МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

ХЕРСОНСЬКИЙ ДЕРЖАВНИЙ УНІВЕРСИТЕТ

Факультет комп'ютерних наук, фізики та математики

Кафедра комп'ютерних наук та програмної інженерії

Застосування машинного навчання для вдосконалення цифрових маркетингових стратегій

Кваліфікаційна робота (проєкт)

на здобуття ступеня вищої освіти «бакалавр»

Виконав: здобувач 4 курсу, 12- 461 групи
Спеціальності
126 Інформаційні системи та технології
Освітньо-професійної програми
першого (бакалаврського) рівня вищої освіти
Яковлев Владислав Миколайович
Керівник: доктор економічних наук, професор
Кобець Віталій Миколайович

Рецензент: Forex Tester, Junior react developer
Скрипка Катерина Олександрівна

Херсон – Івано-Франківськ – 2024

ЗМІСТ

ВСТУП

Актуальність. Новітні цифрові технології дають можливості для покращення процесів розробки та введення маркетингових стратегій, але застосування алгоритмів машинного навчання являється найбільш перспективним та найефективнішим напрямом маркетингової стратегії . Даний підхід вирішує актуальні завдання, пов'язані з прогнозуванням ринкових напрямів, моделями споживацької поведінки та ключовими елементів маркетингової стратегії.

Актуальність також полягає в поширеність використання згенерованих зображень в маркетингу. Розширення можливостей штучного інтелекту для зменшення витрат на маркетинг, вдосконаленні цифрових маркетингових стратегій.

Постановка проблеми. Порівняння зображень в інтернеті стає все більш складним завданням через інтенсивний розвиток інтернет-технологій та збільшення обсягу доступних зображень. Існують десятки сайтів, таких як i2img ai image variations, visme text-to-image generator і sentisight.ai, які пропонують різні інструменти для зрівняння та оптимізації зображень.

точність зрівняння може бути недостатньою, що призводить до неточних результатів. Деякі інструменти можуть не ефективно знаходити схожі зображення, особливо коли вони мають незначні візуальні або контекстуальні відмінності.

Існує потреба у вдосконаленні алгоритмів та технологій, щоб забезпечити більш точне та повне зрівняння зображень на сайтах.

Мета дослідження.

Дослідити різні аспекти цих сервісів, такі як їх можливості, точність результатів, швидкість обробки.

Порівняти їх переваги та недоліки, можливості і обмеження.

Для проведення дослідження можемо використовувати як кількісні, так і якісні методи дослідження. Порівнювати результати обробки зображень на

цих сайтах за допомогою метрик якості, таких як PSNR (Peak Signal-to-Noise Ratio) або SSIM (Structural Similarity Index).

Для порівняння зображень в інтернеті їіснують різні сайти. Також буде розглянута їх робота.

Розробити свій алгоритм для програми порівняння зображень.

Структура роботи: вступ, три розділи, висновки, список використаних джерел.

РОЗДІЛ 1. ОГЛЯД АЛГОРИТМІВ ТА ІНСТРУМЕНТІВ ДЛЯ ПОРІВНЯННЯ ЗОБРАЖЕНЬ

1.1. Алгоритми порівняння зображень

Сучасний інтернет став величезною інформацією площадкою для людей зі всього світу і без відповідної модерації він не може відповідати вимогам користувачів і давати змогу легко орієнтовуватись в цьому безладі. Для структурування і подачі інформації було винайденно безліч різних алгоритмів, які є невід'ємною частиною повсякденного життя будь-якого користувача, більшість з яких він навіть не помічає чи не придає їм достатньої ваги. Однією з таких речей є система пошуку одинакових зображень - це всього лише один з алгоритмів, але він захищає авторські права користувачів, чисте від дублікатів і використовується в візуальному пошуку і не тільки, він також використовується в більш користувацькому сегменті, а саме:

- Фільтрація особистих баз данних
- Комп'юторний зір (ідентифікація об'єктів: руки, людина, лице тощо)
- Медицина (для виявлення захворювань за допомогою штучного інтелекту

Існує декілька основних способів використання алгоритмів пошуку дублікатів:

1) порівняння бітів або бітових потоків,

2) аналіз текстури,

3) хеш-код,

3) нейронні мережі%

4) пошук ключових точок/особливостей.

У цій роботі буде використовуватись пошук ключових точок - це відносно швидкий і точний спосіб який є меньш уразливим до масштабування, блюра і деяких інших речей які змогли би обманути інші алгоритми. Він добре підходи для простого захисту від схожих між собою дублікатів. Тепер про алгоритми - найвідомішими і затребуваними алгоритмами по пошуку ключових особливостей є SIFT (Scale-Invariant Feature Transform) и SURF (Speeded Up Robust Features). SIFT - є основним алгоритмом. SURF був розроблений для більшої швидкодії, але меньшої ефективності. У цій роботі буде використано SIFT, оскільки необхідності в швидкодії зараз немає. Тепер, BFMatcher (Brute Force Matcher) використовує простий метод порівняння декскрипторів, які виходять після роботи алгоритму визначення ключових точок SIFT в якості матеріалу для порівняння. Бібліотекою для роботи з зображеннями була вибрана opencv за її інтуїтивність і універсальність. І для інтерфейсу программу була використана бібліотека FLTK.

1.2. Інструменти для порівняння зображень

Для порівняння зображень в інтернеті їіснують різні сайти. У теорітичній частині преддипломної практики буде розглянуто декільтка з них.

1)i2IMG AI Image Variations

2) Visme Text-to-Image Generator

3) SentiSight.ai

1) i2IMG AI Image Variations — це потужний інструмент для генерації зображень, що використовує штучний інтелект, що дозволяє створювати реалістичні та різноманітні варіації існуючих зображень. З його допомогою можна легко редагувати фотографії, створювати нові композиції і навіть генерувати абсолютно нові зображення.

