною
Коханий, як ти вважаєш, проведемо нашу річницю весілля у театрі? Вже взяла квитки на цей особливий вечір!»	Зустріч з родиною
На наступному тижні приїде мій друг дитинства, ми плануємо сходити в бар.	Зустріч з друзями
Там вийшов новій фільм, ми з подругою підемо у кіно.	Зустріч з друзями
На цьому дзвінку визначимо підходи до співпраці з брендами для рекламного партнерства.	Робоча зустріч
В нас вчора впав продакшен, треба термінова здзвонитись.	Робоча зустріч
В мене заняття з малювання завтра.	Хобі
Я ходжу на вокал по вівторках і четвергам.	Хобі

Продовження таблиці 3.1

Кожну суботу я займаюсь йогою в парку.	Спорт
Пішла в спортзал, ходжу три рази на тиждень.	Спорт
Треба буде спланувати обідню перерву між роботою	Обід
Плануємо у вихідні поїхати на природу, трохи відпочити від міста.	Відпочинок

Зустріч з родиною	13. "Цього вікенду плануємо зустрітися з сім'єю на природі. Надзвичайно раді провести цей час разом і насолоджу
Зустріч з родиною	14. "Сьогодні плануємо вечірку з сестрою та братом. Не можемо дочекатися зустрічі та веселощів разом!"
Зустріч з родиною	15. "Збираємося зустрітися з родичами на каву. Надзвичайно раді провести цей час разом і поділитися новинами."
Зустріч з родиною	16. "Цього вікенду відправляємось до бабусі з дідусем. Не можемо дочекатися зустрічі та спілкування з ними!"
Зустріч з родиною	17. "Плануємо зустріч з друзями та рідними на вечірці. Це буде чудова нагода провести час разом та насолоджува
Хобі	6. "Цього місяця планую присвятити час своєму улюбленому хобі - вишивці хрестиком. Це допомагає мені розслаб
Хобі	7. "Планую провести вечір за виготовленням прикрас. Це моє улюблене хобі, яке допомагає мені відпочити та знай
Хобі	8. "Цього тижня планую відвідати курс малювання. Це буде важливий крок для мого розвитку та вдосконалення на
Хобі	9. "Планую провести вечір за читанням книги. Це моє улюблене хобі, яке допомагає мені розслабитися та розвинут
Хобі	10. "Завтра вирішила відвідати майстер-клас з виготовлення меблів. Це буде цікавий досвід для мене у цій галузі."

Рисунок 3.3 – Приклад даних в дата-сеті

Для більшої реалістичності було також враховано кілька важливих деталей. Для опису робочих зустріч охоплювалось декілька видів професій. Тобто зустрічі та плани на тему роботи для архітектурної сфери, медицини, реклами, IT, менеджерства, фармації і тп. Також при створенні записів на тему спортивних занять і опису повідомлень можна зустріти йогу, спортзал, плавання, волейбол, футбол, поло, гольф і тп. Так само для хобі: танці, вокал, малювання, гончарство, музика і тп.

В результаті вибірка не вийшла досить великою і не охоплює всі можливі варіанти повідомлень для навчання. І в майбутньому її треба буда удосконалювати і розширяти. А також додавати нові види зустрічей. Але для першого етапу використання класифікатора цього було достатньо.

3.4 Персональне календарне планування з використанням генетичного алгоритму

На 3-му етапі методу ми визначали критерії користувача та формат вхідних даних. Джерелами були самі повідомлення, метадата месенджера та параметри введенні користувачем. Приклад представлений на рисунку 3.4.

Рисунок 3.4 – Приклад вхідних даних до алгоритму

На рисунку 3.5 можемо побачити що на даному етапі дані представлені списком зустрічей у вигляді об’єктів. Де в нас є: type, person, mandatory – актуально тільки для зустрічей пов’язаних з роботою, дата, час та тривалість. Передбачається що першу частину даних алгоритм буде отримувати з попередньої обробки повідомлення, друга частина даних буде приходити з базовою інформацією про повідомлення. Наприклад: від кого – кому було відправлене повідомлення, дата відправки і час. І третя частина інформації буде введена користувачем. Більш детально можна побачити у таблиці 3.2.

