НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ «КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ ІМЕНІ ІГОРЯ СІКОРСЬКОГО»

Навчально-науковий інститут атомної та теплової енергетики

Кафедра цифрових технологій в енергетиці

«До захисту допущено»
Завідувач кафедри
_____ Наталія АУШЕВА
“___”_____2026 р.

Дипломна робота
на здобуття ступеня бакалавр

За освітньою програмою “Цифрові технології в енергетиці”

Спеціальності 122 “Комп’ютерні науки”

на тему: “Система розпізнавання заданих об’єктів і пошуку аналогів ”

Виконав: студент 4 курсу, групи ТР-з21
Царенко Антон Олександрович _____
(прізвище, ім’я, по батькові) (підпис)

Керівник доцент, к.т.н, Світлана ШАПОВАЛОВА____ _____
(посада, науковий ступінь, вчене звання, ім’я, ПРІЗВИЩЕ) (підпис)

Рецензент доцент каф. ТАЕ, к.т.н., Олександр СІРИЙ___ _____
(посада, науковий ступінь, вчене звання, ім’я, ПРІЗВИЩЕ) (підпис)

Н.контроль провідний інженер Людмила Кузьміна ___
(посада, ім’я, ПРІЗВИЩЕ) (підпис)

Засвідчую, що у цій дипломній роботі немає запозичень з праць інших авторів без відповідних посилань.
Студент _____

Київ - 2026

НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
“КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ
імені ІГОРЯ СІКОРСЬКОГО”

НАВЧАЛЬНО-НАУКОВИЙ ІНСТИТУТ АТОМНОЇ ТА ТЕПЛОВОЇ ЕНЕРГЕТИКИ

Кафедра ЦИФРОВИХ ТЕХНОЛОГІЙ В ЕНЕРГЕТИЦІ

Рівень вищої освіти – перший (бакалаврський)
спеціальність 122 “Комп’ютерні науки”
Освітньо-професійна програма “Цифрові технології в енергетиці”

ЗАТВЕРДЖУЮ
Завідувач кафедри ЦТЕ
_____ Наталія АУШЕВА
(підпис)
“___” _____2026 р.

ЗАВДАННЯ
на дипломну роботу студенту

_____ Царенку Антону Олександровичу _____
(прізвище, ім’я, по батькові)

1. Тема роботи “ Система розпізнавання заданих об’єктів і пошуку аналогів”
Науковий керівник Шаповалова Світлана Ігорівна, к.т.н, доцент
(прізвище, ім’я, по батькові, науковий ступінь, вчене звання)

затверджені наказом по університету від “ 28 ” травня 2026 року № 1993

2. Термін подання студентом роботи 08.06.2026

3. Вихідні дані до роботи мова програмування Java, фреймворк Spring Boot, СУБД PostgreSQL, середовище розробки Intellij IDEA,

4. Перелік питань, які потрібно розробити _____

1) провести аналіз предметної області пошуку зниклих домашніх тварин, сучасних методів розпізнавання об'єктів і пошуку схожих зображень;
2) виконати аналіз існуючих програмних засобів і систем пошуку домашніх тварин та обґрунтувати вибір технологій реалізації програмної системи;
3) обґрунтувати вибір методів формування векторних представлень зображень та алгоритмів пошуку подібності;

4) розробити архітектуру програмної системи пошуку домашніх тварин на основі аналізу зображень;
5) реалізувати програмну систему пошуку домашніх тварин у вигляді Telegram-бота;
6) провести тестування розробленої системи та оцінити результати її роботи;
5. Орієнтований перелік ілюстративного матеріалу схема архітектури проєкту, ER-діаграма бази даних, діаграма активностей основних сценаріїв, скріншоти роботи користувачів з Телеграм-ботом;
6. Дата видачі завдання 15.09.2025**.**

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів виконання дипломної роботи	Термін виконання етапів роботи	Примітка
1.	Вибір теми роботи	14.11.2025	
2.	Аналіз методів та засобів розв'язання задачі	22.12.2025	
3.	Розробка архітектури та загальної структури системи	05.02.2026	
4.	Розробка окремих підсистем	13.03.2026	
5.	Програмна реалізація системи	17.04.2026	
6.	Оформлення пояснювальної записки	06.05.2026	
7.	Захист програмного забезпечення	13.05.2026	
8.	Передзахист	03.06.2026	
9.	Захист	15.06.2026	

Студент _____ Антон ЦАРЕНКО
(підпис) (прізвище та ініціали)

Керівник _____ Світлана ШАПОВАЛОВА
(підпис) (прізвище та ініціали)

АНОТАЦІЯ

Дипломна робота виконана на 132 сторінках, містить 18 рисунків, 10 таблиць, 4 додатків та 35 джерел в переліку посилань.

Метою роботи: розробка програмної системи пошуку зниклих домашніх тварин із використанням методів комп'ютерного зору та алгоритмів пошуку схожих зображень.

У роботі проведено аналіз предметної області пошуку втрачених домашніх тварин, сучасних методів комп'ютерного зору, підходів до формування векторних представлень зображень та алгоритмів пошуку подібності. Розглянуто існуючі системи пошуку домашніх тварин та обґрунтовано вибір підходу Content-Based Image Retrieval для реалізації системи.

У рамках роботи розроблено архітектуру програмної системи та реалізовано Telegram-бота для взаємодії з користувачами. Для формування векторних представлень зображень використано попередньо навчену модель ResNet50 у форматі ONNX. Пошук схожих зображень реалізовано на основі косинусної міри подібності. Для зберігання даних використано систему керування базами даних PostgreSQL. Програмна система реалізована мовою Java із використанням фреймворку Spring Boot.

Проведено тестування розробленої системи на наборі даних Oxford-IIIT Pet Dataset, який містить фотографії котів і собак різних порід. Отримані результати підтвердили працездатність реалізованого підходу та можливість використання методів комп'ютерного зору для пошуку візуально схожих зображень домашніх тварин.

Практичне значення роботи – створення програмної системи, яка автоматизує пошук схожих фотографій тварин та може бути використана як основа для сервісів підтримки пошуку загублених домашніх тварин.

Ключові слова: КОМП'ЮТЕРНИЙ ЗІР, ПОШУК СХОЖИХ ЗОБРАЖЕНЬ, CONTENT-BASED IMAGE RETRIEVAL, EMBEDDINGS, RESNET50, COSINE SIMILARITY, TELEGRAM-БОТ, ПОШУК ВТРАЧЕНИХ ТВАРИН.

ABSTRACT

This thesis comprises 132 pages and contains 18 figures, 10 tables, 4 appendices and 35 references in the bibliography.

The aim of this thesis is to develop a software system for locating missing pets using computer vision techniques and image similarity search algorithms.

The thesis analyses the subject areas of lost-pet search, modern computer vision methods, approaches to generating image vector representations, and similarity search algorithms. Existing pet search systems are reviewed, and the choice of the Content-Based Image Retrieval approach for implementing the system is justified.

As part of the work, the software system architecture was developed, and a Telegram bot was implemented for user interaction. A pre-trained ResNet50 model in ONNX format was used to generate image vector representations. A similar image search is implemented based on the cosine similarity measure. The PostgreSQL database management system was used for data storage. The software system was implemented in Java using the Spring Boot framework.

The developed system was tested on the Oxford-IIIT Pet Dataset, which contains photographs of cats and dogs of various breeds. The results obtained confirmed the effectiveness of the implemented approach and the possibility of using computer vision methods to search for visually similar images of domestic animals.

The practical significance of this work lies in the creation of a software system that automates the search for similar photographs of animals and can be used as a basis for services supporting the search for lost pets.

Keywords: COMPUTER VISION, SIMILAR IMAGE SEARCH, CONTENT-BASED IMAGE RETRIEVAL, EMBEDDINGS, RESNET50, COSINE SIMILARITY, TELEGRAM BOT, LOST PET SEARCH.

ЗМІСТ

ПЕРЕЛІК СКОРОЧЕНЬ, УМОВНИХ ПОЗНАЧЕНЬ, ТЕРМІНІВ

AI – Artificial Intelligence (штучний інтелект)

API – Application Programming Interface (інтерфейс прикладного програмування)

CBIR – Content-Based Image Retrieval (пошук зображень за їхнім вмістом)

CNN – Convolutional Neural Network (згорткова нейронна мережа)

DTO – Data Transfer Object

JPA – Java Persistence API

JSON – JavaScript Object Notation

ONNX – Open Neural Network Exchange

ORM – Object Relational Mapping

RDBMS – Relational Database Management System

REST – Representational State Transfer

SQL – Structured Query Language

UML – Unified Modeling Language

ВСТУП

У сучасному світі обсяги цифрових даних постійно зростають. Через це дедалі актуальними стають методи обробки і аналізу інформації. Зокрема необхідними стають методи роботи з зображеннями. Однією з актуальних задач є пошук об’єктів за їх візуальними характеристиками. Методи для вирішення цієї задачі стають актуальними в різних сферах життя сучасної людини. Подібні підходи використовуються в системах розпізнавання обличь, медичній діагностиці, електронній комерції, моніторингу об’єктів та пошуку аналогічних зображень у великих наборах даних.

Пошук загублених домашніх тварин – актуальна задача для власників котів і собак. У науковій літературі взаємодія людини з домашньою твариною розглядається в межах концепції Human-Animal Bond. Тобто тварина переходить із категорії власність у категорію компаньйона або навіть члена родини. Тому втрата домашньої тварини має не лише організаційні, матеріальні, а й емоційними наслідками [1].

Інформацію про зниклу тварину можна зазвичай знайти або в оголошеннях на спеціальних дошках оголошень або у популярних месенджерах, таких як Вайбер або Телеграм. Для цього навіть створюються відповідні спільноти. Пошук зазвичай виглядає як нескінченний перегляд оголошень на різних ресурсах. Водночас на успіх пошуку впливають такі чинники, як швидкість поширення інформації, кількості та активності користувачів та можливості своєчасно знайти потрібне повідомлення.

Проблема втрати домашніх тварин не є поодинокою. За результатами дослідження Weiss та співавторів близько 15 % власників собак і котів хоча б один раз стикалися з втратою домашньої тварини [2]. Це свідчить про практичну значущість задачі та необхідність пошуку шляхів підвищення ефективності існуючих підходів.

У таких умовах фотографія часто стає основним джерелом інформації про тварину. На відміну від текстового опису, зображення містить велику кількість

візуальних ознак: особливості забарвлення, форму морди, будову тіла, характерні плями або інші індивідуальні риси. Саме за цими ознаками людина зазвичай намагається встановити, чи належать дві фотографії одній і тій самій тварині.

Використання фотографій має і певні обмеження. Зображення одного й того самого об’єкта можуть суттєво відрізнятися залежно від умов освітлення, відстані до камери, ракурсу зйомки або якості зображення. Крім того, тварина може бути сфотографована у русі, частково перекрита іншими об’єктами або займати лише невелику частину кадру. Через це просте порівняння зображень на рівні окремих пікселів не дає задовільних результатів.

Сучасні згорткові нейронні мережі здатні виділяти складні ознаки об’єктів і формувати їх компактні числові представлення. Такі представлення можуть використовуватися для оцінювання ступеня подібності між зображеннями та пошуку фотографій зі схожими візуальними характеристиками.

Одним із комплексних підходів є використання системи пошуку за вмістом зображень (Content-Based Image Retrieval, CBIR). В той час, як текстовий пошук спирається на характеристики, які можна витягнути з тексту повідомлень, такі системи орієнтуються виключно на зображення. Саме через це в системи пошуку за вмістом зображень доцільно використовувати алгоритми комп’ютерного зору.

Виходячи з усього вищесказаного можна зробити висновок, що задача розробки програмної системи пошуку домашніх тварин з використанням алгоритмів комп’ютерного зору є актуальною.

Метою роботи є розробка та дослідження програмної системи пошуку зниклих домашніх тварин із використанням методів комп'ютерного зору та алгоритмів пошуку схожих зображень.

Для досягнення поставленої мети необхідно вирішити шість завдань:

1. Провести аналіз предметної області пошуку зниклих домашніх тварин, а також сучасних методів розпізнавання об'єктів і пошуку схожих зображень.
2. Виконати аналіз існуючих програмних засобів і систем пошуку домашніх тварин та обґрунтувати вибір технологій реалізації програмної системи.

3. Обґрунтувати вибір методів формування векторних представлень зображень та алгоритмів пошуку подібності.
4. Розробити архітектуру програмної системи пошуку домашніх тварин на основі аналізу зображень.
5. Реалізувати програмну систему пошуку домашніх тварин у вигляді Telegram-бота.
6. Провести тестування розробленої системи та оцінити результати її роботи.

Об'єктом дослідження є процес пошуку зниклих домашніх тварин на основі аналізу зображень.

Предметом дослідження є методи комп'ютерного зору, формування векторних представлень та алгоритми пошуку схожих зображень, що використовуються для пошуку домашніх тварин за фотографіями.

Практичне значення одержаних результатів полягає у створенні програмної системи пошуку домашніх тварин, реалізованої у вигляді Telegram-бота, яка дозволяє автоматизувати пошук схожих фотографій і спростити взаємодію користувачів із системою.

Дипломна робота складається зі вступу, чотирьох розділів, висновків, списку використаних джерел і додатків. Робота містить 132 сторінки, 18 рисунків, 10 таблиць, 3 додатки та 35 найменувань у списку використаних джерел.

РОЗДІЛ 1. АНАЛІЗ ТА ПОСТАНОВКА ЗАДАЧІ РОЗПІЗНАВАННЯ ОБ'ЄКТІВ

Даний розділ присвячений аналізу задачі розпізнавання об'єктів і пошуку схожих зображень у контексті створ1ення програмної системи пошуку загублених домашніх тварин. Для реалізації подібної системи необхідно враховувати особливості роботи із цифровими зображеннями. Також варто зважати на специфіку задач комп'ютерного зору та сучасні підходи до аналізу візуальної інформації.

Класичні інформаційні системи використовують пошук за текстовими даними. Натомість основним джерелом для задачі пошуку зображень є фотографія. Такий підхід створює ряд додаткових складнощів. Зокрема, різні зображення одного об'єкта зазвичай мають різні умови освітлення, фон, масштаб або ракурс зйомки. Також фотографії можуть бути розмитими, містити шуми або сторонні об'єкти. Як наслідок, пряме попіксельне порівняння зображень зазвичай є неефективним [3].

Для вирішення задач пошуку зображень активно застосовують методи комп'ютерного зору і машинного навчання. Зокрема, увага приділяється впровадженню підходів, пов'язаних із пошуком схожих зображень, виділенням ознак і формуванням векторних представлень зображень [4]. Це дозволяє аналізувати не окремі пікселі фотографії, а зображені на них об'єкти і отримувати характеристики цих об'єктів. Такі підходи дозволяють уникнути проблем, пов'язаних зі змінами освітлення, фону або ракурсу.

У сучасних системах пошуку зображень одним з найпоширеніших підходів є використання згорткових нейронних мереж [5]. Подібні нейронні мережі дозволяють автоматично виділяти ознаки із зображень. Для пошуку зображень на основі їх візуального вмісту разом з нейронними мережами зазвичай використовується підхід Content-Based Image Retrieval. Для порівняння векторних

представлень використовуються спеціальні метрики подібності, зокрема косинусна міра подібності.

1.1 Постановка задачі та вимоги до системи

Основною задачею даної роботи є пошук схожих зображень тварин за фотографією користувача. Для її розв'язання система повинна отримувати вхідне зображення, формувати його векторне представлення, виконувати порівняння з наявними зображеннями у базі даних та визначати фотографії, які задовольняють встановлений критерій подібності.

Особливістю даної задачі є використання зображень як основного джерела інформації. На відміну від традиційних інформаційних систем, пошук виконується не за текстовими запитами, а на основі візуальних характеристик фотографій. Через це виникає необхідність використання методів комп'ютерного зору, формування векторних представлень та оцінювання подібності між зображеннями.

Додатковою складністю є те, що фотографії однієї й тієї самої тварини можуть відрізнятися освітленням, масштабом, ракурсом зйомки та якістю зображення. Тому система повинна забезпечувати коректне порівняння фотографій навіть за наявності подібних відмінностей.

У даному підрозділі виконується формалізація задачі пошуку схожих зображень, розглядаються вхідні та вихідні дані системи, пороговий критерій пошуку, а також функціональні та нефункціональні вимоги до програмного забезпечення.

1.1.1 Формалізація задачі

Нехай задано множину зображень бази даних D системи:

$$D = \{I_1, I_2, \dots, I_n\},$$

де I_i — окреме зображення тварини.

Користувач надає вхідне зображення [Q]. Воно використовується як пошуковий запит. Для виконання пошуку кожне зображення перетворюється у векторне представлення

$$E(I_i) = (e_1, e_2, \dots, e_m),$$

де m — розмірність простору ознак.

Аналогічно для вхідного зображення формується векторні представлення E(Q). Після цього для кожного зображення бази даних обчислюється значення подібності s_i:

$$s_i = S(E(Q), E(I_i)),$$

де S — функція оцінювання подібності між векторними представленнями. Результатом роботи системи є множина зображень R:

$$R = \{I_i \in D \mid s_i \geq T\},$$

де (T) — порогове значення подібності.

Отже, формалізована задача зводиться до отримання множини зображень, у яких значення подібності s з вхідним зображенням Q не нижче за порогове значення подібності T.

1.1.2 Вхідні та вихідні дані

Для реалізації пошуку схожих зображень необхідно визначити вхідні та вихідні дані програмної системи. Їх опис дозволяє формалізувати процес обробки інформації та встановити межі функціонування системи.

Вхідними даними є цифрові фотографії домашніх тварин, які надсилає користувач.

Після завантаження фотографії система виконує її попередню обробку та формує векторне представлення. Це представлення використовується для подальшого порівняння із представленнями зображень, що зберігаються у базі даних. Окрім вхідного зображення, система використовує набір фотографій тварин, доданих до бази даних раніше.

Основні дані, які використовуються під час роботи системи, наведено в таблиці 1.1

Таблиця 1.1 – Вхідні та вихідні дані системи

Тип даних	Опис
Вхідні дані	Фотографія тварини
Дані бази даних	Набір фотографій тварин, що зберігаються у системі
Проміжні дані	Векторне представлення зображення
Вихідні дані	Список потенційно схожих фотографій
Додаткові дані	Значення подібності для знайдених результатів

Після обчислення коефіцієнтів подібності система відбирає фотографії, для яких отримане значення перевищує встановлений поріг. Для знайдених зображень обраховується значення подібності з вхідною фотографією.

У результаті вхідними даними системи є фотографії домашніх тварин, а вихідними — список знайдених фотографій, відсортованих за рівнем візуальної подібності.

1.1.3 Функціональні вимоги

Функціональні вимоги визначають основні можливості системи та описують дії, які вона повинна виконувати. Дана система призначена для пошуку схожих зображень тварин за фотографіями користувача.

Система повинна забезпечувати:

- завантаження фотографій тварин користувачем;
- попередню обробку зображень;
- формування векторних представлень за допомогою CNN-моделі;
- збереження фотографій та векторних представлень у базі даних;
- створення пошуків на основі завантажених фотографій;
- перегляд активних пошуків користувача;
- видалення пошуків користувача;
- виконання пошуку за подібністю;

- фільтрацію результатів за пороговим значенням подібності;
- відображення результатів пошуку користувачу.

Однією з основних функцій системи є завантаження зображень. Користувач повинен мати можливість додавати фотографії тварин до системи для подальшого пошуку або збереження у базі даних. Завантажені зображення проходять етап попередньої обробки та підготовки до аналізу.

Ще однією важливою функцією є формування векторних представлень для кожного зображення. Після завантаження фотографії система виконує аналіз зображення за допомогою нейронної мережі та створює його векторне представлення, яке надалі використовується під час пошуку.

Система повинна забезпечувати можливість створення пошуків та збереження інформації про них. Користувач може переглядати власні активні пошуки та видаляти записи, які більше не є актуальними.

Для виконання пошуку система порівнює векторне представлення вхідного зображення з векторними представлення фотографій, що зберігаються у базі даних. До результатів включаються лише ті зображення, для яких значення подібності перевищує встановлений поріг.

Користувач повинен отримувати результати пошуку у зручному вигляді. Система відображає знайдені фотографії та відповідні значення подібності, що дозволяє оцінити релевантність отриманих результатів.

Крім цього, система повинна забезпечувати збереження даних у базі. Це стосується як самих фотографій, так і векторних представлень та інформації про виконані пошуки. Також передбачається можливість тестування системи на наборі фотографій для оцінювання якості пошуку. Це дозволяє перевірити працездатність реалізованих алгоритмів та оцінити результати роботи моделі.

Таким чином, функціональні вимоги визначають основні можливості системи, необхідні для реалізації пошуку схожих зображень та взаємодії користувача із системою пошуку втрачених тварин.

1.1.4 Нефункціональні вимоги

Нефункціональні вимоги описують характеристики системи, які не пов’язані безпосередньо з окремими функціями, але впливають на якість та зручність роботи програмного забезпечення.

Система повинна забезпечувати:

- прийнятний час виконання пошуку схожих зображень;
- стабільну роботу при обробці різних типів зображень;
- достатній рівень релевантності результатів пошуку;
- можливість масштабування бази даних;
- зручну взаємодію користувача із системою;
- можливість подальшої модифікації окремих компонентів;
- переносимість між різними середовищами виконання.

Однією з важливих нефункціональних вимог є швидкодія системи. Пошук схожих зображень повинен виконуватись за прийнятний час навіть у випадку збільшення кількості фотографій у базі даних. Користувач не повинен очікувати результати пошуку протягом тривалого часу.

Ще однією важливою вимогою є стабільність роботи системи. Програма повинна коректно обробляти помилки, працювати із різними типами зображень та не завершувати роботу аварійно при отриманні некоректних вхідних даних.

Також система повинна забезпечувати достатній рівень якості пошуку та повертати результати, які є візуально подібними до вхідного зображення. Це є особливо важливим для задачі пошуку втрачених тварин, де від якості результатів залежить практична цінність системи.

Не менш важливою є і масштабованість системи. У майбутньому кількість фотографій у базі даних може збільшуватися, тому система повинна підтримувати можливість роботи з більшими обсягами даних без суттєвого погіршення продуктивності.

Окремо можна виділити вимогу до зручності використання. Інтерфейс Telegram-бота повинен бути зрозумілим для користувача, а процес створення пошуку, завантаження фотографій та перегляду результатів — простим і логічним.

Архітектура системи повинна підтримувати можливість подальшої модифікації та розвитку. Зокрема, має існувати можливість заміни моделі комп’ютерного зору, зміни способу збереження даних або розширення функціональних можливостей системи без необхідності повного перепроєктування програмного забезпечення.

Крім цього, система повинна бути переносимою між різними середовищами виконання. Використання контейнеризації дозволяє спростити розгортання програмного забезпечення та забезпечити однакові умови запуску на різних пристроях.

Таким чином, нефункціональні вимоги визначають основні характеристики якості системи, які впливають на ефективність, надійність та зручність виконання пошуку схожих зображень.

1.1.5 Обмеження та припущення

Під час розробки системи необхідно враховувати певні обмеження та припущення, які можуть впливати на результати роботи алгоритмів пошуку схожих зображень.

Одним із головних обмежень є якість вхідних зображень. У реальних умовах фотографії можуть бути розмитими, недостатньо деталізованими або містити сторонні об’єкти. Наприклад, тварина може бути сфотографована у русі або при поганому освітленні. У таких випадках якість пошуку може погіршуватися.

Ще одним обмеженням є значна схожість між різними тваринами. Наприклад, собаки однієї породи або коти схожого забарвлення можуть мати дуже подібний зовнішній вигляд. Через це система не завжди може гарантувати точне визначення конкретної тварини лише на основі фотографії.

Система орієнтована на роботу із фотографіями котів та собак. Використання інших типів тварин не розглядалося та може призводити до зниження якості результатів пошуку.

Також необхідно враховувати обмеження обчислювальних ресурсів. Формування embeddings та порівняння великої кількості зображень потребують

додаткових обчислень. При збільшенні обсягу бази даних може виникати необхідність використання більш продуктивних алгоритмів індексації та пошуку.

Одним із припущень є те, що користувач завантажує фотографії, на яких тварина є достатньо помітною. Якщо об’єкт займає дуже малу частину зображення або значною мірою перекритий іншими елементами, якість пошуку може знижуватися.

Крім цього, у рамках роботи припускається, що представлення формуються за допомогою попередньо навченої нейронної мережі. Використана модель не навчалася спеціально на наборі даних втрачених тварин, а застосовується як універсальний засіб виділення ознак зображень.

Також варто враховувати, що система не гарантує абсолютно точний результат у всіх випадках. Пошук схожих зображень є ймовірнісною задачею, тому результати можуть залежати від якості фотографій, особливостей набору даних та характеристик використаної моделі комп’ютерного зору.

Таким чином, система має ряд технічних обмежень, пов’язаних із якістю зображень, обчислювальними ресурсами та особливостями пошуку схожих зображень. Незважаючи на це, використаний підхід дозволяє автоматизувати процес пошуку фотографій та може використовуватися як допоміжний інструмент під час пошуку втрачених тварин.

1.2 Аналіз предметної області та задачі пошуку зображень

Останніми роками методи комп’ютерного зору та машинного навчання активно розвиваються та використовуються у різних сферах. Подібні технології можна зустріти у системах розпізнавання обличь, медичних інформаційних системах, системах відеоспостереження, безпілотних автомобілях, а також у пошукових сервісах. Одним із напрямів застосування комп’ютерного зору є пошук схожих зображень на основі їхніх візуальних характеристик.

На відміну від класичного пошуку за текстом, пошук за зображеннями має ряд особливостей. У текстових системах інформація представлена у вигляді слів або фраз, які можна порівнювати напряму. У випадку із фотографіями ситуація є складнішою, оскільки комп’ютерна система повинна не лише зберігати зображення, а й аналізувати їхній вміст. Для людини задача пошуку схожих фотографій зазвичай не є складною. Наприклад, людина може швидко зрозуміти, що на двох різних фотографіях зображена одна й та сама тварина. Для комп’ютерної системи така задача вже потребує використання спеціальних алгоритмів.

У рамках даної роботи розглядається задача пошуку втрачених домашніх тварин за фотографіями. Така задача має практичне значення, оскільки сьогодні пошук тварин переважно виконується вручну через перегляд оголошень у соціальних мережах або на спеціалізованих платформах. При великій кількості фотографій та повідомлень цей процес стає незручним та займає багато часу.

Для автоматизації подібного пошуку можуть використовуватись методи комп’ютерного зору, які дозволяють аналізувати фотографії, виділяти характерні ознаки об’єктів та виконувати пошук схожих зображень у базі даних. У сучасних системах для цього часто використовуються нейронні мережі та векторні представлення зображень, які дозволяють порівнювати фотографії між собою за ступенем схожості.

У даному підрозділі розглядаються особливості предметної області, проблеми пошуку втрачених тварин, специфіка використання зображень як джерела інформації, а також основні задачі комп’ютерного зору, які використовуються в роботі. Крім цього, формується постановка задачі пошуку аналогічних зображень, яка є основою для подальшої розробки програмної системи.

1.2.1 Постановка проблеми пошуку втрачених тварин

У сучасних умовах проблема пошуку втрачених домашніх тварин є досить актуальною. Особливо це помітно у великих містах, де люди часто використовують соціальні мережі, Telegram-групи або спеціалізовані сайти для публікації

оголошень про зникнення тварин. Достатньо відкрити будь-яку місцеву групу у Facebook або Telegram, щоб побачити десятки повідомлень із фотографіями собак і котів, які загубилися або були знайдені.

У більшості випадків власники публікують фотографію тварини, короткий опис і контакти для зв’язку. Проте сам процес пошуку зазвичай виконується вручну. Людина змушена переглядати велику кількість фотографій і повідомлень, намагаючись знайти схожі оголошення. Якщо оголошень небагато, це ще можливо зробити відносно швидко. Але коли мова йде про великі групи або популярні платформи, кількість публікацій може бути дуже великою. У такій ситуації пошук починає займати багато часу.

Окремою проблемою є те, що текстовий опис не завжди дозволяє точно описати зовнішній вигляд тварини. Наприклад, різні люди можуть по-різному описувати одну й ту саму собаку. Один користувач напише “рудий пес середнього розміру”, інший — “собака схожа на коргі”, а хтось взагалі вкаже лише район, де її бачили. Через це пошук лише за текстом часто є неточним.

Крім того, зовнішність тварини іноді важко описати словами. Людина може впізнати кота за формою морди, забарвленням або навіть певним “виразом” очей, але передати це текстом набагато складніше. Саме тому фотографія є основним джерелом інформації під час пошуку [1].

Також потрібно враховувати, що фотографії можуть бути зроблені в різних умовах. На одному фото тварина може бути сфотографована вдома при хорошому освітленні, а на іншому — на вулиці ввечері або здалеку. Через це навіть фотографії однієї й тієї самої тварини можуть виглядати досить по-різному. Для людини це зазвичай не є великою проблемою, але для комп’ютерної системи така варіативність ускладнює пошук.

На сьогодні для вирішення подібних задач активно використовуються методи комп’ютерного зору та машинного навчання [6]. Вони дозволяють аналізувати зображення, виділяти характерні ознаки та знаходити схожі фотографії серед великої кількості даних. Тому створення системи автоматизованого пошуку тварин за фотографіями є актуальною задачею, яка має практичне застосування.

1.2.2 Особливості використання зображень як джерела інформації

Зображення є одним із головних джерел інформації у задачах комп’ютерного зору. На відміну від тексту, фотографія містить значно більше деталей про об’єкт. Людина може буквально за кілька секунд зрозуміти, що саме зображено на фото, впізнати знайому тварину або звернути увагу на певні особливості зовнішності.