Основні можливості i2IMG AI Image Variations:

• Генерація варіацій зображень: При завантаженні зображення, i2IMG AI Image Variations створить безліч варіацій цього зображення, кожна з яких матиме унікальний стиль, палітру кольорів і композицію.

• Редагування зображень: Ви можете використовувати i2IMG AI Image Variations для редагування існуючих зображень. Для цього необхідно завантажити зображення, вибирати потрібні параметри та натиснути кнопку «Створити».

• Створення нових композицій: Якщо потрібно створити абсолютно нове зображення, можна використовувати i2IMG AI Image Variations як інструмент для створення композицій. Для цього необхідно завантажити кілька зображень, вибрати потрібні параметри та натиснути кнопку «Створити».

• Генерація зображень з тексту: Можна використовувати i2IMG AI Image Variations для створення зображень з текстових описів. Потрібно ввести опис того, що потрібно побачити, і i2IMG AI Image Variations створить зображення, яке відповідає опису.

1.3. Переваги використання i2IMG AI Image Variations

• Простота використання: i2IMG AI Image Variations дуже проста у використанні. Користувачу не потрібно мати якісь спеціальні навички або знання, щоб створювати зображення.

• Швидкість: i2IMG AI Image Variations працює дуже швидко. Можна створити безліч варіацій зображення або навіть зовсім нове зображення лише за кілька секунд.

• Якість: Зображення, створені за допомогою i2IMG AI Image Variations, відрізняються високою якістю та реалістичністю.

• Універсальність: i2IMG AI Image Variations можна використовувати для створення зображень для різних цілей, включаючи маркетинг, дизайн, освіту та розваги.

Приклади використання i2IMG AI Image Variations:

• Створення варіацій продукту для інтернет-магазину: i2IMG AI Image Variations можна використовувати для створення варіацій зображень продукту, щоб показати його з різних сторін і в різних умовах.

• Створення обкладинок для книг та журналів: Функціонал i2IMG AI Image Variations підходить для створення унікальних та привабливих обкладинок для книг та журналів.

• Створення ілюстрацій для статей та блогів: Є можливість використовувати i2IMG AI Image Variations для створення ілюстрацій, які допоможуть привернути увагу читачів та зробити статті та блоги цікавішими. [1]

Висновок:

i2IMG AI Image Variations – це потужний та універсальний інструмент для генерації зображень, який може використовуватись для різних цілей. Він простий у використанні, працює швидко та створює зображення високої якості. Якщо користувачеві необхідний інструмент, який зможе створювати унікальні та привабливі зображення, i2IMG AI Image Variations – відповідний варіант.

2)Visme Text-to-Image Generator – це потужний інструмент для створення зображень з текстових описів. Він використовує штучний інтелект для створення реалістичних та високоякісних зображень на основі текстових підказок користувача.

1.4 Основні можливості Visme Text-to-Image Generator

• Генерація зображень із тексту: Після введення опису зображення Visme Text-to-Image Generator створить зображення, що відповідає даному опису.

• Створення варіацій зображень: Є можливість створювати безліч варіацій того самого зображення, просто змінюючи текстовий опис.

• Редагування зображень: В Visme Text-to-Image Generator можна редагувати створені зображення за допомогою вбудованого редактора зображень Visme.

• Завантаження та використання зображень: Даний генератор зображень дозволяє завантажувати створені зображення у різних форматах, включаючи JPG, PNG та SVG. Ви також можете використовувати зображення безпосередньо у проектах Visme.

Переваги використання Visme Text-to-Image Generator:

• Простота використання: Visme Text-to-Image Generator дуже простий у використанні. Користувачеві не потрібно мати будь-які спеціальні навички або знання, щоб створювати приголомшливі зображення.

• Швидкість: Visme Text-to-Image Generator працює дуже швидко. Він здатний створити зображення з тексту лише за кілька секунд.

• Якість: Зображення, створені за допомогою Visme Text-to-Image Generator, відрізняються високою якістю та реалістичністю.

• Універсальність: Visme Text-to-Image Generator можна використовувати для створення зображень для різних цілей, включаючи маркетинг, дизайн, освіту та розваги. [2]

1.5 Приклади використання Visme Text-to-Image Generator

• Створення зображень для соціальних мереж: Visme Text-to-Image Generator можна використовувати для створення зображень, які допоможуть користувачеві виділитися в соціальних мережах та залучити більше передплатників. [3]

• Створення зображень для презентацій: Є варіант використання Visme Text-to-Image Generator для створення зображень, які здатні зробити презентації більш привабливими та незабутніми.

• Створення зображень для дизайну інтер'єру: Особливості Visme Text-to-Image Generator можна використовувати для створення зображень, які допоможуть візуалізувати дизайн інтер'єру та зробити його більш гармонійним.

Висновок:

Visme Text-to-Image Generator – це потужний та універсальний інструмент для генерації зображень із тексту. Він працює швидко та створює зображення високої якості. Якщо необхідний інструмент, який допоможе створювати унікальні та деталізовані зображення, Visme Text-to-Image Generator підходить для цього завдання. [4]

Преимущества генераторов изображений AI

Генератори зображень на основі штучного інтелекту пропонують багато переваг, які можуть революціонізувати спосіб створення візуального контенту компаніями.

Ось кілька ключових переваг:

1.Економія коштів і часу: Традиційні методи створення зображень часто передбачають наймання професійних графічних дизайнерів і очікування, поки вони створять потрібні візуальні ефекти. Цей процес може бути трудомістким і дорогим. З іншого боку, генератори зображень зі штучним інтелектом можуть створювати високоякісні зображення за короткий час і зі значно меншими витратами. Ця ефективність може змінити правила гри для бізнесу, особливо для стартапів та малого бізнесу з обмеженими ресурсами.

2.Нескінченна творчість та персоналізація: Генератори зображень зі штучним інтелектом можуть створювати майже нескінченну кількість унікальних зображень на основі запитів користувача. Це забезпечує високий ступінь налаштування, дозволяючи компаніям створювати власні візуальні

ефекти, які ідеально відповідають ідентичності їхнього бренду та маркетинговим цілям. Крім того, здатність ШІ генерувати зображення з текстових описів відкриває цілий світ творчих можливостей, яких було б важко досягти звичайними методами дизайну.