Таблиця 3.2 – Ресурси для отримання вхідних даних

1-й ресурс: обробка повідомлення	З цього ресурси ми отримаємо: дату, час, місце і тип зустрічі.
2-й ресурс: метадата повідомлення	З ким планується зустріч, дата отримання повідомлення.
3-й ресурс: вхідні данні від користувача	Пріоритети зустрічей, критерії, ваги.

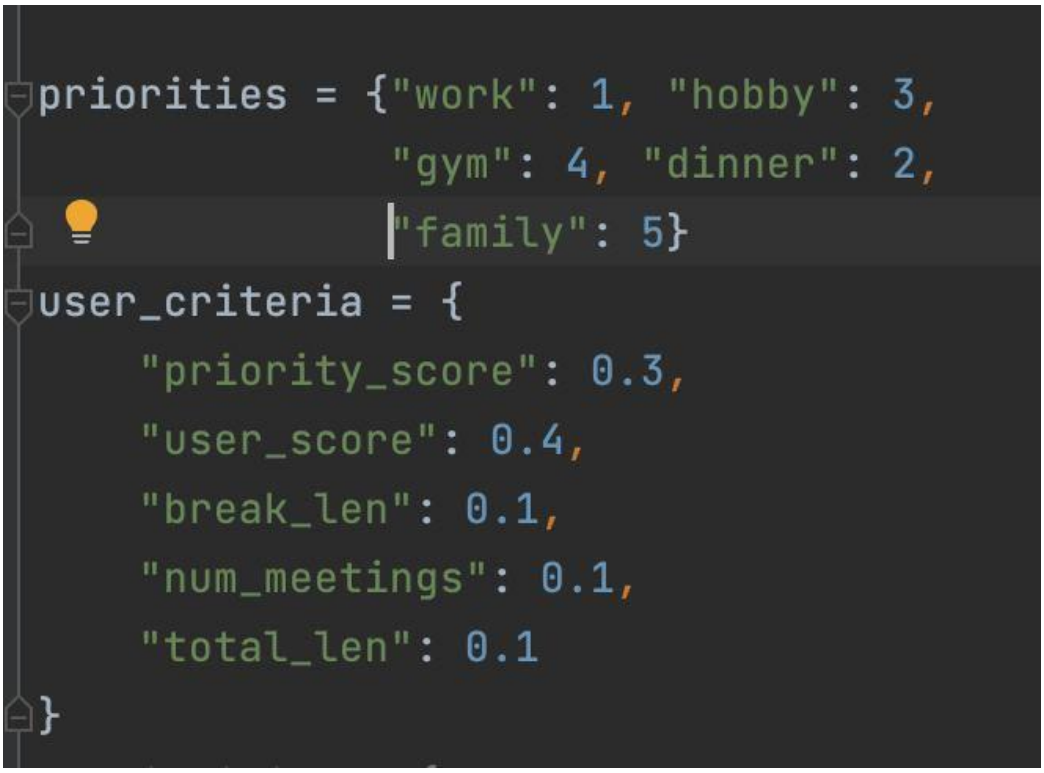

Рисунок 3.5 – Вхідні параметри від користувача

На рисунку 3.5 можемо бачити данні , які задаються користувачем. Це пріоритети для кожного типу зустрічі та критерії від користувача. На сьогодні їх варіації є обмеженими. Тобто кількість можливих типів зустрічі залежить від даних дата-сету. Якщо в майбутньому ми захочемо збільшити кількість типів, треба буде додати у вибірку нові данні і навчити наш класифікатор.

Якщо ми говоримо про user_criteria. У контексті нашого програмного забезпечення, user_criteria – це словник, що визначає вагові коефіцієнти різних критеріїв, які використовуються для обчислення «придатності» розкладу. «Придатність» – це концепція, що використовується у генетичному програмуванні для оцінки ефективності розв’язку проблеми. У нашому коді критерії включають наступне:

– priority_score: Чим вище рейтинг пріоритету, тим важливіше зустріч в розкладі. Він множиться на 0.3 при обчисленні загальної оцінки придатності;

– user_score: Воно керує тим, наскільки важливо враховувати інтереси користувача. Його вага у загальному рейтингу придатності становить 0.4;

– break_len: Втілює важливість тривалості перерв між зустрічами. Воно множиться на 0.1 у загальному рейтингу придатності;

– num_meetings_max: Відображає кількість зустрічей, які потрібно врахувати в розкладі. Його вага у загальному рейтингу придатності становить 0.1;

– total_meetings_min: Представляє мінімальну кількість зустрічей в розкладі. Його вага у загальному рейтингу придатності становить 0.1.

