ДОДАТОК А

Перелік джерел посилання за науковими напрямами керівника та науковців кафедри програмної інженерії

1. Використання машинного навчання для оптимізації доступу до даних в гібридному сховищі зображень / І. Кириченко, Г. Терещенко, Д. Панасенко// Біоніка інтелекту. – Харків: ХНУРЕ. – 2023. – № 1 (99). – С. 59-67.

3. Kyrychenko, I., Tereshchenko, G. Proniuk, G., Geseleva, N. “Predicate Clustering Method and its Application in the System of Artificial Intelligence”, 2023 7th International Conference on Computational Linguistics and Intelligent Systems (COLINS-2023), 2023. – CEUR-WS, 2023, ISSN 16130073. - Volume 3396, PP. 395 - 406.

4. Kyrychenko, I., Nazarov, O., Huliiev, N., Avdieiev, O. “Selection of Artificial Neural Networks for Disease Prediction”, 2023 7th International Conference on Computational Linguistics and Intelligent Systems (COLINS-2023), 2023. – CEUR-WS, 2023, ISSN 16130073. - Volume 3387, PP. 236-248.

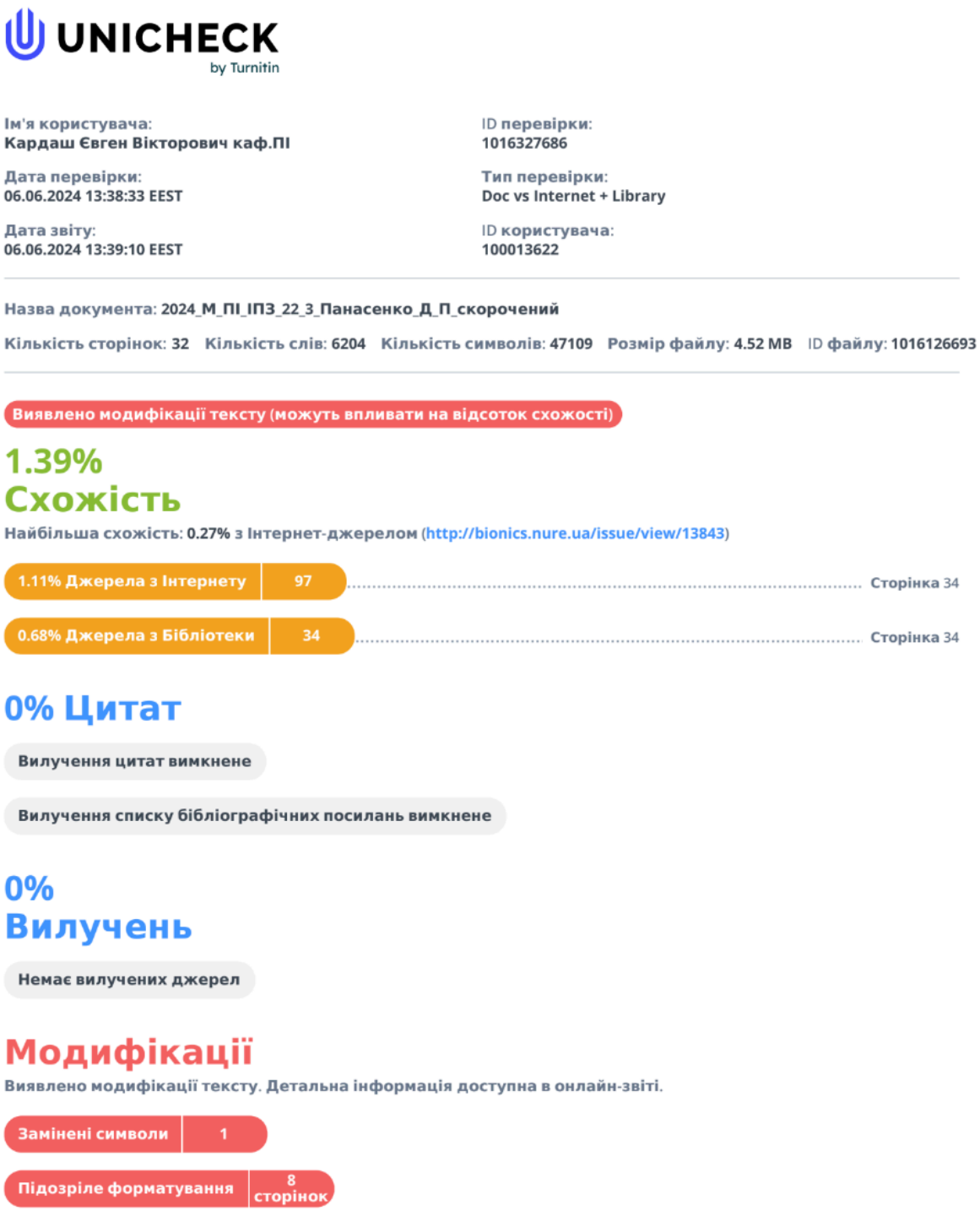
ДОДАТОК Б

Код Docker Compose для розгортання інфраструктури гібридного сховища зображень

```
version: '3'
services:
elasticsearch:
image: docker.elastic.co/elasticsearch/elasticsearch:7.17.4
container_name: elasticsearch
environment:
    - discovery.type=single-node
    -
ports:
    - 9200:9200
networks:
    - elastic

kibana:
  image: docker.elastic.co/kibana/kibana:7.17.4
  container_name: kibana
  ports:
  - 5601:5601
  environment:
  - ELASTICSEARCH_HOSTS: http://elasticsearch:9200
  networks:
  - elastic

  mongo:
  image: mongo:latest
  container_name: mongo
  ports:
  - 27017:27017
  volumes:
  - mongo-data:/data/db
  networks:
  - elastic

  networks:
  elastic:
  driver: bridge

  volumes:
  mongo-data:
```

ДОДАТОК В

Звіт результатів перевірки на унікальність тексту



UNICHECK
by Turnitin

Ім'я користувача:
Кардаш Євген Вікторович каф.ПІ

ID перевірки:
1016327686

Дата перевірки:
06.06.2024 13:38:33 EEST

Тип перевірки:
Doc vs Internet + Library

Дата звіту:
06.06.2024 13:39:10 EEST

ID користувача:
100013622

Назва документа: 2024_М_ПІ_ІПЗ_22_3_Панасенко_Д_П_скорочений

Кількість сторінок: 32 Кількість слів: 6204 Кількість символів: 47109 Розмір файлу: 4.52 MB ID файлу: 1016126693

Виявлено модифікації тексту (можуть впливати на відсоток схожості)

1.39%
Схожість

Найбільша схожість: 0.27% з Інтернет-джерелом (http://bionics.nure.ua/issue/view/13843)

1.11% Джерела з Інтернету 97 ..... Сторінка 34

0.68% Джерела з Бібліотеки 34 ..... Сторінка 34

0% Цитат

Вилучення цитат вимкнене

Вилучення списку бібліографічних посилань вимкнене

0%
Вилучень

Немає вилучених джерел

Модифікації

Виявлено модифікації тексту. Детальна інформація доступна в онлайн-звіті.