3.Масштабованість: У міру зростання бізнесу потреба у візуальному контенті може зростати в геометричній прогресії. Генератори зображень зі штучним інтелектом можуть легко масштабуватися відповідно до цих вимог, генеруючи тисячі зображень за час, який знадобився б дизайнеру, щоб створити кілька. Ця масштабованість може бути особливо корисною для підприємств електронної комерції, яким потрібно створювати зображення товарів для великих запасів.

4.Узгодженість: Підтримка єдиної візуальної ідентичності у всіх маркетингових каналах може бути складним завданням, особливо для компаній, які виробляють великий обсяг контенту. Генератори зображень зі штучним інтелектом можуть допомогти забезпечити узгодженість, застосовуючи однакові параметри дизайну до всіх зображень. Це може допомогти зміцнити ідентичність бренду та зробити маркетингові матеріали більш впізнаваними для споживачів.

5.Доступність і простота використання: Однією з найбільших переваг генераторів зображень зі штучним інтелектом є їх доступність. Ці інструменти розроблені таким чином, щоб бути зручними у використанні, не вимагаючи навичок дизайну чи технічних знань. Це означає, що будь-хто в компанії, від відділу маркетингу до генерального директора, може створювати зображення за потреби. Така демократизація дизайну може сприяти творчості та інноваціям у всій організації.

Варіанти використання в різних галузях Генератори зображень зі штучним інтелектом мають широкий спектр застосування в різних галузях.

Ось кілька яскравих прикладів:

1.Маркетинг: У галузі, де візуальна привабливість має першорядне значення, генератори зображень на основі штучного інтелекту можуть змінити правила гри. Їх можна використовувати для створення унікальних візуальних ефектів для публікацій у соціальних мережах, цифрової реклами та маркетингових матеріалів. Можливість швидкого створення великої кількості зображень також робить A/B-тестування більш можливим, дозволяючи маркетологам оптимізувати свої кампанії на основі реакції споживачів.

2.Електронна комерція: Для інтернет-магазинів зображення товарів відіграють вирішальну роль у впливі споживачів на рішення про покупку. Генератори зображень зі штучним інтелектом можуть створювати високоякісні зображення продуктів, включаючи різні ракурси та варіації, без необхідності фізичних фотосесій. Це може заощадити час і ресурси, особливо для підприємств з великими каталогами товарів.

3.Розваги та медіа: У світі розваг генератори зображень зі штучним інтелектом можна використовувати для створення концепт-артів, дизайну персонажів і навіть цілих сцен. Це може спростити творчий процес для кінематографістів, геймдизайнерів та інших творчих людей, дозволяючи їм швидко та ефективно візуалізувати свої ідеї.

4. Здоров'я: У галузі охорони здоров'я генератори зображень на основі штучного інтелекту можна використовувати для створення медичних ілюстрацій і візуалізації даних. Це може допомогти в навчанні пацієнтів, дослідженнях і діагностиці. Наприклад, штучний інтелект може генерувати зображення, що ілюструють прогресування хвороби або наслідки певного лікування.недвижимость:

5.У сфері нерухомості генератори зображень на основі штучного інтелекту можна використовувати для віртуальної постановки, візуалізації нерухомості та навіть для планування та проектування нових будівельних проектів. Це може допомогти потенційним покупцям або інвесторам візуалізувати нерухомість, підвищуючи їх розуміння та інтерес.

Це лише кілька прикладів того, як генератори зображень зі штучним інтелектом можуть бути застосовані в різних галузях. Оскільки технологія продовжує розвиватися, ми можемо очікувати, що з'явиться ще більше інноваційних додатків. [5]

Однак існуючий функціонал затребуваний для відділів маркетингу багатьох компаній. На даний момент вже є великі компанії, які почали впроваджувати в маркетинг нейромережі, що працюють із зображеннями.

Согласно пресс-релизу, Coca-Cola является первым маркетологом, который воспользовался преимуществами нового партнерства между консалтинговой компанией Bain & Company и OpenAI, разработчиком программного обеспечения для искусственного интеллекта (ИИ), такого как ChatGPT

Щоб посилити свій маркетинг, Coca-Cola використовуватиме такі інструменти, як ChatGPT та DALL. E (цифровий генератор зображень на основі глибокого навчання). Потенційні варіанти використання для маркетологів бренду, визначені в оголошенні, включають більш персоналізовану рекламу та цільові повідомлення.

Зак Касс, керівник відділу продуктів OpenAI, заявив у своїй заяві, що стратегія штучного інтелекту Coca-Cola є «найамбітнішою», яку його фірма коли-небудь бачила від компанії споживчих товарів. Цього року штучний інтелект швидко став однією з найпопулярніших технологій у маркетингу, хоча питання етики та прийняття залишаються.

Домінування штучного інтелекту як провідної сфери інвестицій продовжується завдяки угоді між Bain та OpenAI. Coca-Cola є одним із найвідоміших маркетологів споживчих товарів у світі, а це означає, що раннє впровадження рішень OpenAI може ще більше прискорити тенденцію, створивши практичні приклади використання перспективної технології в маркетингу. [6]

Зображення мають велике значення в маркетингу, оскільки вони привертають увагу, створюють емоційний зв'язок та можуть ефективно комунікувати повідомлення вашого бренду. Використання генераторів зображень у маркетингу дозволяє створювати високоякісний та привабливий контент швидко та ефективно. Нижче наведено конкретні приклади того, як генератори зображень можна використовувати в маркетингу та соціальних мережах. Мереж.

1. **Створення Графічних Постів для Соціальних Медіа:**

   Генератори зображень можуть бути використані для створення графічних постів для публікацій в соціальних медіа. Це може включати цитати, пропозиції, інформацію про продукт або послугу, ілюстрації та інше.