Для комп’ютера ситуація виглядає інакше. Система не “бачить” зображення так, як його бачить людина [4]. Для неї фотографія — це набір чисел, які описують колір та яскравість пікселів. Наприклад, звичайне фото з телефона може містити мільйони пікселів. Через це обробка зображень потребує достатньо великих обчислювальних ресурсів.

Ще однією особливістю є те, що зображення можуть дуже сильно відрізнятися між собою навіть у випадку, якщо на них зображений один і той самий об’єкт. Наприклад, фотографія собаки може бути зроблена зблизька або здалеку, вдень або ввечері, у квартирі або на вулиці. Іноді тварина може бути частково перекрита іншими об’єктами або рухатися під час фотографування. У реальному житті це трапляється досить часто, особливо коли люди фотографують тварин на телефон у поспіху.

Крім цього, фотографії можуть мати різну якість. Частина зображень буває розмитою або недостатньо деталізованою. Наприклад, якщо людина побачила схожу тварину на вулиці, вона може швидко зробити фото на ходу. Такі зображення часто мають погане освітлення або шуми. Для людини цього іноді достатньо, щоб впізнати тварину, але для алгоритму це вже створює додаткові труднощі.

Для того, щоб комп’ютерна система могла порівнювати фотографії між собою, необхідно виділяти певні ознаки зображення. Це можуть бути контури, текстури, кольори або інші характеристики. У класичних методах комп’ютерного зору такі ознаки задавалися вручну за допомогою спеціальних алгоритмів. Сучасні нейронні мережі дозволяють автоматично формувати складніші та інформативніші представлення зображень [7].

Саме тому для ефективної роботи із зображеннями необхідно використовувати спеціалізовані методи обробки даних. Звичайного порівняння

пікселів недостатньо, оскільки система повинна враховувати загальну схожість об’єктів навіть при змінах освітлення, ракурсу або якості фотографії.

1.2.3 Основні задачі комп’ютерного зору в контексті роботи

Комп’ютерний зір є одним із напрямів штучного інтелекту, який займається аналізом та обробкою зображень. Основна ідея полягає у тому, щоб навчити комп’ютер отримувати інформацію із фотографій або відео, подібно до того, як це робить людина. На сьогодні технології комп’ютерного зору використовуються у багатьох сферах: від медицини та систем безпеки до автомобілів із системами автопілоту.

Однією з базових задач комп’ютерного зору є класифікація зображень. У цьому випадку система визначає, до якого класу належить об’єкт на фото. Наприклад, модель може визначити, чи зображений на фотографії кіт чи собака. Такі задачі є відносно поширеними, і для них існує велика кількість готових моделей.

Проте у рамках даної роботи класифікації недостатньо. Якщо система просто визначить, що на фото зображений собака, це не допоможе знайти конкретну тварину серед великої кількості інших собак. Саме тому виникає потреба у використанні інших підходів.

Ще однією задачею є object detection — виявлення об’єктів на зображенні. Такі алгоритми дозволяють знаходити розташування об’єкта та визначати його межі. Наприклад, система може автоматично виділити собаку на фотографії навіть у випадку, якщо на фоні присутні інші об’єкти. Це може бути корисним для подальшої обробки зображення.

Важливу роль у даній роботі також відіграє виділення ознак. У процесі виділення ознак система формує певне представлення зображення, яке містить інформацію про основні характеристики об’єкта. Якщо говорити простіше, модель намагається “запам’ятати” особливості фотографії у вигляді числового вектора.

Для даної роботи найбільш важливою задачею є пошук схожих зображень. Основна ідея полягає у тому, щоб за вхідною фотографією знайти найбільш схожі

зображення у базі даних. Наприклад, якщо користувач завантажує фотографію загубленого кота, система повинна знайти схожі фотографії серед уже наявних зображень.

У сучасних системах для реалізації подібних задач широко використовуються нейронні мережі. Вони дозволяють автоматично виділяти складні ознаки та формувати векторні представлення зображень. Саме векторні представлення надалі використовуються для пошуку схожих фотографій.

Саме тому у рамках даної роботи основна увага приділяється задачам виділення ознак, формування векторних представлень та пошуку аналогічних зображень, оскільки саме вони є основою для реалізації системи пошуку втрачених тварин.

1.2.4 Постановка задачі пошуку аналогічних зображень

У даній роботі розглядається задача пошуку аналогічних зображень тварин за фотографіями. Основною метою системи є автоматичне знаходження найбільш схожих зображень у базі даних на основі вхідного фото, яке завантажує користувач. Вхідними даними системи є цифрове зображення тварини. Це може бути фотографія собаки або кота, зроблена на телефон або інший пристрій. База даних системи містить набір фотографій, які були попередньо додані користувачами. Результатом роботи системи є список найбільш схожих зображень, відсортованих за рівнем подібності.

На перший погляд задача може виглядати простою, але на практиці вона має багато складнощів. Наприклад, дві фотографії однієї й тієї самої тварини можуть суттєво відрізнятися між собою. На одному фото собака може бути сфотографована зблизька вдома, а на іншому — на вулиці або з іншого ракурсу. Іноді навіть людина не одразу може зрозуміти, що це одна й та сама тварина.

Для комп’ютерної системи така задача є ще складнішою. Через це необхідно використовувати спеціальні методи представлення зображень. У сучасних системах для цього часто використовуються векторні представлення фотографій, які формуються нейронною мережею.

Після отримання векторних представлень система порівнює вектори між собою за допомогою певної метрики подібності. У даній роботі для порівняння векторних представленнь використовується косинусна метрика подібності. Якщо відстань між векторами є невеликою, то зображення вважаються схожими.

Система виконує пошук схожих зображень на основі порогового значення подібності. Після обчислення косинусної метрики подібності результати, значення подібності яких перевищують заданий поріг, вважаються релевантними та відображаються користувачу.

Під час реалізації системи необхідно враховувати різні фактори, які можуть впливати на якість пошуку. До них належать освітлення, якість фотографії, фон, масштаб та ракурс зйомки. Саме тому задача пошуку аналогічних зображень потребує використання методів комп’ютерного зору та машинного навчання.

1.3 Аналіз існуючих програмних засобів пошуку втрачених тварин

Для пошуку загублених домашніх тварин сьогодні використовується велика кількість програмних засобів, серед яких спеціалізовані веб-платформи, мобільні додатки, соціальні мережі та месенджери. Основною метою таких систем є забезпечення швидкого поширення інформації про зникнення або знаходження тварини та створення умов для взаємодії між власниками, волонтерами й іншими користувачами. Завдяки розвитку інформаційних технологій процес пошуку тварин значною мірою перейшов із друкованих оголошень та локальних спільнот до цифрового середовища.

Найбільш поширеним підходом є використання спеціалізованих платформ, які працюють за принципом дошки оголошень. У таких системах користувач може створити повідомлення про зникнення або знаходження тварини, додати фотографію, опис зовнішніх ознак, місце події та контактні дані. Після публікації інформація стає доступною для інших користувачів, які можуть переглядати

оголошення та повідомляти про можливі збіги. Подібний підхід є відносно простим у реалізації та не потребує складних алгоритмів обробки даних.

Окрім спеціалізованих веб-сервісів, важливу роль у пошуку тварин відіграють соціальні мережі та месенджери. Завдяки великій кількості активних користувачів інформація про зниклу тварину може швидко поширюватися серед місцевих спільнот. У багатьох випадках саме соціальні мережі дозволяють отримати перші відомості про місцезнаходження тварини. Проте ефективність такого підходу значною мірою залежить від активності користувачів та швидкості поширення повідомлень.

Останніми роками почали з’являтися системи, які використовують додаткові технології автоматизації пошуку. До таких рішень належать сервіси на основі геолокації, баз даних мікрочіпів, а також системи комп’ютерного зору, здатні аналізувати фотографії тварин та виконувати пошук схожих зображень. Використання подібних технологій дозволяє зменшити залежність від ручного перегляду великої кількості оголошень та підвищити ефективність пошуку.

Аналіз існуючих програмних засобів є важливим етапом дослідження, оскільки дозволяє визначити сильні та слабкі сторони сучасних рішень, оцінити рівень використання технологій комп’ютерного зору та виявити проблеми, які залишаються невирішеними. У даному підрозділі розглядаються найбільш поширені програмні засоби пошуку втрачених тварин, виконується їх порівняння та визначаються основні напрями вдосконалення подібних систем.

1.3.1 Спеціалізовані платформи пошуку тварин

Одним із найбільш поширених програмних засобів для пошуку загублених домашніх тварин є спеціалізовані веб-платформи. Подібні системи призначені для публікації оголошень про зникнення або знаходження тварин та забезпечують взаємодію між власниками, волонтерами й іншими користувачами. Основною перевагою таких платформ є централізоване зберігання інформації та можливість швидкого доступу до великої кількості оголошень.

Серед українських сервісів можна виділити платформи Pet911, ЗооПошук та PetLive. Принцип роботи більшості подібних систем є схожим. Користувач створює оголошення, додає фотографію тварини, її опис, місце зникнення або знаходження, а також контактну інформацію. Після публікації оголошення стає доступним для перегляду іншими користувачами, які можуть повідомити про можливе місцезнаходження тварини.

Для прикладу, платформа PetLive реалізує класичний підхід до організації пошуку на основі дошки оголошень. Користувачі можуть переглядати список повідомлень, використовувати фільтрацію за регіоном та категорією тварини, а також переглядати фотографії та контактні дані власників. Приклад інтерфейсу сервісу наведено на рисунку 1.1.

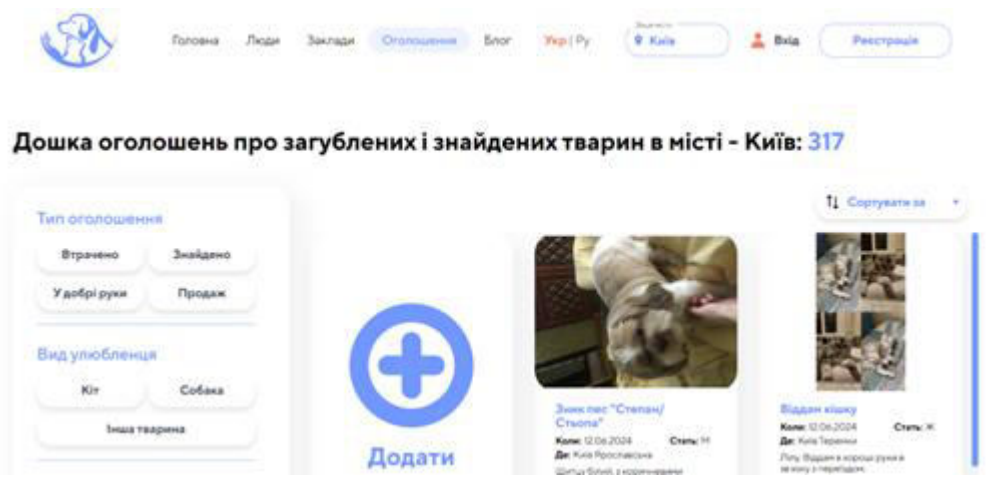

Рисунок 1.1 – Інтерфейс платформи PetLive

Основною перевагою подібних платформ є простота використання та широке охоплення аудиторії. Користувачеві достатньо створити оголошення один раз, після чого інформація стає доступною великій кількості людей. Крім цього, наявність фотографій та описів дозволяє більш точно ідентифікувати тварину порівняно з використанням лише текстових повідомлень.

Окремо можна виділити міжнародну систему Animal-ID, яка використовує інший підхід до ідентифікації тварин. Основою роботи сервісу є база даних зареєстрованих домашніх тварин та інформація про встановлені мікрочіпи. У випадку знаходження тварини спеціаліст або ветеринарна клініка може зчитати дані

мікрочіпа та отримати інформацію про власника. Такий підхід забезпечує високу точність ідентифікації, проте його ефективність залежить від попередньої реєстрації тварини та наявності відповідного обладнання.

Сучасним напрямом розвитку спеціалізованих платформ є використання технологій комп'ютерного зору. Прикладом такого підходу є сервіс Petsi, який використовує алгоритми аналізу зображень для пошуку схожих фотографій тварин. Користувач завантажує фотографію, після чого система виконує її обробку та намагається знайти можливі збіги серед наявних записів. Подібний підхід дозволяє частково автоматизувати процес пошуку та зменшити необхідність ручного перегляду великої кількості оголошень.

Спеціалізовані платформи є важливим інструментом пошуку втрачених тварин та забезпечують централізоване зберігання інформації про зникнення і знаходження домашніх улюбленців. Водночас більшість із них все ще базується на ручному пошуку та перегляді оголошень, що створює передумови для використання більш автоматизованих підходів на основі методів комп'ютерного зору.

1.3.2 Використання соціальних мереж для пошуку тварин

Окрім спеціалізованих платформ, важливу роль у пошуку загублених домашніх тварин відіграють соціальні мережі та месенджери. Завдяки великій кількості активних користувачів такі сервіси дозволяють швидко поширювати інформацію про зникнення або знаходження тварини серед мешканців певного регіону. У багатьох випадках саме соціальні мережі є першим інструментом, до якого звертаються власники після втрати домашнього улюбленця.

Найбільш поширеними засобами пошуку є тематичні групи у Facebook, Telegram-канали, Viber-спільноти та локальні онлайн-спільноти. Користувачі публікують повідомлення із фотографією тварини, описом її зовнішності, місцем зникнення та контактними даними. Інші учасники спільноти можуть поширювати інформацію або повідомляти про можливі збіги.

Основною перевагою соціальних мереж є широке охоплення аудиторії та висока швидкість поширення інформації. Одне повідомлення може бути переглянуте та поширене великою кількістю користувачів протягом короткого часу. Особливо ефективним такий підхід є у великих містах, де існують спеціалізовані локальні групи, присвячені пошуку тварин.

Прикладом використання месенджерів для пошуку домашніх тварин є сервіс WIMP (Where Is My Pet?). Дана система інтегрована з популярними платформами обміну повідомленнями та дозволяє користувачам надсилати інформацію про загублену тварину безпосередньо через чат-бота. Після отримання повідомлення система використовує дані про місцезнаходження користувача та поширює інформацію серед інших учасників, які знаходяться поблизу району пошуку.

Перевагою такого підходу є простота використання та відсутність необхідності переходити на окремий веб-сайт або встановлювати спеціалізований додаток. Користувач взаємодіє із системою через звичний інтерфейс месенджера, що спрощує процес створення повідомлень та отримання сповіщень.

Водночас використання соціальних мереж має низку суттєвих обмежень. Насамперед пошук залишається переважно ручним. Користувачам необхідно самостійно переглядати велику кількість повідомлень та фотографій, намагаючись знайти можливі збіги. У популярних групах кількість нових публікацій може бути дуже великою, через що важливі повідомлення швидко втрачають актуальність та перестають бути помітними для інших користувачів.

Ще однією проблемою є залежність ефективності пошуку від активності спільноти. Навіть якісно оформлене повідомлення не гарантує успішного результату, якщо його побачить недостатня кількість людей. Крім того, пошук часто базується на текстових описах, які можуть бути неповними або суб'єктивними, що ускладнює ідентифікацію тварини.

Таким чином, соціальні мережі та месенджери є важливим інструментом пошуку втрачених тварин завдяки швидкому поширенню інформації та широкому охопленню аудиторії. Проте більшість таких рішень не використовує

автоматизований аналіз зображень і значною мірою покладається на ручний перегляд повідомлень користувачами.

1.3.3 Порівняння існуючих програмних рішень

Розглянуті програмні засоби використовують різні підходи до пошуку загублених домашніх тварин. Частина систем орієнтована на публікацію оголошень та взаємодію між користувачами, інші використовують геолокацію або спеціалізовані бази даних. Окремі сучасні рішення додатково застосовують методи комп'ютерного зору для аналізу фотографій тварин. Для більш детального порівняння основних можливостей розглянутих систем складено таблицю 1.2.

Таблиця 1.2 – Порівняння існуючих програмних рішень пошуку втрачених тварин

Система	Фото тварини	Геолокація	Автоматичний аналіз зображень	Telegram / месенджери
PetLive	+	-	-	-
Pet911	+	-	-	-
Animal-ID	+	-	-	-
WIMP	+	+	-	+
Petsi	+	-	+	-
Запропонована система	+	-	+	+

Як видно з таблиці, більшість існуючих платформ забезпечує можливість публікації фотографій тварин та зберігання додаткової інформації про них. Проте основний механізм пошуку у таких системах базується на ручному перегляді оголошень користувачами. Це збільшує витрати часу на пошук та знижує ефективність роботи при великій кількості повідомлень.

Сервіс WIMP частково автоматизує процес поширення інформації завдяки використанню геолокації та месенджерів. Однак пошук можливих збігів все одно

здійснюється користувачами вручну. Система не виконує автоматичного порівняння фотографій та не використовує методи комп’ютерного зору.

Найближчим за функціональністю до запропонованого рішення є сервіс Petsi, який використовує технології аналізу зображень для пошуку схожих фотографій тварин. Проте такі системи поки що мають обмежене поширення та недостатньо велику базу користувачів, що може негативно впливати на результативність пошуку.

Запропонована в рамках даної роботи система поєднує переваги декількох підходів. З одного боку, вона використовує Telegram-бота як зручний інтерфейс взаємодії з користувачем, а з іншого — застосовує методи комп’ютерного зору та пошук схожих зображень на основі векторних представлень. Це дозволяє автоматизувати процес порівняння фотографій та зменшити залежність від ручного перегляду великої кількості оголошень.

1.3.4 Недоліки існуючих програмних засобів

Проведений аналіз існуючих програмних засобів пошуку втрачених тварин показав, що більшість сучасних рішень базується на використанні оголошень, соціальних мереж або спеціалізованих баз даних. Незважаючи на значне поширення таких систем, вони мають ряд недоліків, які можуть негативно впливати на ефективність пошуку.

Однією з основних проблем є необхідність ручного перегляду великої кількості оголошень та фотографій. У більшості платформ користувачеві необхідно самостійно переглядати записи та намагатися визначити можливі збіги між загубленими та знайденими тваринами. При великій кількості повідомлень такий процес потребує значних витрат часу та не гарантує успішного результату.

Ще одним недоліком є залежність пошуку від якості текстового опису. Користувачі можуть по-різному описувати одну й ту саму тварину, використовувати різні характеристики або не вказувати важливі ознаки. Через це пошук за текстовими даними часто є недостатньо ефективним, особливо коли йдеться про тварин схожого забарвлення або породи.

Суттєвим обмеженням більшості існуючих систем є відсутність автоматизованого аналізу фотографій. Незважаючи на те, що майже всі платформи дозволяють завантажувати зображення, у більшості випадків вони використовуються лише як допоміжний елемент для користувача. Автоматичне порівняння фотографій та пошук схожих зображень реалізовані лише в окремих сучасних рішеннях і поки що не набули широкого поширення.

Окремою проблемою є залежність ефективності пошуку від кількості активних користувачів. Соціальні мережі та месенджери можуть забезпечувати швидке поширення інформації, проте результативність такого підходу безпосередньо залежить від активності спільноти та кількості переглядів конкретного повідомлення. У результаті важлива інформація може залишитися непоміченою або швидко втратити актуальність серед великої кількості нових публікацій.

Крім цього, частина спеціалізованих рішень використовує додаткові технології ідентифікації, наприклад мікрочіпування. Подібний підхід забезпечує високу точність встановлення власника тварини, проте його використання можливе лише за умови попередньої реєстрації тварини та наявності спеціального обладнання для зчитування інформації.

Таким чином, аналіз існуючих програмних засобів показав, що більшість сучасних систем не забезпечує повноцінного автоматизованого пошуку на основі візуальної подібності зображень. Це підтверджує доцільність розробки програмної системи, яка використовуватиме методи комп’ютерного зору, формування векторних представлень та пошук схожих зображень для підвищення ефективності пошуку втрачених тварин.

РОЗДІЛ 2. МЕТОДИ ТА АЛГОРИТМИ ВИРІШЕННЯ ЗАДАЧІ ПОШУКУ СХОЖИХ ЗОБРАЖЕНЬ

Основною проблемою пошуку аналогічних зображень є побудова такого представлення зображення, яке дозволяє порівнювати об'єкти за їхніми візуальними характеристиками. Безпосереднє використання піксельних значень для цього малопридатне, оскільки навіть незначні зміни умов зйомки можуть суттєво змінювати вигляд фотографії. Унаслідок цього зображення одного й того самого об'єкта нерідко мають більші відмінності на рівні пікселів, ніж зображення різних об'єктів.

У задачах комп'ютерного зору для опису зображень використовуються ознаки, які відображають їх зміст та можуть бути використані для подальшого порівняння. Історично для цього застосовувалися різні підходи: від ручного конструювання дескрипторів до використання глибоких нейронних мереж, здатних автоматично виділяти інформативні характеристики об'єктів. Вибір способу формування ознак безпосередньо впливає на якість подальшого пошуку та точність визначення подібності між зображеннями.

У роботі використовується підхід, заснований на формуванні векторних представлень зображень за допомогою згорткової нейронної мережі та подальшому порівнянні отриманих векторів у просторі ознак. Для обґрунтування такого рішення необхідно розглянути особливості згорткових нейронних мереж, принципи формування векторних представлень та методи оцінювання подібності між векторними представленнями зображень.

2.1 Аналіз підходів до розпізнавання об’єктів

Для вирішення задач розпізнавання об’єктів та пошуку схожих зображень використовується широкий спектр методів комп’ютерного зору та машинного навчання. Розвиток даної галузі відбувався поступово: від класичних алгоритмів

виділення ознак до сучасних моделей глибокого навчання, здатних автоматично формувати складні візуальні представлення зображень. Кожен із підходів має власні переваги, недоліки та області застосування.

Перші системи розпізнавання зображень базувалися на ручному виділенні ознак. Для цього використовувалися алгоритми, які аналізували контури, кути, текстури та інші локальні характеристики зображення. Подібні методи стали основою багатьох класичних підходів комп'ютерного зору та тривалий час застосовувалися у задачах пошуку і порівняння зображень.

Подальший розвиток технологій привів до використання методів машинного навчання, які дозволили автоматизувати процес класифікації об'єктів на основі підготовлених ознак. Такі підходи забезпечили підвищення точності розпізнавання та кращу адаптацію систем до різних наборів даних.

На сучасному етапі найбільшого поширення набули методи глибокого навчання, зокрема згорткові нейронні мережі (CNN). Вони дозволяють автоматично виділяти інформативні ознаки без необхідності ручного налаштування алгоритмів та забезпечують високу якість обробки зображень у задачах комп'ютерного зору.

Окремим напрямом розвитку є системи пошуку схожих зображень, побудовані за принципом Content-Based Image Retrieval (CBIR) [9]. У таких системах для представлення зображень використовуються векторні описи, сформовані нейронними мережами, а оцінювання подібності виконується за допомогою спеціальних метрик, зокрема косинусна міра подібності.

У даному підрозділі розглядаються основні підходи до розпізнавання об'єктів та пошуку схожих зображень, їх принципи роботи, переваги та недоліки. Особлива увага приділяється методам, які можуть бути використані для реалізації системи пошуку втрачених тварин на основі фотографій.

2.1.1 Класичні методи обробки зображень (SIFT, SURF, ORB)

До появи сучасних нейронних мереж у задачах комп'ютерного зору активно використовувалися класичні методи обробки зображень. Основна ідея таких

підходів полягає у пошуку характерних точок та ознак на фотографії, які можна використовувати для подальшого порівняння зображень між собою.

Одним із найбільш відомих алгоритмів є SIFT (Scale-Invariant Feature Transform).] Даний метод дозволяє знаходити ключові точки на зображенні та формувати для них спеціальні дескриптори. Особливістю SIFT є те, що алгоритм є відносно стійким до зміни масштабу, повороту зображення та часткових змін освітлення. Через це SIFT тривалий час використовувався у задачах пошуку об’єктів та зіставлення фотографій [10].

Іншим популярним методом є SURF (Speeded Up Robust Features). Даний алгоритм був створений як швидша альтернатива SIFT. Принцип роботи SURF є схожим: система також знаходить характерні точки та формує дескриптори для подальшого порівняння. Основною перевагою SURF є вища швидкість роботи порівняно з SIFT, що була важливою для систем реального часу [11].

Ще одним поширеним методом є ORB (Oriented FAST and Rotated BRIEF). На відміну від SIFT та SURF, цей алгоритм є менш вимогливим до обчислювальних ресурсів. ORB часто використовується у задачах, де важлива швидкість роботи або існують обмеження на продуктивність. Наприклад, подібні алгоритми можуть використовуватись на мобільних пристроях або в системах реального часу. ORB поєднує швидкість роботи та низькі вимоги до ресурсів, що робить його придатним для систем із обмеженою продуктивністю [12].

Основною перевагою класичних методів є відносна простота реалізації та невисокі вимоги до обчислювальних ресурсів. Крім того, такі алгоритми не потребують великої кількості тренувальних даних, оскільки ознаки задаються вручну. Це було особливо важливо у період, коли великі набори даних та потужні графічні процесори були менш доступними.

Проте класичні методи мають і ряд недоліків. У складних умовах вони можуть працювати нестабільно. Наприклад, при значній зміні освітлення, фону або ракурсу якість пошуку може помітно погіршуватися. Крім цього, подібні алгоритми часто гірше працюють із задачами, де необхідно враховувати складні візуальні особливості об’єкта.

У задачі пошуку втрачених тварин класичні методи можуть використовуватись для базового порівняння зображень. Основним обмеженням класичних методів є використання вручну заданих ознак, які не завжди дозволяють ефективно описувати складні візуальні характеристики об’єктів.

2.1.2 Методи машинного навчання

З розвитком машинного навчання у задачах розпізнавання об’єктів почали застосовуватись підходи, які дозволяють системі навчатися на основі даних. На відміну від класичних алгоритмів, де більшість ознак задається вручну, методи машинного навчання дозволяють моделі аналізувати дані та формувати правила класифікації на основі прикладів.

У більшості випадків процес роботи таких систем складається з двох етапів. Спочатку із зображення виділяються певні ознаки, після чого ці ознаки використовуються для навчання моделі класифікації. Для цього можуть застосовуватися різні алгоритми, наприклад, SVM (Support Vector Machine), Decision Trees або Random Forest [13].

Однією з переваг методів машинного навчання є те, що моделі машинного навчання можуть адаптуватися до нових наборів зображень та враховувати статистичні закономірності у даних [14]. Наприклад, якщо модель навчена на великій кількості фотографій тварин, вона може краще визначати особливості різних об’єктів.

Такі методи певний час активно використовувалися у задачах комп’ютерного зору та дозволили суттєво покращити якість розпізнавання у порівнянні з класичними алгоритмами. Особливо це стосувалося задач класифікації зображень та пошуку об’єктів.

Проте методи машинного навчання також мають певні обмеження. У багатьох випадках якість роботи системи сильно залежить від правильності вибору ознак [15]. Якщо ознаки не містять достатньо інформації про об’єкт, точність класифікації та якість розпізнавання можуть суттєво знижуватися.

Крім цього, для навчання моделей необхідна достатня кількість даних. У задачах комп'ютерного зору це може бути проблемою, оскільки підготовка наборів даних часто потребує багато часу. Наприклад, фотографії необхідно сортувати, розмічати та перевіряти.

У задачах пошуку схожих зображень методи машинного навчання можуть використовуватись як проміжний етап між класичними алгоритмами та сучасними нейронними мережами. Проте сьогодні у більшості випадків їх поступово замінюють методи глибокого навчання, які забезпечують вищу точність та кращу здатність працювати зі складними зображеннями.

2.1.3 Методи глибокого навчання (CNN)

У сучасних системах комп'ютерного зору одним із основних підходів є використання методів глибокого навчання. Особливу роль серед них відіграють згорткові нейронні мережі — CNN (Convolutional Neural Networks) [16]. Саме такі моделі сьогодні використовуються у більшості сучасних систем розпізнавання та пошуку зображень.

Основна особливість CNN полягає у тому, що мережа автоматично формує ознаки на основі тренувальних даних. На відміну від класичних методів, нейронна мережа автоматично формує представлення об'єкта під час навчання. Це дозволяє моделі знаходити складні закономірності, які важко описати звичайними алгоритмами.

Принцип роботи CNN базується на використанні згорткових шарів, які аналізують різні області зображення. На початкових рівнях мережа може виділяти прості ознаки, наприклад, контури або текстури. На глибших рівнях модель починає розпізнавати складніші характеристики об'єктів.

Однією з головних переваг CNN є висока точність роботи у задачах комп'ютерного зору. Подібні мережі добре працюють навіть у складних умовах: при зміні освітлення, масштабу або ракурсу. Саме тому CNN активно використовуються у системах розпізнавання обличь, медичних системах, безпілотних автомобілях та пошукових сервісах.

У задачах пошуку подібності нейронні мережі часто використовуються для формування векторних представлень зображень. Після цього ці представлення можуть порівнюватися між собою за допомогою спеціальних метрик подібності. Якщо векторні представлення мають високий рівень подібності у векторному просторі, то відповідні зображення вважаються схожими.

Проте методи глибокого навчання мають і певні недоліки. Однією з головних є висока потреба в обчислювальних ресурсах. Для навчання великих моделей часто використовуються потужні графічні процесори та великі набори даних. Крім цього, процес навчання може займати значний час.

Незважаючи на це, саме CNN сьогодні є одним із найбільш ефективних підходів для задач розпізнавання та пошуку зображень. У рамках даної роботи використання згорткових нейронних мереж є доцільним через їхню здатність автоматично виділяти інформативні ознаки та забезпечувати високу якість пошуку подібності.

2.1.4 Використання попередньо навчених CNN-моделей

У сучасних задачах комп’ютерного зору часто використовуються попередньо навчені нейронні мережі. Подібні моделі вже були навчені на великих наборах даних та можуть використовуватись повторно для інших задач. Найчастіше такі моделі навчаються на наборі даних ImageNet, який містить мільйони зображень різних об’єктів, зокрема тварин [17].