Замінені символи 1

Підозріле форматування 8 сторінок

ДОДАТОК Г

Апробація результатів роботи

УДК 004.4: 004.4

І.В. Кириченко[1], Г.Ю. Терещенко[2], Д.П. Панасенко[3],
[1]ХНУРЕ, м. Харків, Україна, iryna.kyrycheno@nure.ua, ORCID iD: 0000-0002-7686-6439
[2]ХНУРЕ, м. Харків, Україна, hlib.tereshchenko@nure.ua, ORCID iD: 0000-0001-8731-2135
[3]ХНУРЕ, м. Харків, Україна, daniil.panasenko@nure.ua

ВИКОРИСТАННЯ МАШИННОГО НАВЧАННЯ ДЛЯ ОПТИМІЗАЦІЇ ДОСТУПУ ДО ДАНИХ В ГІБРИДНОМУ СХОВИЩІ ЗОБРАЖЕНЬ

Робота присвячена дослідженню можливостей інтеграції машинного навчання для оптимізації доступу до даних в гібридному сховищі зображень. Основна задача полягає у пошуку схожих зображень серед великої кількості візуальних даних, що зберігаються у гібридному сховищі. Було розроблено систему, яка використовує методи глибинного навчання для вилучення ознак зображень, зокрема модель ResNet50, яка забезпечує високу точність вилучення ознак завдяки своїй глибинній архітектурі. Для ефективного пошуку схожих зображень застосовувалися MongoDB для зберігання зображень та метаданих, а також ElasticSearch для швидкої індексації та пошуку за векторами ознак. Проведено експериментальні дослідження з використанням датасету зображень різних тварин для оцінки продуктивності запропонованого підходу. Результати дослідження показали, що обраний підхід забезпечує високу швидкість та точність пошуку схожих зображень, підтверджуючи доцільність використання гібридних сховищ з використанням методів машинного навчання для ефективного управління великими обсягами візуальних даних. Використання попередньо навчених моделей значно знижує витрати на обчислювальні ресурси та час, необхідний для навчання, забезпечуючи при цьому високу точність і ефективність результатів.

ГІБРИДНЕ СХОВИЩЕ ЗОБРАЖЕНЬ, ДОСТУП ДО ДАНИХ, МАШИННЕ НАВЧАННЯ, НЕЙРОННА МЕРЕЖА, ELASTICSEARCH, MONGO DB, PYTHON

This work is dedicated to exploring the integration of machine learning to optimize data access in a hybrid image storage system. The primary task is to search for similar images among a large volume of visual data stored in the hybrid repository. A system has been developed that employs deep learning methods for feature extraction from images, specifically using the ResNet50 model, which provides high accuracy in feature extraction due to its deep architecture. For efficient image search, MongoDB was used for storing images and metadata, while ElasticSearch was utilized for fast indexing and vector feature search. Experimental studies were conducted using a dataset of various animal images to assess the performance of the proposed approach. The research results demonstrated that the chosen approach ensures high speed and accuracy in searching for similar images, confirming the feasibility of using hybrid storage with machine learning methods for effective management of large volumes of visual data. The use of pre-trained models significantly reduces computational resource costs and the time required for training, while still providing high accuracy and efficiency in the results.

HYBRID IMAGE STORAGE, DATA ACCESS, MACHINE LEARNING, NEURAL NETWORK, ELASTICSEARCH, MONGO DB, PYTHON

Вступ

В сучасному світі обсяг даних постійно зростає і значну частину цих даних становлять зображення. Важливість ефективного зберігання, управління та доступу до візуальних даних важко переоцінити, оскільки ці задачі виникають у різних галузях, включаючи медицину, безпеку, комерцію, наукові дослідження та багато інших.

Зберігання та управління зображеннями є важливим завданням, яке потребує ефективних рішень. Звичайні реляційні бази даних часто не забезпечують достатньої продуктивності для зберігання та обробки великих обсягів візуальних даних. Для таких задач підходять нові підходи, такі як NoSQL бази даних, об'єктні сховища та гібридні сховища.

Гібридні сховища поєднують різні типи сховищ, надаючи можливість зберігати дані в оптимальному форматі в залежності від їх характеристик та вимог до доступу. Наприклад, метадані зображень можуть зберігатися в реляційних або документних базах даних, тоді як самі зображення можуть зберігатися в об'єктних сховищах. Це забезпечує ефективне управління та доступ до даних.

Машинне навчання значно змінює підходи до обробки візуальних даних. ML моделі можуть використовуватися для автоматичного витягання ознак зображень, класифікації, розпізнавання об'єктів, покращення якості зображень та багато іншого. Використання попередньо навчених моделей дозволяє значно знизити час та ресурси, необхідні для навчання, забезпечуючи при цьому високу точність і ефективність.

Гібридні сховища даних у поєднанні з методами машинного навчання надають потужний інструмент для ефективного управління великими обсягами візуальних даних.

Таким чином, сучасні підходи до зберігання та обробки зображень базуються на використанні гібридних сховищ та методів машинного навчання, що забезпечує ефективне управління візуальними даними,

підвищує швидкість та точність доступу до них, а також дозволяє вирішувати складні завдання в різних галузях.

1. Гібридні сховища зображень

Гібридні сховища даних представляють собою комбінацію різних типів сховищ, які працюють разом для забезпечення оптимальної продуктивності, масштабованості та гнучкості [1]. Вони поєднують переваги реляційних баз даних (SQL) та нереляційних баз даних (NoSQL) для задоволення різних потреб у зберіганні та обробці даних.

Основна ідея гібридного сховища полягає в тому, щоб використовувати найкращі властивості кожного типу сховища для вирішення конкретних завдань. Наприклад, реляційні бази даних відмінно підходять для транзакційних систем, де важлива цілісність та консистентність даних, тоді як NoSQL бази даних, такі як MongoDB, забезпечують високу продуктивність та масштабованість для зберігання великих обсягів даних, особливо неструктурованих або напівструктурованих.

Основні особливості використання гібридних сховищ зображення.

Гнучкість у виборі технологій: Гібридні сховища дозволяють використовувати різні бази даних для різних типів даних. Це означає, що можна вибрати найкращий інструмент для кожної конкретної задачі.

Масштабованість: Використання NoSQL баз даних, таких як MongoDB, дозволяє легко масштабувати сховище для обробки великих обсягів даних. Це особливо важливо для додатків, які мають справу з великими обсягами неструктурованих даних.

Висока продуктивність: Гібридні сховища забезпечують високу продуктивність за рахунок використання NoSQL баз даних для швидкого доступу до великих обсягів даних та реляційних баз даних для забезпечення цілісності транзакцій.

Забезпечення цілісності даних: Реляційні бази даних, такі як PostgreSQL, або MySQL, забезпечують цілісність даних та підтримку складних транзакцій, що є важливим для критично важливих додатків.

Зниження витрат: Використання гібридних сховищ дозволяє знизити витрати за рахунок оптимізації використання ресурсів. Наприклад, часто використовувані дані можуть зберігатися у швидкодоступних NoSQL базах даних, тоді як менш часто використовувані дані можуть зберігатися у дешевших реляційних базах даних.

Проблеми, які вирішують гібридні сховища.

Управління різноманітними типами даних: Гібридні сховища дозволяють ефективно управляти як структурованими, так і неструктурованими даними. Це особливо корисно для додатків, які обробляють різноманітні типи даних, такі як текстові документи, зображення, відео та інші мультимедійні дані.

Швидкий доступ до даних: Використання NoSQL баз даних дозволяє забезпечити швидкий доступ до великих обсягів даних, що є критично важливим для додатків з високими вимогами до продуктивності.

Масштабованість: Гібридні сховища легко масштабуються, що дозволяє забезпечити безперебійний доступ до даних навіть при збільшенні обсягів даних.

Забезпечення безпеки даних: Реляційні бази даних забезпечують високий рівень безпеки даних, що є важливим для додатків, які обробляють конфіденційну інформацію.