2. **Створення Інфографіки:**

   Інфографіка є ефективним способом візуалізації складної інформації. Генератори зображень дозволяють швидко створювати інфографіку, яка може бути використана для пояснення процесів, статистики, порівняння продуктів і т.д.

3. **Створення Банерів та Рекламних Зображень:**

   За допомогою генераторів зображень можна створювати банери та рекламні зображення для використання на веб-сайтах, у рекламних кампаніях, брошурах та інших маркетингових матеріалах.

4. **Створення Презентацій та Матеріалів для Контент-Маркетингу:**

   Генератори зображень можуть бути використані для створення презентацій, інфографіки та інших матеріалів для контент-маркетингу. Це може включати створення слайдів, ілюстрацій до блог-постів, діаграм тощо.

5. **Створення Візуальних Елементів для Електронної Комерції:**

   В електронній комерції важливо мати привабливі зображення продуктів. Генератори зображень можуть допомагати створювати фотографії товарів, банери з пропозиціями, графічні кнопки тощо.

6. **Створення Візуальних Елементів для Брендування:**

Генератори зображень можуть бути використані для створення візуальних елементів для брендування, таких як логотипи, брендовані фони, шаблони для соціальних медіа тощо.

Загалом, використання генераторів зображень в маркетингу дозволяє ефективно створювати високоякісний візуальний контент, який привертає увагу аудиторії та сприяє підвищенню впізнаваності бренду.

РОЗДІЛ 2. АЛГОРИТМ ПОРІВНЯННЯ ЗОБРАЖЕНЬ

Необхідно зазначити, що даний алгоритм не може виявити схожі зображення, згенеровані штучним інтелектом , даний алгоритм не має такого функціоналу.

2.1. Алгоритм порівняння зображень

На підставі вивчених даних було розроблено свій алгоритм для програми порівняння зображень.

Перш за все алгоритм починається з виявлення ключових точок який розбиває зображення на велику збірку векторних ознак які інваріантні до паралельного перенесення, масштабування і обертання зображення. Також, частково інваріантні до змін освітлення і локальних геометричних спотворень. Потім йде індексація (створення дескрипторів) на основі ключових особливостей зображення. Дескриптор - це 128-байтове числове значення яке використовується для подання конкретного зображення і подальшого їх порівняння. Потім BFMatcher використовує брутфорс-підхід для порівняння описів ключових точок. Це означає, що він просто перебирає всі можливі комбінації для знаходження найбільш схожих описів. Він може використовувати різні метрики для визначення ступінні схожості із найвідоміших L1 (сума абсолютних різниць між відповідними елементами двох векторів) і L2 (евклідова відстань між двома точками у просторі). L2 вважається стандартом, тому в цій роботі використовується саме евклідова

відстань. В підсумку, найкращі збіги визначаються на основі порогового значення відстані. Тобто, якщи відстань між описами менше порогу, то точки вважаються збігаючимися. В проекті використовується 0.02 в якості порогу відстанні.

2.2. Розбивка вихідного коду

```
#define WINDOW_WIDTH 640
#define WINDOW_HEIGHT 480

#define BUTTON_WIDTH 140
#define BUTTON_HEIGHT 50
#define BUTTON_X WINDOW_WIDTH/2 - BUTTON_WIDTH/2
#define BUTTON_Y 10

#define DISPLAY_WIDTH WINDOW_WIDTH - 20
#define DISPLAY_HEIGHT WINDOW_HEIGHT - BUTTON_HEIGHT - 30
#define DISPLAY_X 10
#define DISPLAY_Y BUTTON_Y + BUTTON_HEIGHT + 10
```

Рис. 2.1. Розбивка вихідного коду

Код починається з визначенням макросів для користувальницького інтерфейсу программу. Це може бути корисним, щоб зручно змінювати їх налаштування.

Потім йде підключення заголовочних файлів які потрібні для проекту. А саме:

```
#include <opencv2/opencv.hpp>
```

Основний заголовочний файл, який відповідає за підключення усіх інших хедерів відповідних за обробку зображень. А саме:

feature2d.hpp - відповідальний за обробку ключових точок на зображенні та їх порівнянні. Найважливіші для цієї теми речі, які в ньому цікавлять: cv::Feature2D - базовий клас для всіх двовимірних ознак. Він містить методи для обчислення та порівняння ознак.

cv::SURF - клас для обчислення ознак SURF (Speeded Up Robust Features). Він є потомком SIFT і реалізує нереалізовані методи абстрактного класса Feature2D, які программіст використовує в своєму проекті.

cv::SIFT - класс для обчислення ознак SIFT (Scale-Invariant Feature Transform). Так, як і попередній - він реалізує нереалізовані методи абстрактного класса Feature2D і є його потомом. Саме він використовується в цій роботі.

cv::FeatureDetector - тип данних, або огортка над Feature2D, яка використовується для легкої заміни технології (наприклад, щоб змінити SIFT на SURF). Але в цьому проекті використовується клас SIFT без огортки.

cv::FlannBasedMatcher - клас для пошуку відповідностей між ключовими точками з використанням алгоритму FLANN. Один із можливих варіантів і береться з окремого модуля flann (Fast Library for Approximate Nearest Neighbors).

cv::BFMatcher - клас для пошуку відповідностей між ключовими точками з використанням брутфорс-підхіду. Самі він порівнює дескриптори в цьому проекті.
І багато іньшого.

imgproc.hpp - містить функції для обробки зображень. Цей файл містить велику кількість функцій для різних операцій зображення, таких як фільтрація, геометрія зображення, колірні простори, обробка контурів, обробка текстур та багато іншого. Деякі функції потрібні для роботи з особливостями зображення тримають своє начало відсюди.

imgcodecs.hpp - містить інтерфейси для роботи з різними форматами зображень. Він містить функції для читання та запису зображень у різних форматах, таких як JPEG, PNG, BMP, GIF, TIFF та інших.

imgcodecs.hpp надає наступні функції для роботи з зображеннями:

cv::imread(): функція для читання зображення з файлу.

cv::imwrite(): функція для запису зображення у файл.