Ці критерії досить динамічні, оскільки ми можемо змінити їх значення або взагалі додати нові критерії з додатковими ваговими коефіцієнтами, щоб врахувати різні обставини або вимоги щодо планування.

Ідея полягає в тому, щоб забезпечити можливість налаштування критеріїв для гнучкого графіка для будь-якого конкретного користувача або ситуації.

На 4-му етапі та 5-му етапах починається сама реалізація генетичного алгоритму з використанням функції пристосованості зі змінними вагами. На першому етапі генетичного алгоритму проводиться ініціалізація вхідних даних. Змінні «meetings», «priorities» і «user_criteria» визначають вхідні дані:

– «meetings» – це список зустрічей з визначеними характеристиками (тип, учасник, дата, час, тривалість, обов'язковість тощо);

– «priorities» встановлює пріоритети для типів подій;

– «user_criteria» визначає ваги для критеріїв, що використовуються для обчислення балу пристосованості.

Після цього на другому етапі проходить створення методу для фітнес-функції. Фітнес-функція «eval_schedule» визначає якість розкладу,

згенерованого генетичним алгоритмом. Обчислює різні показники і складає бал їх пристосованості.

Також в 2-й етап входить створення функції «can_meetings_be_scheduled» ,яка використовується для перевірки, відповідають чи зустрічі допустимим стандартам def can_meetings_be_scheduled(individual).

На 3-му етапі створюється головна функція, яка виконує генетичний алгоритм. А на 4-му та 5-му проходить сама робота генетичного алгоритму. Спочатку створюється початкова популяція.

```
pop = toolbox.population(n=10000)
```

Наступним етапом є вимірювання початкової пристосованості всієї популяції за допомогою фітнес – функції

```
fitnesses = list(map(toolbox.evaluate, pop))
for ind, fit in zip(pop, fitnesses):
    ind.fitness.values = fit
```

Після цього прописується виконання генетичного алгоритму до заданої стоп-умови:

```
fits = [ind.fitness.values[0] for ind in pop]
g = 0
gmax = 100  # Maximum number of generations
maxfit = [max(fits)]  # save maximum fitness in each generation
while max(fits) < 100 and g < gmax: ...
```

Після цього проходить процедура обрання наступної генерації:

```
offspring = toolbox.select(pop, len(pop))
offspring = list(map(toolbox.clone, offspring))
```

Та застосування генетичних операцій до потомства:

```
for child1, child2 in zip(offspring[::2], offspring[1::2]):
    if random.random() < 0.5:
        toolbox.mate(child1, child2)
        del child1.fitness.values
        del child2.fitness.values
```

```
        for mutant in offspring:
            if random.random() < 0.2:
                toolbox.mutate(mutant)
                del mutant.fitness.values
```

Далі проходить оцінювання потомства:

```
        invalid_ind = [ind for ind in offspring if not
ind.fitness.valid]
        fitnesses = map(toolbox.evaluate, invalid_ind)
        for ind, fit in zip(invalid_ind, fitnesses):
            ind.fitness.values = fit
```

Та процес оновлення популяції.

Також в цій функції прописано повторення кроків до заданої умови зупинки.

Увесь код детально наведено у додатку А.

3.5 Результати експериментальної перевірки

В рамках реалізації програмного прототипу було створено процес обробки даних та реалізація генетичного алгоритму. Для експериментальної перевірки було обрано мінімальний набір вхідних даних – набір зустрічей. А також реалізоване формування csv файла з фінальним розкладом – як результати роботи системи і фінальний результат для користувача. Приклад вхідних даних для експерименту можна побачити у таблиці 3.3.

Треба зазначити що у вхідних даних були запропоновані різні варіанти конфліктів між зустрічами. Наприклад, 2-го червня об 11-й ранку у користувача є потреба у двох робочих зустрічах, але одна з них обов’язкова, а інша ні.