Загалом, гібридні сховища забезпечують гнучкість, продуктивність та масштабованість, що робить їх ідеальним рішенням для багатьох сучасних завдань, пов'язаних з обробкою та зберіганням даних. Гібридні сховища дозволяють ефективно управляти різноманітними типами даних та забезпечувати швидкий доступ до них, що є критично важливим у сучасному світі великих даних.

2. Застосування машинного навчання

Машинне навчання стає дедалі важливішим інструментом у різних сферах, де зберігаються та обробляються великі обсяги зображень. Його застосування в поєднанні з гібридними сховищами зображень дозволяє вирішувати низку складних задач, що вимагають автоматизації та підвищеної точності.

Ось кілька прикладів задач, які можуть бути ефективно вирішені за допомогою машинного навчання в контексті гібридних сховищ зображень:

- автоматична класифікація зображень: машинне навчання може бути використане для автоматичної класифікації зображень у великій базі даних [2];
- аналіз і обробка зображень: алгоритми машинного навчання здатні автоматично аналізувати зображення для виявлення певних об'єктів або особливостей;
- сегментація зображень: у сфері обробки зображень важливо вміти сегментувати зображення, тобто розділяти його на значущі частини, машинне навчання дозволяє автоматично виділяти об'єкти на зображеннях;
- пошук схожих зображень: це одна з найпоширеніших задач, яка може бути вирішена за допомогою машинного навчання – система, що використовує алгоритми машинного навчання, здатна швидко знайти схожі зображення в базі даних на основі їхніх ознак;
- анотація та розпізнавання зображень: машинне навчання може автоматично додавати анотації до зображень, розпізнаючи об'єкти або сцени і це спрощує процес організації та пошуку зображень у великих базах даних.

Використання машинного навчання в контексті гібридних сховищ зображень дозволяє значно покращити ефективність та точність обробки візуальних даних.

3. Постановка задачі

Розібравшись в особливостях доступу до даних в гібридних сховищах зображень і проблемами які вони вирішують, для експериментальної інтеграції машинного навчання було вирішено взяти показову задачу, яка вирішується завдяки машинному навчанню. Однією з найактуальніших задач в цій області є пошук схожих зображень. Ця задача полягає в тому, щоб знайти велику кількість зображень, схожих на задане, серед інших зображень в гібридному сховищі зображень. Пошук схожих зображень є дуже практичною задачею, вирішення якої може принести значну користь у різних сферах.

Проблема, яка вирішується в даній роботі, полягає в оптимізації процесу пошуку схожих зображень у гібридному сховищі.

Основна мета цього дослідження – визначити, які методи машинного навчання можуть бути інтегровані в процес пошуку для підвищення його ефективності.

Важливо також оцінити, як саме використання гібридного сховища впливає на ефективність рішення даної задачі, а також які переваги застосування такого підходу.

4. Вибір методу машинного навчання

Вибір підходу до інтеграції машинного навчання є критично важливим етапом данного дослідження, оскільки саме методи машинного навчання визначають ефективність і точність пошуку схожих зображень у гібридному сховищі. Існує кілька популярних методів і моделей, які можуть вирішувати цю задачу [3].

Серед найбільш поширених підходів для пошуку схожих зображень можна виділити наступні методи машинного навчання.

Автокодувальники (Autoencoders) [4]: ці нейронні мережі можуть навчитися стискати дані до латентного простору меншої розмірності, зберігаючи важливу інформацію. Вони часто використовуються для вилучення ознак і можуть бути застосовані для порівняння схожості зображень. Автокодувальники можуть працювати з неструктурованими даними і вилучати значущі ознаки, але вони можуть бути менш точними у порівнянні з більш складними моделями, такими як CNN або ResNet.

Сверточні нейронні мережі (Convolutional Neural Networks, CNNs) [5]: CNN широко використовуються для аналізу зображень. Вони можуть вилучати складні ознаки з зображень, що робить їх ідеальними для задач класифікації та пошуку схожих зображень. CNN здатні вилучати складні та значущі ознаки зображень, що робить їх високоефективними для задач класифікації, але навчання CNN може бути ресурсомістким та вимагати великої кількості даних.

Мережі глибинного навчання для вилучення ознак [6] (Deep Feature Extraction): Цей підхід полягає у використанні попередньо навчених моделей, таких як VGG, Inception або ResNet, для вилучення ознак зображень, які потім використовуються для порівняння схожості [7]. Використання попередньо навчених моделей дозволяє знизити витрати на обчислювальні ресурси та час на навчання, а також забезпечує високу точність результатів, але можуть виникати проблеми з адаптацією до специфічних завдань або даних.

Зібрані дані про методи машинного навчання для пошуку схожих зображень, було зведено в порівняльну таблицю.

Таблиця 1

Порівняння методів машинного навчання

Метод	Переваги	Недоліки
Автокоду вальники	Працюють з неструктурованими даними, вилучають значущі ознаки	Менш точні
CNN	Вилучають складні та значущі ознаки, висока ефективність	Ресурсомісткі, потребують великої кількості даних
Глибинне вилучення ознак	Знижені витрати на обчислення та навчання, використання попередньо навчених моделей, висока точність результатів	Можливі проблеми з адаптацією до специфічних завдань чи даних

Виходячи з проведеного аналізу було вирішено для інтеграції машинного навчання в систему пошуку схожих зображень обрати метод глибинного вилучення ознак, так як він найбільш підходить для проведення даного дослідження, не вимагає витрат на навчання, через використання попередньо навчених моделей, а також надає високу точність результатів.

Серед цих методів ми обрали модель ResNet50 (Residual Networks) [8]. ResNet50 є однією з найпопулярніших моделей для вилучення ознак завдяки своїй високій точності. Модель ResNet50 вже попередньо навчена на великому датасеті зображень, що дозволяє використовувати її для вилучення ознак без додаткового навчання, знижуючи витрати на обчислювальні ресурси. Крім того, ResNet50 забезпечує високу точність результатів, що робить її ідеальним вибором для нашого дослідження.

ResNet50 – це глибока нейронна мережа, яка складається з 50 шарів і використовує концепцію "залишкових блоків" (residual blocks) для подолання проблеми затухання градієнтів у глибоких мережах. Основна ідея залишкових блоків полягає в тому, що

кожен блок не намагається вивчити точну карту відображення вхідних даних до вихідних, а замість цього вивчає різницю (залишок) між вхідними та вихідними даними.

Коли зображення подається на вхід ResNet50, воно проходить через кілька згорткових шарів, шарів підвибірки (pooling layers) і повнозв'язаних шарів. На кожному етапі мережа вилучає все більш абстрактні ознаки, що дозволяє моделі будувати багатошарове представлення вхідного зображення. На виході ResNet50 ми отримуємо вектор ознак (feature vector), який можна використовувати для порівняння схожості з іншими зображеннями.

Для порівняння схожості між векторами ознак ми використовуємо метрику косинусної подібності [9]. Косинусна подібність вимірює кут між двома векторами у векторному просторі і визначає, наскільки ці вектори схожі між собою. Воно обчислюється за формулою 1:

$$similarity = \cos(\theta) = \frac{A \cdot B}{\|A\|\|B\|} \quad (1)$$

де A і B – два вектори.

Косинусна подібність набуває значень від -1 до 1, де 1 означає, що вектори ідеально вирівняні (максимальна схожість), 0 означає, що вони ортогональні (немає схожості), а -1 означає, що вони ідеально протилежні. Для задачі пошуку схожих зображень ми зазвичай розглядаємо значення косинусної подібності від 0 до 1.