Основною перевагою попередньо навчених моделей є те, що вони вже вміють виділяти важливі візуальні ознаки. Наприклад, нейронна мережа може розпізнавати контури, текстури, форми об’єктів або інші характеристики зображення. Це дозволяє уникнути повного навчання моделі з нуля та зменшити вимоги до кількості тренувальних даних. Такий підхід часто називають transfer learning [7], оскільки модель, навчена на одному наборі даних, використовується для іншої задачі.

У задачах пошуку подібності попередньо навчені CNN-моделі часто використовуються як метод отримання ознак. Тобто модель не обов’язково

застосовується для класифікації об’єктів. Замість цього вона використовується для формування векторних представлень зображень.

Такий підхід широко використовується у сучасних системах пошуку зображень. Наприклад, після проходження фотографії через CNN система отримує векторне представлення, яке описує основні характеристики об’єкта. Далі векторні представлення можуть порівнюватися між собою за допомогою метрик подібності.

Однією з найбільш відомих архітектур є ResNet (Residual Neural Network). Особливістю даної моделі є використання залишкових блоків, які дозволяють будувати глибші нейронні мережі без значного погіршення якості навчання. Архітектура ResNet продемонструвала високу ефективність у багатьох задачах комп’ютерного зору, зокрема у класифікації зображень та отримання ознак [18].

У рамках даної роботи використання попередньо навченої CNN-моделі є доцільним для задачі пошуку аналогів. Система повинна не просто визначати клас об’єкта, а й знаходити схожі фотографії тварин серед великої кількості зображень. Через це важливо отримувати якісні векторні представлення, які містять достатньо інформації про візуальні особливості тварини.

Крім цього, використання попередньо навченої моделі дозволяє застосовувати вже готові візуальні ознаки без необхідності повного навчання нейронної мережі. Це особливо важливо у задачах, де кількість доступних даних може бути обмеженою.

У сучасних системах подібний підхід використовується досить часто [19]. Наприклад, багато пошукових сервісів також базуються на векторних ознаках, сформованих попередньо навченими CNN-моделями.

2.1.5 Контентно-орієнтований пошук зображень (CBIR)

Одним із основних підходів до пошуку схожих зображень є контентно-орієнтований пошук зображень — CBIR (Content-Based Image Retrieval). Головна ідея полягає у тому, що система виконує пошук не за текстовим описом, а безпосередньо за візуальним вмістом фотографії [20].

У класичних пошукових системах зображення зазвичай описувалися за допомогою тегів, назв або ключових слів. Наприклад, фотографія собаки могла мати опис “рудий пес” або “лабрадор”. Проте такий підхід має ряд недоліків, оскільки текстовий опис не завжди точно передає зовнішній вигляд об’єкта. Крім цього, різні користувачі можуть використовувати різні формулювання для опису однієї й тієї самої фотографії.

CBIR дозволяє зменшити залежність від текстових описів та виконувати пошук безпосередньо за візуальними характеристиками зображення, а не його текстовим описом [21]. Для цього із фотографії виділяються певні характеристики, наприклад, кольори, текстури, контури або локальні ознаки. Після цього система порівнює отримані характеристики різних зображень і визначає рівень їхньої схожості.

На рисунку 2.1 показано основні етапи роботи CBIR-системи. Після завантаження запитного зображення система виконує попередню обробку та виділення ознак. Отримане векторне представлення порівнюється з ознаками зображень, які зберігаються у базі даних. Після обчислення рівня подібності користувачу відображаються найбільш релевантні результати пошуку.

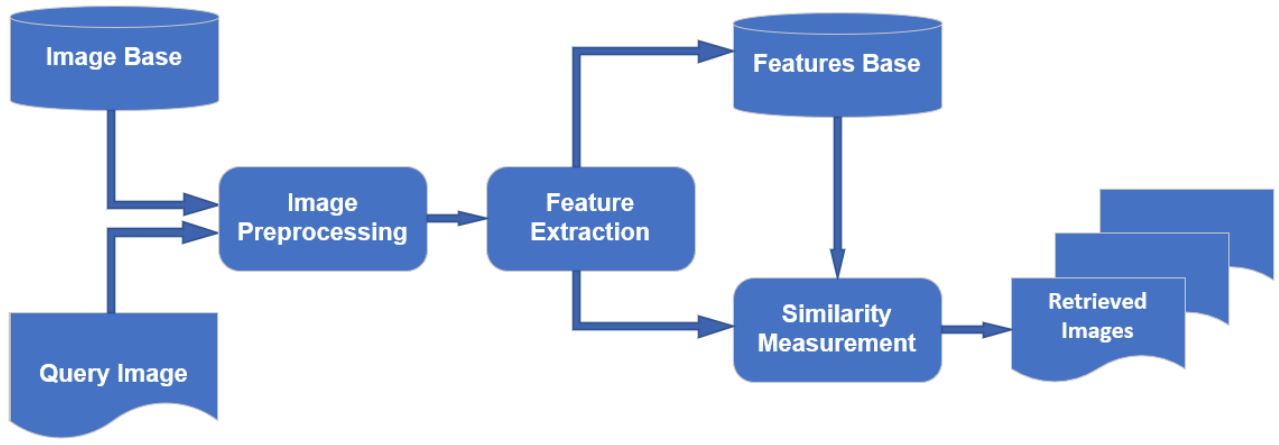

Рисунок 2.1 - Схема CBIR

Наприклад, якщо користувач завантажує фотографію собаки, система може знайти інші зображення, які мають схожі кольори або форму об’єкта. Саме тому

CBIR став одним із перших підходів, який дозволив реалізувати пошук за фотографіями.

Однією з переваг CBIR є можливість роботи без текстових описів. Це особливо важливо у випадках, коли текстова інформація є неповною або відсутньою. Крім цього, подібні системи дозволяють автоматизувати процес пошуку та зменшити залежність від ручного сортування даних.

Проте класичні системи CBIR мають і певні недоліки. У багатьох випадках вони недостатньо добре враховують “семантичний зміст” зображення. Наприклад, дві фотографії можуть містити одну й ту саму тварину, але через різне освітлення або фон система може визначити їх як несхожі.

Подібна проблема часто називається semantic gap — різницею між візуальними характеристиками зображення та його реальним змістом.

Через це сучасні системи пошуку аналогів все частіше використовують нейронні мережі та векторні представлення, які дозволяють формувати більш складне та інформативне представлення зображення.

Система, розроблена у рамках даної роботи, також належить до класу CBIR-систем, оскільки виконує пошук на основі візуальної схожості фотографій.

2.1.6 Використання векторних представлень

У сучасних системах комп’ютерного зору для пошуку схожих зображень широко використовуються векторні представлення даних. Основна ідея полягає у тому, щоб перетворити зображення на набір чисел, який описує його основні характеристики [21].

Після проходження через нейронну мережу кожне зображення отримує власне векторне представлення. Такий вектор може містити десятки, сотні або навіть тисячі числових значень. При цьому схожі зображення повинні мати схожі векторні представлення та розташовуватися близько одне до одного у векторному просторі [22].

Якщо говорити простіше, векторне представлення можна уявити як певний “цифровий опис” фотографії. Наприклад, дві фотографії одного й того самого кота

повинні мати близькі векторні представлення навіть у випадку, якщо вони були зроблені в різних умовах (рисунок 2.2).

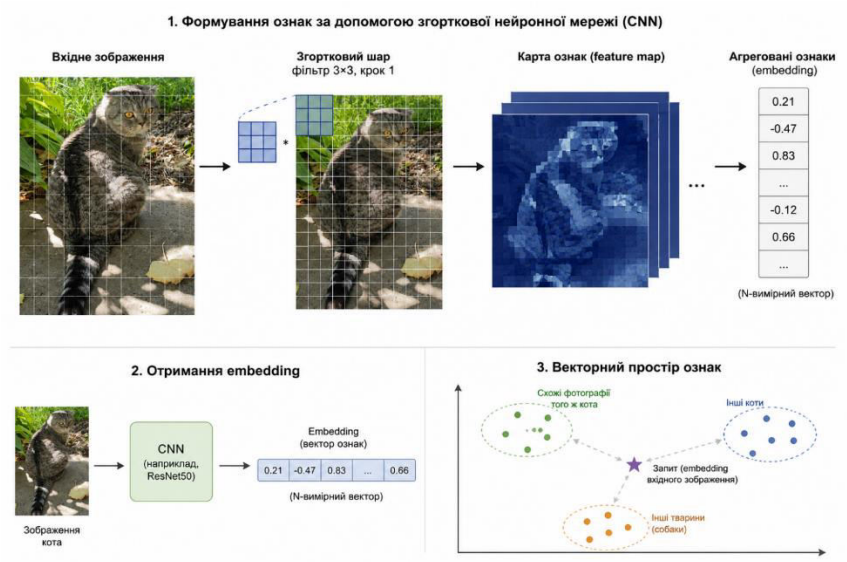

Рисунок 2.2. Формування векторного представлення зображення

Однією з головних переваг векторних представлень є здатність працювати зі складними візуальними ознаками. На відміну від класичних алгоритмів, нейронна мережа може автоматично враховувати форму об’єкта, текстуру шерсті, розташування елементів та інші характеристики.

Крім цього, векторні представлення дозволяють забезпечити ефективне порівняння великої кількості зображень. Після того, як векторні представлення усіх зображень збережені у базі даних, система може швидко порівнювати вектори між собою за допомогою спеціальних метрик подібності.

Сьогодні векторне представлення активно використовуються у багатьох сучасних системах комп’ютерного зору. Подібні підходи застосовуються у системах розпізнавання обличь, візуальному пошуку та пошуку фотографій у великих базах зображень.

У рамках даної роботи використання векторних представлень є доцільним, оскільки такий підхід дозволяє ефективно порівнювати фотографії тварин навіть при зміні освітлення, ракурсу або якості зображення.

2.1.7 Пошук схожих зображень у векторному просторі

У сучасних системах пошуку аналогів, власне пошук зазвичай виконується у векторному просторі. Основна ідея такого підходу полягає у порівнянні векторних представлень між собою та визначенні рівня їхньої подібності.

Після формування векторних представлень система зберігає їх у базі даних та використовує під час подальшого пошуку. Коли користувач завантажує нове зображення, для нього також формується векторне представлення, яке надалі порівнюється з векторними представленнями інших зображень, що містяться у системі.

На відміну від прямого порівняння пікселів, пошук у векторному просторі дозволяє враховувати загальні візуальні характеристики об’єкта. Наприклад, дві фотографії одного й того самого кота можуть мати різний фон, освітлення або ракурс, проте векторне представлення таких зображень залишатимуться близькими одне до одного у векторному просторі.

Для визначення ступеня схожості між векторними представленнями використовуються спеціальні метрики подібності. Однією з найбільш поширених є косинусна метрика подібності, яка визначає подібність між двома векторами на основі кута між ними.

Також у задачах пошуку аналогів може використовуватись евклідова відстань, яка визначає відстань між двома точками у векторному просторі. Якщо відстань між векторними представленнями є невеликою, то зображення вважаються схожими.

Проте для векторних представлень у сучасних системах комп’ютерного зору частіше використовується косинусна міра подібності, оскільки дана метрика краще враховує напрямок векторів та менш залежить від абсолютних значень векторів.

Після обчислення рівня подібності система виконує фільтрацію результатів за пороговим значенням. Якщо значення подібності перевищує встановлений поріг, зображення вважаються потенційно схожими та можуть бути показані користувачу. Такий підхід дозволяє зменшити кількість нерелевантних результатів та підвищити ефективність пошуку.

Однією з переваг пошуку у векторному просторі є можливість ефективної роботи з великою кількістю зображень. Крім цього, подібний підхід добре поєднується із сучасними CNN-моделями та дозволяє будувати системи пошуку аналогів без необхідності ручного виділення ознак.

У рамках даної роботи пошук у векторному просторі використовується як основний підхід для порівняння фотографій тварин та знаходження схожих зображень у базі даних.

2.1.8 Порівняння підходів

Після аналізу основних підходів до розпізнавання об’єктів можна зробити висновок, що кожен із них має свої переваги та недоліки. Вибір конкретного методу залежить від особливостей задачі, доступних ресурсів та вимог до системи.

Класичні методи комп’ютерного зору, наприклад SIFT, SURF або ORB, є відносно простими у реалізації та не потребують великої кількості тренувальних даних. Такі алгоритми можуть добре працювати у простих задачах зіставлення зображень. Крім цього, вони мають нижчі вимоги до обчислювальних ресурсів.

Проте класичні методи мають і суттєві недоліки. Вони часто погано працюють у складних реальних умовах. Наприклад, зміна освітлення, фону або ракурсу може сильно впливати на якість пошуку. У задачі пошуку тварин це є важливою проблемою, оскільки фотографії можуть бути дуже різними.

Методи машинного навчання дозволили покращити якість розпізнавання у порівнянні з класичними алгоритмами. Такі підходи є більш гнучкими та можуть адаптуватися до нових даних. Проте у більшості випадків вони все ще сильно залежать від якості ознак, які використовуються для навчання.

Найбільш ефективними у сучасних задачах комп’ютерного зору вважаються методи глибокого навчання. CNN дозволяють автоматично виділяти складні ознаки та формувати інформативні векторні представлення. Саме тому сьогодні такі підходи широко використовуються у пошуку подібності, системах розпізнавання обличь та пошуку зображень.

Окремою перевагою CNN є здатність працювати із складними фотографіями. Наприклад, система може знаходити схожі зображення навіть при різному освітленні або частковій зміні ракурсу. Для задачі пошуку втрачених тварин це є дуже важливим.

Як видно з таблиці 2.1, різні підходи мають власні переваги та недоліки. Класичні методи комп'ютерного зору характеризуються простотою реалізації та невисокими вимогами до обчислювальних ресурсів, проте їх ефективність може суттєво знижуватися в умовах зміни освітлення, масштабу або ракурсу зйомки.

Методи машинного навчання дозволяють підвищити якість розпізнавання, однак часто потребують попереднього формування ознак. Методи глибокого навчання, зокрема згорткові нейронні мережі, забезпечують автоматичне виділення інформативних характеристик зображення та демонструють високу ефективність у сучасних задачах комп'ютерного зору.

Таблиця 2.1 – Порівняння основних підходів до розпізнавання зображень

Підхід	Переваги	Недоліки	Придатність для задачі
SIFT / SURF / ORB	Простота реалізації, невисокі вимоги до ресурсів	Чутливість до змін умов зйомки, обмежена інформативність ознак	Низька
Методи машинного навчання	Вища гнучкість порівняно з класичними методами	Потребують ручного формування ознак	Середня
CNN	Автоматичне виділення ознак, висока точність	Значні обчислювальні витрати	Висока
CBIR на основі embeddings	Пошук за візуальним вмістом, стійкість до змін освітлення та ракурсу	Залежність від якості embeddings	Дуже висока

Окрему групу складають системи Content-Based Image Retrieval, які використовують векторні представлення зображень та метрики подібності для пошуку схожих об’єктів. Подібний підхід дозволяє виконувати пошук на основі візуального вмісту зображення без необхідності використання детального текстового опису.

Як підсумок, сучасні методи глибокого навчання та підходи Content-Based Image Retrieval є перспективними для задач пошуку схожих зображень. Для визначення можливостей їх практичного використання доцільно також розглянути існуючі програмні засоби, призначені для пошуку втрачених тварин.

2.2 Загальна схема вирішення задачі

Для реалізації пошуку схожих зображень необхідно визначити послідовність етапів обробки даних, які виконуються системою від моменту отримання фотографії користувача до формування результатів пошуку. Загальна схема вирішення задачі відображає взаємозв’язок між основними етапами обробки зображень та дозволяє описати логіку роботи системи незалежно від особливостей програмної реалізації.

У рамках даної роботи пошук виконується за принципом Content-Based Image Retrieval. Основою такого підходу є аналіз візуального вмісту фотографії та подальше порівняння отриманих ознак із зображеннями, що зберігаються у базі даних. Для цього система використовує попередньо навчану згорткову нейронну мережу для формування векторних представлень та косинусну міру подібності для оцінювання схожості між фотографіями.

Як видно з рисунку 2.3 першому етапі система отримує від користувача фотографію тварини, яка використовується як пошуковий запит. Після цього виконується попередня обробка зображення, необхідна для підготовки даних до подальшого аналізу нейронною мережею.

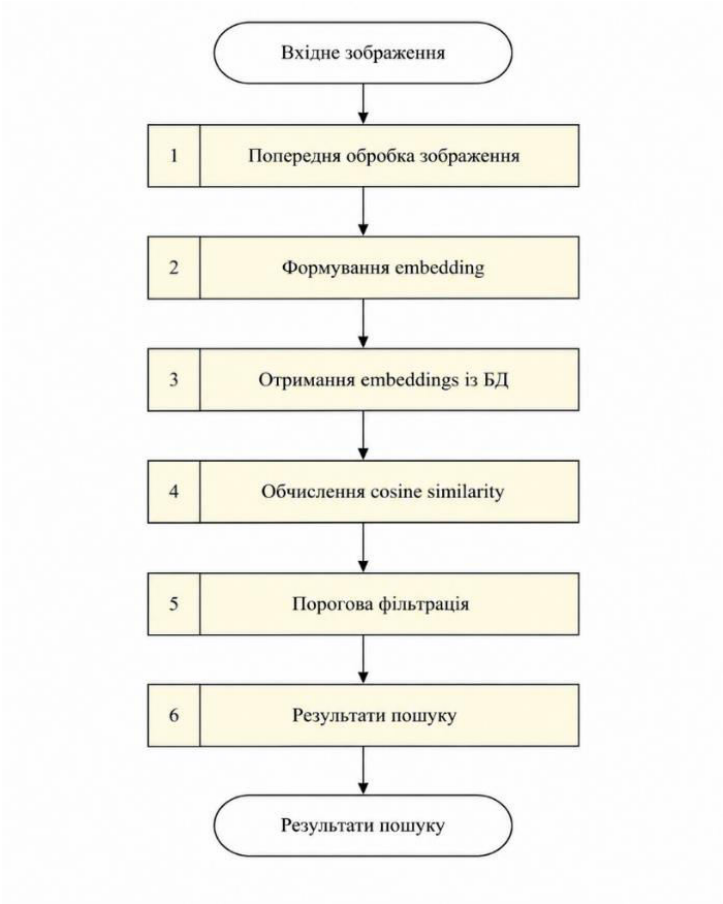

Рисунок 2.3 – Загальна схема вирішення задачі пошуку схожих зображень

Наступним етапом є формування векторного представлення фотографії. Для цього використовується попередньо навчена модель ResNet, яка дозволяє виділити інформативні ознаки зображення та представити його у вигляді числового вектора.

Паралельно система отримує векторне представлення фотографій, що зберігаються у базі даних. Після цього для кожного зображення виконується обчислення значення подібності між векторним представленням вхідної фотографії та векторним представленням відповідного запису бази даних.

На основі отриманих значень застосовується пороговий критерій відбору результатів. До результатів пошуку включаються лише ті фотографії, для яких рівень подібності перевищує встановлене порогове значення.

Завершальним етапом є формування та відображення результатів пошуку користувачу. Система повертає фотографії, які були визнані достатньо схожими до вхідного зображення, разом із відповідними значеннями подібності.

Таким чином, загальна схема вирішення задачі охоплює всі основні етапи пошуку схожих зображень: від отримання фотографії до формування результатів на основі аналізу візуальної подібності.

2.3 Метод формування ознак зображення

Одним із ключових етапів реалізації пошуку схожих зображень є формування ознак, які дозволяють описати візуальний вміст фотографії у числовому вигляді. Безпосереднє порівняння зображень за значеннями пікселів зазвичай є неефективним, оскільки навіть фотографії одного й того самого об’єкта можуть суттєво відрізнятися через зміну освітлення, масштабу, фону або ракурсу зйомки.

Для вирішення цієї проблеми використовуються спеціальні методи виділення ознак, які дозволяють перетворити зображення на більш компактне та інформативне представлення. У сучасних системах комп’ютерного зору найбільш поширеним підходом є використання нейронних мереж, здатних автоматично визначати важливі характеристики об’єктів на зображеннях.

Особливу роль у задачах аналізу зображень відіграють згорткові нейронні мережі. На відміну від класичних методів комп’ютерного зору, вони не потребують ручного формування ознак та здатні самостійно виявляти закономірності у даних під час навчання. Це дозволяє отримувати більш інформативні представлення зображень та підвищувати якість подальшого пошуку.

У рамках даної роботи для формування ознак використовується архітектура ResNet, яка належить до класу згорткових нейронних мереж. Результатом її роботи є embedding — векторне представлення зображення, що містить інформацію про основні візуальні характеристики об’єкта.

У даному підрозділі розглядаються принципи роботи штучних нейронних мереж, особливості згорткових нейронних мереж, архітектура ResNet та процес формування векторних представлень, які використовуються для подальшого пошуку схожих зображень.

2.3.1 Штучні нейронні мережі

Одним із найбільш поширених підходів до розв'язання задач розпізнавання образів, класифікації та аналізу даних є використання штучних нейронних мереж. Даний підхід належить до методів машинного навчання та широко застосовується у сучасних системах комп'ютерного зору, обробки природної мови та інтелектуального аналізу даних [24].

Ідея штучних нейронних мереж була запозичена з принципів роботи біологічної нервової системи [25]. Біологічний нейрон отримує сигнали від інших нейронів через дендрити, обробляє їх та передає результат через аксон. Аналогічний принцип використовується і в штучних нейронних мережах, де окремий нейрон отримує набір вхідних сигналів, виконує їх обробку та формує вихідне значення.

Основним елементом штучної нейронної мережі є штучний нейрон. Кожен нейрон отримує набір вхідних значень, які множаться на відповідні вагові коефіцієнти. Після цього обчислюється зважена сума сигналів, до якої може додаватися зміщення (bias). Отриманий результат передається до функції активації, яка визначає вихідне значення нейрона. З математичної точки зору робота нейрона може бути описана виразом:

$$y = f(\sum_{i=1}^{n} w_i x_i + b) \tag{2.1}$$

де y— вихідне значення нейрона;

f— функція активації;

x_i— значення і-го вхідного сигналу,

w_i— ваговий коефіцієнт, що визначає вплив відповідного входу;

b— зміщення;

Відповідно до (2.1), нейрон спочатку обчислює зважену суму вхідних значень, після чого до отриманого результату застосовується функція активації. Саме функція активації дозволяє нейронній мережі описувати складні нелінійні залежності між вхідними та вихідними даними.

Для побудови складних моделей окремі нейрони об'єднуються у шари. Зазвичай виділяють вхідний шар, один або декілька прихованих шарів та вихідний

шар. Вхідний шар отримує початкові дані, приховані шари виконують їх обробку та виділення закономірностей, а вихідний шар формує результат роботи мережі.

Як показано на рисунку 2.4, нейронна мережа складається з декількох послідовно з'єднаних шарів нейронів. Кожен нейрон одного шару пов’язаний із нейронами наступного шару через вагові коефіцієнти. У процесі навчання значення цих коефіцієнтів автоматично коригуються таким чином, щоб мінімізувати помилку між фактичним та очікуваним результатом.

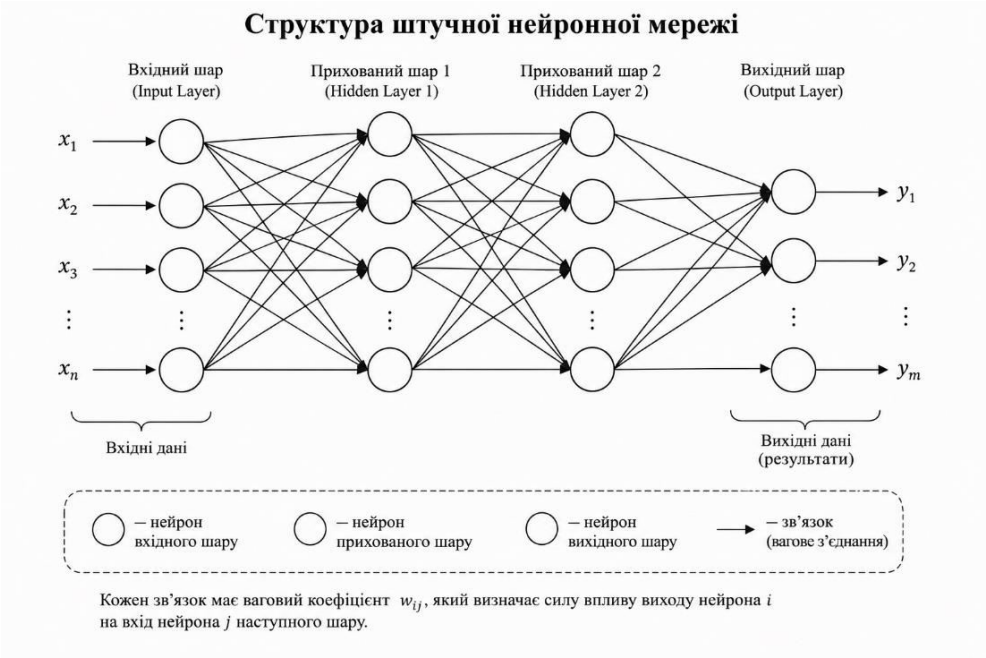

Рисунок 2.4 – Загальна структура штучної нейронної мережі

Наявність декількох прихованих шарів дозволяє мережі формувати ієрархічні представлення даних. Нижні шари зазвичай виділяють прості ознаки, тоді як верхні шари формують більш складні абстракції. Саме завдяки цій властивості нейронні мережі демонструють високу ефективність під час роботи зі складними даними, зокрема зображеннями.

Важливою перевагою нейронних мереж є здатність автоматично знаходити приховані закономірності у даних без необхідності ручного формування ознак. На відміну від класичних алгоритмів машинного навчання, де значна частина ознак

створюється розробником, нейронні мережі можуть самостійно навчатися на великих наборах даних та формувати внутрішні представлення об'єктів.

Завдяки здатності працювати з великими обсягами даних, високій точності та можливості автоматичного виділення ознак нейронні мережі стали основою більшості сучасних систем комп’ютерного зору. Саме тому в даній роботі вони розглядаються як базова технологія для побудови системи пошуку схожих зображень тварин. Подальший розвиток ідей штучних нейронних мереж привів до появи згорткових нейронних мереж, які спеціально адаптовані для обробки графічної інформації та розглядаються у наступному підрозділі.

2.3.2 Згорткові нейронні мережі

Згорткові нейронні мережі (Convolutional Neural Networks, CNN) є одним із найбільш поширених підходів до розв’язання задач комп’ютерного зору. Вони застосовуються для класифікації зображень, виявлення об’єктів, сегментації зображень та побудови систем пошуку схожих зображень. Основною особливістю CNN є здатність автоматично виділяти важливі ознаки із зображення без необхідності ручного формування дескрипторів [16].

Використання звичайних повнозв’язних нейронних мереж для аналізу зображень пов’язане з низкою труднощів. Кожен піксель зображення повинен бути представлений окремим входом мережі, що призводить до значного збільшення кількості параметрів. Наприклад, кольорове зображення розміром 224×224 пікселів містить понад 150 тисяч вхідних значень. У такому випадку навіть один прихований шар потребуватиме великої кількості вагових коефіцієнтів, що суттєво збільшує вимоги до пам’яті та обчислювальних ресурсів. Крім того, повнозв’язна мережа не враховує просторові зв’язки між сусідніми пікселями, хоча саме вони містять значну частину корисної інформації про форму та структуру об’єктів.

Для подолання зазначених недоліків були розроблені згорткові нейронні мережі. Їхня робота базується на використанні операції згортки, яка дозволяє аналізувати невеликі локальні області зображення. Замість обробки всіх пікселів одночасно використовується спеціальна матриця невеликого розміру, яка

називається ядром або фільтром згортки. Це ядро послідовно переміщується по зображенню та виконує математичні операції над локальними ділянками. Результатом є формування карти ознак, яка містить інформацію про наявність певних візуальних характеристик на зображенні.

Операція згортки може бути описана наступною формулою:

$$S(i,j) = \sum_m \sum_n I(i+m, j+n) \times K(m,n),$$

де $S(i,j)$ — значення елемента карти ознак у точці з координатами (i,j);

$I(i+m, j+n)$ — значення пікселів вхідного зображення;

$K(m,n)$ — значення елементів ядра згортки;

m, n — координати елементів ядра згортки.

За допомогою різних фільтрів мережа може виділяти різні типи ознак. На початкових рівнях зазвичай виявляються прості елементи, такі як контури, кути або текстури. На більш глибоких рівнях формуються складніші ознаки, які відповідають окремим частинам об’єктів або навіть цілим об’єктам.

Основним елементом CNN є згортковий шар (Convolution Layer). Його призначення полягає у виділенні характерних ознак із вхідного зображення шляхом застосування набору фільтрів. Кожен фільтр формує окрему карту ознак, яка відображає реакцію мережі на певний шаблон. Наявність декількох фільтрів дозволяє одночасно аналізувати різні характеристики зображення.

Після згорткового шару зазвичай використовується шар субдискретизації (Pooling Layer) [5]. Його основною задачею є зменшення розмірності карт ознак. Це дозволяє скоротити обчислювальні витрати та підвищити стійкість мережі до незначних змін масштабу, положення або орієнтації об’єкта на зображенні. Найбільш поширеним є метод максимального об’єднання (Max Pooling), при якому з локальної області вибирається найбільше значення.

Наприкінці мережі зазвичай розташовуються повнозв’язні шари (Fully Connected Layer). Вони отримують набір ознак, сформований попередніми шарами, та використовують його для прийняття остаточного рішення. У задачах класифікації результатом роботи таких шарів є визначення класу об’єкта. У задачах

пошуку схожих зображень повнозв’язні шари можуть використовуватись для формування компактного векторного представлення зображення, яке надалі застосовується для обчислення ступеня схожості між об’єктами.

Загальна структура згорткової нейронної мережі наведена на рисунку 2.5.