Таблиця 3.3 – Вхідні дані для перевірки

{«type»: «work», «person»: «Maria», «mandatory»: True, «date»: «2023-06-01», «time»: 10, «duration»: 2.0}
{«type»: «work», «person»: «Anastasiia», «mandatory»: True, «date»: «2023-06-02», «time»: 11, «duration»: 1.5}
{«type»: «work», «person»: «Nastya», «mandatory»: False, «date»: «2023-06-02», «time»: 11, «duration»: 1.5}
{«type»: «family», «person»: «Mother», «date»: «2023-06-03», «time»: 14, «duration»: 3.0}
{«type»: «dinner», «person»: «Friend Ira», «date»: «2023-06-03», «time»: 14, «duration»: 2.0}
{«type»: «hobby», «person»: «Dance Teacher», «date»: «2023-06-04», «time»: 10, «duration»: 2.0}
{'type': 'hobby', 'person': 'Teacher', 'date': '2023-05-01', 'time': 14, 'duration': 1, 'location': 'Park'},
{'type': 'hobby', 'person': 'Ivan', 'date': '2023-05-02', 'time': 10, 'duration': 1.5, 'location': 'Cafe'}
{'type': 'work', 'person': 'Manager', 'date': '2023-05-03', 'time': 9, 'duration': 2, 'mandatory': False}
'type': 'gym', 'person': 'Coach', 'date': '2023-05-05', 'time': 19, 'duration': 1, 'location': 'Gym'},
{'type': 'hobby', 'person': 'Coach streching', 'date': '2023-05-04', 'time': 16, 'duration': 1.5, 'location': 'Park'},
{'type': 'dinner', 'person': 'Lyidmila', 'date': '2023-05-05', 'time': 13, 'duration': 1, 'location': 'Restaurant'},
{'type': 'work', 'person': 'Maria', 'date': '2023-05-01', 'time': 10, 'duration': 1, 'mandatory': True},
{'type': 'work', 'person': 'Tester', 'date': '2023-05-03', 'time': 15, 'duration': 2, 'mandatory': True},
{'type': 'work', 'person': 'Designer', 'date': '2023-05-05', 'time': 9, 'duration': 1.5, 'mandatory': False}.

За критеріями заданими користувачем при такій ситуації пріоритет звичайно має обов’язкова робоча зустріч. Інший приклад пов’язаний з пріоритетами. Можна побачити, що користувач 3-го червня об 14-й годині має дві зустрічі. Одна відноситься до типу зустрічей з друзями, інша відноситься до типу зустрічей з родиною. В такому випадку ми очікуємо, що система залишить в розкладі той тип зустрічі, який має вищій пріоритет – заданий користувачем. Пріоритети задані користувачем: priorities = {«work»: 1, «hobby»: 3, «gym»: 4, «dinner»: 2, «family»: 5}.

Так як в наших вхідних даних родина має вище пріоритет, ніж друзі тому система залишити зустріч з родиною. Результат роботи можна побачити на рисунку 3.6.

	type	person	date	time	duration	location	mandatory
2	hobby	Teacher	5/1/2023	14	1	Park	
3	work	Maria	5/1/2023	10	1		True
4							
5	hobby	Ivan	5/2/2023	10	1.5	Cafe	
6							
7	work	Manager	5/3/2023	9	2		False
8	work	Tester	5/3/2023	15	2		True
9							
10	hobby	Coach streching	5/4/2023	16	1.5	Park	
11							
12	gym	Coach	5/5/2023	19	1	Gym	
13	dinner	Lyidmila	5/5/2023	13	1	Restaurant	
14	work	Designer	5/5/2023	9	1.5		False
15							
16	work	Maria	6/1/2023	10	2		True
17							
18	work	Anastasiia	6/2/2023	11	1.5		True
19							
20	family	Mother	6/3/2023	14	3		
21							
22	hobby	Dance Teacher	6/4/2023	10	2		

Рисунок 3.6 – Результат роботи програми

Тобто на рисунку 3.6 можна побачити, що розклад загалом вдалося скласти і зустрічі були обрані правильно опираючись на критерії користувача.

3.5.1 Обговорення ефективності генетичного алгоритму

Під час дослідження було вирішено також побудувати графіки, що відображають динаміку зміни балів пристосованості (fitness scores) у процесі еволюції генетичного алгоритму. Ми обираємо кількість поколінь (`gen`), середню оцінку популяції (`avg`), найнижчу оцінку (`min_`) і найвищу оцінку (`max_`), які будуть відкладені на графіках. На осі x зображена кількість поколінь (`gen`), на осі y - середнє значення балу пристосованості (`avg`), а також максимальний і мінімальний бали пристосованості (`min_` і `max_`). Також треба зазначити що графіки будували на одних і тих самих даних. Але з різними параметрами генетичного алгоритму (рисунок 3.7).