5. Вибір гібридного сховища зображень

Виходячи з проведеного аналізу, визначеної задачі і обраного методу машинного навчання, визначимо що функціональністю гібридного сховища зображень має бути: збереження зображень та пошук за вектором ознак. Вибір БД для виконання поставленої задачі проведемо за даною необхідною функціональністю.

Виділимо головні вимоги до збереження зображень:

- підтримка великих обсягів даних;
- гнучка схема зберігання для різних форматів даних (зображення, метадані);
- висока продуктивність при записі і читанні даних;
- масштабованість і надійність.

Виходячи з вимог до збереження зображень, БД, які відповідають цим вимогам, можуть бути такі БД, як: MongoDB, Cassandra та Couchbase [10]. В якості БД для збереження зображень для виконання дослідження було обрано MongoDB, так як вона являється найбільш популярною, з великою спільнотою користувачів.

Виділимо головні вимоги до пошуку за векторами ознак:

- швидка індексація та пошук за векторними ознаками;
- підтримка складних запитів і агрегацій;
- висока продуктивність при обробці великих обсягів даних;
- масштабованість і надійність.

Виходячи з вимог до пошуку за векторами ознак, БД, які відповідають цим вимогам можуть бути, такі БД, як: Elasticsearch та Faiss [11]. В якості БД для здійснення пошуку за векторами ознак для виконання дослідження було обрано Elasticsearch. Elasticsearch є оптимальним вибором для пошуку за векторами ознак завдяки своїй високій швидкості пошуку, швидкій індексації, підтримці складних запитів та агрегацій, а також розподіленій архітектурі. Крім того, Elasticsearch має велику та активну спільноту, що забезпечує підтримку та розвиток технології.

6. Опис програмного рішення

Програмне рішення базується на використанні двох головних компонентів. Перша компонента – це розгорнутий Docker Compose, який відповідає за контейнеризацію гібридного сховища зображень. Друга компонента це Python – додаток у JupyterLab, який використовується для виконання основного коду даного дослідження.

В якості гібридного сховища зображень було обрано використання комбінації двох баз даних – MongoDB та ElasticSearch.

MongoDB – це документо-орієнтована база даних, яка забезпечує ефективне зберігання та доступ до великих обсягів даних. Це найбільш ефективний спосіб зберігання зображень у базі даних. Кожне зображення зберігається у форматі BSON разом з інформацією про його метадані, такі як назва файлу, розмір та формат.

Так як задача даного дослідження вимагає пошук схожих зображень за допомогою векторних подібностей, було обрано ElasticSearch. ElasticSearch – це найкращий інструмент для швидкого пошуку векторів ознак зображень, так як в цій БД присутня індексація та Elasticsearch підтримує потужні функції, що робить його ідеальним вибором для виконання даного завдання [12].

Docker Compose використовується для управління контейнерами, що забезпечують роботу MongoDB та Elasticsearch. Це дозволяє легко розгортати та масштабувати систему, забезпечуючи ізоляцію середовища виконання для кожного компонента.

Python-додаток складається з кількох функціональних модулів, які забезпечують виконання різних функцій.

Модуль завантаження зображень: відповідає за завантаження зображень у гібридне сховище даних. Він обробляє завантажені зображення та зберігає їх у MongoDB і ElasticSearch.

MongoDB-клієнт: забезпечує взаємодію з базою даних MongoDB для зберігання та отримання зображень та їх метаданих.

ElasticSearch-клієнт: Відповідає за індексацію та пошук векторів ознак зображень у Elasticsearch.

ML–модель: Використовується для генерації векторів ознак зображень. Було обрано модель ResNet50,

яка завантажується з попередньо навченими вагами ImageNet [13]. За допомогою даної моделі відбувається порівняння схожості зображень.

Дана архітектура представлена у вигляді діаграми компонентів:

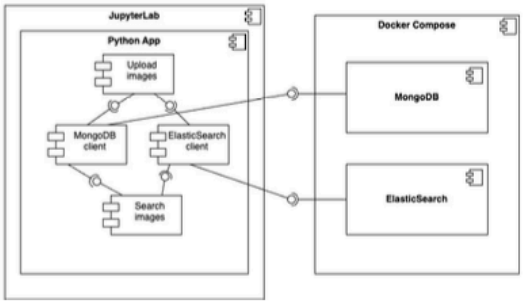


Рисунок 1 – Діаграма компонентів програмної реалізації дослідження

Ця архітектура забезпечить ефективну обробку, зберігання та пошук зображень, використовуючи переваги обраних нами сучасних технологій та інструментів.

Для успішної реалізації проекту було обрано широкий спектр технологій та інструментів, які забезпечують ефективну розробку, тестування та проведення якісного дослідження. У цьому розділі буде детально розглянуто кожну з обраних технологій та пояснені причини їх вибору.

Основною мовою програмування для проведення дослідження, було обрано мову програмування Python. Python відомий своєю простотою та читабельністю коду, що робить його дуже зручним інструментом. Крім того, Python має велику кількість бібліотек для роботи з даними та машинного навчання, що допоможе виконати поставлену задачу.

Для розробки були використані наступні бібліотеки.

Tensorflow Keras [14]. Для реалізації та тренування моделей машинного навчання використовується бібліотека Keras з бекендом TensorFlow. Це дозволило використати модель нейронної мережі, таку як ResNet50, яка використовується для визначення векторів ознак зображень. Використання попередньо натренованої моделі ResNet50 значно спростило процес машинного навчання для обробки зображень та підвищило точність результатів даного дослідження.

Pandas та NumPy [15]. Використання даних бібліотек допомагає дуже зручно обробляти і маніпулювати даними, багатовимірними масивами і числовими обчисленнями.

PyMongo та elasticsearch-py. Дані бібліотеки використовувались у якості клієнтів для забезпечення доступу до даних у сховища даних.

Matplotlib [15]. Для візуалізації результатів та аналізу даних обрана бібліотека Matplotlib. Вона дозволяє створювати графіки для візуального представлення результатів експериментів, а також виводити зображення у вікно виводу IDE.

Sklearn. З даної бібліотеки використовувалась функція cosine_similarity для визначення косинусної подібності векторів.

Розробка виконувалась в веб IDE – JupyterLab, що є інтерактивним середовищем для роботи з Python. JupyterLab дозволяє виконувати код, переглядати результати та візуалізувати дані у реальному часі, що значно спрощує процес дослідження та тестування моделей машинного навчання. Це середовище є особливо корисним для роботи з даними та побудови експериментів.

Для реалізації поставленої задачі також необхідно створити гібридне сховище зображень. В якості гібридного сховища зображень використовується комбінація MongoDB та ElasticSearch.

MongoDB – це документо-орієнтована база даних, яка забезпечує високу гнучкість та масштабованість. MongoDB дозволяє зберігати великі обсяги даних у форматі JSON та легко їх обробляти. MongoDB в даному дослідженні використовується для зберігання зображень та пов'язаних з ними метаданих.

ElasticSearch – це пошуковий рушій з відкритим вихідним кодом, який спеціалізується на повнотекстовому пошуку та аналізі великих обсягів даних у реальному часі. ElasticSearch дозволяє швидко проводити пошук схожих зображень за допомогою методів машинного навчання та підтримує індексацію та пошук векторних даних.

Kibana використовувалась в якості веб GUI для роботи з ElasticSearch.