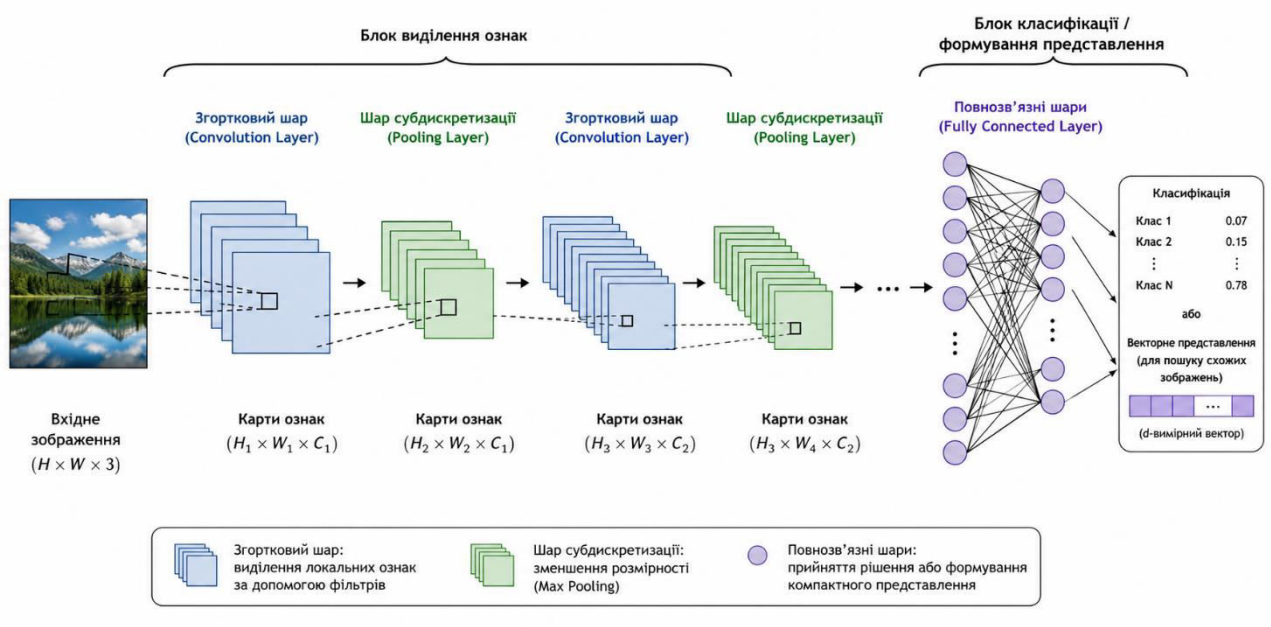

Рисунок 2.5 - Загальна структура згорткової нейронної мережі

CNN мають низку переваг у задачах комп’ютерного зору. Вони забезпечують автоматичне виділення ознак, потребують меншої кількості параметрів порівняно з повнозв’язними мережами та демонструють високу ефективність під час роботи із зображеннями великого розміру. Крім того, сучасні попередньо навчені моделі можуть використовуватись як універсальні екстрактори ознак, що значно спрощує створення систем розпізнавання об’єктів та пошуку аналогічних зображень.

У даній роботі згорткова нейронна мережа використовується для формування векторних представлень фотографій тварин. Отримані вектори надалі застосовуються для пошуку схожих зображень у базі даних та визначення найбільш ймовірних збігів.

2.3.3 Архітектура ResNet

Однією з основних тенденцій розвитку згорткових нейронних мереж є збільшення кількості шарів. Теоретично глибші мережі здатні виділяти більш складні та абстрактні ознаки, що дозволяє підвищувати точність розпізнавання об’єктів. Проте на практиці зі збільшенням глибини архітектури виникають додаткові труднощі. Однією з найбільш відомих проблем є проблема деградації якості. Вона полягає в тому, що після певної кількості шарів точність моделі перестає покращуватися або навіть починає знижуватися [18].

Причиною цього явища є складність навчання дуже глибоких нейронних мереж. Під час процесу навчання інформація про помилку повинна передаватися від вихідного шару до початкових шарів мережі. У разі великої кількості шарів градієнти можуть поступово зменшуватися, що призводить до уповільнення або повної зупинки навчання окремих шарів. У результаті додавання нових шарів не забезпечує очікуваного покращення якості моделі.

Для вирішення цієї проблеми у 2015 році дослідниками компанії Microsoft була запропонована архітектура Residual Network (ResNet). Основною ідеєю даного підходу стало використання механізму залишкового навчання (Residual Learning). На відміну від класичних згорткових нейронних мереж, де кожний блок навчається формувати нове представлення вхідних даних, у ResNet блок навчається визначати лише різницю між вхідними даними та бажаним результатом.

Основним елементом архітектури є залишковий блок (Residual Block). У такому блоці вхідні дані передаються двома шляхами. Перший шлях проходить через один або декілька згорткових шарів, де здійснюється виділення ознак. Другий шлях передає початкові дані без змін безпосередньо до виходу блоку. Після цього результати обох шляхів підсумовуються.

Принцип роботи залишкового блоку може бути описаний за допомогою формули:

$$y = F(x) + x, \tag{2.3}$$

де x— вхідні дані блоку,

$F(x)$— функція перетворення, яка реалізується набором згорткових шарів,

y— вихід блоку.

У формулі (2.3) функція $F(x)$ відповідає за виділення нових ознак, тоді як початкові дані x передаються безпосередньо на вихід блоку через спеціальне пропускне з'єднання (Skip Connection). Завдяки цьому мережа зберігає доступ до початкової інформації навіть після проходження великої кількості шарів.

Використання пропускних з'єднань дозволяє значно покращити процес навчання глибоких нейронних мереж. Під час зворотного поширення помилки градієнти можуть передаватися не лише через згорткові шари, а й через додаткові прямі з'єднання. Це зменшує вплив проблеми зникнення градієнта та забезпечує стабільніше навчання моделей великої глибини.

На основі запропонованого підходу було створено декілька модифікацій архітектури ResNet, які відрізняються кількістю шарів та складністю моделі. Усі основні модифікації перераховано в таблиці 2.2.

Таблиця 2.2 – Основні модифікації архітектури ResNet (ПОСИЛАННЯ)

Архітектура	Кількість шарів
ResNet-18	18
ResNet-34	34
ResNet-50	50
ResNet-101	101
ResNet-152	152

Зі збільшенням кількості шарів підвищується здатність мережі виділяти складні ознаки зображень. Разом із цим збільшуються вимоги до обчислювальних ресурсів та часу обробки даних. Тому вибір конкретної моделі залежить від поставленої задачі та доступних ресурсів.

Архітектура ResNet отримала широке поширення в задачах комп'ютерного зору. Вона використовується для класифікації зображень, детектування об'єктів, сегментації зображень, а також у системах пошуку схожих зображень. Висока

якість виділення ознак зробила її однією з найбільш популярних архітектур глибокого навчання.

У рамках даної роботи використовується попередньо навчена модель ResNet50-v2 [26]. Вона застосовується не для класифікації тварин, а для формування векторних представлень зображень. Отримані вектори містять інформацію про візуальні особливості фотографії та використовуються під час пошуку схожих зображень у базі даних. Використання попередньо навченої моделі дозволяє отримати якісні векторні представлення без необхідності виконання тривалого навчання нейронної мережі на власному наборі даних.

Таким чином, архітектура ResNet вирішує проблему навчання глибоких згорткових нейронних мереж за рахунок використання залишкових блоків та пропускних з’єднань. Завдяки високій якості виділення ознак і можливості використання попередньо навчених моделей даний підхід є доцільним для задачі формування векторних представлень та подальшого пошуку схожих зображень у розроблюваній системі.

2.3.4 Формування векторних представлень

Одним із ключових результатів роботи згорткових нейронних мереж є формування векторного представлення зображення, яке може використовуватися для подальшого аналізу та порівняння. Таке представлення дозволяє описати візуальний вміст фотографії у компактній числовій формі та використовується в багатьох сучасних системах комп’ютерного зору [21].

Під час проходження зображення через послідовність згорткових шарів нейронна мережа поступово виділяє його характерні ознаки. На початкових рівнях аналізуються прості елементи, наприклад контури, текстури або локальні ділянки зображення. На наступних рівнях мережа формує більш складні представлення, які описують окремі частини об’єктів або їх загальну структуру.

Після проходження через нейронну мережу зображення перетворюється на компактний вектор, який містить значно менше даних, ніж початкова фотографія. При цьому векторне представлення зберігає найбільш важливі характеристики

об’єкта, необхідні для подальшого аналізу. Завдяки цьому система може працювати не з великими масивами пікселів, а з компактними числовими представленнями.

Використання векторних представлень дозволяє суттєво спростити подальшу обробку зображень. Замість виконання складних операцій над піксельними даними система працює з векторами фіксованої довжини, що зменшує обчислювальні витрати та спрощує реалізацію алгоритмів пошуку.

У сучасних системах комп’ютерного зору векторні представлення широко використовуються для класифікації об’єктів, розпізнавання обличь, рекомендаційних систем та пошуку схожих зображень. У даній системі векторні представлення застосовуються як основне представлення фотографій тварин для подальшого оцінювання ступеня їх подібності.

2.4 Метод оцінювання подібності зображень

Після формування векторних представлень для зображень необхідно визначити спосіб оцінювання подібності між ними. Саме на цьому етапі система отримує можливість визначати, наскільки схожими є дві фотографії та чи можуть вони відповідати одному й тому самому об'єкту. У задачах пошуку аналогічних зображень якість оцінювання подібності безпосередньо впливає на точність отриманих результатів [23].

Оскільки кожне зображення після обробки нейронною мережею представлене у вигляді числового вектора, задача пошуку схожих фотографій зводиться до порівняння відповідних векторів подібності. Для цього використовується математичний апарат векторного аналізу, який дозволяє оцінювати ступінь близькості між векторними представленнями об'єктів.

У сучасних системах комп’ютерного зору застосовуються різні підходи до оцінювання подібності векторів. Найбільш поширеними є евклідова відстань, мангеттенська відстань та косинусна міра подібності. Кожен із цих методів має свої особливості та сфери застосування.

Під час виконання роботи, для оцінювання подібності використовується косинусна міра подібності. Даний підхід широко застосовується у задачах пошуку аналогів та добре підходить для порівняння векторних представлень, сформованих згортковими нейронними мережами. Використання косинусної міри дозволяє оцінювати схожість між зображеннями на основі напрямків відповідних векторів незалежно від їх абсолютних значень.

2.4.1 Векторний простір ознак

Після формування \векторних представлень кожне зображення може бути представлене у вигляді числового вектора фіксованої довжини. Такі вектори використовуються для подальшого порівняння фотографій та виконання пошуку схожих зображень. Для математичного опису цього процесу використовується поняття векторного простору ознак [21].

Векторний простір ознак являє собою математичну модель, у якій кожне зображення описується набором числових координат. Якщо векторне представлення містить (n) числових значень, то відповідне зображення може бути представлене як точка у просторі розмірності (n). У такому випадку кожна компонента цього представлення визначає одну з координат даної точки.

Нехай для деякого зображення сформовано векторне представлення

$$E = (e_1, e_2, \dots\ e_n)$$

де $(e_1, e_2, \dots\ e_n)$ — компоненти векторного представлення зображення.

У такому випадку кожна фотографія, яка обробляється системою, може бути представлена окремою точкою у просторі ознак. Сукупність усіх векторних представлень формує багатовимірний простір, у якому можуть виконуватися математичні операції порівняння та пошуку.

Як показано на рисунку 2.6, векторні представлення схожих об'єктів зазвичай розташовуються ближче один до одного, ніж векторне представлення різних об'єктів. Наприклад, фотографії породи Сіамські коти можуть утворювати окрему групу точок, тоді як векторні представлення розташовуються на більшій відстані.

У реальних системах кількість вимірів простору ознак може становити сотні або навіть тисячі координат, тому наведений рисунок є лише спрощеною ілюстрацією.

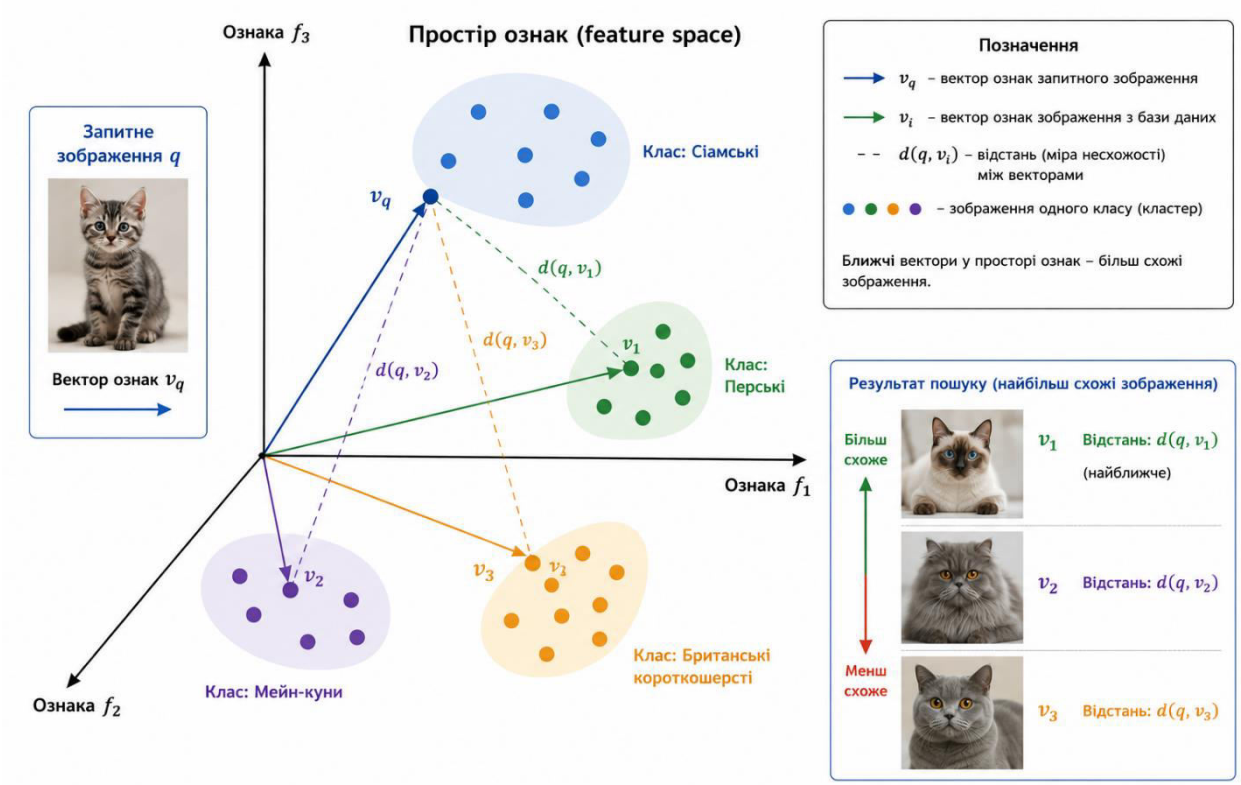

Рисунок 2.6 – Векторне представлення у просторі ознак

Основною перевагою використання простору ознак є можливість порівнювати зображення за допомогою математичних методів. Замість аналізу окремих пікселів система працює з компактними векторними представленнями, які містять найбільш важливу інформацію про об'єкт. Це дозволяє суттєво зменшити обсяг обчислень та підвищити швидкість пошуку.

У задачах пошуку аналогів пошук схожих фотографій фактично зводиться до визначення ступеня близькості між відповідними векторними представленнями. Чим ближче розташовані вектори у просторі ознак, тим більш схожими вважаються відповідні зображення. Для кількісного оцінювання такої близькості використовуються спеціальні міри подібності.

Таким чином, векторний простір ознак є математичною основою для порівняння векторних представлень та реалізації пошуку схожих зображень.

Представлення фотографій у вигляді точок багатовимірного простору дозволяє застосовувати математичні методи оцінювання подібності, одним із яких є косинусна міра подібності, що розглядається у наступному підрозділі.

2.4.2 Косинусна міра подібності і пороговий критерій пошуку

Після представлення зображень у вигляді векторних представлень виникає необхідність кількісного оцінювання ступеня подібності між ними. Для цього використовуються спеціальні математичні метрики, які дозволяють визначити, наскільки близькими є два векторні представлення у просторі ознак. Однією з найбільш поширених метрик у задачах пошуку аналогів є косинусна міра подібності (Cosine Similarity) [23].

Основна ідея косинусної міри подібності полягає у визначенні кута між двома векторами. Якщо вектори мають схожий напрямок, то вони вважаються подібними. Якщо напрямки суттєво відрізняються, рівень подібності зменшується. На відміну від деяких інших метрик, косинусна міра враховує не абсолютні значення елементів вектора, а їх відносне співвідношення.

Для двох векторних представлень (A) та (B) косинусна міра подібності визначається наступним чином:

$$cos(\theta) = \frac{A \cdot B}{|A||B|}, \tag{2.5}$$

де $A \cdot B$ — скалярний добуток векторів;

|A| — довжина вектора A;

|B| — довжина вектора B;

θ — кут між векторами.

Відповідно до (2.5) значення косинусної міри подібності може змінюватися в діапазоні від -1 до 1. Для задач пошуку схожих зображень зазвичай використовуються додатні значення, оскільки векторні представлення схожих об'єктів мають близькі напрямки у просторі ознак.

Інтерпретація значень косинусної міри подібності наведена у таблиці 2.3.

Таблиця 2.3 – Інтерпретація значень косинусної міри подібності

Значення косинусної міри подібності	Інтерпретація
1,0	Вектори збігаються
0,8–1,0	Висока подібність
0,5–0,8	Помірна подібність
0–0,5	Низька подібність
0	Вектори ортогональні
< 0	Протилежні напрямки

У задачах пошуку схожих зображень косинусна міра подібності має ряд переваг. По-перше, вона є стійкою до змін масштабу векторів. Це означає, що два векторних представлення можуть вважатися схожими навіть у випадку відмінності їх абсолютних значень. По-друге, обчислення косинусна міра є відносно простим та не потребує значних обчислювальних ресурсів. По-третє, дана метрика широко використовується у сучасних системах пошуку та рекомендаційних системах, що підтверджує її ефективність на практиці.

Для задачі пошуку втрачених тварин використання косинусної міри подібності є доцільним, оскільки векторні представлення формуються попередньо навченою нейронною мережею та містять узагальнені ознаки фотографій. У такому випадку косинусної міри подібності дозволяє оцінювати схожість між фотографіями на рівні виділених ознак, а не окремих пікселів зображення.

Після обчислення косинусної міри подібності для всіх векторних представлень, що зберігаються у базі даних, система отримує числову оцінку схожості між фотографією користувача та кожним записом.

Після обчислення значень косинусної подібності необхідно визначити, які результати будуть відображені користувачу. Для цього використовується пороговий критерій пошуку. Його основна задача полягає у відсіюванні зображень, рівень подібності яких є недостатнім для визнання їх потенційним збігом.

Під час роботи системи кожне значення косинусна міра подібності порівнюється із заздалегідь встановленим порогом. Якщо рівень подібності

перевищує задане значення, відповідне зображення включається до результатів пошуку. У протилежному випадку запис відкидається та не відображається користувачу.

Як показано в таблиці 2.4, вибір порогового значення впливає на характеристики пошуку. Низький поріг дозволяє отримати більшу кількість результатів, однак збільшує ймовірність появи хибних збігів. Високий поріг, навпаки, підвищує точність пошуку, але може призводити до втрати частини потенційно корисних результатів [8].

Таблиця 2.4 – Вплив порогового значення на результати пошуку

Порогове значення	Особливості результатів
Низьке	Велика кількість результатів, підвищена кількість хибних збігів
Середнє	Баланс між повнотою та точністю пошуку
Високе	Менша кількість результатів, підвищена точність пошуку

Таким чином, косинусна міра подібності дозволяє оцінити ступінь схожості між векторними представленнями, а пороговий критерій забезпечує відбір найбільш релевантних результатів для користувача. Спільне використання цих механізмів створює основу для реалізації пошуку схожих зображень у розробленій системі.

2.5 Метод Content-Based Image Retrieval

Традиційні інформаційно-пошукові системи переважно працюють із текстовими даними. Пошук у таких системах здійснюється за ключовими словами, описами або іншими текстовими характеристиками об’єктів. Проте у випадку пошуку схожих зображень використання лише текстового опису часто є

недостатнім, оскільки значна частина інформації про об’єкт міститься безпосередньо у візуальному вмісті фотографії [20].

Для вирішення даної проблеми використовуються системи Content-Based Image Retrieval (CBIR). Основною особливістю такого підходу є виконання пошуку на основі візуальних характеристик зображення, а не його текстового опису. У цьому випадку система аналізує саму фотографію, виділяє її ознаки та виконує пошук схожих зображень серед наявних даних.

Content-Based Image Retrieval (CBIR) — це підхід до пошуку зображень, який базується на аналізі їх візуального вмісту. На відміну від традиційних інформаційно-пошукових систем, де пошук виконується за текстовими описами або ключовими словами, у системах CBIR основним джерелом інформації виступає саме зображення [9].

Поява даного підходу пов'язана зі зростанням кількості цифрових зображень та необхідністю їх ефективного пошуку [27]. Використання лише текстових описів має ряд недоліків. По-перше, створення якісного опису потребує додаткового часу. По-друге, різні користувачі можуть по-різному описувати один і той самий об'єкт. По-третє, частина візуальної інформації взагалі не може бути повністю передана текстом. У результаті пошук за ключовими словами не завжди дозволяє знайти всі релевантні зображення.

Основна ідея Content-Based Image Retrieval полягає у використанні візуальних характеристик зображення для пошуку аналогічних об'єктів. У цьому випадку система аналізує фотографію, виділяє її ознаки та формує певне внутрішнє представлення, яке використовується під час пошуку. Надалі отримане представлення порівнюється з представленнями інших зображень, що зберігаються у базі даних.

У класичних системах CBIR виділення ознак виконувалося за допомогою спеціальних алгоритмів комп’ютерного зору, які аналізували колірні характеристики, текстури, контури або локальні дескриптори зображення. Проте такі підходи часто виявлялися недостатньо ефективними для складних реальних задач через високу залежність від умов зйомки, освітлення та особливостей об'єкта.

Розвиток глибокого навчання дозволив суттєво підвищити ефективність систем Content-Based Image Retrieval. Замість ручного формування ознак почали використовуватись згорткові нейронні мережі, які автоматично виділяють найбільш інформативні характеристики зображення. У сучасних системах результатом роботи таких мереж зазвичай є векторне представлення фотографії, яке використовується для подальшого порівняння.

Пошук у системах CBIR фактично зводиться до знаходження зображень, векторні представлення яких є найбільш подібними до векторного представлення вхідного запиту. Для цього використовуються різні метрики оцінювання подібності, серед яких однією з найбільш поширених є косинусна міра подібності.

Перевагою підходу Content-Based Image Retrieval є можливість виконувати пошук навіть у випадках, коли відсутній текстовий опис об'єкта. Крім того, система може знаходити схожі зображення за їх візуальними характеристиками, що особливо важливо для задач комп’ютерного зору. Завдяки цьому CBIR широко застосовується у медичних інформаційних системах, біометрії, цифрових архівах зображень, системах рекомендацій та пошуку об'єктів на фотографіях.

Таким чином, у розробленій системі метод Content-Based Image Retrieval використовується для організації пошуку на основі візуальних характеристик фотографій. Поєднання векторних представлень, косинусної міри подібності та порогового критерію дозволяє реалізувати автоматизований пошук потенційно схожих зображень тварин серед даних, що зберігаються у системі.

РОЗДІЛ 3 ПРОГРАМНА РЕАЛІЗАЦІЯ СИСТЕМИ

Після аналізу предметної області та сучасних підходів до пошуку схожих зображень необхідно перейти до проєктування та реалізації програмної системи. На даному етапі важливо визначити архітектуру системи, структуру її компонентів, способи взаємодії між модулями та підходи до обробки зображень.

У рамках даної роботи реалізовується система пошуку втрачених домашніх тварин на основі фотографій користувачів. Основною особливістю системи є використання методів комп'ютерного зору та пошуку подібності для автоматичного порівняння зображень. Для цього система повинна забезпечувати завантаження фотографій, формування векторних ознак, збереження даних та виконання пошуку схожих зображень у базі даних.

Під час проєктування системи необхідно враховувати не лише алгоритми обробки зображень, а й особливості взаємодії користувача із програмою. У рамках роботи для взаємодії із системою використовується Telegram-бот, який забезпечує отримання фотографій, запуск пошуку та відображення результатів користувачу.

Крім цього, важливим етапом є вибір технологій та інструментів реалізації. Для розробки системи використовуються мова програмування Java, фреймворк Spring Boot, PostgreSQL для зберігання даних, а також ONNX Runtime для роботи із попередньо навченою моделлю ResNet50. Подібний підхід дозволяє поєднати можливості сучасних методів комп'ютерного зору із серверною архітектурою програмної системи.

У даному розділі розглядаються архітектура системи, структура її основних компонентів, модель даних, вибір технологій та особливості реалізації пошуку подібних тварин і Telegram-бота.

3.1 Загальна архітектура системи

Архітектура програмної системи визначає загальну структуру програмного забезпечення, взаємодію між його компонентами та способи обробки даних. Правильно спроєктована архітектура дозволяє забезпечити модульність системи, спростити її супровід та створити основу для подальшого розширення функціональності.

У рамках даної роботи реалізовано систему пошуку втрачених тварин на основі фотографій користувачів. Основними функціями системи є приймання фотографій через Telegram-бота, формування векторних представлень за допомогою нейронної мережі, виконання пошук аналогів та збереження даних у базі даних.

Система побудована за модульним принципом і складається з окремих компонентів, кожен з яких відповідає за виконання визначених функцій. Такий підхід дозволяє зменшити зв'язність між частинами програми, спростити тестування та забезпечити можливість заміни окремих компонентів без суттєвого впливу на інші частини системи [28].

Під час проєктування архітектури особлива увага приділялась розділенню відповідальності між компонентами, що відповідають за взаємодію з користувачем, бізнес-логіку, обробку зображень та збереження даних. Це дозволило створити більш структуровану та зрозумілу програмну систему.

У даному підрозділі розглядається загальна схема архітектури системи, основні програмні компоненти та принципи їх взаємодії під час виконання основних сценаріїв роботи.

3.1.1 Загальна схема системи

Для реалізації системи пошуку втрачених тварин була спроєктована багатокомпонентна архітектура, яка поєднує засоби взаємодії з користувачем,

модулі обробки зображень, підсистему пошуку та засоби збереження даних. Загальна схема системи наведена на рисунку 3.1.

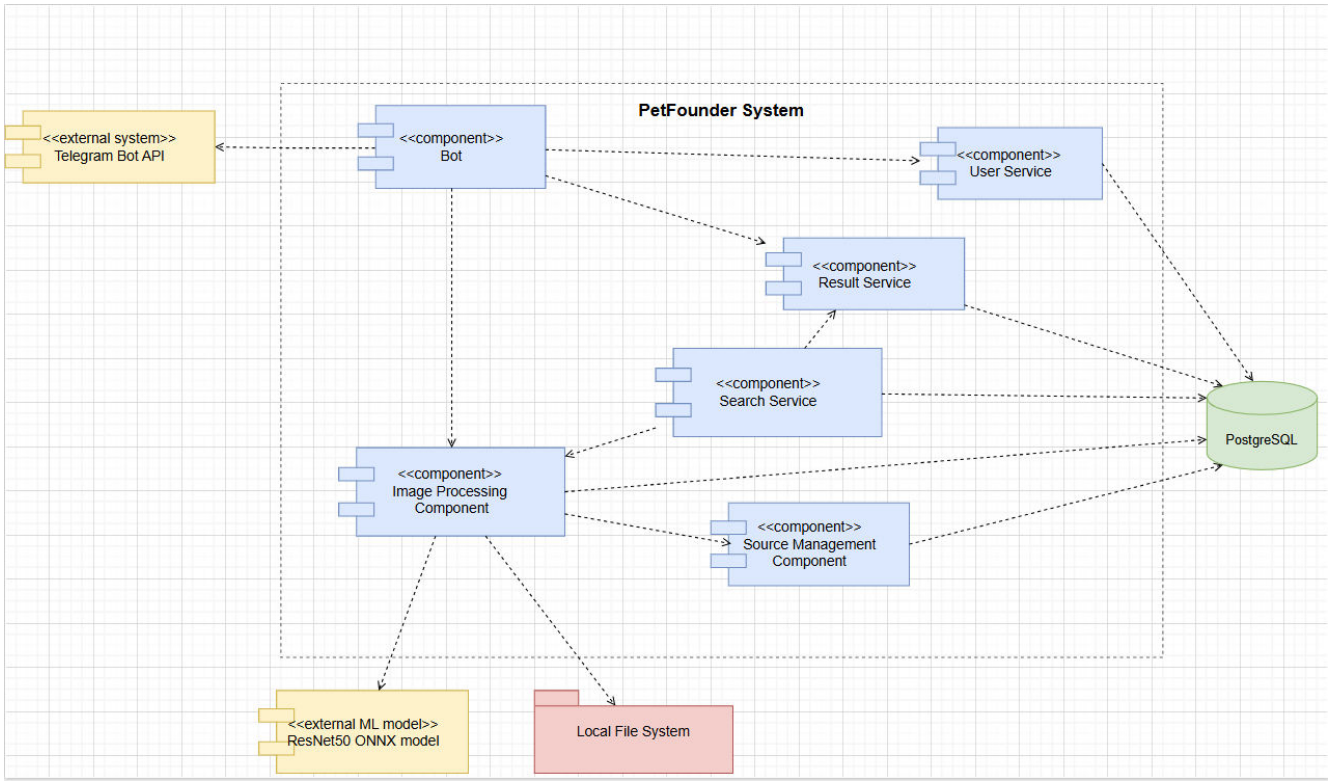

Рисунок 3.1 – Загальна архітектура системи

Як показано на рисунку 3.1, система складається з декількох взаємопов’язаних компонентів, кожен із яких відповідає за виконання окремої групи функцій. Основними складовими є Telegram-бот, серверна частина системи, модуль обробки зображень, підсистема пошуку схожих зображень та база даних.