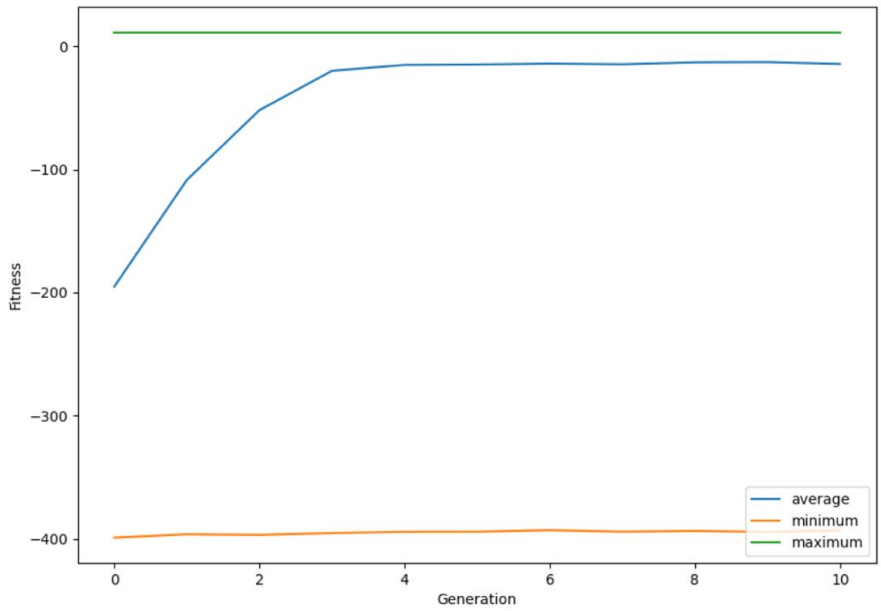

Рисунок 3.7 – Графік динаміки зміни пристосованості

Графік представлений на рисунку 3.7 будувався з такими параметрами генетичного алгороитму. algorithms.eaSimple(pop, toolbox, cxpb=0.7, mutpb=0.3, ngen=100, stats=stats, halloffame=hof, verbose=True) pop = toolbox.population(n=10000)

У випадку рисунка 3.8 маємо такі самі параметри як у 3.7, але з розміром популяції в 100 особин. Графік динаміки зміни пристосованості наведено на рисунку 3.9.