MongoDB Compass використовувався в якості десктоп GUI для роботи з MongoDB.

Для забезпечення зручного розгортання та управління інфраструктурою проекту був обран Docker. Docker дозволяє створювати контейнери, які ізолюють середовище виконання додатків, забезпечуючи їхню портативність та легкість розгортання. Docker Compose був використаний для спрощення управління багатосервісною архітектурою, що включає MongoDB, Elasticsearch та Kibana.

7. Проведення експериментів

Результатами проведених експериментів мають бути дані для аналізу і формування висновків щодо доцільності використання обраного методу машинного навчання ResNet50 з гібридним підходом до зберігання зображень у вирішенні проблеми – швидкого пошуку схожих зображень в сховищі даних.

Для цього визначено провести порівняння трьох різних підходів для вирішення цієї проблеми.

Перший підхід. Збереження зображення в єдиному сховищі MongoDB. При пошуку схожих зображень

перебрати усі зображення зі сховища, розрахувати вектор ознак і вирахувати коефіцієнт їхньої схожості.

Другий підхід. Збереження зображення в єдиному сховищі MongoDB з попереднім прорахунком вектору ознак зображення і збережені їх в цьому ж сховищі. При пошуку схожих зображень перебрати усі зображення зі сховища, взяти вектор ознак зі сховища і вирахувати коефіцієнт їхньої схожості.

Третій підхід. Збереження зображення в сховищі MongoDB. Попередній прорахунок вектора ознак зберігається в ElasticSearch. Для пошуку схожих зображень робимо запит в ElasticSearch, пошук відбувається на стороні сховища.

Проведення експерименту має відбуватись з використанням цих трьох підходів для визначення їх ефективності.

Для якісного проведення експерименту необхідно виконати ці три тести на різних обсягах даних.

Необхідно зробити заміри часу виконання кожного експерименту. Це і буде головним критерієм для оцінки

Експерименти будуть проводитись на локальному комп'ютері з 8 ядрами процесора M1 Apple Silicon, 8 ГБ оперативної пам'яті, використовуючи Python 3.8, MongoDB версії 4.4, Elasticsearch версії 7.10 та TensorFlow версії 2.4.

В якості тестових зображень, необхідно було обрати великий дата-сет зображень, гарної якості, одного формату, для якісного і зручного проведення експериментів. Зображення мають бути, як різноманітні, так і мати певну схожість (деякі з них), для наочного бачення коректності роботи програми.

Було обрано дата-сет з зображеннями різних тварин з Kaggle [16]. Даний дата-сет налічує 5400 зображень 90 видів тварин розподілених за окремими директоріями. Зображення якісні в форматі .jpg.

Під час виконання експериментів даний дата-сет буде завантажено в сховище і на основі нього буде відбуватися пошук схожих зображень.

Правильним результатом виконання тестових програм буде, те що результуючі схожі зображення відносяться до одного типу тварин.

Експеримент буде проведено на різних обсягах даних. Визначимо наступні обсяги даних – 10, 20, 50, 100, 500, 1000, 5400. Для цього буде взято необхідну кількість випадкових зображень з цього дата-сету.

Проведено визначені експерименти.

Рисунок 2 – Результати дослідження

Порівнявши час виконання кожного експерименту, варіант з використанням гібридного сховища зображень виявився найбільш ефективним. Усі зафіксовані результати трьох експериментів представлено в таблиці. Результати представлені у секундах.

Таблиця 2

Результати виконання експериментів

Експеримент	10	20	50	100	500	1000	5400
MongoDB	0.881	1.591	4.008	7.54	38.126	81.656	541.729
MongoDB з попереднім прорахунком	0.08	0.095	0.113	0.143	0.442	0.687	3.254
MongoDB з ElasticSearch	0.156	0.120	0.173	0.138	0.391	0.289	0.208

Результати проведених експериментів показують значні відмінності у швидкості пошуку схожих зображень між трьома підходами: використання тільки MongoDB, використання MongoDB з попереднім прорахунком векторів ознак, та використання гібридного сховища у якості комбінації MongoDB з Elasticsearch.

Для розуміння залежності часу виконання пошуку від обсягів даних побудуємо графік (див. рис. 4).

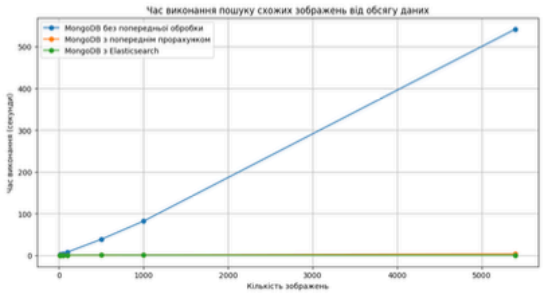


Рисунок 3 – Графік залежності часу виконання від обсягів даних трьох підходів

На графіку наглядно можна побачити лінійну залежність часу виконання від кількості зображень в методі використання тільки MongoDB без попереднього прорахунку. Причому даний підхід досить затратний. Це пояснюється тим, що кожного разу під час пошуку схожих зображень система повинна перебирати всі зображення та обчислювати вектори ознак «на льоту». Це призводить до суттєвих витрат часу при великих обсягах даних. Оцінка складності алгоритму – O(n).

Для більш наглядної оцінки наступних двох підходів, було прийнято рішення виключити цей підхід з графіку (див. рис. 5).

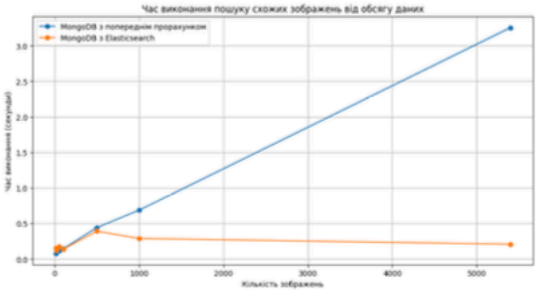


Рисунок 4 – Графік залежності часу виконання від обсягів даних двох підходів

Використання MongoDB з попереднім прорахунком векторів ознак значно покращує час виконання пошуку. Це пояснюється тим, що вектори ознак вже збережені у базі даних і система не витрачає час на їх обчислення під час пошуку. Однак все одно ми бачимо лінійне збільшення часу виконання від обсягів даних. Це пояснюється тим що виконується повний перебір в сховищі для пошуку найбільш схожих зображень. Оцінка складності алгоритму – O(n).

Зовсім інша залежність з використанням гібридного сховища зображень у якості MongoDB з ElasticSearch. На результатах не видно залежності часу від обсягів даних. Це обумовлюється тим що, використання ElasticSearch для пошуку максимально ефективне і за рахунок індексації має логарифмічну складність –O(log(n)), для якої обрані дані об'єми даних значно малі.

Використання гібридного сховища у якості комбінації MongoDB та Elasticsearch дозволяє максимально використати сильні сторони кожної технології. MongoDB забезпечує гнучке та надійне зберігання даних, тоді як Elasticsearch надає потужні можливості ефективного пошуку.

Висновки

В ході проведення дослідження було виконано аналіз предметної області та розглянуто можливості інтеграції машинного навчання для доступу до даних в гібридних сховищах зображення. Описано їх принципи роботи, виділено особливості, основні відмінності, переваги та недоліки.

У процесі роботи було поставлено задачу реалізувати алгоритм пошуку схожих зображень в сховищі, було розроблено та протестовано програмну частину, яка реалізує кілька підходів до зберігання та пошуку зображень, що дозволило зробити висновки щодо їх ефективності та доцільності використання в реальних умовах.