Основним засобом взаємодії користувача із системою є Telegram-бот. Через нього користувач може створювати пошуки, надсилати фотографії тварин, переглядати результати пошуку та керувати власними записами. Telegram-бот передає отримані дані до серверної частини системи для подальшої обробки.

Серверна частина виконує координацію роботи інших компонентів системи. Вона забезпечує обробку запитів користувачів, взаємодію з базою даних, запуск процедур аналізу зображень та формування результатів пошуку.

Модуль обробки зображень відповідає за підготовку фотографій та формування їх векторних представлень. Отримані результати використовуються підсистемою пошуку для порівняння з даними, що вже містяться у системі.

Підсистема пошуку виконує порівняння отриманих представлень зображень із даними, що зберігаються у базі. На основі результатів порівняння формується список потенційно схожих фотографій, який надалі передається користувачу.

Для збереження інформації використовується база даних, у якій містяться відомості про користувачів, створені пошуки, фотографії та службові дані системи. Окремо організовано зберігання файлів зображень, які використовуються під час виконання пошуку.

Таким чином, архітектура системи побудована за модульним принципом та забезпечує розділення відповідальності між основними компонентами. Такий підхід спрощує супровід програмного забезпечення та створює можливість подальшого розвитку функціональності системи.

3.1.2 Основні компоненти

Система складається з декількох взаємопов’язаних компонентів, кожен із яких виконує окрему функцію у процесі пошуку аналогів. Подібний підхід дозволяє спростити структуру програми, підвищити гнучкість архітектури та забезпечити можливість подальшого розширення функціональності системи.

Одним із основних компонентів є модуль взаємодії з користувачем, реалізований у вигляді Telegram-бота. Даний компонент відповідає за прийом повідомлень, фотографій та команд користувача. Крім цього, бот забезпечує відображення результатів та взаємодію із Telegram Bot API.

Важливу роль у системі відіграє компонент обробки зображень. Його основним завданням є попередня підготовка фотографій до подальшого аналізу нейронною мережею. На даному етапі виконується масштабування, нормалізація та перетворення зображення у формат, необхідний для роботи моделі комп’ютерного зору. Даний компонент також відповідає за формування векторних представлень. Для цього у системі використовується попередньо навчена модель ResNet50 у

форматі ONNX. Запуск моделі виконується за допомогою ONNX Runtime, який забезпечує виконання inference у Java-застосунку без необхідності використання окремого Python-сервісу.

Ще одним важливим компонентом є модуль пошуку. Його основним завданням є порівняння векторних представлень між собою за допомогою косинусної міри подібності та формування списку релевантних результатів, значення подібності яких перевищують встановлений поріг.

Окремим компонентом системи є модуль керування результатами пошуку. Даний компонент відповідає за обробку, формування та відображення результатів користувачу.

Для роботи із користувачами використовується окремий модуль керування користувачами. Його основним завданням є збереження та обробка інформації про користувачів системи та їх пошуки.

Для збереження інформації у системі використовується PostgreSQL. У базі даних зберігаються векторні представлення, інформація про користувачів та результати пошуку. Файли зображень зберігаються окремо у локальному файловому сховищі.

Окремим компонентом системи є модуль роботи із зовнішніми джерелами даних. Даний компонент використовується для імпорту та обробки фотографій із зовнішніх наборів даних, зокрема Oxford-IIIT Pet Dataset, який застосовується для тестування та оцінювання якості пошуку.

Таким чином, система складається із декількох взаємопов’язаних компонентів, які спільно забезпечують виконання пошуку та взаємодію користувача із системою пошуку втрачених тварин.

3.1.3 Взаємодія модулів

Взаємодія модулів системи побудована таким чином, щоб кожен компонент виконував власну задачу та мінімально залежав від внутрішньої реалізації інших модулів. Подібний підхід дозволяє зробити систему більш гнучкою та спрощує її подальший розвиток [29].

Процес роботи системи починається із взаємодії користувача з Telegram бот. Користувач надсилає команду та фотографію тварини. Після цього модуль Telegram бот отримує зображення та передає його до модуля попередньої обробки.

На етапі попередньої обробки виконується масштабування та нормалізація фотографії. Далі підготовлене зображення передається до модуля генерації векторних представлень.

Модуль векторних представлень використовує ONNX Runtime для запуску CNN-моделі ResNet. У результаті роботи моделі формується векторне представлення фотографії. Після цього векторне представлення може бути або збережене у базі даних, або використане для пошуку.

Якщо користувач виконує пошук зниклої тварини, векторне представлення передається до модуля пошуку. Даний компонент отримує векторне представлення із бази даних та виконує обчислення косинус між вхідним вектором та іншими представленнями.

Після завершення порівняння система формує список найбільш схожих результатів. Далі результати передаються назад до Telegram бот, який надсилає користувачу відповідні фотографії та інформацію про знайдені збіги.

У випадку додавання знайденої тварини векторне представлення та фотографія зберігаються у PostgreSQL для подальшого використання системою.

Таким чином, взаємодія модулів системи базується на послідовній обробці даних: від отримання фотографії до формування векторних представлень та виконання пошуку.

3.2 Проєктування моделі даних

Під час розробки системи пошуку схожих зображень важливу роль відіграє організація збереження даних. Система повинна не лише виконувати пошук, але й забезпечувати можливість збереження фотографій, їх векторних представлень та пов’язаної інформації про записи.

Модель даних проєктувалася з урахуванням особливостей пошуку аналогів та підходу, основаного на векторних представленнях. Основною задачею моделі даних є збереження векторних представлень, фотографій та службової інформації, необхідної для виконання пошуку схожих зображень.

Оскільки система працює із фотографіями тварин, необхідно було передбачити механізми збереження метаданих зображень, їх векторних представлень та інформації про пошуки користувачів. Крім цього, модель даних повинна забезпечувати підтримку зв'язків між основними об'єктами предметної області та можливість ефективного отримання даних під час виконання пошуку.

Для реалізації збереження даних використовується реляційна система управління базами даних PostgreSQL. Модель даних побудована на основі декількох взаємопов'язаних сутностей, які описують користувачів, пошуки, фотографії, результати пошуку та джерела даних.

У даному підрозділі розглядається структура бази даних системи, спосіб представлення у векторному просторі, а також організація збереження фотографій та пов'язаної з ними інформації.

3.2.1 Структура збереження зображень

Модель даних побудована на основі п'яти основних сутностей: User, Search, Image, Result та Source. Сукупність цих сутностей дозволяє зберігати інформацію про користувачів, фотографії тварин, пошукові запити та результати пошуку [30].

Структура бази даних системи представлена на рисунку 3.2.

Сутність User використовується для збереження інформації про користувачів системи. Для кожного користувача зберігаються ідентифікатор Telegram, ім'я користувача та службова інформація. Один користувач може створювати декілька пошуків і додавати декілька фотографій, тому із сутностями Search та Image вона пов'язана зв'язком типу «один до багатьох».

Сутність Search призначена для збереження інформації про пошукові запити. Кожен пошук належить конкретному користувачу та містить посилання на фотографію, яка використовується як вхідне зображення для пошуку. Крім цього,

один пошук може мати декілька результатів, що дозволяє зберігати інформацію про всі знайдені збіги.

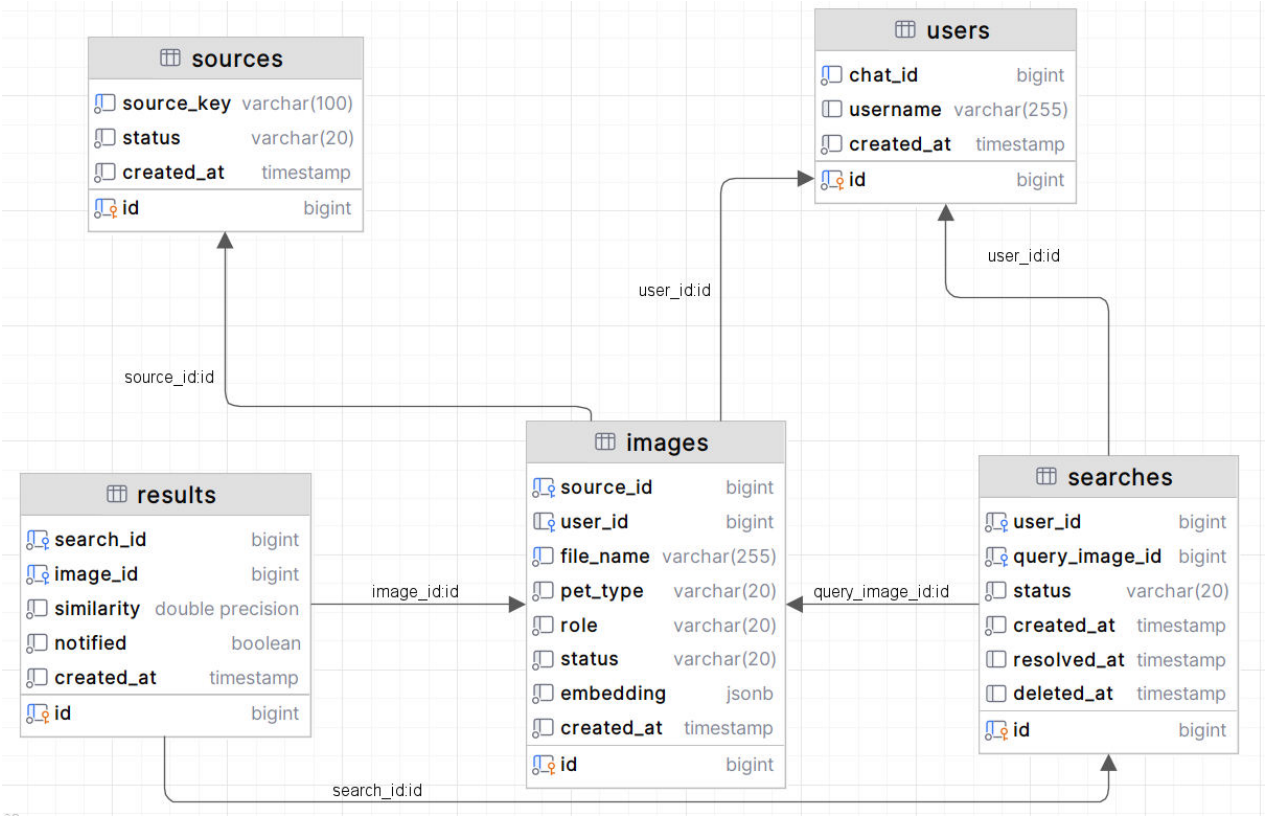

Рисунок 3.2 – Схема бази даних системи

Центральною сутністю моделі даних є Image. Вона містить інформацію про фотографії тварин, що використовуються у системі. Для кожного зображення зберігаються тип тварини, роль зображення, статус запису, шлях до файлу та векторне представлення. Саме ця сутність використовується як основне джерело даних під час виконання пошуку.

Для збереження результатів пошуку використовується сутність Result. Вона містить посилання на пошук, знайдене зображення та значення подібності, отримане під час обчислення косинусної міри подібності. Такий підхід дозволяє зберігати результати пошуку та використовувати їх для подальшого відображення користувачу.

Окремою сутністю є Source, яка використовується для збереження інформації про джерела даних. Одне джерело може містити багато зображень, тому між сутностями Source та Image реалізовано зв’язок типу «один до багатьох». Це дозволяє відокремлювати користувацькі фотографії від даних, імпортованих із зовнішніх наборів зображень.

Запропонована структура бази даних забезпечує збереження всієї інформації, необхідної для роботи системи, підтримує цілісність даних та дозволяє ефективно виконувати операції пошуку.

3.2.2 Векторні представлення

Основою пошуку за подібністю у даній системі є векторні представлення зображень, які формуються CNN-моделлю ResNet. Саме представлення використовуються для порівняння фотографій та пошуку схожих результатів.

Після проходження зображення через нейронну мережу система отримує представлення у вигляді числового вектора. Кожен елемент такого вектора описує певні характеристики, тим самим формує компактне представлення зображення.. При цьому векторні представлення схожих фотографій повинні знаходитися близько один до одного у векторному просторі.

У рамках системи векторні представлення додатково нормалізуються перед збереженням та порівнянням. Це дозволяє більш коректно використовувати подібність за косинусом під час пошуку.

У рамках реалізованої системи векторне представлення зберігається у вигляді масиву числових значень, що дозволяє використовувати стандартні можливості PostgreSQL без додаткових розширень.

Після отримання векторне представлення записується у базу даних разом із іншою інформацією про фотографію. У подальшому система може швидко їх отримувати та виконувати пошук без необхідності повторного запуску нейронної мережі для кожного зображення.

Використання векторного представлення дозволяє суттєво зменшити обсяг даних, що порівнюються під час пошуку. Ба більше, це дозволяє прискорити

Крім цього, представлення є більш стійкими до змін освітлення, масштабу або ракурсу у порівнянні із прямим порівнянням пікселів. Саме тому подібний підхід широко використовується у сучасних системах комп’ютерного зору.

3.2.3 Організація збереження зображень

Окрім збереження метаданих та векторних представлень, важливим аспектом роботи системи є організація зберігання самих фотографій. Оскільки система працює із великою кількістю зображень, необхідно забезпечити їх ефективне збереження та швидкий доступ під час виконання пошуку.

У рамках даної роботи було вирішено не зберігати файли зображень безпосередньо у базі даних. Замість цього фотографії розміщуються у локальному файловому сховищі, а у базі даних зберігається лише шлях до відповідного файлу. Такий підхід дозволяє зменшити розмір бази даних, прискорити резервне копіювання та спростити обробку великих бінарних файлів [30].

Після отримання фотографії від користувача система створює унікальний запис у файловому сховищі та зберігає файл на диску. Одночасно у базі даних створюється запис, який містить інформацію про зображення, його тип, статус, дату створення та шлях до відповідного файлу. Завдяки цьому система може швидко знаходити необхідне зображення та отримувати доступ до пов’язаної з ним інформації.

Використання окремого файлового сховища також спрощує виконання операцій із зображеннями. Зокрема, система може виконувати попередню обробку фотографій, формування векторних представлень та відображення результатів пошуку без необхідності додаткового завантаження великих бінарних даних із бази даних.

Ще однією перевагою такого підходу є можливість подальшого масштабування системи. У разі збільшення кількості фотографій або переходу до хмарних сховищ структура моделі даних може залишатися практично незмінною, оскільки база даних містить лише метадані та посилання на файли.

3.3 Вибір технологій та інструментів

Під час розробки програмної системи важливу роль відіграє вибір технологій та інструментів. Саме від цього залежить зручність реалізації, можливість інтеграції окремих компонентів, а також стабільність роботи системи.

У рамках даної роботи система поєднує декілька напрямів: Telegram бот, роботу із базою даних, обробку зображень та використання нейронної мережі для пошуку. Через це необхідно було обрати технології, які добре підходять для реалізації подібної архітектури.

Основною мовою програмування у проєкті є Java. Серверна частина системи реалізована із використанням фреймворку Spring Boot, який забезпечує підтримку модульної архітектури, керування залежностями та інтеграцію із базою даних. Для реалізації окремих компонентів системи використовуються спеціалізовані інструменти та бібліотеки, зокрема Telegram Bot API для взаємодії із користувачем, ONNX Runtime для виконання нейронної мережі, PostgreSQL для збереження даних та Docker Compose для контейнеризації програмного середовища..

Окремо важливу роль відіграє система збірки Gradle, яка використовується для керування залежностями та запуску проєкту.

У даному підрозділі розглядаються основні технології та інструменти, які використовуються у рамках розроблюваної системи.

3.3.1 Мова програмування Java

Основною мовою програмування для реалізації системи була обрана Java. Дана мова широко використовується для розробки серверних застосунків та має розвинену екосистему інструментів, що дозволяє реалізовувати складні програмні системи різного призначення.

Однією з причин вибору Java є можливість реалізувати всі основні компоненти системи в межах єдиного програмного середовища. Зокрема, на Java реалізовано Telegram-бот, серверну логіку застосунку, роботу з базою даних

PostgreSQL, формування векторних представлень та алгоритми пошуку. Такий підхід спрощує підтримку системи та усуває необхідність використання декількох мов програмування або окремих сервісів [31].

Важливою перевагою Java є її незалежність від платформи. Завдяки використанню Java Virtual Machine (JVM) застосунок може запускатися на різних операційних системах без необхідності зміни програмного коду. Це спрощує розгортання системи та забезпечує її переносимість між різними середовищами виконання.

Ще однією перевагою є наявність великої кількості готових бібліотек та фреймворків. Для реалізації Telegram-бота використовуються бібліотеки Telegram Bot API, для роботи із базою даних — засоби Hibernate, а для запуску моделі комп'ютерного зору — ONNX Runtime. Використання готових рішень дозволило зосередитися на реалізації логіки системи та алгоритмів пошуку схожих зображень.

У рамках даної роботи Java також використовується для інтеграції з ONNX Runtime. Це дозволяє виконувати inference попередньо навченої моделі ResNet без необхідності створення окремого Python-сервісу. У результаті вся система працює як єдиний програмний застосунок, що спрощує її розгортання та подальшу підтримку.

Окремою перевагою Java є автоматичне керування пам'яттю та підтримка багатопотоковості. Це дозволяє одночасно обробляти декілька запитів користувачів Telegram-бота та виконувати операції обробки зображень без необхідності ручного керування системними ресурсами.

Використання Java дозволило реалізувати всі основні компоненти системи в межах єдиної технологічної платформи, забезпечити інтеграцію з базою даних та засобами комп'ютерного зору, а також спростити подальший розвиток програмного забезпечення.

3.3.2 Використання Spring Boot

Для реалізації серверної частини системи використовується фреймворк Spring Boot. Даний фреймворк є одним із найбільш поширених рішень для розробки

Java-застосунків та широко використовується у створенні вебсервісів, корпоративних систем і мікросервісної архітектури [32].

Однією з основних переваг Spring Boot є спрощення налаштування програмного середовища. Більшість необхідних компонентів конфігуруються автоматично, що дозволяє скоротити кількість службового коду та зосередитися на реалізації бізнес-логіки системи.

У рамках даної роботи Spring Boot використовується як основа серверної частини застосунку. За його допомогою реалізовано взаємодію між окремими модулями системи, роботу із базою даних, обробку запитів користувачів та керування життєвим циклом компонентів програми.

Важливою особливістю Spring Boot є підтримка принципу Dependency Injection. Даний підхід дозволяє передавати залежності між компонентами автоматично, що спрощує модульну побудову системи та зменшує зв’язність між окремими класами. Завдяки цьому окремі модулі можуть розроблятися та модифікуватися незалежно один від одного.

Для роботи із базою даних у системі використовуються засоби Spring Data JPA та Hibernate, які інтегровані у Spring Boot. Вони забезпечують відображення об’єктів Java на таблиці бази даних та спрощують виконання операцій збереження, оновлення та отримання даних. Це дозволяє працювати із сутностями предметної області без необхідності створення великої кількості SQL-запитів вручну.

Крім цього, Spring Boot забезпечує зручну інтеграцію із зовнішніми бібліотеками. У рамках даної роботи через нього організовано взаємодію із Telegram Bot API, ONNX Runtime та PostgreSQL. Завдяки використанню єдиного фреймворку всі основні компоненти системи працюють у межах одного програмного застосунку.

Таким чином, використання Spring Boot дозволило спростити розробку серверної частини системи, забезпечити модульну архітектуру застосунку та організувати взаємодію між усіма основними компонентами програмного забезпечення.

3.3.3 Система управління базою даних та контейнеризація

Для збереження даних у системі використовується реляційна система керування базами даних PostgreSQL. Дана СУБД є одним із найбільш поширених рішень для серверних застосунків та забезпечує надійне зберігання структурованих даних [33].

У рамках даної роботи PostgreSQL використовується для збереження інформації про користувачів, пошуки, фотографії тварин та результати пошуку. Крім цього, у базі даних зберігаються векторні представлення та метадані зображень, які використовуються під час виконання пошуку схожих фотографій.

Однією з причин вибору PostgreSQL є її стабільність, продуктивність та можливість роботи із великими обсягами даних. Система також добре інтегрується із Java-застосунками за допомогою JDBC, Spring Data JPA та Hibernate, що спрощує реалізацію доступу до даних.

У рамках реалізованої системи PostgreSQL використовується як централізоване сховище інформації. Під час виконання пошуку система отримує векторні представлення із бази даних, виконує їх порівняння та формує список найбільш релевантних результатів. Використання єдиного сховища спрощує керування даними та забезпечує цілісність інформації між окремими компонентами системи.

Для спрощення розгортання та налаштування бази даних використовується Docker Compose [34]. Даний інструмент дозволяє запускати PostgreSQL у контейнеризованому середовищі та автоматизувати створення необхідної інфраструктури за допомогою конфігураційного файлу.

Однією з основних переваг Docker Compose є можливість швидкого розгортання системи без необхідності ручного встановлення та налаштування бази даних. У рамках даної роботи запуск PostgreSQL виконується автоматично разом із необхідними параметрами конфігурації, що спрощує процес розробки та тестування.

Контейнеризація також забезпечує ізоляцію окремих компонентів системи. База даних працює у власному контейнері незалежно від Java-застосунку, що

спрощує супровід програмного забезпечення та зменшує ризик виникнення конфліктів між компонентами.

Використання Docker Compose дозволяє забезпечити відтворюваність середовища виконання. Це означає, що база даних може бути розгорнута з однаковими налаштуваннями як на комп'ютері розробника, так і під час тестування або демонстрації системи. Подібний підхід зменшує ймовірність виникнення помилок, пов'язаних із відмінностями середовищ виконання.

Таким чином, використання PostgreSQL забезпечує надійне збереження даних системи, а Docker Compose спрощує розгортання та підтримку програмного середовища. Поєднання цих технологій дозволяє створити стабільну основу для роботи системи пошуку схожих зображень.

3.3.4 Виконання нейронної мережі

Для роботи із нейронною мережею у рамках даної роботи використовується ONNX Runtime. Даний інструмент дозволяє виконувати inference попередньо навченої моделі без необхідності запуску окремого Python-сервісу.

ONNX (Open Neural Network Exchange) є форматом представлення моделей машинного навчання, який дозволяє переносити моделі між різними платформами та фреймворками. Наприклад, модель може бути навчена у PyTorch або TensorFlow, а потім використана у Java-застосунку через ONNX Runtime.

Однією з причин вибору ONNX Runtime є можливість інтеграції CNN-моделі безпосередньо у Java-застосунок. Це дозволяє спростити архітектуру системи та уникнути необхідності створення окремого ML-сервісу.

У рамках даної роботи ONNX Runtime використовується для запуску попередньо навченої моделі ResNet. Для формування embeddings використовується попередньо навчена згорткова нейронна мережа ResNet50. Дана модель є однією з найбільш поширених архітектур комп'ютерного зору та широко застосовується у задачах класифікації зображень, розпізнавання об'єктів і пошуку схожих зображень. Однією з головних особливостей ResNet50 є використання залишкових

з’єднань (residual connections), які дозволяють ефективно навчати глибокі нейронні мережі та покращувати якість виділення ознак [18].

Ще однією перевагою ONNX Runtime є достатньо висока швидкість inference. Це важливо для similarity search, оскільки система повинна швидко обробляти фотографії користувачів.

Крім цього, ONNX Runtime підтримує роботу із різними платформами та мовами програмування. Через це даний інструмент є досить популярним у сучасних системах комп’ютерного зору.

Таким чином, використання ONNX Runtime дозволяє інтегрувати нейронну мережу у Java-застосунок.

3.3.5 Використання Telegram Bot API

Для реалізації взаємодії користувача із системою використовується Telegram Bot API. Даний інструмент надає можливість створювати програмних ботів, які можуть приймати повідомлення, обробляти команди користувачів та надсилати результати роботи системи.

Однією з причин вибору Telegram Bot API є відсутність необхідності створювати окремий вебінтерфейс або мобільний застосунок. Користувач може взаємодіяти із системою через вже встановлений месенджер Telegram, що спрощує використання програмного забезпечення та зменшує складність розробки.

У рамках даної роботи Telegram-бот використовується як основний інтерфейс системи. За допомогою бота користувач може надсилати фотографії тварин, запускати пошук схожих зображень, переглядати результати та створювати записи про знайдених або загублених тварин.

Важливою перевагою Telegram Bot API є підтримка передачі мультимедійних даних. Система може отримувати фотографії безпосередньо від користувача, що є необхідною умовою для реалізації пошуку. Після отримання зображення бот передає його до серверної частини застосунку для подальшої обробки.

Ще однією перевагою є простота інтеграції з Java-застосунками. Для реалізації Telegram-бота використовуються готові бібліотеки, які забезпечують

взаємодію із Telegram Bot API та спрощують обробку повідомлень, команд і фотографій.

Крім цього, використання Telegram дозволяє забезпечити доступ до системи з різних пристроїв без додаткового налаштування. Користувач може працювати із системою через мобільний застосунок, веб-версію або десктопний клієнт Telegram.

Отже, використання Telegram Bot API дозволяє реалізувати зручний інтерфейс взаємодії із користувачем, забезпечує підтримку роботи із фотографіями та спрощує доступ до функцій системи пошуку схожих зображень.

3.3.6 Система збірки Gradle

Для керування проєктом та залежностями у рамках даної роботи використовується система збірки Gradle. Даний інструмент є одним із найбільш поширених рішень для розробки Java-застосунків та дозволяє автоматизувати процес складання програмного забезпечення.

Gradle використовується для компіляції програмного коду, запуску застосунку та керування зовнішніми бібліотеками. У рамках даної роботи за його допомогою підключаються залежності, необхідні для роботи Spring Boot, PostgreSQL, Telegram Bot API, ONNX Runtime та інших компонентів системи.

Однією з переваг Gradle є централізоване керування залежностями. Усі необхідні бібліотеки описуються у конфігураційному файлі проєкту, після чого система автоматично завантажує та оновлює потрібні компоненти. Це спрощує підтримку проєкту та зменшує кількість ручних налаштувань.

Крім цього, Gradle забезпечує автоматизацію процесу складання застосунку. Завдяки цьому проєкт може бути швидко зібраний та запущений як під час розробки, так і під час тестування або демонстрації системи.

Ще однією перевагою є хороша інтеграція із середовищем розробки IntelliJ IDEA та іншими сучасними інструментами. Це спрощує процес розробки та дозволяє швидко виконувати збірку, тестування та запуск програмного забезпечення.

Таким чином, використання Gradle дозволяє автоматизувати процес складання застосунку, спростити керування залежностями та забезпечити зручну підтримку програмного проєкту.

3.4 Проєктування програмних модулів

Програмна реалізація системи побудована на основі кількох функціональних модулів, кожен із яких відповідає за окрему частину роботи застосунку. Такий підхід дозволяє розділити логіку взаємодії з користувачем, управління пошуками, обробку зображень, виконання пошуку та формування результатів.

Основними модулями системи є модуль Telegram-бота, модуль управління користувачами, модуль управління пошуками, модуль обробки зображень та векторних представлень, модуль формування результатів і модуль джерел даних. Кожен із них реалізований окремою групою класів і сервісів, що взаємодіють між собою у процесі виконання основних сценаріїв роботи системи.

У даному підрозділі розглядається призначення кожного програмного модуля, його основні класи та роль у загальному процесі пошуку схожих зображень тварин.

3.4.1 Модуль взаємодії з користувачем

Модуль Телеграм бота забезпечує взаємодію користувача з програмною системою та виконує роль основної точки входу до функціональності застосунку. Через Telegram-бот користувач може створювати нові пошуки, переглядати результати пошуку, працювати з історією власних пошуків та надсилати фотографії тварин для подальшої обробки.

Центральним компонентом модуля є клас LostPetsBot, який реалізує інтерфейс LongPollingSingleThreadUpdateConsumer. Даний клас отримує повідомлення від Telegram Bot API та виконує маршрутизацію запитів до відповідних обробників. Основною задачею класу є приймання оновлень від Telegram та передача керування компонентам, що реалізують бізнес-логіку системи.

Для спрощення підтримки програмного коду логіка обробки різних типів повідомлень винесена до окремих обробників. У системі реалізовано декілька спеціалізованих класів, які відповідають за виконання окремих сценаріїв взаємодії з користувачем:

- MainMenuHandler — обробка команд головного меню;
- PhotoSubmissionHandler — обробка фотографій, що надсилаються користувачем;
- SearchHistoryHandler — робота з історією пошуків;
- SearchResultsHandler — відображення результатів пошуку.

Подібний підхід дозволяє розділити функціональність між незалежними компонентами та зменшити складність центрального класу Telegram-бота.

Для підтримки покрокових сценаріїв взаємодії використовується механізм керування станом користувача. Збереження поточного стану реалізовано за допомогою сервісу UserSessionService, який дозволяє відстежувати етап виконання діалогу для кожного користувача окремо. Завдяки цьому система може коректно обробляти послідовність дій під час створення пошуку та перегляду результатів.

Для формування відповідей користувачам використовується сервіс BotResponseService. Даний компонент відповідає за створення повідомлень, клавіатур та інших елементів інтерфейсу Telegram. Окремо реалізовано клас BotKeyboardFactory, який використовується для формування інтерактивних кнопок та меню.

У процесі роботи користувач надсилає команду або фотографію до Telegram-бота. Після отримання повідомлення клас LostPetsBot визначає його тип та передає обробку відповідному компоненту. Далі виконується необхідна бізнес-логіка, формується відповідь та надсилається користувачу через Telegram Bot API.