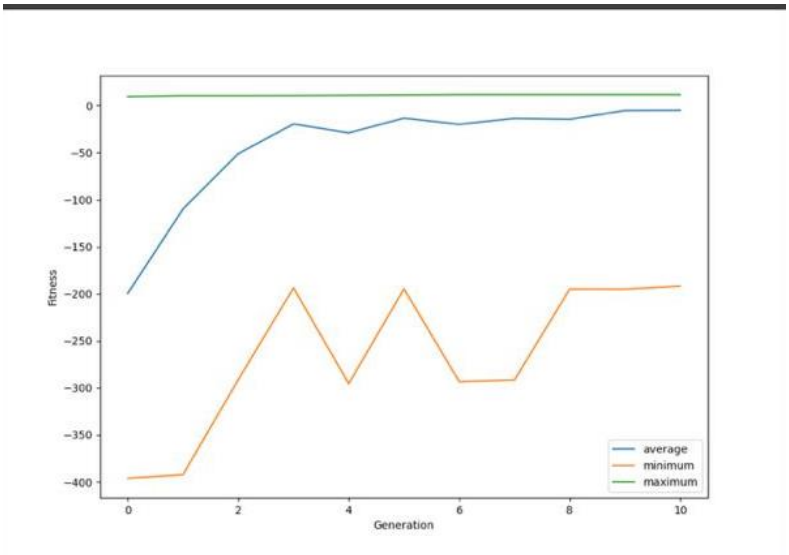

Рисунок 3.8 – Графік динаміки зміни пристосованості

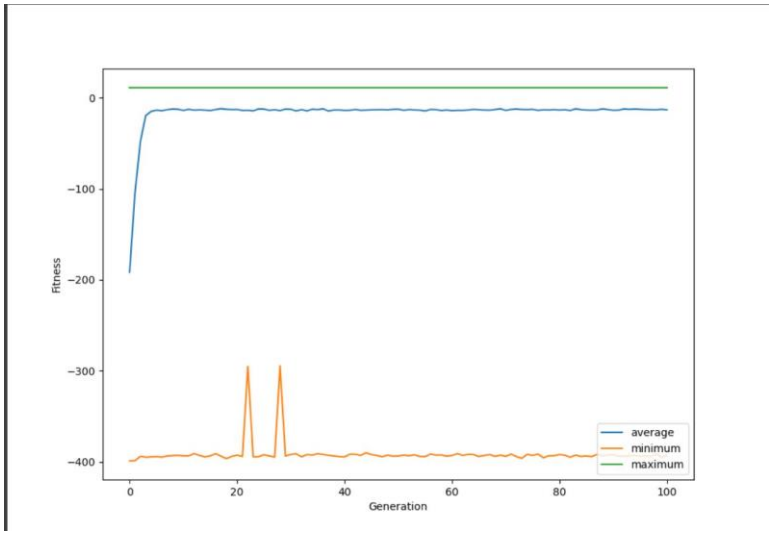

Рисунок 3.9 – Графік динаміки зміни пристосованості

У випадку рисунку 3.9 маємо 100 поколінь, розмір популяції 10 000. Основуючись на фактичних результатах роботи та на тому що функція пристосованості в один момент набувала потрібного значення можна стверджувати, що алгоритм на базовому рівні може виконувати свої функції та надавати розклад.

3.5.2 Порівняння результатів з існуючими рішеннями

У підході з Interactive Genetic Algorithms розглядається взаємодія користувача під час оптимізації з використанням генетичних алгоритмів. Цей підхід досить доречний, однак він вимагає активної участі користувача протягом усього циклу оптимізації. На відміну від цього, наш метод вимагає визначення вимог та критеріїв користувача лише один раз – перед виконанням алгоритму. Це зменшує навантаження на користувача, забезпечує більшу зручність надання послуг.

Multiobjective Genetic Algorithm розглядає багатоцільове планування календаря. Результати включають колекцію рішень, які представляють кращий компроміс між різними цілями. Наш підхід збігається з цим методом, однак ми внесли до алгоритму більшу гнучкість за допомогою вагових коефіцієнтів для критеріїв. Результатом роботи нашого методу є один сформований розклад.

В підході з Adaptive Fitness Function пропонується метод адаптивної функції придатності, що пристосовується до процесу еволюції алгоритму. Наш підхід відрізняється тим, що ми керуємо пристосованістю за вимогами користувача, а не за ступенем просування оптимізації. При використання адаптивної функції пристосованості вона стає менш стабільною з ростом числа поколінь. У порівнянні з цим, наш підхід передбачає уведення стабільних вагових коефіцієнтів, що забезпечують більш сталу роботу алгоритму. Структуроване порівняння представлене у таблиці 3.4.

Таблиця 3.4 – Порівняння існуючих методів та дослідного генетичного алгоритму

Назва методу	Порівняння
Генетичний алгоритм: функція пристосованості з динамічними вагами	Підхід зменшує потребу в постійній взаємодії користувача в процесі роботи. Результатом є одне рішення. Керує пристосованістю за вимогами користувача. Вміння вирішувати конфліктуючі події на основі пріоритетів зустрічей
Багатоцільові генетичні алгоритми (MOGA)	Результатом є не одне рішення, а набір рішень, які користувачу потрібно обрати.
Інтерактивні генетичні алгоритми (IGA)	Під час роботи користувачу потрібно весь час взаємодіяти з методом.
Генетичні алгоритми з адаптивними функціями пристосованості	Керує пристосованістю за ступенем просування оптимізації.

3.6 Аналіз перспектив и варіанти застосування

Розглянемо можливості подальшого використання та вдосконалення пропонованого підходу.

Важливою проекцією стане розробка мобільного додатка на основі цього алгоритму. Це забезпечить можливість кожному користувачеві легко і швидко отримувати оптимальний розклад безпосередньо на своєму смартфоні.