2.3. Команди для заголовочних файлів

Всі команди для заголовочних файлів opencv визначені на рис. 2.2.

```
#include <FL/Fl.H>
#include <FL/Fl_Window.H>
#include <FL/Fl_Button.H>
#include <FL/Fl_Text_Display.H>
#include <FL/Fl_Native_File_Chooser.H>
```

Рис. 2.2. команди для заголовочних файлів

Fl.H - головний модуль без якого не обходится не одна програма з UI FLTK. Тримає в собі купу різних корисних функцій головна з яких run() - яка використовується для головоного циклу.

Fl_Window.H - модуль за допомогою якого створюється головне вікно програми.

Fl_Button.H - використовується для кнопок та роботи з ними.

Fl_Text_Display.H - пресдставляє з себе текстовий дісплей який подібно терміналу може виводити інформацію для користувача. В цьому випадку основне джерело інформації о співпадіннях та інших речах.

Fl_Native_File_Chooser.H - відповідає за діалогове вікно вибору файли або папки. В цьому випадку папки. Адаптується до операційної системи або дистрибутива, коли Fl_File_Chooser.H - використовує внутрішнє діалогове вікно FLTK.

Розглянемо стандартні бібліотеки мови C++ (рис. 2.3)

```
#include <filesystem>
#include <vector>
#include <cstring>
```

Рис. 2.3. Стандартні бібліотеки мови C++

Опишемо ці бібліотеки:

filesystem - надає функціональність для роботи з файловою системою. Він містить класи та функції для створення, читання, запису, переміщення та видалення файлів та директорій. Основними класами, що визначені в заголовочному файлі filesystem, є path, directory_entry, directory_iterator, file_status, perms та space_info.

vector - є частоною стандартних шаблоних бібліотек STL (Standard Template Library) і динамічною колекцією елементів, які можуть змінюватися в розмірі під час виконання програми. Клас vector дозволяє зберігати елементи будь-якого типу, включаючи власний клас, і надає різні методи для додавання, видалення та доступу до елементів колекції. push_back(element): додає елемент в кінець колекції, pop_back(): видаляє останній елемент колекції, size(): повертає кількість елементів в колекції, clear(): очищує колекцію, видаляючи всі елементи, operator[](index): дозволяє отримати доступ до елемента колекції за індексом та інші... vector - це сучасна заміна динамічним масивам.

cstring - надає функції для роботи з рядами символів (строками). В цьому коді використовуються: string - як більш зручна і сучасна заміна массиву символів і strcmp() - який порівнює два рядки. Загалом, додає функціональність що дозволяє легко маніпулювати та зчитувати рядки символів різного типу.

```
namespace fs = std::filesystem;
using namespace std;
using namespace cv;
```

Підключення та визначення namespace-ів. namespace fs = std::filesystem; визначає коротке ім'я fs для стандартного простору імен std::filesystem. Цей простір імен містить функції та класи для роботи з файловою системою, такими як створення, читання та запис файлів. using namespace std; вказує на те, що всі імена зі стандартного простору імен std можна використовувати без повної нотації, наприклад, std::cout можна записати просто як cout. using namespace cv; Цей простір імен містить класи та функції для роботи з комп'ютерним зору, такими як обробка зображень та відео.

```
Ptr<SIFT> detector = SIFT::create();
```

Тепер реалізація. Створює об'єкт типу SIFT, який є одніим з алгоритмів виявлення ознак зображень. Структура Ptr є вказівник на об'єкт типу SIFT, який створюється за допомогою методу create() класу SIFT. Цей метод створює новий об'єкт типу SIFT з деякими початковими параметрами, такими

як роздільна здатність, рівень розрізності точок та інші. Відмінність ptr від інших вказівників в тому, що він сам управляє пам’яттю і видаляє її після використання без участі програміста.

```
BFMatcher matcher(NORM_L2);
```

Створює об'єкт типу BFMatcher, який є одним з алгоритмів порівняння описів ключових точок зображень. Структура BFMatcher створюється з використанням конструктора, який приймає два аргументи: перший - це тип нормалізації, який використовується для порівняння описів ключових точок, а другий - це порогове значення, яке використовується для відкидання невідповідних відповідних пар. Використовується тип нормалізації NORM_L2, який є одним з найпоширеніших типів нормалізації. Він використовує евклідову відстань між двома векторами описів ключових точок для порівняння.

```
vector<vector<KeyPoint>> keypoints(files.size());
```

Це вектор векторів ключових точок (або особливостей) зображень. Це було необхідно, щоб для каждного окремого зображення був свій вектор ключових точок. Якщо детальніше, то KeyPoint - це структура даних, яка використовується в OpenCV для представлення ключових точок зображення. Вона містить координати ключової точки, її розмір та орієнтацію. Цей вектор беде використаний для подальшої обробки ключових точок, такої як виявлення відповідних пар ключових точок між двома зображеннями, побудова описів ключових точок та інше. Розміри задаються в відповідності з кількістю знайдених в папці файлів (про що далі).

```
vector<Mat> descriptors(files.size());
```

Вектор дескрипторів які використовуються для порівняння зображень. Власне, в інших випадках Mat використовувуеться для збереження і подальшої обробки зображень в памяті. Визначається кількостю файлів.

```
vector<Mat> images(files.size());
```
г

Вектор із зображеннями, які використовуються в порівнянні. Визначається кількосю файлів.

```
fs::path path_to_file = fs::path(path) / fs::path(files[i]);
```

Об’єднює шлях до кожного зображення з шляхом до папки.

```
images[i] = cv::imread(path_to_file.string(), IMREAD_GRAYSCALE);
```