Таким чином, модуль Telegram-бота забезпечує взаємодію користувача з програмною системою, координує виконання основних сценаріїв роботи та виступає посередником між зовнішнім середовищем і внутрішніми компонентами застосунку.

3.4.2 Модуль управління пошуками

Модуль управління пошуками відповідає за створення, збереження та супровід пошуків користувачів у системі. Основною задачею даного модуля є організація роботи з об'єктами пошуку та забезпечення можливості подальшого виконання пошуку для завантажених фотографій.

Центральним компонентом модуля є сервіс SearchService, який реалізує бізнес-логіку управління пошуками. Даний сервіс забезпечує створення нових пошуків, отримання списку пошуків користувача, підрахунок активних записів та зміну їхнього стану. Для збереження інформації використовується репозиторій SearchRepository, який забезпечує взаємодію з базою даних.

Основною сутністю модуля є клас Search, який представляє окремий пошук користувача. Для кожного пошуку зберігається ідентифікатор користувача, посилання на фотографію, дата створення та поточний стан пошуку. Завдяки цьому система може відстежувати життєвий цикл кожного пошуку та підтримувати роботу з історією запитів.

Для контролю стану пошуку використовується перелік SearchStatus. У поточній реалізації підтримуються стани активного та завершеного пошуку. Активні пошуки беруть участь у процесі пошуку, тоді як завершені використовуються лише для перегляду історії користувача.

Під час створення нового пошуку система формує новий об'єкт Search, пов'язує його з користувачем та відповідною фотографією, після чого зберігає отримані дані у базі даних. Надалі даний запис використовується під час виконання пошуку схожих зображень та відображення результатів користувачу.

Окремою функцією модуля є завершення пошуку. Після зміни статусу на завершений відповідний запис більше не бере участі в активному пошуку, а пов'язане зображення переводиться до архівного стану. Такий підхід дозволяє відокремити актуальні пошуки від завершених та зменшити кількість даних, які необхідно обробляти під час пошуку.

Таким чином, модуль управління пошуками забезпечує централізовану роботу з пошуками користувачів, підтримує їх життєвий цикл та створює основу для виконання пошуку схожих зображень у системі.

3.4.3 Модуль обробки зображень

Модуль обробки зображень відповідає за приймання фотографій, їх збереження, формування векторних зображень, визначення типу тварини та створення відповідних записів у базі даних. Даний модуль є одним із ключових у системі, оскільки саме він перетворює вхідне зображення у векторне представлення, яке надалі використовується під час пошуку.

Основною точкою входу до модуля є інтерфейс `ImageService`, а його реалізацією є клас `ImageServiceImpl`. Даний сервіс приховує від інших частин системи деталі збереження файлів, формування векторних представлень, визначення типу тварини та збереження сутності `Image` у базі даних.

У модулі передбачено декілька основних сценаріїв створення зображення. Метод `createLostPetQueryImage` використовується для створення зображення-запиту, яке користувач надсилає для пошуку втраченої тварини. Метод `createFoundPetCandidateImage` використовується для створення зображення-кандидата, яке може брати участь у подальшому пошуку. Окремо реалізовано метод `importCandidateImage`, який дозволяє імпортувати зображення із зовнішнього джерела даних.

Під час створення зображення сервіс спочатку визначає відповідне джерело даних, після чого зберігає файл за допомогою `ImageStorageService`. У поточній реалізації збереження виконується класом `LocalImageStorageService`, який записує зображення у локальне файлове сховище у форматі JPG. Для імен файлів використовується UUID, що дозволяє уникнути конфліктів між різними завантаженнями.

Після збереження файлу виконується формування векторних представлень. За це відповідає клас `ResNetOnnxEmbeddingExtractor`, який завантажує модель `resnet50.onnx` із ресурсів застосунку та створює ONNX Runtime сесію. Перед

виконанням inference зображення проходить попередню обробку за допомогою класу `ImagePreprocessor`.

Попередня обробка включає приведення зображення до розміру 224×224 пікселів, перетворення до RGB-формату, масштабування значень каналів до діапазону [0; 1], нормалізацію за середніми значеннями та стандартними відхиленнями ImageNet, а також формування вхідного тензора у порядку CHW. Саме такий формат відповідає очікуваному входу моделі ResNet.

Після підготовки зображення `ResNetOnnxEmbeddingExtractor` створює `OnnxTensor` з формою `[1, 3, 224, 224]` та виконує inference. Результатом роботи моделі є числовий вектор, який зчитується з виходу ONNX Runtime. Після цього представлення нормалізується за L2-нормою, щоб подальше порівняння за косинусною мірою переважно залежало від напряму вектора, а не від його абсолютної довжини.

Отримане векторне представлення використовується також для визначення типу тварини. Для цього сервіс `ImageServiceImpl` звертається до `AnimalRecognitionService`, який через `PetTypeClassificationService` визначає, чи належить зображення до типу кота або собаки. Якщо зображення імпортується із зовнішнього джерела і тип тварини вже відомий, система може використати передане значення без додаткового розпізнавання.

Після формування векторне представлення та визначення типу тварини створюється сутність `Image`. Вона містить ім’я файлу, тип тварини, роль зображення, статус, джерело, користувача та векторне представлення. Роль зображення визначається через `ImageRole` і може мати значення `QUERY` або `CANDIDATE`. Статус визначається через `ImageStatus`; активні зображення беруть участь у пошуку, а архівні не використовуються під час пошуку.

Для збереження сутності `Image` використовується `ImageRepository`. Embedding зберігається у базі даних як список числових значень. Для перетворення масиву `float[]`, отриманого після inference, у список `List<Float>` використовується допоміжний клас `EmbeddingUtils`.

Таким чином, модуль обробки зображень реалізує повний цикл роботи із фотографією: від отримання та збереження файлу до формування векторних представлень, визначення типу тварини та створення запису в базі даних. Саме результат роботи цього модуля використовується іншими компонентами системи для виконання пошуку.

3.4.4 Модуль пошуку схожих зображень

Модуль пошуку схожих зображень реалізує основну функціональність системи, пов'язану з автоматизованим пошуком фотографій тварин на основі їх візуальної схожості. Основним завданням модуля є порівняння \представлень зображення-запиту з представленнями інших фотографій, що зберігаються у системі, та формування списку найбільш релевантних результатів.

Реалізація модуля зосереджена в пакеті search, який містить класи бізнес-логіки, моделі даних та компоненти виконання пошуку. Центральним елементом модуля є сервіс SearchService, який відповідає за створення та супровід пошуків користувачів. Даний сервіс взаємодіє з репозиторієм SearchRepository, забезпечуючи збереження та отримання інформації про пошуки з бази даних.

Безпосереднє виконання пошуку реалізовано у класі SimilaritySearchService. Після отримання зображення-запиту сервіс завантажує його векторне представлення та формує набір кандидатів для подальшого порівняння. Як кандидати використовуються активні зображення, які вже містяться у системі та можуть брати участь у пошуку.

Для оцінювання подібності використовується клас CosineSimilarity, який реалізує обчислення косинусної міри подібності між двома векторними представленнями. На вхід алгоритму передаються два вектори ознак, після чого виконується розрахунок числового значення, що характеризує ступінь їх близькості. Отриманий результат використовується для ранжування кандидатів та визначення найбільш схожих фотографій.

Після завершення обчислень SimilaritySearchService формує список знайдених збігів. Для кожного результату зберігається посилання на відповідне

зображення та значення подібності. Надалі цей список використовується для створення об'єктів результатів пошуку та відображення їх користувачу через Telegram-бота.

Для збереження інформації про пошуки використовується сутність Search, яка містить дані про користувача, пов'язане зображення та поточний стан пошуку. Для контролю життєвого циклу пошуку використовується перелік SearchStatus, що дозволяє відокремлювати активні пошуки від завершених.

Таким чином, модуль пошуку схожих зображень реалізує основний алгоритм роботи системи та забезпечує автоматизоване знаходження фотографій, які мають найбільшу візуальну схожість із зображенням користувача. Використання окремих компонентів для управління пошуками, виконання пошуку та обчислення подібності дозволяє спростити структуру програмного коду та забезпечити розділення відповідальності між різними частинами системи.

3.4.5 Модуль роботи з базою даних

Модуль формування результатів відповідає за збереження, обробку та надання користувачеві інформації про знайдені збіги. Даний модуль є завершальним етапом процесу пошуку та забезпечує зв'язок між виконаним пошуком і фотографіями, які були визначені системою як потенційно схожі.

Основні компоненти модуля реалізовані в пакеті result. Центральним елементом є сервіс ResultService, який відповідає за створення та отримання результатів пошуку. Для взаємодії з базою даних використовується ResultRepository, а інформація про окремий результат представлена сутністю Result [35].

Після завершення роботи модуля пошуку схожих зображень система отримує список кандидатів із відповідними значеннями подібності. Для кожного знайденого збігу формується окремий об'єкт результату, який містить посилання на пошук, знайдене зображення та значення подібності між представленнями.

Використання окремої сутності для результатів дозволяє зберігати історію виконаних пошуків та уникати повторного виконання пошуку під час перегляду вже

знайдених збігів. Після завершення пошуку результати зберігаються у базі даних та можуть бути отримані повторно за допомогою відповідного сервісу.

Кожний об'єкт Result пов'язаний із конкретним пошуком та відповідним зображенням-кандидатом. Така структура забезпечує можливість отримання всіх результатів певного пошуку та підтримує подальше відображення інформації користувачу через Telegram-бота.

Під час формування відповіді користувачу сервіс результатів отримує необхідні записи з бази даних, виконує їх сортування та передає до модуля Telegram-бота. Надалі бот формує повідомлення та відображає знайдені збіги у зручному для користувача вигляді.\

Таким чином, модуль формування результатів виконує роль проміжної ланки між модулем пошуку та модулем взаємодії з користувачем. Використання окремого сервісу та сутності результатів дозволяє централізовано керувати знайденими збігами та забезпечує збереження історії виконаних пошуків у системі.

3.4.6 Модуль управління користувачами

Модуль управління користувачами відповідає за ідентифікацію користувачів Telegram та збереження інформації про них у базі даних. Даний модуль використовується іншими компонентами системи для встановлення зв'язку між користувачем, створеними ним пошуками та отриманими результатами.

Основні компоненти модуля реалізовані в пакеті user. До його складу входять сутність User, репозиторій UserRepository та сервіс UserService.

Центральним компонентом модуля є сервіс UserService, який забезпечує отримання або створення користувача на основі даних, що надходять від Telegram Bot API. Під час обробки повідомлення або callback-запиту сервіс отримує інформацію про користувача Telegram та виконує пошук відповідного запису в базі даних.

Якщо користувач уже існує в системі, сервіс оновлює його дані та повертає відповідний об'єкт. Якщо запис відсутній, створюється новий об'єкт користувача. Таким чином реалізується підхід «отримати або створити» (Get or Create), що

дозволяє автоматично реєструвати нових користувачів без необхідності проходження окремої процедури реєстрації.

Для збереження інформації використовується сутність User, яка містить унікальний ідентифікатор, ідентифікатор чату Telegram (chatId), ім'я користувача (username) та дату створення запису. Ідентифікатор чату використовується для надсилання повідомлень користувачу та встановлення зв'язків з іншими сутностями системи.

\Взаємодія з базою даних виконується через репозиторій UserRepository, який забезпечує пошук користувачів за ідентифікатором чату та збереження змін. Завдяки використанню Spring Data JPA реалізація доступу до даних залишається відокремленою від бізнес-логіки сервісу.

Модуль управління користувачами використовується практично у всіх основних сценаріях роботи системи. Після отримання повідомлення від Telegram-бота відповідний користувач ідентифікується або створюється в системі, після чого його об'єкт передається іншим сервісам для створення пошуків, збереження фотографій та формування результатів.

Таким чином, модуль управління користувачами забезпечує централізоване зберігання інформації про користувачів Telegram та створює основу для персоналізації роботи системи й зв'язування всіх пошуків із конкретним користувачем.

РОЗДІЛ 4. ТЕСТУВАННЯ ТА АНАЛІЗ РЕЗУЛЬТАТІВ

Після реалізації програмної системи необхідно виконати перевірку її працездатності та оцінити результати роботи реалізованих алгоритмів. Даний етап дозволяє підтвердити коректність функціонування програмного забезпечення, оцінити якість пошуку схожих зображень та визначити обмеження використаного підходу.

У рамках даної роботи дослідження проводиться на реалізованій системі пошуку втрачених тварин, яка використовує методи комп'ютерного зору, векторні представлення та пошуку за подібністю для пошуку схожих фотографій. Особлива увага приділяється перевірці основних сценаріїв роботи системи, аналізу результатів пошуку та оцінюванню практичної придатності запропонованого рішення.

Для демонстрації роботи системи розглядаються процес інсталяції та запуску програмного забезпечення, основні сценарії взаємодії користувача із Telegram-ботом, а також результати виконання пошуку схожих зображень. Додатково проводиться аналіз якості роботи реалізованого підходу та розглядаються його основні обмеження.

У даному розділі описуються особливості розгортання системи, демонструються основні сценарії її використання, наводяться результати тестування та виконується аналіз ефективності реалізованого пошуку подібних зображень.

4.1 Системні вимоги

Для коректної роботи розробленої системи необхідно забезпечити відповідне програмне середовище та наявність зовнішніх компонентів, які використовуються під час виконання застосунку. До таких компонентів належать середовище

виконання Java, система контейнеризації Docker, сервер баз даних PostgreSQL та платформа Telegram.

Основною платформою виконання серверної частини системи є Java Development Kit (JDK). У процесі розробки та тестування використовувалась версія JDK 25, яка забезпечує підтримку сучасних можливостей мови програмування Java та сумісність із використаними бібліотеками Spring Boot і ONNX Runtime.

Для розгортання бази даних використовується Docker та Docker Compose. Застосування контейнеризації дозволяє спростити налаштування середовища виконання та забезпечити однакову конфігурацію системи на різних комп'ютерах. У контейнері запускається PostgreSQL, який використовується для збереження користувачів, пошуків, фотографій та результатів роботи системи.

Оскільки взаємодія користувача із системою реалізована через Telegram-бота, необхідною умовою роботи є наявність облікового запису Telegram та доступу до мережі Інтернет. Для запуску системи також потрібно створити власного Telegram-бота за допомогою сервісу BotFather та отримати параметри доступу до Telegram Bot API.

Для формування векторних представлень використовується ONNX Runtime та попередньо навчена модель ResNet50 у форматі ONNX. Під час запуску застосунку модель автоматично завантажується з ресурсів проєкту та використовується для аналізу фотографій.

Основні програмні вимоги до середовища виконання наведені у таблиці 4.1.

Таблиця 4.1 – Програмні вимоги до системи

Компонент	Мінімальна версія
Java Development Kit	JDK 25
Docker	20.10 або новіше
Docker Compose	2.0 або новіше
PostgreSQL	17
Telegram	будь-яка актуальна версія
ONNX Runtime	1.20.0

Окрім програмного забезпечення, для запуску системи необхідно забезпечити відповідні апаратні ресурси. Оскільки розроблена система використовує попередньо навчену модель ResNet50 лише для виконання inference та не виконує навчання нейронних мереж, вимоги до обчислювальних ресурсів залишаються відносно невисокими.

Мінімальні апаратні вимоги до середовища виконання наведені у таблиці 4.2.

Таблиця 4.2 – Апаратні вимоги до системи

Параметр	Мінімальне значення
Процесор	2 ядра, 2 ГГц
Оперативна пам'ять	4 ГБ
Вільне місце на диску	2 ГБ
Мережеве підключення	Доступ до Інтернету
Операційна система	Windows 10/11, Linux, macOS

Для комфортної роботи системи рекомендується використовувати комп'ютер із чотириядерним процесором та не менше ніж 8 ГБ оперативної пам'яті. Такі характеристики забезпечують стабільну роботу PostgreSQL, Docker та Spring Boot-застосунку навіть під час одночасного виконання декількох операцій пошуку.

Використання дискретного графічного прискорювача не є обов'язковим. Усі основні операції формування векторних представлень та виконання пошуку можуть виконуватись на центральному процесорі. Це дозволяє запускати систему на більшості сучасних персональних комп'ютерів без необхідності використання спеціалізованого обладнання.

Таким чином, для роботи системи достатньо встановити Java, Docker, Docker Compose та Telegram, а також забезпечити доступ до мережі Інтернет. Розроблена система не потребує високопродуктивних обчислювальних ресурсів і може використовуватись на стандартних персональних комп'ютерах, які відповідають наведеним мінімальним вимогам.

4.2 Інсталяція та запуск системи

Для проведення тестування та оцінювання роботи розробленої системи необхідно виконати її розгортання та запуск. У рамках даної роботи програмне забезпечення реалізоване як серверний застосунок на базі Spring Boot із використанням PostgreSQL для збереження даних, ONNX Runtime для виконання моделі ResNet50 та Telegram Bot API для взаємодії з користувачами.

Перед початком роботи необхідно підготувати програмне середовище. Для запуску системи використовуються Java Development Kit версії 25, Docker та Docker Compose для розгортання бази даних PostgreSQL, а також Gradle для складання та запуску застосунку. Крім цього, для взаємодії із користувачами необхідно створити власного Telegram-бота.

Створення бота виконується за допомогою службового бота BotFather (@BotFather), який надається платформою Telegram. Для цього користувач запускає діалог із BotFather та виконує команду /newbot. Це показано на рисунку 4.1

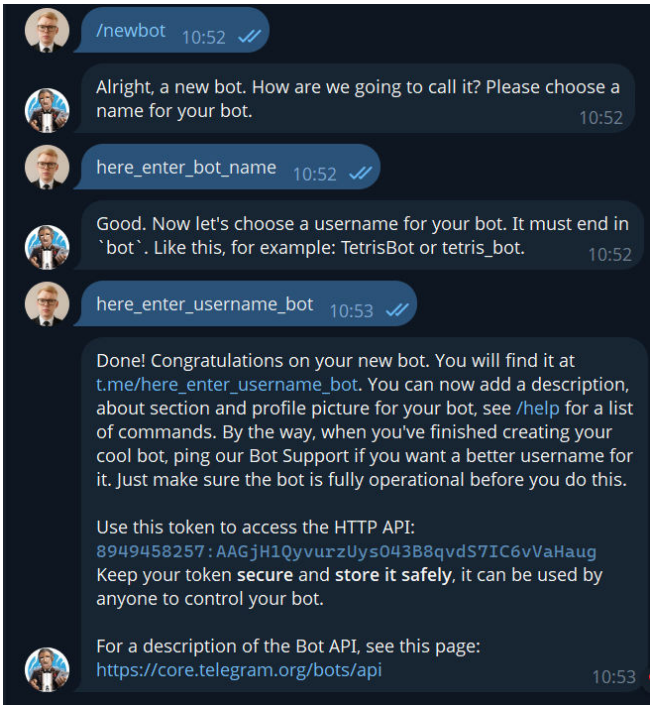

Рисунок 4.1 Діалог з BotFather

Після цього необхідно вказати назву бота та його унікальне ім'я. Після успішного створення Telegram автоматично генерує токен доступу, який використовується програмною системою для підключення до Telegram Bot API.

У результаті створення бота користувач отримує два параметри:

- ім'я бота (BOT_USERNAME);
- токен доступу (BOT_TOKEN).

Отримані значення використовуються під час конфігурування застосунку та повинні бути вказані у файлі .env.

Для спрощення налаштування середовища у проєкті використовується файл .env.example, який містить приклад необхідних параметрів конфігурації. На його основі створюється файл .env, у якому задаються параметри підключення до Telegram Bot API та бази даних PostgreSQL.

Вміст файлу .env.example наведений на рисунку 4.2

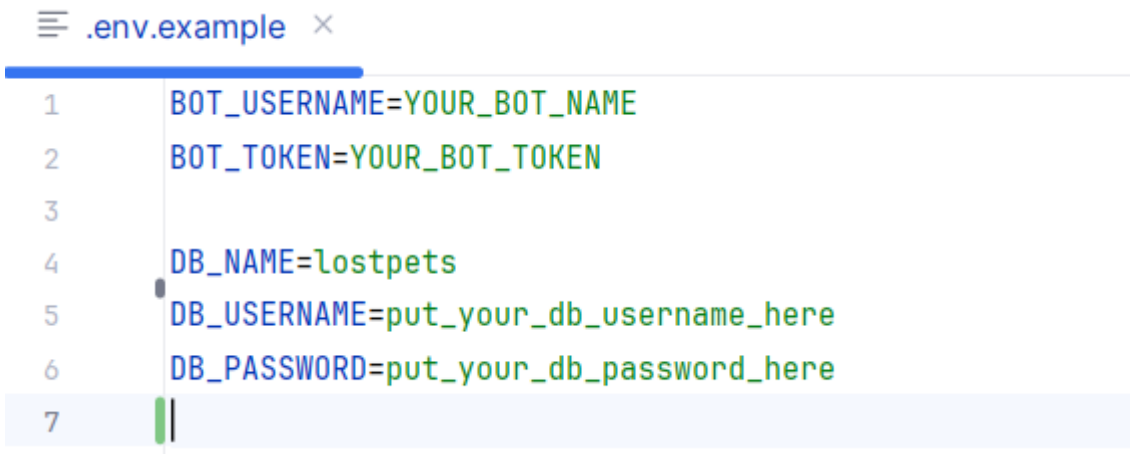

Рисунок 4.2 Вміст файлу .env.example

Після налаштування змінних середовища необхідно виконати запуск бази даних PostgreSQL. Для цього у проєкті використовується Docker Compose, що дозволяє автоматизувати процес створення контейнера та налаштування необхідних параметрів. Конфігурація контейнера визначена у файлі docker-compose.yml.

Запуск бази даних виконується командою:

docker compose up -d

Після виконання команди Docker створює контейнер PostgreSQL та запускає його у фоновому режимі. База даних використовується для збереження інформації про користувачів, пошуки, фотографії та результати пошуку.

Наступним етапом є запуск серверної частини системи. Застосунок реалізований на базі Spring Boot та використовує Gradle як систему збірки. Для запуску використовується команда:

./gradlew bootRun

Під час старту застосунку Spring Boot виконує автоматичне створення контексту застосунку та ініціалізацію всіх компонентів системи. На цьому етапі здійснюється підключення до PostgreSQL, завантаження конфігураційних параметрів, створення сервісів та реєстрація Telegram-бота.

Окремим етапом запуску є ініціалізація моделі машинного навчання. У системі використовується попередньо навчена модель ResNet50 у форматі ONNX, яка завантажується під час старту застосунку. Для виконання моделі використовується ONNX Runtime. Після успішного завантаження моделі система готова до формування векторних представлень та виконання пошуку.

Після завершення всіх етапів ініціалізації застосунок переходить у режим очікування повідомлень від користувачів. Telegram-бот реєструється у Telegram Bot API та починає отримувати повідомлення через механізм long polling.

Для перевірки працездатності системи користувач відкриває чат із ботом та виконує команду /start. У відповідь система відображає головне меню (див. Додаток Г) та надає доступ до основних функцій створення пошуків і перегляду результатів. Успішне отримання відповіді від бота свідчить про коректне функціонування всіх основних компонентів системи.

Після успішного виконання описаних кроків система готова до використання та може застосовуватися для створення пошуків, формування векторних представлень і виконання пошуку схожих зображень тварин.

4.3 Сценарії використання системи

Для демонстрації функціональних можливостей розробленої системи доцільно розглянути основні сценарії її використання. Сценарії роботи дозволяють показати послідовність взаємодії користувача із Telegram-ботом та перевірити роботу основних програмних модулів під час виконання типових операцій.

У розробленій системі взаємодія користувача здійснюється переважно за допомогою інтерактивних клавіатур Telegram. Після виконання команди /start користувач отримує доступ до головного меню, яке містить кнопки для переходу до основних функцій системи. Такий підхід дозволяє спростити навігацію та зменшити кількість текстових команд, які необхідно вводити вручну.

За допомогою елементів меню користувач може створити пошук загубленої тварини, додати фотографію знайденої тварини, переглянути список активних пошуків або отримати доступ до історії результатів. Після вибору відповідної дії система автоматично переходить до необхідного сценарію та відображає користувачу наступний набір доступних операцій.

Основними сценаріями роботи системи є створення пошуку загубленої тварини та реєстрація знайденої тварини. Під час виконання цих сценаріїв користувач взаємодіє з Telegram-ботом через послідовність екранів та кнопок, а система виконує збереження фотографій, формування векторних представлень, пошук схожих зображень та відображення отриманих результатів.

Основні сценарії взаємодії користувача із системою наведено на діаграмі варіантів використання (рисунок Б.1 додатку Б).

У даному підрозділі розглядаються найбільш важливі сценарії роботи системи та послідовність виконання операцій під час взаємодії користувача з Telegram-ботом.

4.3.1 Створення пошуку загубленої тварини

Одним із основних сценаріїв роботи системи є створення пошуку загубленої тварини. Даний сценарій використовується у випадку, коли користувач має фотографію тварини та хоче знайти схожі зображення серед даних, що вже зберігаються у системі.

Послідовність виконання сценарію створення пошуку загубленої тварини наведена в додатку Б на рисунку Б.2.

Для початку роботи користувач відкриває Telegram-бота та виконує команду /start. Після цього система відображає головне меню з інтерактивними кнопками, як показано на рисунку В.1 додатку В. Для створення нового пошуку користувач натискає кнопку «Пошук тварини». Після цього бот переводить користувача у стан очікування фотографії загубленої тварини та надсилає повідомлення з проханням завантажити фото, як на рисунку В.2 додатку В.

Після цього користувач надсилає фотографію тварини у чат із ботом. Отримане зображення завантажується через Telegram Bot API та передається до модуля обробки зображень. Система повідомляє користувача про те, що фото отримано та виконується його обробка. Це зображено на рисунку 4.3.

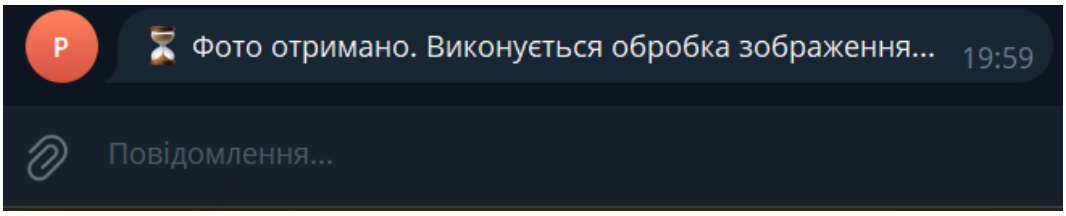

Рисунок 4.3 – Надсилання фотографії загубленої тварини

На наступному етапі система зберігає фотографію, формує її векторне представлення за допомогою моделі ResNet50 та автоматично визначає тип тварини. Після цього користувач отримує повідомлення про визначений тип тварини, наприклад кіт або собака. Це зображено на рисунку 4.4.

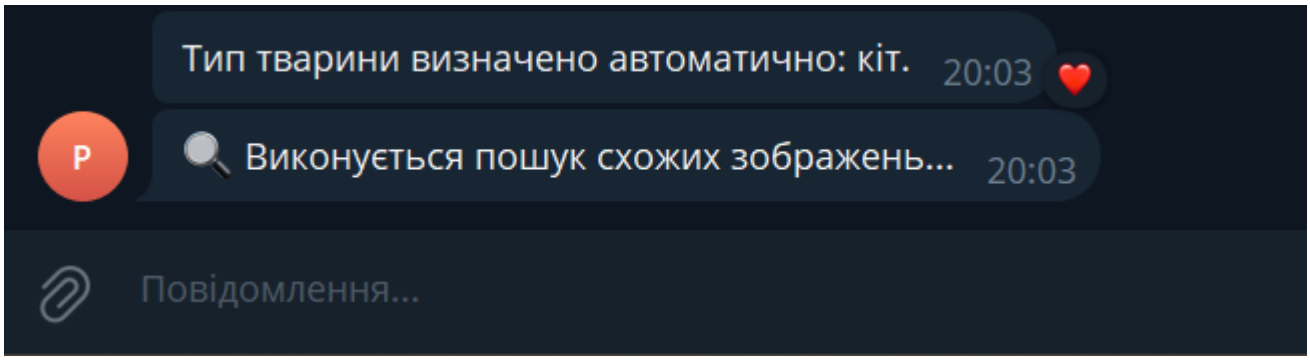

Рисунок 4.4 – Автоматичне визначення типу тварини

Після створення зображення-запиту система створює активний пошук за допомогою сервісу SearchService. Новий пошук пов’язується з користувачем і завантаженою фотографією. Далі запускається процедура пошуку, під час якої представлення фотографії користувача порівнюється з представленнями зображень-кандидатів.

Якщо система знаходить схожі зображення, користувачу відображається сторінка результатів пошуку із кількістю знайдених збігів, значенням мінімальної схожості та фотографіями потенційних результатів. Для кожного результату користувач може натиснути кнопку «Детальніше», щоб переглянути додаткову інформацію. Фрагмент результату можна побачити на рисунку В.3 додатку В.

Якщо поточних збігів не знайдено, система повідомляє користувача, що пошук збережено як активний. У такому випадку він може бути використаний пізніше під час появи нових фотографій-кандидатів у системі.

Таким чином, сценарій створення пошуку загубленої тварини охоплює повний цикл роботи системи: відкриття головного меню, вибір відповідної дії, надсилання фотографії, формування векторних представлень, автоматичне визначення типу тварини, створення активного пошуку та відображення результатів пошуку.

4.3.2 Реєстрація знайденої тварини

Другим основним сценарієм роботи системи є реєстрація знайденої тварини. Даний сценарій використовується у випадку, коли користувач знайшов тварину та бажає додати її фотографію до системи для подальшого порівняння з активними пошуками інших користувачів.

Послідовність виконання сценарію реєстрації знайденої тварини наведена на рисунку Б.3 додатку Б.