Іншим застосуванням може бути інтеграція нашого алгоритму з популярними месенджерами. Такий підхід дозволить користувачам

використовувати розклад, який генерує наш алгоритм, прямо в месенджері, в якому вони обговорюють деталі майбутньої зустрічі.

Робота над покращенням класифікатора для типів зустрічей за допомогою вибірки українською мовою. Якісніша вибірка допоможе покращити точність класифікації зустрічей, сприяючи створенню більш точних розкладів.

Також варто провести додаткові дослідження для вдосконалення роботи самого генетичного алгоритму. Це може стосуватися як вдосконалення ефективності самого алгоритму, так і пошуку нових способів для оптимізації роботи функції пристосованості.

ВИСНОВКИ

В ході виконання кваліфікаційної роботи було проведено аналіз існуючих методів та підходів до календарного планування. На основі аналізу підходів до календарного планування обґрунтовано актуальність розробки удосконаленого еволюційного методу що використовує текстові дані та враховує комбінації ваг показників, пріоритетів зустрічей а також обмежень на кількість та час зустрічей, що свідчить про важливість розробки комбінованого підходу.

Для вирішення задачі було обрано використовувати функції пристосованості на основі комбінацій ваг, рейтингів та обмежень щодо плану, які встановлює користувач для побудови персонального календарного планування. Цей підхід дозволяє системі краще розуміти і враховувати індивідуальні вподобання користувача при плануванні, що робить розклад більш пристосованим до конкретного користувача.

Розроблено еволюційний метод персоналізованого календарного планування з урахуванням інформації з повідомлень про зустрічі в чаті та з можливістю врахування потреб користувача. На першому етапі роботи метод використовує дані повідомлень з месенджерів. Він використовує автоматизовані підходи для виявлення та аналізу інформації про заплановані зустрічі в текстових повідомленнях. Додатково для обробки тексту було розроблено класифікатор, який визначає тип зустрічі, основуючись на вмісті повідомлення. Основною складовою системи став генетичний алгоритм, розроблений для оптимізації розкладу з використанням вищезгаданої комбінації динамічних ваг, критеріїв та рейтингів функцій пристосованості. Метод був імплементований мовою програмування Python. Додатково було порівняно створений метод з альтернативними підходами.

Подальші перспективи удосконалення та імплементації даного методу пов’язані з розробкою мобільного додатка, що дозволить користувачам

легко отримувати оптимальний розклад безпосередньо на своєму смартфоні. Також покращена версія класифікатора для типів зустрічей з розширеним набором даних українською мовою.

Результати наукових досліджені доповідались на 28-му міжнародному молодіжному форумі «Радіоелектроніка та молодь У XXI столітті», у м. Харків.

Таким чином, всі завдання дослідницької роботи були виконані і мета роботи досягнена – розроблено та імплементовано підхід до особистого календарного планування, що враховує індивідуальні потреби користувача на основі обробки текстів повідомлень в месенджерах.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. Jing Xu. Improved Genetic Algorithm to Solve the Scheduling Problem of College English Courses. *Complexity.* 2021. Т. 2021, № 3. P. 1–11.

2. Ping Tang, Gordon K. Lee. An Adaptive Fitness Function for Evolutionary Algorithms Using Heuristics and Prediction. *Automation Congress*, 2006. WAC '06. World. August 2006.

3. Ahmed Suleiman. Use of Genetic Algorithms in Planning and Scheduling Projects Well. URL: https://www.linkedin.com/pulse/use-genetic-algorithms-planning-scheduling-projects-well-suleiman/ (дата звернення: 15.04.2024).

4. Optimizing TV Programs Scheduling Using Genetic Algorithms in Python. URL: https://towardsdatascience.com/optimizing-tv-programs-scheduling-using-genetic-algorithms-in-python-361fab402e75 (дата звернення: 20.04.2024).

5. Tay Tze Hao. Classroom Scheduling Using Heuristics and Genetic Algorithm. URL: https://taytzehao.medium.com/classroom-scheduling-using-heuristics-and-genetic-algorithm-b7279ecdd74c (дата звернення: 15.04.2024).

6. Бурцева А. Використання генетичних алгоритмів для оптимізації індивідуального планування зустрічей. 28-й Міжнародний молодіжний форум «Радіоелектроніка та молодь У XXI столітті», м. Харків, 16–18 квіт. 2024 р. 2024. С. 37.