Перш за все, ми визначили, що традиційний підхід, який полягає у зберіганні зображень у єдиному сховищі MongoDB без попереднього розрахунку векторів ознак, є найменш ефективним з точки зору швидкості пошуку. Результати експериментів показали, що цей метод має значні затримки при обробці великих обсягів даних, що робить його непридатним для задач, де потрібна висока швидкість доступу до зображень. А при збільшені обсягів даних час виконання пошуку збільшується, що унеможливлює пошук.

Водночас, використання підходу з попереднім розрахунком векторів ознак та їх зберіганням у MongoDB дозволяє значно скоротити час пошуку схожих зображень. Такий підхід забезпечує швидший доступ до даних за рахунок попередньої обробки зображень і збереження результатів обчислень, що робить його більш ефективним рішенням на невеликих обсягах даних. Однак даний підхід все одно має лінійну залежність часу виконання від обсягів даних, тому не може використовуватись на великих обсягах даних.

Найбільш ефективним виявився гібридний підхід, який поєднує зберігання зображень у MongoDB та векторів ознак у Elasticsearch. Завдяки індексації і потужним можливостям пошуку ми досягли логарифмічної залежності часу виконання від обсягів даних, що робить його Цей метод дозволяє не тільки швидко знайти схожі зображення, але й забезпечує гнучкість у пошукових запитах та високу точність результатів завдяки можливостям Elasticsearch. Експерименти підтвердили, що цей підхід має найнижчий час виконання запитів, особливо при обробці великих обсягів даних.

Таким чином, результати дослідження підтверджують доцільність використання гібридних сховищ зображень з інтеграцією методів машинного навчання для оптимізації процесу пошуку. Використання MongoDB та Elasticsearch у поєднанні з попереднім розрахунком векторів ознак дозволяє досягти високої ефективності та точності при роботі з великими обсягами візуальних даних.

Важливим висновком є також те, що правильний вибір методів машинного навчання та технологій зберігання даних значно впливає на результати роботи системи. Використання моделі ResNet50 для екстракції ознак зображень та алгоритму косинусної подібності для порівняння векторів ознак виявилося ефективним рішенням для задачі пошуку схожих зображень.

Загалом, проведене дослідження показало, що інтеграція машинного навчання в процеси управління та обробки даних у гібридних сховищах зображень відкриває нові можливості для покращення ефективності та точності роботи з великими обсягами візуальної інформації. Це дозволяє забезпечити швидкий та надійний доступ до необхідних даних у різних сферах застосування, включаючи медицину, безпеку та електронну комерцію.

Мета завдання досягнута за рахунок визначення у роботі ефективності використання методів машинного навчання для доступу до даних в гібридних сховищах зображень. У результаті роботи було підібрано найефективнішу модель машинного навчання і гібридне сховище зображень для вирішення поставленої задачі.

Результати цього дослідження можуть бути використані для прийняття рішення про те, який підхід використовувати для збереження і організації зображень у сховищі і який алгоритм та модель машинного навчання використовувати для пошуку схожих зображень

Список літератури:

[1] James, Blessing & Asagba, P. (2017). Hybrid Database System for Big Data Storage and Management. International Journal of Computer Science, Engineering and Applications. 7. 15-27. 10.5121/ijcsea.2017.7402.

[2] Kyrychenko, I., Tereshchenko, G. Proniuk, G., Geseleva, N. “Predicate Clustering Method and its Application in the System of Artificial Intelligence”, 2023 7th International Conference on Computational Linguistics and Intelligent Systems (COLINS-2023), 2023. – CEUR-WS, 2023, ISSN 16130073. - Volume 3396, PP. 395 - 406.

[3] Kyrychenko, I., Nazarov, O., Huliiev, N., Avdieiev, O. “Selection of Artificial Neural Networks for Disease Prediction”, 2023 7th International Conference on Computational Linguistics and Intelligent Systems (COLINS-2023), 2023. – CEUR-WS, 2023, ISSN 16130073. - Volume 3387, PP. 236-248.

[4] Berahmand, Kamal & Daneshfar, Fatemeh & Salehi, Elaheh & Li, Yuefeng & Xu, Yue. (2024). Autoencoders and their applications in machine learning: a survey. Artificial Intelligence Review. 57. 10.1007/s10462-023-10662-6.

[5] Tabian, I., Fu, H., & Khodaei, Z.S. (2019). A Convolutional Neural Network for Impact Detection and Characterization of Complex Composite Structures. Sensors, 19(22), 4933. DOI: 10.3390/s19224933.

[6] Asokan, Anju & Jude, Anitha & Patrut, Bogdan & Danciulescu, Dana & D, Jude. (2020). Deep Feature Extraction and Feature Fusion for Bi-temporal Satellite Image Classification. Computers, Materials & Continua. 66. 373-388. 10.32604/cmc.2020.012364.

[7] He, K., Zhang, X., Ren, S., & Sun, J. (2016). Deep residual learning for image recognition. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2016, pp. 770-778. doi:10.1109/CVPR.2016.90.

[8] 8. Meghana, Avuthu Sai. "Age and Gender prediction using Convolution, ResNet50 and Inception ResNetV2." International Journal of Advanced Trends in Computer Science and Engineering 9, no. 2 (April 25, 2020): 1328–34. http://dx.doi.org/10.30534/ijatcse/2020/65922020.

[9] 9. Understanding Cosine Similarity and Cosine Distance in Depth. URL: https://pub.aimind.so/understanding-cosine-similarity-and-cosine-distance-in-depth-cc91eac3ef2 (дата звернення: 29.04.2024).

[10] 10. Matallah, Houcine & Belalem, Ghalem & Bouamrane, K.. (2020). Evaluation of NoSQL Databases: MongoDB, Cassandra, HBase, Redis, Couchbase, OrientDB. International Journal of Software Science and Computational Intelligence. 12. 71-91. 10.4018/IJSSCI.2020100105.

[11] 11. Znakhur, Serhii & Znakhur, Liudmyla. (2022). Similar goods search based on FAISS. Bulletin of Kharkov National Automobile and Highway University. 40. 10.30977/BUL.2219-5548.2022.96.0.40.

[12] 12. Wong, Wai-Tak. (2019). Advanced Elasticsearch 7.0: A practical guide to designing, indexing, and querying advanced distributed search engines.

[13] 13. Russakovsky, O., Deng, J., Su, H., Krause, J., Satheesh, S., Ma, S., Huang, Z., Karpathy, A., Khosla, A., Bernstein, M., Berg, A. C., & Fei-Fei, L. (2015). ImageNet Large Scale Visual Recognition Challenge. International Journal of Computer Vision, 115(3), 211-252. doi:10.1007/s11263-015-0816-y.

[14] 14. Keras Applications: ResNet. URL: https://keras.io/api/applications/resnet/ (дата звернення: 20.04.2024).

[15] 15. Nelli, Fabio. (2018). The NumPy Library: With Pandas, NumPy, and Matplotlib. 10.1007/978-1-4842-3913-1_3.

[16] 16. Animal Image Dataset (90 Different Animals) on Kaggle. URL: https://www.kaggle.com/datasets/iamsouravbanerjee/animal-image-dataset-90-different-animals (дата звернення: 04.05.2024).

ДОДАТОК Д

Слайди презентації

ДОСЛІДЖЕННЯ МОЖЛИВОСТЕЙ ІНТЕГРАЦІЇ МАШИННОГО НАВЧАННЯ ДЛЯ ОПТИМІЗАЦІЇ ДОСТУПУ ДО ДАНИХ В ГІБРИДНОМУ СХОВИЩІ ЗОБРАЖЕНЬ

Виконав:
ст. гр. ІПЗм-22-3 Панасенко Д.П.