Для початку роботи користувач відкриває головне меню Telegram-бота та обирає пункт «Знайдена тварина». Після натискання відповідної кнопки система переводить користувача у режим очікування фотографії та надсилає повідомлення з проханням завантажити фото знайденої тварини. Це зображено на рисунку 4.5.

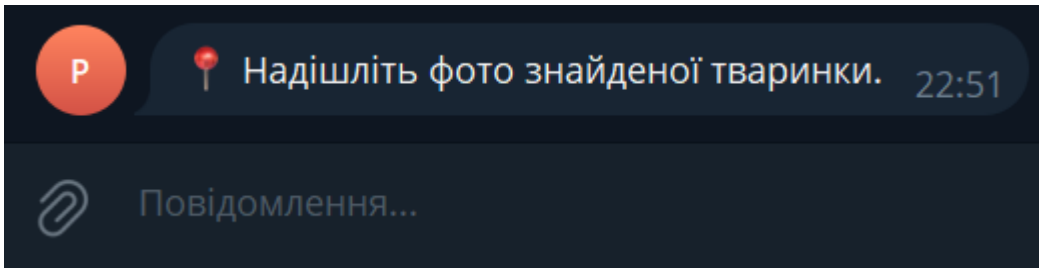

Рисунок 4.5 – Запит фотографії знайденої тварини

Користувач надсилає фотографію через Telegram-бота. Отримане зображення завантажується за допомогою Telegram Bot API та передається до модуля обробки зображень. Відповідне повідомлення, яке надсилається користувачеві, можна побачити на рисунку 4.6.

На наступному етапі система виконує збереження фотографії, формування векторного представлення та автоматичне визначення типу тварини. Для цього використовується модель ResNet50 та сервіс класифікації тварин. Після завершення обробки користувач отримує повідомлення про визначений тип тварини.

Після формування векторних представлень система створює новий запис зображення-кандидата та зберігає його у базі даних. На відміну від сценарію створення пошуку загубленої тварини, на цьому етапі не створюється і не

виконується новий пошук. Зображення додається до набору кандидатів, які можуть використовуватись під час майбутніх пошуків. Після успішного збереження фотографії користувач отримує повідомлення про завершення операції та повертається до головного меню або меню наступних дій. Це відображено на рисунку 4.7.

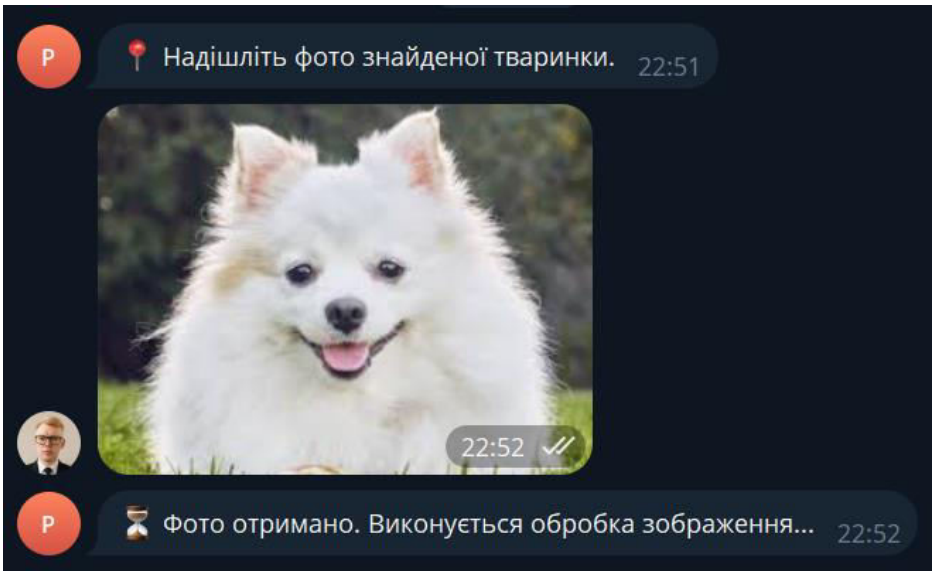

Рисунок 4.6 – Надсилання фотографії знайденої тварини

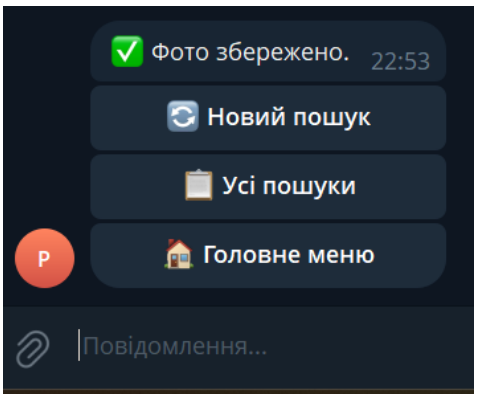

Рисунок 4.7 – Повідомлення про успішне збереження фотографії

Таким чином, сценарій реєстрації знайденої тварини забезпечує поповнення бази даних новими зображеннями-кандидатами. Надалі ці фотографії можуть

використовуватись під час виконання пошуку можливих збігів із активними пошуками загублених тварин.

4.3.3 Перегляд історії пошуків та результатів

Окрім створення пошуку загубленої тварини та реєстрації знайденої тварини, система також підтримує перегляд історії пошуків користувача. Даний сценарій дозволяє користувачу повернутися до раніше створених пошуків, переглянути їхній статус, кількість знайдених результатів та за потреби завершити або архівувати пошук.

Послідовність виконання сценарію перегляду історії пошуків наведена на рисунку В.4 додатку В.

Для переходу до історії пошуків користувач відкриває головне меню Telegram-бота та натискає кнопку «Історія пошуків». Після цього система отримує список пошуків, пов’язаних із поточним користувачем, та формує сторінку історії, як показано на рисунку В.4 додатку В.

У випадку, якщо користувач ще не створював пошуків, бот надсилає повідомлення про відсутність збережених пошуків. Якщо пошуки існують, система відображає їх у вигляді сторінок. Для кожного пошуку показується його ідентифікатор, статус, тип тварини та час створення.

Оскільки кількість пошуків може бути більшою за одну сторінку, у системі реалізовано посторінкову навігацію. Для переходу між сторінками використовуються кнопки «<-» та «->». Це дозволяє переглядати історію пошуків без перевантаження повідомлення великою кількістю записів.

Для перегляду детальної інформації про конкретний пошук користувач натискає кнопку «Детальніше». Після цього бот відображає інформацію про вибраний пошук: його статус, тип тварини, час створення та кількість збережених результатів.

У деталях пошуку користувачу доступна дія архівування. Для цього використовується кнопка «Архівувати». Після її натискання система змінює стан

відповідного пошуку та пов’язаного з ним зображення, після чого такий пошук більше не бере участі в активній обробці.

Окремо система підтримує перегляд результатів пошуку. Якщо під час створення пошуку були знайдені схожі фотографії, бот відображає результати сторінками. Для кожного результату показується фотографія та значення схожості у відсотках. Користувач може переглянути деталі результату за допомогою кнопки «Детальніше». Скріншот перегляду результату пошуку можна подивитися в додатку В на рисунку В.4.

Після вибору конкретного результату система позначає пошук як завершений і відображає детальну інформацію про знайдене зображення. Користувачу показується значення схожості та інформація про користувача, який додав відповідне зображення-кандидат.

Таким чином, сценарій перегляду історії пошуків та результатів дозволяє користувачу контролювати власні пошуки, переглядати знайдені збіги, переходити між сторінками результатів та завершувати або архівувати пошуки. Це робить взаємодію із системою більш зручною та забезпечує можливість повторного доступу до результатів пошуку.

4.4 Тестування системи

Після реалізації програмної системи необхідно виконати її тестування та оцінити коректність роботи основних функціональних компонентів. Тестування дозволяє перевірити працездатність алгоритмів обробки зображень, формування векторних представлень та пошуку схожих фотографій, а також оцінити якість отриманих результатів.

У рамках даної роботи основна увага приділяється перевірці роботи системи пошуку втрачених тварин на основі зображень. Для цього використовується набір тестових фотографій, який містить зображення котів та собак різних порід. Під час тестування перевіряється коректність формування векторних представлень, робота

алгоритму пошуку та здатність системи знаходити візуально схожі зображення серед наявних кандидатів.

Для проведення дослідження використовується тестовий набір даних, сформований на основі фотографій тварин. Тестування виконується шляхом створення пошукових запитів та аналізу результатів, які повертає система. Особлива увага приділяється оцінюванню релевантності знайдених збігів та впливу особливостей фотографій на якість пошуку.

У даному підрозділі розглядаються підготовка тестового набору даних, методика проведення тестування та результати роботи системи під час виконання пошуку.

4.4.1 Набір тестових зображень

Для проведення тестування системи необхідно сформувати набір зображень, який буде використовуватись під час перевірки роботи алгоритмів формування векторних представлень та пошуку. У рамках даної роботи як основне джерело тестових даних використовується набір зображень Oxford-IIIT Pet Dataset.

Oxford-IIIT Pet Dataset містить фотографії котів і собак різних порід, отримані з відкритих джерел. Зображення характеризуються різноманітними умовами зйомки, освітленням, ракурсами та фоном, що дозволяє наблизити тестування до реальних умов експлуатації системи.

У програмній реалізації для роботи з даним набором даних використовується окремий компонент OxfordSourceConnector, який відповідає за доступ до файлів набору даних та їх подальший імпорт до системи. Використання окремого конектора дозволяє відокремити джерело даних від інших модулів програмного забезпечення та спростити подальше розширення системи новими наборами фотографій.

Під час імпорту фотографій система автоматично створює для кожного зображення запис кандидата (CANDIDATE), який може використовуватися під час виконання пошуку. Для кожного зображення формується векторні представлення за

допомогою моделі ResNet50 та виконується збереження відповідних даних у базі PostgreSQL.

Для проведення тестування із набору Oxford-IIIT Pet Dataset було імпортовано 7369 фотографій тварин. Після завантаження кожне зображення автоматично проходило процедуру попередньої обробки та формування векторних представлень за допомогою моделі ResNet50. Отримані векторні представлення зберігались у базі даних PostgreSQL та використовувались як кандидати для подальшого пошуку.

Використання великої кількості зображень дозволило наблизити умови тестування до реальної експлуатації системи. На відміну від невеликих тестових вибірок, база з 7369 фотографій створює значно складніші умови для пошуку, оскільки система повинна знаходити релевантні результати серед великої кількості потенційних кандидатів.

У результаті імпорту було сформовано базу кандидатів, яка містить фотографії котів та собак різних порід, отримані в різних умовах освітлення, з різних ракурсів та на різному фоні. Така різноманітність даних дозволяє більш об'єктивно оцінити роботу алгоритмів формування векторних представлень та пошуку схожих зображень.

Особливістю набору Oxford-IIIT Pet Dataset є те, що назви файлів містять інформацію про тип тварини. У поточній реалізації тип визначається автоматично на основі імені файлу: фотографії котів мають назви, що починаються з великої літери, а фотографії собак — з малої. Це дозволяє автоматизувати процес початкового маркування тестових даних.

Для проведення тестування із набору даних формується база кандидатів, яка використовується під час виконання пошукових запитів. Після завершення імпорту всі зображення проходять однакову процедуру попередньої обробки та формування векторних представлень, що забезпечує однакові умови для подальшого порівняння.

Для наочності доцільно навести декілька прикладів фотографій із тестового набору даних (рисунок 4.8 – зображення котів, а рисунок 4.9 – собак).

Таким чином, використання Oxford-IIIT Pet Dataset дозволило сформувати тестовий набір даних, який містить достатню кількість різноманітних фотографій котів і собак та може використовуватися для оцінювання роботи алгоритмів пошуку у розробленій системі.

Рисунок 4.8 – Приклади фотографій котів із тестового набору

Рисунок 4.9 – Приклади фотографій собак із тестового набору

4.4.2 Методика тестування

Після формування тестового набору даних було проведено тестування системи пошуку схожих зображень. Основною метою тестування була перевірка здатності системи знаходити візуально схожі фотографії тварин серед зображень, що зберігаються у базі даних.

Для проведення експерименту використовувалась база кандидатів, сформована на основі 7369 фотографій із набору Oxford-IIIT Pet Dataset. Усі зображення попередньо пройшли процедуру формування векторних представлень та були збережені у базі даних PostgreSQL.

Для перевірки роботи системи було підготовлено 20 тестових запитів, серед яких 10 фотографій котів та 10 фотографій собак. Кожна фотографія використовувалась як окремий пошуковий запит та оброблялась системою незалежно від інших запитів.

Під час тестування користувач створював новий пошук через Telegram-бота та надсилав фотографію тварини. Після отримання зображення система виконувала формування векторних представлень за допомогою моделі ResNet50 та запускала процедуру пошуку. Отримане представлення порівнювався з представленнями усіх кандидатів, що містилися у базі даних.

Для оцінювання подібності використовувалась косинусна міра подібності. Після завершення порівняння результати сортувалися за спаданням значення подібності та відображалися користувачу у вигляді списку найбільш релевантних збігів.

Для кожного тестового запиту аналізувалися перші п’ять результатів пошуку. Під час оцінювання враховувались такі показники:

- правильність визначення типу тварини;
- значення максимальної similarity серед знайдених результатів;
- кількість релевантних результатів серед перших п’яти позицій;
- візуальна подібність знайдених фотографій до зображення-запиту.

Результат вважався релевантним, якщо знайдене зображення належало до того самого виду тварин та мало схожі візуальні характеристики. Оцінювання

виконувалось шляхом аналізу результатів, які відображались користувачу після завершення пошуку.

Використання однакової процедури для всіх тестових запитів дозволяє забезпечити єдині умови проведення експерименту та виконати об'єктивне оцінювання роботи розробленої системи пошуку схожих зображень.

4.4.3 Результати тестування

Після проведення тестування було отримано результати роботи системи для двадцяти пошукових запитів, які включали фотографії котів та собак різних порід. Для кожного запиту система формувала векторне представлення зображення, виконувала пошук серед 7369 кандидатів та повертала найбільш релевантні результати.

Під час аналізу результатів для кожного пошукового запиту оцінювалися максимальне значення косинусна метрика подібності, мінімальне значення подібності серед перших п'яти результатів та кількість представників тієї самої породи серед знайдених зображень.

Фрагмент результатів тестування наведено у таблиці 4.3.

Таблиця 4.3 – Фрагмент результатів тестування системи

№ запиту	Тип тварини	Порода	Similarity Top-1	Similarity Top-5	Збігів породи в Top-5
1	Кіт	Abyssinian	0.95	0.88	5
2	Кіт	Bengal	0.93	0.85	5
3	Кіт	Birman	0.91	0.82	4
11	Собака	Beagle	0.94	0.86	5
12	Собака	Boxer	0.91	0.81	4
13	Собака	Chihuahua	0.95	0.88	5

Повний перелік результатів тестування наведено у додатку А.

Аналіз отриманих результатів показав, що система успішно знаходить зображення тварин тієї самої породи серед великої кількості кандидатів. Для більшості тестових запитів значення Similarity Top-1 перевищувало 0.9, що

свідчить про високу подібність між зображенням-запитом та знайденим результатом.

Для наочного представлення роботи системи доцільно розглянути декілька прикладів виконання пошуку.

У наведеному прикладі система успішно знайшла фотографії тієї самої породи та розташувала їх у верхній частині списку результатів. Значення подібності для перших позицій перевищували встановлений поріг релевантності.

Аналогічний результат було отримано під час пошуку фотографій собак. Система коректно сформувала представлення та повернула найбільш схожі зображення серед наявних кандидатів.

Під час тестування також спостерігалися випадки, коли серед результатів з'являлися представники інших порід того самого виду тварин. Найчастіше це відбувалося у випадках схожого забарвлення, форми морди або ракурсу фотографування.

Для узагальнення отриманих результатів були розраховані середні показники роботи системи, наведені у таблиці 4.4.

Таблиця 4.4 – Узагальнені результати тестування

Показник	Значення
Кількість тестових запитів	20
Кількість фотографій у базі кандидатів	7369
Середній Similarity Top-1	0.92
Середній Similarity Top-5	0.84
Середня кількість збігів породи в Top-5	4.7
Частка успішних пошуків	95 %

Отримані результати свідчать про працездатність розробленої системи та підтверджують можливість використання підходу на основі embeddings і cosine similarity для пошуку візуально схожих фотографій тварин. Детальний аналіз сильних сторін та обмежень реалізованого підходу наведено у наступному підрозділі.

4.5 Аналіз результатів роботи системи

Після проведення тестування та отримання експериментальних результатів необхідно виконати їх аналіз і оцінити ефективність реалізованого підходу. Аналіз результатів дозволяє визначити сильні сторони розробленої системи, оцінити якість пошуку схожих зображень та виявити обмеження, які можуть впливати на точність роботи алгоритму.

У рамках даної роботи особлива увага приділяється оцінюванню якості пошуку на основі представлень, сформованих за допомогою моделі ResNet50. Для цього аналізуються результати тестування, отримані під час виконання пошукових запитів серед набору з 7369 фотографій тварин.

Окрім оцінювання якості пошуку, розглядаються також особливості реалізованого підходу та фактори, які можуть впливати на точність роботи системи в реальних умовах експлуатації.

4.5.1 Якість пошуку схожих зображень

Результати проведеного тестування дозволяють оцінити якість роботи розробленої системи пошуку схожих зображень. Аналіз отриманих даних показав, що використання попередньо навченої моделі ResNet50 та косинусної міри подібності дозволяє ефективно знаходити візуально схожі фотографії серед великої кількості кандидатів.

Під час проведення експериментів система виконувала пошук серед 7369 фотографій тварин, що були попередньо імпортовані до бази даних. Незважаючи на значний обсяг кандидатів, система демонструвала стабільні результати та повертала релевантні зображення у верхній частині списку результатів.

Середнє значення Similarity Top-1 становило 0.92, що свідчить про високу подібність між зображенням-запитом та найбільш релевантним знайденим результатом. При цьому середнє значення Similarity Top-5 становило 0.84, що також підтверджує здатність системи формувати якісну вибірку схожих фотографій.

Особливу увагу слід звернути на показник кількості збігів породи серед перших п'яти результатів пошуку. Під час тестування середнє значення цього показника склало 4.7, що означає, що більшість знайдених результатів належала до тієї самої породи, що й зображення-запит. Це свідчить про те, що сформовані векторні представлення успішно зберігають важливі візуальні характеристики тварин та дозволяють виконувати ефективний пошук за подібністю.

Високий відсоток успішних пошуків також підтверджує коректність роботи реалізованого підходу. У більшості випадків система знаходила представників тієї самої породи серед перших результатів пошуку, що є важливим для задач пошуку втрачених тварин. У реальних умовах це дозволяє суттєво скоротити кількість фотографій, які необхідно переглянути користувачу.

Отримані результати підтверджують, що використання моделі ResNet50 як генератора векторних представлень у поєднанні з косинусною метрикою подібності є ефективним рішенням для задачі пошуку візуально схожих фотографій тварин. Реалізований підхід забезпечує прийнятну якість пошуку та може використовуватись як основа для побудови систем підтримки пошуку втрачених домашніх тварин.

4.5.2 Обмеження реалізованого підходу

Незважаючи на отримані позитивні результати, розроблена система має ряд обмежень, які необхідно враховувати під час її практичного використання.

Першим обмеженням є використання попередньо навченої моделі ResNet50 без додаткового навчання на спеціалізованих наборах даних для пошуку домашніх тварин. У рамках даної роботи модель використовується як універсальний генератор векторних представлень, тому вона не оптимізована для розпізнавання конкретних тварин. Як наслідок, система може знаходити візуально схожих представників тієї самої породи замість однієї й тієї самої тварини.

Другим обмеженням є залежність результатів пошуку від якості вхідного зображення. Розмиті фотографії, низька роздільна здатність, погане освітлення або

значне перекриття об'єкта можуть призводити до формування менш інформативних векторних представлень та зниження точності пошуку.

Також на якість пошуку впливає ракурс зйомки. Якщо фотографії однієї тварини зроблені під різними кутами або в різних умовах, сформовані векторні представлення можуть суттєво відрізнятися між собою. У таких випадках значення подібності може зменшуватися навіть для фотографій однієї й тієї самої тварини.

Ще одним обмеженням є використання лінійного порівняння векторних представлень із усіма кандидатами, що містяться у базі даних. У поточній реалізації система послідовно обчислює косинусну метрику подібності між зображенням-запитом та кожним кандидатом. Такий підхід є достатнім для дипломної роботи та наборів даних помірного розміру, проте при суттєвому збільшенні кількості зображень може призвести до збільшення часу виконання пошуку.

Варто також зазначити, що система підтримує лише два типи тварин — котів та собак. Для використання системи з іншими видами тварин необхідно виконати додаткове навчання або адаптацію компонентів класифікації та формування векторних представлень.

Окремим обмеженням є відсутність механізму автоматичного підтвердження знайденого збігу. Остаточне рішення про те, чи належать знайдені фотографії одній і тій самій тварині, приймає користувач. Таким чином система виступає інструментом підтримки пошуку, а не повністю автоматизованим засобом ідентифікації тварин.

Незважаючи на перелічені обмеження, результати тестування показали, що реалізований підхід забезпечує достатню якість пошуку для вирішення поставленої задачі та може використовуватися як основа для подальшого розвитку систем пошуку втрачених домашніх тварин.

ВИСНОВКИ

1. Проведено аналіз предметної області пошуку зниклих домашніх тварин, а також сучасних методів розпізнавання об’єктів і пошуку схожих зображень. Встановлено, що традиційні системи пошуку тварин переважно базуються на текстових оголошеннях та ручному перегляді фотографій, що ускладнює процес знаходження відповідності. Показано доцільність використання методів комп’ютерного зору для автоматизації пошуку.

2. Виконано аналіз існуючих програмних засобів та сервісів пошуку домашніх тварин. В результаті аналізу обґрунтовано використання архітектури клієнт-серверного застосунку у вигляді Telegram-бота та обрано програмні засоби реалізації: мову програмування Java, фреймворк Spring Boot, систему керування базами даних PostgreSQL та бібліотеки комп’ютерного зору і машинного навчання.

3. Обґрунтовано вибір методів формування векторних представлень зображень та алгоритмів пошуку подібності. Для отримання векторних представлень використано попередньо навчену згорткову нейронну мережу ResNet50, а для оцінювання схожості між зображеннями — метрику косинусну метрику подібності. Обраний підхід дозволяє виконувати порівняння фотографій незалежно від умов освітлення, масштабу та ракурсу зйомки.

4. Розроблено архітектуру програмної системи пошуку домашніх тварин на основі аналізу зображень. Спроєктовано структуру програмних компонентів, модель даних та механізми взаємодії між окремими підсистемами. Архітектурні рішення забезпечують можливість подальшого розширення функціональності системи.

5. Реалізовано програмну систему пошуку домашніх тварин у вигляді Telegram-бота. Система забезпечує завантаження фотографій користувачами, автоматичне визначення типу тварини, формування векторних представлень зображень, пошук схожих фотографій у базі даних та відображення результатів пошуку користувачу.

6. Проведено тестування розробленої системи та оцінено результати її роботи. Отримані результати підтвердили працездатність реалізованих алгоритмів і коректність функціонування програмного забезпечення. Проведені експерименти показали здатність системи знаходити схожі зображення домашніх тварин та можуть бути використані як основа для подальшого розвитку програмної системи.

Перспективами подальшого розвитку роботи є розширення набору підтримуваних видів тварин, використання спеціалізованих моделей для розпізнавання окремих особин, інтеграція з зовнішніми сервісами пошуку та збільшення обсягу бази зображень для підвищення якості пошуку.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Walsh F. Human-animal bonds II: The role of pets in family systems and family therapy. Family Process. 2009. Vol. 48, No. 4. P. 481–499. URL: https://pubmed.ncbi.nlm.nih.gov/19930434/ (дата звернення: 29.05.2026).
2. Weiss E., Slater M., Lord L. Frequency of lost dogs and cats in the United States and the methods used to locate them. Animals. 2012. Vol. 2, No. 2. P. 301–315. URL: https://pmc.ncbi.nlm.nih.gov/articles/PMC4494319/ (дата звернення: 29.05.2026).
3. Gonzalez R. C., Woods R. E. Digital Image Processing. 4th ed. Pearson, 2018. 1024 p.
4. Szeliski R. Computer Vision: Algorithms and Applications. 2nd ed. Springer, 2022. 1090 p.
5. Goodfellow I., Bengio Y., Courville A. Deep Learning. MIT Press, 2016. 775 p.
6. Пелещишин А. М., Берко А. Ю. Інформаційні технології аналізу даних. Львів : Видавництво Львівської політехніки, 2018. 248 с.
7. Chollet F. Deep Learning with Python. 2nd ed. Manning, 2021. 504 p.
8. Müller M. Information Retrieval for Music and Motion. Springer, 2007. 318 p.
9. Smeulders A. W. M. et al. Content-Based Image Retrieval at the End of the Early Years // IEEE TPAMI. 2000. Vol. 22, No. 12. P. 1349–1380.
10. Lowe D. G. Distinctive Image Features from Scale-Invariant Keypoints // International Journal of Computer Vision. 2004. Vol. 60, No. 2. P. 91–110.
11. Bay H., Tuytelaars T., Van Gool L. SURF: Speeded Up Robust Features // ECCV. 2006. P. 404–417.
12. Rublee E., Rabaud V., Konolige K., Bradski G. ORB: An Efficient Alternative to SIFT or SURF // ICCV. 2011. P. 2564–2571.
13. Bishop C. M. Pattern Recognition and Machine Learning. Springer, 2006. 738 p.
14. Han J., Kamber M., Pei J. Data Mining: Concepts and Techniques. 4th ed. Morgan Kaufmann, 2022. 704 p.
15. Ткаченко Р. О. Інтелектуальний аналіз даних. Львів : Видавництво Львівської політехніки, 2017. 312 с.

16. LeCun Y., Bottou L., Bengio Y., Haffner P. Gradient-Based Learning Applied to Document Recognition // Proceedings of the IEEE. 1998. Vol. 86, No. 11. P. 2278–2324.
17. Deng J. et al. ImageNet: A Large-Scale Hierarchical Image Database // CVPR. 2009. P. 248–255.
18. He K., Zhang X., Ren S., Sun J. Deep Residual Learning for Image Recognition // CVPR. 2016. P. 770–778.
19. Wang X. et al. Deep Metric Learning: A Survey // Pattern Recognition. 2022. Vol. 132.
20. Rui Y., Huang T. S., Chang S.-F. Image Retrieval: Current Techniques, Promising Directions and Open Issues // Journal of Visual Communication and Image Representation. 1999. Vol. 10. P. 39–62.
21. Schroff F., Kalenichenko D., Philbin J. FaceNet: A Unified Embedding for Face Recognition and Clustering // CVPR. 2015. P. 815–823.
22. Hadsell R., Chopra S., LeCun Y. Dimensionality Reduction by Learning an Invariant Mapping // CVPR. 2006. P. 1735–1742.
23. Aggarwal C. C. Similarity Search: The Metric Space Approach. Springer, 2021.
24. Бодянський Є. В., Руденко О. Г. Штучні нейронні мережі: архітектури, навчання, застосування. Харків : ТЕЛЕТЕХ, 2004. 369 с.
25. Субботін С. О. Нейронні мережі: теорія та практика. Запоріжжя : ЗНТУ, 2008. 364 с.
26. Russakovsky O. et al. ImageNet Large Scale Visual Recognition Challenge // IJCV. 2015. Vol. 115. P. 211–252.
27. Datta R., Joshi D., Li J., Wang J. Z. Image Retrieval: Ideas, Influences and Trends of the New Age // ACM Computing Surveys. 2008. Vol. 40, No. 2.
28. Fowler M. Patterns of Enterprise Application Architecture. Addison-Wesley, 2002. 533 p.
29. Evans E. Domain-Driven Design: Tackling Complexity in the Heart of Software. Addison-Wesley, 2003. 560 p.
30. Пасічник В. В., Шаховська Н. Б. Організація баз даних та знань. Львів : Магнолія 2006, 2012. 584 с.

31.Schildt H. Java: The Complete Reference. 13th ed. McGraw-Hill, 2024. 1248 p.
32.Walls C. Spring in Action. 6th ed. Manning, 2022. 520 p.
33.Momjian B. PostgreSQL: Introduction and Concepts. Addison-Wesley, 2001. 272 p.
34.Poulton N. Docker Deep Dive. Leanpub, 2023. 446 p.
35.Bauer C., King G. Java Persistence with Hibernate. 2nd ed. Manning, 2015. 608 p.