7. Jurafsky D., Martin J. H. Speech and Language Processing (3rd ed.). URL: https://web.stanford.edu/~jurafsky/slp3/ (дата звернення: 23.04.2024).

8. Bird S., Klein E., Loper E. Natural Language Processing with Python: Analyzing Text with the Natural Language Toolkit. 2009.

9. Manning C. D., Schütze H. Foundations of Statistical Natural Language Processing. 1999.

10. Indurkhya N., Damerau F. J. (Eds.). Handbook of Natural Language Processing. 2010.

11. Russell S., Norvig P. Artificial Intelligence: A Modern Approach. 2016.

12. Kelleher J. D., Mac Namee B., D'Arcy A. Fundamentals of Machine Learning for Predictive Data Analytics: Algorithms, Worked Examples, and Case Studies. 2015.

13. James G., Witten D., Hastie T., Tibshirani R. An Introduction to Statistical Learning: with Applications in R. 2013.

14. The use of Genetic Algorithms in the planning and scheduling of projects, as well as the comparison of other optimization techniques and limitations. URL: https://www.linkedin.com/pulse/use-genetic-algorithms-planning-scheduling-projects-well-suleiman/ (дата звернення: 15.04.2024).

15. Eiben A. E., Smith J. E. Introduction to Evolutionary Computing. Springer-Verlag Berlin Heidelberg, 2015. 287 p.

16. Booch G. Genetic Algorithms. 2019. URL: https://www.ibm.com/developerworks/library/cc-genetic-algorithms/index.html (дата звернення: 22.04.2024).

17. Jovanovic S., de Carvalho A. An introduction to Genetic Algorithms. 2019. URL: https://towardsdatascience.com/introduction-to-genetic-algorithms-including-example-code-e396e98d8bf3 (дата звернення: 28.04.2024).

18. Ng R. Understanding Genetic Algorithms in the AI World. 2020. URL: https://towardsdatascience.com/understanding-genetic-algorithms-in-the-ai-world-9b9642533e29 (дата звернення: 29.04.2024).

19. PyImageSearch. Genetic algorithms for optimization. 2019. URL: https://www.pyimagesearch.com/2019/04/22/genetic-algorithms-for-optimization/ (дата звернення: 04.04.2024).

20. Kochenderfer M. J., Wheeler T. A. Algorithms for Optimization. The MIT Press, 2019. 520 P.

21. Simon D. Evolutionary Optimization Algorithms. Wiley, 2019. 767 P.

22. Study of Genetic Algorithm Optimization in Automatic Class Timetabling based on University Academic Regulation. IOP Conference Series: *Journal of Physics: Conference Series*. 2021. Vol. 1950. URL: https://iopscience.iop.org/article/10.1088/1742-6596/1950/1/012067 (дата звернення: 24.04.2024).

23. The use of Genetic Algorithms in the planning and scheduling of projects, as well as the comparison of other optimization techniques and limitations. LinkedIn, 2021. URL: https://www.linkedin.com/pulse/use-genetic-algorithms-planning-scheduling-projects-well-suleiman/ (дата звернення: 24.04.2024).

24. Optimization of student personal course schedules with Evolutionary Algorithms. ResearchGate.net. 2013. URL: https://www.researchgate.net/publication/256706269_Optimization_of_student_personal_course_schedules_with_Evolutionary_Algorithms (дата звернення: 24.04.2024).

25. Best AI Scheduling Assistants. Unite.ai, 2021. URL: https://www.unite.ai/best-ai-scheduling-assistants/ (дата звернення: 24.04.2024).

26. Genetic Algorithm for Scheduling and Scheduling Problems. Data-Driven Investor, 2019. URL: https://www.datadriveninvestor.com/2019/04/17/genetic-algorithm-for-scheduling-and-optimisation-problems/ (дата звернення: 25.04.2024).

27. Resource-Constrained Project Scheduling: Models, Algorithms, Extensions and Applications. Google Books, 2018. URL: https://books.google.com.ua/books/about/Resource_Constrained_Project_Scheduling.html?id=X_-mwsnOt-AC&redir_esc=y (дата звернення: 25.04.2024).

28. Чалий, С.Ф., & Дементʼєв, А.М., Динамічне представлення процесу прийняття рішень при побудові пояснень в адаптивній інтелектуальній системі. 2024.