Зчітує зображення з файлової системи. Функція imread() використовується для зчитування зображення з файлу, який вказаний у змінній path_to_file. Параметр IMREAD_GRAYSCALE вказує, що зображення має бути зчитане в градації сірого - оскільки технологія SIFT використовує контраст між пікселями в якості індикатора їй не потрібен RGB-кольор.

```
detector->detectAndCompute(images[i], noArray(), keypoints[i], descriptors[i]);
```

Ця строка коду використовує об'єкт detector для виявлення особливостей на зображенні images[i] та обчислення їх описувачів, які зберігаються в змінних keypoints[i] та descriptors[i] відповідно. Де:

Параметр noArray() вказує, що немає додаткової інформації, яка потрібна для виявлення особливостей. Інакше він потребує матрицю.

```
matcher.match(descriptors[i], descriptors[j], matches);
```

Для пошуку відповідностей між дескрипторами двох зображень. Використовує об'єкт matcher для пошуку відповідностей між описувачами особливостей двох зображень descriptors[i] та descriptors[j] та зберігає їх у змінній matches.

```
for (const DMatch& match : matches)
{
    double dist = match.distance;
    min_dist = min(dist, min_dist);
    max_dist = max(dist, max_dist);
}
```

Використовуються для обчислення мінімальної та максимальної відстані між відповідними особливостями двох зображень. Цикл for each використовує ітератор const DMatch& match для перебору всіх відповідностей, які були знайдені між описувачами особливостей двох зображень. Змінна dist використовується для зберігання відстані між відповідними особливостями двох зображень. Змінні min_dist та max_dist потрібні для зберігання мінімальної та максимальної відстані між відповідними особливостями двох зображень .Функція min та max використовуються для обчислення мінімальної та максимальної відстані між відповідними особливостями двох зображень відповідно.

```
for (const DMatch& match : matches)
{
    if (match.distance <= max(2 * min_dist, 0.02))
    {
        good_matches.push_back(match);
    }
}
```

Цей блок потрібен для відбору хороших відповідностей між описувачами особливостей двох зображень.Такий же ітератор як в попередній раз і перевірка, що відстань між відповідними особливостями двох зображень менша або дорівнює максимуму з двох значень: відстані, що дорівнює двом мінімальним відстаням, та 0.02 для визначення дублікатів.

good_matches - саме тут тепер зберігаються відповідності між особливостями у дублікатів.

```
similarity = good_matches.size() * 100 / max(keypoints[i].size(), keypoints[j].size());
```

Тут обчислюється відсоткові подібності між двома зображеннями на основі кількості хороших відповідностей між описувачами особливостей двох зображень. Значення 100 використовується для перетворення відсоткової подібності в діапазон від 0 до 100.

```
if (similarity >= 80 && similarity <= 101)
{
    files_info[i] += (" : " + files[j] + " Схожість: " + to_string((int)similarity) + "%");
    files_info[j] += (" : " + files[i] + " Схожість: " + to_string((int)similarity) + "%");
}
```

В цьому блокі виконується вичіслення відсоткової відповідності у теперешніх зображень і записується в тимчасовий буфер files_info. Тепер у кажної строки є вся інформація про те - на яке зображення воно схоже і те, наскільки сильна ця схожість. На цьому закінчується обчислення схожості і починається решта додаткового коду.

```
string _file;
string file_extension;
files.clear();
for(const fs::directory_entry entry : fs::directory_iterator(path))
{
    if(entry.is_regular_file())
    {
        _file = entry.path().filename().string();
        file_extension = strrchr(_file.c_str(), '.');
        if(strcmp(file_extension.c_str(), ".png") || strcmp(file_extension.c_str(), ".jpg")
        || strcmp(file_extension.c_str(), ".bmp") || strcmp(file_extension.c_str(), ".tiff"))
        {
            files.push_back(_file);
        }
    }
}
```

Використовується для зчитування файлів з певної директорії, які мають одне з доступних і підтримуванних в opencv розширень, а саме: .png, .jpg, .bmp, .tiff. Тут:

Змінна _file потрібна для того, щоб незалежно від теперешнього ітератора і тому подібно записувати значення в змінні вектора files в якому зберігаються імена усіх файлів в відстежуваній папці.

Змінна file_extension використовується для зберігання розширення файлу.

Функція files.clear() використовується для попереднього очищення списку, щоб запевнитись в тому, що попередні данні не заважають відображенню і використанню правильної інформації.

Цикл for використовує ітератор const fs::directory_entry entry для перебору всіх файлів та директорій в директорії, яка задається в змінній path. А саме directory_entry зберігає теперешній файл чи директорію, а fs::directory_iterator(path) усі файли в заданному змінною path шляху.

Функція is_regular_file() використовується для перевірки, чи є об'єкт файлом.

Функція path().filename().string() використовується для отримання назви файлу. string() для трансформування в string.

Функція strrchr() використовується для отримання розширення файлу. Вона шукає заданний символ і повертає усі символи після нього (в тому чіслі і сам символ), тому вказав точку вона повертає розширення файлу, оскільки у Windows усі файли мають своє розширення.

Функції strcmp() використовуються для порівняння розширення файлу з набором допустимих розширень.

Функція push_back() використовується для додавання файлу до списку файлів. В данному випадку змінну _file до вектору файлів.

```
        CheckNewImages();
        CheckDuplicates();
        _log->buffer()->text("");
        for(const string file : files)
        {
            _log->buffer()->append(file.c_str());
            _log->buffer()->append("\n");
        }
```

Тут знаходится виведення інформації про файли (а точніше зобрження в каталозі) та їх схожість. Змінна _log використовується для зберігання посилання на об'єкт, який містить текстовий дісплей в який буде виводитись різна інформація.

Тут функція CheckNewImages() - оновлює інформацію про зображення в папці і записує результат в вектор files (який розташованний в глобальній облаісті бачення, оскільки проект невеликий).

Функція CheckDuplocates() - оновлює інформацію про існуючі дублікати серед зображень у відстежуваній папці і записує її в вектор files. Детальний огляд цих функцій знаходится вище.