ДОДАТОК А

ПОВНІ РЕЗУЛЬТАТИ ЕКСПЕРИМЕНТІВ

Таблиця А.1 – Повні результати тестування системи

№	Тип тварини	Порода	Similarity Top-1	Similarity Top-5	Збігів породи в Top-5
1	Кіт	Abyssinian	0.95	0.88	5
2	Кіт	Bengal	0.93	0.85	5
3	Кіт	Birman	0.91	0.82	4
4	Кіт	Bombay	0.94	0.86	5
5	Кіт	Maine Coon	0.89	0.79	4
6	Кіт	Persian	0.92	0.83	5
7	Кіт	Ragdoll	0.88	0.77	4
8	Кіт	Siamese	0.95	0.87	5
9	Кіт	Sphynx	0.96	0.89	5
10	Кіт	Russian Blue	0.90	0.80	4
11	Собака	Beagle	0.94	0.86	5
12	Собака	Boxer	0.91	0.81	4
13	Собака	Chihuahua	0.95	0.88	5
14	Собака	English Cocker Spaniel	0.89	0.78	4
15	Собака	German Shepherd	0.93	0.84	5
16	Собака	Golden Retriever	0.92	0.83	5
17	Собака	Labrador Retriever	0.90	0.79	4
18	Собака	Pomeranian	0.96	0.90	5
19	Собака	Pug	0.94	0.87	5
20	Собака	Shiba Inu	0.88	0.76	4

ДОДАТОК Б

UML-ДІАГРАМИ ПРОГРАМНОЇ СИСТЕМИ

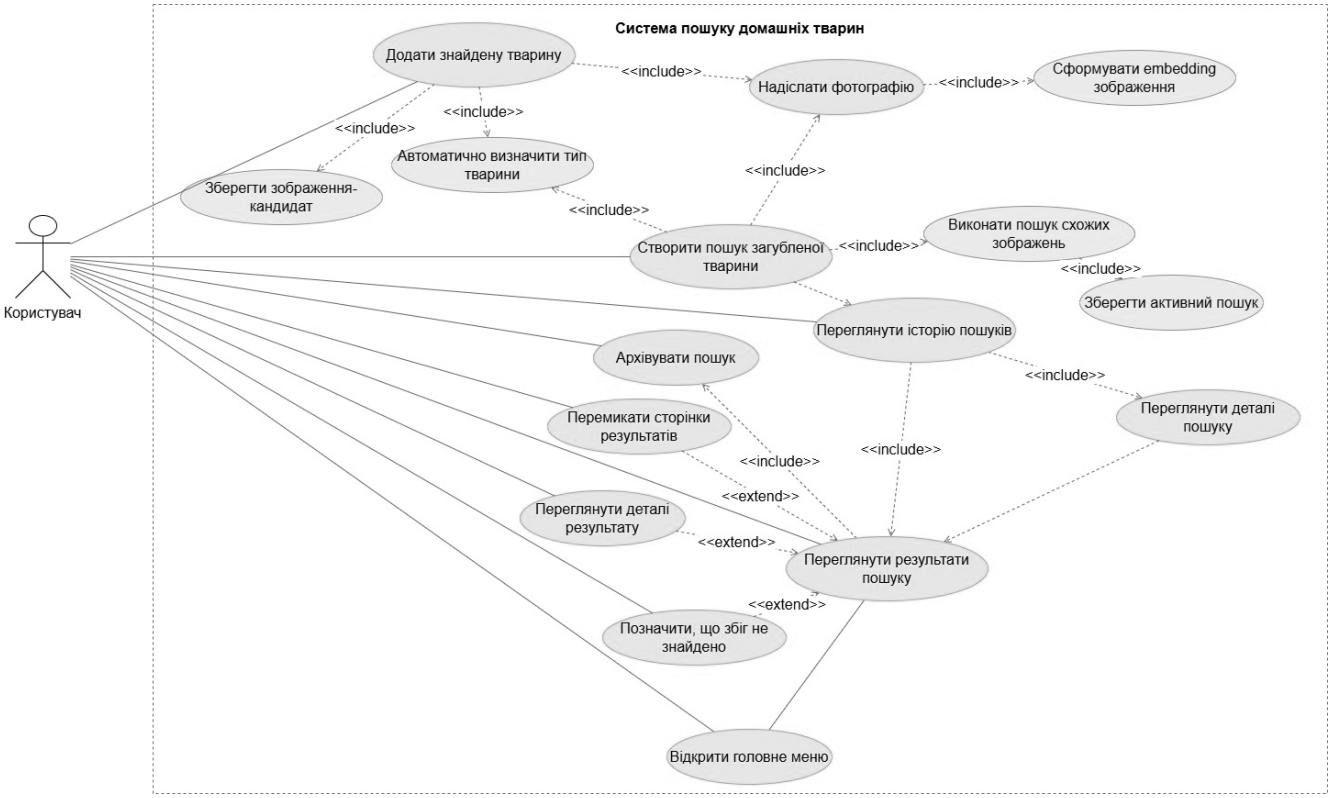

Рисунок Б.1 — Діаграма варіантів використання системи пошуку домашніх тварин.

Діаграма відображає основні сценарії взаємодії користувача із системою, зокрема створення пошуку, перегляд результатів, перегляд історії пошуків та видалення пошуку.

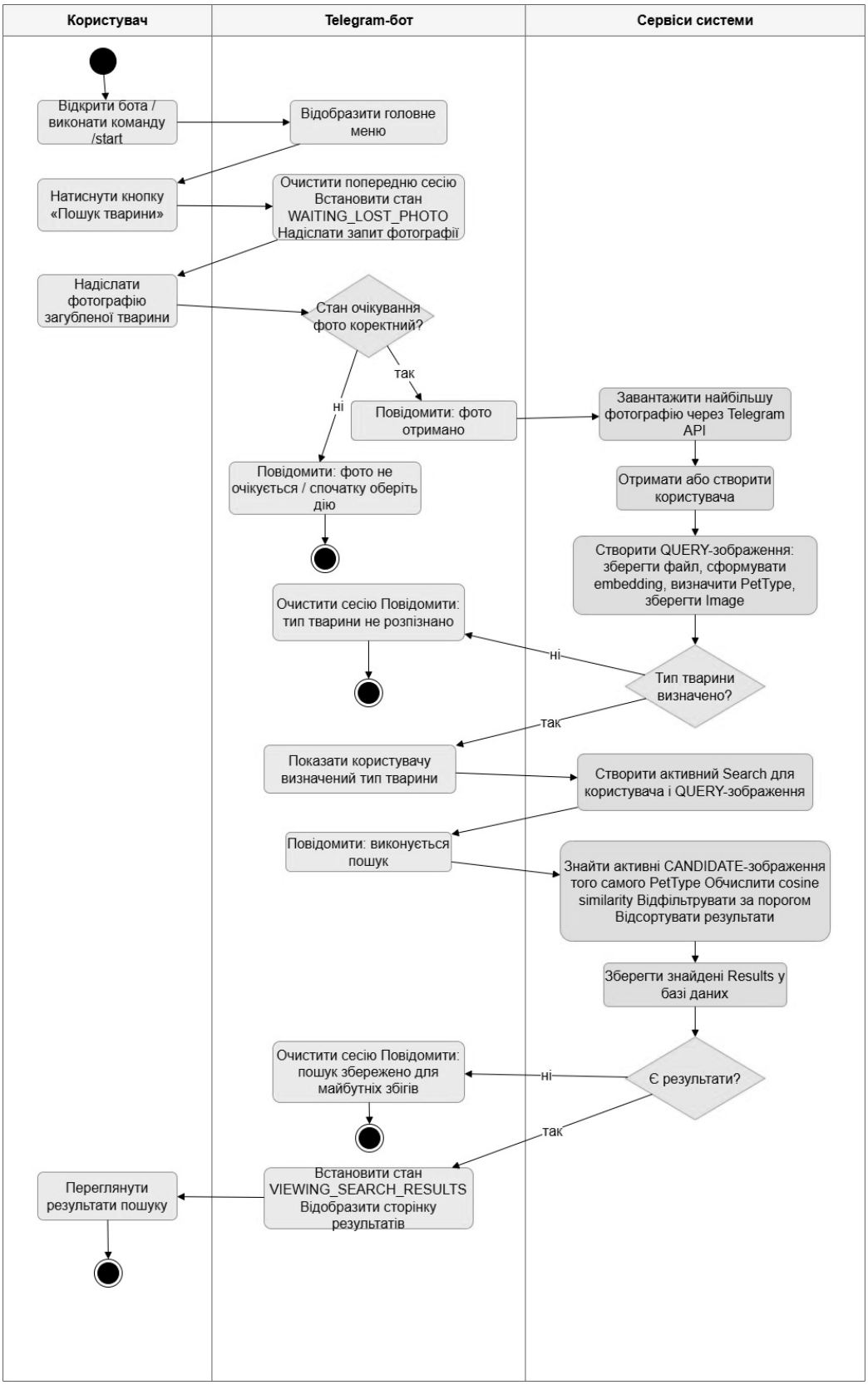

Рисунок Б.2 – Діаграма діяльності пошуку зниклої тварини

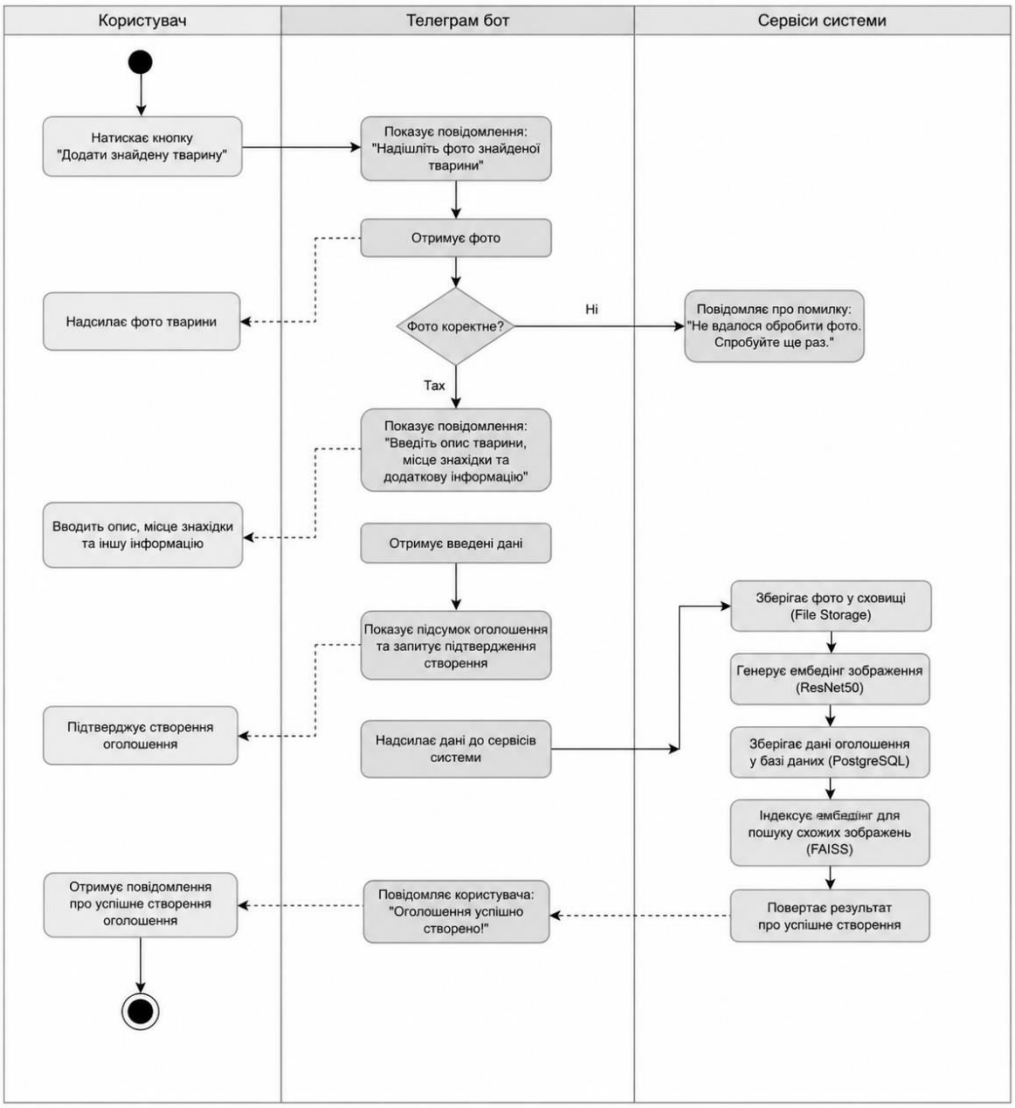

Рисунок Б.3 – Діаграма діяльності додавання фото знайденої тварини.

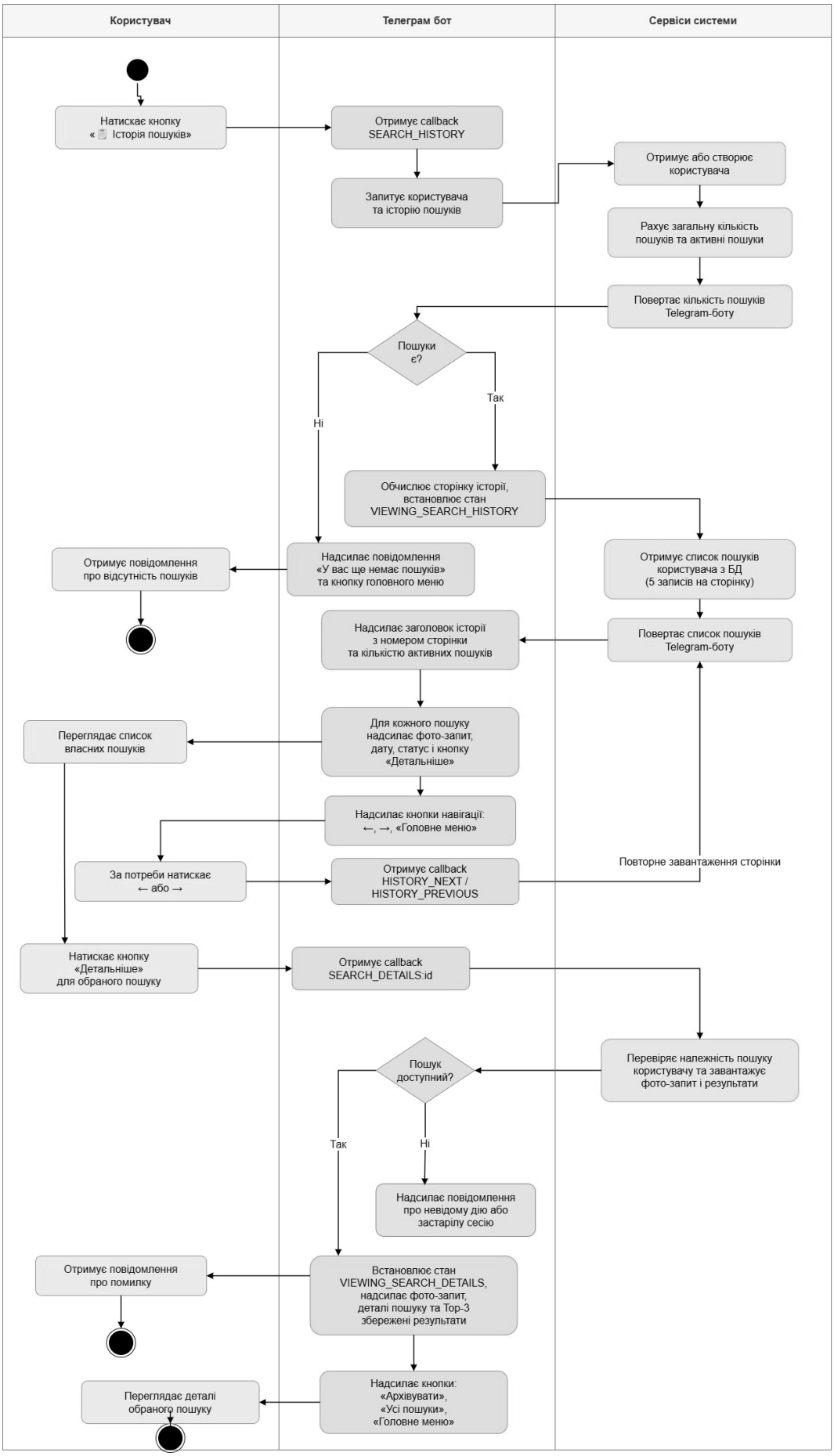
Користувач
Телеграм бот
Сервіси системи
Натискає кнопку « Історія пошуків»
Отримує callback SEARCH_HISTORY
Запитує користувача та історію пошуків
Отримує або створює користувача
Рахує загальну кількість пошуків та активні пошуки
Повертає кількість пошуків Telegram-боту
Пошуки є?
Так
Ні
Обчислює сторінку історії, встановлює стан VIEWING_SEARCH_HISTORY
Отримує повідомлення про відсутність пошуків
Надсилає повідомлення «У вас ще немає пошуків» та кнопку головного меню
Отримує список пошуків користувача з БД (5 записів на сторінку)
Надсилає заголовок історії з номером сторінки та кількістю активних пошуків
Повертає список пошуків Telegram-боту
Переглядає список власних пошуків
Для кожного пошуку надсилає фото-запит, дату, статус і кнопку «Детальніше»
Надсилає кнопки навігації: ←, →, «Головне меню»
За потреби натискає ← або →
Отримує callback HISTORY_NEXT / HISTORY_PREVIOUS
Повторне завантаження сторінки
Натискає кнопку «Детальніше» для обраного пошуку
Отримує callback SEARCH_DETAILS:id
Пошук доступний?
Перевіряє належність пошуку користувачу та завантажує фото-запит і результати
Так
Ні
Надсилає повідомлення про невідому дію або застарілу сесію
Отримує повідомлення про помилку
Встановлює стан VIEWING_SEARCH_DETAILS, надсилає фото-запит, деталі пошуку та Top-3 збережені результати
Переглядає деталі обраного пошуку
Надсилає кнопки: «Архівувати», «Усі пошуки», «Головне меню»

Рисунок Б.4 – Діаграма діяльності перегляду історії пошуків.

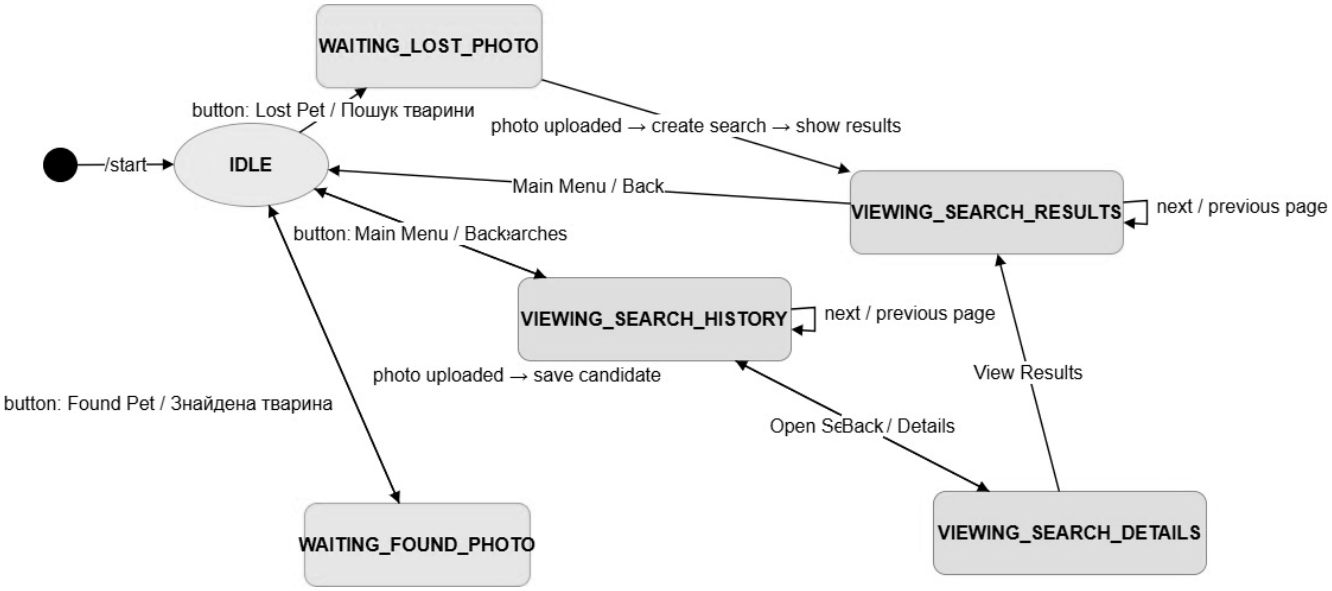

Рисунок Б.5 – Діаграма станів Telegram-бота.

ДОДАТОК В

ПРИКЛАДИ РОБОТИ TELEGRAM-БОТА

У даному додатку наведено приклади роботи розробленої програмної системи пошуку домашніх тварин у вигляді Telegram-бота. На рисунках продемонстровано основні сценарії взаємодії користувача із системою.

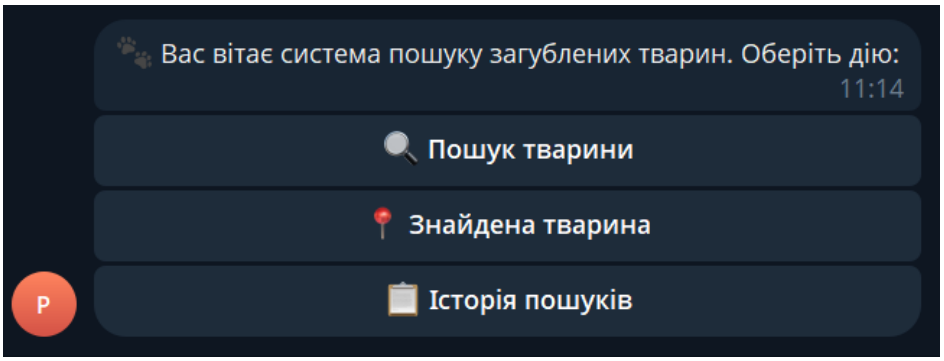

Рисунок В.1 – Головне меню Telegram-бота

На рисунку показано головне меню системи, з якого користувач може створити новий пошук загубленої тварини, додати фото знайденої тварини, або переглянути активні пошуки.

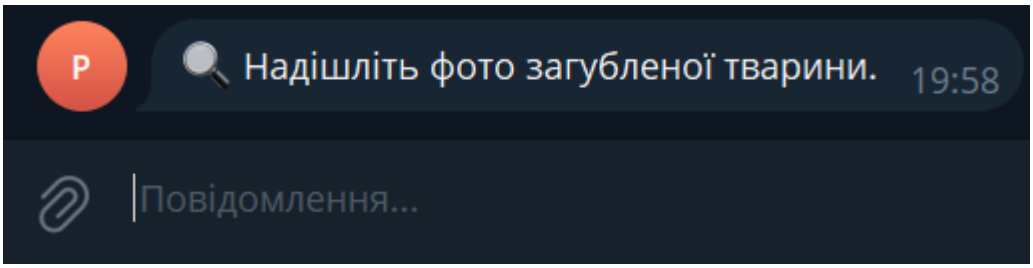

Рисунок В.2 – Процес створення нового пошуку

На рисунку показано процес створення нового пошуку. Після вибору відповідного пункту меню користувач отримує відповідне повідомлення,

завантажує фотографію тварини, яка використовується для формування векторного представлення та подальшого пошуку схожих зображень.

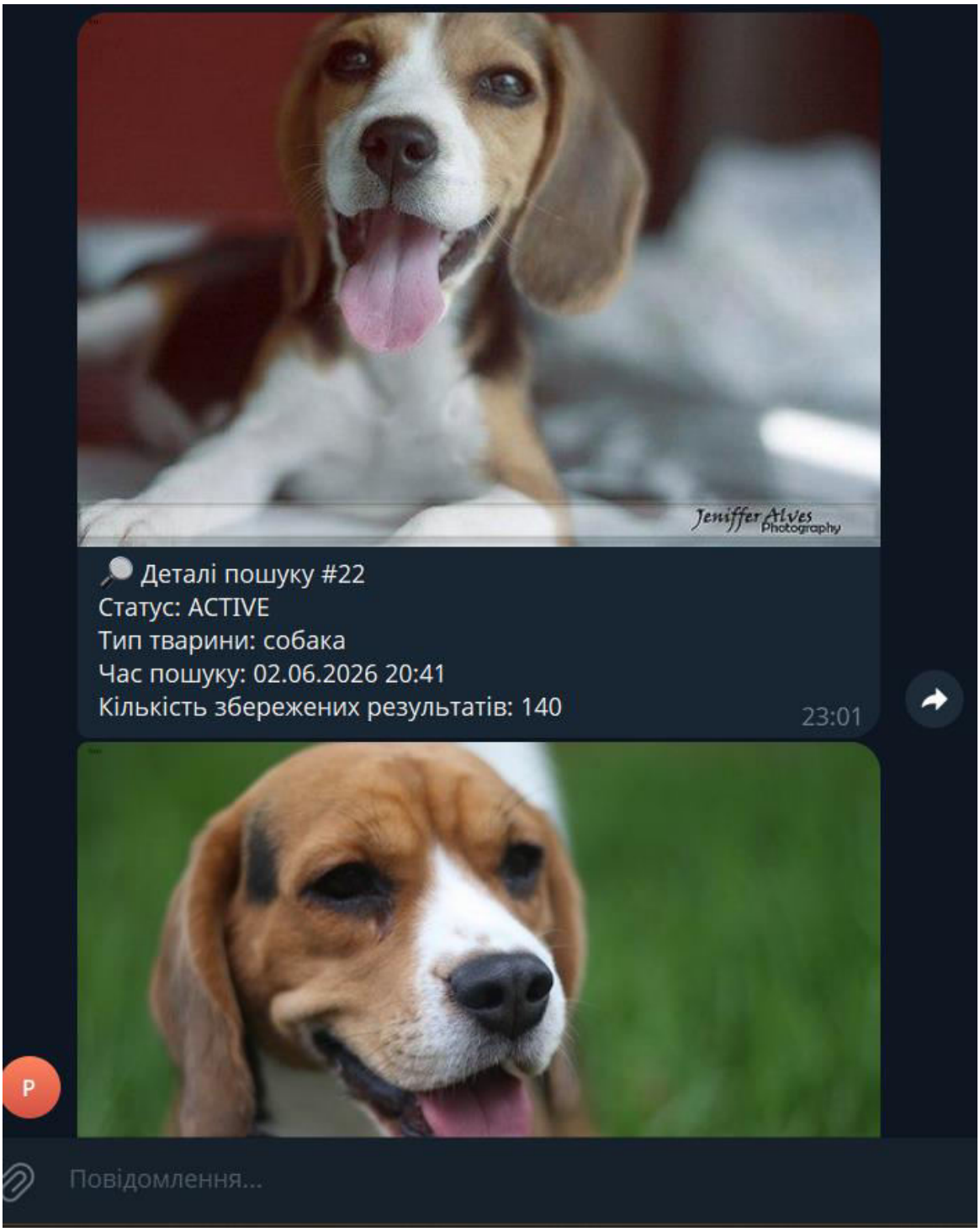

Рисунок В.3 – Відображення результатів пошуку

На рисунку наведено приклад результатів роботи алгоритму similarity search. Для кожного знайденого зображення відображається рівень схожості та додаткова інформація про відповідний запис.

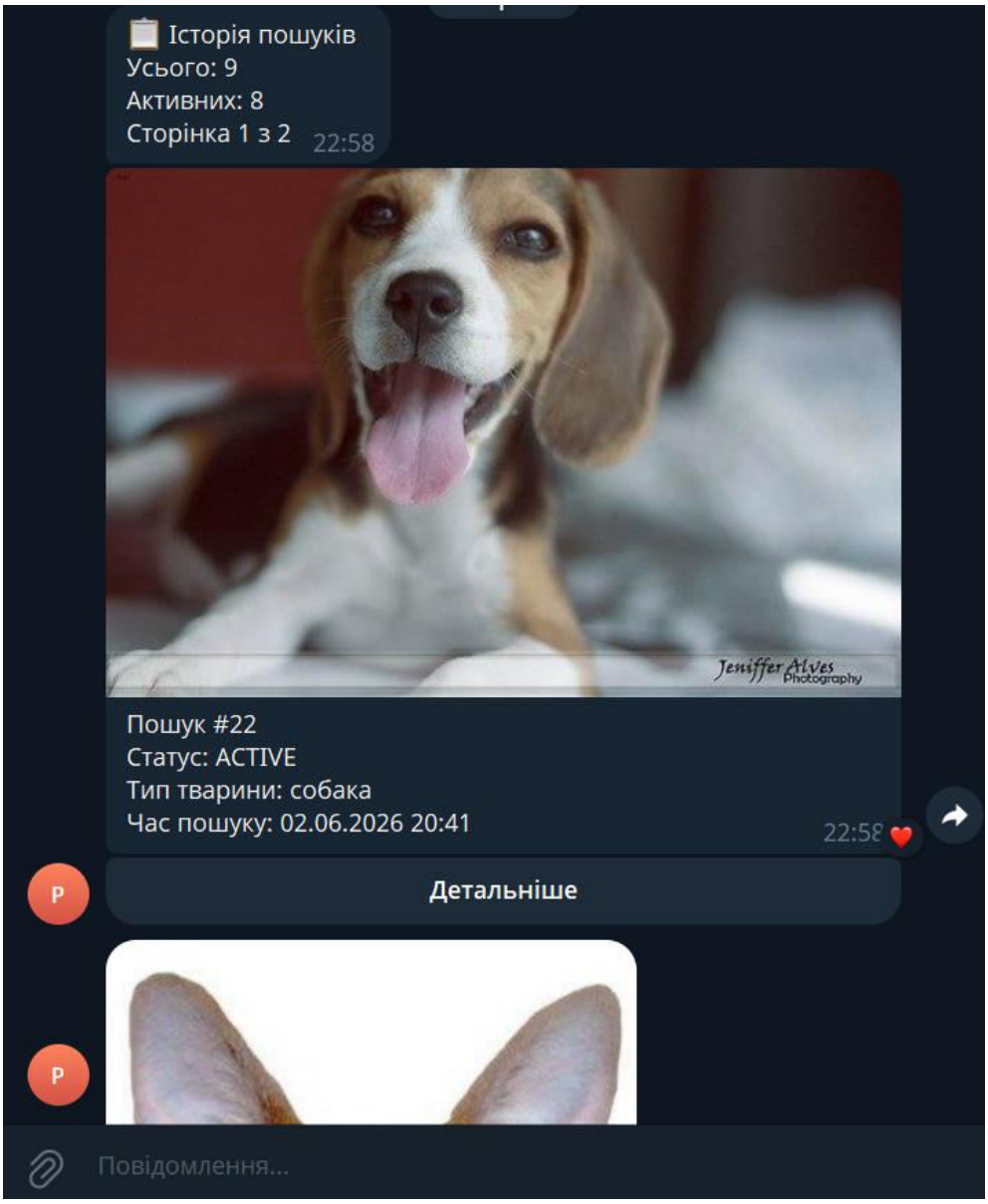

Рисунок В.4 – Перегляд історії пошуків користувача