Керівник:
к.т.н., доц. каф. ПІ Кириченко І.В.

1

Актуальність

Метою роботи є проведення практичного дослідження можливостей інтеграції машинного навчання для оптимізації доступу до даних в гібридному сховищі зображень.

Об’єктом дослідження є доступ до даних в гібридних сховищах зображень.

Предметом дослідження є гібридні сховища та методи машинного навчання

Методами дослідження є аналіз проблемної області, вибір гібридного сховища зображень для проведення, та використання бібліотек Python для реалізації моделей машинного навчання.

2

Аналіз предметної галузі

Гібридні сховища даних представляють собою комбінацію різних типів сховищ, які працюють разом для забезпечення оптимальної продуктивності, масштабованості та гнучкості. Вони поєднують переваги реляційних баз даних (SQL) та нереляційних баз даних (NoSQL) для задоволення різних потреб у зберіганні та обробці даних.

Приклади задач які вирішуються завдяки методам машинного навчання:

- Автоматична класифікація зображень
- Аналіз і обробка зображень
- Сегментація зображень
- Пошук схожих зображень
- Анотація та розпізнавання зображень

3

Постановка задачі

- Розглянути існуючі методи та технології управління зображеннями
- Оцінити потреби та вимоги до системи зберігання та пошуку зображень
- Розглянути та оцінити різні методи машинного навчання для пошуку схожих зображень
- Розробити архітектуру системи для зберігання та пошуку схожих зображень
- Розробити програму для пошуку схожих зображень в гібридному сховищі зображень використовуючи методи машинного навчання
- Провести експериментальні дослідження для оцінки продуктивності системи
- Проаналізувати результати експериментів та надати рекомендації

4

Вибір методу машинного навчання

Метод	Переваги	Недоліки
Автокодувальники	Працюють з неструктурованими даними, вилучають значущі ознаки	Менш точні
CNN	Вилучають складні та значущі ознаки, висока ефективність	Ресурсомісткі, потребують великої кількості даних
Глибинне вилучення ознак	Знижені витрати на обчислення та навчання, використання попередньо навчених моделей, висока точність результатів	Можливі проблеми з адаптацією до специфічних завдань чи даних

Вибір: модель глибинного вилучення ознак ResNet50

5

Вибір гібридного сховища зображень

Збереження зображення:

- Підтримка великих обсягів даних
- Гнучка схема зберігання для різних форматів даних
- Висока продуктивність при записі і читанні даних
- Масштабованість і надійність

mongoDB®

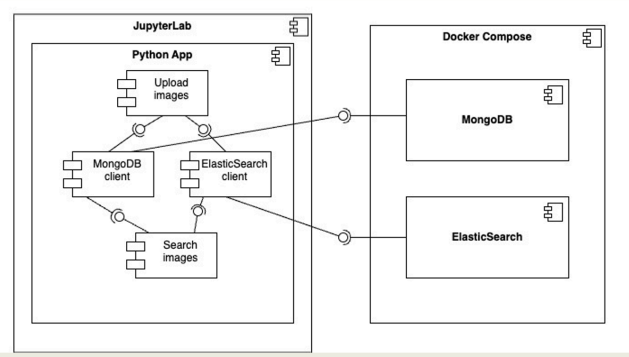
Пошук за векторами ознак:

- Швидка індексація та пошук за векторними ознаками
- Підтримка складних запитів і агрегацій
- Висока продуктивність при обробці великих обсягів даних
- Масштабованість і надійність

elasticsearch

6

Проектування програми



Технології:

- Python
- JupyterLab
- Docker, Docker Compose
- MongoDB, ElasticSearch
- MongoDB Compass, Kibana
- PyMongo, Elasticsearch-Py
- TensorFlow Keras
- Pandas, NumPy, Matplotlib
- Sklearn

Реалізація. Інтеграція з MongoDB

```
mongo_client = pymongo.MongoClient("mongodb://localhost:27017/")
mongo_db = mongo_client["image_database"]
mongo_collection = mongo_db["images"]
```

Підключення

```
def upload_image_mongo(image_path, features):
    with open(image_path, "rb") as image_file:
        image_data = image_file.read()
        image = Image.open(io.BytesIO(image_data))
        metadata = {
            "filename": os.path.basename(image_path),
            "size": image.size,
            "format": image.format
        }
        result = mongo_collection.insert_one({
            "image_data": image_data,
            "metadata": metadata,
            "features": features.tolist()
        })
        print(f"Uploaded {image_path} with ID {result.inserted_id}")
        return result.inserted_id
```

Завантаження зображень

```
def get_image_by_id_mongo(image_id):
    document = mongo_collection.find_one({"_id": ObjectId(image_id)})
    if document:
        image_data = document["image_data"]
        image = Image.open(io.BytesIO(image_data))
        return image
    else:
        print(f"No image found with ID {image_id}")
        return None
```

```
def get_document_by_id_mongo(image_id):
    document = mongo_collection.find_one({"_id": ObjectId(image_id)})
    if document:
        return document
    else:
        print(f"No image found with ID {image_id}")
        return None
```

Отримання зображень

Реалізація. Інтеграція з ElasticSearch

```
curl -X PUT "localhost:9200/image_features" -H 'Content-Type: application/json' -d'
{
  "mappings": {
    "properties": {
      "features": {
        "type": "dense_vector",
        "dims": 2048
      }
    }
  }
}'
```

Створення індексу

```
def index_image_features(image_id, features):
    body = {"features": features.tolist()}
    es.index(index="image_features", id=image_id, body=body)
    print(f"Uploaded image {image_id} with features {features}")
```

Завантаження зображень

```
es = Elasticsearch([{'host': 'localhost', 'port': 9200, 'scheme': 'http'}],
    connections_per_node=25,
    request_timeout=60,
    retry_on_timeout=True
)
```

Підключення

9

Реалізація. Модель машинного навчання

```
model = ResNet50(weights='imagenet', include_top=False, pooling='avg')
```

Модель ResNet50

```
def extract_features(image):
    img = image.resize((224, 224))
    img_data = np.array(img)

    img_data = np.expand_dims(img_data, axis=0)
    img_data = preprocess_input(img_data)

    features = model.predict(img_data, verbose=0)
    return features.flatten()
```

Визначення вектору ознак

```
def compare_images(features1, features2):
    similarity = cosine_similarity([features1], [features2])[0][0]
    return similarity
```

Визначення косинусної подібності

10

Реалізація. Пошук схожих зображень

```python
def search_similar_images_elastic(query_features, num_results):
    query_vector = [float(x) for x in query_features.tolist()]
    body = {
        "query": {
            "script_score": {
                "query": {"match_all": {}},
                "script": {
                    "source": """
                        double score = cosineSimilarity(params.query_vector, 'features') + 1.0;
                        if (score > 1.9) { return 0; }
                        return score;
                    """,
                    "params": {"query_vector": query_vector}
                }
            }
        },
        "size": num_results,
        "min_score": 0.1
    }
    try:
        response = es.search(index="image_features", body=body)
        return response["hits"]["hits"]
    except Exception as e:
        print(f"Error during search: {str(e)}")
        return []
```