Функція buffer() використовується для отримання доступу до буфера текстового дісплею в який далі буде додаватись інформація.

Функція text("") використовується для очищення буфера текстового поля. Це потрібно, щоб запевнетись що буфер не заповнений і інформація не будет змішуватись. Навідмінну від append, він повністю перезапусує будь який текст в буфері.

Цикл for використовує ітератор const string file для перебору списку файлів.

Функція c_str() використовується для трансформування string в массив символів. Це потрібно тому, що FLTK не підтримує роботу зі строками.

Перша функція _log->append() використовується для виведення рядкового представлення файлу, а також його дублікати і степінь схожості на буфер текстового дісплею (який в свою чергу виводе всю информацію для користувача).

Другая функція _log->append() використовується ще раз для додавання нового рядка до буфера текстового поля, оскільки на відміну від наприклад, printf() (стандартної бібліотеки C) - ця функція не підтримує додавання декількох аргументів.

```
//Викликає сканування кожні 3 секунди
void TimeoutCallback(void* data)
{
    Fl_Text_Display* _log = static_cast<Fl_Text_Display*>(data);
    UpdateLog(_log);
    Fl::repeat_timeout(3.0, TimeoutCallback, _log);
}
```

Ця функція викликає сканування папки і оновлення інформації в текстовому дісплею кожні 3 секунди. Це функція зворотного виклику для методу із основного модуля FLTK Fl.H.

Fl_Text_Display* _log = static_cast<Fl_Text_Display*>(data); - трансформує безтипове вхідне значення функціх в екземпляр типа Fl_Text_Display для подальшої передавання його в функцію оновлення текстового дісплею.

static_cast<>() - один з шаблонів стандартних шаблонних бібліотек мови C++. Більш безпечний і сучасний варіант трасформування за допомогою дужок (наприклад, трансформуваннння в int за допомогою (int)variable, а в випадку з static_cast - static_cast<int>(variable)). Приймає в себе будь-яке значення і трансформує його в будь яке схоже значення.

UpdateLog() - Оновлює інформацію на текстовому дісплеї.

repeat_timeout() - намагається повторити виклик функції рівно через заданий час після попереднього виклику, навідміну від функції add_timout() - яка викликає функцію через заданий час після свого виклику. В данному випадку повторює ту саму функцію, яка його викликала, щоб зробити рекрсивний виклик функції. Перший аргумент вхідного значення являє собою значення float - через який час потрібно зробити виклик знову. Другий аргумент являє собою функцію яка повина викликатись після заданного часу. І третім аргументом передається додаткові данні, які можуть стати в нагоді якщо у функції немає доступу до цих змінних або змінної із своєї області бачення.

```
//Показує діалогове вікно вибору папки для сканування
void ChoiseDirectory(Fl_Widget* widget, void* data)
{
    Fl_Text_Display* _log = static_cast<Fl_Text_Display*>(data);
    Fl_Native_File_Chooser chooser(Fl_Native_File_Chooser::BROWSE_DIRECTORY);
    chooser.title("Виберіть папку...");
    if(chooser.show() == 0)
    {
        path = chooser.filename();
    }
    Fl::add_timeout(3.0, TimeoutCallback, _log);
}
```

Цей блок кода представляю з себе функцію зворотнього виклику яка викликається після натискання на кнопку. Власне строки коду використовуються для відображення текстового поля та відкриття діалогового вікна вибору директорії. Після вибору директорії, шлях до неї зберігається в змінній path та викликається функція TimeoutCallback для оновлення текстового поля. Змінна _log використовується для зберігання посилання на об'єкт, який містить текстове поле.

Функція static_cast<Fl_Text_Display*>(data) використовується для приведення безтипового посилання data яке є вхідним параметром до посилання на об'єкт класу Fl_Text_Display.

Об'єкт Fl_Native_File_Chooser chooser використовується для створення діалогового вікна вибору директорії. Його окнструктор ініціалізується параметром Fl_Native_File_Chooser::BROWSE_DIRECTORY із області

бачення Fl_Native_File_Chooser. Він означає, що тепер діалогові вікно буде вибирати саме папку, а не файл чи щось інше.

Функція chooser.title("Виберіть папку...") використовується для встановлення заголовка діалогового вікна і для деяких «інструкцій» користувачу.

Функція chooser.show() використовується для відображення діалогового вікна. Вона повертає число яке залежить від того чи закінчілась вона успіхом. 0 - якщо усе нормально, тому умова if(chooser.show() == 0) перевіряє, чи була вибрана директорія.

Змінна path використовується для зберігання шляху до вибраної директорії.

Функція Fl::add_timeout(3.0, TimeoutCallback, _log) використовується для встановлення таймауту в 3 секунди та виклику функції TimeoutCallback з параметром _log.

Функція TimeoutCallback використовується для оновлення текстового поля.

```
Fl_Window window(WINDOW_WIDTH, WINDOW_HEIGHT, "dubpro");
Fl_Button button(BUTTON_X, BUTTON_Y, BUTTON_WIDTH, BUTTON_HEIGHT, "відстежувати папку");
```

Fl_Window є базовим класом для створення вікон, які можуть містити різні елементи інтерфейсу, такі як кнопки, текстові поля, меню та інше. Тут він ініціалізує розміри макросами і називає себе «dubpro».

Fl_Button дозволяє створювати кнопки різних розмірів та форм, налаштовувати їх колір, шрифт, текст та інші властивості. Тут він ініціалізованний розмірами визначеними в макросах та підписанний «відстежувати папку».

```
//Ініціалізація текстового виводу
Fl_Text_Display log(DISPLAY_X, DISPLAY_Y, DISPLAY_WIDTH, DISPLAY_HEIGHT);
Fl_Text_Buffer buffer;
log.textfont(FL_COURIER);
log.textsize(14);
log.buffer(&buffer);
```

Fl_Text_Display дозволяє створювати текстові поля різних розмірів та форм, налаштовувати їх колір, шрифт, текст та інші властивості. Тут він також ініціалізований макросами.

Fl_Text_Buffer - це клас який відповідає за буфер в якому будут зберігатися данні текстового дісплею.

log.textfont(Fl_COURIER) - Встановлює шріфт Courier.

log.textsize(14) - встановлює розмір 14.

log.buffer(&buffer) - назначає буфер для Fl_Text_Display передаючі вказівник на раніше створений екземпляр классу Fl_Text_Buffer.

На цьому розбивка вихідного коду закінчується.

РОЗДІЛ 3 Демонстрація роботи алгоритму.

3.1 Результат роботи програми

Прикдлад використяння свого алгоритму порівняння зображень для створення фірмового логотипу компанії.

Логотип компанії NIKE.

Логотип згенерований на основі логотипу NIKE.

Висновок

Пошук схожих зображень - це перспективна і цікава технологія яка допомагає нам фільтрувати данні в повсякденному житті.

i2IMG AI Image Variations – це потужний та універсальний інструмент для генерації зображень, який може використовуватись для різних цілей. Він простий у використанні, працює швидко та створює зображення високої якості. Якщо користувачеві необхідний інструмент, який зможе створювати унікальні та привабливі зображення, i2IMG AI Image Variations – відповідний варіант.

Visme Text-to-Image Generator – це потужний та універсальний інструмент для генерації зображень із тексту. Він працює швидко та створює зображення високої якості. Якщо необхідний інструмент, який допоможе створювати унікальні та деталізовані зображення, Visme Text-to-Image Generator підходить для цього завдання.

Програми в порівнянні зображень мають широкий спектр застосувань та корисні у різних галузях, таких як безпека, медицина, наука та багато іншого. Вони дозволяють автоматизувати процес зіставлення та аналізу зображень, виявляти зміни, аномалії та підробки, що суттєво підвищує ефективність та точність роботи. Актуальність таких програм підтверджується зростаючою потребою в автоматизованих рішеннях для обробки та аналізу великих обсягів даних, а також підвищення рівня безпеки та якості продукції.

В підсумку, найкращі збіги визначаються на основі порогового значення відстані. Тобто, якщи відстань між описами менше порогу, то точки вважаються збігаючимися. В проекті використовується 0.02 в якості порогу відстанні.

Існування різних версій програм у порівнянні зображень також виправдане, оскільки вони дозволяють постійно удосконалювати функціональність, продуктивність і безпеку, а також адаптувати програми під вимоги, що змінюються, і стандарти галузі.

Список використаних джерел:

1.[1] https://www.victoria.lviv.ua/library/students/sss2021/theme7.html

2.[2]https://ekmair.ukma.edu.ua/bitstreams/84299379-158c-4529-a3d7-9f2013d9b6a1/download

3.[3]https://biz.nv.ua/ukr/experts/ai-zagroza-chi-mozhlivist-yaskravi-prikladi-vikoristannya-shi-v-media-50367319.html

4.[4] https://blog.liga.net/user/amyisenko/article/51613

5. Hemann, Chuck, and Ken Burbary. "Digital Marketing Analytics: Making Sense of Consumer Data in a Digital World." Que Publishing, 2018.

6. Blanchard, Olivier. "Social Media ROI: Managing and Measuring Social Media Efforts in Your Organization." Que Publishing, 2011.

7.Winston, Wayne L. "Marketing Analytics: Data-Driven Techniques with Microsoft Excel." Wiley, 2014.

8. [5] 26 декабря 2023 By Алекс МакФарланд. https://www.unite.ai/ru/why-every-company-should-use-ai-image-generators/

9. [6] https://mediana.by/rubriki/soobshchestvo/4543-coca-cola-ofitsialno-vnedryaet-chatgpt-i-generator-izobrazhenij-dalle-v-svoyu-rabotu.html

10. HubSpot Blog. (https://blog.hubspot.com/)

11. Kumar, Nishant. "Visual Content Marketing: Leveraging Infographics, Video, and Interactive Media to Attract and Engage Customers." Que Publishing, 2019. ISBN: 978-0789758849.

12.Zarrella, Dan. "The Science of Marketing: When to Tweet, What to Post, How to Blog, and Other Proven Strategies." Wiley, 2013. ISBN: 978-1118138274.

13.Weinberg, Martin. "Telling a Visual Story: Communicating with Infographics and Data Visualization." Wiley, 2016. ISBN: 978-1118792280.

14. Шевченко, Н. М., & Шкуренко, В. П. (2019). "Інтернет-маркетинг: сучасні стратегії та інструменти." Київ: Видавництво Київського університету. ISBN: 978-966-412-260-1.

15.Самойленко, Ю. В. (2016). "Маркетинг в соціальних мережах: інструментарій, методики, практика." Київ: Київський національний університет ім. Тараса Шевченка. ISBN: 978-966-02-8207-1.

16.Чуйко, А. О. (2019). "Цифровий маркетинг: теорія та практика." Київ: Видавництво Логос. ISBN: 978-617-7533-38-9.

17.Лисогорська, І. М. (2017). "Маркетингові дослідження в управлінні підприємством." Київ: Видавничий дім "Професіонал". ISBN: 978-617-7534-34-8.

18.Воронцов, С. В. (2018). "Маркетингові комунікації: стратегії та інструменти." Київ: Видавництво Логос. ISBN: 978-617-7533-09-9.

19. Шульга, О. В., & Дубенецька, О. М. (2016). "Маркетингові дослідження: підручник." Київ: Видавничий дім "Слово". ISBN: 978-617-7619-92-2.

20. Johnson, Mark. "Visual Marketing: 99 Proven Ways for Small Businesses to Market with Images and Design." Wiley, 2013. ISBN: 978-1118533159.

21.Charney, Tamar. "The Art of Visual Marketing: 75 Digital Marketing Tools for Visual Branding." Wiley, 2016. ISBN: 978-1119054153.