Пошук в ElasticSearch

```python
def search_similar_images_mongo_preprocess(features, num_results=3):
    ids = get_all_ids_mongo()
    similarities = []

    for id in ids:
        document = get_document_by_id_mongo(id)
        if document:
            features_mongo = np.array(document["features"])
            similarity = compare_images(features, features_mongo)
            if similarity <= 0.9:
                similarities.append((similarity, id))

    similarities.sort(reverse=True, key=lambda x: x[0])
    top_similarities = similarities[:num_results]

    results = [get_document_by_id_mongo(sim_id) for _, sim_id in top_similarities]
    return results
```

Пошук в MongoDB

11

Планування дослідження

Експерименти:

1. Зберігання зображень та обчислення ознак в MongoDB під час пошуку
2. Зберігання зображень та попередньо обчислених ознак у MongoDB
3. Зберігання зображень у MongoDB та ознак в ElasticSearch

Набір даних:

Дата-сет з 5400 зображень 90 видів різних тварин з Kaggle

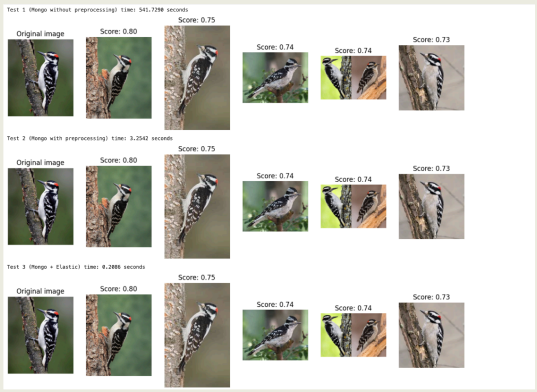
Обсяги даних:

10, 20, 50, 100, 500, 1000, 5400

12

Проведення дослідження

1000 зображень



5400 зображень

13

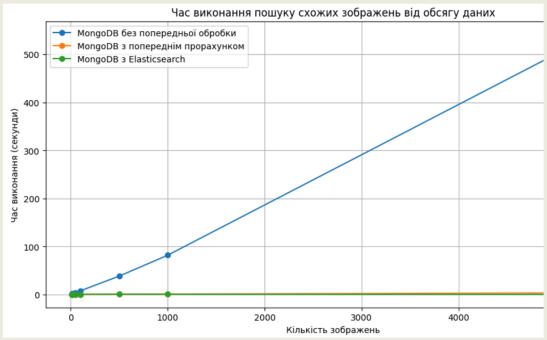
Результати дослідження

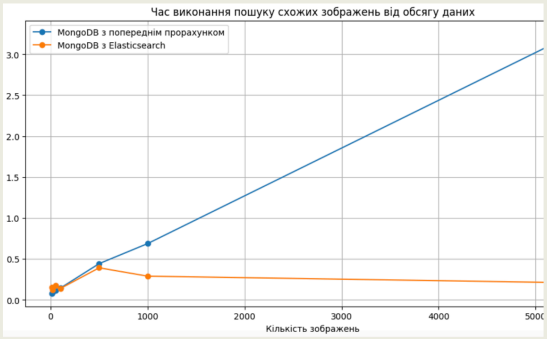
Експеримент	10	20	50	100	500	1000	5400
MongoDB	0.881	1.591	4.008	7.54	38.126	81.656	541.729
MongoDB з попереднім прорахунком	0.08	0.095	0.113	0.143	0.442	0.687	3.254
MongoDB з ElasticSearch	0.156	0.120	0.173	0.138	0.391	0.289	0.208

14

Аналіз результатів

- MongoDB – O(n)
- MongoDB with preprocessing – O(n)
- MongoDB + ElasticSearch – O(log(n))





15

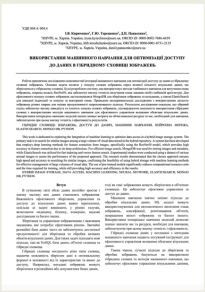
Висновки

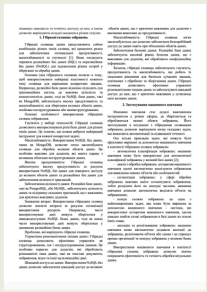
- Проведено аналіз предметної області
- Розглянуто можливості інтеграції машинного навчання для доступу до даних в гібридних сховищах зображення
- Поставлено задачу реалізувати алгоритм пошуку схожих зображень в сховищі
- Обрано модель машинного навчання - ResNet50
- Обрано використання комбінації MongoDB з ElasticSearch в якості гібридного сховища зображення
- Спроектовано архітектуру програмного рішення
- Реалізовано програмну частину дослідження
- Визначено і проведено експерименти
- Отримано і проаналізовано результати
- Використання моделі ResNet50 з гібридним сховищем даних у вигляді комбінації MongoDb та ElasticSearch для рішення задачі пошуку схожих зображень – досить ефективне

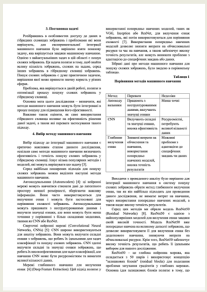
16

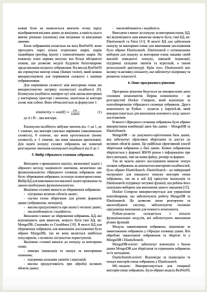
Апробація результатів дослідження

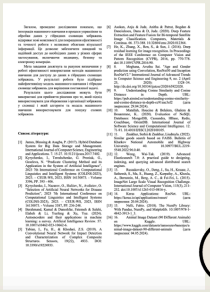
Результати даної кваліфікаційної роботи було опубліковано в журналі «Біоніка інтелекту», який включено до Переліку наукових фахових видань України, категорія «Б», технічні науки (затверджено наказом МОНУ від 02.07.2020 № 886)











17

ДЕМОНСТРАЦІЯ

18

ДЯКУЮ ЗА УВАГУ!

19

ДОДАТОК Е

Експертний висновок результатів перевірки кваліфікаційної роботи на відповідність оформлення «Вимоги ДСТУ 3008:2015»

Експертний висновок результатів перевірки кваліфікаційної роботи

студент (посада) | програмної інженерії (кафедра) | ІПЗм-22-3 (група)

Панасенко Д.П.
(прізвище, ім'я, по батькові)

Зауваження

Пункт ДСТУ 3008-2015	Зміст пункту	Сторінка кваліфікаційної роботи
1	2	3
	7.1 Загальні положення	
	7.3 Нумерація сторінок звіту	
	7.4 Нумерація розділів, підрозділів, пунктів, підпунктів	
	7.5 Рисунки	
	7.6 Таблиці	
	7.7 Переліки	
	7.8 Примітки	
	7.9 Виноски	
	7.10 Формули та рівняння	
	7.11 Посилання	
	7.13 Список авторів	
	7.14 Скорочення та умовні познаки	
	7.15 Додатки	
Методичні вказівки до виконання кваліфікаційної роботи магістра… **ЗАТВЕРДЖЕНО** кафедрою ПІ протокол № 5 від 13.11.2023р. 3.2 Оформлення пояснювальної записки згідно з ДСТУ 3008:2015 Звіти у сфері науки і техніки. Структура та правила оформлювання. **Шаблон** затверджений засіданням кафедри №3 від 16.10.2023.	Назву таблиці друкують з великої літери і розміщують над таблицею з абзацного відступу та в круглих дужках вказується джерело з якого взята ця рисунокаблиця. ПРИКЛАД: шаблон, стор.15	17, далі за текстом

Експерт _____ (підпис) Вадим НЕЧВОЛОД (прізвище, ініціали)

08.06.2024